

Протокол управления каналом (LMP)

Link Management Protocol (LMP)

Статус документа

В этом документе содержится проект стандарта для протокола Internet, предложенного сообществу Internet. Документ служит приглашением к дискуссии в целях развития и совершенствования протокола. Текущее состояние стандартизации протокола вы можете узнать из документа Internet Official Protocol Standards (STD 1). Документ может распространяться без ограничений.

Авторские права

Copyright (C) The Internet Society (2005).

Тезисы

В целях эффективного масштабирования множество каналов передачи данных может объединяться в один канал организации трафика (TE¹). Кроме того, управление каналами TE не ограничивается передачей сообщений в основной полосе и может осуществляться с помощью специальных каналов управления. Этот документ содержит спецификацию протокола управления каналом (LMP²), который поддерживается между парами узлов и служит для управления каналами TE. В частности, LMP будет применяться для поддержки связности каналов управления, проверки физической связности каналов данных, корреляции данных о свойствах каналов, подавления сигналов в нисходящем направлении и локализации отказов на каналах для защиты и восстановления в множестве разнотипных сетей.

Оглавление

1. Введение.....	2
1.1. Терминология.....	3
2. Обзор LMP.....	4
3. Поддержка канала управления.....	5
3.1. Согласование параметров.....	5
3.2. Протокол Hello.....	5
3.2.1. Согласование параметров Hello.....	6
3.2.2. Ускоренная проверка жизнеспособности.....	6
3.2.3. Отключение канала управления.....	6
3.2.4. Вырожденное состояние.....	6
4. Сопоставление свойств каналов.....	7
5. Проверка связности для канала.....	7
5.1. Пример проверки связности для канала.....	8
6. Обработка отказов.....	9
6.1. Детектирование отказов.....	9
6.2. Процедура локализации отказа.....	10
6.3. Примеры локализации отказов.....	10
6.4. Индикация активации канала.....	10
6.5. Индикация деактивации канала.....	11
7. Использование идентификаторов сообщений.....	11
8. Аккуратный перезапуск.....	11
9. Адресация.....	12
10. Процедуры экспоненциального роста интервалов повтора.....	12
10.1. Работа.....	12
10.2. Алгоритм повтора передачи.....	12
11. Машины конечных состояний LMP.....	13
11.1. FSM управляющего канала.....	13
11.1.1. Состояния управляющего канала.....	13
11.1.2. События канала управления.....	13
11.1.3. Описание FSM канала управления.....	14
11.2. FSM канала TE.....	14
11.2.1. Состояния канала TE.....	14
11.2.2. События каналов TE.....	15
11.2.3. Описание машины конечных состояний канала TE.....	15
11.3. FSM канала данных.....	15
11.3.1. Состояния канала данных.....	15
11.3.2. События канала данных.....	16
11.3.3. Описание FSM активного канала данных.....	16

¹Traffic engineering — организация (построения) трафика.

²Link management protocol — протокол управления каналом.

11.3.4. Описание FSM пассивного канала данных.....	16
12. Форматы сообщений LMP.....	17
12.1. Общий заголовок.....	17
12.2. Формат объекта LMP.....	18
12.3. Сообщения при согласовании параметров.....	18
12.3.1. Сообщение Config (Msg Type = 1).....	18
12.3.2. Сообщение ConfigAck (Msg Type = 2).....	18
12.3.3. Сообщение ConfigNack (Msg Type = 3).....	18
12.4. Сообщение Hello (Msg Type = 4).....	19
12.5. Сообщения для проверки канала.....	19
12.5.1. Сообщение BeginVerify (Msg Type = 5).....	19
12.5.2. Сообщение BeginVerifyAck (Msg Type = 6).....	19
12.5.3. Сообщение BeginVerifyNack (Msg Type = 7).....	19
12.5.4. Сообщение EndVerify (Msg Type = 8).....	20
12.5.5. Сообщение EndVerifyAck (Msg Type = 9).....	20
12.5.6. Сообщение Test (Msg Type = 10).....	20
12.5.7. Сообщение TestStatusSuccess (Msg Type = 11).....	20
12.5.8. Сообщение TestStatusFailure (Msg Type = 12).....	20
12.5.9. Сообщение TestStatusAck (Msg Type = 13).....	20
12.6. Сообщения о состоянии канала.....	21
12.6.1. Сообщение LinkSummary (Msg Type = 14).....	21
12.6.2. Сообщение LinkSummaryAck (Msg Type = 15).....	21
12.6.3. Сообщение LinkSummaryNack (Msg Type = 16).....	21
12.7. Сообщения контроля отказов.....	21
12.7.1. Сообщение ChannelStatus (Msg Type = 17).....	21
12.7.2. Сообщение ChannelStatusAck (Msg Type = 18).....	21
12.7.3. Сообщение ChannelStatusRequest (Msg Type = 19).....	21
12.7.4. Сообщение ChannelStatusResponse (Msg Type = 20).....	22
13. Определения объектов LMP.....	22
13.1. Класс CCID (Control Channel ID).....	22
13.2. Класс NODE_ID.....	22
13.3. Класс LINK_ID.....	22
13.4. Класс INTERFACE_ID.....	23
13.5. Класс MESSAGE_ID.....	23
13.6. Класс CONFIG.....	24
13.7. Класс HELLO.....	24
13.8. Класс BEGIN_VERIFY.....	24
13.9. Класс BEGIN_VERIFY_ACK.....	25
13.10. Класс VERIFY_ID.....	25
13.11. Класс TE_LINK.....	25
13.12. Класс DATA_LINK.....	26
13.12.1. Субобъекты канала данных.....	27
13.12.1.1. Субобъект типа 1 — тип коммутации интерфейса.....	28
13.12.1.2. Субобъект типа 2 — длина волны.....	28
13.13. Класс CHANNEL_STATUS.....	28
13.14. Класс CHANNEL_STATUS_REQUEST.....	29
13.15. Класс ERROR_CODE.....	30
14. Литература.....	31
14.1. Нормативные документы.....	31
14.2. Дополнительная литература.....	31
15. Вопросы безопасности.....	31
15.1. Требования безопасности.....	32
15.2. Механизмы защиты.....	32
16. Взаимодействие с IANA.....	33
17. Благодарности.....	36
18. Участники работы.....	36

1. Введение

Разрабатывают сети с маршрутизаторами, коммутаторами, кросс-коннекторами, системами DWDM¹ и мультиплексорами ADM², использующими общий уровень управления (common control plane), например, GMPLS³ для динамического распределения ресурсов и обеспечения жизнестойкости сети с использованием средств защиты и восстановления. Пара узлов может иметь тысячи соединений между собой и каждое такое соединение может включать множество каналов передачи данных с использованием мультиплексирования (например, Frame Relay DLCI на уровне 2 или временные интервалы TDM⁴, длины волн WDM⁵ на уровне 1). В целях масштабирования множество каналов данных может объединяться в один канал TE.

Для обеспечения коммуникаций между узлами для маршрутизации, сигнализации и управления каналами требуется пара доступных один для другого интерфейсов IP. Такую пару интерфейсов будем называть здесь «каналом управления». Отметим, что взаимная доступность интерфейсов не предполагает наличия между ними непосредственного соединения IP — между узлами может находиться сеть IP. Более того, интерфейс, через который передаются и принимаются управляющие сообщения может не совпадать с интерфейсом, используемым для передачи потока данных. Этот документ задает спецификацию протокола управления каналом (LMP), работающего

¹Dense wavelength division multiplexed — мультиплексирование по длине волны с высокой плотностью.

²Add-drop multiplexor — мультиплексор с ответвлением.

³Generalized MPLS — обобщенная коммутация по меткам.

⁴Time division multiplexed — мультиплексирование с разделением по времени.

⁵Wavelength division multiplexed — мультиплексирование с разделением по длине волны.

между парами узлов и используемого для поддержки соединений TE и проверки доступности канала управления. Далее в документе узлы, участвующие в работе протокола, называются соседями LMP или просто соседними узлами.

В GMPLS управляющие каналы между двумя смежными узлами уже не обязаны использовать ту же физическую среду, которая служит для передачи данных между этими узлами. Например, канал управления может использовать отдельное виртуальное устройство, длину волны, волокно, соединение Ethernet, туннель IP через отдельную сеть управления или сеть IP с множеством интервалов пересылки. Следствием возможности организации каналов управления между парой узлов отдельно (логически или физически) от канала передачи данных между этими узлами является то, что жизнестойкость канала управления не связано с жизнестойкостью каналов передачи данных и наоборот. Следовательно, нужно четко различать канал управления и каналы передачи данных. Для управления каналами данных требуется разработать новые механизмы как в плане обеспечения связности, так и для контроля отказов.

К выполняемым протоколом LMP задачам относится проверка группировки каналов в соединения TE, а также проверка совпадения свойств этих каналов на обеих сторонах соединения — это называется корреляцией свойств каналов (link property correlation). Кроме того, LMP может обмениваться этими свойствами с модулем IGP, который, в свою очередь, может анонсировать их другим узлам сети. LMP может также сообщать модулю сигнализации отображения между соединениями TE и каналами управления. Таким образом, LMP исполняет функции «склеивания» на уровне управления.

Следует отметить, что наличие сети управления (одно- или многоинтервальной) требуется для возможности коммуникаций, но не является достаточным для этого. Например, если два интерфейса разделены сетью IP, отказ в этой сети может привести к разрыву пути от одного интерфейса к другому и, следовательно, прерыванию коммуникаций между этими интерфейсами. С другой стороны, не каждый отказ в сети управления воздействует на данный канал управления, следовательно, нужна организация и поддержка каналов управления.

Для целей этого документа канал данных может рассматриваться каждым узлом, как завершающийся на порту или компонентном соединении, в зависимости от возможностей мультиплексирования конечной точки данного канала - компонентное соединение поддерживает мультиплексирование, порт не поддерживает. Это различие важно, поскольку управление такими каналами (включая, например, выделение ресурсов, присвоение меток и из физическую верификацию) зависит от возможностей мультиплексирования. Например, коммутатор Frame Relay может демультимплексировать интерфейс по виртуальным устройствам на основе значений DLCI, кросс-коннектор SONET с интерфейсами OC-192 может демультимплексировать поток OC-192 в 4 потока OC-48. Если множество интерфейсов группируется в один канал TE с использованием связывания (link bundling) [RFC4201], ресурсы канала должны идентифицироваться с использованием трех уровней - Link_Id, идентификатор компонентного соединения и метка, указывающая виртуальное устройство, временной интервал и т. п. Выделение ресурсов происходит на нижнем уровне (метки), а физические соединения выполняются на уровне компонентного канала. В качестве другого примера рассмотрим оптический коммутатор (например, PXC) прозрачно коммутирующий оптические пути OC-192. Если множество интерфейсов снова группируется в один канал TE, связывания каналов [RFC4201] не требуется и нужны только два уровня идентификации - Link_Id и Port_Id. В этом случае выделение ресурсов и физическое подключение выполняются на нижнем уровне (порт).

Для обеспечения взаимодействия между устройствами с разными возможностями мультиплексирования поддерживающим LMP устройствам **следует** разрешать локальную настройку субканалов на компонентных соединениях, как (логических) каналов данных. Например, если маршрутизатор с интерфейсами 4 OC-48 подключен через мультиплексор 4:1 MUX к кросс-коннектору с интерфейсами OC-192, кросс-коннектору следует обеспечивать возможность настройки каждого субканала (например, STS-48с SPE, если 4:1 MUX является мультиплексором SONET), как канала данных.

Протокол LMP предназначен для поддержки агрегирования одного или множества каналов данных в канал TE (порты или компонентные соединения в каналы TE). Целью формирования канала TE является группировка/отображение информации о некоторых физических ресурсах (и их свойствах) в информацию, которая будет применяться CSPF¹ для расчета пути, а также послужит для сигнализации GMPLS.

1.1. Терминология

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].

Предполагается, что читатель знаком с терминологией [RFC3471], [RFC4202] и [RFC4201].

Bundled Link — составной канал

В соответствии с [RFC4201] составной канал является каналом TE и по этой причине для целей сигнализации GMPLS комбинации <link identifier, label> не достаточно для однозначной идентификации соответствующих ресурсов, используемых LSP. Составной канал включает два или более компонентных соединения.

Control Channel — канал управления

Канал управления представляет собой пару доступных один для другого интерфейсов, которые служат для обеспечения связи между узлами с целями маршрутизации, сигнализации и управления каналами.

Component Link — компонентный канал

В соответствии с [RFC4201] компонентное соединение представляет собой подмножество ресурсов канала TE, такое, что (а) отделение от других минимально и (b) в каждом подмноестве метки достаточно для однозначной идентификации соответствующих ресурсов, используемых LSP.

Data Link — канал передачи данных

Канал данных представляет собой пару интерфейсов, служащих для передачи пользовательских данных. Отметим, что в GMPLS канал(ы) управления между двумя смежными узлами больше не обязаны использовать ту же физическую среду передачи, которая применяется для канала данных между этими узлами.

Link Property Correlation — корреляция свойств канала

Процедура организации связи (корреляции) между локальными и удаленными свойствами канала TE.

¹Constrained SPF — поиск кратчайшего пути с учетом ограничений.

Multiplex Capability — возможности мультиплексирования

Способность мультиплексировать/демультиплексировать поток данных по субпотокam для коммутирования.

Node Id — идентификатор узла

Для узла OSPF идентификатор LMP Node_Id совпадает с адресом, содержащимся в OSPF Router Address TLV. Для узла IS-IS при анонсировании TE Router ID TLV идентификатор Node_Id совпадает с анонсируемым Router ID.

Port — порт

Интерфейс, завершающий канал данных.

TE Link — канал TE

В соответствии с [RFC4202] канал TE является логической конструкцией, представляющей способ группировки/отображения информации о неких физических ресурсах (и их свойствах), соединяющих маршрутизаторы LSR, в данные, используемые CSPF для расчета пути и сигнализации GMPLS.

Transparent - прозрачный

Устройство называется X-прозрачным, если оно пересылает входящие сигналы со входа на выход без проверки и изменения X-компоненты этих сигналов. Например, коммутатор Frame Relay прозрачен на сетевом уровне, а оптические коммутаторы прозрачны для электрических сигналов.

2. Обзор LMP

Двумя основными процедурами LMP являются поддержка канала управления и корреляция свойств канала. Поддержка канала управления служит для организации и обслуживания каналов управления между смежными узлами. Это выполняется с использованием сообщений Config и механизма ускоренного обмена сообщениями keep-alive, который нужен в тех случаях, когда не доступны механизмы нижележащих уровней для обнаружения отказов в канале управления. Корреляция свойств канала используется для синхронизации свойств канала TE и проверки его конфигурации.

LMP требует, чтобы между парой узлов имелся хотя бы один активный двухсторонний канал управления. Каждое направление канала управления идентифицируется значением CC_Id¹ и два направления связываются между собой с помощью обмена сообщениями LMP Config. За исключением сообщений Test, которые могут применяться транспортным механизмом для обмена в основной полосе, все пакеты LMP передаются по протоколу UDP через порт LMP. Кодирование для канала управления на уровне логического канала (L2) выходит за рамки этого документа.

LMP-смежность между парой узлов возникает при наличии хотя бы одного двухстороннего канала управления между ними. Для каждой смежности может существовать множество одновременно действующих каналов управления, однако характеристики **должны** согласовываться для каждого канала индивидуально. При использовании LMP fast keep-alive не канале управления обмен сообщениями LMP Hello **должен** выполняться через канал управления. Другие сообщения LMP **могут** передаваться через любой активный канал управления между парой смежных узлов. Активные каналы управления могут группироваться в логический канал управления для сигнализации маршрутизации и коррелирования свойств.

Функция коррелирования свойств канала LMP предназначена для агрегирования множества каналов данных (порты или компонентные соединения) в канал TE и синхронизации свойств канала TE. Частью этой функции является обмен сообщениями LinkSummary. Такие сообщения включают локальный и удаленный идентификатор Link_Id, список всех каналов данные в составе канала TE и различные свойства канала. Сообщения LinkSummaryAck или LinkSummaryNack **должны** передаваться в ответ на сообщение LinkSummary, указывающее согласование свойств канала или отмену такого согласования.

Сообщения LMP передаются с гарантией доставки, основанной на использовании Message_Id и повторов передачи. Идентификаторы сообщений передаются в объектах MESSAGE_ID. В сообщении LMP можно включать не более одного объекта MESSAGE_ID. Для сообщений, относящихся к каналам управления, Message_Id находятся в области действия канала управления, через который сообщение передается. Для сообщений, относящихся к каналам TE, значения Message_Id находятся в области действия смежности LMP. Значения Message_Id монотонно увеличиваются с возвратом в 0 при достижении максимума.

В этом документе определены две дополнительных процедуры LMP — проверка связности канала и контроль отказов. Эти процедуры полезны, в частности, когда каналы управления физически отделены от каналов данных. Проверка связности канала используется для обнаружения уровня данных (data plane discovery), обмена Interface_Id (эти идентификаторы применяются в сигнализации GMPLS в качестве меток порта или идентификатора композитного соединения, в зависимости от конфигурации), а также проверки физической связности. Проверка выполняется путем передачи сообщений Test через каналы данных и возврата в ответ на них сообщений TestStatus через канал управления. Отметим, что сообщение Test является единственным сообщением LMP, передаваемым по каналам данных. Обмен сообщениями ChannelStatus между применяется смежными узлами для подавления нисходящих сигналов и локализации отказов в целях защиты и восстановления.

Для проверки связности канала LMP сообщение Test передается по каналам данных. Для X-прозрачных устройств это требует проверки и изменения аспекта X в сигналах. Процедура проверки связности канала LMP координируется с помощью обмена сообщениями BeginVerify через канал управления. Для поддержки разных аспектов прозрачности в сообщения BeginVerify и BeginVerifyAck включен механизм VTM². Отметим, что не вводятся требования одновременной потери прозрачности всеми каналами данных — считается возможной потеря прозрачности, как минимум, на одном канале. Также не требуется использовать одну физическую среду для канала управления и канала TE, однако канал управления **должен** завершаться на тех же двух элементах управления, которые служат для управления каналом TE. Поскольку обмен сообщениями BeginVerify координирует процедуру Test, он естественным образом координирует режим прозрачности каналов данных.

Процедура контроля отказов LMP основана на обмене сообщениями ChannelStatus, включающем ChannelStatus, ChannelStatusAck, ChannelStatusRequest и ChannelStatusResponse. Сообщение ChannelStatus передается без запроса и служит для уведомления соседа LMP о состоянии одного или множества каналов данных в канале TE. Сообщение ChannelStatusAck служит для подтверждения приема ChannelStatus. Сообщение ChannelStatusRequest используется для запроса у соседа LMP данных о состоянии одного или множества каналов данных в канале TE. Сообщение ChannelStatusResponse служит для подтверждения приема ChannelStatusRequest и содержит запрошенные данные.

¹Control Channel Id — идентификатор канала управления.

²Verify Transport Mechanism — механизм проверки транспорта.

3. Поддержка канала управления

Для организации смежности LMP между двумя узлами **должен** быть активирован хотя бы один двухсторонний канал управления. Каналы управления могут применяться для обмена управляющей информацией типа данных о поддержке канала или контроле отказов (с использованием протокола сообщений типа описанного здесь LMP), данных управления путями или распространением меток (с использованием сигнального протокола типа RSVP-TE [RFC3209]) и данных о сетевой топологии и распространении сведений о состояниях (с использованием расширений для организации трафика в протоколах типа OSPF [RFC3630] или IS-IS [RFC3784]).

Для целей LMP точная реализация канала управления не задается — это может быть, например, отдельная длина волны или волокно, соединение Ethernet, туннель IP через отдельную сеть управления или просто дополнительные байты в канале данных. Каждый узел задает для канала управления уникальный (для себя) 32-битовый целочисленный идентификатор, отличный от 0, `CC_Id`. Значения идентификаторов берутся из одного пространства с идентификаторами безадресных интерфейсов. Пакеты LMP передаются по протоколу UDP с использованием порта LMP. Таким образом, канальное представление управляющего канала не является частью спецификации LMP.

Для организации канала управления нужно знать IP-адрес, которому будут приниматься пакеты на удаленной стороне. Этот адрес можно задать в конфигурации вручную или определить автоматически. Отметим, что для сигнализации по основному каналу (in-band signaling) канал управления можно явно задать в конфигурации конкретного канала данных. В этом случае можно использовать обмен сообщениями Config для динамического определения адреса IP на удаленном конце канала управления. Это выполняется путем передачи сообщения Config с индивидуальным адресом отправителя по групповому адресу IP (224.0.0.1 или ff02::1). В ответ **должны** передаваться сообщения ConfigAck и ConfigNack по адресу IP из поля отправителя в заголовке пакета с сообщением Config.

Каналы управления существуют независимо от каналов TE и между парой узлом может быть организовано множество одновременных каналов управления. Отдельные каналы управления могут быть реализованы разными способами, например, один может быть создан в оптическом волокне, а другой вне волокна. В любом случае параметры **должны** согласовываться через каждый канал управления для поддержки связности LMP, если иные механизмы не доступны. Поскольку каналы управления электрически завершаются на каждом узле, можно детектировать отказы каналов управления с использованием нижележащих уровней (например, SONET/SDH).

Имеется четыре сообщения LMP, которые применяются для управления отдельными каналами - Config, ConfigAck, ConfigNack и Hello. Эти сообщения **должны** передаваться через канал, к которому они относятся. Все прочие сообщения LMP можно передавать через любой активный канал управления между парой смежных узлов LMP.

Для поддержки смежности LMP требуется наличие хотя бы одного активного канала между парой смежных узлов (напомним, что между парой узлом может одновременно существовать множество активных каналов управления). При отказе канала управления можно использовать дополнительный канал управления, как описано ниже.

3.1. Согласование параметров

Активизация канала управления начинается с обмена для согласования параметров, использующего сообщения Config, ConfigAck и ConfigNack. Содержимое этих сообщений создается на основе объектов LMP, которые могут быть согласуемыми или несогласуемыми (указываются битом N в заголовке объектов). Согласуемые объекты могут применяться для согласования партнерами LMP неких значений. Несогласуемые объекты служат для анонсирования конкретных значений, которые не нужно или невозможно согласовать.

Для активизации канала управления удаленному партнеру **должно** быть передано сообщение Config и от него локальный узел **должен** получить сообщение ConfigAck. В сообщении Config содержится локальный идентификатор канала управления (`CC_Id`), идентификатор узла-отправителя (`Node_Id`), идентификатор сообщения `Message_Id` для обеспечения гарантированной доставки, а также объект CONFIG. Возможно одновременное начало процедуры активизации канала управления с обеих сторон. Для предотвращения неоднозначностей в атких случаях «выигрывает» узел с большим значением `Node_Id`, а узел с меньшим `Node_Id` **должен** прекратить передачу сообщения Config и ответить на принятое сообщение Config. Если значения `Node_Id` совпадают, это говорит о некорректной настройке конфигурации одного или обоих узлов. В этом случае узлы **могут** продолжить повторы передачи сообщений Config, надеясь на устранение конфигурационных ошибок. Отметим, что эту проблему может решить оператор, сменив значение `Node_Id` на одной или обеих сторонах.

Сообщения ConfigAck служат для подтверждения приема сообщений Config и выражения согласия со **всеми** конфигурационными параметрами (согласуемыми и несогласуемыми).

Сообщения ConfigNack подтверждают прием Config и указывают какие (если они есть) из несогласуемых объектов CONFIG не приемлемы, а также предлагают дополнительные значения для согласуемых параметров.

Если узел получает сообщение ConfigNack с устраивающими его дополнительными значениями для согласуемых параметров, ему **следует** передать в ответ сообщение Config с соответствующими значениями параметров.

Если узел получает сообщение ConfigNack с неподходящими дополнительными значениями, он **может** продолжить повтор сообщений Config в надежде на исправление конфигурационных ошибок. Отметим, что эту проблему может решить оператор, сменив значения параметров на одной или обеих сторонах.

При использовании множества каналов управления на одном физическом интерфейсе обмен для согласования параметров выполняется для каждого канала управления. Разные сообщения LMP для согласования параметров связываются с соответствующими каналами по уникальным в масштабе узла значениям идентификаторов (`CC_Id`).

3.2. Протокол Hello

Как только будет активизирован канал управления между парой смежных узлов, через него можно начинать использование протокола LMP Hello для организации связности между узлами и детектирования отказов в канале управления. Протокол LMP Hello используется в качестве облегченного механизма проверки «живучести» канала (keep-alive) для быстрого реагирования на отказы, чтобы не терялись сообщения IGP Hello и соответствующие отношения смежности (link-state adjacency) без необходимости не удалялись.

3.2.1. Согласование параметров Hello

До отправки сообщений Hello **должны** быть согласованы параметры HelloInterval и HelloDeadInterval на локальном и удаленном узлах. Обмен этими параметрами выполняется в сообщении Config. Параметр HelloInterval задает частоту передачи сообщений LMP Hello в миллисекундах (мсек). Например, при значении параметра 150 передающий узел будет отправлять сообщения Hello хотя бы один раз за каждые 150 мсек. HelloDeadInterval указывает интервал (в мсек), в течение которого устройство будет ожидать сообщения Hello, пока не сочтет канал управления «умершим».

Значение HelloDeadInterval **должно** быть больше HelloInterval и следует делать его по крайней мере втрое больше HelloInterval. Если не используется механизм ускоренной проверки живучести (fast keep-alive) LMP, для параметров HelloInterval и HelloDeadInterval **должно** устанавливаться значение 0.

Значения HelloInterval и HelloDeadInterval следует выбирать аккуратно для обеспечения быстрого отклика на отказы канала управления и отсутствия перегрузок. В силу этого очевидно использование различных значений параметров в разных реализациях каналов управления. Для каналов управления, организованных через прямое соединение, предлагается по умолчанию использовать значения HelloInterval = 150 и HelloDeadInterval = 500 мсек.

Когда узел передал или получил сообщение ConfigAck, он может начинать отправку сообщений Hello. После отправки сообщения Hello и получения приемлемого Hello (т. е., с ожидаемым порядковым номером — см. параграф 3.2.2) канал управления переходит в активное состояние (up) (возможен переход в состояние up без отправки сообщений Hello, если для индикации двухсторонней связности канала управления применяются другие методы; например, данные о связности канала управления могут быть получены от транспортного уровня). Однако, если узел получает сообщение ConfigNack взамен ConfigAck, он **должен** отказаться от передачи сообщения Hello, а канал управления **не следует** переводить в состояние up. Машина конечных состояний (FSM) для канала управления описана в параграфе 11.1.

3.2.2. Ускоренная проверка жизнеспособности

Каждое сообщение Hello содержит два порядковых номера — первый (TxSeqNum) указывает номер сообщения Hello, которое будет передаваться, а второй (RcvSeqNum) — номер последнего сообщения Hello, полученного от смежного узла через данный канал управления.

Для порядковых номеров имеется два специальных значения. Для TxSeqNum **недопустимо** использовать значение 0. TxSeqNum = 1 используется для индикации того, что отправитель только начал работу и еще не знает номера последнего переданного сообщения. Таким образом, первое сообщение Hello передается с TxSeqNum = 1 и RxSeqNum = 0. Когда TxSeqNum достигает значения $2^{32}-1$, следующим номером будет 2, а не 0 или 1, поскольку эти номера имеют особое значение.

При нормальной работе разница между RcvSeqNum в полученном сообщении Hello и локальным номером TxSeqNum для генерируемых сообщений в большинстве случаев будет составлять 1. Эта разность может становиться больше 1 только при перезапуске управляющего канала или достижении максимального номера.

Поскольку 32-битовые порядковые номера могут достигать максимума, можно воспользоваться приведенным ниже выражением для проверки того, что недавно полученное значение TxSeqNum меньше полученного перед этим.

```
if ((int) old_id - (int) new_id > 0) {  
    новое значение меньше старого;  
}
```

Наличие порядковых номеров в сообщениях Hello позволяет каждому узлу убедиться, что его партнер получает сообщения Hello. За счет включения RcvSeqNum в пакеты Hello локальный узел узнает какие пакеты Hello получены удаленным узлом.

Приведенный ниже пример показывает применение порядковых номеров. Отметим, что показаны операции лишь на одном из узлов и возможны другие варианты.

- 1) После завершения конфигурационного этапа узел А передает сообщения Hello узлу В с {TxSeqNum=1;RcvSeqNum=0}.
- 2) Узел А получает Hello от узла В с {TxSeqNum=1;RcvSeqNum=1}. По истечении интервала HelloInterval узел А передает узлу В сообщения Hello {TxSeqNum=2;RcvSeqNum=1}.
- 3) Узел А получает Hello от узла В с {TxSeqNum=2;RcvSeqNum=2}. По истечении интервала HelloInterval узел А передает узлу В сообщения Hello {TxSeqNum=3;RcvSeqNum=2}.

3.2.3. Отключение канала управления

Для обеспечения возможности аккуратного отключения канала управления с административными целями имеется флаг ControlChannelDown в базовом заголовке (Common Header) пакетов LMP. При наличии между парой узлов активных каналов передачи данных административное отключение канала управления **следует** использовать только в тех случаях, когда имеются другие активные каналы управления, которые можно использовать для поддержки каналов данных.

При административном отключении канала управления узел **должен** устанавливать флаг ControlChannelDown во всех сообщениях LMP, передаваемых через канал управления. Инициировавший отключение управляющего канала узел может прекратить передачу сообщений Hello по истечении HelloDeadInterval секунд или при получении через этот же канал управления сообщения LMP с установленным флагом ControlChannelDown.

Узлу, получившему пакет LMP с установленным флагом ControlChannelDown, **следует** передать сообщение Hello с установленным флагом ControlChannelDown и перевести канал управления в отключенное состояние (down).

3.2.4. Вырожденное состояние

Следствием того, что каналы управления могут физически отличаться от связанных с ними каналов данных, может не оказаться ни одного активного канала управления при наличии используемых каналов данных. Для многих приложений неприемлем разрыв канала передачи данных лишь по причине недоступности канала управления, однако для трафика

через такие каналы данных уже не будет возможности обеспечить прежний уровень обслуживания. Следовательно, канал TE будет находиться в вырожденном (Degraded) состоянии.

Когда канал TE находится в состоянии Degraded, **следует** уведомлять системы маршрутизации и сигнализации о том, что не нужно воспринимать новые соединения через этот канал TE и нужно анонсировать для него отсутствие незарезервированных ресурсов.

4. Сопоставление свойств каналов

В рамках LMP для каналов TE определен обмен для сопоставления свойств каналов с помощью сообщений LinkSummary, LinkSummaryAck и LinkSummaryNack. Содержимое этих сообщений создается с использованием объектов LMP, которые могут быть согласуемыми или несогласуемыми (указывается влагом N в заголовке объекта). Согласуемые объекты служат для того, чтобы стороны могли совместно выбирать значения некоторых параметров канала. Несогласуемые объекты служат для анонсирования конкретных значений, которые не нужно или невозможно согласовать.

Каждый канал TE имеет идентификатор (Link_Id), выделенный на каждой стороне соединения. Эти идентификаторы **должны** быть одного типа (т. е., IPv4, IPv6 или безадресный) на обеих сторонах. Если принимается сообщение LinkSummary с разными типами канала TE на локальной и удаленной стороне, в ответ **должно** передаваться сообщение LinkSummaryNack с кодом ошибки Bad TE Link Object. Аналогично, на каждой стороне канала присваивается идентификатор интерфейса (Interface_Id). Эти идентификаторы также **должны** иметь одинаковый тип на обеих сторонах. При получении LinkSummary с разнотипными Interface_Id на локальной и удаленной стороне в ответ **должно** передаваться сообщение LinkSummaryNack с кодом ошибки Bad Data Link Object.

Сопоставление свойств канала **следует** выполнять до того, как канал будет активизирован (up) и оно **может** быть выполнено в любой момент, когда канал активен и не находится в процессе проверки (Verification).

Сообщения LinkSummary служат для проверки согласованности информации о канале данных и TE на обеих сторонах, а также для (1) агрегирования множества каналов данных (портов или компонентных каналов) в один канал TE, (2) обмена, сопоставления (для обнаружения несогласованности) или изменения параметров канала TE и (3) обмена, сопоставления (для обнаружения несогласованности) или изменения значений Interface_Id (Port_Id или идентификаторы компонентных каналов).

Сообщение LinkSummary включает объект TE_LINK, за которым следует один или множество объектов DATA_LINK. Объект TE_LINK указывает идентификаторы Link_Id на локальной и удаленной стороне канала TE, а также поддержку процедур контроля отказов и верификации для канала TE. Объекты DATA_LINK служат для указания характеристик каналов данных, образующих канал TE. Эти объекты включают локальный и удаленный идентификатор Interface_Id и могут включать один или множество субобъектов с дополнительным описанием свойств каналов данных.

Если от удаленного узла получено сообщение LinkSummary и отображение Interface_соответствует локальной копии, это говорит о том, что два узла согласовали процедуру Verification (см. раздел 5) и настройку идентификации канала данных. Если процедура проверки не используется, сообщение LinkSummary можно применять для проверки согласования при настройке вручную.

Сообщение LinkSummaryAck служит для сигнализации о согласовании отображений Interface_Id и определений свойств каналов. В остальных случаях **должно** передаваться сообщение LinkSummaryNack, показывающее интерфейсы с некорректным отображением и/или неприемлемые свойства канала. Если сообщение LinkSummaryNack показывает, что отображение Interface_Id некорректно, а процедура проверки включена, процесс проверки канала **следует** повторить для всех несоответствующих свободных каналов данных. Если сообщение LinkSummaryNack включает согласуемые параметры, **должны** быть указаны приемлемые значения этих параметров. Если полученное сообщение LinkSummaryNack включает согласуемые параметры, инициатору сообщения LinkSummary **следует** отправить новое сообщение LinkSummary. В это новое сообщение LinkSummary **следует** включить новые значения для согласуемых параметров. В этих значениях **следует** принимать во внимание приемлемые значения параметров из сообщения LinkSummaryNack.

Сообщение LinkSummary может стать довольно большим по причине большого числа объектов DATA LINK. Реализациям LMP **следует** поддерживать возможность фрагментации при передаче сообщений LMP и они **должны** поддерживать сборку фрагментов IP при получении сообщений LMP.

5. Проверка связности для канала

В этом разделе описана необязательная процедура, которая может применяться для проверки физической связности каналов данных и динамического обнаружения связей между каналом TE и Interface_Id. Процедуру **следует** выполнять при организации канала TE, а потом периодически повторять для всех нераспределенных (свободных) каналов данных соединения TE.

Поддержка этой процедуры указывается флагом Link Verification Supported в объекте TE_LINK сообщений LinkSummary.

Если получено сообщение BeginVerify, а проверка каналов не поддерживается для соединений TE, в ответ **должно** передаваться сообщение BeginVerifyNack с кодом ошибки (Error Code), показывающим, что проверка для этого канала TE не поддерживается (Link Verification Procedure not supported for this TE Link).

Уникальной характеристикой прозрачных устройств является то, что данные не изменяются и не проверяются в процессе нормальной работы. Это создает сложности при проверке связности каналов данных и организации отображения меток. Поэтому для обеспечения подобающей проверки связности канала данных требуется непрозрачность таких каналов до того момента, как они будут выделены для пользовательского трафика. Для поддержки той или иной степени непрозрачности (например, проверка служебных байтов, и терминирование полезной нагрузки IP и т. п.) и, следовательно, поддержки различных механизмов транспортировки сообщений Test в сообщениях BeginVerify и BeginVerifyAck включается поле Verify Transport Mechanism.

Требование одновременного завершения (терминирования) всех каналов данных не задается, но каналы данных **должны** (как минимум) обеспечивать поддержку такой возможности. Кроме того, для процедуры проверки каналов предполагается, что архитектура узлов позволяет передавать и принимать сообщения через любой канал данных. Отметим, что это требование является тривиальным для непрозрачных устройств, поскольку каждый канал данных завершается электрически и обрабатывается до пересылки в следующее непрозрачное устройство. Однако для прозрачных устройств это создает дополнительное требование.

Для соединения двух устройств между ними определяется канал TE и **должен** присутствовать хотя бы один активный канал управления между этими узлами. Для проверки связности канал TE **должен** включать хотя бы один канал данных.

После организации между двумя узлами канала управления проверку связности канала данных можно проверить путем обмена сообщениями Test через каждый из каналов данных, указанный в канале TE. Следует отметить, что все сообщения LMP, за исключением Test, будут передаваться через каналы управления и обмен сообщениями Hello через каждый канал управления будет продолжаться и после завершения проверки каналов данных. Сообщение Test передаются через проверяемые каналы. Каналы данных тестируются в направлении передачи, поскольку они являются односторонними. Следовательно, оба узла могут выполнять обмен сообщениями Test одновременно (и независимо).

Для начала процедуры проверки канала локальный узел **должен** отправить через канал управления сообщение BeginVerify. Для ограничения процедуры Link Verification конкретным каналом TE, локальное значение Link_Id **должно** отличаться от 0. Если это поле равно 0, канал данных может охватывать множество каналов TE и/или в число проверяемых каналов TE могут попасть еще не настроенные. Для случаев локального Link_Id = 0 используется флаг Verify all Links в объекте BEGIN_VERIFY, позволяющий различать каналы данных, охватывающие множество каналов TE и каналы данных, еще не связанные с каналом TE. В частности, проверка каналов данных, охватывающих множество каналов TE, указывается локальным полем Link_Id = 0 и установкой флага Verify all Links. Проверка каналов данных, которые еще не привязаны к каналам TE, указывается установкой локального поля Link_Id = 0 и сбросом флага Verify all Links.

Сообщение BeginVerify содержит также число проверяемых каналов данных, интервал, в течение которого будут передаваться сообщения Test (VerifyInterval), поддерживаемую схему кодирования и транспортные механизмы, скорость передачи для сообщений Test, а также (в тех случаях, когда каналы данных соответствуют оптическим волокнам) идентификаторы длин волн, на которых будут передаваться сообщения Test.

Если удаленный узел получил сообщение BeginVerify и готов обрабатывать сообщения Test, он **должен** передать в ответ сообщение BeginVerifyAck, указывающее желаемый для сообщений TEST транспортный механизм. Удаленный узел включает в это сообщение 32-битовое значение Verify_Id, уникальное в масштабах данного узла. Значение Verify_Id **может** выбираться случайно, однако **недопустимо** его совпадение с любым Verify_Id, используемым в это же время выбравшим значение узлом. Verify_Id после этого используется во всех проверочных сообщениях для того, чтобы различать разных партнеров LMP и/или параллельные процедуры Test. Когда локальный узел получит сообщение BeginVerifyAck от удаленного узла, он может начать тестирование каналов данных путем периодической отправки сообщений Test в каждый канал. Сообщение Test включает Verify_Id и локальное значение Interface_Id для соответствующего канала данных. Удаленный узел **должен** передавать в ответ сообщение TestStatusSuccess или TestStatusFailure для каждого проверяемого канала. Для подтверждения приема TestStatusSuccess и TestStatusFailure **должно** передаваться сообщение TestStatusAck. Неподтвержденные сообщения TestStatusSuccess и TestStatusFailure **следует** повторять по получения подтверждения или достижения предельного числа повторов (см. также раздел 10).

Для отправителя допустимо прерывание процедуры Test в любой момент после отправки сообщения BeginVerify. Для этого **следует** передать сообщение EndVerify.

Сопоставление сообщений выполняется с использованием идентификаторов сообщений и значений Verify_Id, это позволяет параллельно проверять каналы данных, относящиеся к разным группам каналов или сессиям LMP.

При получении сообщения Test идентификатор интерфейса Interface_Id из этого сообщения (используется в GMPLS как метка порта или идентификатор компонентного канала в зависимости от конфигурации) записывается и отображается на локальный идентификатор Interface_Id для принявшего сообщение канала данных, а в ответ **должно** передаваться сообщение TestStatusSuccess, включающее локальное значение Interface_Id, а также значения Interface_Id и Verify_Id из принятого сообщения Test. Получение TestStatusSuccess показывает, что сообщение Test было обнаружено удаленным узлом и физическая связность канала данных была подтверждена. При получении TestStatusSuccess локальному узлу **следует** пометить канал данных как активный и отправить удаленному узлу сообщение TestStatusAck. Однако, если сообщение Test не было обнаружено на удаленной стороне в течение периода наблюдения (задается значением VerifyDeadInterval), удаленный узел **должен** передать через канал управления сообщение TestStatusFailure, показывающее неудачу при проверке физической связности канала данных. Локальному узлу, получившему сообщение TestStatusFailure, **следует** пометить канал данных как сбойный (FAILED) и передать удаленному узлу сообщение TestStatusAck. После проверки всех каналов данных локальному узлу **следует** передать сообщение EndVerify для индикации завершения тестирования на данном канале.

Если известно отображение локальных каналов данных на удаленные, процедуру проверки каналов можно оптимизировать, проверяя каналы данных в определенном порядке, который знают оба узла. Предлагается выбирать порядок в соответствии с ростом значений Interface_Id на удаленной стороне.

Локальному и удаленному узлу **следует** поддерживать полный список отображений Interface_Id для целей сопоставления.

5.1. Пример проверки связности для канала

На рисунке 1 показан пример проверки связности, которая выполняется при добавлении канала между узлами A и B. В этом примере канал TE состоит из трех свободных портов (каждый из которых использует свое волокно) и связан с двухсторонним каналом управления (отмечен буквой c). Процесс проверки связности поэтапно описан ниже.

- Узлу В через канал управления передается сообщение BeginVerify, говорящее о начале проверки портов, образующих канал ТЕ. Объект LOCAL_LINK_ID в сообщении BeginVerify содержит идентификатор (адрес IP или индекс интерфейса), присвоенный каналу узлом А.
- Получив сообщение BeginVerify, узел В создает идентификатор Verify_Id и привязывает его к каналу ТЕ от А. Эта привязка используется при получении узлом В сообщений Test от узла А, содержащих Verify_Id. Узел В определяет идентификатор (адрес IP или индекс интерфейса), который узел А присвоил каналу ТЕ, просматривая объект LOCAL_LINK_ID из полученного сообщения BeginVerify (если порты данных еще не выделены для ТЕ Link, привязка ограничена Node_Id для узла А). В ответ на сообщение BeginVerify узел В передает сообщение BeginVerifyAck узлу А. Объект LOCAL_LINK_ID в сообщении BeginVerifyAck служит для передачи идентификатора (адрес IP или индекс интерфейса), который узел В присвоил каналу ТЕ. Объект REMOTE_LINK_ID в сообщении BeginVerifyAck служит для привязки идентификаторов Link_Id, выделенных узлами А и В. Значение Verify_Id возвращается узлу А в сообщении BeginVerifyAck по каналу управления.
- Когда узел А получает сообщение BeginVerifyAck, он начинает переиодическую отпавку сообщений Test через первый порт (Interface Id=1). Сообщение Test включает Interface_Id для порта и значение Verify_Id, присвоенное злом В.
- При получении узлом В сообщений Test он отображает полученный идентификатор Interface_Id на свой локальный идентификатор Interface_Id = 10 и передает сообщение TestStatusSuccess узлу А через канал управления. Сообщение TestStatusSuccess включает локальное и принятое значения Interface_Id для портов, а также Verify_Id. Идентификатор Verify_Id используется для определения локального и удаленного идентификаторов канала ТЕ (адреса IP или индексы интерфейсов), к которым относятся каналы данных.
- Узел А будет передавать через канал управления узлу В сообщение TestStatusAck, показывающее, что он получил сообщение TestStatusSuccess.
- Процесс повторяется до завершения проверки всех портов.
- Аосле этого узел А передает через канал управления сообщение EndVerify, говорящее узлу В о завершении проверки.
- В отвечает на это сообщение отпавкой через канал управления узлу А сообщения EndVerifyAck.

Отметим, что эта процедура может применяться для «обнаружения» связности между портами данных.

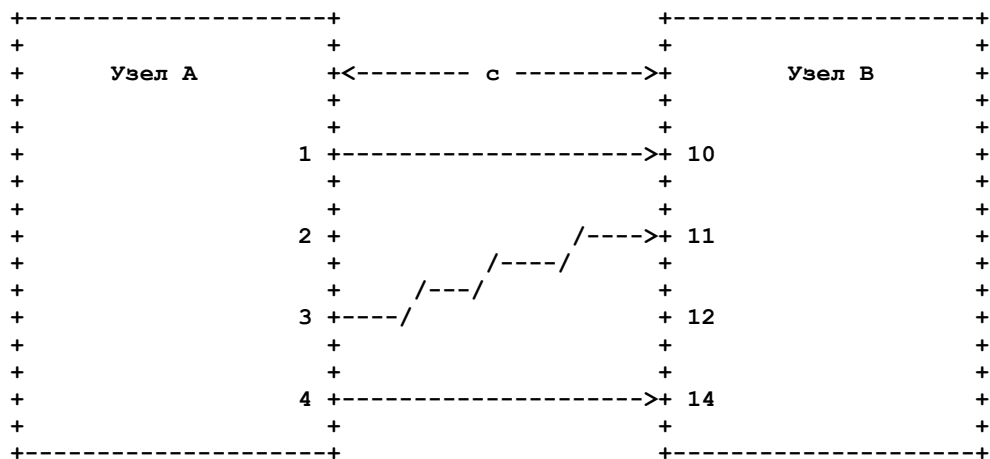


Рисунок 1. Пример проверки связности между узлами А и В

6. Обработка отказов

В этом разделе описана необязательная процедура LMP, служащая для обработки отказов путем быстрого уведомления о состоянии одного или множества каналов данных в канале ТЕ. Областью действия этой процедуры является канал ТЕ и поэтому применение этой процедуры согласуется в процессе обмена LinkSummary. Процедура может служить для быстрой изоляции отказов каналов данных и канала ТЕ и предназначена для использования как на односторонних, так и на двухсторонних LSP.

Важным следствием применения прозрачных устройств является неприменимость традиционных методов отслеживания состояний выделенных каналов данных. Вместо детектирования отказов на уровнях 2 и 3 эта функция перенесена на физический уровень (т. е., обнаружение потери оптического сигнала или оптический мониторинг данных).

Напомним, что канал ТЕ, соединяющий два узла, может состоять из множества каналов данных. В случае отказа одного или нескольких таких каналов нужен механизм быстрого уведомления с целью запуска соответствующей процедуры защиты/восстановления. При восстановлении отказавшего канала этот же механизм должен уведомлять об устранении отказа и восстановлении нормальной работы канала.

6.1. Детектирование отказов

Детектирование отказов должно происходить на ближайшем к отказу уровне — для оптических сетей это физический (оптический) уровень. Одним из способов обнаружения отказов на физическом уровне является детектирование «потери света» (LOL¹). Другие методы мониторинга оптических сигналов пока находятся в стадии разработки и не рассматриваются в этом документе. Однако следует понимать, что механизм, используемый для уведомлений об отказах в LMP, не зависит от способа детектирования отказов и основан просто на фактах обнаружения отказа.

¹Loss of light.

6.2. Процедура локализации отказа

В некоторых ситуациях отказ канала данных между двумя узлами распространяется в нисходящем направлении так, что все нисходящие узлы обнаруживают отказ без его локализации. Для предотвращения множественных уведомлений об одном отказе LMP обеспечивает уведомление об отказах с помощью сообщений ChannelStatus. Это сообщение можно использовать для индикации отказа одного или многих каналов данных, а также всего канала TE. Сопоставление отказов выполняется локально на каждом узле, получившем уведомление об отказе.

Для связывания отказа с конкретным каналом между смежными узлами нисходящий (в смысле потока данных) узел, обнаруживший отказ канала данных будет передавать своему соседу в восходящем направлении сообщение ChannelStatus, показывающее, что обнаружен отказ (уведомления для всех каналов с отказами объединяются). Получивший сообщение ChannelStatus восходящий узел **должен** передать в ответ сообщение ChannelStatusAck, подтверждающее прием ChannelStatus. Восходящему узлу следует провести сопоставление с целью определить, обнаружен ли этот отказ локально для соответствующих LSP. Если, например, будет ясно, что отказ наблюдается на входе восходящего узла или внутри него, восходящий узел будет знать точку отказа. После сопоставления информации об отказах восходящему узлу **следует** передать нисходящему узлу сообщение ChannelStatus, которое показывает наличие или отсутствие отказа в канале. Если сообщение ChannelStatus не будет получено нисходящим узлом, тому **следует** передать сообщение ChannelStatusRequest для соответствующего канала. После определения точки отказа можно использовать сигнальный протокол для инициирования процедур защиты или восстановления интервала (span) или пути.

Если отказ произошел на всех каналах данных соединения TE, восходящий узел **может** быть уведомлен об отказе канала TE без указания конкретных каналов данных. Это выполняется путем отправки уведомления в сообщении ChannelStatus, идентифицирующем канал TE без указания Interface_Id в объекте CHANNEL_STATUS.

6.3. Примеры локализации отказов

На рисунке 2 приведен пример сети, где 4 узла соединены в линейный массив. Каналы управления являются двухсторонними и помечены буквой с. Все LSP также являются двухсторонними.

В первом случае [(a) на рисунке] возникает отказ на одном из направлений двухстороннего LSP. Узел 4 обнаружит этот отказ и передаст узлу 3 сообщение ChannelStatus, сообщающее об отказе (например, LOL) соответствующему восходящему узлу. Узел 3, получив сообщение ChannelStatus от узла 4, вернет тому сообщение ChannelStatusAck и выполнит для него локальное сопоставление. Когда узел 3 выполнит сопоставление для этого отказа и проверит его, он свяжет отказ с каналом данных между узлами 3 и 4. В этот момент узлу 3 следует отправить на узел 4 сообщение ChannelStatus, говорящее о локализации отказа.

Во втором случае [(b) на рисунке] один отказ (например, повреждение волокна) воздействует на оба направления двухстороннего LSP. Узел 2 (узел 3) обнаружит отказ в восходящем (нисходящем) направлении и отправит восходящему (относительно потока данных) узлу сообщение ChannelStatus, говорящее об отказе (например, LOL). Одновременно (не учитывая задержек при распространении) узел 1 (узел 4) обнаружит отказ на восходящем (нисходящем) направлении и будет передавать соответствующему восходящему (относительно потока данных) узлу сообщение ChannelStatus, говорящее об отказе. Узлы 2 и 3 будут локализовать отказ в двух направлениях.

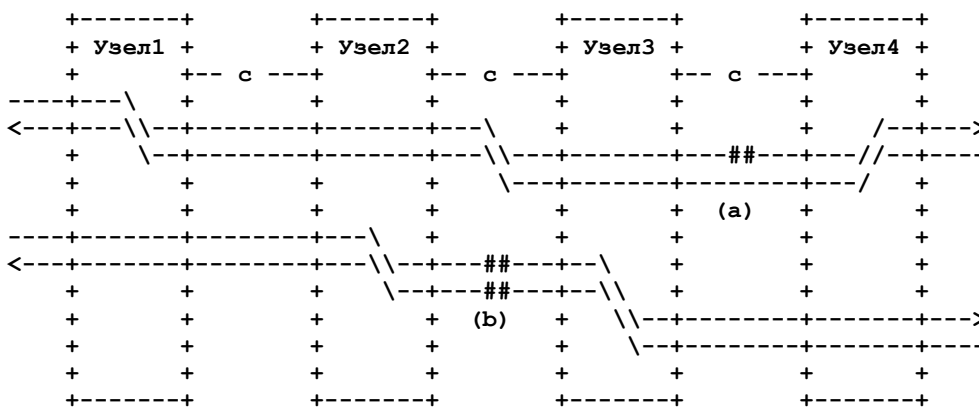


Рисунок 2. Показаны 2 типа отказов на каналах данных (символы ##):

- (A) канал данных нисходящего направления в двухстороннем LSP,
 (B) два канала данных, соответствующих разным направлениям двухстороннего LSP.
 Каналы управления между узлами помечены буквой с.

6.4. Индикация активации канала

Сообщения ChannelStatus могут служить также для уведомления соседа LMP о том, что для канала данных следует вести активный мониторинг. Это называется индикацией активации канала (Channel Activation Indication). Особенно полезно это в сетях с прозрачными узлами, где может потребоваться переключение состояния канала данных с помощью управляющих сообщений. Например, если канал данных подготовлен заранее, а на физическом канале возникает отказ после проверки канала но до начала отправки туда пользовательского трафика, нужен механизм индикации того, что активность канала сохраняется, поскольку иначе отказ может остаться не обнаруженным.

Сообщение ChannelStatus используется для индикации канала или группы каналов, которые в настоящий момент активны. В ответ на такое сообщение **должно** передаваться сообщение ChannelStatusAck. При получении ChannelStatus соответствующий канал(ы) данных **должен** переводиться в состояние Active. Если при переводе канала в активное состояние обнаруживается отказ, **следует** передать сообщение ChannelStatus, как описано в параграфе 6.2.

6.5. Индикация деактивации канала

Сообщения ChannelStatus могут служить также для уведомления соседа LMP о том, что для канала данных больше не следует вести активный мониторинг. Это сообщение по смыслу обратно описанному в предыдущем параграфе.

При получении сообщения с Channel Deactive Indication соответствующий канал(ы) данных должен выйти из активного состояния.

7. Использование идентификаторов сообщений

Объекты MESSAGE_ID и MESSAGE_ID_ACK включаются в сообщения LMP для обеспечения гарантированной доставки сообщений. Применение таких объектов рассматривается в данном разделе. Объекты MESSAGE_ID и MESSAGE_ID_ACK содержат поле Message_Id.

В любом сообщении LMP может присутствовать лишь один объект MESSAGE_ID/MESSAGE_ID_ACK.

Для сообщений, связанных с каналами управления, поле Message_Id относится к области CC_Id, для сообщений, относящихся к каналу TE, Message_Id относится к области смежности LMP.

Поле Message_Id в объекте MESSAGE_ID содержит выбранное генератором значение. Эти значения **должны** монотонно возрастать. Значение считается ранее использованным, если оно было передано в сообщении LMP с таким же CC_Id (для относящихся к каналу управления сообщений) или смежностью LMP (для сообщения, относящихся к TE Link). Поле Message_Id в сообщении MESSAGE_ID_ACK содержит значение Message_Id из подтверждаемого сообщения.

Неподтвержденные сообщения с объектом MESSAGE_ID **следует** передавать повторно, пока оно не будет подтверждено, если число повторов не будет исчерпано раньше (см. раздел 10).

Отметим, что 32-битовое поле Message_Id может достигать верхней границы. Для проверки отношений между последним и предшествующим ему значениями Message_Id можно использовать приведенное ниже выражение:

```
Если ((int) old_id - (int) new_id > 0) {
    Новое значение меньше предыдущего;
}
```

Узлам, обрабатывающим входящие сообщения, **следует** проверять соответствие нового сообщения порядку доставки — сообщения, нарушающие порядок можно игнорировать. Нарушение порядка можно обнаружить путем проверки значений поля Message_Id. Нарушающие порядок доставки сообщения следует отбрасывать без уведомления отправителя.

Если получено сообщение Config и значение Message_Id в нем меньше наибольшего значения Message_Id, принятого от того же отправителя для CC_Id, это сообщение **следует** считать нарушающим порядок.

Если получено сообщение LinkSummary и значение Message_Id в нем меньше наибольшего значения Message_Id, принятого от того же отправителя для TE Link, это сообщение **следует** считать нарушающим порядок.

Если получено сообщение ChannelStatus с Message_Id меньше наибольшего полученного ранее от того же отправителя Message_Id для заданного канала TE, получателю **следует** проверить полученное ранее значение Message_Id для состояния каждого канала данных, указанного в сообщении ChannelStatus. Если значение Message_Id превышает наибольшее принятое раньше значение Message_Id по крайней мере для одного из каналов данных, указанных в сообщении, **недопустимо** считать сообщение нарушающим порядок доставки, в противном случае его **следует** считать нарушающим порядок. Однако **недопустимо** менять состояние любого канала данных, если значение Message_Id меньше полученного недавно Message_Id, связанного с каналом данных.

Все остальные сообщения **недопустимо** считать нарушающими порядок доставки.

8. Аккуратный перезапуск

В этом разделе описан механизм повторной синхронизации состояния LMP после перезапуска уровня управления. Такой перезапуск может быть обусловлен восстановлением первого канала управления после коммуникационного отказа. Сбой коммуникаций может быть результатом отказа LMP-смежности или проблемы на узле, когда состояние управления LMP теряется без воздействия на уровень передачи данных. Последнее обнаруживается установкой флага LMP Restart в общем заголовке (Common Header) сообщений LMP. При отказе на уровне управления в результате потери канала управления информация канала LMP следует сохранять. В некоторых случаях узел может оказаться способным сохранить информацию о состоянии канала LMP даже при отказе на этом узле. Однако в обоих случаях **должна** происходить повторная синхронизация состояний каналов данных.

Предполагается, что значения Node_Id и Local Interface_Id сохраняются при перезапуске уровня управления.

После перезапуска уровня управления на узле требуется заново организовать канал(ы) управления с использованием процедур из параграфа 3.1. При повторной организации каналов управления **следует** передавать сообщение Config с использованием индивидуальных (unicast) адресов IP для отправителя и получателя.

Если отказ уровня управления был результатом отказа узла, при котором состояние управления LMP было потеряно, в сообщениях LMP **должен** устанавливаться флаг LMP Restart, пока не будет получено сообщение Hello с RcvSeqNum = TxSeqNum (локальное значение). Это показывает активизацию канала управления и обнаружение перезапуска соседа LMP.

Далее предполагается, что перезапуск компоненты LMP происходит только на одной стороне канала TE. При перезапуске компоненты LMP на обеих сторонах TE Link следует использовать обычную процедуру LinkSummary, как описано в разделе 4.

После активизации канала управления сосед LMP **должен** передать сообщение LinkSummary для каждого TE Link через смежность. Для всех объектов в сообщении LinkSummary бит N **должен** быть сброшен (0) для индикации того, что параметры не согласуются. Это обеспечивает отображения между локальными/удаленными Link_Id и Interface_Id, параметры соответствующих каналов данных и индикацию каналов данных, выделенных для пользовательского

трафика. Когда узел получает сообщение LinkSummary, он проверяет локальную конфигурацию. Если узел способен сохранить информацию о канале LMP при перезапуске, он должен обработать сообщение LinkSummary в соответствии с разделом 4, но флаг выделения (allocated/de-allocated) в объекте DATA_LINK из принятого сообщения LinkSummary **должен** получать предпочтение по сравнению с любым локальным значением. Однако, если узел не способен сохранить информацию канала LMP при перезапуске, он **должен** воспринять параметры каналов данных из сообщения LinkSummary и ответить сообщением LinkSummaryAck.

По завершении обмена LinkSummary узлу, выполнявшему перезапуск управления, **следует** отправить сообщение ChannelStatusRequest для канала TE. Узлу **следует** также проверить связность всех не выделенных каналов.

9. Адресация

Все сообщения LMP передаются по протоколу UDP через порт LMP (за исключением в некоторых случаях сообщений Test, которые могут быть ограничены транспортным механизмом для передачи сообщений в основной полосе - in-band). Адресом получателя пакета IP **может** быть адрес, полученный в процедуре Configuration (т. е., адрес отправителя из заголовка пакета с сообщением Config), адрес IP, заданный в конфигурации удаленного узла или Node_Id. Сообщение Config является исключением, как отмечено выше.

Манера адресации для сообщений Config может зависеть от транспортного механизма сигнализации. При использовании канала «точка-точка» сообщения Config **следует** направлять по групповому адресу (224.0.0.1 или ff02::1). В остальных случаях сообщения Config **должны** направляться по IP-адресу соседнего узла. Этот адрес может задаваться в конфигурации на обеих сторонах канала управления или определяться автоматически.

10. Процедуры экспоненциального роста интервалов повтора

Этот раздел основан на [RFC2961] и описывает процедуры экспоненциального роста интервалов повторной передачи. Реализации **должны** использовать описанные процедуры или их эквивалент.

10.1. Работа

Ниже описан один из возможных механизмов экспоненциального увеличения интервалов повтора неподтвержденных сообщений LMP. Передающий узел повторяет сообщение, пока не будет получено подтверждение или достигнуто предельное число повторов. Получив подтверждение, передающий узел прекращает отправку повторов. Интервал между повторными сообщениями определяется таймером ускоренного повтора (rapid retransmission timer). Отсчет этого таймера начинается с небольшого значения, а потом интервал экспоненциально увеличивается вплоть до установленного максимума.

Ниже описаны параметры, которые будут полезны для управления процедурой.

Интервал ускоренного повтора - R_i

R_i определяет интервал повтора передачи неподтвержденного сообщения. После отправки сообщения передающий узел будет повторять передачу (если нет подтверждения) через R_i мсек.

Предел ускоренного повтора - R_l

R_l определяет максимальное число повторов передачи неподтвержденного сообщения.

Увеличение интервала повтора Δ

Параметр Δ задает скорость, с которой отправитель увеличивает интервал повтора передачи. Отношение между двумя последовательными интервалами определяется значением $(1 + \Delta)$.

По умолчанию предлагается использовать начальный интервал (R_i) в 500 мсек с удвоением каждого следующего интервала ($\Delta = 1$) и максимальным числом повторов 3.

10.2. Алгоритм повтора передачи

После передачи узлом сообщения, которое требует подтверждения, ему незамедлительно следует запланировать повтор передачи через интервал R_i . Если соответствующее подтверждение придет раньше, повтор передачи **следует** отменить. В противном случае сообщение передается заново по истечении интервала $(1 + \Delta) \cdot R_i^1$. Повтор передачи следует продолжать, пока не будет получено подтверждение или достигнуто предельное число повторов R_l .

Передающий узел может применять для повтора требующих подтверждения сообщений приведенный ниже алгоритм.

```

Перед исходной передачей устанавливаются значения  $R_k = R_i$  и  $R_n = 0$ .
пока ( $R_{n++} < R_l$ ) {
    передать сообщение;
    выждать  $R_k$  мсек;
     $R_k = R_k * (1 + \Delta)$ ;
}
/* подтверждение или отсутствие отклика от получателя за время  $R_l$  */
выполнить всю требуемую очистку;
выход;

```

Асинхронно, когда отправитель получает соответствующее подтверждение, он меняет значения счетчика повторов R_n на R_l .

Отметим, что передающий узел не анонсирует и не согласует использование описанной выше процедуры увеличения интервала в сообщениях Config или LinkSummary.

¹Так написано в оригинале, хотя это явно противоречит началу данного абзаца и приведенному ниже алгоритму. Указанное значение относится ко второму повтору, а не к первому, как написано в тексте. *Прим. перев.*

11. Машины конечных состояний LMP

11.1. FSM управляющего канала

Машина конечных состояний (FSM) канала управления определяет состояния и логику работы канала управления LMP.

11.1.1. Состояния управляющего канала

Канал управления может находиться в одном из перечисленных ниже состояний. Каждое состояние соответствует некоторым условиям в канале управления и обычно связано с конкретным типом сообщений LMP, которые периодически отправляются на удаленную сторону.

Down

Это начальное состояние канала управления. В этом состоянии не предпринимается попыток активизации канала управления и передачи сообщений. Для параметров канала управления следует установить начальные значения.

ConfSnd

Канал находится в состоянии согласования параметров. В этом состоянии узел периодически отправляет сообщения Config, ожидая от удаленной стороны ответа в форме сообщения ConfigAck или ConfigNack. FSM не переходит в активное (Active) состояние, пока удаленная сторона не подтвердит параметры.

ConfRcv

Канал находится в состоянии согласования параметров. В этом состоянии узел ждет подходящих конфигурационных параметров от удаленной стороны. После получения и подтверждения этих параметров FSM может переходить в состояние Active.

Active

В этом состоянии узел периодически отправляет сообщения Hello и ждет в ответ приемлемых сообщений Hello. После приема приемлемого сообщения Hello канал может переходить в рабочее (Up) состояние.

Up

Канал управления находится в рабочем состоянии. Узел получает приемлемые сообщения Hello и передает свои сообщения Hello.

GoingDown

Канал управления может перейти в это состояние в результате действий администратора. Пока канал находится в этом состоянии, узел устанавливает бит ControlChannelDown во всех отправляемых сообщениях.

11.1.2. События канала управления

Работа канала управления LMP описывается состояниями и событиями FSM. События порождаются нижележащими протоколами и программными модулями, а также процедурами обработки пакетов и FSM соответствующих каналов TE. Каждое событие имеет номер и символьное имя. Описания событий канала управления приведены ниже.

- 1: evBringUp - вызванное внешними обстоятельствами событие, показывающее, что следует начать согласование канала управления. Это событие может быть вызвано, например, успешным завершением процедуры организации (bootstrap) или настройкой конфигурации канала управления. В зависимости от конфигурации это будет вызывать одно из указанных ниже действий:
 - 1a) передача сообщения Config;
 - 1b) ожидание приема сообщения Config от удаленного узла.
- 2: EvCCDn - это событие возникает при наличии индикации недоступности канала управления.
- 3: EvConfDone - это событие показывает прием сообщения ConfigAck, подтверждающего параметры из Config.
- 4: EvConfErr - это событие показывает прием сообщения ConfigNack, отвергающего параметры из Config.
- 5: EvNewConfOK - новое сообщение Config было принято от соседа и подтверждено ConfigAck.
- 6: evNewConfErr - новое сообщение Config было принято от соседа и отвергнуто с передачей ConfigNack.
- 7: evContenWin - новое сообщение Config было принято от соседа и в тот же момент соседу было передано сообщение Config. Локальный узел «выиграл состязание» и принятое сообщение Config игнорируется.
- 8: evContenLost - новое сообщение Config было принято от соседа и в тот же момент этому соседу было передано сообщение Config. Локальный узел «проиграл состязание».
 - 8a) Сообщение Config подтверждено ConfigAck;
 - 8b) Сообщение Config отвергнуто ConfigNack.
- 9: EvAdminDown - администратор запросил отключить канал управления административно.
- 10: EvNbrGoesDn - от соседа получен пакет с флагом ControlChannelDown.
- 11: EvHelloRcvd - получен пакет Hello с ожидаемым значением SeqNum.
- 12: EvHoldTimer - завершился отсчет таймера HelloDeadInterval при отсутствии пакетов Hello, что переводит канал управления назад в состояние Negotiation и инициирует один из вариантов (в зависимости от конфигурации)
 - 12a) периодическая отправка сообщений Config;
 - 12b) ожидание сообщений Config от удаленного узла.
- 13: EvSeqNumErr - получено сообщение Hello с неожиданным SeqNum, сообщение отброшено.
- 14: EvReconfig - параметры канала управления были изменены и требуется новое согласование.
- 15: EvConfRet - завершился отсчет таймера повтора и сообщение Config передано снова.
- 16: evHelloRet - завершился отсчет таймера HelloInterval и сообщение Hello передано снова.

17: `evDownTimer` - завершился отсчет таймера, но не было получено ни одного сообщения с флагом `ControlChannelDown`.

11.1.3. Описание FSM канала управления

На рисунке 3 показана работа FSM канала управления в форме диаграммы состояний FSM.

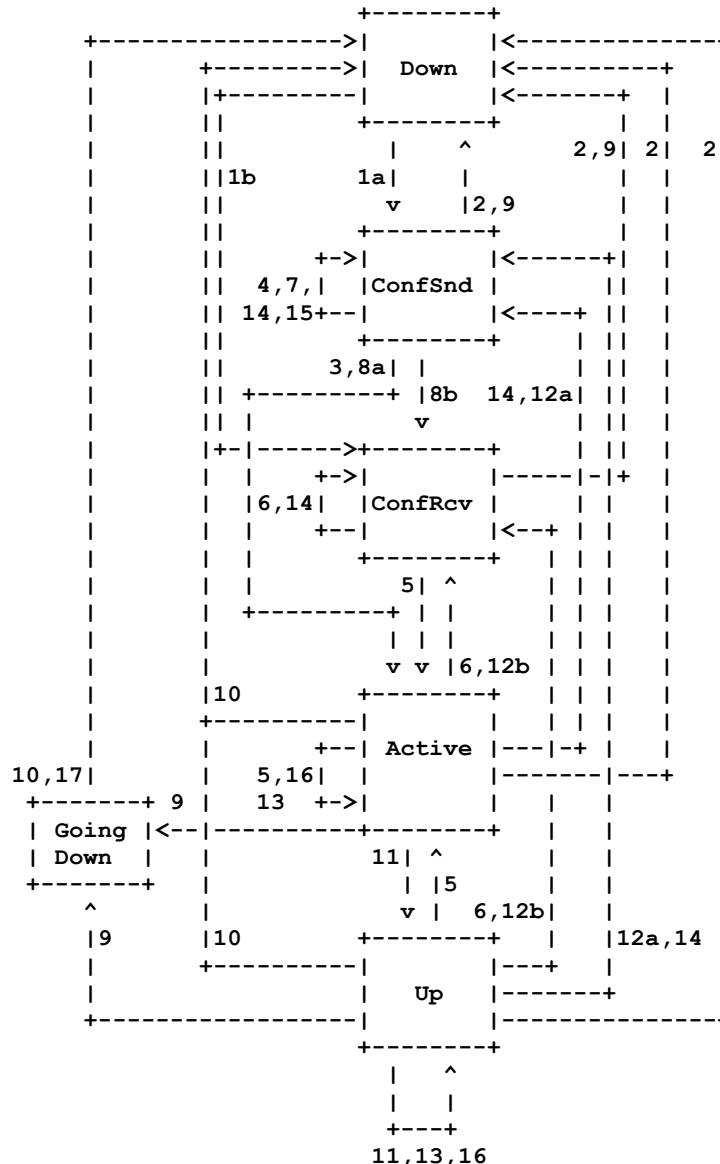


Рисунок 3. Диаграмма состояний FSM канала управления.

Событие `evCCDn` всегда переводит FSM в состояние `Down`, события `evHoldTimer` и `evReconfig` всегда переводят FSM в состояние `Negotiation` (`ConfSnd` или `ConfRcv`).

11.2. FSM канала TE

Машина конечных состояний `TE Link FSM` определяет состояния и логику работы `LMP TE Link`.

11.2.1. Состояния канала TE

Канал `TE` в `LMP` может находиться в одном из описанных ниже состояний. Каждое состояние соответствует некоторым условиям в канале `TE` и обычно связано с конкретными типами сообщений `LMP`, которые периодически передаются на удаленную сторону через соответствующий канал управления или в основной полосе через каналы данных.

- Down**
Для канала `TE` еще не выделено каналов данных.
- Init**
Для канала `TE` выделены каналы данных, но конфигурация еще не синхронизирована с соседом `LMP`. Соседу `LMP` периодически отправляются сообщения `LinkSummary`.
- Up**
Это нормальное рабочее состояние канала `TE`. Для работы между узлами, использующими канал `TE`, требуется организация хотя бы одного канала управления `LMP`. В процессе нормальной работы соседу `LMP` могут передаваться сообщения `LinkSummary`, генерируемые периодически или по внешним запросам.
- Degraded**
В этом состоянии управляющие каналы `LMP` не работают (`Down`), но канал `TE` продолжает включать некоторые каналы данных, выделенные для пользовательского трафика.

11.2.2. События каналов TE

Работа канала LMP TE описывается состояниями и событиями FSM. События канала TE порождаются программами обработки пакетов и машинами FSM соответствующих каналов управления и каналов данных. Каждое событие имеет номер и символическое имя, приведенные ниже.

- 1: EvDCUp - один или множество каналов данных включены и связаны с каналом TE.
- 2: EvSumAck - получено и подтверждено сообщение LinkSummary.
- 3: EvSumNack - получено и отвергнуто сообщение LinkSummary.
- 4: evRcvAck - получено сообщение LinkSummaryAck, подтверждающее конфигурацию TE Link.
- 5: evRcvNack - получено сообщение LinkSummaryNack.
- 6: evSumRet - завершился отсчет таймера повтора и сообщение LinkSummary передано снова.
- 7: evCCUp - включен первый активный канал управления.
- 8: evCCDown - отключен последний активный канал управления.
- 9: evDCDown - удален последний канал данных TE Link.

11.2.3. Описание машины конечных состояний канала TE

На рисунке 4 показана работа машины конечных состояний LMP TE Link в форме диаграммы смены состояний FSM.

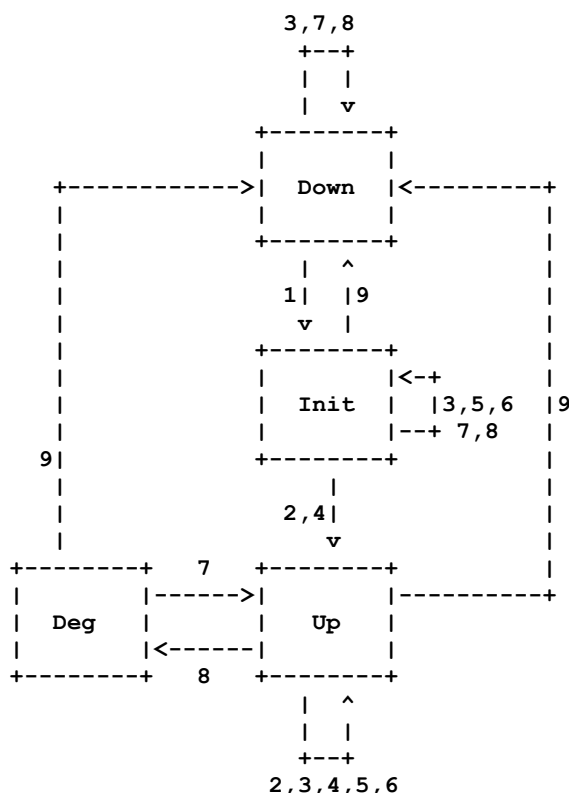


Рисунок 4. Диаграмма состояний FSM канала LMP TE.

На рисунке 4 субсостояния, возникающие в процессе выполнения процедуры проверки каналов, опущены.

11.3. FSM канала данных

FSM определяет состояния и логику работы канала данных в рамках канала LMP TE. Работа канала данных описывается состояниями и событиями FSM. Каналы данных могут находиться в активном (передача), когда через них передаются сообщения Test, или в пассивном (прием), когда сообщения Test передаются через них. Для обеспечения ясности приведены две FSM для активного и пассивного режима, а набор состояний и событий определен один.

11.3.1. Состояния канала данных

Любой канал данных может находиться в одном из описанных ниже состояний, каждое из которых соответствует неким условиям на канале.

Down

Канал данных не включен в пул ресурсов (т. е., канал не обслуживается).

Test

Канал данных тестируется и через него периодически передаются сообщения LMP Test.

PasvTest

Канал данных был проверен входящими тестовыми сообщениями.

Up/Free

Канал данных успешно протестирован и сейчас включен в пул ресурсов (обслуживается). Однако канал еще не выделен для передачи трафика.

Up/Alloc

Канал выделен для передачи трафика.

11.3.2. События канала данных

События канала данных создаются программами обработки пакетов и FSM связанного канала управления и канала TE.

Каждое событие имеет номер и символическое имя, представленные ниже.

- 1: evCCUp — поднят (up) первый активный канал управления.
- 2: evCCDown — потеряна связность с соседом LMP. Это говорит об отказе на последнем канале управления LMP между соседними узлами.
- 3: evStartTst — внешнее событие, включающее передачу сообщений Test через канал данных.
- 4: evStartPsv — внешнее событие, включающее прослушивание (прием) сообщений Test через канал данных.
- 5: evTestOK — проверка канала завершилась успехом и этот канал может использоваться для организации пути.
 - (a) Это событие говорит об успешном завершении процедуры Link Verification (раздел 5) для данного канала и получении через канал управления сообщения TestStatusSuccess.
 - (b) Это событие показывает готовность канала для организации пути, но процедура Link Verification не использовалась. Для сигнализации в основной полосе канала управления организации этого канала может быть достаточно для проверки канала.
- 6: evTestRcv — было получено сообщение Test через порт данных и передано сообщение TestStatusSuccess в канал управления.
- 7: evTestFail — отрицательный результат процедуры проверки канала, который мог быть обусловлен (a) получением сообщения TestStatusFailure или (b) завершением процедуры проверки без приема сообщения TestStatusSuccess или TestStatusFailure для канала данных.
- 8: evPsvTestFail — отрицательный результат процедуры проверки канала, который показывает, что сообщение Test не было обнаружено и (a) время VerifyDeadInterval истекло или (b) процедура проверки завершилась до истечения времени VerifyDeadInterval.
- 9: evLnkAlloc — канал данных выделен.
- 10: evLnkDealloc — выделение канала данных отменено.
- 11: evTestRet — завершился отсчет таймера повтора и сообщение Test передано заново.
- 12: evSummaryFail - LinkSummary не соответствует порту данных.
- 13: evLocalizeFail — для этого канала данных был обнаружен (локализован) отказ.
- 14: evdcDown — канал данных больше не доступен.

11.3.3. Описание FSM активного канала данных

На рисунке 5 показана работа FSM активного канала данных LMP в форме диаграммы смены состояний FSM.

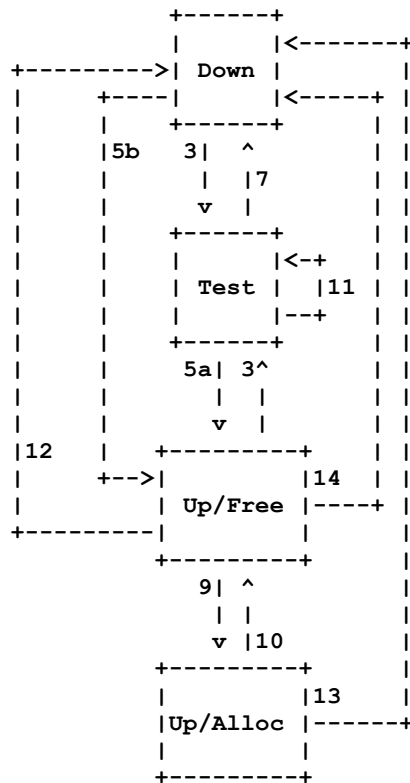


Рисунок 5. FSM активного канала данных.

11.3.4. Описание FSM пассивного канала данных

На рисунке 6 показана работа FSM пассивного канала данных LMP в форме диаграммы смены состояний FSM.

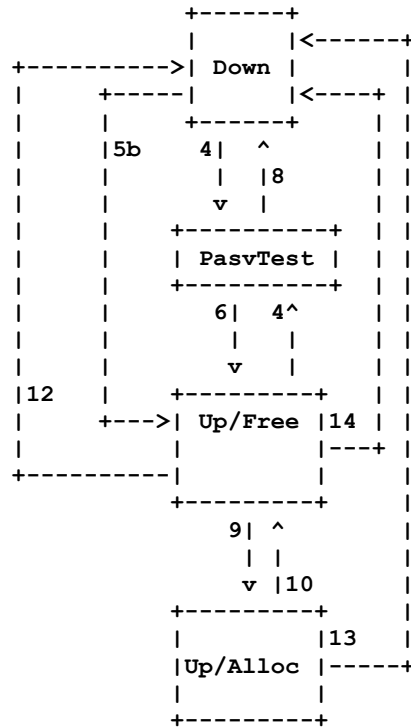


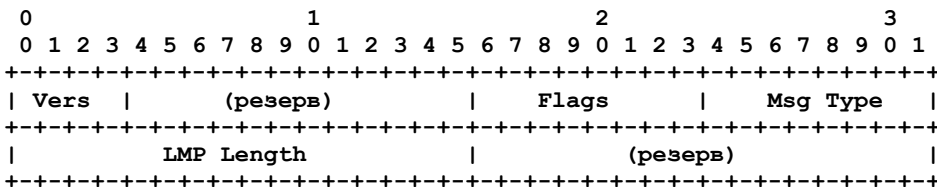
Рисунок 6. FSM пассивного канала данных.

12. Форматы сообщений LMP

Все сообщения LMP (за исключением в некоторых случаях сообщений Test, которые могут быть ограничены транспортным механизмом передачи сообщений по основному каналу) передаются по протоколу UDP через порт LMP (701).

12.1. Общий заголовок

В дополнение к заголовку UDP и стандартному заголовку IP все сообщения LMP (за исключением в некоторых случаях сообщений Test, которые могут быть ограничены транспортным механизмом передачи сообщений по основному каналу) используют показанный ниже формат заголовков.



Все резервные (Reserved) поля следует устанавливать в 0 при передаче и игнорировать на приемной стороне.

Все значения указаны в сетевом порядке байтов (big-endian — сначала старший байт).

Vers (4 бита)

Номер версии протокола (1 для текущей версии).

Flags (8 битов)

Ниже приведены две определенных комбинации битов. Все остальные биты являются резервными и их следует устанавливать в 0 при передаче и игнорировать на приемной стороне.

0x01: ControlChannelDown

0x02: LMP Restart

Этот бит устанавливается для индикации отказа на узле и потери состояния управления LMP. Флаг может быть сброшен в 0 при получении сообщения Hello со значением RcvSeqNum, равным локальному TxSeqNum.

Msg Type (8 битов)

Ниже перечислены определенные значения, все остальные значения являются резервными.

- 1 = Config
- 2 = ConfigAck
- 3 = ConfigNack
- 4 = Hello
- 5 = BeginVerify
- 6 = BeginVerifyAck
- 7 = BeginVerifyNack
- 8 = EndVerify
- 9 = EndVerifyAck
- 10 = Test
- 11 = TestStatusSuccess
- 12 = TestStatusFailure
- 13 = TestStatusAck
- 14 = LinkSummary

15 = LinkSummaryAck
 16 = LinkSummaryNack
 17 = ChannelStatus
 18 = ChannelStatusAck
 19 = ChannelStatusRequest
 20 = ChannelStatusResponse

Все сообщения, передаются через канал управления, **за исключением** сообщений Test, которые передаются через тестируемый канал данных.

LMP Length (16 битов)

Общий размер данного сообщения LMP в байтах с учетом общего заголовка и последующих объектов переменного размера.

12.2. Формат объекта LMP

Сообщения LMP создаются с использованием объектов, каждый из которых идентифицируется классом (Object Class) и типом (Class-type). Каждый объект имеет имя, которое в данном документе всегда указывается заглавными буквами. Объекты LMP могут быть согласуемыми или несогласуемыми (указывается битом N в заголовке объекта). Согласуемые объекты могут использоваться для того, чтобы устройства могли согласовать некие значения. Несогласуемые объекты служат для анонсирования конкретных значений, которые не нужно или невозможно согласовать.

Все значения указываются в сетевом порядке байтов (т. е., big-endian).

Формат объекта LMP показан ниже.

```

      0                1                2                3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|N|  C-Type  |   Class   |           Length           |
+-----+-----+-----+-----+-----+-----+-----+
|                                     |                   |
//                               (содержимое объекта)                //
|                                     |                   |
+-----+-----+-----+-----+-----+-----+-----+

```

N (1 бит)

Флаг N указывает возможность согласования объекта (N=1).

C-Type (7 битов)

Значение Class-type, уникальное в данном Object Class. Значения типов определены в разделе 13.

Class (8 битов)

Поле Class указывает тип объекта. Каждый объект имеет имя, которое в данном документе всегда указывается заглавными буквами.

Length (16 битов)

Указывает размер объекта в байтах с учетом полей N, C-Type, Class и Length.

12.3. Сообщения при согласовании параметров

12.3.1. Сообщение Config (Msg Type = 1)

Сообщения Config используются в фазе согласования канала управления LMP. Содержимое сообщений Config создается с использованием объектов LMP. Формат сообщения Config показан ниже.

```
<Config Message> ::= <Common Header> <LOCAL_CCID> <MESSAGE_ID> <LOCAL_NODE_ID> <CONFIG>
```

Указанный выше порядок объектов **следует** соблюдать.

Объект MESSAGE_ID находится с зоне действия объекта LOCAL_CCID.

Сообщения Config **должны** передаваться периодически, пока не будет (1) получено сообщение ConfigAck или ConfigNack, (2) достигнуто предельное число повторов при отсутствии ConfigAck или ConfigNack или (3) получено сообщение Config от удаленного узла и «состояние» было проиграно (например, Node_Id удаленного узла больше локального Node_Id). Интервал повтора и предельное число попыток задаются в локальной конфигурации.

12.3.2. Сообщение ConfigAck (Msg Type = 2)

Сообщения ConfigAck служат для подтверждения приема сообщений Config и говорят о согласии со всеми предложенными в них параметрами.

```
<ConfigAck Message> ::= <Common Header> <LOCAL_CCID> <LOCAL_NODE_ID>
                           <REMOTE_CCID> <MESSAGE_ID_ACK> <REMOTE_NODE_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объектов REMOTE_CCID, MESSAGE_ID_ACK и REMOTE_NODE_ID **должно** копироваться из подтверждаемого сообщения Config.

12.3.3. Сообщение ConfigNack (Msg Type = 3)

Сообщения ConfigNack служат для подтверждения приема сообщений Config и говорят о несогласии с предложенными (несогласуемыми) параметрами или предлагают другие значения для согласуемых параметров. Параметры, по которым имеется согласие, **недопустимо** включать в сообщение ConfigNack Message. Формат сообщения ConfigNack показан ниже.

```
<ConfigNack Message> ::= <Common Header> <LOCAL_CCID> <LOCAL_NODE_ID> <REMOTE_CCID>
                           <MESSAGE_ID_ACK> <REMOTE_NODE_ID> <CONFIG>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объектов REMOTE_CCID, MESSAGE_ID_ACK и REMOTE_NODE_ID **должно** копироваться из отвергаемого сообщения Config.

В сообщении Config может оказаться множество неприемлемых параметров.

Если в сообщении ConfigNack включен объект CONFIG для согласуемых параметров, он **должен** содержать приемлемые значения этих параметров.

Если в сообщении ConfigNack включен объект CONFIG для несогласуемых параметров, он **должен** содержать копии из объектов CONFIG, полученных в сообщении Config.

Если полученное сообщение ConfigNack включает только согласуемые объекты CONFIG, в ответ **следует** передать новое сообщение SHOULD. В значениях объекта CONFIG в этом новом сообщении Config **следует** принимать во внимание приемлемые значения параметров из сообщения ConfigNack.

Если узел получает сообщение Config и распознает в нем объект CONFIG, но не распознает C-Type, в ответ **должно** быть передано сообщение ConfigNack, включающее неизвестный объект CONFIG.

12.4. Сообщение Hello (Msg Type = 4)

Формат сообщения Hello показан ниже.

```
<Hello Message> ::= <Common Header> <LOCAL_CCID> <HELLO>
```

Указанный выше порядок объектов **следует** соблюдать.

Сообщения Hello **должны** передаваться периодически, хотя бы 1 раз каждые HelloInterval мсек. Если в течение интервала HelloDeadInterval не было получено ни одного сообщения Hello, предполагается отказ канала управления.

12.5. Сообщения для проверки канала

12.5.1. Сообщение BeginVerify (Msg Type = 5)

Сообщение BeginVerify передается через канал управления и служит для инициирования процесса проверки канала. Формат сообщения показан ниже.

```
<BeginVerify Message> ::= <Common Header> <LOCAL_LINK_ID>
<MESSAGE_ID> [<REMOTE_LINK_ID>] <BEGIN_VERIFY>
```

Указанный выше порядок объектов **следует** соблюдать.

Для ограничения области действия Link Verification конкретным каналом TE Link поле Link_Id в объекте LOCAL_LINK_ID **должно** иметь ненулевое значение. Если это поле равно 0, каналы данных будут охватывать множество каналов TE и/или могут включать еще не организованный канал TE. В специальном случае установки локального значения Link_Id = 0 используется флаг Verify all Links объекта BEGIN_VERIFY для того, чтобы различить каналы данных, входящие во множество каналов TE, от тех каналов, которые еще не выделены для канала TE (см. раздел 5).

Если известно отображение между локальным и удаленным Link_Id, может включаться объект REMOTE_LINK_ID.

При включении объекта REMOTE_LINK_ID поле Link_Id в нем **должно** быть отлично от 0.

Сообщения BeginVerify **должны** передаваться периодически, пока (1) узел не получит сообщения BeginVerifyAck или BeginVerifyNack о восприятии или отказе от проверки или (2) не будет достигнуто предельное число повторов без BeginVerifyAck или BeginVerifyNack. Интервал и число повторов являются параметрами локальной конфигурации.

12.5.2. Сообщение BeginVerifyAck (Msg Type = 6)

При наличии готовности обрабатывать сообщения Test в ответ на прием сообщения BeginVerify **должно** передаваться сообщение BeginVerifyAck.

```
<BeginVerifyAck Message> ::= <Common Header> [<LOCAL_LINK_ID>]
<MESSAGE_ID_ACK> <BEGIN_VERIFY_ACK> <VERIFY_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

Если отображение между локальным и удаленным Link_Id известно или может быть определено из сообщения BeginVerify, может включаться объект LOCAL_LINK_ID.

При включении объекта LOCAL_LINK_ID поле Link_Id в нем **должно** быть отлично от 0.

Содержимое объекта MESSAGE_ID_ACK **должно** копироваться из подтверждаемого сообщения BeginVerify.

Объект VERIFY_ID содержит уникальное в рамках узла значение, выделяемое создателем сообщения BeginVerifyAck. Это значение служит для уникальной идентификации проверочных процессов (Verification) от множества соседей LMP и/или параллельных процедур Test между парой соседей LMP.

12.5.3. Сообщение BeginVerifyNack (Msg Type = 7)

При отсутствии готовности или желания обрабатывать сообщения Test в ответ на прием сообщения BeginVerify **должно** передаваться сообщение BeginVerifyNack.

```
<BeginVerifyNack Message> ::= <Common Header> [<LOCAL_LINK_ID>] <MESSAGE_ID_ACK> <ERROR_CODE>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объекта MESSAGE_ID_ACK **должно** копироваться из отвергаемого сообщения BeginVerify.

Если процесс проверки не поддерживается, поле ERROR_CODE **должно** указывать, что процедура Link Verification не поддерживается.

Если процесс проверки поддерживается, но узел не способен начать процедуру, поле ERROR_CODE **должно** указывать Unwilling to verify. При получении BeginVerifyNack с таким значением ERROR_CODE узлу, инициировавшему BeginVerify, **следует** запланировать повтор BeginVerify по истечении Rf секунд (Rf задается локальным параметром).

Если механизм Verification Transport не поддерживается, поле ERROR_CODE **должно** указывать Unsupported verification transport mechanism.

Если удаленная установка Link_Id не поддерживается и содержимое объекта REMOTE_LINK_ID (из сообщения BeginVerify) не соответствует ни одному из заданных в конфигурации значений, поле ERROR_CODE **должно** указывать ошибку конфигурации Link_Id.

Если узел, получивший сообщение BeginVerify, распознает объект BEGIN_VERIFY, но не распознает C-Type, поле ERROR_CODE **должно** указывать Unknown object C-Type.

12.5.4. Сообщение EndVerify (Msg Type = 8)

Сообщение EndVerify передается через канал управления и служит для прерывания процесса тестирования канала. Сообщение EndVerify может передаваться в любой момент узлом, желающим прервать процедуру Verify. Формат сообщения показан ниже.

```
<EndVerify Message> ::= <Common Header> <MESSAGE_ID> <VERIFY_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

Сообщения EndVerify передаются периодически, пока не будет (1) получено сообщение EndVerifyAck или (2) исчерпано число повторов без получения EndVerifyAck. Интервал повтора и число попыток задаются параметрами локальной конфигурации.

12.5.5. Сообщение EndVerifyAck (Msg Type = 9)

Сообщение EndVerifyAck передается через канал управления и служит для подтверждения окончания процесса проверки. Формат сообщения показан ниже.

```
<EndVerifyAck Message> ::= <Common Header> <MESSAGE_ID_ACK> <VERIFY_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объекта MESSAGE_ID_ACK **должно** быть получено из подтверждаемого сообщения EndVerify.

12.5.6. Сообщение Test (Msg Type = 10)

Сообщение Test передается через канал данных и служит для проверки физической связности канала. Если явно не указано иное, эти сообщения **должны** передаваться по протоколу UDP, как и все остальные сообщения LMP. Формат сообщения Test показан ниже.

```
<Test Message> ::= <Common Header> <LOCAL_INTERFACE_ID> <VERIFY_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

Отметим, что эти сообщения передаются через канал данных, а **не** канал управления. Механизм транспортировки сообщений Test согласуется с помощью поля Verify Transport Mechanism в объекте BEGIN_VERIFY и поля Verify Transport Response в объекте BEGIN_VERIFY_ACK (см. параграфы 13.8 и 13.9).

Локальный (передающий) узел передает данное сообщение Test периодически (по крайней мере один раз за каждые VerifyInterval мсек) в соответствующий канал данных, пока (1) не получит ответное сообщение TestStatusSuccess или TestStatusFailure по каналу управления от удаленного (отвечающего) узла или (2) не произойдет отказ на всех активных каналах управления между этими узлами. Удаленный узел будет передавать данное сообщение TestStatus через канал управления периодически, пока не получит соответствующее сообщение TestStatusAck или EndVerify.

12.5.7. Сообщение TestStatusSuccess (Msg Type = 11)

Сообщение TestStatusSuccess передается через канал управления и служит для передачи отображений между локальным Interface_Id и Interface_Id интерфейса, принявшего сообщение Test.

```
<TestStatusSuccess Message> ::= <Common Header> <LOCAL_LINK_ID> <MESSAGE_ID>  
                                     <LOCAL_INTERFACE_ID> <REMOTE_INTERFACE_ID> <VERIFY_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объекта REMOTE_INTERFACE_ID **должно** браться из подтверждаемого сообщения Test.

12.5.8. Сообщение TestStatusFailure (Msg Type = 12)

Сообщение TestStatusFailure передается через канал управления и служит для индикации того, что сообщение Test не было получено.

```
<TestStatusFailure Message> ::= <Common Header> <MESSAGE_ID> <VERIFY_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

12.5.9. Сообщение TestStatusAck (Msg Type = 13)

Сообщение TestStatusAck служит для подтверждения приема сообщения TestStatusSuccess или TestStatusFailure.

```
<TestStatusAck Message> ::= <Common Header> <MESSAGE_ID_ACK> <VERIFY_ID>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объекта MESSAGE_ID_ACK **должно** браться из подтверждаемого сообщения TestStatusSuccess или TestStatusFailure.

12.6. Сообщения о состоянии канала

12.6.1. Сообщение *LinkSummary* (Msg Type = 14)

Сообщения *LinkSummary* служат для синхронизации *Interface_Id* и сопоставления свойств каналов TE. Формат *LinkSummary* показан ниже.

```
<LinkSummary Message> ::= <Common Header> <MESSAGE_ID> <TE_LINK> <DATA_LINK> [<DATA_LINK>...]
```

Указанный выше порядок объектов **следует** соблюдать.

Обмен сообщениями *LinkSummary* может происходить в любой момент, когда на канале не выполняется процесс *Verification*. Сообщения *LinkSummary* **должны** передаваться периодически, пока (1) узел не получит ответного сообщения *LinkSummaryAck* или *LinkSummaryNack* или (2) не будет достигнуто предел повторов без получения *LinkSummaryAck* или *LinkSummaryNack*. Интервал передачи и число повторов задаются в локальной конфигурации.

12.6.2. Сообщение *LinkSummaryAck* (Msg Type = 15)

Сообщение *LinkSummaryAck* служит для индикации согласия на синхронизацию *Interface_Id* и восприятия/согласия всех параметров канала. Именно при получении этого сообщения локальный узел создает привязки *Link_Id*.

```
<LinkSummaryAck Message> ::= <Common Header> <MESSAGE_ID_ACK>
```

Указанный выше порядок объектов **следует** соблюдать.

12.6.3. Сообщение *LinkSummaryNack* (Msg Type = 16)

Сообщение *LinkSummaryNack* служит для индикации несогласия с несогласуемыми (non-negotiated) предложенными параметрами или предложением других значений согласуемых параметров. Параметры, со значениями которых узел согласен, **недопустимо** включать в сообщение *LinkSummaryNack*.

```
<LinkSummaryNack Message> ::= <Common Header> <MESSAGE_ID_ACK> <ERROR_CODE> [<DATA_LINK>...]
```

Указанный выше порядок объектов **следует** соблюдать.

Объект данных *DATA_LINK* **должен** включать подходящие значения всех согласуемых параметров. Если *LinkSummaryNack* включает объекты *DATA_LINK* для несогласуемых параметров, они **должны** копироваться из объектов *DATA_LINK* в принятом сообщении *LinkSummary*.

Если полученное сообщение *LinkSummaryNack* включает лишь согласуемые параметры, в ответ **следует** отправить новое сообщение *LinkSummary*. Значения параметров из принятого сообщения *LinkSummary* **следует** принимать во внимание при включении подходящих параметров в новое сообщение *LinkSummaryNack*.

Если получено сообщение *LinkSummary* с неприемлемыми несогласуемыми параметрами, поле *ERROR_CODE* **должно** указывать Unacceptable non-negotiable LINK_SUMMARY parameters.

Если получено сообщение *LinkSummary* с неприемлемыми согласуемыми параметрами, поле *ERROR_CODE* **должно** указывать Renegotiate LINK_SUMMARY parameters.

Если получено сообщение *LinkSummary* с непригодным объектом *TE_LINK*, поле *ERROR_CODE* **должно** указывать Invalid TE_LINK object.

Если получено сообщение *LinkSummary* с непригодным объектом *DATA_LINK*, поле *ERROR_CODE* **должно** указывать Invalid DATA_LINK object."

Если получено сообщение *LinkSummary* с объектом *TE_LINK*, но тип C-Типе не известен, поле *ERROR_CODE* **должно** указывать Unknown TE_LINK object C-Типе.

Если получено сообщение *LinkSummary* с объектом *DATA_LINK*, но тип C-Типе не известен, поле *ERROR_CODE* **должно** указывать Unknown DATA_LINK object C-Типе.

12.7. Сообщения контроля отказов

12.7.1. Сообщение *ChannelStatus* (Msg Type = 17)

Сообщение *ChannelStatus* передается через канал управления и служит для уведомления соседа LMP о состоянии канала данных. Получивший *ChannelStatus* узел **должен** ответить сообщением *ChannelStatusAck*. Формат *ChannelStatus* показан ниже.

```
<ChannelStatus Message> ::= <Common Header> <LOCAL_LINK_ID> <MESSAGE_ID> <CHANNEL_STATUS>
```

Указанный выше порядок объектов **следует** соблюдать.

Если объект *CHANNEL_STATUS* не содержит ни одного *Interface_Id*, это говорит об отказе канала TE целиком.

12.7.2. Сообщение *ChannelStatusAck* (Msg Type = 18)

Сообщение *ChannelStatusAck* служит для подтверждения приема *ChannelStatus*. Формат сообщения показан ниже.

```
<ChannelStatusAck Message> ::= <Common Header> <MESSAGE_ID_ACK>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объекта *MESSAGE_ID_ACK* **должно** копироваться из подтверждаемого сообщения *ChannelStatus*.

12.7.3. Сообщение *ChannelStatusRequest* (Msg Type = 19)

Сообщения *ChannelStatusRequest* передаются через канал управления для запроса данных о состоянии одного или множества каналов данных. Принявший *ChannelStatusRequest* узел **должен** ответить сообщением *ChannelStatusResponse*. Формат *ChannelStatusRequest* показан ниже.

```
<ChannelStatusRequest Message> ::= <Common Header> <LOCAL_LINK_ID>
                                     <MESSAGE_ID> [<CHANNEL_STATUS_REQUEST>]
```

Указанный выше порядок объектов **следует** соблюдать.

Если объект CHANNEL_STATUS_REQUEST не включен в сообщение, ChannelStatusRequest будет служить запросом состояния **всех** каналов данных в TE Link.

12.7.4. Сообщение ChannelStatusResponse (Msg Type = 20)

Сообщение ChannelStatusResponse служит для подтверждения приема ChannelStatusRequest и уведомления соседа LMP о состоянии каналов данных. Формат сообщения показан ниже.

```
<ChannelStatusResponse Message> ::= <Common Header> <MESSAGE_ID_ACK> <CHANNEL_STATUS>
```

Указанный выше порядок объектов **следует** соблюдать.

Содержимое объекта MESSAGE_ID_ACK **должно** копироваться из подтверждаемого сообщения ChannelStatusRequest.

13. Определения объектов LMP

13.1. Класс CCID (Control Channel ID)

Class = 1

C-Type = 1, LOCAL_CCID

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     CC_Id                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
CC_Id - 32 бита
```

Это значение **должно** отличаться от 0 и быть уникальным в масштабе узла. CC_Id указывает канал управления, который отправитель связывает с сообщением.

Объект не согласуется.

C-Type = 2, REMOTE_CCID

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     CC_Id                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
CC_Id - 32 бита
```

Это значение указывает CC_Id удаленного узла и **должен** отличаться от 0.

Объект не согласуется.

13.2. Класс NODE_ID

Class = 2

C-Type = 1, LOCAL_NODE_ID

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Node_Id (4 байта)                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
Node_Id
```

Это значение указывает узел, создавший пакет LMP.

Объект не согласуется.

C-Type = 2, REMOTE_NODE_ID

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Node_Id (4 байта)                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
Node_Id
```

Это значение указывает удаленный узел.

Объект не согласуется.

13.3. Класс LINK_ID

Class = 3

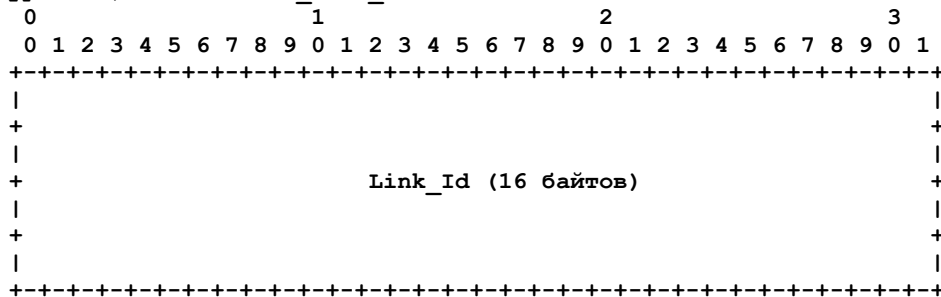
C-Type = 1, IPv4 LOCAL_LINK_ID

C-Type = 2, IPv4 REMOTE_LINK_ID

```
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Link_Id (4 байта)                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
Link_Id
```

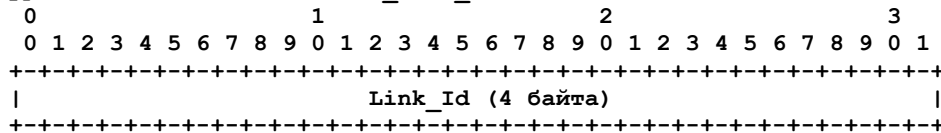
C-Type = 3, IPv6 LOCAL_LINK_ID

C-Type = 4, IPv6 REMOTE_LINK_ID



C-Type = 5, Unnumbered LOCAL_LINK_ID

C-Type = 6, Unnumbered REMOTE_LINK_ID



Link_Id

Для LOCAL_LINK_ID указывает канал отправителя, связанный с сообщением, и **должно** быть отлично от 0.

Для REMOTE_LINK_ID указывает Link_Id удаленного узла и **должно** быть отлично от 0.

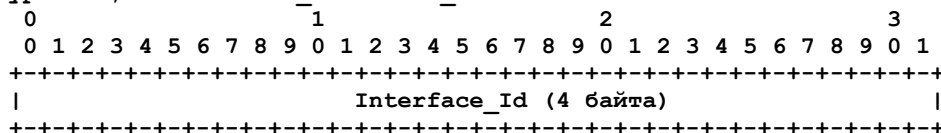
Объект не согласуется.

13.4. Класс INTERFACE_ID

Class = 4

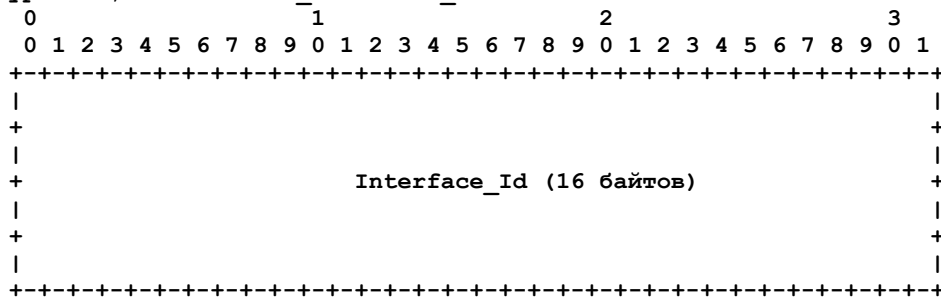
C-Type = 1, IPv4 LOCAL_INTERFACE_ID

C-Type = 2, IPv4 REMOTE_INTERFACE_ID



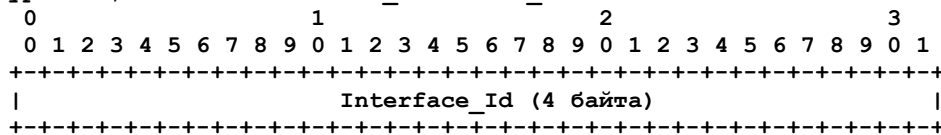
C-Type = 3, IPv6 LOCAL_INTERFACE_ID

C-Type = 4, IPv6 REMOTE_INTERFACE_ID



C-Type = 5, Unnumbered LOCAL_INTERFACE_ID

C-Type = 6, Unnumbered REMOTE_INTERFACE_ID



Interface_Id

Для LOCAL_INTERFACE_ID это значение указывает канал данных, оно **должно** отличаться от 0 и быть уникальным в масштабе узла.

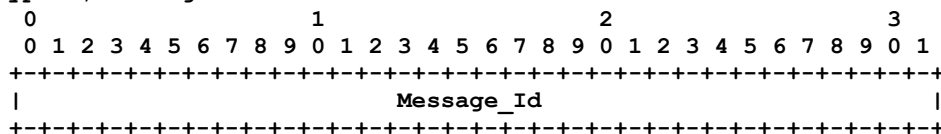
Для REMOTE_INTERFACE_ID это значение указывает канал данных на удаленном узле и **должно** отличаться от 0.

Объект не согласуется.

13.5. Класс MESSAGE_ID

Class = 5

C-Type=1, MessageId



Message_Id

Поле Message_Id служит для идентификации сообщения. Значение поля инкрементируется и может уменьшаться только при достижении максимума. Идентификаторы сообщений используются для подтверждения.

Объект не согласуется.

C-Type = 2, MessageIdAck

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     Message_Id                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Message_Id

Поле Message_Id field служит для идентификации подтверждаемого сообщения и копируется из объекта MESSAGE_ID в этом сообщении.

Объект не согласуется.

13.6. Класс CONFIG

Class = 6.

C-Type = 1, HelloConfig

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|           HelloInterval           |           HelloDeadInterval           |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

HelloInterval - 16 битов

Указывает интервал между передачей последовательных пакетов Hello в миллисекундах.

HelloDeadInterval - 16 битов

Если в течение интервала HelloDeadInterval не было принято пакетов Hello, предполагается отказ канала управления. Значение HelloDeadInterval указывается в миллисекундах. Значение HelloDeadInterval **должно** быть больше HelloInterval и его **следует** делать не меньше $3 * \text{HelloInterval}$.

Если в LMP не применяется ускоренный механизм keep-alive, в полях HelloInterval и HelloDeadInterval **должно** быть установлено значение 0.

13.7. Класс HELLO

Class = 7

C-Type = 1, Hello

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     TxSeqNum                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                     RcvSeqNum                                     |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

TxSeqNum - 32 бита

Это поле указывает текущий порядковый номер для данного сообщения Hello. Это значение будет инкрементироваться, когда порядковый номер будет отражен поле RcvSeqNum пакета Hello, полученного через канал управления.

Значение TxSeqNum=0 не разрешается, TxSeqNum=1 служит для обозначения первого сообщения Hello, передаваемого через канал управления.

RcvSeqNum - 32 бита

Указывает порядковый номер последнего сообщения Hello, принятого через канал управления. Значение RcvSeqNum=0 служит для обозначения того, что сообщений Hello еще не было получено.

Объект не согласуется.

13.8. Класс BEGIN_VERIFY

Class = 8

C-Type = 1

```

0                               1                               2                               3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Flags           |           VerifyInterval           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Number of Data Links           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           EncType           | (резерв) |           Verify Transport Mechanism           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           TransmissionRate           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|           Wavelength           |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Резервные поля следует устанавливать в 0 при передаче и игнорировать при получении.

Flags - 16 битов

Определенные в настоящее время флаги перечислены ниже.

0x0001 Verify all Links — проверить все каналы

Установленный флаг задает проверку всех не распределенных каналов, при сброшенном флаге проверяются лишь новые порты и компонентные соединения, добавленные в этот TE link.

0x0002 Data Link Type — тип канала данных

При установленном флаге будут проверяться порты, при сброшенном — компонентные соединения.

VerifyInterval - 16 бумос

Указывает интервал между последовательными сообщениями Test в миллисекундах.

Number of Data Links - 32 бума

Указывает число проверяемых каналов данных.

EncType - 8 бумос

Указывает тип кодирования для канала данных. Определенные значения EncType согласованы со значениями LSP Encoding Type из [RFC3471].

Verify Transport Mechanism - 16 бумос

Определяет транспортный механизм для сообщений Test. Область действия этой битовой маски ограничена для каждого типа кодирования. Локальный узел будет устанавливать биты, соответствующие разным механизмам, которые он может поддерживать для передачи тестовых сообщений LMP. Получатель выбирает подходящий механизм в сообщении BeginVerifyAck.

Ниже приведен флаг, определенный для всех типов кодирования, все остальные флаги зависят от Encoding Type.

0x8000 - Payload (тестовое сообщение будет передаваться в поле данных - payload)

Возможна передача сообщений Test в поле данных (payload). Сообщение Test передается в форме пакета IP, как описано выше.

TransmissionRate - 32 бума

Указывает скорость передачи канала данных, через который будут отправляться сообщения Test. Скорость указывается в бит/с с использованием формата IEEE floating-point.

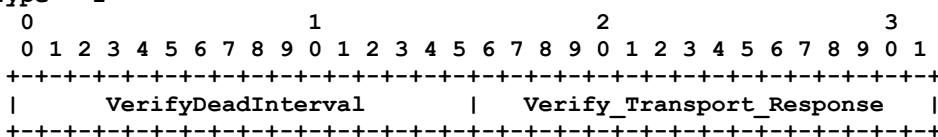
Wavelength - 32 бума

Когда канал данных связан с портом или компонентным каналом, способным передавать на разных длинах волн (т. е. оптический порт с поддержкой диапазона длин волн), важно знать, на какой длине волны будет передаваться тестовое сообщение. Это значение соответствует длине волны, используемой для передачи сообщений Test, и имеет локальную значимость. Если используемая для передачи тестовых сообщений длина волны определена однозначно, в этом поле **следует** указывать значение 0.

13.9. Класс BEGIN_VERIFY_ACK

Class = 9

C-Type = 1



VerifyDeadInterval - 16 бумос

Если сообщение Test не было обнаружено в течение интервала VerifyDeadInterval, узел будет передавать сообщение TestStatusFailure для этого канала данных.

Verify_Transport_Response - 16 бумос

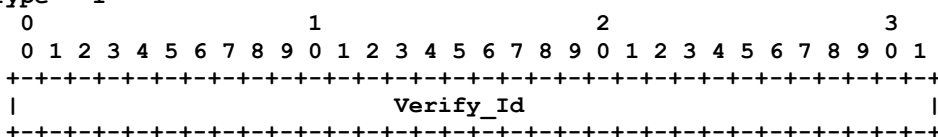
Получатель сообщения BeginVerify (и будущий получатель сообщений TEST) выбирает транспортный механизм из числа предложенных отправителем сообщений Test. В отклике для проверки транспорта **должен** устанавливаться один и только один бит.

Объект не согласуется.

13.10. Класс VERIFY_ID

Class = 10

C-Type = 1



Verify_Id - 32 бума

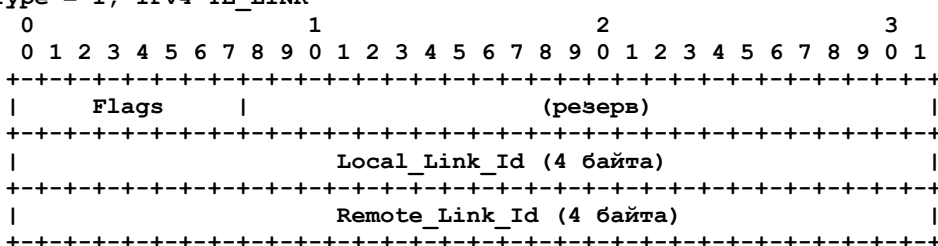
Это значение служит для того, чтобы отделить сообщения Test разных каналов TE и/или партнеров LMP. Значение уникально в масштабе узла и выделяется получателем сообщения BeginVerify.

Объект не согласуется.

13.11. Класс TE_LINK

Class = 11

C-Type = 1, IPv4 TE_LINK



```

C-Type = 2, IPv6 TE_LINK
  0           1           2           3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |  Flags    |                (резерв)                    |
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |
  |
  |           Local_Link_Id (16 байтов)
  |
  |
  |
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |
  |
  |           Remote_Link_Id (16 байтов)
  |
  |
  |
  +-----+-----+-----+-----+-----+-----+-----+-----+
C-Type = 3, Unnumbered TE_LINK
  0           1           2           3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |  Flags    |                (резерв)                    |
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |
  |           Local_Link_Id (4 байта)
  |
  |           Remote_Link_Id (4 байта)
  |
  +-----+-----+-----+-----+-----+-----+-----+-----+

```

Резервные поля следует устанавливать в 0 при передаче и игнорировать при получении.

Flags - 8 битов

Определенные к настоящему моменту флаги приведены ниже. Все остальные биты следует сбрасывать в 0 при передаче и игнорировать при получении.

0x01 — поддерживается контроль отказов:

0x02 — поддерживается проверка канала.

Local_Link_Id

Указывает Link_Id для локального узла и **должен** отличаться от 0.

Remote_Link_Id

Указывает Link_Id для удаленного узла и **должен** отличаться от 0.

13.12. Класс DATA_LINK

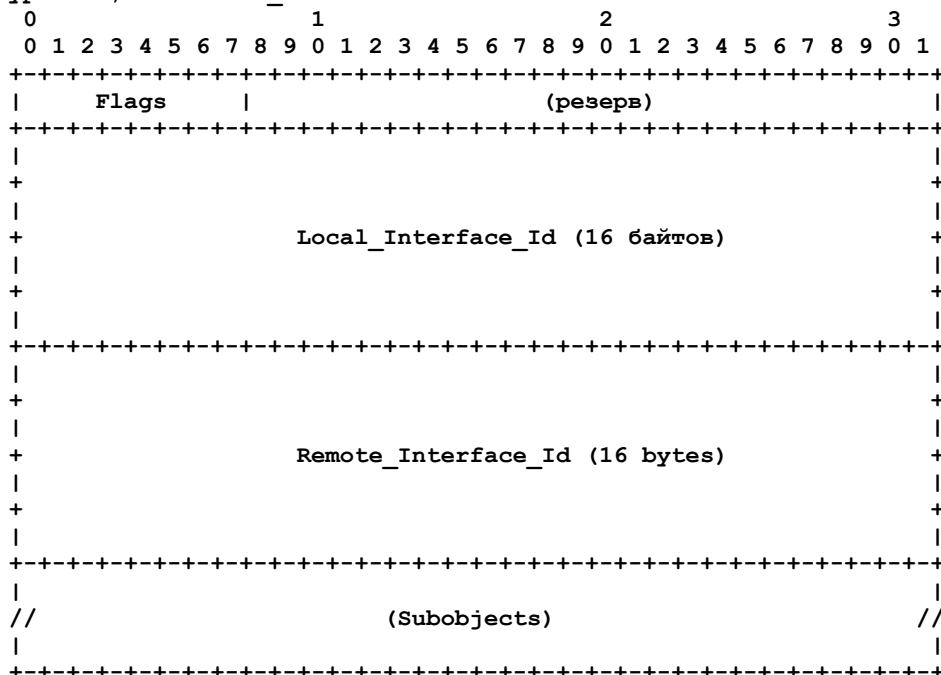
Class = 12

```

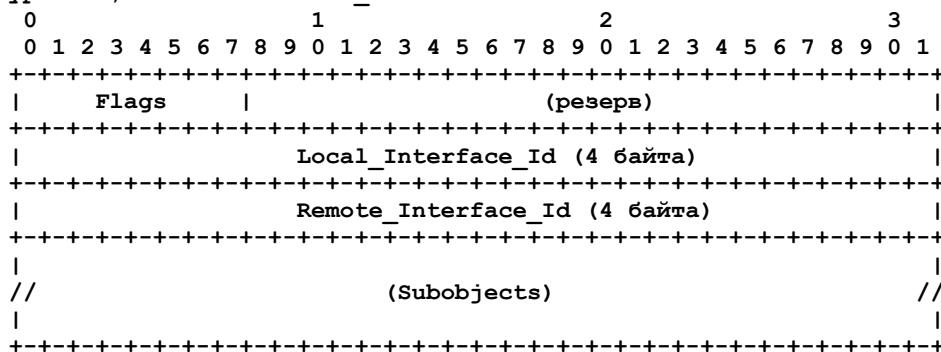
C-Type = 1, IPv4 DATA_LINK
  0           1           2           3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |  Flags    |                (резерв)                    |
  +-----+-----+-----+-----+-----+-----+-----+-----+
  |
  |           Local_Interface_Id (4 байта)
  |
  |           Remote_Interface_Id (4 байта)
  |
  |
  |           //          (Subobjects)              //
  |
  +-----+-----+-----+-----+-----+-----+-----+-----+

```

C-Type = 2, IPv6 DATA_LINK



C-Type = 3, Unnumbered DATA_LINK



Резервные поля следует устанавливать в 0 при передаче и игнорировать при получении.

Flags - 8 битов

Определенные к настоящему моменту флаги приведены ниже. Все остальные биты следует сбрасывать в 0 при передаче и игнорировать при получении.

- 0x01 — тип интерфейса; флаг устанавливается для порта и сбрасывается для компонентного канала;
- 0x02 — распределенный канал; установленный флаг говорит о выделении канала для пользовательского трафика; при использовании одного Interface_Id для приемного и передающего каналов данных этот бит относится лишь к передающему интерфейсу;
- 0x04 — отказавший канал; установленный флаг говорит об отказе канала и его непригодности для пользовательского трафика.

Local_Interface_Id

Локальный идентификатор канала данных. **Должен** быть уникальным в масштабе узла и отличным от 0.

Remote_Interface_Id

Удаленный идентификатор канала данных. **Должен** быть отличным от 0.

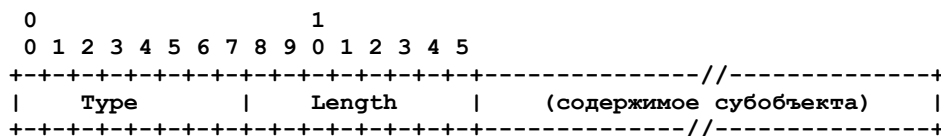
Subobjects

Содержимое объекта DATA_LINK включает последовательность элементов данных переменного размера, называемых субобъектами (определены ниже в параграфе 13.12.1).

Объект DATA_LINK может включать более одного субобъекта. Может присутствовать несколько субобъектов одного типа, если для канала поддерживается множество свойств (scalability).

13.12.1. Субобъекты канала данных

Содержимое объекта DATA_LINK включает последовательность элементов данных переменного размера, называемых субобъектами, форма которых показана ниже.



Type - 8 битов

Поле Type указывает тип содержимого субобъекта. В настоящее время определены:

- Type = 1, тип коммутации интерфейса;
- Type = 2, длина волны.

Length - 8 битов

Поле Length указывает размер субобъекта в байтах с учетом полей Type и Length. Значение поля Length **должно** быть кратно 4.

13.12.1.1. Субобъект типа 1 — тип коммутации интерфейса

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
Type										Length										Switching Type										EncType									
Minimum Reservable Bandwidth																																							
Maximum Reservable Bandwidth																																							

Switching Type - 8 битов

Служит для указания локального Interface Switching Type канала TE в соответствии с определением [RFC3471].

EncType - 8 битов

Тип кодирования канала данных. Определенные значения EncType совместимы с LSP Encoding Type [RFC3471].

Minimum Reservable Bandwidth - 32 бита

Максимальная резервируемая пропускная способность в бит/сек, указанная в формате IEEE floating point.

Maximum Reservable Bandwidth - 32 бита

Минимальная резервируемая пропускная способность в бит/сек, указанная в формате IEEE floating point.

Если интерфейс поддерживает лишь одну скорость, максимальное и минимальное значения совпадают.

13.12.1.2. Субобъект типа 2 — длина волны

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
Type										Length										(резерв)																			
Wavelength																																							

Резервное поле следует устанавливать в 0 при передаче и игнорировать при получении.

Wavelength - 32 бита

Указывает значение длины волны для данного порта. Это поле имеет значимость только между парой соседей.

13.13. Класс CHANNEL_STATUS

Class = 13

C-Type = 1, IPv4 INTERFACE_ID

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
Interface_Id (4 байта)																																							
Channel Status																																							
:																																							
//																														//									
:																																							
Interface_Id (4 байта)																																							
Channel Status																																							

```

C-Type = 2, IPv6 INTERFACE_ID
  0                               1                               2                               3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                                                 |
+                                                                 +
|                                                                 |
+                                                                 +
|                               Interface_Id (16 байтов)          |
+                                                                 +
|                                                                 |
+-----+-----+-----+-----+-----+-----+-----+-----+
|A|D|                               Channel Status              |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               :                               |
//                               :                               //
|                               :                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                                                 |
+                                                                 +
|                                                                 |
+                                                                 +
|                               Interface_Id (16 байтов)          |
+                                                                 +
|                                                                 |
+-----+-----+-----+-----+-----+-----+-----+-----+
|A|D|                               Channel Status              |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

```

C-Type = 3, Unnumbered INTERFACE_ID
  0                               1                               2                               3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Interface_Id (4 байта)           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|A|D|                               Channel Status              |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               :                               |
//                               :                               //
|                               :                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Interface_Id (4 байта)           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|A|D|                               Channel_Status              |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Active (A) - 1 бум

Указывает, что Channel выделен для пользовательского трафика и для канала данных следует использовать активный мониторинг.

Direction (D) - 1 бум

Указывает направление (передача или прием) канала данных, заданного в объекте CHANNEL_STATUS. Установленный флаг говорит об использовании канала данных для передачи.

Channel_Status - 30 бумов

Указывает состояние канала данных. Определенные значения приведены ниже, все прочие остаются в резерве.

- 1 Signal Okay (OK) - канал в рабочем состоянии.
- 2 Signal Degrade (SD) - некритичная ошибка, вызванная превышением порога BER, заданного конфигурацией.
- 3 Signal Fail (SF) - серьезные ошибки, включая (но не ограничиваясь) потерю сигнала (LOS¹) или кадра (LOF²) и Line AIS.

Этот объект содержит один или множество идентификаторов Interface_Id, за которыми следует поле Channel_Status.

Для индикации состояния TE Link в целом **должно** указываться единственное значение Interface_Id = 0.

13.14. Класс CHANNEL_STATUS_REQUEST

Class = 14

```

C-Type = 1, IPv4 INTERFACE_ID
  0                               1                               2                               3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Interface_Id (4 байта)           |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               :                               |
//                               :                               //
|                               :                               |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                               Interface_Id (4 байта)           |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

¹Loss of signal.

²Loss of frame.

Этот объект содержит один или множество идентификаторов Interface_Id.

Поле Length этого объекта содержит значение $4 + 4N$ (в байтах), где N — число идентификаторов Interface_Id.

C-Type = 2, IPv6 INTERFACE_ID

```

      0          1          2          3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                             |
+                                             +
|                                             |
+               Interface_Id (16 байтов)       +
|                                             |
+                                             +
|                                             |
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                             |
//               :                               //
|               :                               |
+-----+-----+-----+-----+-----+-----+-----+
|                                             |
+                                             +
|                                             |
+               Interface_Id (16 байтов)       +
|                                             |
+                                             +
|                                             |
+-----+-----+-----+-----+-----+-----+-----+

```

Этот объект содержит один или множество идентификаторов Interface_Id.

Поле Length этого объекта содержит значение $4 + 16N$ (в байтах), где N — число идентификаторов Interface_Id.

C-Type = 3, Unnumbered INTERFACE_ID

```

      0          1          2          3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                             |
+               Interface_Id (4 байта)         +
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                             |
//               :                               //
|               :                               |
+-----+-----+-----+-----+-----+-----+-----+
|                                             |
+               Interface_Id (4 байта)         +
+-----+-----+-----+-----+-----+-----+-----+

```

Этот объект содержит один или множество идентификаторов Interface_Id.

Поле Length этого объекта содержит значение $4 + 4N$ (в байтах), где N — число идентификаторов Interface_Id.

Объект не согласуется.

13.15. Класс ERROR_CODE

Class = 20

C-Type = 1, BEGIN_VERIFY_ERROR

```

      0          1          2          3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
|                                             |
+               ERROR CODE                     +
+-----+-----+-----+-----+-----+-----+-----+

```

Приведенные ниже битовые значения определены для сетевого порядка байтов (big-endian):

- 0x01 = процедура Link Verification не поддерживается;
- 0x02 = нежелание проверять;
- 0x04 = не поддерживается транспортный механизм проверки;
- 0x08 = ошибка конфигурации Link_Id;
- 0x10 = неизвестный объект C-Type.

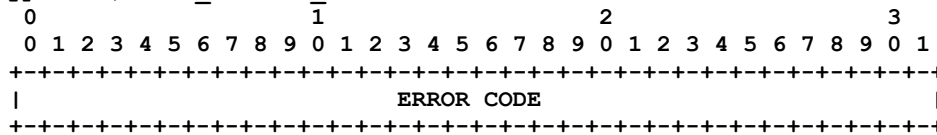
Все прочие битовые значения являются резервными, их следует устанавливать в 0 при передаче и игнорировать при получении.

Для индикации множества ошибок может быть установлено соответствующее множество битов.

Объект не согласуется.

Если сообщение BeginVerifyNack получено с Error Code 2, узлу-инициатору BeginVerify **следует** запланировать повторную передачу BeginVerify по истечении Rf (Rf является параметром локальной конфигурации).

C-Type = 2, LINK_SUMMARY_ERROR



Приведенные ниже битовые значения определены для сетевого порядка байтов (big-endian):

- 0x01 = неприемлемые несогласуемые параметры LINK_SUMMARY;
- 0x02 = повторное согласование параметров LINK_SUMMARY;
- 0x04 = непригодный объект TE_LINK;
- 0x08 = непригодный объект DATA_LINK;
- 0x10 = неизвестный объект TE_LINK C-Type;
- 0x20 = неизвестный объект DATA_LINK C-Type.

Все прочие битовые значения являются резервными, их следует устанавливать в 0 при передаче и игнорировать при получении.

Для индикации множества ошибок может быть установлено соответствующее множество битов.

Объект не согласуется.

14. Литература

14.1. Нормативные документы

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

[RFC4201] Kompella, K., Rekhter, Y., and L. Berger, "Link Bundling in MPLS Traffic Engineering (TE)", RFC 4201, October 2005.

[RFC4202] Kompella, K., Ed. and Y. Rekhter, Ed., "Routing Extensions in Support of Generalized Multi-Protocol Label Switching (GMPLS)", [RFC 4202](#), October 2005.

[RFC2961] Berger, L., Gan, D., Swallow, G., Pan, P., Tommasi, F., and S. Molendini, "RSVP Refresh Overhead Reduction Extensions", RFC 2961, April 2001.

[RFC2402] Kent, S. and R. Atkinson, "IP Authentication Header", RFC 2402¹, November 1998.

[RFC2406] Kent, S. and R. Atkinson, "IP Encapsulating Security Payload (ESP)", RFC 2406², November 1998.

[RFC2407] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407³, November 1998.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409³, November 1998.

[RFC3471] Berger, L., Ed., "Generalized MPLS — Signaling Functional Description", [RFC 3471](#), January 2003.

14.2. Дополнительная литература

[RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, September 2003.

[RFC3784] Smit, H. and T. Li, "Intermediate System to Intermediate System (IS-IS) Extensions for Traffic Engineering (TE)", RFC 3784, June 2004.

[RFC2401] Kent, S. and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401⁴, November 1998.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, [RFC 2434](#), October 1998.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

15. Вопросы безопасности

Существует множество атак, с которыми могут столкнуться протокольные сессии LMP. Ниже приведены несколько примеров:

- злоумышленник может использовать подставные пакеты управления;
- злоумышленник может изменить пакеты управления в пути передачи;
- злоумышленник может повторно использовать собранные пакеты управления;
- злоумышленник может изучить множество пакетов управления и попытаться взломать (подобрать) ключ, используя криптографические средства; если используемый алгоритм хеширования или шифрования имеет известные уязвимости, атакующему будет проще узнать ключ с использованием простых средств.

В этом разделе описан механизм защиты LMP на основе IPsec.

¹Этот документ признан устаревшим и заменен [RFC 4302](#) и [RFC 4305](#). Прим. перев.
²Этот документ признан устаревшим и заменен [RFC 4303](#) и [RFC 4305](#). Прим. перев.
³Этот документ признан устаревшим и заменен [RFC 4306](#). Прим. перев.
⁴Этот документ признан устаревшим и заменен [RFC 4301](#). Прим. перев.

15.1. Требования безопасности

Приведенные ниже требования относятся к описанному в этом разделе механизму.

- Защита LMP **должна** обеспечивать проверку подлинности, целостность и предотвращение повторного использования пакетов (replay).
- Для трафика LMP защита конфиденциальности не требуется. Нужна лишь проверка подлинности для обеспечения уверенности в том, что пакеты управления (пакеты, переданные через LMP Control Channel) исходят от нужного источника и не были изменены в процессе передачи. Для пакетов LMP Test, передаваемых через каналы данных, защита не требуется.
- Для трафика LMP защита отождествления конечных точек LMP в общем случае не требуется.
- Механизму защиты следует поддерживать четко определенные схемы управления ключами, которые следует проанализировать с точки зрения криптографии. Схемы управления ключами должны обеспечивать масштабирование и автоматизацию.
- Используемые для проверки подлинности алгоритмы **должны** быть криптостойкими. Кроме того, протокол защиты **должен** поддерживать использование разных алгоритмов аутентификации и возможность их согласования.

15.2. Механизмы защиты

Стек протоколов IPsec служит для защиты коммуникаций между парой партнеров на сетевом уровне. Этот стек протоколов описан в документах по архитектуре IP Security [RFC2401], IKE [RFC2409], IPsec AH [RFC2402] и IPsec ESP [RFC2406]. Протокол IKE обеспечивает управления ключами в сетях IP, а протоколы AH и ESP служат для защиты трафика IP. IKE определен применительно к протоколу IP.

С учетом требований, приведенных в параграфе 15.1, реализациям рекомендуется в тех случаях, когда требуется защита для LMP, использовать IPsec в соответствии с приведенным ниже описанием.

1. Реализациям LMP на основе протоколов IPsec **следует** поддерживать режим ручной установки ключей.

Этот режим обеспечивает простой способ организации и диагностики функциональности IPsec.

Однако следует отметить, что ручной режим не позволяет эффективно поддерживать такие функции, как защита от повторного использования пакетов (replay) и автоматическая смена ключей. Разработчикам, использующим ручной режим, следует принимать это во внимание.

Разработчикам рекомендуется использовать ручной режим только для диагностики и применять протокол динамической установки ключей для поддержки функций защиты от повторной и автоматической смены ключей.

2. **Должен** поддерживаться протокол IPsec ESP с трейлерной аутентификацией в туннельном режиме.
3. Реализации **должны** поддерживать протоколы аутентифицированного обмена ключами. **Должен** применяться протокол IKE [RFC2409] в качестве протокола обмена ключами, если партнеры согласовали динамическую установку ключей.
4. Реализации **должны** использовать IPsec DOI [RFC2407].
5. Для протокола IKE идентификаторы SA, согласованные в Quick Mode, представляют трафик, для которого партнеры согласовали защиту, и включают информацию о пространстве адресов, протоколе и портах.

При работе LMP на основе IPsec рекомендуется включать в поля данных идентификаторов для режима Quick перечисленную ниже информацию.

Идентификаторы **должны** иметь тип адресов IP, а в качестве значений идентификаторов **следует** указывать IP-адреса коммуникационных партнеров.

В поле протокола **должно** быть указано UDP. В поле порта **следует** указывать значение 0 для индикации того, что номера портов следует игнорировать. Это подразумевает, что весь трафик UDP между партнерами будет передаваться через туннель IPsec. Если реализация поддерживает селекторы по номерам портов, это позволяет сделать селектор более точным, указав поле порта LMP. Однако, если партнер не использует селекторов по номерам портов, реализация **должна** вернуться к использованию в поле порта значения 0.

6. **Должен** поддерживаться «агрессивный» режим согласования IKE.

Когда взаимодействие с партнером настроено для использования IPsec, предполагается, что все сообщения LMP будут передаваться через туннель IPsec (криптоканал). Точно так же получателю LMP, настроенному на работу с использованием IPsec, следует отвергать весь трафик LMP, пришедший не из криптоканала.

Криптоканал к соседу LMP может быть организован заранее или первое сообщение LMP, отправленное партнеру, инициирует создание туннеля IPsec.

Через один криптоканал может быть организовано множество каналов управления. При использовании сообщений LMP Hello для контроля состояния канала управления важно принимать во внимание, что отказ keep-alive в канале управления может быть следствием отказа в криптоканале. Ниже описан метод, рекомендуемый для обеспечения корректной работы коммуникационного пути LMP.

- Если LMP Hello говорит об отказе в канале управления, переключитесь на другой канал и/или попытайтесь организовать новый канал управления.
- Обеспечьте работоспособность каналов управления с помощью сообщений LMP Hello. Если на всех каналах управления зафиксированы отказы и нет возможности создать новый канал управления, отключите (удалите) все имеющиеся каналы управления, а также криптоканал (IKE SA и все IPsec SA).

- Организуйте криптоканал заново. Отказ при организации криптоканала служит индикатором невозможности коммуникаций LMP.
- Активизируйте канал управления. Отказ при организации канала управления служит индикатором невозможности коммуникаций LMP.

При динамическом обнаружении партнеров LMP (особенно инициаторов) следует принимать во внимание приведенные ниже замечания.

При проверке подлинности с заранее распространенным ключом в режиме защиты подлинности (основной режим) общий ключ требуется для расчета значения SKEYID (служит для создания ключей, используемых для шифрования сообщений в процессе обмена ключами). В основном режиме (main mode) IKE заранее известный общий ключ должен быть идентифицирован до получения данных отождествления партнера. Такой общий ключ требуется для расчета SKEYID. В этот момент единственной доступной информацией о партнере является адрес IP с которого выполняется согласование. Определение общего ключа по этому адресу не представляется возможным, поскольку адреса являются динамическими и не известны заранее.

Может использоваться агрессивный режим (aggressive mode) обмена ключами, поскольку идентификационные данные передаются в первом сообщении.

Однако следует отметить уязвимость агрессивного режима к пассивным атакам на службы. Использование общего секрета (групповой секрет) множеством партнеров настоятельно не рекомендуется, поскольку это открывает возможность для организации MITM¹-атак.

Проверка подлинности на основе цифровых подписей не сталкивается с такими проблемами. **рекомендуется** использовать такой механизм аутентификации в тех случаях, когда это возможно.

Если требуется проверка подлинности на основе заранее распространенного общего ключа (pre-shared key), **следует** применять агрессивный режим. Значения общих заранее распространенных ключей аутентификации IKE **следует** защищать, подобно паролям учетных записей пользователей.

16. Взаимодействие с IANA

Агентство IANA выделило номер 701 для порта LMP.

Ниже приведены рекомендации для IANA в части распределения значения в каждом пространстве имен LMP. Диапазоны распределяемых значений поделены на Private Use, Expert Review и Standards Action (в соответствии с [RFC2434]).

Распределение значений LMP из пространства Private Use (т. е., для частных расширений) не требуется документировать. Независимые реализации LMP, использующие одинаковые значения из пространства Private Use в общем случае не смогут взаимодействовать и следует принимать соответствующие меры при использовании таких значения в среде с продукцией разных производителей.

Распределение значений из пространств LMP по процедуре Expert Review осуществляется экспертами, назначенными IESG. Данный документ предполагает использование значений из этих диапазонов лишь для экспериментальных расширений и выделение значений **должно** сопровождаться RFC со статусом Experimental. Если такие расширения будут сочтены полезными для развертывания, их следует описать в RFC со статусом Standards Track и им **должны** быть выделены значения из диапазонов Standards Action.

Выделение значений из диапазонов LMP по процедуре Standards Action **должно** документироваться в Standards Track RFC, которые обычно подаются рабочими группами IETF и в любом случае должны соответствовать обычным процедурам IETF для документов со статусом Proposed Standards.

Резервные биты LMP Common Header следует распределять по процедуре Standards Action, в соответствии с правилами, описанными в [RFC2434].

LMP определяет несколько пространств имен, для которых требуется управление:

- LMP Message Type (тип сообщения);
- LMP Object Class (класс объекта);
- LMP Object Class type (C-Type) (тип класса объектов, который уникален в рамках Object Class);
- LMP Sub-object Class type (Type) (тип класса субобъектов, который уникален в рамках Object Class).

Пространство LMP Message Type следует распределять в соответствии с правилами, заданными в [RFC2434], - значения 0 - 127 распределяются по процедуре Standards Action, 128 - 240 по процедуре Expert Review, а 241 - 255 резервируются для Private Use.

Пространство LMP Object Class следует распределять в соответствии с правилами, заданными в [RFC2434], - значения 0 - 127 распределяются по процедуре Standards Action, 128 - 247 по процедуре Expert Review, а 248 - 255 резервируются для Private Use.

Правила распределения значений LMP Object Class являются частью определения конкретного класса. Определение Class должно включать описание правил выделения значений Object Class.

Правила распределения значений LMP Sub-object Class являются частью определения конкретного класса. Определение Class должно включать описание правил выделения значений для субобъектов.

Ниже перечислены пространства имен, выделенные IANA.

¹Man-in-the-middle — атака с перехватом и изменением данных в пути при участии человека.

Сообщение	Тип
Config	1
ConfigAck	2
ConfigNack	3
Hello	4
BeginVerify	5
BeginVerifyAck	6
BeginVerifyNack	7
EndVerify	8
EndVerifyAck	9
Test	10
TestStatusSuccess	11
TestStatusFailure	12
TestStatusAck	13
LinkSummary	14
LinkSummaryAck	15
LinkSummaryNack	16
ChannelStatus	17
ChannelStatusAck	18
ChannelStatusRequest	19
ChannelStatusResponse	20

Пространства имен LMP Object Class и Class type (C-Type)

CCID Class name (1)

Пространство типов CCID Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- LOCAL_CCID (C-Type = 1)
- REMOTE_CCID (C-Type = 2)

NODE_ID Class name (2)

Пространство типов NODE ID Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- LOCAL_NODE_ID (C-Type = 1)
- REMOTE_NODE_ID (C-Type = 2)

LINK_ID Class name (3)

Пространство типов LINK_ID Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- IPv4 LOCAL_LINK_ID (C-Type = 1)
- IPv4 REMOTE_LINK_ID (C-Type = 2)
- IPv6 LOCAL_LINK_ID (C-Type = 3)
- IPv6 REMOTE_LINK_ID (C-Type = 4)
- Unnumbered LOCAL_LINK_ID (C-Type = 5)
- Unnumbered REMOTE_LINK_ID (C-Type = 6)

INTERFACE_ID Class name (4)

Пространство типов INTERFACE_ID Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- IPv4 LOCAL_INTERFACE_ID (C-Type = 1)
- IPv4 REMOTE_INTERFACE_ID (C-Type = 2)
- IPv6 LOCAL_INTERFACE_ID (C-Type = 3)

- IPv6 REMOTE_INTERFACE_ID (C-Type = 4)
- Unnumbered LOCAL_INTERFACE_ID (C-Type = 5)
- Unnumbered REMOTE_INTERFACE_ID (C-Type = 6)

MESSAGE_ID Class name (5)

Пространство типов MESSAGE_ID Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- MESSAGE_ID (C-Type = 1)
- MESSAGE_ID_ACK (C-Type = 2)

CONFIG Class name (6)

Пространство типов CONFIG Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- HELLO_CONFIG (C-Type = 1)

HELLO Class name (7)

Пространство типов HELLO Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- HELLO (C-Type = 1)

BEGIN_VERIFY Class name (8)

Пространство типов BEGIN_VERIFY Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- Type 1 (C-Type = 1)

BEGIN_VERIFY_ACK Class name (9)

Пространство типов BEGIN_VERIFY_ACK Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- Type 1 (C-Type = 1)

VERIFY_ID Class name (10)

Пространство типов VERIFY_ID Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- Type 1 (C-Type = 1)

TE_LINK Class name (11)

Пространство типов TE_LINK Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- IPv4 TE_LINK (C-Type = 1)
- IPv6 TE_LINK (C-Type = 2)
- Unnumbered TE_LINK (C-Type = 3)

DATA_LINK Class name (12)

Пространство типов DATA_LINK Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- IPv4 DATA_LINK (C-Type = 1)
- IPv6 DATA_LINK (C-Type = 2)
- Unnumbered DATA_LINK (C-Type = 3)

Пространство имен DATA_LINK Sub-object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 127 выделяются по процедуре Standards Action, 128 - 247 по процедуре Expert Review, а 248 - 255 резервируются для Private Use.

- Interface Switching Type (sub-object Type = 1)
- Wavelength (sub-object Type = 2)

CHANNEL_STATUS Class name (13)

Пространство типов CHANNEL_STATUS Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- IPv4 INTERFACE_ID (C-Type = 1)
- IPv6 INTERFACE_ID (C-Type = 2)
- Unnumbered INTERFACE_ID (C-Type = 3)

CHANNEL_STATUS_REQUEST Class name (14)

Пространство типов CHANNEL_STATUS_REQUEST Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- IPv4 INTERFACE_ID (C-Type = 1)
- IPv6 INTERFACE_ID (C-Type = 2)
- Unnumbered INTERFACE_ID (C-Type = 3)

ERROR_CODE Class name (20)

Пространство типов ERROR_CODE Object Class следует распределять в соответствии с правилами [RFC2434] - значения 0 - 111 выделяются по процедуре Standards Action, 112 - 119 по процедуре Expert Review, а 120 - 127 резервируются для Private Use.

- BEGIN_VERIFY_ERROR (C-Type = 1)
- LINK_SUMMARY_ERROR (C-Type = 2)

17. Благодарности

Авторы благодарны Andre Fredette за большой вклад в этот документ. Спасибо также Ayan Banerjee, George Swallow, Adrian Farrel, Dimitri Papadimitriou, Vinay Ravuri и David Drysdale за их значимые комментарии и предложения. Спасибо John Yu, Suresh Katukam и Greg Bernstein за их полезные предложения в части применимости канала управления в основной полосе (in-band).

18. Участники работы

Jonathan P. Lang

Sonos, Inc.

223 E. De La Guerra St.

Santa Barbara, CA 93101

E-Mail: jplang@ieee.org

Krishna Mitra

Independent Consultant

E-Mail: kmitra@earthlink.net

John Drake

Calient Networks

5853 Rue Ferrari

San Jose, CA 95138

E-Mail: jdrake@calient.net

Kireeti Kompella

Juniper Networks, Inc.

1194 North Mathilda Avenue

Sunnyvale, CA 94089

E-Mail: kireeti@juniper.net

Yakov Rekhter

Juniper Networks, Inc.

1194 North Mathilda Avenue

Sunnyvale, CA 94089

EMail: yakov@juniper.net

Lou Berger

Movaz Networks

EMail: lberger@movaz.com

Debanjan Saha

IBM Watson Research Center

EMail: dsaha@us.ibm.com

Debashis Basak

Accelight Networks

70 Abele Road, Suite 1201

Bridgeville, PA 15017-3470

EMail: dbasak@accelight.com

Hal Sandick

Shepard M.S.

2401 Dakota Street

Durham, NC 27705

EMail: sandick@nc.rr.com

Alex Zinin

Alcatel

EMail: alex.zinin@alcatel.com

Bala Rajagopalan

Intel Corp.

2111 NE 25th Ave

Hillsboro, OR 97123

EMail: bala.rajagopalan@intel.com

Sankar Ramamoorthi

Juniper Networks, Inc.

1194 North Mathilda Avenue

Sunnyvale, CA 94089

EMail: sankarr@juniper.net

Информация для контактов

Jonathan P. Lang

Sonos, Inc.

829 De La Vina, Suite 220

Santa Barbara, CA 93101

EMail: jplang@ieee.org

Перевод на русский язык

Николай Малых

nmalykh@protocols.ru

Copyright (C) The Internet Society (2005).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Интеллектуальная собственность

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Подтверждение

Финансирование функций RFC Editor обеспечено Internet Society.