Network Working Group
Request for Comments: 5036
Obsoletes: 3036
Category: Standards Track

L. Andersson, Ed.
Acreo AB
I. Minei, Ed.
Juniper Networks
B. Thomas, Ed.
Cisco Systems, Inc.
October 2007

Спецификация LDP

LDP Specification

Статус документа

Этот документ содержит проект стандартного протокола Internet для сообщества Internet и служит приглашением к дискуссии в целях развития и совершенствования протокола. Информацию о текущем состоянии стандартизации протокола можно найти в документе Internet Official Protocol Standards (STD 1). Документ можно распространять без ограничений.

Аннотация

Архитектура многопротокольной коммутации по меткам (MPLS[1]) описана в RFC 3031. Базовая концепция MPLS заключается в том, что два маршрутизатора с коммутацией по меткам (LSR[2]) должны согласовать между собой толкование меток, используемых для пересылки трафика между маршрутизаторами и через них (транзит). Для согласования служит набор процедур, который называют протоколом распространения меток. С помощью этого протокола каждый LSR информирует других о созданных им связках меток. В данном документе определён набор процедур - протокол LDP[3], с помощью которых маршрутизаторы LSR распространяют метки для поддержки пересылки MPLS по путям с обычной маршрутизацией.

Оглавление

[1]Multiprotocol Label Switching.
[2]Label Switching Router.
[3]Label Distribution Protocol.

1. Обзор LDP

Архитектура MPLS [RFC3031] определяет протокол распространения меток как набор процедур, посредством которых маршрутизатор LSR информирует другой маршрутизатор о значениях меток, используемых для пересылки трафика между маршрутизаторами и через них.

Архитектура MPLS не предполагает использования единственного протокола распространения меток. Фактически стандартизировано множество таких протоколов. Существующие протоколы были расширены для того, чтобы с их помощью можно было распространять метки. Были также определены новые протоколы, явно предназначенные для распространения меток. В архитектуре MPLS приведено рассмотрение некоторых аспектов выбора протокола распространения меток для отдельных приложений MPLS, таких как TE[1] [RFC2702].

Протокол LDP разработан специально для распространения меток. Первый вариант этого протокола был опубликован в RFC 3036 (январь 2001). Документ был подготовлен рабочей группой MPLS в составе IETF, авторами его были Loa Andersson, Paul Doolan, Nancy Feldman, Andre Fredette и Bob Thomas.

Протокол LDP предназначен для распространения меток. Он представляет собой набор процедур и сообщений, с помощью которых маршрутизаторы LSR организуют пути с коммутацией по меткам (LSP[2]) через сеть за счёт отображения маршрутной информации сетевого уровня непосредственно на коммутируемые пути канального уровня. Конечные точки LSP могут располагаться на смежных маршрутизаторах (сравнимо с поэтапной пересылкой IP) или на выходных узлах сети, обеспечивающей коммутацию на всех промежуточных узлах.

LDP связывает класс эквивалентной пересылки (FEC[3]) [RFC3031] с каждым создаваемым LSP. Класс FEC, связанный с LSP, задаёт, какие пакеты будут отображаться на данный LSP. При прохождении LSP через сеть каждый LSR «сращивает» входящую метку для FEC с исходящей меткой, которую выделил для данного FEC следующий маршрутизатор.

Дополнительную информацию о применимости протокола LDP можно найти в [RFC3037].

Данный документ предполагает (но не требует) знакомства читателя с архитектурой MPLS [RFC3031]. Отметим, что [RFC3031] включает глоссарий терминов, связанных с MPLS.

1.1. Партнёры LDP

Пару маршрутизаторов LSR, использующих протокол LDP для обмена информацией о связывании меток с FEC, будем называть партнёрами LDP (LDP Peer), а протокольное взаимодействие между партнёрами - сессией LDP. Одна сессия LDP позволяет каждому из партнёров узнать отображения, выполненные другим партнёром, т. е. протокол является двусторонним.

[1]Traffic Engineering - организация трафика.
[2]Label Switched Path.
[3]Forwarding Equivalence Class.

1.2. Обмен сообщениями LDP

Существует 4 категории сообщений LDP:

1. сообщения об обнаружении (Discovery) используются для анонсирования и поддержки присутствия LSR в сети;
2. сеансовые сообщения (Session) служат для организации, поддержки и завершения сеансов обмена между партнёрами LDP;
3. анонсы (Advertisement) используются для создания, изменения и удаления отображений FEC на метки;
4. уведомления (Notification) служат для передачи дополнительной информации и сведений об ошибках.

Сообщения Discovery обеспечивают механизм, с помощью которого LSR показывает своё присутствие в сети, передавая периодически сообщения Hello. Эти сообщения передаются в форме пакетов UDP на порт LDP с групповым адресом all routers on this subnet[1] в поле получателя. Когда LSR намеревается организовать сеанс взаимодействия с другим LSR, найденным с помощью сообщения Hello, он использует процедуру инициализации LDP по протоколу TCP. После успешной инициализации два LSR становятся партнёрами LDP и могут обмениваться анонсами.

Запрос метки или анонсирование отображения метки партнёру определяется в основном локально маршрутизатором LSR. В общем случае LSR запрашивает отображение метки у соседнего LSR, когда такое отображение нужно, и анонсирует отображение метки соседнему LSR, когда желает, чтобы тот использовал эту метку.

Для корректной работы LDP требуется надёжная и упорядоченная доставка сообщений, поэтому LDP использует транспортный протокол TCP для сообщений Session, Advertisement и Notification, а протокол UDP применяется только для сообщений Discovery.

1.3. Структура сообщения LDP

Все сообщения LDP имеют однотипную структуру, основанную на формате TLV[2] (см. параграф 3.3. Представление TLV). Часть Value объекта в формате TLV может содержать один или множество других элементов TLV.

1.4. Обработка ошибок LDP

Информация об ошибках LDP и других событиях передаётся партнёру LDP в сообщениях Notification.

Существует два типа сообщений LDP Notification.

1. Уведомления об ошибках служат для передачи информации о критических ошибках. Если LSR получает от своего партнёра сообщение Error Notification для сессии LDP, он завершает эту сессию, закрывая транспортное соединение TCP для неё и отбрасывая отображения меток, полученные в этой сессии.
2. Сообщения Advisory Notification служат для передачи информации о состоянии сессии или статусе некоторых сообщений, полученных ранее от партнёра.

1.5. Расширяемость LDP и совместимость с будущими версиями

В будущем к протоколу LDP могут быть добавлены новые функции. Очевидно, что для новой функциональности будут применяться новые сообщения и типы TLV. Может оказаться желательным развёртывание реализаций с новыми сообщениями и TLV в существующих сетях таким образом, чтобы более старые реализации просто не распознавали новинки. Хотя и невозможно сделать все новые расширения совместимыми со старыми версиями, некоторое упреждающее планирование может упростить внесение новых возможностей. В данной спецификации определены правила обработки новых сообщений и новых типов TLV с учётом этих обстоятельств.

1.6. Уровни требований

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не нужно** (SHALL NOT), **следует** (SHOULD), **не следует** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе должны интерпретироваться в соответствии с [RFC2119].

2. Работа LDP

2.1. Классы FEC

Требуется точно указать, какие пакеты могут отображаться на каждый LSP. Это выполняется путём задания спецификации FEC для каждого LSP. Класс FEC идентифицирует множество пакетов IP, которые могут быть отображены на данный LSP.

Класс FEC задается как непустое множество элементов FEC, идентифицирующих множества пакетов, которые могут быть отображены на соответствующий LSP. Когда LSP совместно используется множеством элементов FEC, такой LSP завершается на узле (или до него), где элементы FEC уже не могут далее совместно использовать общий путь.

Эта спецификация определяет единственный тип элементов FEC - «адресный префикс[3]». Этот элемент представляет собой адресный префикс, размер которого может меняться от 0 до полного размера адреса, включительно.

При необходимости в других спецификациях могут быть определены дополнительные элементы FEC.

В оставшейся части этого параграфа приведены правила, которые будут использоваться для отображения пакетов на LSP, организуемые с помощью элементов Address Prefix FEC.

Адрес «соответствует» адресному префиксу тогда и только тогда, когда этот адрес начинается с данного префикса. Тот или иной пакет соответствует LSP тогда и только тогда, когда данный LSP имеет элемент Address Prefix FEC,

[1]Все маршрутизаторы данной подсети - 224.0.0.2 в IPv4. *Прим. перев.*
[2]Type-Length-Value - тип, размер, значение.
[3]Address Prefix FEC.

соответствующий адресу получателя пакета. Применительно к конкретному пакету и конкретному LSP будем называть элемент Address Prefix FEC, который соответствует пакету, «соответствующим префиксом».

Процедура отображения конкретного пакета на конкретный LSP использует приведённые ниже правила, каждое из которых применяется по очереди, пока пакет не будет отображён на LSP.

- Если пакет соответствует в точности одному LSP, он отображается на данный LSP.
- Если пакет соответствует множеству LSP, он отображается на тот LSP, который имеет наиболее длинный соответствующий префикс. Если среди LSP нет одного с самым длинным префиксом, пакет отображается на один из множества LSP, имеющих самые длинные соответствия префиксов. Процедура выбора одного из LSP с соответствующими префиксами одного размера выходит за рамки данного документа.
- Если известно, что пакет должен пройти через конкретный выходной маршрутизатор и имеется LSP с элементом Address Prefix FEC, который точно (/32) соответствует адресу данного маршрутизатора, пакет отображается на этот LSP. Процедура получения такой информации выходит за рамки данного документа.

Процедура определения необходимости прохождения пакета через конкретный выходной маршрутизатор выходит за рамки данной спецификации. В качестве примера укажем, что при использовании алгоритма маршрутизации по состояниям каналов можно получить нужную информацию из базы данных о состояниях каналов. В случае использования BGP информацию можно получить из атрибута BGP next hop в маршруте для данного пакета.

2.2. Пространства меток, идентификаторы, сессии и транспорт

2.2.1. Пространства меток

Понятие «пространства меток» полезно при обсуждении назначения и распространения меток. Используется два типа пространств меток.

- Пространство меток интерфейса. Связанные с интерфейсом входящие метки применяются для интерфейсов, которые используют ресурсы данного интерфейса для меток. Примером такого интерфейса может служить управляемый по меткам интерфейс ATM, использующий VCI[1] в качестве меток, или интерфейс Frame Relay, который использует значения DLCI[2] как метки.

  Отметим, что использовать пространство меток интерфейса имеет смысл лишь в тех случаях, когда партнёры LDP напрямую соединены через этот интерфейс и метки будут применяться только для трафика, передаваемого через этот интерфейс.

- Пространство меток платформы. Входящие метки в масштабе всей платформы используются для интерфейсов, которые могут применять общие метки.

2.2.2. Идентификаторы LDP

Идентификатор LDP представляет собой 6-октетное значение, которое указывает пространство меток LSR. Первые 4 октета указывают маршрутизатор LSR и должны быть уникальными в глобальном масштабе. Этому условию удовлетворяет 32-битовое значение Router Id, назначенное LSR. Два последних октета указывают пространство меток в данном LSR. Эти два октета LDP Identifier для пространства меток платформы всегда имеют нулевые значения. В данной спецификации используется представление идентификаторов LDP в виде:

```
<LSR Id>:<label space id>
```

например, lsr171:0, lsr19:2.

Отметим, что LSR, которые поддерживает и анонсирует множество пространств меток, использует для каждого из таких пространств своё значение LDP Identifier.

Ситуация, когда LSR требуется анонсировать партнёру более одного пространства меток и, следовательно, использовать множество идентификаторов LDP, возникает при наличии у LSR двух ATM-соединений с партнёром и использовании пространства меток в масштабе интерфейса. Другим примером может служить LSR, соединённый с партнёром через интерфейсы Ethernet (пространство меток в масштабе платформы) и ATM.

2.2.3. Сессии LDP

Сеансы LDP организуются между парой LSR для поддержки обмена метками.

Когда LSR использует LDP для анонсирования множества пространств меток другому LSR, для каждого из пространств меток создаётся своя сессия LDP.

2.2.4. Транспорт LDP

LDP использует протокол TCP в качестве надёжного транспорта для поддержки сессий.

Если между парой LSR требуется организовать множество сессий LDP, организуется одно соединение (сессия) TCP на каждую сессию LDP.

2.3. Сессии LDP между LSR, не соединёнными напрямую

В некоторых ситуациях могут оказаться желательными сессии LDP между маршрутизаторами LSR, не соединёнными между собой непосредственно на канальном уровне.

В качестве примера рассмотрим задачу организации трафика для случая, когда LSRa передаёт трафик, соответствующий некому критерию, маршрутизатору LSRb, с которым не имеет непосредственного соединения, по пути LSP вместо пересылки такого трафика по пути с обычной маршрутизацией.

[1]Virtual Channel Identifier - идентификатор виртуального канала.
[2]Data Link Connection Identifier - идентификатор соединения на канальном уровне.

Путь между LSRa и LSRb будет включать один или множество промежуточных маршрутизаторов LSR (LSR1, …, LSRn). Сессия LDP между LSRa и LSRb будет позволять маршрутизатору LSRb помечать коммутируемый трафик, приходящий по LSP от LSRa, благодаря возможности анонсирования маршрутизатором LSRb используемых для этого меток маршрутизатору LSRa.

В такой ситуации LSRa будет использовать для трафика, пересылаемого в LSP для маршрутизатора LSRb, две метки - первая метка будет получена от промежуточного маршрутизатора LSR1 и служит для пересылки трафика по пути LSP от LSRa к LSRb, вторая метка будет получена от LSRb для того, чтобы тот мог помечать коммутируемый трафик, приходящий на LSP.

LSRa сначала добавляет в стек метку, полученную через сеанс LDP с маршрутизатором LSRb (меняя верхнюю метку в стеке пакета, если пакет был помечен или вталкивая метку в стек непомеченного пакета). После этого в стек помещается метка для LSP, полученная от LSR1.

2.4. LDP Discovery

Обнаружение LDP (LDP Discovery) представляет собой механизм, который позволяет маршрутизаторам LSR находить потенциальных партнёров LDP. Обнаружение избавляет от необходимости явно настраивать пары LSR для коммутации по меткам.

Существует два варианта механизма обнаружения:

- основной механизм (Basic Discovery) используется для обнаружения соседних LSR, с которыми есть непосредственное соединение на канальном уровне;
- расширенный механизм (Extended Discovery) используется для нахождения LSR, с которыми нет прямого соединения на канальном уровне.

2.4.1. Основной механизм обнаружения

Для включения на интерфейсе механизма LDP Basic Discovery маршрутизатор LSR периодически шлёт сообщения LDP Link Hello через этот интерфейс. Сообщения LDP Link Hello передаются в форме пакетов UDP, направляемых в стандартный порт LDP по групповому адресу всех маршрутизаторов подсети.

Сообщение LDP Link Hello, переданное LSR, содержит идентификатор LDP для пространства меток, которое LSR предполагает использовать на этом интерфейсе, и может содержать дополнительную информацию.

Получение сообщения LDP Link Hello через некий интерфейс показывает наличие смежности (Hello adjacency) с потенциальным партнёром LDP, доступным через этот интерфейс на канальном уровне, а также пространство меток, которое партнёр предлагает использовать для данного интерфейса.

2.4.2. Расширенный механизм обнаружения

Сессии LDP между LSR, не соединёнными напрямую, поддерживаются с помощью механизма LDP Extended Discovery.

Для включения LDP Extended Discovery маршрутизатор LSR периодически шлёт сообщения LDP Targeted Hello по конкретному адресу. Сообщения LDP Targeted Hello передаются в форме пакетов UDP направленных в стандартный порт LDP по указанному адресу.

Сообщение LDP Targeted Hello, переданное маршрутизатором LSR, содержит идентификатор LDP для пространства меток, которое LSR предполагает использовать на этом интерфейсе, и может содержать дополнительную информацию.

Механизм Extended Discovery отличается от Basic Discovery в следующем:

- сообщение Targeted Hello передаётся по конкретному адресу вместо группового адреса всех маршрутизаторов подсети;
- механизм Basic Discovery является симметричным, а Extended Discovery - асимметричным.

  Маршрутизатор LSR инициирует процедуру Extended Discovery с интересующим его LSR, а тот самостоятельно принимает решение об ответе на сообщение Targeted Hello или игнорировании его. LSR, решивший ответить на приветствие, выполняет это путём периодической отправки сообщений Targeted Hello инициировавшему обмен маршрутизатору LSR.

Получение сообщения LDP Targeted Hello показывает наличие смежности (Hello adjacency) с потенциальным партнёром LDP, доступным на сетевом уровне, а также пространство меток, которое партнёр предлагает использовать.

2.5. Организация и поддержка сессий LDP

2.5.1. Организация сеанса LDP

Обмен сообщениями LDP Discovery Hello между парой LSR включает процесс организации сессии LDP, состоящий из двух этапов:

- организация транспортного соединения;
- инициализация сессии.

Далее описана организация сессии LDP между маршрутизаторами LSR1 и LSR2 с точки зрения LSR1. Предполагается, что при обмене сообщениями Hello было задано пространство меток LSR1:a для маршрутизатора LSR1 и LSR2:b - для LSR2.

2.5.2. Организация транспортного соединения

Обмен сообщениями Hello приводит к организации Hello-смежности на LSR1, которая служит для связывания канала L с пространствами меток LSR1:a и LSR2:b.

1. Если у LSR1 ещё нет сеанса LDP для обмена пространствами меток LSR1:a и LSR2:b, он пытается организовать соединение TCP для новой LDP-сессии с LSR2.

   LSR1 определяет транспортные адреса, которые будут использоваться на его стороне (A1) и на стороне LSR2 (A2) для соединения TCP. Адрес A1 определяется следующим образом:

   a) если LSR1 использует необязательный объект Transport Address (TLV) в сообщениях Hello, передаваемых LSR2 для анонсирования адреса, A1 будет адресом, который LSR1 анонсирует в этом необязательном объекте;

   b) если LSR1 не использует объект Transport Address, A1 будет адресом отправителя, используемым в сообщениях Hello для LSR2.

   Адрес A2 определяется похожим способом:

   a) если LSR2 использует необязательный объект Transport Address, A2 будет адресом, который LSR2 анонсирует в этом объекте;

   b) если LSR2 не использует объект Transport Address, A2 будет адресом отправителя, используемым в сообщениях Hello от LSR2.

2. LSR1 определяет свою роль (активный или пассивный) в организуемой сессии, путём сравнения адресов A1 и A2, как целых чисел без знака. Если A1 > A2, LSR1 будет играть активную роль, в ином случае - пассивную.

   Процедура сравнения беззнаковых целых чисел A1 и A2 выполняется следующим образом:

   – Если адреса A1 и A2 относятся к разным семействам, они являются несравнимыми и сессия не создаётся.

   – Пусть U1 будет абстрактным целым числом без знака, которое получается в результате трактовки A1, как последовательности байтов, в которой первый байт адреса, указанный в сообщении, является старшим, а последний байт адреса в сообщении - младшим.

     U2 получается из адреса A2 таким же способом.

   – Сравниваются значения U1 и U2. Если U1 > U2, то A1 > A2; если U1 < U2, то A1 < A2.

3. Если LSR1 является активным, он пытается инициировать соединение TCP через стандартный порт LDP по адресу A2. Если LSR1 является пассивным, он ждёт вызова от LSR2 для организации соединения TCP через стандартный порт LDP.

Отметим, что маршрутизатор LSR при передаче сообщения Hello выбирает транспортный адрес для своей стороны сеансового соединения и использует сообщение Hello для анонсирования этого адреса (явно в необязательном параметре Transport Address TLV или неявно в качестве адреса отправителя сообщения Hello).

LSR **должен** задавать один и тот же транспортный адрес во всех сообщениях Hello, анонсирующих одно пространство меток. Это обеспечивает гарантию того, что два LSR, связанные множеством отношений Hello-смежности и использующие одни и те же пространства меток, будут играть одинаковую роль (активный-пассивный) во всех отношениях смежности.

2.5.3. Инициализация сессии

После того, как LSR1 и LSR2 организовали транспортное соединение, они согласуют параметры сессии путём обмена сообщениями LDP Initialization. Согласуемые параметры включают версию протокола LDP, метод распространения меток, значения таймеров, диапазоны VPI/VCI[1] для меток, управляемых ATM, диапазоны DLCI для меток, управляемых Frame Relay и т. п.

Согласование завершает процесс организации сессии LDP между LSR1 и LSR2 для анонсирования пространств меток LSR1:a и LSR2:b.

Ниже описана процедура инициализации с точки зрения маршрутизатора LSR1.

Если LSR1 играет активную роль, он после организации соединения инициирует согласование параметров сессии путём передачи сообщения Initialization маршрутизатору LSR2. Если LSR1 является пассивным, он ждёт, пока LSR2 инициирует согласование параметров сессии.

В общем случае при наличии между LSR1 и LSR2 множества соединений и анонсирования каждым маршрутизатором множества пространств меток, пассивный маршрутизатор LSR не может знать, какое из пространств меток анонсировать через вновь созданное соединение TCP, пока не будет получено сообщение LDP Initialization. Это сообщение содержит значения LDP Identifier для пространств меток передающей (активный LSR) и приёмной (пассивный LSR) стороны.

В ожидании сообщения Initialization от партнёра пассивный маршрутизатор LSR может сопоставить пространство меток, которое будет анонсироваться партнёром (определяется значением LDP Identifier в заголовке PDU[2] сообщения Initialization), с Hello-смежностью, созданной при обмене сообщениями Hello.

1. LSR1 играет пассивную роль.

   a) При получении сообщения Initialization маршрутизатор LSR1 пытается сопоставить LDP Identifier из PDU принятого сообщения со смежностью Hello.

[1]Virtual Path Identifier/Virtual Channel Identifier - идентификаторы виртуального пути и виртуального соединения.
[2]Protocol Data Unit - блок данных протокола. *Прим. перев.*

b) Если соответствие найдено, оно будет определять пространство меток для этой сессии.

После этого LSR1 проверяет приемлемость предложенных параметров сессии. Если параметры устраивают, LSR1 передаёт ответное сообщение Initialization с желаемыми параметрами и сообщение KeepAlive, показывающее приемлемость предложенных LSR2 параметров. Если параметры не подходят, LSR1 передаёт сообщение Session Rejected/Parameters Error Notification[1] и закрывает соединение TCP.

c) Если LSR1 не может найти соответствующей Hello-смежности, он передаёт сообщение Session Rejected/No Hello Error Notification[2] и закрывает соединение TCP.

d) Если LSR1 получает сообщение KeepAlive в ответ на своё сообщение Initialization, сессия считается открытой с точки зрения LSR1.

e) Если LSR1 получает сообщение Error Notification, это говорит о том, что LSR2 отказался от предложенной сессии, и LSR1 закрывает соединение TCP.

2. LSR1 играет активную роль.

a) Если LSR1 получает сообщение Error Notification, это говорит о том, что LSR2 отказался от предложенной сессии, и LSR1 закрывает соединение TCP.

b) При получении сообщения Initialization маршрутизатор LSR1 проверяет приемлемость предложенных параметров сессии. Если параметры устраивают, LSR1 передаёт сообщение KeepAlive. Если параметры не подходят, LSR1 передаёт сообщение Session Rejected/Parameters Error Notification и закрывает соединение.

c) Если LSR1 получает сообщение KeepAlive, это говорит о том, что LSR2 принял предложенные параметры сессии.

d) Если LSR1 получает сообщение Initialization о приемлемости предложенных параметров и сообщение KeepAlive, сессия считается открытой с точки зрения LSR1.

Пока сессия LDP не открыта, никакие сообщения, за исключением упомянутых выше, не могут передаваться и правила обработки бита U в сообщениях LDP не могут меняться. При получении любого сообщения, кроме перечисленных выше, в ответ **должно** быть передано сообщение Shutdown и транспортное соединение **должно** быть закрыто.

Возможно, что для пары несовместимых по конфигурации LSR несогласие с параметрами сессии может порождать бесконечную цепочку сообщений, когда каждый из маршрутизаторов будет отвергать сообщение Initialization от другого маршрутизатора с передачей сообщения Error Notification.

LSR **должны** экспоненциально снижать частоту попыток организации сессии в ситуациях, когда их сообщения Initialization отвергаются с помощью NAK[3]. Рекомендуется также, чтобы LSR при обнаружении такой ситуации предпринимал действия по уведомлению оператора.

Попытка повторной организации сессии после отвергнутого (NAK) сообщения Initialization **должна** предприниматься не ранее, чем через 15 секунд, а для последующих попыток задержка **должна** расти с максимальным значением не менее 2 минут. Конкретным действием по организации сессии, которое должно задерживаться, является организация транспортного соединения активным маршрутизатором LSR.

Заторможенная последовательность Initialization NAK может не прекратиться, пока оператор не изменит конфигурацию одного из LSR. После внесения требуемых изменений больше не нужно тормозить последующие попытки организации сессии (до получения Initialization NAK).

В силу асимметрии процесса организации сессии изменение конфигурации пассивного LSR не будет известно активному маршрутизатору LSR без специальных действий. В параграфе 3.5.2. Сообщение Hello описан дополнительный механизм, который LSR может использовать для информирования потенциальных партнёров LDP о смене конфигурации.

2.5.4. Инициализация машины состояний

Для описания процесса согласования сессии LDP удобно использовать модель машины состояний. Определим для машины состояний LDP пять возможных состояний и представим поведение этой машины в форме таблицы и диаграммы переходов. Отметим, что сообщение Shutdown реализуется в форме сообщения Notification, со значением Status TLV, показывающим критическую ошибку.

Таблица состояний при инициализации сеанса.

Состояние	***Событие***	***Новое состояние***
NON EXISTENT	Организация соединения TCP	INITIALIZED
INITIALIZED	Передача сообщения Initialization (активный)	OPENSENT
	Получение приемлемого сообщения Initialization (пассивный) Действие: Передача сообщений Initialization и KeepAlive	OPENREC
	Приём любого другого сообщения LDP Действие: Передача сообщения Error Notification (NAK) и разрыв транспортного соединения	NON EXISTENT
OPENREC	Приём сообщения KeepAlive	OPERATIONAL
	Приём любого другого сообщения LDP Действие: Передача сообщения Error Notification (NAK) и разрыв транспортного соединения	NON EXISTENT

[1]Сессия отвергнута / Индикация ошибок в параметрах.
[2]Сессия отвергнута / Индикация отсутствия ошибок в сообщении Hello.
[3]Отказ. *Прим. перев.*

OPENSENT	Получение приемлемого сообщения Initialization Действие: Передача сообщения KeepAlive	OPENREC
	Приём любого другого сообщения LDP Действие: Передача сообщения Error Notification (NAK) и разрыв транспортного соединения	NON EXISTENT
OPERATIONAL	Приём сообщения Shutdown Действие: Передача сообщения Shutdown и разрыв транспортного соединения	NON EXISTENT
	Приём любого другого сообщения LDP	OPERATIONAL
	Таймаут Действие: Передача сообщения Shutdown и разрыв транспортного соединения	NON EXISTENT

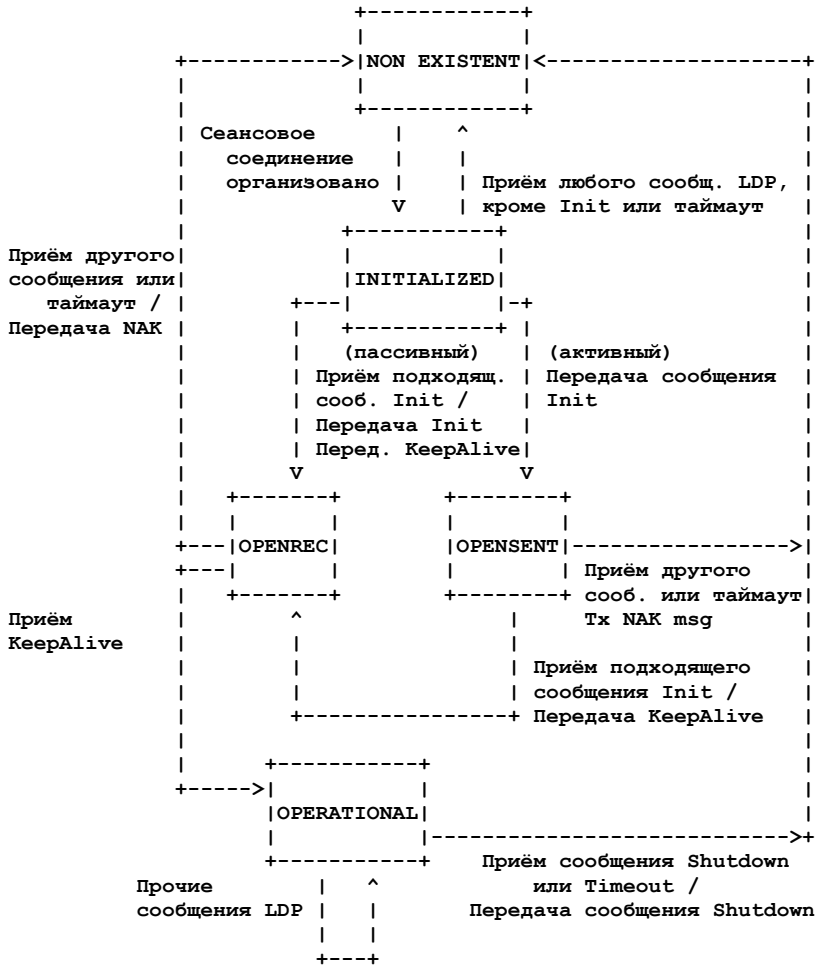


Диаграмма состояний при инициализации сессии LDP.

2.5.5. Поддержка Hello-смежности

Сессия LDP с партнёром включает одно или множество отношений Hello-смежности. Множество отношений смежности присутствует в тех случаях, когда пара LSR соединена множеством каналов с общими пространствами меток (например, множество соединений PPP между парой маршрутизаторов). В такой ситуации сообщения Hello, передаваемые LSR для каждого из каналов, содержат одинаковые значения LDP Identifier.

LDP включает механизмы мониторинга сессий LDP и Hello-смежности.

LDP использует обычные сообщения LDP Discovery Hello для индикации намерения партнёра использовать пространство меток, указанное в сообщении Hello. LSR поддерживает таймер удержания для каждой Hello-смежности, значение которого сбрасывается при получении нового сообщения Hello, соответствующего данной смежности. Если отсчет таймера завершается до получения соответствующего сообщения Hello от партнёра, LDP считает, что партнёр более не желает коммутировать по меткам с использованием данного пространства меток для этого канала (адресата в случае Targeted Hello) или канал разорван. После этого LSR удаляет отношение Hello-смежности. При удалении последнего отношения Hello-смежности для сессии LDP маршрутизатор LSR разрывает сессию LDP, передавая сообщение Notification и закрывая транспортное соединение.

2.5.6. Поддержка сессий LDP

LDP включает механизмы мониторинга целостности сессии LDP.

LDP использует получение обычных LDP PDU в транспортном соединении сессии для мониторинга целостности сеанса. LSR поддерживает для каждой сессии с партнёром таймер KeepAlive, который сбрасывается при получении LDP PDU в этой сессии. Если отсчет таймера KeepAlive завершается до получения от партнёра LDP PDU, маршрутизатор LSR считает, что транспортное соединение не работает должным образом или на стороне партнёра возник отказ в работе, после чего разрывает сессию LDP путём закрытия транспортного соединения.

После организации сеанса LDP маршрутизатор LSR должен удостовериться, что его партнёр получит LDP PDU не позже, чем у него завершится отсчет таймера KeepAlive, чтобы сброс таймера происходил нормально. LSR может для этого передать любое протокольное сообщение. Если никакой информации для передачи партнёру у LSR нет, он просто передаёт сообщение KeepAlive.

LSR может принять решение о разрыве сессии LDP со своим партнёром в любой момент. В этом случае партнёру просто передаётся сообщение Shutdown.

2.6. Распространение меток и управление ими

Архитектура MPLS [RFC3031] позволяет LSR распространять информацию о связывании меток с классом FEC в ответ на явный запрос со стороны другого LSR. Этот процесс называется нисходящим распространением меток по запросу[1]. Можно также распространять информацию о связывании меток маршрутизаторам LSR, которые явно не запрашивали её. В [RFC3031] такой метод называется незапрошенным нисходящим распространением[2].

Оба метода распространения меток могут одновременно использоваться в одной сети. Однако для любой конкретной сессии LDP каждый LSR должен знать используемый партнёром метод распространения меток, чтобы избежать ситуаций, когда один партнёр использует Downstream Unsolicited, предполагая, что другой поступает так же (см. параграф 3.5.7.1.3. Нисходящее анонсирование меток по запросам).

2.6.1. Режим управления LSP

Поведение на начальном этапе организации LSP определяется выбором независимого[3] или согласованного[4] режима управления LSP. LSR может поддерживать оба типа управления (параметр конфигурации).

2.6.1.1. Независимое управление

При независимом управлении LSP каждый LSR может анонсировать отображение меток своим соседям в любой желаемый момент. Например, при работе в режиме Downstream on Demand с независимым управлением LSR может незамедлительно ответить на запрос об отображении меток без ожидания такого отображения от следующего интервала маршрутизации (next hop). При независимом управлении в режиме Downstream Unsolicited маршрутизатор LSR может анонсировать отображение меток для FEC своим соседям, как только он будет готов к коммутации по меткам для данного FEC.

В результате использования независимого режима восходящие метки могут быть анонсированы раньше приёма нисходящих.

2.6.1.2. Согласованное управление

При согласованном управлении LSP маршрутизатор LSR может инициировать передачу отображения меток только тех классов FEC, для которых он имеет отображение на класс FEC для следующего интервала или для которых данных LSR является выходным. Для каждого класса FEC, по отношению к которому LSR не является выходным и нет отображения для следующего интервала, LSR **должен** дождаться приёма метки от нисходящего LSR прежде, чем отображать FEC и передавать соответствующие метки восходящим LSR. Маршрутизатор LSR может быть выходным для некоторых FEC и не являться таковым для других классов.

LSR может действовать в качестве выходного маршрутизатора по отношению к некому классу FEC при выполнении любого из условий:

1. FEC указывает непосредственно на этот LSR (включая непосредственно подключённые к нему интерфейсы);
2. маршрутизатор следующего интервала для этого FEC находится за пределами сети с коммутацией по меткам;
3. элементы FEC достижимы при пересечении границы домена маршрутизации (другая область OSPF, другая автономная система [RFC2328] [RFC4271]).

Отметим, что LSR, являющийся выходом пути для данного FEC, может меняться с течением времени в зависимости от состояния сети и настроек LSR.

2.6.2. Режим удержания меток

Архитектура MPLS [RFC3031] вводит понятие режима удержания меток. Удержанием называют поддержку LSR привязок меток для FEC, полученных от соседа, который не является следующим интервалом для данного FEC.

2.6.2.1. Консервативное удержание меток

В режиме анонсирования Downstream Unsolicited анонсы отображения меток для всех маршрутов могут быть получены от всех партнерских LSR. При использовании консервативного удержания меток[5], анонсируемые привязки меток удерживаются только в тех случаях, когда они будут применяться для пересылки пакетов (т. е. при получении привязок от маршрутизатора, который является корректным следующим интервалом в соответствии с таблицей маршрутизации). При работа в режиме Downstream on Demand маршрутизатор LSR будет запрашивать отображения меток только у маршрутизаторов LSR следующего интервала в соответствии с его таблицей маршрутизации. Поскольку режим Downstream on Demand используется прежде всего для сохранения меток (например, в коммутаторах ATM с ограниченным пространством кросс-соединений), с этим режимом обычно применяется консервативное удержание меток.

Основное преимущество консервативного режима удержания заключается в том, что выделяются и поддерживаются только метки, реально требуемые для пересылки данных. Это особенно важно для LSR с ограниченным пространством меток, таких, как коммутаторы ATM. Недостатком консервативного режима является то, что при изменении следующего интервала для данного получателя нужно сначала получить новую метку от следующего интервала и только потом будет возможна пересылка помеченных пакетов.

[1] Downstream On Demand.
[2] Unsolicited Downstream. В оригинале данного документа используется термин Downstream Unsolicited. *Прим. перев.*
[3] Independent LSP Control.
[4] Ordered LSP Control.
[5] Conservative Label retention.

2.6.2.2. Либеральное удержание меток

В режиме анонсирования Downstream Unsolicited анонсы отображения меток для всех маршрутов могут быть получены от всех LDP-партнеров. При либеральном удержании каждое отображение меток, полученное от партнерского LSR, сохраняется (удерживается), независимо от того, является ли этот LSR следующим интервалом для анонсируемого отображения. В режиме анонсирования Downstream on Demand с либеральным удержанием меток LSR может запрашивать отображения меток для всех известных префиксов от всех партнерских LSR. Отметим, однако, что режим Downstream on Demand обычно используется устройствами LSR типа коммутаторов ATM, для которых рекомендуется применять консервативное удержание.

Основным преимуществом либерального удержания меток является возможность более быстрого изменения маршрутов, поскольку метки для них уже имеются. Основным недостатком является поддержка и распространение ненужных (нетребуемых) меток.

2.6.3. Режим анонсирования меток

Каждый интерфейс LSR настраивается для работы в режиме анонсирования Downstream Unsolicited или Downstream on Demand. Маршрутизаторы LSR обмениваются информацией о режимах анонсирования в процессе инициализации. Основным различием между нисходящим анонсированием по запросам и без запросов является то, какой из LSR принимает на себя ответственность за инициирование запроса и анонсирования отображений.

2.7. Идентификаторы LDP и адреса Next Hop

LSR поддерживает полученные от других маршрутизаторов метки в базе данных LIB[1]. При работе в режиме Downstream Unsolicited запись LIB для адресного префикса связывает с префиксом набор пар (идентификатор LDP, метка) - одна пара для каждого партнёра, анонсирующего метку для данного префикса.

При изменении следующего интервала для префикса LSR должен отыскать метку, анонсируемую новым маршрутизатором следующего интервала, в своей базе LIB для использования этой метки при пересылке. Для поиска метки LSR должен иметь возможность отображения адреса следующего интервала для адресного префикса на идентификатор LDP.

Аналогично при получении маршрутизатором LSR метки для префикса от партнёра LDP маршрутизатор должен иметь возможность определить, является ли в настоящее время этот партнёр следующим интервалом для префикса и нужно ли начинать использование полученной от него метки для пересылки пакетов, соответствующих префиксу. Для принятия решения LSR должен быть способен отобразить идентификатор LDP на адрес партнёра, чтобы проверить, является ли этот партнёр следующим интервалом для префикса.

Для обеспечения возможности отображения между идентификатором LDP и адресом партнёра маршрутизаторы LSR анонсируют свои адреса с использованием сообщений Address и Withdraw Address.

LSR передаёт сообщение Address для анонсирования партнёру своего адреса, а для отзыва ранее анонсированного адреса LSR передаёт сообщение Withdraw Address.

2.8. Детектирование петель

Детектирование петель[2] является конфигурационной опцией, которая обеспечивает механизм нахождения замкнутых LSP и предотвращения зацикливания сообщений Label Request в присутствии LSR без поддержки слияния меток.

Механизм детектирования петель использует параметры Path Vector и Hop Count (TLV), передаваемые в сообщениях Label Request и Label Mapping. Механизм основан на рассмотренных ниже базовых свойствах этих TLV.

- Path Vector TLV содержит список LSR, через которые прошло содержащее параметр сообщение. LSR идентифицируются в списке Path Vector своими уникальными идентификаторами (LSR Id), которые представляют собой 4 первых октета значения LDP Identifier. Когда LSR распространяет[3] сообщение, содержащее Path Vector TLV, он добавляет своё значение LSR Id в список Path Vector. Маршрутизатор LSR, получивший сообщение со значением Path Vector, содержащим LSR Id этого маршрутизатора, делает вывод о том, что сообщение прошло по замкнутому пути (петле). Протокол LDP поддерживает максимально допустимую длину Path Vector - маршрутизатор LSR, обнаруживший, что значение Path Vector достигло предельной длины рассматривает этот факт, как обнаружение петли.

- Hop Count TLV содержит счётчик маршрутизаторов LSR, через которые прошло данное сообщение. Когда LSR распространяет сообщение с Hop Count TLV, он увеличивает значение счётчика на 1. LSR, который обнаруживает, что значение Hop Count достигло заданного в конфигурации максимума, рассматривает этот факт, как обнаружение петли. По соглашению нулевое значение счётчика интерпретируется, как неизвестное число интервалов. Инкрементирование неизвестного числа интервалов ведёт к тому, что значение сохраняется неизвестным (0)[4].

Процедуры детектирования петель LDP описаны в последующих параграфах. В этих (и только в этих) параграфах термин **«должен»** трактуется, как «**должен**, если включено детектирование петель». Эти параграфы описывают сообщения, которые **должны** переносить параметры Path Vector и Hop Count. Отметим, что Hop Count TLV и процедуры для работы с этим параметром используются без Path Vector TLV в тех случаях, когда детектирование петель не задано в конфигурации (см. [RFC3035] и [RFC3034]).

2.8.1. Сообщение Label Request

Использование Path Vector TLV и Hop Count TLV предотвращают «зацикливание» сообщений Label Request в средах, включающих LSR без поддержки слияния меток.

[1]Label Information Base - база информации о метках.
[2]Loop Detection.
[3]Генерирует и отправляет своё сообщение или пересылает чужое. *Прим. перев.*
[4]Объяснение этой кажущейся странности приведено в параграфе 3.4.4.1. Процедуры подсчёта интервалов. *Прим. перев.*

Правила использования Hop Count TLV в сообщениях Label Request LSR-маршрутизатором R со включённым детектированием петель имеют следующий вид:

- сообщение Label Request **должно** включать Hop Count TLV;
- если R передаёт сообщение Label Request потому, что он является входом FEC, это сообщение **должно** включать Hop Count TLV со значением счётчика 1;
- если R передаёт сообщение Label Request в результате получения Label Request от восходящего LSR и этот запрос содержит Hop Count TLV, маршрутизатор R **должен** увеличить значение счётчика на 1 и передать полученное значение счётчика в Hop Count TLV следующему интервалу пути доставки сообщения Label Request.

Правила использования Path Vector TLV в сообщениях Label Request LSR-маршрутизатором R со включённым детектированием петель имеют следующий вид:

- если R передаёт сообщение Label Request потому, что он является входом FEC, и R не поддерживает слияния меток, это сообщение **должно** включать Path Vector TLV размером 1, содержащее LSR Id данного маршрутизатора;
- если R передаёт сообщение Label Request в результате получения Label Request от восходящего LSR и этот запрос содержит Path Vector TLV или R не поддерживает слияния меток:

  R **должен** добавить своё значение LSR Id в Path Vector и **должен** передать полученный в результате параметр Path Vector на следующий интервал доставки сообщения Label Request. Если сообщение Label Request не содержит Path Vector TLV, маршрутизатор R **должен** включить Path Vector TLV размером 1 со своим идентификатором LSR Id.

Отметим, что в случаях, когда R получает сообщение Label Request для некого FEC и уже отправил сообщение Label Request для данного FEC на следующий интервал, но ещё не получил на это сообщение отклика и желает объединить недавно полученное сообщение Label Request со ждущим ответа запросом метки, маршрутизатор R не распространяет недавно полученное сообщение Label Request на следующий интервал.

Если R получает сообщение Label Request от следующего интервала и в этом сообщении имеется Hop Count TLV со значением счётчика, превышающим заданный максимум, Path Vector TLV, включающий LSR Id данного маршрутизатора, или превышающий заданный максимальный размер, R делает вывод о том, что данное сообщение Label Request оказалось в петле.

Когда R детектирует петлю, он **должен** передать сообщение Loop Detected Notification отправителю сообщения Label Request, отбросив последнее.

2.8.2. Сообщение Label Mapping

Использование Path Vector TLV и Hop Count TLV в сообщениях Label Mapping обеспечивает механизм детектирования и устранения петель в LSP. Когда LSR получает сообщение Label Mapping от маршрутизатора следующего интервала, это сообщение распространяется в восходящем направлении, как описано ниже, пока не будет достигнут входной LSR или обнаружена петля.

Правила использования Hop Count TLV в сообщениях Label Mapping, передаваемых LSR-маршрутизатором R при включённом режиме детектирования петель, имеют вид:

- R **должен** включать Hop Count TLV;
- если R является выходным, значение счётчика **должно** быть равным 1;
- если сообщение Label Mapping передаётся для распространения сообщения Label Mapping, полученного от следующего интервала, восходящему партнёру, значение счётчика **должно** устанавливаться как следующим образом:
  - если R является членом краевого набора LSR домена, в котором маршрутизаторы LSR не уменьшают значение TTL (например, домен ATM LSR или Frame Relay LSR) и восходящий партнёр находится в этом домене, R **должен** сбросить значение счётчика в 1 прежде, чем распространять сообщение;
  - в остальных случаях R **должен** увеличить значение счётчика на 1 перед дальнейшим распространением сообщения;
- если сообщение Label Mapping передаётся не для распространения Label Mapping, значение счётчика **должно** быть результатом увеличения на 1 текущего, известного R, значения счётчика интервалов, полученного из предыдущих сообщений Label Mapping. Отметим, что это значение счётчика не будет известно, если R не получал сообщений Label Mapping от следующего интервала.

Любое сообщение Label Mapping **может** включать Path Vector TLV. Правила обязательного использования Path Vector TLV в сообщениях Label Mapping, передаваемых LSR R при включённом детектировании петель, имеют следующий вид:

- если R является выходным, в сообщение Label Mapping не должно включаться Path Vector TLV;
- если R передаёт сообщение Label Mapping для распространения сообщения Label Mapping, полученного от партнёра, который является следующим интервалом в восходящем направлении:
  - если R поддерживает слияние меток и не отправлял ранее сообщений Label Mapping восходящему партнёру, он **должен** включить в сообщение Path Vector TLV;
  - если полученное сообщение содержит неизвестное значение счётчика интервалов, R **должен** включить в сообщение Path Vector TLV;

- если R ранее передавал восходящему партнёру сообщение Label Mapping, он **должен** включить Path Vector TLV; если полученное сообщение говорит об увеличении счётчика интервалов LSP, неизвестное значение счётчика заменяется известным, а известное - неизвестным.

Если в соответствии с приведёнными выше правилами R должен включать Path Vector TLV в сообщение Label Mapping значение параметра определяется следующим образом:

- Если полученное сообщение Label Mapping включает Path Vector, поле Path Vector, передаваемое в восходящем направлении, **должно** быть результатом добавления LSR Id маршрутизатора R к полученному полю Path Vector.
- Если в полученном сообщении нет поля Path Vector, передаваемое в восходящем направлении поле Path Vector должно иметь размер 1 и содержать значение LSR Id маршрутизатора R.

- Если сообщение Label Mapping передаётся не для распространения полученного сообщения Label Mapping в восходящем направлении, это сообщение **должно** включать поле Path Vector размера 1, содержащее LSR Id маршрутизатора R.

Если R получает от следующего интервала сообщение Label Mapping со значением Hop Count TLV, которое превышает заданный конфигурацией максимум, или полем Path Vector TLV, содержащим его значение LSR Id или превосходящим допустимый размер, R делает вывод о наличии петли в соответствующем LSP.

Когда маршрутизатор R детектирует петлю, он **должен** прекратить использование метки для пересылки, отбросить сообщение Label Mapping и сигнализировать о состоянии Loop Detected отправителю сообщения Label Mapping.

2.8.3. Обсуждение

Если в домене MPLS желательно использовать детектирование петель, эту функцию следует включить на **всех** LSR домена MPLS, поскольку в противном случае механизм Loop Detection не будет работать корректно и может давать ложные срабатывания или пропускать имеющиеся петли.

Для LSR, настроенных на поддержку Loop Detection, **не** предполагается сохранение полей Path Vector, как части состояния LSP.

Отметим, что в сети, где используются только LSR без поддержки слияния меток, поля Path Vector передаются в нисходящем направлении от входа к выходу и не передаются в восходящем направлении. Даже при поддержке слияния меток поля Path Vector не требуется передавать в восходящем направлении вдоль путей LSP, о которых известно, что они достигают выхода. Когда LSR сталкивается с изменением следующего интервала, ему нужно передавать Path Vector в восходящем направлении лишь в тех случаях, когда он не может сказать на основании счётчика интервалов, что изменение следующего интервала не приводит к возникновению петли.

В случае согласованного распространения меток сообщения Label Mapping распространяются от выхода в направлении входа, естественным образом создавая Path Vector вдоль пути. При независимом распространении меток LSR может создавать сообщение Label Mapping для FEC до получения Label Mapping от нисходящего партнёра для данного FEC. В этом случае последующие сообщения Label Mapping для FEC, полученные от нисходящего партнёра, трактуются, как обновление атрибутов LSP, и в восходящем направлении должно передаваться сообщение Label Mapping. В соответствии с этим рекомендуется настраивать детектирование петель в комбинации с согласованным распространением меток, чтобы минимизировать число сообщений Label Mapping об обновлениях.

2.9. Аутентичность и целостность сообщений LDP

В этом разделе описан механизм защиты от подставных сегментов TCP в потоке данных сеансов LDP. Использование этого механизма **должно** поддерживаться в форме конфигурационной опции.

Механизм защиты базируется на использовании цифровых подписей (сигнатур) TCP MD5, описанных в [RFC2385] для использования в BGP [RFC4271]. Спецификация функции хэширования MD5 приведена в [RFC1321]. С точки зрения завершённости стандарта данный документ опирается на [RFC2385] точно так же, как это делается в [RFC4271]. Объяснение этого дано в [RFC4278].

2.9.1. Сигнатуры TCP MD5

Приведённые ниже фрагменты[1] [RFC2385] очерчивают свойства защиты, обеспечиваемой за счёт использования сигнатур TCP MD5.

Замечание IESG

В этом документе описывается практика обеспечения безопасности протокола BGP от некоторых типов атак. Понятно, что эти меры не могут обеспечить должную безопасность в случаях согласованных атак.

Тезисы

В этом документе описывается расширение TCP для повышения уровня безопасности протокола BGP. Документ определяет новую опцию TCP для передачи цифровой подписи (digest) MD5 [RFC1321] в сегментах TCP. Такая подпись служит сигнатурой сегмента, включающей информацию, известную только конечным точкам соединения. Поскольку протокол BGP использует транспорт TCP, применение этой опции описанным в документе способом существенно снижает уровень риска для некоторых типов атак на BGP.

Введение

Основным мотивом добавления этой опции является стремление обеспечить протоколу BGP средства самозащиты против вставки обманных сегментов TCP в поток данных через соединение. Особую важность имеют случаи сброса (reset) соединений TCP.

[1]Документ цитируется по переводу RFC 2385. *Прим. перев.*

Для обмана соединения в случае использования описанной здесь схемы атакующему нужно не только предсказать порядковые номера TCP, но и узнать пароль, включенный в цифровую подпись MD5. Этот пароль не передаётся в потоке соединения и реальная форма пароля определяется приложением. Пароль даже может быть сменен во время работы соединения, если эта замена согласована обеими сторонами (отметим, что для некоторых реализаций TCP смена пароля может приводить к возникновению проблем при повторе передачи).

Важно также отметить, что использование этой опции не требует согласования - каждый сайт волен сам решить, будет ли он использовать данную опцию.

...

MD5 в качестве алгоритма хэширования

К моменту выпуска первого варианта этого документа (он имел другое название) в алгоритме MD5 была обнаружена уязвимость для атак с целью поиска коллизий [Dobb] и возникли сомнения в его достаточной надёжности для предлагаемого здесь использования.

В данном документе по-прежнему указывается алгоритм MD5, однако, в силу того, что опция уже используется на практике, в неё не включено поле типа алгоритма, чтобы впоследствии можно было заменить алгоритм без смены номера опции. В исходном документе также не было задано поле типа, поскольку оно потребовало бы по крайней мере 1 байта и полный размер опции стал бы не менее 19 байтов (которые могли бы дополняться до 20 реализациями TCP), что могло бы оказаться неприемлемым с учётом ограниченности размера заголовка.

Это не мешает разработке аналогичных опций с использованием другого алгоритма хэширования (например, SHA-1). Поскольку большинство реализаций дополняют 18 байтов опции до 20, не возникает проблем с определением новой опции, включающей поле типа алгоритма.

Однако рассмотрение этих вопросов требует подготовки отдельного документа.

Конец цитаты из [RFC2385].

2.9.2. Использование опции TCP MD5 Signature в LDP

LDP использует опцию TCP MD5 Signature следующим образом:

- Опция MD5 Signature используется для TCP-соединений LDP, как конфигурационная опция LSR.
- LSR, использующий MD5 Signature, настраивается с паролем (разделяемый секрет) для каждого потенциального партнёра LDP.
- LSR применяет алгоритм MD5 в соответствии с [RFC2385] для расчёта цифровых подписей MD5 сегментов TCP, передаваемых партнёру. При расчёте цифровой подписи используется пароль партнёра и сегмент TCP.
- Когда LSR получает сегмент TCP с сигнатурой MD5, он заново рассчитывает сигнатуру MD5 (используя свою копию пароля) и сравнивает полученное значение с принятой сигнатурой. При наличии расхождений сегмент отбрасывается без уведомления отправителя.
- LSR игнорирует сообщения LDP Hello от любых LSR, для которых ему не известен пароль. Это позволяет LSR организовывать TCP-соединения LDP только с маршрутизаторами LSR, для которых пароль задан.

2.10. Распространение меток для LSP с явной маршрутизацией

Предполагается, что построение трафика[1] [RFC2702] будет одним из важных применений MPLS. Поддержка построения трафика в MPLS использует явно маршрутизируемые LSP, которые не следуют нормально (поэтапно) маршрутизируемым путям, определяемым протоколами маршрутизации по адресу получателя. CR-LDP [CRLDP] определяет расширение LDP для использования протокола LDP с целью организации явно маршрутизируемых LSP.

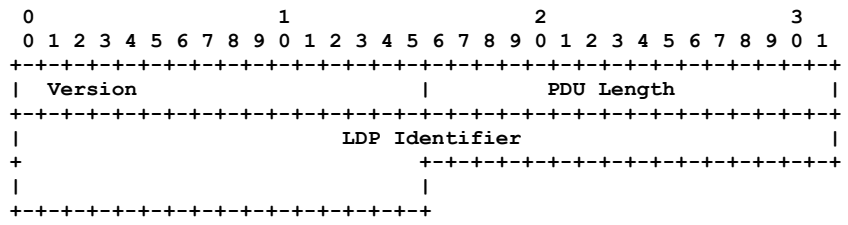
3. Спецификация протокола

В предыдущих параграфах при описании работы LDP рассматривались сценарии, включающие обмен сообщениями между партнёрами LDP. В этом разделе обсуждается представление сообщений и процедуры их обработки.

Обмен сообщениями LDP реализуется путём передачи модулей данных протокола (LDP PDU[2]) через TCP-соединения сеансов LDP.

Каждый LDP PDU может содержать одно или множество сообщений LDP. Отметим, что сообщения в LDP PDU могут быть не связаны между собой. Например, в одном PDU может содержаться сообщение, анонсирующее связывание метки с FEC для нескольких классов FEC, сообщение, запрашивающее связки с метками для некоторых других FEC, и сообщение Notification с информацией о том или ином событии.

3.1. LDP PDU



Каждый LDP PDU имеет заголовок LDP, за которым следует одно или множество сообщений LDP. Заголовок LDP включает указанные ниже поля.

[1]Traffic Engineering.
[2]Protocol data unit.

Version - версия

Двухоктетное целое число без знака, показывающее номер версии протокола. Данная версия спецификации задаёт протокол LDP версии 1.

PDU Length - размер PDU

Двухоктетное целое число, задающее общий размер PDU в октетах без учёта полей Version и PDU Length.

Максимально допустимое значение PDU Length согласуется при инициализации сеанса LDP. До завершения инициализации размер ограничен значением 4096 байта.

LDP Identifier - идентификатор PDU

Шестиоктетное поле, уникально идентифицирующее пространство меток передающего LSR, к которому этот PDU имеет отношение. Первые 4 октета идентифицируют LSR и **должны** быть уникальными в глобальном масштабе. В качестве этой части идентификатора **следует** использовать 32-битовый идентификатор маршрутизатора (Router Id), присвоенный LSR. Эта часть идентификатора используется также для детектирования петель. Два оставшихся октета идентифицируют пространство меток в рамках LSR. Для пространства меток масштаба платформы **следует** использовать нулевое значение этих октетов.

Отметим, что для первого октета LDP PDU не требуется выравнивание по границе.

3.2. Процедуры LDP

LDP определяет сообщения, TLV и процедуры для нескольких типов действий:

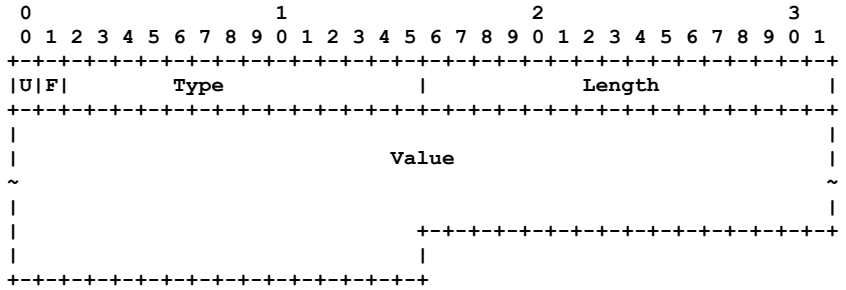
- обнаружение партнёров;
- управление сессией;
- распространение меток;
- уведомления об ошибках и другая информация.

В последующих параграфах описано представление сообщений и TLV для перечисленных типов действий и применимые к ним процедуры.

Процедуры распространения меток сложны и их достаточно трудно описать полно, последовательно и однозначно, как набор спецификаций отдельных сообщений и TLV.

Приложение A. Процедуры распространения меток LDP описывает распространение меток в терминах событий, которые могут происходить в LSR, и откликов LSR на такие события. Приложение A является спецификацией процедур распространения меток в LDP. Если какое-то из описаний процедуры в этом документе противоречит приложению A, поведение LDP должно соответствовать приложению A.

3.3. Представление TLV



LDP использует формат TLV для представления информации, передаваемой в сообщениях LDP.

LDP TLV кодируется, как двухоктетное поле, в котором 14 битов служат для задания типа и 2 бита задают поведение LSR при получении неизвестного типа, далее следует 2-октетное поле размера и поле Value переменной длины.

U-bit

Флаг обработки неизвестных TLV. При получении неизвестного TLV со сброшенным (0) битом U **должно** возвращаться уведомление инициатору сообщения и все сообщение **должно** игнорироваться. Если U=1, неизвестный TLV **должен** игнорироваться, а остальная часть сообщения обрабатывается, как обычно. В последующих параграфах, определяющих TLV, задается значение U-бита.

F-bit

Флаг пересылки неизвестных TLV. Этот флаг имеет значение только для сообщений с установленным флагом U, содержащих неизвестный TLV. Если F=0, неизвестный TLV не пересылается с содержащим его сообщением, а при установленном флаге (F=1) неизвестный TLV пересылается с содержащим его сообщением. В последующих параграфах, определяющих TLV, указывается значение бита F. При установке обоих флагов U и F TLV может распространяться через узлы, не понимающие данный TLV, как неинтерпретируемые данные.

Type - тип

Определяет интерпретацию поля Value.

Length - размер

Указывает размер поля Value в октетах.

Value - значение

Строка размером Length октетов, представляющая информацию, которая интерпретируется в соответствии со значением поля Type.

Отметим, что для первого октета TLV не предъявляется требований по выравниванию.

Отметим также, что поле Value само может содержать TLV (т. е., TLV могут быть вложенными).

Схема представления TLV является весьма общей. В принципе все, что появляется в LDP PDU, можно представить в форме TLV. Однако данная спецификация не применяет схему TLV во всех случаях. Эта схема не используется там, где общность не требуется, а применение этой схемы привело бы к нерациональному расходу пространства. Это

обычно возникает в ситуациях, когда тип значения известен заранее (например, по его положению в сообщении или «объемлющем» TLV), а размер значения фиксирован или легко определяется из самого значения.

Некоторые TLV, определённые для LDP, похожи друг на друга. Например, имеются Generic Label TLV, ATM Label TLV, и Frame Relay TLV (см. параграфы 3.4.2.1. TLV меток общего назначения, 3.4.2.2. TLV меток ATM , 3.4.2.3. TLV для меток Frame Relay).

Когда можно думать об иерархии TLV в терминах типа TLV, который задаёт класс TLV, и подтипа TLV, задающего определённый вид TLV данного класса, в этой спецификации не формализуется нотация подтипа TLV.

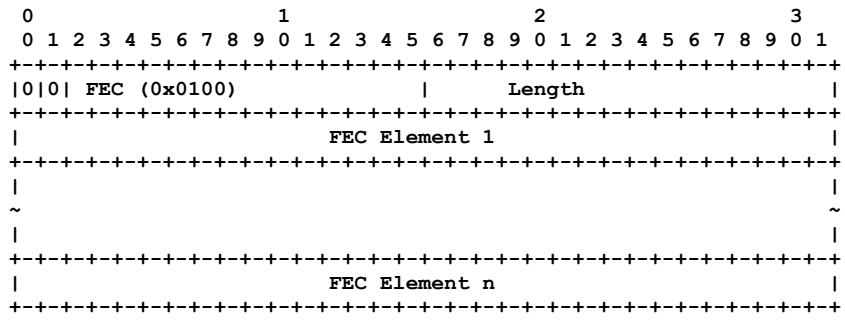
Спецификация выделяет значения типов для соответствующих TLV (таких, как TLV меток) из непрерывного пространства 16-битовых номеров типа TLV.

В параграфе 3.8. Список TLV перечислены TLV, определённые для этой версии протокола, и указаны параграфы данного документа, в которых описаны соответствующие TLV.

3.4. Представление TLV для параметров общего назначения

Существует несколько параметров, используемых множеством сообщений LDP. В этом параграфе описано представление таких параметров в форме TLV.

3.4.1. FEC TLV



Метки привязываются к классам эквивалентной пересылки (FEC[1]). FEC представляет собой список из одного или множества элементов FEC. FEC TLV представляет компоненты FEC.

Формат FEC TLV показан на рисунке справа.

FEC Element 1 - FEC Element n

Имеется несколько типов элементов FEC (см. параграф 2.1. Классы FEC). Представление элементов FEC зависит от их типа.

Значение FEC Element представляется, как 1-октетное поле, задающее тип элемента, и поле переменного размера, содержащее зависимое от типа значение элемента. Отметим, что, несмотря на зависимость кодирования элементов FEC от типа, их представление является одним из случаев, когда стандартное кодирование LDP TLV не используется.

Кодирование значений FEC Element имеет вид:

Имя типа FEC Element	Тип	Значение
Wildcard	0x01	Нет значения (0 октетов), см. ниже.
Prefix	0x02	См. ниже.

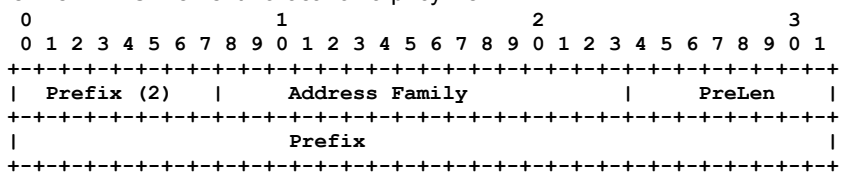
Отметим, что данная версия LDP поддерживает использование множества FEC Element в одном FEC только для сообщений Label Mapping. Использование множества элементов FEC в сообщениях других типов не разрешается для этой версии и требует дальнейшего изучения.

Wildcard FEC Element

Для использования только в сообщениях Label Withdraw и Label Release. Показывает, что отзыв/освобождение будет применяться ко всем FEC, связанным с меткой в следующем TLV для метки. Этот элемент должен быть единственным элементом FEC в FEC TLV.

Prefix FEC Element

Представление Prefix FEC Element показано на рисунке.



Address Family

Двухоктетное поле, содержащее значение из числа ADDRESS FAMILY NUMBER, определённых в [ASSIGNED_AF], которое представляет семейство адресов для префикса в поле Prefix.

PreLen

Однооктетное целое число без знака, показывающее размер в битах следующего за ним адресного префикса. Нулевой размер указывает на префикс, который соответствует всем адресам (получатель, используемый по умолчанию). В этом случае поле Prefix содержит 0 октетов (пусто).

Prefix

Адресный префикс, кодируемый в соответствии со значением поля Address Family. Размер префикса в битах задается полем PreLen, для префикса используется дополнение нулями с целью выравнивания по границе байта.

[1]Forwarding Equivalence Class.

3.4.1.1. Процедуры FEC

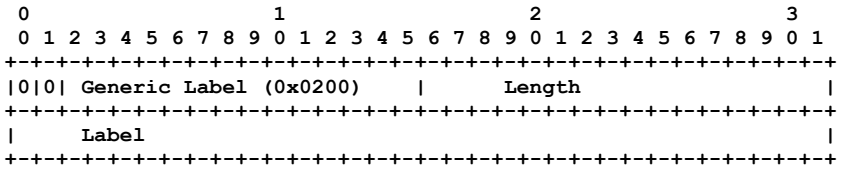
Если декодирующий FEC TLV маршрутизатор LSR встречает FEC Element с неподдерживаемым значением Address Family, ему **следует** остановить декодирование FEC TLV, прервать обработку сообщения, содержащего TLV, и передать уведомление Unsupported Address Family своему партнёру LDP для сигнализации ошибки.

Если встречается FEC Element, тип которого не удаётся декодировать, **следует** прекратить декодирование FEC TLV, прервать обработку сообщения, содержащего TLV, и передать уведомление Unknown FEC своему партнёру LDP для сигнализации ошибки.

3.4.2. TLV для меток

Label TLV используется для представления меток. Структуры Label TLV передаются в сообщениях, используемых для анонсирования, запроса, освобождения и отзыва отображений меток.

Существует несколько типов Label TLV, которые могут появляться в ситуациях, требующих использования Label TLV.

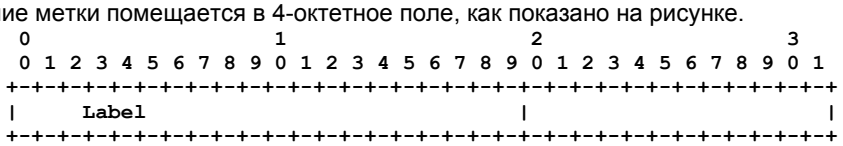


3.4.2.1. TLV меток общего назначения

LSR использует Generic Label TLV для представления меток, применяемых на каналах, где метки не зависят от технологии нижележащего уровня. Примерами таких каналов могут служить PPP и Ethernet.

Label

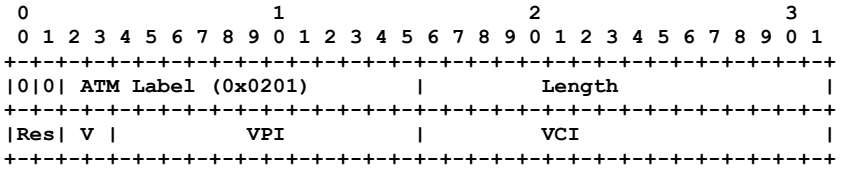
20-битовое значение метки помещается в 4-октетное поле, как показано на рисунке.



Дополнительную информацию можно найти в [RFC3032].

3.4.2.2. TLV меток ATM

LSR используют ATM Label TLV для представления меток, применяемых на каналах ATM.



Res

Это поле является резервным. При передаче **должно** устанавливаться нулевое значение, на приёмной стороне поле **должно** игнорироваться.

V

Двухбитовый индикатор типа коммутации. Значение 00 говорит, что значимы оба значения VPI и VCI. Если V = 01, значимость имеет только VPI, при V = 10 - только VCI.

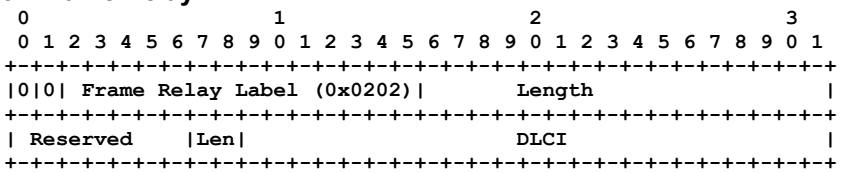
VPI

Идентификатор виртуального пути. Если значение VPI имеет размер менее 12 битов, его **следует** выравнивать по правому краю поля, дополняя слева нулями.

VCI

Идентификатор виртуального канала. Если значение VCI имеет размер менее 16 битов, его **следует** выравнивать по правому краю поля, а свободные биты слева **должны** устанавливаться в 0. Если флаг V задаёт коммутацию только по виртуальным путям, это поле **должно** игнорироваться получателем и устанавливаться в 0 отправителем.

3.4.2.3. TLV для меток Frame Relay



LSR используют Frame Relay Label TLV для представления меток, применяемых на каналах Frame Relay.

Res

Это поле является резервным. При передаче **должно** устанавливаться нулевое значение, на приёмной стороне поле **должно** игнорироваться.

Len

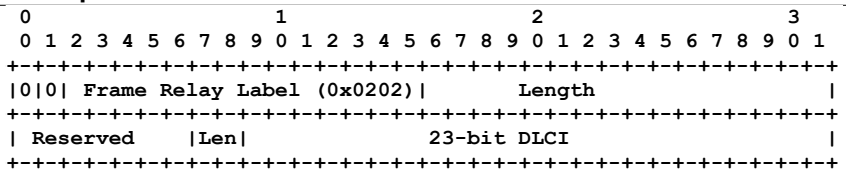
Это поле задаёт число битов идентификатора DLCI. Поддерживается два варианта значения поля:

0 = размер DLCI равен 10 битам;
2 = размер DLCI равен 23 битам.

Значения 1 и 3 являются резервными.

DLCI

Идентификатор соединения на канальном уровне.
10-битовые значения DLCI представляются, как показано на рисунке.
Представление 23-битовых DLCI показано на рисунке ниже.

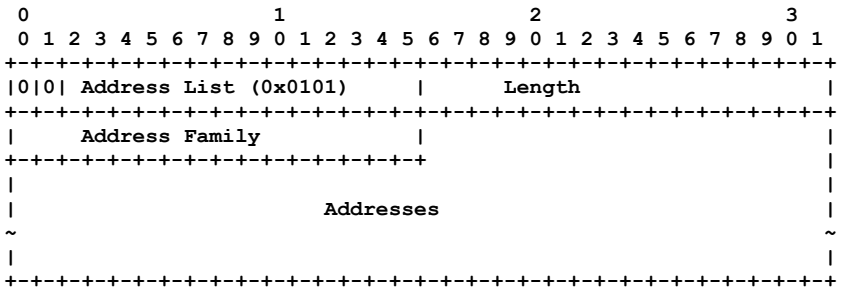


Дополнительную информацию можно найти в [RFC3034].

3.4.3. TLV для списка адресов

Структура Address List TLV появляется в сообщениях Address и Address Withdraw.

Формат TLV показан на рисунке, поля описаны ниже.



Address Family

Двухоктетное поле, содержащее значение из списка ADDRESS FAMILY NUMBERS в документе [ASSIGNED_AF], которое указывает тип адреса в поле Addresses.

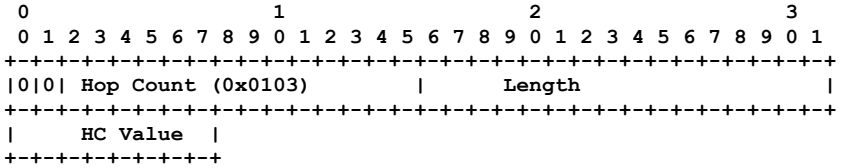
Addresses

Список адресов заданного полем Address Family типа. Представление адресов зависит от значения поля Address Family.

Представление адресов, определённое данной спецификацией, показано в таблице.

Address Family	*Представление адреса*
IPv4	Все 4 октета адреса IPv4
IPv6	Все 16 октетов адреса IPv6

3.4.4. TLV для счётчика интервалов



Структура Hop Count TLV появляется в необязательном поле сообщений, передаваемых при организации LSP. Она служит для учёта числа интервалов LSR на пути LSP при организации последнего.

Отметим, что процедуры организации LSP, проходящих через сети ATM и Frame Relay, требуют использования Hop Count TLV (см. [RFC3035] и [RFC3034]).

HC Value

Однооктетное целое число без знака, содержащее значение счётчика интервалов.

3.4.4.1. Процедуры подсчёта интервалов

При организации LSP маршрутизатор R может получить сообщение Label Mapping или Label Request для LSP, которое содержит Hop Count TLV. В таком случае **следует** записать значение счётчика.

Если LSR R распространяет сообщение Label Mapping для LSP своему восходящему партнёру или сообщение Label Request - нисходящему для продолжения процедуры организации LSP, необходимо определять значение счётчика интервалов для включения в распространяемое сообщение, как показано ниже:

- для сообщение Label Request маршрутизатор R **должен** увеличить значение счётчика на 1;
- для сообщений Label Mapping маршрутизатор R определяет значение счётчика следующим образом:
  - если R является членом множества краевых LSR домена, в котором LSR не увеличивают значение TTL, и восходящий партнёр входит в этот домен, маршрутизатор R **должен** сбросить значение счётчика в 1 перед дальнейшим распространением сообщения;
  - в остальных случаях R **должен** увеличить значение счётчика на 1.

Первому LSR в LSP (входной для сообщения Label Request, выходной для Label Mapping) **следует** установить для счётчика интервалов значение 1.

По договорённости, значение 0 говорит о неизвестном значении счётчика интервалов. В результате инкрементирования неизвестного значения счётчика оно сохраняется неизвестным (0).

Использование неизвестного значения счётчика существенно снижает объем сигнальной информации при использовании режима независимого управления. При создании нового LSP каждый LSR начинает с неизвестного значения счётчика. Добавление LSR с неизвестным значением счётчика приводит к тому, что значение счётчика сохраняется неизвестным. При добавлении в LSP выходного маршрутизатора LSR распространяет обновление значения счётчика в восходящем направлении с помощью сообщений Label Mapping.

Если не использовать неизвестное значение счётчика, то при каждом добавлении LSR в путь LSP потребуется распространять обновление значения счётчика в восходящем направлении, если новый LSR расположен ближе к выходу, чем остальные LSR. Эти обновления бесполезны, поскольку они не отражают значение счётчика на выходе пути.

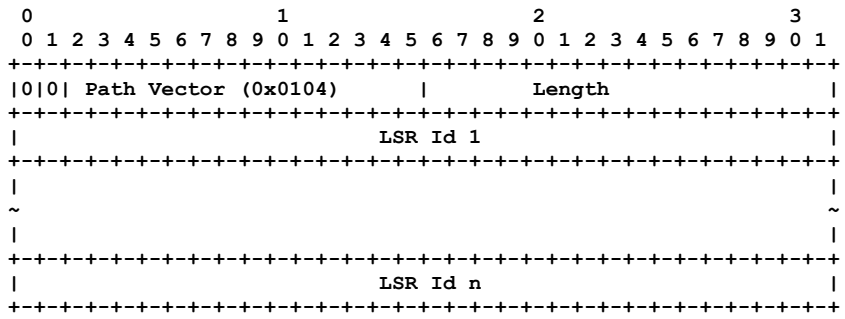
С точки зрения входного узла неизвестное значение счётчика интервалов не говорит о том, что пакет, направленный в LSP будет попадать на выход. Это означает лишь, что обновление счётчика интервалов от выходного узла ещё не достигло входного узла.

Если LSR получает сообщение с Hop Count TLV, он **должен** проверить значение счётчика интервалов, чтобы определить, не превосходит ли это значение допустимого конфигурацией максимума. Если значение счётчика превышает дозволенное, маршрутизатор **должен** сделать вывод о том, что сообщение прошло через петлю и передать его отправителю уведомление о наличии петли (Loop Detected).

Если включён режим Loop Detection, маршрутизатор LSR **должен** следовать процедурам, описанным в параграфе 2.8. Детектирование петель.

3.4.5. TLV для вектора пути

Path Vector TLV используется вместе с Hop Count TLV в сообщениях Label Request и Label Mapping для реализации необязательного механизма LDP Loop Detection (см. параграф 2.8. Детектирование петель). При его использовании в сообщениях Label Request записывается путь в форме списка LSR, через которые проходит запрос. Для сообщений Label Mapping записывается набор LSR, через которые проходит анонс метки для организации LSP. Формат структуры показан на рисунке.



LSR Id

Список идентификаторов LSR, показывающий путь, по которому проходит сообщение. Каждое значение LSR Id в списке представляет собой 4 первых октета (router-id) идентификатора LDP для соответствующего LSR. Такая идентификация LSR обеспечивает однозначное распознавание маршрутизатора в пределах сети.

3.4.5.1. Процедуры Path Vector

Структура Path Vector TLV передаётся в сообщениях Label Mapping и Label Request при включённом режиме Loop Detection.

3.4.5.1.1. Path Vector в сообщении Label Request

В параграфе 2.8. Детектирование петель указаны ситуации, когда LSR должен включать Path Vector TLV в сообщения Label Request.

LSR, получивший Path Vector в сообщении Label Request, **должен** выполнить процедуры, описанные в параграфе 2.8. Детектирование петель.

Если LSR обнаруживает петлю, он **должен** отвергнуть сообщение Label Request. LSR в таких случаях также **должен**:

1. отправить передающему LSR сообщение с сигналом Loop Detected;
2. не распространять дальше сообщение Label Request.

Отметим, что сообщение Label Request с Path Vector TLV пересылается, пока не будет выполнено одно из условий:

1. обнаружена петля;
2. достигнут выход LSP;
3. достигнут максимальный размер Path Vector или максимальное значение счётчика Hop Count (это трактуется, как обнаружение петли).

3.4.5.1.2. Path Vector в сообщении Label Mapping

В параграфе 2.8. Детектирование петель указаны ситуации, когда LSR должен включать Path Vector TLV в сообщения Label Mapping.

LSR, получивший Path Vector в сообщении Label Mapping, **должен** выполнить процедуры, описанные в параграфе 2.8. Детектирование петель.

Если LSR обнаруживает петлю, он **должен** отвергнуть сообщение Label Mapping, чтобы предотвратить пересылку метки, связанной с петлей. LSR в таких случаях также **должен**:

1. отправить сообщение Label Release, содержащее Status TLV (сигнал Loop Detected) передавшему исходное сообщение маршрутизатору LSR;
2. не распространять дальше сообщение Label Mapping;
3. проверить, что сообщение Label Mapping относится к существующему LSP; при положительном результате проверки LSR должен отсоединить все восходящие метки, которые соединены с нисходящей меткой для FEC.

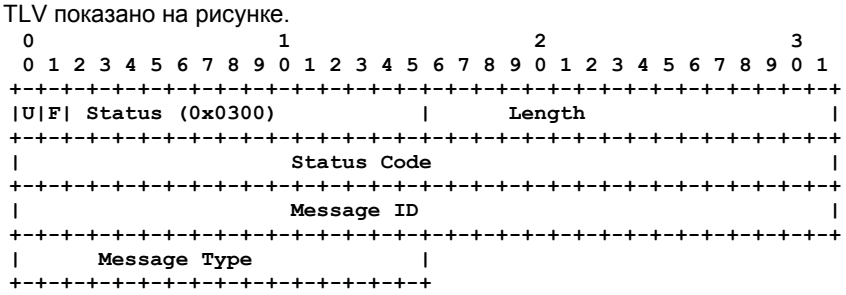
Отметим, что сообщение Label Mapping с Path Vector TLV пересылается, пока не будет выполнено одно из условий:

1. обнаружена петля;
2. достигнут вход LSP;
3. достигнут максимальный размер Path Vector или максимальное значение счётчика Hop Count (это трактуется, как обнаружение петли).

3.4.6. Status TLV

Сообщения Notification используют Status TLV для указания событий, к которым относятся сигналы.

Представление Status TLV показано на рисунке.



U

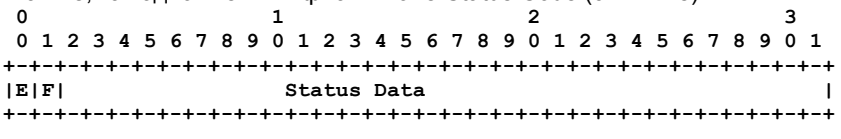
Для этого флага **следует** устанавливать значение 0, когда Status TLV передаётся в сообщении Notification. При передаче Status TLV в сообщениях других типов для флага **следует** задавать значение 1.

F

Этот флаг **следует** использовать так же, как одноимённый флаг в поле Status Code (см. ниже).

Status Code

32-битовое целое число без знака, указывающее код события, о котором передаётся сигнал. Структура поля кода показана на рисунке справа.



E

Флаг критической ошибки. Если этот флаг установлен (1), сообщение относится к критическим сигналам Error Notification. При сброшенном (0) флаге - это Advisory Notification.

F

Флаг управления пересылкой. При установленном (1) флаге уведомление **следует** пересылать LSR следующего или предыдущего интервала LSP (если он есть), связанному с событием, о котором говорит сигнал. При сброшенном (0) флаге уведомление **не следует** пересылать.

Status Data

30-битовое целое число без знака, показывающее информацию о состоянии.

Эта спецификация определяет коды состояния (32-битовое целое число без знака с описанным выше представлением).

Значение Status Code = 0 говорит об успешном выполнении.

Message ID

Отличное от нуля 32-битовое целое число, идентифицирующее сообщение партнёра, на которое указывает Status TLV. Нулевое значение говорит о том, что сообщение партнёра не идентифицировано.

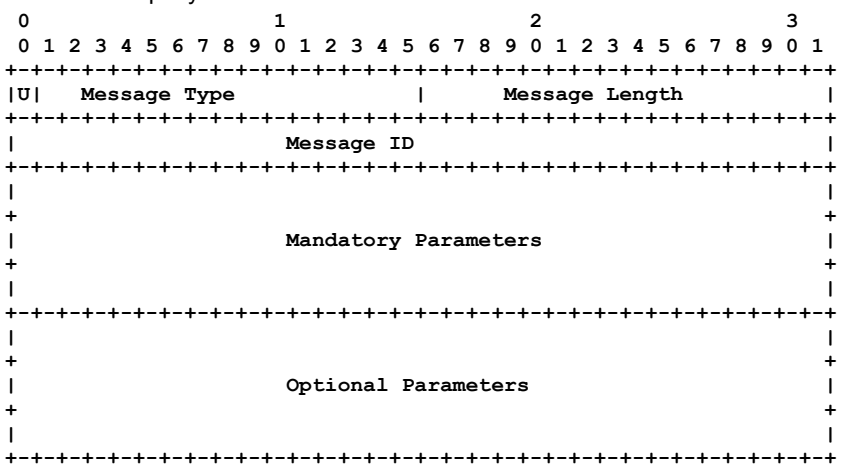
Message Type

Отличное от нуля значение показывает тип партнерского сообщения, к которому относится Status TLV. При нулевом значении Status TLV не указывает на какой-либо конкретный тип сообщения.

Отметим, что использование Status TLV не ограничено сообщениями Notification. Сообщения других типов могут включать Status TLV в качестве необязательного параметра. При передаче в сообщениях, отличных от Notification, для поля U в Status TLV **следует** устанавливать значение 1, показывающее, что получателю **следует** отбросить данный TLV без уведомления, если он не может его обработать.

3.5. Сообщения LDP

Формат сообщения LDP показан на рисунке.



U

Флаг неизвестного сообщения. При получении неизвестного сообщения со сброшенным флагом U отправителю этого сообщения передаётся уведомление; если же в неизвестном сообщении установлен флаг U, такое сообщение просто игнорируется. В последующих параграфах, определяющих сообщения, указывается значение бита U.

Message Type

Идентифицирует тип сообщения.

Message Length

Показывает суммарный размер полей Message ID, Mandatory Parameters и Optional Parameters в октетах.

Message ID

32-битовое значение, служащее для идентификации данного сообщения. Это поле используется передающим LSR для идентификации сообщений Notification, которые могут относиться к данному сообщению. Маршрутизатору LSR, передающему сообщение Notification в ответ на данное сообщение, **следует** включить это значение Message ID в Status TLV, передаваемое в сообщении Notification (см. параграф 3.5.1. Сообщение Notification).

Mandatory Parameters

Набор обязательных параметров сообщения (переменный размер). Для некоторых сообщений параметры могут не требоваться.

Для сообщений, имеющих обязательные параметры, эти параметры **должны** указываться в порядке, заданном спецификацией конкретных сообщений в последующих параграфах.

Optional Parameters

Набор необязательных параметров (переменный размер). Для многих сообщений необязательные параметры не применяются.

Для сообщений с необязательными параметрами эти параметры могут указываться в произвольном порядке.

Отметим, что для первого октета сообщения LDP выравнивание не требуется и в конце сообщения не используется заполнение - т. е., параметры могут заканчиваться на границе нечётного байта.

В таблице показаны типы сообщений, определённые для текущей версии LDP.

Имя сообщения	Заголовок раздела
Notification	Сообщение Notification
Hello	Сообщение Hello
Initialization	Сообщение Initialization
KeepAlive	Сообщение KeepAlive
Address	Сообщение Address
Address Withdraw	Сообщение Address Withdraw
Label Mapping	Сообщение Label Mapping
Label Request	Сообщение Label Request
Label Abort Request	Сообщение Label Abort Request
Label Withdraw	Сообщение Label Withdraw
Label Release	Сообщение Label Release

В последующих параграфах описано представление и процедуры для этих сообщений.

Некоторые из приведённых в таблице сообщений могут быть связаны с другими сообщениями - например, Label Mapping, Label Request, Label Withdraw и Label Release.

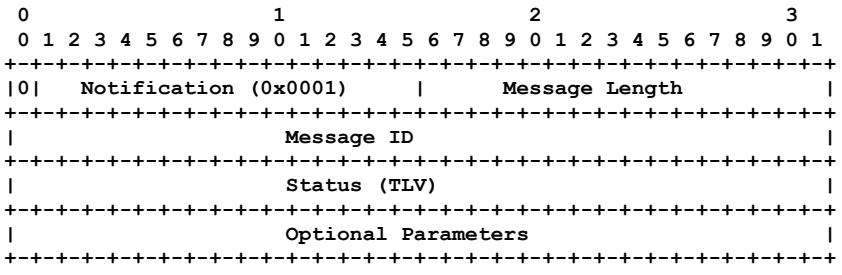
Хотя о таких связанных сообщениях можно говорить в терминах классов и субтипов сообщений, задающих конкретные сообщения данного класса, в этой спецификации не используется нотация с субтипами сообщений.

Спецификация выделяет значения типа для связанных сообщений (таких, как Label) из непрерывного блока 16-битового пространства номеров типов сообщений.

3.5.1. Сообщение Notification

LSR передаёт сообщение Notification для информирования LDP-партнера о значимом событии. Сообщение Notification сигнализирует о критической ошибке или содержит справочную информацию типа результата обработки сообщения LDP или состояния сессии LDP.

Формат сообщения Notification показан на рисунке.



Message ID

32-битовый идентификатор сообщения.

Status TLV

Указывает событие, о котором сигнализирует данное сообщение. Представление StatusTLV описано в параграфе 3.4.6. Status TLV.

Optional Parameters

Это поле переменного размера содержит 0 или более параметров в формате TLV. В таблице справа показаны базовые необязательные параметры, которые могут включаться в любые сообщения Notification.

Для конкретных событий, о которых сигнализируют сообщения Notification, могут использоваться другие необязательные параметры, описанные в соответствующих параграфах.

Необязательный параметр	Тип	Размер	Значение
Extended Status	0x0301	4	См. ниже
Returned PDU	0x0302	переменный	См. ниже
Returned Message	0x0303	переменный	См. ниже

Extended Status

4-октетное значение расширенного кода состояния представляет информацию о состоянии, служащую дополнением к сведениям, содержащимся в поле Status Code сообщения Notification.

Returned PDU

LSR использует этот параметр для возврата части LDP PDU отправившему его маршрутизатору LSR. Значение этого TLV представляет собой заголовок PDU и то количество данных PDU после заголовка, которое соответствует условию, о котором сигнализирует сообщение Notification.

Returned Message

LSR использует этот параметр для возврата части LDP PDU отправившему его маршрутизатору LSR. Значение этого TLV представляет собой поля типа и размера сообщения, а также следующие за ними данные в объёме, соответствующем условию, о котором сигнализирует сообщение Notification.

3.5.1.1. Процедуры для сообщений Notification

Если LSR сталкивается с условиями, когда требуется уведомить партнёра (консультация или информация об ошибке), партнёру передаётся сообщение Notification, содержащее поле Status TLV, которое представляет передаваемую информацию и может включать дополнительные TLV с расширенной информацией об условиях.

Если условие связано с критической ошибкой, об этой ошибке будет говорить значение Status Code в сообщении Notification. В таких случаях после передачи сообщения Notification маршрутизатору LSR **следует** прервать сессию LDP путём закрытия транспортного соединения TCP и отбросить все данные о состоянии, связанные с этой сессией, включая все привязки меток к FEC, полученные в этой сессии.

Когда LSR получает сообщение Notification со значением Status Code, показывающим критическую ошибку, ему **следует** незамедлительно прервать сессию LDP путём разрыва соединения TCP и отбросить все связанные с этой сессией состояния, включая все привязки меток к FEC, полученные в этой сессии.

Приведённое выше требование не относится к обработке сообщения Shutdown в процедуре инициализации сессии. Когда LSR получает сообщение Shutdown в процессе инициализации сессии, ему **следует** передать сообщение Shutdown и закрыть транспортное соединение.

3.5.1.2. События, о которых сигнализируют сообщения Notification

Для более наглядного описания события, о которых сообщается с помощью сообщений Notification, делятся на несколько категорий.

3.5.1.2.1. PDU или сообщение с некорректным форматом

LDP PDU с некорректным форматом или сообщения, являющиеся частью механизма LDP Discovery, отбрасываются без уведомления.

LDP PDU полученные через соединение TCP для сессии LDP, считаются некорректными по форме, если:

- значение LDP Identifier в заголовке PDU не известно получателю или известный получателю идентификатор не является значением LDP Identifier, связанным с партнёром LDP для этой сессии LDP; такая ошибка является критической и указывается значением Bad LDP Identifier в поле Status Code;
- версия протокола LDP не поддерживается получателем или не совпадает с версией, согласованной на этапе организации сессии; такая ошибка является критической и указывается значением Bad Protocol Version в поле Status Code;
- значение поля PDU Length слишком мало (< 14) или слишком велико (превышает максимальный размер PDU); такая ошибка является критической и указывается значением Bad PDU Length в поле Status Code; максимальный размер PDU определяется с соответствии с параграфом 3.5.3. Сообщение Initialization.

Сообщение LDP считается некорректным по форме, если:

- значение Message Type не известно:

  если значение Message Type < 0x8000 (старший бит равен 0), такая ошибка указывается значением Unknown Message Type в поле Status Code;

  сообщения с Message Type ≥ 0x8000 (старший бит равен 1) отбрасываются без уведомления;
- значение Message Length слишком велико (т. е. указывает, что сообщение выходит за рамки содержащего его LDP PDU; такая ошибка является критической и указывается значением Bad Message Length в поле Status Code;
- значение Message Length слишком мало (т. е. меньше минимального возможного значения компоненты); такая ошибка является критической и указывается значением Bad Message Length в поле Status Code;
- в сообщении отсутствует один или несколько обязательных параметров (Mandatory Parameter); такая ошибка не является критической и указывается значением Missing Message Parameters в поле Status Code.

3.5.1.2.2. Неизвестные или некорректные по форме TLV

Некорректные по форме TLV в сообщениях LDP, являющихся частью механизма LDP Discovery, обрабатываются путём отбрасывания содержащего такие TLV без уведомления отправителя.

TLV, содержащиеся в сообщении LDP, полученном через соединение TCP для сессии LDP, считаются некорректными по форме, если:

- значение TLV Length слишком велико (т. е., указывает, что TLV выходит за рамки сообщения); такая ошибка считается критической и указывается значением Bad TLV Length в поле Status Code;
- тип TLV не известен:

  если поле типа TLV имеет значение < 0x8000 (старший бит равен 0), такая ошибка указывается значением Unknown TLV в поле Status Code;

  если значение типа TLV ≥ 0x8000 (старший бит равен h1), TLV отбрасывается без уведомления;

- поле TLV Value некорректно по форме; это относится к случаям, когда принимающая сторона обрабатывает TLV, но не может декодировать поле TLV Value; причиной таких ситуаций могут служить программные ошибки[1] на приёмном или передающем LSR; такие ошибки являются критическими и указываются значением Malformed TLV Value в поле Status Code.

3.5.1.2.3. Завершение отсчёта сеансового таймера KeepAlive

Это критическая ошибка, указываемая значением KeepAlive Timer Expired в поле Status Code.

3.5.1.2.4. Односторонний разрыв сессии

Это критическая ошибка, указываемая значением Shutdown в поле Status Code. Сообщение Notification может включать также Extended Status TLV с объяснением причин разрыва сессии to (Shutdown). Передающий маршрутизатор LSR разрывает сессию сразу же после передачи сообщения Notification.

3.5.1.2.5. События, связанные с сообщением Initialization

Согласование на этапе инициализации сессии (см. параграф 2.5.3. Инициализация сессии) может завершаться отказом, если полученные в сообщении Initialization параметры сессии не приемлемы. Это критическая ошибка. Конкретное значение поля Status Code зависит от параметра, оказавшегося неприемлемым, возможные значения определены в параграфе 3.5.3. Сообщение Initialization.

3.5.1.2.6. События, вызываемые другими сообщениями

Не только сообщения Initialization могут вызывать события, о которых требуется уведомлять партнёров LDP с помощью сообщений Notification. Такие события и значения Status Code в сообщениях Notification описаны в параграфах, где рассматриваются соответствующие сообщения.

3.5.1.2.7. Внутренние ошибки

Реализация LDP может обладать способностью детектировать специфичные для неё проблемные ситуации. Когда такие ситуации препятствуют корректному взаимодействию с партнёром, реализации следует (по возможности) использовать для оповещения партнёра значение Internal Error в поле Status Code. Ошибки этого типа относятся к критическим.

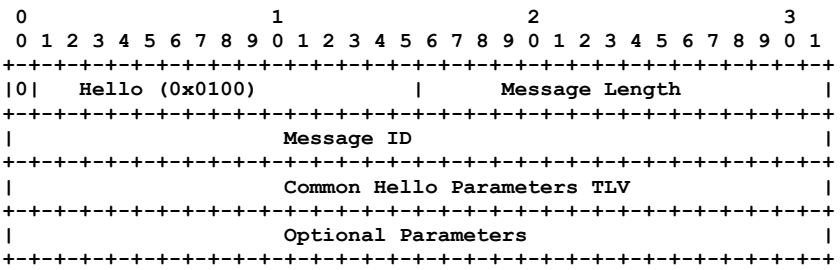
3.5.1.2.8. Прочие события

Ряд событий не относится ни к одной из рассмотренных выше категорий. В настоящей версии протокола такие события не определены.

3.5.2. Сообщение Hello

Сообщения LDP Hello передаются, как часть механизма LDP Discovery (см. параграф 2.4. LDP Discovery).

Формат сообщений Hello показан на рисунке.

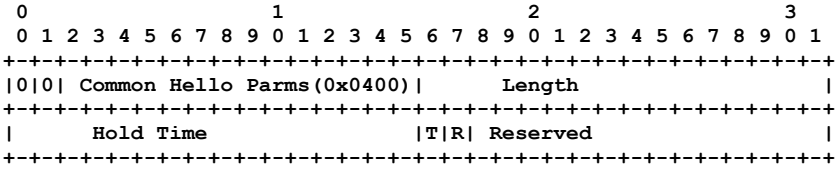


Message ID

32-битовое значение, служащее идентификатором данного сообщения.

Common Hello Parameters TLV

Падает параметры, являющиеся общими для всех сообщений Hello. Формат этого поля показан на рисунке.



Hold Time

Время удержания Hello в секундах. LSR поддерживает запись сообщений Hello, полученных от потенциальных партнёров (см. параграф 3.5.2.1. Процедуры для сообщений Hello). Значение Hold Time в сообщении Hello задаёт время, в течение которого передающий маршрутизатор LSR будет поддерживать свою запись о полученных сообщениях Hello от принимающего LSR без получения последующего Hello.

Пара LSR согласует время удержания для сообщений Hello друг от друга. Каждый из маршрутизаторов предлагает своё значение. Используется меньшее из двух значений, предложенных в соответствующих сообщениях Hello.

Нулевое значение указывает использование принятого по умолчанию времени удержания, которое составляет 15 секунд для сообщений Link Hello и 45 секунд - для Targeted Hello. Значение 0xffff задаёт бесконечное удержание.

T, направленное приветствие

Значение 1 говорит о том, что данное сообщение Hello является Targeted Hello. Нулевое значение поля указывает на Link Hello.

[1] В оригинале используется термин bug. *Прим. перев.*

R, запрос на передачу направленных приветствий

Значение 1 служит для запроса у получателя периодической отправки сообщений Targeted Hello отправителю данного сообщения Hello. Значение 0 показывает отсутствие такого запроса.

Маршрутизатор LSR, инициирующий механизм Extended Discovery, устанавливает R=1. Если R=1, принимающий сообщение LSR проверяет, настроен ли он на передачу сообщений Targeted Hello отправителю Hello в ответ на сообщения Hello с таким запросом. Если в конфигурации отправка сообщений не задана, запрос игнорируется. Если конфигурация включает отправку сообщений Targeted Hellos, данный запрос инициирует периодическую отправку соответствующих сообщений отправителю Hello.

Reserved

Это резервное поле. При передаче в нем **должно** устанавливаться нулевое значение, а на приёмной стороне значение поля **должно** игнорироваться.

Optional Parameters

Это поле переменного размера в сообщении Hello может содержать параметры, представляемые в формате TLV. Необязательные параметры, определённые для данной версии протокола, показаны в таблице.

Параметр	Тип	Размер	Значение
IPv4 Transport Address	0x0401	4	См. ниже
Configuration Sequence Number	0x0402	4	См. ниже
IPv6 Transport Address	0x0403	16	См. ниже

IPv4 Transport Address

Падает адрес IPv4, который будет использоваться для передающего маршрутизатора LSR при организации соединения TCP для сессии LDP. Если этот необязательный параметр отсутствует, **следует** использовать адрес IPv4 отправителя пакета UDP, содержащего сообщение Hello.

Configuration Sequence Number

Падает 4-октетный порядковый номер (целое число без знака), идентифицирующий конфигурационное состояние передающего LSR. Это значение используется принимающим LSR для детектирования смены конфигурации на стороне передающего LSR.

IPv6 Transport Address

Падает адрес IPv6, который будет использоваться для передающего маршрутизатора LSR при организации соединения TCP для сессии LDP. Если этот необязательный параметр отсутствует, **следует** использовать адрес IPv6 отправителя пакета UDP, содержащего сообщение Hello.

3.5.2.1. Процедуры для сообщений Hello

Маршрутизатор LSR, получающий сообщения Hello от другого LSR, организует отношения Hello-смежности, соответствующие сообщениям Hello. LSR поддерживает таймер удержания для Hello-смежности, который запускается заново всякий раз при получении сообщения Hello, соответствующего данной Hello-смежности. Если отсчет таймера для Hello-смежности завершается, LSR разрывает отношения Hello-смежности (см. параграфы 2.5.5. Поддержка Hello-смежности и 2.5.6. Поддержка сессий LDP).

Рекомендуется передавать сообщения Hello с интервалом не более одной трети значения времени удержания Hello.

LSR обрабатывает полученные сообщения LDP Hello следующим образом:

1. LSR проверяет возможность восприятия Hello. Критерии допустимости восприятия зависят от реализации (ниже приведён пример таких критериев).
2. Если сообщение Hello неприемлемо, LSR игнорирует его.
3. Если сообщение Hello приемлемо, LSR проверяет наличие Hello-смежности с отправителем этого сообщения. При наличии смежности перезапускается таймер удержания. Если смежность ещё не организована, она создаётся и запускается таймер удержания.
4. Если сообщение Hello содержит дополнительные TLV, LSR обрабатывает их (см. ниже).
5. В заключение, если LSR не имеет сессии LDP для пространства меток, заданного полем LDP Identifier в заголовке PDU для сообщения Hello, выполняются процедуры, описанные в параграфе 2.5.1. Организация сеанса LDP.

Ниже приведён пример критериев приемлемости для сообщений Link Hello и Targeted Hello.

Сообщение Link Hello считается приемлемым, если интерфейс, через который оно было получено, настроен на коммутацию по меткам.

Сообщение Targeted Hello от отправителя с адресом A считается приемлемым, если выполняется любое из условий:

- LSR настроен на восприятие сообщений Targeted Hello;
- LSR настроен на передачу сообщений Targeted Hello в адрес A.

Ниже описана обработка маршрутизатором LSR сообщений Hello с дополнительными TLV.

Transport Address

LSR связывает указанный транспортный адрес с Hello-смежностью.

Configuration Sequence Number

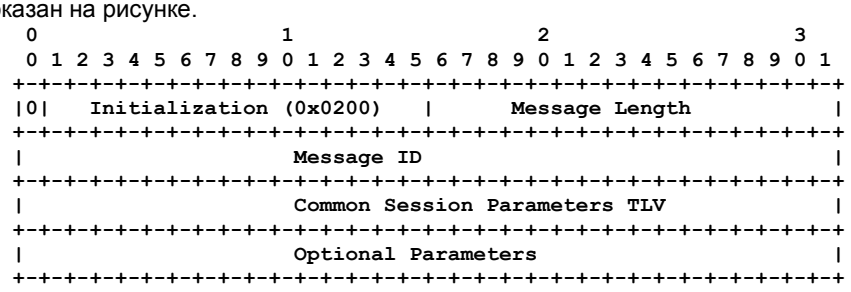
Необязательный параметр Configuration Sequence Number используется передающим маршрутизатором LSR для информирования принимающего LSR об изменении своей конфигурации. Когда принимающий LSR, играющий роль активного в процессе организации сеанса LDP, обнаруживает изменение конфигурации на стороне передающего LSR, он может сбросить увеличение периода повтора попыток соединения с передающим LSR, если таковое было начато (см. параграф 2.5.3. Инициализация сессии).

Передающий маршрутизатор LSR, использующий этот необязательный параметр, отвечает за корректность нумерации конфигурационных изменений, которая используется в сообщениях Hello. При каждом изменении конфигурации передающий маршрутизатор LSR должен увеличивать порядковый номер.

3.5.3. Сообщение Initialization

Сообщения LDP Initialization используются на этапе организации сеанса LDP (см. параграф 2.5.1. Организация сеанса LDP).

Формат сообщения показан на рисунке.



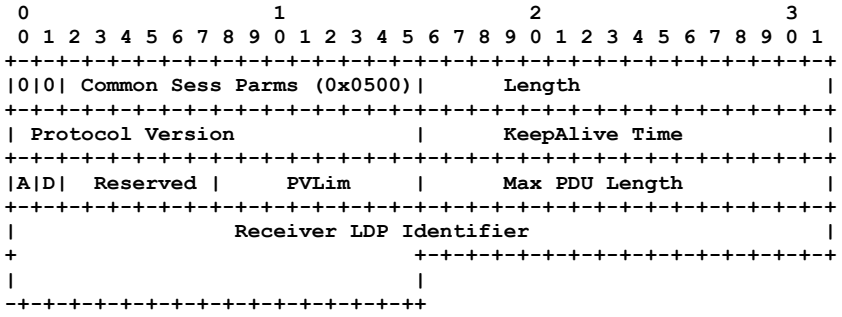
Message ID

32-битовое значение, служащее для идентификации данного сообщения.

Common Session Parameters TLV

Указывает значения, предложенные передающим LSR для параметров, которые должны согласовываться в каждой сессии LDP.

Представление общих параметров сессии (Common Session Parameters TLV) показано на рисунке.



Protocol Version

2-октетное целое число без знака, указывающее номер версии протокола. Данная спецификация определяет протокол LDP версии 1.

KeepAlive Time

2-октетное целое число без знака, которое показывает предложенное передающим LSR значение KeepAlive Time (в секундах). Принимающий маршрутизатор LSR **должен** рассчитать значение KeepAlive Timer, используя меньшее из двух значений (предложенное им и полученное в PDU) KeepAlive Time. Выбранное значение KeepAlive Time показывает максимальное число секунд, которое может разделять приём двух последовательных PDU от партнёра LDP в TCP-соединении протокольной сессии. Таймер KeepAlive сбрасывается при получении каждого PDU.

A, дисциплина анонсирования меток

Показывает тип анонса метки (Label advertisement). Значение 0 говорит о незапрошенном нисходящем анонсе (Downstream Unsolicited); значение 1 говорит о нисходящем анонсировании по запросу (Downstream On Demand). Если один LSR предлагает Downstream Unsolicited, а другой - Downstream on Demand, выбор осуществляется по следующим правилам:

- если сессия организована для управляемого по меткам соединения ATM или Frame Relay, **должен** использоваться режим Downstream on Demand;
- в остальных случаях **должен** использоваться режим Downstream Unsolicited.

  Если определённая описанным способом дисциплина анонсирования меток неприемлема для LSR, этот маршрутизатор **должен** отправить сообщение Session Rejected/Parameters Advertisement Mode Notification в ответ на сообщение Initialization и отказаться от организации сессии.

D, детектирование петель

Показывает возможность детектирования петель (Loop Detection) на базе векторов пути (Path Vector). Значение 0 говорит о том, что механизм Loop Detection отключён, а значение 1 означает возможность детектирования петель.

PVLim, максимальный размер вектора пути

Заданный в конфигурации максимальный размер Path Vector. Если детектирование петель запрещено (D = 0), это поле **должно** иметь значение 0. Если процедуры Loop Detection будут требовать от LSR передачи Path Vector с превышающим заданный предел размером, LSR будет вести себя, как при обнаружении петли для соответствующего FEC.

Если для части сети разрешён механизм Loop Detection, для всех LSR в этой части сети рекомендуется устанавливать одинаковое значение Path Vector Limit. Несмотря на то, что знание партнерского значения Path Vector Limit не будет менять поведения LSR, это значений позволит LSR предупредить оператора о возможной некорректности конфигурации.

Reserved

Это поле является резервным. На передающей стороне значения поля **должно** устанавливаться в 0, а на приёмной - игнорироваться.

Max PDU Length

2-октетное целое число без знака, предлагающее максимально допустимый размер LDP PDU для данной сессии. Значения, не превышающие 255, говорят об использовании принятого по умолчанию максимума - 4096 октетов.

На принимающей стороне LSR **должен** определять максимальный размер PDU для сессии, как меньшее из двух (своего и партнерского) значений Max PDU Length. До завершения инициализации сессии используется принятый по умолчанию максимальный размер PDU.

Если определённый, как указано выше, максимальный размер PDU не приемлем для LSR, тот **должен** передать сообщение Session Rejected/Parameters Max PDU Length Notification в ответ на сообщение Initialization и отказаться от организации сеанса.

Receiver LDP Identifier

Идентифицирует пространство меток получателя. Этот идентификатор LDP вместе с идентификатором LDP на стороне отправителя (в заголовке PDU) позволяет получателю найти соответствие между сообщением Initialization и одной из Hello-смежностей (см. параграф 3.5.2.1. Процедуры для сообщений Hello).

Если соответствия с Hello-смежностью не найдено, LSR **должен** отправить сообщение Session Rejected/No Hello Notification в ответ на сообщение Initialization и отказаться от организации сеанса.

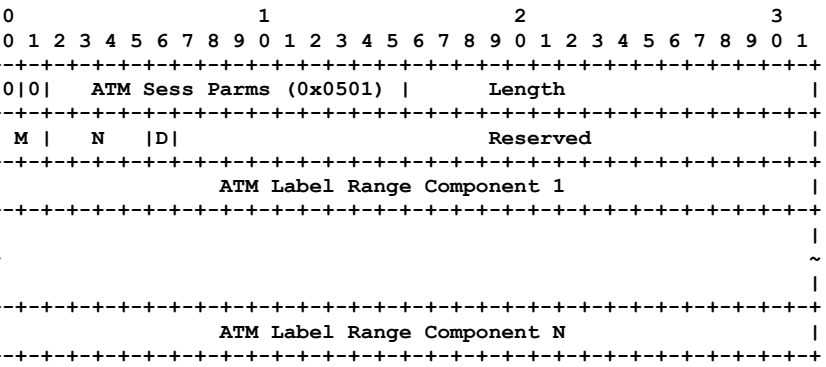
Optional Parameters

Это поле переменного размера может содержать параметры, представляемые в форме TLV. Возможные параметры указаны в таблице.

Параметр	Тип	Размер	Значение
ATM Session Parameters	0x0501	Перем.	См. ниже
Frame Relay Session Parameters	0x0502	Перем.	См. ниже

Параметры сессии ATM

Используются в тех случаях, когда сессия LDP служит для управления обменом метками на канале ATM для задания специфичных для ATM параметров сессии. Формат параметра показан на рисунке.



M, возможности слияния меток в ATM

Показывает возможности слияния, поддерживаемые коммутатором ATM. Определённые в настоящей версии спецификации значения параметра показаны в таблице.

Значение	*Смысл*
0	Слияние не поддерживается
1	Поддерживается слияние VP
2	Поддерживается слияние VC
3	Поддерживается слияние VP и VC

Если параметры слияния маршрутизаторов LSR различаются:

- LSR, не поддерживающий слияния, может свободно обмениваться метками с LSR, поддерживающими слияние VC.
- Взаимодействие коммутаторов, поддерживающих и не поддерживающих слияние VP, требует дополнительного изучения. Если маршрутизаторы LSR не относятся к поддерживающим слияние VP, сессия организуется, но слияние VP не используется.

Отметим, что при использовании слияния VP входной узел принимает на себя ответственность за обеспечение уникальности выбранного значения VCI в масштабе домена LSR.

N, число компонент диапазона меток

Задаёт число ATM Label Range Component, включённых в TLV.

D, направление VC

Нулевое значение флага говорит о двухстороннем VC, позволяющем LSR (в рамках данного VPI) поддерживать использование значения VCI в качестве метки для обоих направлений независимо. Значение 1 указывает на односторонний VC, когда данное значение VCI (в рамках данного VPI) может применяться для отображения меток только в одном направлении. Когда один или оба партнёра указывают однонаправленный характер VC, оба LSR используют одностороннее выделение меток VC для канала. Маршрутизаторы LSR сравнивают свои значения LDP Identifier, как целые числа без знака. LSR с большим значением LDP Identifier может выделять только нечётные значения VCI в своём диапазоне VPI/VCI для использования в качестве меток. Система с меньшим значением LDP Identifier может выделять только чётные значения VCI в своём диапазоне VPI/VCI для использования в качестве меток.

Reserved

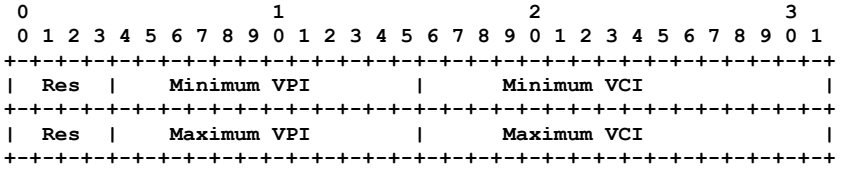
Это поле является резервным. На передающей стороне значения поля **должно** устанавливаться в 0, а на приёмной - игнорироваться.

Одно или несколько полей ATM Label Range Component

Список ATM Label Range Component, которые совместно задают диапазон меток (Label range) поддерживаемых передающим LSR.

Принимающий маршрутизатор LSR **должен** определять пересечение между полученным диапазоном меток и своим диапазоном. Пересечение представляет собой диапазон меток, которые LSR может выделять и воспринимать. Для маршрутизаторов LSR **недопустима** организация сессий с соседями, для которых пересечение диапазонов меток является пустым (NULL). В таких случаях LSR **должен** передать сообщение Session Rejected/Parameters Label Range Notification в ответ на сообщение Initialization и продолжать организацию сеанса.

Представление компонент диапазона меток ATM показано на рисунке.



Res

Это поле является резервным. На передающей стороне значения поля должно устанавливаться в 0, а на приёмной - игнорироваться.

Minimum VPI (12 битов)

Это 12-битовое поле задаёт нижнюю границу диапазона идентификаторов виртуальных путей (VPI), который поддерживается на коммутатором. Если размер VPI меньше 12 битов, его **следует** дополнить недостающим количеством нулей слева.

Minimum VCI (16 битов)

Это 16-битовое поле задаёт нижнюю границу диапазона идентификаторов виртуальных соединений (VCI), который поддерживается на коммутатором. Если размер VCI меньше 16 битов, его **следует** дополнить недостающим количеством нулей слева.

Maximum VPI (12 битов)

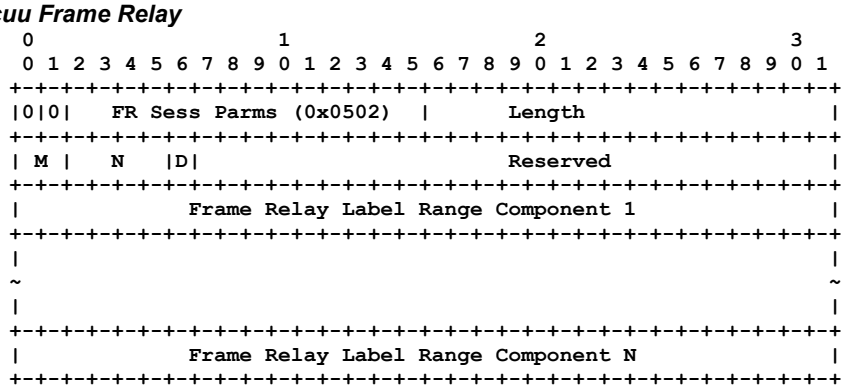
Это 12-битовое поле задаёт верхнюю границу диапазона идентификаторов виртуальных путей (VPI), который поддерживается на коммутатором. Если размер VPI меньше 12 битов, его **следует** дополнить недостающим количеством нулей слева.

Maximum VCI (16 битов)

Это 16-битовое поле задаёт верхнюю границу диапазона идентификаторов виртуальных соединений (VCI), который поддерживается на коммутатором. Если размер VCI меньше 16 битов, его **следует** дополнить недостающим количеством нулей слева.

Когда LSR соединяются не напрямую с помощью ATM VP, передающему маршрутизатору LSR **следует** установить для полей Minimum VPI и Maximum VPI значение 0, а принимающий LSR **должен** игнорировать значения полей Minimum VPI и Maximum VPI.

Параметры сессии Frame Relay



Используются в тех случаях, когда сессия LDP служит для управления обменом метками на канале Frame Relay для задания специфичных для Frame Relay параметров сессии. Формат параметра показан на рисунке.

M, возможности слияния Frame Relay

Показывает возможности слияния в коммутаторе Frame Relay. Определённые настоящей спецификацией значения показаны в таблице.

Значение	***Смысл***
0	Слияние не поддерживается
1	Слияние поддерживается

Маршрутизаторы Frame Relay LSR, поддерживающие и не поддерживающие слияния меток могут быть полностью интероперабельны.

N, число компонент диапазона меток

Задаёт число Frame Relay Label Range Component, включённых в TLV.

D, направление VC

Нулевое значение флага говорит о двухстороннем VC, позволяющем LSR поддерживать использование значения DLCI в качестве метки для обоих направлений независимо. Значение 1 указывает на односторонний VC, когда данное значение DLCI может применяться для отображения меток только в одном направлении. Когда один или оба партнёра указывают однонаправленный характер VC, оба LSR используют одностороннее выделение меток VC для канала. Маршрутизаторы LSR сравнивают свои значения LDP Identifier, как целые числа без знака. LSR с большим значением LDP Identifier может выделять только нечётные значения DLCI в своём диапазоне меток. Система с меньшим значением LDP Identifier может выделять только чётные значения DLCI в своём диапазоне меток.

Reserved

Это поле является резервным. На передающей стороне значения поля **должно** устанавливаться в 0, а на приёмной - игнорироваться.

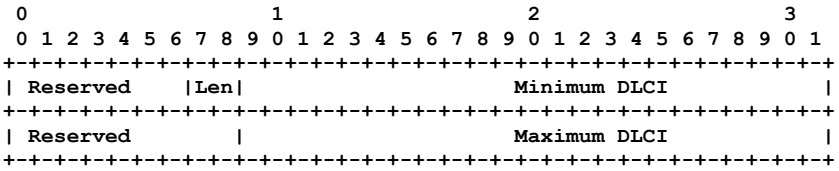
Одно или несколько полей Frame Relay Label Range

Список значений Frame Relay Label Range Component, которые совместно задают диапазон меток, поддерживаемый передающим LSR.

Принимающий маршрутизатор LSR **должен** определять пересечение между полученным диапазоном меток и своим диапазоном. Пересечение представляет собой диапазон меток, которые LSR может выделять и воспринимать. Для маршрутизаторов LSR **недопустима** организация сессий с соседями, для которых пересечение диапазонов меток является пустым (NULL). В таких случаях LSR **должен** передать сообщение

Session Rejected/Parameters Label Range Notification в ответ на сообщение Initialization и продолжать организацию сеанса.

Формат представления Frame Relay Label Range Component показан на рисунке.



Reserved

Это поле является резервным. На передающей стороне значения поля **должно** устанавливаться в 0, а на приёмной - игнорироваться.

Len

Это поле задаёт размер DLCI в битах. Поддерживаемые значения показаны в таблице.

Len	*Размер DLCI*
0	10
2	23

Значения 1 и 3 зарезервированы.

Minimum DLCI

Это 23-битовое поле задаёт нижнюю границу блока идентификаторов канала данных (DLCI) который поддерживается коммутатором. Значение DLCI **следует** выравнивать по правому краю, дополняя при необходимости нулями слева.

Maximum DLCI

Это 23-битовое поле задаёт верхнюю границу блока идентификаторов канала данных (DLCI) который поддерживается коммутатором. Значение DLCI **следует** выравнивать по правому краю, дополняя при необходимости нулями слева.

Отметим, что Generic Session Parameters TLV не определяются для сессий, анонсирующих метки общего назначения (Generic Label).

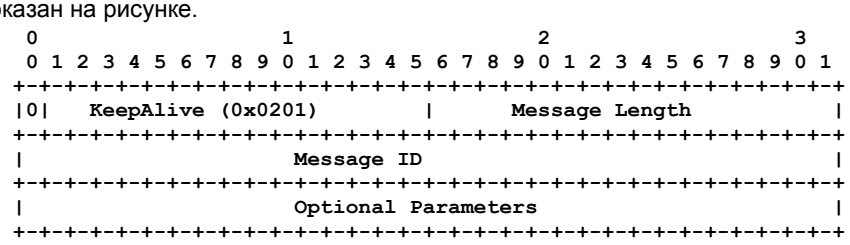
3.5.3.1. Процедуры для сообщений Initialization

См. параграфы 2.5.1. Организация сеанса LDP и 2.5.4. Инициализация машины состояний, где приведено общее описание процедур обработки сообщений Initialization.

3.5.4. Сообщение KeepAlive

LSR передаёт сообщения KeepAlive для мониторинга целостности транспортного соединения сессии LDP.

Формат сообщения показан на рисунке.



Message ID

32-битовое значение, используемое для идентификации сообщений.

Optional Parameters

Для сообщений KeepAlive дополнительные параметры не определены.

3.5.4.1. Процедуры для сообщений KeepAlive

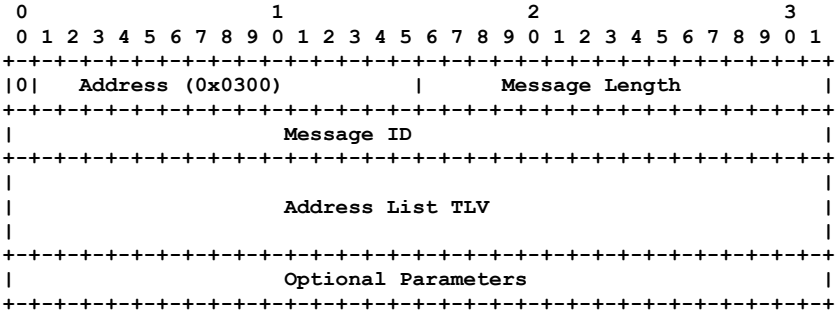
Механизм KeepAlive Timer, описанный в параграфе 2.5.6. Поддержка сессий LDP, сбрасывает сеансовый таймер KeepAlive при получении каждого LDP PDU в соединении TCP для данной сессии. Сообщения KeepAlive позволяют обеспечить сброс таймера KeepAlive в тех случаях, когда у LSR нет информации для передачи партнёру LDP.

LSR **должен** обеспечить своему партнёру получение сообщения LDP не реже, чем период KeepAlive Time. Принимаются во внимание все сообщения протокола LDP, но в тех случаях, когда в течение указанного периода не передавалось никаких сообщений LDP, маршрутизатор **должен** передать сообщение KeepAlive.

3.5.5. Сообщение Address

LSR передаёт сообщения Address своему партнёру LDP для анонсирования адресов своих интерфейсов.

Формат сообщения показан на рисунке.



Message ID

32-битовое значение, используемое для идентификации сообщений.

Address List TLV

Список адресов интерфейсов, которые будут анонсироваться передающим маршрутизатором LSR. Формат представления Address List TLV описан в параграфе 3.4.3. TLV для списка адресов.

Optional Parameters

Для сообщений Address дополнительные параметры не определены.

3.5.5.1. Процедуры для сообщений Address

Маршрутизатор LSR, получивший сообщение Address, использует адреса из этого сообщения для своей базы данных по отображениям между идентификаторами LDP и адресами следующих маршрутизаторов (см. параграф 2.7. Идентификаторы LDP и адреса Next Hop).

При организации нового сеанса LDP до передачи сообщений Label Mapping или Label Request маршрутизатору LSR **следует** анонсировать адреса своих интерфейсов с помощью одного или нескольких сообщений Address.

Когда LSR «активизирует» новый адрес интерфейса, ему **следует** анонсировать этот адрес в сообщении Address.

Когда LSR «деактивирует» анонсированный ранее интерфейс, ему **следует** отозвать этот адрес с помощью сообщения Address Withdraw (см. параграф 3.5.6. Сообщение Address Withdraw).

Если LSR не поддерживает семейство адресов, указанное в Address List TLV, ему **следует** передать сообщение Unsupported Address Family Notification для сигнализации об ошибке и прервать дальнейшую обработку сообщения.

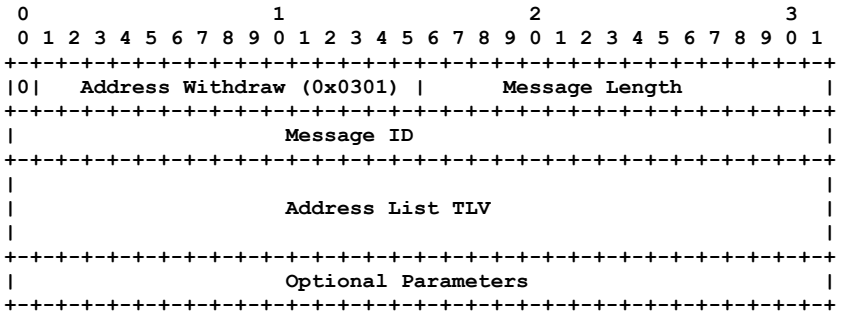
LSR может повторно анонсировать адрес (A) без явного отзыва этого адреса. Если получатель уже имеет привязку для этого адреса (LSR, A), ему не **следует** предпринимать никаких дополнительных действий.

LSR может отозвать адрес (A), который не был анонсирован ранее. Если получатель видит отзыв адреса, для которого у него нет привязки (LSR, A), ему не **следует** предпринимать никаких дополнительных действий.

3.5.6. Сообщение Address Withdraw

LSR передаёт сообщение Address Withdraw своему партнёру LDP для отзыва анонсированных ранее адресов своих интерфейсов.

Формат сообщения Address Withdraw показан на рисунке.



Message ID

32-битовое значение, используемое для идентификации сообщений.

Address List TLV

Список адресов интерфейсов маршрутизатора, которые отзываются передающим сообщение LSR. Представление Address List TLV описано в параграфе 3.4.3. TLV для списка адресов.

Optional Parameters

Для сообщений Address Withdraw дополнительные параметры не определены.

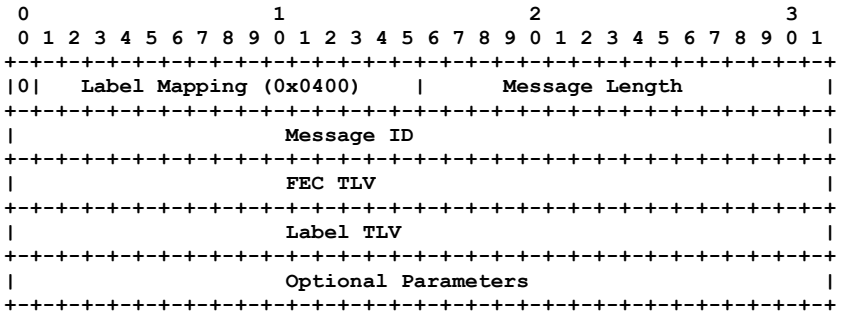
3.5.6.1. Процедуры для сообщений Address Withdraw

См. параграф 3.5.5.1. Процедуры для сообщений Address.

3.5.7. Сообщение Label Mapping

LSR передаёт сообщения Label Mapping своему партнёру LDP для анонсирования ему привязки FEC к меткам.

Формат сообщения Label Mapping показан на рисунке.



Message ID

32-битовое значение, используемое для идентификации сообщений.

FEC TLV

Задаёт FEC-компоненту анонсируемой связки FEC-Label. Представление FEC TLV описано в параграфе 3.4.1. FEC TLV.

Label TLV

Задаёт Label-компоненту анонсируемой связки FEC-Label. Представление Label TLV описано в параграфе 3.4.2. TLV для меток.

Optional Parameters

Это поле переменного размера может содержать параметры в формате TLV. Набор поддерживаемых параметров показан в таблице.

Параметр	*Размер*	*Значение*
Label Request Message ID TLV	4	См. ниже
Hop Count TLV	1	См. ниже
Path Vector TLV	перем.	См. ниже

Представление Hop Count TLV и Path Vector TLV описано в параграфе 3.4. Представление TLV для параметров общего назначения.

Label Request Message ID

Если сообщение Label Mapping является откликом на сообщение Label Request, оно **должно** включать дополнительный параметр Label Request Message ID. Значением этого параметра является поле Message ID из соответствующего сообщения Label Request.

Hop Count

Задаёт общее число интервалов LSR на пути LSP, организуемом с помощью сообщения Label. Обработка этого поля описана в параграфе 3.4.4.1. Процедуры подсчёта интервалов.

Path Vector

Задаёт маршрутизаторы LSR в пути LSP, организуемом с помощью этого сообщения Label. Обработка этого поля описана в параграфе 3.4.5.1. Процедуры Path Vector.

3.5.7.1. Процедуры для сообщений Label Mapping

Сообщения Mapping используются маршрутизаторами LSR для распространения информации об отображении меток на FEC своим партнёрам LDP. Если LSR распространяет отображение для FEC множеству партнёров LDP, он может сам выбрать распространение отображения метки на FEC всем своим партнёрам или использование разных отображений для каждого из своих партнёров.

Маршрутизатор LSR отвечает за корректность распространяемого отображения меток и контроль наличия этих отображений у его партнёров.

Маршрутизатору LSR, получившему сообщение Label Mapping от нисходящего LSR для некого префикса (Prefix), **не следует** использовать метку для пересылки, пока в его таблице маршрутизации нет записи, которая точно соответствует элементу FEC (FEC Element).

Более подробная информация приведена ниже (Приложение A. Процедуры распространения меток LDP).

3.5.7.1.1. Независимое управление отображением

Если LSR настроен на независимое управление, сообщение об отображении меток передаётся этим LSR при возникновении любого из перечисленных ниже условий:

1. LSR распознает новый класс FEC через таблицу пересылки и метки анонсируются в режиме Downstream Unsolicited;
2. LSR получает сообщение Request от восходящего партнёра для FEC, имеющегося в таблице пересылки LSR;
3. следующий интервал для FEC меняется на другого партнёра LDP и включено детектирование петель;
4. изменяются атрибуты отображения;
5. принимается отображение от следующего интервала в нисходящем направлении **И** выполняется любое из условий:
   a) восходящее отображение не было создано;
   b) включено детектирование петель;
   c) атрибуты отображения изменились.

3.5.7.1.2. Упорядоченное управление отображением

Если LSR работает в режиме упорядоченного управления (Ordered Control), сообщение Mapping передаётся нисходящими LSR при выполнении любого из перечисленных ниже условий:

1. LSR распознает новый класс FEC через таблицу пересылки и является выходным для данного FEC;
2. LSR получает сообщение Request от восходящего партнёра для FEC, имеющегося в таблице пересылки LSR и этот LSR является выходным для данного FEC **или** имеет нисходящее отображение для данного FEC$
3. следующий интервал для FEC меняется на другого партнёра LDP и включено детектирование петель;
4. изменяются атрибуты отображения;
5. принимается отображение от следующего интервала в нисходящем направлении **И** выполняется любое из условий:
   a) восходящее отображение не было создано;
   b) включено детектирование петель;
   c) атрибуты отображения изменились.

3.5.7.1.3. Нисходящее анонсирование меток по запросам

В общем случае восходящий маршрутизатор LSR отвечает за запрашивание отображений меток при работе в режиме Downstream on Demand. Однако, если не соблюдать некоторые правила, могут возникать ситуации, когда два соседних LSR с разными режимами анонсирования могут вызывать «блокировку», при которой все работает корректно, но распространения меток не происходит. В качестве примера рассмотрим два LSR (Ru и Rd), где Ru является

восходящим LSR, Rd - нисходящим LSR для некого FEC. Пусть Ru работает в режиме Downstream Unsolicited, Rd - в режиме Downstream on Demand. В этом случае Rd может предполагать, Ru будет запрашивать отображение меток, когда оно потребуется, а Ru может предполагать, что Rd будет анонсировать метку, если захочет, чтобы Ru её использовал. В такой ситуации метки от Rd к Ru просто не будут распространяться.

Описанной блокировки можно избежать, если выполнять приведённое здесь правило - маршрутизатору LSR, работающему в режиме Downstream on Demand, **не следует** полагаться на передачу анонсов отображений без запроса. Следовательно, если нисходящий LSR работает в режиме Downstream on Demand, восходящий LSR является ответственным за запрашивание требуемых меток.

3.5.7.1.4. Нисходящее анонсирование меток без запроса

В общем случае нисходящий LSR отвечает за анонсирование отображений меток, которые он хочет передать для использования восходящему LSR. Восходящий LSR может вводить запросы для отображений по своему усмотрению.

Комбинация режима анонсирования без запроса (Downstream Unsolicited) с консервативным удержанием меток (Conservative Label retention) может приводить к возникновению ситуаций, когда LSR будет освобождать метку для FEC, который позднее потребуется. Например, если LSR Rd анонсирует маршрутизатору LSR Ru метку для FEC, по отношению к которому не является следующим интервалом для Ru, маршрутизатор Ru будет освобождать эту метку. Если на Ru следующий интервал для FEC будет в последствии изменён на Rd, возникнет потребность в освобождённой ранее метке.

Для предотвращения проблем в таких ситуациях Ru может явно запросить метку при возникновении необходимости в ней или Rd может периодически повторять анонсирование метки маршрутизатору Ru. Во многих случаях Ru будет знать, когда ему нужна метка от Rd. Например, при смене следующего интервала для FEC на Rd. Однако могут возникать ситуации, когда Ru не знает о такой необходимости. Например, Rd может попытаться организовать LSP с нестандартными свойствами. Принуждение Ru к явному запрашиванию метки в такой ситуации потребует поддержки информации о состоянии для потенциальных LSP с нестандартными свойствами.

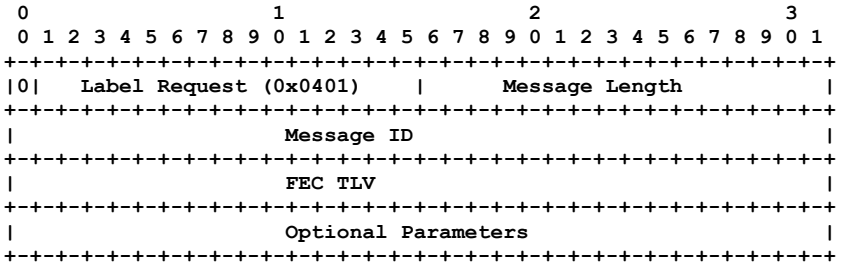
В тех случаях, когда Ru знает о потребности в метке, он несёт ответственность за явный запрос метки с помощью сообщения Label Request. В ситуациях, когда Ru может не знать о потребности в метке, Rd отвечает за периодическое реанонсирование метки маршрутизатору Ru.

Для этой версии протокола LDP единственная ситуация, когда Ru знает о потребности в метке для FEC от Rd, возникает в случае, где Rd является следующим интервалом для FEC, Ru не имеет метки от Rd, а LSP для FEC является одним из тех, которые могут быть организованы с использованием TLV, определённых в этом документе.

3.5.8. Сообщение Label Request

LSR передаёт сообщения Label Request партнёру LDP для запроса у того информации о привязке (отображении) метки для FEC.

Формат сообщения Label Request показан на рисунке.



Message ID

32-битовое значение, используемое для идентификации сообщений.

FEC TLV

Класс FEC, для которого запрашивается метка. Представление этого поля описано в параграфе 3.4.1. FEC TLV.

Optional Parameters

Это поле переменной длины может содержать параметры, представленные в форме TLV. Список дополнительных параметров приведён в таблице.

Параметр	*Размер*	*Значение*
Hop Count TLV	1	См. ниже
Path Vector TLV	перем.	См. ниже

Представление Hop Count TLV и Path Vector TLV описано в параграфе 3.4. Представление TLV для параметров общего назначения.

Hop Count

Задаёт число интервалов LSR на пути LSP, который будет организован с помощью сообщения Label Request. Обработка этого TLV описана в параграфе 3.4.4.1. Процедуры подсчёта интервалов.

Path Vector

Задаёт маршрутизаторы LSR, через которые будет организован путь LSP[1] с помощью сообщения Label Request. Обработка этого TLV описана в параграфе 3.4.5.1. Процедуры Path Vector.

3.5.8.1. Процедуры для сообщений Label Request

Сообщения Label Request используются восходящим LSR для явного запрашивания от нисходящего LSR выделения и анонсирования метки для FEC.

LSR может передать запрос при выполнении любого из перечисленных ниже условий:

1. LSR распознает новый класс FEC через таблицу пересылки, следующим интервалом является партнёр LDP и данный LSR ещё не имеет отображения от следующего интервала для данного FEC;

[1]В оригинале ошибочно указано LSR. *Прим. перев.*

2. следующий интервал для FEC меняется, а данный LSR ещё не имеет отображения от следующего интервала для данного FEC[1];

3. LSR получает сообщение Label Request для FEC от своего восходящего партнёра LDP, следующим интервалом для FEC является партнёр LDP, а данный LSR ещё не имеет отображения от следующего интервала[2].

Принимающему LSR **следует** отвечать на сообщение Label Request сообщением Label Mapping для запрошенной метки или сообщением Notification, говорящим о невозможности выполнения запроса.

Когда FEC, для которого запрашивается метка, является Prefix FEC Element, принимающий LSR использует свою таблицу маршрутизации для определения отклика. Если его таблица маршрутизации не содержит записи, точно соответствующей запрошенному префиксу, LSR **должен** отвечать сообщением No Route Notification.

Поле Message ID в сообщениях Label Request служит идентификатором транзакции для запроса метки. Когда принявший сообщение LSR отвечает сообщением Label Mapping, последнее **должно** включать дополнительный параметр Label Request/Returned Message ID TLV, который содержит значение Message ID из сообщения Label Request. Отметим, что по причине использования маршрутизаторами LSR поля Message ID из сообщения Label Request в качестве идентификатора транзакции, LSR **не следует** повторно использовать значение идентификатора, указанное в сообщении Label Request, до завершения транзакции.

Данная версия протокола определяет перечисленные ниже коды (Status Code) для сообщений Notification, говорящих о невозможности выполнения запроса.

No Route
FEC, для которого была запрошена метка, включает FEC Element, для которого LSR не имеет маршрута.

No Label Resources
LSR не может обеспечить метку по причине нехватки ресурсов. Когда ресурсы станут доступными, LSR **должен** уведомить запрашивавший метку маршрутизатор LSR с помощью сообщения Notification, содержащего код состояния Label Resources Available.
Маршрутизатору LSR, получившему отклик No Label Resources в ответ на сообщение Label Request, **недопустимо** вводить новые запросы Label Request, пока он не получит сообщение Notification с кодом состояния Label Resources Available.

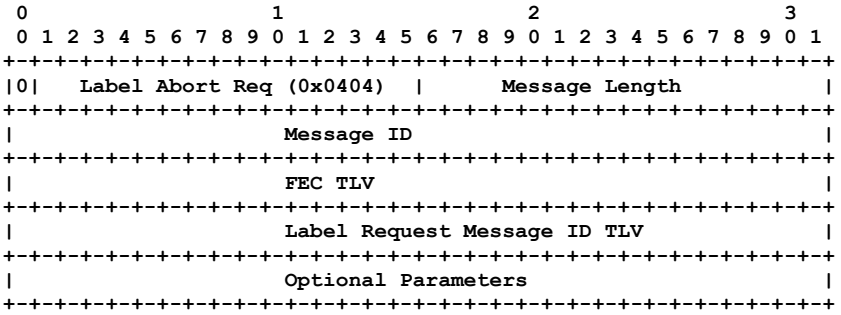
Loop Detected
LSR обнаружил «зацикливание» сообщения Label Request (петлю).

Приложение A. Процедуры распространения меток LDP содержит дополнительную информацию о распространении меток.

3.5.9. Сообщение Label Abort Request

Сообщения Label Abort Request могут служить для прерывания обработки сообщений Label Request.

Формат сообщения Label Abort Request показан на рисунке.



Message ID
32-битовое значение, используемое для идентификации сообщений.

FEC TLV
Указывает FEC, для которого прерывается обработка Label Request.

Label Request Message ID TLV
Задаёт идентификатор сообщения Label Request, для которого прерывается обработка.

Optional Parameters
Для сообщений Label Abort Request дополнительные параметры не определены.

3.5.9.1. Процедуры для сообщений Label Abort Request

LSR Ru может отправить сообщение Label Abort Request для прерывания обработки отправленного ранее маршрутизатору LSR Rd сообщения Label Request для FEC при следующих обстоятельствах:

1. следующий интервал на Ru для класса FEC был изменён с LSR Rd на LSR X;

2. Ru не является входным LSR, не поддерживает слияние и также получил сообщение Label Abort Request для FEC от восходящего партнёра Y;

3. Ru не является входным LSR, поддерживает слияние и получил сообщение Label Abort Request для FEC от восходящего партнёра Y, который был единственным (последним) восходящим LSR, запросившим метку для FEC.

[1]Отметим, что если LSR уже имеет ожидающее отклика сообщение для нового следующего интервала, ему **не следует** отправлять дополнительное сообщение Label Request в ответ на смену следующего интервала.
[2]Отметим, что в результате того, что LSR без поддержки слияния должен организовывать отдельный LSP для каждого восходящего партнёра, запрашивающего метку, он должен передавать отдельное сообщение Label Request для каждого такого партнёра. Следствием этого является то, что не поддерживающий слияния LSR может иметь множество сообщений Label Request для данного FEC, ожидающих завершения обработки.

Могут возникать и другие ситуации, когда у LSR может возникнуть потребность в прерывании обработки запроса Label Request с целью возврата ресурсов, связанных с ожидающим обработки LSP. Однако рассмотрение общей стратегии использования механизм прерывания выходит за рамки спецификации протокола LDP.

Когда LSR получает сообщение Label Abort Request и у него имеется сообщение Label Request, в ответ на которое ещё не было отправлено сообщение Label Mapping или Notification, он **должен** подтвердить прерывание транзакции с помощью сообщения Label Request Aborted Notification. Сообщение Notification **должно** включать поле Label Request Message ID TLV, содержащее значение Message ID из прерываемого запроса Label Request.

Если LSR получает сообщение Label Abort Request Message после отправки сообщения Label Mapping или Notification в ответ на соответствующий запрос Label Request, он просто игнорирует запрос на прерывание.

Если LSR получает сообщение Label Mapping в ответ на сообщение Label Request после того, как он отправил запрос Label Abort Request для прерывания Label Request, метка, принятая в сообщении Label Mapping, остаётся корректной. LSR может по своему усмотрению принять решение об использовании метки или её освобождении с помощью сообщения Label Release.

LSR, прерывающий обработку сообщения Label Request не может повторно использовать значение Message ID для сообщений Label Request, пока он не получит от своего партнёра одно из сообщений:

- Label Request Aborted Notification с подтверждением прерывания обработки;
- Label Mapping в ответ на сообщение Label Request, для которого запрошено прерывание;
- Notification в ответ на сообщение Label Request, для которого запрошено прерывание (например, с кодом Loop Detected, No Label Resources и т. п.).

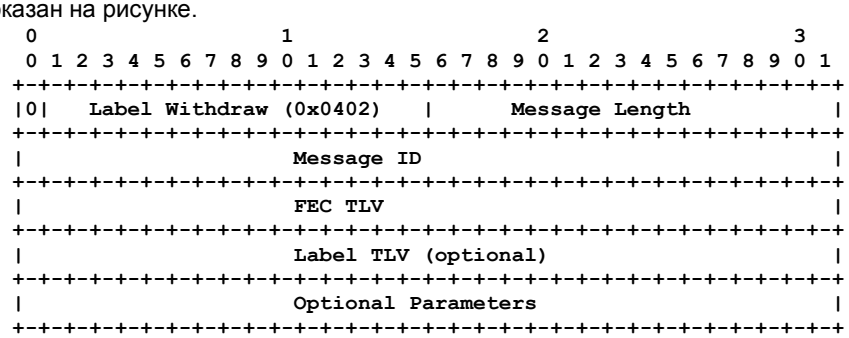
Для защиты от запоздало работающих партнёров или некорректных реализаций LSR может использовать тайм-аут повторного использования идентификаторов. Время ожидания следует делать достаточно большим (несколько минут). По истечении времени ожидания при отсутствии отклика от партнёра LSR может повторно использовать значение Message ID в сообщении Label Request; в таких случаях ему следует также отбрасывать все записи о незавершённых сообщениях Label Request и Label Abort.

Отметим, что отклики на сообщения Label Abort Request никогда не бывают «упорядоченными». Т. е. отклик не зависит от нисходящего состояния LSP, организация которого прерывается. Маршрутизатор LSR, получивший сообщение Label Abort Request, **должен** незамедлительно обработать его, независимо от состояния LSP в нисходящем направлении, отвечая сообщением Label Request Aborted Notification или игнорируя запрос на прерывание, если это допустимо.

3.5.10. Сообщение Label Withdraw

LSR передаёт сообщения Label Withdraw своему партнёру LDP для информирования того о том, что партнёр больше не может использовать указанное отображение FEC-label, которое данный LSR анонсировал ранее. Это используется для разрыва связок между FEC и метками.

Формат сообщения показан на рисунке.



Message ID

32-битовое значение, используемое для идентификации сообщений.

FEC TLV

Идентифицирует класс FEC, для которого отзывается отображение FEC-label.

Optional Parameters

Это поле переменного размера может содержать параметры в форме TLV. Список дополнительных параметров приведён в таблице.

Параметр	*Размер*	*Значение*
Label TLV	перем.	См. ниже

Формат Label TLV описан в параграфе 3.4.2. TLV для меток.

Label

При наличии этого параметра он указывает отзываемую метку (см. описание процедур ниже).

3.5.10.1. Процедуры для сообщений Label Withdraw

LSR передаёт сообщение Label Withdraw при выполнении следующих условий:

1. данный LSR больше не распознает ранее известный класс FEC, для которого он анонсировал метку;
2. LSR в одностороннем порядке (например, в соответствии с конфигурацией) принял решение больше не коммутировать метки для FEC с отзываемым отображением.

FEC TLV задаёт класс FEC, для которого метка отзывается. Если за указанием класса не следует Label TLV, отзываться будут все метки, связанные с этим FEC, в остальных случаях отзывается только метка, заданная Label TLV.

FEC TLV может содержать шаблон Wildcard FEC Element - в таких случаях других FEC Element присутствовать не может. В этом случае если сообщение Label Withdraw содержит необязательное поле Label TLV, метка будет

отзываться для всех FEC, соответствующих шаблону. Если поле Label TLV отсутствует в сообщении Label Withdraw, это означает, что передающий LSR отзывает все метки, анонсированные ранее принимающим LSR.

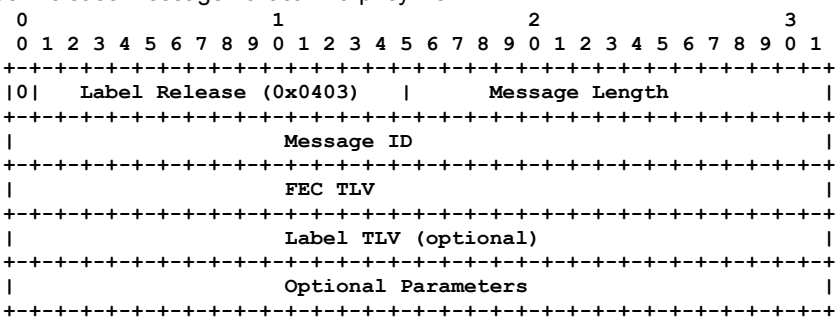
Маршрутизатор LSR, получивший сообщение Label Withdraw, **должен** ответить на него сообщением Label Release.

Приложение A. Процедуры распространения меток LDP содержит дополнительную информацию о процедурах отзыва меток.

3.5.11. Сообщение Label Release

LSR передаёт сообщения Label Release своим партнёрам LDP для передачи им информации о том, что данный LSR больше не нуждается в конкретном отображении FEC-label, которое ранее запрашивалось у партнёра и/или анонсировалось им.

Формат сообщения Label Release Message показан на рисунке.



Message ID

32-битовое значение, используемое для идентификации сообщений.

FEC TLV

Идентифицирует класс FEC, для которого освобождается отображение FEC-label.

Optional Parameters

Это поле переменного размера может содержать параметры в форме TLV. Список дополнительных параметров приведён в таблице.

Параметр	*Размер*	*Значение*
Label TLV	перем.	См. ниже

Представление Label TLV описано в параграфе 3.4.2. TLV для меток.

Label

При наличии этого поля оно указывает освобождаемую метку (см. описание процедур ниже).

3.5.11.1. Процедуры для сообщений Label Release

LSR передаёт сообщение Label Release своему партнёру в тех случаях, когда он больше не нуждается в метке полученной или запрошенной ранее у этого партнёра.

Маршрутизатор LSR **должен** передавать сообщение Label Release при выполнении любого из приведённых ниже условий:

1. маршрутизатор LSR, который передал отображение метки, больше не является следующим интервалом для отображаемого класса FEC, а LSR настроен на консервативное удержание;
2. маршрутизатор LSR получил отображение метки от LSR, который не является следующим интервалом для FEC, а LSR настроен на консервативное удержание;
3. маршрутизатор LSR получил сообщение Label Withdraw.

Отметим, что если LSR настроен на «либеральный режим», сообщение об освобождении меток не будет передаваться в указанных выше случаях (1) и (2). В таких случаях восходящий LSR сохраняет все неиспользуемые метки и может незамедлительно начать их последующее использование, если нисходящий партнёр станет следующим интервалом для FEC.

FEC TLV задаёт класс FEC, для которого освобождается метка. Если вслед за классом нет поля Label TLV, освобождаются все метки, связанные с указанным FEC, в остальных случаях освобождается только метка, заданная полем Label TLV.

FEC TLV может содержать шаблон Wildcard FEC Element; в этом случае других FEC Element не может быть включено. Если сообщение Label Release с шаблоном класса содержит дополнительное поле Label TLV, указанная им метка освобождается для всех FEC, соответствующих классу. Если параметр Label TLV не задан в сообщении Label Release, передающий LSR освобождает все отображения меток, полученные ранее от принимающего сообщение LSR.

Приложение A. Процедуры распространения меток LDP содержит дополнительную информацию о процедурах освобождения меток.

3.6. Сообщения и TLV для расширений

Поддержка расширений LDP включает правила для флагов U и F, которые указывают LSR способы обработки неизвестных TLV и сообщений.

В этом разделе описаны TLV и сообщения для фирменных (vendor-private) и экспериментальных расширений.

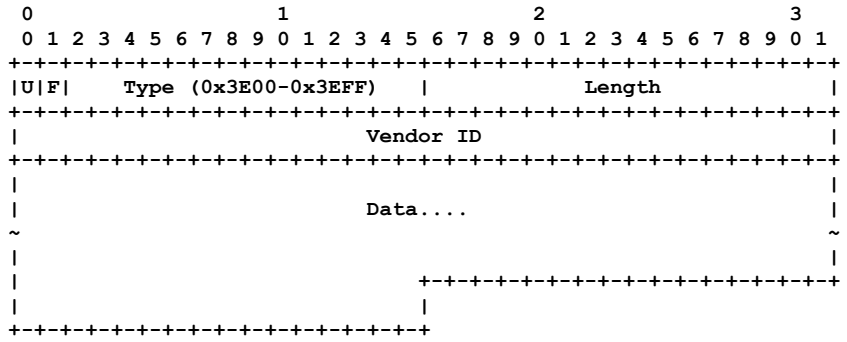
3.6.1. Расширения LDP Vendor-Private

Фирменные TLV и сообщения служат для передачи между LSR специфичной для производителя информации.

3.6.1.1. LDP Vendor-Private TLV

Диапазон значений поля Type от 0x3E00 до 0x3EFF зарезервирован для фирменных (vendor-private) TLV.

Формат сообщения показан на рисунке.



U

Флаг неизвестного TLV. При получении неизвестного TLV, если U=0, в ответ **должно** передаваться уведомление, а полученное сообщение **должно** игнорироваться. Если U=1, неизвестный параметр TLV игнорируется, а остальная часть сообщения обрабатывается, как обычно.

Трактовка сообщения vendor-private определяется полем Type и обязательным полем Vendor ID.

Реализации, поддерживающие фирменные TLV, **должны** поддерживать доступный для пользователя интерфейс настройки конфигурации, который позволяет задать принудительную установку флага U во всех передаваемых фирменных TLV; это требование **можно** выполнить за счёт доступного пользователю конфигурационного интерфейса, предотвращающего передачу фирменных TLV, в которых флаг U не установлен.

F

Флаг пересылки неизвестных TLV. Этот флаг имеет значение только при установленном бите U и пересылается сообщение, содержащее неизвестное поле TLV. Если F =0, неизвестное поле не пересылается в содержащем его сообщении, а при F=1 неизвестное поле TLV пересылается в сообщении.

Type

Значение типа из диапазона 0x3E00 - 0x3EFF. Поля Type и Vendor ID совместно определяют интерпретацию поля Data.

Length

Указывает суммарный размер полей Vendor ID и Data в октетах.

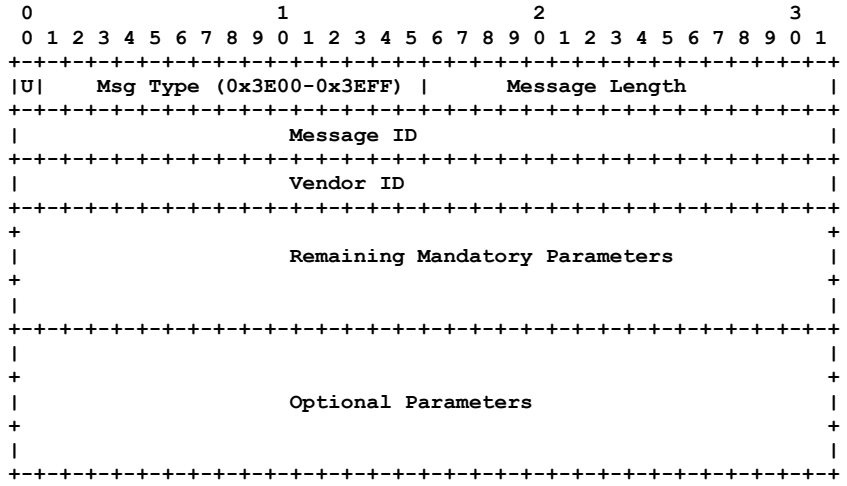
Vendor ID

Значение 802 для Vendor ID выделенное IEEE.

Data

Остальная часть поля Value после Vendor ID содержит дополнительные данные, трактовка которых определяется производителем.

3.6.1.2. Сообщение LDP Vendor-Private



Значение поля Type в диапазоне 0x3E00 - 0x3EFF зарезервированы для сообщений Vendor-Private.

U

Флаг неизвестного сообщения. При получении неизвестного сообщения с U=0 отправителю возвращается уведомление, а при U=1 неизвестное сообщение игнорируется без уведомления.

Трактовка сообщений Vendor-Private определяется значениями полей Msg Type и Vendor ID.

Реализации, поддерживающие сообщения Vendor-Private, **должны** поддерживать доступный пользователю интерфейс для управления принудительной установкой флага U в передаваемых сообщениях Vendor-Private. Это требование **может** быть выполнено за счёт доступного пользователю конфигурационного интерфейса, который позволяет предотвратить передачу сообщений Vendor-Private с U=0.

Msg Type

Значение Type в диапазоне 0x3E00 - 0x3EFF. Поля Msg Type и Vendor ID совместно определяют интерпретацию сообщения.

Message Length

Указывает суммарный размер полей Message ID, Vendor ID, Remaining Mandatory Parameters и Optional Parameters в октетах.

Message ID

32-битовое целое число, служащее для идентификации сообщения. Это поле применяется передающим LSR для сопоставления с сообщениями Notification, которые могут быть связаны с данным сообщением. Маршрутизатор LSR, передающий сообщение Notification в ответ на фирменное сообщение будет включать в уведомление значение поля Message ID (см. параграф 3.5.1. Сообщение Notification).

Vendor ID

Значение 802 для Vendor ID выделенное IEEE.

Remaining Mandatory Parameters

Набор обязательных параметров сообщения (переменный размер).

Optional Parameters

Набор дополнительных параметров сообщения (переменный размер).

3.6.2. Экспериментальные расширения LDP

Поддержка экспериментов в LDP похожа на поддержку фирменных расширений. Отличия перечислены ниже.

- для экспериментальных TLV зарезервированы значения поля Type в диапазоне от 0x3F00 до 0x3FFF;
- для экспериментов зарезервированы значения Message Type в диапазоне от 0x3F00 до 0x3FFF;
- формат экспериментальных TLV и сообщений похож на формат фирменных расширений с перечисленными ниже отличиями:

  экспериментальные TLV и сообщения используют поле Experiment ID взамен поля Vendor ID; поле Experiment ID используется с полем Type или Message Type, определяющим интерпретацию экспериментального TLV или сообщения.

  За управление значениями Experiment ID отвечает экспериментатор.

3.7. Список сообщений

Список сообщений LDP, определённых в данной версии протокола, приведён в таблице.

Сообщение	Тип	Описание
Notification	0x0001	223.5.1. Сообщение Notification
Hello	0x0100	3.5.2. Сообщение Hello
Initialization	0x0200	3.5.3. Сообщение Initialization
KeepAlive	0x0201	3.5.4. Сообщение KeepAlive
Address	0x0300	3.5.5. Сообщение Address
Address Withdraw	0x0301	3.5.6. Сообщение Address Withdraw
Label Mapping	0x0400	3.5.7. Сообщение Label Mapping
Label Request	0x0401	3.5.8. Сообщение Label Request
Label Withdraw	0x0402	3.5.10. Сообщение Label Withdraw
Label Release	0x0403	3.5.11. Сообщение Label Release
Label Abort Request	0x0404	3.5.9. Сообщение Label Abort Request
Vendor-Private	0x3E00-0x3EFF	3.6.1.2. Сообщение LDP Vendor-Private
Experimental	0x3F00-0x3FFF	3.6.2. Экспериментальные расширения LDP

3.8. Список TLV

Список TLV, определённых в данной версии протокола, приведён в таблице.

TLV	Тип	Описание
FEC	0x0100	3.4.1. FEC TLV
Address List	0x0101	3.4.3. TLV для списка адресов
Hop Count	0x0103	3.4.4. TLV для счётчика интервалов
Path Vector	0x0104	3.4.5. TLV для вектора пути
Generic Label	0x0200	3.4.2.1. TLV меток общего назначения
ATM Label	0x0201	3.4.2.2. TLV меток ATM
Frame Relay Label	0x0202	3.4.2.3. TLV для меток Frame Relay
Status	0x0300	3.4.6. Status TLV
Extended Status	0x0301	3.5.1. Сообщение Notification
Returned PDU	0x0302	3.5.1. Сообщение Notification
Returned Message	0x0303	3.5.1. Сообщение Notification
Common Hello Parameters	0x0400	3.5.2. Сообщение Hello
IPv4 Transport Address	0x0401	3.5.2. Сообщение Hello
Configuration Sequence Number	0x0402	3.5.2. Сообщение Hello
IPv6 Transport Address	0x0403	3.5.2. Сообщение Hello
Common Session Parameters	0x0500	3.5.3. Сообщение Initialization
ATM Session Parameters	0x0501	3.5.3. Сообщение Initialization
Frame Relay Session Parameters	0x0502	3.5.3. Сообщение Initialization
Label Request Message ID	0x0600	3.5.7. Сообщение Label Mapping
Vendor-Private	0x3E00-0x3EFF	3.6.1.2. Сообщение LDP Vendor-Private
Experimental	0x3F00-0x3FFF	3.6.2. Экспериментальные расширения LDP

3.9. Список кодов состояния

Список кодов состояния (Status Code), определённых в данной версии протокола, приведён в таблице.

В колонке E показано значение, требуемое для бита E в поле Status Code; Колонка «Данные состояния» содержит значение 30-битового поля Status Data в Status Code TLV. Отметим, что установка флага F в поле Status Code определяется LSR, передающим Status TLV.

Код состояния	E	Данные состояния	Описание
Success	0	0x00000000	3.4.6. Status TLV
Bad LDP Identifier	1	0x00000001	3.5.1.2. События, о которых сигнализируют сообщения Notification
Bad Protocol Version	1	0x00000002	3.5.1.2. События, о которых сигнализируют сообщения Notification
Bad PDU Length	1	0x00000003	3.5.1.2. События, о которых сигнализируют сообщения Notification
Unknown Message Type	0	0x00000004	3.5.1.2. События, о которых сигнализируют сообщения Notification
Bad Message Length	1	0x00000005	3.5.1.2. События, о которых сигнализируют сообщения Notification
Unknown TLV	0	0x00000006	3.5.1.2. События, о которых сигнализируют сообщения Notification
Bad TLV Length	1	0x00000007	3.5.1.2. События, о которых сигнализируют сообщения Notification
Malformed TLV Value	1	0x00000008	3.5.1.2. События, о которых сигнализируют сообщения Notification
Hold Timer Expired	1	0x00000009	3.5.1.2. События, о которых сигнализируют сообщения Notification
Shutdown	1	0x0000000A	3.5.1.2. События, о которых сигнализируют сообщения Notification
Loop Detected	0	0x0000000B	2.8. Детектирование петель
Unknown FEC	0	0x0000000C	3.4.1.1. Процедуры FEC
No Route	0	0x0000000D	3.5.8. Сообщение Label Request
No Label Resources	0	0x0000000E	3.5.8. Сообщение Label Request
Label Resources/Available	0	0x0000000F	3.5.8. Сообщение Label Request
Session Rejected/No Hello	1	0x00000010	2.5.3. Инициализация сессии
Session Rejected/Parameters Advertisement Mode	1	0x00000011	2.5.3. Инициализация сессии
Session Rejected/Parameters Max PDU Length	1	0x00000012	2.5.3. Инициализация сессии
Session Rejected/Parameters Label Range	1	0x00000013	2.5.3. Инициализация сессии
KeepAlive Timer Expired	1	0x00000014	3.5.1.2. События, о которых сигнализируют сообщения Notification
Label Request Aborted	0	0x00000015	3.5.9. Сообщение Label Abort Request
Missing Message Parameters	0	0x00000016	3.5.1.2. События, о которых сигнализируют сообщения Notification
Unsupported Address Family	0	0x00000017	3.4.1.1. Процедуры FEC, 3.5.5.1. Процедуры для сообщений Address
Session Rejected/Bad KeepAlive Time	1	0x00000018	2.5.3. Инициализация сессии
Internal Error	1	0x00000019	3.5.1.2. События, о которых сигнализируют сообщения Notification

3.10. Общепринятые значения

3.10.1. Порты UDP и TCP

Для сообщений LDP Hello используется порт UDP 646.

Для организованных сеансов LDP используется порт TCP 646.

3.10.2. Неявная NULL-метка

Метка Implicit NULL определена в [RFC3031] следующим образом[1]:

> Метка Implicit NULL представляет собой метку специального назначения, которую LSR может связать с адресным префиксом. Если LSR Ru из своего отображения ILM определяет, что помеченный пакет P необходимо переслать маршрутизатору Rd, но Rd распространяет для соответствующего адресного префикса метку Implicit NULL, то вместо замены верхней метки в стеке Ru просто выталкивает метку из стека и пересылает пакет Rd.

Неявная NULL-метка представляется в LDP, как Generic Label TLV с полем Label=3 в соответствии с определением [RFC3032].

4. Согласование с IANA

LDP определяет несколько пространств имён, требующих управления:

- Message Type;
- TLV Type;
- FEC Type;
- Status Code;
- Experiment ID;

В последующих параграфах приведены рекомендации по управлению этими пространствами имён.

4.1. Пространство имён Message Type

LDP делит пространство типов сообщений на три части. Ниже приведены рекомендации по управлению этими частями.

- *Message Type 0x0000 - 0x3DFF*. Сообщения, типы которых относятся к этому диапазону, являются частью базового протокола LDP. В соответствии с политикой [IANA], значения Message type из этого диапазона выделяются по процедуре IETF Consensus.
- *Message Type 0x3E00 - 0x3EFF*. Сообщения, типы которых относятся к этому диапазону, зарезервированы для фирменных (Vendor-Private) расширений и ответственность за них лежит на производителях (см. параграф 3.6.1.2. Сообщение LDP Vendor-Private). Участие IANA в распределении этой части пространства имён не требуется.

[1]Ниже представлен фрагмент перевода RFC 3031, опубликованного на сайте. *Прим. перев.*

- *Message Type 0x3F00 - 0x3FFF*. Сообщения, типы которых относятся к этому диапазону, зарезервированы для экспериментальных расширений и ответственность за них ложится на экспериментаторов (см. параграф 3.6.2. Экспериментальные расширения LDP). Участие IANA в распределении этой части пространства имён не требуется, однако IANA отвечает за управление частью пространства имён Experiment ID (см. ниже).

4.2. Пространство имён TLV Type

LDP делит пространство типов TLV на три части. Ниже приведены рекомендации по управлению этими частями.

- *TLV Type 0x0000 - 0x3DFF*. Типы TLV из этого диапазона являются частью базового протокола LDP. В соответствии с политикой [IANA], значения TLV type из этого диапазона выделяются по процедуре IETF Consensus.
- *TLV Type 0x3E00 - 0x3EFF*. Типы TLV из этого диапазона зарезервированы для фирменных (Vendor-Private) расширений и ответственность за них ложится на производителей (см. параграф 3.6.1.1. LDP Vendor-Private TLV). Участие IANA в распределении этой части пространства имён не требуется.
- *TLV Type 0x3F00 - 0x3FFF*. Типы TLV из этого диапазона зарезервированы для экспериментальных расширений и ответственность за них ложится на экспериментаторов (см. параграфы 3.6.2. Экспериментальные расширения LDP и 4.5. Пространство имён Experiment ID). Участие IANA в распределении этой части пространства имён не требуется, однако IANA отвечает за управление частью пространства имён Experiment ID (см. ниже).

4.3. Пространство имён FEC Type

Диапазон типов FEC - 0 - 255.

В соответствии с политикой [IANA], значения FEC из диапазона 0 - 127 выделяются по процедуре IETF Consensus, из диапазона 128 - 191 - в порядке подачи заявок (процедура First Come First Served), а значения 192 - 255 зарезервированы для приватного использования (Private Use).

4.4. Пространство имён Status Code

Диапазон значений кодов состояния (Status Code) - 0x00000000 - 0x3FFFFFFF.

В соответствии с политикой [IANA], значения значения Status Code из диапазона 0x00000000 - 0x1FFFFFFF выделяются по процедуре IETF Consensus, из диапазона 0x20000000 - 0x3EFFFFFF - в порядке подачи заявок (процедура First Come First Served), а значения из диапазона 0x3F000000 - 0x3FFFFFFF зарезервированы для приватного использования (Private Use).

4.5. Пространство имён Experiment ID

Диапазон значений Experiment ID - 0x00000000 - 0xffffffff.

В соответствии с политикой [IANA], значения значения Experiment ID из диапазона 0x00000000 - 0xefffffff выделяются в порядке подачи заявок (процедура First Come First Served), а значения из диапазона 0xf0000000 - 0xffffffff зарезервированы для приватного использования (Private Use).

5. Вопросы безопасности

В этом разделе рассматриваются угрозы, которым может быть подвержен протокол LDP и меры по снижению влияния таких угроз.

5.1. Обманные пакеты

Два типа коммуникаций LDP могут быть подвержены атакам на базе обманных пакетов.

1. *Раскрытие обмена по протоколу UDP*.

   LSR показывает свои намерения организовать и поддерживать сессии LDP путём периодической передачи сообщений Hello. Приём сообщения Hello служит для организации новой «Hello-смежности», если такие отношения не были организованы ранее, или обновления существующей смежности. Обманные пакеты Hello для существующих отношений смежности могут к возникновению тайм-аутов для смежности в прерыванию связанных сеансов LDP. Угроза может быть реализована путём передачи обманных сообщений Hello с малым значением Hold Time, заставляющим получателя ожидать приёма сообщений Hello в течение обманно заданного короткого интервала, в то время, как легитимный сосед (партнёр по сессии) будут передавать сообщения Hello более редко (с согласованной ранее частотой).

   Маршрутизаторы LSR, непосредственно соединённые между собой на канальном уровне обмениваются через соединяющий их канал сообщениями Basic Hello. Угроза воздействия обманных сообщений Basic Hellos может быть снижена двумя путями:

   - восприятие сообщений Basic Hello только через интерфейсы, к которым подключены доверенные LSR;
   - игнорирование сообщений Basic Hello, не направленных по групповому адресу All Routers on this Subnet (все маршрутизаторы данной подсети).

   Маршрутизаторы LSR, не соединённые непосредственно на канальном уровне, могут использовать сообщения Extended Hello для индикации своего намерения организовать сеанс LDP. LSR может снизить влияние угроз использования обманных пакетов Extended Hello за счёт фильтрации таких пакетов с восприятием только тех пакетов, адреса отправителей которых соответствуют заданному списку.

2. *Сеансовый обмен данными по протоколу TCP*.

LDP задаёт использование опции TCP MD5 Signature для обеспечения достоверности и целостности сеансовых сообщений.

В [RFC2385] отмечено, что алгоритм MD5 некоторые считают недостаточно сильным для защиты протокола LDP. Указано также возможность реализации похожей опции TCP с более строгим алгоритмом хэширования (например, SHA-1). Насколько нам известно, такая опция TCP не была определена и реализована. Тем не менее, отметим, что протокол LDP может использовать любые методы цифровой подписи TCP и при определении и реализации более строгого, нежели MD5, алгоритма обновление LDP для использования такого алгоритма не вызовет существенных затруднений.

5.2. Конфиденциальность

LDP не поддерживает механизмов обеспечения конфиденциальности при распространении меток.

Требования безопасности при распространении меток в значительной мере совпадают с требованиями безопасности для протоколов распространения маршрутной информации. За счёт механизма контроля подлинности и целостности сообщений протокол LDP обеспечивает уровень защиты, который достаточно хорош, но не превосходит уровня защиты самих протоколов маршрутизации. Рассмотрение более общего вопроса о требованиях к уровню конфиденциальности для протоколов маршрутизации выходит за рамки данного документа.

Можно сказать, что защита конфиденциальности при распространении меток нужна для предотвращения угрозы подмены меток. Однако такая защита не поможет против атак с обманными метками, пока пакеты с метками будут передаваться в открытом (незашифрованном) виде. Более того, атаки с использованием обманных меток могут быть организованы без наличия сведений о привязках FEC к меткам.

Для предотвращения атак с обманными метками требуется обеспечить, чтобы пакеты данных исходили от доверенных LSR, а включённые в пакеты метки были корректно получены самими LSR.

5.3. Отказ служб

Протокол LDP потенциально уязвим для двух типов атак на отказ служб (DoS):

1. *Общеизвестный номер порта UDP для LDP Discovery*.

   Администратор LSR может защититься от DoS-атак с использованием сообщений Basic Hello за счёт организации прямых соединений только с доверенными LSR, которые не будут являться источниками атак. Интерфейсы к партнёрам, расположенным в том же административном домене, не рассматриваются в качестве источника угроз, поскольку они находятся под собственным административным контролем. Интерфейсы к внешним партнёрам могут быть потенциальным источником атаки, поскольку они находятся под чужим управлением. Администратор может снизить риск атак за счёт организации соединений LSR только с теми внешними партнёрами, которые не могут послужить источником атак Basic Hello.

   DoS-атаки с использованием сообщений Extended Hello представляют более серьёзную опасность. Эту угрозу можно предотвратить путём фильтрации сообщений Extended Hello с использованием списков доступа, определяющих адреса, для которых разрешён механизм Extended Discovery. Однако фильтрация требует ресурсов LSR.

   В средах, где можно идентифицировать доверенное облако MPLS, маршрутизаторы LSR на краях облака могут служить для защиты внутренних LSR этого облака от DoS-атак с использованием сообщений Extended Hello путём фильтрации сообщений Extended Hello, приходящих из=за пределов доверенного облака MPLS, принимая лишь сообщения с адресов, разрешённых списками доступа. Такая фильтрация позволяет защитить внутренние LSR, но будет потреблять ресурсы краевых маршрутизаторов.

2. *Общеизвестный номер порта TCP для организации сессий LDP*.

   Подобно другим протоколам управляющего плана, которые используют транспорт TCP, протокол LDP может быть целью DoS-атак (например, SYN). Протокол LDP уязвим для таких атак не более и не менее других подобных протоколов, работающих на базе TCP.

   Для снижения угрозы таких атак можно воспользоваться приведёнными ниже рекомендациями.

   - LSR **следует** избегать «неразборчивого» (promiscuous) прослушивания TCP на предмет организации сеансов LDP. **Следует** прослушивать порт только для обнаруженных ранее партнёров. Это позволит отбрасывать пакеты, связанные с атакой, до их обработки, поскольку пакеты атакующего в большинстве случаев не будут соответствовать существующим или организуемым соединениям.

   - Использование опции MD5 позволяет защититься от SYN-атак, поскольку пакеты не будут восприниматься до проверки корректности значения хэша MD5. Однако такая проверка будет требовать расчёта контрольной суммы для каждого принимаемого сегмента SYN (расход ресурсов маршрутизатора).

   - Использование списков доступа на границе облака MPLS (как описано выше для сообщений Extended Hello) позволит защитить внутренние узлы от атак извне облака.

6. Направления дальнейших исследований

Перечисленные ниже вопросы не решены в текущей версии протокола LDP и требуют дополнительного изучения.

- В параграфе 2.16 [RFC3031] требуется, чтобы начальное согласование протокола распространения меток между LSR позволяло каждому LSR определить способность партнёра поддерживать стек меток. В данной версии LDP предполагается, что маршрутизаторы LSR поддерживают это для всех типов меток, кроме ATM и Frame Relay. В будущих версиях протокола определение упомянутой возможности может стать частью согласования сессии.

- Поддержка в LDP классов обслуживания (CoS) в данной версии не определена. Поддержка CoS может быть включена в будущие версии.
- Поддержка протоколом LDP групповой адресации не определена в текущей версии и может быть включена в будущие версии протокола.
- Поддержка коммутации по меткам с множеством путей (multipath label switching) не определена в текущей версии протокола LDP и может быть включена в будущие версии.
- Поддержка сигнализации максимального размера блока данных (MTU) не определена в текущей версии LDP и рассматривается в экспериментальном документе [LDP-MTU].
- В текущей версии спецификации не решён вопрос обнаружения партнёров в средах с множественным доступом без широковещания (NBMA[1]). В рамках данной спецификации возможно обнаружение за счёт использования расширенных процедур детектирования. Задача определения механизма обнаружения семантически похожа на процедуру Basic Discovery (ограничение в 1 интервал и привязка hello-смежности к интерфейсу), которая использует заданный в конфигурации адрес соседа, и требует дальнейшего изучения.
- В текущей спецификации не поддерживается сброс отношений смежности. Мотивация такого сброса и определение подходящего механизма требуют дополнительного изучения.
- В текущей версии спецификации отсутствует метод защиты сообщений Hello для детектирования обманных сообщений. Сценарии, в которых такая защита требуется, и механизм защиты требуют дополнительного изучения.
- В текущей версии спецификации не поддерживается возможность детектирования быстрого рестарта управляющего плана с потерей состояния. Метод решения этой задачи (возможно за счёт использования номера «инкарнации/экземпляра» в сообщении Hello) требует дальнейшего изучения.
- В текущей версии спецификации не поддерживаются сообщения end of LIB (аналог end of RIB в BGP), которые LDP LSR (работающий в режиме DU) может использовать при следующей организации сессии. Обсуждение необходимости такого механизма и его реализации требует дальнейшего изучения.
- В настоящей спецификации не рассматриваются ситуации, когда разные LSR анонсируют один и тот же адрес. Такие ситуации возникают обычно в результате конфигурационных ошибок и цель в данном случае заключается в том, чтобы обеспечить маршрутизаторы LSR, анонсирующие один адрес информацией, которой оператору будет достаточно для исправления ошибок в конфигурации. Спецификация такого механизма будет опубликована в форме отдельного документа.

7. Отличия от RFC 3036

Ниже приведён список изменений, внесённых в RFC 3036.

1. Удалён класс Host Address FEC и ссылки на него, поскольку в реализациях данный класс не используется.
2. Список литературы разделен на нормативные документы и дополнительные источники.
3. Из списка нормативных документов удалён «MPLS using ATM VP Switching», а также удалены ссылки на него.
4. Удалена ссылка на RFC 1700 с заменой на http://www.iana.org/assignments/address-family-numbers.
5. Удалена ссылка на RFC 1771 с заменой на RFC 4271.
6. Прояснено использование флага F.
7. Добавлена опция, позволяющая использовать «расщепление горизонта» (split horizon) при упорядоченном управлении (Ordered Control).
8. Прояснена обработка сообщений с флагом U на этапе инициализации сессии.
9. Прояснена обработка флага E в процессе инициализации сессии.
10. Добавлен текст, разъясняющий, что сообщение Shutdown в диаграмме состояний реализовано, как уведомление с полем Status TLV, показывающим критическую ошибку.
11. Добавлена ситуация, когда размер TLV слишком мал (при рассмотрении TLV с некорректным форматом).
12. Разъяснены угрозы, связанные с обманными сообщениями Hello.
13. Добавлены ссылки на RFC 4271 и RFC 4278, а также пояснительный текст в части использования опции MD5.
14. Добавлен текст из RFC 3031, разъясняющий использование неявных NULL-меток.
15. Включено представление DLCI для удаления нормативной ссылки на RFC 3034.
16. Ссылки на RFC 3031, RFC 3032 и RFC 3034 перенесены в раздел дополнительной литературы.
17. В параграфе, описывающем обработку неизвестных TLV, удалена ссылка на несуществующий раздел (ошибка в исходном документе).
18. Добавлен текст, разъясняющий, как добиться взаимодействия при передаче фирменных TLV и сообщений.
19. В процедурах обработки запросов меток для случая обнаружения петель изменена процедура передачи уведомления перед прерыванием остальной части обработки.
20. В процедурах обработки освобождения меток разъяснено поведение поддерживающих слияние LSR.

[1]Non-Broadcast Multi-Access.

21. В процедурах обработки освобождения меток разъяснено поведение в случае получения неизвестного FEC.
22. В примечании 4 (A.1.7. Детектирование смены FEC Next Hop) изменён текст для указания корректного набора условий передачи запроса на метку (опечатка в исходном документе).
23. В процедурах приложения A.1.14. Принятие LSR решения о прекращении коммутации по меткам для FEC разъяснено, что метку не допускается использовать повторно до получения сообщения о её освобождении.
24. В описании процедуры Prepare_Label_Mapping_Attributes добавлено примечание, относящееся к трактовке неизвестных TLV в соответствии со значениями флагов U и F.
25. В процедурах обработки сообщений Address разъяснено поведение для случаев, когда LSR получает повторный анонс для анонсированного ранее адреса или отзыв для адреса от LSR, который этот адрес не анонсировал ранее.
26. В описании процедур обработки полученного отображения метки разъяснено значение PrevAdvLabel для случая, когда ранее не передавалось сообщения с анонсом этой метки.
27. В описании процедур обработки полученного отображения метки для случая обнаружения петли изменён процесс отправки уведомления до прерывания оставшейся части обработки.
28. В описании процедур обработки полученного отображения метки исправлен этап LMp.10 для обработки сообщений об отображении для дополнительных (несливаемых) LSP того же класса FEC.
29. В описании процедуры обработки полученного отображения метки разъяснено поведение при получении дубликата метки для того же класса FEC.
30. В описании процедур обработки запросов на прерывание (Label Abort Request) разъяснено поведение для не поддерживающих слияния LSR.
31. Добавлен ряд пунктов в список вопросов, требующих дальнейшего изучения:
    - расширение для обмена информацией об MTU;
    - базовое детектирование партнёров в средах NBMA;
    - опция для прерывания отношений смежности;
    - механизмы защиты сообщений Hello;
    - детектирование быстрого рестарта управляющего плана с потерей состояния;
    - поддержка сообщений end of LIB;
    - механизмы для случая анонсирования одного адреса разными LSR.

8. Благодарности

Этот документ был создан в процессе подготовки спецификации протокола LDP в качестве проекта стандарта. Исходный документ был опубликован, как RFC 3036 в январе 2001. Он был подготовлен рабочей группой MPLS (IETF) и написан совместно Loa Andersson, Paul Doolan, Nancy Feldman, Andre Fredette и Bob Thomas.

Идеи и текст RFC 3036 были взяты из множества источников. Мы благодарим Rick Boivie, Ross Callon, Alex Conta, Eric Gray, Yoshihiro Ohba, Eric Rosen, Bernard Suter, Yakov Rekhter и Arun Viswanathan за их вклад в подготовку RFC 3036.

Редакторы выражают свою признательность Eric Gray, David Black и Sam Hartman за просмотр и и замечания к текущей версии документа.

В дополнение к этому редакторы выражают благодарность члена рабочей группы MPLS за их идеи и поддержку при подготовке документа, особенно отмечая Eric Rosen, Luca Martini, Markus Jork, Mark Duffy, Vach Kompella, Kishore Tiruveedhula, Rama Ramakrishnan, Nick Weeds, Adrian Farrel и Andy Malis.

9. Литература

9.1. Нормативные документы

[IANA] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

[RFC1321] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.

[ASSIGNED_AF] http://www.iana.org/assignments/address-family-numbers

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2385] Heffernan, A., "Protection of BGP Sessions via the TCP MD5 Signature Option", RFC 2385, August 1998.

[RFC3035] Davie, B., Lawrence, J., McCloghrie, K., Rosen, E., Swallow, G., Rekhter, Y., and P. Doolan, "MPLS using LDP and ATM VC Switching", RFC 3035, January 2001.

[RFC3037] Thomas, B. and E. Gray, "LDP Applicability", RFC 3037, January 2001.

9.2. Дополнительная литература

[CRLDP] Jamoussi, B., Ed., Andersson, L., Callon, R., Dantu, R., Wu, L., Doolan, P., Worster, T., Feldman, N., Fredette, A., Girish, M., Gray, E., Heinanen, J., Kilty, T., and A. Malis, "Constraint-Based LSP Setup using LDP", RFC 3212, January 2002.

[LDP-MTU] Black, B. and K. Kompella, "Maximum Transmission Unit Signalling Extensions for the Label Distribution Protocol", RFC 3988, January 2005.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, April 1998.

[RFC2702] Awduche, D., Malcolm, J., Agogbua, J., O'Dell, M., and J. McManus, "Requirements for Traffic Engineering Over MPLS", RFC 2702, September 1999.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.

[RFC3032] Rosen, E., Tappan, D., Fedorkow, G., Rekhter, Y., Farinacci, D., Li, T., and A. Conta, "MPLS Label Stack Encoding", RFC 3032, January 2001.

[RFC3034] Conta, A., Doolan, P., and A. Malis, "Use of Label Switching on Frame Relay Networks Specification", RFC 3034, January 2001.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC4278] Bellovin, S. and A. Zinin, "Standards Maturity Variance Regarding the TCP MD5 Signature Option (RFC 2385) and the BGP-4 Specification", RFC 4278, January 2006.

Приложение A. Процедуры распространения меток LDP

В этом разделе описано распространение меток в терминах откликов LSR на следующие события:

- приём сообщения Label Request;
- приём сообщения Label Mapping;
- приём сообщения Label Abort Request;
- приём сообщения Label Release;
- приём сообщения Label Withdraw;
- идентификация нового класса FEC;
- детектирование изменения следующего интервала для FEC;
- приём сообщения Notification - Label Request Aborted;
- приём сообщения Notification - No Label Resources;
- приём сообщения Notification - No Route;
- приём сообщения Notification - Loop Detected;
- приём сообщения Notification - Label Resources Available;
- детектирование доступности локальных ресурсов для меток;
- принятие LSR решения о прекращении коммутации меток для FEC;
- тайм-аут для отложенного запроса метки.

Спецификация поведения LSR в ответ на события представляется в форме трёх частей:

1. *Описание* - обзор отклика LSR на событие.
2. *Контекст* - список элементов, упомянутых в алгоритме (п. 3).
3. *Алгоритм* - алгоритм действий для отклика LSR на событие.

В описании могут быть опущены детали отклика LSR (такие, как учёт или детали поведения, зависящие от режима анонсирования меток LSR, режима управления или режима удержания меток). Цель заключается в полном и однозначном указании отклика LSR в Алгоритме.

При описании алгоритмов используются процедуры, определённые в спецификации архитектуры MPLS [RFC3031] для трафика с поэтапной маршрутизацией. К таким процедурам относятся:

- Процедуры распространения меток (Label Distribution), которые выполняются нисходящим LSR для решения вопроса о распространении меток для FEC партнёрам LDP. Архитектура определяет 4 процедуры Label Distribution:
  - нисходящее распространение меток без запроса с независимым управлением (Downstream Unsolicited Independent Control), называемое PushUnconditional в [RFC3031];
  - нисходящее распространение меток без запроса с упорядоченным управлением (Downstream Unsolicited Ordered Control), называемое PushConditional в [RFC3031];
  - нисходящее распространение меток по запросам с независимым управлением (Downstream On Demand Independent Control), называемое PulledUnconditional в [RFC3031];
  - нисходящее распространение меток по запросам с упорядоченным управлением (Downstream On Demand Ordered Control), называемое PulledConditional в [RFC3031].
- Процедура отзыва меток (Label Withdrawal) которая выполняется нисходящим LSR для решения вопроса от отзыве отображения FEC-label, анонсированного ранее партнёрам LDP. Архитектура определяет одну процедуру отзыва - Label Withdrawal. Всякий раз при разрыве связки метки с классом FEC маршрутизатор LSR

должен отозвать соответствующее отображение для всех партнёров LDP, которым это отображение анонсировалось.

- Процедуры запроса метки (Label Request), которые выполняются восходящим LSR для решения вопроса о явном запросе у нисходящего LSR привязки метки к FEC и передачи соответствующего отображения. Архитектура определяет три процедуры Label Request:
  - никогда не запрашивать (Request Never) - LSR никогда не запрашивает меток;
  - запрос при необходимости (Request When Needed) - LSR запрашивает метку при возникновении потребности в ней;
  - запрос по запросу (Request On Request) - эта процедура используется LSR, не поддерживающими слияния. LSR запрашивает метку при получении запроса на неё или при возникновении потребности в текой метке.
- Процедуры освобождения метки (Label Release), которые выполняются восходящим LSR для решения вопроса об освобождении ранее полученного отображения метки на FEC. Архитектура определяет две процедуры Label Release:
  - консервативное удержание меток (Conservative Label retention), именуемое ReleaseOnChange в [RFC3031];
  - либеральное удержание меток (Liberal Label retention), именуемое NoReleaseOnChange в [RFC3031].
- Процедуры использования меток (Label Use), которые выполняются LSR для решения вопроса о начале использования метки FEC для коммутации/пересылки. Архитектура определяет три процедуры Label Use:
  - использовать сразу (Use Immediate) - LSR незамедлительно начинает использование меток, полученных от следующего интервала для FEC при коммутации/пересылке пакетов;
  - использовать при отсутствии петель (Use If Loop Free) - LSR использует метку FEC, полученную от следующего интервала для этого FEC при коммутации/пересылке, если он определил, что при использовании такой метки не возникает петли;
  - использовать, если петель не обнаружено (Use If Loop Not Detected) - эта процедура совпадает с Use Immediate, пока LSR не детектирует петли в FEC LSP. Использование метки FEC для коммутации/пересылки будет осуществляться до тех пор, пока не сменится следующий интервал для FEC или не будет обнаружена петля.

  Данная версия LDP не включает механизма предотвращения петель, следовательно перечисленные ниже процедуры не используют процедуру Use If Loop Free.
- Процедуры Label No Route (именуется NotAvailable в [RFC3031]), которые выполняются восходящим LSR для решения вопроса о реагировании на сообщения No Route от нисходящего LSR в ответ на запрос отображения FEC-метка. Спецификация архитектуры определяет две процедуры Label No Route:
  - повторный запрос (Request Retry) - LSR следует повторить запрос метки позднее;
  - без повторного запроса (No Request Retry) - LSR следует считать, что нисходящий маршрутизатор LSR будет обеспечивать отображение метки, когда ему известен следующий интервал пересылки, и повторять запрос не следует.

A.1. Обработка событий при распространении меток

В этом разделе определены процедуры распространения меток LDP путём задания алгоритма обработки для каждого события при распространении меток. Требование к реализациям LDP заключается в том, что обработка событий должна давать эффект, указанный в алгоритме. Таким образом, реализации не обязаны в точности выполнять все шаги, указанные в алгоритме, если она обеспечивает тот же самый эффект.

Алгоритмы обработки событий при распространении меток используют общий набор действий. Приведённые ниже спецификации группируют эти действия в процедурные блоки. Спецификации для таких общих процедур даны в отдельном приложении A.2. Общие процедуры распространения меток.

Реализации будут использовать структуры данных для хранения информации о протокольных действиях. В этом приложении указана сохраняемая информация, которая нужна для описания алгоритма, и предполагается возможность нахождения такой информации при возникновении потребности в ней. Детали структур данных спецификация не задаёт.

A.1.1. Получение Label Request

Описание

Отклик LSR на получение запроса отображения FEC-label от партнёра LDP может включать одно или несколько перечисленных ниже действий:

- передача запрашивающему LSR уведомления, показывающего, почему отображение метки для FEC не может быть предоставлено;
- передача отображения FEC-label запрашивающему LSR;
- передача запроса для отображения FEC-label на следующий интервал FEC;
- установка меток для использования LSR при коммутации/пересылке.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- MsgSource - партнёр LDP, передавший сообщение;
- FEC - класс FEC, указанный в сообщении;
- RAttributes - атрибуты, полученные в сообщении (например, Hop Count, Path Vector);
- SAttributes - атрибуты, которые будут включаться в сообщение Label Request, передаваемое FEC Next Hop;
- StoredHopCount - счётчик интервалов, ранее записанных для FEC (при наличии таковых).

Алгоритм

LRq.1 Выполнить процедуру Check_Received_Attributes (MsgSource, LabelRequest, RAttributes).

При обнаружении петли переход на LRq.4.

LRq.2 Это следующий интервал для FEC?

Если нет, перейти на LRq.5.

LRq.3 MsgSource совпадает с адресом следующего интервала (Next Hop)?

Если нет, перейти на LRq.6.

LRq.4 Выполнить процедуру Send_Notification (MsgSource, Loop Detected).

Перейти на LRq.13.

LRq.5 Выполнить процедуру Send_Notification (MsgSource, No Route).

Перейти на LRq.13.

LRq.6 LSR уже получал запрос метки для FEC от MsgSource?

Если нет, перейти на LRq.8 (см. примечание 1).

LRq.7 Запрос метки является дубликатом?

Если да, перейти на LRq.13 (см. примечание 2).

LRq.8 Записать запрос метки для FEC, полученный от MsgSource, и пометить его, как ожидающий.

LRq.9 Выполнить процедуру LSR Label Distribution:

Для режима Downstream Unsolicited Independent Control **или** Downstream On Demand Independent Control

1. LSR ранее получил и удерживает отображение метки для FEC от Next Hop?

   Если да, установить Propagating = IsPropagating.

   Если нет, установить Propagating = NotPropagating.
2. Выполнить процедуру Prepare_Label_Mapping_Attributes(MsgSource, FEC, RAttributes, SAttributes, Propagating, StoredHopCount).
3. Выполнить процедуру Send_Label (MsgSource, FEC, SAttributes).
4. LSR является выходом для FEC? **или** LSR ранее получил и удерживает отображение метки для FEC от Next Hop?

   Если да, перейти на LRq.11.

   Если нет, перейти на LRq.10.

Для режима Downstream Unsolicited Ordered Control **или** Downstream On Demand Ordered Control

1. LSR является выходом для FEC? **или** LSR ранее получил и удерживает отображение метки для FEC от Next Hop? (см. примечание 3)

   Если нет, перейти на LRq.10.
2. Выполнить процедуру Prepare_Label_Mapping_Attributes(MsgSource, FEC, RAttributes, SAttributes, IsPropagating, StoredHopCount)
3. Выполнить процедуру Send_Label (MsgSource, FEC, SAttributes).

   Перейти на LRq.11.

LRq.10 Выполнить процедуру LSR Label Request:

для Request Never

1. Перейти на LRq.13.

для Request When Needed **или** Request On Request

1. Выполнить процедуру Prepare_Label_Request_Attributes (Next Hop, FEC, RAttributes, SAttributes);
2. Выполнить процедуру Send_Label_Request (Next Hop, FEC, SAttributes).

   Перейти на LRq.13.

LRq.11 LSR успешно отправил метку для FEC по адресу MsgSource?

Если нет, перейти на LRq.13 (см. примечание 4).

LRq.12 Выполнить процедуру LSR Label Use.

Для Use Immediate **или** For Use If Loop Not Detected

1. Установить метку, переданную MsgSource, и метку от Next Hop (если LSR не является выходным) для коммутации/пересылки.

LRq.13 **Завершено**.

Примечания

1. Если MsgSource не является LSR со слиянием меток, этот маршрутизатор будет передавать запрос метки для каждого восходящего партнёра LDP, который запросил у него метку для FEC. LSR должен быть способен отличать запросы от MsgSource, не поддерживающих слияния меток, от дубликатов запросов меток.

   LSR использует идентификатор сообщения Label Request для детектирования дубликатов запроса. Это означает, что LSR (восходящий партнёр) не может повторно использовать значение идентификатора, указанного в сообщении Label Request, пока обработка запроса не будет завершена.

2. Когда LSR передаёт партнёру запрос метки, он записывает информацию об этой передаче и помечает запрос, как ожидающий завершения. Пока запрос остаётся помеченным, как ожидающий, LSR **не следует** отправлять партнёру новый запрос на ту же метку. Повторный запрос будет дубликатом. Описанная ниже процедура Send_Label_Request следует этому правилу.

   Дубликат запроса рассматривается, как протокольная ошибка. Такие дубликаты принимающему LSR **следует** отбрасывать (возможно, с генерацией соответствующего уведомления для MsgSource).

3. Если LSR не поддерживает слияния меток, эта проверка будет давать отрицательный результат.

4. Процедура Send_Label может давать отрицательный результат по причине нехватки ресурсов. В таких случаях LSR **не следует** выполнять процедуру Label Use.

A.1.2. Получение Label Mapping

Описание

Отклик LSR га получение отображения FEC-label от партнёра LDP может включать одно или несколько из перечисленных ниже действий:

- передача сообщения Label Release для метки FEC партнёру LDP;
- передача сообщений Label Mapping для FEC одному или множеству партнёров LDP;
- установка вновь полученной метки для использования LSR при коммутации/пересылке.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- MsgSource - партнёр LDP, передавший сообщение;
- FEC - класс FEC, указанный в сообщении;
- Label - метка, указанная в сообщении;
- PrevAdvLabel - метка для FEC (при наличии таковой), которая была анонсирована восходящему партнёру; в предположении, что метка ранее не анонсировалась, эта метка будет совпадать с меткой в обрабатываемом сообщении Label Mapping;
- StoredHopCount - счётчик интервалов, ранее записанных для FEC (при наличии таковых);
- RAttributes - атрибуты, полученные в сообщении (например, Hop Count, Path Vector);
- SAttributes - атрибуты, которые будут включаться в сообщение Label Request, передаваемое FEC Next Hop.

Алгоритм

LMp.1 Получено отображение, соответствующее ожидающему запросу метки для FEC, который ранее был передан MsgSource?

Если нет, перейти на LMp.3.

LMp.2 Удалить запись ожидающего запроса метки для FEC.

LMp.3 Выполнить процедуру Check_Received_Attributes (MsgSource, LabelMapping, RAttributes).

Если не обнаружено петли (No Loop Detected), перейти на LMp.9.

LMp.4 Получал ли LSR отображение метки для FEC от MsgSource? (см. примечание 1)

Если нет, перейти на LMp.8 (см. примечание 2).

LMp.5 Соответствует ли полученная ранее от MsgSource метка значению Label (т. е., метке в полученном сообщении)? (см. примечание 3).

Если нет, перейти на LMp.8 (см. примечание 4).

LMp.6 Удалить соответствующее отображение метки для FEC, полученное ранее от MsgSource.

LMp.7 Прекратить использование метки Label для коммутации/пересылки (см. примечание 5).

LMp.8 Выполнить процедуру Send_Message (MsgSource, Label Release, FEC, Label, Loop Detected Status code). Перейти на LMp.33.

LMp.9 Получал ли LSR отображение метки на FEC от MsgSource для интересующего LSP? (см. примечание 6).

Если нет, перейти на LMp.11.

LMp.10 Соответствует ли полученная ранее от MsgSource метка значению Label (т. е., метке в полученном сообщении)? (см. примечание 3).

или

Получено ли отображение метки в ответ на ожидающий обработки запрос, переданный MsgSource? (см. примечание 12).

Если да, перейти на LMp.11.

LMp.10a LSR работает в режиме Downstream Unsolicited? Если да, удалить отображение для метки, полученное ранее от MsgSource и прекратить использование этой метки для коммутации/пересылки. Выполнить процедуру Send_Message (MsgSource, Label Release, FEC, ранее полученная от MsgSource метка).

LMp.11 Определить Next Hop для FEC.

LMp.12 Является MsgSource следующим интервалом (Next Hop) для FEC?

Если да, перейти на LMp.14.

LMp.13 Выполнить процедуру LSR Label Release:

для консервативного удержания (Conservative Label):

1. перейти на Lmp.32;

для либерального удержания (Liberal Label):

1. записать отображение метки для FEC со значениями Label и RAttributes, полученными от MsgSource; перейти на LMp.33.

LMp.14 Является LSR входом для FEC?

Если нет, перейти на LMp.16.

LMp.15 Организовать использование Label для коммутации/пересылки.

LMp.16 Записать отображение метки для FEC со значениями Label и RAttributes, полученными от MsgSource.

LMp.17 Выполнить операции до LMp.31 для каждого партнёра (см. примечание 7).

LMp.18 LSR ранее передавал отображение метки на FEC партнёру Peer для рассматриваемого LSP? (см. примечание 8).

Если да, перейти на LMp.22.

LMp.19 LSR использует процедуру распространения меток Downstream Unsolicited Ordered Control?

Если нет, перейти на LMp.28.

LMp.20 Выполнить процедуру Prepare_Label_Mapping_Attributes (Peer, FEC, RAttributes, SAttributes, IsPropagating, StoredHopCount).

LMp.21 Выполнить процедуру Send_Message (Peer, Label Mapping, FEC, PrevAdvLabel, SAttributes) (см. примечание 13).

Перейти на LMp.28.

LMp.22 Выполнить операции до LMp.27 для каждого отображения метки на FEC, переданного ранее партнёру Peer.

LMp.23 Значение RAttributes в полученном отображении согласуется с переданным ранее партнёру Peer? Если да, продолжить итерацию с LMp.22 для следующего отображения (см. примечание 9).

LMp.24 Выполнить процедуру Prepare_Label_Mapping_Attributes (Peer, FEC, RAttributes, SAttributes, IsPropagating, StoredHopCount).

LMp.25 Выполнить процедуру Send_Message (Peer, Label Mapping, FEC, PrevAdvLabel, SAttributes) (см. примечание 10).

LMp.26 Обновить запись отображения метки на FEC, переданного ранее партнёру Peer для включения новых атрибутов.

LMp.27 Завершение итерации, начинающейся с LMp.22.

LMp.28 У LSR есть запросы метки для FEC от партнёра Peer, помеченные, как ожидающие?

Если нет, перейти на LMp.30.

LMp.29 Выполнить процедуру Label Distribution:

для режима Downstream Unsolicited Independent Control **или** For Downstream Unsolicited Ordered Control:

1. выполнить процедуру Prepare_Label_Mapping_Attributes (Peer, FEC, RAttributes, SAttributes, IsPropagating, UnknownHopCount);

2. выполнить процедуру Send_Label (Peer, FEC, Sattributes); в случае отказа продолжить итерацию для следующего партнёра с LMp.17;
3. если для партнёра Peer нет ожидающих запросов, перейти на LMp.30 (см. примечание 11);

для режима Downstream On Demand Independent Control **или** For Downstream On Demand Ordered Control:

1. выполнить итерацию до п. 5 для каждого запроса метки для FEC от партнёра Peer, помеченного, как ожидающий;
2. выполнить процедуру Prepare_Label_Mapping_Attributes (Peer, FEC, RAttributes, SAttributes, IsPropagating, UnknownHopCount);
3. выполнить процедуру Send_Label (Peer, FEC, Sattributes); в случае отказа продолжить итерацию для следующего партнёра с LMp.17;
4. удалить запись ожидающего запроса;
5. завершение итерации, начинающейся с п. 1;
6. перейти на LMp.30.

LMp.30 Выполнить процедуру Label Use:

для режима Use Immediate **или** Use If Loop Not Detected:

1. операции до п. 3 для каждого отображения метки на FEC, переданного ранее партнёру Peer;
2. организовать использование полученной и переданной партнёру метки для коммутации/пересылки;
3. завершение итерации, начинающейся с п 1;
4. перейти на LMp.31.

LMp.31 Завершение итерации, начинающейся с LMp.17.

Перейти на LMp.33.

LMp.32 Выполнить процедуру Send_Message (MsgSource, Label Release, FEC, Label).

LMp.33 **Завершено**.

Примечания

1. LSR, поддерживающим слияние, следует иметь не более одного полученного отображения метки на FEC для рассматриваемого LSP. Если слияние меток не поддерживается, число полученных отображений меток FEC может быть более 1 для LSP.
2. Если LSR обнаруживает петлю и ранее не получал отображения меток для FEC от MsgSource, он просто освобождает метку.
3. Соответствует ли поле Label в полученном сообщении любому отображению меток, идентифицированному на предыдущем этапе (LMp.4 или LMp.9)?
4. Незапрошенное отображение с разными метками от одного партнёра будет попыткой организовать коммутацию по множеству путей (multipath label switching), которая не поддерживается в данной версии LDP.
5. Если метка Label не используется для коммутации/пересылки, LMp.7 не даёт никакого эффекта.
6. Если полученное сообщение с отображением метки соответствует ожидающему запросу метки в LMp.1, значит (по определению) LSR ранее не получал отображения метки на FEC для рассматриваемого LSP. Если LSR является склеивающим восходящие метки для рассматриваемого LSP, ему следует иметь не более 1 полученного отображения. Если слияние меток не поддерживается, число полученных отображений на один класс FEC может быть больше 1 - по одному для каждого получающегося LSP.
7. Итерация LMp.17 включает MsgSource для обработки в случаях, когда LSR работает в режиме Downstream Unsolicited Ordered Control. Упорядоченное управления не позволяет LSR анонсировать метку для FEC, пока он не получит отображение метки от своего следующего интервала (MsgSource) для FEC.
8. Если LSR поддерживает слияние LSP, он может иметь отображения меток для FEC LSP, переданные ранее одному или множеству партнёров. Если LSR не поддерживает слияния, у него может быть отображение метки для рассматриваемого LSP, переданное не более, чем одному LSR.
9. При этой проверке рассматривается атрибут Loop Detection Path Vector. Если полученное значение RAttributes включает Path Vector и ранее значение Path Vector не передавалось партнёру Peer или полученное значение Path Vector не соответствует значению Path Vector, ранее переданному партнёру, атрибуты считаются не соответствующими друг другу. Отметим, что LSR не обязан сохранять полученное значение Path Vector после его распространения в сообщении с отображением. Если LSR не сохраняет Path Vector, у него не будет возможности проверить соответствие вновь полученному значению Path Vector. Это означает, что LSR при получении отображения с Path Vector всякий раз должен распространять значение Path Vector.
10. В LMp.22 - LMp.27 обрабатываются ситуации, которые могут возникать, когда LSR, использующий независимое управление, получает отображение от нисходящего партнёра после того, как было передано отображение восходящему партнёру. В таких случаях от LSR требуется распространять в восходящем направлении все изменившиеся атрибуты (такие, как Hop Count). Если включено детектирование петель, в число распространяемых атрибутов необходимо включать Path Vector.

11. LSR, работающий в режиме Downstream Unsolicited, **должен** обрабатывать все получаемые сообщения Label Request. При наличии ожидающих обработки запросов меток для выполнения этих запросов следует переходить к процедурам Downstream on Demand.
12. Как определено в LMp.1.
13. LSR, работающий в режиме Ordered Control, может на этом этапе партнёра, от которого он получил анонс, потребовавший генерации сообщения label-map. Это будет давать эффект «расщепления горизонта».

A.1.3. Получение Label Abort Request

Описание

Когда LSR получает сообщение Label Abort Request от своего партнёра, он проверяет, отвечал ли ранее на запрос указанной метки. Если такой отклик был передан, сообщение игнорируется. Если отклик не передавался, партнёру отправляется уведомление Label Request Aborted. В дополнение к этому при наличии ожидающего обработки запроса метки у нисходящего партнёра для рассматриваемого LSP, тому передаётся сообщение Label Abort Request для прерывания LSP.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- MsgSource - партнёр LDP, передавший сообщение;
- FEC - класс FEC, указанный в сообщении;
- RequestMessageID - идентификатор сообщения с запросом метки, обработка которого будет прервана;
- Next Hop - следующий интервал для FEC.

Алгоритм

LAbR.1 Сообщение соответствует ранее полученному от MsgSource сообщению Label Request? (см. примечание 1).

Если нет, переход на LAbR.12.

LAbR.2 LSR ответил на полученный ранее запрос метки?

Если да, переход на LAbR.12.

LAbR.3 Выполнить процедуру Send_Message (MsgSource, Notification, Label Request Aborted, TLV), где поле TLV содержит Label Request Message ID TLV, полученное в сообщении Label Abort Request.

LAbR.4 LSR имеет ожидающее обработки сообщение Label Request для FEC?

Если да, переход на LAbR.7.

LAbR.5 LSR имеет отображение метки для FEC?

Если нет, переход на LAbR.11.

LAbR.6 Генерация события: сообщение Received Label Release для FEC от MsgSource (см. примечание 2).

Перейти на LAbR.11.

LAbR.7 LSR поддерживает слияние LSP для FEC?

Если нет, переход на LAbR.9.

LAbR.8 Есть ожидающие обработки запросы метки для этого FEC?

Если да, переход на LAbR.11.

LAbR.9 Выполнить процедуру Send_Message (Next Hop, Label Abort Request, FEC, TLV), где поле TLV содержит Label Request Message ID TLV, содержащее значение Message ID, использованное LSR в ожидающем завершения обработки сообщении Label Request.

LAbR.10 Запись информации о том, что запрос на прерывание метки FEC находится в состоянии ожидания.

LAbR.11 Удаление записи о запросе метки для FEC от MsgSource.

LAbR.12 **Завершено**.

Примечания

1. LSR использует FEC и Label Request Message ID TLV из запроса на прерывание метки для поиска своей записи (при её наличии) о полученном ранее запросе метки от MsgSource.
2. Если LSR получил отображение метки от NextHop, ему следует вести себя, как будто он анонсировал отображение метки MsgSource, а тот освободил эту метку.

A.1.4. Получение Label Release

Описание

Когда LSR получает сообщение Label Release для FEC от своего партнёра, он проверяет, удерживают ли освобождаемую метку другие партнёры. Если таких партнёров нет, LSR прекращает использование метки для коммутации/пересылки и, если LSR удерживает отображение метки от следующего интервала FEC, это отображение освобождается.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- MsgSource - партнёр LDP, передавший сообщение;
- FEC - класс FEC, указанный в сообщении;
- Next Hop - следующий интервал для FEC.

Алгоритм

LRI.1 FEC соответствует известному классу FEC?

Если нет, перейти на LRI.14.

LRI.2 Удалить MsgSource из записи партнёров, удерживающих метку Label для FEC (см. примечание 1).

LRI.3 Сообщение соответствует ожидающему обработки отзыву метки для FEC, переданному ранее MsgSource?

Если нет, перейти на LRI.5.

LRI.4 Удалить запись ожидающего отзыва метки для FEC, ранее переданного MsgSource.

LRI.5 LSR использует слияние меток для этого FEC?

Если нет, перейти на LRI.7 (см. примечание 2).

LRI.6 LSR имеет ожидающие завершения обработки анонсы меток для этого FEC?

Если да, перейти на LRI.11.

LRI.7 LSR является выходом для FEC?

Если да, перейти на LRI.11.

LRI.8 Существует Next Hop для FEC? **и** LSR имеет ранее полученное от Next Hop отображение метки для FEC?

Если нет, перейти на LRI.11.

LRI.9 LSR настроен на распространение освобождения меток?

Если нет, перейти на LRI.11 (см. примечание 3).

LRI.10 Выполнить процедуру Send_Message (Next Hop, Label Release, FEC, Label from Next Hop).

LRI.11 Прекратить использование Label при коммутации/пересылке трафика от MsgSource.

LRI.12 Продолжает ли кто-либо из партнёров удерживать метку Label для FEC?

Если да, перейти на LRI.14.

LRI.13 Освободить метку Label.

LRI.14 **Завершено**.

Примечания

1. Если LSR использует режим распространения меток Downstream Unsolicited, ему **не следует** повторно анонсировать MsgSource отображение метки FEC, пока MsgSource не запросит этого.
2. В LRI.5 - LRI.9 определяется, следует ли LSR распространять Label Release своему нисходящему партнёру (LRI.9).
3. Если достигнут LRI.9, это означает, что ни один из восходящих LSR не удерживает метку для FEC, а сам LSR удерживает метку для FEC от FEC Next Hop. Маршрутизатор LSR может распространять Label Release на Next Hop. За счёт распространения Label Release маршрутизатор LSR освобождает потенциально дефицитный ресурс меток. При этом он также увеличивает задержку повторной организации LSP, когда MsgSource или другой восходящий LSR передаст новое сообщение Label Request для FEC.

   Решение вопроса о распространении отзывов меток не задается протоколом. Распространение меток будет корректно работать в любом случае. При решении вопроса о распространении отзыва следует принимать во внимание такие факторы, как дефицит ресурсов меток, важность сохранения малой задержки при организации LSP, управление организацией LSP в рабочей среде (по входу или по выходу).

A.1.5. Получение Label Withdraw

Описание

Когда LSR получает сообщение Label Withdraw для FEC от партнёра LDP, он отвечает сообщением Label Release и прекращает использование метки для коммутации/пересылки. В режиме упорядоченного управления (Ordered Control) LSR передаёт сообщение Label Withdraw каждому партнёру LDP, которому он ранее передавал отображение метки для FEC. Если LSR использует режим анонсирования меток Downstream on Demand с независимым управлением, он действует, как в случае распознания FEC.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- MsgSource - партнёр LDP, передавший сообщение;
- FEC - класс FEC, указанный в сообщении;
- Next Hop - следующий интервал для FEC.

Алгоритм

LWd.1 Прекратить использование метки для коммутации/пересылки (см. примечание 1).

LWd.2 Выполнить процедуру Send_Message (MsgSource, Label Release, FEC, Label).

LWd.3 У LSR имеется ранее полученное и удерживаемое отображение метки для FEC от MsgSource?

Если нет, перейти на LWd.13.

LWd.4 Удалить соответствующее отображение метки на FEC, полученное ранее от MsgSource.

LWd.5 LSR использует Ordered Control?

Если да, перейти на LWd.8.

LWd.6 MsgSource использует режим анонсирования меток Downstream On Demand?

Если нет, перейти на LWd.13.

LWd.7 Сгенерировать событие: Распознание нового FEC. Перейти на LWd.13 (см. примечание 2).

LWd.8 Выполнить итерацию до LWd.12 для каждого партнёра, отличного от MsgSource.

LWd.9 LSR ранее передавал отображение метки на FEC партнёру Peer?

Если нет, продолжить с LWd.8 для следующего партнёра.

LWd.10 Метка, переданная ранее партнёру Peer, «отображает» отзываемую метку?

Если нет, продолжить с LWd.8 для следующего партнёра (см. примечание 3).

LWd.11 Выполнить процедуру Send_Label_Withdraw (Peer, FEC, ранее переданная партнёру Peer метка Label).

LWd.12 Завершение итерации, начинающейся с LWd.8.

LWd.13 **Завершено**.

Примечания

1. Если метка Label не используется для коммутации/пересылки, LWd.1 не даёт никакого эффекта.
2. LWd.7 используется в случаях, когда LSR работает в режиме распространения меток Downstream On Demand с независимым управлением. В такой ситуации LSR следует передавать запрос метки следующему интервалу для FEC, как для случая распознания FEC.
3. LWd.10 обрабатывает ситуации как при поддержке слияния меток (одна или множество входящих меток отображаются в одну исходящую), так и без таковой (одна метка отображается в одну исходящую метку).

A.1.6. Распознание нового FEC

Описание

Отклик LSR на получение информации о новом FEC из таблицы маршрутизации может включать одно или несколько перечисленных ниже действий:

- передача отображения метки для FEC одному или множеству партнёров LDP;
- передача запроса метки для FEC следующему интервалу FEC;
- любое из действий, выполняемых при получении LSR отображения метки на FEC от следующего интервала FEC.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- Label - метка, указанная в сообщении;
- FEC - класс FEC, указанный в сообщении;
- InitAttributes - атрибуты, связанные с новым классом FEC (см. примечание 1);
- SAttributes - атрибуты, включаемые в сообщения Label Mapping или Label Request, если они передаются партнёрам;
- StoredHopCount - счётчик интервалов, ранее записанных для FEC (при наличии таковых).

Алгоритм

FEC.1 Выполнить процедуру Label Distribution:

Для режима Downstream Unsolicited Independent Control:

1. Выполнить итерации до п. 5 для каждого партнёра;
2. LSR ранее получил и удерживает отображение метки на FEC от Next Hop?

   Если да, установить Propagating = IsPropagating;

   Если нет, установить Propagating = NotPropagating;
3. Выполнить процедуру Prepare_Label_Mapping_Attributes (Peer, FEC, InitAttributes, SAttributes, Propagating, Unknown hop count(0));

4. Выполнить процедуру Send_Label (Peer, FEC, Sattributes);
5. Завершение итерации с п. 1;

   Переход на FEC.2.

Для режима Downstream Unsolicited Ordered Control:

1. Выполнить итерации до п. 5 для каждого партнёра;
2. Является LSR выходом для FEC? **или** LSR ранее получил и удерживает отображение метки на FEC от Next Hop?

   Если нет, продолжить итерации для следующего партнёра;
3. Выполнить процедуру Prepare_Label_Mapping_Attributes (Peer, FEC, InitAttributes, SAttributes, Propagating, StoredHopCount).
4. Выполнить процедуру Send_Label (Peer, FEC, SAttributes).
5. Завершение итерации с п. 1;

   Переход на FEC.2.

Для режима Downstream On Demand Independent Control **или** For Downstream On Demand Ordered Control:

1. Переход на FEC.2 (см. примечание 2).

FEC.2 LSR ранее получил и удерживает отображение метки на FEC от Next Hop?

Если да, перейти на FEC.5.

FEC.3 Next Hop является партнёром LDP?

Если нет, перейти на FEC.6.

FEC.4 Выполнить процедуру запроса метки Label Request:

для Request Never:

1. Переход на FEC.6;

для Request When Needed **или** For Request On Request:

1. Выполнить процедуру Prepare_Label_Request_Attributes (Next Hop, FEC, InitAttributes, SAttributes);
2. Выполнить процедуру Send_Label_Request (Next Hop, FEC, Sattributes);

   Переход на FEC.6.

FEC.5 Сгенерировать событие: Получено отображение метки от Next Hop (см. примечание 3).

FEC.6 **Завершено**.

Примечания

1. Примером атрибута, который может быть частью InitAttributes, является параметр, задающий желаемый характеристики LSP - например, класс обслуживания CoS[1]. (отметим, что текущая версия протокола LDP не задаёт атрибута CoS, на расширения LDP могут делать это).

   Способы задания InitAttributes для FEC (если эти атрибуты используются) выходят за пределы спецификации LDP. Отметим, что InitAttributes не будет включать известные атрибуты Hop Count или Path Vector.
2. LSR, использующий режим распространения меток Downstream On Demand, будет передавать метку только в том случае, если у него имеется ранее полученный запрос метки, помеченный, как ожидающий. У LSR в этом режиме не будет ожидающих запросов, поскольку он отвечает на все запросы меток для неизвестных FEC передачей запрашивающему LSR уведомления No Route и отбрасывает запрос (см. LRq.3).
3. Если у LSR имеется метка для FEC от Next Hop, ему следует вести себя, как при получении метки от Next Hop. Такая ситуация возникает для режима либерального удержания меток.

A.1.7. Детектирование смены FEC Next Hop

Описание

Отклик LSR на изменение следующего интервала для FEC может включать одно или несколько перечисленных ниже действий:

- прекращение использования метки старого FEC для коммутации/пересылки;
- передача сообщений Label Mapping для FEC одному или множеству партнёров LDP;
- передача запроса метки новому FEC next hop;
- любые действия, которые могут выполняться при получении LSR отображения метки от нового FEC next hop.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- FEC - класс FEC, указанный в сообщении;

[1]Class of Service.

- New Next Hop - новый следующий интервал для FEC;
- Old Next Hop - прежний следующий интервал для FEC;
- OldLabel - метка, полученная ранее от Old Next Hop (при наличии таковой);
- CurAttributes - атрибуты, связанные с FEC (если таковые имеются);
- Sattributes - атрибуты для включения в сообщение Label Request, передаваемое New Next Hop (при наличии таких атрибутов).

Алгоритм

NH.1 LSR ранее получил и удерживает отображение метки для FEC от Old Next Hop?

Если нет, перейти на NH.6.

NH.2 Прекратить использование метки для коммутации/пересылки (см. примечание 1).

NH.3 LSR использует либеральное удержание меток (Liberal Label retention)?

Если да, перейти на NH.6.

NH.4 Выполнить процедуру Send_Message (Old Next Hop, Label Release, OldLabel).

NH.5 Удалить отображение метки для FEC, полученное ранее от Old Next Hop.

NH.6 LSR имеет ожидающий завершения обработки запрос метки от Old Next Hop?

Если нет, перейти на NH.10.

NH.7 LSR использует консервативное удержание меток (Conservative Label retention)?

Если нет, перейти на NH.10.

NH.8 Выполнить процедуру Send_Message (Old Next Hop, Label Abort Request, FEC, TLV), где TLV содержит значение Label Request Message ID TLV с идентификатором ожидающего запроса метки.

NH.9 Записать информацию об ожидании прерывания запроса метки для FEC от Old Next Hop.

NH.10 Имеется New Next Hop для FEC?

Если нет, перейти на NH.16.

NH.11 LSR ранее получил и удерживает отображение метки для FEC от New Next Hop?

Если нет, перейти на NH.13.

NH.12 Сгенерировать событие: Получение Label Mapping от New Next Hop.

Перейти на NH.20 (см. примечание 2).

NH.13 LSR использует режим анонсирования Downstream on Demand advertisement? **или** Next Hop использует режим анонсирования Downstream on Demand advertisement? **или** LSR использует режим Conservative Label retention? (см. примечание 3).

Если да, перейти на NH.14.

Если нет, перейти на NH.20.

NH.14 Выполнить процедуру Prepare_Label_Request_Attributes (Next Hop, FEC, CurAttributes, SAttributes).

NH.15 Выполнить процедуру Send_Label_Request (New Next Hop, FEC, SAttributes). (See Note 4.)

Перейти на NH.20.

NH.16 Итерация до NH.19 для каждого партнёра Peer.

NH.17 LSR ранее передавал отображение метки на FEC партнёру Peer?

Если нет, продолжить итерацию с NH.16 для следующего партнёра.

NH.18 Выполнить процедуру Send_Label_Withdraw (Peer, FEC, переданная ранее партнёру метка Label).

NH.19 Завершение итерации с NH.16.

NH.20 **Завершено**.

Примечания

1. Если метка Label не используется для коммутации/пересылки, NH.2 не даёт эффекта.
2. Если у LSR имеется метка для FEC от New Next Hop, ему следует вести себя, как при получении метки от New Next Hop.
3. Цель проверки режима удержания меток заключается в том, чтобы избежать «гонки» на этапах LMp.12-LMp.13 при обработке сообщения Label Mapping, когда LSR, работающий в режиме консервативного удержания меток, может освободить отображение метки, полученное от New Next Hop до того, как обнаружит изменение следующего интервала для FEC.
4. Независимо от процедуры Label Request, используемой LSR, маршрутизатор **должен** отправить запрос метки при выполнении условия NH.13. Следовательно, он будет напрямую выполнять процедуру Send_Label_Request вместо LSR Label Request.

A.1.8. Получение сообщения Notification/Label Request Aborted

Описание

Когда LSR получает уведомление Label Request Aborted от партнёра LDP, он записывает, что соответствующая транзакция (если она есть) была завершена.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- FEC - класс FEC, указанный в сообщении;
- RequestMessageID - идентификатор сообщения из запроса метки, обработка которого прерывается.
- MsgSource - партнёр LDP, передавший сообщение Notification.

Алгоритм

LRqA.1 Уведомление соответствует запросу на прерывание метки для FEC? (см. приложение 1).

Если нет, перейти на LRqA.3.

LRqA.2 Записать факт прерывания запроса метки для FEC.

LRqA.3 **Завершено**.

Примечание

1. LSR использует FEC и RequestMessageID для поиска записи о запросе на прерывание метки (если таковой существует).

A.1.9. Получение сообщения Notification/No Label Resources

Описание

Когда LSR получает уведомление No Label Resources от партнёра LDP, он прекращает отправку сообщений с запросом меток до тех пор, пока не получит от этого партнёра уведомление Label Resources Available.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- FEC - класс FEC, указанный в сообщении;
- MsgSource - партнёр LDP, передавший сообщение Notification.

Алгоритм

NoRes.1 Удалить запись об остающемся запросе метки для FEC, переданном MsgSource.

NoRes.2 Сделать запись о том, что отображение метки для FEC от MsgSource требуется, но ресурсов для выделения меток не хватает.

NoRes.3 Установить запись состояния, показывающую, что не следует передавать запросы меток MsgSource.

NoRes.4 **Завершено**.

A.1.10. Получение сообщения Notification/No Route

Описание

Когда LSR получает уведомление No Route от своего партнёра LDP в ответ на сообщение Label Request, его отклик диктуется процедурой Label No Route. Маршрутизатор LSR не будет предпринимать дальнейших действий или отложит запрос метки, запустив таймер, по истечение которого передаст новое сообщение Label Request.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- FEC - класс FEC, указанный в сообщении;
- Attributes - атрибуты, связанные с запросом метки;
- MsgSource - партнёр LDP, передавший сообщение Notification.

Алгоритм

NoNH.1 Удалить запись остающегося запроса метки FEC, переданного MsgSource.

NoNH.2 Выполнить процедуру LSR Label No Route.

для Request No Retry:

1. Переход на NoNH.3.

для Request Retry:

1. Записать отложенный запрос метки для FEC и атрибуты для передачи MsgSource.
2. Запустить таймер.

Переход на NoNH.3.

NoNH.3 **Завершено**.

A.1.11. Получение сообщения Notification/Loop Detected

Описание

Когда LSR получает код состояния Loop Detected от партера LDP в ответ на сообщение Label Request или Label Mapping, он ведёт себя, как при получении уведомления No Route.

Контекст

См. A.1.10. Получение сообщения Notification/No Route.

Алгоритм

См. A.1.10. Получение сообщения Notification/No Route.

Примечание

1. Когда уведомление Loop Detected приходит в ответ на сообщение Label Request, оно содержится в поле Status Code TLV сообщения Notification. В случае отклика на сообщение Label Mapping код состояния приходит в Status Code TLV сообщения Label Release.

A.1.12. Получение сообщения Notification/Label Resources Available

Описание

Когда LSR получает уведомление Label Resources Available от партнёра LDP, он возобновляет запросы меток у партнёра.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- MsgSource - партнёр LDP, передавший сообщение Notification;
- SAttributes - атрибуты, сохранённые с отложенным сообщением Label Request.

Алгоритм

Res.1 Установить запись состояния, показывающую возможность запроса меток у MsgSource.

Res.2 Итерация до Res.6 для каждой записи отображения метки на FEC, требуемого от MsgSource, для которой не хватало ресурсов.

Res.3 MsgSource является следующим интервалом для FEC?

Если нет, переход на Res.5.

Res.4 Выполнить процедуру Send_Label_Request (MsgSource, FEC, Sattributes); при отказе итерация прерывается.

Res.5 Удалить запись о недоступности ресурсов для отображения метки на FEC, требуемого от MsgSource.

Res.6 Завершение итерации, начинающейся с Res.2.

Res.7 **Завершено**.

A.1.13. Обнаружение доступности локальных ресурсов для меток

Описание

После передачи маршрутизатором LSR уведомления No Label Resources партнёру LDP и последующего возобновления доступности ресурсов LSR передаёт партнёру уведомление Label Resources Available.

Контекст

- LSR - маршрутизатор, обрабатывающий событие;
- Attributes - атрибуты, связанные с запросом метки.

Алгоритм

ResA.1 Итерация до ResA.4 для каждого партнёра, которому LSR ранее передал уведомление No Label Resources.

ResA.2 Выполнить процедуру Send_Notification (Peer, Label Resources Available).

ResA.3 Удалить запись о передаче партнёру уведомления No Label Resources.

ResA.4 Завершение итерации, начинающейся с ResA.1.

ResA.5 Итерация до ResA.8 для каждой записи об отображении метки на FEC от партнёра Peer, которая нужна, но недоступна по причине нехватки ресурсов (см. примечание 1).

ResA.6 Выполнить процедуру Send_Label (Peer, FEC, Attributes); при отказе итерация прерывается.

ResA.7 Очистить запись о требуемом, но недоступном отображении метки на FEC.

ResA.8 Завершение итерации, начинающейся с ResA.5

ResA.9 **Завершено**.

Примечание

1. Итерация ResA.5 - ResA.8 служит для обработки ситуаций, когда LSR использует режим анонсирования Downstream Unsolicited и ранее не имел возможности выделить метку для FEC.

A.1.14. Принятие LSR решения о прекращении коммутации по меткам для FEC

Описание

LSR может в одностороннем порядке принять решение от отказе от коммутации по меткам для FEC данного партнёра LDP. Для LSR **недопустима** передача партнёру сообщения Label Withdraw для FEC.

Контекст

- Peer - партнёр;
- FEC - класс FEC;
- PrevAdvLabel - метка для FEC, анонсированная ранее партнёру Peer.

Алгоритм

NoLS.1 Выполнить процедуру Send_Label_Withdraw (Peer, FEC, PrevAdvLabel) (см. примечание 1).

NoLS.2 **Завершено**.

Примечание

1. LSR может прекратить использование метки для коммутации/пересылки в рамках этого события или при обработке сообщения Label Release от партнёра в ответ на отзыв метки. Если LSR не ждёт сообщения Label Release от партнёра, ему **не следует** повторно использовать метку, пока не будет получено сообщение Label Release.

A.1.15. Тайм-аут для отложенного запроса метки

Описание

Запросы меток откладываются в ответ на уведомления No Route и Loop Detected. Когда завершается отсчет таймера для отложенного запроса метки, LSR запрашивает метку повторно.

Контекст

- LSR - маршрутизатор LSR, обрабатывающий событие;
- FEC - класс FEC, с которым связан тайм-аут;
- Peer - партнёр LDP, с которым связан тайм-аут;
- Attributes - атрибуты, сохранённые с отложенным сообщением Label Request.

Алгоритм

TO.1 Найти запись для отложенного запроса метки.

TO.2 Peer является следующим интервалом для FEC?

Если нет, переход на TO.4.

TO.3 Выполнить процедуру Send_Label_Request (Peer, FEC).

TO.4 **Завершено**.

A.2. Общие процедуры распространения меток

В этом приложении описаны базовые процедуры, используемые алгоритмами обработки событий при распространении меток.

A.2.1. Send_Label

Описание

Процедура Send_Label обеспечивает выделение метки FEC для партнёра LDP, если это возможно, и передаёт этому партнёру отображение метки на FEC. Если LSR не может выделить метку и имеет от партнёра ожидающий запрос метки, он передаёт партнёру LDP уведомление No Label Resources.

Параметры

- Peer - партнёр LDP, для которого передаётся отображение метки;
- FEC - класс FEC, для которого передаётся отображение метки;
- Attributes - атрибуты для включения в отображение метки.

Дополнительный контекст

- LSR - маршрутизатор LSR, выполняющий процедуру;
- Label - метка, выделяемая и передаваемая партнёру.

Алгоритм

SL.1 LSR выделил метку?

Если нет, переход на SL.9.

SL.2 Выделить метку Label и связать её с классом FEC.

SL.3 Установить использование метки Label для коммутации/пересылки.

SL.4 Выполнить процедуру Send_Message (Peer, Label Mapping, FEC, Label, Attributes).

SL.5 Записать отображение метки на FEC со значениями Label и Attributes, переданными партнёру Peer.

SL.6 LSR имеет запись о запросе метки для FEC от партнёра Peer, помеченном, как ожидающий?

Если нет, переход на SL.8.

SL.7 Удалить запись об ожидающем запросе метки для FEC от партнёра Peer.

SL.8 Вернуть код успешного завершения.

SL.9 LSR имеет запрос метки для FEC от партнёра Peer, помеченный, как ожидающий?

Если нет, переход на SL.13.

SL.10 Выполнить процедуру Send_Notification (Peer, No Label Resources).

SL.11 Удалить запись об ожидающем запросе метки для FEC от партнёра Peer.

SL.12 Записать факт передачи уведомления No Label Resources партнёру Peer.

Перейти на SL.14.

SL.13 Записать отображение метки, требуемое для FEC, и значение Attributes для партнёра Peer, запрос которого не был выполнен по причине нехватки ресурса меток (см. примечание 1).

SL.14 Вернуть код отказа.

Примечание

1. SL.13 обслуживает ситуации для режима анонсирования Downstream Unsolicited, когда LSR не может выделить метку для FEC, чтобы отправить её партнёру Peer.

A.2.2. Send_Label_Request

Описание

LSR использует процедуру Send_Label_Request для передачи запроса метки для FEC партнёру LDP, если это разрешено.

Параметры

- Peer - партнёр LDP, которому передаётся запрос метки;
- FEC - класс FEC, для которого запрашивается метка;
- Attributes - атрибуты для включения в запрос метки (например, Hop Count, Path Vector).

Дополнительный контекст

- LSR - маршрутизатор LSR, выполняющий процедуру.

Алгоритм

SLRq.1 Запрос метки для FEC был ранее передан партнёру Peer и помечен, как ожидающий?

Если да, вернуть код успешного завершения (см. примечание 1).

SLRq.2 Имеется запись состояния, показывающая возможность запроса метки у партнёра Peer?

Если нет, переход на SLRq.6

SLRq.3 Выполнить процедуру Send_Message (Peer, Label Request, FEC, Attributes).

SLRq.4 Записать факт передачи запроса метки для FEC партнёру Peer и пометки его, как ожидающего.

SLRq.5 Вернуть код успешного завершения.

SLRq.6 Отложить запрос метки, записав, что отображение метки на FEC и значение Attributes от партнёра Peer требуются, но нет ресурсов для меток.

SLRq.7 Вернуть код отказа.

Примечание

1. Если LSR не поддерживает слияния, он должен отличать попытки передачи запросов меток для FEC, инициируемые разными восходящими партнёрами LDP, от дубликатов запросов. Данная процедура не передаёт дубликатов запроса.

A.2.3. Send_Label_Withdraw

Описание

LSR использует процедуру Send_Label_Withdraw для отзыва метки FEC у партнёра LDP. Для этого LSR передаёт партнёру сообщение Label Withdraw.

Параметры

- Peer - партнёр LDP, у которого отзывается метка;
- FEC - класс FEC, для которого отзывается метка;
- Label - отзываемая метка.

Дополнительный контекст

- LSR - маршрутизатор LSR, выполняющий процедуру.

Алгоритм

SWd.1 Выполнить процедуру Send_Message (Peer, Label Withdraw, FEC, Label).

SWd.2 Записать факт передачи отзыва метки для FEC партнёру Peer и пометить его, как ожидающий.

A.2.4. Send_Notification

Описание

LSR использует процедуру Send_Notification для передачи партнёру LDP сообщения Notification.

Параметры

- Peer - партнёр LDP, которому передаётся сообщение Notification;
- Status - код состояния для включения в сообщение Notification.

Дополнительный контекст

Нет.

Алгоритм

SNt.1 Выполнить процедуру Send_Message (Peer, Notification, Status).

A.2.5. Send_Message

Описание

LSR использует процедуру Send_Message для передачи партнёру LDP сообщений LDP.

Параметры

- Peer - партнёр LDP, которому передаётся сообщение;
- Message Type - тип передаваемого сообщения;
- Дополнительные поля сообщения.

Дополнительный контекст

Нет.

Алгоритм

Эта процедура задаёт способ передачи маршрутизатором LSR сообщения LDP заданного типа указанному партнёру LDP.

A.2.6. Check_Received_Attributes

Описание

Проверка атрибутов, полученных в сообщении Label Mapping или Label Request. Если в число атрибутов входит Hop Count или Path Vector, выполняется проверка наличия петель (Loop Detection). При обнаружении петли передаётся уведомление Loop Detected по адресу MsgSource.

Параметры

- MsgSource - партнёр LDP, который передал сообщение;
- MsgType - тип полученного сообщения;
- RAttributes - атрибуты из сообщения.

Дополнительный контекст

- LSR Id - уникальное значение LSR Id маршрутизатора LSR;
- Hop Count - значение поля Hop Count в полученных атрибутах (при его наличии);
- Path Vector - значение поля Path Vector в полученных атрибутах (при его наличии).

Алгоритм

CRa.1 Значение RAttributes включает Hop Count?

Если нет, переход на CRa.5.

CRa.2 Значение Hop Count превосходит максимально допустимое?

Если да, переход на CRa.6.

CRa.3 Значение RAttributes включает Path Vector?

Если нет, переход на CRa.5.

CRa.4 Path Vector включает LSR Id? **или** Размер Path Vector превосходит допустимый?

Если да, переход на CRa.6

CRa.5 Возврат кода No Loop Detected.

CRa.6 Значение MsgType равно LabelMapping?

Если да, переход на CRa.8. (See Note 1.)

CRa.7 Выполнить процедуру Send_Notification (MsgSource, Loop Detected).

CRa.8 Возврат кода Loop Detected.

CRa.9 **Завершено**.

Примечание

1. Когда проверяемые атрибуты были получены в сообщении Label Mapping, LSR передаёт уведомление Loop Detected в поле Status Code TLV сообщения Label Release (см. приложение A.1.2. Получение Label Mapping).

A.2.7. Prepare_Label_Request_Attributes

Описание

Эта процедура используется в тех случаях, когда партнёру передаётся сообщение Label Request, для расчёта значений Hop Count и Path Vector, включаемых в сообщение (при их наличии).

Параметры

- Peer - партнёр LDP, которому передаётся сообщение;
- FEC - класс FEC, для которого запрашивается метка;
- RAttributes - атрибуты, которые данный LSR связывает с LSP для FEC;
- Sattributes - атрибуты для включения в сообщение Label Request.

Дополнительный контекст

- LSR Id - уникальное значение LSR Id маршрутизатора LSR.

Алгоритм

PRqA.1 Значение Hop Count требуется для партнёра Peer? (см. примечание 1) **или** Значение RAttributes включает Hop Count? **или** Детектирование петель (Loop Detection) включено на LSR?

Если нет, переход на PRqA.14.

PRqA.2 LSR является входом для FEC?

Если нет, переход на PRqA.6.

PRqA.3 Включить Hop Count = 1 в SAttributes.

PRqA.4 Механизм Loop Detection включён на LSR?

Если нет, переход на PRqA.14.

PRqA.5 LSR поддерживает слияние?

Если да, переход на PRqA.14.

Если нет, переход на PRqA.13.

PRqA.6 RAttributes включает Hop Count?

Если нет, переход на PRqA.8.

PRqA.7 Увеличить в RAttributes значение Hop Count на 1 и скопировать результат в SAttributes (см. примечание 2).

Перейти на PRqA.9.

PRqA.8 Включить Hop Count = 0 (unknown[1]) в SAttributes.

PRqA.9 Механизм Loop Detection включён на LSR?

Если нет, переход на PRqA.14.

PRqA.10 RAttributes включает Path Vector?

Если да, переход на PRqA.12.

PRqA.11 LSR поддерживает слияние?

Если да, переход на PRqA.14.

Если нет, переход на PRqA.13.

[1]Неизвестно.

PRqA.12 Добавить LSR Id в начало Path Vector из RAttributes и скопировать результат в Path Vector поля SAttributes.

Перейти на PRqA.14.

PRqA.13 Включить Path Vector размера 1, содержащий LSR Id в поле SAttributes.

PRqA.14 **Завершено**.

Примечания

1. Канал к партнёру Peer может требовать включения поля Hop Count в сообщения Label Request (для примера см. [RFC3035] и [RFC3034]).
2. При подсчёте числа интервалов используется арифметика unknown + 1 = unknown.

A.2.8. Prepare_Label_Mapping_Attributes

Описание

Эта процедура используется при подготовке сообщений Label Mapping для расчёта значений Hop Count и Path Vector, если они включаются в сообщение.

Параметры

- Peer - партнёр LDP, которому передаётся сообщение;
- FEC - класс FEC, для которого запрашивается метка;
- RAttributes - атрибуты, которые данный LSR связывает с LSP для FEC;
- Sattributes - атрибуты для включения в сообщение Label Request;
- IsPropagating - LSR передаёт сообщение Label Mapping с целью распространение идентичного сообщения от следующего интервала FEC.
- PrevHopCount - значение Hop Count (при наличии такового), которое данный LSR связывает с LSP для FEC.

Дополнительный контекст

- LSR Id - уникальное значение LSR Id маршрутизатора LSR.

Алгоритм

PMpA.1 RAttributes включает неизвестные TLV?

Если нет, переход на PMpA.4.

PMpA.2 Требуется установка битов U и F до пересылки этих TLV?

Если нет, переход на PMpA.4.

PMpA.3 Скопировать неизвестные TLV в SAttributes.

PMpA.4 Значение Hop Count требуется для этого партнёра? (см. примечание 1) **или** RAttributes включает Hop Count? **или** Режим Loop Detection включён на LSR?

Если нет, переход на PMpA.24.

PMpA.5 LSR является выходом для FEC?

Если нет, переход на PMpA.7.

PMpA.6 Включить Hop Count = 1 в SAttributes. Переход на PMpA.24.

PMpA.7 RAttributes содержит Hop Count?

Если нет, переход на PMpA.11.

PMpA.8 LSR входит в краевой набор домена LSR, члены которого на уменьшают значение TTL? **и** Peer входит в этот домен? (см. примечание 2).

Если нет, переход на PMpA.10.

PMpA.9 Включить Hop Count = 1 в SAttributes. Переход на PMpA.12.

PMpA.10 Увеличить значение Hop Count в RAttributes и скопировать результат в SAttributes (см. примечание 2).

Переход на PMpA.12.

PMpA.11 Включить Hop Count = 0 (unknown) в SAttributes.

PMpA.12 Режим Loop Detection включён на LSR?

Если нет, переход на PMpA.24.

PMpA.13 RAttributes включает Path Vector?

Если да, переход на PMpA.22.

PMpA.14 LSR распространяет полученные сообщения Label Mapping?

Если нет, переход на PMpA.23.

PMpA.15 LSR поддерживает слияние?

Если нет, переход на PMpA.17.

PMpA.16 LSR ранее передавал сообщение Label Mapping для FEC партнёру Peer?

Если нет, переход на PMpA.23.

PMpA.17 RAttributes включает Hop Count?

Если нет, переход на PMpA.24.

PMpA.18 Значение Hop Count в RAttributes =0 (unknown)?

Если да, переход на PMpA.23.

PMpA.19 LSR ранее передавал сообщение Label Mapping для FEC партнёру Peer?

Если нет, переход на PMpA.24.

PMpA.20 Значение Hop Count в RAttributes отличается от PrevHopCount?

Если нет, переход на PMpA.24.

PMpA.21 Значение Hop Count в RAttributes больше PrevHopCount? **или** PrevHopCount = 0 (unknown)?

Если нет, переход на PMpA.24.

PMpA.22 Добавить LSR Id в начало Path Vector из RAttributes и скопировать результат в SAttributes.

Перейти на PMpA.24.

PMpA.23 Включить Path Vector размером 1, содержащий значение LSR Id, d SAttributes.

PMpA.24 **Завершено**.

Примечания

1. Канал к партнёру Peer может требовать включения поля Hop Count в сообщения Label Request (для примера см. [RFC3035] и [RFC3034]).
2. Если LSR расположен на краю облака LSR, не уменьшающих значение TTL, и распространяет сообщение Label Mapping в восходящем направлении в это облако, он устанавливает Hop Count = 1 и значение Hop Count для прохождения через облако рассчитывается корректно. Это обеспечивает корректное управление значениями TTL для пакетов, которые пересылаются через часть LSP, проходящих сквозь облако.
3. При подсчёте числа интервалов используется арифметика unknown + 1 = unknown.

Адреса редакторов

Loa Andersson
Acreo AB
Isafjordsgatan 22
Kista, Sweden
EMail: loa.andersson@acreo.se
loa@pi.se

Ina Minei
Juniper Networks
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
EMail: ina@juniper.net

Bob Thomas
Cisco Systems, Inc.
1414 Massachusetts Ave
Boxborough, MA 01719
EMail: rhthomas@cisco.com

Перевод на русский язык

Николай Малых

nmalykh@gmail.com

Полное заявление авторских прав

Интеллектуальная собственность

IETF не принимает какой-либо позиции в отношении действительности или объема каких-либо прав интеллектуальной собственности (Intellectual Property Rights или IPR) или иных прав, которые, как может быть заявлено, относятся к реализации или использованию описанной в этом документе технологии, или степени, в которой любая лицензия, по которой права могут или не могут быть доступны, не заявляется также применение каких-либо усилий для определения таких прав. Сведения о процедурах IETF в отношении прав в документах RFC можно найти в BCP 78 и BCP 79.

Копии раскрытия IPR, предоставленные секретариату IETF, и любые гарантии доступности лицензий, а также результаты попыток получить общую лицензию или право на использование таких прав собственности разработчиками или пользователями этой спецификации, можно получить из сетевого репозитория IETF IPR по ссылке http://www.ietf.org/ipr.

IETF предлагает любой заинтересованной стороне обратить внимание на авторские права, патенты или использование патентов, а также иные права собственности, которые могут потребоваться для реализации этого стандарта. Информацию следует направлять в IETF по адресу ietf-ipr@ietf.org.