Network Working Group
Request for Comments: 2747
Category: Standards Track

F. Baker
Cisco
B. Lindell
USC/ISI
M. Talwar
Microsoft

January 2000

Криптографическая аутентификация RSVP

RSVP Cryptographic Authentication

Статус документа

Этот документ содержит проект стандартного протокола для сообщества Internet и служит запросом для обсуждения в целях развития протокола. Информацию о текущем состоянии стандартизации протокола можно найти в документе Internet Official Protocol Standards (STD 1). Документ может распространяться свободно.

Аннотация

В этом документе описан формат и применение объекта RSVP INTEGRITY для поэтапного обеспечения целостности и аутентификации сообщений RSVP.

1. Введение

Протокол резервирования ресурсов RSVP[1] [1] предназначен для организации в маршрутизаторах и хостах распределенного состояния с целью резервирования ресурсов (в частности) для реализации интегрированного обслуживания. RSVP позволяет отдельным пользователям получать преимущественный доступ к сетевым ресурсам под контролем механизмов управления доступом. Разрешение на организацию резервирования будет зависеть как от доступности запрашиваемых ресурсов на пути передачи через сеть, так и от выполнения заданных правил.

Для обеспечения защиты целостности этого механизма контроля доступа от протокола RSVP требуется способность защитить сообщения от подмены и повреждения. В этом документе определен механизм поэтапной защиты целостности сообщений RSVP. В предложенной схеме передается аутентификационная подпись сообщения, которая создается с использованием секрета Authentication Key и алгоритма хэширования с ключами. Схема обеспечивает защиту от подмены или изменения сообщений. Объект INTEGRITY в каждом сообщении RSVP помечается порядковым номером одноразового использования. Это позволяет получателю идентифицировать повторно используемые сообщения и, следовательно, предотвратить атаки, основанные на повторном использовании (replay) сообщений. Предложенный механизм не обеспечивает защиты конфиденциальности, поскольку сообщения не шифруются. Однако следует помнить, что этот механизм может использоваться при международном обмене данными, а в разных странах могут применяться различные требования к шифрованию данных и экспорту шифров.

> Примечание. В этом документе смысл терминов «отправитель» (sender) и «получатель» (receiver) отличается от принятого в [1]. Термины относятся к системам, расположенным по разные стороны одного интервала (hop) RSVP, при этом отправителем считается система, генерирующая сообщения RSVP.

Алгоритм предотвращения повторного использования сообщений достаточно прост. Отправитель генерирует пакеты с монотонно возрастающими порядковыми номерами. В свою очередь, получатель воспринимает только те пакеты, у которых порядковый номер превышает номер предыдущего пакета. Для начала использования нумерации получатель согласует с отправителем начальный порядковый номер. В этом документы рассматриваются способы смягчения требований к сохранению порядка доставки сообщений, а также методы генерации монотонно возрастающих порядковых номеров, обеспечивающие устойчивость к отказам и перезапуску отправителя.

Предложенный механизм не зависит от какого-либо криптографического алгоритма, но в документе описывается применение хеширования с ключами для аутентификации сообщений[2] с использованием HMAC-MD5 [7]. Как отмечено в работе [7], существуют более строгие алгоритмы хэширования (типа HMAC-SHA1); с тех случаях, где это целесообразно, реализациям следует делать такие алгоритмы доступными. Однако для общего случая в документе [7] показана адекватность HMAC-MD5 для заявленной цели, а также отмечено преимущество этого алгоритма в части производительности. В работе [7] также приведен исходный код и тестовые векторы для данного алгоритма, позволяющие проверить интероперабельность. HMAC-MD5 требуется в качестве базового механизма RSVP для обеспечения криптографической аутентификации, но опционально могут быть предложены и другие алгоритмы (см. раздел 6).

Контрольные суммы RSVP **могут** быть отключены (0) wпри включении в сообщение объекта INTEGRITY, поскольку цифровая подпись обеспечивает более строгую проверку целостности, нежели контрольная сумма.

[1] Resource ReSerVation Protocol.
[2] Keyed-Hashing for Message Authentication.

1.1. Используемые соглашения

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не следует** (SHALL NOT), **следует** (SHOULD), **не нужно** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе интерпретируются в соответствии с [8].

1.2. Почему не применяется стандартный заголовок IPSEC AH?

Возникает вопрос, почему при наличии стандартного механизма аутентификации IPSEC [3,5] предлагается применять иной механизм. Этот вопрос долго обсуждался в рабочей группе и от применения IPSEC отказались по описанным ниже причинам.

Защищенные связи в IPSEC базируются на адресах получателей. Не очевидно, что сообщения могут быть надежно определены для любого отправителя или получателя на базе защищенных связей, поскольку маршрутизаторы должны пересылать сообщения PATH и PATH TEAR с использованием того же адреса отправителя, который отправитель указал в SENDER TEMPLATE. Иначе служебный трафик RSVP может пойти по пути, отличному от пути доставки данных. Использование связей на основе адреса получателя или отправителя будет требовать создания новых защищенных связей между маршрутизаторами, через которые организуется резервирование.

Кроме того, отмечено, что отношения соседства между системами RSVP не ограничиваются простой смежностью на одном коммуникационном канале. Отношения RSVP могут организовываться через облака, не поддерживающие RSVP (определены в параграфе 2.9 работы [1]), которые могут быть невидимыми для передающей системы. На основании этих аргументов предложена стратегия управления ключами на базе парных связей между маршрутизаторами RSVP взамен применения IPSEC.

2. Структуры данных

2.1. Формат объекта INTEGRITY

Сообщение RSVP состоит из последовательности «объектов» представляющих собой структуры вида TLV[1]. Информация, требуемая для поэтапной (hop-by-hop) проверки целостности, передается в объектах INTEGRITY. Одинаковые объекты INTEGRITY применимы как для IPv4, так и для IPv6.

Формат объекта INTEGRITY показан ниже.

```
INTEGRITY: Class = 4, C-Type = 1

+-------------+-------------+-------------+-------------+
|    Flags    | 0 (резерв)  |                           |
+-------------+-------------+                           +
|                     Key Identifier                    |
+-------------+-------------+-------------+-------------+
|                     Sequence Number                   |
|                                                       |
+-------------+-------------+-------------+-------------+
|                                                       |
+                                                       +
|                                                       |
+                   Keyed Message Digest                |
|                                                       |
+                                                       +
|                                                       |
+-------------+-------------+-------------+-------------+
```

- **Flags** - 8-битовое поле, формат которого показан ниже.

```
                      Flags

  0   1   2   3   4   5   6   7
+---+---+---+---+---+---+---+---+
| H |                           |
| F |             0             |
+---+---+---+---+---+---+---+---+
```

  В настоящее время определен лишь флаг HF, а оставшиеся биты являются резервными и **должны** иметь значение 0.

- Бит 0 - флаг согласования HF[2] относится к согласованию механизма контроля целостности (параграф 4.3). Отправителям сообщений, желающим отвечать на сообщения согласования защиты целостности, **следует** устанавливать для флага значение 1, а отвергающим такое согласование **следует** устанавливать 0.

- **Key Identifier** - 48-битовое целое число без знака, которое **должно** быть уникальным для данного отправителя. Уникальные в локальном масштабе значения Key Identifier можно создавать на основе адресов (IP, MAC или LIH) передающего интерфейса и номера ключа. Комбинация Key Identifier с IP-адресом системы является уникальным идентификатором защищенной связи (параграф 2.2).

- **Sequence Number** - 64-битовый, монотонно возрастающий порядковый номер (целое число без знака).

  В качестве значений Sequence Number могут служить монотонно возрастающие последовательности из объектов INTEGRITY [каждого сообщения RSVP] с тегом, обеспечивающим уникальную связь с временем жизни ключа. Генерация порядковых номеров подробно рассматривается в разделе 3.

[1]Type-length-value - тип-размер-значение.
[2]Handshake Flag.

- **Keyed Message Digest** - подпись **должна** иметь размер, кратный 4 октетам. Для HMAC-MD5 размер подписи составляет 16 байтов.

2.2. Защищенная связь

Передающая и приемная системы поддерживают защищенную связь для каждого совместно используемого ключа аутентификации. Защищенная связь включает следующие параметры:

- алгоритм аутентификации и используемый режим этого алгоритма;
- ключ, применяемый с алгоритмом аутентификации;
- время жизни ключа;
- привязанный передающий интерфейс и другие критерии выбора защищенной связи [**требуется** на передающей системе];
- адрес отправителя на передающей системе [**требуется** на приемной системе];
- последний переданный порядковый номер, использованный с этим идентификатором ключа [**требуется** на приемной системе];
- список из последних N порядковых номеров, принятых с этим идентификатором ключа [**требуется** на приемной системе].

3. Генерация порядковых номеров

В этом параграфе описаны методы, которые могут быть выбраны для генерации порядковых номеров, используемых в объектах INTEGRITY сообщений RSVP. Как было отмечено выше, имеются два важных свойства, которые **должны** быть обеспечены в процедуре генерации. Первым свойством является уникальность (однократное применение) порядковых номеров в течение всего срока жизни текущего используемого ключа защиты целостности. Получатель может использовать это свойство для того, чтобы однозначно различать новые и повторные сообщения. Вторым свойством является монотонный рост номеров с использованием модуля 2^{64}. Это требуется для существенного снижения сохраняемого объема состояний, поскольку получателю достаточно сохранить значение с наибольшим порядковым номером для предотвращения атак с повторным использованием пакетов (replay). Поскольку начальный порядковый номер может быть произвольно большим, требуется применение операций с модулем для обеспечения возможности начала отсчета с нуля в течение жизни одного ключа. Это решение основано на нумерации в TCP [9].

Поле порядкового номера трактуется, как 64-битовое целое число без знака. Размер поля достаточно велик для того, чтобы нумерации хватило на весь срок жизни ключа. Например, если для ключа выбран срок жизни в 1 год, нумерации будет достаточно для передачи сообщений RSVP со средней скоростью около 585 Гига-сообщений в секунду. Использование 32-битовых порядковых номеров снизило бы этот предел до 136 сообщений в секунду.

Способность генерировать уникальные монотонно возрастающие порядковые номера при возникновении отказов и перезапусков подразумевает наличие некого стабильного хранилища (на устройстве или в сети). Ниже описаны три варианта генерации порядковых номеров.

3.1. Простые порядковые номера

Самым простым способом является генерация уникальных порядковых номеров на базе счетчика сообщений. При каждой передаче сообщения для данного ключа счетчик порядковых номеров увеличивается на 1. Текущее значение счетчика постоянно или периодически сохраняется в стабильном хранилище. После перезапуска значение счетчика восстанавливается из этого хранилища. Если сохранение счетчика выполнялось периодически при восстановлении следует добавить к значению счетчика число, заведомо превышающее возможное за время отказа увеличесние показаний счетчика. Это значение можно рассчитать, зная период сохранения.

3.2. Порядковые номера на основе системного времени

Большинство устройств вероятно не имеет возможности сохранять порядковые номера для каждого ключа в стабильной памяти. Более универсальным решением является создание порядковых номеров на базе стабильного хранилища системных часов. В большинстве компьютерных систем имеется модуль отсчета времени (часы - real time clock) со стабильным устройством хранения. Такие модули обычно включают тот или иной тип энергонезависимой памяти для сохранения информации о времени при сбоях питания.

В этой модели можно использовать основанную на NTP временную метку в качестве порядкового номера. Цикл полного отсчета для временных меток NTP составляет около 136 лет, что значительно превышает разумный срок жизни любого ключа. Кроме того, дискретность меток NTP достаточно хороша для того, чтобы можно было генерировать сообщение RSVP для данного ключа каждые 200 пикосекунд. Однако многие часы не поддерживают уровня дискретности меток NTP. В таких случаях младшие биты временной метки можно создавать с использованием счетчика сообщений, который сбрасывается при каждом «тике» системных часов. Например, если часы обеспечивают дискретность в 1 секунду, 32 младших бита порядкового номера можно задавать с использованием счетчика сообщений. В оставшиеся 32 бита порядкового номера следует помещать 32 старших[1] бита временной метки. Если предположить, что время восстановления после отказа занимает более одного «тика» системных часов, значение счетчика для младших битов порядкового номера можно смело сбрасывать после перезапуска системы.

3.3. Порядковые номера на основе сетевого времени

Если в устройстве нет ни стабильного хранилища, ни встроенных часов, можно получить значение текущего времени из сети по протоколу NTP. Поскольку время будет восстанавливаться при перезапуске системы, процедура генерации порядковых номеров не будет отличаться от описанной выше.

[1] В оригинале ошибочно сказано «32 least significant bits». См. http://www.rfc-editor.org/errata_search.php?eid=4313. *Прим. перев.*

4. Обработка сообщений

Реализациям **следует** разрешать спецификацию интерфейсов, которые будут защищены для отправки сообщений, их приема или в обоих случаях. Отправитель должен обеспечить во всех сообщениях RSVP, передаваемых через защищенные интерфейсы, наличие объекта INTEGRITY, созданного с использованием подходящего ключа Key. Получатели проверяют сообщения RSVP (за исключением типа Integrity Challenge, см. параграф 4.3), прибывающие на защищенный приемный интерфейс, на предмет наличия в них объекта INTEGRITY. Если объект INTEGRITY отсутствует, получатель отбрасывает сообщение.

Защитные связи являются односторонними (simplex) - ключи, которые передающая система использует для подписывания своих сообщений, могут отключаться от ключей, которые служат для подписи на приемной стороне. Следовательно, каждая связь ассоциируется с единственной передающей системой и (возможно) множеством принимающих систем.

Каждому отправителю **следует** иметь разные защищенные связи (и ключи) на защищенных передающих интерфейсах (или LIH). Хотя администраторы могут настроить все маршрутизаторы и хосты своей подсети (или сети) на использование одной защищенной связи, реализации **должны** предполагать, что каждый отправитель может передавать, используя отдельную защищенную связь на каждом защищенном интерфейсе. На стороне отправителя выбор защищенной связи происходит по интерфейсам, через которые передается сообщение. Этот выбор **может** учитывать дополнительные критерии типа адреса получателя (при передаче unicast-сообщения через широковещательную ЛВС с большим числом хостов) или идентификации пользователя на стороне отправителя или получателя [2]. Кроме того, все предполагаемые получатели сообщения должны участвовать в этой защищенной связи. Переключения (flap) маршрутов в сети без поддержки RSVP могут приводить к передаче сообщений для одного получателя через разные интерфейсы в разное время. В таких случаях получателю следует участвовать во всех защищенных связях, которые могут быть выбраны для возможных выходных интерфейсов.

Получатели выбирают ключи на основе Key Identifier и IP-адреса передающей стороны. Значение Key Identifier включается в объект INTEGRITY. Адрес передающей стороны может быть получен из объекта RSVP_HOP или (при отсутствии такого объекта, как в случае сообщений PathErr и ResvConf) из IP-адреса отправителя. Поскольку значение Key Identifier уникально для отправителя, этот метод обеспечивает уникальную идентификацию ключа.

Механизм защиты целостности слегка меняет правила обработки сообщений RSVP при включении объектов INTEGRITY в сообщения, передаваемые через защищенный интерфейс и при восприятии сообщений, принятых через защищенный интерфейс. Эти изменения подробно рассмотрены ниже.

4.1. Генерация сообщений

Сообщения RSVP, передаваемые через защищенный выходной интерфейс, создаются, как описано в [1], за исключением перечисленного ниже.

(1) Поле контрольной суммы RSVP устанавливается в 0. При необходимости контрольная сумма RSVP может быть рассчитана по завершении обработки объекта INTEGRITY.

(2) Объект INTEGRITY помещается в подходящее место и его позиция в сообщении запоминается для последующего использования.

(3) Передающий интерфейс и другие подходящие критерии (как указано выше) используются для определения ключа аутентификации Authentication Key и применяемого алгоритма хэширования.

(4) Неиспользуемые флаги и резервные поля объекта INTEGRITY **должны** быть установлены в 0. Флаг согласования HF[1] следует устанавливать в соответствии с правилами, описанными в параграфе 2.1.

(5) Номер передающего интерфейса **должен** обновляться для обеспечения уникальных, монотонно возрастающих порядковых номеров. Этот номер помещается в поле Sequence Number объекта INTEGRITY.

(6) В поле Keyed Message Digest помещается значение 0.

(7) Идентификатор ключа (Key Identifier) включается в объект INTEGRITY.

(8) Рассчитывается аутентификационная подпись сообщения с использованием Authentication Key в комбинации с алгоритмом хеширования. Для случая алгоритма HMAC-MD5 вычисление хэш-значения описано в [7].

(9) Полученная подпись записывается в поле Cryptographic Digest объекта INTEGRITY.

4.2. Прием сообщений

Когда сообщение принимается на защищенном входном интерфейсе и не относится к типу Integrity Challenge, выполняются следующие операции:

(1) Значение поля контрольной суммы RSVP сохраняется, а в поле помещается 0.

(2) Значение поля Cryptographic Digest объекта INTEGRITY сохраняется, а в поле помещается 0.

(3) Поле Key Identifier и адрес передающей системы служат для однозначного (уникального) определения Authentication Key и используемого алгоритма хэширования. Обработка этого пакета может быть задержана , если данная информация запрашивается в системе управления ключами (Приложение 1).

(4) Рассчитывается новая подпись с использованием указанного алгоритма и ключа Authentication Key.

(5) Если рассчитанная подпись не совпадает с полученной, сообщение отбрасывается без дальнейшей обработки.

[1] Handshake Flag.

(6) Если сообщение относится к типу Integrity Response, проверяется соответствие объекта CHALLENGE запросу. При наличии соответствия порядковый номер сохраняется в объекте INTEGRITY, как наибольший принятый номер.

В противном случае для всех остальных сообщений RSVP проверяются порядковые номера с целью предотвращения повторного использования пакетов и все сообщения с некорректными номерами игнорируются получателем. Если сообщение воспринимается, порядковый номер этого сообщения может быть обновлен на сохраненное максимальное принятое значение порядкового номера. Для восприятия каждого последующего сообщения оно должно будет иметь больший порядковый номер (по модулю 2^{64}). Это простое правило предотвращает атаки с повторным использованием сообщений, но его нужно изменить для обеспечения устойчивости к доставке с ограниченным нарушением порядка. Например, если несколько сообщений было передано разом (периодическое обновление от маршрутизатора или результат функции демонтажа), они могут прийти с нарушением порядка и номера не будут монотонно возрастать на приеме.

Реализациям **следует** позволять административную настройку конфигурации, которая определяет устойчивость получателей в нарушению порядка доставки. Простая модель позволит администраторам задать окно, размер которого соответствует наибольшему предполагаемому разупорядочению. Например, можно задать восприятие пакетов, для которых нарушение порядка нумерации не превосходит 32. если порядок доставки не нарушается, можно задать размер окна равным 1.

Получатель должен сохранять список всех порядковых номеров в окне допустимого разупорядочения. Принятый порядковый номер считается корректным, если (a) он больше максимального принятого до этого номера или (b) попадает в окно допустимого разупорядочения и отсутствует в списке принятых номеров. Сообщения с номерами меньше наименьшего значения в списке или уже отмеченными, как принятые, отбрасываются.

При получении на защищенном интерфейсе сообщения Integrity Challenge оно обрабатывается следующим образом:

(1) Формируется сообщение Integrity Response с использованием объекта Challenge, полученного в сообщении-вызове.

(2) Сообщение возвращается получателю по адресу отправителя в сообщении-вызове с использованием процедур, описанных выше в параграфе 4.1. Выбор Authentication Key и используемого алгоритма хэширования определяется идентификатором ключа, представленным в сообщении-вызове.

4.3. Согласование защиты целостности при перезапуске и инициализации получателя

Чтобы получить стартовый порядковый номер для Authentication Key, получатель **может** инициировать согласование защиты целостности с отправителем. Это согласование включает вызов получателя Challenge и отклик отправителя Response, согласование может быть инициировано в процессе перезапуска или отложено до момента приема сообщения, подписанного ключом.

Когда получатель решил инициировать согласование защиты целостности для конкретного ключа Authentication Key, он идентифицирует отправителя с использованием адреса передающей системы, настроенного для соответствующей защищенной связи. Получатель передает сообщение RSVP Integrity Challenge отправителю. Это сообщение содержит Key Identifier для идентификации ключа отправителя и **должно** иметь уникальной значение cookie отправителя, основанное на локальном секрете для предотвращения его подбора (см. параграф 2.5.3 работы [4]). Предполагается, что значение cookie будет представлять собой хэш MD5 для локального секрета и временной метки с целью обеспечения уникальности (см. раздел 9).

Сообщение RSVP Integrity Challenge относится к типу 11 и имеет формат:

```
<Integrity Challenge> ::= <Common Header> <CHALLENGE>
```

Объект CHALLENGE имеет формат:

```
CHALLENGE: Class = 64, C-Type = 1

+-------------+-------------+-------------+-------------+
|         0 (резерв)        |                           |
+-------------+-------------+                           +
|                     Key Identifier                    |
+-------------+-------------+-------------+-------------+
|                     Challenge Cookie                  |
|                                                       |
+-------------+-------------+-------------+-------------+
```

Отправитель воспринимает Integrity Challenge без проверки целостности. Он возвращает сообщение RSVP Integrity Response, содержащее исходный объект CHALLENGE. Сообщение также включает объект INTEGRITY, подписанный с помощью ключа, заданного Key Identifier из Integrity Challenge.

Сообщение RSVP Integrity Response относится к типу 12 и имеет формат:

```
<Integrity Response> ::= <Common Header> <INTEGRITY> <CHALLENGE>
```

Сообщение Integrity Response воспринимается получателем (challenger) только в том случае, когда оно возвращает объект CHALLENGE, соответствующий переданному в сообщении Integrity Challenge. Это предотвращает повторное использование старых сообщений Integrity Response. Если объекты соответствуют, получатель сохраняет номер Sequence Number из объекта INTEGRITY в качестве последнего номера, полученного с идентификатором ключа, включенным в объект CHALLENGE.

Если отклик не приходит в течение заданного периода, запрос (challenge) повторяется. После успешного завершения процедуры согласования защиты целостности получатель начинает воспринимать обычные сигнальные сообщения RSVP от данного отправителя и игнорирует другие сообщения Integrity Response.

Флаг согласования HF (Handshake Flag) используется для того, чтобы обеспечить реализациям гибкость без использования механизма согласования защиты целостности. Путем установки этого флага (1) отправители сообщений, реализующие согласование защиты целостности, отличают себя от остальных. Получателям **не следует** пытаться согласовывать защиту целостности с отправителями, чьи объекты INTEGRITY имеют HF = 0.

Согласование защиты целостности может требоваться не во всех средах. Типичным случаем применения защиты целостности RSVP является связь между маршрутизаторами партнерских доменов, которые очевидно будут обслуживать установившиеся потоки сообщений RSVP в результате агрегирования. Когда маршрутизатор перезапускается после аварии, в короткое время после загрузки могут быть получен поток корректных сообщений RSVP от отправителей из партнерского домена. Предполагая наличие в потоке корректных сообщений RSVP повторно используемых пакетов, в течение очень короткого периода до обработки корректного сообщения могут быть приняты пакеты replay-атаки. Корректное сообщение задаст наибольший порядковый номер, значение которого будет больше любого сохраненного до аварии номера, что предотвратит возможность дальнейшей атаки с повторным использованием пакетов.

С другой стороны, отказ от согласования защиты целостности может открыть продолжительное окно возможности организации replay-атаки, если после перезапуска от данного отправителя достаточно долго не будет пакетов. По этой причине **следует** принимать на административном уровне решение о согласовании защиты целостности для отправителей, которые отвечают на сообщения Integrity Challenge и восприятии сообщений от отправителей, отвергающих такие сообщения. Эти решения следует принимать с учетом конкретной сетевой среды.

5. Управление ключами

Вполне вероятно, что IETF будет определять стандартный протокол управления ключами. Весьма желательно использовать этот протокол для распространения ключей RSVP Authentication Key между взаимодействующими реализациями RSVP. Такой протокол обеспечит масштабируемость и существенное снижение нагрузки на администраторов. Идентификаторы ключей (Key Identifier) могут послужить связкой между RSVP и будущим протоколом управления ключами. Протоколы управления ключами имеют долгую историю недостатков, которые зачастую обнаруживались значительно позже публикации соответствующего протокола. Чтобы избежать необходимости изменения всех реализаций RSVP в результате обнаружения таких недостатков, встроенные средства управления ключами были преднамеренно исключены из данной спецификации.

5.1. Процедуры управления ключами

Каждый ключ имеет связанное с ним время жизни, которое записывается во всех системах (отправители и получатели), настроенных на работу с этим ключом. Концепция «времени жизни» ключа требует лишь понятного для систем представления значения начала (KeyStartValid) и завершения (KeyEndValid) срока действия ключа. Некоторые механизмы генерации ключей типа Kerberos и некоторых схем с открытыми ключами, могут напрямую создавать эфемерные ключи. В таких случаях время жизни ключа неявно определяется самим ключом.

В общем случае ключи не используются за пределами их времени жизни (см. параграф 5.3). Возможные механизмы управления временем жизни ключей включают протокол сетевого времени NTP[1] и системные часы.

Для поддержки безопасности целесообразно регулярно менять ключи RSVP Authentication Key. Следует обеспечить возможность смены RSVP Authentication Key без потери состояния RSVP или отказа от резервирования, а также без необходимости одномоментной смены всех ключей. Это требует от реализации RSVP поддержки хранилища и одновременного использования нескольких RSVP Authentication Key. Это позволит отправителям и получателям иметь множество активных ключей для данной защищенной связи.

Поскольку ключи совместно используются отправителем и (возможно) множеством получателей, существует время неопределенности в окрестностях момента смены ключа, когда некоторые системы будут продолжать пользоваться старым ключом, а другие уже переключатся на новый. Продолжительность этого периода неопределенности связана с синхронизацией системных часов. Администраторам следует задавать перекрытие между завершением срока действия старого ключа (KeyEndValid) и началом срока действия нового (KeyStartValid) не менее удвоенного интервала неопределенности. Это позволит отправителю поменять ключ в средней точке интервала перекрытия и быть уверенным, что все отправители уже восприняли новый ключ. В течение интервала перекрытия получатели должны быть готовы аутентифицировать сообщения с использованием обоих ключей.

В процессе смены ключа каждому получателю требуется согласовать с отправителем использование нового ключа. Как сказано выше, получатель может выбирать между инициированием согласования в процессе смены ключа или откладыванием процедур согласования до момента приема сообщения об использовании этого ключа.

5.2. Требования к управлению ключами

Требования к реализациям перечислены ниже.

- Весьма желательно обеспечить, чтобы гипотетическое нарушение безопасности одного из протоколов Internet не подвергало автоматически риску другие протоколы Internet. Ключи Authentication Key данной спецификации **не следует** хранить с использованием протоколов и алгоритмов, имеющих известные недостатки.
- Реализации **должны** поддерживать одновременное хранение и использование более одного ключа как на передающих, так и на приемных системах.
- Реализации **должны** связывать конкретное время жизни (KeyStartValid и KeyEndValid) с каждым ключом и соответствующим идентификатором Key Identifier.
- Реализации **должны** поддерживать распространение ключей вручную (т. е. привилегированный пользователь должен иметь возможность ввода ключа, времени его жизни и идентификатора со своей консоли). Время жизни может быть задано бесконечным.

[1] Network Time Protocol.

- Если поддерживается более одного алгоритма, реализации **должны** требовать, чтобы указывался алгоритм для каждого ключа вместе с вводом другой ключевой информации.
- Устаревшие ключи **могут** автоматически удаляться реализацией.
- **Должно** также поддерживаться удаление активных ключей вручную.
- Для упрощения использования хранилищам ключей **следует** обеспечивать сохранение информации после перезапуска системы (теплого или холодного).

5.3. Патологический случай

Возможны ситуации, когда срок действия последнего ключа для данной защищенной связи истекает. Когда это происходит, недопустим переход в неаутентифицированное состояние и нежелательно прерывание имеющихся резервирований. По этой причине системе следует отправить уведомление о завершении срока действия последнего ключа аутентификации администратору сети и трактовка имеющегося ключа, как ключа с неограниченным сроком жизни, пока время жизни ключа не будет расширено, не будет создан новый ключ или данный ключ не будет удален администратором.

6. Требования по соответствию

Для соответствия данной спецификации реализация **должна** поддерживать все аспекты спецификации. Во всех соответствующих этому документу реализациях **должен** поддерживаться алгоритм аутентификации HMAC-MD5, определенный в [7]. В соответствии со спецификацией реализации **могут** также поддерживать другие алгоритмы аутентификации типа NIST Secure Hash Algorithm (SHA). Все реализации **должны** поддерживать описанное выше распространение ключей вручную. Все реализации **должны** поддерживать продление срока жизни ключей, как описано в параграфе «5.1. Процедуры управления ключами».

Реализациям **следует** поддерживать стандартный протокол управления ключами для защищенного распространения ключей RSVP Authentication Key после того, как такой протокол будет стандартизован IETF.

7. Генерация Authentication Key с помощью Kerberos

Алгоритм Kerberos[10] **может** использоваться для генерации ключей RSVP Authentication, применяемых для генерации подписи в объектах Integrity, которые передаются отправителем RSVP получателям. Генерация ключей Kerberos избавляет от необходимости применения ключей, разделяемых отправителями и получателями RSVP (хосты и маршрутизаторы). Kerberos позволяет использовать связь между защищаемыми сторонами (отправитель и получатели RSVP) через доверенную третью сторону, когда центр распространения ключей Kerberos (KDC) организует эфемерные сеансовые ключи впоследствии разделяемые между отправителем и получателями RSVP. Для группового случая все получатели группового сообщения RSVP **должны** разделять общий ключ с KDC (т. е., получатели эффективно являются защищаемой стороной относительно Kerberos).

Информация о ключе (Key information), определенная отправителем, **может** задавать использование Kerberos вместо заданных в конфигурации разделяемых ключей в качестве механизма обмена ключами между отправителем и получателем. Kerberos-идентификация получателя организуется в процессе настройки конфигурации интерфейса отправителя или с помощью иного механизма. При генерации первого сообщения RSVP для конкретного идентификатора ключа отправитель запрашивает «квитанцию» от сервиса Kerberos и получет эфемерный сеансовый ключ и квитанцию Kerberos от KDC. Отправитель инкапсулирует квитанцию и свою идентификацию в объект Identity Policy [2]. Объект Policy Object отправитель включает в сообщение RSVP. Сеансовый ключ используется отправителем в качестве ключа RSVP Authentication на этапе (3) (параграф 4.1) и сохраняется как информация о ключе, связанная в идентификатором ключа.

При получении сообщения RSVP приемная сторона извлекает Kerberos Ticket из объекта Identity Policy, дешифрует квитанцию и извлекает из нее сеансовый ключ. Этот ключ совпадает с применяемым отправителем ключом и используется на этапе (3) (параграф 4.2). Получатель сохраняет ключ для использования при обработке последующих сообщений RSVP.

Квитанции Kerberos действительны в течение ограниченного времени и отправителю **недопустимо** использвать квитанции с истекшим временем жизни. Отправитель **должен** запросить и использовать новую квитанцию для получателя до завершения срока действия предыдущей.

7.1. Оптимизация при использовании аутентификации на базе Kerberos

Квитанции Kerberos сравнительно велики (> 500 байтов) и их не требуется передавать в каждом сообщении RSVP. Эфемерный сеансовый ключ можно кэшировать на обеих сторонах и использовать кэш в течение срока жизни квитанции Kerberos. В этом случае отправитель должен включить квитанцию Kerberos только в первое генерируемое сообщение. Последующие сообщения RSVP могут использовать идентификатор ключа для получения кэшированного значения (и, возможно, дополнительных подтверждений идентичности) вместо передачи квитанций получателю в каждом сообщении RSVP.

У получателя может не оказаться кэшированного ключа с его Key Identifier по причине перезагрузки или изменения маршрута. Если политика получателя задает использование ключей Kerberos для проверки целостности, получатель может передать отправителю сообщение Integrity Challenge. При получении такого сообщения отправитель должен передать объект Identity, включающий квитанцию Kerberos, в сообщении Integrity Response, что позволит получателю получить и сохранить сеансовый ключ из квитанции Kerberos для последующих проверок целостности.

8. Благодарности

Этот документ непосредственно базируется на похожей работе, выполненной для протоколов OSPF и RIP Version II совместно Ran Atkinson и Fred Baker. Существенная работа по редактированию была выполнена Bob Braden, что улучшило ясность изложения. Были получены важные комментарии от Steve Bellovin, хорошо знающего эту тему. Matt Crawford и Dan Harkins помогли в пересмотре документа.

9. Литература

[1] Braden, R., Zhang, L., Berson, S., Herzog, S. and S. Jamin, "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification", RFC 2205, September 1997.

[2] Yadav, S., et al., "Identity Representation for RSVP", RFC 2752[1], January 2000.

[3] Atkinson, R. and S. Kent, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[4] Maughan, D., Schertler, M., Schneider, M. and J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, November 1998.

[5] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402, November 1998.

[6] Kent, S. and R. Atkinson, "IP Encapsulating Security Payload (ESP)", RFC 2406, November 1998.

[7] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, March 1996.

[8] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[9] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[10] Kohl, J. and C. Neuman, "The Kerberos Network Authentication Service (V5)", RFC 1510[2], September 1993.

10. Вопросы безопасности

Этот документ в целом описывает и задает механизм аутентификации для протокола RSVP, который должен обеспечить защиту от активных и пассивных атак.

Качество обеспечиваемой этим механизмом защиты зависит от стойкости реализованных алгоритмов аутентификации, качества применяемых ключей и корректности реализации механизма защиты во всех вовлеченных компонентах и узлах RSVP. Этот механизм зависит также от надежности хранения ключей RSVP Authentication Key всеми участниками. Если любое из этих допущений не выполняется или не обеспечивает достаточной защиты, для пользователей данного механизма реальная защита не будет обеспечиваться.

Целостность сообщений Integrity Response проверяется, но целостность Integrity Challenge не контролируется. Это сделано осознанно с целью предотвратить возникновение ситуаций, когда оба маршрутизатора-партнера не имеют начальный порядковых номеров для ключа другой стороны. В таком случае оба маршрутизатора продолжали бы передачу сообщений Integrity Challenge, которые другая стороны просто отбрасывала бы. Более того, проверка целостности только для одного типа сообщений избавляет от необходимости наличия защищенной связи в обратном направлении.

Однако это позволяет атакующему генерировать обманные запросы на согласование с неким challenge cookie. Атакующий может сохранить полученные отклики и попытаться использовать их для атаки на восстанавливаемого получателя. При определенном везении атакующий может угадать значение challenge cookie, используемые получателем в процессе восстановления. Тогда переданный атакующим обманный отклик будет воспринят, поскольку он имеет корректную подпись и меньший, чем у отправителя порядковый номер (поскольку сообщение будет более старым). Таким образом открывается возможность атаки на получателя с повторным использованием пакетов. Однако использование такой возможности представляется весьма проблематичным. Требуется не только заранее угадать значение challenge cookie (основано на локальном секрете), но и иметь возможность замаскироваться под получателя для генерации Integrity Challenge с корректным адресом IP и не оказаться пойманным.

Данный механизм не обеспечивает защиты конфиденциальности. Если такая защита нужна, эффективным решением может быть IPSEC ESP [6], хотя ему полностью присущи недостатки IPSEC Authentication и, следовательно, применение возможно только в специфических средах. Не обеспечивается и защиты от анализа трафика. Для обеспечения такой защиты могут применяться механизмы шифрования данных на уровне канала.

11. Адреса авторов

Fred Baker

Cisco Systems

519 Lado Drive

Santa Barbara, CA 93111

Phone: (408) 526-4257

EMail: fred@cisco.com

Bob Lindell

USC Information Sciences Institute

4676 Admiralty Way

Marina del Rey, CA 90292

Phone: (310) 822-1511

EMail: lindell@ISI.EDU

[1] Этот документ заменен RFC 3182. *Прим. перев.*
[2] Этот документ заменен RFC 4120 и RFC 6649. *Прим. перев.*

Mohit Talwar

Microsoft Corporation

One Microsoft Way

Redmond, WA 98052

Phone: +1 425 705 3131

EMail: mohitt@microsoft.com

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru

12. Приложение 1 - Интерфейс управления ключами

В этом приложении описан базовый интерфейс управления ключами (Key Management). Описание приведено на абстрактном уровне и реализациям могут потребоваться некоторые изменения при создании реального интерфейса.

При запуске RSVP протокол будет использовать этот интерфейс для получения текущего набора актуальных ключей для передаваемых и принимаемых сообщений. В процессе работы RSVP может запрашивать конкретные ключи, указанные параметрами Key Identifier и Source Address, обнаруживать вновь созданные ключи и информировать об удаленных ключах. Интерфейс обеспечивает возможность опроса и асинхронных вызовов.

12.1. Структура данных

Информация о ключах возвращается в структуре данных KeyInfo.

```
KeyInfo {
        Тип ключа (Send или Receive)
        KeyIdentifier
        Ключ
        Тип и режим алгоритма аутентификации
        KeyStartValid
        KeyEndValid
        Статус (Active или Deleted)
        Выходной интерфейс (только для Send)
        Другие критерии выбора исходящей защищенной связи
                (только для Send, не обязательно)
        Адрес передающей системы (только для Receive)
}
```

12.2. Таблица используемых по умолчанию ключей

Эта функция возвращает список структур KeyInfo, соответствующих всем ключам, которые указаны для передачи и приема сообщений RSVP и имеют активный статус (Active). Функция обычно вызывается на начальном этапе исполнения процесса, но число ее вызовов ничем не ограничено.

```
KM_DefaultKeyTable() -> KeyInfoList
```

12.3. Запрос неизвестных ключей приема

При получении сообщения с неизвестной парой (Key Identifier, Sending System Address) RSVP может использовать эту функцию для запроса подходящего ключа у системы управления ключами (Key Management System). Функция возвращает статус элемента (если он найден), который должен иметь значение Active.

```
KM_GetRecvKey( INTEGRITY Object, SrcAddress ) -> KeyInfo
```

12.4. Опрос на предмет обновлений

Эта функция возвращает список структур KeyInfo, соответствующих всем нарастающим изменениям, которые могли быть внесены в используемую по умолчанию таблицы ключей с момента предыдущего вызова KM_KeyTablePoll, KM_DefaultKeyTable или KM_GetRecvKey. Для некоторых возвращенных элементов может быть указан статус Deleted.

```
KM_KeyTablePoll() -> KeyInfoList
```

12.5. Интерфейс асинхронных Upcall-вызовов

Вместо повторяющихся вызовов KM_KeyTablePoll() реализация может воспользоваться асинхронной моделью на основе событий. Эта функция регистрирует интерес к смене ключа для заданного Key Identifier или для всех ключей при отсутствии Key Identifier. Функция будет вызываться при каждой смене ключей.

```
KM_KeyUpdate( Function [, KeyIdentifier ] )
```

где upcall-функция параметризуется следующим образом:

```
Function( KeyInfo )
```

13. Полное заявление авторских прав

Подтверждение

Финансирование функций RFC Editor обеспечено Internet Society.