Network Working Group
Request for Comments: 3549
Category: Informational

J. Salim
Znyx Networks
H. Khosravi
Intel
A. Kleen
Suse
A. Kuznetsov
INR/Swsoft

July 2003

Netlink как протокол для служб IP

Linux Netlink as an IP Services Protocol

Статус документа

Этот документ содержит информацию, предназначенную для сообщества Internet, и не задаёт каких-либо стандартов Internet. Документ может распространяться без ограничений.

Авторские права

Аннотация

Данный документ описывает интерфейс Netlink ОС Linux, который используется операционной системой для обмена сообщениями как между процессами ядра, так и между ядром и пользовательскими процессами. Основное внимание в документе уделяется описанию функциональности Netlink как протокола, связывающего компоненты FEC[1] и CPC[2], которые определяют работу сервиса IP. Прочие варианты использования Netlink, включая обмен сообщениями внутри ядра и между процессами (IPC[3]), а также настройка конфигурации служб, не относящихся к IP (несетевые службы или сетевые службы других протоколов), в данном документе не рассматриваются.

Документ предназначен для создания информационного контекста на начальном этапе работы группы IETF ForCES[4].

Оглавление

[1]Forwarding Engine Component - компонент машины пересылки.
[2]Control Plane Component - компонент плоскости управления.
[3]Inter-process communication - обмен информацией между процессами.
[4]Forwarding & Control Element Separation - разделение элементов пересылки и управления. Страница рабочей группы доступна по адресу http://www.ietf.org/html.charters/forces-charter.html. Работа группы завершена в марте 2015 г. *Прим. перев.*

1. Введение

Концепция разделения служб IP на управление и пересылку впервые была реализована в начале 1990-х годов в сокетах маршрутизации BSD 4.4 [9]. В то время наибольшую важность представляло простое решение вопроса пересылки пакетов IP (v4) и управления таблицами пересылки IPv4 в CPC (с помощью консольного интерфейса или демона динамической маршрутизации).

Мир IP-сетей с тех давних пор существенно изменился. Linux Netlink, с точки зрения обеспечения сервиса и управления, кроме поддержки сокетов маршрутизации, обеспечивает ряд дополнительных функций. Начиная с ядра Linux 2.1, сокет Netlink обеспечивает абстракцию служб IP для нескольких типов сервиса в дополнение к классической пересылке IPv4 в соответствии с RFC 1812.

Мотивом для создания этого документа послужило отнюдь не желание описать весь набор служб, для которых можно использовать Netlink. Фактически многие типы сервиса (групповая маршрутизация, туннелирование, маршрутизация на основе правил и т. п.) просто не рассматриваются в данном документе. Не предназначен документ и для использования в качестве учебника по Netlink. Идея документа заключается в общем описании Netlink и более подробном рассмотрении обязательных компонентов в контексте работы группы ForCES - IPv4 и QoS. Документ также служит предварительным описанием механизмов, изучение которых представляет интерес в рамках ForCES. Рассматривается подмножество функций, доступных в ядре версии Linux 2.4.6, которая была последней во время подготовки данного документа. Документ рассматривает лишь функции, связанные с IPv4.

Документ начинается с концептуальных определений, после чего Netlink рассматривается в свете этих определений.

1.1. Определения

CP[1] представляет собой среду исполнения, которая может иметь несколько субкомпонентов, обозначаемых как CPC. Все CPC, обеспечивающие контроль для разных служб IP, будут выполняться посредством машины пересылки (Forwarding Engine или FE). Такие отношения между компонентами означают возможность наличия нескольких CPC для одной физической CP, если они контролируют несколько служб IP. По сути, связь между CP и FE является абстракцией сервиса.

1.1.1. Компоненты CPC

Компоненты плоскости управления CPC включают сигнальные протоколы от протоколов динамической маршрутизации (например, OSPF [5]) до протоколов распространения тегов (например, CR-LDP [7]). Классические протоколы и операции управления также входят в эту категорию. Среди них такие механизмы, как SNMP [6], COPS [4] и фирменные средства настройки конфигурации CLI/GUI. Задача плоскости управления состоит в обеспечении среды исполнения для перечисленных действий с целью настройки конфигурации и управления вторым компонентом элемента сети (Network Element или NE) - машиной пересылки FE. Результат настройки конфигурации определяет способ обработки пакетов в FE.

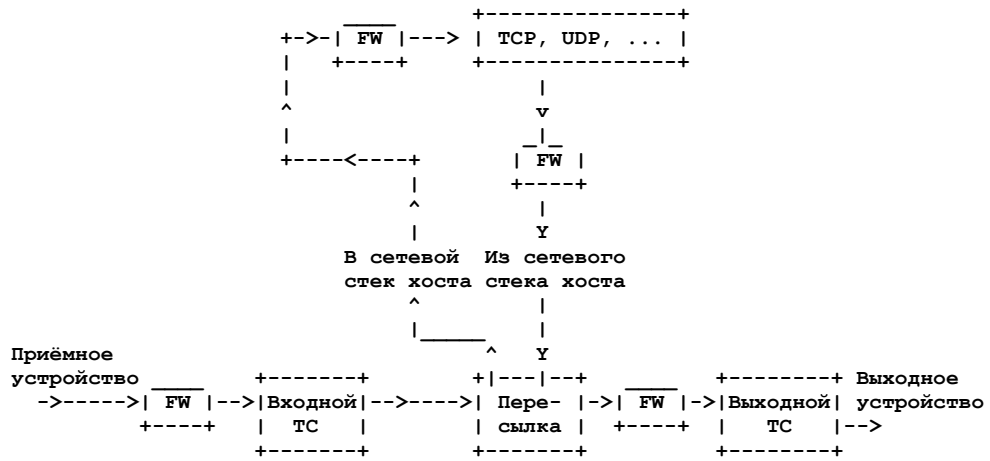
1.1.2. Компоненты FEC

Машина пересылки FE представляет собой объект NE, который первым получает сетевые пакеты (из сети в NE).

Связанная с сервисом компонента FE просматривает пакет с целью обеспечения для него обработки, заданной компонентами CPC для данного типа сервиса IP. Различные службы будут использовать разные компоненты FEC. Сервисные модули могут объединяться в цепочки для поддержки более сложных типов сервиса (в рамках описанной ниже модели Linux FE).

Будучи созданным для поддержки конкретной службы, сервисный компонент FE по-прежнему соответствует принципам модели пересылки.

1.1.2.1. Модель машины пересылки IP в Linux



На рисунке показана модель Linux FE для отдельного устройства. Единственной обязательной частью этой модели является модуль пересылки (Пересылка), соответствующий RFC 1812. Различные модули сетевого экранирования (FW), а также управления входящим и исходящим трафиком (Traffic Control или TC) не являются обязательными и могут даже использоваться для обхода модуля пересылки RFC 1812. Эти модули показаны в виде простых блоков на пути передачи данных и фактически могут представлять собой каскады из множества субмодулей. Дополнительную информацию о таких модулях можно найти в [10] и [11].

[1]Control Plane - плоскость управления

Пакеты, прибывающее на входное устройство, сначала проходят через модуль межсетевого экранирования (FW), который может отбрасывать (drop) и изменять (mangle) пакеты или выполнять с ними иные операции. После прохождения модуля FW входящие пакеты в зависимости от принятой политики могут попадать во входной модуль контроля трафика TC, который выполняет операции по измерению и регулированию потоков входящего трафика. Пакеты могут отбрасываться входным модулем TC в зависимости от результатов измерения уровня трафика и принятой политики. После этого модуля пакет передаётся единственному обязательному модулю, который обеспечивает пересылку в соответствии с требованиями RFC 1812. Пакет может быть отброшен, если он не соответствует требованиям RFC 1812, RFC 1122, а также дополняющих их документов. Этот модуль является точкой выбора пути, из которой пакет, направленный принявшему его элементу NE, может быть передан сетевому стеку хоста.

Пакеты, не адресованные данному NE, могут проходить через субмодуль маршрутизации на базе правил (часть модуля пересылки), если такая маршрутизация поддерживается. Затем пакет передаётся следующему модулю сетевого экранирования, который может отбросить или изменить пакет в зависимости от настроек субмодулей и выбранной политики. После прохождения этого модуля пакет попадает в выходной фильтр контроля трафика (TC).

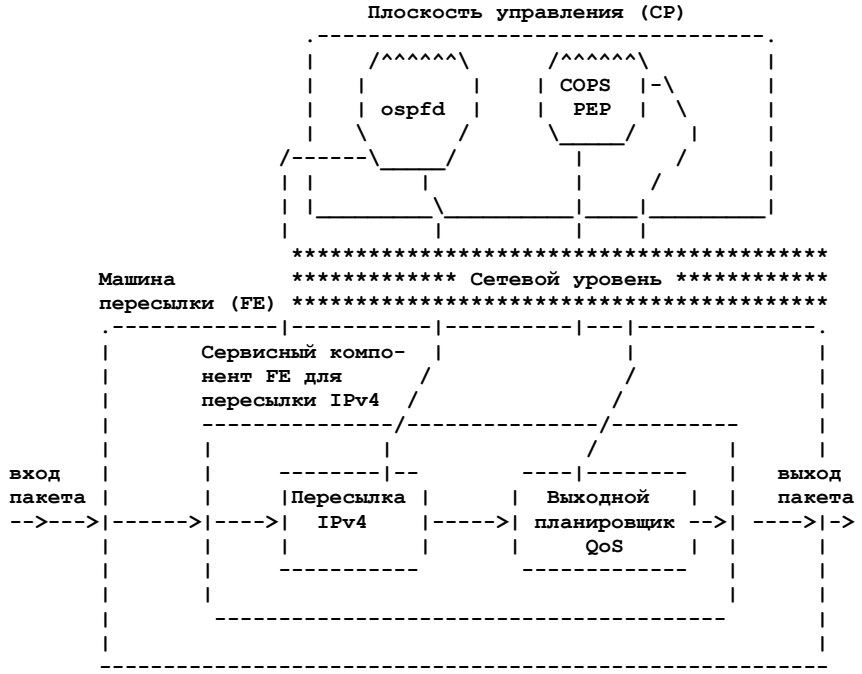
Выходной TC может отбрасывать пакеты с учётом политики, состояния очередей, уровня насыщения и правил управления скоростью исходящего потока. На этом этапе используются выходные очереди и задержки или отбрасывание пакета могут происходить как до его включения в очередь, так и после этого. Судьба пакета определяется выбранными для модуля алгоритмами и политикой.

1.1.3. Службы IP

Служба IP - это процессы обработки пакета IP внутри NE. Эти процессы определяются комбинацией CPC и FEC.

Занимаемое службой время начинается с момента прихода пакета в NE и заканчивается в момент, когда пакет покидает NE. Существенно, что поведение служб IP в этом контексте определяется конкретным хостом. Компоненты CP, запущенные на NE, задают сквозной для всего пути контроль служб с помощью управляющих приложений и сигнальных протоколов. Такие распределенные компоненты CPC унифицируют сквозное представление служб IP. Как было отмечено выше, такие компоненты CP определяют поведение FE (и NE) по отношению к описываемому пакету.

Простым примером службы IP может служить классическая пересылка IPv4. В этом случае управляющие компоненты (протоколы маршрутизации OSPF, RIP и т. п.) и фирменные средства настройки конфигурации CLI/GUI изменяют таблицы пересылки FE для того, чтобы обеспечить простой сервис по пересылке пакетов на следующий интервал (next hop). Обычно NE, обеспечивающие такой сервис, называют маршрутизаторами.



На рисунке показан простой пример реализации FE<->CP для обеспечения классической пересылки IPv4 с некоторыми дополнительными функциями QoS для управления выходными очередями. Демон ospfd управляет работой протокола OSPF, а COPS PEP[1] представляет собой дополнительный компонент CPC. Компонент IPv4 FE включает модули пересылки IPv4 и выходного планировщика QoS. В качестве дополнительной службы может быть добавлен сервис пересылки на основе правил между модулями пересылки IPv4 и планировщика QoS. Простейший классический вариант будет включать только модуль пересылки IPv4.

Опыт использования сетей говорит о важности добавления в маршрутизаторы новых типов сервиса, удовлетворяющих современным требованиям. Для решения этих задач были созданы и стандартизованы новые службы, которые могут выходить за пределы содержимого заголовков сетевого уровня. Однако, для обеспечивающих пересылку пакетов устройств NE по-прежнему используется термин «маршрутизатор». Новые службы (которые могут выходить за пределы заголовков L3) включают межсетевое экранирование, QoS с использованием Diffserv и RSVP, NAT, маршрутизацию на базе правил и т. п. Для таких служб создаются новые протоколы и средства управления.

Одним из определений сервиса IP является «все, за что сервис-провайдеры могут взять деньги».

2. Архитектура Netlink

Управление компонентами IP-сервиса определяется с помощью шаблонов. Компоненты FEC и CPC участвуют в предоставлении услуг IP путём обмена данными с использованием таких шаблонов. FEC может непрерывно получать

[1]Policy Enforcement Point - точка применения политики.

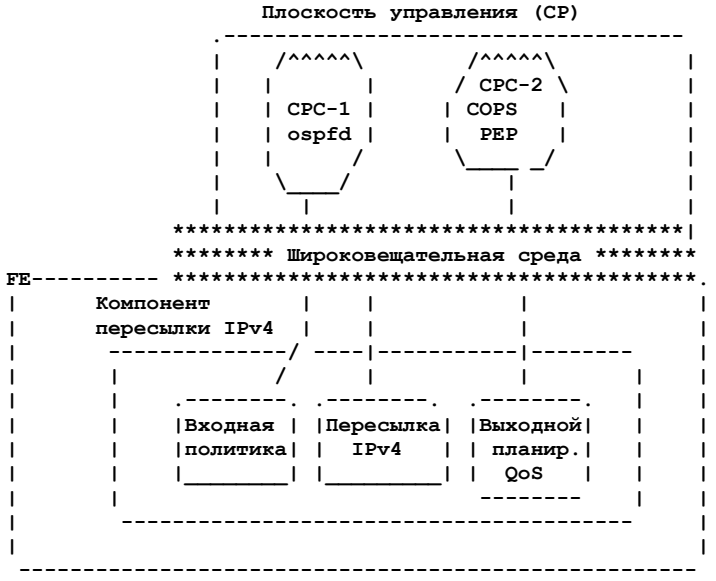
обновления от компонента CPC, указывающие, как предоставлять услуги (например, для пересылки пакетов IPv4, добавления, удаления или изменения маршрутов).

Взаимодействие между FEC и CPC в контексте Netlink определяется протоколом. Netlink предоставляет механизмы для CPC (в пользовательском пространстве) и FEC (в ядре), позволяющие им получить свои собственные определения для протокола. Это связано с тем, что пользовательское пространство и ядро находятся на разных уровнях защиты. Следовательно, для обмена информацией между компонентами требуется протокол. Такой протокол обычно обеспечивается неким привилегированным сервисом, который имеет возможность копирования данных между различными уровнями защиты. Будем называть такую службу сервисом Netlink. Этот сервис может также инкапсулироваться в протоколы транспортного уровня, если CPC и FEC выполняются на разных узлах. Компоненты FEC и CPC, используя механизмы Netlink, могут выбрать надёжный протокол обмена данными. По умолчанию Netlink не обеспечивает гарантированного обмена данными.

Отметим, что FEC и CPC могут располагаться на одном уровне защиты памяти и использовать системный вызов connect() для создания прямого пути и обмена информацией через этот путь. В данном документе этот механизм рассматриваться не будет, отметим лишь возможность его реализации. Предполагается, что FEC является частью ядра, а CPC размещается в пользовательском пространстве. Это не означает однако, что приведённая в документе информация относится лишь к случаю размещения этих компонентов в разных областях защиты и не привязывает компоненты к одному узлу.

Отметим, что Netlink позволяет обоим компонентам участвовать в предоставлении услуг IP.

2.1. Логическая модель Netlink



На рисунке показана простая диаграмма логических связей между компонентами FEC и CPC. В качестве примера использована FEC пересылки IPv4 (см. параграф 3.1. Служба NETLINK_ROUTE). Netlink логически моделирует FEC и CPC в форме узлов, связанных между собой через широковещательную среду. Свойства среды обусловлены сервисом. В приведённом примере показана широковещательная среда, принадлежащая расширенному сервису пересылки IPv4.

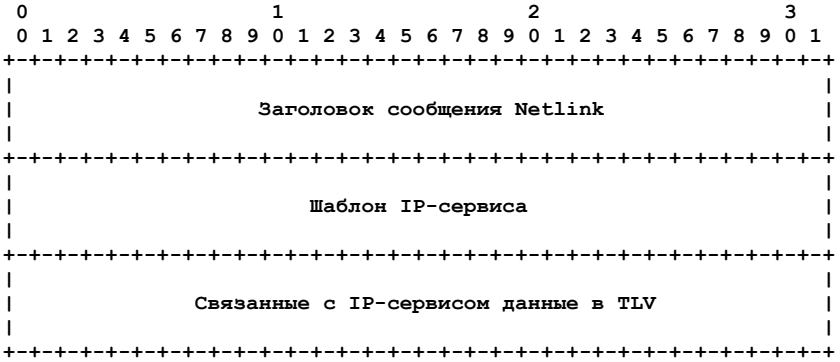
Узлы (CPC и FEC в рассматриваемом примере) подключены к среде передачи и регистрируются для получения сообщений определённых типов. CPC может подключаться к нескольким средам, если это способствует более эффективному управлению сервисом. Все узлы (CPC и FEC) принимают пакеты из широковещательной среды. Пакеты могут отбрасываться средой передачи, если они имеют непригодный формат или содержат ошибки. Отброшенные пакеты не поступают ни на один из узлов. Сервис Netlink может передавать отправителю сигналы об ошибках при обнаружении непригодных пакетов Netlink.

Передаваемые в среду пакеты могут быть широковещательными, групповыми или индивидуальными. Узлы FEC и CPC регистрируют свою заинтересованность в сообщениях определённого типа для их обработки или мониторинга.

В Приложениях 1 и 2 приведено более детальное рассмотрение этого взаимодействия.

2.2. Формат сообщений

В сообщениях Netlink существует три уровня - заголовок Netlink, шаблон IP-сервиса и связанные с сервисом данные.



Сообщения Netlink используются для обмена данными между FEC и CPC, параметризации FEC, асинхронной передачи сведений о событиях FEC компонентам CPC и сбора/просмотра статистики (обычно с помощью CPC).

Заголовок сообщения Netlink используется для всех типов сервиса, тогда как шаблоны (IP Service Template) связаны с конкретными типами. Каждая служба IP передаёт данные параметризации (от CPC к FEC) или отклики (от FEC к CPC). Эти данные передаются в формате TLV[1] и являются уникальными для сервиса. Отдельные компоненты сообщений Netlink подробно рассматриваются ниже.

2.3. Модель протокола

Здесь описано, как Netlink обеспечивает механизм ориентированного на службы взаимодействия между FEC и CPC.

2.3.1. Адресация служб

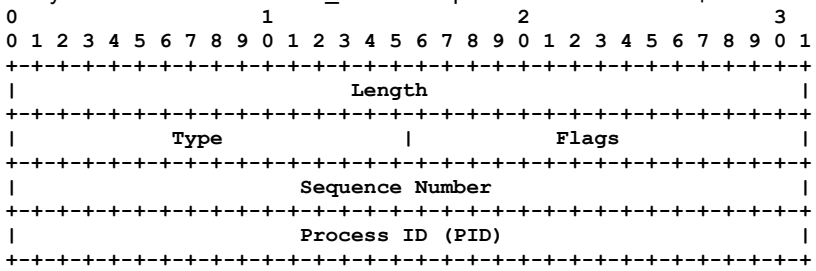
Для получения доступа сначала нужно соединиться с сервисом на FE. Соединение организуется путём системного вызова socket() для домена PF_NETLINK. Каждый компонент FEC указывается номером протокола. В результате вызова могут создаваться сокеты типа SOCK_RAW или SOCK_DGRAM, хотя Netlink не различает сокеты этих типов. Соединение с сокетом обеспечивает основу для адресации FE<->CP.

Затем организуется подключение к сервису (в любой момент в течение срока существования соединения) путём ввода обусловленной сервисом команды (от CPC к FEC, в основном для настройки конфигурации), команды сбора статистики или подписки (отказа) на уведомления о связанных с сервисом событиях. Закрытие сокета прерывает транзакцию.

Примеры рассматриваются в Приложениях 1 и 2.

2.3.2. Заголовок сообщений Netlink

Сообщения Netlink представляют собой поток байтов с одним или несколькими заголовками Netlink и связанными с ними данными (payload). Если данных слишком много для одного сообщения, они могут быть разделены на несколько сообщений Netlink, которые обычно называют многокомпонентным сообщением (multipart message). Для таких сообщений первый и последующие заголовки, за исключением последнего, содержат флаг NLM_F_MULTI. В заголовке последнего сообщения указывается тип NLMSG_DONE. Формат заголовка сообщения Netlink показан на рисунке.



Length - 32 бита

Размер сообщения в байтах с учётом заголовка.

Type - 16 битов

Тип содержимого в сообщении. Поле может включать один из стандартных идентификаторов, указанных ниже.

NLMSG_NOOP	Сообщение игнорируется.
NLMSG_ERROR	Сообщение сигнализирует об ошибке и поле данных содержит структуру nlmsgerr. Такие сообщения обычно передаются от FEC к CPC и могут рассматриваться как NACK[2].
NLMSG_DONE	Сообщение является последней частью многокомпонентного сообщения.

Отдельные службы IP могут использовать добавочные типы сообщений, например сервис NETLINK_ROUTE задаёт несколько таких типов, включая RTM_NEWLINK, RTM_DELLINK, RTM_GETLINK, RTM_NEWADDR, RTM_DELADDR, RTM_NEWROUTE, RTM_DELROUTE и др.

Flags - 16 битов

Стандартные флаги, используемые в заголовках Netlink, приведены в таблице.

NLM_F_REQUEST	Должен устанавливаться для всех откликов (обычно они передаются из пользовательского пространства в ядро).
NLM_F_MULTI	Сообщение является частью (не последней) многокомпонентного сообщения. Для последней части указывается тип NLMSG_DONE.
NLM_F_ACK	Запрос на подтверждение при успехе. Обычно устанавливается в сообщениях из пользовательского пространства (CPC) в ядро (FEC).
NLM_F_ECHO	Возвратить «эхо» для данного запроса. Обычно устанавливается в сообщениях из пользовательского пространства (CPC) в ядро (FEC).

В запросах GET для конфигурационной информации, передаваемых в FEC, используются дополнительные флаги.

NLM_F_ROOT	Возвращать полную таблицу вместо одной записи.
NLM_F_MATCH	Возвращать все записи, соответствующие критерию, переданному в поле данных сообщения.
NLM_F_ATOMIC	Возвращать неделимый снимок таблицы, которая указана. Установка этого флага может требовать специальных привилегий, поскольку флаг способен прерывать сервис FE на достаточно продолжительное время.
NLM_F_DUMP	NLM_F_ROOT OR NLM_F_MATCH (удобный макрос)

В запросах NEW могут использоваться дополнительные флаги.

NLM_F_REPLACE	Заменить существующий объект конфигурации в соответствии с данным запросом.
NLM_F_EXCL	Не заменять существующий объект новым.
NLM_F_CREATE	Создать объект конфигурации, если его не существует.
NLM_F_APPEND	Добавить объект в конце списка имеющихся.

Для тех, кто хорошо знаком с операциями на сокетах маршрутизации BSD, в таблице приведены эквиваленты таких операций.

[1]Type-Length-Value - тип-размер-значение.
[2]Подтверждение отрицательного результата - Negative ACK. *Прим. перев.*

BSD	*Netlink*
ADD	NLM_F_CREATE OR NLM_F_EXCL
CHANGE	NLM_F_REPLACE
Check	NLM_F_EXCL
APPEND	NLM_F_CREATE

Sequence Number - 32 бита

Порядковый номер сообщения.

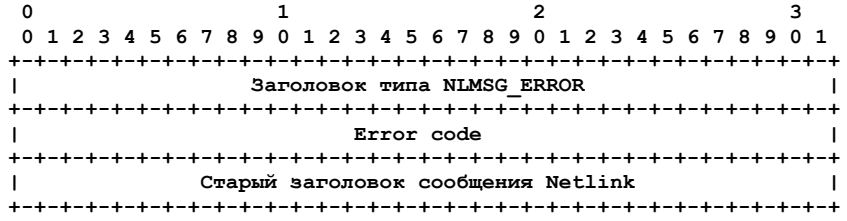
Process ID (PID) - 32 бита

Идентификатор процесса (PID), передающего сообщение. Значение PID используется ядром для мультиплексирования в нужный сокет. При передаче сообщений из ядра в пользовательское пространство устанавливается PID = 0.

2.3.2.1. Механизмы создания протоколов

Один из способов организации надёжного протокола обмена между FEC и CPC является использование комбинации порядковых номеров, ACK и таймеров повтора передачи. Порядковые номера и подтверждения ACK обеспечивает Netlink, таймеры - ОС Linux. Можно также создать протокол heartbeat[1] для обмена между FEC и CPC за счёт использования флагов ECHO и сообщений типа NLMSG_NOOP.

2.3.2.2. Сообщение Netlink ACK



Эти сообщения используются для передачи как подтверждений (ACK), так и сведений об отрицательном результате (NACK). Обычно такие сообщения передаются от FEC к CPC (в ответ на сообщение с запросом подтверждения). Однако CPC должны обеспечивать возможность передачи сообщений ACK в адрес FEC при наличии соответствующего запроса. Семантика этих сообщений зависит от сервиса IP.

Error code - integer (обычно 32 бита)

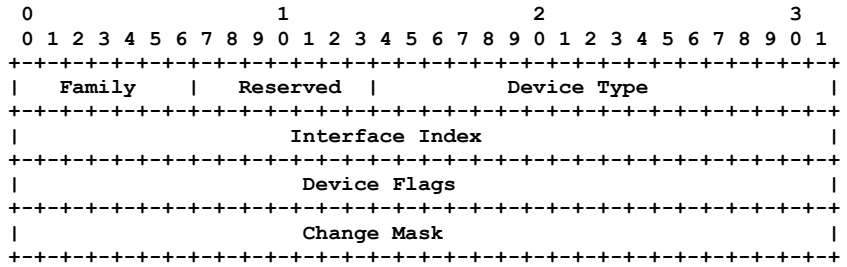
Нулевое значение кода ошибки говорит о том, что сообщение является подтверждением успеха (ACK). Такие сообщения содержат заголовок исходного сообщения Netlink, который может использоваться для сравнения (например, порядкового номера).

Отличный от нуля код говорит об отрицательном результате (NACK). В таких ситуациях данные Netlink, которые были переданы ядру, возвращаются вместе с исходным заголовком Netlink. Устанавливается также пригодное для вывода с помощью perror() значение кода ошибки (не в заголовке сообщения, а в переменной окружения).

2.3.3. Шаблоны FE системных служб

Существуют системные службы, которые предлагают свой сервис для использования другими службами. Обычно они включают возможности настройки конфигурации, сбора статистики, прослушивания сведений об изменении общих ресурсов, управления адресами IP, канальные события и т. п. Данный раздел включает описание подобных служб для их логического разделения (несмотря на то, что все они доступны через FEC NETLINK_ROUTE). Причина этого заключается в том, что они существуют в NETLINK_ROUTE в силу исторически сложившихся причин (ошибки), связанных с тем, что сокеты BSD 4.4 Route реализованы как часть сокетов пересылки IPv4.

2.3.3.1. Сервисный модуль сетевого интерфейса



Эта служба обеспечивает возможность создания и удаления сетевых интерфейсов, а также получения информации о существующем интерфейсе. Интерфейс может быть физическим или виртуальным и не связан с сетевым протоколом (например, с помощью такого сообщения можно указать интерфейс x.25). Шаблон сообщения показан на рисунке.

Family - 8 битов

AF_UNSPEC.

Device Type - 16 битов

Указывает тип канала (Ethernet, туннель и т. п.). В данном документе рассматривается только IPv4, хотя тип канала не зависит от протокола L3.

Interface Index - 32 бита

Уникальный идентификатор интерфейса.

Device Flags - 32 бита

Флаги интерфейса, перечисленные в таблице.

[1] «Пульс».

Флаг	*Значение*
IFF_UP	Интерфейс активизирован администратором
IFF_BROADCAST	Установлен действительный широковещательный адрес.
IFF_DEBUG	Флаг режима отладки для интерфейса.
IFF_LOOPBACK	Петлевой интерфейс (loopback).
IFF_POINTOPOINT	Интерфейс типа «точка-точка».
IFF_RUNNING	Интерфейс находится в работающем состоянии.
IFF_NOARP	Для интерфейса не требуется протокол ARP.
IFF_PROMISC	Интерфейс работает в режиме захвата (promiscuous).
IFF_NOTRAILERS	Избегать использования трейлеров.
IFF_ALLMULTI	Принимать все пакеты с групповыми адресами.
IFF_MASTER	Ведущий интерфейс для транка с распределением нагрузки.
IFF_SLAVE	Ведомый интерфейс для транка с распределением нагрузки.
IFF_MULTICAST	Поддержка групповой адресации.
IFF_PORTSEL	Интерфейс может выбирать тип среды с помощью ifmap.
IFF_AUTOMEDIA	Активизирован автоматический выбор типа среды.
IFF_DYNAMIC	Интерфейс создан в динамическом режиме.

Change Mask - 32 бита

Зарезервированное поле, которое должно иметь значение 0xFFFFFFFF.

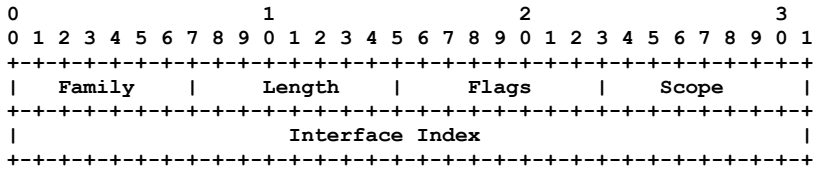
Применимые к данному сервису атрибуты перечислены в таблице.

Атрибут	*Описание*
IFLA_UNSPEC	Не определён.
IFLA_ADDRESS	Аппаратный адрес интерфейса на уровне L2.
IFLA_BROADCAST	Аппаратный широковещательный адрес интерфейса на уровне L2.
IFLA_IFNAME	Имя устройства (строка ASCII).
IFLA_MTU	Значение MTU для устройства
IFLA_LINK	Значение ifindex для канала, к которому подключено устройство.
IFLA_QDISC	Строка ASCII, указывающая имя дисциплины управления выходными очередями.
IFLA_STATS	Статистика для интерфейса.

К данному типу сервиса относятся сообщения Netlink RTM_NEWLINK, RTM_DELLINK и RTM_GETLINK.

2.3.3.2. Модуль службы адресов IP

Эта служба обеспечивает возможность добавления и удаления адресов, а также получения сведений об IP-адресах, связанных с данным интерфейсом. Шаблон сообщения службы предоставления адресов показан на рисунке.



Family - 8 битов

Идентификатор семейства адресов: AF_INET для IPv4 и AF_INET6 для IPv6.

Length - 8 битов

Размер маски адреса.

Flags - 8 битов

Флаг	*Описание*
IFA_F_SECONDARY	Вторичный адрес (псевдоним интерфейса).
IFA_F_PERMANENT	Постоянный адрес, установленный пользователем. Отсутствие этого флага говорит о динамическом выделении адреса (например, с помощью системы автоматической настройки конфигурации без учёта состояния).
IFA_F_DEPRECATED	Недействующий (deprecated) адрес IPv4.
IFA_F_TENTATIVE	Пробный (tentative) адрес IP IPv4 Процедура обнаружения дубликатов адресов находится в стадии разработки.

Scope - 8 битов

Область действия адреса.

SCOPE_UNIVERSE	Адрес глобального действия.
SCOPE_SITE	Адрес действует в пределах данного сайта (только для IPv6).
SCOPE_LINK	Адрес имеет смысл только для данного устройства (канала).
SCOPE_HOST	Адрес имеет смысл только для данного хоста.

Атрибуты сервиса перечислены в таблице.

Атрибут	*Описание*
IFA_UNSPEC	Не определён.
IFA_ADDRESS	Адрес интерфейса для протокола RAW.
IFA_LOCAL	Локальный адрес для протокола RAW.
IFA_LABEL	Имя интерфейса (строка ASCII).
IFA_BROADCAST	Широковещательный адрес для протокола RAW.
IFA_ANYCAST	Anycast-адрес для протокола RAW.
IFA_CACHEINFO	Кэшированная информация об адресе.

К данному типу сервиса относятся сообщения Netlink RTM_NEWADDR, RTM_DELADDR и RTM_GETADDR.

3. Определённые в данный момент IP-службы Netlink

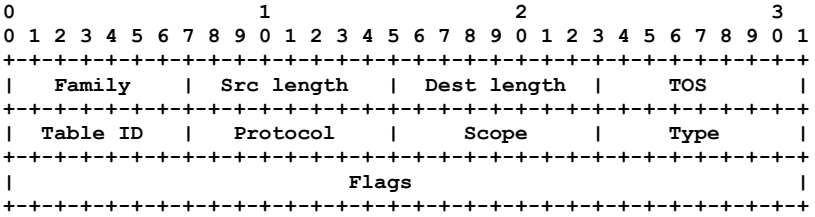
Хотя, как было отмечено выше, существует множество других служб IP, использующих Netlink, в документе рассматривается лишь небольшая часть этих служб, интегрированных в ядро версии 2.4.6[1]. К таким службам относятся NETLINK_ROUTE, NETLINK_FIREWALL и NETLINK_ARPD.

3.1. Служба NETLINK_ROUTE

Эта служба позволяет СРС изменять таблицу маршрутизации IPv4 в машине пересылки FE. Кроме того, данный сервис может применяться СРС для получения данных об обновлении маршрутов и сбора статистики.

3.1.1. Модуль службы маршрутизации

Эта служба обеспечивает возможность создания и удаления маршрутов, а также получения информации о сетевых маршрутах. Формат шаблона сообщения показан на рисунке.



Family - 8 битов

Идентификатор семейства адресов: AF_INET для IPv4 и AF_INET6 для IPv6.

Src length - 8 битов

Размер префикса IP-адреса отправителя.

Dest length - 8 битов

Размер префикса IP-адреса получателя.

TOS - 8 битов

Восьмибитовое поле TOS (следует отказаться от него для освобождения места под DSCP).

Table ID - 8 битов

Идентификатор таблицы. Поддерживается до 255 таблиц маршрутизации.

RT_TABLE_UNSPEC	Неуказанная таблица.
RT_TABLE_DEFAULT	Используемая по умолчанию таблица.
RT_TABLE_MAIN	Основная таблица.
RT_TABLE_LOCAL	Локальная таблица.

Пользователь может выделять произвольные значения из диапазона от RT_TABLE_UNSPEC (0) до RT_TABLE_DEFAULT (253), не включая крайние значения.

Protocol - 8 битов

Указывает, кто добавил маршрут в таблицу.

Протокол	***Источник маршрута***
RTPROT_UNSPEC	Неизвестно.
RTPROT_REDIRECT	Из сообщения ICMP redirect.
RTPROT_KERNEL	Ядро.
RTPROT_BOOT	При загрузке системы.
RTPROT_STATIC	Администратор.

Значения, превышающие RTPROT_STATIC (4)[2], не интерпретируются ядром и включены лишь для информации. Эти значения могут использоваться, чтобы помечать источник маршрутной информации или различать разные демоны маршрутизации. Идентификаторы демонов маршрутизации указаны в файле <linux/rtnetlink.h>.

Scope - 8 битов

Область действия маршрута (корректная дистанция до получателя).

RT_SCOPE_UNIVERSE	Глобальный маршрут.
RT_SCOPE_SITE	Внутренний маршрут локальной автономной системы.
RT_SCOPE_LINK	Маршрут на данном канале (соединении).
RT_SCOPE_HOST	Маршрут на локальном хосте.
RT_SCOPE_NOWHERE	Получателя не существует.

Значения в диапазоне от RT_SCOPE_UNIVERSE (0) до RT_SCOPE_SITE (200), не включая граничные, могут использоваться для пользовательских идентификаторов.

Type - 8 битов

Тип маршрута.

RTN_UNSPEC	Неизвестный маршрут.
RTN_UNICAST	Шлюз или прямой маршрут.
RTN_LOCAL	Маршрут локального интерфейса.
RTN_BROADCAST	Локальный широковещательный маршрут (передача как broadcast).
RTN_ANYCAST	Локальный anycast-маршрут (передача как unicast)
RTN_MULTICAST	Локальный групповой (multicast) маршрут.
RTN_BLACKHOLE	Маршрут для отбрасывания пакетов без уведомления (чёрная дыра).
RTN_UNREACHABLE	Недостижимый получатель. Пакеты отбрасываются с передачей отправителю сообщения ICMP о недоступности адресата.
RTN_PROHIBIT	Запрещённый маршрут. Пакеты отбрасываются с передачей отправителю сообщения ICMP о запрете доступа к адресату.
RTN_THROW	При использовании маршрутизации на базе правил указывает на продолжение просмотра маршрутов в другой таблице. При обычной маршрутизации пакеты отбрасываются с передачей отправителю сообщения ICMP о недоступности адресата.
RTN_NAT	Правило трансляции сетевых адресов.

[1]Информацию о дополнительных службах можно найти в файле <linux/netlink.h>. *Прим. перев.*

[2]В файле <linux/rtnetlink.h> указано, что значение RTPROT_STATIC (4) также не интерпретируется ядром. *Прим. перев.*

RTN_UNSPEC	Неизвестный маршрут.
RTN_XRESOLVE	Указывает на внешний распознаватель (resolver). На момент публикации не реализовано.

Flags - 32 бита

Дополнительная информация о маршруте.

RTM_F_NOTIFY	При изменении маршрута пользователю передаётся уведомление.
RTM_F_CLONED	Маршрут клонирован из другого маршрута.
RTM_F_EQUALIZE	Маршрут допускает случайный выбор следующего интервала (next hop) в случае наличия нескольких путей (не реализовано на момент написания документа).

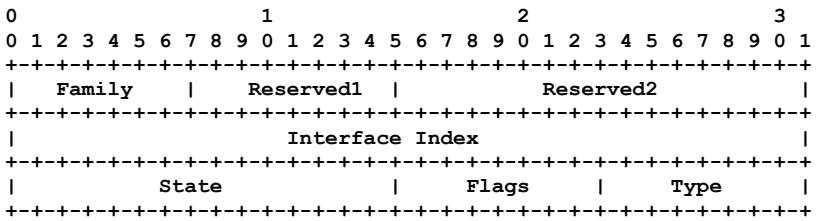
Имеющие отношение к данному сервису атрибуты перечислены в таблице.

Атрибут	*Описание*
RTA_UNSPEC	Игнорируется.
RTA_DST	Протокольный адрес источника маршрута.
RTA_SRC	Протокольный адрес конечной точки маршрута.
RTA_IIF	Индекс входного интерфейса.
RTA_OIF	Индекс выходного интерфейса.
RTA_GATEWAY	Протокольный адрес шлюза для маршрута.
RTA_PRIORITY	Приоритет маршрута.
RTA_PREFSRC	Предпочтительный адрес отправителя при наличии нескольких адресов.
RTA_METRICS	Присвоенная маршруту метрика (например, RTT, начальный размер окна TCP и т. п.).
RTA_MULTIPATH	Атрибуты следующего интервала для маршрута с множеством путей (Multipath route).
RTA_PROTOINFO	Атрибут маршрутизации, основанный на политике межсетевого экрана.
RTA_FLOW	Область маршрута (Route realm).
RTA_CACHEINFO	Кэшированная информация о маршруте.

Для этого типа сервиса поддерживаются дополнительные сообщения Netlink RTM_NEWROUTE, RTM_DELROUTE и RTM_GETROUTE.

3.1.2. Модуль учёта соседей

Этот сервис обеспечивает возможность добавления и удаления записей о соседях (например, ARP, IPv4 neighbor solicitation и т. п.), а также получения информации о существующих записях таблицы соседей. Шаблон сообщений этой службы показан на рисунке.



Family - 8 битов

Идентификатор семейства адресов: AF_INET для IPv4 и AF_INET6 для IPv6.

Interface Index - 32 бита

Уникальный индекс интерфейса.

State - 16 битов

Битовая маска, которая может включать перечисленные в таблице биты состояния.

NUD_INCOMPLETE	Продолжаются попытки распознавания адреса.
NUD_REACHABLE	Подтверждено наличием рабочей записи в кэше.
NUD_STALE	Просроченная запись из кэша.
NUD_DELAY	Сосед больше не достижим. Трафик передан, ожидается подтверждение.
NUD_PROBE	В настоящее время осуществляется запрос на обновление записи в кэше.
NUD_FAILED	Некорректная запись в кэше.
NUD_NOARP	Устройство, которое не выполняет обнаружения соседей (ARP).
NUD_PERMANENT	Статическая запись.

Flags - 8 битов

NTF_PROXY	Запись proxy ARP.
NTF_ROUTER	Маршрутизатор IPv6.

Применимые к этому сервису атрибуты перечислены в таблице.

Атрибут	*Описание*
NDA_UNSPEC	Неизвестный тип.
NDA_DST	Адрес сетевого уровня для кэша соседей.
NDA_LLADDR	Адрес канального уровня для кэша соседей.
NDA_CACHEINFO	Статистика кэширования.

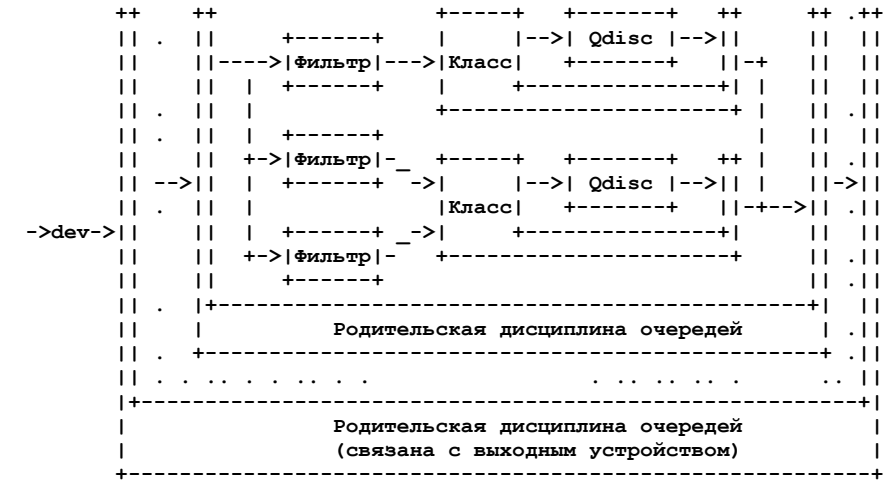
Для этого типа сервиса поддерживаются дополнительные сообщения Netlink RTM_NEWNEIGH, RTM_DELNEIGH и RTM_GETNEIGH.

3.1.3. Служба контроля трафика

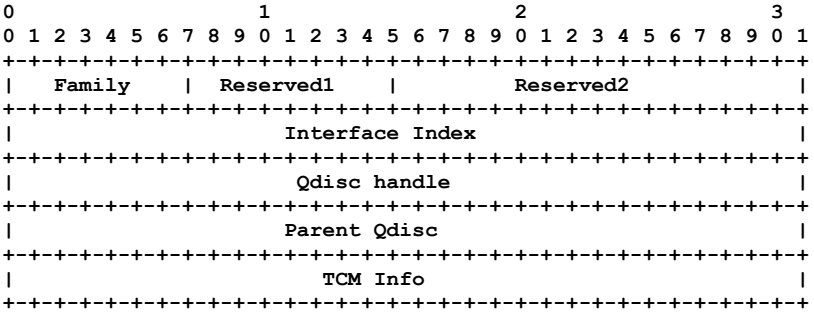
Этот сервис обеспечивает возможность генерации, запроса и прослушивания событий, связанных с контролем трафика. Служба включает дисциплины очередей (планировщики и алгоритмы обслуживания очередей, например, планировщики на основе приоритета или алгоритм RED) и классификаторы трафика. Система управления трафиком Linux обеспечивает высокий уровень гибкости и поддерживает иерархическое каскадирование различных блоков для совместного использования ресурсов каналов передачи трафика.

На рисунке показан пример схемы выходного блока TC. В документе приводится весьма краткое рассмотрение этого вопроса, а дополнительную информацию можно найти в [11]. Пакет сначала проходит через фильтр, используемый для идентификации класса трафика, к которому может быть отнесён данный пакет. Термин «класс» относится к дисциплинам очередей и связан с конкретной очередью. Очередь может использовать простой алгоритм (например,

FIFO) или более сложные механизмы, такие как RED или token bucket. Внешняя дисциплина очереди, которая указывается как родительская, обычно связана с планировщиком. В показанной здесь иерархии планировщик может включать различные алгоритмы планирования, что делает системы управления выходным трафиком ОС Linux очень гибкими.



Шаблон сообщения для этого типа сервиса показан ниже. Этот шаблон используется для дисциплин входных и выходных очередей (относительно модели управления трафиком на выходе, описанной в разделе для модели FE в параграфе 1.1.2.1). Каждый конкретный компонент модели имеет уникальные атрибуты, описывающие его наилучшим способом. Атрибуты общего назначения рассматриваются ниже.



Family - 8 битов

Идентификатор семейства адресов: AF_INET для IPv4 и AF_INET6 для IPv6.

Interface Index - 32 бита

Уникальный индекс интерфейса.

Qdisc handle - 32 бита

Уникальный идентификатор экземпляра дисциплины очередей. Обычно эти идентификаторы рассматриваются как двухкомпонентные (старшая:младшая) по 16 битов в каждой части. Старшая часть номера будет также старшей частью в номере родителя данного экземпляра.

Parent Qdisc - 32 бита

Используется для иерархической структуризации дисциплин очередей. Если это значение совпадает с идентификатором и TC_H_ROOT, данный экземпляр qdisc является называется корневым (Root qdisc).

TCM Info - 32 бита

Для этого поля FE обычно устанавливает значение 1 за исключением тех случаев, когда экземпляр qdisc уже применяется (в этом случае в поле помещается значение счётчика использования данного экземпляра). При передаче со стороны CPC в направлении FEC это поле обычно имеет значение 0 за исключением тех случаев, когда оно используется в контексте фильтрации. В таких случаях 32-битовое поле делится на 16-битовые поля приоритета и протокола. Протоколы определены в исходных кодах ядра (файл <include/linux/if_ether.h>). Наиболее широко используемым протоколом является ETH_P_IP (протокол IP).

Значение приоритета используется для разрешения конфликтов при пересечении фильтрующих выражений.

Базовые атрибуты этого типа сервиса перечислены в таблице.

Атрибут	***Описание***
TCA_KIND	Каноническое имя компонента FE.
TCA_STATS	Базовая статистика использования FEC.
TCA_RATE	Оценка скорости для FEC (расчёт на основе текущего состояния).
TCA_XSTATS	Специфическая статистика FEC.
TCA_OPTIONS	Вложенные атрибуты, связанные с FEC.

В приложении 3 даётся пример конфигурации компонента FE для дисциплины FIFO.

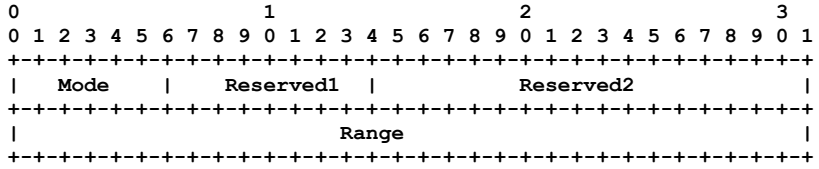
Для этого типа сервиса поддерживаются дополнительные сообщения Netlink RTM_NEWQDISC, RTM_DELQDISC, RTM_GETQDISC, RTM_NEWTCLASS, RTM_DELTCLASS, RTM_GETTCLASS, RTM_NEWTFILTER, RTM_DELTFILTER и RTM_GETTFILTER.

3.2. Служба NETLINK_FIREWALL

Эта служба позволяет CPC принимать пакеты через сервисные модули межсетевого экрана IPv4 в FE, манипулировать этими пакетами и повторно передавать их. Правило межсетевого экрана является первым для активизации перенаправления пакетов. CPC информирует FEC о своём желании получать метаданные для пакета или реальные данные из него, а также сообщает максимальный размер данных, которые будут перенаправляться. Перенаправленные пакеты по-прежнему сохраняются в FEC, ожидая решения о своей судьбе от CPC. Решение может

быть простой командой на восприятие или отбрасывание пакета (в этом случае решение применяется к пакету, все ещё находящемуся в FEC) или включать изменённый пакет, который должен быть передан взамен исходного.

Существует два типа сообщений, передаваемых от CPC к FEC - Mode (режим) и Verdict (решение). Сообщения типа Mode незамедлительно передаются FEC и указывают, что CPC желает принимать от FEC. Сообщения типа Verdict передаются FEC после принятия решения о дальнейшей судьбе полученного пакета. Формат сообщений рассматривается ниже.



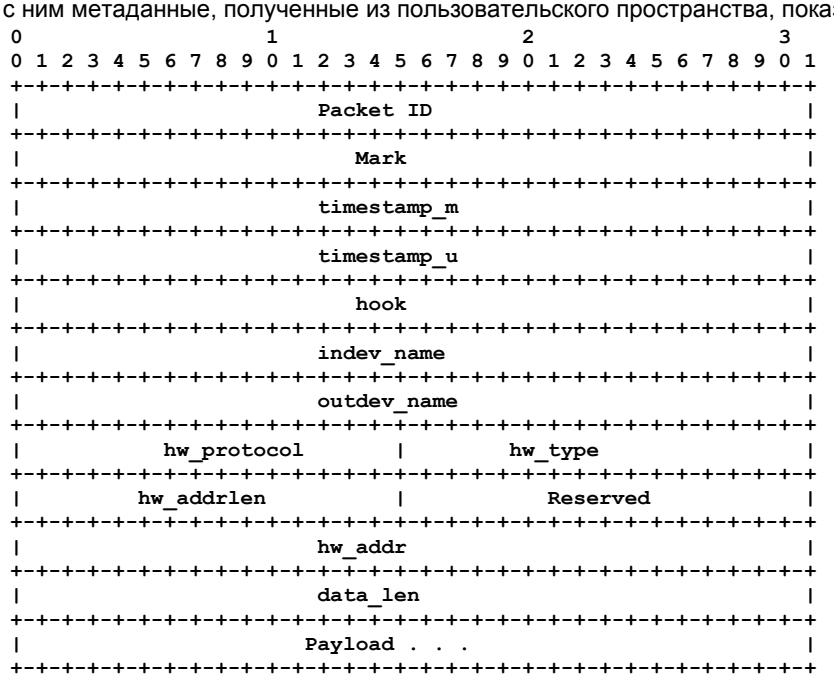
Опишем сначала сообщение, указывающее режим.

Mode - 8 битов

Определяет тип информации в пакетах, отправляемых CPC:
IPQ_COPY_META - копировать в CPC только метаданные пакета;
IPQ_COPY_PACKET - копировать в CPC метаданные и содержимое поля данных пакета.

Range - 32 бита

В режиме IPQ_COPY_PACKET это значение определяет максимальный размер копируемых данных.
Пакет и связанные с ним метаданные, полученные из пользовательского пространства, показаны на рисунке.



Packet ID - 32 бита

Уникальный идентификатор пакета, передаваемый CPC от FEC.

Mark - 32 бита

Значение внутренних метаданных, установленное для описания правила, в котором был отмечен пакет.

timestamp_m - 32 бита

Время прибытия пакета (в секундах).

timestamp_u - 32 бита

Время прибытия пакета (микросекунды, добавляемые к **timestamp_m**).

hook - 32 бита

Модуль межсетевого экрана, в котором был отмечен пакет.

indev_name - 128 битов

Имя приёмного интерфейса (строка ASCII).

outdev_name - 128 битов

Имя выходного интерфейса (строка ASCII).

hw_protocol - 16 битов

Аппаратный протокол (в сетевом порядке битов).

hw_type - 16 битов

Тип оборудования.

hw_addrlen - 8 битов

Размер аппаратного адреса.

hw_addr - 64 бита

Аппаратный адрес.

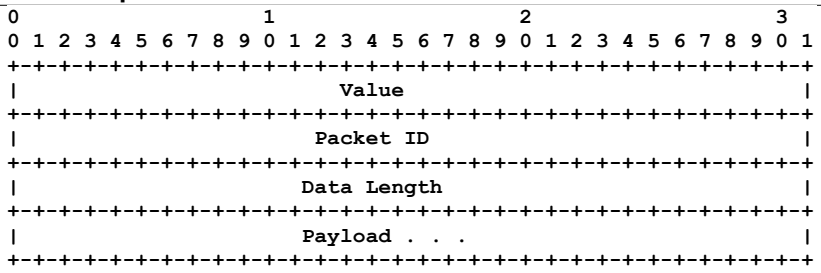
data_len - 32 бита

Размер данных в пакете.

Payload - размер задается полем data_len

Данные из полученного пакета.

Формат сообщений типа Verdict показан на рисунке.



Value - 32 бита

Решение, принятое по отношению к пакету, который по-прежнему находится в FEC. Поддерживаются значения:
NF_ACCEPT - воспринять пакет для дальнейшей обработки;
NF_DROP - отбросить (Drop) пакет.

Packet ID - 32 бита

Уникальный идентификатор пакета, передаваемый CPC от FEC.

Data Length - 32 бита

Размер данных в изменённом пакете (в байтах). Если пакет не был изменён, это поле имеет значение 0.

Payload - данные

Размер определяется значением поля Data Length.

3.3. Служба NETLINK_ARPD

Этот сервис используется CPC для поддержки таблицы соседей в FE. Формат сообщений, передаваемых между FEC и CPC, описан параграфе 3.1.2. Модуль учёта соседей. Предполагается, что сервис CPC принимает участие в работе протоколов организации соседских отношений (neighbor solicitation protocol).

Сообщение типа RTM_NEWNEIGH передаётся CPC от FE для информирования CPC об изменениях, которые могут произойти с записью для этого соседа. Сообщения RTM_GETNEIGH используются для получения информации о конкретном соседе.

4. Литература

4.1. Нормативные документы

[1] Braden, R., Clark, D. and S. Shenker, "Integrated Services in the Internet Architecture: an Overview", RFC 1633, June 1994.

[2] Baker, F., "Requirements for IP Version 4 Routers", RFC 1812, June 1995.

[3] Blake, S., Black, D., Carlson, M., Davies, E, Wang, Z. and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.

[4] Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R. and A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000.

[5] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[6] Case, J., Fedor, M., Schoffstall, M. and C. Davin, "Simple Network Management Protocol (SNMP)", STD 15, RFC 1157, May 1990.

[7] Andersson, L., Doolan, P., Feldman, N., Fredette, A. and B. Thomas, "LDP Specification", RFC 3036, January 2001.

[8] Bernet, Y., Blake, S., Grossman, D. and A. Smith, "An Informal Management Model for DiffServ Routers", RFC 3290, May 2002.

4.2. Дополнительная литература

[9] G. R. Wright, W. Richard Stevens. "TCP/IP Illustrated Volume 2, Chapter 20", June 1995.

[10] http://www.netfilter.org

[11] http://diffserv.sourceforge.net

5. Вопросы безопасности

Netlink работает в доверенной среде на одном хосте с разделением ядра и пользовательского пространства. Средствами Linux обеспечивается возможность открывать сокет только для процессов с флагом возможностей CAP_NET_ADMIN (обычно процессы, запущенные пользователем root).

6. Благодарности

1) Andi Kleen за страницы руководства (man) для netlink и rtnetlink.

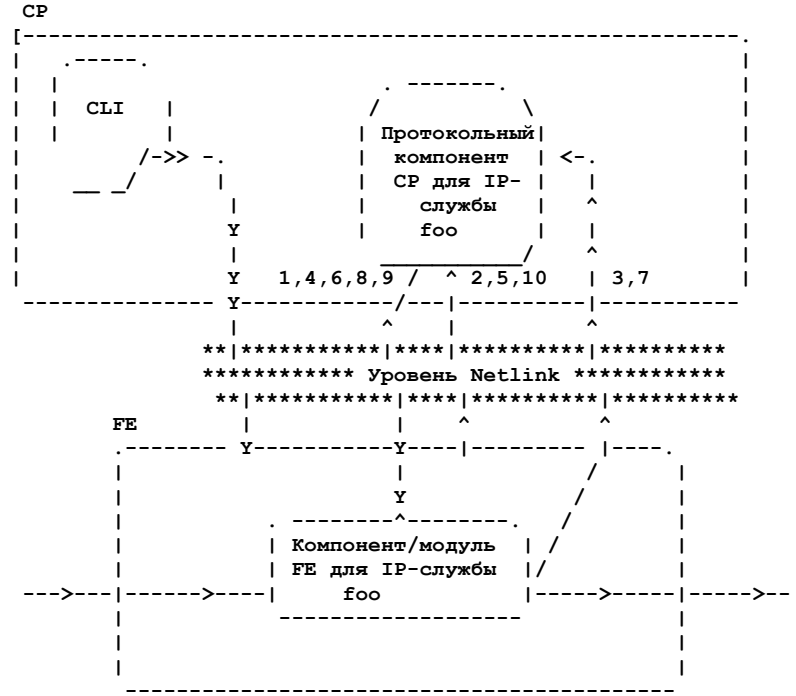
2) Alexey Kuznetsov за добавление модели службы доставки IP в Netlink. Исходный вариант символьного устройства Netlink создал Alan Cox.

3) Jeremy Ethridge за исполнение роли «не понимающего Netlink» и обзор документа с точки зрения его восприятия.

Приложение 1. Пример иерархии служб

На рисунке показан пример единичного IP-сервиса foo и взаимодействие компонентов CP и FE для службы (метки 1-3).

Эта схема служит также примером адресации CP<->FE. В этом приложении иллюстрируется только семантика адресации. В Приложении 2 эта схема рассматривается с точки зрения протокольного взаимодействия между компонентами CPC и FEC сервиса (метки 4-10).



Протокол плоскости управления для IP-службы foo выполняет перечисленные ниже операции для подключения к FE (нумерация в списке соответствует номерам на рисунке).

1) Подключение к IP-сервису foo через сокет. Обычно соединение организуется с помощью вызова socket(AF_NETLINK, SOCK_RAW, NETLINK_FOO).
2) Привязка с целью прослушивания определённых асинхронных событий для сервиса foo.
3) Привязка с целью прослушивания определённых асинхронных событий FE.

Приложение 2. Пример протокола для IP-службы Foo

В этом примере IP-сервис foo используется для демонстрации простого управления сервисом IP с помощью Netlink.

Этапы этого управления выполняются после операций, указанных в Приложении 1, и списки используют общую нумерацию.

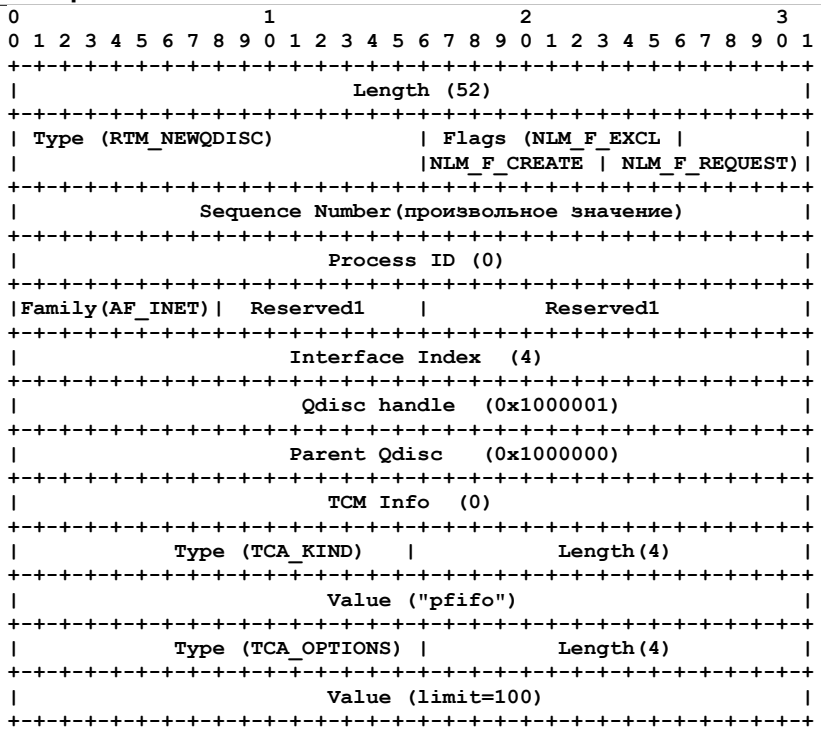
4) Запрашивается текущая конфигурация компоненты FE.
5) Принимается отклик на запрос (4) через канал, организованный на этапе (3).
6) Запрашивается текущее состояние IP-сервиса foo.
7) Принимается отклик на запрос (6) через канал (2).
8) Регистрируются связанные с протоколом пакеты, которые хочется получать от FE.
9) Передаются специфические для данной службы команды foo и (при необходимости) принимаются отклики на них.

Приложение 2a. Взаимодействие с другими службами IP

На схеме в Приложении 1 показан другой компонент, который может настраивать тот же сервис. В данном случае это фирменный командный интерфейс CLI. Интерфейс CLI может (не обязательно) использоваться Netlink для взаимодействия с компонентами foo. Если CLI даёт команды, которые оказывают влияние на политику FEC для сервиса foo, компонент CPC получает уведомления об этом и может принимать решения на их основе. Например, если FE позволяет другому сервису удалять правила, установленные иной службой, и заданные foo правила были удалены сервисом bar, может возникнуть необходимость распространить это всем партнёрам службы foo.

Приложение 3. Пример

В этом примере рассматривается простое конфигурационное сообщение Netlink, передаваемое от TC CPC выходной очереди TC FIFO. Этот алгоритм управления очередью основан на учёте пакетов и отбрасывании их при достижении порогового значения 100. Предполагается, что очередь находится в иерархии с родителем Parent = 100:0, Classid = 100:1 и размещается на устройстве с ifindex = 4.



Адреса авторов

Jamal Hadi Salim
Znyx Networks
Ottawa, Ontario
Canada
EMail: hadi@znyx.com

Hormuzd M Khosravi
Intel
2111 N.E. 25th Avenue JF3-206
Hillsboro OR 97124-5961
USA
Phone: +1 503 264 0334
EMail: hormuzd.m.khosravi@intel.com

Andi Kleen
SuSE
Stahlgruberring 28
81829 Muenchen
Germany
EMail: ak@suse.de

Alexey Kuznetsov
INR/Swsoft
Moscow
Russia
EMail: kuznet@ms2.inr.ac.ru

Перевод на русский язык

Николай Малых
nmalykh@protokols.ru

Полное заявление авторских прав

Подтверждение

Финансирование функций RFC Editor в обеспечено Internet Society.