Internet Engineering Task Force (IETF)
Request for Comments: 8085
BCP: 145
Obsoletes: 5405
Category: Best Current Practice
ISSN: 2070-1721

L. Eggert
NetApp
G. Fairhurst
University of Aberdeen
G. Shepherd
Cisco Systems

March 2017

UDP Usage Guidelines

Рекомендации по использованию UDP

Аннотация

Протокол пользовательских дейтаграмм (User Datagram Protocol или UDP) обеспечивает минимальный транспорт передачи сообщений, не наследующий механизмов контроля перегрузок. В этом документе даны рекомендации по применению UDP для разработчиков приложений, туннелей и других протоколов на основе UDP. Основной внимание уделено контролю насыщения (перегрузки), но документ включает также рекомендации по другим вопросам, включая размер сообщений, гарантии доставки, контрольные суммы, прохождение через промежуточные устройства, использование явных уведомлений о перегрузке (Explicit Congestion Notification или ECN), коды дифференцированного обслуживания (Differentiated Services Code Point или DSCP) и порты.

Поскольку контроль перегрузок очень важен для стабильной работы Internet, приложения и другие протоколы, выбравшие UDP в качестве транспорта Internet, должны реализовать механизмы предотвращения перегрузок и обеспечения той или иной беспристрастности для одновременного трафика. Это может потребовать реализации дополнительных механизмов в зависимости от способа применения UDP.

Некоторые рекомендации связаны с устройством других протоколов (например, протоколов, работающих непосредственно по IP или через туннели IP), особенно в случаях, когда эти протоколы сами не обеспечивают контроля перегрузок.

Этот документ отменяет RFC 5405 и добавляет рекомендации по групповой адресации в UDP.

Статус документа

Документ относится к категории обмена опытом (Best Current Practice).

Документ является результатом работы IETF[1] и представляет согласованный взгляд сообщества IETF. Документ прошел открытое обсуждение и был одобрен для публикации IESG[2]. Дополнительную информацию о стандартах Internet можно найти в разделе 2 в RFC 7841.

Информация о текущем статусе документа, найденных ошибках и способах обратной связи доступна по ссылке http://www.rfc-editor.org/info/rfc8085.

Оглавление

[1]Internet Engineering Task Force.
[2]Internet Engineering Steering Group.

1. Введение

Протокол UDP [RFC768] предоставляет минимальные транспортные услуги без гарантии доставки в стиле best-effort (как получится) для сообщений от приложения или другого протокола (например, туннельного) на основе протокола IP. В оставшейся части документа приложения и протоколы, использующие транспорт, называются просто приложениями.

По сравнению с другими транспортными протоколами UDP и его вариант UDP-Lite [RFC3828] уникальны с том, что не организуют сквозного соединения между взаимодействующими системами. Поэтому с коммуникациями UDP не связаны издержки на организацию и завершение соединений и связь с состояниями конечных систем минимальна. В результате UDP предлагает очень эффективный транспорт для некоторых приложений.

Второй уникальной характеристикой UDP является отсутствие встроенных механизмов контроля перегрузок. На многих платформах приложения могут передавать дейтаграммы UDP с скоростью линии (среды) канального интерфейса платформы, которая часто значительно превышает доступную пропускную способность сквозного пути и это вносит вклад в перегрузку на пути. В [RFC2914] описан опыт контроля перегрузок в Internet. Документ указывает две основных причины важности механизмов контроля перегрузок для стабильной работы Internet.

1. Предотвращение перегрузочных коллапсов, т. е. состояний, где рост загрузки сети ведет к снижению объема полезной работы, выполняемой сетью.

2. Обеспечение той или иной степени беспристрастности, т. е. возможности разным потокам «по справедливости» делить пропускную способность сети.

Поскольку UDP не обеспечивает контроля перегрузок, приложения, применющие UDP для Internet-коммуникаций, реализуют подходящие механизмы предотвращения перегрузочного коллапса и обеспечения беспристрастности. В [RFC2309] рассматривается опасность не отвечающих на перегрузку потоков и отмечено, что «все потоковым приложениям на основе UDP следует включать эффективный механизм предотвращения перегрузок». [RFC7567] подтверждает это заявление. Это требование важно даже для приложений, не применяющих UDP для передачи потоков. Кроме того, транспортировка с контролем перегрузок обеспечивает преимущества самим приложениям, поскольку может снижать самоиндуцированную потерю пакетов, минимизировать повторы передачи и в результате снизить задержки. Контроль перегрузок важен даже при сравнительно низкой скорости передачи. Например, приложение, создающее 5 дейтаграмм UDP по 1500 байтов в секунду, может уже выйти за пределы пропускной способности пути 56 Кбит/с. Для более скоростных приложений, возможно с неограниченной скоростью, контроль перегрузок становится жизненно важным для предотвращения коллапса насыщения и обеспечения той или иной беспристрастности. В разделе 3 приведено множество простых рекомендация для разработчиков таких приложений.

Дейтаграмма UDP передается в одном пакете IP и поэтому ограничена в размере 65507 байтами поля данных в IPv4 и 65527 байтами в IPv6. Передача больших пакетов IP обычно требует фрагментации на уровне IP, что снижает надежность и эффективность коммуникаций, поэтому ее следует избегать. IPv6 обеспечивает опцию для передачи больших пакетов (jumbogram) без фрагментации когда все канальные уровни на пути поддерживают это [RFC2675]. Некоторые из рекомендаций раздела 3 описывают способ определения подходящего для приложения размера сообщений. В других разделах документа даны рекомендации по обеспечению надежности доставки, контрольным суммам, прохождению через промежуточные устройства и использованию групповой адресации.

В документе представлены руководства и рекомендации. Хотя предполагается, что большинство приложений UDP следует этим рекомендациям, у конкретных приложений могут быть причины отсутпать от приведенных здесь рекомендаций . В таких случаях разработчикам **рекомендуется** включить соответствующую часть этого документа в техническую спецификацию приложения или протокола и сопроводить разъяснением причин своего выбора.

[RFC5405] был предназначен в качестве рекомендации лишь для unicast-приложений, тогда как этот документ содержит рекомендации, позволяющие потокам UDP использовать групповую адресацию IP, anycast и широковещание а также рекомендации для приложений, применяющих туннели UDP для потоков IP.

Хотя документ посвящено конкретно применению UDP, дух некоторых рекомендаций позволяет использовать их в других приложениях и протоколах с передачей сообщений (особенно вопросы контроля перегрузок, размера сообщений и гарантии доставки). Примеры включают сигнальные и управляющие приложения, туннели,

предпочитающие работать напрямую по протоколу IP, путем регистрации своего номера протокола IP в IANA. Предполагается, что этот документ предоставит полезные сведения разработчикам таких приложений и протоколов.

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

3. Рекомендации по использованию UDP
Характеристики путей Internet могут меняться в широких пределах по значениям задержки, доступной пропускной способности, уровню насыщения, вероятности изменения порядка, поддерживаемых размеров и частоты потерь. Кроме того, любой путь в Internet может иметь разные характеристики в разное время. Поэтому приложениям, где может использоваться Internet, **недопустимо** принимать какие либо допущения о характеристиках конкретного пути. Вместо этого **должны** применяться механизмы, которые позволят безопасно работать в сильно различающихся условиях. Обычно это требует осторожной проверки текущего состояния пути Internet, через который будет организована связь, для достижения устраивающего поведения, которое будет достаточно беспристрастным к другому трафику на том же пути.

Эти механизмы сложно реализовать корректно. Для большинства приложений применение одного из имеющихся транспортных протоколов IETF является простейшим вариантом обретения нужных механизмов. Это позволяет избежать проблем, с которыми сталкиваются новые протоколы на основе IP, разрабатываемые для Internet, где могут встречаться промежуточные устройства, поддерживающие лишь транспорт TCP и UDP. Поэтому альтернативой описанному в этом документе использованию протокола UDP **рекомендуется** выбирать один из транспортных протоколов IETF, такой как TCP [RFC793], SCTP[1] [RFC4960], SCTP-PR[2] [RFC3758], DCCP[3] [RFC4340] с их разными типами контроля перегрузок [RFC4341][RFC4342][RFC5622] или транспортные протоколы, которые будут разработаны в IETF. SCTP с инкапсуляцией в UDP [RFC6951] и DCCP [RFC6773] могут обеспечивать поддержку работы через межсетевые экраны и другие промежуточные устройства, где естественный протокол не поддерживается.

При корректном использовании эти полнофункциональные протоколы не являются столь «тяжеловесными» как часто утверждают. Например, алгоритмы TCP непрерывно совершенствовались десятки лет и достигли высокого уровня эффективности и корректности, который трудно получить в специализированных механизмах прикладного уровня. Кроме того, многие реализации TCP позволяют приложению настроить соединения в соответствии со своими потребностями. Например, можно отключить алгоритм Nagle [RFC1122], сократив задержки за счет более частой (но контролируемой в части перегрузок) передачи пакетов. Другим примером является механизм TCP SYN cookie [RFC4987], доступный на многих платформах. TCP с SYN cookie не требует от сервера поддержки состояния на уровне соединения до организации соединения. TCP также требует от закрывающей соединение стороны поддерживать состояние TIME-WAIT, которое предотвращает конфликты задержанных сегментов соединения с сегментами нового экземпляра. Приложения, которые знают о таком поведении и разработаны для него, могут менять поддержку состояния TIME-WAIT для экономии ресурсов, выбирая сторону, закрывающую соединение TCP [FABER]. Встроенная функция TCP для проверки пропускной способности и определения размера максимального передаваемого блока, поддерживаемого на пути (PMTU) обеспечивает эффективную передачу данных с быстрой компенсацией задержки организации соединения в случаях обменов данными, не ограничивающихся несколькими сегментами.

3.1. Рекомендации по контролю перегрузок
Если приложение или протокол отказывается от применения транспортного протокола с контролем перегрузок, ему **следует** контролировать скорость передачи дейтаграмм UDP хосту-получателю, чтобы выполнить требования [RFC2914]. Важно подчеркнуть, что приложению **следует** выполнять контроль перегрузки для всего трафика UDP, отправляемого адресату, независимо от способа генерации этого трафика. Например, приложению, использующему процесс с множественным ветвлением или иной вариант с множеством сокетов для генерации трафика UDP, **следует** реализовать контроль перегрузок для всего агрегата трафика.

Некоторые подходы к контролю перегрузок рассмотрены ниже в этом разделе. Здесь рассматриваются базовые вопросы с акцентом на индивидуальную и anycast-адресацию [RFC1546]. Не все описанные ниже подходы подходят для каждого приложения, передающего дейтаграммы UDP. В параграфе 3.1.2 рассматриваются опции контроля перегрузок для приложений с большим объемом передаваемых по протоколу UDP данных. Такие приложения могут использовать схемы, которые выбирают путь за несколько интервалов кругового обхода (round-trip) при обмене данными, где определяется скорость передачи, которую путь может поддерживать при текущей загрузке. В параграфе 3.1.3 рассмотрены варианты контроля перегрузок для приложения с небольшим объемом передаваемых данных. Поскольку в таких случаях сложно определить приемлемую скорость передачи итеративным методом (мало данных), нужен иной механизм контроля перегрузок. В параграфе 3.1.11 рассматривается контроль перегрузки при использовании протокола UDP для туннелирования. В разделе 4 даны дополнительные рекомендации для широковещательного и группового трафика.

Важно подчеркнуть, что контроль перегрузок не должен выглятель как дополнение к готовому приложению. Многие из рассматриваемых ниже механизмов требуют поддержки от приложения для корректной работы. Разработчикам приложений нужно включать механизмы контроля перегрузок в структуру приложения, подобно реализации в приложении механизмов защиты.

В прошлом в рамках IETF исследовались интегрированные механизмы контроля перегрузок, которые работали с агрегатами трафика между парой хостов, т. е. такие схемы, как менеджер перегрузок [RFC3124], где активные сессии могли использовать общую информацию о насыщении независимым от транспортного протокола способом. Такие механизмы не получили распространения, но они могут упростить разработку механизмов контроля перегрузок для сессий UDP в соответствии с требованиями [RFC2914].

[1]Stream Control Transmission Protocol - протокол управления потоковой передачей.
[2]SCTP Partial Reliability Extension - SCTP с частичными гарантиями доставки.
[3]Datagram Congestion Control Protocol - протокол дейтаграмм с контролем перегрузок.

3.1.1. Рекомендации по таймерам

Понимание задержки между взаимодействующими конечными точками обычно очень важно для реализации эффективного контроля перегрузок в протоколах и приложениях. Оценка задержки может использоваться во многих функциях протокола, таких как расчет скорости передачи с контролем перегрузки, инициирование повторной передачи и обнаружение потери пакетов. Дополнительные функции, такие как определение интервалов зондирования пути, отправки сообщений keep-alive, определения качества полученной информации, проверка отклика удаленной конечной точки на запрос выполнения действия, обычно работают с большими по сравнению с контролем перегрузок временными интервалами, поэтому здесь не рассматриваются.

В качестве общей рекомендации этот документ указывает, что приложениям **следует** применять имеющиеся методы контроля перегрузок и заданные для них методы оценки задержек (см. следующий параграф) Ниже приведены рекомендации для приложений, которым нужен свой механизм оценки задержек.

Рекомендации даны с точки зрения задержки и времени кругового обхода, поскольку в некоторых случаях нужно характеризовать лишь задержку в сети (например, TFRC[1] [RFC5348]), а в других требуется включение времени, которое нужно конечной точке для предоставления отклика (например, определение момента повторной передачи сообщения).

Задержка между конечными точками обычно является динамическим свойством, поэтому в оценках **следует** представлять то или иное усреднение множества недавних измерений для учета вариаций. Применение усреднения EWMA[2] доказало свою пользу для таких задач (например, в TCP [RFC6298] и TFRC [RFC5348]).

Независимую оценку задержки **следует** поддерживать для каждого адресата, с которым взаимодействует конечная точка.

Выборку задержек **недопустимо** делать из неоднозначных транзакций. Каноническим примером служит протокол, который повторно передает данные, но затем не модет определить, какая из копий была подтверждена. Такие неоднозначности делают корректный расчет задержки проблематичным (см. обсуждение алгоритма Karn в [RFC6298]). Это требование обеспечивает надежную оценку задержки отправителем без некорректных измерений.

Когда оценка задержки служит для установки таймера, обеспечивающего обнаружение потерь (с повтором или без него) завершение отсчета таймера **должно** считаться индикацией перегрузки в сети, вызывающим приспосабливать скорость передачи к безопасному умеренному значению (например, TCP снижает размер окна перегрузки до одного сегмента [RFC5681]).

Некоторым приложениям нужна предварительная оценка задержки перед тем, как можно будет эмпирически определить задержку между конечными точками. Например, при первом запуске таймера повторной передачи нужно задать начальное значение для защиты сообщений, переданных между конечными точками до того, как они определят задержку. Это начальное значение обычно **следует** устанавливать как можно большим для данного приложения. Например, при отсутствии каких-либо сведений о задержке в пути TCP требует установки для параметра RTO[3] значения не меньше 1 секунды 1 [RFC6298]. Приложениям UDP **следует** использовать аналогичную начальную оценку задержки (1 сек.). Меньшие значения могут создавать проблемы (см. анализ данных в Приложении к [RFC6298]).

3.1.2. Приложения с большим объемом данных

Приложениям, передающим партнеру большой объем данных (более нескольких дейтаграмм за RTT) по протоколу UDP, **следует** реализовать механизм TFRC [RFC5348] - похожий на TCP контроль перегрузок на основе окна - или обеспечить соответствие принципам контроля перегрузок иным способом.

Механизм TFRC разработан для обеспечения контроля перегрузок и беспристрастности совместимым с другими транспортными протоколами IETF способом. Если приложение реализует TFRC, ему не нужно следовать остальным рекомендациям этого параграфа, поскольку они уже выполнены в TFRC, но **следует** выполнять рекомендации последующих параграфов раздела 3.

Приложениям с большим объемом данных, отказавшимся от реализации TFRC или использования окна в стиле TCP, **следует** реализовать схему контроля перегрузок, обеспечивающую использование пропускной способности (емкости), беспристрастно разделяемой с TCP (по порядку значений). В разделе 2 [RFC3551] отмечено, что приложениям **следует** контролировать скорость потери пакетов, чтобы она находилась в допустимых пределах. Потеря пакетов считается приемлемой, если поток TCP по тому же пути через сеть и при таких же условиях будет достигать средней пропускной способности не меньше чем у потока UDP при измерении в разумный интервал времени. Сравнение с TCP невозможно задать точно и оно предназначено для оценки порядка значений по интервалу времени и пропускной способности. Применимы также рекомендации параграфа 3.1.1 для таймеров.

Некоторые приложения с большим объемом данных могут отказаться от использования механизмов контроля перегрузок, полагаясь на передачу по пути с зарезервированной пропускной способностью (см. параграф 3.1.9). Это может быть приемлемо для некоторого подмножества сетевых сред, но не подходит в качестве безопасной практики работы в Internet. Когда трафик UDP от таких приложений попадает на пути Internet без резервирования пропускной способности, это может существенно влиять на остальной трафик в том же пути и даже вызывать коллапс перегрузки. Приложениям, поддерживающим неконтролируемое и неадаптивное поведение, **не следует** использовать его по умолчанию, а **следует** требовать от пользователя явно включать такой режим работы, а также **следует** проверять резервирование для приложения пропускной способности пути.

3.1.3. Приложения с малым объемом данных

Когда приложения, передающие небольшое число дейтаграмм UDP, реализуют TFRC или иные схемы контроля перегрузок, упомянутые в параграфе 3.1.2, сеть получает незначительное преимущество, поскольку эти механизмы эффективны лишь для длительных передач.

[1]TCP-Friendly Rate Control - дружественный к TCP контроль скорости.
[2]Exponentially Weighted Moving Average - экспоненциально взвешенное скользящее среднее значение.
[3]Retransmission Timeout - тайм-аут повторной передачи.

Приложениям с небольшим объемом данных UDP **следует** сохранять контроль за своей передачей, не отправляя получателю в среднем более одной дейтаграммы UDP в течение интервала RTT. Подобно рекомендациям [RFC1536], приложению **следует** поддерживать оценку RTT для любого адресата, с которым он взаимодействует, используя методы, указанные в параграфе 3.1.1.

Некоторые приложения не могут поддерживать надежную оценку RTT до адресата. Для таких приложений не нужно или невозможно использование протокольных таймеров для измерения RTT (параграф 3.1.1). Следует рассмотреть два случая, описанных ниже.

1. Число дейтаграмм UDP между приложениями слишком мало для статистически точной оценки RTT, но можно отслеживать надежность доставки (параграф 3.3). Такие приложения **могут** использовать предопределенный интервал передачи, который экспоненциально увеличивается в случае потери пакетов. TCP задает начальный интервал в 1 секунду [RFC6298] и это значение **рекомендуется** как начальное для приложений UDP. Некоторые приложения с очень малым объемом данных (такие как SIP [RFC3261] и GIST[1] [RFC5971]) используют интервал в 500 мсек и меньше, что во многих случаях может быть проблематичным. Как и в предыдущем случае начальный интервал не является максимально возможным тайм-аутом (параграф 3.1.1).

2. Приложение не может поддерживать оценку RTT для адресата, поскольку тот не передает пакетов. Таким приложениям **не следует** передавать более 1 дейтаграммы UDP каждые 3 секунды и **следует** по возможности передавать еще реже, поскольку сокращение интервала во многих случаях будет создавать проблемы. Отметим, что скорость передачи здесь должна быть более умеренной, нежели в предыдущих случаях, поскольку отсутствие возвратного трафика не позволяет детектировать потерю пакетов (т. е. перегрузку) и приложение не может снижать экспоненциально скорость отправки.

3.1.4. Приложения с двухсторонними коммуникациями

Приложениям с двухсторонним обменом **следует** реализовать контроль перегрузок для обоих направлений. Например, для приложений «клиент-сервер», работающих в стиле «запрос-отклик», клиенту **следует** контролировать перегрузки при отправке запросов, а серверу **следует** контролировать перегрузки для своих откликов клиенту. Контроль в прямом и обратном направлении не согласуется и приложению **следует** независимо детектировать и обрабатывать сигналы перегрузки каждого направления или ограничивать частоту новых и повторяемых запросов на основе подтвержденных откликов для пути всего кругового обхода.

3.1.5. Влияние RTT и учета потерь на контроль перегрузок

Транспорт, подобный TCP, SCTP и DCCP, обеспечивает своевременное обнаружение перегрузки, что ведет к незамедлительному снижению максимальной скорости передачи в случае перегрузки. Эта реакция обычно наблюдается в интервале 1-2 RTT после обнаружения потери или перегрузки. Приложениям UDP **следует** реализовать схему контроля перегрузок, обеспечивающую быструю реакцию на сигналы перегрузки (например, снижением скорости в следующем интервале RTT после сигнала).

Работа алгоритмов контроля перегрузок для UDP может существенно отличаться от TCP. Это включает элементы контроля перегрузки, реагирующие на временные рамки, соответствующие приложению, которые не могут эффективно работать в модели TCP с изменением скорости в каждом интервале RTT.

Приложения с низким или меняющимся значением RTT особенно уязвимы к ошибкам выборки (например, в результате шумов или неточности таймера). Это предполагает необходимость усреднения результатов оценки потери и перегрузки, а также RTT за достаточно длительный интервал, однако это может вносить дополнительную задержку при обнаружении перегрузки. Некоторые приложения могут не реагировать сразу снижением скорости в силу разных причин, включая периодичность измерения RTT и потерь (например, с использованием RTCP), потребность в дополнительном времени для фильтрации данных или возможность приложения менять скорость лишь в предопределенные моменты (как в некоторых видео-кодеках).

При разработке алгоритма контроля перегрузок нужно учесть общее время, требуемое для снижения нагрузки после отсутствия обратной связи или обнаружения перегрузки. Приложение, где последнее измерение RTT меньше фактического RTT или измеренная частота потерь меньше текущей, может переоценить допустимую скорость передачи. В результате может возникнуть перегрузка для этого приложения или других потоков на том же пути, что может способствовать коллапсу насыщения. Таких ситуаций следует избегать.

Контролю перегрузок, разработанному для UDP, **следует** как можно быстрее реагировать на индикацию перегрузки, а также **следует** учитывать при выборе новой скорости частоту потерь и время отклика. Реализованной схеме контроля перегрузок **следует** обеспечивать использование пропускной способности (емкости), сравнимое по порядку величины с TCP, чтобы не истощать другие потоки на том же пути с узким местом.

3.1.6. Синхронизация и снижение пиков

Приложениям UDP **следует** поддерживать механизм регулировки пиков при передаче пакетов приложения в сеть. Многие реализации TCP и SCTP обеспечивают механизмы, препятствующие создание отправителем продолжительных пиков передачи со скоростью линии, поскольку это вызывает преждевременные потери в приложениях, трафик которых проходит через те же узкие места в сети. Показано, что использование синхронизации с TCP [ALLMAN] улучшает сосуществование потоков TCP с другими потоками. Необходимость избегать чрезмерных пиков при передаче следует также отмечать в спецификации приложений (например, [RFC7143]).

Даже небольшие потоки UDP могут выигрывать от синхронизации пакетов. Например, приложению, передающему 3 копии пакета для отказоустойчивости, **рекомендуется** отправлять эти пакеты за несколько интервалов RTT для снижения вероятности потери всех трех в одном случае перегрузки (или иного события, такого как сильный пик).

3.1.7. Явный контроль перегрузок

Приложения Internet могут использовать явный контроль перегрузки (ECN) [RFC3168] для улучшения поддерживаемых ими услуг [RFC8087]. Транспорт Internet, такой как TCP, поддерживает множество механизмов, потребных для

[1]General Internet Signaling Transport - сигнальный транспорт общего назначения в Internet.

использования ECN. Для работы ECN применяются специальные коды ECT(0) или ECT(1) в заголовках IP передаваемых пакетов. Это указывает поддерживающим ECN сетевым устройствам (маршрутизаторы и пр.), что они могут помечать (устанавливать код CE[1]) codepoint) пакеты IP вместо их отбрасывания для сигнализации о наступающей перегрузке.

Приложения UDP также могут получить пользу от ECN, если API поддерживает ECN и реализует нужные механизмы. Набор механизмов, требуемых для использования ECN по протоколу UDP, приведен ниже.

- Отправитель **должен** обеспечить метод определения (например, согласование) способности соответствующего приложения обеспечивать отклики ECN для используемого с совместимым методом ECN.
- Получатель, разрешающий использование ECN для порта UDP, **должен** проверять поле ECN для каждой принятой на этом порту дейтаграммы UDP.
- Принимающее приложение должно обеспечивать отклики на перегрузку передающему приложению. Оно **должно** сообщать о наличии дейтаграмм, принятых с маркировкой CE с помощью механизма, помещающего эти сведения о перегрузке в пакеты для передающего приложения. Отклики **могут** также сообщать о наличии пакетов ECT(1) и ECT(0)/не-ECT [RFC7560]. Методы для TCP заданы в [RFC3168] и [RFC7560].
- Передающее дейтаграммы с поддержкой ECN приложение **должно** обеспечивать должную реакцию на получение откликов, указывающий возникновение перегрузки. Это должно вызывать снижение скорости передачи механизмом контроля перегрузок UDP (параграф 3.1) не слабее реакции TCP в таких же условиях.
- Отправителю **следует** детектировать сетевые пути без корректной поддержки поля ECN и при их обнаружении умеренно реагировать на перегрузку или даже отказываться от применения ECN [RFC8087]. Метод должен быть устойчив к изменениям на сетевых путях, которые могут происходить в течение сессии.
- Отправителям настоятельно рекомендуется детектированить и должным образом реагировать на некорректное поведение получателей, которые отказываются сообщать о пакетах с маркерами CE [RFC8087].

В [RFC6679] даны рекомендации и пример такой поддержки в описании метода, позволяющего использовать ECN для приложений UDP, применяющих протокол RTP (Real-Time Protocol). Приложениям, не способным обеспечить такой набор механизмов, но желающим воспользоваться преимуществами ECN, настоятельно рекомендуется использовать транспортный протокол с поддержкой ECN (например, TCP).

3.1.8. Модель дифференцированных услуг

Приложение UDP может применять модель качества обслуживания (Quality of Service или QoS) с дифференцированными услугами (DiffServ). Для включения дифференцированной обработки отправитель UDP устанавливает код в поле DSCP [RFC2475] передаваемых в сеть пакетов. Обычно пара портов отправителя и получателя UDP будет задавать одно значение DSCP для всех пакетов потока, но можно использовать несколько DSCP, как описано ниже. Значение DSCP может выбираться из небольшого набора фиксированных значений (коды селекторов класса), из рекомендуемых спецификацией поэтапного поведения (Per Hop Behavior или PHB) значений или из локального набора, имеющего смысл в сети с поддержкой DiffServ. В общем случае пакеты могут проходить через множество сетей между отправителем и получателем.

При установке отличного от принятого по умолчанию значения DSCP приложение должно понимать, что маркировка DSCP может быть изменена или удалена на пути между отправителем и получателем. Это влияет на использующие DSCP приложения. В частности, приложениям **следует** не полагаться на определенную трактовку кодов в сети, а взамен нужно реализовать методы контроля перегрузок для определения текущей скорости передачи с учетом условий насыщения в сети.

В [RFC7657] описано влияние использования DSCP и даны рекомендации по применению разных кодов DSCP в одном кортеже (five-tuple) с адресами и номерами портов отправителя и получателя, а также используемым транспортном протоколом (в данном случае UDP или UDP-Lite). В частности, рассмотрено влияние транспортного протокола с контролем перегрузок и гарантиями доставки (повтор передачи, восстановление порядка). Использование нескольких DSCP может приводить к нарушению порядка за счет расширения набора ресурсов пересылки, используемых отправителем. Это может также повысить вероятность истощения ресурсов или отказов.

3.1.9. QoS, заранее предоставленная или зарезервированная полоса

IETF обычно определяет протоколы для использования в модели Best Effort General Internet. Иногда уместно задавать протоколы с иной применимостью. Приложение UDP может использовать схему QoS с интеграцией услуг, которая обычно доступна в контролируемых средах (например, внутри административного домена или между согласованными доменами). Предназначенным для Internet приложениям **не следует** предполагать поддержку механизмов QoS в используемых сетях и поэтому нужно обеспечивать контроль перегрузок, восстановление при ошибках и т. п. В случаях, когда путь через сеть не обеспечивает нужного сервиса.

Для некоторых приложений UDP предполагается использование лишь на путях, обеспечивающих заранее выделенную или резервируемую динамически (например, с помощью протокола RSVP[2]) пропускную способность. Групповые приложения тоже используют выделенную заранее емкость (например, развертывание IPTV в сети доступа). Эти приложения **могут** отказаться от реализации механизмов контроля перегрузок, передавая трафик лишь по путям, где обеспечивается для них достаточная пропускная способность. Это может подходить для ограниченного множества сетевых сред, но не обеспечивает безопасного применения в Internet. Приложениям, выбравшим такой подход, **следует** подробно и аккуратно описывать процедуры обеспечения и управления, которые обеспечивают нужное поведение.

Приложениям с неконтролируемой и неадаптивной передачей **не следует** делать этого по умолчанию и **следует** требовать от пользователя явного включения такого режима работы.

[1]Congestion Experience - наблюдается насыщение.
[2]Resource Reservation Protocol - протокол резервирования ресурсов.

Приложения, разработанные для контролируемых сред (см. параграф 3.6) могут быть способны использовать функции управления сетью для детектирования вызванной ими перегрузки и подобающей реакции на это. Если трафик таких приложений выходит на пути Internet без обеспечения пропускной способности, это может существенно снизить производительность других приложений, использующих тот же путь, и даже вызывать коллапс насыщения. Протоколам, разработанным для таких сетей, следует поддерживать на периметре сети механизмы, предотвращающие выход трафика на пути Internet без обеспечения (см., например, [RFC7510]). Для защиты других приложений, использующих тот же путь, **следует** также реализовать подходящий «выключатель», как описано в параграфе 3.1.10.

Предполагается, что спецификации IETF, предназначенные для контролируемых сред, будут включать заявление о применимости, ограничивающее сферу применения (см. параграф 3.6).

3.1.10. Механизмы выключателей

Транспортный «выключатель» (circuit breaker) — это автоматический механизм, применяемый для оценки вызываемой потоком перегрузки и прерывания (или значительного снижения скорости) потока при обнаружении чрезмерного насыщения [RFC8084]. Это служит мерой защиты от коллапса насыщения (истощение ресурсов, доступных другим потокам) особенно для неоднородной среды Internet с трудно предсказуемым трафиком.

Выключатели служат защитным механизмом, применяемым в крайних случаях. При нормальных условиях использовать их не следует, поскольку они предназначены для защиты от серьезных перегрузок. Цель обычно заключается в ограничении максимальной скорости передачи в соответствии с доступной пропускной способностью сетевого пути. Выключатели могут работать на отдельных потоках UDP или агрегатах трафика, например, для туннеля.

В [RFC8084] приведены рекомендации и примеры использования выключателей, а в [RFC8083] описаны выключатели для RTP.

Приложениям, используемым в Internet, **следует** реализовать транспортные выключатели, если в них нет механизмов контроля перегрузок или объем передаваемых данных достаточно велик (см. параграф 3.6). Транспортные выключатели **могут** реализовать все приложения [RFC8084] и рекомендуется рассматривать реализацию по крайней мере медленно действующего выключателя для защиты в крайних ситуациях.

3.1.11. Туннели UDP

Растет популярность использования UDP как туннельного протокола [INT-TUNNELS], где конечные точки туннеля инкапсулируют пакеты других протоколов в дейтаграммы UDP и передают в другую конечную точку туннеля, которая декапсулирует дейтаграммы UDP и пересылает содержащиеся в них исходные пакеты. Одним из примеров такого протокола является Teredo [RFC4380]. Туннели создают виртуальные каналы, которые представляются прямым соединением между физически разнесенными точка топологии Internet, и могут служит для создания виртуальных (частных) сетей. Применение UDP как туннельного протокола привлекательно в случаях, когда исходный протокол не поддерживается промежуточными устройствами на пути через сеть, поскольку многие промежуточные устройства обеспечивают передачу UDP.

Хорошо реализованные туннели обычно не видны конечным точкам, которые передают трафик по путям с туннельными каналами. С другой стороны, для маршрутизаторов на пути туннеля UDP (т. е. между конечными точками туннеля) трафик туннеля является обычным потоком UDP, а точки инкапсуляции и декапсуляции выглядят как обычные приложения UDP. Поэтому другие потоки могут разделять путь с одним или несколькими туннелями UDP и нужно учитывать контроль перегрузок.

Необходимость развертывания для туннеля UDP определенного механизма контроля перегрузок определяют два фактора - применение для передачи туннельного трафика протокола IP и соответствие объема трафика UDP, создаваемого схемой туннелирования общему объему трафика в туннеле.

Для индивидуального трафика IP обычно предполагается контроль перегрузки, т. е. считается, что транспортные протоколы, создающие индивидуальный трафик IP на стороне отправителя, уже поддерживают механизмы, которых достаточно для предотвращения перегрузки в пути. В результате туннели, передающие индивидуальный трафик IP, уже должны подобающим способом взаимодействовать с другими потоками того же пути и специальный механизм контроля перегрузок для туннеля не требуется.

Однако, если известно об отсутствии контроля перегрузок для туннельного трафика IP, **рекомендуется** ограничивать влияние этого трафика на другие потоки, использующие тот же путь. Рассмотрение туннелей с групповым трафиком IP приведено в параграфе 4.1.

Ниже более подробно описаны возможные варианты.

1. Туннель создает трафик UDP, объем которого соответствует объему полезного трафика (payload) IP, для которого обеспечивается контроль перегрузки.

   Пожалуй, это самый распространенный вариант туннелей в Internet и в этом случае туннелю UDP **не следует** применять свой механизм контроля перегрузок, поскольку потери туннельного трафика в результате насыщения будут вызвать соответствующую реакцию исходных отправителей туннелируемого трафика. Механизм выключателей может обеспечить дополнительное преимущество за счет контроля агрегатов трафика.

   Отметим, что эта рекомендация основана на допущении о наличии контроля перегрузок для большинства коммуникаций IP. Если туннель UDP служит для трафика IP без контроля перегрузок, следует применять приведенные ниже рекомендации.

2. Туннель создает трафик UDP, объем которого соответствует объему полезного трафика (payload), для которого не обеспечивается контроль перегрузки.

   Это может возникать, например, при инкапсуляции в UDP некоторых протоколов канального уровня (не все такие протоколы имеют контроль перегрузок). Поскольку неизвестна реакция на потери пакетов не относящегося к IP туннельного трафика, туннелю UDP **следует** реализовать подходящий механизм контроля

перегрузок или механизм автоматического выключения для туннельного трафика. Поскольку туннели обычно служат для передачи больших объемов трафика и включают промежуточные маршрутизаторы, применимы рекомендации параграфа 3.1.2.

3. Туннель создает трафик UDP, объем которого не соответствует объему полезного трафика (payload).

   Примерами могут служить туннели UDP, передающие данные с постоянной скоростью, увеличивая ее в результате потери при упреждающей корректировке ошибок (Forward Error Correction) или используя иные варианты контроля передачи. Такое специализированное применение UDP для туннелирования выходит за рамки общих рекомендаций этого документа. При создании таких специализированных туннелей **следует** внимательно относится к контролю перегрузок, а также **следует** рассмотреть применение механизмов выключателей.

Тип инкапсулированных данных может быть указан портом UDP, EtherType или номером протокола IP. Туннелям **следует** поддерживать механизмы ограничения типов потоков, которые могут передаваться через туннель. Например, туннелю UDP, предназначенному для трафика IP, следует отфильтровывать на входе пакеты иных протоколов. Это особенно важно при использовании базовой инкапсуляции (например, с использованием EtherType). Для таких туннелей **следует** обеспечивать механизм ограничения типов трафика, которые разрешено инкапсулировать в данный туннель (см. [INT-TUNNELS]).

Разработка механизмов туннелирования требует значительно больше опыта, нежели для других приложений UDP, поскольку туннели обычно должны быть прозрачны для передающих через них данные конечных точек, поэтому туннель должен корректно эмулировать поведение канала IP [INT-TUNNELS]. Например:

- требования по переносу или инкапсуляции кодов ECN [RFC6040];
- использование поля IP DSCP конечными точками туннеля [RFC2983];
- учет инкапсуляции при проектировании туннелей [ENCAP];
- использование сообщений ICMP [INT-TUNNELS];
- обработка фрагментации и размера пакетов для туннелей [INT-TUNNELS];
- использование порта отправителя в туннелях, рассчитанных на маршрутизацию ECMP[1] (параграф 5.1.1);
- потребность в рекомендациях по защите заголовков [INT-TUNNELS] и использованию контрольных сумм для туннелей IPv6 (параграф 3.4.1);
- поддержка эксплуатации и обслуживания [INT-TUNNELS].

В то же время туннельный трафик является обычным трафиком приложения, похожим на любой другой с точки зрения сетей, через которые проходит туннель. В этом документе рассматривается лишь контроль перегрузок при реализации туннелей UDP, а другие аспекты поведения туннелей выходят за рамки документа.

3.2. Рекомендации по размеру сообщений

Фрагментация IP снижает эффективность и надежность коммуникаций в Internet. Потеря одного фрагмента ведет к утрате всего фрагментированного пакета, поскольку даже при получении всех остальных фрагментов собрать и доставить исходный пакет невозможно. Эта проблема фрагментации присуща как IPv4, так и IPv6.

Кроме того, некоторые трансляторы адресов (network address translator или NAT) и межсетевые экраны отбрасывают фрагменты IP. Трансляция сетевых адресов в NAT возможна лишь для полных пакетов IP, а правила некоторых межсетевых экранов требуют проверки пакетов IP целиком. Некоторые устройства NAT и межсетевые экраны просто не включают требуемых для сборки фрагментов средств и поэтому отбрасывают фрагменты. В [RFC4963] рассмотрены также другие проблемы, связанные с фрагментацией IPv4.

В силу отмеченных выше причин приложению **не следует** передавать дейтаграммы UDP, которые приводят в пакетам IP размером больше MTU[2] на пути к получателю. Поэтому приложению **следует** использовать данные о MTU на пути от уровня IP или самостоятельно реализовать определение MTU на пути (Path MTU Discovery или PMTUD) [RFC1191] [RFC1981] [RFC4821] для определения возможности передачи сообщений без фрагментации.

Однако сообщения ICMP, служащие для определения MTU на пути, активно фильтруются промежуточными устройствами (включая межсетевые экраны) [RFC4890]. Когда путь включает туннели, некоторые устройства на входе туннеля отбрасывают сообщения ICMP от сетевых устройств, через которые проходит туннель, препятствуя их доставке в конечную точку UDP.

Механизм PLPMTUD[3] [RFC4821] не опирается на поддержку в сети сообщений ICMP и более устойчив к отказам, нежели PMTUD. Он также не чувствителен к «черным дырам» ICMP. Для работы PLPMTUD требуется изменение способа использования транспорта как для передачи пробных пакетов, так и для учета потери или успешного прохождения проб. Это не только меняет алгоритм PMTU, но и влияет на восстановление потерь, контроль перегрузки и т. п. Эти обновленные механизмы могут быть реализованы в ориентированном на соединения транспорте (например, TCP, SCTP, DCCP), не не являются частью UDP. Данный тип обратной связи обычно отсутствует в однонаправленных приложениях.

Поэтому PLPMTUD вносит дополнительные требования к приложениям UDP, желающим применять этот метод. Особенно важно это для туннелей UDP, поскольку связанные с передачей пробных пакетов издержки должны учитываться и могут потребовать в туннеле дополнительного механизма контроля перегрузок (параграф 3.1.11), а также усложнения пути данных в декапсуляторе туннеля.

Приложениям, не мспользующим PMTU или PLPMTUD, **следует** избегать отправки дейтаграмм UDP, которые будут приводить к передаче пакетов IP, размер которых превысит MTU. Поскольку реальное значение MTU для пути не

[1]Equal cost multipath - множество равноценных путей.
[2]Maximum Transmission Unit - максимальный передаваемый блок.
[3]Packetization Layer Path MTU Discovery - определение Path MTU на уровне пакетизации.

известно, таким приложениям **следует** передавать сообщения, которые короче принятого по умолчанию эффективного значения MTU для передачи (EMTU_S в [RFC1122]). Для IPv4 EMTU_S будет меньшим из значение 576 байтов и MTU первого этапа пересылки [RFC1122], для IPv6 EMTU_S будет 1280 байтов [RFC2460]. Эффективное значение PMTU для подключенного напрямую адресата (без маршрутизаторов на пути) будет определяться заданным для интерфейса значением MTU, которое может быть меньше разрешенного на канале максимума. Передача дейтаграмм UDP минимального размера не эффективна для путей, поддерживающих большие PMTU, и это служит другой причиной реализации определения PMTU.

Для определения подходящего размера данных UDP приложения **должны** вычесть размер заголовка IP (вместе с необязательными заголовками IPv4 или заголовками расширения IPv6), а также размер заголовка UDP (8 батов) из значения PMTU. Этот размер, называемый максимальным размером сегмента (Maximum Segment Size или MSS), можно получить от стека TCP/IP [RFC1122].

Приложения, не передающие сообщений размером больше эффективного PMTU для IPv4 или IPv6, могут не реализовать описанные выше механизмы. Отметим, что наличие туннелей может дополнительно снизить эффективное значение PMTU [INT-TUNNELS], поэтому реализация определения PMTU является предпочтительным решением.

Приложениям, фрагментирующим сообщения прикладного уровня на несколько дейтаграмм UDP, **следует** выполнять эту фрагментацию так, чтобы каждую дейтаграмму можно было доставлять независимо и независимо повторять в случае поддержки приложением своего механизма гарантированной доставки.

3.3. Рекомендации по надежной доставке

Разработчики приложений обычно осознают, что UDP не обеспечивает надежной доставки, например, при потере пакета повторной передачи не происходит. Зачастую это является основной причиной выбора транспорта UDP. Приложения, требующие гарантированной доставки, **должны** сами реализовать подходящий механизм.

UDP также не обеспечивает защиты от дубликатов, т. е. приложение может получить множество копий одной дейтаграммы UDP, при этом некоторые дубликаты могут задерживаться в сети надолго. Разработчикам приложений **следует** аккуратно обрабатывать такие дубликаты дейтаграмм и может потребоваться механизм для их обнаружения. Даже если прием дейтаграммы UDP вызывает лишь идемпотентные операции, приложение может подавлять дубликаты для снижения нагрузки.

Приложения, которым нужна упорядоченная доставка, **должны** самостоятельно упорядочивать дейтаграммы. Некоторые пакеты могут дольше задерживаться в Internet по сравнению с другими, например, в результате изменения маршрутизации, связности или перемещения конечной точки. Это может приводить к нарушению порядка доставки дейтаграмм UDP их получателю.

Приложениям, использующим несколько транспортных портов, нужна устойчивость к нарушению порядка пакетов между сессиями (портами). Методы балансировки нагрузки в сети, такие как маршрутизация по равноценным путям (ECMP), также могут приводить к нарушению порядка в разных транспортных сессиях даже для одной пары конечных точек.

Важно отметить, что интервал, в котором меняется порядок пакетов или могут поступать дубликаты, может быть очень большим. Еще важнее отсутствие четкой верхней границы для таких интервалов. В [RFC793] задана максимальная задержка, возможная для сегмента TCP - максимальный срок жизни сегмента (Maximum Segment Lifetime или MSL) - 2 минуты. RFC, определяющих MSL для других транспортных протоколов или IP, не существует. Значение MSL для TCP достаточно консервативно, чтобы его **следовало** применять в других протоколах, включая UDP. Поэтому приложениям **следует** быть устойчивыми к приему задержанных пакетов и дубликатов в течение двухминутного интервала.

Повтор потерянных пакетов или сообщений является общепринятым механизмом гарантированной доставки. Такие повторы могут повышать нагрузку в сети в ответ на насыщение, что дополнительно увеличивает перегрузку. Любое приложение, использующее повтор передачи, отвечает за контроль перегрузки при повторах (а также для трафика приложения), поэтому на него распространяются рекомендации по контролю перегрузок из параграфа 3.1. Рекомендации по измерению RTT в параграфе 3.1.1 применимы и к таймерам, используемым для повтора передачи при обнаружении потерь.

Вместо реализации сравнительно сложных механизмов гарантированной доставки приложению, которому нужны гарантии, **следует** рассмотреть транспортные протоколы IETF, обеспечивающие гарантии доставки.

3.4. Рекомендации по контрольным суммам

Заголовок UDP включает необязательное 16-битовое поле контрольной суммы с дополнением до 1, которое служит для контроля целостности. Эти проверки не являются строгими с точки зрения кодирования или криптографии и не предназначены для обнаружения ошибок на физическом уровне или умышленного изменения дейтаграмм [RFC3819]. Разработчиками приложений, где важна целостность, **следует** реализовать дополнительные проверки, например с использованием CRC[1] или криптографического хэш-кода (с ключом или без него), включаемого в данные для контроля целостности всего объекта или файла, передаваемого с помощью транспорта UDP.

Контрольная сумма UDP предоставляет статистическую гарантию отсутствия повреждений данных в процессе передачи. Это также позволяет получателю убедиться, что пакет предназначен именно ему, поскольку в контрольной сумме учитывается адрес IP, номер порта и протокола, а также подтверждается отсутствие дополнений или отсечки данных, поскольку размер учитывается в контрольной сумме. Поэтому контрольная сумма защищает приложение от получения поврежденных или дополненных в процессе передачи данных. Более подробное описание проверок на основании поля контрольной суммы приведено в параграфе 3.1 [RFC6396].

Приложениям **следует** включать контрольные суммы UDP [RFC1122]. Для IPv4 [RFC768] имеется опция запрета контрольной суммы с установкой в поле заголовка значения 0. Приложениям разрешено отвергать дейтаграммы UDP с нулевым значением контрольной суммы [RFC1122].

[1]Cyclic Redundancy Check - циклическая контрольная сумма с избыточностью.

При использовании UDP с протоколом IPv6 контрольная сумма UDP служит для защиты заголовков IPv6 и UDP (в заголовке IPv6 нет контрольной суммы) и **должна** использоваться в соответствии с [RFC2460]. В определенных условиях приложениям UDP разрешается использовать режим с нулевой контрольной суммой (UDP zero-checksum) в протоколах туннелирования (см. параграф 3.4.1).

Приложениям, отключающим контрольную сумму UDP, **недопустимо** делать предположения о корректности принятых данных и они **должны** обеспечивать корректное поведение при получении чужих или поврежденных дейтаграмм UDP.

3.4.1. Нулевая контрольная сумма UDP для IPv6

В [RFC6935] задан метод, позволяющий использовать нулевую контрольную сумму UDP с туннельными протоколами при условии соблюдения требований [RFC6936]. Приложение **должно** реализовать механизмы и/или ограничения на использование при включении этого режима. Это включает задание сферы действия и мер по предотвращению утечки трафика в другие приложения UDP (см. Приложение A и параграф 3.6). Эти дополнительные требования к использованию нулевых контрольных сумм IPv6 UDP отсутствуют в IPv4, поскольку заголовок IPv4 проверяет информацию, не защищенную в пакетах IPv6. Основные требования перечислены ниже.

- Использование контрольной суммы UDP с IPv6 **должно** быть включено по умолчанию во всех реализациях [RFC6935]. Принимающая сторона **должна** разрешать использование режима UDP zero-checksum для IPv6 лишь на портах UDP, где это специально разрешено.
- Приложение, поддерживающее контрольную сумму, отличную от [RFC2460], **должно** выполнять все требования к реализации из раздела 4 в [RFC6936] и требования к использованию из раздела 5 в [RFC6936].
- Приложение UDP **должно** проверять действительность адресов IPv6 для отправителя и получателя по всех пакетах с нулевой контрольной суммой UDP и **должно** отбрасывать пакеты при несоответствии адресов. Для защиты от ошибочной доставки в новые схемы инкапсуляции **следует** включать контроль целостности на транспортном уровне с проверкой по меньшей мере заголовков IPv6 и UDP, а также промежуточного заголовка инкапсуляции при его наличии [RFC6936].
- Одним из способов, помогающих выполнить требования [RFC6936], может быть ограничение использования таких туннелей, например, путем ограничения трафика в сети оператора, как описано в параграфе 3.6. Инкапсуляция, заданная для MPLS в UDP [RFC7510], использует такой подход.

Как и в IPv4, приложениям IPv6, выключившим контрольную сумму UDP, **недопустимо** предполагать корректность принятых данных и они **должны** обеспечивать корректное поведение при получении чужих или поврежденных дейтаграмм.

Дейтаграммы IPv6 с нулевой контрольной суммой UDP не будут пропускаться промежуточными устройствами, проверяющими контрольную сумму в соответствии с [RFC2460] или обновляющими поле контрольной суммы UDP (например, трансляторами NAT и межсетевыми экранами). Для смены такого поведения потребуется обновление промежуточных устройств с целью корректной обработки дейтаграмм с нулевой контрольной суммой UDP. Для обеспечения сквозной отказоустойчивости приложения, которые могут работать через Internet, **должны** обеспечивать возможность работы с контрольными суммами при изменении пути с перенаправлением на участки, где промежуточные устройства отбрасывают дейтаграммы IPv6 с нулевой контрольной суммой UDP.

3.4.2. UDP-Lite

Особый класс приложений может получать преимущество в результате доставки частично поврежденных данных вместо их отбрасывания при работе по каналам с большим числом ошибок. Такие приложения устойчивы к повреждению данных и **могут** выбрать облегченный протокол UDP-Lite (Lightweight User Datagram Protocol) [RFC3828] вместо базового UDP. Таким приложениям следует выполнять рекомендации по контролю перегрузок и другие рекомендации раздела 3.

UDP-Lite меняет поле размера данных UDP (payload length) на область покрытия контрольной суммой (checksum coverage length), а в остальном протоколы семантически идентичны. Интерфейс UDP-Lite отличается от интерфейса UDP добавлением одной опции, указывающей область покрытия контрольной суммы. На стороне отправителя это задает учитываемые в контрольной сумме данные (остальная часть считается нечувствительной к ошибкам - error-insensitive part). По умолчанию контрольная сумма UDP-Lite учитывает всю дейтаграмму, но приложение может динамически менять размер учитываемой области, например, для повышенной защиты некоторых сообщений. UDP-Lite всегда проверяет доставку нужному получателю, т. е. всегда проверяется корректность полей заголовка. Ошибки в нечувствительной части не будут вызывать отбрасывания дейтаграммы получателем. Поэтому приложениям UDP-Lite **недопустимо** предполагать корректность данных в нечувствительной к ошибкам части дейтаграмм UDP-Lite.

Отправителю UDP-Lite **следует** выбирать минимальное покрытие контрольной суммой, охватывающее все чувствительные к ошибкам области данных. Например, приложения, использующие протокол RTP [RFC3550], очевидно захотят защитить от повреждений заголовок RTP. По-возможности приложения **должны** также вводить свои подходящие проверки для протокольной информации, передаваемой в незащищенной части данных UDP-Lite (например, встраивать CRC).

Получатель UDP-Lite **должен** задать порог минимального покрытия для входящих пакетов, который должен быть не меньше минимального покрытия, используемого отправителем [RFC3828]. Получателю **следует** выбирать достаточно высокий порог для блокирования пакетов со слишком малым покрытием контрольной суммы. Это может быть фиксированное или согласуемое приложением значение. UDP-Lite не поддерживает механизмов согласования области покрытия для контрольной суммы между отправителем и получателем, поэтому приложение должно делать это самостоятельно.

Приложения, использующие UDP-Lite, также могут сталкиваться с потерей пакетов. Улучшения, предлагаемые UDP-Lite, основаны на том, что канал может «перехватывать» заголовок UDP-Lite для корректной идентификации требуемого частичного покрытия. При использовании туннелей и/или шифрования это может приводить к тому, что дейтаграммы UDP-Lite будут обрабатываться так же, как дейтаграммы UDP, т. е. пакеты будут теряться. Фрагментация IP также может препятствовать особой обработке дейтаграмм UDP-Lite и это является еще одним основанием для предотвращения фрагментации IP (параграф 3.2).

UDP-Lite поддерживается в протокольных стеках некоторых конечных точек. Текущая поддержка в промежуточных устройствах достаточно слаба, поскольку UDP-Lite использует свой номер протокола в IPv4 и значение IPv6 next header (отличные от UDP). В результате лишь немногие промежуточные устройства способны интерпретировать UDP-Lite и выполнять требуемые действия при пересылке пакетов. Это делает протокол UDP-Lite менее подходящим для работы через Internet, пока не будет обеспечена достаточная поддержка UDP-Lite в промежуточных устройствах.

3.5. Рекомендации по работе с промежуточными устройствами

Трансляторы NAT и межсетевые экраны являются примерами промежуточных устройств (middlebox), которые могут присутствовать на сквозном пути. Промежуточные устройства обычно выполняют функции, требуемые для поддержки состояний на уровне потока. Для ориентированных на соединения протоколов (таких как TCP) , промежуточные устройства отслеживают и анализируют данные управления соединениями, а также создают и уничтожают должным образом состояния на уровне потока. Для протоколов без явных соединений, таких как UDP, такой подход невозможен. Поэтому промежуточные устройства могут создавать состояния на уровне потока, когда они видят пакет, соответствующий (по локальным критериям) новому потоку, и удаляют состояние по истечении некоторого времени в течение которого не наблюдается пакетов, относящихся к тому же потоку.

В зависимости от конкретных функций, выполняемых промежуточным устройством, оно может вносить временные зависимости, ограничивающие типы обмена трафиком UDP через это устройство. Например, трансляторы NAT и межсетевые экраны обычно задают часть пути на одной стороне как внутреннюю для обслуживаемого домена, а другую - как внешнюю. Состояние на уровне потока обычно создается при прохождении первого пакета изнутри наружу и при наличии такого состояния трансляторы NAT и межсетевые экраны будут пересылать трафик. Обратный трафик, поступающий после завершения срока действия потока, будет отбрасываться, как и другой трафик извне.

Многие приложения, использующие UDP, работают через промежуточные устройства без использования дополнительных механизмов. Одним из примеров является система доменных имен (Domain Name System или DNS), которая использует строгую модель «запрос-отклик», где операция обычно длится несколько секунд.

В других приложениях могут возникать отказы, когда промежуточные устройства будут разрушать состояние, связанное с сессией приложения, если приложение какое-то время не передает трафика UDP. Приложениям **следует** аккуратно обрабатывать такие отказы и реализовать механизмы восстановления состояний и сессий прикладного уровня. Для некоторых приложений, таких как media-потоки, очень нежелательна такая ресинхронизация, поскольку она может вызывать заметные для пользователя проблемы воспроизведения. Такие специализированные приложения **могут** периодически передавать сообщения keep-alive для обновления состояний на промежуточных устройствах (например, [RFC7675]). Важно отметить, что сообщения keep-alive не рекомендуются в общем случае, поскольку они не нужны для большинства приложений и могут потреблять заметный объем системных и сетевых ресурсов.

Приложению, которому нужно реализовать сообщения keep-alive для поддержки нужного сервиса UDP через промежуточные устройства, **не следует** передавать эти сообщения чаще 1 раза за 15 секунд и по-возможности **следует** использовать больший интервал. Для тайм-аута состояний на уровне потока UDP для произвольных промежуточных устройств общего значения не задано. Трансляторам NAT требуется тайм-аут в 2 минуты или больше [RFC4787]. Однако эмпирические данные показывают, что значительная часть развернутых промежуточных устройств использует более короткий тайм-аут. Протокол интерактивной организации соединений ICE (Interactive Connectivity Establishment) [RFC5245] использует значение 15 секунд. При развертывании приложения в контролируемой среде разработчикам **следует** изучить возможности целевой среды в плане использования более длинных интервалов и наличия механизмов явного управления тайм-аутами на промежуточных устройствах, например с помощью протокола управления портами PCP (Port Control Protocol) [RFC6887], MIDCOM (Middlebox Communications) [RFC3303], NSIS (Next Steps in Signaling) [RFC5973] или UpnP (Universal Plug and Play) [UPnP]. Приложениям **рекомендуется** вносить небольшие случайные вариации (jitter) для интервала передачи сообщений keep-alive с целью предотвращения ненужной синхронизации между сообщениями keep-alive от разных хостов [RFC7675].

Передача сообщений keep-alive не подходит для реализации механизма восстановления разорванных сессий. Как и все дейтаграммы UDP, сообщения keep-alive могут быть задержаны или отброшены, что ведет к тайм-аутам промежуточных устройств. Кроме того, рекомендации по контролю перегрузок в параграфе 3.1 относятся ко всем передачам UDP от приложений, включая передачу сообщений keep-alive промежуточными устройствами. Контроль перегрузок в результате может приводить к задержками или временной приостановке передачи keep-alive.

Сообщения keep-alive **не рекомендуются** для общего применения, они не требуются для многих приложений и могут поглощать много ресурсов. Например, когда устройству с батарейным питанием нужно в течение длительного времени поддерживать соединение с незначительным трафиком, частота отправки сообщений keep-alive может стать определяющим фактором энергопотребления в зависимости от базовой сетевой технологии.

Поскольку многие промежуточные устройства требуют передачи сообщений keep-alive для соединений TCP с частотой много ниже, нежели нужно для UDP, это часто служит основанием для отказа от UDP в пользу TCP. С другой стороны, имеются сведения о том, что прямые коммуникации через промежуточные устройства, например, с применением ICE [RFC5245], реже бывают успешными для TCP, нежели для UDP. Компромиссы при выборе транспортного протокола, особенно при наличии промежуточных устройств, требуют тщательного анализа.

Приложения UDP, которые могут быть развернуты в Internet, должны проектироваться с учетом многочисленных вариантов поведения промежуточных устройств и, хотя UDP не организует прямых соединений, промежуточные устройства часто поддерживают состояние для каждого потока UDP. При наличии множества потоков UDP для таких состояний могут потребоваться значительные ресурсы и этом может менять поведение промежуточного устройства при обработке последующих пакетов (для защиты внутренних ресурсов или предотвращения злоупотреблений). Вероятность отказов в пути может возрастать при использовании приложением множества параллельных потоков UDP (см. рекомендации параграфа 5.1.2 для работы через множество портов).

3.6. Ограничение применимости и контролируемые среды

Для спецификаций приложений IETF, использующих протокол UDP, выделено 2 варианта работы, описанных ниже.

Сеть Internet. По умолчанию в спецификациях IETF целевой средой развертывания считается сеть Internet в целом. Опыт показывает, что успешные протоколы, разработанные в одном конкретном контексте или для

определенного применения, часто начинают применяться в более широком контексте. Например, протокол, изначально развернутый в локальной сети, может затем применяться в виртуальной сети, организованной через Internet, или просто в общей сети Internet. Приложения, предназначенные для использования в Internet, могут по разному вести себя на сетевых устройствах и, в частности, при работе по путям, включающим промежуточные устройства.

Контролируемая среда. Протокол, инкапсуляция или туннель может быть разработан для применения только внутри контролируемой среды. Например, это может быть приложение, созданное для использования в сети оператора, которое развертывается в сети одного оператора или нескольких взаимодействующих смежных операторов. Трафик таких приложений может быть управляемым для предотвращения перегрузок без использования встроенных механизмов, которые нужны при работе через Internet. Приложения, рассчитанные на ограниченное применение, могут использовать преимущества определенного оборудования (например, операторского) или свойства базовых протоколов в подсети, где приложение применяется.

Спецификациям, предназначенным для ограниченного применения или контролируемых сред, **следует** как в ограниченном развертывании обеспечивается защита, эквивалентная защите протоколов, предназначенных для работы в Internet (например, реализация на основе обширного опыта развертывания конкретных методов, обеспечивающих свойства, которых нельзя ожидать от обычного оборудования Internet, и устойчивости MPLS к повреждению заголовков, помогающие использовать дополнительную проверку целостности UDP [RFC7510]).

В спецификациях IETF, нацеленных на контролируемую среду, предполагаются наличие заявления о применимости, которое ограничивает трафик приложения контролируемой средой, и предполагается описание методов предотвращения выхода поврежденных пакетов за пределы контролируемой среды (см., например, раздел 5 в [RFC7510]).

4. Рекомендации по групповой адресации в UDP

Этот раздел дополняет предыдущий рекомендациями, связанными с групповой и широковещательной адресацией в UDP.

Групповая и широковещательная передача [RFC1112] обычно использует транспорт UDP, хотя возможно применение и других транспортных протоколов, например, UDP-Lite.

В настоящее время имеются две модели групповой доставки - Any-Source Multicast (ASM) [RFC1112] и Source-Specific Multicast (SSM) [RFC4607]. Члены групп ASM будут получать все данные, переданные в группу любым отправителем, тогда как SSM ограничивает дерево распространения единственным источником.

Специальные классы приложений также применяют UDP для групповой и широковещательной рассылки IP [RFC919]. Устройство таких приложений требует опыта, выходящего за пределы простых рекомендаций для индивидуальной передачи, поскольку такие отправители могут передавать пакеты очень большому числу адресатов по весьма разнородным путям, что значительно усложняет механизмы контроля перегрузок, управления потоками и гарантированной доставки.

В этом разделе даны рекомендации по использованию UDP с групповой и широковещательной адресацией. Использование широковещания приложением обычно ограничивается маршрутизаторами на границах локальной сети. Однако методы туннелирования и прокси в некоторых случаях позволяют передавать широковещательный трафик через Internet. Поэтому приведенные здесь рекомендации относятся и к широковещательному трафику.

В IETF определена схема гарантированной групповой доставки [RFC3048] и несколько «строительных блоков» в помощью разработчикам групповых приложений, например, [RFC3738] и [RFC4654].

Отправители по адресам anycast должны осознавать, что последовательные сообщения отправленные по тому же anycast-адресу IP могут доставляться разным anycast-узлам, т. е. приходить в разные точки топологии.

Большинство туннелей UDP, передающих групповой трафик IP, используют туннельную инкапсуляцию с индивидуальным адресом получателя, например, Automatic Multicast Tunneling [RFC7450]. Они **должны** выполнять те же требования, которые предъявляются к туннелям с индивидуальным трафиком (см. параграф 3.1.11). Имеются варианты развертывания и решения, где внешний заголовок туннеля UDP содержит групповой адрес получателя [RFC6513]. Такие решения обычно развертываются в контролируемых средах внутри одного административного домена или между доменами с согласованной пропускной способностью путей, поэтому контроль перегрузок является **необязательным**, но **рекомендуется** применять выключатели (circuit breaker) для обеспечения того или иного уровня обслуживания в случае выхода за пределы зарезервированной пропускной способности (например, в результате ошибки при настройке).

4.1. Рекомендации по контролю перегрузок

Механизмы контроля перегрузок для индивидуального трафика зачастую не подходят для групповых услуг или просто не могут работать с большими multicast-деревьями, поскольку им нужна двухсторонняя связь и скорость приспосабливается к условиям пути через сеть к одному получателю. Деревья группового распространения могут включать большое число адресатов, что ограничивает возможности каналов возврата в основной полосе для контроля скорости передачи. А отношения «один со многими» в деревьях группового распространения препятствуют адаптации скорсоти к требованиям отдельных получателей. По это причине генерация совместимых с TCP агрегированных потоков данных для групповой передачи через Internet в естественной форме или через туннель, лежит в зоне ответственности приложения, реализующего контроль перегрузок.

Групповым приложениям **следует** поддерживать подобающий контроль перегрузок, который должен быть реализован с использованием механизмов, устойчивых к возможным неоднородностям дерева группового распространения и входящих в группу получателей. Неоднородность может проявляться в разном уровне потерь, разных задержках и/или возможностях реагировать на возникновение перегрузки в сети. Контроль перегрузок особенно важен для любой групповой сессии, где часть дерева распространения проходит через сеть доступа (например, домовый шлюз). В RFC определено для стиля контроля перегрузок, указанных ниже.

- Контроль перегрузки на основе обратной связи, где отправитель получает от адресатов групповые или индивидуальные сообщения UDP, позволяющие оценить уровень перегрузки и скорректировать скорость отправки (например, [RFC5740],[RFC4654]). Групповые методы могут работать в более продолжительном временном масштабе, нежели индивидуальные (например, по причине большего значения группового RTT в неоднородной группе). Метод управления может отказаться от снижения скорости для всей multicast-группы в ответ на управляющее сообщение от одного получателя (например, отправитель может установить минимальную скорость и предложить связанному с перегрузкой получателю покинуть группу или выбрать для связанных с перегрузкой получателей передачу с меньшей скоростью и применением обычного (unicast) контроля перегрузки).

- Управляемый получателем контроль перегрузок, который не требует передавать явных управляющих сообщений UDP (например, [RFC3738], [RFC5775]), а вместо этого отправитель распространяет данные через множество групп IP (например, используя каналы {S,G}). Каждый получатель определяет свой уровень перегрузки и контролирует скорость приема, используя лишь сообщения по включении и выходе из группы (join/leave) отправляемые на уровне управления сетью. Этот метод расширяется для больших групп.

Любой получатель с поддержкой групп может присоединиться к любой группе и получать ее трафик. Это может предполагать ограничение скорости для отдельных получателей или агрегата групповых услуг. Отметим, что на транспортном уровне нет способа предотвратить распространение сообщений о включении в группу (join) маршрутизатору следующего этапа пересылки.

Некоторые классы групповых приложений поддерживают отслеживание качества передачи на пользовательском уровне у получателей. Приложениям, которые могут обнаруживать существенное снижение качества, **следует** воспринимать это как сигнал перегрузки (например, для выхода из группы с использованием многоуровневого группового кодирования). В противном случае им **следует** использовать этот сигнал для выключателей, позволяющих прервать поток путем выхода из группы.

4.1.1. Групповые приложения с большим объемом данных

Приложениям с большим объемом передачи данных (больше нескольких дейтаграмм UDP в интервале RTT) через групповое дерево распространения **следует** реализовать метод контроля перегрузок. В настоящее время **рекомендуются** методы IETF ALC[1] [RFC5775], TFMCC[2] [RFC4654], WEBRC[3] [RFC3738], транспортный протокол NORM[4] [RFC5740], FLUTE[5] [RFC6726], RTP/RTCP[6] [RFC3550].

Приложение может дополнительно реализовать другую схему контроля перегрузок, следуя рекомендациям [RFC2887] и используя схему [RFC3048]. Приложениям с большим объемом передачи, отказавшимся от реализации [RFC4654], [RFC5775], [RFC3738], [RFC5740], [RFC6726], [RFC3550], **следует** реализовать схему контроля перегрузок, которая обеспечит беспристрастность использования пропускной способности близкую к TCP по порядку значений.

В разделе 2 [RFC3551] сказано, что приложениям multimedia **следует** контролировать потерю пакетов для обеспечения приемлемых параметров. Потеря считается приемлемой, если поток TCP по тому же пути и с теми же условиями обеспечивает среднюю пропускную способность в разумном масштабе времени не ниже пропускной способности для потока UDP. Сравнение с TCP нельзя выполнить точно, но предполагается одинаковый порядок значений для временного интервала и пропускной способности.

4.1.2. Групповые приложения с небольшим объемом данных

Все рекомендации параграфа 3.1.3 применимы к групповым приложениям с небольшим объемом данных.

4.2. Рекомендации по размеру групповых сообщений

Групповому приложению **не следует** передавать дейтаграммы UDP, приводящие к пакетам IP, размер которых превышает эффективное значение MTU, как указано в разделе 3 [RFC6807]. Поэтому приложению **следует** использовать эффективные данные MTU от PIM[7] [RFC6807] или реализовать самостоятельно механизм определения MTU на пути (см. параграф 3.2) для проверки поддержки нужного размера сообщений на пути к каждому получателю.

5. Рекомендации по программированию

Фактическим стандартом API для приложений TCP/является интерфейс сокетов [POSIX]. Некоторые платформы также предлагают приложениям возможность напрямую собирать и передавать пакеты IP через необрабатываемые сокеты (raw socket) или похожие средства. Это второй и более громоздкий метод использования UDP. Рекомендации этого документа охватывают методы, с помощью которых приложение может использовать UDP. Поскольку API сокетов применяется более широко, далее в этом разделе этот метод рассматривается более подробно.

Хотя API разработаны для UNIX в начале 1980-х годов, позднее появилось множество реализация для других ОС, поддерживающих протоколы IPv4 и IPv6 [RFC3493]. API сокетов UDP отличается от TCP в нескольких важных аспектах. Поскольку программистам приложений обычно лучше известны API сокетов TCP, в этом параграфе рассмотрены некоторые различия. В [STEVENS] приведены примеры использования API сокетов UDP.

Дейтаграммы UDP можно передавать и принимать напрямую без организации явного соединения. Используя API сокетов, приложения могут получать пакеты с разных IP-адресов через один сокет UDP. Некоторые серверы применяют это для одновременного обмена данными с несколькими удаленными хостами через один сокет. Многим приложениям требуется обеспечить прием пакетов с определенного адреса отправителя и такие приложения **должны**

[1]Asynchronous Layered Coding - асинхронное многоуровневое кодирование.
[2]TCP-Friendly Multicast Congestion Control - дружественный к TCP контроль перегрузок для группового трафика.
[3]Wave and Equation Based Rate Control - контроль скорости на основе волн и уравнений.
[4]NACK-Oriented Reliable Multicast - ориентированная на негативные подтверждения групповая передача.
[5]File Delivery over Unidirectional Transport - доставка файлов через односторонний транспорт.
[6]Real Time Protocol/Control Protocol - протоколы работы и управления в реальном масштабе времени.
[7]Population Count Extensions to Protocol Independent Multicast - расширения для учета числа участников в независимых от протокола групповых приложениях.

применять соответствующие проверки на прикладном уровне или явно запрашивать у операционной системы фильтрацию пакетов.

Многие операционные системы позволяют подключить сокет UDP, т. е. связать его с конкретной парой адресов и портов. Это похоже на соответствующую функциональность API сокетов TCP, однако в UDP эта операция локальна и служит лишь для упрощения локальных функций приема и передачи, а также фильтрации трафика для конкретных адресов и портов. Привязка сокета UDP не организует соединения и UDP не уведомляет удаленную сторону о привязке локального сокета UDP. Привязка сокета также помогает в настройке опций, влияющих на уровень UDP или IP, например, использование контрольной суммы UDP или опции IP Timestamp. В некоторых стеках привязка сокета также позволяет приложению получать уведомления при получении сообщения ICMP об ошибках для переданных через сокет пакетов [RFC1122].

Если приложение «клиент-сервер» выполняется на хосте с несколькими интерфесами IP, ему **следует** передавать все отклики UDP с IP-адресом отправителя, соответствующим адресу получателя в запросе ([RFC1122], параграф 4.1.3.5). Многие промежуточные устройства предполагают такое поведение и отбрасывают отклики, переданных с других адресов IP, как разъяснено в параграфе 3.5.

Получатель UDP может принять корректную дейтаграмму UDP без данных (payload). Отметим, что это отличается от возврата значения 0 из функции сокета read(), что в TCP указывает завершение соединения.

UDP не обеспечивает управления потоком данных, т. е. отправитель в каждый момент не знает текущие возможности принимающего приложения обрабатывать входящие пакеты. Это служит другой причиной потребности приложений UDP в отказоустойчивости при наличии потерь пакетов. Потери могут происходить на передающем хосте, когда приложение передает данные со скоростью, превышающей возможности выходного интерфейса, или у получателя, где вызовы функции receive не могут вернуть все данные, поскольку передающая сторона отправляет их слишком часто (переполнение входных буферов). Отказоустойчивая механизмы управления потоком данных сложны в реализации, поэтому приложениям, которым такая функциональность нужна, следует рассмотреть применение протокола TCP.

Когда приложение закрывает сокет TCP, SCTP или DCCP, транспортный протокол принимающего хоста возвращается в состояние TIME-WAIT. Это предотвращает ошибочную привязку задержанных пакетов закрытого соединения с последующим экземпляром соединения, использующего тот же адрес IP и порт. В протоколе UDP такой механизм не реализован, поэтому приложениям UDP требуется устойчивость к изменению порядка и задержке пакетов. Приложение может закрыть сокет или завершить работу, после чего другое приложение может принимать пакеты через тот же порт. В результате это приложение может получить «чужие» пакеты, задержанная в сети.

5.1. Использование портов UDP

Правила и процедуры поддержки реестра Service Name and Transport Protocol Port Number заданы в [RFC6335]. Рекомендации по использованию портов UDP приведены в [RFC7605].

Отправителю UDP **не следует** использовать выходной порт 0. Номер выходного порта, который нельзя определить по адресу или типу данных, обеспечивает защиту приемной стороны от атак со вставкой данных со стороны устройств, не находящихся на пути (off-path). Получателю UDP **не следует** привязываться к порту 0.

Приложениям **следует** проверять порт и адрес получателя на прикладном уровне или явно запрашивать у ОС фильтрацию принимаемых пакетов для предотвращения приема трафика из произвольных портов. Это предназначено для дополнительной защиты от атак со вставкой данных со стороны устройств, не находящихся на пути (они могут не знать номер порта).

Приложениям **следует** включать проверки, обеспечивающие защиту от вставки данных со стороны устройств, не находящихся на пути, которая может исходить от неуполномоченных сторон. Стек TCP обычно использует случайный номер выходного порта для такой защиты [RFC6056] и приложениям UDP следует применять такой же подход. Промежуточные устройства и конечные системы часто используют допущения о номерах системных или пользовательских портов, поэтому рекомендуется применять случайные номера из диапазона Dynamic and/or Private Port. Установка «случайного» выходного порта также повышает уверенность в том, что полученные сообщения ICMP об ошибках исходят от систем на пути потока данных. Некоторые приложения (включая групповые) UDP используют предопределенное значение выходного порта, поэтому им нужен другой метод. Защита от атак off-path может также обеспечиваться с помощью случайных начальных значений полей протокола в области данных (payload) и проверкой их пригодности на приемной стороне (например, в RTP применяется случайный начальный номер и случайные смещения временных меток [RFC3550]).

При использовании групповой адресации маршрутизаторы IP выполняют проверку пересылки по обратному пути (reverse-path forwarding или RPF) для каждого группового пакета. Это обеспечивает защиту от вставки данных off-path, ограничивая возможности подмены адреса отправителя. При подключении получателя к группе и фильтрации по адресам отправителя проверяется IP-адрес источника. Это происходит всегда при использовании канала SSM {S,G}.

5.1.1. Использование UDP для энтропии выходного порта и метки потока IPv6

Некоторые приложения используют заголовки дейтаграмм UDP в качестве источника энтропии для сетевых устройств, реализующих ECMP [RFC6438]. Туннельные приложения UDP, нацеленные на такое использование, инкапсулируют внутренний заголовок с помощью UDP, где выходной порт UDP образует часть энтропии, которая может служить для балансирования пересылки сетевого трафика устройствами ECMP. Передающая сторона туннеля выбирает значение выходного порта в дейтаграммах UDP на основе информации внутреннего потока (например, заголовков инкапсулированных пакетов). Для обеспечения достаточной энтропии передающая сторона туннеля отображает инкапсулируемый трафик на диапазон выходных портов UDP. Значение **следует** выбирать из диапазона «эфемерных» портов 49152 - 65535, где два старших бита ноера имеют значение 1. Доступная энтропия номера порта составляет 14 битов (диапазон «эфемерных» портов) с добавлением внешних адресов IP представляется достаточной для большинства приложений ECMP [ENCAP].

Для предотвращения изменения порядка в потоке IP **следует** использовать один выходной порт UDP для всеха пакетов инкапсулированного потока (например, используя хэш относящихся к потоку заголовков). Отображение энтропии для потока **может** меняться в течение срока действия инкапсулированного потока [ENCAP]. Например, это

может применяться для ослабления DoS-атак[1] или эффективной маршрутизации через сеть ECMP. Однако выходной порт для потока **не следует** менять чаще 1 раза в 30 секунд (например, как в [RFC8086]).

При использовании выходного порта как источника энтропии возникает ряд побочных эффектов.

- Может возрасти вероятность ошибочной доставки поврежденных пакетов, повышающая потребность в использовании контрольной суммы или эквивалентного механизма для защиты других приложений UDP от ошибочной доставки (параграф 3.4).
- Предполагается снижение вероятности успешного прохождения через промежуточные устройства (параграф 3.5). Такое использование выходного порта часто не подходит для приложений, развертываемых в Internet.
- Может препятствовать использованию поля для защиты от атак off-path (параграф 5.1). Разработчикам следует рассмотреть иные механизмы для обеспечения эквивалентной защиты (например, ограничиваться применением в контролируемых средах [RFC7510], см. параграф 3.6).

Поле выходного порта UDP также может служить для создания энтропии в IPv6. Однако в IPv6 для создания энтропии при балансировке [RFC6438] можно использовать также метки потоков [RFC6437]. Применение метки потока для балансировки совместимо с определением этого поля, хотя нужны дополнительные разъяснения в части согласованного использования поля. Поэтому было определено использование обновленных меток потоков IPv6 [RFC6437] и маршрутизации ECMP [RFC6438].

Для обеспечения будущего применения меток потоков приложениям UDP **следует** устанавливать поле Flow Label, даже в тех случаях, когда энтропия уже задана выходным портом (например, конечная точка туннеля IPv6 может копировать значение энтропии выходного порта в поле метки потока IPv6 [RFC8086]). Производители маршрутизаторов призываются к использованию метки потока IPv6 как части хэш-значения для потока с целью обеспечения ECMP на уровне IP без необходимости применения UDP. Сквозное применение меток потоков для балансировки нагрузки является долгосрочным решением. Даже при разъяснении использования меток потоков сохранится переходный период до того, как существенная часть конечных точек начнет назначать эффективные значения меток для создаваемых потоков. Балансировка нагрузки с помощью полей транспортных заголовков очевидно будет продолжаться до широкого внедрения работы с метками потоков.

5.1.2. Приложения, использующие несколько портов UDP

Одно приложение может обмениваться несколькими типами данных и в некоторых случаях это может требовать множества портов UDP (например, несколько наборов потоков, идентифицируемых квинтетами). [RFC6335] рекомендует разработчикам приложений не применять несколько общеизвестных (системных или пользовательских), выделенных IANA. В документе не рассматривается использование нескольких потоков с одним общеизвестным портом или парами динамических портов (например, определяемых именем службы или сигнальным протоколом).

Использование множества потоков может оказывать несколько типов влияния на сеть.

- Начало серии последовательных соединений может увеличить число привязок состояний в промежуточных устройствах (например, NAPT или МСЭ) на пути через сеть. Прохождение UDP через такие устройства обычно основано на тайм-аутах для удаления старых состояний, поскольку промежуточные устройства не знают, когда приложение прекращает использование потока.
- Одновременное использование нескольких потоков может приводить к разным характеристикам для каждого потока. Нельзя предполагать, что все потоки пойдут по одному пути (например, при использовании ECMP трафик преднамеренно распределяется по разным параллельным путям на основе номеров портов).
- Использование нескольких потоков может также повысить заполнение таблиц поиска и привязки в промежуточных устройствах (например, NAPT или МСЭ), что может заставить устройство изменить способ поддержки состояний потоков.
- Кроме того, использование избыточного числа потоков может снизить возможности контроля перегрузок для индивидуального трафика, если состояния контроля перегрузки не являются общими для всех потоков в сессии. Выполняемый получателем контроль перегрузок для группового трафика требует от передающего приложения распространять данные по нескольким multicast-группам IP, и для каждого получателя предполагается получение трафика от небольшого числа активных одновременно портов UDP.

Следовательно, приложениям **недопустимо** предполагать согласованное поведение промежуточных устройств при использовании множества потоков UDP. Многие устройства могут по-разному реагировать на увеличение числа применяемых портов. Множество потоков с разными требованиями QoS требует от приложения проверять обеспечение ожидаемой производительности для каждого отдельного потока, как указано в параграфе 3.1.9.

5.2. Рекомендации для ICMP

Приложения могут использовать сведения сообщений ICMP об ошибках, которые уровень UDP передает «вверх» с разными целями [RFC1122]. Приложениям **следует** должным образом проверять содержимое сообщений ICMP, на предмет соответствия переданному трафику (например, сообщения об ошибках для фактически переданных приложением дейтаграмм UDP). Для этого нужен контекст, такой как локальное состояние экземпляров коммуникаций для каждого адресата, что не всегда доступно в приложениях UDP, хотя и поддерживается протоколами на основе соединений. Отметим, что не на всех платформах имеются API для таких проверок, а некоторые платформы сами выполняют проверки до передачи данных ICMP приложению.

Все откликам приложений на сообщения ICMP об ошибках **следует** быть устойчивыми к временным отказам маршрутизации (иногда их называют «мягкими ошибками»). Например, временные сообщения ICMP о недоступности не должны вести к прерыванию связи.

Сообщения ICMP активно фильтруются промежуточными устройствами, поэтому приложениям UDP **недопустимо** полагаться на их доставку для корректной и безопасной работы.

[1]Denial of Service - отказ в обслуживании.

6. Вопросы безопасности

UDP не обеспечивает защиты коммуникаций и приложениям, которым нужна защита от перехвата, подмены или поддержки сообщений, **следует** реализовать сквозные услуги защиты на основе других протоколов IETF.

Приложениям UDP **следует** поддерживать защиту от атак со вставкой данных извне путем использования случайного номера выходного порта или эквивалентного метода (параграф 5.1).

Приложения, которые могут отвечать на короткие запросы большими сообщениями, что создает возможность организации атак с усилением, **следует** принимать меры защиты от злонамеренного использования для организации DoS-атак. Это можно сделать путем проверки подлинности отправителя до отправки отклика, с учетом того, что IP-адрес отправителя не поможет при аутентификации, поскольку его легко подменить. Можно применять иное ограничение ситуаций, когда небольшой и непроверенный запрос может порождать отклик большого размера. Приложения **могут** также применять способы ограничения числа запросов, на которые будут передаваться отклики в течение определенного интервала времени для снижения расхода пропускной способности.

Одним из вариантов защиты коммуникаций UDP является использование IPsec [RFC4301], где возможна проверка подлинности потока пакетов IP с помощью заголовков аутентификации (Authentication Header или AH) [RFC4302], а также шифрование и/или аутентификация ESP[1] [RFC4303]. Приложения используют обмен ключами (Internet Key Exchange или IKE) [RFC7296] для настройки IPsec в своих сессиях. В зависимости от настройки IPsec для потока можно проверять подлинность и шифровать заголовки и данные UDP. Если приложению нужна лишь проверка подлинности, ESP без шифрования но с аутентификации часто бывает эффективней AH, поскольку ESP может работать через промежуточные устройства. Приложению, использующему IPsec, нужна ОС, реализующая стек протоколов IPsec, и путь в сети, поддерживающий трафик IKE и IPsec. Это может оказаться проще для реализаций IPv6 [RFC6092].

Хотя использование IPsec для защиты коммуникаций UDP возможно, не все ОС поддерживают IPsec и позволяют приложениям легко настроить защиту для их потоков. Другим вариантом защиты коммуникаций UDP является DTLS[2] [RFC6347][RFC7525], где обеспечивается защита конфиденциальности путем шифрования данных UDP без защиты заголвков UDP. Приложения могут реализовать DTLS без поддержки со стороны ОС.

Имеется много других вариантов аутентификации и шифрования трафика UDP. Например, можно применить модель GSS-API [RFC2743] или синтаксис криптографических сообщений (Cryptographic Message Syntax или CMS) [RFC5652]. Есть множество вариантов защиты RTP [RFC3550] при работе по протоколу UDP, в частности, для управления ключами [RFC7201]. Эти опции охватывают разные варианты применения, включая канала «точка-точка» и централизованные групповые коммуникации, а также групповую адресацию. В некоторых приложениях лучшим решением является защита больших автономных объектов, таких как файлы или сообщения, а не данных в отдельном пакете UDP. В таких случаях можно применять CMS [RFC5652], S/MIME [RFC5751] или OpenPGP [RFC4880]. Имеется также много протоколов защиты, разработанных вне IETF.

Подобно контролю перегрузок, механизмы защиты не просты в устройстве и корректной реализации. Поэтому **рекомендуется** применять проверенные механизмы защиты, такие как DTLS и IPsec, вместо разработки своих.

Механизм защиты с обобщенным TTL (Generalized TTL Security Mechanism или GTSM) [RFC5082] можно применять с приложениями UDP, когда целевая конечная точка подключена к тому же каналу. Этот облегченный механизм позволяет получателю фильтровать нежелательные пакеты.

В части контроля перегрузок [RFC2309] и [RFC2914] рассматривают опасность не реагирующих на перегрузку потоков для сети Internet. В [RFC8084] описаны методы, которые можно использовать для установки диапазона производительности, что может помочь для предотвращения коллапса насыщения или при некорректной реакции системы контроля перегрузок на факты насыщения. Этот документ содержит рекомендации для разработчиков приложений UDP по контролю перегрузок и не создает новых проблем безопасности.

Некоторые сетевые операторы столкнулись с всплесками трафика UDP, на несколько порядков превышающими обычную скорость для UDP. Это может побудить оператора к ограничению скорости для трафика UDP, что может привести к ограничению пропускной способности приложений при использовании UDP, а также увеличить потери пакетов UDP, которых не было бы при использовании других транспортных протоколов.

Приложение UDP с долгосрочной связью между отправителем и получателем должно позволять отправителю периодически проверять согласие получателя на прием данных, если для этого не используются явные подтверждения [RFC7675]. Приложения, которым нужна двухсторонняя связь для реализации функций протокола (таких как гарантии доставки или контроль перегрузок), потребуется независимая проверка в обоих направлениях, и может потребоваться дополнительный обмен сообщениями keep-alive для работы через промежуточные устройства (параграф 3.5).

7. Заключение

В этом разделе приведена сводка основных рекомендаций разделов 3 - 6 в форме таблицы (таблица 1) для удобства.

Таблица 1. Сводка рекомендаций.

Рекомендация	***Параграф***
Должна поддерживаться работа по широкому диапазону путей Internet.	3
Следует применять полнофункциональный транспорт (например, TCP)	
Следует контролировать скорость передачи	3.1
Следует перегрузки для всего трафика	
Для передачи больших объемов данных	3.1.2
следует рассмотреть реализацию TFRC, в противном случае	
следует иным способом использовать пропускную способность по аналогии с TCP	
Для передачи небольших объемов данных	3.1.3

[1]Encapsulating Security Payload - инкапсуляция защищенных данных.
[2]Datagram Transport Layer Security - защита транспортного уровня дейтаграмм.

Следует измерять RTT и передавать не более 1 дейтаграммы за интервал RTT,	3.1.1
иначе **следует** передавать не более 1 дейтаграммы за 3 секунды.	
Следует восстанавливать (back-off) таймеры повтора после потери пакетов	
Следует поддерживать механизмы для контроля пиков передачи	3.1.6
Можно реализовать ECN, при этом будет **требоваться** определенный набор механизмов в приложении	3.1.7
Для DiffServ **не следует** опираться на реализацию PHB	3.1.8
Для путей с поодержкой QoS **можно** отказаться от использования CC	3.1.9
Не следует опираться лишь на QoS для пропускной способности	3.1.10
Для потоков без управления **следует** реализовать транспортные выключатели	
Можно реализовать транспортны выключатели для других приложений	
Для туннелей с трафиком IP	3.1.11
не следует применять контроль перегрузок	
и **должно** корректно обрабатываться поле IP ECN	
Для туннелей не-IP или не задаваемой трафиком скорости	3.1.11
следует выполнять CC или применять выключатель,	
следует ограничивать типы трафика, доставляемого через туннель	
Не следует передавать дейтаграммы размером больше PMTU, т. е.	3.2
следует определять PMTU или передавать дейтаграммы меньше минимального PMTU	
При использовании PLPMTUD **требуются** конкретные механизмы приложения	
Следует обрабатывать потерю, дублирование и нарушение порядка дейтаграмм	3.3
Следует обеспечивать устойчивость к задержкам доставки до 2 минут	
Следует включать контрольную сумму UDP для IPv4	3.4
Следует включать контрольную сумму UDP для IPv6; **требуются** конкретные механизмы для нулевой	3.4.1
контрольной суммы UDP	
Следует поддерживать механизм защиты от атак off-path,	5.1
иначе можно применять UDP-Lite с подходящим покрытием контрольной суммы	3.4.2
Не следует всегда передавать сообщения keep-alive для промежуточных устройств	3.5
При необходимости **можно** использовать keep-alive с интервалом не менее 15 секунд	
Приложениям с ограниченным применением (или контролируемой средой) **следует** указывать	3.6
эквивалентные механизмы и описывать их примененеи	
Групповым приложениям с большим трафиком **следует** реализовать контроль перегрузок	4.1.1
Групповым приложениям с небольшим трафиком **следует** реализовать контроль перегрузок	4.1.2
Групповым приложениям **следует** применять безопасное значение PMTU	4.2
Следует избегать применения множества портов	5.1.2
Должны проверяться адреса отправителей в принятых пакетах	
Следует проверять данные в сообщениях ICMP	5.2
Следует использовать случайный выходной порт или эквивалентный метод, а для приложений «клиент-сервер» **следует** передавать отклики с адреса, соответствующего запросу	6
При необходимости **следует** использовать стандартные протоколы защиты IETF	6

8. Литература

8.1. Нормативные документы

[RFC768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, DOI 10.17487/RFC0768, August 1980, <http://www.rfc-editor.org/info/rfc768>.

[RFC793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://www.rfc-editor.org/info/rfc793>.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <http://www.rfc-editor.org/info/rfc1122>.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <http://www.rfc-editor.org/info/rfc1191>.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, DOI 10.17487/RFC1981, August 1996, <http://www.rfc-editor.org/info/rfc1981>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, DOI 10.17487/RFC2914, September 2000, <http://www.rfc-editor.org/info/rfc2914>.

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., Ed., and G. Fairhurst, Ed., "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, DOI 10.17487/RFC3828, July 2004, <http://www.rfc-editor.org/info/rfc3828>.

[RFC4787] Audet, F., Ed. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, DOI 10.17487/RFC4787, January 2007, <http://www.rfc-editor.org/info/rfc4787>.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <http://www.rfc-editor.org/info/rfc4821>.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, DOI 10.17487/RFC5348, September 2008, <http://www.rfc-editor.org/info/rfc5348>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, DOI 10.17487/RFC5405, November 2008, <http://www.rfc-editor.org/info/rfc5405>.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, DOI 10.17487/RFC6040, November 2010, <http://www.rfc-editor.org/info/rfc6040>.

[RFC6298] Paxson, V., Allman, M., Chu, J., and M. Sargent, "Computing TCP's Retransmission Timer", RFC 6298, DOI 10.17487/RFC6298, June 2011, <http://www.rfc-editor.org/info/rfc6298>.

[RFC8084] Fairhurst, G., "Network Transport Circuit Breakers", BCP 208, RFC 8084, DOI 10.17487/RFC8084, March 2017, <http://www.rfc-editor.org/info/rfc8084>.

8.2. Дополнительная литература

[ALLMAN] Allman, M. and E. Blanton, "Notes on burst mitigation for transport protocols", March 2005.

[BEHAVE-APP] Ford, B., "Application Design Guidelines for Traversal through Network Address Translators", Work in Progress, draft-ford-behave-app-05, March 2007.

[ENCAP] Nordmark, E., Ed., Tian, A., Gross, J., Hudson, J., Kreeger, L., Garg, P., Thaler, P., and T. Herbert, "Encapsulation Considerations", Work in Progress, draft-ietf-rtgwg-dt-encap-02, October 2016.

[FABER] Faber, T., Touch, J., and W. Yue, "The TIME-WAIT State in TCP and Its Effect on Busy Servers", Proc. IEEE Infocom, March 1999.

[INT-TUNNELS] Touch, J. and W. Townsley, "IP Tunnels in the Internet Architecture", Work in Progress, draft-ietf-intarea-tunnels-03, July 2016.

[POSIX] IEEE Std. 1003.1-2001, , "Standard for Information Technology - Portable Operating System Interface (POSIX)", Open Group Technical Standard: Base Specifications Issue 6, ISO/IEC 9945:2002, December 2001.

[RFC919] Mogul, J., "Broadcasting Internet Datagrams", STD 5, RFC 919, DOI 10.17487/RFC0919, October 1984, <http://www.rfc-editor.org/info/rfc919>.

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, DOI 10.17487/RFC1112, August 1989, <http://www.rfc-editor.org/info/rfc1112>.

[RFC1536] Kumar, A., Postel, J., Neuman, C., Danzig, P., and S. Miller, "Common DNS Implementation Errors and Suggested Fixes", RFC 1536, DOI 10.17487/RFC1536, October 1993, <http://www.rfc-editor.org/info/rfc1536>.

[RFC1546] Partridge, C., Mendez, T., and W. Milliken, "Host Anycasting Service", RFC 1546, DOI 10.17487/RFC1546, November 1993, <http://www.rfc-editor.org/info/rfc1546>.

[RFC2309] Braden, B., Clark, D., Crowcroft, J., Davie, B., Deering, S., Estrin, D., Floyd, S., Jacobson, V., Minshall, G., Partridge, C., Peterson, L., Ramakrishnan, K., Shenker, S., Wroclawski, J., and L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", RFC 2309, DOI 10.17487/RFC2309, April 1998, <http://www.rfc-editor.org/info/rfc2309>.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, DOI 10.17487/RFC2475, December 1998, <http://www.rfc-editor.org/info/rfc2475>.

[RFC2675] Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, DOI 10.17487/RFC2675, August 1999, <http://www.rfc-editor.org/info/rfc2675>.

[RFC2743] Linn, J., "Generic Security Service Application Program Interface Version 2, Update 1", RFC 2743, DOI 10.17487/RFC2743, January 2000, <http://www.rfc-editor.org/info/rfc2743>.

[RFC2887] Handley, M., Floyd, S., Whetten, B., Kermode, R., Vicisano, L., and M. Luby, "The Reliable Multicast Design Space for Bulk Data Transfer", RFC 2887, DOI 10.17487/RFC2887, August 2000, <http://www.rfc-editor.org/info/rfc2887>.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, DOI 10.17487/RFC2983, October 2000, <http://www.rfc-editor.org/info/rfc2983>.

[RFC3048] Whetten, B., Vicisano, L., Kermode, R., Handley, M., Floyd, S., and M. Luby, "Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfer", RFC 3048, DOI 10.17487/RFC3048, January 2001, <http://www.rfc-editor.org/info/rfc3048>.

[RFC3124] Balakrishnan, H. and S. Seshan, "The Congestion Manager", RFC 3124, DOI 10.17487/RFC3124, June 2001, <http://www.rfc-editor.org/info/rfc3124>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <http://www.rfc-editor.org/info/rfc3168>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC3303] Srisuresh, P., Kuthan, J., Rosenberg, J., Molitor, A., and A. Rayhan, "Middlebox communication architecture and framework", RFC 3303, DOI 10.17487/RFC3303, August 2002, <http://www.rfc-editor.org/info/rfc3303>.

[RFC3493] Gilligan, R., Thomson, S., Bound, J., McCann, J., and W. Stevens, "Basic Socket Interface Extensions for IPv6", RFC 3493, DOI 10.17487/RFC3493, February 2003, <http://www.rfc-editor.org/info/rfc3493>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, DOI 10.17487/RFC3551, July 2003, <http://www.rfc-editor.org/info/rfc3551>.

[RFC3738] Luby, M. and V. Goyal, "Wave and Equation Based Rate Control (WEBRC) Building Block", RFC 3738, DOI 10.17487/RFC3738, April 2004, <http://www.rfc-editor.org/info/rfc3738>.

[RFC3758] Stewart, R., Ramalho, M., Xie, Q., Tuexen, M., and P. Conrad, "Stream Control Transmission Protocol (SCTP) Partial Reliability Extension", RFC 3758, DOI 10.17487/RFC3758, May 2004, <http://www.rfc-editor.org/info/rfc3758>.

[RFC3819] Karn, P., Ed., Bormann, C., Fairhurst, G., Grossman, D., Ludwig, R., Mahdavi, J., Montenegro, G., Touch, J., and L. Wood, "Advice for Internet Subnetwork Designers", BCP 89, RFC 3819, DOI 10.17487/RFC3819, July 2004, <http://www.rfc-editor.org/info/rfc3819>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <http://www.rfc-editor.org/info/rfc4301>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <http://www.rfc-editor.org/info/rfc4302>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <http://www.rfc-editor.org/info/rfc4303>.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, DOI 10.17487/RFC4340, March 2006, <http://www.rfc-editor.org/info/rfc4340>.

[RFC4341] Floyd, S. and E. Kohler, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control", RFC 4341, DOI 10.17487/RFC4341, March 2006, <http://www.rfc-editor.org/info/rfc4341>.

[RFC4342] Floyd, S., Kohler, E., and J. Padhye, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 3: TCP-Friendly Rate Control (TFRC)", RFC 4342, DOI 10.17487/RFC4342, March 2006, <http://www.rfc-editor.org/info/rfc4342>.

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, DOI 10.17487/RFC4380, February 2006, <http://www.rfc-editor.org/info/rfc4380>.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, DOI 10.17487/RFC4607, August 2006, <http://www.rfc-editor.org/info/rfc4607>.

[RFC4654] Widmer, J. and M. Handley, "TCP-Friendly Multicast Congestion Control (TFMCC): Protocol Specification", RFC 4654, DOI 10.17487/RFC4654, August 2006, <http://www.rfc-editor.org/info/rfc4654>.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, DOI 10.17487/RFC4880, November 2007, <http://www.rfc-editor.org/info/rfc4880>.

[RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", RFC 4890, DOI 10.17487/RFC4890, May 2007, <http://www.rfc-editor.org/info/rfc4890>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <http://www.rfc-editor.org/info/rfc4960>.

[RFC4963] Heffner, J., Mathis, M., and B. Chandler, "IPv4 Reassembly Errors at High Data Rates", RFC 4963, DOI 10.17487/RFC4963, July 2007, <http://www.rfc-editor.org/info/rfc4963>.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", RFC 4987, DOI 10.17487/RFC4987, August 2007, <http://www.rfc-editor.org/info/rfc4987>.

[RFC5082] Gill, V., Heasley, J., Meyer, D., Savola, P., Ed., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, DOI 10.17487/RFC5082, October 2007, <http://www.rfc-editor.org/info/rfc5082>.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, DOI 10.17487/RFC5245, April 2010, <http://www.rfc-editor.org/info/rfc5245>.

[RFC5622] Floyd, S. and E. Kohler, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion ID 4: TCP-Friendly Rate Control for Small Packets (TFRC-SP)", RFC 5622, DOI 10.17487/RFC5622, August 2009, <http://www.rfc-editor.org/info/rfc5622>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <http://www.rfc-editor.org/info/rfc5652>.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, DOI 10.17487/RFC5681, September 2009, <http://www.rfc-editor.org/info/rfc5681>.

[RFC5740] Adamson, B., Bormann, C., Handley, M., and J. Macker, "NACK-Oriented Reliable Multicast (NORM) Transport Protocol", RFC 5740, DOI 10.17487/RFC5740, November 2009, <http://www.rfc-editor.org/info/rfc5740>.

[RFC5751] Ramsdell, B. and S. Turner, "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, DOI 10.17487/RFC5751, January 2010, <http://www.rfc-editor.org/info/rfc5751>.

[RFC5775] Luby, M., Watson, M., and L. Vicisano, "Asynchronous Layered Coding (ALC) Protocol Instantiation", RFC 5775, DOI 10.17487/RFC5775, April 2010, <http://www.rfc-editor.org/info/rfc5775>.

[RFC5971] Schulzrinne, H. and R. Hancock, "GIST: General Internet Signalling Transport", RFC 5971, DOI 10.17487/RFC5971, October 2010, <http://www.rfc-editor.org/info/rfc5971>.

[RFC5973] Stiemerling, M., Tschofenig, H., Aoun, C., and E. Davies, "NAT/Firewall NSIS Signaling Layer Protocol (NSLP)", RFC 5973, DOI 10.17487/RFC5973, October 2010, <http://www.rfc-editor.org/info/rfc5973>.

[RFC6056] Larsen, M. and F. Gont, "Recommendations for Transport-Protocol Port Randomization", BCP 156, RFC 6056, DOI 10.17487/RFC6056, January 2011, <http://www.rfc-editor.org/info/rfc6056>.

[RFC6092] Woodyatt, J., Ed., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, DOI 10.17487/RFC6092, January 2011, <http://www.rfc-editor.org/info/rfc6092>.

[RFC6335] Cotton, M., Eggert, L., Touch, J., Westerlund, M., and S. Cheshire, "Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry", BCP 165, RFC 6335, DOI 10.17487/RFC6335, August 2011, <http://www.rfc-editor.org/info/rfc6335>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[RFC6396] Blunk, L., Karir, M., and C. Labovitz, "Multi-Threaded Routing Toolkit (MRT) Routing Information Export Format", RFC 6396, DOI 10.17487/RFC6396, October 2011, <http://www.rfc-editor.org/info/rfc6396>.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, DOI 10.17487/RFC6437, November 2011, <http://www.rfc-editor.org/info/rfc6437>.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <http://www.rfc-editor.org/info/rfc6438>.

[RFC6513] Rosen, E., Ed. and R. Aggarwal, Ed., "Multicast in MPLS/BGP IP VPNs", RFC 6513, DOI 10.17487/RFC6513, February 2012, <http://www.rfc-editor.org/info/rfc6513>.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, DOI 10.17487/RFC6679, August 2012, <http://www.rfc-editor.org/info/rfc6679>.

[RFC6726] Paila, T., Walsh, R., Luby, M., Roca, V., and R. Lehtonen, "FLUTE - File Delivery over Unidirectional Transport", RFC 6726, DOI 10.17487/RFC6726, November 2012, <http://www.rfc-editor.org/info/rfc6726>.

[RFC6773] Phelan, T., Fairhurst, G., and C. Perkins, "DCCP-UDP: A Datagram Congestion Control Protocol UDP Encapsulation for NAT Traversal", RFC 6773, DOI 10.17487/RFC6773, November 2012, <http://www.rfc-editor.org/info/rfc6773>.

[RFC6807] Farinacci, D., Shepherd, G., Venaas, S., and Y. Cai, "Population Count Extensions to Protocol Independent Multicast (PIM)", RFC 6807, DOI 10.17487/RFC6807, December 2012, <http://www.rfc-editor.org/info/rfc6807>.

[RFC6887] Wing, D., Ed., Cheshire, S., Boucadair, M., Penno, R., and P. Selkirk, "Port Control Protocol (PCP)", RFC 6887, DOI 10.17487/RFC6887, April 2013, <http://www.rfc-editor.org/info/rfc6887>.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, DOI 10.17487/RFC6935, April 2013, <http://www.rfc-editor.org/info/rfc6935>.

[RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, DOI 10.17487/RFC6936, April 2013, <http://www.rfc-editor.org/info/rfc6936>.

[RFC6951] Tuexen, M. and R. Stewart, "UDP Encapsulation of Stream Control Transmission Protocol (SCTP) Packets for End-Host to End-Host Communication", RFC 6951, DOI 10.17487/RFC6951, May 2013, <http://www.rfc-editor.org/info/rfc6951>.

[RFC7143] Chadalapaka, M., Satran, J., Meth, K., and D. Black, "Internet Small Computer System Interface (iSCSI) Protocol (Consolidated)", RFC 7143, DOI 10.17487/RFC7143, April 2014, <http://www.rfc-editor.org/info/rfc7143>.

[RFC7201] Westerlund, M. and C. Perkins, "Options for Securing RTP Sessions", RFC 7201, DOI 10.17487/RFC7201, April 2014, <http://www.rfc-editor.org/info/rfc7201>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

[RFC7450] Bumgardner, G., "Automatic Multicast Tunneling", RFC 7450, DOI 10.17487/RFC7450, February 2015, <http://www.rfc-editor.org/info/rfc7450>.

[RFC7510] Xu, X., Sheth, N., Yong, L., Callon, R., and D. Black, "Encapsulating MPLS in UDP", RFC 7510, DOI 10.17487/RFC7510, April 2015, <http://www.rfc-editor.org/info/rfc7510>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <http://www.rfc-editor.org/info/rfc7525>.

[RFC7560] Kuehlewind, M., Ed., Scheffenegger, R., and B. Briscoe, "Problem Statement and Requirements for Increased Accuracy in Explicit Congestion Notification (ECN) Feedback", RFC 7560, DOI 10.17487/RFC7560, August 2015, <http://www.rfc-editor.org/info/rfc7560>.

[RFC7567] Baker, F., Ed. and G. Fairhurst, Ed., "IETF Recommendations Regarding Active Queue Management", BCP 197, RFC 7567, DOI 10.17487/RFC7567, July 2015, <http://www.rfc-editor.org/info/rfc7567>.

[RFC7605] Touch, J., "Recommendations on Using Assigned Transport Port Numbers", BCP 165, RFC 7605, DOI 10.17487/RFC7605, August 2015, <http://www.rfc-editor.org/info/rfc7605>.

[RFC7657] Black, D., Ed. and P. Jones, "Differentiated Services (Diffserv) and Real-Time Communication", RFC 7657, DOI 10.17487/RFC7657, November 2015, <http://www.rfc-editor.org/info/rfc7657>.

[RFC7675] Perumal, M., Wing, D., Ravindranath, R., Reddy, T., and M. Thomson, "Session Traversal Utilities for NAT (STUN) Usage for Consent Freshness", RFC 7675, DOI 10.17487/RFC7675, October 2015, <http://www.rfc-editor.org/info/rfc7675>.

[RFC8083] Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", RFC 8083, DOI 10.17487/RFC8083, March 2017, <http://www.rfc-editor.org/info/rfc8083>.

[RFC8086] Yong, L., Ed., Crabbe, E., Xu, X., and T. Herbert, "GRE-in-UDP Encapsulation", RFC 8086, DOI 10.17487/RFC8086, March 2017, <http://www.rfc-editor.org/info/rfc8086>.

[RFC8087] Fairhurst, G. and M. Welzl, "The Benefits of Using Explicit Congestion Notification (ECN)", RFC 8087, DOI 10.17487/RFC8087, March 2017, <http://www.rfc-editor.org/info/rfc8087>.

[STEVENS] Stevens, W., Fenner, B., and A. Rudoff, "UNIX Network Programming, The sockets Networking API", Addison-Wesley, 2004.

[UPnP] UPnP Forum, , "Internet Gateway Device (IGD) Standardized Device Control Protocol V 1.0", November 2001.

Приложение A. Пример использования режима IPv6 UDP Zero-Checksum

В этом приложении дан краткий обзор MPLS-in-UDP в качестве примера туннельной инкапсуляции UDP. Цель приложения состоит в предоставлении конкретного примера безопасного использования режима UDP zero-checksum для туннелей MPLS-in-UDP по протоколу IPv6. По умолчанию UDP требует использования контрольных сумм с IPv6. Задана опция, позволяющая использовать нулевую контрольную сумму дейтаграмм UDP в IPv6 для работы в определенных средах, указанных в [RFC7510], и определен набор ограничений для работы в этом режиме.

Туннели или инкапсуляцию UDP с использованием нулевой контрольной суммы в IPv6 можно разворачивать лишь в сети одного оператора или в смежных сетях взаимодействующих операторов с контролем перегрузки, а не через Internet, где требуется контроль перегрузок. Решение MPLS-in-UDP было разработано для сетей с единым администрирование (таких, как сеть одного оператора), где известно (возможно из сведений о типах оборудования и проверках нижележащих уровней), что повреждения пакетов очень редки и оператор готов принять риск повреждения незащищенных пакетов.

Инкапсулятору туннеля **следует** выбирать разные адреса IPv6 для каждого туннеля с нулевой контрольной суммой UDP, независимо от декапсулятора и строгости проверки им адресов отправителей IPv6 (один групповой или индивидуальный адрес отправителя IPv6 **не следует** использовать для разных получателей). Применение MPLS-in-UDP можно расширить на смежные сети взаимодействующих операторов с согласованным администрированием (сети, где операторы согласны взаимодействовать для предоставления общего набора услуг) [RFC7510].

Требования к конечным точкам MPLS-in-UDP проверять адрес отправителя IPv6 в дополнение к адресу получателя и строгая рекомендация не применять один адрес отправителя IPv6 для разных туннелей MPLS-in-UDP обеспечивают некоторую компенсацию отсутствия контрольной суммы UDP в заголовке IPv6. Кроме того, плоскость данных MPLS пересылает лишь пакеты с действительными метками (т. е. метками, распространенными выходными LSR[1] туннелей), что обеспечивает дополнительные возможности обнаружения ошибочно доставленных пакетов MPLS-in-UDP, если те содержат недействительную метку для пересылки на принимающем LSR. Ожидаемый результат для IPv6 с нулевой контрольной суммой UDP для MPLS-in-UDP заключается в том, что повреждение адреса получателя IPv6 обычно будет приводить к отбрасыванию пакетов.

Дополнительную уверенность обеспечивают ограничения приведенных выше исключений, позволяющие применять режим IPv6 UDP zero-checksum лишь в хорошо управляемых сетях, где повреждения пакетов MPLS не создают практических проблем. Поэтому MPLS-in-UDP подходит для передачи через нижние уровни в хорошо управляемых сетях, разрешенных отмеченными выше исключениями, где не ожидается рост частоты повреждения внутренних пакетов IP по сравнению с трафиком MPLS без инкапсуляции UDP. Упомянутые причины позволяют отказаться в MPLS-in-UDP от дополнительной проверки целостности в режиме с нулевой контрольной суммой UDP для IPv6 и счесть это решение соответствующим требованиям 2, 3 и 5 раздела 5 в [RFC6936].

Инкапсуляция MPLS-in-UDP не включает механизма безопасного возврата к использованию контрольных сумм при изменении пути, перенаправляющем туннель на участки с промежуточными устройствами, которые отбрасывают дейтаграммы IPv6 с нулевой контрольной суммой UDP. В этом случае промежуточное устройство делает туннель MPLS-in-UDP «черной дырой». Рекомендации, позволяющие МСЭ, трансляторам NAT и другим промежуточным устройствам поддерживать использование IPv6 с нулевой контрольной суммой UDP, приведены в разделе 5 [RFC6936]. MPLS не аккумулирует некорректное состояние в результате повреждения стека меток. Поврежденная метка MPLS ведет к отбрасыванию пакета или его пересылки (и забвению) без накопления состояния протокола MPLS. **Требуется** активный мониторинг ошибок в трафике MPLS-in-UDP, поскольку возникновение ошибки приведет к некоторому накоплению данных об ошибках за пределами протокола MPLS для целей эксплуатации и управления. Это соответствует требованию 4 из раздела 5 в [RFC6936]. Кроме того, оператор сети **должен** активно отслеживать наличие ошибок в трафике IPv6 с нулевой контрольной суммой.

Операторам **следует** также развернуть пакетные фильтры для предотвращения утечки пакетов IPv6 с нулевой контрольной суммой UDP за пределы сети в результате ошибок в конфигурации или пакетах. Оператор сети **должен** обеспечивать активный мониторинг трафика IPv6 с нулевой контрольной суммой UDP.

[1]Label Switched Router - маршрутизатор с коммутацией по меткам.

Благодарности

Рекомендации по работе через промежуточные устройства в параграфе 3.5 включают идеи из раздела 5 работы [BEHAVE-APP] Bryan Ford, Pyda Srisuresh, Dan Kegel. Рекомендации по таймерам протокола в параграфе 3.1.1 в значительной степени представлены Mark Allman.

G. Fairhurst получил финансирование в рамках программы Европейского союза Horizon 2020 и инновационной программы 2014-2018 по гранту 644334 (NEAT). Lars Eggert получил финансирование в рамках программы Европейского союза Horizon 2020 и инновационной программы 2014-2018 по гранту 644866 (SSICLOPS). Этот документ отражает лишь точку зрения авторов и Европейская комиссия не несет ответственности за любое использование содержащейся в документе информации.

Адреса авторов

Lars Eggert

NetApp

Sonnenallee 1

Kirchheim 85551

Germany

Phone: +49 151 120 55791

Email: lars@netapp.com

URI: https://eggert.org/

Godred Fairhurst

University of Aberdeen

Department of Engineering

Fraser Noble Building

Aberdeen AB24 3UE

Scotland

Email: gorry@erg.abdn.ac.uk

URI: http://www.erg.abdn.ac.uk/

Greg Shepherd

Cisco Systems

Tasman Drive

San Jose

United States of America

Email: gjshep@gmail.com

Перевод на русский язык
Николай Малых

nmalykh@protocols.ru