Open Source Security Systems

Защита сетей на основе систем с открытым кодом

Санкт-Петербург
2004

Оглавление

1 Введение....26
1.1 Основные различия между коммерческими и открытыми программами....26
1.1.1 Цели создания программ....26
1.1.2 Возможность адаптации программ....26
1.1.3 Уязвимость программ....27
1.1.3.1 Устранение уязвимостей....27
1.1.4 Оптимизация работы систем обеспечения безопасности....28
1.1.5 Ответственность разработчиков....28
1.1.6 Возможность разработки собственных средств....28
2 Безопасность хостов Linux....29
2.1 Инсталляция ОС, организация файловой системы, управление пользователями....29
2.1.1 Разбиение диска....29
2.1.1.1 fdisk....30
2.1.1.1.1 Создание области подкачки и корневого раздела....31
2.1.1.1.2 Создание расширенного раздела....31
2.1.1.1.2.1 Создание логических дисков в расширенном разделе....32
2.1.1.2 cfdisk....32
2.1.1.3 DiskDruid....32
2.1.2 Менеджер загрузки....33
2.1.2.1 Конфигурация lilo....33
2.2 Управление учетными записями....34
2.2.1 Структура учетных записей....34
2.2.2 Пользователи....35
2.2.2.1 Синтаксис....35
2.2.2.2 Добавление пользователей....35
2.2.2.3 Изменение принятых по умолчанию значений....36
2.2.2.4 Известные проблемы....36
2.2.2.5 Файлы....37
2.2.2.6 Удаление и редактирование учетных записей пользователей....37
2.2.3 Группы....37
2.2.3.1 Синтаксис и параметры....37
2.2.4 Теневые учетные записи....38
2.3 Управление доступом к файлам....39
2.3.1 Права доступа и принадлежность файлов....39
2.3.2 Права доступа, устанавливаемые по умолчанию....41
2.3.3 Специальные права доступа к файлам....42
2.4 Поддержка целостности системы....43
2.4.1 Враждебный код....43
2.4.1.1 Троянские программы....43
2.4.1.2 Вирусы....43
2.4.1.3 Сканеры....44
2.4.1.4 Анализаторы протоколов....45
2.5 Способы обеспечения безопасности при удаленном доступе к хостам Linux....45
2.6 Обеспечение безопасности служб Internet/intranet....45
2.6.1 Безопасность Web-серверов....45
2.6.2 Безопасность электронной почты....45
2.6.3 Безопасность служб FTP, удаленного доступа, Web-приложений....45
2.7 Средства проверки полномочий пользователей и управления доступом к сетевым ресурсам....45
2.8 Системные журналы Linux....45
2.8.1 Журналы регистрации пользователей....46
2.8.1.1 lastlog....46
2.8.1.2 last....47
2.8.1.3 Обход журналов регистрации пользователей....48
2.8.2 xferlog....48
2.8.3 Системный журнал messages....49
2.8.4 Настройка syslog....49
2.8.4.1 Типы сообщений....50
2.8.4.2 Уровни протоколирования....50
2.8.4.3 Назначение сообщений....51
3 Готовые решения на базе специализированных дистрибутивов Linux....52
3.1 MandrakeSecurity MNF....52
3.2 Trustix....52
3.3 EnGarde Secure Linux....53
3.4 Euronode....53
3.4.1 Euronode Simple Firewall....54
3.4.2 Euronode Advanced Firewall....54
3.4.3 FrazierWall Linux....54
3.5 Immunix....54
3.6 IPCop Firewall....55
3.7 NSA Security Enhanced Linux....55
3.8 OpenNA Linux....55
3.9 Openwall GNU/Linux....56
3.10 redWall....56

3.11 Securepoint Firewall & VPN....56
3.12 SmoothWall....57
3.13 Devil-Linux....57
4 Межсетевой экран на базе дистрибутива Linux общего назначения....58
4.1 Установка Linux....58
4.1.1 Разбиение дисков....58
4.1.1.1 Корневой раздел....58
4.1.1.2 Область подкачки (swap)....58
4.1.1.3 /usr....58
4.1.1.4 /var....59
4.1.1.5 /tmp....59
4.1.1.6 /home....59
4.1.2 Выбор программ для установки....59
4.1.3 Системные библиотеки....59
4.1.4 Системные утилиты....60
4.1.5 Сетевые службы....60
4.1.6 Дополнительные программы мониторинга, управления и т. п....60
4.1.7 Инструментальные средства....60
4.2 Пользовательские приложения....60
4.3 Подготовка к созданию ядра....60
4.3.1 Загрузка и подготовка исходных кодов....61
4.3.1.1 Установка расширений Netfilter....61
4.3.2 Команды настройки опций....61
4.4 Выбор опций ядра с учетом требований безопасности....62
4.4.1 Опции общего назначения....63
4.4.1.1 Меню Code maturity level options....63
4.4.1.1.1 EXPERIMENTAL....63
4.4.1.1.2 CLEAN_COMPILE....63
4.4.1.1.3 STANDALONE....63
4.4.1.2 Меню General setup....63
4.4.1.2.1 SWAP....63
4.4.1.2.2 SYSVIPC....63
4.4.1.2.3 POSIX_MQUEUE....64
4.4.1.2.4 BSD_PROCESS_ACCT....64
4.4.1.2.4.1 BSD_PROCESS_ACCT_V3....64
4.4.1.2.5 SYSCTL....64
4.4.1.2.6 AUDIT....64
4.4.1.2.6.1 AUDITSYSCALL....65
4.4.1.2.7 LOG_BUF_SHIFT....65
4.4.1.2.8 HOTPLUG....65
4.4.1.2.9 IKCONFIG....65
4.4.1.2.9.1 IKCONFIG_PROC....65
4.4.1.2.10 Меню Configure standard kernel features (for small systems)....65
4.4.1.2.10.1 KALLSYMS....65
4.4.1.2.10.2 FUTEX....66
4.4.1.2.10.3 EPOLL....66
4.4.1.2.10.4 IOSCHED_NOOP....66
4.4.1.2.10.5 IOSCHED_AS....66
4.4.1.2.10.6 IOSCHED_DEADLINE....66
4.4.1.2.10.7 IOSCHED_CFQ....66
4.4.1.2.10.8 CC_OPTIMIZE_FOR_SIZE....66
4.4.1.3 Меню Loadable module support....66
4.4.1.3.1 MODULES....66
4.4.1.3.2 MODULE_UNLOAD....66
4.4.1.3.2.1 MODULE_FORCE_UNLOAD....67
4.4.1.3.3 MODVERSIONS....67
4.4.1.3.4 KMOD....67
4.4.1.4 Меню Power management options (ACPI, APM)....67
4.4.1.4.1 PM....67
4.4.1.5 Меню File systems....67
4.4.1.5.1 PROC_FS....67
4.4.1.5.2 NFS_FS....68
4.4.2 Сетевые опции ядра (Networking support)....68
4.4.2.1 NET....68
4.4.2.2 Меню Networking options....68
4.4.2.2.1 PACKET....68
4.4.2.2.1.1 PACKET_MMAP....68
4.4.2.2.2 NETLINK_DEV....69
4.4.2.2.3 UNIX....69
4.4.2.2.4 NET_KEY....69
4.4.2.2.5 INET (опции ядра для стека TCP/IP)....69
4.4.2.2.5.1 IP_MULTICAST (групповая адресация)....69
4.4.2.2.5.2 IP_ADVANCED_ROUTER (опции маршрутизации IP)....69
4.4.2.2.5.2.1 IP_MULTIPLE_TABLES....70
4.4.2.2.5.2.1.1 IP_ROUTE_FWMARK....70
4.4.2.2.5.2.1.2 IP_ROUTE_NAT....70

4.4.2.2.5.2.2 IP_ROUTE_MULTIPATH....70
4.4.2.2.5.2.3 IP_ROUTE_TOS....70
4.4.2.2.5.2.4 IP_ROUTE_VERBOSE....70
4.4.2.2.5.3 IP_PNP (опции сетевой загрузки)....70
4.4.2.2.5.3.1 IP_PNP_DHCP....71
4.4.2.2.5.3.2 IP_PNP_BOOTP....71
4.4.2.2.5.3.3 IP_PNP_RARP....71
4.4.2.2.5.4 NET_IPIP (туннелирование)....71
4.4.2.2.5.5 NET_IPGRE (туннелирование)....71
4.4.2.2.5.5.1 NET_IPGRE_BROADCAST....71
4.4.2.2.6 IP_MROUTE (опции групповой маршрутизации)....71
4.4.2.2.6.1 IP_PIMSM_V1....71
4.4.2.2.6.2 IP_PIMSM_V2....71
4.4.2.2.7 ARPD (демон ARP)....72
4.4.2.2.8 SYN_COOKIES....72
4.4.2.2.9 INET_AH....72
4.4.2.2.10 INET_ESP....72
4.4.2.2.11 INET_IPCOMP....72
4.4.2.2.12 IP_VS....72
4.4.2.2.13 IPV6 (опции протокола IPv6)....73
4.4.2.2.14 Меню Network packet filtering (replaces ipchains) - фильтрация пакетов....73
4.4.2.2.14.1 NETFILTER....73
4.4.2.2.14.1.1 NETFILTER_DEBUG....74
4.4.2.2.14.1.2 BRIDGE_NETFILTER....74
4.4.2.2.14.2 Меню IP: Netfilter Configuration....74
4.4.2.2.14.2.1 IP_NF_CONNTRACK....74
4.4.2.2.14.2.1.1 IP_NF_FTP....74
4.4.2.2.14.2.1.2 IP_NF_IRC....74
4.4.2.2.14.2.1.3 IP_NF_TFTP....75
4.4.2.2.14.2.1.4 IP_NF_AMANDA....75
4.4.2.2.14.2.2 IP_NF_QUEUE....75
4.4.2.2.14.2.3 IP_NF_IPTABLES....75
4.4.2.2.14.2.3.1 IP_NF_MATCH_LIMIT....75
4.4.2.2.14.2.3.2 IP_NF_MATCH_IPRANGE....75
4.4.2.2.14.2.3.3 IP_NF_MATCH_MAC....75
4.4.2.2.14.2.3.4 IP_NF_MATCH_PKTTYPE....75
4.4.2.2.14.2.3.5 IP_NF_MATCH_MARK....76
4.4.2.2.14.2.3.6 IP_NF_MATCH_MULTIPORT....76
4.4.2.2.14.2.3.7 IP_NF_MATCH_TOS....76
4.4.2.2.14.2.3.8 IP_NF_MATCH_RECENT....76
4.4.2.2.14.2.3.9 IP_NF_MATCH_ECN....76
4.4.2.2.14.2.3.10 IP_NF_MATCH_DSCP....76
4.4.2.2.14.2.3.11 IP_NF_MATCH_AH_ESP....76
4.4.2.2.14.2.3.12 IP_NF_MATCH_LENGTH....76
4.4.2.2.14.2.3.13 IP_NF_MATCH_TTL....77
4.4.2.2.14.2.3.14 IP_NF_MATCH_TCPMSS....77
4.4.2.2.14.2.3.15 IP_NF_MATCH_HELPER....77
4.4.2.2.14.2.3.16 IP_NF_MATCH_STATE....77
4.4.2.2.14.2.3.17 IP_NF_MATCH_CONNTRACK....77
4.4.2.2.14.2.3.18 IP_NF_MATCH_OWNER....77
4.4.2.2.14.2.3.19 IP_NF_MATCH_PHYSDEV....77
4.4.2.2.14.2.3.20 IP_NF_FILTER....77
4.4.2.2.14.2.3.20.1 IP_NF_TARGET_REJECT....77
4.4.2.2.14.2.3.21 IP_NF_NAT....78
4.4.2.2.14.2.3.22 IP_NF_NAT_NEEDED....78
4.4.2.2.14.2.4 IP_NF_TARGET_MASQUERADE....78
4.4.2.2.14.2.5 IP_NF_TARGET_REDIRECT....78
4.4.2.2.14.2.6 IP_NF_TARGET_NETMAP....78
4.4.2.2.14.2.7 IP_NF_TARGET_SAME....78
4.4.2.2.14.2.8 IP_NF_NAT_LOCAL....78
4.4.2.2.14.2.9 IP_NF_NAT_SNMP_BASIC....78
4.4.2.2.14.2.10 IP_NF_NAT_IRC....79
4.4.2.2.14.2.11 IP_NF_NAT_FTP....79
4.4.2.2.14.2.12 IP_NF_NAT_TFTP....79
4.4.2.2.14.2.13 IP_NF_NAT_AMANDA....79
4.4.2.2.14.2.14 IP_NF_MANGLE....79
4.4.2.2.14.2.14.1 IP_NF_TARGET_TOS....79
4.4.2.2.14.2.14.2 IP_NF_TARGET_ECN....79
4.4.2.2.14.2.14.3 IP_NF_TARGET_DSCP....79
4.4.2.2.14.2.14.4 IP_NF_TARGET_MARK....79
4.4.2.2.14.2.14.5 IP_NF_TARGET_CLASSIFY....79
4.4.2.2.14.2.15 IP_NF_TARGET_LOG....80
4.4.2.2.14.2.16 IP_NF_TARGET_ULOG....80
4.4.2.2.14.2.17 IP_NF_TARGET_TCPMSS....80
4.4.2.2.14.2.18 IP_NF_ARPTABLES....80
4.4.2.2.14.2.18.1 IP_NF_ARPFILTER....80

4.4.2.2.14.2.18.2 IP_NF_ARP_MANGLE....80
4.4.2.2.14.2.19 IP_NF_COMPAT_IPCHAINS....81
4.4.2.2.14.2.19.1 IP_NF_COMPAT_IPFWADM....81
4.4.2.2.14.2.20 IP_NF_TARGET_NOTRACK....81
4.4.2.2.14.2.21 IP_NF_RAW....81
4.4.2.2.14.2.22 IP_NF_MATCH_ADDRTYPE....81
4.4.2.2.14.2.23 IP_NF_MATCH_REALM....81
4.4.2.2.14.3 Меню IPv6: Netfilter Configuration....81
4.4.2.2.14.4 Меню DECnet: Netfilter Configuration....81
4.4.2.2.14.5 Меню Bridge: Netfilter Configuration....82
4.4.2.2.14.5.1 BRIDGE_NF_EBTABLES....82
4.4.2.2.14.5.1.1 BRIDGE_EBT_BROUTE....82
4.4.2.2.14.5.1.2 BRIDGE_EBT_T_FILTER....82
4.4.2.2.14.5.1.3 BRIDGE_EBT_T_NAT....82
4.4.2.2.14.5.1.4 BRIDGE_EBT_802_3....82
4.4.2.2.14.5.1.5 BRIDGE_EBT_AMONG....82
4.4.2.2.14.5.1.6 BRIDGE_EBT_ARP....82
4.4.2.2.14.5.1.7 BRIDGE_EBT_IP....83
4.4.2.2.14.5.1.8 BRIDGE_EBT_LIMIT....83
4.4.2.2.14.5.1.9 BRIDGE_EBT_MARK....83
4.4.2.2.14.5.1.10 BRIDGE_EBT_PKTTYPE....83
4.4.2.2.14.5.1.11 BRIDGE_EBT_STP....83
4.4.2.2.14.5.1.12 BRIDGE_EBT_VLAN....83
4.4.2.2.14.5.1.13 BRIDGE_EBT_ARPREPLY....83
4.4.2.2.14.5.1.14 BRIDGE_EBT_DNAT....83
4.4.2.2.14.5.1.15 BRIDGE_EBT_MARK_T....83
4.4.2.2.14.5.1.16 BRIDGE_EBT_REDIRECT....83
4.4.2.2.14.5.1.17 BRIDGE_EBT_SNAT....84
4.4.2.2.14.5.1.18 BRIDGE_EBT_LOG....84
4.4.2.2.15 XFRM....84
4.4.2.2.16 XFRM_USER....84
4.4.2.2.17 Меню SCTP Configuration....84
4.4.2.2.18 ATM....84
4.4.2.2.19 BRIDGE....84
4.4.2.2.20 VLAN_8021Q....85
4.4.2.2.21 DECNET....85
4.4.2.2.22 LLC....85
4.4.2.2.23 LLC2....85
4.4.2.2.24 IPX....85
4.4.2.2.25 ATALK....85
4.4.2.2.26 X25....85
4.4.2.2.27 LAPB....85
4.4.2.2.28 NET_DIVERT....86
4.4.2.2.29 ECONET....86
4.4.2.2.30 WAN_ROUTER....86
4.4.2.2.31 NET_FASTROUTE....86
4.4.2.2.32 NET_HW_FLOWCONTROL....86
4.4.2.2.33 Меню QoS....87
4.4.2.2.33.1 NET_SCHED....87
4.4.2.2.33.2 Меню Packet scheduler clock source....87
4.4.2.2.33.2.1 NET_SCH_CLK_JIFFIES....87
4.4.2.2.33.2.2 NET_SCH_CLK_GETTIMEOFDAY....88
4.4.2.2.33.2.3 NET_SCH_CLK_CPU....88
4.4.2.2.33.2.4 NET_SCH_CBQ....88
4.4.2.2.33.2.5 NET_SCH_HTB....88
4.4.2.2.33.2.6 NET_SCH_HFSC....88
4.4.2.2.33.2.7 NET_SCH_CSZ....88
4.4.2.2.33.2.8 NET_SCH_ATM....88
4.4.2.2.33.2.9 NET_SCH_PRIO....88
4.4.2.2.33.2.10 NET_SCH_RED....89
4.4.2.2.33.2.11 NET_SCH_SFQ....89
4.4.2.2.33.2.12 NET_SCH_TEQL....89
4.4.2.2.33.2.13 NET_SCH_TBF....89
4.4.2.2.33.2.14 NET_SCH_GRED....89
4.4.2.2.33.2.15 NET_SCH_DSMARK....89
4.4.2.2.33.2.16 NET_SCH_NETEM....89
4.4.2.2.33.2.17 NET_SCH_INGRESS....89
4.4.2.2.33.2.18 NET_QOS....89
4.4.2.2.33.2.18.1 NET_ESTIMATOR....90
4.4.2.2.33.2.19 NET_CLS....90
4.4.2.2.33.2.19.1 NET_CLS_TCINDEX....90
4.4.2.2.33.2.19.2 NET_CLS_ROUTE4....90
4.4.2.2.33.2.19.2.1 NET_CLS_ROUTE....90
4.4.2.2.34 NET_CLS_FW....90
4.4.2.2.35 NET_CLS_U32....90
4.4.2.2.35.1 CLS_U32_PERF....90

4.4.2.2.36 NET_CLS_RSVP....90
4.4.2.2.37 NET_CLS_RSVP6....90
4.4.2.2.38 NET_CLS_ACT....91
4.4.2.2.38.1 NET_ACT_POLICE....91
4.4.2.2.39 NET_CLS_POLICE....91
4.4.2.2.40 Меню Network testing....91
4.4.2.2.40.1 NET_PKTGEN....91
4.4.2.3 NETPOLL....91
4.4.2.4 NETPOLL_RX....91
4.4.2.5 NETPOLL_TRAP....91
4.4.2.6 NET_POLL_CONTROLLER....91
4.4.2.7 Меню Network device support....91
4.4.2.7.1 NETDEVICES....92
4.4.2.7.1.1 DUMMY....92
4.4.2.7.1.2 BONDING....92
4.4.2.7.1.3 EQUALIZER....92
4.4.2.7.1.4 TUN....92
4.4.2.7.1.5 ETHERTAP....93
4.4.2.7.1.6 Меню драйверов устройств....93
4.4.2.7.1.7 Меню Wan interfaces....93
4.4.2.7.1.7.1 HDLC....94
4.4.2.7.1.7.1.1 HDLC_RAW....94
4.4.2.7.1.7.1.2 HDLC_RAW_ETH....94
4.4.2.7.1.7.1.3 HDLC_CISCO....94
4.4.2.7.1.7.1.4 HDLC_FR....94
4.4.2.7.1.7.1.5 HDLC_PPP....94
4.4.2.7.1.7.1.6 HDLC_X25....94
4.4.2.7.1.7.2 DLCI....94
4.4.2.7.1.7.2.1 DLCI_COUNT....94
4.4.2.7.1.7.3 LAPBETHER....94
4.4.2.7.1.7.4 X25_ASY....95
4.4.2.7.1.8 PPP....95
4.4.2.7.1.8.1 PPP_MULTILINK....95
4.4.2.7.1.8.2 PPP_FILTER....95
4.4.2.7.1.8.3 PPP_ASYNC....95
4.4.2.7.1.8.4 PPP_SYNC_TTY....95
4.4.2.7.1.8.5 PPP_DEFLATE....95
4.4.2.7.1.8.6 PPP_BSDCOMP....95
4.4.2.7.1.8.7 PPPOE....96
4.4.2.7.1.8.8 PPPOATM....96
4.4.2.7.1.9 SLIP....96
4.4.2.7.1.10 SHAPER....96
4.4.2.7.1.11 NETCONSOLE....96
4.4.3 Опции выбора модели безопасности для ядра....96
4.4.3.1 SECURITY....97
4.4.3.1.1 SECURITY_NETWORK....97
4.4.3.1.2 SECURITY_CAPABILITIES....97
4.4.3.1.3 SECURITY_ROOTPLUG....97
4.4.3.1.4 Опции поддержки NSA SELinux....97
4.4.3.1.4.1 SECURITY_SELINUX....97
4.4.3.1.4.2 SECURITY_SELINUX_BOOTPARAM....97
4.4.3.1.4.3 SECURITY_SELINUX_DISABLE....97
4.4.3.1.4.4 SECURITY_SELINUX_DEVELOP....98
4.4.3.1.4.5 SECURITY_SELINUX_MLS....98
4.4.4 Опции поддержки криптографии....98
4.4.4.1 CRYPTO_HMAC....98
4.4.4.2 CRYPTO_NULL....98
4.4.4.3 CRYPTO_MD4....98
4.4.4.4 CRYPTO_MD5....98
4.4.4.5 CRYPTO_SHA1....98
4.4.4.6 CRYPTO_SHA256....98
4.4.4.7 CRYPTO_SHA512....99
4.4.4.8 CRYPTO_DES....99
4.4.4.9 CRYPTO_BLOWFISH....99
4.4.4.10 CRYPTO_TWOFISH....99
4.4.4.11 CRYPTO_SERPENT....99
4.4.4.12 CRYPTO_AES_586....99
4.4.4.13 CRYPTO_CAST5....99
4.4.4.14 CRYPTO_CAST6....99
4.4.4.15 CRYPTO_TEA....99
4.4.4.16 CRYPTO_ARC4....99
4.4.4.17 CRYPTO_KHAZAD....100
4.4.4.18 CRYPTO_DEFLATE....100
4.4.4.19 CRYPTO_MICHAEL_MIC....100
4.4.4.20 CRYPTO_CRC32C....100
4.4.4.21 CRYPTO_TEST....100

4.4.5 Меню Library routines....100
4.4.5.1 CRC_CCITT....100
4.4.5.2 CRC32....100
4.4.5.3 LIBCRC32C....100
4.4.5.4 ZLIB_INFLATE....101
4.4.5.5 ZLIB_DEFLATE....101
4.5 Компиляция и установка ядра....101
5 Средства работы с пакетами в Linux....102
5.1 Netfilter....102
5.1.1 Основные возможности программ....102
5.1.2 Что можно делать с помощью netfilter/iptables?....102
5.1.3 Система выбора пакетов iptables....103
5.1.4 Ловушки Netfilter....103
5.1.5 Цепочки iptables....103
5.1.5.1 Встроенные цепочки....103
5.1.5.1.1 PREROUTING....103
5.1.5.1.2 INPUT....104
5.1.5.1.3 FORWARD....104
5.1.5.1.4 POSTROUTING....104
5.1.5.1.5 OUTPUT....104
5.1.5.2 Пользовательские цепочки....104
5.1.6 Таблицы iptables....104
5.1.6.1 Таблица raw....104
5.1.6.2 Фильтрация пакетов (filter)....104
5.1.6.3 Трансляция адресов и номеров портов (таблица nat)....105
5.1.6.3.1 Маскирование (Masquerading), пересылка в другие порты (Port Forwarding) и прозрачные службы Proxy....105
5.1.6.4 Изменение пакетов (таблица mangle)....105
5.1.6.5 Контроль состояния соединений....105
5.1.6.5.1 Состояния соединений....106
5.1.7 Прохождение пакетов через таблицы и цепочки....107
5.1.7.1 Пакеты, адресованные данному хосту....107
5.1.7.2 Локально сгенерированные пакеты....107
5.1.7.3 Пересылаемые пакеты....108
5.1.8 Операции iptables....109
5.1.8.1 Основные операции....109
5.1.8.1.1 ACCEPT....109
5.1.8.1.2 DROP....109
5.1.8.1.3 QUEUE....109
5.1.8.1.4 RETURN....109
5.1.8.2 Дополнительные операции....110
5.1.8.2.1 BALANCE....110
5.1.8.2.2 CLASSIFY....110
5.1.8.2.3 CLUSTERIP....110
5.1.8.2.4 CONNMARK....110
5.1.8.2.5 DNAT....111
5.1.8.2.6 DSCP....111
5.1.8.2.7 ECN....111
5.1.8.2.8 LOG....112
5.1.8.2.9 MARK....112
5.1.8.2.10 MASQUERADE....112
5.1.8.2.11 NETMAP....113
5.1.8.2.12 NOTRACK....113
5.1.8.2.13 REDIRECT....113
5.1.8.2.14 REJECT....113
5.1.8.2.15 ROUTE....114
5.1.8.2.16 SAME....114
5.1.8.2.17 SNAT....114
5.1.8.2.18 TCPMSS....115
5.1.8.2.19 TOS....115
5.1.8.2.20 TRACE....115
5.1.8.2.21 TTL....116
5.1.8.2.22 ULOG....116
5.1.8.3 Операции расширения....116
5.1.9 Создание правил фильтрации пакетов....116
5.1.9.1 Опции команд iptables....117
5.1.9.1.1 Выбор таблицы (-t)....117
5.1.9.1.2 Выбор операции для правила (-j)....117
5.1.9.1.3 Объем выводимой информации (-v)....117
5.1.9.1.4 Формат адресов при выводе (-n)....117
5.1.9.1.5 Формат представления значений счетчиков (-x)....117
5.1.9.1.6 Инициализация счетчиков (-c)....117
5.1.9.1.7 Нумерация строк....117
5.1.9.1.8 Загрузка модулей....118
5.1.9.2 Команды iptables....118
5.1.9.2.1 Команды управления цепочками и таблицами в целом....118

5.1.9.2.1.1 Создание цепочки....118
5.1.9.2.1.2 Удаление цепочки....118
5.1.9.2.1.3 Переименование цепочки....118
5.1.9.2.1.4 Сброс цепочки....118
5.1.9.2.1.5 Просмотр списка правил в цепочках....118
5.1.9.2.1.6 Сброс счетчиков....119
5.1.9.2.1.7 Установка политики для цепочек....119
5.1.9.2.2 Команды управления отдельными правилами в цепочках....119
5.1.9.2.2.1 Добавление правила в цепочку....120
5.1.9.2.2.2 Вставка правила в указанную позицию....120
5.1.9.2.2.3 Замена правила в указанной позиции....120
5.1.9.2.2.4 Удаление правила....120
5.1.9.2.2.4.1 Удаление по номеру....120
5.1.9.2.2.4.2 Удаление по спецификации....120
5.1.9.3 Соответствия для правил iptables....121
5.1.9.3.1 Встроенные соответствия....121
5.1.9.3.1.1 Соответствие протокола....121
5.1.9.3.2 Соответствие адресов IP....121
5.1.9.3.2.1 Адрес отправителя....121
5.1.9.3.2.2 Адрес получателя....121
5.1.9.3.3 Соответствие физического интерфейса....122
5.1.9.3.3.1 Приемный интерфейс....122
5.1.9.3.3.2 Передающий интерфейс....122
5.1.9.3.4 Состояние фрагментации....122
5.1.9.4 Дополнительные сопоставления....122
5.1.9.4.1 Протокол TCP....123
5.1.9.4.1.1 Флаги TCP....123
5.1.9.4.1.1.1 Флаг SYN....123
5.1.9.4.1.2 Соответствие MSS (TCP SYN)....123
5.1.9.4.1.3 Опции TCP....123
5.1.9.4.2 Соответствие портов TCP и UDP....123
5.1.9.4.2.1 Порт отправителя....124
5.1.9.4.2.2 Порт получателя....124
5.1.9.4.3 Протокол ICMP....124
5.1.9.4.3.1 Тип сообщения ICMP....124
5.1.9.5 Загружаемые соответствия....125
5.1.9.5.1 Соответствие MAC-адресов....125
5.1.9.5.2 Пороговые значения частоты совпадений....125
5.1.9.5.3 Соответствие маркеров....126
5.1.9.5.4 Соответствие владельца пакета....126
5.1.9.5.4.1 Идентификатор пользователя....126
5.1.9.5.4.2 Идентификатор группы....126
5.1.9.5.4.3 Идентификатор процесса....126
5.1.9.5.4.4 Идентификатор сессии....127
5.1.9.5.5 Соответствие состояния соединения....127
5.1.9.5.6 Соответствия для множества портов....127
5.1.9.5.6.1 Соответствие для группы портов....127
5.1.9.5.6.2 Соответствие mport....128
5.1.9.5.7 Соответствие типа обслуживания (TOS)....128
5.1.9.5.8 Соответствие времени жизни (TTL)....128
5.1.9.5.9 Соответствие ah....128
5.1.9.5.10 Соответствие esp....128
5.1.9.5.11 Расширенное соответствие состояния соединений (conntrack)....129
5.1.9.5.12 Соответствие DSCP....129
5.1.9.5.13 Соответствие helper-модулей....129
5.1.9.5.14 Соответствие размера пакетов (length)....130
5.1.9.5.15 Соответствие физического устройства (physdev)....130
5.1.9.5.16 Соответствие типа пакетов (pkttype)....130
5.1.9.5.17 Соответствие типа адреса (addrtype)....130
5.1.9.5.18 Соответствие recent....131
5.1.9.5.19 Соответствие ECN....132
5.1.9.5.20 Соответствие unclean....132
5.1.9.6 Дополнительные соответствия....132
5.1.9.6.1 Соответствие condition....132
5.1.9.6.2 Соответствие fuzzy....133
5.1.9.6.3 Соответствие iplimit....133
5.1.9.6.4 Соответствие ipv4options....134
5.1.9.6.5 Соответствие nth....134
5.1.9.6.6 Соответствие psd....135
5.1.9.6.7 Соответствие quota....135
5.1.9.6.8 Соответствие random....135
5.1.9.6.9 Соответствие realm....136
5.1.9.6.10 Соответствие record_rpc....136
5.1.9.6.11 Соответствие string....136
5.1.9.6.12 Соответствие time....136
5.1.9.6.13 Соответствие u32....137

5.1.9.6.13.1 Проверка значений 2-байтовых полей....137
5.1.9.6.13.2 Проверка значений 1-байтовых полей....138
5.1.9.6.13.3 Просмотр 4 байтов сразу....138
5.1.9.6.13.4 Проверка каждого байта в заголовке....138
5.1.9.6.13.5 Проверка отдельных битов....138
5.1.9.6.13.6 Объединение проверок....139
5.1.9.6.13.7 Работа с заголовками пакетов....139
5.1.9.6.13.7.1 TCP....139
5.1.9.6.13.7.2 ICMP....140
5.1.9.6.13.7.3 UDP....140
5.1.9.6.13.8 Примеры правил....140
5.1.10 Утилиты iptables....141
5.1.10.1 iptables-save....141
5.1.10.2 iptables-restore....142
5.1.11 Компиляция и установка iptables....142
5.2 Программа ebtables....143
5.2.1 Цепочки ebtables....143
5.2.2 Таблицы ebtables....143
5.2.2.1 Таблица filter....143
5.2.2.2 Таблица nat....143
5.2.2.3 Таблица broute....143
5.2.3 Операции ebtables....144
5.2.4 Команды и опции ebtables....144
5.2.4.1 Основные команды....144
5.2.4.2 Дополнительные команды....145
5.2.5 Спецификации правил....145
5.2.6 Основные сопоставления....146
5.2.6.1 Проверка полей DSAP/SSAP и SNAP в кадрах 802.3....146
5.2.6.2 Проверка наличия адресов в списке....146
5.2.6.3 Проверка полей ARP....147
5.2.6.4 Проверка полей IP....147
5.2.6.5 Ограничение темпа совпадений (limit)....148
5.2.6.6 Проверка маркеров....148
5.2.6.7 Проверка типа кадров....148
5.2.6.8 Проверка STP....148
5.2.6.9 Проверка параметров VLAN....149
5.2.7 Сторожа (WATCHER)....149
5.2.7.1 Операция log....149
5.2.8 Дополнительные операции ebtables....150
5.2.8.1 Соответствие arpreply....150
5.2.8.2 Операция dnat....150
5.2.8.3 Операция mark....150
5.2.8.4 Операция redirect....150
5.2.8.5 Операция snat....151
5.3 Программа arptables....151
5.3.1 Цепочки arptables....151
5.3.2 Таблица filter....151
5.3.3 Операции arptables....151
5.3.4 Команды и параметры arptables....151
5.3.4.1 Основные команды arptables....152
5.3.4.2 Дополнительные команды....152
5.3.4.3 Опции спецификации правил....152
5.3.5 Дополнительные операции arptables....153
6 Организация соединений VPN....155
6.1 FreeS/WAN IPsec VPN....155
6.1.1 Программа ipsec....156
6.1.2 Настройка параметров соединений IPsec VPN....156
6.1.3 Известные проблемы ipsec....157
6.2 SSL VPN....158
6.2.1 Опции команды stunnel....158
6.3 PPTP VPN....162
6.3.1 Сервер....162
6.3.1.1 Опции команды pptpd....162
6.3.1.2 Конфигурационный файл pptpd.conf....163
6.3.1.2.1 Опции....163
6.3.2 Клиент....164
6.3.2.1 Опции....164
7 Централизованные средства управления пользователями....166
7.1 RADIUS....166
7.1.1 FreeRADIUS....166
7.1.1.1 Radiusd....166
7.1.1.1.1 Опции....166
7.1.1.1.2 Конфигурационные файлы....167
7.1.1.1.2.1 Файл radiusd.conf....167
7.1.1.1.2.1.1 Формат файла....167
7.1.1.1.2.2 Файл dictionary....168

7.1.1.1.2.3 Файл clients....168
7.1.1.1.2.4 Файл clients.conf....168
7.1.1.1.2.5 Файл naslist....169
7.1.1.1.2.6 Файл hints....169
7.1.1.1.2.7 Файл huntgroups....169
7.1.1.1.2.8 Файл users....169
7.1.1.1.2.8.1 Операторы....169
7.1.1.1.2.8.2 Предостережения....170
7.1.1.1.2.8.2.1 Примеры....170
7.1.1.1.2.8.3 Рекомендации....170
7.1.1.1.2.9 Файл acct_users....171
7.1.1.2 radclient....171
7.1.1.2.1 Опции....171
7.1.1.3 Radlast....171
7.1.1.4 Radtest....172
7.1.1.4.1 Опции....172
7.1.1.5 Radwho....172
7.1.1.5.1 Опции....172
7.1.1.6 Radzap....173
7.1.1.6.1 Опции....173
7.1.2 GNU Radius....173
7.1.2.1 Схемы аутентификации....173
7.1.2.2 Схемы учета работы пользователей....174
7.1.2.3 Возможности расширения....174
7.1.3 Cistron RADIUS....174
7.2 LDAP....174
7.2.1 OpenLDAP....174
8 Инструменты для создания и поддержки политики безопасности....175
8.1 Firewall Builder....175
8.2 Конвертер политики Checkpoint Firewall-1 в формат FirewallBuilder....175
8.3 Модуль Firewall программы Webmin....176
9 Персональные брандмауэры для платформы Windows....178
9.1 NetDefender....178
9.2 Privaria....178
10 Предотвращение спама и проникновения вирусов в сеть....179
10.1 Антивирусные средства....179
10.1.1 ClamAV....179
10.1.2 F-prot....179
10.1.2.1 F-Prot Antivirus для рабочих станций Linux....179
10.1.2.2 F-Prot Antivirus для почтовых серверов Linux x86....179
10.2 Средства предотвращения спама....180
10.2.1 Фильтрация на граничном шлюзе....180
10.2.2 DNSBL....181
10.2.2.1 Функционирование DNSBL....182
10.2.2.2 Запросы DNSBL....182
10.2.2.3 Политика DNSBL....183
10.2.2.4 DRBL....183
10.2.3 Фильтрация по содержимому....184
10.2.4 MailScanner....184
10.2.5 SpamAssassin....185
11 Инструменты администратора....187
11.1 Системные утилиты UNIX....187
11.1.1 Пакет coreutils....187
11.1.1.1 Chgrp....187
11.1.1.1.1 Опции POSIX....187
11.1.1.1.2 Опции чернового стандарта AUSTIN....187
11.1.1.1.3 Опции расширения GNU....187
11.1.1.1.4 Соответствие стандартам....188
11.1.1.2 Chmod....188
11.1.1.2.1 Опции chmod....188
11.1.1.2.1.1 Опции POSIX....188
11.1.1.2.1.2 Опции GNU....188
11.1.1.2.2 Символьный формат задания прав доступа....188
11.1.1.2.3 Числовой формат задания прав доступа....189
11.1.1.2.4 Соответствие стандартам....189
11.1.1.3 Chown....189
11.1.1.3.1 Опции POSIX....190
11.1.1.3.2 Опции расширения GNU....190
11.1.1.4 Groups....190
11.1.1.5 Id....190
11.1.1.6 Kill....191
11.1.1.7 Md5sum....191
11.1.1.7.1 Опции....191
11.1.1.8 Su....192
11.1.1.9 Uname....192
11.1.1.10 Who....192

11.1.2 Пакет net-tools....193
11.1.2.1 arp....193
11.1.2.2 Hostname....195
11.1.2.2.1 Определение имени....195
11.1.2.2.2 Установка имени....195
11.1.2.2.2.1 Установка FQDN....195
11.1.2.2.3 Опции....195
11.1.2.3 Ifconfig....196
11.1.2.4 Mii-tool....198
11.1.2.4.1 Опции....198
11.1.2.5 netstat....199
11.1.2.5.1 Опции типа информации....199
11.1.2.5.2 Опции вывода....199
11.1.2.5.3 Формат вывода....200
11.1.2.5.3.1 Активные соединения Internet (TCP, UDP, raw)....200
11.1.2.5.3.2 Активные сокеты UNIX domain....201
11.1.2.6 Route....203
11.1.2.6.1 Опции....203
11.1.2.6.2 Примеры использования....204
11.1.2.6.3 Формат вывода....204
11.2 Дополнительные утилиты....206
11.2.1 dmesg....206
11.2.2 Ethtool....206
11.2.2.1 Опции....206
11.2.3 Free....207
11.2.3.1 Опции....208
11.2.4 Ifplugstatus....208
11.2.4.1 Опции....208
11.2.4.2 Возвращаемые значения....209
11.2.5 Pgrep, pkill....209
11.2.5.1 Опции....209
11.2.5.2 Примеры использования....210
11.2.5.3 Коды возврата....210
11.2.5.4 Известные проблемы....210
11.2.6 Ps....210
11.2.6.1 Опции командной строки....210
11.2.6.1.1 Простой выбор процессов....210
11.2.6.1.2 Выбор процессов по списку....211
11.2.6.1.3 Опции управления форматом вывода....211
11.2.6.1.3.1 Модификаторы формата вывода....211
11.2.6.1.4 Опции управления выводом информации о потоках....212
11.2.6.1.5 Информационные опции....212
11.2.6.2 Пользовательский формат вывода....212
11.2.6.3 Специфика работы с программой....213
11.2.6.4 Флаги процессов....213
11.2.6.5 Коды состояния процессов....213
11.2.6.6 Ключи сортировки....214
11.2.6.6.1 Стандартные указатели формата....214
11.2.6.6.2 Указатели формата AIX....215
11.2.6.6.3 Переменные окружения....215
11.2.6.6.4 Управление трактовкой опций....216
11.2.6.7 Примеры использования....216
11.2.7 Pstree....216
11.2.7.1 Опции....217
11.2.8 SysCtl....217
11.2.8.1 Параметры....217
11.2.9 Top....218
11.2.9.1 Опции командной строки....219
11.2.9.2 Формат вывода....220
11.2.9.2.1 Описания полей вывода....220
11.2.9.2.2 Выбор и упорядочивание колонок....221
11.2.9.3 Интерактивное управление выводом....221
11.2.9.3.1 Глобальные команды....222
11.2.9.3.2 Команды для области системной информации (SUMMARY Area)....222
11.2.9.3.3 Команды для списка задач....223
11.2.9.3.3.1 Представление списка задач....223
11.2.9.3.3.2 Управление содержимым списка задач....223
11.2.9.3.3.3 Управление размером списка задач....223
11.2.9.3.3.4 Управление сортировкой задач в списке....223
11.2.9.3.4 Цветовое выделение....224
11.2.9.4 Альтернативный режим отображения....224
11.2.9.4.1 Окна альтернативного режима....224
11.2.9.4.2 Команды управления окнами....225
11.2.9.5 Конфигурационные файлы....225
11.2.9.5.1 Системный файл конфигурации....225
11.2.9.5.2 Персональный файл конфигурации....225

11.2.10 Uptime....225
11.2.11 Vmstat....225
11.2.11.1 Опции....226
11.2.11.2 Описания полей вывода....226
11.2.11.2.1 Процессы и память....226
11.2.11.2.2 Дисковые операции (-d)....227
11.2.11.2.3 Дисковые разделы (-p)....227
11.2.11.2.4 Именованные блоки памяти....227
11.2.12 w....228
11.2.12.1 Опции....228
11.3 Система удаленного управления Webmin....228
11.4 Сканеры безопасности....231
11.4.1 COPS....231
11.4.2 SATAN....231
11.4.3 Internet Security Scanner (ISS)....231
11.4.4 Nessus....231
11.4.4.1 Сервер nessusd....231
11.4.4.1.1 Опции nessusd....231
11.4.4.1.2 Конфигурационный файл сервера....231
11.4.4.2 Управление пользователями....233
11.4.4.3 Формат файла правил....233
11.4.4.3.1 База правил nessus....233
11.4.4.4 Клиент nessus....234
11.4.4.4.1 Опции....234
11.4.4.4.2 Графический интерфейс программы....234
11.4.4.4.2.1 Панель Nessusd host....235
11.4.4.4.2.2 Панель Plugins....235
11.4.4.4.2.3 Панель Prefs....236
11.4.4.4.2.4 Панель Scan Options....237
11.4.4.4.2.5 Панель Target Selection....237
11.4.4.4.2.6 Панель User....237
11.4.4.4.2.7 Панель KB....237
11.4.4.4.2.8 Преобразование отчетов....237
11.4.4.5 Переменные окружения....238
11.4.4.6 Файлы....238
11.5 Средства контроля целостности и обнаружения враждебного кода....238
11.5.1 chkrootkit....238
11.5.1.1 Опции....239
11.5.1.2 Сообщения программы....240
11.5.2 Tripwire....241
11.5.2.1 Режим инициализации базы данных....241
11.5.2.1.1 Опции режима Database Initialization....241
11.5.2.2 Режим проверки целостности....241
11.5.2.2.1 Опции режима Integrity Checking....242
11.5.2.3 Режим обновления базы данных....243
11.5.2.3.1 Опции режима Database Update....243
11.5.2.4 Режим обновления политики....243
11.5.2.4.1 Опции режима Policy Update....244
11.5.2.5 Тестовый режим....244
11.5.2.5.1 Опции режима Test....244
11.5.3 samhain....244
11.5.3.1 Опции команд samhain/yule....245
11.5.3.2 Сигналы....246
11.5.3.3 База данных....246
11.5.3.4 Журнальный файл....246
11.5.3.5 Почтовые сообщения....247
11.5.3.6 Использование программы в режиме клиент-сервер....247
11.5.3.7 Скрытый режим работы программы - STEALTH....247
11.5.3.8 Безопасность....247
11.5.3.8.1 Известные проблемы....248
11.6 Средства обнаружения вторжений (IDS)....248
11.6.1 Snort....248
11.6.1.1 Опции....249
11.6.1.2 Фильтрация пакетов....251
11.6.1.3 Правила Snort....251
11.6.2 portsentry....251
11.6.2.1 Установка PortSentry....252
11.6.2.2 Режимы работы программы....253
11.6.2.2.1 Базовый режим мониторинга TCP (-tcp)....254
11.6.2.2.2 Базовый режим мониторинга UDP (-udp)....254
11.6.2.2.3 Режим Stealth TCP (-stcp)....254
11.6.2.2.4 Режим "Stealth" UDP (-sudp)....254
11.6.2.2.5 Расширенные режимы детектирования скрытого сканирования....254
11.6.2.2.5.1 Режим Advanced TCP stealth scan detection (-atcp)....254
11.6.2.2.5.2 Режим Advanced UDP "stealth" scan detection (-audp)....254
11.6.2.3 Проверка инсталляции....255

11.6.2.4 Сообщения программы....255
11.6.2.5 Файлы программы....255
11.6.3 Courtney....255
11.7 Средства поиска анализаторов протоколов....256
11.7.1 Sniffdet....256
11.7.1.1 Опции....256
11.7.1.2 Конфигурационный файл sniffdet.conf....256
11.7.1.2.1 Пример конфигурационного файла....257
11.7.1.3 Библиотека libsniffdet....258
11.7.1.4 Примеры использования....258
11.8 Программы анализа системных журналов....258
11.8.1 logcheck....258
11.8.2 swatch....260
11.8.2.1 Опции....260
11.8.2.2 Конфигурационный файл....261
11.8.2.2.1 Поиск соответствий....261
11.8.2.2.2 Операции при соответствии....261
11.8.2.3 Пример конфигурации....262
11.9 Средства мониторинга сетевого трафика и анализа пакетов....263
11.9.1 Библиотека pcap....263
11.9.2 tcpdump....263
11.9.2.1 Опции tcpdump....264
11.9.2.2 Фильтрация при сборе пакетов....267
11.9.2.2.1 Допустимые примитивы фильтрации пакетов....268
11.9.2.2.2 Логические выражения....271
11.9.2.2.3 Примеры фильтров....272
11.9.2.3 Формат вывода....273
11.9.2.3.1 Заголовки канального уровня....273
11.9.2.3.2 Пакеты ARP/RARP....273
11.9.2.3.3 Пакеты TCP....274
11.9.2.3.3.1 Сбор пакетов TCP с заданными комбинациями флагов (SYN-ACK, URG-ACK и т. п.). 275
11.9.2.3.4 Пакеты UDP....275
11.9.2.3.4.1 Запросы UDP к серверам DNS....276
11.9.2.3.4.2 UDP-отклики от серверов DNS....276
11.9.2.3.4.3 Декодирование SMB/CIFS....276
11.9.2.3.4.4 Запросы и отклики NFS....276
11.9.2.3.5 Запросы и отклики AFS....277
11.9.2.3.5.1 KIP AppleTalk (DDP in UDP)....277
11.9.2.3.5.2 Фрагментация IP....278
11.9.2.3.5.3 Временные метки....279
11.9.3 Анализатор протоколов Ethereal....279
11.9.3.1 Опции Ethereal....280
11.9.3.2 Графический интерфейс Ethereal....283
11.9.3.2.1 Главное окно программы....283
11.9.3.2.1.1 Верхняя панель....284
11.9.3.2.1.2 Средняя панель....284
11.9.3.2.1.3 Нижняя панель....284
11.9.3.2.1.4 Панель Filter....285
11.9.3.2.2 Меню File....285
11.9.3.2.2.1 Open, Close, Reload....285
11.9.3.2.2.2 Save, Save As....285
11.9.3.2.2.3 Print....286
11.9.3.2.2.4 Print Packet....286
11.9.3.2.2.5 Quit....286
11.9.3.2.3 Меню Edit....286
11.9.3.2.3.1 Find Frame....286
11.9.3.2.3.2 Find Next....287
11.9.3.2.3.3 Find Previous....287
11.9.3.2.3.4 Go To Frame....287
11.9.3.2.3.5 Субменю Time Reference....287
11.9.3.2.3.6 Mark Frame....287
11.9.3.2.3.7 Mark All Frames....287
11.9.3.2.3.8 Unmark All Frames....287
11.9.3.2.3.9 Preferences....287
11.9.3.2.3.10 Capture Filters....288
11.9.3.2.3.11 Display Filters....288
11.9.3.2.3.11.1 Диалоговое окно Edit ... Filter List....288
11.9.3.2.4 Диалоговое окно Filter Expression....289
11.9.3.2.4.1 Protocols....289
11.9.3.2.5 Меню Capture....290
11.9.3.2.5.1 Start....290
11.9.3.2.5.1.1 Диалоговое окно Capture Options....290
11.9.3.2.5.2 Stop....291
11.9.3.2.6 Меню Display....291
11.9.3.2.6.1 Options....291
11.9.3.2.6.1.1 Диалоговое окно Display Options....291

11.9.3.2.6.2 Match....291
11.9.3.2.6.3 Prepare....291
11.9.3.2.6.4 Colorize Display - цветовая маркировка пакетов в списке....291
11.9.3.2.6.4.1 Механизм цветового выделения....292
11.9.3.2.6.4.2 Диалоговое окно Apply Color Filters....292
11.9.3.2.6.4.2.1 Список фильтров....292
11.9.3.2.6.4.2.2 Кнопки управления фильтрами цветовой маркировки....292
11.9.3.2.6.4.2.3 Диалоговое окно Edit Color Filter....293
11.9.3.2.6.5 Collapse All....293
11.9.3.2.6.6 Expand All....293
11.9.3.2.6.7 Show Packet In New Window....293
11.9.3.2.6.8 User Specified Decodes....293
11.9.3.2.7 Меню Tools....293
11.9.3.2.7.1 Plugins....293
11.9.3.2.7.2 Follow TCP Stream....294
11.9.3.2.7.3 Decode As....294
11.9.3.2.7.4 Go To Corresponding Frame....295
11.9.3.2.7.5 TCP Stream Analysis....295
11.9.3.2.7.5.1 Time-sequence Graph (Stevens)....295
11.9.3.2.7.5.2 Time-sequence Graph (tcptrace)....295
11.9.3.2.7.5.3 Throughput Graph....296
11.9.3.2.7.5.4 RTT Graph....296
11.9.3.2.7.6 Summary....296
11.9.3.2.7.7 Protocol Hierarchy Statistics....297
11.9.3.2.7.8 Statistics....298
11.9.3.2.7.8.1 Watch protocol....298
11.9.3.2.7.8.1.1 BOOTP-DHCP....298
11.9.3.2.7.8.1.2 ITU-T H.225....298
11.9.3.2.7.8.1.3 HTTP....299
11.9.3.2.7.8.1.4 WAP-WSP....299
11.9.3.2.7.8.2 Service Response Time....299
11.9.3.2.7.8.2.1 DCE-RPC....299
11.9.3.2.7.8.2.2 Fibre Channel....300
11.9.3.2.7.8.2.3 MGCP....300
11.9.3.2.7.8.2.4 ONC-RPC....300
11.9.3.2.7.8.2.5 SMB....300
11.9.3.2.7.8.3 Conversation List....301
11.9.3.2.7.8.3.1 IO-Stat....301
11.9.3.2.7.8.4 ONC-RPC....303
11.9.3.2.7.8.4.1 Programs....303
11.9.3.2.7.8.5 RTP Streams....303
11.9.3.2.7.8.5.1 Show All....303
11.9.3.2.7.8.5.2 Analyse....303
11.9.3.2.8 Меню Help....304
11.9.3.2.8.1 About....304
11.9.3.2.9 Панель инструментов Ethereal....304
11.9.3.2.10 Диалоговое окно Preferences....304
11.9.3.2.10.1 Страница Printing....304
11.9.3.2.10.2 Страница Columns....305
11.9.3.2.10.3 Страница TCP Streams....306
11.9.3.2.10.4 Страница User Interface....306
11.9.3.2.10.5 Страница Capture....307
11.9.3.2.10.5.1 Диалоговое окно Inreface Options....307
11.9.3.2.10.6 Страница Name Resolutions....308
11.9.3.2.10.1 Страница Protocols....308
11.9.3.3 Фильтры сбора пакетов....308
11.9.3.4 Фильтры отображения....308
11.9.3.4.1 Синтаксис фильтров....309
11.9.3.5 Файлы Ethereal....311
11.9.3.6 Tethereal....312
11.9.3.7 Утилиты Ethereal....312
11.9.3.7.1 editcap....312
11.9.3.7.2 mergecap....313
11.10 Программы мониторинга соединений....313
11.10.1 IPTraf....313
11.10.1.1 Опции командной строки....314
11.10.1.2 Сигналы программы....315
11.10.1.3 Преобразование адресов....315
11.10.1.4 Режим IP Traffic Monitor....315
11.10.1.4.1 Панель TCP....315
11.10.1.4.2 Панель мониторинга трафика, не связанного с соединениями....317
11.10.1.4.2.1 Содержимое записей....317
11.10.1.4.2.1.1 ICMP....317
11.10.1.4.2.1.2 OSPF....318
11.10.1.5 Статистика для интерфейсов....319
11.10.1.5.1 Режим General Interface Statistics....319

11.10.1.5.2 Режим Detailed Interface Statistics....319
11.10.1.6 Статистика пакетов....320
11.10.1.6.1 Статистика распределения пакетов по размеру....320
11.10.1.6.2 Статистика распределения по портам TCP и UDP....321
11.10.1.6.2.1 Сортировка списка....321
11.10.1.7 Режим LAN Station Statistics....321
11.10.1.7.1 Сортировка записей для станций ЛВС....322
11.10.1.8 Фильтры IPTraf....322
11.10.1.8.1 Фильтры TCP....322
11.10.1.8.1.1 Создание нового фильтра....322
11.10.1.8.2 Активизация фильтра....323
11.10.1.8.3 Редактирование фильтров....323
11.10.1.8.4 Удаление существующего фильтра....323
11.10.1.8.5 Дезактивация фильтра....323
11.10.1.8.6 Фильтры UDP....323
11.10.1.8.7 Фильтрация прочих пакетов IP....323
11.10.1.8.8 Фильтры RP, RARP и Non-IP....323
11.10.1.9 Меню настройки параметров Iptraf....324
11.10.1.9.1 Преобразование IP-адресов....324
11.10.1.9.2 Преобразование номеров портов TCP/UDP....324
11.10.1.9.3 Режим захвата....324
11.10.1.9.4 Color....324
11.10.1.9.5 Logging....324
11.10.1.9.6 Activity mode....324
11.10.1.9.7 Source MAC addrs in traffic monitor....324
11.10.1.9.8 Таймеры программы....324
11.10.1.9.8.1 TCP Timeout....324
11.10.1.9.8.2 Log Interval....324
11.10.1.9.8.3 Screen Update Interval....324
11.10.1.9.8.4 TCP closed/idle persistence....325
11.10.1.9.9 Additional ports....325
11.10.1.9.10 Delete port/range....325
11.10.1.9.11 Идентификаторы станций ЛВС....325
11.10.2 Conntrack Viewer....325
11.10.3 Etherape....326
11.10.3.1 Опции командной строки....326
11.10.4 MRTG....327
11.10.4.1 Опции командной строки....328
11.11 Генераторы трафика....329
11.11.1 Встроенный генератор пакетов Linux....329
11.11.2 Hping2....330
11.11.2.1 Опции....331
11.11.2.1.1 Опции общего назначения....331
11.11.2.1.2 Опции выбора протокола....331
11.11.2.1.3 Опции IP....332
11.11.2.1.4 Опции ICMP....333
11.11.2.1.5 Опции TCP/UDP....333
11.11.2.1.6 Опции для всех протоколов....334
11.11.2.2 Формат вывода для протокола TCP....335
11.11.2.3 Формат вывода для пакетов UDP....335
11.11.2.4 Формат вывода для пакетов ICMP....335
11.11.2.5 Известные проблемы....336
11.11.3 packETH....336
11.11.4 Packit....337
11.11.4.1 Анализ и генерация пакетов....338
11.11.4.2 Опции командной строки....338
11.11.4.2.1 Опции режима сбора пакетов....338
11.11.4.2.2 Опции режимов генерации и трассировки....338
11.11.4.2.2.1 Опции общего назначения для режимов генерации и трассировки....338
11.11.4.2.2.2 Опции заголовков IP....339
11.11.4.2.2.3 Опции заголовков TCP....339
11.11.4.2.2.4 Опции заголовков UDP....340
11.11.4.2.2.5 Опции заголовков ICMP....340
11.11.4.2.2.5.1 Опции запросов и откликов ICMP ECHO....340
11.11.4.2.2.5.2 Опции откликов ICMP UNREACHABLE/REDIRECT/TIME EXCEEDED....340
11.11.4.2.2.5.3 Опции запросов и откликов ICMP MASK....340
11.11.4.2.2.5.4 Опции запросов и откликов ICMP TIMESTAMP....341
11.11.4.2.2.5.5 Типы и коды сообщений ICMP....341
11.11.4.2.2.6 Опции заголовков ARP....341
11.11.4.2.2.7 Опции заголовков Ethernet....342
11.11.4.3 Примеры команд....342
11.11.4.3.1 Сбор пакетов....342
11.11.4.3.2 Генерация пакетов....342
11.11.4.3.3 Трассировка....343
11.11.4.4 Известные проблемы....343
11.12 Сетевые сканеры....343

11.12.1 Nmap....343
11.12.1.1 Опции....344
11.12.1.1.1 Тип сканирования....344
11.12.1.1.1.1 Сканирование TCP SYN (-sS)....344
11.12.1.1.1.2 Сканирование TCP connect (-sT)....344
11.12.1.1.1.3 Скрытое сканирование Stealth FIN, Stealth Xmas Tree, Stealth Null (-sF -sX -sN)....344
11.12.1.1.1.4 Ping-сканирование (-sP)....345
11.12.1.1.1.5 Определение версии (-sV)....345
11.12.1.1.1.6 Сканирование UDP (-sU)....345
11.12.1.1.1.7 IP-сканирование (-sO)....346
11.12.1.1.1.8 Метод скрытого сканирования Idlescan (-sI)....346
11.12.1.1.1.9 ACK-сканирование (-sA)....347
11.12.1.1.1.10 Window-сканирование (-sW)....347
11.12.1.1.1.11 Сканирование RPC (-sR)....347
11.12.1.1.1.12 Сканирование по списку (-sL)....347
11.12.1.1.1.13 Сканирование FTP bounce attack (-b)....347
11.12.1.1.2 Опции общего назначения....348
11.12.1.1.3 Опции синхронизации....351
11.12.1.2 Выбор цели сканирования....352
11.12.1.3 Примеры....352
11.12.1.4 Графические интерфейсы nmap....353
11.12.1.4.1 Модуль Webmin....353
11.12.1.5 nmapfe....353
12 Приложения....354
12.1 Регулярные выражения (regex)....354
12.1.1 Расширенный формат....354
12.1.2 Базовый формат....355
12.1.3 Функции для работы с виртуальными выражениями....356
12.1.3.1 Компиляция регулярных выражений в POSIX....356
12.1.3.2 Совпадения POSIX....356
12.1.3.2.1 Смещения совпадающих подстрок....356
12.1.3.3 Сообщения об ошибках POSIX....357
12.1.3.4 Освобождение буферов шаблонов поиска POSIX....357
12.1.3.5 Возвращаемые значения....357
12.1.3.5.1 Коды ошибок....357
12.2 Параметры SysCtl....357
12.2.1 Виртуальная файловая система /proc....357
12.2.1.1 Файлы параметров системы....358
12.2.1.2 Каталоги процессов....359
12.2.1.2.1 Поля файла stat....361
12.2.1.3 Каталог bus....362
12.2.1.4 Каталог driver....362
12.2.1.5 Каталог fs....362
12.2.1.6 Каталог ide....362
12.2.1.7 Каталог irq....363
12.2.1.8 Каталог net....363
12.2.1.9 Каталог scsi....365
12.2.1.10 Каталоги драйверов SCSI....365
12.2.1.11 Каталог self....365
12.2.1.12 Каталог sys....365
12.2.1.12.1 Каталог sys/abi....366
12.2.1.12.2 Каталог sys/debug....366
12.2.1.12.3 Каталог sys/dev....366
12.2.1.12.4 Каталог sys/fs....366
12.2.1.12.4.1 Подкаталог binfmt_misc....366
12.2.1.12.4.1.1 Ограничения....366
12.2.1.12.4.1.2 Примеры использования....366
12.2.1.12.4.2 Файлы /proc/sys/fs....367
12.2.1.12.5 Каталог sys/kernel....368
12.2.1.12.6 Каталог sys/net....369
12.2.1.12.7 Каталог sys/proc....369
12.2.1.12.8 Каталог sys/sunrpc....369
12.2.1.12.9 Каталог sys/vm....369
12.2.1.13 Каталог sysvipc....369
12.2.1.14 Каталог tty....369
12.3 Параметры SysCtl для стека IP....369
12.3.1 Параметры IPv4....370
12.3.1.1 ip_forward - пересылка пакетов....370
12.3.1.2 ip_default_ttl - время жизни пакетов....370
12.3.1.3 ip_no_pmtu_disc....370
12.3.1.4 ip_queue_maxlen - максимальное число пакетов в очереди пользовательского пространства....370
12.3.1.5 Параметры фрагментации пакетов IP....370
12.3.1.5.1 ipfrag_high_thresh - максимальный размер памяти для сборки фрагментов....370
12.3.1.5.2 ipfrag_low_thresh - нижний предел размера для буфера сборки пакетов....370
12.3.1.5.3 ipfrag_time - время хранения фрагментов....370
12.3.1.5.4 ipfrag_secret_interval - время жизни хэш-ключа....370

12.3.1.6 Переменные INET peer storage....370
12.3.1.6.1 inet_peer_threshold....370
12.3.1.6.2 inet_peer_minttl....371
12.3.1.6.3 inet_peer_maxttl....371
12.3.1.6.4 inet_peer_gc_mintime....371
12.3.1.6.5 inet_peer_gc_maxtime....371
12.3.1.7 Переменные TCP....371
12.3.1.7.1 tcp_syn_retries....371
12.3.1.7.2 tcp_synack_retries....371
12.3.1.7.3 tcp_keepalive_time....371
12.3.1.7.4 tcp_keepalive_probes....371
12.3.1.7.5 tcp_keepalive_intvl....371
12.3.1.7.6 tcp_retries1....371
12.3.1.7.7 tcp_retries2....371
12.3.1.7.8 tcp_orphan_retries....372
12.3.1.7.9 tcp_fin_timeout....372
12.3.1.7.10 tcp_max_tw_buckets....372
12.3.1.7.11 tcp_tw_recycle....372
12.3.1.7.12 tcp_tw_reuse....372
12.3.1.7.13 tcp_max_orphans....372
12.3.1.7.14 tcp_abort_on_overflow....372
12.3.1.7.15 tcp_syncookies....372
12.3.1.7.16 tcp_stdurg....373
12.3.1.7.17 tcp_max_syn_backlog....373
12.3.1.7.18 tcp_window_scaling....373
12.3.1.7.19 tcp_timestamps....373
12.3.1.7.20 tcp_sack....373
12.3.1.7.21 tcp_fack....373
12.3.1.7.22 tcp_dsack....373
12.3.1.7.23 tcp_ecn....373
12.3.1.7.24 tcp_reordering....373
12.3.1.7.25 tcp_retrans_collapse....373
12.3.1.7.26 tcp_wmem....373
12.3.1.7.27 tcp_rmem....374
12.3.1.7.28 tcp_mem....374
12.3.1.7.29 tcp_app_win....374
12.3.1.7.30 tcp_adv_win_scale....374
12.3.1.7.31 tcp_rfc1337....374
12.3.1.7.32 tcp_low_latency....374
12.3.1.7.33 tcp_westwood....374
12.3.1.7.34 tcp_vegas_cong_avoid....374
12.3.1.7.35 tcp_bic....375
12.3.1.7.36 tcp_bic_low_window....375
12.3.1.7.37 tcp_bic_fast_convergence....375
12.3.1.8 ip_local_port_range....375
12.3.1.9 ip_nonlocal_bind....375
12.3.1.10 ip_dynaddr....375
12.3.1.11 Переменные ICMP....375
12.3.1.11.1 icmp_echo_ignore_all....375
12.3.1.11.2 icmp_echo_ignore_broadcasts....375
12.3.1.11.3 icmp_ratelimit....375
12.3.1.11.4 icmp_ratemask....376
12.3.1.11.5 icmp_ignore_bogus_error_responses....376
12.3.1.12 igmp_max_memberships....376
12.3.1.13 Конфигурация интерфейсов....376
12.3.1.13.1 log_martians....376
12.3.1.13.2 accept_redirects....376
12.3.1.13.3 forwarding....377
12.3.1.13.4 mc_forwarding....377
12.3.1.13.5 medium_id....377
12.3.1.13.6 proxy_arp....377
12.3.1.13.7 shared_media....377
12.3.1.13.8 secure_redirects....377
12.3.1.13.9 send_redirects....377
12.3.1.13.10 bootp_relay....377
12.3.1.13.11 accept_source_route....377
12.3.1.13.12 rp_filter....378
12.3.1.13.13 arp_filter....378
12.3.1.13.14 arp_announce....378
12.3.1.13.15 arp_ignore....378
12.3.1.13.16 tag....378
12.3.2 Переменные IPv6....379
12.3.3 Управление работой моста....379
12.3.3.1 bridge-nf-call-arptables....379
12.3.3.2 bridge-nf-call-iptables....379
12.3.3.3 bridge-nf-filter-vlan-tagged....379

12.4 Интерфейс сокетов Linux....379
12.4.1 Функции уровня сокета....379
12.4.2 Опции сокета....380
12.4.3 Сигналы....381
12.4.4 Параметры SysCtl....381
12.4.5 Операции IOCTL....382
12.4.6 Операции fcntl....382
12.5 Реализация протокола IP в Linux....382
12.5.1 Адресация сокетов IP....383
12.5.2 Опции сокета IP....383
12.5.3 Параметры SysCtl....387
12.5.4 Операции IOCTL....387
12.5.5 Коды ошибок....387
12.5.6 Известные проблемы....387
12.6 Реализация протокола TCP в Linux....388
12.6.1 Форматы адресов....388
12.6.2 Параметры SysCtl....388
12.6.3 Опции сокета TCP....388
12.6.4 Операции IOCTL....389
12.6.5 Обработка сетевых ошибок....389
12.6.6 Коды ошибок TCP....390
12.6.7 Известные проблемы....390
12.7 Реализация протокола UDP в Linux....390
12.7.1 Формат адреса....390
12.7.2 Обработка ошибок....390
12.7.3 Операции IOCTL....391
12.7.4 Коды ошибок....391
12.8 RAW-сокет....391
12.8.1 Формат адреса....392
12.8.2 Опции сокета....392
12.8.3 Замечания по использованию raw-сокетов....392
12.8.4 Обработка ошибок....392
12.8.5 Коды ошибок....392
12.8.6 Известные проблемы....393
12.9 Пакетный сокет в Linux....393
12.9.1 Типы адресов....393
12.9.2 Опции сокета....394
12.9.3 Операции IOCTL....394
12.9.4 Обработка ошибок....394
12.9.5 Коды ошибок....394
12.9.6 Известные проблемы....395
12.10 Протокол netlink в Linux....395
12.10.1 Семейство netlink....395
12.10.2 Форматы адресов....397
12.11 Реализация ARP в Linux....397
12.11.1 Операции IOCTL....398
12.11.2 Параметры SYSCTL....398
12.11.3 Известные ограничения....399
12.12 Интерфейс netdevice....399
12.12.1 Операции IOCTL....399
12.12.2 Известные ограничения....401
12.13 Функция ioctl....401
12.13.1 Коды ошибок....401
12.14 Структуры данных utmp и wtmp....401
12.15 SYN cookie....403
12.15.1 Что такое SYN cookie?....403
12.15.2 Атаки вслепую....404
12.15.3 Кто создал SYN cookie?....404
12.15.4 Страшилки о SYN cookie....404
12.16 Структура сетевого буфера Linux - skb....405
12.16.1 Буфер сокетов skbuff....405
12.16.1.1 struct sk_buff....405
12.16.1.2 Функции для работы с буфером skb....406
12.16.1.2.1 Функции распределения памяти для буферов....406
12.16.1.2.2 Дополнительные функции....406
12.16.1.2.3 Функции для работы со списками skb (очередями)....407
12.16.1.2.4 Операции с данными skb....407
12.17 Управление модулями ядра Linux (module-init-tools)....407
12.17.1 modprobe....407
12.17.1.1 Опции....408
12.17.1.2 Стратегия поиска модулей....409
12.17.1.3 Примеры использования....409
12.17.1.4 Безопасный режим....409
12.17.1.5 Протоколирование команд....409
12.17.1.6 Конфигурационный файл modprobe.conf....409
12.17.1.6.1 Команды....409

12.17.2 depmod....410
12.17.2.1 Опции....411
12.17.3 lsmod....411
12.17.4 insmod....411
12.17.5 rmmod....412
12.17.5.1 Опции....412
12.17.6 Конфигурационный файл modules.conf....412
12.17.6.1 Семантика....413
12.17.6.2 Синтаксис....413
12.17.6.3 Используемая по умолчанию конфигурация....416
12.17.6.4 Старый конфигурационный файл....416
12.18 Конфигурационный файл lilo.conf....416
12.19 Конфигурационный файл ipsec.conf....425
12.19.1 Секция CONN....426
12.19.1.1 Общие параметры CONN....426
12.19.1.2 Параметры CONN - автоматическая генерация ключей....427
12.19.1.3 Параметры CONN - генерация ключей вручную....428
12.19.2 Секция CONFIG....429
12.19.2.1 Неявные соединения....430
12.19.3 Файлы политики для групп....431
12.19.3.1 Используемая по умолчанию политика группы....432
12.19.4 Выбор соединения....432
12.20 Поля протоколов, используемые в фильтрах отображения Ethereal....432
12.21 Источники информации....432
12.21.1 Книги, журналы....432
12.21.2 Internet....432
12.21.2.1 Центры и команды по информационной безопасности....432
12.21.2.2 Стандарты, протоколы....432
12.21.2.3 Порталы, сетевые издания, обзоры, ссылки....433
12.21.2.4 Программы....433
12.21.2.5 Документация....433
12.21.2.6 Сайты организаций, связанных с информационной безопасностью....433

Список рисунков

Рисунок 2.1 Справочная информация программы fdisk....31
Рисунок 2.2 Меню программы cfdisk....33
Рисунок 2.3 Интерфейс программы DiskDruid....34
Рисунок 2.4 Выбор способа установки lilo....34
Рисунок 2.5 Учетные записи пользователей в файле /etc/passwd....36
Рисунок 2.6 Добавление пользователя с помощью интерфейса webmin....39
Рисунок 2.7 Файл /etc/shadow....40
Рисунок 2.8 Права доступа к файлам....41
Рисунок 2.9 Содержимое каталога /var/log....47
Рисунок 2.10. Вывод команды lastlog....48
Рисунок 2.11. Сведения о регистрации пользователей, выводимые по команде last....48
Рисунок 2.12 Фрагмент файла messages....50
Рисунок 2.13 Сообщения iptables в файле messages....51
Рисунок 4.1 Распределение памяти на межсетевом экране Linux....59
Рисунок 4.2 Сценарий установки patch-o-matic....62
Рисунок 4.3 Настройка конфигурации с использованием make config....63
Рисунок 4.4 Настройка конфигурации с использованием make menuconfig....63
Рисунок 4.5 Настройка конфигурации с использованием make xconfig....63
Рисунок 4.6 Меню Code maturity level options....64
Рисунок 4.7 Меню General setup....65
Рисунок 4.8. Меню Configure standard kernel features (for small systems)....67
Рисунок 4.9 Меню File systems....68
Рисунок 4.10 Меню Network File Systems....69
Рисунок 4.11 Меню Networking support....69
Рисунок 4.12. Меню Networking options....70
Рисунок 4.13. Меню IP virtual server support....74
Рисунок 4.14. Меню Network packet filtering....74
Рисунок 4.15 Меню IP: Netfilter Configuration....75
Рисунок 4.16 Меню IPv6: Netfilter Configuration....82
Рисунок 4.17 Меню DECnet: Netfilter Configuration....83
Рисунок 4.18 Меню Bridge: Netfilter Configuration....83
Рисунок 4.19. Меню The SCTP Protocol....86
Рисунок 4.20. Меню QoS and/or fair queueing....88
Рисунок 4.21 Меню Network testing....93
Рисунок 4.22 Опции поддержки сетевых устройств в ядре Linux....93
Рисунок 4.23 Меню Wan interfaces....95
Рисунок 4.24 Меню Security options....98
Рисунок 4.25 Меню NSA SELinux support....99
Рисунок 4.26 Меню Cryptographic otions....100
Рисунок 4.27 Меню Library routines....102
Рисунок 5.1. Просмотр списка правил в цепочках....121
Рисунок 5.2. Соответствие recent....134
Рисунок 5.3 Результат сохранения таблиц....144
Рисунок 6.1 Настройка FreeS/WAN VPN с помощью Webmin....159
Рисунок 6.2 Настройка IPsec VPN с помощью Webmin....160
Рисунок 6.3 Создание туннеля SSL VPN с использованием Webmin....164
Рисунок 6.4 Настройка сервера PPTP VPN с помощью Webmin....166
Рисунок 6.5 Создание туннеля PPTP VPN с помощью Webmin....167
Рисунок 8.1. Интерфейс программы Firewall Builder....177
Рисунок 8.2. Настройка межсетевого экрана с помощью модуля Webmin....178
Рисунок 8.3. Добавление правила с помощью интерфейса Webmin....179
Рисунок 9.1. Интерфейс прораммы Privaria....180
Рисунок 10.1. Интерфейс Webmin для настройки программы MailScanner....186
Рисунок 10.2 Интерфейс Webmin для настройки параметров SpamAssassin....188
Рисунок 11.1. Форматы вывода таблицы адресов ARP....197
Рисунок 11.2 Вывод команды ifconfig....200
Рисунок 11.3. Информация, полученная с помощью команды netstat....205
Рисунок 11.4 Вывод отчета об использовании памяти....210
Рисунок 11.5 Вывод команды top....220
Рисунок 11.6 Экран выбора и упорядочивания полей вывода программы top....224
Рисунок 11.7 Управление цветами вывода top....227
Рисунок 11.8 Персональный файл конфигурации top....228
Рисунок 11.9 Сведения о процессах и памяти....229
Рисунок 11.10 Сведения о дисковых операциях....229
Рисунок 11.11 Сведения о разделе диска....230
Рисунок 11.12 Сведения об именованных блоках памяти....230
Рисунок 11.13 Интерфейс программы Webmin....232
Рисунок 11.14 Панель Nessus host....237
Рисунок 11.15 Панель выбора модулей имитации атак....237
Рисунок 11.16 Диалоговое окно с описанием модуля....238
Рисунок 11.17 Диалоговое окно фильтрации модулей....238
Рисунок 11.18. Панель предпочтений....238
Рисунок 11.19 Панель выбора опций сканирования....239
Рисунок 11.20 Панель Target Selection....239

Рисунок 11.21 Панель User....240
Рисунок 11.22 Панель управления базой знаний....240
Рисунок 11.23 Настройка Snort с помощью Webmin....251
Рисунок 11.24 Интерфейс Webmin для настройки LogCheck....261
Рисунок 11.25 Интерфейс программы Ethereal....282
Рисунок 11.26 Окно IO-Stat....284
Рисунок 11.27 Панели главного окна Ethereal....286
Рисунок 11.28 Диалоговые окна выбора файлов (справа) и фильтров....287
Рисунок 11.29 Диалоговое окно записи файла....288
Рисунок 11.30 Выбор формата записи файла....288
Рисунок 11.31 Диалоговое окно Print....288
Рисунок 11.32 Диалоговое окно Find Frame....289
Рисунок 11.33 Диалоговое окно Preferences....290
Рисунок 11.34 Диалоговое окно Edit Capture Filter List....290
Рисунок 11.35 Диалоговое окно Filter Expression....291
Рисунок 11.36. Диалоговое окно Protocols....292
Рисунок 11.37 Диалоговое окно Capture Options....292
Рисунок 11.38 Диалоговое окно Display Options....293
Рисунок 11.39 Диалоговое окно Apply Color Filters....294
Рисунок 11.40 Диалоговое окно Edit Color Filter....295
Рисунок 11.41 Диалоговое окно выбора цвета....295
Рисунок 11.42 Вывод дерева протоколов и дампа пакета в новом окне....296
Рисунок 11.43 Диалоговое окно Decode As Show....296
Рисунок 11.44 Диалоговое окно Plugins....296
Рисунок 11.45 Окно вывода данных для выбранного соединения TCP....297
Рисунок 11.46 Диалоговое окно Decode As....297
Рисунок 11.47 Диалоговое окно Graph Control....297
Рисунок 11.48 График роста порядковых номеров TCP (формат Stevens)....298
Рисунок 11.49 График роста порядковых номеров TCP (формат tcptrace)....298
Рисунок 11.50 График зависимости потока данных через соединение TCP....299
Рисунок 11.51 График зависимости RTT от времени для соединения TCP....299
Рисунок 11.52 Диалоговое окно Summary....299
Рисунок 11.53 Диалоговое окно Protocol Hierarchy Statistics....300
Рисунок 11.54 Диалоговое окно генерации статистики....300
Рисунок 11.55 Статистика DHCP....300
Рисунок 11.56 Статистика H.225....300
Рисунок 11.57 Статистика HTTP....301
Рисунок 11.58 Статистика WAP-WSP....301
Рисунок 11.59 Диалоговое окно генерации статистики SRT....301
Рисунок 11.60 Статистика SRT для программ DCE-RPC....302
Рисунок 11.61 Статистика SRT для Fibre Channel....302
Рисунок 11.62 Статистика SRT для MGCP....302
Рисунок 11.63 Статистика SRT для ONC-RPC....302
Рисунок 11.64 Статистика SRT для SMB....303
Рисунок 11.65 Статистика трафика между парами хостов IP....303
Рисунок 11.66 Статистика сбора кадров....304
Рисунок 11.67 Статистика RTT для программ ONC-RPC....305
Рисунок 11.68 Диалоговое окно RTP Streams....305
Рисунок 11.69 Диалоговое окно RTP Stream Analysis....305
Рисунок 11.70 Диалоговое окно справочной системы Ethereal....306
Рисунок 11.71 Страница Printing....307
Рисунок 11.72 Страница Columns....307
Рисунок 11.73 Страница TCP Streams....308
Рисунок 11.74 Страница User Interface....308
Рисунок 11.75 Страница Capture....309
Рисунок 11.76 Диалоговое окно Inreface Options....309
Рисунок 11.77 Страница Name Resolutions....310
Рисунок 11.78 Страница Protocols....310
Рисунок 11.79 Вывод программы Tethereal....314
Рисунок 11.80 Программа IPtraf....316
Рисунок 11.81 Режим IP Traffic Monitor....317
Рисунок 11.82 Режим General Interface Statistics....321
Рисунок 11.83 Режим Detailed Interface Statistics....322
Рисунок 11.84 Статистика по размерам пакетов....323
Рисунок 11.85 Статистика по номерам портов....323
Рисунок 11.86 Режим LAN station monitor....324
Рисунок 11.87 Диалог выбора параметров фильтра TCP....325
Рисунок 11.88 Диалог создания фильтров для прочих протоколов IP....326
Рисунок 11.89 Диалог добавления портов....327
Рисунок 11.90 Интерфейс программы Etherape....329
Рисунок 11.91 Статистика MRTG....330
Рисунок 11.92 Сценарий для работы с pktgen....331
Рисунок 11.93 Интерфейс выбора параметров пакетов программы packETH....338
Рисунок 11.94 Интерфейс генерации однотипных пакетов....339
Рисунок 11.95 Интерфейс режима генерации последовательностей пакетов....339
Рисунок 11.96 Результат сканирования TCP SYN....347

Рисунок 11.97 Результат сканирования Stealth FIN....347
Рисунок 11.98 Проверка подставного хоста....349
Рисунок 11.99 Интерфейс Webmin для сканера nmap....355
Рисунок 11.100 Интерфейс nmapfe....356
Рисунок 12.1 Формат файла /proc/net/dev....368
Рисунок 12.2 Содержимое файла /proc/net/ip_conntrack....369
Рисунок 12.3 Таблица сокетов UDP в файле /proc/net/udp....369

Список таблиц

Таблица 1 Разделы файловой системы Linux....30
Таблица 2 Поля учетных записей пользователей....35
Таблица 3 Поля записей файла /etc/shadow....40
Таблица 4 Формат представления прав доступа....41
Таблица 5 Восьмеричные маски прав доступа....41
Таблица 6. Восьмеричное представление возможных комбинаций прав доступа....42
Таблица 7. Состояния сетевых соединений Linux....108
Таблица 8 Этапы обработки пакетов, адресованных локальному хосту....109
Таблица 9. Этапы обработки пакетов, сгенерированных локальным хостом....110
Таблица 10. Этапы обработки пересылаемых маршрутизатором пакетов....110
Таблица 11: Типы адресов сетевого уровня....133
Таблица 12 Опции radiusd....168
Таблица 13 Опции radclient....173
Таблица 14. Опции radtest....174
Таблица 15. Опции radwho....174
Таблица 16. Опции radzap....175
Таблица 17 Способы хранения учетных данных пользователей в GNU Radius....176
Таблица 18 Способы хранения учетных данных GNU Radius....176
Таблица 19 Опции команды id....192
Таблица 20 Опции команды kill....193
Таблица 21 Опции команды md5sum....193
Таблица 22 Опции команды su....194
Таблица 23 Опции команды uname....194
Таблица 24. Опции команды who....195
Таблица 25 Опции команды arp....195
Таблица 26 Опции команды ifconfig....199
Таблица 27. Опции команды mii-tool....200
Таблица 28 Поля вывода информации об активных соединениях....202
Таблица 29 Поля информации о сокетах UNIX domain....203
Таблица 30. Поля вывода команды route....207
Таблица 31 Опции команды dmesg....208
Таблица 32 Опции команды ethtool....208
Таблица 33 Опции команды free....210
Таблица 34 Опции команды ifplugstatus....211
Таблица 35 Опции pgrep и pkill....211
Таблица 36 Стандартные указатели формата ps....217
Таблица 37 Переменные окружения, используемые ps....218
Таблица 38 Ключи трактовки параметров ps....218
Таблица 39 Опции pstree....219
Таблица 40 Параметры команды sysctl....220
Таблица 41. Используемые по умолчанию параметры top....221
Таблица 42 Опции команды top....221
Таблица 43 Поля вывода программы top....222
Таблица 44 Глобальные команды интерактивного режима top....224
Таблица 45 Команды интерактивного управления для системной области....225
Таблица 46 Команды интерактивного управления для представления списка задач....225
Таблица 47 Команды интерактивного управления содержимым списка задач....225
Таблица 48. Команды интерактивного управления размером окна списка задач....226
Таблица 49. Старые команды управления сортировкой списка top....226
Таблица 50 Команды перемещения ключа сортировки списка задач....226
Таблица 51 Команды интерактивного управления сортировкой списка задач....226
Таблица 52. Опции команды vmstat....228
Таблица 53. Опции команды w....231
Таблица 54 Опции nessusd....233
Таблица 55 Параметры конфигурации в файле nessusd.conf....233
Таблица 56 Каталоги пользователей nessusd....235
Таблица 57 Опции nessusd....236
Таблица 58 Опции управления пакетным режимом....236
Таблица 59. Опции chkrootkit....242
Таблица 60 Опции команд samhain/yule....247
Таблица 61. Сигналы программ samhain/yule....248
Таблица 62 Переменные конфигурации PortSentry....254
Таблица 63. Опции sniffdet....258
Таблица 64 Опции командной строки tcpdump....266
Таблица 65 Примитивы фильтров tcpdump....270
Таблица 66 Примеры фильтров tcpdump....274
Таблица 67. Структура заголовка TCP....277
Таблица. 68 Биты флагов TCP....277
Таблица 69 Кнопки диалогового окна выбора фильтров....290
Таблица 70 Кнопки диалогового окна Apply Color Filters....294
Таблица 71 Кнопки управления колонками списка пакетов Ethereal....307
Таблица 72 Кнопки страницы Columns диалогового окна Preferences....307
Таблица 73 Операции в фильтрах отображения....311

Таблица 74 Типы полей в фильтрах отображения....311
Таблица 75 Опции команды editcap....314
Таблица 76 Опции команды mergecap....315
Таблица 77 Опции команды iptraf....316
Таблица 78 Поля вывода в режиме IP Traffic Monitor....318
Таблица 79 Флаги TCP в режиме IP Traffic Monitor....318
Таблица 80 Цветовая маркировка пакетов....319
Таблица 81 Типы сообщений ICMP....320
Таблица 82 Коды причин сообщений destination unreachable....320
Таблица 83 Типы сообщений OSPF....321
Таблица 84. Опции команды mrtg....330
Таблица 85 Параметры генерации пакетов....331
Таблица 86 Опции hping2 общего назначения....333
Таблица 87 Опции hping2 для выбора протокола....333
Таблица 88 Опции hping2 для протокола IP....334
Таблица 89 Опции hping2 для протокола ICMP....335
Таблица 90 Опции hping2 для протоколов TCP/UDP....335
Таблица 91 Опции hping2 для всех протоколов....336
Таблица 92 Опции режима capture....340
Таблица 93 Опции packit для режимов генерации и трассировки....340
Таблица 94 Опции заголовков IP....341
Таблица 95 Опции заголовков TCP....341
Таблица 96 Опции заголовков UDP....342
Таблица 97 Опции заголовков ICMP....342
Таблица 98 Опции запросов и откликов ICMP ECHO....342
Таблица 99 Опции откликов ICMP UNREACHABLE/REDIRECT/TIME EXCEEDED....342
Таблица 100 Опции запросов и откликов ICMP MASK....342
Таблица 101 Опции запросов и откликов ICMP MASK....343
Таблица 102 Типы и коды сообщений ICMP....343
Таблица 103 Опции генерации заголовков ARP....344
Таблица 104 Опции генерации заголовков Ethernet....344
Таблица 105 Опции команды nmap....350
Таблица 106 Варианты политики синхронизации nmap....354
Таблица 107 Опции синхронизации....354
Таблица 108 Стандартные классы символов....358
Таблица 109. Флаги компиляции regcomp....359
Таблица 110 Коды ошибок regcomp....360
Таблица 111 Параметры ядра и системы в целом....361
Таблица 112 Файлы в подкаталогах процессов....363
Таблица 113 Поля файлов /proc/*/stat....364
Таблица 114 Файлы каталога /proc/net....366
Таблица 115 Поля описания формата бинарных файлов....370
Таблица 116 Файлы каталога /proc/sys/fs....371
Таблица 117. Файлы каталога /proc/sys/kernel....372
Таблица 118 Опции сокетов....384
Таблица 119. Параметры SysCtl для сокетов....386
Таблица 120 Операции IOCTL для сокетов....387
Таблица 121 Опции сокетов IP (SOL_IP)....388
Таблица 122. Коды ошибок протокола IP....392
Таблица 123 Опции сокета TCP (SOL_TCP)....394
Таблица 124 Операции IOCTL для протокола TCP....394
Таблица 125 Коды ошибок протокола IP....395
Таблица 126 Операции IOCTL для протокола UDP....396
Таблица 127 Сообщения об ошибках для протокола UDP....396
Таблица 128 Изменение полей заголовка IP для пакетов RAW....397
Таблица 129 Коды ошибок для сокетов raw....398
Таблица 130 Коды ошибок для пакетных сокетов....400
Таблица 131 Макросы netlink....400
Таблица 132 Члены семейства netlink....401
Таблица 133 Флаги сообщений netlink....402
Таблица 134 Флаги ARP....403
Таблица 135 Параметры sysctl для ARP....403
Таблица 136 Операции IOCTL для netdevice....405
Таблица 137 Флаги устройств для netdevice....406
Таблица 138: Коды ошибок IOCTL....407
Таблица 139 Поля структуры sk_buff....410
Таблица 140. Опции modprobe....414
Таблица 141. Опции depmod....416
Таблица 142 Опции rmmod....417
Таблица 143 Глобальные параметры загрузчика LILO....422
Таблица 144 Частные параметры LILO....428
Таблица 145 Частные параметры LILO для загрузки других ОС....428
Таблица 146 Параметры ядра, передаваемые через LILO....430
Таблица 147 Конфигурационные параметры соединений ipsec....432
Таблица 148 Параметры автоматической генерации ключей....433
Таблица 149 Параметры ручной генерации ключей....434

Курс предназначен для администраторов безопасности и сетевых администраторов. Основной задачей курса является рассмотрение возможности реализации систем обеспечения информационной безопасности хостов и сетей на базе программных средств с открытым исходным кодом. Курс содержит рекомендации по выбору и развертыванию систем обеспечения безопасности, основанные на реальном опыте тестирования и эксплуатации подобных систем на базе операционной системы Linux в реальных условиях корпоративных сетей и сайтов Internet. Достаточно большое внимание уделяется вопросам создания эффективной системы фильтрации на уровне пакетов, как одной из основных компонент систем обеспечения безопасности. Приводится обзор средств организации и поддержки соединений VPN. Кроме того, рассматриваются вопросы, связанные с фильтрацией незапрошенной электронной почты и вирусов на граничных узлах сети. Отдельно рассматриваются программные средства, позволяющие администратору обеспечить эффективный мониторинг сети и обнаружение потенциальных брешей в системе безопасности граничных узлов, сетевых серверов и пользовательских компьютеров.
Курс практически не содержит готовых решений и может лишь подтолкнуть думающих администраторов к поиску пути обеспечения безопасности с использованием открытых решений. Данная книга является справочным приложением к курсу и содержит краткий обзор ряда вопросов, связанных с обеспечением информационной безопасности.

1 Введение

В ИТ-инфраструктуре каждого современного предприятия вопросы обеспечения информационной безопасности имеют большое значение. Однако важность этих вопросов и уж тем более, необходимость соответствующих расходов осознают далеко не все руководители, принимающие решения. Зачастую приходиться наблюдать, что уже и гром гремит, а мужик все никак не перекрестится. Проблема антивирусной защиты, также входящая в сферу информационной безопасности, по-видимому проще и понятнее руководству компаний, поэтому на ее решение обычно находятся средства и силы. Нередко приходиться наблюдать, что ресурсов на этом направлении расходуется даже слишком много, но обсуждение этого вопроса уведет нас в сторону от темы курса.

Основными задачами каждого администратора безопасности являются обеспечение максимального уровня:

- конфиденциальности;
- целостности;
- доступности
    - внешних ресурсов из корпоративной сети;
    - публичных ресурсов сети из внешних сетей;
    - корпоративных ресурсов для удаленных пользователей корпоративной сети.

CERT/CC[1] рассматривает в качестве основных направлений деятельности команд по обеспечению информационной безопасности:

1) создание и поддержку корпоративной политики безопасности;
2) готовность к вторжениям в сеть;
3) обнаружение попыток вторжения;
4) отклик на инциденты;
5) повышение уровня информационной безопасности с учетом происходящих инцидентов.

В этой главе мы попытаемся в свете перечисленных выше задач и направлений деятельности увидеть сильные и слабые стороны систем обеспечения безопасности на базе программ с открытым исходным кодом.

Обычно программы с открытым кодом распространяются свободно на условиях лицензии GNU, поэтому мы будем называть их здесь для простоты свободными или бесплатными программами. Программы, которые продаются в виде законченного набора исполняемых модулей и конфигурационных файлов, не позволяющего пользователю вносить какие-либо изменения в исполняемый код, будем для краткости называть коммерческими.

1.1 Основные различия между коммерческими и открытыми программами

1.1.1 Цели создания программ

Компании или отдельные разработчики создают коммерческие программы с целью зарабатывания денег на продаже и обслуживании этих программ. Несомненно, коммерческие программы служат для решения той или иной задачи (иначе бы их просто не покупали), но это все-таки вторично - основной целью разработки коммерческих приложений является получение **прибыли компанией производителем**.

Поскольку в реальной жизни пользователь зачастую покупает «кота в мешке», не имея возможности опробовать программу в «боевых» условиях, на широту распространения (популярность) коммерческих программ можно оказать существенное влияние за счет использования эффективных средств маркетинга и рекламы. Даже мимолетный взгляд на современный рынок программного обеспечения показывает, что маркетинговые рычаги на нем работают значительно надежнее, нежели качество выпускаемой на рынок продукции.

Бесплатные же программы создаются с **целью решения той или иной задачи**[2], поэтому популярность той или иной свободно распространяемой программы обусловлена прежде всего эффективностью решения задач, для которых эта программа создавалась. Никто не заставит меня использовать ненужную мне программу, если она не делает того, что я хочу. Отсутствие необходимости покупки программы до начала ее использования и широкий выбор бесплатных приложений позволяют каждому выбрать те бесплатные программы, которые наиболее точно соответствуют реальным потребностям.

1.1.2 Возможность адаптации программ

Коммерческие программы создаются с учетом использования в типовых ситуациях и зачастую достаточно плохо вписываются в конкретную инфраструктуру ИТ. Очевидно, что производитель программ не может адаптировать свою систему для каждого пользователя, а для обеспечения желаемого уровеня продаж и **получения прибыли** разработчики создают программы в расчете на усредненного пользователя.

Поскольку коммерческие программы поставляются в виде исполняемых модулей, возможности настройки программ

1 *Координационный центр информационной безопасности при университете Карнеги-Мэллона в США. Дополнительную информацию о деятельности этого центра вы сможете найти на его официальном сайте http://www.cert.org*

2 *Кто-то скажет, что движущими силами создания таких программ могут быть амбиции разработчиков и другие причины подобного типа, но никакие амбиции и желания разработчика не заставят использовать его программу, если ее никто не рекламирует и она не решает поставленных задач.*

пользователем ограничены теми пределами, которые соизволил открыть производитель. Как правило, это тот или иной набор конфигурационных опций, для выбора и настройки которых коммерческие программы обычно включают специальные средства. Конфигурационные параметры коммерческих программ чаще всего хранятся в бинарном формате, поэтому отрегулировать что-либо сверх того, что делают программы настройки из комплекта поставки обычно не представляется возможным.

Открытые программы распространяются обычно в исходных кодах[1], что предоставляет каждому квалифицированному пользователю вносить в программы изменения в соответствии с реальными условиями использования программ. Конфигурационные параметры свободных программ обычно хранятся в текстовых файлах, содержащих комментарии, достаточные для самостоятельного выбора параметров и тонкой настройки программ.

Очевидно, что современные приложения, имеющие множество параметров можно настроить так, что они просто не будут работать или будут это делать неоптимально. На первый взгляд, представляется, что коммерческие программы с приложенными средствами настройки решают эту проблему, не позволяя пользователям создавать логически противоречивые или неработоспособные наборы конфигурационных параметров. Однако практика показывает, что разработчики не всегда могут обеспечить должный уровень проверки конфигурационных параметров на непротиворечивость. Кто из вас не сталкивался хотя бы раз с тем, что программа переставала работать после внесения незначительных изменений в ее настройки. Говорить про оптимизацию конфигурационных параметров коммерческих программ в большинстве случаев просто не приходится, да и проверить это почти не реально, поскольку параметры надежно спрятаны в бинарных файлах недокументированного формата.

Используемые в открытых программах конфигурационные файлы в тестовом формате позволяют легко создавать резервные копии конфигурации и при возникновении той или иной проблемы находить вызвавшие ее причины путем простого сравнения тестовых файлов. Более того, интуитивно понятные и хорошо документированные конфигурационные файлы открытых программ зачастую позволяю вручную оптимизировать конфигурацию, что обеспечивает наиболее эффективную работу программ.

1.1.3 Уязвимость программ

Программы создаются людьми, а люди нередко ошибаются. Поэтому в программах всегда присутствуют те или иные уязвимости, которыми не преминут воспользоваться злоумышленники или те, кто не отдает себе отчета о возможных последствиях. Вероятность наличия программных уязвимостей достаточно велика как для коммерческих программ, так и для открытых. Однако наличие исходного кода и широкий круг разработчиков открытых систем обеспечивает более быстрое обнаружение уязвимостей. Кроме того, производители коммерческих программ зачастую совершенно не заинтересованы обнародовать информацию о наличии в их программах слабых мест.

Существует несколько организаций, поддерживающих базы данных о программных уязвимостях с рекомендациями по их устранению:

- CERT (http://www.cert.org/)
- FIRST (http://www.first.org/)
- AUSCERT (http://www.auscert.org.au/)
- CERIAS (http://www.cerias.purdue.edu/)
- CIAC (http://ciac.llnl.gov/ciac/)
- FedCIRC (http://www.fedcirc.gov/)
- NIST CSRC (http://csrc.nist.gov/)
- SANS (http://www.sans.org)
- USENIX (http://www.usenix.org)

1.1.3.1 Устранение уязвимостей

При обнаружении уязвимости в программе и публикации сведений об этом обычно резко возрастает число попыток использования слабых мест для организации атак или несанкционированного доступа в систему.

В случаях использования коммерческих программ вам придется ждать, пока производитель сочтет нужным потратить время и средства на внесение исправлений в программный код. Этот процесс зачастую может тянуться достаточно долго. Удивляться этому не следует, ведь производитель программ в соответствии с лицензионным соглашением не несет никакой ответственности за результаты использования программ[2].

При использовании открытых программ ошибки как правило исправляются гораздо быстрее, поскольку доступ к исходным кодам имеет большое число квалифицированных программистов.

1 *Как правило, такие приложения сопровождаются и бинарными дистрибутивами, подготовленными для использования в “стандартных” условиях.*
2 *Как показывает опыт, даже необходимость нести ответственность за результаты использования продукции не всегда заставляет производителей исправлять собственные ошибки. Например, компания Боинг, осведомленная о дефектах в хвостовом оперении своих самолетов, предпочитает выплачивать компенсации семьям погибших, потому что расходы на исправление будут существенно больше.*

1.1.4 Оптимизация работы систем обеспечения безопасности

В коммерческих системах обеспечения безопасности политика безопасности обычно хранится в том или ином закрытом бинарном формате и администратор безопасности просто не ведает как она выполняется, довольствуясь лишь результатами ее реализации. Поскольку коммерческие программы рассчитаны прежде всего на типовые (массовые) варианты применения, надеяться на оптимальную реализацию правил выбранной администратором политики в таких случаях просто не приходится. То что оптимально для решения ваших задач, может оказаться совсем неудобным для вашего коллеги из другой компании, поэтому разработчики оптимизируют системы для безликого массового потребителя, у которого не предполагается сколь-нибудь нетривиальных задач.

Закрытый формат хранения политики не позволяет создавать средств оптимизации самим пользователям или третьим фирмам.

В открытых системах администратор практически всегда может сохранить политику безопасности в понятном человеку текстовом формате и при необходимости оптимизировать набор правил вручную или с помощью программ оптимизации, которые могут быть созданы самостоятельно или заказаны профессиональным программистам.

1.1.5 Ответственность разработчиков

Как в открытых, так и в коммерческих приложениях приняты лицензионные соглашения, оговаривающие ответственность разработчиков за ошибки в программах и т. п. Если вы почитаете эти соглашения, то поймете, что даже в случае покупки коммерческой программы вы не сможете предъявить ее разработчику никаких претензий за исключением требования замены поврежденных носителей в течение весьма ограниченного срока после покупки программы[1].

Очевидно, что в любом случае вся ответственность за результаты работы программ ложится на пользователя. Но в случае открытых программ вы по крайней мере имеете возможность заглянуть в исходный код и даже внести изменения в него, т. е., ваша ответственность подкреплена правом инспекции кода и возможностью самостоятельного решения вопросов об использовании тех или иных модулей. Случай же с коммерческими программами лучше всего описывается поговоркой “без меня меня женили”. Вам просто дают “черный ящик”, потребовав взамен денег и даже не пообещав, что этот ящик будет хоть как-то работать.

Плохо, когда возникают сбои в работе офисных приложений, но во сто крат хуже, когда из-за недосмотра или по умыслу разработчиков купленная вами система обеспечения безопасности может в любой момент отказать и вы даже не сможете никому предъявить претензий. Представьте себе, что солдату дали бронежилет, в котором за красивым внешним покрытием спрятаны дыры в броне. Да еще и претензии предъявлять запретили. Во и воюй с таким бронежилетом.

Дыры могут быть и в бесплатном бронежилете, но их по крайней мере сложней спрятать, поскольку у него внешнее покрытие съемное.

1.1.6 Возможность разработки собственных средств

В коммерческий системах пользователю предоставляются обычно только интерфейсы API (не всегда бесплатные), а системы с открытым кодом позволяют самостоятельно изменять любое приложение, создавать дополнительные модули, утилиты и т. п.

1 *Обычно этот срок составляет 90 дней.*

2 Безопасность хостов Linux

One should not increase, beyond what is necessary, the number of entities required to explain anything[1]

Occam

В этой главе кратко рассматриваются способы повышения уровня безопасности хостов на базе Linux в различных вариантах их использования (пограничные узлы с поддержкой маршрутизации и межсетевых экранов, сайты Internet, пользовательские рабочие станции). Приводятся рекомендации по выбору служб, активизируемых на хостах Linux с учетом использования этих хостов. Рассматриваются вопросы обеспечения безопасности файловой системы, отдельных служб (сервиса), резервирования и т. п. Использованию хостов Linux в качестве межсетевых экранов и маршрутизаторов посвящена глава 4.

2.1 Инсталляция ОС, организация файловой системы, управление пользователями

В отличие от ОС Windows операционные системы UNIX (включая Linux) предоставляют пользователю существенно более широкие возможности выбора опций установки, организации файловых систем, управления доступом пользователей к системным ресурсам и т. п. В последующих параграфах рассматриваются варианты установки Linux с учетом предполагаемого использования компьютера и обеспечения оптимального уровня безопасности и производительности системы.

Во многих современных вариантах Linux при инсталляции пользователю предлагается выбрать модель безопасности для данного хоста. Очевидно, что выбор модели безопасности определяется будущим использованием хоста и способом его подключения к сети. В общем случае при инсталляции рекомендуется устанавливать средний (или более низкий) уровень, поскольку при более высоком уровне на первых этапах работы с хостом Linux вы можете столкнуться с проблемами, которые начинающему пользователю просто покажутся неразрешимыми. Впоследствии вы всегда сможете выбрать желаемый уровень безопасности для своего хоста.

Основная идея этого раздела выражена в эпиграфе. Применительно к нашей ситуации слова Оккама можно трактовать как рекомендацию устанавливать на компьютере только те приложения и службы, которые будут реально использоваться в процессе работы с данным компьютером. Если вы хотите поэкспериментировать с той или иной программой, используйте для этого отдельный компьютер, на котором не будет храниться никакой важной информации и не будет выполняться критичных для бизнеса приложений. Эксперименты на “боевой” машине могут оказаться слишком дорогим удовольствием.

2.1.1 Разбиение диска

В большинстве современных дистрибутивов Linux при установке предлагается автоматическое разбиение диска на разделы в соответствии с выбранным вариантом использования данного компьютера (сервер, рабочая станция и т. п.). В большинстве случаев предлагается достаточно оптимальный вариант разбиения диска и вы вполне можете с ним согласиться. Если же вы планируете использовать данный компьютер для решения тех или иных специальных задач, целесообразно будет разбить диск на разделы в соответствии с реальными потребностями. Разбиение диска для Linux-машины, которую планируется использовать в качестве маршрутизатора/межсетевого экрана, подробно обсуждается в параграфе 4.1.1 (стр. 58).

Никогда не устанавливайте Linux на один раздел, поскольку в этом случае уровень безопасности системы существенно снижается, поскольку злоумышленники или просто неразумные пользователи с помощью SUID-программ[2] могут получить доступ во все области диска. Последствия в таких случаях могут оказаться весьма печальными. Кроме того, установка системы на один раздел существенно усложняет администрирование системы, а при возникновении ошибок в файловой системе значительно повышается вероятность возникновения серьезных проблем в работе системы в целом вплоть до необходимости новой установки операционной системы. Усложняются также задачи резервного копирования и обновления программ. Целесообразно разбить диск по крайней мере на 5 разделов, перечисленных ниже.

Таблица 1 Разделы файловой системы Linux

Имя	Назначение раздела
/	Корневой раздел, обычно включающий основные компоненты операционной системы, библиотеки, временные файлы. Для современных вариантов Linux обычно вполне достаточно 300 - 500 Мбайт.
Область подкачки	Область подкачки используется для перебрасывания содержимого оперативной памяти на диск в целях освобождения памяти для активных приложений. Существует множество мнений относительно размеров области подкачки и в каждом из таких мнений есть доля истины. Очевидно, что размер области подкачки зависит от имеющейся в компьютере оперативной памяти и “прожорливости” используемых приложений. На мой взгляд, для современных компьютеров и приложений целесообразно создавать область подкачки размером 500 - 1000 Мбайт.
/usr	Этот раздел должен быть достаточно большим, поскольку в нем хранятся все пользовательские программы, библиотеки, файлы документации и т. п.

1 *“Не умножай сущностей сверх необходимого”. Оккам*
2 *Информацию о SUID-программах вы найдете в параграфе 2.3.3 (стр. 42)*

Имя	Назначение раздела
/var	В этом разделе хранятся журнальные файлы Linux, задания для печати, рабочие файлы электронной почты, базы данных и т. п., поэтому он также должен быть достаточно большим. Отметим, что при нехватке места для записи журнальных файлов могут возникнуть серьезные проблемы вплоть до остановки системы.
/home	В этом разделе хранятся пользовательские файлы, поэтому его размер определяется прежде всего числом пользователей данного компьютера и объемами хранимой ими информации.

Для разбиения диска на разделы используется программа fdisk или иные программы, включенные в используемый вами дистрибутив Linux[1].

2.1.1.1 fdisk

Программа fdisk представляет собой консольный инструмент для создания и управления дисковыми разделами Linux. Ниже кратко рассматривается процесс создания разделов, требуемых для работы Linux. Более подробное описание возможностей программы можно получить с помощью команды **man fdisk**. Для запуска программы введите команду

```
fdisk имя дискового устройства[2]
```

После загрузки программы на экран будет выведено сообщение

```
The number of cylinders for this disk is set to 1048.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):
```

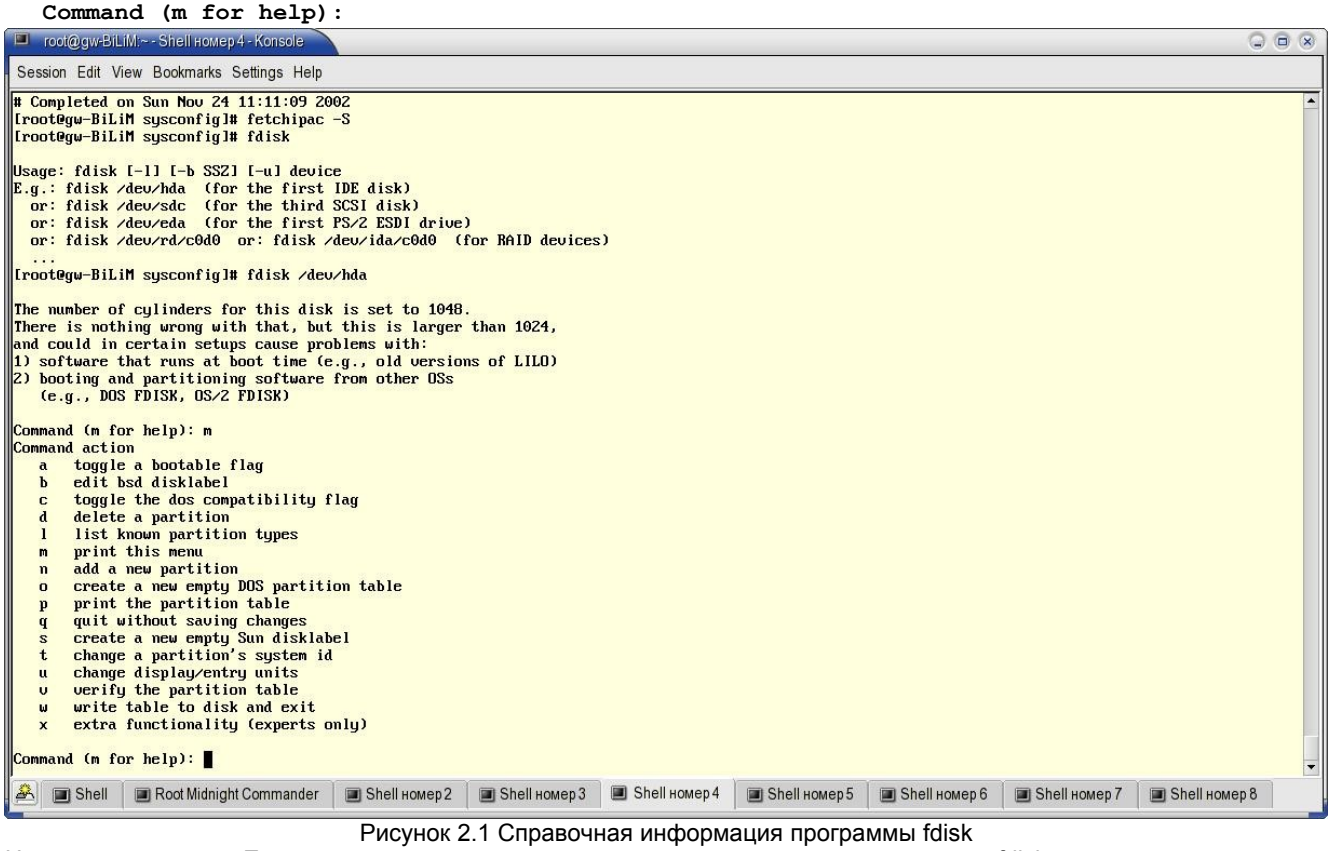


Рисунок 2.1 Справочная информация программы fdisk

Нажав клавишу **m** и **Enter**, вы получите на экране краткую справку о системе команд fdisk, показанную на рисунке 2.1. Краткое описание доступных команд приведено ниже.

a поменять флаг возможности загрузки для раздела;

b изменить раздел BSD на диске;

c поменять флаг DOS-совместимости;

d удалить раздел;

l вывести список поддерживаемых типов разделов;

m вывести справочную информацию;

1 *Чаще всего для этих целей используются программы cfdisk и DiskDruid (RedHat), работа с которыми еще проще, чем с программой fdisk.*

2 *В Linux-системах для дисковых устройств IDE/ATAPI обычно используются имена /dev/hda, /dev/hdb и т. д., для SCSI - /dev/sda, dev/sdb ...*

n добавить новый раздел;

o создать новую (пустую) таблицу разделов DOS;

p вывести информацию об имеющихся на диске разделах;

q завершить работу программы без сохранения результатов;

s создать новый (пустой) Sun-раздел на диске;

t изменить тип раздела;

u сменить единицу измерения (цилиндр или сектор);

v проверить корректность таблицы разделов;

w записать таблицу разделов и завершить работу программы;

x расширенные функции (только для специалистов).

Прежде, чем вносить какие-либо изменения, посмотрите таблицу имеющихся на диске разделов (команда **p**). Если на диске еще не создано ни одного раздела, на экран будет выведено сообщение

```
Disk /dev/hda: 255 heads, 63 sectors, 1048 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System

Command (m for help):
```

Убедившись в отсутствии разделов, можно приступать к их созданию. Приведенные ниже сведения описывают реальный процесс разбиения диска. При его повторении не забывайте указывать свои значения параметров создаваемых разделов.

2.1.1.1.1 Создание области подкачки и корневого раздела

Прежде всего на диске следует создать область подкачки (swap) и корневой раздел. Для создания нового раздела нажмите клавишу **n** и **Enter**. Программа в ответ на это попросит указать тип создаваемого раздела (основной или расширенный).

```
Command action
   e   extended
   p   primary partition (1-4)
p
```

Для создания основного раздела нажмите клавишу **p** и **Enter**. Программа запросит номер нового раздела. Поскольку новый раздел является первым на диске, для него следует указать номер 1.

```
Partition number (1-4): 1
```

Далее программа запросит номер первого цилиндра для создаваемого раздела. Первый раздел целесообразно разместить в начале диска, поэтому нажмите клавишу 1 и **Enter**.

```
First cylinder:  (1-1048)  1
```

Следующий вопрос будет задан относительно размера нового раздела. Укажите номер последнего цилиндра для раздела или размер раздела в мегабайтах. В нашем примере будет создана область подкачки размером 512 Мбайт.

```
Last cylinder or +size or +sizeM or sizeK (1-1048):  +512M
```

Созданный раздел будет иметь тип Linux native и нам нужно изменить тип раздела, поскольку мы создаем область подкачки. Нажмите клавишу **t** и **Enter**. Программа fdisk запросит номер раздела для изменения типа (в нашем случае нужно указать раздел 1). После указания раздела на экране появится запрос на выбор нового типа для данного раздела. Для областей подкачки Linux использует тип 82.

```
Hex Code (L to list): 82
```

После этого целесообразно посмотреть таблицу с помощью команды **p**. Убедившись в наличии на диске раздела для области подкачки, можно переходить к созданию корневого раздела. Повторите описанные выше процедуры, указав в качестве номера раздела 2. Отметим, что для корневого раздела менять тип не требуется. Однако для этого раздела следует установить флаг загрузки с помощью команды **a**.

2.1.1.1.2 Создание расширенного раздела

После создания корневого раздела и области подкачки можно создать расширенный раздел, включающий все оставшееся на диске пространство. Для создания нового раздела нажмите клавишу **n** и **Enter**. Программа в ответ на это попросит указать тип создаваемого раздела (основной или расширенный).

```
Command action
   e   extended
   p   primary partition (1-4)
e
```

Для создания расширенного раздела нажмите клавишу **e** и **Enter**. Программа запросит номер первого цилиндра расширенного раздела, указав в качестве принятого по умолчанию значения номер первого свободного цилиндра на диске. В ответ на этот запрос достаточно просто нажать клавишу **Enter**. После выбора стартового цилиндра на экране появится уже знакомый вам запрос размера нового раздела.

```
Last cylinder or +size or +sizeM or sizeK (131-1048):  1048
```

Вы можете просто нажать клавишу для создания раздела, занимающего все свободное пространство диска.

Если вы после этого воспользуетесь командой **p**, на экран будет выведен список созданных разделов

```
Disk /dev/hda: 255 heads, 63 sectors, 1048 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hda1             1        96    771120   82  Linux swap
/dev/hda2   *        97       199    819283+  83  Linux
/dev/hda3           200      1048   6827894   83  extended
```

2.1.1.1.2.1 Создание логических дисков в расширенном разделе

После создания расширенного раздела в нем следует создать логические диски для /usr, /var и /home. Для создания логического диска в расширенном разделе Linux нажмите клавиши **n** и **Enter**. На экране появится запрос типа создаваемого раздела

```
Command action
   l   logical (5 or over)
   p   primary partition (1-4)
l
```

Нажмите **l** для создания логического диска и в ответ на запрос номера раздела нажмите клавишу 5 и **Enter**. Программа предложит вам уже знакомые процедуры выбора стартового цилиндра и размера. Укажите подходящие для вашей задачи значения и повторите процедуру создания логических дисков для всех точек монтирования.

После завершения работы по созданию логических разделов еще раз проверьте корректность созданных на диске разделов (команда p) и нажмите клавиши **w** и **Enter** для записи созданной таблицы на диск и завершения работы программы.

2.1.1.2 cfdisk

Построенная на базе меню программа cfdisk в работе еще проще, чем fdisk, поэтому мы не будем рассматривать работу с этой программой, ограничившись лишь рисунком, показывающим основное меню.

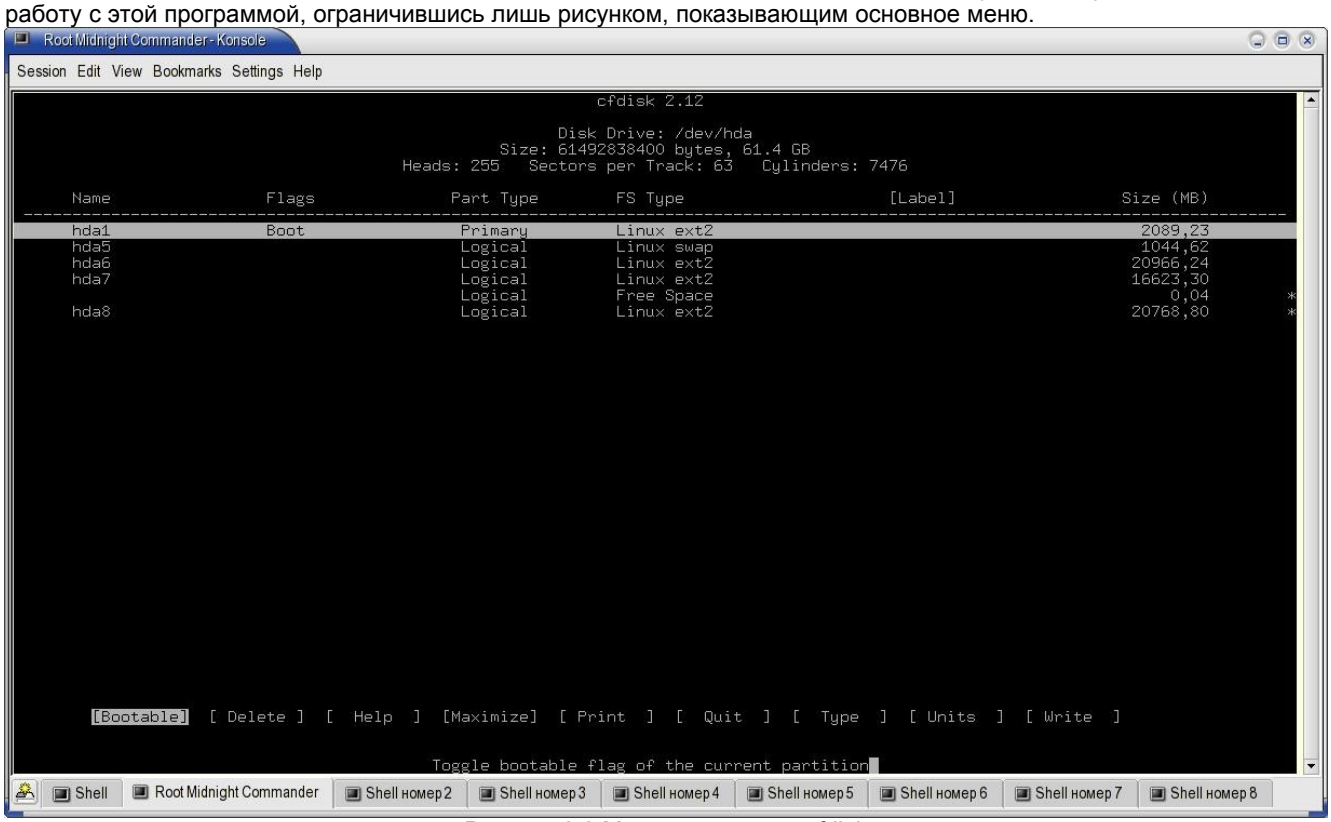


Рисунок 2.2 Меню программы cfdisk

2.1.1.3 DiskDruid

Программа DiskDruid, используемая в основном системами RedHat Linux имеет еще более дружественный интерфейс (см. рисунок 2.3) и в отличие от программ fdisk и cfdisk сразу связывает дисковые разделы с точками монтирования (/, /usr, /var и т. д.). Мы не будем тратить время на изучение этой программы, поскольку она проста и понятна даже для новичков.

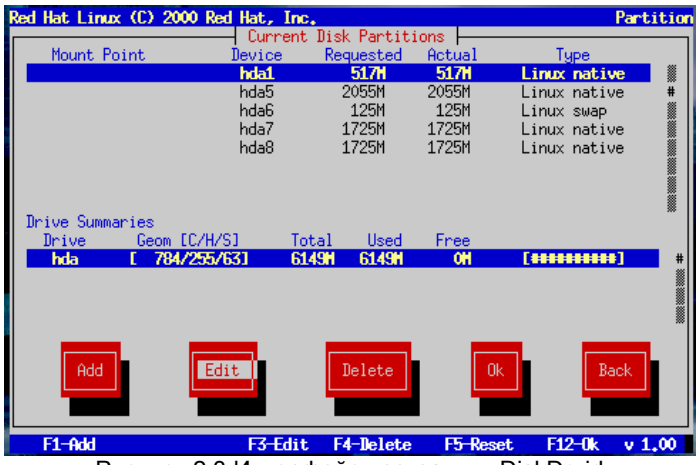


Рисунок 2.3 Интерфейс программы DiskDruid

2.1.2 Менеджер загрузки

После завершения работы по созданию разделов программа инсталляции обычно переходит к выбору менеджера загрузки. В современных дистрибутивах Linux обычно для управления менеджерами загрузки используются программы **lilo**[1] или **GRUB**. Менеджер загрузки представляет собой небольшую программу, размещаемую в MBR или первом секторе корневого раздела (см. рисунок 2.4), которая позволяет выбрать загружаемую операционную систему или ее вариант.

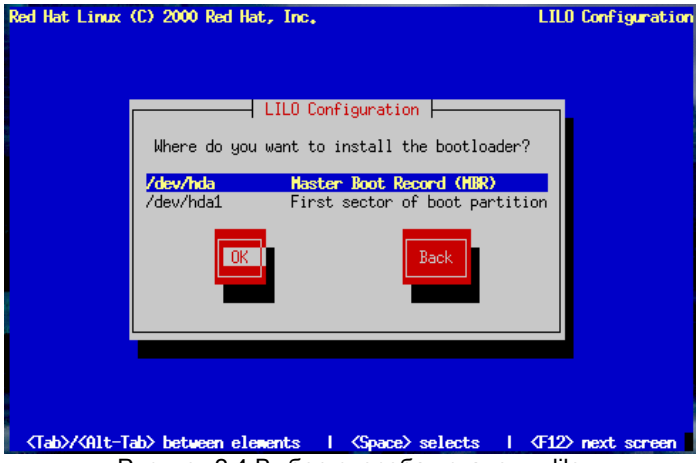


Рисунок 2.4 Выбор способа установки lilo

2.1.2.1 Конфигурация lilo

Программа lilo, служащая для установки загрузчика операционной системы, считывает параметры из конфигурационного файла **/etc/lilo.conf**. В этом файле пользователь может задать, наряду с прочими параметрами, пароль для доступа к меню выбора варианта загрузки, что может создать дополнительные сложности при попытках несанкционированного физического доступа к компьютеру.

Рассмотрим основные параметры, хранящиеся в файле lilo.conf. Ниже приведен фрагмент этого файла с моей рабочей станции.

```
# WARNING: do not forget to run lilo after modifying this file

boot=/dev/hda
map=/boot/map
vga=normal
default="266"
keytable=/boot/us-latin1.klt
prompt
nowarn
timeout=50
message=/boot/message
menu-scheme=wb:bw:wb:bw

image=/boot/vmlinuz
       label="linux"
       root=/dev/hda1
       initrd=/boot/initrd.img
       append="quiet devfs=mount acpi=ht resume=/dev/hda5 splash=silent"
       vga=788
       read-only
image=/boot/vmlinuz
       label="linux-nonfb"
```

1 *Linux Loader - загрузчик Linux.*

```
        root=/dev/hda1
        initrd=/boot/initrd.img
        append="quiet devfs=mount splash=silent acpi=ht resume=/dev/hda5"
        read-only
  image=/boot/vmlinuz-2.6.6
        label=266
        root=/dev/hda1
        read-only
        optional
        vga=normal
        append="quiet devfs=mount acpi=ht resume=/dev/hda5 splash=silent"
        initrd=/boot/initrd-2.6.6.img
```

Параметры загрузчика делятся на две группы - глобальные, которые относятся ко всем вариантам загрузки, и частные, применяемые отдельно для каждого варианта загрузки ОС. Подробное описание параметров конфигурации LILO приведено в приложении 12.18 (стр. 413).

2.2 Управление учетными записями

Для доступа пользователя в систему UNIX этот пользователь должен пройти процедуру регистрации, при которой проверяются параметры учетной записи пользователя и устанавливается уровень привилегий. Учетную запись имеет каждый пользователь и именно эта запись определяет всю дальнейшую работу пользователя. Для управления учетными записями пользователей Linux служит набор утилит с развитым командным интерфейсом. Программы **useradd** (параграф 2.2.2 на стр. 35) и **groupadd** (параграф 2.2.3 на стр. 37) служат для добавления учетных записей пользователей и групп, а программы **userdel**/**groupdel** и **usermov**/**groupmov** для удаления и редактирования учетных записей.

Если вас не устраивает по тем или иным причинам консольные программы, вы можете воспользоваться какой-либо из программ с более развитым интерфейсом. Обычно в каждом дистрибутиве имеется по крайней мере одна программа с графическим или текстовым (меню) интерфейсом для создания и редактирования пользовательских учетных записей. На рисунке 2.6 показан Web-интерфейс модуля добавления (редактирования) пользователей программы Webmin, описанной ниже (раздел 11.3 на стр. 228).

Прежде, чем перейти к вопросам создания и управления учетными записями, рассмотрим структуру таких записей.

2.2.1 Структура учетных записей

Учетные записи всех пользователей системы хранятся в специальном файле **/etc/passwd**. Пример такого файла показан на рисунке 2.5. Как вы можете видеть, каждая учетная запись состоит из 7 полей (таблица 2), разделенных двоеточием (**:**). Если поле необязательно использовать, оно указывается пустым значением.

Таблица 2 Поля учетных записей пользователей

Поле	Назначение
Имя пользователя	Первое поле каждой учетной записи содержит регистрационное имя пользователя в системе. Часто регистрационные имена создают на основе реальных имен пользователей (например, первая буква имени и фамилия). Такой подход упрощает работу администратора, но позволяет злоумышленникам сделать некоторые предположения на основе полученной информации. Размер имени пользователей в некоторых системах ограничен 8 символами. Регистр символов обычно не различается.
Пароль или метка пароля	Это поле в старых системах содержало зашифрованный пароль пользователя. В разных версиях UNIX используются различные варианты шифрования пароля, но возможность доступа к файлу **/etc/passwd** позволяет злоумышленникам скопировать файл и подобрать пароль, используя специальные программы. В современных системах данное поле содержит лишь метку пароля (**x**), а зашифрованные пароли хранятся в специальном файле **/etc/shadow** с ограниченным доступом (см. параграф 2.2.4 на стр. 38).
Идентификатор пользователя (UID)	В этом поле хранится числовой идентификатор пользователя (UID), который служит для отслеживания связи с запущенными данным пользователем процессами. Для идентификаторов используется диапазон значений от 0 до 65535. Значения от 0 до 99 обычно резервируются для системных имен, а оставшиеся номера администратор может распределять по своему разумению. Целесообразно выделить для пользователей диапазон идентификаторов и последовательно выдавать значения из этого диапазона. Такой подход позволяет администратору с первого взгляда определить по списку процессов кто эти процессы использует и сразу же увидеть появление в системе чужеродных процессов. Многие современные дистрибутивы Linux автоматически выделяют для новых пользователей последовательные значения из диапазона чисел, превышающих 500.
Идентификатор группы (GID)	Числовой идентификатор первичной (основной) группы, к которой относится пользователь. В различных вариантах Linux выбор первичной группы для нового пользователя осуществляется с использованием различных подходов[1].

Поле	Назначение
Реальное имя пользователя	Это необязательное поле, обычно обозначаемое аббревиатурой GECOS[2], может содержать достаточно много информации, в том числе реальное имя пользователя. Некоторые ОС при пропуске данного поля автоматически включают в него то или иное значение.
Начальный (домашний) каталог	Это поле указывает стартовый каталог, в который пользователь попадает после регистрации системы. Обычно для таких каталогов используется дерево **/home**, которое целесообразно монтировать в отдельный раздел диска (см. параграф 2.1.1 на стр. 29).
Командный процессор	Это поле указывает стандартный командный процессор (интерпретатор команд или shell), который предоставляется пользователю сразу же после регистрации. В зависимости от опций установки вашего дистрибутива могут быть доступны разные варианты командных интерпретаторов. Отметим, что при выборе устанавливаемых в системе командных процессоров не следует отходить от принципа Оккама. Чем больше программ (а командный процессор является программой), тем больше шансов вы даете злоумышленникам для взлома.

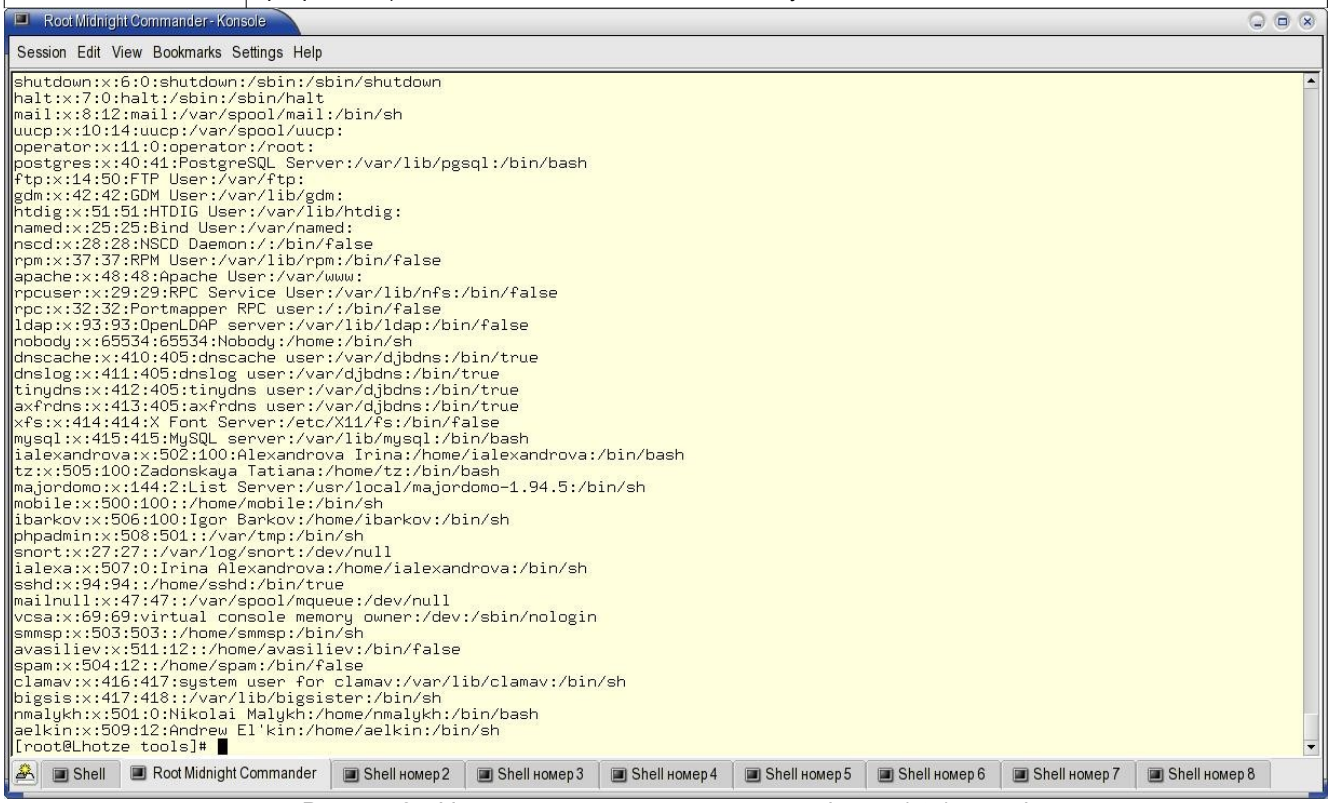


Рисунок 2.5 Учетные записи пользователей в файле /etc/passwd

Файл **/etc/group** имеет аналогичную структуру и схожие по смыслу поля, поэтому мы не будем останавливаться на его рассмотрении.

2.2.2 Пользователи

Для добавления пользователей в Linux существует специальная команда **useradd**, которая может также изменять параметры, заданные ранее для существующих пользователей.

2.2.2.1 Синтаксис

```
useradd [-c comment] [-d home_dir] [-e expire_date] [-f inactive_time] [-g initial_group]
        [-G group[,...]] [-m [-k skeleton_dir] " |" " " -M ]] [-o] [-p passwd]
        [-s shell] [-u uid] [-N] [-r]login


useradd -D [-g default_group] [-b default_home] [-e default_expire_date]
        [-f default_inactive] [-s default_shell]
```

2.2.2.2 Добавление пользователей

При вводе команды

```
useradd
```

без опции **-D** создается учетная запись (user account) для нового пользователя с учетом других указанных в команде опций и принятых по умолчанию параметров. Новый пользователь включается в соответствующие

1 *Некоторые системы всех новых пользователей помещают автоматически в группу **users**, а иные создают для пользователя одноименную группу.*
2 *General Electric Comprehensive Operating System*

системные файлы, для него создается домашний каталог с начальным набором файлов. Применимые для создания новых учетных записей опции перечислены ниже.

-c <комментарий>
задает комментарии для учетной записи нового пользователя.

-d <домашний каталог>
указывает домашний каталог, который будет создан для нового пользователя. По умолчанию создается каталог, имя которого совпадает с регистрационным именем пользователя, который размещается в принятом по умолчанию пользовательском каталоге системы[1].

-e <expire_date>
задает дату окончания срока действия учетной записи в формате YYYY-MM-DD.

-f <inactive_days>
задает число дней после завершения срока действия пароля, по истечении которых учетная запись удаляется из системы. Значение 0 блокирует учетную запись сразу после завершения срока действия пароля, -1 отключает функция удаления записей. По умолчанию используется значение -1.

-g <initial_group>
задает имя или номер группы, к которой пользователь присоединяется в момент регистрации в системе (эта группа должна уже существовать, а номер должен указывать на существующую группу). По умолчанию используется группа 1 или иная группа, указанная в файле /etc/default/useradd.

-G <group,[...]>
список дополнительных групп, в которые входит пользователь. Идентификаторы групп перечисляются через запятую (без пробелов). Все указанные в строке идентификаторы должны относиться к существующим группам. По умолчанию пользователь не включается ни в одну дополнительную группу.

-m
задает создание домашнего каталога для пользователя, если этот каталог отсутствует. Если задана опция **-k**, в домашний каталог копируются файлы из каталога skeleton_dir, в противном случае - файлы, содержащиеся в /etc/skel. При копировании в домашнем каталоге создаются все подкаталоги skeleton_dir или /etc/skel. Опцию **-k** можно использовать лишь вместе с опцией **-m**. По умолчанию домашний каталог для пользователя не создается.

-o
позволяет добавить пользователя с уже имеющимся в системе (неуникальным) идентификатором UID.

-p <passwd>
задает шифрованный пароль (как возвращает crypt).

-s <shell>
задает имя командного процессора (shell) для пользователя. Если эта опция не задана, пользователь получает принятый по умолчанию командный процессор.

-u <uid>
задает числовой идентификатор пользователя. Идентификатор должен быть уникальным, если в строке не задана опция **-o**. Идентификаторы должны быть целыми неотрицательными числами. По умолчанию используется наименьшее значение, которое превышает 99 и максимальное значение имеющегося в системе пользовательского идентификатора. Значения от 0 до 99 обычно резервируются для системных имен.

2.2.2.3 Изменение принятых по умолчанию значений

При использовании с опцией **-D** команда **useradd** будет выводить на экран принятые по умолчанию значения или заменять их в соответствии с описанными ниже опциями командной строки.

-b <default_home>
задает префикс имен домашних каталогов пользователей. При создании пользовательского каталога имя пользователя добавляется к заданному параметром префиксу, если в командной строке не задана опция **-d**.

-e <default_expire_date>
указывает дату завершения срока действия учетной записи пользователя.

-f <default_inactive>
задает число дней, по истечении которого учетная запись удаляется в случае завершения срока действия пароля.

-g <default_group>
задает идентификатор (имя или GID) группы, в которую пользователи включаются по умолчанию. Этот параметр должен содержать имя или номер существующей в системе группы.

-s default_shell
задает имя выбираемого по умолчанию для новых пользователей командного процессора.

При запуске **useradd** без параметров программа выводит на экран список принятых по умолчанию настроек для создания новых пользователей.

2.2.2.4 Известные проблемы

Программа **useradd** не позволяет добавлять пользователей в группу NIS - эта операция должна выполняться сервером NIS.

1 *Обычно это каталог /home.*

2.2.2.5 Файлы

/etc/passwd - информация об учетных записях пользователей (см. параграф 2.2.1 на стр. 34)

/etc/shadow - теневые записи для пользователей

/etc/group - информация о группах

/etc/gshadow - теневые записи для групп

/etc/default/useradd - используемые по умолчанию параметры для добавления пользователей

/etc/login.defs - системные установки для регистрации пользователей

/etc/skel - каталог, содержащий используемые по умолчанию файлы[1].

2.2.2.6 Удаление и редактирование учетных записей пользователей

Для удаления учетной записи пользователя служит команда

```
userdel [-r] <имя пользователя>
```

которая удаляет информацию о пользователей из конфигурационных файлов системы и может также удалять домашний каталог пользователя (опция **-r**). С помощью команды

```
usermod <имя пользователя>
```

администратор может изменить параметры учетной записи пользователя. Информацию о параметрах usermod вы можете получить по команде

```
man usermod
```

2.2.3 Группы

Для создания в системе новой группы используется команда **groupadd.**

2.2.3.1 Синтаксис и параметры

```
groupadd [-g gid [-o]] [-r] [-f] group
```

Команда **groupadd** создает в системе новую учетную запись для группы с использованием заданных в командной строке параметров и принятых по умолчанию значений. Новая группа включается в соответствующие системные файлы. Опции командной строки могут включать перечисленные ниже параметры.

```
-g <gid>
```

задает числовой идентификатор группы (GID). Этот идентификатор должен быть уникальным, если в командной строке не указана опция **-o**. Значения идентификаторов должны быть неотрицательными целыми числами. По умолчанию для новой группы используется минимальное значение, которое больше 500 и идентификаторов всех существующих в системе групп. Значения от 0 до 499 обычно резервируют для системных групп.

```
-r
```

этот флаг используется для добавления системной группы. Для новой группы будет автоматически выбран идентификатор менее 499, если с помощью параметра **-g** не задано иное значение. Данная опция поддерживается не всеми дистрибутивами Linux.

```
-f
```

При использовании этого флага **groupadd** будет завершать работу с выдачей на экран сообщения об ошибке, если группа с заданным именем уже существует в системе[2].

Данный флаг также изменяет способ работы параметра **-g**. При запросе неуникального идентификатора в отсутствии опции **-o** команда будет работать так, как будто опция **-g** также не задана. Этот флаг поддерживается не всеми вариантами Linux.

Файлы

/etc/group - информация об учетных записях для групп;

/etc/gshadow - теневые записи для групп.

1 *Ответственность за создание и поддержку используемых по умолчанию файлов лежит на администраторе системы.*
2 *Существующая группа при этом не изменяется.*

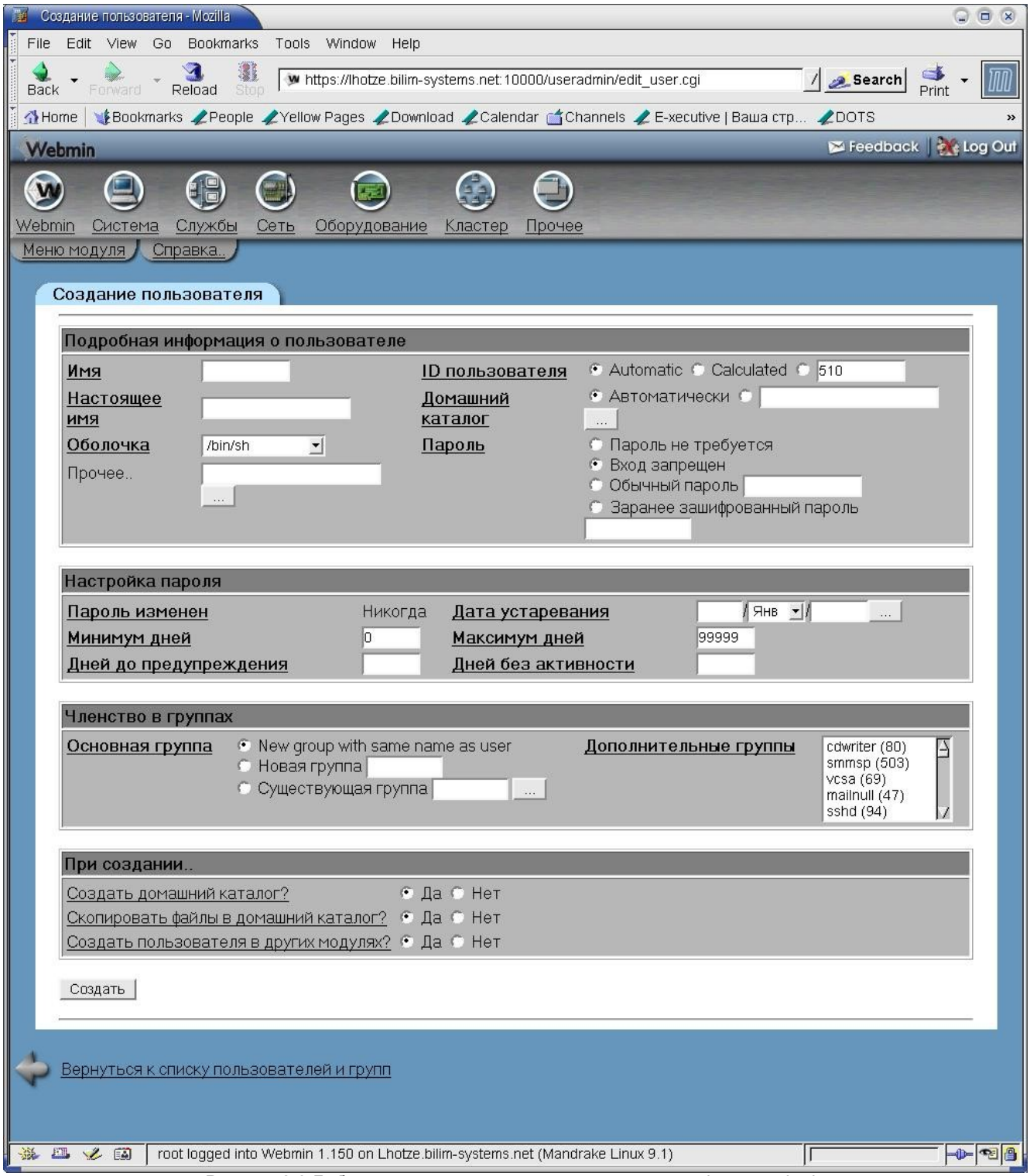


Рисунок 2.6 Добавление пользователя с помощью интерфейса webmin

2.2.4 Теневые учетные записи

Как было отмечено выше сохранение зашифрованных паролей в файле /etc/passwd не обеспечивает требуемого уровня безопасности, поскольку этот файл открыт для чтения все пользователям и любой злоумышленник может спокойно подбирать пароли, скопировав файл на свой компьютер. Для решения этой проблемы в современных дистрибутивах Linux применяется теневое хранение паролей.

Файл /etc/passwd остается доступным для чтения, но уже не содержит зашифрованных паролей пользователей системы. Взамен этого пользовательские учетные записи содержат ссылку на пароль (см. параграф 2.2.1 на стр. 34), а зашифрованные пароли хранятся теперь в файле /etc/shadow, доступ к которому имеет только пользователь root.

В файле /etc/shadow хранятся зашифрованные пользовательские пароли и метки специальных правил, которые рассматриваются ниже. Пример файла /etc/shadow показан на рисунке 2.7.

Каждая запись файла состоит из 9 полей, разделенных двоеточием (:). Поля записей файла **/etc/shadow** перечислены в таблице 3.

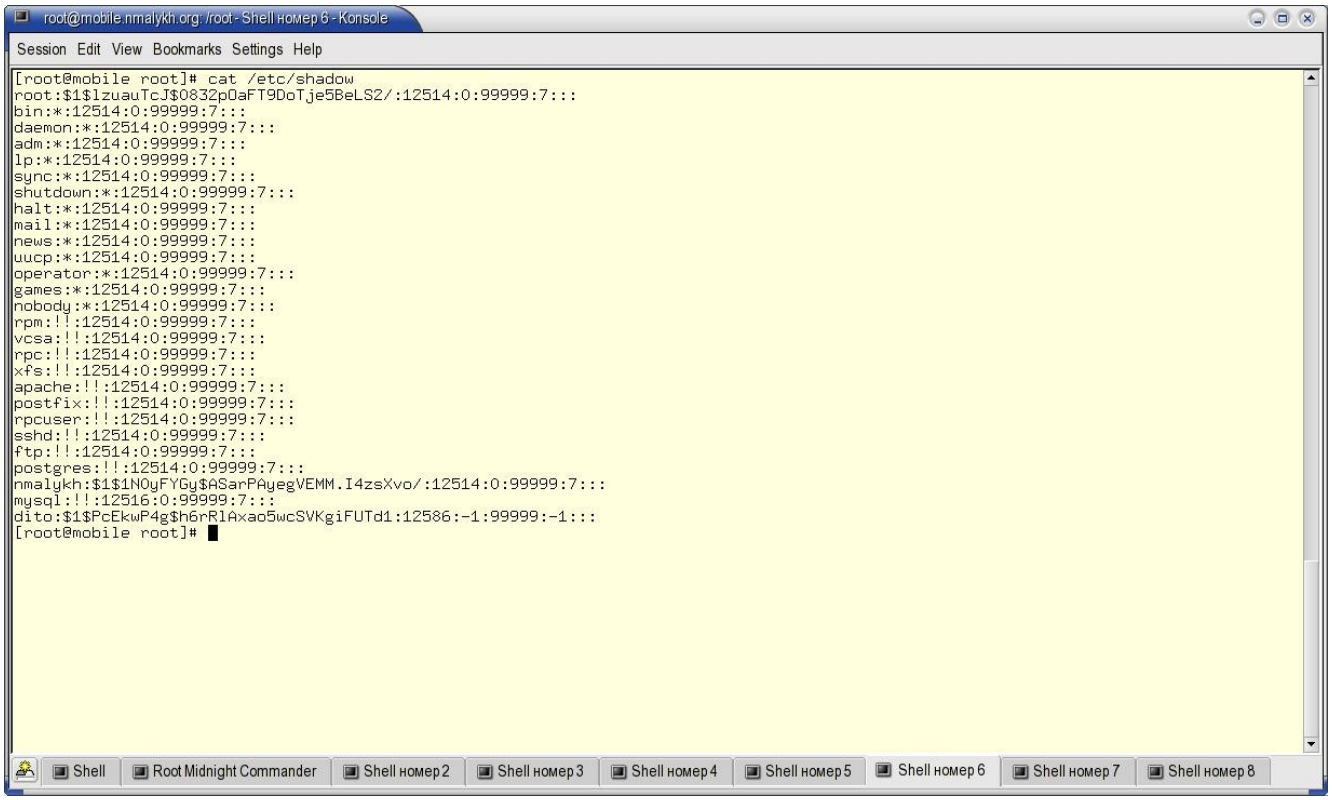


Рисунок 2.7. Файл /etc/shadow.

Таблица 3 Поля записей файла /etc/shadow

Поле	Назначение
Имя пользователя	Регистрационное имя пользователя в системе (такое же, как в файле **/etc/passwd**).
Зашифрованный пароль	Обязательное поле, которое содержит зашифрованный пароль. Размер шифрованного пароля составляет от 13 до 24 символов, включая буквы латиницы (строчные и прописные), цифры и некоторые специальные символы (. и /). Более подробные сведения о шифрованных паролях можно получить с помощью команды **man 3 crypt**. В поле пароля могут использоваться специальные значения ***** (пользователю запрещен вход в систему) и **!!**.
Дата изменения пароля	Число дней после 1 января 1970 года до момента последней смены пароля.
Срок возможной замены пароля	Количество дней, по истечении которого пользователь **может** изменить пароль. Если значение этого поля превышает значение следующего поля, пользователь не может самостоятельно изменить пароль.
Срок обязательной замены пароля	Количество дней, по истечении которого пользователь **обязан** сменить пароль.
Срок предупреждения	Количество дней до обязательной замены пароля, когда пользователю выдается предупреждение о необходимости изменить пароль.
Срок изменения	Количество дней, по истечении которого пользовательская учетная запись будет заблокирована, если пароль не будет заменен.
Дата блокировки	Число дней после 1 января 1970 года до момента блокирования учетной записи пользователя.
Резервное поле	Не используется.

2.3 Управление доступом к файлам

В системах UNIX для каждого файла[1] устанавливаются права доступа, определяющие возможности различных пользователей при работе с этим файлом. Для управления доступом подходит любой метод, позволяющий избирательно разрешать или запрещать пользователям доступ к файлу.

2.3.1 Права доступа и принадлежность файлов

Доступ пользователей UNIX к файлам и каталогам определяются специальными параметрами файлов, называемыми **правами доступа** (permission). Поддерживается три основных типа доступа к файлам:

- **чтение** (**r**) - пользователи могут читать файл при наличии данного атрибута;

1 *К файлам в контексте управления доступом будем относить собственно файлы, каталоги и некоторые устройства, доступные через специальные файлы.*

- **запись** (**w**) - этот атрибут разрешает пользователям запись в файл, однако он не разрешает чтения
- **выполнение** (**x**) - атрибут позволяет пользователям выполнить программу или сценарий, содержащийся в файле.

Каждый файл в системе имеет владельца (пользователь, идентификатор которого указан в соответствующем атрибуте файла и связан с определенной группой пользователей (идентификатор группы является одним из атрибутов файла). Доступ к файлу определяется тремя группами параметров, определяющими возможности владельца файла, членов группы, к которой приписан файл и всех остальных пользователей.

Если вы посмотрите содержимое какого-либо из каталогов Linux с помощью команды **ls -l**, то увидите примерно следующее (рисунок 2.8):

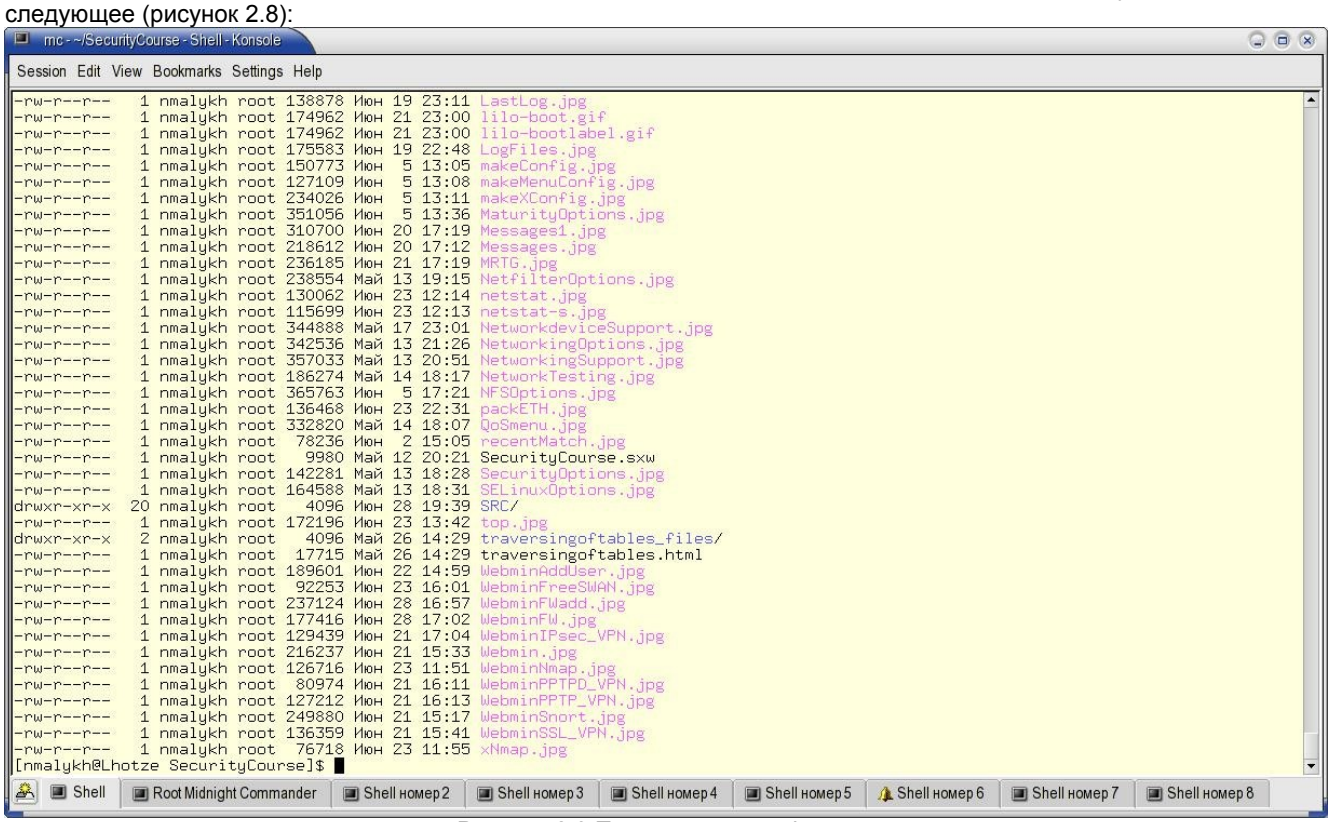


Рисунок 2.8 Права доступа к файлам

Для каждого файла строка списка содержит строку прав доступа, а также информацию о владельце файла и группе (3 и 4 колонка, соответственно). Поле прав доступа выводится в виде строки, состоящей из 10 символов, каждый из которых указывает наличие или отсутствие соответствующих прав доступа для владельца файла, членов группы, к которой приписан файл, и всех прочих пользователей. Формат поля прав доступа описан в таблице 4.

Таблица 4 Формат представления прав доступа

1	2	3	4	5	6	7	8	9	10
d	**r**	**w**	**x**	**r**	**w**	**x**	**r**	**w**	**x**
Тип	Права доступа для владельца			Права доступа для группы			Права доступа для прочих		

Первый символ указывает тип ресурса и может принимать следующие значения:

- **-** - файл;
- **b -** блочное устройство;
- **c -** символьное устройство;
- **d -** каталог;
- **l -** символьная связь.

Оставшиеся 9 символов разбиты на три группы, определяющие права доступа к ресурсу для его владельца, членов группы и прочих пользователей. Первым указывается атрибут возможности чтения, затем записи и последним - выполнения. Наличие соответствующего символа в строке прав говорит о возможности выполнения действия, а символ - указывает на запрет такой возможности.

Используется также числовое представление прав доступа в форме 4-значного восьмеричного числа. Первая (старшая) цифра определяет тип, вторая - права прочих пользователей, третья - права группы и четвертая - права владельца. Таблица содержит восьмеричные маски представления прав доступа.

Таблица 5 Восьмеричные маски прав доступа

Значение	Описание прав
0000	Файл, к которому никто не имеет доступа (**---**)
0001	Владелец может выполнять программу или сценарий (**--x**)
0002	Владелец имеет право записи в файл (**-w-**)
0004	Владелец может читать файл (**r--**)
0010	Члены группы могут выполнять программу или сценарий (**--x**)
0020	Члены группы имеют право записи в файл (**-w-**)
0040	Члены группы могут читать файл (**r--**)
0100	Все прочие пользователи имеют право выполнять программу или сценарий (**--x**)
0200	Все прочие пользователи имеют право записи в файл (**-w-**)
0400	Все прочие пользователи имеют читать файл (**r--**)
1000	Sticky-бит, используемый для важных каталогов типа **/tmp**. Наличие этого флага позволяет удалять из каталога файлы только владельцу данного каталога или самих файлов. Такая возможность позволяет создавать каталоги, запись в которые разрешается всем пользователям, но удалять каждый может только свои файлы. В символьном представлении каталоги с данным атрибутом обозначаются буквой **t** взамен атрибута **x** для владельца.
2000	Файл со специальными правами доступа SGID (см. параграф 2.3.3 на стр. 42).
4000	Файл со специальными правами доступа SUID (см. параграф 2.3.3 на стр. 42).

При описании комбинации прав доступа указанные в таблице маски складываются. Такое представление прав может показаться слишком сложным, однако это не так и вы легко согласитесь, посмотрев таблицу 6. Комбинации прав доступа для владельца, группы и прочих пользователей представляются восьмеричными цифрами (от 0 до 7), объединенными в триаду. Например число 755 разрешает владельцу все операции, а членам группы и прочим пользователям - только чтение и выполнение.

Таблица 6. Восьмеричное представление возможных комбинаций прав доступа.

Значение	Комбинация прав
0	Отсутствие прав доступа
1	Возможно выполнение программы или сценария
2	Разрешена запись
3	Разрешено выполнение и запись в файл
4	Разрешается читать файл
5	Можно читать файл и выполнять программу или сценарий
6	Разрешается чтение и запись в файл
7	Разрешены все операции

2.3.2 Права доступа, устанавливаемые по умолчанию

При создании в системе новых файлов актуален вопрос изначальной установки прав доступа для нового файла и его принадлежности. Если с владельцем файла все понятно (кто создал, тот и владеет), то выбор группы в разных вариантах UNIX может осуществляться по разному. В Linux при создании файла в качестве группы обычно используется первичная группа создавшего файл пользователя. Вы можете поэкспериментировать со своей системой, используя для смены первичной группы команду

```
newgrp <имя группы>
```

Для определения текущего идентификатора и имена первичной группы можно воспользоваться командой **id**.

Права доступа для нового файла определяются заданной администратором маской. Значение маски представляет собой трехзначное восьмеричное число, определяющие биты прав доступа, которые при создании файла следует маскировать (снять). Таким образом, сумма значения маски и прав доступа для создаваемых файлов равняется 777. Если вы хотите устанавливать для новых файлов права 751 (все права для владельца, чтение и выполнение для группы, только выполнение для прочих пользователей), маска должна иметь значение 026. В современных дистрибутивах Linux обычно используется маска 022 (права доступа 755). Для установки маски служит команда

```
umask <маска>
```

которая обычно выполняется в каком-либо из сценариев инициализации системы. Отметим, что пользователь может выбрать для создания файлов значение маски, отличное от принятого в системе по умолчанию. Для этого достаточно включить команду **umask** в свой сценарий входа в систему[1]. Очевидно, что и администратор может задать различные маски для файлов, создаваемых процессами, работающими от имени системных пользователей (mail, named, daemon и т. п.)

Для задания принадлежности файла и установки прав доступа к нему служат команды **chown** (стр. 189), **chgrp** (стр.

1 *Конфигурационный файл используемого командного интерпретатора.*

187) и **chmod** (стр. 188) из пакета **coreutils**, входящего в состав большинства дистрибутивов Linux. Программы этого пакета рассматриваются в параграфе 11.1.1 (стр. 187).

2.3.3 Специальные права доступа к файлам

В этом разделе мы неоднократно упоминали о битах SUID и SGID в поле прав доступа к файлу. Эти биты задают специфические условия для работы программ, хранящихся в соответствующих файлах:

- **SGID** предоставляет процессу права группы, которой принадлежит данный файл, независимо от того, какой пользователь запустил программу.
- **SUID** предоставляет процессу права владельца данного файла, независимо от того, какой пользователь запустил программу.

Очевидно, что программы с такими атрибутами расширяют возможности обычных пользователей предоставляя им возможности привилегированных пользователей[1], но эти же биты существенно повышают уровень риска для вашей системы. Если проникший в вашу систему злоумышленник воспользуется SUID-программой, исполняемый файл которой принадлежит пользователю root, этот злоумышленник сможет много добиться в реализации своих черных замыслов.

В некоторых дистрибутивах Linux общего назначения бит SUID установлен для большого числа файлов и некоторые из таких файлов могут иметь существенные прорехи в безопасности. Для определения имеющихся в вашей системе SUID-программ воспользуйтесь командой

```
find / -perm +4000
```

которая выведет на экран список, подобный приведенному ниже[2]. Вот число программ в этом списка для многих стандартных инсталляций Linux будет существенно больше.

```
/usr/bin/chage
/usr/bin/gpasswd
/usr/bin/at
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/newgrp
/usr/bin/crontab
/usr/bin/lppasswd
/usr/bin/ssh
/usr/bin/suidperl
/usr/bin/sperl5.6.1
/usr/bin/rcp
/usr/bin/rlogin
/usr/bin/rsh
/usr/bin/sudo
/usr/lib/mc/bin/cons.saver
/usr/sbin/ping6
/usr/sbin/traceroute6
/usr/sbin/sendmail.sendmail
/usr/sbin/userhelper
/usr/sbin/usernetctl
/usr/sbin/traceroute
/usr/sbin/suexec
/usr/X11R6/bin/XFree86
/bin/ping
/bin/mount
/bin/umount
/bin/su
/sbin/pwdb_chkpwd
/sbin/unix_chkpwd
```

Для того чтобы снизить риск, связанный с наличием SUID-программ я рекомендую воспользоваться приведенными ниже советами.

1) Удалите с диска все SUID-программы, которые не являются абсолютно необходимыми.
2) Удалите бит SUID для тех программ, которые могут обойтись без этого бита.
3) Убедитесь, что файлы сценариев, имеющие бит SUID защищены от записи.
4) Для защиты необходимых SUID-программ воспользуйтесь разработанными для таких случаев средствами (например, программой SUID/SGID Generic Wrapper[3]).
5) Периодически проверять не появилось ли в системе новых SUID-программ (см параграф 11.5 на стр. 238).

1 *Обычно пользователя root.*
2 *Этот список содержит SUID-программы на одном из граничных шлюзов, поэтому число включенных в него программ весьма невелико.*
3 *Исходный текст этой программы вы сможете найти в каталоге SRC/ (файл wrapper.c) приложенного к книге компакт-диска или загрузить с сайта http://www.scn.rain.com/pub/security/wrapper.c.*

2.4 Поддержка целостности системы

2.4.1 Враждебный код

Одной из важнейших угроз для любой компьютерной системы является проникновение враждебного кода, который может использоваться для нанесения непосредственного ущерба данной системе, сбора конфиденциальной информации или выполнения несанкционированных действий по отношению к другим системам. Что же такое враждебный код:

- спрятанный в обычные программы код, выполняющий неизвестные (потенциально опасные) пользователю действия; такой код может появиться при получении бинарных программ (а иногда и исходных текстов) из непроверенных источников;
- обычная программа, измененная тем или иным способом и содержащая в себе код, выполняющий неизвестные (потенциально опасные) функции; появление такого кода может быть следствием работы вирусов или троянских программ;
- любая программа, маскирующая свою несанкционированную деятельность (потенциально опасную) выполнением нужных для пользователя операций; такие программы обычно называют троянскими;
- любая программа, повреждающая или уничтожающая информацию в системе;
- любая программа, занимающимся несанкционированным сбором информации в системе;
- любой несанкционированный код, скрывающий свое присутствие в системе.

Существует достаточно много разновидностей враждебного кода, но основную роль играют вирусы и троянские программы.

2.4.1.1 Троянские программы

Троянскими (по аналогии с печально известным троянским конем) называют программы, которые выполняют несанкционированные действия по сбору конфиденциальной информации, повреждению данных и т. п., маскируясь под обычные программы (чаще всего повседневно используемые и не вызывающие тревоги). Примером такого враждебного кода этого типа может служить достаточно распространенный вариант подмены программы **/boot/login** измененной программой, которая не только проверяет пароль при регистрации пользователя в системе, но и сохраняет пароли в скрытом от глаз пользователей файле, который потом может быть переправлен по электронной почте злоумышленнику или получен им иным способом.

Троянские программы могут появляться под видом любых пользовательских приложений или утилит, но они не могут распространяться без прямого участия пользователей[1]. Так или иначе такие программы кто-то должен доставить в вашу систему на съемных носителях или через сеть. Поэтому одной из задач администратора является контроль за установкой программ на рабочих станциях пользователей и тем более на серверах или сетевых шлюзах. Задача осложняется еще и тем, что зачастую нет возможности проверить легитимность программы до ее установки. Достаточно часто встречаются ситуации, когда троянские программы распространяются с официальными дистрибутивами, а иногда возникают и варианты подмены исходного кода свободно распространяемых программ[2].

Обнаружение троянских программ может оказаться достаточно сложной задачей, требующей контроля за изменениями файлов вашей системы и поиска подозрительных событий. Существует целый ряд программ, способных облегчить эту задачу. Некоторые программы этого типа рассматриваются в параграфе 11.5 (стр. 238). Следы наличия в системе троянских программ можно обнаружить при внимательном просмотре системных журналов (параграф 2.8 на стр. 45). Программы мониторинга системных журналов рассмотрены в параграфе 11.8 (стр. 257).

2.4.1.2 Вирусы

В отличие от троянских программ вирусы способны распространяться без явного участия пользователей, поскольку содержат в своем коде те или иные механизмы размножения. В общем случае к вирусам относятся программы (зачастую написанные на макроязыках популярных пользовательских приложений), содержащие в своем коде механизм распространения и те или иные несанкционированные действия. Вирусы можно поделить на две категории:

- загрузочные вирусы, изменяющие код системного загрузчика или главную загрузочную запись (MBR);
- файловые вирусы, распространяющиеся в виде исполняемых файлов (программ), библиотек, а также в форме документов, содержащих макрокод и сообщений электронной почты с вложенными файлами.

Загрузочные вирусы были широко распространены в 80-х годах прошлого века, а сейчас они стали экзотикой, поскольку такие вирусы очень просто обнаружить и совсем непросто написать достаточно эффективный вирус с исполняемым кодом небольшого размера. Сегодня в основном распространяются файловые вирусы и в связи с повсеместным использованием Internet и электронной почты все большее распространение получают макровирусы, которые можно встраивать в обычные документы и распространять различными способами, включая рассылку по электронной почте.

Каждый вирус содержит в себе код распространения, обеспечивающий присоединение враждебного кода к

1 *В этом и состоит коренное отличие между троянскими программами и вирусами.*
2 *Достаточно вспомнить историю с подменой исходных текстов программы tcpdump, используемой практически на каждой UNIX-системе.*

нормальным исполняемым файлам, библиотекам или документам, поддерживающим макрокоманды. В результате заражения файла он сам уже становится вирусоносителем, способным заражать другие файлы, обеспечивая распространение вируса. В зависимости от эффективности кода размножения в том или ином конкретном вирусе этот вирус распространяется по файлам зараженной системы и передается на другие компьютеры по каналам обмена информацией (сеть или сменные носители). Поскольку большая часть современных компьютеров имеет доступ в Internet и пользователи работают с электронной почтой, почтовые вирусы с эффективной системой размножения могут порождать в мире настоящие эпидемии, когда число зараженных компьютеров достигает миллионов[1].

Если в прошлом вирусы часто выполняли те или иные разрушительные функции (уничтожение файлов, искажение информации и т. п.), то в последние годы вирусы стали выполнять более “созидательные” функции. Многие из сегодняшних почтовых вирусов, например, при заражении делают пользовательский компьютер открытым транслятором, который может служить для распространения спама и вирусов без ведома пользователя. Встречаются вирусы, собирающие конфиденциальную информацию, а также обеспечивающие возможность использования зараженного компьютера для организации атак через сеть на хосты Internet.

Основной средой обитания вирусов являются операционные системы с бесконтрольным доступом к файлам и системным функциям, которые по сути являются питательной средой для вирусов. К таким ОС относятся прежде всего операционные системы компании Microsoft, в которых файлы не имеют хозяина и, следовательно, в большинстве случаев доступны любому пользователю[2]. Системы UNIX с четкой системой контроля и разграничения прав доступа к файлам и каталогам, не обеспечивают условий для существования (тем более, размножения) вирусов, поэтому вирусов для ОС UNIX (включая Linux) практически не существует.

Однако, несмотря на практически полное отсутствие UNIX-вирусов, существует достаточно много средств обнаружения вирусов, разработанных для платформ UNIX. Здесь нет противоречия. Дело в том, что большинство почтовых трансляторов и граничных шлюзов работают на UNIX-платформах и фильтрация проходящего трафика (включая проверку электронной почты) может послужить надежным заслоном от проникновения вирусов через сеть. В параграфе 10.1 (стр. 179) рассматривается несколько систем обнаружения и фильтрации вирусов для INUX-платформ.

2.4.1.3 Сканеры

Сканер представляет собой программу, способную просматривать открытые в системе порты. Как и многие другие приложения, сканеры портов могут использоваться с разными намерениями - как добрыми, так и враждебными. Один человек может написать мощную систему сканирования для обнаружения уязвимостей в своей системе с целью повышения уровня безопасности, но никто не помешает другому использовать этот сканер для поиска лазеек в чужие системы. Поэтому сканеры следует рассматривать как враждебный код, поскольку любое несанкционированное сканирование портов вашей системы или сети говорит о недобрых намерениях или чьей-то глупости. Оба случая требуют пристального внимания администратора.

Практически все сканеры в той или иной форме реализуют следующий алгоритм проверки систем:

1) загрузка определенного набора тестов или атак;

2) выполнение набора тестов/атак;

3) генерация отчета о результатах проверки.

Если сканер работает в интерактивном режиме или реализует достаточно интеллектуальные алгоритмы, то созданный на этапе 3 отчет может служить основой для повторения процедуры с иным набором проверок или атак.

Будем разделять сканеры на две категории - локальные и сетевые сканеры. Локальные сканеры требуют наличия терминального доступа в систему и связаны в основном с проверкой уязвимостей, позволяющих зарегистрированному в системе пользователю получить несанкционированный доступ к тем или иным ресурсам. Сетевые сканеры проверяют уязвимости системы, которыми можно воспользоваться через сеть (например, несанкционированный доступ в систему или использование хоста для рассылки спама или организации DDoS-атаки)

Локальный сканер проверяет локальный хост, пытаясь найти на нем уязвимые точки, которые позволят получить доступ к тем или иным ресурсам данного хоста или других систем, недостаточно хорошо закрытым администратором. Локальные уязвимости не обязательно связаны с сетевыми службами - это может быть слабая проверка при регистрации в системе, наличие файлов со слишком широкими возможностями доступа, установка битов SUID, SGID (см. параграф 2.3.3) для пользовательских программ и т. п. Работу сканера на локальной машине обычно достаточно трудно скрыть и администратор может быстро принять адекватные меры при обнаружении такого сканирования. Мы не будем останавливаться подробней на работе локальных сканеров, лишь рассмотрим ниже работу некоторых программ такого типа с точки зрения проверки уровня безопасности хостов (см. параграф 11.4).

Сетевые сканеры, которые часто называют сканерами портов, проверяют удаленные хосты через сеть на предмет обнаружения открытых портов, определения ОС и т. п. По результатам сканирования можно определить слабые места системы и попытаться проникнуть в нее или нанести иной вред. Сканеры портов (см. параграф 11.12) позволяют получить достаточно много информации о хосте, которая может быть использована злоумышленником для несанкционированного доступа в сеть или использования хоста для выполнения враждебных действий по отношению к другим системам. Для обнаружения сетевых сканеров могут использоваться средства мониторинга соединений (параграф 11.10) и IDS (параграф 11.6). Немаловажную роль может сыграть при обнаружении фактов сканирования и анализ журнальных файлов (параграф 11.8).

1 *Достаточно вспомнить вирус Melissa, заразивший компьютеры всей планеты за считанные часы.*
2 *Некоторым исключениям является файловая система NTFS, где присутствуют средства контроля доступа на уровне файлов и каталогов.*

2.4.1.4 Анализаторы протоколов

Программы сбора пакетов (анализаторы протоколов[1]), строго говоря, не относятся к враждебному коду, однако несанкционированный сбор пакетов в системе может свидетельствовать о проникновении в сеть внешнего злоумышленника или наличии внутри сети “засланного казачка” или просто человека, не ведающего, что он творит. В любом случае администратор безопасности должен обращать пристальное внимание на факты присутствия в системе анализаторов протоколов и создаваемых ими файлов. Отметим также, что сбор пакетов может осуществляться и с других хостов локальной сети - на это тоже нужно обращать внимание.

Анализатор протоколов представляет собой программу сбора и анализа передаваемых по сети пакетов, работающую на специальной платформе или компьютере общего назначения. Обычно сетевые интерфейсы принимают из среды и обрабатывают лишь пакеты, в которых адрес канального уровня[2] соответствует адресу данного интерфейса. Однако интерфейс можно программным путем перевести в режим захвата (promiscuous), в котором интерфейс будет прослушивать все передаваемые в сетевой среде пакеты. Используя программы, способные анализировать (фильтровать) собранные интерфейсом пакеты, злоумышленник может получить много важной и интересной информации, передаваемой по сети:

- регистрационные имена и пароли, которые зачастую передаются в незашифрованном виде;
- конфиденциальную информацию, предназначенную для узкого круга пользователей;
- сведения личного характера, которые не подлежат разглашению.

Поскольку анализаторы протоколов являются пассивными программами, их обнаружение достаточно непросто. Во время сбора пакетов можно обнаруживать некоторые анализаторы по создаваемым ими файлам, однако во многих случаях злоумышленнику не требуется сохранять собранные данные и он может удовлетвориться их выводом на экран. После завершения процедуры захвата пакетов файлы с данными скорей всего будут уничтожены или перенесены в другое место, поэтому поиск таких файлов не является эффективным способом обнаружения анализаторов. Однако и отказываться от него не стоит. Для поиска таких файлов можно использовать программу **chkrootkit** (параграф 11.5.1).

Другим способом поиска анализаторов является выявление интерфейсов, работающих в режиме захвата пакетов. В этом вам могут помочь системные утилиты **ifconfig** (параграф 11.1.2.3), **ifstatus** (параграф 11.2.4) и некоторые другие программы, описанные в параграфе 11.7.

2.5 Способы обеспечения безопасности при удаленном доступе к хостам Linux

Средства и способы удаленного доступа к хостам Linux через корпоративные и публичные сети. Важность защиты управляющего трафика при удаленном администрировании хостов.

Рассматриваются способы и средства обеспечения безопасного доступа администратора к хостам Linux для удаленного управления через сеть.

2.6 Обеспечение безопасности служб Internet/intranet

2.6.1 Безопасность Web-серверов

Рассматриваются способы обеспечения безопасной работы Web-серверов на базе Apache. Обсуждаются вопросы защиты передаваемой через публичные сети информации (шифрование, сертификаты и т. д.).

2.6.2 Безопасность электронной почты

Требования к почтовым трансляторам с точки зрения безопасности. Предотвращение несанкционированного использования почтовых трансляторов (рассылка спама). Безопасность систем почтовой рассылки.

Пользовательские службы электронной почты, сравнение серверов POP3 и IMAP, управление доступом пользователей к почтовым ящикам.

2.6.3 Безопасность служб FTP, удаленного доступа, Web-приложений

Организация безопасной работы серверов FTP и удаленного доступа к хостам UNIX. Разработка безопасных Web-приложений.

2.7 Средства проверки полномочий пользователей и управления доступом к сетевым ресурсам

Средства централизованного управления пользовательскими учетными записями и проверки полномочий пользователей при доступе в сеть (LDAP, RADIUS и т. п.).

2.8 Системные журналы Linux

Ведение системных журналов является одним из важнейших преимуществ UNIX-подобных систем и, в частности, Linux. Системные журналы Linux содержат сведения о всех важных событиях, происходящих в системе и позволяют

1 *Некоторые анализаторы протоколов подробно рассматриваются в параграфе 11.9.*
2 *Для сетей Ethernet это MAC-адрес.*

администратору найти причины неэффективной работы той или иной компоненты, обнаружить попытки несанкционированного доступа или взлома.

Журнализация (logging) представляет собой процессы записи операционной системой или прикладными программами сведений о происходящих событиях в специальные файлы. Записи о событиях делаются по мере возникновения событий и журнальные файлы сохраняются для их последующего анализа.

В Linux журнализация поддерживается операционной системой, большинством прикладных программ и даже отдельными протоколами. Большинство служб Linux сохраняют сведения в своих собственных журнальных файлах и системных журналах общего пользования. Например, на моей рабочей станции Mandrake в каталоге /var/log содержится множество журнальных файлов общего назначения и отдельных программ и служб (см. параграф 2.9).

Рассмотрим некоторые из журнальный файлов общего назначения и утилиты для работы с ними. Дополнительные программы для анализа системных журналов рассматриваются в параграфе 11.8.

2.8.1 Журналы регистрации пользователей

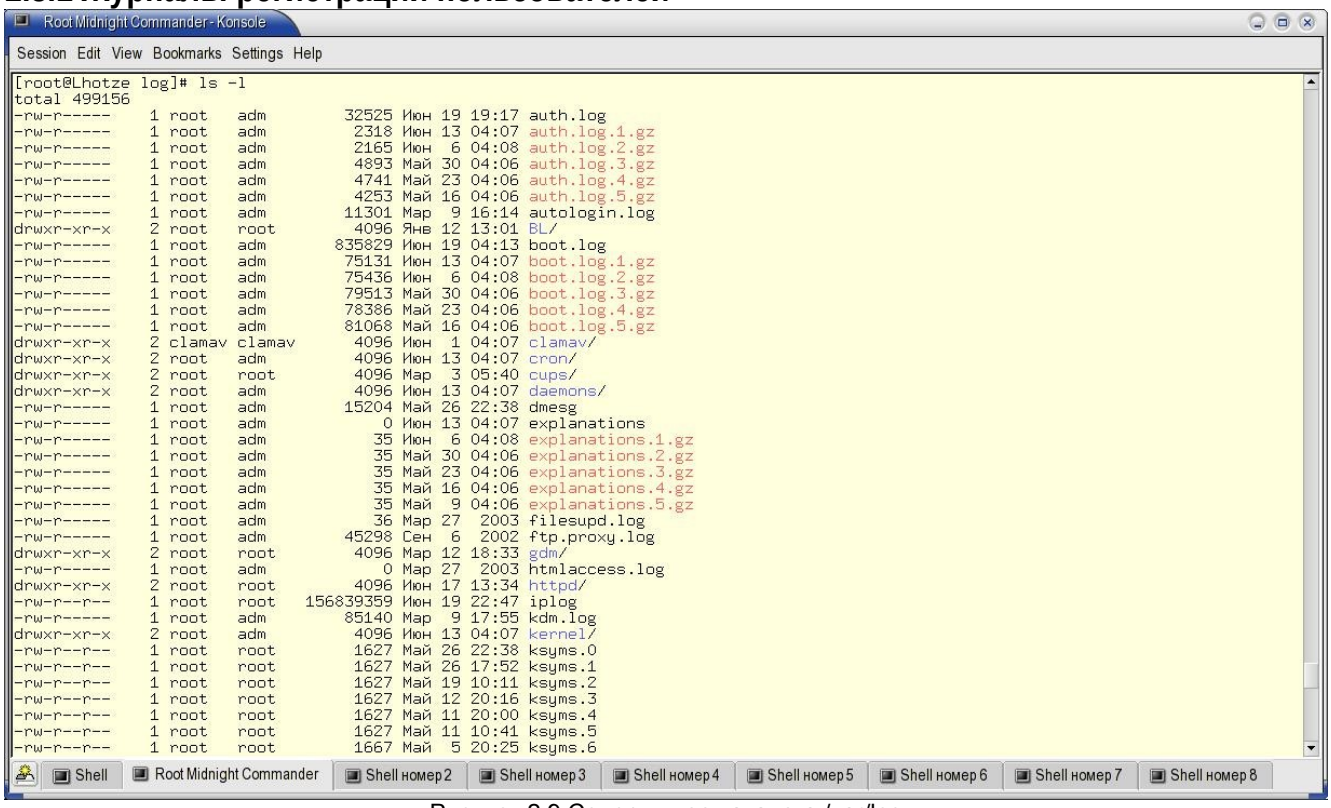


Рисунок 2.9 Содержимое каталога /var/log

В системах Linux фиксируется информация о каждом входе пользователя в систему с локальной консоли, удаленного терминала или через ту или иную сетевую службу. Журналы регистрации могут оказать неоценимую услугу при обнаружении попыток взлома системы чужими пользователями или недопустимого использования предоставленных прав легитимными пользователями системы.

2.8.1.1 lastlog

В системном журнале **lastlog** записываются сведения о регистрации пользователей в системе. При входе пользователя в файл **/var/log/lastlog** записывается регистрационное имя пользователя, время последней регистрации в системе и порт (терминал), использованный для подключения. Файл **lastlog** использует бинарный формат и для работы с этим файлом используется утилита **lastlog**. По умолчанию команда **lastlog** выводит записи о регистрации в системе всех пользователей, указанных в файле **/etc/passwd**, как показано на рисунке 2.10. Можно посмотреть сведения о последней регистрации в системе отдельного пользователя с помощью команды

```
lastlog -u <имя пользователя>
```

В системах с большим числом пользователей может оказаться удобной опция -t, позволяющая задать временной интервал, для которого выводятся сведения.

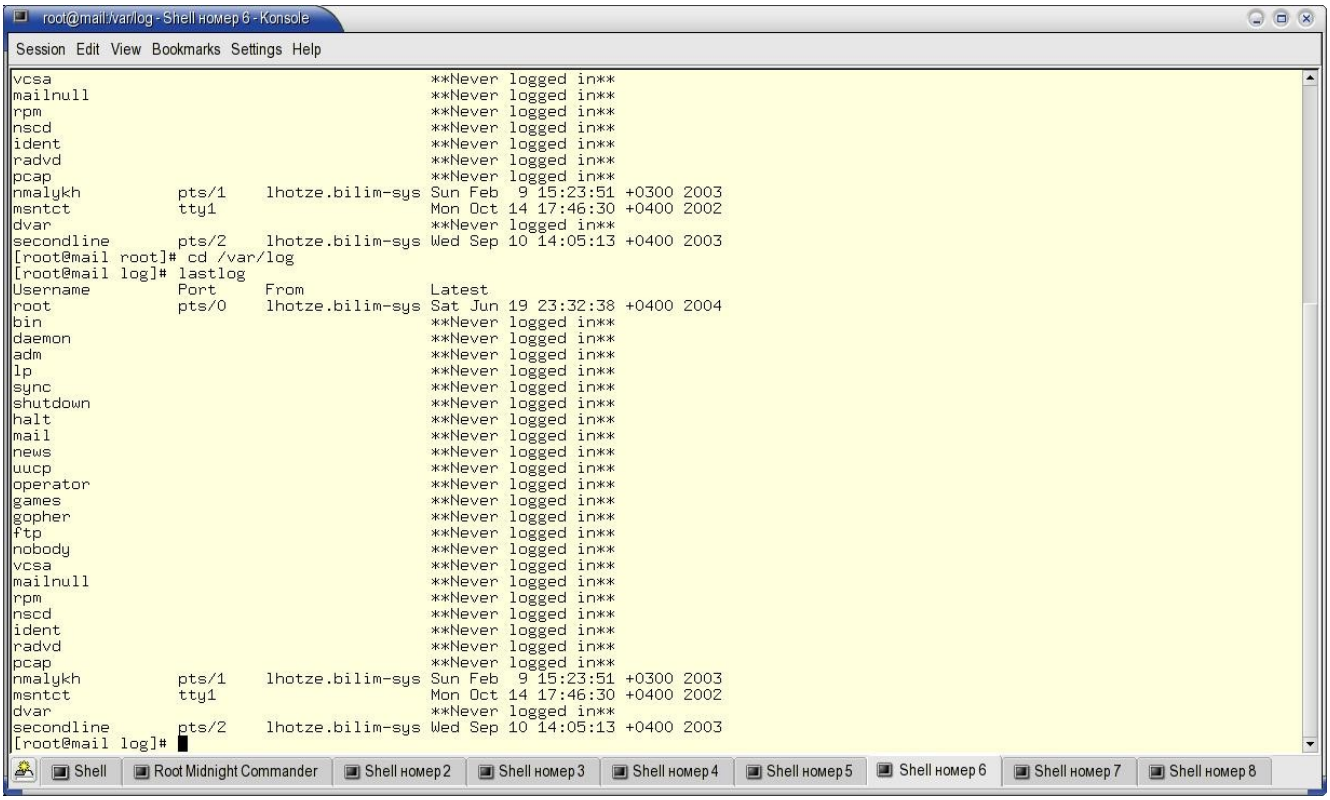


Рисунок 2.10. Вывод команды lastlog

2.8.1.2 last

Утилита **last** просматривает в обратном направлении (от конца к началу) системный журнал wtmp[1] и выводит сведения о всех фактах регистрации каждого пользователя в системе с момента создания журнального файла. На рисунке 2.11 вы можете видеть пример выводимой по команде last информации. В каждой строке вывода содержатся следующие данные:

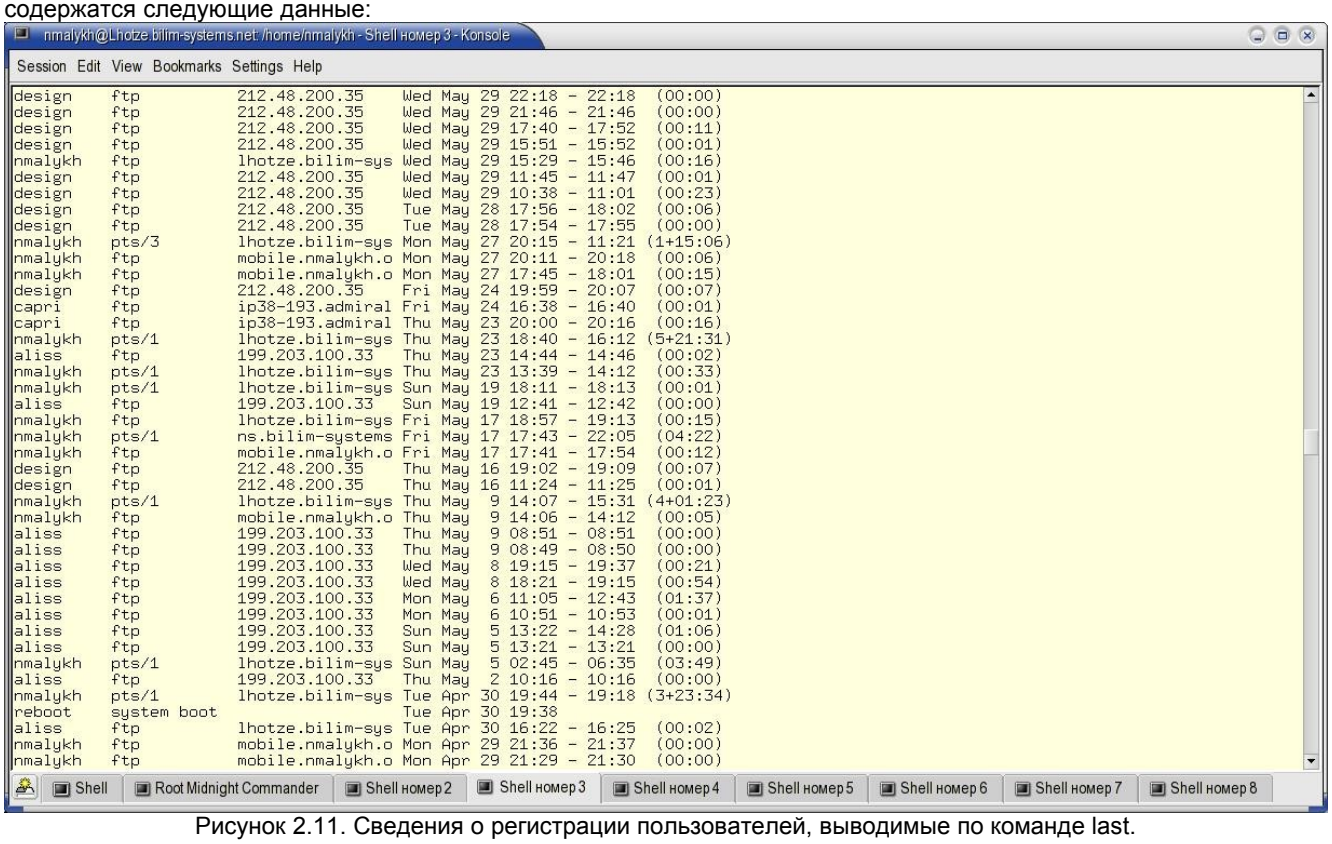


Рисунок 2.11. Сведения о регистрации пользователей, выводимые по команде last.

- имя пользователя;
- терминал или служба, использованные для входа в систему;
- IP-адрес или имя хоста, использованные в данном сеансе;
- дата и время начала сеанса, время завершения сеанса;
- продолжительность сеанса.

1 *Или иной файл, указанный параметром -f.*

При продолжительном использовании системы в файле **wtmp** может накопиться очень много информации и список получится длинным. Для получения сведений о регистрации того или иного пользователя можно ввести команду

last <имя пользователя>

Команда last в Linux-системах поддерживает ряд опций, управляющих выводом информации

-num

задает число строк вывода. Может использоваться в сокращенном формате **-n**.

-t YYYYMMDDHHMMSS

показывает регистрацию пользователей для заданного момента. Эта опция может оказаться весьма полезной при обнаружении несанкционированных действий.

-R

подавляет вывод информации об именах хостов.

-a

перемещает имя хоста в последнюю колонку. Может использоваться совместно с опцией -d.

-d

при регистрации удаленных пользователей Linux сохраняет не только имя хоста, но и его IP-адрес. Использование опции -d обеспечивает преобразование сохраненных IP-адресов в имена хостов.

-i

эта опция подобна опции **-d**, но выводит IP-адреса.

-o

используется для работы со старыми файлами wtmp (linux-libc5).

-x

выводит сведения о перезагрузке системы и изменении **run level**.

Формат файла **wtmp** кратко описан а Приложении 12.14.

2.8.1.3 Обход журналов регистрации пользователей

Поскольку в файлах lastlog и wtmp хранятся все сведения о регистрации пользователей, атакующие часто предпринимают попытки изменения регистрационных журналов или их уничтожения в целях сокрытия следов своей деятельности. Существует множество программ, которые могут использоваться для обмана или обхода регистрационных журналов Linux[1]. Большинство программ заметания следов регистрации пользователя в системе рассчитано на работу со стандартными журналами регистрации, поэтому весьма эффективной мерой защиты будет использование дополнительных[2] средств записи сведений о регистрации пользователей в системе. Маловероятно, что взломщики будут предполагать наличие дополнительных средств журнализации и даже очень предусмотрительным злоумышленникам придется сначала разобраться с нестандартными средствами, чтобы замести следы своего пребывания.

2.8.2 xferlog

В файле xferlog сохраняются сведения о передаче файлов по протоколу FTP, получаемые от демона ftpd. Каждая запись файл включает сведения об отдельной транзакции FTP:

- дата и время операции;
- продолжительность передачи файла;
- имя хоста или его IP-адрес;
- размер переданного файла в байтах;
- имя файла;
- тип передачи (binary/ASCII);
- выполненные при передаче специальные операции (например, архивирование);
- направление передачи (i - на сервер, o - с сервера);
- режим доступа (a - анонимный, g - гость, r - зарегистрированный пользователь);
- имя пользователя;
- использованный сервис;
- метод аутентификации;
- идентификатор аутентифицированного пользователя.

Файл xferlog имеет текстовый формат и для работы с ним вы можете воспользоваться любой программой просмотра и редактирования текстов.

1 *Ряд таких программ вы сможете найти на сайте http://www.antiserver.it/Unix/Log-Wipers/, а в каталоге /SRC/ приложенного к курсу компакт-диска имеются исходные тексты программ cloak.c и cloak2.c.*

2 *Вы можете написать такие программы самостоятельно или воспользоваться готовыми программами сторонних производителей, не входящими в стандартные дистрибутивы.*

2.8.3 Системный журнал messages

В UNIX-системах файл **/var/log/messages**[1] используется для записи сообщений, выдаваемых демонами **syslogd** и **klogd**. Сообщения от демонов ядра записываются в файл в порядке их поступления, как вы можете видеть на рисунке 2.12.

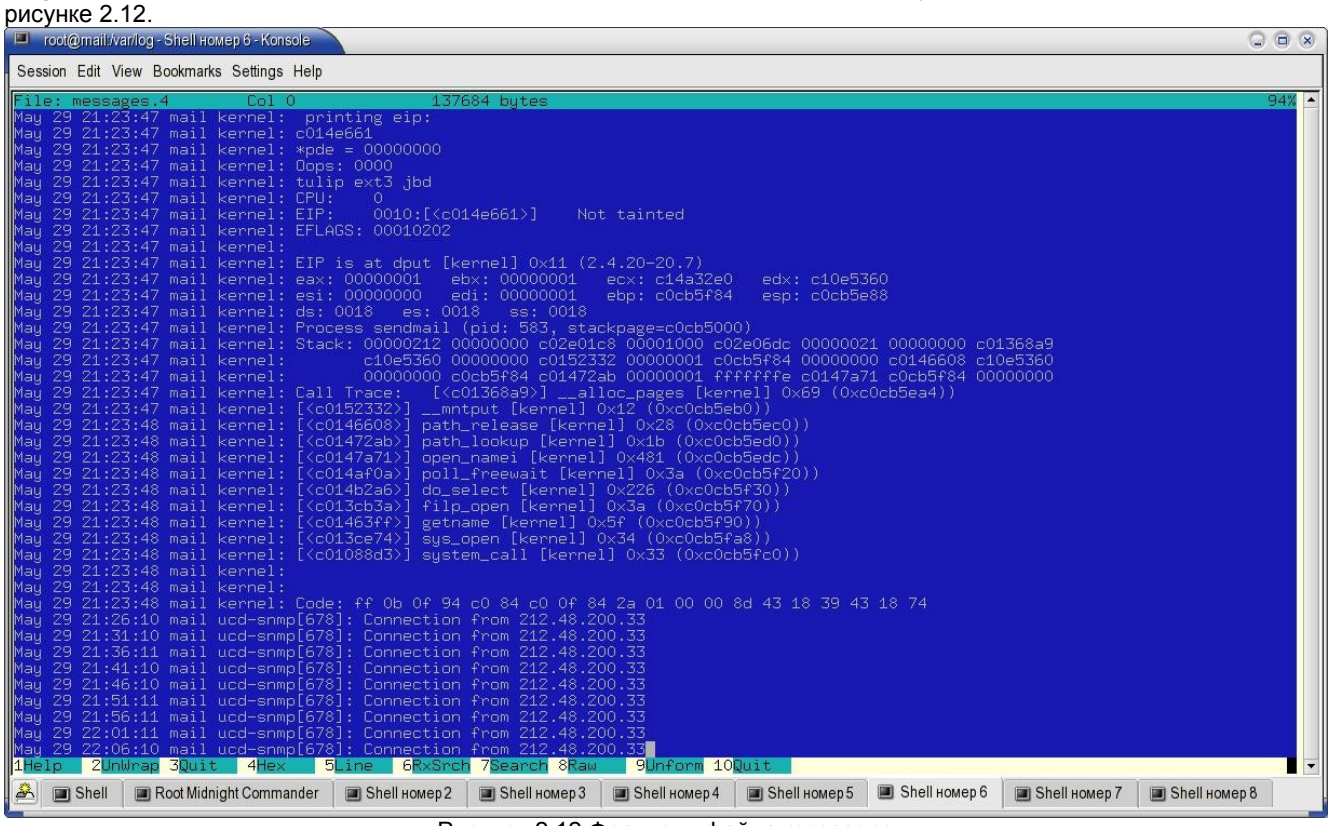


Рисунок 2.12 Фрагмент файла messages

В файл messages записываются сообщения от ядра, службы syslog, сетевых служб и другая информация, определяемая конфигурацией службы syslog, заданной в файле syslog.conf. На рисунке 2.13 вы можете видеть сообщения об отбрасывании пакетов фильтрами iptables на межсетевом экране.

2.8.4 Настройка syslog

Для настройки журнализации, выполняемой с помощью системы syslog следует указать правила журнализации в

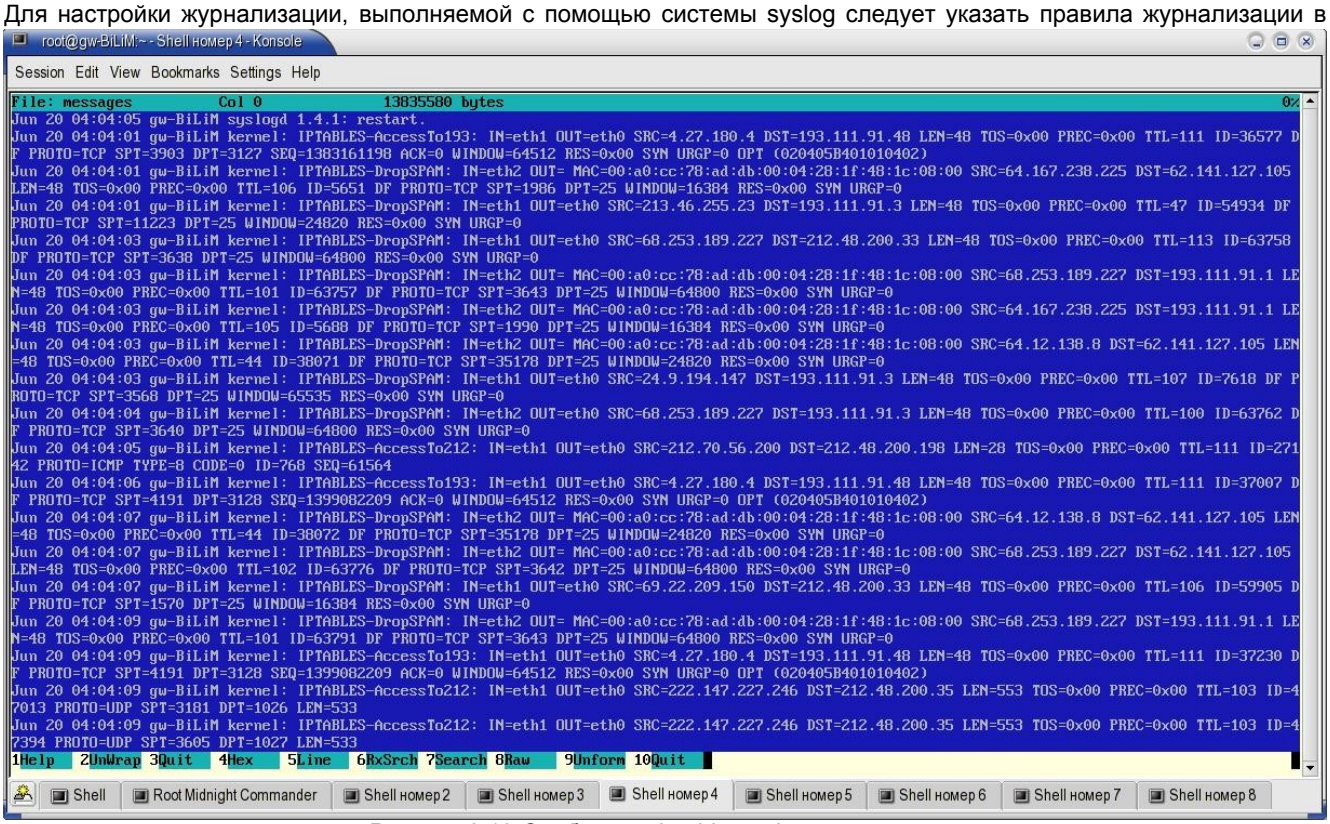


Рисунок 2.13 Сообщения iptables в файле messages

файле /etc/syslog.conf. Ниже показан фрагмент файла syslog.conf для хоста Mandrake Linux.

1 *В некоторых системах файл messages может храниться в другом каталоге (например, /var/adm).*

```
# Various entry
auth,authpriv.*                                 /var/log/auth.log
*.*;auth,authpriv.none                          -/var/log/syslog
user.*                                          -/var/log/user.log

# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;;news.none;authpriv.none       -/var/log/messages

# The authpriv file has restricted access.
authpriv.*                                      /var/log/secure

# Mail logging
mail.=debug;mail.=info;mail.=notice             -/var/log/mail/info
mail.=warn                                      -/var/log/mail/warnings
mail.err                                        -/var/log/mail/errors
mail.=notice                                    -/var/log/mail/notice

# Cron logging
cron.=debug;cron.=info;cron.=notice             -/var/log/cron/info
cron.=warn                                      -/var/log/cron/warnings
cron.err                                        -/var/log/cron/errors

# Kernel logging
kern.=debug;kern.=info;kern.=notice             -/var/log/kernel/info
kern.=warn                                      -/var/log/kernel/warnings
kern.err                                        /var/log/kernel/errors
```

Каждая строка файла, не являющаяся комментарием[1] содержит 2 поля - поле селектора размещается в начале строки и указывает, что нужно записывать, а поле назначения указывает куда записывать информацию.

Поле селектора должно содержать по крайней мере одно из двух значений:

- тип сообщения;
- приоритет сообщения (уровень протоколирования).

2.8.4.1 Типы сообщений

Поле типа сообщения[2] может содержать одно из следующих значений:

- **auth** - устаревший способ записи сообщений системы аутентификации и защиты, используемой для аутентификации пользователей в различных службах системы (FTP, login и т. п.), требующих ввода имени и пароля для доступа к тому или иному ресурсу. В современных системах рекомендуется использовать взамен auth объект authpriv.
- **authpriv** - современный вариант записи сообщений системы защиты и аутентификации.
- **cron** - сообщения от демонов **crond** и **atd**, выполняющих команды по указанному расписанию.
- **daemon** - сообщения от других демонов системы.
- **ftp** - сообщения от демона **ftpd**.
- **kern** - сообщения ядра Linux.
- **local0** - **local7** - эти имена зарезервированы для локального использования. Вы можете связать с ними те или иные сообщения от своих программ.
- **lpr** - сообщения от системы сетевой печати.
- **mail** - сообщения от почтовой системы.
- **news** - сообщения от системы обмена новостными сообщениями USENET.
- **syslog** - внутренние сообщения демона **syslogd**.
- **user** - базовые сообщения пользовательского уровня. Этот тип сообщений используется по умолчанию.
- **uucp** - сообщения почтовой системы UUCP.

Знак * используется для обозначения всех типов сообщений.

2.8.4.2 Уровни протоколирования

Вторая часть поля селектора определяет приоритет сообщений (уровень протоколирования). Поле приоритета может принимать одно из значений, перечисленных ниже в порядке снижения уровня приоритета (важности).

- **emerg** (0) - непредвиденная ситуация, влекущая за собой серьезные последствия вплоть до остановки работы системы.

1 *Строки комментариев начинаются знаком #*
2 *В английском языке для обозначения типа сообщений используется термин **facility** (устройство, объект).*

- **alert** (1) - событие, требующее немедленной реакции.
- **crit** (2) - критически важное сообщение, говорящее о возникновении в системе серьезных проблем.
- **err** (3) - сообщение об ошибке, не имеющей фатальных последствий. Обычно такие сообщения выдаются в стандартный поток ошибок.
- **warning** (4) - предупреждение, выдаваемое в тех случаях, когда система или ресурс не могут выполнить запрошенные действия.
- **notice** (5) - уведомление о возникновении ситуации, которая может иметь важные последствия.
- **info** (6) - информационное сообщение. Такие сообщения обычно содержат информацию, выдаваемую пользователю прикладными программами.
- **debug** (7) - отладочные сообщения работающих в системе процессов.

Существует два специальных случая для обозначения уровня приоритета:

***** - сообщения с любым уровнем;

none - никаких сообщений.

В записи селектора сначала указывается тип сообщения, потом ставится точка и указывается уровень протоколирования. Перед идентификатором уровня могут использоваться знаки:

= - только сообщения с заданным уровнем приоритета;

! - знак инверсии (все уровни, кроме указанных справа).

Например, селектор

```
kern.*
```

говорит о необходимости записи всех сообщений от ядра, а

```
*.alert
```

о необходимости записи предупреждений всех типов.

Селектор **daemon.!=debug** указывает на необходимость записи от демонов всех сообщений, имеющих уровень приоритета, отличный от **debug**.

2.8.4.3 Назначение сообщений

Поле назначения, расположенное в правой части строки правила syslog, указывает, куда следует передать полученное сообщение. В качестве назначения могут использоваться:

- именованные каналы;
- терминалы (локальные или удаленные);
- удаленные хосты с демоном syslogd;
- указанные пользователи;
- все пользователи.

Например, строка

```
mail.err                                        -/var/log/mail/errors
```

указывает необходимость записи всех сообщений почтовой системы с уровнем приоритета **err** и выше в файл /var/log/mail/errors, а строка

```
mail.=notice                                    -/var/log/mail/notice
```

говорит о необходимости записи сообщений почтовой системы с уровнем **notice** в файл /var/log/mail/notice.

Знак - перед именем файла говорит системе syslog о том, что не следует выполнять синхронизацию для этого файла после каждой записи. Если сообщения нужно передать на другой хост, в качестве назначения указывается имя этого хоста с префиксом **@** (например, **@loghost**). Для передачи сообщений определенному пользователю в поле назначения указывается просто имя этого пользователя, а для передачи сообщений всем пользователям - символ *****.

3 Готовые решения на базе специализированных дистрибутивов Linux

Если тебе дадут линованную

бумагу, пиши поперек.

Хуан Рамон Хименес

Сегодня не составит труда найти дистрибутив Linux, специально подготовленный для организации межсетевых экранов. Выпускаются как дистрибутивы с открытыми кодами, в которых подобран набор программ и сценариев для создания и поддержки межсетевых экранов, так и специализированные бинарные дистрибутивы, оптимизированные для работы в качестве брандмауэров и поставляющиеся без исходных кодов[1]. Как правило, такие дистрибутивы достаточно хорошо документированы, поэтому мы не будем здесь подробно рассматривать такие платформы, а лишь дадим краткий обзор некоторых известных решений.

3.1 MandrakeSecurity MNF

http://www.mandrake.com

Дистрибутив **MandrakeSecurity Multi Network Firewall**[2] представляет собой специализированный 1-дисковый дистрибутив Linux для использования на компьютерах с процессором i586 или выше. MandrakeSecurity MNF объединяет в себе функции межсетевого экрана с поддержкой виртуальных частных сетей VPN, системы детектирования попыток вторжений (IDS[3]) и системы управления трафиком. Web-интерфейс, построенный с использованием безопасных протоколов и поддерживающий мощный набор “мастеров настройки” позволяет сетевым администраторам достаточно быстро развертывать межсетевые экраны, определять и поддерживать политику доступа, систему фильтрации и мониторинга трафика Internet, а также решать другие задачи сетевого администрирования и обеспечения безопасности.

Эффективная система поддержки VPN-соединений обеспечивает возможность безопасной работы для неограниченного числа клиентов VPN.

MNF работает на базе ядра Linux 2.4 и обеспечивает поддержку множества виртуальных частных сетей и демилитаризованных зон (DMZ). Высокопроизводительные средства шифрования IPSec позволяют интегрировать брандмауэры MNF в существующие сетевые инфраструктуры и работать с клиентами различных ОС, включая MS Windows™.

Средства настройки предоставляют администратору легко создать систему подавления баннерной рекламы, организовать фильтрацию пакетов и систему управления трафиком. Для разных пользователей могут применяться различные уровни ограничений при доступе к ресурсам Internet. Брандмауэр MNF поддерживает эффективный набор средств мониторинга, предоставляющих администратору полную информацию об активности пользователей и происходящих в системе событиях.

3.2 Trustix

http://www.trustix.net

Идея создания Trustix Secure Linux родилась в 1999 г., когда его будущие разработчики осознали потребность в специализированном дистрибутиве, из которого исключены все избыточные программы, службы и офисные приложения, присутствующие в системах Linux общего назначения. Первая стабильная версия Trustix Secure Linux v0.80 была выпущена 21 февраля 2000 г. Под кодовым названием PaperJoe.

В 2003 компания Comodo осознала необходимость включения безопасной ОС в предлагаемую компанией линейку продукции и приобрела компанию Trustix AS, создав на ее базе Comodo Trustix. В результате использования Trustix Secure Linux в глобальной системе услуг обеспечения безопасности Comodo Group продуктивность разработчиков существенно повысилась и дистрибутив обрел ряд целый новых функций и возможностей.

Сегодня Trustix Secure Linux представляет собой эффективную и надежную серверную ОС, обеспечивающую высокий уровень безопасности для корпоративных сетей и не содержащую избыточных программ. Типичная инсталляция требует около 100 Мбайт дискового пространства, а полная[4] занимает на диске около 900 Мбайт. При полной инсталляции устанавливаются все программы (некоторые из них могут вам никогда не потребоваться), но многие службы не запускаются по умолчанию, если они не требуются для решения конкретных задач обеспечения безопасности.

Современный вариант **Trustix Secure Linux 2.1**[5] распространяется в виде ISO-образов 2 компакт дисков (общий размер составляет около 650 Мбайт). Основными отличиями новой версии являются:

- Swup 2.3 - программа автоматического обновления с предотвращением возможности использования подставных зеркал;
- Samba 3.0.2
- Perl 5.8.3

1 *Некоторые из таких дистрибутивов являются коммерческими или условно бесплатными (например, бесплатный дистрибутив и коммерческая служба поддержки и т. п.).*
2 *Вы можете загрузить образ ISO этого дистрибутива с сайта http://www.mandrakelinux.com или с одного из многочисленных зеркал, ссылки на которые приведены на странице загрузки указанного сайта.*
3 *Intrusion Detection System*
4 *Разработчики не рекомендуют использовать полную инсталляцию.*
5 *Trustix можно загрузить с любого из зеркал, указанных на сайте http://www.trustix.net/mirrors/.*

- PHP 4.3.4
- Mysql 4.0.18
- Postfix 2.0.18
- MIT Kerberos v5
- DKMS
- Postgresql 7.4.2
- OpenSSH 3.8p1
- поддержка файловых систем XFS с функциями ведения журнала.

3.3 EnGarde Secure Linux

http://www.guardiandigital.com

EnGarde Secure Linux представляет собой полнофункциональную серверную платформу, обеспечивающую поддержку функций DNS, Web и почтового сервера. Использование EnGarde снижает расходы и затраты времени на развертывание безопасных систем с публичным доступом.

Платформа EnGarde разработана компанией Guardian Digital на основе программ с открытым исходным кодом, скомпилированных специалистами Guardian Digital с учетом всех современных требований безопасности. EnGarde обеспечивает безопасность в сочетании с надежностью, производительностью и эффективным управлением.

- Система Secure Web обеспечивает полный доступ к функциям управления Linux с соблюдением мер безопасности.
- Поддерживается управление сервером DNS, виртуальными серверами Web, электронной почтой и другими стандартными службами с использованием Web-интерфейса.
- Обеспечивается эффективная защита от всех типов атак и вторжений со стороны Internet.
- Брандмауэр обеспечивает защиту внутренней сети от внешних злоумышленников.
- Многие традиционные приложения заменены более современными и безопасными программами, существенно повышающими общий уровень безопасности системы и защищаемой ею сети.
- Система аудита EnGarde обеспечивает в реальном масштабе времени анализ журнальных файлов и их статистическую обработку.
- Поддержка программных и аппаратных функций RAID, возможность использования нескольких процессоров и файловая система с поддержкой функций ведения журнала обеспечивают высокую производительность системы.
- Центр управления Security Control Center обеспечивает администратору возможность постоянного мониторинга системы, тонкой настройки системы контроля доступа, управления ключами SSH и SSL и т. п.
- Система детектирования попыток вторжения позволяет строить на базе этого дистрибутива эффективные IDS.
- Встроенные средства защиты от троянских программ, переполнения буфера и т. п.
- Автоматическое обновление программ.
- Поддержка широкополосных соединений с использованием DSL или кабельных модемов.
- Встроенная система резервирования и восстановления предотвращает потерю данных.
- Открытая платформа позволяет использовать приложения, написанные участниками проектов open source.

3.4 Euronode

http://euronode.org

Euronode представляет собой серию дистрибутивов на базе Debian GNU/Linux (Woody Release 2), позволяющих за короткое время разворачивать и настраивать межсетевые экраны или высокопроизводительные серверы.

Euronode включает набор сценариев автоматизированной установки и настройки, включающих автоматическое детектирование устройств, разбиение дисков на разделы и т. п. Подробная документация дистрибутивов Euronode позволяет администратору легко и быстро настроить систему в соответствии с реальными задачами.

Выпускается 3 варианта дистрибутива:

- **Euronode Minimal Woody** - минимальный комплект, обеспечивающий только базовые функции и включающий 186 программных пакетов, которые занимают на диске около 210 Мбайт;
- **Euronode Simple Firewall** - простой и безопасный шлюз доступа в Internet, поддерживающий функции автоматического определения кабельных и DSL-модемов с портами Ethernet и USB;
- **Euronode Advanced Firewall** - простой брандмауэр с поддержкой фильтрации электронной почты в целях предотвращения вирусов и спама.

Euronode автоматически разбивает диск и форматирует созданные разделы, определяет компоненты ПК (порты USB, сетевые адаптеры и т. п.). Минимальный вариант дистрибутива включает.

- ядро Linux версии 2.4.25;
- операционную систему на базе Debian Woody release 2 [21 ноября 2003], обновленного 29 февраля 2004;
- исправленные версии программ adduser, iptables, procps, e2fsprogs, lsof и др.;
- 186 пакетов, занимающих на диске 210 Мбайт.

3.4.1 Euronode Simple Firewall

Простой шлюз доступа в Internet включает

- **базовый дистрибутив Euronode Minimal Woody**;
- межсетевой экран с поддержкой технологии Stateful Inspection;
- поддержку функций автоматического определения модемов (DSL и кабельные) с портами Ethernet и USB;
- автоматическое определение сетевых адаптеров и устройств SCSI;
- средства автоматической настройки конфигурации системы и доступа в Internet;
- множество полезных для администратора приложений (vim, ntpdate, nmap, mc, netdiag, tcputils, tcpdump и т. п.);
- систему управления с Web-интерфейсом;
- поддержку прозрачного proxy-кэширования;
- 236 пакетов, занимающих на диске 310 Мбайт.

3.4.2 Euronode Advanced Firewall

межсетевой экран **Euronode** включает

- **Euronode Simple Firewall**;
- почтовый сервер Postfix с программами фильтрации вирусов (Clamav) и спама (Spamassassin);
- сервер Fetchmail (POPa3d) для поддержки пользовательских почтовых ящиков;
- модули Webmin для управления Postfix, Spamassassin, Fetchmail и др. Службами;
- 268 пакетов, занимающих на диске 350 Мбайт.

3.4.3 FrazierWall Linux

Специализированный дистрибутив **FrazierWall Linux** был разработан на базе Linux Router Project[1] и Coyote Linux v1.03[2] для использования в качестве межсетевого экрана.

FrazierWall использует ядро Linux 2.2.18 и поддерживает маскирование адресов IP (NAT Routing). Дистрибутив содержит предустановленный набор правил для организации межсетевого экрана и включает широкий набор средств настройки и мониторинга с дружественным пользовательским интерфейсом. Поддерживается сервер DHCP для динамического выделения IP-адресов пользователям из внутренней сети.

3.5 Immunix

http://www.immunix.org

Специализированный дистрибутив Immunix Secured OS 7.3 создан на базе ядра 2.4 с использованием glibc 2.2.5 и GCC 2.96. Операционная система включает модуль контроля доступа SubDomain, систему предотвращения переполнения буфера StackGuard и систему контроля формата строк FormatGuard. Система SubDomain поставляется с предопределенными профилями, которые включают более 30 базовых вариантов и дополнительные профили для обеспечения безопасной работы приложений.

Immunix Secured OS 7.3 не является бесплатной системой и включает компоненты, распространяемые по лицензии GPL, наряду с фирменными модулями, распространяемыми на основе лицензии Immunix Commercial License[3].

Immunix Secured OS 7.3 включает:

- компилятор StackGuard 3, обеспечивающий повышение уровня совместимости и безопасности приложений. Стандартные версии программ GDB, gprof, Mozilla (с plug-in), виртуальные Java-машины (JIT-style Java JVM) работают в библиотеками и кодом StackGuard без каких-либо ограничений.
- FormatGuard предотвращает атаки с использованием некорректно отформатированных строк для стандартных функций языка C.
- ядро Linux 2.4 с интерфейсом LSM (Linux Security Modules[4]).
- LSM-модуль SubDomain для повышения уровня безопасности при использовании приложений:
    - 30 предопределенных профилей для основных служб и программ (для некоторых служб предлагается по несколько профилей).
    - Правила SubDomain позволяют использовать регулярные выражения (например, правилу /usr/lib/python[12].[0-9]/**.{py,pyc} будут соответствовать библиотеки python независимо от номера версии и расширения имени файлов).
    - Существенно повышена скорость работы SubDomain.

1 *Дополнительную информацию вы сможете найти на официальном сайте проекта - http://www.linuxrouter.org.*
2 *Дискетный вариант Linux (см. http://www.coyotelinux.com).*
3 *Текст лицензии см. на сайте http://www.immunix.org/Immunix_Commercial_License*
4 *См. описание этого интерфейса на сайте http://lsm.immunix.org/.*

- Сервис обновления программ up2date.

3.6 IPCop Firewall

http://www.ipcop.org

Дистрибутив IPCop v1.3.0, выпущенный 22 апреля 2003, построен на основе ядра Linux 2.4 и поддерживает использование программы iptables, обеспечивающей эффективные средства фильтрации пакетов и контроля соединений.

3.7 NSA Security Enhanced Linux

http://www.nsa.gov/selinux

Агентство национальной безопасности США (National Security Agency) активно занимается исследованиями в области компьютерной безопасности, включая вопросы безопасности операционных систем. Признавая критическую роль механизмов обеспечения безопасности ОС, специалисты NSA приложили много усилий по исследованию архитектуры, обеспечивающей высокий уровень безопасности при различных вариантах использования компьютеров (серверы, настольные системы и т. п.).

Конечные системы должны быть способны разделять информацию на основе требований целостности и конфиденциальности, чтобы обеспечить требуемый уровень безопасности системы в целом. Механизмы обеспечения безопасности операционной системы являются основой для такого разделения информации. К сожалению, большинство современных ОС не поддерживает одного из основных механизмов обеспечения безопасности - системы контроля доступа. Как следствие этого механизмы безопасности прикладных программ являются весьма уязвимыми, поскольку их можно без особого труда обойти.

В результате реализации нескольких исследовательских проектов усилия разработчиков были сосредоточены на разработке Linux-системы с повышенным уровнем безопасности. Linux обеспечивает мощную и гибкую систему контроля доступа, встроенную в ядро ОС. Эта операционная система обеспечивает эффективные механизмы разделения информации на основе требований конфиденциальности и целостности, не позволяющие обойти механизмы безопасности приложений.

На выбор Linux в качестве базовой платформы оказал влияние и успех открытых платформ на современном рынке операционных систем. Интеграция результатов исследований специалистов по информационной безопасности в открытую платформу может привести к новому витку исследований сообщества Open Source в сфере безопасности открытых систем и соответствующему росту уровня безопасности ОС.

SE Linux не является окончательным решением вопросов безопасности ОС Linux и даже попыткой исправления возможных уязвимостей Linux-систем. Этот вариант Linux просто служит примером использования обязательной системы контроля доступа для управления работой всех процессов в системе, включая процессы, выполняющиеся от имени пользователя с неограниченными правами (root). SE Linux не содержит переработанных механизмов общего обеспечения безопасности системы или аудита безопасности, хотя эти элементы также достаточно важны.

Реализованные в SE Linux механизмы обеспечения безопасности поддерживают широкий выбор вариантов политики безопасности, позволяющих настроить параметры безопасности системы для различных вариантов ее использования. Гибкость системы управления политикой безопасности и встроенные варианты политики для типовых применений позволяют администратору без больших усилий адаптировать систему с учетом реальных требований безопасности для каждого хоста.

SE Linux по сути представляет собой переработанный специалистами NSA вариант ядра Linux в комплекте с набором программ пользовательского пространства. На сайте NSA представлены различные варианты дистрибутива, среди которых вы можете выбрать наиболее подходящий для решения ваших задач.

3.8 OpenNA Linux

http://www.openna.com

OpenNA Linux представляет собой распространяемый по лицензии GPL дистрибутив Linux для процессоров с архитектурой 686. Безопасная, быстрая и современная реализация ОС Linux предназначена для использования на серверах, требующих высокого уровня безопасности.

OpenNA Linux поддерживает несколько предопределенных вариантов установки для различных типов серверов (например, web). Это избавляет начинающих администраторов от тяжких раздумий при выборе программ для установки на диск сервера. Программа инсталляции автоматически выберет и установит все требуемые для работы выбранного типа сервера приложения и компоненты ОС. К числу предопределенных типов серверов относятся:

- Web
- FTP
- DNS
- Сервер электронной почты (E-Mail)
- Сервер баз данных
- Шлюз
- Виртуальный сервер

Каждая из активизируемых на сервере служб будет настроена в процессе инсталляции в соответствии с вашими потребностями. Это значит, что вам не придется долго редактировать конфигурационные файлы, настраивать сетевые параметры и т. п. Все конфигурационные файлы сервера будут оптимизированы с точки зрения безопасности и производительности.

Для каждого из предопределенных типов серверов поддерживается возможность установки дополнительных служб и требуемых для их использования и настройки приложений. Кроме серверных вариантов поддерживается также установка OpenNA Linux для использования компьютера в качестве рабочей станции, обеспечивающая полнофункциональный набор приложений и средств разработки.

3.9 Openwall GNU/Linux

http://www.openwall.com/Owl/

Owl (Openwall GNU/*/Linux) представляет собой серверную платформу с повышенным уровнем безопасности, созданную на базе Linux и программ GNU. Этот бесплатный вариант дистрибутива совместим с большинством реализаций GNU/*/Linux.

Платформа Owl включает интегрированный набор служб Internet и комплект средств разработки, позволяющий заново построить всю систему из исходных кодов с помощью одной команды **make buildworld**. Платформа Owl реализована для различных вариантов архитектуры, включая x86, SPARC и Alpha.

Безопасность

Основным преимуществом Owl с точки зрения безопасности является тщательный аудит исходных кодов с точки зрения наличия уязвимостей. Наиболее тщательной проверке подвергается код системных библиотек, программ с атрибутами SUID/ SGID (см. параграф 2.3.3 на стр. 42), демонов и сетевых служб. При создании дистрибутива Owl программные компоненты конфигурируются и при необходимости изменяются для достижения максимального уровня безопасности. Специальные средства администрирования (owl-control) обеспечивают управление компонентами системы, оказывающими существенное влияние на уровень безопасности. Каждый программный пакет, включенный в состав Owl документирован по результатам аудита.

Аудит исходных текстов программ является необходимой, но недостаточной мерой предотвращения программных уязвимостей. Пользователь всегда может установить программы других фирм или собственной разработки, не прошедшие должного контроля, поэтому в Owl применяются способы ограничения привилегий для таких программ.

В Owl используются мощные криптографические механизмы для основных компонент и встроенные политики обеспечения безопасности, включающие упреждающую проверку паролей (pam_passwdqc), ограничение срока действия учетных записей и паролей, контроль доступа на основе сетевых адресов и средства проверки целостности (mtree).

Среда разработки и управления пакетами.

В отличие от многих дистрибутивов Linux платформа Owl включает полную среду разработки, позволяющую заново скомпилировать все компоненты системы с помощью одной команды **make buildworld**. Однако реализация **make buildworld** в Owl существенно отличается от аналогичных средств *BSD. Пожалуй эта операция больше похожа на портирование *BSD, перекрывающее все пользовательское пространство Owl (т. е., все программы, за исключением ядра Linux).

Исходные коды приложений пользовательского пространства Owl включают два дерева каталогов, обеспечивающие для каждого пакета из состава Owl два варианта. Одно дерево содержит оригинальные варианты исходных кодов от их разработчиков, а во втором дереве (репозиторий CVS), сохраняются файлы исправлений, спецификации компиляции и другие дополнения от Owl. Некоторые приложения, разработанные специально для Owl, могут полностью содержаться в репозитории CVS.

3.10 redWall

http://redwall.sourceforge.net

redWall представляет собой специализированный дистрибутив Linux, предназначенный для использования в качестве межсетевого экрана. Дистрибутив распространяется в форме загружаемого компакт-диска, который используется для работы брандмауэра. Управление обеспечивается с помощью Web-интерфейса.

- Конфигурационные параметры сохраняются на дискете, USB-диске (Memory Stick) или винчестере, а в будущих версиях их можно будет передавать по электронной почте.
- Большая часть функций[1] генерации отчетов реализована на базе mysql (за исключением отчетов squid).
- Компакт-диск с дистрибутивом можно использовать как консоль мониторинга и журнализации (Management/Logging Console) для других межсетевых экранов, работающих в аналогичной среде.
- Дистрибутив собран на основе RedHat 9.0.
- Поддерживаются функции моста.
- Обеспечивается фильтрация электронной почты на предмет спама и вирусов.
- Виртуальная файловая система (tmpfs) /etc с возможностью записи для конфигурационных параметров.
- Файловая система /var с возможностью записи (ram-диск или винчестер).

3.11 Securepoint Firewall & VPN

http://www.securepoint.cc

Основной задачей межсетевого экрана является предотвращение несанкционированного (нежелательного) доступа к ресурсам из локальной сети и извне. Платформа Securepoint обеспечивает требуемую защиту в сочетании с высоким уровнем производительности сервера. Система Securepoint Firewall представляет собой программное решение, которое может быть развернуто на любом ПК или специализированной аппаратной платформе. Строго говоря, Securepoint Firewall, не является бесплатной системой, но исходные коды основных ее компонент доступны

1 *Исключение составляют отчеты о работе прокси-сервера squid.*

по лицензии GPL и могут быть загружены с сайта.

3.12 SmoothWall

http://www.smoothwall.org

SmoothWall представляет собой специализированный дистрибутив Linux для использования в качестве межсетевого экрана/маршрутизатора на недорогих платформах x86. Основными задачами авторы считают:

- защиту локальной сети от внешних атак с минимальными затратами времени на настройку;
- простоту установки системы;
- поддержку широкого спектра сетевых адаптеров, модемов и другого оборудования;
- поддержка различных вариантов соединения с провайдерами Internet;
- простоту использования, настройки и управления, обеспечиваемую web-интерфейсом.

Платформа SmoothWall предназначена прежде всего для небольших компаний и домашнего использования и работает любом 32-разрядном процессоре совместимом с i386 (начиная с 486 и заканчивая Athlon или Pentium 4).

3.13 Devil-Linux

http://www.devil-linux.org

Devil-Linux - это специализированный дистрибутив Linux, обеспечивающий загрузку и работу с компакт-диска. Конфигурационные параметры могут сохраняться на дискете или flash-диске USB. Платформа Devil Linux разрабатывалась для использования в качестве маршрутизатора-брандмауэра, но сейчас ее можно использовать и в качестве сервера. Devil-Linux позволяет легко подключать дисковые разделы, а дистрибутив включает большинство сетевых служб.

Для использования системы требуется привод CD-ROM и защищенная от записи дискета, на которой хранятся параметры конфигурации.

4 Межсетевой экран на базе дистрибутива Linux общего назначения

Не боги горшки обжигают.

Для организации межсетевых экранов, систем обнаружения попыток вторжения, сетевых серверов можно использовать в качестве основы какой-либо дистрибутив Linux общего назначения. Для более эффективной работы компьютера вам потребуется установить на нем соответствующие вашим задачам приложения, без раздумий отбрасывая все лишнее. Кроме того, крайне желательно для такого хоста или маршрутизатора самостоятельно собрать ядро из дистрибутива, выбрав опции компиляции ядра в соответствии с предполагаемым использованием компьютера. Если вы также самостоятельно соберете из исходных текстов наиболее важные компоненты приложения Linux, уровень безопасности вашей сети будет полностью соответствовать вашим ожиданиям.

4.1 Установка Linux

Большинство дистрибутивов Linux включает интерактивную программу инсталляции, которая обеспечивает автоматическое или полуавтоматическое выполнение всех задач по подготовке, выбору, установке и настройке программ с учетом установленного в компьютере оборудования. Поскольку в этом разделе мы рассматриваем создание межсетевого экрана на базе дистрибутива общего назначения, предполагается, что у вас уже есть опыт инсталляции Linux и нет необходимости в использовании сценариев автоматической установки.

Отказ от автоматической установки потребует большего времени, но обеспечит более оптимальную работу межсетевого экрана и повысит уровень безопасности системы. Кроме того, при автоматической инсталляции будет установлено столько ненужных на брандмауэре приложений, что их удаление отнимет много больше времени, нежели сэкономит автоматическая инсталляция.

4.1.1 Разбиение дисков

Для разбиения диска на разделы обычно используется программа fdisk с текстовым интерфейсом или DiskDruid с системой меню. Выбор той или иной программы - дело вкуса, - поскольку результаты будут одинаковыми. Более подробное описание программ для разбиения дисков приводится в параграфе 2.1.1 (стр. 29)

Рассмотрим лучше вопросы распределения дискового пространства для использования Linux-компьютера в качестве межсетевого экрана/маршрутизатора.

4.1.1.1 Корневой раздел

Размер корневого раздела лучше выбрать минимальным. Как показывает опыт создания и эксплуатации межсетевых экранов на базе Linux для современных вариантов Linux вполне достаточно корневого раздела размером 250 - 300 Мбайт. Такого раздела вполне достаточно для записи всех файлов, которые должны находиться в корневом разделе. Например, на нашем корпоративном брандмауэре все файлы помещаются в разделе размером 250 Мбайт с учетом того, что на диске храниться 5 - 7 копий различных версий ядра и требуемых для их использования библиотек.

4.1.1.2 Область подкачки (swap)

Как уже обсуждалось ранее размер области подкачки определяется требованиями задач к оперативной памяти и наличием физического ОЗУ в компьютере. При современных ценах на память разумно установить для межсетевого экрана по крайней мере 512 Мбайт оперативной памяти. Если вы к этому добавите область подкачки такого же размера, этого будет достаточно практически для любой ситуации.

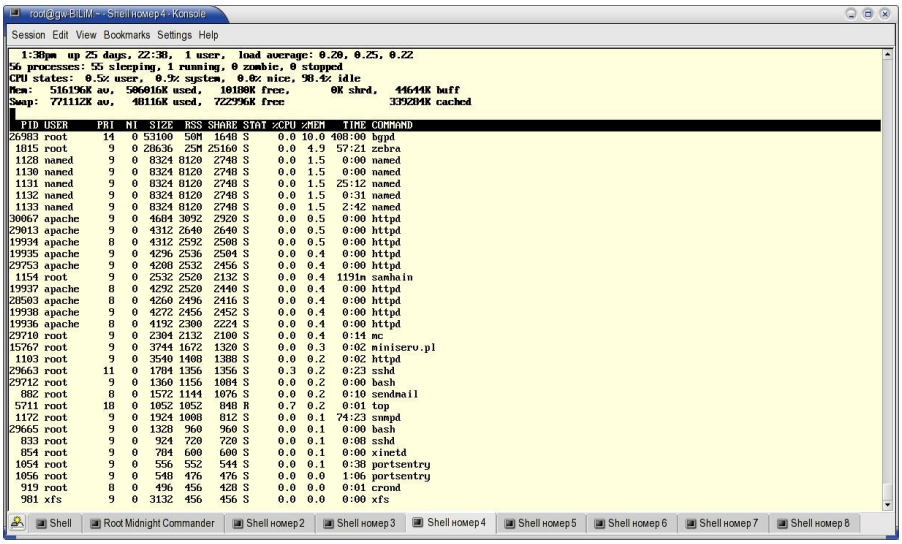


Рисунок 4.1 Распределение памяти на межсетевом экране Linux

Опыт использования граничного шлюза с мощной системой фильтрации пакетов, IDS, серверами электронной почты, DNS и HTTP, а также маршрутизатором BGP (zebra) оперативная память 512 Мбайт позволяет обходиться практически без использования области подкачки. На рисунке 4.1 показано распределение памяти для используемого в нашей компании межсетевого экрана. Вы можете видеть, что для подкачки при обычном режиме работы используется лишь около 48 Мбайт.

4.1.1.3 /usr

Файловая система usr используется для хранения программ пользовательского пространства и требуемых для их работы библиотек. В каталоге /usr/share обычно хранится множество файлов системной документации, а каталог /usr/src служит для записи исходных текстов программ перед их компиляцией. С учетом сказанного мне представляется целесообразным выделить для раздела /usr 3 - 5 Гбайт.

4.1.1.4 /var

Раздел /var для межсетевого экрана должен быть достаточно большим, поскольку в нем будут храниться журнальные файлы, а объем этих файлов только за одну неделю может увеличиваться на сотни Мбайт. Не следует жалеть место для раздела /var потому, что хранящаяся в нем информации может оказать значительную помощь при анализе работы системы или обнаружении несанкционированных действий. Мне представляется целесообразным создавать раздел /var размером не менее 5 Гбайт.

4.1.1.5 /tmp

Используемый для хранения временных файлов раздел /tmp не обязательно делать большим, поскольку объем временных файлов ОС Linux невелик. Однако при некоторых операциях (например, архивирование большого числа файлов или больших журнальных файлов) может потребоваться значительное пространство для записи временных файлов. Кроме того, раздел /tmp можно использовать как “помойку” для файлов одноразового использования. Не забывайте только периодически очищать эту “помойку”. В большинстве случаев для записи временных файлов будет вполне достаточно раздела размером 500 Мбайт.

4.1.1.6 /home

Раздел /home служит для создания домашних каталогов пользователей. Поскольку для межсетевых экранов число пользователей обычно измеряется единицами, нет смысла делать этот раздел большим.

4.1.2 Выбор программ для установки

При выборе приложений лучше исходить из необходимого минимума. При необходимости вы всегда сможете установить недостающую программу впоследствии, а вот ежели благодаря наличию на компьютере совершенно ненужной вам программы злоумышленник получит несанкционированный доступ в вашу систему, сокрушаться будет уже поздно. Кроме того, используемые программы лучше компилировать самому с учетом реальных потребностей, а при установке из дистрибутива вы получите скомпилированную для массового использования версию программы.

Для того, чтобы разобраться с установкой программ на межсетевом экране разобьем программы на несколько категорий и рассмотрим плюсы и минусы установки и использования на брандмауэре программ каждой категории. Основным критерием выбора программ для установки на межсетевом экране является их **абсолютная** необходимость и отсутствие известных уязвимостей. Зачастую администраторы (особенно начинающие) пытаются поставить все подряд (чтоб было под рукой), но это далеко не всегда оправдано. Приведу простейший пример: “Злой парень из террористической организации получил root-овый доступ к вашему брандмауэру, на котором присутствует подробно рассмотренная выше утилита fdisk (параграф 2.1.1.1). Ничтоже сумняшеся, он вводит команду fdisk и без особого труда удаляет вам таблицу разделов.” Грустная картина получается, не правда ли. Поэтому, прежде, чем оставить на диске брандмауэра ту или иную программу (очень нужную и полезную), попытайтесь представить себе, а какой урон злоумышленник может нанести вам с помощью этой программы. Еще один тривиальный пример - команда rm. Очень хорошая программа, но в руках дикаря или злоумышленника может оказаться страшным оружием. Пользователю с правами root достаточно ввести лишь две команды

```
cd /
rm -Rf
```

и вашего брандмауэра как не бывало.

Примеров таких можно привести множество, поэтому я еще раз повторю, что на диске имеет смысл сохранять только **абсолютно** необходимые программы. Если же рука ваша никак не поднимается удалить какую-либо из любимых утилит, спрячьте ее куда-нибудь на задворки файловой системы, чтобы переменная PATH не знала туда дороги. Если вы на оставите на видном месте каких-либо средств поиска файлов, злоумышленник может быть и не найдет вашу утилиту.

Требования к наличию на диске брандмауэра тех или иных программ с точки зрения безопасности и гибкости системы весьма противоречивы. Присутствие на диске некоторых программ, помогающих администратору в работе, может оказаться совершенно нежелательным с точки зрения безопасности, а их отсутствие может лишить администратора возможности выполнения целого ряда важных функций[1]. Как же быть? Выход из этого тупика существует. Соберите требуемый вам для обслуживания системы набор программ и поместите их на компакт-диск или USB flash. Размеры современных носителей более, чем достаточны для хранения “джентльменского набора администратора.” Сделав этот носитель загружаемым, вы обеспечите себя средствами восстановления даже при серьезных авариях, когда операционная система не может быть загружена без стороннего вмешательства.

Существует и другой вариант решения - разместить операционную систему и требуемый для работы межсетевого экрана набор программ на загружаемом компакт-диске и держать этот диск постоянно в приводе CD. Для записи конфигурационных параметров, журнальных файлов и т. п. можно в таких случаях установить в компьютере винчестер небольшого объема или записать конфигурационные файлы на flash-диск, а системные журналы передавать на сервер syslog. У этого варианта есть недостаток - сложность обновления, но при нынешних ценах на носители эта проблема не является непреодолимой.

При творческом подходе и склонности к анализу вы сможете выработать для себя верную стратегию размещения программ для работы и обслуживания системы.

4.1.3 Системные библиотеки

Значительная часть системных библиотек ставится в процессе инсталляции по умолчанию. Различные пакеты

1 *Вспомним уже упомянутую утилиту **rm**. Удалив ее с диска вы уже не сможете удалить ничего.*

мониторинга, управления и т. п. также могут добавлять те или иные библиотеки в систему. Настоятельно рекомендую вам внимательно ознакомиться с описаниями всех установленных библиотек и представить варианты их злонамеренного или неосторожного использования. Это потребует достаточно много времени, но в результате вы сможете удалить все то, без чего можно обойтись и существенно повысить уровень безопасности.

4.1.4 Системные утилиты

На маршрутизаторе или межсетевом экране польза от системных утилит несомненна, поскольку они позволяют оператору выполнять различные операции мониторинга и контроля состояния системы. Однако, не следует забывать, что этими же утилитами может воспользоваться проникший в систему супостат для реализации своих злокозненных намерений.

Вариант с установкой утилит на съемный носитель не слишком удобен, поскольку операции мониторинга и контроля приходиться вести постоянно, а межсетевой экран обычно располагается не в самом доступном месте вашего офиса. Мне представляется целесообразным следующее решение - из числа используемых для контроля и мониторинга утилит выбираются самые необходимые и безопасные, которые можно оставить непосредственно на диске маршрутизатора. Остальной набор лучше держать на съемном носителе и монтировать при обслуживании системы. Особенно строго нужно подходить к отбору утилит для работы с дисками и файловыми системами, поскольку последствия их несанкционированного использования трудно переоценить.

4.1.5 Сетевые службы

Выбор служб на межсетевом экране/маршрутизаторе определяется решаемыми задачами и следует исходить из необходимого минимума. Например, вряд ли целесообразно на граничном узле активизировать такие службы, как NFS или Samba. Однако без поддержки SSH вы не сможете обеспечить безопасный удаленный доступ для управления хостом. Ниже перечислены службы, которые мне представляется разумным активизировать на вашем граничном шлюзе:

- **SSH** - безопасный протокол удаленного доступа
- **ntp** - протокол “сетевого” времени. Точная синхронизация часов может оказать существенную услугу при анализе инцидентов.
- **HTTP** - если в вашей системе используется Web-интерфейс для управления граничным шлюзом.
- **SMTP** - этот протокол имеет свои слабые места, но без почтового сервера будет сложно организовать передачу уведомлений и сигналов тревоги по электронной почте. Не забудьте тщательно настроить почтовый сервер, чтобы ваш граничный шлюз не превратился в средство рассылки спама. Может оказаться разумным блокирование внешних соединений с портом SMTP, поскольку граничный шлюз неразумно использовать в качестве публичного почтового транслятора.
- **SNMP** - этот протокол также не совсем безопасен, но он обеспечивает возможность эффективного удаленного мониторинга системы[1]. Доступ извне в порту SMTP целесообразно закрыть фильтрами.

4.1.6 Дополнительные программы мониторинга, управления и т. п.

В качестве дополнительных программ на компьютер, используемый в качестве межсетевого экрана и маршрутизатора целесообразно установить средства мониторинга хоста и сетевого трафика, системы детектирования вторжений и т. п. Подробное рассмотрение некоторых программ этого типа приводится в разделах 8 и 11.

4.1.7 Инструментальные средства

Установка на межсетевом экране средств разработки (компиляторов, отладчиков и т. п.) не такой простой вопрос, как может показаться на первый взгляд. С одной стороны, наличие этих средств упрощает процесс установки и обновления программ из исходных кодов, поскольку вы можете транслировать программу непосредственно в среде ее исполнения. Но этими же инструментами может воспользоваться проникший в систему злодей для трансляции и запуска вредоносных приложений.

Можно компилировать программы на другой машине, где имеется аналогичный набор библиотек и потом переносить на межсетевой экран бинарный код. Это несколько осложняет работу по обновлению и установке программ, а также требует более высокой квалификации и аккуратности.

Другим вариантом решения задачи является установка средств разработки на компакт-диск или иной сменный носитель, который монтируется в системе на время работы, а по завершении удаляется.

4.2 Пользовательские приложения

Спаси вас бог от установки на граничном шлюзе каких-либо пользовательских приложений - офисных систем, игрушек, multimedia и т. п. Лучшей помощи от вас злоумышленники просто не ждут.

4.3 Подготовка к созданию ядра

Ядро является основой операционной системы и поддерживаемый им набор функций имеет очень важное значение для функционирования вашей системы. В дистрибутивах общего назначения ядра транслируются для наиболее

1 *Например, с помощью программы mrtg (см. параграф 11.10.4).*

массовых (типовых) конфигураций, поэтому при создании межсетевого экрана в первую очередь следует подготовить свой вариант ядра, который будет обеспечивать эффективную поддержку всех требуемых брандмауэру функций и не будет включать в себя ничего лишнего или потенциально опасного.

4.3.1 Загрузка и подготовка исходных кодов

Этот параграф предназначен для тех, кто впервые взялся за создание ядра для своего компьютера. Если у вас уже имеется подобный опыт, можно смело переходить к параграфу 4.4.

Исходные тексты ядра Linux доступны на сайте http://www.kernel.org и многочисленных зеркалах, список которых опубликован на сайте. Выберите последнюю стабильную версию ядра (обычно она указывается в списке первой) и загрузите ее на свой компьютер (например, в каталог /usr/src/). Распакуйте исходные тексты ядра с помощью подходящей программы[1].

Перейдите в каталог linux-XXX[2] и внимательно ознакомьтесь с файлом README. Полезно также прочесть документы, помещенные в каталоге Documentation/. Объем включенной в архив ядра документации достаточно велик и вы найдете там много полезного, но читать документацию можно очень долго, а мы должны создать оптимально работающее ядро для нашей системы.

4.3.1.1 Установка расширений Netfilter

При создании межсетевого экрана или маршрутизатора Вы с высокой вероятностью будете использовать на нем систему фильтрации пакетов и контроля соединений **Netfilter/iptables** (см. параграф 5.1). Пакет Netfilter содержит ряд дополнений к ядру, которые целесообразно установить до компиляции ядра, чтобы потом не пришлось повторять эту процедуру[3]. Рассмотрим вкратце процесс установки исходных текстов Netfilter, а компиляцию и установку самого пакета рассмотрим позднее (параграф 5.1.11).

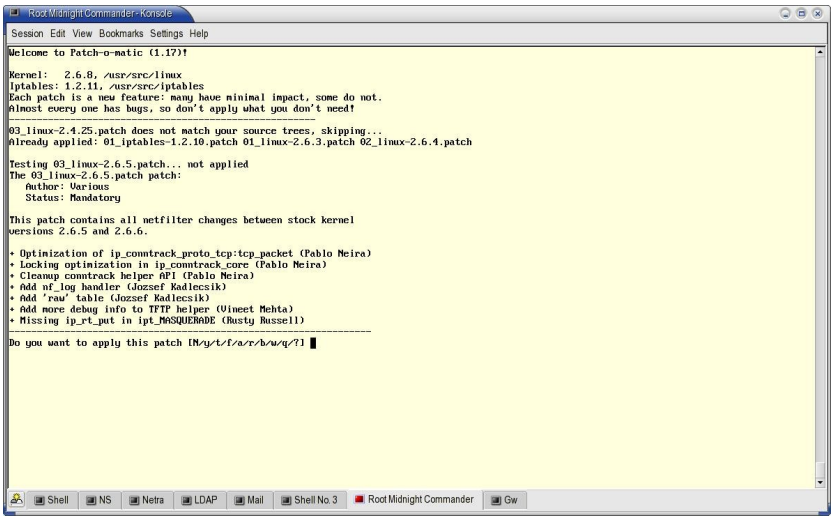


Рисунок 4.2. Сценарий установки patch-o-matic.

1) Загрузите с сайта www.netfilter.org исходные тексты программы и пакет расширений **patch-o-matic**.
2) Распакуйте полученные архивы в каталог **/usr/src** или иной каталог, который используется в вашей системе для хранения исходных текстов программ.
3) Перейдите в каталог, содержащий исходные тексты модулей расширения patch-o-matic и выполните операции по установке исходных текстов модулей расширения в дерево ядра Linux.
   a) Введите команду

      KERNEL_DIR=<каталог ядра> IPTABLES_DIR=<каталог iptables> ./runme pending

      для установки стандартных компонент Netfilter, которые еще не включены в ядро Linux. Укажите в команде реальные имена каталогов, содержащих исходные тексты ядра и iptables.
   b) Сценарий установки исходных текстов будет задавать вопросы о необходимости инсталляции того или модуля, сопровождаемые краткими комментариями о назначении каждого модуля и состоянии его разработки (см. рисунок). Если вы считаете нужным установить тот или иной модуль, нажмите клавишу **Y** в ответ на запрос.
   c) Если вы хотите установить также дополнительные модули из базового комплекта iptables, введите команду

      KERNEL_DIR=<каталог ядра> IPTABLES_DIR=<каталог iptables> ./runme base
   d) Для установки расширений служит команда

      KERNEL_DIR=<каталог ядра> IPTABLES_DIR=<каталог iptables> ./runme extra

4.3.2 Команды настройки опций

Для настройки конфигурации ядра используется команда make с параметром, определяющим вариант настройки параметров. В настоящее время поддерживается 4 варианта настройки конфигурационных параметров:

1 *В зависимости от типа архива это может быть программа gzip (файл .gz или .tgz) или bzip2 (.bz2). Раскройте полученный из архива файл linux-XXX.tar с помощью программы tar.*
2 *В вашем случае вместо XXX будет указан номер версии ядра.*
3 *Расширения, которые могут быть реализованы в форме загружаемых модулей ядра можно установить и после инсталляции ядра без его повторной компиляции. Достаточно будет только скомпилировать и установить добавленные модули ядра.*

1) Команда **make config** (рисунок 4.3) обеспечивает простой текстовый интерфейс для выбора значений опций и параметров конфигурации ядра. Основным недостатком этой программы является “прямолинейный” характер процесса настройки. Программа последовательно выводит на экран опции с возможными значениями и вы должны выбрать одно из возможных значений, после чего программа автоматически перейдет к следующей опции. Если вы допустите ошибку, выбрав значение, которое потом сочтете неверным, это значение уже нельзя будет исправить и придется начать процесс заново. Кроме того, в силу ограниченной высоты экрана ранее выбранные опции в конце концов уйдут за пределы экрана и вы не сможете посмотреть их.

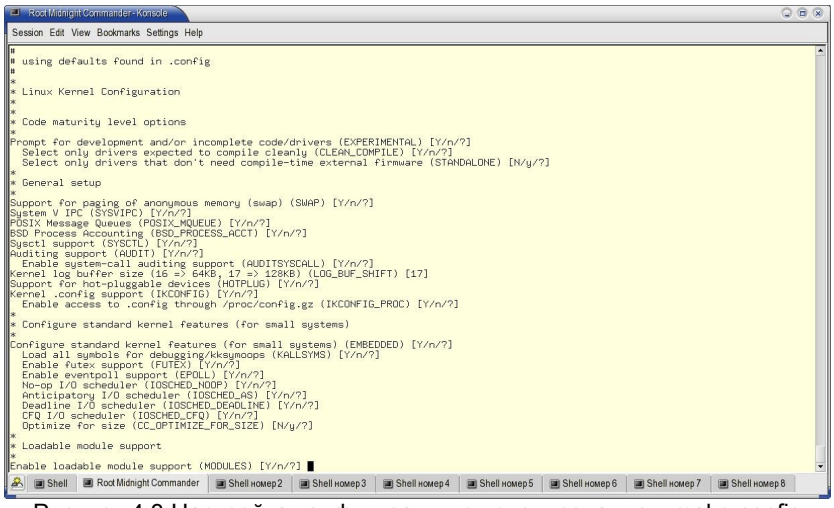


Рисунок 4.3 Настройка конфигурации с использованием make config

Если при выборе опции вам потребуется доступ к справочной информации, нажмите клавишу **?**.

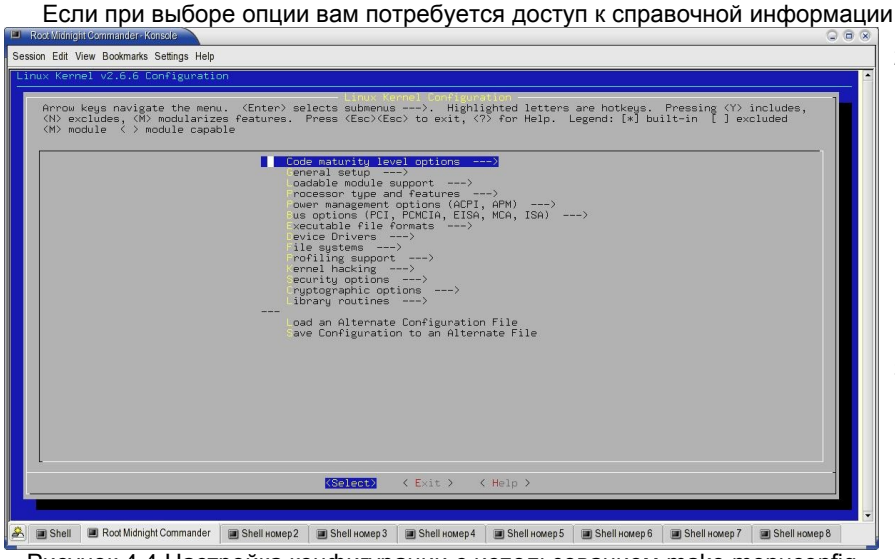


Рисунок 4.4 Настройка конфигурации с использованием make menuconfig

2) Команда **make menuconfig** (рисунок 4.4) также работает в текстовом режиме, но опции собраны в группы, доступ к которым обеспечивается через меню. Основным удобством является возможность возврата к ранее выбранным опциям для их просмотра или изменения. Для доступа к справочной информации используйте кнопку **Help**.

3) Команда **make xconfig** (рисунок 4.5) предназначена для использования в графической среде и является, пожалуй, самым удобным и эффективным инструментом настройки конфигурации ядра.

4) Если вы уже имеете оптимальную конфигурацию для будущего ядра, можно воспользоваться командой **make oldconfig**. В этом случая вам будет предложен “прямолинейный” текстовый интерфейс выбора значений только для тех опций, которые отсутствуют в имеющемся у вас конфигурационном файле (файл .config с корневом каталоге дистрибутива ядра). Такой вариант настройки конфигурации очень удобен для добавления создания новых загружаемых модулей ядра (например, при установке Netfilter после компиляции ядра).

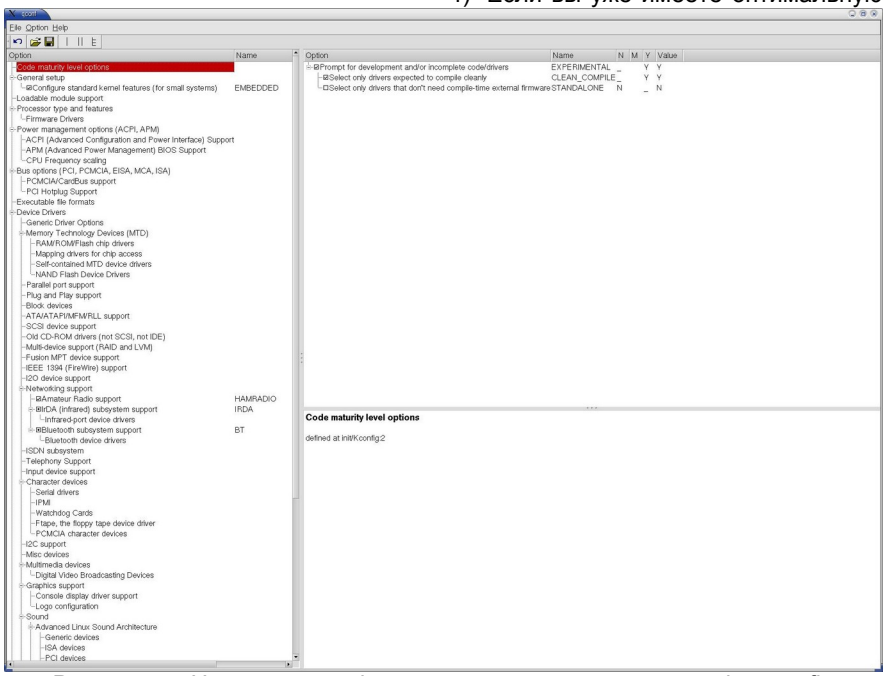


Рисунок 4.5 Настройка конфигурации с использованием make xconfig

Отметим в заключении, что некоторые дистрибутивы Linux включают средства настройки параметров ядра в число предоставляемых системному администратору инструментов.

4.4 Выбор опций ядра с учетом требований безопасности

Ядро Linux включает множество опций и их рассмотрение выходит за рамки обсуждаемых в книге тем, поэтому ниже рассматриваются лишь те опции, которые оказывают достаточно прямое и существенное влияние на работу Linux-системы в качестве межсетевого экрана или общий уровень безопасности системы.

4.4.1 Опции общего назначения

Рассматриваемые здесь опции могут не оказывать прямого влияния на безопасность хоста, но имеют важное значение для реализации тех или иных функций, имеющих отношение к безопасности данного хоста или “спрятанной” за ним локальной сети. Приведенное ниже описание опций компиляции относится к ядру Linux версии 2.6.8. В более старых ядрах некоторые опции могут отсутствовать, а в новых могут появиться дополнительные опции.

4.4.1.1 Меню Code maturity level options

Опции этого меню определяют возможностью использования в ядре экспериментальных модулей, а также драйверов для устройств, которые могут вызывать проблемы или требовать при компиляции загрузки программного кода для соответствующего оборудования (firmware).

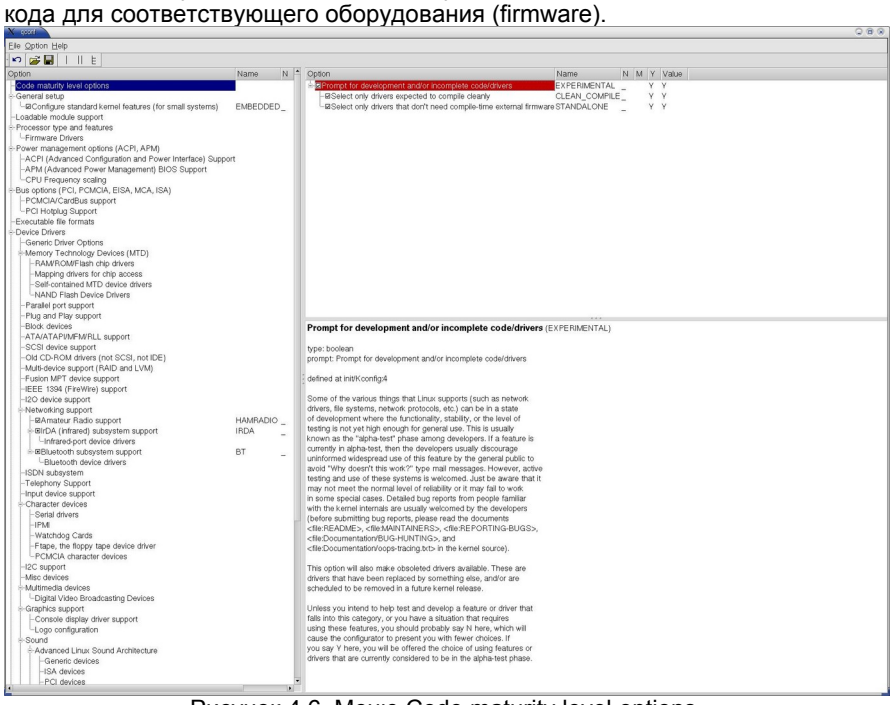


Рисунок 4.6. Меню Code maturity level options.

4.4.1.1.1 EXPERIMENTAL

Опция **Prompt for development and/or incomplete code/drivers** определяет возможность включения в ядро функций и опций, работа над которыми еще не закончена. В общем случае на шлюзах не следует использовать экспериментальные функции ядра, но достаточно часто эти функции обеспечивают новые возможности именно в сфере безопасности. Как правило, такие функции могут быть реализованы в виде модулей, поэтому даже при включении соответствующих опций уровень безопасности системы не снизиться. Для того, чтобы проверить возможности экспериментальных функций вам достаточно будет загрузить соответствующие модули с помощью команды

```
modprobe <имя модуля>
```

а после завершения экспериментов удалить модуль, используя команду

```
rmmod <имя модуля>
```

Если вы решились включить в ядро экспериментальные функции, выберите для опции значение **Y**.

4.4.1.1.2 CLEAN_COMPILE

Для опции **Select only drivers expected to compile cleanly** целесообразно выбрать значение **Y**, поскольку в этом случае в ядро не будут включаться модули и функции, для которых заведомо известно о возможности возникновения тех или иных проблем

4.4.1.1.3 STANDALONE

Если вы не используете на своем компьютере экзотических устройств, драйверы которых требуют при компиляции ядра загрузки кода firmware, выберите для опции **Select only drivers that don't need compile-time external firmware** значение **Y**.

4.4.1.2 Меню General setup

4.4.1.2.1 SWAP

Опция **Support for paging of anonymous memory (swap)** управляет поддержкой так называемой системы “дисковой подкачки памяти” (swap), обеспечивающей возможность сброса части содержимого оперативной памяти компьютера на диск. Использование подкачки позволяет существенно увеличить эффективный размер компьютерного ОЗУ.

В большинстве случаев для этой опции следует выбрать значение **Y**.

4.4.1.2.2 SYSVIPC

Опция **System V IPC** управляет поддержкой функций обмена информацией между процессами[1], позволяющих процессам обмениваться информацией между собой и синхронизировать данные. Эти функции весьма полезны, а некоторые программы просто не будут работать при выключенной поддержке IPC.

1 *IPC или Inter Process Communication.*

В большинстве случаев для этой опции следует выбирать значение **Y**.

Информацию о функциях IPC вы можете получить с помощью команды **info ipc**[1].

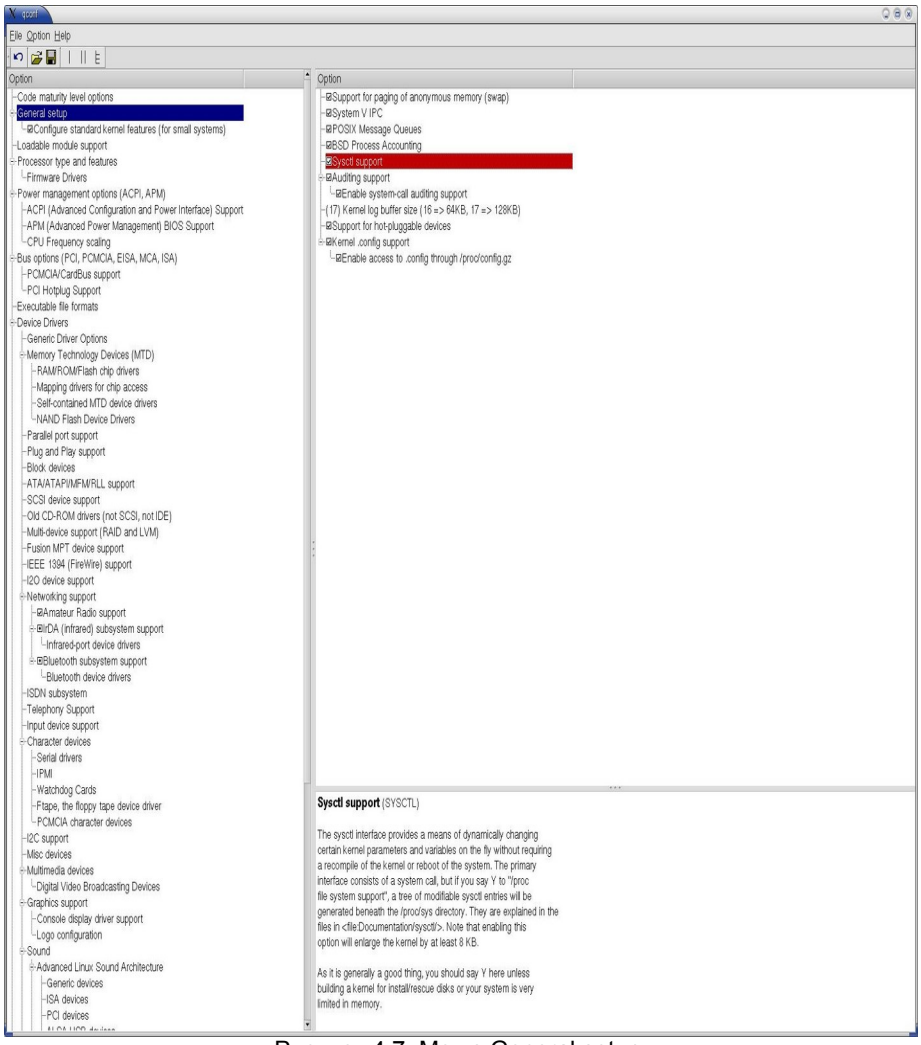


Рисунок 4.7. Меню General setup.

4.4.1.2.3 POSIX_MQUEUE

Опция **POSIX Message Queues** управляет поддержкой очередей сообщений POSIX (часть IPC). В очередях POSIX каждое сообщение имеет уровень приоритета, определяющий получение процессами этого сообщения. Опцию следует включить, если вы планируете компилировать и использовать программы, работающие с очередями POSIX (например, программы, разработанные для ОС Solaris). Для работы с очередями POSIX вам потребуется также библиотека **mqueue**, доступная на сайте http://www.mat.uni.torun.pl/~wrona/posix_ipc/.

Очередь сообщений POSIX доступна через файловую систему **mqueue**, которую вы можете смонтировать в удобном для работы месте.

4.4.1.2.4 BSD_PROCESS_ACCT

Опция **BSD Process Accounting** позволяет пользовательским программам давать ядру инструкции по записи в журнальный файл учетной информации о работе процессов. Всякий раз по завершении процесса ядро записывает в журнальный файл сведения о времени создания процесса, его владельце, команде запуска процесса, и использовании памяти, терминале и т. п. Полное описание структуры данных, используемой для хранения и записи учетной информации вы сможете найти в файле **/usr/include/linux/acct.h**.

В большинстве случаев для этой опции целесообразно выбрать значение **Y**.

4.4.1.2.4.1 BSD_PROCESS_ACCT_V3

Опция **BSD Process Accounting version 3 file format** включает использование нового формата записи информации о процессах. Этот формат поддерживает запись в журнальный файл сведений о каждом процессе и его родителях. Отметим, что новая версия формата несовместима с форматами версий 0 - 2, поэтому для работы с учетной информацией потребуется новый набор утилит, предварительную версию которого можно загрузить с сайта http://http://www.de.kernel.org/pub/linux/utils/acct/.

4.4.1.2.5 SYSCTL

Опция **Sysctl support** обеспечивает ядру интерфейс для обмена параметрами и переменными, позволяющими изменять режим работы без перезагрузки компьютера. Если вы активизируете эту опцию вместе с описанной ниже (стр. 67) опцией **/proc file system support**, будет создана виртуальная файловая система **/proc**, содержащая множество текстовых файлов, допускающих непосредственное редактирование и определяющих многие параметры системы. В частности, каталог **/proc/sys** будет содержать файлы общесистемного значения. Дополнительную информацию о файлах системных параметров можно найти в документах каталога Documentation/sysctl/ в дистрибутиве ядра Linux. Описание файлов **/proc** приводится в Приложении 12.2.1.

Отметим, что активизация этой опции приводит к увеличению размера ядра примерно на 8 Кб.

1 *Описание функций IPC приводится также в параграфе 6.4 документа Linux Programmer's Guide, который вы можете загрузить с сайта http://www.tldp.org/guides.html.*

4.4.1.2.6 AUDIT

Опция **Auditing support** позволяет использовать инфраструктуру системного аудита с другими подсистемами ядра (например, SELinux, см параграф 4.4.3.1.4 на стр. 97). Не вызывайте функций системного аудита, если при компиляции ядра не была включена опция **AUDITSYSCALL** (см. ниже).

4.4.1.2.6.1 AUDITSYSCALL

Опция **Enable system-call auditing support** обеспечивает поддержку инфраструктуры функций системного аудита, которая может использоваться независимо или вкупе с другими подсистемами ядра типа SELinux (см. параграф 4.4.3.1.4 на стр. 97).

4.4.1.2.7 LOG_BUF_SHIFT

Параметр **Kernel log buffer size (16 => 64KB, 17 => 128KB)** задает размер буфера сообщений ядра, используемого для записи в журнальные файлы системы с помощью функций syslog (параграф 2.8.4 на стр. 49). Значение параметра указывает двоичный логарифм размера буфера ядра в байтах. Рекомендуемые значения приведены ниже:

17 => 128 KB для систем S/390;
16 => 64 KB для систем x86 NUMAQ или IA-64;
15 => 32 KB для многопроцессорных систем SMP;
14 => 16 KB для однопроцессорных систем;
13 => 8 KB;
12 => 4 KB;

4.4.1.2.8 HOTPLUG

Опция **Support for hot-pluggable devices** включает поддержку устройств, подключаемых и отключаемых во время работы системы (PCMCIA, USB и т. п.).

Если вы выберете для этой опции значение **Y**, включите также опцию **KMOD** (параграф 4.4.1.3.4 на стр. 67), управляющую загрузкой модулей ядра и установите программный агент[1], ядро системы будет автоматически вызывать агент политики пользовательского режима (**/sbin/hotplug**) для загрузки модулей и установки программ, требуемых для работы подключенного устройства.

Если вы включите поддержку **HOTPLUG** на своем шлюзе, не забывайте, что имеющий физический доступ к этому шлюзу пользователь сможет подключить к шлюзу свое устройство и воспользоваться им без ведома администратора. В журнальных файлах останутся следы такого подключения, но может оказаться, что “поздно пить “Боржоми”.

4.4.1.2.9 IKCONFIG

Опция **Kernel .config support** позволяет сохранить в ядре всю информацию из конфигурационного файла ядра Linux **.config**. Этот файл содержит сведения об опциях компиляции ядра, которые можно прочесть с помощью сценария **scripts/extract-ikconfig** и использовать при компиляции нового ядра или внесения изменений в используемое. Параметры используемого при работе системы ядра можно также прочесть из файла **/proc/config.gz**, если включена опция **IKCONFIG_PROC**.

4.4.1.2.9.1 IKCONFIG_PROC

Опция **Enable access to .config through /proc/config.gz** позволяет просматривать параметры работающего ядра в файле **/proc/config.gz**.

4.4.1.2.10 Меню Configure standard kernel features (for small systems)

Опция **EMBEDDED** позволяет отключить некоторые базовые компоненты ядра. Такая возможность полезна при создании ядер для работы на специализированных платформах с ограниченной функциональностью. При включенной опции становится доступным одноименное меню (рисунок 4.8), опции которого следует использовать с осторожностью.

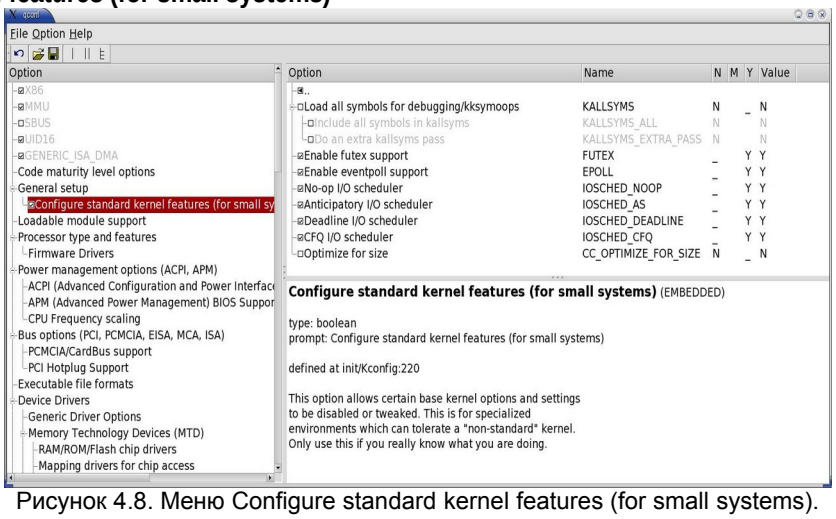


Рисунок 4.8. Меню Configure standard kernel features (for small systems).

4.4.1.2.10.1 KALLSYMS

Опция **Load all symbols for debugging/kksymoops** управляет включением в ядро отладочной информации, которая может быть полезна при возникновении в системе неисправимых ошибок. Выбор значения **Y**

1 *Исходные тексты агента вы можете загрузить с сайта http://linux-hotplug.sourceforge.net/.*

делает отладочную информацию доступной, он увеличивает размеры ядра, поскольку оно будет включать всю таблицу экспортируемых имен.

4.4.1.2.10.2 FUTEX

Опция **Enable futex support** управляет поддержкой "fast userspace mutexes". При отключенной опции некоторые приложения на базе **glibc** могут работать некорректно.

4.4.1.2.10.3 EPOLL

Опция **Enable eventpoll support** управляет поддержкой ядром системных вызовов **epoll**.

4.4.1.2.10.4 IOSCHED_NOOP

Опция **No-op I/O scheduler** управляет работой планировщика ввода-вывода[1], выполняющего основные операции объединения и сортировки. Основной работой планировщика является управление недисковыми блочными устройствами (например, модулями памяти) и обеспечения сервиса для программных и аппаратных подсистем, которым требуется лишь минимальное участие ядра в работе.

4.4.1.2.10.5 IOSCHED_AS

Опция **Anticipatory I/O scheduler** позволяет отключить предварительный планировщик ввода вывода[2], который используется по умолчанию для дисковых операций. Обычно этого планировщика достаточно для большинства систем, но он громоздок и сложен по сравнению с планировщиком **deadline I/O scheduler**. Кроме того, предварительный планировщик в некоторых случаях работает медленней например, с некоторыми базами данных).

4.4.1.2.10.6 IOSCHED_DEADLINE

Опция **Deadline I/O scheduler** управляет работой **deadline I/O scheduler** - простого и компактного планировщика, который для большинства задач столь же хорош, как **anticipatory I/O scheduler**, а с некоторыми базами данных работает даже быстрее. Когда в каждый момент дисковые операции выполняет единственный процесс, этот планировщик почти идентичен по возможностям планировщику **anticipatory I/O scheduler**.

4.4.1.2.10.7 IOSCHED_CFQ

Опция **CFQ I/O scheduler** управляет поддержкой планировщика ввода вывода CFQ, который пытается распределять полосу равномерно между всеми процессами системы. Возможностей этого планировщика достаточно для большинства настольных систем.

4.4.1.2.10.8 CC_OPTIMIZE_FOR_SIZE

Опция **Optimize for size** задает при компиляции ядра использование **gcc** с флагом **-Os** взамен **-O2**. Это ведет к снижению размеров ядра, но некоторые версии компилятора **gcc** могут при использовании этой опции порождать некорректный код.

4.4.1.3 Меню Loadable module support

Опции этого меню управляют возможностью построения модульного ядра и (при включенной поддержке модулей) загрузкой модулей ядра Linux в ручном и автоматическом режиме.

4.4.1.3.1 MODULES

Опция **Enable loadable module support** управляет возможностью реализации некоторых функций ядра в виде загружаемых модулей. Модули ядра представляют собой небольшие фрагменты исполняемого кода, которые могут “вставляться” в работающее ядро. Для загрузки модулей служит утилита **modprobe** (см. параграф 12.17.1).

Если вы выберете для опции значение **Y** многие компоненты ядра станут доступными в форме загружаемых модулей (выбор для опций значений **YMN** взамен обычного выбора **YN**). Реализация в виде модулей полезна для тех функций, которые не используются постоянно и не требуются при загрузке операционной системы. Дополнительную информацию о работе с модулями ядра вы найдете в Приложении 12.17).

При выборе для этой опции значения **Y** вы должны будете после компиляции ядра выполнить команды

```
make modules
make modules_install
```

для компиляции модулей и их установки в каталог /lib/modules/.

4.4.1.3.2 MODULE_UNLOAD

Опция **Module unloading** позволяет удалять из памяти неиспользуемые модули ядра, которые были загружены ранее. Отметим, что некоторые модули невозможно удалить из памяти ни при каких обстоятельствах.

1 *No-op I/O scheduler.*
2 *Anticipatory I/O scheduler.*

4.4.1.3.2.1 MODULE_FORCE_UNLOAD

Опция **Forced module unloading** позволяет удалять модули даже в тех случаях, когда ядро считает такую операцию небезопасной. При активизации этой опции ядро будет удалять модули без ожидания пока все прочие модули прекратят использовать удаляемый модуль. Эта опция полезна в основном для разработчиков и включать ее на маршрутизаторах и межсетевых экранах неразумно.

4.4.1.3.3 MODVERSIONS

Экспериментальная опция **Module versioning support** управляет возможностью использования модулей, скомпилированных для другого ядра. Такое использование модулей сопряжено с определенным риском возникновения нестабильности в системе, поэтому применять его следует с осторожностью.

4.4.1.3.4 KMOD

Опция **Automatic kernel module loading** позволяет ядру загружать модули автоматически, по мере возникновения потребности в них. Обычно модули ядра загружаются с помощью программы **modprobe**, но при выборе для этой опции значения **Y** вы позволите ядру самостоятельно вызывать команду **modprobe** с требуемыми аргументами, если какой-либо из компонент ядра потребовался этот модуль.

4.4.1.4 Меню Power management options (ACPI, APM)

Опции этого меню не оказывают прямого влияния на обеспечение безопасности, но влияют на работу хоста в целом. Отметим, что межсетевые экраны и маршрутизаторы предназначены для безостановочной работы, поэтому на них вряд ли целесообразно использовать функции энергосбережения.

4.4.1.4.1 PM

Опция **Power Management support** управляет поддержкой функций управления энергопотреблением APM и ACPI. Функции управления питанием важны для переносных компьютеров с батарейным питанием, а на ПК, используемом в качестве маршрутизатора, межсетевого экрана или сервера Internet эти функции вряд ли целесообразно включать.

Отметим, что на компьютерах x86 даже при отключенной опции управления питанием Linux будет использовать в периоды безделья процессора команду hlt, усыпляющую процессор и снижающую потребляемую им мощность.

4.4.1.5 Меню File systems

4.4.1.5.1 PROC_FS

Выбор опции **/proc file system support** приводит к созданию на компьютере в процессе загрузки ОС виртуальной файловой системы **/proc**, содержащей множество файлов с параметрами различных узлов (оборудования) и процессов, используемых на данном компьютере.

Файловая система **/proc** создается и монтируется в процессе загрузки ядра (строка типа **mount -t proc proc /proc** в файле **/etc/fstab**). Файлы дерева /proc можно просматривать и редактировать, но при редактировании следует соблюдать осторожность и не делать того, что не понятно. Информацию о файловой системе /proc вы сможете найти в документе **Documentation/filesystems/proc.txt** дистрибутива ядра Linux или получить с помощью команды **man 5 proc**. Краткое описание файлов **/proc/*** приведено в Приложении 12.2.1.

Использование этой опции увеличивает размер ядра приблизительно на 67 Кб, но нет

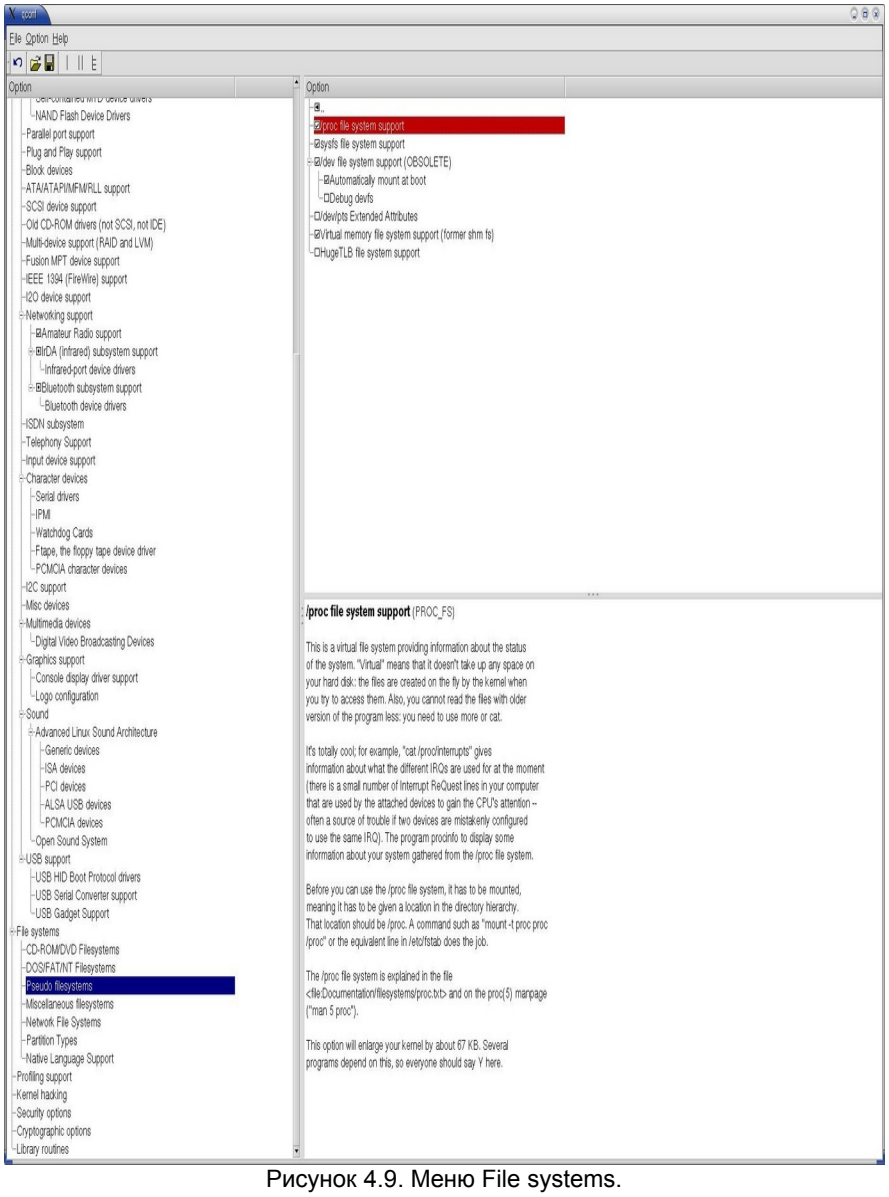


Рисунок 4.9. Меню File systems.

никаких причин отказываться от возможностей, предоставляемых файловой системой **/proc**[1].

4.4.1.5.2 NFS_FS

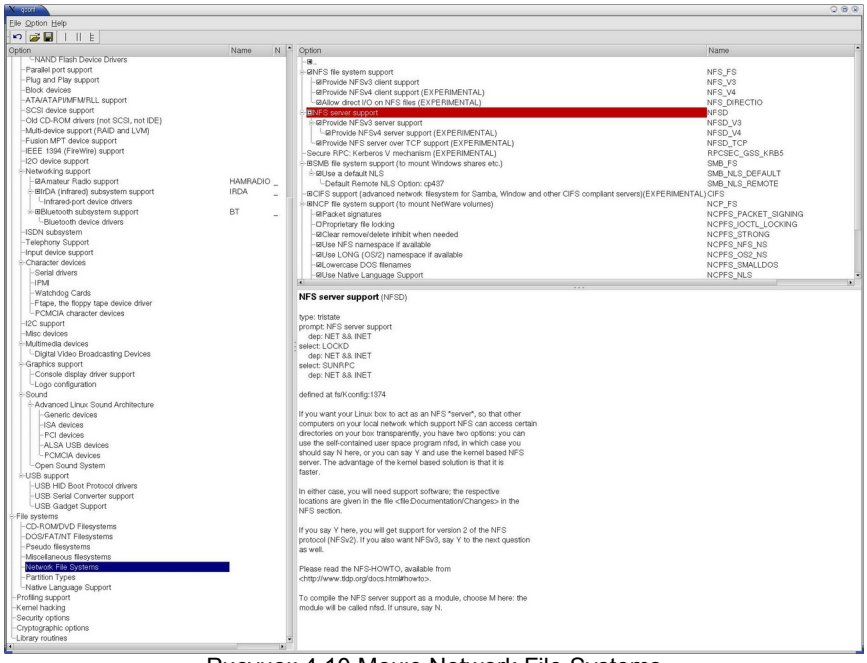


Рисунок 4.10 Меню Network File Systems

Опция **NFS server support** (**YNM**) позволяет использовать Linux-систему как сервер NFS, обеспечивающий другим компьютерам доступ к своим дисковым ресурсам через сеть. Протокол NFS не обеспечивает достаточного уровня безопасности, поэтому на межсетевых экранах и граничных маршрутизаторах лучше всего выбрать для этой опции значение **N**[2].

При выборе для опции значения M функции сервера NFS будут реализованы в модуле **nfsd**.

4.4.2 Сетевые опции ядра (Networking support)

Ядро Linux содержит множество опций, связанных с работой сети. Здесь мы лишь кратко рассмотрим некоторые из этих опций, наиболее важные с точки зрения безопасности и обеспечения эффективной работы хоста Linux в качестве межсетевого экрана с поддержкой функций маршрутизации.

4.4.2.1 NET

Выбор опции **Networking support** обеспечивает поддержку ядром сетевых функций. Информацию о поддержке сетевых функций ядром Linux можно найти на сайте Linux Documentation Project (http://www.tldp.org/docs.html#howto). При обновлении ядра целесообразно обновить и связанные с работой сети программные средства, используемые на этом компьютере. Информацию об изменениях можно найти к файле Documentation/Changes, дистрибутива ядра Linux. Описанные ниже опции доступны только при выборе **Networking support = Y**.

4.4.2.2 Меню Networking options

4.4.2.2.1 PACKET

Опция **Packet socket** управляет поддержкой пакетных сокетов, который позволяют приложениям, напрямую взаимодействовать с сетевыми интерфейсами. К числу таких приложений относится, например, программа сбора и анализа пакетов tcpdump (параграф 11.9.2 на стр. 262). Опция может принимать 3 значения - **Y** (включена), **N** (отключена) и **M** (загружаемый модуль)[3]. В большинстве случаев разумно выбрать значение **Y**. При выборе значения **M** для использования пакетных сокетов потребуется загрузка модуля **af_packet**[4].

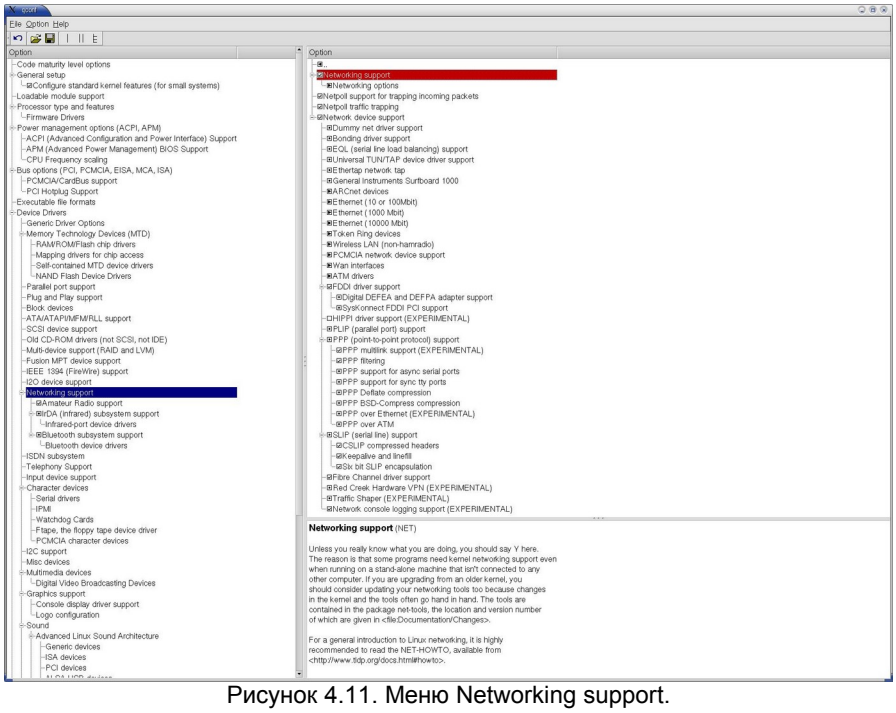


Рисунок 4.11. Меню Networking support.

4.4.2.2.1.1 PACKET_MMAP

При выборе опции **Packet socket: mmapped IO** драйвер пакетного протокола будет использовать механизмы ввода-вывода, ускоряющие работу. Если вы не

1 *Некоторые программы просто не будут работать без этой файловой системы.*
2 *Отметим, что выбор значения N отключает лишь поддержку функций сервера NFS непосредственно в ядре, но не запрещает реализовать эти функции за пределами ядра. Еще раз подчеркнем, что поддержка функций сервера NSF на межсетевом экране или граничном маршрутизаторе не представляется разумной с точки зрения безопасности (независимо от того, реализована эта поддержка в ядре или вне его).*

уверены в необходимости такого ускорения, выберите значение N.

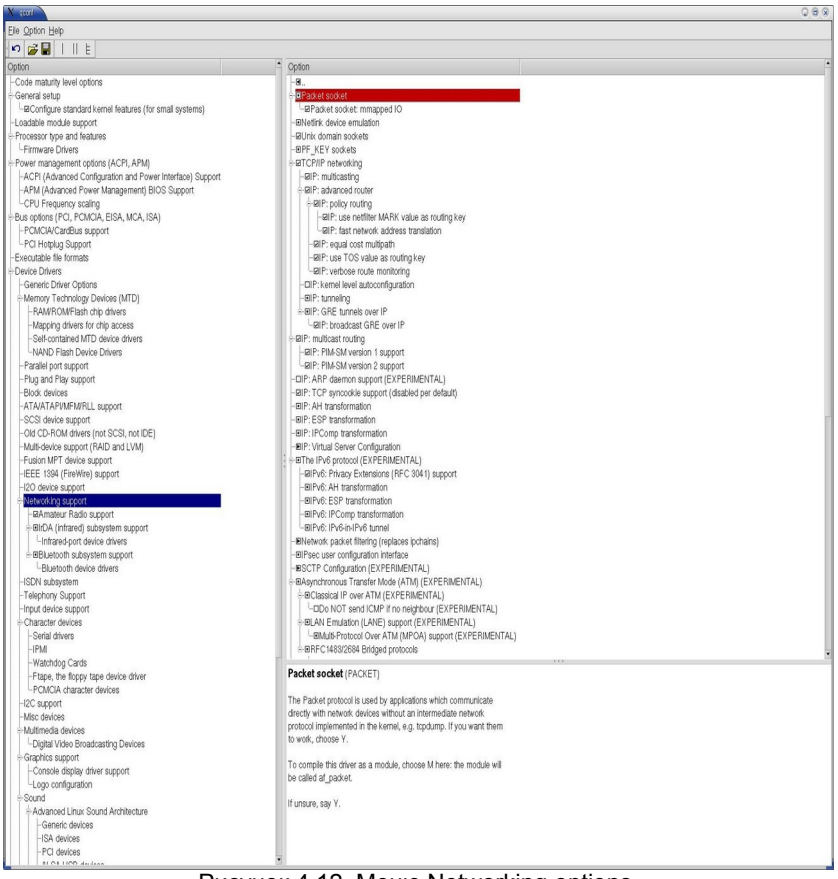


Рисунок 4.12. Меню Networking options.

4.4.2.2.2 NETLINK_DEV

Опция **Netlink device emulation**[1] (**YNM**) управляет эмуляцией устройства NETLINK (параграф 12.10), которая требуется всем приложениям, использующим устройства типа **/dev/tap0** или **/dev/route**.

При выборе значения **M** функции эмуляции устройства NETLINK будут реализованы в загружаемом модуле **netlink_dev**.

4.4.2.2.3 UNIX

При выборе для опции **Unix domain sockets** значения Y будет включена поддержка сокета доменов Unix, обеспечивающая стандартный для Unix-систем механизм организации сетевых соединений и доступа к ним. Многие программы общего назначения (например, оконная система X Window и программа syslog) будут пользоваться этим сокетом даже в тех случаях, когда компьютер не подключен к сети, поэтому целесообразно включать эту опцию во всех случаях, когда отсутствуют явные противопоказания. При выборе значения **M** соответствующие функции будут реализованы в модуле, носящем название **unix**[2].

4.4.2.2.4 NET_KEY

Опция **PF_KEY sockets** (**YNM**) управляет поддержкой семейства сокетов PF_KEYv2, совместимого с KAME и обеспечивающего возможность использования протоколов IPsec, перенесенных от KAME. При отсутствии явных противопоказаний выберите значение Y.

4.4.2.2.5 INET (опции ядра для стека TCP/IP)

Опция **TCP/IP networking** определяет возможность использования на компьютере стека протоколов TCP/IP. Для межсетевого экрана эта опция должна иметь значение Y и даже на компьютерах, не подключаемых к сети имеет смысл использовать это значение (хотя это и приведет к увеличению размера ядра примерно на 144 Кб), поскольку некоторые программы (например, X window) используют стек TCP/IP даже для локальных операций, выполняя их через loopback-интерфейс.

При включенной поддержке TCP/IP и выборе значения Y для опций **/proc file system support** (стр. 67) и **Sysctl support** (стр. 64) можно управлять параметрами работы стека TCP/IP, записывая соответствующие значения в файлы виртуальной файловой системы **/proc** (каталоги **/proc/sys/net/ipv4/***). Описание этих файлов приведено в Приложении 12.3[3].

4.4.2.2.5.1 IP_MULTICAST (групповая адресация)

При выборе опции **IP: multicasting** обеспечивается поддержка маршрутизации для групповых адресов. Размер ядра увеличивается приблизительно на 2 Кб. Для большинства хостов поддержка групповой адресации не требуется. Дополнительную информацию можно найти в файле **Documentation/networking/multicast.txt** дистрибутива ядра Linux.

4.4.2.2.5.2 IP_ADVANCED_ROUTER (опции маршрутизации IP)

Опция **IP: advanced router** повышает эффективность пересылки пакетов при использовании компьютера в качестве маршрутизатора. Включение этой опции активизирует ряд дополнительных опций настройки ядра. Отметим, что выбор значения **IP: advanced router = N** не оказывает прямого влияния на ядро - просто становятся недоступными несколько описанных ниже опций маршрутизации.

3 *Далее будем просто указывать возможные значения **Y**, **N**, **M** без дополнительных пояснений.*
4 *Для проверки и загрузки модулей служит команда **modprobe** (параграф 12.17.1) или **insmod** (параграф 12.17.4). Список загруженных модулей можно посмотреть с помощью команды **lsmod** (параграф 12.17.3).*
1 *В будущих версиях ядра планируется удалить эту опцию.*
2 *Отметим, что при использовании модульного варианта некоторые функции операционной системы не будут корректно работать, если вы забудете загрузить модуль unix.*
3 *Описание для используемой вами версии ядра вы сможете найти в файле **Documentation/networking/ip-sysctl.txt** дистрибутива ядра вашей системы.*

Для того, чтобы компьютер можно было использовать для пересылки пакетов (маршрутизации) такая пересылка должна быть разрешена на уровне ядра. Включить маршрутизацию можно с помощью команды

```
echo "1" > /proc/sys/net/ipv4/ip_forward
```

после монтирования файловой системы **/proc** (например, в процессе загрузки компьютера), если при компиляции ядра были включены опции **/proc file system support** (параграф 4.4.1.5.1 на стр. 67) и **Sysctl support** (параграф 4.4.1.2.5 на стр. 64).

При включенной маршрутизации автоматически активизируется фильтр **rp_filter** (см. параграф 12.3.1.13.12 на стр. 374), который будет отвергать все входящие пакеты, если запись таблицы маршрутизации для адреса отправителя не будет соответствовать сетевому интерфейсу, через который пакет был принят. Такая фильтрация повышает уровень безопасности системы, предотвращая пересылку пакетов с обманными адресами (IP spoofing). Для отключения этого фильтра можно воспользоваться командой

```
echo 0 > /proc/sys/net/ipv4/conf/<device>/rp_filter
```

для одного интерфейса device или

```
echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter
```

для всех интерфейсов.

4.4.2.2.5.2.1 IP_MULTIPLE_TABLES

Обычно маршрутизатор пересылает пакеты, принимая решение на основе адреса получателя. Выбор опции **IP: policy routing** позволяет Linux принимать решение о пересылке пакета с учетом адреса отправителя. Более того, при включенной опции **Use TOS value as routing key** (параграф 4.4.2.2.5.2.3 на стр. 70) на маршрутизацию будет также влиять значение поля TOS (тип обслуживания) в заголовке пакета. Если же для описанной ниже опции **Fast network address translation** выбрано значение **Y**, маршрутизатор сможет изменять адреса отправителя и получателя в пересылаемых пакетах.

Дополнительные сведения по использованию этой опции вы сможете найти в документах http://www.compendium.com.ar/policy-routing.txt и ftp://post.tepkom.ru/pub/vol2/Linux/docs/advanced-routing.tex. Если вам нужны дополнительные программы для поддержки маршрутизации, загляните на сайт ftp://ftp.tux.org/pub/net/ip-routing/.

4.4.2.2.5.2.1.1 IP_ROUTE_FWMARK

Опция **IP: use netfilter MARK value as routing key** позволяет использовать различные маршруты для пакетов с разными маркерами. Для установки маркеров служит программа iptables (операция **MARK**, описанная в параграфе 5.1.8.2.9).

4.4.2.2.5.2.1.2 IP_ROUTE_NAT

Опция **IP: fast network address translation** позволяет изменять адреса отправителя и получателя в пакетах, проходящих через маршрутизатор (трансляция адресов - NAT). Дополнительную информацию по вопросам трансляции адресов вы сможете найти на сайте http://www.hasenstein.com/linux-ip-nat/diplom/nat.html. Изменение адресов осуществляется с помощью операций **BALANCE** (параграф 5.1.8.2.1 на стр. 110), **DNAT** (параграф 5.1.8.2.5 на стр. 111) и **SNAT** (параграф 5.1.8.2.17 на стр. 114).

4.4.2.2.5.2.2 IP_ROUTE_MULTIPATH

Выбор опции **IP: equal cost multipath** обеспечивает возможность поддержки в таблице нескольких равноценных маршрутов к одному адресату. Пакеты могут передаваться по всем таким маршрутам сразу.

4.4.2.2.5.2.3 IP_ROUTE_TOS

Опция **IP: use TOS value as routing key** управляет использованием битов поля TOS (тип обслуживания) в заголовке IP для принятия решений о пересылке пакетов. Поле типа обслуживания может запрашивать для пакетов малую задержку (незамедлительная обработка), высокую пропускную способность или гарантированную доставку (пакеты не следует отбрасывать). Описание поля TOS можно найти в RFC 791.

4.4.2.2.5.2.4 IP_ROUTE_VERBOSE

При выборе опции **IP: verbose route monitoring** ядро Linux будет поддерживать (и выводить по запросу) подробную информацию о маршрутах (например, при получении странных пакетов или атаке из сети). Эта информация обеспечивается демоном **klogd**[1].

4.4.2.2.5.3 IP_PNP (опции сетевой загрузки)

Опция **IP: kernel level autoconfiguration** позволяет автоматически задавать сетевую конфигурацию (адреса IP и т. п.) и таблицу маршрутизации при загрузке ядра на основе сведений, полученных из команды загрузки ядра или с помощью протоколов BOOTP или RARP. Обычно эта опция используется только для бездисковых станций, которым доступ в сеть требуется в процессе загрузки операционной системы[2]

1 *Информацию об этом демоне можно получить с помощью команды **man klogd**.*
2 *В таких случаях может оказаться полезной опция ядра **Root file system on NFS***

4.4.2.2.5.3.1 IP_PNP_DHCP

Выбор опции **IP: DHCP support** позволяет компьютеру монтировать корневую файловую систему на сетевом диске NFS и получать адрес IP в процессе загрузки ОС по протоколу DHCP. Если ваш сетевой адаптер поддерживает загрузку ОС из сети и получение адреса от сервера DHCP, вы можете выбрать для этой опции значение **N** даже для бездисковой станции. Дополнительную информацию о загрузке ОС из сети вы сможете найти в файле **Documentation/nfsroot.txt** дистрибутива ядра Linux.

4.4.2.2.5.3.2 IP_PNP_BOOTP

Опция **IP: BOOTP support** включает поддержку протокола удаленной загрузки BOOTP, позволяющего хосту получить при загрузке ОС параметры сетевой конфигурации с сервера BOOTP и смонтировать корневую файловую систему на сетевом диске NFS. Если ваш сетевой адаптер поддерживает загрузку ОС из сети и получение адреса от сервера BOOTP, вы можете выбрать для этой опции значение **N** даже для бездисковой станции. Дополнительную информацию о загрузке ОС из сети вы сможете найти в файле **Documentation/nfsroot.txt** дистрибутива ядра Linux.

4.4.2.2.5.3.3 IP_PNP_RARP

Опция **IP: RARP support** позволяет смонтировать корневую файловую систему на сетевом диске NFS и получить конфигурационные параметры из сети по протоколу RARP[1]. Дополнительную информацию о загрузке ОС из сети вы сможете найти в файле **Documentation/nfsroot.txt** дистрибутива ядра Linux.

4.4.2.2.5.4 NET_IPIP (туннелирование)

Опция **IP: tunneling** (**YNM**) управляет возможностью организации туннельных соединений, когда данные одного протокола инкапсулируются в другой протокол и передаются последним по каналу, поддерживающему протокол инкапсуляции. Данный драйвер обеспечивает туннелирование путем инкапсуляции IP в IP. Такая инкапсуляция может показаться странной и бессмысленной, но она будет весьма полезна в тех случаях, когда ваш компьютер относится не к той сети, с которой он соединен физически (например, при доступе в сеть с использованием мобильного подключения). Вы сможете работать с динамическим адресом IP, полученным из публичной сети, сохраняя для своей сети обычный IP-адрес, с которым вы подключаетесь к офисной сети.

4.4.2.2.5.5 NET_IPGRE (туннелирование)

Опция **IP: GRE tunnels over IP** (**YNM**) управляет возможностью туннелирования с использованием протокола GRE (Generic Routing Encapsulation), обеспечивающего инкапсуляцию трафика IPv4 и IPv6 для передачи через существующие сети IPv4. Драйвер весьма полезен для тех случаев, когда на другой стороне туннеля используется маршрутизатор Cisco (последние понимают протокол GRE значительно лучше, чем протокол туннелирования Linux - **IP tunneling**). Кроме того, протокол GRE поддерживает для туннелей возможность использования групповой адресации.

4.4.2.2.5.5.1 NET_IPGRE_BROADCAST

Одним из вариантов применения протокола туннелирования GRE/IP является создание широковещательных распределенных сетей (WAN), которые с точки зрения пользователя выглядят как обычные ЛВС Ethernet, но реально передают потоки данных через Internet. Опция **IP: broadcast GRE over IP** определяет возможность передачи широковещательных пакетов через туннели. Для использования такой возможности следует также включить описанную ниже в следующем параграфе опцию **IP multicast routing**.

4.4.2.2.6 IP_MROUTE (опции групповой маршрутизации)

Опция **IP: multicast routing** может быть полезна для маршрутизаторов, пересылающих пакеты с групповыми адресами (множество получателей). Использование этой опции потребуется при работе с MBONE[2]. Скорей всего для работы с маршрутизацией групповых пакетов вам потребуется также программа **mrouted** (демон маршрутизации для групповых адресов). Сведения о поддержке групповой адресации различными сетевыми адаптерами можно найти в файле **Documentation/networking/multicast.txt** дистрибутива ядра linux.

4.4.2.2.6.1 IP_PIMSM_V1

Опция **IP: PIM-SM version 1 support** управляет поддержкой ядром протокола Sparse Mode PIM (Protocol Independent Multicast - независимая от протокола групповая передача) версии 1. Этот протокол маршрутизации для групповых адресов используется достаточно широко, благодаря его поддержке оборудованием компании Cisco. Для использования протокола потребуется программа **pimd-v1**. Информацию об этом протоколе можно найти на сайте http://netweb.usc.edu/pim/.

4.4.2.2.6.2 IP_PIMSM_V2

Опция **IP: PIM-SM version 2 support** управляет поддержкой протокола групповой маршрутизации Sparse Mode PIM версии 2. Для использования протокола потребуется экспериментальный демон маршрутизации **pimd** или **gated-5**. Протокол версии 2 используется достаточно редко.

1 *Этот протокол достаточно устарел и вместо него обычно используются более современные протоколы DHCP и BOOTP.*
2 *Служба высокоскоростной передачи аудиовизуальной информации через Internet.*

4.4.2.2.7 ARPD (демон ARP)

Экспериментальная опция **IP: ARP daemon support** (**YNM**) позволяет сократить размер внутреннего кэша ARP, используемого ядром, до 256 записей за счет переноса части таблицы в пользовательское пространство. Обычно ядро сохраняет в кэше всю информацию о соответствии аппаратных и сетевых адресов, но в больших сетях для этого может потребоваться значительное количество памяти. Использование демона ARP позволяет перенести таблицы соответствия адресов из ядра в пользовательское пространство и держать во внутреннем кэше ядра только самые свежие записи. Для использования этой опции потребуется программа-демон **arpd**. Кроме того, должна быть включена опция **Netlink device emulation** (параграф 4.4.2.2.2 на стр. 69).

4.4.2.2.8 SYN_COOKIES

Опция **IP: TCP syncookie support** управляет возможностью использования отключенного по умолчанию режима защиты от атак SYN-flood[1]. При включенной опции стек TCP/IP будет использовать криптозащищенный протокол SYN cookies, обеспечивающий легитимным пользователям возможность работы даже во время атаки. Пользователям не потребуется менять свои программы стека TCP/IP, поскольку протокол SYN cookies обеспечивает полную прозрачность. Более подробные сведения об этом протоколе вы сможете найти на сайте http://cr.yp.to/syncookies.html[2].

Во время атаки SYN flood адреса отправителя, сообщаемые ядру, скорей всего, являются подставными, поэтому им не следует придавать важного значения.

Протокол SYN cookies может блокировать выдачу корректных сообщений об ошибках на клиентских машинах при перегрузке сервера в результате атаки. Если это происходит достаточно часто, разумно будет отключить использование этого протокола.

Для включения механизма SYN вы можете использовать команду[3]

```
echo 1 >/proc/sys/net/ipv4/tcp_syncookies
```

во время загрузки ОС после монтирования файловой системы /proc.

4.4.2.2.9 INET_AH

Опция **IP: AH transformation**" (**YNM**) управляет преобразованием заголовков IPsec AH (заголовки аутентификации). Протокол IP Authentification Header описан в RFC 2402). Для этой опции целесообразно выбрать значение **Y**.

4.4.2.2.10 INET_ESP

Опция **IP: ESP transformation** (**YNM**) управляет поддержкой протокола IPsec ESP (Encapsulation Security Payload), описанного а RFC 2406. Для этой опции целесообразно выбрать значение **Y**.

4.4.2.2.11 INET_IPCOMP

Опция **IP: IPComp transformation** управляет поддержкой протокола IP Payalod Compression (RFC 3173), которая обычно нужна для приложений IPsec. Для этой опции целесообразно выбрать значение **Y**.

4.4.2.2.12 IP_VS

Экспериментальная опция **IP virtual server support** (**YNM**) управляет поддержкой режима **IP Virtual Server**, позволяющего создавать высокопроизводительные кластеры на базе двух и более серверов. Эта опция должна быть включена по крайней мере на одном из входящих в кластер компьютеров, который будет в этом случае принимать входящие соединения по своему IP-адресу и распределять вызовы между реальными серверами кластера.

В настоящее время Linux поддерживает три технологии диспетчеризации запросов - NAT, туннелирование и прямая маршрутизация. Для

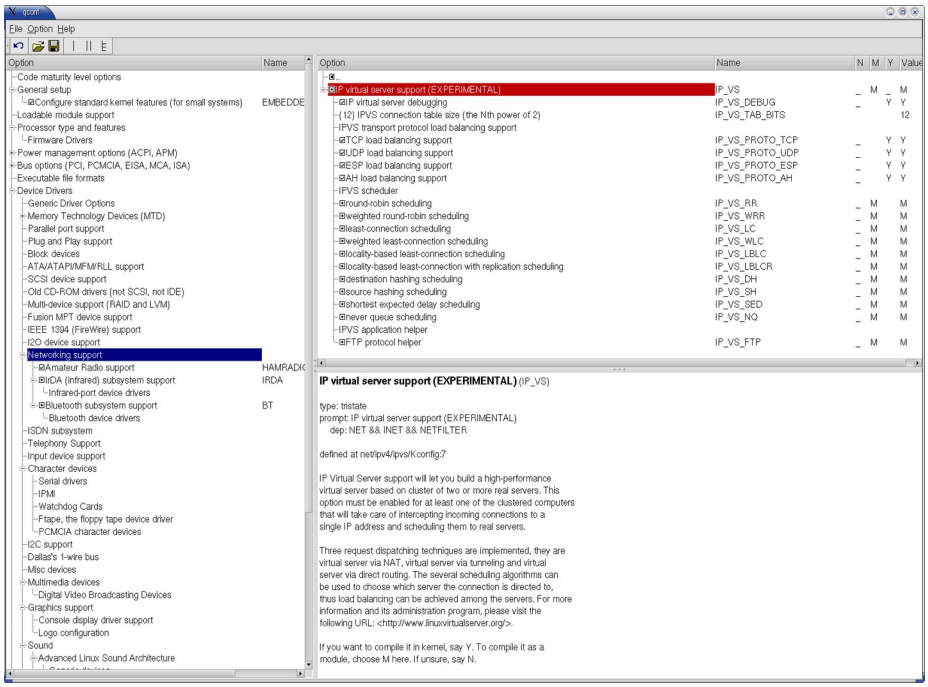


Рисунок 4.13. Меню IP virtual server support.

1 *Загрузка хоста большим количеством попыток оранизации соединений TCP.*
2 *Копия этого файла имеется на приложенном к книге компакт-диске (каталог Documents), а его перевод на русский язык приведен в Приложении 12.15.*
3 *Для того, чтобы приведенная команда работала, следует установить значение **Y** для опций ядра **/proc file system support** и **Sysctl support**.*

распределения запросов может использоваться несколько алгоритмов планирования, позволяющих распределять нагрузку между физическими серверами кластера. Дополнительную информацию о возможностях такой кластеризации вы найдете на сайте http://www.linuxvirtualserver.org/.

При выборе для опции значения **M** поддержка функций кластеризации будет реализована в виде загружаемого модуля **ip_vs**.

Если для опции было выбрано значение Y или M, вы получите доступ к опциям управления режимом поддержки виртуального сервера. Мы не будем здесь останавливаться на этом вопросе, приведя лишь список опций меню **IP virtual server support** (рисунок 4.13).

4.4.2.2.13 IPV6 (опции протокола IPv6)

Поддержка протокола IP версии 6 в ядре Linux пока является экспериментальной и сам протокол распространен недостаточно широко, поэтому мы не будем останавливаться на этом вопросе. Если вы не планируете экспериментировать с IPv6, отключите опцию **The IPv6 protocol**.

4.4.2.2.14 Меню Network packet filtering (replaces ipchains) - фильтрация пакетов

Опции этого меню играют важную роль в обеспечении безопасности данного хоста и защищаемой им сети, поэтому мы рассмотрим их достаточно подробно. Опции вы можете видеть на рисунке 4.14.

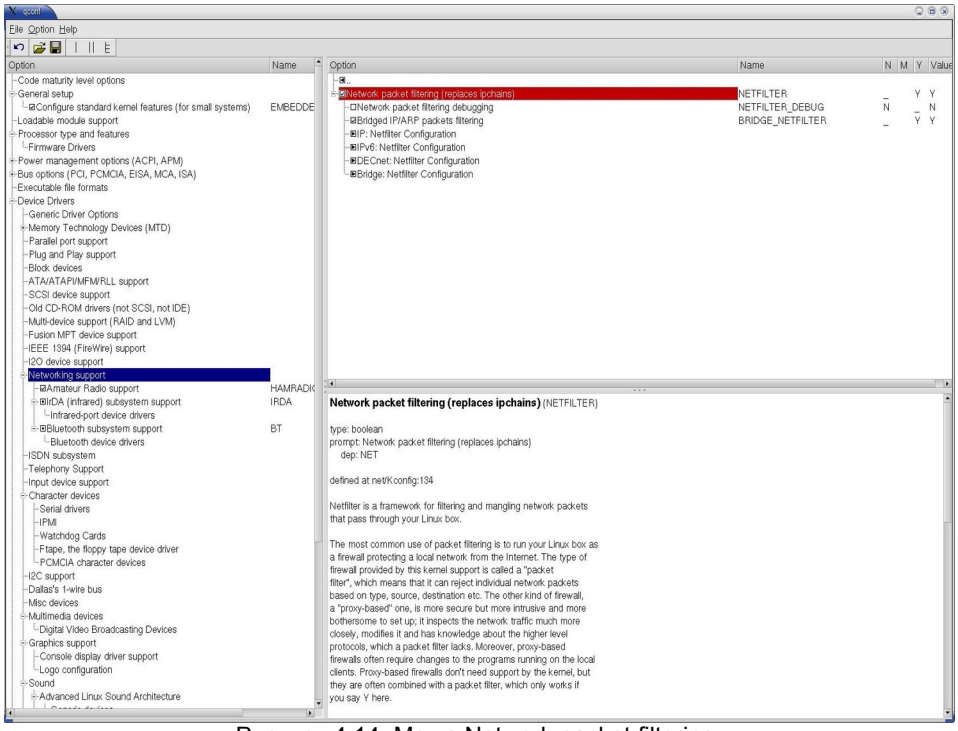


Рисунок 4.14. Меню Network packet filtering.

4.4.2.2.14.1 NETFILTER

Опция **Network packet filtering (replaces ipchains)** управляет работой программ iptables/netfilter, обеспечивающих фильтрацию и изменение пакетов, проходящих через хост Linux, принимаемых или генерируемых этим хостом. В большинстве случаев для этой опции следует выбрать значение **Y**.

Фильтрация пакетов используется чаще всего на Linux-машинах, служащих в качестве маршрутизаторов и межсетевых экранов, соединяющих локальные сети с Internet. Межсетевые экраны такого типа обычно называют пакетными фильтрами (packet filter). Это означает, что брандмауэр может отвергать отдельные пакеты на основе их типа, а также адресов получателя/отправителя и других параметров. Другой тип межсетевых экранов, работающих в режиме “посредников” (proxy-based), обеспечивает, как правило, более высокий уровень защиты, но они более уязвимы для несанкционированного доступа и требуют к себе большего внимания. Такие брандмауэры “глубже” заглядывают в поток сетевого трафика, изменяют этот поток и поддерживают информацию о протоколах вышележащих уровней, отсутствующую у пакетных фильтров. Однако, использование межсетевых экранов на базе proxy зачастую требует внесения изменений в программы, используемые на клиентских компьютерах. Proxy-брандмауэры не поддерживаются ядром Linux, но их можно использовать на одном компьютере с пакетными фильтрами.

Отметим, что современные реализации систем контроля трафика на базе Linux не являются простыми пакетными фильтрами, поскольку они обеспечивают контроль трафика с учетом состояния соединений - **stateful inspection**.

Если вы планируете использовать компьютер в качестве граничного шлюза Internet и в вашей сети не используется публичный блок адресов IP, маскирование (masquerading) сетевых адресов обеспечит доступ в Internet из локальной сети при наличии единственного публичного адреса. В выходящих из локальной сети пакетах в качестве адреса отправителя используется IP-адрес внешнего интерфейса Linux-машины, а при получении откликов из Internet ядро Linux аккуратно восстанавливает адрес получателя и пересылает пакет соответствующему компьютеру локальной сети. В результате преобразования адресов все компьютеры вашей ЛВС получают доступ в Internet как при работе с публичными адресами, а компьютеры вашей локальной сети совершенно не видны из Internet, поскольку не имеют публичных адресов IP. Маскирование адресов часто также называют трансляцией сетевых адресов NAT (Network Address Translation).

Другим вариантом использования Netfilter является создание прозрачных служб proxy. Если компьютер из вашей ЛВС пытается соединиться с компьютером внешней сети, Linux-машина будет обеспечивать прозрачную пересылку трафика локальному серверу, обеспечивающему поддержку функций кэширующего proxy.

Netfilter позволяет также создавать мосты с поддержкой функций фильтрации пакетов. Программа iptables "видит" проходящий через мост трафик и может фильтровать пакеты. Для фильтрации пакетов на канальном уровне используется программа **ebtables** (в конфигурации моста с поддержкой netfilter). Эту программу можно загрузить с

сайта проекта http://ebtables.sourceforge.net/[1]. Краткое описание программы приводится в параграфе 5.2.

Пакет Netfilter включает набор модулей, используемых взамен старых программ трансляции адресов (ipmasqadm), фильтрации пакетов (ipchains), поддержки прозрачных proxy и механизмов пересылки между портами. Дополнительную информацию по этим модулям можно найти в файле Documentation/Changes (раздел iptables) дистрибутива ядра Linux.

Если вы планируете использовать средства фильтрации пакетов, не забудьте отключить опцию быстрой пересылки пакетов (**Fast switching = N)**, поскольку при быстрой пересылке пакетные фильтры просто не будут использоваться.

4.4.2.2.14.1.1 NETFILTER_DEBUG

Опция **Network packet filtering debugging** управляет выводом дополнительной информации, которая может быть полезна при отладке программного кода netfilter.

4.4.2.2.14.1.2 BRIDGE_NETFILTER

Включенная по умолчанию опция **Bridged IP/ARP packets filtering** служит для управления фильтрацией проходящего через мост трафика ARP. Изменение этой опции не оказывает влияния на работу цепочек **ebtables** (стр. 143) и **arptables** (стр. 151).

4.4.2.2.14.2 Меню IP: Netfilter Configuration

Опции этого меню определяют работу пакетных фильтров для протокола IPv4 и играют важную роль в обеспечении безопасности данного хоста и расположенной за ним сети при использовании Linux-машины в качестве маршрутизатора/межсетевого экрана.

4.4.2.2.14.2.1 IP_NF_CONNTRACK

Опция **Connection tracking (required for masq/NAT)** (**YNM**) управляет контролем принадлежности пакетов к существующим сетевым соединениям.

Эта опция требуется для поддержки функций трансляции сетевых адресов (Masquerading и Network Address Translation, за исключением функции Fast NAT). Кроме того, эта опция нужна для фильтрации пакетов с учетом состояния соединений (см. описанную на стр. опцию **Connection state match support**).

Отметим, что при активизации функция контроля за соединениями автоматически включается дефрагментация пакетов.

Если для опции было выбрано значение **M** функции контроля соединений будут реализованы в загружаемом модуле **ip_conntrack**.

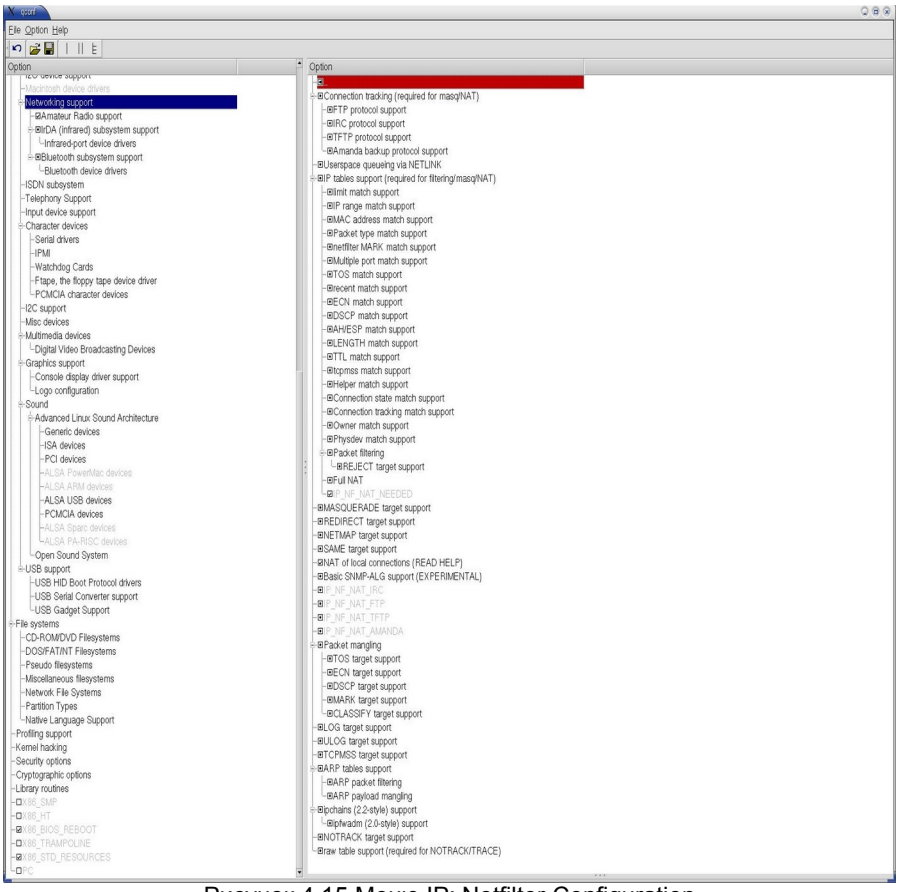


Рисунок 4.15 Меню IP: Netfilter Configuration

4.4.2.2.14.2.1.1 IP_NF_FTP

Опция **FTP protocol support** (**YNM**) служит для контроля соединений FTP.

Отслеживание FTP-соединений представляет собой достаточно сложную задачу, требующую применения специальных программ (helper), которые контролируют соединения и выполняют для них маскирование, трансляцию адресов и другие функции.

При выборе значения **M** функции контроля соединений FTP будут реализованы в модуле **ip_conntrack_ftp**.

4.4.2.2.14.2.1.2 IP_NF_IRC

Опция **IRC protocol support** (**YNM**) позволяет использовать протокол DCC одновременно с трансляцией адресов.

Существует достаточно широко используемое расширение IRC, называемое протоколом DCC[2]. Этот протокол дает пользователям возможность напрямую обмениваться файлами и сообщениями (чат) без привлечения промежуточных серверов. Функции DCC Sending используются всякий раз при передаче файлов через IRC, а

1 *Исходные тексты программы вы можете найти на компакт-диске (каталог SRC/ebtables-v2.0.6). Эта программа включается в новые версии ядра Linux.*
2 *Direct Client-to-Client Protocol - протокол прямого взаимодействия между клиентами*

функции DCC Chat используются при работе с Eggdrop. Если вы используете NAT, данная опция позволит пересылать файлы и сообщения. Отметим, что эта опция не является обязательной для приема файлов или вызовов chat, а также реализации иных функций IRC.

Для реализации функций контроля соединений IRC в модуле **ip_conntrack_irc** выберите значение **M**.

4.4.2.2.14.2.1.3 IP_NF_TFTP

Опция **TFTP protocol support** (**YNM**) управляет возможность организации и отслеживания соединений TFTP при использовании трансляции адресов.

Если вы используете клиентские приложения tftp после фильтров **-j SNAT** или **-j MASQUERADING**, следует включить эту опцию.

При выборе значения **M** функции контроля соединений TFTP будут реализованы в модуле **ip_conntrack_tftp**.

4.4.2.2.14.2.1.4 IP_NF_AMANDA

Опция **Amanda backup protocol support** (**YNM**) служит для обеспечения возможности использования программ резервного копирования Amanda (http://www.amanda.org/) одновременно с трансляцией адресов (**MASQUERADING**). Опция позволяет создавать и контролировать соединения, а также обеспечивает поддержку субканалов, которые протокол Amanda требует при создании резервных копий, обмене сообщениями и индексировании.

Для реализации функций контроля соединений протокола Amanda в модуле **ip_conntrack_amanda** выберите значение **M**.

4.4.2.2.14.2.2 IP_NF_QUEUE

Опция **Userspace queueing via NETLINK** (**YNM**) управляет возможностью поддержки очередей пакетов в пользовательском пространстве. Этот драйвер использует доступ к таким очередям с помощью виртуального устройства **netlink** (Приложение 12.10). При включенной опции обеспечивается возможность использования операции QUEUE (см. параграф 5.1.8.1.3 на стр. 109) в цепочках фильтрации iptables.

Если выбрано значение опции M поддержка очередей пользовательского пространства реализуется в загружаемом модуле **ip_queue**.

4.4.2.2.14.2.3 IP_NF_IPTABLES

Опция **IP tables support (required for filtering/masq/NAT)** (**YNM**) служит для управления возможностью использования wtgjxtr **iptables**.

Для поддержки фильтрации пакетов и NAT (маскирование, пересылка между портами и т. п.) следует выбрать значение **Y** или **M**. Если был выбран модульный вариант, функции поддержки **iptables** реализуются в загружаемом модуле **ip_tables**.

4.4.2.2.14.2.3.1 IP_NF_MATCH_LIMIT

Опция **limit match support** (**YNM**) позволяет задавать для фильтров iptables уровень соответствия (число выполнения заданных условий в единицу времени), по достижении которого выполняется заданное действие. Эту возможность используют прежде всего для записи в журнальные файлы вместе в действием LOG (стр. 112) или ULOG (стр. 116) и предотвращения атак на службы (Denial of Service). Использование порогов соответствия рассматривается ниже (стр. 125).

При выборе значения **M** поддержка соответствия **limit** реализуется в загружаемом модуле **ipt_limit**.

4.4.2.2.14.2.3.2 IP_NF_MATCH_IPRANGE

Опция **IP range match support** (**YNM**) позволяет использовать в правилах фильтрации диапазон адресов IP взамен отдельных адресов. Включив эту опцию, вы сможете существенно снизить число правил фильтрации и, тем самым, ускорить работу фильтров. Описание работы с диапазонами адресов IP приведено ниже (стр. 121).

Если для опции выбрано значение **M**, функции поддержки диапазонов адресов реализуются в модуле **ipt_iprange**.

4.4.2.2.14.2.3.3 IP_NF_MATCH_MAC

Опция **MAC address match support** (**YNM**) позволяет использовать фильтрацию на основе MAC-адресов отправителей пакетов Ethernet. Проверка соответствия аппаратных адресов подробно рассматривается ниже (стр. 125).

Проверка соответствия MAC-адресов реализуется в виде загружаемого модуля **ipt_mac**, если для опции выбрано значение **M**.

4.4.2.2.14.2.3.4 IP_NF_MATCH_PKTTYPE

Опция **Packet type match support** (**YNM**) позволяет осуществлять фильтрацию по типу пакетов (BROADCAST, MULTICAST и т. п.) канального уровня. Примером использования такого соответствия может служить правило:

```
iptables -A INPUT -m pkttype --pkt-type broadcast -j LOG
```

Возможности проверки соответствия типа пакетов рассматриваются ниже (стр. 130).

Для реализации функций проверки типа пакетов в модуле **ipt_pkttype** выберите значение **M**.

4.4.2.2.14.2.3.5 IP_NF_MATCH_MARK

Опция **netfilter MARK match support** (**YNM**) позволяет использовать в правилах проверку соответствия значения **nfmark**, устанавливаемого с помощью операции MARK (см. параграф 5.1.8.2.9 на стр. 112). Маркировка пакетов поддерживается при выборе опции **MARK target support**, описанной ниже (стр. 79). Проверка соответствия маркеров описывается ниже (стр. 126).

Если вы хотите реализовать модульный вариант (**ipt_mark**) функций проверки маркеров, выберите для опции значение M.

4.4.2.2.14.2.3.6 IP_NF_MATCH_MULTIPORT

Опция **Multiple port match support** (**YNM**) позволяет использовать в правилах проверку соответствия множеству номеров портов TCP и UDP. Если эта опция отключена, проверятся может соответствие только конкретному (одному) номеру порта. Описание возможностей использования проверки соответствия множеству портов приводится ниже (стр. 127).

Если вы выберете для опции значение **M**, поддержка работы с группами портов будет реализована в загружаемом модуле **ipt_multiport**.

4.4.2.2.14.2.3.7 IP_NF_MATCH_TOS

Опция **TOS match support** (**YNM**) позволяет проверять соответствие значений поля TOS (тип обслуживания) в заголовках пакетов IP. Возможности проверки типа обслуживания рассматриваются в параграфе 5.1.9.5.7 (стр. 128).

При выборе значения **M** проверка типа обслуживания реализуется в виде модуля **ipt_tos**.

4.4.2.2.14.2.3.8 IP_NF_MATCH_RECENT

Опция **recent match support** (**YNM**) позволяет создавать один или множество списков недавно использованных адресов и проверять соответствие адреса в пакете адресам из таких списков. Краткую информацию о возможностях использования списков можно получить с помощью команды[1]

```
iptables -m recent -h
```

Более подробные сведения можно найти на официальном сайте проекта http://snowman.net/projects/ipt_recent/ и сайте Netfilter (http://netfilter.org/documentation/HOWTO//netfilter-extensions-HOWTO-3.html#ss3.16[2]). Использование возможностей модуля **recent** описано в параграфе 5.1.9.5.18 (стр. 131).

При выборе значения **M** функции работы со списками адресов будут реализованы в загружаемом модуле **ipt_recent**.

4.4.2.2.14.2.3.9 IP_NF_MATCH_ECN

Опция **ECN match support** (**YNM**) позволяет проверять соответствие значений полей ECN[3] в заголовках пакетов TCP протокола IPv4. Описание возможностей таких проверок приводится ниже (стр. 132).

Функции проверки поля ECN реализуются в модуле **ipt_ecn**, если для опции выбрано значение **M**.

4.4.2.2.14.2.3.10 IP_NF_MATCH_DSCP

Опция **DSCP match support** (**YNM**) управляет возможностью проверки соответствия значений полей DSCP в заголовках пактов IPv4. Поле DSCP (DSCP codepoint) может содержать значения в диапазоне от 0x0 до 0x4f. Описание возможностей модуля **dscp** приведено в параграфе 5.1.9.5.12 (стр. 129).

Проверка соответствия полей DSCP при для опции выборе значения **M** реализуется в форме модуля **ipt_dscp**.

4.4.2.2.14.2.3.11 IP_NF_MATCH_AH_ESP

Опция **AH/ESP match support** (**YNM**) управляет возможностью проверки соответствия диапазона значений SPI в заголовках AH и ESP пакетов IPSec[4]. Описание возможностей такой проверки приводится ниже (стр. 128).

При выборе для опции значения **M** проверка полей SPI будет реализована в загружаемом модуле **ipt_ah**.

4.4.2.2.14.2.3.12 IP_NF_MATCH_LENGTH

Опция **LENGTH match support** (**YNM**) управляет проверкой соответствия размера пакетов заданному диапазону значений. Описание возможностей такой проверки приведено на странице 130.

Если для опции выбрано значение **M**, функции проверки размера пакетов будут включены в загружаемый модуль **ipt_length**.

1 *Информация выдается лишь в тех случаях, когда данная опция была включена при компиляции ядра.*
2 *Копия этого документа в текстовом формате имеется в каталоге Documents/ приложенного к книге компакт-диска.*
3 *Explicit Congestion Notification -явное уведомление о насыщении*
4 *Протокол IP Authentification Header описан в RFC 2402 (http://rfc-editor.org/rfc/rfc2402.txt), копию которого вы найдете в каталоге Documents/ приложенного к книге компакт-диска..*

4.4.2.2.14.2.3.13 IP_NF_MATCH_TTL

Опция **TTL match support** (**YNM**) определяет возможность проверки значения времени жизни (TTL) в полях заголовков пакетов. Описание проверки TTL приведено на странице 128.

Если вы хотите использовать модульный (**ipt_ttl**) вариант проверки TTL, выберите для опции значение **M**.

4.4.2.2.14.2.3.14 IP_NF_MATCH_TCPMSS

Опция **tcpmss match support** (**YNM**) позволяет проверять соответствия значений поля MSS в заголовках пакетов TCP SYN, управляющие максимальным размером пакетов для организуемого соединения. Описание проверки соответствия приведено на странице 123.

Если для опции выбрано значение **M**, функции проверки полей MSS включаются в загружаемый модуль **ipt_tcpmss**.

4.4.2.2.14.2.3.15 IP_NF_MATCH_HELPER

Опция **Helper match support** (**YNM**) управляет возможностью проверки соответствия helper-модулей для динамических соединений, контролируемых такого типа модулями (например, **ip_conntrack_ftp**). Возможности использования модуля **helper** рассматриваются на странице 129.

При выборе значения **M** функции проверки соответствия helper-модулей включаются в загружаемый модуль **ipt_helper**.

4.4.2.2.14.2.3.16 IP_NF_MATCH_STATE

Опция **Connection state match support** (**YNM**) определяет возможность проверки состояния соединений. Это позволяет проверить принадлежность пакетов к тому или иному соединению и их связь с предыдущими пакетами. Описанная ниже проверка состояния соединений (стр. 127) дает широкие возможности классификации пакетов (например, маркировки).

Проверка состояния реализуется в форме модуля **ipt_state**, если для опции выбрано значение **M**.

4.4.2.2.14.2.3.17 IP_NF_MATCH_CONNTRACK

Опция **Connection tracking match support** (YNM) определяет возможность использования модуля **conntrack** в правилах iptables (стр. 129), который по сути является надмножеством функция проверки состояния (стр. 127). Машина состояний сетевых соединений Linux описана в параграфе 5.1.6.5 на стр. 105.

Эта опция позволяет проверять соответствие дополнительных параметров соединений, полезных для более сложных конфигураций (шлюзы с трансляцией адресов и многочисленными туннелями или каналами связи с Internet).

При выборе для опции значения **M**, функции проверки включаются в загружаемый модуль **ipt_conntrack**.

4.4.2.2.14.2.3.18 IP_NF_MATCH_OWNER

Опция **Owner match support** (**YNM**) позволяет проверять соответствие владельца пакета (для локально сгенерированных пакетов) определенному пользователю, группе, процессу или сеансу. Описание проверки владельца пакета приводится на странице 126.

Если для опции выбрано значение **M**, функции проверки идентификатора владельца включаются в загружаемый модуль **ipt_owner**.

4.4.2.2.14.2.3.19 IP_NF_MATCH_PHYSDEV

Опция **Physdev match support** (**YNM**) позволяет проверить соответствие пакета IP тому или иному физическому интерфейсу моста. Описание проверки соответствия пакетов физическому интерфейсу моста дано на странице 130.

При выборе значения **M** функции проверки соответствия физическому интерфейсу реализуются в модуле **ipt_physdev**.

4.4.2.2.14.2.3.20 IP_NF_FILTER

Опция **Packet filtering** (**YNM**) определяет возможность использования таблицы **filter** (фильтрация пакетов программой **iptables**), представляющей собой последовательность правил для приема, пересылки и отправки (локально сгенерированных) пакетов. Дополнительную информацию о таблице фильтрации можно получить с помощью команды **man iptables**.

Работа с таблицей **filter** и ее цепочками подробно рассматривается в параграфе 5.1.6.2 (стр. 104).

При выборе для опции значения M функции фильтрации пакетов реализуются в загружаемом модуле **iptable_filter**.

4.4.2.2.14.2.3.20.1 IP_NF_TARGET_REJECT

Опция **REJECT target support** (**YNM**) определяет возможность отвергать (операция **REJECT**) пакеты с генерацией сообщения ICMP. Операция **REJECT** подробно рассмотрена в параграфе 5.1.8.2.14.

Если для опции выбрано значение **M**, поддержка операции **REJECT** реализуется в загружаемом модуле **ipt_REJECT** .

4.4.2.2.14.2.3.21 IP_NF_NAT

Опция **Full NAT** (**YNM**) управляет использованием режима **Full NAT**, обеспечивающего поддержку маскирования, пересылки между портами и других вариантов преобразования адресов и номеров портов (NAPT). Для трансляции адресов служит таблица **nat**, рассмотренная в параграфе 5.1.6.3. Для получения базовых сведений о трансляции адресов с помощью программы iptables можно воспользоваться командой **man iptables**.

Функции поддержки трансляции адресов реализуются в модуле **iptable_nat**, если для опции выбрано значение **M**.

4.4.2.2.14.2.3.22 IP_NF_NAT_NEEDED

Состояние этой опции полностью определяется выбором других опций **netfilter**, поэтому она не выводится обычно в меню конфигурации ядра.

4.4.2.2.14.2.4 IP_NF_TARGET_MASQUERADE

Опция **MASQUERADE target support** (**YNM**) определяет возможность использования операции **MASQUERADE** (стр. 112), служащей для маскирования адресов. Маскирование представляет собой специальный случай трансляции адресов (NAT) - все исходящие соединения изменяются таким образом, чтобы они выглядели как связанные с заданным IP-адресом (при отключении интерфейса с этим адресом такие соединения разрываются). Такая трансляция адресов удобна для коммутируемых соединений с динамическим адресом IP .

Операция **MASQUERADE** подробно рассматривается в параграфе 5.1.8.2.10.

Если опция имеет значение **M**, поддержка операции **MASQUERADE** будет реализована в форме загружаемого модуля **ipt_MASQUERADE** .

4.4.2.2.14.2.5 IP_NF_TARGET_REDIRECT

Опция **REDIRECT target support** (**YNM**) определяет возможность использования в цепочках фильтрации операции **REDIRECT**, которая представляет собой частный случай трансляции адресов NAT - все входящие соединения отображаются на адрес принимающего интерфейса. В результате пакеты принимаются локальной машиной, а не пересылаются другим хостам. Такая трансляция полезна для создания прозрачных proxy.

Подробное рассмотрение операции **REDIRECT** приведено в параграфе 5.1.8.2.13.

При выборе для опции значения M, поддержка операции REDIRECT реализуется в форме загружаемого модуля **ipt_REDIRECT**.

4.4.2.2.14.2.6 IP_NF_TARGET_NETMAP

Опция **NETMAP target support** (**YNM**) обеспечивает поддержку операции **NETMAP** (статическое отображение сетевых адресов в режиме 1:1). Эта операция изменяет сетевую часть адреса, на меняя номер хоста. Она подобна функции Fast NAT, но последняя не позволяет функциям **netfilter** осуществлять контроль соединений. Рассмотрению операции **NETMAP** посвящен параграф 5.1.8.2.11.

При выборе значения **M**, поддержка операции **NETMAP** будет реализована в виде загружаемого модуля **ipt_NETMAP**.

4.4.2.2.14.2.7 IP_NF_TARGET_SAME

Опция **SAME target support** (YNM) управляет использованием операции SAME, похожей на стандартную операцию SNAT, но пытающуюся предоставить клиентам один адрес IP для всех соединений. Описанию операции **SAME** посвящен параграф 5.1.8.2.16.

При выборе для опции значения M поддержка операции **SAME** реализуется как загружаемый модуль i**pt_SAME**.

4.4.2.2.14.2.8 IP_NF_NAT_LOCAL

Опция **NAT of local connections** (**YN**) используется для управления возможностью использования функций NAT для локально сгенерированных пакетов. Эта опция может потребоваться при использовании трансляции DNAT для соединений, инициированных данным хостом.

Отметим, что для использования этой опции потребуется программа **iptables** версии 1.2.6a (или более новая)[1].

4.4.2.2.14.2.9 IP_NF_NAT_SNMP_BASIC

Экспериментальная опция **Basic SNMP-ALG support** (**YNM**) управляет использованием алгоритма ALG[2] для трафика SNMP. Вкупе с функциями NAT это позволяет системам сетевого управления получить доступ ко множеству частных сетей с конфликтующими (пересекающимися) адресами IP. Алгоритм изменяет IP-адреса в пакетах SNMP для приведения их в соответствие с трансляцией адресов на уровне IP.

Данная опция реализует базовый вариант алгоритма SNMP-ALG, описанный в RFC 2962[3].

Если для опции выбрано значение **M**, функции преобразования адресов для трафика SNMP реализуются в загружаемом модуле **ipt_nat_snmp_basic**.

1 *Программу можно загрузить с сайта www.iptables.org. На приложенном к книге компакт-диске имеются исходные тексты программы iptables (каталог SRC/).*
2 *Application Layer Gateway - шлюз прикладного уровня*
3 *Документ можно загрузить с сайта http://rfc-editor.org/rfc/rfc2962.txt.*

4.4.2.2.14.2.10 IP_NF_NAT_IRC

Опция **IP_NF_NAT_IRC** (**YNM**) определяет возможность трансляции адресов для соединений IRC. Значение этой опции определяется выбором опции **IRC protocol support** (параграф 4.4.2.2.14.2.1.2). При значении **M** создается загружаемый модуль **ip_nat_irc**.

4.4.2.2.14.2.11 IP_NF_NAT_FTP

Опция **IP_NF_NAT_FTP** (**YNM**) определяет возможность трансляции адресов для соединений FTP. Значение этой опции определяется выбором опции **FTP protocol support** (параграф 4.4.2.2.14.2.1.1). При значении **M** создается загружаемый модуль **ip_nat_ftp**.

4.4.2.2.14.2.12 IP_NF_NAT_TFTP

Опция **IP_NF_NAT_TFTP** (**YNM**) определяет возможность трансляции адресов для соединений TFTP. Значение этой опции определяется выбором опции **TFTP protocol support** (параграф 4.4.2.2.14.2.1.3). При значении **M** создается загружаемый модуль **ip_nat_tftp**.

4.4.2.2.14.2.13 IP_NF_NAT_AMANDA

Опция **IP_NF_NAT_AMANDA** (**YNM**) определяет возможность трансляции адресов для соединений AMANDA. Значение этой опции определяется выбором опции **Amanda backup protocol support** (параграф 4.4.2.2.14.2.1.4). При значении **M** создается загружаемый модуль **ip_nat_amanda**.

4.4.2.2.14.2.14 IP_NF_MANGLE

Опция **Packet mangling** (**YNM**) управляет возможностью модификации пакетов, осуществляемой с помощью таблицы **mangle** программы **iptables**. Краткую информацию об этой таблице можно получить с помощью команды **man iptables**, а подробное рассмотрение таблицы **mangle** приведено в параграфе 5.1.6.4 (стр. 105). Цепочки таблицы **mangle** служат для изменения некоторых полей в заголовках проходящих через маршрутизатор пакетов IP.

Если вы хотите реализовать поддержку операций по изменению пакетов в виде модуля **iptable_mangle**, выберите для опции значение **M**.

4.4.2.2.14.2.14.1 IP_NF_TARGET_TOS

Опция **TOS target support** (**YNM**) определяет возможность использования операции **TOS**, позволяющей создавать в таблице **mangle** правила, изменяющие значение поля TOS (тип обслуживания) в пакетах IP до маршрутизации. Если маршрутизатор может принимать во внимание тип обслуживания пакетов, эта опция позволяет существенно влиять на обработку трафика. Операция **TOS** подробно рассматривается в параграфе 5.1.8.2.19 (стр. 115).

При выборе для опции значения **M** функции поддержки операции TOS будут реализованы в загружаемом модуле **ipt_TOS**.

4.4.2.2.14.2.14.2 IP_NF_TARGET_ECN

Опция **ECN target support** (**YNM**) управляет возможностью удаления битов ECN[1] из заголовков IPv4 с помощью правил таблицы **mangle**. Удаление этих битов может быть весьма полезно при работе в окрестности “черных дыр” ECN в Internet, однако не следует удалять эти биты во всех пакетах. Операция **ECN** подробно рассматривается в параграфе 5.1.8.2.7 (стр. 111).

Если вы выберете для опции значение M, поддержка операции ECN будет реализована в загружаемом модуле **ipt_ECN**.

4.4.2.2.14.2.14.3 IP_NF_TARGET_DSCP

Опция **DSCP target support** (**YNM**) позволяет изменять значения полей DSCP в заголовках пакетов IPv4 (DSCP codepoint). Операция DSCP описывается в параграфе 5.1.8.2.6 (стр. 111).

При выборе значения **M** функции поддержки операции DSCP будут реализованы в форме загружаемого модуля **ipt_DSCP**.

4.4.2.2.14.2.14.4 IP_NF_TARGET_MARK

Опция **MARK target support** (**YNM**) позволяет использовать описанную ниже (параграф 5.1.8.2.9) операцию **MARK** в цепочках таблицы **mangle** (см. параграф 5.1.6.4 на стр. 105) для изменения поля маркера (**nfmark**) до маршрутизации пакетов. Значения маркера можно использовать для маршрутизации (см. стр. 70), а также иных операций с пакетами IP.

Поддержка операции MARK будет реализована в загружаемом модуле **ipt_MARK**, если для этой опции вы выберете значение **M**.

4.4.2.2.14.2.14.5 IP_NF_TARGET_CLASSIFY

Опция **CLASSIFY target support** (**YNM**) управляет возможностью использования операции CLASSIFY (см. параграф 5.1.8.2.2 на стр. 110), которая позволяет устанавливать для пакетов уровень приоритета. Некоторые системы

1 *Флаги уведомления о насыщении канала на пути доставки.*

управления трафиком qdiscs (Classful Queuing Disciplines) могут использовать эти уровни для классификации пакетов. К таким системам относятся в частности atm, cbq, dsmark, pfifo_fast, htb, prio.

Выбор для опции значения M приведет к реализации функций поддержки операции **CLASSIFY** в загружаемом модуле **ipt_CLASSIFY** .

4.4.2.2.14.2.15 IP_NF_TARGET_LOG

Опция **LOG target support** (**YNM**) управляет возможностью использования операции **LOG** (см. параграф 5.1.8.2.8 на стр. 112), позволяющей создавать правила **iptables**, записывающие заголовки пакетов в журнальные файлы **syslog**.

При выборе для опции значения **M** операция записи в журнальный файл будет реализована в форме модуля **ipt_LOG**.

4.4.2.2.14.2.16 IP_NF_TARGET_ULOG

Опция **ULOG target support** (**YNM**) управляет поддержкой операции **ULOG** (см. параграф 5.1.8.2.22 на стр. 116), позволяющей создавать правила протоколирования во всех таблицах **iptables**. Эта операция передает пакеты демону протоколирования из пользовательского пространства (с помощью сокетов **netlink**, описанных в Приложении 12.10) в отличие от операции **LOG**, работающей только через **syslog**.

Исходные тексты пользовательского демона протоколирования **ulogd** можно загрузить с сайта http://www.gnumonks.org/projects/ulogd/[1].

Выбор значения **M** приведет к реализации функций записи в пользовательские журнальные файлы в форме загружаемого модуля **ipt_ULOG**.

4.4.2.2.14.2.17 IP_NF_TARGET_TCPMSS

Опция **TCPMSS target support** (**YNM**) позволяет использовать операцию **TCPMSS** (см. параграф 5.1.8.2.18 на стр. 115), дающую возможность изменять значения MSS в заголовках пакетов TCP SYN для управления максимальным размером пакетов[2] для соединения.

Управление максимальным размером пакетов часто приходится использовать для обуздания страдающих паранойей ISP и серверов, блокирующих пакеты ICMP Fragmentation Needed. Симптомом этой проблемы является то, что на самой Linux-машине все работает хорошо, а расположенных за ней станции не могут обмениваться пакетами большого размера:

1) Web-браузеры подключаются к сайтам, но не могут получить с них информации;
2) мелкие почтовые сообщения приходят нормально, а крупные не доходят;
3) протокол ssh работает нормально, scp прекращает работу после начального согласования параметров (handshaking).

Для решения проблемы следует активизировать данную опцию и включить в цепочки вашего брандмауэра правило типа приведенного ниже:

```
iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS -clamp-mss-to-pmtu
```

Для реализации функций поддержки операции **TCPMSS** в форме загружаемого модуля **ipt_TCPMSS** выберите значение **M**.

4.4.2.2.14.2.18 IP_NF_ARPTABLES

Опция **ARP tables support** (**YNM**) управляет возможностью фильтрации и изменения пакетов ARP. Для работы с пакетами ARP существует специальная программа **arptables** (параграф 5.3 на стр. 151).

Если для опции выбрано значение **M**, функции поддержки фильтрации и изменения пакетов ARP включаются в модуль **arp_tables**.

4.4.2.2.14.2.18.1 IP_NF_ARPFILTER

Опция **ARP packet filtering** (**YNM**) позволяет включить или отключить фильтрацию (таблица **filter**, описанная в параграфе 5.3.2) пакетов ARP, адресованных данному хосту или сгенерированных им. В мостах фильтры могут использоваться также для пересылаемых пакетов (см. параграф 5.3 на стр. 151).

Для создания модуля **arptable_filter** выберите для опции значение **M**.

4.4.2.2.14.2.18.2 IP_NF_ARP_MANGLE

Опция **ARP payload mangling** (**YNM**) управляет возможностью изменения пакетов ARP (сетевые и аппаратные адреса отправителя и получателя). Операции изменения пакетов ARP описаны в параграфе 5.3.5 (стр. 153).

Чтобы реализовать функции изменения пакетов ARP в модуле **arpt_mangle**, укажите для опции значение **M**.

1 *В каталоге SRC/ приложенного к книге компакт-диска вы сможете найти исходные тексты этого демона и файлы документации.*
2 *Обычно в качестве максимального размера пакетов используется значение MTU для передающего интерфейса - 40 байтов*

4.4.2.2.14.2.19 IP_NF_COMPAT_IPCHAINS

Опция **ipchains (2.2-style) support** (**YNM**) управляет поддержкой фильтрации пакетов с помощью программы **ipchains**, применявшейся в ядрах серии 2.2.x. Эта опция служит только для обратной совместимости и ее не следует включать при установке ядра на новые станции.

Вы можете создать модуль (**ipchains**) поддержки совместимости с **ipchains**, если для опции будет выбрано значение **M**.

4.4.2.2.14.2.19.1 IP_NF_COMPAT_IPFWADM

Опция **ipfwadm (2.0-style) support** (**YNM**) позволяет включить поддержку фильтров **ipfwadm**, использовавшихся в ядрах серии 2.0.x. Эта опция служит только для обратной совместимости и ее не следует включать при установке ядра на новые станции.

Можно реализовать функции поддержки совместимости с **ipfwadm** в форме загружаемого модуля **ipfwadm**, если выбрать для опции значение M.

4.4.2.2.14.2.20 IP_NF_TARGET_NOTRACK

Опция **NOTRACK target support** (**YNM**) управляет возможностью использования операции **NOTRACK**, позволяющей задать правила для исключения пакетов из числа передаваемых подсистеме контроля соединений/NAT. Описание работы с **NOTRACK** приведено в параграфе 5.1.8.2.12.

Для реализации функций поддержки операции **NOTRACK** в модуле **ipt_NOTRACK** выберите для опции значение **M**.

4.4.2.2.14.2.21 IP_NF_RAW

Опция **raw table support (required for NOTRACK/TRACE)** (**YNM**) управляет возможностью использования программой **iptables** таблицы **raw**, описанной в параграфе 5.1.6.1. Эта таблица используется программами netfilter в первую очередь (еще до функций отслеживания соединений) и включает цепочки **PREROUTING** (стр. 103) и **OUTPUT** (стр. 104).

Чтобы работать с таблицей raw как загружаемым модулем **ipt_RAW**, выберите для опции значение **M**.

4.4.2.2.14.2.22 IP_NF_MATCH_ADDRTYPE

Опция **address type match support** (**YNM**) позволяет использовать в правилах **iptables** проверку соответствия типа адреса сетевого уровня (**UNICAST**, **LOCAL**, **BROADCAST** и т. п.). Описание работы с модулем проверки типа сетевого адреса приведено в параграфе 5.1.9.5.17 (стр. 130).

Выбор для опции значения **M** приведет к реализации функций проверки типа адреса в загружаемом модуле **ipt_addrtype**.

4.4.2.2.14.2.23 IP_NF_MATCH_REALM

Опция **realm match support** (**YNM**) управляет возможностью использования модуля **realm** (параграф 5.1.9.6.9) программы **iptables** для проверки соответствия полученных от локальной системы маршрутизации ключей **realm**.

Использование этой проверки очень похоже на классификацию, обеспечиваемую для системы управления трафиком QoS опцией классификации **NET_CLS_ROUTE4** (параграф 4.4.2.2.33.2.19.2).

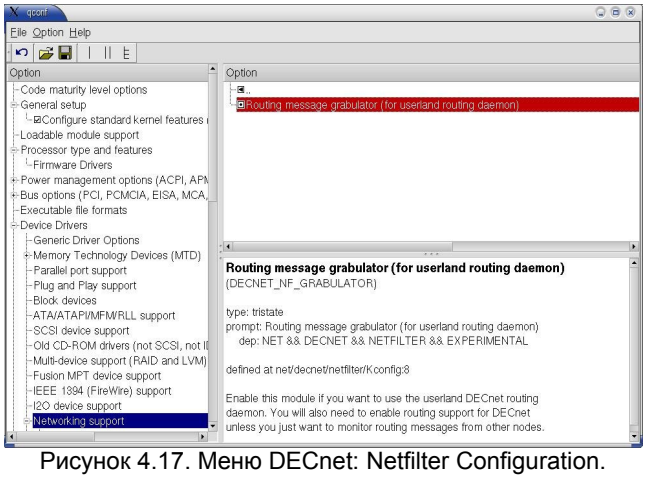


Рисунок 4.17. Меню DECnet: Netfilter Configuration.

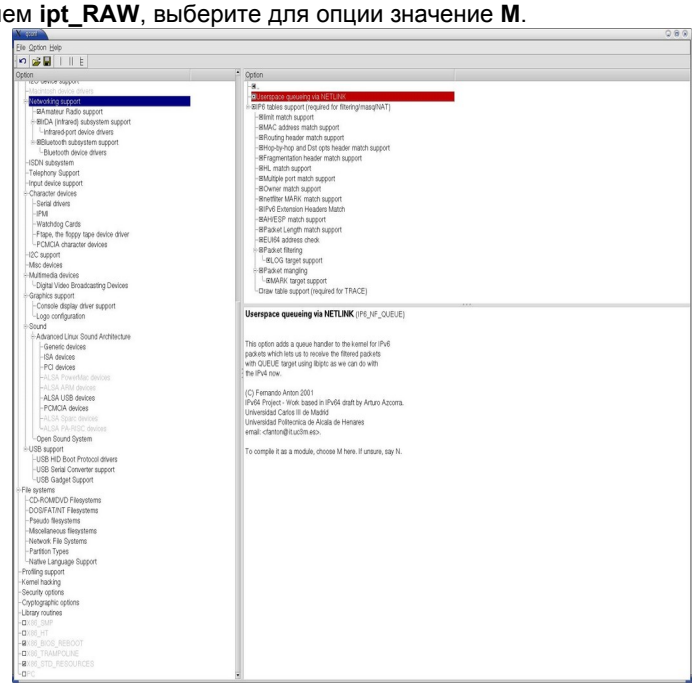


Рисунок 4.16. Меню IPv6: Netfilter Configuration.

При выборе для опции значения **M** функции поддержки **realm** будут реализованы в загружаемом модуле **ipt_realm**.

4.4.2.2.14.3 Меню IPv6: Netfilter Configuration

Поскольку рассмотрение протокола IPv6 выходит за рамки книги, мы ограничимся рисунком 4.16, на котором показаны опции конфигурации netfilter/iptables для этого протокола.

4.4.2.2.14.4 Меню DECnet: Netfilter Configuration

Рассмотрение протоколов DECnet выходит за рамки книги, поэтому мы ограничимся иллюстрацией (рисунок 4.17), показывающей список опций netfilter/iptables для данного протокола.

4.4.2.2.14.5 Меню Bridge: Netfilter Configuration

Это меню служит для управления возможностями фильтрации кадров мостами (канальный уровень). Опции меню становятся доступными при включенной фильтрации пакетов и выборе значения **Y** или **M** для опции **802.1d Ethernet Bridging** (см. параграф 4.4.2.2.19 на стр. 84)

4.4.2.2.14.5.1 BRIDGE_NF_EBTABLES

Опция **Ethernet Bridge tables (ebtables) support** (**YNM**) управляет возможностью использования программы **ebtables** (стр. 143), обеспечивающей функции идентификации кадров на канальном уровне для их фильтрации и изменения.

При выборе значения **M** функции поддержки **ebtables** реализуются в одноименном модуле.

4.4.2.2.14.5.1.1 BRIDGE_EBT_BROUTE

Опция **ebt: broute table support** (**YNM**) определяет возможность использования программой **ebtables** (стр. 143) таблицы **broute** (стр. 143), служащей для задания правил, определяющих использование для кадров режима моста или маршрутизатора. Это позволяет хостам Linux поддерживать функции мостов-маршрутизаторов (brouter). Информацию о программе **ebtables** можно получить по команде **man ebtables** или на сайте проекта http://ebtables.sourceforge.net/. Краткое описание этой программы приводится ниже в разделе 5.2, а исходные тексты программы вы найдете в каталоге SRC/ приложенного к книге компакт-диска.

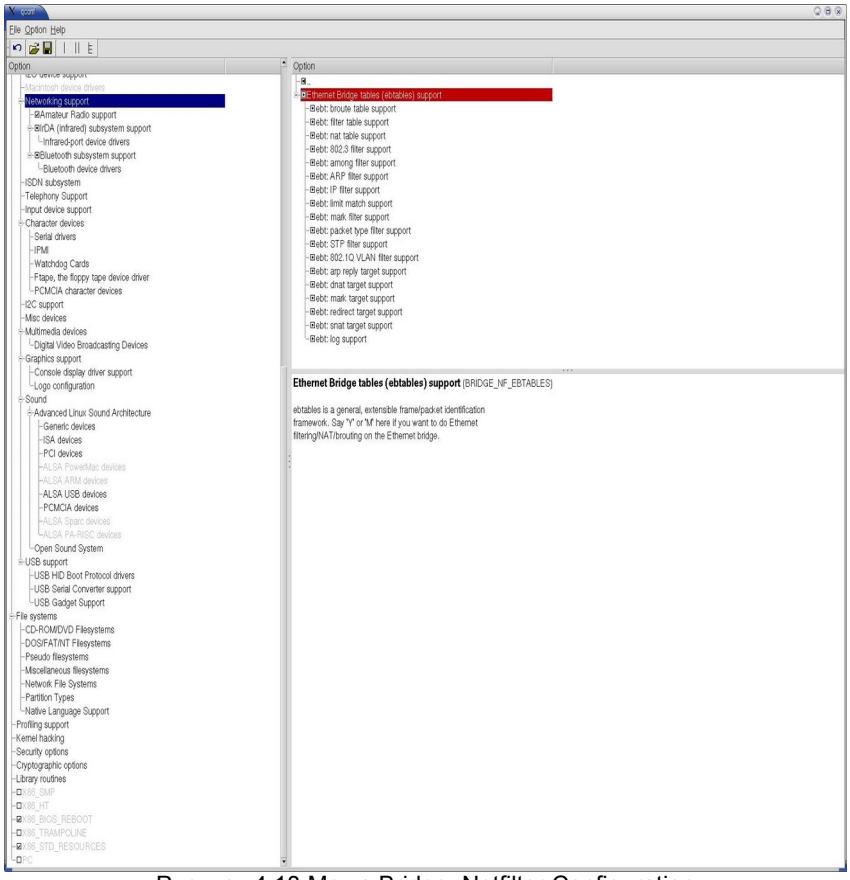


Рисунок 4.18 Меню Bridge: Netfilter Configuration

Если для опции выбрано значение **M**, поддержка таблицы broute реализуется в модуле **ebtable_broute**.

4.4.2.2.14.5.1.2 BRIDGE_EBT_T_FILTER

Опция **ebt: filter table support** (**YNM**) управляет использованием таблицы **filter**, которая содержит цепочки правил фильтрации кадров, принимаемых, пересылаемых или порожденных данным мостом. Работа с таблицей фильтрации кадров рассматривается в параграфе 5.2.2.1 (стр. 143).

Для включения функций фильтрации в загружаемый модуль **ebtable_filter** установите для опции значение **M**.

4.4.2.2.14.5.1.3 BRIDGE_EBT_T_NAT

Опция **ebt: nat table support** (**YNM**) определяет возможность использования программой **ebtables** таблицы **nat**, служащей для преобразования MAC-адресов отправителей (MAC SNAT) и получателей (MAC DNAT). Описание таблицы **nat** приведено на странице 143.

Функции трансляции адресов можно реализовать в виде модуля **ebtable_nat**, если выбрать для опции значение **M**.

4.4.2.2.14.5.1.4 BRIDGE_EBT_802_3

Опция **ebt: 802.3 filter support** (**YNM**) позволяет проверять соответствие полей **DSAP**/**SSAP** и **SNAP** для кадров 802.3 Ethernet. Описание этой проверки приводится в параграфе 5.2.6.1 (стр. 146).

При выборе для опции значения M функции будут реализованы в форме загружаемого модуля **ebt_802_3**.

4.4.2.2.14.5.1.5 BRIDGE_EBT_AMONG

Опция **ebt: among filter support** (**YNM**) позволяет проверять соответствие MAC-адресов отправителя и получателя имеющемуся списку адресов с помощью соответствия **among** (параграф 5.2.6.2 на стр. 146) программы **ebtables**. Кроме того, могут поддерживаться списки соответствия MAC - IP, позволяющие создавать правила для блокирования обманных адресов (anti-spoofing).

Если для опции было выбрано значение **M**, функции будут реализованы в загружаемом модуле **ebt_among**.

4.4.2.2.14.5.1.6 BRIDGE_EBT_ARP

Опция **ebt: ARP filter support** (**YNM**) позволяет проверять соответствие ARP, обеспечивающее для программы **ebtables** возможность фильтрации по полям заголовков пакетов ARP и RARP с помощью проверки соответствия **arp** (параграф 5.2.6.3 на стр. 147).

При выборе для опции значения **M** функции поддержки фильтрации ARP будут реализованы в загружаемом модуле **ebt_arp**.

4.4.2.2.14.5.1.7 BRIDGE_EBT_IP

Опция **ebt: IP filter support** (**YNM**) позволяет проверять соответствие основных полей заголовков IP (см. параграф 5.2.6.4 на стр. 147) для реализации базовых функций IP-фильтрации на уровне моста.

При выборе для опции значения **M** функции проверки полей IP будут реализованы в форме загружаемого модуля **ebt_ip**.

4.4.2.2.14.5.1.8 BRIDGE_EBT_LIMIT

Опция **ebt: limit match support** (**YNM**) позволяет устанавливать пороговые значения для уровня соответствия (число совпадений в единицу времени) правилам **ebtables**. Использование этой опции может быть весьма полезно при записи информации в журнальные файлы. Описание проверки частоты совпадения приводится в параграфе 5.2.6.5.

Если для опции было выбрано значение M, функции будут реализованы в загружаемом модуле **ebt_limit**.

4.4.2.2.14.5.1.9 BRIDGE_EBT_MARK

Опция **ebt: mark filter support** (**YNM**) позволяет проверять соответствие маркеров (поле **nfmark** в кадрах), которые могут устанавливаться операцией **mark** (параграф 5.2.8.3). Для проверки соответствия используется опция **mark** (параграф 5.2.6.6) программы **ebtables**.

Если для опции выбрано значение **M**, функции работы с маркерами реализуются в загружаемом модуле **ebt_mark_m**.

4.4.2.2.14.5.1.10 BRIDGE_EBT_PKTTYPE

Опция **ebt: packet type filter support** (**YNM**) позволяет проверять соответствие типа кадров (класса Ethernet) - broadcast, multicast и т. п. С помощью соответствия **pkttype** (параграф 5.2.6.7). программы **ebtables**. При выборе для опции значения **M**, код проверки типа кадров будет сохранен в виде загружаемого модуля **ebt_pkttype**.

4.4.2.2.14.5.1.11 BRIDGE_EBT_STP

Опция **ebt: STP filter support** (**YNM**) позволяет использовать правила соответствия для протокола STP (Spanning Tree), обеспечивающие возможность фильтрации по заголовкам пакетов STP. Для проверки полей STP используется соответствие **stp** (параграф 5.2.6.8) программы **ebtables**.

При выборе для опции значения **M** создается загружаемый модуль **ebt_stp**.

4.4.2.2.14.5.1.12 BRIDGE_EBT_VLAN

Опция **ebt: 802.1Q VLAN filter support** (**YNM**) позволяет проверять соответствие тегов 802.1Q при создании правил фильтрации пакетов. Для проверки тегов используется соответствие **vlan** (параграф 5.2.6.9 на стр. 149) программы **ebtables**.

Если вы выбрали для опции значение **M**, код проверки тегов будет скомпилирован в форме модуля **ebt_vlan**.

4.4.2.2.14.5.1.13 BRIDGE_EBT_ARPREPLY

Опция **ebt: arp reply target support** (**YNM**) управляет возможностью использования программой **ebtables** проверки соответствия **arpreply** (параграф 5.2.8.1 на стр. 150), которая позволяет автоматически передавать отклики **ARP** на соответствующие запросы.

При выборе для опции значения **M** функции для работы с откликами ARP реализуется в загружаемом модуле **ebt_arpreply**.

4.4.2.2.14.5.1.14 BRIDGE_EBT_DNAT

Опция **ebt: dnat target support** (**YNM**) управляет возможностью использования операции **DNAT** (параграф 5.2.8.2 на стр. 150), служащей для изменения в кадрах MAC-адреса получателя с помощью правил программы **ebtables**.

Если для опции выбрано значение M, код поддержки трансляции MAC-адресов будет реализован в виде загружаемого модуля **ebt_dnat**.

4.4.2.2.14.5.1.15 BRIDGE_EBT_MARK_T

Опция **ebt: mark target support** (**YNM**) управляет возможностью использования операции **mark** (параграф 5.2.8.3 на стр. 150), служащей для маркировки кадров (поле nfmark). Проверка маркеров может осуществляться с помощью соответствия **mark** (параграф 5.2.6.6 на стр. 148) программы **ebtables**.

Если для опции выбрано значение **M**, функции работы с маркерами реализуются в загружаемом модуле **ebt_mark**.

4.4.2.2.14.5.1.16 BRIDGE_EBT_REDIRECT

Опция **ebt: redirect target support** (**YNM**) определяет возможность использования операции **redirect** (параграф

5.2.8.4 на стр. 150) в правилах **ebtables**, позволяющей изменить в кадрах MAC-адрес получателя на адрес принявшего этот кадр интерфейса.

При выборе значения M код изменения MAC-адресов будет реализован в загружаемом модуле **ebt_redirect**.

4.4.2.2.14.5.1.17 BRIDGE_EBT_SNAT

Опция **ebt: snat target support** (**YNM**) управляет возможностью использования операции **SNAT** (параграф 5.2.8.5 на стр. 151) программы **ebtables**, служащей для изменения в кадрах MAC-адреса отправителя.

При выборе для опции значения **M** код замены MAC-адреса отправителя будет реализован в загружаемом модуле **ebt_snat**.

4.4.2.2.14.5.1.18 BRIDGE_EBT_LOG

Опция **ebt: log support** (**YNM**) определяет возможность использования в правилах **ebtables** операции **log** (параграф 5.2.7.1 на стр. 149), служащей для записи заголовков кадров в журнальные файлы **syslog**.

Если для опции было выбрано значение **M**, функции записи заголовков в журнальные файлы реализуются в загружаемом модуле **ebt_log**.

4.4.2.2.15 XFRM

Опция **XFRM** устанавливается автоматически при активизации сетевой поддержки в ядре (см. описанную на стр. 68 опцию **NET**), поэтому ее значение нельзя изменить и обычно эта опция не выводится в меню. Методы XFRM используются для обработки пакетов в ядрах, начиная с версии 2.5. Информацию об XFRM можно найти на сайте http://archive.linuxsymposium.org/ols2003/Proceedings/All-Reprints/Reprint-Yoshifuji-OLS2003.pdf[1].

4.4.2.2.16 XFRM_USER

Опция **IPsec user configuration interface** определяет поддержку пользовательского интерфейса IPsec для приложений Linux. Разумно для этой опции выбрать значение **Y**.

4.4.2.2.17 Меню SCTP Configuration

Экспериментальное меню **The SCTP Protocol** служит для выбора опций поддержки в ядре потокового протокола **SCTP**[2], описанного в RFC 2960.

Рассмотрение возможностей экспериментальных протоколов выходит за рамки книги, поэтому мы просто порекомендуем отключить в ядре поддержку протокола SCTP, если вы не планируете создать компьютер для экспериментов. Опции меню **The SCTP Protocol** вы можете увидеть на рисунке 4.19.

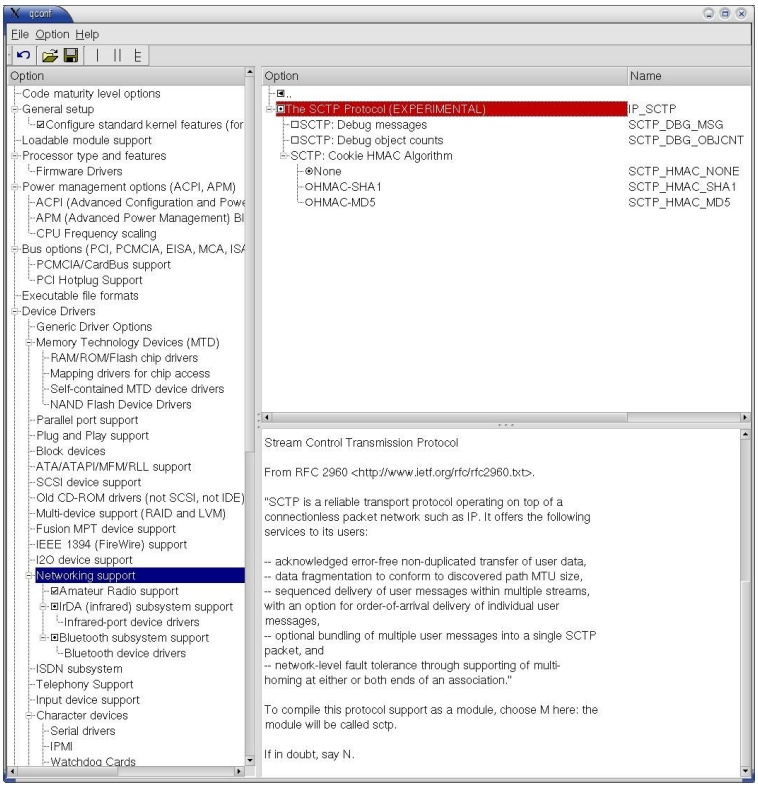


Рисунок 4.19. Меню The SCTP Protocol.

4.4.2.2.18 ATM

Экспериментальная опция **Asynchronous Transfer Mode (ATM)** (**YNM**) управляет поддержкой ядром функций работы с устройствами и протоколами ATM. Рассмотрение этого вопроса выходит за рамки книги. Если вы не используете адаптеров ATM в своем компьютере, выберите для этой опции значение **N**. Информацию о поддержке ATM ядром Linux можно найти в файле **Documentation/networking/atm.txt** дистрибутива ядра.

4.4.2.2.19 BRIDGE

Опция **802.1d Ethernet Bridging** (**YNM**) управляет поддержкой функций моста Ethernet ядром Linux. Мост позволяет соединить между собой два различных сегмента Ethernet так, чтобы они представлялись как единая сеть Ethernet. Множество соединенных между собой мостов могут создавать крупные сети Ethernet, использующие алгоритм IEEE 802.1 Spanning Tree. При включенной опции хост Linux (мост) обеспечивает интероперабельность с мостами других производителей.

Информация по использованию мостов содержится в файле **Documentation/networking/bridge.txt** дистрибутива ядра.

При включенной поддержке функций моста и фильтрации пакетов (параграф 4.4.2.2.14.1 на стр. 73) вы можете создать мост с поддержкой функций межсетевого экрана (IP firewall). Программа **iptables** (раздел 5.1) будет

1 *Копия этого документа присутствует в каталоге Documents/ приложенного к книге компакт-диска.*
2 *Stream Control Transmission Protocol - протокол управления потоковой передачей.*

просматривать проходящие через мост пакеты IP и поступать с ними в соответствии с заданными правилами фильтрации. Для работы с кадрами Ethernet на канальном уровне может использоваться также специальная программа **ebtables** (раздел 5.2), если были включены соответствующие опции (параграф 4.4.2.2.14.5). При включенной поддержке **arptables** (параграф 4.4.2.2.14.2.18.1 на стр. 80) проходящий через мост трафик ARP будет пропускаться через цепочку фильтров **FORWARD** (параграф 5.3.2 на стр. 151).

При использовании для этой опции значения **M** будет создан модуль **bridge**.

Если вы не уверены в необходимости поддержки функций моста на данном компьютере, выберите значение **N**.

4.4.2.2.20 VLAN_8021Q

Опция **802.1Q VLAN Support** (**YNM**) управляет поддержкой виртуальных сетей в соответствии со стандартов IEEE 802.1Q.

При выборе для опции значения **M** функции работы с тегами 802.1Q будут реализованы в модуле **8021q**.

4.4.2.2.21 DECNET

Опция **DECnet Support** (**YNM**) определяет поддержку стека протоколов DECnet. В настоящее время использование этих протоколов достаточно экзотично и мы не будем останавливаться на их рассмотрении. Интересующиеся могут найти информацию на сайте http://linux-decnet.sourceforge.net/ и в файле **Documentation/networking/decnet.txt** дистрибутива ядра Linux.

Если вы решили включить поддержку протоколов DECnet, не забудьте установить значения **Y** для описанных выше опций **/proc file system support** (стр. 67) и **Sysctl support** (стр. 64).

Вы можете включить поддержку сетевых функций DECnet в виде загружаемого модуля (**decnet**).

4.4.2.2.22 LLC

Значение опции LLC определяется установкой описанной выше опции **NET** (стр. 68).

4.4.2.2.23 LLC2

Опция **ANSI/IEEE 802.2 LLC type 2 Support** (**YNM**) определяет поддержку соединений канального уровня типа 2. Включайте эту опцию, если вы планируете использовать сокеты **PF_LLC**.

4.4.2.2.24 IPX

Опция **The IPX protocol** (**YNM**) управляет поддержкой функций стека сетевых протоколов IPX/SPX компании Novell. Мы не будем останавливаться на рассмотрении этой темы и укажем лишь источник информации http://www.tldp.org/docs.html#howto.

Драйвер IPX увеличивает размер ядра приблизительно на 16 Кб. При выборе для опции значения **M** будет создан загружаемый модуль **ipx**.

Если вы не планируете соединять Linux-машину с локальной сетью Novell, выберите значение **N**.

4.4.2.2.25 ATALK

Опция **Appletalk protocol support** (**YNM**) управляет поддержкой стека протоколов AppleTalk компании Apple computers. Мы не будем рассматривать здесь вопросы использования протоколов AppleTalk и укажем лишь ссылку на источник информации - http://www.tldp.org/docs.html#howto.

При выборе для опции значения **M** будет создан модуль **appletalk**. Если вы не планируете связывать Linux-машину с компьютерами Macintosh, выберите значение **N**.

4.4.2.2.26 X25

Экспериментальная опция **CCITT X.25 Packet Layer** (**YNM**) определяет поддержку стека протоколов X.25. Желающие узнать больше о поддержке X.25 могут прочесть это на сайте http://www.sangoma.com/x25.htm и в файлах **Documentation/networking/x25.txt**, **Documentation/networking/x25-iface.txt** дистрибутива ядра.

4.4.2.2.27 LAPB

Экспериментальная опция **LAPB Data Link Driver** (YNM) управляет поддержкой протокола **LAPB**[1], используемого на канальном уровне X.25. Обычно протокол LAPB используется со специальными сетевыми адаптерами X.21, но Linux в настоящее время поддерживает LAPB только для устройств Ethernet. Если вы планируете использовать такие соединения, выберите также значение **Y** для опции **LAPB over Ethernet driver** (параграф 4.4.2.7.1.7.3 на стр. 94). Дополнительную информацию можно найти в файле **Documentation/networking/lapb-module.txt** дистрибутива ядра.

При выборе для опции значения **M** будет создан загружаемый драйвер **lapb**. Если вы не уверены в необходимости поддержки протокола LAPB, выберите значение **N**.

1 *Link Access Procedure, Balanced*

4.4.2.2.28 NET_DIVERT

Экспериментальная опция **Frame Diverter** управляет использованием функций “отвода” из сети пакетов, которые не адресованы непосредственно принявшему интерфейсу. Это позволяет использовать мост Linux со включенной поддержкой Frames Diverter и proxy-сервером (например, Squid) в качестве прозрачного кэша www. Такая возможность очень эффективна для тех случаев, когда отсутствует возможность изменения конфигурации маршрутизатора. Кроме того, эти функции можно использовать для решения целого ряда задач:

- пересылка трафика SMTP на другой интерфейс;
- формирование трафика для некоторых сетевых потоков;
- создание прозрачных proxy-соединений для протокола smtp и т. п.

Дополнительную информацию можно найти на сайте http://diverter.sourceforge.net/ или http://perso.wanadoo.fr/magpie/EtherDivert.html.

Если вы не уверены в необходимости поддержки функций “отвода” пакетов, выберите значение **N**.

4.4.2.2.29 ECONET

Экспериментальная опция **Acorn Econet/AUN protocols** управляет поддержкой устаревшего протокола Econet, используемого в компьютерах Acorn и мы не будем останавливаться на ее обсуждении.

4.4.2.2.30 WAN_ROUTER

Опция **WAN router** (YNM) управляет использованием Linux-машины в качестве WAN-маршрутизатора.

Распределенные сети (Wide Area Networks или WAN), построенные на основе X.25, frame relay и выделенных линий, зачастую используются для связи между собой расположенных на значительном удалении одна от другой ЛВС. Такое решение обеспечивает существенное повышение скорости обмена информацией по сравнению со связью между сетями на базе асинхронных модемных каналов. Обычно для подключения каналов WAN используются достаточно дорогие маршрутизаторами с одним или несколькими синхронными портами. Альтернативой таким устройствам могут служить маршрутизаторы на базе Linux, в которых используются недорогие интерфейсные платы с синхронными портами.

Если вы планируете создать такой маршрутизатор, выберите для этой опции значение Y и не забудьте включить опцию драйвера для используемого в вашем компьютере адаптера WAN. Для работы маршрутизатора вам также потребуется пакет программ wanpipe, доступный на сайте ftp://ftp.sangoma.com/[1]. Дополнительную информацию можно найти в файле Documentation/networking/wan-router.txt дистрибутива ядра Linux.

При выборе для опции значения M функции WAN-маршрутизатора будут включены в загружаемый модуль **wanrouter**.

Для выбора драйверов устройств, поддерживающих WAN-интерфейсы служит меню **Wan interfaces**, описанное в параграфе 4.4.2.7.1.7 (стр. 93).

4.4.2.2.31 NET_FASTROUTE

Опция **Fast switching**[2] управляет возможностью быстрой пересылки пакетов непосредственно между парами сетевых адаптеров, установленных в компьютере. Отметим, что использование быстрой пересылки **несовместимо** с функциями фильтрации пакетов (опция **NETFILTER**, стр. 73).

Однако быстрая пересылка между интерфейсами совместима с функциями маршрутизации (опции **Advanced router** - параграф 4.4.2.2.5.2) за исключением поддержки опций **Use TOS value as routing key** (см. параграф 4.4.2.2.5.2.3) и **Use FWMARK value as routing key** (параграф 4.4.2.2.5.2.1.1).

В настоящее время функции быстрой пересылки пакетов поддерживают немногочисленные устройства (драйвер tulip и модифицированный драйвер 8390, который можно загрузить с сайта ftp://ftp.tux.org/pub/net/ip-routing/fastroute/)[3].

Если вы не уверены в необходимости поддержки быстрой пересылки пакетов, выберите значение **N**.

4.4.2.2.32 NET_HW_FLOWCONTROL

Опция **Forwarding between high speed interfaces** разрешает аппаратное управление сетевым адаптером в периоды максимальной загрузки. В настоящее время такие функции поддерживают немногочисленные устройства (реально только tulip, для которого можно загрузить модифицированный драйвер 8390 с сайта ftp://ftp.tux.org/pub/net/ip-routing/fastroute/)[4].

На практике эту опцию можно использовать для любых машин, подключенных к достаточно быстрым сетям, поскольку даже 10-мегабитный интерфейс способен существенно затормозить работу не самого медленного ПК (скажем, Pentium 120 МГц).

1 *Исходные тексты wanpipe вместе с документацией вы найдете на приложенном к книге компакт-диске (каталог SRC/wanpipe/).*
2 *В ядре 2.6.8 эта опция была исключена.*
3 *Копии этих файлов вы можете найти в каталоге SRC/ip-routing/ приложенного к книге компакт-диска.*
4 *См. предыдущую сноску.*

Если вы не сталкиваетесь с реальными проблемами перегрузки компьютера обработкой сетевого трафика, не используйте для этой опции значения Y.

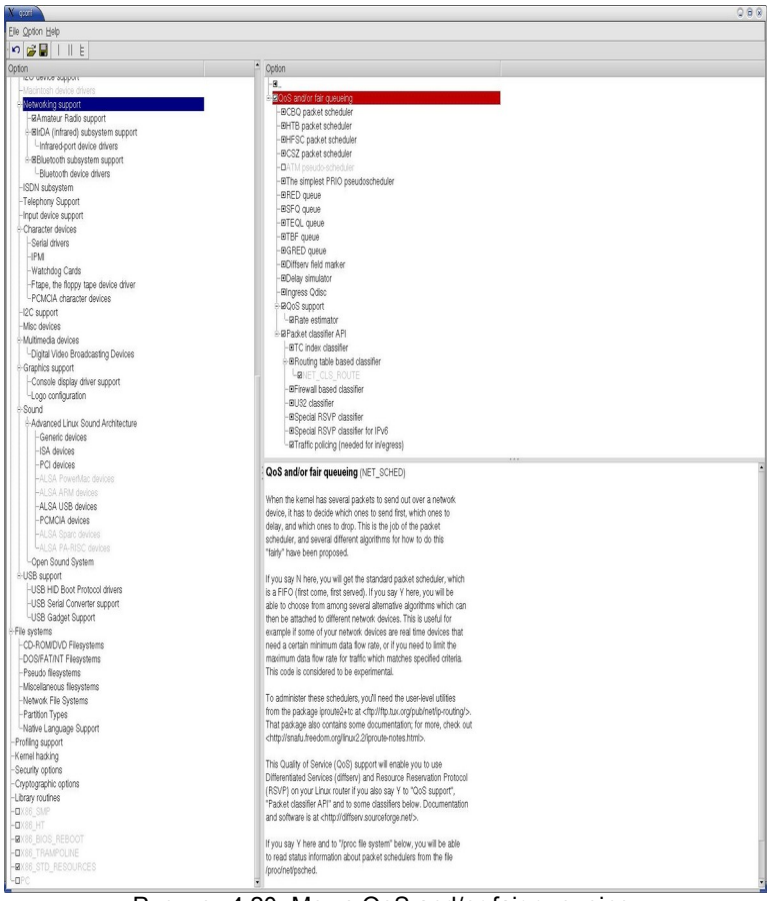


Рисунок 4.20. Меню QoS and/or fair queueing.

4.4.2.2.33 Меню QoS

4.4.2.2.33.1 NET_SCHED

Опция **QoS and/or fair queueing** позволяет управлять функциями контроля очередей пакетов.

Когда ядро имеет несколько пакетов для передачи в сетевые устройства, требуется принять решение об очередности передачи и (в некоторых случаях) о возможности отбрасывания пакета. Эту работу выполняет планировщик очередей, использующий для управления несколько алгоритмов.

Если вы выберете для этой опции значение **N**, будет использоваться только стандартный планировщик, работающий на основе алгоритма **FIFO** (первым пришел, первого обслужили). При выборе значения **Y** вы сможете воспользоваться несколькими дополнительными алгоритмами, которые могут работать с различными сетевыми устройствами. Поддержка данной опции полезна в тех случаях, когда вы используете сетевые устройства, работающие в реальном масштабе времени, или требуется распределять ограниченную полосу внешнего канала в соответствии с теми или иными критериями.

Для управления планировщиками очередей вам потребуются программы из пакета iproute2+tc[1]. Информацию о пакете можно найти на сайте http://snafu.freedom.org/linux2.2/iproute-notes.html.

Эта опция позволит вам использовать протоколы **diffserv**[2] и **RSVP**[3] на вашем Linux-маршрутизаторе, если вы включите также описанные ниже опции **QoS support** и **Packet classifier API**. Документацию и программы для работы вы сможете найти на сайте http://diffserv.sourceforge.net/.

Если вы включите при компиляции ядра описанную на стр. 67 опцию **/proc file system**, вы сможете получать информацию о работе планировщика пакетов из файла **/proc/net/psched**.

Доступные планировщики пакетов рассматриваются ниже.

4.4.2.2.33.2 Меню Packet scheduler clock source

Планировщикам пакетов требуется точный таймер, значения которого растут с постоянной скоростью и фиксированным шагом. Ядро Linux поддерживает несколько таких таймеров, отличающихся по свойствам:

- высокое разрешение (не хуже 1 мксек);
- быстрый доступ (минимальная блокировка, отсутствие операций ввода-вывода);
- синхронизация всех процессов;
- учет изменения тактовой частоты процессора.

Однако ни один из таймеров не обладает всем набором перечисленных свойств, поэтому вам следует выбрать наиболее подходящий для ваших задач таймер из числа описанных ниже.

4.4.2.2.33.2.1 NET_SCH_CLK_JIFFIES

Опция **Timer interrupt** (**YN**) управляет использованием таймера прерываний (jiffy) в качестве источника синхронизации. Этот таймер является быстрым, способен синхронизировать все процессы и учитывать изменение тактовой частоты процессора, но его разрешение недостаточно высоко и не обеспечивает требуемой точности в большинстве приложений.

1 *Этот пакет можно загрузить с сайта ftp://ftp.tux.org/pub/net/ip-routing/. Исходные тексты и документацию к этому пакету вы сможете также найти в приложенном к курсу компакт-диске (каталог SRC/ip-routing/*
2 *Differentiated Services - протокол, описанный в документах RFC 2474 и RFC 2475, которые вы можете загрузить с сайта http://rfc-editor.org/rfc/.*
3 *Resource Reservation Protocol - протокол резервирования ресурсов, описанный в RFC 2205. Этот документ вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc2205.txt.*

4.4.2.2.33.2.2 NET_SCH_CLK_GETTIMEOFDAY

Опция **gettimeofday** (**YN**) управляет использованием в качестве источника синхронизации функции **gettimeofday**. Этот таймер имеет высокое разрешение, может синхронизировать все процессы и учитывать изменение тактовой частоты процессора, но не обладает достаточной скоростью.

Выбирайте эту опцию в тех случаях, когда вам требуется таймер с высоким разрешением, не использующий счетчик циклов CPU.

4.4.2.2.33.2.3 NET_SCH_CLK_CPU

Опция **CPU cycle counter** (**YN**) управляет использованием в качестве источника синхронизации счетчик циклов процессора. Этот таймер обеспечивает высокое разрешение, но подвержен влиянию изменения тактовой частоты и в некоторых системах не может использоваться для синхронизации всех процессов.

4.4.2.2.33.2.4 NET_SCH_CBQ

Опция **CBQ packet scheduler** (**YNM**) позволяет использовать алгоритм управления очередями на базе классов (Class-Based Queueing или CBQ) для некоторых сетевых устройств. Этот алгоритм делит ожидающие отправки пакеты по классам, распределенным в древовидной структуре. Для каждой ветви дерева используется свой алгоритм управления (дисциплина в контексте алгоритма CBQ) .

Алгоритм CBQ является одним из наиболее распространенных, поэтому целесообразно выбрать для этой опции значение **Y**. В этом случае следует также выбрать значение **Y** для всех алгоритмов, которые вы планируете использовать в качестве дисциплин CBQ. Установите также значение **Y** для опции **Packet classifier API** (см. стр. 90) и всех классификаторов[1], которые вы хотите использовать.

Вы можете также использовать для этой опции значение **M**, в результате чего будет создан загружаемый модуль **sch_cbq**.

4.4.2.2.33.2.5 NET_SCH_HTB

Опция **HTB packet scheduler** (**YNM**) управляет использованием алгоритма управления очередями пакетов **HTB**[2] для некоторых сетевых устройств.

Алгоритм HTB очень похож по решаемым задачам на CBQ, однако он имеет иные свойства и пользуется другими методами управления очередями.

При выборе для опции значения **M**, алгоритм **HTB** будет реализован в загружаемом модуле **sch_htb**.

4.4.2.2.33.2.6 NET_SCH_HFSC

Опция **HFSC packet scheduler** (**YNM**) управляет возможностью использования алгоритма **HFSC** (Hierarchical Fair Service Curve) для некоторых сетевых устройств.

При выборе для этой опции значения **M** алгоритм будет реализован в загружаемом модуле s**ch_hfsc**.

4.4.2.2.33.2.7 NET_SCH_CSZ

Опция **CSZ packet scheduler**[3] (**YNM**) управляет возможностью использования экспериментального алгоритма **CSZ**[4] для некоторых сетевых устройств. В настоящий момент это единственный алгоритм, способный обеспечить гарантированное обслуживание для приложений в реальном масштабе времени.

При выборе значения **M** алгоритм будет реализован в загружаемом модуле **sch_csz**.

4.4.2.2.33.2.8 NET_SCH_ATM

Опция **ATM pseudo-scheduler** (**YNM**) управляет возможностью использования псевдо-планировщика ATM, обеспечивающего классификаторы (фильтры) для распределения пакетов по классам. Каждый класс отображает обслуживаемые им потоки а виртуальное устройство (**VC**).

При выборе для опции значения **M** алгоритм будет реализован в загружаемом модуле **sch_atm**.

4.4.2.2.33.2.9 NET_SCH_PRIO

Опция **The simplest PRIO pseudoscheduler** (**YNM**) определяет возможность использования псевдо-планировщика n-band priority для некоторых сетевых устройств или в качестве дисциплины алгоритма **CBQ** (параграф 4.4.2.2.33.2.4).

При выборе значения **M** алгоритм будет реализован в загружаемом модуле **sch_prio**.

1 *Классификаторами называют программы, распределяющие исходящие пакеты по классам на основе тех или иных критериев.*
2 *Hierarchical Token Buckets. Детальную информацию об этом алгоритме можно найти на сайте http://luxik.cdi.cz/~devik/qos/htb/.*
3 *Поддержка этого алгоритма была исключена из ядра 2.6.8 в силу отсутствия законченного варианта алгоритма.*
4 *Сокращение от фамилий авторов - Clark-Shenker-Zhang*

4.4.2.2.33.2.10 NET_SCH_RED

Опция **RED queue** (**YNM**) управляет использованием алгоритма управления очередями пакетов **RED**[1] для некоторых сетевых устройств.

При выборе для опции значения **M** алгоритм будет реализован в загружаемом модуле **sch_red**.

4.4.2.2.33.2.11 NET_SCH_SFQ

Опция **SFQ queue** (**YNM**) управляет возможностью использования алгоритма **SFQ**[2] для некоторых сетевых устройств или в качестве дисциплины алгоритма **CBQ** (параграф 4.4.2.2.33.2.4).

При выборе значения **M** алгоритм будет реализован в загружаемом модуле **sch_sfq**.

4.4.2.2.33.2.12 NET_SCH_TEQL

Опция **TEQL queue** (**YNM**) управляет использованием алгоритма **TLE**[3] для некоторых сетевых устройств или в качестве дисциплины алгоритма **CBQ** (параграф 4.4.2.2.33.2.4). Этот алгоритм позволяет группировать несколько физических устройств в одно виртуальное устройство.

При выборе значения **M** алгоритм реализуется в загружаемом модуле **sch_teql**.

4.4.2.2.33.2.13 NET_SCH_TBF

Опция **TBF queue** (**YNM**) управляет использованием алгоритма **TBF**[4] для некоторых сетевых устройств или в качестве дисциплины алгоритма **CBQ** (параграф 4.4.2.2.33.2.4).

При выборе значения **M** алгоритм будет реализован в загружаемом модуле **sch_tbf**.

4.4.2.2.33.2.14 NET_SCH_GRED

Опция **GRED queue** (YNM) управляет использованием алгоритма Generic Random Early Detection (RED) для некоторых сетевых устройств.

При выборе значения **M** алгоритм будет реализован в загружаемом модуле **sch_gred**.

4.4.2.2.33.2.15 NET_SCH_DSMARK

Опция **Diffserv field marker** (**YNM**) определяет возможность управления очередями в соответствии с архитектурой Differentiated Services (diffserv), предложенной в RFC 2475. Техническую информацию об этом методе и ссылки на связанные с ним документы RFC вы сможете найти на сайте http://www.gta.ufrj.br/diffserv/.

При выборе для этой опции значения **M** алгоритм будет реализован в загружаемом модуле **sch_dsmark**.

4.4.2.2.33.2.16 NET_SCH_NETEM

Опция **Network emulator** (**YNM**) позволяет эмулировать, задержки, потерю и изменение порядка доставки пакетов, присущие передаче через реальные сети. Эта опция может оказаться весьма полезной при тестировании приложений или протоколов.

При выборе значения **M** функции эмуляции передачи пакетов через сеть будут реализованы в загружаемом модуле **sch_netem**.

Эта опция поддерживается, начиная с версии ядра 2.6.8. В более ранних версиях вместо нее использовалась опция **Delay simulator** (**YNM**), управлявшая возможностью задерживать пакеты на фиксированное время (например, для имитации задержек в сети или тестирования приложений и протоколов.

4.4.2.2.33.2.17 NET_SCH_INGRESS

Опция **Ingress Qdisc** (**YNM**) управляет возможностью использования политики (правил) для распределения полосы входящего канала и отбрасывания пакетов, выходящих за пределы разрешенной полосы.

При выборе значения **M** функции контроля входящего трафика будут реализованы в загружаемом модуле **sch_ingress**.

4.4.2.2.33.2.18 NET_QOS

Опция **QoS support** определяет поддержку функций планирования отправки пакетов QoS, способных задавать ограничения уровня трафика для некоторых сетевых устройств.

Эта опция позволяет вам пользоваться протоколами **diffserv**[5] и **RSVP**[6] на вашем Linux-маршрутизаторе, если вы выберете также значение **Y** для опция Packet classifier API (параграф 4.4.2.2.33.2.19 на стр. 90) и некоторые из

1 *Random Early Detection*
2 *Stochastic Fairness Queueing*
3 *True Link Equalizer*
4 *Simple Token Bucket Filter*
5 *Протокол Differentiated Services, описанный в документах RFC 2474 и RFC 2475, который вы загрузить с сайта http://rfc-editor.org/rfc/.*
6 *Resource Reservation Protocol - протокол резервирования ресурсов, описанный в RFC 2205. Этот документ вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc2205.txt.*

описанных ниже опций поддержки классификаторов пакетов. Информацию об использовании этих возможностей вы найдете на сайте http://diffserv.sourceforge.net/.

Отметим, что выбор значения этой опция не влияет непосредственно на ядро, ее значение лишь определяет доступность для выбора описанных ниже опций QoS.

4.4.2.2.33.2.18.1 NET_ESTIMATOR

Опция **Rate estimator** определяет возможность оценки ядром текущего уровня трафика для сетевого устройства, Эта оценка будет использоваться при планировании работы системы QoS.

4.4.2.2.33.2.19 NET_CLS

Опция **Packet classifier API** определяет возможность использования перечисленных ниже опций для классификаторов, работающих с алгоритмом **CBQ** (параграф 4.4.2.2.33.2.4).

Эта опция позволяет вам пользоваться протоколами **diffserv** и **RSVP** на вашем Linux-маршрутизаторе. Информацию об использовании этих возможностей вы найдете на сайте http://diffserv.sourceforge.net/.

4.4.2.2.33.2.19.1 NET_CLS_TCINDEX

Опция **TC index classifier** (**YNM**) определяет возможность классификации исходящих пакетов в соответствии со значением поля **tc_index**[1] в сетевом буфере Linux (**skb**)[2]. Эта опция потребуется вам для использования возможностей протокола **diffserv**.

При выборе значения **M** классификатор будет реализован в модуле **cls_tcindex**.

4.4.2.2.33.2.19.2 NET_CLS_ROUTE4

Опция **Routing table based classifier** (**YNM**) позволяет классифицировать исходящие пакеты в соответствии с записями таблицы маршрутизации.

При выборе для опции значения **M** классификатор будет реализован в модуле **cls_route**.

4.4.2.2.33.2.19.2.1 NET_CLS_ROUTE

Значение этой опции полностью определяется выбором предыдущей опции.

4.4.2.2.34 NET_CLS_FW

Опция **Firewall based classifier** (**YNM**) позволяет классифицировать исходящие пакеты в соответствии с заданными критериями брандмауэра.

При выборе значения **M** классификатор реализуется в виде загружаемого модуля **cls_fw**.

4.4.2.2.35 NET_CLS_U32

Опция **U32 classifier** (**YNM**) позволяет классифицировать исходящие пакеты по адресу получателя.

При выборе значения **M** классификатор будет реализован в виде загружаемого модуля **cls_u32**.

4.4.2.2.35.1 CLS_U32_PERF

Опция **U32 classifier performance counters** (**YN**) управляет сбором статистики, которая может быть использована для настройки работы классификатора **u32**.

Для использования этой опции требуется новая версия пакета **iproute2**, доступная на сайте ftp://ftp.tux.org/people/alexey-kuznetsov/ip-routing/. Не включайте опцию, если у вас нет этого пакета.

4.4.2.2.36 NET_CLS_RSVP

Опция **Special RSVP classifier** (**YNM**) позволяет классифицировать пакеты на основе запросов **RSVP**[3]. Протокол **RSVP** позволяет конечным системам запрашивать минимальную и максимальную скорость для соединений. Такая возможность играет важную роль для передачи информации в реальном масштабе времени (аудиовизуальные потоки).

При выборе значения **M** классификатор реализуется в виде загружаемого модуля **cls_rsvp**.

4.4.2.2.37 NET_CLS_RSVP6

Опция **Special RSVP classifier for IPv6** (**YNM**) позволяет классифицировать пакеты IPv6 на основе запросов **RSVP**. Протокол **RSVP** позволяет конечным системам запрашивать минимальную и максимальную скорость для соединений. Такая возможность играет важную роль для передачи информации в реальном масштабе времени (аудиовизуальные потоки).

1 *Индекс управления трафиком (см. параграф 12.16.1.1 на стр. 403).*
2 *Информацию о структуре и функциях skb вы можете найти в Приложении 12.16.*
3 *Resource Reservation Protocol - протокол резервирования ресурсов, описанный в RFC 2205. Этот документ вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc2205.txt.*

При выборе значения **M** классификатор реализуется в виде загружаемого модуля **cls_rsvp6**.

4.4.2.2.38 NET_CLS_ACT

Опция **Packet ACTION** (**YN**) включает поддержку расширения **tc**, которое может использоваться для классификации пакетов. В настоящий момент поддерживаются только классификаторы **u32** (параграф 4.4.2.2.35 на стр. 90) и **fw** (параграф 4.4.2.2.34 на стр. 90).

Для использования этой опции требуется обновленный пакет **iproute2**, доступный на сайте ftp://ftp.tux.org/people/alexey-kuznetsov/ip-routing/. Не включайте опцию, если у вас нет этого пакета.

4.4.2.2.38.1 NET_ACT_POLICE

Опция **Policing Actions** (**YNM**) используется для управления трафиком совместно с новой версией **iproute2**. В остальных случаях для управления трафиком следует использовать опцию **NET_CLS_POLICE**.

При выборе значения **M** будет создан загружаемый модуль **police**.

4.4.2.2.39 NET_CLS_POLICE

Опция **Traffic policing (needed for in/egress)** позволяет управлять трафиком (ограничивать полосу) для входящих (ingress) и исходящих (egress) потоков. При использовании последней версии пакета iproute2 вы можете выбрать опцию **NET_ACT_POLICE**, описанную выше.

При выборе значения **M** будет создан загружаемый модуль **police**.

4.4.2.2.40 Меню Network testing

4.4.2.2.40.1 NET_PKTGEN

Опция **Packet Generator** (YNM) управляет возможностью использования хоста Linux в качестве генератора пакетов. Функция обеспечивает генерацию и передачу через выбранный интерфейс пакетов с заданными параметрами и частотой. Данную опцию можно применять в целях тестирования систем, но использовать ее следует с осторожностью.

Краткое описание работы с генератором пакетов вы найдете в параграфе 11.11.1 (стр. 326).

При выборе значения **M** генератор пакетов будет реализован в виде загружаемого модуля **pktgen**.

Работа со встроенным генератором пакетов описана в параграфе 11.11.1 (стр. 326).

4.4.2.3 NETPOLL

Группа опций **NETPOLL** управляет использованием сетевых устройств в критических ситуациях на основе опроса на аппаратном уровне. Режим **NETPOLL** используется в настоящее время для передачи сообщений ядра через сеть, поэтому опция автоматически активизируется при включении опции **Network console logging support** (параграф на стр. 96).

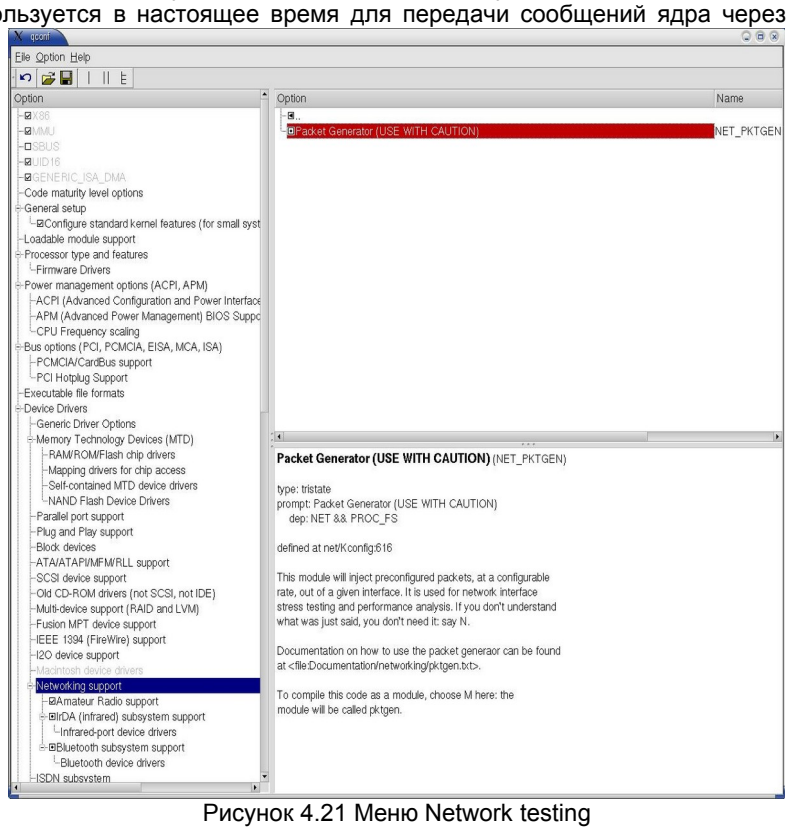


Рисунок 4.21 Меню Network testing

4.4.2.4 NETPOLL_RX

Опция **Netpoll support for trapping incoming packets** (**YN**) управляет возможностью захвата входящих пакетов с помощью функций **NETPOLL**.

4.4.2.5 NETPOLL_TRAP

Опция **Netpoll traffic trapping** (**YN**) управляет возможностью захвата трафика с помощью функций **NETPOLL**.

4.4.2.6 NET_POLL_CONTROLLER

Состояние этой опции определяется состоянием **NETPOLL**.

4.4.2.7 Меню Network device support

Опции меню Network device support управляют поддержкой драйверов сетевого оборудования. Мы не будем здесь останавливаться на этом вопросе, поскольку список поддерживаемых ядром Linux устройств слишком обширен и рассмотрение всех доступных опций потребует очень много времени. Разделы этого меню можно увидеть на рисунке 4.22, мы же остановимся лишь на некоторых опциях, имеющих важное значение с

точки зрения безопасности и работы шлюзов Linux.

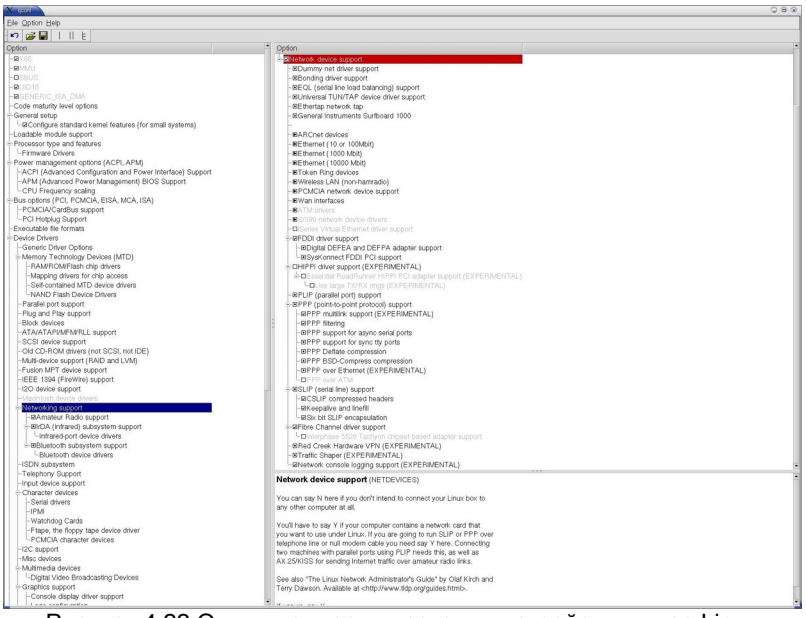


Рисунок 4.22 Опции поддержки сетевых устройств в ядре Linux

4.4.2.7.1 NETDEVICES

Опция **Network device support** определяет возможность подключения Linux-машины к какой-либо сети с помощью какого-либо из существующих сетевых интерфейсов. Выберите значение **Y**, если вы планируете соединять этот компьютер с сетью с помощью сетевого адаптера, модема (по протоколу SLIP или PPP) или даже нуль-модемного кабеля, а также через какой-либо из имеющихся в компьютере портов, способных поддерживать сетевые протоколы. Практически все современные варианты использования компьютера требуют поддержки сетевых служб и выбора для этой опции значения **Y**. Информацию по сетевым возможностям Linux вы сможете найти в документе "**The Linux Network Administrator's Guide**", написанном Олафом Кирчем (Olaf Kirch) и Томом Доусоном (Terry Dawson)[1].

4.4.2.7.1.1 DUMMY

Опция **Dummy net driver support** (**YNM**) управляет поддержкой dummy-драйвера с задаваемым пользователем адресом IP. Dummy-устройство в какой-то степени аналогично устройству /dev/null, т. е., переданные в это устройство пакеты улетают в "черную дыру". Такое псевдо-устройство полезно для компьютеров, использующих коммутируемые соединения (например, SLIP или PPP). При отсутствии реального соединения с сетью можно указать взамен адреса реального устройства адрес dummy-устройства и программы будут работать с ним как с любым другим сетевым интерфейсом. Информацию о dummy-устройстве вы можете найти в **Network Administrator's Guide**.

При выборе для этой опции значения **M** драйвер будет создан в виде загружаемого модуля **dummy**. Это значение опции следует выбирать в тех случаях, когда вам может потребоваться более одного dummy-устройства (они будут именоваться dummy0, dummy1 и т. д.).

4.4.2.7.1.2 BONDING

Опция **Bonding driver support** (**YNM**) управляет возможностью объединения двух каналов Ethernet в один канал с большей пропускной способностью[2].

Для использования этого режима на другой стороне соединения Ethernet должно использоваться устройство, поддерживающее Bonding-драйвер Linux, коммутатор Cisco 5500 или устройство, поддерживающее драйвер SunSoft SunTrunking.

Этот драйвер подобен драйверу EQL (см. ниже), используемому для объединения последовательных каналов.

При выборе для опции значения **M** драйвер будет скомпилирован в форме загружаемого модуля **bonding**.

4.4.2.7.1.3 EQUALIZER

Опция **EQL (serial line load balancing) support** (YNM) управляет возможностью объединения двух последовательных каналов в одно логическое соединение с большей полосой.

При использовании модемных соединений на основе протокола SLIP или PPP вы можете сгруппировать два канала в один. На другой стороне соединения должно использоваться устройство, поддерживающее аналогичный драйвер EQL или Livingston Portmaster 2e.

Информацию об использовании драйвера EQL вы сможете найти в файле **Documentation/networking/eql.txt** дистрибутива ядра Linux или в разделе 6.2 документа NET-3-HOWTO, который можно загрузить с сайта http://www.tldp.org/docs.html#howto.

При выборе для опции значения **M** драйвер будет реализован в виде загружаемого модуля **eql**.

4.4.2.7.1.4 TUN

Опция **Universal TUN/TAP device driver support** (**YNM**) обеспечивает возможность приема и передачи пакетов от программ пользовательского пространства (не входящих в состав ядра). В системе создается псевдо-устройство (оно может выглядеть как обычное устройство PPP или Ethernet), которое обменивается пакетами не с физической средой, а с пользовательской программой.

1 *Этот документ можно загрузить с сайта http://www.tldp.org/guides.html. Копия документа имеется также в каталоге Documents/ приложенного к книге компакт-диска.*
2 *Такую функцию называют Etherchannel применительно к оборудованию Cisco, Trunking - применительно к продукции Sun и Bonding в контексте Linux*

Когда программа открывает файл **/dev/net/tun**, драйвер создает и регистрирует в системе соответствующее устройство **tunX** или **tapX**. После закрытия файл драйвер автоматически удаляет созданное устройство и все связанные с ним маршруты.

Дополнительную информацию об использовании псевдо-устройств вы сможете найти в файле **Documentation/networking/tuntap.txt** дистрибутива ядра Linux.

При выборе значения **M** драйвер создается в форме загружаемого модуля **tun**.

4.4.2.7.1.5 ETHERTAP

Опция **Ethertap network tap** (**YNM**) позволяет программам из пользовательского пространства записывать кадры Ethernet в специальный файл **/dev/tap0** и читать кадры из этого файла. Для использования таких функций потребуется также включить опцию **Netlink device emulation** (см. параграф 4.4.2.2.2 на стр. 69) и создать специальный файл **/dev/tap0** со старшим номером версии 36 и младшим 16, используя команду **mknod**. Псевдоустройство **tap0** можно настраивать с помощью команд **ifconfig** (параграф на стр.) и **route** (параграф на стр.) подобно другим устройствам Ethernet, но оно не связано физически ни с какой ЛВС. Вся информация, записанная пользователем (программой) в файл **/dev/tap0**, трактуется ядром как данные, принятые устройством **tap0** из локальной сети, а все данные, которые ядро передаст в устройство **tap0**, будут просто записаны в файл **dev/tap0**. Такие функции могут оказать большую пользу при тестировании приложений или настройке конфигурации сети. Информацию о драйвере вы сможете найти в файле **Documentation/networking/ethertap.txt** дистрибутива ядра Linux.

При выборе для опции значения **M** драйвер будет реализован как загружаемый модуль **ethertap**.

4.4.2.7.1.6 Меню драйверов устройств

Меню управления опциями поддержки ядром драйверов устройств включает множество опций управления использованием драйверов физических устройств различных типов, включая:

- ARCnet
- Ethernet
- Token Ring
- Wireless LAN
- PCMCIA
- WAN
- ATM
- S390
- FDDI
- HIPPI
- PLIP (параллельный порт)

Рассмотрение отдельных драйверов выходит за рамки этой книги и мы не будем останавливаться на этом вопросе. Входящая в дистрибутив ядра документация содержит достаточно много сведений о поддерживаемых сетевых устройствах и ссылки на источники дополнительной информации. Ограничимся рассмотрением некоторых опций работы с интерфейсами WAN, PPP и SLIP, поскольку они имеют достаточно важное значение в контексте работы с сетевыми шлюзами.

Отметим также, что если в вашем компьютере не планируется использование того или иного типа сетевых интерфейсов, целесообразно отключить поддержку всего класса устройств, воспользовавшись для этого первой опцией соответствующего меню.

4.4.2.7.1.7 Меню Wan interfaces

Это меню (рисунок 4.23) служит для выбора опций поддержки распределенных сетей (WAN) таких как, X.25, Frame Relay, модемные соединения по выделенным линиям и т. п.

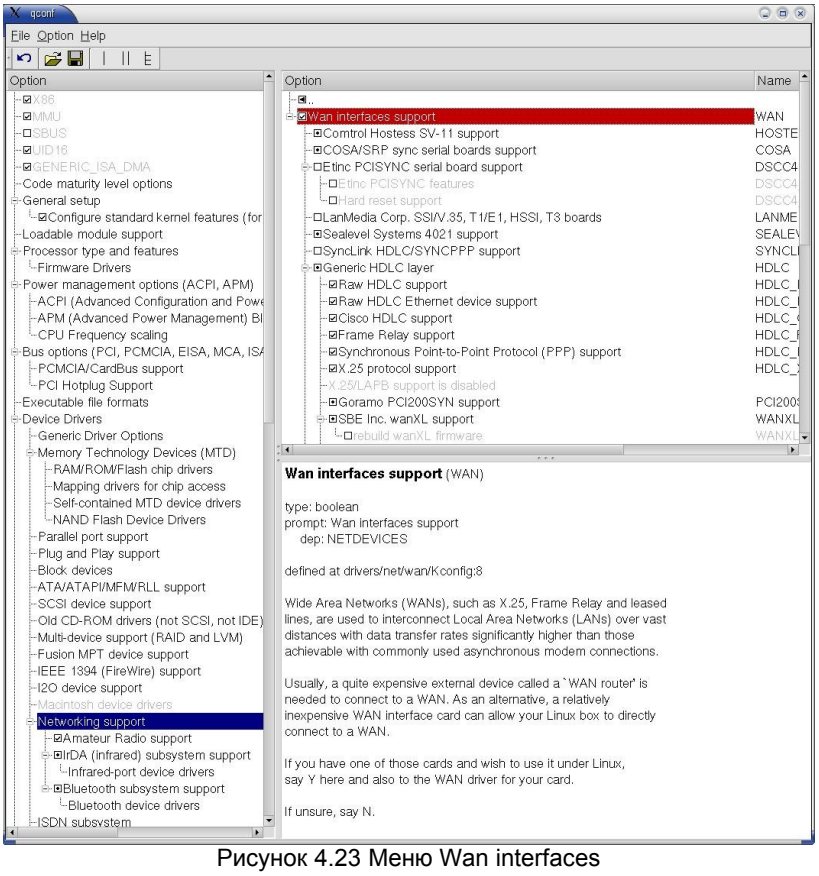


Рисунок 4.23 Меню Wan interfaces

Хосты Linux способны выполнять функции WAN-маршрутизаторов, обеспечивая недорогую и эффективную

альтернативу имеющимся на рынке “коробочным” маршрутизаторам.

Если в вашем компьютере установлен какой-либо из WAN-интерфейсов, включите эту опцию и выберите в списке драйвер соответствующего устройства.

Для поддержки функций маршрутизации требуется также включить опцию **WAN router** (параграф 4.4.2.2.30).

Мы не будем рассматривать опции драйверов устройств WAN, ограничившись лишь общими вопросами поддержки протоколов.

4.4.2.7.1.7.1 HDLC

Опция **Generic HDLC layer** (**YNM**) управляет поддержкой драйвера HDLC для устройств **raw HDLC**, **Cisco HDLC**, **Frame Relay**, **X.25** и синхронных соединений PPP.

Драйверы устройств, работающих с использованием протокола HDLC можно загрузить с сайта http://hq.pm.waw.pl/hdlc/

При выборе для опции значения M драйвер будет создан в виде загружаемого модуля **hdlc**.

4.4.2.7.1.7.1.1 HDLC_RAW

Опция **Raw HDLC support** управляет поддержкой **raw HDLC** для соединений WAN.

4.4.2.7.1.7.1.2 HDLC_RAW_ETH

Опция **Raw HDLC Ethernet device support** управляет поддержкой эмуляции **raw HDLC Ethernet** через каналы WAN.

4.4.2.7.1.7.1.3 HDLC_CISCO

Опция **Cisco HDLC support** управляет поддержкой протокола **Cisco HDLC** для соединений WAN.

4.4.2.7.1.7.1.4 HDLC_FR

Опция **Frame Relay support** управляет поддержкой протокола **HDLC** для соединений Frame Relay.

4.4.2.7.1.7.1.5 HDLC_PPP

Опция **Synchronous Point-to-Point Protocol (PPP) support** управляет поддержкой протокола **HDLC** для синхронных каналов PPP.

4.4.2.7.1.7.1.6 HDLC_X25

Опция **X.25 protocol support** управляет поддержкой протокола **HDLC** для соединений X.25.

4.4.2.7.1.7.2 DLCI

Опция **Frame Relay DLCI support** (**YNM**) управляет поддержкой идентификаторов **DLCI** для устройств Frame Relay[1]. Сети Frame Relay позволяют организовать несколько логических соединений по одному физическому каналу. Для адресации этих соединений используются специальные идентификаторы **DLCI**[2].

Для подключения к сетям Frame Relay вам потребуется устройство доступ (их обычно называют термином FRAD[3]). Дополнительные сведения об использовании устройств Frame Relay и требуемых для этого программах вы найдете в файле **Documentation/networking/framerelay.txt** из дистрибутива ядра.

Если для опции было выбрано значение **M**, драйвер будет скомпилирован в виде загружаемого модуля **dlci**.

4.4.2.7.1.7.2.1 DLCI_COUNT

Опция **Max open DLCI** определяет максимальное число открытых соединений **DLCI**, которые могут обслуживаться драйвером устройства Frame Relay. По умолчанию драйвер поддерживает 8 DLCI.

4.4.2.7.1.7.3 LAPBETHER

Экспериментальная опция **LAPB over Ethernet driver** (**YNM**) управляет поддержкой драйвера псевдо-устройства (обычно **/dev/lapb0**), позволяющего создавать соединения LAPB point-to-point через локальную сеть Ethernet.

Для использования этого драйвера вы должны будете также выбрать значение **Y** или **M** для опции **LAPB Data Link Driver** (параграф 4.4.2.2.27 на стр. 85).

Если вы выбрали для опции **LAPBETHER** значение **M** драйвер будет реализован как загружаемый модуль **lapbether**.

1 *Дополнительную информацию о сетях Frame Relay вы найдете на сайте http://www.mplsforum.org/. Информация о протоколе FR имеется также на сайте http://www.protokols.ru.*
2 *Data Link Channel Identifier - идентификатор канального уровня.*
3 *Frame Relay Access Device - устройство доступа Frame Relay.*

4.4.2.7.1.7.4 X25_ASY

Экспериментальная опция **X.25 async driver** (**YNM**) управляет поддержкой драйвера для асинхронных соединений X25. При включенной опции вы сможете передавать пакеты X.25 по обычным асинхронным каналам на базе модемов и телефонных линий. Текущая реализация драйвера еще не соответствует полностью требованиям рекомендаций X.25 для асинхронных каналов.

При выборе для опции значения **M** драйвер будет скомпилирован как загружаемый модуль **x25_asy**.

4.4.2.7.1.8 PPP

Опция **PPP (point-to-point protocol) support** (**YNM**) управляет поддержкой протокола PPP (Point to Point Protocol), используемого для передачи трафика IP по телефонным линиям и иным последовательным каналам.

Для использования протокола PPP вам потребуется демон **pppd**, работа с которым описана в документе PPP-HOWTO[1]. Убедитесь, что вы используете версию демона, рекомендованную в файле **Documentation/Changes** дистрибутива ядра Linux.

Драйвер PPP увеличивает размер ядра приблизительно на 16 Кбайт.

Существуют два варианта использования PPP - один для традиционных асинхронных каналов на базе модемов и телефонных линий, а другой - для синхронных каналов (например, ISDN). Если вы хотите использовать PPP для телефонных соединений, вам также следует выбрать значение **Y** или **M** для опции **PPP support for async serial ports** (параграф 4.4.2.7.1.8.3 на стр. 95). При использовании PPP на синхронных каналах выберите значение **Y** или **M** для опции **Support synchronous PPP** (параграф 4.4.2.7.1.8.4 на стр. 95).

При выборе значения Y для опции **Version information on all symbols**[2] вы не сможете включить драйвер PPP непосредственно в ядро, его придется создать в формате загружаемого модуля (**ppp_generic**). Дополнительную информацию по компиляции драйвера как модуля вы найдете в файле **Documentation/networking/net-modules.txt** дистрибутива ядра Linux.

4.4.2.7.1.8.1 PPP_MULTILINK

Экспериментальная опция **PPP multilink support** управляет возможностью использования протокола PPP multilink (RFC 1990), позволяющего объединять несколько логических или физических линий в одно логическое соединение PPP, обеспечивающее более широкую полосу.

Для использования этой опции требуется поддержка протокола на другой стороне соединения и наличие демона **pppd**, поддерживающего возможность объединения каналов.

4.4.2.7.1.8.2 PPP_FILTER

Опция **PPP filtering** управляет возможностью фильтрации пакетов, проходящих через интерфейсы PPP. Такая фильтрация позволяет контролировать поток трафика и организовывать соединения по запросу.

4.4.2.7.1.8.3 PPP_ASYNC

Опция **PPP support for async serial ports** (**YNM**) управляет поддержкой протокола PPP для асинхронных портов.

При выборе значения **M** драйвер будет скомпилирован в загружаемый модуль **ppp_async**.

4.4.2.7.1.8.4 PPP_SYNC_TTY

Опция **PPP support for sync tty ports** (**YNM**) определяет возможность использования протокола PPP для синхронных (**HDLC**) устройств **tty**.

Драйвер будет создан как загружаемый модуль **ppp_synctty**, если вы выберете для этой опции значение **M**.

4.4.2.7.1.8.5 PPP_DEFLATE

Опция **PPP Deflate compression** (**YNM**) управляет поддержкой для протокола PPP компрессии на основе алгоритма Deflate (такой же алгоритм используется программой **gzip**) для сжатия пакетов PPP до их передачи в линию. На другой стороне также должен поддерживаться алгоритм сжатия Deflate, чтобы вы могли использовать эту опцию. Если же вы включили компрессию, а на другой стороне она не поддерживается, пакеты будут передаваться без сжатия.

При выборе для опции значения **M** функции компрессии будут реализованы в загружаемом модуле **ppp_deflate**.

4.4.2.7.1.8.6 PPP_BSDCOMP

Опция **PPP BSD-Compress compression** (**NM**) управляет поддержкой для протокола PPP сжатия на основе метода BSD-Compress, использующего алгоритм LZW для компрессии пакетов PPP до их передачи в линию. Для использования этой опции требуется поддержка BSD-Compress и на другой стороне соединения. Если вы включили компрессию, а на другой стороне она не поддерживается, пакеты будут передаваться без сжатия.

Отметим, что поддерживаемый предыдущей опцией алгоритм компрессии Deflate является предпочтительным, поскольку он обеспечивает более сильное сжатие и не защищен патентами.

1 *Документ вы можете загрузить с сайта http://www.tldp.org/docs.html#howto.*
2 *В новых версиях ядра эта опция отсутствует и данное требование утратило актуальность.*

Код сжатия по методу BSD всегда компилируется в виде модуля **bsd_comp**.

4.4.2.7.1.8.7 PPPOE

Экспериментальная опция **PPP over Ethernet** (**YNM**) управляет поддержкой протокола PPP для соединений Ethernet.

Для использования этого драйвера потребуется последняя версия демона **pppd**, доступная на сайте ftp://ftp.samba.org/pub/ppp/[1]. Можно также воспользоваться программой RoaringPenguin, доступной на сайте http://www.roaringpenguin.com/pppoe,[2], в комплект которой входят инструкции по использованию драйвера.

При выборе для опции значения M драйвер будет создан как загружаемый модуль **pppoe**.

4.4.2.7.1.8.8 PPPOATM

Опция **PPP over ATM** (**YNM**) управляет поддержкой протокола PPP для соединений ATM.

Используемый драйвером метод инкапсуляции не полностью совместим с разделом 8 документа RFC 2364, поэтому могут возникать проблемы в тех случаях, когда другая сторона потеряет соединение и в одностороннем порядке изменит метод инкапсуляции.

Если вы выбрали для опции значение **M**, драйвер будет создан как загружаемый модуль **pppoatm**.

4.4.2.7.1.9 SLIP

Опция **SLIP (serial line) support** (**YNM**) определяет возможность поддержки протоколов SLIP и CSLIP (compressed SLIP - протокол со сжатием) для связи с Internet-провайдером или доступа к другим UNIX-машинам, а также в тех случаях, когда компьютер предполагается использовать в качестве сервера Slip/CSlip для удаленного доступа пользователей по телефонным линиям.

Протокол SLIP является предшественником PPP и последний обеспечивает более эффективное решение для передачи трафика IP по телефонным линиям.

Протокол SLIP в настоящее время используется достаточно редко и мы не будем подробно останавливаться на опциях драйвера этого протокола. Интересующиеся могут почерпнуть достаточную информацию из документа NET-3-HOWTO, доступного на сайте http://www.tldp.org/docs.html#howto.

При выборе значения M для этой опции драйвер будет реализован в виде загружаемого модуля **slip**.

4.4.2.7.1.10 SHAPER

Экспериментальная опция **Traffic Shaper** (**YNM**) управляет возможностью использования средств управления трафиком (traffic shaper). При использовании этой опции создается виртуальное устройство, позволяющее ограничить скорость исходящего потока данных. Виртуальное устройство можно использовать для маршрутизации трафика, который вы хотите ограничить по скорости. Информацию о виртуальном устройстве traffic shaper вы можете найти в файле **Documentation/networking/shaper.txt** дистрибутива ядра Linux.

В качестве альтернативного решения для управления исходящим трафиком может использоваться экспериментальный драйвер алгоритма CBQ, если вы включите опцию **QoS and/or fair queueing** (см. параграф 4.4.2.2.33.1 на стр. 87).

Для настройки и управления устройством формирования трафика служит программа **shapecfg**.

При выборе для опции значения M будет создан загружаемый модуль shaper.

4.4.2.7.1.11 NETCONSOLE

Экспериментальная опция **Network console logging support** (**YNM**) позволяет записывать сообщения ядра в журнальные файлы через сеть. Дополнительную информацию об этой функции вы сможете найти в файле **Documentation/networking/netconsole.txt** дистрибутива ядра Linux.

Выбор этой опции автоматически активизирует опцию **NETPOLL** (параграф 4.4.2.3 на стр. 91).

4.4.3 Опции выбора модели безопасности для ядра

Описанные здесь опции включены в раздел **Security options** и позволяют выбрать модель безопасности при компиляции ядра.

1 *Исходные тексты и документацию для этой программы вы можете найти в каталоге SRC/ приложенного к книге компакт-диска.*
2 *Исходные тексты и документацию для этой программы вы можете найти в каталоге SRC/ приложенного к книге компакт-диска.*

4.4.3.1 SECURITY

При выборе поля **Enable different security models** обеспечивается возможность компиляции ядра с включением различных модулей безопасности, набор которых определяется значениями последующих полей. Если это поле не выбрано, при компиляции ядра будет использоваться принятая по умолчанию модель безопасности, а описанные ниже опции не будут доступны.

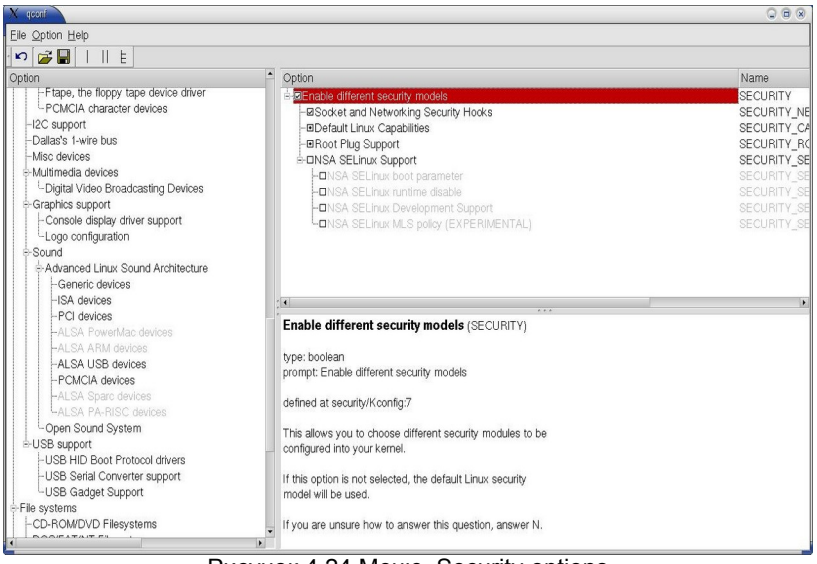


Рисунок 4.24 Меню Security options

4.4.3.1.1 SECURITY_NETWORK

При выборе поля **Socket and Networking Security Hooks** включаются “ловушки безопасности” на уровне сетевых функций и сокетов, позволяющие контролировать доступ к этим функциям и сокетам.

4.4.3.1.2 SECURITY_CAPABILITIES

Поле **Default Linux Capabilities** (**YNM**) определяет использование принятых по умолчанию опций безопасности Linux.

При выборе для опции значения **M** будет создаваться загружаемый модуль **capability**.

4.4.3.1.3 SECURITY_ROOTPLUG

Опция **Root Plug Support** (**YNM**) позволяет управлять запуском программ от имени пользователя **root** с помощью идентификаторов, подключаемых к порту USB.

При выборе для опции значения M функции проверки полномочий пользователя root с помощью ключа USB будут реализованы ы загружаемом модуле **root_plug**.

4.4.3.1.4 Опции поддержки NSA SELinux

http://www.nsa.gov/selinux/

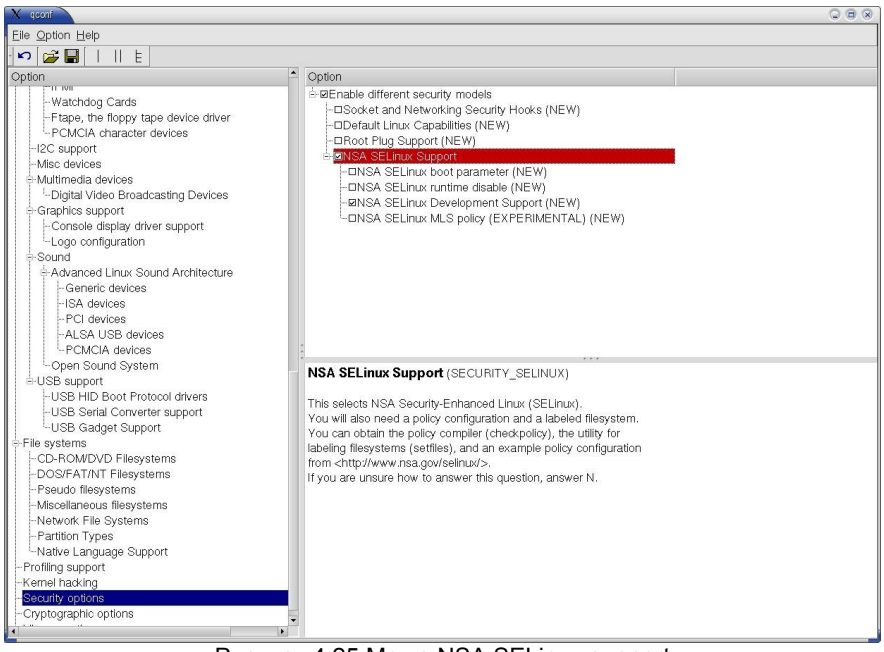


Рисунок 4.25 Меню NSA SELinux support

Описанные здесь опции доступны при выборе описанной выше опции **SECURITY** (параграф 4.4.3.1) и управляют поддержкой дополнительных функций обеспечения безопасности, разработанных в рамках проекта **NSA Security-Enhanced Linux** (параграф 3.7 на стр. 55). Более подробную информацию об этом проекте можно найти, воспользовавшись приведенной выше ссылкой на сайт проекта.

4.4.3.1.4.1 SECURITY_SELINUX

При выборе поля **NSA SELinux Support** включается поддержка функций **NSA SELinux**. Для использования этих функций потребуется также компилятор политики (checkpolicy), утилита для создания меток файловых систем (setfiles) и примеры конфигурационной политики, которые можно загрузить с сайта NSA.

4.4.3.1.4.2 SECURITY_SELINUX_BOOTPARAM

При выборе опции **NSA SELinux boot parameter** обеспечивается возможность отключения функций SELinux при загрузке операционной системы (selinux=0 в командной строке загрузки ядра). Такая возможность позволяет включить функции SELinux в ядро операционной системы и использовать их при возникновении необходимости, указав соответствующую опцию в команде загрузки ядра.

4.4.3.1.4.3 SECURITY_SELINUX_DISABLE

При выборе опции **NSA SELinux runtime disable** обеспечивается возможность запрета использования функций

SELinux при работе операционной системы до загрузки политики. После такого запрета функции SELinux не будут использоваться до следующей загрузки компьютера. Эта опция похожа на использование параметра selinux=0 в команде загрузки ядра, но позволяет отключить использование SELinux в работающей системе (например, с помощью команды **/sbin/init**), что весьма полезно для платформ, в которых сложно управлять параметрами загрузки ядра.

4.4.3.1.4.4 SECURITY_SELINUX_DEVELOP

Выбор опции **NSA SELinux Development Support** позволяет включить поддержку функций разработки в NSA SELinux, что может оказаться весьма полезно при экспериментах с SELinux и разработке политики. При выбранной опции ядро будет загружаться в “либеральном” режиме (permissive mode), разрешающем любые операции и обеспечивающем полную запись информации в журнальные файлы до тех пор, пока в команде загрузки ядра не будет задано enforcing=1. С помощью программы **/selinux/enforce** можно явно переключаться между “либеральным” и жестким режимом (enforcing mode), если это не запрещено реализованной политикой.

4.4.3.1.4.5 SECURITY_SELINUX_MLS

Экспериментальная[1] опция **NSA SELinux MLS policy** позволяет использовать политику NSA SELinux Multi-Level Security (MLS) в дополнение к принятой по умолчанию политике RBAC/TE. Эта политика еще находится в стадии экспериментов, поэтому опцию не следует включать на реально используемых для работы хостах.

4.4.4 Опции поддержки криптографии

Ядро Linux поддерживает множество функций криптографии, перечисленных ниже. Для настройки связанных с криптографией опций служит меню **Cryptographic options** (рисунок 4.26).

Выбор опции **Cryptographic API** (**CRYPTO**) включает поддержку ядром интерфейса функций шифрования **Cryptographic API**.

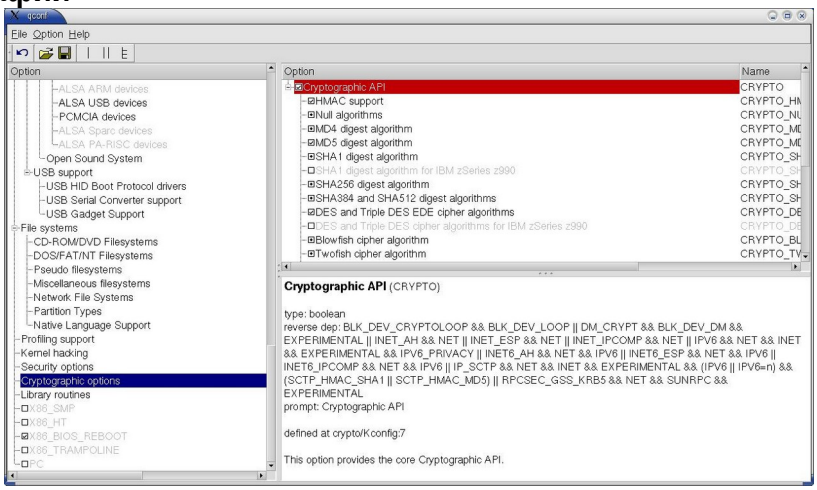


Рисунок 4.26 Меню Cryptographic otions

4.4.4.1 CRYPTO_HMAC

Опция **HMAC support** включает требуемую приложениями IPsec поддержку аутентификации сообщений на основе метода **HMAC**, описанного в RFC 2104.

4.4.4.2 CRYPTO_NULL

Опция **Null algorithms** (**YNM**) управляет поддержкой используемых в приложениях IPsec Null-алгоритмов, которые реально не делают ничего.

При выборе для опции значения M создается загружаемый модуль **crypto_null**.

4.4.4.3 CRYPTO_MD4

Опция **MD4 digest algorithm** (**YNM**) управляет поддержкой цифровых подписей MD4 в соответствии с RFC 1320. Эти функции будут реализованы в модуле **md4**, если для опции выбрано значение **M**.

4.4.4.4 CRYPTO_MD5

Опция **MD5 digest algorithm** (**YNM**) управляет поддержкой цифровых подписей MD5 в соответствии с RFC 1321. Эти функции будут реализованы в модуле **md5**, если для опции выбрано значение **M**.

4.4.4.5 CRYPTO_SHA1

Опция **SHA1 digest algorithm** (**YNM**) управляет поддержкой шифрования в соответствии со стандартом SHA-1 (FIPS 180-1/DFIPS 180-2).

При вы боре для опции значения **M**, функции будут реализованы в загружаемом модуле **sha1**.

4.4.4.6 CRYPTO_SHA256

Опция **SHA256 digest algorithm** (**YNM**) управляет поддержкой алгоритма **SHA256** в соответствии со стандартом DFIPS 180-2. Если для опции выбрано значение **M**, алгоритм будет реализован в загружаемом модуле **sha256**.

4.4.4.7 CRYPTO_SHA512

Опция **SHA512 digest algorithm** (**YNM**) управляет поддержкой алгоритма **SHA384** и **SHA512** в соответствии со стандартом DFIPS 180-2. Если для опции выбрано значение **M**, алгоритм будет реализован в загружаемом модуле **sha512**.

1 *Для ядра 2.6.8*

4.4.4.8 CRYPTO_DES

Экспериментальная опция **DES and Triple DES EDE cipher algorithms** (**YNM**) управляет поддержкой алгоритмов **DES** и **Triple DES EDE** в соответствии со стандартами FIPS 46-2 и FIPS 46-3.

Алгоритмы реализуются в виде модуля **des**, если для опции было выбрано значение **M**.

4.4.4.9 CRYPTO_BLOWFISH

Опция **Blowfish cipher algorithm** (**YNM**) управляет поддержкой алгоритма **Blowfish**, предложенного Брюсом Шнейером (Bruce Schneier). Этот алгоритм позволяет использовать ключи размером от 32 до 448 битов. Дополнительную информацию вы можете найти на сайте http://www.schneier.com/blowfish.html.

При выборе для опции значения M алгоритм будет реализован в загружаемом модуле **blowfish**.

4.4.4.10 CRYPTO_TWOFISH

Опция **Twofish cipher algorithm** (**YNM**) управляет поддержкой алгоритма **Twofish**, работающего с ключами размером 128, 192 и 256 битов. Дополнительную информацию об этом алгоритме вы найдете на сайте http://www.schneier.com/twofish.html.

Алгоритм реализуется как загружаемый модуль **twofish**, если для опции было выбрано значение **M**.

4.4.4.11 CRYPTO_SERPENT

Опция **Serpent cipher algorithm** (**YNM**) управляет поддержкой алгоритма **Serpent**, авторами которого являются Anderson, Biham и Knudsen. Этот алгоритм работает с ключами размером от 0 до 256 битов (кратные 8 значения). Дополнительная информация об этом алгоритме доступна на сайте http://www.cl.cam.ac.uk/~rja14/serpent.html.

При выборе значения **M** алгоритм реализуется в виде загружаемого модуля **serpent**.

4.4.4.12 CRYPTO_AES_586

Опция **AES cipher algorithms (i586)** (**YNM**) управляет поддержкой алгоритмов **AES** (FIPS-197), построенных на базе алгоритма Rijndael. Алгоритм Rijndael хорошо реализуется на программном и аппаратном уровне и может работать с ключами размером 128, 192 и 256 битов. Дополнительную информацию об этом алгоритме вы найдете на сайте http://csrc.nist.gov/encryption/aes/.

Если для опции было выбрано значение **M**, алгоритм будет реализован в загружаемом модуле **aes**.

4.4.4.13 CRYPTO_CAST5

Опция **CAST5 (CAST-128) cipher algorithm** (**YNM**) управляет поддержкой алгоритма **CAST5 (CAST-128)** в соответствии с RFC 2144[1].

Вы можете выбрать для опции значение **M** и алгоритм будет реализован в форме загружаемого модуля **cast5**.

4.4.4.14 CRYPTO_CAST6

Опция **CAST6 (CAST-256) cipher algorithm** (**YNM**) управляет поддержкой алгоритма **CAST6 (CAST-256)** в соответствии с RFC 2612[2].

Вы можете выбрать для опции значение **M** и алгоритм будет реализован в форме загружаемого модуля **cast6**.

4.4.4.15 CRYPTO_TEA

Опция **TEA and XTEA cipher algorithms** (**YNM**) управляет поддержкой алгоритмов **TEA**[3] и **XTEA**[4], обеспечивающих эффективное шифрование при скромном расходе памяти.

Для реализации алгоритма в виде загружаемого модуля **tea** выберите значение опции **M**.

4.4.4.16 CRYPTO_ARC4

Опция **ARC4 cipher algorithm** (**YNM**) определяет поддержку алгоритма **ARC4**, использующего ключи размером от 8 до 2048 битов. Этот алгоритм требуется для реализации драйвера WEP, но его не следует использовать для других приложений по причине недостаточной криптоустойчивости.

Алгоритм будет реализован в модуле **arc4**, если вы выберете значение **M** для данной опции.

4.4.4.17 CRYPTO_KHAZAD

Опция **Khazad cipher algorithm** (**YNM**) управляет поддержкой алгоритма **Khazad**, использующего ключи размером 128 битов. Этот алгоритм оптимизирован для 64-разрядных процессоров, но обеспечивает высокую производительность и на 32-битовых машинах. Дополнительную информацию об этом алгоритме вы можете найти

1 *Копию этого документа вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc2144.txt.*
2 *Копию этого документа вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc2612.txt.*
3 *Tiny Encryption Algorithm - крошечный алгоритм шифрования.*
4 *Расширенный вариант алгоритма TEA.*

на сайте http://planeta.terra.com.br/informatica/paulobarreto/KhazadPage.html.

При выборе для опции значения **M** алгоритм будет реализован в загружаемом модуле **khazad**.

4.4.4.18 CRYPTO_DEFLATE

Опция **Deflate compression algorithm** (**YNM**) управляет поддержкой алгоритма компрессии **Deflate** (RFC 1951[1]), используемого в приложениях IPSec, поддерживающих протокол **IPCOMP** (RFC 3173, RFC 2394[2]).

При выборе для опции значения **M** алгоритм компрессии будет реализован как загружаемый драйвер **deflate**.

4.4.4.19 CRYPTO_MICHAEL_MIC

Опция **Michael MIC keyed digest algorithm** (**YNM**) управляет поддержкой цифровых подписей **Michael MIC**, используемых для проверки целостности сообщений протоколом TKIP (IEEE 802.11i). Не следует использовать этот алгоритм для других приложений, поскольку его криптоустойчивость достаточно мала.

При выборе для опции значения **M** алгоритм будет реализован в виде загружаемого модуля **michael_mic**.

4.4.4.20 CRYPTO_CRC32C

Опция **CRC32c CRC algorithm** (**YNM**) управляет поддержкой контрольных сумм CRC32c. Этот алгоритм используется, в частности, протоколом iSCSI для проверки целостности заголовков и данных. Реализация алгоритма использует библиотеку **/lib/libcrc32c**.

При выборе значения **M** алгоритм будет реализован в загружаемом модуле **crc32c**.

4.4.4.21 CRYPTO_TEST

Опция **Testing module** (**YNM**) управляет поддержкой тестирования криптографических функций и приложений. При выборе для опции значения M функции тестирования будут реализованы в загружаемом модуле **tcrypt**.

4.4.5 Меню Library routines

Опции этого меню не связаны напрямую с безопасностью хоста и сети, но оказывают влияние на работу некоторых функций и модулей, вовлеченных в обеспечение безопасности. Ниже кратко рассмотрены опции меню, показанного на рисунке 4.27.

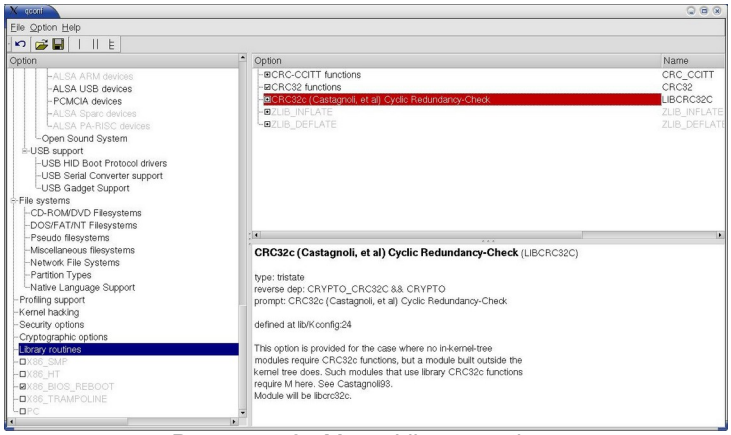


Рисунок 4.27 Меню Library routines

4.4.5.1 CRC_CCITT

Опция **CRC-CCITT functions** (**YNM**) управляет поддержкой контрольных сумм **CRC-CCITT**. Эта опция используются в тех случаях, когда компоненты ядра не используют CRC-CCITT, но эти функции нужны внешним модулям. При использовании функций CRC-CCITT внутри ядра эта опция активизируется автоматически.

Выбрав для этой опции значение **M**, вы получите загружаемый модуль **crc-ccitt**.

4.4.5.2 CRC32

Опция **CRC32 functions** (**YNM**) управляет поддержкой контрольных сумм **CRC32**. Эта опция используется в тех случаях, когда код ядра не требует поддержки CRC32, но эти функции нужны внешним модулям. При использовании функций CRC32 внутри ядра эта опция активизируется автоматически.

Если для опции выбрано значение **M**, функции поддержки контрольных сумм будут реализованы в модуле **crc32**.

4.4.5.3 LIBCRC32C

Опция **CRC32c (Castagnoli, et al) Cyclic Redundancy-Check** (**YNM**) управляет поддержкой контрольных сумм CRC32c. Эта опция используется в тех случаях, когда коду ядра не требуется поддержка **CRC32c**, но эти функции нужны внешним модулям. Если контрольные суммы CRC32c используются внутри ядра, опция активизируется автоматически.

При выборе значение **M** поддержка контрольных сумм реализуется как загружаемый модуль **libcrc32c**.

4.4.5.4 ZLIB_INFLATE

Эта опция управляет поддержкой функций декомпрессии и будет включена автоматически, если она нужна какому-либо из модулей ядра.

1 *Копию этого документа вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc1951.txt.*
2 *Копии этих документов вы можете загрузить с сайта http://rfc-editor.org/rfc/.*

4.4.5.5 ZLIB_DEFLATE

Эта опция управляет поддержкой функций компрессии **Deflate** и будет включена автоматически, если она нужна какому-либо из модулей ядра.

4.5 Компиляция и установка ядра

После завершения настройки конфигурационных параметров ядра введите команду[1]

```
make bzImage
```

и устройтесь поудобнее для наблюдения за процессом компиляции. В зависимости от выбранных опций и параметров вашего компьютера процесс может занять от нескольких минут до нескольких десятков минут. Не пугайтесь обилия выводимых на экран предупреждений - так и должно быть. Если при компиляции возникнет та или иная ошибка, на экран будет выведено соответствующее сообщение и процесс компиляции прекратится.

После завершения компиляции собственно ядра, нужно создать модули. Для этого служит команда

```
make modules
```

Процесс создания модулей обычно занимает больше времени, нежели компиляция самого ядра, поэтому наберитесь терпения - оно будет вознаграждено вашей безопасностью в будущем. По завершении процесса компиляции модулей введите команду

```
make modules_install
```

для установки новых модулей в каталог **/lib**. Далее следует выполнить операции установки самого ядра и создание загрузочной конфигурации. Введите команду

```
make install
```

Возможно, что после установки ядра вам придется внести изменения в файл конфигурации менеджера загрузки (см. параграф 2.1.2 на стр. 33). После завершения работы по настройке конфигурации менеджера загрузки не забудьте активизировать новую конфигурацию.

1 *Для ядер серии 2.4 и более ранних требуется ввести две команды **make dep** и **make bzImage**. Если вы уже пытались компилировать это ядро, не забудьте ввести команду **make clean** для удаления старых модулей.*

5 Средства работы с пакетами в Linux

Хочешь мира, готовься к войне.

5.1 Netfilter

http://www.netfilter.org/ - базовый сайт проекта

http://lists.netfilter.org/ - списки рассылок

http://www.netfilter.org/patch-o-matic/ - репозиторий дополнительных модулей

Netfilter представляет собой основу для фильтрации и изменения пакетов в системах Linux. **Netfilter** включает в себя 3 основных элемента, обеспечивающих поддержку широкого спектра возможностей управления пакетами на сетевом и транспортом уровнях модели OSI, а также возможность разработки дополнительных приложений.

1) Для каждого поддерживаемого протокола[1] определяются функции-ловушки (hook) - четко определенные точки на пути пакетов по стеку протоколов. Эти функции встраиваются непосредственно в ядро или реализуются в форме загружаемых модулей. В каждой из таких точек-ловушек протокол может использовать функции **netfilter**, передавая им пакет и номер ловушки.

2) Модули ядра могут регистрироваться (callback-функции) для просмотра разных ловушек для каждого протокола. Поэтому при прохождении пакета через **netfilter** проверяется регистрация модулей ядра для данного протокола и ловушки. При наличии зарегистрированных функций последние могут проверить (а в некоторых случаях и изменить) пакет - отбросить (**NF_DROP**) или пропустить (**NF_ACCEPT**) его, сказать netfilter о необходимости забыть этот пакет (**NF_STOLEN**) или попросить поместить пакет в очередь пользовательского пространства (**NF_QUEUE**).

3) Пакеты, помещенные в очередь, собираются драйвером **ip_queue** и обрабатываются асинхронно.

Обширные комментарии в программном коде и документации, позволяющие разобраться в работе программ и написать свои дополнения к ним.

В дополнение к описанной схеме могут создаваться различные модули, обеспечивающие дополнительные возможности работы с пакетами. К таким модулям, в частности, относятся NAT (трансляция адресов) и iptables (фильтрация пакетов).

На примере протокола IPv4 рассмотрим работу ловушек, поскольку это весьма важно для понимания функций netfilter.

Модули (встроенные или загружаемые) ядра Linux могут регистрироваться для прослушивания ловушек netfilter. Регистрирующий свою функцию модуль должен указать приоритет своей функции для данной ловушки. Когда ловушка netfilter вызывается кодом ядра, функции модулей ядра, зарегистрированные для этой ловушки, вызываются в соответствии с их уровнем приоритета. Получив пакет, зарегистрированная функция может выполнять с ним те или иные операции, после чего передает его netfilter для выполнения одной из перечисленных ниже операций:

1. **NF_ACCEPT** - продолжить нормальную обработку пакета;
2. **NF_DROP** - отбросить пакет без дальнейшей обработки;
3. **NF_STOLEN** - забрать (украсть) пакет, исключая его из дальнейшего прохождения стека протоколов;
4. **NF_QUEUE** - поместить пакет в очередь (обычно для обработки программами пользовательского пространства);
5. **NF_REPEAT** - повторно вызвать функцию.

С помощью описанных здесь схематически операций можно строить мощные системы обработки пакетов, позволяющие решать различные классы задач (в том числе и задачи обеспечения информационной безопасности).

5.1.1 Основные возможности программ

- фильтрация пакетов без учета состояния соединений (**stateless**) для протоколов IPv4 и Ipv6;
- фильтрация пакетов с учетом состояния соединений (**stateful**) для протокола IPv4;
- поддержка всех типов трансляции (преобразования сетевых адресов и номеров портов NAT/NAPT);
- гибкая, расширяемая система правил (цепочек и таблиц);
- многоуровневый интерфейс API для создания пользовательских приложений;
- большое число дополнительных модулей, доступных в репозитории **patch-o-matic**[2]

5.1.2 Что можно делать с помощью netfilter/iptables?

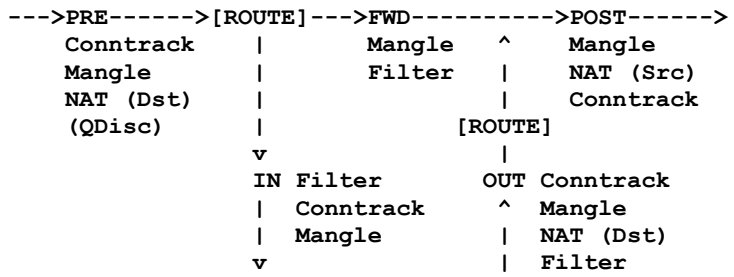
- строить межсетевые экраны с фильтрацией пакетов и трансляцией адресов;
- использовать трансляцию и маскирования адресов при использовании разделяемых каналов доступа во внешние сети даже при наличии единственного публичного адреса IP
- использовать NAT для создания прозрачных proxy-служб
- с использованием программ tc и iproute2 строить эффективные системы маршрутизации на основе правил с поддержкой QoS
- изменять биты TOS/DSCP/ECN в заголовках пакетов IP

1 *IPv4, IPv6, DECnet*
2 *http://cvs.netfilter.org/patch-o-matic-ng/*

5.1.3 Система выбора пакетов iptables

Система выбора (фильтрации) пакетов в **netfilter** носит название **iptables**. Эта система является прямым наследником программ **ipchains** (ядра серии 2.2) и **ipfwadm** (ядра серии 2.0), которые берут свое начало от **ipfw IIRC** (BSD), но существенно расширена по своим возможностям в сравнении с предшественниками. Модули ядра могут теперь регистрировать новые таблицы правил (условий) и запрашивать прохождение пакетов через такие таблицы. Такие методы выбора пакетов позволяют создавать системы фильтрации (таблица **filter**), преобразования адресов (таблица **nat**) и модификации пакетов (таблица **mangle**).

На рисунке показаны ловушки, регистрируемые в netfilter для протокола IPv4, и около каждой ловушки указаны функции в том порядке, в котором они будут вызываться[1]:



5.1.4 Ловушки Netfilter

```
#define HOOK_PRE_ROUTING        NF_IP_PRE_ROUTING
#define HOOK_LOCAL_IN           NF_IP_LOCAL_IN
#define HOOK_FORWARD            NF_IP_FORWARD
#define HOOK_LOCAL_OUT          NF_IP_LOCAL_OUT
#define HOOK_POST_ROUTING       NF_IP_POST_ROUTING
#ifdef NF_IP_DROPPING
#define HOOK_DROPPING           NF_IP_DROPPING
#endif
```

5.1.5 Цепочки iptables

Программы netfilter/iptables поддерживает в ядре Linux несколько (по крайней мере, три) таблиц с цепочками правил для пакетов IP. Таблицы ядра используются для распределения функциональности по нескольким наборам правил, называемых цепочками. Каждая цепочка представляет собой упорядоченный список правил соответствия для пакетов IP. Если данный пакет соответствует правилу, для этого пакета применяется заданная правилом операция (target). Если же пакет не соответствует спецификации данного правила, этот пакет передается следующему правилу цепочки и т. д. Пользователь может создавать свои цепочки, которые могут служить в качестве операций для встроенных и пользовательских цепочек.

Для добавления, удаления, изменения и просмотра правил в цепочках служат команды iptables (раздел 5.1.9.2).

5.1.5.1 Встроенные цепочки

Встроенные цепочки являются базовым элементом системы обработки и фильтрации пакетов netfilter/iptables. Каждая цепочка представляет собой упорядоченный список правил, имеющих вид

<условие> [опции условия] [...<условие> [опции условия]] <операция> [опции операции]

Пакеты проходят через таблицы и цепочки в определенном порядке (см. параграф 5.1.7), зависящем от происхождения и назначения пакета. При попадании правила в ту или иную цепочку, для пакета проверяется соответствие заданным спецификацией правила условиям. Если пакет соответствует условиям, по отношению к нему выполняется заданная правилом операция, в качестве каковой может использоваться одна из стандартных операций iptables (см. параграф 5.1.8 на стр. 109) или пользовательская цепочка (стр. 104). В результате выполнения заданной операции обработка правила в данной цепочке может прекратиться (операции DROP, ACCEPT, REJECT и др.) или продолжиться после завершения операции (LOG, ULOG и др.). В последнем случае или при несоответствии пакета условиям данного правила этот пакет передается следующей цепочке. Процесс продолжается до выхода из цепочки (операция RETURN, стр. 109) или завершения цепочки. Если пакет дошел до конца одной из встроенных цепочек INPUT, FORWARD или OUTPUT, к этому пакету применяется операция, заданная политикой цепочки (см. параграф 5.1.9.2.1.7 на стр. 119). Для всех прочих цепочек в конце применяется неявная операция RETURN, обеспечивающая завершение работы цепочки и возврат в точку вызова.

5.1.5.1.1 PREROUTING

Цепочка **PREROUTING** служит для выполнения операций, которые должны быть применены к пакету до принятия окончательного решения о маршрутизации. К таким операциям относится преобразование (трансляция) адресов, изменение полей заголовков пакета и др. Не следует включать в эту цепочку какие-либо средства фильтрации пакетов, поскольку такая фильтрация может привести к нежелательным результатам.

Цепочки PREROUTING используются в таблицах **raw** (стр. 104), **mangle** (стр. 105), **nat** (стр. 105).

1 *Более подробное рассмотрение этапов обработки пакетов приводится в параграфе 5.1.7 (стр. 107).*

5.1.5.1.2 INPUT

Цепочка **INPUT** служит для работы с пакетами, адресованными непосредственно данному хосту (брандмауэру). Правила цепочек INPUT могут использоваться для фильтрации пакетов, трансляции адресов, изменения полей и других операций.

Цепочки INPUT используются в таблицах **mangle** (стр. 105) и **filter** (стр. 104).

5.1.5.1.3 FORWARD

Цепочка **FORWARD** служит для выполнения большинства операций по фильтрации и изменению пакетов, пересылаемых между интерфейсами шлюза.

Цепочки FORWARD используются в таблицах **mangle** (стр. 105) и **filter** (стр. 104).

5.1.5.1.4 POSTROUTING

Встроенная цепочка **POSTROUTING** используется для выполнения действий, которые целесообразно проводить после принятия решения о пересылке пакета (маршрутизации). К таким операциям относится трансляция адресов, изменение заголовков пакетов и др. Не следует использовать в правилах цепочек POSTROUTING средства фильтрации пакетов, поскольку это может привести к нежелательным последствиям.

Цепочки **POSTROUTING** используются в таблицах **mangle** (стр. 105) и **nat** (стр. 105).

5.1.5.1.5 OUTPUT

Цепочка **OUTPUT** применяется по отношению к пакетам, сгенерированным процессами на данном хосте (брандмауэре). Правила этой цепочки могут применяться для изменения заголовков пакетов, трансляции адресов, фильтрации и др.

Цепочки FORWARD используются в таблицах **mangle** (стр. 105), **nat** (стр. 105) и **filter** (стр. 104).

5.1.5.2 Пользовательские цепочки

Пользовательские цепочки могут добавляться в любую из таблиц (см. раздел 5.1.6) **iptables** и служат для выполнения различных операций по обработке пакетов и протоколированию работы. Пользовательские цепочки могут применяться в качестве операций (target) в правилах всех встроенных цепочек iptables, наряду со стандартными операциями (раздел 5.1.8).

Для создания пользовательских цепочек служит команда **-N** (стр. 118), для переименования - **-E** (стр. 118), для удаления - **-X** (стр. 118).

5.1.6 Таблицы iptables

5.1.6.1 Таблица raw

Таблица raw используется в netfilter самой первой (даже до прохождения подсистемы контроля соединений conntrack). Эта таблица содержит две встроенные цепочки PREROUTING (параграф 5.1.5.1.1 на стр. 103) и OUTPUT (параграф 5.1.5.1.5 на стр. 104). Для работы с этой таблицей в ядре должна быть включена опция **raw table support** (стр. 81). Если для опции было выбрано значение M, работа с таблицей станет возможна после загрузки модуля **iptable_raw**.

В цепочках этой таблицы может использоваться операция NOTRACK (стр. 113), позволяющая выбрать пакеты, которые не следует передавать подсистемам контроля соединений (conntrack) и трансляции адресов (NAT). Следует помнить, что для пакетов, помеченных с помощью операции NOTRACK, не может использоваться ни одна из функций контроля состояния соединений (включая отслеживание ICMP error, протокольные функции (protocol helper) и т. п.), а также ни одна из функций NAT. Состояние NOTRACK можно использовать в качестве критерия соответствия (состояние UNTRACKED, см. параграф 5.1.9.5.5 на стр. 127) правил фильтрации, как можно видеть из приведенного ниже примера:

```
# Для загруженного web-сервера отменим операции контроля соединений и трансляции
# адресов
iptables -t raw -A PREROUTING -d 1.2.3.4 -p tcp --dport 80 -j NOTRACK
iptables -t raw -A PREROUTING -s 1.2.3.4 -p tcp --sport 80 -j NOTRACK
...

# в правилах фильтрации укажем, что все подобные пакеты следует принимать для
# пересылки между интерфейсами
iptables -A FORWARD -m state --state UNTRACKED -j ACCEPT
```

Операция TRACE (стр. 115) позволяет контролировать прохождение пакета через цепочки правил. Если пометить пакет с помощью данной операции (-j TRACE), то при выполнении условий любого из последующих правил iptables будет делаться запись в журнальном файле как при использовании операции LOG (стр. 112) или ULOG (стр. 116).

5.1.6.2 Фильтрация пакетов (filter)

Таблица filter используется лишь для фильтрации и никогда не изменяет фильтруемые пакеты. Для реализации системы пакетных фильтров должна быть включена опция при компиляции ядра Linux. Если для этой опции было

выбрано значение M, перед использованием таблицы filter потребуется загрузить модуль **iptable_filter**. Таблица фильтрации содержит встроенные цепочки INPUT (параграф 5.1.5.1.2 на стр. 104), FORWARD (параграф 5.1.5.1.3 на стр. 104) и OUTPUT (параграф 5.1.5.1.5 на стр. 104).

Одним из преимуществ фильтров iptables перед фильтрами ipchains является то, что они работают быстрее и могут собирать пакеты в точках NF_IP_LOCAL_IN[1], NF_IP_FORWARD[2] и NF_IP_LOCAL_OUT[3]. Это означает, что каждый пакет может фильтроваться в одной (**и только одной**) точке (ловушке)[4]. Фильтрация в одной точке значительно упрощает понимание работы фильтров и процесс создания цепочек. Дополнительное упрощение фильтров обеспечивается тем, что ловушка NF_IP_FORWARD работает как для входного, так и для выходного интерфейсов пересылающего пакет маршрутизатора.

Отметим также, что правила ipchains/ipfwadm и работающие с ними пользовательские программы можно использовать совместно с netfilter, включив опции совместимости IP_NF_COMPAT_IPCHAINS (стр. 81) и IP_NF_COMPAT_IPFWADM (стр. 81) при компиляции ядра.

5.1.6.3 Трансляция адресов и номеров портов (таблица nat)

Правила таблицы nat используются для трансляции сетевых адресов и номеров портов. Ловушки таблицы размещаются в двух точках - **NF_IP_PRE_ROUTING**[5] (трансляция адресов получателей - DNAT) и **NF_IP_POST_ROUTING** (трансляция адресов отправителей - SNAT). При включенной опции ядра **NAT of local connections** (стр. 78) используются две дополнительных ловушки **NF_IP_LOCAL_OUT** и **NF_IP_LOCAL_IN**, позволяющие изменить адрес получателя для сгенерированных данным хостом пакетов.

Эта таблица отличается от таблицы filter тем, что через цепочки nat проходит только первый пакет каждого соединения, а для остальных пакетов в данном соединении автоматически применяются выбранные операции преобразования адресов и номеров портов.

5.1.6.3.1 Маскирование (Masquerading), пересылка в другие порты (Port Forwarding) и прозрачные службы Proxy

В контексте Linux функции NAT обычно разделяют на Source NAT (SNAT - изменение адреса отправителя в первом пакете) и Destination NAT (DNAT - изменение адреса получателя в первом пакете).Маскирование представляет собой частный случай трансляции Source NAT, а функции **port forwarding** (пересылка в другой порт) и **transparent proxying** (прозрачный proxy) являются частным случаем трансляции Destination NAT. Все эти функции реализуются в одном блоке NAT.

5.1.6.4 Изменение пакетов (таблица mangle)

Средства изменения пакетов (таблица **mangle**) применяются для изменения данных, содержащихся в некоторых полях заголовков пакетов. Примером таких изменений могут служить операции **TOS** (стр. 115) и **TCPMSS** (стр. 115). Таблица **mangle**, начиная с ядра 2.4.18[6], может использовать все 5 типов ловушек, имеющихся для протокола IPv4.

Для использования возможностей изменения пакетов требуется ядро со включенной опцией **Packet mangling** (стр. 79). Если для опции было выбрано значение M, преобразование пакетов станет возможным после загрузки модуля **iptable_mangle**.

5.1.6.5 Контроль состояния соединений

В этом параграфе рассматривается машина состояний, обеспечивающая контроль за состоянием сетевых соединений Linux. Понимание работы машины состояний имеет важное значение при создании системы фильтрации пакетов.

Машина состояний представляет собой специальный модуль в ядре Linux, обеспечивающий в реальности систему отслеживания состояния соединений для сетевого стека - conntrack. Для использования возможностей этой системы требуется ядро со включенной при компиляции опцией **Connection tracking (required for masq/NAT)**[7]. Контроль соединений требуется модулям Netfilter для того, чтобы иметь информацию о состоянии того или иного сетевого соединения. Для межсетевых экранов обычно требуется поддержка функций stateful inspection - проверки пакетов с учетом состояния соединений. Такая функциональность существенно повышает уровень безопасности в защищаемой брандмауэром сети и позволяет создавать более эффективные и понятные правила фильтрации.

Таблицы iptables могут прослеживать связь пакетов с тем или иным соединением и различать 4 варианта состояний:

- **NEW** - пакет служит для организации нового соединения;
- **ESTABLISHED** - пакет связан с одним из существующих соединений;

1 *Пакеты, адресованные на какой-либо из интерфейсов данного хоста. Для обработки этих пакетов используется встроенная цепочка INPUT.*
2 *Пакеты, пересылаемые маршрутизатором между интерфейсами и обрабатываемые с помощью правил встроенной цепочки FORWARD.*
3 *Пакеты, сгенерированные данным хостом, которые должны быть переданы каким-либо из его интерфейсов. Для работы с такими пакетами используются правила встроенной цепочки OUTPUT.*
4 *В ipchains пакеты могли проходить через правила всех цепочек - INPUT, FORWARD и OUTPUT.*
5 *В эти две ловушки могут попадать только пакеты, принятые маршрутизатором через один из своих интерфейсов. Сгенерированные данным хостом пакеты минуют ловушки.*
6 *Более ранние версии поддерживали для этой таблицы только цепочки PREROUTING и OUTPUT.*
7 *См. параграф 4.4.2.2.14.2.1 на стр. 74.*

- **RELATED** - соединение связано с другим соединением, находящемся в состоянии ESTABLISHED;
- **INVALID** - пакет не относится ни к одному из существующих соединений и не служит для организации нового соединения.

Ниже мы рассмотрим каждое из этих состояний более подробно. Поддерживаемая iptables проверка соответствия --state (см. параграф 5.1.9.5.5 на стр. 127) позволяет создавать правила фильтрации с учетом состояния соединений и легко обнаруживать попытки организации новых сетевых соединений.

Все функции контроля соединений осуществляются модулем ядра Linux, носящим название conntrack. Этот модуль может быть загружаемым или встраивается в ядро в зависимости от выбранных опций компиляции ядра (см. параграф 4.4.2.2.14.2.1 на стр. 74.). Кроме основного модуля контроля состояний , обеспечивающего базовые функции, используются модули для отдельных протоколов (TCP, UDP, ICMP), предоставляющие более специфичную для соединений каждого из протоколов информацию, позволяющую контролировать все потоки данных через межсетевой экран. Собранные системой контроля соединений данные позволяют определить состояние каждого из существующих информационных потоков. Например, потоки данных UDP обычно можно идентифицировать по IP-адресам и номерам портов отправителя и получателя пакетов.

В ядрах версий до 2.4 имелась возможность управлять дефрагментацией пакетов. Поскольку iptables и Netfilter обеспечивают контроль состояния соединений, эта опция была исключена из новых версий ядра. Дело в том, что обеспечить контроль соединений при отключенной дефрагментации весьма проблематично, поэтому функции дефрагментации были включены в модуль conntrack и включаются автоматически. Поддержку дефрагментации невозможно отключить, не отказавшись от контроля соединений. При включенной опции **Connection tracking** (параграф 4.4.2.2.14.2.1 на стр. 74.) дефрагментация подключается автоматически.

Все операции контроля соединений осуществляются до передачи пакетов в цепочку PREROUTING за исключением операций контроля для сгенерированных данным хостом пакетов, которые проходят только через цепочку OUTPUT. Если мы передаем в поток новый пакет, сгенерированный данным хостом, цепочка OUTPUT получит состояние NEW, а при получении ответного пакета состояние будет изменено для цепочки PREROUTING на ESTABLISHED. Если первый в соединении пакет не является локальным, цепочка PREROUTING получит состояние NEW.

5.1.6.5.1 Состояния соединений

При обработке внутри ядра пакеты могут находиться в различных состояниях по отношению к соединениям в зависимости от протокола. Однако за пределами ядра для описания всех возможных ситуаций достаточно 4 состояний, перечисленных выше и более подробно рассмотренных в таблице 7. Эти состояния будут в дальнейшем использоваться с опцией state для определения принадлежности пакетов к тому или иному соединению с учетом данных системы контроля соединений.

Таблица 7. Состояния сетевых соединений Linux.

Имя	Описание
NEW	Состояние **NEW** говорит о том, что пакет является первым, который был обнаружен модулем conntrack для данного (возможно не существующего) соединения. Например, пакет SYN, являющийся первым пакетом для данного соединения, соответствует состоянию **NEW**. Однако не только пакеты SYN могут быть связаны с состоянием **NEW**. Такой подход может вызывать проблемы в некоторых случаях, но эта возможность очень полезна при попытках восстановить оборванные соединения с другими брандмауэрами или когда соединение разорвано по тайм-ауту, но не закрыто.
ESTABLISHED	Состояние **ESTABLISHED** говорит о потоке трафика в обоих направлениях. Соединения в состоянии **ESTABLISHED** просты для понимания. Для перехода в это состояние достаточно того, чтобы хост передал пакет и получил на него отклик от другого хоста. Соединения, находящиеся в состоянии **NEW** после получения отклика переходят в состояние **ESTABLISHED**. Сообщения об ошибках или перенаправления ICMP также могут рассматриваться как соединения **ESTABLISHED**, если это отклик на сгенерированное локально сообщение ICMP.
RELATED	Соединение считается находящимся в состоянии **RELATED**, если оно связано с другим соединением, уже находящимся в состоянии **ESTABLISHED**. Это означает, что до констатации состояния **RELATED** мы должны иметь соединение в состоянии **ESTABLISHED**. Последнее **ESTABLISHED** может породить новое соединение, которое будет рассматриваться как **RELATED**, если модуль conntrack сможет отследить связь между этими соединениями. Хорошим примером соединений в состоянии **RELATED** являются соединения FTP-data относительно соединений с портом FTP control, а также соединения DCC, организуемые через IRC. Использование контроля состояния позволяет передавать отклики ICMP, организовывать передачу данных по протоколу FTP и организовывать соединения DCC через межсетевые экраны. Отметим, что большинство протоколов TCP и некоторые протоколы UDP, использующие подобные механизмы, достаточно сложны и могут передавать информацию о соединениях в сегментах данных TCP или UDP и, следовательно, требуют специальной обработки (с помощью helper-модулей).
INVALID	Состояние **INVALID** означает, что пакет не может быть отнесен ни к одному из перечисленных выше состояний. Это может происходить в результате нехватки памяти в системе, сообщений об ошибках ICMP, не относящихся ни к одному из имеющихся соединений, и т. д. В общем случае имеет смысл отбрасывать (**DROP**) пакеты, для которых установлено состояние **INVALID**.

Описанные здесь состояния можно использовать вместе с соответствием **--state** (см. параграф 5.1.9.5.5 на стр. 127) для фильтрации пакетов в зависимости от состояния соединений. Такая возможность позволяет создавать

межсетевые экраны с мощной системой контроля трафика, учитывающей состояние каждого соединения. До появления таких фильтров зачастую приходилось открывать все порты с номерами более 1024 для свободного прохода пакетов-откликов в сеть. Система отслеживания соединений и контроля их состояний обеспечивает возможность эффективной фильтрации входящего трафика.

5.1.7 Прохождение пакетов через таблицы и цепочки

Для понимания процессов фильтрации важно разобраться в процессах, происходящий при перемещении пакетов по стеку сетевых протоколов. В последующих параграфах мы рассмотрим все этапы обработки пакетов, принятых хостом для себя или пересылки другим хостам. Особое внимание будет уделено процессам принятия решения о пересылке пакетов (маршрутизация) и преобразовании сетевых адресов (**DNAT** и **SNAT**).

Когда пакет принимается одним из сетевых интерфейсов брандмауэра, данные передаются драйверу соответствующего устройства (драйвер является частью ядра или загружаемым модулем ядра). Далее пакет проходит многоэтапную обработку в ядре, после чего передается подходящему приложению (пакет адресован данному хосту) или пересылается в другой интерфейс (маршрутизация). Возможно, что в процессе обработки пакет будет отброшен без передачи приложению или пересылки в другой интерфейс (фильтрация).

5.1.7.1 Пакеты, адресованные данному хосту

Рассмотрим сначала процесс обработки пакетов, адресованных данному хосту и не требующих маршрутизации. Такие пакеты проходят цепочку операций, перечисленных в таблице до передачи пакета тому или иному приложению.

Таблица 8 Этапы обработки пакетов, адресованных локальному хосту

№	Таблица или модуль	Цепочка	Действия
1			Пакет принимается из сетевой среды (кабеля)
2			Пакет поступает в сетевой интерфейс (например, eth0 или ppp0)
3	**conntrack**		Собирается и запоминается в таблице (см. стр. 106) информация о состоянии соединения, к которому может относиться данный пакет
4	**raw**	**PREROUTING**	Эта цепочка позволяет отменить для пакета операции контроля соединений
5	**mangle**	**PREROUTING**	Данная цепочка служит для изменения пакетов до трансляции адресов и маршрутизации (изменение параметров TOS и т. п.).
6	**nat**	**PREROUTING**	Эта цепочка служит прежде всего для трансляции адресов DNAT. Не следует использовать в данной цепочке какие-либо фильтры, поскольку в некоторых случаях эти фильтры очень просто обходятся.
7	**QDiscipline**		На этом этапе используется система управления трафиком, помещающая пакет в очередь или отбрасывающая его[1].
8	**routed**		Принимается решение о маршрутизации пакета. В рассматриваемом случае маршрутизация не нужна[2] (пакет адресован локальному хосту).
9	**filter**	**INPUT**	Эта цепочка используется для фильтрации пакетов, адресованных данному хосту. Отметим, что через данную цепочку проходят все пакеты, адресованные этому хосту, независимо от того, через какой интерфейс был принят пакет.
10	**conntrack**		Собирается и запоминается в таблице (см. стр. 106) информация о состоянии соединения, к которому может относиться данный пакет
11	**mangle**	**INPUT**	В этой точке с помощью правил цепочки INPUT могут выполняться дополнительные операции по изменению пакета, которые следует осуществлять после принятия решения о маршрутизации, но до передачи пакета локальным процессам.
12			Пакет передается локальному процессу или прикладной программе для дальнейшей обработки.

Отметим, что для рассмотренного случая не используются цепочки таблиц FORWARD и OUTPUT. На первый взгляд это может показаться странным[3], но при более внимательном рассмотрении кажущаяся странность исчезает. Цепочка INPUT служит только для обработки пакетов, адресованных данному хосту, FORWARD применяется к пересылаемым между сетевыми интерфейсами пакетам, а OUTPUT служит только для работы с пакетами, сгенерированными данным хостом.

5.1.7.2 Локально сгенерированные пакеты

В таблице 3 перечислены этапы обработки пакетов, сгенерированных данным хостом.

1 *Информацию о системе управления трафиком в Linux вы сможете найти в на сайте http://www.tldp.org/HOWTO/Traffic-Control-HOWTO/*
2 *Вариант пересылки пакета в другой интерфейс (маршрутизация) рассматривается в параграфе 5.1.7.3.*
3 *Кажущаяся странность усиливается при сравнении с цепочками ipchains, где каждый пакет проходил через цепочки INPUT, FORWARD и OUTPUT.*

Таблица 9. Этапы обработки пакетов, сгенерированных локальным хостом.

№	Таблица или модуль	Цепочка	Действия
1			Пакет принят от локального процесса или приложения.
2	**conntrack**		Собирается и запоминается в таблице (см. стр. 106) информация о состоянии соединения, к которому может относиться данный пакет
3	**mangle**	**OUTPUT**	На этом этапе используются правила цепочки OUTPUT для изменения пакета. Не рекомендуется в данной цепочке использовать какие-либо фильтры, поскольку в таких случаях могут возникать побочные эффекты.
4	**nat**	**OUTPUT**	Трансляция адресов (DNAT) в пакетах, сгенерированных брандмауэром.
5	**filter**	**OUTPUT**	Фильтрация пакетов, сгенерированных локальным хостом.
6	**routed**		Принимается решение о маршрутизации пакета для выбора интерфейса, через который пакет должен быть передан для доставки адресату. На этом этапе также может выбираться адрес, указываемый в поле отправителя для данного пакета.
7	**mangle**	**POSTROUTING**	Правила цепочки POSTROUTING из таблицы mangle используются для изменения пакетов перед их передачей в сетевой интерфейс.
8	**nat**	**POSTROUTING**	В этой цепочке осуществляется трансляция адресов SNAT. Не следует создавать в данной цепочке какие-либо фильтры, поскольку их достаточно просто обойти даже при выборе для цепочки политики **DROP**.
9	**conntrack**		Собирается и запоминается в таблице (см. стр. 106) информация о состоянии соединения, к которому может относиться данный пакет
10			Пакет передается в один из сетевых интерфейсов (например, eth0)
11			Сетевой интерфейс передает данные (пакет) в сетевую среду.

5.1.7.3 Пересылаемые пакеты

Для пакетов, пересылаемых маршрутизатором между своими интерфейсами, осуществляется самое большое число операций. Этапы обработки таких пакетов рассмотрены в таблице 4.

Таблица 10. Этапы обработки пересылаемых маршрутизатором пакетов.

№	Таблица или модуль	Цепочка	Действия
1			Пакет принимается из сетевой среды (кабеля).
2			Пакет поступает в сетевой интерфейс (например, eth0 или ppp0).
3	**conntrack**		Собирается и запоминается в таблице (см. стр. 106) информация о состоянии соединения, к которому может относиться данный пакет.
4	**raw**	**PREROUTING**	Эта цепочка позволяет отменить для пакета операции контроля соединений
5	**mangle**	**PREROUTING**	Данная цепочка служит для изменения пакетов до трансляции адресов и маршрутизации (изменение параметров TOS и т. п.).
6	**nat**	**PREROUTING**	Эта цепочка служит прежде всего для трансляции адресов DNAT. Не следует использовать в данной цепочке какие-либо фильтры, поскольку в некоторых случаях эти фильтры очень просто обойти.
7	**QDiscipline**		Система управления трафиком, помещающая пакет в очередь или отбрасывающая его[1].
8	**routed**		Принимается решение о маршрутизации пакета.
9	**mangle**	**FORWARD**	После принятия решения о маршрутизации пакет передается цепочке FORWARD таблицы mangle. Правила этой цепочки могут служить для внесения в пакеты изменений, которые целесообразно выполнить после маршрутизации, но до передачи пакета в сетевой интерфейс.
10	**filter**	**FORWARD**	Здесь выполняется практически вся работа по фильтрации пересылаемых маршрутизатором пакетов. Помните, что пересылаемые маршрутизатором пакеты проходят через эту цепочку независимо от направления.
11	**mangle**	**POSTROUTING**	Правила цепочки POSTROUTING из таблицы mangle используются для изменения пакетов перед их передачей в сетевой интерфейс.
12	**nat**	**POSTROUTING**	В этой цепочке осуществляется трансляция адресов SNAT и маскирование. Не следует создавать в данной цепочке какие-либо фильтры, поскольку их достаточно просто обойти даже при выборе для цепочки политики **DROP**.
13	**conntrack**		Собирается и запоминается в таблице (см. стр. 106) информация о состоянии соединения, к которому может относиться данный пакет
14			Пакет передается в один из сетевых интерфейсов (например, eth1)
15			Сетевой интерфейс передает данные (пакет) в сетевую среду.

1 *Информацию о системе управления трафиком в Linux вы сможете найти в на сайте http://www.tldp.org/HOWTO/Traffic-Control-HOWTO/*

5.1.8 Операции iptables

В каждом правиле цепочек iptables указывается действие (target), выполняемое по отношению к пакету при совпадении с условиями, заданными спецификацией правила. Если пакет не удовлетворяет спецификации, он передается следующему правилу, а для соответствующих спецификации пакетов используется операция, заданная параметром -j (см. параграф 5.1.9.1.2 на стр. 117). В качестве действия может использоваться одна из описанных ниже стандартных операций iptables или пользовательская цепочка (стр. 104). Кроме встроенных операций можно использовать дополнительные операции, загрузив соответствующий код с сайта http://www.netfilter.org/patch-o-matic/ и собрав модули на своем компьютере из исходных кодов. Отметим, что некоторые расширения netfilter/iptables могут потребовать компиляции ядра.

5.1.8.1 Основные операции

Основные операции поддерживаются программой iptables независимо от загрузки модулей ядра и подключения модулей netfilter, если при компиляции ядра была включена опция глобальной поддержки Netfilter/iptables (**Network packet filtering**), описанная в параграфе 4.4.2.2.14.1 (стр. 73).

5.1.8.1.1 ACCEPT

Операция **ACCEPT** означает восприятие пакета, завершение его обработки в цепочках данной таблицы и передачу на дальнейшие этапы обработки. Для такого пакета последующие правила цепочек данной таблицы уже не используются, однако следует помнить, что пакет может попасть в другие цепочки на этапах дальнейшей обработки.

5.1.8.1.2 DROP

В результате операции **DROP** пакет отбрасывается без дальнейшей обработки и передачи отправителю пакета какого-либо уведомления.

Отметим, что в некоторых случаях отбрасывание пакетов без уведомления их отправителя может вызывать побочные эффекты (в частности, возникновение “мертвых” сокетов). В таких случаях разумно использовать описанную ниже операцию **REJECT** в результате которой пакет удаляется из обработки, но отправителю выдается сообщение об ошибке. В частности, использование операции **REJECT** позволяет предотвратить получение с помощью сканеров информации о закрытых в вашей системе номерах портов и т. п.

Важно понимать, что отброшенный с помощью операции **DROP** пакет просто исключается из какой-либо дальнейшей обработки и не передается никаким приложениям или протоколам вышележащих уровней - пакет просто “тихо умирает”.

5.1.8.1.3 QUEUE

Операция QUEUE ведет к передаче пакета в пользовательское пространство, если такая возможность поддерживается ядром (см. параграф 4.4.2.2.14.2.2 на стр. 75). Для использования этой операции также требуется приложение[1] пользовательского пространства, способное работать с очередями пакетов.

Стандартный обработчик очередей для iptables и протокола IPv4 реализуется в модуле ip_queue.

Ниже показан пример использования операции QUEUE в пакетном фильтре:

```
modprobe iptable_filter
modprobe ip_queue
iptables -A OUTPUT -p icmp -j QUEUE
```

В соответствии с этим правилом локально сгенерированные пакеты ICMP (скажем, ping) будут передаваться модулю ip_queue, который попытается доставить эти пакеты приложению пользовательского пространства. Если ожидающего пакетов приложения пользовательского пространства не будет обнаружено, пакеты будут отброшены.

Для создания приложений используется интерфейсная библиотека **libipq** (API), включенная в дистрибутив iptables. Примеры таких приложений можно найти среди тестовых программ из дистрибутива iptables (например, redirect.c).

Состояние очереди ip_queue можно посмотреть в файле:

```
/proc/net/ip_queue
```

Максимальный размер очереди (число пакетов, доставляемых в пользовательское пространство до вынесения решения об их судьбе) определяется переменной в файле:

```
/proc/sys/net/ipv4/ip_queue_maxlen
```

По умолчанию максимальный размер очереди составляет 1024 пакета. По достижении этого значения новые пакеты будут отбрасываться, пока размер очереди не уменьшится. “Деликатные” протоколы типа TCP интерпретируют отбрасывание пакетов как насыщение и будут снижать скорость передачи пакетов при заполнении очереди. Однако для определения оптимального значения порога переполнения очереди в вашем конкретном случае могут потребоваться эксперименты.

5.1.8.1.4 RETURN

Операция **RETURN** завершает обработку пакета в данной цепочке и возвращает управление в вызвавшую ее (предыдущую) цепочку. Если операция **RETURN** используется во встроенной цепочке или достигнут конец

1 *Такие приложения могут использоваться для решения различных задач, включая учет трафика, дополнительную фильтрацию и т. д.*

встроенной цепочки, для пакета используется операция, заданная политикой данной цепочки (см. параграф 5.1.9.2.1.7 на стр. 119). Логически операция **RETURN** эквивалентна переходу в конец цепочки и во многих случаях позволяет значительно сократить время прохождения пакета через последовательность правил цепочки.

5.1.8.2 Дополнительные операции

Поддержка дополнительных операций обусловлена опциями компиляции ядра и загрузкой соответствующих модулей ядра. Однако, эти операции не требуют подключения дополнительных модулей **iptables**.

5.1.8.2.1 BALANCE

Эта операция позволяет транслировать адреса получателя (как это делает операция **DNAT**, описанная на стр. 111), но с перебором адресов из заданного опцией блока.

`--to-destination <начало>-<конец блока>`
задает блок адресов из которого данная операция будет по кругу выбирать адреса для трансляции **DNAT**.

Для использования операции **BALANCE** требуется ядро со включенной опцией **IP: fast network address translation** (параграф на стр. 70).

5.1.8.2.2 CLASSIFY

Операция **CLASSIFY** позволяет устанавливать для пакетов значение поля уровня приоритета **skb->priority**, которое может использоваться некоторыми дисциплинами для классификации пакетов в системах QoS. К такого типа дисциплинам относятся:

- atm
- cbq
- dsmark
- pfifo_fast
- htb
- prio

Операция может применяться только в цепочке POSTROUTING таблицы mangle и использует единственный параметр

`--set-class MAJOR:MINOR`
задающий значение для поля приоритета в сетевом буфере **skb** (см. Приложение 12.16).

`iptables -t mangle -A POSTROUTING .. -j CLASSIFY --set-class MAJOR:MINOR`
Для использования операции **CLASSIFY** требуется ядро со включенной поддержкой опции **CLASSIFY target support** (стр. 79). Если для опции было выбрано значение **M**, использовать классификацию пакетов можно будет после загрузки модуля **ipt_CLASSIFY**.

5.1.8.2.3 CLUSTERIP

Операция **CLUSTERIP** позволяет создать простой кластер узлов, использующих общую пару адресов IP и MAC, без явной системы распределения (балансировки) трафика перед этим кластером. Соединения будут статически распределяться между узлами кластера.

Операция поддерживает несколько опций для создания и управления кластером.

`--new`
создает новый кластер **ClusterIP**. Эта операция всегда должна быть первой среди операций для данного **ClusterIP**.

`--hashmode <режим>`
задает режим хэширования[1] и может принимать значения **sourceip** (распределение по адресу отправителя), **sourceip-sourceport** (адрес и порт отправителя), **sourceip-sourceport-destport** (адрес и порт отправителя, порт получателя).

`--clustermac mac`
задает MAC-адрес для кластера. По сути, этот адрес является multicast-адресом канального уровня.

`--total-nodes <количество>`
задает общее число узлов в кластере.

`--local-node <номер>`
определяет локальный номер узла в кластере.

`--hash-init rnd`
задает случайное значение, используемое при инициализации hash-функции.

5.1.8.2.4 CONNMARK

Эта операция позволяет установить маркер netfilter для данного соединения. Маркеры могут использоваться другими правилами (см. описание проверки **mark** на стр. 148) для фильтрации соединений или распределения

1 *Распределения соединений между узлами кластера.*

трафика. Операция использует несколько опций для установки и управления маркерами соединений.

--set-mark <маркер>[/<маска>]
устанавливает маркер для соединения. Если опция содержит маску, устанавливаются только те биты маркера, которые соответствуют этой маске.

--save-mark [--mask <маска>]
копирует установленное для пакета значение маркера в качестве маркера соединения. Если опция содержит маску, копируются только соответствующие этой маске биты маркера.

--restore-mark [--mask <маска>]
копирует маркер соединения в поле маркера пакета. Если опция содержит маску, копируются только соответствующие этой маске биты маркера. Эту опцию можно использовать только в цепочках таблицы **mangle** (параграф 5.1.6.4 на стр. 105).

5.1.8.2.5 DNAT

Операция **DNAT** может использоваться в цепочках **PREROUTING** (параграф 5.1.5.1.1 на стр. 103) и **OUTPUT**(параграф 5.1.5.1.5 на стр. 104) таблицы **nat** (параграф 5.1.6.3 на стр. 105), а также пользовательских цепочках, которые могут вызываться **только** из указанных цепочек и служит для трансляции адреса получателя в исходящих пакетах. При использовании этой операции в заголовке данного пакета и всех последующих пакетов данного потока (соединения) изменяется значение поля IP-адреса получателя. При маршрутизации пакетов они пересылаются в интерфейс, соответствующий измененному адресу получателя.

Опция

--to-destination <адрес>[-<адрес>][:<порт>-<порт>]
служит для указания адреса (диапазона адресов) и номера (диапазона номеров) порта, используемого в качестве адреса получателя после трансляции. Отметим, что задание портов допускается только в правилах, содержащих опцию **-p tcp** или **-p udp**. Если порт получателя не указан, сохраняется исходный номер порта. Допускается использование в одном правиле нескольких опций **--to-destination**, задающих различные адреса. Из всех указанных опциями значений адресов создается пул с последовательным круговым перебором адресов.

Эта операция очень полезна для тех случаев, когда к тому или иному серверу, находящемуся внутри вашей сети и не имеющему публичного адреса, требуется обеспечить доступ извне. Хорошим примером может служить ситуация размещения web-сервера в локальной сети и трансляция на адрес (частный) этого сервера всех пакетов, обращающихся к порту http по имеющемуся у вас[1] публичному адресу. Операция может использоваться с диапазоном адресов, значения из которого будут выбираться последовательно (по кругу), что может быть весьма полезно для распределения нагрузки между множеством серверов, которые для внешнего пользователя имеют общий адрес, а реально работают на разных хостах с различными адресами.

Например, команда

```
iptables -t nat -A PREROUTING -p tcp -d 15.45.23.67 --dport 80 -j DNAT --to-destination
192.168.1.1-192.168.1.10
```
обеспечит переадресацию всех запросов http, отправленных по адресу 15.45.23.67 на хосты с адресами из диапазона 192.168.1.1-192.168.1.10 без смены номера порта.

Вы можете также использовать круговую подстановку адреса получателя из заданного блока, выполняемую с помощью операции **BALANCE** (параграф 5.1.8.2.1 на стр. 110).

Для использования DNAT требуется ядро со включенной опцией **IP: fast network address translation** (стр. 70).

5.1.8.2.6 DSCP

Операция **DSCP** служит для изменения битов DSCP[2] в полях TOS заголовков IPv4. Поскольку операция меняет содержимое пакетов, ее можно использовать только в цепочках таблицы **mangle** (параграф 5.1.6.4 на стр. 105). Операция может использоваться с двумя опциями.

--set-dscp value
устанавливает для поля DSCP значение value (его можно указать в десятичном или шестнадцатеричном формате)

--set-dscp-class class
устанавливает в поле DSCP значение класса DiffServ.

Для использования операции потребуется ядро со включенной опцией **DSCP target support** (стр. 79). Если при компиляции ядра для опции было выбрано значение M, следует загрузить модуль **ipt_DSCP**.

5.1.8.2.7 ECN

Эта операция может использоваться только в цепочках таблицы mangle (параграф 5.1.6.4 на стр. 105) служит для борьбы с “черными дырами ECN[3]”, возникающими при обмене пакетами с хостами и маршрутизаторами, которые не понимают или некорректно трактуют биты ECN. Поддерживаемая для этой операции опция

--ecn-tcp-remove
удаляет все биты ECN из заголовка TCP. Отметим, что использование такой возможности требует наличия в правиле параметра **-p tcp**.

1 *Возможно, единственному*
2 *DSCP (differentiated services code point) - шестибитовое значение в поле TOS, позволяющее запросить при доставке пакетов желаемый тип обслуживания (QoS).*
3 *ECN (Explicit Congestion Notification) - явное уведомление о насыщении.*

Для работы с операцией ECN требуется включить опцию **ECN target support** (стр. 79) при компиляции ядра. Если для опции было выбрано значение M, потребуется загрузка модуля **ipt_ECN**.

5.1.8.2.8 LOG

Операция **LOG** служит для записи в системный журнал информации о соответствующих пакетах. Для записи используются встроенные средства ядра Linux (syslog). Увидеть эти сообщения можно в файле **/var/log/messages**, а последние записи выводятся на экран по команде dmesg. Эта операция не прерывает прохождения пакета через цепочку (после передачи ядру информации о пакете данный пакет передается следующему правилу цепочки). Если вы хотите использовать эту операцию о записи сведений об отвергнутых или отброшенных пакетах, следует использовать два последовательных правила с одинаковым набором условий[1]. В первом правиле используется операция **LOG**, а во втором - **DROP** или **REJECT**.

Операция LOG поддерживает несколько опций:

`--log-level <уровень протоколирования>`
определяет уровень протоколирования для syslog. Уровень может быть задан числом или предопределенным именем[2].

`--log-prefix <префикс>`
определяет префикс для записей в журнальном файле. Префикс может включать до 29 символов и весьма полезен при анализе и обработке журнальных файлов.

`--log-tcp-sequence`
говорит ядру о необходимости записи в системный журнал порядковых номеров TCP[3].

`--log-tcp-options`
говорит ядру о необходимости записи в журнал значений некоторых опций из заголовков TCP.

`--log-ip-options`
говорит ядру о необходимости записи в журнал значений большинства опций из заголовков IP.

Для работы с операцией LOG требуется ядро со включенной при компиляции опцией **LOG target support** (стр. 80). Если для опции было задано значение M, операция LOG будет работать после загрузки модуля **ipt_LOG**.

5.1.8.2.9 MARK

Эта операция используется для маркировки пакетов и может применяться только в цепочках таблицы **mangle** (параграф 5.1.6.4 на стр. 105). Установленные с помощью этой операции маркеры впоследствии могут использоваться для маршрутизации (например, с помощью iproute2), фильтрации, управления трафиком[4]. Операция использует 1 опцию

`--set-mark mark`
которая задает значение[5] устанавливаемого маркера.

Для использования операции маркировки пакетов ядро должно иметь включенную опцию **MARK target support** (стр. 79). Если для опции было выбрано значение M, потребуется загрузка модуля **ipt_MARK**.

5.1.8.2.10 MASQUERADE

Эта операция служит для маскирования (Masquerading) адресов и может применяться только в цепочке POSTROUTING (параграф 5.1.5.1.4 на стр. 104) таблицы nat (параграф 5.1.6.3 на стр. 105). Операцию следует применять для соединений с динамическими адресами IP, а при работе со статическими адресами разумнее использовать операцию SNAT (параграф 5.1.8.2.17 на стр. 114).

Маскирование эквивалентно отображению пакетов на IP-адрес исходящего интерфейса с удалением информации об этом отображении при отключении (down) интерфейса. Такое поведение корректно для случаев, когда при следующем соединении вероятность получить такой же адрес мала и, следовательно, все организованные ранее соединения так или иначе будут потеряны. Операция **MASQUERADE** может использоваться с опцией

`--to-ports port[-port]`
задающей номер (или диапазон номеров) порта отправителя для маскируемых пакетов. Эта опция позволяет изменить используемое по умолчанию значение номера порта, выбранное эвристической машиной SNAT (параграф 5.1.8.2.17). Использование опции ограничивается правилами, содержащими в спецификации указание на протокол TCP или UDP[6] (**-p tcp** или **-p udp**).

Для поддержки операции маскирования при компиляции ядра должна быть включена опция **MASQUERADE target support** (параграф 4.4.2.2.14.2.4 на стр. 78). Если при компиляции было выбрано значение M, для маскирования адресов потребуется загрузка модуля **ipt_MASQUERADE**.

1 *Зачастую более эффективно создать пользовательскую цепочку, записывающую информацию в журнальный файл и прерывающую обработку для всех пакетов, а направлять пакеты в эту цепочку по интересующему вас набору условий. Такой подход позволяет сократить общее число правил в таблице и снизить время обработки, поскольку условия соответствия проверяются один раз вместо двух.*
2 *Имена уровней протоколирования можно узнать с помощью команды Linux* ***man syslog.conf***
3 *Информация о порядковых номерах может быть использована злоумышленниками, поэтому не следует применять эту опцию, если к журнальным файлам имеют доступ рядовые пользователи.*
4 *Дополнительную информацию вы сможете найти в документе Linux Advanced Routing and Traffic Control HOW-TO, доступном на сайте http://www.faqs.org/docs/iptables/otherresources.html#LARTC*
5 *Целое число.*
6 *См. параграф 5.1.9.3.1.1 на стр. 121.*

5.1.8.2.11 NETMAP

Операций **NETMAP** позволяет создавать статическое взаимно-однозначное (1:1) отображение сетевых адресов без смены адреса хоста. Операция NETMAP использует опцию

```
--to адрес[/маска]
```

для задания диапазона адреса или адресов, в который выполняется отображение.

Приведенная ниже команда позволяет изменить адреса получателей для входящих соединений с 1.2.3.0/24 на 5.6.7.0/24

```
iptables -t nat -A PREROUTING -d 1.2.3.0/24 -j NETMAP --to 5.6.7.0/24
```

Для использования операции **NETMAP** требуется ядро со включенной поддержкой опции **NETMAP target support** (стр. 78). Если для опции было выбрано значение M, операция станет доступной после загрузки модуля **ipt_NETMAP**.

5.1.8.2.12 NOTRACK

Эта операция может использоваться в цепочках таблицы raw (см. параграф 5.1.6.1 на стр. 104), которая используется в netfilter самой первой (даже до системы отслеживания соединений conntrack) и включает две встроенных цепочки PREROUTING (параграф 5.1.5.1.1 на стр. 103) и OUTPUT (параграф 5.1.5.1.5 на стр. 104).

Операция **NOTRACK** позволяет выбрать пакеты, которые **не будут** передаваться в подсистему conntrack/NAT для контроля соединений и трансляции адресов. Следует помнить, что при использовании операции **NOTRACK** для пакетов

- не используется система контроля соединений conntrack (не отслеживаются сообщения ICMP об ошибках, helper-модули и т. п.);
- не выполняется никаких действий по преобразованию сетевых адресов и номеров портов - NAT.

Пакеты, отмеченные с помощью операции NOTRACK, соответствуют предопределенному состоянию UNTRACKED, которое может использоваться в других правилах (см. параграф 5.1.9.5.5 на стр. 127). Например, для сильно загруженного web-сервера можно использовать правила

```
iptables -t raw -A PREROUTING -d 1.2.3.4 -p tcp --dport 80 -j NOTRACK
iptables -t raw -A PREROUTING -s 1.2.3.4 -p tcp --sport 80 -j NOTRACK
```

а в цепочку фильтрации включить правило

```
iptables -A FORWARD -m state --state UNTRACKED -j ACCEPT
```

в соответствии с которым все такие пакеты будут безоговорочно приниматься.

Для использования операции NOTRACK в ядре должна быть включена опция **NOTRACK target support** (стр. 81). Если при компиляции ядра для опции было выбрано значение M, для работы с NOTRACK потребуется загрузить модуль **ipt_NOTRACK**.

5.1.8.2.13 REDIRECT

Операция **REDIRECT** позволяет переправлять пакеты и потоки трафика на данный хост. Операция может использоваться в цепочках PREROUTING (параграф 5.1.5.1.1) и OUTPUT (параграф 5.1.5.1.5) таблицы nat (параграф 5.1.6.3), а также пользовательских цепочках, которые могут вызываться **только** из указанных цепочек. При использовании **REDIRECT** изменяется IP-адрес получателя и пакеты перенаправляются маршрутизатором на себя[1]. Опция

```
--to-ports port[-port]
```

позволяет задать порт или диапазон портов, куда будут перенаправляться пакеты. Эту опцию можно использовать только в правилах, задающих протокол **-p tcp** или **-p udp**.

Операция **REDIRECT** очень эффективна для создания прозрачных proxy, о которых пользователи вашей локальной сети просто не догадываются.

Для использования операции перенаправления трафика следует включить опцию **REDIRECT target support** (стр. 78) при компиляции ядра. Если для опции было выбрано значение M, перенаправление пакетов станет возможным после загрузки модуля **ipt_REDIRECT**.

5.1.8.2.14 REJECT

Эта операция задает отбрасывание пакета с возвратом отправителю сообщения об ошибке (причине отказа). Как и в случае DROP дальнейшая обработка пакета прекращается. Эту операцию можно использовать только во встроенных цепочках INPUT, FORWARD или OUTPUT и пользовательских цепочках, которые вызываются **только** из трех перечисленных встроенных цепочек. Возвращаемая отправителю информация задается опцией

```
--reject-with <тип сообщения ICMP>
```

В качестве значения параметра **тип сообщения ICMP** можно использовать:

icmp-net-unreachable - сеть недоступна;

icmp-host-unreachable - хост недоступен;

icmp-port-unreachable - порт недоступен;

1 *Для локально сгенерированных пакетов в качестве адреса получателя указывается адрес loopback-интерфейса - 127.0.0.1.*

icmp-proto-unreachable - протокол недоступен;

icmp-net-prohibited - доступ в сеть закрыт;

icmp-host-prohibited - доступ к хосту закрыт;

icmp-admin-prohibited - доступ закрыт администратором[1].

Отметим, что в соответствии с требованиями RFC 1122 сообщения ICMP не передаются в следующих случаях:

- отброшенный пакет был сообщением ICMP об ошибке или пакетом ICMP неизвестного типа;
- отброшенный пакет был не первым фрагментом;
- за последнее время по этому адресу было передано слишком много сообщений ICMP об ошибках (см. описание переменной **/proc/sys/net/ipv4/icmp_ratelimit** на стр. 372).

В правилах для протокола TCP можно также использовать значение tcp-reset. В этом случае отправителю передается пакет TCP RST. Эта возможность полезна для блокировки запросов ident (113/tcp), которые часто используются при попытках отправить почту на неработающие (отказывающиеся принимать вашу почту) почтовые серверы.

Например команда

```
iptables -A FORWARD -p TCP --dport 22 -j REJECT --reject-with tcp-reset
```

будет добавлять в цепочку FORWARD правило, отвергающее все пакеты TCP, адресованные в порт 22, с возвратом отправителю пакета TCP RST.

Для использования операции **REJECT** при компиляции ядра должна быть включена опция **REJECT target support** (см. параграф 4.4.2.2.14.2.3.20.1 на стр. 77). Если вы выбрали для данной опции значение M для использования операции REJECT потребуется загрузить модуль **ipt_REJECT**.

5.1.8.2.15 ROUTE

Операция ROUTE служит для явной отмены решения о маршрутизации, принятого на уровне ядра и указания иного маршрута передачи пакета. Операция использует несколько опций, обеспечивающих управление маршрутизацией пакетов.

```
--oif <интерфейс>
```

задает отправку пакета через заданный именем интерфейс.

```
--iif <интерфейс>
```

меняет для пакета информацию о принявшем его интерфейсе (поле **rx_dev** структуры **skb**, описанное на стр. 402).

```
--gw <IP-адрес>
```

указывает шлюз для маршрутизации пакета.

```
--continue
```

задает продолжение обработки пакета в цепочке правил. Эта опция не может использоваться совместно с -**iif**.

5.1.8.2.16 SAME

Операция **SAME** транслирует сетевые адреса подобно SNAT и предоставляет клиенту один и тот же адрес для каждого соединения. Операция использует опцию

```
--to <ipaddr>-<ipaddr>
```

для задания диапазона адресов, используемых операцией при подстановке, а опция

```
--nodst
```

говорит о том, что при выборе адреса отправителя для подстановки, адрес получателя не должен приниматься во внимание.

Приведенная ниже команда

```
iptables -t nat -A POSTROUTING -j SAME --to 1.2.3.4-1.2.3.7
```

будет менять в исходящих пакетах поле адреса отправителя на значение из диапазона 1.2.3.4-1.2.3.7.

Для использования операции **SAME** требуется ядро со включенной поддержкой опции **SAME target support** (стр. 78). Если для опции было выбрано значение M, операция станет доступной после загрузки модуля **ipt_ SAME**.

5.1.8.2.17 SNAT

Операция **SNAT**, использовать которую можно только в цепочке **POSTROUTING** (параграф 5.1.5.1.4 на стр. 104) таблицы **nat** (параграф 5.1.6.3 на стр. 105), обеспечивает трансляцию (замену) адреса отправителя для исходящего пакета (и всех последующих пакетов в этом соединении). Команда используется с опцией

```
--to-source  ipaddr[-ipaddr][:port-port]
```

которая позволяет задать новый адрес (диапазон адресов) IP и номер (диапазон номеров) порта[2] в поле заголовка IP. Если номер порта для протокола TCP или UDP не указан, порты с номерами < 512 будут отображаться в порты с номерами ниже 512, порты из диапазона 512 - 1023 будут отображаться в порты с номерами <1024, а все остальные порты будут отображаться в порты с номерами 1024 и выше. В таких случаях преобразование номера

1 *При использовании типа icmp-admin-prohibited с ядрами, не поддерживающими такую возможность взамен операции REJECT по сути будет использоваться DROP (отправитель не получит сообщения).*
2 *Номера или диапазоны портов можно указывать только в правилах, содержащих спецификацию протокола -p tcp или -p udp (см. параграф 5.1.9.3.1.1 на стр. 121).*

порта происходит только при реальной необходимости. Допускается использование в одном правиле нескольких опций --to-source. Если задано использование множества адресов отправителя (диапазон или несколько опций -to-source), эти адреса будут перебираться последовательно (по кругу).

Опция **SNAT** полезна в тех случаях, когда в вашей сети используются адреса из частных блоков[1] IP, поскольку она позволяет отобразить для всех исходящих соединений адреса отправителей на публичный адрес, предоставленный вам провайдером Internet.

Для использования операции **SNAT** требуется ядро со включенной опцией **IP: fast network address translation** (параграф на стр. 70).

5.1.8.2.18 TCPMSS

Эта операция изменяет значение MSS[2] в заголовках пакетов TCP SYN, позволяя управлять максимальным размером сегмента для соединения. Операция может использоваться только для пакетов TCP и требует наличия в строке правила спецификации протокола **-p tcp**. Кроме того, для использования этой операции нужно включить опцию **TCPMSS target support** (параграф 4.4.2.2.14.2.17 на стр. .80). Эта операция может служить для борьбы с "отмороженными" провайдерами и серверами, которые блокируют сообщения ICMP Fragmentation Needed[3]. Симптомы проблемы состоят в том, что с Linux-брандмауэра (маршрутизатора) обеспечивается беспрепятственный доступ, но стоящие за брандмауэром компьютеры не могут обмениваться большими пакетами с внешними хостами:

- подключение к Web-серверам происходит беспрепятственно, но при загрузке содержимого процесс "умирает";
- мелкие почтовые сообщения приходят нормально, а большие не доходят совсем;
- ssh работает хорошо, но scp зависает после стартовой инициализации.

Для решения проблемы можно использовать команду типа приведенной ниже:

```
iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu
```

Опция

```
--set-mss value
```

используется для явной установки значения MSS, а

```
--clamp-mss-to-pmtu
```

автоматически выбирает для MSS значение (path_MTU - 40). Опции исключают одна другую и не могут использоваться совместно.

Для работы с операцией TCPMSS в ядре должна быть включена опция **TCPMSS target support** (стр. 80). Если для опции было выбрано значение M, потребуется также загрузка модуля **ipt_TCPMSS**.

5.1.8.2.19 TOS

Эта операция используется для установки значения 8-битового поля Type of Service (TOS) в заголовках пакетов IP. Операция может использоваться только в цепочках таблицы **mangle** (5.1.6.4, стр. 105). Для использования этой операции нужно включить поддержку опции **TOS target support** при компиляции ядра (стр. 79). При выборе для опции значения M, использование операции TOS потребует предварительной загрузки модуля **ipt_TOS**.

Опция

```
--set-tos tos
```

определяет значение поля TOS, устанавливаемое для пакета. Для просмотра возможных значений используйте команду Linux

```
iptables -j TOS -h
```

Установленное значение TOS может использоваться локальными программами обработки сетевых пакетов (например, iproute2) или внешними маршрутизаторами[4]. Значительная часть современных маршрутизаторов поддерживает TOS, поэтому имеет смысл предпринять попытку запросить нужные условие доставки перед отправкой пакетов во внешние сети. Ничего страшного не произойдет, если маршрутизаторы на пути доставки не поддерживают TOS - в таких случаях это поле просто не принимается во внимание. При использовании же этого поля в пределах своей сети можно оказать существенное влияние на картину внутрисетевого трафика.

Приведенная ниже команда

```
iptables -t mangle -A PREROUTING -p TCP --dport 22 -j TOS --set-tos 0x10
```

запрашивает для пакетов TCP, передающих данные протокола SSH, минимальную задержку при доставке.

5.1.8.2.20 TRACE[5]

С помощью операции TRACE можно отслеживать прохождение пакета через цепочки правил. Если при прохождении цепочек таблицы raw (стр. 104) для пакета указана операция TRACE, при выполнении условий любого из следующих правил на пути прохождения пакета через цепочки в журнальный файл системы будет помещаться запись вида:

1 *10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 (см. RFC 1918).*
2 *MSS (maximum segment size) - запрашиваемый размер максимального сегмента для пакетов TCP. Обычно этот размер составляет MTU - 40.*
3 *Указывает на необходимость фрагментации.*
4 *Операция TOS существенно отличается от операции MARK (5.1.8.2.9, 112) - маркеры действуют только внутри ядра, а значение TOS может приниматься во внимание на всем пути доставки пакетов.*
5 *Эта операция может использоваться только в цепочках таблицы raw.*

TRACE: <имя таблицы>/<имя цепочки>/<номер правила> <пакет>
Операция не использует каких-либо опций.

Для записи в журнал требуется также загрузка по крайней мере одного из модулей протоколирования (ipt_LOG или ipt_ULOG). Если вы загружены оба модуля, будет использоваться “встроенное” журналирование с помощью ipt_LOG, но вы можете отдать предпочтение пользовательскому модулю ipt_ULOG, если при его загрузке воспользуетесь параметром takeover

```
modprobe ipt_ULOG takeover=1
```

5.1.8.2.21 TTL

Операция **TTL** позволяет задать значение поля времени жизни в заголовках пакетов IP. Операцию можно применять только в цепочках таблицы **mangle** (5.1.6.4, стр. 105). Операция может использовать 3 параметра

--ttl-set <значение TTL>
позволяет явно задать значение времени жизни для пакета. Не следует устанавливать слишком большое значение TTL для пакетов, направленных в вашу внутреннюю сеть, это может вызвать проблемы.

--ttl-dec <шаг уменьшения TTL>
позволяет задать дополнительное снижение значения TTL. Не следует забывать, что каждый маршрутизатор при пересылке пакетов уменьшает это значение на 1. Например, при установке для параметра значения 3 реальное уменьшение TTL после маршрутизации составит 4.

--ttl-inc <шаг увеличения TTL>
позволяет увеличить значение времени жизни для пакетов. С помощью этого параметра можно попытаться “спрятать” ваш брандмауэр от программ трассировки.

На практике эта операция нужна пожалуй лишь для борьбы с провайдерами, которые не позволяют подключать по одному каналу более одной машины. В таких случаях мы просто устанавливаем одинаковое значение (скажем, стандартное для Linux значение 64) для всех исходящих пакетов. Может эта операция быть полезной также в тех случаях, когда желательно ограничить сферу досягаемости тех или иных служб[1].

5.1.8.2.22 ULOG

Операция **ULOG** служит для записи информации о соответствующих заданным условиям пакетов с помощью средств ведения рабочих журналов на уровне пользовательского пространства. Эта операция не прерывает прохождения пакета через цепочку и служит лишь для записи сведений о пакете. При соответствии пакета заданным условиям ядро Linux будет передавать этот пакет с использованием групповой адресации (multicast) через сокет **netlink** (Приложение 12.10). Включенные в соответствующую multicast-группу приложения пользовательского пространства будут получать такие пакеты. Команда ULOG поддерживает несколько опций:

--ulog-nlgroup nlgroup
позволяет задать номер группы netlink (1-32), которой будут адресоваться пакеты. По умолчанию используется группа 1.

--ulog-prefix prefix
задает префикс, который включается в начало записи журнального файла. Префикс может включать до 32 символов. Использование префиксов упрощает анализ и обработку журнальных файлов.

--ulog-cprange size
число байтов из пакета, копируемых в пользовательское пространство. При выборе значения 0 (используется по умолчанию) пакет копируется целиком, независимо от его размера.

--ulog-qthreshold size
определяет размер очереди пакетов в ядре. При установке отличного от 1 значения ядро будет аккумулировать заданное число пакетов и передавать его в пользовательское пространство как одно сообщение netlink (это сообщение может быть разбито на части). Для обратной совместимости по умолчанию используется значение 1.

Для работы с операцией ULOG требуется ядро со включенной при компиляции опцией **ULOG target support** (стр. 80). Если для опции было задано значение M, операция ULOG будет работать после загрузки модуля **ipt_ULOG**.

5.1.8.3 Операции расширения

Кроме основных и дополнительных операций программа iptables может работать с операциями расширения, которые существуют в виде отдельных модулей и не входят в стандартный комплект распространения программы. Такие модули доступны на сайте http://www.netfilter.org. Установка дополнительных модулей из пакета **patch-o-matic** описана в параграфе 4.3.1.1 (стр. 61).

5.1.9 Создание правил фильтрации пакетов

Сценарии загрузки политики безопасности включают команды загрузки модулей, требуемых для работы и набора цепочек правил для таблиц iptables. Каждая строка правил iptables имеет вид

```
iptables [-t <таблица>] -<команда> [<цепочка>] [<номер правила>] [<спецификация правила]
-j операция [опции операции]
```

Для правил конкретного типа синтаксис может несколько отличаться от показанного выше и будет более подробно

1 *Например, если вы не хотите предоставлять свой сервер DNS слишком удаленным клиентам, можно искусственно уменьшить время жизни для пакетов DNS.*

рассмотрен при описании каждой проверки, поддерживаемой программой iptables.

5.1.9.1 Опции команд iptables

5.1.9.1.1 Выбор таблицы (-t)

По умолчанию все команды iptables выполняются применительно к таблице filter (параграф 5.1.6.2 на стр. 104). Для выбора другой таблицы[1] служит команда

```
iptables -t <имя таблицы>
```

или

```
iptables --table <имя таблицы>
```

Если при компиляции ядра была выбрана опция автоматической загрузки модулей и требуемый модуль еще не загружен, будет предпринята попытка загрузить модуль. Для загрузки модулей вручную служит опция **-modprobe** (см. параграф 5.1.9.1.8 на стр. 118) или команда Linux **modprobe <имя модуля>**.

5.1.9.1.2 Выбор операции для правила (-j)

Большинство цепочек iptables заканчиваются указанием операции, которая должна быть применена по отношению к пакету, при выполнении условий, заданных спецификацией правила (см. раздел 5.1.9.3). Для указания операции служит опция -j. Обычно эта опция используется последней в строке задания правила и имеет формат

```
-j <имя операции>
```

или

```
-jump <имя операции>
```

Параметр <имя операции> может быть именем одной из встроенных операций iptables (см. раздел 5.1.8) или пользовательской цепочки (см. параграф 5.1.5.2).

5.1.9.1.3 Объем выводимой информации (-v)

Опция **-v** (или **--verbose**) позволяет вывести с помощью команды -L (см. параграф 5.1.9.2.1.5 на стр. 118) дополнительную информацию, включающую опции правила, маски TOS, имена интерфейсов и значения счетчиков пакетов и байтов. По умолчанию значения счетчиков выводятся с округлением и суффиксами K (1 000), M (1 000 000) и G (1 000 000 000). Используя описанную ниже опцию -x, можно задать вывод полных (точных) значений счетчиков.

При использовании опции -v в командах добавления, вставки, замены и удаления правил в цепочках обеспечивается вывод детальной информации о соответствующих правилах.

5.1.9.1.4 Формат адресов при выводе (-n)

Опция -n (или -numeric) позволяет выбрать для вывода информации представление хостов с помощью адресов IP взамен используемых по умолчанию доменных имен. Использование этой опции значительно ускоряет вывод информации, поскольку не требуется делать многочисленных запросов DNS для преобразования IP-адресов в имена хостов.

Эта опция используется совместно с командой **-L** (параграф 5.1.9.2.1.5 на стр. 118).

5.1.9.1.5 Формат представления значений счетчиков (-x)

Эта опция служит для управления форматом вывода значений счетчиков пакетов и байтов. По умолчанию значения счетчиков выводятся с округлением и суффиксами K (1 000), M (1 000 000) или G (1 000 000). Опция **-x** (**--exact**) позволяет выводить точные значения счетчиков без использования префиксов и округления.

Эта опция используется совместно с командой -L (параграф 5.1.9.2.1.5 на стр. 118) и опцией -v.

5.1.9.1.6 Инициализация счетчиков (-c)

Опция **-c** (**--set-counters**) позволяет задать стартовые значения для счетчиков пакетов и байтов при добавлении (5.1.9.2.2.1, стр. 120), вставке (5.1.9.2.2.2, стр. 120) или замене (5.1.9.2.2.3, стр. 120) правил в цепочках iptables. Опция может использоваться в формате

```
-c <значение счетчика пакетов> <значение счетчика байтов>
```

или

```
--set-counters <значение счетчика пакетов> <значение счетчика байтов>
```

5.1.9.1.7 Нумерация строк

Опция **-line-numbers** позволяет при выводе списков правил (команда -L, параграф 5.1.9.2.1.5 на стр. 118) показывать в каждой строке номер правила в цепочке. Информация о номере правила может быть полезна при работе с командами добавления (5.1.9.2.2.1, стр. 120), вставки (5.1.9.2.2.2, стр. 120), замены (5.1.9.2.2.3, стр. 120) и удаления (5.1.9.2.2.4, стр. 120) правил в цепочках iptables.

1 *raw (параграф 5.1.6.1 на стр. 104), nat (параграф 5.1.6.3 на стр. 105) или mangle (параграф 5.1.6.4 на стр. 105).*

5.1.9.1.8 Загрузка модулей

При использовании правил, компоненты (действия, соответствия и т. п.) опция

```
--modprobe=command
```

служит для загрузки модуля, указанного значением параметра command. Вы можете также загрузить модули заранее с помощью команды Linux

```
modprobe <имя модуля>
```

5.1.9.2 Команды iptables

Программа iptables поддерживает множество команд и опций, позволяющих задать точную спецификацию правил фильтрации, преобразования адресов и изменения пакетов. Отметим, что все команды по умолчанию связаны с правилами таблицы filter и для применения команд к цепочкам других таблиц следует использовать опцию[1]

```
-t <имя таблицы>
```

или

```
--table <имя таблицы>
```

5.1.9.2.1 Команды управления цепочками и таблицами в целом

Команды управления цепочками и таблицами в целом используют форма

```
iptables [-t имя таблицы] команда <имя цепочки>
```

Если в команде отсутствует имя таблицы, эта команда выполняется применительно к таблице filter.

5.1.9.2.1.1 Создание цепочки

Команда

```
iptables -N <имя цепочки>
```

или

```
iptables --new-chain <имя цепочки>
```

позволяет создать пользовательскую цепочку с указанным именем.

5.1.9.2.1.2 Удаление цепочки

Для удаления существующей пользовательской цепочки служит команда

```
iptables -X <имя цепочки>
```

или

```
iptables --delete-chain <имя цепочки>
```

Удаляемая цепочка не должна содержать правил[2] и не должна использоваться в качестве операции (target) ни в одной из оставшихся цепочек. Если в команде не указано имя удаляемой цепочки, будет предпринята попытка удалить все пользовательские цепочки. При попытке удаления встроенной цепочки выдается сообщение об ошибке:

```
iptables: Can't delete built-in chain
```

5.1.9.2.1.3 Переименование цепочки

Для изменения имени пользовательской цепочки служит команда

```
iptables -E <старое имя цепочки> <новое имя цепочки>
```

или

```
iptables --rename-chain <старое имя цепочки> <новое имя цепочки>
```

Эта команда не оказывает реального влияния на работу цепочек и служит лишь для “косметических” целей.

5.1.9.2.1.4 Сброс цепочки

Для удаления из цепочки всех правил (сброса) можно использовать команду

```
iptables -F <имя цепочки>
```

или

```
iptables --flush <имя цепочки>
```

Если в команде не указано имя цепочки, будет предпринята попытка удаления всех правил из каждой цепочки.

5.1.9.2.1.5 Просмотр списка правил в цепочках

Команда

```
iptables -L <имя цепочки>
```

или

1 *См. параграф 5.1.9.1.1 на стр. 117.*
2 *Для удаления правил служат команды -d (см. параграф на стр.) и -F (см. параграф 5.1.9.2.1.4).*

```
iptables --list <имя цепочки>
```
служит для просмотра всех правил в указанной цепочке. На рисунке показан пример вывода для встроенных и пользовательских цепочек.

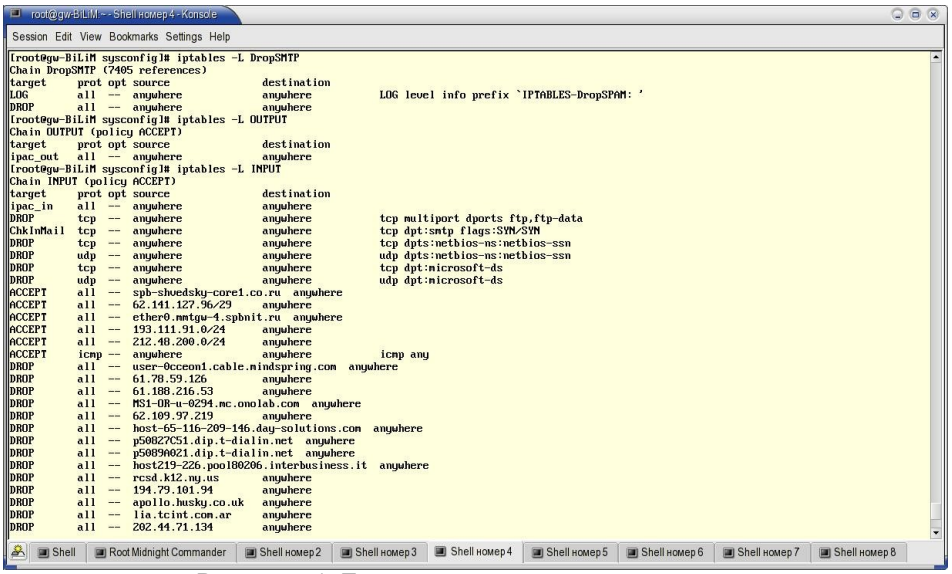


Рисунок 5.1. Просмотр списка правил в цепочках.

В первой строке вывода указывается имя цепочки и принятая для нее политика (встроенные цепочки) или количество правил, использующих данную цепочку в качестве действия (target). Нулевое значение счетчика использования цепочки говорит о возможности ее удаления.

Если команда не содержит имени цепочки, выводится список правил во всех цепочках (даже пустых).

Команда -L поддерживает три опции:

- **-n** - при выводе списка правил используются IP-адреса вместо имен хостов (см. параграф 5.1.9.1.4 на стр. 117);
- **-v** - вывод дополнительной информации (см. параграф 5.1.9.1.3 на стр. 117), включающей значения счетчиков пакетов и байтов, параметры TOS, имена интерфейсов;
- **-x** - задает формат вывода значений счетчиков при использовании опции -v (см. параграф 5.1.9.1.5 на стр. 117).

5.1.9.2.1.6 Сброс счетчиков

Для сброса счетчиков может использоваться команда

```
iptables -Z <имя цепочки>
```
или

```
iptables --zero <имя цепочки>
```
Эта команда может быть весьма полезна для установки нулевых значений счетчиков после просмотра их значений с помощью команды -L

```
iptables -L -Z FORWARD
```
Можно выполнить эту операцию с помощью 2 раздельных команд -L и -Z, но следует помнить, что за время последовательного выполнения команд значения счетчиков могут измениться.

5.1.9.2.1.7 Установка политики для цепочек

Команда

```
iptables -P <имя цепочки> <название политики>
```
или

```
iptables --policy <имя цепочки> <название политики>
```
позволяет выбрать для таблицы принятую по умолчанию политику для любой из встроенных цепочек INPUT (параграф 5.1.5.1.2 на стр. 104), FORWARD (параграф 5.1.5.1.3 на стр. 104) и OUTPUT (параграф 5.1.5.1.5 на стр. 104). В качестве политики могут использоваться стандартные операции iptables (см. параграф 5.1.8). Заданная политикой цепочки операция выполняется по отношению к пакетам, прошедшим через все правила данной цепочки[1]. Политика цепочки определяет судьбу пакета, не соответствующего ни одному из правил данной цепочки.

Политика не задается для пользовательских цепочек, поскольку после завершения проверки соответствия пакета всем заданным цепочкой правилам происходит возврат в точку вызова данной пользовательской цепочки. В качестве политики недопустимо использовать встроенные цепочки iptables (параграф 5.1.5.1).

Пример выбора для цепочки INPUT политики отбрасывания пакетов показан ниже

```
iptables -P INPUT DROP
```

5.1.9.2.2 Команды управления отдельными правилами в цепочках

Описанные ниже команды служат для создания правил фильтрации, преобразования адресов или изменения пакетов, а также для замены и удаления существующих в цепочке правил. Команды этой группы имеют вид

```
iptables [-t имя таблицы] -[AIRD] <имя цепочки> <спецификация соответствия> <операция>
```
Параметр <операция> определяет действие, выполняемое по отношению к пакету при его соответствии заданным правилом условиям. В качестве операции могут использоваться стандартные операции iptables - target (параграф

1 *Т. е., не соответствующих ни одному из заданных в правилах цепочки условий.*

5.1.8 на стр.) или пользовательские цепочки (параграф 5.1.5.2 на стр. 104).

Каждое правило указывает имя таблицы (по умолчанию правила включаются в таблицу filter), команду (что делать с данным правилом), условия соответствия (что проверять для пакета) и операцию, выполняемую по отношению к пакету при его соответствии заданным условиям.

5.1.9.2.2.1 Добавление правила в цепочку

Для добавления правила в конец цепочки служит команда

```
iptables -A <имя цепочки> <спецификация соответствия> <операция>
```

или

```
iptables --append <имя цепочки> <спецификация соответствия> <операция>
```

При необходимости вставленное в цепочку правило можно заменить другим (параграф 5.1.9.2.2.3 на стр. 120) или удалить (параграф 5.1.9.2.2.4 на стр. 120). Описание спецификаций соответствия приводится в разделе 5.1.9.3).

5.1.9.2.2.2 Вставка правила в указанную позицию

Если вам нужно поместить новое правило в определенную строку цепочки, можно воспользоваться командой

```
iptables -I <имя цепочки> <номер правила> <спецификация соответствия> <операция>
```

или

```
iptables --insert <имя цепочки> <номер правила> <спецификация соответствия> <операция>
```

Нумерация правил в цепочке начинается с 1. Посмотреть нумерацию имеющихся в таблице правил можно определить с помощью команды (см. параграф 5.1.9.2.1.5 на стр. 118) типа

```
iptables -L INPUT -line-numbers
```

После вставки правила в цепочку номера строк для последующих правил автоматически увеличиваются на 1 - об этом следует помнить при выполнении других операций. Команда -I достаточно редко используется в сценариях загрузки правил, но может быть полезна для вставки в цепочку временных правил.

При необходимости вставленное в цепочку правило можно заменить другим (параграф 5.1.9.2.2.3 на стр. 120) или удалить (параграф 5.1.9.2.2.4 на стр. 120). Описание спецификаций соответствия приводится в разделе 5.1.9.3).

5.1.9.2.2.3 Замена правила в указанной позиции

Команда

```
iptables -R <имя цепочки> <номер правила> <спецификация соответствия> <операция>
```

или

```
iptables --replace <имя цепочки> <номер правила> <спецификация соответствия> <операция>
```

позволяет заменить правило в строке <номер правила> на новое, спецификация которого задается данной командой. Номер правила можно определить с помощью команды (см. параграф 5.1.9.2.1.5 на стр. 118) типа

```
iptables -L INPUT -line-numbers
```

Команда -R предназначена в основном для экспериментов с цепочками iptables - обычно не возникает необходимости использовать ее в рабочих цепочках[1].

5.1.9.2.2.4 Удаление правила

Для удаления правила из цепочки служит команда **-D** (**--delete**). Команда может использоваться в двух вариантах - для удаления правила в заданной строке или для удаления правила, полностью совпадающего с указанной спецификацией.

5.1.9.2.2.4.1 Удаление по номеру

Для удаления правила в указанной номером строке цепочки служит команда

```
iptables -D <имя цепочки> <номер правила>
```

или

```
iptables --delete <имя цепочки> <номер правила>
```

Номер правила можно определить с помощью команды (см. параграф 5.1.9.2.1.5 на стр. 118) типа

```
iptables -L INPUT -line-numbers
```

При удалении правил из таблицы не забывайте об автоматическом уменьшении на 1 номеров строк для всех последующих правил данной цепочки.

5.1.9.2.2.4.2 Удаление по спецификации

Для удаления правила из цепочки можно воспользоваться командой, содержащей полную спецификацию удаляемого правила

```
iptables -D <имя цепочки> <спецификация соответствия> <операция>
```

или

1 *Команда может быть также полезна для внесения временных изменений, если вы используете множество правил и не хотите перезапускать все таблицы целиком.*

```
iptables --delete <имя цепочки> <спецификация соответствия> <операция>
```
При использовании данного варианта команды заданная в строке спецификация должна в точности соответствовать имеющемуся в цепочке правилу, которое вы хотите удалить.

5.1.9.3 Соответствия для правил iptables

Соответствия iptables - это те или иные параметры пакетов, которые могут проверяться в правилах фильтрации для определения дальнейшей судьбы пакетов. Существует три группы соответствий

- **встроенные**, которые включаются в стандартный дистрибутив ядра и netfilter и могут использоваться без загрузки дополнительных модулей; в публикациях на английском языке такие соответствия, не мудрствуя лукаво, относят к числу параметров команды iptables;
- **дополнительные**[1], которые могут потребовать загрузки модулей ядра, но не требуют в правилах явного указания загрузки модулей iptables;
- **загружаемые**[2], которые требуют в правилах наличия явной команды загрузки модуля (**-m**).

5.1.9.3.1 Встроенные соответствия

Для использования этой группы соответствий не требуется загрузка дополнительных модулей ядра - достаточно лишь включить поддержку **netfilter** (стр. 73).

5.1.9.3.1.1 Соответствие протокола

Опция

```
-p протокол
```
позволяет задать протокол (TCP, UDP или ICMP), к которому должен относиться пакет, чтобы соответствовать данному правилу. Протокол можно указать по имени или номеру, используемому в конфигурационном файле **/etc/ptotocols**. Кроме того, может использоваться значение **ALL**, которому соответствуют все протоколы[3]. Допускается указание списка протоколов, разделенных запятыми. Восклицательный знак перед названием или номером протокола

```
-p !протокол
```
задает инверсию (все протоколы, кроме указанного). Например, опции **-p !UDP** будут соответствовать пакеты протоколов TCP и ICMP.

Можно использовать также запись этой опции в форме

```
--protocol протокол
```

5.1.9.3.2 Соответствие адресов IP

Соответствие адресов проверяется практически в каждом правиле iptables, поэтому следует внимательно прочесть следующие параграфы - они послужат основой для дальнейшей работы.

5.1.9.3.2.1 Адрес отправителя

Для проверки соответствия адреса отправителя (source) служит опция **-s**, которую можно представлять также в формате --**source** или --**src.**

Выражению

```
-s 192.168.0.11
```
будут соответствовать все пакеты, в заголовке которых указан IP-адрес 192.168.0.11. Допускается использование опции -s с адресом сети и маской. Например, выражению

```
-s 172.16.0.0/24
```
будут соответствовать все пакеты, принятые с адресами отправителя из диапазона 172.16.0.0 - 172.16.0.255[4]. Маску сети можно задать и в полном представлении (для приведенного выше примера - 255.255.255.0). При задании адреса допускается использование знака ! для обращения условия. Скажем, запись

```
--source !10.0.0.0/8
```
будет соответствовать всему пространству адресов IP за исключением диапазона 10.0.0.0 - 10.255.255.255.

5.1.9.3.2.2 Адрес получателя

Для проверки соответствия адреса получателя (destination) служит опция **-d**, которую можно записывать также в формате --**destination** или --**dst.**

Выражению

1 *Соответствия этого типа в английском языке обозначают термином **implicit match**, поскольку они не указывают явно используемый для сравнения модуль iptables (эти модули не требуется загружать специально, поскольку они всегда присутствуют в коде **iptables**.*
2 *Такие соответствия обозначают термином **explicit match**, поскольку они требуют явного включения в правило команды загрузки модуля проверки соответствия.*
3 *В этом случае можно просто не использовать опцию -p.*
4 *Для работы с диапазонами адресов IP требуется поддержка опции **IP range match support** в ядре (стр. 75). Если для опции было выбрано значение M, потребуется также загрузить модуль ipt_iprange.*

```
-d 192.168.0.1
```
будут соответствовать все пакеты, адресованные хосту с IP-адресом 192.168.0.1. Допускается использование опции -d с адресом сети и маской. Например, выражению

```
-d 172.16.0.0/24
```
будут соответствовать все пакеты, адресованные в сеть с адресами из диапазона 172.16.0.0 - 172.16.0.255[1]. Маску сети можно задать и в полном представлении (для приведенного выше примера - 255.255.255.0). В спецификации адреса получателя допускается использование знака ! для обращения условия. Скажем, условию

```
--dst !10.0.0.0/8
```
будут соответствовать пакеты, адресованные любому хосту за исключением диапазона адресов 10.0.0.0 - 10.255.255.255.

5.1.9.3.3 Соответствие физического интерфейса

В отличие от цепочек ipchains правила iptables могут не быть связанными с каким-то конкретным интерфейсом. Если вам нужна такая связь, используйте описанные ниже опции проверки соответствия пакета физическому интерфейсу.

5.1.9.3.3.1 Приемный интерфейс

Опция **-i** (**--in-interface**) позволяет проверить соответствие пакета тому или иному физическому интерфейсу, используемому для приема пакетов.

Например, выражению

```
-i eth0
```
будут соответствовать все пакеты, принятые через интерфейс eth0, а условию

```
--in-interface !ppp0
```
все пакеты, кроме тех, которые были приняты из интерфейса ppp0. Допускается также использование масок интерфейсов. Например, условию **-i eth+** будут соответствовать пакеты, принятые через любой из имеющихся на маршрутизаторе интерфейсов Ethernet.

Эту опцию можно использовать только в таблицах **INPUT** (стр. 104), **PREROUTING** (стр. 103) и **FORWARD** (стр. 104). При попытке использования опции в других таблицах произойдет ошибка.

5.1.9.3.3.2 Передающий интерфейс

Для указания исходящего интерфейса служит опция **-o** (**--out-interface**). Эта опция по своему назначению и применению аналогична опции -i, но может использоваться только в цепочках **FORWARD** (стр. 104), **POSTROUTING** (стр. 104) и **OUTPUT** (стр. 104).

5.1.9.3.4 Состояние фрагментации

Условие **-f** или (**--fragment**) позволяет проверить не является ли пакет вторым, третьим и т. д. фрагментом более крупного пакета. Необходимость такой проверки обусловлена тем, что для всех фрагментов, начиная со второго, отсутствует возможность определения портов отправителя и получателя, типа сообщения ICMP и некоторых других параметров. Кроме того, фрагменты зачастую используются при организации атак на компьютеры. Для фрагментов, начиная со второго, могут не выполняться условия соответствия, которым удовлетворяет первый пакет, поэтому для таких пакетов могут потребоваться иные условия обработки.

Опцию **-f** можно использовать совместно со знаком инверсии. Например, условию

```
!-f
```
будут соответствовать все пакеты, которые не являются фрагментами с номером 2 и более[2].

Отметим, что вместо контроля фрагментов можно воспользоваться поддерживаемой ядром опцией дефрагментации. Дефрагментация пакетов включается автоматически при активизации в ядре функций контроля за состоянием соединений (опция **Connection tracking**, описанная на стр. 74), поэтому при использовании таких функций фрагменты просто не будут попадать в цепочки и таблицы **iptables**.

5.1.9.4 Дополнительные сопоставления

Существует группа сопоставлений, не требующих задания дополнительных условий, но допускающих такие дополнения. Такие условия называют полными или имплицитными[3]. Условия этой группы применяются автоматически без указания модуля **iptables** и могут не включать никаких дополнительных проверок. Например, мы можем просто задать проверку соответствия протокола

--protocol tcp

не используя никаких дополнительных критериев. Существуют три группы таких условий соответствия для протоколов TCP, UDP и ICMP.

Кроме полных условий существует также группа неполных (эксплицитных) условий, описанных ниже. Для использования таких условий в спецификации правила требуется использовать опцию **-m** или **--match**, задающую

1 *Для работы с диапазонами адресов IP требуется поддержка опции **IP range match support** в ядре (стр. 75). Если для опции было выбрано значение **M**, потребуется также загрузить модуль **ipt_iprange**.*
2 *Т. е., нефрагментированные пакеты и первые фрагменты в случае использования фрагментации.*
3 *Implicit match - неявное совпадение.*

модуль netfilter, требуемый для проверки условия.

5.1.9.4.1 Протокол TCP

Описанные ниже условия соответствия можно использовать только для пакетов TCP, поэтому в каждом правиле, включающем такие условия должно присутствовать также условие

```
-p tcp
```

или

```
--protocol tcp
```

При отсутствии условия принадлежности пакетов к протоколу TCP будет выдаваться сообщение об ошибке.

5.1.9.4.1.1 Флаги TCP

Условие

```
--tcp-flags <флаг1,флаг2,...> <флаг11>
```

служит для проверки наличия (или отсутствия) тех или иных флагов TCP. Условие содежит два параметра - первый параметр задает проверяемые флаги, а второй указывает те, значения которых должны быть установлены. Например, условию

```
-t tcp --tcp-flags SYN,FIN,ACK SYN
```

будут соответствовать все пакеты TCP, в которых установлен флаг SYN, а флаги FIN и ACK сброшены. В качестве параметров условия можно использовать флаги SYN, ACK, FIN, RST, URG, PSH, а также значения ALL (все флаги) и NONE (ни одного флага). При использовании в условии нескольких флагов в одной группе, имена этих флагов разделяются запятыми[1].

Условие -tcp-flags можно использовать со знаком инверсии !. Например, условию

```
--tcp-flags ! SYN,FIN,ACK SYN
```

будут соответствовать все пакеты TCP, в которых установлены флаги FIN и ACK, но сброшен флаг SYN.

5.1.9.4.1.1.1 Флаг SYN

Для проверки состояния флага SYN существует специальное условие

```
--syn
```

которому соответствуют все пакеты организации соединений (установлен флаг SYN и сброшены флаги RST и ACK). Это условие сохранено в целях обратной совместимости с цепочками ipchains и логически эквивалентно условию

```
--tcp-flags SYN,RST,ACK SYN
```

Условие --syn можно использовать со знаком инверсии !.

5.1.9.4.1.2 Соответствие MSS (TCP SYN)

Опция --mss позволяет проверить для пакетов TCP SYN и SYN/ACK значение MSS[2], определяющее максимальный размер пакетов для данного соединения. Условие

```
--mss value[:value1]
```

проверяет соответствие MSS значению value или диапазону значений value - value1.

Для использования этого условия при компиляции ядра должна быть включена опция **tcpmss match support** (стр. 77). Если для опции было выбрано значение M, потребуется загрузить модуль **ipt_tcpmss**.

5.1.9.4.1.3 Опции TCP

Для проверки опций TCP служит условие

```
--tcp-option <значение>
```

Этому условию будут соответствовать пакеты TCP имеющие заданное условие значение (десятичное) в поле опций. Если пакет TCP не содержит опций, он не будет соответствовать условию. Условие можно использовать со знаком инверсии **--tcp-option ! <значение>**.

Опции TCP содержатся в заголовке и включают три различных поля. Первое поле имеет размер 8 битов и указывает тип опции, второе поле (8 битов) содержит размер опции, а третье поле может иметь переменную длину и содержит собственно опцию[3].

5.1.9.4.2 Соответствие портов TCP и UDP

Проверка соответствия портов TCP требует наличия в спецификации правила условия

```
-p tcp
```

или

```
--protocol udp
```

1 *Без пробела - SYN,FIN,ACK.*
2 *Максимальный размер сегмента.*
3 *Более подробную информацию об опциях вы сможете найти в спецификации протокола TCP.*

5.1.9.4.2.1 Порт отправителя

Для проверки соответствия номера порта отправителя служит условие

```
-sport <номер порта>
```

или

```
--source-port <номер порта>
```

Допускается использование в условии непрерывного диапазона номеров, заданного в формате

```
<первый порт>:<последний порт>
```

В случае задания диапазона опущенное стартовое значение трактуется как 0, а при отсутствии верхней границы используется значение 65535.

Если вам требуется указать группу номеров портов или диапазонов, разделенных промежутками, следует воспользоваться опцией **-m multiport** (стр. 127). Не забывайте только о недопустимости использования в одной спецификации условия для одного порта или диапазона и группы портов.

Для задания портов можно также использовать символьные имена, указанные в файле /etc/services вашего компьютера.

Условия проверки номеров портов можно использовать со знаком инверсии

```
--source-port ! <номер порта>
```

5.1.9.4.2.2 Порт получателя

Для проверки соответствия номера порта получателя служит условие

```
-dport <номер порта>
```

или

```
--destination-port <номер порта>
```

Допускается использование в условии непрерывного диапазона номеров, заданного в формате

```
<первый порт>:<последний порт>
```

В случае задания диапазона опущенное стартовое значение трактуется как 0, а при отсутствии верхней границы используется значение 65535.

Если вам требуется указать группу номеров портов или диапазонов, разделенных промежутками, следует воспользоваться опцией **-m multiport** (стр. 127). Не забывайте только о недопустимости использования в одной спецификации условия для одного порта или диапазона и группы портов.

Для задания портов можно также использовать символьные имена, указанные в файле /etc/services вашего компьютера.

Условия проверки номеров портов можно использовать со знаком инверсии

```
--destination-port ! <номер порта>
```

5.1.9.4.3 Протокол ICMP

Протокол ICMP используется в основном для контроля доступности хостов и передачи сообщений об ошибках. Заголовки пакетов ICMP очень похожи на заголовки IP, но между этими заголовками имеется и ряд существенных отличий. В заголовках ICMP присутствует поле типа сообщений, определяющее назначение пакетов ICMP. Например, при недоступности хоста в ответ на обращения к нему обычно отправляются сообщения типа **ICMP host unreachable**. Полный список поддерживаемых протоколом ICMP типов сообщений можно найти в спецификации протокола ICMP. Для пакетов ICMP в iptables поддерживается единственное специальное условие соответствия, но этого вполне достаточно для решения возникающих на практике задач. Для проверки этого соответствия не забывайте включить в спецификацию правила условие **--protocol ICMP**. Отметим, что для протокола ICMP все правила соответствия общего назначения (параграфы 5.1.9.3.2.1 - 5.1.9.3.4) могут использоваться как и для пакетов других протоколов.

5.1.9.4.3.1 Тип сообщения ICMP

Условие

```
--icmp-type [!] <тип сообщения>
```

позволяет проверить для пакетов тип сообщения ICMP. Типы сообщений можно указывать по их номерам или символьным номерам. Числовые значения соответствуют спецификации RFC 792. Для получения полного списка имен сообщений ICMP, поддерживаемых iptables, воспользуйтесь командой

```
iptables --protocol icmp -help
```

В условиях проверки типа сообщений ICMP можно использовать знак инверсии **!**. Например, условию

-p icmp **--icmp-type ! 8**

будут соответствовать все типы сообщений ICMP, кроме сообщений ICMP echo (ping). Отметим, что некоторые типы сообщений ICMP устарели и не должны использоваться, а применение иных типов может быть “опасным” для незащищенных хостов.

5.1.9.5 Загружаемые соответствия

Если для полных условий соответствия модули netfilter загружаются автоматически, то неполные[1] соответствия можно использовать, лишь явно указав в спецификации правила опцию **-m** (**--match**) с именем модуля netfilter в качестве параметра. Например, для проверки состояния соединения в спецификации правила должна присутствовать опция **-m state** до указания проверяемых условий. Отметим также, что для использования таких проверок может потребоваться загрузка модулей ядра.

5.1.9.5.1 Соответствие MAC-адресов

Условие

```
--mac-source <MAC-адрес>
```

позволяет проверять соответствие MAC-адресов отправителей в заголовках канального уровня пакетов IP. Не следует забывать, что для использования этого условия в строке спецификации правила должно присутствовать также опция загрузки модуля

```
-m mac
```

Пример проверки соответствия MAC-адреса показан ниже

```
iptables -A INPUT -m mac --mac-source 00:00:00:00:00:01 -j DROP
```

Условие можно также использовать со знаком инверсии (!). Например, правило

```
iptables -A INPUT -m mac ! --mac-source 00:00:00:00:00:ff -j DROP
```

будет отбрасывать все пакеты, кроме тех, которые получены от устройства с MAC-адресом 00:00:00:00:00:ff.

Отметим, что для использования проверки по MAC-адресам должна быть включена опция ядра **MAC address match support** (стр. 75). Если для опции было выбрано значение M, потребуется также загрузить модуль **ipt_mac**.

5.1.9.5.2 Пороговые значения частоты совпадений

Опция **-m limit** позволяет задать пороговое значение частоты выполнения условий, по достижении которого выполняется заданная правилом операция. Такая возможность весьма полезна для организации записи о событиях в системные журналы с помощью операций **LOG** (стр. 112) и **ULOG** (стр. 116). Вы можете делать запись в журнал не для каждого случая совпадения с заданными условиями, а лишь при достижении определенного порога частоты таких событий. Это позволяет снизить размер журнальных файлов и сделать их более читаемыми.

Опция **-m limit**

может использоваться с параметрами

```
--limit <средняя частота>
```

и

```
--limit-burst <пиковое значение>
```

Первый параметр задает пороговую частоту событий (число событий в единицу времени) и указывается в формате **значение/суффикс**. Значение определяет число событий, а суффикс - единицу времени (/s или /second- секунда, /m или /minute- минута, /h или /hour - час, /d или /day - сутки). По умолчанию используется пороговая частота 3 пакета в час. Второй параметр определяет пик “разовой” доставки пакетов. По умолчанию для пика используется значение 5. Модуль работает следующим образом:

- условие считается выполненным, пока значение счетчика пакетов не превысит пика **limit-burst**;
- каждый пакет, соответствующий правилу, увеличивает значение счетчика на 1;
- по истечении каждого интервала 1/**limit** значение счетчика уменьшается на 1.

Проиллюстрировать работу этого условия проще всего на примере ведра с отверстием в дне. Параметр **limit-burst** задает объем ведра (количество помещающихся в него пакетов), а параметр **limit** определяет скорость вытекания через отверстие в дне. Пока в ведре есть место, правило выполняется, а как только ведро наполнится, пакеты перестанут соответствовать правилу (польется через край). Очевидно, что в пустое ведро может поместиться **limit-burst** пакетов, а за счет вытекания через отверстие в дне число пакетов в ведре уменьшается на **limit** в единицу времени. Следовательно за это время в ведро можно поместить до **limit** новых пакетов. Если пакеты не приходят, ведро постепенно опустошается и может принять в себя больший объем.

Очевидно, что в результате выполнение условия имеет определенный гистерезис по частоте доставки пакетов, т. е., частота доставки пакетов при которой условие “включается”, будет отличаться от частоты, при которой условие будет “выключено”. Рассмотрим работу модуля на примере правила

```
iptables -A FORWARD -m limit -j LOG
```

с принятыми по умолчанию значениями параметров. Информация о первых пяти пакетах, переданных правилу, будет записана в журнальный файл, поскольку значение счетчика не достигло пика. После этого, в течение 20 минут (1/3 часа) записи производиться не будут, независимо от частоты поступления пакетов. В конце этого интервала значение счетчика уменьшится на 1 и в течение следующего 20-минутного интервала в журнальный файл может быть записан еще один пакет (т. е., будет записываться в информация о первом пакете в течение каждой 20-минутки). Счетчик пикового значения уменьшается на 1 каждые 20 (60/30) минут при отсутствии пакетов, поэтому, если в течение 100 минут (60/3 * 5) не поступит ни одного пакета, счетчик пикового значения достигнет нуля и процесс начнется заново.

Отметим, что в текущей версии не могут обрабатываться интервалы времени более 59 часов, поэтому при выборе частоты 1 пакет в сутки, значение пика не может быть меньше 3.

1 *Explicit match.*

Вы можете использовать этот модуль для защиты от атак на службы (DoS).

Защита от атак Syn-flood:

```
iptables -A FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT
```

Защита от хитроумных сканеров портов:

```
iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT
```

Защита от Ping of death:

```
iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT
```

Для использования этого условия в ядре должна быть включена поддержка опции **limit match support** (стр. 75). Если для опции выбрано значение M, потребуется загрузка модуля **ipt_limit**.

5.1.9.5.3 Соответствие маркеров

Модуль **mark** используется для проверки соответствия маркеров, установленных для пакета. Маркеры записываются функциями ядра Linux в специальное поле буферов skb (стр. 402) и могут использоваться в течение всего времени существования этого буфера (пакета) на данном компьютере. В современных вариантах Linux поддерживается единственный способ установки маркеров - операция **MARK** (стр. 112) в **iptables**. В более ранних версиях (ядра серии 2.2) для этих целей использовалась операция **FWMARK** в **ipchains**, поэтому в литературе можно встретить достаточно много упоминаний об этой операции. Маркер трактуется как беззнаковое целое число и в 32-битовых системах может принимать 4294967296 значений. Такая широта диапазона значений маркеров открывает широкий простор для их использования в целях фильтрации или управления потоками трафика.

Условие

```
--mark <маркер>
```

позволяет проверить соответствие маркера в данном пакете заданному значению и выполнить по результатам проверки ту или иную операцию. Для работы с условием **--mark** не забывайте указывать в спецификации правила опцию загрузки модуля **-m mark**. При указании значения маркера для проверки можно использовать маски, записывая параметр в формате

```
--mark <маркер>/<маска>
```

В этом случае для сравнения со значением маркера в пакете используется значение (**<маркер> AND <маска>**).

Для работы с этим условием в ядре должна быть включена опция **netfilter MARK match support** (стр. 76). Если для опции было выбрано значение M, потребуется загрузка модуля **ipt_mark**.

5.1.9.5.4 Соответствие владельца пакета

Модуль **owner** позволяет проверять пакеты по идентификаторам их владельцев. Для указания владельца можно использовать идентификатор процесса, сессии, группы или отдельного пользователя. Поскольку данные о владельце являются локальными, этот модуль можно использовать только в цепочках таблицы OUTPUT (стр. 104). Отметим, что и для локально сгенерированных пакетов информация о владельце доступна не всегда. Многие пакеты, среди которых отклики ICMP, не содержат сведений о владельце - для таких пакетов заданное условие никогда не будет выполняться.

Работая с модулем **owner**, не забывайте указывать в спецификации правила опцию

```
-m owner
```

Для использования этого модуля в ядре должна быть включена опция **Owner match support** (стр. 77). Если при компиляции ядра для опции было выбрано значение M, потребуется также загрузка модуля **ipt_owner**.

5.1.9.5.4.1 Идентификатор пользователя

Условие --**uid-owner** позволяет отбирать пакеты по идентификатору пользователя (ГШВ), запустившего приложение, которое сгенерировало данный пакет. Например правило

```
iptables -A OUTPUT -m owner --uid-owner 500
```

будет отбирать все пакеты, сгенерированные приложениями, которые работают от имени пользователя с идентификатором 500. Таким образом вы сможете заблокировать организацию исходящих соединений с брандмауэра от того или иного пользователя (например, разрешить такие соединения только для пользователя root).

5.1.9.5.4.2 Идентификатор группы

Условие --**gid-owner** позволяет отбирать пакеты по идентификатору группы (GID), к которой относится запустивший приложение пользователь. Например, условию

```
iptables -A OUTPUT -m owner --gid-owner 0
```

будут соответствовать только пакеты, сгенерированные приложениями пользователей из группы 0 (root). Это позволяет разрешать или запрещать организацию исходящих соединений с брандмауэра для отдельных групп пользователей.

5.1.9.5.4.3 Идентификатор процесса

Условие --**pid-owner** позволяет проверить принадлежность пакета к указанному идентификатором (PID) процессу. Правилу

```
iptables -A OUTPUT -m owner --pid-owner 73
```

будут соответствовать только пакеты, сгенерированные процессом с PID=73. Идентификаторы процессов могут достаточно часто меняться, что усложняет использование этого условия. Однако ничто не мешает собирать сведения об идентификаторах процессов в системе с помощью команды **ps**. Приведенный ниже пример обеспечивает беспрепятственную передачу пакетов, сгенерированных процессом **xinetd**.

```
PID=`ps aux |grep xinetd |head -n 1 |cut -b 10-14`

/usr/local/sbin/iptables -A OUTPUT -p TCP -m owner --pid-owner $PID -j ACCEPT
```

5.1.9.5.4.4 Идентификатор сессии

Условие --**sid-owner** позволяет проверить принадлежность пакета к указанной идентификатором (SID) сессии. Идентификатор сессии относится к процессу и всем порожденным им процессам, Правилу

```
iptables -A OUTPUT -m owner --pid-owner 120
```

будут соответствовать все пакеты, сгенерированные процессами с идентификатором сессии SID=120. Идентификаторы сессий могут достаточно часто меняться, что усложняет использование этого условия. Однако ничто не мешает собирать сведения об идентификаторах процессов в системе с помощью команды **ps**. Приведенный ниже пример обеспечивает беспрепятственную передачу пакетов, сгенерированных web-сервером Apache и всеми порожденными им процессами

```
SID=`ps -eo sid,args |grep httpd |head -n 1 |cut -b 1-5`

/usr/local/sbin/iptables -A OUTPUT -p TCP -m owner --sid-owner $SID -j ACCEPT
```

5.1.9.5.5 Соответствие состояния соединения

Опция соответствия **--state** используется совместно с модулями контроля состояния соединений в ядре Linux. Проверка состояния соединения для пакетов осуществляется с использованием данных машины состояний (см. параграф 5.1.6.5 на стр. 105) ядра. Это позволяет связывать пакет с тем или иным потоком трафика практически для всех протоколов, включая те, которые не используют явных соединений (stateless protocol) - например, ICMP и UDP. Для всех соединений существует время бездействия (тайм-аут), по истечении которого сведения об это соединении удаляются из таблицы контроля соединений. Использование использования соответствия --state включается с помощью команды **-m state** в строке спецификации правила.

В настоящее время netfilter различает 4 состояния соединения - **INVALID**, **ESTABLISHED**, **NEW** и **RELATED** (см параграф 5.1.6.5 на стр. 105). Состояние **INVALID** означает, что пакет не связан ни с одним из известных потоков или соединений и не является попыткой организации нового соединения. Такой пакет может появиться в результате ошибки в данных или заголовке. Состояние **ESTABLISHED** означает, что пакет относится к известному соединению, в котором существует двухсторонний обмен данными. Состояние **NEW** констатирует попытку организации нового соединения или говорит о том, что пакет связан с соединением в котором пока не организован двухсторонний обмен данными. Состояние **RELATED** означает, что пакет инициирует новое соединение, связанное с одним из уже открытых соединений. Например, это может быть передача данных по протоколу FTP или сообщение ICMP об ошибке в одном из имеющихся соединений TCP или UDP. Пример использования условия показан ниже

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED
```

Для работы с модулем state при компиляции ядра должна быть включена опция **Connection state match support** (стр. 77). Если для опции было выбрано значение M, потребуется также загрузить модуль **ipt_state**.

Дополнительные функции контроля состояния соединений обеспечиваются при использовании модуля conntrack (стр. 129)

5.1.9.5.6 Соответствия для множества портов

Кроме проверки соответствия для отдельных портов и диапазонов портов, существуют два модуля **multiport** и **mport**, обеспечивающие, соответственно, проверку соответствия группы отдельных портов и группе отдельных портов и диапазонов.

5.1.9.5.6.1 Соответствие для группы портов

При задании для протоколов TCP и UDP группы портов разделенных промежутками, используется модуль

```
-m multiport
```

позволяющий задать множество отдельных портов через запятую. Модуль -m multiport может использоваться с 3 опциями

```
-m multiport --source-port 22,53,80,110
-m multiport --destination-port 22,53,80,110
-m multiport --port 22,53,80,110
```

для задания группы номеров порта отправителя, получателя или отправителя и получателя, соответственно. В последнем случае, когда задаются номера портов для отправителя и получателя с помощью одного условия --port, для выполнения условия требуется дополнительно совпадение номеров портов для отправителя и получателя[1].

В условиях для группы портов может использоваться до 15 значений, задающих номера портов.

Отметим, что использование в одной строке соответствия для порта (диапазона) и группы портов не допускается. Такие условия просто не будут работать. Ниже приведен пример некорректного использования условий

1 *Приведенному правилу будут соответствовать пакеты, направленные из порта 22 в порт 22, из порта 53 в порт 53 и т. п., но не будут соответствовать пакеты, отправленные из порта 80 в порт 110 и т. п.*

```
--sport 1024:63353 -m multiport --dport 21,23,80
```
Результат трактовки таких некорректно заданных условий зависит от конкретной формы спецификации условий, но в любом случае не следует ожидать от подобного правила корректной работы.

Для использования этого условия требуется ядро с поддержкой опции **Multiple port match support** (стр. 76). Если для опции было выбрано значение M, потребуется также загрузка модуля **ipt_multiport**.

5.1.9.5.6.2 Соответствие mport

Модуль **mport** похож на описанный выше модуль **multiport**, но в отличие от того позволяет в одном параметре указывать отдельные порты и диапазоны портов. Например, правило

```
iptables -A INPUT -p tcp -m mport --ports 20:23,80 -j DROP
```
будет отбрасывать все пакеты, адресованные в порты с 20 по 23, включительно, и порт 80.

Отметим, что для работы с модулем **mport** потребуется включить поддержку этой функции при компиляции iptables[1].

5.1.9.5.7 Соответствие типа обслуживания (TOS)

Модуль **tos** служит для проверки соответствия битов поля типа обслуживания (Type Of Service) в заголовках IP. Для использования модуля в строке правила нужно явно указать **-m tos**.

Условие проверки битов TOS имеет вид

```
--tos <значение>
```
где параметр <значение> задает шестнадцатеричное (в формате 0xNN) или символьное представление проверяемых битов TOS. Для получения списка поддерживаемых символьных имен типов обслуживания можно воспользоваться командой

```
iptables -m tos -h
```
Поле TOS обычно используется для информирования промежуточных хостов об уровнях предпочтения для данного потока и его содержимого. Никто не обязан удовлетворять эти запросы, но они могут быть приняты во внимание промежуточными маршрутизаторами при обработке пакетов.

На основании проверки битов TOS можно устанавливать для пакетов маркеры (см. описание операции **MARK** на стр. 112), которые могут быть использоваться впоследствии при обработке пакетов на данном хосте.

Для использования модуля tos требуется ядро с поддержкой опции **TOS match support** (стр. 76). Если для опции было выбрано значение M, потребуется также загрузка модуля **ipt_tos**.

5.1.9.5.8 Соответствие времени жизни (TTL)

Модуль **ttl** позволяет отбирать пакеты по значению поля времени жизни (**TTL**) в заголовках IP. Для проверки значения времени жизни служит условие

```
--ttl <TTL>
```
Значение TTL может лежать в интервале от 0 до 255 и каждый маршрутизатор на пути доставки пакета уменьшает значение этого поля на 1. При достижении нулевого значения пакет отбрасывается, а его отправителю передается сообщение ICMP типа 11 с кодом 0 (значение TTL достигло 0 в процессе доставки пакета) или 1 (значение TTL достигло 0 при сборке фрагментов).

Описываемое здесь условие может использоваться только для проверки значения TTL, а для его изменения служит специальная операция TTL (стр.). Для проверки TTL не забывайте указывать в строке правила команду загрузки модуля **-m ttl**.

Для использования проверки TTL в ядре должна быть включена поддержка опции **TTL match support** (стр. 77). Если для опции было выбрано значение M, потребуется также загрузить модуль **ipt_ttl**.

5.1.9.5.9 Соответствие ah

Модуль **ah** служит для проверки значений SPI[2] в заголовках AH[3] пакетов IPSec[4]. Для использования модуля в строке спецификации следует явно указывать команду его загрузки **-m ah**. Условие

```
-m ah [!] --ahspi spi1[:spi2]
```
позволяет проверить соответствие поля SPI в заголовке пакета значению spi1 или диапазону значений spi1 - spi2.

Для использования этого модуля в ядре должна быть включена опция **AH/ESP match support** (стр. 76). При выборе для опции значения M потребуется также загрузить модуль **ipt_ah**.

5.1.9.5.10 Соответствие esp

Модуль **esp** служит для проверки значений SPI в заголовках ESP[5] пакетов IPSec[6]. Для использования модуля в

1 *В бинарных пакетах iptables эта функция зачастую отключена.*
2 *Security Parameter Index - индекс параметра безопасности*
3 *Authentication Header - заголовок аутентификации.*
4 *Протокол IP Authentification Header описан в RFC 2402 (http://rfc-editor.org/rfc/rfc2402.txt), копию которого вы найдете в каьалоге Documents/ приложенного к курсу компакт-диска.*
5 *Encapsulating Security Payload - инкапсулированные данные безопасности*
6 *Протокол IP Encapsulating Security Payload описан в RFC 2406.*

строке спецификации следует явно указывать команду его загрузки **-m esp**. Условие

-m esp [!] --ahspi spi1[:spi2]

позволяет проверить соответствие поля SPI в заголовке пакета значению spi1 или диапазону значений spi1 - spi2.

Для использования этого модуля в ядре должна быть включена опция **AH/ESP match support** (стр. 76). При выборе для опции значения M потребуется также загрузить модуль **ipt_esp**.

5.1.9.5.11 Расширенное соответствие состояния соединений (conntrack)

Модуль **conntrack** обеспечивает поддержку расширенных функций контроля за состоянием соединений. По сути, этот модуль является дополнением к описанным выше соответствиям модуля state (стр. 127). Модуль поддерживает целый ряд условий:

--ctstate state

Проверяет соответствие пакета указанному состоянию или группе состояний. Параметр state представляет собой список разделенных запятыми состояний для которых проверяется соответствие. Состояние INVALID относится к пакетам, не связанным с известными соединениями и не являющихся попытками организации новых, ESTABLISHED указывает на состояние при котором соединение организовано и используется для двухстороннего обмена данными, NEW относится к пакетам, инициирующим новые соединения, а RELATED - к пакетам, инициирующим новые соединения, связанные с уже известными соединениями (например, передача данных по протоколу FTP или возврат сообщения ICMP об ошибке в соединении TCP или UDP)[1]. Виртуальному состоянию SNAT соответствуют пакеты, у которых исходный адрес отправителя отличается от адреса получателя в отклике, а виртуальному состоянию DNAT - пакеты, в которых исходный адрес получателя отличается от адреса отправителя в отклике.

--ctproto proto

Проверяет принадлежность пакета к указанному номером или символьным именем протоколу[2].

--ctorigsrc [!] address[/mask]

Проверяет соответствие исходного (нетранслированного) адреса отправителя указанному адресу или диапазону адресов.

--ctorigdst [!] address[/mask]

Проверяет соответствие исходного (нетранслированного) адреса получателя указанному адресу или диапазону адресов.

--ctreplsrc [!] address[/mask]

Проверяет соответствие адреса отправителя отклика указанному адресу или диапазону адресов.

--ctrepldst [!] address[/mask]

Проверяет соответствие адреса получателя отклика указанному адресу или диапазону адресов.

--ctstatus [NONE|EXPECTED|SEEN_REPLY|ASSURED][,...]

Проверяет соответствие пакета одному или группе внутренних состояний **conntrack**.

--ctexpire time[:time]

Проверяет соответствие оставшегося для пакета времени жизни заданному значению или диапазону.

Для использования этого модуля в ядре должна быть включена опция **Connection tracking match support** (стр. 77). Если для этой опции было при компиляции выбрано значение M, потребуется также загрузить модуль **ipt_conntrack**. Программа iptables должна быть скомпилирована с использованием поддерживающего опцию ядра.

5.1.9.5.12 Соответствие DSCP

Этот модуль позволяет задать условия проверки 6-битового значения DSCP[3] в поле TOS заголовка IP. В соответствии с RFC 2474 значение DSCP заменяет собой значение TOS. Для проверки значений DSCP в спецификации правила должна явно присутствовать команда загрузки модуля **-m dscp**. Условие

--dscp <значение>

позволяет проверить числовое значение поля DSCP. Параметр можно указывать в десятичном или шестнадцатеричном формате, а его значение должно лежать в диапазоне [0-32].

Условие

--dscp-class DiffServ

служит для проверки соответствия класса обслуживания DiffServ. Параметр может принимать значения классов BE, EF, AFxx или CSx.

Для использования модуля требуется ядро со включенной опцией **DSCP match support** (стр. 76). Если при компиляции ядра для опции было выбрано значение M, потребуется также загрузка модуля **ipt_dscp**.

5.1.9.5.13 Соответствие helper-модулей

Модуль **helper** служит для проверки соответствия указанному модулю conntrack-helper. Для работы с модулем в строке спецификации правила должна присутствовать команда **-m helper**. Условие

--helper <имя conntrack-helper>

1 *Более подробное описание возможных состояний соединений вы найдете в параграфе 5.1.6.5.1 (стр. 106).*
2 *Имена и номера сетевых протоколов вы сможете найти в файле /etc/protocols вашего Linux-компьютера.*
3 *DSCP (differentiated services code point) - шестибитовое значение в поле TOS, позволяющее запросить при доставкке пакетов желаемый тип обслуживания (QoS).*

проверяет соответствие связанного с пакетом модуля conntrack-helper указанному параметром значению. Параметр содержит имя helper-модуля, если проверка осуществляется для используемых по умолчанию портов. Если же вы хотите проверить соответствие для соединений с другими портами, укажите номер порта вслед за именем модуля через дефис. Например, для пакетов связанных с FTP-сессиями можно записать параметр в форме **ftp-2121**.

Для ядра версии 2.6.6 поддерживаются helper-модули для протоколов TCP (модуль ip_conntrack_tcp, стр.), UDP (модуль ip_conntrack_udp, стр.), ICMP (модуль ip_conntrack_icmp, стр.), Amanda (модуль ip_conntrack_amanda, стр. 75), FTP (модуль ip_conntrack_ftp, стр.), IRC (модуль ip_conntrack_irc, стр.), TFTP (модуль ip_conntrack_tftp, стр.), а также базовый модуль контроля соединений (ip_conntrack, стр.).

Для использования этого модуля требуется ядро со включенными опциями **Connection tracking** (стр. 74), **Helper match support** (стр. 77) и опциями поддержки используемых helper-модулей. При выборе для той или иной из этих опций значения M потребуется также загрузка соответствующего модуля.

5.1.9.5.14 Соответствие размера пакетов (length)

Модуль **length** позволяет проверять размеры пакетов. При использовании модуля в спецификации правила должна быть указана команда загрузки этого модуля **-m length**. Условие

`--length length[:length]`

позволяет проверить соответствие размера пакета заданному значению или диапазону значений.

Для работы с этим модулем требуется ядро со включенной опцией **LENGTH match support** (стр. 76). Если при компиляции ядра для опции было выбрано значение M, потребуется также загрузка модуля **ipt_length**.

5.1.9.5.15 Соответствие физического устройства (physdev)

Этот модуль позволяет проверять для пакетов соответствие входных и выходных портов моста. Модуль является частью инфраструктуры, обеспечивающих возможность организации прозрачных мостов с поддержкой функций брандмауэра IP и может использоваться с ядрами, начиная с версии 2.5.44. Для использования модуля в строке спецификации правила должна присутствовать команда **-m physdev**. Условие

`--physdev-in <имя устройства>`

позволяет проверить для пакета имя принявшего его интерфейса (только для цепочек INPUT, FORWARD и PREROUTING). Знак **+** в конце имени устройства обозначает шаблон имен, которому соответствуют все устройства, начала имени которых (все, что слева от знака +) совпадает со значением параметра. Условие можно использовать со знаком инверсии **!**.

Другое условие

`--physdev-out <имя устройства>`

позволяет проверить для пакета соответствие выходного интерфейса (только для цепочек FORWARD, OUTPUT и POSTROUTING). Отметим, что для цепочек OUTPUT в таблицах **nat** и **mangle** проверка выходного интерфейса возможна не всегда, однако такая проверка работает для цепочек OUTPUT таблицы **filter**. Если пакет не будет уходить с моста или на момент проверки непонятно, через какой интерфейс уйдет пакет, условие не выполняется, если в правиле не задан один из дополнительных параметров, обеспечивающих выполнение условия

`--physdev-is-in`

если пакет получен через интерфейс моста;

`--physdev-is-out`

если пакет будут передан через интерфейс моста;

`--physdev-is-bridged`

если для пакета используются функции моста, а не маршрутизатора (этот параметр может быть полезен только в цепочках FORWARD и POSTROUTING).

Для работы с этим модулем требуется ядро, поддерживающее опцию **Physdev match support** (стр. 77). Если при компиляции ядра для опции было выбрано значение M, потребуется также загрузка модуля **ipt_physdev**.

5.1.9.5.16 Соответствие типа пакетов (pkttype)

Этот модуль позволяет проверять соответствие типа пакетов на канальном уровне. Для работы с модулем в строке спецификации правила не забывайте указывать команду **-m pkt-type**. Условие

`--pkt-type [unicast|broadcast|multicast]`

позволяет различать пакеты с индивидуальной и групповой адресацией, а также широковещательные пакеты.

Для использования этого условия в модуле должна быть включена опция **Packet type match support** (стр. 75). Если для опции было выбрано значение M, потребуется также загрузить модуль **ipt_pkttype**.

5.1.9.5.17 Соответствие типа адреса (addrtype)

Этот модуль позволяет проверять соответствие типа адреса получателя и отправителя. Типы адресов используются стеком протоколов ядра и набор возможных типов зависит от используемого протокола сетевого уровня. Для сокета netlink (см. Приложение) в идентификаторы типа адресов определены в файле **<linux/netlink.h>**.

Таблица 11: Типы адресов сетевого уровня

Тип	Описание
UNSPEC	Адрес не указан (0.0.0.0).
UNICAST	Индивидуальный адрес IP (адрес шлюза или прямой маршрут в получателю)
LOCAL	Локальный адрес, принадлежащий одному из хостов, подключенных к маршрутизатору сетей.
BROADCAST	Адрес, который воспринимается и пересылается как широковещательный.
ANYCAST	Пакет, адресованный всем - воспринимается как широковещательный, но пересылается как UNICAST.
MULTICAST	Групповой адрес
BLACKHOLE	Пакет, адресованный в “черную дыру” (DROP).
UNREACHABLE	Недоступный адрес.
PROHIBIT	Запрещенный адрес.
THROW	Адрес отсутствует в таблице.
NAT	Адрес должен транслироваться (NAT).
XRESOLVE	Нужно использовать внешние средства определения типа (external resolver).

Опция

--src-type <тип>

используется для проверки типа адреса отправителя

--dst-type <тип>

служит для проверки типа адреса получателя.

При использовании этого модуля не забывайте включать в спецификацию правила команду загрузки модуля **-m addrtype**. Для использования модуля при компиляции ядра должна быть включена опция **address type match support** (параграф 4.4.2.2.14.2.22 на стр. 81). Если при компиляции ядра для опции было выбрано значение **M**, потребуется также загрузить модуль ядра **ipt_addrtype**.

5.1.9.5.18 Соответствие recent

Модуль **recent** позволяет создавать динамические списки адресов IP для недавних пакетов и проверять соответствие адресов в полученных пакетах этим спискам. Не забывайте указывать в спецификации правила команду **-m recent** для работы с этим модулем. Модуль поддерживает ряд опций.

--name name

указывает имя списка, с которым работает команда. По умолчанию для списка используется имя DEFAULT.

[!] --set

эта команда служит для добавления адреса отправителя пакета в список. Если такой адрес уже есть в списке, запись будет обновлена. Команда возвращает позитивный результат (**success**), а в случае использования инверсии (**!**) - негативный (failure).

[!] --rcheck

эта опция будет проверять наличие адреса отправителя пакета в списке и возвращать **true** при его наличии. Знак инверсии (**!**) меняет возвращаемый результат на обратный.

[!] --update

эта опция будет проверять наличие адреса отправителя пакета в списке. Если адрес присутствует, соответствующая запись обновляется и возвращается значение **true**. При отсутствии адреса в списке возвращается значение **false**. Знак инверсии (**!**) меняет возвращаемый результат на обратный.

[!] --remove

эта опция будет проверять наличие адреса отправителя в списке и, при обнаружении, удалять удалять соответствующую запись, возвращая значение **true**. При отсутствии адреса в списке возвращается значение **false**. Знак инверсии (**!**) меняет возвращаемый результат на обратный.

[!] --seconds seconds

эта опция может использоваться вместе с опцией **rcheck** или **update**, задавая период (отсчет от текущего момента), в течение которого искомый адрес может находиться в списке (возраст записи). Знак инверсии (**!**) меняет значение опции на обратное (поиск записей, находящихся в списке более указанного параметром времени).

[!] --hitcount hits

эта опция может использоваться вместе с опцией **rcheck** или **update** - при наличии этого счетчика указанные команды возвращают позитивный результат при наличии адреса в списке только в том случае, когда число принятых пакетов не меньше заданного значения. Опцию можно использовать совместно с **seconds** для дополнительного ужесточения условий соответствия (не менее заданного числа пакетов в заданный интервал времени. Знак инверсии (**!**) меняет значение опции на обратное.

--rttl

при использовании этой опции условие выполняется, если адрес присутствует в списке и значение поля TTL в текущем пакет совпадает со значением этого поля в пакете, для которого была использована опция **--set**. Это

может быть полезно для предотвращения организации DoS-атак с использованием функций данного модуля[1] и подставных адресов.

В качестве примера использования модуля рассмотрим блокировку попыток доступа к вашему брандмауэру через порт 139. Создание списка badguy позволяет в дальнейшем отбрасывать пакеты с этих адресов без дополнительных проверок.

```
iptables -A FORWARD -m
recent --name badguy --
rcheck --seconds 60 -j
DROP
iptables -A FORWARD -p tcp
-i eth0 --dport 139 -m
recent --name badguy --set
-j DROP
```

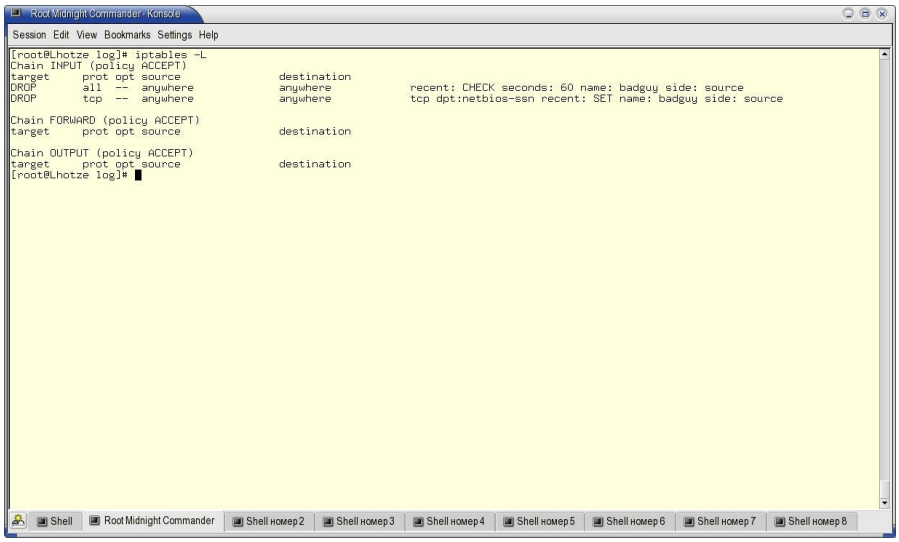


Рисунок 5.2. Соответствие recent.

Если после этого вы введете команду **iptables -L**, то увидите (5.2), что пакеты с адресов, откуда были запросы к порту netbios-ssn в течение последней минуты будут блокироваться первым правилом без дополнительной проверки.

Для использования этого модуля требуется ядро со включенной опцией **recent match support** (стр. 76). Если для опции было выбрано значение M, потребуется также загрузка модуля **ipt_recent**.

5.1.9.5.19 Соответствие ECN

Модуль ecn используется для проверки соответствия значения поля ECN[2] в заголовках TCP. Для использования модуля в строке спецификации правила должна присутствовать команда **-m ecn**. Модуль поддерживает проверку нескольких условий

[!] --ecn-tcp-cwr
проверяет бит CWR[3] в заголовках пакетов TCP;

[!] --ecn-tcp-ece
проверяет бит ECE[4] в заголовках пакетов TCP;

[!] --ecn-ip-ect [0..3]
проверяет код ECN в заголовке IPv4.

Для использования этого модуля в ядре должна быть включена опция **ECN match support** (стр. 76). Если для опции было выбрано значение M, потребуется также загрузить модуль **ipt_ecn**.

5.1.9.5.20 Соответствие unclean

Условие **unclean** не имеет никаких параметров и служит лишь для попытки удаления некорректно форматированных и необычных пакетов. Пакеты, обнаруженные с помощью проверки этого условия можно отбрасывать или записывать в системные журналы.

Это условие является пока экспериментальным и его не следует использовать в реальных межсетевых экранах.

5.1.9.6 Дополнительные соответствия

Кроме описанных выше условий соответствия, поддерживаемых современной версией iptables[5], существует ряд дополнительных условий соответствия, которые могут не поддерживаться версиями iptables, установленными из бинарных файлов. Чтобы сделать доступным то или иное условие, скомпилируйте iptables самостоятельно из исходных текстов, которые можно загрузить с сайта http://www.iptables.org[6]. В некоторых случаях может также потребоваться загрузка исходного кода дополнительных модулей, которые можно найти на сайте http://www.netfilter.org/patch-o-matic/.

5.1.9.6.1 Соответствие condition

Этот модуль позволяет проверять соответствие пакетов условиям, записанным в переменные, которые хранятся в файлах /proc[7]. Для работы с функциями этого модуля в строке спецификации правил должна присутствовать команда **-m condition**.

1 *Злоумышленник теоретически может заблокировать пакеты от легитимных пользователей, за счет включения в список адресов из подставных пакетов. Контроль значения TTL снижает шансы на успех такой блокировки.*
2 *Explicit Congestion Notification - явное уведомление о насыщении.*
3 *Congestion Window Reduced - уменьшено окно насыщения.*
4 *ECN echo.*
5 *iptables v1.2.9*
6 *Исходные коды iptables v1.2.9 имеются в каталоге SRC/ приложенного к курсу компакт-диска.*

К поддерживаемым этим модулем условиям предъявляется ряд требований:

- условия должны храниться в файлах каталога /proc/net/ipt_condition/;
- переменные условий могут принимать только логические значения FALSE (0) и TRUE (1);
- каждая переменная может влиять на состояние одного или нескольких условий;
- файл условий создается автоматически при первом упоминании нового условия;
- файл условий автоматически удаляется после удаления последней ссылки на соответствующее условие.

Модуль использует единственную опцию

```
--condition [!] conditionfile
```

которая позволяет проверить соответствие условий для пакета переменной в файле conditionfile. Например, для запрета доступа к web-серверу на время его обслуживания можно воспользоваться командой

```
iptables -A FORWARD -p tcp -d 192.168.1.10 --dport http -m condition --condition webdown
-j REJECT --reject-with tcp-reset
```

указав значение TRUE в файле условия с помощью команды

```
echo 1 > /proc/net/ipt_condition/webdown
```

Пока в файле /proc/net/ipt_condition/webdown будет записано значение 1, доступ к web-серверу будет заблокирован.

5.1.9.6.2 Соответствие fuzzy

Модуль **fuzzy** позволяет проверять соответствие пакетов динамическим профилям, реализованным с помощью простого контроллера FLC (Fuzzy Logic Controller).

Это соответствие проверяется на основе метода TSK FLC (Takagi-Sugeno-Kang Fuzzy Logic Controller), определяющего результат проверки в зависимости от скорости доставки пакетов с учетом значений нижнего и верхнего порога скорости; указанных правилом:

- пока скорость доставки пакетов не достигает нижнего порога, условие никогда не будет выполняться;
- при скорости в диапазоне между нижним и верхним порогами частота выполнения условий будет расти пропорционально скорости доставки пакетов;
- после превышения верхнего порога скорости частота выполнения условий достигнет максимального значения - 99%.

При использовании модуля в строке спецификации должна явно присутствовать команда его загрузки

```
-m fuzzy
```

Модуль поддерживает 2 параметра, определяющих верхний и нижний пределы скорости

```
--lower-limit n
```

нижний предел по достижении которого условие начинает выполняться с вероятностью, пропорциональной скорости получения пакетов;

```
--upper-limit n
```

верхний предел, после превышения условие выполняется с вероятностью 99%.

С учетом условий работы современных сетей и достаточно высокой загрузки межсетевого экрана приведенное ниже правило должно хорошо работать в условиях атак на службы (Denials Of Service).

```
iptables -A INPUT -m fuzzy --lower-limit 100 --upper-limit 1000 -j REJECT
```

Правило будет работать следующим образом

- при скорости менее 100 пак/с условие не будет выполняться и пакеты будут беспрепятственно приниматься хостом;
- при скоростях от 100 до 1000 пак/с вероятность принятия пакетов будет снижаться от 100% (при скорости 100 пак/с) до 1% (при скорости 1000 пак/с).
- при скоростях более 1000 хост будет принимать только 1% адресованных ему пакетов.

5.1.9.6.3 Соответствие iplimit

Модуль **iplimit** позволяет ограничить число одновременных соединений TCP для одного хоста или сети. Для работы с модулем спецификация правила должна содержать команду **-m iplimit**. Модуль поддерживает 2 опции:

```
[!] --iplimit-above n
```

задает порог числа одновременных соединений TCP. Опция может использоваться со знаком инверсии.

```
--iplimit-mask n
```

указывает маску для группы хостов (подсети).

Например, если вы хотите каждому клиенту ограничить до 4 число одновременных соединений HTTP с данным хостом, можно использовать правило:

```
iptables -A INPUT -p tcp --syn --dport http -m iplimit --iplimit-above 4 -j REJECT
```

Если после ввода правила воспользоваться командой **iptables -L**, на экран будет выведено:

```
Chain INPUT (policy ACCEPT)
target   prot opt source    destination
REJECT   tcp  --  anywhere  anywhere     tcp dpt:http flags:SYN,RST,ACK/SYN #conn/32 > 4
reject-with icmp-port-unreachable
```

7 *Для поддержки виртуальной фаловой системы требуется включить опцию **/proc file system support** (стр. 67) при компиляции ядра.*

Обратите внимание, что ни в команде спецификации правила, ни в выводе результатов не присутствует адрес отправителя. Это правило действует для ограничения числа соединений **любого** хоста. Можно также ограничить число соединений для всех хостов сети, заданной значением маски. Например, команда:

```
iptables -A INPUT -p tcp --syn --dport http -m iplimit --iplimit-mask 8 --iplimit-above 4
-j REJECT
```

будет позволять не более 4 соединений с портом HTTP для всех хостов **любой** сети класса A. Результат использования этой команды показан ниже:

```
Chain INPUT (policy ACCEPT)
target   prot opt source    destination
REJECT   tcp  --  anywhere  anywhere     tcp dpt:http flags:SYN,RST,ACK/SYN #conn/8 > 4
reject-with icmp-port-unreachable
```

5.1.9.6.4 Соответствие ipv4options

Модуль ipv4options позволяет проверять соответствие пакетов заданному набору опций IP. Для работы с модулем в спецификации правила должна присутствовать команда загрузки модуля **-m ipv4options**. Модуль поддерживает несколько опций:

--ssrr
проверяет соответствие флага strict source routing;

--lsrr
проверяет соответствие флага loose source routing;

--no-srr
проверяет отсутствие флагов source routing;

[!] --rr
проверяет наличие или отсутствие флага record route;

[!] --ts
проверяет наличие или отсутствие флага timestamp;

[!] --ra
проверяет значение опции router-alert;

[!] --any-opt
проверяет наличие или отсутствие в заголовке IP хотя бы одной опции.

Например, для отбрасывания всех пакетов с установленными опциями record-route и timestamp можно использовать команды:

```
iptables -A INPUT -m ipv4options --rr -j DROP
iptables -A INPUT -m ipv4options --ts -j DROP
```

Если после этих команд посмотреть список правил с помощью команды -L, вы увидите:

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  anywhere             anywhere            IPV4OPTS RR
DROP       all  --  anywhere             anywhere            IPV4OPTS TS
```

5.1.9.6.5 Соответствие nth

Модуль **nth** позволяет создавать правила, которым будет соответствовать каждый N-й пакет, удовлетворяющий остальным условиям данного правила. Для работы с этим модулем в правилах должна присутствовать команда **-m nth**. Модуль поддерживает 4 опции:

--every N
указывает, что условий будет соответствовать каждый N-й пакет;

[--counter] num
задает номер используемого правилом счетчика. Модуль поддерживает до 16 счетчиков с номерами от 0 до 15. По умолчанию используется счетчик 0.

[--start] num
указывает начало отсчета для счетчика пакетов. Этот параметр может принимать значения от 0 до N-1;

[--packet] num
задает номер пакета, для которого выполняются условия. Параметр может принимать значения от 0 до N-1.

Например, для отбрасывания каждого второго запроса packets вы можете воспользоваться командой:

```
iptables -A INPUT -p icmp --icmp-type echo-request -m nth --every 2 -j DROP
```

Этот модуль может быть очень эффективен для распределения нагрузки между несколькими входящими или исходящими каналами, а также для распределения нагрузки на серверы. Например, для распределения трафика между 3 адресами 10.0.0.5, 10.0.0.6 и 10.0.0.7 можно воспользоваться правилами:

```
iptables -t nat -A POSTROUTING -o eth0 -m nth --counter 7 --every 3 --packet 0 -j SNAT --
to-source 10.0.0.5
iptables -t nat -A POSTROUTING -o eth0 -m nth --counter 7 --every 3 --packet 1 -j SNAT --
to-source 10.0.0.6
```

```
iptables -t nat -A POSTROUTING -o eth0 -m nth --counter 7 --every 3 --packet 2 -j SNAT --
to-source 10.0.0.7
```
Если вы после этого воспользуетесь командой iptables -L, на экран будут выведено

```
Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
SNAT       all  --  anywhere             anywhere           every 3th packet #0
to:10.0.0.5
SNAT       all  --  anywhere             anywhere           every 3th packet #1
to:10.0.0.6
SNAT       all  --  anywhere             anywhere           every 3th packet #2
to:10.0.0.7
```

5.1.9.6.6 Соответствие psd

Модуль psd обеспечивает поддержку функций обнаружения сканирования портов. Для использования модуля в строке спецификации правила должна присутствовать команда **-m psd**. Модуль поддерживает опции:

[--psd-weight-threshold threshold]
пороговый уровень “веса” при сканировании портов.

[--psd-delay-threshold delay]
пороговый уровень задержки при обнаружении попыток сканирования.

[--psd-lo-ports-weight lo]
“вес” для привилегированных портов.

[--psd-hi-ports-weight hi]
“вес” для портов со старшими номерами.

Если вы создадите правило:

```
iptables -A INPUT -m psd -j DROP
```
а потом воспользуетесь командой iptables -list, вы увидите в цепочке INPUT условия детектирования попыток сканирования портов:

```
Chain INPUT (policy ACCEPT)
target  prot opt source    destination
DROP    all  --  anywhere  anywhere    psd weight-threshold: 21 delay-threshold: 300 lo-
ports-weight: 3 hi-ports-weight: 1
```

5.1.9.6.7 Соответствие quota

Этот модуль позволяет задавать количественные ограничения (квоты), по достижении которых условие перестает выполняться. Для использования модуля в строке спецификации правила должна присутствовать команда **-m quota**. Модуль поддерживает единственную опцию:

--quota quota
которая задает количественное значение, по достижении которого условие перестает выполняться.

Команды

```
iptables -A INPUT -p tcp --dport 80 -m quota --quota 52428800 -j ACCEPT
iptables -A INPUT -p tcp --dport 80 -j DROP
```
позволяют ограничить входящий трафик HTTP значением 50 Мбайт. Если вы после этих команд воспользуетесь командой просмотра правил, то увидите цепочки

```
Chain INPUT (policy ACCEPT)
target     prot opt source          destination
ACCEPT     tcp  --  anywhere        anywhere           tcp dpt:http quota: 52428800 bytes
DROP       tcp  --  anywhere        anywhere           tcp dpt:http
```
Таким образом, весь входящий трафик HTTP будет отбрасываться после приема первых 50 Мбайт.

5.1.9.6.8 Соответствие random

Этот модуль позволяет устанавливать для пакетов соответствие на основе случайных (вероятностных) значений. Для работы с модулем спецификация правила должна содержать команду **-m random**. Модуль использует единственную опцию:

[--average percent]
определяющую процент пакетов, которые будут соответствовать данному правилу. Целочисленное значение опции может лежать в диапазоне от 1 до 99. По умолчанию уровень соответствия составляет 50%.

Например, для случайного отбрасывания половины пакетов ping можно воспользоваться правилом:

```
iptables -A INPUT -p icmp --icmp-type echo-request -m random --average 50 -j DROP
```
Если после этого вы воспользуетесь командой **iptables -list**, можно будет увидеть, что 50% случайно выбранных пакетов icmp echo-request будут отброшены

```
Chain INPUT (policy ACCEPT)
target     prot opt source       destination
DROP       icmp --  anywhere     anywhere        icmp echo-request  random 50%
```

5.1.9.6.9 Соответствие realm

Этот модуль позволяет проверять в пакетах значение маршрутного ключа realm. Для работы с модулем спецификация правила должна содержать команду **-m realm**. Модуль использует единственную опцию:

```
--realm [!] value[/mask]
-> Match realm
```

Например, для записи в журнал информации обо всех исходящих пакетах со значением realm = 10 вы можете использовать правило:

```
iptables -A OUTPUT -m realm --realm 10 -j LOG
```

Если вы после этого воспользуетесь командой **iptables -list**, то увидите в цепочке правило

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source          destination
LOG        all  --  anywhere        anywhere           REALM match 0xa LOG level warning
```

Для использования модуля при компиляции ядра должна быть включена опция (параграф 4.4.2.2.14.2.23 на стр. 81). Если для опции ядра было выбрано значение **M**, потребуется также загрузка модуля ядра **ipt_realm**.

5.1.9.6.10 Соответствие record_rpc

Модуль record_rpc позволяет проверить был ли порт запрошен ранее отправителем пакета с помощью portmapper или это новый запрос GET к portmapper. Использование модуля может обеспечить эффективную фильтрацию RPC[1]. Не забывайте указывать в строке спецификации правила команду загрузки модуля **-m record_rpc**.

Для записи информации о соединениях RPC достаточно использовать команду:

```
iptables -A INPUT -m record_rpc -j ACCEPT
```

Если вы после этого посмотрите список правил с помощью команды **iptables --list**

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
```

то увидите, что сам модуль record_rpc не выполняет по отношению к пакету никаких действий. Отсутствие вывода от данного модуля объясняется тем, что функция print() просто ничего не делает:

```
/* Prints out the union ipt_matchinfo. */
static void
print(const struct ipt_ip *ip,
      const struct ipt_entry_match *match,
      int numeric)
{
}
```

5.1.9.6.11 Соответствие string

Этот модуль позволяет находить текстовые строки в любом месте пакетов(контекстный поиск). Для работы с модулем в строке спецификации правила следует указывать команду **-m** string. Модуль использует единственную опцию:

```
--string [!] string
```

для проверки наличия в пакете заданного фрагмента текста.

Например, для обнаружения пакетов, содержащих текст **cmd.exe** с целью их передачи в очередь системы IDS вы можете воспользоваться командой:

```
iptables -A INPUT -m string --string 'cmd.exe' -j QUEUE
```

При использовании этого модуля следует соблюдать осторожность. Многим начинающим администраторам может показаться целесообразным использование этого модуля для блокирования известных червей и вирусов по их сигнатурам, задавая в качестве действия правила операцию DROP. Однако в этом случае пакеты не будут передаваться системе IDS.

Другой ошибкой при использовании этого модуля является попытка предотвратить использование некоторых функций HTTP (например, POST или GET) путем отбрасывания пакетов, содержащих соответствующую строку. Отметим, что с этой работой гораздо лучше справляются фильтрующие прокси-серверы. Кроме того, при использовании такого подхода будут заблокированы все пакеты HTTP, которые содержат документы с данным словом.

Этот модуль предназначен только для передачи пакетов программам пользовательского пространства для дальнейшего анализа этих пакетов. Отбрасывание пакетов с помощью данного модуля будет просто приводить к обходу систем IDS.

5.1.9.6.12 Соответствие time

Модуль **time** позволяет проверить для пакетов время прибытия или отправки[2]. Для использования модуля в строке спецификации правила должна присутствовать команда **-m time**. Модуль поддерживает несколько опций:

```
--timestart value
```

1 *Remote Procedure Call - удаленный вызов процедур.*
2 *Время отправки проверяется только для локально сгенерированных пакетов.*

проверяет, что значение временной метки не меньше указанного времени (HH:MM)

```
--timestop value
```

проверяет, что временная метка не превышает указанного значения (HH:MM)

```
--days listofdays
```

проверяет для временной метки соответствие заданному дню недели

- Mon
- Tue
- Wed
- Thu
- Fri
- Sat
- Sun

При указании дня недели следует соблюдать регистр символов.

Приведенная ниже команда обеспечит восприятие всех входящих пакетов по рабочим дням с 8:00 до 18:00:

```
iptables -A INPUT -m time --timestart 8:00 --timestop 18:00 --days Mon,Tue,Wed,Thu,Fri -j
ACCEPT
```

Если вы воспользуетесь командой **iptables --list**, то сможете увидеть включенное в цепочку правило фильтрации

```
Chain INPUT (policy ACCEPT)
target     prot opt source     destination
ACCEPT     all  --  anywhere   anywhere        TIME from 8:0 to 18:0 on
Mon,Tue,Wed,Thu,Fri
```

5.1.9.6.13 Соответствие u32

Разработанный Доном Коэном (Don Cohen) модуль **u32** позволяет взять из пакета **любые байты**, выполнить по отношению к ним те или иные действия, и проверить результат. Например, вы можете взять из заголовка пакета IP поле Fragmentation, отбросить в нем все флаги, за исключением More Fragments и посмотреть значение этого флага.

Модуль позволяет выполнить множество операций и хорошо документирован, поэтому вы сможете самостоятельно создавать правила на основе этого модуля, прочитав последующие параграфы и предварительно поэкспериментировав. Для работы с модулем вы должны хорошо знать структуру пакетов и особенно их заголовков. Краткие сведения о заголовках пакетов вы сможете найти в документе TCP/IP and tcpdump Packet Reference Guide[1]. Будет полезно также прочесть спецификации протоколов стека TCP/IP, тексты которых вы найдете на сайте http://rfc-editor.org[2].

Нумерация байтов в описании этого модуля всегда начинается с 0 (первый байт заголовка пакета). Например, в заголовках IP байт 0 содержит 4-битовое поле Version (номер версии) и 4-битовое поле IP Header Length (размер заголовка IP), а байт 1 содержит поле TOS.

При работе с функциями модуля не забывайте в спецификации правила указывать команду загрузки модуля **-m u32**.

5.1.9.6.13.1 Проверка значений 2-байтовых полей

Простейшим вариантом использования модуля u32 является выборка из пакета 4-байтового блока, начиная с позиции **Start**, применение к нему маски **Mask** и сравнение результата с диапазонов **Range**. Ниже приведен пример такого использования:

```
iptables -m u32 --u32 "Start=Range"
```

В качестве значения поля **Start** следует указывать номер последнего интересующего вас байта минус 3 (например, если вас интересуют байты 4 и 5 в заголовке IP, следует использовать в качестве параметра **Start** значение 5 - 3 = 2). Битовая маска позволяет отбросить все биты, которые вас не интересуют. Значения маски лежат в диапазоне от 0 0xFFFFFFFF. Для того, чтобы получить интересующие нас байты 4 и 5, нужно отбросить байты 2 и 3. В этом случае маска должна иметь значение 0x0000FFFF (можно использовать сокращенную запись без нулей слева - 0xFFFF).

Таким образом, для того, чтобы проверить принадлежность значения IP ID (байты 4 и 5) к диапазону 2 - 256, мы должны использовать правило:

```
iptables -m u32 --u32 "2&0xFFFF=0x2:0x0100"
```

Переведем это правило на человеческий язык: "Загрузить модуль u32 и выполнить по отношению к пакету проверку u32 - выбрать из пакета 4 байта, начиная со второго[3], применить к ним маску 0x0000FFFF (для отбрасывания байтов 2 и 3) и проверить, что значение IP ID лежит в диапазоне от 2 до 256."

В iptables отсутствует специальная операция проверки поля IP ID и описанное правило эквивалентно условию **ip[2:2] >= 2 & ip[2:2] <= 256** для фильтров tcpdump/bpf.

В приведенном примере не было указано действие для пакетов, соответствующих правилу. В зависимости от ваших

1 *Этот документ можно загрузить с сайта http://www.sans.org/resources/tcpip.pdf, копия документа имеется в каталоге Documents/ приложенного к курсу компакт-диска.*
2 *Переводы основных стандартов Internet на русский язык доступны на сайте http://www.protokols.ru.*
3 *Байты 2 и 3 содержат поле Total Length, а байты 4 и 5 - поле IP ID.*

задач вы можете выбрать что-то типа :

```
-j LOG --log-prefix "ID-in-2-256 "
```

или

```
-j DROP
```

При необходимости вы можете включить в спецификацию правила и другие условия.

Вы можете использовать это правило, например, для проверки размера пакетов. Чтобы отобрать пакеты, размер которых не менее 256 байтов, можно использовать правило:

```
iptables -m u32 --u32 "0&0xFFFF=0x100:0xFFFF"
```

Эту операцию можно также выполнить с помощью правила iptables:

```
iptables -m length --length 256:65535
```

или фильтра bpf:

```
"len >= 256"
```

5.1.9.6.13.2 Проверка значений 1-байтовых полей

Проверка однобайтовых значений отличается от операций для 4-байтовых слов лишь использованием маски 0x000000FF (или короткого варианта 0xFF) для выделения 1 байта из 4 прочитанных модулем u32. Например, для отбора пакетов, в которых значение поля TTL не превышает 3[1]:

```
iptables -m u32 --u32 "5&0xFF=0:3"
```

которое эквивалентно правилу

```
iptables -m ttl --ttl-lt 4
```

или bpf-фильтру

```
"ip[8] <= 3"
```

5.1.9.6.13.3 Просмотр 4 байтов сразу

Для проверки значения IP-адреса получателя нам потребуются биты 16-19 из заголовка пакета. Поскольку мы должны проверить все 4 бита, маска в этом случае становится ненужной. Например, для обнаружения в поле адреса получателя значения 224.0.0.1 можно использовать правило

```
iptables -m u32 --u32 "16=0xE0000001"
```

эквивалентное правилу

```
iptables -d 224.0.0.1/32
```

Если мы хотим проверить только три первых байта адреса (принадлежность к сети класса C), снова потребуется маска (0xFFFFFF00), которая позволит избавиться от ненужного байта. Для проверки принадлежности адреса получателя к сети 192.168.15.0/24 (0xC0A80F00) можно использовать правило

```
iptables -m u32 --u32 "12&0xFFFFFF00=0xC0A80F00"
```

эквивалентное правилу

```
iptables -s 192.168.15.0/24
```

5.1.9.6.13.4 Проверка каждого байта в заголовке

Если вы хотите посмотреть значение поля TOS (байт 1 в заголовке IP), в соответствии с приведенными в параграфе 5.1.9.6.13.1 (стр. 137) в качестве стартовой позиции нужно указать значение -2 (1-3). Вместо этого можно начать выборку с байта 0, выделить его с помощью маски и переместить в последнюю позицию для упрощения проверки. Это не единственный способ решения задачи, но он служит достаточно хорошей иллюстрацией метода.

Для выделения поля TOS сначала сделаем выборку байтов 0 - 3 с помощью модуля u32, задав смещение 0. После этого выделим нужный байт (второй байт выбранного блока) с помощью маски 0x00FF0000. Далее для упрощения проверки значения поля TOS следует сдвинуть этот байт на 16 битовых позиций вправо. Для решения этой задачи модуль u32 поддерживает специальную операцию сдвига вправо, задаваемую символом **>>**, вслед за которым указывается величина сдвига (в нашем случае 16). В результате правило проверки наличия в поле TOS значения 0x08 (максимальная пропускная способность) будет иметь вид:

```
iptables -m u32 --u32 "0&0x00FF0000>>16=0x08"
```

Это правило эквивалентно строке:

```
iptables -m ttl --tos 8
```

5.1.9.6.13.5 Проверка отдельных битов

Если вам потребуется проверить значения некоторых битовых флагов (например, поля More Fragments) в iptables может не оказаться готовых правил для такой проверки. В частности, условию **-f** будут соответствовать все фрагменты, начиная со второго, а вам для работы может потребоваться идентификация всех фрагментов, кроме последнего. Рассмотрим, как можно решить задачу на примере уже упомянутого поля фрагментации. Интересующий нас бит находится в шестом байте, поэтому мы можем сделать выборку с байта 3 и отбросить с помощью маски (0x000000FF) байты 3-5. Но для решения нашей задачи не требуется весь байт полностью, поэтому мы можем воспользоваться маской 0x00000020 (0010 0000), которая позволит нам выделить только интересующий нас бит. После этого можно пойти двумя путями: сдвинуть бит в крайнюю правую позицию для сравнения с 1 или

1 *Это может быть полезно для обнаружения трассировки ваших хостов извне.*

оставить все как есть, сравнивая с соответствующим значениям (32). В первом случае правило будет иметь вид

```
iptables -m u32 --u32 "3&0x20>>5=1"
```

а во втором

```
iptables -m u32 --u32 "3&0x20=0x20"
```

Оба варианта будут возвращать значение true при установленном флаге More Fragments.

5.1.9.6.13.6 Объединение проверок

Если вы хотите объединить в одном правиле несколько проверок, используйте знак

```
&&
```

5.1.9.6.13.7 Работа с заголовками пакетов

В трех следующих параграфах рассматриваются примеры реализации правил с использованием модуля u32 для анализа заголовков TCP, UDP и ICMP.

5.1.9.6.13.7.1 TCP

Рассмотрим задачу анализа поля порядкового номера TCP (байты 4 - 7 заголовка TCP). Предположим для простоты, что размер заголовка IP составляет 20 байтов[1]. Первым считываемым байтом будет четвертый октет заголовка TCP, следующего непосредственно после заголовка IP. Для нашего примера будем проверять соответствие порядкового номера значению 41 (0x29) с помощью правила

```
iptables -m u32 --u32 "24=0x29"
```

Однако это правило не будет работать для пакетов, в которых размер заголовка IP не равен 20 байтам, поэтому постараемся его усовершенствовать.

Для начала убедимся, что пакет относится к протоколу TCP. Информация о протоколе содержится в байте 9 заголовка IP, поэтому разумно будет взять из заголовка IP 4 байта, начиная с байта 6, отбросить байты 6-8 и убедиться, что оставшийся байт содержит идентификатор протокола EC3 (6). Усовершенствованное правило проверяет принадлежность пакета к протоколу TCP и значение порядкового номера (41)

```
iptables -m u32 --u32 "6&0xFF=0x6 && 24=0x29"
```

Вернемся к проблеме размера заголовка IP, который может меняться в зависимости от наличия опций. Для решения этой задачи нужно выполнить ряд действий:

1) Определим размер заголовка из первого байта заголовка IP (младший полубайт, указывающий размер заголовка в 4-байтовых словах). Для выделения байта, содержащего размер нужно прочесть первые 4 байта заголовка и сдвинуть полученное значение на 3 байта вправо

   ```
   "0>>24"
   ```

2) Далее нужно выделить четыре младших бита и умножить полученное значение на 4 (размер слова). Умножение эквивалентно сдвигу влево на 2 бита, а для выделения результата следует взять 6-битовую маску 0x3C (<< 0x0F). В результате получается

   ```
   "0>>22&0x3C"
   ```

   Для заголовка IP без опций это выражение дает в результате значение 20, в остальных случаях - размер заголовка с учетом опций.

3) Добавим смещение поля порядкового номера в заголовке TCP (4) и передадим его в качестве стартовой позиции правилу u32. Для передачи значения используется знак **@**, который трактует содержащееся слева от него значение как смещение для отсчета стартовой позиции u32.

Результирующее правило будет иметь вид

```
iptables -m u32 --u32 "6&0xFF=0x6 && 0>>22&0x3C@4=0x29"
```

Таким образом мы получаем для проверки значение порядкового номера TCP для всех пакетов (независимо от размера заголовка IP).

Однако мы еще не приняли во внимание возможность фрагментации пакетов. Протокол IP устроен так, что заголовки IP не могут быть разделены на фрагменты, однако заголовки и данные TCP могут быть фрагментированы. Поэтому при просмотре второго и последующих фрагментов даже последний вариант правила будет давать некорректные результаты и нам придется научиться принимать во внимание фрагменты.

Для обеспечения корректной работы правила нужно научиться отличать первый фрагмент и нефрагментированные пакеты. Для решения этой задачи нужно проверить значение байтов 6 и 7 заголовка IP (смещение фрагмента и флаги), отбросив значение поля флагов. Это можно сделать с помощью операции

```
"4&0x1FFF=0"
```

Окончательное правило (идентификация протокола TCP и первого фрагмента или отсутствия фрагментации, проверка значения порядкового номера) приобретает форму

```
iptables -m u32 --u32 "6&0xFF=0x6 && 4&0x1FFF=0 && 0>>22&0x3C@4=0x29"
```

Это правило будет корректно работать во всех случаях, поскольку любая реализация IP должна обрабатывать пакеты размером, по крайней мере 68 байтов, а заголовок IP не может превышать в размере 60 байтов. Поэтому даже при столь жесткой фрагментации первые 8 байтов заголовка TCP окажутся в первом фрагменте.

Отметим также, что модуль u32 возвращает значение false (несоответствие правилу), для тех случаев, когда

1 *Во многих случаях это предположение верно, но за счет опций IP размер заголовка может увеличиваться.*

указанные правилом значения выходят за размеры пакета.

5.1.9.6.13.7.2 ICMP

В качестве примера рассмотрим правило для обнаружения пакетов **ICMP Host Unreachable** (ICMP, тип 3, код 1). Как и в предыдущем примере нам нужно проверить в заголовке IP поле протокола (1 для ICMP) и после этого просматривать полные пакеты и первые фрагменты:

```
"6&0xFF=1 && 4&0x1FFF=0"
```

Для проверки типа и кода ICMP нам опять потребуется пропустить заголовок IP

```
"0>>22&0x3C@..."
```

Для выборки двух первых байтов, начнем чтение с байта 0 и сдвинем вправо первые 16 битов. В результате получится правило

```
iptables -m u32 --u32 "6&0xFF=1 && 4&0x1FFF=0 && 0>>22&0x3C@0>>16=0x0301"
```

5.1.9.6.13.7.3 UDP

Попытаемся заглянуть внутрь пакетов и выделить все пакеты UDP с запросами DNS. В этом случае следует не только проверить порт получателя (53), но и определить значение старшего бита в байте 2 поля данных пакета (DNS query).

Начнем с проверки принадлежности пакета протоколу UDP:

```
"6&0xFF=17"
```

Как обычно, выделим первый фрагмент или нефрагментированный пакет:

```
"4&0x1FFF=0"
```

Для проверки порта получателя нам потребуются байты 2 и 3 из заголовка UDP, который следует вслед за заголовком IP (который мы просто пропускаем):

```
"0>>22&0x3C@0&0xFFFF=53"
```

Если пакет соответствует всем рассмотренным выше условиям, мы проверяем в нем содержимое поля данных (не забудьте пропустить заголовок IP переменной длины и 8-байтовый заголовок UDP):

```
"0>>22&0x3C@8 ..."
```

для того, чтобы идентифицировать запрос DNS. Для того, чтобы отличить запрос от отклика, нам потребуется старший бит байта 2, используем смещение 8 для выборки первых 4 байтов данных и сдвинем результат на 15 битов вправо для выделения бита Query с помощью маски 0x01:

```
"0>>22&0x3C@8>>15&0x01=1"
```

Полное правило будет иметь вид:

```
iptables -m u32 --u32 "6&0xFF=17 && 4&0x1FFF=0 && 0>>22&0x3C@0&0xFFFF=53 &&
0>>22&0x3C@8>>15&0x01=1"
```

Эквивалентное правило можно записать и в более простой форме с использованием других проверок iptables. Выберем для этого нефрагментированные пакеты или первые фрагменты UDP, адресованные в порт 53 и применим к ним последнее условие u32:

```
iptables -p udp --dport 53 \! -f -m u32 --u32 "0>>22&0x3C@8>>15&0x01=1"
```

5.1.9.6.13.8 Примеры правил

Сначала повторим описанные в предыдущих параграфах проверки, а потом добавим несколько новых правил.

```
"2&0xFFFF=0x2:0x0100"
```

проверяет, что значение поля IP ID лежит в диапазоне от 2 до 256.

```
"0&0xFFFF=0x100:0xFFFF"
```

проверяет, что размер пакета не менее 256 байтов.

```
"5&0xFF=0:3"
```

проверяет, что пакет имеет значение TTL не более 3.

```
"16=0xE0000001"
```

проверяет что IP-адрес получателя равен 224.0.0.1.

```
"12&0xFFFFFF00=0xC0A80F00"
```

проверяет, что пакет получен из сети класса C 192.168.15.X.

```
0&0x00FF0000>>16=0x08
```

проверяет, что поле TOS имеет значение 8 (максимальная пропускная способность).

```
"3&0x20>>5=1"
```

проверяет наличие флага More Fragments.

```
"6&0xFF=0x6"
```

проверяет принадлежность пакета к протоколу TCP.

```
"4&0x1FFF=0"
```

проверяет что пакет не фрагментирован или является первым фрагментов (смещение фрагмента = 0).

```
"0>>22&0x3C@4=0x29"
```

проверяет, что порядковый номер TCP равен 41 (для этого теста также требуется проверить два предыдущих условия).

"0>>22&0x3C@0>>16=0x0301"
проверяет принадлежность пакета ICMP (type=3, code=1); для этого требуется проверка принадлежности к протоколу UDP и отсутствие фрагментации или первый фрагмент).

"0>>22&0x3C@0&0xFFFF=53"
проверяет, что пакет UDP адресован в порт 53 (требуется также проверка на отсутствие фрагментации или первый фрагмент).

"0>>22&0x3C@8>>15&0x01=1"
проверяет наличие бита DNS Query в пакетах UDP (сначала проверяется принадлежность к протоколу UDP, отсутствие фрагментации/первый фрагмент и адресация в порт 53).

Несколько новых проверок:

"6&0xFF=1"
проверяет принадлежность к протоколу ICMP.

"6&0xFF=17"
проверяет принадлежность к протоколу UDP.

"4&0x3FFF=0"
проверяет нулевое смещение фрагмента и отсутствие флага MF (наличие дополнительных фрагментов). Выполнение условия говорит о том, что пакет не является фрагментом.

"4&0x3FFF=1:0x3FFF"
проверяет нулевое смещение фрагмента и присутствие флага MF (наличие дополнительных фрагментов). Выполнение условия говорит о том, что пакет является фрагментом.

0>>22&0x3C@12>>26&0x3C@-3&0xFF=0:255
проверяет принадлежность поля данных к протоколу TCP, обеспечивая изящный способ обнаружения пакетов TCP SYN.

5.1.10 Утилиты iptables

5.1.10.1 iptables-save

Команда **iptables-save** позволяет записать текущий набор правил из цепочек iptables в файл, который впоследствии можно использовать с командой **iptables-restore** (стр. 142) для восстановления таблиц. Команда достаточно проста и использует лишь два аргумента.

iptables-save [-c] [-b] [-t *table***]**
По умолчанию вывод осуществляется на стандартное устройство вывода (консоль), но его можно перенаправить в файл стандартными средствами Linux

iptables-save > <имя файла>
Опция **-c** говорит программе **iptables-save** о необходимости сохранения значений счетчиков пакетов и байтов для каждого правила. Сохранение этих данных может быть полезно при перезагрузке брандмауэра, поскольку позволяет сохранить все данные о работе правил (статистика, учет трафика и т. п.). По умолчанию значения счетчиков не сохраняются.

Аргумент **-t** позволяет указать программе **iptables-save** имя таблицы, для которой должны быть сохранены правила. При отсутствии этого аргумента автоматически сохраняются все правила из каждой таблицы. Раздельное сохранение правил из каждой таблицы может быть полезно для анализа и изменения списка правил.

Опция **-d** (**--dump**) на первый взгляд должна как-то управлять выводом информации, но в реальности она не меняет ничего.

Опция **-b** (**--binary**) в программе еще не реализована.

На рисунке 5.3 можно видеть строки, начинающиеся со стандартного для Linux знака комментария **#**. Каждая таблица указана строкой типа ***<имя таблицы>*** (например, *mangle). После имени таблицы следует набор строк, указывающий включенные в таблицу цепочки. Эти строки имеют вид

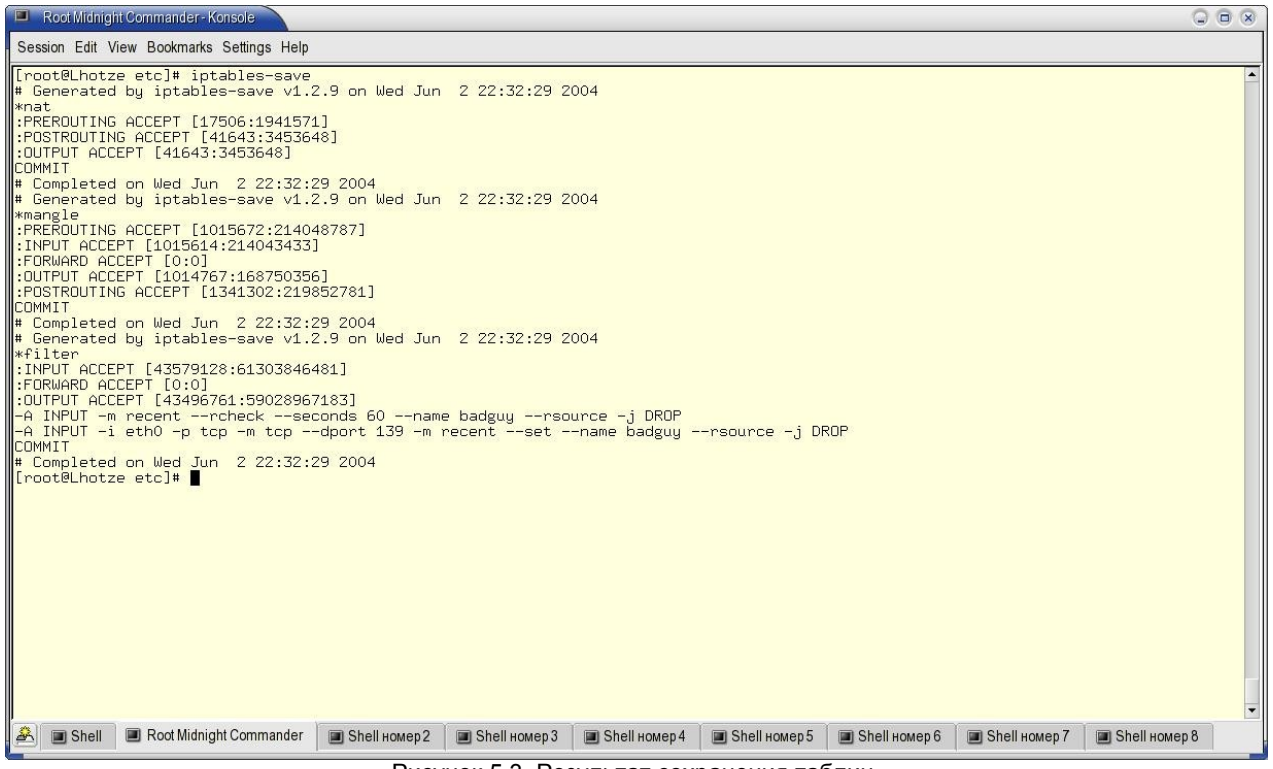


Рисунок 5.3. Результат сохранения таблиц.

:<имя цепочки> <политика цепочки> [<счетчик пакетов>:<счетчик байтов>].

Далее для каждой таблицы выводятся списки правил в каждой из встроенных и пользовательских цепочек таблицы. Завершается вывод информации для каждой таблицы строкой, содержащей ключевое слово **COMMIT**, указывающее, что в этой точке все правила таблицы передаются ядру.

5.1.10.2 iptables-restore

Программа **iptables-restore** служит для загрузки правил **iptables**, сохраненных ранее с помощью команды **iptables-save** (стр. 141)[1]. Команду можно использовать с несколькими опциями:

iptables-restore [-b] [-c] [-v] [-h] [M] <имя файла>

Опция **-c** используется для установки значений счетчиков пакетов и байтов в соответствии с введенными значениями. Если вы использовали команду **iptables-save** с такой же опцией, значения счетчиков будут сохранены в файле. Для задания этой опции в командной строке можно также использовать полное название параметра **--counters**.

Аргумент **-n** (**--noflush) говорит программе iptables-restore** о необходимости сохранения имеющихся в таблицах правил. По умолчанию **iptables-restore** сбрасывает все таблицы и заполняет их впоследствии правилами из файла. Опция **-n** позволяет добавлять правила без удаления существующих. Такая возможность очень удобна в тех случаях, когда набор правил сохраняется в нескольких файлах (например, отдельный файл для каждой таблицы.

Опция **-v** (**--verbose**) обеспечивает при загрузке правил из файла вывод на консоль информации о процессе (строки комментариев из набора правил, сообщения об ошибках). При работе с большими списками правил эта опция очень удобна, хотя и несколько замедляет процесс загрузки правил.

Опция **-b** (**--binary**) в программе еще не реализована.

Аргумент

--modprobe=<команда>

служит для загрузки модулей, которые могут потребоваться для работы правил.

5.1.11 Компиляция и установка iptables

Вы можете воспользоваться бинарным дистрибутивом iptables, но в этом случае придется использовать тот набор возможностей, который сочли нужным включить в пакет при его компиляции. Часть этих возможностей может оказаться ненужной для вас, а каких-то возможностей вы лишитесь. Если уж мы взялись за создание межсетевого экрана своими руками, то не будем отступать с полпути и скомпилируем пакет iptables из исходных текстов.

Процесс установки исходных текстов программ и дополнений был описан выше (параграф 4.3.1.1 на стр. 61).

Для компиляции программы используйте команду

make KERNEL_DIR=<каталог>

указав в качестве параметра имя каталога, в котором хранятся исходные тексты ядра. После завершения

1 *Старые версии программы принимали данные только со стандартного устройства ввода (клавиатуры), поэтому для них следует использовать стандартные средства перенаправления ввода-вывода Linux. В руководстве man для программы по-прежнему указывается только возможность ввода с устройства STDIN, но практика показывает, что программа умеет работать с файлами без перенаправления ввода.*

компиляции следует воспользоваться командой

```
make install KERNEL_DIR=<каталог>
```

для установки программы.

По умолчанию файлы помещаются в подкаталоги дерева **/usr/local/**. Если вы хотите задать иной вариант установки программы, добавьте в команды компиляции и установки опции

```
BINDIR=<каталог> LIBDIR=<каталог> MANDIR=<каталог>
```

содержащие реальные имена каталогов для размещения исполняемых файлов, библиотек и руководств iptables.

Для статического включения библиотек в исполняемый файл программы, воспользуйтесь опцией **NO_SHARED_LIBS=1**.

5.2 Программа ebtables

ebtables.sourceforge.net

Программа **ebtables**[1] является средством фильтрации пакетов для мостов с поддержкой функций межсетевого экранирования (bridging firewall). Фильтрация осуществляется на основании значений полей в заголовках кадров Ethernet (канальный уровень модели OSI). Кроме фильтрации пакетов программа также может изменять MAC-адреса в кадрах Ethernet и выполнять функции моста-маршрутизатора.
Для использования ebtables требуется стандартное ядро версии 2.6.

ebtables представляет собой программу пользовательского пространства, поддерживающую таблицы правил для кадров Ethernet в ядре Linux. Эта программа работает аналогично программам пользовательского пространства iptables, но значительно проще в использовании.

Для использования программы в ядре должна быть включена опция **Ethernet Bridge tables (ebtables) support** (стр. 82). При выборе для опции значения M потребуется также загрузить модуль **ebtables**. В зависимости от деталей использования программы может потребоваться поддержка ядром дополнительных опций и загрузка других модулей, как указано ниже.

5.2.1 Цепочки ebtables

Программа поддерживает в ядре Linux три таблицы с цепочками правил для кадров Ethernet. Таблицы ядра используются для распределения функциональности по нескольким наборам правил, называемых цепочками. Каждая цепочка представляет собой упорядоченных правил соответствия для кадров Ethernet. Если данный кадр соответствует правилу, для этого кадра применяется заданная правилом операция (target). Если же кадр не соответствует спецификации данного правила, этот кадр передается следующему правилу цепочки и т. д. Пользователь может создавать свои цепочки, которые могут служить в качестве операций для встроенных и пользовательских цепочек.

5.2.2 Таблицы ebtables

Как было сказано выше в ядре Linux программа поддерживает три таблицы для кадров Ethernet - **filter**, **nat** и **broute**. По умолчанию все операции (правила) **ebtables** относятся к таблице **filter**. Для работы с другими таблицами используется опция

```
-t <имя таблицы>
```

в строке спецификации правила. Опция -t должна использоваться в начале строки правила (сразу же после ebtables)

5.2.2.1 Таблица filter

Используемая по умолчанию таблица **filter** содержит 3 встроенных цепочки - **INPUT** (для кадров, адресованных данному хосту), **OUTPUT** (для кадров, сгенерированных данным хостом) и **FORWARD** (для пересылаемых мостом кадров).

Для использования таблицы фильтрации в ядре должна быть включена опция **ebt: filter table support** (стр. 82). При выборе для опции значения M потребуется также загрузка модуля **ebtable_filter**.

5.2.2.2 Таблица nat

Таблица **nat** служит для изменения MAC-адресов и содержит 3 встроенных цепочки[2] - **PREROUTING** (изменение кадров на входе), **OUTPUT** (изменение локально сгенерированных кадров до передачи их мосту) и **POSTROUTING** (изменение кадров на выходе).

Для работы с таблицей nat в ядре должна быть включена опция **ebt: nat table support** (стр. 82). Если для опции было выбрано значение M, потребуется также загрузка модуля **ebtable_nat**.

5.2.2.3 Таблица broute

Таблица **broute** служит для выполнения функций моста-маршрутизатора (brouter) и включает 1 цепочку -

1 *Программу можно загрузить с сайта проекта http://ebtables.sourceforge.net/. Исходные тексты и документацию вы можете найти в каталоге SRC/ приложенного к книге компакт-диска.*
2 *Названия цепочек, равно как и название самой таблицы не совсем корректны, но сохранены в соответствии с именами, принятыми в iptables для удобства.*

BROUTING. Операции **DROP** и **ACCEPT** для таблицы broute имеют отличный от общепринятого смысл. **DROP** означает, что кадр будет маршрутизироваться, а **ACCEPT** говорит об использовании для кадра функций моста. Цепочка **BROUTING** используется на самых ранних этапах обработки пакетов. В эту цепочку передаются только пакеты, принимаемые через интерфейсы моста, которые находятся в состоянии forwarding (пересылка пакетов). Обычно для пересылки кадров используются функции моста, но вы можете поступить иначе. Для этого очень удобна операция **redirect**.

Для работы с этой таблицей в ядре должна быть включена опция **ebt: broute table support** (стр. 82). При выборе для опции значения M потребуется также загрузить модуль **ebtable_broute**.

5.2.3 Операции ebtables

Правила межсетевого экранирования задают критерии (условия) для кадров Ethernet и операции (target), выполняемые ядром при соответствии кадра заданным условиям. В качестве действий в правилах могут использоваться встроенные операции **ACCEPT**, **DROP**, **CONTINUE**, **RETURN**, дополнительные операции (extension, стр. 150) и пользовательские цепочки правил.

Операция **ACCEPT** означает восприятие кадра с передачей обработки на следующий этап (в другую цепочку или программу). **DROP** прерывает обработку кадра и удаляет этот кадр. **CONTINUE** передает кадр следующему правилу цепочки (такая операция может быть полезна для подсчета кадров или записи их в журнальные файлы системы). Операция **RETURN** завершает обработку кадра в данной цепочке и возвращает его в точку вызова этой цепочки. Дополнительные операции описываются в параграфе 5.2.8 (стр. 150).

5.2.4 Команды и опции ebtables

5.2.4.1 Основные команды

Аргументы и опции команд ebtables определяет спецификации правил и выполняемые этими правилами действия для таблицы, заданной параметром **-t**[1]. В каждой строке (команде Linux) может использоваться только одна из перечисленных в этом параграфе основных опций. Исключением из этого являются только опции **-Z** и -**atomic-file**, которые могут использоваться в дополнение к другим основным опциям.

`-A, --append`
добавляет правило в конец указанной цепочки.

`-D, --delete`
удаляет заданное правило из указанной цепочки. Удаляемое правило можно указать по его номеру[2] или использовать в команде спецификацию условий и действие, в точности соответствующие правилу, которое нужно удалить. При удалении правил по номеру можно удалить сразу несколько правил, задав диапазон номеров в форме

`start_nr[:end_nr]`
При удалении правил по номерам допускается использовать отрицательные значения номеров, смысл которых разъясняется в описании команды **-I**.

`-I, --insert`
вставляет правило в указанную номером строку списка. Если в цепочке имеется N правил, в качестве параметра команды I можно использовать значения от -N до N+1. Для случая положительных значений i, позиции с номером i-N-1 и i. Нулевое значение задает вставку правила вслед за последним из имеющихся в таблице правил как по команде -A.

`-P, --policy`
задает политику для данной цепочки. В качестве политики могут использоваться операции **ACCEPT**, **DROP** и **RETURN**[3].

`-F, --flush`
удаляет все правила из указанной цепочки. Если цепочка не указана, удаляются правила из всех цепочек. Удаление из цепочки всех правил не меняет выбранной для этой цепочки политики.

`-Z, --zero`
устанавливает нулевые значения счетчиков пакетов и байтов для указанной цепочки. Команду **-Z** можно использовать вместе с командой просмотра списка правил **-L**. При таком использовании команд на экран выводится список правил с текущими значениями счетчиков, после чего все счетчики сбрасываются.

`-L, --list`
выводит на экран список правил указанной цепочки. Команда **-L** поддерживает ряд опций:

`--Ln`
для вывода номера строки в начале каждого правила.

`--Lc`
показывает значения счетчиков пакетов и байтов в конце каждого выводимого правила.

`--Lx`
позволяет вывести список команд, использованных для создания правил цепочки. Эту команду можно использовать в сценариях загрузки или перезагрузки для ввода набора правил **ebtables**. Отметим, что опция

1 *По умолчанию правила относятся к цепочкам таблицы **filter**.*
2 *Для определения номера воспользуйтесь командой **ebtables -L**.*
3 *Описание операций приведено в параграфе 5.2.3 (стр. 144).*

--Lx несовместима с опциями **--Ln** и **--Lc**.

--Lmac2
выравнивает размер MAC-адресов, добавляя при необходимости нули слева. По умолчанию нули в начале адресов отбрасываются, если в них нет необходимости.

При использовании команды **-L --Lx** без имени цепочки выводится полный список команд, использованных для создания и переименования стандартных и пользовательских цепочек **ebtables**.

-N, --new-chain
создает новую пользовательскую цепочку с заданным именем. Имя цепочки может содержать до 31 символа, число пользовательских цепочек не ограничено.

-X, --delete-chain
удаляет указанную пользовательскую цепочку. Удалить можно только те цепочки, которые не используются в качестве операции в какой-либо из остающихся цепочек. Если команда вводится без имени цепочки **ebtables** будет удалять все неиспользуемые пользовательские цепочки.

-E, --rename-chain
переименовывает указанную цепочку. В отличие от iptables программа ebtables позволяет менять имена не только у пользовательских, но и у встроенных цепочек. Например, вы можете переименовать в PREBRIDGING цепочку PREROUTING, с помощью команды **-E PREROUTING**. Переименование цепочек не оказывает никакого влияния на работу **ebtables**[1].

--init-table
сбрасывает все цепочки таблицы в исходное (только пустые встроенные цепочки) состояние.

--atomic-init
копирует инициализационные данные для таблицы в файл. Эту команду можно использовать для сохранения инициализационной таблицы с целью последующего добавления в нее команд. Файл задается с помощью опции **--atomic-file** или указывается в переменной окружения **EBTABLES_ATOMIC_FILE**.

--atomic-save
копирует текущую информацию из таблицы ядра в файл. Эту команду можно использовать для сохранения текущей таблицы с целью последующего добавления в нее команд. Файл задается с помощью опции **--atomic-file** или указывается в переменной окружения **EBTABLES_ATOMIC_FILE**.

--atomic-commit
заменяет таблицу ядра данными из указанного файла. Эта команда может быть весьма полезна при настройке правил, когда вы можете загрузить таблицы из сохраненного ранее файла и вносить в нее пошаговые изменения. Загружаемые таблицы должны быть записаны в файл с помощью команды **--atomic-init** или **--atomic-save**. Файл, с которым работает данная команда, задается с помощью опции **--atomic-file** или указывается в переменной окружения **EBTABLES_ATOMIC_FILE**.

--atomic-file -Z
Команда **--atomic-file** может использоваться вместе с командой **-Z** для обнуления значений счетчиков при записи в файл. Обнуление счетчиков возможно и с помощью переменной окружения **EBTABLES_ATOMIC_FILE**.

5.2.4.2 Дополнительные команды

-V, --version
выводит информацию о номере версии программы **ebtables**.

-h, --help
выдает краткую справку о синтаксисе команд **ebtables**.

-j, --jump target
служит для задания действия, выполняемого правилом при соответствии пакета заданным условиям. В качестве действий могут служить основные операции **ACCEPT**, **DROP**, **CONTINUE**, **RETURN** (см. параграф 5.2.3 на стр. 144), дополнительные операции (см. параграф 5.2.8 на стр. 150) или пользовательские цепочки.

--atomic-file file
эта команда служит для задания имени файла, с которым работают команды **atomic-init**, **--atomic-save** и -**atomic-commit**. Данная команда должна быть указана в спецификации правила до команды, которая будет работать с файлом.

-M, --modprobe program
служит для автоматической загрузки требуемых модулей ядра.

5.2.5 Спецификации правил

Описанные в этом параграфе опции служат для задания правил, по которым проверяется соответствие пакетов. Флаг инверсии **!** позволяет изменить смысл опции или условия на обратный (логическое отрицание). Кроме описанных здесь опций спецификации правил, существует ряд дополнительных опций, описанных в параграфах 5.2.6 (стр. 146) и 5.2.7 (стр. 149).

-p, --protocol [!] ***protocol***
позволяет проверить соответствие протокола, с помощью которого был создан кадр. Протокол может быть задан

1 *Если вы не забудете изменить соответствующие параметры в ссылающихся на эту цепочку правилах.*

шестнадцатеричным номером[1], именем (например, *ARP*[2]) или значением **LENGTH**. Если параметр ***protocol*** имеет значение не более **0x0600**, это значение должно трактоваться как **LENGTH**, а не номер протокола. Предполагается, что все кадры для которых вместо типа протокола в заголовке указывается размер, относятся у одному типу. Программа **ebtables** использует для таких кадров имя протокола **LENGTH**.
Допускается также использование опции **--proto** в качестве синонима.

`-i, --in-interface [!]` ***`name`***
позволяет проверить интерфейс, через который был принят кадр (для цепочек **INPUT**, **FORWARD**, **PREROUTING** и **BROUTING**). Допускается также использование синонима **--in-if**.

`--logical-in [!]` ***`name`***
позволяет проверить логический интерфейс моста, через который был принят кадр (для цепочек **INPUT**, **FORWARD**, **PREROUTING** и **BROUTING**).

`-o, --out-interface [!]` ***`name`***
позволяет проверить для кадра соответствие интерфейса, через который кадр будет передаваться (для цепочек **OUTPUT**, **FORWARD** и **POSTROUTING**). Допускается использование синонима **--out-if**.

`--logical-out [!]` ***`name`***
позволяет проверить соответствие логического интерфейса моста, через который будет передан кадр (для цепочек **OUTPUT**, **FORWARD** и **POSTROUTING**).

`-s, --source [!]` ***`address`*****`[/`*****`mask`*****`]`**
позволяет проверить MAC-адрес отправителя кадра. Как адрес, так и маска задаются шестью шестнадцатеричными значениями, разделенными двоеточием (:). Допускается также использование специальных значений **Unicast**, **Multicast**, **Broadcast** и **BGA**[3].
Unicast = 00:00:00:00:00:00/01:00:00:00:00:00,

Multicast = 01:00:00:00:00:00/01:00:00:00:00:00,

Broadcast = ff:ff:ff:ff:ff:ff/ff:ff:ff:ff:ff:ff,

BGA = 01:80:c2:00:00:00/ff:ff:ff:ff:ff:ff.

Отметим, что широковещательный адрес будет соответствовать также значению **Multicast**. Допускается использование синонима **--src**.

`-d, --destination [!]` ***`address`*****`[/`*****`mask`*****`]`**
позволяет проверить MAC-адрес получателя кадра. Для указания адресов получателей используются такие же правила, как для адресов отправителей. Допускается использование синонима **--dst**.

5.2.6 Основные сопоставления

Сопоставления **ebtables** не требуется явно загружать с помощью опции **-m**, поскольку они включены в программы пользовательского пространства. Однако для использования этих соответствий в ядре должны быть активизированы требуемые для них модули (см. стр.).

5.2.6.1 Проверка полей DSAP/SSAP и SNAP в кадрах 802.3

Описанные здесь соответствия служат для проверки полей 802.3 DSAP/SSAP или типа SNAP. Для проверки соответствия требуется указать протокол **LENGTH** (см. стр. 145).

`--802_3-sap [!]` ***`sap`***
проверяет однобайтовые поля DSAP и SSAP в заголовках кадров 802.3. Значения этих полей всегда совпадают, поэтому в качестве аргумента указывается только одно значение.

`--802_3-type [!]` ***`type`***
Если в кадре 802.3 поля **DSAP** и **SSAP** имеют значение **0xaa**, для определения типа содержимого пакета должно использоваться 2-байтовое поле SNAP. Аргумент задается шестнадцатеричным значением. Поле **SNAP** проверяется только для кадров 802.3 с полями **DSAP**/**SSAP** = **0xaa**.

Для использования этой проверки в ядре должна быть включена опция **ebt: 802.3 filter support** (параграф 4.4.2.2.14.5.1.4 на стр. 82). Если для этой опции было выбрано значение **M**, потребуется также загрузить модуль ядра **ebt_802_3**.

5.2.6.2 Проверка наличия адресов в списке

Это условие служит для проверки наличия MAC-адресов или пар MAC/IP в списке MAC-адресов и пар MAC/IP. Записи списка имеют формат **xx:xx:xx:xx:xx:xx[=ip.ip.ip.ip][,]**. Записи списка разделяются запятыми, поле адреса IP является необязательным. Для проверки соответствия можно указывать один MAC-адрес в паре с различными адресами IP и наоборот. Если MAC-адрес не присутствует ни в одной записи списка, кадр считается несоответствующим данному правилу (если не указана инверсия с помощью знака !). Для проверки соответствия можно использовать две опции:

`--among-dst [!]` ***`list`***

1 *Более* 0x0600.
2 *Соответствие имен и номеров протоколов указывается в файле /etc/ethertypes вашей Linux-системы. Например, значение 0x0800 соответствует протоколу IPv4. При указании имен протоколов регистр символов не принимается во внимание.*
3 *Bridge Group Address*

проверяет наличие MAC-адреса получателя в заданном списке. Если кадр Ethernet имеет тип **IPv4** или **ARP**, можно проводить сравнение для пары MAC/IP.

--among-src [!] ***list***
проверяет наличие MAC-адреса отправителя в заданном списке. Если кадр Ethernet имеет тип **IPv4** или **ARP**, можно проводить сравнение для пары MAC/IP.

Для использования такой проверки в ядре должна быть включена опция **ebt: among filter support** (параграф 4.4.2.2.14.5.1.5 на стр. 82). Если при компиляции ядра для опции было выбрано значение **M**, потребуется также загрузить модуль ядра **ebt_among**.

5.2.6.3 Проверка полей ARP

Описанные ниже соответствия служат для проверки полей arp. Для использования этих условий должен быть указан протокол **ARP** или **RARP** (см. стр. 145).

--arp-opcode [!] ***opcode***
проверяет значение поля opcode пакетов ARP/RARP. Код может быть указан именем или десятичным значением. Соответствие имен и значений приведено ниже. Для просмотра списка имен и значений кодов можно также воспользоваться командой **ebtables -h arp**.

1=Request (запрос)
2=Reply (отклик)
3=Request_Reverse (запрос обратного преобразования)
4=Reply_Reverse (отклик на запрос обратного преобразования)
5=DRARP_Request (запрос DRARP)
6=DRARP_Reply (отклик на запрос DRARP)
7=DRARP_Error (ошибка DRARP)
8=InARP_Request (запрос InARP)
9=ARP_NAK

--arp-htype [!] ***hardware type***
проверяет тип оборудования. В качестве идентификатора типа может использоваться символьное имя **Ethernet** или десятичное значение 1.

--arp-ptype [!] ***protocol type***
проверяет тип используемого протокола для пакетов ARP/RARP. Протокол может быть задан символьным именем (IPv4) или шестнадцатеричным числом (0x0800).

--arp-ip-src [!] ***address*****[/*****mask*****]**
проверяет IP-адрес отправителя пакетов ARP.

--arp-ip-dst [!] ***address*****[/*****mask*****]**
проверяет IP-адрес получателя пакетов ARP.

--arp-mac-src [!] ***address*****[/*****mask*****]**
проверяет MAC-адрес отправителя пакетов ARP.

--arp-mac-dst [!] ***address*****[/*****mask*****]**
проверяет MAC-адрес получателя пакетов ARP.

Для использования такой проверки в ядре должна быть включена опция **ebt: ARP filter support** (параграф 4.4.2.2.14.5.1.6 на стр. 82). Если при компиляции ядра для опции было выбрано значение **M**, потребуется также загрузить модуль ядра **ebt_arp**.

5.2.6.4 Проверка полей IP

Эти условия служат для проверки полей IP и требуют указания в спецификации правила протокола **IPv4** (см. стр. 145).

--ip-source [!] ***address*****[/*****mask*****]**
проверяет IP-адрес отправителя. Допускается использование синонима **--ip-src**.

--ip-destination [!] ***address*****[/*****mask*****]**
проверяет IP-адрес получателя. Допускается использование синонима **--ip-dst**.

--ip-tos [!] ***tos***
проверяет тип обслуживания, задаваемый шестнадцатеричным значением.

--ip-protocol [!] ***protocol***
проверяет протокол IP. Допускается использование синонима **--ip-proto**.

--ip-source-port [!] ***port*****[:*****port*****]**
проверяет порт (диапазон портов) отправителя для протоколов TCP (6) и UDP (17). Если опущена нижняя граница диапазона, ее значение предполагается нулевым, при опущенной верхней границе предполагается значение 65535. Допускается использование синонима **--ip-sport**.

--ip-destination-port [!] ***port*****[:*****port*****]**
проверяет порт (диапазон портов) получателя для протоколов TCP (6) и UDP (17). Если опущена нижняя граница диапазона, ее значение предполагается нулевым, при опущенной верхней границе предполагается значение 65535. Допускается использование синонима -**ip-dport**.

Для проверки полей IP требуется ядро со включенной опцией **ebt: IP filter support** (параграф 4.4.2.2.14.5.1.7 на стр.

83). Если при компиляции ядра для опции было выбрано значение **M**, потребуется также загрузить модуль ядра **ebt_ip**.

5.2.6.5 Ограничение темпа совпадений (limit)

Это условие позволяет ограничить темп совпадений с остальными условиями данного правила. Подробное описание работы алгоритма ограничения темпа совпадений приведено в параграфе 5.1.9.5.2 (стр. 125). Для ограничения темпа соответствия могут использоваться параметры

`--limit <средняя частота>`

и

`--limit-burst <пиковое значение>`

Первый параметр задает пороговую частоту событий (число событий в единицу времени) и указывается в формате **значение/суффикс**. Значение определяет число событий, а суффикс - единицу времени (/s или /second - секунда, /m или /minute - минута, /h или /hour - час, /d или /day - сутки). По умолчанию используется пороговая частота 3 пакета в час. Второй параметр определяет пик “разовой” доставки пакетов. По умолчанию для пика используется значение 5. Модуль работает следующим образом:

- условие считается выполненным, пока значение счетчика пакетов не превысит пика **limit-burst**;
- каждый пакет, соответствующий правилу, увеличивает значение счетчика на 1;
- по истечении каждого интервала 1/**limit** значение счетчика уменьшается на 1.

Для установки пороговых значений темпа соответствия требуется ядро со включенной опцией (параграф 4.4.2.2.14.5.1.8 на стр. 83). Если при компиляции ядра для опции было выбрано значение **M**, потребуется также загрузить модуль ядра **ebt_limit**.

5.2.6.6 Проверка маркеров

Условие

`--mark [!] [`*`value`*`][/`*`mask`*`]`

позволяет проверить соответствие целочисленного беззнакового значения маркера. Если в условии задано также значение маски, для сравнения с заданным значением используется результат операции **AND** по отношению к маске и значению маркера в кадре. Если в условии указана только маска, полученное в результате операции **AND** значение сравнивается с нулем.

Для проверки маркеров требуется ядро с поддержкой опций **ebt: mark target support** (параграф 4.4.2.2.14.5.1.15 на стр. 83) и **ebt: mark filter support** (параграф 4.4.2.2.14.5.1.9 на стр. 83). В зависимости от выбора значения этих опций при компиляции ядра может также потребоваться загрузка модулей **ebt_mark** и **ebt_mark_m**.

5.2.6.7 Проверка типа кадров

Условие

`--pkttype-type [!] `*`type`*

позволяет проверить для кадров “класс” Ethernet. Допустимы значения классов:

- **broadcast** (широковещательный MAC-адрес получателя),
- **multicast** (групповой MAC-адрес получателя),
- **host** (MAC-адрес принимающего устройства),
- **otherhost** (не относится ни к одному из перечисленных типов).

Для проверки соответствия кадров тому или иному классу требуется ядро с поддержкой опции **ebt: packet type filter support** (параграф 4.4.2.2.14.5.1.10 на стр. 83). Если при компиляции ядра для опции было выбрано значение **M**, потребуется также загрузить модуль ядра **ebt_pkttype**.

5.2.6.8 Проверка STP

Этот модуль используется для проверки полей STP[1] BPDU[2]. Адрес получателя должен быть указан как BGA-адрес (см. параграф 5.2.5 на стр. 145).

`--stp-type [!] `*`type`*

проверяет тип BPDU (0-255); распознаются типы **config** (0) и **tcn** (128 - изменение топологии).

`--stp-flags [!] `*`flag`*

проверяет флаг BPDU (0-255); распознаются флаги **topology-change** (1 - изменение топологии) и **topology-change-ack** (128 - подтверждение изменения топологии).

`--stp-root-prio [!] [`*`prio`*`][:`*`prio`*`]`

проверяет значение приоритета корневого моста (0-65535).

`--stp-root-addr [!] [`*`address`*`][/`*`mask`*`]`

проверяет MAC-адрес корневого моста. Описание адресов приводится в параграфе 5.2.5 (стр. 145).

1 *Spanning Tree Protocol - протокол остовного дерева, используемый для реализации беспетлевой топологии на канальном уровне.*
2 *Bridge protocol data unit - пакет данных протокола мостов.*

--stp-root-cost [!] [*cost*][:*cost*]
проверяет стоимость пути до корневого моста (0-4294967295).

--stp-sender-prio [!] [*prio*][:*prio*]
проверяет значение приоритета для отправителя BPDU (0-65535).

--stp-sender-addr [!] [*address*][/*mask*]
проверяет MAC-адрес отправителя BPDU. Описание адресов приводится в параграфе 5.2.5 (стр. 145).

--stp-port [!] [*port*][:*port*]
проверяет идентификатор порта (0-65535).

--stp-msg-age [!] [*age*][:*age*]
проверяет значение таймера “старения” (age timer - 0-65535).

--stp-max-age [!] [*age*][:*age*]
проверяет максимальное значение таймера “старения” (0-65535).

--stp-hello-time [!] [*time*][:*time*]
проверяет значение таймера hello (0-65535).

--stp-forward-delay [!] [*delay*][:*delay*]
проверяет значение таймера задержки пересылки (0-65535).

Возможность проверки полей STP BPDU обеспечивается при включенной опции ядра **ebt: STP filter support** (параграф 4.4.2.2.14.5.1.11 на стр. 83). Если для опции было выбрано значение **M**, потребуется также загрузка модуля ядра **ebt_stp**.

5.2.6.9 Проверка параметров VLAN

Этот модуль служит для проверки управляющих полей тегов 802.1Q. Для использования модуля в спецификации правила должен быть задан протокол **802_1Q** (0x8100).

--vlan-id [!] *id*
проверяет значение идентификатора VLAN (VID); допустимые значения лежат в диапазоне от 0 до 4095.

--vlan-prio [!] *prio*
проверяет значение поля user_priority (0 - 7). Поле VID должно иметь значение 0 (null VID) или быть пустым (в этом случае предполагается VID = 0).

--vlan-encap [!] *type*
проверяет тип/размер инкапсулированного кадра Ethernet, заданный шестнадцатеричным значением (0x0000 - 0xFFFF) или символьным именем[1]. Значения меньше 0x0800 задают размер инкапсулированного кадра Ethernet.

Для проверки тегов VLAN требуется ядро со включенной опцией **ebt: 802.1Q VLAN filter support** (параграф 4.4.2.2.14.5.1.12 на стр. 83). Если при компиляции ядра для этой опции использовалось значение M, потребуется также загрузка модуля **ebt_vlan**.

5.2.7 Сторожа (WATCHER)

Сторожа (Watcher)лишь просматривают попадающие к ним кадры, не выполняя по отношению к ним никаких специальных операций.

5.2.7.1 Операция log

Модуль log позволяет записывать информацию о кадрах в журнальные файлы[2]. В отличие от программы iptables в ebtables не используется операция LOG или ULOG.

--log
эта опция просто задает запись в системный журнал без каких-либо дополнительных условий. По умолчанию в этом случае не осуществляется запись для ARP и IP, не используется префикса в записях журнала и установлен уровень протоколирования **log-level=info**.

--log-level *level*
задает уровень протоколирования[3]. По умолчанию установлен уровень **info**. Для получения дополнительной информации вы можете воспользоваться командой **ebtables -h log**.

--log-prefix *text*
определяет префикс, используемый при записи в журнальный файл.

--log-ip
обеспечивает запись сведений IP для кадров, соответствующих протоколу IP. По умолчанию эта информация не записывается в журнальный файл.

--log-arp
обеспечивает запись сведений ARP/RARP для кадров, соответствующих протоколам ARP/RARP. По умолчанию эта информация не записывается в журнальный файл.

Для записи заголовков кадров в журнальные файлы требуется ядро со включенной опцией **ebt: log support**

1 *Соответствие имен и значений указывается в файле **/etc/ethertypes**.*
2 *Работа с журнальными файлами Linux рассматривается в параграфе 2.8.*
3 *Описание уровней протоколирования для системных журналов Linux приводится в параграфе 2.8.4.2 (стр. 50).*

(параграф 4.4.2.2.14.5.1.18 на стр. 84). если ядро компилировалось со значением опции **M**, потребуется также загрузка модуля **ebt_log**.

5.2.8 Дополнительные операции ebtables

Кроме основных операций **ACCEPT**, **DROP**, **CONTINUE**, **RETURN** (см. параграф 5.2.3 на стр. 144), программа ebtables поддерживает ряд описанных ниже дополнительных операций.

5.2.8.1 Соответствие arpreply

Проверка соответствия **arpreply** может использоваться в цепочке **PREROUTING** таблицы **nat** (параграф 5.2.2.2 на стр. 143). Если эта операция получает запрос ARP, она будет автоматически генерировать ARP-отклик, используя для него указанный правилом MAC-адрес. Если полученный кадр не является запросом ARP, никаких действий эта операция не выполняет. Опция

--arpreply-mac ***address***

задает MAC-адрес для откликов ARP. Указанное значение помещается в поле адреса отправителя кадра Ethernet и поле данных ARP.

--arpreply-target ***target***

задает стандартную операцию для правила. После отправки отклика ARP программа **ebtables** будет выполнять эту операцию. По умолчанию после передачи отклика кадр отбрасывается (операция **DROP)**.

Для работы с откликами ARP требуется ядро со включенной опцией **ebt: arp reply target support** (параграф 4.4.2.2.14.5.1.13 на стр. 83). Если для опции было выбрано значение M, потребуется также загрузить модуля ядра **ebt_arpreply**.

5.2.8.2 Операция dnat

Операция **dnat** может использоваться в цепочке **BROUTING** таблицы **broute** (параграф 5.2.2.3 на стр. 143), а также цепочках **PREROUTING** и **OUTPUT** таблицы **nat** (параграф 5.2.2.2 на стр. 143). Опция

--to-destination ***address***

Задает значение MAC-адреса, используемое для подстановки в пакеты. Можно использовать также краткую форму опции **--to-dst**.

--dnat-target ***target***

задает для правила стандартную операцию, выполняемую после замены адреса. По умолчанию в этом качестве используется операция **ACCEPT**. Указав стандартную операцию **CONTINUE**, вы сможете выполнить для пакета другие дополнительные операции. Использовать стандартную операцию **DROP** имеет смысл только для цепочки **BROUTING**, но логичней для таких случаев применять описанную ниже операцию **redirect** (стр. 150). В качестве итоговой можно указывать и операцию **RETURN**.

Для использования функций трансляции MAC-адресов получателей требуется ядро со включенной опцией **ebt: dnat target support** (параграф 4.4.2.2.14.5.1.14 на стр. 83). Если для этой опции было выбрано значение M, потребуется также загрузка модуля ядра **ebt_dnat**.

5.2.8.3 Операция mark

Операция **mark** может использоваться во всех цепочках любой из таблиц и обеспечивает возможность маркировки кадров при выполнении заданных правилом условий. Значение маркера задается опцией

--set-mark ***value***

а параметр

--mark-target ***target***

указывает стандартную операцию, выполняемую после маркировки кадра. По умолчанию маркировка завершается операцией **ACCEPT**. Воспользовавшись стандартной операцией **CONTINUE** вы сможете передать кадр для дальнейшей обработки правилами цепочки.

Для маркировки кадров требуется ядро со включенной опцией **ebt: mark target support** (параграф 4.4.2.2.14.5.1.15 на стр. 83). Если для опции было выбрано значение M, потребуется также загрузить модуль ядра **ebt_mark**.

5.2.8.4 Операция redirect

Операция **redirect** будет заменять MAC-адрес получателя на адрес интерфейса моста, через который был принят кадр. Операцию можно использовать только в цепочке **BROUTING** таблицы **broute** (стр. 143) и цепочке **PREROUTING** таблицы **nat** (стр. 143). Опция

--redirect-target ***target***

задает стандартную операцию, выполняемую после замены адреса получателя. По умолчанию используется операция **ACCEPT**. Если вы хотите использовать для правила дополнительные проверки, укажите операцию **CONTINUE**. Операция **DROP** в цепочке **BROUTING** будет передавать кадр маршрутизатору (взамен использования функций моста). Операция **RETURN** будет возвращать управление в вызвавшую правило цепочку (**RETURN** нельзя использовать для встроенных цепочек).

Для поддержки операций замены адреса получателя требуется ядро со включенной опцией **ebt: redirect target support** (параграф 4.4.2.2.14.5.1.16 на стр. 83). Если для опции ядра было выбрано значение **M**, потребуется также загрузить модуль **ebt_redirect**.

5.2.8.5 Операция snat

Операция **snat** может использоваться только в цепочке **POSTROUTING** таблицы **nat** (стр. 143). Опция

`--to-source` ***`address`***

задает помещаемое в кадр значение MAC-адреса отправителя и может использоваться в сокращенной форме **--to-src**.

`--snat-target` ***`target`***

задает стандартную операцию, выполняемую после замены адреса отправителя. По умолчанию используется операция **ACCEPT**. Если вы хотите использовать для правила дополнительные проверки, укажите операцию **CONTINUE**. Операция **DROP** не имеет смысла, но вы можете ее использовать, если ничего иного не приходит в голову. Операция **RETURN** будет возвращать управление в вызвавшую правило цепочку (**RETURN** нельзя использовать для встроенных цепочек).

Для замены адреса отправителя требуется ядро со включенной опцией **ebt: snat target support** (параграф 4.4.2.2.14.5.1.17 на стр. 84). При выборе для опции значения **M**, вы должны будете также загрузить модуль ядра **ebt_snat**.

5.3 Программа arptables

ebtables.sourceforge.net

Приложение пользовательского пространства **arptables** позволяет создавать и поддерживать таблицу правил ARP[1] в ядре Linux. Программа проверяет пакеты ARP, работая аналогично модулям пользовательского пространства iptables, но отличается от последних простотой использования.

Для работы с программой требуется ядро со включенной опцией **ARP tables support** (стр. 80). Если для опции было выбрано значение M, перед использованием **arpatables** потребуется загрузить модуль **arp_tables**.

5.3.1 Цепочки arptables

Программа использует таблицу ядра для распределения своих функций по нескольким наборам правил, называемых цепочками (chain). Каждая из цепочек представляет собой упорядоченный список правил, по которым проверяется соответствие для кадров ARP. Если кадр соответствует заданным правилом условиям, по отношению к этому кадру выполняется заданное правилом действие (target), в качестве которого может использоваться одна из стандартных операций arptables (параграф 5.3.3) или пользовательская цепочка. После выполнения этой операции обработка кадра в данной цепочке может завершиться[2]. Если кадр не соответствует данному правилу, он передается следующему в списке правилу и процесс продолжается до завершения списка. После проверки последнего правила к пакету применяется операция, заданная политикой цепочки (стр. 152).

5.3.2 Таблица filter

Таблица фильтрации программы arptables содержит две (ядра серии 2.4) или 3 (ядра серии 2.6) встроенных цепочки - **INPUT** (фильтрация пакетов, адресованных данному хосту), **OUTPUT** (фильтрация пакетов, сгенерированных данным хостом) и **FORWARD**[3] (фильтрация пакетов, пересылаемых с использованием функций моста). Для использования фильтрации пакетов с помощью программы arptables в ядре Linux должна быть включена опция **ARP packet filtering** (параграф 4.4.2.2.14.2.18.1 на стр. 80). Если для опции было выбрано значение M, потребуется также загрузка модуля **arptable_filter**.

5.3.3 Операции arptables

Правила arptables содержат спецификации условий, которым должен соответствовать кадр, и операцию, выполняемую в случае соответствия (target). Выполнение заданной правилом операции может завершить прохождение кадра через данную цепочку или передать кадр следующему правилу. Если кадр не соответствует заданным условиям, для него проверяются условия следующего правила и процесс продолжается пока одна из операций не прервет прохождение кадра через цепочку или не будет достигнут конец цепочки[4]. В качестве операций по над пакетами могут использоваться основные операции arptables (ACCEPT, DROP, CONTINUE, RETURN - стр. 151), дополнительные операции (см. стр. 153) или пользовательские цепочки.

Операция **ACCEPT** означает восприятие кадра и завершение его обработки в данной цепочке. Операция **DROP** приводит к завершению обработки кадра и его отбрасыванию. Операция **CONTINUE** просто передает кадр следующему правилу цепочки и может быть полезна для учета и протоколирования работы. Операция **RETURN** прерывает прохождение кадра через данную цепочку и возвращает его в точку вызова (предыдущую цепочку). Дополнительные операции arptables описаны в параграфе 5.3.5 (стр. 153).

5.3.4 Команды и параметры arptables

Программа arptables позволяет задать в командной строке различные команды, опции и параметры, которые могут служить для создания, переименования и удаления правил и цепочек, а также для просмотра информации о

1 *Address Resolution Protocol - протокол преобразования адресов, описанный в RFC 826.*
2 *Это необязательно, поскольку некоторые операции могут возвращать кадр в цепочку для передачи следующему правилу из списка.*
3 *Эта цепочка не поддерживается в ядрах серии 2.4.*
4 *В последнем случае для встроенных цепочек по отношению к пакету выполняется операция, заданная политикой цепочки (стр. 152)*

текущем состоянии.

5.3.4.1 Основные команды arptables

Команды arptables используются для создания, изменения, просмотра и удаления правил в цепочках. Поскольку программа поддерживает единственную таблицу filter, необходимости использования опции -t не возникает, хотя она и поддерживается программой. В каждой строке можно использовать только одну команду arptables[1].

-A, --append
добавляет правило в конец указанной цепочки.

-D, --delete
удаляет правило из указанной цепочки. Удаляемые правила можно задать номером[2] (или диапазоном номеров в формате start_nr:end_nr) или ввести в строке точную копию спецификации, заданной при добавлении правила в таблицу.

-I, --insert
вставляет правило в указанную номером строку списка. Если в цепочке имеется N правил, в качестве параметра команды I можно использовать значения от -N до N+1. Для случая положительных значений i, позиции с номером i-N-1 и i. Нулевое значение задает вставку правила вслед за последним из имеющихся в таблице правил как по команде -A.

-R, --replace
Меняет спецификацию правила в указанной строке. Изменяемая строка задается номером, интерпретация отрицательных значений идентичная используемой в команде -I.

-P, --policy
задет политику для указанной встроенной цепочки (ACCEPT, DROP или RETURN). Заданная политикой операция используется, когда кадр доходит до конца встроенной цепочки.

-F, --flush
сбрасывает все правила указанной цепочки, не изменяя политики цепочки.

-Z, --zero
сбрасывает для указанной цепочки значения счетчиков пакетов и байтов в 0. Если цепочка не указана, сбрасываются значения счетчиков для всех цепочек таблицы. Команду -Z можно использовать совместно с командой -L для просмотра значений счетчиком с последующим сбросом этих значений.

-L, --list
выводит список команд указанной цепочки. Если цепочка не задана в команде, выводятся правила всех цепочек.

-N, --new-chain
создает новую пользовательскую цепочку с указанным именем (до 31 символа). Количество пользовательских цепочек не ограничено.

-X, --delete-chain
удаляет указанную пользовательскую цепочку. Удаляемая цепочка не должна использоваться в качестве операции (действия) в какой-либо из остающихся цепочек. Если в команде не указано имя цепочки, программа будет удалять все неиспользуемые пользовательские цепочки данной таблицы.

-E, --rename-chain
переименовывает указанную цепочку. В отличие от iptables программа ebtables позволяет менять имена не только у пользовательских, но и у встроенных цепочек. Например, вы можете переименовать в PREBRIDGING цепочку PREROUTING, с помощью команды **-E PREROUTING**. Переименование цепочек не оказывает никакого влияния на работу **arpables**[3].

5.3.4.2 Дополнительные команды

-V, --version
выводит информацию о номере версии программы arptables.

-h, --help
выдает краткую справку о синтаксисе команд arptables.

-j, --jump target
служит для задания действия, выполняемого правилом при соответствии пакета заданным условиям. В качестве действий могут служить основные операции ACCEPT, DROP, CONTINUE, RETURN (см. параграф 5.3.3 на стр. 151), дополнительные операции (см. параграф 153 на стр. 153) или пользовательские цепочки.

5.3.4.3 Опции спецификации правил

Описанные ниже опции служат для задания условий соответствия, используемых при отборе пакетов. Знак инверсии **!** перед той или оной опцией меняет смысл условия на обратный (логическое отрицание).

-s, --source-ip [!] address[/mask]

1 *Единственным исключением является команда -Z, которую можно использовать вместе с программой просмотра списка -L.*
2 *Для просмотра номеров используйте команду -L. В командах удаления можно использовать отрицательные значения номеров, смыл которых разъясняется в описании команды -I.*
3 *Если вы не забудете изменить соответствующие параметры в ссылающихся на эту цепочку правилах.*

позволяет проверить соответствие IP-адреса отправителя указанному адресу или диапазону адресов.

-d, --destination-ip [!] address[/mask]
позволяет проверить соответствие IP-адреса получателя указанному адресу или диапазону адресов.

--source-mac [!] address[/mask]
позволяет проверить соответствие MAC-адреса отправителя указанному значению. Для задания адреса и маски используется 6 шестнадцатеричных значений, разделенных двоеточием (:).

--destination-mac [!] address[/mask]
позволяет проверить соответствие MAC-адреса получателя указанному значению. Для задания адреса и маски используется 6 шестнадцатеричных значений, разделенных двоеточием (:).

-i, --in-interface [!] name
позволяет проверить интерфейс, через который кадр был принят (для цепочек INPUT и FORWARD). Допускается использование синонима **--in-if**.

-o, --out-interface [!] name
позволяет проверить интерфейс, через который кадр будет передаваться (для цепочек OUTPUT и FORWARD). Допускается использование синонима **--out-if**.

-l, --h-length length[/mask]
позволяет проверить размер пакета в байтах.

--opcode code[/mask]
позволяет проверить значение 2-байтового поля кода операции. Допустимы значения:

1=Request (запрос)

2=Reply (отклик)

3=Request_Reverse (запрос обратного преобразования)

4=Reply_Reverse (отклик на запрос обратного преобразования)

5=DRARP_Request (запрос DRARP)

6=DRARP_Reply (отклик на запрос DRARP)

7=DRARP_Error (ошибка DRARP)

8=InARP_Request (запрос InARP)

9=ARP_NAK

--h-type type[/mask]
позволяет проверить значение типа оборудования. Двухбайтовое поле типа оборудования может принимать единственное шестнадцатеричное значение

1=Ethernet

--proto-type type[/mask]
позволяет проверить 2-байтовое значение типа протокола. Допустимым значением типа является 0x800=IPv4.

5.3.5 Дополнительные операции arptables

Дополнительные операции arptables реализованы в пользовательском пространстве программы и не требуют загрузки каких-либо модулей arptables в отличие от программы iptables. Однако для использования этих операций при компиляции ядра должна быть включена опция **ARP tables support** (параграф 4.4.2.2.14.2.18 на стр.80). Если для опции было выбрано значение M, потребуется также загрузить модуль ядра **arp_tables**.

--mangle-ip-s IP-адрес
устанавливает заданное значение в поле IP-адреса отправителя. Требует наличия в ядре опции **ARP payload mangling** (стр. 80). Если для опции было выбрано значение M, потребуется также загрузить модуль ядра **arpt_mangle**.

--mangle-ip-d IP-адрес
устанавливает заданное значение в поле IP-адреса получателя. Требует наличия в ядре опции **ARP payload mangling** (стр. 80). Если для опции было выбрано значение M, потребуется также загрузить модуль ядра **arpt_mangle**.

--mangle-mac-s MAC-адрес
устанавливает заданное значение в поле MAC-адреса получателя. Требует наличия в ядре опции **ARP payload mangling** (стр. 80). Если для опции было выбрано значение M, потребуется также загрузить модуль ядра **arpt_mangle**.

--mangle-mac-d MAC-адрес
устанавливает заданное значение в поле MAC-адреса отправителя. Требует наличия в ядре опции **ARP payload mangling** (стр. 80). Если для опции было выбрано значение M, потребуется также загрузить модуль ядра **arpt_mangle**.

--mangle-target операция
задает операцию, используемую для пакета после его изменения с помощью одной из перечисленных выше дополнительных операций. В качестве действия может служить любая из основных операций arptables (параграф 5.3.3 на стр. 151). По умолчанию для измененных пакетов используется операция **ACCEPT**.

6 Организация соединений VPN

Для современного бизнеса организация соединений VPN через сети общего пользования и Internet стала насущной потребностью. Linux обеспечивает достаточно мощные средства организации и поддержки VPN-соединений, которые мы вкратце рассмотрим в следующих параграфах.

6.1 FreeS/WAN IPsec VPN

http://www.freeswan.org/

FreeS/WAN представляет собой Linux-реализацию протокола IPsec (IP security), обеспечивающего шифрование и аутентификацию на сетевом (IP) уровне стека протоколов. IPsec может защитить весь передаваемый по протоколу IP трафик в отличие от других методов, которые защищают только отдельные протоколы вышележащих уровней[1]. Такой подход обеспечивает преимущества, но и порождает некоторые ограничения.

Протокол IPsec можно использовать на любой машине, поддерживающей работу в сетях IP. Для защиты корпоративного трафика можно использовать выделенные шлюзы IPsec. Протокол часто используется на маршрутизаторах, межсетевых экранах и различных серверах приложений, а также пользовательских станциях и переносных компьютерах.

В IPsec применяются два протокола:

- ESP (Encapsulating Security Payload) для шифрования и аутентификации;
- IKE (Internet Key Exchange) для согласования параметров (включая ключи), используемых ESP.

Реализация FreeS/WAN IPsec включает три основных компоненты:

- KLIPS (kernel IPsec) реализует протокол ESP и обеспечивает обработку пакетов в ядре;
- Pluto (демон IKE) реализует протокол IKE, согласующий соединения с другими системами;
- набор сценариев для обеспечения административного интерфейса.

Использование IPsec не является обязательным для протокола IPv4. FreeS/WAN обеспечивает поддержку протокола IPsec для сетевого стека Linux IPv4, ведутся также работы по интеграции FreeS/WAN со стеком Linux IPv6.

Протокол IPsec разработан с учетом обеспечения интероперабельности различных реализаций. FreeS/WAN IPsec может работать с широким спектром продукции других фирм[2].

Протокол IPsec имеет целый ряд преимуществ, о которых написано много книг и статей[3]. Поскольку протокол полностью реализован на сетевом уровне, для него характерен высокий уровень гибкости и возможность использования практически для всех типов трафика Internet. Основными областями применения IPsec являются:

- виртуальные частные сети VPN (Virtual Private Network), обеспечивающие безопасный обмен данными между сайтами по открытым каналам Internet;
- системы безопасного доступа удаленных пользователей в корпоративные сети.

FreeS/WAN IPsec поддерживает оба варианта использования IPsec. Кроме того, используемые программой методы шифрования позволяют любой паре шлюзов FreeS/WAN шифровать весь передаваемый между ними трафик.

Создание безопасных туннелей VPN

VPN или виртуальная частная сеть (Virtual Private Network) позволяет организовать безопасное шифрованное соединение между сетями с использованием открытых и не обеспечивающих безопасности каналов (например, Internet). Установленные в каждой из сетей шлюзы VPN обеспечивают шифрование всего трафика, передаваемого в другую сеть через открытый канал. Такое соединение называют туннелем VPN.

Использование более сложных методов криптографии усложняет реализацию и повышает уровень требований к администраторам шлюзов, но повышает уровень безопасности туннелей. Две разделенных сети могут работать как единая безопасная сеть, несмотря на то, что часть трафика передается по туннелям через небезопасные внешние сети.

Реальная структура VPN обычно сложнее. Организация может иметь множество филиалов, а также партнеров, заказчиков, которым нужно обеспечить безопасный доступ к корпоративным ресурсам. Может потребоваться организация VPN и внутри корпоративной сети для передачи конфиденциальной информации.

Организацию и поддержку VPN упрощает то, что в большинстве случаев приходиться иметь дело со статическими параметрами. Обычно IP-адреса хостов, причастных к организации виртуальной частной сети известны и изменяются в процессе работы. Это сильно упрощает работу администратора.

Удаленный доступ

Кроме организации туннелей VPN между сайтами зачастую требуется обеспечить безопасный доступ в корпоративную сеть для работающих за пределами офиса сотрудников. В этом случае задача осложняется тем, что удаленные пользователи зачастую не имеют статических адресов IP. Кроме того, это может вызывать некоторые осложнения:

1 *PGP для электронной почты, SSH для удаленного доступа (login), SSL для web и т.п.*
2 *Список совместимых реализаций можно найти на сайте http://www.freeswan.org/freeswan_trees/freeswan-2.06/doc/interop.html#interop*
3 *См., например, раздел Encryption &VPNs на сайте SANS (http://www.sans.org/rr/).*

- удаленные пользователи зачастую получают адреса по протоколу DHCP и FreeS/WAN не имеет информации об окончании срока действия выделенного пользователю адреса. В результате туннельные соединения могут разрываться и единственным методом решения проблемы является повторная организация туннелей.
- трансляция адресов (NAT) между шлюзами IPsec препятствует работе протокола IPsec, поскольку протокол использует сквозную аутентификацию пакетов во избежание подмены маршрутов, а NAT изменяет пакеты.

Однако, несмотря на упомянутые сложности FreeS/WAN обеспечивает в большинстве случаев эффективный и безопасный доступ для удаленных пользователей.

6.1.1 Программа ipsec

Программа **ipsec** управляет работой нескольких утилит, вовлеченных в процессы аутентификации и шифрования, передавая им опции и параметры, как это обычно делается в случаях прямого обращения к программам. Такой подход существенно упрощает работу и позволяет избавиться от множества ошибок при передаче параметров различным программам, а также обеспечивает необходимый уровень централизации.

В частности, вызывает используемые утилиты с подходящими значениями переменных окружения IPSEC_DIR, IPSEC_CONFS, IPSEC_VERSION, включающими полные имена каталогов, где хранятся утилиты IPsec, конфигурационные файлы и номер версии IPsec. Ниже кратко рассматриваются параметры и опции команды IPsec.

`--help`
выводит краткую информацию о поддерживаемых командах. Многие команды ipsec имеют собственные man-руководства (например, ipsec_auto(8) для auto).

`--version`
выводит информацию об имени и номере версии FreeS/WAN, а также авторских правах. Номер указывается в формате Uxxx/Kyyy - первая часть указывает версию пользовательских утилит, а вторая - версию модулей ядра. Если номера версий совпадают, указывается только одно значение.

`--versioncode`
выводит информацию о номере версии программы. Эта команда используется для проверки совместимости с другими программами и сценариями.

`--copyright`
выводит информацию об авторских правах.

`--directory`
выводит полное имя каталога, в котором хранятся утилиты IPsec.

`--confdir`
выводит полное имя каталога, в котором хранятся конфигурационные файлы IPsec.

6.1.2 Настройка параметров соединений IPsec VPN

Параметры работы программы и организации соединений VPN определяются обычно конфигурационным файлом ipsec.conf, подробно описанном в Приложении 12.19 (стр. 422). Количество конфигурационных параметров достаточно велико и при их настройке могут встретиться определенные сложности.

Для упрощения настройки FreeS/WAN VPN можно использовать один из модулей для программы Webmin (параграф 11.3 на стр. 228). Web-интерфейс (см. рисунки 6.1 и 6.2) этих модулей позволяет создавать и настраивать соединения VPN для локальных и удаленных машин.

Рисунок 6.1 Настройка FreeS/WAN VPN с помощью Webmin

6.1.3 Известные проблемы ipsec

При выборе для параметров **type** (стр. 423) или **failureshunt** (стр. 425) значения **drop** или **reject** FreeS/WAN блокирует исходящие пакеты с использованием **eroutes**, но предполагает, что входящий трафик обрабатывается межсетевым экраном. FreeS/WAN поддерживает для брандмауэра специальные ловушки (firewall hook) на основе сценариев **updown**. Однако используемый по умолчанию сценарий **ipsec _updown** не поможет при управлении брандмауэром, обслуживающим модемы.

Включение атрибутов канала обмена ключами[1] (методы аутентификации, ikelifetime и т. п.) в качестве атрибутов соединения[2] является весьма сомнительным способом, который может повлечь за собой возникновение проблем.

Программа ipsec_manual, используемая при управлении соединениями с генерацией ключей вручную, имеет не столь удобный синтаксис задания подсетей, адресов и т. п., как обычный пользовательский интерфейс FreeS/WAN. Для всех адресов должна использоваться стандартная форма представления адресов IP в виде 4 десятичных значений, разделенных точками.

Возможность использования различных подтверждений подлинности (identity), способов аутентификации (authby) и публичных ключей (public key) для разных соединений с автоматической генерацией ключей для одной пары участников является весьма обманчивой. В таких случаях могут возникать проблемы, поскольку тождественность (identity) участников невозможно проверить на достаточно ранних этапах. Особенно неудобно это делать для систем удаленного доступа (Road Warrior), где удаленный IP-адрес задается как 0.0.0.0.

Может потребоваться контроль значения MTU[3] для каждого интерфейса, поскольку опция **overridemtu** позволяет изменить глобальное значение этого параметра.

Достаточно большое число функций, которые были бы полезны в ручном и автоматическом режиме генерации ключей, еще не реализованы в программе.

1 *keying channel*
2 *Вместо из указания только для пары участников обмена ключами.*
3 *Максимальный размер пакета.*

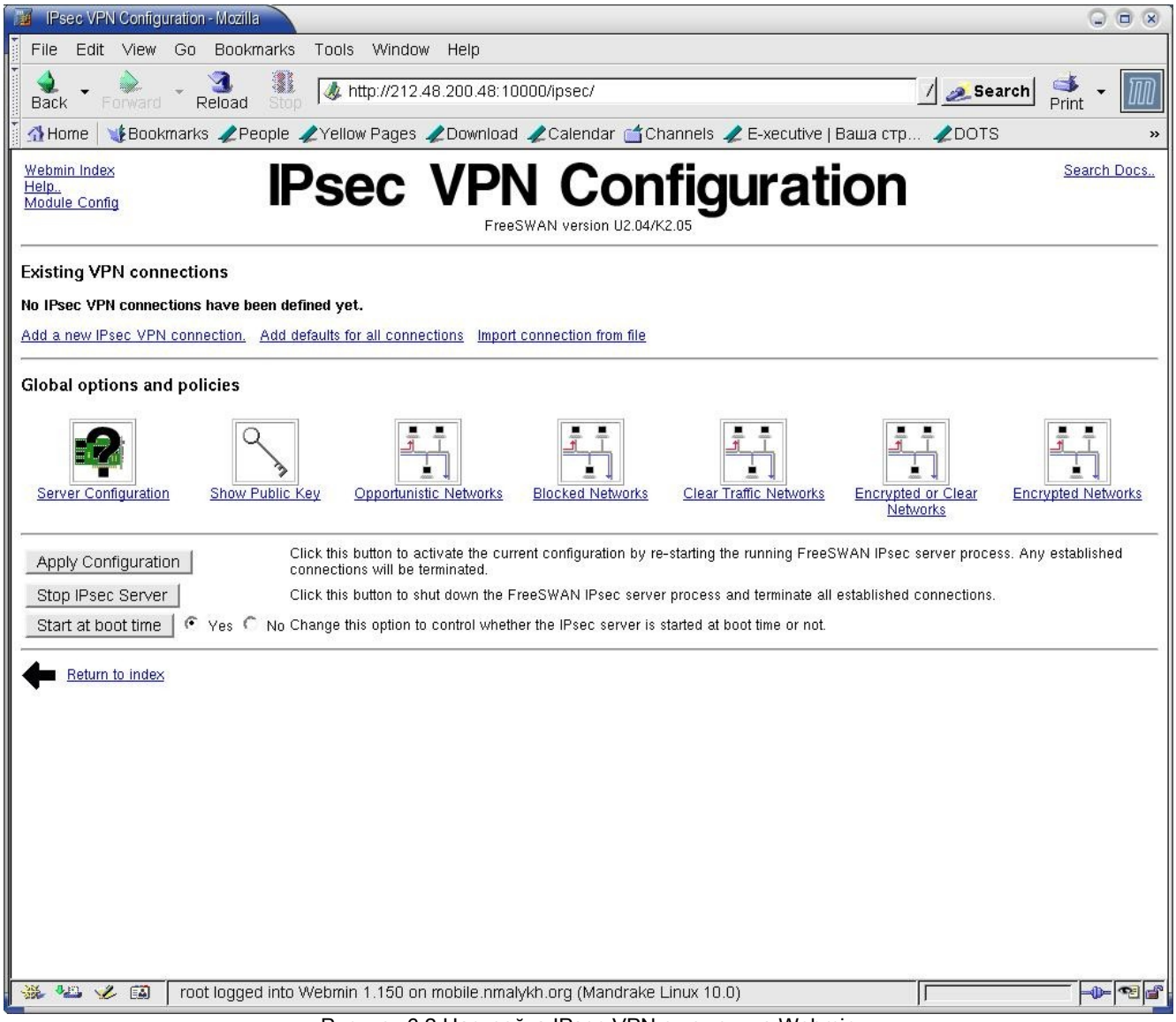


Рисунок 6.2 Настройка IPsec VPN с помощью Webmin

При организации соединения до того, как станет доступен сервер DNS использование параметров **left=FQDN**[1], **leftnextop=FQDN** и **leftrsasigkey=%dnsonload** будет приводить к отказам.

Программа генерации ключей **ipsec_pluto** не имеет дел с публичным ключом локальной стороны соединения, но в момент добавление может отсутствовать информация о том, которая сторона является локальной.

Опция **myid** (стр. 426) не оказывает влияния на команды **ipsec auto --add** и **ipsec auto --replace** для неявнных соединений (параграф 12.19.2.1 на стр. 427)..

6.2 SSL VPN

Реализация соединений VPN на основе протокола SSL в системах Linux обеспечивается с помощью пакета **stunnel**, включаемого в состав большинства современных дистрибутивов.

Программа stunnel обеспечивает SSL-шифрование для обмена трафиком между удаленным клиентом и локальным (запускается из inetd/xinetd) или удаленным сервером. Туннели позволяют демонам вашей системы, не поддерживающим SSL организовывать безопасные SSL-соединения с клиентами.

Программу stunnel можно использовать для реализации функций SSL с загружаемыми через inetd (например, POP-2, POP-3, IMAP) или автономными (NNTP, SMTP, HTTP) демонами, а также для организации туннелей PPP через сетевые сокеты без изменения программного кода. Программа использует криптографические модули, разработанные Эриком Юнгом (Eric Young - eay@cryptsoft.com).

6.2.1 Опции команды stunnel

`-h`

выводит краткую справку о программе.

`-V`

выводит номер версии stunnel и параметры компиляции.

`-D level`

1 *Fully Qualified Domain Name - полное доменное имя.*

задает уровень отладочных сообщений. Параметр level может принимать одно из значений, принятых в программе syslog (см. параграф 2.8.4.2 на стр. 50) - emerg (0), alert (1), crit (2), err (3), warning (4), notice (5), info (6) или debug (7). Максимальный уровень отладочной информации обеспечивает опция **-D debug** или **-D 7**. По умолчанию используется уровень notice (5).

Если тип сообщения (syslog facility) не задан, будет использоваться тип **authpriv** (см. параграф 2.8.4.1 на стр. 50). Для имен типов и уровня сообщений регистр символов не имеет значения.

-O a|l|r:option=value[:value]
задает опции восприятия (accept), локального и удаленного сокетов.

Опция **linger** может принимать значения **l_onof:l_linger**, опция **time** - **tv_sec:tv_usec**.

Примеры

-O l:SO_LINGER=1:60 - задает тайм-аут (1 минута) для закрытия локального сокета.

-O r:TCP_NODELAY=1 - отключает алгоритм Nagle для удаленных сокетов.

-O r:SO_OOBINLINE=1 - помещает данные out-of-band непосредственно в поток принимаемых данных для удаленных сокетов.

-O a:SO_REUSEADDR=0 - запрещает повторное использование адресов (по умолчанию разрешено)

-O a:SO_BINDTODEVICE=lo - разрешает восприятие соединений только loopback-интерфейсом

Список поддерживаемых вашей версией опций можно получить с помощью команды **stunnel -V**.

-o file
добавляет сообщение в конец журнального файла.

-C cipherlist
выбирает разрешенные шифры SSL. Для разделения шифров в списке разрешенных используется двоеточие (:). Примером такого списка может служить DES-CBC3-SHA:IDEA-CBC-MD5

-c client mode
переводит stunnel в режим клиента (удаленный доступ с использованием). По умолчанию используется режим сервера.

-T transparent proxy mode
задает режим прозрачного proxy, при котором адрес переписывается так, чтобы он выглядел, как при подключении с клиента SSL, а не с машины, на которой работает программа stunnel. Этот режим может использоваться только в некоторых операционных системах (в частности, Linux) и только на сервере stunnel. Отметим, что эта опция не будет объединяться с режимом proxy (**-r**), если используемый клиентом по умолчанию маршрут к целевой машине не проходит через stunnel.

-p pemfile
имя файла PEM, содержащего приватный ключ и сертификат. PEM всегда используется в режиме сервера. Флаг **-p** в клиентском режиме будет приводить к использованию заданной цепочки в качестве цепочки сертификата клиентской стороны. Использование сертификата на клиентской стороне не является обязательным. Используемые сертификаты должны иметь формат PEM и сортироваться по уровню, начиная с высшего (root CA).

-v level
проверяет сертификат партнера в соответствии с указанным уровнем:

- 1 - проверка наличия сертификата
- 2 - проверка сертификата;
- 3 - проверка партнера с использованием локального сертификата.

По умолчанию сертификат партнера не проверяется.

-a directory
задает каталог с сертификатами клиентов. В этом каталоге **stunnel** будет искать сертификаты при использовании опции **-v**. Отметим, что сертификаты в этом каталоге должны иметь имена вида **XXXXXXXX.0**, где **XXXXXXXX** представляет собой хэш-значение для сертификата. Этот параметр отменяет заданное при компиляции имя каталога сертификатов (см. команду **stunnel -V**).

-A certfile
указывает файл Certificate Authority, в котором содержатся сертификаты CA, используемые с опцией **-v**. Значение этого параметра отменяет заданное при компиляции имя файла (см. **stunnel -V**).

-S sources
задает используемый по умолчанию источник сертификатов. Как **stunnel**, так и SSL-библиотека, использованная при компиляции **stunnel**, используют принятое по умолчанию местоположение для поиска вашего сертификата (опция **-A**) и каталога сертификатов (опция **-a**). Флаг **-S** позволяет задать, который из принятых по умолчанию источников сертификатов следует использовать.

0 = игнорировать все принятые по умолчанию источники сертификатов;

1 = использовать источники ssl-библиотеки;

2 = использовать источники stunnel;

3 = использовать источники ssl-библиотеки и stunnel**.**

Принятые по умолчанию параметры вы можете узнать с помощью команды **stunnel -V.**

Отметим, что значения параметров **-A** и **-a** используются взамен принятых по умолчанию источников **stunnel**, но в дополнение к источникам библиотеки **ssl** (если параметр **-S** разрешает их использовать).

Чтобы впоследствии не пришлось ломать голову, используйте **-S 0** и явно задавайте параметры **-A** и/или **-a**.

-t timeout
задает тайм-аут кэша для сеанса. По умолчанию используется период 300 секунд (5 минут).

-N servicename
имя сервиса, используемое для tcpwrapper. Если это имя не указано, соответствующее значение будет генерироваться автоматически.

-u ident_username
задает использование IDENT (RFC 1413) для проверки имени пользователя.

-n proto
задает согласование SSL с указанным протоколом. В настоящее время поддерживаются протоколы smtp, pop3, nntp.

-E socket
указывает сокет демона EGD (Entropy Gathering Daemon) используемого в OpenSSL для генерации случайных чисел[1].

-R filename
задает имя файла, содержащего случайные данные. Библиотека SSL будет использовать этот файл первым при генерации случайных чисел.

-W
блокирует переписывание “затравочных” файлов.

-B bytes
задает число байтов, читаемых из затравочных файлов. Для версий SSL до 0.9.5a этот параметр также определяет размер данных, требуемых для генерации PRNG. Более поздние версии OpenSSL самостоятельно проверяют “степень случайности”.

-I host
задает применение IP-адреса исходящего интерфейса в качестве адреса отправителя для удаленных соединений.

-d [host:]port
задает работу в режиме демона, прослушивающего соединения по адресу **[host:]port**. Если хост не задан, прослушиваются все IP-адреса локального хоста. По умолчанию режим демона отключен и программа запускается из inetd (xinetd).

-f foreground mode
задает работу в режиме foreground и вывод сообщений на **stderr** вместо их передачи **syslog** (если не используется опция **-o**). По умолчанию фоновый процесс используется в режиме демона.

-l program [-- programname [arg1 arg2 arg3...]]
выполнить локальную программу inetd-типа.

-L program [-- programname [arg1 arg2 arg3...]]
открыть локальный порт pty и выполнить программу.

-s username
установить идентификатор пользователя (setuid) в режиме демона.

-g groupname
установить идентификатор группы (setgid) в режиме демона, сбросив все прочие группы.

-P { dir/ | file | none }
указывает местоположение PID-файла. Если значение параметра заканчивается символом /, в указанном каталоге создается файл с именем **stunnel.servicename.pid**. Если параметр содержит полное имя файла (в конце нет символа /), создается файл с заданным параметром именем. Если параметр имеет значение **none**, PID-файл не создается.

-r [host:]port
подключиться к удаленному сервису. Если параметр не задает хост, соединение организуется с локальным хостом.

Примеры

Для использования SSL-инкапсуляции на локальном сервере **imapd**, служит команда

```
stunnel -d 993 -l /usr/sbin/imapd -- imapd
```

Если вы хотите организовать туннелирование для демона **pppd** с использованием порта 2020, подойдет команда

```
stunnel -d 2020 -L /usr/sbin/pppd -- pppd local
```

Параметры среды

Если команда stunnel используется для создания локального процесса с параметром **-l** или **-L**, устанавливаются перечисленные ниже параметры окружения.

REMOTE_HOST

1 *Эта опция поддерживается только при компиляции с OpenSSL 0.9.5a и более поздних версий.*

IP-адрес удаленной стороны соединения.

SSL_CLIENT_DN

значение DN (Distinguished Name - имя субъекта) сертификата партнера, если этот сертификат имеется и проверен.

SSL_CLIENT_I_DN

Значение Issuer DN для сертификата партнера, если этот сертификат имеется и проверен.

Сертификаты

Каждый поддерживающий SSL демон должен представить партнеру корректный сертификат X.509. Требуется также приватный ключ для расшифровки принимаемых данных. Простейшим способом получения сертификата и ключа является генерация с помощью свободно распространяемого пакета **openssl**.

При генерации пар сертификат-ключ для stunnel нужно принимать во внимание два аспекта.

1) Приватные ключи не могут быть зашифрованы, поскольку сервер не имеет возможности получить от пользователя пароль. Для получения нешифрованного ключа следует использовать команду **req -nodes** (из пакета openssl.

2) Порядок содержимого файла .pem имеет важное значение. Сначала в файле должен быть указан приватный ключ, а после него - подписанный сертификат (не запрос сертификата). После приватного ключа и сертификата должна присутствовать по крайней мере одна пустая строка. Текстовая информация в верхней части сгенерированного сертификата должна быть отброшена. В результате файл должен иметь вид:

```
-----BEGIN RSA PRIVATE KEY-----
[кодированный ключ]
-----END RSA PRIVATE KEY-----
[пустая строка]
-----BEGIN CERTIFICATE-----
[кодированный сертификат]
-----END CERTIFICATE-----
[пустая строка]
```

Генерация случайных чисел

Программе **stunnel** требуется “затравка” PRNG[1] для SSL, позволяющая обеспечить эффективную генерацию случайных чисел. Перечисленные ниже источники загружаются в указанном здесь порядке до тех пор, пока не будет собран достаточный для генерации случайных чисел объем информации:

- файл, указанный с помощью опции **-R**[2];
- файл, заданный переменной окружения **RANDFILE** (если такая переменная присутствует);
- файл **.rnd** в домашнем каталоге пользователя, если переменная **RANDFILE** не задана.
- Файл, указанный опцией **--with-random** при компиляции программы;
- содержимое экрана[3] (при использовании в среде Windows);
- egd-сокет, заданный флагом **-E**;
- egd-сокет, указанный опцией **--with-egd-sock** во время компиляции;
- устройство **/dev/urandom**.

В новых версиях OpenSSL (>=0.9.5a) загрузка случайных данных прекращается автоматически после повышения энтропии до требуемого уровня. Более ранние версии будут продолжать собирать информацию из перечисленных источников, пока функция SSL не завершит работу.

При наличии устройства **/dev/urandom** OpenSSL имеет свойство использовать его в качестве затравки PRNG, несмотря на то, что устройство это находится в самом конце списка источников случайных данных[4].

Ограничения

Программу stunnel невозможно использовать с демоном FTP, поскольку протокол FTP использует при работе несколько соединений через различные порты. Однако существуют версии демонов FTP и telnet, поддерживающие SSL.

1 *Pseudo random number generator - генератор псевдослучайных чисел.*
2 *Заданный опцией **-R** файл должен содержать случайную информацию. Это означает, что содержимое файла должно меняться при каждом запуске **stunnel**. Замена содержимого файла происходит автоматически, если вы не используете флаг **-W**. Если вы предпочитаете обновлять содержимое файла самостоятельно, полезна будет программа **rand**, включенная в последние версии OpenSSL.*
3 *Отметим, что на Windows-машинах при отсутствии действий пользователя (перемещение мыши, открытие окон и т. п.) содержимое экрана не обеспечивает достаточного количества пеерменных составляющих, поэтому в таких случаях требуется задать случайный файл, используя флаг -R.*
4 *Это особенность OpenSSL, а не stunnel.*

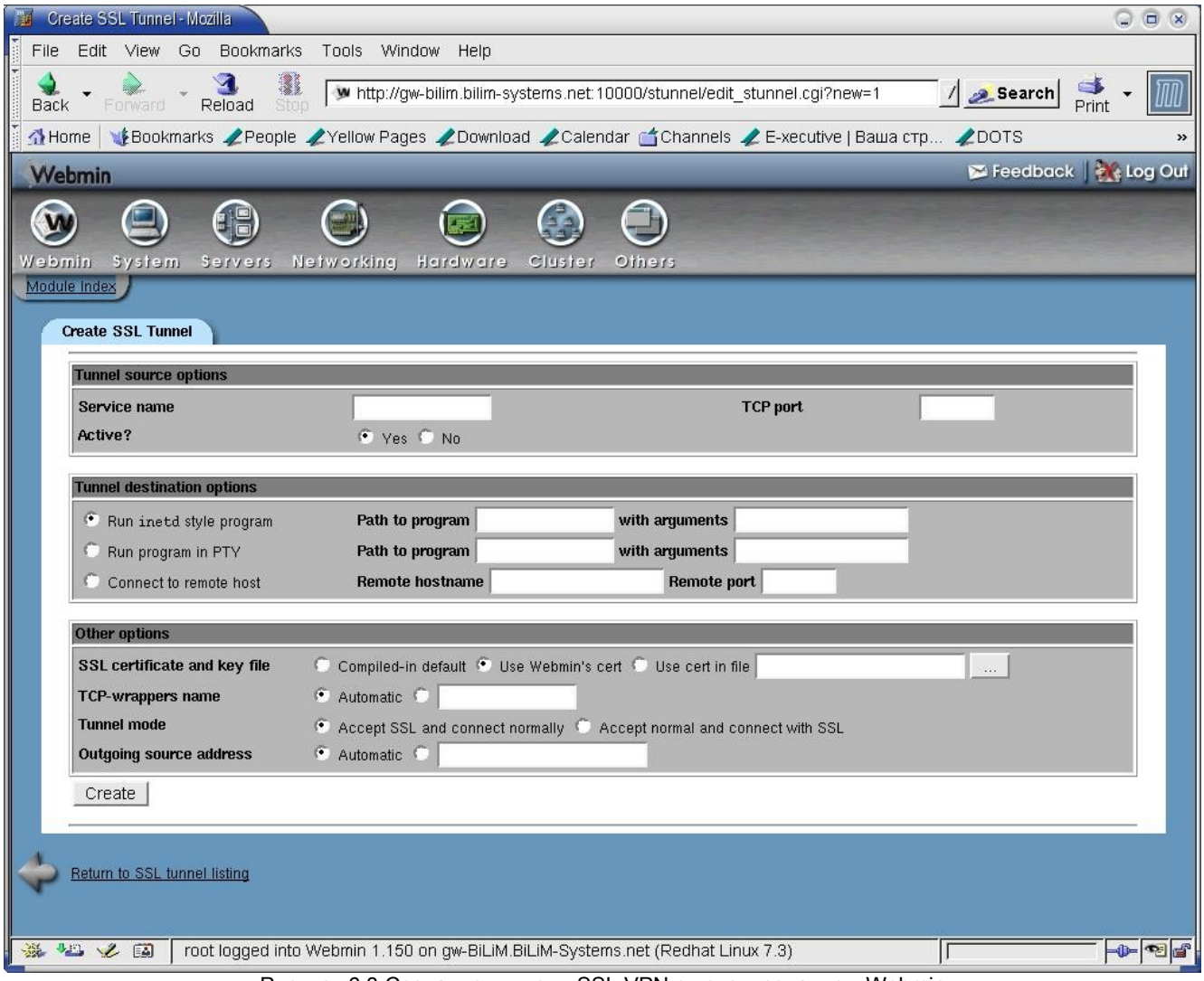


Рисунок 6.3 Создание туннеля SSL VPN с использованием Webmin

6.3 PPTP VPN

Протокол PPTP[1] был разработан консорциумом с участием компании Microsoft и используется для организации туннелей VPN через сеть Internet. Такие туннели обеспечивают пользователям недорогой и безопасный доступ в корпоративные сети из любого места, где имеется доступ в Internet.

PPTP использует модель клиент-сервер для организации соединений VPN. Многие ОС компании Microsoft включают в комплект поставки клиентский модуль PPTP, поэтому не возникает необходимости в приобретении дополнительных клиентских программ. Одним из основных преимуществ протокола PPTP по сравнению с другими технологиями VPN является простота установки и настройки.

6.3.1 Сервер

pptpd представляет собой демон Poptop PPTP, который обслуживает туннельные соединения PPP, инкапсулируемые в GRE с использованием протокола PPTP VPN. Демон может поддерживать функции управления адресами IP или TCP wrapper, если при компиляции были активизированы соответствующие опции.

6.3.1.1 Опции команды pptpd[2]

`-b (--bcrelay) internal-interface`
указывает необходимость трансляции клиентам всех широковещательных пакетов, полученных внутренним интерфейсом сервера.

`-c (--conf) conf-file`
задает конфигурационный файл демона pptpd (по умолчанию используется файл /etc/pptpd.conf)

`-d (--debug)`
включает режим отладки, передающий отладочные сообщения демону syslog (см. параграф 2.8.4 на стр. 49).

`-f (--fg)`
выводит демон pptpd из фонового режима в режим foreground (по умолчанию демон работает как фоновая задача).

`-h (--help)`
выводит на экран краткую справку.

1 *Point to Point Tunneling Protocol (протокол организации туннелей “точка-точка”).*
2 *В скобках указаны полные варианты опций.*

-l (--listen) x.x.x.x
задает IP-адрес локального интерфейса для приема пакетов.

-o (--option) ppp-conf-file
указывает демону pptpd на необходимость использования альтернативного конфигурационного файла для демона ppp (обычно по умолчанию используется /etc/ppp/options, но имя файла зависит от демона ppp).

-p (--pidfile) pid-file
задает альтернативное имя файла с идентификатором процесса PID (по умолчанию идентификатор хранится в файле /var/run/pptpd.pid).

-s (--speed) baud
задает значение скорости, передаваемое демону ppp как скорость работы используемого терминала tty (в некоторых случаях демон ppp может игнорировать это значение).

-t (--stimeout) seconds
определяет для первого пакета значение тайм-аута, по истечение которого пакет отбрасывается. Эта опция может быть полезна для защиты от DOS-атак.

-v (--version)
выводит номер версии демона pptpd.

Пример использования

Команда **pptpd** будет активизировать демон Poptop PPTP VPN (который сразу же отключиться от терминала и возвратит управление демону pptpd), используя принятые по умолчанию опции (конфигурация из файла /etc/pptpd.conf, идентификатор процесса в файле /var/run/pptpd.pid).

С помощью команды

```
pptpd --debug -l 192.168.0.1 -fg -c /etc/pptpd.conf.test -p /etc/pptpd.pid
```

будет активизироваться демон Poptop PPTP VPN, прослушивающий пакеты на интерфейсе с адресом 192.168.0.1. Демон работает в режиме **foreground** с использованием конфигурационных параметров из файла **/etc/pptpd.conf.test**, выдает отладочную информацию и записывает идентификатор процесса в файл **/etc/pptpd.pid**.

6.3.1.2 Конфигурационный файл pptpd.conf

При загрузке программа **pptpd** сначала считывает параметры из конфигурационного файла (по умолчанию **/etc/pptpd.conf**). Это позволяет администратору задать диктуемое задачами поведение демона Poptop PPTP VPN. Заданные конфигурационным файлом установки могут быть изменены описанными выше опциями командной строки (за исключением локального и удаленного адресов IP, которые нельзя задать в командной строке).

6.3.1.2.1 Опции

speed <значение>
определяет скорость (байт/сек) для передачи демону PPP в качестве значения скорости интерфейса для пары tty/pty. Заданное этой опцией значение скорости некоторые демоны PPP могут игнорировать. Эта опция эквивалентна передаваемой в командной строке опции **-s** (**--speed**). По умолчанию используется значение скорости 115200 байт/сек, которое некоторые реализации интерпретируют как “неограниченную скорость”.

option <файл опций>
задает имя файла опций, передаваемого демону PPP взамен стандартного файла опций PPP[1]. Эта опция эквивалентна опции командной строки **-o** (**--option**).

stimeout <seconds>
Демон Poptop имеет встроенные средства защиты от DOS-атак. Одним из таких средств является захват первого пакета в соединении и выполнение некоторых проверок до того, как соединение будет организовано. Опция stimeout определяет интервал времени, в течение которого Poptop может удерживать пакет. Обычно по умолчанию используется время удержания 10 секунд. Эта опция эквивалентна опции командной строки **-t** (**--stimeout**).

debug
включает режим отладки, передающий информацию демону syslog (см. параграф 2.8.4 на стр. 49). Эта опция эквивалентна параметру командной строки **-d** (**--debug**).

bcrelay internal-interface
включает режим широковещательной трансляции (broadcast relay mode), при котором клиентам пересылаются все широковещательные пакеты, полученные внутренним интерфейсом. Эта опция эквивалентна параметру командной строки **-b** (**--bcrelay**).

1 *Обычно файл /etc/ppp/options*

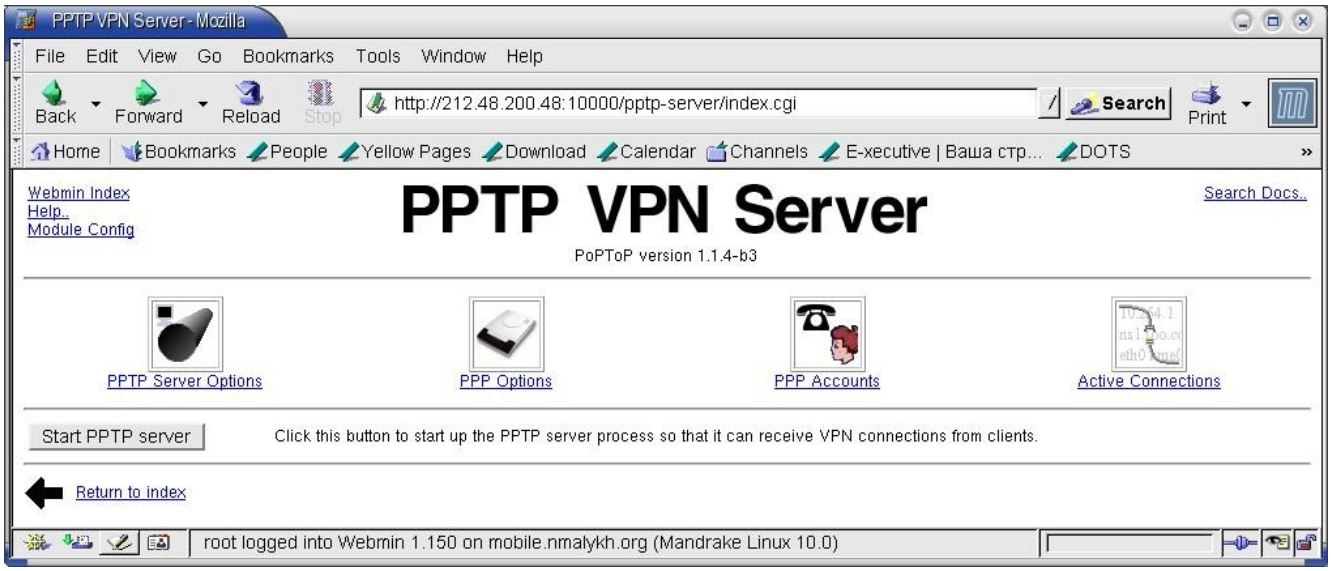


Рисунок 6.4 Настройка сервера PPTP VPN с помощью Webmin

6.3.2 Клиент

Программа pptp обеспечивает поддержку функций клиента VPN на основе протокола PPTP (Point-to-Point Tunneling Protocol). Вы можете использовать эту программу для подключения удаленных пользователей к PPTP VPN.

По умолчанию pptp вызывает сервер PPTP и запускает демон pppd для поддержки обмена данными. Однако программа pptp может использоваться как менеджер соединений без применения pppd.

6.3.2.1 Опции

Первым аргументом командной строки **pptp** должно быть доменное имя или IP-адрес сервера PPTP. Остальные аргументы трактуются как опции **pptp** и все аргументы до первой нераспознанной опции передаются демону **pppd**, если в командной строке не указана опция **--nolaunchpppd**.

`--nolaunchpppd`
не запускать демон **pppd**, используя в качестве сетевого соединения устройство **stdin**. Применяйте этот в тех случаях, когда pptp работает как процесс для соединения pppd с использованием опции pty (см. пример подключения к серверу Microsoft).

`--phone number`
задает телефонный номер для включения в исходящий запрос соединения PPTP.

`--localbind address`
необязательный параметр для привязки к одному из локальных IP-адресов многодомного хоста.

`--quirks name`
использовать специальные способы обработки для отдельных серверов PPTP и модемов ADSL.

В настоящий момент для QUIRKS поддерживается одно значение

`BEZEQ_ISRAEL`
используемое для обеспечения совместимости с модемами Orckit ADSL, применяемыми в израильской сети BEZEQ.

Пример соединения с сервером Microsoft Windows VPN

```
pppd noauth nobsdcomp nodeflate mppe-40 mppe-128 mppe-stateless name domain\\\\username
remotename PPTP require-chapms-v2 pty "pptp 10.0.0.5 --nolaunchpppd"
```

Отметим, что используемый pppd файл chap-secrets должен включать запись для **domain\\username**

Статистика

Процесс pptp собирает статистические данные при передаче и приеме пакетов GRE. Эта информация предназначена для отлаживания работы PPTP и мониторинга качества соединений. Статистика собирается за все время с момента старта процесса pptp.

Для просмотра статистики можно послать сигнал **SIGUSR1** процессу **GRE-to-PPP Gateway**, что приведет к выводу дампа системных журналов (уровень LOG_NOTICE).

Собранная статистическая информация включает:

`rx accepted`
число пакетов GRE, успешно переданных PPP.

`rx lost`
число неполученных пакетов, которые предположительно были потеряны в сети.

`rx under win`
число пакетов, которые оказались дубликатами или содержали старые порядковые номера (это может быть вследствие изменения порядка доставки в сети по причине слишком малого значения тайм-аута

разупорядочивания).

rx over win
число пакетов, полученных со слишком большим нарушением порядка (это может быть связано с потерей более 300 пакетов в строке).

rx buffered
число пакетов, принятых с незначительным нарушением порядка, исправленным за счет буферизации.

rx OS errors
число случаев, когда ОС сообщала об ошибках при попытке чтения пакетов.

rx truncated
число случаев приема пакетов, размеры которых меньше длины, предполагаемой заголовком GRE.

rx invalid
число случаев приема пакетов с некорректными или неподдерживаемыми флагами в заголовке, некорректным номером версии или протоколом.

rx acks
число принятых подтверждений, которые не содержали данных (чистое подтверждение). Слишком большое число таких пакетов приводит к лишнему расходу полосы - проблема может быть решена настройкой параметров удаленного хоста.

tx sent
число переданных пакетов GRE с данными.

tx failed
число попыток передачи пакетов, когда ОС сообщала об ошибках.

tx short
число случаев, когда ОС не позволяла полностью записать пакет.

tx acks
число случаев передачи чистых подтверждений (без данных).

tx oversize
число случаев, когда не удавалось передать пакет в результате превышения размера PACKET_MAX.

round trip
оценка времени кругового обхода (в миллисекундах).

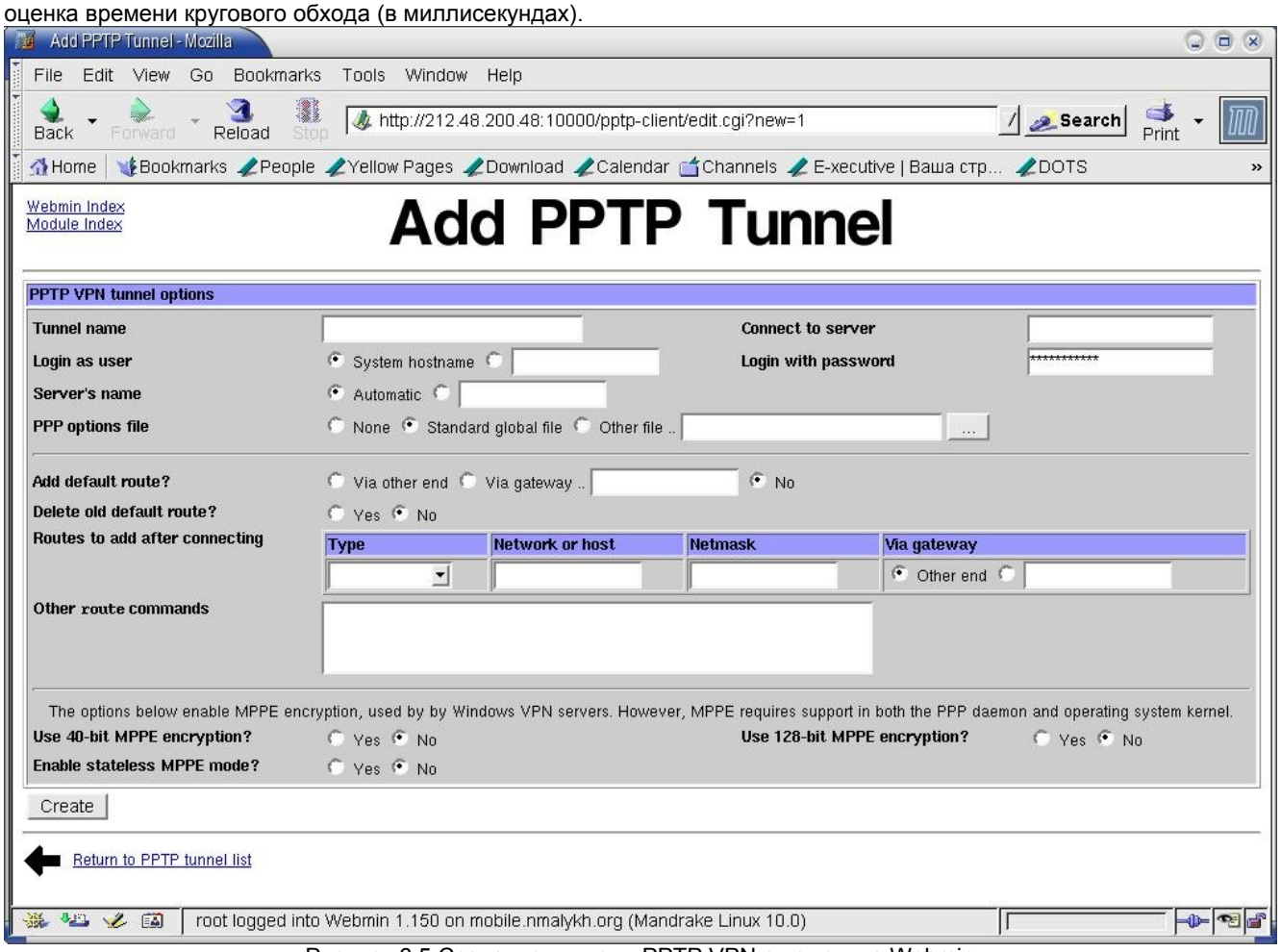


Рисунок 6.5 Создание туннеля PPTP VPN с помощью Webmin

7 Централизованные средства управления пользователями

7.1 RADIUS

RADIUS[1] представляет собой протокол и программы, обеспечивающие серверам доступа возможность обращения к централизованному серверу для аутентификации пользователей, подключенных по коммутируемой линии, и предоставления таким пользователям доступа к тем или иным ресурсам. RADIUS позволяет хранить учетные записи пользователей в централизованной базе данных, к которой могут обращаться любые службы с ограниченным доступом, требующие проверки полномочий пользователя. Централизованное хранение учетных записей также упрощает организацию контроля и учета работы пользователей. Разработанный компанией Livingston протокол RADIUS стал стандартом de facto в сфере телекоммуникаций. Спецификация протокола RADIUS, предложенная в качестве стандарта Intrenet, опубликована в RFC 2865.

7.1.1 FreeRADIUS

http://www.freeradius.org

Пакет FreeRADIUS представляет собой распространяемый по лицензии GPL бесплатный сервер RADIUS. В простейшем варианте FreeRADIUS очень напоминает сервер Livingston версии 2.0. Параметры конфигурации и общие механизмы работы покажутся знакомыми всем, кто использовал сервер Livingston. На начальных этапах пакет FreeRADIUS был просто вариантом сервера Cistron RADIUS (см. параграф 7.1.3 на стр. 174), но сейчас они различаются достаточно сильно.

Пакет FreeRADIUS более 40 словарей (файлы dictionary.*), описывающих серверы доступа различных компаний. Пакет поставляется с поддержкой баз данных LDAP (глава 7.2), MySQL, PostgreSQL и Oracle. Обеспечивается поддержка EAP подтипов EAP-MD5, EAP-TLS, EAP-TTLS, EAP-PEAP и Cisco LEAP.

Для управления сервером в дистрибутив включен Web-интерфейс, работающий на базе сервера Apache2 с поддержкой PHP.

7.1.1.1 Radiusd

Программа **radiusd** представляет собой демон RADIUS, обеспечивающий проверку пользователей, контроль их полномочий и учет работы (Authentication, Authorization and Accounting).

Синтаксис

```
radiusd [-A] [-S] [-a <каталог>] [-b] [-c] [-d <каталог>] [-f] [-i <адрес>] [-l
<каталог>] [-g <тип>]  [-p  <порт>] [-s] [-v] [-x] [-X] [-y] [-z]
```

Протокол RADIUS используется для обмена информацией между сервером доступа или коммутатором и сервером RADIUS с целью идентификации и проверки полномочий пользователя на основании имени (**login**) и пароля (password). Клиент передает имя пользователя и пароль серверу RADIUS, а последний проверяет полученную информацию и возвращает отклик с разрешением (**access OK**) или отказом (**access denied**). Если идентификация пользователя прошла успешно, клиенту также возвращается дополнительная информация (например, адрес IP).

Сервер RADIUS обеспечивает также учет работы пользователей, что позволяет на основании этой информации готовить отчеты, счета за использование услуг и т. п. Учетная информация хранится в журнальном файле **/var/log/radwtmp**, использующем формат **wtmp** (параграф 12.14 на стр. 398).

7.1.1.1.1 Опции

Таблица 12 Опции radiusd

Опция	Описание
`-A`	Эта опция задает запись файла **detail.auth** в дополнение к стандартному файлу **detail**[2] в тот же каталог. Дополнительный файл будет содержать записи для запросов на аутентификацию, которые могут быть полезны при отладке (в нормальном режиме работы этого не требуется).
`-S`	Задает исключение суффиксов и префиксов имен пользователей при записи в файл **detail**. Эту опцию не следует применять, задавая режим записи имен пользователей с помощью параметра **log_stripped_names** в конфигурационном файле **radiusd.conf** (параграф 7.1.1.1.2.1 на стр. 167).
`-a`	Эта опция задает каталог для записи учетных данных (по умолчанию **/var/log/radacct**). Если указанный параметром каталог существует, программа будет включать в файл **detail** текстовую запись для каждой регистрации или выхода (login/logout). По умолчанию такие записи помещаются в файл **<каталог для записи учетных данных>/terminal_server/detail**. Этой опции следует избегать, используя взамен параметр **radacctdir** в файле **radiusd.conf** (стр. 167).
`-l`	Эта опция задает каталог для записи журнальных файлов (по умолчанию (**/var/log**). Записи помещаются в файл с именем **radius.log**. Этот файл будет включать информационные записи и сообщения об ошибках, а также может содержать записи о все попытках регистрации в системе. Этой опции следует избегать, используя взамен параметр **log_dir** в файле **radiusd.conf** (стр. 167).

1 *Remote Authentication Dial-In User Service - протокол удаленной аутентификации пользователей с доступом по телефонной линии.*
2 *Информацию о модуле записи деталей можно получить по команде **man rlm_detail**.*

Опция	Описание
-g	Эта опция служит для задания типа (**facility**) записей в журнальный файл. По умолчанию используется тип **daemon**. Другим разумным выбором представляется тип **authpriv**.
-d	Опция служит для задания каталога, в котором хранятся файлы конфигурации (по умолчанию это **/etc/raddb**). Программа **radiusd** просматривает этот каталог на предмет поиска конфигурационных файлов, словарей (**dictionary.***) и файла **users**.
-i	Задает IP-адрес, с которым связывается сервер RADIUS. Этой опции следует избегать, используя взамен параметр **bind_address** в файле **radiusd.conf** (параграф 7.1.1.1.2.1 на стр. 167).
-b	Если сервер скомпилирован с поддержкой **dbm**, эта опция указывает необходимость использования файлов базы данных вместо текстового файла **users**[3].
-c	Эта экспериментальная опция включает поддержку кэширования паролей, групп и теневых файлов в hash-таблице, хранящейся в памяти. Такой подход повышает расход памяти, но ускоряет работу сервера. При использовании этой опции после каждого реального изменения файла паролей (добавления пользователя, смена пароля и т. п.), нужно отправить серверу RADIUS сигнал **SIGHUP** для повторной загрузки конфигурационных параметров и файлов **password/group/shadow**. Не следует использовать эту опцию в командной строке - лучше задать соответствующее значение параметра **cache** для модуля **unix** module в конфигурационном файле **radiusd.conf**.
-f	Отключает ветвление процесса, заставляя его работать в режиме **foreground**.
-p	Задает номер порта для сервера. Обычно **radiusd** работает с портами, указанными в файле **/etc/services** (**radius** и **radacct**). Эта опция задает номер порта для прослушивания запросов на аутентификацию, а для запросов учета (**accounting request**) будет использоваться порт со следующим номером. Не следует задавать номер порта в командной строке. Используйте для этого параметр **port** в конфигурационном файле **radiusd.conf** (параграф 7.1.1.1.2.1 на стр. 167).
-s	Обычно сервер создает отдельный процесс для учета и отдельные процессы для каждого запроса на аутентификацию. С помощью этой опции можно отключить создание дочерних процессов, переводя всю работу сервера в фоновый режим.
-x	Включает режим отладки, в котором сервер выводит детальную информацию для каждого запроса на устройство stderr[4]. Допускается двукратное использование опции (**-x -x** или **-xx**) для вывода дополнительной информации. Наибольшую пользу опция отладки оказывает при совместном использовании с флагом **-s**.
-X	Включает расширенный режим отладки. Эквивалентна комбинации флагов **-sfxx**.
-y	Задает вывод деталей каждого запроса на аутентификацию в файл **radius.log**. Это опцию лучше не использовать, задавая взамен параметр **log_auth** в конфигурационном файле **radiusd.conf**.
-z	Задает включение пароля в запись файл **radius.log** даже при успешной регистрации. Это опцию лучше не использовать, задавая взамен параметр **log_auth_goodpass** в конфигурационном файле **radiusd.conf**. ***Отметим, что запись паролей при успешной регистрации существенно снижает уровень безопасности системы.***

7.1.1.1.2 Конфигурационные файлы

Программа **radiusd** использует множество конфигурационных файлов, кратко описанных ниже. Эти файлы обычно хранятся в каталоге базы данных FreeRADIUS (по умолчанию **/etc/raddb**). Благодаря оператора включения **$INCLUDE** можно также использовать конфигурационные файлы, хранящиеся в других каталогах.

Строки конфигурационных файлов, начинающиеся с символа **#**, содержат комментарии и не принимаются во внимание программой.

7.1.1.1.2.1 Файл radiusd.conf

Этот файл содержит основные конфигурационные параметры сервера FreeRADIUS. Файл **radiusd.conf** хранится в каталоге базы данных RADIUS (по умолчанию **/etc/raddb**).

7.1.1.1.2.1.1 Формат файла

Файл содержит разделы, конфигурационные пары “**параметр = значение**” и комментарии. Каждый раздел конфигурационного файла начинается с имени раздела, после которого следует строка, повторяющая это имя и содержащая открывающую фигурную скобку **{**. Раздел может содержать пары **параметр = значение** и вложенные разделы. В конце каждого раздела должна помещаться закрывающая скобка **}** в пустой строке.

Любая строка, начинающаяся с символа **#**, является комментарием и не принимается программой во внимание. Пустые строки или строки пробелов также игнорируются программой.

Каждое имя раздела, пара **параметр = значение** или комментарий должны начинаться с новой строки.

В качестве значений параметров могут использоваться определенные выше в том же файле именованные объекты.

3 *В современных версиях программы эта опция не делает ничего и нет смысла ее использовать.*
4 *По умолчанию таким устройством является пользовательский терминал.*

В таких случаях имени объекта, используемого в качестве значения параметра должен предшествовать символ $, а само имя должно помещаться в фигурные скобки **{}**.

Описания параметров конфигурации приведены в файле **radiusd.conf**, и снабжены подробными комментариями, поэтому мы не будем здесь останавливаться на их рассмотрении.

7.1.1.1.2.2 Файл dictionary

Этот файл содержит определения атрибутов RADIUS, используемые в других конфигурационных файлах. В файле также указаны другие файлы словарей, которые обычно хранятся в том же каталоге. Не следует менять этот файл без четкого понимания необходимости выполняемых действий.

Строки, начинающиеся с символа **#**, являются комментариями и не принимаются во внимание программой.

Каждая из строк файла словаря, не являющаяся комментарием, должна начинаться с ключевого слова **$INCLUDE**, **ATTRIBUTE**, **VALUE** или **VENDOR**. Строки включения файлов имеют форму

```
$INCLUDE <имя файла>
```

Включаемый файл обычно находится в том же каталоге и может быть указать только именем, в противном случае требуется указывать полный путь поиска файла.

Строки атрибутов имеют вид

```
ATTRIBUTE <имя> <значение> <тип>
```

и определяют числовые эквиваленты атрибутов RADIUS. Поле имени может включать произвольный текст без пробелов. Значение должно быть указано цифрами в десятичном формате в соответствии с нумерацией имен, приведенной в RFC 2865. Идентификатором типа могут служить строки **string**, **ipaddr**, **integer** или **date**.

Строки задания числовых значений имеют форму

```
VALUE <атрибута> <имя> <значение>
```

и определяют числовые эквиваленты имен. Поле атрибута должно быть определено ранее в одной из записей **ATTRIBUTE** данного файла или включенного в него файла. Поле имени может включать любой текст без пробелов, а значение должно быть выражено десятичным числом в соответствии с RFC 2865.

Строки

```
VENDOR <имя> <значение>
```

задают числовые эквиваленты производителей сетевого оборудования.

Допускается также создавать разделы параметров, связанных с определенным производителем и содержащих описанные выше строки определения атрибутов и значений. Такие разделы должны иметь вид:

```
BEGIN-VENDOR    <имя производителя>
...
END-VENDOR      <имя производителя>
```

7.1.1.1.2.3 Файл clients

Файл **clients** содержит информацию о клиентах и используемых имя публичных ключах. В настоящее время не рекомендуется использовать этот файл и он сохраняется только для совместимости со старыми версиями. Для хранения информации о пользователях в новых версиях служит файл **clients.conf**.

Каждая строка файла содержит пару значений “**клиент - ключ**”, разделенных между собой пробелами или символами табуляции.

Клиент указывается в начале строки именем хоста или IP-адресом. Публичный ключ клиента используется при обмене информацией между клиентом и сервером RADIUS для шифрования паролей в пакетах RADIUS и аутентификации. В конфигурационном файле клиента (например, терминального сервера), должно быть указано идентичное значение ключа.

Файл **clients** читается только при загрузке демона **radiusd**.

7.1.1.1.2.4 Файл clients.conf

Файл **clients.conf** содержит информацию о клиентах RADIUS и серверах доступа, которая в старых версиях программы хранилась в файлах **clients** и **naslist**.

Файл содержит одну запись для каждого клиента в форме

```
client 10.10.10.10 {
        # secret and password are mapped through the "secrets" file.
        secret      = testing123
        shortname   = liv1
        # the following three fields are optional, but may be used by
        # checkrad.pl for simultaneous usage checks
        nastype     = livingston
        login       = !root
        password    = someadminpas
}
```

После ключевого слова **client** указывается имя хоста или IP-адрес. Допускается указывать в качестве клиентов подсети, заданные номером сети и маской, которая выражена числом битов номера сети (например, 192.168.1.0/16). Поле secret содержит публичный ключ, используемый для шифрования данных при передаче

между сервером и клиентом, а также для аутентификации. Публичный ключ представляет собой строку с числом символов до 32 и не должен включать пробелов. Указанный в конфигурационном файле клиента публичный ключ должен совпадать с этим значением. Поле **shortname** используется в качестве синонима полного имени или IP-адреса хоста. Поля **nastype**, **login** и **password** являются необязательными, но из может использовать сценарий **checkrad.pl**. Параметр **nastype** определяет тип сервера доступа и может принимать одно из перечисленных в файле значений. Поля **login** и **password** пока не используются программой

7.1.1.1.2.5 Файл naslist

Этот файл содержит записи для всех серверов доступа (NAS[1]) в сети. Список серверов доступа не обязательно совпадает со списком клиентов, особенно при использовании в вашей сети прокси-серверов (в таких случаях сервер proxy является клиентом RADIUS и шлет серверу запросы для различных NAS).

Файл также содержит сокращенные имена для каждого терминального сервера, используемые при создании каталогов, в которые записываются файлы **detail**, а также включаемые в регистрационные записи файла **/var/log/radwtmp**. Кроме того, в этом файле указаны типы используемых NAS (Cisco, Livingston, Portslave и т. п.).

Каждая строка файла, не являющаяся комментарием, содержит три поля, разделенных пробелами или символами табуляции. Первое поле содержит имя или IP-адрес сервера доступа, второе (необязательное) поле указывает сокращенное имя NAS, а третье - тип сервера доступа (идентификатор типа передается программе **checklogin** при обнаружении попыток дважды зарегистрироваться в сети с одним именем).

Файл **naslist** считывается однократно при загрузке демона **radiusd**.

7.1.1.1.2.6 Файл hints

Этот файл содержит набор рекомендаций для сервера RADIUS, основанных на передаваемых серверами доступа именах пользователей и других атрибутах. Кроме того, файл содержит ряд отображений для имен пользователей (например, **Pusername** -> **username**). Преобразования, выполняемые с использованием этого файла, напоминают работу с префиксами и суффиксами в сервере Livingston 2.0, но обеспечивают более широкие возможности.

7.1.1.1.2.7 Файл huntgroups

Этот файл определяет группы (huntgroup) для ограничения доступа к ним некоторых пользователей или групп.

7.1.1.1.2.8 Файл users

Файл **users** содержит группы конфигурационных параметров, управляющие аутентификацией и проверкой полномочий для каждого пользовательского запроса.

Каждая запись файла начинается с имени пользователя, за которым следует список (возможно пустой) проверяемых элементов. Следующая строка начинается с символа табуляции и содержит список (возможно пустой) элементов отклика. Каждый элемент в обоих типах списков представляется парой

имя – значение

Строка может содержать множество элементов, разделенных между собой запятыми. Элементы откликов допускается указывать в нескольких строках, причем все строки, кроме последней, должны заканчиваться запятой.

Список проверяемых элементов используется для входящих запросов. Если имя пользователя соответствует записи файла и все проверяемые элементы этой записи также соответствуют входящему запросу, список элементов отклика из этой записи добавляется к списку атрибутов, который будет использоваться в отклике на данный запрос. Описанный процесс повторяется для всех записей файла **users**. Если для входящего запроса не обнаружено соответствующей ему записи в файле **users**, такой запрос отвергается.

7.1.1.1.2.8.1 Операторы

В парах **имя - значение** допускается использование различных операторов, описанных ниже.

Имя = значение

Этот оператор не допускается в выражениях для проверки атрибутов протокола RADIUS, но его можно использовать для проверки атрибутов конфигурации (**Auth-Type** и т.п.), а также для установки значения атрибутов (если не существует другого элемента с тем же атрибутом). Для атрибутов откликов это трактуется как: “добавить элемент в список отклика, если в нем нет элемента с тем же атрибутом”.

Имя := значение

Оператор присваивает значение указанному атрибуту, добавляя атрибут в список, если его там нет. В элементах проверки результат всегда будет положительным.

Имя == значение

В элементах проверки проверяется наличие атрибута, заданного именем, и совпадение его значения с указанным. Не допускается использование этого оператора в элементах отклика.

Имя += значение

Оператор добавляет атрибут и его значение в список элементов. В элементах проверки результат всегда будет положительным.

Имя != значение

При проверке положительный результат будет возвращен в случаях наличия данного атрибута в списке и несовпадения значения атрибута с указанным значением. Не допускается использование этого оператора в

1 *Network Access Server - сервер доступа в сеть.*

элементах отклика.

Имя > значение
Проверяется наличие в запросе атрибута со значением больше указанного. Не допускается использование этого оператора в элементах отклика.

Имя >= значение
Проверяется наличие в запросе атрибута со значением не меньше указанного. Не допускается использование этого оператора в элементах отклика.

Имя < значение
Проверяется наличие в запросе атрибута со значением меньше указанного. Не допускается использование этого оператора в элементах отклика.

Имя <= значение
Проверяется наличие в запросе атрибута со значением не больше указанного. Не допускается использование этого оператора в элементах отклика.

Имя =~ выражение
Проверяется соответствие значения указанного атрибута заданному регулярному выражению. Оператор можно применять только для атрибутов типа **string**. Не допускается использование этого оператора в элементах отклика.

Имя !~ выражение
Проверяется несоответствие значения указанного атрибута заданному регулярному выражению. Оператор можно применять только для атрибутов типа **string**. Не допускается использование этого оператора в элементах отклика.

Имя =* значение
Проверяется факт наличия в запросе указанного атрибута, независимо от его значения. Не допускается использование этого оператора в элементах отклика.

Имя !* значение
Проверяется факт отсутствия в запросе указанного атрибута, независимо от его значения. Не допускается использование этого оператора в элементах отклика.

7.1.1.1.2.8.2 Предостережения

Зарезервированное имя **DEFAULT** соответствует любому имени пользователя.

Записи файла обрабатываются в порядке их следования в файле **users**. Если запись содержит специальный элемент **Fall-Through = No** (или 0) в качестве атрибута отклика, просмотр файла завершается на этой записи. Все записи, не содержащие атрибута **Fall-Through**, трактуются, как записи с атрибутом **Fall-Through = No**. При наличии в записи атрибута **Fall-Through = Yes** (или 1) обработка файла продолжается. При работе с атрибутами **Fall-Through** следует соблюдать осторожность и проверять записи с различными вариантами пользовательских запросов.

Для указания типа аутентификации, применяемого для пользователя, служит специальный атрибут **Auth-Type**. Список поддерживаемых типов аутентификации можно найти в файле **dictionary**. После завершения обработки файла **users** проводится аутентификация запроса с использованием метода, заданного атрибутом **Auth-Type**.

7.1.1.1.2.8.2.1 Примеры

```
bob  Auth-Type := Local, User-Password == "bob"
```
Запросы, содержащие атрибут **User-Name** со значением **bob**, будут аутентифицироваться с использованием локального пароля **bob**. Запись не содержит элементов отклика, поэтому отклик будет пустым.

```
DEFAULT   Auth-Type := System
     Fall-Through = Yes
```
Для всех пользователей устанавливается аутентификация типа **system** и продолжается обработка последующих записей файла.

```
DEFAULT Service-Type==Framed-User, Framed-Protocol==PPP
     Service-Type = Framed-User,
     Framed-Protocol = PPP,
     Fall-Through = Yes
```
Если запрос содержит атрибуты **Service-Type** и **Framed-Protocol** с указанными значениями, эти атрибуты будут включаться в отклик (пользователь узнает, что он запрашивал)..

7.1.1.1.2.8.3 Рекомендации

Для проверки корректности записей файла **users** запустите сервер в режиме отладки (опция **-X**) и используйте программу **radclient** для отправки тестовых запросов. Сервер будет выводить на экран записи, соответствующие запросу, что позволит вам проверить корректность работы. Эту проверку следует выполнять в первую очередь при возникновении или ожидании проблем с файлом **users**.

Будьте предельно внимательны при создании записей в файле **users**. Не забывайте о важности порядка следования записей. Не используйте атрибут **Fall-Through = 1** без необходимости.

Записи отвергающие некоторые типы запросов должны находиться в начале файла и не должны содержать атрибут **Fall-Through** в списках откликов. Вслед за такими записями должны включаться записи для отдельных пользователей, не содержащие атрибута **Fall-Through**. Все записи **DEFAULT** должны размещаться в конце файла, за исключением записей с атрибутом **Fall-Through**, содержащих атрибуты откликов.

7.1.1.1.2.9 Файл acct_users

Файл **acct_users** содержит параметры учета работы пользователей и включает последовательности директив конфигурации, используемые модулем **files** для записи в журнальные файлы информации о работе пользователей.

Формате этого файла идентичен формату файла **users**, но содержит отклики на запросы учета (accounting), а не аутентификации.

7.1.1.2 radclient

Программа radclient передает запросы серверу RADIUS и выводит полученные от сервера отклики.

Синтаксис

```
radclient [-d <каталог>] [-f <файл>] [-i <адрес>] [-x] <сервер> {acct|auth} <ключ>
```

Программа **radclient** является обычным radius-клиентом и может передавать серверу RADIUS любые запросы и выводить полученные отклики. Эта программа может быть полезна при тестировании сервера на этапах настройки или изменения его конфигурации.

Программа **radclient** принимает пары **атрибут-значение** со стандартного устройства ввода или из файла, указанного в командной строке. Полученные пары кодируются с использованием словаря (файл **dictionary**) и передаются серверу.

Программа автоматически шифрует атрибуты **User-Password**.

7.1.1.2.1 Опции

Таблица 13. Опции radclient.

Опция	Описание
-d	Опция служит для задания каталога, в котором хранятся файлы конфигурации (по умолчанию это **/etc/raddb**).
-f	Задает имя файла, используемого для чтения пар **атрибут-значение**, взамен их приема с клавиатуры.
-i	Задает IP-адрес, с которым связываются исходящие запросы RADIUS.
-x	Включает режим отладки, в котором выводится на экран дополнительная информация.
-q	Задает работу без вывода на экран какой-либо информации.
<сервер>[:<порт>]	Задает IP-адрес сервера RADIUS и может также включать номер порта UDP, используемого этим сервером. Если порт не указан, берется номер порта из файла **/etc/services**. Для запросов учета используется номер порта **radacct**, для всех прочих запросов - порта **radius**. Если эти порты не указаны в файле **/etc/services**, программа будет использовать порты 1813 и 1812, соответственно.
acct | auth	Задает режим передачи запросов на аутентификацию (**auth**) или учет работы (acct). Допускается также задание десятичных значений кодов (например, 12 для Status-Server).
<ключ>	Задает публичный ключ, используемый при обмене данными между клиентом и сервером. В конфигурационном файле сервера для IP-адреса клиентского хоста должен быть указан идентичный ключ.

Пример использования

Чтобы убедиться в поддержке удаленным сервером атрибута **Status-Server**, введите команду

```
echo "User-Name = fnord" | radclient 192.168.1.42 12 s3cr3t
```

В результате на экран будет выведен отклик типа показанного ниже.

```
Sending request to server 192.168.1.42, port 1812.
radrecv: Packet from host 192.168.1.42 code=2, id=140, length=54
Reply-Message = "Cistron Radius up 21 days, 02:05"
```

7.1.1.3 Radlast

Программа **radlast** выводит сведения о последней регистрации каждого пользователя из системного журнала **radwtmp**.

Синтаксис

```
radlast [options]
```

Сервер FreeRADIUS может записывать учетные сведения в файл формата **wtmp** (Приложение 12.14). Программа **radlast** на самом деле просто вызывает системную утилиту с опцией **-f**, указывающей на файл **radwtmp**.

Опции программы **radlast** полностью совпадают с опциями системной утилиты **last**, описанной в параграфе 2.8.1.2 (стр. 47).

7.1.1.4 Radtest

Программа **radtest** передает запросы серверу RADIUS и выводит на экран отклики сервера.

Синтаксис

```
radtest [-d <каталог>] <пользователь> <пароль> <сервер> <порт NAS> <ключ> [ppphint]
[nasname]
```

По сути **radtest** представляет собой просто shell-сценарий, использующий программу **radclient** (стр. 171). Сценарий создает список пар **атрибут-значение** и передает их программе **radclient**.

7.1.1.4.1 Опции

Таблица 14. Опции radtest.

Опция	Описание
-d	Опция служит для задания каталога, в котором хранятся файлы конфигурации (по умолчанию это **/etc/raddb**).
<пользователь>	Задает имя пользователя для запроса.
<пароль>	Задает пользовательский пароль.
<сервер>	Задает IP-адрес сервера RADIUS и может включать номер порта UDP, используемого сервером. Если порт не указан, берется номер порта из файла **/etc/services.** Для запросов учета используется номер порта **radacct**, для всех прочих запросов - **radius**. Если эти порты не указаны в **/etc/services**, программа будет использовать порты 1813 и 1812, соответственно.
порт NAS	Задает значение атрибута **NAS-Port**, которое должно быть целым числом в диапазоне от 0 до 2^{31}.
<ключ>	Задает открытый ключ, используемый при обмене данными между клиентом и сервером. В конфигурационном файле сервера для IP-адреса клиентского хоста должен быть указан идентичный ключ.
ppphint	Неотрицательное целое число будет добавлять атрибут **Framed-Protocol = PPP** в передаваемый серверу запрос.
nasname	При наличии этого параметра, указанное имя сервера доступа преобразуется в IP-адрес, а в пакет запроса добавляется атрибут **NAS-IP-Address** с адресом сервера в качестве значения.

7.1.1.5 Radwho

Программа **radwho** позволяет просматривать список активных (online) пользователей.

Синтаксис

```
radwho [-d <каталог>] [-l] [-h] [-f] [-n] [-s] [-i] [-p] [-c] [-r]
```

Сервер FreeRADIUS можно настроить так, чтобы он поддерживал базу данных об активных соединениях в файле **radutmp**[1]. Утилита **radwho** просматривает эти данные и выводит список активных пользователей.

7.1.1.5.1 Опции

Таблица 15. Опции radwho.

Опция	Описание
-d	Опция служит для задания каталога, в котором хранятся файлы конфигурации (по умолчанию это **/etc/raddb**).
-l	Задает вывод информации также о локальных пользователях. В этом случае программа **radwho** читает информацию о локальных пользователях из системного журнала **utmp** (параграф 12.14 на стр. 398).
-h	Задает вывод информации только о пользователях сеансов SLIP и PPP.
-f	Задает работу в построчном режиме (как демон **fingerd**) - программа дожидается получения полной строки, затем выводит полученную строку, завершая ее символами **\r\n**.
-n	Эта опция отключает поиск полных имен пользователей в файле **password**.
-s	Задает вывод полных имен.
-i	Задает вывод идентификаторов сессии вместо имени.
-p	Задает вывод дополнительной колонки, указывающей тип порта (I для ISDN, A для аналоговых линий).
-c	Задает вывод идентификатора (Caller ID) вместо имени, если номер известен.
-r	Задает вывод данных в необработанном виде с разделением полей запятыми.

7.1.1.6 Radzap

Программа **radzap** удаляет некорректные записи из базы данных об активных соединениях.

1 *Возможно также организовать хранение этой информации в базе данных SQL.*

Синтаксис

```
radzap [-d <каталог>] [-r <сервер>] [-p <порт учета>] [-v] nas [<порт>] [<имя>]
```

Поддерживаемая программой FreeRadius в файле **radutmp** база данных об активных соединениях может потерять синхронизацию, в результате чего программа **radwho** (стр. 172) будет выводить некорректную информацию. Программа **radzap** удаляет некорректные записи и синхронизирует базу данных.

7.1.1.6.1 Опции

Таблица 16. Опции radzap.

Опция	Описание
`-d`	Опция служит для задания каталога, в котором хранятся файлы конфигурации (по умолчанию это **/etc/raddb**).
`-r`	Задает имя хоста или IP-адрес сервера RADIUS.
`-p`	Задает номер порта, в который передаются пакеты учета (accounting). Обычно для этих целей используют порт с номером 1813 или 1646, который в файле /**etc/services** указывается именем **radacct**.
`-v`	Задает вывод подробной информации о выполняемых действиях.
`nas`	Задает имя хоста или IP-адрес сервера доступа, чью сессию вы хотите удалить.
`<порт>`	Указывает номер порта для удаляемой сессии. Этот параметр совпадает со значением атрибута **NAS-Port**.
`<имя>`	Этот необязательный параметр позволяет указать имя пользователя, чью сессию следует удалить.

7.1.2 GNU Radius

http://www.gnu.org/software/radius/radius.html

Сервер **GNU Radius** обеспечивает поддержку функций аутентификации и учета работы пользователей. Сервер отличается достаточно высоким уровнем масштабирования. GNU Radius совместим со всеми существующими терминальными серверами и может взаимодействовать даже с серверами, использующими нестандартные реализации протокола RADIUS. Встроенный язык расширения позволяет администратору создавать свои правила для разбора и реструктуризации нестандартных запросов и откликов от терминальных серверов.

GNU Radius непосредственно поддерживает взаимодействие с серверами MySQL и PostgreSQL. Работа с другими серверами баз данных возможна через стандартный интерфейс ODBC. Сервер не предъявляет каких-либо специальных требований к формату таблиц базы данных. Запросы для работы с сервером формируются администратором, который, следовательно, может выбрать удобный для себя формат таблиц.

Сервер **GNU Radius** поддерживает протокол SNMP с деревом MIB в соответствии с RFC 2619 и RFC 2621, а также фирменными расширениями.

Пакет включает сервер аутентификации и учета, а также набор средств администрирования.

7.1.2.1 Схемы аутентификации

Сервер поддерживает несколько различных схем аутентификации. Пользователь передает сведения о себе[1] серверу RADIUS напрямую в ответах на запросы терминального сервера **login/password** или и использованием протоколов PAP/CHAP. Сервер сверяет полученные данные с учетной информацией пользователей. Варианты хранения учетной информации, поддерживаемые сервером перечислены в таблице 17.

Таблица 17 Способы хранения учетных данных пользователей в GNU Radius

Способ хранения	Описание
Системная база данных	Имена и пароли пользователей хранятся на сервере в файлах **/etc/passwd**, как это обычно делается на хостах UNIX.
Внутренняя база данных сервера	Регистрационные имена пользователей, пароли и другая информация сохраняются во внутренней базе данных сервера RADIUS. Пароли перед записью в базу данных шифруются с использованием алгоритма MD5 или DES. Возможно также хранение паролей в нешифрованном виде, если используется протокол CHAP. Отметим, что хранение нешифрованных паролей существенно снижает уровень безопасности
База данных SQL	Информация о пользователях храниться в базе данных SQL. Структура базы данных определяется администратором системы.
PAM-аутентификация	Аутентификация пользователей осуществляется с использованием методов PAM[2].

7.1.2.2 Схемы учета работы пользователей

Сервер поддерживает три варианта учета работы пользователей, перечисленные в таблице .

1 *Регистрационное имя и пароль*
2 *Pluggable Authentication Service - подключаемая служба аутентификации. Информацию о методах PAM вы сможете найти на сайте http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam.html.*

Таблица 18 Способы хранения учетных данных GNU Radius

Способ хранения	Описание
Системные средства UNIX	Учетные данные хранятся в файлах radutmp/radwtmp и могут просматриваться с помощью команд **radwho** и **radlast**, аналогичных штатным командам UNIX **who** (см. стр. 192) и **last** (см. стр. 47). Формат файлов описан а Приложении 12.14.
Детализированный учет	Учетные данные хранятся в текстовых файлах, которые можно анализировать с помощью внешних программ.
База данных SQL	Учетные данные хранятся в базе данных SQL и могут обрабатываться с помощью стандартных запросов.

С помощью встроенного языка расширения администратор может реализовать свои методы учета.

7.1.2.3 Возможности расширения

Сервер GNU Radius поддерживает два языка расширения - **Rewrite** и **Scheme**. **Rewrite** использует синтаксис, подобный синтаксису языка **C** и предназначен в основном для модификации[1] содержимого входящих запросов.

Использование языка **Scheme** требует установки Guile версии 1.4 или выше. Этот язык позволяет администратору создавать свои методы аутентификации и учета.

Языки расширения полностью совместимы один с другим - программы **Scheme** могут вызывать функции **Rewrite** и наоборот.

7.1.3 Cistron RADIUS

http://www.radius.cistron.nl

Сервер Cistron RADIUS является прообразом FreeRAIUS (см. параграф 7.1.1 на стр. 166), но в последнее время существенно отстает в развитии от своего потомка. Возможности Cistron RADIUS существенно скромнее по сравнению FreeRAIUS (в частности нет интеграции с базами данных и LDAP), а основные параметры настройки совпадают с параметрами FreeRAIUS.

7.2 LDAP

7.2.1 OpenLDAP

http://www.openldap.org

1 *Rewriting - переписывание.*

8 Инструменты для создания и поддержки политики безопасности

В системах Linux вы можете создавать правила политики безопасности для большинства случаев с помощью обычного текстового редактора. Однако, существует ряд программ с развитым графическим интерфейсом для создания и компиляции политики безопасности. Рассмотрим вкратце некоторые из таких программ.

8.1 Firewall Builder

http://www.fwbuilder.org/

Firewall Builder представляет собой инструмент для настройки и поддержки межсетевых экранов, работающих на базе различных платформ. Сама программа также способна работать на разных платформах, что позволяет администратору сохранить привычную среду при работе с брандмауэром, независимо от используемой для него аппаратно-программной платформы. Пакет **Firewall Builder** включает графический пользовательский интерфейс и комплект компиляторов для различных систем межсетевого экранирования. Работа Firewall Builder основана на объектно-ориентированных моделях, что помогает администратору поддерживать базу данных для сетевых объектов и обеспечивает возможность редактирования политики безопасности с использованием технологии drag-and-drop. В настоящее время Firewall Builder поддерживает **iptables**, **ipfilter**, **OpenBSD PF** и **Cisco PIX**. Технические детали поддерживаемых компиляторов политики для всех платформ вы сможете найти на сайте (раздел **Modules** на титульной странице).

Firewall Builder может создавать конфигурационные файлы для всех поддерживаемых платформ на основе единой политики с использованием удобного в работе дружественного графического интерфейса. Единообразие политики для различных платформ предоставляет администратору возможность выбора наиболее подходящего решения для различных узлов неоднородной сети и обеспечивает простоту перехода с одной платформы на другую.

Рисунок 8.1. Интерфейс программы Firewall Builder.

Firewall Builder позволяет управлять множеством брандмауэров с использованием единой базы данных о сетевых объектах. Внесенные в свойства объектов изменения незамедлительно учитываются в политике всех брандмауэров, использующих данный объект. Администратору остается лишь заново скомпилировать политику и установить ее на соответствующие шлюзы.

При использовании Firewall Builder администратор имеет дело с абстракциями политики безопасности и правил трансляции адресов (NAT). Программа “прячет” специфику платформы, на которой будет использоваться политика, позволяя администратору полностью сосредоточиться на вопросах реализации политики безопасности. Компиляторы политики получают информацию из базы данных о сетевых объектах и службах и способны генерировать конфигурационные файлы для межсетевых экранов различных типов, избавляя администратора от необходимости запоминать множество мелких деталей и ограничений. Компиляторы политики могут также выполнять проверку правил межсетевого экранирования и предупреждать администратора о наличии ошибок или противоречий еще до реализации политики.

Дополнительную информацию о возможностях программы и рекомендации по работе с ней вы сможете найти в руководстве пользователя[1] и серии статей **Firewall Builder Cookbook**, опубликованных на сайте проекта.

Все основные модули Firewall Builder, включая интерфейсную библиотеку (API) **libfwbuilder**, графический интерфейс пользователя и компиляторы политики для iptables, ipfilter, pf и ipfw распространяются на условиях GNU General Public License. Компилятор политики для PIX разработан компанией NetCitadel LLC и распространяется по лицензии этой компании.

8.2 Конвертер политики Checkpoint Firewall-1 в формат FirewallBuilder

http://sourceforge.net/projects/cp2fwbuilder/

Этот конвертер представляет собой сценарий perl[2], позволяющий преобразовать существующие правила Checkpoint Firewall-1 в набор правил для брандмауэров Linux или *BSD. При конвертации базы правил (Rulebase) и

1 *Копия руководства пользователя имеется в каталоге Documents/ приложенного к книге компакт-диска.*
2 *Исходный текст конвертера вы можете найти в каталоге SRC/ приложенного к книге компакт-диска.*

объектов (Objects) FW-1 преобразуются в XML-формат программы FirewallBuilder.

8.3 Модуль Firewall программы Webmin

Система удаленного управления Webmin (см. параграф 11.3) среди прочих включает и несколько модулей для создания и поддержки правил фильтрации пакетов на основе iptables (см. рисунок 8.2). Модуль может работать с цепочками таблиц filter (параграф 5.1.6.2), mangle (параграф 5.1.6.4) и nat (параграф 5.1.6.3). Для каждой из встроенных цепочек вы можете задать политику (см. параграф 5.1.9.2.1.7). Поддерживается также возможность создания пользовательских цепочек.

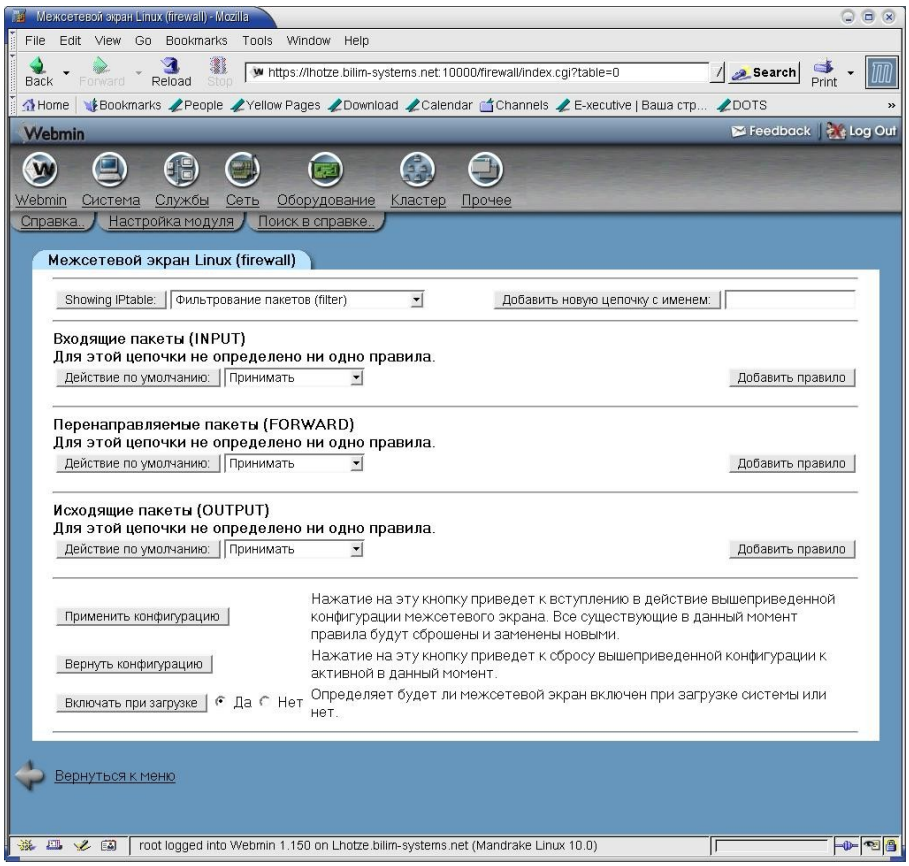


Рисунок 8.2. Настройка межсетевого экрана с помощью модуля Webmin.

Web-интерфейс позволяет создавать и редактировать правила фильтрации как для локальной машины, так и для удаленных хостов (см. рисунок 8.3). При создании новых правил вы можете изначально указать позицию в списке существующих правил, куда будет помещаться создаваемое правило. Кроме того, интерфейс обеспечивает возможность перемещения отдельного правила вверх и вниз по списку.

Модуль позволяет активизировать созданные цепочки и таблицы и задать режим загрузки правил iptables при старте системы.

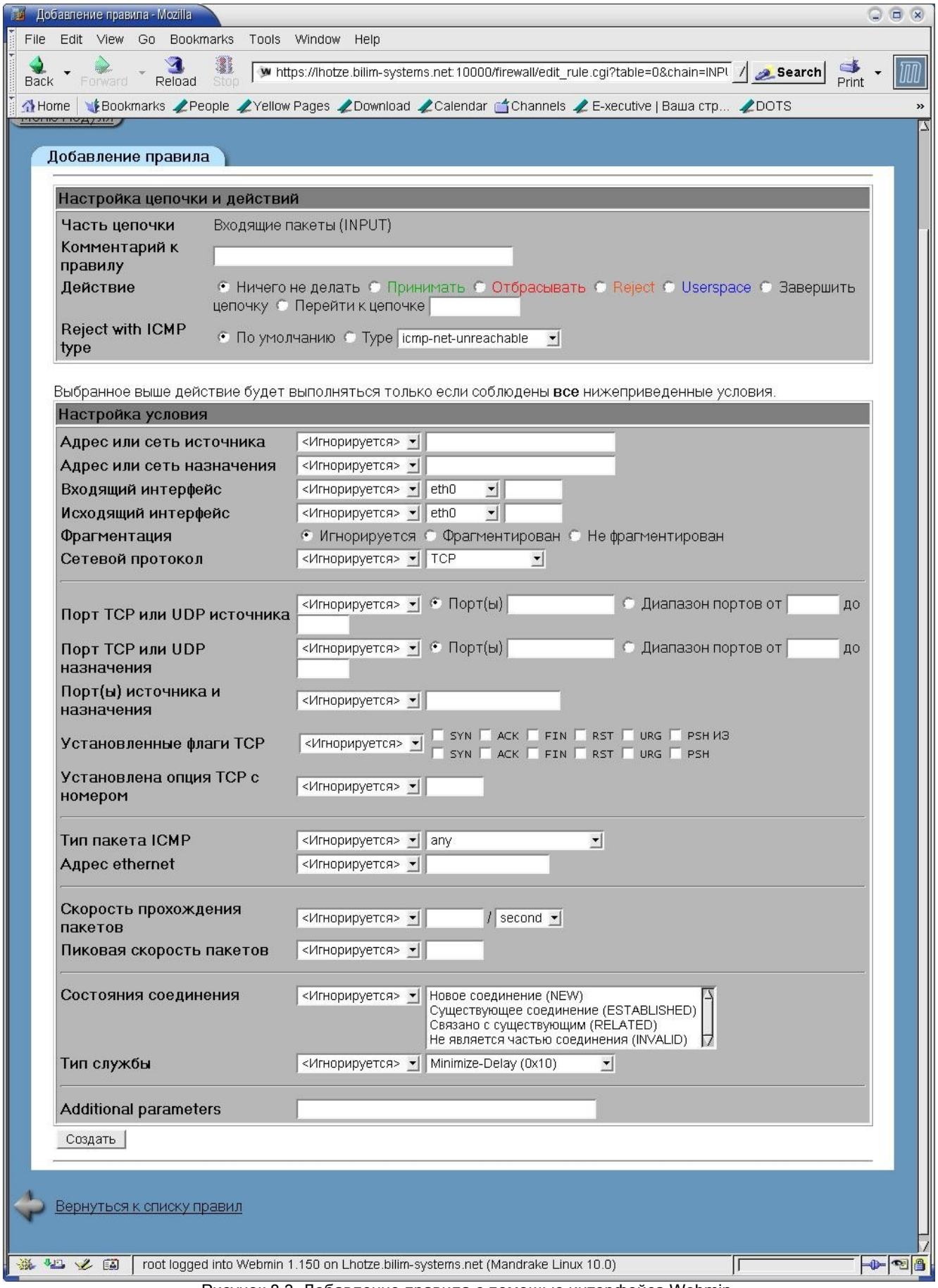


Рисунок 8.3. Добавление правила с помощью интерфейса Webmin.

9 Персональные брандмауэры для платформы Windows

Дополнительную информацию вы сможете найти на сайте http://osswin.sourceforge.net/#firewall.

9.1 NetDefender

http://sudhirmangla.i6networks.com/personal/Firewall.htm

Межсетевой экран NetDefender бесплатно распространяется в исходных текстах и может работать на различных Windows-платформах, начиная с Windows 2000. Программа использует простой и дружественный интерфейс, позволяющий выполнить все требуемые настройки даже неопытному администратору
Программный код NetDefender полностью написан на Visual C++ с использованием MFC, Windows API, драйвера Filter Hook (из комплекта поставки Windows 2000). Программу можно загрузить с сайта как в виде исходных текстов, так и в форме исполняемых файлов.

Основные возможности:

- глобальное управление трафиком (разрешить или блокировать) с помощью мыши;
- эффективная фильтрация пакетов;
- возможность создания пользовательских правил фильтрации;
- фильтрация по IP-адресам и номерам портов отправителя и получателя, а также протоколам (IP,TCP,ICMP);
- сканер для обнаружения в системе открытых портов;
- справочная система с подробной информацией о работе с программой.

9.2 Privaria

http://www.privaria.org

Пакет PRIVARIA Secure Networking Suite представляет собой набор клиентских программ с открытым кодом для организации шифрованных соединений peer-to-peer. Пользователи Privaria могут организовать между собой защищенные соединения с шифрованием на базе 128-битовых ключей. Пакет Privaria не конфликтует с имеющимися в сетях брандмауэрами, для организации защищенных соединений требуется лишь доступ по протоколу FTP.
Программа поддерживает функции удаленного доступа и обмена файлами, а для пользователей Windows 2000 и XP поддерживаются также шифрованные каналы видео-конференций.

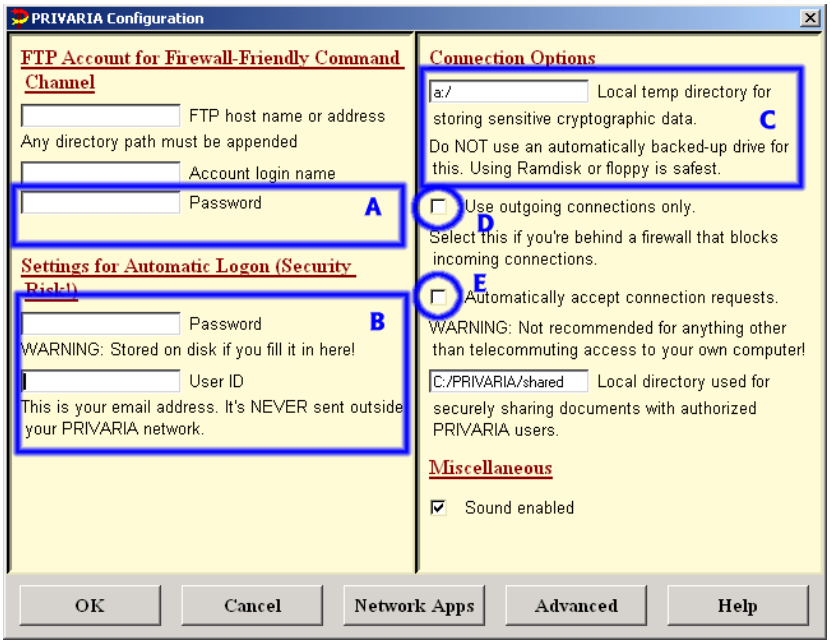


Рисунок 9.1. Интерфейс прораммы Privaria.

Программа имеет простой графический интерфейс (см. рисунок 9.1) и для настройки не требуется никаких специальных знаний. Пользователю достаточно указать адрес или имя удаленного компьютера, а также имя и пароль для доступа к удаленному хосту.
Режим работы своего компьютера пользователь задает простым выбором опций (D и E на рисунке 9.1).

10 Предотвращение спама и проникновения вирусов в сеть

Современная среда обмена сообщениями электронной почты содержит уже более 50% незапрошенных сообщений (так называемого спама) и писем, содержащих вирусы или троянские программы, которые зачастую способны превратить пользовательский компьютер в открытый транслятор для рассылки спама и вирусов или включить машину в распределенную DoS-атаку. Поэтому вопросы фильтрации электронной почты занимают важное место в работе администратора безопасности. В этом разделе рассматриваются концептуальные вопросы предотвращения нежелательной электронной почты и вирусов, а также некоторые программы, обеспечивающие безопасность системы электронной почты.

10.1 Антивирусные средства

10.1.1 ClamAV

http://www.clamav.net/

Распространяемый по лицензии GPL сканер вирусов **ClamAV** для сред UNIX обеспечивает широкий набор функций:

- сканирование файлов из командной строки;
- высокоскоростной многопотоковый режим работы демона;
- milter-интерфейс для программы sendmail;
- средства обновления базы данных с поддержкой цифровых сигнатур;
- C-библиотека для сканирования вирусов;
- сканирование при обращении к файлу (Linux и FreeBSD);
- обнаружение более 20000 вирусов, червей и троянских программ;
- сканирование архивов RAR (2.0), Zip, Gzip, Bzip2
- сканирование почтовых файлов формата Mbox, Maildir и raw.

Основной задачей разработчиков была интеграция сканера вирусов с постовыми серверами для сканирования вложенных в письма файлов. Программа поддерживает гибкий многопотоковый режим работы в качестве демона, включает утилиту для сканирования файлов из командной строки и средство автоматического обновления через Internet. Работа программы основана на использовании разделяемой библиотеки, которая распространяется с пакетом Clam AntiVirus и позволяет создавать свои приложения.

10.1.2 F-prot

http://www.f-prot.com

10.1.2.1 F-Prot Antivirus для рабочих станций Linux

Пакет **F-Prot Antivirus for Linux Workstations** предназначен для домашних пользователей Linux и основан на машине сканирования F-Prot Antivirus, позволяющий обнаруживать практически все типы вирусов. Программа обеспечивает защиту не только от вирусов, но и макровирусов, червей и троянских программ:

- распознавание более 116 тысяч вирусов и их вариантов;
- возможность выполнить сканирование по расписанию;
- сканирование разделов винчестера, компакт-дисков, дискет, сетевых накопителей, каталогов и отдельных файлов;
- сканирование загрузочных образов на предмет наличия вредоносного кода.

10.1.2.2 F-Prot Antivirus для почтовых серверов Linux x86

Пакет **F-Prot Antivirus for Linux x86 Mail Servers** обеспечивает высокопроизводительную систему сканирования сообщений электронной почты в целях обнаружения вирусов, червей и троянских программ. Пакет работает с почтовыми серверами **Sendmail, Postfix** и **Qmail**.

При обнаружении зараженного или опасного вложения в почтовом сообщении опасный код удаляется из письма. Если такая очистка невозможна, вложенный файл или сообщение целиком удаляется с передачей адресату информации о предпринятых действиях. F-Prot Antivirus может перед дезинфекцией создавать резервные копии входящей почты для обеспечения целостности информации.
F-Prot Antivirus for Linux x86 Mail Servers включает:

- консольный сканер вирусов F-Prot Antivirus;
- демон сканирования F-Prot Antivirus Daemon Scanner;
- программу обновления F-Prot Antivirus
- сканер электронной почты F-Prot Antivirus Mail Scanner
- средства загрузки библиотек F-Prot Antivirus Preloadable Library Call Wrapper
- подключаемые модули (Plug-in) для сканирования на лету сообщений электронной почты в системах Sendmail, Postfix и Qmail

F-Prot Antivirus for Linux x86 Mail Servers обеспечивает:
- распознавание более 116 тысяч вирусов и их вариантов;
- безопасное удаление вирусов без повреждения исходного файла;
- сканирование всех смонтированных файловых систем, каталогов или файлов;
- сканирование архивов;
- автоматическое обновление программ и баз данных;

- запланированное сканирование с использованием заданий cron;
- сканирование сообщений электронной почты, передаваемых и принимаемых серверами Sendmail, Postfix и Qmail.

10.2 Средства предотвращения спама

Методы фильтрации спама можно разделить на три категории

1) фильтрация по адресам хостов и сетей на граничном шлюзе;

2) фильтрация с использованием “черных списков” NSBL;

3) фильтрация по содержимому.

Очевидно, что первый из перечисленных способов является самым эффективным и мощным средством, поскольку позволяет заблокировать попытки соединения с портом SMTP со стороны нежелательных хостов или сетей и тем самым существенно снизить уровень паразитного почтового трафика. Однако этот метод является и самым трудоемким. На мой взгляд, целесообразно применять этот метод для первичной фильтрации, позволяющей избавиться от спама, рассылаемого взломанными или зараженными пользовательскими компьютерами. Для такой фильтрации достаточно составить список блоков адресов, используемых провайдерами для организации доступа в Internet конечных пользователей (кабельное телевидение, модемные соединения, ADSL, беспроводные каналы Wi-Fi, домашние сети). Как показывает опыт, значительная часть спама приходит именно с таких “трансляторов” и блокирование для них доступа к порту SMTP ваших трансляторов позволяет существенно снизить уровень спама и число зараженных писем.

Второй метод также достаточно эффективен, поскольку позволяет заблокировать прием по IP-адресам почтовых трансляторов, включенных в “черные” списки, поддерживаемые с Internet. Однако здесь могут скрываться подводные камни, поскольку приходиться использовать списки, созданные кем-то с учетом своих интересов, а эти интересы могут и не совпадать с вашими[1]. Однако методом анализа представленных администраторами списков описаний политики и пробного использования таких списков вы сможете подобрать для себя один или несколько списков DSBL, которые будут фильтровать достаточно большую часть электронной почты.

Последний метод пожалуй является наименее эффективным с точки зрения администратора, поскольку он способен фильтровать почту только после ее приема. Достаточно часто от разработчиков систем content-фильтрации приходиться слышать совершенно обратные заявления о том, что все системы фильтрации за исключением проверки содержимого совершенного неэффективны[2]. Оставим эти заявления на совести тех, кто их делает и отметим лишь, что системы фильтрации по содержимому почты могут быть достаточно эффективны для конечного пользователя, не желающего видеть мусор в своем почтовом ящике, но вклад их о обеспечение безопасности корпоративной сети достаточно скромен, поскольку при эффективном использовании двух первых методов до content-фильтров доходит уже весьма малая часть спама и вирусов.

На мой взгляд весьма эффективно комбинированное использование всех трех методов с корректировкой правил фильтрации для методов 1 и 2 по результатам фильтрации содержимого.

10.2.1 Фильтрация на граничном шлюзе

Фильтрация спама на граничном шлюзе осуществляется путем создания статических списков доступа, блокирующих подключение к портам SMTP ваших почтовых серверов для определенных хостов и сетей, откуда вы не желаете по тем или иным причинам принимать почту. Выбор фильтруемых трансляторов и сетей определяется принятой в вашей компании политикой и я ограничусь здесь лишь перечислением некоторых категорий, для каждой из которых вы сами должны будете принять решение о фильтрации.

1) Блоки адресов, используемые провайдерами для обеспечения доступа в Internet отдельных пользователей и небольших компаний (модемные подключения по коммутируемым линиям, DSL, Wi-Fi, сети кабельного телевидения и т. п.). Как правило для таких пользователей провайдер обеспечивает услуги по трансляции электронной почты с помощью выделенного для этих целей сервера и появление трансляторов в таких блоках с большой вероятностью говорит о несанкционированной рассылке почты (зачастую это происходит без ведома владельца компьютера).

2) Территориально-зависимые блоки адресов. Распределение IP-адресов осуществляется большими блоками для нескольких первичных центров (NIC) и, если вы совершенно не ждете почты из Южной Америки вы можете совершенно спокойно закрыть порт 25 для всей сети 200.0.0.0/8, поскольку весь этот блок адресов распределен между хостами и сетями данного континента. Аналогичным образом можно выделить и другие блоки адресов для фильтрации.

Статическая фильтрация требует ручной работы и является достаточно утомительным занятием, но дает хорошие результаты, поскольку просто не пускает в вашу сеть весь почтовый трафик из нежелательных сетей. Еще раз подчеркну, что выбор адресов для фильтрации должен определяться интересами вашей компании и принятой политикой безопасности.

При наличии большого числа фильтров проверка пакетов во всех правилах цепочки может занять достаточно много времени и привести к задержкам на граничном шлюзе. Если для почтового трафика задержки не имеют существенного значения, то для других типов данных это может оказаться совершенно нежелательным. Для того, чтобы избавиться от задержек достаточно просто сначала выделить трафик SMTP и проверять адреса уже только для SMTP-пакетов, для которых задержка не имеет существенного значения. Сделать это можно например так:

1 *Возможны и сюрпризы типа того, который устроили держатели списка Osirusoft. Поняв, что они просто не справляются с поддержкой созданной службы, они просто зафильтровали одной строкой весь Internet.*
2 *См. например публикации на сайте http://www.spamtest.ru.*

- в начале цепочки FORWARD поместить правило типа

```
-A INPUT -p tcp -m tcp --dport smtp ! -i eth0 --tcp-flags SYN SYN -j ChkInMail
```

- в пользовательскую цепочку ChkInMail включить все фильтры по адресам отправителей пакетов SMTP

```
   ...
-A ChkInMail -s 4.0.0.0/15 -j RETURN
-A ChkInMail -s 4.2.0.0/16 -j RETURN
-A ChkInMail -s 4.9.0.0/16 -j RETURN
-A ChkInMail -s 4.0.0.0/12 -j DropSMTP

-A ChkInMail -s 4.16.128.0/19 -j DropSMTP
-A ChkInMail -s 4.16.160.0/21 -j DropSMTP
-A ChkInMail -s 4.16.192.0/18 -j DropSMTP

-A ChkInMail -s 4.21.41.0/24 -j DropSMTP
-A ChkInMail -s 4.21.42.0/24 -j DropSMTP
...
```

В приведенном примере первое правило позволяет выделить попытки соединения (**--tcp-flags SYN SYN**) с портом SMTP (**--dport smtp**) для пакетов, принятых от любых интерфейсов, кроме eth0[1] (**! -i eth0**) для передачи цепочке **ChkInMail**. Эта пользовательская цепочка содержит набор фильтров, отбрасывающих нежелательные попытки доступа к порту SMTP с записью информации о таких попытках в журнальный файл (эти операции выполняются пользовательской цепочкой **DropSMTP**). Обратите внимание, что в некоторых правилах цепочки **ChkInMail** указана операция **RETURN**. Это сделано для минимизации числа правил в цепочке. В нашем случае, блокируется вся почта из блока адресов 4.0.0.0-4.15.255.255, за исключением блоков 4.0.0.0-4.2.255.255 и 4.9.0.0-4.9.255.255. Очевидно, что такой же фильтр можно было задать и с помощью набора

```
-A ChkInMail -s 4.3.0.0/16 -j DropSMTP
-A ChkInMail -s 4.4.0.0/16 -j DropSMTP
-A ChkInMail -s 4.5.0.0/16 -j DropSMTP
-A ChkInMail -s 4.6.0.0/16 -j DropSMTP
-A ChkInMail -s 4.7.0.0/16 -j DropSMTP
-A ChkInMail -s 4.8.0.0/16 -j DropSMTP
-A ChkInMail -s 4.10.0.0/16 -j DropSMTP
-A ChkInMail -s 4.11.0.0/16 -j DropSMTP
-A ChkInMail -s 4.12.0.0/16 -j DropSMTP
-A ChkInMail -s 4.13.0.0/16 -j DropSMTP
-A ChkInMail -s 4.14.0.0/16 -j DropSMTP
-A ChkInMail -s 4.15.0.0/16 -j DropSMTP
```

но в последнем случае число правил существенно больше, а при наличии большого числа цепочек это может оказать существенное влияние на скорость обработки пакетов.

10.2.2 DNSBL

http://en.wikipedia.org/wiki/DNSBL

Практически все современные почтовые трансляторы поддерживают функции DNSBL[2], позволяющие отвергать или принимать почту на основании проверки наличия адреса передающего транслятора в тех или иных “черных списках”

Механизм **DNSBL**, поддерживаемый большинством почтовых трансляторов, работает на основе публичных списков IP-адресов, позволяющих делать запросы через сеть Internet. По названию технологии можно догадаться, что этот метод работает на основе системы DNS (Domain Name System), используемой в сети Internet для преобразования имен хостов в адреса IP и обратно. DNSBL просто использует публичные списки адресов, с которых приходит достаточно большой по мнению держателя списка объем незапрошенной почты (спама). Почтовый сервер получателя (MTA) можно настроить так, чтобы он отвергал почту, пришедшую с адресов, включенных в проверяемые списки. Например, в наиболее распространенном почтовом сервере sendmail это осуществляется с помощью строк типа

```
FEATURE(dnsbl, `opm.blitzed.org', ` Host $&{client_addr} is open proxy - message
rejected. See http://opm.blitzed.org')
FEATURE(dnsbl, `proxies.blackholes.easynet.nl', ` Host $&{client_addr} is open proxy -
message  rejected. See http://proxies.blackholes.easynet.nl/errors.html')
```

помещаемых в конфигурационный файл sendmail.mc[3]. При получении запроса на прием почты программа sendmail будет отправлять по указанным в конфигурационном файле именам списков запрос на проверку адреса транслятора. Если список DNSBL возвращает значение типа 127.0.0.2[4].

Первый список DNSBL (Real-time Blackhole List или RBL) был основан в 1997 Paul Vixie как часть службы по

1 *Интерфейс брандмауэра в сторону локальной сети*
2 DNS-based Blackhole List - “черный список” на основе сервера доменных имен.
3 *После редактирования файла sendmail.mc не забудьте скомпилировать его в конфигурационный файл sendmail.cf с помощью препроцессора m4. Инструкции по компиляции обычно приводятся в заголовке файла sendmail.mc.*
4 *Для обозначения пазличных типов некорректного использования почтовых трансляторов в последнем байте возвращаемого IP-адреса обычно указываются значения от 2 до 9. Описание использования этих значений в некоторых популярных списках можно найти на сайте http://www.email-policy.com/Spam-black-lists.htm.*

предотвращению неправомерного использования электронной почты MAPS[1]. Будучи влиятельным программистом и администратором, Vixie сумел убедить разработчиков sendmail и других почтовых серверов реализовать в своих программах поддержку клиентских функций RBL. Это позволяло программам запрашивать службу RBL и по результатам отклика принимать решение о трансляции почты. Благодаря возможности фильтрации нежелательной почты сервис RBL стал очень популярным и вскоре появилось множество подобных служб.

Кроме предоставления доступа к “черным спискам” сотрудники MAPS и множество добровольцев вступало в контакты с администраторами хостов и сетей, откуда приходил спам, объясняя некорректность работы их серверов и недопустимость рассылки спама. Поскольку до середины 1990-х годов для почтовых серверов считалась нормальной политика открытой трансляции (open relay), разъяснительная работа по отказу от такой практики сыграла важную роль и помогла существенно снизить число открытых трансляторов, через которые каждый желающий мог рассылать спам.

Несколько позднее Алан Браун (Alan Brown) создал систему ORBS (Open Relay Behavior-modification System), которая пыталась в автоматическом режиме находить в сети открытые трансляторы. Эта система сыграла важную роль, поскольку на начальных этапах существования спама далеко не все администраторы понимали необходимость закрывать свои почтовые трансляторы от нежелательных пользователей.

В 2003 году многие службы DNSBL подверглись DoS-атакам. Организаторы этих атак обнаружены не были, но некоторые аналитики сочли атаки делом рук спамеров, которым эти службы усложняли жизнь. В результате некоторые компании прекратили поддержку служб DNSBL. В частности, в августе 2003 гола компания Osirusoft, поддерживавшая несколько DNSBL (в том числе, SPEWS), прервала[2] работу этих служб после нескольких недель непрерывных атак.

MAPS с 2001 года предоставляет свои услуги DNSBL на коммерческой основе.

Многие организации и частные лица возражают против использования DNSBL и других методов предотвращения спама провайдерами Internet. Правомерность или неправомерность такой фильтрации весьма неоднозначны и споры на эту тему могут длиться десятилетиями. Однако эти дебаты не затрагивают техническую сторону вопроса, относясь скорее к юридической и моральной, посему мы не будем здесь останавливаться на рассмотрении данной проблемы, а попытаемся лучше разобраться в принципах работы DNSBL.

10.2.2.1 Функционирование DNSBL

Для организации сервиса DNSBL требуется домен, в котором будет размещаться сервер DNSBL, сервер имен (DNS) для этого домена и список публикуемых адресов.

Возможна реализация DNSBL на основе популярного сервера доменных имен BIND, однако эта программа малоэффективна для зон, содержащих большое число адресов[3]. Существуют разработанные специально для поддержки списков DNSBL программы (в частности, rbldnsd[4] и rbldns[5]).

Основная работа при создании DNSBL будет заключаться в наполнении списка адресов трансляторов и сетей, откуда следует блокировать почту. Списки DNSBL, предназначенные для публичного использования должны содержать только адреса, соответствующие политике (см. параграф 10.2.2.3), провозглашенной и опубликованной для данного списка.

10.2.2.2 Запросы DNSBL

Почтовый сервер, получив входящий запрос на организацию соединения, может обращаться к выбранным администратором службам DNSBL для проверки наличия адреса вызывающего транслятора в “черных списках”

1. Почтовый сервер определяет IP-адрес вызывающей стороны (скажем, 192.168.42.23) и преобразует его в реверсный формат записи (23.42.168.192).
2. К реверсному адресу добавляется имя зоны DNSBL для генерации запроса (23.42.168.192.spammers.example.net).
3. Сформированный запрос DNS передается серверу DNSBL на предмет проверки наличия в его списке записи типа A. В ответ на запрос почтовый сервер получает адрес, указывающий наличие проверяемого транслятора в списке или код **NXDOMAIN**[6], если адреса нет в списке DNSBL.
4. При наличии адреса в списке сервер DNSBL может дополнительно возвращать текстовую запись TXT), содержащую те или иные комментарии. Многие службы DNSBL в качестве комментария указывают причину включения адреса в список.

Поиск записей в DNSBL весьма напоминает обратное преобразование имен, когда по адресу хоста определяется его доменное имя. Разница заключается лишь в том, что DNSBL ищет адресные записи A вместо используемых при обратном преобразовании записей типа PTR и поиск осуществляется в прямой (например, spammers.example.net), а не реверсной зоне (in-addr.arpa).

Серверы DNSBL при обнаружении адреса в списке возвращают запрашивающему почтовому серверу адрес из сети 127.0.0.0/8. Списки значений, возвращаемых некоторыми популярными службами DNSBL, можно найти на сайте http://www.email-policy.com/Spam-black-lists.htm.

1 *Mail Abuse Prevention System - система предотвращения недопустимого использования электронной почты.*
2 *Предварительно поместив в список фильтрации всю сеть Internet (0.0.0.0/0)*
3 *Списки DNSBL могут содержать миллионы записей.*
4 *См. сайт* http://www.corpit.ru/mjt/rbldnsd.html
5 http://cr.yp.to/djbdns.html
6 *No such domain - домен неизвестен серверу.*

10.2.2.3 Политика DNSBL

Различные службы DNSBL используют разные принципы включения адресов в “черный список”. Политика обычно публикуется на сайте и включает:

- **Тип списка** - какого типа сайты содержатся в списке (открытые трансляторы, серверы proxy, источники спама, блоки адресов, используемые провайдерами для доступа и т. п.).
- **Условия включения в список -** способ определения адресов для включения в список (автоматический поиск в сети, факты реального спама с этих адресов, жалобы пользователей и т. п.).
- **Время жизни записей в списке** - как долго запись находится в списке, способ удаления (ручной или автоматический) и т. д.
- **Возможность исключения адреса** - описание процедур исключения адреса из списка и требований, предъявляемых к хостам, для которых получен запрос на исключение из “черного списка”.

Публикуемая политика может содержать описание и других аспектов работы данной службы, но перечисленные выше элементы политики должны быть указаны, чтобы пользователи этого сервиса и “пострадавшие” от него могли понимать, как служба работает и как можно исключить адрес из списка.

10.2.2.4 DRBL

http://www.drbl.ofisp.org/

В российском сегменте Internet несколько лет назад возник интересный, но, к сожалению, не получивший достаточно широкого распространения и поддержки проект по созданию распределенной системы “черных списков”, в которой могли участвовать все желающие. Основная идея проекта заключается в создании множества списков DNSBL, поддерживаемых администраторами различных сетей. При генерации зоны DRBL на конкретном узле для создания рабочего списка фильтрации может использоваться собственная зона и взвешенный результат любых других зон участников проекта. Однако незначительное число участников проекта не позволяет до сих пор создать действительно “взвешенную” результирующую зону, поэтому при использовании чужих списков увеличивается вероятность ложных срабатываний. Ниже приведена информация об этом проекте, почерпнутая из опубликованного на сайте документа[1].

Аббревиатура DRBL означает Distributed Realtime Block List, что можно перевести как "распределенный черный список реального времени". До недавнего времени в Internet успешно функционировал сервис MAPS RBL. К сожалению, владельцы MAPS LLC объявили о прекращении бесплатного обслуживания 31 июля 2001 года, что привело к резкому снижению числа пользователей MAPS. Однако, существуют и другие списки фильтрации DNSBL, которые продолжают предоставлять свои услуги пользователям Internet бесплатно.

DRBL работает несколько иначе, чем MAPS RBL. Вместо одной базы данных, поддерживаемой конкретной компанией, DRBL предлагает каждому желающему завести свою базу данных (многие администраторы имеют такие списки) и сделать информацию из нее доступной для остальных пользователей Internet. В результате обеспечения свободного доступа к таким спискам фильтрации другие пользователи могут строить свои фильтры, анализируя и сопоставляя информацию, полученную из различных источников.

Основная цель проекта DRBL - предоставить механизм автоматического обмена информацией из различных фильтров, поддерживаемых администраторами для решения своих задач. В проекте участвуют самые разные сети, от крупных провайдеров до отдельных групп пользователей. Разумеется, каждый участник проекта самостоятельно принимает решение о приеме или фильтрации почты от того или иного транслятора или сети. Если у вас есть свой почтовый сервер, никто не имеет права навязать вам те или иные положения вашей политики фильтрации.

Единственным жестким требованием проекта DRBL является необходимость реального использования в своей сети зоны, предоставляемой для публичного доступа. Здесь важно понимать, что тот или иной администратор может использовать фильтры не только для блокирования спама, поскольку существует множество других причин, по которым почту могут фильтровать. Установка фильтров на конечном узле не может считаться насилием и не нарушает связности сети.

Кроме прочего, все это означает, что никто не вправе **требовать** убрать какой-либо адрес из DRBL-зоны. Речь может идти только о просьбе. Об этом следует помнить при обращениях к администраторам других зон.

Сеть, принявшая решение об участии в системе DRBL, создает две DNS-зоны для использования в DRBL - голосующую и рабочую. Обычно эти зоны называются vote.drbl.<domain>.ru и work.drbl.<domain>.ru.

Сети и хосты, почта из которых в данную сеть по каким-либо причинам не принимается, заносятся в голосующую зону в виде записей типа **A** и **TXT**:

```
*.57.168.192    IN      A       127.0.0.2
                IN      TXT     "Spammers network blocked"
```

Это означает, что почта из сети 192.168.57.0/24 будет блокироваться, а в качестве причины отказа (комментария) будет передаваться строка "Spammers network blocked".

После создания голосующей зоны администратор должен определить политику генерации рабочей зоны, которая и будет служить для создания реальных фильтров. При разработке политике следует принять решение о том, какие голосующие зоны будут принимать участие в генерации рабочей зоны и с каким уровнем достоверности принимаются данные из каждой используемой зоны. Каждой голосующей зоне приписывается некоторое скалярное значение, называемое **весом** зоны, а при генерации рабочей зоны задается пороговое значение (**threshold**), по достижении которого для соответствующей сети или хоста создается фильтр. Адресу, включенному в любую из голосующих зон, приписывается вес, равный сумме весов для зон, в которых данный адрес присутствует. В результирующую зону адрес попадает **тогда и только тогда, когда его вес превышает пороговое значение**.

1 *http://www.drbl.croco.net/faq.html*

Предположим, что мы решили создать узел DRBL в своей сети **OurNet.ru** и создали зоны **vote.drbl.OurNet.ru** и **work.drbl.OurNet.ru**. Предположим также, что мы решили использовать при генерации зоны из других сетей, называющиеся vote.drbl.network-1.ru, vote.drbl.network-2.ru, vote.drbl.network-3.ru, vote.drbl.network-4.ru, vote.drbl.network-5.ru администраторам сетей network-1 и network-2 мы доверяем как самим себе, в сети network-3 мы уверены чуть меньше, а сети network-4 и network-5 знаем достаточно мало, хотя и не имеем причин для недоверия (зачем иначе мы бы стали пользоваться их зонами). Установим порог срабатывания равным единице, а зонам присвоим следующие веса:

```
vote.drbl.OurNet.ru         1
vote.drbl.network-1.ru      1
vote.drbl.network-2.ru      1
vote.drbl.network-3.ru      0.8
vote.drbl.network-4.ru      0.4
vote.drbl.network-5.ru      0.4
```

В этом случае, достаточно какому-либо адресу появиться в голосующей зоне нашей сети или сетей network-1 и network-2, которым стопроцентно доверяем - и этот адрес автоматически попадет в нашу рабочую зону, то есть почта с него приниматься не будет.

Если администрация сети network-3 примет решение о блокировке какого-то адреса, то сразу он в нашу рабочую зону не попадет - для этого необходимо, чтобы его внес в свою голосующую зону кто-то еще (например, если этот адрес блокируется хотя бы в одной из сетей network-4 и network-5, его вес достигает порога). Суммирование весов из различных зон обусловлено тем, что включение адреса в зоны не связанных между собой сетей повышает достоверность трактовки этого адреса, как источника нежелательной почты.

Если адрес включен только в зоны сетей 4 и 5, этого будет недостаточно для его включения в нашу рабочую зону.

10.2.3 Фильтрация по содержимому

Фильтрация по содержимому писем используется на последнем этапе обработки электронной почты, когда сообщение уже принято транслятором. Благодаря наличию полного текста сообщения и включенных в него файлов, можно провести анализ письма и определить присутствие в нем недопустимых вложений (в частности, вирусов). Современные content-фильтры используют множество весьма изощренных методов проверки электронной почты и практически не дают ложных срабатываний при правильной настройке. Однако зачастую такая настройка достаточно сложна и не все фильтры способны анализировать тексты, содержащие символы кириллицы.

10.2.4 MailScanner

http://www.mailscanner.info

MailScanner - это широко распространенная система фильтрации электронной почты с открытым кодом, которая на сегодняшний день обрабатывает более 500 миллионов почтовых сообщений ежедневно, удаляя по 2 миллиона вирусов и 75 миллионов образцов спама. MailScanner используется более чем на 20 000 сайтов, обеспечивая защиту государственных организаций, коммерческих структур, учебных заведений и т. п. Эту программу используют сегодня многие провайдеры для защиты своих пользователей от спама и вирусов.

MailScanner проверяет все почтовые сообщения на предмет наличия в них вирусов, спама или кода, направленного против уязвимых мест системы. Программа не привязана к какому-либо конкретному сканеру вирусов и может работать с различными программами сканирования (включая открытые).

- Сканирование электронной почты с использованием любой комбинации из 14 поддерживаемых сканеров.
- Автоматическое обновление всех установленных сканеров вирусов (по умолчанию каждый час).
- Фильтрация более 95% спама[1] с использованием различных технологий, включая эвристические.
- При обнаружении спама программа может помечать его, отвергать,

Рисунок 10.1. Интерфейс Webmin для настройки программы MailScanner.

1 *Для спама на русском языке уровень фильтрации может быть несколько ниже.*

отбрасывать, архивировать или пересылать по другим адресам для проверки администратором.
- Из порно-спама могут удаляться все графические компоненты.
- Сканирование электронной почты на предмет наличия кода, направленного на несанкционированное использование уязвимых мест популярных почтовых программ с автоматической корректировкой сообщений “на лету” (когда это возможно) или отправка таких сообщений в “карантинную зону”
- Высокая производительность (1 ПК может обрабатывать более полутора миллионов сообщений в сутки).
- Защита от всех известных DoS-атак и чрезмерного потребления системных ресурсов.
- Эффективная система настройки позволяет каждому администратору создать свои наборы правил и конфигурации в соответствии с задачами[1].
- Простота установки и настройки. Программа может использоваться даже с заданными по умолчанию параметрами конфигурации.

Установка программы достаточно проста. Нужно распаковать архив и запустить установочный сценарий **install.sh** из текущего каталога.

Для настройки параметров работы программы и правил фильтрации используются конфигурационные файлы, хранящиеся в каталоге /etc/MailScanner/. Существует модуль[2] для настройки программы с помощью Web-интерфейса Webmin (параграф 11.3 на стр. 228), показанный на рисунке 4.2.

10.2.5 SpamAssassin

http://spamassassin.org

Программа SpamAssassin[tm] является почтовым фильтром, обеспечивающим идентификацию и блокировку спама. Используя набор правил, программа выполняет большое число проверок (в том числе, эвристических) заголовков и текста сообщений для идентификации спама:

- **анализ заголовков** - спамеры используют множество методов, позволяющих скрыть источник спама и представляющих каждое письмо как вполне легитимное (например, подписку на ту или иную рассылку).
- **анализ текста письма** - для спама характерен определенный набор стилей сообщения и большой набор ухищрений для преодоления почтовых фильтров (подмена символов, преднамеренные ошибки и т. п.).
- “**черные списки**” - SpamAssassin поддерживает фильтрацию с использованием большинства существующих черных списков (mail-abuse.org, ordb.org и др).
- **Razor** - Vipul's Razor[3] представляет собой базу данных по спаму, содержащую сигнатуры сообщений. Поскольку спамеры рассылают обычно множество копий однотипных сообщений, база данных Razor может получать новую сигнатуру еще до того, как к вам придет первая копия такого письма, что позволит вам автоматически блокировать прием таких сообщений.

После проверки почта по желанию может помечаться как спам для последующей фильтрации средствами пользовательских почтовых клиентов.

SpamAssassin не требует сколь-нибудь серьезной настройки или постоянного обновления. Фильтрация осуществляется практически только на основе свойств сообщений. Настройки параметров фильтрации спама хранятся в конфигурационном файле, который можно изменять с помощью текстового редактора. Имеется интерфейс (см. рисунок 10.2) для настройки SpamAssassin в системе удаленного управления Webmin (см. параграф 11.3 на стр. 228).

Программа распространяется с консольным интерфейсом для фильтрации почты, Perl-модулем Mail::SpamAssassin и набором дополнительных модулей Perl, позволяющих использовать SpamAssassin в соответствии с потребностями пользователей.

1 *Например, провайдеры Internet могут создавать различные правила фильтрации для разных доменов.*
2 *Вы можете загрузить этот модуль с сайта http://lushsoft.dyndns.org/mailscanner-webmin/.*
3 *http://razor.sourceforge.net/*

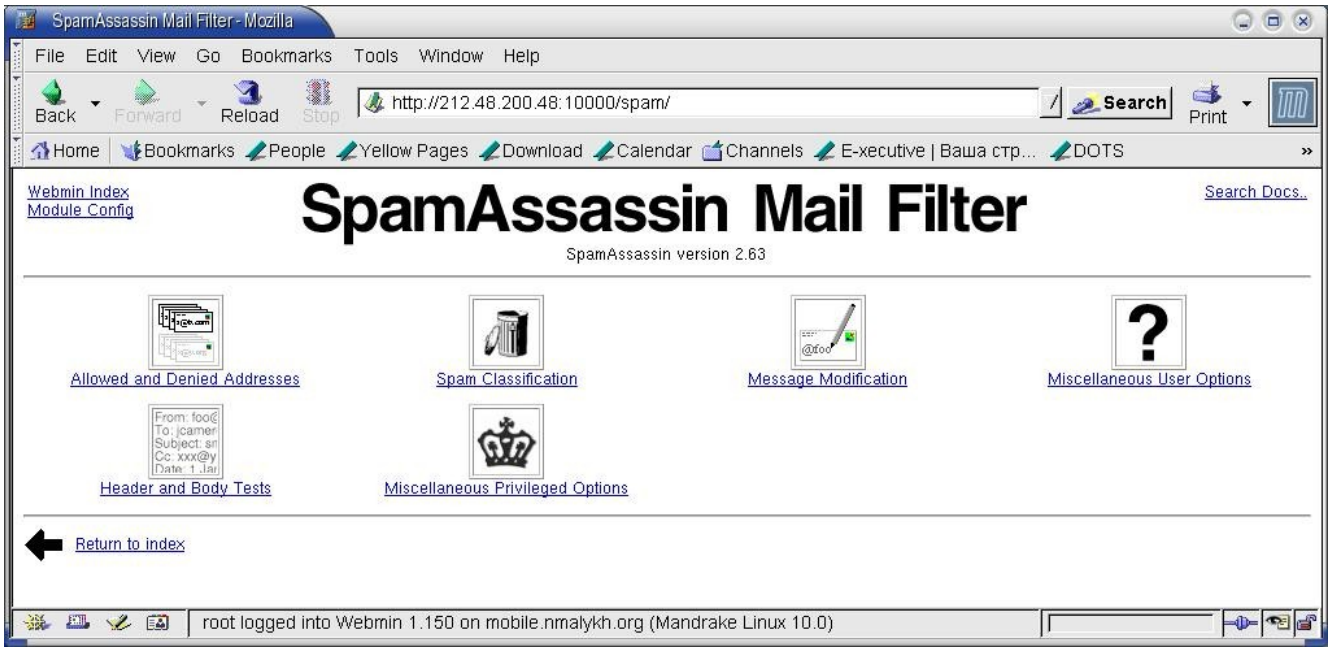


Рисунок 10.2 Интерфейс Webmin для настройки параметров SpamAssassin

Основные свойства программы

- **Универсальность** - SpamAssassin широко используется для обнаружения спама на сетевых серверах и пользовательских компьютерах. Широкое использование программы в сети Internet существенно осложняет жизнь спамеров и вынуждает их постоянно искать новые методы преодоления фильтров.
- **Открытый код** - программа представляет собой набор сценариев на языке Perl, которые распространяются бесплатно в исходных текстах.
- **Простота расширения** - правила и параметры конфигурации сохраняются в простых и понятных администратору (или пользователю) текстовых файлах, которые можно изменять и дополнять с помощью любого редактора.
- **Гибкость** - логика SpamAssassin реализована в хорошо организованном абстрактном интерфейсе API. Благодаря этому возможности программы могут использоваться не только на этапе окончательной доставки почты, но и в других приложениях, способных работать с классами Mail::SpamAssassin.

11 Инструменты администратора

Для эффективной работы администратора безопасности требуется множество инструментов, позволяющих настраивать параметры системы, управлять хостами и службами, обеспечивать мониторинг работы сети, поиск неисправностей и анализ инцидентов. В этой главе рассматриваются примеры некоторых программ, которые на мой взгляд могут оказать существенную помощь в работе администратора. Несомненно, у каждого администратора имеется свой набор утилит и инструментов и приведенные здесь описания могут лишь послужить для начинающих администраторов UNIX к подбору своего инструментария.

11.1 Системные утилиты UNIX

11.1.1 Пакет coreutils

Пакет **coreutils** создан на базе широко распространенных пакетов GNU **fileutils**, **sh-utils** и **textutils** и включает:

- средства для работы с файлами;

- сценарии shell;

- средства обработки текста.

Ниже описаны некоторые из утилит пакета, так или иначе связанные с обеспечением безопасности системы.

11.1.1.1 Chgrp

Команда **chgrp** служит для изменения группы, которой принадлежит файл.

chgrp [опции] <группа> <файл...>

Новая группа для файла может быть указана именем или числовым идентификатором (GID).

11.1.1.1.1 Опции POSIX

-R

задает рекурсивное изменение владельца для каталогов и их содержимого.

--

завершает список опций.

11.1.1.1.2 Опции чернового стандарта AUSTIN

-h

задает необходимость изменения группы для символьной ссылки, а не объекта, на который эта ссылка указывает. Если система не поддерживает установку группы для символьных ссылок, команда ничего не делает.

-H

при использовании вместе с рекурсией (**-R**) задает замену группы для каждого объекта, указанного в командной строке и являющегося символьной ссылкой на каталог. Изменяется также группа для всех файлов в иерархии этого каталога. Этот метод называют полулогическим.

-L

при использовании вместе с рекурсией (**-R**) задает замену группы для каждого объекта, указанного в командной строке или встреченного при прохождении по дереву каталогов и являющегося ссылкой на каталог. Изменяется также группа для всех файлов в иерархии этого каталога. Этот метод называют логическим.

-P

при использовании вместе с рекурсией (**-R**) задает для каждого объекта, указанного в командной строке или встреченного при прохождении по дереву каталогов и являющегося ссылкой на каталог, замену группы только для символьной ссылки без изменения группы для объекта, на который эта ссылка указывает. Если система не поддерживает группы для символьных ссылок, команда не делает ничего. Этот метод замены группы называется физическим и используется командой по умолчанию.

-R

задает рекурсивную замену группы для всего дерева каталогов, указанного в командной строке. Поведение команды зависит от наличия опций **-H**, **-L** и **-P**.

11.1.1.1.3 Опции расширения GNU

-c (--changes)

задает вывод подробного отчета для каждого файла, для которого действительно изменяется группа.

-f (--silent, --quiet)

отключает вывод сообщений о невозможности смены группы.

-h (--no-dereference)

указывает на необходимость работы с символьными ссылками, а не с файлами, на которые эти ссылки указывают. Данная опция доступна только при поддержке в системе функции lchown.

-v (--verbose)

задает вывод детальной информации для каждого файла.

-R (--recursive)
задает рекурсивное изменение владельца для каталогов и их содержимого.

--dereference
задает необходимость изменения группы для файла, а не для указывающей на этот файл символьной ссылки.

--reference=rfile
устанавливает для файла принадлежность к группе в соответствии с группой для файла **rfile**.

--help
выводит краткую справку о работе с программой.

--version
выдает информацию о номере версии программы.

--
указывает на завершение списка опций.

11.1.1.1.4 Соответствие стандартам

Стандарт POSIX 1003.2 требует поддержку только для опции **-R**. Использование других опций в POSIX-системах может не поддерживаться.

11.1.1.2 Chmod

Программа **chmod** служит для изменения прав доступа к файлам и каталогам. Программа использует синтаксис

chmod [опции] <режим> <файл...>
Команда chmod изменяет права доступа каждого указанного файла в соответствии со значением параметра **<режим>**, которое может быть представлено в символьном (параграф 11.1.1.2.2) или восьмеричном (параграф 11.1.1.2.3) формате.

Команда **chmod** никогда не изменяет права, установленные для символьных ссылок, поскольку системный вызов chmod() не работает с символьными ссылками. Однако команда изменяет права доступа для связанного со ссылкой реального файла. Символьные ссылки, встречающиеся при рекурсивной обработке файлов, команда chmod игнорирует.

11.1.1.2.1 Опции chmod

Поддерживаемые программой chmod опции определяются типом вашей ОС (POSIX или GNU). Вопросы соответствия стандартам кратко описаны в параграфе 11.1.1.2.4 (стр. 189).

11.1.1.2.1.1 Опции POSIX

-R
задает рекурсивное изменение прав доступа к файлам для каталогов и их содержимого.

11.1.1.2.1.2 Опции GNU

-R
задает рекурсивное изменение прав доступа к файлам для каталогов и их содержимого.

-c (--changes)
задает вывод подробного описания действий для каждого файла, права доступа к которому реально изменяются.

-f (--silent, --quiet)
отключает вывод сообщений об ошибках для файлов, права доступа к которым не удается изменить.

-v (--verbose)
задает вывод информации об всех действиях по отношению к каждому файлу.

--help
обеспечивает вывод на экран краткой справки о работе с программой и завершает работу.

--version
выводит на экран информацию о номере версии программы и завершает работу.

--reference=rfile
задает установку для файлов прав доступа, полностью совпадающих с правами доступа к эталонному файлу **rfile**. Эта опция поддерживается, начиная с fileutils4.0.

--
завершает список опций.

11.1.1.2.2 Символьный формат задания прав доступа

Символьный формат представления прав доступа имеет вид:

[ugoa][[+-=][rwxXstugo]...][,...]
Первый элемент строки параметра задает пользователей, для которых изменяются права доступа. Этот элемент может содержать произвольную комбинацию символов **ugoa**, каждая буква которой определяет категорию

пользователей, чьи права будут установлены командой:

u - владелец файла;

g - другие пользователи из группы;

o - прочие пользователи, не входящие в группу;

a - все пользователи (эквивалент **ugo**).

Если параметр не содержит ни одного из описанных символов, по умолчанию будет использоваться режим **a**, но биты **umask** не будут изменены командой.

Второй элемент указывает тип вносимых командой изменений прав доступа. Оператор **+** добавляет указанные далее права доступа к имеющимся правам доступа, **-** удаляет указанные права, а **=** присваивает заданные права независимо от текущего состояния прав доступа к файлу.

Символы **rwxXstugo** в третьем элементе параметра определяют права доступа, устанавливаемые или удаляемые командой.

r - определяет возможность чтения файла;

w - определяет возможность записи в файл;

x - для файлов определяет возможность выполнения содержащейся в файле программы или сценария; для каталогов - доступ внутрь данного каталога;

X - этот флаг действует аналогично **x**, но применяется для файлов и каталогов, в правах доступа к которым уже присутствует атрибут **x** хотя бы для одной категории пользователей;

s - определяет состояние битов SUID и SGID (см. параграф 2.3.3 на стр. 42);

t - определяет состояние sticky-бита, при наличии которого в правах доступа к каталогу удаление файлов из данного каталога разрешается только владельцам конкретного файла или владельцу всего каталога. Этот бит удобен для таких каталогов, куда каждый пользователь может записывать свои файлы, но никто не имеет права удалять чужие (например, /tmp);

u - установить для других пользователей такие же права доступа, которые имеет владелец файла;

g - устанавливает для других пользователей такие же права доступа, которые предоставлены пользователям из владеющей файлом группы;

o - установить для указанной категории пользователей такие же права доступа, которые предоставлены прочим пользователям[1].

Например, команда **chmodg-s** будет сбрасывать флаг SGID, **chmod ug+s** - установит биты SUID и SGID,

а команда **chmod o+s** просто не делает ничего.

11.1.1.2.3 Числовой формат задания прав доступа

При использовании числового формата для записи прав доступа используется строка из 4 восьмеричных цифр[2], каждая из которых представляет сумму (объединение) масок прав доступа для чтения, записи и исполнения (4, 2 и 1, соответственно). Если при задании прав доступа указано меньшее количество восьмеричных цифр, старшие разряды дополняются нулями. Первая цифра определяет установку бита SUID (4), SGID (2) и sticky-бита (1). Вторая цифра задает права владельца, третья - права группы и четвертая - права прочих пользователей системы.

11.1.1.2.4 Соответствие стандартам

Стандарт POSIX 1003.2 требует поддержку только для опции -R, а другие опции в POSIX-системах могут не работать. Кроме того, этот стандарт не включает бит **t** и не требует от программы **chmod** соблюдения корректности при сбросе или отказе в установке битов SUID и SGID, если биты, дающие права на выполнение программы, сброшены. Не требует этот стандарт и учета программой chmod значения бита **s**.

Трактовка дополнительных битов доступа (в частности sticky-бита **t**) отличается в разных версиях UNIX. Описание трактовки таких битов в системах Linux вы можете найти в файле Documentation/mandatory.txt дистрибутива ядра.

11.1.1.3 Chown

Команда **chown** используется для смены владельца и группы применительно к файлу или множеству файлов.

```
chown [опции] пользователь[:группа] файл...
```

В качестве имени владельца/группы команда воспринимает первый аргумент, не являющийся опцией. Если задано только имя пользователя (или UID), этот пользователь становится владельцем указанных файлов, а принадлежность к группе для файлов не изменяется. Если после пользователя через двоеточие[3] указана группа (имя или GID), изменяется также принадлежность файлов к группе.

Версии GNU позволяет указывать только группу (с точкой или двоеточием перед именем или GID), не задавая владельца. В таких случаях изменяется только принадлежность файлов к группе, как это делает команда chgrp (см.

1 *Т. е. всем кроме владельца и членов группы, которой принадлежит файл.*
2 *Цифры от 0 до 7.*
3 *В версиях GNU разрешается в качестве разделителя использовать точку (.), в то время как стандарт POSIX не допускает такого использования, поскольку точка может быть частью имени пользователя.*

стр. 187)

11.1.1.3.1 Опции POSIX

-R
задает рекурсивное изменение владельца для каталогов и их содержимого.

--
завершает список опций.

11.1.1.3.2 Опции расширения GNU[1]

-c (--changes)
задает вывод подробного отчета для каждого файла, для которого действительно изменяется владелец или группа.

-f (--silent, --quiet)
отключает вывод сообщений о невозможности смены владельца или группы.

-h (--no-dereference)
указывает на необходимость работы с символьными ссылками, а не с файлами, на которые эти ссылки указывают. Данная опция доступна только при поддержке в системе функции lchown.

-v (--verbose)
задает вывод детальной информации для каждого файла.

-R (--recursive)
задает рекурсивное изменение владельца для каталогов и их содержимого.

--dereference
задает необходимость изменения владельца для файла, а не для указывающей на этот файл символьной ссылки.

--reference=rfile
устанавливает владельца файла в соответствии с владельцем файла **rfile**.

--help
выводит краткую справку о работе с программой.

--version
выдает информацию о номере версии программы.

--
указывает на завершение списка опций.

11.1.1.4 Groups

Команда **groups** служит для просмотра списка групп, в которые включен пользователь.

Синтаксис

groups [<имя пользователя>, <имя пользователя>]
Команда **groups** выводит основную и дополнительные группы для каждого пользователя, указанного в командной строке. Вывод списка групп для каждого пользователя начинается с отдельной строки, содержащей в начале имя пользователя. После имени пользователя и двоеточия указывается имя первичной группы, а затем список имен прочих групп, в которые входит пользователь.

Команда groups без параметров обеспечивает вывод имени и списка групп для текущего пользователя.

11.1.1.5 Id

Утилита **id** выводит реальные и действующие (эффективные) значения UID и GID для текущего пользователя или пользователя, заданного в командной строке.

Синтаксис

id [<опции>]... [<имя пользователя>]
Опции

Таблица 19 Опции команды id

Опция		Значение
-a		Не используется и сохранена только для совместимости со старыми версиями.
-g	**--group**	Выводит только действующий идентификатор группы.
-G	**--groups**	Выводит все идентификаторы групп.
-n	**--name**	Обеспечивает вывод имен взамен числовых значений (для опций **-ugG**).
-r	**--real**	Выводит реальные идентификаторы взамен действующих (для опций **-ugG**).
-u	**--user**	Выводит только действующий идентификатор пользователя.

1 *Сначала указан сокращенный вариант опции, а в скобках приводится полное написание.*

Опция		Значение
	--help	Выводит краткую справку и завершает работу программы.
	--version	Выводит сведения о номере версии и завершает работу программы.

При использовании команды без опций выводится набор сведений для текущего или указанного в команде пользователя.

11.1.1.6 Kill

Команда **kill** служит для прерывания работы указанного в командной строке процесса или группы процессов.

Синтаксис

```
kill [ -s signal | -p ] [ -a ] [ -- ] pid ...
kill -l [ signal ]
```

Команда **kill** передает заданный в командной строке сигнал указанному процессу или группе процессов. Если сигнал не задан в командной строке используется сигнал **TERM**, который будет прерывать процессы, не имеющие ловушки для него. Для некоторых процессов может потребоваться использование сигнала **KILL**, которым нельзя пренебречь.

Большинство современный командных интерпретаторов имеет встроенную функцию **kill**, для управления которой используются опции, подобные описанным ниже[1]. Исключением являются опции **-a** и **-p**, а также возможность указать PID именем команды - эти операции могут не поддерживаться встроенными функциями **kill**.

Опции

Таблица 20 Опции команды kill

Опция	Описание
pid...	Задает список процессов, которые должны быть прерваны. Каждый прерываемый процесс должен быть указан одним из перечисленных ниже способов: **n** - идентификатор процесса, которому должен передаваться сигнал; **0** - сигнал передается всем процессам группы; **-1** - сигнал передается всем процессам с PID > 1; **-n** - сигнал передается всем процессам в группе **n** (**n > 1**). При использовании аргумента **-n** требуется сначала указать передаваемый процессам сигнал или аргумент должен содержать 2 символа дефиса (**--**), поскольку иначе он будет трактоваться как сигнал, который нужно передать. **<имя команды>** - сигнал передается всем процессам, запущенным с использованием указанной команды.
-s <сигнал>	Задает передаваемый процессам сигнал. Для указания сигнала можно использовать имя или номер.
-l	Выводит список имен и номеров сигналов[2].
-a	Отключает ограничение преобразования “имя команды - PID” лишь процессами, для которых идентификатор пользователя (UID) совпадает с идентификатором пользователя для заданного процесса.
-p	Задает для программы необходимость вывода идентификаторов процессов, указанных именем, без передачи им какого-либо сигнала.

11.1.1.7 Md5sum

Утилита **md5sum** используется для проверки аутентичности файлов с помощью цифровых подписей MD5.

Синтаксис

```
md5sum [OPTION] [FILE]...
md5sum [OPTION] --check [FILE]
```

Команда выводит или проверяет 128-битовые контрольные суммы MD5 для указанного файла. Если файл не указан или в качестве имени задан дефис (**-**) входные данные принимаются со стандартного устройства ввода.

11.1.1.7.1 Опции

Таблица 21 Опции команды md5sum

Опция		Значение
-b	**--binary**	Читает файл как бинарный.
-c	**--check**	Сравнивает контрольные суммы с указанными.

1 *При наличии встроенной в командный интерпретатор команды **kill** для запуска внешней утилиты **kill** потребуется полное указание пути к исполняемому файлу (например, **/bin/kill**)*

2 *Вы можете найти этот список в файле **/usr/include/linux/signal.h***

Опция		Значение
-t	**--text**	Читает файл как текстовый. Этот режим используется по умолчанию.
	--status	Отключает вывод информации. Результат проверки определяется как код возврата.
-w	**--warn**	Выдает предупреждения при обнаружении строк контрольных сумм с некорректным форматом.
	--help	Выводит краткую справку о работе с программой.
	--version	Выводит сведения о версии программы.

Контрольные суммы вычисляются и проверяются в соответствии с RFC 1321.

11.1.1.8 Su

Команда **su** позволяет запустить новый командный процессор от имени другого пользователя (например, **root**).

Синтаксис

```
su [<опции>]... [-] [<пользователь>[<аргументы>]...]
```

Таблица 22 Опции команды su

Опция		Описание
-, -l	**--login**	Задает имя пользователя для запуска нового командного процессора.
-c	**--commmand**	Передает указанную параметром команду новому командному процессору.
-f	**--fast**	Передает опцию **-f** новому командному процессору **csh** или **tcsh**.
-m	**--preserve-environment**	Задает сохранение переменных окружения для нового командного процессора.
-p		Синоним опции **-m**.
-s	**--shell**	Задает тип нового командного процессора. Указанный опцией процессор будет использован только в том случае, если его имя присутствует в файле **/etc/shells**.
	--help	Выводит на экран краткую справку и завершает работу программы.
	--version	Выводит на экран информацию о номере версии и завершает работу программы.

Если командная строка не содержит опции **-l** новый командный процессор будет запущен от имени пользователя **root**.

11.1.1.9 Uname

Команда **uname** служит для просмотра системных параметров.

Синтаксис

```
uname [<опции>]...
```

При запуске команды без каких-либо опций на экран выводится имя ядра ОС.

Таблица 23 Опции команды uname

Опция		Описание
-a	**--all**	Выводит весь набор информации в порядке перечисленных далее опций.
-s	**--kernel-name**	Выводит имя ядра ОС.
-n	**--nodename**	Выводит имя хоста.
-r	**--kernel-release**	Выводит номер версии ядра.
-v	**--kernel-version**	Выводит сведения о номере и дате компиляции ядра.
-m	**--machine**	Выводит аппаратное имя машины (архитектура).
-p	**--processor**	Выводит информацию о типе процессора.
-i	**--hardware-platform**	Выводит информацию об аппаратной платформе.
-o	**--operating-system**	Выводит информацию о типе операционной системы.
	--help	Выводит на экран краткую справку и завершает работу программы.
	--version	Выводит на экран информацию о номере версии и завершает работу программы.

11.1.1.10 Who

Команда **who** выводит информацию о зарегистрированных в системе пользователях

Синтаксис

```
who [<опции>]... [ FILE | ARG1 ARG2 ]
```

Таблица 24. Опции команды who.

Опция		Описание
-a	**--all**	Выводит полный набор информации, как при одновременном использовании опций **-b -d --login -p -r -t -T -u**.
-b	**--boot**	Выводит время последней загрузки системы.
-d	**--dead**	Выводит сведения об умерших процессах.
-H	**--heading**	Задает вывод строки заголовка.
-i	**--idle**	Добавляет колонку с информацией о времени бездействия процессов пользователя.
-l	**--login**	Выводит сведения о регистрации пользователей в системе.
	--lookup	Задает определение имен хостов с помощью DNS.
-m		Ограничивает вывод информации только хостами и пользователями, связанными с **stdin**.
-p	**--process**	Показывает все активные процессы, порожденные пользователем.
-q	**--count**	Показывает количество зарегистрированных в системе пользователей и их имена.
-r	**--runlevel**	Показывает текущее значение **runlevel**.
-s	**--short**	Задает сокращенный формат вывода (имя пользователя, терминал и время).
-t	**--time**	Показывает время последней корректировки системных часов.
-T	**--mesg**	Показывает наличие (**+**) или отсутствие (**-**) сообщений для пользователя (как **-w**).
-u	**--users**	Показывает список зарегистрированных в системе пользователей.
-w	**--message** **--writable**	Показывает наличие (**+**) или отсутствие (**-**) сообщений для пользователя (как **-T**).
	--help	Выводит на экран краткую справку и завершает работу программы.
	--version	Выводит на экран информацию о номере версии и завершает работу программы.

Параметр **FILE** задает системный журнал, из которого программа получает информацию (по умолчанию **/var/run/utmp**).

11.1.2 Пакет net-tools

Пакет net-tools включает базовый набор программ, обеспечивающих возможность настройки и мониторинга сети.

11.1.2.1 arp

Команд **arp** позволяет работать с системным кэшем протокола ARP. Структура таблиц и поля SysCtl для протокола ARP описаны в приложении 12.11.

Синтаксис

```
arp [-evn] [-H <тип>] [-i <интерфейс>] -a [<хост>]
arp [-v] [-i <интерфейс>] -d <хост> [pub]
arp [-v] [-H <тип>] [-i <интерфейс>] -s <хост> <адрес> [temp]
arp [-v] [-H <тип>] [-i <интерфейс>] -s <хост> <адрес> [<маска> nm] pub
arp [-v] [-H <тип>] [-i <интерфейс>] -Ds <хост> ifa [<маска> nm] pub
arp [-vnD] [-H <тип>] [-i <интерфейс>] -f [<файл>]
```

Программа **arp** обеспечивает доступ к поддерживаемым ядром таблицам ARP, позволяя администратору добавлять записи в таблицу и удалять из нее существующие записи. Кроме того, программа позволяет вывести полный дамп таблицы из кэша ARP.

Опции arp перечислены в таблице 10. Первая колонка таблицы содержит краткий вариант опции, вторая - полный.

Таблица 25 Опции команды arp

Опция		Действие
-v	**--verbose**	Обеспечивает вывод дополнительной информации.
-n	**--numeric**	Выводит числовые адреса, не пытаясь определить соответствующие им символьные имена.
-H <тип> **-t <тип>**	**--hw-type <тип>**	Задает класс записей кэша ARP, к которым относится команда. По умолчанию используется тип **ether**. Допустимо указание в качестве типа значений **arcnet**, **pronet**, **ax25** (AX.25) и **netrom** (NET/ROM).
-a [<хост>]	**--display [<хост>]**	Показывает записи таблицы для указанного опцией хоста. Если имя хоста не указано, выводятся все записи таблицы. При выводе используется стиль BSD.

Опция		Действие
-d <хост>	**--delete <хост>**	Удаляет из таблицы запись для указанного хоста.
-D	**--use-device**	Задает необходимость использования аппаратного адреса устройства, указанного параметром **ifa**.
-e		Задает вывод таблицы в формате Linux[1].
-i <интерфейс>	**--device <интерфейс>**	Задает интерфейс, к которому относится команда. В командах добавления записей этот параметр указывает интерфейс, с которым будет связана запись (при отсутствии параметра ядро будет выбирать интерфейс в соответствии с таблицей маршрутизации). Для публикуемых (pub) записей указанный параметром интерфейс используется при адресации запросов ARP. Отметим, что этот интерфейс не обязан совпадать с тем, который используется для маршрутизации дейтаграмм IP.
-s <хост> <адрес>	**--set <хост> <адрес>**	Создает ARP-запись для указанной пары “хост-адрес”. При добавлении записей **proxy arp** (т. е., записей с флагом публикации) в старых версиях ядра Linux (до 2.2) можно было указывать маску для записей **proxy arp**, относящихся к подсетям. В более новых версиях ядро Linux взамен поддержки записей для подсетей автоматически создает записи для хостов, подключенных к другим интерфейсам при поддержке таким интерфейсом пересылки пакетов и функций proxy arp.
-f <файл>	**--file <файл>**	Добавляет в таблицу записи подобно опции **-s**, но берет параметры записей из указанного файла. Если файл не указан явно, параметры считываются из файла **/etc/ethers**. Каждая строка файла параметров должна содержать имя или IP-адрес хоста и аппаратный адрес, отделенный пробелами или знаками табуляции. Запись в файле может также содержать флаги **pub**, **temp** и **netmask**.

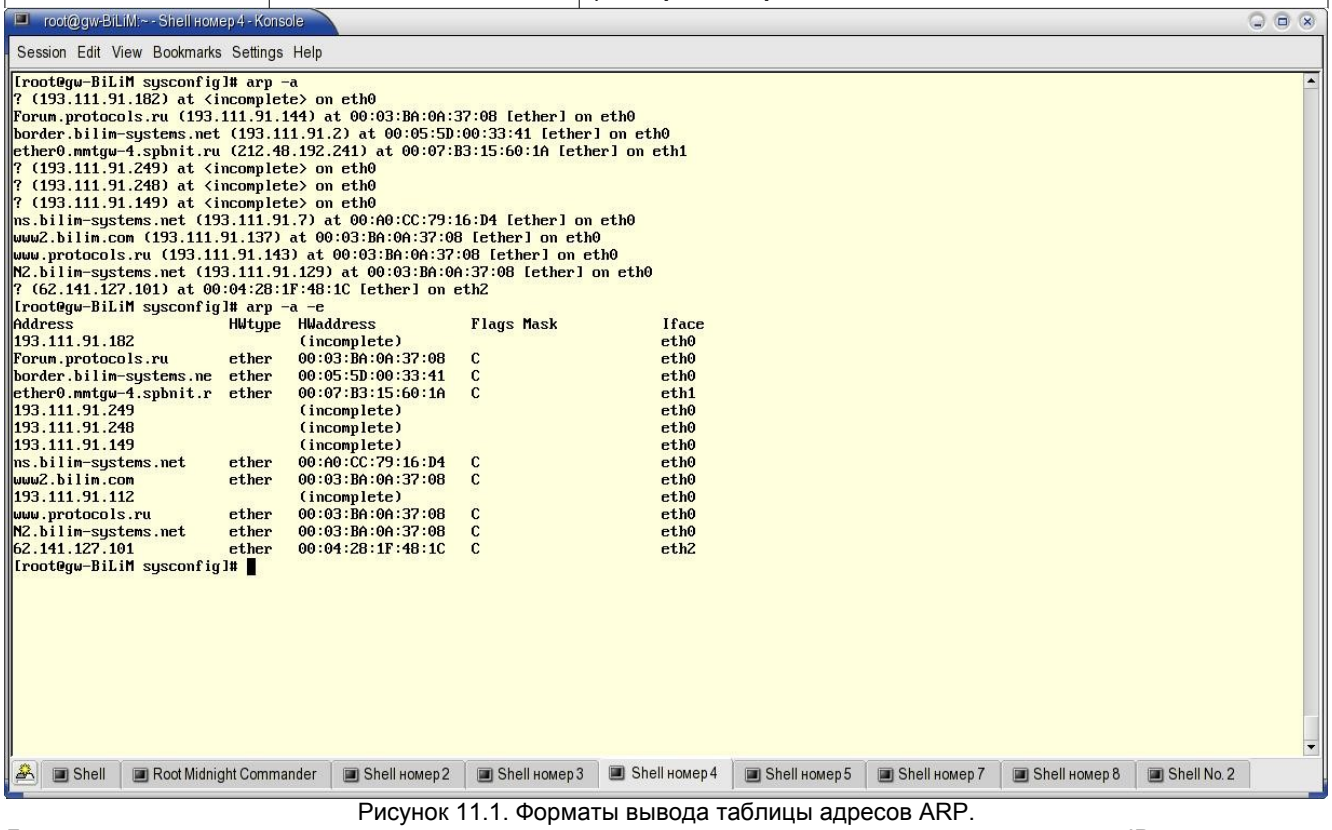


Рисунок 11.1. Форматы вывода таблицы адресов ARP.

Для указания хостов в командной строке могут использоваться символьные имена или адреса IP в стандартной записи с разделением точками.

Для совместимости со старыми версиями поддерживается возможность менять местами сетевой и аппаратный адреса.

Каждая полная запись таблицы ARP помечается флагом **C**, для постоянных записей используется флаг **M**, и для публикуемых - **P**.

1 *Как можно видеть на рисунке 11.1 этот формат удобней для восприятия, нежели используемый по умолчанию формат BSD.*

11.1.2.2 Hostname

hostname - устанавливает или определяет имя локального хоста;

domainname - устанавливает или определяет доменное имя (NIS/YP) для локального хоста;

dnsdomainname - выводит доменное имя (DNS) для локального хоста;

nisdomainname - устанавливает или определяет доменное имя (NIS/YP) для локального хоста;

ypdomainname - устанавливает или определяет доменное имя (NIS/YP) для локального хоста.

Синтаксис

```
hostname  [-v] [-a] [--alias] [-d] [--domain] [-f] [--fqdn] [-i] [--ip-address] [--long]
[-s] [--short] [-y] [--yp] [--nis] [-n] [--node]
hostname [-v] [-F filename] [--file filename] [hostname]
domainname [-v] [-F filename] [--file filename] [name]
nodename [-v] [-F filename] [--file filename] [name]
hostname [-v] [-h] [--help] [-V] [--version]
dnsdomainname [-v]
nisdomainname [-v]
ypdomainname [-v]
```

Команда **hostname** служит для установки или определения текущего имени хоста или доменного имени. Имена используются многими сетевыми программами для идентификации хостов. Доменные имена также используются системой NIS/YP.

11.1.2.2.1 Определение имени

При вводе команды без параметров на экран выводятся текущие имена:

hostname будет выводить имя системы, возвращаемое функцией **gethostname;**

domainname, **nisdomainname** и **ypdomainname** будут выводить имя системы, возвращаемое функцией **getdomainname**. Это имя называют также доменным именем YP/NIS;

dnsdomainname будет выводить доменную часть полного доменного имени **FQDN**[1]. Для вывода полного имени FQDN (а не только доменной части) можно воспользоваться командой **hostname --fqdn**.

11.1.2.2.2 Установка имени

При вызове с одним аргументом или опцией **--file** команда будет устанавливать имя хоста, доменное имя NIS/YP или имя узла. Право изменения имен имеет только пользователь с UID=0.

С помощью команды **dnsdomainname** невозможно установить FQDN или доменное имя DNS (см. параграф 11.1.2.2.2.1).

Имя хоста обычно устанавливается при загрузке системы в процессе выполнения сценария **/etc/rc.d/rc.inet1** или **/etc/init.d/boot**. Обычно имя хоста считывается из файла (например, **/etc/hostname**).

11.1.2.2.2.1 Установка FQDN

Вы не можете изменить FQDN (имя, возвращаемое командой **hostname --fqdn**) или доменное имя DNS (возвращается по команде **dnsdomainname**) с помощью данной команды. FQDN - это имя системы, возвращаемое функцией **resolver**.

Технически FQDN представляет собой имя, возвращаемое функцией **gethostbyname**, получившей в качестве параметра имя, возвращенное функцией **gethostname**. Доменным именем DNS называют часть FQDN справа от первой точки.

Следовательно, для изменения имени нужно отредактировать конфигурационный файл (обычно, **/etc/host.conf**)[2].

11.1.2.2.3 Опции

Опция		Значение
`-a`	`--alias`	Выводит псевдоним имени хоста, если таковой существует.
`-d`	`--domain`	Выводит имя домена DNS. Не используйте для получения этого имени команду **domainname**, поскольку она будет выводить доменное имя NIS, а не DNS. Для вывода доменного имени DNS можно воспользоваться командой **dnsdomainname**.
`-F <имя>`	`--file <имя>`	Читает имя хоста из указанного файла. Строки комментариев (#) в этом файле игнорируются командой.
`-f`	`--fqdn` `--long`	Выводит полное имя FQDN, состоящее из имени хоста и доменного имени DNS. Если вы не используете для определения имен сервер DNS (например, bind) или NIS, вы можете изменить доменное имя и FQDN, отредактировав файл **/etc/hosts**[3].

1 *Fully Qualified Domain Name*
2 *Не забывайте в таких случаях изменить соответствующие записи серверов DNS и NIS/YP.*
3 *В этом случае потребуется использовать команду **/etc/init.d/network restart** для активизации нового имени.*

Опция		Значение
-h	**--help**	Выводит краткую справку и завершает работу программы.
-i	**--ip-address**	Выводит IP-адрес хоста.
-s	**--short**	Выводит короткое имя хоста (до первой точки в доменном имени).
-V	**--version**	Выводит информацию о номере версии и завершает работу программы.
-v	**--verbose**	Задает подробный вывод информации.
-y	**--yp** **--nis**	Выводит доменное имя NIS. Эта опция с дополнительным параметром или опцией **--file** позволяют пользователю с UID=0 изменить доменное имя NIS.

11.1.2.3 Ifconfig

Программа **ifconfig** обеспечивает возможность настройки и контроля параметров сетевых интерфейсов UNIX, поддерживаемых ядром системы. Во время загрузки ОС программа устанавливает параметры, требуемые для работы каждого активного интерфейса, а после загрузки может служить для мониторинга интерфейса или изменения его режима.

Синтаксис

```
ifconfig [<интерфейс>]
ifconfig <интерфейс> [<семейство адресов>] <опции> | <адрес> ...
```

При запуске команды **ifconfig** без параметров на экран выводится информация о всех активных интерфейсах системы. Если в командной строке задано только имя одного интерфейса, команда выведет на экран сведения о состоянии указанного интерфейса. При запуске команды с ключом **-a** обеспечивается вывод сведений для всех присутствующих в системе интерфейсов, независимо от их активности. Все остальные варианты параметров командной строки служат для настройки конфигурационных параметров интерфейса.

Если параметр, следующий за именем интерфейса, распознается программой как имя поддерживаемого семейства адресов сетевого уровня, для декодирования и вывода протокольных адресов программа будет использовать формат указанного семейства. В настоящее время программа распознает следующие семейства адресов:

- **inet** (TCP/IP);
- **inet6** (IPv6);
- **ax25** (AMPR Packet Radio);
- **ddp** (Appletalk Phase 2);
- **ipx** (Novell IPX);
- **netrom** (AMPR Packet radio).

Если семейство адресов не задано, по умолчанию будет предполагаться принадлежность к семейству **inet**.

Опции, передаваемые в командной строке **ifconfig**, перечислены в таблице .

Таблица 26. Опции команды ifconfig.

Опция	Значение
<интерфейс>	Указывает имя интерфейса, которое обычно состоит из имени базового драйвера и порядкового номера устройства. Например, первый интерфейс Ethernet в системах Linux обычно имеет имя **eth0**.
up	Флаг активизации интерфейса[2].
down	Флаг дезактивации (отключения) интерфейса.
[-]arp	Служит для разрешения или запрета поддержки протокола ARP для указанного в команде интерфейса.
[-]promisc	Включает или отключает для интерфейса режим захвата пакетов (promiscuous[3]).
[-]allmulti	Включает или отключает режим **all-multicast**[4].
metric N	Задает метрику для интерфейса.
mtu N	Задает размер максимального кадра (MTU[5]) для данного интерфейса.
dstaddr addr	Задает IP-адрес удаленного партнера для соединений “точка-точка” (например, PPP). Эту опцию не рекомендуется использовать - взамен ее служит опция **pointopoint**, описанная ниже.

2 *Этот флаг будет использоваться неявно, если команда включает адрес интерфейса.*
3 *В режиме захвата интерфейс будет считывать из сетевой среды все кадры, а в обычном - только те кадры, которые ему адресованы (включая широковещательные и групповые).*
4 *При включенном режиме **all-multicast** интерфейс будет считывать из среды все кадры с групповыми адресами.*
5 *Maximum Transfer Unit - максимальный блок передачи данных.*

Опция	Значение
netmask addr	Задает IP-маску подсети для данного интерфейса. По умолчанию устанавливается маска класса A, B или C в зависимости от заданного для интерфейса адреса IP.
add addr/prefixlen	Добавляет адрес IPv6 для указанного интерфейса.
del addr/prefixlen	Удаляет адрес IPv6 для указанного интерфейса.
tunnel aa.bb.cc.dd	Создает новое устройство SIT (IPv6-in-IPv4) для туннелирования в указанную сеть.
irq addr	Задает аппаратное прерывание (IRQ) для устройства[6].
io_addr addr	Задает базовый адрес ввода-вывода (I/O base) для устройства.
mem_start addr	Задает стартовый адрес разделяемой памяти, используемой устройством[7].
media type	Задает физический порт или тип среды передачи, используемый устройством[8].
[-]broadcast [addr]	Устанавливает для устройства широковещательный адрес сетевого уровня или устанавливает (сбрасывает) для интерфейса флаг **IFF_BROADCAST**, если команда не включает широковещательного адреса.
[-]pointopoint [addr]	Разрешает или запрещает для интерфейса работу в режиме “точка-точка”, при котором канал связи разделяют только два устройства. Если в команде задан адрес, он используется в качестве адреса сетевого уровня для удаленной стороны соединения. При отсутствии адреса команда просто устанавливает или сбрасывает для интерфейса флаг **IFF_POINTOPOINT**.
hw class address	Задает аппаратный адрес интерфейса, если физическое устройство поддерживает такую возможность. В команде требуется присутствие идентификатора класса аппаратного адреса (**ether** - Ethernet, **ax25** - AMPR AX.25, **ARCnet** или **netrom** - AMPR NET/ROM).
multicast	Устанавливает для интерфейса флаг **multicast**[9].
address	Задает для интерфейса адрес IP.
txqueuelen length	Задает для устройства размер очереди на передачу. Для медленных устройств с высокой задержкой (например, модемов) имеет смысл устанавливать небольшое значение размера очереди, чтобы предотвратить конфликты между передачей больших объемов данных (копирование файлов) и работой интерактивных приложений (например, telnet или ssh).

Пример вывода команды **ifconfig** показан на рисунке 11.2.

6 *Возможность изменения IRQ поддерживается не для всех устройств.*
7 *Эту опцию поддерживают достаточно редкие устройства.*
8 *Не все устройства поддерживают возможность выбора типа среды на уровне конфигурационных параметров. Для многих старых устройств может потребоваться программа настройки или установка перемычек/переключателей на плате.*
9 *Обычно этот флаг устанавливается multicast-приложениями и задавать его явно не требуется.*

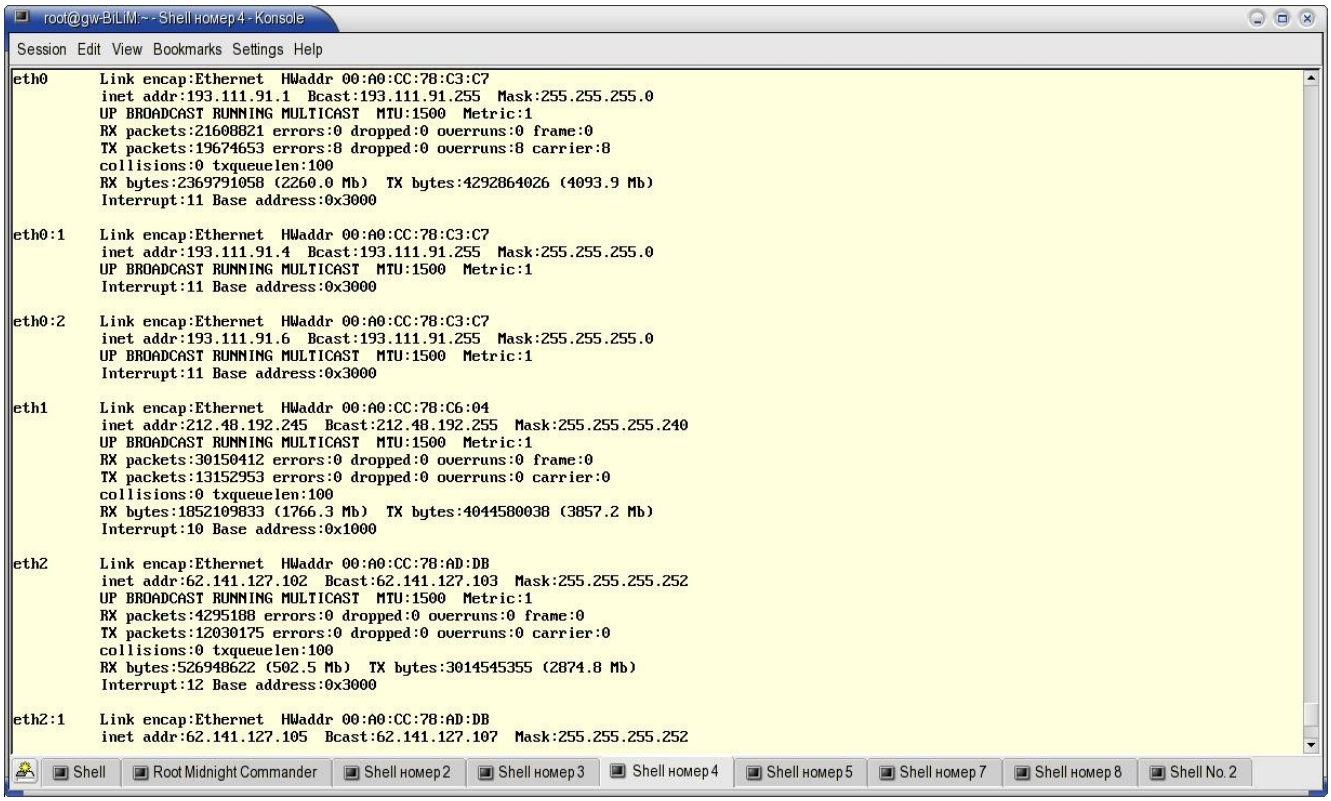


Рисунок 11.2 Вывод команды ifconfig

11.1.2.4 Mii-tool

Команда **mii-tool** позволяет просматривать и изменять физические параметры сетевого интерфейса MII (media-independent interface). Большинство адаптеров Fast Ethernet использует MII для согласования скорости и режима работы.

Синтаксис

```
mii-tool [-v, --verbose] [-V, --version] [-R, --reset] [-r, --restart] [-w, --watch] [-
l, --log] [-A, --advertise=media,...] [-F, --force=media] [interface ...]
```

Большинство современных сетевых устройств использует те или иные механизмы согласования рабочих параметров для определения типа сетевой среды и возможностей передачи данных. По результатам определения подключенных к среде устройств выбирается наиболее эффективный вариант из числа поддерживаемых всеми подключенными к разделяемой среде устройствами. Опция **-A** (**--advertise**) позволяет задать интерфейсу MII анонсирование только части поддерживаемых им возможностей (это может быть полезно в тех случаях, когда в сети присутствуют устройства, не поддерживающие автоматическое согласование параметров). Протокол MII также позволяет явно задать тип соединения (**10baseT** или **100baseT**). Опция **-F** (--force) может использоваться для перевода MII в заданный режим без использования функций согласования параметров. Отметим, что опции **-A** и **-F** не могут использоваться совместно.

Используемый по умолчанию сжатый формат вывода показывает согласованную скорость и состояние соединения для каждого интерфейса.

```
# mii-tool
eth0: negotiated 100baseTx-FD flow-control, link ok
eth1: no autonegotiation, 10baseT-HD, link ok
eth2: no autonegotiation, 10baseT-HD, link ok
```

Если в командной строке не указано имя интерфейса, **mii-tool** будет проверять все интерфейсы с именами от **eth0** до **eth7**.

11.1.2.4.1 Опции

Таблица 27. Опции команды mii-tool.

Опция		Описание
-v	**--verbose**	Обеспечивает более детализированный вывод информации. При использовании опции дважды (**-vv**) обеспечивается также вывод содержимого регистров MII.
-V	**--version**	Выводит информацию о номере версии программы.
-R	**--reset**	Сбрасывает MII в исходное состояние.
-r	**--restart**	Активизирует процедуру согласования параметров MII.
-w	**--watch**	Включает режим отслеживания состояния интерфейса и оповещения о смене состояния соединения. Опрос интерфейсов MII производится один раз в секунду.

Опция		Описание
-l	**--log**	При использовании вместе с опцией **-w** обеспечивает запись сведений о смене состояния в журнальный файл системы вместо вывода на **stdout**.
-F <среда>	**--force=<среда>**	Отключает автоматическое согласование параметров и переводит MII в режим работы с указанной в команде средой (**100baseT4, 100baseTx-FD**, **100baseTx-HD**, **10baseT-FD** или **10baseT-HD**)
-A <среда>	**--advertise=<среда>**	Включает и инициирует режим автоматического определения среды с анонсированием только заданного типа. Команда может содержать несколько идентификаторов сред, разделенных запятыми.

11.1.2.5 netstat

Программа netstat, входящая в состав большинства дистрибутивов UNIX (и даже Windows) позволяет просматривать информацию о состоянии сетевой подсистемы хоста или маршрутизатора. Тип выводимой информации определяется опциями командной строки[1].

При отсутствии каких либо опций команда

netstat

выводит список открытых сокетов. Если не задано никакого семейства адресов, будет выводиться список активных сокетов для всех имеющихся адресов.

11.1.2.5.1 Опции типа информации

--route (-r)

выводит таблицу маршрутизации ядра Linux.

--groups (-g)

выводит информацию о принадлежности к multicast-группам для протоколов IPv4 и IPv6.

--interface=[имя интерфейса] (-i)

выводит статистическую информацию для указанного интерфейса. Если интерфейс не задан, выводятся сведения для всех имеющихся в системе интерфейсов.

--masquerade (-M)

выводит список маскируемых соединений.

--statistics (-s)

выводит статистическую информацию для каждого протокола.

11.1.2.5.2 Опции вывода

--verbose (-v)

задает максимальный объем вывода информации (включая полезные сведения о несконфигурированных семействах адресов).

--numeric (-n)

задает вывод числовых значений взамен попытки определения символьных имен хостов, портов и пользователей.

--numeric-hosts

задает вывод адресов хостов без попыток определения имен; опция не влияет на вывод имен портов и пользователей.

--numeric-ports

задает вывод номеров портов без попыток определения символьных имен; на вывод имен пользователей и хостов эта опция не влияет.

--numeric-users

задает вывод значений UID без попыток определения имени пользователя; на определение имен хостов и портов эта опция не оказывает влияния.

--protocol=<список протоколов> (-A)

задает семейство протоколов, для которых выводится информация о соединениях. Идентификаторы семейств (inet, unix, ipx, ax25, netrom, ddp) разделяются запятыми. Для каждого семейства протоколов существует также отдельная опция --inet, --unix (-x), --ipx, --ax25, --netrom, --ddp.

Протоколы семейства inet включают сокеты raw, udp и tcp.

-c (--continuous)

задает ежесекундное повторение команды netstat с выбранными опциями, для непрерывного мониторинга.

-e (--extend)

задает вывод дополнительной информации, если таковая имеется. Допускается двукратное использование опции для дополнительного расширения выводимых сведений.

-o (--timers)

задает вывод информации о сетевых таймерах.

1 *При описании опций в круглых скобках указаны сокращенные варианты записи.*

-p (--program)
задает вывод PID и имени программы, с которой связан каждый сокет.

-l (--listening)
задает вывод только списка сокетов, находящихся в состоянии прослушивания (LISTENING). По умолчанию этот список не выводится.

-a (--all)
задает вывод списка всех сокетов.

-F
выводит маршрутную информацию из FIB[1]. Этот флаг включен по умолчанию.

-C
Выводит маршрутную информацию из кэша.

delay
задает для программы netstat вывод статистики каждые **delay** секунд. Например, команда

netstat 5
будет обновлять информацию каждые 5 секунд.

11.1.2.5.3 Формат вывода

11.1.2.5.3.1 Активные соединения Internet (TCP, UDP, raw)

Информация об активных соединениях выводится в виде нескольких колонок, перечисленных в таблице 28.

Таблица 28 Поля вывода информации об активных соединениях

Поле	Содержимое
Proto	Протокол, используемый сокетом (tcp, udp, raw)
Recv-Q	Счетчик байтов для данных, еще не скопированных пользовательскими программами, подключенными к сокетам.
Send-Q	Счетчик байтов, прием которых еще не подтвержден удаленным хостом.
Local Address	Хост и порт с локальной стороны соединения. Если в команде не была указана опция **-n**, для хоста и порта указываются символьные имена[2], в противном случае - IP-адрес и номер порта
Foreign Address	Удаленный хост и порт, указываемые по таким же принципам, которые применяются для локальной стороны.

1 *Forwarding Information Base - база информации о пересылке пакетов.*
2 *Для хоста указывается каноническое имя FQDN.*

Поле	Содержимое
State[1]	Состояние соединения: **ESTABLISHED** соединение организовано; **SYN_SENT** сокет предпринимает попытки организации соединения (передан запрос SYN); **SYN_RECV** получен запрос из сети на организацию соединения; **FIN_WAIT1** сокет закрыт, происходят процедуры разрыва соединения; **FIN_WAIT2** соединений закрыто и сокет ждет подтверждения закрытия от удаленной стороны; **TIME_WAIT** сокет после закрытия соединения ожидает прибытия оставшихся пакетов из сети; **CLOSED** сокет не используется; **CLOSE_WAIT** удаленная сторона закрыла соединение, ожидается закрытие сокета; **LAST_ACK** удаленная сторона закрыла соединение и сокет закрыт, ожидается подтверждение; **LISTEN** сокет прослушивает входящие соединения. Такие сокеты не указываются в списке, если команда не содержит опции **-l** или **-a**; **CLOSING** Оба сокета (локальный и удаленный закрыты, но локальная сторона еще имеет непереданную информацию; **UNKNOWN** состояние сокета не удается определить.
User	Имя или идентификатор (UID) пользователя, владеющего сокетом.
PID/Program name	Идентификатор/имя процесса, владеющего сокетом. Для вывода этой колонки в командной строке должна присутствовать опция --**program**. Для просмотра списка чужих процессов требуются права администратора (root).
Timer	В настоящее время не поддерживается.

11.1.2.5.3.2 Активные сокеты UNIX domain

Информация об активных сокетах UNIX-домена выводится в несколько колонок, перечисленных в таблице).

Таблица 29 Поля информации о сокетах UNIX domain

Поле	Содержимое
Proto	Используемый сокетом протокол (обычно unix).
RefCnt	Счетчик ссылок (число процессов, подключенных через данный сокет).
Flags	Поле флаг выводится, при наличии флага SO_ACCEPTON (выводится как ACC), SO_WAITDATA (W) или SO_NOSPACE (N). Флаг SO_ACCECPTON устанавливается для неподключенных сокетов, если подключенный к сокету процесс ожидает запроса на соединение. Остальные флаги обычно не представляют интереса.

1 *Для пакетов raw (обычно и для UDP) поле State не содержит никакой информации о состоянии.*

Поле	Содержимое
Type	Тип сокета: **SOCK_DGRAM** сокет используется в режиме дейтаграмм (connectionless - без организации соединений) без сохранения порядка и гарантии доставки. Обычно для таких сокетов устанавливается максимальный размер дейтаграммы. **SOCK_STREAM** сокет, поддерживающий упорядоченные потоки с гарантированной доставкой в обоих направлениях на основе организации явных соединений. **SOCK_RAW** raw-сокет. **SOCK_RDM** сокет, используемый для передачи дейтаграмм с гарантированной доставкой, но без гарантии соблюдения порядка. **SOCK_SEQPACKET** сокет, поддерживающий упорядоченные и надежные[2] двухсторонние пути передачи дейтаграмм с фиксированным максимальным размером. Потребитель данных должен считывать при каждом обращении к сокету пакет целиком. **SOCK_PACKET** сокет доступа с raw-интерфейсом. Этот тип считается устаревшим и не должен использоваться в новых программах. **UNKNOWN** Тип сокета не удалось определить.
State	Состояние сокета: **FREE** сокет не выделен. **LISTENING** сокет прослушивает запросы на организацию соединений. По умолчанию такие сокеты не показываются и для их просмотра нужно указать в командной строке опцию -**listening** (**-l**) или **--all** (**-a**). **CONNECTING** сокет находится в стадии организации соединения. **CONNECTED** сокет соединен. **DISCONNECTING** сокет находится в стадии разрыва соединения. **(пустое поле)** сокет не соединен с другим сокетом. **UNKNOWN** состояние сокета неизвестно. Такого не должно происходить.
I-Node	
Path	Путь к файлу, через который процесс соединен с сокетом.

2 *С гарантированной доставкой.*

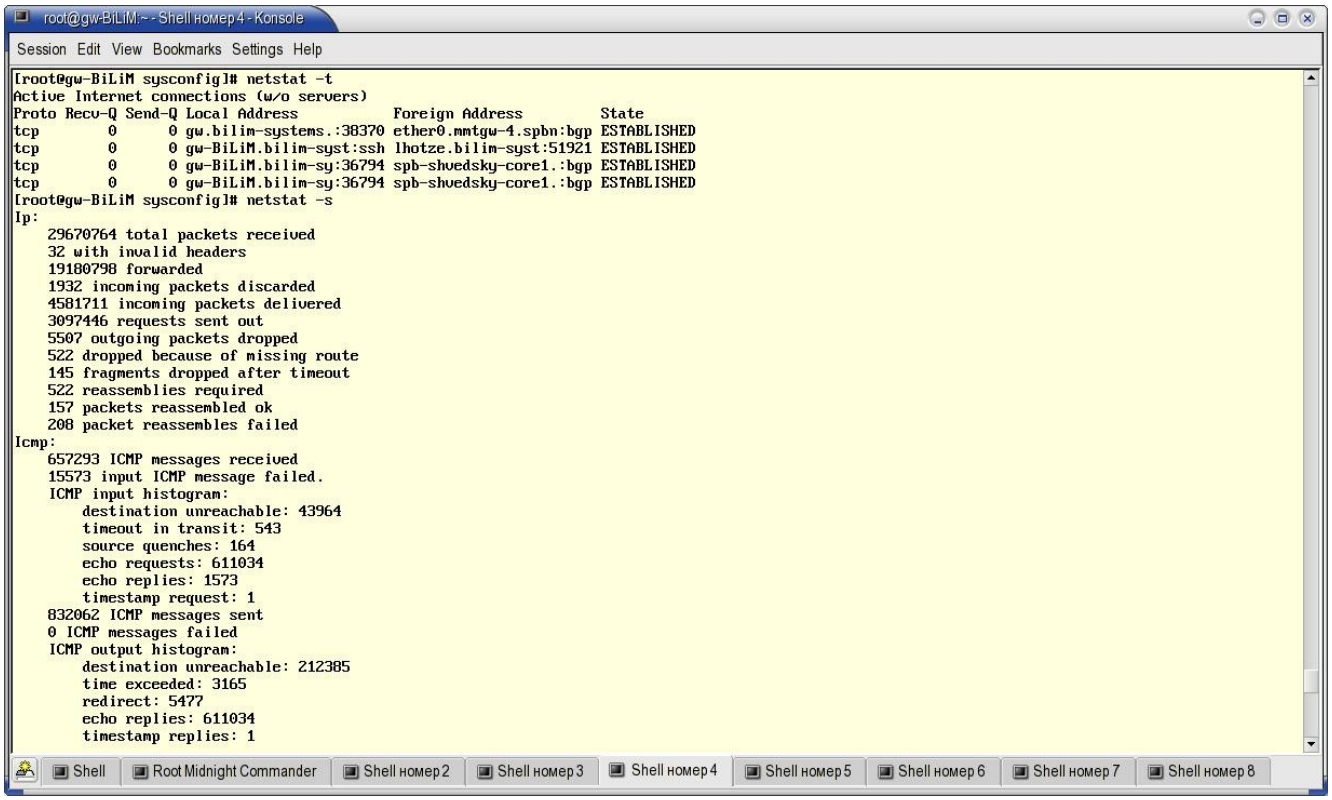


Рисунок 11.3. Информация, полученная с помощью команды netstat.

11.1.2.6 Route

Утилита **route** служит для просмотра и управления таблицей маршрутизации IP.

Синтаксис

```
route [-CFvnee]
route [-v] [-A  family] add [-net|-host] target [netmask Nm] [gw Gw] [metric N] [mss M]
[window W] [irtt I] [reject] [mod] [dyn] [reinstate] [[dev] If]
route [-v] [-A  family] del [-net|-host] target [gw Gw] [netmask Nm] [metric N] [[dev]
If]
route [-V] [--version] [-h] [--help]
```

Команда **route** работает с таблицами маршрутизации IP ядра Linux. Основным вариантом использования программы является создание и удаление статических маршрутов к хостам и сетям через интерфейсы, настроенные с помощью программы **ifconfig** (см. параграф 11.1.2.3 на стр. 196).

При использовании с опцией **add** или **del** команда изменяет таблицу маршрутизации. Во всех прочих случаях команда обеспечивает вывод информации о текущем состоянии таблицы маршрутов.

11.1.2.6.1 Опции

Опция	Описание
-A <семейство адресов>	Задает использование команды для указанного семейства адресов (например, **inet**). Список семейств адресов можно получить с помощью опции **--help**.
-F	Задает работу с таблицами маршрутизации ядра FIB[1]. Этот режим используется по умолчанию.
-C	Задает работу программы с маршрутным кэшем ядра.
-v	Повышает уровень информативности вывода из программы.
-n	Задает числовое представление адресов взамен символьных имен. Такой режим полезен при работе без доступа к серверу DNS.
-e	Задает использование формата **netstat -r** (параграф 11.1.2.5 на стр. 199) для вывода таблицы маршрутов. Опция **-ee** будет генерировать длинные строки, содержащие все параметры из таблицы маршрутизации.
del	Удаляет маршрут из таблицы
add	Добавляет маршрут в таблицу.
target	Хост или сеть в которую ведет маршрут. Указать сеть или хост можно по имени или адресу IP.
-net	Конечной точкой маршрута является сеть.

1 *Forwarding Information Base - таблица пересылки пакетов.*

Опция	Описание
-host	Конечной точкой маршрута является хост.
netmask <маска>	Маска, используемая при задании маршрута в сеть
gw <шлюз>	Задает маршрутизацию пакетов через указанный шлюз[1].
metric <метрика>	Задает значение поля метрики в таблице маршрутизации, используемое демонами маршрутизации.
mss <размер>	Задает максимальный размер сегмента TCP (MSS) для соединений, использующих данный маршрут. По умолчанию максимальный размер сегмента вычисляется как значение MTU для интерфейса за вычетом заголовков или минимальное значение MTU для пути доставки. Этот параметр можно использовать для дробления пакетов на одной стороне соединения TCP в тех случаях, когда MTU Discovery не работает (обычно это бывает из некорректных настроек межсетевых экранов, блокирующих сообщения ICMP Fragmentation Needed)
window <размер>	Задает размер (в байтах) окна для соединений TCP, использующих данный маршрут. Обычно этот параметр используется только для сетей AX.25 или с драйверами, неспособными обеспечивать сквозное обслуживание кадров.
irtt <значение>	Задает начальное значение времени кругового обхода (**irtt**) для соединений TCP через данный маршрут. Время указывается в миллисекундах (1 - 12000). Обычно этот параметр используется только для сетей AX.25. Принятое по умолчанию значение **irtt** составляет 300 мсек, в соответствии со стандартом RFC 1122.
reject	Создает блокирующий маршрут, который будет приводить к отказам при попытке просмотра данного маршрута. Эта опция может служить для маскирования некоторых маршрутов и использования взамен принятого по умолчанию шлюза. Не следует использовать эту опцию на межсетевых экранах.
mod, dyn, reinstate	Устанавливает динамический (обновляемый) маршрут. Эти флаги поддерживаются командой для диагностики, а соответствующие флаги маршрутов обычно устанавливает демон маршрутизации.
dev <интерфейс>	Связывает маршрут с указанным интерфейсом, предотвращая попытки ядра определить интерфейс самостоятельно. Для большинства сетей использование этой опции совершенно не требуется. Если эта опция задана последней в строке, ключевое слово **dev** можно опустить.

11.1.2.6.2 Примеры использования

route add -net 127.0.0.0
добавляет маршрут в loopback-сеть с принятой по умолчанию для класса A маской 255.0.0.0 и связывает этот маршрут с интерфейсом **lo** (предполагается, что этот интерфейс активизирован с помощью команды ifconfig).

route add -net 192.56.76.0 netmask 255.255.255.0 dev eth0
добавляет маршрут в сеть 192.56.76.x через интерфейс **eth0**. Указание маски в команде не является обязательным, поскольку сеть относится к классу C и выбранная маска соответствует тому же классу. Ключевое слово **dev** также можно опустить, поскольку оно используется в конце командной строки.

route add default gw mango-gw
добавляет используемый по умолчанию маршрут. Все пакеты будут передаваться на этот маршрут через шлюз **mango-gw**. Локальное устройство, которое будет использоваться для передачи пакетов, определяет параметрами статического маршрута к шлюзу **mango-gw**, который должен быть организован заранее.

route add ipx4 sl0
добавляет маршрут к хосту **ipx4** через последовательный (SLIP) интерфейс **sl0** (предполагается, что хост **ipx4** также имеет последовательный интерфейс).

route add -net 192.57.66.0 netmask 255.255.255.0 gw ipx4
добавляет маршрут в сеть 192.57.66.x через шлюз **IPX4**.

route add -net 224.0.0.0 netmask 240.0.0.0 dev eth0
организует все групповые (multicast) маршруты IP через интерфейс **eth0**.

route add -net 10.0.0.0 netmask 255.0.0.0 reject
устанавливает блокирующий маршрут для приватной сети 10.x.x.x.

11.1.2.6.3 Формат вывода

Информация из таблицы маршрутов выводится на экран в несколько колонок, описанных в таблице 30.

1 *Указанный шлюз должен быть доступен с данного хоста. Обычно для этого требуется сначала указать статический маршрут к шлюзу. Если вы укажете какой-либо из локальных интерфейсов, этот интерфейс будет использоваться для принятия решения о маршрутизации пакета (тяжкое наследие BSD).*

Таблица 30. Поля вывода команды route.

Название	Содержание
Destination	Сеть или хост на другой стороне маршрута.
Gateway	Адрес шлюза или значок *****, если адреса нет.
Genmask	Маска сети для адресата маршрута (255.255.255.255 для хоста, 0.0.0.0 для используемого по умолчанию маршрута).
Flags	Поле флагов маршрута, которое может включать **U** - активный маршрут; **H** - маршрут к хосту; **G** - маршрут с использованием шлюза; **R** - динамический маршрут (reinstate route); **D** - динамический маршрут от демона или **redirect**; **M** - измененный маршрут от демона или **redirect**; **A** - маршрут добавлен **addrconf** **C** - запись из кэша; **!** - блокирующий маршрут.
Metric	“Дистанция” до конечной точки маршрута (обычно измеряется числом интервалов - хопов). Метрика не используется в последних версиях ядра, но может потребоваться для работы демонов.
Ref	Число ссылок на данный маршрут. Это поле не используется ядром Linux.
Use	Число просмотров (lookup) данного маршрута. В зависимости от наличия опций **-F** и **-C** показывает количество случаев обнаружения (**-C**) или отсутствия (**-F**) маршрута в кэше ядра.
Iface	Интерфейс, через который передаются пакеты для данного маршрута.
MSS	Максимальный размер сегмента TCP для соединений через данный маршрут.
Window	Используемый по умолчанию размер окна для соединений TCP с использованием данного маршрута.
irtt	Начальное значение времени кругового обхода (RTT) для данного маршрута. Ядро использует это значение для приблизительной оценки параметров TCP без ожидания ответа.
HH	Это поле выводится только для кэшированных записей и указывает записи ARP и кэшированные маршруты, которые указывают на кэш заголовков с аппаратными адресами. Если для кэширования маршрута не требуется аппаратного адреса[2], поле будет содержать значение -1
Arp	Это поле выводится только для кэшированных записей и показывает актуальность аппаратного адреса для кэшированного маршрута.

2 *Например, для интерфейса **lo**.*

11.2 Дополнительные утилиты

В этом приложении описаны некоторые полезные программы, не вошедшие в пакеты **coreutis** и **net-utils**.

11.2.1 dmesg

Команда **dmesg** позволяет просматривать текущее содержимое кольцевого буфера сообщений ядра.

Синтаксис

```
dmesg [-c] [-n <уровень>] [-s <размер>]
```

Опции

Таблица 31. Опции команды dmesg.

Опция	Описание
-c	Задает очистку буфера сообщений ядра после вывода его содержимого.
-s	Задает размер буфера для запроса данных из кольцевого буфера сообщений ядра. По умолчанию используется буфер размером 16392 байта. Если при компиляции ядра вы задали больший размер буфера сообщений, эта опция позволит вам считывать содержимое буфера ядра полностью.
-n	Устанавливает уровень сообщений, выводимых программой на консоль. Например, **-n 1** будет выводить только сообщения о критических ошибках ядра (**panic**).

При совместном использовании опций **-s** и **-n** будет приниматься во внимание только та, которая указана в командной строке последней.

11.2.2 Ethtool

http://sourceforge.net/projects/gkernel/

Утилита **ethtool** позволяет проверять и менять параметры конфигурации сетевых адаптеров Ethernet.

Синтаксис

```
ethtool ethX
ethtool -h
ethtool -a ethX
ethtool -A ethX [autoneg on|off] [rx on|off] [tx on|off]
ethtool -c ethX
ethtool -C ethX [adaptive-rx on|off] [adaptive-tx on|off] [rx-usecs N] [rx-frames N]
[rx-usecs-irq N] [rx-frames-irq N] [tx-usecs N] [tx-frames N] [tx-usecs-irq N]
[tx-frames-irq N] [stats-block-usecs N] [pkt-rate-low N] [rx-usecs-low N] [rx-frames-low
N] [tx-usecs-low N] [tx-frames-low N] [pkt-rate-high N] [rx-usecs-high N] [rx-frames-high
N] [tx-usecs-high N] [tx-frames-high N] [sample-interval N]
ethtool -g ethX
ethtool -G ethX [rx N] [rx-mini N] [rx-jumbo N] [tx N]
ethtool -i ethX
ethtool -d ethX
ethtool -e ethX [raw on|off] [offset N] [length N]
ethtool -E ethX [magic N] [offset N] [value N]
ethtool -k ethX
ethtool -K ethX [rx on|off] [tx on|off] [sg on|off] [tso on|off]
ethtool -p ethX [N]
ethtool -r ethX
ethtool -S ethX
ethtool -t ethX [offline|online]
ethtool -s ethX [speed 10|100|1000] [duplex half|full] [port tp|aui|bnc|mii]
[autoneg on|off] [phyad N] [xcvr internal|external] [wol p|u|m|b|a|g|s|d...]
[sopass xx:yy:zz:aa:bb:cc] [msglvl N]
```

11.2.2.1 Опции

Команда **ethtool** с единственным аргументом, задающим устройство, служит для просмотра текущих параметров указанного устройства. В остальных случаях работа команды определяет заданными опциями (таблица 32).

Таблица 32. Опции команды ethtool.

Опция или параметр	Значение
-h	Выводит краткую справку.
-a	Запрашивает для устройства информацию о параметрах паузы.
-A	Изменяет параметры паузы для указанного устройства.
autoneg on|off	Управляет режимом автоматического согласования параметров среды.
rx on|off	Управляет поддержкой паузы для приема.

Опция или параметр	Значение
tx on|off	Управляет поддержкой паузы для передачи.
-c	Запрашивает для указанного устройства параметры **coalescing**.
-C	Задает для указанного устройства параметры **coalescing**.
-g	Запрашивает для указанного устройства параметры **rx/tx ring**.
-G	Задает для указанного устройства параметры **rx/tx ring**.
rx N	Изменяет число записей для **Rx ring**.
rx-mini N	Изменяет число записей для **Rx Mini ring**.
rx-jumbo N	Изменяет число записей для **Rx Jumbo ring**.
tx N	Изменяет число записей для **Tx ring**.
-i	Запрашивает для указанного устройства информацию о связанном с ним драйвере.
-d	Выводит дамп регистров для указанного устройства.
-e	Выводит дамп EEPROM для указанного устройства. Если возможен вывод необработанной информации, на стандартное устройство вывода передается raw-дамп данных EEPROM. По умолчанию выводится полный дамп EEPROM.
-E	Изменяет байт EEPROM для указанного устройства. Смещение и значение байта указываются в качестве параметров опции. Для записи в EEPROM требуется указать ключ (magic key).
-k	Определяет для указанного устройства параметры разгрузки (**offload**).
-K	Устанавливает для указанного устройства параметры разгрузки (**offload**).
rx on|off	Управляет проверкой контрольных сумм на приеме.
tx on|off	Управляет проверкой контрольных сумм при передаче.
sg on|off	Управляет режимом **scatter-gather**.
tso on|off	Управляет режимом разгрузки сегментации TCP
-p N	Инициирует специфические для адаптера операции, позволяющие оператору идентифицировать этот адаптер. Обычно эти операции включают сигналы одного или нескольких светодиодных индикаторов на панели адаптера. Параметр **N** задает продолжительность выполнения операции **phys-id** в секундах.
-r	Инициирует процедуру автоматического согласования параметров среды для указанного устройства, если режим автоматического согласования включен.
-S	Запрашивает статистику работы указанного устройства и его драйвера.
-t offline|online	Выполняет встроенные процедуры тестирования адаптера. В режиме используемом по умолчанию режиме **offline** выполняется полный набор тестов, который может прерывать нормальную работу адаптера во время проверки. Режим **online** включает ограниченный набор тестов, не нарушающих обычную работу адаптера.
-s	Позволяет изменять параметры указанного устройства. Параметр **speed 10|100|1000** устанавливает скорость передачи, **duplex half|full** устанавливает полудуплексную или полнодуплексную связь, **port tp|aui|bnc|mii** задает используемый физический порт адаптера, **autoneg on|off** управляет режимом автоматического согласования параметров, **phyad N** устанавливает физический адрес, **xcvr internal|external** определяет использование внутреннего или внешнего приемопередатчика, **wol p|u|m|b|a|g|s|d...** устанавливает параметры режима Wake-on-LAN[1], **sopass xx:yy:zz:aa:bb:cc** задает пароль **SecureOn**™ (аргумент должен содержать MAC-адрес устройства), а **msglvl N** задает уровень сообщений для драйвера.

11.2.3 Free

Команда **free** показывает объем свободной и использованной памяти системы на основании информации из файла **/proc/meminfo** (стр. 356). Отчет об использовании памяти включает информацию о физической памяти (ОЗУ) и области подкачки (swap), а также сведения об используемых ядром буферах. Выводимые в колонке shared сведения следует игнорировать - эта информация не актуальна и будет удалена в последующих версиях. Ниже показан пример выводимой по команде **free** информации.

1 *Этот режим поддерживается не всеми устройствами. Параметры имеют следующие значения:*
***p** - пробуждение по физической активности;*
***u** - пробуждение по unicast-сообщениям;*
***m** - пробуждение по multicast-сообщениям;*
***b** - пробуждение по широковещательным сообщениям;*
***a** - пробуждение по пакетам ARP;*
***g** - пробуждение по пакетам **MagicPacket**™;*
***s** - включает использование пароля **SecureOn**™ для **MagicPacket**™;*
***d** - отключает режим Wake-on-LAN и отменяет все предыдущие параметры.*

```
             total       used       free     shared    buffers     cached
Mem:        516656     492856      23800          0       2528     154496
-/+ buffers/cache:     335832     180824
Swap:      1020088     464544     555544
```

Рисунок 11.4 Вывод отчета об использовании памяти

Синтаксис

```
free [-b|-k|-m] [-o] [-s delay] [-t] [-V]
```

11.2.3.1 Опции

Таблица 33 Опции команды free

Опция	Описание
`-b`	Задает вывод числовых значений в байтах, взамен принятого по умолчанию вывода в килобайтах (**-k**).
`-m`	Задает вывод числовых значений в мегабайтах, взамен принятого по умолчанию вывода в килобайтах (**-k**).
`-o`	Отключает вывод строки -/+ **buffers/cache**. Если опция **-o** не задана команда **free** вычитает объем буферной памяти из числа занятой и добавляет его к свободной памяти.
`-s`	Активизирует повторяющийся вывод информации с заданным периодом (в секундах). Вы можете указывать значение задержки с точностью до 1 мксек, поскольку для организации периодического процесса используется функция **usleep**, обеспечивающая разрешение в 1 мксек.
`-t`	Задает вывод строки с данными о суммарном расходе памяти (ОЗУ + файл подкачки) в строке **Total**.
`-V`	Выводит сведения о номере версии программы.

11.2.4 Ifplugstatus

http://0pointer.de/lennart/projects/ifplugd/

Утилита **ifplugstatus**[1] из пакета **ifplugd**[2] служит для проверки состояния соединений локальных интерфейсов Ethernet.

Синтаксис

```
ifplugstatus [<опции>] [<интерфейс>]
```

Утилита **ifplugstatus** может использоваться для контроля состояния соединений локальных устройств Ethernet для Linux0систем, подобно программам **mii-diag**, **mii-tool** (см. параграф 11.1.2.4 на стр. 198) и **ethtool** (см. параграф 11.2.2 на стр. 206). Фактически, программа поддерживает три различных интерфейса API для работы с перечисленными утилитами, что обеспечивает максимальный уровень гибкости и совместимости. Кроме того, программа поддерживает проверку соединений с флагом **IFF_RUNNING**, который поддерживает большинство современных устройств (не только Ethernet). Обеспечивается также возможность проверки беспроводных соединений.

Интерфейсы API используются в перечисленном ниже порядке:

1) наиболее современный API **SIOCETHTOOL** (**ethtool API**);
2) более старый интерфейс **SIOCGMIIREG** (**mii-diag API**);
3) расширение **WLAN** API (**iwconfig API**);
4) **IFF_RUNNING** (**ifconfig API**).

Самый старый интерфейс **SIOCPRIV** (**mii-tool API**) уже не используется программой.

Утилиту можно использовать в shell-сценариях, поскольку она возвращает текущее состояние интерфейса. Особенно полезна возможность детектирования поддерживаемых драйвером API с помощью опции **-v**.

11.2.4.1 Опции

Если в командной строке не указано конкретное устройство, команда **ifplugstatus** будет проверять все доступные сетевые интерфейсы локального хоста в соответствии с заданными опциями (таблица 34).

Таблица 34 Опции команды ifplugstatus

Опция		Значение
`-a`	`--auto`	Автоматически активизирует интерфейсы перед проведением проверки. По умолчанию режим автоматической активизации отключен.

1 *В ранних версиях эта утилита носила название **ifstatus**.*
2 *Исходные тексты ifplugd вы сможете найти в каталоге SRC/ приложенного к книге компакт-диска.*

Опция		Значение
-h	**--help**	Выводит на экран справочную информацию.
-q	**--quiet**	Уменьшает объем выводимой информации. Может использоваться в команде неоднократно.
-v	**--verbose**	Увеличивает объем выводимой информации. Может использоваться в команде неоднократно.
-V	**--version**	Выводит сведения о номере версии программы.

11.2.4.2 Возвращаемые значения

0 - успешное завершение проверки;

1 - отказ при проверке;

2 - обнаружено активное соединение (только при указанном в командной строке интерфейсе);

3 - кабель отключен (только при указанном в командной строке интерфейсе).

11.2.5 Pgrep, pkill

Утилиты **pgrep** и **pkill** служат для просмотра списка процессов или передачи процессам сигнала. Для выбора процессов используются их имена и другие атрибуты (сами процессы **pgrep** и **pkill** никогда не входят в число выбранных).

Синтаксис

```
pgrep [-flvx] [-d delimiter] [-n|-o] [-P ppid,...] [-g pgrp,...] [-s sid,...] [-u
euid,...] [-U uid,...] [-G gid,...] [-t term,...] [pattern]

pkill [-signal] [-fvx] [-n|-o] [-P ppid,...] [-g pgrp,...] [-s sid,...] [-u euid,...] [-U
uid,...] [-G gid,...] [-t term,...] [pattern]
```

Утилита pgrep просматривает список процессов в системе и выводит список процессов, соответствующих заданным критериям отбора; **pkill** передает выбранным процессам указанный в командной строке сигнал (по умолчанию **SIGTERM**). Например, команда

```
pgrep -u root sshd
```

будет выводить список процессов с именем **sshd**, которыми владеет пользователь **root**. Команда

```
pgrep -u root,daemon
```

покажет все процессы, которыми владеют пользователи **root** и **daemon**.

11.2.5.1 Опции

Таблица 35 Опции pgrep и pkill

Опция	Описание
-d delimiter	Задает последовательность символов, используемую для разграничения идентификаторов процессов при выводе информации. По умолчанию каждый процесс указывается в новой строке. Эта опция поддерживается только для команды **pgrep**.
-f	Эта опция задает поиск шаблона **pattern** во всей командной строке. По умолчанию проверяется только имя процесса.
-g pgrp,...	Задает отбор процессов по идентификатору группы. Процессы с идентификатором группы 0 транслируются в группу, владеющую процессом **pgrep** или **pkill**.
-G gid,...	Задает отбор процессов по реальной группе, заданной именем или идентификатором.
-l	Задает вывод имен и идентификаторов процессов (только для **pgrep**).
-n	Задает вывод списка последних (недавно запущенных) процессов.
-o	Задает вывод списка самых старых (давно запущенных) процессов.
-P ppid,...	Задает выбор по идентификатору родительского процесса.
-s sid,...	Задает выбор по идентификатору сессии. Сессии с идентификатором 0 транслируются в сессию **pgrep** или **pkill**.
-t term,...	Задает выбор процессов по управляющему терминалу. Имена терминалов следует указывать без префикса **/dev/**.
-u euid,...	Задает выбор процессов по эффективному идентификатору пользователя, указанного именем или значением UID.
-U uid,...	Задает выбор процессов по реальному пользователю, указанному именем или UID.
-v	Меняет (инвертирует) условие отбора.
-x	Задает выбор процессов, имя[1] соответствует указанному в команде шаблону поиска[2].

1 *Или командная строка полностью при использовании опции **-f**.*

Опция	Описание
-signal	Определяет сигнал, передаваемый отобранным процессам. Сигнал может быть задан именем или номером. Эта опция может использоваться только с командой **pkill**.
pattern	Задает расширенное регулярное выражение (Extended Regular Expression) для поиска соответствия в именах процессов или строках команд.

11.2.5.2 Примеры использования

Команда поиска процессов демона named:

```
pgrep -u root named
```

Команда для инициирования повторного прочтения конфигурационного файла демоном **syslog** (перезапуск процесса с новыми параметрами):

```
pkill -HUP syslogd
```

Команда просмотра информации о всех процессах **xterm**:

```
ps -fp $(pgrep -d, -x xterm)
```

Команда повышения уровня приоритета для всех процессов Netscape:

```
renice +4 `pgrep netscape`
```

11.2.5.3 Коды возврата

0 - найден по крайней мере один процесс, соответствующий заданным критериям;

1 - не найдено ни одного процесса, соответствующего критериям поиска;

2 - синтаксическая ошибка в командной строке;

3 - критическая ошибка при работе программы (нехватка памяти и т. п.).

11.2.5.4 Известные проблемы

1) Опция **-v** не может использоваться совместно с **-n** или **-o**.

2) Вывод включает информацию об умерших процессах.

11.2.6 Ps

Утилита **ps** позволяет просматривать состояние процессов в системе

Синтаксис

```
ps [<опции>]
```

Утилита **ps** дает “моментальный кадр” состояния процессов системы. Если вы хотите просматривать динамику процессов, воспользуйтесь программой **top** (см. параграф 11.2.9 на стр. 218). Работа утилиты **ps** основана на просмотре переменных SysCtl, хранящихся в структуре каталогов **/proc**.

11.2.6.1 Опции командной строки

Работающая с переменными SysCtl версия **ps** принимает опции трех типов, которые могут использоваться вперемешку.

- Опции **Unix** могут объединяться в группы и каждая группа опция должна начинаться с дефиса (**-**);
- Опции **BSD** также могут группироваться, но для группы не должен использоваться дефис[1];
- Опции Gnu должны начинаться с двух дефисов (**--**).

11.2.6.1.1 Простой выбор процессов

Опция	Значение	Опция	Значение
-A	Выбрать все процессы.	**a**	Выбрать все процессы на терминале, включая процессы других пользователей.
-N	Исключить выбранные процессы.	**g**	Выбрать все процессы, включая лидеров групп.
-a	Выбрать все терминальные процессы, исключив процессы инициирования терминальных сессий (session leader)	**r**	Выводить информацию только для работающих процессов.

2 *При поиске процессов по имени (без учета командной строки в целом) принимается во внимание только 15 символов имени процесса, включенные в файл **/proc/pid/stat**. Если этого недостаточно, используйте опцию **-f**.*

1 *Переменная окружения **I_WANT_A_BROKEN_PS** позволяет трактовать опции как BSD даже при наличии перед группой опций дефиса. Переменная окружения **PS_PERSONALITY** (см. стр. 216) позволяет контролировать трактовку опций **ps**.*

Опция	Значение	Опция	Значение
-d	Выбрать все процессы, исключив процессы инициирования терминальных сессий (session leader)	**x**	Выбрать процессы без управляющего терминала.
-e	Выбрать все процессы.	**--deselect**	Отменить выбор процессов
T	Выбрать все процессы данного терминала.		

11.2.6.1.2 Выбор процессов по списку

Опция	Значение	Опция	Значение
-C	Выбрать процессы по имени команды.	**--Group**	Выбрать процессы по реальному идентификатору группы (GID).
-G	Выбрать процессы по реальному идентификатору группы (GID).	**--User**	Выбрать процессы по реальному идентификатору пользователя (UID).
-U	Выбрать процессы по реальному идентификатору пользователя (UID).	**--group**	Выбрать процессы по эффективному идентификатору группы (GID).
-g	Выбрать процессы по идентификатору группы GID или инициатору сессии.	**--pid**	Выбрать процессы по идентификатору процесса (PID).
-p	Выбрать процессы по идентификатору процесса (PID).	**--ppid**	Выбрать процессы по идентификатору родительского процесса (PID).
-s	Выбрать процессы, относящиеся к данной сессии.	**--sid**	Выбрать процессы по идентификатору сессии (SID).
-t	Выбрать процессы по номеру терминала tty.	**--tty**	Выбрать процессы по номеру терминала tty.
-u	Выбрать процессы по эффективному идентификатору пользователя (UID).	**--user**	Выбрать процессы по эффективному идентификатору пользователя (UID).
U	Выбрать процессы указанных пользователей.	**-123**	Выбрать процессы по указанному идентификатору сессии
p	Выбрать процессы по идентификатору процесса (PID)	**123**	Выбрать процессы по указанному идентификатору процесса
t	Выбрать процессы по номеру терминала tty.		

11.2.6.1.3 Опции управления форматом вывода

Опция	Значение	Опция	Значение
-O	Предварительно загруженный пользовательский формат вывода (preloaded **-o**). Пользовательские форматы описана в параграфе 11.2.6.2 (стр. 212).	**X**	Старый формат с регистрами Linux i386
-F	Максимально подробный вывод.	**j**	Формат управления заданиями.
-c	Подобна опции **-l**, но выводит другой набор информации.	**l**	Подробный вывод информации (long-формат).
-f	Подробный вывод информации.	**o**	Пользовательский формат вывода.
-j	Формат управления заданиями.	**s**	Вывод информации о сигналах.
-l	Подробный вывод информации (long-формат).	**u**	Задает вывод информации о пользователях.
-o	Пользовательский формат вывода (см. параграф 11.2.6.2 на стр. 212).	**v**	Задает вывод информации об использовании виртуальной памяти.
-y	Отключает вывод флагов и выводит **rss** взамен адреса	**--format**	Пользовательский формат вывода (см. параграф 11.2.6.2 на стр. 212).
O	Предварительно загруженный пользовательский формат вывода (preloaded **-o**). Пользовательские форматы описана в параграфе 11.2.6.2 (стр. 212).	**--context**	Выводит информацию о безопасности (NSA SELinux и т. п.)

11.2.6.1.3.1 Модификаторы формата вывода

Опция	Значение	Опция	Значение
-H	Показывать иерархию процессов (дерево).	**w**	Широкий формат вывода (длинные строки).
-n	Задает файл со списком имен (namelist) взамен файла System.map.	**--cols**	Задает ширину экрана для вывода.

Опция	Значение	Опция	Значение
-w	Широкий формат вывода (длинные строки).	**--columns**	Задает ширину экрана для вывода.
C	Выводить в колонке %CPU необработанное значение времени CPU взамен среднего	**--cumulative**	Задает вывод некоторых данных для “умерших” дочерних процессов.
N	Задает файл со списком имен (namelist).	**--forest**	Показывать иерархию процессов (дерево) с использованием ASCII-art.
O	Задает порядок сортировки при выводе.	**--headers**	Задает повтор заголовка на каждой странице вывода.
S	Задает вывод некоторых данных для “умерших” дочерних процессов.	**--no-headers**	Отключает вывод заголовка.
c	Задает корректный вывод имен команд.	**--lines**	Задает число строк экрана для вывода.
e	Задает вывод переменных окружения после имени команды	**--rows**	Задает число строк экрана для вывода.
f	Показывать иерархию процессов (дерево) с использованием псевдографики.	**--sort**	Задает порядок сортировки при выводе.
h	Отключает вывод заголовка.	**--width**	Задает ширину экрана для вывода.
n	Цифровой формат полей **WCHAN** и **USER**.		

11.2.6.1.4 Опции управления выводом информации о потоках

Опция	Значение
-L	Показывать потоки (возможно с выводом колонок **LWP** и **NLWP**).
-T	Показывать потоки (возможно с выводом колонки **SPID**).
-m	Показывать потоки после процессов.
H	Показывать потоки вместе с процессами.
m	Показывать потоки после процессов.

11.2.6.1.5 Информационные опции

Опция	Значение
-V	Выводит информацию о номере версии программы.
L	Выводит список указателей формата вывода.
V	Выводит информацию о номере версии программы.
--help	Выводит краткую справку о работе с программой.
--info	Выводит отладочную информацию (параметры компиляции).
--version	Выводит информацию о номере версии программы.

11.2.6.2 Пользовательский формат вывода

Опции пользовательского формата вывода (**o**, **-o**, **O** и **-O**) обеспечивают эффективный способ управления отдельными колонками отчета программы. Пользовательские форматы позволяют изменять названия заголовков

```
ps -o pid,ruser=RealUser -o comm=Command
```

Если опции пользовательского формата содержат только пустые имена заголовков

```
ps -o pid= -o comm=
```

строки заголовка не будет в выводе программы. Ширина колонок вывода будет определяться размером заголовка колонки. Например, для расширения колонки **WCHAN** можно воспользоваться командой

```
ps -o pid,wchan=WIDE-WCHAN-COLUMN -o comm
```

Поддерживается также возможность явного задания ширины колонок вывода

```
ps o pid,wchan:42,cmd
```

Формат вывода команд типа

```
ps -o pid=X,comm=Y
```

зависит от выбранного способа трактовки опций (параграф 11.2.6.6.4 на стр. 216) - информация может выводиться в одной колонке с именем **X,comm=Y** или в двух колонках с именами **X** и **Y**. Чтобы избавиться от подобных неоднозначностей, можно использовать в команде несколько опций **-o**. Переменная окружения **$PS_FORMAT** позволяет задать используемый по умолчанию формат вывода, а макросы **DefSysV** и **DefBSD** могут применяться для вывода колонок в принятом по умолчанию формате UNIX или BSD.

Указатели колонок пользовательских форматов **comm**, **args**, **cmd**, **comm**, **command**, **fname**, **ucmd**, **ucomm**, **lstart**, **bsdstart**, **start** могут включать пробелы.

11.2.6.3 Специфика работы с программой

Опция **-g** служит для выбора процессов по инициатору сессии (session leader) или имени группы. Выбор процессов по инициатору сессии поддерживается многими стандартами, однако выбор по имени группы может осуществляться в разных ОС по своему. Версия **ps**, работающая на основе переменных SysCtl будет осуществлять выбор по инициаторам процессов, когда список содержит только числовые идентификаторы, а номера групп будут учитываться лишь в тех случаях, когда указаны также имена некоторых групп.

Опцию **m** не следует включать в команду, а вместо нее лучше использовать опции **-m** или **-o** со списком (**m** может задавать вывод сведений об использовании памяти, просмотр потоков или сортировку по расходу памяти, поэтому легко ошибиться при включении этой опции в команду).

Использование опции **h** может порождать проблемы. Стандартная утилита **ps** в BSD использует эту опцию для вывода заголовка на каждой странице, но в старых версиях Linux **ps** та же опция глобально отключает вывод заголовка. Работающая с переменными SysCtl современная версия **ps** для Linux не будет выводить заголовок, если не выбрана трактовка опций в стиле BSD (параграф 11.2.6.6.4 на стр. 216), а в этом случае заголовки будут выводиться на каждой странице. Независимо от выбранного режима трактовки длинные опции **--headers** и **--no-headers** позволяют управлять выводом заголовка.

Терминалы (**tty** или **pst**) можно задавать в форматах **/dev/ttyS1**, **ttyS1**, **S1**. Поддерживаются также устаревшие варианты **ps t** (данный терминал) и **ps t?** (процессы, не использующие терминал), но лучше использовать вместо них более современные варианты (**T**, **-t** со списком, **x**, **t** со списком).

Опция **O** в BSD может работать подобно **-O** (пользовательский формат с предопределенными полями) или служить для задания порядка сортировки. Для определения роли опции в конкретной команде используются эвристические методы. Для получения однозначного результата старайтесь использовать другие опции.

Для сортировки используется устаревший синтаксис BSD в формате **O[+|-]k1[,[+|-]k2[,...]]**. Ключи сортировки применяются в заданном опцией порядке **k1, k2, ...** Необязательный символ **+** задает прямой порядок[1] сортировки (используется по умолчанию), а знак **-** меняет порядок сортировки на обратный. Опция **O** должна быть последней в аргументе команды, но это не запрещает использовать после нее другие аргументы.

В GNU для сортировки используется синтаксис **-sortX[+|-]key[,[+|-]key[,...]]** и порядок применения ключей также соответствует их порядку в опции. В качестве **X** может использоваться любой допустимый символ-разделитель (в GNU обычно используется символ **=**). Символ **+** задает прямой порядок сортировки и может быть опущен, а символ **-** меняет порядок сортировки на обратный. Примером может служить команда

```
ps jax --sort=uid,-ppid,+pid
```

Работающей с файлами **/proc** версии программы **ps** не требуется каких-либо специальных привилегий или установки бита SUID. Поэтому не следует предоставлять программе ps какие-либо дополнительные привилегии.

Утилите **ps** нужен доступ к списку имен для корректного отображения значений WCHAN. Этот список должен соответствовать текущей версии ядра Linux. Программа **ps** ищет список имен в фале **System.map**, просматривая в указанном порядке:

```
$PS_SYSTEM_MAP
/boot/System.map-`uname -r`
/boot/System.map
/lib/modules/`uname -r`/System.map
/usr/src/linux/System.map
```

Программы, сброшенные на диск (swap) указываются без аргументов командной строки и, если не задана опция **c**, выводятся в квадратных скобках.

Колонка **%CPU** показывает уровень загрузки процессора/использования времени.

Поля **SIZE** и **RSS** не учитывают таблицы страниц и структуру **task_struct** из **proc** - на это расходуется по крайней мере 12 кбайт оперативной памяти. Поле **SIZE** показывает виртуальный размер (код + данные + стек).

Процессы, помеченные как **<defunct>**, являются “мертвыми”[2] - такие процессы образуются в тех случаях, когда родительский процесс не удаляет корректно порожденные им процессы. Эти процессы могут быть удалены функцией **init**, если родительский процесс еще существует.

11.2.6.4 Флаги процессов

Флаг	Значение	Описание
FORKNOEXEC	1	Ответвлен, но не выполнен.
SUPERPRIV	4	Имеет привилегии суперпользователя.

11.2.6.5 Коды состояния процессов

D - спит беспробудно (обычно это процессы ввода-вывода);

R - выполняется (или находится в очереди на исполнение);

S - спит;

T - трассируется или остановлен;

1 *По возрастанию или алфавиту.*
2 *Их называют также zombie (зомби).*

W - сбрасывание на диск (paging);

X - умер;

Z - на функционирует ("zombie").

Для форматов BSD и при использовании ключевого слова **stat** могут также выводиться дополнительные идентификаторы кодов:

W - не имеет резидентных страниц;

< - процесс с высоким приоритетом;

N - задача с низким приоритетом;

L - имеет страницы в памяти.

11.2.6.6 Ключи сортировки

Ключ		Сортировка
c	**cmd**	По имени исполняемого файла.
C	**cmdline**	По полной командной строке.
f	**flags**	По флагам.
g	**pgrp**	По идентификаторам группы.
G	**tpgid**	По идентификаторам группы управляющего терминала.
j	**cutime**	По суммарному времени пользователей.
J	**cstime**	По суммарному использованию системного времени.
k	**utime**	По занимаемому пользователем времени.
K	**stime**	По системному времени.
m	**min_flt**	По количеству несущественных сбоев для страниц.
M	**maj_flt**	По количеству существенных сбоев для страниц.
n	**cmin_flt**	По суммарному количеству несущественных сбоев.
N	**cmaj_flt**	По суммарному количеству существенных сбоев.
o	**session**	По идентификаторам сессий.
p	**pid**	По идентификаторам процессов.
P	**ppid**	По идентификаторам родительских процессов.
r	**rss**	По размеру резидентной части.
R	**resident**	По количеству резидентных страниц.
s	**size**	По размеру памяти.
S	**share**	По количеству разделяемых страниц памяти.
t	**tty**	По младшему номеру терминала tty.
T	**start_time**	По времени запуска.
U	**uid**	По идентификаторам пользователей.
u	**user**	По именам пользователей.
v	**vsize**	По размеру виртуальной памяти.
y	**priority**	По запланированному ядром приоритету.

11.2.6.6.1 Стандартные указатели формата

Перечисленные здесь поля могут использоваться как для форматирования вывода, та и для сортировки. Например,

```
ps -eo pid,user,args --sort user
```

Таблица 36 Стандартные указатели формата ps

Код	Заголовок	Код	Заголовок	Код	Заголовок
%cpu	%CPU	euid	EUID	m_drs	DRS
%mem	%MEM	euser	EUSER	m_trs	TRS
alarm	ALARM	f	F	maj_flt	MAJFL
args	COMMAND	fgid	FGID	majflt	MAJFLT
blocked	BLOCKED	fgroup	FGROUP	min_flt	MINFL

Код	Заголовок	Код	Заголовок	Код	Заголовок
bsdstart	START	flag	F	minflt	MINFLT
bsdtime	TIME	flags	F	ni	NI
c	C	fname	COMMAND	nice	NI
caught	CAUGHT	fsgid	FSGID	nwchan	WCHAN
cmd	CMD	fsgroup	FSGROUP	opri	PRI
comm	COMMAND	fsuid	FSUID	pagein	PAGEIN
command	COMMAND	fsuser	FSUSER	pcpu	%CPU
context	CONTEXT	fuid	FUID	pending	PENDING
cputime	TIME	fuser	FUSER	pgid	PGID
drs	DRS	gid	GID	pgrp	PGRP
dsiz	DSIZ	group	GROUP	pid	PID
egid	EGID	ignored	IGNORED	pmem	%MEM
egroup	EGROUP	intpri	PRI	ppid	PPID
eip	EIP	lim	LIM	pri	PRI
esp	ESP	longtname	TTY	priority	PRI
etime	ELAPSED	lstart	STARTED	rgid	RGID
rgroup	RGROUP	sigcatch	CAUGHT	tmout	TMOUT
rss	RSS	sigignore	IGNORED	tname	TTY
rssize	RSS	sigmask	BLOCKED	tpgid	TPGID
rsz	RSZ	stackp	STACKP	trs	TRS
ruid	RUID	start	STARTED	trss	TRSS
ruser	RUSER	start_stack	STACKP	tsiz	TSIZ
s	S	start_time	START	tt	TT
sess	SESS	stat	STAT	tty	TT
session	SESS	state	S	tty4	TTY
sgi_p	P	stime	STIME	tty8	TTY
sgi_rss	RSS	suid	SUID	ucmd	CMD
sgid	SGID	suser	SUSER	ucomm	COMMAND
sgroup	SGROUP	svgid	SVGID	uid	UID
sid	SID	svgroup	SVGROUP	uid_hack	UID
sig	PENDING	svuid	SVUID	uname	USER
sig_block	BLOCKED	svuser	SVUSER	user	USER
sig_catch	CATCHED	sz	SZ	vsize	VSZ
sig_ignore	IGNORED	time	TIME	vsz	VSZ
sig_pend	SIGNAL	timeout	TMOUT	wchan	WCHAN

11.2.6.6.2 Указатели формата AIX

Программа **ps** поддерживает дескрипторы формата AIX, работающие как коды форматирования для функции printf. Например,

```
ps -eo "%p %y %x %c"
```

Код		Заголовок	Код		Заголовок	Код		Заголовок
%C	pcpu	%CPU	%c	comm	COMMAND	%t	etime	ELAPSED
%G	group	GROUP	%g	rgroup	RGROUP	%u	ruser	RUSER
%P	ppid	PPID	%n	nice	NI	%x	time	TIME
%U	user	USER	%p	pid	PID	%y	tty	TTY
%a	args	COMMAND	%r	pgid	PGID	%z	vsz	VSZ

11.2.6.6.3 Переменные окружения

Команда ps использует переменные окружения, перечисленные в таблице 37.

Таблица 37 Переменные окружения, используемые ps.

Переменная	Описание
COLUMNS	Задает ширину экрана в символах.
LINES	Количество строк на экране.
PS_PERSONALITY	Задает способ трактовки (**posix**, **old**, **linux**, **bsd**, **sun**, и т. п. - см. таблицу 38).
CMD_ENV	Задает способ трактовки (**posix**, **old**, **linux**, **bsd**, **sun**, и т. п. - см. таблицу 38).
I_WANT_A_BROKEN_PS	Задает старый стиль интерпретации командной строки
LC_TIME	Задает формат вывода дат.
PS_COLORS	Не поддерживается.
PS_FORMAT	Задает используемый по умолчанию формат вывода.
PS_SYSMAP	Задает используемое по умолчанию местоположение файла System.map.
PS_SYSTEM_MAP	Задает используемое по умолчанию местоположение файла System.map.
POSIXLY_CORRECT	Не искать оправданий для игнорирования некорректно заданных параметров.
UNIX95	Не искать оправданий для игнорирования некорректно заданных параметров.
_XPG	Отключает нестандартный режим **CMD_ENV=irix**.

11.2.6.6.4 Управление трактовкой опций

В общем случае не следует устанавливать перечисленные в таблице 38 значения для переменных окружения. Единственным исключением может служить установка для переменных **CMD_ENV** и **PS_PERSONALITY** значения Linux, поскольку без этого **ps** будет следовать старому и бесполезному формату **Unix98**.

Таблица 38 Ключи трактовки параметров ps

Ключ	Трактовка	Ключ	Трактовка
390	Стиль **S/390 OpenEdition**.	old	Старый стиль **Linux**.
aix	Стиль **AIX**.	posix	Стандартный стиль.
bsd	Стиль **FreeBSD** (нестандартный).	sco	Стиль **SCO**.
compaq	Стиль **Digital Unix**.	sgi	Стиль **Irix**.
debian	Старый стиль **Debian**.	sun	Стиль **SunOS 4** (нестандартный).
digital	Стиль **Digital Unix**.	sunos	Стиль **SunOS 4** (нестандартный).
gnu	Старый стиль **Debian**.	sysv	Стандартный стиль.
hp	Стиль **HP-UX**.	unix	Стандартный стиль.
hpux	Стиль **HP-UX**.	unix95	Стандартный стиль.
irix	Стиль **Irix**.	unix98	Стандартный стиль.
linux	Стиль **Linux**.		

11.2.6.7 Примеры использования

Просмотр всех процессов с использованием стандартного синтаксиса:

```
ps -e
```

Просмотр всех процессов с использованием синтаксиса BSD:

```
ps ax
```

Просмотр всех процессов, за исключением процессов пользователя root (реальный и эффективный UID):

```
ps -U root -u root -N
```

Просмотр всех процессов с выводом в пользовательском формате:

```
ps -eo pid,tt,user,fname,tmout,f,wchan
```

Использование для описания полей дескрипторов AIX:

```
ps -o "%u : %U : %p : %a"
```

Показать только идентификатор процесса **syslogd**:

```
ps -C syslogd -o pid=
```

11.2.7 Pstree

Утилита **pstree** выводит дерево активных процессов системы. В качестве корня дерева используется указанный идентификатором процесс или процесс **init**, если команда не включает идентификатор процесса. Если в командной строке указан пользователь, выводятся только процессы этого пользователя.

Синтаксис

```
pstree [-a] [-c] [-h|-Hpid] [-l] [-n] [-p] [-u] [-G|-U] [pid|user]
pstree -V
```

Программа визуально связывает идентичные ветви (компактирует), помещая их в квадратные скобки и указывая впереди количество повторений, как показано ниже.

```
init---4*[getty]
```

Развернутый вариант представления дерева будет иметь вид:

```
init-+-getty
     |-getty
     |-getty
     `-getty
```

11.2.7.1 Опции

Таблица 39. Опции pstree.

Опция	Описание
-a	Задает вывод аргументов командной строки процессов. Если процесс сброшен в область подкачки, он обычно указывается в квадратных скобках. Опция **-a** неявно отменяет такой подход.
-c	Отключает объединение идентичных ветвей дерева, используемое по умолчанию.
-G	Задает использование псевдографики VT100.
-h	Выделяет текущий процесс и его родителей. Если терминал не поддерживает выделения или вывод не содержит текущего процесса и его родителей, опция игнорируется.
-H	Эта опция подобна флагу **-h**, но при невозможности выделения процесса выдается сообщение об ошибке.
-l	Задает вывод длинных строк. По умолчанию длина строки ограничена шириной терминала или значением 132 (для устройств, отличных от tty и терминалов с неизвестной шириной экрана).
-n	Задает сортировку процессов, имеющих общего родителя по значению PID вместо имени процесса.
-p	Задает вывод идентификаторов процессов (PID), выводимых в форме десятичных чисел, заключенных в скобки. Опция **-p** неявно запрещает компактирование вывода.
-u	Показывает смену идентификатора пользователя. Если порожденным процессом владеет другой пользователь, после имени этого процесса указывается значение идентификатора нового пользователя.
-U	Задает использование символов UTF-8 (Unicode) для рисования линий.
-V	Выводит информацию о номере версии программы.

11.2.8 SysCtl

Программа **sysctl** позволяет просматривать и изменять конфигурационные параметры ядра на работающей системе.

Синтаксис

```
sysctl [-n] [-e] variable ...
sysctl [-n] [-e] -w variable=value ...
sysctl [-n] [-e] -p <filename>
sysctl [-n] [-e] -a
sysctl [-n] [-e] -A
```

Команда **sysctl** позволяет без перезагрузки системы изменять некоторые параметры ядра, хранящиеся в файлах **/proc/sys/** (параграф 12.2.1.12 на стр. 362). Для использования этой команды требуется ядро со включенной опцией **Sysctl support** (параграф 4.4.1.2.5 на стр. 64).

11.2.8.1 Параметры

Таблица 40 Параметры команды sysctl

Параметр	Описание
variable	Имя переменной (файла) для чтения информации (например, **kernel.ostype**). В качестве разделителей полей имени могут использоваться символы '**/**' и '**.**'.
variable=value	Выражение, используемое для задания значения указанной переменной. Если значение содержит кавычки или иные символы, имеющие специальное значение для командного процессора, значение параметра следует указывать в двойных кавычках. Для изменения параметров команда должна включать опцию **-w**.
-n	Отключает вывод имен переменных вместе с их значениями.
-v	Включает вывод имен переменных вместе с их значениями.

Параметр	Описание
-e	Отключает вывод сообщений об ошибках, связанных с некорректным именованием переменных.
-w	Задает изменение параметра, указанного парой “переменная=значение”.
-p	Задает имя конфигурационного файла (по умолчанию **/etc/sysctl.conf**).
-a	Выводит полный набор доступных параметров.
-A	Выводит полный набор доступных параметров в форме таблицы.

11.2.9 Top

Утилита **top** выводит список задач Linux, упорядоченный в соответствии с указанными критериями.

Синтаксис

```
top -hv | -bcisS -d delay -n iterations -p pid [, pid ...]
```

Программа **top** выводит на экран динамически обновляемый список задач, работающих в системе. Программа обеспечивает широкие возможности выбора выводимой информации - от простого резюме загрузки системы до подробного списка процессов, работающих под управлением ядра Linux. Все выводимые программой поля можно выбирать; обеспечивается также возможность записи выбранных параметров для использования при следующих запусках программы.

Поддерживается также некоторый набор интерактивных функций управления выводом, описанных в параграфе 11.2.9.3 (стр. 221).

При использовании программы наиболее важными командами интерактивного режима являются команда вывода справки (**h** или **?**) и команда завершения сеанса (**q**)[1].

При первом запуске программы на экран выводятся заданные по умолчанию элементы (см. рисунок 11.5):

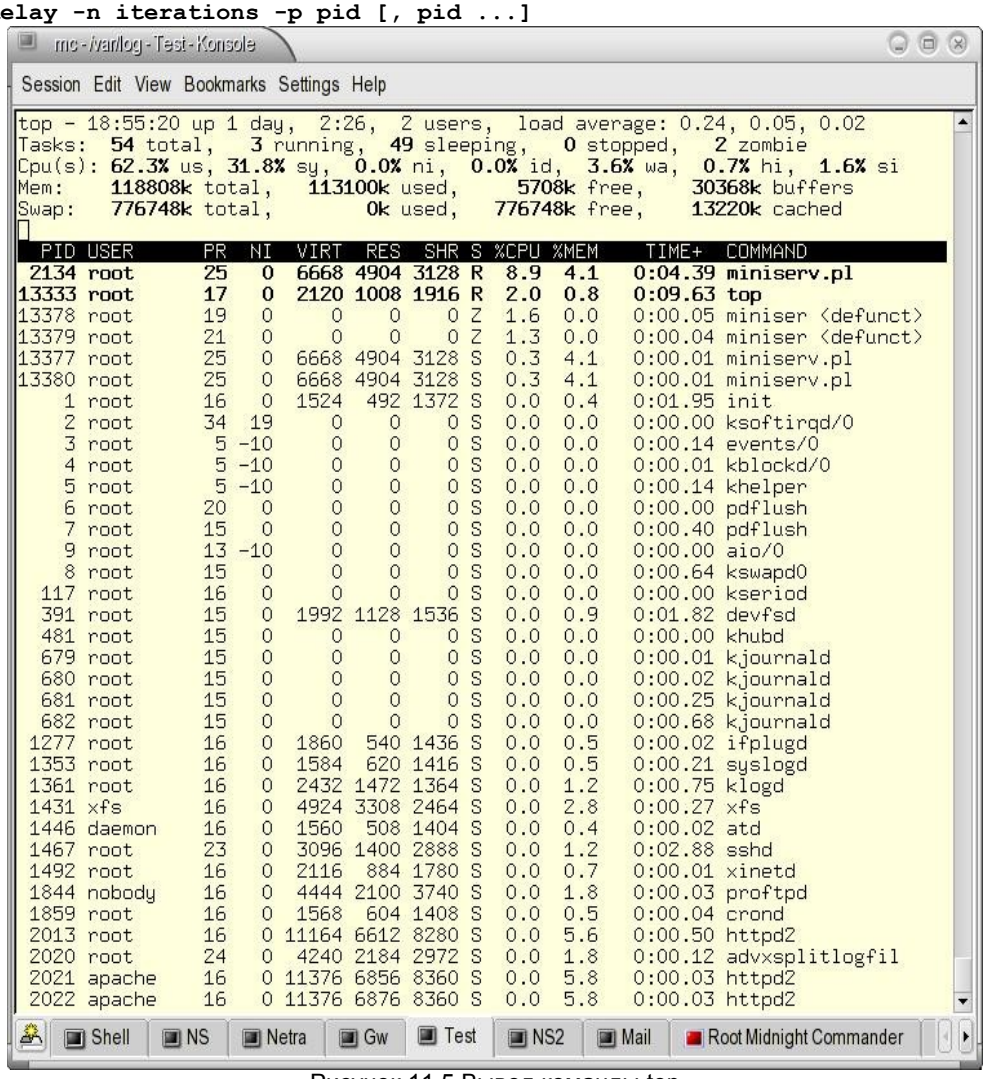


Рисунок 11.5 Вывод команды top

- Данные о работе системы в целом;
- Строка сообщений или приглашения;
- Срока заголовков колонок;
- Список задач.

В списке процессов работающие в данный момент или готовые к работе процессы выделяются жирным шрифтом.

При запуске программы она может считывать параметры из конфигурационного файла, за исключением перечисленных в таблице 41 параметров, которые приняты по умолчанию и не могут быть заданы в конфигурационном файле[2].

1 *Для прерывания работы программы можно использовать и стандартную комбинацию клавиш **Ctrl+C**.*
2 *Параметры выделенные курсивом, могут быть изменены при запуске программы с помощью опций командной строки (параграф 11.2.9.1 на стр. 219).*

Таблица 41. Используемые по умолчанию параметры top.

Параметр	Значение	Описание
Глобальные параметры		
A	Отключено (полно-экранный режим)	Использование дополнительного дисплея.
d	*3 секунды*	*Период обновления информации*
I	Включен	Режим Irix.
p	*Отключен*	*Мониторинг PID.*
s	*Отключен*	*Безопасный режим.*
B	Отключен	Запрет использования жирного шрифта (bold).
Информация о системе в целом		
l	Включено	Load Avg/Uptime - выводить строку с информацией о средней загрузке и времени работы.
t	Включено	Task/Cpu - выводить строки с информацией о задачах и суммарной загрузке процессора.
m	Включено	Mem/Swap - выводить строки сведений об использовании ОЗУ и файла подкачки.
1	Включено	Single Cpu - указывать сведения для одного процессора.
Список задач		
b	Включено	Выделение жирным шрифтом (взамен инверсии цвета).
c	*Выключено*	*Выводить командную строку.*
i	*Включено*	*Показывать бездействующие (Idle) задачи.*
R	Включено	Сортировка в обратном порядке.
S	*Выключено*	*Выводить суммарное время.*
x	Выключено	Выделение колонки (сортировка)
y	Включено	Выделение строки (активные задачи)
z	Включено	Монохромный режим.

11.2.9.1 Опции командной строки

Строка параметров команды top имеет вид[1]:

```
-hv|-bcisS -d delay -n iterations -p pid [,pid...]
```

Опции командной строки перечислены в таблице 42.

Таблица 42 Опции команды top

Опция	Описание
-b	Задает пакетный режим работы (Batch mode) программы, который может быть удобен при выводе результатов в файл или их передаче другой программе. В это режиме программа не будет воспринимать данные со стандартного устройства ввода и будет продолжать работу, пока не достигнет заданного опцией **-n** числа итераций или не будет прервана с помощью команды **kill**.
-c	Задает запуск программы с обращением последнего сохраненного состояния вывода имен программ или командной строки[2]. См. Также описание интерактивной команды **c** на стр. 223.
-d	Параметр этой опции задает период опроса в формате **ss.tt** (секунды и сотые доли секунд). Значение этого параметра отменяет установки, заданные в конфигурационном файле или принятые по умолчанию. Период опроса можно также изменить с помощью интерактивных команд **d** и **s**. При работе в безопасном режиме (стр.) изменять период опроса может только пользователь **root**.
-h	Выводит на экран краткую справку и завершает работу программы.
-i	Задает запуск программы с обращением последнего сохраненного состояния вывода информации о бездействующих процессах.
-n	Значение параметра этой опции задает количество повторов сбора и вывода информации в сеансе работы программы. Эта опция весьма полезна для пакетного режима, задаваемого ключом **-b**.
-u	Параметр этой опции (имя или эффективный идентификатор) указывает пользователя, для чьих процессов должен осуществляться мониторинг.

1 *Отметим, что пробелы и символы дефиса можно не включать в команду.*
2 *Если при предыдущем запуске выводились емена программ, сейчас будет выводиться командная строка и наоборот.*

Опция	Описание
-U	Параметр этой опции (имя или UID) указывает пользователя, для чьих процессов должен осуществляться мониторинг. Мониторинг будет обеспечиваться для всех процессов с совпадающим значением реального, эффективного и сохраненного UID а также значения UID из файловой системы.
-p	Параметры этой опции задают идентификаторы процессов, для которых должен производиться мониторинг. Опция может включать до 20 значений идентификаторов, разделенных запятыми; допускается многократное (до 20 раз) использование опции в командной строке. Для выхода из режима мониторинга отдельных процессов, заданных этой опцией можно использовать интерактивную команду **=** (стр. 225).
-s	Задает безопасный режим работы программы, управляемый прежде всего с помощью конфигурационных файлов.
-S	Задает запуск программы с обращением последнего сохраненного состояния суммирования временных параметров. Для изменения режима может также использоваться интерактивная команда **S** (стр. 223).
-v	Выводит информацию о номере версии и завершает работу программы.

11.2.9.2 Формат вывода

11.2.9.2.1 Описания полей вывода

В этом параграфе описаны все поля вывода, поддерживаемые программой **top**. Эти поля всегда связаны с показанными для них символами, независимо от порядка расположения полей в реальном отчете. Для изменения порядка вывода колонок может использоваться команда **o** интерактивного режима (стр. 223).

Любое из полей может быть выбрано в качестве ключа сортировки; вы можете также задать порядок сортировки - по возрастанию или убыванию значения поля. Информация о сортировке строк приведена в параграфе 11.2.9.3.3 (стр. 223).

Таблица 43 Поля вывода программы top

Символ	Колонка	Описание
a	**PID**	Уникальный идентификатор процесса.
b	**PPID**	Идентификатор родительского процесса.
c	**RUSER**	Реальное имя пользователя, владеющего процессом.
d	**UID**	Эффективный идентификатор владеющего процессом пользователя.
e	**USER**	Эффективное имя пользователя, владеющего процессом.
f	**GROUP**	Эффективное имя группы для владельца процесса.
g	**TTY**	Имя управляющего процессом терминала. Обычно это устройство (последовательный порт, pty, и т. п.) с которого был запущен процесс и которое используется данным процессом для вывода информации. Однако задача не обязательно связана с терминалом - в таких случаях поле имеет значение **?**.
h	**PR**	Приоритет задачи.
i	**NI**	Значение nice для задачи. Отрицательные значения имеют более высокий приоритет, а положительные - более низкий. Нулевое значение говорит о том, что при диспетчеризации для данной задачи не был установлен приоритет.
j	**#C**	Номер использованного последним процессора. В корректных средах SMP значение этого поля будет часто меняться по причине распределения нагрузки между процессорами.
k	**%CPU**	Процент времени CPU, занятого данной задаче с момента предыдущего обновления информации.
l	**TIME**	Общее время процессора, использованное задачей с момента запуска. При включенном режиме суммирования для каждого процесса учитывается сумма времени, занятого данным процессом и его потомками. Для переключения режима суммирования может использоваться опция командной строки **-S** или команда интерактивного режима **S** (стр. 223).
m	**TIME+**	Это поле идентично предыдущему, но содержит более точные значения, определенные с дискретностью 10 мсек.
n	**%MEM**	Доля используемой процессом разделяемой физической памяти (ОЗУ) в процентах.
o	**VIRT**	Общее количество виртуальной памяти, используемой задачей (в килобайтах). Учитывается пространство, занимаемое кодом программы, данными и разделяемыми библиотеками, а также содержимое используемой задачей памяти, сброшенной на диск (VIRT = SWAP + RES).
p	**SWAP**	Размер (в килобайтах) образа памяти задачи, сброшенного в область подкачки.
q	**RES**	Размер (в килобайтах) резидентной области памяти задачи (RES = CODE + DATA).
r	**CODE**	Размер (в килобайтах) физической памяти системы, отведенной для программного кода задачи. Эту область памяти иногда называют TRS[1].

Символ	Колонка	Описание
s	**DATA**	Размер (в килобайтах) физической памяти системы, отведенной для целей, отличных от хранения исполняемого кода. Эту область памяти называют также DRS[1].
t	**SHR**	Размер (в килобайтах) используемой задачей разделяемой памяти системы.
u	**nFLT**	Число существенных сбоев для страниц, произошедших в процессе работы программы. Такие сбои могут быть связаны с попытками чтения или записи на виртуальные страницы, отсутствующие в адресном пространстве программы. Сбой считается существенным в тех случаях, когда для обеспечения доступности страницы приходится подключать дисковые операции.
v	**nDRT**	Количество страниц, которые были обновлены с момента последней записи на диск. Такие страницы должны быть сброшены на диск до того, как соответствующая область ОЗУ будет отдана для другой виртуальной страницы.
w	**S**	Состояние процесса: D - беспробудно спит; R - работает или поставлена в очередь на исполнение; S - спит; T - трассируется или остановлен; Z - зомби.
x	**Command**	Командная строка или имя программы в зависимости от состояния опции **c**. Переключение возможно с помощью опции **-c** в командной строке или команды интерактивного режима **c** (стр. 223). При выводе командной строки процессы, запущенные без таковой (например, потоки ядра) будут указываться с именем программы, заключенным в скобки (например, **(mdrecoveryd)**). При любом варианте значения этого поля могут усекаться в силу ограничения размера строк вывода[2].
y	**WCHAN**	В зависимости от доступности файла отображения ядра (System.map), это поле будет показывать имя или адрес (если файл недоступен) функции ядра, в которой процесс в данное время спит[3]. Для работающих задач поле содержит символ “**-**”.
z	**Flags**	Текущее значение флагов планирования для данной задачи в шестнадцатеричном формате. Описания флагов вы можете найти в файле **<linux/sched.h>**.

11.2.9.2.2 Выбор и упорядочивание колонок

Выбор колонок для отображения и управление порядком их расположения обеспечивается с подошью команд интерактивного режима **f** (Fields select) и **o** (Order fields), описанных на стр. 223. При использовании той или иной из этих команд на экран выводится список полей с их краткими описаниями, показанный на рисунке 11.6.

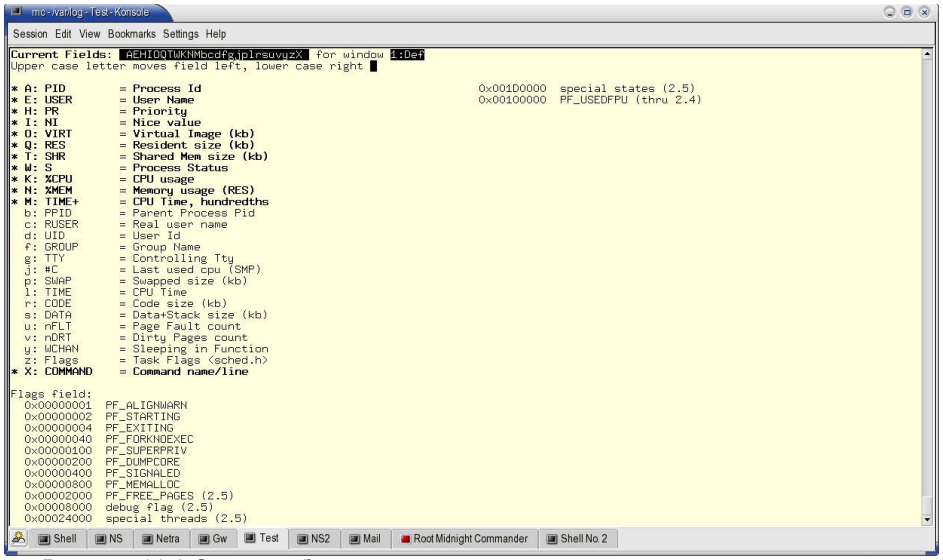


Рисунок 11.6 Экран выбора и упорядочивания полей вывода программы top

Верхняя строка экрана показывает текущий набор выводимых полей и их порядок. В режиме выбора полей прописная буква включает вывод соответствующего поля, а строчная - отключает. В режиме выбора порядка строчная буква смещает соответствующее поле влево, прописная - вправо. Нажимая соответствующие символьные клавиши в окнах выбора полей и их порядка вы можете задать удобный для вас режим отображения информации.

11.2.9.3 Интерактивное управление выводом

Ниже приводятся краткие описания команд интерактивного управления выводом, разделенных по категориям в зависимости от контекста и выполняемых задач.

1 *text resident set - область хранения исполняемого кода.*
1 *data resident set - область хранения данных и стека*
2 *Эта колонка отличается от прочиз тем, что для нее не задан фиксированный размер. При выводе на экран это поле будет занимать всю оставшуюся часть строки (максимальный размер строки составляет 512 символов).*
3 *При выводе этого поля размер используемой программой top памяти возрастает более, чем на 700 кбайт.*

11.2.9.3.1 Глобальные команды

Эти команды оказывают влияние как на полноэкранный, так и на альтернативный вариант[1] работы программы. Однако некоторые из этих команд не будут работать в безопасном режиме (**Secure mode**).

Для просмотра текущего режима работы программы нажмите клавишу **h** или **?** и прочтите информацию, представленную во второй строке.

Таблица 44 Глобальные команды интерактивного режима top

Команда	Описание
<Enter>	На самом деле эти команды не делают ничего. Однако они “будят” программу **top** и полученные после этого данные незамедлительно отображаются на экране.
<пробел>	
?	Эти команды обеспечивают доступ к интерактивной справочной системе программы. В безопасном режиме доступ к справочной системе несколько ограничен.
h	
=	Эта команда снимает ограничения, введенные для списка отображаемых задач. Команда будет обращать действие активных команд **i** (бездействующие задачи) и **n** (максимальное число задач). Кроме того, эта команда отключает режим мониторинга PID, в котором выводится информация только для заданных идентификаторами процессов. При работе программы с несколькими списками эта команда предоставляет несколько расширенные возможности.
A	Эта команда переключает между полноэкранным и альтернативным режимом работы, использующим несколько “окон” для вывода информации (см. параграф 11.2.9.4 на стр. 224).
B	Эта команда переключает режим выделения с помощью жирного (bold) шрифта в области системной информации и списке задач.
d	Эти команды служат для управления периодом обновления данных. Нулевая задержка задает постоянное обновление данных[2]. Отметим, что повышение частоты обновления увеличивает загрузку системы. Текущее значение периода обновления можно увидеть во второй строке экрана справочной информации, нажав клавишу **h** или **?**.
s	
G	Позволяет выбрать активное окно при использовании альтернативного режима отображения. При вызове команды выдается запрос на выбор номера окна (1 - 4), которое будет активизировано.
I	Выбор режима **Solaris/Irix** для отображения загрузки процессора. В режиме **Solaris** выдается значение, поделенное на число процессоров в системе.
u	Позволяет выбрать пользователя для мониторинга его процессов. Пользователя можно указать по имени или идентификатору. Данная опция выбирает процессы, соответствующие эффективному значению UID.
U	Позволяет выбрать пользователя для мониторинга его процессов. Пользователя можно указать по имени или идентификатору. Опция выбирает процессы, соответствующие по реальному, эффективному или сохраненному значению UID, а также значениям UID для файловой системы.
k	Эта команда позволяет передать указанный сигнал выбранному процессу. По умолчанию процессу передается сигнал **SIGTERM**. Для отмены команды можно нажать клавишу **<Enter>** в ответ на запрос PID или указать для сигнала код 0.
q	Завершает работу программы.
r	Эта команда позволяет настроить уровень приоритета (**nice**) для задачи. При вызове команды выдается запрос на ввод PID и значения **nice** для выбранного процесса. Положительное значение **nice** будет снижать приоритет задачи, а отрицательное - повышать.
W	Команда записи конфигурационного файла, в котором будут сохранены опции, режим отображения и период опроса. Впоследствии конфигурационный файл можно будет использовать для запуска программы **top** с желаемым режимом работы.
Z	Позволяет задать цветовой режим для текущего окна или всех окон программы. Управление цветами описано в параграфе 11.2.9.3.4 (стр. 224).

11.2.9.3.2 Команды для области системной информации (SUMMARY Area)

Команды интерактивного управления доступны как в полноэкранном, так и в альтернативном режиме отображения. Эти команды управляют выводом информации в верхних строках окна, а также положением строки сообщений и ввода параметров.

Таблица 45 Команды интерактивного управления для системной области

Команда	Описание
l	Эта команда служит для управления выводом строки **Load Average/Uptime**.
m	Эта команда служит для управления выводом строк **Memory/Swap Usage**.

1 *С несколькими списками задач.*
2 *Частота обновления ограничивается возможностями используемого терминала.*

Команда	Описание
t	Команда управления выводом строк **Task/Cpu State**.
1	Команда выбора режима **Single/Separate Cpu** для вывода информации по всем процессорам или по каждому CPU отдельно.

Можно полностью исключить вывод строк в области системной информации (за исключением строки сообщений и ввода данных). Это позволяет увеличить размер списка задач.

11.2.9.3.3 Команды для списка задач

Команды управления списком задач всегда доступны для полноэкранного режима работы. В альтернативном многосписочном режиме эти команды не будут доступны, если текущее окно находится в пассивном режиме (**Off**).

11.2.9.3.3.1 Представление списка задач

Доступность команд управления видом списка задач зависит от глобальной опции **B** (стр. 222).

Таблица 46 Команды интерактивного управления для представления списка задач

Команда	Описание
b	Команда переключения режима выделения полей **Bold/Reverse** (шрифт/цвет).
x	Команда управления выделением колонки ключа сортировки. При включенном режиме соответствующая колонка выводится жирным шрифтом или в негативном представлении, в зависимости от выбранного режима выделения.
y	Команда управления выделением в списке строк активных задач. При включенном режиме соответствующие строки выводятся жирным шрифтом или в негативном представлении, в зависимости от выбранного режима выделения.
z	Управляет режимом цвета **Color/Monochrome**. В режиме цветного вывода используется выбранная пользователем схема цветов (см. параграф 11.2.9.3.4 на стр. 224).

11.2.9.3.3.2 Управление содержимым списка задач

Таблица 47 Команды интерактивного управления содержимым списка задач

Команда	Описание
c	Определяет вывод в колонке **Command** имени программы или полной строки команды.
f	Команды управления набором и порядком расположения колонок в списке задач (см. параграф 11.2.9.2.2 на стр. 221).
o	
S	Команда переключения режима суммирования (**Cumulative mode**) процессорного времени, занятого задачей и порожденными ею процессами, для вывода в колонках **TIME** и **TIME+**. При выключенном режиме суммирования задача, поделенная на много ветвей (процессов) может быть представлена некорректно, поскольку не будет показан расход процессорного времени дочерними процессами. Текущее состояние режима суммирования можно посмотреть в справочном окне, нажав для этого клавишу **?** или **h**.
u	Эта команда позволяет выбрать процессы определенного пользователя, заданного именем или UID. Для возврата в режим просмотра всех задач достаточно повторно нажать клавишу **u** и в ответ на запрос имени или идентификатора пользователя просто нажать клавишу **<Enter>**.

11.2.9.3.3.3 Управление размером списка задач

Таблица 48. Команды интерактивного управления размером окна списка задач.

Команда	Описание
i	Эта команда меняет режим включения в список задач строк для бездействующих (Idle) процессов. При выключенном режиме процессы, находящиеся в состоянии **idle** или **zombie**, не будут помещаться в список. В альтернативном режиме команда влияет только на список задач активного окна.
n	Эти команды служат для ограничения числа строк в списке задач. Для возврата к неограниченному списку нажмите клавишу **n** или **#** повторно и укажите в качестве числа задач значение 0. В альтернативном режиме команда влияет только на список задач активного окна.
#	

11.2.9.3.3.4 Управление сортировкой задач в списке

В целях совместимости со старыми версиями программа **top** поддерживает старые команды управления ключами сортировки (таблица 49). Однако эти команды не указываются в окне справочной информации.

Новая версия поддерживает набор команд интерактивного управления сортировкой строк списка процессов. Отметим, что поле ключа сортировки может быть выделено цветом или шрифтом (команда **x**).

Перечисленные в таблице 50 команды управления сортировкой будут работать только при включенном выделении ключевого поля. Не забывайте, что ключевое поле может отсутствовать на экране по двум причинам:

1) недостаточная ширина экрана для вывода всех выбранных полей;
2) поле отключено с помощью команды f (см. параграф 11.2.9.2.2)

Таблица 49. Старые команды управления сортировкой списка top.

Команда	Ключ сортировки	Поддержка
A	Время запуска (не выводится)	Нет
M	%MEM	Есть
N	PID	Есть
P	%CPU	Есть
T	TIME+	Есть

Таблица 50. Команды перемещения ключа сортировки списка задач.

Команда	Описание
<	Перемещает ключ сортировки на предыдущую (расположенную слева) колонку. Если в настоящий момент сортировка осуществляется по первой колонке, команда не делает ничего.
>	Переносит ключ сортировки в следующую (расположенную справа) колонку. Если сортировка осуществляется по последней колонке, команда не делает ничего.

Команды управления сортировкой, перечисленные в таблице 51 работают независимо от состояния выделения ключевого поля.

Таблица 51 Команды интерактивного управления сортировкой списка задач

Команда	Описание
F	Выбор поля для использования в качестве ключа сортировки. Если выбранное поле не было ранее включено в число отображаемых, оно будет автоматически включено в это число. Однако (в зависимости от ширины экрана и порядка расположения колонок), это поле может остаться невидимым.
O	
R	Переключает порядок сортировки по выбранному ключу (по возрастанию или убыванию).

11.2.9.3.4 Цветовое выделение

Команда интерактивного управления **Z** позволяет переключаться в цветной режим отображения. При вызове этой команды на экране появляется окно выбора цвета, позволяющее задать цвета отображения (см. рисунок 11.7). Это окно можно использовать для установки цветовых параметров областей вывода и “окон” альтернативного режима.

11.2.9.4 Альтернативный режим отображения

В альтернативном режиме вместо одного списка процессов выводится несколько (от 1 до 4) “окон”, содержащих различные списки процессов, отличающиеся набором полей, ключами сортировки и т. п.

11.2.9.4.1 Окна альтернативного режима

В полноэкранном режиме программа использует один список задач, а в альтернативном режиме можно создать от 1 до 4 групп, каждая из которых будет иметь независимо настраиваемую системную область (summary area) и свой список отображаемых задач.

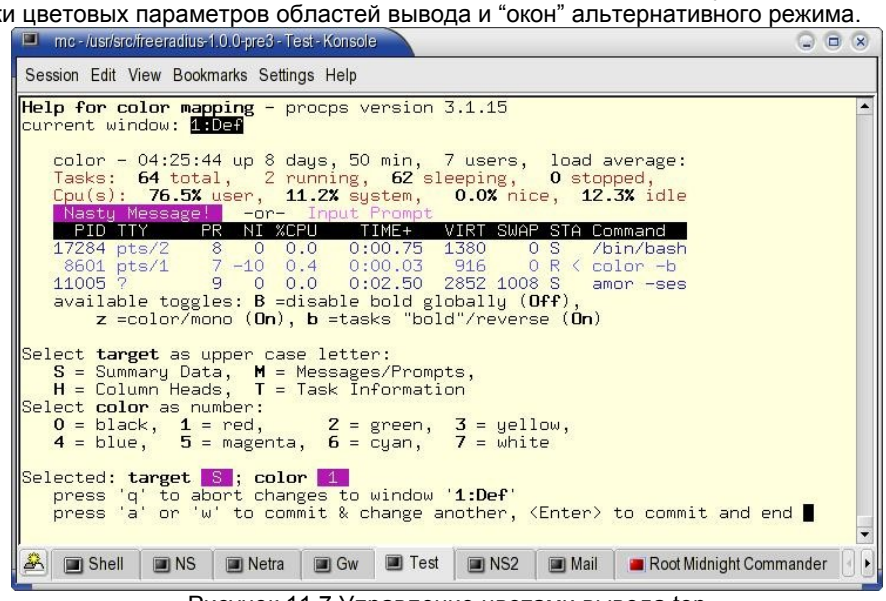


Рисунок 11.7 Управление цветами вывода top

Все группы в альтернативном режиме могут выводиться на экран одновременно (одна группа выбирается в качестве текущей), а при необходимости можно отключить вывод для любой из групп независимо. Системная область всегда выводится на экран, даже при наличии в ней только строки сообщений и ввода. В альтернативном режиме системная область выводится для текущей группы.

Текущее окно воспринимает все команды интерактивного управления. Поскольку в альтернативном режиме вывод списка задач можно отключить, некоторые команды для текущего окна могут не работать. Если вы отключите вывод первой строки системной области (команда **l**), то потеряете информацию о текущем окне, поскольку она выводится в начале первой строки.

11.2.9.4.2 Команды управления окнами

Команда	Описание
- **_**	Команда **-** переключает режим отображения текущей группы (включает или выключает вывод списка задач группы). Команда **_** переключает между текущим набором включенных групп и выключенной группой. Если включен вывод всех групп, команда переключает в режим вывода “все - ни одной”.
= **+**	Команда **=** активизирует вывод списка задач текущего окна и меняет состояние активных команд **i** (бездействующие задачи) и **n** (максимальное число задач в списке). Команда **+** выполняет те же операции для всех групп.
A	Эта команда переключает между полноэкранным и групповым режимом отображения. При первом включении альтернативного режима будут выводиться все 4 группы, а при последующих переключениях в альтернативный режим - только включенные группы.
a **w**	Переключают между группами альтернативного режима отображения, делая текущей следующую (**a**) или предыдущую (**w**) группу.
G	Команда позволяет выбрать текущую группу для альтернативного режима вывода, указав номер нужной группы. Команда работает и в полноэкранном режиме, меняя список задач в соответствии с настройками выбранной группы.
g	Эта команда позволяет изменить имя группы для альтернативного режима вывода.

11.2.9.5 Конфигурационные файлы

11.2.9.5.1 Системный файл конфигурации

Системный файл конфигурации **top** позволяет ограничить возможности пользователей при работе с программой, блокируя команды **k** (послать сигнал процессу), **r** (изменить значение nice), **d** и **s** (изменить период опроса).

Программа **top** не создает конфигурационный файл - вы должны создать его самостоятельно и поместить в каталог **/etc**. Файл конфигурации должен использовать имя **toprc** и включать только две строки:

```
s          # переход в безопасный режим (secure mode)
5.0        # интервал опроса в секундах
```

11.2.9.5.2 Персональный файл конфигурации

Персональный файл конфигурации может быть записан с помощью команды интерактивного управления **W**. Файл может содержать глобальные параметры конфигурации и настройки для каждой группы альтернативного режима. Пример персонального файла конфигурации показан на рисунке 11.8.

```
RCfile for "top with windows"           # shameless braggin'
Id:a, Mode_altscr=0, Mode_irixps=1, Delay_time=3.000, Curwin=0
Def     fieldscur=AEHIOQTWKNMbcdfgjplrsuvyzX
        winflags=62776, sortindx=10, maxtasks=0
        summclr=1, msgsclr=1, headclr=3, taskclr=1
Job     fieldscur=ABcefgjlrstuvyzMKNHIWOPQDX
        winflags=62776, sortindx=0, maxtasks=0
        summclr=6, msgsclr=6, headclr=7, taskclr=6
Mem     fieldscur=ANOPQRSTUVbcdefgjlmyzWHIKX
        winflags=62776, sortindx=13, maxtasks=0
        summclr=5, msgsclr=5, headclr=4, taskclr=5
Usr     fieldscur=ABDECGfhijlopqrstuvyzMKNWX
        winflags=62776, sortindx=4, maxtasks=0
        summclr=3, msgsclr=3, headclr=2, taskclr=3
```

Рисунок 11.8 Персональный файл конфигурации top

11.2.10 Uptime

Команда **uptime** показывает сведения о текущем состоянии системы. Строка вывода команды включает:

- текущее время;
- продолжительность работы системы;
- количество зарегистрированных пользователей;
- средняя загрузка системы за последние 1, 5 и 15 минут.

Эти сведения также включаются в первую строку вывода команды **w** (см. следующий параграф).

11.2.11 Vmstat

Команда **vmstat** выводит статистику использования физической (ОЗУ) и виртуальной (файл подкачки) памяти системы, а также статистику дисковых операций.

Синтаксис

```
vmstat [-a] [-n] [<задержка> [<счетчик>]]
vmstat [-f] [-s] [-m]
vmstat [-S <k|K|m|M>]
vmstat [-d]
```

```
vmstat [-p <раздел>]
vmstat [-V]
```

Программа выводит сведения о процессах, памяти, страницах, блоках адресов ввода-вывода, ловушках и активности процессора. При запуске программы выводятся данные, усредненные за время с момента загрузки системы. Далее можно просматривать информацию, собранную за интересующий период.

11.2.11.1 Опции

Таблица 52. Опции команды vmstat.

Опция	Описание
-a	Выводит сведения об активной/неактивной памяти вместо буферов и кэша.
-f	Количество ветвлений (fork) с момента загрузки системы. Учитываются все системные вызовы **fork**, **vfork** и **clone**, поэтому подсчет показывает точное количество созданных задач. Для этой команды не поддерживается режим повторяющихся опросов.
-m	Выводит содержимое файла **slabinfo** (см. стр. 356).
-n	Задает однократный вывод строки заголовка.
-s	Выводит таблицу счетчиков событий и статистику памяти. Для этой команды не поддерживается режим повторяющихся опросов.
<задержка>	Задает интервал повторных опросов (в секундах). Если этот параметр не указан в командной строке, программа выводит единственный отчет в соответствии с заданными опциями. При использовании параметра **delay** для завершения программы используйте клавиши **Ctrl**+**C** или указывайте количество повторов с помощью параметра **<счетчик>**.
-d	Выводит статистику использования дисков для каждого устройства.
-D	Выводит таблицу статистики для всех дисков системы. Для этой команды не поддерживается режим повторяющихся опросов.
-p <раздел>	Выводит статистику указанного раздела диска[1].
-S <k|K|m|M>	Задает единицы, в которых указываются значения счетчиков[2].
-V	Выводит номер версии программы

11.2.11.2 Описания полей вывода

Набор полей вывода зависит от выбранных опций.

11.2.11.2.1 Процессы и память

```
procs -----------memory---------- ---swap-- -----io---- --system-- ----cpu----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in    cs us sy id wa
 3  0 493560  11852   4016 115056    5    9    23    26  167     5 19  2 78  1
```

Рисунок 11.9 Сведения о процессах и памяти

Информация о процессах и памяти выводится при использовании команды без опций или с опцией **-a**. Формат вывода показан на рисунке 11.9.

procs

r - число процессов, ожидающий выделения времени CPU;

b - число беспробудно уснувших процессов.

memory

swpd - объем используемой виртуальной памяти;

free - объем свободной памяти;

buff - объем памяти, используемой для буферов;

cache - объем памяти, используемой для кэширования;

inact - объем неактивной памяти (опция **-a**);

active - объем активной памяти (опция **-a**).

swap

si - объем памяти прочитанной из области подкачки (за секунду);

so - объем памяти сброшенной на диск (за секунду).

io

bi - количество блоков, полученных с устройства (за секунду);

1 *При указании раздела префикс следует опускать (hda8, а не /dev/hda8).*
2 *k = 1000, K = 1024, m = 1000000, M = 1048576.*

bo - количество блоков, записанных на устройство (за секунду).

system

in - число прерываний за секунду с учетом системного таймера;

cs - число переключений контекста за секунду.

CPU

us - время, затраченное процессами, не входящими в ядро (с учетом **nice**);

sy - время, затраченное процессами ядра;

id - время бездействия;

wa - время ожидания операций ввода-вывода.

Время процессора указывается в процентах.

11.2.11.2.2 Дисковые операции (-d)

```
disk ----------reads------------ -----------writes-----------   -------IO-------
     total merged sectors     ms  total merged sectors     ms      cur     s
ram0       0      0       0      0      0      0       0      0        0      0
hda   472025 245135 11395064 5259537 632081 691553 13042264 71287608  0   3410
md0        0      0       0      0      0      0       0      0        0      0
hdc      688   3972   18832  46840      0      0       0      0        0     46
fd0        4      0       8    164      0      0       0      0        0      0
```

Рисунок 11.10 Сведения о дисковых операциях

Информация о дисковых операциях выводится для всех дисков системы в формате, показанном на рисунке 11.10.

reads

total - общее количество успешных операций чтения диска;

merged - количество групповых операций чтения;

sectors - число прочитанных секторов;

ms - время, затраченное на операции чтения диска (в миллисекундах).

writes

total общее количество успешных операций записи;

merged - количество групповых операций записи;

sectors - число записанных секторов;

ms - время, затраченное на операции записи (в миллисекундах).

io

cur - число активных операций ввода-вывода;

s - затраты времени на операции ввода-вывода (в секундах).

11.2.11.2.3 Дисковые разделы (-p)

```
hda7          reads   read sectors  writes    requested writes
              131005    3171722     441244    5518160
```

Рисунок 11.11 Сведения о разделе диска

Информация о разделе диска выводится в формате, показанном на рисунке 11.11.

reads - число операций чтения для данного раздела;

read sectors - число прочитанных с данного раздела секторов;

writes - число операций записи для данного раздела;

requested writes - число запросов на запись для данного раздела.

11.2.11.2.4 Именованные блоки памяти

Формат вывода информации об именованных блоках памяти (опция **-m**) показан на рисунке 11.12.

Cache	Num	Total	Size	Pages
isofs_inode_cache	5	22	352	11
fib6_nodes	5	113	32	113
ip6_dst_cache	5	17	224	17
ndisc_cache	1	24	160	24
raw6_sock	0	0	640	6
udp6_sock	0	0	608	6
tcp6_sock	5	7	1088	7
ip_fib_hash	20	203	16	203
fat_inode_cache	0	0	384	10
uhci_urb_priv	4	84	44	84
scsi_cmd_cache	1	11	352	11
sgpool-128	32	32	2048	2
sgpool-64	32	32	1024	4
sgpool-32	32	32	512	8
sgpool-16	32	45	256	15
sgpool-8	32	60	128	30
rpc_buffers	8	8	2048	2
rpc_tasks	8	24	160	24
rpc_inode_cache	0	0	416	9
unix_sock	161	176	352	11
ip_mrt_cache	0	0	96	40
tcp_tw_bucket	0	0	128	30
tcp_bind_bucket	25	203	16	203
tcp_open_request	0	0	96	40
inet_peer_cache	2	59	64	59
secpath_cache	0	0	128	30

Рисунок 11.12 Сведения об именованных блоках памяти

cache - имя блока;

num - число активных в данный момент объектов;

total - общее число доступных объектов;

size - размер каждого объекта;

pages - число страниц, содержащих по крайней мере один активный объект;

totpages - общее число выделенных страниц;

pslab - число страниц на именованный блок памяти (slab).

Программа **vmstat** получает информацию из файлов **/proc**(см. Приложение 12.2.1), поэтому для использования этой программы не требуется каких-либо привилегий.

11.2.12 w

Команда **w** выводит сведения о зарегистрированных в системе пользователях и некоторые сведения о работе системы общего характера.

Синтаксис

```
w - [husfV] [user]
```

Команда **w** выводит сведения о зарегистрированных в системе пользователях и процессах этих пользователей. Первая строка вывода включает общие сведения о работе системы, совпадающие с выводом команды **uptime**, описанной в предыдущем параграфе. Далее выводится информация о пользователях, снабженная строкой заголовка с полями:

USER - имя пользователя;

TTY - терминал, применяемый пользователем для доступа в систему;

FROM - хост, с которого пользователь зарегистрировался в данной системе;

LOGIN@ - время регистрации пользователя в системе;

IDLE - время бездействия;

JCPU - системное время, использованное всеми процессами данного терминала (без учета прошлых фоновых задач, но с учетом текущей фоновой задачи);

PCPU - время, использованное текущим процессом (указан в поле **WHAT**);

WHAT - текущий процесс пользователя.

11.2.12.1 Опции

Таблица 53. Опции команды w.

Опция	Описание
`-h`	Отключает вывод строки заголовка.
`-u`	Задает игнорирование пользователя при выводе информации о текущем процессе и времени CPU.
`-s`	Задает краткий формат вывода без указания имени пользователя и полей **JCPU**, **PCPU**.
`-f`	Отключает вывод поля **FROM** (удаленный хост).
`-V`	Выводит сведения о версии программы.
`user`	Задает вывод информации только для указанного пользователя.

11.3 Система удаленного управления Webmin

http://www.webmin.com

Пакет Webmin представляет собой набор сценариев Perl, используемых на специальном сервере HTTP и обеспечивающих Web-интерфейс для настройки, управления и мониторинга большинства параметров и служб UNIX-станций. Программы настройки работают со стандартными конфигурационными файлами системы и не

вносят в эти файлы никаких недокументированных параметров, поэтому настройка с помощью Webmin никогда не конфликтует с другими программами настройки конфигурации.

Программа имеет модульную структуру (см. рисунок 11.13), а модули собраны в группы:

- **Webmin** - модули настройки и управления для программы;
- **System** - модули для настройки, управления и мониторинга общесистемных функций и параметров;
- **Servers** - модули для настройки управления и мониторинга отдельных служб;
- **Networking** - модули для настройки и управления сетевыми службами;
- **Hardware** - модули для настройки оборудования;
- **Cluster** - модули для настройки и мониторинга UNIX-кластеров;
- **Others** - прочие модули.

Каждый модуль можно независимо настраивать, удалять и клонировать[1]. Обычно модуль включает в себя группу функций настройки, управления и мониторинга различными аспектами работы UNIX-систем. Каждый модуль включает собственный конфигурационный интерфейс, что позволяет настроить Webmin даже для работы с нестандартными инсталляциями.

Кроме входящих в базовый комплект модулей Webmin существует также множество модулей сторонних разработчиков, которые также доступны на сайте http://www.webmin.com.

Программа обычно запускается от имени пользователя **root** или **admin** и обеспечивает оператору большие полномочия. Поэтому следует предотвратить доступ к программе пользователей сети и злоумышленников. Модули **Webmin Configuration** и **Usermin Configuration** позволяют настроить глобальную систему контроля доступа к функциям сервера Webmin, а модуль **Webmin Users** позволяет создавать новых пользователей Webmin и указывать модули и функции, к которым каждый пользователь будет иметь доступ. Пользователей Webmin можно объединять в группы.

1 *Это может оказаться весьма удобным при использовании на станции нескольких экземпляров одного приложения. В этом случае для каждого экземпляра можно создать свою копию модуля Webmin.*

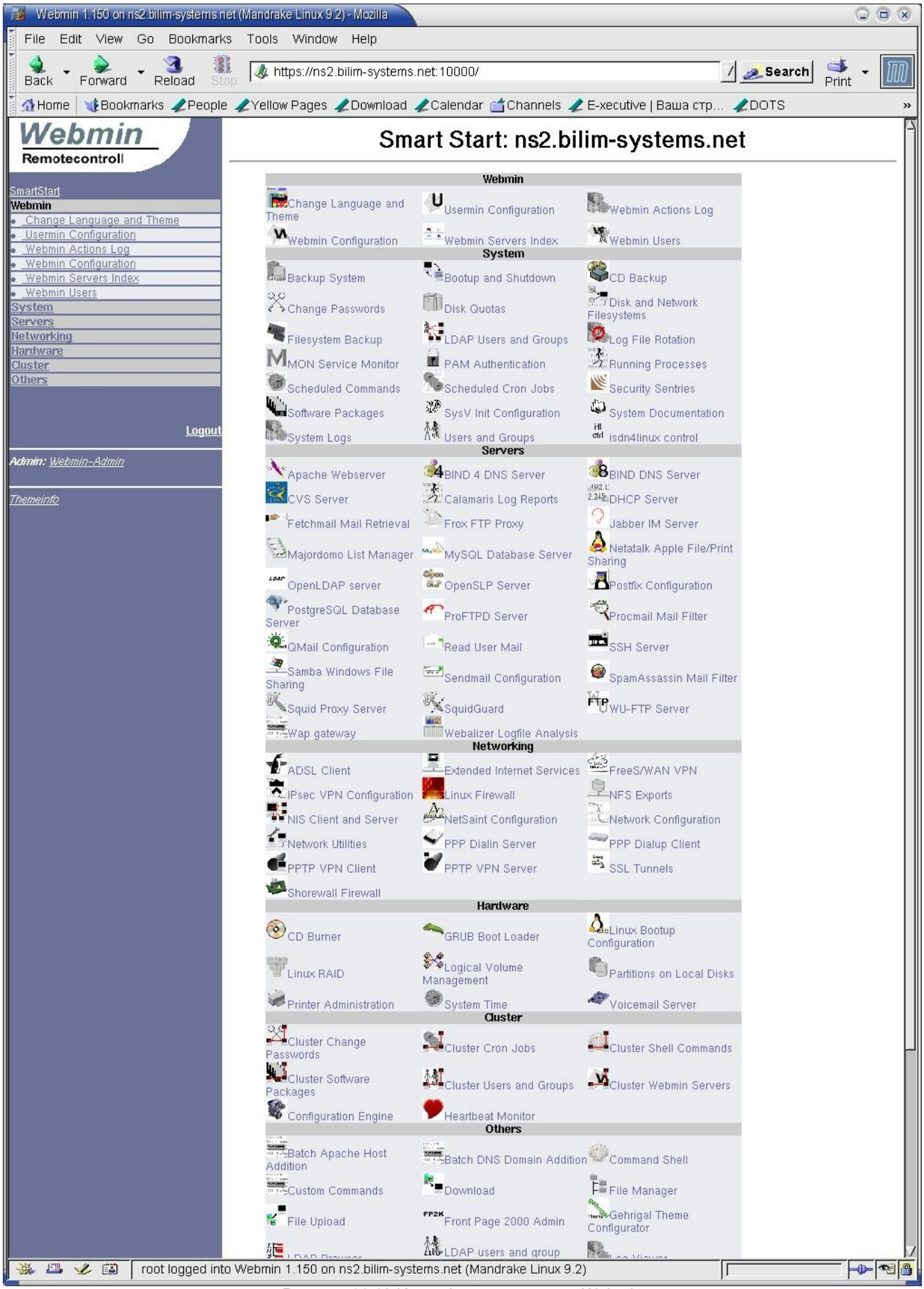


Рисунок 11.13 Интерфейс программы Webmin

Ниже более подробно рассматриваются некоторые модули Webmin, используемые для настройки тех или иных компонент системы обеспечения безопасности.

11.4 Сканеры безопасности

11.4.1 COPS

ftp://ftp.cerias.purdue.edu/pub/tools/unix/scanners/cops/

11.4.2 SATAN

http://www.fish.com/satan

11.4.3 Internet Security Scanner (ISS)

ftp://ftp.cerias.purdue.edu/pub/tools/unix/scanners/iss/

11.4.4 Nessus

http://www.nessus.org

Сканер безопасности Nessus состоит из двух основных компонент - серверной (демон **nessusd**) и клиентской (графический интерфейс **nessus**).

11.4.4.1 Сервер nessusd

Сервер **nessusd** обеспечивает организацию атак для тестирования безопасности хоста. Управление атаками осуществляется с помощью клиентского интерфейса программы **nessus** (параграф 11.4.4.4 на стр. 234). Серверный модуль **nessusd** проверяет заданный хост и выводит список обнаруженных уязвимостей и некорректных параметров конфигурации удаленного или локального хоста.

Синтаксис

```
nessusd [-v] [-h] [-c config-file] [-a address] [-p port-number] [-D] [-d]
```

11.4.4.1.1 Опции nessusd

Таблица 54. Опции nessusd.

Опция		Описание
`-c`	`--config-file=`	Задает использование указанного конфигурационного файла взамен принятого по умолчанию файла **/etc/nessus/nessusd.conf**.
`-a`	`--listen=`	Задает IP-адрес, которых будет прослушивать сервер. Пакеты, направленные на другие адреса данного хоста, сервер не будет воспринимать. Эта опция полезна при использовании программы на шлюзах со множеством интерфейсов, если вы не хотите, чтобы к вашему серверу **nessusd** подключались извне.
`-p`	`--port=`	Задает номер порта, который будет прослушивать сервер. По умолчанию используется порт 1241.
`-D`	`--background`	Задает работу сервера в режиме демона.
`-d`	`--dump-cfg`	Задает вывод конфигурационных опций сервера.
`-v`	`--version`	Задает вывод номера версии и завершение работы программы.
`-h`	`--help`	Задает вывод справочной информации и завершение работы программы.

11.4.4.1.2 Конфигурационный файл сервера

Конфигурационный файл сервера (по умолчанию **/etc/nessusd.conf**) содержит перечисленные в таблице 55 параметры конфигурации сервера.

*Таблица 55 Параметры конфигурации в файле **nessusd.conf***

Параметр	Описание
`plugins_folder`	Имя каталога, хранящего подключаемые модули (обычно **/usr/lib/nessus/plugins/**).
`logfile`	Полное имя журнального файла сервера. Если вы планируете записывать сообщения **nessusd** с помощью демона **syslogd**, можно указать **syslog** вместо имени файла. Вы можете также указать **stderr** для записи сообщений **nessusd** в устройство **stderr**. Поскольку сообщения **nessusd** достаточно важны, имеет смысл сохранять их в отдельном файле.

Параметр	Описание
max_hosts	Задает максимальное количество хостов для одновременной проверки. Клиент может изменить это значение. При выборе параметра следует учитывать возможности сетевого соединения, наличие памяти и производительность процессора. Задать число проверяемых одновременно хостов возможно также с помощью пользовательского интерфейса **nessus** (см. параграф 11.4.4.4.2.4 на стр. 236).
max_checks	Определяет число подключаемых модулей (plugin), которые могут одновременно использоваться для проверки каждого хоста[1]. При выборе параметра следует учитывать возможности сетевого соединения, наличие памяти и производительность процессора. Отметим, что одновременное использование множества тестов может привести к тому, что удаленный хост просто заблокирует пакеты с вашего сервера. Задать число проверяемых одновременных соединений возможно также с помощью пользовательского интерфейса **nessus** (см. параграф 11.4.4.4.2.4 на стр. 236).
be_nice	Если этот параметр имеет значение **yes**, каждый дочерний процесс **nessusd** будет устанавливать для себя низкий уровень приоритета с помощью nice. Это может ускорить сканирование, поскольку основной процесс **nessusd** будет способен продолжать порождение новых процессов. Кроме того, это значение гарантирует, что **nessusd** не отнимет ресурсы у других важных процессов в системе.
log_whole_attack	При выборе значения **yes** для этой опции **nessusd** будет сохранять **имя**, **pid**, **дату** и **цель** для каждого подключенного модуля. Это может быть полезно для мониторинга и отладки, но ведет к быстрому заполнению диска информацией **nessusd**.
log_plugins_name_at_load	При выборе значения **yes** для этой опции **nessusd** будет записывать в журнальный файл информацию об именах подключаемых модулей.
port_range	Задает диапазон портов, которые модули сканирования будут использовать по умолчанию. Вы можете задавать диапазоны портов или отдельные номера, разделяя их запятыми. Можно выделить порты UDP префиксом **U:**, а порты TCP - префиксом **T:**. Например, для сканирования портов UDP с 1 по 1024 и портов TCP с 1 по 65535 опция будет иметь вид: **T:1-65535,U:1-1024**. Задать диапазон сканируемых портов можно также с помощью пользовательского интерфейса **nessus** (см. параграф 11.4.4.4.2.4 на стр. 236).
optimize_test	По умолчанию **nessusd** не доверяет представлению удаленными хостами самих себя. Это значит, что Web-сервер, представившийся как IIS, будет проверяться и на уязвимости Apache и т. д. При таком подходе могут возникать ложные срабатывания и снизится скорость проверки. Если вы уверены в том, что сервер корректно представляет себя, вы можете воспользоваться этой опцией для повышения уровня достоверности и быстродействия. В этом случае будут использоваться только те модули, которые предназначены для проверки сообщенного сервером типа.
checks_read_timeout	Время ожидания (в секундах) при вызове функции **recv** для проверки безопасности. При использовании **nessusd** на медленных каналах это значение следует увеличить.
non_simult_ports	Некоторые службы (в частности, SMB) не принимают множества одновременных обращений от одного хоста. Эта опция позволяет предотвратить одновременные соединения **nessusd** с одним портом проверяемого хоста. Значением параметра является список портов (port1[, port2....]), для которых не следует допускать попыток одновременных соединений. Список может содержать не только номера портов, но и их имена. Например, запись **139, Services/www** будет предотвращать одновременные соединения с портом 139 и всеми портами, которые указаны как порты **www** в файле **/etc/services**.
plugins_timeout	Задает максимальное время работы plugin-модуля. Некоторые модули могут оказаться весьма медленными по своему устройству или по причине задержек откликов от проверяемого хоста. Данная опция позволяет ограничить время работы модуля для предотвращения слишком долгого сканирования хоста. Клиентский интерфейс **nessus** позволяет установить максимальное время выполнения проверки независимо для каждого модуля (см. стр. 235).
safe_checks	Большую часть времени сервер **nessusd** пытается создать исключительные условия для определения уязвимости проверяемого хоста. Сюда включаются попытки переполнения буферов, строки с некорректным форматом и другие тесты, которые могут привести к нарушению работы проверяемого сервера. Выбрав для этой опции значение **yes**, вы отключите использование потенциально опасных модулей. Это ускоряет тестирование и снижает нагрузку, но может привести к пропуску некоторых важных уязвимостей. Включить режим использования только безопасных модулей можно также непосредственно перед тестированием хоста с помощью клиентского интерфейса **nessus** (см. параграф 11.4.4.4.2.4 на стр. 236).

1 *Число процессор в этом случае будет равно* ***max_checks*** * ***max_hosts***

Параметр	Описание
auto_enable_dependencies	Модули **nessusd** используют результаты работы предыдущих модулей при выполнении своих тестов. Например, модулю, который подключается к удаленному реестру SMB, будут нужны результаты работы модуля, определившего SMB-имя удаленного хоста, и модуля, пытавшегося подключиться к этому хосту. Если вы хотите использовать только часть модулей, эта опция позволит быстро подобрать связанные между собой модули. При выборе значения **yes** сервер **nessusd** будет автоматически активизировать модули, которые требуются для работы выбранных вами модулей. Клиентский интерфейс также позволяет управлять режимом автоматической загрузки связанных модулей (см. стр. 235)
use_mac_addr	Установите для этой опции значение **yes**, если вы тестируете хосты локальной сети и каждая машина использует динамический адрес IP. В этом случае результаты тестирования будут привязываться к MAC-адресам хостов.
plugin_upload	Выбор для этой опции значения **yes** позволит серверу **nessusd** выгружать модули из памяти после завершения работы с ними.
admin_user	Пользователи, указанные в списке этой опции, могут загружать свои модули в глобальный каталог plugin-модулей **nessus**.
rules	Указывает путь к файлу правил.

Остальные опции сканирования можно изменять с помощью клиентского интерфейса.

11.4.4.2 Управление пользователями

Для добавления пользователей сервера **nessusd** служит утилита **nessus-adduser**. Каждый пользователь **nessusd** получает свой каталог в каталоге **users/**. Этот каталог содержит набор подкаталогов, перечисленных в таблице .

Таблица 56. Каталоги пользователей nessusd.

Каталог	Назначение
auth/	Этот каталог содержит данные для аутентификации пользователя. Он включает файл **dname**, если при аутентификации пользователя применяется сертификат, или файл **hash** (или **passwd**), если используется аутентификация по паролю. Файл **hash** содержит зашифрованный с помощью алгоритма MD5 пароль пользователя вместе со случайным числом. Файл **password** содержит пароль в незашифрованном виде. Этот каталог включает также файл **rules**, которые содержит применимые для этого пользователя правила. Содержимое этого каталога недоступно пользователю.
kbs/	Каталог, содержащий базу знаний (KB) для каждого протестированного пользователем хоста, если пользователь в настройках клиента активизировал опцию **save_kb** (параграф на стр.).
sessions/	Этот каталог включает список содержимого пользовательских сеансов.
plugins/	В этом каталоге хранятся пользовательские plugin-модули.

При подключении пользователя к серверу **nessusd** пытается проверить наличие пользовательского каталога **users/<username>** и после его обнаружения шифрует полученный от пользователя пароль с помощью случайного значения из файла **<username>/auth/hash**, сравнивая результат с зашифрованным паролем из того же файла. Если для аутентификации пользователя служит сертификат, **nessusd** проверяет подлинность этого сертификата.

Для удаления пользователей служит утилита **nessus-rmuser**.

11.4.4.3 Формат файла правил

Для управления возможностями сканирования тех или иных хостов служат правила, определяющие доступные для сканирования диапазоны адресов. Все правила используют простой формат вида:

```
<ключевое слово> <IP-адрес>/<маска>
```

Пользователь может создавать для себя отдельный набор правил, однако эти правила могут лишь сужать разрешенные сервером диапазоны сканирования.

11.4.4.3.1 База правил nessus

База правил программы содержит набор глобальных правил, действующих для каждого пользователя. Синтаксис правил был определен в предыдущем параграфе. Пример правил показан ниже.

```
accept 127.0.0.0/8
deny 192.168.1.1/32
deny !192.168.0.0/16
default deny
```

Этот набор правил разрешает сканирование локального хоста и всех хостов сети **192.168.0.0/16**, за исключением хоста **192.168.1.1/32**. В правилах можно использовать зарезервированное слово **client_ip**, которое в момент соединения заменяется IP-адресом компьютера, с которого пользователь обратился к серверу **nessusd**. Например приведенные ниже правила разрешат пользователю сканирование только его компьютера.

```
accept client_ip/32
```

```
default deny
```

11.4.4.4 Клиент nessus

Программа **nessus** работает как клиент X11 на базе GTK (Gimp ToolKit). Интерфейс программы описан ниже.

Синтаксис

```
nessus [-v] [-h] [-n] [-T <type>] [-q [-pPS] host port user password targets results]
nessus -i in.[nsr|nbe] -o out.[html|xml|nsr|nbe]
```

11.4.4.4.1 Опции

Таблица 57 Опции nessusd

Опция		Описание
`-c`	`--config-file=`	Задает использование указанного конфигурационного файла взамен принятого по умолчанию файла **/etc/nessus/nessusd.conf**.
`-n`	`--no-pixmaps`	Отключает вывод изображений в окне программы. Эта опция полезна при работе с удаленными клиентами.
`-q`	`--batch-mode`	Задает работу программы в пакетном режиме без использования графического интерфейса. Перечисленные в таблице 58 опции позволяют задать параметры сканирования в пакетном режиме.
`-T`	`--output-type=`	Формат данных при сохранении результатов - **nbe**, **html**, **html_graph**, **text**, **xml**, **old-xml**, **tex** или **nsr**.
`-V`	`--verbose`	Задает вывод статусных сообщений для пакетного режима работы.
`-x`	`--dont-check-ssl-cert`	Отключает проверку сертификатов SSL.
`-v`	`--version`	Задает вывод номера версии и завершение работы программы.
`-h`	`--help`	Задает вывод справочной информации и завершение работы программы.

В таблице 58 перечислены опции управления пакетным режимом работы программы (опция **-q**).

Таблица 58 Опции управления пакетным режимом

`-p`	Задает получение от сервера списка установленных plugin-модулей.
`-P`	Задает получение предпочтительных установок для сервера plugin-модуля.
`-S`	Задает для опций **-p** и **-P** вывод в формате SQL с полями: **host** - имя хоста **nessusd**, к которому выполняется подключение; **port** - порт удаленного хоста, через который осуществляется подключение к серверу **nessusd**; **user** - имя пользователя для подключения к серверу **nessusd**; **password** - пароль пользователя.

11.4.4.4.2 Графический интерфейс программы

В следующих параграфах кратко описаны основные элементы графического интерфейса клиента **nessus**. Окно программы состоит из нескольких панелей, описанных ниже. Каждая из панелей содержит ряд специфических элементов управления, связанных с отдельными аспектами использования программы.

Основные кнопки управления процессом сканирования выбранного хоста присутствуют в нижней части окна для каждой панели.

Кнопка	Действия
`Start the scan`	Активизирует процесс сканирования выбранного хоста в соответствии с установленными параметрами проверки.
`Load report`	Используется для загрузки и просмотра сохраненного отчета о предыдущем сканировании того или иного хоста.
`Quit`	Используется для завершения работы программы.

11.4.4.4.2.1 Панель Nessusd host

При запуске клиентской программы nessus вы автоматически попадаете в стартовую панель **Nessus host** (см. рисунок 11.14), которая позволяет указать сервер **nessusd**, с которым вы планируете работать, и номер порта для подключения к серверу.

Поле **Login** служит для ввода имени пользователя, а поле **Password** - для ввода пароля. После набора имени и пароля вы можете нажать кнопку **Log in** для подключения к серверу. После проверки имени пользователя и пароля будет организовано соединение с сервером **nessusd** и кнопка **Log in** сменится на кнопку **Log out**, используемую для отключения от сервера.

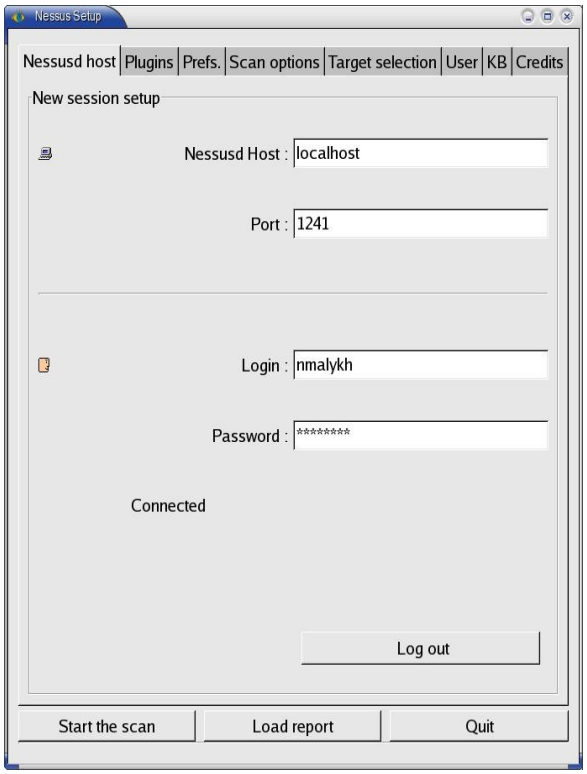


Рисунок 11.14 Панель Nessus host

11.4.4.4.2.2 Панель Plugins

После подключения к серверу **nessusd** вы можете использовать панель **Plugins** (см. рисунок 11.15) для выбора класса модулей атак, которые будут применяться для тестирования выбранного хоста. Окно содержит список классов plugin-модулей в верхней части. Справа от имени класса в каждой строке имеется поле выбора, позволяющее управлять использованием данного класса модулей в предстоящем тестировании хоста. Отмеченные в списке классы модулей будут применяться для имитации атак при проверке безопасности хоста. При выборе того или иного класса модулей в этом списке в нижней части окна появится список модулей выбранного класса с полями выбора модулей для использования в предстоящем тесте. Отметим, что некоторые строки списка модулей могут быть достаточно длинными и поля выбора окажутся за пределами видимой части окна. В этом случае вы можете воспользоваться горизонтальным полем прокрутки в нижней части списка модулей.

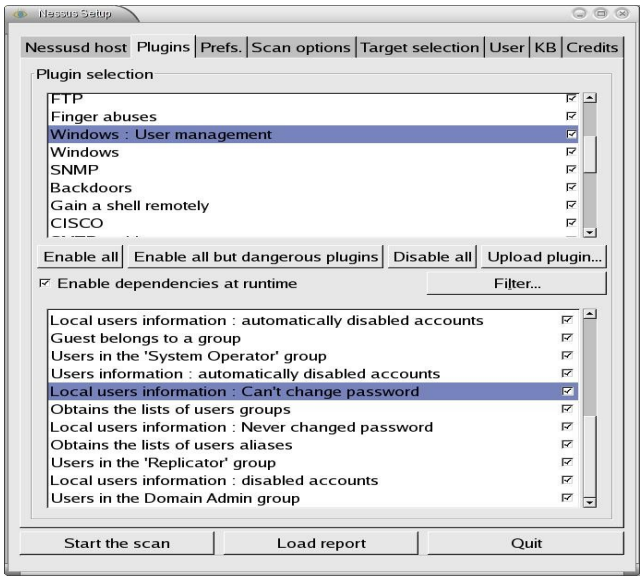


Рисунок 11.15 Панель выбора модулей имитации атак

Как было отмечено выше, модули связаны между собой (одни модули используют результаты работы других) и поле выбора **Enable dependencies at runtime** позволяет автоматически активизировать модули, результаты которых требуются для работы выбранных вами моделей (см. описание опции **auto_enable_dependencies** на стр. 233).

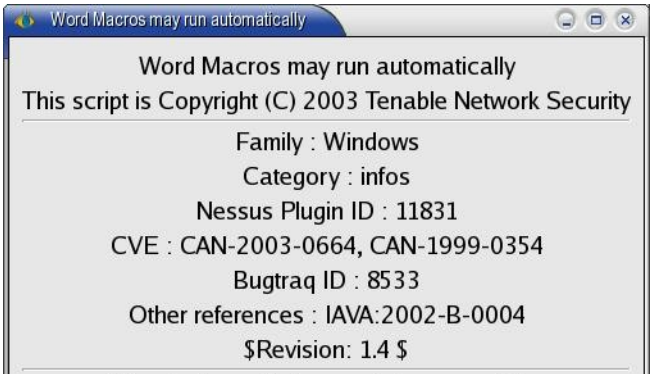
Если вы щелкнете кнопкой мыши по строке с именем модуля в нижнем списке, на экран будет выведено диалоговое окно с кратким описанием этого модуля (см. рисунок 11.16) и выполняемой им проверки. Кроме описания выполняемой модулем проверки окно содержит три кнопки:

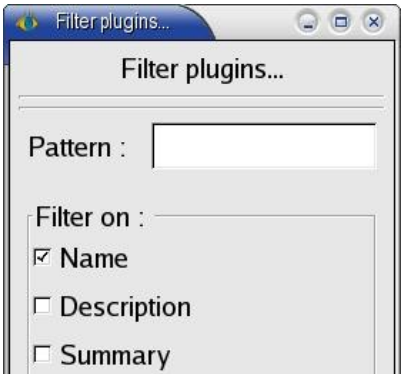
Set plugin timeout... - активизирует одноименное диалоговое окно для задания максимального времени проводимой с помощью этого модуля проверки (см. описание опции **plugins_timeout** на стр. 232).

Show dependencies - активизирует диалоговое окно с информацией о модулях, которые требуются для работы данного модуля.

Close - закрывает диалоговое окно.

Кнопка **Filter...** активизирует диалоговое окно (см. рисунок), позволяющее упростить выбор модулей с помощью фильтрации по тем или иным признакам (имя, автор, категория и т. п.). При большом количестве plugin-модулей фильтрация существенно упрощает поиск и выбор нужного модуля.





11.4.4.4.2.3 Панель Prefs

Панель **Prefs** позволяет задать множество параметров, используемых при тестировании выбранного хоста. Мы не будем здесь перечислять все параметры, поскольку это отнимет слишком много времени и ограничимся лишь кратким списком групп настраиваемых параметров тестирования хоста:

- параметры **ping**;
- параметры сканирования портов с помощью **nmap** (параграф 11.12.1 на стр. 340);
- параметры сканирования портов SNMP;
- параметры тестирования SMB;
- параметры тестирования FTP;
- параметры тестирования HTTP;
- параметры тестирования SMTP;
- параметры тестирования SSH;
- имена и пароли, используемые при попытках доступа к проверяемому хосту;
- глобальные параметры генерации отчетов и записи сообщений в журнальный файл.

Если панель выбора модулей определяет, **что** будет проверяться, то панель предпочтений, по сути, указывает, **как** будут проводиться проверки и что вы получите в результате.

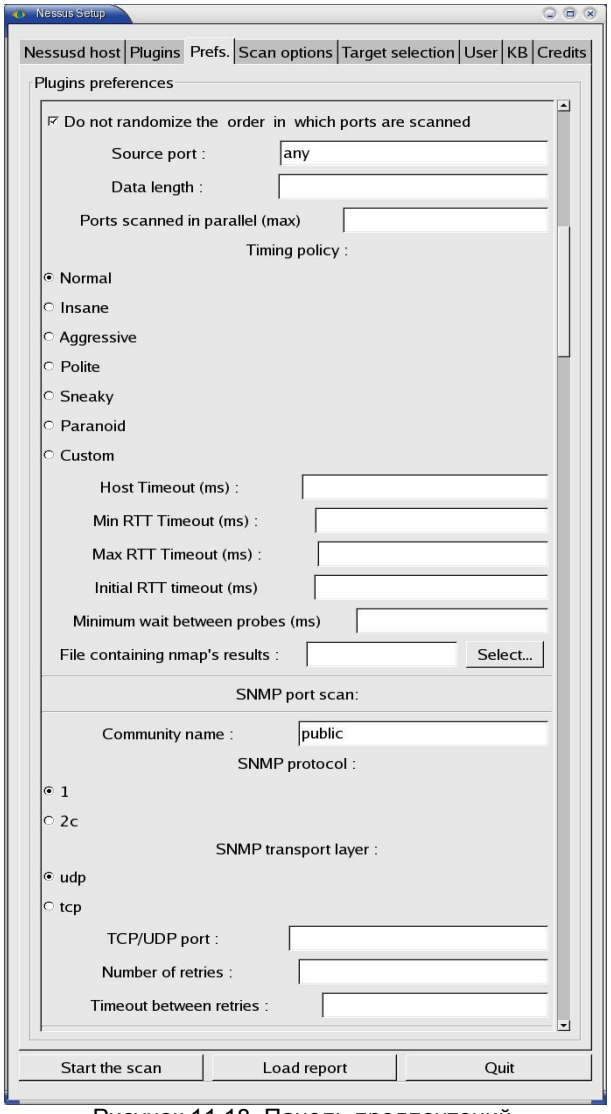


Рисунок 11.18. Панель предпочтений.

11.4.4.4.2.4 Панель Scan Options

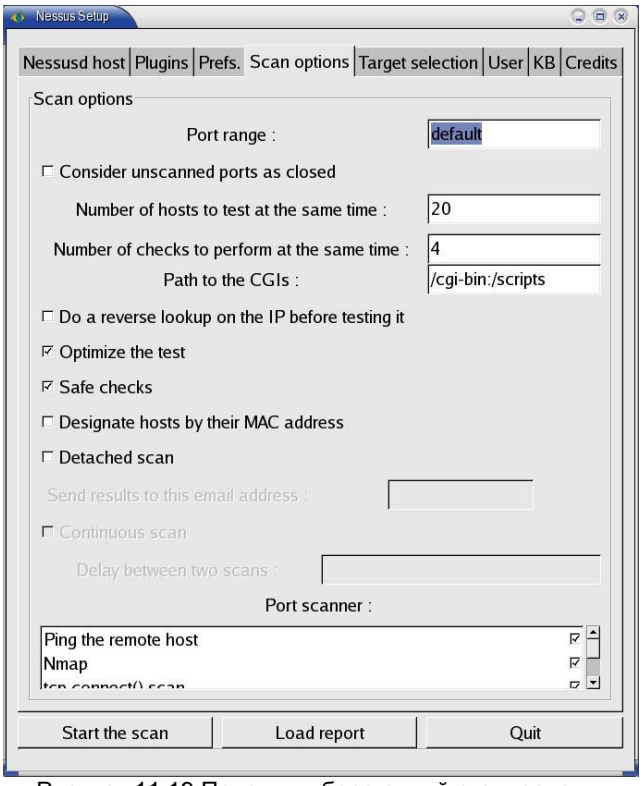


Рисунок 11.19 Панель выбора опций сканирования

Эта панель позволяет выбрать программы, которые будут использоваться для сканирования портов при проверке хоста и указать опции сканирования.

Поле **Port range** позволяет задать диапазон сканируемых портов. Значение **default** будет задавать сканирование портов, установленных в параметрах настройки сервера **nessusd** (см. описание опции **port_range** на стр. 232).

Число одновременно сканируемых хостов (см. описание опции **max_hosts** на стр. 232) и число одновременных проверок (см. описание опции **max_checks** на стр. 232) существенно влияют на загрузку сервера **nessusd** и полосу канала, используемого для тестирования хостов.

Опция **Path to the CGIs** позволяет указать пути поиска сценариев **CGI** на Web-серверах при их проверке.

Опция **Do a reverse lookup on the IP before testing it** определяет режим преобразования адресов IP в символьные имена. При проверке большого числа хостов такое преобразование может существенно замедлить тестирование.

Опция **Safe checks** исключает потенциально опасные тесты, чтобы не нарушить работу проверяемого хоста. Однако при использовании лишь безопасных методов проверки вы рискуете пропустить некоторые уязвимости.

11.4.4.4.2.5 Панель Target Selection

Эта панель служит для выбора цели проверки и очевидно, что указание такой цели является обязательным. Вы должны указать по крайней мере одну цель проверки (хост или сеть). Хост можно указать именем или адресом IP, а сеть - номером IP и маской или списком проверяемых хостов, разделенных запятыми. Список хостов и сетей для выполнения проверки можно прочесть из файла, нажав для выбора этого файла кнопку **Read File...**

Максимальное количество сканируемых одновременно хостов может быть ограничено глобальной опцией **max_hosts** (стр. 232) или установками панели **Scan Options** (см. параграф 11.4.4.4.2.4 на стр. 236).

Поле выбора **Perform a DNS zone transfer** позволяет управлять попытками сервера перенести всю зону тестируемого домена с помощью протокола DNS. При включенной опции в случае тестирования хоста из домена domain.com сервер будет пытаться получить полный список хостов этого домена. Использовать эту опцию следует с осторожностью, поскольку список хостов домена может оказаться весьма большим. Если же вы указали в качестве цели тестирования подсеть, содержащую несколько доменов, сервер будет пытаться получить списки хостов всех доменов проверяемой подсети.

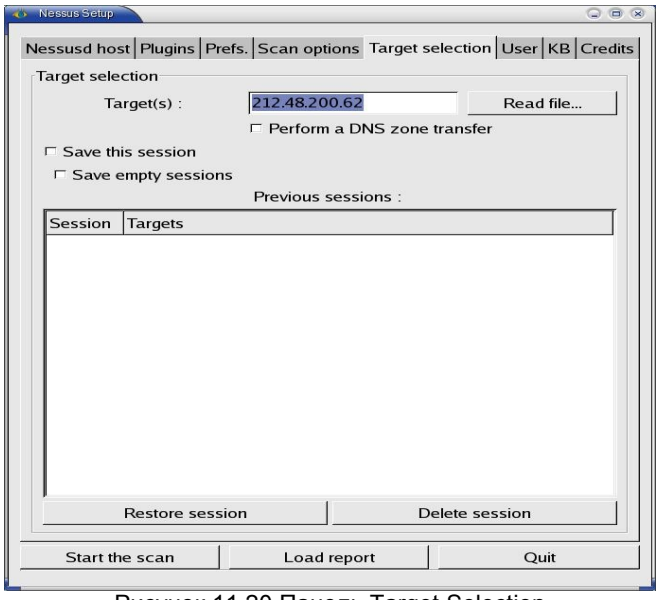


Рисунок 11.20 Панель Target Selection

11.4.4.4.2.6 Панель User

Эта панель (см. рисунок 11.21) позволяет пользователю задать свой набор правил выбора целей проверки. Глобальные правила выбора целей задаются на уровне сервера и пользователю разрешается только вносить дополнительные ограничения в заданные администратором сервера правила проверки удаленных хостов. Синтаксис правил описан в параграфе 11.4.4.3 (стр. 233).

11.4.4.4.2.7 Панель KB

Эта панель позволяет управлять базой знаний, формируемой сервером **nessusd** по результатам проверки выбранных пользователем хостов. Вид панели показан на рисунке 11.22.

11.4.4.4.2.8 Преобразование отчетов

Программа **nessus** позволяет сохранять отчеты о тестировании в различных форматах и обеспечивает возможности преобразования из одного формата в другой. Вы можете загрузить подготовленный ранее отчет в формате NSR или NBE и преобразовать этот отчет в формат HTML, XML, NSR или NBE.

Описания форматов NSR и NBE вы сможете найти в файлах **nsr_file_format.txt** и **nbe_file_format.txt**, распространяемых вместе с программой.

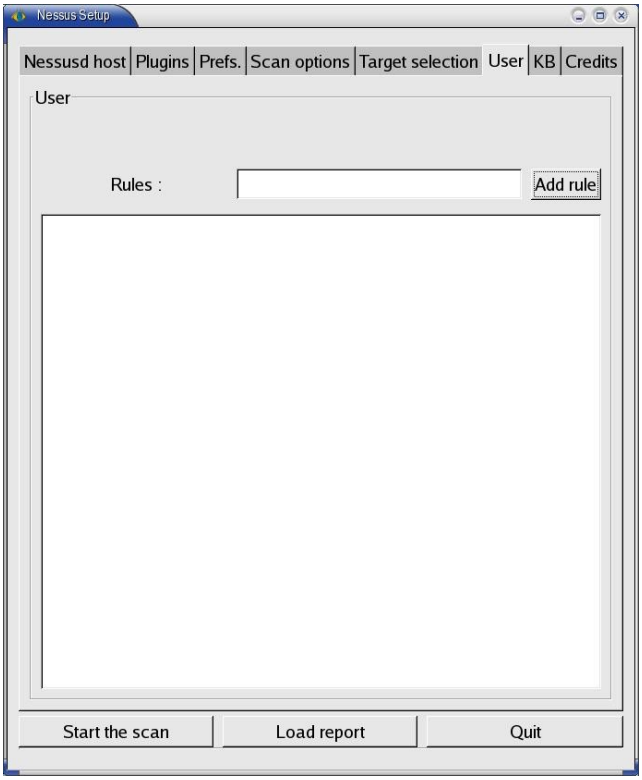


Рисунок 11.21. Панель User.

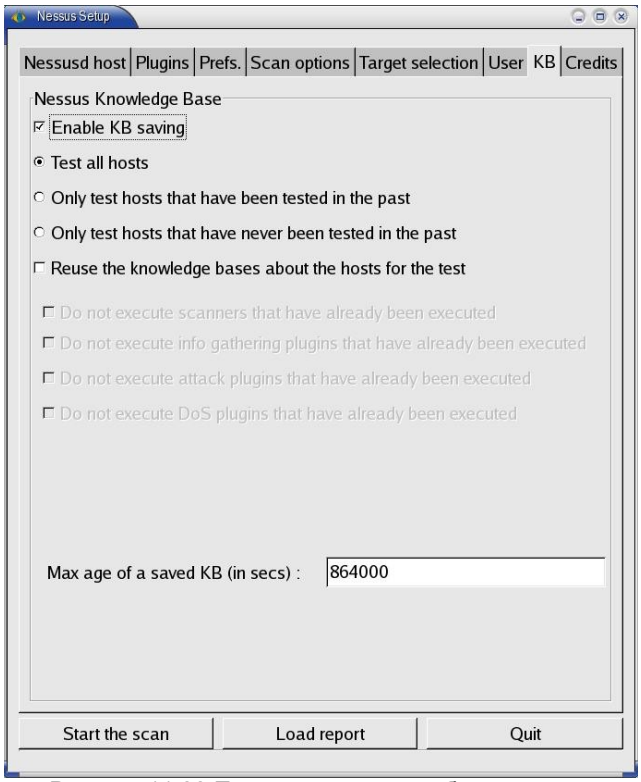


Рисунок 11.22 Панель управления базой знаний

11.4.4.5 Переменные окружения

HOME - эта переменная указывает домашний каталог пользователя, в котором хранится файл пользовательских настроек **.nessusrc**.

NESSUSHOME - если эта переменная установлена, она используется взамен пути, заданного переменной HOME.

11.4.4.6 Файлы

Файл **.nessusrc** содержит выбранные пользователем опции работы с сервером, тестирования и т. п. Если при запуске программы такого файла не обнаружено, он создается автоматически.

11.5 Средства контроля целостности и обнаружения враждебного кода

11.5.1 chkrootkit

http://www.chkrootkit.org

Программа **chkrootkit** предназначена для поиска враждебного кода в системе (rootkit). Программа включает в себя несколько модулей:

- **chkrootkit** - сценарий проверки системы;
- **ifpromisc** - поиск интерфейсов, работающих в режиме захвата пакетов.
- **chklastlog** - обнаружение фактов удаления журнального файла **lastlog** (стр. 46).
- **chkwtmp** - обнаружение фактов удаления журнального файла **wtmp** (стр. 47).
- **check_wtmpx** - обнаружение фактов удаления журнального файла (только для ОС Solaris).
- **chkproc** - поиск следов троянских программ LKM[1].
- **chkdirs** - поиск следов троянских программ LKM.
- **strings** - программа для быстрого поиска и замены текстовых строк.

Модули **chkwtmp** и **chklastlog** пытаются обнаруживать факты удаления системных журналов **wtmp** и **lastlog**, но полное обнаружение всех изменений этих файлов не гарантируется.

Предпринимаются попытки поиска файлов, собранных анализаторами (тест aliens) в обычных местах расположения подобных файлов. Возможность нестандартного расположения файлов не позволяет гарантировать их обнаружение во всех случаях.

Модуль **chkproc** проверяет файлы **/proc** для обнаружения скрытых от команд **ps** и **readdir** системных вызовов, которые могут быть связаны с троянскими модулями LKM. Вы можете использовать эту команду с ключом **-v** для вывода более подробного отчета.

Chkrootkit может обнаруживать широкий спектр враждебного кода, перечисленного ниже.

lrk3, lrk4, lrk5, lrk6 (и варианты)	Maniac-RK	Scalper
Solaris rootkit	dsc-rootkit	Slapper A, B, C и D
FreeBSD rootkit	Ducoci rootkit	OpenBSD rk v1
t0rn (и варианты)	x.c Worm	Illogic rootkit
Ambient's Rootkit (ARK)	RST.b trojan	SK rootkit
Ramen Worm	duarawkz	sebek LKM
rh[67]-shaper	knark LKM	Romanian rootkit
RSHA	Monkit	LOC rootkit
Romanian rootkit	Hidrootkit	shv4 rootkit
RK17	Bobkit	Aquatica rootkit
Lion Worm	Pizdakit	ZK rootkit
Adore Worm	t0rn v8.0	55808.A Worm
LPD Worm	Showtee	TC2 Worm
kenny-rk	Optickit	Volc rootkit
Adore LKM	T.R.K	Gold2 rootkit
ShitC Worm	MithRa's Rootkit	Anonoying rootkit
Omega Worm	George	Shkit rootkit
Wormkit Worm	SucKIT	AjaKit rootkit

1 *Linux Kernel Module - модуль ядра Linux.*

zaRwT rootkit;

Программа работает на различных программных платформах и была успешно протестирована на системах

- Linux с ядрами 2.0.x, 2.2.x и 2.4.x;
- FreeBSD 2.2.x, 3.x, 4.x и 5.x;
- OpenBSD 2.x и 3.x.;
- NetBSD 1.5.2;
- Solaris 2.5.1, 2.6 и 8.0;
- HP-UX 11;
- Tru64;
- BSDI.

Для использования программы **chkrootkit** требуются полномочия пользователя root. Простейший способ проверки обеспечивается командой

```
./chkrootkit
```

В этом случае программа будет автоматически выполнять весь набор поддерживаемых тестов. Для выбора отдельных тестов вы можете воспользоваться параметрами командной строки:

```
./chkrootkit [опции] [<имя теста> ...]
```

11.5.1.1 Опции

Таблица 59. Опции chkrootkit.

Опция	Описание
-h	Выводит справочную информацию о работе с программой.
-V	Выводит сведения о номере версии программы и завершает работу.
-l	Показывает список поддерживаемых программой проверок.
-d	Задает вывод подробной информации о работе программы (режим отладки).
-q	Задает минимальный вывод информации.
-x	Задает вывод дополнительной информации.
-r <каталог>	Задает имя каталога для использования в качестве корневого (root). Указанный в команде каталог служит стартовой точкой для просмотра дерева каталогов.
-p dir1:dir2:dirN	Указывает пути к внешним программам, используемым **chkrootkit.**
-n	Отключает просмотр смонтированных каталогов NFS.

Параметр **<имя теста>** может содержать одно или несколько имен поддерживаемых программой тестов:

aliens	slapper	du	hdparm	login	pop2	sshd	vdir
asp	z2	dirname	su	ls	pop3	syslogd	w
bindshell	amd	echo	ifconfig	lsof	ps	tar	write
lkm	basename	egrep	inetd	mail	pstree	tcpd	
rexedcs	biff	env	inetdconf	mingetty	rpcinfo	tcpdump	
sniffer	chfn	find	identd	netstat	rlogind	top	
wted	chsh	fingerd	init	named	rshd	telnetd	
w55808	cron	gpm	killall	passwd	slogin	timed	
scalper	date	grep	ldsopreload	pidof	sendmail	traceroute	

Например, приведенная ниже команда обеспечивает поиск троянских программ **ps** и **ls**, а также обнаружение интерфейсов, работающих в режиме захвата пакетов.

```
./chkrootkit ps ls sniffer
```

С помощью опции **-q** можно задать работу программы с выводом минимальной информации. В этом случае отчет будет содержать лишь сведения о найденных в системе троянских программах или следах работы анализаторов протоколов и сканеров.

Опция **-x** позволяет пользователю провести поиск подозрительных строк в бинарных файлах, которые могут говорить о присутствии в системе троянских программ. Все решения об идентификации троянских программ пользователь должен будет принять сам. Поскольку в режиме поиска текстовых строк на экран будет выводиться значительный объем информации, целесообразно воспользоваться постраничным выводом:

```
./chkrootkit -x | more
```

Команда

```
./chkrootkit -x | egrep '^/bin'
```

позволяет найти в бинарных файлах текстовые строки, начинающиеся с символов **/bin**, которые могут содержать имена исполняемых файлов. Программа **chkrootkit** может использовать для выполнения проверки другие программы, включая **awk**, **cut**, **egrep**, **find**, **head**, **id** (параграф 11.1.1.5 на стр. 190), **ls**, **netstat** (параграф 11.1.2.5 на стр. 199), **ps** (параграф 11.2.6 на стр. 210), **strings**, **sed**, **uname** (параграф 11.1.1.9 на стр. 192). Если эти программы недоступны в пути поиска, укажите путь к ним с помощью опции **-p**. Такая возможность позволяет использовать при проверке системы заведомо нормальные версии перечисленных программ, которые могут храниться на отдельном диске без возможности записи на него. Приведенная ниже команда обеспечивает выполнение тестов **chkrootkit** с использованием программ, хранящихся в каталоге **/bin** на компакт-диске, смонтированном в системе как **/cdrom**

```
./chkrootkit -p /cdrom/bin
```

Вы можете указать в командной строке несколько каталогов для поиска требуемых для работы программ, разделяя имена каталогов двоеточием (:)

```
./chkrootkit -p /cdrom/bin:/floppy/mybin
```

Иногда может возникнуть необходимость проверки диска вашей системы на другом компьютере, где заведомо нет враждебного кода. Для этого служит опция **-r** , позволяющая задать точку монтирования для корневого раздела проверяемого диска. Например, при монтировании корневого раздела как **/mnt1**, можно использовать команду:

```
./chkrootkit -r /mnt1
```

11.5.1.2 Сообщения программы

Ниже перечислены префиксы, используемые программой **chkrootkit** (за исключением случаев использования с опциями **-x** или **-q**) при выводе отчета о проверке:

- **INFECTED** - проверка показала, что данная программа может относиться к известным образцам враждебного кода (rootkit);
- **not infected** - проверка показала отсутствие сигнатур известных rootkit;
- **not tested** - тест не был выполнен по одной из перечисленных ниже причин:
  a) неприменимость проверки для данной ОС;
  b) отсутствие возможности использования требуемой для теста внешней программы;
  c) заданы опции командной строки, отключающие эту проверку (например, **-r**).
- **not found** - программа не была найдена и по этой причине не проверялась;
- **Vulnerable but disabled** - программа заражена, но не используется (не работала в момент проверки или “закомментирована” в **inetd.conf**).

11.5.2 Tripwire

http://www.tripwire.com

http://www.tripwire.org

Программа Tripwire позволяет контролировать произошедшие в системе изменения, благодаря постоянному мониторингу файлов в масштабе системы. Программа контролирует атрибуты, которые не должны изменяться (например, сигнатуры бинарных файлов, размеры и т. п.), обеспечивая баланс безопасности, управляемости и функциональности системы.

Программа Tripwire является в общем случае коммерческой и распространяется в виде бинарных файлов с соответствующим лицензированием, но для Linux существует открытая версия Tripwire. Благодаря раскрытию исходных кодов сообществу разработчиков и пользователей Linux, программа Tripwire обрела много новых возможностей.

Программа поддерживает несколько режимов работы.

11.5.2.1 Режим инициализации базы данных

Использование команды **tripwire** в режиме инициализации базы данных обычно происходит лишь один раз в процессе настройки программы Tripwire для нормальной работы. Этот режим создает базу данных в каталоге, заданном переменной **DBFILE** в конфигурационном файле Tripwire. База данный является своего рода “моментальной фотографией” состояния системы и используется впоследствии на этапах проверки целостности для фиксации произошедших изменений.

При запуске tripwire в режиме **Database Initialization** программа читает файл политики и генерирует на основе этой политики базу данных, включая по завершении процесса зашифрованную подпись. Имя файла для хранения базы данных задает администратор. Если при запуске команды в режиме генерации базы данных не указано никаких дополнительных параметров, используются принятые по умолчанию значения параметров из текущего конфигурационного файла.

11.5.2.1.1 Опции режима Database Initialization

```
-m i (--init)
```

выбирает режим инициализации базы данных.

```
-v (--verbose)
```

режим вывода подробной информации. Не может использоваться одновременно с флагом **-s**.

-s (--silent, --quiet)
режим минимального вывода. Не может использоваться одновременно с флагом **-v**.

-c <конфигурационный файл> (--cfgfile <конфигурационный файл>)
задает используемый конфигурационный файл.

-p <файл политики> (--polfile <файл политики>)
задает используемый файл политики.

-d <база данных> (--dbfile <база данных>)
указывает местоположение базы данных.

-S <ключ сайта> (--site-keyfile <ключ сайта>)
задает использование файла ключей для сайта, позволяющего читать файлы конфигурации и политики.

-L <локальный ключ> (--local-keyfile <локальный ключ>)
задает использование локального файла ключей для записи нового файла базы данных. Не может использоваться вместе с флагом **-e**.

-P <пароль> (--local-passphrase <пароль>)
задает парольную фразу (passphrase), используемую с локальным ключом для создания подписи к новой базе данных. Не может использоваться вместе с ключом **-e**.

-e (--no-encryption)
отменяет подпись при создании базы данных. Файл базы данных по-прежнему остается сжатым и непонятным для человека. Не может использоваться вместе с опциями **-L** и **-P**.

11.5.2.2 Режим проверки целостности

После создания базы данных Tripwire обычно используется команда **tripwire** в режиме проверки целостности, пытающемся определить нарушения, заданные в файле политики. Используя правила из файла политики, Tripwire сравнивает текущее состояние файловой системы с информацией, сохраненной в базе данных. Отчет о результатах проверки целостности выводится на **stdout** и сохраняется в каталоге, заданном переменной **REPORTFILE** в конфигурационном файле Tripwire.

Отчет содержит детальное описание каждого обнаруженного нарушения политики с указанием произошедших изменений объекта (добавлен, удален, изменен). Каждый элемент отчета содержит текущий список свойств объекта и, если это возможно, список свойств, сохраненный в базе данных. При наличии расхождений администратор может решить проблему заменой испорченного файла на корректный или внесением правок в базу данных с учетом изменений.

Опция **-I** (**--interactive**) активизирует встроенный редактор, обеспечивающий возможность быстрого и эффективного обновления базы данных. Для обновления базы данных можно также использовать специальный режим программы, описанный в следующем параграфе.

11.5.2.2.1 Опции режима Integrity Checking

-m c (--check)
задает режим проверки целостности.

-I (--interactive)
задает интерактивный режим, при котором в конце проверки программа загружает отчет о результатах в текстовый редактор и позволяет администратору отметить записи базы данных для обновления.

-v (--verbose)
режим вывода подробной информации. Не может использоваться одновременно с флагом **-s**.

-s (--silent, --quiet)
режим минимального вывода. Не может использоваться одновременно с флагом **-v**.

-c <конфигурационный файл> (--cfgfile <конфигурационный файл>)
задает используемый конфигурационный файл.

-p <файл политики> (--polfile <файл политики>)
задает используемый файл политики.

-d <база данных> (--dbfile <база данных>)
указывает местоположение базы данных.

-r <имя файла> (--twrfile <имя файла>)
задает запись отчета в указанный файл.

-S <ключ сайта> (--site-keyfile <ключ сайта>)
задает использование файла ключей для сайта, позволяющего читать файлы конфигурации и политики.

-L <локальный ключ> (--local-keyfile <локальный ключ>)
задает использование локального файла ключей для записи нового файла базы данных. Не может использоваться вместе с флагом **-e**.

-P <пароль> (--local-passphrase <пароль>)
задает парольную фразу (passphrase), используемую с локальным ключом для создания подписи к новой базе данных. Не может использоваться вместе с ключом **-e**.

-e (--no-encryption)

отменяет подпись при создании базы данных. Файл базы данных по-прежнему остается сжатым и непонятным для человека. Не может использоваться вместе с опциями **-L** и **-P**.

-V <редактор> (--visual <редактор>)
задает редактор для выбора изменяемых записей (ballot box) в отчете. Имеет смысл только при совместном использовании с флагом **-l**.

-E, --signed-report
задает необходимость подписи Tripwire в создаваемом отчете. Если в командной строке не был указан пароль, программа будет запрашивать локальный пароль.

-i <список> (--ignore <список>)
задает пропуск расчета и сравнения для свойств, указанных в списке. Любый символьные коды (**abcdgimnprstulCHMS**), указанные в масках свойств, пропускаются при проверке. Использование этой опции отменяет соответствующие правила из файла политики. Список задается в двойных кавычках с разделением элементов запятыми (например, **--ignore "p,c,m"**).

-l { уровень | имя } (--severity { уровень | имя })
задает проверку только для правил, уровень важности (severity) которых не меньше заданного. Для указания уровня может использоваться имя или число. Программа поддерживает следующие уровни важности:

Low (низкий) 33

Medium (средний) 66

High (высокий) 100

Опция не может использоваться вместе с флагом **-R**.

-R <правило> (--rule-name <правило>)
задает проверку только для указанного правила. Эта опция не может использоваться одновременно с флагом **-l**.

-x <имя секции> (--section <имя секции>)
задает проверку только для правил указанной секции. Для Tripwire 2.3.1 единственным поддерживаемым значением этого параметра является **FS**.

-M, --email-report
задает отправку отчета по электронной почте. Адрес получателя указывается в файле политики.

-t <уровень> (--email-report-level <имя секции>)
задает уровень детализации передаваемого по электронной почте отчета. Эта опция отменяет уровень детализации, заданной переменной **EMAILREPORTLEVEL** в конфигурационном файле. Допустимы значения уровня от 0 до 4. Опция должна использоваться совместно с флагом **-M**.

[объект1 [объект2...]]
задает список файлов и каталогов, для которых должна выполняться проверка целостности. По умолчанию программа проверяет все дерево каталогов системы. При заданном в командной строке списке проверяемых файлов опции **--severity** и **--rule-name** игнорируются.

11.5.2.3 Режим обновления базы данных

Команда **tripwire** в режиме Database Update (обновление базы данных) позволяет учесть произошедшие в системе изменения и предотвратить появление ненужных сообщений при последующих проверках. Если обнаруженное несоответствие неожиданно и потенциально опасно, измененный файл следует заменить исходной версией. Если же изменения объясняются разумными причинами, следует внести поправки в базу данных, чтобы она отражала актуальное состояние.

При работе в режиме обновления базы данных элементы, которые должны быть изменены помещаются специальным значком (**x**) в предоставляемом пользователю текстовом отчете. Если запись следует изменить, пользователь просто сохраняет в строке отчета знак **x** и **tripwire** автоматически вносит поправки в базу данных после того, как редактирование отчета будет завершено и пользователь введет пароль по запросу программы. Опции управления режимом обновления базы данных включают флаги **-Z** (**--secure-mode**) и **-a** (**-accept-all**).

11.5.2.3.1 Опции режима Database Update

-m u (--update)
задает для программы работу в режиме обновления базы данных.

-v (--verbose)
режим вывода подробной информации. Не может использоваться одновременно с флагом **-s**.

-s (--silent, --quiet)
режим минимального вывода. Не может использоваться одновременно с флагом **-v**.

-c <конфигурационный файл> (--cfgfile <конфигурационный файл>)
задает используемый конфигурационный файл.

-p <файл политики> (--polfile <файл политики>)
задает используемый файл политики.

-d <база данных> (--dbfile <база данных>)
указывает местоположение базы данных.

-r <имя файла> (--twrfile <имя файла>)

задает запись отчета в указанный файл.

-S <ключ сайта> (--site-keyfile <ключ сайта>)
задает использование файла ключей для сайта, позволяющего читать файлы конфигурации и политики.

-L <локальный ключ> (--local-keyfile <локальный ключ>)
задает использование локального файла ключей для записи нового файла базы данных. Не может использоваться вместе с флагом **-e**.

-P <пароль> (--local-passphrase <пароль>)
задает парольную фразу (passphrase), используемую с локальным ключом для создания подписи к новой базе данных. Не может использоваться вместе с ключом **-e**.

-V <редактор> (--visual <редактор>)
задает редактор для выбора изменяемых записей (ballot box) в отчете. Имеет смысл только при совместном использовании с флагом **-l**.

-a (--accept-all)
задает необходимость обновления всех записей из отчета без запросов. Эта опция не может использоваться вместе с флагом **-V**.

-Z { low | high } (--secure-mode { low | high })
задает уровень безопасности, определяющий поведение программы в тех случаях, когда обнаружено то или иное несоответствие между базой данных и файлом отчета:

- **High** (высокий) - если файл не соответствует указанным в отчете свойствам, Tripwire выдает предупреждение с информацией о различиях и завершает работу без обновления базы данных.
- **Low** (низкий) - при обнаружении расхождений программа выдает предупреждение и обновляет базу данных.

11.5.2.4 Режим обновления политики

Режим **Policy Update** используется программой **Tripwire** для изменения или обновления файла политики и синхронизации базы данных с новой политикой. Имя нового текстового файла с политикой задается в командной строке. Новый файл политики сравнивается с существующей версией и база данных обновляется в соответствии с новой политикой. Все изменения в базе данных с момента последней проверки целостности будут включены в отчет. Эти изменения будут интерпретироваться в зависимости от заданного режима безопасности (опция **-Z** или --secure-mode). В принятом по умолчанию жестком режиме (high security) **Tripwire** будет выводить список обнаруженных нарушений политики и завершать работу без изменения базы данных. В мягком режиме (low security) будет выводиться отчет и корректировка базы данных.

Поскольку файлы правил и базы данных хранятся в бинарном виде с криптографическими подписями, у пользователя будут запрошены пароли для изменения. После обновления базы данных правила и база заново кодируются и подписываются.

11.5.2.4.1 Опции режима Policy Update

-m p (--update-policy)
Задает режим обновления политики.

-v (--verbose)
Задает подробный вывод информации. Не может использоваться вместе с опцией **-s**.

-s (--silent, --quiet)
Задает режим минимального вывода информации. Не может использоваться вместе с флагом **-v**.

-c <файл> (--cfgfile <файл>)
Задает использование указанного файла конфигурации.

-p <файл> (--polfile <файл>)
Задает запись указанного файла политики.

-d <файл> (--dbfile <файл>)
Задает использование базы данных из указанного файла.

-S <ключ> (--site-keyfile <ключ>)
Задает использование указанного файла ключей сайта для чтения конфигурационного файла, а также чтения и записи файла политики.

-L <ключ> (--local-keyfile <ключ>)
задает использование локального файла ключей для чтения и записи в файл базы данных.

-P <пароль> (--local-passphrase <пароль>)
Задает парольную фразу, которая вместе с локальным ключом используется при подписывании базы данных.

-Q <пароль> (--site-passphrase <пароль>)
Задает парольную фразу, которая вместе с ключом сайта используется при подписывании нового файла политики.

-Z { low | high } (--secure-mode { low | high })
Задает уровень безопасности, определяющий поведение системы при обнаружении несоответствия файловой системы сохраненной в базе данных информации. Поскольку база данных созданная при обновлении политики будет служить основой для последующих проверок, следует убедиться, что в системе не произошло критичных изменений с момента последней проверки целостности. Поддерживаются два режима безопасности при

обновлении политики:

High (жесткий): все обнаруженные несоответствия указываются в отчете и программа **Tripwire** завершает работу без обновления базы данных и файла политики.

Low (мягкий): в отчете указываются все обнаруженные несоответствия и производится автоматическая корректировка базы данных и файла политики.

Файл **policyfile.txt** содержит текстовый вариант политики, на основе которого создается новый бинарный файл политики **Tripwire**.

11.5.2.5 Тестовый режим

Режим **Test** используется для проверки работы системы почтовых уведомлений программы **Tripwire**. При работе в этом режиме программа использует параметры системы оповещения по электронной почте, заданные в конфигурационном файле. При установке **MAILMETHOD = SMTP**, параметры **SMTPHOST** и **SMTPPORT** будут использоваться для отправки уведомлений по электронной почте. Если выбрано значение **MAILMETHOD = SENDMAIL**, будет использоваться значение **MAILPROGRAM**. При корректной работе системы уведомления по электронной почте получателю будет приходить тестовое сообщение:

```
To: user@domain.com
From: user <user@domain.com>
Subject: Test email message from Tripwire
```

тестовый режим полезен только для тестирования доставки почтовых уведомлений по адресу, заданному в командной строке и не проверяет синтаксис адреса, заданного в файле политики.

11.5.2.5.1 Опции режима Test

-m t (--test)

Задает работу в режиме проверки системы оповещения.

-e <почтовый адрес> (--email <почтовый адрес>)

Задает адрес, по которому будет отправлено тестовое сообщение программы. Этот параметр является обязательным для режима Test и может включать только один адрес.

11.5.3 samhain

http://www.la-samhna.de

Пакет **samhain/yule** представляет собой открытую систему проверки целостности файлов и детектирования попыток вторжения (IDS) для систем UNIX. Программа распространяется как в исходных кодах, так и в форме бинарных файлов. Работа программы была протестирована на платформах Linux, FreeBSD, AIX 4.x, HP-UX 10.20, Unixware 7.1.0, Solaris 2.x и Alpha/True64. Имеется опыт использования программы также на платформах OpenBSD, AIX 5.x, HP-UX 11 и Mac OS X. Фактически, программа может работать на всех POSIX-совместимых платформах. Известны случая использования **samhain** под управлением ОС Windows 2000/XP.

Основные возможности программы

- полная проверка целостности:
  - samhain использует шифрование контрольных сумм для обнаружения изменений в файлах;
  - обеспечивается поиск на диске подставных злонамеренных программ SUID (см. параграф 2.3.3 на стр. 42);
  - обеспечивается обнаружение программ **kernel rootkit** для платформ Linux и FreeBSD.
  - samhain может работать как демон, запоминая всю информацию об изменении файлов и не повторяя сообщений о таких изменениях.
- Поддерживается централизованный мониторинг с использованием шифрованных соединений TCP/IP с центральным сервером. База данных с контрольными суммами и клиентские параметры конфигурации могут храниться на сервере.
- Обеспечивается журнализация с использованием SQL (MySQL, PostgreSQL или Oracle; поддерживается unixODBC).
- База контрольных сумм и конфигурационные файлы могут использовать сигнатуры PGP.
- Поддерживается скрытый (**stealth**) режим работы.
- Для систем клиент-сервер имеется web-интерфейс в виде отдельного пакета Beltane.

Пакет включает программу мониторинга **samhain**, используемую на проверяемых хостах и необязательный сервер журнальных файлов **yule**. Программа мониторинга **samhain** может проверять целостность файлов и каталогов, а также отслеживать факты регистрации и выхода пользователей из системы.

Информация **samhain/yule** может передаваться по электронной почте, записываться в защищенный от подделки журнальный файл с цифровой подписью, журнальный файл **syslog** или передаваться на стандартное устройство вывода[1]. Система мониторинга может использоваться в режиме демона. Синхронизация операций может осуществляться от системного таймера или внешнего сервера точного времени. Большинство параметров программы задается в конфигурационном файле, считываемом с диска при запуске программы.

1 *Устройство **/dev/console** при работе в режиме демона.*

11.5.3.1 Опции команд samhain/yule

Большинство опций работы программы задается в конфигурационном файле. Однако опции командной строки имеют более высокий приоритет и могут использоваться для изменения режима работы программы без редактирования файла конфигурации.

Таблица 60 Опции команд samhain/yule

Опция		Описание
-t	**--set-checksum-test=**	Задает режим инициализации (**init**) или обновления (**update**) базы данных или режим проверки файлов (**check**). Если у вас имеется старая база данных и вы хотите повторно инициализировать данные, старый файл базы данных нужно удалить до запуска программы **samhain**, поскольку в противном случае новая база будет добавлена в старый файл.
-D	**--daemon**	Задает работу программы в режиме демона.
-r	**--recursion=**	Параметр этой опции задает глубину рекурсии при просмотре каталогов. По умолчанию рекурсия отключена (0).
-s	**--set-syslog-severity=**	Параметр это опции задает пороговый уровень для записи событий в журнальный файл с использованием syslogd. Возможными значениями параметра являются **debug**, **info**, **notice**, **warn**, **mark**, **err**, **crit**, **alert** и **none** (см. параграф 2.8.4.2 на стр. 50). По умолчанию в журнальный файл записываются все события, уровень которых не превышает заданный порог. Временные метки программы имеют уровень **warn**, сообщения о системных ошибках - **err**, а важные сообщения при старте программы **alert**. Сигнатура журнального файла никогда не записывается в syslog или журнальный файл программы.
-l	**--set-log-severity=**	Параметр это опции задает пороговый уровень для записи событий в журнальный файл программы.
-m	**--set-mail-severity=**	Параметр это опции задает пороговый уровень для передачи информации о событиях по электронной почте.
-e	**--set-export-severity=**	Параметр это опции задает пороговый уровень для пересылки информации о событиях на сервер syslog по протоколу TCP.
-x	**--set-extern-severity=**	Параметр это опции задает пороговый уровень для вызова внешней программы, указанной в конфигурационном файле. Эта опция не работает для команды **yule**.
-p	**--set-print-severity=**	Параметр это опции задает пороговый уровень для вывода информации о событиях на устройство **stdout**. При работе программы в режиме демона такие сообщения передаются устройству **/dev/console**.
-L	**--verify-log=**	Проверяет целостность журнального файла **samhain**, указанного параметром. При использовании этой опции программа будет запрашивать исходную сигнатуру файла или полное имя файла, в котором хранится эта сигнатура. Сигнатура генерируется автоматически и передается указанным в конфигурационном файле адресатам по электронной почте. Использование такой процедуры позволяет обеспечить контроль доступа к содержимому журнального файла программы.
-j	**--just-list**	Эта опция используется для вывода имени журнального файла.
-M	**--verify-mail=**	Эта опция используется для проверки целостности полученной электронной почты. В качестве параметра должно передаваться имя почтового ящика (файла), в котором хранится проверяемая почта. Возвращаемая программой сигнатура будет считываться из почтового сообщения с номером 0.
-H	**--hash-string=**	Рассчитывает контрольную сумму заданного файла или строки. Имена файлов должны быть полными (т. е., начинаться с символа */*).
-c	**--copyright**	Выводит на экран сокращенный текст лицензии GPL.
-h	**--help**	Выводит на экран справочную информацию о работе с программой.
-S	**--server**	Задает работу программы в режиме log-сервера.
-q	**--qualified**	При работе в режиме log-сервера задает запись полных имен клиентских хостов.
	--chroot=	Задает корневой каталог при работе в режиме log-сервера.
-G	**--gen-password**	Генерирует и выводит на экран случайный пароль. Пароль выводится в виде шестнадцатеричных кодов символов пароля. Эта опция используется только для команды **yule**.
-P	**--password=**	Рассчитывает регистрационную запись (**salt** и **verifier**) для клиента TCP с указанным паролем. Пароль следует задавать в виде последовательности шестнадцатеричных кодов символов пароля. Эта опция используется только для команды **yule**.

11.5.3.2 Сигналы

Таблица 61. Сигналы программ samhain/yule.

Сигнал	Описание
SIGUSR1	Переключает в режим вывода дополнительной информации на консоль.
SIGUSR2	Снижает информативность вывода на консоль до предыдущего уровня.
SIGHUP	Инициирует повторное считывание параметров из конфигурационного файла.
SIGTERM	Прерывает работу программы
SIGQUIT	Прерывает работу программы после выполнения всех клиентских запросов из очереди.
SIGABRT	Снимает блокировку журнального файла, и начинает после трехсекундной паузы новый процесс аудита.

11.5.3.3 База данных

База данных представляет собой двоичный файл (по умолчанию **samhain_file**), который может быть создан или обновлен с помощью опций **-t init** или **-t update**. При использовании **-t init** нужно сначала удалить старую базу данных, поскольку в противном случае новая база будет просто добавлена к старому файлу. Файл может использовать цифровую подпись PGP/GnuPG. Рекомендуется использовать для создания цифровой подписи команду

```
gpg -a --clearsign -not-dash-escaped
```

Программа **samhain** будет проверять сигнатуру файла, если при компиляции была включена опция проверки.

При запуске программы **samhain** вычисляется контрольная сумма базы данных и при каждом последующем обращении к базе эта контрольная сумма проверяется. Контрольная сумма не записывается на диск из соображений безопасности, поэтому при каждом новом запуске программы вычисление контрольной суммы происходит заново.

11.5.3.4 Журнальный файл

Каждая запись журнального файла имеет формат

```
Уровень важности :  [временная метка] сообщение
```

временная метка обычно берется от сервера точного времени, а не от системных часов, если программа была скомпилирована с поддержкой такой возможности. Каждая запись журнального файла сопровождается сигнатурой, рассчитываемой как **Hash(Entry Key_N)**, а **Key_N** рассчитывается как **Hash(Key_N-1)**. Таким образом для проверки целостности журнального файла требуется знать первую сигнатуру в цепочке. Эта сигнатура генерируется автоматически и передается заданному пользователю по электронной почте.

По умолчанию журнальный файл **samhain** носит имя **samhain_log**. Для предотвращения записи в один файл разными экземплярами программы этот файл блокируется путем создания lock-файла, который обычно удаляется при завершении работы программы. По умолчанию файл блокировки называется **samhain.pid**. При некорректном завершении работы **samhain** (например, по команде (**kill -9**) файл блокировки может сохраниться, но программа обычно способна распознавать старые файлы блокировки и удалять их при старте.

11.5.3.5 Почтовые сообщения

Сообщения по электронной почте с использованием встроенного агента SMTP передаются только одному адресату. Поле темы сообщения (subject) содержит имя хоста и временную метку, которые включаются также в текст сообщения. Переданное администратору сообщение включает строку с сигнатурой, подобную той, что записывается в журнальный файл. Сигнатуры создаются с использованием цепочки ключей и первый ключ передается по электронной почте администратору, указанному в конфигурационном файле. Это письмо следует сохранить в надежном месте, поскольку исходный ключ может потребоваться впоследствии.

На некоторых хостах может осуществляться проверка имени отправителя электронной почты, поэтому следует использовать реальные регистрационные имена пользователей.

Для передачи сообщений по электронной почте в конфигурационном файле программы должен быть указан почтовый транслятор для домена отправителя.

11.5.3.6 Использование программы в режиме клиент-сервер

Для мониторинга нескольких хостов и сбора данных на центральном сервере программу **samhain** можно скомпилировать для работы в режиме **клиент-сервер**. Сервер журнализации **yule** будет принимать запросы на соединение только от зарегистрированных клиентов. При каждом обращении клиента сервер сначала проверяет его аутентичность и полномочия, после чего создает сеансовый ключ. Такой режим требует наличия клиентского пароля, который проверяется на сервере модулем верификации.

Для регистрации клиента нужно выполнить следующие операции:

1) С помощью утилиты **samhain_setpwd** установите для исполняемого файла клиента желаемый пароль. Если вы хотите использовать в пароле непечатаемые символы, пароль следует задавать в форме шестнадцатеричной строки, соответствующей 8-символьному паролю.

2) На сервере введите команду

```
yule -P password
```
с соответствующим шестнадцатеричным представлением выбранного пароля. Программа выдаст значение **verifier** для регистрации нового клиента.

3) Включите полученную на этапе 2 запись в секцию **[Clients]** конфигурационного файла сервера. Запись должна иметь форму
```
Client=hostname@salt@verifier
```
где вместо **hostname** указывается полное доменное имя клиентского хоста.

При попытке подключения клиента сервер будет искать для него запись в конфигурационном файле и использовать значение параметра **verifier** для организации обмена сеансовыми ключами.

11.5.3.7 Скрытый режим работы программы - STEALTH

Программа **samhain** может быть скомпилирована с поддержкой скрытого режима работы, означающего, что на диске вашей системы не будет обычных следов присутствия программы. Исполняемый файл программы в таком случае не содержит печатаемых символов (строк текста), а конфигурационные параметры прячутся в файлы postscript с достаточно большими изображениями без компрессии. Для создания таких файлов можно воспользоваться утилитой **convert** из пакета **ImageMagick** и каким-либо графическим файлом.

Для того, чтобы скрыть конфигурационные параметры в файле **postscript** или извлечь из такого файла служит утилита **samhain_stealth**.

В качестве файла базы данных и журнального файла можно использовать, например, существующие графические файлы, к которым добавляются данные программы с использованием логической операции XOR в целях маскировки.

Пользователь может дополнительно изменить имя исполняемого файла и выбрать при компиляции какие-либо экзотические имена для файлов конфигурации, базы данных и т. п.

Применение комплекса таких мер поможет достаточно хорошо скрыть наличие программы **samhain**.

11.5.3.8 Безопасность

Из соображений безопасности **samhain** не будет записывать журнальный файл или данные в каталог, снимать блокировку и читать конфигурационный файл, если любым элементом пути владеет пользователь, к которому нет должного доверия, или такой пользователь имеет право записи в такой элемент (каталог или файл). К таким элементам относятся и те, которые разрешают запись для группы, включающей пользователей без должного доверия, или для всех пользователей. К числу доверенных пользователей относится **root** и пользователи, чьи имена указаны в конфигурационном файле.

В передаваемых по электронной почте сообщениях разумней использовать адреса IP, нежели символьные имена хостов.

При установке бинарного дистрибутива **samhain** вы рискуете тем, что злоумышленник может воспользоваться такой же копией программы и провести ее анализ с целью создания троянских программ для получения доступа к вашей системе. Для предотвращения такой возможности вы можете просто добавить в свой исполняемый файл дополнительные ключи с помощью команды:

```
samhain --add-key=key@/path/to/executable
```
Добавляемый ключ не должен содержать символа **@**, поскольку этот символ используется командой в качестве разделителя между ключом и строкой пути к исполняемому файлу, в который добавляется ключ. Перед выполнением команды поместите копию исполняемого файла **samhain** в указанный командой каталог, а после выполнения команды переместите этот файл на место используемой копии исполняемого файла **samhain**.

Не рекомендуется использовать бинарные дистрибутивы программы без описанной процедуры добавления ключа.

Для инициализации ключей используется устройство **/dev/random**, если оно доступно. Это устройство добавляет случайный шум для повышения криптоустойчивости системы. Это устройство работает достаточно медленно в силу своей природы и может создать иллюзию зависания программы **samhain** при старте. Просто наберитесь терпения.

Если в вашей системе не поддерживается устройство **/dev/random**, программа будет использовать для генерации случайных чисел статистику vmstat и другую информацию с достаточно случайным распределением.

11.5.3.8.1 Известные проблемы

Для изменения журнального файла и передачи сообщений программы по электронной почты достаточно знать исходную сигнатуру (ключ). Этот ключ передается по электронной почте при старте программы в зашифрованном виде (one-time pad encryption). Такая мера спасает от перехвата сообщения в сети, но она не поможет, если перехвативший почту злоумышленник получит доступ для чтения файлов samhain.

11.6 Средства обнаружения вторжений (IDS)

11.6.1 Snort

http://www.snort.org/

Пакет Snort представляет собой систему детектирования попыток вторжения (IDS) с открытым кодом, способную в реальном масштабе времени обеспечивать анализ трафика и протоколирование работы в сетях IP. Программа

анализирует протоколы и проверяет содержимое пакетов (поиск соответствий), что позволяет использовать ее для обнаружения широкого класса атак и несанкционированных зондов, включая переполнение буферов, скрытое сканирование портов, CGI-атаки, зонды SMB, попытки идентификации ОС и многое другое. Snort использует гибкий язык создания правил для описания трафика, который следует собирать или пропускать, а также детектор, использующий модульную (plugin) архитектуру. Snort также включает модульную систему выдачи сигналов тревоги в реальном масштабе времени, включающую модули для генерации сигналов и записи информации в syslog (см. параграф 2.8.4 на стр. 49), ASCII-файлы, сокеты UNIX, базы данных (Mysql/PostgreSQL/Oracle/ODBC) и XML.

Пакет Snort предназначен для использования в качестве системы:

- сбора пакетов (см. параграф 11.9 на стр. 261);
- протоколирования пакетов (полезно при отладке и тонкой настройке картины сетевого трафика);
- детектирования попыток вторжения.

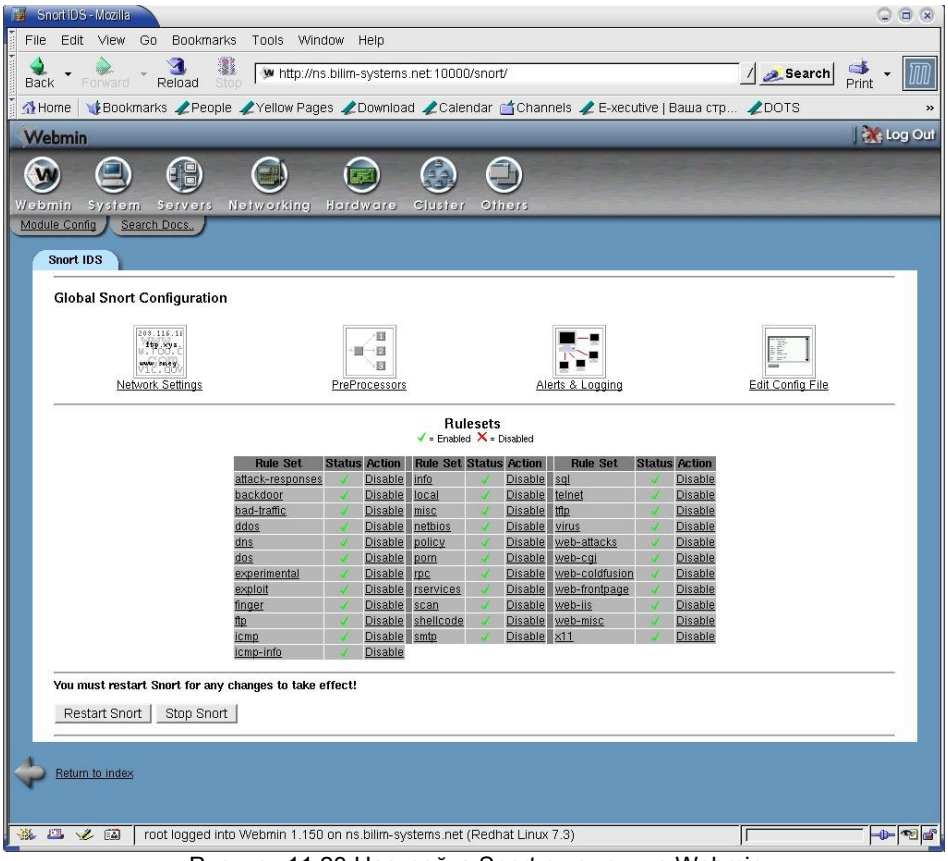


Рисунок 11.23 Настройка Snort с помощью Webmin

Snort записывает пакеты в бинарном формате tcpdump (см. параграф 11.9.2 на стр. 262) в базу данных или в декодированном ASCII-формате в файлы системных журналов с именами, соответствующими IP-адресам хостов.

11.6.1.1 Опции

`-A <режим>`

обеспечивает выдачу предупреждений в соответствии с заданным режимом. Параметр может принимать значения **fast**, **full**, **none** и **unsock**. В режиме **fast** сигналы тревоги записываются в заданный по умолчанию файл с использованием стиля тревожных сообщений syslog (одна строка текста на каждое сообщение). В режиме **full** сигнал записывается в alert-файл с полным декодированием заголовка и текстом тревожного сообщения. Режим **none** отключает генерацию сигналов тревоги. Режим **unsock** является экспериментальным и передает информацию через сокет UNIX другому процессу, подключенному к тому же сокету.

`-b`

задает запись пакетов в файл формата tcpdump (см. параграф 11.9.2 на стр. 262). Все пакеты записываются в их естественном бинарном состоянии в файл, имя которого включает временную метку запуска команды **snort** и суффикс **snort.log**. Использование этой опции ускоряет работу программы, поскольку избавляет от необходимости преобразования пакетов в текстовый формат. При использовании с этой опцией Snort может нормально работать с сетевыми интерфейсами вплоть до скорости 100 Мбит/с. Для выбора иного имени log-файла вы можете использовать опцию **-L**.

`-B <маска>`

преобразует все IP-адреса “домашней” сети в адреса, заданные маской. Эта опция служит для сокрытия IP-адресов в бинарном файле захвата пакетов. Домашняя сеть[1] задается с помощью опции **-h**.

`-c config-file`

задает имя конфигурационного файла.

`-C`

1 *Не путайте “домашнюю” сеть со значением переменной $HOME_NET.*

задает вывод текстовой информации из пакетов данных. Шестнадцатеричное представление в этом случае не используется.

-d
задает вывод дампа данных прикладного уровня при работе в режиме протоколирования пакетов (**logging**) или **verbose**.

-D
задает работу программы Snort в режиме демона. Демон записывает сигналы тревоги **/var/log/snort/alert**, если явно не указано иное имя файла.

-e
задает вывод и запись в журнальный файл заголовков канального уровня.

-F bpf-file
задает чтение фильтров BPF из bpf-файла. Эта опция создана для любителей сверхсложных фильтров BPF, а также для тех, кто использует Snort взамен SHADOW. Краткое описание фильтров BPF приводится ниже).

-g group
изменяет группу, от имени которой программа Snort будет работать после инициализации. Эта опция позволяет в целях безопасности отказаться от использования привилегий суперпользователя (root) после завершения инициализации программы.

-h <домашняя сеть>
задает "домашнюю" сеть. Блок адресов "домашней" сети указывается в формате **сеть/маска** (например, 192.168.1.0/24). После установки этой переменной запись всех декодированных пакетов будет осуществляться с учетом домашней сети. Особенно полезна эта опция при записи пакетов в формате ASCII. Если параметр указывает на вашу локальную сеть, декодированные файлы будут записываться в каталоги, имена которых совпадают с адресами чужих хостов. Такая организация файлов бывает весьма удобна при анализе результатов.

-i <интерфейс>
задает сбор пакетов из указанного интерфейса.

-I
задает включение имени интерфейса в передаваемые программой сигналы тревоги.

-k <режим>
включает внутренний режим проверки контрольных сумм с выдачей сигналов тревоги. Для проверки контрольных сумм могут использоваться режимы **all**, **noip**, **notcp**, **noudp**, **noicmp** и **none**. В режима **all** контрольные суммы проверяются для всех протоколов, **noip** отключает проверку контрольных сумм для протокола IP (это удобно, если шлюз отбрасывает пакеты с некорректной контрольной суммой IP), **notcp** отключает проверку контрольных сумм TCP, **noudp** - UDP, **noicmp** - ICMP. В режиме **none** контрольные суммы не проверяются ни для одного из протоколов.

-l <каталог>
задает запись журнального файла в указанный каталог. В файл записываются все текстовые сигналы и пакеты. Если эта опция не используется, журнальный файл сохраняется в каталоге **/var/log/snort**.

-L <файл>
задает имя бинарного журнального файла. Если этот параметр не задан, создается файл, имя которого образуется из текущего времени (timestamp) и суффикса **snort.log**.

-m umask
задает маску прав доступа (см. параграф 2.3.2 на стр. 41) для создаваемых программой файлов .

-n <значение счетчика>
задает сбор заданного числа пакетов с последующим завершением работы.

-N
выключает запись пакетов в журнальный файл. Программа продолжает генерировать сигналы тревоги как обычно.

-o
изменяет порядок применения правил к пакетам. Взамен обычного порядка **Alert->Pass->Log** используется **Pass->Alert->Log**.

-O
скрывает IP-адреса при выводе пакетов в режиме ASCII-дампа. На экран и в журнальный файл взамен адресов IP выводятся строки **xxx.xxx.xxx.xxx**. При использовании опции **-h** (домашняя сеть) скрываются только адреса из домашней сети. Эта опция удобна в тех случаях, когда вы планируете пересылать отчеты другим людям (например, в дискуссионные группы).

-p
отключает режим захвата пакетов.

-P <размер>
устанавливает размер области захвата (snaplen) для пакетов.

-q
задает работу с минимальным выводом информации. Заставка и результаты инициализации не выводятся на экран.

-r tcpdump-file
задает чтение данных из собранного ранее файла в формате tcpdump (см. параграф 11.9.2 на стр. 262). Эта опция может быть полезна для анализа трафика с большим количеством фрагментов, собранного в файл tcpdump с

использованием дефрагментации.

-s
задает отправку тревожного сообщения в syslog. В системах Linux это сообщение будет сохранено в файле /var/log/secure, на большинстве остальных платформ - в файле /var/log/messages.

-S <переменная>=<значение>
задает значение указанной переменной из конфигурационного файла Snort. Например, вы можете указать в командной строке значение переменной HOME_NET.

-t chroot
изменяет корневой каталог Snort на каталог, указанный значением **chroot** после инициализации. Отметим, что при использовании этой опции имена журнальных файлов будут задаваться относительно каталога **chroot**.

-T
задает запуск Snort в режиме самотестирования с проверкой всех параметров командной строки. Этот ключ разумно задавать при использовании программы в режиме демона, чтобы быть уверенным в корректности переданных команде **snort** параметров.

-u <пользователь>
задает замену пользователя, от имени которого программа Snort будет работать после инициализации. Пользователь может быть указан именем или идентификатором UID.

-U
задает использование временных меток во всех журнальных файлах в формате UTC (время по Гринвичу).

-v
Задает подробный вывод информации. Содержимое пакетов будет выводиться на консоль. Этот режим существенно замедляет работу программы, поэтому его не следует применять при использовании Snort в качестве IDS (программа будет просто отбрасывать пакеты).

-V
выводит на экран информацию о номере версии и завершает работу программы.

-X
задает вывод дампа пакета данных, начиная с канального уровня. Этот флаг отменяет действие опции **-d**.

-y
задает программе необходимость включения года в сигналы тревоги и журнальные файлы.

-z
эта опция используется с препроцессором stream4 и обеспечивает возможности использования поддерживаемых stream4 функций stateful inspection для предотвращения использования обманок против Snort. По умолчанию Snort не принимает во внимание состояний TCP для пакетов, с которыми программа связывает сигналы тревоги. Ключ **-z** говорит программе Snort, что сигналы тревоги следует генерировать только для пакетов, относящихся к известным соединениям в состоянии **established**. Это позволяет программе Snort противостоять воздействию контр-IDS (например, программ stick или snot).

-?
выводит на экран справку о ключах и завершает работу программы.

11.6.1.2 Фильтрация пакетов

Программа snort в режиме захвата пакетов поддерживает мощный набор средств фильтрации пакетов. Синтаксис языка описания фильтров в точности соответствует синтаксису фильтров tcpdump, описанных в параграфе 11.9.2.2 (стр. 265).

11.6.1.3 Правила Snort

Snort использует гибкий и мощный язык правил для описания сигнатур сетевых пакетов и связанных с ними действий. В настоящее время поддерживается более 2500 правил, описать которые в данной книге не представляется возможным. На сайте http://www.snort.org/ вы сможете найти информацию о всех правилах Snort.

11.6.2 portsentry

http://sourceforge.net/projects/sentrytools/

PortSentry представляет собой программу-детектор сканирования портов и иных опасных действий в системе, выполняемых через сеть.

Программа **PortSentry** поддерживает множество опций детектирования сканеров портов и при обнаружения фактов сканирования может реагировать одним из перечисленных ниже способов:

- записывать сведения об инциденте в журнальный файл с использованием функций syslog (параграф 2.8.4 на стр. 49);
- автоматически блокировать сканирующий хост, помещая его адрес в файл **/etc/hosts.deny**;
- автоматически менять картину маршрутизации локального хоста с тем, чтобы сделать сканирующий хост недоступным;
- автоматически менять конфигурацию локального хоста для отбрасывания всех пакетов от сканирующего хоста с

использованием пакетных фильтров.

Задачей программы является обнаружение фактов сканирования и передача информации о них администратору хоста или сети. Для решения этой задачи существует немало программ[1], но portsentry отличается от них возможностью автоматического блокирования и детектированием фактов скрытого сканирования (stealth scan).

PortSentry поддерживает несколько режимов обнаружения скрытого сканирования:

1) наблюдение за портами из предопределенного списка;
2) наблюдение за всеми портами заданного диапазона, кроме тех, которые были связаны с сетевыми демонами в момент запуска **PortSentry**, или были исключены из наблюдения вручную. Этот способ может эффективно обнаруживать сканеры, но дает достаточно высокий уровень ложных срабатываний.

11.6.2.1 Установка PortSentry

Отредактируйте конфигурационный файл **portsentry_config.h** в соответствии с условиями будущего использования программы, задав значения перечисленных здесь переменных:

CONFIG_FILE - полный путь к конфигурационному файлу PortSentry.

WRAPPER_HOSTS_DENY - полный путь к файлу **hosts.deny**.

SYSLOG_FACILITY - тип сообщений[2] (facility, см. стр. 50), которые PortSentry будет передавать в syslog (см. параграф 2.8.4 на стр. 49).

SYSLOG_LEVEL - уровень **syslog** (см. стр. 50), используемый для отправки сообщений.

После этого отредактируйте файл **portsentry.conf**, установив подходящие для вашего случая значения описанных в таблице переменных.

Таблица 62. Переменные конфигурации PortSentry.

Переменная	Описание
`TCP_PORTS`	Список разделенных запятыми номеров портов TCP, которые должна прослушивать программа PortSentry. Список заключается в двойные кавычки и не должен содержать пробелов. Программа будет пытаться осуществить привязку[3] к сокетам всех указанных в списке портов (по умолчанию может привязываться до 64 портов). В режиме **Advanced Stealth Scan Detection** (стр. 253) этот параметр игнорируется.
`UDP_PORTS`	Список разделенных запятыми номеров портов UDP, которые должна прослушивать программа PortSentry. Список заключается в двойные кавычки и не должен содержать пробелов. Следует осторожно относиться к выбору портов UDP для прослушивания, поскольку злоумышленник может организовать фиктивную атаку с использованием различных номеров портов и подставных адресов, которые в результате окажутся заблокированными. На сайтах Internet не рекомендуется использовать такое прослушивание портов UDP. В режиме **Advanced Stealth Scan Detection** (стр. 253) этот параметр игнорируется.
`ADVANCED_PORTS_TCP`	Значение, указывающее максимальный номер прослушиваемого порта TCP. По умолчанию используется значение 1024, но вы можете указать любой номер порта от 0 до 65535. Прослушивать с помощью программы PortSentry порты с номерами более 1023 не рекомендуется, поскольку эти порты не являются привилегированными.
`ADVANCED_PORTS_UDP`	Максимальный номер порта UDP (см. предыдущую переменную).
`ADVANCED_EXCLUDE_TCP`	Список разделенных запятыми номеров портов TCP, которые должны быть исключены из прослушивания при работе в режиме **Advanced**. Обычно в этот список включают порты, к которым удаленные клиенты могут достаточно часто обращаться по ошибке[4], вызывая ложные срабатывания программы.
`ADVANCED_EXCLUDE_UDP`	Порты UDP, исключаемые из прослушивания в расширенном режиме (см. предыдущую переменную).
`IGNORE_FILE`	Полное имя файла, содержащего список игнорируемых адресов IP (см. стр.).
`BLOCKED_FILE`	Полное имя файла, содержащего список адресов IP для блокируемых хостов.
`RESOLVE_HOST`	Эта опция управляет преобразованием адресов в имена хостов с помощью службы DNS. Значение 1 включает преобразования, остальные значения отключают.

1 *Аннотированный список таких программ вы найдете на сайте http://www.softpanorama.org/Security/port_scan_detectors.shtml.*
2 *Те, кто хорошо знаком с работой syslog, могут установить для этого параметра значение LOG_LOCAL0 (вместо 0 вы можете задать другой локальный уровень) и, включив соответствующие строки в файл **syslog.conf**, записывать сообщения PortSentry непосредственно в свои файлы системы мониторинга.*
3 *В скрытом режиме работы программа не будет привязываться к портам сокетов, осуществляя вместо этого мониторинг попыток соединения с заданными портами на уровне сокетов.*
4 *Например, порты **ident**, **SSL** и т. п.*

Переменная	Описание
BLOCK_UDP	Эта опция управляет автоматическими откликами на зонды UDP (например, сканирование портов). Поскольку пакеты UDP достаточно просто фальсифицировать, автоматическая реакция позволит злоумышленникам заблокировать любые хосты путем организации сканирования портов UDP с подставными адресами отправителей пакетов. Установка нулевого значения для этой переменной будет отключать автоматическое блокирование и программа будет просто вносить соответствующие записи в журнальный файл. Для сайтов Internet целесообразно отключить режим автоматического блокирования адресов. Однако для хостов внутренней сети, имеет смысл сохранить блокировку в ответ на сканирование портов UDP, поскольку это позволит обнаружить несанкционированные действия внутренних пользователей.
BLOCK_TCP	Эта опция управляет автоматическими откликами на зонды TCP (например, сканирование портов). Для протокола TCP имеет смысл сохранить режим автоматического отклика (блокировки адреса), поскольку организовать фиктивные соединения TCP с подменой адресов отправителей гораздо сложнее, нежели фальсифицировать UDP-трафик.
KILL_ROUTE	Этот параметр определяет команду, которая будет использоваться для блокировки доступа атакующего хоста путем внесения соответствующего изменения в таблицу маршрутизации или локальные фильтры пакетов. Параметр должен содержать полный путь к команде изменения маршрута и все параметры, требуемые для изменения таблицы. Взамен макроса **$TARGET$** программа будет подставлять IP-адрес атакующего хоста, поэтому данный макрос является обязательным в команде. Конфигурационный файл, создаваемый при инсталляции программы содержит множество примеров команды **KILL_ROUTE** для различных операционных систем.
KILL_HOSTS_DENY	Этот параметр задает строку, добавляемую для адреса атакующего хоста в файл **hosts.deny**. Обязательный макрос **$TARGET$** заменяется программой адресом атакующего хоста. Параметр может включать также макрос **$PORT$**, который позволит блокировать доступ хоста только к подвергшимся атаке портам. Необязательный макрос **$MODE$** позволяет задать режим программы (**tcp**, **udp**, **stcp**, **sudp**, **atcp**, **audp**), в котором будет использоваться правило блокировки.
KILL_RUN_CMD	Этот параметр конфигурации позволяет задать команду, которая будет выполнена перед удалением маршрута к атакующему хосту или сразу после этой операции. Вы можете указать здесь любую программу. **Недопустимо с помощью таких программ выполнять какие-либо действия, направленные во вред атакующему хосту - вашей задачей является предотвращение атаки, а отнюдь не вендетта**. Необязательные макросы **$TARGET$**, **$PORT$** и **$MODE$** используются так же, как для предыдущих параметров.
KILL_RUN_CMD_FIRST	Нулевое значение этого параметра задает выполнение команды, указанной предыдущим параметром до внесения изменений в таблицу маршрутизации или фильтры, а **1** - после.
SCAN_TRIGGER	Этот параметр задает уровень срабатывания для программы **PortSentry**. Машина состояний программы сохраняет информацию о подключениях хостов. Значение этого параметра задает число обращений хоста к порту, после превышения которого PortSentry воспринимает это как сканирование или иной несанкционированный доступ. Нулевое значение параметра означает реакцию программы на первое обращение к порту и может приводить к ложным срабатываниям. Установка значения 1 или 2 снизит количество ложных срабатываний без существенной потери чувствительности программы. Значения 3 и больше существенно снижают чувствительность детектора.
PORT_BANNER	Текстовое сообщение, которое будет отправляется атакующему хосту в пакетах при срабатывании триггера PortSentry. Этот параметр необязателен и вы можете поставить в начале строки знак комментария. Если вы решите изменить принятое по умолчанию сообщение, не включайте в свой текст оскорбительных сообщений. Заданное этим параметром сообщение не передается при использовании программы в режиме детектора скрытого сканирования.

После завершения работы с конфигурационным файлом следует отредактировать файл **portsentry.ignore**, содержащий список адресов хостов, которые не принимаются во внимание при обнаружении попыток вторжения или сканирования портов. Этот файл в любом случае должен содержать адрес loopback-интерфейса **127.0.0.1** и IP адрес данного хоста. Нецелесообразно включать в список все хосты локальной сети, разумней указать в файле внутреннюю подсеть с формате:

```
<IP-сеть>/<маска>
```

Не забывайте, что включив адрес хоста в список игнорируемых, вы теряете всякую возможность контроля за действиями этого хоста с помощью программы PortSentry. Не следует a-priori считать дружественными все компьютеры локальной сети, это может привести к весьма тяжелым последствиям. Мне пришлось однажды столкнуться с ситуацией, когда параноидально настроенный администратор сети в моей компании занимался сканированием компьютера руководителя фирмы на предмет обнаружения анализаторов, перехватывающих в сети пакеты с частной перепиской этого администратора. Обнаружен этот факт был именно с помощью PortSentry.

После завершения работы с файлами нужно скомпилировать и установить программу. По умолчанию программа

устанавливается в каталог **/usr/local/psionic/portsentry**. Если вас не устраивает это, можно изменить каталог, указав нужно имя в файле **Makefile**. В таких случаях не забудьте указать имя каталога программы в файлах **portsentry.conf** и **portsentry_config.h**.

Для компиляции и установки программы служат команды **make** и **make install**, соответственно.

11.6.2.2 Режимы работы программы

После завершения установки вы можете загрузить программу в желаемом режиме:

Команда	Режим
portsentry -tcp	Базовый режим мониторинга TCP с привязкой к портам.
portsentry -udp	Базовый режим мониторинга UDP с привязкой к портам.
portsentry -stcp	Режим детектирования скрытого сканирования портов TCP.
portsentry -atcp	Расширенный режим детектирования скрытого сканирования портов TCP.
portsentry -sudp	Режим детектирования скрытого сканирования портов UDP.
portsentry -audp	Расширенный режим детектирования скрытого сканирования портов UDP.

Отметим, что команда может включать два режима для разных протоколов (по одному для TCP и UDP), но не допускается задание в командной строке двух режимов работы для одного протокола. Например, вы можете использовать команду

```
portsentry -atcp -sudp
```

но команда

```
portsentry -tcp -stcp
```

является некорректной.

11.6.2.2.1 Базовый режим мониторинга TCP (-tcp)

В этом режиме PortSentry будет просматривать конфигурационный файл и привязывать все указанные в нем порты TCP для фонового мониторинга соединений. Если вы хотите увидеть результат инициализации программы в этом режиме, нужно посмотреть файл syslog[1], в который передаются сообщения.

11.6.2.2.2 Базовый режим мониторинга UDP (-udp)

В этом режиме PortSentry будет просматривать конфигурационный файл и привязывать все указанные в нем порты UDP для фонового мониторинга соединений. Если вы хотите увидеть результат инициализации программы в этом режиме, нужно посмотреть файл syslog, в который передаются сообщения.

11.6.2.2.3 Режим Stealth TCP (-stcp)

В этом режиме программа PortSentry будет использовать raw-сокет (Приложение 12.8) для мониторинга всех входящих пакетов TCP. Если входящий пакет адресован в один из контролируемых программой портов, PortSentry активизирует блокировку передавшего пакет хоста. Этот режим позволяет обнаруживать сканирование путем попыток организации соединений (connect), в также сканирования с помощью пакетов с флагами SYN/FYN и с помощью полуоткрытых соединений.

11.6.2.2.4 Режим "Stealth" UDP (-sudp)

Этот режим подобен режиму **Stealth TCP** и обеспечивает мониторинг пакетов, направленных в порты UDP, перечисленные в конфигурационном файле. Данный режим не использует каких-либо сокетов и не обеспечивает в реальности детектирования скрытого сканирования (обычно такое сканирование и не используется для протокола UDP), просто этот режим похож на детектирование скрытого сканирования TCP и обеспечивает реакцию на любой пакет UDP.

11.6.2.2.5 Расширенные режимы детектирования скрытого сканирования

11.6.2.2.5.1 Режим Advanced TCP stealth scan detection (-atcp)

В этом режиме PortSentry будет создавать список всех портов, перечисленных в конфигурационной опции **ADVANCED_PORTS_TCP**, с учетом списка исключений (портов, которые реально используются службами контролируемого хоста; например, SMTP, HTTP и т. п.). Блокироваться в этом режиме будет любой хост, направивший пакет в порт из списка контролируемых, если этот порт не относится к числу исключенных.

1) Этот режим является наиболее чувствительным и эффективным. Он обеспечивает незамедлительную реакцию на сканирование портов и практически не дает ложных срабатываний.

Незамедлительная реакция на сканирование все же может приводить к ложной блокировке, если вы недостаточно продумали список портов для мониторинга. Например, сервер FTP может передать вашему хосту запрос **ident** и, если вы осуществляете мониторинг порта **ident** (113 TCP), программа заблокирует этот сервер FTP, когда вы обратитесь к нему!

1 *Обычно это файл **/var/log/messages**.*

PortSentry осуществляет в этом режиме достаточно интеллектуальный мониторинг портов. Для некоторых протоколов (в частности, FTP) клиент после обращения к серверу открывает порт с номером из диапазона 1024-65535 и сервер организует соединение с этим портом. Такое поведение может приводить к ложным срабатываниям детектора сканеров, поэтому PortSentry будет просматривать входящие соединения и определять пакеты, адресованные в такие “временные” порты, не блокируя доступ к ним. После разрыва соединения порт будет снова включен в список мониторинга с блокировкой обращающихся к этому порту хостов.

11.6.2.2.5.2 Режим Advanced UDP "stealth" scan detection (-audp)

Этот режим подобен режиму **Advanced TCP stealth scan detection**, но работает с портами UDP и, следовательно, может давать множество ложных срабатываний. Это связано с тем, что программа PortSentry не может отличать широковещательный трафик от направленного. Например, если ваш маршрутизатор использует широковещательную рассылку пакетов RIP в направлении локальной сети, адрес этого маршрутизатора будет включен программой в список блокировки. Поэтому данный режим следует использовать с осторожностью и указывать все потенциально “опасные” порты в список исключений (параметр конфигурационного файла **ADVANCED_EXCLUDE_UDP**).

11.6.2.3 Проверка инсталляции

После первого запуска программы PortSentry в журнальном файле будут записаны сообщения при инициализации программы, подобные приведенным ниже:

```
Oct  9 09:11:35 nemesis portsentry[1644]: adminalert: portsentry is starting.
Oct  9 09:11:36 nemesis portsentry[1644]: adminalert: Going into listen mode on TCP port:
143
. . .
Oct  9 09:11:37 nemesis portsentry[1644]: adminalert: PortSentry is now active and
listening.
```

Последняя строка говорит о корректности установки и успешной загрузке программы PortSentry. Отсутствие этой строки говорит об ошибке, допущенной при инсталляции или настройке программы.

При использовании программы в стандартном режиме журнальный файл должен содержать информацию о всех портах, с которым связана программа PortSentry. Если привязку к какому-либо из портов не удалось осуществить, в журнальный файл записывается предупреждение об этом. Если не удалось привязать программу ни к одному из указанных в конфигурационном файле портов, в файл помещается сообщение об ошибке.

В режиме **Advanced stealth scan detection** программа будет указывать в журнальном файле список портов, которые она исключает из прослушивания.

Убедившись в нормальной загрузке программы, вы можете проверить ее работу с другого хоста с помощью сессии **telnet** для одного из контролируемых программой портов. Не выполняйте такую проверку с единственной точки доступа к данному хосту, поскольку при нормальном функционировании программы она просто заблокирует доступ. В результате проверки ы журнальном файле появятся записи, подобные приведенным ниже:

```
Oct  9 09:12:44 nemesis portsentry[1644]: attackalert: Connect from host: 123.345.56.78
to TCP port: 143
Oct  9 09:12:46 nemesis portsentry[1644]: attackalert: Host
server.haxor.org/123.345.56.78 has been blocked via dropped route.
Oct  9 09:12:46 nemesis portsentry[1644]: attackalert: Host
server.haxor.org/123.345.56.78 has been blocked via wrappers.
```

Не забудьте снять блокировку, созданную программой в результате тестирования, иначе с использованного для проверки хоста у вас больше не будет доступа в проверенную систему.

11.6.2.4 Сообщения программы

Программа использует три типа префиксов для обозначения уровня важности сообщения и упрощения поиска сообщений в журнальных файлах

adminalert - сообщение о состоянии PortSentry;

securityalert - сообщение о событии, связанном с безопасностью хоста;

attackalert - сообщение о срабатывании детектора сканирования.

11.6.2.5 Файлы программы

Все заблокированные программой хосты включаются в файл блокировки **portsentry.blocked.***. Вместо звездочки указывается префикс, определяемый протоколом и режимом программы, в котором произошло срабатывание детектора. Кроме того, информация о блокировке хоста сохраняется в файле **portsentry.history**. При каждом запуске файлы блокировок PortSentry удаляются, а файл **history** сохраняется и записи добавляются в конец этого файла при каждом случае блокировки хоста.

11.6.3 Courtney

11.7 Средства поиска анализаторов протоколов

Обнаружение работающих в системе анализаторов протоколов является достаточно сложной задачей. Ниже приводится краткое описание некоторых программ, способных помочь при обнаружении анализаторов. Отметим программы **ifconfig** (стр. 196) и **ifstatus** (стр. 208), позволяющие обнаружить сетевые интерфейсы, работающие в режиме захвата, а также пакет **chkrootkit** (стр. 238), способный в некоторых случаях обнаруживать следы работы анализаторов протоколов.

11.7.1 Sniffdet

http://sniffdet.sourceforge.net

Пакет Sniffdet - свободно распространяемый в исходных кодах набор программ для обнаружения удаленных анализаторов протоколов в среде TCP/IP. Пакет включает гибкую и простую в использовании библиотеку и программу обнаружения анализаторов.

Sniffdet использует различные проверки для обнаружения компьютеров, интерфейсы которых работают в режиме захвата. Кроме функций общего пользования библиотека **libsniffdet** обеспечивает поддержку тестов:

- ICMP;
- ARP;
- DNS;
- LATENCY (задержка откликов ICMP).

Программа обнаружения имеет гибкий интерфейс, позволяющий настраивать все тесты с помощью конфигурационного файла. Поддерживается возможность одновременной проверки нескольких хостов и plugin-интерфейс для подготовки отчетов или работы без привилегий пользователя root.

Синтаксис

```
sniffdet [options] TARGET
```

11.7.1.1 Опции

Проверяемый хост может быть задан его полным именем или адресом IPv4. Опции команды перечислены в таблице 63.

Таблица 63. Опции sniffdet.

Опция		Описание
-i	**--iface=**	Задает используемый для тестирования интерфейс. По умолчанию **eth0**.
-l	**--log=**	Указывает имя журнального файла для записи отчета. По умолчанию журнальный файл не используется.
-c	**--configfile=**	Указывает конфигурационный файл, используемый программой для загрузки параметров. По умолчанию **/etc/sniffdet.conf**.
-f	**--hostsfile**	Указывает имя файла с информацией о проверяемых хостах. Файл должен содержать по одному адресу или имени проверяемого хоста в каждой строке. Строки комментариев должны начинаться с символа **#**.
-u	**--uid=**	Задает идентификатор пользователя для программы после выхода из режима root. По умолчанию используется UID=280 (задан в конфигурационном файле).
-g	**--gid=**	Задает идентификатор группы для программы после выхода из режима root. По умолчанию используется GID=280 (задан в конфигурационном файле).
-t	**--test=**	Задает имя выполняемого программой теста или список таких имен. Поддерживаются тесты: **dns** - проверка DNS; **arp** - проверка откликов ARP; **icmp** - проверка откликов на **ping**; **latency** - проверка задержки откликов ICMP.
	--pluginsdir=	Указывает каталог, в котором программа ищет подключаемые модули (plugin).
-p	**--plugin=**	Указывает подключаемый программой модуль (**xml**, **stdout** и т. п.)
-f	**--targetsfile=**	Задает сканирование всех хостов, перечисленных в указанном файле.
-v	**--verbose**	Задает вывод дополнительной информации.
-s	**--silent**	Отключает вывод информации на консоль.
-h	**--help**	Выводит справку о работе с программой.
	--version	Выводит номер версии программы.

11.7.1.2 Конфигурационный файл sniffdet.conf

Конфигурационный файл **sniffdet** позволяет настроить параметры выполняемых программой проверок. По

умолчанию конфигурационные параметры хранятся в файле **/etc/sniffdet.conf**.

Синтаксис конфигурационного файла очень прост. Каждая секция имеет имя и набор строк параметров, заключенных в фигурные скобки {}. Внутри каждой секции размещаются строки параметров в формате

переменная = значение

и строки комментариев, начинающиеся с символа **#**. Комментарием считается содержимое строки от символа **#** до конца строки. Пустые строки не принимаются во внимание.

11.7.1.2.1 Пример конфигурационного файла

Приведенный ниже пример конфигурационного файла содержит значения некоторых параметров, используемые программой по умолчанию. Для использования программы рекомендуется создать файл конфигурации в соответствии с вашими задачами.

Приведенный здесь пример конфигурационного файла не содержит описания всего набора тестов, поскольку они содержат однотипные переменные и вы сможете создать соответствующие секции своего конфигурационного файла по аналогии с приведенными ниже.

```
# Пример конфигурационного файла snifdet
# http://sniffdet.sourceforge.net
#
# см. руководство man 5 sniffdet.conf

# глобальные параметры конфигурации
global {
     verbose = 0;
     # переменная logtype может принимать значения FILE, STDOUT, STDERR, SYSLOG
     logtype = FILE;
     # имя журнального файла, используемое по умолчанию
     logfile = "sniffdet.log";
     # каталог, в котором хранятся дополнительные модули
     plugins_dir = "/usr/lib/sniffdet/plugins";
     # имя подключаемого модуля
     plugin = "stdout.so";
     # идентификатор пользователя с которым программа работает после выхода из режима
root
     UID = 280;
     # идентификатор группы с которым программа работает после выхода из режима root
     GID = 280;
     # имя интерфейса, используемого программой
     iface = "eth0";
     # значение фиктивного MAC-адреса
     fake_hwaddr = {0xff, 0x00, 0x00, 0x00, 0x00, 0x00};
     # фиктивный адрес IP
     fake_ipaddr = "192.168.1.100";
}

# переменные теста icmp
icmptest {
     # выбор интерфейса (пока не поддерживается)
     #iface = "eth0";
     timeout = 20;         # время ожидания в секундах
     tries = 10;           # число попыток
     interval = 1000       # интервал в миллисекундах
     # значение фиктивного MAC-адреса
     fake_hwaddr = {0xff, 0x00, 0x00, 0x00, 0x00, 0x00};
}

# переменные теста arp
arptest {
     # выбор интерфейса (пока не поддерживается)
     #iface = "eth0";
     timeout = 20;         # время ожидания в секундах
     tries = 10;           # число попыток
     interval = 1000       # интервал в миллисекундах
     # значение фиктивного MAC-адреса
     fake_hwaddr = {0xff, 0x00, 0x00, 0x00, 0x00, 0x00};
}

# переменные теста dns
dnstest {
     # выбор интерфейса (пока не поддерживается)
     #iface = "eth0";
     timeout = 20;         # время ожидания в секундах
     tries = 10;           # число попыток
```

```
        interval = 1000       # интервал в миллисекундах
        # значение фиктивного адреса IP
        fake_ipaddr = "10.0.0.10"
        # значение фиктивного MAC-адреса
        fake_hwaddr = {0x46, 0x0f, 0xA4, 0x33, 0x11, 0xD1};
        sport = 22;           # порт отправителя
        dport = 22;           # порт получателя
        # содержимое пакетов (пока не поддерживается)
        #payload = "login: foobar";
}

# latency test variables
latencytest {
        # выбор интерфейса (пока не поддерживается)
        #iface = "eth0";
        timeout = 300;        # время ожидания в секундах
        interval = 1500       # интервал в миллисекундах
        # флаги TCP (поддерживаются флаги SYN, FIN, RST, ACK, PUSH и URG)
        tcpflags = SYN;
        # содержимое пакетов (пока не поддерживается)
        #payload = "";
}

# опции подключаемых модулей
plugins {
        xmlplugin_filename = "xmloutput.xml"
}
# EOF
```

11.7.1.3 Библиотека libsniffdet

Библиотека **libsniffdet** может быть полезна при разработке собственных программ поиска удаленных анализаторов протоколов или хостов, чьи интерфейсы работают в режиме захвата. Полную документации для библиотеки вы сможете найти на сайте http://sniffdet.sourceforge.net.

11.7.1.4 Примеры использования

Команда

```
sniffdet -i eth1 -t dns,arp,icmp foo.localdomain
```

проверяет хост **foo.localdomain** с помощью тестов **dns**, **arp** и **icmp**, используя интерфейс **eth1**.

Для проверки хоста **foo.localdomain** с помощью теста **latency** через интерфейс **eth0** и вывода результатов с использованием подключаемого модуля **xml** можно воспользоваться командой

```
sniffdet -i eth0 -t latency foo.localdomain —plugin=xml
```

11.8 Программы анализа системных журналов

11.8.1 logcheck

Пакет **logcheck** предназначен для автоматической проверки системных журналов на предмет обнаружения опасных событий (security violations) и необычных действий. Logcheck использует программу logtail, которая помнит позицию последнего чтения журнального файла и при последующем запуске начинает обработку с этой позиции. Пакет **logcheck** построен на основе сценария frequentcheck.sh из состава брандмауэра Trusted Information Systems[1] Gauntlet[tm] с разрешения авторов.

Аудит журнальных файлов системы весьма актуален с точки зрения безопасности. Что может быть важнее для администратора, чем своевременное получение информации о возникновении в системе проблем, которые могут в дальнейшем осложнить ее функционирование или совсем блокировать нормальную работу системы.

Практически все современные реализации Unix поддерживают тип сообщений syslog для передачи отчетов и обеспечивают достаточный уровень гибкости, предоставляя администратору информацию о всех важных событиях в системе. Программа **logcheck** автоматически выполняет работу по аудиту системных процессов на основе журнальных файлов и показывает администратору проблемные точки в системе.

Работа программы logcheck основана на периодической проверке журнальных файлов с целью обнаружения опасных и необычных ситуаций в работе системы. При обработке журнальных файлов используются два основных метода уведомления администратора:

1) генерация отчетов обо всех событиях, заданных администратором ключевыми словами;
2) генерация отчетов обо всех событиях, которые администратор не счел нужным игнорировать; такие события также задаются ключевыми словами.

Такая обработка журнальных файлов обеспечивает гарантию того, что важная и интересующая вас (заданная

1 *http://www.tis.com*

ключевыми словами) информация не будет оставлена без внимания.

Сценарий **logcheck** следует запускать по крайней мере один раз в час с использованием демона cron. Все обнаруженные события указываются в сообщении, передаваемом администратору по электронной почте.

Программа распространяется с базовым набором ключевых слов для различных вариантов ОС, но вы можете создать свои наборы ключевых слов для их поиска в журнальных файлах системы. Редактировать файлы ключевых слов можно с помощью привычного вам редактора. Существует модуль Security Sentries для редактирования ключевых слов и режима работы logcheck в программе Webmin (параграф 11.3 на стр. 228). Интерфейс этого модуля показан на рисунке 11.24.

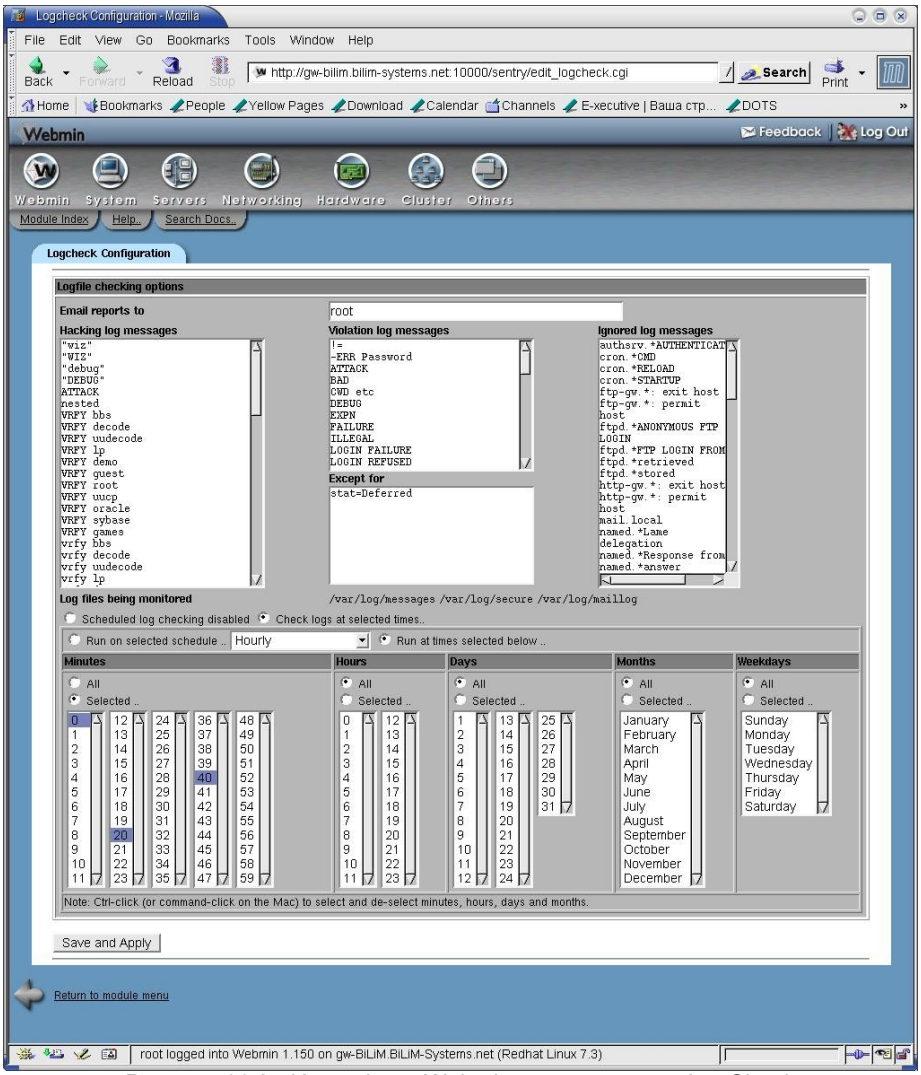


Рисунок 11.24 Интерфейс Webmin для настройки LogCheck

Пакет Logcheck включает несколько файлов:

logcheck.sh - основной сценарий, используемый для просмотра журнальных файлов и генерации сообщений для администратора.

logtail - исполняемый файл, который сохраняет информацию о позициях журнальных файлов, на которых была завершена обработка при прошлом запуске сценария. При каждом запуске обработка файла начинается с сохраненной позиции, что позволяет существенно ускорить процесс анализа и снизить загрузку системы. Особенно важна эта возможность при обработке больших журнальных файлов на маршрутизаторах и межсетевых экранах. Периодическое обновление журнальных файлов утилитой logrotate, поэтому при переходе к новому файлу счетчики смещения автоматически сбрасываются.

logcheck.hacking - содержит список ключевых слов, по которым идентифицируются возможные атаки на вашу систему. Этот файл не требуется редактировать, если только вы не сочтете нужным добавить в него новые ключевые фразы, которые обнаружите в своих журнальных файлах после атаки. Включенный по умолчания список содержит ключевые фразы, генерируемые сканерами безопасности и программой sendmail (при некорректном синтаксисе строки адреса). При обнаружении в журнальном файле заданных ключевых слов будет генерироваться предупреждение с заголовком **ACTIVE SYSTEM ATTACK**

logcheck.violations - содержит ключевые слова для идентификации системных событий, которые представляются нежелательными. Связанные с ключевыми словами из этого файла появляются в отчете под заголовком **Security Violations**.

logcheck.violations.ignore - этот файл содержит ключевые слова, при наличии которых запись из журнального файла не включается в раздел отчета **Security Violations**. Покажем это на примере обработки строк:

```
Feb 28 21:00:08 nemesis sendmail[5475]: VAA05473: to=crowland, ctladdr=root (0/0),
delay=00:00:02, xdelay=00:00:01, mailer=local, stat=refused
Feb 28 22:13:53 nemesis rshd: refused connect from hacker@evil.com:1490
```

Первая запись описывает достаточно часто встречающуюся ситуацию, когда удаленный транслятор по каким-то причинам отказался от организации соединения с вашей системой (**stat=refused**). Во второй записи говорится, что некто (hacker@evil.com) безуспешно пытался инициировать сессию **rsh** на вашей машине - это, скорей всего, плохо. Файл **logcheck.violations** включает ключевое слово **refused** и обе записи должны были появиться в отчете. Однако мы можем избавиться от первой записи (таких записей может быть весьма много и они не содержат актуальной информации), поместив в файл **logcheck.violations.ignore** ключевые слова

```
mailer=local, stat=refused
```

В результате отчет будет содержать лишь вторую запись из журнального файла.

При включении ключевых слов в файл **logcheck.violations.ignore** следует соблюдать осторожность. Слишком короткая фраза, на основании которой запись из журнального файла будет пропущена, может привести к тому, что вы не увидите достаточно важных событий.

logcheck.ignore - этот файл содержит ключевые слова, при наличии которых в записях журнального файла не включаются ни в один раздел. Все строки журнальных файлов, которые не содержат ключевых слов из этого файла

и не были включены в другие разделы отчета, помещаются в раздел **Unusual System Activity**. Следует с осторожностью подходить к выбору ключевых фраз для этого файла и не включать к него слишком коротких ключей поиска.

11.8.2 swatch

http://swatch.sourceforge.net

SWATCH[1] - это сценарий на языке Perl[2], написанный Тодом Эткинсом (Todd Atkins) для мониторинга журнальных файлов в реальном масштабе времени. Swatch просматривает log-файлы и генерирует заданные администратором отчеты при срабатывании указанных администратором триггеров.

Программа чрезвычайно проста и распространяется с установочным сценарием, который копирует все требуемые библиотеки, документацию и сценарии Perl в соответствующие каталоги. В зависимости от имеющегося в системе набора модулей Perl вам может потребоваться установка дополнительных модулей - сценарий инсталляции swatch уведомит вас о такой необходимости. После завершения установки вам достаточно будет отредактировать конфигурационный файл в соответствии со своими потребностями и запустить программу swatch.

Для эффективной работы программы Swatch требуется конфигурационный файл с шаблонами поиска и операциями, которые должны выполняться при обнаружении записей, соответствующих заданным шаблонам.

Конфигурационный файл **swatchrc** определяет все аспекты работы программы, задавая имена журнальных файлов для просмотра, триггеры и т. п. Программа Swatch выполняет поиск с использованием регулярных выражений (триггеры), заданных в файле swatchrc. При обнаружении соответствия выполняется процедура уведомления администратора, определенная в файле **swatchrc**. Программа Swatch обеспечивает мониторинг журнальных файлов в реальном масштабе времени с использованием команды

```
/usr/bin/tail -f
```

11.8.2.1 Опции

`--config-file=<имя файла> (-c <имя файла>)`
задает имя конфигурационного файла. По умолчанию используется файл **${HOME}/.swatchrc**.

`--help`
выводит на экран справочную информацию о работе с программой.

`--input-record-separator=regular_expression`
задает использование регулярного выражения для определения границы каждой записи. По умолчанию в качестве границы записи принимается символ возврата каретки (CR).

`--restart-time=[+]hh:mm[am|pm] (-r [+]hh:mm[am|pm])`
задает перезапуск программы в указанное время (часы и минуты). Если флаг **am/pm** не задан, используется 24-часовой формат. Если перед значением времени указан знак **+** программа будет перезапущена по истечении заданного периода от момента ввода команды. В этом случае флаг **am/pm** игнорируется.

`--script-dir=<каталог>`
задает каталог для записи временных сценариев, которые служат в качестве “сторожа” и обычно записываются в домашний каталог пользователя. Не используйте для временных сценариев каталоги, куда разрешена запись любому пользователю (например, /tmp).

`--version (-V)`
выводит информацию о номере версии программы.

`--use-cpan-file-tail`
задает использование Perl-модуля **File::Tail** взамен системной команды tail. При использовании этого модуля поддерживаются дополнительные опции:

`--tail-file=<имя файла> (-t <имя файла>)`
проверять текстовые строки сразу после их добавления в файл;

`--read-pipe=<команда> (-p <команда>)`
проверять каналы ввода;

`--examine=<имя файла> (-f <имя файла>)`
задает имя файла для проверки. Swatch будет выполнять однократный просмотр этого файла.

Опция

`--dump-script[=<имя файла>]`
используется для отладки и задает запись временного сценария в файл или **STDOUT** вместо исполнения этого сценария.

Если команда **swatch** используется без каких-либо опций, это эквивалентно набору опций

```
swatch --config-file=~/.swatchrc --tail-file=/var/log/syslog
```
или (если существует файл /var/log/messages)

```
swatch --config-file=~/.swatchrc --tail-file=/var/log/messages
```
При отсутствии конфигурационного файла будет использоваться конфигурация:

1 *The **S**imple **WATCH**er and filter - простой сторож и фильтр.*
2 *Вы найдете исходные тексты программы в каталоге SRC/ приложенного к книге компакт-диска.*

```
watchfor  /.*/
    echo
```

11.8.2.2 Конфигурационный файл

Конфигурационный файл используется программой **swatch** для определения шаблонов поиска информации и действий, выполняемых при совпадении с шаблоном поиска. Каждая строка конфигурационного файла содержит ключевое слово и значение[1] для этого слова. Между ключевым словом и значением ключа помешается пробел или знак равенства (**=**).

11.8.2.2.1 Поиск соответствий

Команды поиска соответствий задают режим обнаружения или пропуска (игнорирования) строк, соответствующих регулярному выражению[2].

watchfor regex
задает поиск в журнальных файлах соответствия с заданным регулярным выражением.

ignore regex
задает игнорирование при просмотре журнальных файлов строк, соответствующих заданному регулярному выражению.

11.8.2.2.2 Операции при соответствии

При обнаружении соответствия регулярному выражению программа выполняет операции, заданные в конфигурационном файле для этого регулярного выражения.

echo [mode]
задает вывод совпадающей строки. Необязательный параметр **mode** позволяет управлять атрибутами текста (**normal** - обычный шрифт, **bold** - полужирный шрифт, **underscore** - подчеркивание, **blink** - мигающие символы, **inverse** - инверсия цвета, **black** - черный, **red** - красный, **green** - зеленый, **yellow** - желтый, **blue** - синий, **magenta** - фуксиновый, **cyan** - бирюзовый, **white** - белый, **black_h**[3], **red_h**, **green_h**, **yellow_h**, **blue_h**, **magenta_h**, **cyan_h**, **white_h**). По умолчанию используется режим **normal**.

bell [N]
задает подачу N звуковых сигналов при обнаружении соответствия. По умолчанию выдается 1 гудок.

exec command
задает выполнение указанной команды. Команда может содержать переменные, которые при вызове будут заменяться полями соответствующей регулярному выражению записи. Например, **$N** будет заменяться N-нам полем строки, **$0** и **$*** полной строкой.

mail [addresses=address:address:...][,subject=your_text_here]
задает передачу по указанному адресу или группе адресов сообщения, содержащего найденные строки. По умолчанию сообщение передается по адресу пользователя программы.

pipe command[,keep_open]
передает найденные строки в заданную команду с использованием канала (Pipe). Если требуется сохранение канала открытым, пока используется другая операция с каналом, включите в строку опцию **keep_open**.

write [user:user:...]
задает передачу найденной строки пользователям с помощью команды **write**.

throttle hours:minutes:seconds,[use=message|regex|<regex]>
Эта команда опция позволяет ограничить число операций, выполняемых при совпадении с шаблоном. Опция **use=regex** задает использование регулярного выражения взамен сообщения.

threshold events:seconds,[repeat=no|yes]
Эта опция позволяет ограничить число выполнения операции при совпадении с шаблоном поиска в течение определенного интервала времени. Используемая по умолчанию команда **threshold 4:60** не позволит выполнять операцию более 4 раз в секунду. Опция **repeat=no** будет отключать сброс таймера при достижении порогового значения. По умолчанию (**repeat=yes**) при наступлении нового кадра отсчета времени счетчик сбрасывается и операция может выполняться снова, пока не будет в очередной раз достигнут порог.

Команда **threshold** похожа но команду **throttle**, но отличается от нее тем, что **throttle** показывает первую строку, игнорируя остальные, а **threshold** показывает последнюю строку, игнорируя предшествующие.

continue
Эта команда заставляет программу **swatch** продолжать попытки поиска других соответствий после завершения текущего блока операций.

quit
завершает работу программы **swatch**.

Специальная опция

when=day_of_week:hour_of_day
может использоваться с любой из перечисленных выше команд за исключением **throttle** и **threshold**. Опция задает

1 *В некоторых случаях значения являются необязательными.*
2 *Краткая информация о регулярных выражениях UNIX приводится в Приложении 12.1.*
3 *Идентификаторы цветов с префиксом **_h** задают тон подсветки.*

временные рамки (дни недели и часы) для выполнения операции. Например,

```
mail=sysad-pager@somehost.somedomain,when=1-6:8-17
```

11.8.2.3 Пример конфигурации

```
# Swatch configuration file for Linux box
#
# Last Modified 7 April, 2000
# Lance Spitzner <lance@spitzner.net>
#
# swatch -c /etc/swatchrc -t /var/log/messages
#

### Snort honeypot alerts from firewall
watchfor /IDS/
        echo bold
        mail addressess=admin,subject=--- Snort IDS Alert ---
        exec echo $0 >> /var/log/IDS-scans
        throttle 01:00 use=IDS27

watchfor /PORTSCAN DETECTED/
        echo bold
        mail addresses=admin,subject=--- Snort Port Scan Alert ---
        exec echo $0 >> /var/log/IDS-scans

### DNS zone transfers
watchfor /approved AXFR/
        echo bold
        mail addresses=admin,subject=--- Zone transfer Alert ---
        exec echo $0 >> /var/log/IDS-scans

############################################################
#                        EXAMPLES                          #
############################################################

### Bad login attempts
# watchfor   /failed/
#        echo bold
#        mail addressess=root,subject=Failed Authentication

### Some is sniffing!
# watchfor   /promiscuous/
#        echo bold
#        mail addressess=root,subject=Someone is sniffing the network!

### Ignore this stuff
# ignore   /sendmail/,/nntp/,/xntp|ntpd/,/faxspooler/

### Kernel problems or system reboots
# watchfor   /(panic|halt|SunOS Release)/
#        echo bold
#        mail addresses=root,subject=System Panic,Halt, or Reboot!

# watchfor   /file system full/
#        echo bold
#        mail addresses=root,subject=File system Full
#        throttle 01:00

# watchfor   /su:/
#        echo bold
#        mail addresses=root,subject=Someone sued to root access
```

11.9 Средства мониторинга сетевого трафика и анализа пакетов

11.9.1 Библиотека pcap

http://www.tcpdump.org

Библиотека **pcap**[1] входит в состав большинства дистрибутивов UNIX/Linux и обеспечивает поддержку функций сбора (копирования на лету) сетевых пакетов для последующего анализа. Проходящие через хост или маршрутизатор пакеты доступны для функций pcap независимо от местоположения их отправителя и получателя. Обеспечиваемые библиотекой функции позволяют создавать анализаторы протоколов и другие программы, для

1 *Packet Capture - захват пакетов. Эту библиотеку часто называют* ***libpcap****.*

работы которых требуется сбор пакетов.

Мы не будем здесь останавливаться на описании функций библиотеки, поскольку они представляют интерес главным образом для разработчиков программ[1]. Упомянута библиотека прежде всего потому, что без нее не будут работать многие приложения, предназначенные для мониторинга и анализа сетевого трафика. Поэтому без раздумий устанавливайте эту библиотеку на своем шлюзе.

11.9.2 tcpdump

http://www.tcpdump.org

Программа **tcpdump**, включаемая во все дистрибутивы UNIX, выводит заголовки пакетов для сетевого интерфейса в соответствии с заданным логическим выражением. Программа также допускает использование с флагом **-w** для записи пакетов данных в файл, которым может впоследствии использоваться для анализа. Возможен и просмотр заголовков из таких файлов с помощью флага **-r**. Во всех случаях tcpdump имеет дело только с пакетами, соответствующими заданному логическому выражению (фильтру).

Tcpdump (если в команде не был указан флаг **-c**) продолжает собирать пакеты до тех пор, пока процесс не будет прерван сигналом SIGINT (например, при нажатии клавиш control-C) или SIGTERM (например, в результате команды kill(1)). Если команда используется с флагом **-c**, сбор пакетов кроме описанных выше способов может быть прекращен также после обработки определенного числа пакетов.

При завершении работы tcpdump выводит значения счетчиков:

- собранных (captured) пакетов (число пакетов, полученных и обработанных tcpdump);
- полученных фильтром (received by filter) пакетов; толкование этого значения зависит от ОС, под управлением которой работала программа tcpdump (в некоторых ОС указывается число пакетов независимо от числа совпадений с условиями фильтрации, а в других - число пакетов, соответствующих фильтрам);
- отброшенных ядром (dropped by kernel) пакетов (число пакетов, отброшенных ядром по причине нехватки ресурсов или фильтрации внутри ядра).

На платформах, поддерживающих сигналы SIGINFO (например, BSD), могут выводиться значения перечисленных выше счетчиков по сигналу SIGINFO (этот сигнал может быть подан обычно с помощью клавиш control-T) без прерывания работы команды.

Отметим, что чтение пакетов из сетевого интерфейса может потребовать от пользователя специальных привилегий в зависимости от используемой ОС:

- **SunOS 3.x** или **4.x** с NIT или BPF
  требуется доступ для чтения к файлам устройств **/dev/nit** или **/dev/bpf***.
- **Solaris** с DLPI
  требуется доступ для чтения и записи к сетевому псевдо-устройству (например, **/dev/le**). На некоторых версиях Solaris таких прав недостаточно для работы **tcpdump** в режиме захвата[2]; в таких ситуациях для использования tcpdump требуются полномочия root или установка для tcpdump флага **SUID** (см. параграф 2.3.3 на стр. 42). Отметим, что на многих (возможно, на всех) системах при работе устройства в обычном режиме вы не сможете видеть никаких исходящих пакетов, поэтому сбор данных в таком режиме может оказаться практически бесполезным.
- **HP-UX** с DLPI
  требуются полномочия **root** или установка для **tcpdump** флага **SUID**.
- **IRIX** с snoop
  требуются полномочия **root** или установка для **tcpdump** флага **SUID**.
- **Linux**
  требуются полномочия **root** или установка для **tcpdump** флага **SUID**, если ваша система не использует ядро с поддержкой битов возможностей[3] (таких, как CAP_NET_RAW). В последнем случае для вам потребуется установка бита **CAP_NET_RAW** для захвата пакетов и бита **CAP_NET_ADMIN** для просмотра списка устройств помощью опции **-D**. Для просмотра текущего состояния битов возможностей служит функция **getcap**, а для управления этими битами - **setcap** из библиотеки **libcap**. Дополнительную информацию о поддерживаемых битах возможностей вы найдете, воспользовавшись командой **man capabilities**.
- **ULTRIX** и **Digital UNIX/Tru64 UNIX**
  всем пользователям разрешено использование программы tcpdump. Однако никому из пользователей не разрешено использовать режим захвата пакетов, пока администратор (super-user) не разрешит этот режим для данного интерфейса с помощью команды **pfconfig**. Захват принимаемых или передаваемых интерфейсом unicast-пакетов не будет возможен до тех пор, пока администратор (super-user) не включит для этого интерфейса режим **copy-all** с помощью команды **pfconfig**. Поскольку сбор пакетов обычно требует включения обоих упомянутых режимов, реальное использование tcpdump возможно только с позволения администратора.
- **BSD** и **Mac OS X**
  требуется доступ для чтения к устройству **/dev/bpf***. На системах BSD с поддержкой **devfs** (сюда относятся и системы Mac OS X) кроме установки принадлежности и прав доступа к устройствам BPF может потребоваться

1 *Список поддерживаемых библиотекой функций можно получить по команде **man pcap**.*
2 ***Promiscuous -** "Неразборчивый" режим, при котором драйвер устройства захватывает все передаваемые через среду пакеты. В нормальном режиме драйвер обычно читает из среды лишь пакеты, адресованные данному устройству.*
3 *Поддержка "битов возможностей" (capability bit) обеспечивается в ядре Linux начиная с версии 2.2.*

настройка конфигурации **devfs**, позволяющая задавать принадлежность и права доступа всякий раз при перезагрузке системы.

Чтение собранных пакетов из файла не требует специальных привилегий.

11.9.2.1 Опции tcpdump

Программа tcpdump позволяет в командной строке задать все опции сбора и отображения пакетов, а также спецификацию фильтров захвата, описанных ниже (параграф на стр.). Таблица содержит список опций tcpdump и описание каждой их них. Отметим, что некоторые опции поддерживаются не всеми платформами, на которых может использоваться программа tcpdump.

Таблица 64. Опции командной строки tcpdump.

Опция	Описание
-A	задает вывод каждого пакета (без заголовков канального уровня) в формате ASCII. Этот режим удобен для сбора трафика HTTP.
-c <число пакетов>	задает завершение работы программы после захвата заданного числа пакетов.
-C <размер файла>	задает необходимость проверки размера файла захвата перед записью в него каждого нового пакета. Если размер файла превышает значение параметра **file_size**, этот файл закрывается и создается новый файл для записи в него пакетов. Для файлов захвата используется имя, заданное параметром **-w** и, начиная со второго файла к имени добавляется в качестве суффикса номер файла. Переменная file_size задает размер файла в миллионах байтов (не в мегабайтах = 1 048 576 байт).
-d	задает вывод дампа скомпилированного кода соответствия пакетов (packet-matching code) в понятном человеку формате и завершение работы программы.
-dd	выводит дамп кода соответствия в виде фрагмента C-программы.
-ddd	выводит дамп кода соответствия в виде строки десятичных значений, перед которой следует строка со значением счетчика.
-D	выводит список сетевых интерфейсов системы, с которых tcpdump может собирать пакеты. Для каждого сетевого интерфейса указывается имя и номер, за которыми может следовать текстовое описание интерфейса. Имя и номер интерфейса могут использоваться с флагом **-i** для задания сбора пакетов с одного интерфейса. Эта опция может быть весьма полезна для систем, не дающих информации об имеющихся сетевых интерфейсах[1]. Флаг **-D** не поддерживается, если программа tcpdump была скомпилирована со старой версией libpcap, которая не поддерживает функцию pcap_findalldevs().
-e	выводит заголовок канального уровня в каждой строке дампа.
-E <algo:secret>	задает использование алгоритма и секрета **spi@ipaddr** для расшифровки пакетов IPsec ESP, направленных по адресу **ipaddr** и содержащих and в поле Security Parameter Index значение **spi**. Комбинация spi и адреса может быть повторена с использованием в качестве разделителя запятой или новой строки. Отметим, что установка секрета для пакетов IPv4 ESP в настоящее время поддерживается. В качестве алгоритмов могут использоваться **des-cbc**, **3des-cbc**, **blowfish-cbc**, **rc3-cbc**, **cast128-cbc** или **none**. По умолчанию применяется алгоритм **des-cbc**. Возможность дешифровки пакетов обеспечивается только в тех случаях, когда при компиляции **tcpdump** были включены опции поддержки криптографии. Параметр **secret** содержит ASCII-текст секретного ключа ESP. Если секрет начинается с символов 0x, будет считываться шестнадцатеричное значение. Опция предполагает использование ESP в соответствии с RFC 2406, а не RFC 1827. Эта опция поддерживается только для отладки и использовать ее с реальными секретными ключами не следует, поскольку введенный в командной строке ключ IPsec доступен другим пользователям системы[2]. Кроме явного указания параметров в командной строке их можно задать в файле опций, который tcpdump будет читать при получении первого пакета ESP.
-f	задает вывод чужих адресов IPv4 в числовом формате. Использование этой опции позволяет избавиться от проблем, возникающих на серверах Sun NIS при попытках трансляции нелокальных адресов. Проверка чужеродности адреса IPv4 осуществляется с использованием адреса и маски принявшего пакет интерфейса. Если адрес и маска интерфейса недоступны (например, при использовании unnumbered-интерфейсов или при захвате пакетов со всех адресов в Linux с использованием фиктивного интерфейса **any**), эта опция будет работать некорректно.
-F <файл>	задает использование фильтров, содержащихся в указанном файле. В этом случае заданные в командной строке фильтры игнорируются.

1 *Например, Windows или UNIX-системы, в которых не поддерживается команда **ifconfig -a***
2 *Например, его можно увидеть с помощью команды **ps**.*

Опция	Описание
-i <интерфейс>	задает сбор пакетов с указанного интерфейса. Если интерфейс не задан, tcpdump ищет в системе список доступных интерфейсов и выбирает в нем активное устройство с минимальным номером (исключая loopback). В системах Linux, начиная с ядра 2.2 поддерживается фиктивный интерфейс с именем **any**, обеспечивающий сбор пакетов со всех активных интерфейсов системы. Отметим, что сбор пакетов с устройства **any** осуществляется в обычном (не promiscuous) режиме. Если в системе поддерживается флаг -D, можно в качестве аргумента задавать номер интерфейса, выводимый при использовании этого флага.

-l	задает буферизацию строк **stdout**. Эта опция полезна в тех случаях, когда вы хотите просматривать данные во время сбора пакетов. Например, команды
	`tcpdump -l | tee dat`
	или
	`tcpdump -l > dat & tail -f dat`
	обеспечивают запись пакетов в файл **dat** и одновременный вывод на консоль.

-L	задает вывод списка известных типов канального уровня и завершение работы программы.
-m <файл>	загружает модуль определений SMI MIB из указанного файла. Эта опция может использоваться неоднократно для загрузки нескольких модулей MIB.
-n	отключает преобразование адресов и номеров портов в символьные имена.
-N	задает использование только имен хостов, а не полных доменных имен. Например, вместо **lhotze.bilim-systems.net** при использовании этой опции моя рабочая станция будет обозначаться как **lhotze**.
-O	отключает оптимизатор кода проверки соответствия пакетов условиям фильтрации (см. параграф 11.9.2.2 на стр. 265). Используйте эту опцию, если вам покажется, что оптимизатор работает с ошибками.
-p	указывает программе, что интерфейс не нужно переводить в режим захвата[1]. Опцию **-p** нельзя использовать вместе с фильтром **ether host {local-hw-addr} or ether broadcast**.
-q	задает вывод минимального объема информации.
-R	при установке этого флага предполагается, что пакеты ESP/AH используют старый вариант спецификации[2] и tcpdump не будет выводить поля replay prevention (защита от воспроизведения). Поскольку спецификация ESP/AH не включает поля с номером версии, tcpdump не может определить версию протокола ESP/AH по заголовкам пакетов.
-r <файл>	задает чтение данных из файла, созданного ранее с использованием команды **tcpdump -w** или с помощью другой программы, поддерживающей формат tcpdump (например, Ethereal). Если в качестве имени файла задан символ **-**, используется поток данных от стандартного устройства ввода (**stdin**).
-S	задает вывод абсолютных порядковых номеров TCP взамен относительных.
-s <snaplen>	задает захват из каждого пакета **snaplen** байтов вместо отбираемых по умолчанию 68 байтов[3]. Значение 68 подходит для протоколов IP, ICMP, TCP и UDP но может приводить к потере протокольной информации для некоторых пакетов DNS (см. стр. 274) и NFS (см. стр. 274). Потеря части пакетов по причине малого размера кадра захвата (snapshot) указывается в выходных данных полями вида **[|proto]**', где **proto** - имя протокольного уровня, на котором произошло отсечение части пакета[4]. Отметим, что увеличение кадра захвата приведет к дополнительным временным затратам на обработку пакетов и уменьшению числа буферизуемых пакетов, что может привести к потере части пакетов. Используйте минимальное значение **snaplen**, которое позволит обойтись без потери информации об интересующем вас протоколе. Установка **snaplen = 0** приведет к захвату полных пакетов.
-T <тип>	задает интерпретацию пакетов, выбранных с помощью фильтра (см. параграф на стр.), как пакетов указанного параметром типа. В настоящее время поддерживаются типы **aodv**[5], **cnfp**[6], **rpc**[7], **rtp**[8], **rtcp**[9], **snmp**[10], **tftp**[11], **vat**[12] и **wb**[13].

1 *Интерфейс может быть переведен в режим захвата другими программами, поэтому использование флага **-p** отнюдь не гарантирует работу интерфейса в обычном режиме - программа просто не будет переводить этот интерфейс в режим захвата. Кроме того, даже в обычном режиме захватываться будут не только пакеты, адресованные этому интерфейсу, поскольку в сети всегда присутствуют широковещательные пакеты и могут использоваться пакеты с групповыми адресами (multicast).*
2 *RFC1825 - RFC1829*
3 *В SunOS NIT минимум составляет 96 байтов.*
4 *Например, при захвате пакетов в сети Ethernet с помощью команда **tcpdump -s 12** на выходе будут появляться строки типа **22:31:43.385357 [\|ether]**, показывающие, что усекновение пакетов произошло на уровне Ethernet*
5 *Протокол Ad-hoc On-demand Distance Vector.*
6 *Протокол Cisco NetFlow.*
7 *Протокол Remote Procedure Call (удаленный вызов процедур).*
8 *Протокол Real-Time Applications (приложения в реальном масштабе времени).*
9 *Протокол управления приложениями реального времени (Real-Time Applications control protocol).*

Опция	Описание
-t	отключает вывод временных меток в каждой строке дампа.
-tt	задает вывод в каждой строке дампа неформатированных временных меток.
-ttt	задает вывод временных интервалов (в микросекундах) между захватом предыдущего и данного пакетов в каждой строке дампа.
-tttt	задает вывод временных меток в принятом по умолчанию формате для каждой строки дампа.
-u	задает вывод манипуляторов (handle) NFS без декодирования.
-U	задает режим “буферизации на уровне пакетов” для файлов, сохраняемых с помощью опции **-w**. В этом режиме каждый пакет записывается в выходной файл как только он будет захвачен (не дожидаясь заполнения выходного буфера). Флаг **-U** не будет поддерживаться, если программа **tcpdump** была скомпилирована со старой опцией **libpcap**, не поддерживающей функцию **pcap_dump_flush()**.
-v	задает вывод дополнительной информации при захвате файлов. К такой информации может относиться значение TTL (время жизни), идентификация, общий размер, опции IP и т. п. При использовании этого флага также выполняется дополнительная проверка целостности пакетов с помощью контрольных сумм (например, для протоколов IP и ICMP).
-vv	задает дополнительное увеличение объема выводимой информации (например, полное декодирование пакетов SMB, вывод дополнительных полей откликов NFS и т. п.).
-vvv	задает максимальный объем выводимой информации (например, полностью выводятся опции **telnet SB ... SE**). При использовании вместе с ключом **-X** опции Telnet выводятся также в шестнадцатеричном представлении.
-w <файл>	задает запись необработанных (raw) пакетов. Собранные в файл пакеты можно впоследствии просматривать с использованием флага **-r** или передавать для анализа другим программам (например, Ethereal). Если в качестве имени файла указан символ **-**, запись осуществляется на стандартное устройство вывода (**stdout**).
-x	задает вывод шестнадцатеричного дампа (без заголовка канального уровня) для каждого захваченного пакета. Объем выводимой информации определяется меньшим из двух значений - размер пакета и значение параметра **snaplen** (см. стр. 264). Отметим, что при захвате полных кадров канального уровня дамп может включать также байты заполнения, если пакет сетевого уровня имеет малый размер.
-xx	задает вывод шестнадцатеричного дампа для каждого пакета с включением заголовков канального уровня.
-X	задает вывод дампа в шестнадцатеричном и ASCII-формате без заголовков канального уровня. Эта опция может быть очень удобна при анализе новых протоколов.
-XX	задает вывод дампа в шестнадцатеричном и ASCII-формате с включением заголовков канального уровня.
-y <тип>	задает тип канального уровня, используемого при захвате пакетов. Поддерживаемые значения можно посмотреть с помощью флага **-L**.

11.9.2.2 Фильтрация при сборе пакетов

В командной строке tcpdump наряду с опциями могут задаваться выражения, определяющие фильтрацию пакетов на этапе их сбора. Если никакого фильтра не задано, программа будет собирать все пакеты.

Каждое выражение, задающее фильтр, включает один или несколько примитивов, состоящих обычно из одного или нескольких идентификаторов объекта и предшествующих ему классификаторов. Идентификатором объекта может служить его имя или номер. Классификаторы объектов могут относиться к одному из трех видов:

type

указывает тип объекта, заданного идентификатором. В качестве типа объектов могут указываться значения **host** (хост), **net** (сеть) и **port** (порт). Если тип объекта не указан, предполагается значение **host**.

dir

задает направление по отношению к объекту. Для этого классификатора поддерживаются значения **src** (объект является отправителем), **dst** (объект является получателем), **src or dst** (отправитель или получатель) и **src and dst** (отправитель и получатель). Например, **src foo** указывает на пакеты, отправленные с хоста **foo**, **dst net 128.3** - пакеты, адресованные в сеть **128.3.0.0/16**, а **src or dst port ftp-data** - пакеты данных протокола FTP (порт **ftp-data**), передаваемые в обоих направлениях. Если классификатор **dir** не задан, предполагается значение **src or dst**. Для некоторых типов соединений (например, SLIP) и режимов захвата (например, захват с фиктивного интерфейса **any** в Linux-системах) могут использоваться классификаторы **inbound** и **outbound**.

proto

10 *Простой протокол сетевого управления (Simple Network Management Protocol).*
11 *Тривиальный протокол обмена файлами (Trivial File Transfer Protocol).*
12 *Visual Audio Tool.*
13 *Распределенные доски White Board.*

задает протокол, к которому должны относиться пакеты. Этот классификатор может принимать значения **ether**, **fddi**[1], **tr**[2], **wlan**[3], **ip**, **ip6**, **arp**, **rarp**, **decnet**, **tcp** и **udp**. Если примитив не содержит классификатора протокола, предполагается, что данному фильтру удовлетворяют все протоколы, совместимые с типом объекта[4].

Кроме объектов и квалификаторов примитивы могут содержать ключевые слова **gateway** (шлюз), **broadcast** (широковещательный), **less** (меньше), **greater** (больше) и арифметические выражения.

Сложные фильтры могут содержать множество примитивов, связанных между собой с использованием логических операторов **and**, **or** и **not** (например, **host foo and not port ftp and not port ftp-data**). Для сокращения задающих фильтры выражений можно опускать идентичные списки квалификаторов. Например, выражение **tcp dst port ftp or ftp-data or domain** будет краткой формой выражения

```
tcp dst port ftp or tcp dst port ftp-data or tcp dst port domain
```

11.9.2.2.1 Допустимые примитивы фильтрации пакетов

Ниже приводится список допустимых примитивов с краткими комментариями для каждого из них.

Таблица 65 Примитивы фильтров tcpdump

Примитив	Описание
dst host <хост>	будет отбирать пакеты, в которых поле адреса получателя IPv4/v6 содержит адрес хоста, заданного в примитиве.
src host <хост>	будет выбирать все пакеты, в которых поле отправителя содержит адрес указанного хоста.
host <хост>	будет отбирать все пакеты, для которых адрес хоста указан в поле получателя или отправителя. Все три приведенных выше выражения могут содержать идентификаторы протоколов **ip**, **arp**, **rarp** или **ip6**, как в выражении **ip host <хост>** эквивалентном фильтру: **ether proto \ip and host <хост>** Если именем задан хост, с которым связано несколько адресов IP, фильтру будут соответствовать пакеты с любым из этих адресов в заголовках пакетов.
ether dst <ehost>	будет выбирать все кадры, в которых поле MAC-адреса получателя содержит значение ehost (имя хоста из файла /etc/ethers или шестнадцатеричное представление MAC-адреса[5]).
ether src <ehost>	будет выбирать все кадры, в которых поле MAC-адреса отправителя содержит значение ehost.
ether host <ehost>	будет отбирать все пакеты с адресом, указанным значением **ehost** в поле отправителя или получателя.
gateway <шлюз>	будет отбирать все пакеты, использующие указанный именем хост в качестве шлюза[6]. Указанное параметром имя хоста должно преобразовываться в IP-адрес механизмами преобразования имен, доступными локальному компьютеру (файл /etc/hosts, DNS, NIS и т. п.), а также механизмами определения MAC-адреса по имени хоста (/etc/ethers и т. п.). Эквивалентное выражение **ether host ehost and not host <хост>** позволяет указывать хост по имени или адресу, указанному в файле host/ehost. Отметим, что данный примитив пока не поддерживается для конфигураций IPv6.
dst net <сеть>	отбирает все пакеты IPv4/v6, направленные в указанную сеть. Для указания сети можно использовать имя из файла **/etc/networks** или номер сети.
src net <сеть>	выбирает все пакеты Ipv4/v6, отправленные из указанной сети.
net <сеть>	выбирает все пакеты IPv4/v6, содержащие адреса из указанной сети в поле отправителя или получателя.

1 ***fddi** в действительности является псевдонимом **ether**; при анализе примитива оба классификатора трактуются как “канальный уровень, используемый указанным интерфейсом”. Заголовки FDDI содержат адреса отправителя и получателя, подобные адресам Ethernet, поля типа также зачастую содежат значения, подобные используемым для Ethernet, поэтому можно фильтровать эти поля в кадрах FDDI как и аналогичные поля кадров Ethernet. Заголовки FDDI содержат и другие поля, но их нельзя указать в фильтрах.*

2 ***tr** также является в действительности псевдонимом **ether**, поскольку оба типа кадров используют весьма похожую структуру заголовков.*

3 *Идентификатор протокола **wlan** (беспроводные сети 802.11) является псевдонимом **ether**. В заголовках 802.11 адрес получателя содержится в поле DA, адрес отправителя - в поле SA, а поля BSSID, RA и TA в заголовках 802.11 не проверяются фильтрами.*

4 *Например, примитиву **src foo** будут соответствовать все пакеты **ip**, **arp** и **rarp**, исходящие от хоста foo.*

5 *MAC-адрес записывается в формате **xx:xx:xx:xx:xx:xx***

6 *Т. е., адрес отправителя или получателя на канальном уровне (например, ethernet) соответствует адресу хоста, заданному значением <шлюз>, но IP-адреса отправителя и получателя в заголовке пакета не совпадают с IP-адресом указанного шлюза.*

Примитив	Описание
net <сеть> mask <маска сети>	будет отбирать все пакеты IP[1], содержащие в поле отправителя или получателя адреса из сети, указанной с использованием маски.
net <сеть/размер маски>	будет отбирать все пакеты IPv4/v6, содержащие в поле отправителя или получателя адреса из сети, указанной с использованием маски.
dst port <порт>	отберет все пакеты **ip/tcp**, **ip/udp**, **ip6/tcp** и **ip6/udp**, направленные в указанный порт. Номера портов могут задаваться номерами или именами из файла **/etc/services**. При указании имени (протокол/порт) проверяется как порт, так и протокол. Если примитив содержит номер или неоднозначное обозначение порта (только порт, без протокола) фильтру будут соответствовать пакеты обоих протоколов (tcp и udp). Например, фильтру **dst port 513** будут соответствовать пакеты **tcp/login** и **udp/who**, а фильтру **port domain** - трафик **tcp/domain** и udp/domain.
src port <порт>	отбирает все пакеты, отправленные из указанного порта.
port <порт>	отбирает все пакеты, содержащие указанный номер порта в поле отправителя или получателя. Любое из трех перечисленных правил для портов может включать в качестве префикса идентификатор протокола **tcp** или **udp** (например, **tcp src port <порт>**, будет отбирать пакеты tcp, отправленные из указанного порта).
less <размер>	будет собирать пакеты, размер которых не превышает указанного значения.
greater <размер>	будет собирать пакеты, размер которых не меньше указанного значения.
ip proto <протокол>	отбирает все пакеты IP, содержащие заданный идентификатор типа в поле типа протокола. Типы протоколов IP можно указывать по именам или (**icmp**, **icmp6**, **igmp**, **igrp**, **pim**, **ah**, **esp**, **vrrp**, **udp**, **tcp**) или номерам. Поскольку **tcp**, **udp** и **icmp** используются также в качестве ключевых слов, перед этими идентификаторами следует помешать символ **** (слэш)[2]. Отметим, что этот примитив не проверяет цепочки протокольных заголовков.
ip6 proto <протокол>	будет отбирать все пакеты IPv6 указанного типа. Отметим, что этот примитив не проверяет цепочки протокольных заголовков.
ip6 protochain <протокол>	отберет все пакеты IPv6, содержащие в цепочке протокольных заголовков идентификатор указанного типа протокола. Например, фильтру **ip6 protochain 6** будут соответствовать все пакеты IPv6 с заголовками TCP в цепочке заголовков. Такой пакет может содержать, например, заголовок аутентификации (AH), маршрутный заголовок (routing header), или заголовок опции hop-by-hop между заголовками IPv6 и TCP. Отметим, что порождаемый этим примитивом код BPF достаточно сложен и не может быть оптимизирован средствами tcpdump, поэтому использование данного фильтра может замедлять работу программы.
ip protochain <протокол>	эквивалентно примитиву **ip6 protochain protocol**, но работает с пакетами IPv4.
ether broadcast	обеспечивает отбор всех широковещательных кадров Ethernet. Ключевое слово **ether** может быть опущено.
ip broadcast	отбирает все широковещательные пакеты IPv4. Этому правилу будут соответствовать широковещательные адреса, содержащие только нули (all-zeroes) и только единицы (all-ones) с учетом маски подсети для интерфейса, который используется для захвата пакетов. Если маска подсети для интерфейса недоступна[3], фильтр может работать некорректно.
ether multicast	собирает все кадры с групповыми адресами Ethernet. Ключевое слово **ether** использовать необязательно. Логически это правило эквивалентно выражению **ether[0] & 1 != 0**.
ip multicast	отбирает пакеты с групповыми адресами IP.
ip6 multicast	отбирает пакеты с групповыми адресами IPv6.

1 *Этот примитив не может использоваться для сетей IPv6.*
2 *При работе с командным интерпретатором C-shell следует добавлять еще один слэш (****)*
3 *Интерфейс не имеет маски или сбор пакетов осуществляется со всех интерфесов хоста Linux (фиктивный интерфейс **any**).*

Примитив	Описание
ether proto <протокол>	отбирает кадры Ethernet с заданным типом протокола. Протокол может быть указан по номеру или имени[1] (**ip**, **ip6**, **arp**, **rarp**, **atalk**, **aarp**, **decnet**, **sca**, **lat**, **mopdl**, **moprc**, **iso**, **stp**, **ipx**, **netbeui**). В случаях использования правила для протоколов FDDI (например, **fddi protocol arp**), Token Ring (например, tr protocol arp) и IEEE 802.11 (например, **wlan protocol arp**) в большинстве случаев идентификация протокола производится на основании заголовка 802.2 Logical Link Control (LLC), который обычно помещается после заголовка FDDI, Token Ring или 802.11. При фильтрации для большинства протоколов FDDI, Token Ring и 802.11 программа **tcpdump** проверяет только поле идентификатора протокола (protocol ID) в заголовке LLC так называемого SNAP-формата с идентификатором OUI = 0x000000 (Organizational Unit Identifier), указывающим на инкапсуляцию Ethernet. Проверка для пакетов использования формата SNAP с OUI = 0x000000 не выполняется за исключением перечисленных ниже случаев: **iso** tcpdump проверяет поля **DSAP**[2] и **SSAP**[3] в заголовках LLC; **stp** **netbeui** tcpdump проверяет поле DSAP в заголовке LLC; **atalk** tcpdump проверяет использование в кадре формата SNAP с OUI = 0x080007 и тип (etype) AppleTalk. Для случая Ethernet проверяются поля типа Ethernet для большинства протоколов. Исключениями являются протоколы **iso** **sap** **netbeui** tcpdump проверяет для кадра принадлежность к 802.3 и заголовок LLC (как это описано выше для FDDI, Token Ring и 802.11); **atalk** проверяется типа **AppleTalk** в кадре Ethernet и формат заголовка SNAP (как для FDDI, Token Ring и 802.11); **aarp** проверяется тип **AppleTalk ARP** в кадре Ethernet или использование формата 802.2 SNAP с OUI = 0x000000; **ipx** tcpdump проверяет тип IPX в кадре Ethernet, поле IPX DSAP в заголовке LLC, инкапсуляцию IPX и тип IPX в кадре SNAP.
decnet src <хост>	собирает все пакеты от указанного хоста DECNET, который может быть задан по адресу или имени DECNET[4].
decnet dst <хост>	отбирает все пакеты, адресованные указанному хосту DECNET.
decnet host <хост>	собирает все пакеты, содержащие адрес указанного хоста DECNET в поле отправителя или получателя.
ifname <интерфейс>	отбирает все пакеты, полученные от указанного интерфейса[5].
on <интерфейс>	синоним **ifname**.
rnr <номер>	собирает только пакеты, записанные в файл программой pf в соответствии с правилом, имеющим указанных номер. Это правило работает только при просмотре файлов, собранных с помощью **pf**.
rulenum <номер>	синоним для **rnr**.
reason <код>	собирает только пакеты, соответствующие указанному коду причины (PF reason code). Известные коды причин включают **match**, **bad-offset**, **fragment**, **short**, **normalize**, **memory**. Правило работает только при просмотре файлов, собранных с помощью **pf**.
action <действие>	отбирает пакеты, записанные в файл указанной операцией PF (**pass** или **block**). Правило работает только при просмотре файлов, собранных с помощью **pf**.

1 *Перечисленные здесь имена протоколов могут использоваться также в качестве ключевых слов, поэтому имени протокола должен предшествовать символ* **** *(например,* ***\arp****).*
2 *Destination Service Access Point - точка доступа к сервису для получателя.*
3 *Source Service Access Point - точка доступа к сервису для отправителя.*
4 *Поддержка имен DECNET обеспечивается только на хостах ULTRIX, настроенных для использования DECNET.*
5 *Это правило применимо только к пакетам, собранным в файл с помощью программы* ***pf*** *(OpenBSD).*

Примитив	Описание
ip, ip6, arp, rarp, atalk, aarp, decnet, iso, stp, ipx, netbeui	используются в качестве сокращения для: **ether proto p** где p - один из перечисленных протоколов.
lat, moprc, mopdl	сокращения для: **ether proto p** где p - один из перечисленных протоколов. Отметим, что в настоящее время tcpdump не умеет разбирать эти протоколы.
vlan [vlan_id]	отбирает кадры IEEE 802.1Q VLAN. Если указан необязательный идентификатор VLAN, выделяются лишь пакеты, относящиеся в указанной виртуальной ЛВС. Отметим, что первое ключевой слово **vlan** изменяет расчет смещения полей для оставшейся части выражения с учетом размеров полей VLAN в заголовке кадра.
tcp, udp, icmp	используется в качестве сокращения для **ip proto p** или **ip6 proto p** где p - один из перечисленных протоколов.
iso proto <протокол>	собирает пакеты с указанным типом протокола OSI. Протокол может быть указан по номеру или имени (**clnp**, **esis**, **isis**).
clnp, esis, isis	сокращения для выражений: **iso proto p** где p - один из перечисленных протоколов.
l1, l2, iih, lsp, snp, csnp, psnp	сокращения для типов IS-IS PDU.
vpi n	собирает пакеты ATM с указанным идентификатором виртуального пути для SunATM (Solaris).
vci n	собирает пакеты ATM с указанным идентификатором виртуального канала для SunATM (Solaris).
lane	собирает пакеты эмуляции ЛВС (ATM LANE) для SunATM (Solaris). Первое ключевое слово **lane** в выражении изменяет проверки для остальной части фильтра в предположении что пакет относится к пакетам эмуляции Ethernet или LANE LE Control. Если ключевое слово **lane** не указано, проверки выполняются в предположении LLC-инкапсуляции.
llc	собирает пакеты ATM с инкапсуляцией LLC для SunATM (Solaris).
oamf4s	собирает пакеты ATM для SunATM (Solaris), являющиеся сегментами потока ячеек OAM F4 (VPI=0, VCI=3).
oamf4e	собирает пакеты ATM для SunATM (Solaris), относящиеся к сквозным потокам OAM F4 (VPI=0, VCI=4).
oamf4	собирает пакеты ATM для SunATM (Solaris), являющиеся сегментами сквозного потока ячеек OAM F4 (VPI=0, (VCI=3 или VCI=4)).
oam	собирает пакеты ATM для SunATM (Solaris), являющиеся сегментами сквозного потока ячеек OAM F4 (VPI=0, (VCI=3 или VCI=4)).
metac	собирает пакеты ATM для SunATM (Solaris), относящиеся к сигнальным мета-устройствам (VPI=0, VCI=1).
bcc	собирает пакеты ATM для SunATM (Solaris), относящиеся к широковещательным сигнальным устройствам (VPI=0, VCI=2).
sc	собирает пакеты ATM для SunATM (Solaris), относящиеся к сигнальным устройствам (VPI=0, VCI=5).
ilmic	собирает пакеты ATM для SunATM (Solaris), относящиеся к клиентским устройствам ILMI (VPI=0, VCI=16).
connectmsg	собирает пакеты ATM для SunATM (Solaris), относящиеся к сигнальным устройствам и содержащие сообщения **Q.2931 Setup**, **Call Proceeding**, **Connect**, **Connect Ack**, **Release**, **Release Done**.
metaconnect	собирает пакеты ATM для SunATM (Solaris), относящиеся к сигнальным мета-устройствам и содержащие сообщения **Q.2931 Setup**, **Call Proceeding**, **Connect**, **Connect Ack**, **Release**, **Release Done**.

11.9.2.2.2 Логические выражения

Выражения типа

```
expr <операция> expr
```
возвращают логическое значение, соответствующее отношениям между левой и правой часть. В качестве операции могут использоваться **>**, **<**, **>=**, **<=**, **=**, **!=**, а операнды **expr** могут быть арифметическими выражениями, включающими целые константы (запись в стандарте С), бинарные операторы **+**, **-**, ***, */*, **&**, **|**, **<<**, **>>**, оператор длины (offset) и данные из пакетов. Для получения значений полей из пакетов применяется синтаксис:

```
proto [offset : size]
```
Параметр **proto** может содержать идентификатор одного из протоколов (**ether**, **fddi**, **tr**, **wlan**, **ppp**, **slip**, **link**, **ip**, **arp**, **rarp**, **tcp**, **udp**, **icmp**, **ip6**) и задает уровень протокола[1], для которого извлекаются данные. Отметим, что **tcp**, **udp** и другие протоколы верхних уровней относятся только к пакетам IPv4, а не IPv6[2]. Параметр **offset** задает смещение в байтах относительно начала заголовка указанного уровня. Необязательный параметр **size** определяет размер интересующего поля в байтах. Допустимы значения размера 1, 2 и 4, по умолчанию просматривается 1 байт.

Оператор длины, указываемый ключевым словом **len**, определяет размер пакета в байтах.

Например, выражению

```
ether[0] & 1 != 0
```
будет соответствовать весь multicast-трафик, выражение

```
ip[0] & 0xf != 5
```
позволяет собрать все пакеты IP, в которых присутствует поле опций, а фильтр

```
ip[6:2]  & 0x1fff  = 0
```
соберет только нефрагментированные дейтаграммы и первые фрагменты.

При выборе полей из заголовков учитывается структура пакетов соответствующего уровня. Например, tcp[0] всегда будет возвращать первый байт заголовка TCP, игнорируя фрагменты.

Некоторые поля и значения смещений могут задаваться не только числами, но и именами. В частности, для протокола поддерживается параметр **icmptype** (поле типа ICMP), который может принимать значения **icmp-echoreply**, **icmp-unreach**, **icmp-sourcequench**, **icmp-redirect**, **icmp-echo**, **icmp-routeradvert**, **icmp-routersolicit**, **icmp-timxceed**, **icmp-paramprob**, **icmp-tstamp**, **icmp-tstam-preply**, **icmp-ireq**, **icmp-ireqreply**, **icmp-maskreq**, **icmp-maskreply**. Для флагов TCP можно использовать идентификаторы **tcp-fin**, **tcp-syn**, **tcp-rst**, **tcp-push**, **tcp-ack** и **tcp-urg**.

Примитивы в выражениях можно группировать с использованием

- скобок[3];
- отрицания (**!** или **not**);
- логического пересечения (**&&** или **and**);
- логического объединения (**||** или **or**).

Оператор отрицания имеет высший уровень приоритета, операции объединения и пересечения имеют одинаковый приоритет и выполняются слева направо в порядке следования. Отметим, что для операции логического пересечения недостаточно просто указать операнды рядом, а требуется явно задать операцию (**&&** или **and**).

Если идентификатор указан без ключевого слова, предполагается ключевое слово, которое до этого использовалось последним. Например, выражение

```
not host vs and ace
```
является простым сокращением от

```
not host vs and host ace
```
Отметим, что эти выражения не эквивалентны фильтру **not (host vs or ace)**.

Аргументы выражений могут передаваться программе tcpdump как один или множество аргументов (используйте более удобную для вас форму). В общем случае выражения, содержащие мета-символы командного интерпретатора, должны передаваться как один аргумент, заключенный в кавычки.

11.9.2.2.3 Примеры фильтров

Таблица 66. Примеры фильтров tcpdump.

Фильтр	Выполняемые действия
`tcpdump host sundown`	Выводит все пакеты, принимаемые и передаваемые хостом **sundown**
`tcpdump host helios and \( hot or ace \)`	Выводит пакеты, передаваемые между хостом **helios** и любым из хостов **hot** или **ace.**
`tcpdump ip host ace and not helios`	Выводит пакеты передаваемые между хостом **ace** и любым хостом, за исключением **helios**.
`tcpdump net ucb-ether`	Выводит все пакеты, передаваемые или принимаемые хостами сети **ucb-ether**.

1 *Протоколы **ether**, **fddi**, **wlan**, **tr**, **ppp**, **slip** и **link** указывают на канальный уровень.*
2 *Поддержка IPv6 будет реализована в следующих версиях.*
3 *В зависимости от используемого командного интерпретатора перед символами открывающих и закрывающих скобок может потребоваться включение esc-символа.*

Фильтр	Выполняемые действия
`tcpdump 'gateway snup and (port ftp or ftp-data)'`	Выводит весь трафик **ftp**, проходящий через шлюз **snup**[1].
`tcpdump ip and not net localnet`	Выводит весь трафик, за исключением исходящего от локальных хостов и адресованного им.
`tcpdump 'tcp[tcpflags] & (tcp-syn|tcp-fin) != 0 and not src and dst net localnet'`	Выводит стартовые (SYN) и конечные (FIN) пакеты TCP, исключая соединения локальных хостов.
`tcpdump 'gateway snup and ip[2:2] > 576'`	Выводит переданные через шлюз **snup** пакеты IP, размер которых превышает 576 байтов.
`tcpdump 'ether[0] & 1 = 0 and ip[16] >= 224'`	Выводит широковещательные и групповые пакеты, которые не были переданы с использованием широковещательных и групповых адресов Ethernet.
`tcpdump 'icmp[icmptype] != icmp-echo and icmp[icmptype] != icmp-echoreply'`	Выводит все пакеты ICMP, кроме запросов и откликов **echo** (т. е., кроме пакетов **ping**).

11.9.2.3 Формат вывода

Формат вывода программы **tcpdump** зависит от протокола. Ниже приведены краткие описания и примеры для большинства используемых форматов вывода.

11.9.2.3.1 Заголовки канального уровня

При использовании опции **-e** выводится содержимое заголовков канального уровня. Для сетей Ethernet информация из заголовков включает адреса отправителя и получателя, протокол и размер кадра.

Для сетей FDDI опция **-e** обеспечивает вывод поля управления (frame control), адресов отправителя и получателя, а также размера кадра. Значение поля управления определяет интерпретацию остальной части кадра. Нормальные кадры (например, содержащие дейтаграммы IP) являются “асинхронными” с уровнем приоритета от 0 до 7 (например async4). Предполагается, что такие кадры содержат пакеты 802.2 LLC. Заголовок LLC выводится в тех случаях, когда пакет не является дейтаграммой ISO или так называемым SNAP-пакетом.

В сетях Token Ring опция **-e** обеспечивает вывод полей контроля доступа (**access control**) и управления кадром (**frame control**), адресов отправителя и получателя, а также размера кадра. Как и для сетей FDDI предполагается, что кадры содержат пакеты LLC. Независимо от наличия опции **-e** выводится информация о заданном отправителем маршруте (source routing), если пакет содержит такую информацию.

При использовании в сетях 802.11 опция **-e** выводит значения полей управления (**frame control**), все адреса из заголовка 802.11, а также размер кадра. Предполагается, что кадры содержат пакеты LLC.

Для каналов SLIP информация канального уровня включает индикатор направления (**I** для входящего трафика, **O** - для исходящего), тип пакета и сведения о компрессии[2]. Поле типа пакета выводится первым и может принимать значение **ip**, **utcp** или **ctcp**. Остальные сведения относятся к пакету IP. Для пакетов TCP вслед за типом выводится идентификатор соединения. Если для пакета используется компрессия, сжатый заголовок декодируется перед выводом. Для специальных случаев[3] выводятся значения ***S+n** и ***SA+n** (где n - числовое значение), которые указывают величину изменения порядкового номера для пакета и подтверждения, соответственно. Для остальных пакетов могут выводиться индикаторы изменений **U** (указатель важности - urgent pointer), **W** (окно - window), **A** (подтверждение - ack), **S** (порядковый номер - sequence number) и I (идентификатор пакета - packet ID), сопровождаемые величиной изменения (+n или -n) или указателем на новое значение параметра (=n). Далее выводится информация о количестве данных в пакете и размер сжатого заголовка.

Например строка вывода

```
O ctcp * A+6 S+49 I+6 3 (6)
```

относится к исходящему сжатому пакету TCP с неявным идентификатором соединения. Порядковый номер подтверждения увеличился на 6, порядковый номер пакета - на 49, идентификатор пакета - на 6. Пакет содержит 3 байта данных и 6-байтовый сжатый заголовок.

11.9.2.3.2 Пакеты ARP/RARP

Информация, выводимая для пакетов arp/rarp, включает тип запроса и его аргументы. Выводимой информации вполне достаточно для понимания происходящих процессов. Ниже показан пример вывода для случая, когда хост **rtsg** открывает сессию **rlogin** с хостом **csam**:

```
arp who-has csam tell rtsg
arp reply csam is-at CSAM
```

1 *Кавычки позволяют избавиться от ошибок при анализе скобок командным интерпретатором.*
2 *Компрессия заголовков TCP/IP для каналов SLIP описана в документе RFC 1144, который вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc1144.txt или найти в каталоге Documents/ приложенного к книке компакт-диска.*
3 *RFC 1144 определяет как специальные случаи интерактивный трафик и передачу больших объемов трафика.*

Первая строка показывает запрос **arp** от хоста **rtsg** для получения адресов (MAC и IP) хоста **csam**. В ответ на это **csam** возвращает свои адреса (в нашем примере IP-адрес обозначен как **csam**, а MAC-адрес - как **CSAM**). Если ввести команду с опцией **-n**, результат будет выглядеть как

```
arp who-has 128.3.254.6 tell 128.3.254.68
arp reply 128.3.254.6 is-at 02:07:01:00:01:c4
```

Если же воспользоваться опцией **-e**, можно увидеть, что первый пакет является широковещательным (MAC-адрес отправителя показан как **RSTG**), поле типа содержит значение 0806 (**ETHER_ARP**), а размер пакета составляет 64 байта:

```
RTSG Broadcast 0806  64: arp who-has csam tell rtsg
CSAM RTSG 0806  64: arp reply csam is-at CSAM
```

11.9.2.3.3 Пакеты TCP

Для понимания приведенной здесь информации вы должны быть знакомы с протоколом TCP. Если протокол известен вам недостаточно хорошо, обратитесь к документу RFC 793.

Формат вывода для протокола TCP в общем случае имеет вид:

```
src > dst: flags data-seqno ack window urg options
```

Поля **src** и **dst** содержат IP-адреса и номера портов для отправителя и получателя. Поле **flags** содержит комбинацию символов **S** (SYN), **F** (FIN), **P** (PUSH), **R** (RST), **W** (ECN CWR) и **E** (ECN-Echo) в соответствии с установленными для пакета флагами или один символ "**.**" (нет флагов). Поле **data-seqno** описывает занятую данным пакетом часть пространства порядковых номеров. Поле **ack** содержит порядковый номер, ожидаемый для следующей порции данных, передаваемой через это соединение в обратном направлении. Поле **window** показывает число байтов в приемном буфере, доступных для обратного направления в этом соединении. Поле **urg** показывает состояние флага важности (**urgent**) для данных из этого пакета. Поле options содержит опции TCP, заключенные в угловые скобки.

Поля **src**, **dst** и **flags** присутствуют во всех случаях, а вывод остальных полей зависит от информации в заголовке пакета TCP.

Ниже показан набор пакетов, передаваемых при организации хостом **rtsg** сессии **rlogin** с хостом **csam**.

```
1) rtsg.1023 > csam.login: S 768512:768512(0) win 4096 <mss 1024>
2) csam.login > rtsg.1023: S 947648:947648(0) ack 768513 win 4096 <mss 1024>
3) rtsg.1023 > csam.login: . ack 1 win 4096
4) rtsg.1023 > csam.login: P 1:2(1) ack 1 win 4096
5) csam.login > rtsg.1023: . ack 2 win 4096
6) rtsg.1023 > csam.login: P 2:21(19) ack 1 win 4096
7) csam.login > rtsg.1023: P 1:2(1) ack 21 win 4077
8) csam.login > rtsg.1023: P 2:3(1) ack 21 win 4077 urg 1
9) csam.login > rtsg.1023: P 3:4(1) ack 21 win 4077 urg 1
```

Первая строка показывает, что порт TCP с номером 1023 хоста **rtsg** отправил пакет в порт **login** хоста **csam**. Символ **S** говорит о наличии в пакете флага **SYN**. Порядковый номер пакета равен 768512 и данных в пакете не содержится[1]. Пакет не содержит в себе подтверждения, доступный размер приемного окна составляет 4096 байтов, а запрошенное значение MSS составляет 1024 байта.

Сайт **csam** отправляет в ответ подобный полученному пакет, включающий подтверждение для полученного от **rtsg** пакета **SYN**. После этого **rtsg** подтверждает хосту **csam** получение от него пакета **SYN**. Точка (**.**) в поле флагов говорит о том, что пакет не содержит ни одного флага. Данных в пакете не содержится, поэтому в строке вывода отсутствуют значения порядковых номеров. Отметим, что для подтверждения в строке 3 указан порядковый номер 1. Когда программа **tcpdump** видит первый пакет в данном соединении, она выводит порядковый номер из этого пакета. Для последующих пакетов данного соединения выводятся значения разницы между порядковым номером для текущего пакета и начальным порядковым номером. Это значит, что для всех пакетов, кроме первого, порядковые номера указываются относительно начала потока данных для соединения и первый байт данных имеет номер 1. Опция **-S** (стр. 264) отключает относительную нумерацию и обеспечивает вывод порядковых номеров в соответствии со значениями в пакетах.

В строке 6 показан пакет, который **rtsg** отправляет хосту **csam** с 19 байтами данных (байты со 2 по 20). Пакет передается с флагом PUSH. Строка 7 содержит отправленное хостом **csam** подтверждение приема данных от хоста **rtsg** вплоть до байта с номером 21 (но не включая этот байт). Большая часть этих данных сохраняется в приемном буфере, поскольку **csam** показывает уменьшение приемного окна на 19 байтов. В этом пакете **csam** также передает хосту **rtsg** 1 байт данных. В строках 8 и 9 показана передача хостом **csam** двух важных (**urg**) байтов данных, отправленных с использованием флага выталкивания **PUSH**.

Если используется достаточно малый кадр захвата (см описание опции **-s** на стр. 264), tcpdump может не получить заголовок TCP полностью. В таких случаях интерпретируется полученная часть заголовка, а в строке вывода помещается маркер **[|tcp]**, показывающий невозможность полной интерпретации. Если заголовок содержит некорректную опцию[2], строка вывода будет содержать маркер **[bad opt]** и последующие опции не будут интерпретироваться, поскольку невозможно корректно определить начало следующей опции. Если поле размера заголовка указывает на присутствие опций, но размер пакета IP недостаточно велик для включения всех опций в пакет, **tcpdump** будет помещать в строке вывода маркер **[bad hdr length]**.

1 *Об этом говорит запись **first:last(nbytes)**, в которой указывается порядковый номер первого байта в этом и следующем за ним (т. е. номер последнего байта в данном пакете + 1) пакетах и число байтов, содержащихся в пакете.*

2 *Размер опции слишком мал или указанный размер опции выходит за пределы указанного размера заголовка.*

11.9.2.3.3.1 Сбор пакетов TCP с заданными комбинациями флагов (SYN-ACK, URG-ACK и т. п.)

Поле флагов заголовка TCP содержит 8 полей:

```
CWR | ECE | URG | ACK | PSH | RST | SYN | FIN
```

Предположим, нам нужно собрать пакеты, используемые для организации нового соединения TCP. Напомним, что протокол TCP использует 3-этапную процедуру организации новых соединений:

1) Инициатор соединения передает пакет с установленным флагом SYN.
2) Получатель этого пакета передает отклик с флагами SYN и ACK.
3) Инициатор передает в ответ пакет с флагом ACK.

Давайте создадим фильтр, который будет собирать пакеты, в которых установлен только флаг SYN (этап 1). Отметим, что пакеты этапа 2 (SYN-ACK) нас не будут интересовать, поскольку они являются просто откликами на стартовый запрос SYN. Прежде, чем строить выражение для фильтра, вспомним структуру заголовка TCP без опций:

Таблица 67. Структура заголовка TCP.

0									**15**	**31**
Порт отправителя										Порт получателя
Порядковый номер										
Номер подтверждения										
HL	резерв	C	E	U	A	P	R	S	F	Размер окна
Контрольная сумма TCP										Указатель важности

Заголовок TCP состоит из 20 октетов, если не используются необязательные поля опций. Первая строка таблицы 67 соответствует октетам 0 - 3, вторая - октетам 4 - 7 и т. д. Поле битов управления (флагов) TCP содержится в октете 13. Пронумеруем биты флагов справа налево (в соответствии с ростом значимости битов).

Таблица. 68 Биты флагов TCP.

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
2^7=128	2^6=64	2^5=32	2^4=16	2^3=8	2^2=4	2^1=2	2^0=1
CWR	ECE	URG	ACK	PSH	RST	SYN	FIN

Таким образом, значение октета флагов при наличии в нем только флага SYN будет составлять

```
0*128 + 0*64 + 0*32 + 0*16 + 0*8 + 0*4 + 1*2 + 0*1 = 2
```

и для выделения пакетов с флагом SYN можно воспользоваться выражением

```
tcp[13] == 2
```

Указав интересующий интерфейс, мы можем собрать пакеты с помощью команды:

```
tcpdump -i <интерфейс> tcp[13] == 2
```

Эту команду можно перевести на человеческий язык словами: “Собрать с указанного интерфейса пакеты TCP, имеющие в тринадцатом октете заголовка значение 2".

Далее предположим, что нам нужно собрать пакеты с флагом SYN независимо от состояния флага ACK и иных флагов. Посмотрим, какое значение будет иметь октет флагов для пакетов SYN-ACK:

```
|C|E|U|A|P|R|S|F|
|0 0 0 1 0 0 1 0|
```

В десятичном формате значение 00010010 будет равно 18

```
0*128 + 0*64 + 0*32 + 1*16 + 0*8 + 0*4 + 1*2 + 0*1 = 2
```

Однако, мы не можем использовать выражение

```
tcp[13] == 18
```

поскольку ему будут соответствовать только пакеты с установленными флагами SYN и ACK, но не будут соответствовать пакеты, имеющие только флаг SYN, который интересует нас в первую очередь.

Для сбора пакетов SYN независимо от значения флага ACK нам следует использовать маску 00000010 (десятичное значение 2). Таким образом, мы можем ввести выражение:

```
tcpdump -i <интерфейс> 'tcp[13] & 2 == 2'
```

которое позволит нам собирать пакеты с установленным флагом SYN независимо от присутствия других флагов.

11.9.2.3.4 Пакеты UDP

Формат вывода пакетов UDP можно проиллюстрировать на примере пакета **rwho**:

```
actinide.who > broadcast.who: udp 84
```

Приведенная строка говорит о том, что порт **who** хоста **actinide** передал дейтаграмму **UDP** в порт **who** с использованием широковещательного адреса IP. Пакет содержит 84 байта пользовательских данных.

Для некоторых служб, работающих по протоколу UDP распознаются протоколы вышележащего уровня (на основе номера порта) и для этих протоколов выводится соответствующая информация. В частности, программа tcpdump

выводит дополнительные сведения для пакетов DNS[1] и вызовов NFS с помощью Sun RPC[2].

11.9.2.3.4.1 Запросы UDP к серверам DNS

Формат вывода для запросов DNS имеет вид

```
src > dst: id op? flags qtype qclass name (len)
```

Например

```
h2opolo.1538 > helios.domain: 3+ A? ucbvax.berkeley.edu. (37)
```

говорит, что хост **h2opolo** запрашивает у сервера имен **helios** адресную запись (qtype=A) для имени **ucbvax.berkeley.edu**. Идентификатор запроса имеет значение **3**. Знак **+** показывает наличие флага **recursion desired**[3]. Размер пакета составляет 37 байтов без учета заголовков UDP и IP. Поскольку пакет содержит нормальный запрос, поле **op** опущено. Если бы это поле содержало что-то иное, соответствующий код был бы выведен между **3** и **+**. Поле **qclass** также содержит стандартное значение **C_IN**, которое опущено при выводе. Любое другое значение **qclass** было бы выведено вслед за символом **A**.

При анализе пакета проверяется наличие в нем аномалий и в результате такой проверки строка вывода может содержать дополнительные поля, заключенные в квадратные скобки. Если запрос содержит секции answer (ответ), authority records (запись о полномочиях) или additional records (дополнительные записи), значения ancount, nscount или arcount выводятся как **[na]**, **[nn]** или **[nau]**, где n показывает значение соответствующего счетчика. Если установлены какие-либо биты отклика (**AA**, **RA** или **rcode**) или в байтах 2 и 3 установлены любые биты **must be zero** (должно быть нулем, выводится поле **[b2&3=x]**, где x - шестнадцатеричное значение байтов 2 и 3 из заголовка.

11.9.2.3.4.2 UDP-отклики от серверов DNS

Для вывода откликов сервера имен используется формат

```
src > dst:  id op rcode flags a/n/au type class data (len)
```

Например,

```
helios.domain > h2opolo.1538: 3 3/3/7 A 128.32.137.3 (273)
helios.domain > h2opolo.1537: 2 NXDomain* 0/1/0 (97)
```

Первая строка показывает, что сервер **helios** отвечает на запрос с **id=3** от хоста **h2opolo** сообщениям с тремя записями **answer**, 3 записями **NS** и 7 дополнительными записями. Первая запись **answer** имеет тип **A** (address - адрес) и содержит IP-адрес указанного в запросе хоста (128.32.137.3). Общий размер отклика составляет 273 байта без учета заголовков UDP и IP. Поля **op** (Query) и код отклика (NoError) были опущены при выводе.

Во второй строке **helios** отвечает на запрос 2 с кодом **NXDomain** (несуществующий домен) без записей **answer**, с одной записью **NS** и без записей **authority**. Символ ***** показывает, установленный бит **authoritative answer**. Ввиду отсутствия записей answer не выводится никакой информации о типе, классе и данных.

В строке вывода могут также появляться индикаторы флагов RA (рекурсия доступна) "**-**" и TC (усеченное сообщение) "**|**". Если секция **question** (вопрос) не содержит в точности одну запись, выводится поле **[nq]**.

Отметим, что запросы и отклики DNS могут быть достаточно велики и принятое по умолчанию значение кадра захвата (68 байтов) может не обеспечить достаточное количество данных из пакета. Для просмотра трафика DNS целесообразно увеличить размер кадра захвата вдвое с помощью опции **-s 128**.

11.9.2.3.4.3 Декодирование SMB/CIFS

Программа tcpdump поддерживает функции декодирования пакетов SMB/CIFS/NBT, использующих порты **UDP/137**, **UDP/138** и **TCP/139**. Поддерживаются также некоторые примитивы декодирования данных **IPX** и **NetBEUI SMB**.

По умолчанию декодирование происходит с минимальным выводом информации, а для увеличения информативности служит опция **-v**. Отметим, что при использовании опции **-v** один пакет **SMB** может занимать больше экранной страницы, поэтому используйте данную опцию только при необходимости, чтобы не утонуть в море выводимых на экран данных.

При декодировании сеансов **SMB**, содержащих текстовые строки Unicode может потребоваться установка переменной окружения **USE_UNICODE =1**.

Информацию о формате пакетов **SMB** вы сможете найти на сайте http://www.samba.org/ или одном из его зеркал.

11.9.2.3.4.4 Запросы и отклики NFS

Запросы и отклики Sun NFS[4] имеют вид:

```
src.xid > dst.nfs: len op args
src.nfs > dst.xid: reply stat len op results
```

Ниже показан пример вывода информации для пакетов NFS

1 *Спецификация протокола DNS (Domain Name System) приведена в RFC 1034 и RFC 1035, которые можно загрузить с сайта http://rfc-editor.org/rfc/ или найти в каталоге Documents/ приложенного к книге компакт-диска.*
2 *Спецификацию протокола RPC (Remote Procedure Call - удаленный вызов процедур) содержится в документе RFC 1050, который вы можете загрузить с сайта http://rfc-editor.org/rfc/rfc1050.txt или найти в каталоге Documents/ приложенного к книге компакт-диска.*
3 *При отсутствии записи на сервере имен следует сделать рекурсивный запрос к другому серверу.*
4 *Network File System - сетевая файловая система.*

```
sushi.6709 > wrl.nfs: 112 readlink fh 21,24/10.73165
wrl.nfs > sushi.6709: reply ok 40 readlink "../var"
sushi.201b > wrl.nfs: 144 lookup fh 9,74/4096.6878 "xcolors"
wrl.nfs > sushi.201b: reply ok 128 lookup fh 9,74/4134.3150
```

Первая строка показывает, что хост **sushi** передает транзакцию с идентификатором 6709[1] хосту **wrl**. Размер запроса составляет 112 байтов без учета заголовков **UDP** и **IP**. Запрашиваемая операция **readlink**[2] для файла с идентификатором (handle) **fh 21,24/10.731657119** была успешно выполнена и хост **wrl** возвращает результат **ok** с содержимым символьной ссылки.

В третьей строке **sushi** запрашивает у хоста **wrl** поиск файла **xcolors** в каталоге **9,74/4096.6878**.

При использовании опции **-v** вывод становится более информативным[3]:

```
sushi.1372a > wrl.nfs: 148 read fh 21,11/12.195 8192 bytes @ 24576
wrl.nfs > sushi.1372a: reply ok 1472 read REG 100664 ids 417/0 sz 29388
```

В первой строке показан запрос **sushi** к хосту **wrl** на чтение 8192 байтов из файла **21,11/12.195**, начиная со смещения 24576. Хост **wrl** возвращает результат **ok**, показанный во второй строке пакет является первым фрагментом отклика, содержащим 1472 байта прочитанных данных. Последующие фрагменты не имеют заголовков **NFS** и **UDP**, поэтому информация об этих пакетов может не появиться на экране, если вы задали в команде тот или иной фильтр. Благодаря использованию опции **-v** выводится также некоторые атрибуты прочитанного файла (тип **REG** - обычный файл, восьмеричное представление прав доступа, идентификаторы владельца и группы, а также размер файла).

При использовании опции **-vv** объем выводимой информации может дополнительно возрасти.

Отметим, что запросы NFS могут быть достаточно большими и при использовании опции **-v** выводимая информация может занять несколько экранных страниц. В некоторых случаях будет полезно уменьшить размер кадра захвата с помощью опции **-s** (например, **-s 192)**.

Отклики NFS не идентифицируют явно операции RPC. Программа tcpdump сохраняет информацию о последних запросах и при выводе откликов указывает соответствующие идентификаторы транзакций.

11.9.2.3.5 Запросы и отклики AFS

Вывод информации для запросов и откликов AFS[4] имеет вид:

```
src.sport > dst.dport: rx packet-type
src.sport > dst.dport: rx packet-type service call call-name args
src.sport > dst.dport: rx packet-type service reply call-name args
```

Ниже показан пример вывода информации для пакетов AFS

```
elvis.7001 > pike.afsfs:
     rx data fs call rename old fid 536876964/1/1 ".newsrc.new"
     new fid 536876964/1/1 ".newsrc"
pike.afsfs > elvis.7001: rx data fs reply rename
```

в первой строке хост **elvis** передает пакет **RX** хосту **pike**. Этот пакет адресован файловому серверу (**fs**) и начинает вызов удаленной процедуры (RPC). Вызов RPC содержит команду **rename** (переименовать) с идентификатором старого каталога **536876964/1/1** и именем **.newsrc.new**, а также новым идентификатором **536876964/1/1** и именем **.newsrc**. Хост **pike** возвращает отклик RPC с информацией об успешном изменении имени файла.

В общем случаем все пакеты AFS RPC декодируются по крайней мере как имена процедур RPC. Во многих случаях также декодируется один или несколько передаваемых процедуре аргументов.

Формат вывода должен быть достаточно понятен для тех, кто знаком с AFS и RX.

При использовании опции **-v** обеспечивается вывод дополнительной информации Идентификаторы вызовов RX, номера вызовов, порядковые номера, флаги пакетов RX). Опция **-vv** дополнительно увеличивает объем выводимой информации (в частности, сведений о согласовании MTU для пакетов RX ack), а опция **-vvv** обеспечивает также вывод параметров безопасности и идентификаторов сервиса.

Для пакетов **abort** выводятся коды ошибок (за исключением пакетов Ubik, поскольку эти пакеты используются для обозначения пакетов **yes vote** протокола Ubik).

Отметим, что запросы AFS могут быть достаточно велики, поэтому использование флага **-v** иной раз будет приводить к выводу для пакета многостраничной информации. Вы можете задать размер кадра захвата с помощью опции **-s** для обеспечения более читаемых результатов (например, **-s 256**).

Отклики AFS явно не идентифицируют операции RPC, поэтому tcpdump отслеживает последние запросы и помечает отклики идентификаторами соответствующих запросов.

11.9.2.3.5.1 KIP AppleTalk (DDP in UDP)

Пакеты AppleTalk DDP, инкапсулированные в дейтаграммы UDP, извлекаются из дейтаграмм и отображаются как пакеты DDP (т. е., все заголовки UDP отбрасываются). Для преобразования имен сетей и хостов AppleTalk используется файл **/etc/atalk.names**, строки которого имеют форму **адрес (номер) - имя**

1 *Отметим, что номер, указанный после имени отправителя, задает не порт, а номер транзакции.*
2 *Прочесть символьную ссылку.*
3 *При использовании опции **-v** будут выводиться также поля заголовков **IP** (**TTL**, **ID**, **length**, **fragmentation**), которые были опущены в приведенном примере.*
4 *Andrew File System.*

```
1.254     ether
16.1      icsd-net
1.254.110 ace
```
В приведенном примере первые две строки содержат имена сетей AppleTalk, а третья строка - имя хоста[1]. Для разделения номера и имени в файле могут использоваться пробелы или символы табуляции. Файл **/etc/atalk.names** может содержать пустые строки и строки комментариев, начинающиеся с символа **#**.

Адреса AppleTalk выводятся в формате

```
net.host.port
```
Например,

```
144.1.209.2 > icsd-net.112.220
office.2 > icsd-net.112.220
jssmag.149.235 > icsd-net.2
```
Если файл **/etc/atalk.names** не содержит записи для той или иной сети или хоста, соответствующее поле выводится в цифровом формате. В первой строке показан пакет NBP (DDP порт 2) отправленный узлом **209** сети **144.1** в порт **220** узла **112** сети **icsd-net**. Вторая строка отличается от первой только тем, что указано также символьное имя отправителя (**office**). В третьей строке показан пакет, отправленный из порта **235** хостом **149** сети **jssmag** всем хостам[2] сети **icsd-net**, прослушивающим порт NBP

Пакеты протоколов NBP (name binding protocol) и ATP (AppleTalk transaction protocol) выводятся с интерпретацией их содержимого. Для остальных протоколов просто выводится дамп имени протокола или его номера, если имя неизвестно, и размер пакета.

Пакеты NBP выводятся в формате, подобном приведенному ниже:

```
icsd-net.112.220 > jssmag.2: nbp-lkup 190: "=:LaserWriter@*"
jssmag.209.2 > icsd-net.112.220: nbp-reply 190: "RM1140:LaserWriter@*" 250
techpit.2 > icsd-net.112.220: nbp-reply 190: "techpit:LaserWriter@*" 186
```
Первая строка показывает запрос на преобразование имени для принтеров **LaserWriter**, переданный хостом **112** сети **icsd** по широковещательному адресу сети **jssmag**. Идентификатор запроса **nbp** имеет значение 190. Вторая строка содержит отклик на этот запрос от хоста **jssmag.209**, сообщающего о наличии ресурса **LaserWriter** с именем **RM1140**, зарегистрированного на порту **250**. В третьей строке показан другой отклик на тот же запрос, говорящий, что хост **techpit** имеет ресурс **LaserWriter** с именем **techpit**, зарегистрированный на порту 186.

Пример формата вывода пакетов ATP показан ниже:

```
jssmag.209.165 > helios.132: atp-req  12266<0-7> 0xae030001
helios.132 > jssmag.209.165: atp-resp 12266:0 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp 12266:1 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp 12266:2 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp 12266:3 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp 12266:4 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp 12266:5 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp 12266:6 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp*12266:7 (512) 0xae040000
jssmag.209.165 > helios.132: atp-req  12266<3,5> 0xae030001
helios.132 > jssmag.209.165: atp-resp 12266:3 (512) 0xae040000
helios.132 > jssmag.209.165: atp-resp 12266:5 (512) 0xae040000
jssmag.209.165 > helios.132: atp-rel  12266<0-7> 0xae030001
jssmag.209.133 > helios.132: atp-req* 12267<0-7> 0xae030002
```
хост **jssmag.209** инициирует транзакцию **12266** с хостом **helios**, запрашивая до 8 пакетов (**<0-7>**). Шестнадцатеричное число в конце строки содержит значение поля **userdata** из запроса.

Хост **helios** отвечает на полученный запрос 8 пакетами по 512 байтов. Число после номера транзакции указывает порядковый номер пакета для данной транзакции, а число в скобках - размер данных в пакете без учета заголовка ATP. Символ ***** для пакета 7 показывает наличие флага **EOM**.

Хост **jssmag.209** после получения пакетов запрашивает повторную передачу пакетов 3 и 5 и **helios** повторяет эти пакеты, после чего **jssmag.209** завершает транзакцию. В последней строке показан новый запрос хоста **jssmag.209**. Символ ***** показывает, что флаг **XO** (exactly once) для пакета не установлен.

11.9.2.3.5.2 Фрагментация IP

Фрагментированные дейтаграммы IP выводятся в формате:

```
(frag id:size@offset+)
(frag id:size@offset)
```
Знак **+** в первой строке показывает наличие дополнительных фрагментов.

Поле **id** показывает идентификатор фрагмента, **size** - его размер в байтах без учета заголовка IP, а **offset** - смещение (в байтах) фрагмента в исходной дейтаграмме.

Информация выводится для каждого фрагмента. Первый фрагмент выводится с заголовком протокола вышележащего уровня и сведениями о фрагментации. Все последующие фрагменты не включают заголовка протокола вышележащего уровня и сведения о фрагменте выводятся сразу после адресов отправителя и

1 *Хост отличается от сети наличием в номере третьего октета.*
2 *Отметим, что широковещательный адрес 255 показывается просто именем или номером сети без указания хоста. По этой причине разумно сохранять имена хостов и сетей в файле **/etc/atalk.names** по-отдельности.*

получателя. Ниже приведен пример пакетов связанных с передачей файла по протоколу FTP с сайта arizona.edu на хост **rtsg** через сеть CSNET, которая не поддерживает передачу дейтаграмм размером 576 байтов.

```
arizona.ftp-data > rtsg.1170: . 1024:1332(308) ack 1 win 4096 (frag 595a:328@0+)
arizona > rtsg: (frag 595a:204@328)
rtsg.1170 > arizona.ftp-data: . ack 1536 win 2560
```

Отметим, что адреса во второй строке приведены без номеров портов, поскольку фрагменты (за исключением первого) не содержат заголовков TCP, что не позволяет получить сведений о номера портов и порядковых номерах TCP. Порядковые номера в первой строке показывают наличие в пакет 308 байтов пользовательских данных, хотя фактически в дейтаграмме содержится 512 байтов (308 байтов в первом фрагменте и 204 - во втором).

Пакеты с флагом запрета фрагментирования (don't fragment) помечаются символами **DF**.

11.9.2.3.5.3 Временные метки

По умолчанию каждая строка вывода включает временную метку, содержащую время захвата кадра в формате:

```
hh:mm:ss.frac
```

Точность отображения времени зависит от точности системного таймера. Временная метка фиксирует момент, когда кадр становится доступным ядру системы. Время между получением кадра из среды передачи и его доставкой ядру не принимается во внимание.

11.9.3 Анализатор протоколов Ethereal

http://www.ethereal.com

Ethereal представляет собой анализатор сетевых протоколов с графическим интерфейсом (GUI)[1]. Программа позволяет просматривать и анализировать пакеты, полученные из сетевого интерфейса или ранее собранного файла. В Ethereal по умолчанию используется для файлов захвата формат libpcap, используемый программой tcpdump (см. параграф 11.9.2 на стр. 262) и другими анализаторами. Кроме того, Ethereal может читать файлы в форматах snoop и atmsnoop, Shomiti/Finisar Surveyor, Novell LANalyzer, Network General/Network Associates Sniffer[2] (DOS-версии), Microsoft Network Monitor, AIX iptrace, Cinco Networks NetXRay, Network Associates Sniffer (Windows-версии), AG Group/WildPackets EtherPeek/TokenPeek/AiroPeek, RADCOM WAN/LAN, Lucent/Ascend router debug, HP-UX nettl, Toshiba ISDN router dump, ISDN4BSD, Cisco Secure IDS IPLog, pppd (формат pppdump), VMS TCPIPtrace/TCPtrace/UCX$TRACE, DBS Etherwatch VMS (текстовый формат, Visual Networks Visual UpTime, CoSine L2, Accellent 5Views LAN agent, Endace Measurement Systems ERF, Linux Bluez Bluetooth, Network Instruments Observer v9. Программе даже не нужно указывать тип исходного файла, она распознает форматы автоматически. Ethereal может также читать файлы перечисленных выше форматов, сжатые с использованием gzip. Получить более подробную информацию о поддерживаемых программой форматах и других возможностях Ethereal можно из руководства пользователя, доступного на сайте[3].

Подобно другим анализаторам протоколов окно Ethereal включает 3 области просмотра с разными уровнями детализации (см. рисунок 11.25). Верхнее окно содержит список собранных пакетов с кратким описанием, в среднем окне показывается дерево протоколов, инкапсулированных в кадр. Ветви дерево могут быть раскрыты для повышения уровня детализации выбранного протокола. Последнее окно содержит дамп пакета в шестнадцатеричном и текстовом представлении.

Программа Ethereal предоставляет пользователю ряд уникальных возможностей, не поддерживаемых другими анализаторами протоколов. Программа обеспечивает возможность сбора всех пакетов заданного соединения TCP и представления данных в удобном для просмотра формате (ASCII, EBCDIC или шестнадцатеричный). При выводе пакетов можно использовать мощную систему фильтрации Ethereal, отбирающую пакеты по большему, нежели в других анализаторах, числу полей.

1 *Приведенное здесь описание соответствует версии 0.9.16. В каталоге SRC/приложенного к книге компакт-диска вы сможете найти исходные тексты более новой версии.*
2 *В сжатом и несжатом формате*
3 *Копия этого руководства содержится в каталоге Documents/ приложенного к курсу компакт-диска.*

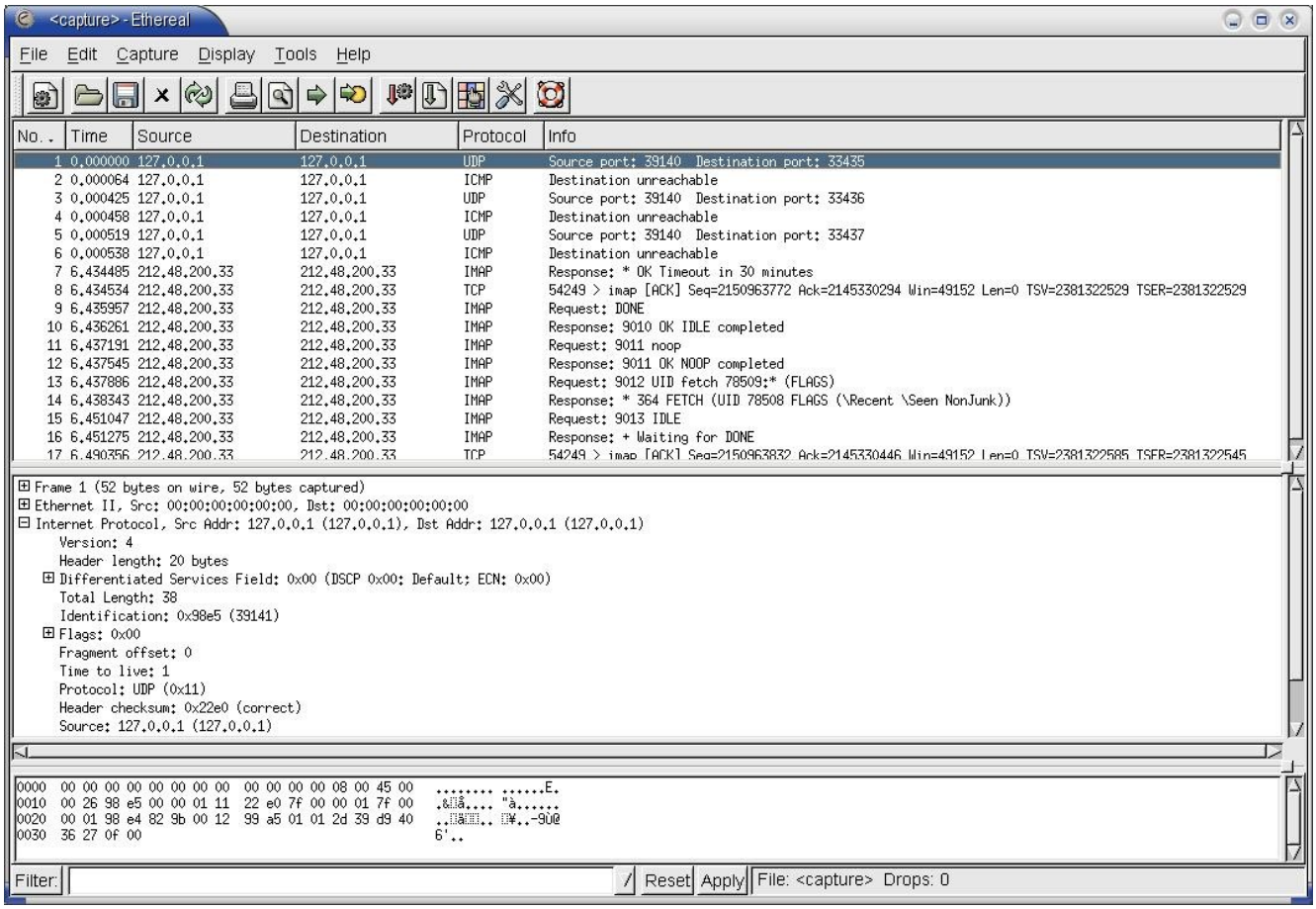


Рисунок 11.25 Интерфейс программы Ethereal

Сбор пакетов осуществляется с использованием библиотеки pcap (см. параграф 11.9.1 на стр. 261), входящей во все дистрибутивы UNIX[1]. Синтаксис фильтров сбора пакетов соответствует правилам, используемым библиотекой pcap[2] (см. параграф 11.9.2.2 на стр. 265).

Для поддержки анализа данных из сжатых файлов требуется библиотека zlib. Отсутствие этой библиотеки не мешает компиляции Ethereal, но в этом случае работа со сжатыми файлами не поддерживается.

11.9.3.1 Опции Ethereal

Большинство пользователей запускает графический интерфейс Ethereal без каких-либо опций, однако приведенная в этом параграфе информация позволяет более эффективно использовать возможности программы за счет выбора режима работы с помощью опций командной строки.

`-a`

задает для программы Ethereal критерий прекращения записи в файл захвата. Критерий может иметь формат **test:value**, где параметр **test** может принимать значения:

`duration`

задает продолжительность сбора пакетов в секундах.

`filesize`

задает максимальный размер файла в тысячах байтов (не килобайтах = 1024 байт).

`-b`

если задан максимальный размер файла захвата, эта опция заставляет Ethereal работать в режиме кольцевого буфера **(ring buffer)** с указанным числом файлов. В режиме кольцевого буфера Ethereal будет записывать собранные данные в несколько файлов, давая им имена по дате и времени создания файла.

После заполнения первого файла Ethereal перейдет к записи во второй и так далее, пока не будет создано указанное число файлов. При достижении максимального числа файлов первый файл (самый старый) будет уничтожен перед созданием нового файла. Если для максимального числа файлов указано значение 0, сбор данных будет продолжаться неограниченно долго.

Если задана продолжительность сбора пакетов, Ethereal будет создавать новый файл по истечении заданного времени даже в тех случаях когда текущий файл не достиг максимального размера.

`-B <высота>`

задает начальную высоту (в пикселях) панели дампа пакетов (нижняя часть окна программы, показанного на рисунке 11.25).

`-c <число пакетов>`

задает используемое по умолчанию число пакетов для чтения при сборе “живых” данных.

1 *Windows-версия Ethereal работает с библиотекой winpcap, доступной на сайте http://winpcap.polito.it.*
2 *Для фильтров отображения пакетов используется другой синтаксис.*

-f <фильтр>
задает выражение для фильтра захвата пакетов.

-h
выводит информацию о номере версии и опциях программы, после чего завершает работу.

-i <интерфейс>
задает имя сетевого интерфейса или канала (pipe), используемого для сбора пакетов.

Имена сетевых интерфейсов должны соответствовать именам из списка поддерживаемых системой, которым можно получить по команде **tethereal -D**. Для Unix-систем список присутствующих в системе интерфейсов можно получить также по команде **netstat -i** или **ifconfig -a** (последняя команда может не работать в старых версиях Unix).

В качестве имен каналов могут использоваться имена буферов FIFO (named pipe - именованные каналы) или символ "**-**" для сбора пакетов со стандартного устройства ввода. Получаемые из канала данные должны использовать стандартный формат libpcap.

-k
инициирует начало сбора пакетов. Если задан флаг **-i**, пакеты собираются только с указанного этим параметром интерфейса. Если интерфейс не задан, Ethereal находит список интерфейсов системы и выбирает из него первый интерфейс, пропуская loopback. Если в системе присутствует только интерфейс loopback, Ethereal выводит сообщение об ошибке, не начиная сбор пакетов.

-l
включает автоматическую прокрутку списка пакетов, если включен режим автоматического обновления списка по мере захвата пакетов (опция **-S**).

-L
выводит список поддерживаемых типов кадров канального уровня, после чего работа программы завершается.

-m <имя шрифта>
задает имя растрового шрифта, используемого программой Ethereal в большинстве случаев. Для вывода в панели дампа данных, соответствующих выбранному в панели дерева протоколов полю, жирным шрифтом (bold) Ethereal будет создавать имя шрифта на основе имени, заданного этой опцией.

-n
отключает функцию определения имен сетевых объектов (например, названий портов TCP и UDP, имен хостов).

-N <тип>
включает функцию определения имен для отдельных типов адресов и номеров портов; прочие типы адресов и номера портов выводятся в цифровом представлении. Поле **<тип>** содержать значение **m** для преобразования MAC-адресов, **n** для преобразования адресов сетевого уровня (IP) или **t** для преобразования номеров портов. Данная опция отменяет действие флага **-n** при одновременном использовании. Значение **C** добавляет поддержку асинхронных запросов DNS.

-o <имя:значение> [<имя:значение> ...]
задает предпочтительное значение указанного параметра. Это значение отменяет принятое по умолчанию и указанное в файле предпочтений значение параметра. Имена параметров должны совпадать с именами в файле предпочтений.

-p
указывает программе, что интерфейс не нужно переводить в режим захвата[1].

-P <число пикселей>
задает начальную высоту панели со списком пакетов (верхняя панель программы, см. рис. 11.25).

-Q
задает завершение работы программы Ethereal после окончания сеанса сбора пакетов. Эта опция полезна при работе в пакетом режиме, задаваемом флагом **-c**. Для использования данной опции в командной строке должны присутствовать также флаги **-i** и **-w**.

-r <имя файла>
задает чтение пакетов из файла.

-R <фильтр>
при использовании совместно с флагом чтения данных из файла (**-r**) эта опция активизирует указанный фильтр[2] для всех читаемых из файла пакетов. Не соответствующие фильтру пакеты просто отбрасываются.

-S
задает выполнение операций по захвату пакетов в форме отдельного процесса с автоматическим обновлением отображаемого списка собранных пакетов.

-s <размер захвата>
задает принятый по умолчанию размер захвата (snapshot length) для использования при "живом" сборе данных. Для каждого пакета в память или на диск записывается не более заданного этим параметром числа байтов.

-T <число пикселей>

1 *Интерфейс может быть переведен в режим захвата другими программами, поэтому использование флага **-p** отнюдь не гарантирует работу интерфейса в обычном режиме - программа просто не будет переводить этот интерфейс в режим захвата. Кроме того, даже в обычном режиме захватываться будут не только пакеты, адресованные этому интерфейсу, поскольку в сети всегда присутствуют широковещательные пакеты и могут использоваться пакеты с групповыми адресами (multicast).*
2 *Для этого фильтра используется синтаксис фильтров отображения, а не фильтров захвата.*

задает начальную высоту панели дерева протоколов (средняя панель на рисунке 11.25).

`-t <формат>`
задает формат временных меток для списка пакетов - **r** (relative - относительно времени старта), **a** (absolute - абсолютное время), **ad** (absolute with date - абсолютное время с указанием даты) или **d** (delta - интервал после захвата предыдущего пакета). По умолчанию временные метки выводятся относительно начала захвата (**r**).

`-v`
задает вывод номера версии программы и завершение работы.

`-w <имя файла>`
задает используемое по умолчанию имя файла захвата.

`-y <тип>`
при захвате пакетов, инициированном флагом **-k**, эта опция задает тип канального уровня для сеанса сбора. В качестве значения параметра могут использоваться значения, идентификаторы типов, выводимые при использовании команды с флагом **-L**.

`-z <параметры>`
задает программе Ethereal необходимость сбора статистики и вывода результатов в окне с периодическим обновлением содержимого. В настоящее время поддерживается несколько параметров сбора статистики:

`-z dcerpc,srt,uuid,major.minor[,filter]`
Программа собирает данные **SRT**[1] для вызовов/откликов интерфейса **DCERPC** с идентификатором **uuid**, версии **major.minor**. К собираемым данным относятся число вызовов каждой процедуры, MinSRT, MaxSRT и AvgSRT[2]. Например опция **-z dcerpc,srt,12345778-1234-abcd-ef00-0123456789ac,1.0** будет обеспечивать сбор информации для интерфейса CIFS SAMR. Такие опции можно использовать в командной строке неоднократно.

При использовании необязательного фильтра в результатах будут учитываться только соответствующие фильтру статистические данные. Например, опция **-z dcerpc,srt,12345778-1234-abcd-ef00-0123456789ac,1.0,ip.addr==1.2.3.4** будет обеспечивать сбор статистики **SAMR SRT** только для хоста с IP-адресом 1.2.3.4. Опция

`-z io,stat`
будет обеспечивать статистику по захвату кадров и байтов в течение каждой секунды. При использовании такой опции будет открываться диалоговое окно **IO-Stat** (см. рисунок 11.26), содержащее статистическую информацию с отдельным графиком для каждого фильтра. Такие опции можно использовать в командной строке неоднократно.

Окно статистики, показанное на рисунке 11.26, можно открыть также с использованием меню **Tools|Statistics|Traffic|IO-Stat** (параграф 11.9.3.2.7.8.3.1 на стр 298).

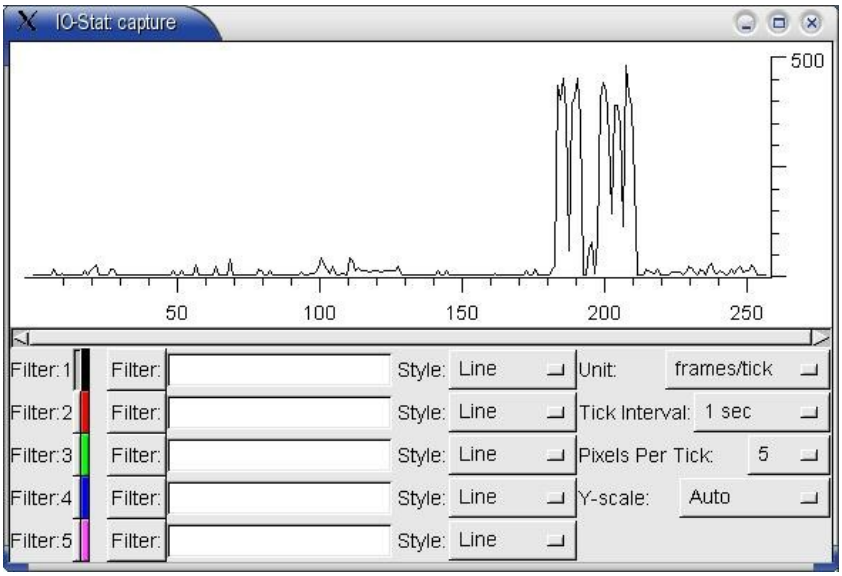


Рисунок 11.26 Окно IO-Stat

`-z rpc,srt,program,version[,<filter>]`
обеспечивает сбор статистики вызовов/откликов SRT для указанной программы и версии. Данные включают число вызовов для каждой процедуры, MinSRT, MaxSRT и AvgSRT. Например, опция **-z rpc,srt,100003,3** позволит собрать статистику вызовов для NFS v3. Допускается неоднократное использование опции в командной строке.

При указании в строке необязательного фильтра статистика будет выводиться только для соответствующих этому фильтру пакетов. Например, опция **-z rpc,srt,100003,3,nfs.fh.hash==0x12345678** задает сбор статистики NFS v3 SRT только для указанного файла.

`-z rpc,programs`
задает сбор статистики вызовов/откликов RTT для всех известных программ и версий **ONC-RPC**. Данные включают число вызовов , MinRTT, MaxRTT и AvgRTT.

`-z smb,srt[,filter]`
задает сбор статистики SRT для протокола SMB. Данные включают число вызовов каждой команды SMB, MinSRT, MaxSRT и AvgSRT.

Данные представляются в отдельных таблицах для всех нормальных команд каждой SMB, а также команд Transaction2 и всех команд NT. Отображается статистика только для тех команд, которые встретились в собранных пакетах. В цепочке команд **xAndX** для расчета используется только первая команда. Так, для распространенных цепочек **SessionSetupAndX** + **TreeConnectAndX** в статистике будут учитываться только вызовы **SessionSetupAndX**. Это ограничение планируется снять в будущих версиях программы. Допускается неоднократное использование опции в командной строке.

При использовании в командной строке необязательного фильтра статистика рассчитывается только с учетом

1 *Service Response Time - время отклика службы.*
2 *Минимальное, максимальное и среднее время отклика, соответственно.*

пакетов, соответствующих заданному фильтру. Например, опция **-z "smb,srt,ip.addr==1.2.3.4"** будет выводить статистику только для пакетов SMB, обмен которыми происходит с сайтом, имеющим IP-адрес 1.2.3.4 .

-z fc,srt[,filter]

собирает статистику SRT для FC[1]. Данные включают число вызовов каждой команды Fibre Channel, MinSRT, MaxSRT и AvgSRT. Значение времени отклика SRT рассчитывается как временной интервал от первого до последнего кадра в сеансе обмена данными. Данные представляются в виде отдельной таблицы для каждой нормальной команды FC. Выводятся, данные только для тех команд, которые встречались в собранных пакетах. Допускается неоднократное использование опции в командной строке.

При использовании фильтра статистика рассчитывается с использованием только тех данных, которые соответствуют заданному фильтру. Например, опция **-z "fc,srt,fc.id==01.02.03"** задает сбор статистики только для обмена данными с хостом, имеющим FC-адрес 01.02.03 .

-z mgcp,srt[,filter]

задает сбор статистики SRT для MGCP. Выводятся данные по вызовам для каждого известного типа MGCP Type, Minimum SRT, Maximum SRT и Average SRT. Допускается неоднократное использование опции в командной строке.

При указании в командной строке фильтра, расчет статистики проводится только с учетом тех данных, которые соответствуют условиям фильтрации. Например, опция **-z "mgcp,srt,ip.addr==1.2.3.4"** задает сбор статистики MGCP только для пакетов, обмен которыми осуществляется с IP-хостом 1.2.3.4 .

-z conv,type[,filter]

создает таблицу, в которой указываются список всех сеансов обмена данными (conversation - “разговор”) в собранных пакетах. Параметр **type** задает тип соединений, для которых нужно генерировать статистику. Поддерживаются типы

eth - Ethernet;

fc - адреса Fibre Channel;

fddi - адреса FDDI;

ip - IP-адреса;

ipx - адреса IPX;

tcp - пары сокетов TCP/IP (поддерживаются протоколы IPv4 и IPv6);

tr - Token Ring;

udp - пары сокетов UDP/IP (поддерживаются протоколы IPv4 и IPv6).

Если командная строка включает фильтр, статистика генерируется только для тех соединений, которые соответствуют заданному фильтру.

Выводимая таблица содержит по одной строке для каждого соединения и показывает число пакетов/байтов, переданных в каждом направлении, а также общее число пакетов/кадров. Строки таблицы сортируются в в соответствии с общим числом кадров для соединения.

Для просмотра таблицы можно также воспользоваться во время сбора пакетов опцией меню **Tools|Statistics| Conversation List** (параграф 11.9.3.2.7.8.3 на стр. 298).

-z h225,counter[,filter]

собирает сообщения ITU-T H.225 и сведения о причинах (reason) их генерации. В первой колонке создаваемой этой опцией таблицы указывается список сообщений H.225 и их причин для собранных программой пакетов. Вторая колонка таблицы указывает количество для каждого сообщения и причины. Допускается неоднократное использование опции в командной строке.

При указании в командной строке фильтра статистика будет рассчитываться только с учетом соответствующих этому фильтру пакетов. Например, опция **-z "h225,counter,ip.addr==1.2.3.4"** задает расчет статистики H.225 только для пакетов, обмен которыми ведется с IP-хостом 1.2.3.4 .

11.9.3.2 Графический интерфейс Ethereal

Работа с программой Ethereal построена на базе графического интерфейса (GUI), показанного на рисунке 11.27. Режим захвата и отображения пакетов задается с помощью опций командной строки и описанных в последующих параграфах команд меню и диалоговых окон.

11.9.3.2.1 Главное окно программы

Главное окно Ethereal разделено на три панели. Размеры каждой из панелей можно менять, используя маркер в нижней правой части соответствующей панели.

11.9.3.2.1.1 Верхняя панель

Верхняя панель окна Ethereal содержит список пакетов. По умолчанию в списке выводится 6 колонок - номер пакета в списке собранных, временная метка, адреса и номера портов отправителя и получателя, протокол и краткое описание пакета. Вы можете изменить набор отображаемых колонок с помощью страницы **Columns** (стр. 302) диалогового окна **Preferences** (см. параграф 11.9.3.2.10 на стр. 301). Для активизации диалогового окна можно использовать команду меню **Edit:Preferences** (см. параграф 11.9.3.2.3.9 на стр. 285) или кнопку на панели

1 *Fibre Channel*

инструментов окна Ethereal.

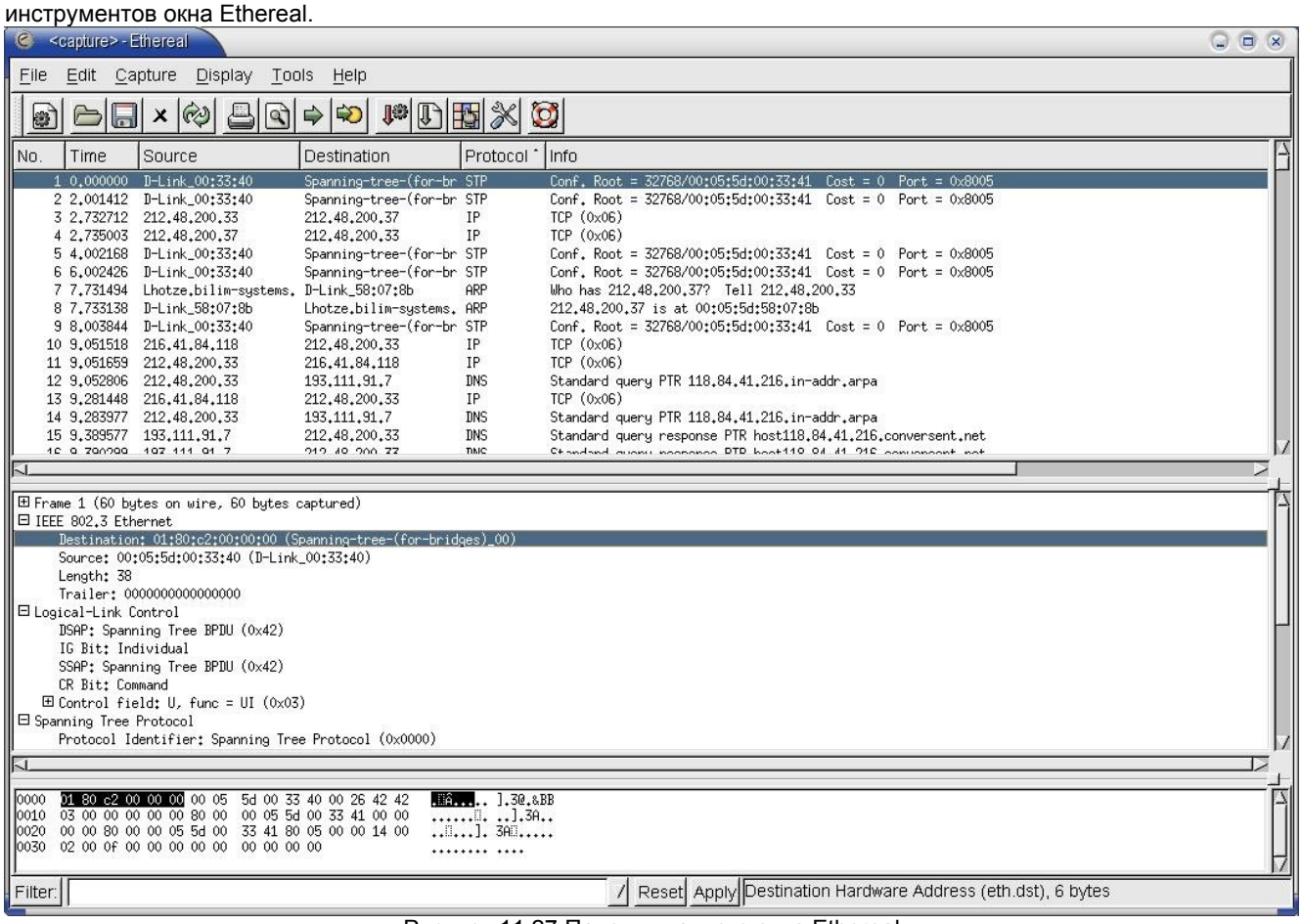


Рисунок 11.27 Панели главного окна Ethereal

Щелкнув кнопкой мыши на поле с именем колонки в верхней строке списка пакетов, вы можете задать сортировку пакетов по содержимому данной колонки. Повторный щелчок на этом поле изменит порядок сортировки.

В полях адресов выводится информация максимально доступного уровня. Например, для кадров Ethernet, содержащих пакеты IP будут указываться адреса IP, но если тип передаваемого в кадрах протокола неизвестен, поле будет содержать MAC-адрес.

Правая кнопка мыши активизирует всплывающее меню для списка пакетов.

Средняя кнопка мыши может использоваться для маркировки пакетов в списке.

11.9.3.2.1.2 Средняя панель

Средняя панель окна Ethereal содержит дерево протоколов для выбранного из списка верхней панели пакета. Дерево отображает каждое поле и его значение для заголовков всех протоколов стека. Структуру каждой ветви дерева можно раскрыть или свернуть, нажимая кнопку мыши на квадратике в начале строки соответствующего протокола.

Правая кнопка мыши активизирует всплывающее меню для панели дерева протоколов.

11.9.3.2.1.3 Нижняя панель

Нижняя панель окна содержит дамп указанного в списке пакета в шестнадцатеричном и ASCII-формате. Выбранное в панели дерева протоколов поле выделяется цветом соответствующей области дампа, как показано на рисунке 11.27.

Правая кнопка мыши активизирует всплывающее меню для панели дампа.

11.9.3.2.1.4 Панель Filter

Размещенная в строке состояния окна Ethereal панель управления фильтрами отображения позволяет выбирать фильтр из числа сохраненных или задавать условия фильтрации непосредственно в строке ввода. Кнопка **Apply** служит для активизации фильтра, кнопка **Reset** отключает текущий фильтр отображения. Для создания и редактирования фильтров отображения используется диалоговое окно (параграф 11.9.3.2.3.11.1 на стр. 286), активизируемое с помощью команды меню **Edit:Display Filters** (параграф 11.9.3.2.3.11 на стр. 286) или кнопки на панели инструментов Ethereal (параграф 11.9.3.2.9 на стр. 301).

Фильтр для отображения только трафика HTTP, HTTPS и DNS может иметь форму:

```
tcp.port == 80 || tcp.port == 443 || tcp.port == 53
```

Кнопка **Filter** позволяет выбрать фильтр из числа созданных ранее. После выбора фильтра или его ввода в строке набора нажмите кнопку **Apply** или клавишу **Enter** для активизации указанного фильтра. Кнопка **Reset** сбрасывает фильтр и отключает фильтрацию, обеспечивая вывод всех собранных программой пакетов.

11.9.3.2.2 Меню File

Функции меню **File** служат для выбора файлов при анализе собранных ранее данных, сохранения собранных программой пакетов, печати информации о пакетах, а также для завершения работы программы.

11.9.3.2.2.1 Open, Close, Reload

Команды чтения (**Open**), закрытия (**Close**) и повторной загрузки (**Reload**) файла собранных пакетов. Диалоговое окно **File:Open** (см. рисунок 11.28) позволяет использовать указать фильтр, который будет использоваться по отношению к загружаемым из файла пакетам (фильтр захвата). Вы можете выбрать существующий фильтр или создать новый фильтр для обработки данного файла.

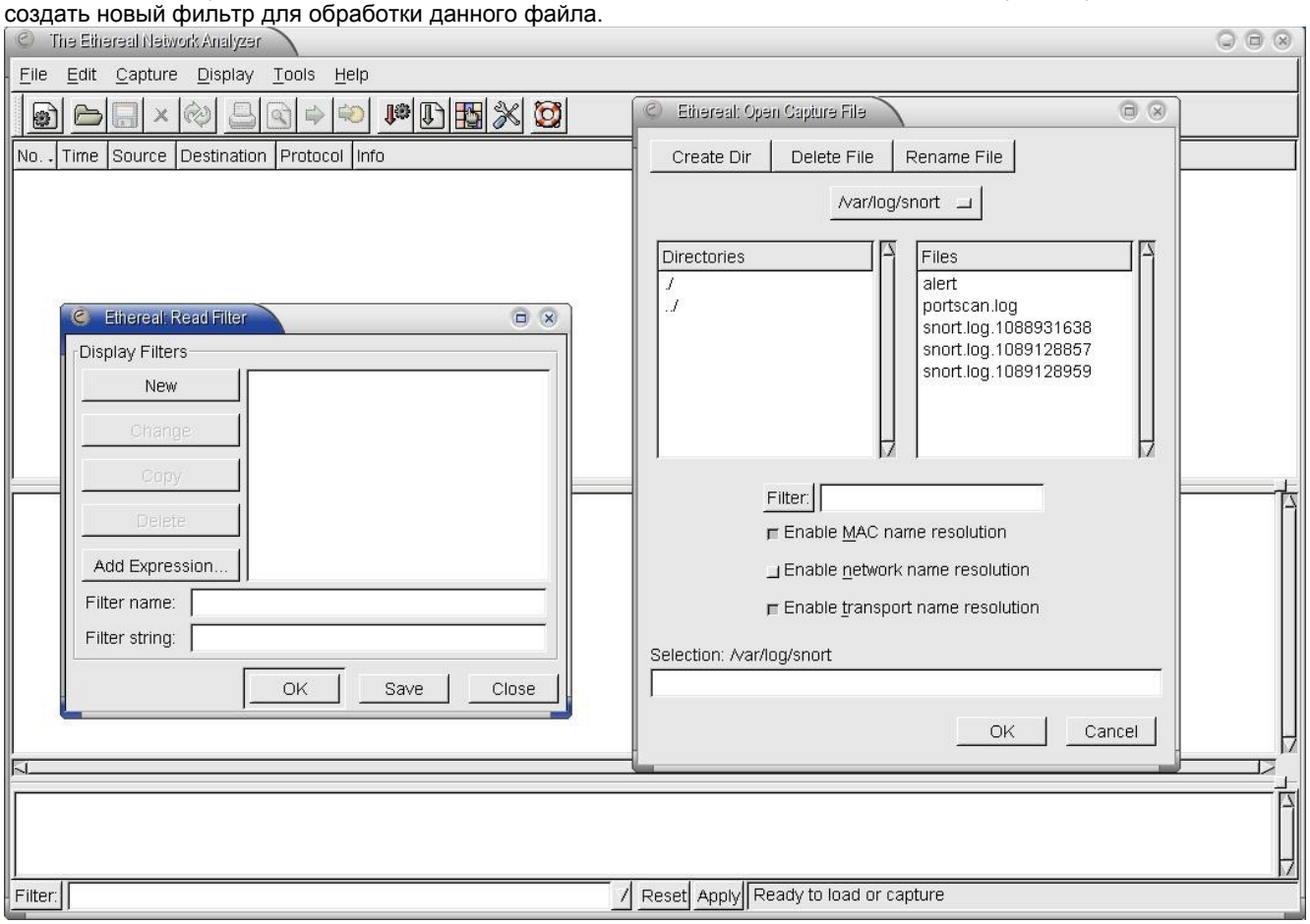


Рисунок 11.28 Диалоговые окна выбора файлов (справа) и фильтров

Перечисленные здесь команды работы с файлами захвата доступны также с помощью кнопок, выведенных на панель инструментов Ethereal (параграф 11.9.3.2.9 на стр. 301), или комбинаций клавиш **Ctrl+O** (Open), **Ctrl+W** (Close), **Ctrl+R** (Reload).

11.9.3.2.2.2 Save, Save As

Команды сохранения и записи с новым именем для файлов захвата. Поле выбора **Save only packets currently being displayed** (см. рисунок 11.29) позволяет записать в файл только пакеты, удовлетворяющие условиям фильтрации при отображении (параграф 11.9.3.2.3.11 на стр. 286), а поле **Save only marked packets** - только отмеченные пакеты. Вы можете также выбрать из списка (см. рисунок 11.30) один из поддерживаемых программой форматов записи файлов захвата.

Для записи собранных программой пакетов на диск можно также использовать кнопку на панели инструментов программы Ethereal (см. параграф 11.9.3.2.9 на стр. 301) или комбинацию клавиш **Ctrl+S**.

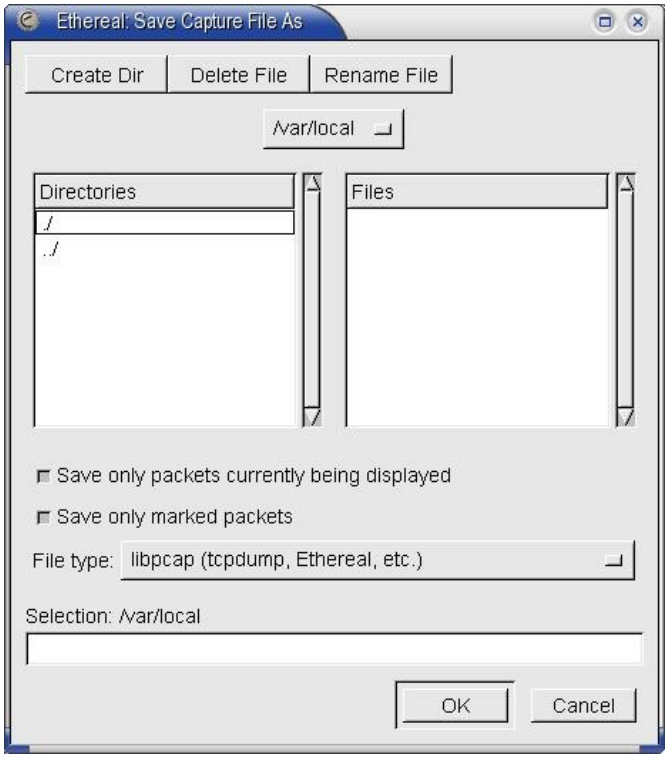


Рисунок 11.29 Диалоговое окно записи файла

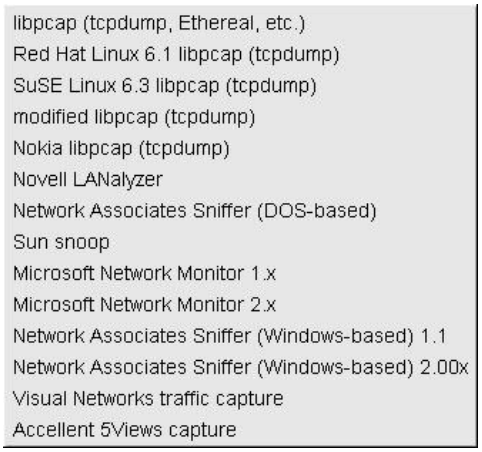


Рисунок 11.30 Выбор формата записи файла

11.9.3.2.2.3 Print

Эта команда служит для вывода на печать информации для всех собранных пакетов или отмеченных пакетов из списка. Для каждого пакета выводится строка из списка пакетов (**Print summary**) или содержимое окна дерева каталогов (**Print detail**). В режиме детального вывода может также печататься шестнадцатеричный дамп каждого пакета (опция **Print hex data**). Кроме того, для этого режима можно выбрать вывод всего дерева протоколов (**Expand all levels**) или только раскрытых ветвей дерева (**Print as displayed**).

Операцию печати можно активизировать также с помощью кнопки на панели инструментов программы Ethereal (см. параграф 11.9.3.2.9 на стр. 301).

Опции печати общего назначения (принтер, формат и т. п.) можно указать в диалоговом окне **Print** (рисунок 11.31) или на странице **Printing** (параграф 11.9.3.2.10.1 на стр. 301) диалогового окна **Preferences**.

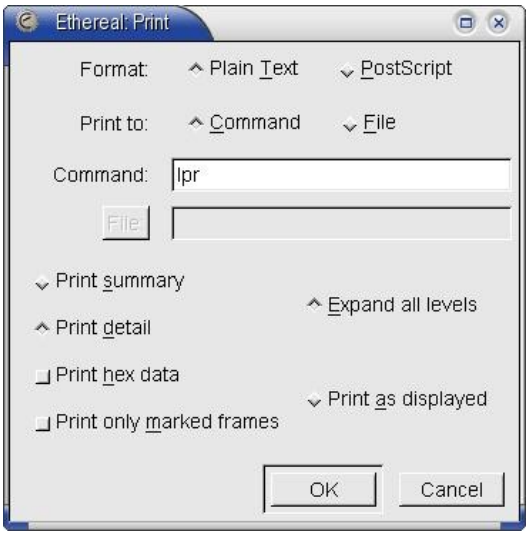


Рисунок 11.31 Диалоговое окно Print

11.9.3.2.2.4 Print Packet

Эта команда выводит на печать полностью раскрытое дерево протоколов для пакета, выбранного в списке. При печати используются опции, заданные на странице **Printing** (параграф 11.9.3.2.10.1 на стр. 301) диалогового окна **Preferences**.

Команду печати пакета можно также активизировать с помощью комбинации клавиш **Ctrl+P**.

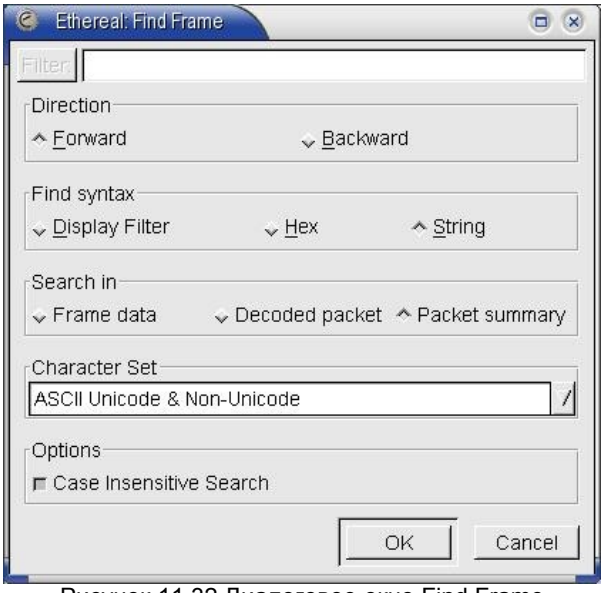


Рисунок 11.32 Диалоговое окно Find Frame

11.9.3.2.2.5 Quit

Завершает работу программы. Для завершения работы программы можно также использовать комбинацию клавиш **Ctrl+Q**.

11.9.3.2.3 Меню Edit

11.9.3.2.3.1 Find Frame

Эта команда (**Ctrl+F**) позволяет просматривать в прямом и обратном направлении список собранных пакетов на предмет поиска кадра, соответствующего заданному шаблону поиска. При активизации команды на экран выводится диалоговое окно **Find Frame** (см. рисунок 11.32), позволяющее задать

направления, критерий поиска и дополнительные опции для выбранного критерия.

Для поиска кадров можно использовать указанный фильтр отображения (см. параграф 11.9.3.4 на стр. 305) или контекст, заданный в шестнадцатеричном или символьном виде. При задании шаблона поиска в шестнадцатеричном формате разделителями шестнадцатеричных цифр могут служить пробел, двоеточие или дефис. При текстовом поиске вы можете задать кодировку (ASCII, Unicode или обе кодировки сразу) и опцию учета регистра символов.

Во всех режимах поиск начинается с выбранного в списке пакета. Если в данный момент указатель не установлен ни на один пакет в списке, в качестве начальной точки используется тот пакет, который был выбран последним.

11.9.3.2.3.2 Find Next

Продолжает поиск в направлении роста порядковых номеров (forward) в соответствии с заданными ранее критериями поиска. Для поиска следующего пакета можно также воспользоваться комбинацией клавиш **Ctrl+N** или кнопкой на панели инструментов программы Ethereal (см. параграф 11.9.3.2.9 на стр. 301).

11.9.3.2.3.3 Find Previous

Продолжает поиск в направлении уменьшения порядковых номеров (backward) в соответствии с заданными ранее критериями поиска. Найти предыдущий пакет можно также с помощью комбинации клавиш **Ctrl+B**.

11.9.3.2.3.4 Go To Frame

Обеспечивает переход к кадра, указанному номером в списке. Для перехода к интересующему кадру вы можете также использовать комбинацию клавиш **Ctrl+G** или кнопки на панели инструментов программы Ethereal (см. параграф 11.9.3.2.9 на стр. 301).

11.9.3.2.3.5 Субменю Time Reference

Команда (**Ctrl+T**) позволяет установить для указанного в списке пакета метку ***REF** или снять ранее установленную метку. Пакеты с такой меткой используются в качестве стартовых точек для отсчета временных интервалов, указываемых в списке пакетов. При наличии в списке нескольких пакетов с такой меткой для первого из них (минимальный номер в списке) отсчет от данного пакета происходит в обоих направлениях. Каждая последующая метка задает стартовую точку для отсчета временных интервалов только в направлении роста порядковых номеров.

Отметим, что пакеты с установленной меткой сохраняются в списке, независимо от заданных фильтров отображения (см. параграф 11.9.3.4Ошибка: источник перекрёстной ссылки не найден на стр. 305).

При наличии в списке пакетов нескольких меток для перемещения от одной метки к другой могут использоваться команды этого субменю **Find Next** и **Find Previous**.

11.9.3.2.3.6 Mark Frame

Команда (**Ctrl+M**) служит для выбора указанного в списке пакета или снятия ранее установленной метки выбора. При установке метки выбора в поле **frame.marked** помещается флаг выбора, который может впоследствии использоваться фильтрами отображения или командами поиска кадров.

11.9.3.2.3.7 Mark All Frames

Эта команда служит для выделения всех имеющихся в списке кадров.

11.9.3.2.3.8 Unmark All Frames

Эта команда позволяет снять разом все установленные ранее метки выделения кадров.

11.9.3.2.3.9 Preferences

Эта команда позволяет пользователю настроить параметры программы (печать, содержимое колонок списка пакетов, цветовое представление потоков TCP и другие опции). Диалоговое окно Preferences (см. рисунок 11.33) содержит множество страниц для настройки отдельных аспектов работы программы, подробно описанных ниже (параграф 11.9.3.2.10 на стр. 285).

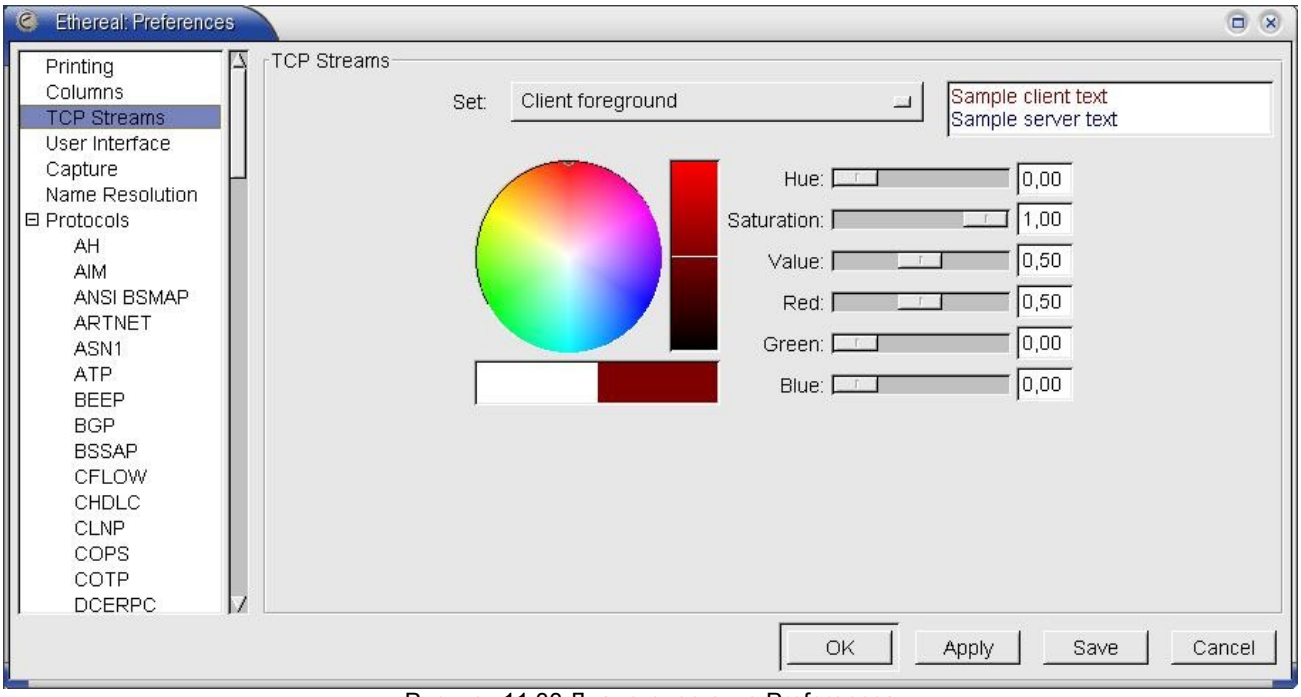


Рисунок 11.33 Диалоговое окно Preferences

11.9.3.2.3.10 Capture Filters

Эта команда позволяет создавать, редактировать и удалять фильтры захвата пакетов, используемые программой Ethereal. Синтаксис фильтров захвата полностью соответствует фильтрам программы tcpdump, описанным выше (параграф 11.9.2.2 на стр. 265).

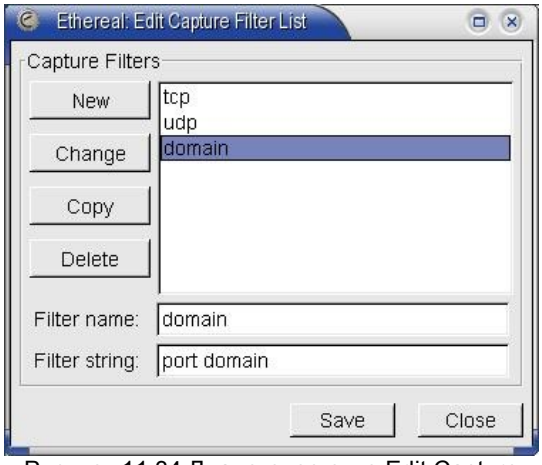


Рисунок 11.34 Диалоговое окно Edit Capture Filter List

При активизации команды на экран выводится диалоговое окно **Edit Capture Filter List** (см. рисунок 11.34) со списком существующих фильтров. Для создания нового фильтра укажите имя (поле **Filter name:**) и выражение (поле **Filter string:**) после чего нажмите кнопку **New**. При создании сложных фильтров большую помощью окажет функция копирования фильтров.

Для редактирования и выбора фильтров захвата вы можете также воспользоваться кнопкой на панели инструментов Ethereal (параграф 11.9.3.2.9 на стр. 301).

11.9.3.2.3.11 Display Filters

Данная команда позволяет управлять фильтрами Ethereal, используемыми программой для отбора пакетов, отображаемых в списке. Фильтры отображения применяются после фильтров захвата и позволяют выбрать из собранных программой пакетов только те, которые в данный момент представляют интерес. При сборе большого числа пакетов возможность выбора отображаемых пакетов существенно упрощает работу администратора по анализу трафика. Синтаксис фильтров отображения рассматривается в параграфе 11.9.3.4 (стр. 305).

Для задания фильтра отображения служит также поле выбора в нижней части главного окна Ethereal (см. рисунок 11.27).

Активизировать диалоговое окно можно также с помощью кнопки на панели инструментов Ethereal (параграф 11.9.3.2.9 на стр. 301).

11.9.3.2.3.11.1 Диалоговое окно Edit ... Filter List

Диалоговое окно **Edit ... Filter List** (см. рисунок 11.66) позволяет создавать, редактировать и удалять фильтры отображения, поиска и статистики. Синтаксис фильтров описан в параграфе 11.9.3.4 (стр. 305).

Окно содержит список существующих фильтров, поля для ввода имени фильтра и выражения, обеспечивающего фильтрацию, а также ряд кнопок, описанных в таблице 69.

Таблица 69 Кнопки диалогового окна выбора фильтров

Кнопка	Выполняемые действия
New	При заполненных полях **Filter name** и **Filter string** эта кнопка создает для нового фильтра соответствующую запись в списке.
Change	Эта кнопка служит для изменения указанного в фильтре списка в соответствии с новыми значениями полей ввода **Filter name** и **Filter string**.
Copy	Создает копию выбранного в списке фильтра.

Кнопка	Выполняемые действия
Delete	Удаляет указанный фильтр из списка.
Add Expression...	Кнопка открывает диалоговое окно **Filter Expression** (параграф 11.9.3.2.4 на стр. 287), позволяющее включать в фильтры логические выражения с использованием поддерживаемых синтаксисом языка описания фильтров примитивов и операций. Созданное в диалоговом окне **Filter Expression** выражение добавляется в поле **Filter string**.
OK	В диалоговых окнах **Display Filter**, **Read Filter** и **Search Filter** эта кнопка закрывает диалог, активизируя фильтр.
Apply	Вносит изменения в текущий фильтр отображения и применяет этот фильтр для списка пакетов.
Save	Сохраняет текущий список фильтров.
Close	Закрывает диалоговое окно без изменения фильтров.

11.9.3.2.4 Диалоговое окно Filter Expression

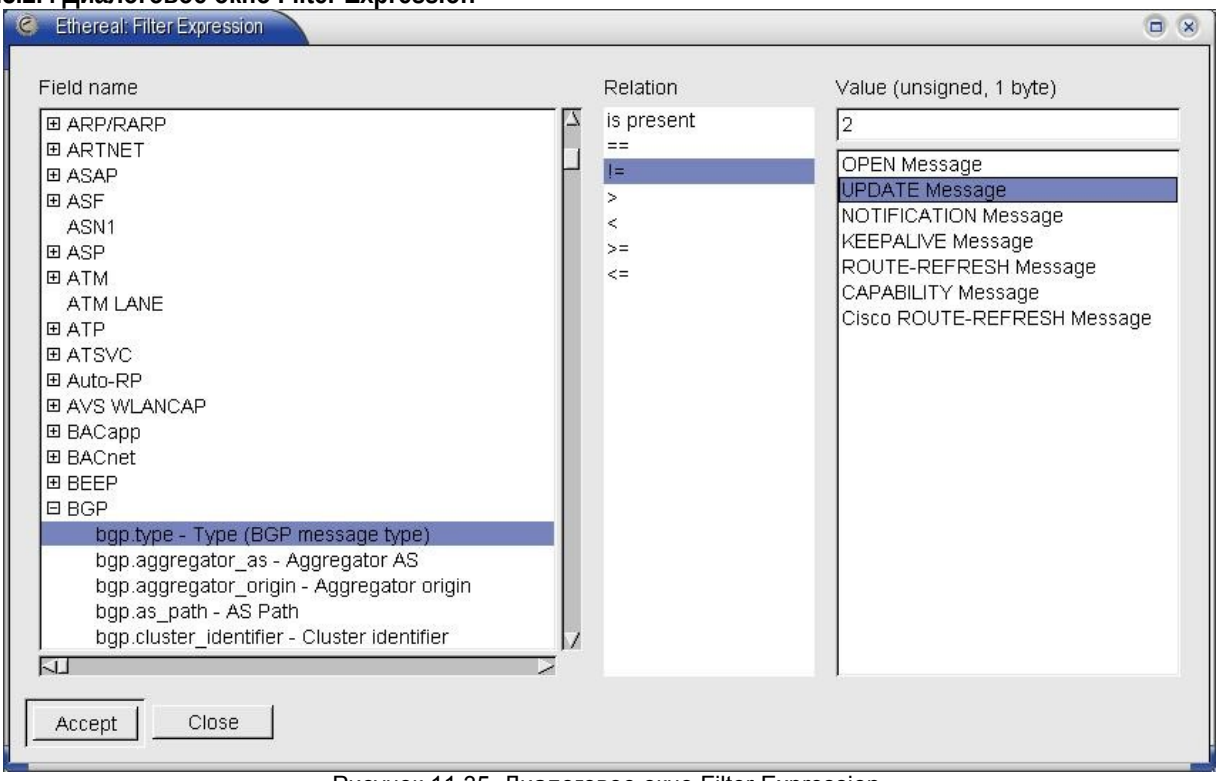


Рисунок 11.35. Диалоговое окно Filter Expression.

Диалоговое окно служит для добавления выражений в строку описания фильтра с использованием поддерживаемых программой примитивов фильтрации и логических операций. При нажатии кнопки **Accept** созданное выражение добавляется в спецификацию фильтра. Кнопка **Close** закрывает окно без сохранения созданного выражения.

11.9.3.2.4.1 Protocols

Эта команда используется для выбора протоколов, которые принимаются во внимание при сборе пакетов. Диалоговое окно **Enabled Protocols** (рисунок 11.36) позволяет указать протоколы, которые будут учитываться при анализе собранных пакетов. Кнопки **Enable All** и **Disable All** позволяют включить и отключить все протоколы. Кнопка **Invert** изменяет для каждого протокола в списке состояние на обратное (разрешенные протоколы запрещаются, ранее запрещенные - разрешаются).

Если тот или иной протокол указан в числе запрещенных, обнаружив этот протокол в принятом пакете, программа Ethereal переходит к следующему пакету. Все протоколы вышележащих уровней для пакетов запрещенного к анализу протокола также не будут анализироваться и отображаться в списке. Например, при запрете протокола TCP не будет отображаться информация для протоколов TCP, HTTP, SMTP, Telnet и любых других протоколов, передаваемых с помощью пакетов TCP.

Список выбранных протоколов можно сохранить и

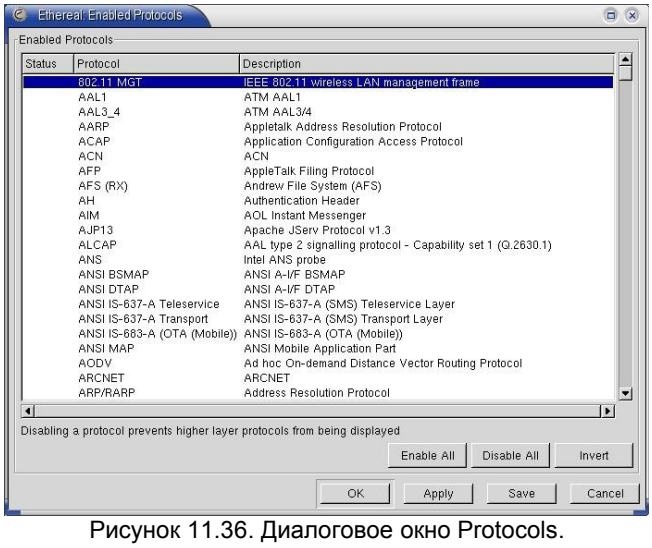


Рисунок 11.36. Диалоговое окно Protocols.

программа Ethereal при следующем запуске будет учитывать этот список.

11.9.3.2.5 Меню Capture

Меню Capture включает команды управления сбором пакетов.

11.9.3.2.5.1 Start

Команда **Start** выводит на экран диалоговое окно **Capture Options**, показанное на рисунке 11.37.

В процессе сбора все захваченные пакеты записываются во временный файл (местоположение этого файла указывает переменная окружения TMPDIR).

Для активизации процесса сбора пакетов можно также использовать комбинацию клавиш **Ctrl+K** или кнопку на панели инструментов Ethereal (параграф 11.9.3.2.9 на стр. 301).

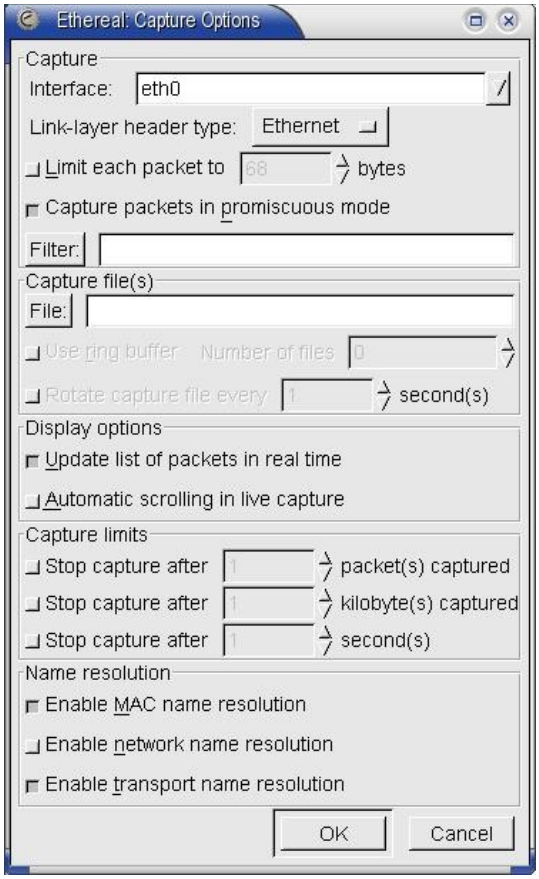


Рисунок 11.37 Диалоговое окно Capture Options

11.9.3.2.5.1.1 Диалоговое окно Capture Options

Диалог **Capture Options** позволяет задать опции сбора пакетов. При активизации окна в нем указаны принятые по умолчанию параметры, которые задаются на странице **Capture** (параграф 11.9.3.2.10.5 на стр. 304) диалогового окна **Preferences**.

Поле **Interface** служит для выбора интерфейса, с которого будут собираться пакеты. Отметим, что для систем Linux поддерживается фиктивных интерфейс **any** позволяющий собирать пакеты со всех активных интерфейсов системы..

Поле Link-layer header type для некоторых типов интерфейсов позволяет выбрать тип заголовков канального уровня при захвате пакетов. Например, некоторые ОС и версии libpcap позволяют для интерфейсов 802.11 выбирать тип заголовка канального уровня Ethernet или 802.11.

Поле выбора **Limit each packet to ... bytes** позволяет задать размер кадра захвата, используемый при сборе пакетов. Выбрав это поле следует задать размер кадра захвата в поле ввода текста. Если опция не выбрана, размер кадра захвата составляет 65535 байтов, что позволяет собирать все пакеты целиком.

Поле выбора **Capture packets in promiscuous mode** позволяет использовать при сборе пакетов режим захвата, при котором интерфейс прослушивает все передаваемые через среду пакеты, а не только кадры, адресованные данному интерфейсу.

Поле **Filter** позволяет указать фильтр, который будет применяться при сборе пакетов (см. параграф 11.9.3.2.3.10 на стр. 286).

Поле **File** служит для задания имени файла, в который будут записываться собранные программой пакеты. Если файл не указан, программа будет записывать пакеты во временный файл. По завершении сбора пакетов вы можете сохранить информацию в желаемом файле с помощью команды меню **File|Save As** (стр. 283).

Поле выбора **Use ring buffer** позволяет собирать пакеты в режиме “кольцевого буфера”. Поле **Number of files** позволяет указать число файлов захвата в кольцевом буфере. Для создания неограниченного числа файлов захвата используйте значение 0.

Поле выбора **Rotate capture file every ... second(s)** позволяет переходить к записи пакетов в новый файл кольцевого буфера по истечении заданного числа секунд, если ранее не был достигнут заданный размер файла захвата.

Поле **Update list of packets in real time** позволяет задать режим обновления списка пакетов непосредственно в процессе их сбора. Выбор этой опции активизирует поле выбора **Automatic scrolling in live**, которое позволяет включить автоматическую прокрутку списка собранных пакетов при которой в окне списка всегда выводится последний захваченный пакет.

Опция **Stop capture after ... packet(s)** позволяет ограничить процесс сбора захватом указанного числа пакетов.

Опция **Stop capture after ... kilobyte(s)** задает максимальный размер файла захвата. В режиме кольцевого буфера эта опция автоматически заменяется полем **Rotate capture file every ... kilobyte(s) -** после сбора заданного объема данных будет автоматически создаваться новый файл захвата. Отметим, что размер файла захвата задается в тысячах байтов, а не к килобайтах (1 кбайт = 1024 байт).

Поле выбора **Stop capture after ... second(s)** позволяет задать продолжительность сбора пакетов в секундах.

Опции **Enable MAC name resolution**, **Enable network name resolution** и **Enable transport name resolution** позволяют задать преобразование MAC-адресов, адресов IP, и номеров портов транспортного уровня в соответствующие имена.

11.9.3.2.5.2 Stop

Эта команда служит для остановки процесса сбора пакетов. Прервать процесс сбора пакетов можно также с помощью комбинации клавиш **Ctrl+E** или кнопки на панели инструментов Ethereal (параграф 11.9.3.2.9 на стр. 301).

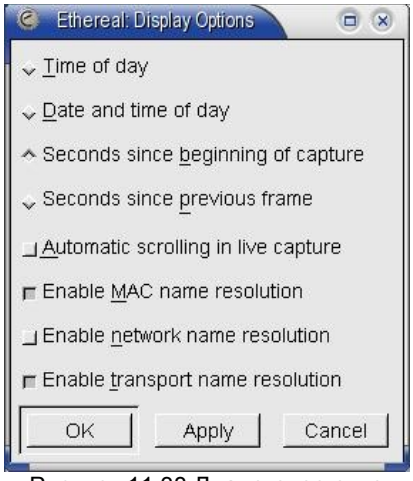


Рисунок 11.38 Диалоговое окно Display Options

11.9.3.2.6 Меню Display

11.9.3.2.6.1 Options

Команда **Options** активизирует диалоговое окно **Display Options**, позволяющее установить опции отображения собранных программой пакетов.

11.9.3.2.6.1.1 Диалоговое окно Display Options

В верхней части диалогового окна находятся 4 кнопки выбора режима отображения временных меток для собранных пакетов.

- **Time of day -** задает вывод временных меток как текущего времени суток.
- **Date and time of day** - временные метки выводятся в виде полной даты (год, число месяца и время суток).
- **Seconds since beginning of capture** - временные метки отсчитываются по числу секунд с момента начала сбора пакетов.
- **Seconds since previous frame** - временные метки задаются в виде интервала с момента прибытия предыдущего пакета.

Опция **Automatic scrolling in the live capture** позволяет задать режим автоматической прокрутки списка собранных пакетов. В этом режиме вы всегда будете видеть в списке последний захваченный пакет.

Кроме того, вы можете выбрать преобразования MAC-адресов, адресов IP и номеров портов транспортного уровня в соответствующие имена.

11.9.3.2.6.2 Match

Команда **Match** позволяет создать фильтр отображения или добавить к фильтру, указанному в нижней части окна программы, выражение на основе выбранной в дереве протоколов информации и применить этот фильтр для текущего отображения пакетов. Для создания фильтра отображения достаточно выбрать интересующее вас поле заголовке в дереве протоколов и указать операцию, которая будет служить для выбора отображаемых пакетов. Допускается последовательный выбор нескольких полей в одном или разных пакетах для создания сложных фильтров.

Selected - в списке остаются только те пакеты, которые имеют в указанном поле соответствующее выбранному пакету значение. Эта опция позволяет легко выделить из числа собранных все пакеты, идентичные указанному.

Not selected - в списке остаются только те пакеты, которые имеют в указанном поле не соответствующее выбранному пакету значение. Эта опция позволяет легко исключить из списка все пакеты, идентичные указанному.

And Selected - в списке остаются только пакеты, соответствующие условиям первого **и** второго выбора полей.

Or Selected - в списке остаются только пакеты, соответствующие условиям первого **или** второго выбора полей.

And Not Selected - в списке остаются только пакеты, соответствующие условиям первого **и** не соответствующие условиям второго выбора полей.

Or Not Selected в списке остаются только пакеты, соответствующие условиям первого **или** не соответствующие условиям второго выбора полей.

Две первые опции используются для задания первого выражения в фильтре отображения, остальные команды служат для добавления выражений в сложный фильтр.

Отметим, что для фильтров отображения используется абсолютное значение смещения полей, поэтому при наличии в заголовках полей переменной длины[1] фильтрация может не соответствовать вашим ожиданиям. Будьте аккуратны при создании фильтров отображения.

11.9.3.2.6.3 Prepare

Эта команда позволяет создать фильтр отображения, как описано в предыдущем параграфе, но не активизирует этот фильтр. Команда может быть весьма удобным инструментом для создания библиотеки фильтров изображения, поскольку при нажатии кнопки **Filter** активизируется диалоговое окно **Edit Display Filter List** (параграф 11.9.3.2.3.11.1 на стр. 286), позволяющее отредактировать и сохранить созданные фильтры.

11.9.3.2.6.4 Colorize Display - цветовая маркировка пакетов в списке

Команда служит для выделения пакетов в списке в соответствии с фильтрами отображения. Список выбранных фильтров отображения применяется последовательно к каждому из пакетов пока не будет обнаружено соответствие какому-либо из фильтров (на этом процесс проверки прекращается). Следовательно, сначала разумно проверять протоколы более высоких уровней, постепенно спускаясь к канальному уровню.

1 *Например, при работе с кадрами Token Ring, содержащими поля source-routed.*

11.9.3.2.6.4.1 Механизм цветового выделения

Строки списка выводятся с использованием различных цветов, определяемых списком фильтров цветовой маркировки. Цвет вывода для пакета определяется первым фильтром, которому этот пакет соответствует. Выражения для цветовых фильтров используют такой же синтаксис, какой применяется в фильтрах отображения (см. параграф 11.9.3.4 на стр. 305).

Программа Ethereal при запуске будет загружать фильтры цветовой маркировки из пользовательского файла (если он существует) или глобального файла фильтров маркировки. Если программа не найдет ни одного из этих файлов, цветовая маркировка пакетов не будет использоваться.

11.9.3.2.6.4.2 Диалоговое окно Apply Color Filters

Это диалоговое окно содержит список существующих фильтров цветовой маркировки и позволяет добавлять и удалять правила из списка, а также редактировать существующие правила и менять порядок их расположения в списке. Кроме списка фильтров, окно содержит ряд кнопок (таблица 70), служащих для выполнения операций над выделенными в списке фильтрами или всем набором фильтров. Внешний вид окна **Apply Color Filters** показан на рисунке 11.39, а отдельные элементы окна описаны ниже.

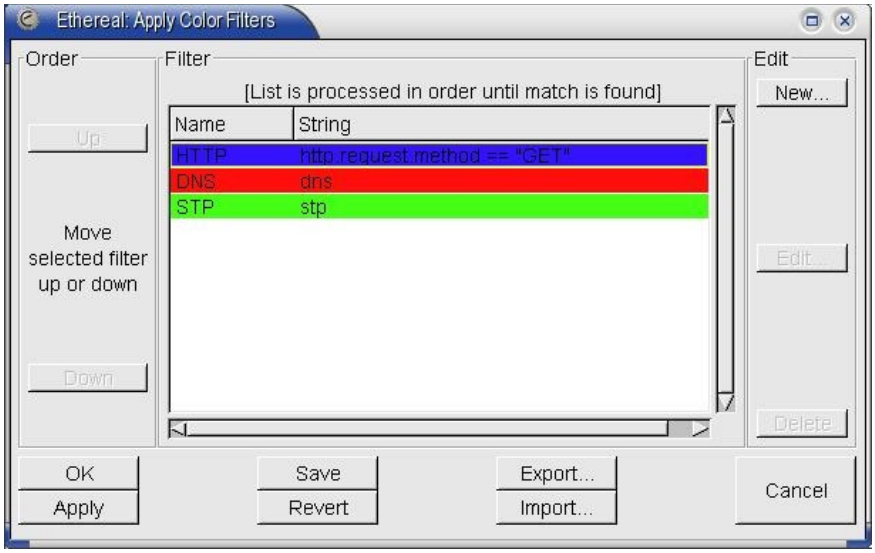


Рисунок 11.39. Диалоговое окно Apply Color Filters.

11.9.3.2.6.4.2.1 Список фильтров

Центральная часть диалогового окна содержит список существующих фильтров цветовой маркировки пакетов. Для выбора строки в списке фильтров достаточно щелкнуть на этой строке кнопкой мыши. Для выделения нескольких строк из списка используйте кнопку мыши при нажатой клавише **Ctrl** или **Shift**.

11.9.3.2.6.4.2.2 Кнопки управления фильтрами цветовой маркировки

Таблица 70 Кнопки диалогового окна Apply Color Filters

Кнопка	Действия
Up	Перемещает выбранные строки списка фильтров на одну позицию вверх[1].
Down	Перемещает выбранные строки списка фильтров на одну позицию вниз.
New	Добавляет новый фильтр в начало списка и активизирует диалоговое окно **Edit Color Filter** (см. параграф 11.9.3.2.6.4.2.3 на стр. 291). После создания фильтра вы можете переместить его в нужное место списка.
Edit	Кнопка редактирования фильтра открывает диалоговое окно **Edit Color Filter** (см. параграф 11.9.3.2.6.4.2.3 на стр. 291), позволяющее изменить указанный в списке фильтр. Если в списке выделено несколько фильтров, кнопка блокируется.
Delete	Удаляет из списка выделенные фильтры.
OK	Закрывает диалоговое окно, оставляя фильтры цветовой маркировки в текущем состоянии.
Apply	Активизирует фильтры из текущего списка, используя заданную этими фильтрами цветовую маркировку для существующего списка пакетов. Диалоговое окно при нажатии кнопки не закрывается.
Save	Сохраняет текущий список фильтров цветовой маркировки в персональном файле. Сохраненный список фильтров будет автоматически активизирован при следующем запуске программы Ethereal.
Revert	Удаляет файл с персональным списком фильтров, загружает глобальный фильтр цветовой маркировки (если такой фильтр имеется) и закрывает диалоговое окно.
Export	Позволяет сохранить текущий список фильтров цветовой маркировки в указанном файле. Вы можете сохранить весь список фильтров или только отмеченные в нем строки. Основным назначением этой кнопки является создание глобального списка фильтров цветовой маркировки (у вас должны быть для этого соответствующие полномочия).
Import	Позволяет включить фильтры из выбранного файла в начало текущего списка фильтров цветовой маркировки. После добавления фильтров из файла они находятся в состоянии выделенных, поэтому вы можете переместить весь набор добавленных из файла фильтров в нужную позицию списка.
Cancel	Закрывает диалоговое окно без изменения цветовой маркировки пакетов.

1 *Напомним, что цвет вывода для пакетов определяется первым фильтром, которому соответствует этот пакет. Поэтому порядок следования фильтров в списке имеет важное значение.*

11.9.3.2.6.4.2.3 Диалоговое окно Edit Color Filter

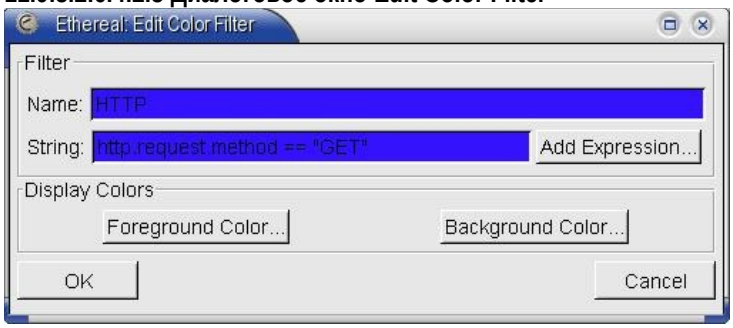


Рисунок 11.40 Диалоговое окно Edit Color Filter

Диалоговое окно редактирования фильтров цветовой маркировки позволяет изменить параметры фильтра, указанного в списке диалогового окна **Apply Color Filters**. Окно редактирования содержит строку с именем фильтра и выражение, используемое для фильтрации. Вы можете задать условия фильтрации вручную или создать фильтр с помощью диалога Filter Expression (параграф 11.9.3.2.4 на стр. 287). Для выбора цветов фона и переднего плана при выводе пакетов, соответствующих фильтру, служат кнопки **Backgroung Color ...** и **Foregroung Color ...**. При нажатии на эти кнопки на экран выводится диалоговое окно выбора цвета, показанное на рисунке 11.41.

11.9.3.2.6.5 Collapse All

Сворачивает все развернутые ветви дерева протоколов.

11.9.3.2.6.6 Expand All

Разворачивает все свернутые ветви дерева протоколов.

11.9.3.2.6.7 Show Packet In New Window

Эта команда создает новое окно, содержащее панели дерева протоколов и дампа для выбранного в списке пакета (рисунок 11.42). Это окно будет сохраняться на экране в неизменном виде даже при выборе списке другого пакета. Таким образом вы можете просматривать содержимое нескольких пакетов одновременно.

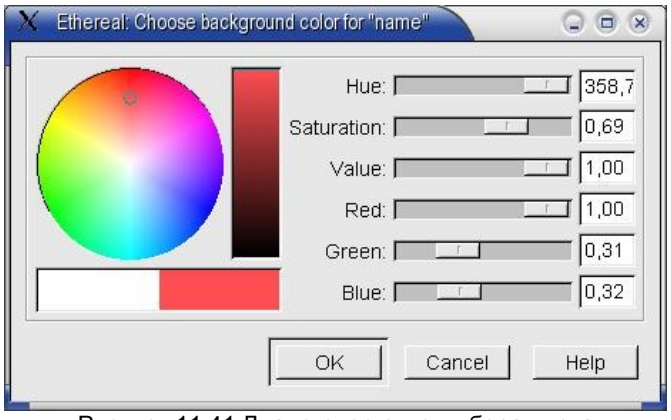


Рисунок 11.41 Диалоговое окно выбора цвета

11.9.3.2.6.8 User Specified Decodes

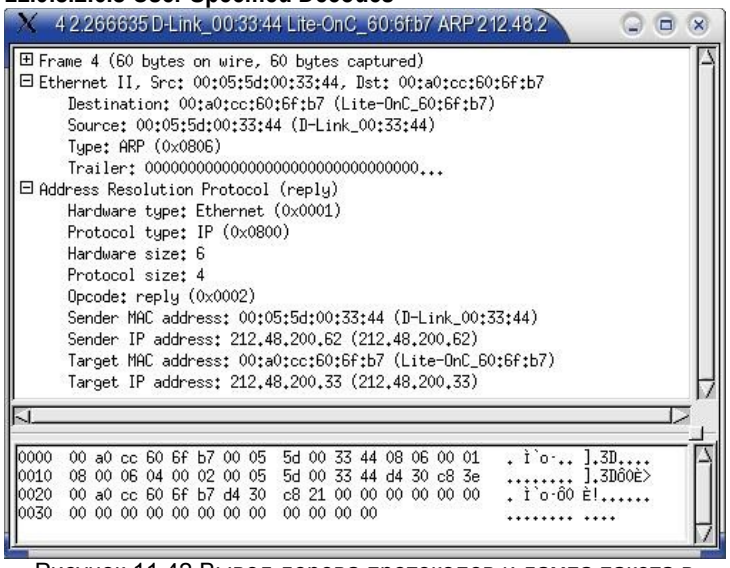


Рисунок 11.42 Вывод дерева протоколов и дампа пакета в новом окне

Эта команда выводит на экран диалоговое окно **Decode As Show** со списком заданных пользователем изменений схемы декодирования (см. параграф 11.9.3.2.7.3 на стр. 292). Кроме списка заданных пользователем изменений окно содержит кнопку **OK** для закрытия окна и кнопку **Reset Changes** для отказа от всех внесенных ранее изменений схемы декодирования. Вид диалогового окна показан на рисунке 11.44.

11.9.3.2.7 Меню Tools

11.9.3.2.7.1 Plugins

Команда **Plugins** выводит на экран одноименное диалоговое окно (рисунок 11.43) со списком доступных plugin-модулей.

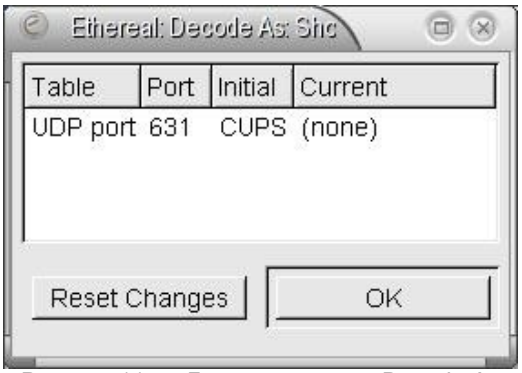


Рисунок 11.44 Диалоговое окно Decode As Show

Список модулей показывает имя и номер версии всех найденных в системе подключаемых модулей Ethereal. Поиск модулей выполняется в каталоге **lib/ethereal/plugins/$VERSION** в основном каталоге программы ($VERSION - номер версии программы). Отметим, что каждый модуль может поддерживать не один протокол, поэтому не пытайтесь найти модель для каждого протокола. Анализ поддерживаемых подключаемыми модулями протоколов можно включать и отключать с помощью команды меню **Edit:Protocols** (параграф 11.9.3.2.4.1 на стр. 287).

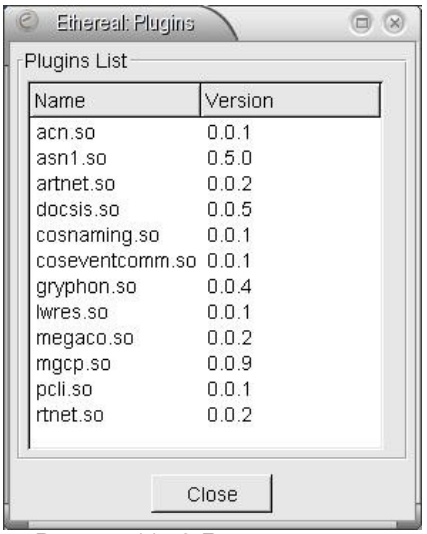


Рисунок 11.43 Диалоговое окно Plugins

11.9.3.2.7.2 Follow TCP Stream

При выборе в списке пакета TCP или использующего этот протокол пакета вышележащего уровня, команда **Follow TCP Stream** открывает новое окно (см. рисунок 11.45), содержащее информацию из потока данных соединения TCP, к которому относится указанный в списке пакет. При этом в списке пакетов основного окна Ethereal автоматически активизируется фильтр отображения, показывающий только пакеты, относящиеся к данному соединению TCP. Для отключения этого фильтра можно использовать кнопку **Reset** в нижней части окна программы (справа от поля **Filter**).

Диалоговое окно **Follow TCP stream** позволяет просматривать весь поток данных для выбранного соединения TCP или потоки в отдельных направлениях и обеспечивает вывод информации в формате ASCII, EBCDIC, Hex Dump (дамп отдельных пакетов в текстовом и шестнадцатеричном формате) или C Arrays (массив шестнадцатеричных значений в формате языка C).

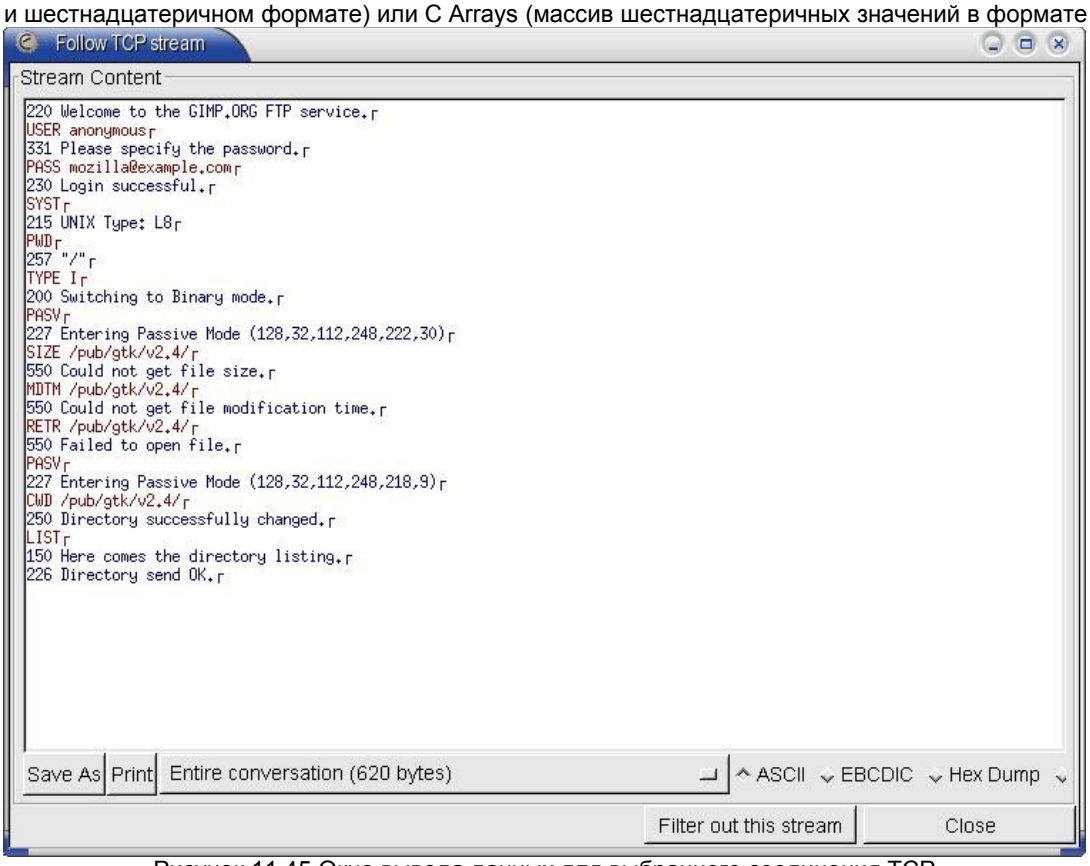


Рисунок 11.45 Окно вывода данных для выбранного соединения TCP

Содержимое окна можно напечатать с использованием опций команды меню **File:Print Packet** (параграф 11.9.3.2.2.4 на стр. 284) или сохранить в текстовом файле. Цвета вывода в окне можно изменить с помощью страницы **TCP Streams** (параграф 11.9.3.2.10.3 на стр. 303) диалогового окна **Preferences**.

11.9.3.2.7.3 Decode As

Команда **Decode As** позволяет для выбранного в списке пакета задать тип декодирования, отличный от принятого по умолчанию. Например, вы можете попытаться декодировать пакеты пакеты TCP, адресованные в порт 10000, который используется программой Webmin (параграф 11.3 на стр. 228), как пакеты HTTP[1] и видеть дерево протоколов в соответствии с выбранной трактовкой пакетов. Можно с помощью этой команды выполнить и обратную операцию - отказаться от принятого по умолчанию декодирования для указанного в списке кадра.

1 *Каковыми они обычно и являются.*

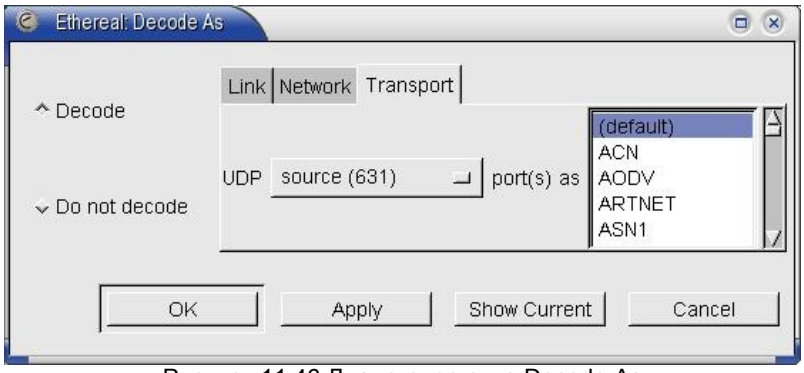


Рисунок 11.46 Диалоговое окно Decode As

Активизируемое по этой диалоговое окно команде Decode As включает панель выбора для каждого из уровней, которые поддерживаются для выбранного в списке кадра (канальный, сетевой и транспортный), позволяя задать отображение при декодировании независимо для каждого уровня. Поля выбора **Decode**/**Do not decode** задают режим смены типа декодирования или отказа от принятого типа декодирования, соответственно.

Кнопка **OK** закрывает диалоговое окно и выполняет заданные в нем операции, **Apply** выполняет заданные операции, не закрывая окна, а **Cancel** закрывает окно без изменения существующей схемы декодирования. Кнопка **Show Current** выводит на экран список заданных пользователем изменений схемы декодирования протоколов (см. параграф 11.9.3.2.6.8 на стр. 291).

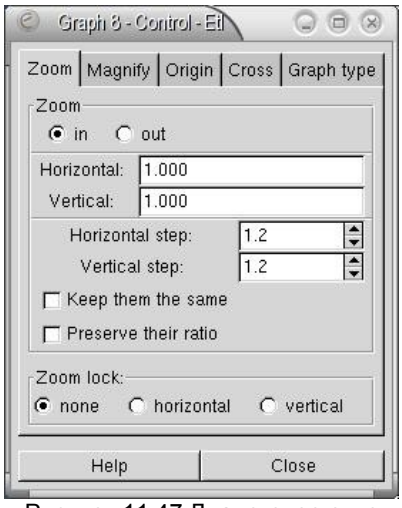


Рисунок 11.47 Диалоговое окно Graph Control

11.9.3.2.7.4 Go To Corresponding Frame

Если выбранное поле в панели дерева протоколов содержит номер кадра, данная команда обеспечивает переход к соответствующему кадру в списке. Такая возможность обеспечивается только в тех случаях, когда модуль анализа (dissector), который поместил запись в дерево протоколов, включил туда эту запись как фильтруемое поле, а не просто текст.

Обеспечиваемая этой командой возможность перехода может быть полезна для переходов между запросами и откликами, если номер кадра помещается в дерево протоколов.

11.9.3.2.7.5 TCP Stream Analysis

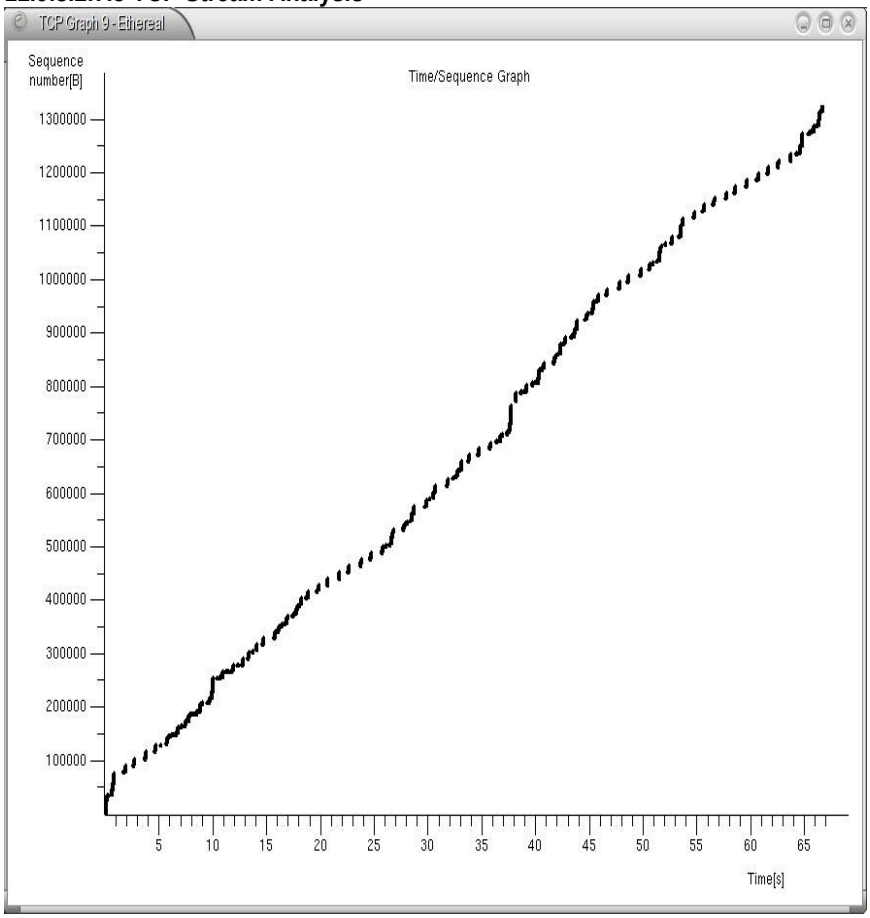


Рисунок 11.48 График роста порядковых номеров TCP (формат Stevens)

Это субменю обеспечивает группу команд анализа потоков TCP. Управление выводом информации обеспечивается с помощью диалогового окна **Graph Control** (рисунок 11.47), выводимого при активизации любой из перечисленных здесь команд. Диалоговое окно позволяет задавать параметры отображения информации и выбирать тип отображения.

11.9.3.2.7.5.1 Time-sequence Graph (Stevens)

Эта команда позволяет увидеть на графике зависимость роста порядковых номеров пакетов TCP от времени, как показано на рисунке 11.48. Для перехода к просмотру другого типа графика можно использовать страницу **Graph Type** диалогового окна **Graph Control** или соответствующую команду субменю **TCP Stream Analysis**.

11.9.3.2.7.5.2 Time-sequence Graph (tcptrace)

Эта команда позволяет увидеть на графике зависимость роста порядковых номеров пакетов TCP от времени в формате **tcptrace**, как показано на рисунке 11.49. Для перехода к просмотру другого типа графика можно использовать страницу **Graph Type** диалогового окна **Graph Control** или соответствующую команду субменю **TCP Stream Analysis**.

11.9.3.2.7.5.3 Throughput Graph

Эта команда позволяет увидеть на графике зависимость потока данных через соединение TCP от времени, как показано на рисунке 11.50. Для перехода к просмотру другого типа графика можно использовать страницу **Graph Type** диалогового окна **Graph Control** или соответствующую команду субменю **TCP Stream Analysis**.

11.9.3.2.7.5.4 RTT Graph

Эта команда позволяет увидеть на графике временную зависимость RTT (время кругового обхода) для соединения TCP, как показано на рисунке 11.51. Для перехода к просмотру другого типа графика можно использовать страницу **Graph Type** диалогового окна **Graph Control** или соответствующую команду субменю **TCP Stream Analysis**.

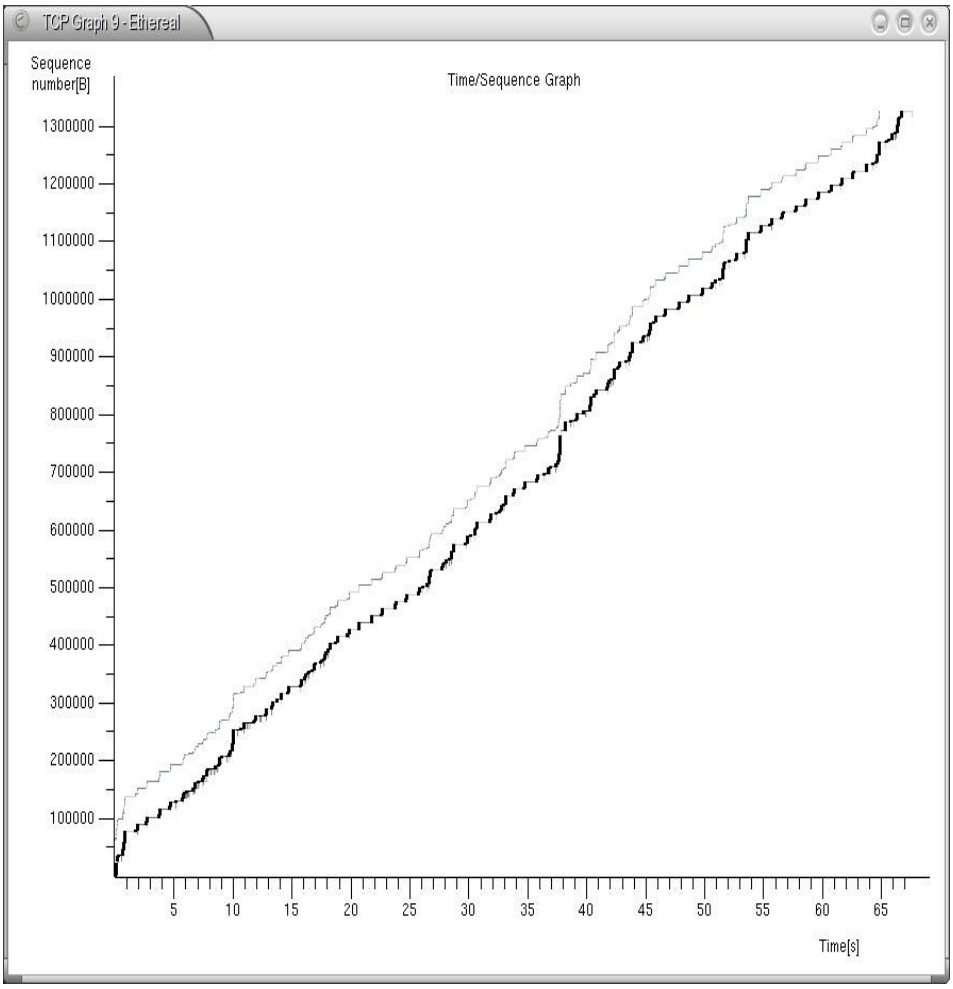


Рисунок 11.49 График роста порядковых номеров TCP (формат tcptrace)

11.9.3.2.7.6 Summary

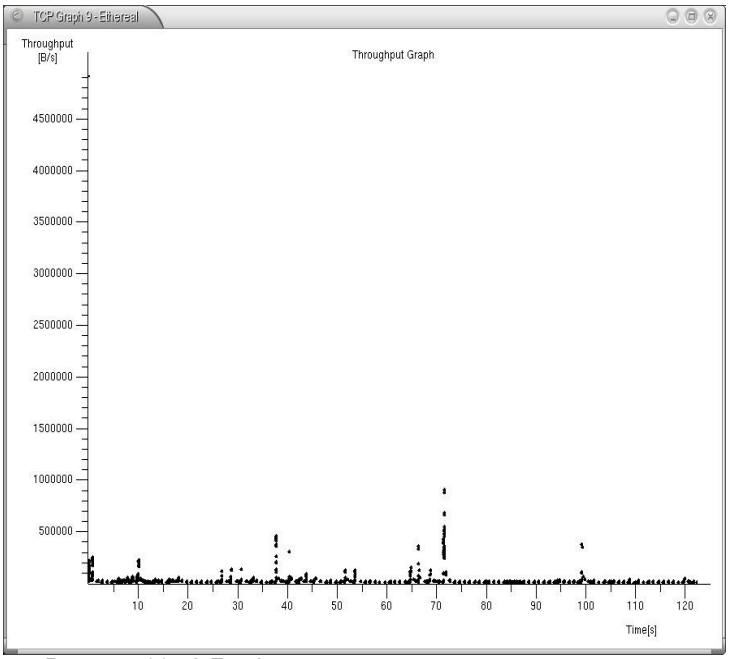
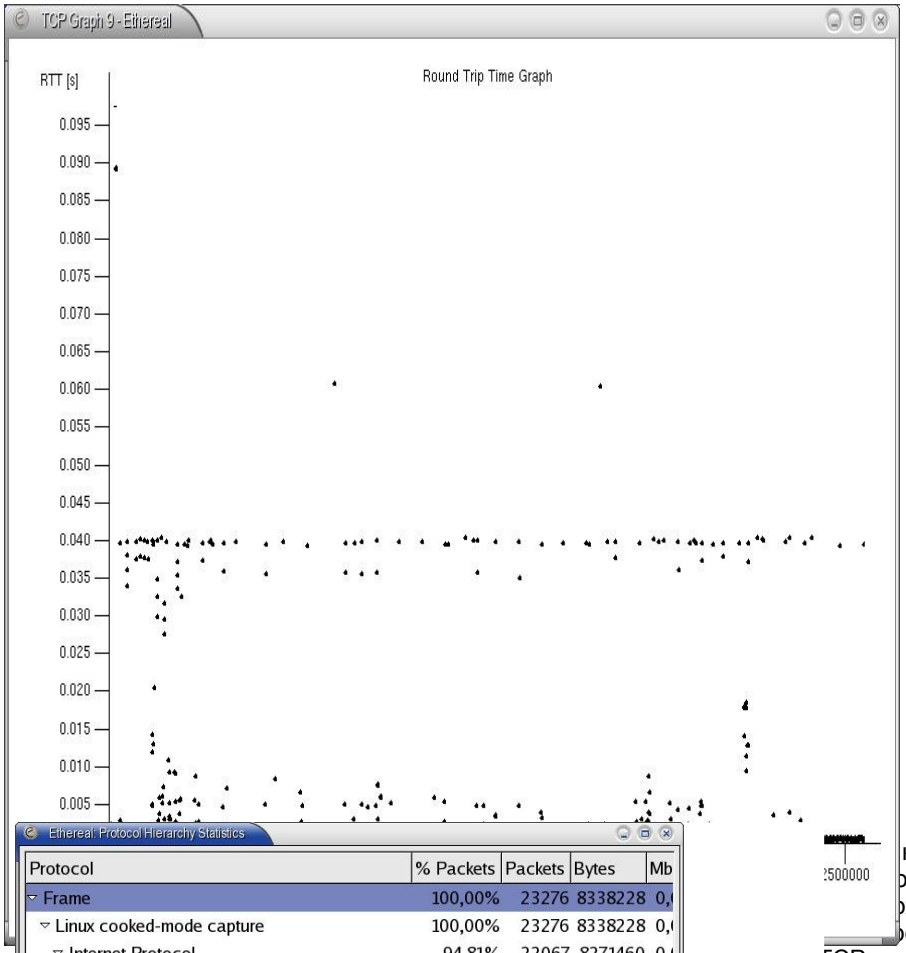


Рисунок 11.50 График зависимости потока данных через соединение TCP

Эта команда обеспечивает вывод на экран одноименного диалогового окна (рисунок 11.52), содержащего сведения общего характера о текущем или последнем завершенном сеансе сбора пакетов (имя файла захвата и его формат, размер кадра захвата, продолжительность сбора пакетов, их число, статистика использования фильтров и т. п.).



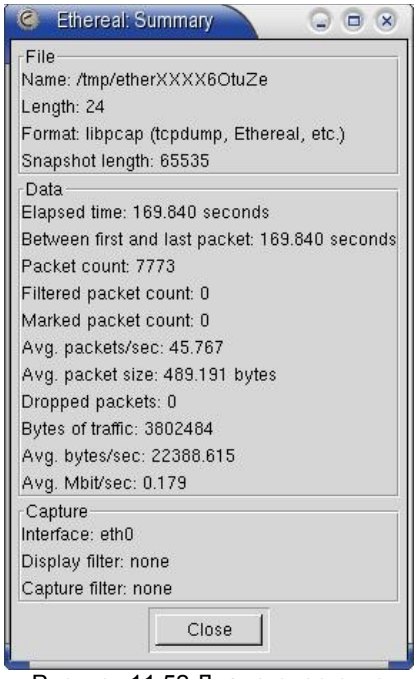


Рисунок 11.52 Диалоговое окно Summary

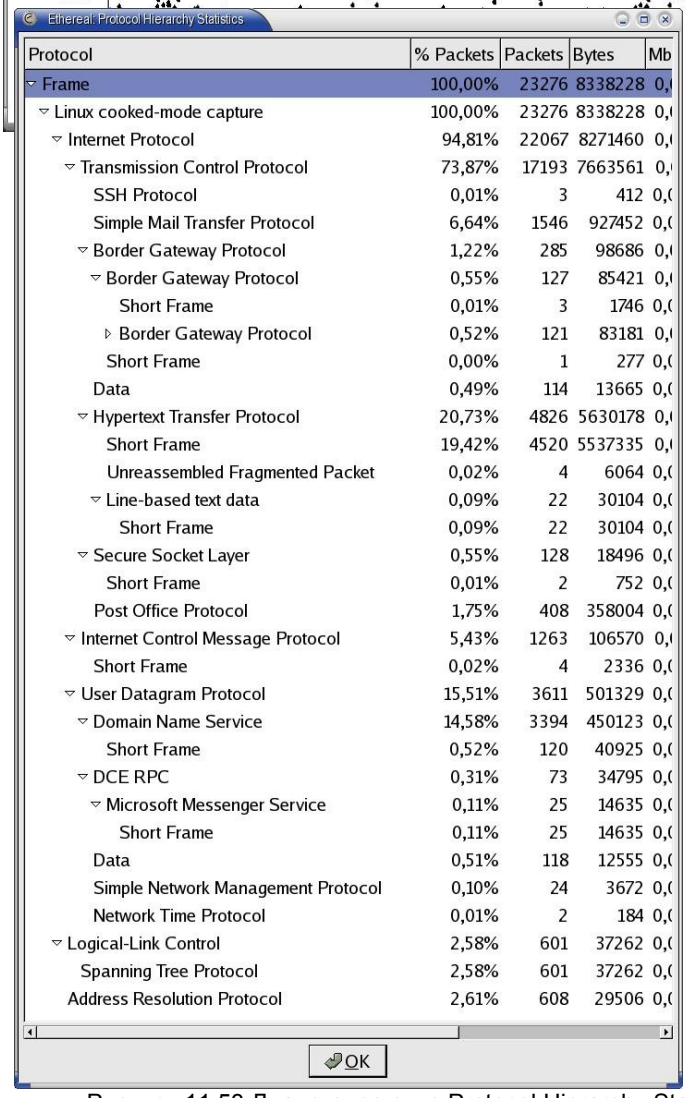


Рисунок 11.53 Диалоговое окно Protocol Hierarchy Statistics

команда активизирует одноименное
говое окно (рисунок 11.53),
жащее статистические сведения о
еделении пакетов по протоколам за
TCP я сбора кадров. Для каждого
.ро.окола в иерархии указывается число пакетов и байтов, а также процент от общего трафика. Кроме того, в отдельной колонке учитываются количество и суммарный размер пакетов, для которых соответствующий протокол явился последним в стеке (т. е., не содержал в себе пакетов других протоколов). Эти счетчики выводятся в колонках **End Packets** и **End Bytes**.

11.9.3.2.7.8 Statistics

Субменю Statistics включает команды просмотра статистической информации, описанные ниже.

Рисунок 11.54 Диалоговое окно генерации статистики

11.9.3.2.7.8.1 Watch protocol

Субменю Watch protocol позволяет просматривать статистику для протоколов BOOTP-DHCP, ITU-T H.225, HTTP и WAP-WSP. При активизации любой из команд этого субменю на экране появится диалоговое окно выбора фильтра и генерации статистики, показанное на рисунке 11.54. Вы можете использовать фильтр для учета соответствующего ему трафика или проста нажать кнопку **Create Stat** для генерации статистики по всем собранным пакетам интересующего протокола. Кнопка **Filter** позволяет выбрать из числа готовых или создать заново фильтр (см. параграф 11.9.3.2.3.11 на стр. 286), который будет использоваться для отбора пакетов, принимаемых во внимание при расчете статистики. Выражение для фильтрации пакетов можно задать непосредственно в поле ввода справа от кнопки **Filter**.

11.9.3.2.7.8.1.1 BOOTP-DHCP

Диалоговое окно BOOTP-DHCP содержит статистику использования протоколов удаленной конфигурации хостов, включающую сведения о полученных запросах и откликах серверов DHCP-BOOTP, как показано на рисунке 11.55.

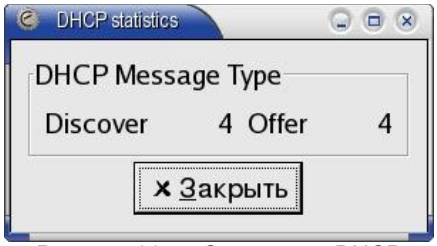


Рисунок 11.55 Статистика DHCP

11.9.3.2.7.8.1.2 ITU-T H.225

225

Диалоговое окно **ITU-T H.225 Message and Message Reason Counter** содержит статистику обмена сообщениями H.225. Форма диалогового окна показана на рисунке 11.56. Первая колонка списка содержит список сообщений H.225 и вызвавших их причин, а во второй указано количество сообщений в текущем файле захвата. Содержимое окна динамически обновляется в процессе сбора пакетов или загрузки новых файлов данных в программу Ethereal.

Вы можете использовать фильтр для отбора интересующих вас сообщений, указав его в диалоговом окне генерации статистики (см. рисунок 11.54).

11.9.3.2.7.8.1.3 HTTP

Рисунок 11.57 Статистика HTTP

Диалоговое окно статистики HTTP содержит сведения о количестве запросов и откликов HTTP в собранных программой пакетах. Содержимое окна статистики динамически обновляется при добавлении новых пакетов или загрузке новых файлов захвата. Форма окна статистики HTTP показана на рисунке 11.57.

Вы можете использовать фильтр для отбора интересующих вас сообщений, указав его в диалоговом окне генерации статистики (см. рисунок 11.54).

11.9.3.2.7.8.1.4 WAP-WSP

Окно статистики WAP-WSP (рисунок 11.58) включает сведения о количестве пакетов различных типов и данные о состоянии.

При подготовке статистического отчета можно задать фильтр, который позволит просматривать сведения только для интересующего вас трафика. Фильтр указывается в диалоговом окне генерации статистики (см. рисунок 11.54).

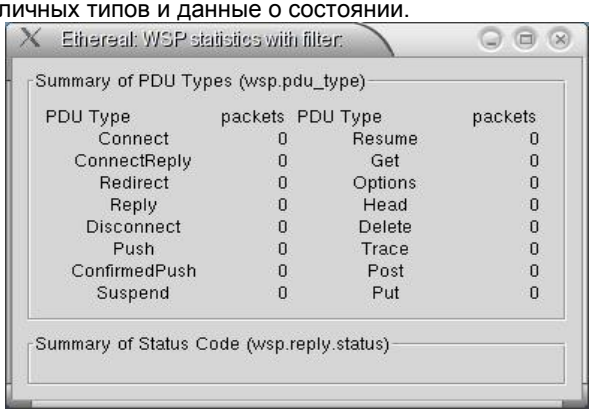


Рисунок 11.58 Статистика WAP-WSP

11.9.3.2.7.8.2 Service Response Time

Это субменю позволяет получить статистические сведения о времени отклика различных сетевых приложений и служб. Для выбора программы и номера версии, а также задания фильтра служит диалоговое окно **Compute ... SRT Statistics**, показанное на рисунке 11.59. Окно включает кнопки выбора программы, для которой генерируется статистика и номера версии. Кнопка **Filter** позволяет выбрать из числа готовых или создать заново фильтр (см. параграф 11.9.3.2.3.11 на стр. 286), который будет использоваться для отбора пакетов, принимаемых во внимание при расчете статистики. Выражение для фильтрации пакетов можно задать непосредственно в поле ввода справа от кнопки **Filter**.

11.9.3.2.7.8.2.1 DCE-RPC

Команда открывает диалоговое окно **Compute DCE-RPC SRT Statistics** (рисунок 11.59) для выбора программы и номера версии, а также задания фильтра (если вы хотите собрать статистику для части трафика). После выбора опций генерации отчета нажмите кнопку **Create Stat** для генерации статистического отчета. Диалоговое окно статистики DCE-RCP[1] (рисунок 11.60) включает информацию о количестве вызовов процедур выбранной программы и времени отклика для каждой процедуры (минимальное, максимальное и среднее). Содержимое окна будет

1 *Distributed Computing Environment - среда распределенных вычислений, Remote Procedure Call - удаленный вызов процедур.*

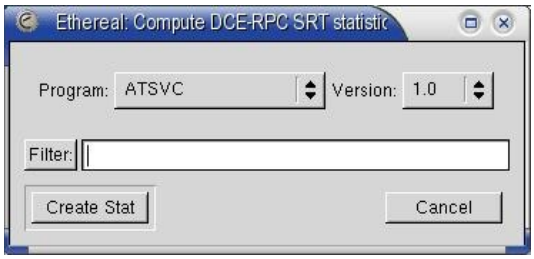
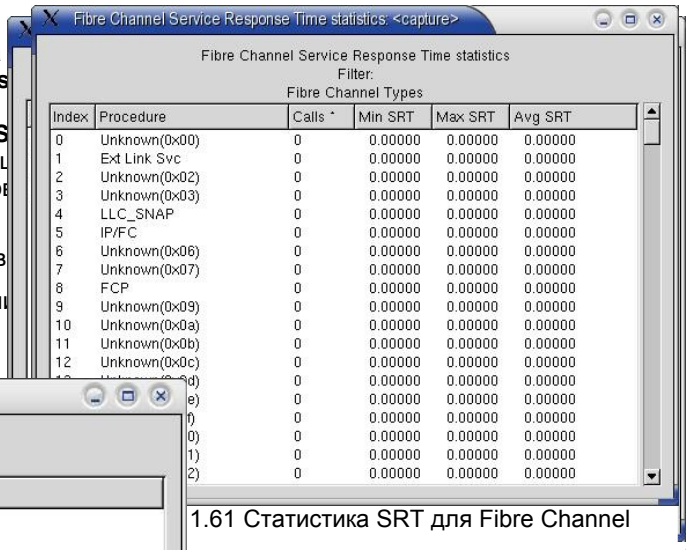


Рисунок 11.59 Диалоговое окно генерации статистики SRT

автоматически обновляться, если процесс сбора пакетов продолжается.

11.9.3.2.7.8.2.2 Fibre Channel

Команда **Statistics** собрать **Create S** содержащ



максимальное и среднее время отклика для всех типов мере получения новых пакетов.

Время отклика рассчитывается как интервал между перв

Если при расчете статистики фильтр не был задан, прин

1.61 Статистика SRT для Fibre Channel

11.9.3.2.7.8.2.3 MGCP

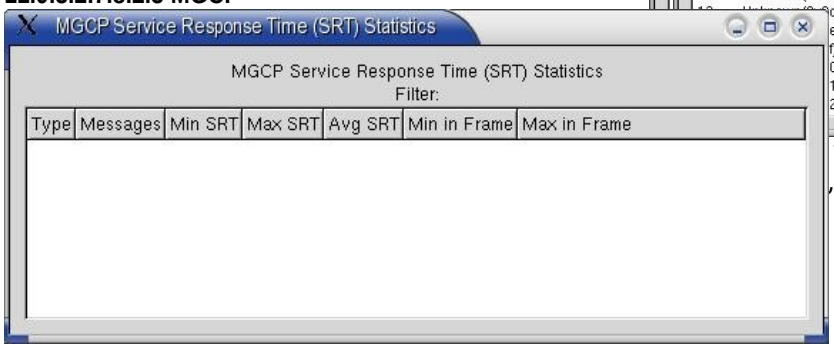


Рисунок 11.62 Статистика SRT для MGCP

Статистика SRT для программ DCE-RPC создается статистический отчет (см. рисунок 11.62), содержащий в каждой строке тип, сообщение, количество вызовов и время отклика сервиса (минимальное, максимальное и среднее). Статистика выводится для всех известных типов MGCP. Выводимые в отчете значения автоматически обновляются по мере получения новых пакетов MGCP.

11.9.3.2.7.8.2.4 ONC-RPC

Команда обеспечивает генерацию статистики вызовов ONC-RPC[2] для выбранной в диалоговом окне **Compute ONC-RPC Statistics** (рисунок 11.59) программы и номера версии. Если вы хотите получить статистику для части трафика, можно использовать фильтр. После нажатия кнопки **Create Stat** создается статистический отчет и на экран выводится диалоговое окно (рисунок 11.6311.63), содержащее имена процедур, количество вызовов, минимальное, максимальное и среднее время отклика для всех процедур выбранной версии программы. Если процесс сбора пакетов продолжается, статистические данные в диалоговом окне будут автоматически обновляться.

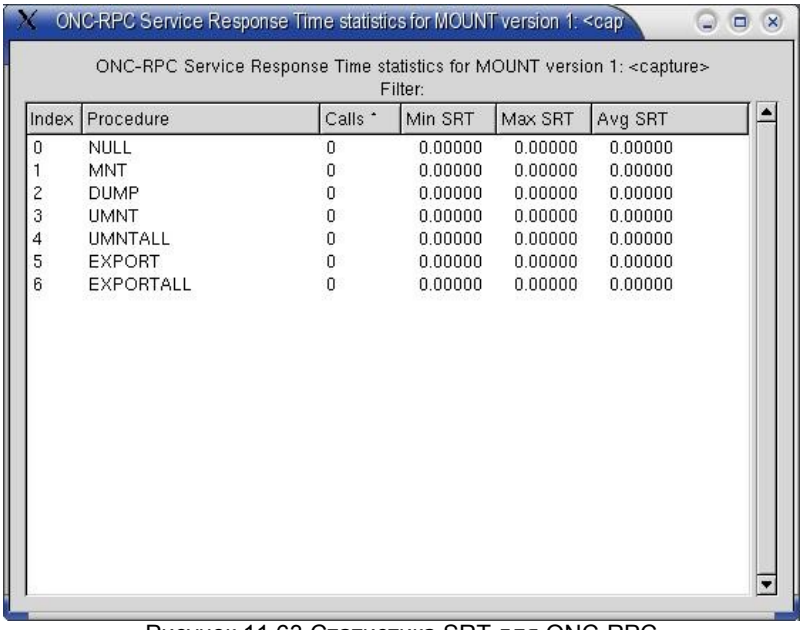


Рисунок 11.63 Статистика SRT для ONC-RPC

11.9.3.2.7.8.2.5 SMB

Эта команда обеспечивает генерацию и просмотр статистики SRT для трафика SMB. При вызове команды на экране появляется диалоговое окно **Compute ONC-RPC Statistics** (рисунок 11.59), позволяющее задать фильтр при генерации статистического отчета. После нажатия на кнопку **Create Stat** создается отчет и выводится диалоговое окно (см. рисунок 11.64), содержащее список всех команд SMB с числом вызовов и временем отклика (минимальное, максимальное и среднее) для каждой команды.

1 *Media Gateway Control Protocol - протокол управления шлюзом сред*
2 *Open Network Computing Remote Procedure Call*

Отчет представляется в форме трех списков, содержащих обычные команды SMB, команды Transaction2 и команды NT Transaction. При расчете статистики используется только первая команда цепочек **xAndX** (т.е., для цепочки **SessionSetupAndX + TreeConnectAndX**)при подготовке статистики будет учитываться только команда **SessionSetupAndX**).

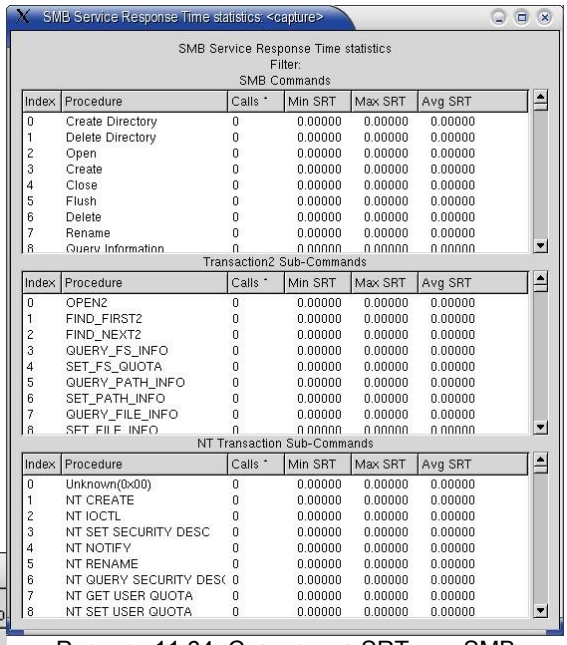


Рисунок 11.64 Статистика SRT для SMB

По умолчанию строки списка сортируются в порядке убывания числа кадров, но вы можете поменять порядок сортировки, щелкнув кнопкой мыши на заголовке соответствующей колонки. Повторный щелчок по тому же заголовку изменит порядок сортировки на обратный.

Статистика обеспечивается для протоколов:

- Ethernet
- Fibre Channel
- FDDI
- IPv4
- IPX
- TCP (IPv4/v6)
- Token Ring
- UDP (Ipv4/v6)

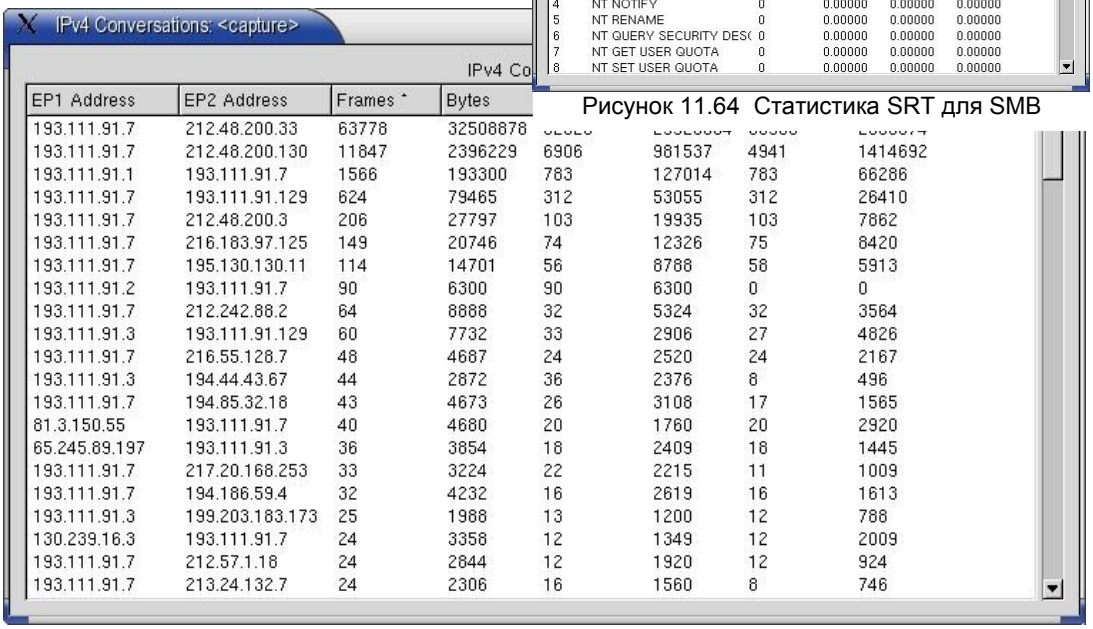


Рисунок 11.65 Статистика трафика между парами хостов IP

11.9.3.2.7.8.3 Conversation List

Это субменю позволяет просматривать обмен кадрами между парами конечных точек. Список содержит одну строку для каждого уникального “разговора” - в этой строке указываются адреса узлов, общее количество байтов и пакетов для этого “разговора”, а также количества пакетов и байтов, переданные в каждом направлении. На рисунке 11.65 показан пример такого списка для протокола IPv4.

11.9.3.2.7.8.3.1 IO-Stat

Команда IO-Stat открывает одноименное диалоговое окно (рисунок 11.66), содержащее до 5 графиков, выведенных различными цветами и показывающих число пакетов или байтов в секунду для кадров, соответствующих каждому из пяти поддерживаемых фильтров. По умолчанию выводится один график, показывающий количество кадров, собранных программой в секунду.

Верхняя часть окна содержит графики сбора пакетов. При продолжительном сборе данных график перестает помещаться в окне и для возможности его просмотра по частям выводится горизонтальное поле прокрутки. По горизонтальной оси графика откладывается время, а по вертикальной - количественная характеристика скорости сбора пакетов, соответствующих фильтру.

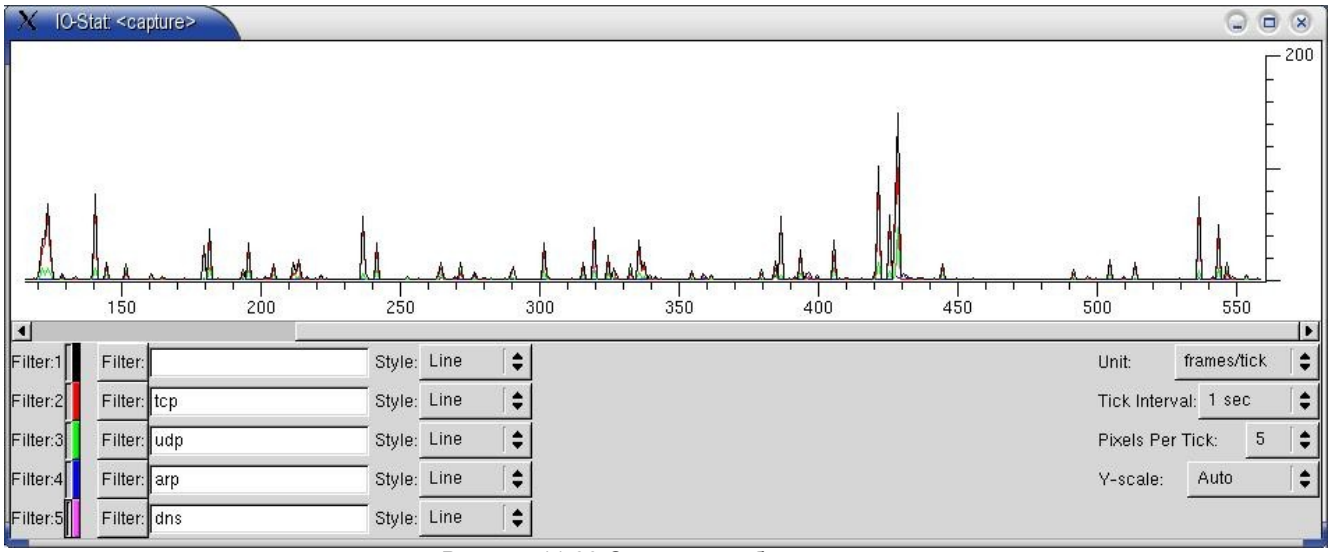


Рисунок 11.66 Статистика сбора кадров

Под графиком размещены элементы управления сбором и выводом статистики. В левой части окна расположены 5 строк, каждая из которых содержит однотипный набор полей:

Элемент	Описание
Номер фильтра	Filter1, Filter2, Filter3, Filter4, Filter5
Кнопка активизации	Нажатие кнопки мыши на небольшое поле между номером фильтра и цветовым маркером включает или отключает вывод графика для пакетов, соответствующих данному фильтру.
Цвет графика	Показывает предопределенный цвет[1] вывода графика для данного фильтра.
Кнопка выбора фильтра	Активизирует окно выбора фильтров (см. параграф 11.9.3.2.3.11 на стр. 286).
Фильтр	Выражение, используемое для фильтрации при сборе статистики.
Стиль графика	Задает линейный, импульсный или столбчатый график для данного фильтра.

Если поле имени фильтра пусто, соответствующий график будет строиться с учетом всех собранных пакетов (без фильтрации), в противном случае будут приниматься во внимание только пакеты, соответствующие выбранному фильтру.

В правой части окна находятся общие для всех графиков элементы управления выводом.

Элемент	Описание
Unit	Задает единицы измерения по вертикальной оси (пакеты или байты). Вы можете также выбрать для этого поля значение **advanced...** (см. ниже)
Tick Interval	Задает гранулярность отсчета времени для построения графиков (10 мсек, 100 мсек, 1 сек или 10 сек).
Pixels per Tick	Задает размер временного интервала на графике в пикселях (1, 2, 5 или 10).
Y-scale	Задает масштаб вертикальной оси. Вы можете выбрать одно из приведенных в списке значений или задать режим Auto для автоматического масштабирования.

В режиме **advanced...** каждая строка слева будет включать два дополнительных элемента. Один элемент (текстовое поле) задает имя одного поля используемого для этого графика фильтра отображения, а второй - способ расчета значения - SUM (сумма), COUNT (текущее значение счетчика), MAX (максимум), MIN (минимум), AVG (среднее) или LOAD (загрузка). Значение SUM может использоваться для любых целочисленных полей, COUNT - для всех полей, MAX, MIN и AVG - для численных и временных[2] полей, LOAD - только для временных полей.

Указанное в строке ввода имя поля должно быть частью фильтра, используемого для данного графика, в противном случае вычисление станет невозможным.

Например, для просмотра изменений времени отклика NFS (MAX/MIN/AVG) можно установить для первого графика

```
filter:nfs&&rpc.time Calc:MAX rpc.time
```

для второго

```
filter:nfs&&rpc.time Calc:AVG rpc.time
```

и для третьего

```
filter:nfs&&rpc.time Calc:MIN rpc.time
```

Для просмотра среднего количества размера от хоста **a.b.c.d** можно установить для графика

```
filter:ip.addr==a.b.c.d&&frame.pkt_len Calc:AVG frame.pkt_len
```

1 *Изменение цвета графика возможно только при включенной поддержке GTK+ версии 2.x*
2 *время отклика*

11.9.3.2.7.8.4 ONC-RPC

11.9.3.2.7.8.4.1 Programs

Эта команда активизирует диалоговое окно, содержащее статистические данные RTT[1] для всех программ ONC-RPC, с которыми связаны пакеты из файла захвата. Выводимые сведения включают имя и номер версии программ, типы вызовов RPC, а также минимальное, максимальное и среднее время кругового обхода.

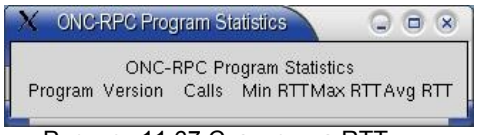


Рисунок 11.67 Статистика RTT для программ ONC-RPC

11.9.3.2.7.8.5 RTP Streams

Это субменю позволяет генерировать и просматривать статистику трафика для приложений, работающих в реальном масштабе времени на базе протокола RTP[2].

11.9.3.2.7.8.5.1 Show All

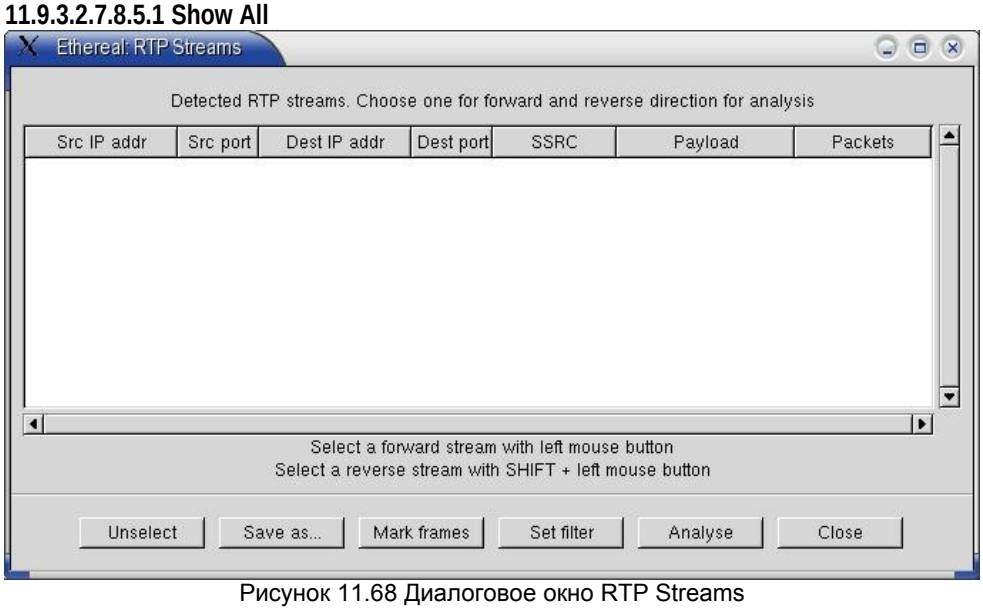


Рисунок 11.68 Диалоговое окно RTP Streams

Это диалоговое окно выводит информацию обо всех потоках RTP, для которых были собраны пакеты, включая адреса и номера портов, источник синхронизации, тип данных, число пакетов. Формат окна показан на рисунке 11.68.

Выбрав в списке интересующий вас поток RTP, вы можете проанализировать этот поток (кнопка Analyse).

Для списка доступных потоков обеспечивается возможность записи на диск, маркировки кадров, фильтрации и анализа.

11.9.3.2.7.8.5.2 Analyse

Результаты анализа выбранных потоков RTP выводятся в диалоговом окне **RTP Stream Analysis** (см. рисунок 11.69). В окне выводится список пакетов проанализированного потока с указанием порядковых номеров, задержки и ее флуктуаций, маркеров и состояния.

Кроме того, в нижней части диалогового окна выводятся сведения для всего потока в целом (число пакетов, число потерянных пакетов, количество пакетов с нарушением порядка доставки). Обеспечивается возможность просмотра списка пакетов для обоих направлений потока RTP.

Вы можете сохранить результаты анализа данных из потока в файле.

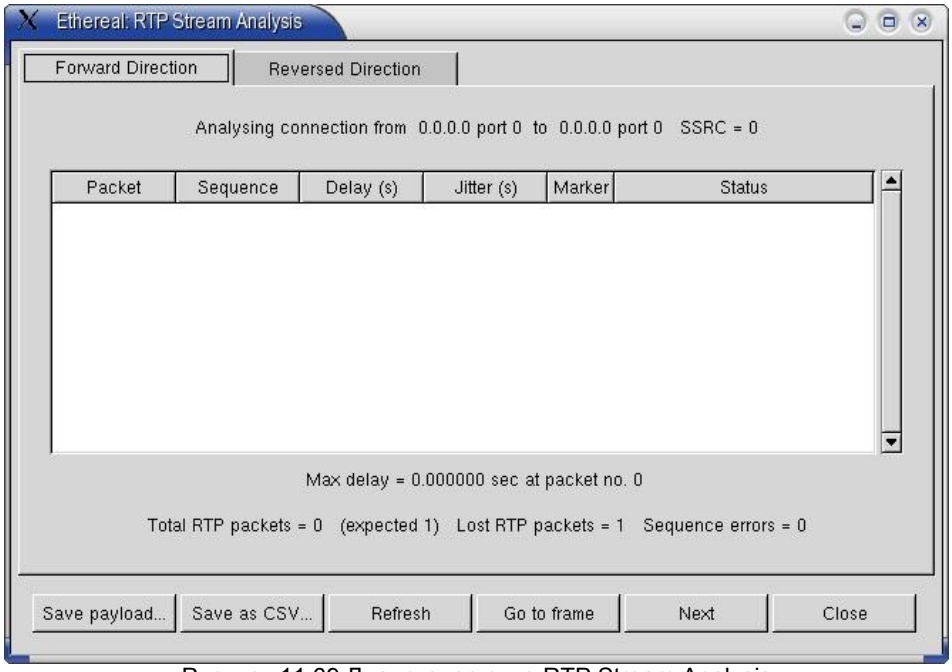


Рисунок 11.69 Диалоговое окно RTP Stream Analysis

1 *Round-trip Time - время кругового обхода.*
2 *Real Time Protocol*

11.9.3.2.8 Меню Help

Это меню обеспечивает доступ к справочной системе программы Ethereal. При активизации команды на экране появляется диалоговое окно **Help** (рисунок 11.70), обеспечивающее доступ к справочной информации. Окно включает 5 страниц:

Overview - обзорные сведения о программе.

Protocols - список поддерживаемых протоколов с краткой информацией о них.

Display Filters - информация о возможностях фильтрации выводимых в окне программы пакетов.

Capture Filters - информация о возможностях фильтрации пакетов в процессе их сбора.

FAQ - ответы на часто задаваемые вопросы.

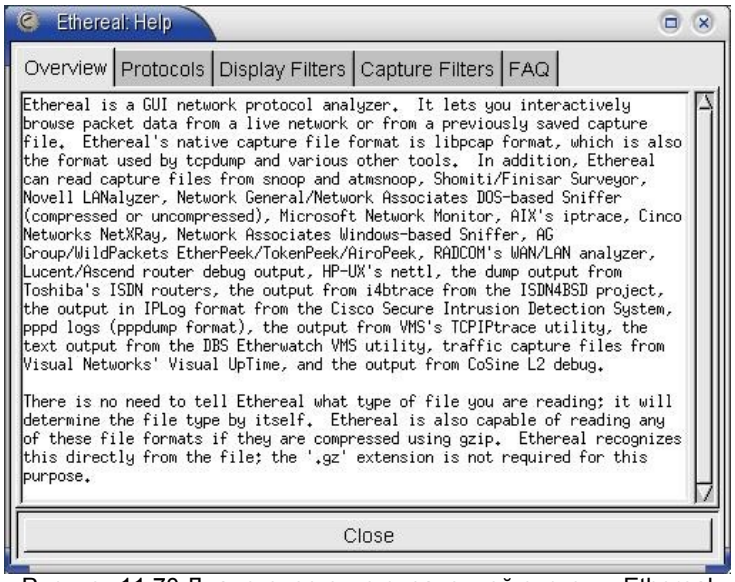


Рисунок 11.70 Диалоговое окно справочной системы Ethereal

11.9.3.2.8.1 About

Команда About выводит на экран окно с информацией о программе, включающей опции компиляции и версию используемой библиотеки libpcap.

11.9.3.2.9 Панель инструментов Ethereal

Кнопка	Действие
	Инициирует или прерывает процесс сбора пакетов (см. параграф на стр.)
	Прерывает или прерывает процесс сбора пакетов (см. параграф на стр.)
	Загружает собранные ранее пакеты из файла (см. параграф 11.9.3.2.2.1 на стр. 283)
	Сохраняет собранные программой пакеты в файле (см. параграф 11.9.3.2.2.2 на стр. 283)
	Закрывает файл захвата (см. параграф 11.9.3.2.2.1 на стр. 283)
	Повторно загружает собранные пакеты из файла (см. параграф 11.9.3.2.2.1 на стр. 283)
	Выводит на печать информацию о пакетах (см. параграф 11.9.3.2.2.3 на стр. 284)
	Находит пакет, соответствующий заданным условиям (см. параграф 11.9.3.2.3.1 на стр. 284)
	Находит следующий пакет (см. параграф 11.9.3.2.3.2 на стр. 285)
	Находит в списке пакет по указанному номеру (см. параграф 11.9.3.2.3.4 на стр. 285)
	Выводит диалоговое окно **Edit Capture Filters** (см. параграф 11.9.3.2.3.10 на стр. 286)
	Выводит диалоговое окно **Edit Display Filters** (см. параграф 11.9.3.2.3.11 на стр. 286)
	(см. параграф на стр.)
	Выводит диалоговое окно **Prefernces** (см. параграф 11.9.3.2.10 на стр. 301)
	Выводит на экран диалоговое окно справочной системы (см. параграф 11.9.3.2.8 на стр. 301)

11.9.3.2.10 Диалоговое окно Preferences

Диалоговое окно **Preferences**, активизируемое с помощью команды меню **Edit:Preferences** или кнопки на панели инструментов, позволяет пользователю задать предпочтительные режимы поведения программы Ethereal. Диалоговое окно содержит несколько страниц, переключение между которыми обеспечивается выбором соответствующего элемента из списка в левой части диалогового окна.

11.9.3.2.10.1 Страница Printing

Страница **Printing** служит для управления параметрами печати для команды меню **File:Print Packet** (параграф 11.9.3.2.2.4 на стр. 284). В двух верхней строках расположены переключатели **Format** и **Print to,** определяющие режим печати. Пакеты могут выводиться в текстовом виде или в формате PostScript на принтер или в файл.

Поле **Command:** позволяет ввести команду, используемую для печати[1], а поле **File:** - имя файла для записи в режиме **File**. Кнопка **File:** открывает диалоговое окно просмотра каталогов и выбора файлов для записи.

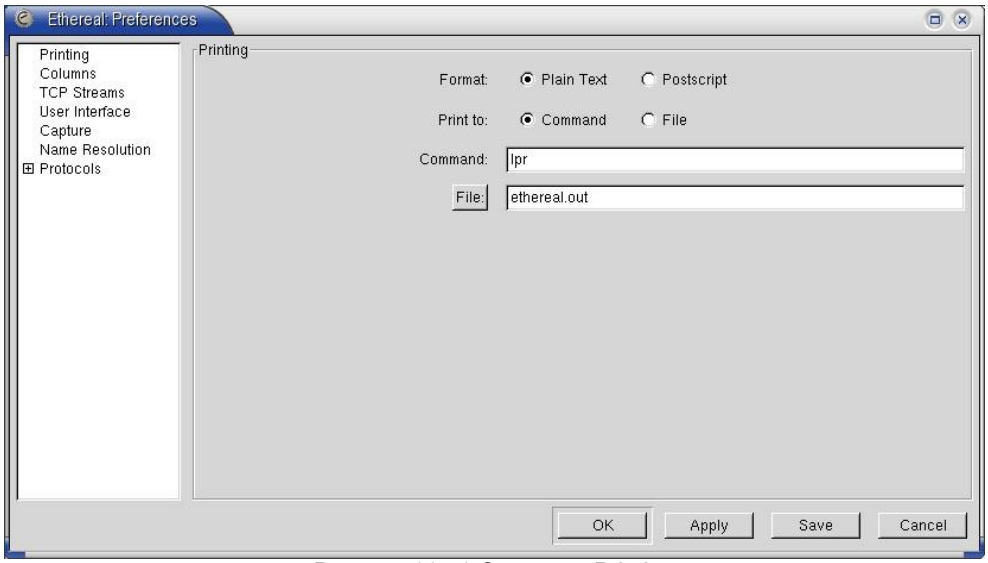


Рисунок 11.71 Страница Printing

11.9.3.2.10.2 Страница Columns

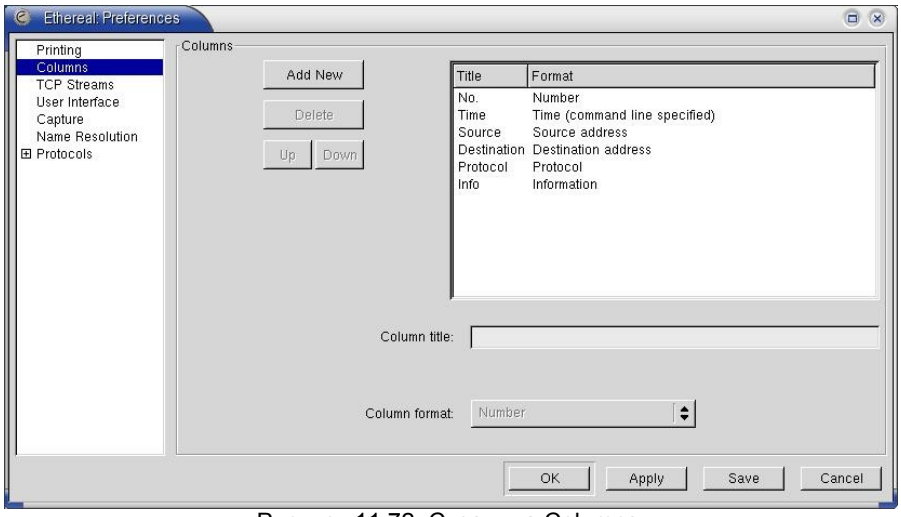


Рисунок 11.72. Страница Columns.

Страница **Columns** позволяет управлять столбцами списка пакетов, выводимого в главном окне программы.

Поле **Title** задает имя столбца, выводимое в заголовке списка пакетов. Для ввода имени новой колонки или изменения существующего имени служит поле ввода **Column title**. Тип данных в столбце определяется выбором одного из элементов списка **Column format**.

Группа кнопок слева от списка столбцов служит для управления отдельными элементами списка. Действия кнопок описаны в таблице 71.

Таблица 71 Кнопки управления колонками списка пакетов Ethereal

Кнопка	Действие
Add New	Добавляет в список новую колонку.
Delete	Удаляет из списка указанную колонку.
Up	Перемещает указанную колонку на одну позицию вверх по списку[2].
Down	Перемещает указанную колонку на одну позицию вниз по списку

В нижней части окна расположены кнопки управления списком в целом.

Таблица 72 Кнопки страницы Columns диалогового окна Preferences

Кнопка	Действие
OK	Закрывает окно с сохранением внесенных изменений до конца текущего сеанса работы программы. При завершении работы изменения будут потеряны, если вы не воспользуетесь кнопкой Save.
Apply	Эта кнопка не выполняет никаких действий в данном окне.
Save	Сохраняет список столбцов для использования по умолчанию при следующем запуске программы.
Cancel	Закрывает окно без сохранения изменений.

Отметим, что изменения в структуре списка пакетов реально произойдут только при следующем запуске программы.

1 *В системах UNIX это поле обычно содержит команду lpr.*
2 *В списке пакетов эта колонка будет перемещаться влево.*

11.9.3.2.10.3 Страница TCP Streams

Страница **TCP Streams** позволяет управлять цветами вывода текста в окне **TCP stream** (параграф 11.9.3.2.7.2 на стр. 292). Для изменения цвета достаточно выбрать атрибут в раскрывающемся списке **Set:** и указать желаемый цвет в поле выбора оттенков. Можно выбрать цвет и с помощью явного задания цветовых компонент. Выбранный текст будет показан в тестовом поле (справа вверху диалогового окна).

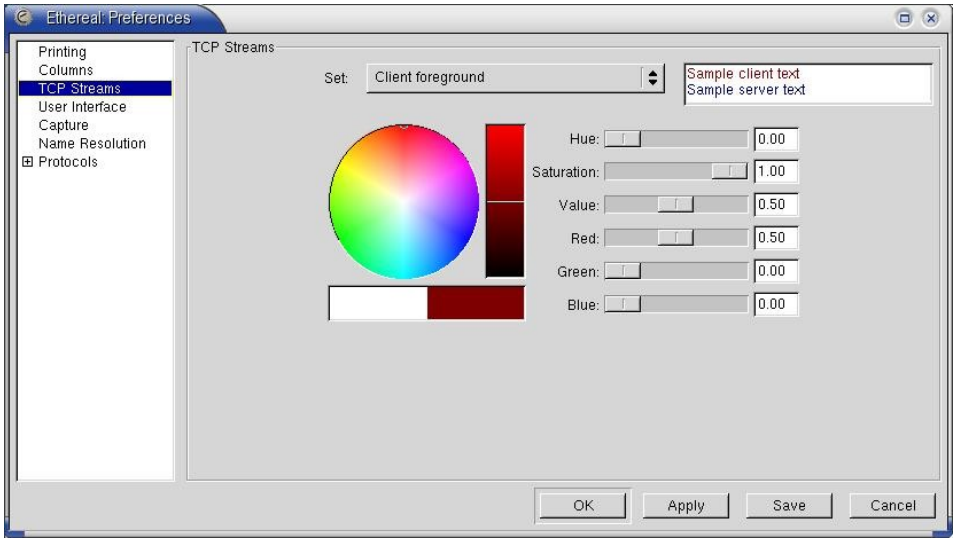


Рисунок 11.73. Страница TCP Streams.

11.9.3.2.10.4 Страница User Interface

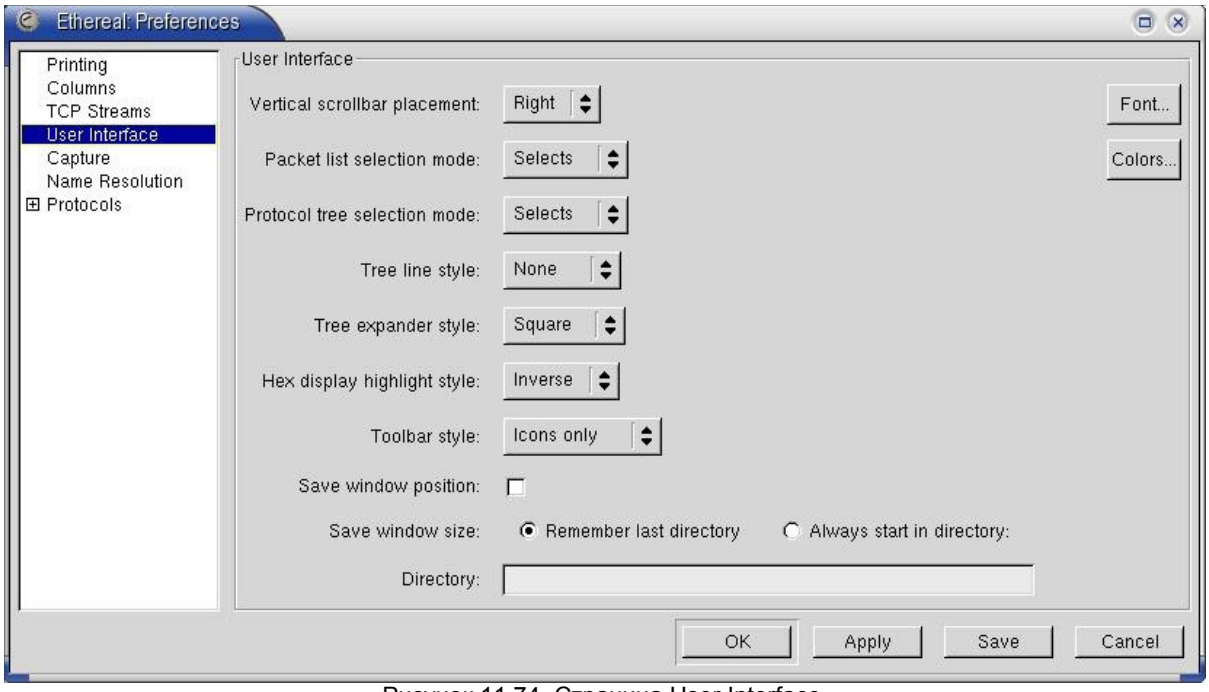


Рисунок 11.74. Страница User Interface.

Страница **User Interface** позволяет задать поведение отдельных элементов графического интерфейса программы:

Элемент	Назначение
Vertical scrollbar placement	Позволяет разместить поля прокрутки в панелях главного окна программы справа или слева.
Packet list selection mode	В режиме **Selects** перемещение указателя по списку пакетов приводит к смене содержимого панели дерева протоколов и дампа в соответствии с пакетом, на который установлена строка указателя. В режиме **Browses** перемещение указателя не приводит к изменению содержимого панели дерева протоколов и дампа, пока пакет в списке не будет указан явно (щелчок кнопкой мыши или нажатие клавиши пробела).
Protocol tree selection mode	В режиме **Selects** перемещение указателя по дереву протоколов приводит к смене содержимого панели дампа в соответствии с полем, на которое установлена строка указателя. В режиме **Browses** перемещение указателя не приводит к изменению содержимого панели дампа, пока поле не будет указано явно (щелчок кнопкой мыши или нажатие клавиши пробела).
Tree line style	Задает стиль линий, используемых для вывода дерева протоколов. Дерево может выводиться без соединительных линий (**none**), со сплошными (**solid**) или прерывистыми (**dotted**) линиями, а также в форме закладок (**tabbed**)

Элемент	Назначение
Tree expander style	Задает стиль вывода элементов раскрытия/закрытия ветвей дерева протоколов - без значков (**none**), треугольники (**triangle**), квадраты (**square**) и кружки (**circle**). В первом случае для раскрытия или закрытия ветви требуется двойной щелчок кнопкой мыши на соответствующей строке дерева, в остальных случаях достаточно однократного щелчка на соответствующем элементе.
Hex display highlight style	Задает стиль выделения в панели дампа - инверсия цвета (**inverse**) или жирный шрифт (**bold**).
Toolbar style	Задает стиль панели инструментов - пиктограммы (**icons only**), текст (**text only**) или то и другое (**icons & text**)

Поле выбора **Save Window Position** задает сохранение положения окна, а поле **Save Window Size** обеспечивает сохранение размеров окна при следующем запуске программы.

Кнопки **Fonts** и **Colors** в правом верхнем углу окна позволяют выбрать шрифт, используемый в программе и цвет используемый для маркированных кадров.

11.9.3.2.10.5 Страница Capture

Страница **Capture** позволяет управлять параметрами захвата кадров, устанавливаемыми по умолчанию для стартового диалога **Capture Options** (параграф 11.9.3.2.5.1.1 на стр. 288).

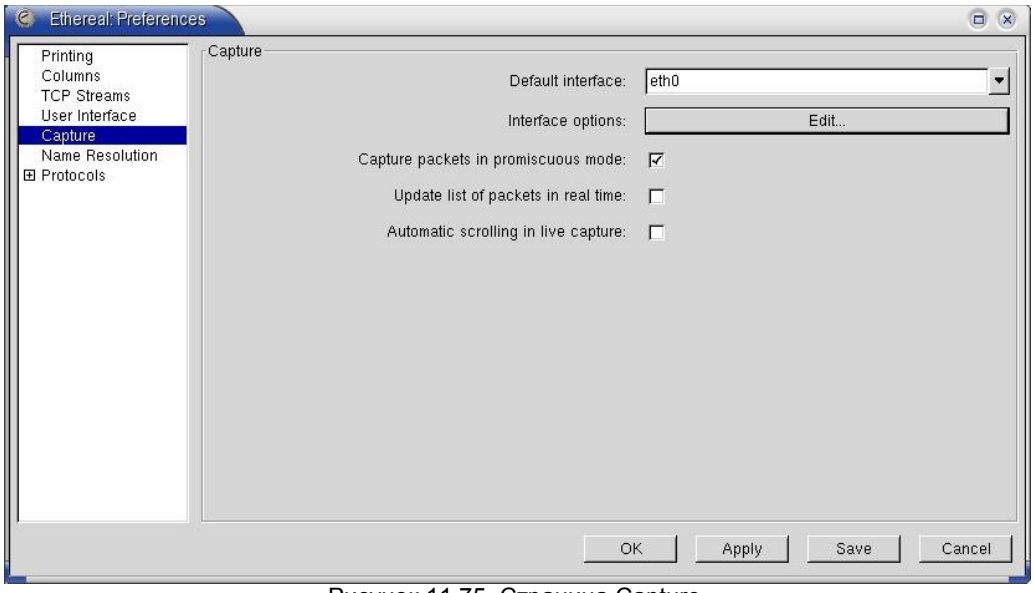


Рисунок 11.75. Страница Capture.

Поле **Default Interface**: (см. рисунок 11.75) служит для выбора интерфейса или буфера FIFO, который будет служить для сбора пакетов. Фиктивный интерфейс **all** в Linux-системах служит для сбора пакетов со всех интерфейсов системы.

Кнопка **Edit** активизирует диалоговое окно **Inreface Options** (рисунок 11.76), позволяющее выбрать некоторые опции интерфейса.

Поле выбора **Capture packets in promiscuous mode** определяет режим в котором должен находиться интерфейс, собирающий пакеты. В обычном режиме интерфейс будет принимать из среды только те кадры, в которых указан адрес канального уровня данного интерфейса. Режим захвата позволяет интерфейсу принимать из среды все передаваемые через нее кадры.

Поле выбора **Update list of packets in real time** позволяет задать режим обновления списка пакетов при “живом” захвате. Если вы поставите отметку в этом поле, пакеты будут появляться в списке на панели Ethereal по мере их захвата. В противном случае список пакетов появится только после завершения процедуры сбора пакетов.

Поле выбора **Automatic scrolling in live capture** обеспечивает управление режимом прокрутки списка пакетов при включенной опции обновления списка в реальном масштабе времени. Если вы отметите это поле, список собранных программой пакетов будет автоматически прокручиваться вверх так, чтобы последний пакет всегда находился в видимой части списка.

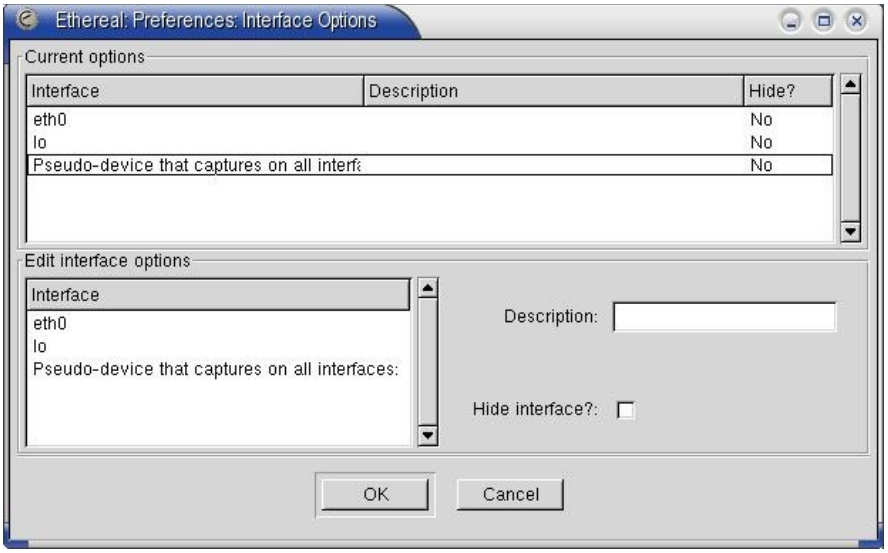


Рисунок 11.76. Диалоговое окно Inreface Options.

11.9.3.2.10.5.1 Диалоговое окно Inreface Options

Этот диалог служит для управления списком доступных интерфейсов, появляющимся на странице **Capture** диалогового окна **Preferences** и в окне выбора опций захвата кадров **Capture Options** (параграф 11.9.3.2.5.1.1 на стр. 288). Вы можете

изменить описание появляющихся в списке интерфейсов (поле **Description**)или спрятать (поле **Hide**) некоторые интерфейсы, чтобы они не включались в список доступных.

Кнопка **OK** служит для сохранения внесенных изменений, а кнопка **Cancel** закрывает диалоговое окно без сохранения результатов.

11.9.3.2.10.6 Страница Name Resolutions

Страница **Name Resolutions** (рисунок 11.77) управляет преобразованием адресов канального, сетевого и транспортного уровней в символьные имена.

Поле	Назначение
Enable MAC name resolutions	Включает или выключает преобразование MAC-адресов в символьные имена. Преобразование адресов осуществляется с использованием файлов **/etc/ethers** и **$HOME/.ethereal/ethers**, а при отсутствии такой записи адреса преобразуются в соответствии с записями из файла **manuf** программы Ethereal
Enable network name resolutions	Включает или выключает преобразование адресов сетевого уровня в имена хостов.
Enable transport name resolutions	Включает или выключает преобразование номеров портов транспортного уровня в символьные имена связанных с портами служб.

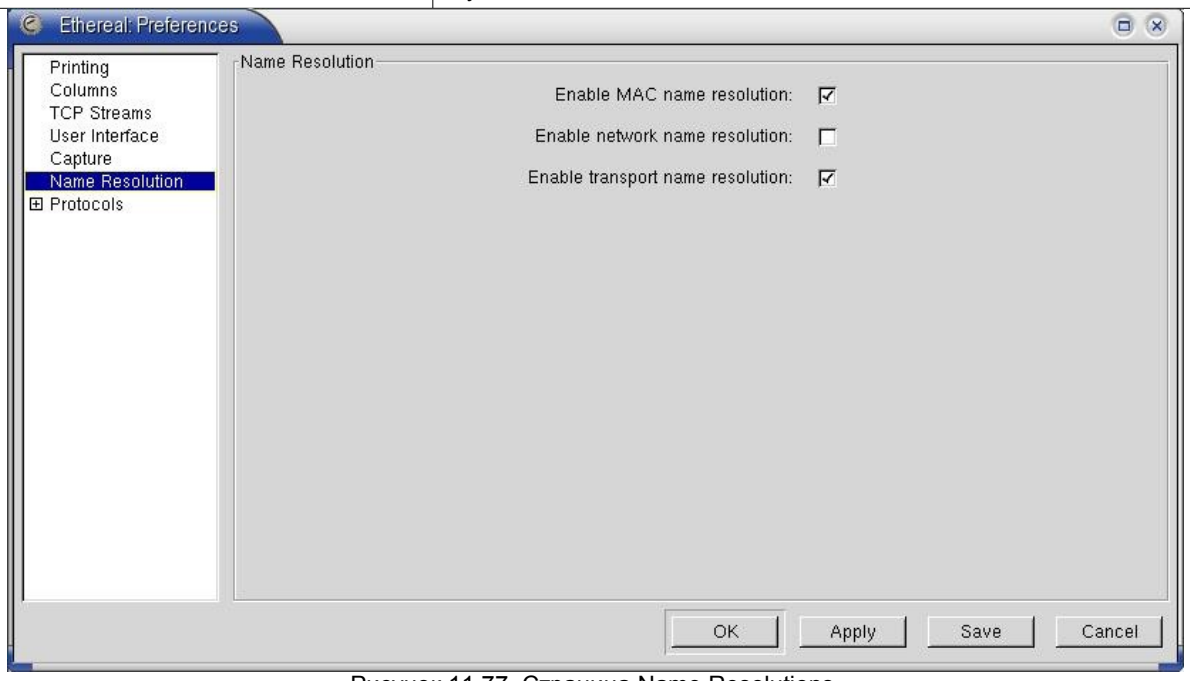


Рисунок 11.77. Страница Name Resolutions.

11.9.3.2.10.1 Страница Protocols

Эта страница (см. рисунок 11.78) служит для управления опциями трактовки полей отдельных протоколов, обрабатываемых программой Ethereal. Набор доступных опций зависит от выбранного протокола.

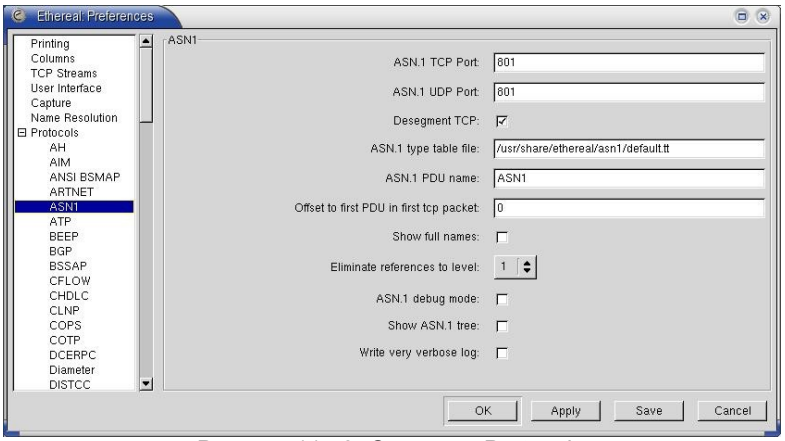


Рисунок 11.78. Страница Protocols.

11.9.3.3 Фильтры сбора пакетов

Фильтры, используемые при сборе пакетов, работают точно так же, как фильтры tcpdump и используют аналогичный синтаксис. Описание фильтров tcpdump приводится в параграфе 11.9.2.2 (стр. 265).

11.9.3.4 Фильтры отображения

Программы **Ethereal** и **Tethereal** (см. параграф 11.9.3.6 на стр. 309) поддерживают мощный язык фильтров отображения, позволяющий выводить для просмотра и анализа только интересующие пакеты из числа собранных этой или другой программой захвата. Полное описание системы фильтров отображения можно получить по команде **man ethereal-filter**.

Фильтры отображения позволяют выбирать пакеты на основе сравнения полей с заданными значениями, одного поля с другим или проверки существования указанных полей или протоколов.

Фильтры могут также использоваться для подготовки статистических отчетов (см. параграф 11.9.3.2.7.8.1 на стр. 295 и параграф 11.9.3.2.7.8.2 на стр. 296) или цветовой маркировки пакетов в списке пакетов Ethereal (см.

параграф 11.9.3.2.6.4 на стр. 289). В последующих параграфах описывается синтаксис фильтров отображения, а Приложение 12.20 (стр. 429) содержит список всех полей, которые могут использоваться в фильтрах отображения.

11.9.3.4.1 Синтаксис фильтров

Простейший фильтр позволяет проверить для кадра присутствие протокола или поля. Если вы хотите получить в списке все кадры, содержащие пакеты IPX, фильтр будет состоять лишь из идентификатора этого протокола - **ipx**. Для просмотра кадров Token Ring, содержащих поле RIF можно использовать выражение **tr.rif**.

Фильтры могут также использовать операции сравнения, перечисленные в таблице 73. Для записи операторов сравнения может использоваться синтаксис, подобный принятому в языке C, или сокращенная запись соответствующих терминов английского языка.

Таблица 73 Операции в фильтрах отображения

C	Английский термин	Операция
==	**eq**	равно
!=	**ne**	Не равно
>	**gt**	больше
<	**lt**	меньше
>=	**ge**	больше или равно
<=	**le**	меньше или равно
	contains	содержит протокол, строку или последовательность байтов

Каждое поле протокола, используемое в фильтрах отображения, имеет определенных тип (см. приложение 12.20). Поддерживаемый набор типов полей перечислен в таблице .

Таблица 74 Типы полей в фильтрах отображения

Обозначение	Тип поля
Unsigned integer	Беззнаковое целое число размером 6, 16, 24 или 32 бита.
Signed integer	Целое число со знаком размером 6, 16, 24 или 32 бита.
Boolean	Логическое значение.
Ethernet	MAC-адрес Ethernet (6 байтов).
Byte string	Строка байтов.
IPv4	Адрес IPv4 (4 байта).
IPv6	Адрес IPv6 (16 байтов).
IPX	Номер сети IPX.
String	Строка символов.
Double-precision floating point	Действительное число с плавающей запятой.

Целые числа могут задаваться в десятичном, восьмеричном или шестнадцатеричном формате. Например, три приведенных ниже выражения эквивалентны одно другому:

```
frame.pkt_len > 10
frame.pkt_len > 012
frame.pkt_len > 0xa
```

Логические поля могут принимать значения **true** (1) или **false** (0). В фильтрах отображения применяются только числовые эквиваленты логических значений. Например, для выбора кадров Token Ring с установленным полем source-routed может использоваться выражение:

```
tr.sr == 1
```

Адреса Ethernet и строки байтов представляются в шестнадцатеричной записи с разделением байтов двоеточием, точкой или дефисом:

```
fddi.dst eq ff:ff:ff:ff:ff:ff
ipx.srcnode == 0.0.0.0.0.1
eth.src == aa-aa-aa-aa-aa-aa
```

Если строка байтов содержит единственный байт, он представляется как целое число без знака. Т.е., если вы хотите проверить наличие в однобайтовом поле значения **ff**, вы должны сравнивать поле с **0xff** (а не с **ff**).

Адреса IPv4 представляются в десятичном формате с разделением байтов точками или задаются именами хостов:

```
ip.dst eq www.mit.edu
ip.src == 192.168.1.1
```

Адреса IPv4 можно сравнивать как числовые значения с использованием операций **eq**, **ne**, **gt**, **ge**, **lt** и **le**. При проверке адресов IPv4 может использоваться CIDR-нотация[1], если проверяемые адреса относятся к одной подсети.

1 *Classless InterDomain Routing - бесклассовая междоменная маршрутизация. Спецификации CIDR приведены в RFC 1518 и RFC 1519, которые можно загрузить с сайта http://rfc-editor.org/rfc/ или найти в каталоге Documents/ приложенного к книге компакт-диска.*

Например, для вывода списка всех адресов из сети класса B 129.111 можно воспользоваться выражением:

```
ip.addr == 129.111.0.0/16
```

Помните, что число справа от дробной черты указывает количество битов, используемых для представления номера сети. Нотацию CIDR можно использовать даже с именами хостов. Например, для выбора всех пакетов из сети класса C, к которой относится хост **sneezy** можно использовать выражение:

```
ip.addr eq sneezy/24
```

Нотацию CIDR можно использовать только с константами (адресами IP или именами хостов), но не с переменными. В частности, выражения типа

```
ip.src/24 == ip.dst/24
```

являются некорректными.

Сети IPX указываются 32-битовыми целыми числами без знака. Обычно для задания этих номеров используют шестнадцатиричное представление:

```
ipx.srcnet == 0xc0a82c00
```

текстовые строки указываются в двойных кавычках:

```
http.request.method == "POST"
```

Если строка содержит двойные кавычки, следует использовать символ обратной дробной черты перед знаком кавычек внутри строки или указывать взамен символа кавычек его шестнадцатеричный или восьмеричный код. Ниже приведены примеры использования этих вариантов:

```
browser.comment == "An embedded \" double-quote"
browser.comment == "An embedded \0x22 double-quote"
browser.comment == "An embedded \042 double-quote"
```

Если внутри строки используется символ \ его следует задавать последовательностью \\. Например, для вывода пакетов, содержащих строку **\\SERVER\SHARE** в поле **smb.path** следует задавать выражение:

```
smb.path contains "\\\\SERVER\\SHARE"
```

Существует возможность проверки наличия подстроки в поле любого протокола. Например, для проверки принадлежности адреса Ethernet определенному производителю (три старших байта MAC-адреса) можно воспользоваться выражением:

```
eth.src[0:3] == 00:00:83
```

Если для проверки используется только один байт поля, можно задавать для проверки шетнадцатеричное значение байта без префикса **0x**:

```
llc[3] == aa
```

Проверку подстроки можно использовать не только для полей, но и для любой последовательности байтов в кадре. Помните, что кадр канального уровня содержит пакет целиком и любое поле этого пакета можно задать в формате **смещение:размер**. и для имен протоколов. Например, выражение

```
eth[0x1a:3] == d4:30:c8
```

позволяет показать пакеты, отправленные всеми станциями нашей локальной сети 212.48.200.0/24[1]. Для выборки полей по смещению можно использовать несколько вариантов синтаксиса:

[i:j] - i задает стартовое смещение, j - размер;

[i-j] - i задает первый байт, j - последний (включительно);

[i] - i задает смещение для единственного сравниваемого байта;

[:j] - стартовое смещение равно 0, j задает размер;

[i:] - i задает стартовое смещение и данные считываются до конца поля.

Для смещения и размера можно использовать отрицательные значения. В этом случае отсчет ведется от конца поля. Например, для проверки последних 4 байтов кадра можно использовать выражение :

```
frame[-4:4] == 0.1.2.3
```

или

```
frame[-4:] == 0.1.2.3
```

Можно задать сложную выборку байтов, задавая смещения и диапазоны с использованием запятых и дефисов:

```
field[1,3-5,9:] == 01:03:04:05:09:0a:0b
```

Все описанные выше проверки можно комбинировать с использованием логических выражений, задаваемых в стиле языка C или сокращениями английских терминов:

and (**&&**) - логическая операция AND (И);

or (**||**) - логическая операция OR (ИЛИ);

not (**!**) - логическая операция NOT (отрицание).

Для группировки выражений можно использовать скобки. Ниже приведены примеры корректных выражений с использованием скобок и логических операций:

```
tcp.port == 80 and ip.src == 192.168.2.1
not llc
```

1 *Существует и более естественный способ задания такого фильтра, и приведенное выражение лишь иллюстрирует возможности проверки полей по смещению*

```
(ipx.srcnet == 0xbad && ipx.srnode == 0.0.0.0.0.1) || ip
tr.dst[0:3] == 0.6.29 xor tr.src[0:3] == 0.6.29
```

Особенно аккуратно следует относиться к полям, которые могут встречаться в пакетах неоднократно. Например, поле **ip.addr** встречается в каждом пакете IP дважды - как адрес получателя и как адрес отправителя пакета. Другим примером может служить поле **tr.rif.ring**, которое также неоднократно появляется в пакете. С учетом сказанного очевидно, что выражения:

```
    ip.addr ne 192.168.4.1
not ip.addr eq 192.168.4.1
```

не являются эквивалентными. Первой строка задает поиск пакетов, в которых существует поле **ip.addr**, значение которого не совпадает с **192.168.4.1**. Очевидно, что этому условию соответствуют пакеты, в которых хотя бы один из адресов отправителя и получателя не равен **192.168.44.1**. Второму правилу соответствуют все пакеты, кроме тех, где хотя бы одно значение **ip.addr** совпадает с **192.168.4.1**. Значит пакеты, в которых адрес отправителя или получателя имеет значение **192.168.4.1**, не будут соответствовать данному фильтру.

Следует также очень аккуратно задавать фильтры при необходимости избавиться от “шума”. Например, если вы хотите исключить из списка все широковещательные пакеты, адресованные хосту **224.1.2.3**, правило:

```
ip.dst ne 224.1.2.3
```

будет слишком жестким, поскольку оно будет отбирать только те кадры, в которых существует поле **ip.dst** и значение этого поля не равно указанному. В результате из списка будут удалены все кадры, не содержащие пакетов IP. Лучше будет воспользоваться одним из фильтров:

```
not ip or ip.dst ne 224.1.2.3
not ip.dst eq 224.1.2.3
```

Первое правило обеспечит включение в список кадров, не относящихся к протоколу IP (**not ip**) и пакетов IP, адресованных другим хостам (**ip.dst ne 224.1.2.3**). Второе правило обеспечит включение в список всех кадров, кроме тех, которые содержат указанный адрес IP в поле получателя.

11.9.3.5 Файлы Ethereal

Файлы **/etc/ethereal.conf** и **$HOME/.ethereal/preferences** содержат глобальные и персональные параметры настройки **Ethereal**, соответственно. Параметры задаются в формате **prefname:value**, где **prefname** совпадает с именем параметра в соответствующем диалоговом окне программы, а поле **value** содержит значение параметра. Между двоеточием после имени параметра и значением допускается использование пробелов и символов табуляции. Часть строки справа от символа **#** является комментарием.

При загрузке программы сначала просматриваются глобальные параметры, а потом файл персональных настроек.

Протоколы, для которых анализ полей запрещен, указываются в файле **$HOME/.ethereal/disabled_protos**, содержащем список имен протоколов, для которых анализ полей не производится. Каждая строка должна содержать не более одного имени протокола. Текст справа от символа **#** является комментарием.

Файл **/etc/ethers** служит для преобразования MAC-адресов в символьные имена. Если адрес не найден в этом файле, программа просматривает файл **$HOME/.ethereal/ethers**. Каждая строка такого файла содержит пару адрес-имя. В качестве разделителя между именем и адресом могут использоваться пробелы или символы табуляции, а для разделения байтов адреса могут использоваться двоеточие (:), дефис (-) или точка (.). Ниже приведен пример записей:

```
ff:ff:ff:ff:ff:ff          Broadcast
c0-00-ff-ff-ff-ff          TR_broadcast
00.00.00.00.00.00          Zero_broadcast
```

Файл **/usr/local/etc/manuf**[1] содержит список 3-байтовых идентификаторов, выделенных производителям оборудования. Эти идентификаторы используются для указания в списке пакетов в тех случаях, когда файлы **ethers** отсутствуют или не содержат искомого адреса. Записи в файле имеют формат

```
00:00:0C      Cisco
```

Кроме того, в этом файле перечислены специальные значения MAC-адресов, используемые для тех или иных целей. Например, запись

```
00-00-0C-07-AC/40    All-HSRP-routers
```

будет приводить к появлению в списке адресов поля **All-HSRP-routers** для всех MAC в диапазоне от **00-00-0C-07-AC-00** до **00-00-0C-07-AC-FF**. Размер значимой части специального адреса указывает маска, приведенная справа от дробной черты в строке адреса.

Файл **/etc/ipxnets** связывает 4-байтовые номера сетей IPX с символьными именами. Если искомая сеть не указана в этом файле, программа просматривает файл **$HOME/.ethereal/ipxnets**. Формат этих файлов аналогичен формату файлов **ethers**, но адреса указываются 4-байтовыми значениями вместо 6-байтовых MAC-адресов. Кроме того, номера сетей могут записываться без разделения отдельных байтов.

```
C0.A8.2C.00              HR
c0-a8-1c-00              CEO
00:00:BE:EF              IT_Server1
110f                     FileServer3
```

Файлы **/usr/local/etc/colorfilters** и **$HOME/.ethereal/colorfilters** содержат глобальные и персональные настройки цветовых фильтров, соответственно.

1 *В зависимости от параметров компиляции программы Ethereal каталог, где хранится этот файл, может изменяться*

11.9.3.6 Tethereal

Программа tethereal является текстовым вариантом анализатора Ethereal и поддерживает такие же функции и опции, за исключением тех, которые не применимы к текстовому интерфейсу. Пример вывода программы tethereal показан на рисунке 11.79.

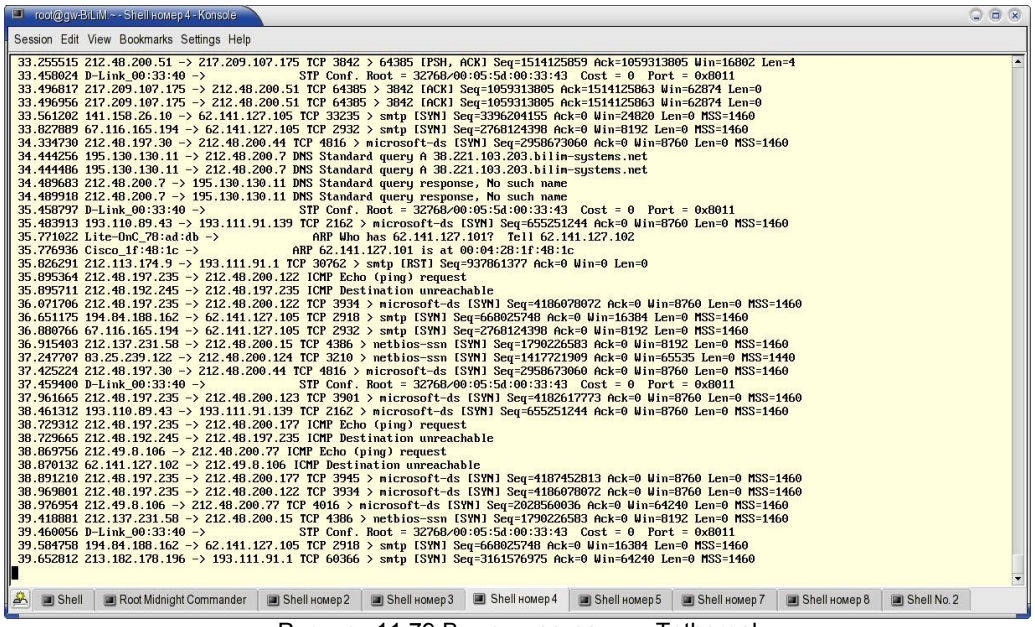


Рисунок 11.79 Вывод программы Tethereal

11.9.3.7 Утилиты Ethereal

Пакет Ethereal, кроме анализаторов протоколов с текстовым и графическим интерфейсом, включает утилиты для работы с файлами захвата, собранными Ethereal или другими программами.

Утилиты Ethereal могут читать и сохранять файлы захвата с использованием различных форматов, включая **libpcap** (**tcpdump**, **Ethereal** и др.), **snoop** и **atmsnoop**, **Shomiti/Finisar Surveyor**, **Novell LANalyzer**, **Network General/Network Associates Sniffer** (DOS, сжатые и несжатые/Windows), **Microsoft Network Monitor**, **iptrace** (AIX), Cinco Networks NetXRay, **AG Group/WildPackets EtherPeek/TokenPeek/AiroPeek**, **RADCOM WAN/LAN**, **Lucent/Ascend** (router debug), **nettl** (HP-UX), **дампы ISDN-маршрутизаторов Toshiba**, **i4btrace**, **IPLog** (Cisco Secure IDS), **pppd** (формат pppdump), **VMS TCPIPtrace/TCPtrace/UCX$TRACE**, **DBS Etherwatch VMS** (текстовый формат), **Visual Networks Visual UpTime**, **CoSine L2** (debug), **Accellent's 5Views LAN**, **Endace Measurement Systems ERF**, **Linux Bluez Bluetooth** (hcidump -w), **Network Instruments Observer** (v9). Утилитам не нужно указывать формат входного файла, он определяется автоматически. Утилиты Ethereal способны также работать со всеми перечисленными форматами файлов, сжатыми с использованием формата gzip, который распознается автоматически.

По умолчанию утилиты сохраняют файлы в формате **libpcap**, записывая в выходной файл все пакеты. Флаг **-F** позволяет задать формат выходного файла (стандартный и модифицированный формат **libpcap**, **snoop**, **Sniffer** (без компрессии), **Microsoft Network Monitor 1.x**, **Visual Networks**).

11.9.3.7.1 editcap

Программа **editcap** позволяет удалять пакеты из файлов захвата и обеспечивает преобразование файлов из одного формата в другой.

Синтаксис

```
editcap [-F <формат>] [-T <тип>] [-r] [-v] [-s snaplen] [-t time adjustment] [-h] infile
outfile [record# ...]
```

Опции

Таблица 75 Опции команды editcap

Опция	Значение
-F <формат>	Задает формат выходного файла.
-T <тип>	Задает тип инкапсуляции для выходного файла.
-r	Управляет записью в выходной файл пакетов с номерами из списка, указанного в командной строке.
-v	Заставляет **editcap** выводить номер версии при работе программы.
-s <размер>	Задает размер пакетов при записи в выходной файл.
-t <значение>	Задает корректировку временных меток при записи пакетов в выходной файл.
-h	Выводит справочную информацию и завершает работу программы.

В командной строке **editcap** можно указать список номеров пакетов, которые не будут записаны в выходной файл, если не задана опция **-r**, при использовании которой будут записываться только пакеты из указанного списка. Диапазоны номеров пакетов задаются в формате начало-конец.

При использовании в командной строке опции **-s** пакеты из входного файла, имеющие больший размер, усекаются при записи до заданного значения. Это может быть полезно в тех случаях, когда программа анализа не умеет

работать с большими пакетами[1].

При использовании флага **-t** корректировка времени осуществляется для всех пакетов выходного файла. Величина корректировки задается числом секунд и знаком. Например, **-t 3600** увеличит на один час значение временных меток для всех пакетов, **-t -0.5** уменьшит значение временных меток на полсекунды. Корректировка временных меток полезна для синхронизации данных, собранных в различных точках сети, если разницу в показаниях локальных часов можно определить или хотя бы оценить.

Задаваемый флагом **-T** тип инкапсуляции для выходного файла является лишь “рекомендательным” - если в заголовке входного пакета явно указан тип инкапсуляции, отличный от заданного опцией и несовместимый с ним, тип инкапсуляции в выходном файле не будет изменен.

Параметр **record# ...** задает список номеров пакетов, записываемых в выходной файл.

11.9.3.7.2 mergecap

Утилита **mergecap** позволяет объединить два файла данных (записанные пакеты) в один.

Синтаксис

```
mergecap [-hva] [-s snaplen] [-F <формат>] [-T encapsulation type] -w outfile infile ...
```

Опции

Таблица 76 Опции команды mergecap

Опция	Значение
-w	Задает имя выходного файла.
-F <формат>	Задает формат выходного файла.
-T <тип>	Задает тип инкапсуляции для выходного файла.
-a	Заставляет программу игнорировать временные метки и записывать в выходной файл сначала все пакеты из первого входного файла, затем из второго и т. д. По умолчанию пакеты из входных файлов при записи выходного файла упорядочиваются в соответствии с имеющимися в файлах временными метками для каждого пакета[2].
-v	Заставляет editcap выводить номер версии при работе программы.
-s <размер>	Задает размер пакетов при записи в выходной файл.
-h	Выводит справочную информацию и завершает работу программы.

Программа собирает в хронологическом порядке (если не задан флаг **-a**) пакеты из заданных входных файлов и записывает их в выходной файл.

При использовании в командной строке опции **-s** пакеты из входного файла, имеющие больший размер, усекаются при записи до заданного значения. Это может быть полезно в тех случаях, когда программа анализа не умеет работать с большими пакетами (см. примечание 2).

Для выходного файла используется такой же тип инкапсуляции, как во входных файлах (если тип инкапсуляции для них совпадает). Если входные файлы используют разные типы инкапсуляции, тип инкапсуляции выходного файла будет помещаться в поля **WTAP_ENCAP_PER_PACKET**[3]. Если поле **WTAP_ENCAP_PER_PACKET** не поддерживается выбранным форматом выходного файла, объединения данных в один выходной файл не происходит.

Задаваемый флагом **-T** тип инкапсуляции для выходного файла является лишь “рекомендательным” - если в заголовке входного пакета явно указан тип инкапсуляции, отличный от заданного опцией и несовместимый с ним, тип инкапсуляции в выходном файле не будет изменен.

11.10 Программы мониторинга соединений

Для мониторинга сетевых соединений в UNIX-системах существует множество средств - некоторых из них кратко описаны ниже. Самым простым и достаточно эффективным средством контроля соединений является программа netstat (параграф 11.1.2.5 на стр. 199) из пакета net-tools, входящего в состав большинства дистрибутивов. Одним из преимуществ **netstat** является присутствие практически на каждой UNIX-системе.

11.10.1 IPTraf

http://cebu.mozcom.com/riker/iptraf/

IPTraf является утилитой для мониторинга сетей IP. Программа перехватывает пакеты и выдает различную сведения о текущем трафике на основании данных из заголовков перехваченных пакетов. Выводимая программой IPTraf информация может включать:

- значения счетчиков объема данных **Total**, **IP**, **TCP**, **UDP**, **ICMP**, **non-IP**;

1 *Например программа **snoop** под управление Solaris 2.5.1 и Solaris 2.6 будет отбрасывать кадры Ethernet, размер которых превышает стандартное значение Ethernet MTU, что делает невозможным анализ кадров Gigabit Ethernet с использованием **jumbo**.*
2 *Программа предполагает, что пакеты в каждом из входных файлов упорядочены по времени.*
3 *Поле WTAP_ENCAP_PER_PACKET поддерживается не всеми форматами файлов захвата. В частности, формат libpcap его не поддерживает.*

- адреса и номера портов получателей пакетов TCP и UDP;
- счетчики пакетов и байтов TCP;
- состояние флагов TCP;
- типы сообщений ICMP;
- данные протокола маршрутизации OSPF;
- статистику для служб TCP и UDP;
- счетчики пакетов для каждого интерфейса;
- счетчики ошибок в контрольных суммах IP;
- индикаторы активности интерфейсов;
- статистику для станций ЛВС.

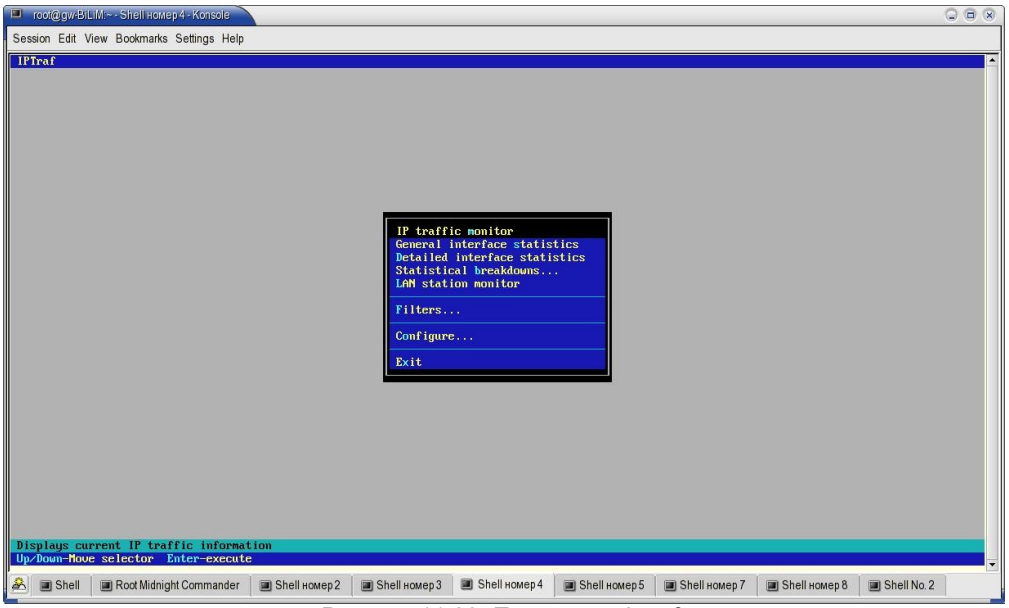


Рисунок 11.80. Программа iptraf.

Программу IPTraf можно использовать для мониторинга загрузки IP-сетей, большинства используемых типов сетевого сервиса, соединений TCP и т. д.

IPTraf является программным анализатором и не требует установки какого-либо специального оборудования. Программа использует встроенный raw-интерфейс (см. Приложение 12.8) ядра Linux с различными адаптерами Ethernet, FDDI, ISDN, Token Ring, асинхронными устройствами SLIP/PPP и другими сетевыми интерфейсами.

Для эффективной работы с программой вам потребуются базовые знания о протоколах стека TCP/IP (IP, TCP, UDP, ICMP и др.), которые вы можете почерпнуть на сайте http://www.protokols.ru.

11.10.1.1 Опции командной строки

Синтаксис

```
iptraf  [-f] [-q] [-i iface|-g|-d iface|-s iface|-z iface|-l iface [-t time] [-B [-L file ]]]|[-h]
```

При вводе команды **iptraf** без каких-либо опций программа запускается в интерактивном режиме и на экран выводится меню выбора режима работы (рисунок 11.80).

Описанные ниже опции (см. таблицу 77) позволяют сразу же выбрать режим работы программы **iptraf**, минуя главное меню программы.

Таблица 77. Опции команды iptraf.

Опция	Описание
`-i iface`	Запускает программу в режиме **IP traffic monitor** (параграф 11.10.1.4 на стр. 312) для мониторинга пакетов на указанном интерфейсе. Для мониторинга всех интерфейсов можно использовать опцию **-i all**.
`-g`	Запускает программу в режиме сбора статистики по всем интерфейсам системы (параграф 11.10.1.5.1 на стр. 316).
`-d iface`	Запускает программу в режиме вывода подробной статистики для отдельного интерфейса(параграф 11.10.1.5.2 на стр. 316).
`-s iface`	Запускает программу в режиме мониторинга TCP и UDP для указанного интерфейса .(параграф на стр.).
`-z iface`	Активизирует режим сбора статистики по размерам пакетов для указанного интерфейса(параграф на стр.).
`-l iface`	Запускает программу в режиме мониторинга станций ЛВС для указанного интерфейса или всех интерфейсов системы (параграф на стр.).
`-t time`	Задает продолжительность мониторинга (в минутах) для режимов **-i**, **-g**, **-d**, **-s**, **-z** или **-l**.
`-B`	Запускает программу в фоновом режиме мониторинга с использованием опций **-i**, **-g**, **-d**, **-s**, **-z** или **-l**. Программа закрывает стандартное устройство ввода, перенаправляет вывод в **/dev/null** и обеспечивает мониторинг в течение периода, заданного опцией **-t**. Работу программы в фоновом режиме можно прервать также с помощью сигнала **SIGUSR2**. При работе в фоновом режиме результаты мониторинга сохраняются в файле, заданном опцией **-t**. Если имя журнального файла не указано, будет использовано принятое по умолчанию имя журнального файла.

Опция	Описание
`-L file`	Эта опция позволяет задать имя журнального файла для записи данных мониторинга взамен принятого по умолчанию имени[1]. Если задано неполное имя файла, файл будет сохранен в каталоге **/var/log/iptraf**.
`-f`	Снимает все установленные блокировки и сбрасывает значения счетчиков для данного экземпляра программы, заставляя IPTraf думать, что это первый запуск. Эта опция может потребоваться для повторного запуска программы после неудачного завершения работы или системного сбоя.
`-h`	Выводит справку о работе с программой.

11.10.1.2 Сигналы программы

Сигнал **SIGUSR1** служит для обновления журнальных файлов в процессе работы. Получив такой сигнал программа прекращает запись в открытый файл, закрывает его и создает новый журнальный файл для записи информации.

Сигнал **SIGUSR2** служит для прерывания работы программы **IPTraf**, запущенной в фоновом режиме.

11.10.1.3 Преобразование адресов

Программа использует специальный демон **rvnamed** для ускорения процессов преобразования адресов IP в имена хостов, если включен режим такого преобразования (параграф 11.10.1.9 на стр. 321). Если демон **rvnamed** по каким-либо причинам не удается запустить (некорректная инсталляция, нехватка памяти и т. п.), преобразование адресов (если оно включено) будет выполняться с помощью обычных запросов DNS, что весьма замедляет работу программы и снижает скорость реакции на нажатие клавиш. Это связано с тем, что стандартная процедура преобразования адресов не возвращает управления, пока не будет определено имя хоста или получен отказ.

Демон **rvnamed** может создавать одновременно до 200 дочерних процессов с запросами DNS.

11.10.1.4 Режим IP Traffic Monitor

Режим **IP traffic monitor** можно активизировать с помощью одноименной команды меню, или воспользовавшись командой **iptraf -i <имя интерфейса>**. В режиме мониторинга обеспечивается декодирование заголовков всех пакетов IP в реальном масштабе времени и вывод информации об этих пакетах (рисунок 11.81). Для каждой сессии (соединения) выводятся адреса отправителя и получателя, а также указывается инкапсулированный в пакетах IP протокол и некоторые важные сведения о нем.

Отметим, что счетчики пакетов не принимают во внимание фрагментацию (т. е., 4 фрагмента одной дейтаграммы будут учтены как 4 отдельных пакета).

В режиме мониторинга окно программы делится на две панели - верхняя содержит сведения о соединениях TCP, а нижняя показывает весь трафик IP, не связанный с прямыми соединениями (UDP, ICMP и пр.). В каждой панели окна можно пользоваться прокруткой списка выводимой информации с помощью клавиш ↑ и ↓, а для переключения между панелями служит клавиша **w**.

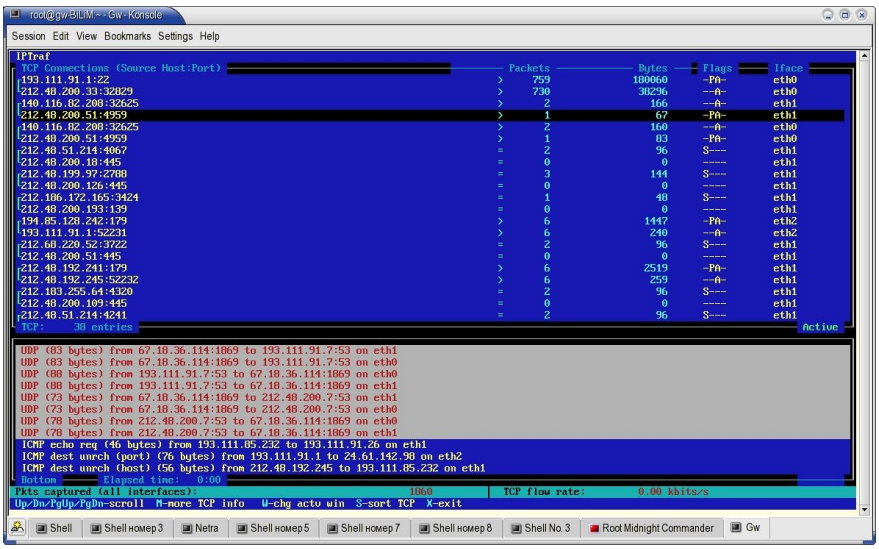


Рисунок 11.81 Режим IP Traffic Monitor

11.10.1.4.1 Панель TCP

Верхняя панель содержит сведения о присутствующих в системе соединениях TCP, включая:

- IP-адрес и номер порта отправителя;
- счетчики пакетов и байтов;
- размеры пакетов и окна TCP;
- MAC-адрес отправителя;
- состояния флагов TCP;
- имя интерфейса.

Каждое соединение TCP выводится в списке двумя строками (по одной строке для каждой стороны соединения). Пары, участвующие в соединении объединены скобкой **[** слева от адреса или имени в списке. Сведений об инициаторе соединения TCP программа не выводит.

Клавиши ↑ и ↓ позволяют перемещать указатель по списку соединений TCP, а клавиши **PgUp** и **PgDn** позволяют менять страницы вывода при большом количестве соединений.

Для выбранного в списке соединения TCP рассчитывается скорость потока данных, которая выводится в правой части строки состояния (поле **TCP flow rate** на рисунке 11.81). Установив указатель на соответствующую строку вы можете получить скорость потока для обоих направлений передачи в данном соединении.

Каждая запись в списке соединений TCP содержит поля, перечисленные в таблице 78.

1 *Используемые по умолчанию имена зависят от выбранного интерфейса и режима мониторинга.*

Таблица 78 Поля вывода в режиме IP Traffic Monitor

Поле	Описание
Source Host:Port	Адрес и номер порта отправителя пакетов для каждого направления в соединении TCP. Получателем пакетов является другая сторона соединения.
Packets	Счетчик пакетов, переданных отправителем в данном соединении TCP.
Bytes	Счетчик байтов, переданных отправителем в данном соединении TCP с учетом заголовков TCP и IP, но без учета заголовков канального уровня.
Pkt Size	Размер последнего пакета, переданного отправителем
Win Size	Размер окна TCP, анонсируемый отправителем.
Source MAC addr	MAC-адрес отправителя
Flags	Состояние флагов TCP (см. таблицу 79).
Iface	Интерфейс локальной системы, через который организовано данное соединение.

Для переключения вывода поля **Source MAC addr** и пар (**Packets** - **Bytes**), (**Pkt Size** - **Win Size**) используется клавиша **M**.

Таблица 79 Флаги TCP в режиме IP Traffic Monitor

Флаг	Описание
S	Флаг **SYN**, устанавливаемый в пакетах, которые передаются в процессе организации соединения TCP. Если в поле содержится только этот флаг (**S---**), пакет является запросом на организацию соединений. Присутствие дополнительного флага **A** (**S-A-**) говорит о том, что пакет является подтверждением приема запроса на организацию соединения и откликом на этот запрос.
A	Флаг **ACK**, устанавливаемый для пакетов подтверждения приема.
P	Флаг **PSH**, используемый для выталкивания пакета в начало очереди.
U	Флаг **URG**, используемый для индикации срочности содержащихся в пакете данных.
RESET	Флаг **RST**, показывающий индикацию отправителем сброса соединения в данном направлении.
DONE	Передача данных для этого соединения завершена и отправителем передан пакет **FIN**, который пока не подтвержден.
CLOSED	От другой стороны соединения получено подтверждение пакета **FIN**.
-	Флаг не установлен.

Вывод некоторых полей, описанных выше, зависит от настроек программы (см. параграф 11.10.1.9 на стр. 321). Клавиша **M** позволяет переключаться между режимами отображения полей. Значение **N/A** в каком-либо из полей говорит о недоступности соответствующей информации (например, MAC-адреса для соединения PPP).

Если запись не обновлялась в течение заданного пользователем времени, такая запись может быть удалена из списка, независимо от состояния соединения[1], с которым она связана. По умолчанию тайм-аут для обновления записей составляет 15 минут, но вы можете изменить это значение в меню настройки конфигурации (стр. 321).

Некоторые записи будут содержать символ **>** перед значением счетчика пакетов. Это означает, что в момент запуска режима мониторинга соединение уже существовало и значения счетчиков не отражают реально переданное через это соединение количество данных. Отметим, что строки с символом **>** могут достаточно долго оставаться в неизменном состоянии, если в момент запуска режима мониторинга соединение находилось в полузакрытом состоянии (одна сторона передала пакет FIN, а другая продолжает передачу).

Нижняя строка окна содержит суммарные значения счетчиков байтов для IP, TCP, UDP, ICMP и прочих (не IP) протоколов. Счетчики протоколов стека IP приводятся без учета заголовков канального уровня, для остальных протоколов (не IP) значения счетчиков учитывают и заголовки канального уровня.

Для маршрутизаторов при одновременном мониторинге всех интерфейсов каждое соединение будет появляться дважды - для входящего и исходящего интерфейса. Если маршрутизатор использует маскирование (трансляцию) адресов, адреса внутренней ЛВС будут отображаться в соответствии с реальными (не транслированными) значениями.

Наличие большого числа запросов SYN (записи с полем флагов **S---**) может говорить об атаке SYN-flood или сканировании портов. В таких случаях следует незамедлительно принимать меры, поскольку результатом атаки может быть перегрузка атакуемого хоста. Для блокировки атаки можно воспользоваться фильтрами **iptables** (параграф 5.1).

Записи для соединений TCP, которые были сброшены, получили пакет RST или бездействуют слишком долго, будут удаляться из списка, при включенной опции **TCP closed/idle persistence...** (стр. 321). Клавиша **F** позволяет провести такую очистку в любой момент.

Для сортировки списка соединений TCP нажмите клавишу **S** и выберите критерий сортировки:

- счетчик пакетов (**P**);
- счетчик байтов (**B**).

1 *Некоторые соединения могут весьма продолжительное время сохраняться в открытом состоянии без реальной передачи данных через это соединение.*

При сортировке списка выбирается большее из двух значений счетчиков для каждого соединения и сортировка производится в порядке убывания. Отметим, что сортировка осуществляется однократно (во избежание перегрузки процессора автоматическим обновлением списка) и для повторной сортировки нужно снова нажать соответствующие клавиши

11.10.1.4.2 Панель мониторинга трафика, не связанного с соединениями

В нижней части окна мониторинга выводится панель с информацией о протоколах, не связанных с организацией прямых соединений TCP. Обеспечивается декодирование заголовков для протоколов:

- UDP - протокол пользовательских дейтаграмм;
- ICMP - протокол управляющих сообщений Internet;
- OSPF - протокол маршрутизации;
- IGRP - протокол маршрутизации;
- IGP - протокол маршрутизации;
- IGMP - протокол управления группами Internet;
- GRE - General Routing Encapsulation;
- ARP - протокол преобразования адресов;
- RARP- протокол преобразования адресов.

Для нераспознанных протоколов указывается номер протокола IP, а пакеты, не относящиеся к протоколу IP, помечаются префиксом **Non-IP**. Для пакетов ARP и RARP[1] указываются также MAC-адреса.

Если пакеты содержат фрагменты дейтаграмм, информация выводится только для первого фрагмента, поскольку только он содержит заголовок инкапсулированного протокола. Для пакетов UDP также указывается IP-адрес и номер порта, а для ICMP - тип сообщения ICMP. Для простоты идентификации протоколов используется цветовая маркировка пакетов различных протоколов, как показано в таблице 80.

Нижняя панель может включать до 512 записей. После достижения максимального количества новые записи заменяют самые старые. Для просмотра записей этой панели следует активизировать панель с помощью клавиши **W** и перемещаться по списку, используя клавиши ↑ и ↓. Содержимое окна автоматически прокручивается по мере поступления новых пакетов. Некоторые записи списка могут превышать по размеру ширину экрана. В этом случае для перемещения можно использовать клавиши ← и → при активной панели.

Записи для пакетов, принятых с помощью интерфейсов ЛВС включают MAC-адрес хоста, с которого был получен пакет, если включена соответствующая опция меню конфигурации (стр. 321).

Таблица 80 Цветовая маркировка пакетов

Протокол	Цвет
UDP	Красный на белом фоне.
ICMP	Желтый на синем фоне
OSPF	Черный на бирюзовом (Cyan) фоне.
IGRP	Коричневый на бирюзовом (Cyan) фоне.
IGP	Красный на бирюзовом (Cyan) фоне.
IGMP	Ярко-зеленый на синем фоне.
GRE	Синий на белом фоне.
ARP	Ярко-белый на красном фоне.
RARP	Ярко-белый на красном фоне.
Прочие IP	Желтый на красном фоне.
Non-IP	Желтый на красном фоне.

11.10.1.4.2.1 Содержимое записей

В общем случае каждая запись нижней панели содержит идентификатор протокола, размер дейтаграммы IP или полного кадра (для протоколов non-IP, включая ARP и RARP), адреса отправителя и получателя, а также сетевой интерфейс, принявший пакет. Для некоторых протоколов выводится также дополнительная информация, перечисленная ниже.

11.10.1.4.2.1.1 ICMP

Записи для пакетов ICMP имеют форму:

`ICMP` ***`type`*** **`[(`**`subtype`**`)] (`**`size`** **`bytes) from`** ***`src`*** **`to`** ***`dst`*** **`[(src HWaddr`** ***`srcMACaddress`***`)] on` ***`interface`***

Поле типа (**type**) может принимать одно из перечисленных в таблице 81 значений.

Таблица 81 Типы сообщений ICMP

Тип	Описание
`echo req, echo rply`	Запросы и отклики **ICMP echo**, обычно используемые программой **ping** и другими средствами мониторинга и диагностики.

1 *Строго говоря, эти протоколы не относятся к стеку IP, поскольку они не инкапсулируются в пакеты IP. Однако эти пакеты не помечаются префиксом Non-IP, чтобы показать их тесную связь с IP.*

Тип	Описание
dest unrch	Сообщение о недоступности адресата **ICMP destination unreachable**. Такие сообщения сопровождаются дополнительной информацией, которая может прояснить причину и характер недоступности. Коды причин приведены в таблице 82.
redirct	Сообщение **ICMP redirect**, генерируемое обычно маршрутизатором и сообщающее о том, что к хосту имеется лучший маршрут.
src qnch	Сообщение **ICMP source quench**, используемое для остановки передачи макетов хостом. Такие сообщения являются частью механизма управления потоком данных в IP.
time excd	Показывает, что время жизни пакета исчерпалось на пути к адресату. Такие сообщения часто используются программами трассировки (например, **traceroute**).
router adv	**ICMP router advertisement** - анонс маршрутизатора.
router sol	**ICMP router solicitation** - запрос маршрутизатора.
timestmp req	**ICMP timestamp request** - запрос временной метки.
timestmp rep	**ICMP timestamp reply** - отклик на запрос временной метки.
info req	**ICMP information request** - запрос информации.
info rep	**ICMP information reply** - отклик на запрос информации.
addr mask req	**ICMP address mask request** - запрос адресной маски.
addr mask rep	**ICMP address mask reply** - отклик на запрос адресной маски.
param prob	**ICMP parameter problem** - некорректные параметры.
bad/unknown	Получен поврежденный или неопознанный пакет ICMP.

Сообщения о недоступности адресата (**destination unreachable**) сопровождаются сведениями о причинах, перечисленных в таблице .

Таблица 82 Коды причин сообщений destination unreachable

Код	Причина недоступности
ntwk	Сеть недоступна.
host	Хост недоступен.
proto	Протокол недоступен.
port	Порт недоступен.
pkt fltrd	Пакет отфильтрован[1].
DF set	Требуется фрагментирование при наличии флага DF[2].
src rte fail	Отправитель задал некорректный маршрут.
src isltd	Отправитель изолирован[3].
net comm denied	Соединение с сетью отвергнуто.
host comm denied	Соединение с хостом отвергнуто.
net unrch for TOS	Сеть недоступна для заданного типа обслуживания (IP TOS).
host unrch for TOS	Хост недоступен для заданного типа обслуживания (IP TOS).
prec violtn	Нарушение приоритета.
prec cutoff	Приоритет урезан.
dest net unkn	Сеть адресата неизвестна.
dest host unkn	Адресат неизвестен.

Дополнительные сведения о сообщениях ICMP вы сможете найти в RFC 792.

11.10.1.4.2.1.2 OSPF

Для пакетов протокола маршрутизации OSPF также выводится дополнительная информация. Строки для пакетов OSPF имеют формат:

OSPF *type* (a=*area* r=*router*) (*size* bytes) from *source* to *destination* [(src HWaddr *srcMACaddress*)] on *interface*

Поле **type** определяет тип сообщения и может содержать одно из перечисленных в таблице 83 значений.

1 *Обычно такие сообщения говорят о том, что пакет противоречит тому или иному правилу межсетевого экрана.*
2 *Запрет фрагментирования.*
3 *Этот тип сообщений устарел и практически не используется.*

Таблица 83 Типы сообщений OSPF

Тип	Описание
hlo	Приветствие **OSPF hello**, используемое для организации и поддержки сеанса между маршрутизаторами.
DB desc	**OSPF Database Description** - описание базы данных протокола.
LSR	**OSPF Link State Request** - запрос состояния канала.
LSU	**OSPF Link State Update** - обновление машины состояний, показывающее состояние сетевых соединений OSPF.
LSA	**OSPF Link State Acknowledgment** - подтверждение.

В квадратных скобках указываются значения номера области OSPF (**a=area**) и IP-адрес маршрутизатора, создавшего сообщение (**r=router**). Отметим, что этот адрес не обязан совпадать с адресом отправителя в заголовке IP.

Во многих случаях для пакетов OSPF в качестве адреса получателя указывается групповой адрес класса D или (при включенном преобразовании адресов) домен **MCAST.NET**. Для адресов этого типа определены значения:

224.0.0.5 (OSPF-ALL.MCAST.NET)
для обозначения всех маршрутизаторов OSPF

224.0.0.6 (OSPF-DSIG.MCAST.NET)
для всех означенных (designated) маршрутизаторов OSPF.

Более подробные сведения о сообщениях OSPF вы сможете найти в RFC 2178.

11.10.1.5 Статистика для интерфейсов

Программа поддерживает два режима сбора статистики для интерфейсов. Один из режимов (опция **-g**)обеспечивает вывод основных параметров статистики для всех интерфейсов системы, а во втором (опция **-d**) выводится детальная статистика для выбранного интерфейса.

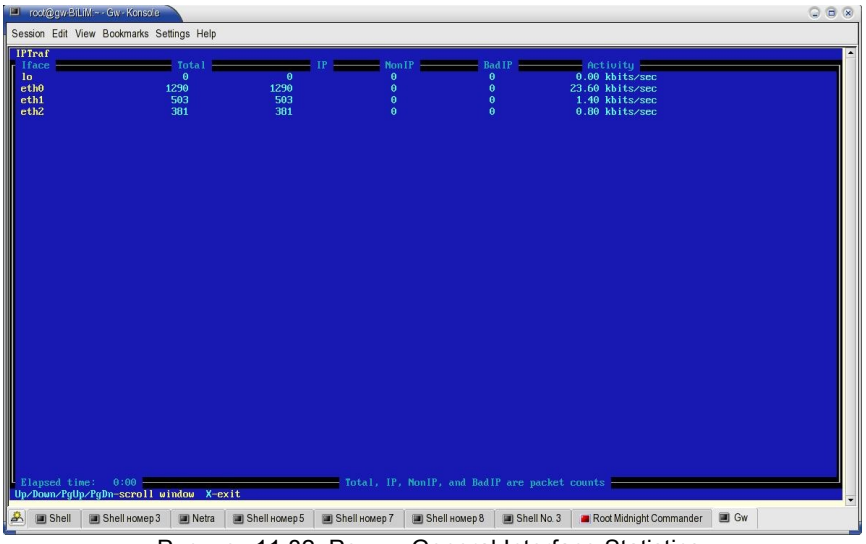


Рисунок 11.82. Режим General Interface Statistics.

Счетчики пакетов в режиме сбора статистики реально учитывают кадры канального уровня. Поэтому при получении 3 фрагментов одной дейтаграммы IP значения счетчиков увеличатся на 3, а не на 1.

На результаты вывода статистики оказывают влияние фильтры IPTraf, описанные в параграфе 11.10.1.8 (стр. 319).

11.10.1.5.1 Режим General Interface Statistics

В этом режиме выводится базовая статистика для всех имеющихся в системе сетевых интерфейсов. Статистика включает объем трафика для всех протоколов, протоколов IP, non-IP, а также количество поврежденных или некорректных пакетов IP (несоответствие контрольной суммы). В последней колонке выводится уровень активности интерфейса, определяемый числом пакетов или байтов, проходящих через этот интерфейс за секунду. Для переключения режима учета активности (пакеты или килобайты) служит меню настройки конфигурации программы (параграф 11.10.1.9 на стр. 321). При расчете статистики принимаются во внимание как входящие, так и исходящие пакеты.

При активизации в системе новых интерфейсов они будут автоматически добавляться в список окна мониторинга. При большом количестве интерфейсов для перемещения по списку можно использовать клавиши ↑ и ↓, **PgUp** и **PgDn**.

При включенном режиме записи в журнальные файлы (стр. 321) копия выводимой на экран статистики записывается в файл **iface_stats_general.log** с заданным интервалом.

Программу можно запустить сразу в режиме сбора базовой статистики по всем интерфейсам с помощью команды

```
iptraf -g
```

11.10.1.5.2 Режим Detailed Interface Statistics

В режиме детальной статистики для выбранного интерфейса обеспечивается вывод более подробных сведений, нежели в режиме **General interface statistics**. Для заданного интерфейса выводятся:

- значения счетчиков пакетов и байтов (суммарные, а также отдельно для входящего и исходящего трафика) с разбиением по протоколам:

- все протоколы IP;
- TCP;
- UDP;
- ICMP;
- другие протоколы IP;
- протоколы, отличные от IP;

- значения счетчика ошибок в контрольных суммах пакетов;
- скорости передачи и приема данных в пакетах и килобайтах;
- счетчики пакетов и байтов для широковещательного трафика.

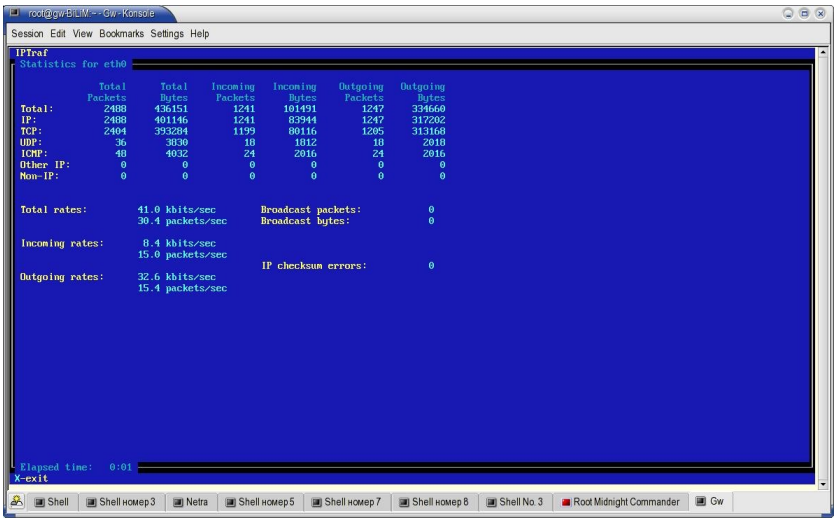


Рисунок 11.83. Режим Detailed Interface Statistics.

Для всех протоколов IP (IP, TCP, UDP, ICMP, other IP) счетчики байтов учитывают размер пакетов на сетевом уровне (заголовок IP и данные) без учета заголовков канального уровня. Для прочих протоколов и суммарной статистики по всем протоколам счетчики байтов учитывают также заголовки канального уровня.

В нижней части экрана (см. рисунок 11.83) указываются значения суммарной скорости потока данных и отдельные значения для исходящего и входящего потоков в пакетах и килобайтах или килобитах[1]. Исходящими считаются не только пакеты, порожденные данным хостом, но и транзитные пакеты, передаваемые интерфейсом маршрутизатора. Входящими считаются все пакеты, принимаемые интерфейсом, независимо от того, адресованы они данному хосту или пересылаются другим машинам через данный интерфейс маршрутизатора. Если интерфейс находится в режиме захвата пакетов, входящий трафик будет включать весь объем данных, проходящих через среду передачи.

Отметим, что на скорость исходящего потока данных могут оказывать существенное влияние процессы буферизации пакетов.

Если вы хотите запустить программу в режиме детального сбора статистики для одного интерфейса, можно воспользоваться опцией **-d** с именем интересующего интерфейса в качестве параметра. Например, команда

```
iptraf -d eth0
```

обеспечит вывод детальной статистики для интерфейса **eth0**.

При включенном режиме записи в журнальные файлы (стр. 321) копия выводимой на экран статистики записывается в файл **iface_stats_detailed-<имя интерфеса>.log** по окончании сбора данных.

11.10.1.6 Статистика пакетов

Статистика пакетов показывает распределение пакетов по размерам и номерам портов TCP/UDP.

11.10.1.6.1 Статистика распределения пакетов по размеру

Окно статистики содержит информацию о распределении пакетов по размерам для 20 градаций в интервале от 1 до MTU. Выводимая в окне статистика постоянно обновляется по мере доставки новых пакетов. Пример вывода статистики вы можете видеть на рисунке 11.84.

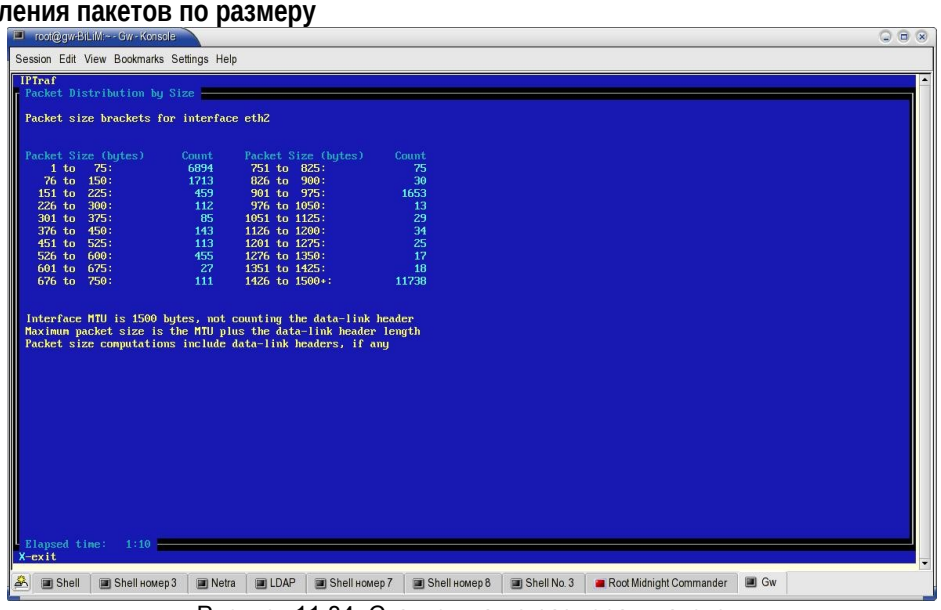


Рисунок 11.84. Статистика по размерам пакетов.

Если включена запись данных в журнальные файлы, копия статистики регулярно записывается в файл. По умолчанию журнальный файл статистики использует имя **packet_size-iface.log** (вместо **iface** используется имя интерфейса, собирающего пакеты).

Фильтры IPTraf не влияют на выводимую в этом режиме статистику.

Программу можно сразу запустить в режиме сбора статистики по размерам пакетов для одного интерфейса с помощью опции **-z**. Например, команда

1 *В зависимости от параметров конфигурации (см. стр. 321)*

```
iptraf -z eth0
```
будет обеспечивать сбор и вывод статистики распределения по размерам пакетов для интерфейса **eth0**.

11.10.1.6.2 Статистика распределения по портам TCP и UDP

Программ IPTraf позволяет также собирать и просматривать статистику распределения пакетов по портам TCP и UDP. В этом режиме выводится распределение количества пакетов и байтов по портам с номерами от 1 до 1023 (привилегированные порты). Пакет учитывается в таблице статистики если порт отправителя или получателя попадает в указанный диапазон номеров. Для трафика TCP и UDP с совпадающими номерами портов окно статистики будет содержать две строки.

Каждая строка статистики содержит имя протокола (TCP или UDP), номер порта, суммарные счетчики пакетов и байтов для данной пары **порт-протокол**, а также счетчики пакетов, адресованных в этот порт и отправленных из порта с таким номером. Счетчики байтов не учитывают заголовки канального уровня.

Строки списка промаркированы желтым цветом для протокола TCP и зеленым для протокола UDP.

Выводимые в этом режиме результаты зависят от фильтров IPtraf (параграф 11.10.1.8 на стр. 319).

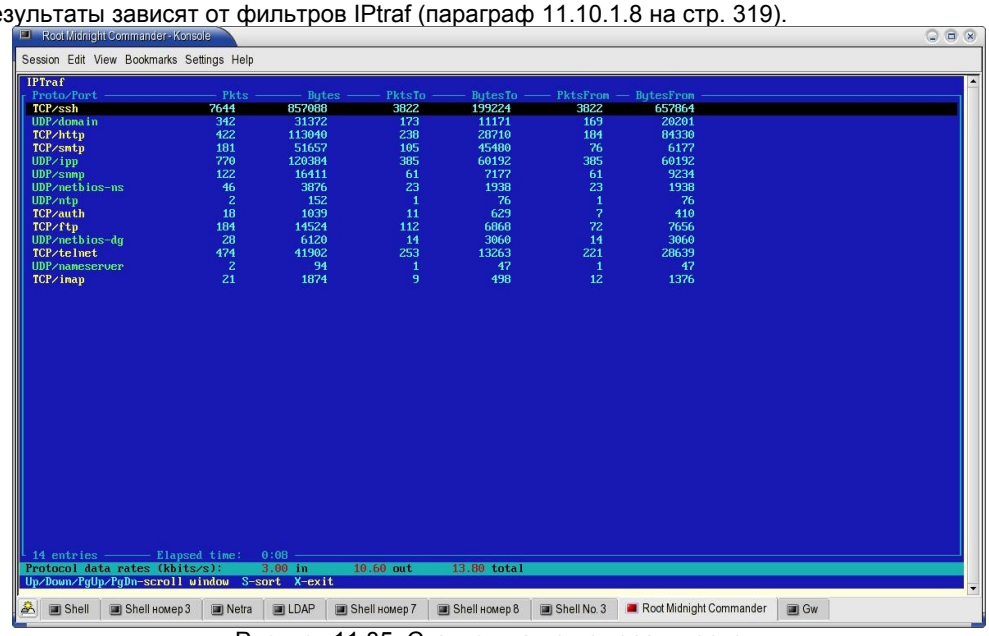


Рисунок 11.85. Статистика по номерам портов.

Если вас интересует распределение пакетов по портам с номерами больше 1023, вы можете настроить сбор информации для таких портов в меню **Configure.../Additional ports...** (стр. 321).

При включенной записи в журнальные файлы копия статистики будет также сохраняться в файле **tcp_udp_services-iface.log** (взамен **iface** будет использоваться имя реального интерфейса; например, **tcp_udp_services-eth0.log**).

IPTraf рассчитывает скорость (суммарный, входящий и исходящий трафик) для протокола, указанного в списке и выводит полученные значения в нижней части окна.

На выводимые в окне статистики распределения по портам результаты оказывают влияние фильтры IPtraf (параграф 11.10.1.8 на стр. 319).

Для запуска программы в режиме сбора статистики по портам непосредственно из командной строки служит опция **-s** с именем интерфейса в качестве параметра.

11.10.1.6.2.1 Сортировка списка

Для сортировки списка можно воспользоваться клавишей **S**, которая активизирует всплывающее меню выбора ключа сортировки. Для сортировки по портам нажмите клавишу **R**, для сортировки по общему числу пакетов - **P**, для сортировки по суммарному количеству байтов - **B**, для сортировки по числу входящих и исходящих пакетов - **T** и **F**, соответственно, а для сортировки по количеству принятых и переданных байтов - **O** и **M**.

Сортировка по портам осуществляется в порядке роста номеров, во всех прочих режимах сортировка осуществляется в порядке уменьшения значений счетчиков.

11.10.1.7 Режим LAN Station Statistics

Режим **LAN station monitor** позволяет определять MAC-адреса станций локальной сети и выводить статистику по числу входящих и исходящих пакетов, а также скорость входящего и исходящего трафика для каждой найденной станции ЛВС.

Каждая запись списка станций ЛВС (см. рис. 11.86) указывает тип сетевого интерфейса станции (**Ethernet**, **PLIP**, **Token Ring**, **FDDI**) и ее MAC-адрес. Строка статистики для станции содержит значения счетчиков:

- суммарного входящего трафика (пакеты);
- входящих пакетов IP;
- суммарного объема принятых данных (в байтах);
- скорости входящего потока;
- суммарного исходящего трафика (пакеты);
- исходящих пакетов IP;
- суммарного объема переданной информации (в байтах);
- скорости исходящего потока.

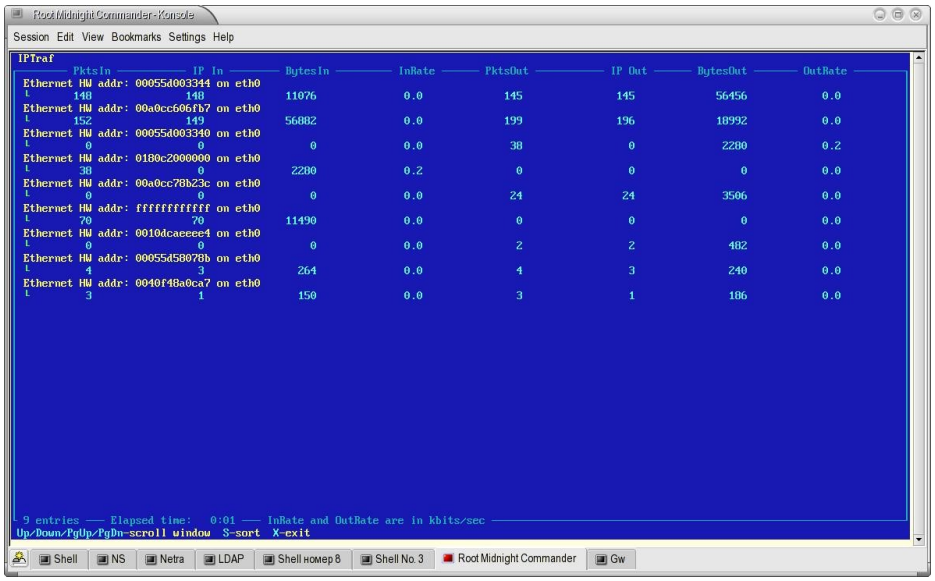


Рисунок 11.86. Режим LAN station monitor.

Счетчики байтов учитывают размер заголовков канального уровня. Скорости потоков данных могут измеряться в кбит/с или Кбайт/с в зависимости от параметров конфигурации (стр. 321) .

Этот режим поддерживается только для интерфейсов **Ethernet**, **PLIP**, **Token Ring** и **FDDI**. Мониторинг интерфейсов **Loopback**, **ISDN**, **SLIP**/**PPP** не поддерживается.

Копия статистики периодически записывается в журнальный файл, если включена опция поддержки записи. По умолчанию для записи статистики используется файл **lan_statistics-*n*.log**, где **n** показывает номер экземпляра такого файла (например, в первый раз будет создан файл **lan_statistics-1.log**).

11.10.1.7.1 Сортировка записей для станций ЛВС

Для сортировки станций ЛВС можно воспользоваться клавишей **S**. На экран будет выведено всплывающее меню выбора ключа сортировки. Клавиша **P** задает сортировку по числу входящих кадров, **I** - по числу входящих пакетов IP, **B** - по количеству принятых байтов, **K** - по числу переданных кадров, **O** - по числу переданных пакетов IP и **Y** - по скорости исходящего потока данных.

11.10.1.8 Фильтры IPTraf

Фильтры позволяют отбирать пакеты, информация о которых учитывается в режиме **IP traffic monitor**, общей и детальной статистики, а также статистики TCP/UDP. Фильтрация пакетов учитывается и при записи результатов в журнальные файлы. С помощью фильтров вы можете избавиться от ненужной информации и сделать результаты более наглядными. Для управления фильтрами служит меню **Filters...**

11.10.1.8.1 Фильтры TCP

Меню **Filters/TCP...** позволяет создавать, редактировать и удалять фильтры трафика TCP. При выборе этой команды появляется меню следующего уровня, используемое для управления фильтрами TCP.

11.10.1.8.1.1 Создание нового фильтра

Команда **Define new filter...** позволяет создать новый фильтр. После выбора команды на экране появляется запрос на ввод имени для нового фильтра, а после указания имени - диалоговое окно задания параметров фильтра (рисунок 11.87).

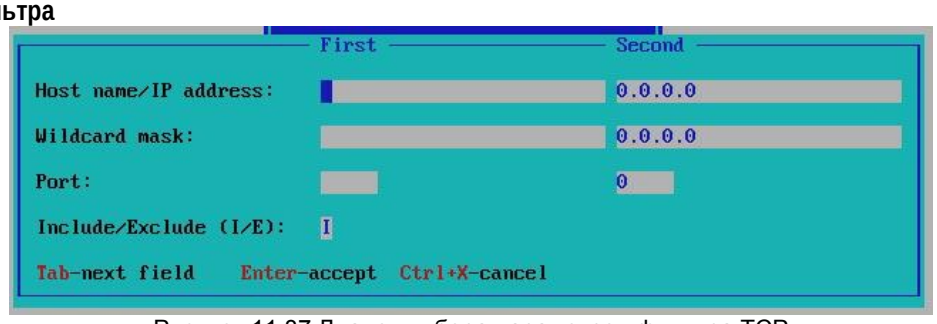


Рисунок 11.87 Диалог выбора параметров фильтра TCP

В этом окне вы можете указать адрес хоста, номер сети или шаблон адресов.

В колонках **First** и **Second** указываются адреса сторон соединения. В первой строке (**Host name/IP Address**) указывается имя хоста или IP-адрес, а во второй (**Wildcard mask**) шаблон маски. Шаблоны похожи на привычные маски подсетей, но несколько отличаются от них, поскольку позволяют задать сравнение с любыми[1] битами адреса IP. Шаблон маски 255.255.255.255 задает точное совпадение адресов, 0.0.0.0 соответствует любому адресу IP.

Поле **Port** позволяет указать номер порта. Значению **0** будут соответствовать пакеты, связанные с любым портом.

1 *Стандартные маски задают сравнение только старших битов адреса (номер сети) со значением маски и игнорирование номера хоста.*

По умолчанию вторая колонка содержит значения **0.0.0.0**, **0.0.0.0**, **0**, которым будут соответствовать любые адреса и номера портов второй стороны соединения. Оставьте эти значения, если вас интересует только одна сторона.

Последняя опция диалогового окна (**Include/Exclude**) определяет отношение к соответствующим фильтру пакетам. Значение **I** будет включать учет соответствующих фильтру пакетов, значение **E** - исключать. По умолчанию установлено значение **I**.

Вы можете задать для фильтра одно или несколько правил.

11.10.1.8.2 Активизация фильтра

После создания фильтра вы можете активизировать его с помощью команды меню **Apply filter...**. После выбора этой команды на экране появится список имеющихся фильтров, среди которых вы можете выбрать нужный. Этот фильтр будет использоваться (даже при завершении работы с программой и повторном ее запуске), пока вы не отключите его.

При активизации фильтра активный ранее фильтр автоматически отключается (т. е., программа использует только один фильтр TCP).

11.10.1.8.3 Редактирование фильтров

Для изменения существующих фильтров служит команда меню **Edit filter...**. После выбора этой команды на экран выводится список имеющихся фильтров. Выберите в списке нужный фильтр и нажмите клавишу **Enter**.

Вы можете изменить имя выбранного фильтра или сразу перейти к его редактированию, нажав клавишу **Enter**. На экран будет выведено диалоговое окно со списком правил выбранного фильтра. Укажите в списке нужное правило и нажмите клавишу **Enter**. На экране появится диалоговое окно редактирования правила, отличающееся от диалога создания фильтров (рисунок 11.87) только тем, что поля окна уже содержат текущие параметры фильтра.

Вы можете добавить в фильтр новое правило, нажав клавишу **I** для включения правила в текущую позицию списка или **A** для добавления правила в конец списка. Клавиша **D** позволяет удалить указанное правило из списка. Не забывайте, что порядок размещения правил в списке может оказывать существенное влияние на результат фильтрации пакетов.

Если вы редактируете активный фильтр, не забудьте повторить процедуру его активизации для учета внесенных изменений.

11.10.1.8.4 Удаление существующего фильтра

Для удаления фильтра выберите в меню команду **Delete filter...** и, указав в списке удаляемый фильтр, нажмите клавишу **Enter**.

11.10.1.8.5 Дезактивация фильтра

Для того, чтобы отключить фильтр выберите в меню команду **Detach filter** и нажмите клавишу **Enter**.

11.10.1.8.6 Фильтры UDP

Поскольку число пакетов UDP обычно достаточно велико, просмотр пакетов определенного типа может потребовать фильтрации, избавляющей от “шума”. Работа с фильтрами UDP обеспечивается с помощью меню **Filters.../UDP...**. Вы можете выбрать просмотр всех пакетов UDP, полностью отключить информацию для UDP или создать свой фильтр для пакетов UDP. Операции с фильтрами UDP не отличаются от соответствующих действий для пакетов TCP, описанных выше.

11.10.1.8.7 Фильтрация прочих пакетов IP

IPTraf поддерживает фильтры и для других протоколов IP на основе IP-адресов получателей и отправителей. Для управления такими фильтрами служит меню **Filters.../Other IP...**, имеющее такой же набор команд, как меню **Filters.../TCP...**.

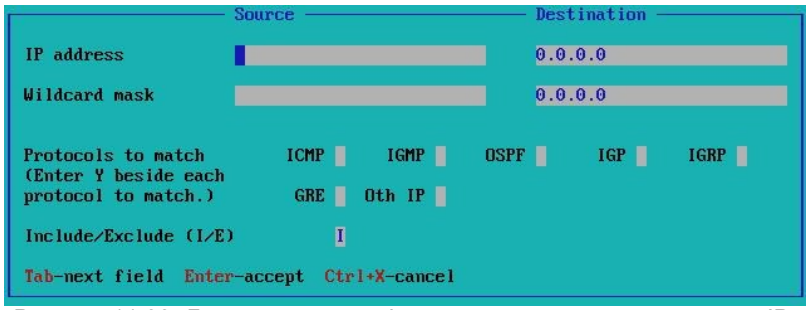


Рисунок 11.88. Диалог создания фильтров для прочих протоколов IP.

Похоже и диалоговое окно задания правил фильтрации, но в этом окне вместо ввода номеров портов вы можете указать идентификаторы протоколов для фильтрации. Введите Y в соответствующем поле для каждого протокола, который вы хотите включить в данное правило фильтрации.

Остальные операции выполняются так же, как это делается для фильтров TCP.

11.10.1.8.8 Фильтры RP, RARP и Non-IP

Для пакетов **non-IP**, **ARP** и **RARP** обеспечивается только возможность включить или отключить весь вывод для каждого из этих протоколов независимо.

11.10.1.9 Меню настройки параметров Iptraf

Параметры работы IPTraf можно настраивать с помощью меню **Configure...** Параметры конфигурации хранятся в файле **/var/local/iptraf/iptraf.cfg**. Если при старте программа не находит этот файла, используются принятые по умолчанию параметры конфигурации.

11.10.1.9.1 Преобразование IP-адресов

Опция **Reverse DNS Lookups** управляет преобразованием адресов IP в доменные имена хостов. При включенной опции программа IPTraf в режиме **IP traffic monitor** будет запускать свой сервер преобразования имен **rvnamed**, обеспечивающий высокоскоростное определение доменных имен в фоновом режиме.

По умолчанию режим преобразования адресов отключен.

11.10.1.9.2 Преобразование номеров портов TCP/UDP

Опция **TCP/UDP Service Names** управляет преобразованием номеров портов TCP/UDP в имена сетевых служб (например, **smtp**, **www**, **pop3**). Отображение выполняется с использованием информации из файла **/etc/services**.

По умолчанию преобразование номеров отключено.

11.10.1.9.3 Режим захвата

Опция **Force promiscuous** переводит интерфейс ЛВС в режим захвата пакетов, в котором устройство считывает из среды все кадры, независимо от того, кому они адресованы. С помощью этой опции вы можете осуществлять мониторинг соединений TCP для всего сегмента локальной сети. В режимах статистики использование этой опции позволяет достоверно оценить уровень сетевого трафика в сегменте локальной сети.

Эта опция управляет режимом захвата для всех интерфейсов ЛВС, которые поддерживают такой режим (устройства Ethernet, FDDI и часть устройств Token Ring).

При завершении работы программы устройство переводится в тот режим, который использовался ранее.

11.10.1.9.4 Color

Эта опция служит для выбора цветного или монохромного режима. Изменение цветового режима произойдет при следующем запуске программы.

11.10.1.9.5 Logging

Эта опция управляет записью журнальных файлов программы IPTraf. При включенной опции программа записывает результаты мониторинга в файл для последующего просмотра и анализа. Для всех режимов мониторинга определены используемые по умолчанию имена журнальных файлов, которые пользователь может при сохранении файла изменить по своему усмотрению. Набор записываемых в журнальный файл сведений зависит от режима использования программы и параметров конфигурации.

11.10.1.9.6 Activity mode

Эта опция управляет индикаторами уровня трафика активности и позволяет выбрать режим вывода в килобитах или килобайтах за секунду (**kbits/s** и **kbytes/s**, соответственно).

По умолчанию скорость выводится в килобитах за секунду.

11.10.1.9.7 Source MAC addrs in traffic monitor

Эта опция управляет выводом в режиме **IP traffic monitor** значений MAC-адресов для интерфейсов Ethernet, FDDI и PLIP. По умолчанию мониторинг MAC-адресов отключен.

11.10.1.9.8 Таймеры программы

Субменю **Timers...** позволяет управлять временными параметрами программы **IPTraf**.

11.10.1.9.8.1 TCP Timeout

Это значение определяет время (в минутах), в течение которого может сохраняться вывод записи для бездействующего соединения. По умолчанию время жизни неактивной записи составляет 15 минут.

11.10.1.9.8.2 Log Interval

Этот параметр задает период (в минутах) записи в журнальный файл статистики интерфейсов, TCP/UDP и ЛВС. По умолчанию запись в файл производится каждые 60 минут.

11.10.1.9.8.3 Screen Update Interval

Этот параметр задает период обновления содержимого экрана (в секундах). По умолчанию используется значение 0, при котором обновление происходит с максимально возможной скоростью для отражения реальной сетевой активности. Однако при запуске программы IPTraf на удаленном хосте столь частое обновление может привести к слишком высокому трафику, связанному с передачей обновлений. В таких случаях целесообразно установить

период обновления информации в несколько секунд.

Этот параметр не влияет на скорость сбора пакетов и определяет лишь частоту обновления собранной программой информации.

11.10.1.9.8.4 TCP closed/idle persistence

Этот параметр задает интервал (в минутах) удаления в режиме **IP Traffic Monitor** записей о соединениях TCP, которые были закрыты, бездействуют или находятся в состоянии **timeout**. При нулевом значении параметра такие записи не будут удаляться, пока не появятся новые соединения.

Опция **TCP timeout...** не определяет напрямую время присутствия записи на экране, поскольку задает лишь время, по истечении которого IPTraf будет считать соединение бездействующим. Такие записи удаляются программой по истечении периода, заданного параметром **closed/idle persistence...**

11.10.1.9.9 Additional ports...

Эта команда позволяет вам включить в список мониторинга служб (параграф 11.10.1.6.2 на стр. 318) дополнительные порты TCP/UDP с номерами выше 1023. Вы можете использовать эту опцию неоднократно для добавления множества портов или диапазонов.

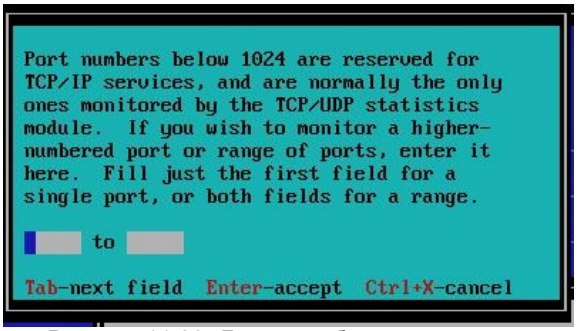


Рисунок 11.89. Диалог добавления портов.

11.10.1.9.10 Delete port/range...

Эта команда позволяет удалить добавленный ранее порт или диапазон портов из списка мониторинга TCP/UDP служб (параграф 11.10.1.6.2 на стр. 318).

11.10.1.9.11 Идентификаторы станций ЛВС

Команды **Ethernet/PLIP host descriptions...** и **FDDI/Token Ring host descriptions...** в меню конфигурации позволяют создать краткие записи, описывающие станции с указанным MAC-адресом. Указанная здесь информация о станции будет выводиться программой вместе с MAC-адресом в режиме **LAN station monitor** (параграф 11.10.1.7 на стр. 318).

IPTraf 2.4 также может читать информацию из файла **/etc/ethers**.

11.10.2 Conntrack Viewer

http://cv.intellos.net/

Программа **Conntrack Viewer** - это простой сценарий на языке **perl** для просмотра маскируемых соединений в системах Linux с ядром версии 2.4.x и выше. Для получения информации о соединениях используется файл **/proc/net/ip_conntrack**.

В системах с ядром 2.2.x для просмотра маскируемых соединений можно было использовать команду **netstat** с ключом **-M** или -**masquerade**. Однако в новых версиях ядра при использовании такой команды вы просто получите сообщение

```
netstat: no support for `ip_masquerade' on this system
```

Начиная с версии 2.4, информация о маскируемых соединениях хранится в файле **/proc/net/ip_conntrack**.
Свободно распространяемая программа Conntrack Viewer поможет вам решить эту проблему. Выводимые программой сведения о соединениях понятны без дополнительных пояснений.

```
./conntrack-viewer
Active Connections according to /proc/net/ip_conntrack
Proto   Source Address       Remote Address     Service     State          Name Resolution
tcp     192.168.0.3:1708     207.6.235.85:1214  kazaa       ESTABLISHED    simba > f.bc.hsia.telus.net
tcp     192.168.0.3:1717     192.168.1.103:1214 kazaa       SYN_SENT       simba > UNRESOLVED!
tcp     192.168.0.3:1721     192.168.1.100:1214 kazaa       SYN_SENT       simba > UNRESOLVED!
Tcp     192.168.0.3:1373     68.10.104.11:1214  kazaa       ESTABLISHED    simba > a4-11.hr.hr.cox.net
tcp     192.168.0.3:1030     64.12.25.116:5190  icq         ESTABLISHED    simba > UNRESOLVED!
tcp     192.168.0.3:1718     24.66.255.215:1214 kazaa       TIME_WAIT      simba > a5.ss.shawcable.net
tcp     192.168.0.3:1730     216.191.240.2:110  pop3        TIME_WAIT      simba > comnet.ca
tcp     192.168.0.3:1731     216.191.240.2:110  pop3        TIME_WAIT      simba > comnet.ca
tcp     192.168.0.3:1720     192.168.2.31:1214  kazaa       SYN_SENT       simba > UNRESOLVED!
tcp     213.233.73.121:32871 64.39.176.22:80    http        ESTABLISHED    d1.xnet.ro > 22.comnet.ca
```

Опция **-n** позволяет отказаться от попыток определения имен хостов, что может существенно ускорить работу программы при большом количестве соединений.

```
./conntrack-viewer -n
Active Connections according to /proc/net/ip_conntrack
Proto   Source Address          Remote Address          Service     State
tcp     192.168.0.3:1708        207.6.235.85:1214       kazaa       ESTABLISHED
tcp     192.168.0.3:1717        192.168.1.103:1214      kazaa       SYN_SENT
tcp     192.168.0.3:1721        192.168.1.100:1214      kazaa       SYN_SENT
tcp     192.168.0.3:1373        68.10.104.11:1214       kazaa       ESTABLISHED
tcp     192.168.0.3:1030        64.12.25.116:5190       icq         ESTABLISHED
tcp     192.168.0.3:1718        24.66.255.215:1214      kazaa       TIME_WAIT
```

```
tcp      192.168.0.3:1730         216.191.240.2:110        pop3         TIME_WAIT
tcp      192.168.0.3:1731         216.191.240.2:110        pop3         TIME_WAIT
tcp      192.168.0.3:1720         192.168.2.31:1214        kazaa        SYN_SENT
tcp      213.233.73.121:32871     64.39.176.22:80          http         ESTABLISHED
```

11.10.3 Etherape

http://etherape.sourceforge.net[1]

Программа **etherape** обеспечивает удобный графический интерфейс на базе библиотек GNOME для просмотра активных сеетвых соединений. Для сбора и фильтрации пакетов **etherape** использует функции библиотеки libpcap (см. параграф 11.9.1 на стр. 261). Программа представляет сетевые соединения в виде секторов, окрашенных в зависимости от протокола. Угол сектора обращен в сторону передающего узла, а ширина сектора пропорциональна текущему уровню трафика. Таким образом программа позволяет отслеживать динамику соединений.

Синтаксис

```
etherape  [<опции командной строки>]
```

11.10.3.1 Опции командной строки

-m (--mode) <ethernet|fddi|ip|tcp>
задает режим работы программы. По умолчанию используется максимально низкий уровень, поддерживаемый для текущего устройства.

-i (--interface) <интерфейс>
задает имя интерфейса для сбора пакетов.

-f (--filter) <спецификация фильтра>
задает имя фильтра, используемого при сборе пакетов.

-r (--infile) <файл>
задает имя файла, сдержащего собранные ранее пакеты.

-n (--numeric)
отключает преобразование адресов в имена хостов.

-d (--diagram-only)
отключает вывод каких бы то ни было текстовых сведений об узлах сети.

-F (--no-fade)
отключает функцию удаления с экрана старых соединений.

-N (--node-color) <цвет>
задает цвет для вывода узлов сети.

-L (--link-color) <цвет>
задает цвет для вывода соединений.

-T (--text-color) <цвет>
задает цвет текста.

-? (--help)
выводит краткую справку о работе с программой.

Файл /etc/ethers может содержать список пар “адрес-имя” типа

1 *Исходные тексты etherape вы найдете в каталоге SRC/ приложенного к книге компакт-диска.*

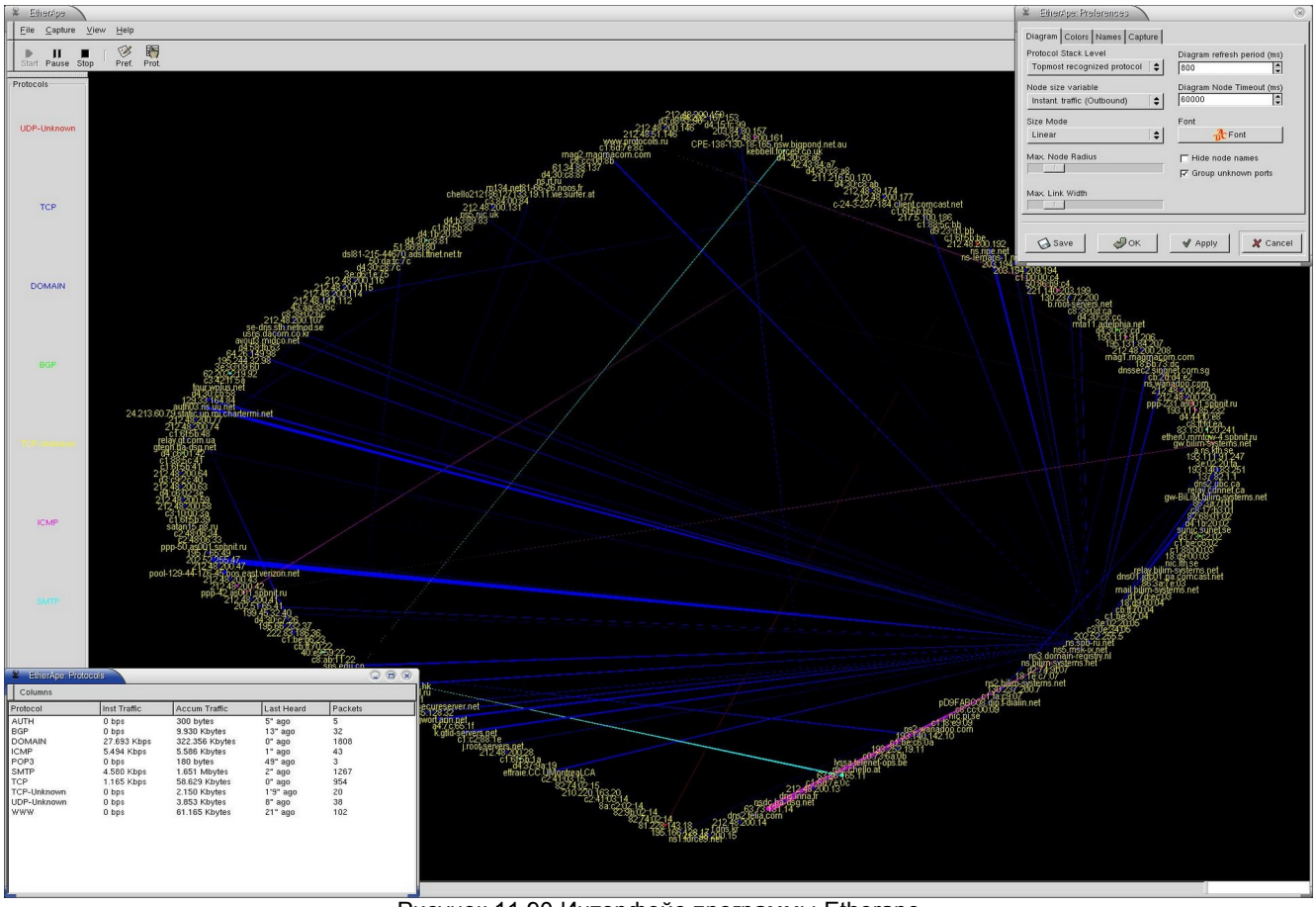


Рисунок 11.90 Интерфейс программы Etherape

```
00:40:33:35:80:5F LAZARO
00:40:33:35:80:6D NEBAJ
00:C0:26:A2:58:FE ARGOS
```

Если этот файл отсутствует, Etherape будет пытаться использовать протокол RARP для определения адресов IP. При работе со шлюзом в режиме **ethernet** наличие в файле /etc/ethers записи для интерфейса шлюза играет важную роль, поскольку при отсутствии такой записи имя и адрес шлюза могут отображаться некорректно.

Недостатком программы является полное отсутствие документации, но прочтой интерфейс и наличие исходных кодов позволяют разобраться с программой и без описания.

11.10.4 MRTG

Простая в эксплуатации и удобная программа MRTG[1] обеспечивает сбор статистики загрузки сетевых каналов и генерацию страниц HTML с иллюстрациями в формате PNG для просмотра этой статистики. Страницы включают теги автоматического обновления, что позволяет вести мониторинг сетевых устройств со станции управления.

MRTG включает сценарий на языке Perl, использующий протокол SNMP для сбора информации от маршрутизаторов и хостов, а также простую программу **rateup**, записывающую собранные сведения в журнальные файлы и генерирующую графическое представление уровня загрузки сетевых интерфейсов. Графики встраиваются в гипертекстовые страницы, которые можно просматривать с помощью любого Web-браузера.

Кроме сбора информации о текущем состоянии MRTG генерирует усредненную статистику за различные периоды и также строит графики на основании полученных данных. Для расчета усредненных данных MRTG использует журнальные файлы с данными, полученными по протоколу SNMP. На рисунке 11.91 вы можете видеть статистику трафика через один из интерфейсов коммутатора, собранную с 5-минутными интервалами и усредненную за полчаса, 2 часа и сутки. Графика с суточным усреднением показывает картину трафика за период более 12 месяцев, позволяя увидеть тенденции изменения картины сетевого трафика.

1 *Multi Router Traffic Grapher*

Для сбора и представления данных используются конфигурационные файлы (отдельный файл для каждого хоста или маршрутизатора), для создания которых можно использовать утилиту cfgmaker, включенную в пакет MRTG. Для автоматического создания конфигурационного файла достаточно ввести команду

```
cfgmaker <хост> >
<конфигурационный файл>
```

Программа MRTG может обеспечивать не только мониторинга сетевого трафика, но и сбор значений любый переменных SNMP. Обеспечивается также возможность использования для сбора данных независимых программ и мониторинга собранной информации с помощью MRTG. Более того, MRTG позволяет вывести на один график информацию из разных источников.

При наличии большого числа источников данных для их объединения поможет утилита **indexmaker**, создающая индексную страницу HTML на основе анализа конфигурационных файлов MRTG.

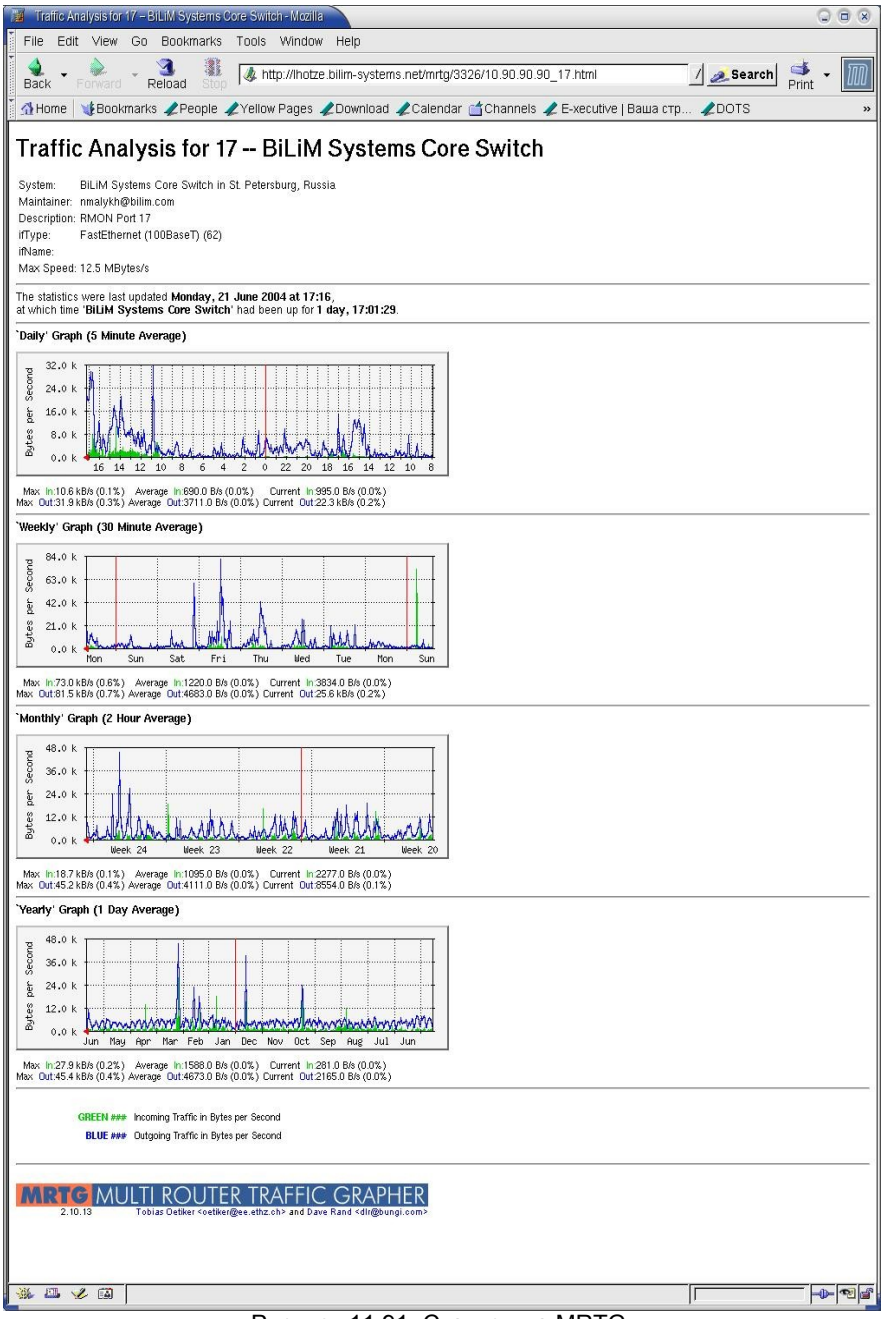


Рисунок 11.91. Статистика MRTG.

11.10.4.1 Опции командной строки

Большинство параметров работы MRTG задается в конфигурационных файлах. Опции командной строки перечислены в таблице 84.

Таблица 84. Опции команды mrtg.

Опция	Описание
`--user <имя>`	Задает работу программы от имени заданного пользователя.
`--group <имя>`	Задает работу программы от имени заданной группы.
`--lock-file <файл>`	Задает использование указанного файла блокировки взамен принятого по умолчанию (имя конфигурационного файла, к которому добавлен суффикс **_l**).
`--confcache-file <файл>`	Задает использование указанного файла confcache (по умолчанию - имя конфигурационного файла с расширением **.ok**)
`--logging <файл>`	Если в качестве параметра этой опции передано имя открытого для записи файла, весь вывод **mrtg** (предупреждения, отладочные сообщения, сообщения об ошибках) будет записываться в указанный файл.
`--daemon`	Задает работу **MRTG** в режиме демона. Эта опция может потребоваться для корректной работы программы в режиме FHS, поскольку каталог **/var/run** открыт для записи только пользователю **root**.
`--fhs`	Задает все пути **mrtg** в соответствии со спецификацией FHS[1].

1 *Filesystem Hierarchy Standard - стандарт для иерархии файловых систем. Информацию об этом стандарте вы сможете найти на сайте http://www.pathname.com/fhs/. Копия спецификации имеется также в каталоге Documents/ приложенного к книге компакт-диска.*

Опция	Описание
--check	Проверяет конфигурационный файл, не выполняя никаких дополнительных операций.
--pid-file=	Задает полное имя pid-файла при работе **mrtg** в режиме демона.
--debug=	Разрешает использование опций отладки. Параметром данной опции служит список разделенных запятыми опций отладки: **cfg** - отслеживать чтение конфигурационных файлов; **dir** - отслеживать изменение каталогов; **base** - основной поток программы; **tarp** - разбор цели; **snpo** - опрос SNMP; **fork** - показывать ветвление процессов; **time** - выводить информацию о времени; **log** - запись данных в журнальный файл с помощью **rateup** или **rrdtool**.

11.11 Генераторы трафика

11.11.1 Встроенный генератор пакетов Linux

Современные версии ядра Linux включают генератор пакетов, позволяющий создавать и передавать сетевые пакеты без использования дополнительных программ.

Для использования встроенного генератора потребуется ядро со включенной при компиляции опцией **NET_PKTGEN** (параграф 4.4.2.2.40.1 на стр. 91). Если функции генератора пакетов реализованы в виде модуля[1] нужно загрузить модуль с помощью команды

```
modprobe pktgen
```

Подготовьте сценарий генерации пакетов по типу показанного на рисунке 11.92, указав в нем имя интерфейса для генерации пакетов и IP-адрес получателя этих пакетов. Вы можете создать множество сценариев такого типа для разных вариантов генерации пакетов. Одновременно можно использовать до 32 процессов **pktgen** с различными сценариями генерации. Для каждого процесса параметры задаются в соответствующем файле **/proc/net/pktgen/pg***.

```
#! /bin/sh
modprobe pktgen
PGDEV=/proc/net/pktgen/pg0
function pgset() {
    local result
    echo $1 > $PGDEV
    result=`cat $PGDEV | fgrep "Result: OK:"`
    if [ "$result" = "" ]; then
         cat $PGDEV | fgrep Result:
    fi
}
function pg() {
    echo inject > $PGDEV
    cat $PGDEV
}
pgset "odev eth0"
pgset "dst 0.0.0.0"
#поместите ниже команды управление генерацией
```

Рисунок 11.92 Сценарий для работы с pktgen

Включите в сценарий функции управления генерацией пакетов. Функция **pg** служит для активизации процесса генерации пакетов и вывода результатов, а функция **pgset** задает параметры генерации пакетов. Функция **pgset** принимает в качестве параметра заключенную в кавычки строку опций генерации пакетов. Варианты опций генерации перечислены в таблице .

Таблица 85 Параметры генерации пакетов

Параметр	Описание
clone_skb 100	Задает число передаваемых копий пакета до того, как будет создан новый пакет.
clone_skb 0	Задает использование множества буферов SKB (см. Приложение) для генерации пакетов.
pkt_size 9014	Задает размер пакета в 9014 байтов.
frags 5	Задает разбиение пакета на 5 фрагментов.
count 200000	Задает количество передаваемых пакетов. Нулевое значение счетчика обеспечивает генерацию неограниченного числа пакетов, пока работа не будет прервана явно.
ipg 5000	Задает интервал между пакетами в 5000 наносекунд.
dst 10.0.0.1	Задает адрес получателя пакетов.
dst_min 10.0.0.1	Задает нижнюю границу диапазона адресов получателей.
dst_max 10.0.0.254	Задает верхнюю границу диапазона адресов получателей.
src_min 10.0.0.1	Задает нижнюю границу диапазона адресов отправителей.
src_max 10.0.0.254	Задает верхнюю границу диапазона адресов отправителей.

1 *Лучше поступить именно так, если вы не создаете специализированную систему для генерации пакетов.*

Параметр	Описание
dstmac 00:00:00:00:00:00	Устанавливает MAC-адрес получателя.
srcmac 00:00:00:00:00:00	Устанавливает MAC-адрес отправителя.
src_mac_count 1	Задает количество MAC-адресов отправителя в используемом диапазоне с начальным значением **srcmac**.
dst_mac_count 1	Задает количество MAC-адресов получателей в используемом диапазоне с начальным значением **dstmac**.
flag [name]	Задает флаг управления генерацией. Возможны значения флагов: IPSRC_RND - использовать случайный адрес IP для отправителя; IPDST_RND - использовать случайный адрес IP для получателя; UDPSRC_RND - использовать случайный порт UDP для отправителя; UDPDST_RND - использовать случайный порт UDP для получателя; MACSRC_RND - использовать случайный MAC-адрес для отправителя; MACDST_RND - использовать случайный MAC-адрес для получателя. Адреса выбираются из диапазона, заданного минимальным и максимальным значением.
udp_src_min 9	Задает минимальный номер порта UDP для отправителя.
udp_src_max 9	Задает максимальный номер порта UDP для отправителя.
udp_dst_min 9	Задает минимальный номер порта UDP для получателя.
udp_dst_max 9	Задает максимальный номер порта UDP для получателя.
stop	Останавливает генерацию пакетов. Этот параметр передается без кавычек.

Если минимальный номер порта больше максимального, используются номера портов из всего диапазона возможных значений, за исключением интервала (**max**, **min**).

11.11.2 Hping2

www.hping.org

Программа **hping2** может генерировать почти любые пакеты TCP/IP, адресованные указанному в командной строке хосту. Программа распространяется свободно на условиях лицензии GPL[1].

Программа **hping2** является мощным инструментом тестирования сетей и может передавать пакеты IP с заданными параметрами, выводя на экран полученные от адресата отклики, подобно программам, работающим с откликами ICMP. **Hping2** поддерживает фрагментацию, позволяет задавать произвольное содержимое поля данных пакета, менять размер пакетов и может использоваться для передачи файлов, инкапсулированных с использованием поддерживаемых протоколов. Используя **hping2**, вы сможете решить множество задач, включая:

- тестирование межсетевых экранов;
- сканирование портов с широким выбором вариантов сканирования;
- проверка производительности сети для различных протоколов;
- проверка передачи пакетов с различным размером и TOS (тип обслуживания);
- проверка передачи фрагментированных пакетов;
- определение Path MTU;
- передача файлов даже через враждебно настроенные брандмауэры;
- трассировка пакетов для различных протоколов;
- определение ОС удаленных хостов;
- аудит стека TCP/IP.

Кроме того, программа может оказать существенную помощью при изучении протоколов стека TCP/IP.

Синтаксис

```
hping2 [-hvnqVDzZ012WrfxykQbFSRPAUXYjJBuTG] [-c count] [-i wait] [--fast] [-I interface]
[-9 signature] [-a host] [-t ttl] [-N ip id] [-H ip protocol] [-g fragoff] [-m mtu]
[-o tos] [-C icmp type] [-K icmp code] [-s source port] [-p[+][+] dest port][-w tcp
window] [-O tcp offset] [-M tcp sequence number] [-L tcp ack] [-d data size] [-E
filename]
[-e signature] [--icmp-ipver version] [--icmp-iphlen length] [--icmp-iplen length]
[--icmp-ipid id] [--icmp-ipproto protocol] [--icmp-cksum checksum] [--icmp-ts]
[--icmp-addr] [--tcpexitcode] [--tcp-timestamp] [--tr-stop] [--tr-keep-ttl] [--tr-no-rtt]
[--rand-dest] [--rand-source] [--beep] hostname
```

1 *Исходные тексты hping2 вы сможете найти в каталоге SRC/ приложенного к книге компакт-диска.*

11.11.2.1 Опции

11.11.2.1.1 Опции общего назначения

Таблица 86. Опции hping2 общего назначения.

Опция		Описание
-h	**--help**	Выводит на экран краткую справку о работе с программой.
-v	**--version**	Выводит на экран номер версии программы и сведения об используемых API
-c	**--count**	Задает прекращение работы программы после передачи или приема заданного числа пакетов. После передачи последнего пакета hping2 будет ждать отклика в течение заданного в секундах периода **COUNTREACHED_TIMEOUT**. Значение периода ожидания можно установить, отредактировав заголовочный файл **hping2.h**.
-i	**--interval**	Задает период повтора передачи пакетов в секундах или микросекундах (префикс u перед значением интервала). По умолчанию интервал передачи составляет 1 секунду. При использовании **hping2** для передачи файлов эта опция оказывает существенное влияние на реальную скорость передачи. Существенное влияние период передачи может оказывать и на сканирование в режимах idle/spoofing.
	--fast	Псевдоним для **-i u10000**, обеспечивающий передачу пакетов с периодом 10 мсек.
	--faster	Псевдоним для **-i u1**, обеспечивающий передачу пакетов с периодом 1 мксек.
	--flood	Задает передачу пакетов с максимально возможной скоростью без ожидания приема откликов.
-n	**--numeric**	Отключает преобразование IP-адресов в символьные имена.
-q	**--quiet**	Отключает вывод всей информации за исключением стартовой и финишной строки.
-I	**--interface**	Указывает программе **hping2** интерфейс, который будет использоваться для передачи пакетов[1]. В системах Linux и BSD программа **hping2** по умолчанию передает пакеты в интерфейс, используемый для принятого по умолчанию маршрута. В других системах и при отсутствии принятого по умолчанию маршрута **hping2** будет использовать первый реальный (не loopback) интерфейс.

-V	**--verbose**	Задает вывод максимального количества информации. Отклики TCP при использовании этой опции имеют вид:
		`len=46 ip=192.168.1.1 flags=RA DF seq=0 ttl=255 id=0 win=0`
		`rtt=0.4 ms tos=0 iplen=40 seq=0 ack=1380893504 sum=2010 urp=0`

-D	**--debug**	Включает режим отладки, позволяющий отыскать и разрешить некоторые проблемы, возникающие при работе с **hping2**. При включенной отладке программа выводит дополнительную информацию об интерфейсе и его параметрах, канальном уровне, разборе опций командной строки, фрагментации и др.
-z	**--bind**	Позволяет менять для передаваемых пакетов значение поля TTL с помощью клавиш **CTRL+Z**.
-Z	**--unbind**	Отключает возможность изменения времени жизни пакетов с помощью **CTRL+Z**.
	--beep	Включает подачу звукового сигнала при поступлении каждого входящего пакета (за исключением сообщений ICMP об ошибках).

11.11.2.1.2 Опции выбора протокола

По умолчанию программа генерирует пакеты TCP, адресованные в порт 0, использующие окно размером 64 и не имеющие каких-либо флагов TCP. Такие пакеты могут быть весьма полезны для скрытого “прощупывания” удаленного хоста, особенно если последний спрятан за брандмауэром, отбрасывающим пакеты ICMP. Более того, адресованные в порт 0 пакеты без опций зачастую не протоколируются в журнальных файлах проверяемой системы.

Таблица 87. Опции hping2 для выбора протокола.

Опция		Описание
-0	**--rawip**	Режим **RAW IP**, при котором **hping2** будет передавать заголовки IP с прицепленными к ним данными, указанными опцией **--signature** и/или -**file**. Опция **--ipproto** дополнительно позволяет задать значение поля протокола IP.
-1	**--icmp**	Режим **ICMP** - по умолчанию программа будет передавать пакеты **ICMP echo-request**, но вы можете выбрать тип и код ICMP с помощью опций **--icmptype** и **--icmpcode**.
-2	**--udp**	Режим **UDP** - по умолчанию **hping2** будет передавать пакеты UDP, адресованные в порт 0. Для управления параметрами заголовков UDP могут служить опции -**baseport**, **--destport, --keep.**

1 *Имя интерфейса не обязательно указывать полностью - hping2 умеет находить интерфейсы по частично заданному имени. Если программа не сможет найти указанный интерфейс, пакеты будут передаваться в интерфейс **lo**.*

Опция		Описание
-8	**--scan**	Режим **сканирования.** Эта опция должна использоваться с аргументом, описывающим группу портов для сканирования. Номера портов в группе разделяются запятыми, при сканировании непрерывного множества портов можно указывать границы диапазона, используя в качестве разделителя дефис (-). Ключевое слово **all** задает сканирование всех портов в диапазоне от 0 до 65535, а ключевое слово **known** задает сканирование всех портов из файла **/etc/services**. При задании группы портов можно использовать все варианты одновременно. Например, команда **hping --scan 1-1000,8888,known** обеспечивает сканирование портов диапазона 1 - 1000, порта 8888 и портов, указанных в файле **/etc/services**. Допускается при задании группы портов использовать знак инверсии (!). Например, команда **hping --scan '1-1024,!known'** обеспечивает сканирование всех портов диапазона 1-1024, за исключением портов, указанных в файле **/etc/services**. Программа **hping** в этом режиме похожа на обычный сканер портов, однако вы сохраняете возможности использования других опций. Например, для SYN-сканирования вы можете задать в командной строке опцию **-S**. Обеспечивается возможность изменения размера окна TCP, времени жизни пакетов (TTL), фрагментации IP и т. п. В отличие от многих сканеров **hping** выводит некоторые интересные сведения о принятых пакетах, включая **IP ID**, **TCP win**, **TTL** и т. п. Не забывайте принимать во внимание эту дополнительную информацию.
-9	**--listen**	Режим **прослушивания** - hping2 ждет пакеты, соответствующие заданной сигнатуре (параметр опции) и выводит дамп таких пакетов от завершения сигнатуры до конца пакета.

11.11.2.1.3 Опции IP

Таблица 88. Опции hping2 для протокола IP.

Опция		Описание
-a	**--spoof**	Эта опция позволяет указывать в передаваемых пакетах подставной адрес отправителя, заданный параметром опции. В таких случаях следует помнить, что отклики на ваши пакеты также будут передаваться на подставной адрес и вы не сможете их увидеть. Режим сканирования **spoofed/idle**[1] позволяет решить эту проблему.
	--rand-source	Эта опция позволяет использовать для передаваемых пакетов случайные адреса отправителя. Такой режим может быть полезен при тестировании правил межсетевых экранов и других таблиц, основанных на адресах отправителей.
	--rand-dest	Эта опция обеспечивает генерацию пакетов со случайным адресом получателя. Адреса выбираются из диапазона, заданного в командной строке. Например, команда **hping 10.0.0.x -rand-dest** будет генерировать пакеты, адресованные хостам сети 10.0.0.0/24, а команда **hping x.x.x.x -rand-dest** будет использовать все адресное пространство IPv4. Если вам трудно определить диапазон адресов, попытайтесь использовать опцию **--debug**. При использовании этой опции отклики будут приниматься от всего диапазона адресатов. Отметим, что при использовании этой опции **hping** не сможет корректно определить интерфейс для передачи пакетов и вам следует использовать опцию **-i** для указания выходного интерфейса.
-t	**--ttl**	Задает время жизни генерируемых пакетов (TTL). Эта опция весьма полезна при совместном использовании с опциями **--traceroute** и **--bind**.
-N	**--id**	Задает значение идентификатора IP ID. По умолчанию программа использует случайные значения поля идентификации IP или ID = getpid() & 0xFF (при включенной фрагментации).
-H	**--ipproto**	Переводит стек IP в режим RAW.
-W	**--winid**	Обеспечивает корректное отображение идентификаторов IP для откликов от систем Windows.
-r	**--rel**	Задает вывод значений инкремента идентификаторов вместо самих идентификаторов (см. HPING2-HOWTO[2]). Отметим, что значение инкремента не рассчитывается просто как id[N] - id[N-1], а учитывает потерю пакетов.
-f	**--frag**	Задает режим фрагментации пакетов, который может быть полезен для тестирования стека IP или проверки пакетных фильтров. По умолчанию используются фрагменты размером 16 байтов. Для установки другого размера фрагментов можно использовать опцию --mtu (см. ниже).

1 *Описание этого режима приводится в документе HPING2-HOWTO, который вы найдете на приложенном к книге компакт-диске.*
2 *Вы сможете найти этот документ на приложенном к книге компакт-диске.*

Опция		Описание
-x	**--morefrag**	Устанавливает флаг наличия других фрагментов (**more fragments**)IP. Эта опция полезна в тех случаях, когда хотите получать от тестируемого хоста сообщения **ICMP time-exceeded** в процессе сборки фрагментов.
-y	**--dontfrag**	Устанавливает флаг запрета фрагментирования. Эту опцию можно использовать для определения **Path MTU**.
-g	**--fragoff**	Задает смещение фрагмента.
-m	**--mtu**	Задает значение виртуального MTU, отличное от 16, для использования при включенном режиме фрагментации. Пакеты будут делиться на фрагменты автоматически, если их размер превышает значение виртуального MTU.
-o	**--tos**	Задает шестнадцатеричное значение поля TOS (Type Of Service - тип обслуживания.). Перечень возможных значения можно получить по команде **hping --tos help**.
-G	**--rroute**	Задает режим записи маршрута (Record route). В каждом передаваемом пакете устанавливается флаг **RECORD_ROUTE**, а для возвращенных пакетов выводится содержимое буфера маршрута. Отметим, что размеров заголовка IP достаточно лишь для 9 маршрутных записей и все последующие хосты будут игнорировать или отбрасывать эту опцию. При использовании hping запись маршрута возможна даже в тех случаях, когда тестируемый хост фильтрует пакеты ICMP. Опция записи маршрута относится к протоколу IP, а не ICMP, поэтому запись маршрута возможна даже в режимах TCP и UDP.

11.11.2.1.4 Опции ICMP

Таблица 89. Опции hping2 для протокола ICMP.

Опция		Описание
-C	**--icmptype**	Задает для генерируемых пакетов тип ICMP (по умолчанию - echo request).
-K	**--icmpcode**	Задает для генерируемых пакетов код ICMP (по умолчанию - 0).
	--icmp-ipver	Задает номер версии IP в заголовке пакета IP, содержащемся в поле данных ICMP (по умолчанию - 4).
	--icmp-iphlen	Задает размер заголовка пакета IP, содержащегося в поле данных ICMP (по умолчанию - 5; 5 слов по 32 бита = 20 байтов).
	--icmp-iplen	Задает значение поля размер пакета IP для заголовка IP, содержащегося в поле данных ICMP. По умолчанию используется реальный размер пакета.
	--icmp-ipid	Задает значение поля IP ID для заголовка IP, содержащегося в поле данных ICMP. По умолчанию используется случайное значение.
	--icmp-ipproto	Задает значение поля протокола IP для заголовка IP, содержащегося в поле данных ICMP (по умолчанию - TCP).
	--icmp-cksum	Задает контрольную сумму пакета ICMP (по умолчанию используется реальное значение контрольной суммы).
	--icmp-ts	Псевдоним для **--icmptype 13** (передача запросов **ICMP timestamp**).
	--icmp-addr	**Псевдоним для --icmptype 17** (передача запросов **ICMP address mask**).

11.11.2.1.5 Опции TCP/UDP

Таблица 90. Опции hping2 для протоколов TCP/UDP.

Опция		Описание
-s	**--baseport**	Программа **hping2** использует порт отправителя для предсказания порядковых номеров откликов. Нумерация начинается с заданного этим параметром значения и увеличивается на такую же величину для каждого передаваемого пакета. При получении пакета порядковый номер может быть рассчитан как **replies.dest.port - base.source.port**. По умолчанию для базового порта отправителя используется случайный номер и данная опция позволяет задать желаемое значение номера порта. Если вы не хотите, чтобы номер порта увеличивался для каждого переданного пакета, используйте опцию **-k** (**--keep**).
-p	**--destport**	**[+][+]dest port** Эта опция задает порт получателя (по умолчанию - 0). Если номер порта задан с префиксом **+** (например, +1024) номер порта получателя будет увеличиваться с каждым принятым откликом, а префикс **++** задает увеличение номера порта для каждого передаваемого пакета. По умолчанию порт получателя можно менять в интерактивном режиме с помощью клавиш **CTRL+z**.
	--keep	Сохраняет номер порта отправителя для передаваемых пакетов (см. **--baseport**).
-w	**--win**	Задает размер окна TCP (по умолчанию - 64).

Опция		Описание
-O	**--tcpoff**	Задает смещение данных в пакете TCP. Обычно используется смещение в четверть размера заголовка TCP[1].
-M	**--tcpseq**	Задает порядковый номер TCP.
-L	**--tcpack**	Задает передачу подтверждений TCP ACK.
-Q	**--seqnum**	Эта опция может использоваться для сбора порядковых номеров, генерируемых проверяемым хостом - это полезно при анализе предсказуемости порядковых номеров. Ниже показан пример вывода по команде **hping2 --seqnum**: **HPING uaz (eth0 192.168.4.41): S set, 40 headers + 0 data bytes** **2361294848 +2361294848** **2411626496 +50331648** **2545844224 +134217728** **2713616384 +167772160** **2881388544 +167772160** **3049160704 +167772160** **3216932864 +167772160** **3384705024 +167772160** **3552477184 +167772160** **3720249344 +167772160** В первой колонке указывается порядковый номер, а во второй разница порядковых номеров с предыдущим пакетом.
-b	**--badcksum**	Задает передачу пакетов UDP/TCP с некорректной контрольной суммой.
	--tcp-timestamp	Включает опцию **TCP timestamp** и пытается предсказать частоту обновления временных меток и время работы проверяемого хоста (uptime).
-F	**--fin**	Устанавливает флаг TCP **FIN**.
-S	**--syn**	Устанавливает флаг TCP **SYN**.
-R	**--rst**	Устанавливает флаг TCP **RST**.
-P	**--push**	Устанавливает флаг TCP **PUSH**.
-A	**--ack**	Устанавливает флаг TCP **ACK**.
-U	**--urg**	Устанавливает флаг TCP **URG**.
-X	**--xmas**	Устанавливает флаг TCP **Xmas**[2].
-Y	**--ymas**	Устанавливает флаг TCP **Ymas**[3].

11.11.2.1.6 Опции для всех протоколов

Таблица 91. Опции hping2 для всех протоколов.

Опция		Описание
-d	**--data**	Задает размер генерируемых пакетов без учета заголовков. **Hping2** показывает размер генерируемых пакетов в первой строке вывода.
-E	**--file**	Задает использование указанного файла в качестве источника информации для поля данных генерируемых пакетов.
-e	**--sign**	Задает включение указанной в качестве параметра сигнатуры в начало поля данных генерируемых пакетов. Если размер сигнатуры превышает размер поля данных, выдается сообщение об ошибке. Если вы не указали размер поля данных, **hping** будет создавать пакеты с размером заданной сигнатуры (без учета заголовков). Эта опция может без риска пользоваться вместе с опцией **--file** - оставшаяся после включения сигнатуры часть поля данных пакета будет заполнена информацией из файла.
-j	**--dump**	Задает вывод шестнадцатеричного дампа принятых пакетов.
-J	**--print**	Задает вывод дампа ASCII для принятых пакетов.
-B	**--safe**	Включает безопасный протокол передачи файлов, который обеспечивает повторную передачу потерянных пакетов. Например, для передачи файла с хоста A на хост B **/etc/passwd** на хосте A следует ввести команду **hping2 host_b --udp -p 53 -d 100 --sign signature --safe --file /etc/passwd** а на хосте B **hping2 host_a --listen signature --safe --icmp**

1 *Это связано с тем, что величина смещения задается в 4-байтовых словах.*
2 *Нестандартный флаг 0x40.*
3 *Нестандартный флаг 0x80.*

Опция		Описание
-u	**--end**	При использовании вместе с опцией **--file** эта опция обеспечивает выдачу сообщения при достижении конца файла (EOF).
-T	**--traceroute**	Включает режим трассировки (**Traceroute**). При использовании этой опции **hping2** будет увеличивать значение TTL всякий раз при получении от промежуточных узлов сообщения ICMP о достижении нулевого значения TTL. Эта опция неявно предполагает наличие опций **--bind** и **--ttl 1**. Вы можете указать иное время жизни пакетов с помощью опции **--ttl**. Начиная с версии 2.0.0 программа обеспечивает при использовании данной опции вывод значений RTT[1].
	--tr-keep-ttl	Сохраняет фиксированное значение TTL для режима **traceroute**, что позволяет осуществлять мониторинг одного интервала (hop) используемого маршрута. Например, для контроля пятого интервала и изменений RTT можно воспользоваться командой **hping2 host --traceroute --ttl 5 --tr-keep-ttl**.
	--tr-stop	Эта опция задает завершение работы при получении первого пакета, отличного от сообщения **ICMP time exceeded**. Это позволяет эмулировать поведение утилиты **traceroute**.
	--tr-no-rtt	Отключает расчет и вывод RTT в режиме **traceroute**.
	--tcpexitcode	Задает завершение работы с кодом **tcp->th_flag**. Эта опция полезна при использовании программы в сценариях, которым требуется информация об откликах проверяемого хоста.

11.11.2.2 Формат вывода для протокола TCP

Стандартный формат вывода для пакетов TCP имеет вид:

```
len=46 ip=192.168.1.1 flags=RA DF seq=0 ttl=255 id=0 win=0 rtt=0.4 ms
```

- **len** - размер принятого от канального уровня пакета (в байтах) без учета заголовка канального уровня. Это значение может отличаться от размера дейтаграмм IP в результате исключения заполняющих байтов.
- **ip** - IP адрес отправителя.
- **flags** - флаги TCP (**R** - RESET, **S** - SYN, **A** - ACK, **F** - FIN, **P** - PUSH, **U** - URGENT, **X**[2] - 0x40, **Y**[3] - 0x80).
- **DF** - флаг запрета фрагментирования в заголовке IP.
- **seq** - порядковый номер пакета (для пакетов TCP/UDP устанавливается с использованием номера порта отправителя, для ICMP совпадает с порядковым номером).
- **id** - значение поля идентификации IP ID.
- **win** - размер окна TCP.
- **rtt** - время кругового обхода в миллисекундах.

При использовании опции **-V** будут выводиться дополнительные поля, как показано ниже:

```
len=46  ip=192.168.1.1  flags=RA DF seq=0 ttl=255 id=0 win=0 rtt=0.4 ms tos=0 iplen=40
seq=0 ack=1223672061 sum=e61d urp=0
```

- **tos** - поле типа обслуживания заголовка IP.
- **iplen** - поле размера из заголовка IP.
- **seq** - порядковый номер пакета из заголовка TCP.
- **ack** - порядковый номер подтверждения из заголовка TCP.
- **sum** - контрольная сумма из заголовка TCP.
- **urp** - значение флага срочности из заголовка TCP.

11.11.2.3 Формат вывода для пакетов UDP

Стандартный формат вывода для пакетов UDP имеет форму:

```
len=46 ip=192.168.1.1 seq=0 ttl=64 id=0 rtt=6.0 ms
```

Значение полей вывода совпадает со значением одноименных полей для протокола TCP.

11.11.2.4 Формат вывода для пакетов ICMP

Для пакетов ICMP информация выводится в формате, подобном показанному ниже:

```
ICMP Port Unreachable from ip=192.168.1.1 name=nano.marmoc.net
```

Каждая строка вывода начинается с идентификатора протокола **ICMP**, за которым следует тип сообщения ICMP. В поле **ip** указывается IP-адрес отправителя дейтаграммы IP, содержащей сообщение ICMP, поле **name** содержит символьное имя хоста-отправителя, определенное с помощью DNS (запись PTR) или ключевое слово **UNKNOWN**, если имя определить не удалось.

1 *Round Trip Time - время кругового обхода.*
2 *Нестандартный флаг 0x40.*
3 *Нестандартный флаг 0x80.*

Сообщения **ICMP Time exceeded** для доставки или сборки фрагментов выводятся с использованием несколько отличающегося формата:

```
TTL 0 during transit from ip=192.168.1.1 name=nano.marmoc.net
TTL 0 during reassembly from ip=192.70.106.25 name=UNKNOWN
```

Разница заключается в добавлении поля **TTL 0** перед описанием типа сообщения.

11.11.2.5 Известные проблемы

Даже при указании опций **--end** и **--safe** последний пакет, используемый для передачи файла, будет дополнен байтами 0x00.

Данные читаются без учета выравнивания, что может приводить к возникновению проблем в некоторых системах, где поля заголовков TCP/IP не выравниваются по естественным границам.

При работе под управлением ОС Solaris **hping** не может использовать интерфейс **loopback**. По-видимому это проблема ОС Solaris, поскольку библиотека libpcap так же не может корректно работать с этим интерфейсом.

11.11.3 packETH

http://packeth.sourceforge.net/

Генератор пакетов **packETH** представляет собой графическое приложение Linux обеспечивающее возможность генерации и передачи произвольных пакетов или последовательностей пакетов Ethernet. Программа использует сокет RAW (Приложение 12.8) поэтому ее не заботят проблемы протокола IP, маршрутизация и т. п. Генератор просто передает сетевому интерфейсу созданные пакеты. Программа предназначена для генерации пакетов с возможностью управления всеми опциями с использованием как корректных, так и некорректных значений.

Программа позволяет генерировать пакеты различных протоколов, включая:

- Ethernet II, Ethernet 802.3, 802.1q;
- ARP, Ipv4;
- UDP, TCP, ICMP;
- RTP.

Программа может передавать одиночные пакеты или последовательности с управляемыми параметрами:

- период передачи пакетов и число генерируемых пакетов;
- передача пакетов с максимальной скоростью, приближающейся к теоретическому пределу;
- изменение параметров в процессе передачи (адреса MAC и IP, данные UDP и т. п.)

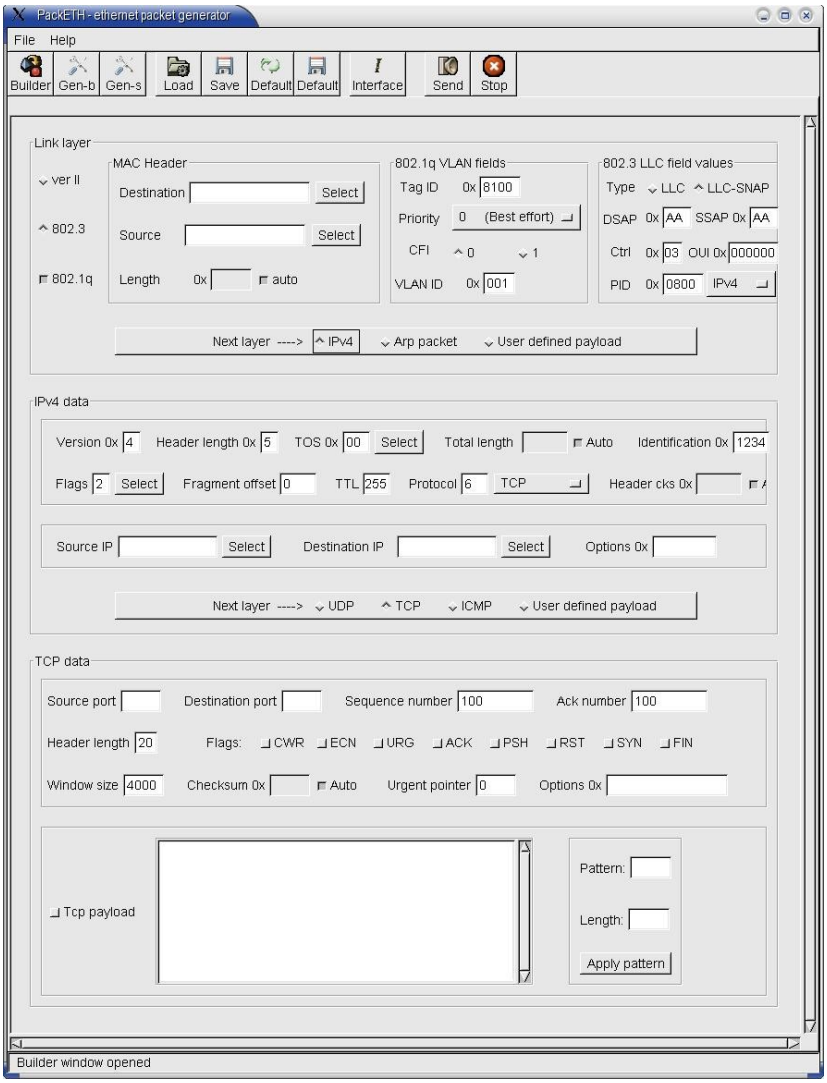


Рисунок 11.93 Интерфейс выбора параметров пакетов программы packETH

Работа с программой достаточно проста. Как можно видеть на рисунке 11.93 графический интерфейс программы позволяет задать все параметры генерируемых пакетов. Три основных функции программы - подготовка пакетов, генерация и передача однотипных пакетов и генерация последовательностей пакетов, - активизируются с помощью кнопок в панели инструментов главного окна программы.

При запуске программа автоматически переходит в режим создания пакетов (Builder), окно которого показано на рисунке 11.93. Задав параметры генерируемых пакетов вы можете передать пакет или серию однотипных пакетов (кнопка **Gen-b**) или генерировать последовательности разных пакетов (подготовленных заранее) с заданными параметрами (кнопка **Gen-s**). Можно также однократно передать подготовленный пакет с помощью кнопки **Send**, обеспечивающей передачу одного пакета. Режим **Gen-b**, служащий для передачи однотипных пакетов позволяет задать ряд опций (см. рисунок 11.94), недоступных при использовании кнопки **Send**. В этом режиме вы можете указать число передаваемых пакетов и задержку между ними, а также изменять

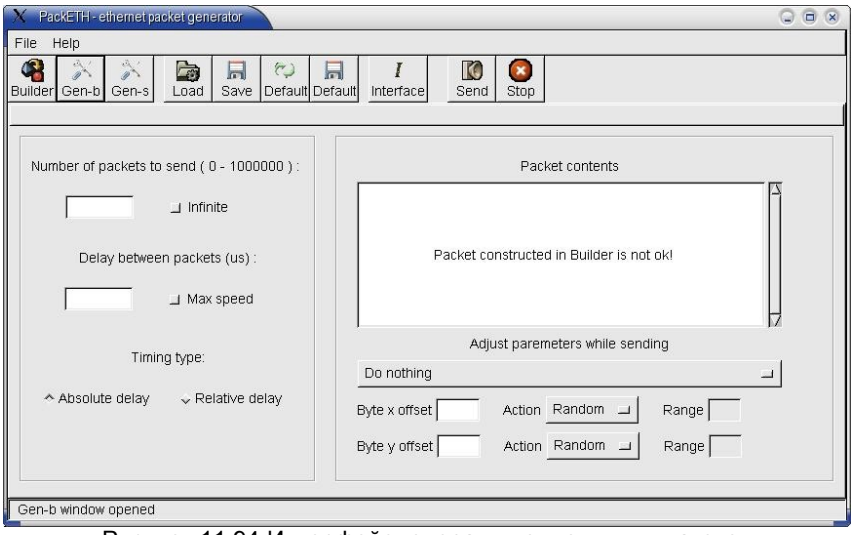


Рисунок 11.94 Интерфейс генерации однотипных пакетов

ряд параметров при передаче каждого пакета. В режиме генерации последовательностей **Gen-s** вы можете выбрать набор ранее подготовленных и сохраненных пакетов и передавать последовательности этих пакетов с заданными параметрами задержки, числом пакетов, количеством циклов передачи и т. п. Внешний вид интерфейса показан на рисунке .

Кнопка **Save** на панели инструментов служит для записи в файл параметров, заданных в текущем окне программы. Такие файлы могут впоследствии использоваться для генерации пакетов. Кнопка **Load** позволяет загрузить созданные ранее наборы параметров для генерации пакетов в текущем сеансе. Следующие две кнопки с именем **Default** служат для сброса параметров и загрузки используемого по умолчанию набора параметров[1] (первая кнопка) для каждого из режимов и для сохранения текущего набора параметров в качестве используемых по умолчанию (вторая кнопка). Кнопка **Interface** служит для выбора интерфейса, который будет использоваться для передачи пакетов. Вы можете выбрать любой интерфейс, поскольку используемый программой сокет RAW (Приложение 12.8) позволяет игнорировать таблицу маршрутизации при передаче пакетов. Две последние кнопки панели инструментов служат для запуска и остановки процесса генерации пакетов.

При генерации последовательности пакетов вы можете задать интервалы передачи в диапазоне от 1 мксек до 999 секунд. Однако следует отметить, что при малых задержках между передачей отдельных пакетов точность отсчета зависит от целого ряда причин и является недостаточно высокой. Генератор пакетов работает в двух режимах отсчета времени **absolute delay** и **relative delay**. В абсолютном режиме программа пытается передавать пакеты с заданным интервалом, поэтому задержка передачи одного пакета не влияет на передачу следующих пакетов. В относительном режиме интервал отсчитывается от момента отправки предыдущего пакета, поэтому после задержки отправки одного из пакетов будут задержаны и все последующие пакеты.

11.11.4 Packit

http://packit.sourceforge.net

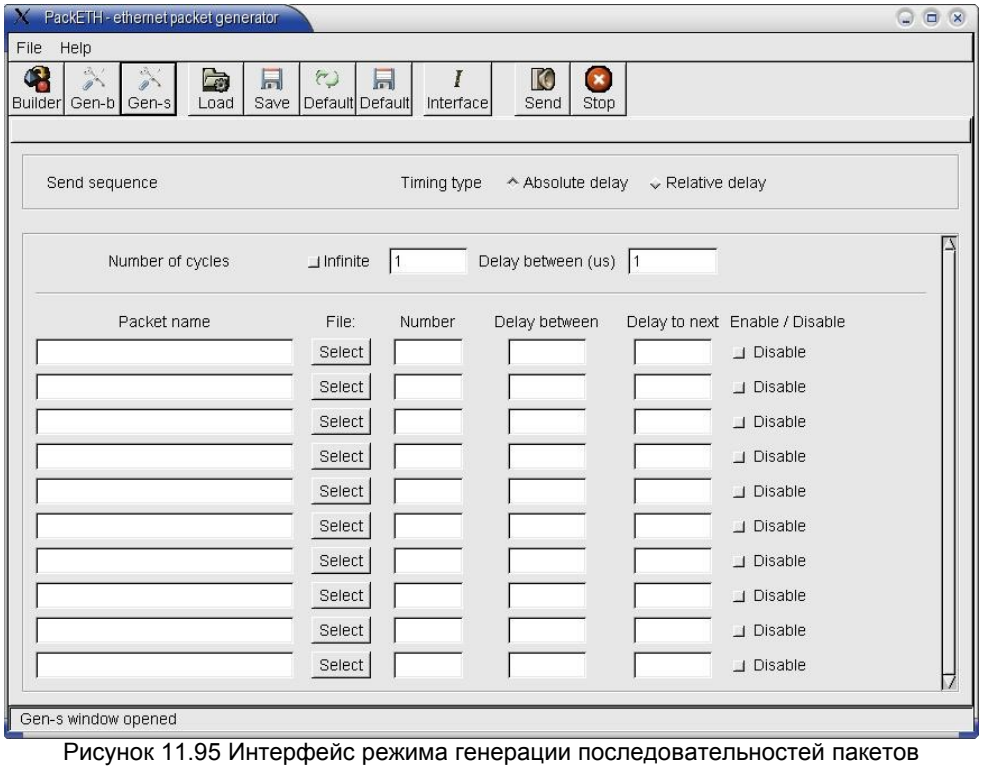


Рисунок 11.95 Интерфейс режима генерации последовательностей пакетов

Программа Packit (**Pac**ket tool**kit**[2]) представляет собой эффективное средство сетевого аудита с консольным интерфейсом. Программа обеспечивает возможность подготовки и генерации пакетов IP, а также мониторинга трафика и изменения пакетов. Возможность задать практически все параметры и опции пакетов TCP, UDP, ICMP, IP, ARP, RARP и кадров Ethernet делает программу Packit незаменимым инструментом для тестирования межсетевых экранов, обнаружения попыток вторжения, сканирования портов, имитации сетевого трафика и общего аудита сеетй TCP/I. Кроме того, Packit можно использовать в качестве эффективного пособия при изучении протоколов стека TCP/IP.

Программа Packit 1.0 требует для работы библиотеку **libnet** версии 1.1.2 или выше, а также библиотеку **libpcap** (параграф 11.9.1 на стр. 261). Программа была протестирована на платформах FreeBSD, NetBSD, OpenBSD, MacOS X и Linux.

1 *Эти наборы параметров хранятся в файлах* ***.defaultBuilder****,* ***.defaultGen-b*** *и* ***.defaultGen-s*** *отдельно для каждого из режимов.*
2 *Инструменты для пакетов.*

11.11.4.1 Анализ и генерация пакетов

Синтаксис

Режим сбора пакетов:

```
packit -m capture [-cGHnvsX] [-i interface] [-r|-w file] expression
```

Режим генерации пакетов:

```
packit -m inject [-t prot] [-aAbcCdDeFgGhHjJkKlLmMnNoOpPqQrRsSTuUvwWxXyYzZ] [-i
interface]
```

Режим трассировки:

```
packit -m trace [-t prot] [-i interface] -d dest [-S port] [-FS]
```

11.11.4.2 Опции командной строки

-m <режим>

задает режим работы программы - **capture**, **inject** или **trace**. По умолчанию программа запускается в режиме **inject**. Допускается использование сокращенных обозначений режима работы программы (вплоть до 1 символа).

11.11.4.2.1 Опции режима сбора пакетов

Таблица 92. Опции режима capture.

Опция	Описание
-c	Задает сбор заданного параметром числа пакетов и завершение работы.
-e	Задает вывод информации из заголовков канального уровня.
-G	Задает вывод временных меток в формате GMT взамен локального времени.
-i	Указывает интерфейс для сбора пакетов. По умолчанию программа использует активный интерфейс с минимальным номером (исключая **lo**).
-n	Отключает преобразование IP-адресов в имена хостов, сохраняя преобразование номеров портов.
-nn	Отключает преобразование номеров портов, сохраняя преобразование IP-адресов в имена хостов.
-nnn	Отключает преобразование IP-адресов и номеров портов в символьные имена.
-r	Читает пакеты из указанного файла в формате tcpdump (см. параграф 11.9.2 на стр. 262). Такой файл можно получить при использовании опции **-w** или взять результат работы другой программы сбора пакетов.
-s	Задает размер буфера захвата, ограничивающий количество данных, читаемых из каждого пакета. По умолчанию программа считывает 68 байтов.
-v	Задает вывод дополнительной информации.
-w	Задает запись собранных пакетов в указанный файл. Для записи используется формат tcpdump (см. параграф 11.9.2 на стр. 262).
-X	Задает вывод дампа собранных пакетов в шестнадцатеричном и ASCII-формате.
expression	Задает фильтр для отбора пакетов. Выражения для фильтрации используют синтаксис tcpdump (см. параграф 11.9.2.2 на стр. 265).

11.11.4.2.2 Опции режимов генерации и трассировки

Режим генерации позволяет создавать пакеты IP и передавать их в сеть. Вы можете задать параметры заголовков ARP, IP, TCP, UDP, ICMP и Ethernet. Режим генерации пакетов может быть весьма полезен при настройке межсетевых экранов, тестировании ISD, имитации потоков трафика и общем аудите сетей TCP/IP.

11.11.4.2.2.1 Опции общего назначения для режимов генерации и трассировки

Таблица 93. Опции packit для режимов генерации и трассировки.

Опция	Описание
-t	Задает тип генерируемых программой пакетов. Параметр опции может принимать значения ARP, TCP, UDP и ICMP. По умолчанию используется протокол TCP в режиме генерации пакетов и ICMP в режиме трассировки.
-c	Задает общее число генерируемых пакетов. При нулевом значении число пакетов не ограничивается.
-w	Задает интервал (в секундах) между передачей последовательных групп пакетов. По умолчанию группа пакетов передаются каждую секунду.
-b	Задает количество пакетов, передаваемое в течение заданного опцией **-w** интервала. При нулевом значении пакеты генерируются и передаются с максимально возможной скоростью.
-h	При включенной опции программа выводит информацию о переданном пакете и ждет (см. опцию **-H**) отклика от получателя.

Опция	Описание
-H	Задает время ожидания (в секундах) откликов при использовании опции **-h**. По умолчанию 1.
-i	Задает интерфейс для передачи пакетов.
-v	Задает вывод информации о каждом переданном в сеть пакете.
-p	Параметр этой опции определяет содержимое поля данных генерируемых пакетов. Вы можете задавать данные в формате ASCII **-p 'hello, this is my packet'** или в шестнадцатеричном формате **-p '0x 70 61 63 6B 69 74'**
-Z	Задает размер пакета. Максимальное значение составляет 65535 байтов.

11.11.4.2.2.2 Опции заголовков IP

В таблице 94 перечислены опции командной строки, используемые для управления заголовками IP в генерируемых программой пакетах.

Таблица 94. Опции заголовков IP.

Опция	Описание
-s	Задает адрес отправителя для генерируемых программой пакетов. Если адрес отправителя не указан, программа будет использовать IP-адрес активного интерфейса системы с минимальным номером (исключая интерфейс **lo**).
-sR	Задает использование случайных значений для адреса отправителя.
-d	Задает адрес получателя для генерируемых программой пакетов.
-dR	Задает использование случайных значений для адреса получателя.
-o	Задает значение поля IP TOS (тип сервиса) для генерируемых пакетов: **Minimize delay**: 16 (0x10) - минимальная задержка. **Maximize throughput**: 8 (0x08) - максимальная пропускная способность. **Maximize reliability**: 4 (0x04) - максимальная надежность. **Minimize monetary cost**: 2 (0x02) - минимальная стоимость в денежном выражении.
-n	Определяет значение идентификатора IP ID для генерируемых пакетов. Значение параметра этой опции будет служить в качестве стартового, а для каждого следующего пакета идентификатор будет увеличиваться на 1. По умолчанию отсчет идентификаторов начинается со случайного номера.
-T	Задает значение времени жизни для генерируемых программой пакетов. По умолчанию TTL=128.
-V	Задает номер протокола IP для генерируемых пакетов RAW. По умолчанию 255.

11.11.4.2.2.3 Опции заголовков TCP

В таблице 95 перечислены опции командной строки, используемые для управления заголовками TCP в генерируемых программой пакетах.

Таблица 95. Опции заголовков TCP.

Опция	Описание
-S	Задает номер порта отправителя для генерируемых программой пакетов TCP.
-D	Задает номер порта получателя для генерируемых программой пакетов TCP. По умолчанию в режиме генерации пакеты адресованы в порт 0, а в режиме трассировки используется случайное значение. Вы можете задать в качестве параметра этой опции диапазон номеров (например, **-D 1:1024**)
-f	Запрещает фрагментирование пакета.
-F	Задает для генерируемых пакетов флаги TCP: **S**: SYN (синхронизация порядковых номеров для пакетов в данном соединении) **F**: FIN (отправитель закончил передачу пакетов) **A**: ACK (подтверждение) **P**: PSH (флаг выталкивания данных из очереди) **U**: URG (срочные данные) **R**: RST (сброс соединения)
-q	Эта опция служит для задания 32-битовых порядковых номеров идентифицирующих данные в потоке TCP.
-a	Задает 32-битовый номер подтверждения (порядковый номер, который отправитель ждет в следующем принятом от другой стороны пакете).

Опция	Описание
-W	Задает размер окна TCP, используемый для управления потоком данных. 16-битовое значение размера окна определяет количество данных, которые получатель готов принять. По умолчанию размер окна для генерируемых программой пакетов TCP составляет 1500.
-u	Задает указатель важности для пакета. В корректных соединениях TCP указатель на срочные данные используется только при установленном флаге URG. С помощью этого значения можно определить последний байт срочных данных в потоке TCP.

11.11.4.2.2.4 Опции заголовков UDP

В таблице 96 перечислены опции заголовков UDP, которые могут быть установлены для генерируемых программой пакетов. Протокол UDP используется по умолчанию в режиме трассировки.

Таблица 96. Опции заголовков UDP.

Опция	Описание
-S	Задает номер порта отправителя для генерируемых программой пакетов UDP. По умолчанию используется случайный номер порта.
-D	Задает номер порта получателя для генерируемых программой пакетов UDP. По умолчанию в режиме генерации пакеты адресованы в порт 0, а в режиме трассировки используется случайное значение. Вы можете задать в качестве параметра этой опции диапазон номеров (например, **-D 1:1024**)

11.11.4.2.2.5 Опции заголовков ICMP

В таблице 97 перечислены опции заголовков ICMP, которые могут быть установлены для генерируемых программой пакетов.

Таблица 97. Опции заголовков ICMP.

Опция	Описание
-K	Задает тип пакета ICMP (см. таблицу 102).
-C	Задает код пакета ICMP (см. таблицу 102).

11.11.4.2.2.5.1 Опции запросов и откликов ICMP ECHO

Таблица 98. Опции запросов и откликов ICMP ECHO.

Опция	Описание
-N	Задает 16-битовый идентификационный номер ICMP. По умолчанию используется случайный номер.
-Q	Задает 16-битовый порядковый номер ICMP. По умолчанию используется случайный номер.

11.11.4.2.2.5.2 Опции откликов ICMP UNREACHABLE/REDIRECT/TIME EXCEEDED

Таблица 99. Опции откликов ICMP UNREACHABLE/REDIRECT/TIME EXCEEDED.

Опция	Описание
-g	Задает адрес шлюза, на который следует перенаправить пакеты. Эта опция используется только с сообщениями **ICMP redirect** (тип 5).
-j	Задает адрес отправителя для исходного пакета (данные ICMP).
-J	Задает номер порта отправителя для исходного пакета (данные ICMP).
-l	Задает адрес получателя для исходного пакета (данные ICMP).
-L	Задает номер порта получателя для исходного пакета (данные ICMP).
-m	Задает время жизни для исходного пакета (данные ICMP). По умолчанию TTL=128.
-M	Задает идентификатор IP ID для исходного пакета (данные ICMP). По умолчанию используется случайное значение.
-O	Задает IP TOS (тип обслуживания) для исходного пакета (данные ICMP). Возможные значения этого параметра описаны на стр. 336.
-P	Задает номер протокола IP для исходного пакета (данные ICMP). По умолчанию UDP.

11.11.4.2.2.5.3 Опции запросов и откликов ICMP MASK

Таблица 100. Опции запросов и откликов ICMP MASK.

Опция	Описание
-N	Задает 16-битовый идентификационный номер ICMP. По умолчанию используется случайный номер.
-Q	Задает 16-битовый порядковый номер ICMP. По умолчанию используется случайный номер.
-G	Задает маску подсети. По умолчанию 255.255.255.0.

11.11.4.2.2.5.4 Опции запросов и откликов ICMP TIMESTAMP

Таблица 101. Опции запросов и откликов ICMP MASK.

Опция	Описание
-N	Задает 16-битовый идентификационный номер ICMP. По умолчанию используется случайный номер.
-Q	Задает 16-битовый порядковый номер ICMP. По умолчанию используется случайный номер.
-U	Задает 32-битовое значение исходной временной метки. По умолчанию 0.
-k	Задает 32-битовое значение временной метки приема. По умолчанию 0.
-z	Задает 32-битовое значение временной метки передачи. По умолчанию 0.

11.11.4.2.2.5.5 Типы и коды сообщений ICMP

Таблица 102. Типы и коды сообщений ICMP.

Тип	Код	Имя	Описание
0		Echo Reply	Отклик на запрос Echo.
3	0	Network Unreachable	Сеть недоступна.
	1	Host Unreachable	Хост недоступен.
	2	Protocol Unreachable	Протокол недоступен.
	3	Port Unreachable	Порт недоступен.
	4	Need Fragment	Требуется фрагментация, но установлен флаг DF.
	5	Source Failed	Некорректно задан маршрут source route.
	6	Network Unknown	Неизвестная сеть.
	7	Host Unknown	Неизвестный хост.
	8	Isolated	Изолирован.
	9	Network Prohibited	Доступ в сеть запрещен.
	10	Host Prohibited	Доступ к хосту запрещен.
	11	Type of Service Network Unreachable	Сеть недоступна с заданным типом сервиса.
	12	Type of Service Host Unreachable	Хост недоступен с заданным типом сервиса.
	13	Filtered	Обмен данными запрещен администратором (фильтр).
	14	Host Precedence Violation	Нарушение предпочтений для хоста.
	15	Precedence Cutoff in effect	Действует ограничение предпочтений
4		Source Quench	Запрос снижения скорости передачи пакетов.
5	0	Network Redirect	Перенаправление дейтаграмм для сети.
	1	Host Redirect	Перенаправление дейтаграмм для хоста.
	2	Type of Service Network Redirect	Перенаправление дейтаграмм для сети и типа сервиса.
	3	Type of Serivce Host Redirect	Перенаправление дейтаграмм для хоста и типа сервиса.
8		Echo Request	Запрос Echo.
9		Router Advertise	Анонс маршрутизатора.
10		Router Solicit	Предложение маршрутизатора.
11	0	Time Exceeded In transit	Время жизни истекло в процессе доставки.
	1	Time Exceeded Reassemble	Время жизни истекло при сборке фрагментов.
12		Parameter Problem	Некорректные параметры.
	1	Option Absent	Отсутствует нужная опция.
13		Timestamp Request	Запрос временной метки.
14		Timestamp Reply	Отклик на запрос временной метки.
17		Mask Request	Запрос маски подсети.
18		Mask Reply	Отклик на запрос маски подсети.

11.11.4.2.2.6 Опции заголовков ARP

Описанные в таблице 103 опции заголовков ARP при неаккуратном использовании могут создать множество проблем, поэтому обращаться с ними следует очень осторожно.

Таблица 103. Опции генерации заголовков ARP.

Опция	Описание
-A	Задает тип операции ARP/RARP/IRARP и может принимать одно из перечисленных здесь значений: 1: ARP Request - запрос ARP; 2: ARP Reply - отклик ARP; 3: Reverse ARP Request - запрос RARP; 4: Reverse ARP Reply - отклик RARP; 5: Inverse ARP Request - запрос IARP; 6: Inverse ARP Reply - отклик IARP.
-y	Задает IP-адрес целевого хоста.
-yR	Задает использование случайных значений IP-адреса целевого хоста.
-Y	Задает MAC-адрес целевого хоста.
-YR	Задает использование случайных значений MAC-адреса целевого хоста.
-x	Задает IP-адрес хоста-отправителя.
-xR	Задает использование случайных значений IP-адреса хоста-отправителя.
-X	Задает MAC-адрес хоста-отправителя.
-XR	Задает использование случайных значений MAC-адреса хоста-отправителя.

11.11.4.2.2.7 Опции заголовков Ethernet

Опции командной строки для управления заголовка Ethernet перечислены в таблице 104.

Таблица 104. Опции генерации заголовков Ethernet.

Опция	Описание
-e	Задает MAC-адрес отправителя кадров.
-eR	Задает использование случайных значений MAC-адреса отправителя кадров.
-E	Задает MAC-адрес получателя кадров[1].
-ER	Задает использование случайных значений MAC-адреса получателя кадров.

При использовании опций **-e** и **-E** маршрутизация пакетов IP может быть нарушена, поэтому следует аккуратно задавать MAC-адреса получателя и отправителя с учетом реальной картины маршрутизации пакетов в вашей сети. Приведенные ниже правила помогут вам получить желаемый результат:

1) Если получатель находится за пределами вашей подсети, в заголовке кадров Ethernet следует указывать в качестве получателя MAC-адрес интерфейса маршрутизатора на пути к получателю. Определить этот адрес обычно можно с помощью просмотра таблицы маршрутов и команды **arp** (параграф 11.1.2.1 на стр. 193).

2) Если получатель находится внутри вашей подсети, в заголовке кадра следует указывать MAC-адрес интерфейса, который обычно можно определить с помощью команды **arp**.

11.11.4.3 Примеры команд

11.11.4.3.1 Сбор пакетов

При сборе пакетов TCP на удаленном хосте, к которому вы подключены по протоколу SSH (порт 22) команда

```
packit -m cap 'tcp and not port 22'
```

позволит вам увидеть все пакеты, за исключением тех, которые используются для сеанса SSH между вашей станцией и удаленным хостом.

Следующая команда позволит увидеть все пакеты организации и завершения соединений TCP (пакеты SYN и FIN) с хостами на пределами вашей локальной сети без преобразования адресов в имена и с выводом дампа пакетов. Взамен localnet укажите номер вашей подсети и маску (например, 192.168.0.0/27).

```
packit -m cap -nX 'tcp[tcpflags] & (tcp-syn|tcp-fin) != 0 and not src and dst net
localnet'
```

Для записи в файл **/tmp/mylog** первых 10 пакетов ICMP можно использовать команду:

```
packit -m cap -c 10 -w /tmp/mylog 'icmp'
```

11.11.4.3.2 Генерация пакетов

Для генерации 10 запросов **ICMP echo request** (тип 8) с хоста **3.1.33.7** и передачи их по адресу **192.168.0.1** с с выводом результатов на экран можно использовать команду.

1 *Если вы планируете отправить пакет IP за пределы вашей подсети, параметр этой опции должен содержать MAC-адрес интерфейса шлюза на пути к получателю.*

```
packit -t icmp -s 3.1.33.7 -d 192.168.0.1 -c 10 -h
```
Для генерации и передачи по адресу **127.0.0.1** отклика **ICMP mask reply** (тип 18) с идентификатором ICMP 211 и маской **255.255.255.0** используйте команду.

```
packit -t icmp -K 18 -d 127.0.0.1 -N 211 -G 255.255.255.0
```
Для генерации 5 запросов на соединения TCP (флаг **SYN**) с сервером **www.microsoft.com**, использующих окно размером 666, случайный MAC-адрес отправителя, MAC-адрес получателя **00:05:5D:00:33:44**[1] и содержащих в поле данных строку **HIBILL**, можно воспользоваться командой.

```
packit -sR -d www.microsoft.com -F S -c 5 -W 666 -eR -E 00:05:5D:00:33:44 -p 'HI BILL' -v
```
Следующая команда обеспечит генерацию 1000 пакетов TCP (по 20 пакетов в секунду) с адресом отправителя **192.168.0.1** (порт **403**) для передачи хосту **192.168.0.20** (порт **80**). Пакеты будут иметь флаги **SYN** и **RST**, начальный порядковый номер **12345678910** и MAC-адрес отправителя **0:0:0:0:0:0**.

```
packit -s 192.168.0.1 -d 192.168.0.20 -S 403 -D 80 -F SR -q 12345678910 -c 1000 -b 20 -e
0:0:0:0:0:0
```
Приведенная ниже команда обеспечивает генерацию пакетов TCP с флагом SYN, адресованных хосту **172.16.1.3** (порты с **1** по **1024**), с адресом отправителя **10.22.41.6**.

```
packit -s 10.22.41.6 -d 172.16.1.3 -D 1-1024 -F S -v
```
Команда

```
packit -t arp -A 2 -x 4.3.2.1 -X 5:4:3:2:1:0 -e 5:4:3:2:1:0 -p '0x 70 61 63 6B 69 74'
```
обеспечивает генерацию широковещательных откликов ARP, связывающих IP-адрес **4.3.2.1** с аппаратным адресом **5:4:3:2:1:0**. В пакетах используется подставной MAC-адрес отправителя и поле данных, заданное в шестнадцатеричном формате.

11.11.4.3.3 Трассировка

Приведенная ниже команда обеспечивает трассировку хоста 192.168.2.35 под видом откликов DNS (пакеты UDP из порта 53)

```
packit -m trace -t UDP -d 192.168.2.35 -S 53
```
Следующая команда будет проводить трассировку с использованием пакетов TCP из порта 80 (трафик HTTP).

```
packit -m trace -t TCP -d www.google.com -S 80 -FS
```

11.11.4.4 Известные проблемы

При сборе пакетов ARP выводится неполная информация.

11.12 Сетевые сканеры

Сканеры являются важным инструментом администратора сети, но могут служить и для обнаружения слабых мест в чужой защите. Описанные здесь программы обладают широкими возможностями и могут стать для вас как средством обеспечения безопасности, так и опасным оружием. Не используйте их во вред кому бы то ни было.

11.12.1 Nmap

http://www.insecure.org

http://cherepovets-city.ru/insecure/runmap/

Алексей Волков. Определение операционной системы удаленного хоста http://www.insecure.org/nmap/nmap-fingerprinting-article-ru.html

Программа **nmap** относится к числу сканеров портов и сканеров безопасности систем.

Программа позволяет администраторам сканировать отдельные хосты и целые сети, определяя поддерживаемые типы сервиса и другие параметры. Nmap поддерживает множество методов сканирования - UDP, TCP connect(), TCP SYN (half open), ftp proxy (bounce attack), Reverse-ident, ICMP (ping sweep), FIN, ACK sweep, Xmas Tree, SYN sweep, IP Protocol, Null scan. Более подробное описание этих методов приводится в параграфе 11.12.1.1.1. Кроме обычного сканирования программа **nmap** может определять тип операционной системы удаленного хоста, выполнять скрытое сканирование, параллельное сканирование, детектирование фильтров, прямое сканирование RPC (без portmapper), сканирование с использованием фрагментов и др.

Для выполнения большинства операций nmap требуются полномочия пользователя **root**, поскольку многие интерфейсы ядра (в частности, сокеты **raw**) требуют привилегий **root**. При запуске **nmap** от имени обычного пользователя значительная часть функций программы теряется.

По результатам работы программа **nmap** генерирует отчет, содержащий сведения об интересных портах просканированных хостов, если таковые были обнаружены. Для “хорошо известных” портов **nmap** всегда указывает имя сервиса, номер порта, его состояние и протокол. Состояние порта может быть **open** (открыт), **filtered** (фильтруется) или **unfiltered** (не фильтруется). Порт считается открытым если хост принимает адресованные в этот порт соединения. К фильтруемым относятся порты, которые активны, но доступ к ним заблокирован межсетевым экраном, пакетным фильтром или иными системами контроля трафика, которые не позволили программе **nmap**

1 *Как было отмечено выше, в этом случае поле MAC-адреса получателя должно содержать адрес интерфейса вашего шлюза, через который пакет будет передаваться получателю. Поэтому для использования такой команды в своей сети вы должны будете указать реальный адрес интерфейса своего маршрутизатора.*

организовать соединение с портом. К нефильтруемым портам относятся те, которые программа **nmap** определила как закрытые, не встретив при этом брандмауэра или иного средства предотвращения доступа к портам. Это состояние является обычным для большинства портов, поэтому они указываются в отчете лишь в тех случаях, когда большинство просканированных портов оказались фильтруемыми.

В зависимости от заданных опций **nmap** может также определять ряд характеристик удаленного хоста - операционную систему, порядковые номера TCP, имена пользователей, которые работают с программами, привязанными к портам, доменное имя DNS и другие параметры.

Синтаксис

```
nmap [<тип сканирования>] [<опции>] <хост или сеть #1 ... [#N]>
```

11.12.1.1 Опции

В силу широких возможностей программы число опций командной строки, управляющих режимом и параметрами сканирования весьма велико. Программа **nmap** проверяет заданный в командной строке набор опций и при наличии в них ошибок или противоречий выдает пользователю предупреждение. Список опций с краткими комментариями можно получить по команде **nmap -h**, более подробное описание вы получите с помощью команды **man nmap**.

Ниже приводятся описания поддерживаемых программой опций, объединенных в группы по их назначению.

11.12.1.1.1 Тип сканирования

В последующих параграфах описаны опции выбора типа сканирования или дополнительных операций, выполняемых программой **nmap**. Опции активизации этих методов указаны в скобках после названия метода.

11.12.1.1.1.1 Сканирование TCP SYN (-sS)

Этот метод часто называют сканированием с использованием полуоткрытых (half-open) соединений, поскольку при сканировании полные соединения TCP не организуются. Сканирующий хост передает пакет SYN как при обычной организации соединения и ожидает отклика. Полученный в ответ пакет **SYN ACK** говорит о том, что порт прослушивает входящие соединения, пакет **RST** показывает, что порт не прослушивается. При получении отклика **SYN ACK** незамедлительно передается пакет **RST** для сброса запрошенного соединения.

Основным преимуществом данного метода сканирования является то, что большинство сайтов не сохраняют записей о нем в своих журнальных файлах. Однако для использования метода требуются привилегии пользователя root, чтобы создавать пакеты SYN с нужными параметрами. Этот метод сканирования применяется по умолчанию для привилегированного пользователя.

11.12.1.1.1.2 Сканирование TCP connect (-sT)

Это один из основных методов сканирования TCP. Для организации соединения с каждым проверяемым портом служит системный вызов **connect()**. Если порт находится в состоянии **listening**, connect() возвращает позитивный результат, в противном случае функция сообщает о недоступности порта. Преимуществом этого метода является то, что он не требует каких-либо специальных привилегий для пользователя, поскольку вызов функции **connect** на большинстве систем UNIX доступен любому пользователю. Данный метод применяется по умолчанию для пользователей, не имеющих привилегий.

Сканирование с использованием этого метода легко обнаружить, поскольку проверяемый хост будет фиксировать в журнальных файлах многочисленные вызовы и сообщения об ошибках при обращении к закрытым портам.

11.12.1.1.1.3 Скрытое сканирование Stealth FIN, Stealth Xmas Tree, Stealth Null (-sF -sX -sN)

В ряде случаев сканирование **TCP SYN** не обеспечивает скрытности. Некоторые брандмауэры и системы фильтрации пакетов следят за пакетами SYN, направленными в закрытые порты, а программы типа **PortSentry** (параграф 11.6.2 на стр. 250) и **Courtney** (параграф 11.6.3 на стр. 254) способны детектировать сканирование **TCP SYN**. Эти методы сканирования достаточно эффективны и практически не оставляют следов.

Идея этих методов состоит в том, что при обращении к закрытым портам вы должны получить отклик **RST**, а открытые порты должны игнорировать такие пакеты в соответствии со стандартом (RFC 793). При сканировании **Stealth FIN** в качестве зондов передаются пакеты с флагом **FIN**, метод **Stealth Xmas tree** использует пакеты с флагами **FIN**, **URG** и **PUSH**, а сканирование **Stealth Null** основано на передаче пробных пакетов без флагов.

Эти методы не позволяют сканировать большинство систем Windows, поскольку компания Microsoft, по своему обыкновению, проигнорировала стандарт и реализовала протокол как получилось[1]. Существуют и другие системы, в которых реакция на сканирование не соответствует стандарту. Системы Cisco, BSDI, HP/UX, MVS и IRIX передают пакет **RST** при сканировании открытых портов, хотя в соответствии со стандартом должны просто отбрасывать пакеты.

Вы можете видеть сравнить результаты сканирования одного хоста с использованием методов **TCP SYN** (рисунок 11.96) и **Stealth FIN** (рисунок 11.97).

1 *Нет худа без добра и поведение стека протоколов Microsoft позволяет с помощью этих методов сканирования достаточно достоверно идентифицировать хосты, работающие в среде Windows. Если после сканирования любым из этих методов вы получили информацию хотя бы об одном открытом порте, это говорит о том, что хост не использует Windows. Если же сканирование в режиме **-sF**, **-sX** или **-sN** говорит, что все порты закрыты, а сканирование SYN (**-sS**) показывает наличие открытых портов, это с высоким уровнем достоверности указывает на систему Windows. Польза от такой возможности невелика, поскольку **nmap** поддерживает эффективные средства детектирования ОС.*

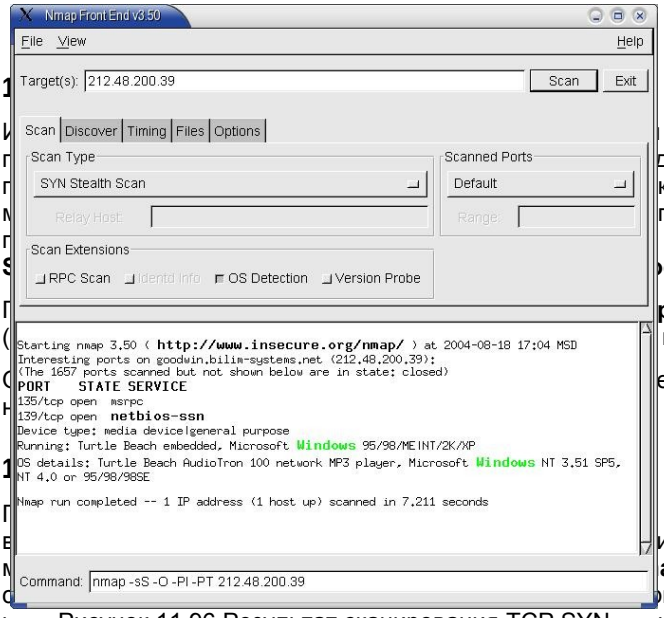


Рисунок 11.96 Результат сканирования TCP SYN

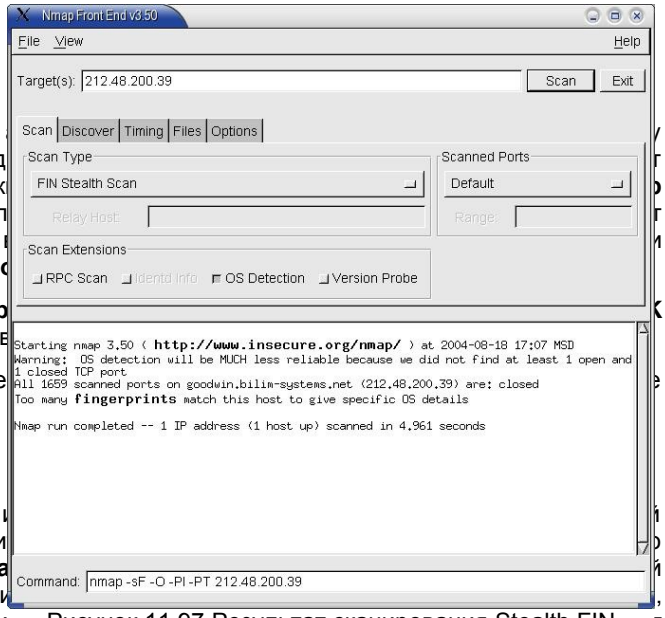


Рисунок 11.97 Результат сканирования Stealth FIN

пытаться организовать соединение с серверами SSL для определения возможности использования шифрованных соединений. При обнаружении служб RPC используется модуль **Nmap RPC grinder** для детектирования программы RPC и номера версии этой программы. Добавочная опция **--version_trace** обеспечит вывод отладочной информации в процессе детектирования версии для удаленного хоста.

Дополнительные сведения о системе детектирования версий вы найдете на сайте http://www.insecure.org/nmap/versionscan.html.

11.12.1.1.1.6 Сканирование UDP (-sU)

Этот метод применяется для детектирования открытых портов UDP. Метод основан на передаче пустых пакетов UDP проверяемому хосту. Получение в ответ пакета **ICMP port unreachable** будет говорить о том, что порт закрыт, а отсутствие такого отклика позволяет предположить наличие открытого порта. Однако зачастую пакеты **ICMP unreachable** фильтруются межсетевыми экранами, поэтому достоверной при таком сканировании можно считать только информацию о закрытых портах. В некоторых случаях Internet-провайдеры блокируют некоторые “опасные” порты (например 31337 - back orifice и 139 - Windows NetBIOS), что может создать иллюзию открытости таких портов. Не впадайте в панику, если вы обнаружили нечто подобное при сканировании своей системы извне.

Иногда приходится слышать, что сканирование портов UDP не имеет никакого смысла. Автор nmap[1] в качестве контраргумента приводит ситуацию с уязвимостью Solaris **rpcbind**. Этот сервис можно связать с недокументированным портом UDP, имеющим номер более 32770. Найти такой порт без сканирования UDP достаточно сложно.

Сканирование портов UDP может занять продолжительное время, поскольку на большинстве хостов[2] реализованы рекомендации RFC 1812 (параграф 4.3.2.8) по ограничению скорости передачи откликов ICMP. Например, ядро Linux[3] ограничивает скорость генерации откликов **ICMP destination unreachable** 80 пакетами в течение 4 секунд с использованием паузы в 250 мсек после превышения заданного порога. В Solaris ограничение еще жестче (около 2 откликов в секунду). Такое ограничение скорости откликов существенно замедляет сканирования. **Nmap** старается определить скорость откликов и в соответствии с ней задать темп передачи запросов, поскольку в противном случае часть запросов пропадет втуне.

11.12.1.1.1.7 IP-сканирование (-sO)

Этот метод используется для определения поддерживаемых хостом протоколов IP. Метод основан на передаче raw-пакетов IP без дополнительных протокольных заголовков, адресованных всем протоколам проверяемого хоста. Получение отклика **ICMP protocol unreachable** говорит, что соответствующий протокол не поддерживается хостом, а отсутствие такого сообщения позволяет предположить наличие протокола[4].

Используемый этим методом вариант передачи пакетов похож на сканирование UDP и ему присущи те же ограничения, связанные с ограничением темпа генерации сообщений ICMP. Однако поле номера протоколов IP имеет размер 8 битов, поэтому проверяется лишь 256 протоколов и это не должно занять много времени.

11.12.1.1.1.8 Метод скрытого сканирования Idlescan (-sI)

```
-sI <хост[:порт]>
```

1 *Известный по имени Fyodor.*
2 *Реализации протокола в продукции Microsoft, как обычно, не соответствуют RFC и скорость передачи откликов в системах Windows не ограничивается. Это позволяет просканировать все 65K портов UDP за достаточно короткое время*
3 *См. файл* ***<net/ipv4/icmp.h>***
4 *Некоторые ОС (AIX, HP-UX, Digital UNIX) и межсетевые экраны не передают сообщений о недоступности протокола, поэтому отсутствие таких сообщений не говорит однозначно о наличии протокола и лишь позволяет предположить его поддержку хостом.*

Этот метод позволяет полностью замести следы сканирования портов TCP и проверяемый хост не будет даже получать пакетов с IP-адресом сканирующей машины. Вместо передачи пакетов со сканирующего хоста используется подставной хост, доступный через Internet. Системы IDS будут показывать сканирование с указанного параметром подставного хоста и не смогут получить адрес вашего компьютера.

Предложенная в конце 1998 года технология сканирования **dumb host scan**[1] основана на предсказуемости значений поля IP ID в пакетах IP. Для реализации этого метода требуется промежуточный хост, имеющий по крайней мере один открытый порт TCP. Для успешного сканирования требуется, чтобы во время такой операции этот хост не проявлял собственной сетевой активности, но таких хостов в сети достаточно много. Для описания сути метода обозначим используемый для сканирования хост буквой **A**, промежуточный хост буквой **Z**, а проверяемый - **T**.

Хост **A** может осуществлять мониторинг хоста **Z** по значениям поля ID в заголовке передаваемых этим хостом пакетов IP. Дело в том, что большинство реализаций протокола IP просто увеличивают значение поля **ID** в заголовке пакетов IP на 1 для каждого следующего пакета. В этом легко убедиться с помощью описанной в Приложении 11.11.2 программы **hping2**. Таким образом по значению поля **ID** в заголовке IP полученных от хоста откликов можно определить количество пакетов, переданных этим хостом в интервале между генерацией откликов.

Вспомним, что при получении пакета **SYN** для открытого порта хост передает в ответ пакет **SYN ACK**. Если же порт закрыт, в ответ на **SYN** передается пакет **RST ACK**. При получении неожиданного пакета **SYN ACK** хост передает в ответ пакет **RST**, а при получении неожиданного пакета **RST** просто отбрасывает такой пакет.

На основании сказанного легко построить модель скрытого сканирования.

1) Хост **A** генерирует серию запросов ICMP, позволяющую контролировать рост значений поля ID в заголовке IP полученных от **Z** откликов.
2) Хост **A** генерирует пакет SYN, адресованный в интересующий порт хоста **T**, используя в качестве адреса отправителя IP-адрес хоста **Z**.
3) Хост **T** при получении пакета **SYN** шлет хосту **Z** пакет **SYN ACK**, если проверяемый порт открыт и **RST ACK**, если этот порт закрыт.
4) Хост **Z** получает от хоста **T** неожиданный пакет **SYN ACK** или **RST ASK**.
   a) при получении **SYN ACK** хост **Z** передает отклик **RST**, вследствие чего увеличивается значение поля ID;
   b) при получении пакета **RST ASK** хост **Z** просто отбрасывает такой пакет и увеличения ID не происходит.
5) Хост **A**, анализируя значения поля ID в заголовках откликов ICMP от хоста **Z**, может фиксировать передачу хостом **Z** пакета в интервале между откликами. Исходя из предположения об отсутствии собственной активности хоста **Z**, это позволяет говорить о передаче отклика на пакет **SYN ACK** от хоста **T** и наличии у последнего открытого порта

Как видите, метод очень прост и при грамотной реализации метода позволяет с высокой степенью достоверности скрытно определять состояние портов проверяемого хоста без риска быть замеченным.

Необязательный параметр **<порт>**, передаваемый программе с этой опцией, позволяет указать порт подставного хоста, который будет использоваться при сканировании. По умолчанию **nmap** будет использовать **tcp ping**.

Поскольку состояние хоста **Z** во время сканирования играет достаточно важную роль, разумно будет сначала убедиться в том, что этот хост действительно подходит для наших целей. Сделать это можно, например, с помощью команды **hping -r <IP-адрес>**. Если вывод этой команды будет подобен приведенному на рисунке (содержит **id=+1** в течение достаточно продолжительного времени), этот хост вполне подходит для использования в качестве подставного.

Программа nmap может корректно работать только с хостами, увеличивающими значение ID в каждом пакете на 1. Проверка показывает, что даже при стабильном значении **id=+1** программа просто не работает.

1 *Сообщение о возможности такого метода можете найти на сайте* *http://seclists.org/bugtraq/1998/Dec/0082.html.*

```
bash-2.05b# hping -r <IP-адрес>
HPING <IP-адрес> (eth0 <IP-адрес>): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=<IP-адрес> ttl=64 DF id=3064 sport=0 flags=RA seq=0 win=0 rtt=0.6 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=1 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=2 win=0 rtt=0.4 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=3 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=4 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=5 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=6 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=7 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=8 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=9 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=10 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=11 win=0 rtt=0.4 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=12 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=13 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=14 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=15 win=0 rtt=0.3 ms
len=46 ip=<IP-адрес> ttl=64 DF id=+1 sport=0 flags=RA seq=16 win=0 rtt=0.3 ms
```

Рисунок 11.98 Проверка подставного хоста

Если в процессе сканирования подставной хост будет генерировать пакеты по каким-либо иным причинам, кроме реакции на отклики от проверяемого хоста, это приведет к появлении в результатах ложной информации об открытых портах. Никто не мешает вам повторить сканирование, используя этот же хост или указав в качестве подставного другой хост.

11.12.1.1.1.9 ACK-сканирование (-sA)

Этот метод обычно используется для получения данных о политике межсетевого экрана. В частности, с помощью этого метода можно определить, учитывает брандмауэр состояние соединений (stateful inspection) или является простым пакетным фильтром, который блокирует входящие пакеты SYN.

В этом режиме программа передает пакеты ACK с кажущимися случайными номерами подтверждений и порядковыми номерами в сканируемые порты. При получении отклика **RST** порт считается нефильтруемым. Если же отклика просто не приходит или возвращается сообщение **ICMP unreachable**, порт считается фильтруемым. Программа **nmap** обычно не выводит сведений о нефильтруемых портах, поэтому отсутствие какого-либо списка портов в результате сканирования говорит о том, что ни один из проверенных портов не фильтруется.

В этом режиме список открытых портов обычно не выводится программой.

11.12.1.1.1.10 Window-сканирование (-sW)

Этот метод основан на определении размера окна TCP и похож на ACK-сканирование, но отличается от него тем, что наряду с детектированием состояния портов **filtered/unfiltered** он иногда может детектировать порты в состоянии **open** (вследствие получения аномальных размеров окна TCP, возвращаемых некоторыми ОС, включая AIX, Amiga, BeOS, BSDI, Cray, Tru64 UNIX, DG/UX, OpenVMS, Digital UNIX, FreeBSD, HP-UX, OS/2, IRIX, MacOS, NetBSD, OpenBSD, OpenStep, QNX, Rhapsody, SunOS 4.X, Ultrix, VAX, VxWorks).

11.12.1.1.1.11 Сканирование RPC (-sR)

Этот метод используется в сочетании с другими методами сканирования, поддерживаемыми программой **nmap** и служит для проверки всех обнаруженных при сканировании открытых портов TCP/UDP на предмет поддержки функций RPC[1]. Проверка осуществляется путем передачи в порт команды **NULL** с помощью **SunRPC**. Для поддерживающих RPC портов предпринимается попытка идентификации связанной с портом программы и номера ее версии. Таким образом можно определить порты и функции RPC даже в тех случаях когда порт **sunrpc** (**111**, **portmapper**) закрыт с помощью межсетевого экрана или иными средствами. В режиме сканирования RPC не работают приманки (decoy), описанные на стр. 347.

11.12.1.1.1.12 Сканирование по списку (-sL)

В этом режиме просто выводится список адресов IP или имен хостов, заданных другими параметрами командной строки без реального сканирования этих хостов.

11.12.1.1.1.13 Сканирование FTP bounce attack (-b)

```
-b <промежуточный сервер FTP>
```

Этот метод основан на возможности опосредованной передачи файлов по протоколу FTP (RFC 959[2]). Этот протокол имеет странную по сегодняшним меркам особенность, позволяющую пользователю хоста A подключиться к серверу FTP на другом хосте и запросить у этого сервера передачу файла **любому** хосту Internet. Как было отмечено еще в 1995 году, протокол FTP можно использовать для неконтролируемой рассылки электронной почты и новостных

1 *Remote Procedure Call - удаленный вызов процедур.*
2 *Копию документа вы можете найти в каталоге Documents/ приложенного к книге компакт-диска или загрузить с сайта http://rfc-editor.org/rfc/rfc959.txt.*

сообщений, заполнения чужих дисков ненужными файлами, попыток обхода межсетевых экранов и иных анонимных пакостей.

Программа **nmap** использует эту особенность протокола для сканирования портов TCP с использованием промежуточных серверов FTP. Сканер соединяется с сервером FTP, находящимся за брандмауэром, который закрывает сканируемый хост, и сканирует с его помощью закрытые межсетевым экраном порты (например, порт 139). Если сервер FTP имеет открытый для записи каталог, можно организовать передачу в порты сканируемого хоста любых данных, позволяющих найти в системе открытые порты.

Опция используется с параметром, указывающим параметры подключения к промежуточному серверу FTP в стандартной нотации URL (**username:password@server:port**). Все компоненты этого параметра, за исключением адреса или имени сервера, являются необязательными.

Следует отметить, что далеко не все современные серверы FTP пригодны для этого типа сканирования.

11.12.1.1.2 Опции общего назначения

Ни одна из перечисленных в таблице 105 опций не является обязательной, но многие опции весьма полезны. Отметим, что опции **-P** можно объединять - это поможет преодолеть даже весьма изощренные брандмауэры за счет использования различных портов и флагов TCP и кодов ICMP.

Таблица 105. Опции команды nmap.

Опция	Описание
-P0	Эта опция отключает попытки использования команды **ping** перед сканированием хоста. С помощью этой опции вы сможете сканировать сети, блокирующие пакеты **ICMP echo** на межсетевом экране. Примером такой сети является **microsoft.com** и при сканировании хостов Microsoft всегда следует задавать опцию **-P0** или **-PT80**. Отметим, что **ping** в контексте сканирования портов может означать не только традиционную передачу запросов **ICMP echo**. **Nmap** поддерживает множество типов проб, включая зонды TCP, UDP и ICMP. По умолчанию **Nmap** передает пакеты **ICMP echo request** и пакеты **TCP ACK**, адресованные в порт 80.
-PT [<порты>]	Эта опция задает использование “TCP ping” с указанными номерами портов для определения доступности хоста. Взамен передачи запросов **ICMP echo** генерируются пакеты **TCP ACK** и анализируются отклики на эти запросы. Активные хосты должны передавать в качестве отклика на такие запросы пакет **RST**. Эта опция может быть весьма полезна для проверки доступности хостов в сетях, где брандмауэры блокируют пакеты ICMP. Если эту опцию указал пользователь, не имеющий привилегий **root** для сканирования будет вызываться функция **connect()**. Используемые для сканирования порты задаются в виде списка номеров или имен, разделенных запятыми **-PT<порт1>[,порт2][...]**. По умолчанию используется порт 80, поскольку его фильтруют достаточно редко.
-PS [<порты>]	Эта опция задает использование пакетов **SYN** вместо пакетов **ACK**, доступных только для пользователя **root**. Хосты должны отвечать на такие запросы пакетами **RST** или **SYN ACK**. Вы можете указать номера портов так же, как для опции **-PT**.
-PU [<порты>]	Эта опция задает передачу пакетов UDP в заданные порты и ожидание откликов **ICMP port unreachable** (порт закрыт) или UDP (порт открыт), если хост активен. Поскольку многие службы UDP не отвечают на пустые пакеты, эта опция полезна скорее для поиска закрытых портов, нежели открытых.
-PE	Задает использование стандартной операции **ping** (пакеты **ICMP echo request**) для определения доступности хостов.
-PP	Задает использование запросов **ICMP timestamp** (тип 13) для поиска активных хостов.
-PM	Подобна опциям **-PE** и **-PP**, но использует запросы **ICMP netmask** (тип 17).
-PB	Эта опция задает использование стандартной операции **ping** (**-PE**) в параллель с пакетами **ACK** (**-PT**). Такой способ позволяет обойти брандмауэры, которые блокируют один из этих вариантов проб. Для пакетов **ACK** можно задать номер порта, как было описано выше для опции **-PT**.

Опция	Описание
-O	Эта опция задает определение операционной системы сканируемых хостов с помощью методов **TCP/IP fingerprinting**[1]. Для детектирования ОС используется множество методов анализа стека протоколов сканируемого хоста. Полученная при сканировании информация сравнивается с “отпечатками” известных ОС[2] для идентификации операционной системы данного хоста. Эта опция также включает несколько дополнительных тестов, в частности - определение времени с момента загрузки хоста (Uptime) с помощью опции **TCP timestamp** (RFC 1323), если она поддерживается проверяемым хостом. Кроме того, опция **-O** определяет уровень предсказуемости порядковых номеров TCP, определяющий сложность организации обманных соединений TCP с проверяемым хостом. Информация о предсказуемости порядковых номеров выводится только при наличии в командной строке опции **-v**. При одновременной использовании опций **-v** и **-O** определяется также алгоритм генерации порядковых номеров IP ID. Большинство хостов относится к классу **incremental**, использующему увеличение значения поля **ID** в заголовках IP на 1 для каждого генерируемого пакета. Такой алгоритм генерации порядковых идентификаторов может оказать весьма большую услугу злоумышленникам при организации атаки на хост или использовании этого хоста для скрытого сканирования других сетей (см. параграф 11.12.1.1.1.8 на стр. 342).
-A	Этот флаг позволяет использовать расширенные возможности программы по детектированию ОС (**-O**), определению версии (**-sV**) и др. Эта опция не влияет на опции синхронизации программы, описанные ниже.
-6	Эта опция включает поддержку протокола IPv6. Все цели сканирования должны задаваться адресами IPv6 (например, **3ffe:501:4819:2000:210:f3ff:fe03:4d0**) или полными доменными именами (записи AAAA). Версия **nmap6** доступна на сайте http://nmap6.sourceforge.net/.
-I	Эта опция включает режим сканирования **TCP reverse ident**. Дейв Голдсмит (Dave Goldsmith) в 1996 году отметил, что протокол **ident** (RFC 1413) позволяет раскрыть имена пользователей, владеющих любыми процессами, подключенными по протоколу TCP, даже если этот процесс не был инициатором соединения. Таким образом можно, подключившись, например, к порту **http**, с помощью **identd** определить на сервере наличие процессов пользователя **root**. При использовании опции **-I** удаленному демону **identd** передаются запросы для каждого найденного на сервере открытого порта. Опция -I может использоваться только в режиме **TCP connect (-sT)**.
-f	Эта опция задает использование мелких фрагментов IP при сканировании в режимах **SYN** (стр. 341), **FIN** (стр. 341), **XMAS** (стр. 341) или **NULL** (стр. 341). Смысл заключается в разбиении заголовка TCP на множество компонент, передаваемых в разных фрагментах IP для затруднения работы пакетных фильтров, систем IDS и других способов обнаружения фактов сканирования. Эту опцию следует использовать с осторожностью, поскольку многие программы недостаточно корректно обрабатывают мелкие фрагменты.
-v	Задает вывод дополнительной информации в процессе сканирования и по завершении. Для дополнительного увеличения объема выводимых данных можно указать опцию дважды. Вы можете также указать в командной строке одну или несколько опций **-d** для вывода отладочной информации.
-h	Выводит на экран краткую справку о работе с программой.
-oN <файл>	Задает запись результатов сканирования в указанный текстовый файл.
-oX <файл>	Задает запись результатов сканирования в указанный файл XML. В качестве параметра опции можно указать символ **-** для вывода информации на **stdout** (например, в канал и т. п.); в этом случае программа отключает вывод информации на экран, а сообщения об ошибках будут передаваться на **stderr**. Описание вывода результатов в формате XML вы можете найти на сайте http://www.insecure.org/nmap/nmap.dtd.
-oG <файл>	Задает запись результатов сканирования в указанный файл, пригодный для операций поиска с помощью команды **grep**. В этом случае все результаты выводятся в одну строку файла. Такой формат может быть удобен для передачи результатов сканирования в другие программы, но формат XML (опция **-oX**) обеспечивает более эффективное решение. При использовании вместо имени файла символа **-** весь вывод будет направлен на **stdout** (например, в канал), а сообщения об ошибках будут направляться на **stderr**.
-oA <имя>	Говорит программе о необходимости записи результатов во всех поддерживаемых форматах (**-oN, -oG**, **-oX**). Параметр опции задает имя файла, к котором добавляется расширение **.nmap**, **.gnmap** и **.xml**, соответственно.
-oS <файл>	Задает запись результатов сканирования в указанный файл с использованием формата ScriptKiddie. Вместо имени файла можно указать символ **-** для вывода результатов на **stdout**.

1 *Буквально - отпечатки пальцев.*
2 *См. файл **nmap-os-fingerprints** из пакета **nmap***

Опция	Описание
--resume <файл>	С помощью этой опции может быть возобновлено сканирование, прерванное по тем или иным причинам, если его результаты были сохранены ув укащзанном параметром опции текстовом файле (запись с опцией **-oN** или **-oG**). **Nmap** возобновит прерванное сканирование с использованием исходного набора опций.
--append_output	Говорит программе **nmap** о необходимости добавления информации в конец файла вместо переписывания этого файла.
-iL <файл>	Задает использование целей сканирования из указанного файла, который должен содержать список адресов и/или имен, разделенных пробелами, символами табуляции или новой строки (см. параграф 11.12.1.2 стр. 349). Если вы укажете вместо имени файла символ **-**, программа будет ждать список целей от устройства **stdin**.
-iR <количество>	Эта опция задает программе **nmap** сканирование указанного количества случайно выбранных адресов. Для бесконечного сканирования случайных адресов можно задать **-iR 0**. Такой способ может быть полезен для статистической оценки того или иного интересующего вас параметра (задается другими опциями) в сети Internet. Например, команда **nmap -sS -PS80 -iR 0 -p 80** поможет сделать случайную выборку web-серверов.
-p <порты>	Эта опция задает порт для проверки. Например, **-p 23** будет указывать программе, что на сканируемом хосте нужно проверить лишь порт 23, а **-p 20-30,139,60000-** будет проверять порты с 20 по 30, порт 139 и все порты с номерами выше 60000. По умолчанию программа сканирует порты с номерами от 1 до 1024 и все порты, указанные в файле **services** из пакета **nmap**. В режиме сканирования IP (**-sO**) эта опция задает номера проверяемых протоколов (0-255). При одновременном сканировании портов TCP и UDP вы можете задать номера портов отдельно для каждого протокола с помощью префиксов **T:** и **U:**. Все номера портов после префикса относятся к указанному протоколу, пока в строке не будет найден префикс другого протокола.
-F	Эта опция задает режим быстрого сканирования при котором проверяются только порты, указанные в файле **services** из пакета **nmap** (или в файле **protocols** при использовании режима **-sO**). Ограниченный набор проверяемых портов позволяет существенно ускорить процесс сканирования.
-D <decoy1[,decoy2][,ME],...>	Задает режим обмана сканируемого хоста, при котором последнему кажется, что сканирование осуществляется не только с вашего хоста, но и с хостов, указанных параметрами **decoy**. В результате системы IDS будут выдавать список из множества сканирующих хостов с уникальными адресами IP, среди которых ваш хост может просто затеряться. Для разделения подставных адресов используются запятые, а параметр **ME** указывает позицию списка адресов, в которую вы хотите поместить свой реальный адрес IP. Если вы укажете **ME** после пятого элемента списка или далее, некоторые детекторы сканирования просто никогда не покажут ваш адрес. Если параметр **ME** не указан, **nmap** будет помещать реальный адрес в случайную позицию. Не используйте в качестве подставных адреса неактивных хостов, поскольку это может привести к возникновению SYN-атаки на сканируемый хост. Кроме того, если вы укажете бездействующие адреса, среди них будет гораздо проще идентифицировать ваш реальный адрес. Некоторые детекторы сканеров (например, **portsentry**[1]) будут подавлять маршрут к сканирующему хосту. Учитывая, что адрес сканирующего хоста может быть подставным, не следует принимать таких мер. Подставные адреса используются как при начальном ping-сканировании (с использованием **ICMP**, **SYN**, **ACK** и т. д.), так и при последующем реальном сканировании портов исследуемого хоста. Можно использовать подставные адреса и при определении ОС (опция **-O**). Нет ничего плохого в использовании большого числа подставных адресов, но это будет замедлять сканирование, а в некоторых случаях - снижать достоверность результатов. Отметим также, что некоторые операторы не выпускают из сети пакеты с обманными адресами отправителя и эта опция в таком случае не будет работать.
-S <адрес>	В некоторых случаях **nmap** не может определить адрес отправителя (вы получите сообщение об этом) и данная опция позволяет задать IP-адрес интерфейса, который будет использоваться для передачи пакетов в сеть.
-e <интерфейс>	Указывает программе **nmap** интерфейс, который следует использовать для передачи пакетов. Обычно интерфейс определяется автоматически.
-g <порт>	Задает номер порта отправителя для используемых при сканировании пакетов. Это может помочь в тех случаях, когда проверяемый хост защищен брандмауэром, но на последнем открыты некоторые порты[2]. Отметим, что в иногда задание порта ведет к снижению производительности сканирования.

1 *См. параграф 11.6.2 на стр. 250.*

Опция	Описание
--data_length <number>	Обычно **nmap** передает пакеты минимального размера, достаточного для включения заголовка транспортного уровня (для TCP 40 байтов, для ICMP - 28). Эта опция указывает программе на необходимость дополнения пакета случайными значениями до заданного размера. Опция не влияет на пакеты, используемые для определения ОС (режим **-O**), но влияет на большинство других пакетов. Отметим, что использование пакетов заданного размера несколько снижает производительность, но более крупные пакеты обычно привлекают меньше внимания, поскольку напоминают обычный трафик.
-n	Отключает преобразование адресов в символьные имена с помощью DNS. Эта опция может существенно ускорить процесс сканирования.
-R	Задает обязательное преобразование адресов в доменные имена с помощью DNS. Обычно преобразование осуществляется только для активных адресов.
-r	Отключает случайный выбор порядка сканируемых портов.
-ttl <value>	Устанавливает значение TTL в заголовках передаваемых пакетов IPv4.
--randomize_hosts	Говорит программе о необходимости перемешивания перед сканированием адресов каждой группы, содержащей до 2048 хостов. Такое перемешивание позволит избавиться от пристального внимания некоторых систем сетевого мониторинга, особенно если ее использовать совместно с опциями синхронизации(параграф 11.12.1.1.3).
-M <max sockets>	Задает максимальное число сокетов, которые будут использоваться при параллельном сканировании **TCP connect()**. Такое ограничение несколько замедляет сканирование, но снижает риск возникновения критических ошибок на сканируемых хостах.
--packet_trace	Говорит программе о необходимости вывода информации о всех передаваемых пакетах. Эта опция может быть полезна при отладке и обучении.
--datadir [<каталог>]	Задает имя каталога, в котором хранятся используемые программой файлы **nmapservices**, **nmap-protocols**, **nmap-rpc** и **nmap-os-fingerprints**. Nmap сначала ищет файлы в каталоге, заданном этой опцией, после чего просматривается каталог, указанный в переменной окружения **NMAPDIR**, далее - каталог **~/nmap**, и после этого - **/usr/share/nmap**. В качестве последней попытки **nmap** просматривает текущий каталог.

11.12.1.1.3 Опции синхронизации

В большинстве случаев программа **nmap** способна подстроить параметры работы с учетом состояния сети. Однако существуют ситуации, когда используемая программой по умолчанию политика синхронизации не будет соответствовать вашим задачам. В таких случаях вы можете самостоятельно выбрать политику синхронизации с помощью опции:

-T <Paranoid|Sneaky|Polite|Normal|Aggressive|Insane>

Таблица 106. Варианты политики синхронизации nmap.

Политика		Описание
Paranoid	**0**	Медленное сканирование с целью избежать внимания систем IDS. Все операции выполняются последовательно с паузами не менее 5 минут между передачей пакетов.
Sneaky	**1**	Аналогичен предыдущему режиму, но паузы уменьшены до 15 секунд.
Polite	**2**	Эта политика синхронизации обеспечивает невысокий уровень загрузки сети и малую вероятность возникновения критических ошибок. Пакеты передаются с паузами не менее 4 мсек. Сканирование в этом режиме по крайней мере на порядок медленней, нежели в используемом по умолчанию режиме **Normal**.
Normal	**3**	Используемый по умолчанию режим, при котором сканирование осуществляется быстро и без перегрузки сети или пропуска хостов/портов.
Aggressive	**4**	Режим параллельного сканирования для уменьшения времени работы и снижения шансов на принятие ответных мер.
Insane	**5**	Этот режим пригоден только для очень скоростных сетей в тех случаях, когда потеря части информации не имеет существенного значения. Время ожидания для каждой пробы снижено до 300 мсек, а по истечение 15 минут сканирование хоста прекращается (тайм-аут).

Опция **-T0** будет задавать режим **Paranoid**, а **-T5** - **Insane**.

Кроме выбора политики, вы можете явно указать временные параметры работы программы с помощью опций, перечисленных в таблице 107.

Таблица 107. Опции синхронизации.

Опция	Описание
--host_timeout <мсек>	Задает максимальное время сканирования одного хоста до перехода к следующему адресу. По умолчанию в режиме **Normal** время сканирования не ограничивается.

2 *Обычно открыты порты **DNS** (53), **HTTP** (80), **FTP-DATA** (20).*

Опция	Описание
--max_rtt_timeout <мсек>	Задает максимальное время ожидания отклика хоста на пробный пакет **nmap** до передачи повторного пакета или констатации тайм-аута. В используемом по умолчанию режиме синхронизации это время составляет приблизительно 9 секунд (9000).
--min_rtt_timeout <мсек>	Задает минимальную паузу между передачей последовательных пробных пакетов. Обычно **nmap** сокращает интервал между передачей пакетов, если сканируемый хост отвечает достаточно быстро.
--initial_rtt_timeout <мсек>	Задает тайм-аут для первого пробного пакета. Обычно такое ограничение полезно при сканировании хостов, закрытых межсетевым экраном, в режиме -P0. Обычно **nmap** оценивает RTT по результатам **ping** и откликам на несколько первых пакетов. По умолчанию используется время ожидания 6 сек. (6000).
--max_parallelism <number>	Задает ограничение количества параллельных операций сканирования. При установке значения 1 **nmap** будет выполнять операции сканирования последовательно. Значение этого параметра влияет на все операции, которые могут выполняться в параллельном режиме (ping sweep, RPC scan и т. п.).
--min_parallelism <number>	Задает минимальное число сканируемых одновременно портов. Параллельное сканирование ускоряет процесс, но может снижать достоверность результатов.
--scan_delay <мсек>	Задает минимальную между передачей последовательных пакетов для снижения нагрузки на сеть и привлечения меньшего внимания со стороны IDS.

11.12.1.2 Выбор цели сканирования

Цель сканирования является единственным обязательным параметром команды **nmap**. В простейшем случае программа сканирует единственный хост, заданный именем или адресом IP в командной строке. Вы можете также задать сканирование подсети с указанной маской (**IP-адрес/маска**).

Nmap поддерживает широкие возможности задания цели сканирования. Например, для проверки сети класса B **192.168.*.*** вы можете указать цель как **192.168.*.***, **192.168.0-255.0-255**, **192.168.0.0/16** и даже **192.168.1-50,51-255.1,2,3,4,5-255**. Не забывайте, что многие командные процессоры при наличии в параметре символов ***** или **/** требуют использования двойных кавычек (**“”**).

Программа поддерживает и совсем экзотические варианты задания целей. Например, параметр ***.*.5.6-7** будет задавать сканирование хостов с номерами **.5.6** и **.5.7** во всех сетях класса B.

11.12.1.3 Примеры

```
nmap -v target.example.com
```
сканирование всех зарезервированных портов TCP хоста **target.example.com**; опция **-v** задает вывод подробного отчета.

```
nmap -sS -O target.example.com/24
```
сканирование **stealth SYN** каждого хоста сети, в которой находится хост **target.example.com**; при сканировании предпринимаются попытки определения ОС; использование такой команды требует привилегий **root**.

```
nmap -sX -p 22,53,110,143,4564 198.116.*.1-127
```
сканирование **Xmas tree** первой половины хостов (1 - 127) каждой сети класса C в сети класса B **198.116.0.0**; проверяются порты **ssh**, **DNS**, **pop3**, **imap** и **4564**[1].

```
nmap -v --randomize_hosts -p 80 *.*.2.3-5
```
определяет хосты с открытым портом **http** (80), имеющие значения **.2.3**, **.2.4** или **.2.5** в двух последних байтах адреса IP; адреса хостов выбираются случайно среди всех сетей класса B.

```
host -l company.com | cut  -d  -f 4 | ./nmap -v -iL -
```
копирует зону DNS для определения хостов домена **company.com** и выполняет сканирование хостов этого домена; в разных вариантах ОС детали команды могут отличаться.

1 *Напомним, что режим **Xmas** не обеспечивает сканирование хостов Windows по причине некорректной реализации стека протоколов TCP/IP, а также хостов CISCO, IRIX, HP/UX, BSDI.*

11.12.1.4 Графические интерфейсы nmap

Программа nmap поддерживает большой набор опций командной строки и может показаться слишком сложной в использовании. Для тех, кто предпочитает работать с графическими интерфейсами, существует по крайней мере два варианта.

11.12.1.4.1 Модуль Webmin

Для программы Webmin (см. главу 11.3) существует модуль **Network Utilities**, в состав которого входит интерфейс управления сканером nmap. К сожалению, это интерфейс не поддерживает всех возможностей сканера, зато он позволяет работать с удаленными хостами. Вид интерфейса показан на рисунке 11.99.

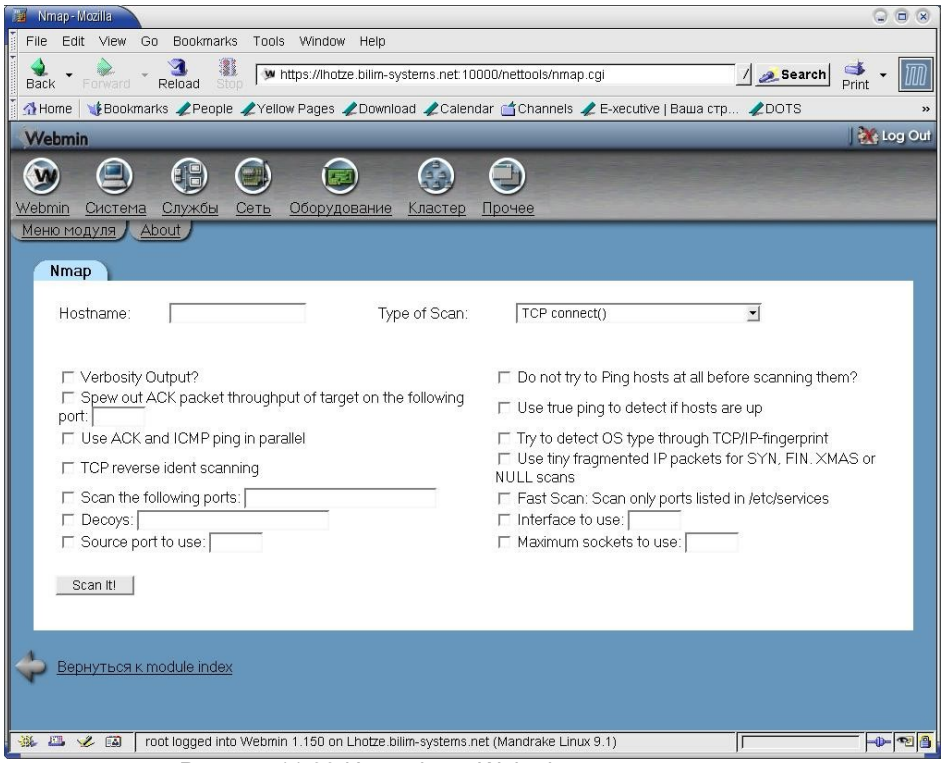


Рисунок 11.99 Интерфейс Webmin для сканера nmap

Модуль **Network Utilities** вы можете загрузить с сайта http://www.niemueller.de/webmin/modules/nettools/.

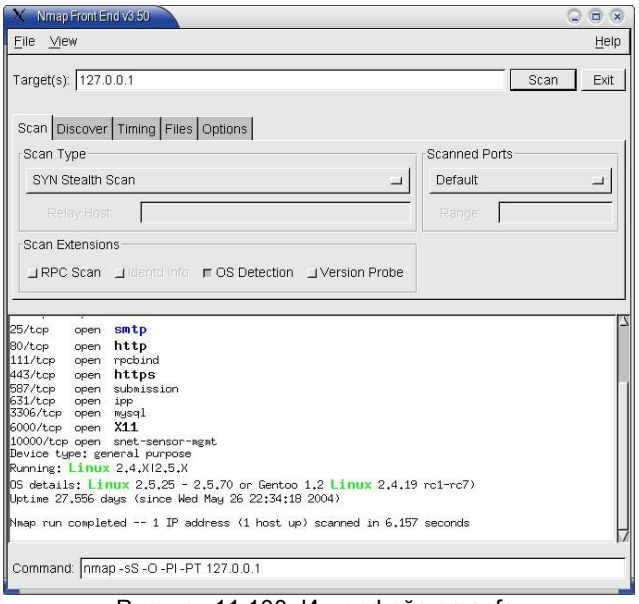


Рисунок 11.100. Интерфейс nmapfe.

11.12.1.5 nmapfe

Программа **nmapfe** (xnmap) обеспечивает графический интерфейс для сканера безопасности **nmap**, построенный на базе GTK+.

Синтаксис

```
nmapfe [опции Glib]
```

12 Приложения

12.1 Регулярные выражения (regex)

Регулярные выражения (RE[1]) в соответствии со стандартом POSIX 1003.2 могут использовать две формы - современную (например, выражения **egrep**), которая в стандарте 1003.2 называется расширенной (**extended RE**), и устаревшую (например, выражения **ed**), которая в стандарте 1003.2 называется базовой (**basic RE**). Базовый формат RE поддерживается в основном для обеспечения совместимости со старыми версиями программ, это формат обсуждается в параграфе 12.1.2 (стр. 352).

В стандарте 1003.2 не рассматриваются некоторые вопросы синтаксиса и семантики RE и некоторые аспекты реализации регулярных выражений в Linux могут оказаться несовместимыми с другими реализациями 1003.2.

12.1.1 Расширенный формат

RE представляет собой одну или несколько **субвыражений** (**branch**), разделенных символом **|**. Совпадением считается соответствие любому из заданных субвыражений.

Субвыражение включает одну или несколько объединенных между собой (concatenated) **частей** (piece). Совпадение считается соответствие всем частям субвыражения.

Часть представляет собой **атом** (atom), за которым следует один символ *****, **+**, **?**, или **ограничитель** (bound). Если после **атома** указан символ ***** совпадением будет считаться наличие любого (включая 0) количества **атомов**. Выражению **атом+** будут соответствовать строки, содержащие 1 или несколько включений **атома**. Выражению **атом?** будут соответствовать строки, содержащие не более 1 **атома**.

Ограничитель представляет собой выражение вида

```
{n1,n2}
```

т. е., заключенное в фигурные скобки целых чисел, за которым может следовать еще одно целое число, отделенное запятой. Целочисленные значения в скобках должны лежать в диапазоне от 0 до **RE_DUP_MAX**[2] и при наличии двух значений второе должно быть не меньше первого. Выражению вида **атом{i}** будут соответствовать строки, содержащие в точности **i атомов**, выражению **атом{i,}** - строки, содержащие не менее **i атомов**, а выражению **атом{i,j} -** строки, содержащие от **i** до **j**[3] **атомов**.

Атом может представлять собой:

1) регулярное выражение, заключенное в скобки **(regex)**[4];
2) пустые скобки **()**[5], **выражение в квадратных скобках** (см. ниже);
3) символ точки (**.**)[6];
4) символ **^**[7];
5) символ **$**[8];
6) символ ****, за которым следует один из символов[9] **^.[$()|*+?{**;
7) символ ****, за которым следует любой из символов, за исключением указанных в 6); вы таких случаях символ **** игнорируется и следующий за ним символ трактуется как обычно (см. 8);
8) любой символ, не имеющий специального значения[10].

Открывающая фигурная скобка **{**, за которой символ, отличный от цифры или не являющийся частью целого числа, не является началом ограничителя и трактуется как обычный символ. В конце регулярного выражения недопустимо использование комбинации **\{**.

Выражение в квадратных скобках (bracket expression или **BE**) представляет собой последовательность символов, помещенную внутрь квадратных скобок **[]**. Обычно такому выражению соответствует любой одиночный символ из числа указанных внутри скобок. Если после открывающей скобки помещен символ **^**, выражению будет соответствовать любой одиночный символ, не указанный в скобках. Если два соседних символа в скобках разделены дефисом (**-**), это задает сокращенную запись для всех символов между двумя указанными граничными (включая указанные символы). Например, выражению **[0-9]** будет соответствовать любая десятичная цифра. В Linux для указания диапазона символов не допускается использование записей типа **a-c-e**. Трактовка диапазона зависит от используемой кодировки символов, поэтому для переносимых систем нужно весьма аккуратно относиться к использованию диапазонов в выражениях.

Для включения символа **]** в выражение в квадратных скобках укажите этот символ первым в списке[11], а для

1 *Regular expressions.*
2 *В Linux **RE_DUP_MAX** = 255.*
3 *Включительно.*
4 *Соответствие выражениям, соответствующим регулярному выражению **regex** в скобках .*
5 *Соответствие пустой строке.*
6 *Соответствие одному (любому) символу.*
7 *Соответствие пустой строке в начале [строки].*
8 *Соответствие пустой строке в конце [строки].*
9 *Символы **^.[$()|*+?{** имеют специальное значение в регулярных выражениях (мета-символы) и знак **** перед мета-символом означает трактовку последнего как обычного символа.*
10 *Соответствие указанному символу.*
11 *Перед ним может быть указан символ **^**.*

включения символа **-** укажите его первым или последним. Для использования символа **-** в качестве старта диапазона, поместите его внутрь специальных скобок **[.** и **.]**[1], чтобы сделать данный символ элементом сравнения, как описано ниже. За исключением этого случая и некоторых других комбинаций с использованием символа **[**, описанных в следующих абзацах, все остальные специальные символы (включая \) теряют свое специальное значение при использовании внутри **BE**[2].

Внутри **BE** объект сравнения (символ, последовательность символов, используемая при сравнении как 1 символ, или имя), помещенный внутрь специальных скобок **[.** и **.]** используется как последовательность символов, входящих в этот элемент[3]. Последовательность представляет собой один элемент списка **BE**. Выражения **BE**, содержащие элементы сравнения из нескольких символов, могут соответствовать более, чем одному символу. Например, если последовательность содержит элемент сравнения **ch**, то регулярному выражению **[[.ch.]]*c** будут соответствовать первые пять символов строки **chchcc** (т. е., подстрока **chchc**).

Внутри **BE** объект сравнения, заключенный в специальные скобки **[=** и **=]**, является классом эквивалентности, указывающим что последовательность символов всех элементов сравнения эквивалентна данному символу. Например, если **o** и **^** являются членами класса эквивалентности, записи **[[=o=]]**, **[[=^=]]** и **[o^]** будут иметь одинаковый смысл. Класс эквивалентности не может использоваться в качестве конечной точки диапазона.

Внутри **BE** имя класса символов, заключенное в специальные скобки **[:** и **:]** означает список всех символов, относящихся к данному классу. Стандартные классы символов перечислены в таблице 108. Эти классы символов системы определены в **wctype**. Локальные установки[4] могут менять состав символов отдельных классов. Класс символов не допускается использовать в качестве конечной точки диапазона.

Таблица 108. Стандартные классы символов.

Класс	Описание
`alnum`	Буквы и цифры
`alpha`	Буквы
`blank`	Пробел и символ табуляции
`cntrl`	Управляющие символы
`digit`	Цифры
`graph`	Символы псевдо-графики
`lower`	Строчные буквы
`print`	Печатаемые символы
`punct`	Знаки пунктуации
`space`	Пробел, символы табуляции, перевода строки, возврата каретки, перевода страницы.
`upper`	Прописные буквы
`xdigit`	Шестнадцатеричные цифры (0-9, A-F).

Существует два варианта специального использования **BE** - **[[:<:]]** и **[[:>:]]** обозначающих пустую строку в начале и в конце слова, соответственно. Слово определяется как последовательность элементов (символов) слова (word character), перед которой и после которой нет других элементов слова. Элементом слова может быть символ класса **alnum** или символ подчеркивания (_). Это расширение Linux совместимо со стандартом POSIX 1003.2, но не описано в нем, поэтому его не следует использовать в переносимых приложениях.

Если RE может соответствовать более, чем одной подстроке данной строки, выбирается та подстрока, которая находится ближе к началу строки. Если RE может соответствовать нескольким подстрокам, начинающимся из одной точки, выбирается наиболее длинная подстрока. Вышесказанное относится также к субвыражениям. Отметим, что субвыражение более высокого уровня имеет больший приоритет.

Длина совпадения измеряется в символах, а не количеством элементов сравнения. Пустая строка считается более длинной, нежели отсутствие совпадения. Например, выражению **bb*** соответствует подстрока **bbb** строки **abbbc**, выражению **(wee|week)(knights|nights)** - все 10 символов строки **weeknights**. Когда **(.*).*** сравнивается с **abc** выражение в скобках соответствует всем трем символам, а при сравнении **(a*)*** со строкой **bc** все выражение **RE** и субвыражение в скобках соответствуют пустым строкам.

Если регистр символов не принимается во внимание, это эквивалентно преобразованию каждой буквыв выражение в скобках, содержащее строчную и прописную букву. Например, символ **x**, по сути, трансформируется в выражение **[xX]**. Если символ уже был частью **BE**, в это выражение просто добавляется парный символ (т. е., **[x]** становится **[xX]**, а **[^x]** - **[^xX]**).

Размер регулярных выражений **RE** не ограничивается, однако в переносимых программах не следует применять **RE**, размер которых превышает 256 байтов.

12.1.2 Базовый формат

Устаревший базовый формат RE отличается тем, что символы **|**, **+** и **?** не имеют специального значения (являются обычными символами). Обозначениями границ служат комбинации **\{** и **\}**, а сами скобки **{** и **}** - обычными символами. Скобками для вложенных выражений служат комбинации **\(** и **\)**, а символы **(** и **)** трактуются как обычные. Символ **^** не имеет специального значения за исключением его присутствия в начале **RE** или в начале выражения в скобках. Символ **$** также не имеет специального значения за исключением случаев, когда он указан в конце **RE** или субвыражения в скобках. Звездочка (*) в начале **RE** или субвыражения в скобках считается обычным символом.

Базовый формат использует дополнительный тип атомов - символ \, за которым следует отличная от 0 десятичная цифра (**d**), соответствует такой же последовательности символов, которой соответствует субвыражение в скобках с номером **d** (нумерация субвыражений - слева направо от открывающей скобки)Например, **\([bc]\)\1** соответствует **bb** или **cc**, но не **bc**.

1 *Т. е. **[.-.]***
2 *Т. е. трактуются как обычные символы.*
3 *Т. е. скобки **[.** и **.]** не принимаются во внимание при операции сравнения.*
4 *Вы можете получить информацию о локальных установках системы с помощью команды **man locale**.*

12.1.3 Функции для работы с виртуальными выражениями

Системы, соответствующие стандарту POSIX, поддерживают ряд функций для работы с регулярными выражениями (**regcomp**, **regexec**, **regerror**, **regfree**) описанных в заголовочном файле **<regex.h>**.

```
#include <sys/types.h>
#include <regex.h>

int regcomp(regex_t *preg, const char *regex, int cflags);
int regexec(const regex_t *preg, const char *string, size_t nmatch, regmatch_t pmatch[],
int eflags);
size_t regerror(int errcode, const regex_t *preg, char *errbuf, size_t errbuf_size);
void regfree(regex_t *preg);
```

12.1.3.1 Компиляция регулярных выражений в POSIX

Для преобразования регулярных выражений в формат, подходящий для поиска с помощью функции **regexec** служит функция **regcomp**. Функция вызывается с указателем на буфер для записи шаблона поиска, указателем на регулярное выражение (regex), и флагами (cflags), используемыми для задания типа компиляции. Все операции поиска для регулярного выражения должны осуществляться с использованием полученного при компиляции шаблона, поэтому функцию **regexec** следует во всех случаях вызывать с адресом созданного и инициализированного буфера для хранения шаблона. Параметр **cflags** может быть битовой маской или содержать предопределенные значения флагов (см. таблицу 109).

Таблица 109. Флаги компиляции regcomp.

Ключ	Описание
REG_EXTENDED	Задает использование расширенного синтаксиса регулярных выражений[1] при интерпретации **regex**. При отсутствии этого флага используется базовый синтаксис[2].
REG_ICASE	Не различать регистр символов.
REG_NOSUB	Поддержка адресов найденных совпадений подстрок не требуется. При использовании этого флага параметры **nmatch** и **pmatch** игнорируются.
REG_NEWLINE	Символ новой строки не соответствует оператору **Match-any-character** (любой символ). Список несовпадения (**[^...]**), не содержащий символа новой строки, не соответствует новой строке. Оператору совпадения начала строки (**^**) будет соответствовать пустая строка сразу же после символа новой строки, независимо от наличия в параметре **eflags** функции **regexec** флага **REG_NOTBOL**. Оператору совпадения конца строки (**$**) будет соответствовать пустая строка непосредственно перед символом новой строки, независимо от наличия в параметре **eflags** функции **regexec** флага **REG_NOTEOL**.

12.1.3.2 Совпадения POSIX

Для проверки совпадения завершающихся нуль-символом строк с созданным при компиляции шаблоном **preg** используется функция **regexec**. Параметры **nmatch** и **pmatch** служат для передачи информации о местах поиска совпадений. Параметр **eflags**, определяющий правила поиска совпадений, может содержать битовую маску или флаги (один или оба) **REG_NOTBOL** и **REG_NOTEOL** .

REG_NOTBOL

Оператор совпадения с началом строки никогда не дает позитивного результата (если при компиляции не был задан флаг **REG_NEWLINE**, описанный выше). Этот флаг может использоваться с различными частями строки, переданной функции **regexec**, начинающимися с последовательности символов, которая не может быть интерпретирована как начало новой строки.

REG_NOTEOL

Оператор совпадения с концом строки никогда не будет давать позитивного результата (если при компиляции не был задан описанный выше флаг **REG_NEWLINE**).

12.1.3.2.1 Смещения совпадающих подстрок

Если при компиляции шаблона поиска не был установлен флаг REG_NOSUB (стр. 353), можно получить информацию об адресах совпадающих подстрок. Размер структуры **pmatch** должен обеспечивать размещение по крайней мере **nmatch** элементов. Эти элементы заполняются функцией **regexec** адресами совпадающих подстрок. Неиспользуемые элементы структуры будут содержать значение -1.

Структура **regmatch_t**, используемая для хранения адресных элементов, определена в файле **regex.h**.

```
typedef struct
{
        regoff_t rm_so;
```

1 *POSIX Extended Regular Expression syntax.*
2 *POSIX Basic Regular Expression.*

```
        regoff_t rm_eo;
} regmatch_t;
```

Каждый элемент **rm_so**, отличный от -1, указывает смещение начала следующего наибольшего совпадения подстроки в данной стоке, а **rm_eo** показывает смещение конца совпадающей подстроки.

12.1.3.3 Сообщения об ошибках POSIX

Для возврата кодов ошибок при работе **regcomp** и **regexec** используется функция **regerror**. Этой функции передается код ошибки **errcode**, буфер шаблона поиска **preg**, указатель на буфер символьной строки **errbuf** и размер этого буфера **errbuf_size**. Функция возвращает размер буфера **errbuf** , требуемого для записи сообщения об ошибке. Если оба значения **errbuf** и **errbuf_size** отличны от нуля, в буфер **errbuf** помещается начало (**errbuf_size - 1**) текстового сообщения об ошибке, завершаемого нуль-символом.

12.1.3.4 Освобождение буферов шаблонов поиска POSIX

Функция **regfree**, получив в качестве параметра буфер с шаблоном поиска, будет освобождать выделенную для этого буфера при компиляции память.

12.1.3.5 Возвращаемые значения

Функция **regcomp** возвращает нулевое значение при успешной компиляции и код ошибки в противном случае.

Функция **regexec** возвращает нулевое значение при успешном поиске и код **REG_NOMATCH**, если не найдено соответствия.

12.1.3.5.1 Коды ошибок

Коды ошибок, возвращаемые функцией **regcomp**, перечислены в таблице 110.

Таблица 110. Коды ошибок regcomp.

Код	Описание
REG_BADRPT	Некорректное использование операций повтора (например, *) в качестве первого символа.
REG_BADBR	Некорректное использование оператора обратной ссылки (back reference).
REG_EBRACE	Несоответствие скобок в операторе интервала.
REG_EBRACK	Несоответствие скобок в операторе списка.
REG_ERANGE	Некорректное использование оператора диапазона (например, конечная точка предшествует начальной).
REG_ECTYPE	Неизвестное имя класса символов.
REG_ECOLLATE	Некорректный элемент сравнения.
REG_EPAREN	Несоответствие круглых скобок в групповых операторах.
REG_ESUBREG	Некорректная обратная ссылка на субвыражение.
REG_EEND	Неизвестная ошибка. Этот тип сообщений отсутствует в POSIX.2.
REG_EESCAPE	Завершающий символ \.
REG_BADPAT	Некорректное использование шаблонов (например, группы или списка).
REG_ESIZE	Компилируемое регулярное выражение требует для шаблона буфер замером более 64 кбайт. Этот тип сообщений отсутствует в POSIX.2.
REG_ESPACE	Нехватка памяти.

12.2 Параметры SysCtl

Интерфейс SysCtl и утилита **sysctl** предназначены для настройки конфигурационных параметров ядра в процессе работы операционной системы. Для использования возможностей SysCtl при компиляции ядра должна быть включена опция **Sysctl support** (параграф 4.4.1.2.5 на стр. 64). Параметры конфигурации хранятся в файлах дерева **/proc**, описанного ниже (стр. 354).

Утилита **sysctl** (параграф 11.2.8 на стр. 217) используется для просмотра и изменения конфигурационных параметров ядра на работающей системе. Доступные для этой утилиты файлы хранятся в ветви **/proc/sys** (стр. 362). Для чтения параметров SysCtl в большинстве случаев можно использовать команду **cat <имя файла>**, а менять значения параметров (когда это допустимо) можно с помощью команды

```
echo <значение> > <имя файла>
```

12.2.1 Виртуальная файловая система /proc

В ОС Linux виртуальная файловая система **/proc** содержит файлы с текущей информацией о процессах и системе в целом. Файловая система создается в оперативной памяти компьютера в процессе загрузки и используется в качестве интерфейса обмена данными со структурами ядра. Использование файлов **/proc/*** в большинстве случаев

позволяет избавиться от необходимости чтения и записи в устройство **/dev/kmem**. Большинство расположенных в структуре **/proc** файлов доступны пользователям только для чтения, а для записи требуют полномочий root.

Для поддержки виртуальной файловой системы **/proc** при компиляции ядра должна быть включена опция **/proc file system support** (параграф 4.4.1.5.1 на стр. 67).

12.2.1.1 Файлы параметров системы

В каталоге **/proc** храниться группа файлов, содержащих параметры, определяющие работу ядра Linux и системы в целом. Основные файлы, содержащие такие параметры перечислены в таблице 111.

Таблица 111. Параметры ядра и системы в целом.

Файл	Описание
apm	Параметры состояния и опции системы управления питанием APM[1].
cmdline	Аргументы, переданные при загрузке ядру Linux. Для передачи аргументов обычно используются менеджеры загрузки типа **lilo** (параграф 2.1.2.1 на стр. 33).
config.gz	Сжатая копия конфигурационного файла, использованного для компиляции рабочего ядра. Этот файл присутствует, если при компиляции ядра была включена опция **IKCONFIG_PROC** (параграф 4.4.1.2.9.1 на стр. 65).
cpufreq	Параметры управления частотой процессора.
cpuinfo	Набор параметров процессора, зависящий системной архитектуры и используемого типа процессора. Два поля этого файла поддерживаются независимо от архитектуры: **cpu** - указывает тип процессора; **bogomips** - производительность процессора, определенная во время инициализации ядра. В системах SMP файл содержит набор сведений для каждого процессора.
crypto	Информация об используемых в системе средствах шифрования.
devices	Текстовый список старших номеров для символьных и блоковых устройств, поддерживаемых ядром для данной системы. Этот список может использоваться сценариями MAKEDEV при создании устройств.
diskstats	Информация о состоянии имеющихся в системе разделах дисковых устройств. Сведения о полях записей этого файла можно найти в файле **Documentation/iostats.txt** дистрибутива ядра Linux.
dma	Список зарегистрированных и используемых каналов ISA DMA (прямой доступа к памяти).
execdomain	Список доменов исполнения[2], поддерживаемых ядром Linux, и диапазон поддерживаемых “индивидуальностей” (ABI personality).
fb	Список устройств **frame buffer**[3] с номерами и именами обслуживающих устройства драйверов.
filesystems	Список файловых систем, поддержка которых была включена при компиляции ядра. Этот список может использоваться командой **mount** для выбора файловой системы, если последняя не была указана в командной строке или конфигурационном файле.
interrupts	Информация о количестве прерываний с момента загрузки системы для каждого IRQ.
iomem	Карта распределения адресов системной памяти.
ioports	Список зарегистрированных диапазонов портов ввода-вывода.
kallsyms	Файл со списком определений экспортируемых ядром символьных имен, которые используются программами для работы с загружаемыми модулями типа (например, **insmod**. В более старых версиях ядра этот файл может называться **ksyms**.
kcore	Файл представляющий физическую память системы в elf-формате **core**. Используя этот псевдофайл и ядро, из которого не удалены таблицы символов(**/usr/src/linux/tools/zSystem**), **GDB** может проверить текущее состояние любой структуры данных в ядре. Размер файла **kcore** равен размеру ОЗУ + 4 кбайт.
kmsg	Файл буфера сообщений ядра. Недопустимо чтение этого файла при запущенном процессе **syslog**, использующем системные вызовы **syslog** для доступа к сообщениям ядра. Информацию из файла **kmsg** можно прочитать с помощью команды **dmesg** (параграф 11.2.1 на стр. 206).
loadavg	Параметры средней загрузки, определяемые количеством заданий в очереди на запуск (состояние **R**) или ожидающих выполнения дисковых операций ввода-вывода (состояние **D**), усредненным за 1, 5 и 15 минут. Эти же параметры средней загрузки выводит команда **uptime** (параграф 11.1.1.9 на стр. 192) и другие программы.
locks	Список заблокированных файлов.

1 *Advanced Power Management.*
2 *Рассматривайте “домен исполнения” (**execution domain**) как характеристику “индивидуальности” ОС. В Linux могут использоваться бинарные форматы других ОС (Solaris, UnixWare, FreeBSD). Изменяя “индивидуальность” работающей в Linux задачи, программист может сменить способ трактовки ОС системных вызовов из данной задачи. За исключением домена исполнения **PER_LINUX** программы могут быть реализованы как динамически загружаемые модули.*
3 *Для поддержки устройств **frame buffer** при компиляции ядра должна быть включена опция **CONFIG_FB**.*

Файл	Описание
mdstat	Файл информации о состоянии системы RAID.
meminfo	Файл, содержащий информацию о свободной и используемой памяти (как ОЗУ, так и области подкачки). **MemTotal** - размер ОЗУ; **MemFree** - размер свободной части ОЗУ; **Buffers** - объем ОЗУ, используемый для буферов; **Cached** - объем ОЗУ, используемый для кэширования; **SwapCached** - объем области подкачки, используемый для кэширования; **Active** - общий объем активно используемой памяти; **Inactive** - объем памяти, которая давно не использовалась и может быть освобождена; **HighTotal** - объем памяти, не отображенной непосредственно в пространство ядра; **HighFree** - объем свободной памяти, не отображенной напрямую в пространство ядра; **LowTotal** - общий объем памяти, отображенной непосредственно в пространство ядра; **LowFree** - объем свободной памяти, отображенной напрямую в пространство ядра; **SwapTotal** - размер файла подкачки; **SwapFree** - размер свободной части файла подкачки; **Dirty** - объем памяти, для которой ожидается запись в область подкачки; **Writeback** - объем памяти, сбрасываемой на диск (в файл подкачки); **Mapped** - размер файлов, отображенных в память с использованием функции **mmap**; **Slab** - размер кэша внутренних структур данных ядра; **Committed_AS** - объем памяти, который может обеспечить гарантию 99,99% против переполнения при данном уровне загрузки. **PageTables** - объем памяти, выделенной под таблицы страниц самого нижнего уровня; **ReverseMaps** - количество выполненных обратных отображений; **VmallocTotal** - общий размер области памяти **vmalloc**; **VmallocUsed** - размер использованной части области памяти **vmalloc**; **VmallocChunk** - размер максимального свободного блока в области **vmalloc**. Данные из этого файла используются программой **free** (параграф 11.2.3 на стр. 207).

misc	Список драйверов, зарегистрированных устройством со старшей частью номера 10 (**misc**)
	
	`135 rtc`
	`  1 psaux`
	`134 apm_bios`
	Первая колонка списка содержит младшую часть номера устройства.

modules	Список имен загруженных модулей ядра. Этот список можно получить с помощью команды **lsmod** (параграф 12.17.3 на стр. 408).
mounts	Символьная ссылка на файл **/proc/self/mounts**, содержащий список смонтированных файловых систем. Этот список выводится по команде **mount**.
mtrr	Текущее содержимое регистров MTRR[4], используемых системой. Эти регистры используются в системах с процессорами Pentium Pro и выше для управления доступом процессора к оперативной памяти. Отметим, что при использовании видео-плат PCI или AGP корректная настройка MTRR может повысить производительность системы на 150% и более.
partitions	Список дисковых разделов, содержащий старший и младший номер версии, количество блоков и имя для каждого раздела имеющихся в системе дисков.
pci	Список устройств PCI, присутствующих в системе с конфигурационными параметрами каждого устройства.
slabinfo	Список выделенных в системе именованных блоков памяти (slab). Ядро Linux версий старше 2.2 использует пулы таких блоков для распределения памяти выше страничного уровня. Объекты общего пользования могут иметь свои slab-пулы. Информацию о выделенных блоках можно получить с помощью команды **slabinfo**.
stat	Файл со статистической информацией о работе ядра и системы в целом.
swaps	Список организованных в системе областей подкачки (swap) и их параметров.
uptime	Файл, содержащий информацию о времени работы системы с момента ее последней загрузки. Первое число показывает количество секунд с момента загрузки, а второе число - продолжительность пребывания в состоянии **idle** (безделье) с момента загрузки.
version	Файл, содержащий информацию о номере версии загруженного ядра Linux, компилятора **gcc** и операционной системы.
vmstat	Информация о распределении виртуальной памяти.

12.2.1.2 Каталоги процессов

Файловая система **/proc** содержит множество каталогов с численными именами - эти каталоги создаются для каждого запущенного в системе процесса. Имя каталога определяется идентификатором (PID) соответствующего процесса. Каждый из таких каталогов содержит ряд подкаталогов файлов, кратко описанных ниже (таблица 112).

4 *Memory Type Range Register.*

Таблица 112 Файлы в подкаталогах процессов

Имя	Назначение
auxv	Начальные значения aux-вектора процесса, передаваемые операционной системой динамическому компоновщику в качестве стартовых значений.
cmdline	Содержимое командной строки, использованной для запуска процесса[1]. Командная строка в файле завершается нуль-символом и не содержит символа новой строки.
cwd	Ссылка на текущий рабочий каталог процесса. Для определения рабочего каталога процесса можно воспользоваться командой типа **cd /proc/<PID>/cwd; /bin/pwd**[2]
environ	Переменные окружения, используемые процессом. Записи в этом файле разделяются нуль-символами, а в конце файла также может использоваться нуль-символ. Для просмотра переменных окружения процесса можно воспользоваться командой типа **(cat /proc/<PID>/environ; echo) | tr "\000" "\n"**
exe	Символьная ссылка на исполняемый файл процесса. Использование команды readlink с именем этого файла в качестве параметра возвращает полное имя исполняемого процесса в системах Linux с ядром версии 2.2 и выше. В Linux 2.0 и более старых версиях возвращается строка вида **[устройство]:индексный_дескриптор** содержащая индексный дескриптор файла и номер устройства, на котором файл хранится. С символьной ссылкой **exe** можно работать как с обычным файлом. Например, команда **/proc/<PID>/exe** приведет к запуску новой копии процесса. Для поиска файла может использоваться команда **find -inum <PID>**
fd	Подкаталог, содержащий по одной символьной ссылке для каждого открытого процессом файла. Имя ссылки соответствует номеру файлового дескриптора для открытого файла, а сама ссылка указывает на открытый процессом файл. 0 указывает на стандартное устройство ввода, 1 - на стандартное устройство вывода, 2 - на стандартный вывод ошибок и т. д. Символьные ссылки в подкаталоге **fd** позволяют обманывать программы, которые в качестве входного файла не принимают **stdin** или в качестве выходного - **stdout**. В командной строке такой программы можно просто указывать **/proc/self/fd/0** в качестве входного файла и в качестве выходного **/proc/self/fd/1**. Отметим, что предложенная хитрость не сработает для программ, использующих при файловых операциях команду **seek**, поскольку такие операции не поддерживаются для файлов стандартного ввода-вывода. Файлы **/proc/self/fd/N** в Linux - это почти то же самое, что и файлы **/dev/fd/N**. Фактически, большинство сценариев MAKEDEV в Linux делает символьные ссылки **/dev/fd** на **/proc/self/fd**.
maps	Файл отображения используемых программой областей памяти с указанием прав доступа. Этот файл использует формат: **address perms offset dev inode** **00000000-0002f000 r-x-- 00000400 03:03 1401** **0002f000-00032000 rwx-p 0002f400 03:03 1401** **00032000-0005b000 rwx-p 00000000 00:00 0** **60000000-60098000 rwx-p 00000400 03:03 215** **60098000-600c7000 rwx-p 00000000 00:00 0** **bfffa000-c0000000 rwx-p 00000000 00:00 0** Колонка **address** указывает используемое процессом адресное пространство памяти, perms - задает права доступа к данной области памяти (**r** - чтение, **w** - запись, **x** - исполнение, **s** - возможность использования другими процессами, **p** - приватный блок, копируемый при записи). Колонка **offset** задает текущее смещение от начала области памяти (указатель позиции), **dev** задает устройство, а **inode** указывает индексный дескриптор[3]. Ядро Linux версии 2.2 использует дополнительное поле, в котором указывается (при возможности) полное имя файла.
mem	Файл соответствующий содержимому страниц памяти процесса и обеспечивающий доступ к памяти с помощью функций **open**, **read** и **fseek**[4].

1 *Для сброшенных на диск (свопинг) и зомби-процессов, в этом файле не содержится никакой информации и при попытке чтения такого файла возвращается строка нулевой длины.*
2 *Первая часть команды обеспечивает переход в рабочий каталог процесса, а команда **/bin/pwd** выводит имя текущего каталога, каким в результате исполнения первой части команды стал рабочий каталог процесса. Отметим, что использование встроенной в командный процессор команды **pwd** может приводить к некорректным результатам.*
3 *0 говорит о том, что с данной областью памяти не связано никакого дескриптора*
4 *Эти операции поддерживаются только новыми версиями ядра Linux.*

Имя	Назначение
mounts	Список смонтированных в файловых систем. Содержимое этого файла выводится по команде **mount**.
root	Символьная ссылка на корневой каталог, используемый данным процессом[5].
stat	Файл с информацией о состоянии процесса. Эти файлы используются командой **ps** (параграф 11.2.6 на стр. 210) для сбора информации о процессах. Поля файла stat описаны ниже (таблица 113 на стр. 358) в порядке их следования в файле.
statm	Файл с информацией об используемых процессом страницах памяти. Поля файла перечислены ниже: **size** общий размер программы **resident** размер резидентной части **share** разделяемые страницы **trs** текст (код) **drs** данные/стек **lrs** библиотека **dt** число “грязных” страниц
status	Файл с информацией из **stat** и **statm** в более удобном для человеческого восприятия формате.
wchan	Имя функции ядра, в которой процесс в данный момент “спит”.

12.2.1.2.1 Поля файла stat

Таблица 113 Поля файлов /proc//stat*

Имя	Формат[1]	Описание
pid	**%ld**	Идентификатор процесса.
comm	**%ls**	Имя исполняемого файла в круглых скобках.
state	**%lc**	Состояние процесса, выраженное одним из символов **RSDZTW** (**R** - работает, **S** - спит и ждет прерывания, **D -** спит на диске, не ожидая прерывания (свопинг), **Z** - “зомби”, **T -** трассируется или остановлен (по сигналу), **W** - перемещение в памяти - paging).
ppid	**%ld**	Идентификатор родительского процесса.
pgrp	**%ld**	Идентификатор группы для процесса.
session	**%ld**	Идентификатор сессии для процесса.
tty_nr	**%ld**	Терминал **tty**, используемый процессом.
tpgid	**%ld**	Идентификатор группы, владеющей в настоящий момент терминалом, к которому подключен данный процесс.
flags	**%lu**	Флаги процесса. Флагу **math** (арифметический сопроцессор) соответствует десятичное значение 4, а флагу бит трассировки (**traced**) - десятичное число 10.
minflt	**%lu**	Количество несущественных сбоев в работе процесса, не требовавших загрузки страниц памяти с диска.
cminflt	**%lu**	Количество несущественных сбоев в работе данного процесса или порожденных им процессов.
majflt	**%lu**	Количество существенных сбоев в работе процесса, требовавших загрузки страниц памяти с диска.
cmajflt	**%lu**	Количество несущественных сбоев в работе данного процесса или порожденных им процессов.
utime	**%ld**	Время в jiffy[2], которые данный процесс работал в пользовательском режиме.
stime	**%ld**	Время в jiffy, которые данный процесс работал в режиме **kernel**.
cutime	**%ld**	Время в jiffy, которые данный процесс и порожденные им процессы работали в пользовательском режиме.
cstime	**%ld**	Время в jiffy, которые данный процесс и порожденные им процессы работали в режиме **kernel**.
priority	**%ld**	Стандартное значение **nice** плюс 15. Это значение никогда не бывает отрицательным в ядре.
nice	**%lu**	Уровень приоритета от -19 (минимальный приоритет) до 19 (nicest - максимальный приоритет).

5 *В UNIX-системах поддерживается парадигма корневого каталога файловой системы процесса, который может задаваться с помощью системного вызова **chroot**.*

1 *Идентификатор формата, используемый функцией **scanf**.*

2 *Специальная единица времени, используемая ядрами Linux. Для процессоров x86 составляет приблизительно 10 мсек.*

Имя	Формат	Описание
0	%ld	Пустышка взамен использованного раньше поля.
itrealvalue	%lu	Время в jiffy до передачи процессу следующего сигнала **SIGALRM** от таймера интервалов.
starttime	%ld	Время в jiffy, которое прошло с момента загрузки системы до старта процесса.
vsize	%lu	Размер виртуальной памяти в байтах.
rss	%lu	**Resident Set Size** (размер резидентной части) - количество страниц реальной памяти, занимаемых процессом, за вычетом 3 страниц, используемых для администрирования. Резидентные страницы могут включать текст, данные или программный стек. В резидентную часть не включаются страницы, которые не были загружены по запросу или находятся в области подкачки.
rlim	%lu	Текущее ограничение размера резидентной части (**rss**) в байтах. Обычно это поле содержит значение 4294967295.
startcode	%lu	Нижняя граница адресов программного кода.
endcode	%lu	Верхняя граница адресов программного кода.
startstack	%lu	Адрес начала стека.
kstkesp	%lu	Текущее значение **esp** (укаатель стека), найденное для данного процесса на странице стека в ядре.
kstkeip	%lu	Текущее значение EIP (указатель команд процессора).
signal	%ld	Битовое отображение ожидающих сигналов (обычно 0).
blocked	%ld	Битовое отображение блокированных сигналов (обычно 0, для командных интерпретаторов - 2).
sigignore	%ld	Битовое отображение игнорируемых сигналов.
sigcatch	%ld	Битовое отображение перехватываемых сигналов.
wchan	%lu	"Канал", в котором ожидает процесс. Канал указывается адресом системной функции, имя которой можно определить с помощью системного списка имен (System.map)[3].

12.2.1.3 Каталог bus

Этот каталог содержит несколько подкаталогов, соответствующих системным шинам компьютера. В каждом из таких каталогов содержатся подкаталоги и файлы, связанные с присутствующими в системе устройствами. Для просмотра имеющихся в системе устройств PCI и USB можно воспользоваться утилитами **lspci** и **lsusb**, соответственно, а команда **lshw** покажет вам полный список имеющихся в системе устройств.

12.2.1.4 Каталог driver

Этот каталог содержит информацию для некоторых драйверов, используемых ядром Linux.

Обычно в каталоге содержится файл **rtc** с информацией от драйвера системных часов RTC. Файл имеет вид:

```
rtc_time        : 14:44:48
rtc_date        : 2004-07-22
rtc_epoch       : 1900
alarm           : 01:57:**
DST_enable      : no
BCD             : yes
24hr            : yes
square_wave     : no
alarm_IRQ       : no
update_IRQ      : no
periodic_IRQ    : no
periodic_freq   : 1024
batt_status     : okay
```

12.2.1.5 Каталог fs

Этот каталог содержит подкаталоги с информацией о некоторых файловых системах. Основная информация о файловых системах хоста храниться в подкаталоге **/proc/sys/fs** (стр. 363).

12.2.1.6 Каталог ide

Каталог **ide** создается в системах, использующих устройства IDE. Этот каталог содержит подкаталоги для каждого канала IDE и каждого подключенного устройства IDE. Подкаталоги включают файлы из приведенного ниже списка.

3 *В новых версиях Linux имя этой функции можно увидеть в файле **/proc/<PID>/wchan**.*

cache	размер буфера кэширования в килобайтах
capacity	число секторов
driver	версия драйвера
geometry	физическая и логическая геометрия устройства
media	тип среды
model	заданный производителем номер модели
settings	параметры устройства

Получить информацию о параметрах устройств можно с помощью утилиты **hdparm**, обеспечивающей более понятное для человека представление данных.

12.2.1.7 Каталог irq

Этот каталог используется в многопроцессорных системах для связывания IRQ с определенными процессорами из числа имеющихся в системе CPU.

Для каждого IRQ создается подкаталог, имя которого определяется номером IRQ.

12.2.1.8 Каталог net

Каталог содержит группу подкаталогов и файлов с различными параметрами сетевых устройств и статистическими данными. Все файлы используют формат ASCII и прекрасно читаются утилитой **cat**. Программа netstat (параграф 11.1.2.5 на стр. 199) также обеспечивает эффективный доступ к информации, содержащейся в структуре **/proc/net**.

Файлы системного уровня из каталога /proc/net перечислены в таблице 114. Параметры, соответствующие протоколу IPv6, опущены, равно как и параметры фильтров **ipfwadm**/**ipcahins**.

Таблица 114. Файлы каталога /proc/net.

Файл	Описание
arp	Таблица ARP, поддерживаемая ядром Linux для преобразования адресов сетевого уровня в MAC-адреса. Таблица адресов имеет формат: **IP address HW type Flags HW address Mask Device** 193.111.91.2 0x1 0x2 00:05:5D:00:33:41 * eth0 193.111.91.145 0x1 0x2 00:03:BA:0A:37:08 * eth0 193.111.91.3 0x1 0x2 00:A0:CC:79:16:D4 * eth0 212.48.192.241 0x1 0x2 00:07:B3:15:60:1A * eth1 193.111.91.7 0x1 0x2 00:A0:CC:79:16:D4 * eth0 193.111.91.137 0x1 0x2 00:03:BA:0A:37:08 * eth0 62.141.127.101 0x1 0x2 00:04:28:1F:48:1C * eth2 193.111.91.132 0x1 0x2 00:03:BA:0A:37:08 * eth0 Поле **IP address** содержит адреса IPv4, **HW type** - идентификаторы типов оборудования в соответствии с RFC 826, **Flags** - флаги структуры ARP (см. **/usr/include/linux/if_arp.h**), **HW address** - аппаратный (MAC) адрес интерфейса, соответствующий адресу IP-адреса. Символьные имена в поле аппаратного адреса берутся из файла **/etc/ethers**.
dev	Статистика работы сетевых интерфейсов (количество принятых и отправленных пакетов и байтов, количество ошибок и конфликтов и т. п.). Информация из этого файла используется утилитой **ifconfig** (параграф 11.1.2.3 на стр. 196) для вывода отчета о состоянии устройств. Формат данных в файле показан на рисунке 12.1.
dev_mcast	Список multicast-групп канального уровня, прослушиваемых каждым устройством. **indx ifterface_name dmi_u dmi_g dmi_address** 2 eth0 1 0 01005e000001 3 eth1 1 0 01005e000001 4 eth2 1 0 01005e000001
igmp	Записи протокола IGMP, описанные в файле **/net/ipv4/igmp.c** дистрибутива ядра.
ip_conntrack	Таблица контроля состояния соединений. Фрагмент таблицы показан на рисунке 12.2. Система контроля соединений Linux более подробно рассматривается в параграфе 5.1.6.5.1 (стр. 106)
ip_mr_cache	Список записей маршрутного кэша для групповой адресации.
ip_mr_vif	Список виртуальных multicast-интерфейсов.
ip_tables_matches	Список загруженных модулей соответствия iptables (параграф 5.1.9.3 на стр. 121).
ip_tables_names	Список загруженных модулей таблиц iptables (параграф 5.1.6 на стр. 104).
ip_tables_targets	Список загруженных модулей операций iptables (параграф 5.1.8 на стр. 109).
netlink	Список сокетов **PF_NETLINK** (Приложение 12.10).
netstat	Статистика работы сети, которую можно также просматривать с помощью команды **netstat** (параграф 11.1.2.5 на стр. 199).
packet	Список сокетов **PF_PACKET** (Приложение 12.9)
psched	Список глобальных параметров планировщика пакетов.

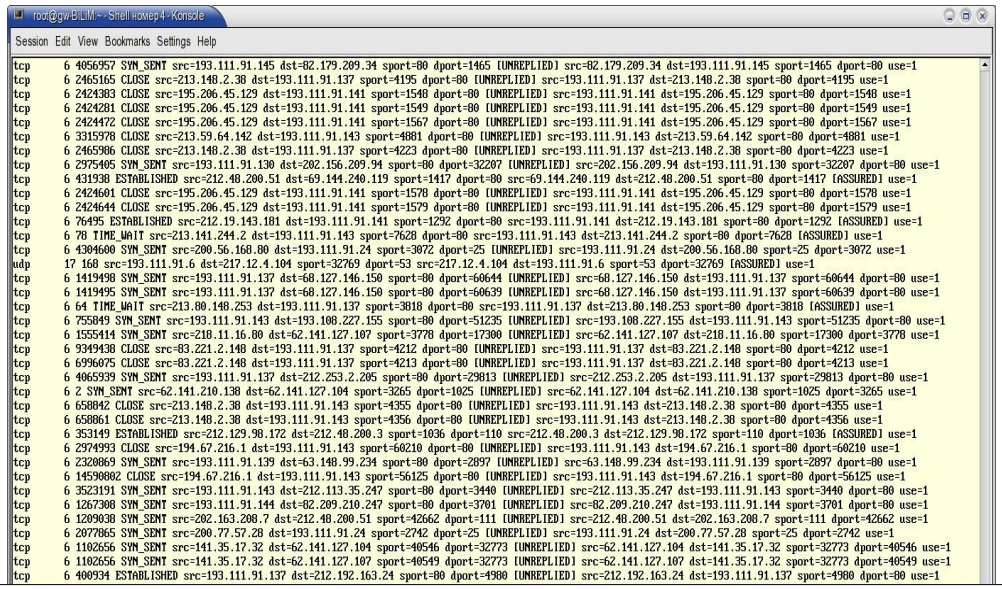
Файл	Описание
rarp	Таблица адресов, используемая для трансляции RARP (определение сетевых адресов по аппаратным). Содержимое таблицы можно просматривать с помощью команды **rarp**. Если при компиляции ядра поддержка опции **IP: RARP support**(параграф 4.4.2.2.5.3.3 на стр. 71) не была активизирована, файла **rarp** не будет в каталоге.
raw	Содержит дамп таблицы беспротокольных сокетов RAW. Большая часть информации из этого файла используется только для отладки. Поле **sl** содержит хэш-слот ядра для сокета, **local_address** содержит пару адрес-порт для локальной стороны, а **remote_address** - для удаленной. Колонка **st** показывает внутреннее состояние сокета. Параметры **tx_queue** и **rx_queue** показывают размер передающей и приемной очереди в единицах использования памяти ядра. Колонки **tr**, **tm->when** и **rexmits** не используются RAW-сокетами и содержат нулевые значения. Поле **uid** содержит эффективный идентификатор пользователя, создавшего сокет.
route	Таблица маршрутизации. Для просмотра маршрутной таблицы можно использовать команду **route** (параграф 11.1.2.6 на стр. 203).
rt_cache	Кэш маршрутов.
snmp	Переменные SNMP для протоколов IP, ICMP, TCP и UDP.
sockstat	Статистика сетевых сокетов в формате: **sockets: used 60** **TCP: inuse 21 orphan 0 tw 0 alloc 21 mem 1** **UDP: inuse 12** **RAW: inuse 2** **FRAG: inuse 0 memory 0**
tcp	Дамп таблицы сокетов TCP. Большая часть содержащейся в таблицы информации используется только для отладки. Поле **sl** содержит хеш-слот ядра для сокета, **local_address** содержит пару адрес-порт для локальной стороны, а **remote_address** - для удаленной (если соединение установлено). Колонка **st** показывает внутреннее состояние сокета. Параметры **tx_queue** и **rx_queue** показывают размер передающей и приемной очереди в единицах использования памяти ядра. Колонки **tr**, **tm->when** и **rexmits** содержат служебную информацию ядра о состоянии сокета, используемую только для отладки. Поле **uid** содержит эффективный идентификатор пользователя, создавшего сокет.
udp	Дамп таблицы сокетов TCP. Большая часть содержащейся в таблицы информации используется только для отладки. Поле **sl** содержит хэш-слот ядра для сокета, **local_address** содержит пару адрес-порт для локальной стороны, а **remote_address** - для удаленной. Колонка **st** показывает внутреннее состояние сокета. Параметры **tx_queue** и **rx_queue** показывают размер передающей и приемной очереди в единицах использования памяти ядра. Поля **tr**, **tm->when** и **rexmits** не используются для сокетов UDP. Поле **uid** содержит эффективный идентификатор пользователя, создавшего сокет. Пример таблицы сокетов UDP показан на рисунке 12.3.
unix	Таблица доменных сокетов UNIX с информацией о состоянии каждого сокета: **Num RefCount Protocol Flags Type St Inode Path** **c1214540: 0000000F 00000000 00000000 0002 01 964 /dev/log** **c23a4b00: 00000002 00000000 00010000 0001 01 1278 /dev/gpmctl** **c13cfa80: 00000002 00000000 00010000 0001 01 1186 /var/run/pdns.controlsocket** **c04b3ae0: 00000002 00000000 00000000 0002 01 474419** **c04b3600: 00000002 00000000 00000000 0002 01 409504** **c0bb35e0: 00000002 00000000 00000000 0002 01 387149** Поле **Num** укаывает номер слота в таблице ядра, **RefCount** указывает число пользователей сокета, поле **Protocol** всегда имеет значение 0, поле **Flags** представляет внутренние флаги ядра, отражающие состояние сокета. Поле **Type** показывает тип сокета, **St** - внутреннее состояние сокета, а **Path** - путь к связанному с сокетом файлу.
wireless	Таблица информации о состояниях беспроводных интерфейсов IEEE 802.11.

```
Inter-|Receive                                                |Transmit
 face |bytes      packets errs drop fifo frame compres mcast|bytes      packets errs drop fifo colls carrier compres
    lo:129202448  356869   0    0    0     0      0      0   129202448  356869   0    0    0     0      0       0
 teql0:0          0        0    0    0     0      0      0   0          0        0    0    0     0      0       0
  eth0:3271724319 22948474 0    0    0     0      0      0   273605437  20823767 8    0    8     0      8       0
  eth1:2123122576 31743724 0    0    0     0      0      0   218986909  13730076 0    0    0     0      0       0
  eth2:562248815  4635052  0    0    0     0      0      0   3453659523 12852457 0    0    0     0      0       0
```

Рисунок 12.1 Формат файла /proc/net/dev

12.2.1.9 Каталог scsi

Этот каталог файлы параметров SCSI среднего уровня и каталоги драйверов низкого уровня для каждого хоста




```
  sl  local_address rem_address   st tx_queue rx_queue tr tm->when retrnsmt   uid  timeout inode
   2: 00000000:0202 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 966 2 c3917580
  16: 00000000:2710 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 295191 2 c19a9580
  33: 00000000:00A1 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 1173 2 c3fa7aa0
  37: 00000000:0025 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 1156 2 c393fac0
  53: 075B6FC1:0035 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 474420 2 c3fa70e0
  81: 075B6FC1:D2D1 015B6FC1:0035 01 00000000:00000000 00:00000000 00000000     0        0 474422 2 c23a4620
 123: 075B6FC1:007B 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 1590 2 c19a90a0
 123: 0100007F:007B 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 1589 2 c1adca40
 123: 00000000:007B 00000000:0000 07 00000000:00000000 00:00000000 00000000     0        0 1588 2 c1adc560
```

Рисунок 12.3 Таблица сокетов UDP в файле /proc/net/udp

SCSI в данной системе. Эти каталоги содержат файлы состояния подсистемы ввода-вывода SCSI. Файлы хранятся в формате ASCII, что обеспечивает возможность доступа к ним с помощью команды **cat**. Некоторые файлы открыты для записи, что позволяет выполнять некоторые операции по настройке без перезагрузки системы или драйвера.

Файл	Описание
`device_info`	Список устройств SCSI.
`scsi`	Список всех устройств SCSI, присутствующих в системе и известных ядру.

12.2.1.10 Каталоги драйверов SCSI

Каталог	Описание
`<имя драйвера>`	Каталоги в настоящее время могут использовать имена **NCR53c7xx**, **aha152x**, **aha1542**, **aha1740**, **aic7xxx**, **buslogic**, **dc395x**, **eata_dma**, **eata_pio**, **fdomain**, **in2000**, **pas16**, **qlogic**, **scsi_debug**, **seagate**, **sg**, **t128**, **u15-24f**, **ultrastore** или **wd7000**. Каждый каталог содержит по одному файлу для каждого зарегистрированного хоста SCSI.

12.2.1.11 Каталог self

Данный каталог является просто символьной ссылкой на каталог процесса, обращающегося к файловой системе **/proc**.

12.2.1.12 Каталог sys

Этот каталог присутствует начиная с ядра версии 1.3.57 и содержит множество файлов и подкаталогов с переменными ядра. Переменные можно читать, а в некоторых случаях допускается изменять их значения путем прямого редактирования файлов **/proc/sys** или команды **sysctl** (параграф 11.2.8 на стр. 217). При изменении переменных следует соблюдать осторожность и менять только те значения, которые вам понятны.

12.2.1.12.1 Каталог sys/abi

Этот необязательный каталог может содержать файлы с информацией о бинарных приложениях.

12.2.1.12.2 Каталог sys/debug

Этот каталог используется для записи переменных при включенном режиме отладки и обычно бывает пустым.

12.2.1.12.3 Каталог sys/dev

Этот каталог содержит связанную с устройствами информацию в подкаталогах, названных по именам устройств. В некоторых случаях каталог может быть пустым.

12.2.1.12.4 Каталог sys/fs

Этот каталог содержит подкаталог **binfmt_misc** и файлы **dentry-state**, **dir-notify-enable**, **dquot-nr**, **file-max**, **file-nr**, **inode-max**, **inode-nr**, **inode-state**, **lease-break-time**, **leases-enable**, **overflowgid**, **overflowuid**, **super-max** и **super-nr**.

12.2.1.12.4.1 Подкаталог binfmt_misc

Ядро Linux поддерживает различные форматы бинарных файлов, что позволяет загружать большинство программ путем простого ввода имени исполняемого файла в строке командного интерпретатора. К числу поддерживаемых относятся программы Java™, Python, Emacs.

Для поддержки различных бинарных форматов нужно указать в **/proc/sys/fs/binfmt_misc** какой командный интерпретатор следует использовать для того или иного бинарного формата. Распознавание форматов осуществляется по сигнатурам в начале бинарных файлов путем их сравнения с известными сигнатурами. Возможно также использовать для распознавания формата расширение имени файла.

Для поддержки различных бинарных форматов нужно сначала смонтировать файловую систему **binfmt_misc** с помощью команды

```
mount binfmt_misc -t binfmt_misc /proc/sys/fs/binfmt_misc
```

Для регистрации нового типа бинарных файлов нужно поместить в файл **/proc/sys/fs/binfmt_misc/register** строки описания формата, имеющие вид

```
:name:type:offset:magic:mask:interpreter:
```

Таблица 115. Поля описания формата бинарных файлов.

Поле	Описание
name	Строка идентификации формата (имя). В каталоге **/proc/sys/fs/binfmt_misc** будет создан файл с соответствующим именем.
type	Способ распознавания формата - **M** для распознавания по сигнатуре, **E** для распознавания по расширению.
offset	Смещение сигнатуры в бинарном файле, заданное в байтах. По умолчанию предполагается нулевое смещение.
magic	Сигнатура, используемая для идентификации формата. Сигнатуры могут содержать буквы, цифры и шестнадцатеричные коды символов (например, **\x0a** или **\xA4**[1]). Если вы задали распознавание по расширению имени, в качестве сигнатуры нужно указать строку расширения без точки перед ней. Строки расширения задаются в символьном виде с учетом регистра.
mask	Необязательная маска, для исключения некоторых битов при сравнении с заданной сигнатурой. При сравнении сигнатуры маска накладывается с помощью логической операции AND (И). По умолчанию значение маски равно **0xff**.
interpreter	Полный путь к программе, которая должна использоваться для загрузки (запуска) бинарного файла.

Для добавления строк описания фв файл можно использовать команду **echo**, как показано в приведенных ниже примерах

12.2.1.12.4.1.1 Ограничения

1) размер строки регистрации формата не должен превышать 255 символов;
2) сигнатура должна располагать в первых 128 байтах файла (т. е., сумма смещения и размера сигнатуры не должна превышать 127);
3) размер поля **interpreter** не должен превышать 127 символов.

12.2.1.12.4.1.2 Примеры использования

Для использования различных бинарных форматов требуется сначала смонтировать файловую систему **binfmt_misc**. Вы можете сделать это вручную с помощью команды

```
mount -t binfmt_misc none /proc/sys/fs/binfmt_misc
```

или добавить строку

```
none  /proc/sys/fs/binfmt_misc binfmt_misc defaults 0 0
```

в файл **/etc/fstab** для автоматического монтирования при загрузке системы.

Чтобы включить поддержку приложений **em86** можно использовать команду:

1 *При задании сигнатуры в среде командного интерпретатора может потребоваться включение дополнительного символа \ (**\\x0a**).*

```
echo ':i386:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\x03:\xff\
xff\xff\xff\xff\xfe\xfe\xff\xff\xff\xff\xff\xff\xff\xff\xff\xfb\xff\xff:/bin/em86:' >
/proc/sys/fs/binfmt_misc/register
```
или
```
echo ':i486:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\x06:\xff\
xff\xff\xff\xff\xfe\xfe\xff\xff\xff\xff\xff\xff\xff\xff\xff\xfb\xff\xff:/bin/em86:' >
/proc/sys/fs/binfmt_misc/register
```
Для включения поддержки упакованных приложений DOS подойдет команда:
```
echo ':DEXE:M::\x0eDEX::/usr/bin/dosexec:' > /proc/sys/fs/binfmt_misc/register
```
Для использования исполняемых файлов Windows с помощью прогграммы **wine** можно ввести команду:
```
echo ':DOSWin:M::MZ::/usr/local/bin/wine:' > /proc/sys/fs/binfmt_misc/register
```
Для включения поддержки приложений Java можно использовать команды:
```
echo ':Java:M::\xca\xfe\xba\xbe::/usr/local/bin/javawrapper:' > /proc/sys/fs/binfmt_misc/register
```
для Jar-файлов:
```
echo ':ExecutableJAR:E::jar::/usr/local/bin/jarwrapper:' > /proc/sys/fs/binfmt_misc/register
```
для Java-апплетов:
```
echo ':Applet:E::html::/usr/bin/appletviewer:' > /proc/sys/fs/binfmt_misc/register
```
или:
```
echo ':Applet:M::<!--applet::/usr/bin/appletviewer:' > /proc/sys/fs/binfmt_misc/register
```
Для того, чтобы включить или отключить поддержку **binfmt_misc** или отдельного типа бинарных файлов можно воспользоваться командой **echo**, передающей 0 (выключить) или 1 (включить) в общий файл **/proc/sys/fs/binfmt_misc/status** или файлы для отдельных форматов (**/proc/.../<имя>**). Для просмотра текущего состояния можно воспользоваться командой **cat**.

Для удаления записей можно также воспользоваться командой **echo**
```
echo -1 > /proc/sys/fs/binfmt_misc/status
```
будет удалять все записи, а
```
echo -1 > /proc/sys/fs/binfmt_misc/<имя>
```
запись для указанного формата.

12.2.1.12.4.2 Файлы /proc/sys/fs

Таблица 116. Файлы каталога /proc/sys/fs.

Файл	Описание
dentry-state	Шесть чисел: **nr_dentry** - номер записи dentry; **nr_unused** - число неиспользованных записей dentry; **age_limit** - количество секунд, по истечении которых запись может быть удалена при нехватке памяти; **want_pages** - число страниц, запрашиваемых системой 2 пустых значения.
dir-notify-enable	Включает (1) или отключает (0) интерфейс уведомлений **dnotify** для функции **fcntl** в масштабе всей системы.
dquot-max	Максимальное число кэшированных записей для дисковых квот.
dquot-nr	Количество выделенных и свободных дисковых квот.
file-max	Верхнее ограничение числа открытых файлов в масштабе системы.
file-nr	Доступный только для чтения файл, показывающий общее число открытых в системе файлов. Первое значение показывает число выделенных файловых дескрипторов, второе - число свободных дескрипторов, а третье число указывает максимальное число файловых дескрипторов для данной системы (**file-max**).
inode-max	Максимальное число хранящихся в памяти индексных дескрипторов inode. Это число должно в 3-4 раза превышать максимальное значение числа файловых дескрипторов, заданное в **file-max**.
inode-nr	Два первых значения из файла **inode-state**.
inode-state	Содержит 7 числовых параметров: **nr_inodes** - число выделенных системой индексных дескрипторов[1]; **nr_free_inodes** - число свободных индексных дескрипторов; **preshrink** - отличное от нуля значение, если **nr_inodes** > **inode-max** и системе нужно уничтожить группу индексных дескрипторов вместо выделения новых; 4 пустых значения.

1 *Это значение может незначительно превышать значение **inode-max**, поскольку в Linux используется постраничное распределение.*

Файл	Описание
lease-break-time	Задает продолжительность периода, в течение которого процесс может удерживать взятый файл после того, как ядром был передан сигнал, уведомляющий о запросе другого процесса на открытие этого файла. Если держатель файла не освободит его или не снизит уровень владения в течение заданного времени, ядро прервет принадлежность файла данному процессу.
leases-enable	Включает или отключает поддержку функций “аренды” (lease) файлов в масштабе системы. Нулевое значение запрещает аренду, 1 - разрешает.
overflowgid	Максимальное значение идентификатора владельца UID при записи файлов.
overflowuid	Максимальное значение идентификатора группы GID при записи файлов.
super-max	Определяет максимальное число суперблоков (superblock) и, следовательно, ограничивает число монтируемых файловых систем.
super-nr	Число смонитрованных файловых систем.

12.2.1.12.5 Каталог sys/kernel

Таблица 117. Файлы каталога /proc/sys/kernel.

Файл	Описание
acct	Файл содержит 3 значения **highwater**, **lowwater** и **frequency**. Если при компиляции ядра была включена опция **BSD process accounting** (параграф 4.4.1.2.4 на стр. 64), эти значения управляют запись учетной информации в журнальный файл системы. Если процент свободного пространства на диске для записи журнальных файлов становится ниже значения **lowwater**, запись учетной информации прекращается и будет восстановлена лишь после того, как процент свободного пространства превысит значение **highwater**. Параметр **frequency** определяет период проверки свободного пространства (в секундах). По умолчанию используются значения 4, 2 и 30. Запись учетной информации прекращается при снижении объема свободного пространства до 2% и восстанавливается при наличии 4% свободного пространства. Проверка осуществляется каждые 30 секунд.
cap-bound	Используемая ядром маска ограничения возможностей приложений. Заданное в файле десятичное значение маски накладывается на значение поля возможностей приложения с помощью операции **AND**.
core_pattern	Имя, используемое для записи дампа памяти при аварийном завершении приложений.
core_uses_pid	Этот файл управляет именами файлов с дампом памяти, записываемыми при аварийном закрытии приложений. Если файл содержит ненулевое значение, дамп памяти будет записываться в файл с именем **core.PID**, в противном случае, имя файла не будет содержать идентификатора аварийного процесса..
ctrl-alt-del	Определяет режим обработки клавиатурных прерываний **Ctrl**-**Alt**-**Del**. При нулевом значении информация о нажатии клавиш **Ctrl**-**Alt**-**Del** перехватывается и передается функции **init** для корректного выполнения операции перезагрузки. Отличное от нуля значение разрешает незамедлительную перезагрузку системы без синхронизации файловой системы, что может вызвать проблемы при следующей загрузке. Отметим, что некоторые программы (типа **dosemu**), использующие клавиатуру в raw-режиме, могут самостоятельно обрабатывать прерывания **Ctrl**-**Alt**-**Del** еще до того, как информация о нажатии клавиш попадет в ядро.
domainname	Доменное имя NIS/YP. Отметим, что это имя отличается от доменного имени DNS.
hostname	Имя хоста.
hotplug	Указывает местоположение программы **hotplug**.
modprobe	Указывает местоположение программы **modprobe**, используемой для проверки и загрузки модулей ядра.
msgmax	Максимальный размер сообщения, записываемого в очередь **System V**. По умолчанию 8192 байта.
msgmnb	Максимальный размер очереди сообщений **System V**. По умолчанию 16384 байта.
msgmni	Максимальное значение идентификаторов сообщений в очереди **System V**. По умолчанию 16.
osrelease	Номер версии используемого системой ядра.
ostype	
overflowgid	Дубликат файла **/proc/sys/fs/overflowgid**, определяющего максимальное значение идентификатора группы (GID).
overflowuid	Дубликат файла **/proc/sys/fs/overflowuid**, определяющего максимальное значение идентификатора пользователя (UID).

Файл	Описание
panic	Определяет поведение системы при невозможности загрузки ядра (kernel panic). При нулевом значении система будет ожидать реакции пользователя, а отличные от нуля значения задают время (в секундах), по истечении которого будет выполнена автоматическая перезагрузка.
printk	Этот файл содержит 4 целых числа определяющих вывод и протоколирование сообщений об ошибках: **console_loglevel** - уровень сообщений, с которого начинается вывод на консоль; **default_message_loglevel** - уровень приоритета, используемый по умолчанию для сообщений, уровень которых не задан; **minimum_console_level** - минимальное значение, которое может быть установлено в поле **console_loglevel**; **default_console_loglevel** - используемый по умолчанию уровень **console_loglevel**. Информацию о записи сообщений в журнальные файлы вы сможете найти в параграфе 2.8.4 на стр. 49).
random	Подкатлог, содержащий файлы, управляющие работой системного генератора случайных чисел **/dev/random**.
rtsig-max	Максимальное число сигналов POSIX, которые могут находиться в очереди.
rtsig-nr	Текущее число сигналов POSIX в очереди.
sem	Параметры управления семафорами **System V IPC**: **SEMMSL** - максимальное число семафоров в одном наборе; **SEMMNS** - максимальное число операций, которые могут быть заданы при вызове функции **semop**; **SEMMNI** - максимальное число идентификаторов семафоров.
shmall	Максимальное число страниц разделяемой памяти **System V**.
shmmax	Максимальный размер сегмента разделяемой памяти **System V IPC**.
shmmni	Максимальное число создаваемых сегментов разделяемой памяти **System V IPC**.
version	Номер и время создания ядра.

12.2.1.12.6 Каталог sys/net

Переменные каталога **/proc/sys/net/*** описаны в Приложении 12.3 (стр. 366).

12.2.1.12.7 Каталог sys/proc

Этот каталог может быть пустым.

12.2.1.12.8 Каталог sys/sunrpc

Параметры Sun RPC для использования NFS.

12.2.1.12.9 Каталог sys/vm

Параметры управления распределением памяти, буферами и кэшированием.

12.2.1.13 Каталог sysvipc

Этот каталог содержит файлы **msg**, **sem** и **shm**, описывающие связи между процессами System V IPC[1]. Файлы включают заголовки, облегчающие понимание приведенных значений.

12.2.1.14 Каталог tty

Этот каталог содержит файлы и каталоги, описывающие параметры присутствующих в системе терминалов.

12.3 Параметры SysCtl для стека IP[2]

В этом приложении описаны параметры протокола IP, значения которых содержатся в файлах виртуальной файловой системы **/proc**. Значения параметров задаются во время загрузки операционной системы и могут быть изменены в любой момент с помощью команд типа

```
echo "значение" > /proc/sys/net/ipv4/имя_файла
```

Отметим, что измененые в процессе работы значения не будут сохранены после перезагрузки системы. Если вы хотите сохранить изменения, включите соответствующую команду в один из сценариев загрузки системы[3].

При установке значений тех или иных параметров следует руководствоваться требованиями к хостам и маршрутизаторам, изложенными в документах RFC 1122, RFC 1123 и RFC 1812.

1 *Interprocess Communication - обмен информацией между процессами.*
2 *Приложение основано на переводе файла Documentation/networking/ip-sysctl.txt из состава ядра Linux 2.6.8*
3 *Разумно использовать для этих целей файл /etc/rc.d/rc.local.*

12.3.1 Параметры IPv4

12.3.1.1 ip_forward - пересылка пакетов

Логическая переменная **ip_forward** управляет возможностью пересылки пакетов между сетевыми интерфейсами (маршрутизации). Значение 1 разрешает маршрутизацию, 0 - запрещает. По умолчанию маршрутизация пакетов обычно отключена.

Изменение значения этой переменной приводит к сбросу всех конфигурационных параметров сети в принятые по умолчанию значения (в соответствии с RFC 1122 для хостов и RFC 1812 для маршрутизаторов).

12.3.1.2 ip_default_ttl - время жизни пакетов

Целочисленная переменная (1 байт) **ip_default_ttl** определяет время жизни пакетов, передаваемых данным хостом. Значение поля времени жизни уменьшается на 1 каждым маршрутизатором на пути следования пакетов к адресату. По умолчанию устанавливается значение TTL=64.

12.3.1.3 ip_no_pmtu_disc

Логическая переменная **ip_no_pmtu_disc** управляет использованием функции Path MTU Discovery (определение максимального размера пакетов для пути). По умолчанию эта функция отключена (0), для ее включения используйте команду

```
echo "1" > /proc/sys/net/ipv4/ip_no_pmtu_disc
```

12.3.1.4 ip_queue_maxlen - максимальное число пакетов в очереди пользовательского пространства

Целочисленное значение в файле **/proc/sys/net/ipv4/ip_queue_maxlen** определяет максимальное число пакетов, которые могут быть переданы в очередь пользовательского пространства без получения вердикта об их дальнейшей судьбе. При достижении порога все последующие пакеты будут отбрасываться, пока размер очереди не снизится.

По умолчанию для этого параметра используется значение 1024. Если это значение слишком мало для вашей системы, подберите большее значение экспериментальным путем.

12.3.1.5 Параметры фрагментации пакетов IP

12.3.1.5.1 ipfrag_high_thresh - максимальный размер памяти для сборки фрагментов

Целочисленная переменная **ipfrag_high_thresh** задает максимальный размер памяти 9в байтах), которая может использоваться для сборки фрагментов IP. После выделения заданного количества байтов памяти обработчик фрагментов будет выталкивать (toss) пакеты из буфера, пока не будет достигнуто значение нижнего порога ipfrag_low_thresh.

12.3.1.5.2 ipfrag_low_thresh - нижний предел размера для буфера сборки пакетов

Целочисленное значение **ipfrag_low_thresh** определяет нижний порог размера буфера сборки фрагментов IP.

12.3.1.5.3 ipfrag_time - время хранения фрагментов

Целочисленная переменная **ipfrag_time** определяет время (в секундах), в течение которого фрагмент дейтаграммы IP может храниться в памяти.

12.3.1.5.4 ipfrag_secret_interval - время жизни хэш-ключа

Целочисленное значение переменной **ipfrag_secret_interval** задает интервал регенерации (в секундах) секретного хэш-ключа, используемого для фрагментов IP. По умолчанию время жизни ключа составляет 10 минут (600 секунд).

12.3.1.6 Переменные INET peer storage

12.3.1.6.1 inet_peer_threshold

Целочисленная переменная **inet_peer_threshold** задает приблизительный размер буфера. По достижении этого порога начинается жесткое отбрасывание записей. Данный порог влияет также на время жизни записей и интервалы между проходами для очистки памяти (сборки мусора - garbage collection). Чем больше записей, тем меньшее время они живут и чаще происходит сборка мусора.

12.3.1.6.2 inet_peer_minttl

Целочисленное значение **inet_peer_minttl** определяет минимальное время жизни записей. Этого времени должгл быть достаточно для покрытия времени жизни фрагмента на стороне сборки. Минимальное время жизни определяет гарантированный период существования для тех случаев, когда размер пула меньше значения

переменной inet_peer_threshold. Время жизни задается в специальных единицах jiffy[1].

12.3.1.6.3 inet_peer_maxttl

Целочисленная переменная **inet_peer_maxttl** определяет максимальное время жизни записей. Неиспользованные записи по истечении этого периода могут сохраняться при отсутствии дефицита памяти (когда число записей в пуле достаточно мало). Время жизни задается в jiffy.

12.3.1.6.4 inet_peer_gc_mintime

Целочисленная переменная **inet_peer_gc_mintime** задает минимальный интервал между проходами для очистки (сборки мусора - garbage). Такой короткий интервал используется при нехватке памяти в пуле. Интервал задается в единицах jiffy.

12.3.1.6.5 inet_peer_gc_maxtime

Целочисленная переменная **inet_peer_gc_maxtime** определяет максимальный интервал между проходами для очистки памяти. Максимальный переиод между очистками памяти используется при отсутствии дефицита памяти в пуле. Интервал задается в единицах jiffy.

12.3.1.7 Переменные TCP

12.3.1.7.1 tcp_syn_retries

Целочисленная переменная (1 байт) **tcp_syn_retries** определяет число попыток передачи пакетов SYN, которые протокол TCP будет предпринимать для организации соединения. Число попыток не должно превышать 255. Используемое по умолчанию значение 5 соответствует приблизительно 180 секундам на выполнение попыток организации соединения.

12.3.1.7.2 tcp_synack_retries

Целочисленное значение (1 байт) **tcp_synack_retries** определяет число попыток повтора передачи пакетов SYNACK для пассивных соединений TCP. Число попыток не должно превышать 255. Используемое по умолчанию значение 5 соответствует приблизительно 180 секундам на выполнение попыток организации соединения.

12.3.1.7.3 tcp_keepalive_time

Целочисленное значение **tcp_keepalive_time** определяет интервал (в секундах) передачи протоколом TCP сообщений keepalive. По умолчанию интервал передачи таких сообщений составляет 7200 (2 часа).

12.3.1.7.4 tcp_keepalive_probes

Целочисленная переменная **tcp_keepalive_probes** задает число передач проб **keepalive**, после которого соединение считается разорванным. По умолчанию передается 9 проб.

12.3.1.7.5 tcp_keepalive_intvl

Целочисленная переменная **tcp_keepalive_intvl** определяет интервал передачи проб. Произведение **tcp_keepalive_probes** * **tcp_keepalive_intvl** определяет время, по истечении которого соединение будет разорвано при отсутствии откликов. По умолчанию установлен интервал 75 секунд, т.е., время разрыва соединения при отсутствии откликов составит приблизительно 11 минут.

12.3.1.7.6 tcp_retries1

Целочисленная переменная **tcp_retries1** определяет число неудачных попыток, после которого должна быть передана информация на сетевой уровень. В соответствии с RFC минимальное значение составляет 3 (по умолчанию установлено именно это значение), что соответствует периоду приблизительно от 3 секунд до 8 минут в зависимости от значения тайм-аута повторной передачи RTO (Retransmission time-out).

12.3.1.7.7 tcp_retries2

Целочисленная переменная **tcp_retries2** определяет число неудачных попыток, после которого существующее соединение уничтожается. В соответствии с RFC 1122 тайм-аут должен быть больше 100 секунд. Такое значение слишком мало и по умолчания установлено число попыток 15, соответствующее тайм-ауту приблизительно от 13 до 30 минут в зависимости от RTO.

12.3.1.7.8 tcp_orphan_retries

Целочисленной значение **tcp_orphan_retries** определяет число неудачных попыток, после которого уничтожается соединение TCP, закрытое на локальной стороне. По умолчанию используется значение 7, соответствующее приблизительно периоду от 50 секунд до 16минут в зависимости от RTO. На сильно загруженных WEB-серверах имеет смысл уменьшить значение этого параметра, поскольку закрытые соединения могут поглощать достаточно

1 *Специальная единица времени, используемая ядрами Linux. Для процессоров x86 составляет приблизительно 10 мсек.*

много ресурсов (см. стр. 369).

12.3.1.7.9 tcp_fin_timeout

Целое число в файле **tcp_fin_timeout** определяет время сохранения сокета в состоянии **FIN-WAIT-2** после его закрытия локальной стороной. Партнер может не закрыть это соединение никогда, поэтому следует закрыть его по своей инициативе по истечении тайм-аута. По умолчанию тайм-аут составляет 60 секунд. В ядрах серии 2.2 обычно использовалось значение 180 секунд и вы можете сохранить это значение, но не следует забывать, что на загруженных WEB-серверах вы рискуете израсходовать много памяти на сохранение полуразорванных мертвых соединений (см. стр. 369). Сокеты в состоянии **FIN-WAIT-2** менее опасны, нежели **FIN-WAIT-1**, поскольку поглощают не более 1,5 Кбайт памяти, но они могут существовать дольше.

12.3.1.7.10 tcp_max_tw_buckets

Целочисленное значение в файле **tcp_max_tw_buckets** задает максимальное число сокетов, находящихся в состоянии ожидания (timewait socket), которые могут существовать в системе одновременно. При достижении этого значения лишние сокеты незамедлительно уничтожаются и выводится предупреждение. Этот порог служит только для предотвращения простых DoS-атак и вам не следует значение порога (скорее следует увеличить пороговое значение, если условия вашей сети требуют этого).

12.3.1.7.11 tcp_tw_recycle

Логическая переменная **tcp_tw_recycle** управляет рециркуляцией сокетов TIME-WAIT. По умолчанию используется значение 0 (отключено), которое не следует менять без консультации с техническими специалистами.

12.3.1.7.12 tcp_tw_reuse

Логическая переменная **tcp_tw_reuse** определяет возможность повторного использования сокетов TIME-WAIT для новых соединений, когда это безопасно с точки зрения протокола. По умолчанию такая возможность отключена (0) и включать ее не следует без консультации с техническими специалистами.

12.3.1.7.13 tcp_max_orphans

Целочисленное значение в файле **tcp_max_orphans** определяет максимальное число допустимых в системе сокетов TCP, не связанных каким-либо идентификатором пользовательского файла (user file handle). При достижении порогового значения “осиротевшие” (orphan[1]) соединения незамедлительно сбрасываются с выдачей предупреждения. Этот порог помогает предотвращать только простые атаки DoS и вам не следует уменьшать пороговое значение (скорее увеличить его в соответствии с требованиями системы - например, после добавления памяти. Отметим, что каждое orphan-соединение поглощает около 64 Кбайт несбрасываемой на диск (unswappable) памяти.

12.3.1.7.14 tcp_abort_on_overflow

Логическая переменная **tcp_abort_on_overflow** управляет возможность отказа от приема новых соединений при недостаточной производительности соответствующей службы. По умолчанию установлено значение FALSE (0), означающее, что при переполнении в результате пиковой нагрузки соединение будет восстановлено. Используйте для этой переменной значение TRUE (1) только в тех случаях, когда вы реально уверены, что прослушивающий соединения демон не способен работать быстрее. В этом случае часть клиентов будет получать отказы при попытке соединения с сервером.

12.3.1.7.15 tcp_syncookies

Логическая переменная в файле **tcp_syncookies** (требуется ядро со включенной опцией **CONFIG_SYNCOOKIES** - см. стр. 72) управляет записью предупреждений synflood в журнальные файлы при переполнении сокета. Эта функция может использоваться для защиты от атак **SYN flood**. По умолчанию используется значение FALSE (0).

Отметим, что функции syncookies относятся к типу fallback и их не следует использовать для попыток ограничить число легитимных соединений с сервером. Если вы видите предупреждения synflood в журнальных файлах и их исследование показывает, что причиной является значительное число легитимных соединений с сервером, вам следует настроить другие параметры (**tcp_max_syn_backlog** - параграф 12.3.1.7.17, **tcp_synack_retries** - параграф 12.3.1.7.2, **tcp_abort_on_overflow** - параграф 12.3.1.7.14), чтобы избавиться от таких сообщений.

Использование syncookie оказывает серьезное влияние на работу протокола TCP[2], не позволяя использовать расширения TCP, что может привести к существенному ухудшению работы отдельных служб (например, транслятора SMTP), незаметному на локальной машине, но отражающемуся на доступе клиентов и других серверов к службам данного сервера. Наличие предупреждений synflood в журнальных файлах при отсутствии реальных атак говорит о серьезных ошибках в конфигурации вашего сервера.

Более подробную информацию и иной взгляд на проблему вы сможете найти в Приложении 12.15 на странице 400.

12.3.1.7.16 tcp_stdurg

Логическая переменная **tcp_stdurg** управляет интерпретацией поля URG в заголовках TCP. Большинство хостов

1 *Разорванные в одностороннем порядке.*
2 *В статье D. J. Bernstein, перевод которой содержится в Приложении 12.15 (стр. 400), приводится иной взгляд на эту проблему.*

использует старую интерпретацию BSD, установка значения TRUE (1) приведет к интерпретации этого поля в соответствии с документом Host requirements. Использование значения TRUE может вызывать проблемы при взаимодействии с хостами, поддерживающими старый вариант интерпретации. По умолчанию используется значение FALSE

12.3.1.7.17 tcp_max_syn_backlog

Целочисленное значение в файле **tcp_max_syn_backlog** определяет максимальное число запоминаемых запросов на соединение, для которых не было получено подтверждения от подключающегося клиента. По умолчанию используется значение 1024 для систем, в которых объем ОЗУ превышает 128 Мбайт и 128 для систем с меньшим объемом памяти. Если на сервере возникают перегрузки, попытайтесь увеличить это значение.

12.3.1.7.18 tcp_window_scaling

Логическая переменная **tcp_window_scaling** управляет возможностью масштабирования размера окна в соответствии с RFC 1323.

12.3.1.7.19 tcp_timestamps

Логическая переменная в файле **tcp_timestamps** определяет интерпретацию временных меток в соответствии с документом RFC 1323.

12.3.1.7.20 tcp_sack

Логическая переменная **tcp_sack** включает режим SACKS (select acknowledgments).

12.3.1.7.21 tcp_fack

Логическая переменная **tcp_fack** управляет режимом предотвращения насыщения FACK и быстрым повтором передач. При установке **tcp_sack=0** (см. стр. 370) эта переменная не используется.

12.3.1.7.22 tcp_dsack

Логическая переменная **tcp_dsack** позволяет протоколу TCP передавать “дубликаты” пакетов SACK.

12.3.1.7.23 tcp_ecn

Логическая переменная **tcp_ecn** управляет механизмом ECN[1] для протокола TCP.

12.3.1.7.24 tcp_reordering

Целочисленная переменная **tcp_reordering** определяет максимальное разупорядочивание пакетов в потоке TCP. По умолчанию используется значение 3.

12.3.1.7.25 tcp_retrans_collapse

Логическая переменная **tcp_retrans_collapse** управляет режимом совместимости (Bug-to-bug) для некоторых устаревших принтеров. При повторной передаче предпринимается попытка отправить пакет большего размера, чтобы преодолеть проблемы, связанные с ошибками в некоторых реализациях стека TCP.

12.3.1.7.26 tcp_wmem

Векторная переменная в файле **tcp_wmem** содержит 3 целочисленных значения, определяющих минимальное, принятое по умолчанию и максимальное количество памяти, резервируемой для буферов передачи сокета TCP.

Минимум: каждый сокет TCP имеет право использовать эту память по факту своего создания. Размер минимального буфера по умолчанию составляет 4 Кбайт (4096)

Значение по умолчанию: количество памяти, допустимое для буфера передачи сокета TCP по умолчанию. Это значение применяется взамен параметра **/proc/sys/net/core/wmem_default** (см. стр. 379), используемого другими протоколами и обычно меньше, чем **/proc/sys/net/core/wmem_default**. Размер принятого по умолчанию буфера обычно (по умолчанию) составляет 16 Кбайт (16384)

Максимум: максимальное количество памяти, которое может быть автоматически выделено для буфера передачи сокета TCP. Это значение не отменяет максимум, заданный в файле **/proc/sys/net/core/wmem_max** (см. стр. 379). При “статическом” выделении памяти с помощью **SO_SNDBUF** этот параметр не имеет значения. По умолчанию размер максимального буфера составляет 128 Кбайт (131072).

12.3.1.7.27 tcp_rmem

Векторная (минимум, по умолчанию, максимум) переменная в файле **tcp_rmem** содержит 3 целых числа, определяющих размер приемного буфера сокетов TCP.

Минимум: каждый сокет TCP имеет право использовать эту память по факту своего создания. Возможность использования такого буфера гарантируется даже при достижении порога ограничения (moderate memory pressure,

1 *Explicit Congestion Notification -явное уведомление о насыщении.*

см. параграф 12.3.1.7.28 на стр. 371). Размер минимального буфера по умолчанию составляет 8 Кбайт (8192).

Значение по умолчанию: количество памяти, допустимое для буфера передачи сокета TCP по умолчанию. Это значение применяется взамен параметра **/proc/sys/net/core/rmem_default** (стр. 378), используемого другими протоколами. Размер принятого по умолчанию буфера обычно составляет 87830 байт. Это определяет размер окна 65535 с заданным по умолчанию значением tcp_adv_win_scale (параграф 12.3.1.7.30) и **tcp_app_win** = 0 (параграф 12.3.1.7.29), несколько меньший, нежели определяет принятое по умолчанию значение tcp_app_win.

Максимум: максимальный размер буфера, который может быть автоматически выделен для приема сокету TCP. Это значение не отменяет максимума, заданного в файле **/proc/sys/net/core/rmem_max** (см. стр. 378). При "статическом" выделении памяти с помощью SO_RCVBUF этот параметр не имеет значения. Используемый по умолчанию максимум составляет 87380 * 2 байт.

12.3.1.7.28 tcp_mem

Векторная переменная в файле **tcp_mem** содержит 3 целых числа (порога), определяющих отношение протокола TCP к выделению памяти.

Нижний порог: при значениях ниже этого уровня TCP не заботится о расходе памяти.

Порог ограничения: при достижении этого порога TCP контролирует размер выделяемой памяти и переходит в режим **memory pressure** (нехватка памяти), из которого выходит при снижении расхода памяти до нижнего порога.

Верхний порог: число страниц памяти, доступных для создания очередей всеми сокетами TCP.

Значения всех порогов рассчитываются во время загрузки ОС с учетом доступной на компьютере памяти.

12.3.1.7.29 tcp_app_win

Целочисленное значение **tcp_app_win** определяет размером окна, резервируемого для буфера приложений - max(window/2^tcp_app_win, mss). Нулевое значение в этом файле означает отсутствие резервирования.

По умолчанию используется значение 31.

12.3.1.7.30 tcp_adv_win_scale

Целочисленный параметр в файле **tcp_adv_win_scale** определяет степень перекрытия буферов - bytes/2^tcp_adv_win_scale при положительных значениях параметра и bytes - bytes/2^(-tcp_adv_win_scale) при tcp_adv_win_scale <= 0 (bytes - размер буфера в байтах). По умолчанию используется значение 2.

12.3.1.7.31 tcp_rfc1337

Логическая переменная **tcp_rfc1337** определяет соответствие стека TCP требованиям RFC 1337. Значение FALSE (0), используемое по умолчанию, говорит о несоответствии RFC 1337 и сохранении действия параметра TCP TIME_WAIT.

12.3.1.7.32 tcp_low_latency

Логическая переменная в файле **tcp_low_latency** определяет приоритет "задержка - пропускная способность". При установленном значении (1) стек TCP предпочитает меньшую задержку большему пропусканию. По умолчанию эта опция не установлена (0) и предпочтение отдается пропускной способности. Примером приложения, где не следует использовать принятое по умолчанию значения, является кластер Beowulf.

12.3.1.7.33 tcp_westwood

Логическая переменная **tcp_westwood** управляет поддержкой алгоритма контроля насыщения TCP Westwood+, использующего на стороне отправителя модифицированный стек TCP Reno, в котором оптимизированы функции контроля насыщения. Этот алгоритм основан на оценке сквозной полосы для выбора окна насыщения и порога медленного старта после случая насыщения. Используя такую оценку, алгоритм TCP Westwood+ адаптивно устанавливает порог медленного старта и размер окна насыщения с учетом ожидаемой полосы в период насыщения.

Алгоритм TCP Westwood+ является существенно более беспристрастным по сравнению TCP Reno в проводных и беспроводных сетях.

По умолчанию установлено значение 0 (алгоритм не используется).

12.3.1.7.34 tcp_vegas_cong_avoid

Логическая переменная **tcp_vegas_cong_avoid** управляет использованием алгоритма блокировки насыщения TCP Vegas.

TCP Vegas использует на серверной стороне модифицированный стек TCP, который позволяет предупредить насыщение за счет оценки полосы. TCP Vegas регулирует исходящий поток трафика путем изменения окна насыщения. Стек TCP Vegas должен обеспечивать меньший уровень потери пакетов, но этот алгоритм менее агрессивен, нежели TCP Reno.

По умолчанию алгоритм TCP Vegas отключен (0).

12.3.1.7.35 tcp_bic

Логическая переменная **tcp_bic** управляет механизмом контроля насыщения BIC-TCP, использующим на серверной стороне модифицированный стек TCP, который обеспечивает линейное изменение RTT[1] при больших окнах в сочетании с масштабируемостью и дружественностью TCP. Протокол объединяет в себе две схемы - additive increase и binary search increase. При большом окне насыщения аддитивный рост с большим инкрементом обеспечивает линейное изменение RTT в сочетании с хорошей масштабируемостью При небольшом окне насыщения, метод binary search increase обеспечивает дружественность TCP. Описание алгоритма можно загрузить с сайта http://www4.ncsu.edu:8030/~lxu2/xu_INFOCOM_2004.pdf.

По умолчанию алгоритм отключен (0).

12.3.1.7.36 tcp_bic_low_window

Целочисленная переменная в файле **tcp_bic_low_window** задает (в пакетах) размер порогового окна, при котором алгоритм BIC-TCP начинает подстраивать размер окна насыщения. Ниже этого порога алгоритм BIC-TCP ведет себя подобно TCP Reno. По умолчанию используется значение 14.

12.3.1.7.37 tcp_bic_fast_convergence

Логическая переменная **tcp_bic_fast_convergence** заставляет алгоритм BIC-TCP быстрее реагировать на изменения окна насыщения. Это позволяет двум потокам в одном канале сходиться быстрее. По умолчанию опция включена (1).

12.3.1.8 ip_local_port_range

Файл **ip_local_port_range** содержит 2 целых числа, определяющих диапазон значений, используемых протоколами TCP и UDP для выбора локальных номеров портов. Первое число задает минимальный номер локального порта, второе - максимальный. Установленные по умолчанию значения зависят от размера оперативной памяти на компьютере - при объеме памяти 128 Мбайт используется диапазон 32768 - 61000, при меньшем объеме памяти - 1024 - 4999 или еще меньше.

Заданные в этом файле значения определяют число активных соединений, которая система может инициировать (вводить) одновременно, если в данной системе не поддерживается расширение TCP extensions (timestamp - временные метки). При включенной опции tcp_tw_recycle (принято по умолчанию) диапазона 1024 - 4999 достаточно для возможности организации до 2000 соединений в секунду, если система поддерживает временные метки.

12.3.1.9 ip_nonlocal_bind

Логическая переменная **ip_nonlocal_bind** позволяет связывать (bind) процессы с нелокальными адресами IP. Такое связывание может быть весьма полезно, но мешает работе некоторых приложений. По умолчанию связывание отключено (0).

12.3.1.10 ip_dynaddr

Логическая переменная **ip_dynaddr** определяет возможность использования динамических адресов IP. При установке для этой переменной значения >1, факты смены динамических адресов будут протоколироваться в журнальных файлах системы. По умолчанию поддержка динамических адресов отключена (0).

12.3.1.11 Переменные ICMP

12.3.1.11.1 icmp_echo_ignore_all

Логическая переменная **icmp_echo_ignore_all** позволяет включить блокировку всех пакетов ICMP Echo. При значении TRUE (1) ядро будет игнорировать все запросы ICMP Echo.

12.3.1.11.2 icmp_echo_ignore_broadcasts

Логическая переменная **icmp_echo_ignore_broadcasts** позволяет включить блокировку широковещательных запросов ICMP Echo. При значении TRUE (1) ядро будет игнорировать все широковещательные запросы ICMP Echo.

12.3.1.11.3 icmp_ratelimit

Целочисленная переменная **icmp_ratelimit** задает максимальную скорость передачи пакетов ICMP, тип которых соответствует маске icmp_ratemask, описанной в следующем параграфе. Значение 0 отменяет все ограничения, остальные значения задают максимальное количество пакетов в течение периода jiffy[2].

По умолчанию используется значение 100

12.3.1.11.4 icmp_ratemask

Целочисленное значение переменной **icmp_ratemask** определяет маску типов пакетов ICMP, для которых

1 *Round trip time - время кругового обхода.*
2 *Единица времени, используемая ядром Linux. В системах x86 составляет приблизительно 10 мсек.*

действуют ограничения, заданные переменной icmp_ratelimit (см. параграф 12.3.1.11.3).

```
Значимые биты:          IHGFEDCBA9876543210
Маска по умолчанию:     0000001100000011000
```

используемое по умолчанию значение в десятичном формате составляет 6168.

Идентификаторы значимых битов:

0 Echo Reply (отклик)

3 Destination Unreachable (адресат недоступен) *

4 Source Quench (умерьте свой пыл) *

5 Redirect (перенаправление)

8 Echo Request (запрос)

B Time Exceeded (время истекло) *

C Parameter Problem (некорректные параметры) *

D Timestamp Request (запрос временной метки)

E Timestamp Reply (временная метка)

F Info Request (запрос информации)

G Info Reply (запрошенная информация)

H Address Mask Request (запрос маски сети)

I Address Mask Reply (маска сети)

Значения для отмеченных звездочкой (*) типов пакетов ограничиваются по умолчанию (показанной выше маской)

12.3.1.11.5 icmp_ignore_bogus_error_responses

Логическая переменная **icmp_ignore_bogus_error_responses** управляет выдачей предупреждений при получении от маршрутизаторов, не соответствующих требованиям RFC 1122, фиктивных откликов на широковещательные запросы. Значение TRUE (1) отключает выдачу таких предупреждений и появление соответствующих записей в журнальных файлах.

По умолчанию используется значение FALSE (0).

12.3.1.12 igmp_max_memberships

Целочисленная переменная **igmp_max_memberships** устанавливает максимальное число multicast-групп, в которые может входить данный хост.

По умолчанию установлено значение 20.

12.3.1.13 Конфигурация интерфейсов

Конфигурационные параметры интерфейсов хранятся в каталогах **/proc/sys/net/ipv4/conf/interface/*** (interface - имя интерфейса, например, eth0) для каждого из сетевых интерфейсов компьютера и в каталоге **/proc/sys/net/ipv4/conf/all/***, содержащем параметры, применяемые ко всем интерфейсам сразу.

12.3.1.13.1 log_martians

Логическая переменная **log_martians** управляет записью информации о пакетах с невозможными адресами в журнальные файлы системы. Запись для конкретного интерфейса активизируется, если установлено значение TRUE (1) в файле **/proc/sys/net/ipv4/conf/all/log_martians** или одноименном файле для данного интерфейса.

12.3.1.13.2 accept_redirects

Логическая переменная **accept_redirects** определяет восприятие пакетов ICMP. Такие пакеты принимаются интерфейсом, при выполнении любого из приведенных ниже условий:

- для интерфейса, поддерживающего пересылку пакетов[1], установлено значение TRUE (1) в обоих файлах /proc/sys/net/ipv4/conf/{all,interface}/accept_redirects;
- для интерфейса, поддерживающего пересылку пакетов, установлено значение TRUE (1) по крайней мере в одном из файлов /proc/sys/net/ipv4/conf/{all,interface}/accept_redirects.

По умолчанию в этих файлах задано значение TRUE (1) для хостов и FALSE (0) для маршрутизаторов.

12.3.1.13.3 forwarding

Логическая переменная **forwarding** управляет возможностью пересылки пакетов IP для одного или всех интерфейсов.

1 *См. описание следующего параметра.*

12.3.1.13.4 mc_forwarding

Логическая переменная **mc_forwarding** управляет пересылкой пакетов с групповыми (multicast) адресами. Для использования групповой адресации требуется ядро, со включенной опцией **CONFIG_MROUTE** (см. параграф 4.4.2.2.6 на стр. 71) и демон, поддерживающий групповую маршрутизацию (mrouted). Требуется также установить значение TRUE (1) в файле **/proc/sys/net/ipv4/conf/all/mc_forwarding**.

12.3.1.13.5 medium_id

Целочисленная переменная **medium_id** указывает тип среды, к которой подключен интерфейс с точки зрения распространения широковещательных пакетов. Если два устройства подключены к разнотипным средам, широковещательные пакеты данной среды будет получать только одно из этих устройств (подключенное к данной среде).

Используемое по умолчанию значение 0 говорит, что к сетевой среде подключен только один интерфейс, а значение 1 говорит о том, что тип среды неизвестен.

В настоящее время этот параметр используется для управления поведением proxy_arp - функции proxy_arp разрешены для пакетов, пересылаемых между устройствами, подключенные к разнотипным средам.

12.3.1.13.6 proxy_arp

Логическая переменная **proxy_arp** управляет выполнением proxy-функций для пакетов ARP. Функция proxy_arp для интерфейса используется, если установлено значение TRUE (1) по крайней мере в одном из файлов **/proc/sys/net/ipv4/conf/{all,interface}/proxy_arp** (см. параграф 12.11.2 на стр. 395).

12.3.1.13.7 shared_media

Логическая переменная **shared_media** определяет поведение по отношению к перенаправлению пакетов для разделяемой среды (shared media redirect). При значении TRUE (1) маршрутизатор будет передавать, а хост - воспринимать перенаправленные пакеты в соответствии с документом RFC 1620. Значение этой переменной может отменять значение переменной secure_redirects (см. описание следующего параметра).

Параметр shared_media для интерфейса будет иметь значение TRUE (1) при установке значений 1 в любом из файлов /proc/sys/net/ipv4/conf/{all,interface}/shared_media. По умолчанию используется значение TRUE.

12.3.1.13.8 secure_redirects

Логическая переменная **secure_redirects** позволяет установить режим восприятия пакетов **ICMP redirect** только от шлюзов, указанных в списке используемых по умолчанию. Параметр **secure_redirects** для интерфейса будет иметь значение TRUE (1), если указана 1 по крайней мере в одном из двух файлов **/proc/sys/net/ipv4/conf/{all,interface}/secure_redirects**. По умолчанию параметр имеет значение TRUE. Значение этого параметра может отменяться значением параметра shared_media, описанного в предыдущем параграфе.

12.3.1.13.9 send_redirects

Логическая переменная **send_redirects** управляет для маршрутизаторов возможностью перенаправления (redirect). Параметр **send_redirects** для интерфейса будет иметь значение TRUE (1), если указано значение 1 по крайней мере в одном из двух файлов **/proc/sys/net/ipv4/conf/{all,interface}/send_redirects**. По умолчанию параметр имеет значение TRUE.

12.3.1.13.10 bootp_relay

При установленном (1) значении логической переменной **bootp_relay** пакеты с адресом отправителя 0.b.c.d, не адресованные данному хосту, будут восприниматься как локальные. Предполагается, что демон BOOTP relay будет воспринимать и пересылать такие пакеты. Для поддержки протокола BOOTP relay интерфейсами хоста в файле **/proc/sys/net/ipv4/conf/all/bootp_relay** также должно быть установлено значение TRUE (1). По умолчанию используется значение FALSE (0).

Эта функция еще не реализована.

12.3.1.13.11 accept_source_route

Логическая переменная **accept_source_route** управляет восприятием пакетов с установленной опцией SRR[1]. Для того, чтобы интерфейс мог принимать такие пакеты значение TRUE (1) должно быть указано также в файле **/proc/sys/net/ipv4/conf/all/ accept_source_route**.

По умолчанию пакеты с опцией SRR не принимаются (FALSE) хостами и принимаются (TRUE) маршрутизаторами.

12.3.1.13.12 rp_filter

Переменная **rp_filter** управляет возможностью проверки пути к отправителю (reversed path) в соответствии с RFC 1812. Значение TRUE (1) включает такую проверку и рекомендуется для хостов с одним сетевым интерфейсом и маршрутизаторов тупиковых (stub) сетей. При использовании такой проверки могут возникать некоторые проблемы в сетях с петлями, использующих протоколы без гарантии доставки (например, sort или RIP), а также статические маршруты. При значении 0 проверка обратного пути не проводится.

1 *Source routing - маршрутизация, заданная отправителем.*

Для проверки корректности отправителя тем или иным интерфейсом системы требуется также установка значения TRUE (1) в файле **/proc/sys/net/ipv4/conf/all/rp_filter**.

По умолчанию используется значение FALSE (0), но некоторые дистрибутивы изменяют это значение в сценариях загрузки системы.

При использовании программы IPsec (см. параграф 12.19.2 на стр. 426), для большинства случаев требуется установка значения 0.

12.3.1.13.13 arp_filter

Логическая переменная **arp_filter** управляет фильтрацией пакетов ARP.

Значение TRUE (1) разрешает использовать множество интерфейсов в одной подсети и выдавать отклики ARP для каждого из этих интерфейсов в зависимости от того, будет ли ядро передавать в ответ на запрос IP-адрес данного интерфейс (этот требует поддержки в сети маршрутизации по адресу отправителя - source based routing). Иными словами, это позволяет определить, какой из интерфейсов одной подсети будет отвечать на запросы ARP.

Используемое по умолчанию значение FALSE (0) позволяет ядру отвечать на запросы arp с адресами от других интерфейсов. Это может показаться ошибкой, но обычно имеет смысл, поскольку повышает вероятность успешного обмена информацией. Адреса IP принадлежат хосту как целому, а не отдельным его интерфейсам. Однако в более сложных системах (например, при распределении нагрузки между каналами - load-balancing) передача адресов других интерфейсов может вызывать проблемы.

Для того, чтобы разрешить (TRUE) **arp_filter** для интерфейса, значение TRUE должно быть установлено по крайней мере в одном из 2 файлов **/proc/sys/net/ipv4/conf/{all,interface}/arp_filter**.

12.3.1.13.14 arp_announce

Целочисленная переменная **arp_announce** определяет тип (уровень) отклика интерфейсов с множеством IP-адресов на запросы ARP.

Установленное по умолчанию значение 0 позволяет использовать любой локальный адрес, заданный для данного интерфейса. Значение 1 используется для попытки избежать передачи локальных адресов, которые не относятся к подсети получателя для данного интерфейса. Этот режим полезен в тех случаях, когда адресат, доступный через данный интерфейс, в откликах ARP требует IP-отправителя из подсети, заданной для локального интерфейса получателя. При генерации запроса проверяются все подсети, включающие IP-адрес получателя и выбирается в качестве исходящего адрес в одной из таких подсетей. При отсутствии таких подсетей используются правила уровня 2. Значение 2 показывает необходимость использования лучшего из локальных адресов для данного получателя. В этом режиме адрес отправителя в пакете IP игнорируется и предпринимается попытка выбрать локальный интерфейс, наиболее подходящий для связи с хостом-получателем. Выбор адресов осуществляется путем просмотра первичных IP-адресов для всех подключенных подсетей и из них выбирается тот, который входит в подсеть получателя. Если подходящего локального адреса нет, выбирается первый локальный адрес исходящего интерфейса или всех прочих интерфейсов в надежде на получение отклика на запрос (онога даже независимо от анонсируемого адреса IP).

Уровень определяется максимальным из двух значений **/proc/sys/net/ipv4/conf/{all,interface}/arp_announce**.

Повышения уровня ограничений увеличивает шансы на получение отклика на запрос, а снижение дает более точную об отправителе.

12.3.1.13.15 arp_ignore

Целочисленная переменная **arp_ignore** определяет режим передачи откликов на полученные запросы ARP, относящиеся к локальным IP-адресам:

0 (по умолчанию) - отклики выдаются для любого локального адреса IP, связанного с любым интерфейсом;

1 - отклики выдаются только в тех случаях, когда интересующий адрес IP связан с принявшим запрос интерфейсом;

2 - отклики выдаются только в тех случаях, когда интересующий адрес IP связан с принявшим запрос интерфейсом и этот адрес находится в той же подсети, из которой поступил запрос;

3 - отклики выдаются только на запросы для глобальных и канальных адресов;

4-7 - зарезервированы;

8 - отклики не выдаются ни для каких локальных интерфейсов.

Используется большее из 2 значений в файлах **/proc/sys/net/ipv4/conf/{all,interface}/arp_ignore**.

12.3.1.13.16 tag

Целочисленная переменная **tag** позволяет задать значение, которое может использоваться при необходимости. По умолчанию установлено значение 0.

12.3.2 Переменные IPv6

Переменные конфигурации IPv6 доступны в файлах **/proc/sys/net/ipv6/***. Для протокола IPv6 не используется отдельных глобальных переменных типа **tcp_***[1]. Поскольку рассмотрение протокола IPv6 выходит за пределы этой

1 *В частности, значения tcp_* для протокола IPv4 применимы и IPv6.*

книги, мы не будем останавливаться на описании этих переменных.

12.3.3 Управление работой моста

Переменные этой группы сохраняются в файлах каталога **/proc/sys/net/bridge/**.

12.3.3.1 bridge-nf-call-arptables

Логическая переменная **bridge-nf-call-arptables** управляет передачей трафика ARP в цепочку FORWARD пакетного фильтра arptables (стр. 151). Установленное по умолчанию значение 1 разрешает передачу пакетов фильтрам, 0 - запрещает.

12.3.3.2 bridge-nf-call-iptables

Логическая переменная **bridge-nf-call-iptables** управляет передачей проходящего через мост трафика IPv4 в цепочки iptables. Используемое по умолчанию значение 1 разрешает передачу пакетов для фильтрации, 0 - запрещает.

12.3.3.3 bridge-nf-filter-vlan-tagged

Логическая переменная **bridge-nf-filter-vlan-tagged** определяет возможность передачи трафика IP/ARP с тегами VLAN программам фильтрации пакетов (arptables/iptables). Значение 1 (установлено по умолчанию) разрешает передачу пакетов с тегами VLAN программам фильтрации, 0 - запрещает.

12.4 Интерфейс сокетов Linux

Интерфейс сетевых сокетов Linux совместим с аналогичным интерфейсом BSD и обеспечивает взаимодействие между пользовательскими процессами и стеком сетевых протоколов, реализованным в ядре. Модули протоколов в ядре сгруппированы по семействам протоколов (**PF_INET**, **PF_IPX**, **PF_PACKET**) и типу сокетов (например, **SOCK_STREAM** или **SOCK_DGRAM**).

Для работы с интерфейсом требуется заголовочный файл

```
#include <sys/socket.h>
```

а вызовы имеют форму

```
mysocket = socket(int socket_family, int socket_type, int protocol);
```

12.4.1 Функции уровня сокета

Эта группа функций применяется пользовательскими процессами для приема и передачи пакетов, а также выполнения других операций с сокетами. Более подробные описания функций доступны в соответствующих страницах руководства[1].

Функция **socket** создает сокет, **connect** соединяет с удаленным сокетом, **bind** связывает сокет с локальным адресом[2], **listen** задает для сокета режим восприятия входящих вызовов, **accept** используется для создания нового сокета принимающего новый входящий вызов, **socketpair** возвращает пару соединенных анонимных сокетов[3].

Функции **send**, **sendto** и **sendmsg** служат для передачи данных через сокет, а функции **recv**, **recvfrom** и **recvmsg** принимают из сокета данные. Функции **poll** и **select** используются для ожидания входящих данных и обозначения готовности к передаче, соответственно. Кроме пеерчисленных функций для чтения и записи данных в сокет могут использоваться стандартные функции ввода-вывода[4] (**write**, **writev**, **sendfile**, **read**, **readv**).

Функция **getsockname** возвращает адрес локального сокета, а **getpeername** - адрес удаленного сокета. Функции **getsockopt** и **setsockopt** служат для определения и установки опций уровня сокета или протокола.

Функция **close** служит для закрытия сокета, а **shutdown** закрывает полнодуплексное соединение между сокетами.

Для сокетов можно использовать операции ввода-вывода без блокировки, установив флаг **O_NONBLOCK** для файлового дескриптора сокета с помощью функции **fcntl**. В таких случаях операции, требующие блокировки, будут обычно возвращать флаг **EAGAIN** (операция должна быть повторена позднее); функция **connect** будет возвращать флаг ошибки **EINPROGRESS**. Пользователь может перейти в режим ожидания событий ввода-вывода с помощью функций **poll** или **select**.

Событие	Флаг	Действия
Read	POLLIN	Прибытие новых данных.
Read	POLLIN	Завершена организация соединения (для сокетов, использующих явные соединения).
Read	POLLHUP	Другая сторона инициировала разрыв соединения.
Read	POLLHUP	Соединение разорвано (только для сокетов, использующих явные соединения. После записи сокета передается также сигнал **SIGPIPE**.

1 *Для получения такой информации можно использовать команду **man 2 <имя функции>***
2 *Адрес сокета задается парой значений “адрес интерфейса - номер порта).*
3 *Эта функция реализована только для некоторых семейств протоколов (в частности, для **PF_UNIX**)*
4 *Стандартные функции ввода-вывода **pread** и **pwrite** для сокетов поддерживаются только с нулевым смещением.*

Событие	Флаг	Действия
Write	POLLOUT	Сокет имеет достаточно места в буфере передачи для записи новых данных.
Read/Write	POLLIN POLLOUT	Вызов функции **connect** для исходящего соединения завершен.
Read/Write	POLLERR	Произошла асинхронная ошибка.
Read/Write	POLLHUP	Другая сторона закрыла соединение в одном направлении.
Read/Write	POLLPRI	Получены срочные (Urgent) данные. После операции передается сигнал **SIGURG**.

Другим способом обработки событий является установка для ядра режима уведомления приложений с помощью сигналов **SIGIO**. В этом случае для файлового дескриптора сокета с помощью функции **fcntl** должен быть установлен флаг **FASYNC**, а для обслуживания сигналов **SIGIO** должен быть установлен соответствующий обработчик (с помощью функции **sigaction**). Обработка сигналов описана ниже (стр. 378).

12.4.2 Опции сокета

Для установки опция сокета служит функция **setsockopt**, а для определения установленных опций - **getsockopt**. Обе функции используются с установленным уровнем **SOL_SOCKET**.

Таблица 118 Опции сокетов

Опция	Описание
SO_KEEPALIVE	Разрешает передачу сообщений keep-alive для сокетов, использующих явные соединения.
SO_OOBINLINE	При установке этого флага данные, полученные не из потока (out-of-band data), помещаются напрямую в принимаемый поток данных. В противном случае такие данные проходят только при установке флага **MSG_OOB** во время приема потока данных.
SO_RCVLOWAT **SO_SNDLOWAT**	Задают минимальный объем данных (в байтах) в буфере, при достижении которого уровень сокета будет передавать данные протоколу (**SO_SNDLOWAT**) или пользователю (**SO_RCVLOWAT**). Эти значения в системах Linux неизменны и равны 1. Функция **getsockopt** будет возвращать значение опции (1), а **setsockopt** всегда возвращает **ENOPROTOOPT**.
SO_RCVTIMEO **SO_SNDTIMEO**	Значения времени ожидания для операций передачи и приема, по истечение которого выдается сообщение об ошибке. В Linux значения этих опций устанавливаются на протокольном уровне и недоступны функциям сокета для чтения или записи.
SO_BSDCOMPAT	Определяет режим совместимости **BSD bug-to-bug**. Эта опция используется только протоколом UDP и в будущих версиях планируется ее удалить[1]. При включенной опции сообщения об ошибках ICMP, полученные для сокета UDP, не будут передаваться пользовательским программам.
SO_PASSCRED[2]	Контролирует прием управляющих сообщений **SCM_CREDENTIALS**.
SO_PEERCRED	Возвращает полномочия (credentials) чужих процессов, подключенных к сокету. Эта опция полезна только для протоколов семейства **PF_UNIX** и может использоваться только с функцией **getsockopt**.
SO_BINDTODEVICE[3]	Связывает данный сокет с указанным отдельным устройством (например, eth0). Если имя устройства не указано (пустое) или размер опции равен 0, связь сокета с устройством удаляется. Опция представляет собой строку переменной длины, завершающуюся нулем. Максимальный размер строки составляет **IFNAMSIZ**. Если сокет связан с интерфейсом, он обслуживает только пакеты, полученные от этого интерфейса. Данная опция работает только с некоторыми типами сокетов (в частности, **AF_INET**) и не поддерживается для пакетного сокета (см. параграф 12.9 на стр. 390).
SO_DEBUG	Управляет режимом отладки для сокета и может использоваться только при установленном флаге возможностей **CAP_NET_ADMIN** или процессами от имени пользователя с UID=0ю
SO_REUSEADDR	Показывает, что правила, применяемые для проверки корректности адресов в вызовах функции **bind**, должны повторять неоднократное использование локальных адресов. Для сокетов **PF_INET** это означает, что сокет может быть связан с адресом (за исключением тех случаев, когда с адресом уже связан прослушивающий сокет). Когда прослушивающий сокет связан с адресом **INADDR_ANY** и заданным портом, не допускается повторное связывание того же порта с любым из локальных адресов.
SO_TYPE	Возвращает тип сокета как целое число (например, SOCK_STREAM). Может использоваться только с функцией **getsockopt**.
SO_ACCEPTCONN	Возвращает значение, показывающее, отмечен ли данный сокет для прослушивания соединений с помощью функции **listen**. Нулевое значение показывает, что сокет не является прослушивающим, а 1 используется для прослушивающих сокетов. Опция может использоваться только с функцией **getsockopt**.

1 *Linux 2.0 также поддерживает эту опцию для сокетов raw, но в версии 2.2 эта опция уже не поддерживалась. Разумней изменить пользовательские программы, нежели применять этот флаг.*
2 *Поддерживается, начиная с Linux 2.2*
3 *Поддерживается, начиная с Linux 2.0.30.*

Опция	Описание
SO_DONTROUTE	Отключает передачу через шлюзы, разрешая передачу только непосредственно подключенным хостам. Такой же эффект может быть достигнут путем установки флага **MSG_DONTROUTE** для операции **send**.
SO_BROADCAST[1]	Устанавливает или возвращает значение флага широковещания. При включенном широковещании сокеты дейтаграмм принимают пакеты, направленные широковещательным адресам, и могут использовать такую адресацию при передаче пакетов. Для потоковых сокетов эта опция не принимается во внимание.
SO_SNDBUF	Устанавливает или возвращает максимальный размер буфера передачи для сокета (в байтах). Принятый по умолчанию размер буфера устанавливается переменной SysCtl **wmem_default**, а максимальный - **wmem_max**.
SO_RCVBUF	Устанавливает или возвращает максимальный размер буфера приема для сокета (в байтах). Принятый по умолчанию размер буфера устанавливается переменной SysCtl **rmem_default**, а максимальный - **rmem_max**.

SO_LINGER	Устанавливает или возвращает значения параметров замедления для сокета. В качестве аргумента опции используется структура
	`struct linger {`
	`  int  l_onoff;  /* замедление активизировано */`
	`  int  l_linger;  /* величина замедления в секундах */`
	`};`
	При включенной опции функции **close** и **shutdown** не будут возвращать управление, пока все помещенные в очередь сокета сообщения не будут успешно переданы или пока не истечет заданное параметром **linger** значение. Если же опция выключена, функцию будут незамедлительно возвращать управление, закрывая соединение в фоновом режиме. Когда сокет закрывается в процессе выполнения процедуры **exit**, всегда применяется выход с задержкой (опция включена).

SO_PRIORITY	Устанавливает заданный протоколом приоритет для всех передаваемых сокетом пакетов. Linux использует уровень приоритета для управления очередями - пакеты с высоким приоритетом могут обрабатываться первыми в зависимости от выбранной дисциплины очередей. Для протокола IP опция также устанавливает поле TOS в исходящих пакетах.
SO_ERROR	Возвращает и сбрасывает флаг ошибки для сокета. Может использоваться только с функцией **getsockopt**.

12.4.3 Сигналы

При попытках записи в протокольный (использующий явные соединения) сокет, переведенный в состояние shutdown локальной или удаленной стороной, записывающему процессу передается сигнал **SIGPIPE** и функция записи возвращает значение **EPIPE**. Сигнал не будет передаваться, если функция была вызвана с флагом **MSG_NOSIGNAL**.

При запросе **FIOSETOWN** (fcntl) или **SIOCSPGRP** (ioctl) передается сигнал **SIGIO** в момент события ввода-вывода. Можно использовать в обработчике сигналов функции **poll** и **select** для определения сокета, с которым связано событие. Другим вариантом определения (Linux 2.2) является передача сигнала в реальном масштабе времени с использованием **F_SETSIG** (fcntl); обработчик таких сигналов будет вызываться с дескриптором файла в поле **si_fd** структуры **siginfo_t**[2].

В некоторых случаях (например, при обращении множества процессов к одному сокету) условие, вызвавшее сигнал SIGIO, может уже исчезнуть к тому моменту, когда процесс будет реагировать на сигнал. В таких случаях процесс должен ждать, пока Linux не повторит передачу сигнала.

12.4.4 Параметры SysCtl

Параметры sysctl для сокетов поддерживаются ядром Linux, начиная с версии 2.2 и хранятся в дереве каталогов **/proc/sys/net/core/***.

Linux предполагает, что половина буферов приема-передачи используется внутренними структурами ядра, поэтому доступные процессам буферы имеют размеры вдвое меньшие по сравнению со значениями соответствующих параметров sysctl.

Таблица 119. Параметры SysCtl для сокетов.

Параметр	Описание
rmem_default	Размер используемого по умолчанию буфера приема для сокета в байтах.
rmem_max	Максимальный размер буфера приема для сокета, который пользователь может установить с помощью опции **SO_RCVBUF**.

1 *Будьте осторожны при использовании опции SO_BROADCAST для протокола IP - эта не является привилегированной в Linux. Небрежное отношение к широковещательным пакетам может привести к перегрузке сети. При разработке прикладных протоколов лучше использовать групповую адресацию вместо передачи широковещательных пакетов.*

2 *Дополнительную информацию вы сможете получить по команде **man fcntl**.*

Параметр	Описание
wmem_default	Размер используемого по умолчанию буфера передачи для сокета в байтах.
wmem_max	Максимальный размер буфера передачи для сокета, который пользователь может установить с помощью опции **SO_SNDBUF** (стр. 378).
message_cost **message_burst**	Задают параметры, ограничивающие количество предупреждений, вызываемых внешними сетевыми событиями.
netdev_max_backlog	Максимальное число пакетов в глобальной входной очереди.
optmem_max	Максимальный объем вспомогательных данных и пользовательских данных управления для сокета.

12.4.5 Операции IOCTL

Для доступа к операциям IOCTL служит системная функция ioctl (Приложение 12.13).

```
error = ioctl(ip_socket, ioctl_type, &value_result);
```

Таблица 120. Операции IOCTL для сокетов.

Параметр	Описание
SIOCGSTAMP	Возвращает структуру **timeval**[1], содержащую временную метку приема последнего пакета, переданного пользователю. Параметр полезен для точного измерения времени кругового обхода.
SIOCSPGRP	Задает процесс или группу процессов для передачи им сигналов **SIGIO** или **SIGURG** при завершении асинхронных операций ввода-вывода или поступлении срочных данных. Аргументом служит указатель на **pid_t**. Положительный аргумент используется для передачи сигнала процессу, отрицательный - для передачи сигнала группе с использованием в качестве аргумента абсолютного значения. Процесс может передавать сигналы только себе или своей группе за исключением тех случаев, когда он запущен от имени пользователя с UID=0 или имеет флаг возможностей **CAP_KILL**.
FIOASYNC	Меняет флаг **O_ASYNC** управления асинхронным вводом-выводом для сокета. Асинхронный режим ввода-вывода означает, что сигнал **SIGIO** или сигнал, установленный **F_SETSIG**, сбрасывается при новом событии ввода-вывода.
SIOCGPGRP	Определяет текущий процесс или группу, получающие сигнал **SIGIO** или **SIGURG**. Если такого процесса или группы нет, возвращается 0.

12.4.6 Операции fcntl

Операции fctnl **FIOGETOWN** и **FIOSETOWN** совпадают с **SIOCSPGRP** (ioctl).

12.5 Реализация протокола IP в Linux

В ядре Linux реализована поддержка протокола IP версии 4 в соответствии с требованиями RFC 791 и RFC 1122. Реализация IP в ядре Linux включает поддержку групповой маршрутизации уровня 2 в соответствии с RFC 1122. Кроме того, ядро Linux обеспечивает маршрутизацию трафика IP с возможностью фильтрации пакетов.

Интерфейс API совместим с интерфейсом сокетов BSD. IP-сокеты создается путем вызова функции

```
socket(PF_INET, socket_type, protocol).
```

Параметр **socket_type** задает тип создаваемого сокета - **SOCK_STREAM** открывает сокет tcp (приложение на стр.), **SOCK_DGRAM** - сокет udp (приложение на стр.), а **SOCK_RAW** - сокет raw (приложение на стр.). Параметр **protocol** задает тип протокола IP, указываемый в заголовке пакета IP. Этот параметр может принимать значение 0 или **IPPROTO_TCP** для сокетов TCP, 0 или **IPPROTO_UDP** для сокетов UDP. Для SOCK_RAW можно указывать любой зарегистрированный протокол IP[2].

Если процесс планирует принимать входящие пакеты или соединения, он должен связать созданный сокет с адресом локального интерфейса, используя функцию **bind**. С каждой локальной парой адрес-порт может быть связан только один сокет IP. Если при вызове функции bind указать параметр **INADDR_ANY**, сокет будет связан со всеми локальными интерфейсами. При вызове функций **listen** или **connect** для несвязанного сокета, такой сокет будет автоматически привязан к случайно выбранному свободному порту, а в качестве локального адреса будет использован **INADDR_ANY**.

Адрес локального сокета TCP, который был связан, в течение некоторого времени после закрытия сокета остается недоступным, если не был установлен флаг **SO_REUSEADDR**[3]. Следует проявлять

1 *Описание структуры можно получить по команде man setitimer.*
2 *Список зарегистрированных протоколов можно посмотреть в RFC 1700. Однако этот документ изрядно устарел и более не будет обновляться, поэтому лучше обращаться к списку зарегистрированных протоколов на сайте http://www.iana.org/assignments/protocol-numbers*
3 *Этот флаг следует использовать с осторожностью, поскольку он снижает уровень надежности TCP.*

12.5.1 Адресация сокетов IP

Адресация сокетов IP обеспечивается парами “адрес IP - номер порта”. Протокол IP, как таковой, не связан с номерами портов, но эти номера используются протоколами вышележащих уровней типа UDP и TCP. Для сокетов типа **raw** поле **sin_port** указывает на протокол IP.

```
struct sockaddr_in {
    sa_family_t    sin_family; /* семейство адресов: AF_INET */
    u_int16_t      sin_port;   /* номер порта[1] */
    struct in_addr  sin_addr;  /* адрес IP */
};

/* адрес IP */
struct in_addr {
    u_int32_t      s_addr;     /* адрес[3] */
};
```

Значение поля **sin_family** всегда должно быть равно **AF_INET**. Большинство сетевых функций ядра Linux 2.2 будет возвращать код ошибки **EINVAL**, если это поле опущено. Переменная **sin_port** содержит номер порта, используемого сокетом. Номера портов менее 1024 зарезервированы и такие порты часто называют привилегированными. Связывать сокеты с привилегированными портами могут только процессы, выполняющиеся от имени пользователя с UID=0, или процессы с флагом возможности **CAP_NET_BIND_SERVICE**.

Поле **sin_addr** содержит IP-адрес хоста как структуру с единственным полем **s_addr**, содержащим адрес хоста, заданный в формате **network byte order**. Для работы со структурами **in_addr** следует использовать библиотечные функции **inet_aton**, **inet_addr**, **inet_makeaddr** или функцию прямой трансляции имен **gethostbyname**. Адреса IPv4 делятся на индивидуальные (unicast), групповые (multicast) и широковещательные (broadcast). Unicast-адрес задает один интерфейс хоста, групповой адрес - множество интерфейсов (группу), а широковещательный адрес указывают на все хосты сети (подсети). Дейтаграммы с широковещательным адресом получателя можно принимать или передавать только при наличии у сокета флага **SO_BROADCAST** (стр. 378). В текущей реализации протокола для сокетов, работающих на основе явных соединений, можно использовать только индивидуальную адресацию.

Подчеркнем еще раз что все параметры, содержащие адреса и номера портов используют так называемый сетевой формат (big-endian или network byte order). Все функции стандартной библиотеки, работающие с сетевыми адресами и номерами портов, принимают параметры в сетевом формате. Для преобразования числа в сетевой формат служит системная функция **htons**.

Некоторые адреса IP зарезервированы для определенных целей: **INADDR_LOOPBACK** (127.0.0.1) всегда используется для петлевого (loopback) интерфейса локального хоста; **INADDR_ANY** (0.0.0.0) обозначает все адреса для привязки сокета; **INADDR_BROADCAST** (255.255.255.255) обозначает любой хост и в силу сложившейся традиции, также используется для связывания сокета со всеми адресами хоста.

12.5.2 Опции сокета IP

Для сокетов IP поддерживается специфических набор опций, которые можно устанавливать с помощью функции **setsockopt** и считывать с помощью **getsockopt**. Параметр "уровень опции сокета" этих функций равен **SOL_IP**. Для логических переменных 0 соответствует значению FALSE, все остальные числа - значению TRUE.

Таблица 121. Опции сокетов IP (SOL_IP).

Опция	Описание
IP_OPTIONS	Устанавливает или возвращает опции IP для пакетов, передаваемых через данный сокет. Аргументами служат указатель на буфер опций в памяти и размер поля опций. Опции для сокета устанавливаются с помощью функции **setsockopt**. Для IPv4 максимальный размер опций составляет 40 байтов. Описание опций IP вы сможете найти в RFC 791. Если пакет запроса на соединение типа SOCK_STREAM содержит опции IP, такие же опции (с обращенными заголовками маршрутизации) будут использоваться для созданного по запросу сокета. Не допускается изменение опций входящих пакетов после организации соединения. По умолчанию обработка опций **source routing** из входящих пакетов отключена и включить ее можно с помощью параметра **accept_source_route** (sysctl, см. параграф 12.3.1.13.11 на стр. 374). Остальные опции таких пакетов обрабатываются как обычно. Для сокетов дейтаграмм (см. параграф 12.7 на стр. 387) опции IP может устанавливать только локальный пользователь. Вызов функции **getsockopt** с аргументом **IP_OPTIONS** помещает заданные опцией аргументы в заголовок IP передаваемого пакета.

1 *Номера портов и адреса представляются в формате big-endian или network byte order, когда первым указывается (передается) наиболее значимый (старший) байт*

Опция	Описание

IP_PKTINFO[1]	Передает служебное сообщение **IP_PKTINFO**, содержащее структуру **pktinfo** с информацией о входящем пакете. Эта опция применима только к сокетам дейтаграмм (см. параграф 12.7 на стр. 387). Аргументом служит флаг, который говорит сокету нужно ли передавать сообщение **IP_PKTINFO**. Само сообщение может быть передано или принято только в качестве управляющего (вместе с пакетом) с помощью функции **recvmsg** или **sendmsg**.
	`struct in_pktinfo {`
	`  unsigned int  ipi_ifindex;  /* индекс интерфейса */`
	`  struct in_addr ipi_spec_dst; /* локальный адрес */`
	`  struct in_addr ipi_addr;  /* адрес получателя из заголовка */`
	`};`
	ipi_ifindex - уникальный интдекс интерфейса, через который принят пакет; **ipi_spec_dst** - локальный адрес пакета; **ipi_addr** - адрес получателя в заголовке пакет. Если **IP_PKTINFO** передается функции **sendmsg**, исходящий пакет будет передан через интерфейс, заданный параметром ipi_ifindex по адресу, указанному в **ipi_spec_dst**.

IP_RECVTOS	Управляет передачей служебных сообщений **IP_TOS** со входящими пакетами. Управляющие сообщения содержат поле **Type of Service/Precedence**, определяющие соответствующие поля заголовка пакетов.
IP_RECVTTL	При установке этого флага передается управляющее сообщение **IP_RECVTTL** с полем времени жизни из принятого пакета. Эта опция не поддерживается для сокетом **SOCK_STREAM**.
IP_RECVOPTS	Передает пользователю все опции IP из входящего пакета как управляющее сообщение **IP_OPTIONS**. Заголовок маршрутизации и другие опции уже установлены для локального хоста. Эта опция не поддерживается для сокетов **SOCK_STREAM**.
IP_RETOPTS	Идентична **IP_RECVOPTS**, но возвращает необработанные опции с пустой временной меткой и маршрутной записью для данного хопа.
IP_TOS	Устанавливает или определяет значение поля TOS (Type-Of-Service) для каждого пакета IP, передаваемого данным сокетом. Поле типа обслуживания может использоваться для приоритизации пакетов. Поле TOS занимает 1 байт. Определены следующие флаги TOS: **IPTOS_LOWDELAY** - для минимизации задержки интерактивного трафика; **IPTOS_THROUGHPUT** - для оптимизации пропускной способности; **IPTOS_RELIABILITY** - для оптимизации надежности доставки; **IPTOS_MINCOST** - малозначимые данные, которые могут передаваться в последнюю очередь. В поле TOS может быть указано не более одного из перечисленных значений. Все остальные биты поля не используются и должны быть сброшены. Linux по умолчанию передает дейтаграммы **IPTOS_LOWDELAY** первыми, но порядок обработки может быть изменен дисциплинами очередей. Некоторые уровни с высоким приоритетом могут потребовать запуска процессов от имени пользователя с UID=0 или наличия флага возможностей **CAP_NET_ADMIN**. Уровень приоритета можно установить независимо от протокола с помощью опции сокета **SO_PRIORITY** (параграф 12.4.2 на стр. 378).
IP_TTL	Устанавливает или определяет значение поля TTL для каждого пакета, передаваемого с использованием данного сокета.
IP_HDRINCL	Установленный флаг означает, что пользователь уже добавил заголовок IP перед своими данными. Опцию допустимо использовать только с сокетами SOCK_RAW (параграф 12.8 на стр. 388). При установке этого флага значения опций **IP_OPTIONS**, **IP_TTL** и **IP_TOS** игнорируются.

1 *Некоторые реализации сокетов BSD в других ОС используют опции сокета **IP_RCVDSTADDR** и **IP_RECVIF** для определения адреса получателя и физического интерфейса. Linux использует для решения таких задач опцию **IP_PKTINFO**.*

Опция	Описание

IP_RECVERR[1]	Включает расширенные средства передачи сообщений об ошибках с повышением гарантии их доставки. При использовании с сокетами дейтаграмм все сгенерированные сообщения об ошибках будут помещаться в специальную очередь данного сокета. Когда возникает ошибка при пользовательской операции с сокетом, сообщение об этой ошибке можно получить с помощью функции **recvmsg** с установленным флагом **MSG_ERRQUEUE**. Описывающая ошибку структура **sock_extended_err** будет возвращена в служебном сообщении с типом **IP_RECVERR** и уровнем **SOL_IP**. Такая возможность весьма полезна для надежной обработки ошибок в неподключенных сокетах. Данные служебного сообщения, получаемого из очереди, содержат ошибочный пакет.
	
	Структура **sock_extended_err** определена как:
	
	`#define SO_EE_ORIGIN_NONE 0`
	`#define SO_EE_ORIGIN_LOCAL 1`
	`#define SO_EE_ORIGIN_ICMP 2`
	`#define SO_EE_ORIGIN_ICMP6 3`
	
	`struct sock_extended_err {`
	`u_int32_t ee_errno; /* код ошибки */`
	`u_int8_t ee_origin; /* код источника ошибки */`
	`u_int8_t ee_type; /* тип ICMP */`
	`u_int8_t ee_code; /* лщв ICMP */`
	`u_int8_t ee_pad;`
	`u_int32_t ee_info; /* дополнительная информация */`
	`u_int32_t ee_data; /* прочие данные */`
	`/* Структура может включать дополнительные поля данных */`
	`};`
	
	`struct sockaddr *SO_EE_OFFENDER(struct sock_extended_err *);`
	Макрос **SO_EE_OFFENDER** возвращает указатель на адрес сетевого объекта, с которым связана ошибка, на основании принятого указателя на служебное сообщение. Если адрес неизвестен, поле **sa_family** структуры **sockaddr** содержит значение **AF_UNSPEC**, а остальные поля **sockaddr** не определены.
	
	IP использует структуру **sock_extended_err** следующим образом: в поле **ee_origin** устанавливается значение **SO_EE_ORIGIN_ICMP** для сообщений об ошибках, полученных в пакетах ICMP, и **SO_EE_ORIGIN_LOCAL** для локально сгенерированных сообщений. Неизвестные значения следует игнорировать. Поля **ee_type** и **ee_code** устанавливаются из соответствующих полей заголовка ICMP. Для сообщений **EMSGSIZE п**оле **ee_info** содержит определенное значение MTU.
	
	Сообщение включает также значение **sockaddr_in** вызвавшего ошибку узла, к которому можно обратиться с помощью макроса **SO_EE_OFFENDER**. Поле **sin_family**, возвращенное макросом **SO_EE_OFFENDER**. Содержит значение **AF_UNSPEC**, если источник ошибки неизвестен. Если ошибка произошла в сети, все опции IP (IP_OPTIONS, IP_TTL и т. д.), поддерживаемые сокетом и содержащиеся в связанном с ошибкой пакете, передаются как управляющее сообщение. Данные из вызвавшего ошибку пакета возвращаются как обычные данные. Отметим, что протокол TCP не использует очереди ошибок и флаг **MSG_ERRQUEUE** недопустим для сокетов **SOCK_STREAM**. Таким образом, функции сокета всегда возвращают результат или код ошибки **SO_ERROR**.
	
	Для сокетов **raw** опция **IP_RECVERR** разрешает передачу всех полученных ICMP-сообщений об ошибках приложению. При выключенной опции сообщения об ошибках выдаются только для подключенных сокетов.
	
	По умолчанию опция **IP_RECVERR** отключена.

1 *Опция объявлена в заголовочном файле **linux/errqueue.h***

Опция	Описание
IP_PMTU_DISCOVER	Устанавливает или определяет значение **Path MTU Discovery** для сокета. При включенной опции Linux будет выполнять для сокета операции **Path MTU Discovery** в соответствии с RFC 1191. Для всех исходящих пакетов будет устанавливаться флаг запрета фрагментирования. По умолчанию режим определения MTU для сокетов SOCK_STREAM задается параметром **ip_no_pmtu_disc** (sysctl, см. параграф 12.3.1.3 на стр. 367), а для всех остальных протоколов отключена. Для сокетов, не относящихся к типу **SOCK_STREAM**, пользователь отвечает за размер передаваемых пакетов и их повторную передачу при необходимости. Если данный флаг установлен, ядро будет отбрасывать пакеты, размер которых превышает известное для пути передачи значение MTU, возвращая код ошибки **EMSGSIZE**. Определены следующие значения флагов **Path MTU Discovery** **IP_PMTUDISC_WANT** Использовать помаршрутные установки **IP_PMTUDISC_DONT** Никогда не выполнять Path MTU Discovery. **IP_PMTUDISC_DO** Всегда выполнять операции Path MTU Discovery. При включенном определении MTU ядро автоматически отслеживает значение MTU для получателей пакетов. При подключении к заданному хосту с помощью функции **connect** известное в данный момент значение MTU можно определить с помощью опции сокета **IP_MTU** (например, после получения сообщения об ошибке EMSGSIZE). Значение MTU может изменяться в процессе работы. Для сокетов, не организующих явных соединений и работающих с множеством адресатов, новое значение MTU для конкретного адресата можно получить с использованием очереди ошибок (см. описание опции **IP_RECVERR** на стр. 382). Сообщение об ошибке будет помещаться в очередь при каждом входящем обновлении MTU. В процессе определения MTU первые пакеты из сокетов дейтаграмм могут отбрасываться. Приложения, использующие UDP, должны понимать это обстоятельство и учитывать его в своей стратегии повтора передачи. Чтобы инициировать процесс определения MTU для неподключенных сокетов, можно начать с передачи дейтаграмм большого размера (с заголовком до 64K) и впоследствии изменить значение в соответствии с обновлениями **path MTU**. Для начальной оценки MTU соедините сокет дейтаграмм с адресатом, используя функцию **connect** и определите значение MTU путем вызова функции **getsockopt** с опцией IP_MTU.
IP_MTU	Определяет текущее значение **path MTU** для данного сокета. Опция корректна только для соединенных сокетов и может использоваться только с функцией **getsockopt**.
IP_ROUTER_ALERT	Передает данному сокету все пакеты, которые должны пересылаться с установленным флагом **IP Router Alert**. Опция может использоваться только для raw-сокетов. Этот флаг может быть полезен, например, для демонов RSVP, работающих в пользовательском пространстве. Перехваченные пакеты не будут пересылаться ядром и пользовательское приложение должно принять на себя ответственность за их дальнейшую пересылку. Привязка сокета не принимается во внимание и пакеты фильтруются только протоколом.
IP_MULTICAST_TTL	Устанавливает или определяет значение поля TTL для исходящих от данного сокета пакетов с групповыми адресами. Для групповых пакетов важно установить минимально возможное значение времени жизни. По умолчанию TTL=1 и пакеты с групповыми адресами не будут покидать локальную сеть, пока пользовательская программа явно не запросит иного.
IP_MULTICAST_LOOP	Устанавливает или определяет логическое значение флага возврата multicast-пакетов локальным сокетам.

IP_ADD_MEMBERSHIP	Присоединяет сокет к multicast-группе. Аргументом опции служит структура **structip_mreqn**
	
	`struct ip_mreqn {`
	`  struct in_addr imr_multiaddr; /* IP-адрес multicast-группы */`
	`  struct in_addr imr_address;/* IP-адрес локального интерфейса */`
	`  int  imr_ifindex;  /* индекс интерфеса */`
	`};`
	Поле **imr_multiaddr** задает корректный multicast-адрес группы, к которой присоединяется сокет, а **in_addr -** адрес локального интерфейса[1], через который осуществляется подключение к группе. Поле **imr_ifindex** содержит индекс интерфейса, который должен быть добавлен/удален в группу, или 0 для индикации любого интерфейса.
	
	Для обеспечения обратной совместимости поддерживаются также старые структуры **ip_mreq**, отличающиеся только отсутствием поля **imr_ifindex**.
	
	Опция применима только с функцией **setsockopt**.

IP_DROP_MEMBERSHIP	Удаляет сокет из multicast-группы. В качестве аргумента может использоваться структура **ip_mreqn** или **ip_mreq**, описанная в предыдущей опции.

1 *Если это поле содержит значение INADDR_ANY, система выбирает интерфейс, подходящий для соединения с группой.*

Опция	Описание
IP_MULTICAST_IF	Задает локальное устройство для группового (multicast) сокета. В качестве аргумента передается структура **ip_mreqn** или **ip_mreq**, описанная выше (опция **IP_ADD_MEMBERSHIP**). При передаче сокету некорректной опции возвращается значение **ENOPROTOOPT**.

12.5.3 Параметры SysCtl

Протокол IP использует интерфейс SysCtl для контроля и настройки ряда глобальных опций. Описание параметров SysCtl для протокола IP приведено в Приложении 12.3.

12.5.4 Операции IOCTL

Все операции IOCTL для протокола socket (параграф 12.4.5 на стр. 379) применимы и для протокола IP. Операции IOCTL для настройки межсетевых экранов описаны в руководствах соответствующих программ (**ipfw**, **ipchains**, **iptables**). Операции работы с сетевыми устройствами на аппаратном уровне описаны в Приложении 12.12.

12.5.5 Коды ошибок

Таблица 122. Коды ошибок протокола IP.

Код	Описание
ENOTCONN	Для неподключенного сокета была использована операция, поддерживаемая только для подключенных сокетов.
EINVAL	Передан недопустимый аргумент. Для передачи пакетов такая ошибка может быть связана с адресацией пакета в “черную дыру” (blackhole route).
EMSGSIZE	Размер дейтаграммы, которую запрещено фрагментировать, превышает значение MTU.
EACCES	Попытка выполнения пользователем недозволенной операции. К таким операциям относятся: передача пакетов по широковещательному адресу при отсутствии флага **SO_BROADCAST**, передача пакетов через запрещенный маршрут, попытка изменения настроек межсетевого экрана без флага возможностей **CAP_NET_ADMIN** или эффективного идентификатора UID=0, связывание сокета с зарезервированным портом при отсутствии флага возможностей **CAP_NET_BIND_SERVICE** и эффективном значении UID<>0.
EADDRINUSE	Попытка связать сокет с используемым адресом.
ENOPROTOOPT	Передана некорректная опция.
EOPNOTSUPP	Передана некорректная опция.
EPERM	Пользователь не имеет полномочий для установки высокого уровня приоритета, изменения конфигурации или передачи сигнала запрошенному процессу или группе.
EADDRNOTAVAIL	Запрошен несуществующий интерфейс или указан нелокальный адрес отправителя.
EAGAIN	Операция над неблокируемым сокетом, требующая блокировки.
ESOCKTNOSUPPORT	Запрошен неизвестный тип сокета или конфигурация сокета не настроена.
EISCONN	Для сокета, уже установившего соединение, вызвана функция **connect**.
EALREADY	Операция соединения для неблокируемого сокета уже выполняется.
ECONNABORTED	Соединение было закрыто в период работы функции **accept**.
EPIPE	Соединение неожиданно закрылось или было разорвано удаленной стороной.
ENOENT	Вызов **SIOCGSTAMP** для сокета, который еще не получил ни одного пакета.
EHOSTUNREACH	В таблице маршрутизации нет корректной записи для адресата. Причиной может послужить сообщение ICMP от удаленного маршрутизатора или состояние локальной таблицы маршрутов.
ENODEV	Сетевое устройство недоступно или не может передавать пакеты IP.
ENOPKG	Не настроена поддержка IP в ядре Linux.
ENOBUFS	Недостаточно свободной памяти. Это сообщение вызвано ограниченным размером буфера сокетов или нехваткой оперативной памяти в системе.
ENOMEM	Недостаточно свободной памяти. Это сообщение вызвано ограниченным размером буфера сокетов или нехваткой оперативной памяти в системе.

Протоколы вышележащих уровней **TCP** (Приложение 12.6), **raw** (Приложение 12.8), **UDP** (Приложение 12.7) и **socket** (Приложение 12.4) могут добавлять свои сообщения об ошибках.

12.5.6 Известные проблемы

1) Слишком много противоречивых кодов ошибок.

2) Не описаны операции IOCTL для настройки IP-опций интерфейсов и таблиц ARP.

3) Некоторые версии glibc не включают описание структуры **in_pktinfo**. Вы можете просто скопировать это описание в glibc со стр. 381.

12.6 Реализация протокола TCP в Linux

Модуль реализация протокола TCP в Linux соответствует спецификациям RFC 793, RFC 1122 и RFC 2001 с дополнениями NewReno и SACK. Протокол обеспечивает поддержку полнодуплексных потоковых соединений с гарантированной доставкой между парами сокетов, работающих на основе стека IP (параграф 12.5 на стр. 379) версии 4 или 6. Протокол TCP гарантирует доставку пакетов с соблюдением их очередности и обеспечивает повтор передачи в случае потери пакетов. Протокол обеспечивает подсчет и проверку контрольных сумм для каждого пакета. Протокол TCP не сохраняет границы записей.

Сокет TCP сразу после его создания не связан с локальным или удаленным адресом и не определен полностью. Для организации исходящего соединения TCP используется системный вызов **connect**, позволяющий организовать связь с другим сокетом TCP. Для приема нового входящего соединения сокет нужно связать с локальным адресом и портом путем вызова функции **bind**, а после этого следует воспользоваться функцией **listen** для перевода сокета в состояние приема входящих данных. После этого можно принимать входящие соединения с помощью функции **accept**, создающей новые сокеты. Сокет после успешного вызова функции **accept** или **connect** является полностью определенным и может передавать данные. Данные не могут передаваться через сокеты, находящиеся в состоянии прослушивания (**listen**) или еще не подключенные.

Linux поддерживает расширения TCP в соответствии с документом RFC 1323. Эти расширения включают защиту от поддельных порядковых номеров PAWS[1], масштабирование окон (Window Scaling) и временные метки (Timestamp). Масштабирование позволяет использовать окна TCP размером более 64 кбайт, что может быть весьма полезно для каналов в большими задержками или широкой полосой. Для использования больших окон требуется увеличение размера приемных и передающих буферов, что можно сделать для всех сокетов с помощью глобальных переменных SysCtl **/proc/net/ipv4/tcp_wmem** (параграф 12.3.1.7.26 на стр. 370) и **/proc/net/ipv4/tcp_rmem** (параграф 12.3.1.7.27 на стр. 370) или для отдельного сокета с помощью вызова функции **setsockopt** с опциями **SO_SNDBUF** (стр. 378) и **SO_RCVBUF** (стр. 378).

Максимальный размер буферов, выделяемых с помощью опций **SO_SNDBUF** и **SO_RCVBUF**, ограничен глобальными переменными SysCtl **/proc/net/core/rmem_max** (стр. 378) и **/proc/net/core/wmem_max** (стр. 379). Отметим, что протокол TCP в реальности выделяет удвоенный размер буферов, запрошенных с помощью **setsockopt** и последующий вызов функции **getsockopt** не будет возвращать такое же значение размера буфера, как было задано при вызове **setsockopt**. Модуль TCP использует дополнительную память для внутренних структур ядра и решения административных задач, а переменные SysCtl отражают наибольшие размеры в сравнении с реальными окнами TCP. Для отдельных соединений размер буфера сокета должен быть установлен до вызова функции **listen** или **connect**, иначе будет использоваться принятый по умолчанию размер буферов.

Модуль TCP поддерживает приоритетную доставку срочных (urgent) данных. Флаг **urgent** служит для того, чтобы передать приемной стороне информацию о наличии в потоке данных важного сообщения, требующего срочной обработки. Для передачи срочных данных функция send должна вызываться с опцией **MSG_OOB**. При получении срочных данных ядро передает сигнал **SIGURG** читающему данные процессу, либо процессу или группе процессов, заданных для сокета с использованием **SIOCSPGRP** или **FIOSETOWN**. При включенной для сокета опции **SO_OOBINLINE** срочные данные помещаются в обычный поток данных (и могут быть проверены с помощью **SIOCATMARK**), в противном случае эти данные могут быть получены только при установке флага **MSG_OOB** для вызова функции **sendmsg**.

В ядро серии 2.4 было внесено множество изменений, повышающих производительность и уровень масштабирования TCP, а также расширения функциональности протокола. К таким изменениям относится поддержка **zerocopy sendfile**, явное уведомление о насыщении (ECN[2]), новые возможности управления сокетами **TIME_WAIT**, опции **keep-alive** и расширение Duplicate SACK.

12.6.1 Форматы адресов

Протокол TCP работает на основе протокола IP (Приложение 12.5) и использует формат адресов IP. Реализация TCP поддерживает только парные соединения (точка-точка); широковещательные и групповые сообщения TCP не поддерживаются.

12.6.2 Параметры SysCtl

Параметры SysCtl для протокола TCP описаны в параграфе 12.3.1.7 (стр. 368). Кроме того, на работу протокола влияет большинство параметров SysCtl, определенных для протокола IP (Приложение 12.3).

12.6.3 Опции сокета TCP

Для прочтения опций сокета TCP служит функция **getsockopt**, а для установки опций - **setsockopt.** Опции сокетов TCP относятся к семейству **SOL_TCP**. Кроме того, на работу протокола TCP оказывает влияние большинство опций семейства **SOL_IP** (параграф 12.5.2 на стр. 380).

Все опции TCP, за исключением **TCP_MAXSEG** и **TCP_NODELAY**, не следует использовать в переносимых приложениях.

1 *Protection Against Wrapped Sequence Numbers.*
2 *Explicit Congestion Notification*

Таблица 123 Опции сокета TCP (SOL_TCP)

Опция	Описание
TCP_CORK	Установка этого флага отключает передачу неполных кадров (partial frame). Все помещенные в очередь неполные кадры будут передаваться после сброса флага. Эта опция может быть полезна при работе с заголовками до вызова функции **sendfile**, а также для оптимизации пропускной способности. Не допускается совместное использование данного флага с опцией **TCP_NODELAY**.
TCP_DEFER_ACCEPT	Эта опция позволяет прослушивающему сокету “спать” до момента прибытия входящих данных. Целое значение опции задает максимальную продолжительность (в секундах) попыток TCP завершить организацию соединения.
TCP_INFO	Эта опция служит для сбора информации о сокете. Ядро возвращает структуру **tcp_info**, определенную в файле **/usr/include/linux/tcp.h**.
TCP_KEEPCNT	Задает максимальное число проб keepalive, которые протокол TCP должен передать до того, как соединение будет разорвано (drop).
TCP_KEEPIDLE	Задает количество секунд бездействия, по прошествии которых протокол TCP начинает передачу проб keepalive, если для сокета установлена опция **SO_KEEPALIVE**.
TCP_KEEPINTVL	Интервал (в секундах) между передачей последовательных проб keepalive.
TCP_LINGER2	Задает время жизни сокетов в полуразорванном (orphaned) состоянии FIN_WAIT2. Эту опцию можно использовать для отмены глобального параметра SysCtl **tcp_fin_timeout** (стр. 369) для данного сокета. Значение этой опции не конфликтует с установкой опции сокета **SO_LINGER** (стр. 378).
TCP_MAXSEG	Максимальный размер сегмента для исходящих пакетов TCP. Если эта опция установлена до организации соединения, она также меняет значение MSS, анонсируемое удаленной стороне в пакете организации соединения. В качестве максимального размера сегмента могут использоваться значения, не превышающие размер MTU для используемого интерфейса. Протокол TCP также имеет верхнее и нижнее ограничение для этой опции.
TCP_NODELAY	Флаг запрета алгоритма Нэгла (Nagle). При наличии этого флага пакеты передаются с максимально возможной частотой без внесения дополнительных задержек даже при наличии лишь малого объема данных. При отключенной опции данные буферизуются и передаются в сеть после того, как их объем достигнет определенного порога. Использование алгоритма Нэгла позволяет оптимизировать расход полосы каналов. Данную опцию нельзя использовать вместе с флагом **TCP_CORK**.
TCP_QUICKACK	Эта опция переключает флаг использования режима **quickack**. При включенном режиме подтверждения **ACK** отправляются незамедлительно, в при обычном режиме работы TCP передача подтверждений может задерживаться. Данная опция не является флагом использования режима **quickack** - она лишь меняет состояние этого флага.
TCP_SYNCNT	Задает число повторов передачи пакетов **SYN** до того, как будет принято решение о невозможности организации соединения. Число таких попыток не может превышать 255.

12.6.4 Операции IOCTL

Операции ioctl для протокола TCP доступны с использованием функции **ioctl** (Приложение 12.13):

```
ioctl(tcp_socket, ioctl_type, &value);
```

Таблица 124 Операции IOCTL для протокола TCP

Параметр	Описание
SIOCINQ	Возвращает размер непрочитанных данных в очереди приемного буфера. При вызове для сокета, находящегося в состоянии LISTEN функция возвращает код ошибки **EINVAL**.
SIOCATMARK	Функция возвращает значение TRUE, если все срочные данные уже прочитаны пользовательской программой. Этот параметр используется вместе с опцией **SO_OOBINLINE**.
SIOCOUTQ	Возвращает размер неотправленных данных в очереди буфера передачи. При вызове для сокета, находящегося в состоянии LISTEN функция возвращает код ошибки **EINVAL**.

12.6.5 Обработка сетевых ошибок

При возникновении ошибки в сети модуль TCP пытается повторить передачу пакета. Если в течение заданного времени такие попытки не увенчаются успехом, выдается сообщение об ошибке ETIMEDOUT или последнее сообщение об ошибке для данного соединения.

Некоторым приложениям требуется получать уведомлению об ошибках быстрее, нежели обычно происходит в TCP. Для предоставления такой возможности служит опция IP **IP_RECVERR** (стр. 382). При включенной опции все входящие сообщения об ошибках незамедлительно передаются пользовательской программе. Эту опцию следует использовать с осторожностью, поскольку она снижает устойчивость TCP к изменению маршрутов и другим достаточно часто встречающимся состояниям сети.

При возникновении ошибок, требующих повторной организации соединения сигнал **SIGPIPE** передается только в случаях наличия у сокета флага **SO_KEEPALIVE**.

Протокол не использует данных, передаваемых по отдельным каналам (out-of-band data), вместо этого протокол поддерживает концепцию срочных данных (urgent). В реализации Linux это означает, что при передаче удаленной стороной новых срочных данных более старые данные с флагом **urgent** помещаются в поток как обычные данные (даже при отсутствии флага **SO_OOBINLINE**). Стек TCP в системах BSD использует иной подход.

Интерпретация флага важности (urgent pointer) в Linux по умолчанию совместима с системами BSD и не соответствует требованиям RFC 1122. Такое решение было принято для обеспечения совместимости с другими реализациями стека протоколов. Изменить интерпретацию флага важности можно с помощью переменной SystCtl **tcp_stdurg** (стр. 369).

12.6.6 Коды ошибок TCP

Таблица 125 Коды ошибок протокола IP

Код	Описание
ETIMEDOUT	Удаленная сторона не подтвердила прием переданных повторно данных в течение заданного времени.
EAFNOTSUPPORT	Переданный в **sin_family** тип адреса сокета не совпадает с **AF_INET**.

Для протокола TCP также используются все коды ошибок, определенные для протокола IP (стр. 384).

12.6.7 Известные проблемы

1) Описаны не все коды ошибок.
2) Не описана реализация протокола для IPv6.

12.7 Реализация протокола UDP в Linux

Протокол UDP (User Datagram Protocol) обеспечивает передачу дейтаграмм в сетях IP. Реализация UDP в Linux соответствует спецификации RFC 768. Протокол обеспечивает передачу дейтаграмм без организации прямых соединений и без гарантии доставки пакетов. При передаче пакетов может нарушаться порядок доставки, некоторые пакеты могут теряться, а некоторые дублироваться. Для обнаружения ошибок при передаче пакетов протокол UDP использует контрольные суммы в заголовках пакетов.

При создании сокета UDP адреса (локальный и удаленный) не указываются. Дейтаграммы можно передавать сразу же после создания сокета с помощью функций **sendto** и **sendmsg**, которым в качестве параметра передается корректный адрес получателя. При вызове для сокета функции **connect** устанавливается принятый по умолчанию адрес получателя и дейтаграммы можно передавать с помощью функций **send** и **write** без указания адреса получателя. Если дейтаграммы нужно передавать по другим адресам, можно использовать функции **sendto** и **sendmsg**. Для приема входящих пакетов сокет должен быть связан с локальным адресом с помощью функции **bind**. Если функция вызвана без локального адреса, сокет связывается с первым свободным портом из числа указанных в **net.ipv4.ip_local_port_range** (параграф 12.3.1.8 на стр. 372) и адресом **INADDR_ANY**.

Все операции приема возвращают только один пакет. Когда размер пакета меньше размера буфера, возвращается только часть буфера, содержащая пакет. Если пакет превышает размер буфера, часть пакета отсекается с установкой для этого пакета флага **MSG_TRUNC**. Режим **MSG_WAITALL** реализацией протокола не поддерживается.

Опции IP могут передаваться или приниматься с использованием стандартных способов, описанных для протокола **IP** (параграф 12.5.2 на стр. 380). Для обработки опций ядром требуется наличие соответствующих флагов SysCtl, но пользовательским программам опции будут передаваться в любом случае.

При установленном флаге **MSG_DONTROUTE** для передаваемых дейтаграмм адрес получателя должен быть связан с локальным интерфейсом и пакет может передаваться только через этот интерфейс.

UDP фрагментирует пакеты, когда общий размер превышает значение MTU[1] для интерфейса. Более эффективным способом является определение MTU для маршрута доставки с помощью опции **IP_PMTU_DISCOVER** (стр. 383).

12.7.1 Формат адреса

Протокол UDP использует формат адресов Ipv4, передаваемых в структурах **sockaddr_in** (стр. 380).

12.7.2 Обработка ошибок

Сообщения о всех критических ошибках будут передаваться пользователю с возвратом кода ошибки даже в тех случаях, когда сокет не подключен. К числу критических ошибок относятся и асинхронные ошибки в сети. Приложение может также получать сообщения об ошибках для пакетов, которые были ранее отправлены в сеть с использованием данного сокета. Такое поведение отличается от большинства реализаций BSD, которые никогда не передают ошибок для неподключенных сокетов. Реализация Linux соответствует требованиям RFC 1122.

Для обеспечения совместимости со старыми версиями можно установить опцию сокета **SO_BSDCOMPAT** (стр.

1 *Maximum Transmission Unit - максимальный передаваемый блок данных.*

377), обеспечивающую получение сообщений об ошибках[1] только при подключенном сокете. Однако более эффективным решением будет замена старого кода, не обеспечивающего корректную обработку ошибок. Сообщения о локальных ошибках передаются в любом случае.

При включенной опции **IP_RECVERR** (стр. 382) все сообщения об ошибках сохраняются в очереди сокета и могут быть прочитаны с помощью функции **recvmsg**, которой передается флаг **MSG_ERRQUEUE**.

12.7.3 Операции IOCTL

Функция **ioctl** позволяет определить размер ожидающей дейтаграммы и данных в очереди передачи в байтах.

Таблица 126 Операции IOCTL для протокола UDP

Тип	Описание
`SIOCINQ`	Возвращает указатель на целое число, определяющее размер ожидающей обработки дейтаграммы в байтах.
`SIOCOUTQ`	Возвращает указатель на целое число, показывающее объем данных (в байтах) в локальной очереди передачи. Поддерживается, начиная с ядра версии 2.4.

Кроме того, поддерживаются все операции IOCTL, определенные для IP (параграф 12.5.4 на стр. 384) и socket (параграф 12.4.5 на стр. 379).

12.7.4 Коды ошибок

Модуль UDP может возвращать все коды ошибок, определенные для IP (параграф на стр.) и socket (параграф на стр.).

Таблица 127 Сообщения об ошибках для протокола UDP

Ошибка	Описание
`ECONNREFUSED`	С указанным адресом не связано никакого получателя. Причина такого сообщения может быть связана с пакетом, переданным через данный сокет ранее.

12.8 RAW-сокет

Реализация протокола IPv4 в ОС Linux поддерживает беспротокольные сокеты **raw** (SOCK_RAW). Эти сокеты позволяют реализовать в пользовательском пространстве новые протоколы стека Ipv4. Сокеты **raw** принимают и передают необработанные дейтаграммы без добавления в них каких-либо заголовков.

Уровень IPv4 генерирует заголовки IP для передаваемых пакетов, если для сокета не установлен флаг **IP_HDRINCL** (стр. 381), означающий, что пользовательская программа уже включила в данный пакет свой заголовок.

Беспротокольные сокеты могут использоваться только процессами имеющими флаг возможностей **CAP_NET_RAW** или запущенными от имени пользователя с эффективным UID=0 (root).

Сокету передаются все пакеты и сообщения об ошибках, соответствующие номеру протокола, заданному для этого сокета. Списки номеров протоколов можно найти в документе RFC 1700[2] или получить с помощью системного вызова **getprotobyname**.

Протокол **IPPROTO_RAW** предполагает наличие флага **IP_HDRINCL** и может передавать любые протоколы IP, указанные в полученном заголовке. Прием пакетов IP с использованием беспротокольных сокетов через **IPPROTO_RAW** невозможен ни для каких протоколов IP.

При установленном флаге **IP_HDRINCL** некоторые поля заголовков IP изменяются в соответствии с приведенной ниже таблицей.

Таблица 128 Изменение полей заголовка IP для пакетов RAW

Поле	Изменение
`IP Checksum`	Заполняется в соответствии с реальной контрольной суммой.
`Source Address`	Устанавливается нулевое значение.
`Packet Id`	Устанавливается нулевое значение.
`Total Length`	Заполняется в соответствии с реальным размером пакета.

Если задан флаг **IP_HDRINCL** и заголовок IP содержит ненулевой адрес получателя, тогда для маршрутизации пакета используется адрес получателя, заданный для сокета. При наличии флага **MSG_DONTROUTE** адрес получателя должен быть связан с локальным интерфейсом, иначе интерфейс будет определяться из таблицы маршрутов (при этом маршруты через шлюзы игнорируются).

Если флаг **IP_HDRINCL** не установлен, можно задать опции IP с использованием функции **setsockopt** (см. Приложение 12.5).

В Linux все поля и опции заголовков IP можно устанавливать с использованием опций сокета IP. Это означает, что

1 *Сообщения **EPROTO** и **EMSGSIZE** будут приниматься независимо от этой опции.*
2 *Этот документ изрядно устарел и более не будет обновляться, поэтому лучше обращаться к списку зарегистрированных протоколов на сайте www.iana.org/assignments/protocol-numbers*

сокеты типа **raw** обычно нужны только для новых протоколов или протоколов, не имеющих пользовательского интерфейса (например, ICMP).

При получении пакета всем сокетам **raw**, которые связаны с указанным в пакете протоколом, до передачи какому-либо иному протокольному модулю (например, модулю ядра).

12.8.1 Формат адреса

Сокеты **raw** используют для адресации стандартные структуры **sockaddr**, определенные для протокола IP (стр. 380). Поле **sin_port** можно использовать для указания номера протокола IP[1]. Во входящих пакетов в поле **sin_port** содержит номер протокола для данного пакета. Корректные номера протоколов IP можно найти в файле **<netinet/in.h>**.

12.8.2 Опции сокета

Опции беспротокольных сокетов можно устанавливать с помощью функции **setsockopt** и читать с помощью **getsockopt**, передавая этим функциям флаг семейства **SOL_RAW**.

Сокеты **raw** поддерживают все опции IP (семейство **SOL_IP**, см. стр. 380), которые пригодны для сокетов дейтаграмм. Кроме того, для беспротокольных сокетов поддерживается специальная опция **ICMP_FILTER**, которая включает специальный фильтр для raw-сокетов, связанных с протоколом **IPPROTO_ICMP**. Этот флаг устанавливается для всех типов сообщений ICMP, которые должны быть отфильтрованы. По умолчанию фильтрация ICMP отключена.

12.8.3 Замечания по использованию raw-сокетов

Сокеты **raw** фрагментируют пакеты, размер которых превышает значение MTU для интерфейса (см. стр. 390). Более эффективным решением является использование механизма определения MTU (Path MTU discovery) с помощью опции **IP_PMTU_DISCOVER** (стр. 383).

Сокет **raw** можно связать с локальным адресом, используя функцию **bind**. Если сокет не привязан к адресу, он будет получать все пакеты, заданного для сокета протокола IP. Беспротокольные сокеты можно так же привязать к указанному сетевому интерфейсу с помощью опции **SO_BINDTODEVICE** (см. стр 377).

Сокет **IPPROTO_RAW** предназначен только для передачи пакетов. Если вы реально хотите принимать все пакеты IP, используйте сокет **packet** (Приложение 12.9) с протоколом **ETH_P_IP**. Отметим, что пакетные сокеты не собирают фрагментированные дейтаграммы IP в отличие от сокетов **raw**.

Если вы хотите принимать все пакеты ICMP для сокета дейтаграмм, зачастую будет удобней воспользоваться опцией **IP_RECVERR** (см. стр. 382) для конкретного сокета.

Сокеты **raw** могут перехватывать пакеты всех протоколов IP в системе Linux, включая протоколы типа ICMP или TCP, имеющие в ядре собственные протокольные модули. В таких случаях пакеты передаются сразу модулю ядра и беспротокольному сокету. Такие возможности не следует использовать в переносимых программах, поскольку другие ОС (в частности, BSD) не могут работать в таком режиме.

Linux никогда не изменяет заголовки, передаваемые от пользователя, за исключением заполнения некоторых полей при использовании опции IP_HDRINCL (см. табл. 128). Большинство реализаций беспротокольных сокетов в других ОС, изменяют пользовательские заголовки.

Сокеты **raw** в большинстве случаев плохо переносимы и их не следует применять в программах, предназначенных для других ОС.

12.8.4 Обработка ошибок

Приходящие из сети сообщения об ошибках передаются пользовательской программе только в тех случаях, когда сокет подключен или установлен флаг **IP_RECVERR**. Для подключенных сокетов в целях обеспечения совместимости передаются только сообщения **EMSGSIZE** и **EPROTO**. При установке флага **IP_RECVERR** все сетевые ошибки помещаются в очередь.

12.8.5 Коды ошибок

Таблица 129 Коды ошибок для сокетов raw

Код	Описание
EMSGSIZE	Размер пакета слишком велик. Такая ошибка может возникать при включенном режиме Path MTU Discovery (флаг IP_PMTU_DISCOVER - см. стр. 383), а также в тех случаях, когда размер пакета превышает максимальное значение для IPv4 (64 кбайт).
EACCES	Попытка пользователя передать пакет по широковещательному адресу при отсутствии у сокета флага широковещания.
EPROTO	Поступило сообщение ICMP об ошибке, вызванной некорректными параметрами.
EFAULT	Передан некорректный адрес блока памяти.
EOPNOTSUPP	При вызове функции **socket** передан некорректный флаг (например, MSG_OOB).

1 *Это поле игнорируется в ядрах серии 2.2 и для него следует использовать значение 0 (см. стр. 390).*

Код	Описание
EINVAL	Некорректный параметр.
EPERM	Пользователь не имеет права открывать raw-сокеты. Такие права имеет пользователь с эффективным идентификатором UID=0 и приложения с установленным флагом возможностей **CAP_NET_RAW**.

12.8.6 Известные проблемы

1) Не описано расширение для прозрачного proxy.
2) При установке опции **IP_HDRINCL** дейтаграммы не будут фрагментироваться, следовательно их размер ограничен значением MTU для интерфейса. Это ограничение присуще ядрам серии 2.2.
3) Начиная с ядрес серии 2.2 установка номера протокола IP в поле **sin_port** не поддерживается. Всегда используется протокол, с которым сокет связан, или протокол, заданный при первом вызове функции **socket**.

12.9 Пакетный сокет в Linux

Сокет **packet** (тип **PF_PACKET**) используется для приема и передачи необработанных (raw) пакетов на канальном уровне (OSI Layer 2). Интерфейс сокета позволяет реализовать в пользовательском пространстве модули протоколов, использующих только сервис канального уровня.

В качестве типа сокета может указываться значение **SOCK_RAW** (необработанные пакеты, включающие заголовок канального уровня) или **SOCK_DGRAM** (обработанные пакеты с удаленным заголовком канального уровня). Заголовки канального уровня представляются в виде структуры **sockaddr_ll**, описанной ниже.

Поле **sll_protocol** содержит номер протокола IEEE 802.3[1] (значение **ETH_P_ALL** соответствует всем протоколам Ethernet). Все входящие пакеты заданного в адресной структуре протокола, будут передаваться пакетному сокету до их передачи протокольным модулям ядра.

Для того, чтобы открыть пакетный сокет приложение должно иметь флаг возможности **CAP_NET_RAW** или исполняться от имени пользователя с эффективным идентификатором UID=0.

Пакеты **SOCK_RAW** передаются и принимаются от драйвера устройства без каких-либо изменений в данных. При получении пакета адресная информация разбирается и передается в структуру **sockaddr_ll**. При передаче пакета полученный от пользователя буфер должен содержать заголовок канального уровня. Такой пакет передается без изменений сетевому драйверу интерфейса, указанного в поле адреса получателя.

Обработка пакетов **SOCK_DGRAM** происходит на более высоком уровне. Заголовок канального уровня удаляется из пакета до передачи этого пакета пользователю. Передаваемые через пакетный сокет пакеты **SOCK_DGRAM** получают подходящий заголовок канального уровня в структуре **sockaddr_ll** до их размещения в очереди.

По умолчанию все пакеты заданного типа протокола передаются пакетному сокету. Для того, чтобы получать только пакеты от интересующего интерфейса, следует использовать функцию **bind**, передав ей в качестве параметра структуру **sockaddr_ll** для соответствующего интерфейса, чтобы связать сокет с этим интерфейсом. В этом случае из адресной структуры используются лишь поля **sll_protocol** и **sll_ifindex**.

Операции подключения (connect) не поддерживаются для пакетного сокета.

Если при вызове функций **recvmsg**, **recv**, **recvfrom** был установлен флаг **MSG_TRUNC**, функции всегда будут возвращать реальный размер кадра в среде, даже если он отличается от размера кадра в буфере (заполнение для выравнивания).

Для переносимых программ разумно использовать сокеты **PF_PACKET** через библиотеку **pcap** (параграф 11.9.1), хотя эта библиотека не перекрывает всех возможностей пакетных сокетов.

Пакетные сокеты **SOCK_DGRAM** не пытаются создавать или разбирать заголовки IEEE 802.2 LLC для кадров IEEE 802.3. При выборе для передачи в качестве протокола **ETH_P_802_3** ядро создает кадр 802.3 и заполняет поле размера. Пользовательская программа должна предоставить заголовок LLC для создания законченного пакета. Входящие пакеты 802.3 не демультиплексируются по полям протокола DSAP/SSAP - они просто передаются пользовательскому приложению как пакеты протокола **ETH_P_802_2** с готовым заголовком LLC. Таким образом, привязка к протоколу **ETH_P_802_3** невозможна и приложению следует использовать привязку к протоколу **ETH_P_802_2** и самостоятельно выполнять демультиплексирование. По умолчанию для передаваемых пакетов используется инкапсуляция Ethernet DIX.

Пакетные сокеты никак не связаны с цепочками правил межсетевых экранов.

12.9.1 Типы адресов

Для адресации пакетный сокет использует независимую от адресов физического уровня структуру **sockaddr_ll**.

```
struct sockaddr_ll {
    unsigned short  sll_family;    /* AF_PACKET */
    unsigned short  sll_protocol;  /* протокол канального уровня */
    int             sll_ifindex;   /* номер интерфейса */
    unsigned short  sll_hatype;    /* тип заголовка */
    unsigned char   sll_pkttype;   /* тип пакета */
```

1 *Поддерживаемые номера протоколов перечислены в файле* ***<linux/if_ether.h>***

```
    unsigned char   sll_halen;     /* размер адреса */
    unsigned char   sll_addr[8];   /* аппаратный адрес */
};
```

Поле **sll_protocol** содержит идентификатор стандартного типа протокола Ethernet (см. файл **linux/if_ether.h**). Поле **sll_ifindex** содержит индекс интерфейса[1]; **sll_hatype** задает тип ARP[2]. Поле **sll_pkttype** указывает тип адресации пакета; допустимыми типами являются **PACKET_HOST** (пакеты, адресованные локальному хосту), **PACKET_BROADCAST** (широковещательные пакеты канального уровня), **PACKET_MULTICAST** (групповая адресация на канальном уровне), **PACKET_OTHERHOST** (пакет для другого хоста, который может быть захвачен интерфейсом, работающим в режиме **promiscuous**), и **PACKET_OUTGOING** (для пакетов от локального хоста, возвращенных пакетному сокету). Перечисленные типы адресации имеют смысл только для принимаемых пакетов. Поля **sll_addr и sll_halen** содержат аппаратный адрес (например, IEEE 802.3) и его размер. Интерпретация этих полей зависит от конкретного устройства.

При передаче пакетов достаточно задать значения **sll_family=AF_PACKET**, **sll_addr**, **sll_halen** и **sll_ifindex**. Остальные поля должны иметь значение 0. Поля **sll_hatype** и **sll_pkttype** устанавливаются в принимаемых пакетах только для информации. Для связывания сокетов используются только поля **sll_protocol** и **sll_ifindex**.

12.9.2 Опции сокета

Пакетные сокеты могут использоваться для групповой рассылки на канальном уровне и работы интерфейсов в режиме захвата. Опции сокета задаются с помощью функции **setsockopt** для **SOL_PACKET**. Опция **PACKET_ADD_MEMBERSHIP** добавляет привязку, а опция **PACKET_DROP_MEMBERSHIP** удаляет ее. В обоих случаях в качестве аргумента передается структура:

```
struct packet_mreq
{
     int             mr_ifindex;    /* индекс интерфейса */
     unsigned short  mr_type;       /* действие */
     unsigned short  mr_alen;       /* размер адреса */
     unsigned char   mr_address[8]; /* аппаратный адрес */
};
```

Поле **mr_ifindex** содержит индекс интерфейса, для которого нужно изменить состояние, **mr_type** задает выполняемое действие (**PACKET_MR_PROMISC** разрешает прием из среды всех пакетов - режим захвата, **PACKET_MR_MULTICAST** привязывает сокет к multicast-группе канального уровня, а **PACKET_MR_ALLMULTI** разрешает сокету принимать все пакеты с групповыми адресами, поступающие в данный интерфейс).

Для решения этих задач можно использовать также стандартные операции IOCTL **SIOCSIFFLAGS**, **SIOCADDMULTI**, **SIOCDELMULTI**.

12.9.3 Операции IOCTL

Для получения временной метки последнего доставленного пакета может использоваться **SIOCGSTAMP** со структурой **timeval** в качестве аргумента.

Кроме того для пакетных сокетов поддерживаются все операции IOCTL, определенные для **netdevice** (параграф 12.12) и **socket** (параграф 12.4.5 на стр. 379).

12.9.4 Обработка ошибок

Пакетные сокеты не выполняют обработки каких-либо ошибок за исключением тех, которые происходят при передаче пакетов драйверу устройства.

12.9.5 Коды ошибок

Таблица 130 Коды ошибок для пакетных сокетов

Код	Описание
ENETDOWN	Интерфейс неактивен.
ENOTCONN	Функции не передан адрес интерфейса.
ENODEV	Задано неизвестное имя устройства или индекс интерфейса.
EMSGSIZE	Размер пакета превышает значение MTU для интерфейса.
ENOBUFS	Недостаточно памяти для размещения пакета.
EFAULT	Пользователь передал некорректный адрес памяти.
EINVAL	Некорректный аргумент
ENXIO	В адресе содержится некорректный индекс интерфейса.
EPERM	Пользователь не имеет прав на выполнение операции.

1 *Значению 0 соответствуют все интерфейсы системы - это значение можно использовать только для привязки сокетов*

2 *Допустимые идентификаторы типов перечислены в файле* ***<linux/if_arp.h>***

Код	Описание
EADDRNOTAVAIL	Передан неизвестный адрес multicast-группы.
ENOENT	Пакет не был получен

Драйверы устройств могут генерировать дополнительные коды ошибок.

12.9.6 Известные проблемы

1) Обработка IEEE 802.2/803.3 LLC может трактоваться как ошибка.
2) Отсутствует документация для фильтров сокета.
3) Расширение **MSG_TRUNC** для функции **recvmsg** является слишком опасным - вместо него следует использовать управляющее сообщение.
4) Отсутствует способ определения исходного адреса получателя пакетов **SOCK_DGRAM**.

12.10 Протокол netlink в Linux

Протокол **netlink** (**PF_NETLINK**) обеспечивает обмен информацией между модулями ядра и программами пользовательского пространства. Протокол использует стандартный интерфейс для пользовательских программ и внутренний интерфейс API для модулей ядра. Ниже будет рассмотрен только пользовательский интерфейс сокета, а интерфейс ядра и старый вариант пользовательского интерфейса для работы через символьные устройства **netlink** не обсуждается.

Netlink обеспечивает для приложений сервис передачи дейтаграмм. Для **netlink** допустимо указывать тип сокета **SOCK_RAW** (Приложение 12.8) и **SOCK_DGRAM** (Приложение 12.7) и протокол не делает между ними различий.

Отметим, что использовать функции протокола **netlink** зачастую проще через библиотеку **libnetlink**, нежели напрямую через интерфейс с ядром.

Включаемый файл **netlink.h** содержит определения нескольких стандартных макросов для доступа или создания дейтаграмм **netlink**. Для доступа к буферам, передаваемым и принимаемым сокетом **netlink**, следует использовать только макросы из этого файла, которые кратко описаны в таблице 131.

```
#include <asm/types.h>
#include <linux/netlink.h>
int NLMSG_ALIGN(size_t len);
int NLMSG_LENGTH(size_t len);
int NLMSG_SPACE(size_t len);
void *NLMSG_DATA(struct nlmsghdr *nlh);
struct nlmsghdr *NLMSG_NEXT(struct nlmsghdr *nlh, int len);
int NLMSG_OK(struct nlmsghdr *nlh, int len);
int NLMSG_PAYLOAD(struct nlmsghdr *nlh, int len);
```

Таблица 131 Макросы netlink

Имя	Описание
NLMSG_ALIGN	Округляет размер сообщения **netlink** до ближайшего большего значения, выровненного по границе.
NLMSG_LENGTH	Принимает в качестве параметра размер поля данных (payload) и возвращает выровненное по границе значение размера для записи в поле **nlmsg_len** заголовка **nlmsghdr**.
NLMSG_SPACE	Возвращает размер, который займут данные указанной длины в пакете **netlink**.
NLMSG_DATA	Возвращает указатель на данные, связанные с переданным заголовком **nlmsghdr**.
NLMSG_NEXT	Возвращает следующую часть сообщения, состоящего из множества частей. Макрос принимает следующий заголовок **nlmsghdr** в сообщении, состоящем из множества частей. Вызывающее приложение должно проверить наличие в текущем заголовке **nlmsghdr** флага **NLMSG_DONE** - функция не возвращает значение NULL при завершении обработки сообщения. Второй параметр задает размер оставшейся части буфера сообщения. Макрос уменьшает это значение на размер заголовка сообщения.
NLMSG_OK	Возвращает значение TRUE (1), если сообщение не было усечено и его разборка прошла успешно.
NLMSG_PAYLOAD	Возвращает размер данных (payload), связанных с заголовком **nlmsghdr**.

12.10.1 Семейство netlink

Семейство **netlink_family** выбирает модуль ядра или группу **netlink** для обмена информацией. Члены семейства перечислены в таблице 132.

Таблица 132 Члены семейства netlink

Имя	Описание
NETLINK_ROUTE	Принимает обновления маршрутов и может использоваться для модификации маршрутной таблицы IPv4.
NETLINK_SKIP	Зарезервирован для ENskip.
NETLINK_USERSOCK	Зарезервирован для новых протоколов пользовательского пространства.
NETLINK_FIREWALL	Принимает пакеты от межсетевых экранов IPv4.
NETLINK_TCPDIAG	Мониторинг сокета TCP.
NETLINK_NFLOG	Операция ULOG (параграф 5.1.8.2.22 на стр. 116).
NETLINK_XFRM	IPsec.
NETLINK_ARPD	Используется для управления таблицами **arp** в пользовательском пространстве.
NETLINK_ROUTE6	Принимает и передает обновления таблицы маршрутов IPv6 (af_inet6).
NETLINK_IP6_FW	Служит для приема сообщений о неудачном результате проверки правил на брандмауэре IPv6 (пока не реализован).
NETLINK_DNRTMSG	Маршрутные сообщения DECnet.
NETLINK_TAPBASE	Экземпляры фиктивного устройства **ethertap**, позволяющее имитировать драйвер Ethernet из пользовательского пространства.
...	
NETLINK_TAPBASE+15	

Сообщения **netlink** представляют собой поток байтов с одним или несколькими заголовками **nlmsghdr** и связанными с ними данными (payload). В сообщениях, состоящих из множества частей все заголовки, за исключением последнего содержат флаг **NLM_F_MULTI**, а в заголовке последнего сообщения установлен флаг **NLMSG_DONE**. Для доступа к байтовым потокам следует использовать только макросы **NLMSG_***.

Протокол **netlink** не обеспечивает гарантированной доставки сообщений, пытаясь лишь приложить все разумные усилия для доставки сообщения адресату. При нехватке памяти или возникновении иных ошибок протокол может отбрасывать пакеты. Для обеспечения гарантированной доставки отправитель может запрашивать у получателя подтверждение, устанавливая в заголовке флаг **NLM_F_ACK**. В качестве подтверждений используются пакеты **NLMSG_ERROR** с кодом ошибки 0. Функции генерации подтверждений должно обеспечивать пользовательское приложение. Ядро пытается передавать сообщения **NLMSG_ERROR** для каждого поврежденного пакета. Пользовательским программам следует придерживаться такой же практики.

Каждое семейство **netlink** имеет свой набор из 32 multicast-групп. При вызове для сокета функции **bind** поле **nl_groups** в структуре **sockaddr_nl** должно содержать битовую маску групп, которым следует слышать сообщение. По умолчанию для этого поля установлено нулевое значение, которое отключает групповую передачу сообщений. Сокет может передавать групповые сообщения любым группам, установив в поле **nl_groups** битовую маску нужных групп перед вызовом функции **sendmsg** или **connect**. Возможность работы (приема или передачи) с групповыми сообщениями **netlink** имеют лишь приложения с флагом возможностей **CAP_NET_ADMIN** и программы, запущенные пользователем с эффективным идентификатором UID=0. Все отклики на групповые сообщения должны передаваться процессу-отправителю и членам группы.

Структура заголовка сообщений **netlink** показана ниже

```
struct nlmsghdr
{
    __u32    nlmsg_len;  /* размер сообщения с учетом заголовка */
    __u16    nlmsg_type; /* тип сообщения (содержимое) */
    __u16    nlmsg_flags;/* стандартные и дополнительные флаги */
    __u32    nlmsg_seq;  /* порядковый номер */
    __u32    nlmsg_pid;  /* Идентификатор процесса (PID), открывшего сокет */
};
```

Сообщения об ошибках имеют структуру:

```
struct nlmsgerr
{
    int      error;      /* отрицательное значение кода ошибки или 0 для подтверждений */
    struct nlmsghdr msg; /* заголовок сообщения, связанного с ошибкой */
};
```

После каждого заголовка **nlmsghdr** размещаются данные, указанного параметром **nlmsg_type** типа:

- **NLMSG_NOOP** - пустое сообщение (игнорируется);
- **NLMSG_ERROR** - сообщение об ошибке, содержащее в поле данных структуру **nlmsgerr**;
- **NLMSG_DONE** - последняя часть сообщения.

Члены семейства **netlink** могут поддерживать дополнительные типы сообщений, описанные с соответствующих страницах руководства и доступных с помощью команды **man** (например, **man 7 rtnetlink** для **NETLINK_ROUTE**).

Флаги сообщений **netlink**, передаваемые в поле **nlmsg_flags**, перечислены в таблице .

Таблица 133. Флаги сообщений netlink.

Флаг	Описание
Основные флаги	
NLM_F_REQUEST	Устанавливается для всех запросов.
NLM_F_MULTI	Сообщение является частью составного сообщения, завершающегося флагом **NLMSG_DONE**.
NLM_F_ACK	Отклик с подтверждением успеха.
NLM_F_ECHO	Запрос эхо-отклика.
Дополнительные флаги для запросов GET	
NLM_F_ROOT	Задает корень дерева
NLM_F_MATCH	Возвращает все найденные соответствия.
NLM_F_ATOMIC	Atomic GET. Этот флаг требует полномочий пользователя с эффективным UID=0 или наличия флага возможностей **CAP_NET_ADMIN**.
NLM_F_DUMP	**NLM_F_ROOT|NLM_F_MATCH**
Дополнительные флаги для запросов NEW	
NLM_F_REPLACE	Записать взамен существующего объекта.
NLM_F_EXCL	Не переписывать, если объект уже существует.
NLM_F_CREATE	Создать объект, если он еще не создан.
NLM_F_APPEND	Добавить в конце списка объектов.

12.10.2 Форматы адресов

Адреса **netlink** для пользовательских программ и модулей ядра описываются структурой **sockaddr_nl**. Заданный такой структурой адрес может быть индивидуальным (unicast) или групповым.

```
struct sockaddr_nl
{
    sa_family_t nl_family;    /* AF_NETLINK */
    unsigned short nl_pad;    /* заполнение нулями */
    pid_t       nl_pid;       /* идентификатор процесса */
    __u32       nl_groups;    /* маска групп */
};
```

Поле **nl_pid** содержит идентификатор процесса, владеющего сокетом-адресатом или 0, если сообщение адресовано ядру. Параметр **nl_groups** содержит маску, каждый бит которой представляет одну из групп **netlink**.

12.11 Реализация ARP в Linux

Функции протокола ARP[1] реализованы в ядре Linux в соответствии со спецификацией RFC 826. Протокол используется для преобразования между аппаратными адресами канального уровня и сетевыми адресами IPv4 для подключенных непосредственно сетей. Пользователь обычно не взаимодействует с реализующим протокол модулем ядра за исключением этапа настройки конфигурации при компиляции ядра (см. главу 4.4). Модуль протокола обеспечивает сервисные функции для других протоколов ядра.

Пользовательские процессы могут получать пакеты ARP с помощью сокета **packet** (параграф 12.9 на стр. 390). Существует также механизм управления кэшем ARP в пользовательском пространстве с помощью сокетов netlink (параграф 12.10 на стр. 392). Контроль таблиц ARP возможен также с использованием операций **ioctl** и любого сокета **PF_INET**. В пакет nettools входит утилита **arp** (параграф 11.1.2.1 на стр. 193), обеспечивающая возможность просмотра и изменения таблицы ARP.

Модуль ARP поддерживает кэш отображений между аппаратными адресами и адресами сетевого уровня. Размер кэша ограничен, поэтому старые и редко используемые записи удаляются из таблицы системой сбора мусора. Записи, помеченные как постоянные (permanent) системой сбора мусора из таблицы не удаляются. Поддерживаются возможности прямого управления записями в таблице с использованием описанных ниже операций **ioctl** (параграф 12.11.1) с учетом параметров **sysctl** (параграф 12.11.2 на стр. 395).

По истечении некоторого времени (в зависимости от параметров **sysctl**) запись таблицы, которая не обновлялась[2], будет рассматриваться как устаревшая. В таких случаях ARP сначала пытается **app_solicit** раз обратиться к локальному демону **arp** для получения обновленного MAC-адреса. Еслт эти попытки не увенчались удачей и в таблице присутствует старый MAC-адрес, по этому адресу **ucast_solicit** раз передается тестовый запрос (unicast probe). Если и это не приведет к успеху в сеть передается новый широковещательный запрос ARP (при условии наличия данных для отправки по соответствующему адресу сетевого уровня).

Linux будет автоматически добавлять временные (non-permanent) записи в таблицу **proxy arp** при получении

1 *Address Resolution Protocol - протокол преобразования адресов, обеспечивающий возможность определения MAC-адреса устройства Ethernet по сетевому адресу интерфейса.*
2 *В качестве обновлений записи рассматривается информация от протоколов вышележащих уровней (например, пакеты TCP ACK или сигналы, переданные с использованием системной функции **sendmsg** и содержащие флаг **MSG_CONFIRM**).*

запросов для адресов, по которым хост пересылает кадры, если функции **proxy arp** разрешены для принимающего интерфейса. Если путь к получателю неизвестен запись в таблицу **proxy arp** не включается.

12.11.1 Операции IOCTL

Вызовы **ioctl** доступны для всех сокетов **PF_INET**. Сокет принимает в качестве параметра структуру **arpreq**.

```
struct arpreq
{
    struct sockaddr arp_pa;      /* протокольный адрес */
    struct sockaddr arp_ha;      /* аппаратный адрес */
    int             arp_flags;   /* флаги */
    struct sockaddr arp_netmask; /* маска протокольного адреса */
    char            arp_dev[16];
};
```

Функции **SIOCSARP**, **SIOCDARP** и **SIOCGARP** устанавливают, удаляют и читают отображение ARP, соответственно. Установка и удаление записей ARP являются привилегированными операциями и могут выполняться только процессами с флагом возможностей CAP_NET_ADMIN или выполняющимися от имени пользователя с UID = 0.

Параметр **arp_pa** должен содержать сокет **AF_INET**, и тип адреса **arp_ha** должен совпадать с типом устройства, заданным в **arp_dev**. Строка **arp_dev** содержит имя устройства в формате языка C (строка, завершающаяся 0. Флаги ARP перечислены в таблице 134.

При установке флага **ATF_NETMASK** параметр **arp_netmask** должен содержать корректную маску подсети. Ядро Linux 2.2 не поддерживает записи **proxy arp**. Флаг **ATF_USETRAILERS** является устаревшим и не должен использоваться.

Таблица 134 Флаги ARP

Флаг	Значение
ATF_COM	Просмотр завершен
ATF_PERM	Постоянная запись
ATF_PUBL	Публикуемая запись
ATF_USETRAILERS	Запрошены трейлеры
ATF_NETMASK	Использовать маску
ATF_DONTPUB	Непубликуемая запись

12.11.2 Параметры SYSCTL

Модуль ARP поддерживает интерфейс **sysctl** для настройки глобальных параметров и параметров отдельного интерфейса. Доступ к параметрам обеспечивается через файлы **/proc/sys/net/ipv4/neigh/*/*** или функции sysctl. Каждый интерфейс системы имеет свой каталог **/proc/sys/net/ipv4/neigh/**. Параметры в файлах каталога **default** используются при создании каталогов для новых устройств. Все временные параметры **sysctl** задаются в секундах, если явно не указано иное.

Таблица 135 Параметры sysctl для ARP

Параметр	Описание	Значение по умолчанию
anycast_delay	Максимальное время задержки отклика на сообщения **IPv6 neighbour**[1].	100 jiffy[2]
app_solicit	Максимальное число проб, передаваемых демону ARP в пользовательском пространстве через сокет **netlink**, прежде, чем перейти к multicast-пробам.	0
base_reachable_time	После обнаружения соседа запись для него считается корректной в течение по крайней мере случайного промежутка времени из диапазона **base_reachable_time/2** - **3*base_reachable_time/2**. Продолжительность периода корректности записи может быть увеличена при получении позитивной обратной связи от протоколов вышележащих уровней.	30 секунд
delay_first_probe_time	Задержка перед отправкой первой пробы с момента принятия решения о том, что запись для соседа устарела.	5 секунд
gc_interval	Период активизации процесса сборки мусора в таблице адресов.	30 секунд
gc_stale_time	Определяет период проверки старения записей. Когда адресная запись сочтена устаревшей, ее нужно обновить до передачи данных по этому адресу.	60 секунд
gc_thresh1	Минимальное число записей, которое должно присутствовать в кэше ARP. Если число записей меньше этого значения, сборщик мусора не будет очищать таблицу.	128
gc_thresh2	Мягкое ограничение числа записей в кэше ARP. При достижении этого порога сборщик мусора будет активизироваться в течение 5 секунд.	512

1 *Поддержка anycast еще не реализована в ядре Linux.*
2 *Специальная единица времени, используемая ядрами Linux. Для процессоров x86 составляет приблизительно 10 мсек.*

Параметр	Описание	Значение по умолчанию
gc_thresh3	Жесткое ограничение числа записей в кэше ARP. При достижении этого порога должна быть незамедлительно выполнена сборка мусора.	1024
locktime	Минимальный интервал времени хранения записи в кэше ARP. Это ограничение предотвращает переполнение кэша ARP при наличии множества потенциальных записей для одного адреса (например, в результате конфигурационных ошибок).	100 jiffy
mcast_solicit	Максимальное число попыток преобразования адреса с помощью проб **multicast/broadcast** прежде, чем будет принято решение о недоступности адреса.	3
proxy_delay	При получении запроса ARP для известного адреса proxy-ARP этот параметр определяет задержку передачи отклика, позволяющую в некоторых случаях предотвратить лавину пакетов (network flooding).	80 jiffy
proxy_qlen	Максимальное число пакетов которые могут быть помещены в очередь proxy-ARP.	64
retrans_time	Задержка повторной передачи запроса.	100 jiffy
ucast_solicit	Максимальное число попыток передачи unicast-проб перед тем, как будет запрошен демон ARP (см. **app_solicit** на стр. 395).	3
unres_qlen	Максимальное число пакетов от других сетевых уровней, которые могут быть помещены в очередь для каждого неизвестного адреса.	3

12.11.3 Известные ограничения

1) Значения некоторых таймеров задаются не в абсолютных единицах, а в jiffy, величина которых зависит от аппаратной платформы. Например в системах Alpha = 1/1024 секунды, а для большинства других платформ = 1/100s.
2) Не существует обратной связи с программами пользовательского пространства. Это означает, что работающие на основе явных соединений (connection oriented) протоколы, реализованные в пользовательском пространстве, будут генерировать избыточный трафик ARP, поскольку **ndisc** будет регулярно повторять пробы MAC-адресов. Такая же проблема характерна для некотрых протоколов, реализованных в ядре (например, NFS на базе UDP).
3) Структура **arpreq** была изменена в ядре Linux версии 2.0 (добавлено поле **arp_dev** и изменена нумерация **ioctl**). В версии ядра 2.2 была прекращена поддержка старых вызовов **ioctl**.
4) Поддержка записей **proxy arp** для сетей (маска адреса не равна 0xffffffff) была прекращена в версии ядра Linux 2.2 с заменой на поддержку ядром proxy arp для всех доступных хостов на других интерфейсах системы (для этого интерфейсы должны поддерживать пересылку кадров и proxy arp).
5) Параметры **sysctls**, включенные в **neigh/***, не поддерживаются ядрами Linux до версии 2.2.

12.12 Интерфейс netdevice

Интерфейс **netdevice** обеспечивает возможность взаимодействия с сетевыми устройствами Linux. Операции IOCTL для интерфейса **netdevice** описаны в заголовочных файлах **<sys/ioctl.h>** и **<net/if.h>**.

С помощью этих операций и параметров **netdevice** обеспечивается возможность настройки конфигурационных параметров сетевых устройств.

ОС Linux поддерживает для настройки конфигурации сетевых устройств стандартные операции IOCTL. Эти операции могут использоваться с файловыми дескрипторами любого сокета, независимо от его семейства и типа. Параметры интерфейсов передаются в виде структур **ifreq** и **ifconf**, описанных в заголовочном файле **<net/if.h>**.

Обычно пользователь задает интерфейс, указывая в поле **ifr_name** имя нужного интерфейса.

12.12.1 Операции IOCTL

Для использования привилегированных операций IOCTL приложение должно иметь флаг возможностей **CAP_NET_ADMIN** или исполняться от имени пользователя с эффективным значением UID=0.

Таблица 136. Операции IOCTL для netdevice.

Имя	Описание
SIOCGIFNAME	Принимая индекс интерфейса **ifr_ifindex**, возвращает имя этого интерфейса в поле **ifr_name**. Это единственная операция IOCTL, возвращающая результат в поле **ifr_name**.
SIOCGIFINDEX	Определяет индекс интерфейса, заданного полем **ifr_ifindex**.
SIOCGIFFLAGS	Возвращает слово флагов (см. таблицу 137). для устройства в поле **ifr_flags**.
SIOCSIFFLAGS	Устанавливает флаги устройства(см. таблицу 137), заданные в поле **ifr_flags**. Установка флагов является привилегированной операцией

Имя	Описание
SIOCGIFMETRIC	Возвращает метрику устройства в поле **ifr_metric**.
SIOCSIFMETRIC	Устанавливает для устройства метрику, переданную в поле **ifr_metric**. Если установка метрики не поддерживается, функция возвращает код ошибки **EOPNOTSUPP**.
SIOCGIFMTU	Возвращает значение MTU[1] для интерфейса в поле **ifr_mtu**.
SIOCSIFMTU	Устанавливает для интерфейса значение MTU, переданное в поле **ifr_mtu**. Установка слишком малого значения MTU может привести к серьезным ошибкам в работе ядра.
SIOCGIFHWADDR	Возвращает аппаратный адрес устройства в поле **ifr_hwaddr**. Аппаратный адрес содержится в структуре данных **sockaddr** - поле **sa_family** содержит идентификатор типа устройства **ARPHRD_***, **sa_data** - адрес канального уровня, начинающийся с байта 0.
SIOCSIFHWADDR	Устанавливает аппаратный адрес устройства, полученный в поле **ifr_hwaddr**. Аппаратный адрес помещается в структуру данных **sockaddr** - поле **sa_family** содержит идентификатор типа устройства **ARPHRD_***, **sa_data** - адрес канального уровня, начинающийся с байта 0. Установка аппаратного адреса является привилегированной операцией.
SIOCSIFHWBROADCAST	Эта привилегированная операция устанавливает для устройства широковещательный адрес, полученный в поле **ifr_hwaddr**.

SIOCGIFMAP	Возвращает аппаратные параметры устройства в поле **ifr_map**, содержащем структуру данных.
	`struct ifmap`
	`{`
	`unsigned long mem_start;`
	`unsigned long mem_end;`
	`unsigned short base_addr;`
	`unsigned char irq;`
	`unsigned char dma;`
	`unsigned char port;`
	`};`
	Интерпретация полей структуры **ifmap** зависит от архитектуры и драйвера устройства.

SIOCSIFMAP	Устанавливает аппаратные параметры устройства, полученные в структуре **ifr_map**. Установка параметров является привилегированной операцией.
SIOCADDMULTI	Добавляет адрес, переданный в поле **ifr_hwaddr**, в multicast-фильтры интерфейса. Операция является привилегированной.
SIOCDELMULTI	Удаляет адрес, переданный в поле **ifr_hwaddr**, из multicast-фильтров интерфейса. Операция является привилегированной.
SIOCGIFTXQLEN	Возвращает размер очереди на передачу в поле **ifr_qlen**.
SIOCSIFTXQLEN	Устанавливает размер очереди на передачу в соответствии со значением поля **ifr_qlen**. Операция является привилегированной.
SIOCSIFNAME	Заменяет имя интерфейса на значение, переданное в поле **ifr_name**. Операция является привилегированной и может использоваться только при отключенном (down) интерфейсе.
SIOCGIFCONF[2]	Возвращает список адресов сетевого уровня для интерфейса. В настоящее время поддерживается только определение адресов Ipv4 (семейство **AF_INET**). Пользовательская программа передает в качестве аргумента функции **ioctl** структуру данных **ifconf**, содержащую указатель на массив структур **ifreq** в поле **ifc_req** и размер массива в поле **ifc_len**. Ядро заполняет структуры **ifreq** данными активных интерфейсов сетевого уровня, помещая в поле **ifr_name** имена интерфейсов (**eth0:1** и т. п.), а в поле **ifr_addr** - IP-адреса. Актуальный размер массива адресов ядро возвращает в поле **ifc_len**. Если возвращенное значение **ifc_len** совпадает с переданным, это может говорить о недостаточном размере переданного функции массива. В таких случаях целесообразно повторить попытку с большим размером буфера для передачи адресов.

Таблица 137. Флаги устройств для netdevice.

Флаг	Описание
IFF_UP	Интерфейс активен.
IFF_BROADCAST	Для интерфейса установлен корректный широковещательный адрес.
IFF_DEBUG	Внутренний флаг отладки.
IFF_LOOPBACK	Интерфейс является петлевым (loopback).
IFF_POINTOPOINT	Интерфейс подключен к каналу “точка-точка”.
IFF_RUNNING	Ресурсы выделены.

1 *Maximum Transfer Unit - максимальный размер передаваемого блока.*
2 *Эта операция неразрывно связана с протоколом IP (Приложение 12.5) и относится скорее к этому протоколу, нежели к интерфейсу netdevice.*

Флаг	Описание
IFF_NOARP	Адрес канального уровня не задан, протокол ARP не используется.
IFF_PROMISC	Интерфейс находится в режиме захвата пакетов.
IFF_NOTRAILERS	Избегать использования трейлеров.
IFF_ALLMULTI	Принимать все пакеты с групповыми адресами.
IFF_MASTER	Интерфейс является ведущим в транке с распределением нагрузки.
IFF_SLAVE	Интерфейс является ведомым в транке с распределением нагрузки.
IFF_MULTICAST	Включает поддержку групповой адресации.
IFF_PORTSEL	Интерфейс может выбирать тип среды с помощью **ifmap**.
IFF_AUTOMEDIA	Автоматический выбор среды активизирован.
IFF_DYNAMIC	При отключении интерфейса адрес был потерян.

Большинство протоколов поддерживает дополнительные операции IOCTL для настройки связанных с протоколом опций интерфейса. В частности, протокол IP поддерживает большой набор операций IOCTL, описанных в Приложении 12.5 (стр. 384).

Кроме того, некоторые устройства могут поддерживать фирменные расширения IOCTL.

Имена интерфейсов (включая те, которые не имеют адресов или флага **IFF_RUNNING**) можно найти в файле **/proc/net/dev** (см. 360).

12.12.2 Известные ограничения

Библиотека **glibc 2.1** не содержит макроса **ifr_newname** в файле **net/if.h**. Для решения проблемы достаточно добавить приведенный ниже текст в этот файл.

```
#ifndef ifr_newname
#define ifr_newname     ifr_ifru.ifru_slave
#endif
```

12.13 Функция ioctl

Функция **ioctl**, описанная в заголовочном файле **<sys/ioctl.h>**, работает с параметрами устройств через специальные файлы, связанные с устройствами.

```
int ioctl(int d, int request, argp);
```

Параметр **d** при вызове функции должен содержать дескриптор открытого файла. Вторым аргументом функции является связанный с устройством код операции (запроса), а третьим параметром может быть указатель на объект произвольного типа в памяти системы.

При успешном завершении заданной операции функция обычно возвращает нулевое значение, однако ряд операций может использовать код возврата функции для передачи выходного значения[1]. При возникновении функция возвращает значение -1 или отрицательный код ошибки.

Список поддерживаемых операций можно получить с помощью команды **man ioctl_list**, но это руководство зачастую является устаревшим[2], поэтому лучше найти соответствующую информацию в дистрибутиве используемого ядра Linux[3].

12.13.1 Коды ошибок

Таблица 138. Коды ошибок IOCTL.

Код	Описание
EBADF	Параметр **d** не содержит корректный файловый дескриптор.
EFAULT	Параметр **argp** указывает на недоступную область памяти.
ENOTTY	Параметр **d** не связан со специальным символьным устройством.
ENOTTY	Запрошенная операция не применима к объекту, заданному параметром **d**.
EINVAL	Некорректная операция или **argp**.

12.14 Структуры данных utmp и wtmp

Структуры данных **utmp** и **wtmp** используются для записи учетной информации в журнальные файлы системы. Описание структуры содержится в файле

1 *Это значение не должно быть отрицательным.*
2 *Самая свежая версия, которую мне довелось видеть, была написана для ядра 1.3.27.*
3 *Файл Documentation/ioctl-number.txt из дистрибутива ядра содержит ссылки на заголовочные файлы, в которых определены операции IOCTL.*

```
#include <utmp.h>
```

Файл **utmp** содержит информацию об активных пользователях системы. Число реальных пользователей может быть больше, поскольку не все программы делают записи в системный журнал **utmp**.

Запись в файл **utmp** можно открывать только для пользователя root, поскольку целостность этого файла оказывает влияние на работу множества компонент системы. Разрешив запись в файл другим пользователям, вы лишите себя достоверной информации о состоянии системы и рискуете ее работоспособностью.

Файл представляет собой последовательность структур данных **utmp**, определенных в одном из заголовочных файлов вашей системы. Отметим, что детали описанной ниже структуры данных могут меняться в зависимости от используемой библиотеки **libc**. Ниже приведен пример структуры данных **utmp** и связанных с ней констант.

```
#define UT_UNKNOWN            0
#define RUN_LVL               1
#define BOOT_TIME             2
#define NEW_TIME              3
#define OLD_TIME              4
#define INIT_PROCESS          5
#define LOGIN_PROCESS         6
#define USER_PROCESS          7
#define DEAD_PROCESS          8
#define ACCOUNTING            9

#define UT_LINESIZE           12
#define UT_NAMESIZE           32
#define UT_HOSTSIZE           256

struct exit_status {
    short int e_termination;    /* статус прерывания процесса */
    short int e_exit;           /* статус выхода */
};
struct utmp {
    short ut_type;              /* тип регистрации в системе (login) */
    pid_t ut_pid;               /* идентификатор процесса регистрации */
    char ut_line[UT_LINESIZE];  /* имя устройства, использованного для входа в систему */
                               /* "/dev/" */
    char ut_id[4];              /* init id или сокращенное имя tty */
    char ut_user[UT_NAMESIZE];  /* имя пользователя */
    char ut_host[UT_HOSTSIZE];  /* имя удаленного хоста */
    struct exit_status ut_exit; /* статус выхода для процесса, помеченного как */
                                /* DEAD_PROCESS. */
    long ut_session;            /* идентификатор сессии */
    struct timeval ut_tv;       /* время создания записи */
    int32_t ut_addr_v6[4];      /* IP-адрес удаленного хоста */
    char pad[20];               /* зарезервировано */
};
/* для совместимости со старыми версиями */
#define ut_name ut_user
#ifndef _NO_UT_TIME
#define ut_time ut_tv.tv_sec
#endif
#define ut_xtime ut_tv.tv_sec
#define ut_addr ut_addr_v6[0]
```

Описанная структура включает имя специального файла, связанного с пользовательским терминалом, регистрационное имя пользователя и время регистрации пользователя в системе. Строки, размер коотрых меньше соответствующих полей структуры данных, завершаются нуль-символом.

Первая запись в файле создается в результате обработки функцией **init** конфигурационного файла **inittab**. Перед началом обработки функция **init** очищает **utmp**, устанавливая **ut_type** = **DEAD_PROCESS** и сбрасывая содержимое полей **ut_user**, **ut_host** и **ut_time** для всех записей, в которых значение поля **ut_type** не равно **DEAD_PROCESS** или **RUN_LVL** и не существует процесса с идентификатором, указанным в поле **ut_pid**. Если не найдено пустой записи с требуемым значением **ut_id**, функция **init** создает новую запись, устанавливая в поле **ut_id** значение из файла **inittab**, для полей **ut_pid** и **ut_time** - текущие значения и **ut_type** = **INIT_PROCESS**.

Функция **getty** определяет запись по идентификатору процесса, устанавливает **ut_type** = **LOGIN_PROCESS**, изменяет значение **ut_time**, устанавливает **ut_line** и ждет организации соединения. После успешной аутентификации пользователя функция **login** устанавливает **ut_type** = **USER_PROCESS**, изменяет время регистрации **ut_time** и устанавливает значения полей **ut_host** и **ut_addr**.

Когда функция **init** узнает о завершении процесса, она находит в файле **utmp** запись по значению **ut_pid**, устанавливает **ut_type** = **DEAD_PROCESS** и сбрасывает значения полей **ut_user**, **ut_host** и **ut_time**.

Эмуляторы терминал (например, **xterm**) напрямую создают запись USER_PROCESS и генерируют **ut_id**, используя два последних символа имени устройства **/dev/ttyp%c** или **p%d** для устройства **/dev/pts/%d**. Если для процесса программа эмуляции терминала обнаруживает запись **DEAD_PROCESS**, она использует такую запись, а при ее отсутствии создает новую. По возможности при завершении работы программа эмуляции терминала помечает созданную запись как **DEAD_PROCESS** и устанавливает пустые значения для полей **ut_line**, **ut_time**, **ut_user** и **ut_host**.

Графические менеджеры экрана **xdm** не должны создавать записи **utmp**, поскольку они не связаны с терминалом. Если позволить создание таких записей, в системе будут возникать ошибки. Менеджерам экрана следует создавать записи **wtmp**.

Демон **telnetd** создает запись **LOGIN_PROCESS** и выполняет остальную часть процесса регистрации с помощью функции **login**. После завершения сеанса демон **telnetd** очищает запись **utmp** как описано выше для функции **init**.

Файл **wtmp** содержит записи для всех случаев регистрации (login) и выхода пользователей из системы. Формат этого файла в точности совпадает с форматом файла **utmp**, однако пустые имена пользователей показывают выход из системы (logout) для соответствующего терминала. Записи, где в качестве имени пользователя указана строка **shutdown** или **reboot**, говорят об отключении или перезагрузке системы. Записи в файл **wtmp** обеспечиваются программами **login**, **init** и некоторыми версиями getty. Ни одна из этих программ не создает файл, поэтому при его удалении сведения о регистрации пользователей в системе будут теряться. Для просмотра записей из файла **wtmp** служит утилита **last** (параграф 2.8.1.2 на стр. 47).

12.15 SYN cookie[1]

D. J. Bernstein
Работа почтового сервиса Panix, принадлежащего ISP в Нью-Йорке, была блокирована атакой SYN flood, начавшейся 6 сентября 1996. Неделей позже история повторилась с RISKS Digest, Wall Street Journal, Washington Post и многими другими газетами.

Атаки SYN flood[2] были предсказаны экспертами до их реального обнаружения. Многие считают проблему таких атак неразрешимой. Например, Garfinkel и Spafford в книге “Practical UNIX and Internet Security” (стр. 778) пишут:

> *На стороне адресата[3] возникает большое число полуоткрытых соединений, поглощающих ограниченные системные ресурсы. Обычно в таких соединениях указываются подставные адреса отправителей, которые указывают на несуществующие или недоступные хосты. Таким образом, атакующий не имеет возможности отследить источник атаки. В такой ситуации вы сможете сделать очень мало для предотвращения атаки ... любой ограниченный ресурс когда-либо будет исчерпан.*

Увеличение очередей SYN и случайное упреждающее отбрасывание соединений (drop) усложняет жизнь инициаторам атак SYN flood, но не решает проблему полностью.

Функции SYN cookie используют криптографические алгоритмы для решения проблемы. Я писал как это можно сделать[4] 16 сентября 1996; Вместе с Эриком Шенком (Eric Schenk) в течение нескольких следующих недель были выработаны детальные предложения[5] по решению проблемы. Джеф Вайсберг (Jeff Weisberg) реализовал это в программе для SunOS в октябре 1996, а Эрик Шенк создал в феврале 1997 программную реализацию для 1997.

Функции SYN cookie в настоящее время являются стандартным решением для операционных систем Linux и FreeBSD. К сожалению, в ОС Linux эти функции по умолчанию отключены. Для того, чтобы включить их, достаточно добавить команду

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

в сценарий загрузки[6].

12.15.1 Что такое SYN cookie?

SYN cookie - это одно из решений задачи выбора начальных порядковых номеров TCP серверами TCP. Разница между начальными порядковыми порядковыми номерами на серверах и клиентах заключается в:

- 5 старших битов: значение t mod 32, где t - 32-разрядный счетчик временных интервалов, значение которого увеличивается на 1 каждые 64 секунды;
- следующие 3 бита: кодированное значение MSS[7], выбранное сервером в ответ на MSS клиента;
- младшие 24 бита: выбранная сервером на основе IP-адресов и номеров портов отправителя и получателя, а также величины t значение секретной функции.

Такой алгоритм выбора начального порядкового номера соответствует основным требованиям протокола TCP, в соответствии с которыми номера должны увеличиваться достаточно медленно и начальные порядковые номера для серверов растут несколько быстрее, нежели порядковые номера для клиентов.

Серверы, использующие функции SYN cookie, не отвергают соединения при заполнении очереди SYN. Взамен они передают инициатору соединения пакет SYN+ACK, в точности соответствующий пакету, который был бы передан при большем размере очереди SYN (исключения: сервер должен отвергать (reject) опции TCP такие, как большое окно, и должен использовать 1/8 значения MSS, которое он может кодировать). При получении пакета ACK, сервер убеждается в работе секретной функции для последнего (recent) значения t и перестраивает запись очереди SYN в соответствии со значением MSS.

1 *Оригинал этого документа можно найти на сайте cr.yp.to/syncookies.html, а копия имеется в каталоге Documents/ приложенного к книге компакт-диска.*
2 *Интенсивный поток пакетов TCP с установленным битом SYN (попытка организации соединения). Прим. перев.*
3 Объекта атаки.
4 *http://cr.yp.to/syncookies/idea*
5 *http://cr.yp.to/syncookies/archive*
6 *Требуется также включить поддержку этой функции я ядре Linux при его компиляции (см. параграф 4.4.2.2.8 на стр. 72).*
7 *Максимальный размер сегмента TCP.*

Атаки SYN flood представляют собой просто серии пакетов SYN с подставными адресами IP. Эти адреса выбираются случайным образом и не содержат никакой информации об атакующей стороне. Атаки SYN flood заполняют SYN-очереди серверов. Обычно это приводит к тому, что сервер отвергает входящие соединения. Использующий функции SYN cookie сервер будет продолжать нормально работать во время таких атак, Максимальное воздействие атаки SYN flood на такой сервер будет состоять в блокировке использования больших окон.

12.15.2 Атаки вслепую

Если атакующий сможет угадать порядковый номер, переданный какому-либо из хостов, он сможет организовать обманное соединение от имени этого хоста.

Атакующие могут попытаться предпринять криптоанализ выбранной сервером секретной функции, просматривая последовательность корректных cookie и пытаясь предсказать следующее значение cookie. При эффективной реализации функций шанс корректно предсказать порядковый номер незначительно превышает шансы на угадывание случайного числа при равномерном распределении. Для обеспечения безопасности были созданы средства аутентификации секретных ключей (Secret-key message authenticator). Достаточную скорость и безопасность обеспечивает функция, кодирующая входные данные в 16 байтов, обрабатывающая их с помощью алгоритма Rijndael и представляющая в качестве результата первые 24 бита.

Независимо от используемой функции атакующий для достижения успеха в попытках организации подставного соединения должен принять миллионы случайных пакетов ACK. Серверы могут осложнить организацию таких атак двумя путями:

- Хранение информации о времени последнего переполнения очереди SYN (для каждой очереди отдельно, а не в глобальной переменной). Отсутствующие записи очереди SYN не создаются заново, если не было недавнего переполнения очередей. Это позволяет предотвратить прохождение подставных пакетов ACK через брандмауэры с блокировкой SYN.
- Добавление другого числа в cookie: выбранная сервером 32-битовая секретная функция от адресов клиента и сервера (без учета текущего времени). Это потребует от атакующего подбора 32 битов взамен 24.

Протокол с поддержкой 128-битовых порядковых номеров сделает атаки вслепую практически невозможными.

12.15.3 Кто создал SYN cookie?

Как мне известно, Фил Кэрн (Phil Karn) был первым, кто разработал протокол Internet, использовавший cookie для защиты от DoS-атак вслепую. Однако идея более стара.

По моему мнению, я был первым, кто указал, что серверы TCP могут использовать cookie без каких-либо изменений протокола TCP. (Perry Metzger впоследствии заявлял, что он сделал это раньше. Однако Metzger не ответил мне, когда я попросил его прислать[1] мне копии связанных с этим сообщений. Архивы NANOG содержат информацию, что Metzger заявлял 9 сентября 1996, что функции cookie требуют создания нового протокола “TCP++”, а 17 сентября 1996, - что ISP должны фильтровать исходящие от них пакеты).

Мое первое предложение не соответствует требованиям TCP к использованию возрастающих порядковых номеров. У Эрика Шенка (Eric Schenk) возникла идея добавить что-либо к порядковым номерам на стороне клиентов.

Я предложил зависящие от времени функции SYN cookie. Временная зависимость не дает атакующим возможности (1) собирать корреткные cookie на компьютере общего пользования и (2) впоследствии повторно использовать эти cookie для атаки с другого компьютера.

12.15.4 Страшилки о SYN cookie

Некоторые люди (в частности Alexey Kuznetsov, Wichert Akkerman и Perry Metzger) распространяют дезинформацию и функциях SYN cookie. Ниже приведено несколько примеров таких некорректных заявлений[2]:

- *Функции SYN cookie “являются серьезным нарушением протокола TCP.”* В реальности функции SYN cookie полностью соответствуют требованиям протокола TCP. Каждый пакет, передаваемый сервером с поддержкой SYN cookie представляет собой совокупность данных, которая может быть передана и сервером, не поддерживающим SYN cookie.
- *Фунции SYN cookie “не позволяют использовать расширения TCP" такие, как большой размер окна.* В реальности SYN cookie не оказывают влияния на расширения TCP. Соединения, сохраненные с помощью SYN cookie, не могут использовать окна большого размера, но то же самое произойдет и без SYN cookie, поскольку соединение будет просто уничтожено.
- *Функции SYN cookies могут вызывать “масштабное “зависание” соединений.”* В реальности соединения время от времени “зависают”, независимо от использования SYN cookie, при перегрузке компьютеров или сетей. Приложения в таких случаях просто отбрасывают “умершие” соединения.
- *Функции SYN cookie вызывают “серьезное снижение производительности служб.”* Реально функции SYN cookie повышают эффективность сервиса. Во время вычислений функции отнимают незначительные ресурсы CPU, но это процессорное время так или иначе было бы потрачено на создание трудно предсказуемых порядковых номеров (см. RFC 1948).
- *SYN cookie может вызывать “магический сброс (magic reset).”* В реальности функции SYN cookie никогда не приводят к сбросу.

1 *http://cr.yp.to/syncookies/metzger*

2 *Некоторые из этих заявлений приводятся в рассмотренном выше файле ip-sysctl.txt из дистрибутива ядра Linux (см. стр. 369).*

Эти люди распространяют свои заблуждения другим людям, в частности, мне. Я не знаю, что является причиной - умысел или заблуждение, - но в любом случая буду рад установлению истины.

Я предлагал Кузнецову отказаться от его заявления или отстаивать свое мнение в дискуссии. Он отказался от этого предложения. Я уверен, что сейчас он понимает ошибочность своего заявления и любые попытки отстаивать эту точку зрения обречены на неудачу. Очень жаль, что он не захотел участвовать в установлении истины.

Был приглашен к обсуждению и Akkerman, но он просто не ответил.

12.16 Структура сетевого буфера Linux - skb

В этом приложении описана структура буфера сокетов Linux и функции, определенные для этого буфера. Описание дается в соответствии с файлом **include/linux/skbuff.h** из дистрибутива ядра версии 2.6.6

12.16.1 Буфер сокетов skbuff

Структура **skbuff** определяет набор полей данных, которые используются ядром Linux при обработке сетевых пакетов. Принятые сетевыми интерфейсами пакеты помещаются в буферы **skbuff**, которые передаются сетевому стеку, использующему буфер в течение всего процесса обработки пакета.

12.16.1.1 struct sk_buff

Структура **sk_buff**, определенная в файле **<linux/skbuff.h>**, включает поля, перечисленные в таблице 139.

Таблица 139. Поля структуры sk_buff.

Имя	Описание
next	Указатель на следующий буфер **skbuff** (пакет) в списке.
prev	Указатель на предыдущий буфер **skbuff** (пакет) в списке.
list	Указатель на список (структура **sk_buff_head**), к которому относится данных буфер.
sk	Указатель на сокет (структура **sock**), к которой относится буфер.
stamp	Время прибытия пакета.
dev	Указатель на устройство, принявшее пакет, или устройство, через которое он будет передан.
rx_dev	Указатель на реальное устройство, принявшее пакет. Операция **ROUTE** (параграф 5.1.8.2.15 на стр. 114) программы **iptables** может изменять значение этого поля после принятия ядром решения о маршрутизации.
h	Указатель на структуру, содержащую заголовок транспортного уровня для пакета (tcp, udp, icmp, igmp, spx, raw)
nh	Указатель на заголовок сетевого уровня для пакета (ip, ipv6, arp, ipx, raw)
mac	Указатель на заголовок канального уровня для пакета.
dst	Указатель на получателя (структура **dst_entry**, используемая XFRM[1]).
sp	Указатель на безопасный путь (структура **sec_path**, используемая XFRM).
cb	Управляющий буфер размером 48 байт. Использование этого буфера на каждом уровне не оговаривается, что позволяет помещать в буфер любые служебные переменные. Если нужно передавать буфер между уровнями, следует воспользоваться функцией **skb_clone**.
len	Размер актуальных данных в октетах.
data_len	Размер данных в октетах.
mac_len	Размер заголовка канального уровня в октетах.
csum	Контрольная сумма.
local_df	Это поле типа unsigned char в настоящее время не используется.
cloned	Определяет возможность клонирования head (см. ниже). Следует проверять значение счетчика **refcnt**.
pkt_type	Класс пакета.
ip_summed	Драйвер сообщил контрольную сумму IP.
priority	Приоритет пакета для очередей.
protocol	Протокол, используемый данным пакетом (сообщает драйвер).
security	Уровень безопасности для пакета.
destructor	Указатель на функцию уничтожения буфера.
nfmark	Маркер netfilter, который может использоваться для связи между ловушками (см. стр. 102). Этот параметр присутствует только при включенной опции ядра **NETFILTER** (см. стр. 73).

1 *См. описание опции XFRM на стр. 84.*

Имя	Описание
nfcache	Информация о внутреннем кэше netfilter. Этот параметр присутствует только при включенной опции ядра **NETFILTER** (см. стр. 73).
nfct	Указатель на связанные с пакетом соединения, если таковые присутствуют. Этот параметр присутствует только при включенной опции ядра **NETFILTER** (см. стр. 73).
nf_debug	Отладочная информация netfilter[2]. Этот параметр присутствует только при включенной опции ядра **NETFILTER** (см. стр. 73).
nf_bridge	Указатель на данные о кадре, проходящем через мост (bridged frame)[3]. Этот параметр присутствует только при включенной опции ядра **NETFILTER** (см. стр. 73).
ifield	Этот параметр присутствует только при включенной опции ядра **HIPPI** (поддержка сетевых устройств HIPPI).
tc_index	Индекс управления трафиком. Этот параметр присутствует только при включенной опции ядра **NET_SCHED** (стр. 87). Индекс может использоваться для классификации исходящих пакетов при включенной опции ядра **TC index classifier** (см. параграф 4.4.2.2.33.2.19.1 на стр. 90).
truesize	Реальный размер буфера в октетах.
users	Счетчик использования (см. файлы **datagram.c** и **tcp.c** из дистрибутива ядра)
head	Указатель на начало буфера.
data	Указатель на начало данных в буфере.
tail	Указатель на конец данных.
end	Указатель на конец.

12.16.1.2 Функции для работы с буфером skb

Существует достаточно большой набор функций для работы с буферами **skb** на уровне **sk_buff**. Ниже кратко рассмотрены основные из этих функций.

12.16.1.2.1 Функции распределения памяти для буферов

struct sk_buff *alloc_skb(unsigned int size, int gfp_mask)
Эта функция служит для выделения памяти и создания нового буфера skb. При создании буфера происходит инициализация требуемых переменных и учет выделяемой памяти.

void kfree_skb(struct sk_buff *skb)
Уменьшает на 1 значение счетчика использования буфера и удаляет указанный skb, если его больше никто не использует.

struct sk_buff *skb_get(struct sk_buff *skb)
Увеличивает на 1 значение счетчика использования буфера и возвращает указатель на этот буфер.

struct sk_buff *skb_clone(struct sk_buff *skb, int gfp_mask)
Эта функция служит для клонирования буфера skb. Обе копии используют общий блок данных из пакета, но каждая имеет свою структуру sk_buff. Новая копия не принадлежит ни одному из сокетов и значение счетчика использования равно 1.

struct sk_buff *skb_copy(const struct sk_buff *skb, int gfp_mask)
Создает полную копию skb, включая данные из пакета. Такое копирование требуется в тех случаях, когда планируется изменение содержащихся в пакете данных. Счетчик использования для нового skb имеет значение 1.

struct skb_copy_expand(const struct sk_buff *skb, int new_headroom, int new_tailroom, int gfp_mask)
Создает полную копию skb (включая данные из пакета), выделяя дополнительную память в начале (new_headroom байт) и в конце (new_tailroom байт) пространства данных буфера.

12.16.1.2.2 Дополнительные функции

int skb_cloned(struct sk_buff *skb)
Эта функция позволяет проверить наличие клонов skb.

int skb_shared(struct sk_Buff *skb)
Эта функция позволяет проверить факт совместного использования данного skb (значение счетчика фактов использования > 1).

static inline struct sk_buff *skb_share_check(struct sk_buff *skb, int pri)
Функция проверяет факт совместного использования буфера. Если разделяемый буфер является клоном, функция создает новую копию, уменьшая счетчик использования для первоначального буфера и возвращая указатель на новый клон, для которого счетчик использования имеет значение 1. Если проверяемый буфер не является разделяемым, функция возвращает указатель на исходный буфер.

static inline struct sk_buff *skb_unshare(struct sk_buff *skb, int pri)
Функция создает новую копию разделяемого буфера. Если разделяемый буфер является клоном, функция создает новую копию, уменьшая счетчик использования для первоначального буфера и возвращая указатель на новый клон, для которого счетчик использования имеет значение 1. Если проверяемый буфер не является клоном, функция возвращает указатель на исходный буфер.

2 *Используется только при включенной опции CONFIG_NETFILTER_DEBUG, см. стр. 74.*
3 *Используется только при включенной опции CONFIG_BRIDGE_NETFILTER, см. стр. 74.*

12.16.1.2.3 Функции для работы со списками skb (очередями)

static inline void skb_queue_head_init(struct sk_buff_head *list)
Функция инициализирует список буферов skb.
struct sk_buff *skb_peek(struct sk_buff_head *list_)
Читает первый буфер skb в списке, не удаляя этот skb из списка.
struct sk_buff *skb_peek_tail(struct sk_buff_head *list_)
Читает последний буфер skb в списке, не удаляя этот skb из списка.
__u32 skb_queue_len(sk_buff_head *list_)
Возвращает размер указанного списка skb.
void skb_queue_head(struct sk_buff_head *list_, struct sk_buff *newsk)
Помещает skb в начало указанного списка (очереди).
void skb_queue_tail(struct sk_buff_head *list_, struct sk_buff *newsk)
Помещает skb в конец указанного списка (очереди).
struct sk_buff *skb_dequeue(struct sk_buff_head *list_)
Извлекает первый буфер skb из списка (очереди).
struct sk_buff *sbk_dequeue_tail(struct sk_buff_head *list_)
Извлекает последнй буфер skb из списка (очереди).
extern void skb_insert(struct sk_buff *old, struct sk_buff *newsk);
Вставляет буфер в очередь.
extern void skb_append(struct sk_buff *old, struct sk_buff *newsk);
Вставляет буфер в очередь вслед за указанным буфером.
extern void skb_unlink(struct sk_buff *skb);
Удаляет буфер из очереди.
static inline int skb_queue_empty(const struct sk_buff_head *list)
Проверяет наличие в очереди буферов skb.

12.16.1.2.4 Операции с данными skb

unsigned char *skb_put(struct sk_buff *sbk, int len)
Расширяет размер области данных skb. Если общий размер будет превосходить размер skb, возникает критическая ситуация (kernel will). Функция возвращает указатель на первый октет нового блока данных.
unsigned char *skb_push(struct sk_buff *skb, int len)
Расширяет размер области данных skb в начале буфера. Если общий размер будет превосходить размер skb, возникает критическая ситуация (kernel will). Функция возвращает указатель на первый октет нового блока данных.
unsigned char *skb_pull(struct sk_buff *skb, int len)
Удаляет данные из начала буфера, возвращая память в to headroom. Функция возвращает указатель на первый октет нового блока данных.
int skb_headroom(struct sk_buff *skb)
Определяет размер свободного пространства в начале (head) буфера skb.
int skb_tailroom(struct sk_buff *skb)
Определяет размер свободного пространства в конце (tail) буфера skb.
static inline void skb_reserve(struct sk_buff *skb, unsigned int len)
Освобождает пространство в начале буфера , занимая место в конце его.
struct sk_buff *skb_cow(struct sk_buff *skb, int headroom)
Если в буфере ощущается нехватка памяти в headroom или буфер является клоном буфер копируется с выделением дополнительного пространства в начале.
static inline void skb_trim(struct sk_buff *skb, unsigned int len)
Удаляет len байтов данных из хвоста буфера. Если размер данных не превышает len, буфер не изменяется.

12.17 Управление модулями ядра Linux (module-init-tools)

Серия утилит для управления модулями ядра Linux обеспечивает возможность просмотра списка загруженных модулей ядра, загрузки дополнительных модулей и удаления из памяти ненужных модулей.

Утилиты **modprobe** (параграф 12.17.1) и **depmod** (стр. 407) предназначены для управления загружаемыми модулями ядра Linux пользователями и администраторами системы. Программа **modprobe** использует файл зависимостей, созданный с помощью **depmod**, для автоматической загрузки набора модулей из определенного каталога файловой системы.

Если установлена переменная окружения **UNAME_MACHINE**, утилиты работы с модулями будут использовать значение этой переменной вместо информации, возвращаемой системным вызовом **uname** (см. параграф 11.1.1.9 на стр. 192). Это может быть полезно при компиляции 64-битовых модулей на 32-битовой системе и наоборот.

12.17.1 modprobe

Программа **modprobe** позволяет загружать модули ядра Linux и удалять загруженные ранее модули, обеспечивая функции интеллектуального управления модулями. Для реальной загрузки или удаления модулей используются утилиты **insmod** (параграф 12.17.4 на стр. 408) и **rmmod** (параграф 12.17.5 на стр. 409), соответственно, а **modprobe** выполняет для них функции командного процессора (front-end). Программа ищет в каталоге **/lib/modules/<ядро>**[1] загружаемые модули и другие файлы, требуемые для работы программы, за исключением конфигурационного файла **/etc/modprobe.conf**. Отметим, что для удобства использования программа **modprobe** не

1 ***<ядро>*** *означает номер версии используемого ядра, возвращаемый командой* ***uname -r*** *(см. параграф 11.1.1.9 на стр. 192).*

различает в именах модулей символы “**-**” и “**_**”.

Modprobe позволяет загружать одиночные модули, группу (стек) связанных модулей или все модули, соответствующие спецификации командной строки. Команда **modprobe** будет автоматически загружать все базовые модули, требуемые для стека модулей, в соответствии с зависимостями, описанными в файле **modules.dep**. Если при загрузке одного или нескольких модулей стека возникает ошибка, весь стек модулей текущей сессии будет автоматически удален из памяти.

Программа использует два варианта загрузки модулей. В первом (probe mode) программа будет пытаться загрузить модули из списка, соответствующего спецификации командной строки. После успешной загрузки первого модуля работа программы завершается. Во втором режиме программа будет пытаться загрузить все модули из заданного командой списка.

Поведение программы мере зависит о параметров конфигурации, заданных в файле **/etc/modules.conf** (параграф 12.17.6 на стр. 409).

Отметим, что заданные в конфигурационном файле команды подготовки к удалению и “зачистки” после удаления модулей (pre-remove и post-remove) не будут выполняться при удалении модуля в процессе автоматической очистки (autoclean), выполняемой модулем kerneld. Если вы хотите использовать эти команды, следует отключить режим **autoclean** для модуля **kerneld**, поместив вместо этого в файл **crontab** команду периодического удаления неиспользуемых модулей. Например, приведенная ниже строка будет обеспечивать проверку и удаление ненужных модулей каждые две минуты.

```
*/2 * * * * test -f /proc/modules && /sbin/modprobe -r
```

Синтаксис

```
modprobe [-adnqv] [-C <файл>] <модуль> [symbol=<значение> ...]
modprobe [-adnqv] [-C <файл>] [-t <тип>] <шаблон>
modprobe -l [-C <файл>] [-t <тип>] <шаблон>
modprobe -c [-C <файл>]
modprobe -r [-dnv] [-C <файл>] [<модуль> ...]
modprobe -Vh
```

12.17.1.1 Опции

Таблица 140. Опции modprobe.

Опция		Описание
-a	**--all**	Загружает все соответствующие условиям модули, не останавливаясь после успешной загрузки первого найденного модуля.
-c	**--showconfig**	Показывает текущие параметры конфигурации (используемые по умолчанию и заданные в конфигурационном файле).
-C	**--config**	Загружает указанный параметром файл взамен используемого по умолчанию конфигурационного файла **/etc/modules.conf** (параграф 12.17.6 на стр. 409). Для выбора конфигурационного файла можно также использовать переменную окружения **MODULECONF**.
-d	**--debug**	Задает вывод отладочной информации о внутреннем представлении стека модулей.
-h	**--help**	Выводит на экран справочную информацию и завершает работу программы.
-k	**--autoclean**	Устанавливает для загружаемых модулей флаг **autoclean**, используемый ядром при обнаружении отсутствия функций, реализованных в загружаемом ядре. Опция **-k** неявно предполагает наличие ключа **-q**. Обе эти опции передаются программе **insmod**.
-l	**--list**	Выводит список соответствующих другим опциям модулей. Например, при использовании вместе с опцией **-t** показывает модули определенного типа.
-n	**--show**	Просто показывает список действий, которые были бы выполнены без этой опции.
-q	**--quiet**	Отключает вывод сообщений **insmod** о неудачных попытках загрузки модулей.
-r	**--remove**	Удаляет модуль (стек модулей) или выполняет очистку (autoclean) в зависимости от наличия в командной строке имени модуля. Команда **modprobe -r** без дополнительных параметров будет удалять загруженные автоматически и не используемые модули, выполняя перед удалением и после него все команды, указанные для этих операций в файле **/etc/modules.conf** (стр. 409).
-s	**--syslog**	Задает вывод сообщений об ошибках в **syslog** взамен их вывода на экран (stderr). Эта опция автоматически передается программе **insmod**.
-t	**--type**	Задает тип модулей, которые будут соответствовать данной команде. Программа **modprobe** будет смотреть только в каталогах, имя которых включает указанный параметром опции тип. Например, **-t drivers/net** будет задавать выбор модулей только из каталога **xxx/drivers/net/** и его подкаталогов.
-v	**--verbose**	Задает вывод информации о всех выполняемых командах.
-V	**--version**	Выводит информацию о номере версии и завершает работу программы.

Отметим, что имена модулей в командной строке **modprobe** не должны включать полного пути к ним или суффикса имени файла (.**o** или .**ko**)

12.17.1.2 Стратегия поиска модулей

Программа **modprobe** будет сначала искать модули в каталоге текущей версии ядра. Если модуль не найден в этом каталоге, **modprobe** будет смотреть в каталогах близких версиях ядра (например, каталоги ядер версии 2.6). Если модуль отсутствует и там, **modprobe** будет обращаться к каталогу, содержащему модули используемой по умолчанию версии и т. д..

При установке новой версии ядра модули должны помещаться в каталог, связанный с версией и выпуском (release) устанавливаемого ядра.

При наличии модулей, не зависящих от версии ядра, их следует помещать в отдельные подкаталоги дерева **/lib/modules**.

Описанная здесь стратегия поиска модулей может быть изменена в файле /etc/modules.conf (параграф 12.17.6).

12.17.1.3 Примеры использования

```
modprobe -t net
```
Будет загружать модули из каталога **net** , пока один из них не будет удачно загружен.

```
modprobe -a -t boot
```
будет загружать все модули из каталога boot.

```
modprobe slip
```
будет пытаться загрузить модуль **slhc** (если он не был загружен ранее), поскольку модуль **slip** использует функции модуля **slhc**. Зависимости между модулями описываются в файле **modules.dep**, создаваемом программой **depmod** (параграф 12.17.2 на стр. 407).

```
modprobe -r slip
```
будет удалять модуль **slip**. Модуль **slhc** также будет автоматически удален, если он не используется другими модулями (например, ppp).

12.17.1.4 Безопасный режим

Если эффективный идентификатор пользователя отличается от реального UID, программа **modprobe** очень жестко проверяет введенную пользователем команду. Последний параметр всегда трактуется как имя модуля, даже если он начинается с символа "**-**". Опции в стиле **variable=value** просто не принимаются программой. Имя модуля всегда трактуется как строка и в безопасном режиме не допускается использование мета-символов. Однако и в этом режиме допустимо использование мета-символов в числе данных, получаемых из конфигурационного файла.

Эффективный идентификатор пользователя может отличаться от реального в случаях вызов **modprobe** ядра[1]. В идеальной ситуации **modprobe** может доверять вызовам из ядра, не предполагая в них некорректных параметров. Однако известен по крайней мере один пример передачи ядром программе **modprobe** непроверенных параметров от пользователя. С тех пор **modprobe** не доверяет даже ядру.

Программа **modprobe** автоматически переходит в безопасный режим при наличии любой из перечисленных ниже переменных окружения:

```
HOME=/
TERM=linux
PATH=/sbin:/usr/sbin:/bin:/usr/bin
```

12.17.1.5 Протоколирование команд

Если существует каталог **/var/log/ksymoops** и программа **modprobe** запускается с опциями загрузки или удаления модулей, **modprobe** будет передавать все команды в журнальный файл и возвращать результат операции в файл **/var/log/ksymoops/`date +%Y%m%d.log**. Не существует опций для запрета автоматической записи в журнальный файл и единственным способом избавиться от протоколирования является удаление каталога **/var/log/ksymoops**.

12.17.1.6 Конфигурационный файл modprobe.conf

Поскольку программа **modprobe** может добавлять и удалять целые группы модулей, связанных между собой, требуется метод задания опций, используемых при загрузке модулей. Файл **modprobe.conf** используется для хранения таких опций. Этот файл можно также использовать для создания псевдонимов имен модулей. И, наконец, этот файл оказывает существенное влияние на поведение программы **modprobe**, позволяя управлять такими операциями как загрузка или удаление групп модулей.

Отметим, что в файле **modprobe.conf** символы "**-**" и "**_**" в именах и псевдонимах модулей не различаются, как и в командной строке **modprobe**.

Формат файла **modprobe.conf** весьма прост - каждая строка файла содержит одну команду, а пустые строки и строки, начинающиеся символом **#** (комментарии) не принимаются во внимание. Символ "****" в конце строки означает продолжение команды на следующей строке - это делает файл более читаемым для человеке.

12.17.1.6.1 Команды

```
alias <шаблон> <имя модуля>
```
Эта команда позволяет задать для модуля альтернативное имя - псевдоним. Например, команда **alias my-mod**

1 *Такая возможность поддерживается ядрами, начиная с версии 2.4.0-test11.*

really_long_modulename позволит вам использовать команду **modprobe my-mod** вместо команды **modprobe really_long_modulename**. При создании псевдонимов можно использовать поддерживаемые командным процессором шаблоны. Допускается создавать псевдонимы для псевдонимов, но это может привести к путанице. Отметим, что псевдонимы могут включать опции, которые будут добавляться к реальному имени.

```
options <имя модуля> <опция>...
```

Эта команда позволяет добавить опции к имени модуля[1]. Эти опции будут автоматически использоваться при загрузке указанного в команде модуля[2].

```
install <имя модуля> <команда>...
```

Эта команда является наиболее мощной среди поддерживаемых для файла **modprobe.conf** - она задает программе **modprobe** необходимость использования указанной команды вместо обычной загрузки заданного в строке модуля. В качестве параметра **<команда>** может использоваться любая команда, понятная используемому командному интерпретатору[3] (shell) - это позволяет вам выполнять практически любые операции. Например, если вам известно, что модуль **fred** работает лучше, если до него установлен модуль **barney**, но эти модули не связаны зависимостью и **modprobe** не будет автоматически загружать модуль **barney**, вы можете использовать в файле строку

```
install fred /sbin/modprobe barney; /sbin/modprobe --ignore-install fred
```

которая выполнит требуемые операции.

Вы можете также использовать эту команду для загрузки другого модуля при отсутствии нужного. Например, команда:

```
install probe-ethernet /sbin/modprobe e100 || /sbin/modprobe eepro100
```

будет пытаться загрузить драйвер **e100**, а при его отсутствии - драйвер **eepro100**, если пользователь введет команду

```
modprobe probe-ethernet
```

Команда

```
remove <имя модуля> <команда>...
```

подобна описанной выше команде **install**, но используется с командой **modprobe -r** для удаления модулей. Эквиваленты приветеддных выше примеров для случая удаления модулей будут иметь вид:

```
remove fred /sbin/modprobe -r --ignore-remove fred && /sbin/modprobe -r barney
```

и

```
remove probe-ethernet /sbin/modprobe -r eepro100 || /sbin/modprobe -r e100
```

Команда

```
include <имя файла>
```

позволяет использовать включаемые конфигурационные файлы.

12.17.2 depmod

Команда **depmod** создает описание зависимостей между загружаемыми модулями ядра и сохраняет его в файле **modules.dep** каталога **/lib/modules/<ядро>**. Файл зависимостей впоследствии используется программой **modprobe** (параграф 12.17.1 на стр. 404) для автоматической загрузки связанных модулей

Синтаксис

```
depmod [-aA] [-ehnqrsuvV] [-C <файл>] [-F kernelsyms] [-b <каталог>] [forced_version]
depmod [-enqrsuv] [-F kernelsyms] module1.o module2.o ...
```

Обычно описание связей между модулями создается в процессе загрузки ОС с помощью команды

```
/sbin/depmod -a
```

включаемой в один из сценариев инициализации (см. каталог **/etc/rc.d**). Это обеспечивает наличие информации о связях между модулями сразу же после загрузки операционной системы. Отметим, что использование опции **-a** в команде необязательно. При использовании в сценариях загрузки удобна будет опция **-q**, которая избавит вас от сообщений об отсутствующих связанных модулях (unresolved symbol).

Можно создать файл зависимостей сразу же после компиляции нового ядра. Команда

```
depmod -a <номер версии ядра>
```

создаст файл зависимостей для модулей нового ядра и поместит его в каталог **/lib/modules/<номер версии ядра>** несмотря на то, что в момент запуска программы используется другое ядро.

При построении файла зависимостей между модулями и экспортируемых модулями символьных строк **depmod** не принимает во внимание статус GPL для модулей или экспортируемых строк. Т. е., **depmod** не будет генерировать сообщение об ошибке если модуль без GPL-совместимой лицензии ссылается на символьную строку с лицензией “только GPL” (флаг EXPORT_SYMBOL_GPL в ядре). Однако программа **insmod** (параграф 12.17.4 на стр. 408) откажется загружать модуль в таких случаях.

1 *В качестве имени могут указываться и псевдонимы.*
2 *При загрузке модуля будут использоваться все опции - из командной строки, данной команды и команд для псевдонимов, поэтому нужно быть очень аккуратным при создании псевдонимов и автоматическом включении опций с помощью данной команды.*
3 *Например в качестве такой команды может быть передано имя shell-сценария, содержащего набор команд.*

12.17.2.1 Опции

Таблица 141. Опции depmod.

Опция		Описание
-a	**--all**	Задает поиск всех модулей во всех каталогах, указанных в конфигурационном файле (по умолчанию файл **/etc/modules.conf**).
-A	**--quick**	Задает сравнение временных меток поиск зависимостей при обнаружении обновлений. Файл зависимостей обновляется только при наличии реальных изменений.
-b	**--basedir**	Задает переданный в качестве параметра опции каталог как базовый. Это позволяет создавать файлы зависимостей при нестандартном месте хранения модулей. Созданный в результате файл зависимостей **modules.dep** не будет включать имя базового каталога и после перемещения дерева модулей на стандартное место (каталог **/lib/modules**)все ссылки станут корректными.
-C	**--config**	Загружает указанный параметром файл взамен используемого по умолчанию конфигурационного файла **/etc/modules.conf** (параграф 12.17.6 на стр. 409). Для выбора конфигурационного файла можно также использовать переменную окружения **MODULECONF**.
-e	**--errsyms**	Показывает ненайденные строки (unresolved symbol) для каждого модуля.
-F	**--filesyms**	При создании файла зависимостей для ядра, которое не используется в настоящий момент важно обеспечить использование программой **depmod** корректной таблицы символьных строк ядра для каждого модуля. Эту таблицу можно взять из файла **System.map** соответствующего ядра или из файла **/proc/ksyms**.
-h	**--help**	Выводит на экран справочную информацию и завершает работу программы.
-n	**--show**	Задает вывод файла зависимостей на **stdout** вместо записи в каталог **/lib/modules**.
-q	**--quiet**	Отключает вывод сообщений об отсутствующих символьных строках.
-r	**--root**	Некоторые пользователи компилируют модули в своей рабочей среде (non-root) и потом пытаются устанавливать модули от имени пользователя **root**. В таких случаях владельцем модулей может остаться создавший их пользователь, несмотря на установку модулей в каталог, принадлежащий пользователю **root**. В результате создается возможность для злоумышленников подменять такие модули. По умолчанию утилиты работы с модулями будут отвергать все попытки использования модулей, которыми владеет пользователь, отличный от **root**. Опция **-r** позволяет решить эту проблему и дает пользователю **root** возможность загрузки чужих модулей. Использование опции **-r** может привести к серьезному снижению уровня защиты системы.
-s	**--syslog**	Задает вывод сообщений об ошибках в **syslog** взамен их вывода на экран (stderr).
-u	**--unresolved-error**	Задает установку кода возврата при обнаружении отсутствующих ссылок для старых версий.
-v	**--verbose**	Задает вывод информации о всех выполняемых командах.
-V	**--version**	Выводит информацию о номере версии и завершает работу программы

12.17.3 lsmod

Программа **lsmod** выводит список загруженных модулей ядра Linux, указывая количество ссылок на каждый модуль из других модулей ядра. Программа предельно проста и ее задача состоит лишь в некотором форматировании содержимого файлв **/proc/modules** при выводе на экран.

12.17.4 insmod

Программа **insmod** обеспечивает загрузку модулей ядра Linux. Программа **modprobe** (параграф 12.17.1 на стр. 404) обеспечивает более эффективные средства управления модулями ядра.

Синтаксис

```
insmod [filename] [module options ...]
```

Если в качестве имени командная строка содержит дефис (**-**), программа будет принимать имена модулей со стандартного устройства ввода.

Программа выдает минимальную информацию о результатах работы. Сообщения об ошибках выводятся в кольцевой буфер ядра, содержимое которого можно прочесть с помощью команды **dmesg** (параграф 11.2.1 на стр. 206).

12.17.5 rmmod

Утилита **rmmod** позволяет удалять из памяти загруженные ранее модули ядра Linux. Программа **modprobe** (параграф 12.17.1 на стр. 404) обеспечивает более эффективные средства управления модулями ядра.

Синтаксис

```
rmmod [-f] [-w] [-s] [-v] [<имя модуля>]
```

12.17.5.1 Опции

Таблица 142 Опции rmmod

Опция		Описание
-v	**--verbose**	Задает вывод информации о всех выполняемых программой операциях.
-f	**--force**	Эта опция позволяет удалять даже используемые модули ядра, если последнее было скомпилировано со включенной опцией **CONFIG_MODULE_FORCE_UNLOAD**. Использование этой опции достаточно опасно и может приводить к серьезным последствиям для вашей системы.
-w	**--wait**	Обычно **rmmod** возвращает отказ при попытке удаления используемых модулей. С помощью этой опции программа **rmmod** будет изолировать модуль и ждать завершения работы с ним, после чего модуль будет удален.
-s	**--syslog**	Задает вывод сообщений об ошибках в **syslog** взамен их вывода на экран (stderr).
-V	**--version**	Выводит информацию о номере версии и завершает работу программы

12.17.6 Конфигурационный файл modules.conf

Конфигурационный файл **modules.conf** управляет загрузкой модулей ядра Linux. Многие аспекты поведения программ **modprobe** (параграф 12.17.1 на стр. 404) и **depmod** (параграф 12.17.2 на стр. 407) зависят от конфигурационного файла **/etc/modules.conf**.

Файл содержит строки команд и комментарии (текст справа от символа **#)**. Каждая строка файла содержит одну команду. Если команда не помещается в одной строке, она может быть перенесена в следующую строку с помощью символа **** в конце строки.

Строки команд (директив) должны использовать один из перечисленных ниже форматов:

```
[add] above module module_list
alias alias_name result
[add] below module module_list
define VARIABLE WORD
depfile=A_PATH
else
elseif EXPRESSION
endif
if EXPRESSION
include PATH_TO_CONFIG_FILE
insmod_opt=GENERIC_OPTIONS_TO_INSMOD
install module command ...
keep
[add] options module MODULE_SPECIFIC_OPTIONS
path=A_PATH
path[TAG]=A_PATH
generic_stringfile=A_PATH
pcimapfile=A_PATH
isapnpmapfile=A_PATH
usbmapfile=A_PATH
parportmapfile=A_PATH
ieee1394mapfile=A_PATH
pnpbiosmapfile=A_PATH
[add] probe name module_list
[add] probeall name module_list
prune filename
post-install module command ...
post-remove module command ...
pre-install module command ...
pre-remove module command ...
remove module command ...
persistdir directory_name
```

Все аргументы команд могут содержать метасимволы командных интерпретаторов и строки команд ОС, заключенные в обратные кавычки (``), как показано ниже:

```
path[misc]=/lib/modules/1.1.5?/local
path[net]=/lib/modules/`uname -r`/net
```

Использование shell-команд в пользовательских параметрах представляет достаточно серьезную опасность. Программы работы с модулями Linux допускают такое расширение только для проверенных данных. В частности, такие команды можно использовать в конфигурационном файле[1]. Программы[2], использующие команды работы с модулями от имени **root** с полученными от пользователя параметрами, должны установить безопасный режим работы (safe mode), поскольку в противном случае возникает риск локальных атак (local root exploit). Информация о безопасном режиме приведена в описании программы **modprobe** (параграф 12.17.1.4 на стр. 406).

Директивы могут повторяться в файле много раз. Отметим, что директивы могут использоваться с префиксом **add**, которая задает добавление нового списка модулей к существующему взамен используемой по умолчанию замены существующего списка новым.

12.17.6.1 Семантика

Ключевое слово **A_PATH** означает полный путь к файлу или каталогу. Путь может содержать мета-символы, включая вывод от других команд (например, **`uname -r`** или **`kernelversion`**).

WORD означает последовательность символов (строку), не содержащую пробелов. Если строка содержит символы кавычек (**' "** или `) , часть строки до закрывающей кавычки (**' "** или `) может содержать любые символы, включая пробелы. Каждый параметр **WORD** преобразуется с использованием мета-символов. Если результат будет содержать несколько слов (последовательность символов без пробелов), использоваться будет только первое слово.

Ключевое слово **compare_op** обозначает операцию сравнения, в качестве которой могут использоваться арифметические и логическое операторы **==**, **!=**, **<**, **<=**, **>=** и **>**.

Ключевое слово **EXPRESSION** означает выражение, в качестве которого могут использоваться записи типа:

`WORD compare_op WORD`
для сравнения символьных строк или

`-n WORD compare_op WORD`
для сравнения числовых значений **WORD**.

`WORD`
если преобразование **WORD** приводит к ошибке или результат равен **0**, **false** или содержит пустую строку, будет возвращаться значение **FALSE**. В остальных случаях выражение будет возвращать значение **TRUE**.

`-f FILENAME`
проверка существования указанного параметром **FILENAME** файла.

`-k`
проверка флага **autoclean**.

`! EXPRESSION`
проверка неверности выражения.

12.17.6.2 Синтаксис

`define VARIABLE WORD`
задает переменную окружения **VARIABLE=WORD**. Созданная переменная доступна всем командам, выполняемым в текущем сеансе.

`depfile=A_PATH`
задает путь к файлу зависимостей, который будет создаваться программой **depmod** (параграф 12.17.2 на стр. 407) и использоваться впоследствии программой **modprobe** (параграф 12.17.1 на стр. 404). Обычно для размещения этого файла используется принятый по умолчанию каталог.

`if EXPRESSION`
Если заданное параметром выражение дает результат **TRUE**, выполняются все директивы, расположенные в строках до ключевого слова **else**, **elseif** или **endif**. Директивы **if** могут быть вложенными (до 20 уровней).

Старайтесь избегать любых условных директив для путей. Интеллектуальных возможностей **modprobe** достаточно, чтобы обойтись без таких директив, а проблем они могут породить много.

`else`
Если выражение в предыдущей директиве **if** или **elseif** дает значение **FALSE**, выполняются все директивы между данной строкой **else** и соответствующей ей строкой **endif**.

`elseif EXPRESSION`
Если выражение в предыдущей директиве **if** или **elseif** дает результат **FALSE**, а выражение для данной директивы возвращает значение **TRUE**, выполняются все директивы до следующего ключевого слова **elseif**, **else** или **endif**.

`endif`
Это ключевое слово завершает структуру, открытую строкой **if**, **elseif** или **else** для условного выполнения директив конфигурационного файла, как показано ниже.

```
if EXPRESSION
    любые конфигурационные директивы
    ...
```

1 *Для этого случая также существуют ограничения - пользователь не может вводить команду **modprobe** как **root**, если задает ей работу со своим (не **/etc/modules.conf**) конфигурационным файлом.*
2 *Включая ядро Linux.*

```
elseif EXPRESSION
    любые конфигурационные директивы
    ...
else
    любые конфигурационные директивы
    ...
endif
```

Директивы **else** и **elseif** не являются обязательными для структур обработки по условию.

include PATH_TO_CONFIG_FILE

в некоторых системах использование одного конфигурационного файла может оказаться неудобным по причине слишком большого размера этого файла. Данная директива позволяет распределить информацию по нескольким файлам. Использование включаемых файлов вместе со структурами обработки по условиям **if** позволяет создавать простые и понятные конфигурации с различными вариантами загрузки модулей в зависимости от тех или иных условий.

insmod_opt=GENERIC_OPTIONS_TO_INSMOD

Если команде **insmod** нужно передать некие опции, не указанные в других местах, эта директива позволяет задать такие опции.

keep

если это ключевое слово встречается в строке до описания пути, последующие описания будут добавляться к используемому по умолчанию набору путей, вместо замены старого набора путей на новый (заданный в конфигурационном файле).

path=A_PATH
path[TAG]=A_PATH

Параметр **A_PATH** этих директив задает дополнительный каталог[1] для поиска модулей. Директива может включать необязательный тег, который содержит дополнительную информацию о назначении модулей добавляемого каталога. Использование таких тегов позволяет автоматизировать некоторые операции, выполняемые с помощью программы modprobe (параграф 12.17.1 на стр. 404). Тег указывается после ключевого слова **path** и помещается в квадратные скобки. Если тег не указан в директиве, предполагается значение **misc**. Использование тегов весьма полезно при загрузке ОС, поскольку позволяет отметить все каталоги, содержащие модули, загружаемые при старте операционной системы.

generic_stringfile=A_PATH

Эта директива указывает путь к файлу **generic_string**, создаваемому программой **depmod** (параграф 12.17.2 на стр. 407) и используемому сценариями установки, которым требуется информация об экспортируемых модулями текстовых строках. Обычно для имени этого файла используется принятое по умолчанию значение (см. стр. 413).

pcimapfile=A_PATH

задает путь к файлу **pcimap**, создаваемому программой **depmod** и используемому сценариями установки для поиска модулей, поддерживающих устройства PCI. Обычно для имени этого файла используется принятое по умолчанию значение (см. стр. 413).

isapnpmapfile=A_PATH

задает путь к файлу **isapnpmap**, создаваемому программой **depmod** и используемому сценариями установки для поиска модулей, поддерживающих устройства ISA PNP. Обычно для имени этого файла используется принятое по умолчанию значение (см. стр. 413).

usbmapfile=A_PATH

задает путь к файлу **usbmap**, создаваемому программой **depmod** и используемому сценариями установки для поиска модулей, поддерживающих устройства USB. Обычно для имени этого файла используется принятое по умолчанию значение (см. стр. 413).

parportmapfile=A_PATH

задает путь к файлу **parportmap**, создаваемому программой **depmod** и используемому сценариями установки для поиска модуля, поддерживающего устройство **parport**[2]. Обычно для имени этого файла используется принятое по умолчанию значение (см. стр. 413).

ieee1394mapfile=A_PATH

задает путь к файлу **ieee1394map**, создаваемому программой **depmod** и используемому сценариями установки для поиска модулей, поддерживающих устройства IEEE 1394. Обычно для имени этого файла используется принятое по умолчанию значение (см. стр. 413).

pnpbiosmapfile=A_PATH

задает путь к файлу **pnpbiosmap**, создаваемому программой **depmod** и используемому сценариями установки для поиска модуля, поддерживающего устройство **pnpbios**. Обычно для имени этого файла используется принятое по умолчанию значение (см. стр. 413).

alias alias_name result

Эта директива позволяет создавать псевдонимы для имен модулей. Например, строка

```
alias iso9660 isofs
```

позволяет использовать команду **modprobe iso9660**, хотя в системе не существует модуля с таким именем.

Отметим, что строки вида

1 *Не забывайте использовать ключевое слово **keep** для сохранения определенных ранее и используемых по умолчанию путей поиска модулей.*
2 *Параллельный порт.*

```
alias <имя модуля> off
```
будет говорить программе **modprobe** о необходимости игнорировать запросы на загрузку указанного в строке модуля. Строка вида

```
alias <имя модуля> null
```
будет приводить к “успешной загрузке никакого модуля”.

Вы можете создавать псевдонимы для псевдонимов со значительной глубиной вложенности (до 1000). Преобразование псевдонимов осуществляется до тех пор, пока не будет определено реальное имя модуля. Ограничение уровня вложенности позволяет предотвратить петли типа показанной ниже:

```
alias a b
alias b a
```
Если за 1000 итераций не будет найдено окончательное имя модуля, **modprobe** будет использовать **probe** и **probeall**. Это обеспечивает использования конструкций типа приведенного ниже примера из реального файла **/etc/devfs**.

```
alias           /dev/sg*        /dev/sg
probeall        /dev/sg         scsi-hosts sg
```
С формальной точки зрения допустимо использование в качестве псевдонимов имен реальных модулей, но лучше избегать этого.

```
[add] probe name module_list
[add] probeall name module_list
```
эти директивы могут использоваться только в тех случаях, когда параметр **name** задает имя модуля, указанного в командной строке **modprobe**. Эффект состоит в том, что при запросе загрузки модуля **name** будут загружаться модули из заданного директивой списка в порядке их расположения в этом списке. Различие между приведенными выше строками состоит в том, что **probe** будет предпринимать попытки до успешной загрузки первого модуля, а **probeall** будет пытаться загрузить все модули из списка. Результат операции определяется результативностью загрузки по крайней мере одного модуля. Необязательный префикс **add** добавляет новый список к имеющемуся вместо замены старого списка новым.

```
prune filename
```
Каталог верхнего уровня с модулями ядра (**/lib/modules/<ядро>**) кроме модулей содержит файлы **modules.dep**, **modules.generic_string**, **modules.pcimap**, **modules.isapnpmap**, **modules.usbmap**, **modules.parportmap**, **modules.ieee1394map**, **modules.pnpbiosmap**, символьную ссылку на каталог исходных текстов ядра **build** и другие файлы. Директива **prune** позволяет избавиться от потока предупреждений "**not an ELF file**" от программы **depmod**. Обычно **depmod** создает список таких файлов, который не следует удалять, поскольку он содержит полный список стандартных файлов такого типа. Если вы храните в каталоге модулей дополнительные файлы, укажите их в качестве параметра директивы **prune** (для каждого файла отдельно). Отметим, что эта директива позволяет исключить только файлы, хранящиеся в каталоге верхнего уровня и никак не влияет на обработку файлов в подкаталогах дерева модулей.

```
[add] options [-k] module [MODULE_SPECIFIC_OPTIONS]
```
Все имена модулей (включая псевдонимы) могут быть связаны со своими наборами опций. Опции, указанные для псевдонима имеют более высокий приоритет. Для разрешения конфликтов при наличии противоречивых опций используется приведенное выше правило. Это правило задает наивысший приоритет для опций командной строки. Опция **-k**, указанная в директиве перед именем модуля, отменяет заданную в командной строке опцию **-k** (autoclean). Необязательный префикс **add** служит для добавления нового списка к существующему вместо замены старого списка новым. Если результатом является псевдоним, а не реальный модуль, все опции, включенные в этот псевдоним, отбрасываются до вызова **probe[all]**. Если любой из параметров **MODULE_SPECIFIC_OPTIONS** содержит мета-символы командного процессора (например, пробелы, кавычки, скобки) эта опция должна быть заключена в кавычки **'"..."'**[1], как показано ниже

```
abc='"def,ghi jkl (xyz)"'
```
Директива

```
[add] above module module_list
```
позволяет модулю поместить другой набор модулей “поверх себя” в стеке модулей[2]. Такая директива полезна в тех случаях, когда реальные зависимости сложнее описанных в файле **modules.dep**. Эта директива является примером оптимизации директив, выполняемых после установки модуля или перед удалением. Отметим, что сбой при попытке загрузки модуля в таких случаях не меняет код возврата **modprobe**. Необязательный префикс **add** служит для добавления нового списка к существующему вместо замены старого списка новым.

Директива

```
[add] below module module_list
```
позволяет модулю “втолкнуть” (push) другой набор модулей ниже себя в стеке модулей. Директива **below** полезна в тех случаях, когда реальные зависимости сложнее описанных в файле **modules.dep**. Эта директива является примером оптимизации директив, выполняемых перед установкой модуля или после его удаления. Отметим, что сбой при попытке загрузки модуля в таких случаях не меняет код возврата **modprobe**. Необязательный префикс **add** служит для добавления нового списка к существующему вместо замены старого списка новым.

Приведенные ниже директивы могут быть полезны для выполнения специфических команд при загрузке или удалении модулей. Отметим, что даже при указании в таких директивах псевдонимов, а не реальных имен, обеспечивает для реального модуля корректный порядок выполнения операций.

1 *Двойные кавычки внутри одинарных. Одинарные кавычки указывают границу опции для файла modules.conf, а двойные передаются командному процессору.*
2 *Для просмотра стека модулей служит команда **lsmod** (параграф 12.17.3 на стр. 408).*

pre-install module command
задает выполнение указанной команды перед установкой модуля.

install module command
задает команду установки модуля, используемую взамен принятой по умолчанию команды **insmod**.

post-install module command
задает выполнение указанной команды после установки модуля.

pre-remove module command
задает выполнение указанной команды перед удалением модуля.

remove module command
задает команду удаления модуля взамен принятой по умолчанию команды **rmmod**.

post-remove module command
задает выполнение указанной команды после удаления модуля.

persistdir=directory_name
Когда программа **rmmod** (параграф 12.17.5 на стр. 409) удаляет модуль, содержащий постоянный набор параметров, она сохраняет эти параметры (возможно, модифицируя их) в каталоге **directory_name**. Когда программа **modprobe** (параграф 12.17.1 на стр. 404) загружает такой модуль, она просматривает каталог **directory_name** в поисках ранее использованных значений параметров. Указываемый в этой директиве каталог должен быть доступен во время загрузки модуля; по умолчанию параметры записываются в каталог **/var/lib/modules/persist**. Многие дистрибутивы Linux загружают модули до монтирования файловых систем и это может вызывать проблемы, если указанный каталог находится на несмонтированном еще разделе диска. Если **/var** является отдельным разделом и монтируется после загрузки модулей, **insmod** не сможет прочитать параметры, сохраненные в принятом по умолчанию каталоге. Для решения этой проблемы существует два варианта.

1) Поместить каталог в корневом разделе (например, **/lib/modules/persist**). Это требует возможности записи в корневой раздел при работе программы **rmmod**.

2) Загрузить модули связанные с файловыми системами, смонтировать раздел **/var** и после этого перейти к загрузке остальных модулей. Предполагается, что модули для работы с файловыми системами не используют сохраненных параметров.

12.17.6.3 Используемая по умолчанию конфигурация

Если файл **/etc/modules.conf** отсутствует и не указан другой конфигурационный файл, используется перечисленный ниже набор директив загрузки модулей:

```
depfile=/lib/modules/`uname -r`/modules.dep
generic_stringfile=/lib/modules/`uname -r`/modules.generic_string
pcimapfile=/lib/modules/`uname -r`/modules.pcimap
isapnpmapfile=/lib/modules/`uname -r`/modules.isapnpmap
usbmapfile=/lib/modules/`uname -r`/modules.usbmap
parportmapfile=/lib/modules/`uname -r`/modules.parportmap
ieee1394mapfile=/lib/modules/`uname -r`/modules.ieee1394map
pnpbiosmapfile=/lib/modules/`uname -r`/modules.pnpbiosmap

path[boot]=/lib/modules/boot
path[toplevel]=/lib/modules/`uname -r`
path[toplevel]=/lib/modules/`kernelversion`
path[toplevel]=/lib/modules/default
path[toplevel]=/lib/modules

persistdir=/var/lib/modules/persist
```

Существует также набор используемых по умолчанию псевдонимов и опций. Поскольку эти наборы постоянно обновляются, нет смысла приводить их здесь. Вы можете увидеть этот набор с помощью команды **modprobe -c** при пустом файлк **/etc/modules.conf**.

12.17.6.4 Старый конфигурационный файл

В силу исторических причин при отсутствии файла **/etc/modules.conf** работающие с модулями утилиты будут читать файл **/etc/conf.modules**. Однако использование этого файла не рекомендуется и его следует заменить файлом **/etc/modules.conf**.

12.18 Конфигурационный файл lilo.conf

Файл **lilo.conf** управляет работой загрузчика LILO (параграф 2.1.2.1 на стр. 33).

В таблице 143 приведено описание глобальных параметров, а таблицы 144 и 145 содержат описание частных параметров конфигурации LILO. Отметим, что некоторые параметры (например, vga) могут включаться как в общий раздел, так и в некоторые частные разделы файла lilo.conf.

Кроме того, существует также группа параметров ядра, которые передаются ядру перед его загрузкой. Эти параметры перечислены в таблице 146.

Таблица 143. Глобальные параметры загрузчика LILO.

Параметр	Описание
backup=<имя>	Указывает имя для создания резервной копии загрузочного сектора. В качестве имени может использоваться: 1. каталог, в котором создается резервный файл boot.NNNN; 2. шаблон полного имени файла, к которому будет добавляться суффикс .NNNN; 3. полное имя файла с корректным суффиксом NNNN. При использовании устройств RAID должны применяться только два первых варианта, поскольку может создаваться несколько копий загрузочного сектора. Суффикс .NNNN является шестнадцатеричным представлением старшего (major) и младшего (minor) номера устройства или раздела. Если эта опция не используется, резервная копия загрузочного сектора записывается в файл /boot/boot.NNNN. При наличии такого файла новая копия создаваться не будет. См. также описание опции **force-backup**, приведенное ниже (стр. 416)
bios-passes-dl=yes|no|unknown	Эта экспериментальная опция показывает, передает ли BIOS код текущего загрузочного устройства в регистр DL. По умолчанию используется значение **unknown**.
bitmap=<имя файла>	Позволяет указать растровый файл (640x480x16), используемый в качестве фона для вывода меню загрузки. Недопустимо использование этой опции совместно с описанной ниже опцией **message**. При задании фонового изображения будет использоваться графический интерфейс меню загрузки, если иное не задано опцией **install** (см. ниже).
bmp-colors=<fg>,<bg>,<sh>, <hfg>,<hbg>,<hsh>	Задает десятичные значения цветов, используемые для вывода меню на фоне растра, заданного параметром **bitmap**. Список цветов содержит 6 элементов - первые 3 задают цвета нормального текста, а остальные - цвета текста выбранной строки (указателя). В каждой триаде первое значение указывает цвет переднего плана, второе - цвет фона и третье - цвет тени. Если цвет фона не указан, используется прозрачный фон. Если не указан цвет тени, символы выводятся без теней. Элементы списка разделяются запятыми без пробелов.
bmp-table=<x>,<y>, <ncol>,<nrow>,<xsep>,<spill>	Указывает положение меню на экране и задает схему меню. Параметры **x**, **y** указывают координаты верхнего левого угла меню (x может принимать значения от 1 до 80, а y - от 1 до 30). Параметр **ncol** определяет число колонок в меню (1 - 5), **nrow** - число строк (вариантов загрузки) в каждой колонке. Если указано несколько колонок, параметр **xsep** задает расстояние (число символов) между левыми краями колонок (18 - 40), а **spill** - число записей, которое должно быть помещено в колонку до перехода в следующую колонку. Значение **spill** не должно превышать **nrow**. При использовании пиксельных координат, значения **x**, **y** и **xsep** должны указываться с суффиксом **p**.
bmp-timer=<x>,<y>, <fg>,<bg>,<sh>	Эта опция указывает координаты вывода значения таймера ожидания (см. параметр **timeout)**. Параметры **x** и **y** задают (в символах или пикселях) координаты поля вывода таймера (см. описание опции **bmp-table**), а следующие 3 параметра определяют цвет для символов таймера (см. описание опции **bmp-colors**). В отличие от **bmp-colors** для данной опции задание фонового цвета обязательно. Если вы воспользуетесь опцией **bmp-timer = none**, таймер не будет выводиться на экран.
boot=<загрузочное устройство>	Указывает диск, на котором размещается менеджер загрузки. В приведенном примере загрузочным диском является /dev/hda. Если эта опция не используется, загрузочный сектор будет считываться[1] с устройства, на котором смонтирован корневой раздел. При использовании RAID в качестве загрузочного указывается первое устройство (например, boot=/dev/md0).

1 *И возможно записываться.*

Параметр	Описание

change-rules	Определяет необходимость изменения значения типа раздела во время загрузки (в целях сокрытия).
	`change-rules`
	`  reset`
	`  type=DOS12`
	`  normal=1`
	`  hidden=0x11`
	`  type=DOS16_small`
	`  normal=4`
	`  hidden=0x14`
	`  type=DOS16_big`
	`  normal=0x06`
	`  hidden=0x16`
	Первая строка опций приведенного выше правила сбрасывает все существующие правила (reset), а после этого определяются новые правила для 3 указанных разделов. Без сброса новые правилы были бы просто добавлены к используемым по умолчанию правилам, которых обычно бывает вполне достаточно. Строки, определяющие типы разделов используются в секции **change** (стр. 420) с суффиксами **_normal** или **_hidden**. Дополнительную информацию об изменении типов вы сможете найти в разделе "Partition type change rules" руководства пользователя из комплекта документации lilo[1].

compact	При наличии этой опции будут предприниматься попытки объединения запросов на чтение соседних секторов в один запрос. Такое объединение операций чтения существенно снижает время загрузки и размеры файла отображения (map). Настоятельно рекомендуется использовать эту опцию для загрузки с компакт-дисков.
default=”метка варианта”	Указывает секцию файла lilo.conf (метку), которая описывает используемый по умолчанию вариант загрузки ОС. Заданный этой меткой образ загружается в том случае, когда пользователь не выбрал варианта загрузки в течение заданного параметром времени. Если используемый по умолчанию вариант не указан, загружается вариант из первой секции конфигурационного файла.
delay=<время задержки>	Задает количество десятых долей секунды (100 мсек), в течение которого менеджер загрузки ожидает выбора варианта загрузки. По истечении этого времени загружается образ, заданный опцией командной строки **-R** или параметром **default**, или образ из первой секции конфигурационного файла. Если параметр не указан или задано нулевое значение, загрузка происходит без задержки. В приведенном выше примере на выбор варианта загрузки отводится 5 секунд (50/10). Действие этого параметра может быть изменено опцией **prompt** (см. стр. 417).

disk=<имя устройства>	Эта опция позволяет задать для указанного диска нестандартный набор параметров[2]. Особенно полезен параметр **bios**. Система BIOS нумерует диски 0x80, 0x81 и т. д., что никак не связано с используемыми в Linux идентификаторами дисков. Для обеспечения такой связи вы можете использовать строки
	`disk=/dev/sda`
	`  bios=0x80`
	`disk=/dev/hda`
	`  bios=0x81`
	которые говорят, что ваш SCSI-диск является первым диском BIOS, а ваш диск IDE (primary master) - вторым диском BIOS. Можно также указать для диска параметры геометрии:
	`disk=/dev/fd0`
	`  sectors=18`
	`  heads=2`
	`  cylinders=80`
	В тех случаях, когда для устройства нет информации о разделах (например, для loopback-устройств), опция **disk** позволяет явно указать такую информацию:
	`disk=/dev/loop0`
	`  bios=0x80`
	`  partition=/dev/loop1`
	`  start=2048  # смещение от сектора 0`
	`  partition=/dev/loop2`
	`  start=102400  # смещение от сектора 0`

1 *Копии файлов документации вы сможете найти в каталоге /Documents/LILO приложенного к курсу компакт-диска.*

2 *См. раздел Disk geometry в руководстве пользователя lilo, которое имеется в каталоге /Documents/LILO/ приложенного к книге компакт-диска.*

Параметр	Описание
disktab=<disktab-файл>	Задает имя файла с параметрами дисков[2]. Если этот параметр не указан, используется файл **/etc/disktab**.
el-torito-bootable-CD	Флаг второго этапа загрузки для прерывания эмуляции дисковода при реальной загрузке с компакт-диска El Torito. Эта опция применяется только утилитой **mkrescue** при запуске с ключом **--iso**.
fix-table	Эта опция позволяет корректировать 3D-адреса в таблице разделов. Каждая запись таблицы содержит адрес 3D (цилиндр/головка/сектор) и линейный адрес для первого и последнего сектора раздела. Если граница раздела не совпадает с границей дорожки, в некоторых ОС (например, PC/MS-DOS и OS/2) используется подстройка 3D-адресов. Программа lilo может сохранять загрузочный сектор только на тех разделах, для которых оба типа адресов совпадают (указывают на один сектор). С помощью опции **fix-table** lilo может корректировать 3D-адреса[3].
force-backup=<имя>	Эта опция аналогична описанной выше опции **backup** (стр. 414) и отличается лишь безусловным переписыванием существующего файла.
geometric	Переключает на использование режима адресации, совместимого со старыми версиями LILO. Геометрическая адресация использует триады цилиндр/головка/сектор, причем нумерация цилиндров ограничена значением 1023. если будет указан некорректный номер цилиндра, система диагностики обнаружит это скорей при инсталляции, чем при загрузке. С новыми версиями BIOS рекомендуется использовать адресацию lba32 (см. ниже).
ignore-table	Говорит lilo о необходимости игнорировать поврежденную таблицу разделов.
install=text|menu|bmp	Задает пользовательский интерфейс для этапа загрузки ОС. Традиционно в LILO использовался текстовый интерфейс, но в современных версиях по умолчанию устанавливается режим **menu**, если параметром **bitmap** (стр. 414) не задано растровое изображение. Текстовый интерфейс требует ввода варианта загрузки с клавиатуры, меню позволяет организовать выбор с помощью текстового меню, bmp обеспечивает графический экран выбора с использованием хранящегося в файле растрового изображения 640x480 (16 или 256 цветов).
large-memory	Обычно создаваемый при загрузке виртуальный диск (initrd) помещается в старшие адреса памяти, но не выше границы 15 Мбайт. Это ограничение обусловлено системой BIOS в старых ПК. Новые компьютеры позволяют использовать память в более высоких адресах (вплоть до связанного с ядром ограничения 768 Мбайт) для initrd. Присутствие этой опции просто указывает, что ваша версия BIOS не имеет описанного выше ограничения. Опция не передается ядру и не оказывает никакого влияния на использование физической памяти компьютера.
lba32	Опция задает использование 32-битовых адресов LBA взамен адресации цилинтр/головка/сектор. Если BIOS поддерживает пакетную адресацию, можно будет организовать загрузку с любого раздела диска даже при числе цилиндров более 1024. Если пакетная адресация не поддерживается системой BIOS адреса lba32 транслируются в геометрические адреса (см. опцию geometric), как для опции linear (см. ниже). Для всех дискет используется адресация C:H:S. Опцию lba32 следует использовать на всех компьютерах, выпущенных после 1998 года.
linear	Используется 24-битовая линейная адресация секторов взамен формата цилиндр/головка/сектор (см. geometric на стр. 416). Линейные адреса преобразуются “на лету” в геометрические и для этой схемы адресации присутствует ограничение на число цилиндров (не более 1023). При попытке использования опции **linear** для дисков с недопустимым числом цилиндров программа /sbin/lilo может сообщать о недопустимости выбранной адресации. Для дисков с большим числом цилиндров служит опция lba32 (стр. 416), но для ее использования требуются версии BIOS, выпущенные после 1998 года.
lock	Включает автоматическую запись команд загрузки для повторного их использования при следующей загрузке. Таким образом lilo "блокирует" выбранный вариант, пока эта опция не будет удалена вручную.
mandatory	Задает для каждого варианта обязательность использования пароля (см. опцию **password** на стр. 419).

2 *Не рекомендуется использовать эту опцию.*
3 *Нет никакой гарантии, что другая ОС потом не проведет обратной корректировки. Поэтому старайтесь создавать разделы так, чтобы они включали целое число дорожек.*

Параметр	Описание
map=<map-файл>	Задает местоположение map-файла (по умолчанию **/boot/map**). На машинах с BIOS до 1998 года может не присутствовать расширение EDD, требуемое для использования адресации lba32 (см. стр. 416). В этом случае загрузчик будет автоматически переключаться на геометрическую адресацию секторов (см. опцию geometric на стр. 416). использование режимов **geometric** и **linear** (стр. 416) требует размещения map-файла на первых 1024 цилиндрах диска. Это ограничение не распространяется на системы с поддержкой EDD в BIOS.
menu-title=<title-string>	Задает заголовок (до 37 символов) для меню загрузки. Этот заголовок используется взамен принятой по умолчанию строки "LILO Boot Menu". Если загрузчик не использует меню (см. **install** на стр. 416), эта опция не имеет смысла.
menu-scheme=<color-scheme>	Эта опция позволяет изменить принятую по умолчанию цветовую схему меню для мониторов VGA[1]. Строка задания цветов имеет вид: **<текст>:<выбранная строка>:<рамка>:<заголовок>** Каждый элемент включает два символа - для переднего плана и фонового цвета. Обязательным является только первый элемент. По умолчанию выбранная строка отображается в инверсных цветах, а рамка и заголовок выводятся цветом текста. Для задания цветов используются символы **kbgcrmyw** (blacK - черный, Blue - синий, Green - зеленый, Cyan - бирюзовый, Red - красный, Magenta - малиновый, Yellow - желтый, White - белый). Заглавные буквы в обозначении цвета переднего плана позволяют задать повышение интенсивности. Ниже приведены примеры возможных цветовых схем: **menu-scheme=Wm** - интенсивный белый цвет на малиновом фоне **menu-scheme=wr:bw:wr:Yr** - используемая по умолчанию схема LILO **menu-scheme=Yk:kw** - Ярко-желтые буквы на черном фоне. Если загрузчик не использует меню (см. **install** на стр. 416), эта опция не имеет смысла.
message=<файл>	Указывает текстовый файл, содержащий сообщение, которое будет выводиться на экран перед приглашением к загрузке системы. Сообщение не выводится на экран пока не будет нажата какая-либо клавиша после появления на экране строки **LILO**. Включение в файл символа **FF** (Ctrl L) приведет к очистке экрана (нежелательно применять очистку экрана при использовании загрузочного меню). Размер файла с тестовым сообщением может достигать 65535 байтов. При изменении или удалении файла с сообщением перестраивается map-файл. Опции **message** и **bitmap** (стр. 414) не могут использоваться одновременно.
nowarn	Отключает вывод предупреждений о возможных проблемах.
optional	Использовать режим **optional** (стр. 419) для всех загрузочных образов.
password=<пароль>	Использовать режим **password** (стр. 419) для всех загрузочных образов. Использование пароля блокирует автоматическую загрузку системы, если для принятого по умолчанию варианта загрузки опция **password** защищена используемым по умолчанию уровнем **mandatory** (), который сильнее уровня **restricted** ().
prompt	Автоматическая загрузка (см. опцию **delay** на стр. 415) не будет выполняться без пользовательского ввода, если в командной строке не использовалась опция отмены (**lilo -R**). Загрузчик будет выводить на экран приглашение и ждать пользовательского выбора (см. опцию timeout на стр. 418). Если вы используете опцию **prompt**, не указав значения **timeout** или с защищенной паролем (с уровнем выше **restricted**) загрузкой принятого по умолчанию образа, автоматической загрузки системы происходить не будет.

1 *Для мониторов MGA схема цветов не меняется.*

Параметр	Описание
raid-extra-boot=<option>	Эта опция имеет смысл только для систем RAID1. Поле <**option**> может содержать значения **none**, **auto**, **mbr**, **mbr-only** или разделенный запятыми[1] список устройств (например, **/dev/hda,/dev/hdc6**). Начиная с версии LILO v22.0, загрузочная запись обычно хранится в первом секторе раздела RAID1. Для параллельных систем RAID не требуется других загрузочных записей. По умолчанию используется режим **auto**, при котором автоматически генерируется дополнительная загрузочная запись, требуемая для SKEWED-систем RAID. Режим **none** подавляет автоматическую генерацию дополнительных записей, **mbr-only** подавляет генерацию загрузочной записи на устройстве RAID и обеспечивает совместимость с версиями LILO до 22.0 путем помещения загрузочных записей в MBR (Master Boot Record) всех дисков, имеющих разделы в массиве RAID. Режим **mbr** подобен **mbr-only**, но не подавляет сохранение загрузочной записи в разделе RAID. Указание явного списка устройств приводит к размещению дополнительных загрузочных записей на каждом из указанных устройств в дополнение к записи на устройстве RAID1. Поскольку с версии 22 коды RAID1 никогда не будут автоматически помещать загрузочную запись в MBR устройства 0x80, явное указание списка устройств является одним из способов организации такой записи. Другой способ обеспечивает режим **mbr**.
restricted	Опция парольной защиты **restricted** (см. стр. 419) применяется для всех образов.
serial=<пар>	Включает управление загрузкой по последовательному каналу. Указанный последовательный порт инициализируется и загрузчик пытается читать из него данные. Если последовательное соединение не обеспечивает достаточного уровня безопасности (например, модемное соединение через сеть общего пользования), все варианты загрузки должны быть защищены паролями. Строка параметров опции имеет формат: **<порт>[,<скорость>[<четность>[<слово>]]]** <**порт**> - номер последовательного порта; 0 соответствует порту COM1 (/dev/ttyS0). Программа может работать с любым из портов COM1 - COM4. <**скорость**> - скорость последовательного порта в бит/с; поддерживаются скорости 110, 150, 300, 600, 1200, 2400 (по умолчанию), 4800, 9600, 19200, 38400 и 57600 (56000); скорость 115200 также можно использовать, но не все порты ее поддерживают. <**четность**> - режим контроля четности, используемый для последовательной линии; допустимы значение **n** (или **N**) для работы без контроля четности, **e** (или **E**) для проверки на четность и **o** (или **O**) для контроля на нечетность. <**слово**> - размер слова данных; поддерживаются значения 7 и 8; по умолчанию используются 8-битовые слова при работе без контроля четности и 7-битовые для режимов контроля **e** и **o**. При использовании опции **serial** значение параметра delay (стр. 415) автоматически увеличивается до 20.
single-key	Эта опция позволяет выбирать варианты загрузки с помощью одной клавиши, основываясь на первой букве имени соответствующей метки[2]. Опцию не следует использовать при выборе варианта загрузки с помощью меню или графического интерфейса (опция install, стр. 416).
suppress-boot-time-BIOS-data	Эта глобальная опция подавляет во время загрузки сбор данных BIOS, поскольку на некоторых системах это может приводить к зависанию. Опция эквивалентна использованию переключателя **nobd** во время загрузки. Эта опция отключает распознавание меток дисков и кодов устройств BIOS на системах, имеющих более одного диска. Поэтому при ее использовании в таких системах на экран будут выдаваться предупреждения, которые невозможно подавить.
timeout=<tsecs>	Задает тайм-аут (в десятых долях секунды) для ожидания выбора пользователя при загрузке. Значение этого параметра имеет смысл только при использовании опции **prompt** (стр. 417). Если в течение заданного времени не будет нажато никакой клавиши, автоматически будет загружен принятый по умолчанию вариант. Используемое по умолчанию значение тайм-аута бесконечно.
verbose=<уровень>	Задает объем выводимой при загрузке информации Максимальный объем обеспечивается при значении 5.

1 *Без пробелов.*
2 *В этом случае первые символы меток для всех вариантов должны быть уникальными.*

Параметр	Описание
vmdefault=<имя>	Указанный именем образ используется по умолчанию, если загрузка происходит в “виртуальном” режиме с использованием виртуального монитора (например, VMware[tm]. Таким образом для реального и виртуального режимов загрузки могут использоваться по умолчанию различные варианты.

В дополнение к перечисленным параметрам в секции глобальных параметров могут использоваться параметры конфигурации ядра (см. таблицу 146) **append**, **ramdisk**, **read-only**, **read-write**, **root** и **vga**. Эти значения используются, если в выбранной для загрузки секции не указаны соответствующие параметры конфигурации ядра.

Каждая секция варианта загрузки (частные параметры) начинается со строки

```
image=<pathname>
```

указывающей файл или устройство, содержащее загружаемый образ ядра Linux, или строки

```
other=<device>
```

указывающей загрузку другой операционной системы.

В первом случае, если параметр **<pathname>** задает загрузку с устройства, он должен указывать используемый для отображения диапазон секторов

```
range=<начало>-<конец>
range=<начало>+<число секторов>
range=<сектор>
```

В последнем варианте предполагается загрузка с одного сектора.

Для случаев **image** и **other** могут использоваться перечисленные в таблице 144 опции.

Таблица 144. Частные параметры LILO.

Параметр	Описание
label=<метка>	Загрузчик будет использовать образ ядра, указанный в секции, заданной меткой.
alias=<псевдоним>	Дополнительное имя секции загрузки.
lock	Описание опции **lock** приведено в таблице глобальных параметров (стр. 416)
optional	Задает пропускание загрузочного образа, если он недоступен в момент создания map-файла. Такое поведение может быть задано как глобальная опция. Данная опция позволяет указывать в конфигурационном файле секции загрузки тестовых ядер, которые не всегда присутствуют в системе.
password=<пароль>	Задает парольную защиту секции **image=** или **other=**. Парольная защита может быть указана как глобальная опция (см. стр. 417). Интерпретация параметра **password=** может изменяться описанными ниже параметрами **mandatory**, **restricted** и **bypass**. Пароль можно записать в конфигурационный файл (недостаточно безопасно) или вводить при установке загрузчика. Для запроса на ввод пароля следует использовать форму **password=""**. Запрашиваемые в интерактивном режиме пароли не требуется вводить снова при повторном запуске установки загрузчика. Пароль кэшируется в hash-форме вместе с конфигурационным файлом (по умолчанию **/etc/lilo.conf.crc**). При обновлении конфигурационного файла выдается предупреждение о необходимости использования команды **lilo -p** для обновления парольного кэш-файла.
mandatory	Задает необходимость использования пароля для секции загрузки (принято по умолчанию). Опция может использоваться в секциях **image=** и **other=** для изменения глобальной установки.
restricted	Пароль требуется для загрузочной секции только в тех случаях, когда в командной строке задаются параметры ядра (например, **single**). Опция может использоваться в секциях **image=** и **other=** для изменения глобальной установки.
bypass	Пароль для загрузочной секции не требуется. Эта опция показывает, что заданный глобально пароль не применяется для данной секции **image=** или **other=**.
vmwarn	При загрузке под управлением виртуального монитора (например, Vmware[tm]) образ с такой меткой будет выдавать предупреждение для привлечения внимания пользователя.
vmdisable	При загрузке под управлением виртуального монитора образ с такой меткой не будет выводиться в списке опций загрузки. Образы с такими метками можно загружать только в реальном режиме (см. описание **vmdefault** на стр. 419).

Секции **other** используются для загрузки операционных систем, отличных от Linux. Строка

```
other = <device>
```

задает загрузочный сектор альтернативной системы, содержащейся на устройстве или разделе диска (например, для случая DOS можно указать загрузку с раздела **/dev/hda2** или дискеты **/dev/fd0**).

Опции загрузки другиз операционных систем перечислены в таблице 145.

Таблица 145 Частные параметры LILO для загрузки других ОС

Параметр	Описание
loader=<chain-загрузчик>	Эта опция задает используемый chain-загрузчик[1]. По умолчанию используется загрузчик **/boot/chain.b**, который передает информацию об устройстве и разделе в загрузочный сектор и служит только для загрузки DOS (FAT12 и FAT16), Windows (FAT16 и FAT32) или OS/2 (FAT16 и HPFS). Альтернативный загрузчик **os2_d** передает информацию без перечисленных выше ограничений, и использует формат, приемлемый для OS/2 и DOS (см. описание опции table=<letter>, приведенное ниже).
table=<устройство>	Задает устройство, содержащее таблицу разделов. Загрузчик будет передавать загружаемой ОС информацию об используемом по умолчанию разделе, если этот параметр не задан. Некоторые ОС используют другие способы определения раздела, с которого следует загружать операционную систему. Например, MS-DOS обычно хранит геометрию загрузочного диска или раздела в своем загрузочном секторе. Отметим, что при изменении значения параметра **table** следует заново ввести команду **/sbin/lilo** для учета загрузчиком внесенных изменений.
table=<имя диска>	Эта опция используется загрузчиком **os2_d** и задает принятое в DOS односимвольное имя раздела, с которого будет загружаться ОС. Использование этого параметра обязательно для случаев загрузки OS/2, установленной в расширенном разделе. Имя диска можно указывать без двоеточия.

change	Это ключевое слово указывает начало секции, описывающей изменение идентификатора первичного раздела (primary partition) и состояния активности для него. Если эта опция не используется, правила изменения генерируются как в режиме **automatic**. Ключевое слово **change** без указания правил замены будет отключать автоматическое изменение. Например,
	
	`other=/dev/hda2`
	`label=dos`
	`table=/dev/hda`
	`change`
	`automatic`
	`partition=/dev/hda1`
	`set=DOS12_hidden`
	`deactivate`
	`partition=/dev/hda2`
	`set=DOS16_big_normal`
	`activate`
	
	задает, что при загрузке с первичного раздела **/dev/hda2** работают правила автоматического изменения и раздел 1 (DOS12) прячется и деактивируется, а раздел 2 (DOS16) открывается и активизируется. Активизация устанавливает для раздела флаг загрузки. Ключевое слово **automatic** может вызывать конфликты с принятыми по умолчанию правилами изменения, поэтому для гарантии заданы строки **set=**.

boot-as=<bios>	Задает код BIOS, который должен быть присвоен устройству для того, чтобы загружалась ОС из секции **other=**. Если chain-загрузчик видит другой код BIOS для этого диска, он будет динамически заменять код указанным значением. Эта опция проще, нежели **map-drive=** и более универсальна, чем **master-boot**, поскольку позволяет задать код любого устройства. В отличие от **map-drive=** определение необходимости смены кода происходит во время загрузки, а не при инсталляции. Такое решение более приемлемо для систем, в которых BIOS присутствует в меню загрузки и коды дисков могут меняться в зависимости от выбранных опций BIOS. Эта опция может быть указана в глобальном разделе и будет применяться ко всем секциям **other=**, если иной вариант не задан с помощью **master-boot**. Если в глобальном разделе должна быть задана опция **boot-as=** или **master-boot**, второй вариант предпочтительней, поскольку он не вызывает конфликтов с кодами BIOS для дисководов; опцию **boot-as=** в этом случае можно использовать в тех секциях, где следует отменить действие **master-boot**.

1 *Этот загрузчик может также указываться в секции глобальных параметров.*

Параметр	Описание
master-boot	Этот флаг указывает, что DOS/Windows/OS2 или иная ОС будет загружаться только с устройства с BIOS-кодом 0x80 (диск C:) или 0 (дисковод A:). если этот флаг установлен и загрузочное устройство имеет другой код BIOS, chain-загрузчик будет динамически переключать значения кодов между данным устройством и диском, в действительности имеющим код 0x80 или 0, чтобы этот диск появлялся в системе как C: или A:. Этот флаг проще в использовании, нежели опция **map-drive=** (см. ниже) и лучше применять его, если требуется только изменение кода диска. Кроме того, при использовании этого флага необходимость переключения кода BIOS определяется динамически во время загрузки, а не задается статически при установке загрузчика, как это происходит при использовании опции **map-drive=**. Возможность динамического переключения кодов для устройств 0 и 0x80 делает эту опцию более мощной по сравнению с флагом **boot-as=**. Эту опцию можно задавать в глобальном разделе, а в секциях **other=** (при необходимости) использовать **boot-as=**.

map-drive=<num>	Отображает вызовы BIOS для заданного устройства на устройство с кодом, указанным в следующей строке **to=<num>**. Такое отображение полезно призагрузке некоторых ОС (например, DOS) со второго диска. Соответственно изменяются и имена дисков (C: и D:),
	
	`map-drive=0x80`
	`  to=0x81`
	`map-drive=0x81`
	`  to=0x80`
	Эта опция практически утратила смысл после появления опции **boot-as=** в LILO v22.5.

unsafe	Этот флаг блокирует доступ к загрузочному сектору на время генерации отображения (map). Этот запрет отключает некоторые проверки, включая тестирование таблицы разделов. Если загрузочный сектор находится на дискете, данная опция избавляет от необходимости установки дискеты в дисковод во время генерации map-файла. Если загрузочный сектор находится на винчестере, BIOS-код диска будет явно указан в конфигурационном файле строкой типа **disk=/dev/XXXX bios=0x8X inaccessible**. Опцию **unsafe** нельзя использовать вместе с опцией **table** (явной или неявной).

При загрузке Linux ядру ОС могут передаваться дополнительные параметры с помощью опций, перечисленных в таблице 146. Эти опции могут указываться в глобальном разделе или в отдельных секциях.

Таблица 146. Параметры ядра, передаваемые через LILO.

Параметр	Описание
append=<параметры>	Добавляет строку параметров к параметрам, передаваемым при загрузке ядру Linux. Обычно эта опция используется для передачи параметров оборудования, которое не может быть определено автоматически, или при определении которого могут возникнуть проблемы. Отдельные параметры ядра разделяются пробелами и вся строка параметров заключается в двойные кавычки. Опцию **append** можно использовать в каждой секции **image=** только один раз.
initrd=<имя>	Задает образ стартового RAM-диска, который будет загружаться с ядром. Этот образ должен содержать модули, которые требуются во время загрузки (например, драйверы устройств scsi и сетевых адаптеров). Для создания образа служит команда **mkinitrd**.
literal=<параметры>	Эта опция похожа на **append**, но заданные параметры не добавляются к имеющимся параметрам ядра, а используются вместо них. Эту опцию небезопасно использовать в глобальном разделе, поскольку ошибка в параметрах может сделать систему неработоспособной для всех вариантов загрузки.
ramdisk=<размер>	Задает размер используемого RAM-диска. Нулевое значение говорит об отказе от использования виртуального диска во время загрузки. Если эта опция опущена, размер RAM-диска определяется параметрами компиляции ядра.
read-only	Эта опция указывает, что корневой раздел должен быть смонтирован с доступом только для чтения[1]. Опция может быть задана как глобальная.
read-write	Задает монтирование корневого раздела с доступом для чтения и записи. Опция может быть задана как глобальная.
root=<root-device>	Задает устройство (раздел), которое должно монтироваться в качестве корневого. Опция может быть задана в глобальном разделе. Если в качестве имени указано ключевое слово **current**, в качестве корневого указывается устройство, на котором корневой раздел смонтирован в настоящий момент. Если опция **root** не задана, используется корневой раздел, заданный в ядре (опция ROOT_DEV при компиляции ядра)[2].

1 *Обычно на заключительных этапах загрузки корневой раздел монтируется заново с обеспечением возможности записи на него.*

2 *Этот раздел впоследствии можно сменить с помощью команды rdev.*

Параметр	Описание
vga=<режим>	Задает текстовый режим VGA, который должен быть выбран для загрузки. Параметр может быть указан в глобальном разделе. Поддерживаются следующие режимы: **normal** - 80x25; **extended** (**ext**) - 80x50; **ask** - загрузка останавливается до момента выбора режима пользователем; **<значение>** - идентификатор желаемого текстового режима[1]. Если режим не указан, используются параметры, заданные при компиляции ядра (опция SVGA_MODE) или установленные с помощью команды **rdev**.

12.19 Конфигурационный файл ipsec.conf

Конфигурационный файл ipsec.conf содержит большинство параметров, определяющих работу подсистемы FreeS/WAN IPsec. Если в файле не используются созданные вручную ключи, раскрытие содержимого этого никак не влияет на состояние безопасности. Конфигурационный файл использует текстовый формат и состоит из одной или нескольких секций. Строки, начинающиеся с символа #, используются для комментариев, пустые строки файла не принимаются во внимание.

Строка, содержащая ключевое слово **include**, за которым следует имя файла, заменяется в процесс работы реальным содержимым этого файла. Если имя файла не включает полного пути к нему, поиск этого файла ведется от каталога, в котором хранится файл ipsec.conf. Допускается многоуровневое включение файлов друг в друга, а имена включаемых файлов могут содержать символы-шаблоны (например, include ipsec.*.conf).

Использование включаемых файлов позволяет выделить в отдельные файлы информацию о соединениях. Файлы с информацией о соединениях могут использоваться в неизменном виде на нескольких шлюзах. Такая возможность сильно упрощает настройку систем со множеством шлюзов VPN. Отметим также, что параметры **also** и **alsoflip** позволяют расщеплять логические секции конфигурационного файла (например, раздел описания соединений) на несколько отдельных секций.

Первая значимая строка конфигурационного файла должна содержать сведения о версии спецификации, которой файл соответствует:

```
version 2
```

Секции файла начинаются с строк вида:

```
type name
```

где type определяет тип секции, а name - имя, позволяющее разделять между собой однотипные секции. **Имена секций должны начинаться с латинской буквы и могут содержать только буквы, цифры, точки (.), знаки подчеркивания (_) и дефис (-). Все последующие непустые строки секции, которые начинаются с пробела, являются частью этой секции. Строки комментариев внутри секции также должны начинаться с пробела. Не допускается совпадение имен секций.**

Строки внутри секции обычно имеют вид[2]

```
parameter=value
```

Имена параметров используют такой же синтаксис, как имена секций. Если явно не указано иное, каждый параметр может появляться в секции не более одного раза.

Опущенные значения параметров заменяются принятыми по умолчанию значениями. Поле значения может содержать пробелы только в том случае, когда оно целиком заключено в двойные кавычки ("). Значение не может включать более одной строки и не должно содержать символов двойных кавычек.

Числовые значения могут быть целыми (только цифры) либо десятичными дробями (две последовательности цифр, разделенные точкой).

Параметр

```
also
```

может использоваться в любой секции. В качестве значения этого параметра могут использоваться имена секций. Указанная параметром секция должна существовать, располагаться после текущей секции и относиться к тому же типу секций. Допускается использование вложенных секций, но нельзя присоединять (append) одну секцию несколько раз. Такая организация позволяет, например, сохранять ключи шифрования для соединений в отдельном файле, указывая его с помощью параметра **also** или строки **include**[3].

Параметр

```
alsoflip
```

может использоваться в секции conn (параграф 423 на стр. 423). Он действует подобно **also**, но меняя местами правые и левые части записей указанной секции.

Имена параметров, начинающиеся с **x-**, **X-**, **x_** или **X_**, зарезервированы для пользовательских расширений и

1 *Список поддерживаемых вашей системой режимов можно получить при нажатии клавиши **Enter** в случае **vga=ask**.*
2 *Напомним, что каждая строка внутри секции должна начинаться с пробела. Допускается использование пробелов с любой стороны от знака равенства (parameter = value)*
3 *В параграфе 6.1.3 (стр. 157) рассматриваются некоторые ограничения.*

никогда не используются приложениями IPsec. Параметры с такими именами должны использовать такой же синтаксис, как в именах типов. Все остальные имена зарезервированы для будущего развития IPsec.

Секция с именем **%default** задает принятые по умолчанию параметры для секций того же типа. Допускается использование нескольких секций **%default** одного типа, но каждый параметр в них может задаваться только один раз. Секции **%default** должны предшествовать однотипным секциям с конкретными параметрами. Использование параметров **also** и **alsoflip** в секциях **%default** не допускается.

В настоящее время определено два типа секций - **config** определяет параметры общего назначения, а секция **conn** описывает параметры соединений IPsec.

12.19.1 Секция CONN

Секция **conn** содержит спецификацию соединения, определяющую параметры организации сетевого соединения IPsec. Имена соединений могут быть произвольными, но их синтаксис должен соответствовать приведенным выше (стр. 422) правилам. Ниже показан пример секции **conn**.

```
conn snt
    left=10.11.11.1
    leftsubnet=10.0.1.0/24
    leftnexthop=172.16.55.66
    right=192.168.22.1
    rightsubnet=10.0.2.0/24
    rightnexthop=172.16.88.99
    keyingtries=%forever
```

Прежде, чем перейти к описанию параметров, внесем некоторые уточнения в используемую терминологию. В системах с автоматическим созданием ключей существуют два типа обмена информацией - передача пользовательских пакетов IP и обмен данными между шлюзами для согласования ключей, их смены и общего контроля. “Путь данных” (множество **IPsec SA**) служит для передачи пользовательских пакетов, а соединения, используемые для согласования между шлюзами (множество **ISAKMP SA**), будем называть “каналами обмена ключами” (keying channel).

Чтобы избавить администраторов от ненужной работы по редактированию конфигурационных файлов всех участвующих в соединениях систем, взамен терминов “локальный” и “удаленный” используются обозначения **left** (левый) и **right** (правый). Выбор правой и левой стороны произволен, важно только соблюдать однотипное именование во всех системах. Это позволяет просто переносить конфигурационные файлы с одного компьютера на другой. В тех случаях, когда симметрия отсутствует, разумно использовать термин **left** для обозначения локальной системы, а термин **right** - для удаленной (по совпадению первых букв).

Многие из описанных ниже параметров относятся к одной из сторон соединения. Названия таких параметров и их описания приведены только для левой стороны. Вам следует помнить, что для правой стороны существует идентичный набор параметров с учетом смена направления (левый-правый).

Большинство параметров относятся к числу необязательных[1]. Параметры, используемые для ручной генерации ключей, не включаются в секцию для соединения с автоматической генерацией ключей и наоборот.

12.19.1.1 Общие параметры CONN

Перечисленные в таблице 147 параметры применимы как для автоматической, так и для ручной генерации ключей. В общем случае для работы соединения требуется точное соответствие значений общих параметров на обоих концах соединения.

Таблица 147 Конфигурационные параметры соединений ipsec

Параметр	Описание
type	Определяет тип соединения. В текущей версии поддерживаются значения **tunnel** (используется по умолчанию для соединений между хостами, хостом и подсетью или подсетями); **transport** (транспортное соединение между хостами); **passthrough** (соединение без использования IPsec); **drop** (пакеты должны отбрасываться) и **reject** (пакеты должны отвергаться с возвратом сообщения ICMP).
left	Обязательный параметр, задающий IP-адрес публичного интерфейса левого участника соединения в любом формате, приемлемом для ipsec_ttoaddr или содержащий одно из нескольких магических значений (magic value). Если этот параметр содержит значение **%defaultroute** и спецификация интерфейса в секции **config** (параграф 12.19.2 на стр. 426) включает **%defaultroute**, значение параметра **left** будет автоматически заменяться локальным адресом интерфейса на используемом по умолчанию маршруте (этот адрес определяется при старте IPsec); это также отменяет любые значения, заданные параметром **leftnexthop**. Значение **%defaultroute** может использоваться справа или слева но не с обеих сторон. Значение **%any** указывает, что поле адреса задается (при автоматической генерации ключей) в процессе согласования параметров. Значение **%opportunistic** указывает, что параметры **left** и **leftnexthop** определяются (при автоматической генерации ключей) из данных DNS для левого клиента. Значения **%group** и **%opportunisticgroup** расширяют это правило для групповых секций conn - один такой параметр помещается в секцию обычного или opportunistic-соединения для каждого блока CIDR, указанного в файле политики для групп (см. параграф 12.19.3 на стр. 428) с таким же именем, как секция **conn**.

1 *Имена обязательных параметров выделены подчеркиванием и явным упоминанием в тексте их описания.*

Параметр	Описание
leftsubnet	Частная подсеть, к которой относится левый участник соединения. Для указания подсети может использоваться формат **сеть/маска** или иной формат, поддерживаемый **ipsec_ttosubnet**. Если подсеть не указана, используется значение **left/32**, показывающее, что левый участник соединения является единственным на своей стороне.
leftnexthop	IP-адрес следующего шлюза публичной сети для левого участника соединения. По умолчанию используется значение **%direct** (прямое соединение с правой стороной). Если адрес шлюза меняется с использованием метода **left=%defaultroute** (см. выше), параметр не должен содержать явного адреса. Если этот метод не используется, но **leftnexthop=%defaultroute** и в секции **config** указано **interfaces=%defaultroute**, в качестве адреса следующего шлюза будет использоваться адрес принятого по умолчанию интерфейса.
leftupdown	Указывает updown-сценарий, используемый для настройки маршрутизатора и/или брандмауэра при изменении состояния соединения (по умолчанию используется сценарий ipsec _updown). Параметр может включать несколько значений, разделенных пробелами (не забывайте в таких случаях заключить все строку в двойные кавычки) и содержащих метасимволы командного процессора. Для получения более подробной информации вы можете воспользоваться командой **man 8 ipsec_pluto**. Этот параметр имеет отношение только к локальной стороне, удаленная может использовать свой сценарий.
leftfirewall	Указывает, должны ли для левого участника соединения выполняются функции маршрутизации/межсетевого экранирования (включая маскирование адресов) трафика, приходящего из сети **leftsubnet** (эти функции должны отключаться после организации соединения). Параметр может принимать значения **yes** и **no** (используется по умолчанию). Параметр не может использоваться в описании соединений, содержащих параметр **leftupdown**. Этот параметр имеет отношение только к локальной стороне. Если один или оба шлюза выполняют пересылку с межсетевым экранированием (возможно, включающим маскирование адресов) трафик через туннель IPsec исключается из экранирования и пакеты проходят через туннель в неизменном виде. Это означает, что подсети, соединенные с помощью такого туннеля, не должны перекрываться по адресам. Дополнительную информацию по этому вопросу вы сможете получить с помощью команды **man 8 ipsec_pluto**.

12.19.1.2 Параметры CONN - автоматическая генерация ключей

Перечисленные в таблице 148 параметры относятся только к режиму автоматической генерации ключей[1]. Если явно не указано иное, то для работы соединения требуется, чтобы на обоих концах использовались совпадающие значения перечисленных здесь параметров.

Таблица 148 Параметры автоматической генерации ключей

Параметр	Описание
keyexchange	Задает метод обмена ключами (в текущей версии поддерживается только метод ike, используемый по умолчанию).
auto	Задает необязательную операцию, которая автоматически будет выполняться при старте IPsec. В текущей версии поддерживаются операции **add** (эквивалент команды **ipsec auto --add**), **route** (**ipsec auto --route**), **start** (**ipsec auto --up), manual** (**ipsec manual -up**) и **ignore** (эквивалент используемого по умолчанию запуска без автоматического выполнения дополнительных операций). Дополнительные сведения по стартовым операциям вы сможете найти в параграфе 12.19.2 (стр. 426). Этот параметр имеет локальное значение и другая сторона не обязана его использовать или согласовывать. Однако в общем случае для соединений, которые предназначены быть постоянными, на обеих сторонах следует использовать **auto=start** для того, чтобы после любой перезагрузки соединение было восстановлено незамедлительно.
auth	Указывает должна ли аутентификация выполняться как часть процесса шифрования ESP (**auth=esp**) или отдельно с использованием протокола AH (**auth=ah**). По умолчанию используется значение **esp**.
authby	Задает способ выполнения аутентификации между парами шлюзов. Параметр может принимать значение **secret** (аутентификация с использованием разделяемого ключа), **rsasig** (цифровые сигнатуры RSA), **secret|rsasig** (для использования любого из этих методов) или **never** (без аутентификации). По умолчанию используется метод **rsasig**. Метод **never** следует использовать только для шунтирующих соединений.

1 *В ручном режиме эти параметры просто игнорируются.*

Параметр	Описание
keylife	Задает срок существования отдельного экземпляра соединения (набор ключей шифрования/аутентификации для пользовательских пакетов) с момента завершения согласования до окончания срока действия ключей. Параметр может принимать любое целочисленное значение с суффиксом **s** (секунды) или положительное десятичное значение с суффиксом **m**, **h** или **d** (минут, часов и суток, соответственно). По умолчанию время жизни составляет 8 часов (**8.0h**), максимальное значение - 24 часа. Обычно до завершения срока действия соединения выполняется процедура повторного согласования через канал обмена ключами. Стороны не обязаны использовать одинаковое время жизни соединений, но разные значения приведут к возникновению дополнительных процедур повторного согласования.
rekey	Определяет необходимость повторного согласования перед завершением срока жизни соединения. Параметр может принимать значения **yes** (используется по умолчанию) или **no**. Стороны не обязаны использовать одинаковые значения. Значение **no** приведет к тому, что данная сторона не будет запрашивать повторное согласование, но это значение не отменяет необходимость отвечать на аналогичные запросы другой стороны.
rekeymargin	Задает упреждение процедуры повторного согласования по отношению к моменту окончания срока жизни соединения или канала обмена ключами. Параметр может принимать такой же набор значений, который допускается для параметра **keylife**. По умолчанию процедура повторного согласования начинается за 9 минут до завершения срока жизни. Параметр имеет локальное значение и не должен согласовываться с другой стороной.
rekeyfuzz	Задает максимальное увеличение значения **rekeymargin** (в процентах) для задания случайного времени упреждения. Этот параметр имеет важное значение для шлюзов с многочисленными соединениями. Параметр может принимать целочисленные значения, с префиксом `%. Допускается задавать значения, превышающие 100%. Используемое по умолчанию значение задается с помощью **ipsec_pluto** (и в текущей версии составляет 100%). Отметим, что значение **rekeymargin** после его увеличения на случайное значение не должен превышать время жизни **keylife**. Нулевое значение параметра отключает случайное изменение времени упреждения. Параметр имеет локальное значение и не должен согласовываться с другой стороной.
keyingtries	Задает количество попыток (целое число или **%forever**) согласования соединения. Используемое по умолчанию значение **%forever** означает неограниченное число попыток (это значение устарело и в настоящее время неограниченное число попыток можно задать значением 0). Параметр имеет локальное значение и не должен согласовываться с другой стороной.
ikelifetime	Задает время жизни канала обмена ключами. Форматы значений соответствуют требованиям к значению параметра **keylife** (стр. 425). Принятое по умолчанию значение задается с помощью **ipsec_pluto** и составляет для текущей версии 1h (максимальное время жизни - 8h). Стороны не обязаны использовать одинаковое время жизни соединений, но разные значения приведут к возникновению дополнительных процедур повторного согласования.
compress	Задает использование для соединения компрессии содержимого IPComp. Сжатие на канальном уровне неприменимо к шифрованным данным, поэтому компрессия должна выполняться до шифрования. По умолчанию компрессия отключена (**no**). Стороны не обязаны согласовывать использование компрессии. При выборе значения **yes** IPsec будет поддерживать передачу с компрессией и без таковой, предпочитая использовать сжатие. При выборе значения **no** IPsec не будет предлагать использование компрессии другой стороне, но сможет принимать сжатую информацию.
disablearrivalcheck	Управляет отключением режима проверки пакетов (корректность адреса в заголовке) на выходе из туннеля. Такая проверка повышает уровень безопасности и не конфликтует с другими параметрами. По умолчанию функция проверки на выходе не запрещена (**no**). Параметр имеет локальное значение и его не нужно согласовывать с другой стороной.
failureshunt	Указывает, что следует делать с пакетами в случаях отказа при согласовании параметров (negotiation fail). По умолчанию не используется никаких операций (**none**), значения **passthrough** (пропускать), **drop** (отбрасывать) и **reject** (отвергать) имеют обычный смысл.

12.19.1.3 Параметры CONN - генерация ключей вручную

Описанные в этом параграфе параметры оказывают влияние только на генерацию ключей вручную и игнорируются при автоматической генерации, параметры которой описаны в предыдущем параграфе. Если явно не указано иное, стороны соединения должны использовать согласованные (совпадающие) значения параметров. Соединения с генерацией ключей вручную должны задавать по крайней мере одно значение **AH**[1] или **ESP**[2].

Отметим, что один из параметров **spi** или **spibase** является обязательным для соединения.

1 *Authentication Header - заголовок аутентификации.*
2 *Encapsulation Security Payload - протокол описанный RFC 2406 (http://rfc-editor.org/rfc/rfc2406.txt). Копию этого документа вы найдете на приложенном к книге компакт-диске (каталог Documents/).*

Таблица 149 Параметры ручной генерации ключей

Параметр	Описание
spi	Номер SPI, который будет использоваться для соединения (см. **man 8 ipsec**). Параметр должен использовать формат **0xhex** (**hex** - одна или несколько шестнадцатеричных цифр). Отметим, что в большинстве случаев требуется по крайней мере значение **spi = 0x100** (для KLIPS). Рекомендуется использовать значения SPI из диапазона **0x100 - 0xfff**.
spibase	Базовый номер SPI, который будет использоваться для соединения (см. **man 8 ipsec**). Параметр должен использовать формат **0xhex** (**hex** - одна или несколько шестнадцатеричных цифр). Отметим, что в большинстве случаев требуется по крайней мере значение **spibase = 0x100** (для KLIPS). Рекомендуется использовать базовые значения SPI из диапазона **0x100 - 0xff0**.
esp	Алгоритм шифрования/аутентификации ESP, который будет использоваться для соединения (например, **3des-md5-96**). Алгоритм должен быть приемлем, как значение опции **ipsec_spi -esp**. По умолчанию ESP не используется.
espenckey	Ключ шифрования ESP (должен быть приемлем как значение опции **ipsec_spi -enckey**). Кличи можно задать для каждого направления независимо с помощью параметров **leftespenckey** и **rightespenckey**.
espauthkey	Ключ аутентификации ESP (должен быть приемлем как значение опции **ipsec_spi --authkey**) Кличи можно задать для каждого направления независимо с помощью параметров **leftespauthkey** и **rightespauthkey**.
espreplay_window	Установка окна ESP replay (целое число от 0 до 64). Этот параметр имеет смысл только при использовании аутентификации ESP.
leftespspi	Значение SPI, используемое для левой стороны ESP SA. Это значение отменяет для данной стороны значение параметра **spi** или **spibase**. Обычно выражается шестнадцатеричным значением с префиксом **0x**.
ah	Алгоритм аутентификации AH, который будет использоваться для соединения (например, **hmac-md5-96**) Алгоритм должен быть приемлем, как значение опции **ipsec_spi -ah**. По умолчанию AH не используется.
ahkey	Ключ аутентификации AH (должен быть приемлем как значение опции **ipsec_spi -authkey**). Этот параметр является обязательным при наличии параметра **ah** и может быть задан независимо для каждой стороны с помощью параметров **leftahkey** и **rightahkey**.
ahreplay_window	Установка окна AH replay (целое число от 0 до 64).
leftahspi	Значение SPI, используемое для левой стороны AH SA. Это значение отменяет для данной стороны значение параметра **spi** или **spibase**. Обычно выражается шестнадцатеричным значением с префиксом **0x**.

12.19.2 Секция CONFIG

В текущей версии IPsec используется только одна конфигурационная секция по имени **setup**, содержащая сведения, которые программа использует при старте. Ниже приведен пример такой секции:

```
config setup
    interfaces="ipsec0=eth1 ipsec1=ppp0"
    klipsdebug=none
    plutodebug=all
    manualstart=
```

Параметры в этой секции являются необязательными, если явно не сказано иное. Ниже перечислены параметры, поддерживаемые текущей версией программы.

myid

Задает идентификатор, используемый в качестве **%myid**[1]. Если этот параметр не задан, в качестве **%myid** используется IP-адрес в **%defaultroute** (если этот адрес указан в записи TXT обратной зоны для домена) или имя хоста (если оно указано в записи TXT прямой зоны для домена). При отсутствии того и другого параметр остается неопределенным. Явные значения идентификаторов обычно начинаются с символа **@**.

interfaces

задает виртуальные и физические интерфейсы для использования IPsec в виде заключенного в кавычки списка пар **виртуальный адрес=физический адрес**, разделенных пробелами. Этот параметр может также содержать значение **%none** (нет интерфейсов), а одна из пар списка может быть указана как **%defaultroute**.

forwardcontrol

включает режим пересылки пакетов IP при старте IPsec и выключает его снова при остановке IPsec; Параметр может принимать значения **yes** и **no**, по умолчанию режим пересылки отключен. Для того, чтобы эта опция реально работала, следует отключить маршрутизацию на уровне ядра (например, установив значение net.ipv4.ip_forward = 0, см. стр 367), поскольку программа IPsec не может сделать этого.

rp_filter

1 *%myid применяется в политике групп для неявных соединений (см. стр. 427) и в качестве идентификатора для явных соединений.*

управляет использованием механизма фильтрации по обратному пути[2] для физических устройств, которые будут использоваться. Параметр может принимать значения **%unchanged**[3], 0 (по умолчанию), 1 или 2.

syslog
тип и уровень **syslog** (см. параграф 2.8.4 на стр. 49) для записи в журнальный файл сообщений при старте и отключении программы. По умолчанию **daemon.error**.

klipsdebug
задает уровень записи отладочной информации KLIPS в журнальные файлы. Пустое значение или **none** (используется по умолчанию) отключают запись отладочной информации. Значение **all** задает полную запись отладочной информации. Параметр может также содержать заключенный в кавычки список имен отладочных уровней, разделенных пробелами[3].

plutodebug
задает уровень записи отладочной информации Pluto в журнальные файлы. Пустое значение или **none** (используется по умолчанию) отключают запись отладочной информации. Значение **all** задает полную запись отладочной информации. Параметр может также содержать заключенный в кавычки список имен отладочных уровней, разделенных пробелами[4].

plutoopts
Дополнительная опция, передаваемая при старте модулю **pluto** (см. **man ipsec_pluto**).

plutostderrlog
Отключает использование **syslog** с выводом сообщений на устройство **stderr** или заданный аргументом файл.

dumpdir
Указывает каталог для записи дампа памяти при возникновении критических ошибок. Пустое значение блокирует запись дампа.

manualstart
Указывает организуемые вручную соединения. Параметр может содержать пустое значение (none), имя соединения или заключенный в кавычки список имен, разделенных пробелами. По умолчанию используется значение **none**. Дополнительную информацию об организации соединений вручную можно получить с помощью команды **man ipsec_manual**.

pluto
включает или отключает запуск модуля **Pluto**; по умолчанию используется значение **yes** (включено).

plutowait
включает или отключает режим ожидания завершения попытки согласования параметров перед началом следующей попытки для модуля Pluto. По умолчанию ожидание отключено (**no**).

prepluto
команда, выполняемая перед стартом **Pluto** (например, для шифрования или расшифровки файла **ipsec.secrets**). Любой вывод команды перенаправляется в систему ведения журнала, поэтому использование интерактивного режима в таких случаях затруднительно, если не применяется устройство **/dev/tty** или его эквивалент. По умолчанию никакой команды не выполняется (**none**).

postpluto
команда, выполняемая после старта **Pluto** (например, для удаления расшифрованной копии файла **ipsec.secrets**). Любой вывод команды перенаправляется в систему ведения журнала, поэтому использование интерактивного режима в таких случаях затруднительно, если не применяется устройство **/dev/tty** или его эквивалент. По умолчанию никакой команды не выполняется (**none**).

fragicmp
управляет использованием туннеля для передачи сообщений ICMP о необходимости фрагментации. По умолчанию такие сообщений передаются через туннель (**yes**).

hidetos
управляет установкой значения поля TOS для передаваемых через туннель пакетов. Используемое по умолчанию значение **yes** сбрасывает значение поля в 0, а **no** сохраняет исходное значение поля TOS.

uniqueids
определяет необходимость обеспечения уникальности идентификаторов участников соединения.

overridemtu
задает значение MTU для интерфейсов **ipsecn**, которое должно быть установлено взамен используемого IPsec по умолчанию большого значения MTU. Этот параметр используется весьма редко.

12.19.2.1 Неявные соединения

Система автоматически определяет несколько соединений для реализации принятой по умолчанию групповой политики. Каждое такое соединение можно переопределить, задав явно соединение с таким же именем. Если новое соединение имеет атрибут **auto=ignore**, автоматическое определение подавляется.

Ниже перечислены используемые автоматически определения.

conn clear

2 *Reverse path filtering.*
3 *Сохраняется значение, заданное в переменной SysCtl (см. параграф 12.3.1.13.12 на стр. 374).*
3 *Дополнительную информацию об уровнях отладки можно получить с помощью команды **man ipsec_klipsdebug***
4 *Дополнительную информацию об уровнях отладки можно получить с помощью команды **man ipsec_ pluto***

```
        type=passthrough
        authby=never
        left=%defaultroute
        right=%group
        auto=route

    conn clear-or-private
        type=passthrough
        left=%defaultroute
        leftid=%myid
        right=%opportunisticgroup
        failureshunt=passthrough
        keyingtries=3
        ikelifetime=1h
        keylife=1h
        rekey=no
        auto=route

    conn private-or-clear
        type=tunnel
        left=%defaultroute
        leftid=%myid
        right=%opportunisticgroup
        failureshunt=passthrough
        keyingtries=3
        ikelifetime=1h
        keylife=1h
        rekey=no
        auto=route

    conn private
        type=tunnel
        left=%defaultroute
        leftid=%myid
        right=%opportunisticgroup
        failureshunt=drop
        keyingtries=3
        ikelifetime=1h
        keylife=1h
        rekey=no
        auto=route
    conn block
        type=reject
        authby=never
        left=%defaultroute
        right=%group
        auto=route

    # политика, используемая по умолчанию
    conn packetdefault
        type=tunnel
        left=%defaultroute
        leftid=%myid
        left=0.0.0.0/0
        right=%opportunistic
        failureshunt=passthrough
        keyingtries=3
        ikelifetime=1h
        keylife=1h
        rekey=no
        auto=route
```

На эти соединения не оказывают влияния параметры секции **%default**. Для работы этих соединений требуется корректное значение **%defaultroute**. Параметр **leftid** будет содержать реальный адрес IP (это требуется для корректной установки реверсных записей DNS).

Неявные соединения определяются после всех прочих соединений.

12.19.3 Файлы политики для групп

Необязательные файлы каталога **/etc/freeswan/ipsec.d/policies/**, включая

```
/etc/freeswan/ipsec.d/policies/clear
/etc/freeswan/ipsec.d/policies/clear-or-private
/etc/freeswan/ipsec.d/policies/private-or-clear
/etc/freeswan/ipsec.d/policies/private
/etc/freeswan/ipsec.d/policies/block
```

могут содержать конфигурационные параметры политики для групп в дополнение к параметрам, заданным в файле **ipsec.conf**. Содержимое этих файлов не требует безопасного хранения и не может снизить уровень безопасности соединений.

Все эти файлы являются текстовыми и содержат списки CIDR-блоков по одному в каждой строке. После префикса адреса может следовать пробел и символ #, справа от которых помещается текст комментариев.

Соединения, указанные в файле /etc/freeswan/ipsec.conf с right=%group или right=%opportunisticgroup относятся к соединениям с групповой политикой. При загрузке файла политики группы с помощью команды

```
ipsec auto --rereadgroups
```

или во время загрузки системы, каждый блок CIDR обслуживается как экземпляр значения **right**. Система трактует такие экземпляры как нормальные соединения.

Например, если для системы определена политика private и файл /etc/freeswan/ipsec.d/policy/private включает запись 192.0.2.3, система создаст экземпляр соединения private#192.0.2.3. Это соединение будет наследовать все параметры private, но адрес правого клиента будет 192.0.2.3.

12.19.3.1 Используемая по умолчанию политика группы

Стандартная инсталляция FreeS/WAN включает несколько вариантов политики групп, обеспечивающих возможность классификации возможных партнеров в различные классы безопасности IPsec: приватный (private) с шифрованием всего трафика, private-or-clear (предпочтительно использовать шифрование), clear-or-private (шифровать по запросу партнера), открытый или блокировка. Неявные группы применяются только для локальных хостов, и реализуются с помощью описанных выше неявных соединений.

12.19.4 Выбор соединения

При выборе соединения для исходящего пакета с a%trap, система предпочтет наиболее специфичный маршрут eroute, который включает IP-адреса отправителя и получателя пакета. Сначала проверяются подсети отправителей, затем подсети получателей. При инициализации рассматриваются только маршрутизируемые соединения (routed connections). Для откликов принимаются во внимание также немаршрутизируемые, но добавленные соединения.

12.20 Поля протоколов, используемые в фильтрах отображения Ethereal

12.21 Источники информации

12.21.1 Книги, журналы

1) Garfinkel, Spafford. Practical UNIX and Internet Security
2) Anonymous. Maximum Linux Security. A Hacker's Guide to Protecting Your Linux Server and Workstation. (имеется перевод на русский язык - Максимальная безопасность в Linux. Руководство по защите серверов и рабочих станций, написанное хакером).

12.21.2 Internet

12.21.2.1 Центры и команды по информационной безопасности

http://www.cert.org - Координационный центр информационной безопасности при университете Карнеги-Меллона в США.

http://www.us-cert.gov - United States Computer Emergency Readiness Team (команда по обеспечению готовности к отражению компьютерных аварий США).

http://www.fedcirc.gov - Federal Computer Incident Response Center (федеральный центр по компьютерным инцидентам США).

http://www.first.org - Forum of Incident Response and Security Teams (форум команд по обеспечению безопасности и откликам на инциденты).

http://www.cerias.purdue.edu - Center for Education and Research in Information Assurance and Security (Учебно-исследовательский центр информационной безопасности) университета Purdue (США).

http://csrc.nist.gov - Computer Security Resource Center (центр компьютерной безопасности) при Институте стандартов и технологий США (NIST).

http://www.sans.org - SANS Institute.

http://www.usenix.org - USENIX Advanced Computing Systems Association.

http://www.auscert.org.au/ - Computer Emergency Response Team for Australia (Австралийская команда по компьютерным авариям).

12.21.2.2 Стандарты, протоколы

http://rfc-editor.org - репозиторий документов RFC.

http://www.ietf.org - IETF

http://www.protokols.ru - энциклопедия сетевых протоколов (документы на русском языке).

12.21.2.3 Порталы, сетевые издания, обзоры, ссылки

http://secinf.net - Network Security Library

http://infosecdaily.net - InfosecDaily - новости, обзоры, рекомендации.

http://www.linuxjournal.com - Linux Journal - сетевой вариант одноименного журнала.

http://www.securityfocus.com/infocus/1410 - Anton Chuvakin, A Comparison of iptables Automation Tools - обзор нескольких средств автоматизации работы с правилами iptables.

12.21.2.4 Программы

lxr.linux.no - гипертектовый депозитарий исходных кодов Linux.

http://lwn.net/security/ - LWN Security Resources (обзоры, программы, дистрибутивы).

http://lwn.net/Distributions/ - аннотированный список дистрибутивов Linux общего и специального назначения.

http://www.lids.org - проект LIDS (набор патчей и средств повышения уровня безопасности систем Linux).

http://www.kerneltrap.org -

http://www.kernelnewbies.org -

http://sourceforge.net/projects/tcpslice/ - tcpslice is a tool for extracting portions of packet trace files generated using tcpdump's -w flag. It can combine multiple trace files, and/or extract portions of one or more traces based on time.

12.21.2.5 Документация

The Linux Documentation Project - документация по системам Linux.

http://www.linuxsecurity.com/docs/colsfaq.html - comp.os.linux.security FAQ - ответы на вопросы по безопасности в Linux

http://www.faqs.org/docs/iptables/ - учебник по IPtables

http://www.linuxhq.com/guides/TLK/tlk.html - книга **The Linux Kernel**

12.21.2.6 Сайты организаций, связанных с информационной безопасностью

http://www.vpnc.org - VPNC (Virtual Private Network Consortium) - консорциум производителей VPN-продукции.