

Independent Submission
Request for Comments: 9498
Category: Informational
ISSN: 2070-1721

M. Schanzenbach
Fraunhofer AISEC
C. Grothoff
Bernern Fachhochschule
B. Fix
GNUnet e.V.
November 2023

The GNU Name System

Система имён GNU

Аннотация

Этот документ содержит техническую спецификацию системы имён GNU (GNU Name System или GNS). GNS является децентрализованным и устойчивым к цензуре протоколом распознавания доменных имён, являющийся альтернативой протоколам системы доменных имён (Domain Name System или DNS), повышающий степень приватности.

Документ описывает нормативный формат в линии для записей о ресурсах, процессы распознавания, криптографические процедуры, а также соображения безопасности и приватности для использования разработчиками.

Эта спецификация разработана вне IETF и не согласована с IETF. Документ публикуется для информирования читателей о функциях GNS, рекомендация для будущих реализаций GNS и обеспечения функциональной совместимости реализаций (например, имеющихся реализаций GNUnet).

Статус документа

Документ не относится к категории Internet Standards Track и публикуется для информации.

Это вклад в RFC Series, независимый от других потоков RFC. RFC Editor принял решение о публикации документа по своему усмотрению и не делает каких-либо заявлений о его ценности для реализации или внедрения. Документы, одобренные для публикации RFC Editor, не претендуют на статус Internet Standard (см. раздел 2 в RFC 7841).

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке <https://www.rfc-editor.org/info/rfc9498>.

Авторские права

Авторские права (Copyright (c) 2023) принадлежат IETF Trust и лицам, указанным в качестве авторов документа. Все права защищены.

К документу применимы права и ограничения, указанные в BCP 78 и IETF Trust Legal Provisions и относящиеся к документам IETF (<https://trustee.ietf.org/license-info>), на момент публикации данного документа. Прочтите упомянутые документы внимательно. Фрагменты программного кода, включённые в этот документ, распространяются в соответствии с упрощённой лицензией BSD, как указано в параграфе 4.e документа IETF Trust Legal Provisions, без каких-либо гарантий (как указано в Revised BSD License).

Оглавление

1. Введение.....	2
1.1. Уровни требований.....	2
2. Термины.....	3
3. Обзор.....	3
3.1. Имена и зоны.....	4
3.2. Публикация сведений о привязке.....	4
3.3. Распознавание имён.....	5
4. Зоны.....	5
4.1. Домен верхнего уровня для зоны (zTLD).....	6
4.2. Отзыв зоны.....	6
5. Записи о ресурсах.....	8
5.1. Запись делегирования зоны.....	9
5.1.1. PKEY.....	9
5.1.2. EDKEY.....	10
5.2. Записи перенаправления.....	12
5.2.1. REDIRECT.....	12
5.2.2. GNS2DNS.....	12
5.3. Вспомогательные записи.....	12
5.3.1. LEHO.....	13
5.3.2. NICK.....	13
5.3.3. BOX.....	13
6. Кодирование записей для удалённого хранилища.....	14
6.1. Ключ хранилища.....	14
6.2. Текстовые данные записей (RDATA).....	15
6.3. Блок записей о ресурсах.....	15
7. Распознавание имени.....	16

7.1. Стартовая зона.....	16
7.2. Рекурсия.....	17
7.3. Обработка записей.....	17
7.3.1. REDIRECT.....	18
7.3.2. GNS2DNS.....	18
7.3.3. BOX.....	18
7.3.4. Записи делегирования зон.....	18
7.3.5. NICK.....	19
8. Другие языки и кодировка символов.....	19
9. Вопросы приватности и безопасности.....	19
9.1. Доступность.....	19
9.2. Стойкость.....	19
9.3. Криптография.....	19
9.4. Смягчение злоупотреблений.....	20
9.5. Поддержка зон.....	20
9.6. DHT как удалённое хранилище.....	20
9.7. Отзывы.....	20
9.8. Приватность зоны.....	21
9.9. Управление зонами.....	21
9.10. Неоднозначность пространства имён.....	21
10. Взаимодействие с GANA.....	21
10.1. Реестр GNUet Signature Purposes.....	21
10.2. Реестр GNS Record Types.....	22
10.3. Реестр субдоменов .alt.....	22
11. Взаимодействие с IANA.....	22
12. Статус реализации и внедрения.....	23
13. Литература.....	23
13.1. Нормативные документы.....	23
13.2. Дополнительная литература.....	24
Приложение А. Использование и переход.....	25
А.1. Распространение зон.....	25
А.2. Конфигурация Start Zone.....	25
А.3. Глобально уникальные имена и Web.....	25
А.4. Пути перехода.....	26
Приложение В. Примеры.....	26
В.1. Пример распознавания AAAA.....	26
В.2. Пример распознавания REDIRECT.....	27
В.3. Пример распознавания GNS2DNS.....	27
Приложение С. Base32GNS.....	28
Приложение D. Тестовые векторы.....	28
D.1. Кодирование и декодирование Base32GNS.....	28
D.2. Наборы записей.....	28
D.3. Отзыв зоны.....	35
Благодарности.....	37
Адреса авторов.....	37

1. Введение

Эта спецификация описывает GNS - устойчивый к цензуре и сохраняющий приватность распределенный протокол распознавания доменных имён. GNS криптографически защищает привязку к именам произвольных маркеров что позволяет протоколу в определённых отношениях выступать в качестве альтернативы некоторым современным инфраструктурам открытых ключей.

В терминологии системы доменных имён (Domain Name System или DNS) [RFC1035] GNS примерно соответствует идее развёртывания локальной корневой зоны (см. [RFC8806]), отличаясь тем, что разрешаются дополнительные (альтернативные) корни и не предполагается, что все внедрения будут использовать одну и ту же или какую-то конкретную корневую зону. В эталонной реализации GNS пользователи могут автономно и свободно передавать полномочия (делегировать) по управлению именами в зоны через локальную конфигурацию. В GNS предполагается, что каждый пользователь контролирует свою установку. В соответствии с рекомендациями параграфа 9.10 пользователям следует избегать путаницы в способах распознавания имён.

Распознавание имён и распространение зон основано на принципе именования домашних питомцев, где пользователи могут назначать зонам локальные имена. В основе GNS лежат идеи простой распределенной инфраструктуры защиты (Simple Distributed Security Infrastructure) [SDSI], позволяющей децентрализованно сопоставлять идентификаторы защиты с запоминаемыми именами. Одно из первых академических описаний криптографических идей, лежащих в основе GNS, приведено в [GNS].

Документ описывает нормативный формат в линии для записей о ресурсах, процессы распознавания, криптографические процедуры, а также соображения безопасности и приватности для использования разработчиками.

1.1. Уровни требований

Ключевые слова **должно** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не следует** (SHALL NOT), **следует** (SHOULD), **не нужно** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **не рекомендуется** (NOT RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе интерпретируются в соответствии с BCP 14 [RFC2119] [RFC8174] тогда и только тогда, когда они выделены шрифтом, как показано здесь.

2. Термины

Apex Label - метка вершины

Этот тип метки служит для публикации записей о ресурсах в зоне, которые можно распознать без предоставления конкретной метки. Это метод GNS для реализации того, что в DNS называется вершиной зоны (zone apex) [RFC4033]. Метка вершины представляется с использованием символа U+0040 (@).

Application - приложение

Компонент, применяющий реализацию GNS для преобразования имён в записи и обработки их содержимого.

Blinded Zone Key - слепой ключ зоны

Ключ, выведенный из ключа зоны и метки. Ключ зоны и любой выведенный из него слепой ключ зоны не могут быть связаны между собой без знания конкретной метки, использованной при выводе.

Extension Label - метка расширения

Тип метки, служащий для указания полномочной зоны, в которой находится запись. Метки расширения применяются в основном при перенаправлениях, когда цель перенаправления указана относительно полномочной зоны из записи о перенаправлении (см. параграф 5.2). Метка расширения представляется символом U+002B (+).

Label Separator - разделитель меток

Для разделения меток в именах применяется символ точки U+002E (.). В GNS каждый разделитель в имени указывает передачу полномочий (делегирование) в другую зону, за исключением зоны доменов верхнего уровня (zone Top-Level Domain или zTLD, см. ниже) и коробочных (boxed) записей (см. параграф 5.3.3).

Label - метка

Метки GNS соответствуют определению из [RFC8499]. Метки являются строками UTF-8 в нормализованной форме C (Unicode Normalization Form C или NFC) [Unicode-UAX15]. Метки вершины и расширения имеют особое значение в протоколе распознавания, заданном далее в этом документе. Администраторы зон **могут** с помощью процедур регистрации запретить некоторые метки, которые легко спутать с другими (см. параграф 9.4).

Name - имя

Имя в GNS является доменным именем в соответствии с [RFC8499] - строка UTF-8 [RFC3629], представляющая собой упорядоченный набор меток через символ-разделитель. Имена распознаются, начиная с правой метки. GNS не вносит ограничений на размер имён и меток, однако приложения **могут** обеспечивать совместимость размеров имён и меток с DNS и, в частности, с именами на национальных языках (Internationalized Domain Names for Applications или IDNA) [RFC5890]. В духе [RFC5895] приложения **могут** предварительно обрабатывать имена и метки для обеспечения совместимости с DNS или поддержки определённых ожиданий пользователей, например, соответствия [Unicode-UTS46]. Имя GNS может быть неотличимо от имени DNS поэтому приложения и разработчики должны осторожно относиться к обработке имён GNS (см. параграф 9.10). Для предотвращения ложной интерпретации имён а примерах этого документа как (зарезервированных) имён DNS, здесь применяется суффикс .gns.alt в соответствии с [RFC9476], включенный также в реестр GANA .alt Subdomains [GANA].

Resolver - распознаватель

Компонент реализации GNS, обеспечивающий рекурсивное распознавание имён на основе логики из раздела 7.

Resource Record - запись о ресурсе

Запись о ресурсе GNS представляет собой сведения, связанные с меткой в зоне GNS. Содержимое записи о ресурсе GNS определяется типом этой записи.

Start Zone - стартовая зона

Для распознавания любого конкретного имени GNS нужно определить для него исходную стартовую зону (Start Zone). Стартовую зону можно задать явно как часть имени с помощью zTLD или определить через локальной сопоставление суффиксов с зонами (см. параграф 7.1).

Top-Level Domain (TLD) - домен верхнего уровня

Самая правая часть в имени GNS называется GNS TLD и может состоять из одной или нескольких меток. В отличие от DNS TLD (см. [RFC8499]), в GNS не предполагается использование одной глобальной корневой зоны всеми пользователями. Вместо этого GNS TLD (за исключением zTLD, см. параграф 4.1) обычно являются частью конфигурации локального распознавателя (см. параграф 7.1) и может не быть глобально уникальным.

Zone - зона

Зона GNS содержит полномочные (authoritative) сведения (записи о ресурсах). Зона однозначно идентифицируется её ключом. В отличие от зон DNS зонам GNS не требуется иметь запись SOA под меткой вершины (apex).

Zone Key - ключ зоны

Ключ, однозначно указывающий зону. Обычно это открытый ключ из асимметричной пары. Однако устоявшийся технический термин «открытый ключ» (public key) вводит в заблуждение, поскольку в зоне GNS ключ может быть общим секретом, который не следует раскрывать неполномоченным сторонам.

Zone Key Derivation Function - функция вывода ключа зоны

Функция вывода ключа зоны (zone key derivation function или ZKDF) скрывает ключ зоны при использовании метки.

Zone Publisher - издатель зоны

Компонент реализации GNS, обеспечивающие управление и публикацию локальной зоны, как указано в разделе 6.

Zone Owner - владелец зоны

Держатель секрета (обычно, секретного ключа), который в сочетании с меткой и значением для подписи позволяет создавать подписи зоны, которые можно проверить с помощью соответствующего спрятанного ключа зоны.

Zone Top-Level Domain (zTLD)

GNS zTLD - это последовательность меток GNS в конце имени GNS. zTLD кодирует тип и ключ зоны (см. параграф 4.1). Благодаря статистической уникальности ключей зон, zTLD уникальны в глобальном масштабе. Последовательность меток zTLD можно отличить от обычной последовательности меток TLD лишь при попытке декодировать метки в тип и ключ зоны.

Zone Type - тип зоны

Тип зоны GNS определяет систему шифрования и формат двоичного кодирования ключа зоны, спрятанных ключей зоны и криптографических подписей.

3. Обзор

Система именования GNS имеет три базовых свойства, указанных ниже.

Глобальность на основе концепции zTLD

Зона однозначно указывается её ключом и статистически уникальна, поэтому zTLD являются глобально уникальными отображениями зон, а имена доменов GNS с суффиксом zTLD глобально уникальны, но не являются запоминающимися.

Запоминающиеся имена зон

Пользователи могут настраивать для зон локальные, легко запоминающиеся ссылки. Такие имена служат псевдонимами (moniker) zTLD, обеспечивающими удобные имена зон для локального оператора. Эти имена могут также публиковаться как предложения для других пользователей, ищущих подходящую метку для ссылок на соответствующую зону.

Защищённое сопоставление имён с записями

GNS позволяет владельцам зон сопоставлять метки с записями о ресурсах или передавать полномочия управления именами в субдомене, указываемом меткой, в другие зоны. Владельцы зон могут публиковать такие сведения для информирования других пользователей. Сопоставления шифруются и подписываются с использованием ключей, выведенных из соответствующей метки до публикации в удалённом хранилище. При распознавании имён подписи в записях о ресурсах, включая делегирование, проверяются рекурсивным распознавателем.

Далее в документе применяется термин «исполнитель» (implementer) для обозначения разработчика реализации GNS, включающей распознаватель, издатель зоны и поддерживающие конфигурации, такие как Start Zone (параграф 7.1).

3.1. Имена и зоны

Из сказанного выше ясно, что GNS не поддерживает имена, которые одновременно являются глобальными, защищёнными и запоминаемыми. Т. е. имена или глобально, но не запоминаемы, или запоминаемы, но не уникальны. Пример глобального имени, указывающего на запись example в зоне, приведён ниже.

```
example.000G006K2TJNMD9VTCYRX7BRVV3NAEPS15E6NHDХКРJA1KAJLEG9AFF884
```

Рассмотрим случай, где пользователь локально настроил имя pet.gns.alt для зоны с записью example, представленной выше. Имя example.pet.gns.alt будет указывать на ту же запись, что и глобально уникальное имя, приведённое выше, но распознавание имён будет работать лишь в локальной системе, где настроено имя pet.gns.alt.

Делегирование имён (petname) и последующее его распознавание основано на идеях из простой распределенной инфраструктуры защиты (Simple Distributed Security Infrastructure) [SDSI]. В GNS любой пользователь может создать и поддерживать любое число зон (см. раздел 4), если его система поддерживает реализацию издателя зон. Для каждой зоны её тип определяет соответствующий набор криптографических операций и формат передачи зашифрованных данных, открытых ключей и подписей. Зона может быть заполнена её владельцем отображениями меток на записи о ресурсах (см. раздел 5). Метка может отображаться на запись о делегировании, в результате чего соответствующий субдомен делегируется в другую зону. В явном виде разрешена циклическая передача полномочий, включая делегирование субдомена в его непосредственно родительскую зону. Для поддержки (унаследованных) приложений и облегчения использования имён (petname) GNS определяет вспомогательные типы записей с дополнением к поддержке имеющихся записей DNS.

3.2. Публикация сведений о привязке

Содержимое зоны шифруется и подписывается перед публикацией в удалённом хранилище ключей и значений (см. параграф 6), как показано на рисунке 1. В этом процессе уникальная идентификация зоны скрывается от сети за счёт применения привязки ключей. Привязка позволяет создавать подписи для содержимого зоны, используя «ослепленную» пару открытого и секретного ключа. Эта привязка реализуется использованием детерминированного вывода ключа из исходного ключа зоны и соответствующего секретного ключа с использованием значений меток в качестве входных данных для вывода коэффициентов сокрытия. В частности, владелец зоны может вывести ослепленные секретные ключи для каждого набора записей, опубликованного под меткой, а распознаватель может вывести соответствующие ослепленные открытые ключи. Предполагается, что реализации GNS используют децентрализованные структуры удалённого хранения, такие как распределенные хэш-таблицы (distributed hash table или DHT) для обеспечения доступности внутри сети без необходимости в выделенной инфраструктуре. Спецификация такого распределенного или децентрализованного хранилища выходит за рамки этого документа. Возможные реализации включают варианты, основанные на [RFC7363], [Kademlia] или [R5N].

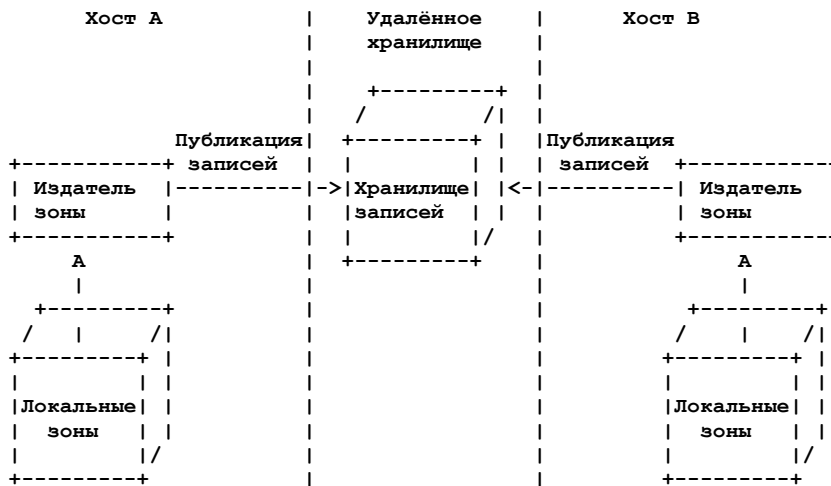


Рисунок 1. Пример с двумя хостами, публикующими зоны GNS.

Реализацию издателя зон **следует** предоставлять как часть реализации GNS, чтобы пользователи могли создавать и поддерживать зоны. Если такая функциональность не реализована, имена все равно могут быть распознаны при настройке ключей зоны для начальной установки распознавания имён (см. параграф 7) или имена имеют суффикс zTLD.

4.1. Домен верхнего уровня для зоны (zTLD)

Строка zTLD кодирует тип и ключ зоны в суффиксе доменного имени и служит уникальной в глобальном масштабе ссылкой на зону в процессе распознавания имён. zTLD создаётся путём кодирования двоичной конкатенации типа и ключа зоны (см. рисунок 3). Для кодирования служит вариант Crockford Base32 [CrockfordB32], называемый Base32GNS. Символы кодирования и декодирования для Base32GNS (включая этот вариант) даны в таблице 4 (Приложение C). Функции кодирования и декодирования на основе таблицы 4 называются Base32GNS-Encode и Base32GNS-Decode, соответственно.

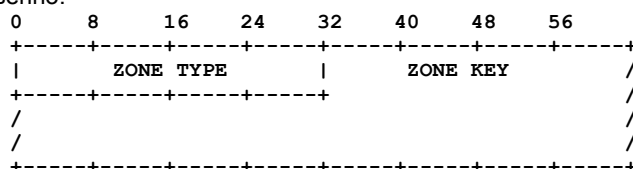


Рисунок 3. Двоичное представление zTLD.

Поле ZONE TYPE **должно** кодироваться с сетевым порядком байтов. Формат ZONE KEY целиком зависит от ZONE TYPE. Таким образом zTLD кодируется и декодируется, как показано ниже.

```
zTLD := Base32GNS-Encode(ztype||zkey)
ztype||zkey := Base32GNS-Decode(zTLD)
```

Здесь || указывает конкатенацию.

zTLD можно использовать «как есть» в качестве крайней справа метки в имени GNS. Если приложение хочет обеспечить совместимость имён с DNS, оно может представлять zTLD размером не более 63 символов без изменения, а более длинные zTLD разбиваются символом-разделителем на более короткие метки (старшие байты конкатенации ztype||zkey должны включаться в правую метку полученной в результате строки, а младшие - в левую). Это позволяет распознавателю определить ztype и размер zTLD по правой метке, а затем определить число меток в zTLD. Реализации GNS **должны** поддерживать деление zTLD на метки, размер которых совместим с DNS. Например, zTLD из 130 символов можно представить как zTLD[126..129].zTLD[63..125].zTLD[0..62]

4.2. Отзыв зоны

Для отзыва ключа зоны **должно** быть опубликовано отзывающее сообщение, которое **должно** подписываться с применением секретного ключа зоны. Отзывающее сообщение широковещательно передаётся в сеть. Спецификация механизма широковещательной отправки выходит за рамки этого документа. Один из возможных механизмов широковещательной передачи в распределённую сеть реализован в [GUnet]. Как вариант, отзывающие сообщения могут распространяться через распределённый реестр (ledger) или доверенный центральный сервер. Для предотвращения лавинных атак сообщение об отзыве **должно** включать доказательство работы (proof of work или PoW). Сообщение об отзыве, включая PoW, **может** создаваться заранее для поддержки своевременного отзыва.

Во всех приведённых ниже случаях Argon2id указывает функцию вывода ключа по паролю, заданную в [RFC9106]. Для расчёта PoW алгоритм применяется с указанными ниже параметрами.

- S** Затравка (salt) в форме неизменной 16-байтовой строки GnsRevocationPow.
- t** Число итераций (3).
- m** Размер памяти в KiB (1024).
- T** Размер выходного хэша в байтах (64)
- p** Параметр распараллеливания (1)
- v** Версия алгоритма (0x13)
- y** Тип алгоритма (Argon2id) (2)
- X** Не используется.
- K** Не используется.

На рисунке 4 показан формат данных P, по которым рассчитывается PoW.

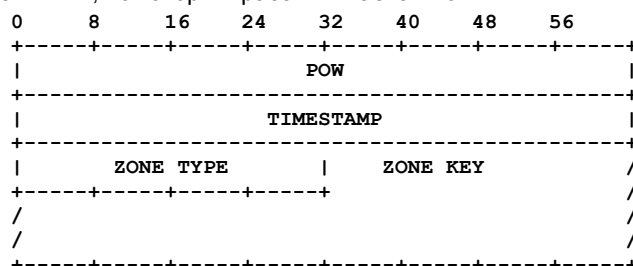


Рисунок 4. Формат данных PoW.

POW

64-битовое значение, которое является решением для PoW (сетевой порядок байтов).

TIMESTAMP

64-битовое значение абсолютной даты расчёта отзывающего сообщения (время с полуночи 1 января 1970 г. UTC в микросекндах) с сетевым порядком байтов.

CRITICAL

Установленный флаг означает, что обработка является критической. Реализация, не поддерживающая этот тип записи или не способная обработать данную запись, **должна** прервать распознавание встретившейся записи.

SHADOW

Если этот флаг установлен, запись **должна** игнорироваться распознавателями, пока не истечёт срок действия всех (иных) записей того же типа. Флаг служит для того, чтобы издатели зон могли обеспечивать хорошую производительность при изменении записей, позволяя помещать в хранилище будущие значения. За счёт этого будущие значения могут распространяться и кэшироваться до перехода на них.

SUPPLEMENTAL

Флаг дополнительной записи, указывающий, что запись не поддерживается явно вместе с другими записями под соответствующим именем, но может быть полезна для приложения.

5.1. Запись делегирования зоны

В этом параграфе задаётся исходный набор типов записей делегирования зон. Каждой реализации **следует** поддерживать все определённые здесь типы и **можно** поддерживать любое число записей делегирования из реестра GANA GNS Record Types [GANA]. Отсутствие поддержки какого-либо типа приведёт к отказам при распознавании, если в процессе встретится соответствующая зона. Это может быть оправданным выбором, если некоторые типы записей делегирования сочтены криптографически незащищёнными. Записи делегирования зон **недопустимо** сохранять или публиковать под меткой вершины (арех). Значение типа записи делегирования совпадает с соответствующим *ztype*. Значение *ztype* определяет криптографические примитивы для делегируемой зоны. Содержимое записи делегирования зоны включает открытый ключ делегируемой зоны. В записи делегирования зоны **должен** быть установлен флаг CRITICAL и она **должна** быть единственной не дополнительной (non-supplemental) записью под меткой. **Могут** быть другие неактивные записи того же типа с установленным флагом SHADOW для упрощения плавной смены ключей.

Далее в документе символы || указывают конкатенацию двух байтовых строк. В спецификации алгоритма используются строки символов, такие как метки GNS и значения констант. При их использовании в конкатенации или в качестве входных данных функции **недопустимо** включать завершающие null-символы.

5.1.1. PKEY

В GNS делегирование метки в зону типа PKEY представляется записью PKEY. Формат передачи PKEY DATA показан на рисунке 9.

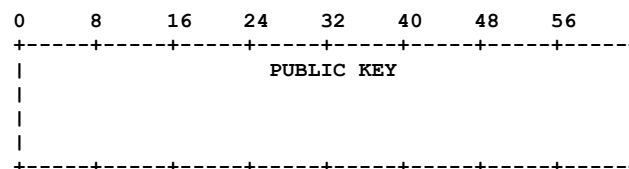


Рисунок 9. Формат передачи PKEY.

PUBLIC KEY

256-битовый открытый ключ Ed25519.

Для зон PKEY ключевой материал выводится с использованием параметров кривой «скрученного» (twisted) представления Edwards для Curve25519 [RFC7748] (причина выбора этой кривой описана в параграфе 9.3) со схемой ECDSA [RFC6979]. Для криптографических примитивов зон PKEY используются приведённые ниже обозначения.

d

256-битовый секретный ключ Ed25519 (секретный зажатый скаляр).

zkey

Открытый ключ зоны Ed25519, соответствующий *d*.

P

Простое число (prime) *edwards25519*, как задано в [RFC7748], т. е. $2^{255} - 19$.

G

Генератор группы $(X(P), Y(P))$ с $X(P), Y(P)$ *edwards25519* в соответствии с [RFC7748].

L

Порядок подгруппы prime-order *edwards25519* в соответствии с [RFC7748].

KeyGen()

Генерация секретного скаляра *d* и точки кривой *zkey* := *d***G* (*G* - генератор группы эллиптической кривой), как задано в параграфе 2.2 [RFC6979], представляет функцию KeyGen().

Тип и ключ зоны PKEY имеют размер 4 + 32 байтов. Это означает, что *zTLD* всегда будет помещаться в одну метку и дополнительного преобразования не требуется. С учётом метки вывод *zkey'* функции ZKDF(*zkey*, *label*) рассчитывается для зон PKEY, как показано ниже.

```

ZKDF(zkey, label) :
  PRK_h := HKDF-Extract("key-derivation", zkey)
  h := HKDF-Expand(PRK_h, label || "gns", 512 / 8)
  zkey' := (h mod L) * zkey
  return zkey'

```

Криптосистема PKEY использует основанную на HMAC функцию вывода ключа (HKDF) в соответствии с [RFC5869], применяя SHA-512 [RFC6234] в фазе извлечения и SHA-256 [RFC6234] в фазе преобразования (expansion). PRK_h - это ключевой материал, полученный с помощью HKDF, где применяется строка *key-derivation* в качестве затравки (salt) и ключ зоны в качестве исходного ключевого материала. 512-битовый результат преобразования HKDF (*h*) должен интерпретироваться с сетевым порядком байтов. Входные данные преобразования - это конкатенация метки и строки *gns*. Умножение *zkey* на *h* в функции ZKDF() является точечным умножением, а умножение *d* на *h* в SignDerived() - скалярным.

Функции Sign() и Verify() для зон PKEY реализованы с использованием 512-битовых детерминированных подписей ECDSA, как указано в [RFC6979]. Те же функции могут применяться для производных ключей.

```

SignDerived(d, label, message):
  zkey := d * G
  PRK_h := HKDF-Extract("key-derivation", zkey)
  h := HKDF-Expand(PRK_h, label || "gns", 512 / 8)
  d' := (h * d) mod L
  return Sign(d', message)

```

Подпись действительна для производного открытого ключа $zkey' := ZKDF(zkey, label)$, если выполняется

```

VerifyDerived(zkey', message, signature):
  return Verify(zkey', message, signature)

```

Функции S-Encrypt() и S-Decrypt() используют AES в режиме счётчика, как задано в [MODES] (CTR-AES256)

```

S-Encrypt(zkey, label, expiration, plaintext):
  PRK_k := HKDF-Extract("gns-aes-ctx-key", zkey)
  PRK_n := HKDF-Extract("gns-aes-ctx-iv", zkey)
  K := HKDF-Expand(PRK_k, label, 256 / 8)
  NONCE := HKDF-Expand(PRK_n, label, 32 / 8)
  BLOCK_COUNTER := 0x0000000000000001
  IV := NONCE || expiration || BLOCK_COUNTER
  return CTR-AES256(K, IV, plaintext)

```

```

S-Decrypt(zkey, label, expiration, ciphertext):
  PRK_k := HKDF-Extract("gns-aes-ctx-key", zkey)
  PRK_n := HKDF-Extract("gns-aes-ctx-iv", zkey)
  K := HKDF-Expand(PRK_k, label, 256 / 8)
  NONCE := HKDF-Expand(PRK_n, label, 32 / 8)
  BLOCK_COUNTER := 0x0000000000000001
  IV := NONCE || expiration || BLOCK_COUNTER
  return CTR-AES256(K, IV, ciphertext)

```

Ключ K и вектор инициализации (Initialization Vector или IV) счётчика выводятся из метки записи и ключа зоны $zkey$ с использованием HKDF, как задано в [RFC5869]. В фазе извлечения применяется SHA-512 [RFC6234], а в фазе преобразования (expansion) - SHA-256 [RFC6234]. Выходной ключевой материал - это 32 байта (256 битов) для симметричного ключа и 4 байта (32 бита) для NONCE. Симметричный ключ K - это 256-битовый ключ AES [RFC3826].

Значение nonce объединяется с 64-битовым IV и 32-битовым счётчиком блоков, как задано в [RFC3686]. Счет блоков начинается с 1 и значение инкрементируется при генерации каждой следующей части потока ключей. Счётчик блоков - это 32-битовое целое число с сетевым порядком байтов. Формат IV счётчика, примененного в функциях S-Encrypt() и S-Decrypt(), показан на рисунке 10.

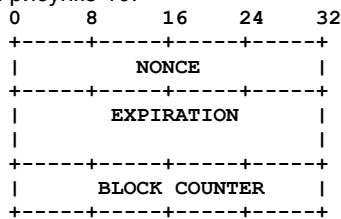


Рисунок 10. Структура Counter IV при использовании в S-Encrypt() и S-Decrypt().

5.1.2. EDKEY

В GNS делегирование метки в зону типа EDKEY представляется записью EDKEY (рисунок 11).

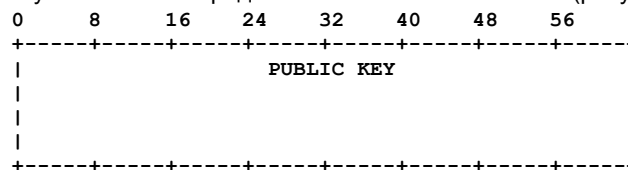


Рисунок 11. Формат передачи EDKEY DATA.

PUBLIC KEY

256-битовый ключ зоны EdDSA.

Для зон EDKEY ключевой материал выводится с использованием параметрой кривой «скрученного» представления Edwards для Curve25519 [RFC7748] (или Ed25519) со схемой Ed25519 [ed25519], как указано в [RFC8032]. Для криптографических примитивов зон EDKEY используются приведённые ниже обозначения.

d

256-битовый секретный ключ EdDSA.

a

Целое число, выведенное из d с использованием хэш-функции SHA-512 в соответствии с [RFC8032].

zkey

Открытый ключ EdDSA, соответствующий d . Ключ определяется как точка кривой $a * G$, где G - генератор группы эллиптической кривой, как задано в [RFC8032].

p

Простое число (prime) edwards25519, как задано в [RFC7748], т. е. 2255 - 19.

G

Генератор группы $(X(P), Y(P))$ с $X(P), Y(P)$ edwards25519 в соответствии с [RFC7748].

L

Порядок подгруппы prime-order edwards25519 в соответствии с [RFC7748].

KeyGen()

Генерация секретного скаляра d и связанного открытого ключа $zkey := a * G$ (G - генератор группы эллиптической кривой, a - целое число, полученное из d с помощью хэш-функции SHA-512) в соответствии с параграфом 5.1.5 в [RFC8032] представляет функцию KeyGen().

Тип и ключ зоны EDKEY имеют размер 4 + 32 байтов. Это означает, что zTLD всегда будет помещаться в одну метку и дополнительного преобразования не требуется.

Создание EDKEY с ZKDF основано на [Tor224]. Как указано выше для KeyGen(), расчёт выполняется на основе d с использованием хэш-функции SHA-512, как задано в параграфе 5.1.5 [RFC8032]. С учётом метки вывод функции ZKDF рассчитывается, как показано ниже.

```
ZKDF(zkey, label):
/* Расчёт коэффициент ослепления */
PRK_h := HKDF-Extract("key-derivation", zkey)
h := HKDF-Expand(PRK_h, label || "gns", 512 / 8)
/* Обеспечение  $h == \bar{h} \bmod L$  */
h := h mod L

zkey' := h * zkey
return zkey'
```

Разработчикам **следует** применять скалярное умножение с постоянным временем (constant-time) для приведённых выше конструкций, чтобы защититься от атак с синхронизацией. В противном случае такие атаки могут приводить к утечке секретного цифрового материала, если атакующий сможет предсказать начало процесса публикации системой.

Криптосистема EDKEY использует HKDF в соответствии с [RFC5869], применяя SHA-512 [RFC6234] в фазе извлечения и HMAC-SHA-256 [RFC6234] в фазе преобразования. Ключевой материал PRK_h выводится с помощью HKDF, где строка key-derivation служит затравкой, а ключ зоны - исходным ключевым материалом. Фактором ослепления h является 512-битовый результат преобразования HKDF. Входом для преобразования служит конкатенация метки и строки gns. Результат HKDF должен «зажиматься» и интерпретироваться в сетевом порядке байтов. 256-битовое целое число a соответствует 256-битовому секретному ключу d. Умножение zkey на h является точечным.

Процедуры Sign(d, message) и Verify(zkey, message, signature) **должны** быть реализованы в соответствии с [RFC8032].

Подписи для зон EDKEY используют секретный производный скаляр d' и это не соответствует [RFC8032]. Поскольку секретный ключ, соответствующий секретному производному скаляру, неизвестен, невозможно детерминированно вывести часть подписи R в соответствии с [RFC8032]. Вместо этого генерация подписи для любого данного сообщения и секретного ключа зоны **должна** выполняться путём расчёта поNonce из 32 старших байтов преобразования секретного ключа d и коэффициента ослепления h. Значение поNonce затем хэшируется с сообщением, давая значение r. Таким образом, полный путь вывода включается в расчёт значения подписи R, гарантируя, что он не будет использован повторно для двух разных путей вывода или сообщений.

```
SignDerived(d, label, message):
/* Преобразование ключа */
dh := SHA-512(d)
/* Зажатие EddSA */
a := dh[0..31]
a[0] := a[0] & 248
a[31] := a[31] & 127
a[31] := a[31] | 64
/* Расчёт zkey, соответствующего d */
zkey := a * G

/* Расчёт коэффициент ослепления */
PRK_h := HKDF-Extract("key-derivation", zkey)
h := HKDF-Expand(PRK_h, label || "gns", 512 / 8)
/* Обеспечение  $h == \bar{h} \bmod L$  */
h := h mod L

d' := (h * a) mod L
nonce := SHA-256(dh[32..63] || h)
r := SHA-512(nonce || message)
R := r * G
S := r + SHA-512(R || zkey' || message) * d' mod L
return (R,S)
```

Подпись (R,S) действительна для производного открытого ключа zkey' := ZKDF(zkey, label) при выполнении условия

```
VerifyDerived(zkey', message, signature):
(R,S) := signature
return S * G == R + SHA-512(R, zkey', message) * zkey'
```

Функции S-Encrypt() и S-Decrypt() используют XSalsa20 в соответствии с [XSalsa20] и функцию шифрования XSalsa20-Poly1305.

```
S-Encrypt(zkey, label, expiration, plaintext):
PRK_k := HKDF-Extract("gns-xsalsa-ctx-key", zkey)
PRK_n := HKDF-Extract("gns-xsalsa-ctx-iv", zkey)
K := HKDF-Expand(PRK_k, label, 256 / 8)
NONCE := HKDF-Expand(PRK_n, label, 128 / 8)
IV := NONCE || expiration
return XSalsa20-Poly1305(K, IV, plaintext)
```

```
S-Decrypt(zkey, label, expiration, ciphertext):
PRK_k := HKDF-Extract("gns-xsalsa-ctx-key", zkey)
PRK_n := HKDF-Extract("gns-xsalsa-ctx-iv", zkey)
K := HKDF-Expand(PRK_k, label, 256 / 8)
NONCE := HKDF-Expand(PRK_n, label, 128 / 8)
IV := NONCE || expiration
return XSalsa20-Poly1305(K, IV, ciphertext)
```

Результатом функции шифрования XSalsa20-Poly1305 является шифротекст, за которым следует 128-битовый тегаутентификации. Поэтому размер зашифрованных данных увеличивается на 16 байтов тега проверки подлинности.

Ключ K и IV счётчика выводятся из метки записи и ключа зоны zkey с использованием HKDF, как задано в [RFC5869]. В фазе извлечения применяется SHA-512 [RFC6234], а в фазе преобразования (expansion) - SHA-256 [RFC6234]. Выходной ключевой материал - это 32 байта (256 битов) для симметричного ключа и 16 байтов (128 битов) для NONCE. Симметричный ключ K - это 256-битовый ключ XSalsa20 [XSalsa20]. Дополнительные данные аутентификации (additional authenticated data или AAD) не используются.

Значение nonce объединяется с 8-байтовым IV, который указывает срок действия блока записей ресурсов в сетевом порядке байтов. Формат передачи получаемого счётчика (IV) приведён на рисунке 12.

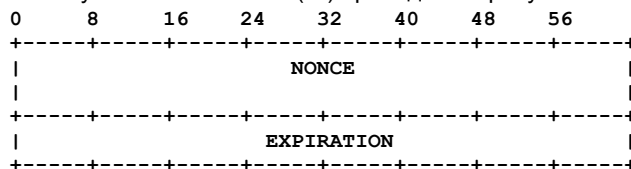


Рисунок 12. Counter Block Initialization Vector.

5.2. Записи перенаправления

Записи перенаправления служат для перенаправления процесса распознавания. Каждой реализации **следует** поддерживать все типы заданных здесь записей перенаправления и **можно** поддерживать любое число дополнительных записей перенаправления из реестра GANA GNS Record Types [GANA]. В записях перенаправления **должен** быть установлен флаг CRITICAL. Отказ от поддержки тех или иных типов записей может приводить к отказам при распознавании. Это может быть оправданным выбором, если некоторые типы записей перенаправления сочтены небезопасными или у приложения есть причины не поддерживать перенаправление в DNS, например, из-за сложности или небезопасности. Записи перенаправления **недопустимо** сохранять или публиковать под меткой вершины.

5.2.1. REDIRECT

Запись REDIRECT является эквивалентом GNS записи CNAME в DNS. Запись REDIRECT **должна** быть единственной не дополнительной (non-supplemental) записью под меткой. **Могут** быть другие неактивные записи того же типа с установленным флагом SHADOW для упрощения плавной смены ключей. Другие записи не разрешены. Детали обработки REDIRECT описаны в параграфе 7.3.1, формат REDIRECT DATA показан на рисунке 13.

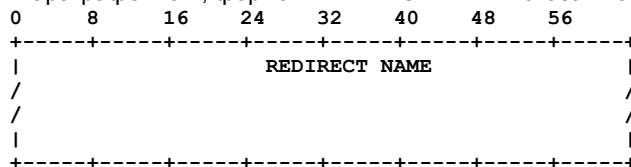


Рисунок 13. Формат передачи REDIRECT DATA.

REDIRECT NAME

Имя для продолжения работы (обычное или относительное). Относительные имена в GNS обозначаются меткой расширения + (U+002B) справа. Строка использует кодировку UTF-8 и завершается null-символом.

5.2.2. GNS2DNS

Запись GNS2DNS передаёт распознавание в DNS. Запись содержит DNS-имя распознавания в DNS, а затем - сервер DNS. Оба имени указываются в формате, заданном в [RFC1034] для имён DNS. **Можно** размещать несколько записей GNS2DNS под одной меткой. Под той же меткой **могут** присутствовать записи DNSSEC DS или иные записи, служащие для защиты соединения с сервером DNS. **Могут** быть другие неактивные записи того же типа (типов) с установленным флагом SHADOW для упрощения плавной смены ключей. Формат GNS2DNS DATA приведён на рисунке 14.

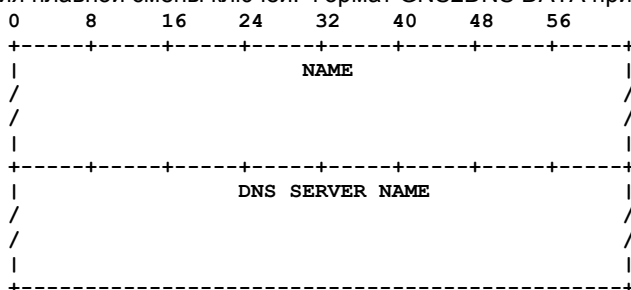


Рисунок 14. Формат передачи GNS2DNS DATA.

NAME

Имя для продолжения распознавания в DNS (кодировка UTF-8 и null-символ в конце).

DNS SERVER NAME

Используемый сервер DNS, указанный адресом IPv4 (через точки), адресом IPv6 (через двоеточия) или именем DNS. Это может быть также относительное имя GNS с символом + в качестве правой метки. Реализация **должна** проверять синтаксис строки для адресов IP в соответствии с подходящей нотацией перед проверкой относительного имени GNS. Если все три проверки дают отрицательный результат строка **должна** считаться именем DNS. Строка указывается в кодировке UTF-8 и завершается null-символом.

Примечание. Если приложение использует имена DNS полученные из записей GNS2DNS в запросе DNS, оно **должно** сначала преобразовать их в представление, совместимое с IDNA [RFC5890].

5.3. Вспомогательные записи

В этом параграфе задан исходный набор вспомогательных типов записей GNS. Каждой реализации **следует** поддерживать обработку указанных здесь типов в соответствии с параграфом 7.3.

5.3.1. LEHO

Запись LEHO (LEgacy HOstname) служит подсказкой для унаследованных имён хостов. Приложения могут применять GNS при поиске адресов IPv4 или IPv6 для служб Internet, однако для подключения к таким службам может потребоваться передавать через транспортный протокол не только IP-адрес и порт, но и каноническое имя DNS для службы. В GNS записи с унаследованными именами предоставляют приложениям имя DNS, требуемое для соединения с такой службой. Наиболее распространённым случаем является работа с виртуальными хостами HTTP и TLS Server Name Indication [RFC6066], где имя DNS должно быть представлено в заголовке Host для HTTP и в согласовании TLS, соответственно. Имя GNS в таких случаях может не работать, поскольку оно может оказаться не уникальным глобально. Кроме того, даже если не уникальность не вызывает проблем, унаследованные службы могут просто не знать о GNS.

Предполагается, что запись LEHO будет встречаться вместе с записями A (IPv4) или AAAA (IPv6). Формат LEHO DATA показан на рисунке 15.

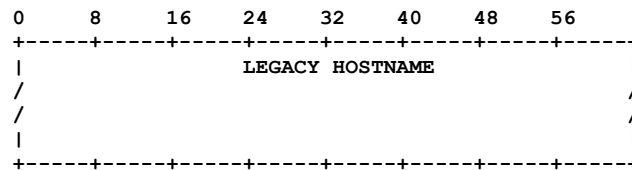


Рисунок 15. Формат передачи LEHO DATA.

LEGACY HOSTNAME

Строка UTF-8 (без null-символа в конце), представляющая унаследованное имя хоста.

Примечание. Если приложение использует значение LEHO в заголовке запроса HTTP (например, Host), оно **должно** преобразовываться в представление, соответствующее IDNA [RFC5890].

5.3.2. NICK

Записи с псевдонимами (nickname) могут применяться администраторами для публикации меток/. Которые зона предпочитает использовать в ссылках. Это является предложением для других зон в части выбора метки при создании записи делегирования (параграф 5.1), содержащей ключ зоны. Эту запись **следует** сохранять лишь локально под меткой вершины @, но её **можно** возвращать в наборе записей под любой меткой как дополнительную запись. В параграфе 7.3.5 описано, как распознаватель должен обрабатывать дополнительные и не дополнительные записи NICK. Форма NICK DATA показан на рисунке 16.

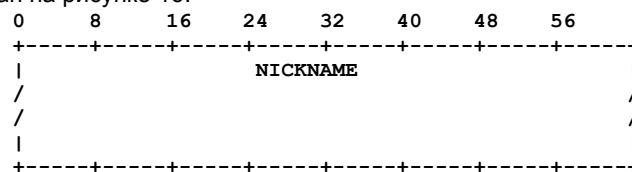


Рисунок 16. Формат передачи NICK DATA.

NICKNAME

Строка UTF-8 (без null-символа в конце), представляющая предпочтительную метку зоны. Строка **должна** быть действительной меткой GNS.

5.3.3. BOX

Предполагается, что поиск GNS будет возвращать все требуемые полезные сведения в одном наборе записей. Это позволяет избежать ненужных дополнительных поисков и криптографически связывает информацию, делая для враждебных хранилищ невозможным представление частичных ответов, в которых могут отсутствовать критически важные для безопасности сведения.

Эта общая стратегия несовместима с особыми метками, используемыми в DNS для записей SRV и TLSA. Поэтому в GNS определён формат записи BOX для упаковки записей SRV и TLSA со включением в набор записей метки, с которой они связаны. Например, запись TLSA для `_https._tcp.example.org` будет сохранена в записи `example.org` как запись BOX с сервисом (SVC) 443 (https), протоколом (PROTO) 6 (tcp) и TYPE TLSA (см. [RFC2782]). Запись BOX DATA показана на рисунке 17.

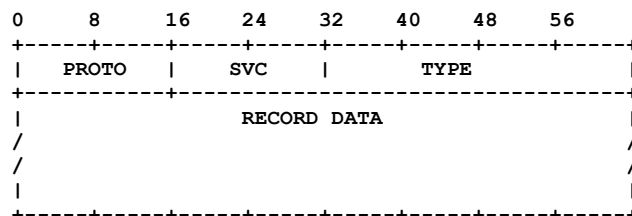


Рисунок 17. Формат передачи BOX DATA.

PROTO

16-битовое значение номера протокола с сетевым порядком байтов. Значения меньше 2^8 зарезервированы для 8-битовых номеров протоколов IP, выделенных IANA [RFC5237] (например, 6 для TCP). Значения больше 2^8 выделяются через реестр GANA GNUnet Overlay Protocols [GANA].

SVC

16-битовое значение сервиса для коробочной записи с сетевым порядком байтов. Для TCP и UDP это номер порта.

TYPE

32-битовый идентификатор типа коробочной записи с сетевым порядком байтов.

RECORD DATA

Поле переменного размер, содержащее формат DATA типа TYPE. Для значений TYPE меньше 2^{16} форма совпадает с двоичным форматом соответствующего типа записи в DNS.

6. Кодирование записей для удалённого хранилища

Любой API, позволяющий хранить блок под 512-битовым ключом и извлекать из ключа 1 или несколько блоков, можно применять в реализации как удалённое хранилище. Чтобы быть полезным и обеспечивать поддержку определённого кодирования для делегирования зон, API **должен** разрешать хранение блоков размером не менее 176 байтов и следует разрешать блоки размером 1024 байта и более. Далее предполагается реализация в хранилище процедур

```
PUT(key, block)
GET(key) -> block
```

Реализация GNS публикует блоки в соответствии со свойствами и рекомендациями базового удалённого хранилища. Это может включать периодическое обновление для сохранения доступности опубликованных блоков.

Механизм явного удаления отдельных блоков в удалённом хранилище не задан, однако блоки включают поле EXPIRATION, которое позволяет реализации удалённого хранилища удалять старые блоки. При наличии нескольких блоков для одного ключа реализации удалённого хранилища **следует** пытаться сохранить и вернуть блок с наибольшим значением EXPIRATION.

Все записи о ресурсах из одной зоны используют общую метку, шифруются и публикуются вместе в одном блоке записи о ресурсах (RRBLOCK) на удалённом хранилище под ключом q, как показано на рисунке 18. Реализации GNS **недопустимо** включать в блоки просроченные записи о ресурсах. Реализация **должна** использовать процедуру хранилища PUT при изменении наборов записи для обновления содержимого зоны. Реализация **должна** гарантировать, что поля EXPIRATION в RRBLOCK монотонно возрастают при каждом изменении, даже когда наименьший оставшийся срок действия записи не увеличивается.



Рисунок 18. Поддержка и публикация локальных зон в распределенном хранилище.

Вывод ключа зоны и создание блока записи описано в следующих параграфах и показано на рисунке 19.

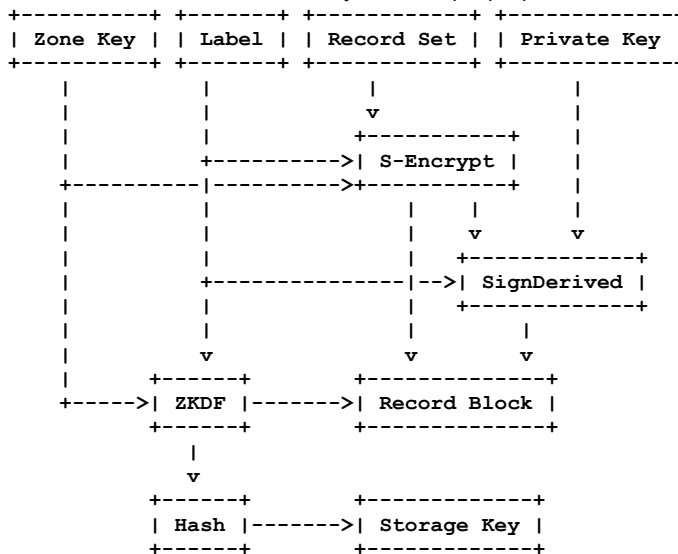


Рисунок 19. Обзор создания Storage Key и Record Block.

6.1. Ключ хранилища

Ключ хранилища выводится из ключа зоны и соответствующей метки записей. Требование знать ключ и метку в сочетании с аналогично выведенными симметричными секретными ключами и скрытыми ключами зон обеспечивает конфиденциальность запросов (см. параграф 3.5 в [RFC8324]). С учётом метки ключ хранилища q выводится как

```
q := SHA-512(ZKDF(zkey, label))
```

label

Строка UTF-8, под которой публикуются записи ресурсов.

zkey

Ключ зоны.

q

512-битовый ключ хранилища, с которым публикуется блок данных ресурса. Это хэш SHA-512 [RFC6234] от выведенного ключа зоны.

6.2. Текстовые данные записей (RDATA)

Записи GNS группируются по меткам, чтобы все записи под одной меткой публиковались вместе, единым блоком хранилища. Такие сгруппированные записи **могут** сопровождаться дополнительными записями.

RDATA служит для кодирования таких групп записей GNS. Двоичный формат RDATA показан на рисунке 20.

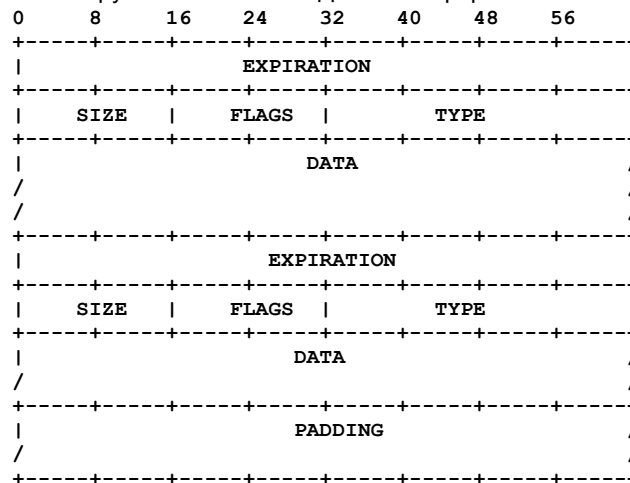


Рисунок 20. Формат передачи RDATA.

EXPIRATION, SIZE, TYPE, FLAGS, DATA

Определения этих полей представлены под рисунком 7 в разделе 5.

PADDING

При сериализации записей в RDATA реализация GNS **должна** обеспечить для размера этого поля значение, равное целой степени 2 с помощью этого поля. Поле **должно** заполняться нулями и игнорироваться при получении. Как исключение, набора, содержащие только записи делегирования зон, никогда не дополняются.

6.3. Блок записей о ресурсах

Записи о ресурсах, сгруппированные в RDATA, шифруются с помощью функции S-Encrypt(), определяемой типом зоны, к которой относятся записи, добавляется префикс метаданных и все вместе помещается в блок записей о ресурсах (RRBLOCK) для удалённого хранилища. Формат передачи GNS RRBLOCK показан на рисунке 21.

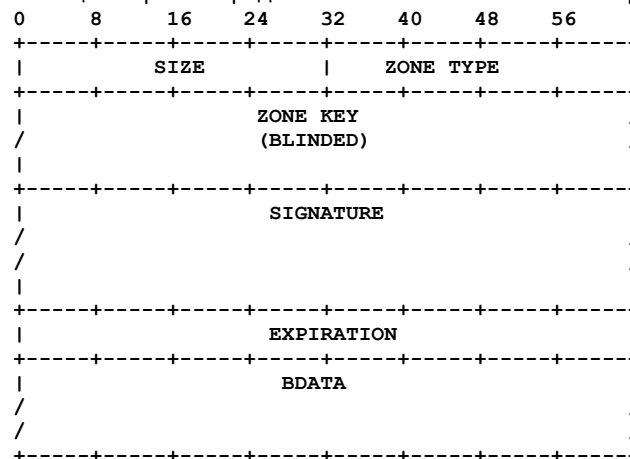


Рисунок 21. Формат передачи RRBLOCK.

SIZE

32-битовое значение размера блока в байтах с сетевым порядком байтов. Несмотря на использование 32-битового значения, реализации **могут** отказываться публиковать блоки, превышающие некий размер, существенно меньше теоретического ограничения в 4 Гбайта.

ZONE TYPE

32-битовое значение ztype с сетевым порядком байтов.

ZONE KEY (BLINDED)

Ослепленный (скрытый) ключ зоны ZKDF(zkey, label) для использования при проверке SIGNATURE. Размер и формат ослепленного ключа зависит от ztype.

SIGNATURE

Подпись, охватывающая поля EXPIRATION и BDATA, как показано на рисунке 22. Размер и формат подписи зависит от ztype. Подпись создаётся с помощью функции SignDerived() в криптосистеме зоны (см. раздел 4).

EXPIRATION

Указывает время завершения срока действия RRBLOCK, когда зашифрованный блок **следует** удалить из хранилища и кэшей, поскольку он, скорее всего, устарел. Однако приложения **могут** продолжать использовать отдельные не устаревшие записи, пока срок их действия не завершится. Для определения срока действия RRBLOCK сначала должен определяться максимальный срок действия среди записей каждого типа в RRBLOCK, включая теневые записи. Затем берётся минимальное из полученных значений. Окончательным временем завершения срока действия будет большее из (1) прежнего значения EXPIRATION в предыдущем RRBLOCK (при наличии) для того

Для имён с суффиксом zTLD стартовая зона явно указывается в суффиксе распознаваемого имени. Для обеспечения однозначности имён с zTLD любая реализация **должна** использовать эту зону в качестве Start Zone. реализация **должна** сначала попытаться интерпретировать самую правую метку данного имени как начало zTLD (параграф 4.1). Если крайнюю справа метку не удастся (частично) декодировать или она не указывает поддерживаемый тип ztype, имя считается обычным и поиск Start Zone **должен** продолжаться с применением локального сопоставления суффиксов с зонами. Если в правой метке присутствует действительный тип ztype, реализация **должна** попытаться синтезировать и декодировать zTLD для извлечения ключа стартовой зоны в соответствии с параграфом 4.1. Если zTLD не удастся синтезировать или декодировать, распознавание имени завершается отказом и приложению возвращается информация об ошибке. В остальных случаях ключ зоны **должен** использоваться как Start Zone

```
www.example.<zTLD>
=> Start Zone: zkey типа ztype
=> Имя для распознавания из Start Zone: www.example
```

Для имён без суффикса zTLD распознаватель **должен** определять Start Zone по локальному сопоставлению суффиксов с зонами. Эти сопоставления **должны** быть настраиваемыми через локальный файл конфигурации или базу данных пользователем или администратором системы. Суффикс **может** состоять из нескольких меток GNS через разделители. Если распознаваемому имени соответствует несколько суффиксов, **должен** выбираться самый длинный из них. Размерам суффиксов из двух результатов **недопустимо** быть равными. Это указывает ошибочную настройку и реализация **должна** возвращать ошибку. Ниже приведён ненормативный пример отображения Start Zone.

```
www.example.xyz.gns.alt
локальные отображения суффиксов:
xyz.gns.alt = zTLD0 := Base32GNS(ztype0||zkey0)
example.xyz.gns.alt = zTLD1 := Base32GNS(ztype1||zkey1)
example.com.gns.alt = zTLD2 := Base32GNS(ztype2||zkey2)
...
=> Start Zone: zkey1
=> Имя для распознавания из Start Zone: www
```

Описанный выше процесс **можно** дополнять другими механизмами, если конкретное приложение требует иного процесса. Если Start Zone не удалось идентифицировать, распознавание **должно** завершаться отказом и приложению **должна** возвращаться ошибка.

7.2. Рекурсия

На каждом этапе рекурсивного распознавания имеется ключ полномочной зоны zkey и имя для распознавания. Имя может быть пустым и в этом случае оно интерпретируется как метка вершины зоны @. Исходно полномочной зоной является стартовая - Start Zone. Далее операции выполняются рекурсивно в показанном ниже порядке.

1. Извлечение из имени правой метки для поиска.
2. Расчёт q с использованием метки и zkey, как указано в параграфе 6.1.
3. Выполнение запроса GET(q) к хранилищу для извлечения RRBLOCK.
4. Проверка того, что (a) срок действия блока не истек, (b) хэш SHA-512 производного ключа полномочной зоны 'zkey' из RRBLOCK совпадает со значением q в запросе и (c) подпись действительна. Если любое из этих условий не выполняется, RRBLOCK **должен** игнорироваться и поиск (если это применимо) GET(q) в хранилище **должен** продолжаться для извлечения других RRBLOCK.
5. Получение RDATA путём расшифровки BDATA из RRBLOCK с использованием функции S-Decrypt(), определяемой типом зоны (фактически обращение процесса, описанного в параграфе 6.3).

После расшифровки корректно сформированного блока выполняется обработка записей из RDATA.

7.3. Обработка записей

Обработка выполняется только для действительных записей. Для фильтрации непригодных записей распознаватель **должен** проверить в записях хотя бы поля срока действия и FLAGS. Просроченные записи распознаватель **должен** отбрасывать, а записи с флагами SHADOW и SUPPLEMENTAL могут исключаться из рассмотрения. Если распознаватель встречает запись с флагом CRITICAL и неподдерживаемым типом, распознавание **должно** прерываться с возвратом ошибки. В описание ошибки **следует** включать сведения, указывающие, что критическая запись не может быть обработана. Реализация **может** не указывать причину отказа, но это осложнит пользователям поиск неполадок. Дальнейшие действия зависят от контекста, в котором происходит распознавание имени.

Вариант 1

Если после фильтрации в наборе осталась лишь 1 запись REDIRECT, остаток имени помещается перед REDIRECT DATA и выполняется распознавание имени для результата (см. параграф 7.3.1).

Вариант 2

Если после фильтрации в наборе остались только записи GNS2DNS, распознавание продолжается в DNS (см. параграф 7.3.2).

Вариант 3

Если оставшаяся для распознавания часть имени имеет формат _SERVICE._PROTO и набор записей содержит одну или несколько соответствующих записей BOX, записи из BOX являются конечным результатом и рекурсия завершается, как указано в параграфе 7.3.3.

Вариант 4

Если текущий набор содержит лишь одну запись делегирования, распознавание оставшейся части имени передаётся в целевую зону, как указано в параграфе 7.3.4.

Вариант 5

Если оставшаяся часть имени пуста, набор записей является финальным результатом. Если в нем имеются записи NICK, сначала они **должны** быть обработаны в соответствии с параграфом 7.3.5. Затем набор записей возвращается как окончательный результат.

Если ни один из указанных вариантов не применим, распознавание завершается отказом и распознаватель **должен** возвращать пустой набор записей.

7.3.1. REDIRECT

Если оставшееся имя не пусто и желаемым типом записи является REDIRECT, распознавание завершается записью REDIRECT. Если правой частью REDIRECT NAME является метка расширения (U+002B, +), распознавание продолжается в GNS с новым именем в текущей зоне. В иных случаях результирующее имя распознаётся через принятый по умолчанию в операционной системе процесс распознавания имён. Это может вызывать процесс распознавания имён GNS при соответствующей настройке системы. Если распознавание продолжается в DNS, имя сначала **должно** приводиться в совместимое с IDNA представление [RFC5890].

Для предотвращения бесконечных циклов (петель) распознаватель **должен** реализовать обнаружение петель в рекурсивном распознавании. Такое обнаружение должно работать даже в случае, когда REDIRECT из GNS вызывает последующий поиск GNS через стандартный процесс распознавания в операционной системе.

7.3.2. GNS2DNS

Распознаватель возвращает записи GNS2DNS при выполнении трёх указанных ниже условий.

1. Распознаватель встречает одну или несколько записей GNS2DNS.
2. Оставшаяся часть имени пуста.
3. Желаемым типом записи является GNS2DNS.

В иных случаях предполагается, что распознаватель сначала узнает IP-адреса указанных серверов имён DNS. Имя DNS **должно** преобразовываться в совместимое с IDNA представление [RFC5890] для распознавания в DNS. Записи GNS2DNS **могут** включать численные адреса IPv4 или IPv6, позволяя распознавателю пропустить этот шаг. Имена серверов DNS сами могут быть именами GNS или DNS. Если правой меткой в имени сервера DNS является метка расширения (U+002B, +), остальная часть имени интерпретируется относительно зоны записи GNS2DNS. Если имя сервера DNS завершается меткой, представляющей ключ зоны, имя сервера DNS распознаётся по ключу зоны GNS.

Под одной меткой может храниться несколько записей GNS2DNS и в этом случае распознаватель **должен** попытаться использовать каждую из них (он **может** делать это в любом порядке или в параллель). При наличии нескольких записей GNS2DNS имена DNS для них **должны** быть идентичны, иначе будет непонятно, к какому распознавателю имён следует обращаться. Если указаны разные имена DNS распознавание **следует** завершать отказом и возвращать приложению соответствующую ошибку.

Если под меткой имеются записи DNSSEC DS или иные записи, служащие для защиты соединений с серверами DNS, распознавателю DNS **следует** использовать их для защиты соединения с сервером DNS.

После определения IP-адресов серверов DNS имя DNS из записи GNS2DNS добавляется в конец оставшейся части имени для распознавания и для результата выполняется распознавание путём запросов к серверам имён DNS. Синтезированное имя **должно** быть преобразовано в совместимое с IDNA представление [RFC5890] для распознавания в DNS. Если такое преобразование невозможно, распознавание должно прерываться с возвратом ошибки. В описание ошибки **следует** включать сведения, указывающие, что критическая запись не может быть обработана. Реализация **может** не указывать причину отказа, но это осложнит пользователям поиск неполадок.

Поскольку указанные серверы DNS могут оказаться полномочными, распознаватель GNS **должен** поддерживать рекурсивное распознавание DNS и эту функцию **недопустимо** делегировать полномочным серверам DNS. Приложению возвращается первый успешный результат рекурсивного распознавания. Кроме того, распознавателю **следует** возвращать запрошенное имя DNS как дополнительную запись LEHO (см. параграф 5.3.1) с относительным сроком действия в 1 час.

После перехода из GNS в DNS с помощью записи GNS2DNS «возврата» уже не будет. Распознавание (возможно, рекурсивное) имени DNS **недопустимо** делегировать обратно в GNS и следует применять только спецификации DNS. Например, имена из записей DNS CNAME распознавателям, поддерживающим DNS и GNS, **недопустимо** считать именами GNS.

Распознавателям GNS **следует** поддерживать опцию для запрета обработки DNS, чтобы избежать утечки информации и обеспечить согласованный профиль защиты для всех случаев распознавания. Такие распознаватели будут возвращать пустой набор записей при обнаружении в процессе распознавания меток GNS2DNS. Однако при наличии GNS2DNS в наборе записей под меткой вершины и явном запросе записи GNS2DNS приложением, такие записи **должны** будут возвращаться даже при отключённой поддержке DNS в конфигурации распознавателя GNS.

7.3.3. BOX

При получении записи BOX распознаватель GNS должен распаковать её, если распознаваемое имя продолжается `_SERVICE._PROTO`. В ином случае запись BOX остаётся нетронутой. Таким образом, записи TLSA (и SRV) не требуют отдельного сетевого запроса и записи TLSA становятся неотделимыми от соответствующих адресных записей.

7.3.4. Записи делегирования зон

Когда распознаватель встречает запись делегирования поддерживаемого типа (например, PKEY или EDKEY) и оставшаяся часть имени не пуста, распознавание продолжается рекурсивно для оставшейся части имени в зоне GNS, указанной записью делегирования.

Всякий раз, когда распознаватель встречает новую зону GNS, он **должен** проверить по локальному списку отзыва (см. параграф 4.2), не был ли отозван соответствующий ключ зоны. Если ключ отозван, распознавание **должно** возвращать пустой результат (отказ).

Реализациям **недопустимо** разрешать несколько делегирований под одной меткой (за исключением теневых записей). Реализация **может** поддерживать любое подмножество `ztype`. Реализациям **недопустимо** обрабатывать записи делегирования зоны, сохранённые под меткой вершины (`@`). Если такая запись встречается, распознавание прерывается и **должна** возвращаться ошибка. Реализация **может** не указывать причину отказа, но это осложнит пользователям поиск неполадок.

Если оставшаяся часть имени пуста и был получен набор, включающий лишь запись делегирования, рекурсия продолжается со значением этой записи в качестве полномочной зоны и меткой вершины @ как оставшимся именем. Исключением является случай, когда указанный приложением желаемый тип записи совпадает с `ztype`, и в этом случае возвращается запись делегирования.

7.3.5. NICK

Записи NICK имеют значение для рекурсивного распознавателя лишь в том случае, когда рассматриваемый набор записей является конечным результатом, возвращаемым приложению. Встречающиеся записи NICK могут быть дополнительными (см. параграф 5) и не дополнительными. Если запись NICK является дополнительной, распознаватель возвращает набор записей лишь при соответствии одной из не дополнительных записей запрошенному типу записи. Один набор записей может включать дополнительные и не дополнительные записи NICK.

Различие между дополнительными и не дополнительными записями NICK позволяет приложению сопоставить запись с полномочной зоной, например,

```
Query: alice.example.gns.alt (type=A)
Result:
A: 192.0.2.1
NICK: eve (non-supplemental)
```

Здесь возвращённая запись NICK является не дополнительной. Для приложения это означает, что NICK относится к зоне `alice.example.gns.alt` и публикуется под меткой вершины вместе с записью A. Запись NICK интерпретируется как «зона, заданная `alice.example.gns.alt`, хочет, чтобы её указывали как `eve`». Рассмотрим другой пример

```
Query: alice.example.gns.alt (type=AAAA)
Result:
AAAA: 2001:db8::1
NICK: john (supplemental)
```

Здесь запись NICK помечена как дополнительная. Это означает, что NICK относится к зоне `example.gns.alt` и публикуется под меткой `alice` вместе с записью AAAA. Запись NICK интерпретируется как «зона, заданная `example.gns.alt`, хочет, чтобы её указывали как `john`». Это различие полезно и для других записей, публикуемых как дополнительные.

8. Другие языки и кодировка символов

Все имена в GNS используют кодировку UTF-8 [RFC3629]. Метки должны канонизироваться с использованием нормализованной формы C (Normalization Form C или NFC) [Unicode-UAX15]. Это не относится к именам DNS в записях DNS, таких как CNAME, которые могут использовать другие языки в соответствии со спецификацией IDNA [RFC5890].

9. Вопросы приватности и безопасности

9.1. Доступность

Для обеспечения доступности записей по завершении абсолютного срока действия реализация **может** разрешать локальное задание относительного срока действия записей. Затем записи могут периодически публиковаться реализацией с обновлённым абсолютным сроком действия.

Реализация **может** разрешать пользователям управлять в своих зонах приватными записями, которые не публикуются в хранилище. Приватные записи обрабатываются при распознавании меток в локальных зонах как обычные записи, но их данные недоступны для нелокальных пользователей.

9.2. Стойкость

Защищённость криптографических систем зависит от строгости как выбранных криптоалгоритмов, так и применяемых этими алгоритмами ключей, а также от устройства применяемого системой протокола в части предотвращения некриптографических методов обхода защиты всей системы. Именно поэтому разработчикам приложений, управляющих зонами GNS, **следует** выбирать для применения по умолчанию тип `ztype`, считающийся безопасным на момент выпуска программы. Для приложений, предназначенных конечным пользователям, от которых не ожидается понимания криптографии, разработчикам **недопустимо** предоставлять пользователю выбор `ztype` для новых зон.

Этот документ связан с выбором криптоалгоритмов для использования в GNS. Для указанных в документе алгоритмов не известно о случаях взлома (в криптографическом смысле) и криптографические исследования позволяют предполагать, что алгоритмы останутся безопасными в обозримом будущем. Однако это не навсегда и предполагается, что время от времени будут выпускаться новые версии этого документа, соответствующие современной практике.

В части криптостойкости при возникновении необходимости в новой криптосхеме (например, ECDSA на основе Ed25519 для записей PKEY) её можно просто ввести через новый тип записей. Администраторы зон могут поменять тип для будущих записей делегирования. Прежний тип записи остаётся и зоны могут постепенно (итерациями) переходить на обновлённые ключи. Чтобы реализации корректно выдавали сообщения об ошибках при обнаружении неподдерживаемого `ztype`, имеющиеся и будущие записи делегирования должны иметь флаг CRITICAL.

9.3. Криптография

Приведённые ниже соображения служат основой для выбора `ztype`, заданных в этом документе. Эти же соображения применимы при задании новых `ztype` в соответствии с разделом 4.

Ключи зон GNS PKEY используют ECDSA на основе Ed25519. Это нетрадиционный выбор, поскольку ECDSA обычно применяется с другими кривыми. Однако стандартизованные кривые ECDSA проблематичны по ряду причин, как указано в статьях `Curve25519` и `EdDSA` [RFC7748] [ed25519]. Применение EdDSA напрямую также невозможно, поскольку для секретного ключа применяется хэш-функция, нарушающая линейность, от которой зависит сокрытие ключа в GNS. Неизвестно, чтобы кто-то предполагал, что применение Ed25519 вместо другой распространённой

кривой аналогичного размера снижало бы безопасность ECDSA. В GNS применяются 256-битовые кривые, поэтому зашифрованные (открытые) ключи помещаются в одну метку DNS, что удобно для применения.

Для обеспечения неотличимости (indistinguishability) шифротекста нужно внимательно относиться к выбору IV в блоке счётчика. В описываемом решении IV всегда включает время завершения срока действия блока записей. Когда приложения хранят записи с относительным сроком действия, монотонность обеспечивается неявно, поскольку при каждой публикации блока в хранилище его значение IV уникально, так как время завершения срока действия рассчитывается динамически и монотонно возрастает с течением системного времени. Тем не менее, реализация **должна** гарантировать, что при уменьшении относительного срока действия время завершения срока действия следующего блока записей **должно** быть позже последнего опубликованного блока. Для записей с абсолютным сроком действия реализация **должна** гарантировать, что время завершения срока действия всегда увеличивается при изменении данных в записи. Например, время завершения срока действия может увеличиться при передаче на 1 мксек, даже если пользователь не запрашивал изменений. В случае удаления всех записей о ресурсах под меткой реализация **должна** отследить последнее абсолютное время завершения срока действия последнего опубликованного блока ресурсов. Реализация **может** задать и использовать специальный тип записи в качестве «надгробия» сохраняющего последнее абсолютное время завершения срока действия, но затем **должна** принять меры для предотвращения публикации блока с таким надгробием. При добавлении новых записей под этой меткой позднее реализация **должна** гарантировать, что срок их действия завершается после опубликованного последним блоком. Для обеспечения монотонного роста времени завершения срока действия реализация **должна** локально хранить запись последнего времени по системным часам, чтобы создать монотонные часы при переводе системных часов назад.

9.4. Смягчение злоупотреблений

Имена GNS - это строки UTF-8, поэтому GNS сталкивается с проблемами подделки имён, похожими на проблемы подделки в DNS, связанные с доменными именами на местных языках. В DNS злоумышленники могут регистрировать схожие визуально или на слух имена для организации фишинговых атак. Администраторы зон GNS должны учитывать это и принимать правила для смягчения таких атак.

DNS может применяться для борьбы с нелегальным содержимым в Internet путём изъятия соответствующих доменов. Однако такие же механизмы могут применяться для государственной цензуры. Предотвращение таких возможностей стало одной из причин разработки GNS, где домены TLD не являются счётными. Стартовая зона (Start Zone) распознавателя задаётся локально, поэтому изъятие доменов сложно и неэффективно в GNS.

9.5. Поддержка зон

В GNS администраторам зон нужно поддерживать и защищать ключи своих зон. Потерянный секретный ключ зоны восстановит невозможно и сообщение для отзыва зоны уже не рассчитать. Сообщения об отзыве можно рассчитать заранее на случай потребности в отзыве при потере секретного ключа. Администраторы зон (а в GNS и конечные пользователи) должны тщательно и ответственно беречь свои криптографические ключи. GNS позволяет подписывать записи заранее (offline) для поддержки процессов защиты (воздушного зазора) своих секретных ключей.

Пользователи должны управлять конфигурацией своей локальной стартовой зоны. Для обеспечения целостности и доступности имён, пользователь должен гарантировать, что его локальные сведения Start Zone не были скомпрометированными или устаревшими. Можно ожидать, что обработка отзывов зон и начальная Start Zone будут обеспечиваться реализацией GNS (прямая доставка). Поставка исходной конфигурации Start Zone фактически организует корневую зону, а расширение и настройка зоны полностью отдаются пользователю.

Хотя реализации, соблюдающие эту спецификацию, будут совместимы, при подключении двух реализаций к разным удалённым хранилищам они будут недоступны друг другу. Это может привести к состоянию, когда в глобальном пространстве имён имеется запись для определённого имени, но реализация не взаимодействует с удалённым хранилищем, где размещён соответствующий блок, и поэтому не способна распознать имя. Это похоже на ситуацию с расщеплением горизонта (split-horizon) в DNS. Используемый объект удалённого хранилища скорее всего будет зависеть от контекста конкретного приложения, применяющего распознавание GNS. Например, одним из приложений является распознавание скрытых служб в сети Tor [TorRendSpec], что предполагает использование маршрутизаторов Tor в качестве удалённых хранилищ. Реализации «агрегированных» сущностей удалённого хранения возможны, но предполагаются исключительными случаями, нежели нормой.

9.6. DHT как удалённое хранилище

Этот документ не задаёт свойств базового удалённого хранилища, требуемых для любой реализации GNS. Важно отметить, что такие свойства реализация GNS наследует напрямую. Это включает свойства защиты и другие не функциональные свойства, такие как расширяемость и производительность. Разработчикам следует очень внимательно относиться к выбору DHT для использования в качестве удалённого хранилища в реализации GNS. Имеются DHT с разумными характеристиками безопасности и производительности [R5N]. Следует также учитывать, что реализации GNS, основанные на разных наложенных DHT, вряд ли будут доступны друг другу.

9.7. Отзывы

Администраторам зон рекомендуется заранее создавать и надёжно хранить отзывы зон на случай потери, компрометации или замены ключа зоны. Предварительно рассчитанный отзыв может утратить силу в результате завершения срока действия или протокольных изменениях, таких как корректировка эпохи. Поэтому разработчики и пользователи должны принимать меры предосторожности для подобающей поддержки отзывов. Данные (payload) отзыва не включают «новый» ключ для замены, поскольку с этим связано два основных недостатка.

1. Если отзыв публикуется после компрометации секретного ключа, разрешение на замену будет опасным - злоумышленник, владеющий секретным ключом может широковещательно разослать отзыв с заменой ключа. При замене у скомпрометированного владельца не останется шансов ввести отзыв. Таким образом, разрешение на замену секретного ключа ухудшает ситуацию в случае компрометации ключа.
2. Иногда отзыв ключей служит для смены криптосистемы. Переход на иную криптосистему путём смены ключей через сообщение об отзыве будет безопасен до тех пор, пока обе криптосистемы защищены от подделки.

Плановый (не экстренный) переход на другую криптосистему следует выполнять путём параллельного запуска обеих систем и завершается отзывом старого ключа, когда он больше не считается безопасным и (будем надеяться) большинство пользователей уже перешло на применение нового ключа.

9.8. Приватность зоны

GNS не поддерживает аутентифицированное отрицание существования имён в зоне. Данные записей публикуются в шифрованной форме с использованием ключей, выведенных из ключа зоны и метки записи. Администраторам зон следует внимательно рассмотреть (1) являются ли метка и ключ зоны открытыми (public) или (2) один или оба из них следует использовать как общий секрет для ограничения доступа в соответствующим данным записи. В отличие от открытых ключей зоны, метки с малой энтропией могут быть легко угаданы злоумышленником. Если злоумышленник знает открытый ключ зоны, использование общеизвестных или предсказуемых меток позволяет раскрыть соответствующие записи.

Следует отметить, что атаки с угадыванием меток применимы лишь в случаях, когда ключ зоны каким-то способом раскрыт атакующему. GNS не раскрывает ключ при поиске и публикации записей о ресурсах (в сети применяются только скрытые ключи зон). Однако ключи зон раскрываются в процессе отзыва. Поэтому **рекомендуется** применять метки с достаточной энтропией для предотвращения атак с угадыванием, если какие-либо данные в наборе записей считаются конфиденциальными.

9.9. Управление зонами

Хотя система DNS является распределенной, на практике она полагается для обеспечения глобальной уникальности имён на централизованных, доверенных регистраторов. По мере осознания центральной роли DNS в Internet различные организации используют свои возможности (в том числе юридические) для атак на DNS, создавая угрозы глобальной доступности и целостности информации в Internet. Широкое обсуждение этого вопроса выходит за рамки документа, а об исследованиях и анализе можно прочитать в свежих научных работах, включая [SecureNS].

GNS предназначена обеспечить защищённую и повышающую уровень приватности альтернативу протоколу распознавания имён DNS, особенно при наличии цензуры или манипулирования информацией. В частности, решаются проблемы DNS, связанные с приватностью запросов. Однако в зависимости от управления корневой зоной любое развёртывание столкнётся с проблемой единой иерархии с централизованно управляемым корнем и связанной с этим проблемой распространения и поддержки корневых серверов DNS, рассмотренными в параграфах 3.12 и 3.10 [RFC8324]. В DNS эти проблемы напрямую связаны с централизованным управлением корневой зоной Корпорацией по назначению имён и значений в Internet (Internet Corporation for Assigned Names and Numbers или ICANN), которое позволяет обеспечивать глобальную уникальность имён.

В GNS стартовые зоны (Start Zone) предоставляют пользователям локальные полномочия управления их предпочтительной корневой зоной. Это даёт пользователям возможность заменить или улучшить конфигурацию доверенной корневой зоны, предоставляемую сторонней организацией (например, исполнителем или органом коллективного управления, подобным ICANN), защищённой передачей полномочий с использованием локальных имён (retname), работая в условиях наличия очень сильных противников. В сочетании с zTLD это обеспечивает пользователям GNS глобальное, защищённое и запоминающееся сопоставление имён без доверенного органа.

Любая реализация GNS **может** предоставлять принятую по умолчанию модель управления в форме начального сопоставления Start Zone.

9.10. Неоднозначность пространства имён

Технически можно применять протокол GNS для распознавания имён в глобальном пространстве DNS, однако для этого требуется стандартизация применения GNS для этого частного случая соответствующими органами управления и заинтересованными сторонами (например, IETF и ICANN). Эта возможность предполагает, что имена GNS могут быть не отличимыми от имён DNS в соответствующем общем формате отображения [RFC8499] или специальных форматах доменных имён [RFC6761], если конфигурация Start Zone сопоставляет глобальные суффиксы DNS с зонами GNS. Для приложений выбор системы для распознавания данного имени будет неоднозначным. Это создаёт риск при попытке распознавания через DNS имени GNS, как отмечено в [RFC8244]. В этом случае имя GNS может быть раскрыто в процессе распознавания DNS. Для предотвращения раскрытия запрашиваемых имён GNS **рекомендуется** понимающим GNS приложениям пытаться распознать имя в GNS до применения других методов (с учётом возможных сопоставлений суффиксов с зонами и zTLD). Предполагается, что сопоставления суффиксов с зонами настраиваются локальным администратором, поэтому распознавание в GNS будет соответствовать ожиданиям пользователя, даже если имя можно распознать в DNS. Если для имени нет сопоставления суффиксов с зонами и не найдено zTLD, распознавание **можно** продолжить с другими методами, такими как DNS. Если сопоставление для имени имеется или имя заканчивается zTLD, оно **должно** распознаваться с применением GNS и распознавание **недопустимо** продолжать с другими методами, независимо от результата распознавания GNS.

Примером реализации и применения такого процесса распознавания могут служить такие механизмы, как переключение систем имён (Name Service Switch или NSS) в UNIX-подобных операционных системах. NSS позволяет администратору системы настроить предпочтения для распознавания имён и интегрируется с реализацией системного распознавателя.

Для случаев, когда имена GNS можно спутать с именами других механизмов распознавания (в частности, DNS), **следует** применять домен .gns.alt. В случаях использования ловушек (sinkhole) для блокировки вредоносных сайтов или обслуживания доменов DNS через GNS для обхода цензуры GNS **можно** намеренно применять таким образом, чтобы мешать распознаванию другими системами.

10. Взаимодействие с GANA

10.1. Реестр GUNet Signature Purposes

Агентство GANA [GANA] поддерживает реестр назначений подписей GUNet Signature Purposes (таблица 1).

Таблица 1. Реестр GANA GUNet Signature Purposes.

Назначение	Имя	Документ	Комментарий
3	GNS_REVOCATION	RFC 9498	Отзыв ключа зоны GNS
15	GNS_RECORD_SIGN	RFC 9498	Подпись набора записей GNS

10.2. Реестр GNS Record Types

GANA [GANA] поддерживает реестр GNS Record Types, каждая запись которого включает указанные ниже поля.

Name
Имя типа записи (строка букв и цифр ASCII без учёта регистра). Для записей делегирования выделенное значение представляет зтуру зоны.

Number
32-битовое целое число больше 65535.

Comment
Необязательное краткое описание назначения типа на английском языке (кодировка UTF-8).

Contact
Необязательные контактные сведения для получения дополнительной информации.

References
Необязательные ссылки на документ, описывающий тип записи (например, RFC).

Регистрация выполняется по мере подачи запросов (First Come First Served), похожей на одноимённую процедуру из [RFC8126] и описывающей действия, предпринимаемые GANA.

- Добавление записей возможно после рецензирования любым полномочным участником GANA с регистрацией выделения уникальных имён в порядке поступления запросов. Рецензенты отвечают за соответствие выбранного значения Name типу записи. Реестр задаёт уникальный номер для записи.
- Уполномоченные участники GANA для рецензирования заявок доступны по адресу <gns-registry@gnunet.org>.
- Запрос **должен** содержать уникальное имя и контактные данные. Данные для контактов **могут** быть включены в реестр с согласия запрашивающего. В запрос **можно** добавлять ссылки на документы и краткое описание, как указано выше.

Агентство GANA выделило номера для типов записей, определённых в этом документе, в реестре GNS Record Types (таблица 2).

Таблица 2. Реестр GANA GNS Record Types.

Значение	Имя	Контактные данные	Документ	Комментарии
65536	PKEY	gns-registry@gnunet.org	RFC 9498	Делегирование зоны GNS (PKEY)
65537	NICK	gns-registry@gnunet.org	RFC 9498	Псевдоним зоны GNS
65538	LEHO	gns-registry@gnunet.org	RFC 9498	Унаследованное имя хоста GNS
65540	GNS2DNS	gns-registry@gnunet.org	RFC 9498	Делегирование в DNS
65541	BOX	gns-registry@gnunet.org	RFC 9498	Коробочные записи
65551	REDIRECT	gns-registry@gnunet.org	RFC 9498	Запись перенаправления
65556	EDKEY	gns-registry@gnunet.org	RFC 9498	Делегирование зоны GNS (EDKEY)

10.3. Реестр субдоменов .alt

GANA [GANA] поддерживает реестр .alt Subdomains, который могут (но не обязаны) принимать во внимание конкретные исполнители. Реестр не утверждён IETF или ICANN и никак не связан с ними. Содержимое реестра указано ниже.

Label
Метка субдомена в формате DNS «буквы, цифры, дефис» (letters, digits, hyphen или LDH) заданном в параграфе 2.3.1 [RFC5890]).

Description
Необязательное краткое описание назначения типа на английском языке (кодировка UTF-8).

Contact
Необязательные контактные сведения для получения дополнительной информации.

References
Необязательные ссылки на документ, описывающий тип записи (например, RFC).

Регистрация выполняется по мере подачи запросов (First Come First Served), похожей на одноимённую процедуру из [RFC8126] и описывающей действия, предпринимаемые GANA.

- Добавление записей возможно после рецензирования любым полномочным участником GANA с регистрацией выделения уникальных субдоменов в порядке поступления запросов. Рецензенты отвечают за соответствие выбранного значения Subdomain назначению субдомена.
- Уполномоченные участники GANA для рецензирования заявок доступны по адресу <alt-registry@gnunet.org>.
- Запрос **должен** содержать уникальное имя и контактные данные. Данные для контактов **могут** быть включены в реестр с согласия запрашивающего. В запрос **можно** добавлять ссылки на документы и краткое описание, как указано выше.

Агентство GANA выделило заданный этим документом субдомен в реестре .alt Subdomains (таблица 3).

Таблица 3. Реестр GANA .alt Subdomains Registry.

Метка	Контактные данные	Документ	Описание
gns	alt-registry@gnunet.org	RFC 9498	Субдомен .alt для GNS

11. Взаимодействие с IANA

Этот документ не действует действий IANA.

12. Статус реализации и внедрения

Имеются две реализации, соответствующие данной спецификации, одна из которых написана на языке C, другая - на Go. Реализация на C, являющаяся частью проекта GNUnet [GNUnetGNS], является оригинальной и эталонной. Реализация на Go [GoGNS] демонстрирует совместимость двух реализаций GNS, работающих на основе одного базового хранилища DHT. В настоящее время в одноранговой (peer-to-peer) сети GNUnet [GNUnet] активно внедряется GNS на основе DHT [R5N]. Реализация Go [GoGNS] использует это для построения на основе GNUnet DHT служб, доступных любому партнёру GNUnet. Это показывает, как реализации GNS могут подключаться к имеющемуся развёртыванию и участвовать в распознавании имён и публикации зон.

Суверенная (self-sovereign) система идентификации ge:claimID [reclaim] применяется в GNS для выборочного обмена атрибутами отождествлений и аттестатами с третьими сторонами.

Инструмент Ascension [Ascension] облегчает перевод зон DNS в зоны GNS путём трансляции информации, полученной из зоны DNS, в зону GNS.

13. Литература

13.1. Нормативные документы

- [RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, [RFC 1034](#), DOI 10.17487/RFC1034, November 1987, <<https://www.rfc-editor.org/info/rfc1034>>.
- [RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, [RFC 1035](#), DOI 10.17487/RFC1035, November 1987, <<https://www.rfc-editor.org/info/rfc1035>>.
- [RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <<https://www.rfc-editor.org/info/rfc2782>>.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, [RFC 2119](#), DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, [RFC 3629](#), DOI 10.17487/RFC3629, November 2003, <<https://www.rfc-editor.org/info/rfc3629>>.
- [RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, DOI 10.17487/RFC3686, January 2004, <<https://www.rfc-editor.org/info/rfc3686>>.
- [RFC3826] Blumenthal, U., Maino, F., and K. McCloghrie, "The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model", RFC 3826, DOI 10.17487/RFC3826, June 2004, <<https://www.rfc-editor.org/info/rfc3826>>.
- [RFC5237] Arkko, J. and S. Bradner, "IANA Allocation Guidelines for the Protocol Field", BCP 37, RFC 5237, DOI 10.17487/RFC5237, February 2008, <<https://www.rfc-editor.org/info/rfc5237>>.
- [RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <<https://www.rfc-editor.org/info/rfc5869>>.
- [RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <<https://www.rfc-editor.org/info/rfc5890>>.
- [RFC5895] Resnick, P. and P. Hoffman, "Mapping Characters for Internationalized Domain Names in Applications (IDNA) 2008", RFC 5895, DOI 10.17487/RFC5895, September 2010, <<https://www.rfc-editor.org/info/rfc5895>>.
- [RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <<https://www.rfc-editor.org/info/rfc6234>>.
- [RFC6895] Eastlake 3rd, D., "Domain Name System (DNS) IANA Considerations", BCP 42, RFC 6895, DOI 10.17487/RFC6895, April 2013, <<https://www.rfc-editor.org/info/rfc6895>>.
- [RFC6979] Pornin, T., "Deterministic Usage of the Digital Signature Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 6979, DOI 10.17487/RFC6979, August 2013, <<https://www.rfc-editor.org/info/rfc6979>>.
- [RFC7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves for Security", RFC 7748, DOI 10.17487/RFC7748, January 2016, <<https://www.rfc-editor.org/info/rfc7748>>.
- [RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <<https://www.rfc-editor.org/info/rfc8032>>.
- [RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, [RFC 8126](#), DOI 10.17487/RFC8126, June 2017, <<https://www.rfc-editor.org/info/rfc8126>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, [RFC 8174](#), DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.
- [RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, [RFC 8499](#), DOI 10.17487/RFC8499, January 2019, <<https://www.rfc-editor.org/info/rfc8499>>.
- [RFC9106] Biryukov, A., Dinu, D., Khovratovich, D., and S. Josefsson, "Argon2 Memory-Hard Function for Password Hashing and Proof-of-Work Applications", RFC 9106, DOI 10.17487/RFC9106, September 2021, <<https://www.rfc-editor.org/info/rfc9106>>.
- [GANA] GNUnet e.V., "GNUnet Assigned Numbers Authority (GANA)", 2023, <<https://gana.gnunet.org/>>.

- [MODES] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: Methods and Techniques", NIST Special Publication 800-38A, DOI 10.6028/NIST.SP.800-38A, December 2001, <<https://doi.org/10.6028/NIST.SP.800-38A>>.
- [CrockfordB32] Crockford, D., "Base 32", March 2019, <<https://www.crockford.com/base32.html>>.
- [XSalsa20] Bernstein, D. J., "Extending the Salsa20 nonce", 2011, <<https://cr.yp.to/papers.html#xsalsa>>.
- [Unicode-UAX15] Davis, M., Whistler, K., and M. Dürst, "Unicode Standard Annex #15: Unicode Normalization Forms", Revision 31, The Unicode Consortium, Mountain View, September 2009, <<https://www.unicode.org/reports/tr15/tr15-31.html>>.
- [Unicode-UTS46] Davis, M. and M. Suignard, "Unicode Technical Standard #46: Unicode IDNA Compatibility Processing", Revision 31, The Unicode Consortium, Mountain View, September 2023, <<https://www.unicode.org/reports/tr46>>.

13.2. Дополнительная литература

- [RFC1928] Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and L. Jones, "SOCKS Protocol Version 5", [RFC 1928](#), DOI 10.17487/RFC1928, March 1996, <<https://www.rfc-editor.org/info/rfc1928>>.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", [RFC 4033](#), DOI 10.17487/RFC4033, March 2005, <<https://www.rfc-editor.org/info/rfc4033>>.
- [RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <<https://www.rfc-editor.org/info/rfc6066>>.
- [RFC7363] Maenpaa, J. and G. Camarillo, "Self-Tuning Distributed Hash Table (DHT) for REsource LOcation And Discovery (RELOAD)", RFC 7363, DOI 10.17487/RFC7363, September 2014, <<https://www.rfc-editor.org/info/rfc7363>>.
- [RFC8324] Klensin, J., "DNS Privacy, Authorization, Special Uses, Encoding, Characters, Matching, and Root Structure: Time for Another Look?", RFC 8324, DOI 10.17487/RFC8324, February 2018, <<https://www.rfc-editor.org/info/rfc8324>>.
- [RFC8806] Kumari, W. and P. Hoffman, "Running a Root Server Local to a Resolver", RFC 8806, DOI 10.17487/RFC8806, June 2020, <<https://www.rfc-editor.org/info/rfc8806>>.
- [RFC6761] Cheshire, S. and M. Krochmal, "Special-Use Domain Names", RFC 6761, DOI 10.17487/RFC6761, February 2013, <<https://www.rfc-editor.org/info/rfc6761>>.
- [RFC8244] Lemon, T., Droms, R., and W. Kumari, "Special-Use Domain Names Problem Statement", RFC 8244, DOI 10.17487/RFC8244, October 2017, <<https://www.rfc-editor.org/info/rfc8244>>.
- [RFC9476] Kumari, W. and P. Hoffman, "The .alt Special-Use Top-Level Domain", [RFC 9476](#), DOI 10.17487/RFC9476, September 2023, <<https://www.rfc-editor.org/info/rfc9476>>.
- [TorRendSpec] Tor Project, "Tor Rendezvous Specification - Version 3", commit b345ca0, June 2023, <<https://github.com/torproject/torspec/blob/main/rend-spec-v3.txt>>.
- [Tor224] Goulet, D., Kadianakis, G., and N. Mathewson, "Next-Generation Hidden Services in Tor", Appendix A.2 ("Tor's key derivation scheme"), November 2013, <<https://gitweb.torproject.org/torspec.git/tree/proposals/224-rend-spec-ng.txt#n2135>>.
- [SDSI] Rivest, R. L. and B. Lampson, "SDSI - A Simple Distributed Security Infrastructure", October 1996, <<https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=3837e0206bf73e5e8f0ba6db767a2f714ea7c367>>.
- [Kademlia] Maymounkov, P. and D. Mazières, "Kademlia: A Peer-to-peer Information System Based on the XOR Metric", DOI 10.1007/3-540-45748-8_5, 2002, <<https://css.csail.mit.edu/6.824/2014/papers/kademlia.pdf>>.
- [ed25519] Bernstein, D. J., Duif, N., Lange, T., Schwabe, P., and B-Y. Yang, "High-speed high-security signatures", DOI 10.1007/s13389-012-0027-1, 2011, <<https://ed25519.cr.yp.to/ed25519-20110926.pdf>>.
- [GNS] Wachs, M., Schanzenbach, M., and C. Grothoff, "A Censorship-Resistant, Privacy-Enhancing and Fully Decentralized Name System", 13th International Conference on Cryptology and Network Security (CANS), DOI 10.13140/2.1.4642.3044, October 2014, <https://sci-hub.st/10.1007/978-3-319-12280-9_9>.
- [R5N] Evans, N. S. and C. Grothoff, "R5N: Randomized Recursive Routing for Restricted-Route Networks", 5th International Conference on Network and System Security (NSS), DOI 10.1109/ICNSS.2011.6060022, September 2011, <<https://sci-hub.st/10.1109/ICNSS.2011.6060022>>.
- [SecureNS] Grothoff, C., Wachs, M., Ermert, M., and J. Appelbaum, "Toward secure name resolution on the Internet", Computers and Security, Volume 77, Issue C, pp. 694-708, DOI 10.1016/j.cose.2018.01.018, August 2018, <<https://sci-hub.st/https://doi.org/10.1016/j.cose.2018.01.018>>.
- [GNetGNS] GNet e.V., "gnunet.git - GNet core repository", 2023, <<https://git.gnet.org/gnet.git>>.
- [Ascension] GNet e.V., "ascension.git - DNS zones to GNS migrating using incremental zone transfer (AXFR/IXFR)", 2023, <<https://git.gnet.org/ascension.git>>.
- [GNet] GNet e.V., "The GNet Project (Home Page)", 2023, <<https://gnet.org>>.
- [reclaim] GNet e.V., "re:claimID - Self-sovereign, Decentralised Identity Management and Personal Data Sharing", 2023, <<https://reclaim.gnet.org>>.
- [GoGNS] Fix, B., "gnunet-go (Go GNS)", commit 5c815ba, July 2023, <<https://github.com/bfix/gnet-go/tree/master/src/gnet/service/gns>>.

Приложение А. Использование и переход

В этом приложении описываются конкретные примеры использования, которые могут помочь читателю лучше понять протокол. Приведённые ниже соображения не являются нормативной частью протокола GNS и представлены для указания контекста и сведений о предполагаемом разработчиками использовании протокола. Кроме того, здесь рассматриваются пути перехода на описанный протокол.

А.1. Распространение зон

Чтобы стать владельцем зоны, достаточно создать ключ зоны и соответствующий секретный ключ с помощью реализации GNS. С этого момента владелец зоны может управлять записями о ресурсах GNS в базе данных локальной зоны. Затем записи можно опубликовать через реализацию GNS, как описано в разделе 6. Чтобы другие пользователи могли распознать (resolve) записи о ресурсах, сначала нужно распространить соответствующие сведения о зоне. Владелец зоны может раскрыть ключ зоны и метки лишь ограниченному кругу пользователей или сделать их доступными для всех. Совместное использование данных зоны ограниченным кругом пользователей позволяет не только сохранить приватность зоны и записей, но и установить строгие доверительные отношения. Например, банк может отправить своему клиенту письмо с QR-кодом, содержащим GNS-зону банка. Это позволит отсканировать QR-код и установить прочную связь с зоной банка, а также, например, с IP-адресом web-интерфейса банка.

Большинство служб Internet, вероятно, пожелает сделать свои зоны доступными широкому кругу наиболее эффективным способом. Во-первых, разумно предположить, что зоны с высоким уровнем доверия и репутации будут, скорей всего, включены в сопоставления суффиксов с зонами в реализациях. Поэтому распространение зоны через делегирование под такими зонами может стать жизнеспособным способом распространения зон. Например, можно предположить, что организации, подобные ICANN и регистраторам TLD для стран, также будут поддерживать зоны GNS и предоставлять услуги по регистрации и делегированию.

Передовой опыт, особенно в части безопасности и смягчения последствий злоупотреблений, включает методы, позволяющие владельцам зон и будущим регистраторам обрести хорошую репутацию и, в конечном счёте, доверие. Это, конечно, включает качественную защиту секретного ключевого материала зон. Формализация такого опыта выходит за рамки этого документа и ей следует посвятить отдельный документ с учётом соображений из раздела 9.

А.2. Конфигурация Start Zone

Предполагается, что пользователь установит реализацию GNS, если она ещё не представлена такими средствами, как операционная система или браузер. Вполне вероятно, что пользователь поставит реализацию с принятой по умолчанию конфигурацией Start Zone. Это означает, что пользователь сможет распознавать имена GNS, завершающиеся zTLD или суффиксом из сопоставления суффиксов с зонами, на основе заданной по умолчанию конфигурации Start Zone. На этом этапе пользователь может удалить или изменить принятые по умолчанию настройки.

- Удаление сопоставлений суффиксов с зонами может потребоваться при утрате доверия пользователем к владельцу зоны, включённой в сопоставление. Например, это может быть связано с небрежной политикой регистрации, приводящей к фишингу. Изменение сопоставлений или добавление новых позволяет устранить «перфорацию» пространства имён, которая может возникнуть в результате удаления, или просто установить прочные доверительные отношения. Однако для этого пользователю нужно знать ключи соответствующих зон. Эти сведения должны получаться по отдельному каналу (out of band), как показано в Приложении А.1 (банк может отправить пользователю письмо с QR-кодом, содержащим GNS-зону банка и пользователь сканирует QR-код, добавляя новое сопоставление суффикса с именем для этого банка). Другие примеры включают сканирование сведений о зоне с дружественного устройства, витрины магазина или из рекламы. Уровень доверия к соответствующей зоне зависит от контекста и, вероятно, будет разным у пользователей. Доверие к зоне, представленной письмом из банка, к которому может быть приложена кредитная карта, будет отличаться от доверия к зоне из случайной рекламы на улице. Однако уровень доверия сразу же виден пользователю и может быть отражён в локальном именовании.
- Пользователям, которые являются клиентами, следует облегчать изменение конфигурации Start Zone, например, предоставляя считыватель QR-кода или иные механизмы импорта. В идеале реализации нужно следовать передовому опыту с учётом применимых соображений из раздела 9. Формализация этого выходит за рамки этой спецификации.

А.3. Глобально уникальные имена и Web

Виртуальный хостинг HTTP и указание имени сервера TLS (SNI) широко распространены в Web. Клиенты HTTP предоставляют имя DNS в поле Host заголовка HTTP или при согласовании TLS. Это позволяет серверу HTTP обслуживать указанный виртуальный хост с соответствующим сертификатом TLS. Для этого требуется глобальная уникальность имён DNS.

В GNS не все имена уникальны в глобальном масштабе, однако запись о ресурсе GNS можно представить конкатенацией метки GNS и zTLD зоны. Глобальные имена GNS неудобны для запоминания, но их можно использовать в упомянутых выше случаях. Рассмотрим имя GNS `www.example.gns.alt`, введённое в понимающем GNS клиенте HTTP. Сначала `www.example.gns.alt` распознаётся в GNS с возвратом набора записей, затем клиент HTTP определяет виртуальный хост, как описано ниже.

При наличии в наборе записи LEHO (параграф 5.3.1) с `www.example.com` клиент HTTP использует это имя в поле заголовка HTTP Host.

```
GET / HTTP/1.1
Host: www.example.com
```

Если записи LEHO нет, требуется дополнительное распознавание GNS, чтобы проверить, указывает ли `www.example.gns.alt` на запись делегирования зоны, что подразумевает публикацию исходно распознанного набора

записей под меткой вершины. Если это так, уникальное имя GNS является просто zTLD-представлением делегированной зоны

```
GET / HTTP/1.1
Host: 000G0037FH3QTVCK15Y8VCCNRVWPV17ZC7TSGB1C9ZG2TPGHZVVFV1GMG3W
```

Если записи делегирования зоны для www.example.gns.alt нет, уникальное имя GNS является конкатенацией левой метки (например, www) и zTLD-представления зоны

```
GET / HTTP/1.1
Host: www.000G0037FH3QTVCK15Y8VCCNRVWPV17ZC7TSGB1C9ZG2TPGHZVVFV1GMG3W
```

Отметим, что второе распознавание GNS не требует дополнительной сетевой операции, поскольку отличается лишь локальная обработка записи в соответствии с исключением, указанным в последнем предложении параграфа 7.3.4.

Если клиент HTTP является браузером, использование уникального имени GNS для виртуального хоста или TLS SNI не обязательно показывать пользователю. Например, в поле ввода URL может указываться имя www.example.gns.alt, даже когда в поле заголовка Host используется иное уникальное имя.

A.4. Пути перехода

Распознавание DNS встроено во многие имеющиеся программные компоненты - чаще всего в операционные системы и клиенты HTTP. Здесь рассмотрены возможные пути перехода к распознаванию имён GNS для тех и других.

Одним из способов эффективного распознавания имён GNS является реализация серверов DNS с поддержкой GNS. Для локальных запросов DNS применяется перенаправление или явная настройка на распознавание таким локальным сервером (DNS-to-GNS). Этот сервер DNS пытается интерпретировать все входящие запросы для имён как запросы распознавания GNS. Если для имени не найдено Start Zone и имя не завершается zTLD, сервер пытается распознать это имя в DNS. В иных случаях имя распознаётся в GNS, полученный набор записей преобразуется в пакет отклика DNS, который возвращается запрашивающему. Реализация сервера DNS-to-GNS представлена в [GNUnet]. Похожий подход обеспечивается использованием расширений операционной системы, таких как NSS [nsswitch], позволяющих администратору системы настроить подключаемые модули (plugin) для распознавания имён. Модуль GNS nsswitch можно применять аналогично использованию сервера DNS-to-GNS. Совместимая с glibc реализация модуля nsswitch для GNS представлена в [GNUnet].

Эти методы обычно подходят и для клиентов HTTP, однако эти клиенты обычно применяются в сочетании с TLS. Для проверки сертификатов TLS (в частности, SNI) нужна дополнительная логика в клиентах HTTP при работе с именами GNS (Приложение A.3). Такую функциональность в целях перехода можно обеспечить с помощью локального прокси SOCKS5 [RFC1928] с поддержкой GNS для распознавания доменных имён. Прокси SOCKS5, как и сервер DNS-to-GNS, может распознавать имена GNS и DNS. При запросе соединения TLS по имени GNS прокси SOCKS5 может прервать завершить соединение TLS и сам организовать защищённое соединение с нужным хостом, используя для этого записи LEHO и TLSA из набора записей для имени GNS. Прокси должен предоставить клиенту HTTP локально доверенный сертификат для имени GNS, для чего обычно нужно создать и настроить локальную привязку доверия в браузере. Реализация SOCKS5-прокси представлена в [GNUnet].

Приложение В. Примеры

В.1. Пример распознавания AAAA

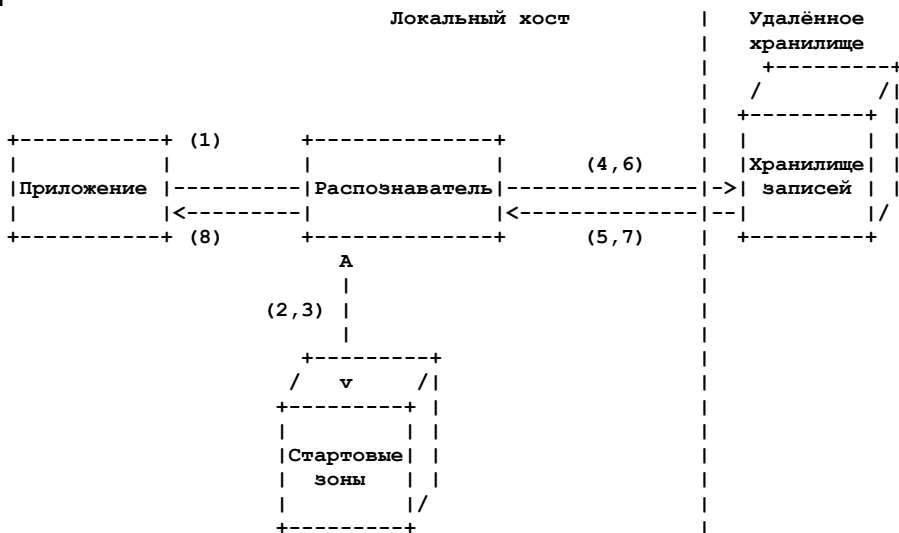


Рисунок 24. Пример распознавания адреса IPv6.

1. Поиск записи AAAA для имени www.example.gnu.gns.alt.
2. Определение Start Zone для www.example.gnu.gns.alt.
3. Start Zone: zkey0, остаток: www.example.
4. Расчёт q0=SHA512(ZKDF(zkey0, "example")) и инициирование GET(q0).
5. Извлечение и расшифровка RRBLOCK, состоящего из одной записи PKEY с zkey1.
6. Расчёт q1=SHA512(ZKDF(zkey1, "www")) и инициирование GET(q1).
7. Извлечение RRBLOCK с одной записью AAAA содержащей адрес IPv6 2001:db8::1.
8. Возврат набора записей приложению.

В.2. Пример распознавания с REDIRECT

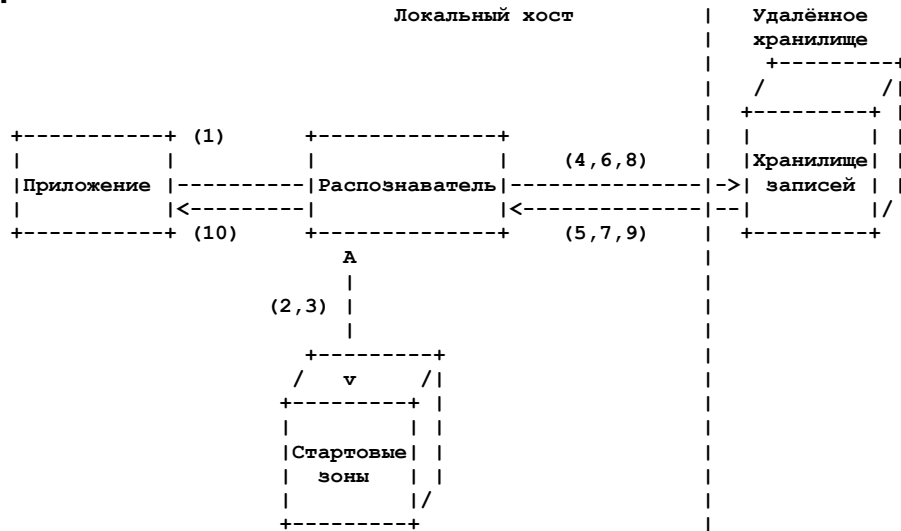


Рисунок 25. Пример распознавания адреса IPv6 с Redirect.

1. Поиск записи AAAA для имени www.example.tld.gns.alt.
2. Определение Start Zone для www.example.tld.gns.alt.
3. Start Zone: zkey0, остаток: www.example.
4. Расчёт $q_0 = \text{SHA512}(\text{ZKDF}(\text{zkey0}, \text{"example"}))$ и инициирование GET(q_0).
5. Извлечение и расшифровка RRBLOCK, состоящего из одной записи PKEY с zkey1.
6. Расчёт $q_1 = \text{SHA512}(\text{ZKDF}(\text{zkey1}, \text{"www"}))$ и инициирование GET(q_1).
7. Извлечение и расшифровка RRBLOCK, состоящего из одной записи REDIRECT с www2.+.
8. Расчет $q_2 = \text{SHA512}(\text{ZKDF}(\text{zkey1}, \text{"www2"}))$ и инициирование GET(q_2).
9. Извлечение и расшифровка RRBLOCK, состоящего из одной записи AAAA с адресом IPv6 2001:db8::1.
10. Возврат набора записей приложению.

В.3. Пример распознавания GNS2DNS



Рисунок 26. Пример распознавания адреса IPv6 с DNS Handover.

1. Поиск записи AAAA для имени www.example.gnu.gns.alt.
2. Определение Start Zone для www.example.gnu.gns.alt.
3. Start Zone: zkey0, остаток: www.example.
4. Расчёт $q_0 = \text{SHA512}(\text{ZKDF}(\text{zkey0}, \text{"example"}))$ и инициирование GET(q_0).
5. Извлечение и расшифровка RRBLOCK, состоящего из одной записи GNS2DNS с именем example.com и адресом IPv4 сервера DNS 192.0.2.1.
6. Использование системного распознавателя для поиска записи AAAA по DNS-имени www.example.com.
7. Получение отклика DNS с одной записью AAAA содержащей адрес IPv6 2001:db8::1.
8. Возврат набора записей приложению.

Приложение C. Base32GNS

Кодирование преобразует массив байтов в строку символов, декодирование выполняет обратное преобразование. При декодировании возникает отказ, если входная строка содержит символы, не включённые в заданный набор.

В таблице 4 показаны символы кодирования и декодирования для каждого значения. Каждый символ кодируется 5 битами. Например, символ A или a декодируется в 5-битовое значение, а 5-блок со значением 18 кодируется символом J. Если размер строки байтов для кодирования не кратен 5 битам, строка дополняется нулями. Для повышения отказоустойчивости при распознавании символов буква U **должна** декодироваться в то же значение, что и буква V.

Таблица 4. Алфавит Base32GNS с дополнительным символом кодирования U.

Значение символа	Символ декодирования	Символ кодирования
0	0 O o	0
1	1 I i L l	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7
8	8	8
9	9	9
10	A a	A
11	B b	B
12	C c	C
13	D d	D
14	E e	E
15	F f	F
16	G g	G
17	H h	H
18	J j	J
19	K k	K
20	M m	M
21	N n	N
22	P p	P
23	Q q	Q
24	R r	R
25	S s	S
26	T t	T
27	V v U u	V
28	W w	W
29	X x	X
30	Y y	Y
31	Z z	Z

Приложение D. Тестовые векторы

Представленные ниже тестовые векторы могут служить для проверки совместимости с этой спецификацией. Если не указано иное, тестовые векторы представляются шестнадцатеричными массивами байтов.

D.1. Кодирование и декодирование Base32GNS

Ниже приведены тестовые векторы для кодирования Base32GNS, применяемого с zTLD (входные строки не содержат null-символ).

Base32GNS-Encode:

Входная строка: Hello World

Выходная строка: 91JPRV3F41BPYWKCCG

Входные байты: 474e55204e616d652053797374656d

Выходная строка: 8X75A82EC5PPA82KF5SQ8SBD

Base32GNS-Decode:

Входная строка: 91JPRV3F41BPYWKCCG

Выходная строка: Hello World

Входная строка: 91JPRU3F41BPYWKCCG

Выходная строка: Hello World

D.2. Наборы записей

Тестовые векторы включают наборы записей разных типов с флагами для зон PKEY и EDKEY. Включены метки с символами UTF-8 для демонстрации меток на других языках.

(1) Зона PKEY с меткой ASCII и записью делегирования.

Секретный ключ зоны (d, big-endian):

50 d7 b6 52 a4 ef ea df

f3 73 96 90 97 85 e5 95

21 71 a0 21 78 c8 e7 d4

50 fa 90 79 25 fa fd 98

Идентификатор зоны (ztype|zkey):

00 01 00 00 67 7c 47 7d

```
2d 93 09 7c 85 b1 95 c6
f9 6d 84 ff 61 f5 98 2c
2c 4f e0 2d 5a 11 fe df
b0 c2 90 1f
```

zTLD:

```
000G0037FH3QTВСК15Y8BCCNRVWPV17ZC7TSGB1C9ZG2TPGHZVFFV1GMG3W
```

Метка:

```
74 65 73 74 64 65 6c 65
67 61 74 69 6f 6e
```

Число записей (integer): 1

Запись 0 := (

```
EXPIRATION: 8143584694000000 us
00 1c ee 8c 10 e2 59 80
```

```
DATA_SIZE:
00 20
```

```
TYPE:
00 01 00 00
```

```
FLAGS: 00 01
```

DATA:

```
21 e3 b3 0f f9 3b c6 d3
5a c8 c6 e0 e1 3a fd ff
79 4c b7 b4 4b bb c7 48
d2 59 d0 a0 28 4d be 84
```

)

RDATA:

```
00 1c ee 8c 10 e2 59 80
00 20 00 01 00 01 00 00
21 e3 b3 0f f9 3b c6 d3
5a c8 c6 e0 e1 3a fd ff
79 4c b7 b4 4b bb c7 48
d2 59 d0 a0 28 4d be 84
```

Шифрование NONCE|EXPIRATION|BLOCK COUNTER:

```
e9 0a 00 61 00 1c ee 8c
10 e2 59 80 00 00 00 01
```

Ключ шифрования (K):

```
86 4e 71 38 ea e7 fd 91
a3 01 36 89 9c 13 2b 23
ac eb db 2c ef 43 cb 19
f6 bf 55 b6 7d b9 b3 b3
```

Ключ хранилища (q):

```
4a dc 67 c5 ec ee 9f 76
98 6a bd 71 c2 22 4a 3d
ce 2e 91 70 26 c9 a0 9d
fd 44 ce f3 d2 0f 55 a2
73 32 72 5a 6c 8a fb bb
b0 f7 ec 9a f1 cc 42 64
12 99 40 6b 04 fd 9b 5b
57 91 f8 6c 4b 08 d5 f4
```

ZKDF(zkey, label):

```
18 2b b6 36 ed a7 9f 79
57 11 bc 27 08 ad bb 24
2a 60 44 6a d3 c3 08 03
12 1d 03 d3 48 b7 ce b6
```

Производный секретный ключ (d', big-endian):

```
0a 4c 5e 0f 00 63 df ce
db c8 c7 f2 b2 2c 03 0c
86 28 b2 c2 cb ac 9f a7
29 aa e6 1f 89 db 3e 9c
```

BDATA:

```
0c 1e da 5c c0 94 a1 c7
a8 88 64 9d 25 fa ee bd
60 da e6 07 3d 57 d8 ae
8d 45 5f 4f 13 92 c0 74
e2 6a c6 69 bd ee c2 34
62 b9 62 95 2c c6 e9 eb
```

RRBLOCK:

```
00 00 00 a0 00 01 00 00
18 2b b6 36 ed a7 9f 79
57 11 bc 27 08 ad bb 24
2a 60 44 6a d3 c3 08 03
```

```
12 1d 03 d3 48 b7 ce b6
0a d1 0b c1 3b 40 3b 5b
25 61 26 b2 14 5a 6f 60
c5 14 f9 51 ff a7 66 f7
a3 fd 4b ac 4a 4e 19 90
05 5c b8 7e 8d 1b fd 19
aa 09 a4 29 f7 29 e9 f5
c6 ee c2 47 0a ce e2 22
07 59 e9 e3 6c 88 6f 35
00 1c ee 8c 10 e2 59 80
0c 1e da 5c c0 94 a1 c7
a8 88 64 9d 25 fa ee bd
60 da e6 07 3d 57 d8 ae
8d 45 5f 4f 13 92 c0 74
e2 6a c6 69 bd ee c2 34
62 b9 62 95 2c c6 e9 eb
```

(2) Зона RKEY с меткой UTF-8 и тремя записями.

Секретный ключ зоны (d, big-endian):

```
50 d7 b6 52 a4 ef ea df
f3 73 96 90 97 85 e5 95
21 71 a0 21 78 c8 e7 d4
50 fa 90 79 25 fa fd 98
```

Идентификатор зоны (ztype|zkey):

```
00 01 00 00 67 7c 47 7d
2d 93 09 7c 85 b1 95 c6
f9 6d 84 ff 61 f5 98 2c
2c 4f e0 2d 5a 11 fe df
b0 c2 90 1f
```

zTLD:

```
000G0037FH3QTBCK15Y8BCCNRVWPV17ZC7TSGB1C9ZG2TPGNZVFV1GMG3W
```

Метка:

```
e5 a4 a9 e4 b8 8b e7 84
a1 e6 95 b5
```

Число записей (integer): 3

Запись 0 := (

```
EXPIRATION: 8143584694000000 us
00 1c ee 8c 10 e2 59 80
```

```
DATA_SIZE:
00 10
```

```
TYPE:
00 00 00 1c
```

```
FLAGS: 00 00
```

```
DATA:
00 00 00 00 00 00 00 00
00 00 00 00 de ad be ef
```

)

Запись 1 := (

```
EXPIRATION: 17999736901000000 us
00 3f f2 aa 54 08 db 40
```

```
DATA_SIZE:
00 06
```

```
TYPE:
00 01 00 01
```

```
FLAGS: 00 00
```

```
DATA:
e6 84 9b e7 a7 b0
```

)

Запись 2 := (

```
EXPIRATION: 11464693629000000 us
00 28 bb 13 ff 37 19 40
```

```
DATA_SIZE:
00 0b
```

```
TYPE:
00 00 00 10
```

```
FLAGS: 00 04
```

```
DATA:
```

```
48 65 6c 6c 6f 20 57 6f
72 6c 64
```

)

RDATA:

```
00 1c ee 8c 10 e2 59 80
00 10 00 00 00 00 00 1c
00 00 00 00 00 00 00 00
00 00 00 00 de ad be ef
00 3f f2 aa 54 08 db 40
00 06 00 00 00 01 00 01
e6 84 9b e7 a7 b0 00 28
bb 13 ff 37 19 40 00 0b
00 04 00 00 00 10 48 65
6c 6c 6f 20 57 6f 72 6c
64 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

Шифрование NONCE|EXPIRATION|BLOCK COUNTER:

```
ee 96 33 c1 00 1c ee 8c
10 e2 59 80 00 00 00 01
```

Ключ шифрования (K):

```
fb 3a b5 de 23 bd da e1
99 7a af 7b 92 c2 d2 71
51 40 8b 77 af 7a 41 ac
79 05 7c 4d f5 38 3d 01
```

Ключ хранилища (q):

```
af f0 ad 6a 44 09 73 68
42 9a c4 76 df a1 f3 4b
ee 4c 36 e7 47 6d 07 aa
64 63 ff 20 91 5b 10 05
c0 99 1d ef 91 fc 3e 10
90 9f 87 02 c0 be 40 43
67 78 c7 11 f2 ca 47 d5
5c f0 b5 4d 23 5d a9 77
```

ZKDF(zkey, label):

```
a5 12 96 df 75 7e e2 75
ca 11 8d 4f 07 fa 7a ae
55 08 bc f5 12 aa 41 12
14 29 d4 a0 de 9d 05 7e
```

Производный секретный ключ (d', big-endian):

```
0a be 56 d6 80 68 ab 40
e1 44 79 0c de 9a cf 4d
78 7f 2d 3c 63 b8 53 05
74 6e 68 03 32 15 f2 ab
```

BDATA:

```
d8 c2 8d 2f d6 96 7d 1a
b7 22 53 f2 10 98 b8 14
a4 10 be 1f 59 98 de 03
f5 8f 7e 7c db 7f 08 a6
16 51 be 4d 0b 6f 8a 61
df 15 30 44 0b d7 47 dc
f0 d7 10 4f 6b 8d 24 c2
ac 9b c1 3d 9c 6f e8 29
05 25 d2 a6 d0 f8 84 42
67 a1 57 0e 8e 29 4d c9
3a 31 9f cf c0 3e a2 70
17 d6 fd a3 47 b4 a7 94
97 d7 f6 b1 42 2d 4e dd
82 1c 19 93 4e 96 c1 aa
87 76 57 25 d4 94 c7 64
b1 55 dc 6d 13 26 91 74
```

RRBLOCK:

```
00 00 00 f0 00 01 00 00
a5 12 96 df 75 7e e2 75
ca 11 8d 4f 07 fa 7a ae
55 08 bc f5 12 aa 41 12
14 29 d4 a0 de 9d 05 7e
08 5b d6 5f d4 85 10 51
ba ce 2a 45 2a fc 8a 7e
4f 6b 2c 1f 74 f0 20 35
d9 64 1a cd ba a4 66 e0
00 ce d6 f2 d2 3b 63 1c
8e 8a 0b 38 e2 ba e7 9a
22 ca d8 1d 4c 50 d2 25
35 8e bc 17 ac 0f 89 9e
```

```

00 1c ee 8c 10 e2 59 80
d8 c2 8d 2f d6 96 7d 1a
b7 22 53 f2 10 98 b8 14
a4 10 be 1f 59 98 de 03
f5 8f 7e 7c db 7f 08 a6
16 51 be 4d 0b 6f 8a 61
df 15 30 44 0b d7 47 dc
f0 d7 10 4f 6b 8d 24 c2
ac 9b c1 3d 9c 6f e8 29
05 25 d2 a6 d0 f8 84 42
67 a1 57 0e 8e 29 4d c9
3a 31 9f cf c0 3e a2 70
17 d6 fd a3 47 b4 a7 94
97 d7 f6 b1 42 2d 4e dd
82 1c 19 93 4e 96 c1 aa
87 76 57 25 d4 94 c7 64
b1 55 dc 6d 13 26 91 74

```

(3) Зона EDKEY с меткой ASCII и записью делегирования.

Секретный ключ зоны (d):

```

5a f7 02 0e e1 91 60 32
88 32 35 2b bc 6a 68 a8
d7 1a 7c be 1b 92 99 69
a7 c6 6d 41 5a 0d 8f 65

```

Идентификатор зоны (ztype|zkey):

```

00 01 00 14 3c f4 b9 24
03 20 22 f0 dc 50 58 14
53 b8 5d 93 b0 47 b6 3d
44 6c 58 45 cb 48 44 5d
db 96 68 8f

```

zTLD:

```
000G051WYJWJ80S04BRDRM2R2H9VQGCKP13VCFA4DHC4BJT88HEXQ5K8HW
```

Метка:

```

74 65 73 74 64 65 6c 65
67 61 74 69 6f 6e

```

Число записей (integer): 1

Запись 0 := (

```

EXPIRATION: 8143584694000000 us
00 1c ee 8c 10 e2 59 80

```

DATA SIZE:

```
00 20
```

TYPE:

```
00 01 00 00
```

FLAGS: 00 01

DATA:

```

21 e3 b3 0f f9 3b c6 d3
5a c8 c6 e0 e1 3a fd ff
79 4c b7 b4 4b bb c7 48
d2 59 d0 a0 28 4d be 84

```

)

RDATA:

```

00 1c ee 8c 10 e2 59 80
00 20 00 01 00 01 00 00
21 e3 b3 0f f9 3b c6 d3
5a c8 c6 e0 e1 3a fd ff
79 4c b7 b4 4b bb c7 48
d2 59 d0 a0 28 4d be 84

```

Шифрование NONCE|EXPIRATION:

```

98 13 2e a8 68 59 d3 5c
88 bf d3 17 fa 99 1b cb
00 1c ee 8c 10 e2 59 80

```

Ключ шифрования (K):

```

85 c4 29 a9 56 7a a6 33
41 1a 96 91 e9 09 4c 45
28 16 72 be 58 60 34 aa
e4 a2 a2 cc 71 61 59 e2

```

Ключ хранилища (q):

```

ab aa ba c0 e1 24 94 59
75 98 83 95 aa c0 24 1e
55 59 c4 1c 40 74 e2 55
7b 9f e6 d1 54 b6 14 fb
cd d4 7f c7 f5 1d 78 6d
c2 e0 b1 ec e7 60 37 c0

```

```
a1 57 8c 38 4e c6 1d 44
56 36 a9 4e 88 03 29 e9
```

ZKDF(zkey, label):

```
9b f2 33 19 8c 6d 53 bb
db ac 49 5c ab d9 10 49
a6 84 af 3f 40 51 ba ca
b0 dc f2 1c 8c f2 7a 1a
```

nonce := SHA-256(dh[32..63] || h):

```
14 f2 c0 6b ed c3 aa 2d
f0 71 13 9c 50 39 34 f3
4b fa 63 11 a8 52 f2 11
f7 3a df 2e 07 61 ec 35
```

Производный секретный ключ (d', big-endian):

```
3b 1b 29 d4 23 0b 10 a8
ec 4d a3 c8 6e db 88 ea
cd 54 08 5c 1d db 63 f7
a9 d7 3f 7c cb 2f c3 98
```

BDATA:

```
57 7c c6 c9 5a 14 e7 04
09 f2 0b 01 67 e6 36 d0
10 80 7c 4f 00 37 2d 69
8c 82 6b d9 2b c2 2b d6
bb 45 e5 27 7c 01 88 1d
6a 43 60 68 e4 dd f1 c6
b7 d1 41 6f af a6 69 7c
25 ed d9 ea e9 91 67 c3
```

RRBLOCK:

```
00 00 00 b0 00 01 00 14
9b f2 33 19 8c 6d 53 bb
db ac 49 5c ab d9 10 49
a6 84 af 3f 40 51 ba ca
b0 dc f2 1c 8c f2 7a 1a
9f 56 a8 86 ea 73 9d 59
17 50 8f 9b 75 56 39 f3
a9 ac fa ed ed ca 7f bf
a7 94 b1 92 e0 8b f9 ed
4c 7e c8 59 4c 9f 7b 4e
19 77 4f f8 38 ec 38 7a
8f 34 23 da ac 44 9f 59
db 4e 83 94 3f 90 72 00
00 1c ee 8c 10 e2 59 80
57 7c c6 c9 5a 14 e7 04
09 f2 0b 01 67 e6 36 d0
10 80 7c 4f 00 37 2d 69
8c 82 6b d9 2b c2 2b d6
bb 45 e5 27 7c 01 88 1d
6a 43 60 68 e4 dd f1 c6
b7 d1 41 6f af a6 69 7c
25 ed d9 ea e9 91 67 c3
```

(4) Зона EDKEY с меткой UTF-8 и тремя записями.

Секретный ключ зоны (d):

```
5a f7 02 0e e1 91 60 32
88 32 35 2b bc 6a 68 a8
d7 1a 7c be 1b 92 99 69
a7 c6 6d 41 5a 0d 8f 65
```

Идентификатор зоны (ztype|zkey):

```
00 01 00 14 3c f4 b9 24
03 20 22 f0 dc 50 58 14
53 b8 5d 93 b0 47 b6 3d
44 6c 58 45 cb 48 44 5d
db 96 68 8f
```

zTLD:

```
000G051WYJWJ80S04BRDRM2R2H9VGQCKP13VCFA4DHC4BJT88HEXQ5K8HW
```

Метка:

```
e5 a4 a9 e4 b8 8b e7 84
a1 e6 95 b5
```

Число записей (integer): 3

Запись 0 := (

```
EXPIRATION: 8143584694000000 us
00 1c ee 8c 10 e2 59 80
```

DATA_SIZE:

```
00 10
```

TYPE:

```
00 00 00 1c
```

```
FLAGS: 00 00
```

```
DATA:
```

```
00 00 00 00 00 00 00 00  
00 00 00 00 de ad be ef
```

```
)
```

```
Запись 1 := (
```

```
EXPIRATION: 17999736901000000 us  
00 3f f2 aa 54 08 db 40
```

```
DATA_SIZE:
```

```
00 06
```

```
TYPE:
```

```
00 01 00 01
```

```
FLAGS: 00 00
```

```
DATA:
```

```
e6 84 9b e7 a7 b0
```

```
)
```

```
Запись 2 := (
```

```
EXPIRATION: 11464693629000000 us  
00 28 bb 13 ff 37 19 40
```

```
DATA_SIZE:
```

```
00 0b
```

```
TYPE:
```

```
00 00 00 10
```

```
FLAGS: 00 04
```

```
DATA:
```

```
48 65 6c 6c 6f 20 57 6f  
72 6c 64
```

```
)
```

```
RDATA:
```

```
00 1c ee 8c 10 e2 59 80  
00 10 00 00 00 00 00 1c  
00 00 00 00 00 00 00 00  
00 00 00 00 de ad be ef  
00 3f f2 aa 54 08 db 40  
00 06 00 00 00 01 00 01  
e6 84 9b e7 a7 b0 00 28  
bb 13 ff 37 19 40 00 0b  
00 04 00 00 00 10 48 65  
6c 6c 6f 20 57 6f 72 6c  
64 00 00 00 00 00 00 00  
00 00 00 00 00 00 00 00  
00 00 00 00 00 00 00 00  
00 00 00 00 00 00 00 00  
00 00 00 00 00 00 00 00  
00 00 00 00 00 00 00 00
```

```
Шифрование NONCE|EXPIRATION:
```

```
bb 0d 3f 0f bd 22 42 77  
50 da 5d 69 12 16 e6 c9  
00 1c ee 8c 10 e2 59 80
```

```
Ключ шифрования (K):
```

```
3d f8 05 bd 66 87 aa 14  
20 96 28 c2 44 b1 11 91  
88 c3 92 56 37 a4 1e 5d  
76 49 6c 29 45 dc 37 7b
```

```
Ключ хранилища (q):
```

```
ba f8 21 77 ee c0 81 e0  
74 a7 da 47 ff c6 48 77  
58 fb 0d f0 1a 6c 7f bb  
52 fc 8a 31 be f0 29 af  
74 aa 0d c1 5a b8 e2 fa  
7a 54 b4 f5 f6 37 f6 15  
8f a7 f0 3c 3f ce be 78  
d3 f9 d6 40 aa c0 d1 ed
```

```
ZKDF(zkey, label):
```

```
74 f9 00 68 f1 67 69 53  
52 a8 a6 c2 eb 98 48 98  
c5 3a cc a0 98 04 70 c6  
c8 12 64 cb dd 78 ad 11
```

```

nonce := SHA-256(dh[32..63] || h):
f8 6a b5 33 8a 74 d7 a1
d2 77 ea 11 ff 95 cb e8
3a cf d3 97 3b b4 ab ca
0a 1b 60 62 c3 7a b3 9c

```

```

Производный секретный ключ (d', big-endian):
17 c0 68 a6 c3 f7 20 de
0e 1b 69 ff 3f 53 e0 5d
3f e5 c5 b0 51 25 7a 89
a6 3c 1a d3 5a c4 35 58

```

BDATA:

```

4e b3 5a 50 d4 0f e1 a4
29 c7 f4 b2 67 a0 59 de
4e 2c 8a 89 a5 ed 53 d3
d4 92 58 59 d2 94 9f 7f
30 d8 a2 0c aa 96 f8 81
45 05 2d 1c da 04 12 49
8f f2 5f f2 81 6e f0 ce
61 fe 69 9b fa c7 2c 15
dc 83 0e a9 b0 36 17 1c
cf ca bb dd a8 de 3c 86
ed e2 95 70 d0 17 4b 82
82 09 48 a9 28 b7 f0 0e
fb 40 1c 10 fe 80 bb bb
02 76 33 1b f7 f5 1b 8d
74 57 9c 14 14 f2 2d 50
1a d2 5a e2 49 f5 bb f2
a6 c3 72 59 d1 75 e4 40
b2 94 39 c6 05 19 cb b1

```

RRBLOCK:

```

00 00 01 00 00 01 00 14
74 f9 00 68 f1 67 69 53
52 a8 a6 c2 eb 98 48 98
c5 3a cc a0 98 04 70 c6
c8 12 64 cb dd 78 ad 11
75 6d 2c 15 7a d2 ea 4f
c0 b1 b9 1c 08 03 79 44
61 d3 de f2 0d d1 63 6c
fe dc 03 89 c5 49 d1 43
6c c3 5b 4e 1b f8 89 5a
64 6b d9 a6 f4 6b 83 48
1d 9c 0e 91 d4 e1 be bb
6a 83 52 6f b7 25 2a 06
00 1c ee 8c 10 e2 59 80
4e b3 5a 50 d4 0f e1 a4
29 c7 f4 b2 67 a0 59 de
4e 2c 8a 89 a5 ed 53 d3
d4 92 58 59 d2 94 9f 7f
30 d8 a2 0c aa 96 f8 81
45 05 2d 1c da 04 12 49
8f f2 5f f2 81 6e f0 ce
61 fe 69 9b fa c7 2c 15
dc 83 0e a9 b0 36 17 1c
cf ca bb dd a8 de 3c 86
ed e2 95 70 d0 17 4b 82
82 09 48 a9 28 b7 f0 0e
fb 40 1c 10 fe 80 bb bb
02 76 33 1b f7 f5 1b 8d
74 57 9c 14 14 f2 2d 50
1a d2 5a e2 49 f5 bb f2
a6 c3 72 59 d1 75 e4 40
b2 94 39 c6 05 19 cb b1

```

D.3. Отзыв зоны

Ниже представлен пример отзыва для зоны PKEY.

Секретный ключ зоны (d, big-endian):

```

6f ea 32 c0 5a f5 8b fa
97 95 53 d1 88 60 5f d5
7d 8b f9 cc 26 3b 78 d5
f7 47 8c 07 b9 98 ed 70

```

Идентификатор зоны (ztype|zkey):

```

00 01 00 00 2c a2 23 e8
79 ec c4 bb de b5 da 17
31 92 81 d6 3b 2e 3b 69
55 f1 c3 77 5c 80 4a 98
d5 f8 dd aa

```

zTLD:

```
000G001CM8HYGYFCRJXXXDET2WRS50EP7CQ3PTANY71QE0409ACDBY6XN8
```

Сложность (базовая 5, эпохи 2): 7

Подписанное сообщение:

```
00 00 00 34 00 00 00 03
00 05 ff 1c 56 e4 b2 68
00 01 00 00 2c a2 23 e8
79 ec c4 bb de b5 da 17
31 92 81 d6 3b 2e 3b 69
55 f1 c3 77 5c 80 4a 98
d5 f8 dd aa
```

Доказательство:

```
00 05 ff 1c 56 e4 b2 68
00 00 39 5d 18 27 c0 00
38 0b 54 aa 70 16 ac a2
38 0b 54 aa 70 16 ad 62
38 0b 54 aa 70 16 af 3e
38 0b 54 aa 70 16 af 93
38 0b 54 aa 70 16 b0 bf
38 0b 54 aa 70 16 b0 ee
38 0b 54 aa 70 16 b1 c9
38 0b 54 aa 70 16 b1 e5
38 0b 54 aa 70 16 b2 78
38 0b 54 aa 70 16 b2 b2
38 0b 54 aa 70 16 b2 d6
38 0b 54 aa 70 16 b2 e4
38 0b 54 aa 70 16 b3 2c
38 0b 54 aa 70 16 b3 5a
38 0b 54 aa 70 16 b3 9d
38 0b 54 aa 70 16 b3 c0
38 0b 54 aa 70 16 b3 dd
38 0b 54 aa 70 16 b3 f4
38 0b 54 aa 70 16 b4 42
38 0b 54 aa 70 16 b4 76
38 0b 54 aa 70 16 b4 8c
38 0b 54 aa 70 16 b4 a4
38 0b 54 aa 70 16 b4 c9
38 0b 54 aa 70 16 b4 f0
38 0b 54 aa 70 16 b4 f7
38 0b 54 aa 70 16 b5 79
38 0b 54 aa 70 16 b6 34
38 0b 54 aa 70 16 b6 8e
38 0b 54 aa 70 16 b7 b4
38 0b 54 aa 70 16 b8 7e
38 0b 54 aa 70 16 b8 f8
38 0b 54 aa 70 16 b9 2a
00 01 00 00 2c a2 23 e8
79 ec c4 bb de b5 da 17
31 92 81 d6 3b 2e 3b 69
55 f1 c3 77 5c 80 4a 98
d5 f8 dd aa 08 ca ff de
3c 6d f1 45 f7 e0 79 81
15 37 b2 b0 42 2d 5e 1f
b2 01 97 81 ec a2 61 d1
f9 d8 ea 81 0a bc 2f 33
47 7f 04 e3 64 81 11 be
71 c2 48 82 1a d6 04 f4
94 e7 4d 0b f5 11 d2 c1
62 77 2e 81
```

Ниже приведён пример отзыва для зоны EDKEY

Секретный ключ зоны (d):

```
5a f7 02 0e e1 91 60 32
88 32 35 2b bc 6a 68 a8
d7 1a 7c be 1b 92 99 69
a7 c6 6d 41 5a 0d 8f 65
```

Идентификатор зоны (ztype|zkey):

```
00 01 00 14 3c f4 b9 24
03 20 22 f0 dc 50 58 14
53 b8 5d 93 b0 47 b6 3d
44 6c 58 45 cb 48 44 5d
db 96 68 8f
```

zTLD:

000G051WYJWJ80S04BRDRM2R2H9VGQCKP13VCFA4DHC4BJT88HEXQ5K8HW

Сложность (базовая 5, эпохи 2): 7

Подписанное сообщение:

```
00 00 00 34 00 00 00 03
00 05 ff 1c 57 35 42 bd
00 01 00 14 3c f4 b9 24
03 20 22 f0 dc 50 58 14
53 b8 5d 93 b0 47 b6 3d
44 6c 58 45 cb 48 44 5d
```

Доказательство:

```
00 05 ff 1c 57 35 42 bd
00 00 39 5d 18 27 c0 00
58 4c 93 3c b0 99 2a 08
58 4c 93 3c b0 99 2d f7
58 4c 93 3c b0 99 2e 21
58 4c 93 3c b0 99 2e 2a
58 4c 93 3c b0 99 2e 53
58 4c 93 3c b0 99 2e 8e
58 4c 93 3c b0 99 2f 13
58 4c 93 3c b0 99 2f 2d
58 4c 93 3c b0 99 2f 3c
58 4c 93 3c b0 99 2f 41
58 4c 93 3c b0 99 2f fd
58 4c 93 3c b0 99 30 33
58 4c 93 3c b0 99 30 82
58 4c 93 3c b0 99 30 a2
58 4c 93 3c b0 99 30 e1
58 4c 93 3c b0 99 31 ce
58 4c 93 3c b0 99 31 de
58 4c 93 3c b0 99 32 12
58 4c 93 3c b0 99 32 4e
58 4c 93 3c b0 99 32 9f
58 4c 93 3c b0 99 33 31
58 4c 93 3c b0 99 33 87
58 4c 93 3c b0 99 33 8c
58 4c 93 3c b0 99 33 e5
58 4c 93 3c b0 99 33 f3
58 4c 93 3c b0 99 34 26
58 4c 93 3c b0 99 34 30
58 4c 93 3c b0 99 34 68
58 4c 93 3c b0 99 34 88
58 4c 93 3c b0 99 34 8a
58 4c 93 3c b0 99 35 4c
58 4c 93 3c b0 99 35 bd
00 01 00 14 3c f4 b9 24
03 20 22 f0 dc 50 58 14
53 b8 5d 93 b0 47 b6 3d
44 6c 58 45 cb 48 44 5d
db 96 68 8f 04 ae 26 f7
63 56 5a b7 aa ab 01 71
72 4f 3c a8 bc c5 1a 98
b7 d4 c9 2e a3 3c d9 34
4c a8 b6 3e 04 53 3a bf
1a 3c 05 49 16 b3 68 2c
5c a8 cb 4d d0 f8 4c 3b
77 48 7a ac 6e ce 38 48
0b a9 d5 00
```

Благодарности

Авторы благодарят всех рецензентов за комментарии. Спасибо D. J. Bernstein, S. Bortzmeier, A. Farrel, E. Lear, and R. Salz за их глубокие и подробные технические обзоры. Спасибо J. Yao и J. Klensin за анализ применимости для других языков. Спасибо Dr. J. Appelbaum за предложение имени GNU Name System и Dr. Richard Stallman за одобрение его использования. Спасибо T. Lange и M. Wachs за ранний вклад в разработку и реализацию GNS. Спасибо NLnet и NGI DISCOVERY за финансирование работы над GNU Name System.

Адреса авторов

Martin Schanzenbach
Fraunhofer AISEC
Lichtenbergstrasse 11
85748 Garching
Germany
Email: martin.schanzenbach@aisec.fraunhofer.de

Christian Grothoff
Berner Fachhochschule
Hoeheweg 80

CH-2501 Biel/Bienne
Switzerland
Email: christian.grothoff@bfh.ch

Bernd Fix
GNUnet e.V.
Boltzmannstrasse 3
85748 Garching
Germany
Email: fix@gnunet.org

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru