Network Working Group
Request for Comments: 2205
Category: Standards Track

R. Braden, Ed.
ISI
L. Zhang
UCLA
S. Berson
ISI
S. Herzog
IBM Research
S. Jamin
Univ. of Michigan
September 1997

Функциональная спецификация протокола резервирования ресурсов (RSVP) версии 1

Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification

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.

Тезисы

В этом документе описана версия 1 протокола RSVP, обеспечивающего организацию путей с резервированием ресурсов для интегрированных служб Internet. RSVP обеспечивает инициируемое отправителем резервирование ресурсов для групповых и индивидуальных потоков данных с высоким уровнем надежности и масштабирования.

Оглавление

Изменения

Данная версия документа содержит ряд незначительных отличий от версии ID14, перечисленных ниже.

- Для большей ясности каждый тип сообщений определен раздельно в параграфе 3.1.
- Добавлены более точные и полные правила восприятия сообщений Path для индивидуальных и групповых адресатов (параграф 3.1.3).
- Добавлены более точные и полные правила обработки и пересылки сообщений PathTear (параграф 3.1.5).
- Добавлено примечание об игнорировании объекта SCOPE в сообщениях ResvTear (параграф 3.1.6).
- Добавлено примечание об игнорировании объектов SENDER_TSPEC и ADSPEC в сообщениях PathTear (параграф 3.1.5).
- Удален отмененный код ошибки Ambiguous Filter Spec (09) и уточнено имя ошибки с кодом 08 (Приложение B).
- В базовый интерфейс управления трафиком добавлен параметр Adspec при вызовах AddFlow и ModFlow (3.11.2) для аккомодации узла, обновляющего slack term (S) для гарантированного сервиса (Guaranteed).
- Добавлен субтип ошибки для Adspec (Приложение B).
- Расширено разъяснение обработки объектов CONFIRM (параграф 3.1.4).
- Разъяснены правила пересылки объектов с неизвестным типом класса.
- Во введении и параграфе 3.11.2 расширено обсуждение связи RSVP с канальным уровнем (параграф 3.10).
- Параграф 2.7, посвященный политике и безопасности, разделен на два параграфа с расширением обсуждения некоторых вопросов.
- Внесены незначительные редакторские правки.

1. Введение

В этом документе определен протокол RSVP, предназначенный для организации путей с резервированием ресурсов для интегрированных служб [RSVP93, RFC 1633]. Протокол RSVP используется хостами для запроса специфических параметров обслуживания в сети отдельных потоков данных. RSVP также используется маршрутизаторами для выполнения запросов к качеству обслуживания (QoS[1]) на всех узлах по пути доставки потока и поддержки состояния, позволяющего обеспечить запрошенные услуги. Запросы RSVP приводят к резервированию ресурсов на каждом узле по пути доставки данных.

RSVP запрашивает ресурсы для однонаправленных (симплексных) потоков. Следовательно, RSVP логически различает отправителя и получателя, хотя один и тот же прикладной процесс может быть одновременно отправителем и получателем. RSVP работает на базе протоколов IPv4 и IPv6, занимая место транспортного протокола в стеке протоколов. Однако RSVP не является транспортным протоколом, а служит скорее протоколом управления Internet, подобно ICMP, IGMP или протоколам маршрутизации. Подобно реализациям протоколов маршрутизации и управления, реализации RSVP обычно работают в фоновом режиме (не на пути пересылки), как показано на рисунке 1.

RSVP сам по себе не является протоколом маршрутизации и рассчитан на работу с существующими и разрабатываемыми протоколами индивидуальной и групповой маршрутизации. Процесс RSVP получает маршруты из локальных баз данных. При групповой маршрутизации хост отправляет сообщения IGMP для присоединения к группе и после этого передает сообщения RSVP для резервирования ресурсов на путях доставки для этой группы. Протоколы маршрутизации определяют, куда следует пересылать пакеты, а RSVP лишь рассматривает QoS для пакетов, рассылаемых по выбранным системой маршрутизации путям.

Для эффективной работы с большими группами, динамическим включением в группы, а также с учетом разнородности требований получателей RSVP делает последних ответственными за запрос соответствующих параметров QoS [RSVP93]. Запрос QoS от приложения-получателя передается локальному процессу RSVP. После этого протокол RSVP передает запрос всем узлам (хостам и маршрутизаторам) на пути (путях) передачи данных от их источника (источников), но лишь после того, как маршрутизатор на пути данных к получателю присоединится к дереву группы. В

[1]Quality-of-service.

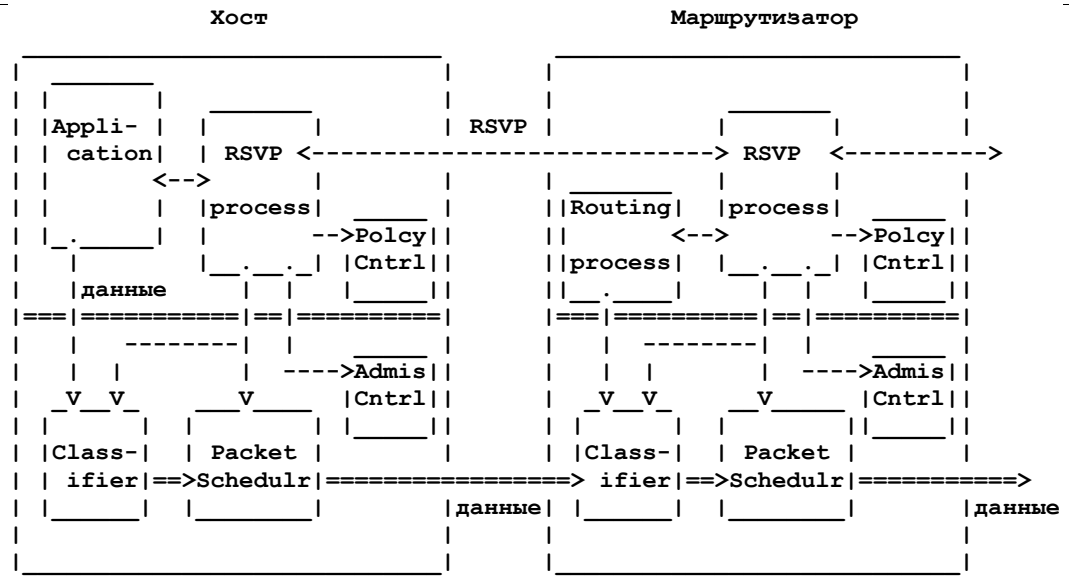


Рисунок 1. RSVP на хостах и маршрутизаторах

результате дополнительные издержки RSVP растут в зависимости от числа получателей логарифмически, а не линейно.

Качество обслуживания для отдельного потока данных реализуется с помощью механизмов, называемых «управлением трафиком» (traffic control). Эти механизмы включают (1) классификацию пакетов, (2) входной контроль и (3) «планировщик пакетов» или механизм канального уровня для решения вопроса о времени пересылки конкретного пакета. Классификатор пакетов определяет класс QoS (и, возможно, маршрут) для каждого пакета. Для каждого выходного интерфейса планировщик пакетов или механизм канального уровня обеспечивает обещанный уровень QoS. Управление трафиком реализует модель обслуживания QoS, определенную рабочей группой Integrated Services.

При организации резервирования передается запрос RSVP QoS двум локальным модулям принятия решений - «входной контроль» (admission control) и «контроль правил» (policy control). Входной контроль определяет наличие у хоста ресурсов, достаточных для поддержки запрошенного QoS. Контроль правил определяет наличие у пользователя прав, достаточных для организации резервирования. Если обе проверки дают положительный результат, устанавливаются параметры классификатора пакетов и интерфейса канального уровня (например, в планировщике пакетов) для получения желаемого QoS. Если любая из проверок дает отрицательный результат, программа RSVP возвращает прикладному процессу сообщение об ошибке.

Механизмы протокола RSVP обеспечивают общие методы создания и поддержки состояния резервирования ресурсов на пути через многосвязную сеть для индивидуального и группового трафика. Сам протокол RSVP использует параметры QoS и контроля политики, передавая их соответствующим модулям управления политикой и трафиком для интерпретации. Структура и содержимое параметров QoS описаны в спецификациях, разработанных группой Integrated Services [RFC 2210]. Структура и содержимое параметров управления политикой находятся в разработке.

Поскольку принадлежность к большим multicast-группам и топология дерева групповой рассылки с течением времени меняются, протокол RSVP предполагает, что состояния RSVP и управления трафиком задаются и изменяются хостами и маршрутизаторами в инкрементальном режиме. Для этого RSVP организует soft-состояние, когда протокол периодически передает обновления для поддержки состояния вдоль обратного пути (путей). При отсутствии таких сообщений состояние считается устаревшим и удаляется.

Протокол RSVP использует перечисленные ниже атрибуты.

- RSVP выполняет резервирование для приложений unicast и multicast (многие со многими), динамически адаптируясь к изменению членства в группах маршрутах;
- протокол RSVP является симплексным, т. е. резервирование выполняется для однонаправленных потоков;
- RSVP ориентирован на получателей, т. е. инициирование и поддержка выделения ресурсов осуществляется получателями данных;
- RSVP поддерживает soft-состояние на хостах и маршрутизаторах, что обеспечивает динамический учет изменений в составе групп и маршрутах;
- RSVP не является протоколом маршрутизации и зависит от существующих и будущих протоколов этого типа;
- RSVP доставляет и поддерживает параметры управления трафиком и политикой без их обработки;
- RSVP поддерживает несколько моделей (стилей) резервирования для разных приложений;
- RSVP обеспечивает прозрачную работу через маршрутизаторы, не поддерживающие этот протокол;
- RSVP может работать на базе протоколов IPv4 и IPv6.

Дополнительное обсуждение задач и общего устройства RSVP можно найти в документах [RSVP93] и [RFC 1633].

В оставшейся части этого раздела описаны услуги RSVP по резервированию. В разделе 2 приведен обзор механизмов протокола RSVP. Раздел 3 дает функциональное описание RSVP, а в разделе 4 описаны правила обработки сообщений. Приложение A определяет типизованные объекты данных протокола RSVP. В Приложении B определены ошибки и их коды. Приложение C посвящено инкапсуляции сообщений RSVP в дейтаграммы UDP для хостов, операционные системы которых не могут корректно работать с неразобранными (raw) сетевыми данными.

1.1 Потоки данных

RSVP определяет «сессию», как поток данных с определенным адресатом и протоколом транспортного уровня. RSVP трактует каждую сессию независимо и в документе зачастую опускаются подразумеваемые слова «для той же сессии».

Сессия RSVP определяется тремя параметрами: DestAddress, ProtocolId [, DstPort]. DestAddress указывает IP-адрес получателя пакетов данных и может быть индивидуальным или групповым. Параметр ProtocolId указывает идентификатор протокола IP. Необязательный параметр DstPort представляет собой «обобщенный порт получателя» - некоторую демультиплексируемую далее «точку» транспортного или прикладного уровня. Значение DstPort может определяться номером порта UDP/TCP на стороне адресата, его эквивалентом в других протоколах транспортного уровня или некой информацией прикладного уровня.

Хотя при разработке RSVP ставились задачи простоты расширения, документированный здесь базовый протокол поддерживает в качестве обобщенных портов лишь порты UDP/TCP. Отметим, что значительной потребности включать DstPort при групповом адресе DestAddress не возникает, поскольку разные сессии всегда будут отличаться по групповым адресам получателей. Однако значение DstPort требуется для того, чтобы можно было организовать более одной сессии с хостом по индивидуальному адресу.

На рисунке 2 показан поток пакетов данных в одной сессии RSVP в предположении групповой рассылки. Стрелки показывают данные от отправителей S1 и S2 к получателям R1, R2 и R3, а облако представляет сеть распределения, созданную групповой маршрутизацией. При групповом распространении копия каждого пакета от Si передается каждому получателю Rj; при индивидуальной адресации получатель один (R). Каждый отправитель Si может работать на отдельном хосте Internet или один хост может поддерживать множество отправителей с разными обобщенными портами отправителя.

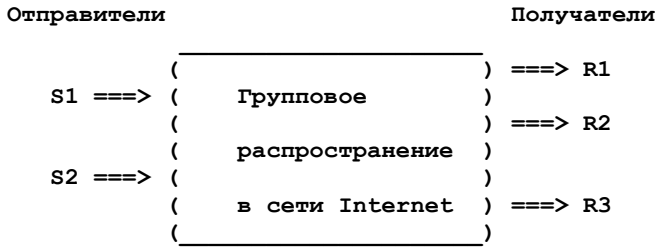


Рисунок 2 Сеанс группового распространения данных.

Для индивидуальной передачи будет один хост-получатель, но может быть множество отправителей; RSVP позволяет организовать резервирование для передачи «многие одному».

1.2 Модель резервирования

Элементарный запрос резервирования RSVP включает спецификацию потока (flowspec) вместе со спецификацией фильтров (filter spec); вместе их называют дескриптором потока. Спецификация потока указывает желаемые параметры QoS. Спецификация фильтров совместно со спецификацией сессии определяет набор пакетов данных потока, к которым относятся параметры QoS, заданные спецификацией потока. Спецификация потока служит для установки на узле параметров планировщика пакетов или иного механизма канального уровня, а спецификация фильтров служит для установки параметров классификатора пакетов. Пакеты данных, адресованные конкретной сессии, но не соответствующие любому из заданных спецификацией фильтров для данной сесии, будут обслуживаться по мере возможности (best-effort).

Спецификация потока в запросе на резервирование в общем случае включает класс обслуживания, а также два набора числовых параметров: (1) Rspec (R от reserve), определяющий желаемый уровень QoS и (2) Tspec (T от traffic), описывающий поток данных. Форматы и содержимое Tspec и Rspecs определяются моделями интегрированного обслуживания [RFC 2210] и в общем случае не обрабатываются RSVP.

Точный формат спецификации фильтров зависит от используемого протокола (IPv4 или IPv6) и описан в Приложении A. В более общей модели [RSVP93] фильтры позволяют выбирать произвольные подмножества пакетов данной сессии. Такие подмножества можно определить в терминах отправителей (IP-адрес и обобщенный порт), протоколов вышележащего уровня или, в общем случае, значениях любых полей любого заголовка в пакете. Например, спецификация фильтра может выбирать разные субпотоки видео-потока с иерархическим кодированием путем выбора полей заголовка прикладного уровня. Для простоты (и минимизации использования уровней) базовая спецификация фильтров, определяемая в данной спецификации RSVP ограничивается фильтрацией по IP-адресам отправителей и (необязательно) номерам портов UDP/TCP в SrcPort.

Поскольку номера портов UDP/TCP используются для классификации пакетов, каждый маршрутизатор должен быть способен проверять эти поля. Эта необходимость может вызывать три проблемы.

1. Требуется избегать фрагментирования пакетов данных IP, для которых желательно резервирование.

   В [RFC 2210] задана процедура для приложений, использующих RSVP, по расчету минимального значения MTU в дереве путей групповой доставки с возвратом результата отправителю.

2. IPv6 добавляет переменное число заголовков уровня Internet с переменным размером до заголовка транспортного уровня, что усложняет и удорожает классификацию пакетов для QoS.

   Для эффективной классификации пакетов IPv6 можно использовать поле Flow Label в заголовке IPv6. Детали этого процесса будут представлены позднее.

3. IPSec для IPv4 или IPv6 может целиком шифровать заголовки транспортного уровня, скрывая номера портов от промежуточных маршрутизаторов.

Другая опция управляет выбором отправителей — это может быть «явный» список или шаблон, неявно задающий всех отправителей данной сессии. При резервировании с явным указанием отправителей каждой спецификации фильтра должен соответствовать единственный отправитель, а при использовании шаблона фильтр не нужен.

Определенные в настоящее время стили показаны на рисунке 3.Небольшое расширение RSVP для IPSec на базе IPv4 и IPv6 описано в документе [RFC 2207].

Сообщения RSVP с запросами на резервирование исходят от получателей и передаются в восходящем[1] направлении к отправителю (отправителям).

На каждом промежуточном узле запрос на резервирование инициирует два описанных ниже действия.

1. Резервирование на канале.

   Процесс RSVP передает запрос на входной контроль и контроль правил. Если какая-либо из проверок не проходит, резервирование отвергается и процесс RSVP возвращает сообщение об ошибке отправившему запрос получателю (получателям). После успешной проверки узел организует классификатор пакетов для выбора пакетов данных, указанных спецификацией фильтров и взаимодействует с канальным уровнем для получения параметров QoS, заданных спецификацией потока.

   Конкретные правила выполнения запросов RSVP QoS зависят от технологии канального уровня, используемой на каждом интерфейсе. Рабочая группа ISSLL готовит спецификации для отображения запросов резервирования на популярные технологии канального уровня. Например, для простой выделенной линии желаемые параметры QoS могут быть получены от планировщика пакетов на канальном уровне. Если технология канального уровня поддерживает свои механизмы управления QoS, протокол RSVP должен согласовать с канальным уровнем получение запрошенного QoS. Отметим, что действия по управлению QoS выполняются на входе данных в среду канального уровня, т. е. на восходящей стороне физического или логического канала, хотя запросы RSVP на резервирование приходят с нисходящего направления.

2. Пересылка запроса в восходящем направлении.

   Запрос на резервирование распространяется в восходящем направлении к соответствующим источникам данных. Набор хостов-отправителей, для которых распространяется запрос на резервирование называют областью (scope) данного запроса.

   Запрос на резервирование, пересылаемый узлом в восходящем направлении, может отличаться от полученного этим узлом запроса по двум причинам. Механизмы управления трафиком могут менять спецификацию потока на этапах пересылки. И, более важно, запросы на резервирование от множества нисходящих узлов разных ветвей дерева групповой передачи к одному отправителю (набору отправителей) должны объединяться по мере перемещения запросов в восходящем направлении.

Когда получатель инициирует запрос на резервирование, он может также запросить подтверждающее сообщение, которое будет показывать, что его запрос (возможно) будет выполнен в сети. Распространение запроса в восходящем направлении по дереву групповой рассылки продолжается, пока не будет достигнута точка, где существующее резервирование совпадает или превосходит запрашиваемое. В этой точке прибывающие запросы объединяются с имеющимся резервированием и необходимость дальнейшей пересылки запросов отпадает. В этом случае узел может передать в нисходящем направлении подтверждение отправителю запроса. Отметим, что такое подтверждение говорит лишь о высокой вероятности резервирования запрошенных ресурсов, но не гарантирует его. Более подробное рассмотрение этого вопроса приведено в параграфе 2.6.

Базовая модель резервирования RSVP является однопроходной — получатель отправляет запрос на резервирование в восходящем направлении и каждый узел на пути запроса может принять или отвергнуть его. Такая схема не обеспечивает получателям простого способа сквозной оценки качества обслуживания. Поэтому RSVP поддерживает расширенный однопроходный сервис OPWA[2] [OPWA95]. При использовании OPWA пакеты управления RSVP передаются в нисходящем направлении, следуя по путям передачи данных, для сбора информации, позволяющей оценить сквозной уровень QoS. Результаты этого (анонсы) доставляются протоколом RSVP хостам-получателям и, возможно, принимающим данные приложениям. Анонсы могут использоваться получателем для создания или динамического изменения запросов на резервирование.

1.3 Стили резервирования

Запросы на резервирование включают набор опций, называемый «стилем» (style) резервирования.

Выбор отправителя	Резервирование: Раздельное	Общее
Явный	Фиксированный фильтр (FF)	Разделяемый с явным заданием (SE)
Шаблон	(не определен)	Шаблонный фильтр (WF)

Рисунок 3 Атрибуты и стили резервирования

Одна из опций резервирования относится к трактовке резервов для разных отправителей в рамках одной сессии — организовать «отдельное» резервирование для каждого отправителя или использовать одно резервирование, разделяемое между потоками от всех отправителей.

Атрибуты и стили запросов на резервирование показаны на рисунке 3.

- Стиль с шаблонным фильтром (Wildcard-Filter - WF)

[1] В этом документе термины «восходящий» и «нисходящий», «предыдущий» и «последующий» интервал, «входной» и «выходной» интерфейс трактуются по отношению к основному потоку данных.
[2] One Pass With Advertising — один проход с анонсированием.

Стиль WF предполагает совместно используемое резервирование с заданными шаблоном отправителями. Таким образом, для этого стиля создается одно резервируемое для передачи потоков данных от всех выше расположенных отправителей. Это резервирование можно рассматривать, как «трубу», размер которой определяется самым крупным запросом, независимо от числа использующих данное резервирование отправителей. Резервирование в стиле WF распространяется в восходящем направлении в сторону всех хостов-отправителей и автоматически распространяется на новых отправителей при их добавлении.

Символически запрос резервирования в стиле WF представляется, как

WF(* {Q})
звездочка представляет шаблон для выбора всех отправителей, а Q указывает спецификацию потока.

- Стиль с фиксированным фильтром (Fixed-Filter - FF)

  Стиль FF предполагает раздельное резервирование с явным выбором отпраителей. Таким образом, элементарный запрос резервирования в стиле FF создает резервирование для пакетов данных от конкретного отправителя, не разделяемое с пакетами от других отправителей в той же сессии.

  Символически элементарный запрос резервирования в стиле FF можно представить, как

  FF(S{Q})
  S задает отправителя, а Q — соответствующий поток данных; эта пара формирует дескриптор потока. RSVP поддерживает множество одновременных элементарных запросов в стиле FF с использованием списка дескрипторов вида

  FF(S1{Q1}, S2{Q2}, ...)
  Общее резервирование для данной сессии будет «суммой» Q1, Q2, ... для всех отправителей.

- Разделяемый с явным заданием (Shared Explicit - SE)

  Стиль SE предполагает разделяемое резервирование с явным заданием отправителя. Таким образом, при резервировании в стиле SE создается один используемый совместно резерв для указанных вышележащих отправителей. В отличие от WF стиль SE позволяет получателю явно указать набор отправителей.

  Будем представлять запрос в стиле SE для спецификации потока Q и набора отправителей S1, S2, …, как

  SE((S1,S2,...){Q})

Разделяемое резервирование (стили WF и SE) подходит для multicast-приложений, где имеется множество отправителей, но одновременная передача от них маловероятна. Примером таких приложений может служить пакетная передача звука — поскольку одновременно может говорить ограниченное число людей, каждый получатель может ввести запрос на резервирование в стиле WF или SE с удвоенной по отношению к одному говорящему полосой (это позволит передать два звуковых канала одновременно). С другой стороны, стиль FF, создающий раздельные резервы для потоков от разных отправителей, хорошо подходит для видео-приложений.

Правила RSVP не позволяют объединять разделяемые резервы с раздельными, поскольку эти режимы принципиально не совместимы. Запрещается также объединение резервов с явным и шаблонным выбором отправителей, поскольку это может приводить к появлению незапрошенного сервиса у получателей, задавших явный выбор отправителя. В результате этих ограничений стили WF, SE и FF являются взаимно не совместимыми.

Представляется возможным имитировать эффект резервирования WF при использовании стиля SE. Когда приложение запрашивает резервирование WF процесс RSVP на хосте-получателе может использовать локальное состояние для создания эквивалентного резервирования SE, в котором явно перечислены все отправителя. Однако резервирование SE заставляет классификаторы пакетов на каждом узле явно выбирать каждого отправителя из списка, тогда как WF позволяет классификатору использовать шаблон для адресов и портов отправителей. При большом списке отправителей резервирование в стиле WF может, следовательно, вносить существенно меньшие издержки, нежели его эквивалент в стиле SE. По этой причине в протокол включены оба стиля - SE и WF.

В будущем возможно определение других опций и стилей резервирования.

1.4 Примеры стилей

В этом параграфе представлены примеры каждого стиля резервирования и рассмотрены эффекты слияния.

На рисунке 4 показан маршрутизатор с двумя входными интерфейсами (a) и (b), через которые поток данных будет приходить, и двумя выходными (c) и (d), через которые данные будут пересылаться. Такая топология предполагается в приведенных далее примерах. Имеется три отправителя — пакеты от S1 (S2 и S3) приходят с предыдущего интервала через интерфейс (a) (и, соответственно, (b)). Имеется также три получателя — пакеты для R1 (R2 и R3) маршрутизируются через выходной интерфейс (c) (и, соответственно, (d)). Дополнительно предполагается, что выходной интерфейс (d) подключен к широковещательной ЛВС), т. е. репликация выполняется в сети), а для пересылки к получателям R2 и R3 используются разные маршрутизаторы следующего интервала (не показаны).

На узле, показанном на рисунке 4, требуется также задать групповые маршруты. Предположим, что первый пакет от источника Si на рисунке 4 маршрутизируется на оба выходных интерфейса. С учетом этого допущения на рисунках 5, 6 и 7 показаны примеры для стилей WF, FF и SE, соответственно.

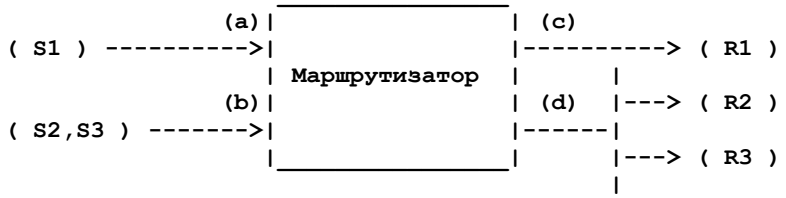


Рисунок 4 Конфигурация маршрутизатора

Для простоты в примерах спецификации потоков задаются, как множители некого базового ресурса B. Колонка «Получение» показывает запросы RSVP на резервирование через выходные интерфейсы (c) и (d), а колонка «Резервы» - результирующее состояние резервирования для каждого интерфейса. В колонке «Передачи» показаны запросы на резервирование, пересылаемые в восходящем направлении на предыдущие интервалы для (a) и (b). В колонке «Резервы» каждый прямоугольник представляет «трубу» на выходном канале с соответствующим дескриптором потока.

На рисунке 5 (стиль WF) показаны две разных ситуации, требующих объединения резервов. (1) Каждый из двух следующих интервалов на интерфейсе (d) приводит к разным запросам RSVP на резервирования; эти два запроса должны быть объединены в одну эффективную спецификацию потока (3B), которая используется для организации резерва на интерфейсе (d). (2) Резервы на интерфейсах (c) и (d) должны быть объединены для пересылки запросов на резервирование в восходящем направлении; в результате самый крупный резерв (4B) пересылается в восходящем направлении каждому из предыдущих интервалов.

Рисунок 5 Пример резервирования в стиле WF

Рисунок 6 иллюстрирует резервирование в стиле FF. На каждом выходном интерфейсе организуется независимое резервирование для каждого отправителя, по отношению к которому был сделан запрос, но эти резервы являются общими для всех получателей, отправивших запросы. Дескрипторы потоков для отправителей S2 и S3, полученные через выходные интерфейсы (c) и (d), помещаются (без слияния) в запрос, пересылаемый на предыдущий интервал (b). С другой стороны три разных дескриптора потоков, задающих отправителя S1, сливаются в один запрос FF(S1{4B}), который передается на предыдущий интервал (a).

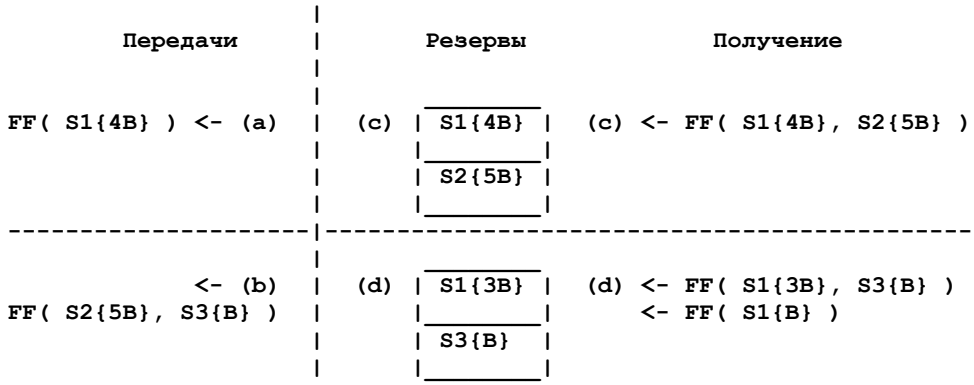


Рисунок 6 Пример резервирования в стиле FF

На рисунке 7 показан пример резервирования в стиле SE. При слиянии резервов SE спецификация результирующего фильтра является объединением спецификаций исходных фильтров, а в качестве результирующей спецификации потока используется спецификация более крупного потока.

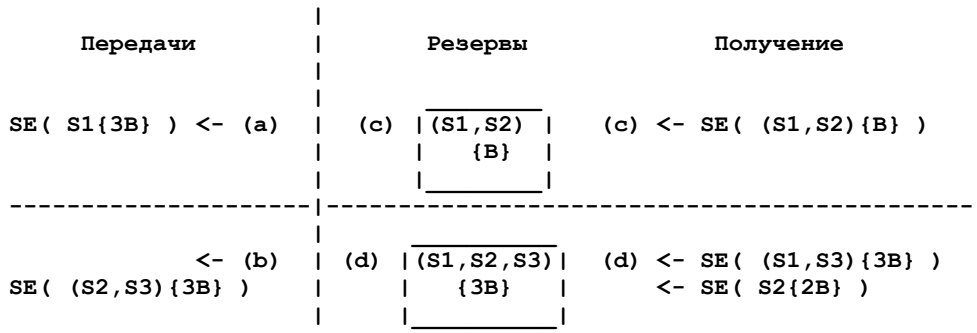
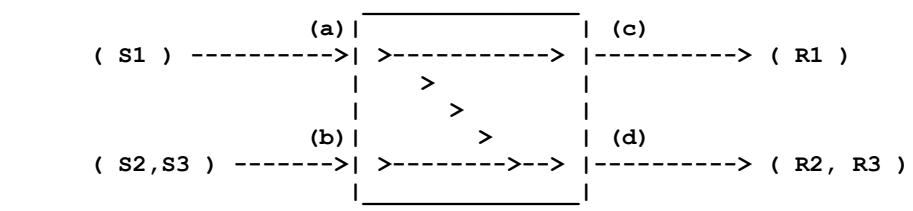


Рисунок 7 Пример резервирования в стиле SE

В приведенных выше примерах иллюстрируется допущение о том, что пакеты данных от источников S1, S2, S3 маршрутизируются на оба выходных интерфейса. На верхней части рисунка 8 показано другое допущение о маршрутизации — пакеты данных от источников S2 и S3 не пересылаются на интерфейс (c) — например, в результате того, что топология сети обеспечивает для этих отправителей более короткий путь в направлении R1 мимо данного узла. В нижней части рисунка 8 показано резервирование в стиле WF для таких условий. Поскольку нет маршрута от (b) к (c), запросы на резервирование, пересылаемые интерфейсом (b), относятся только к резервам на интерфейсе (d).



Конфигурация маршрутизатора

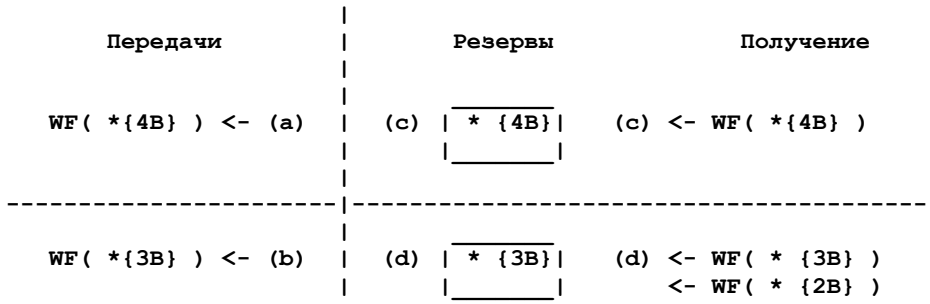


Рисунок 8 Пример резервирования в стиле WF. Частичная маршрутизация

2. Механизмы протокола RSVP

2.1 Сообщения RSVP

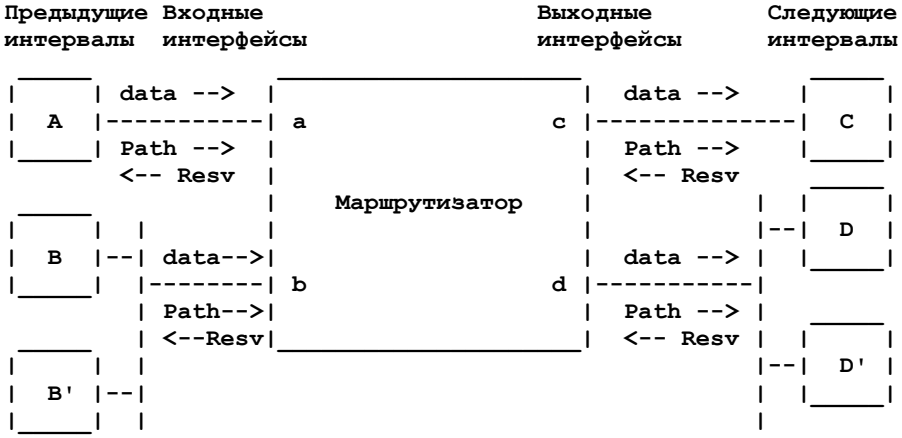


Рисунок 9: Маршрутизатор, использующий RSVP

На рисунке 9 показана модель RSVP на узле-маршрутизаторе. Каждый поток данных поступает с «предыдущего интервала» через соответствующий «входной интерфейс» и покидает узел через один или несколько «выходных интерфейсов». Один и тот же интерфейс может служить как в качестве входного, так и в качестве выходного для разных потоков данных в одной сессии. Множество предыдущих и/или следующих интервалов могут быть доступны через один физический интерфейс. Например, D и D' на рисунке соединены через широковещательную ЛВС.

В RSVP имеется два фундаментальных типа сообщений: Resv и Path.

Каждый хост-получатель шлет запросы RSVP на резервирование (Resv) в направлении отправителей. Эти сообщения должны в точности следовать пути, по которому пакеты будут доставляться получателям от всех восходящих отправителей, выбранных получателем. В результате создается и поддерживается «состояние резерва» на каждом узле вдоль пути. Сообщения Resv должны в конечном итоге доставляться на хосты-отправители, которые могут задать параметры управления трафиком для первого интервала пересылки. Обработка сообщений Resv рассматривалась выше в параграфе 1.2.

Каждый хост-отправитель RSVP передает сообщения RSVP Path в нисходящем направлении по путям индивидуальной/групповой маршрутизации, представленным протоколом маршрутизации и следующим путям передачи данных. Эти сообщения Path хранят «состояние пути» в каждом узле пути. Состояния пути включают по крайней мере индивидуальный адрес IP предыдущего интервала, который использовался для поэтапной маршрутизации сообщений Resv в обратном направлении. (В будущем некоторые протоколы маршрутизации смогут непосредственно использовать информацию о пересылке обратного пути вместо использования функции обращения маршрута в состоянии пути).

Сообщение Path в дополнение к адресу предыдущего интервала содержит следующую информацию:

- Шаблон отправителя (Sender Template)

  В сообщении Path должен содержаться шаблон Sender Template, описывающий формат пакетов данных, которые будет передавать отправитель. Шаблон имеет вид спецификации фильтра, который может применяться для выбора пакетов данного отправителя среди других пакетов в той же сессии на том же канале.

  Для шаблонов отправителей характерны такие же возможности выражения и такой же формат, которые используются в спецификации фильтров для сообщений Resv. Следовательно, Sender Template может

задавать только IP-адрес отправителя или дополнять его номером порта UDP/TCP на стороне отправителя; предполагается, что идентификатор протокола указывается для сессии.

- Sender Tspec

  В сообщении Path должна содержаться спецификация Sender Tspec, определяющая характеристики трафика для генерируемого отправителем потока данных. Эта спецификация используется системой контроля трафика для предотвращения избыточного резервирования, а также возникновения ненужных отказов Admission Control.

- Adspec

  Сообщение Path может включать анонсирующую информацию OPWA, называемую также Adspec. Данные Adspec из сообщений Path передаются в систему локального управления трафиком, которая возвращает обновленную информацию Adspec. Эта информация пересылается в сообщениях Path, передаваемых в нисходящем направлении.

Сообщения Path передаются с теми же адресами отправителя и получателя, что и данные, поэтому сообщения корректно маршрутизируются через облака без поддержки RSVP (см. параграф 2.9). С другой стороны, сообщения Resv передаются поэтапно (hop-by-hop); каждый узел RSVP пересылает сообщение Resv по индивидуальному адресу предшествующего интервала RSVP.

2.2 Слияние спецификаций потоков

Сообщения Resv, пересылаемые на предыдущий интервал, содержат спецификацию потока, который является «самым большим» из числа запрошенных следующими интервалами, для которых будет передаваться поток (в некоторых случаях могут применяться иные правила слияния, описанные в параграфе 3.5). Будем называть такую спецификацию потока «объединенной» (merged). Примеры, показанные в параграфе 1.4, иллюстрируют другой случай слияния потоков, когда имеется множество запросов резервирования от разных следующих интервалов для одной сессии с одинаковыми спецификациями фильтров, но RSVP следует задавать только один резерв на данном интерфейсе. Организованное резервирование и в этом случае должно иметь эффективную спецификацию потока, соответствующую «большему» из потоков, запрошенных разными следующими интервалами.

Поскольку спецификация потока «непрозрачна» для RSVP, реальные правила сравнения потоков должны определяться и реализоваться за пределами RSVP. Правила сравнения определяются в соответствующих спецификациях интегрированных служб. Реализация RSVP должна будет вызывать связанные с сервисом процедуры для слияния потоков.

Отметим, что спецификации потоков в общем случае представляют собой многомерные векторы — они могут содержать обе компоненты Tspec и Rspec, каждая из которых так же может быть многомерной. Следовательно, упорядочивание потоков может оказаться невозможным. Например, если один из запросов указывает большую полосу, а другой более жестко ограничивает задержки, нельзя выделить из этих запрсов «больший». В таких случаях вместо определения «большего» запроса специфичные для службы программы слияния должны возвращать третью спецификацию потока, который будет не меньше каждого из двух сравниваемых. Математически это представляет собой операцию «не меньше большего» - LUB (least upper bound). В некоторый случаях однако приходится выполнять операцию «не больше меньшего» - GLB (greatest lower bound).

Для расчета эффективной спецификации потока (Re, Te), организуемой на интерфейсе [RFC 2210], требуется выполнить перечисленные ниже операции. Здесь Te означает эффективное значение Tspec, а Re — эффективное значение Rspec.

1. Эффективная спецификация потока определяется для выходного интерфейса. В зависимости от технологии канального уровня может потребоваться слияние спецификаций потоков от нескольких предыдущих интервалов. Это означает расчет эффективной спецификации, как значения LUB для нескольких спецификаций. Отметим, что спецификации потоков для слияния определяются средой канального уровня (см. параграф 3.11.2), а механизм их слияния — используемой моделью обслуживания [RFC 2210].

   Результатом будет спецификация потока, не понятная для RSVP, но содержащая пару (Re, Resv_Te), где Re представляет собой эффективную спецификацию Rspec, а Resv_Te — эффективную спецификацию Tspec.

2. Специфичный для сервиса расчет Path_Te, определяемый суммой всех Tspec, представленных в сообщениях Path от разных предшествующих интервалов (например, некоторые или все из A, B, B' на рисунке 9).

3. (Re, Resv_Te) и Path_Te передаются в систему управления трафиком, которая будет расчитывать эффективную спецификацию арирка, как «минимальное» из значений Path_Te и Resv_Te с учетом специфики сервиса.

В параграфе 3.11.6 определен базовый набор специфичных для служб процедур для сравнения спецификаций потоков, расчета LUB и GLB, а также сравнения и суммирования Tspec.

2.3 Модель Soft State

RSVP использует модель soft state для управления состоянием резервирования на маршрутизаторах и хостах. RSVP soft state создается и периодически обновляется с помощью сообщений Path и Resv. Состояние удаляется, если по истечении интервала очистки (cleanup timeout) не поступило ни одного обновляющего сообщения. Состояние может быть также удалено явно с помощью специального сообщения teardown, описанного в следующем параграфе. По истечении каждого периода ожидания обновлений и после смены состояния RSVP сканирует свое состояние для создания и пересылки обновляющих сообщений Path и Resv на последующие интервалы.

Сообщения Path и Resv являются идемпотентными. При изменении маршрута следующее сообщение Path будет инициализировать состояние пути для нового маршрута, а будущие сообщения Resv станут организовывать состояние резервирования на нем; состояние на неиспользуемом сегменте маршрута будет устаревать. Таким образом, вопрос является ли сообщение является «новым» или «обновляющим» (refresh) решается каждым узлом раздельно в зависимости от наличия состояния на данном узле.

RSVP передает свои сообщения в форме дейтаграмм IP без гарантии доставки. Предполагается периодическая передача сообщений с обновлениями от хостов и маршрутизаторов для предотвращения влияния возможной потери сообщений RSVP. Если тайм-аут эффективной очистки превышает тайм-аут повтора в K раз, протокол RSVP может перенести потерю K-1 последовательного пакета RSVP без ложного удаления состояний. Механизмы управления сетевым трафиком должны быть статически настроены на обеспечение сообщениям протокола RSVP некой минимальной полосы, чтобы предотвратить потерю пакетов при возникновении перегрузки.

Поддерживаемое протоколом RSVP состояние является динамическим — для изменения набора отправителей Si или смены запроса QoS хост просто начинает передачу измененных сообщений Path и/или Resv. Результатом этого будет соответствующее изменение состояния RSVP на всех узлах вдоль пути; неиспользуемые состояния будут удаляться по тайм-ауту, если они не будут уничтожены явно.

В статичных условиях состояние обновляется поэтапно (hop-by-hop) для обеспечения возможности слияния. Когда принятое состояние отличается от сохраненного, последнее обновляется. Если это обновление приводит к смене состояния, пересылаемого в сообщениях refresh, такие сообщения должны генерироваться и пересылаться без промедления, чтобы обеспечить сквозную смену состояний без задержки. Однако распространение изменений останавливается по достижении точки, в которой слияние не приводит к смене состояния. Это минимизирует управляющий трафик RSVP, что имеет важное значение для работы с большими multicast-группами.

Состояние, принятое через тот или иной интерфейс I*, никогда не следует пересылать через этот же интерфейс. Состояние, пересылаемое через интерфейс I*, должно рассчитываться только с использованием состояний, полученных через интерфейсы, отличные от I*. Тривиальный пример этого показан на рисунке 10, где транзитный маршрутизатор имеет одного получателя и одного отправителя на каждом интерфейсе (в предположении одного предыдущего/следующего интервала на интерфейс). Интерфейсы (a) и (c) для данного сеанса служат как входными, так и выходными. Оба получателя делают резервирование в стиле шаблона (wildcard-style), при котором сообщения Resv передаются на все предыдущие интервалы для отправителей в группе, за исключением следующего интервала (next hop) с которого они поступают. Результатом будет независимое резервирование в двух направлениях.

Есть дополнительное правило, относящееся к пересылке сообщений Resv — состояние из сообщения Resv, полученное от выходного интерфейса Io следует пересылать во входной интерфейс Ii только в тех случаях, когда сообщения Path от Ii пересылаются в Io.

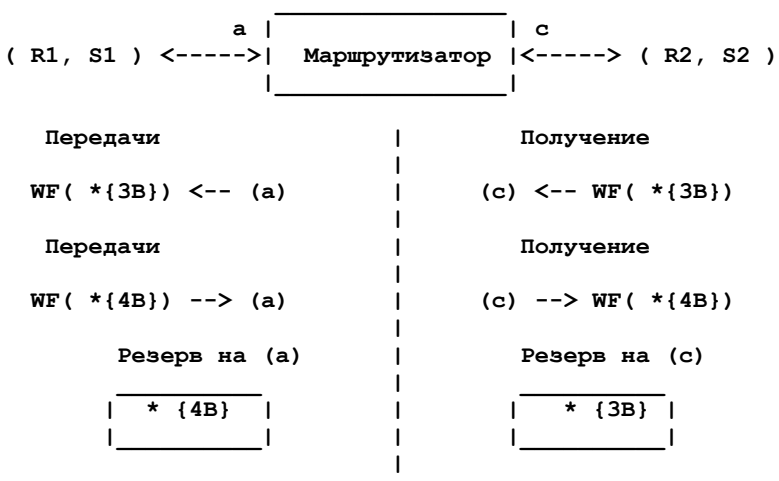


Рисунок 10 Независимое резервирование

2.4 Сообщения Teardown

Сообщения RSVP teardown незамедлительно удаляют путь или состояние резервирования. Хотя и не требуется явно уничтожать старое резервирование, всем конечным хостам рекомендуется отправлять запросы teardown сразу по завершении работы приложения.

Имеется два типа сообщений RSVP teardown - PathTear и ResvTear. Сообщение PathTear передается в направлении всех получателей в нисходящем потоке, а также для всех вовлеченных состояний резервирования вдоль пути. Сообщение ResvTear удаляет состояние резервирования и передается в направлении всех получателей в восходящем направлении от точки инициирования. Сообщение PathTear (ResvTear) можно концептуально рассматривать, как чувствительное к резервированию сообщение Path message (и, соответственно, Resv).

Запрос на удаление может быть инициирован приложением в оконечной системе (получатель или отправитель) или маршрутизатором в результате тайм-аута для состояния или упреждения от службы. После инициирования запрос на удаление поэтапно (hop-by-hop) пересылается без задержки. Сообщения teardown удаляют указанное состояние в принявшем сообщение узле. Как и в других случаях, информация о смене состояния незамедлительно пересылается следующему узлу, если после слияния состояние реально изменилось. В результате сообщение ResvTear будет «урезать» резервирование с возможной быстротой.

Подобно всем остальным сообщениям RSVP, сообщения teardown не доставляются гарантированно. Потеря сообщения teardown с запросом не будет приводить к протокольному отказу, поскольку неиспользуемое состояние «умрет» по истечении времени даже без явного удаления. Если сообщение teardown будет потеряно, маршрутизатор, который не получил это сообщение будет ждать тайм-аута для состояния и инициировать новое сообщение teardown за точкой потери сообщения. В предположении малой вероятности потери сообщений RSVP можно считать, что наибольшее время удаления состояния в некоторых случаях будет превышать один период обновления по тайм-ауту.

Можно удалить любой набор организованных состояний. Для состояния пути гранулярность задается отдельной спецификацией фильтра (см., например, рис. 7). Получатель R1 может отправить сообщение ResvTear только для отправителя S2 (или любого подмножества из спецификации фильтра), сохранив S1.

Сообщение ResvTear задает стиль и фильтры, любые спецификации потоков игнорируются. Имеющаяся спецификация потоков будет удаляться, если все спецификации фильтров для нее удалены.

2.5 Ошибки

Существует два типа сообщений RSVP об ошибках - ResvErr и PathErr. Сообщения PathErr очень просты, они передаются в восходящем направлении породившему ошибку отправителю и не меняют состояния пути на узлах, через которые проходят. Существует лишь несколько возможных причин возникновения таких ошибок.

Однако имеется множество синтаксически корректных запросов на резервирование, которые могут вызывать отказы в той или иной точке пути. Узел также может принять решение об овладении организованным резервированием. Обработка сообщений ResvErr несколько сложнее (параграф 3.5). Поскольку вызвавший отказ запрос может быть результатом слияния множества запросов, сообщение об ошибке резервирования должно быть передано всем вовлеченным получателям. Кроме того, слияние разнородных запросов создает потенциальную сложность, известную как «убойное резервирование» (killer reservation), когда один запрос может препятствовать выполнению другого. На практике возникают две проблемы «убойного резервирования».

1. Первая проблема (KR-I) возникает в тех случаях, когда уже имеется резервирование Q0. Если другой получатель будет делать более крупное резервирование Q1 > Q0, результат слияния Q0 и Q1 может быть отвергнут контролем допуска одного из узлов в восходящем направлении. В этом случае обслуживание Q0 будет прервано.

   Для этой проблемы имеется простое решение — когда контроль допуска отвергает запрос на резервирование, существующие резервы следует сохранять.

2. Вторая проблема (KR-II) является обратной — получатель, заказавший резервирование Q1, продолжает упорствовать, несмотря на отказ контроля допуска для Q1 на том или ином узле. Это может воспрепятствовать другому узлу в организации менее крупного резервирования Q0, которое не удастся без слияния cQ1.

   Для решения этой проблемы сообщение ResvErr организует другое состояние, называемое «блокадой» (blockade state), на каждом узле, через который проходит это сообщение. Блокада на узле меняет процедуру слияния с целью предотвратить слияние в недопустимой спецификацией потока (Q1 в нашем примере), что позволит пересылать и организовывать резервы для меньших запросов. Состояние резервирования для Q1 называют «заблокированным». Подробное описание блокирования приведено в параграфе 3.5.

Запрос на резервирование, для которого Admission Control дает отказ, порождает блокаду, но сохраняется на узлах в нисходящем направлении от точки отказа. Высказывались предположения о том, что сохранение таких резервов нецелесообразно и их следует удалять явно или по тайм-ауту. Ниже перечислены несколько причин, по которым такие резервы все-таки следует сохранять.

- Существует две причины, по которым получатель может сохранять резерв с отказом: (1) определение доступности ресурсов на всем пути; (2) желание получить требуемые параметры QoS на максимальной протяженности в пути доставки. Во втором (а возможно, и в первом) случае получатель захочет сохранить организованные резервы на участках пути в нисходящем направлении от точки отказа.

- Если резервы ниже точки отказа не сохранены, реакция RSVP на некоторые временные отказы может быть нарушена. В качестве примера предположим, что путь переключается (flap) на другой маршрут, который оказывается перегруженным и для организованных резервов возникнут, вследствие чего будет быстро восстановлен первоначальный путь. Для состояния блокады на каждом маршрутизаторе нисходящего направления недопустимо удалять состояние или препятствовать его незамедлительному обновлению.

- Если не обновлять резервы в нисходящем направлении, возможен тайм-аут для восстановления каждые Tb секунд (Tb — тайм-аут для состояния блокады). Такое нестабильное поведение может оказаться неприемлемым для пользователей.

2.6 Подтверждения

Для запроса подтверждения своего запроса на резервирование получатель Rj включает в свое сообщение Resv объект запроса подтверждения (confirmation-request), содержащий IP-адрес Rj. В каждой точке слияния в восходящем направлении пересылается лишь наиболее крупная спецификация потока и все сопровождающие ее запросы подтверждения. Если запрос на резервирование от Rj не превышает имеющееся на узле резервирование, сообщение Resv не пересылается дальше и при наличии в Resv запроса на подтверждение получателю Rj возвращается сообщение ResvConf. Если запрос подтверждения пересылается, это выполняется незамедлительно и не более одного раза для каждого запроса.

Механизм подтверждения вызывает рассмотренные ниже последствия.

- Новый запрос на резервирование со спецификацией потока, превышающей любую из имеющихся для сессии обычно будут приводить к возврату сообщения ResvErr или ResvConf в адрес получателя от каждого отправителя. В таких случаях сообщение ResvConf будет сквозным подтверждением.

- Получение ResvConf не дает гарантий. Предположим, что два первых запроса на резервирование от получателей R1 и R2 поступили на узел, где были слиты в один запрос. R2, запрос которого поступил вторым, может получить ResvConf от узла, тогда как запрос R1 еще не прошел весь путь до соответствующего отправителя и может привести к отказу. Таким образом, R2 может получить ResvConf, хотя сквозного резервирования не было сделано; более того, вслед за ResvConf получатель R2 может принять ResvErr.

2.7 Управление правилами

Направляемые RSVP запросы QoS позволяют отдельным пользователям получать преимущественный доступ к сетевым ресурсам. Для предотвращения злоупотреблений такой возможностью обычно требуется некоторый механизм обратного воздействия на пользователей, выполняющих такое резервирование. Например, это можно реализовать с помощью административных правил доступа или запроса от пользователей обратной связи в форме реальной или

виртуальной оплаты резервирования. В любом случае будут полезны надежная идентификация пользователей и селективное исполнение запросов на резервирование.

Термин «контроль политики» (policy control) используется для обозначения механизма, который нужен для поддержки правил доступа и реализации обратного воздействия при резервировании RSVP. При запросе нового резервирования каждый узел должен ответить на два вопроса: «Достаточно ли ресурсов для выполнения этого запроса?» и «Пользователь имеет право на такое резервирование?». Ответы на эти вопросы определяют «контроль исполнения» (admission control) и «контроль политики» (policy control), соответственно. Для резервирования RSVP на оба вопроса нужен положительный ответ. Разные административные домены Internet могут придерживаться разных правил резервирования.

Исходную информацию для контроля политики называют «данными политики» (policy data) - RSVP передает эти данные в объектах POLICY_DATA. Данные могут включать идентифицирующие пользователей или их классы «мандаты» (credential), учетные номера, ограничения, квоты и т. п. Подобно спецификациям потоков, данные политики «непрозрачны» для протокола RSVP, просто передающего их при необходимости системе контроля политики. Кроме того, слияние данных политики должно выполняться механизмом контроля политики, а не протоколом RSVP. Отметим, что точки слияния данных политики очевидно будут располагаться на границах административных доменов. Следовательно, может возникнуть необходимость передачи собранных и не слитых воедино данных политики восходящего направления через множество узлов до одной из точек слияния.

Слияние представленных пользователем данных политики из сообщений Resv может вызывать проблему при масштабировании. Когда multicast-группа включает большое число получателей, может оказаться невозможной или нежелательной передача данных политики от всех получателей в восходящем направлении. Данные политики будут административными методами сливаться неподалеку от получателей для предотвращения излишнего объема таких данных. Дополнительное рассмотрение этого вопроса и пример схемы контроля политики можно найти в работе [PolArch96]. Спецификации формата объектов данных политики и правила их обработки в RSVP еще разрабатываются.

2.8 Безопасность

С протоколом RSVP связано несколько вопросов безопасности, рассмотренных ниже.

- Целостность сообщений и аутентификация узлов

  Поврежденные или подставные запросы на резервирование могут порождать захват служб не уполномоченными потребителями или DoS-атаки за счет блокировки сетевых ресурсов. Протокол RSVP обеспечивает защиту от подобных атак с помощью механизма поэтапной (hop-by-hop) аутентификации, использующего шифрованную хэш-функцию. Этот механизм поддерживается за счет объектов INTEGRITY, которые могут включаться в любое сообщение RSVP. Эти объекты используют криптографические методы цифровой подписи, которые предполагают, что соседи RSVP знают общий секрет. Хотя этот механизм является частью базовой спецификации RSVP, он описан в отдельном документе [Baker96].

  Повсеместное распространение механизма защиты целостности RSVP потребует доступности инфраструктуры управления и распространения ключей для маршрутизаторов. До появления такой инфраструктуры обеспечение целостности сообщений RSVP будет требовать ручного управления ключами.

- Аутентификация пользователей

  Политика управления зависит от аутентификации пользователей, ответственных за запросы резервирования. Следовательно, данные политики включают криптографически защищенные сертификаты пользователей. Спецификация таких сертификатов является вопросом будущего.

  Даже без верифицируемых в глобальном масштабе пользовательских сертификатов во многих случаях возможно обеспечить аутентификацию пользователей путем организации цепочек доверия с использованием описанного выше поэтапного механизма INTEGRITY.

- Защищенные потоки данных

  Первые два вопроса безопасности связаны с работой RSVP. Третий вопрос относится к резервированию ресурсов для защищенных потоков данных. В частности, применение IPSEC (IP Security) для потоков данных вызывает проблемы для RSVP — если заголовки транспортного и вышележащих уровней будут зашифрованы, обобщенные номера портов RSVP будет невозможно использовать для определения сессии или получателя.

  Для решения этой проблемы было разработано расширение RSVP, в котором идентификатор защищенной связи (IPSEC SPI) играет роль эквивалента обобщенного порта [RFC 2207].

2.9 Облака без RSVP

Невозможно развернуть RSVP (как и любой новый протокол) разом во всей сети Internet. Более того, RSVP в некоторых частях сети может просто не разворачиваться совсем. Следовательно, RSVP должен корректно работать даже в тех случаях, когда два поддерживающих RSVP маршрутизатора связаны между собой через «облако» не поддерживающих RSVP маршрутизаторов. Естественно, на промежуточных узлах без поддержки RSVP не будет возможности резервировать ресурсы. Однако при достаточной пропускной способности такого облака оно не будет создавать помех.

RSVP разработан с учетом передачи через облака, не поддерживающие RSVP. Маршрутизаторы (RSVP и не-RSVP) пересылают сообщения Path в направлении получателя с использованием своих таблиц индивидуальной и групповой маршрутизации. Следовательно, на маршрутизацию сообщений Path отсутствие поддержки RSVP на пути передачи влиять не будет. При прохождении сообщений Path через облако без поддержки RSVP эти сообщения будут приходить на следующий маршрутизатор RSVP с IP-адресом последнего маршрутизатора RSVP перед входом в облако. Сообщения Resv будут пересылаться напрямую следующему маршрутизатору RSVP по пути в направлении источника.

Хотя RSVP корректно работает через облака без RSVP, не поддерживающие RSVP узлы способны нарушать параметры QoS, представленные получателем. По этой причине RSVP передает флаг NonRSVP локальному механизму управления трафиком в тех случаях, когда на пути к данному отправителю встречаются интервалы без поддержки RSVP. Система управления трафиком комбинирует полученный флаг с данными из своих источников и

пересылает результирующую информацию о поддержке интегрированных услуг по пути к получателям, используя сообщения Adspec [RFC 2210].

В некоторых топологиях, где только часть маршрутизаторов поддерживает RSVP, возможна доставка сообщений Resv не тому маршрутизатору RSVP или не на тот интерфейс нужного маршрутизатора RSVP. Процесс RSVP должен быть готов к работе в таких ситуациях. Если адрес получателя не соответствует ни одному из локальных интерфейсов и сообщение не относится к типу Path или PathTear, такое сообщение следует пересылать дальше без локальной обработки. Для случаев приема сообщения через некорректный интерфейс используется значение LIH[1]. Информация предыдущего интервала в сообщении Path включает не только IP-адрес предыдущего узла, но и данные LIH, указывающие локальный выходной интерфейс (оба значения хранятся в состоянии пути). Сообщение Resv, прибывшее на узел, содержит значения IP-адреса и LIH корректного выходного интерфейса (т. е., интерфейса, через который следует получать запрошенное резервирование, независимо от того интерфейса, через который пришло сообщение).

Идентификатор LIH может быть полезен также в случаях, когда резервирование RSVP выполняется через сложный канальный уровень, для отображения объектов уровня IP на объекты канального уровня.

2.10 Модель хоста

До того, как можно будет организовать сессию, требуется присвоить идентификаторы сессии (DestAddress, ProtocolId [, DstPort]) и передать их всем отправителям и получателям с использованием того или иного механизма передачи по независимым каналам (out-of-band mechanism). При организации сессии RSVP в конечных системах происходит ряд событий:

H1 получатель присоединяется к multicast-группе, указанной DestAddress, используя IGMP;
H2 потенциальный отправитель начинает передачу сообщений RSVP Path по адресу DestAddress;
H3 принимающее приложение получает сообщение Path;
H4 получатель начинает передачу соответствующих сообщений Resv, задающих дескрипторы потока;
H5 передающее приложение получает сообщение Resv;
H6 отправитель начинает передачу пакетов данных.

С этими событиями связаны несколько вопросов синхронизации.

- H1 и H2 могут происходить в произвольном порядке.
- Предположим, что новый отправитель начинает передачу данных (H6), но групповых маршрутов еще нет, поскольку ни один получатель не присоединился к группе (H1). Данные будут отбрасываться на одном из узлов маршрутизации (в зависимости от протокола маршрутизации), пока не появится получатель.
- Предположим, что новый отправитель одновременно начинает передачу сообщений Path (H2) и данных (H6), имеются получатели, но сообщений Resv к отправителю еще не приходило (например, по причине того, что его сообщения Path еще не дошли до получателей). Начальные данные могут прийти к получателям без желаемого QoS. Отправитель может ослабить эту проблему, дождавшись первого сообщения Resv (H5), однако удаленные от него получатели могут еще не успеть организовать резервирование.
- Если получатель начинает передачу сообщений Resv (H4) до получения какого-либо сообщения Path (H3), RSVP будет возвращать получателю сообщения об ошибках.

  Получатель может просто игнорировать такие сообщения или избежать их появления, дождавшись получения Path без передачи сообщение Resv.

В этой спецификации не определяется конкретный прикладной интерфейс (API[2]) для RSVP, поскольку он может зависеть от операционной системы хоста. Однако в параграфе 3.11.1 рассматриваются общие требования и схема базового интерфейса.

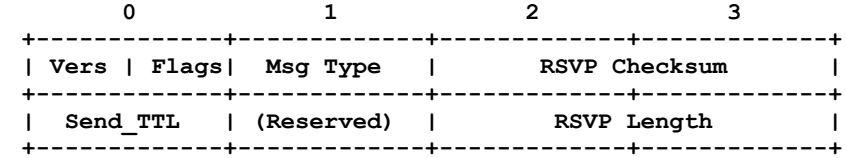
3. Функциональные спецификации RSVP

3.1 Формат сообщений RSVP

Сообщение RSVP включает общий заголовок, за которым следует тело сообщения, содержащее переменное число типизованных «объектов» разного размера. В последующих параграфах определены общий заголовок, стандартный заголовок объекта и все типы сообщений RSVP.

Для каждого типа сообщений RSVP имеется набор правил выбора допустимых типов объектов. Правила задаются с использованием синтаксиса Бэкуса-Наура (BNF[3]), дополненного квадратными скобками для указания необязательных субпоследовательностей. В формах BNF используется порядок размещения объектов в сообщении. Однако во многих (но не во всех) случаях порядок объектов не имеет логического значения. Реализациям следует создавать сообщения с указанным здесь порядком объектов, но следует принимать объекты в любом допустимом порядке.

3.1.1 Общий заголовок



Поля общего заголовка рассматриваются ниже.

Vers: 4 бита

Номер версии протокола. Данная версия имеет номер 1.

[1]Logical Interface Handle — идентификатор логического интерфейса.
[2]Application program interface — интерфейс с прикладными программами.
[3]Backus-Naur Form/

Flags: 4 бита

0x01-0x08: Резерв

Биты флагов пока не определены.

Msg Type: 8 битов

1 = Path
2 = Resv
3 = PathErr
4 = ResvErr
5 = PathTear
6 = ResvTear
7 = ResvConf

RSVP Checksum: 16 битов

Дополнение до 1 суммы дополнений до 1 для сообщения. Значение самого поля контрольной суммы на момент ее расчета принимается нулевым. Значение контрольной суммы, содержащее только нули (all-zero) говорит о том, что контрольная сумма не передается.

Send_TTL: 8 битов

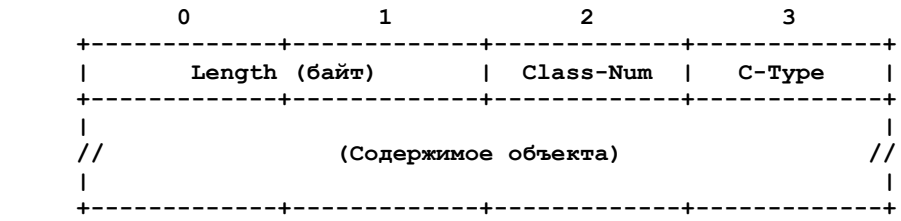
Значение IP TTL, с которым сообщение было передано (см. параграф 3.8).

RSVP Length: 16 битов

Общий размер сообщения RSVP в байтах с учетом общего заголовка и следующих за ним объектов переменного размера.

3.1.2 Форматы объектов

Каждый объект включает по крайней мере одно 32-слово и заголовок размером в одно слово.



Поля заголовка перечислены ниже.

Length

16-битовое полу, указывающее общий размер объекта в байтах. Значения поля кратны 4, минимальное значение 4.

Class-Num

Идентифицирует класс объекта; значения поля определены в Приложении A. Каждый класс объектов имеет имя, которое в этом документе указывается заглавными буквами. Реализации RSVP должны распознавать следующие классы объектов:

NULL

Пустой (NULL) объект имеет Class-Num=0 и его значение C-Type игнорируется. Размер объекта должен быть не меньше 4 и может принимать значения, кратные 4. NULL-может находиться в любом месте последовательности объектов, а его содержимое получатель игнорирует.

SESSION

Содержит IP-адрес получателя (DestAddress), идентификатор протокола IP и ту или иную форму обобщенного порта получателя, определяющие конкретную сессию для последующих объектов. Должен присутствовать в каждом сообщении RSVP.

RSVP_HOP

Содержит IP-адрес поддерживающего RSVP узла, который передает это сообщение и логический идентификатор выходного интерфейса (LIH; см. параграф 3.3). В этом документе объект RSVP_HOP рассматривается, как объект PHOP[1] для нисходящих сообщений и как NHOP[2] для восходящих.

TIME_VALUES

Содержит значение интервала обновления R, используемое создателем данного сообщения (см. параграф 3.7). Требуется для каждого сообщения Path и Resv.

STYLE

Определяет стиль резервирования и специфическую для этого стиля информацию, которая не включена в объекты FLOWSPEC или FILTER_SPEC. Требуется для каждого сообщения Resv.

FLOWSPEC

Определяет желаемый уровень QoS в сообщении Resv.

FILTER_SPEC

Определяет подмножество пакетов данных сессии, которые должны получить желаемый уровень QoS (задается объектом FLOWSPEC) в сообщении Resv.

SENDER_TEMPLATE

Содержит IP-адрес отправителя и может включать дополнительные данные демультиплексирования для идентификации отправителя. Требуется в сообщениях Path.

SENDER_TSPEC

Определяет характеристики трафика для потока данных отправителя. Требуется в сообщениях Path.

ADSPEC

Передает данные OPWA в сообщениях Path.

ERROR_SPEC

Указывает ошибку в сообщениях PathErr, ResvErr или подтверждение в ResvConf.

POLICY_DATA

Содержит информацию, которая позволит локальному модулю политики определить, допустимо ли запрошенное резервирование на административном уровне. Может присутствовать в сообщениях Path, Resv, PathErr, ResvErr.

[1]Previous hop — предыдущий интервал.
[2]Next hop — следующий интервал.

Использование объектов POLICY_DATA данная спецификация не определяет полностью; это будет сделано в последующих документах.

INTEGRITY

Содержит криптографические данные для аутентификации узла-источника и проверки содержимого данного сообщения RSVP. Применение объектов INTEGRITY описано в [Baker96].

SCOPE

Содержит явный список хостов-отправителей, в направлении которых сообщение пересылается. Может присутствовать в сообщениях Resv, ResvErr, ResvTear (см. параграф 3.4).

RESV_CONFIRM

Содержит IP-адрес получателя, запросившего подтверждение. Может присутствовать в сообщениях Resv и ResvConf.

C-Type

Тип объекта, уникальный в рамках Class-Num. Значения типов определены в Приложении A.

Максимальный размер содержимого объекта составляет 65528 байтов. Поля Class-Num и C-Type могут использоваться совместно, как 16-битовое сило, уникально определяющее тип каждого объекта.

Два старших бита Class-Num используются для определения действия, которое узлу следует предпринять, если значение Class-Num не распознано (см. параграф 3.10).

3.1.3 Сообщения Path

Каждый хост-отправитель периодически отправляет сообщения Path для каждого порождаемого им потока данных. Сообщение включает объект SENDER_TEMPLATE, определяющий формат пакетов данных, и объект SENDER_TSPEC, задающий характеристики трафика в потоке. В дополнение к этому сообщение может включать объект ADSPEC, содержащий анонс (OPWA) данных для потока.

Сообщение Path перемещается от отправителя к получателю(ям) по тому же пути, который служит для доставки данных. IP-адрес источника в сообщении Path должен совпадать с адресом отправителя, к которому относится сообщение, а в качестве адреса получателя должно указываться значение DestAddress для данной сессии. Эти адреса обеспечивают корректную маршрутизацию сообщения через сети без поддержки RSVP.

Формат сообщения Path показан ниже.

```
<Path Message> ::= <Common Header> [ <INTEGRITY> ] <SESSION> <RSVP_HOP> <TIME_VALUES>
                   [ <POLICY_DATA> ... ] [ <sender descriptor> ]

<sender descriptor> ::= <SENDER_TEMPLATE> <SENDER_TSPEC> [ <ADSPEC> ]
```

При наличии объекта INTEGRITY он должен размещаться сразу после общего заголовка. Других требований к порядку размещения не вводится, хотя рекомендуется применять показанный выше порядок. Число объектов POLICY_DATA может быть любым.

Объект PHOP (т. е., RSVP_HOP) в каждом сообщении Path содержит адрес предыдущего интервала (IP-адрес интерфейса, через который было передано сообщение Path). В нем также содержится идентификатор логического интерфейса (LIH).

Каждый узел пути, поддерживающий RSVP, принимает сообщение Path и обрабатывает его, создавая состояние пути для отправителя, определяемое объектами SENDER_TEMPLATE и SESSION. Любые объекты POLICY_DATA, SENDER_TSPEC и ADSPEC также сохраняются в состоянии пути. Если при обработке сообщения Path возникает ошибка,исходному отправителю этого сообщения передается сообщение PathErr. Сообщения Path должны удовлетворять правилам в части SrcPort и DstPort, описанным в параграфе 3.2.

Процесс RSVP на узле периодически сканирует состояние пути для создания новых сообщений Path с целью их пересылки в направлении получателей. Каждое сообщение содержит дескриптор отправителя, определяющий одного отправителя, а также IP-адреса исходного отправителя в качестве IP-адреса источника сообщения. Сообщения Path в конечном итоге приходят к приложениям на всех получателях, однако они не возвращаются к получателю, работающему в одном прикладном процессе с отправителем.

Процесс RSVP пересылает сообщения Path и реплицирует их в соответствии с потребностями групповых сессий, используя маршрутные данные, получаемые от процесса маршрутизации (групповой или индивидуальной). Маршрут зависит от значения DestAddress для сессии, а для некоторых протоколов маршрутизации — от IP-адреса источника (отправителя). Маршрутная информация в общем случае включает список (возможно, пустой) выходных интерфейсов, в которые пересылается сообщение Path. Поскольку каждый выходной интерфейс имеет свой адрес IP, переданные через разные интерфейсы сообщения Path будут содержать разные адреса PHOP. Кроме того, объекты ADSPEC, передаваемые в сообщениях Path также обычно меняются в зависимости от выходного интерфейса.

Состояние пути для данной сессии и отправителя не обязательно будет иметь уникальное значение PHOP или уникальный входной интерфейс. Для индивидуальных и групповых сессий возникают два случая.

- Multicast-сессии

  Групповая маршрутизация обеспечивает стабильное дерево распределения, по которому сообщения Path от одного отправителя приходят через разные PHOP и протокол RSVP должен быть готов к поддержке таких состояний путей. Правила RSVP для обработки таких ситуаций рассмотрены в параграфе 3.9. Для RSVP недопустимо (в соответствии с правилами 3.9) пересылать сообщения Path, которые поступили на входной интерфейс, отличающийся от представленного маршрутизацией.

- Unicast-сессии

  В течение короткого интервала после смены unicast-маршрута в восходящем направлении узел может получать сообщения Path для пары сессия-отправитель от разных PHOP. Узел не может определить корректный PHOP, хотя данные будут приходить все время только от одного из PHOP. Одни реализации RSVP игнорируют PHOP с соответствии с прошлым состоянием и позволяют смену PHOP среди кандидатов. Другие реализации поддерживают состояние пути для каждого PHOP и передают сообщения Resv в восходящем

направлении всем таким PHOP. В любом случае ситуация является переходной — неиспользуемое состояние пути отбрасывается по тайм-ауту или сносится (по тайм-ауту состояния восходящего пути).

3.1.4 Сообщения Resv

Сообщения Resv поэтапно (hop-by-hop) переносят запросы на резервирование от получателей к отправителям по пути обратному пути доставки потока данных в этой сессии. IP-адресом получателя сообщения Resv служит индивидуальный адрес предыдущего узла, полученный из состояния пути. В качестве IP-адреса отправителя служит адрес передавшего сообщение узла.

Формат сообщение Resv показан ниже.

```
<Resv Message> ::= <Common Header> [ <INTEGRITY> ]
                          <SESSION>  <RSVP_HOP>
                          <TIME_VALUES>
                          [ <RESV_CONFIRM> ]  [ <SCOPE> ]
                          [ <POLICY_DATA> ... ]
                          <STYLE> <flow descriptor list>

<flow descriptor list> ::=  <empty> |
                       <flow descriptor list> <flow descriptor>
```

При наличии объекта INTEGRITY он должен размещаться сразу же за общим заголовком. Объект STYLE со следующим за ним списком дескрипторов потока должен размещаться в конце сообщения, а объекты в списке дескрипторов потока должны использовать показанный ниже синтаксис BNF. Других требований к порядку передачи не предъявляется, но рекомендуется использовать показанный выше порядок.

Объект NHOP (т. е., RSVP_HOP) содержит IP-адрес интерфейса, через который было передано сообщение Resv и значение LIH для логического интерфейса, на котором требуется резервирование.

Присутствие объекта RESV_CONFIRM говорит о подтверждении запроса на резервирование; этот объект содержит IP-адрес получателя, которому следует отправить сообщение ResvConf. Число объектов POLICY_DATA не ограничено.

Приведенная выше форма BNF определяет список дескрипторов потока, как простой список дескрипторов потока. Приведенные ниже правила для разных стилей уточняют форму списка дескрипторов потока для каждого стиля.

- Стиль WF:

```
<flow descriptor list> ::=  <WF flow descriptor>

<WF flow descriptor> ::= <FLOWSPEC>
```

- Стиль FF:

```
<flow descriptor list> ::= <FLOWSPEC> <FILTER_SPEC>
                         | <flow descriptor list> <FF flow descriptor>
<FF flow descriptor> ::= [ <FLOWSPEC> ] <FILTER_SPEC>
```

Каждый элементарный запрос в стиле FF определяется одной парой (FLOWSPEC, FILTER_SPEC) и множество таких запросов может паковаться в список дескрипторов одного сообщения Resv. Объект FLOWSPEC может быть опущен, если он идентичен самому свежему объекту этого типа в списке; первый дескриптор FF должен включать FLOWSPEC.

- Стиль SE:

```
<flow descriptor list> ::= <SE flow descriptor>
<SE flow descriptor> ::= <FLOWSPEC> <filter spec list>
<filter spec list> ::=  <FILTER_SPEC> |  <filter spec list> <FILTER_SPEC>
```

Область резервирования, т. е. набор отправителей, которым должно быть переслано конкретное резервирование (после слияния), определяется следующим образом:

- Явный выбор отправителя

  Резервирование пересылается всем отправителям, для которых сохраненные в состоянии пути объекты SENDER_TEMPLATE соответствуют объекту FILTER_SPEC в резервировании. Правила соответствия описаны в параграфе 3.2.

- Выбор отправителей по шаблону

  Запросу с выбором отправителя по шаблону будут соответствовать все отправители, маршруты к которым проходят через данный выходной интерфейс.

  Всякий раз, когда сообщение Resv с шаблонным выбором отправителей передается на несколько предыдущих интервалов, в сообщение должен включаться объект SCOPE (см. параграф 3.4); в этом случае область пересылки резервирования ограничивается IP-адресами отправителей, указанными в объекте SCOPE.

  Пересылаемые узлом сообщения Resv в общем случае являются результатом слияния набора принятых сообщений Resv (тех, которые не блокируются, см. параграф 3.5). Если одно из сливаемых сообщений содержит объект RESV_CONFIRM и имеет спецификацию резервирования (FLOWSPEC), превышающую запросы других сливаемых сообщений, этот объект RESV_CONFIRM пересылается в выходном сообщении Resv. Объект RESV_CONFIRM в одном из остальных сливаемых сообщений (чья спецификация потока совпадает, меньше или несовместима со спецификацией после слияния, но запрос не блокируется) будет вызывать генерацию сообщения ResvConf, содержащего RESV_CONFIRM. Объект RESV_CONFIRM из заблокированного не будет ни пересылаться, ни возвращаться — он просто отбрасывается данным узлом.

3.1.5 Сообщения PathTear

Результатом получения PathTear (демонтаж пути) является удаление соответствующего состояния пути. Соответствие должно выполняться для объектов SESSION, SENDER_TEMPLATE и PHOP. Кроме того, сообщение PathTear для групповой сессии может соответствовать только состоянию пути для входного интерфейса, через который поступило это сообщение. При отсутствии соответствующего состояния пути сообщение PathTear следует отбросить без пересылки.

Сообщения PathTear явно инициируются отправителями или (при возникновении тайм-аута для состояния пути) промежуточными узлами и передаются в нисходящем направлении для всех получателей. Сообщение PathTear с индивидуальным адресом не пересылается, если имеется состояние пути для той же пары (сессия, отправитель), но с другим PHOP. Пересылка сообщений PathTear осуществляется по правилам, описанным в параграфе 3.9.

Сообщения PathTear должны маршрутизироваться явно, подобно соответствующим сообщениям Path. Следовательно, IP-адресом получателя в таком сообщении должно быть значение DestAddress для сесии, а IP-адресом отправителя должен быть адрес отправителя из состояния пути, который будет демонтирован.

```
<PathTear Message> ::= <Common Header> [ <INTEGRITY> ]
                                <SESSION> <RSVP_HOP> [ <sender descriptor> ]

<sender descriptor> ::= (см. определение выше)
```

Сообщение PathTear может включать объект SENDER_TSPEC или ADSPEC в своем дескрипторе отправителя, но этот объект должен игнорироваться. Порядок объектов был описан ранее для сообщений Path, но рекомендуется использовать показанный в этом параграфе порядок.

Удаление состояния пути в результате приема сообщения PathTear или тайм-аута должно также вызывать корректировку связанного с этим путем состояния резервирования для обеспечения согласованности на локальном узле. Корректировка зависит от стиля резервирования. Предположим в качестве примера, что PathTear удаляет путь для отправителя S. Если стиль задает явный выбор отправителя (FF или SE), все резервы, связанные со спецификацией фильтра, соответствующей S, следует удалить. Если стиль задает шаблонный выбор отправителя (WF), резервирование следует удалять, если S является последним отправителем для сессии. Эти изменения резервов не должны инициировать незамедлительную отправку обновления Resv, поскольку сообщение PathTear уже внесло требуемые изменения в восходящем направлении. Не следует отправлять и ResvErr, поскольку это может приводить к лавинной генерации таких сообщений.

3.1.6 Сообщения ResvTear

Получение ResvTear (демонтаж резервирования) ведет к удалению соответствующего состояния резервирования. Удаляемое состояние должно соответствовать объектам SESSION, STYLE и FILTER_SPEC, а также LIH в объекте RSVP_HOP. При отсутствии соответствующего состояния резервирования сообщение ResvTear следует отбрасывать. Сообщение ResvTear может демонтировать любое подмножество спецификации фильтров в состоянии резервирования со стилем FF или SE.

Сообщения ResvTear явно инициируются получателем или узлом, на котором для состояния возник тайм-аут, и перемещаются в восходящем направлении ко всем соответствующим отправителям.

Сообщения ResvTear должны маршрутизироваться подобно соответствующим сообщениям Resv и в качестве IP-адреса получателя в них указывается индивидуальный адрес предыдущего интервала.

```
<ResvTear Message> ::= <Common Header> [<INTEGRITY>] <SESSION> <RSVP_HOP>
                                [ <SCOPE> ] <STYLE> <flow descriptor list>

<flow descriptor list> ::= (см. определение выше)
```

Объекты FLOWSPEC в списке дескрипторов потока сообщения ResvTear будут игнорироваться и могут быть опущены. Требования к порядку для объекта INTEGRITY, дескриптора отправителя, объекта STYLE и списка дескрипторов потока приведены выше в описании сообщений Resv, однако рекомендуется использовать показанный выше порядок. Сообщение ResvTear может включать объект SCOPE, но он должен игнорироваться.

Пересылка ResvTear будет прекращаться на узле, где слияние было подавлено пересылкой соответствующего сообщения Resv. В зависимости от результирующего состояния на узле сообщение ResvTear может привести к дальнейшей его пересылке, пересылке измененного сообщения Resv или отсутствию какой-либо пересылки. Иллюстрация этих трех случаев для резервирования в стиле FF показана на рисунке 6.

- Если получатель R2 передает сообщение ResvTear для своего резерва S3{B}, соответствующее резервирование на интерфейсе (d) удаляется и сообщение ResvTear для S3{B} пересылается далее (b).
- Если получатель R1 передает сообщение ResvTear для своего резерва S1{4B}, соответствующее резервирование на интерфейсе (c) удаляется и сразу же передается измененное сообщение Resv FF(S1{3B}) (a).
- Если получатель R3 передает сообщение ResvTear для своего резерва S1{B}, эффективное резервирование для S1{3B} на интерфейсе (d) не меняется и сообщение не пересылается.

3.1.7 Сообщения PathErr

Сообщения PathErr (ошибка пути) говорят об ошибках при обработке сообщений Path. Они передаются в восходящем направлении в сторону отправителей и маршрутизируются поэтапно (hop-by-hop) с использованием состояния пути. На каждом этапе (hop) IP-адресом получателя служит индивидуальный адрес предыдущего интервала. Сообщения PathErr не меняют состояний узлов, через которые они проходят, они предназначены лишь для информирования приложения-отправителя.

```
<PathErr message> ::= <Common Header> [ <INTEGRITY> ] <SESSION> <ERROR_SPEC>
                                [ <POLICY_DATA> …] [ <sender descriptor> ]
<sender descriptor> ::= (см. определение выше)
```

Объект ERROR_SPEC указывает ошибку и включает IP-адрес обнаружившего ее узла (Error Node Address). В сообщение могут включаться объекты POLICY_DATA для предоставления относящейся к ошибке информации. Дескриптор отправителя копируется из сообщения, с которым была связана ошибка. Требования к порядку следования объектов были приведены в описании сообщений Path, но рекомендуется использовать приведенный выше порядок.

3.1.8 Сообщения ResvErr

Сообщения ResvErr (ошибка резервирования) говорят об ошибках при обработке сообщений Resv и могут также информировать о спонтанных проблемах в резервировании (например, воздействие администратора).

Сообщения ResvErr перемещаются в нисходящем направлении в сторону соответствующих получателей и маршрутизируются поэтапно с использованием состояний резервирования. На каждом этапе в качестве IP-адреса получателя служит индивидуальный адрес следующего узла.

```
<ResvErr Message> ::= <Common Header> [ <INTEGRITY> ] <SESSION>  <RSVP_HOP> <ERROR_SPEC>
                      [ <SCOPE> ] [ <POLICY_DATA> …] <STYLE> [ <error flow descriptor> ]
```

Объект ERROR_SPEC указывает ошибку и включает IP-адрес обнаружившего ее узла (Error Node Address). В сообщение об ошибке могут включаться объекты POLICY_DATA с относящейся к ошибке информацией (например, при ошибке, связанной с контролем политики). Объект RSVP_HOP содержит адрес предыдущего интервала и объект STYLE, копируемый из вызвавшего ошибку сообщения Resv. Использование объекта SCOPE в сообщениях ResvErr определено ниже в параграфе 3.4. Требования к порядку объектов приведены в описании сообщения Resv, но рекомендуется использовать показанный выше порядок.

Приведенные ниже правила, зависящие от стиля, определяют состав корректного дескриптора потока для ошибки, правила упорядочения объектов были даны выше в описании дескриптора потока.

- Стиль WF:

  ```
  <error flow descriptor> ::= <WF flow descriptor>
  ```

- Стиль FF:

  ```
  <error flow descriptor> ::= <FF flow descriptor>
  ```

  Каждый дескриптор в сообщении Resv для стиля FF должен обрабатываться незамедлительно и для каждой ошибки должно генерироваться отдельное сообщение ResvErr.

- Стиль SE:

  ```
  <error flow descriptor> ::= <SE flow descriptor>
  ```

  Сообщение ResvErr для стиля SE может содержать подмножество спецификаций фильтров, с которыми связана ошибка, из соответствующего сообщения Resv.

Отметим, что сообщение ResvErr включает единственный дескриптор потока. Следовательно, для сообщений Resv со множеством дескрипторов (стиль FF) может создаваться соответствующее число сообщений ResvErr.

В общем случае сообщение ResvErr следует пересылать в направлении всех получателей, с которыми может быть связана возникшая ошибка. Более конкретные действия приведены ниже.

- Узел, обнаруживший ошибку в запросе на резервирование, передает сообщение ResvErr следующему узлу, от которого пришло ошибочное сообщение.

  Это сообщение ResvErr должно содержать информацию, требуемую для идентификации ошибки и маршрутизации сообщений на другие узлы. Следовательно, сообщение должно включать объект ERROR_SPEC, копию объекта STYLE и подходящий дескриптор потока. Если ошибкой является отказ при контроле допуска в результате попытки расширения существующего резервирования, это резервирование должно быть сохранено, а флаг InPlace должен быть установлен в объекте ERROR_SPEC сообщения ResvErr.

- Последующие узлы пересылают сообщение ResvErr на следующие интервалы, имеющие локальное состояние резервирования. Для резервов с шаблонной областью действия имеется дополнительное ограничение на пересылку сообщений ResvErr во избежание петель (см. параграф 3.4). Существует также правило, ограничивающее пересылку сообщения Resv после отказа Admission Control (см. параграф 3.5).

  В пересылаемые сообщения ResvErr следует включать объекты FILTER_SPEC из соответствующих состояний резервирования.

- Когда сообщение ResvErr приходит получателю, объект STYLE, список дескрипторов потока и объект ERROR_SPEC (включая его флаги) следует направлять принимающему приложению.

3.1.9 Сообщения ResvConf

Сообщения ResvConf передаются для (вероятностного) подтверждения запросов на резервирование. Эти сообщения отправляются в результате присутствия объекта RESV_CONFIRM в сообщении Resv.

Сообщение ResvConf передается по индивидуальному адресу хоста-получателя, этот адрес берется из объекта RESV_CONFIRM. Однако пересылаются сообщения ResvConf получателю поэтапно с проверкой целостности на каждом этапе.

```
<ResvConf message> ::= <Common Header> [ <INTEGRITY> ] <SESSION> <ERROR_SPEC>
                       <RESV_CONFIRM> <STYLE> <flow descriptor list>

<flow descriptor list> ::= (см. определение выше)
```

Требования к порядку размещения объектов приведены выше в описании сообщений Resv, но рекомендуется использовать показанный выше порядок.

Объект RESV_CONFIRM является копией одноименного объекта из сообщения Resv, вызвавшего отправку подтверждения. ERROR_SPEC используется только для передачи IP-адреса породившего узла в поле Error Node

Address; поля Error Code и Value для подтверждений имеют нулевые значения. Список дескрипторов потока указывает конкретные резервирования, которые подтверждаются сообщением. Список может быть подмножеством списка дескрипторов потока из сообщения Resv с запросом подтверждения.

3.2 Использование портов

Обычно сессия RSVP определяется триплетом (DestAddress, ProtocolId, DstPort). Здесь DstPort обозначает номер порта UDP/TCP на стороне получателя (16-битовое значение, расположенное в транспортном заголовке со смещением 2). Значение DstPort можно опустить (установить в 0), если ProtocolId задает протокол, не имеющий поля номера порта получателя в формате, используемом UDP и TCP.

RSVP позволяет использовать любые значения ProtocolId. Однако реализации оконечных систем RSVP могут знать некоторые значения этого поля и, в частности, значения для UDP и TCP (17 и 6, соответственно). Конечная система может давать ошибку приложению в указанных ниже случаях:

- приложение задает отличное от 0 значение DstPort для протокола, который не поддерживает портов подобно UDP/TCP;
- задает DstPort = 0 для протокола, поддерживающего номера портов подобно UDP/TCP.

Спецификации фильтров и шаблоны отправителей задают пару (SrcAddress, SrcPort), где SrcPort — номер порта UDP/TCP на стороне отправителя (16-битовое поле расположенное в транспортном заголовке со смещением 0). В некоторых случаях значение SrcPort может быть опущено (установлено в 0).

Приведенные ниже правила определяют использование нулевого значения в полях DstPort и/или SrcPort для RSVP.

1. Порты получателей должны быть согласованы (Destination ports must be consistent).

   Состояния пути и резервирования для одинаковых пар DestAddress и ProtocolId должные использовать значения DstPort содержащие только 0 или не содержащие 0. При нарушении этого условия возникает ошибка Conflicting Dest Ports.

2. Правило портов получателя (Destination ports rule).

   Если в определении сессии DstPort = 0, все поля SrcPort для данной сессии также должны иметь значение 0. Это означает, что используемый протокол не имеет портов типа UDP/TCP. При нарушении этого условия возникает ошибка Bad Src Ports.

3. Порты отправителей должны быть согласованы (Source Ports must be consistent).

   Для хостов-отправителей недопустима передача состояния пути без номера порта-отправителя или с SrcPort = 0. При нарушении этого условия возникает ошибка Conflicting Sender Port.

Отметим, что RSVP не использует «шаблонных» (wildcard) портов, т. е. нулевое значение не может соответствовать отличному от 0.

3.3 Передача сообщений RSVP

Сообщения RSVP передаются поэтапно между поддерживающими RSVP маршрутизаторами, как дейтаграммы IP с номером протокола 46. Raw-дейтаграммы IP предназначены также для использования между оконечной системой и первым/последним маршрутизатором, хотя для обмена с оконечными системами можно инкапсулировать сообщения RSVP в дейтаграммы UDP, как описано в Приложении C. Инкапсуляция в UDP требуется для систем, которые не могут выполнять сетевые операции ввода-вывода для неразобранных (raw) дейтаграмм.

Сообщения Path, PathTear и ResvConf должны передаваться с IP-опцией Router Alert [RFC 2113] с заголовках IP. Эта опция может использоваться на путях быстрой пересылки в скоростных маршрутизаторах для обнаружения пакетов, требующих специальной обработки.

По прибытии RSVP-сообщения M, которое меняет состояние, узел должен незамедлительно переслать информацию о смене состояния. Однако при этом недопустима передача сообщения через интерфейс, который принял сообщение M (это может быть следствием простого инициирования реализацией незамедлительного обновления всех состояний для сессии). Это правило требуется для предотвращения пакетных лавин в широковещательных ЛВС.

В данной версии спецификации каждое сообщение RSVP должно занимать в точности одну дейтаграмму IP. Если размер дейтаграммы превышает MTU, она будет фрагментирована средствами IP и восстановлена на приемном узле. Отметим два важных аспекта:

- размер сообщения RSVP не может превосходить максимальный размер дейтаграммы IP (около 64K байт);
- в неподдерживающем RSVP облаке с перегрузкой отдельные фрагменты могут теряться, что будет приводить к потере всего сообщения.

Если проблема сохранится, она будет решена в будущих версиях протокола. Наиболее очевидным решением представляется выполнение «семантической фрагментации», т. е. разбиение передаваемого состояния пути или резервирования на множество самодостаточных сообщений приемлемого размера.

RSVP использует механизмы периодического обновления для восстановления при случайных потерях пакетов. Однако при перегрузке в сети существенные потери сообщений RSVP могут приводить к отказам в резервировании. Для контроля задержек в очередях и отбрасывания пакетов RSVP маршрутизаторы следует настраивать так, чтобы этим пакетам предоставлялось преимущественное обслуживание. Если значимое число пакетов RSVP теряется при прохождении через загруженное облако без поддержки RSVP, можно воспользоваться увеличением значений тайм-аута K (см. параграф 3.7).

Некоторые протоколы групповой маршрутизации обеспечивают поддержку multicast-туннелей, выполняя инкапсуляцию IP для групповых пакетов, передаваемых через маршрутизаторы, не поддерживающие групповых адресов. Такой туннель выглядит подобно логическому выходному интерфейсу, который отображается на некий физический

интерфейс. Протокол групповой маршрутизации, поддерживающий такие туннели, будет описывать маршрут с использованием логических интерфейсов вместо физических. RSVP может работать через такие туннели, следующим образом:

1. Когда узел N пересылает сообщение Path через логический выходной интерфейс L, он включает в сообщение некое представление L, называемое «логическим идентификатором интерфейса» (logical interface handle или LIH). Значение LIH передается в объекте RSVP_HOP.
2. Узел следующего интервала N' сохраняет значение LIH в своем состоянии пути.
3. Когда N' передает сообщение Resv узлу N, он включает значение LIH из своего состояния пути (снова в объект RSVP_HOP).
4. Когда сообщение Resv приходит на узел N, значение LIH обеспечивает информацию, требуемую для связывания резервирования с подходящим логическим интерфейсом. Отметим, что узел N создает и интерпретирует LIH, а для узла N' это значение «не прозрачно».

Отметим, что это решает лишь проблему маршрутизации для туннелей. Туннели с точки зрения RSVP выглядят, как облако без поддержки RSVP. Для организации резервирования RSVP в туннеле требуются дополнительные действия и решения, которые будут определены в будущем.

3.4 Предотвращение петель для сообщений RSVP

При пересылке сообщений RSVP должны предотвращаться петли. В установившемся состоянии сообщения Path и Resv передаются на каждом интервале только один раз за период обновления. Это предотвращает зацикливание пакетов, но еще сохраняется возможность «петель автообновления», возникающих в период обновления. Такие петли автообновления сохраняют активное состояние «навсегда», даже если оконечные узлы прекращают его обновлять, пока получатели не покинут группу или/и отправители не прекратят отправку сообщений Path. С другой стороны, сообщения об ошибках или демонтаже состояний пересылаются незамедлительно и, следовательно, могут порождать петли.

Рассмотрим все типы сообщений.

- Path

  Сообщения Path пересылаются точно так же, как пакеты данных IP. Следовательно, для сообщений Path не должно возникать петель (за исключением ситуаций с маршрутными петлями, которые мы не рассматриваем) даже в топологии с петлями.

- PathTear

  Для сообщений PathTear используется такая же маршрутизация, как для сообщений Path и, следовательно, петель не может возникать.

- PathErr

  Поскольку сообщения Path не порождают петель, они создают состояние пути, определяющее обратный путь без петель к каждому отправителю. Сообщения PathErr всегда направляются конкретным отправителям и, следовательно, не могут порождать петель.

- Resv

  Сообщения Resv направляемые конкретным отправителям (явный выбор отправителя) и не могут создавать петель. Однако сообщения Resv с шаблонным выбором отправителя (стиль WF) могут вызывать петли автообновления.

- ResvTear

  Хотя сообщения ResvTear маршрутизируются так же, как сообщения Resv, на втором проходе через петлю нужно состояния не будет и любые сообщения ResvTear будут отбрасываться. Следовательно, проблемы петель здесь не возникает.

- ResvErr

  Сообщения ResvErr для стиля WF могут создавать петли по тем же причинам, что и для сообщений Resv.

- ResvConf

  Сообщения ResvConf пересылаются в направлении фиксированных получателей с индивидуальными адресами и не могут порождать петель.

Если топология не содержит петель, зацикливания сообщений Resv и ResvErr при шаблонном выборе отправителя можно избежать просто исполняя приведенное выше правило — состояние, полученное через какой-либо интерфейс никогда не должно пересылаться через этот же интерфейс. Однако при наличии топологических петель требуются дополнительные усилия для предотвращения петель автообновления с шаблонными сообщениями Resv и быстрых петель с шаблонными сообщениями ResvErr. В данной спецификации решением этой проблемы для указанных сообщений является явная передача списка адресов отправителей в объекте SCOPE.

Когда сообщение Resv со стилем WF пересылается на конкретный предыдущий интервал, новый объект SCOPE рассчитывается на основе объектов SCOPE, которые были получены в соответствующих сообщениях Resv. Если рассчитанный объект SCOPE пуст, сообщение не пересылается на предыдущий интервал, в остальных случаях передается сообщение с новым объектом SCOPE. Правила расчета нового объекта SCOPE для сообщения Resv приведены ниже:

1. Формируется объединение набора IP-адресов отправителей из всех объектов SCOPE в состоянии резервирования для данной сессии.

Если состояние резервирования из некого NHOP не содержит объекта SCOPE, должен быть создан и включен в объединение список замены отправителей. Для сообщения, полученного через выходной интерфейс OI, такой список представляет набор отправителей, маршруты к которым идут через OI.

2. Все локальные отправители (т. е. приложения-отправители на данном узле) удаляются из набора адресов.

3. Если объект SCOPE передается PHOP, из набора удаляются все отправители, не приходящие с этого PHOP.

На рисунке 11 показан пример сообщений Resv для шаблонного выбора (стиль WF). Списки адресов в объектах SCOPE указаны в квадратных скобках. Отметим, что могут присутствовать дополнительные соединения между узлами, создающие топологические петли (не показаны на рисунке).

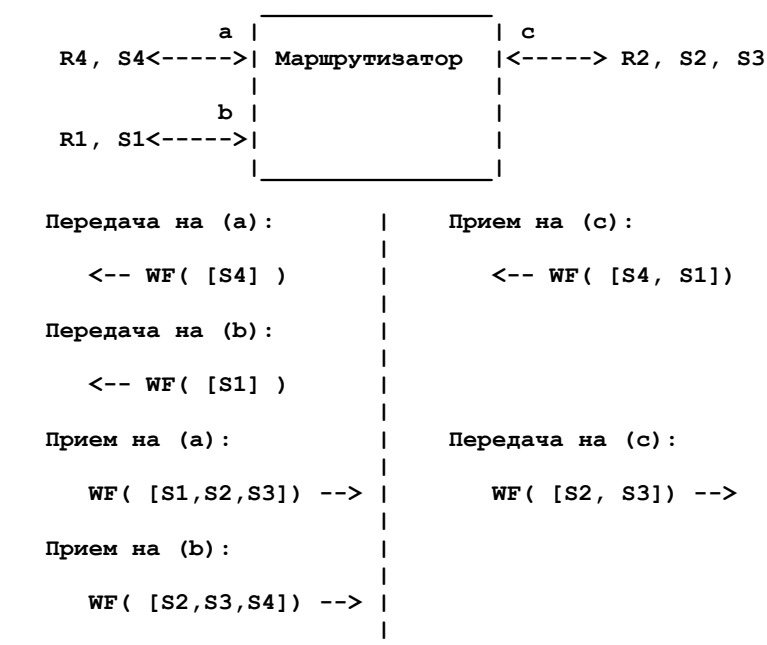


Рисунок 11. Объекты SCOPE в резервировании Wildcard-Scope

Объекты SCOPE не требуются, если групповая маршрутизация использует разделяемые деревья или стиль резервирования явно задает отправителей. Кроме того, присоединение объекта SCOPE к резервированию следует откладывать до узлов, имеющих более одного предыдущего интервала для состояния резервирования.

Ниже приведены правила, используемые для объектов SCOPE в сообщениях ResvErr при стиле резервирования WF.

1. Узел, обнаруживший ошибку, создает сообщение ResvErr, содержащее копию объекта SCOPE, который связан с состоянием резервирования или сообщением, вызвавшим ошибку.

2. Предположим, что при шаблонном выборе сообщение ResvErr прибывает на узел с объектом SCOPE, содержащим список адресов хостов-отправителей L. Узел пересылает сообщение ResvErr, используя правила параграфа 3.1.8. Однако сообщение ResvErr, пересылаемое через интерфейс OI, должно содержать объект SCOPE, созданный из L путем включения лишь тех узлов, маршруты к которым ведут через OI. Если этот объект SCOPE будет пустым, сообщение ResvErr не следует пересылать через OI.

3.5 Состояние блокады

Базовым правилом создания обновлений Resv является слияние спецификаций потоков из имеющихся на узле запросов на резервирование путем расчета LUB. Однако это правило меняется фактом существования состояний блокады (blockade state) в результате сообщений ResvErr для решения проблемы KR-II (см. параграф 2.5). Состояние блокады влияет также на маршрутизацию сообщений ResvErr для отказов в доступе (Admission Control).

При получении сообщения ResvErr для отказа Admission Control спецификация потока Qe из него используется для создания или обновления элемента локального состояния блокады. Каждый элемент состояния блокады включает спецификацию потока Qb, которая берется из сообщения ResvErr, и таймер блокады Tb. По завершении отсчета таймера соответствующее состояние блокады удаляется.

Гранулярность состояния блокады зависит от стиля сообщения ResvErr, создавшего это состояние. Для явного стиля может присутствовать элемент состояния блокады (Qb(S),Tb(S)) для каждого отправителя S. Для шаблонного стиля состояние блокады относится к предыдущему интервалу P.

Элемент состояния блокады со спецификацией потока Qb называют «блокирующим» резервирование со спецификацией Qi, если Qb (строго) не превышает Qi. Например, предположим, что LUB для двух спецификаций потоков создается путем выбора максимального значения каждой из соответствующих компонент. Тогда Qb будет блокировать Qi, если для некой компоненты j выполняется условие Qb[j] <= Qi[j].

Предположим, что узел получает сообщение ResvErr от своего предыдущего интервала P (или, при явном стиле, от отправителя S) в результате отказа Admission Control в восходящем направлении. Тогда:

1. Создается элемент состояния блокады для P (или S), если такого элемента нет.

2. Qb(P) (или Qb(S)) устанавливается равным спецификации потока Qe из сообщения ResvErr.

3. Запускается или перезапускается соответствующий таймер блокады Tb(P) (или Tb(S)) на время Kb*R. Kb является фиксированным коэффициентом, а R — интервал обновления для состояния резервирования. Значение Kb следует делать настраиваемым.

4. Если существует некое локальное состояние резервирования, которое не блокируется (см. ниже), генерируется незамедлительное обновление резервирования для P (или S).

5. Сообщение ResvErr пересылается следующим интервалам, как описано далее. Если бит InPlace сброшен, сообщение ResvErr пересылается всем следующим интервалам, для которых имеется состояние резервирования. При установленном бите InPlace сообщение ResvErr пересылается только тем следующим интервалам, для которых Qi блокируется Qb.

В заключение представим измененное правило слияния спецификаций потоков для создания сообщений об обновлении резервирования.

- Если имеются любые локальные запросы на резервирование Qi, которые не блокированы, они сливаются путем расчета LUB. Блокированные резервирования игнорируются - это позволяет пересылку более мелких резервирований, для которых не возникало отказов и которые могут оказаться выполненными после отказа для более крупного резервирования.

- В противном случае (все локальные запросы Qi заблокированы) запросы сливаются путем принятия GLB[1] из Qi (использование того или иного определения «минимума» повышает производительность за счет блокирования уровня отказов между максимальным успешным и минимальным неудачным запросом; выбор GLB, в частности, обусловлен простотой определения и реализации, поэтому нет причин искать здесь другое определение «минимума»).

Этот алгоритм обновления применяется раздельно для каждого потока (каждого отправителя или PHOP), участвующего в разделяемом резервировании (стиль WF или SE).

На рисунке 12 показан пример применения блокады для разделяемого резервирования (стиль WF). Имеется два предыдущих интервала (a) и (b), а также два следующих интервала (c) и (d). Наибольшее резервирование 4B приходит от (c) первым, но для него возникает отказ где-то в восходящем направлении через PHOP (a), при этом нет отказа для PHOP (b). Рисунок показывает финальное установившееся состояние после прибытия далее меньшего резервирования 2B от (d). Это установившееся состояние нарушается приблизительно каждые Kb*R секунд, когда таймер блокады завершает отсчет. Тогда следующее обновление передает 4B на предыдущий интервал (a); в предположении отказа отправка сообщения ResvErr будет восстанавливать блокаду, возвращая показанную на рисунке ситуацию. В то же время сообщение ResvErr будет пересылаться на следующий интервал (c) и всем получателям нисходящего направления, связанным с резервированием 4B.

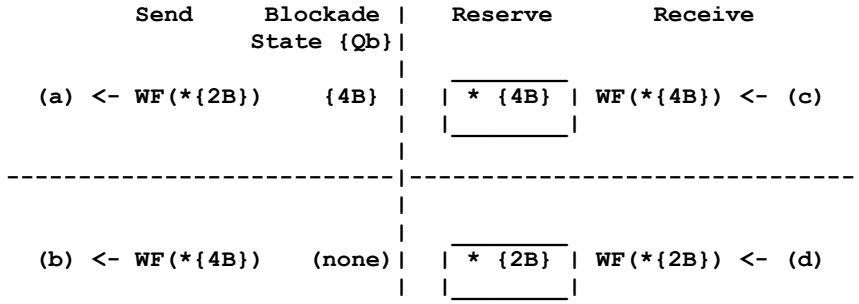


Рисунок 12. Блокада при разделяемом стиле.

3.6 Локальные изменения

Когда маршрут изменяется, следующее сообщение Path или Resv будет создавать новое состояние пути или резервирования (соответственно) вдоль нового маршрута. Для обеспечения быстрой адаптации к изменениям маршрутов без значительных издержек в короткие периоды обновления RSVP обрабатывает изменения маршрутов для конкретных получателей. Процессу RSVP следует использовать эту информацию для включения быстрого обновления состояния для тех получателей, которые используют новый маршрут.

Конкретные правила перечислены ниже:

- Когда маршрутизация детектирует изменение набора выходных интерфейсов для адресата G, RSVP следует обновить состояние пути, выждать короткий интервал W и после этого отправить сообщения Path для всех сессий G/* (т. е. для любой сессии с адресатом G, независимо от порта адресата).

  Короткий период ожидания перед отправкой обновлений Path позволяет дождаться схождения протокола маршрутизации и значение W следует подбирать с учетом этого. В настоящее время предлагается значение W = 2 с; это значение следует делать настраиваемым на уровне интерфейса.

- Когда приходит сообщение Path с адресом Previous Hop, который отличается от сохраненного в состоянии пути, RSVP следует незамедлительно передать обновления Resv для этого PHOP.

3.7 Временные параметры

Есть два временных параметра, относящихся к каждому элементу состояния пути или резервирования RSVP на узле - период обновления R (интервал между генерацией двух последовательных обновления состояния соседним узлом) и время жизни локального состояния L. Каждое сообщение RSVP Resv или Path может включать объект TIME_VALUES, задающий значение R, которое было использовано при генерации этого (обновляющего) сообщения. Это значение R далее используется для определения значения L, когда состояние принято и сохранено. Значения R и L могут меняться на разных интервалах.

[1]Greatest Lower Bound — наибольшая нижняя граница.

Ниже работа с временными параметрами описана более подробно.

1. Floyd и Jacobson [FJ94] показали, что периодические сообщения, генерируемые независимыми сетевыми узлами, могут оказаться синхронизированными. Это может приводить к нарушению работы сетевых служб, поскольку периодические сообщения будут бороться с остальным трафиком за ресурсы канала и пересылки. Поскольку RSVP передает периодические обновления, этот протокол должен избегать синхронизации сообщений и обеспечивать нестабильность любой возникающей синхронизации.

   По этой причине для таймера обновления следует выбирать случайное значение из диапазона [0,5R, 1,5R].

2. Для предотвращения преждевременной потери состояния значение L должно удовлетворять условию L >= (K + 0,5)*1,5*R, где K — небольшое целое число. Тогда в худшем случае может быть потеряно K-1 сообщений подряд без удаления состояния. При расчете времени жизни L для набора состояний с различными значениями R (R0, R1, …) вместо R следует использовать max(Ri).

   В настоящее время предлагается по умолчанию использовать K = 3. Однако для интервалов с высокими потерями может потребоваться установка большего значения K. Значение K можно задавать вручную при настройке интерфейса или использовать тот или иной адаптивный механизм, который пока не включен в спецификацию.

3. Каждое сообщение Path или Resv переносит объект TIME_VALUES содержащий параметр R, используемый для генерации обновлений. Принимающий узел использует это значение R для определения времени жизни сохраненного состояния (L), создаваемого или обновляемого этим сообщением.

4. Время обновления R выбирается локально каждым узлом. Если узел не реализует локального восстановления резервирований, поврежденных изменениями маршрутов, меньшее значение R ускоряет адаптацию к изменениям маршрутов, но увеличивает издержки RSVP. При локальном восстановлении маршрутизатор может обращать меньше внимания на R, поскольку периодическое обновление становится лишь механизмом надежности. Узел, следовательно, может подстраивать эффективное значение R динамически для контроля уровня издержек, связанных с обновлениями.

   Предлагаемое значение R для использования по умолчанию составляет 30 секунд. Однако используемое по умолчанию значение Rdef следует делать настраиваемым на уровне интерфейса.

5. При динамическом изменении R имеется предел скорости роста этого значения. В частности, отношение двух последовательных значений R2/R1 не должно превышать 1 + Slew.Max.

   В настоящее время принято Slew.Max = 0.30. При K = 3 один пакет может быть потерян без возникновения тайм-аута, когда R увеличивается на 30% за цикл обновления.

6. Для повышения устойчивости узел может временно передавать обновления после смены состояния (включая изначальную организацию) чаще, чем задает R.

7. Значения Rdef, K и Slew.Max, используемые реализацией, следует делать легко изменяемыми на уровне интерфейса, поскольку опыт может приводить к изменению их значений. Возможность динамической адаптации K и/или Slew.Max к измеренным значениям потерь требует дополнительного изучения.

3.8 Управление трафиком на интервалах без интегрированных услуг

Некоторые службы управления QoS могут требовать правил для трафика на некоторых или всех (1) краях сети, (2) точках слияния для множества отправителей и/или (3) точках ветвления, где трафик из восходящего направления может превышать запрашиваемое резервирование в нисходящем направлении. RSVP знает о таких точках и должен указывать их механизму управления трафиком. С другой стороны, RSVP не интерпретирует сервис, указанный в спецификации потока, и, следовательно, не знает где правила будут реально применяться в конкретной ситуации.

Процесс RSVP передает системе управления трафиком отдельный флаг для каждой из трех указанных ниже ситуаций.

- E_Police_Flag - входные правила

  Этот флаг устанавливается на узле первого интервала RSVP, который реализует управление трафиком (и, следовательно, способен применить правила).

  Например, передающий хост должен реализовать RSVP, но в настоящее время многие из таких хостов не поддерживают управления трафиком. В этом случае флаг E_Police_Flag на хосте-отправителе следует сбрасывать, устанавливая его только на первом узле, способном управлять трафиком. Это контролируется с помощью флага E_Police в объектах SESSION.

- M_Police_Flag - правила слияния

  Этот флаг следует устанавливать для резервирования с разделяемым стилем (WF или SE), когда сливаются потоки от множества отправителей.

- B_Police_Flag - правила ветвления

  Этот флаг следует устанавливать в тех случаях, когда устанавливаемая спецификация потока меньше или несовместима с имеющейся FLOWSPEC на любом другом сетевом интерфейсе для тех же FILTER_SPEC и SESSION.

RSVP должен также проверять присутствие интервалов без поддержки RSVP и передавать эту информацию системе управления трафиком. На основании флага, полученного от процесса RSVP, и своей собственной информации система управления трафиком может определить присутствие на пути интервала, не поддерживающего управление QoS, и передать эти данные получателям в объектах adspec [RFC 2210].

При обычной пересылке IP протокол RSVP может обнаружить интервалы без поддержки RSVP путем сравнения IP TTL, с которым передано сообщение Path и TTL, с которым оно было принято - для этого значение TTL передается в общем заголовке. Однако TTL не всегда является надежным индикатором интервалов без поддержки RSVP и порой приходиться пользоваться другими методами. Например, если протокол маршрутизации применяет туннели с

инкапсуляцией IP, этот протокол должен информировать RSVP о наличии интервалов без поддержки RSVP. Если автоматический механизм не будет работать, потребуется ручная настройка конфигурации.

3.9 Многодомные хосты

Поддержка многодомных хостов в RSVP требует выполнения некоторых специальных правил. Термин «многодомный хост» здесь означает как собственно хосты (оконечные системы) с множеством сетевых интерфейсов, так и маршрутизаторы, поддерживающие локальные прикладные программы.

Выполняющееся на многодомном хосте приложение может явно указать сетевой интерфейс, который оно желает использовать для передачи и/или приема пакетов данных, вместо принятого по умолчанию в данной системе интерфейса. Процесс RSVP должен знать используемый по умолчанию интерфейс и, если приложение задает другой интерфейс, оно должно передать информацию об этом процессу RSVP.

- Передача данных

  Передающее приложение использует вызов API (SENDER, параграф 3.11.1) для информирования RSVP о характеристиках порождаемого им потока данных. Этот вызов может включать локальный IP-адрес отправителя. Если приложение устанавливает этот параметр, он должен соответствовать адресу используемого для передачи пакетов интерфейса - в противном случае будет использоваться принятый по умолчанию интерфейс.

  Процесс RSVP на хосте после этого передает сообщения Path для данного приложения (только) через заданный интерфейс.

- Организация резервирования

  Принимающее приложение использует вызов API (RESERVE, параграф 3.11.1) для запроса резервирования от RSVP. Этот вызов включать локальный IP-адрес получателя, через который тот желает получать пакеты. В случае групповых сессий это будет интерфейс, через который выполняется подключение к группе. Если параметр опущен, используется принятый по умолчанию в системе интерфейс.

- В общем случае процессу RSVP следует передавать сообщения Resv для приложения через указанный интерфейс. Однако, если приложение выполняется на маршрутизаторе и сессия является групповой, возникает более сложная ситуация. Предположим, что в таком случае принимающее приложение присоединяется к группе через интерфейс Iapp, который отличается от интерфейса Isp, обеспечивающего кратчайший путь к отправителю. В этом случае существует два варианта групповой маршрутизации для доставки пакетов данных приложению. Процесс RSVP должен выбрать один из этих вариантов, проверяя состояние пути, и использовать соответствующий интерфейс для передачи сообщений Resv.

  1. Протокол групповой маршрутизации может создавать отдельную ветвь группового распространения «дерева» для доставки на интерфейс Iapp. В этом случае будут существовать состояния пути для обоих интерфейсов Isp и Iapp. Состояние пути на Iapp будет соответствовать только резервированию от локального приложения и должно маркироваться процессом RSVP, как Local_only. Если состояние пути Local_only для интерфейса Iapp существует, сообщения Resv следует передавать через этот интерфейс.

     Отметим, что возможна блокировка состояний пути для Isp и Iapp на один следующий интервал, если нет облака без поддержки RSVP.

  2. Протокол групповой маршрутизации может пересылать данные внутри маршрутизатора с интерфеса Isp на Iapp. В этом случае интерфейс Iapp будет появляться в списке выходных интерфейсов состояния пути для Isp и сообщение Resv следует передавать через Isp.

  3. При пересылке сообщений Path и PathTear состояние пути Local_Only должно игнорироваться.

3.10 Совместимость с будущими версиями

Можно предположить определение в будущем новых объектов C-Type для существующих классов объектов, а возможно и определение новых классов. Желательно будет разворачивать такие объекты в Internet с использованием старых реализаций, которые не смогут распознавать эти объекты. К сожалению, такое возможно лишь с существенными ограничениями, обусловленными сложностью. Правила приведены ниже (b означает бит).

1. Неизвестный класс

   Существует три возможных варианта трактовки объектов неизвестного класса реализацией RSVP. Выбор варианта определяется двумя старшими битами октета Class-Num.

   - Class-Num = 0bbbbbbb

     Сообщение целиком следует отвергать с возвратом ошибки Unknown Object Class.

   - Class-Num = 10bbbbbb

     Узлу следует игнорировать объект, не пересылая его и не возвращая ошибки.

   - Class-Num = 11bbbbbb

     Узлу следует игнорировать объект, но пересылать его без проверки и изменения во всех сообщениях, порожденных данным сообщением.

   - Ниже приведены более детальные правила для объектов неизвестного класса с Class-Num = 11bbbbbb:

     1. Такие объекты неизвестного класса, полученные в сообщениях PathTear, ResvTear, PathErr, ResvErr, следует пересылать незамедлительно в тех же сообщениях.

     2. Такие объекты неизвестного класса, полученные в сообщениях Path или Resv, следует сохранять с соответствующим состоянием и пересылать во всех обновлениях, исходящих от этого состояния.

3. Когда обновление Resv генерируется путем слияния множества запросов на резервирование, в обновляющее сообщение следует включать объединение (union) объектов неизвестного класса из запросов-компонент. В объединение следует включать только одну копию каждого уникального объекта неизвестного класса.

4. Исходный порядок объектов неизвестного класса сохранять не требуется; однако пересылаемые сообщения должны соответствовать общим требованиям к порядку объектов для данного типа.

Хотя объекты неизвестного класса не могут быть слиты, эти правила позволяют пересылать такие объекты до тех пор, пока ини не достигнут узла, знающего, как их слить воедино. Пересылка объектов с неизвестным классом позволяет разворачивать новые объекты постепенно, однако следует внимательно изучить пределы масштабирования до развертывания новых объектов с установленными старшими битами.

2. Неизвестный тип C-Type для известного класса

Можно предположить, что известный номер класса Class-Num обеспечит информацию, которая позволит интеллектуальную обработку объекта. Однако на практике такая обработка в зависимости от класса сложна, а во многих случаях бесполезна.

В общем случае появление объекта с неизвестным C-Type должно приводить к отбрасыванию сообщения целиком и генерации сообщения об ошибке (ResvErr или PathErr, по ситуации). Сообщение об ошибке будет включать значения Class-Num и C-Type, приведшие к отказу (см. Приложение B); конечная система, которая породила вызвавшее отказ сообщение может быть способна воспользоваться этой информацией для повтора запроса с использованием другого объекта C-Type пока не будет достигнут успех или исчерпаны все варианты.

Объекты некоторых классов (FLOWSPEC, ADSPEC и POLICY_DATA) непрозрачны для RSVP, который просто передает их модулям контроля трафика или исполнения политики. В зависимости от внутренних правил любой из этих модулей может отвергнуть C-Type и информировать процесс RSVP, которому в этом случае следует отвергнуть сообщение и передать сообщение об ошибке, как описано в предыдущем абзаце.

3.11 Интерфейсы RSVP

RSVP на маршрутизаторе имеет интерфейсы для маршрутизации и контроля трафика. RSVP на хосте имеет интерфейс с приложениями (API), а также интерфейс для контроля трафика (если он поддерживается на хосте).

3.11.1 Прикладной интерфейс RSVP

В этом параграфе описан базовый интерфейс между приложением и управляющим процессом RSV. Детали реального интерфейса могут зависеть от операционной системы и приведенные ниже описания предлагают лишь базовые функции. Некоторые из этих вызовов приводят к асинхронному возврату информации.

- Регистрация сессии

```
SESSION(DestAddress, ProtocolId, DstPort [, SESSION_object ] [, Upcall_Proc_addr ])
        -> Session-id
```

Этот вызов инициирует обработку RSVP для сессии, определнную DestAddress вместе с ProtocolId и, возможно, номером порта DstPort. При успешном завершении SESSION незамедлительно возвращает локальный идентификатор сессии Session-id, который может использоваться в последующих вызовах.

Параметр Upcall_Proc_addr определяет адрес процедуры anupcall для получения асинхронных уведомлений о событиях и ошибках. Параметр SESSION_object включен, как механизм обхода для поддержки некого более общего определения сессии (обобщенный порт получателя), которое может потребоваться в будущем. Обычно параметр SESSION_object можно опустить.

- Определение отправителя

```
SENDER(Session-id [, Source_Address ] [, Source_Port ] [, Sender_Template ]
      [, Sender_Tspec ] [, Adspec ] [, Data_TTL ] [, Policy_data ])
```

Отправитель использует этот вызов для задания или изменения атрибутов потока данных. Первый вызов SENDER, зарегистрированной как Session-id будет инициировать передачу RSVP сообщений Path для данной сессии, последующие вызовы могут менять информацию о пути.

Интерпретация параметров SENDER приведена ниже:

- Source_Address

  Это адрес интерфейса, через который будут передаваться данные. При отсутствии параметра будет использоваться принятый по умолчанию интерфейс. Параметр требуется только для многодомных хостов-отправителей.

- Source_Port

  Этот параметр задает номер порта UDP/TCP, через который будут передаваться данные.

- Sender_Template

  Этот параметр включен, как механизм обхода для поддержки более общего определения отправителя (обобщенный порт отправителя. Обычно этот параметр может быть опущен.

- Sender_Tspec

  Этот параметр определяет поток трафика, который будет передаваться - см. [RFC 2210].

- Adspec

  Этот параметр может быть задан для инициализации расчета параметров QoS на пути - см. [RFC 2210].

- Data_TTL

  Этот параметр задает время жизни IP (отличное от TTL по умолчанию), которое будет устанавливаться для пакетов данных. Это требуется для того, чтобы сообщения Path не уходили по сети дальше групповых пакетов данных.

- Policy_data

  Этот необязательный параметр передает данные политики для отправителя. Данные могут предоставляться системной службой и трактоваться приложением, как непрозрачные.

- Резервирование

```
RESERVE(session-id, [receiver_address, ] [CONF_flag, ] [Policy_data,]
        style, style-dependent-parms)
```

Получатель использует этот вызов для организации или изменения резервирования ресурсов в сессии, зарегистрированной с session-id. Первый вызов RESERVE будет инициировать периодическую передачу сообщений Resv. Последующий вызов RESERVE может служить для изменения параметров, заданных предшествующим вызовом (следует отметить, что изменение имеющегося резервирования может приводить к отказам контроля доступа).

Необязательный параметр receiver_address может использоваться получателем на многодомном хосте (или маршрутизаторе) для указания IP-адреса одного из своих интерфейсов. Флаг CONF_flag следует устанавливать, если желательно получить подтверждение резервирования. Параметр Policy_data задает данные политики для получателя, а параметр style указывает стиль резервирования. Остальные параметры зависят от стиля и обычно включают спецификации потока и фильтров.

RESERVE возвращает результат незамедлительно. При последующих вызовах RESERVE в любой момент могут происходить асинхронные вызовы ERROR/EVENT.

- Освобождение

```
RELEASE( session-id )
```

Этот вызов удаляет состояние RSVP для сессии, заданной session-id. После этого узел передает соответствующие сообщения teardown и прекращает передачу обновлений для данного session-id.

- Ошибки и события

Общая форма вызовов имеет вид:

```
<Upcall_Proc>( ) -> session-id, Info_type, information_parameters
```

Upcall_Proc обозначает upcall-процедуру, адрес которой был задан при вызове SESSION. Обращение к такой процедуре может выполняться асинхронно после вызова SESSION (вплоть до вызова RELEASE) для индикации ошибки или события.

В настоящее время определено 5 типов upcall, различающихся параметром Info_type. Выбор информационных определяется типом вызова.

1. Info_type = PATH_EVENT

   Вызов Path Event обусловлен получением первого сообщения Path для данной сессии, показывающего приложению-получателю наличие по крайней мере одного активного отправителя, или изменением состояния пути.

```
<Upcall_Proc>( ) -> session-id, Info_type=PATH_EVENT, Sender_Tspec,
                    Sender_Template [, Adspec ] [, Policy_data ]
```

   В этом вызове указываются Sender_Tspec, Sender_Template, Adspec и все данные политики из сообщения Path.

2. Info_type = RESV_EVENT

   Вызов Resv Event инициируется приемом первого сообщения RESV или изменением состояния резервирования для данной сессии.

```
<Upcall_Proc>( ) -> session-id, Info_type=RESV_EVENT, Style, Flowspec,
                    Filter_Spec_list [, Policy_data ]
```

   Flowspec может указывать эффективные параметры QoS. Отметим, что сообщение Resv для стиля FF может приводить к множеству вызовов RESV_EVENT (для каждого дескриптора).

3. Info_type = PATH_ERROR

   Событие Path Error указывает на ошибку в информации отправителя, который был задан при вызове SENDER.

```
<Upcall_Proc>( ) -> session-id, Info_type=PATH_ERROR, Error_code,
                    Error_value, Error_Node, Sender_Template
                    [, Policy_data_list ]
```

   Парметр Error_code будет определять ошибку, а Error_value может содержать дополнительные данные о ней (возможно, зависящие от системы). Параметр Error_Node будет задавать IP-адрес обнаружившего ошибку узла. При наличии параметра Policy_data_list он будет содержать все объекты POLICY_DATA из связанного с отказом сообщения Path.

4. Info_type = RESV_ERR

   Событие Resv Error говорит об ошибке в сообщении для резервирования приложению, которое в нем участвует.

```
<Upcall_Proc>( ) -> session-id, Info_type=RESV_ERROR, Error_code,
```

Error_value, Error_Node, Error_flags, Flowspec, Filter_spec_list [, Policy_data_list]

Параметр Error_code будет определять ошибку, а Error_value может содержать некие дополнительные данные (возможно, зависящие от системы). Параметр Error_Node будет указывать IP-адрес узла, обнаружившего событие.

Существует два флага Error_flags:

- InPlace

  Этот флаг может устанавливаться для отказов Admission Control с целью указания наличия и сохранения резервирования на связанном с отказом узле. Флаг устанавливается в точке отказа и пересылается в сообщениях ResvErr.

- NotGuilty

  Этот флаг может устанавливаться для отказов Admission Control с целью показать, что спецификация flowspec, запрошенная данным получателем, существенно меньше flowspec, вызвавшей ошибку. Флаг устанавливается API получателя.

  Filter_spec_list и Flowspec будут содержать соответствующие объекты из дескриптора потока ошибок (см. параграф 3.1.8). List_count будет указывать число FILTER_SPECS в Filter_spec_list. Параметр Policy_data_list будет содержать все объекты POLICY_DATA из сообщения ResvErr.

5. Info_type = RESV_CONFIRM

   Событие Confirmation показывает, что сообщение ResvConf было получено.

   <Upcall_Proc>( ) -> session-id, Info_type=RESV_CONFIRM, Style, List_count, Flowspec, Filter_spec_list [, Policy_data]

   Эти параметры интерпретируются так же, как параметры Resv Error.

Хотя сообщения RSVP, указывающие события path или resv, могут приниматься периодически, API следует выполнять соответствующие асинхронные upcall-вызовы приложения только для первого сообщения или при изменении информации в таких сообщениях. Все сообщения, связанные с ошибками или подтверждениями, следует передавать приложению.

3.11.2 Интерфейс управления трафиком RSVP

Базовый интерфейс управления трафиком организовать достаточно сложно, поскольку детали резервирования ресурсов сильно зависят от технологии канального уровня, используемой на интерфейсе.

Требуется обеспечить слияние резервирований RSVP, поскольку при групповой доставке данных пакеты реплицируются для передачи разным узлам следующего интервала (next-hop). В каждой такой точке репликации протокол RSVP должен согласовывать запросы на резервирование от соответствующих узлов next-hop, рассчитывая «максимум» для своих потоков данных. На хосте или маршрутизаторе может использоваться одна или несколько точек репликации, описанных ниже.

1. Уровень IP

   Групповая пересылка IP выполняется путем репликации на уровне IP. В этом случае протокол RSVP должен объединить резервирования на соответствующих выходных интерфейсах для того, чтобы переслать запрошенный поток (upstream).

2. «Сеть»

   Репликация может происходить в нисходящем от узла направлении (например, в широковещательной ЛВС, коммутаторе канального уровня или сети не поддерживающих RSVP маршрутизаторов (параграф 2.8)). В таких случаях протокол RSVP должен слить резервирования от разных узлов next-hop для резервирования на единственном выходном интерфейсе. Он должен также объединить запросы на резервирование для пересылки в восходящем направлении.

3. Драйвер канального уровня

   Для технологий с множественным доступом репликация может происходить в драйвере канального уровня или интерфейсном модуле. Например, такая ситуация может возникать при наличии раздельных соединений «точка-точка» ATM VC в направлении каждого узла next-hop. От протокола RSVP может потребоваться независимое управление трафиком для каждого VC без слияния запросов от разных узлов next-hop.

В общем случае эти сложности не влияют на протокольную обработку, требуемую RSVP, за исключением точного определения запросов на резервирование, которые требуют слияния. Может оказаться желательной реализация RSVP в виде двух частей - ядра, выполняющего независимую от канального уровня обработку, и модуля адаптации к канальному уровню. Однако здесь представлен базовый интерфейс, который предполагает, что репликация может происходить только на уровне IP или «в сети».

- Резервирование

  TC_AddFlowspec(Interface, TC_Flowspec, TC_Tspec, TC_Adspec, Police_Flags) -> RHandle [, Fwd_Flowspec]

  Параметр TC_Flowspec определяет желаемое эффективное качество обслуживания (QoS) для контроля доступа; значение определяется, как макимальное из flowspec для различных следующих интервалов (см. Compare_Flowspecs ниже). Параметр TC_Tspec определяет эффективное значение Tspec Path_Te для отправителя (см. параграф 2.2). Параметр TC_Adspec определяет эффективное значение Adspec. Параметр Police_Flags передает 3 флага - E_Police_Flag, M_Police_Flag и B_Police_Flag (см. параграф 3.8).

При успешном завершении вызова организуется новый канал, соответствующий RHandle, в противном случае возвращается код ошибки. «Непрозрачное» число RHandle используется вызывающей стороной для последующих ссылок на данное резервирование. Если служба контроля трафика обновляет flowspec, вызов будет также возвращать обновленный объект в Fwd_Flowspec.

- Изменение резервирования

  TC_ModFlowspec(Interface, RHandle, TC_Flowspec, TC_Tspec, TC_Adspec, Police_flags) [-> Fwd_Flowspec]
  Этот вызов служит для изменения имеющегося резервирования. Значение TC_Flowspec передается Admission Control – если новый запрос отвергается, остается в силе текущая спецификация flowspec. Если имеются соответствующие спецификации фильтров, они остаются без изменений. Отсальные параметры определяются так же, как при вызове TC_AddFlowspec. Если служба обновит flowspec, результатом вызова будет обновленный объект в Fwd_Flowspec.

- Удаление спецификации потока

  TC_DelFlowspec(Interface, RHandle)
  Этот вызов служит для удаления существующего резервирования, включая flowspec и все связанные спецификации фильтров.

- Добавление спецификации фильтра

  TC_AddFilter(Interface, RHandle, Session , FilterSpec) -> FHandle
  Этот вызов используется для связывания дополнительной спецификации фильтров с резервированием, заданным параметром RHandle, после успешного вызова TC_AddFlowspec. Вызов возвращает идентификатор фильтра FHandle.

- Удаление спецификации фильтра

  TC_DelFilter(Interface, FHandle)
  Этот вызов служит для удаления фильтра, заданного FHandle.

- Обновление OPWA

  TC_Advertise(Interface, Adspec, Non_RSVP_Hop_flag) -> New_Adspec
  Этот вызов используется для OPWA чтобы рассчитать исходящие анонсы New_Adspec для указанного интерфейса. Бит Non_RSVP_Hop_flag следует устанавливать, если демон RSVP обнаружил, что предыдущий интервал RSVP включает хотя бы один маршрутизатор, не поддерживающий RSVP. TC_Advertise будет включать эту информацию в New_Adspec для индикации обнаружения интервала без интегрированных услуг (см. параграф 3.8).

- Preemption Upcall

  TC_Preempt() -> RHandle, Reason_code
  Для обеспечения нового запроса на резервирование модули контроля доступа и/или политики могут перенимать один или несколько существующих резервов. При этом будет инициирован вызов TC_Preempt() для каждого перенимаемого резерва с передачей идентификатора резерва RHandle и субкода причины.

3.11.3 Интерфейс RSVP - управление политикой

Этот интерфейс будет определен в будущем документе.

3.11.4 Интерфейс RSVP - маршрутизация

Реализации RSVP требуется поддержка механизмами маршрутизации узла указанных ниже функций.

- Route Query - запрос маршрута

  Для пересылки сообщений Path и PathTear процесс RSVP должен иметь возможность запроса маршрутов у процесса(ов) маршрутизации.

  Ucast_Route_Query([SrcAddress,] DestAddress, Notify_flag) -> OutInterface
  Mcast_Route_Query([SrcAddress,] DestAddress, Notify_flag) -> [IncInterface,] OutInterface_list
  В зависимости от протокола маршрутизации запрос может зависеть от SrcAddress, т. е. IP-адреса хоста отправителя, который также является адресом источника сообщения. IncInterface указывает адрес интерфеса, через который предполагается прибытие пакета (некоторые протоколы групповой маршрутизации могут не возвращать этот адрес). Если установлен флаг Notify_flag, система маршрутизации будет сохранять состояние, требуемое для обратных вызовов при незапрошенном изменении заданного маршрута (см. ниже).

  Запрос группового маршрута может возвращать пустое значение OutInterface_list при отсутствии получателей в нисходящем направлении от конкретного маршрутизатора. Запрос маршрута может также возвращать ошибку No such route (нет такого маршрута), которая может быть связана с временной несогласованностью картины маршрутизации (поскольку сообщение Path или PathTear для запрошенного маршрута прибыло на данный узел). В любом случае локальное состояние следует обновлять в соответствии с запросом в сообщении, которое невозможно переслать дальше. Обновление локального состояния незамедлительно сделает состояние пути доступным для нового локального получателя или незамедлительно демонтирует состояние пути.

- Route Change Notification - уведомление о смене маршрута

  При запросе маршрута с флагом Notify_flag процесс маршрутизации может обеспечивать асинхронные вызовы (callback) процесса RSVP в случае изменения указанного маршрута.

  Ucast_Route_Change( ) -> [SrcAddress,] DestAddress, OutInterface

```
Mcast_Route_Change( ) -> [ SrcAddress, ] DestAddress, [ IncInterface, ]
OutInterface_list
```

- Interface List Discovery - обнаружение списка интерфейсов

  RSVP должен иметь возможность узнавать об активных интерфейсах (реальных и виртуальных) с адресами IP.

  Следует обеспечивать возможность логического запрета интерфейса для RSVP. Когда интерфейс запрещен для RSVP, сообщения Path не следует передавать через этот интерфейс, а при получении через него сообщения RSVP такое сообщение следует отбрасывать без уведомления (возможно, с записью в журнал).

3.11.5 Интерфейс ввода-вывода пакетов RSVP

Реализации RSVP требуется поддержка перечисленных ниже операций от механизмов ввода-вывода и пересылки пакетов на узле.

- Режим приема всех сообщений RSVP

  Пакеты, полученные для протокола IP с номером 46, но не адресованные данному узлу, должны доставляться программе RSVP для обработки без их пересылки. Сообщения RSVP, к которым это относится, включают Path, PathTear и ResvConf. Эти типы сообщений содержат опцию IP Router Alert, которая может применяться для их захвата из пути высокоскоростной пересылки. В качестве другого варианта узел может перехватывать все пакеты протокола 46.

  На маршрутизаторах и многодомных хостах процессу RSVP должны быть доступны также идентификация интерфейсов (физических или виртуальных), через которые были приняты перехватываемые сообщения, а также IP-адреса источника и значения IP TTL, с которыми поступили пакеты.

- Задание выходного канала

  RSVP должен иметь возможность задания передачи (групповых) дейтаграмм через указанный физический или виртуальный канал в обход обычных механизмов маршрутизации. Виртуальный канал может быть, например, multicast-туннелем. Задание выходного канала требуется для передачи разных версий исходящих сообщений Path через разные выходные интерфейсы и (в некоторых случаях) для предотвращения маршрутных петель.

- Задание адреса отправителя и TTL

  RSVP должен иметь возможность задавать IP-адрес отправителя и IP TTL при передаче сообщений Path.

- Опция Router Alert

  RSVP должен иметь возможность передачи сообщений Path, PathTear и ResvConf с опцией IP Router Alert.

3.11.6 Зависящие от сервиса действия

Объекты Flowspec, Tspec и Adspec непрозрачны для RSVP и их содержимое определяется спецификациями сервиса. Для работы с такими объектами процесс RSVP должен иметь возможность выполнения перечисленных ниже операций, зависимых от сервиса.

- Сравнение Flowspec

  Compare_Flowspecs(Flowspec_1, Flowspec_2) → result_code
  Возможные значения result_code включают: совпадение flowspec, Flowspec_1 больше, Flowspec_2 больше, flowspec не совместимы, но можно рассчитать LUB, flowspec не совместимы.

  Отметим, что при сравнении двух flowspec неявно сравниваются содержащиеся в них Tspec. Хотя процесс RSVP сам по себе не способен разбирать flowspe для выделения Tspec, он может воспользоваться вызовом Compare_Flowspecs для неявного расчета Resv_Te (см. параграф 2.2).

- Расчет LUB для Flowspec

  LUB_of_Flowspecs(Flowspec_1, Flowspec_2) → Flowspec_LUB
- Расчте GLB для Flowspec

  GLB_of_Flowspecs(Flowspec_1, Flowspec_2) → Flowspec_GLB
- Сравнение Tspec

  Compare_Tspecs(Tspec_1, Tspec_2) -> result_code
  Возможные коды результата result_code включают: Tspec равны, Tspec не равны.

- Суммирование Tspec

  Sum_Tspecs(Tspec_1, Tspec_2) -> Tspec_sum
  Этот вызов может применяться для расчета Path_Te (см параграф 2.2).

4. Благодарности

Протокол RSVP основан на исследованиях, выполненных в 1992-1993 группой специалистов, включающей Lixia Zhang (UCLA), Deborah Estrin (USC/ISI), Scott Shenker (Xerox PARC), Sugih Jamin (USC/Xerox PARC), Daniel Zappala (USC). Sugih Jamin разработал первый прототип реализации RSVP и продемонстрировал его в мае 1993. Shai Herzog и, позднее, Steve Berson продолжили разработку прототипов RSVP.

С 1993 множество участников исследовательского сообщества Internet внесли свой вклад в разработку и развитие RSVP. В число участников этих процессов входят (в алфавитном порядке) Steve Berson, Bob Braden, Lee Breslau, Dave Clark, Deborah Estrin, Shai Herzog, Craig Partridge, Scott Shenker, John Wroclawski, Daniel Zappala, Lixia Zhang. Кроме того, многие производители хостов и маршрутизаторов внесли существенный вклад в документирование RSVP - Fred Baker (Cisco), Mark Baugher (Intel), Lou Berger (Fore Systems), Don Hoffman (Sun), Steve Jakowski (NetManage), John Krawczyk (Bay Networks), Bill Nowicki (SGI), а также многие другие.

Приложение A. Определения объектов

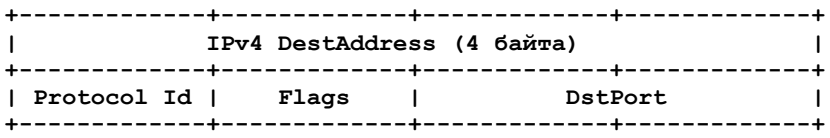
Значения C-Type определены для двух семейств адресов - IPv4 и IPv6. Легко могут быть определены C-Type и для других семейств адресов. Эти определения приведены в приложениях для простоты их обновления.

Для всех неиспользуемых полей следует устанавливать нулевые значения и игнорировать их на приемной стороне.

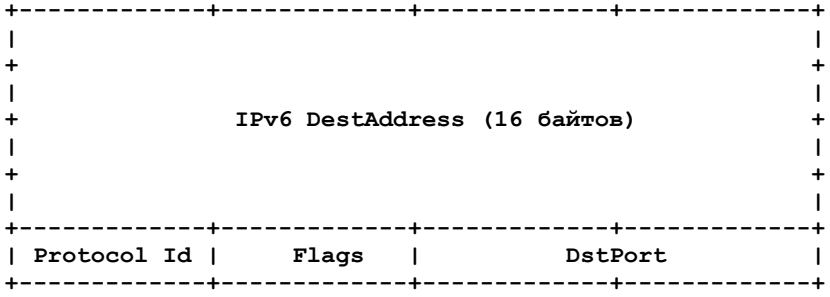
A.1 Класс SESSION

SESSION = 1

- Объект IPv4/UDP SESSION: Class = 1, C-Type = 1



- Объект IPv6/UDP SESSION: Class = 1, C-Type = 2



DestAddress

Индивидуальный или групповой IP-адрес получателя для сессии. Поле должно быть отлично от 0.

Protocol Id

Идентификатор протокола IP для потока данных. Поле должно быть отлично от 0.

Flags

0x01 = E_Police flag

Флаг E_Police используется в сообщениях Path для указания эффективного «края» сети при контроле трафика. Если хост-отправитель не может сам контролировать трафик, он будет устанавливать данный флаг в передаваемых сообщениях Path. Первый узел RSVP, способный управлять трафиком, будет реализовать такое управление (если оно подходит для сервиса) и сбрасывать флаг.

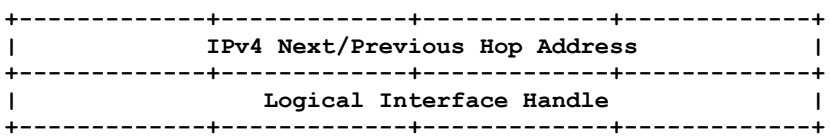
DstPort

Порт получателя UDP/TCP для данной сессии. Может иметь нулевое значение, если порт не задан. Другие значения SESSION C-Type могут быть определены в будущем для поддержки иных соглашений о демультиплексировании на транспортном или прикладном уровне.

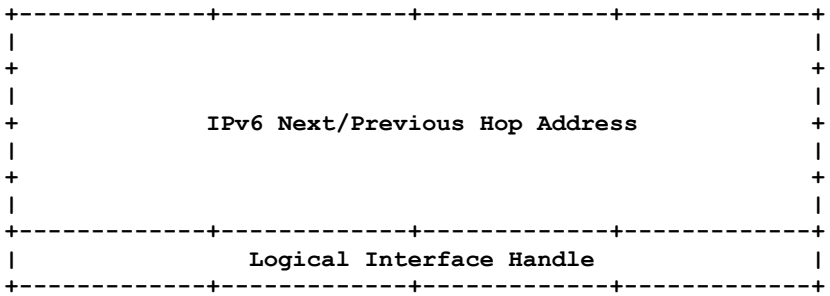
A.2 Класс RSVP_HOP

RSVP_HOP = 3

- Объект IPv4 RSVP_HOP: Class = 3, C-Type = 1



- Объект IPv6 RSVP_HOP: Class = 3, C-Type = 2



Этот объект передает IP-адрес интерфейса, через который понимающий RSVP интервал (маршрутизатор) переслал данное сообщение. Для различения выходных логических интерфейсов используется идентификатор LIH, как было описано в параграфах 3.3 и 3.9. Узел, получивший LIH в сообщении Path, сохраняет это значение и возвращает его в объектах HOP последующих сообщений Resv, передаваемых узлу, от которого исходит LIH. При отсутствии идентификатора логического интерфейса в поле LIH следует устанавливать значение, идентичное 0.

A.3 Класс INTEGRITY

INTEGRITY = 4.

См. [Baker96].

A.4 Класс TIME_VALUES

TIME_VALUES = 5.

- Объект TIME_VALUES: Class = 5, C-Type = 1

```
+-------------+-------------+-------------+-------------+
|                   Refresh Period R                    |
+-------------+-------------+-------------+-------------+
```

Refresh Period

Тайм-аут обновления, служащий для генерации данного сообщения (в миллисекундах).

A.5 Класс ERROR_SPEC

ERROR_SPEC = 6.

- Объект IPv4 ERROR_SPEC: Class = 6, C-Type = 1

```
+-------------+-------------+-------------+-------------+
|            IPv4 Error Node Address (4 байта)          |
+-------------+-------------+-------------+-------------+
|    Flags    |  Error Code |        Error Value        |
+-------------+-------------+-------------+-------------+
```

- Объект IPv6 ERROR_SPEC: Class = 6, C-Type = 2

```
+-------------+-------------+-------------+-------------+
|                                                       |
+                                                       +
|                                                       |
+           IPv6 Error Node Address (16 байтов)         +
|                                                       |
+                                                       +
|                                                       |
+-------------+-------------+-------------+-------------+
|    Flags    |  Error Code |        Error Value        |
+-------------+-------------+-------------+-------------+
```

Error Node Address

IP-адрес узла, обнаружившего ошибку.

Flags

0x01 = InPlace

Этот флаг используется только для объекта ERROR_SPEC в сообщениях ResvErr. Установленный флаг означает наличие и сохранение резервирования в точке отказа.

0x02 = NotGuilty

Этот флаг используется только для объекта ERROR_SPEC в сообщениях ResvErr и устанавливается только на интерфейсе к принимающему приложению. Установленный флаг показывает, что вызвавшая отказ спецификация FLOWSPEC была существенно больше запрошенной данным получателем спецификации FLOWSPEC.

Error Code

Однооктетное описание ошибки.

Error Value

Двухоктетное поле с дополнительной информацией об ошибке. Содержимое поля зависит от Error Type.

Значения полей Error Code и Error Value определены а Приложении B.

A.6 Класс SCOPE

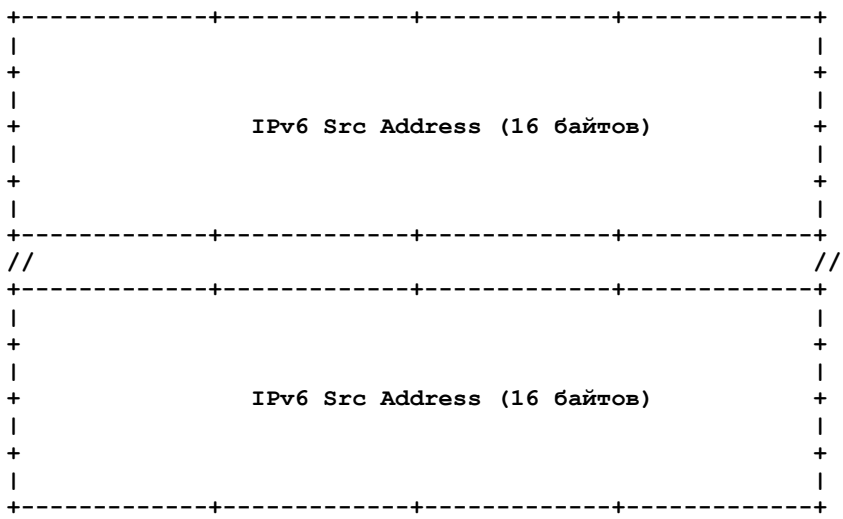
SCOPE = 7

Этот объект содержит список адресов IP, используемых для маршрутизации сообщений с шаблонной областью действия без петель. Адреса должны упорядочиваться по возрастанию числовых значений.

- Объект IPv4 SCOPE List: Class = 7, C-Type = 1

```
+-------------+-------------+-------------+-------------+
|                IPv4 Src Address (4 байта)             |
+-------------+-------------+-------------+-------------+
//                                                      //
+-------------+-------------+-------------+-------------+
|                IPv4 Src Address (4 байта)             |
+-------------+-------------+-------------+-------------+
```

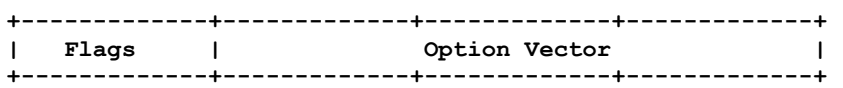
- Объект IPv6 SCOPE List: Class = 7, C-Type = 2



A.7 Класс STYLE

STYLE = 8

- Объект STYLE: Class = 8, C-Type = 1



Flags: 8 битов

(еще не выделены)

Option Vector: 24 бита

Набор битовых полей, задающих значения для опций резервирования. При добавлении в будущем новых опций будут выделяться соответствующие поля вектора опций в более старших его разрядах. Если узел не распознает идентификатор стиля, он может интерпретировать часть вектора опций, игнорируя новые поля.

Биты вектора опций (слева направо) распределены следующим образом:

19 битов: резерв

2 бита: управление совместным использованием резервов

00b: резерв

01b: раздельное резервирование

10b: совместно используемое резервирования

11b: резерв

3 бита: управление выбором отправителя

000b: резерв

001b: шаблон

010b: явное

011b - 111b: резерв

Младшие биты вектора опций определяются стилем, как показано ниже:

WF 10001b

FF 01010b

SE 10010b

A.8 Класс FLOWSPEC

FLOWSPEC = 9

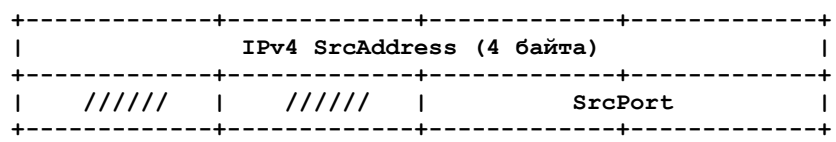
- Зарезервированный (устаревший) объект Flowspec: Class = 9, C-Type = 1
- Объект Inv-serv Flowspec: Class = 9, C-Type = 2

Содержимое и правила представления этих объектов заданы в документах, подготовленных группой int-serv [RFC 2210].

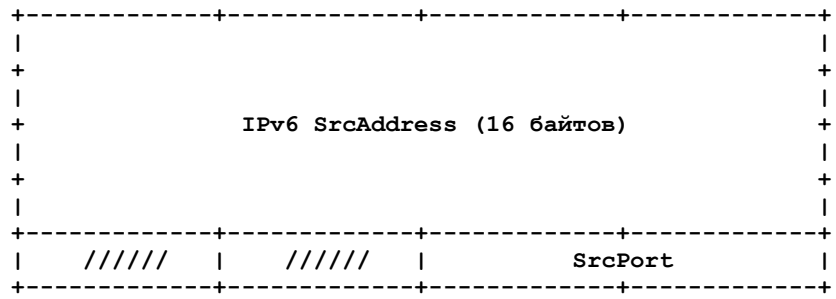
A.9 Класс FILTER_SPEC

FILTER_SPEC = 10

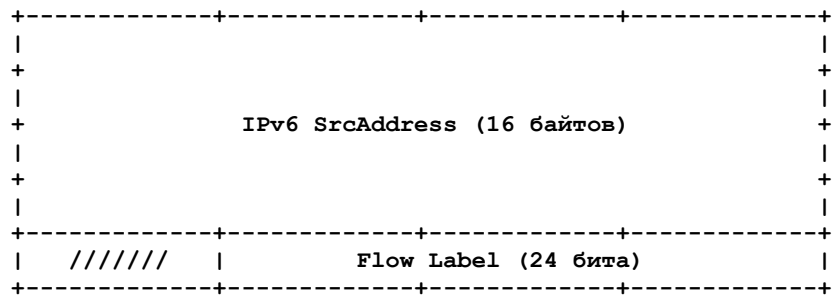
- Объект IPv4 FILTER_SPEC: Class = 10, C-Type = 1



- Объект IPv6 FILTER_SPEC: Class = 10, C-Type = 2



- Объект IPv6 Flow-label FILTER_SPEC: Class = 10, C-Type = 3



SrcAddress

IP-адрес хоста-отправителя (должен быть отличным от 0).

SrcPort

Номер порт UDP/TCP для отправителя или 0 (для индикации none).

Flow Label

24-битовая метка потока (Flow Label), определенная в IPv6. Это значение может использоваться классификатором пакетов для эффективной идентификации пакетов, относящихся к конкретному потоку данных (sender->destination).

A.10 Класс SENDER_TEMPLATE

SENDER_TEMPLATE = 11

- Объект IPv4 SENDER_TEMPLATE: Class = 11, C-Type = 1

  Определение совпадает с определением объекта IPv4/UDP FILTER_SPEC.

- Объект IPv6 SENDER_TEMPLATE: Class = 11, C-Type = 2

  Определение совпадает с определением объекта IPv6/UDP FILTER_SPEC.

- Объект IPv6 Flow-label SENDER_TEMPLATE: Class = 11, C-Type = 3

A.11 Класс SENDER_TSPEC

SENDER_TSPEC = 12

- Intserv SENDER_TSPEC object: Class = 12, C-Type = 2

  Содержимое и правила представления этого объекта заданы в документах, подготовленных группой int-serv.

A.12 Класс ADSPEC

ADSPEC = 13

- Intserv ADSPEC object: Class = 13, C-Type = 2

  Содержимое и формат этого объекта заданы в документах, подготовленных группой int-serv.

A.13 Класс POLICY_DATA

POLICY_DATA = 14

- Объект POLICY_DATA типа 1: Class = 14, C-Type = 1

  Содержимое этого объекта является предметом дальнейшего изучения.

A.14 Класс Resv_CONFIRM

RESV_CONFIRM = 15

- Объект IPv4 RESV_CONFIRM: Class = 15, C-Type = 1

```
+-------------+-------------+-------------+-------------+
|             IPv4 Receiver Address (4 байта)            |
+-------------+-------------+-------------+-------------+
```

- Объект IPv6 RESV_CONFIRM: Class = 15, C-Type = 2

```
+-------------+-------------+-------------+-------------+
|                                                       |
+                                                       +
|                                                       |
+             IPv6 Receiver Address (16 байта)          +
|                                                       |
+                                                       +
|                                                       |
+-------------+-------------+-------------+-------------+
```

Приложение B. Коды и значения ошибок

Перечисленные ниже значения Error Code могут появляться в объектах ERROR_SPEC и передаваться конечным системам. Если явно не указано иное, Error Code может присутствовать только в сообщениях ResvErr.

Error Code = 00: Confirmation - подтверждение

Этот код зарезервирован для использования в объектах ERROR_SPEC сообщений ResvConf. Значение Error Value также будет нулевым.

Error Code = 01: Admission Control failure — отказ котроля допуска

Запрос на резервирование был отвергнут Admission Control по причине нехватки ресурсов.

Для этого Error Code 16 битов Error Value имеют значения:

```
ssur cccc cccc cccc
```

где:

ss = 00: младшие 12 битов, содержащие глобальный субкод (значения приведены ниже).

ss = 10: младшие 12 битов, содержащие специфичный для организации субкод. RSVP будет интерпретировать их лишь как числовые значения.

ss = 11: младшие 12 битов, содержащие специфичный для сервиса субкод. RSVP будет интерпретировать их лишь как числовые значения.

Поскольку механизм контроля трафика может менять сервис, это кодирование может включать некоторое представление об используемом сервисе.

u = 0: RSVP отверг сообщение без изменения локального состояния.

u = 1: RSVP может использовать сообщение для обновления состояния и переслать сообщение дальше. Это означает, что сообщение является информационным.

r: резервный бит (должен иметь значение 0)

cccc cccc cccc: 12-битовый код.

Приведенные ниже глобальные субкоды могут появляться в младших 12 битах, когда ssur = 0000:

- Sub-code = 1: границы по задержкам невозможно соблюсти
- Sub-code = 2: запрошенная полоса не доступна
- Sub-code = 3: MTU в спецификации превышает MTU для интерфейса.

Error Code = 02: Policy Control failure — отказ контроля политики

Сообщение Resv или Pathбыло отвергнуто по административным причинам (например, не представлены требуемые учетные данные, не достаточно квот или баланса, действия администратора. Это значение Error Code может появляться в сообщениях PathErr и ResvErr.

Содержимое поля Error Value будет определено в других документах.

Error Code = 03: No path information for this Resv message — нет информации о пути для этого сообщения

Нет состояния пути для данного сообщения. Сообщение Resv невозможно переслать.

Error Code = 04: No sender information for this Resv message — нет данных об отправителе для сообщения

Имеется состояние пути для данной сессии, но оно не включает отправителя, соответствующего тому или иному дескриптору потока в сообщении Resv. Сообщение Resv невозможно переслать.

Error Code = 05: Conflicting reservation style — конфликт стилей резервирования

Стиль резервирования конфликтует со стилем имеющегося состояния резервирования. Поле Error Value содержит 16 младших битов вектора опций Option Vector существующего стиля, с которым возникает конфликт. Сообщение Resv невозможно переслать.

Error Code = 06: Unknown reservation style — неизвестный стиль резервирования

Стиль резервирования не известен. Сообщение Resv невозможно переслать.

Error Code = 07: Conflicting dest ports — конфликт портов получателей

Сессия с тем же адресом получателя и протоколом указана с нулевым и ненулевым номером порта. Этот код ошибки может присутствовать в сообщениях PathErr и ResvErr.

Error Code = 08: Conflicting sender ports — конфликт портов отправителя

Порт отправителя имеет нулевое и ненулевое значение в сообщениях Path для одной сессии. Этот код може присутствовать только в сообщениях PathErr.

Error Code = 09, 10, 11: (резерв)

Error Code = 12: Service preempted — сервис перенят

Запрос сервиса, определенный объектом STYLE и дескриптором потока, был перенят административно.

Для этого кода 16 битов поля Error Value имеют вид:

```
ssur cccc cccc cccc
```

Старшие биты ssur определены в описании Error Code 01. В младших 12 битах могут указываться глобальные субкоды при определении в будущем ssur = 0000.

Error Code = 13: Unknown object class — неизвестный класс объекта

Error Value содержит 16-битовое значение (Class-Num, C-Type) для неизвестного объекта. Этот код следует передавать лишь в тех случаях, когда RSVP отвергает сообщение, как определено старшими битами Class-Num. Данный код может присутствовать в сообщениях PathErr и ResvErr.

Error Code = 14: Unknown object C-Type – неизвестный объект C-Type

Error Value содержит 16-битовое значение (Class-Num, C-Type) для объекта.

Error Code = 15-19: (резерв)

Error Code = 20: Reserved for API — зарезервирован для API

Поле Error Value содержит код ошибки для детектированной асинхронно ошибки API, сообщение о которой должно передаваться через upcall-вызов.

Error Code = 21: Traffic Control Error — ошибка управления трафиком

Вызов Traffic Control привел к отказу, связанному с форматом или содержимым параметров запроса. Вызвавшее обращение к Traffic Control сообщение Resv или Path не может быть переслано и повторение вызова будет бесполезно.

Для этого кода 16 битов поля Error Value имеют вид:

```
ss00 cccc cccc cccc
```

где старшие биты ss определены в описании Error Code 01.

Ниже приведены глобальные субкоды, которые могут появляться в младших 12 битах (cccc cccc cccc) при ss = 00:

- 01: конфликт служб

  Попытка слияния двух несовместимых запросов сервиса.

- 02: сервис не поддерживается

  Система управления трафиком не может предложить ни запрошенный сервис, ни приемлемую замену.

- 03: некорректное значение Flowspec

  Некорректный по форме или необоснованный запрос.

- 04: некорректное значение Tspec

  Некорректный по форме или необоснованный запрос.

- 05: некорректное значение Adspec

  Некорректный по форме или необоснованный запрос.

Error Code = 22: Traffic Control System error — ошибка системы управления трафиком

Модули контроля трафика обнаружили системную ошибку и сообщают о ней. Значение Error Value будет содержать зависящую от системы дополнительную информацию об ошибке. Не предполагается интерпретация этого значения RSVP.

Error Code = 23: RSVP System error — ошибка системы RSVP

Значение Error Value будет содержать зависящие от реализации сведения об ошибке. Не предполагается интерпретация этого значения RSVP.

В общем случае каждое сообщение RSVP перестраивается на каждом интервале (hop) и создавший сообщение RSVP узел отвечает за корректность его конструкции. От каждого узла требуется проверка корректности конструкции каждого принимаемого сообщения RSVP. Если ошибка программирования позволяет RSVP создавать сообщения с некорректным форматом, сведения о такой ошибке в общем случае не передаются конечным система в объектах ERROR_SPEC — вместо этого информация об ошибке просто записывается в локальный системный журнал и может передаваться через систему сетевого управления.

Единственной ошибкой форматирования сообщений, о которой уведомляются конечные системы, является рассогласование версий — в таких случаях конечные системы способны повлиять на ситуацию (например, вернуться для объекта к предыдущему Ctype — см. коды 13 и 14 выше).

Выбор ошибок форматирования сообщений, которые RSVP может детектировать и протоколировать в локальном системном журнале, зависит от реализации, но обычно в число таких ошибок входят следующие:

- некорректный размер сообщения — значение поля RSVP Length не соответствует размеру сообщения;
- неизвестная или неподдерживаемая версия RSVP;
- некорректная контрольная сумма RSVP;
- отказ INTEGRITY;
- некорректный тип сообщения RSVP;
- некорректный размер объекта (не кратный 4);
- некорректный адрес Next hop/Previous hop в объекте HOP;
- некорректный порт отправителя — отличное от 0 значение в спецификации фильтра или шаблоне отправителя для сессии с нулевым портом получателя;
- отсутствует требуемый класс объекта;
- некорректный для данного типа сообщений класс объекта;
- нарушение требуемого порядка объектов;
- ошибочный для стиля или типа сообщения счетчик дескриптора потока;

- некорректный идентификатор логического интерфейса (LIH);
- неизвестное значение Class-Num для объекта;
- адрес получателя сообщения ResvConf не соответствует Receiver Address содержащегося в сообщении объекта RESV_CONFIRM.

Приложение C. Инкапсуляция в UDP

От реализаций RSVP в общем случае будет требоваться поддержка сетевых операций ввода-вывода без разбора (raw), т. е. передача дейтаграмм IP, использующих протокол 46. Однако некоторые важные классы хост-систем могут не поддерживать такие операции. Для использования RSVP такие хосты должны инкапсулировать сообщения RSVP в UDP.

Базовая схема инкапсуляции в UDP основана на двух допущениях:

1. все хосты могут принимать и передавать групповые пакеты, если он поддерживают multicast-получателей;
2. маршрутизаторы первого/последнего интервала поддерживают RSVP.

Метод ослабления второго допущения будет рассмотрен ниже.

Предположим, что Hu является хостом UDP-only, который требует инкапсуляции в UDP, а хост Hr может работать с raw-пакетами. Схема инкапсуляции UDP должна обеспечивать возможность работы RSVP в произвольной топологии хостов Hr, Hu и маршрутизаторов.

Сообщение Resv, ResvErr, ResvTear и PathErr передаются по индивидуальным адресам, определенным из состояния пути или резервирования на узле. Если узел хранит данные о предыдущих интервалах и необходимости UDP-инкапсуляции для них, эти сообщения могут передаваться при необходимости с использованием инкапсуляции в UDP. С другой стороны, сообщения Path и PathTear передаются по адресу получателя в данной сессии, который может быть индивидуальным или групповым.

На рисунках 13 и 14 показаны базовые правила инкапсуляции в UDP сообщений Path и PathTear, для индивидуальных и групповых DestAddress, соответственно. Для других сообщений, которые передаются по индивидуальным адресам, следует использовать правила рисунка 13. В колонках RSVP Send на этих рисунках используется обозначение mode(destaddr, destport); destport для пакетов raw опускается. Колонки RSVP Receive показывают группу и (когда это применимо), прослушиваемый порт UDP.

Полезно определить два варианта инкапсуляции в UDP — один для передачи Hu, а второй для передачи Hr и R, чтобы предотвратить двойную обработку на приемной стороне. На практике эти два варианта различаются номерами портов UDP - Pu и Pu'.

На рисунках используются следующие обозначения:

- D - DestAddress для конкретной сессии;
- G* - адрес общеизвестной группы в форме 224.0.0.14 (группа, ограниченная локальной сетью);
- Pu и Pu' — общеизвестные порты UDP для UDP-инкапсуляции RSVP (1698 и 1699);
- Ra - IP-адрес интерфейса маршрутизатора a.
- Интерфейс маршрутизатора a подключен к локальной сети, соединенной с Hu и Hr.

Ниже приведены примечания, относящиеся к рисункам:

[1] Hu передает индивидуальное сообщение Path по адресу D, если он локальный, или по адресу Ra маршрутизатора первого интервала. Предполагается, что Ra известен хосту.
[2] D — адрес локального интерфейса, через который поступило сообщение.
[3] Предполагается, что приложение присоединилось к группе D.

Индивидуальный получатель D:

Узел	RSVP Send	RSVP Receive
Hu	UDP(D/Ra,Pu) [1]	UDP(D,Pu) и UDP(D,Pu') [2]
Hr	Raw(D) и, если (UDP), то UDP(D,Pu')	Raw() и UDP(D, Pu) [2] (Игнорировать Pu')
R (Интерфейс a):	Raw(D) и если (UDP), то UDP(D,Pu')	Raw() и UDP(Ra, Pu) (Игнорировать Pu')

Рисунок 13: Правила инкапсуляции UDP для индивидуальных сообщений Path и Resv

```
Групповой получатель D:

Узел            RSVP Send              RSVP Receive
Hu              UDP(G*,Pu)             UDP(D,Pu') [3]
                                     и UDP(G*,Pu)


Hr              Raw(D,Tr)              Raw()
                 и, если (UDP)          и UDP(G*,Pu)
                 то UDP(D,Pu')         (Игнорировать Pu')

R (Интерфейс a):
                Raw(D,Tr)              Raw()
                 и, если (UDP),         и UDP(G*,Pu)
                 то UDP(D,Pu')         (Игнорировать Pu')
```

Рисунок 14: Правила инкапсуляции UDP для групповых сообщений Path и Resv

Маршрутизатор может определить необходимость UDP-инкапсуляции на своем интерфейсе X, прослушавая инкапсулированные в UDP сообщения Path, которые были переданы по адресу G* (multicast D) или интерфейса X (unicast D). В этой схеме существует один вариант отказа — если ни один из хостов подключенной сети не является отправителем RSVP, не будет сообщений Path для инициирования инкапсуляции в UDP. В этом (маловероятном) случае потребуется явно настроить инкапсуляцию в UDP на локальном сетевом интерфейсе маршрутизатора.

При получении пакетов, инкапсулированных в UDP, значение IP TTL не будет доступно приложениям в большинстве систем. Процессу RSVP, который получает инкапсулированные в UDP сообщения Path или PathTear, следует по этой причине использовать поле Send_TTL в общем заголовке RSVP, как эффективное значение TTL для принимаемых сообщений. Это может быть изменено настройкой конфигурации.

Предполагается, что поддерживающий RSVP маршрутизатор первого интервала R находится в подключенной непосредственно сети. Для случаев, когда это условие не выполняется, есть несколько вариантов.

1. Hu может передавать индивидуальные и групповые сессии в UDP(Ra,Pu) с TTL=Ta

   Здесь Ta должно указывать значение TTL, позволяющее достигнуть R, но не более того. Если значение Ta слишком мало, сообщения Path не достигнут R. При слишком большом значении Ta маршрутизатор R и следующие за ним маршрутизаторы могут пересылать пакет UDP, пока не закончится счетчик интервалов. Это будет включать UDP-инкапсуляцию между маршрутизаторами в Internet, что может породить ненужный трафик UDP. Хост Hu должен быть явно настроен для Ra и Ta.

2. Конкретный хост в ЛВС, подключенной к Hu может быть назначен транслятором RSVP (RSVP relay host), который будет прослушивать (G*,Pu) и пересылать сообщения Path напрямую R, хотя это не является путем передачи данных. Транслятор будет настроен для Ra и Ta.

Приложение D. Глоссарий

Admission control — контроль доступа
Функция управления трафиком, принимающая решение о предоставлении планировщиком пакетов на узле запрошенного QoS с сохранением поддержки ранее одобренных запросов QoS (см. policy control и traffic control).

Adspec — спецификация анонса
Adspec представляет собой элемент данных (объект) в сообщении Path, передающий анонсы OPWA См. OPWA.

Auto-refresh loop — петля автообновлений
Ошибочные условия, когда возникает топологическая петля в маршрутизации по которой продолжаются обновления существующих резервирований даже после того, как все получатели прекратили запросы этих резервов. Более подробные сведения приведены в параграфе 3.4.

Blockade state — состояние блокады
Блокирование помогает преодолеть проблему «убойного резервирования» (killer reservation). См. параграфы 2.5 и 3.5, а также killer reservation.

Branch policing
Политика управления трафиком в групповой точке ветвления, когда на одном выходном интерфейсе резервируется «меньше» ресурсов, нежели на другом для того же потока. См. traffic policing.

C-Type
Тип класса объекта, уникальный в рамках имени класса. См. class-name.

Class-name — имя класса
Класс объекта. См. object.

DestAddress
IP-адрес получателя, часть идентификатора сессии. См. session.

Distinct style — раздельное резервирование
Атрибут стиля (резервирования); раздельные ресурсы резервируются для каждого получателя. См. также shared style.

Downstream — нисходящий поток
В направлении получателей данных.

DstPort
(Обобщенный) номер порт получателя IP, используемый, как часть идентификатора сессии. См. generalized destination port.

Entry policing — входная политика
Политика управления трафиком, выполняемая на первом поддерживающем RSVP (и политику) в пути передачи данных.

ERROR_SPEC
Объект, содержащий данные об ошибке в сообщении PathErr или ResvErr.

Explicit sender selection — явный выбор отправителя
Атрибут стиля (резервирования) — все указанные в резервировании отправители явно перечисляются в сообщении о резервировании. См. также wildcard sender selection.

FF style — стиль с фиксированной фильтрацией
Стиль резервирования Fixed Filter, в котором явно выбирается отправитель и атрибуты.

FilterSpec — спецификация фильтра
Вместе с информацией о сесии определяет множество пакетов данных, которым будет обеспечиваться QoS, заданное в flowspec. Спецификация фильтра используется для установки параметров в функции классификации пакетов. FilterSpec может передаваться в объектах FILTER_SPEC и SENDER_TEMPLATE.

Flow descriptor — дескриптор потока
Комбинация спецификаций потока (flowspec) и фильтров (filterspec).

Flowspec — спецификация потока
Определяет QoS для потока. Спецификация потока используется для установки параметров в функции планировщика пакетов с целью обеспечения запрошенного качества обслуживания. Спецификация передается в объекте FLOWSPEC. Формат спецификации непрозрачен для RSVP и определен группой Integrated Services.

Generalized destination port — обобщенный порт назначения
Часть определения сессии, обеспечивающая дополнительный уровень демультиплексирования для транспортного или прикладного протокола сверх DestAddress. См. session.

Generalized source port — обобщенный порт источника
Компонента спецификации фильтра, обеспечивающая дополнительный уровень демультиплексирования для транспортного или прикладного протокола сверх адреса отправителя.

GLB
Наибольшая нижняя граница.

Incoming interface — входной интерфейс
Интерфейс, через который ожидается прибытие пакетов данных и в который передаются сообщения Resv.

INTEGRITY
Объект управляющего сообщения RSVP, содержащий криптографические данные для аутентификации узла-источника и верификации содержимого сообщения RSVP.

Killer reservation problem — проблема «убойного резервирования»
Проблема «убойного резервирования» возникает при неудачной попытке получателя запросить резервирование значительного QoS, когда меньшие QoS не могут быть зарезервированы. Дополнительную информацию см. в параграфах 2.5 и 3.5.

LIH
Логический идентификатор интерфейса LIH (Logical Interface Handle) оказывает помощь при работе с облаками, не поддерживающими RSVP. Дополнительная информацмя приведена в параграфе 2.9.

Local repair — локальный ремонт
Позволяет RSVP быстро адаптироваться к изменениям картины маршрутизации. Дополнительные сведения приведены в параграфе 3.6.

LPM
Локальный модуль политики — функция, осуществляющая контроль политики.

LUB
Наименьшая верхняя граница.

Merge policing — слияние правил
Политика для трафика в точке слияния при совместном резервировании.

Merging - слияние
Процесс принятия максимального (чаще наименьшей верхней границы) резервирования из числа приходящих на выходной интерфейс и пересылки этого максимума на входной интерфейс. Подробности даны в параграфе 2.2.

MTU
Максимальный размер передаваемого блока.

Next hop — следующий интервал
Следующий маршрутизато в направлении передачи потока данных.

NHOP
Объект, передающий информацию Next Hop в управляющих сообщениях RSVP.

Node - узел
Маршрутизатор или хост.

Non-RSVP clouds — облака без поддержки RSVP
Группа хостов и маршрутизаторов, не поддерживающих RSVP. Поддержка работы с такими узлами важна в плане совместимости со старыми версиями. См. параграф 2.9.

Object - объект
Элемент управляющего сообщения RSVP — триплет (тип, размер, значение).

OPWA
Сокращение One Pass With Advertising (один проход с анонсом). Описывает модель резервирования, в которой сообщения (Path), передаваемые в нисходящем направлении, собирают информацию, которую получатели могут использовать для оценки сквозного сервиса. Собираемая информация называется анонсом. См. также Adspec.

Outgoing interface — выходной интерфейс
Интерфейс, через который пересылаются пакеты данных и сообщения Path.

Packet classifier — классификатор пакетов
Функция управления трафиком на основном пути пересылки пакетов данных, выбирающая класс обслуживания для каждого пакета в соответствии с установленным RSVP состоянием резервирования. Функции классификации пакетов и маршрутизации могут объединяться. См. также Traffic control.

Packet scheduler — планировщик пакетов
Функция управления трафиком на основном пути пересылки пакетов данных, реализующая QoS для каждого потока с использованием одной из моделей обслуживания, определенных группой Integrated Services. См. также Traffic control.

Path state — состояние пути
Информация об отправителях RSVP, сохраняемая на хостах и маршрутизаторах.

PathErr
Path Error — управляющее сообщение RSVP.

PathTear
Path Teardown — управляющее сообщение RSVP.

PHOP
Объект, передающий информацию Previous Hop в управляющих сообщениях RSVP.

Police - политика
См. traffic policing.

Policy control — контроль политики
Функция проверки наличия административных разрешений для нового запроса качества обслуживания при организации запрошенного резервирования. Может также обеспечивать учет использования (обратная связь об использовании) для резервирования.

Policy data — данные политики
Данные, переносимые в сообщениях Path или Resv и служащие входной информацией для системы контроля политики при решении вопроса о предоставлении полномочий и/или учете использования для данного потока.

Previous hop — предыдущий интервал
Предыдущий маршрутизатор в направлении потока данных. Сообщения Resv передаются в направлении этого маршрутизатора.

ProtocolId
Часть идентификации сессии, указывающая номер протокола IP, используемого для потока данных.

QoS
Качество обслуживания.

Reservation state — состояние резервирования
Информация об успешных запросах на резервирование RSVP, хранящаяся в узлах RSVP.

Reservation style — стиль резервирования
Описывает набор атрибутов для резервирования, включая атрибуты совместного использования резерва и выбора отправителя. Подробности приведены в параграфе 1.3.

Resv message
Управляющее сообщение RSVP с запросом резервирования.

ResvConf
Управляющее сообщение RSVP с подтверждением резервирования, говорящее об успешной организации резерва на неком узле восходящего направления.

ResvErr
Управляющее сообщение Reservation Error, говорящее об отказе в запросе на резервировании или перенятии активного резервирования.

ResvTear
Управляющее сообщение RSVP, которое удаляет состояние резервирования.

Rspec
Часть flowspec, определяющая запрашиваемый QoS. Формат Rspec непрозрачен для RSVP и определен группой IETF Integrated Services.

RSVP_HOP
Объект управляющего сообщения RSVP, передающий адрес PHOP или NHOP источника сообщения.

Scope - область
Набор хостов-отправителей, которым распространяется данный запрос на резервирование.

SE style — стиль совместного резервирования
Стиль Shared Explicit с явным выбором отправителя и совместно используемыми атрибутами.

Semantic fragmentation — семантическая фрагментация
Метод фрагментации больших сообщений RSVP, использующий информацию о структуре и содержимом сообщения для того, чтобы каждый фрагмент был логически полным сообщением RSVP.

Sender template — шаблон отправителя
Параметр в сообщении Path, определяющий отправителя (передается в объекте SENDER_TEMPLATE). Является формой спецификации фильтра, которая может использоваться для выбора пакетов отправителя среди других пакетов в той же сессии на том же канале.

Sender Tspec
Параметр в сообщении Path - значение Tspec, характеризующее параметры трафика для потока данных от соответствующего отправителя. Передается в объекте SENDER_TSPEC.

Session
Сессия RSVP определяет односторонний (simplex) индивидуальный или групповой поток данных, для которого требуется резервирование. Сессия идентифицируется адресом получателя и протоколом транспортного уровня, к которым может добавляться (обобщенный) порт получателя.

Shared style — разделяемый стиль
Атрибут стиля (резервирования) — все отправители используют общий набор зарезервированных ресурсов. См. также distinct style.

Soft state
Состояние хоста или маршрутизатора, завершающееся при отсутствии обновлений в течение заданного времени.

STYLE
Объект в сообщении RSVP, задающий желаемый стиль резервирования.

Style
См. reservation style.

TIME_VALUES
Объект в управляющем сообщении RSVP, задающий время для таймера, используемого при обновлении состояния в данном сообщении.

Traffic control — управление трафиком
Полный набор компонент узла, обеспечивающих запрошенные параметры QoS для потоков данных. Включает классификатор и планировщик пакетов, а также контроль доступа.

Traffic policing — правила для трафика
Функция управления трафиком, приводящая поток данных в соответствие с параметрами трафика для резервирования. Например, такая функция может отбрасывать не соответствующие правилам пакеты или пересылать их с низким приоритетом.

TSpec

Набор параметров трафика, описывающих поток. Формат Tspec непрозрачен для RSVP и определен рабочей группой Integrated Service.

UDP encapsulation — инкапсуляция в UDP

Метод, с помощью которого хосты, не поддерживающие сокеты raw, могут участвовать в работе RSVP за счет инкапсуляции неразобранных (raw) пакетов RSVP в стандартные пакеты UDP. Дополнительная информация приведена в Приложении C.

Upstream — восходящий поток

Направление к источнику трафика. В этом направлении передаются сообщения RSVP Resv.

WF style — шаблонный стиль

Стиль резервирования Wildcard Filter с выбором отправителя по шаблону и совместно используемыми атрибутами.

Wildcard sender selection — шаблонный выбор отправителя

Атрибут стиля (резервирования) — трафик от любого отправителя в конкретную сессию получает одинаковые параметры QoS. См. также explicit sender selection.

Литература

[Baker96] Baker, F., "RSVP Cryptographic Authentication", Work in Progress[1].

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

[FJ94] Floyd, S. and V. Jacobson, "Synchronization of Periodic Routing Messages", IEEE/ACM Transactions on Networking, Vol. 2, No. 2, April, 1994.

[RFC 2207] Berger, L. and T. O'Malley, "RSVP Extensions for IPSEC Data Flows", RFC 2207, September 1997.

[RFC 2113] Katz, D., "IP Router Alert Option", RFC 2113, cisco Systems, February 1997. (перевод)

[RFC 2210] Wroclawski, J., "The Use of RSVP with Integrated Services", RFC 2210, September 1997. (перевод)

[PolArch96] Herzog, S., "Policy Control for RSVP: Architectural Overview". Work in Progress[2].

[OPWA95] Shenker, S. and L. Breslau, "Two Issues in Reservation Establishment", Proc. ACM SIGCOMM '95, Cambridge, MA, August 1995.

[RSVP93] Zhang, L., Deering, S., Estrin, D., Shenker, S., and D. Zappala, "RSVP: A New Resource ReSerVation Protocol", IEEE Network, September 1993.

Вопросы безопасности

См. параграф 2.8.

Адреса авторов

Bob Braden

USC Information Sciences Institute

4676 Admiralty Way

Marina del Rey, CA 90292

Phone: (310) 822-1511

EMail: Braden@ISI.EDU

Lixia Zhang

UCLA Computer Science Department

4531G Boelter Hall

Los Angeles, CA 90095-1596 USA

Phone: 310-825-2695

EMail: lixia@cs.ucla.edu

Steve Berson

USC Information Sciences Institute

4676 Admiralty Way

Marina del Rey, CA 90292

Phone: (310) 822-1511

EMail: Berson@ISI.EDU

[1] Работа завершена и опубликована в RFC 2747 (перевод). *Прим. перев.*

[2] Работа завершена и опубликована в RFC 2748, RFC 2749. *Прим. перев.*

Shai Herzog

IBM T. J. Watson Research Center

P.O Box 704

Yorktown Heights, NY 10598

Phone: (914) 784-6059

EMail: Herzog@WATSON.IBM.COM

Sugih Jamin

University of Michigan

CSE/EECS

1301 Beal Ave.

Ann Arbor, MI 48109-2122

Phone: (313) 763-1583

EMail: jamin@EECS.UMICH.EDU

Перевод на русский язык

Николай Малых

nmalykh@protocols.ru