Network Working Group
Request for Comments: 1661
STD: 51
Obsoletes: 1548
Category: Standards Track

W. Simpson, Editor
Daydreamer
July 1994

# Point-to-Point Protocol (PPP)

# Протокол соединений «точка-точка»

**Статус документа**
Этот документ задает проект стандарта Интернет для сообщества Интернет и служит приглашением к дискуссии в целях совершенствования протокола. Информацию о состоянии стандартизации и текущем статусе протокола можно найти в «Internet Official Protocol Standards" (STD 1). Распространение этого документа не ограничивается.

**Тезисы**
Протокол PPP[1] обеспечивает стандартный метод для транспортировки дейтаграмм разных протоколов через соединения «точка-точка». PPP включает три основных компоненты, перечисленных ниже.

1. Метод инкапсуляции дейтаграмм разных протоколов.
2. Протокол управления соединением LCP[2].
3. Протоколы управления сетью (NCP[3]) для установки и настройки протоколов сетевого уровня.

Этот документ определяет организацию и методологию PPP, а также инкапсуляцию PPP и расширяемый механизм согласования опций, позволяющий использовать многочисленные конфигурационные параметры и набор дополнительных функций управления. Протокол управления каналом PPP (LCP) описан в терминах этого механизма.

# Содержание



[1]Point-to-Point Protocol – протокол соединений «точка-точка».
[2]Link Control Protocol.
[3]Network Control Protocol.

# 1. Введение

Протокол PPP разработан для простых линий связи, которые транспортируют пакеты между двумя узлами-партнерами. Эти соединения обеспечивают полнодуплексную двухстороннюю связь с сохранением порядка доставки пакетов. Предполагается, что PPP обеспечивает базовое решение для простого соединения различных хостов, мостов и маршрутизаторов [1].

***Инкапсуляция***

Инкапсуляция PPP обеспечивает мультиплексирование различных протоколов сетевого уровня через один канал. Эта инкапсуляция была разработана с учетом совместимости с наиболее распространенным оборудованием.

Для инкапсуляции применяется лишь 8 при использовании принятого по умолчанию кадрирования в стиле HDLC. В средах с дефицитной пропускной способностью издержки на инкапсуляцию и кадрирование могут быть сокращены до 2 или 4 октетов.

Для поддержки высокоскоростных реализаций принятая по умолчанию инкапсуляция использует лишь простые поля, из которых только одно нужно проверять для демультиплексирования. Используемый по умолчанию заголовок и информационные поля выравниваются по 32-битовым границам, а трейлер может дополняться до произвольной границы.

***Протокол LCP***

Для обеспечения достаточной универсальности и переносимости в разные среды протокол PPP включает протокол управления соединением (LCP). Этот протокол служит для автоматического согласования опций формата инкапсуляции, обработки ограничений на размеры пакетов, детектирования канальных петель и других конфигурационных ошибок, а также для разрыва соединений. Поддерживаются другие необязательные возможности для проверки подлинности отождествления партнера по каналу, контроля корректности работы соединения и обнаружения отказов.

***Протоколы управления сетью (NCP)***

Для соединений «точка-точка» характерна тенденция усугублять многие проблемы стека сетевых протоколов. Например, назначение и и поддержка адресов IP, которые являются проблемой в локальных сетях, дополнительно осложняются при работе по коммутируемым соединениям «точка-точка» (таким, как соединения с модемными пулами). Эти проблемы обрабатываются семейством протоколов управления NCP, которые решают конкретные задачи соответствующих протоколов сетевого уровня. Эти протоколы определены в сопутствующих документах.

***Настройка***

Исходным допущением была простота настройки каналов PPP. Протокол устроен так, что принятые по умолчанию значения подходят для большинства ситуаций. Разработчики могут улучшать принятую по умолчанию конфигурацию, о чем партнер будет проинформирован без участия оператора. Оператор может явно задать параметры соединения, что позволяет работать в средах, где иной подход невозможен.

Автоматическая настройка конфигурации реализована на основе расширяемого механизма согласования опций, где каждая сторона соединения описывает партнеру свои возможности и потребности. Хотя механизм согласования описан в этом документе в терминах LCP, те же средства разработаны для использования с другими протоколами управления, в частности NCP.

## 1.1 Соглашения

В этом документе некоторые слова служат для задания уровня требований. Такие слова выделяются **шрифтом**.

***MUST - должно***

Это слово, а также термин **требуется** (REQUIRED) используется для требований, которые являются абсолютно необходимыми в данной спецификации.

***MUST NOT - недопустимо***

Эта фраза означает абсолютный запрет в рамках спецификации.

***SHOULD - следует***

Это слово, а также глагол **рекомендуется** (RECOMMENDED) используется для обозначения требований, от выполнения которых можно отказаться при наличии разумных причин. Однако при таком отказе следует помнить о возможных проблемах в результате отказа и принимать взвешенное решение.

***MAY - можно, возможно***

Это слово, а также прилагательное **необязательный** (OPTIONAL) обозначают элементы, реализация которых является необязательной. Реализация, не включающая ту или иную опцию, **должна** быть готова к работе с реализациями, которые используют эту опцию (возможно совместная работа будет обеспечиваться за счет некоторого ущерба функциональности).

## 1.2 Терминология

Ниже приведены определения терминов, часто используемых в этом документе.

***Datagram - дейтаграмма***

Единица передачи на сетевом уровне (например, IP). Дейтаграмма может инкапсулироваться в один или несколько пакетов, передаваемых канальному уровню.

***Frame - кадр***

Единица передачи на уровне логического канала данных. Кадр может включать заголовок и/или трейлер вместе с некоторым числом блоков данных.

***Packet - пакет***

Базовый блок инкапсуляции, который передается через интерфейс между сетевым и канальным уровнем. Пакет обычно отображается в кадр, за исключением случаев фрагментации на канальном уровне или встраивания в один кадр множества пакетов.

***Peer - партнер***

Другая сторона соединения «точка-точка».

***Silently discard – отбрасывание без уведомления***

Реализация отбрасывает пакет без дальнейшей обработки. Реализации **следует** поддерживать возможность записи в системный журнал ошибок, включая содержимое отброшенных без уведомления кадров, а также **следует** учитывать такие события в счетчике статистики.

# 2. Инкапсуляция PPP

Инкапсуляция PPP позволяет различать дейтаграммы разных протоколов. Инкапсуляция требует кадрирования для указания начала и конца. Методы кадрирования описаны в сопутствующих документах.

Инкапсуляция PPP показана на рисунке. Поля передаются слева направо.

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Protocol  |Information| Padding       |
| 8/16 битов|     *     |     *         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

***Protocol***

Поле Protocol состоит из одного или двух октетов и его значение указывает дейтаграмму, инкапсулированную в поле Information. Поле передается, начиная со старшего октета.

Структура этого поля совместима с механизмом расширения ISO 3309 для полей адреса. Все значения Protocol **должны** быть нечетными, т. е. младший бит младшего октета **должен** иметь значение 1. Значения поля **должны** назначаться так, чтобы младший бит старшего октета был равен 0. Принятые кадры, которые не соответствуют этим правилам **должны** быть считаться кадрами с неизвестным протоколом.

Значения поля Protocol от 0*** до 3*** указывают пакеты пакеты конкретного протокола сетевого уровня, а значения от 8*** до b*** - пакеты, относящиеся в соответствующему протоколу NCP (если он имеется).

Значения поля Protocol 4*** до 7*** используются для протоколов с незначительным трафиком, не имеющих соответствующего NCP. Значения поля от c*** до f*** указывают пакеты протоколов управления соединением (такие, как LCP).

Современные значения поля Protocol указаны в действующем RFC «Assigned Numbers» [2]. Эта спецификация резервирует приведенные в таблице значения.

| *Шестнадцатеричное значение* | *Имя протокола* |
|---|---|
| 1 | Протокол заполнения |
| 0003 - 001f | Резерв (прозрачность неэффективна) |
| 007d | Резерв (Control Escape) |
| 00cf | Резерв (PPP NLPID) |
| 00ff | Резерв (сжатие неэффективно) |
| 8001 - 801f | Не используется |
| 807d | Не используется |
| 80cf | Не используется |
| 80ff | Не используется |
| c021 | LCP |
| c023 | PAP[1] |
| c025 | Link Quality Report |
| c223 | CHAP[2] |

Разработчики новых протоколов **должны** получить номер в агентстве IANA[3], по адресу IANA@isi.edu.

***Information***

Поле Information может быть пустым и обычно включает дейтаграмму протокола, указанного в поле Protocol.

Максимальный размер поля Information, включая Padding, но без учета поля Protocol определяется размером максимального принимаемого блока переменной (MRU[4]), который по умолчанию составляет 1500 октетов. По согласованию совместимые реализации PPP могут установить другие значения MRU.

***Padding***

При передаче, поле Information может дополняться произвольным числом октетов, вплоть до MRU. Каждый протокол самостоятельно отличает заполнение от от реальной информации.

# 3. Работа соединений PPP

## 3.1 Обзор

Для коммуникаций через соединение «точка-точка», каждая сторона соединения PPP **должна** сначала передать пакеты LCP, служащие для настройки и тестирования канала данных. После организации соединения партнеры **могут** проверить подлинность друг друга.

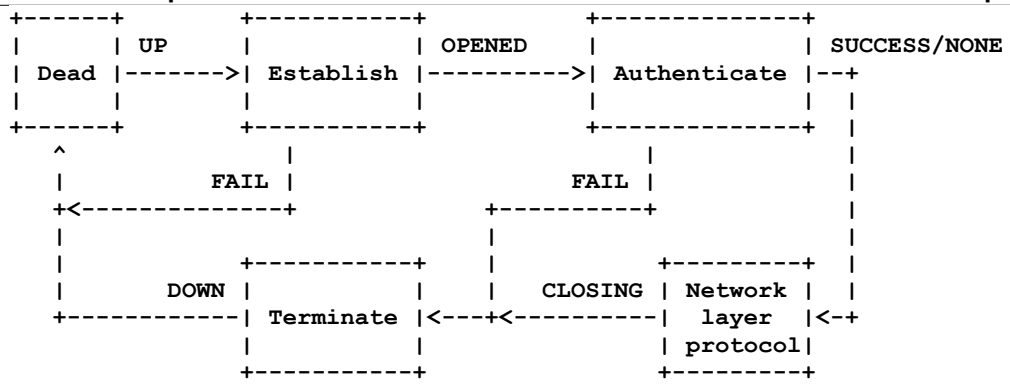
После этого PPP **должен** передать пакеты NCP для выбора и настройки одного или множества протоколов сетевого уровня. После настройки каждого из этих протоколов дейтаграммы протокола могут передаваться через канал.

Соединения остается настроенными для передачи данных до тех пор, пока заключительный соединение не будет закрыто явным пакетом LCP или NCP или не произойдет то или иное внешнее событие (таймер бездействия или действия администратора).

## 3.2 Фазы протокола

В процессе настройки, поддержки и разрыва соединение PPP проходит через несколько разных фаз, которые показаны на рисунке ниже.

[1]Password Authentication Protocol – протокол аутентификации по паролю.
[2]Challenge Handshake Authentication Protocol – протокол аутентификации «запрос-отклик».
[3]Internet Assigned Numbers Authority.
[4]Maximum Receive Unit.




На диаграмме указаны не все переходы. **Должна** соблюдаться описанная ниже семантика.

## 3.3 Link Dead

Соединение всегда начинается и завершается этой фазой. Когда внешнее событие (например, детектирование соединения на физическом уровне или настройка сетевым администратором) покажет, что физический уровень готово к использованию, PPP перейдет в фазу Link Establishment.

В течение этой фазы, машина LCP (см. ниже) будет находиться в состояниях Initial или Starting. Переход в фазу Link Establishment будет сигналом Up для машины состояний LCP.

> Примечание для разработчиков. Обычно соединение возвращается в эту фазу после отключения модема. В случае фиксированного соединения (hard-wired link) эта фаза может быть крайне непродолжительной – достаточно просто определить присутствие другого устройства.

## 3.4 Link Establishment

Протокол LCP служит для организации соединения путем обмена пакетами Configure. Этот обмен завершается и соединение переходит в состояние LCP после обмена пакетами Configure-Ack (см. ниже).

Для всех опций конфигурации предполагаются принятые по умолчанию значения, если они не изменены во время конфигурационного обмена. Дальнейшее обсуждение конфигурационных опций LCP приведено ниже.

Важно отметить, что LCP может настраивать лишь опции конфигурации, которые не зависят от конкретного протокола сетевого уровня. Настройка протоколов сетевого уровня выполняется отдельными протоколами NCP в фазе Network-Layer Protocol.

Все пакеты, не относящиеся к LCP, которые полученны в этой фазе, **должны** отбрасываться без уведомления.

Получение пакета LCP Configure-Request вызывает возврат из фаз Network-Layer Protocol и Authentication в фазу Link Establishment.

## 3.5 Authentication

На некоторых каналах может оказаться желательным запрос аутентификации партнера до начала обмена пакетами сетевого уровня.

По умолчанию аутентификация не требуется. Если реализация желает аутентифицировать партнера с помощью конкретного протокола проверки подлинности, она **должна** запросить использование этого протокола аутентификации в фазе Link Establishment.

Аутентификацию **следует** выполнять сразу после организации соединения. Однако одновременно с этим **может** выполняться проверка качества соединения. Реализации **недопустимо** разрешать обмен пакетами проверки качества соединения с задержкой аутентификации на неопределенный срок.

Переход из фазы Authentication в фазу Network-Layer Protocol **недопустим** до завершения проверки подлинности. Если аутентификация завершилась отказом, инициировавшей ее стороне **следует** перейти в фазу Link Termination.

В данной фазе может происходить обмен лишь пакетами LCP, протокола аутентификации и мониторинга качества соединения. Все остальные пакеты, полученные в течение этой фазы, **должны** отбрасываться без уведомления.

> Примечание для разработчиков. Реализации **не следует** выдавать отказ при аутентификации на основании лишь тайм-аута или отсутствия ответа. Для аутентификации **следует** поддерживать тот или иной метод повтора передачи и переходить в фазу Link Termination только после заданного числа попыток.
>
> За переход в фазу Link Termination отвечает реализация, отклонившая аутентификацию партнера.

## 3.6 Network-Layer Protocol

После того как PPP завершил предыдущие фазы, каждый протокол сетевого уровня (IP, IPX, AppleTalk) **должен** быть настроен отдельно соответствующим протоколом NCP.

Каждый протокол NCP в любой момент **может** находиться в состоянии Opened или Closed.

> Примечание для разработчиков. Поскольку реализация в начальной стадии может затратить значительное время на определение качества соединения, **следует** избегать фиксированных тайм-аутов при ожидании партнерами настройки NCP.

После того как NCP перейдет в состояние Opened, PPP может передавать пакеты соответствующего протокола сетевого уровня. Все пакеты протоколов сетевого уровня, принятые до того, как соответствующий NCP перейдет в состояние Opened, **должны** отбрасываться без уведомления.

Примечание для разработчиков. Пока LCP находится в состоянии Opened, для любого пакета не поддерживаемого реализацией протокола **должен** возвращаться отклик Protocol-Reject (см. ниже). Отбрасываются без уведомления лишь пакеты поддерживаемых протоколов.

В этой фазе трафик соединения может включать любые комбинации пакетов LCP, NCP и протоколов сетевого уровня.

## 3.7 Link Termination

PPP может завершить соединение в любой момент. Это может быть обусловлено потерей несущей, отказом при аутентификации или проверке качества соединения, а завершением отсчета таймера бездействия или административным разрывом соединения.

LCP используется для закрытия соединения путем обмена пакетами Terminate. При закрытии соединения PPP информирует протоколы сетевого уровня, чтобы они могли выполнить соответствующие действия.

После обмена пакетами Terminate, реализации **следует** сообщить физическому уровню о разрыве соединения, чтобы разорвать соединение на физическом уровне, особенно в случае отказа при аутентификации. Отправителю пакета Terminate-Request **следует** отключиться после получения пакета Terminate-Ack или по завершении отсчета Restart. Получателю пакета Terminate-Request **следует** ждать отключения партнера и **недопустимо** отключаться пока не пройдет как минимум один интервал Restart после передачи Terminate-Ack. PPP **следует** перейти в фазу Link Dead.

Все пакеты, не относящиеся к LCP, полученные в течение этой фазы, **должны** отбрасываться без уведомления.

Примечание для разработчиков. Закрытия соединения с помощью LCP достаточно и протоколам NCP не требуется передавать пакеты Terminate. И наоборот, перехода NCP в состояние Closed не является достаточным основанием для закрытия соединения PPP, даже если этот NCP был единственным протоколом в состоянии Opened.

# *4. Машина согласования опций*

Автоматизация конечных состояний определяется событиями, действиями и сменой состояний. События включают получение внешних команд, таких как Open и Close, завершение отсчета таймера Restart, получение пакетов от партнера. Действия включают запуск таймера Restart и передачу пакетов партнеру.

Некоторые типы пакетов - Configure-Nak и Configure-Reject, Code-Reject и Protocol-Reject, Echo-Request, Echo-Reply и Discard-Request не различаются в описаниях автомата. Как будет показано ниже, эти пакеты в действительности служат для разных функций. Однако они всегда приводят к одинаковым сменам состояний.

| ***События*** | | ***Действия*** | |
|---|---|---|---|
| Up | Нижележащий уровень в состоянии Up | tlu | Данный уровень в состоянии Up |
| Down | Нижележащий уровень в состоянии Down | tld | Данный уровень в состоянии Down |
| Open | Административный переход в состояние Open | tls | Данный уровень в состоянии Started |
| Close | Административный переход в состояние Close | tlf | Данный уровень в состоянии Finished |
| TO+ | Тайм-аут с незавершенным (>0) счетчиком | irc | Initialize-Restart-Count |
| TO- | Тайм-аут с завершенным счетчиком | zrc | Zero-Restart-Count |
| RCR+ | Получен корректный запрос Configure-Request | scr | Передан Configure-Request |
| RCR- | Получен некорректный запрос Configure-Request | | |
| RCA | Получен Configure-Ack | sca | Передан Configure-Ack |
| RCN | Получен Configure-Nak/Rej | scn | Передан Configure-Nak/Rej |
| RTR | Получен Terminate-Request | str | Передан Terminate-Request |
| RTA | Получен Terminate-Ack | sta | Передан Terminate-Ack |
| RUC | Получен неизвестный код | scj | Передан Code-Reject |
| RXJ+ | Получен Code-Reject (разрешено)<br>или Protocol-Reject | | |
| RXJ- | Получен Code-Reject (катастрофа)<br>или Protocol-Reject | | |
| RXR | Получен Echo-Request<br>или Echo-Reply<br>или Discard-Request | ser | Передан Echo-Reply |

## 4.1 Таблица переходов состояний

Полная таблица смены состояний приведена ниже. Состояния указаны в верхней строке, события - в левом столбце. Смены состояний и действия представлены в форме действие/новое состояние. Множественные действия разделены запятыми и могут продолжаться в следующих строках, множественные действия могут быть реализованы в любом удобном порядке. Дефис (-) показывает недопустимость смены состояния.

| | *Состояние* | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| ***События*** | ***0 Initial*** | ***1 Starting*** | ***2 Closed*** | ***3 Stopped*** | ***4 Closing*** | ***5 Stopping*** | ***6 Req-Sent*** | ***7 Ack-Rcvd*** | ***8 Ack-Sent*** | ***9 Opened*** |
| Up | 2 | irc,scr/6 | - | - | - | - | - | - | - | - |
| Down | - | - | 0 | tls/1 | 0 | 1 | 1 | 1 | 1 | tld/1 |
| Open | tls/1 | 1 | irc,scr/6 | 3r[1] | 5[1] | 5[1] | 6 | 7 | 8 | 9[1] |
| Close | 0 | tlf/0 | 2 | 2 | 4 | 4 | irc,str/4 | irc,str/4 | irc,str/4 | irc,str/4 |
| TO+ | - | - | - | - | str/4 | str/5 | scr/6 | scr/6 | scr/8 | - |
| TO- | - | - | - | - | tlf/5 | tlf/3 | tlf/3[2] | tlf/3[2] | tlf/3[2] | - |
| RCP+ | - | - | sta/2 | irc,scr,sca/8 | 4 | 5 | sca/8 | sca,tlu/8 | sca/8 | tld,scr,sca/8 |
| RCP- | - | - | sta/2 | irc,scr,sca/8 | 4 | 5 | scn/6 | scn/7 | scn/6 | tld,scr,scn/6 |
| RCA | - | - | sta/2 | sta/3 | 4 | 5 | irc/7 | scr/6[3] | irc,tlu/9 | tld,scr/6[3] |
| RCN | - | - | sta/2 | sta/3 | 4 | 5 | irc,scr/6 | scr/6[3] | irc,scr/8 | tld,scr/6[3] |
| RTR | - | - | sta/2 | sta/3 | sta/4 | sta/5 | sta/6 | sta/6 | sta/6 | tld,zrc,sta/5 |
| RTA | - | - | 2 | 3 | tlf/2 | tlf/3 | 6 | 6 | 8 | tld,scr/6 |
| RUC | - | - | scj/2 | scj/3 | scj/4 | scj/5 | scj/6 | scj/7 | scj/8 | scj/9 |
| RXJ+ | - | - | 2 | 3 | 4 | 5 | 6 | 6 | 8 | 9 |
| RXJ- | - | - | tlf/2 | tlf/3 | tlf/2 | tlf/3 | tlf/3 | tlf/3 | tlf/3 | tld,irc,str/5 |
| RXR | - | - | 2 | 3 | 4 | 5 | 6 | 7 | 8 | ser/8 |

Состояния, в которых таймер Restart запущен, узнаваемы по присутствию событий TO. Таймер Restart запускают или перезапускают лишь действия Send-Configure-Request, Send-Terminate-Request и Zero-Restart-Count. Таймер Restart останавливается, когда происходит переход из состояния, где таймер работает, в состояние, где таймер не работает.

События и действия, определяются в соответствии с проходящими сообщениями, а не архитектурой сигнализации. Если для действия желательно управление конкретными сигналами (например, DTR), могут потребоваться дополнительные действия.

## 4.2 Состояния

Ниже переведено детальное описание каждого состояния машины конечных состояний.

***Initial***

В состояние Initial нижний уровень недоступен (Down) и не происходит событий Open. Таймер Restart не работает в состоянии Initial.

***Starting***

Состояние Starting является аналогом Open для состояния Initial. Был инициирован административный переход в Open, но нижний уровень все еще не доступен (Down). Таймер Restart не работает в этом состоянии.

Когда нижний уровень станет доступен (Up), передается сообщение Configure-Request.

***Closed***

В состоянии Closed соединение доступно (Up), но событие Open еще не произошло. Таймер Restart не работает в состоянии Closed.

В ответ на прием пакетов Configure-Request посылается Terminate-Ack. Terminate-Ack отбрасывается для предотвращения образования петли.

***Stopped***

Состояние Stopped является аналогом Open для состояния Closed. Это состояние возникает, когда машина ждет события Down после действия This-Layer-Finished или после отправки Terminate-Ack. Таймер Restart не работает в состоянии Stopped.

При получении пакетов Configure-Request, передается подходящий ответ, при получении других пакетов - Terminate-Ack. Сообщения Terminate-Ack отбрасывается без уведомления для предотвращения петель.

Обоснование. Состояние Stopped является переходным при разрыве соединения, отказе при настройке и отказах машины состояний. Оно потенциально разделяет состояния, которые были объединены.

Это вариант состязания между откликом на событие Down (в результате This-Layer-Finished) и событием Receive-Configure-Request. Когда сообщение Configure-Request приходит до события Down, это событие будет вытеснено возвратом машины в состояние Starting. Это предотвращает атаку на основе повторов.

Вариант реализации. После того как партнер отказался ответить на сообщения Configure-Request, реализация **может** пассивно ждать, когда партнер передаст Configure-Request. В этом случае действие This-Layer-Finished не используется для события TO- в состояниях Req-Sent, Ack-Rcvd и Ack-Sent.

Эта опция полезна на выделенных устройствах и каналах не имеющих сигналов состояния, но ее **не следует** применять для коммутируемых каналов.

***Closing***

В состоянии Closing предпринимается попытка разорвать соединение. Запрос Terminate-Request передан и таймер Restart запущен, но сообщение Terminate-Ack еще не получено.

При получении Terminate-Ack происходит переход в состояние Closed. По завершении отсчета таймера Restart передается новое сообщение Terminate-Request и таймер перезапускается. После того, как отсчет таймера завершился Max-Terminate раз, происходит переход в состояние Closed.

***Stopping***

Состояние Stopping является аналогом Open для состояния Closing. Запрос Terminate-Request передан и таймер Restart запущен, но сообщение Terminate-Ack еще не получено.

[1]Перезапуск опции. См. обсуждение состояния Open.
[2]Пассивная опция. См. обсуждение состояния Stopped.
[3]Пересекающееся соединение. См. обсуждение события RCA.

Обоснование. Состояние Stopping предоставляет шанс завершить соединения до того, как появится новый трафик. После разрыва соединения может быть создана новая конфигурация через состояния Stopped или Starting.

***Request-Sent***

Состояние Request-Sent является попыткой настроить соединение. Запрос Configure-Request передан, таймер Restart запущен, но сообщение Configure-Ack еще не получено и не передано.

***Ack-Sent***

В состоянии Ack-Sent сообщения Configure-Request и Configure-Ack уже посланы, но подтверждение Configure-Ack еще не получено. Таймер Restart запущен, поскольку сообщение Configure-Ack не получено.

***Opened***

В состоянии Opened подтверждение Configure-Ack передано и получено. Таймер Restart не запущен.

При переходе в состояние Opened реализации **следует** сообщить вышележащим уровням о событии Up. При выходе из состояния Opened реализации **следует** сообщить вышележащим уровням о событии Down.

## 4.3 События

Смены состояния и действия машины состояний обусловлены событиями, перечисленными ниже.

***Up***

Это событие происходит, когда нижележащий уровень показывает что он готов передавать пакеты.

Обычно это событие используется для работы с модемом, вызывающим процессорм или какой-то другой связкой PPP с физической средой, чтобы сообщить LCP о переходе соединения в фазу Link Establishment.

Событие также используется LCP для уведомления каждого NCP о переходе соединения в фазу Network-Layer Protocol, т. е. действие This-Layer-Up в LCP вызывает событие Up в NCP.

***Down***

Это события происходит, когда нижний уровень показывает, что он не готов дальше передавать пакеты.

Обычно это событие используется для работы с модемом, вызывающим процессорм или какой-то другой связкой PPP с физической средой, чтобы сообщить LCP о переходе соединения в фазу Link Dead.

Событие также используется LCP для уведомления каждого NCP о выходе канала из фазы Network-Layer Protocol, т. е. действие This-Layer-Down в LCP вызывает событие Down в NCP.

***Open***

Это событие показывает, что соединение административно доступно для трафика, т. е. сетевой администратор (человек или программа) указал, что соединению разрешен переход в состояние Opened. Если это событие происходит, когда канал находится не в состоянии Opened, машина состояний пытается передать партнеру пакеты настройки конфигурации.

Если автомат не может начать настройку конфигурации (нижележащий уровень в состоянии Down или предыдущее событие Close не завершено), организация соединения автоматически задерживается.

При получении запроса Terminate-Request или других событиях, которые делают канал недоступным, машина перейдет в состояние, где канал будет готов к посторному соединению. Административного вмешательства не требуется.

Примечание для разработчиков. Опыт показывает, что пользователи выполняют дополнительную команду Open, если им нужно заново согласовать соединение. Это может указывать согласование новых значений.

Поскольку это не означает события Open, предполагается, что при выполнении пользовательской команды Open в состоянии Opened, Closing, Stopping или Stopped реализация создает событие Down, за которым сразу же следует событие Up. Нужно принять меры против получения промежуточного состояния Down из другого источника.

Событие Down, за которым следует Up, приводит к упорядоченному повторному согласованию канала путем прохождения от состояния Starting к состоянию Request-Sent. Это приведет к повторному согласованию канала без негативных побочных эффектов.

***Close***

Это событие показывает, что канал недоступен для трафика, т. е. сетевой администратор (человек или программа) указал, что каналу запрещен переход в состояние Opened. Когда такое событие происходит в отличном от Closed состоянии канала, автоматически предпринимается попытка разорвать соединение. Дальнейшие попытки перенастроить канал отвергаются, пока не произойдет новое событие Open.

Примечание для разработчиков. При отказе аутентификации канал **следует** разорвать для предотвращения повторных атак и отказа в обслуживании других пользователей. Поскольку соединение административно доступно (по определению), это может быть достигнуто имитацией события Close для LCP, за которой незамедлительно следует событие Open. Нужно позаботиться о том, чтобы событие Close не могло быть вызвано другим источником.

Событие Close, за которым следует Open, будет приводить к упорядоченному завершению соединения, проходящему через состояния Closing и Stopping, когда действие This-Layer-Finished может отключить соединение. Машина будет ожидать в состоянии Stopped или Starting для следующей попытки соединения.

***Timeout (TO+,TO-)***

Это событие показывает завершение отсчета таймера Restart, который определяет время ответа на запросы Configure-Request и Terminate-Request.

Событие TO+ показывает, что значение счетчика Restart остается больше нуля, что вызывает повторную передачу пакетов Configure-Request и Terminate-Request.

Событие TO- показывает, что значение счетчика Restart не больше нуля и больше не требуется повторять передачу пакетов.

***Receive-Configure-Request (RCR+,RCR-)***

Это событие обусловлено получением от партнера пакета Configure-Request, который показывает желание создать соединение и может содержать параметры конфигурации. Пакет Configure-Request будет описан ниже.

Событие RCR+ говорит о том, что запрос Configure-Request был воспринят и вызвал отправку соответствующего пакета Configure-Ack.

Событие RCR- показывает, что пакет Configure-Request не был воспринят и вызвал отправку соответствующего пакета Configure-Nak или Configure-Reject.

Примечание для разработчиков. Эти события могут произойти, когда соединение уже находится в состоянии Opened. Реализация **должна** быть готова повторно согласовать конфигурационные опции.

***Receive-Configure-Ack (RCA)***

Это событие обусловлено приемом корректного пакета Configure-Ack от партнера. Пакет Configure-Ack является положительным ответом на пакет Configure-Request. Пакеты с нарушением порядка доставки и другие недействительные пакеты отбрасываются.

Примечание для разработчиков. Поскольку корректный пакет уже был получен до перехода в состояние Ack-Rcvd или Opened, получение других таких пакетов крайне маловероятно. Как было отмечено, все недействительные пакеты Ack/Nak/Rej отбрасываются и не влияют на состояние машины.

Однако не исключено прибытие корректно сформированного пакета из совпадающего по времени кросс-соединения. Скорей всего это будет ошибкой реализации. Такие события **следует** по меньшей мере записывать в системный журнал.

***Receive-Configure-Nak/Rej (RCN)***

Это событие обусловлено получением от партнера действительного пакета Configure-Nak или Configure-Reject, которые являются негативными откликами на Configure-Request. Пакеты с нарушением порядка доставки и другие недействительные пакеты отбрасываются без уведомления.

Примечание для разработчиков. Хотя Configure-Nak и Configure-Reject вызывают одинаковые смены состояния, эти пакеты по-разному влияют на конфигурационные параметры, передаваемые в результирующем пакете Configure-Request.

***Receive-Terminate-Request (RTR)***

Это событие обусловлено получением пакета Terminate-Request. Запрос Terminate-Request уазывает желание партнера закрыть соединение.

Примечание для разработчиков. Это событие не идентично событию Close (см. выше) и не отменяет команды Open от локального сетевого администратора. Реализация **должна** быть готова получить новый пакет Configure-Request без вмешательства сетевого администратора.

***Receive-Terminate-Ack (RTA)***

Это событие происходит, когда от удаленной стороны получен пакет Terminate-Ack. Пакет Terminate-Ack обычно является ответом на пакет Terminate-Request. Так же он может указывать на то, что удаленная сторона находится в состоянии Closed или Stopped, а так же служит для повторной синхронизации конфигурации соединения.

***Receive-Unknown-Code (RUC)***

Это событие обусловлено получением от партнера пакета, который не удается интерпретировать. В ответ передается пакет Code-Reject.

***Receive-Code-Reject, Receive-Protocol-Reject (RXJ+,RXJ-)***

Это событие обусловлено получением от партнера пакета Code-Reject или Protocol-Reject.

Событие RXJ+ происходит, когда отвергнутое значение приемлемо, например, Code-Reject из расширенного кода или Protocol-Reject от NCP. Это не выходит за пределы нормального функционирования. Реализация **должна** прекратить передачу соответствующего типа пакетов.

Событие RXJ- происходит, когда отвергнутое значение критично, например, Code-Reject для Configure-Request или Protocol-Reject от LCP. Это событие вызывает непоправимую ошибку и разрывает соединение.

***Receive-Echo-Request, Receive-Echo-Reply, Receive-Discard-Request (RXR)***

Это событие обусловлено получением от партнера пакетов Echo-Request, Echo-Reply или Discard-Request. Пакет Echo-Reply является откликом на Echo-Request. Для пакетов Echo-Reply или Discard-Request откликов не бывает.

## 4.4 Действия

Действия в машине состояний обусловлены событиями и обычно указывают отправку пакетов и/или запуском или остановкой таймера Restart.

***Illegal-Event (-)***

Это действие указывает событие, которого не могло произойти в правильно реализованной машине. Реализация имеет внутренние ошибки, которые следует занести в системный журнал. Смены состояний не происходит и реализации **не следуе** выполнять сброс или останавливаться.

***This-Layer-Up (tlu)***

Это действие показывает вышележащим уровням переход в состояние Opened.

Обычно это действие используется протоколом LCP для информирования о событии Up протокола аутентификации, контроля качества канала или NCP, а также **может** применяться NCP для индикации доступности канала для трафика сетевого уровня.

***This-Layer-Down (tld)***

Это действие показывает вышележащим уровням выход из состояния Opened.

Обычно это действие используется протоколом LCP для информирования о событии Down протокола аутентификации, контроля качества канала или NCP, а также **может** применяться NCP для индикации недоступности канала для трафика сетевого уровня.

***This-Layer-Started (tls)***

Это действие показывает нижележащим уровням выход из состояния Starting и потребность в канале нижележащего уровня. Нижележащему уровню **следует** ответить событием Up, когда этот уровень будет доступен. Результат этого действия зависит от реализации.

***This-Layer-Finished (tlf)***

Это действие показывает нижележащим уровням переход машины в состояние Initial, Closed или Stopped, когда нижележащий уровень больше не требуется для канала. Нижележащему уровню **следует** ответить событием Down, когда канал будет разорван.

Обычно это действие **может** использоваться протоколом LCP перед фазой Link Dead и **может** применяться также NCP для индикации протоколу LCP возможности разорвать соединение при отсутствии других открытых NCP.

Результат этого действия зависит от реализации.

***Initialize-Restart-Count (irc)***

Это действие устанавливает подходящее значение счетчика Restart (Max-Terminate или Max-Configure). Счетчик декрементируется после каждой отправки, включая первую.

Примечание для разработчиков. В дополнение к установке счетчика Restart реализация **должна** установить начальное значение для тайм-аута при снижении значения для таймера Restart.

***Zero-Restart-Count (zrc)***

Это действие устанавливает Restart = 0.

Примечание для разработчиков. Это действие позволяет приостановить FSA перед переходом в желаемое финальное состояние, позволяя партнеру обработать трафик. Кроме сброса в 0 счетчика Restart реализация **должна** установить подходящее значение для тайм-аута.

***Send-Configure-Request (scr)***

Передается пакет Configure-Request. Это указывает желание создать соединение с заданным набором конфигурационных опций. При отправке Configure-Request запускается таймер Restart для защиты от потери пакетов. Счетчик Restart декрементируется при каждой передаче пакета Configure-Request.

***Send-Configure-Ack (sca)***

Передается пакет Configure-Ack, подтверждающий прием Configure-Request с приемлемым набором конфигурационных опций.

***Send-Configure-Nak (scn)***

Передан пакет Configure-Nak или Configure-Reject, служащий негативным откликом на пакет Configure-Request с неприемлемым набором конфигурационных опций.

Пакеты Configure-Nak служат для отказа от значения Configuration Option и предлагают приемлемый набор опций. Пакеты Configure-Reject используются для отказа от согласования конфигурационных опций, обычно по причине того, что они не распознаны или не реализованы.

***Send-Terminate-Request (str)***

Передан пакет Terminate-Request, указывающий желание закрыть соединение. При передаче пакета Terminate-Request запускается таймер Restart. Счетчик Restart декрементируется при каждой передаче Terminate-Request.

***Send-Terminate-Ack (sta)***

Передан пакет Terminate-Ack, служащий для подтверждения приема Terminate-Request или синхронизации машин состояний.

***Send-Code-Reject (scj)***

Передан пакет Code-Reject, указывающий получение пакета неизвестного типа.

***Send-Echo-Reply (ser)***

Передан пакет Echo-Reply, подтверждающий прием пакета Echo-Request.

## 4.5 Предотвращение петель

Протокол делает предпринимает попытки избавится от петель согласования конфигурационных опций. Однако протокол **не** гарантирует отсутствие петель. Как при любом согласовании можно настроит две реализации PPP с конфликтующими политиками, которые никогда не сойдутся. Можно также настроить политики, которые будут сходиться, но это займет осень много времени. Разработчикам следует принимать это во внимание, а также **следует** реализовать механизм обнаружения петель или тайм-аутов на вышележащих уровнях.

## 4.6 Счетчики и таймеры

***Restart***

В машине состояний используется один таймер. Таймер Restart служит для управления временем передачи пакетов Configure-Request и Terminate-Request. Завершение отсчета этого таймера вызывает событие Timeout и повтор передачи соответствующего пакета Configure-Request или Terminate-Request. Таймер Restart **должен** быть настраиваемым и по умолчанию **следует** установить значение три (3) секунды.

Примечание для разработчиков. Значение таймера Restart **следует** устанавливать в зависимости от скорости соединения. Принятое по умолчанию значение рассчитано на низкоскоростные (2400 - 9600 бит/с) коммутируемые линии с медленной коммутацией (обычные телефонные линии). Для высокоскоростных соединений или соединений с быстрой коммутацией **следует** выбирать меньшие значения.

Вместо постоянного значения **можно** изначально устанавливать для таймер Restart небольшое значение и постепенно повышать его до заданного в конфигурации конечного значения. При этом каждое новое значение **следует** делать по как минимум вдвое больше предыдущего, пока не достигнут заданный максимум. Начальное значение **следует** выбирать не меньше удвоенного времени передачи пакета с поддерживаемой в канале скоростью с добавлением не менее 100 мсек, чтобы позволить партнеру обработать пакет перед отправкой отклика. На некоторых соединениях добавляется еще 200 мсек задержки в спутниковом канале. Время кругового обхода для модемов со скоростью 14400 бит/с составляет от 160 до 600 миллисекунд.

***Max-Terminate***

Имеется один обязательный счетчик повторов для запросов Terminate-Request. Счетчик Max-Terminate показывает число переданных пакетов Terminate-Request, на которые не было получено откликов Terminate-Ack до того, как будет сделан вывод о невозможности партнера ответить. Счетчик Max-Terminate **должен** быть настраиваемым и по умолчанию **следует** установить два (2) повтора.

***Max-Configure***

Аналогичный предыдущему счетчик рекомендуется и для запросов Configure-Request. Счетчик Max-Configure показывает число переданных пакетов Configure-Request, на которые не было получено откликов Configure-Ack, Configure-Nak или Configure-Reject до того, как будет сделан вывод о невозможности партнера ответить. Счетчик Max-Configure **должен** быть настраиваемым и по умолчанию **следует** установить десять (10) повторов.

***Max-Failure***

Похожий счетчик рекомендуется для Configure-Nak. Счетчик Max-Failure показывает число переданных пакетов Configure-Nak без передачи Configure-Ack до того, как будет сделан вывод о том, что конфигурация не сходится. Все дополнительные пакеты Configure-Nak для опций, запрошенных партнером, будут преобразованы в пакеты Configure-Reject, а опции, запрошенные локальной стороной, не будут добавляться. Счетчик Max-Failure **должен** быть настраиваемым и по умолчанию **следует** установить пять (5) повторов.

# *5. Форматы пакетов LCP*

Существует три класса пакетов LCP, перечисленных ниже.

1. Организация и настройка соединения (Configure-Request, Configure-Ack, Configure-Nak и Configure-Reject).
2. Завершение соединения (Terminate-Request и Terminate-Ack).

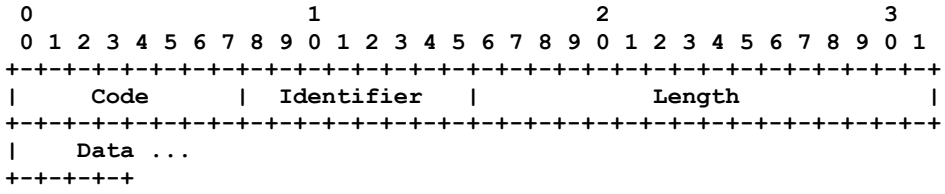
3. Обслуживание и отладка канала (Code-Reject, Protocol-Reject, Echo-Request, Echo-Reply и Discard-Request).

Для упрощения в пакетах LCP отсутствует поле версии. Корректно работающая реализация LCP будет всегда отвечать на незнакомые значения Protocol и Code легко распознаваемым пакетом LCP, предоставляя таким образом детерминированный механизм «отступления» для реализаций других версий протокола.

Независимо от разрешенных опций конфигурации, все LCP-пакеты Link Configuration, Link Termination и Code-Reject (коды 1 - 7) всегда передаются как при отсутствии согласования опций конфигурации. В частности каждая конфигурационная опция задает принятое по умолчанию значение. Это гарантирует распознаваемость таких пакетов LCP даже в тех случаях, когда одна сторона соединения ошибочно считает соединение отрытым.

В поле PPP Information инкапсулируется один пакет LCP, при этом поле PPP Protocol имеет шестнадцатеричное значение c021 (LCP).

Формат пакета LCP показан ниже. Поля передаются слева направо.




***Code***

Однооктетное поле Code указывает тип пакета LCP. При получении пакета с неизвестным полем Code передается пакет Code-Reject.

Современные значения поля Code определены в актуальном RFC «Assigned Numbers» [2]. Связанные с этим документом значения перечислены ниже.

1. Configure-Request
2. Configure-Ack
3. Configure-Nak
4. Configure-Reject
5. Terminate-Request
6. Terminate-Ack
7. Code-Reject
8. Protocol-Reject
9. Echo-Request
10. Echo-Reply
11. Discard-Request

***Identifier***

Однооктетное поле Identifier служит для сопоставления откликов с запросами. При получении пакета с недействительным полем Identifier такой пакет отбрасывается без уведомления и обработки.

***Length***

Двухоктетное поле Length указывает размер пакета LCP с учетом полей Code, Identifier, Length и Data. Значению поля Length **недопустимо** превышать MRU для соединения.

Октеты за пределами размера, заданного полем Length, считаются заполнением и игнорируются получателем пакета. Пакеты с недопустимым значением поля Length отбрасываются без уведомления и обработки.

***Data***

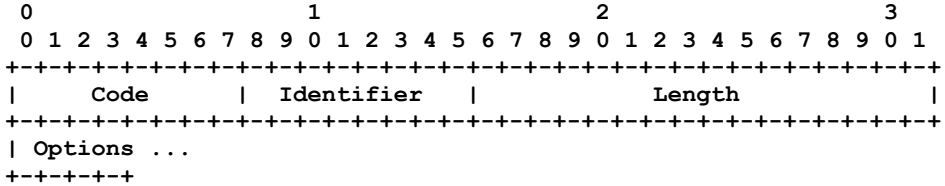
Поле Data содержит октеты данных в соответствии со значением поля Length. Формат поля Data определяется полем Code.

## 5.1 Configure-Request

Реализация, желающая организовать соединение должна передать пакет Configure-Request. Поле Options заполняется любыми желательными значениями, изменяющими принятые по умолчанию. Не следует включать опции с принятыми по умолчанию значениями.

При получении пакета Configure-Request должен передаваться соответствующий отклик.

Поля пакета Configure-Request показаны ниже. Поля передаются слева направо.




***Code***

1 (Configure-Request).

***Identifier***

Поле Identifier **должно** изменяться всякий раз при изменении содержимого поля Options или получении действительного отклика на предыдущий запрос. При повторе поле Identifier **может** сохраняться.

***Options***

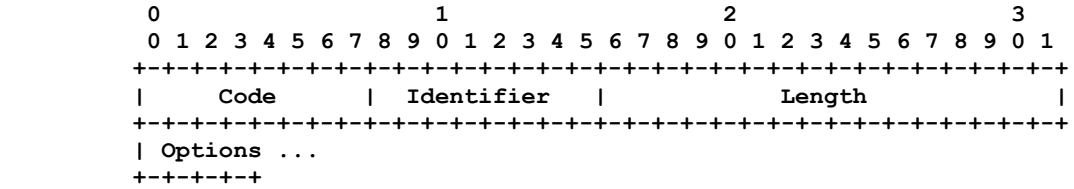
Поле Options переменного размера содержит список (возможно пустой) конфигурационных опций, которые отправитель желает согласовать. Все конфигурационные опции согласуются совместно. Формат конфигурационных опций описан ниже.

## 5.2 Configure-Ack

Если каждая конфигурационная опция, полученная в Configure-Request, опознана и все значения приемлемы, реализация **должна** передать отклик Configure-Ack. Для подтвержденных опций конфигурации **недопустимо** менять порядок или вносить иные изменения.

Поле Identifier в принятом пакете Configure-Ack **должно** соответствовать значению в последнем переданном пакете Configure-Request. Кроме того, конфигурационные опции в Configure-Ack **должны** совпадать с опциями в последнем переданном пакете Configure-Request. Недействительные пакеты отбрасываются без уведомления.

Формат пакета Configure-Ack показан ниже. Поля передаются слева направо.




***Code***

2 (Configure-Ack).

***Identifier***

Поле Identifier содержит копию одноименного поля в пакете Configure-Request, вызвавшем передачу Configure-Ack.

***Options***

Поле Options переменного размера содержит список (возможно пустой) конфигурационных опций, которые отправитель подтверждает. Все конфигурационные опции подтверждаются совместно.

## 5.3 Configure-Nak

Если полученные конфигурационные опции опознаны, но некоторые значения не приемлемы, реализация **должна** передать пакет Configure-Nak. В поле Options указываются только неприемлемые опции из пакета Configure-Request, а пригодные опции исключаются из Configure-Nak, но при этом **недопустимо** менять порядок опций из Configure-Request.

Для опций, не имеющих значений (логические), вместо Configure-Nak **должен** использоваться отклик Configure-Reject.

Для каждой конфигурационной опции, которая допустима лишь в одном экземпляре, значение в Configure-Nak **должно** быть заменено приемлемым. **Можно** использовать принятое по умолчанию значение, если оно отличается от запрошенного.

Когда конкретная опции может указываться более одного раза с разными значениями, пакет Configure-Nak **должен** включать список всех значений этой опции, приемлемых для отправителя Configure-Nak. Включаются также приемлемые значения, которые присутствовали в Configure-Request.

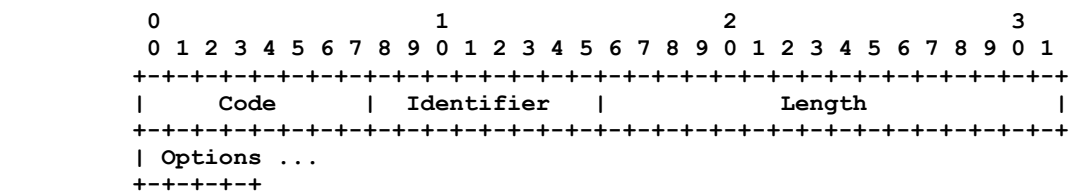
Реализация может быть настроена на запрос согласования определенной опции. Если эта опция не указана, ее **можно** добавить в список опций Configure-Nak, как предложение партнеру включить эту опцию в следующий пакет Configure-Request. Все поля опций **должны** содержать значения, приемлемые для отправителя Configure-Nak.

Поле Identifier в принятом пакете Configure-Nak **должно** совпадать с переданным в последнем пакете Configure-Request. Недействительные пакеты отбрасываются без уведомления.

Получение действительного пакета Configure-Nak показывает, что при передаче нового пакета Configure-Request конфигурационные опции **можно** изменить в соответствии с Configure-Nak. При наличии множества экземпляров опции партнеру **следует** выбрать одно значение и включить его в следующий пакет Configure-Request.

Некоторые конфигурационные опции имеют переменный размер. Поскольку в Configure-Nak опции были изменены партнером, реализация **должна** быть способна ботать опции, размер которых отличается от размера в исходном пакете Configure-Request.

Формат пакета Configure-Nak показан ниже. Поля передаются слева направо.




***Code***

3 (Configure-Nak).

***Identifier***

Поле Identifier содержит копию одноименного поля в пакете Configure-Request, вызвавшем передачу Configure-Nak.

***Options***

Поле Options переменного размера включает список (возможно пустой) конфигурационных опций, неприемлемых для отправителя. Все конфигурационные опции обрабатываются совместно.

## 5.4 Configure-Reject

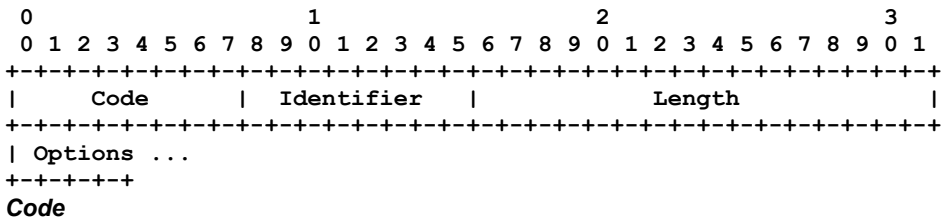
Если некоторые опции из пакета Configure-Request не удается распознать или они не приемлемы при согласовании (административные настройки), реализация **должна** передать отклик Configure-Reject. В поле Options указываются

только неприемлемые опциями из Configure-Request, а все понятные и пригодные опции исключаются из Configure-Reject, но при этом **недопустимо** менять порядок опций из Configure-Request.

Поле Identifier в принятом пакете Configure-Reject **должно** совпадать с переданным в последнем пакете Configure-Request. Кроме того, опции в пакете Configure-Reject **должны** быть подмножеством переданных в последнем пакете Configure-Request опций. Недействительные пакеты отбрасываются без уведомления.

Получение действительного пакета Configure-Reject указывает, что при передаче нового Configure-Request **недопустимо** включать какую-либо из опций, содержащихся в Configure-Reject.

Формат пакета Configure-Reject показан ниже. Поля передаются слева направо.




***Code***

4 (Configure-Reject).

***Identifier***

Поле Identifier содержит копию одноименного поля в пакете Configure-Request, вызвавшем Configure-Reject.

***Options***

Поле Options переменного размера включает список (возможно пустой) конфигурационных опций, отвергнутых отправителем. Все конфигурационные опции отвергаются совместно.

## 5.5 Terminate-Request и Terminate-Ack

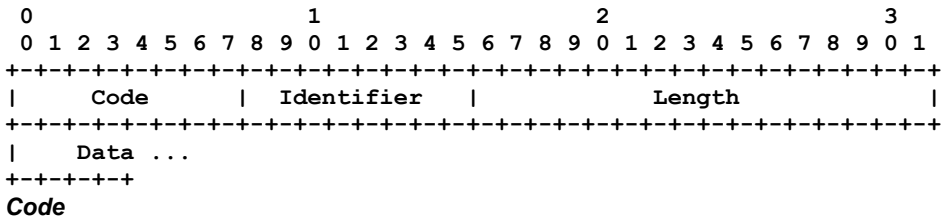
LCP включает коды Terminate-Request и Terminate-Ack в качестве механизма закрытия соединений.

Реализации, желающей закрыть соединение, **следует** передать пакет Terminate-Request. Передачу этих пакетов **следует** повторять, пока не будет получено подтверждение Terminate-Ack и нижележащий уровень не покажет отключение или пока не будет передано достаточно большое число пакетов, позволяющее с уверенностью считать, что партнер отключился.

При получении Terminate-Request **должен** возвращаться пакет Terminate-Ack.

Получение не запрошенного пакета Terminate-Ack показывает, что партнер находится в состоянии Closed или Stopped или повторное согласование требуется по иным причинам.

Формат пакетов Terminate-Request и Terminate-Ack показан ниже. Поля передаются слева направо.




***Code***

5 (Terminate-Request) или 6 (Terminate-Ack).

***Identifier***

При передаче поле Identifier **должно** быть изменено при изменении содержимого поля Data или получении действительного отклика на предыдущий запрос. При повторе поле Identifier **может** сохраняться.

На приемной стороне поле Identifier копируется из пакета Terminate-Request в пакет Terminate-Ack.

***Data***

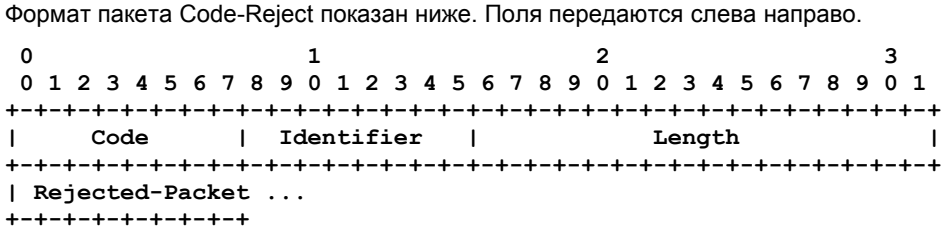
Поле Data (возможно пустое) содержит неинтерпретируемые данные для использования отправителем. Поле может включать произвольное двоичное значение. Окончание данных указывается значением поля Length.

## 5.6 Code-Reject

Получение пакета LCP с неизвестным кодом указывает, что партнер использует другую версию протокола. Этот код **должен** быть возвращен отправителю в пакете Code-Reject.

При получении Code-Reject с кодом, который требуется в этой версии протокола, реализации **следует** сообщить о проблеме и разорвать соединение, поскольку автоматическое решение проблемы маловероятно.

Формат пакета Code-Reject показан ниже. Поля передаются слева направо.




***Code***

7 (Code-Reject).

***Identifier***

Поле Identifier **должно** изменяться для каждого передаваемого пакета Code-Reject.

***Rejected-Packet***

Поле Rejected-Packet содержит копию отвергнутого пакета LCP, начиная с поля Information без заголовков канального уровня и FCS. Размер пакета Rejected-Packet **должен** отсекаться в соответствии с согласованным партнерами значением MRU.

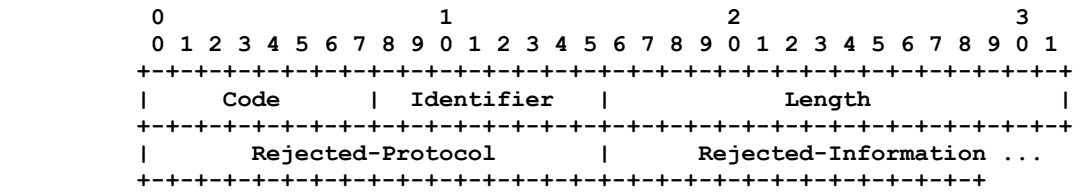
## 5.7 Protocol-Reject

Получение пакета PPP с неизвестным полем Protocol показывает, что партнер пытается использовать не поддерживаемый протокол. Это обычно происходит, когда партнер пытается настроить новый протокол. Если машина LCP находится в состоянии Opened, партнеру **должен** быть передан пакет Protocol-Reject.

При получении пакета Protocol-Reject реализация **должна** прекратить передачу пакетов указанного протокола при первой возможности.

Пакеты Protocol-Reject могут передаваться лишь в том случае, когда LCP находится в состоянии Opened. Пакеты Protocol-Reject, полученные в состоянии, отличном от Opened, **следует** отбрасывать без уведомления.

Формат пакета Protocol-Reject показан ниже. Поля передаются слева направо.




***Code***

8 (Protocol-Reject).

***Identifier***

Поле Identifier **должно** изменяться для каждого передаваемого пакета Protocol-Reject.

***Rejected-Protocol***

Двухоктетное поле Rejected-Protocol содержит поле PPP Protocol из отвергнутого пакета.

***Rejected-Information***

Поле Rejected-Information содержит копию отвергнутого пакета, начиная с поля Information без заголовков канального уровня и FCS. Размер пакета Rejected-Protocol **должен** отсекаться в соответствии с согласованным партнерами значением MRU.

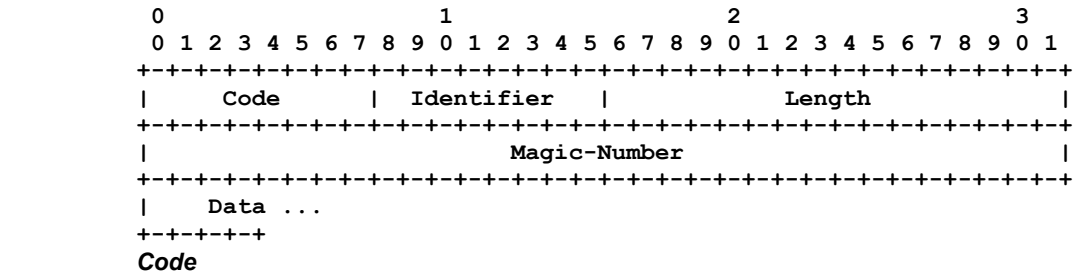
## 5.8 Echo-Request и Echo-Reply

LCP включает коды Echo-Request и Echo-Reply для предоставления канальному уровню механизма loopback, позволяющего проверить соединение в обоих направлениях. Это может служить для отладки, определения качества линии, тестирования производительности и многих других функций.

При получении пакета Echo-Request в состоянии LCP Opened **должен** передаваться отклик Echo-Reply.

Пакеты Echo-Request и Echo-Reply **должны** передаваться из состояния LCP Opened. Пакеты Echo-Request и Echo-Reply, полученные в состоянии, отличном от Opened, **следует** отбрасывать без уведомления.

Формат пакетов Echo-Request и Echo-Reply показан ниже. Поля передаются слева направо.




***Code***

9 (Echo-Request) или 10 (Echo-Reply).

***Identifier***

При передаче поле Identifier **должно** быть изменено при изменении содержимого поля Data или получении действительного отклика на предыдущий запрос. При повторе поле Identifier **может** сохраняться.

На приемной стороне поле Identifier копируется из пакета Echo-Request в пакет Echo-Reply.

***Magic-Number***

Четырехоктетное поле Magic-Number служит для детектирования наличия петли (loopback) на канале. Если не была согласована опция конфигурации Magic-Number, в поле Magic-Number **должно** помещаться значение 0. Дополнительная информация приведена в описании опции Magic-Number.

***Data***

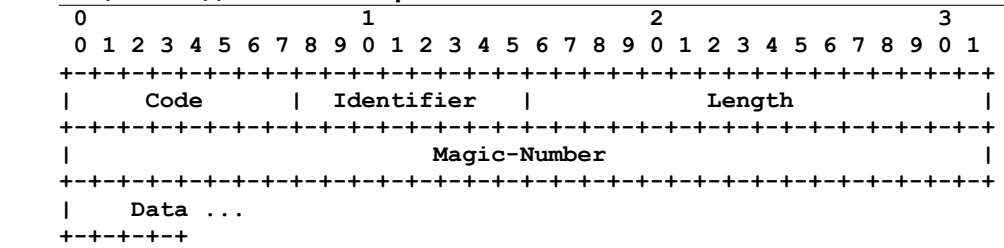
Поле Data (возможно пустое) содержит неинтерпретируемые данные для использования отправителем. Поле может включать произвольное двоичное значение. Окончание данных указывается значением поля Length.

## 5.9 Discard-Request

LCP включает код Discard-Request, предоставляющий канальному уровню механизм проверки соединения в направлении от локального узла к удаленному. Механизм полезен для отладки, тестирования производительности и множества других функций.

Пакеты Discard-Request **должны** передаваться только из состояния LCP Opened. Получатель пакета Discard-Request **должен** отбрасывать его без уведомления.

Формат пакета Discard-Request показан ниже. Поля передаются слева направо.




***Code***

11 (Discard-Request).

***Identifier***

Поле Identifier **должно** изменяться для каждого передаваемого пакета Discard-Request.

***Magic-Number***

Четырехоктетное поле Magic-Number служит для детектирования наличия петли (loopback) на канале. Если не была согласована опция конфигурации Magic-Number, в поле Magic-Number **должно** помещаться значение 0. Дополнительная информация приведена в описании опции Magic-Number.

***Data***

Поле Data (возможно пустое) содержит неинтерпретируемые данные для использования отправителем. Поле может включать произвольное двоичное значение. Окончание данных указывается значением поля Length.

# *6. Конфигурационные опции LCP*

Конфигурационные опции LCP позволяют согласовывать изменение принятых по умолчанию характеристик канала точка-точка. Если опция не включена в пакет Configure-Request, она будет иметь принятое по умолчанию значение.

Некоторые конфигурационные опции **могут** быть включены несколько раз. Результат этого зависит от опции и описан в определении опции (ни одна из включенных в эту спецификацию опций не может присутствовать многократно).

Конец списка конфигурационных опций определяется полем Length в пакете LCP.

Если не указано иное, все конфигурационные опции применяются в полудуплексном стиле - обычно это приемное направление с точки зрения отправителя Configure-Request.

Философия разработки.

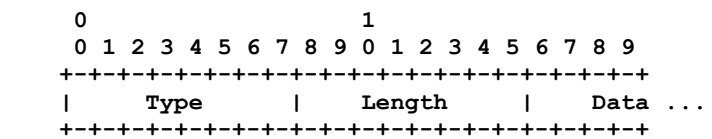
Опции указывают дополнительные возможности или требования реализации, запросившей опцию. Реализации, не понимающей никаких опций, **следует** быть интероперабельной с реализаций, которая поддерживает все опции.

Принятые по умолчанию значения заданы для каждой опции и это позволяют соединению корректно функционировать без согласования опций, хотя производительность может быть ниже оптимальной.

За исключением явно указанных случаев, подтверждение опции не требует от партнера выполнения каких-либо дополнительных действий, кроме принятых по умолчанию.

Передача принятых по умолчанию значения в пакете Configure-Request не требуется.

Формат конфигурационных опций показан ниже. Поля передаются слева направо.




***Type***

Однооктетное поле Type указывает тип опции. Актуальные в настоящий момент опции LCP указаны в свежей версии RFC «Assigned Numbers» [2].

0 резерв;

1 Maximum-Receive-Unit;

3 Authentication-Protocol;

4 Quality-Protocol;

5 Magic-Number;

7 Protocol-Field-Compression;

8 Address-and-Control-Field-Compression.

***Length***

Однооктетное поле Length уоказывает размер опции с учетом полей Type, Length и Data.

Если в пакете Configure-Request получена согласуемая опция с некорректным или непонятным полем Length, **следует** передать пакет Configure-Nak, включающий желаемую опцию с подходящими полями Length и Data.

***Data***

Поле Data (возможно пустое) включает данные конкретной опции. Формат и размер поля Data определяется полями Type и Length.

Если размер поля Data, указанный в поле Length, выходит на пределы поля Information, пакет отбрасывается без уведомления и обработки.

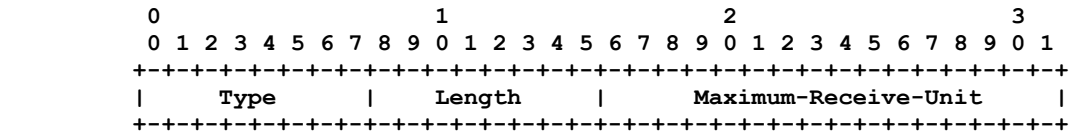
## 6.1 Maximum-Receive-Unit (MRU)

Эта конфигурационная опция может передаваться для информирования партнера о возможности приема более крупных пакетов или для запроса передачи более мелких пакетов.

По умолчанию используется значение 1500 октетов. Если запрошена передача более мелких пакетов, реализация **должна** сохранять возможность приема информационных полей размером 1500 октетов в случае потери синхронизации соединении.

Примечание для разработчиков. Эта опция служит для указания возможностей реализации. Партнер не обязан полностью использовать такую возможность. Например, при указании MRU в 2048 октетов, партнер не обязан передавать пакеты размером 2048 октетов. Партнер не обязан передавать пакет Configure-Nak для индикации того, что он будет передавать более мелкие пакеты, поскольку от реализации всегда требуется поддержка пакетов не менее 1500 октетов.

Формат опции Maximum-Receive-Unit показан ниже. Поля передаются слева направо.




***Type***

1

***Length***

4

***Maximum-Receive-Unit***

Двухоктетное поле Maximum-Receive-Unit указывает максимальное число октетов в полях Information и Padding. Размер не учитывает кадрирования, полей Protocol и FCS, а также биты и байты прозрачности.

## 6.2 Authentication-Protocol

На некоторых соединениях может быть желательно потребовать от партнера подтверждения его подлинности до начала обмена пакетами сетевого уровня.

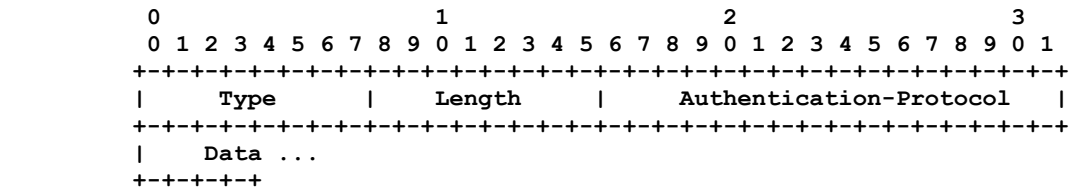
Эта опция обеспечивает способ согласования протокола аутентификации. По умолчанию аутентификация не нужна.

Реализациям **недопустимо** включать множество опций Authentication-Protocol в пакеты Configure-Request. Вместо этого **следует** пытаться сначала настроить наиболее желанный протокол. Если этот протокол будет указан в Configure-Nak, **следует** попытаться согласовать другой приемлемый протокол в следующем пакете Configure-Request.

Реализация, передающая Configure-Request показывает, что она ожидает аутентификации партнера. Если реализация передает Configure-Ack, это означает ее согласие на аутентификацию по указанному протоколу. Реализации, получившей Configure-Ack, **следует** ожидать от партнера аутентификации по подтвержденному протоколу.

Не задается требований полнодуплексной аутентификации или использования одного протокола для обоих направления. Применение своего протокола для каждого направления является совершенно нормальным. Однако это зависит от согласованных протоколов.

Формат опции Authentication-Protocol Configuration Option показан ниже. Поля передаются слева направо.




***Type***

3

***Length***

>= 4

***Authentication-Protocol***

Двухоктетное поле Authentication-Protocol указывает желаемый протокол аутентификации. Для этого поля применяются те же значения, которые помещаются в поле PPP Protocol для того же протокола аутентификации.

Действующие значения поля Authentication-Protocol указаны в RFC «Assigned Numbers» [2]. Выделенные в настоящее время значения приведены ниже.

| **Значение** | **Протокол** |
|---|---|
| c023 | Password Authentication Protocol (PAP) |
| c223 | Challenge Handshake Authentication Protocol (CHAP) |

***Data***

Поле Data (возможно пустое) содержит дополнительные данные, зависящие от протокола аутентификации.

## 6.3 Quality-Protocol

На некоторых соединениях желательно определить, когда и как часто канал отбрасывает данные. Этот процесс называется мониторингом качества канала.

Это опция позволяет согласовать использование конкретного протокола для мониторинга качества канала. По умолчанию мониторинг не применяется.

Реализация, передающая Configure-Request, показывает, что она ожидает получения данных мониторинга от своего партнера. Передача пакета Configure-Ack означает согласие реализации на применение этого протокола. Реализации, получившей Configure-Ack, **следует** ожидать от партнера пакетов подтвержденного протокола.

Не задается требований полнодуплексного мониторинга или использования одного протокола в обоих направлениях. Применение своего протокола для каждого направления является совершенно нормальным. Однако это зависит от согласованных протоколов.

Формат опции Quality-Protocol показан ниже. Поля передаются слева направо.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |        Quality-Protocol       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Data ...
+-+-+-+-+
```

***Type***

4

***Length***

>= 4

***Quality-Protocol***

Двухоктетное поле Quality-Protocol указывает желательный протокол мониторинга качества канала. Для этого поля применяются те же значения, которые помещаются в поле PPP Protocol для того же протокола мониторинга.

Действующие значения поля Quality-Protocol указаны в RFC «Assigned Numbers» [2]. Выделенные в настоящее время значения приведены ниже.

| **Значение** | **Протокол** |
|---|---|
| c025 | Link Quality Report |

***Data***

Поле Data (возможно пустое) содержит дополнительные данные, зависящие от протокола.

## 6.4 Magic-Number

Эта опция предоставляет метод детектирования шлейфовых петель (loopback) и других аномалий канального уровня. Опция **может** быть нужна для других опций конфигурации, например, Quality-Protocol. По умолчанию значение Magic-Number не согласовывается и вместо него используется 0.

До того, как эта опция будет запрошена, реализация **должна** выбрать значение Magic-Number. Рекомендуется задавать Magic-Number случайным образом для получения высокой вероятности выбора уникального значения. Хорошим способом получить уникальное случайное число является использование «затравки» (seed). Источниками уникальности могут быть серийные номера машин, аппаратные сетевые адреса, показания часов и т.п. Очень хорошей затравкой для случайных чисел служит точное измерение интервалов между физическими событиями, такими как получение пакетов в других присоединенных сетях, время отклика сервера или скорость нажатия клавиш человеком. Разумно использовать одновременно множество таких источников.

При получении Configure-Request с опцией Magic-Number, принятое значение Magic-Number сравнивается с Magic-Number в последнем пакете Configure-Request, переданном партнеру. Если значения Magic-Number различаются, это говорит об отсутствии петли на канале и Magic-Number **следует** подтвердить. Если два Magic-Number совпадают, это может говорить о наличии (не обязательно) петли на соединении, в результате чего полученный пакет Configure-Request является последним из отправленных. Для проверки этого **должен** быть передан пакет Configure-Nak с другим значением Magic-Number. Новые пакеты Configure-Request **не следует** передавать партнеру, пока этого не потребует обычный порядок обработки (т.е. до получения Configure-Nak или тайм-аута Restart).

Получение Configure-Nak с Magic-Number, отличным от переданного в последнем Configure-Nak, подтверждает отсутствие петли на канале и показывает уникальное значение Magic-Number. Если Magic-Number совпадает с переданным в последнем пакете Configure-Nak, вероятность наличия петли на канале увеличивается и **должно** быть выбрано новое значение Magic-Number. В любом случае новый пакет Configure-Request **следует** передавать с новым Magic-Number.

Если соединение действительно имеет петлю, последовательность (передача Configure-Request, прием Configure-Request, передача Configure-Nak, прием Configure-Nak) будет повторяться раз за разом. Если петли нет, последовательность может повториться несколько раз, но ее продолжительное повторение маловероятно. Скорее всего, значения Magic-Number, выбранные разными сторонами, быстро разойдутся и последовательность прервется. В таблице показаны вероятности совпадения Magic-Number на разных сторонах канала при однородном распределении.

| ***Числоконфликтов*** | ***Вероятность*** |
|---|---|
| 1 | $1/2^{32}$=2,3 E-10 |
| 2 | $1/(2^{32})^2$=5,4 E-20 |
| 3 | $1/(2^{32})^3$=1,3 E-29 |

Для такой дивергенции нужен хороший источник уникальности или случайности. Если такого источника нет, рекомендуется не применять эту опцию - пакеты Configure-Request пакеты с этой опцией **не следует** передавать, а все опции Magic-Number от партнера **следует** применять после подтверждения или отбрасывать. В этом случае петли на канале не удастся надежно детектировать, хотя для партнера эта возможность сохранится.

Если реализация передает Configure-Request с опцией Magic-Number, ей **недопустимо** отвечать пакетом Configure-Reject на получение Configure-Request с опцией Magic-Number. Если реализация желает использовать Magic-Number, то она **должна** разрешать это своему партнеру. Если реализация получила Configure-Reject в ответ на Configure-Request, это может означать лишь отсутствие петли на канале и отказ партнера от использования Magic-Number. В этом случае реализации **следует** вести себя как при успешном согласовании опций (получение Configure-Ack).

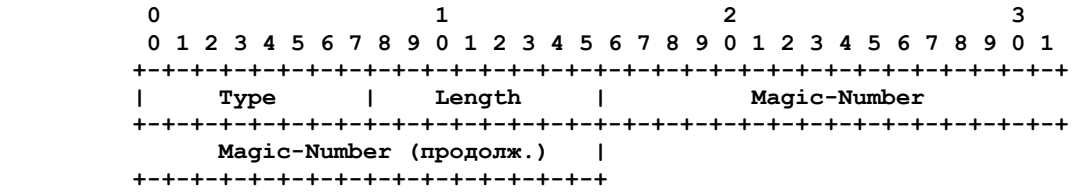
Magic-Number может также применяться для детектирования петель при нормальной работе и в процессе согласования конфигурационных опций. Все пакеты LCP Echo-Request, Echo-Reply и Discard-Request имеют поле Magic-Number. Если значение Magic-Number согласовано, реализация **должна** передавать такие пакеты с согласованным значением в поле Magic-Number.

Поле Magic-Number в таких пакетах **следует** проверять при получении. Все полученные поля Magic-Number **должны** быть равны 0 или совпадать с уникальным Magic-Number партнера, в зависимости от согласования сторонами опции Magic-Number.

Получение Magic-Number, совпадающего с согласованным локальным значением Magic-Number говорит о наличии петли на канале. Получение Magic-Number, отличного от согласованного локального значения, согласованного значения Magic-Number партнера или нуля при отсутствии согласования говорит о том, что соединение было настроено для взаимодействия с другим партнером.

Процедуры для восстановления для любого из этих случаев не описываются и могут отличаться в разных реализациях. Пессимистическим вариантом является допущение события LCP Down. Следующее событие Open начнет процесс повторной организации соединения, которой не может быть завершен без устранения петли и согласования Magic-Number. Более оптимистической процедурой (при наличии петли) является начало передачи пакетов LCP Echo-Request, продолжающейся по получения приемлемого отклика Echo-Reply, указывающего на устранение петли.

Формат опции Magic-Number показан ниже. Поля передаются слева направо.




***Type***

5

***Length***

6

***Magic-Number***

Четырехоктетное поле Magic-Number содержит значение, которое с большой вероятностью различается у партнеров соединения. Нулевое значение Magic-Number неприемлемо и в ответ **должен** передаваться пакет Nak, если не требуется отвергнуть пакет совсем.

## 6.5 Protocol-Field-Compression (PFC)

Эта опция позволяет согласовать метод сжатия поля PPP Protocol. По умолчанию, все реализации **должны** передавать пакеты с двухоктетными полями Protocol.

Номера протоколов PPP выбраны так, чтобы некоторые значения можно было сжать до 1 октета, что будет явно отличать их от двухоктетных полей. Эта опция служит для информирования партнера о возможности принимать поля Protocol размером 1 октет.

Как отмечено выше, поле Protocol использует механизм расширения, согласующийся с ISO 3309 для поля Address. Младший бит (LSB[1]) каждого октета служит для индикации расширения поля Protocol. Значение 0 в LSB показывает, что поле Protocol продолжается в следующем октете, а 1 — указывает последний октет поля Protocol. Отметим, что полю может предшествовать любой число октетов по значением 0 и номер протокола при этом не изменится (например, для протокола 3 могут использоваться представления 00000011 и 00000000 00000011).

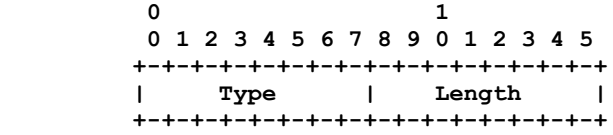
На низкоскоростных каналах желательно сохранять полосу пропускания, передавая меньше избыточных данных. Опция Protocol-Field-Compression обеспечивает компромисс между простотой реализации и эффективностью использования полосы пропускания. Если эта опция согласованна, может применяться механизм расширения ISO 3309 для сжатия поля Protocol в один октет вместо двух. Для большинства пакетов такое сжатие возможно, поскольку значения поля Protocol меньше 256.

Сжатые поля Protocol **недопустимо** передавать, пока не согласована эта опция. При согласованной опции реализации PPP **должны** воспринимать PPP пакеты как с двумя, так и с одним октетом в поле Protocol, а различать такие пакеты **недопустимо**.

Поле Protocol никогда не сжимается при передаче пакетов LCP, это гарантирует их однозначное распознавание.

При сжатых полях Protocol значение FCS канального уровня рассчитывается для сжатого, а не исходного кадра.

Формат опции Protocol-Field-Compression показан ниже. Поля передаются слева направо.




***Type***

7

***Length***

2

## 6.6 Address-and-Control-Field-Compression (ACFC)

Эта опция позволяет согласовать метод сжатия полей канального уровня Address и Control. По умолчанию все реализации **должны** передавать кадры с полями Address и Control, соответствующими методу кадрирования.

Поскольку для соединений точка точка значения этих полей обычно постоянны, поля легко сжать. Эта опция передается для информирования партнера о возможности воспринимать сжатые поля Address и Control.

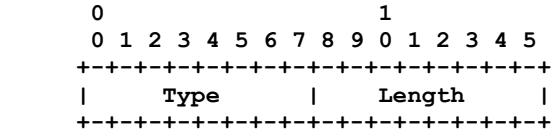
Если сжатый кадр был получен без согласованной опции Address-and-Control-Field-Compression, реализация **может** отбросить кадр без уведомления.

Поля Address и Control **недопустимо** сжимать для пакетов LCP пакет. Это правило гарантирует однозначное распознавание пакетов LCP.

[1]Least Significant Bit.

При сжатых полях Address и Control значение FCS на канальном уровне рассчитывается для сжатого кадра.

Формат опции Address-and-Control-Field-Compression показан ниже. Поля передаются слева направо.




***Type***

8

***Length***

2

## *Вопросы безопасности*

Вопросы безопасности кратко рассмотрены в разделах, описывающих фазу Authentication, событик Close и опцию Authentication-Protocol.

## *Литература*

[1] Perkins, D., "Requirements for an Internet Standard Point-to-Point Protocol", RFC 1547, Carnegie Mellon University, December 1993.

[2] Reynolds, J., and Postel, J., "Assigned Numbers", STD 2[1], RFC 1340, USC/Information Sciences Institute, July 1992

## *Благодарности*

Этот документ подготовлен рабочей группой Point-to-Point Protocol под эгидой IETF. Комментарии следует направлять в список рассылки ietf-ppp@merit.edu.

Значительная часть текста документа заимствована из требований рабочей группы [1], а также RFC 1171 и RFC 1172 от Drew Perkins из университета Carnegie Mellon и Russ Hobby из Калифорнийского университета в Дэвисе.

William Simpson отвечал за согласованность терминологии и философии, а также за устройство машин состояния фаз и согласования.

Множество людей потратили много времени разработку протокола PPP. Полный список людей слишком велик для перечисления здесь, но некоторых следует отметить. Это Rick Adams, Ken Adelman, Fred Baker, Mike Ballard, Craig Fox, Karl Fox, Phill Gross, Kory Hamzeh, David Kaufman, Mark Lewis, John LoVerso, Bill Melohn, Mike Patton, Greg Satz, John Shriver, Vernon Schryver и Asher Waldfogel.

Отдельная благодарность Morning Star Technologies за предоставление компьютерных ресурсов и сетевого доступа при написании этой спецификации.

**Адрес руководителя**

С рабочей группой можно связаться через ее руководителя

**Fred Baker**
Advanced Computer Communications
315 Bollay Drive
Santa Barbara, California 93117
fbaker@acc.com

**Адрес редактора**

Вопросы по этому документу могут быть направлены по приведенному ниже адресу

**William Allen Simpson**
Daydreamer
Computer Systems Consulting Services
1384 Fontaine
Madison Heights, Michigan 48071
Bill.Simpson@um.cc.umich.edu bsimpson@MorningStar.com

**Перевод на русский язык**

Николай Малых

nmalykh@protocols.ru

[1]В соответствии с RFC 3232 документ «Assigned Numbers» утратил силу и заменен публикуемыми на сайте документами. *Прим. перев.*