Internet Engineering Task Force (IETF)
Request for Comments: 9040
Obsoletes: 2140
Category: Informational
ISSN: 2070-1721

J. Touch
Independent
M. Welzl
S. Islam
University of Oslo

July 2021

TCP Control Block Interdependence

Взаимозависимость блоков управления TCP

Аннотация

Этот документ содержит рекомендации для разработчиков TCP, предназначенные для улучшения сходимости соединений к устойчивому состоянию без ущерба для функциональной совместимости. Документ обновляет и отменяет описание RFC 2140 для совместного использования состояния TCP, обычно представленного в блоках управления TCP (Control Block), похожими одновременными или последовательными соединениями.

Статус документа

Документ не относится к категории Internet Standards Track и публикуется для информации.

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

Информацию о текущем статусе документа, ошибках и способах обратной связи можно найти по ссылке https://www.rfc-editor.org/info/rfc9040.

Авторские права

Copyright (c) 2021. Авторские права принадлежат IETF Trust и лицам, указанным в качестве авторов документа. Все права защищены.

К документу применимы права и ограничения, указанные в BCP 78 и IETF Trust Legal Provisions и относящиеся к документам IETF (http://trustee.ietf.org/license-info), на момент публикации данного документа. Прочтите упомянутые документы внимательно. Фрагменты программного кода, включённые в этот документ, распространяются в соответствии с упрощённой лицензией BSD, как указано в параграфе 4.e документа IETF Trust Legal Provisions, без каких-либо гарантий (как указано в Simplified BSD License).

Оглавление

[1]Internet Engineering Task Force - комиссия по решению инженерных задач Internet.
[2]Internet Engineering Steering Group - комиссия по инженерным разработкам Internet.

1. Введение

TCP - это ориентированный на соединения транспортный протокол с гарантией доставки, работающий на основе [RFC0793]. Для каждого соединения TCP поддерживается состояние, обычно в структуре данных, называемой блоком управления TCP (TCP Control Block или TCB). TCB содержит сведения о состоянии соединения, связанных с ним локальных процессах и параметры обратной связи о свойствах передачи данных через соединение. Как задано исходно и принято в большинстве реализаций, большая часть сведений TCB поддерживается на уровне каждого соединения. В некоторых реализациях часть данных TCB совместно используется соединениями с одним хостом [RFC2140]. Предполагается, что это обеспечивает повышение общей производительности в переходные периоды, особенно при большом числе краткосрочных или одновременных соединений, как это бывает в WWW и других приложениях [Be94] [Br02]. обобществление состояния призвано помочь соединениям TCP быстрее сходиться к долгосрочному поведению (при условии стабильной нагрузки приложения, т. е. в установившемся состоянии) без ущерба для функциональной совместимости TCP.

Этот документ обновляет обсуждение совместного использования состояния TCB в RFC 2140 и полностью заменяет упомянутый документ. обобществление состояния влияет лишь на инициализацию TCB [RFC2140] и не оказывает воздействия на долгосрочное состояние TCP после организации соединения и на функциональную совместимость. Сведения о пути, совместно используемые для портов получателя SYN, предполагают, что сегменты TCP между одной парой хостов имеют общие свойства пути, т. е. трафик не маршрутизируется по-разному в зависимости от номера порта или других параметров соединения (см. параграф 8.1). Замечания о совместном использовании TCB в этом документе применимы к любому протоколу с состоянием контроля перегрузок, включая протокол управления потоковой передачей (Stream Control Transmission Protocol или SCTP) [RFC4960] и протокол контроля перегрузок для дейтаграмм (Datagram Congestion Control Protocol или DCCP) [RFC4340], а также отдельным субпотокам в Multipath TCP [RFC8684].

2. Используемые соглашения

Ключевые слова **должно** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не следует** (SHALL NOT), **следует** (SHOULD), **не нужно** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **не рекомендуется** (NOT RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе интерпретируются в соответствии с BCP 14 [RFC2119] [RFC8174] тогда и только тогда, когда они выделены шрифтом, как показано здесь.

Основная часть этого документа посвящена описанию поведения, уже разрешённого стандартами TCP. В результате документ содержит нормативные указания, но не использует язык описания требований за исключением случаев цитирования других документов. Нормативные требования используются в Приложении C как примеры требований для рассмотрения в будущем.

3. Терминология

Ниже перечислены термины, часто используемые в документе. Элементы с префиксом + могут быть частью поддерживаемого состояния соединения TCP в TCB связанных соединений и являться предметом обобществления, как описано в этом документе. Отметим, что термины используются в их исходном смысле, когда это возможно. В некоторых случаях направление указывается суффиксом _S для передачи и _R для приёма, а иногда явно (sendcwnd).

+cwnd
Размер окна перегрузки TCP [RFC5681].

host
Источник или приёмник сегментов TCP, связанных с одним адресом IP.

host-pair
Пара хостов и их соответствующих адресов IP.

ISN
Initial Sequence Number - начальный порядковый номер.

+MMS_R
Максимальный размер сообщения, которое может быть принято, наибольший размер принимаемых данных транспортного уровня в дейтаграмме IP [RFC1122].

+MMS_S
Максимальный размер сообщения, которое может быть передано, наибольший размер передаваемых данных транспортного уровня в дейтаграмме IP [RFC1122].

path
Путь Internet между IP-адресами пары хостов.

PCB
Протокольный блок управления - связанные с протоколом данные, поддерживаемые конечной точкой. Для протокола TCP блок PCB называется TCB.

PLPMTUD
Обнаружение MTU на пути для уровня пакетизации - механизм, использующий транспортные пакеты для определения максимального передаваемого по пути блока (Path Maximum Transmission Unit или PMTU) [RFC4821].

+PMTU
Наибольший размер дейтаграммы IP, которая может быть передана через путь [RFC1191] [RFC8201].

PMTUD
Обнаружение MTU на уровне пути - механизм, основанный на сообщениях ICMP об ошибках, для определения PMTU [RFC1191] [RFC8201].

+RTT
Время кругового обхода при обмене пакетами TCP [RFC0793].
+RTTVAR
Вариации времени кругового обхода при обмене пакетами TCP [RFC6298].
+rwnd
Размер приёмного окна TCP [RFC5681].
+sendcwnd
Размер окна перегрузки на приёмной стороне TCP (cwnd) [RFC5681].
+sendMSS
Максимальный размер сегмента TCP - значение, передаваемое в опции TCP, которое представляет максимальный размер пользовательских данных TCP, которые могут быть приняты [RFC6691].
+ssthresh
Порог замедленного старта TCP [RFC5681].
TCB
Блок управления TCP - поддерживаемые конечной точкой данные, связанные с соединением TCP.
TCP-AO
Опция аутентификации TCP [RFC5925].
TFO
Опция TCP Fast Open [RFC7413].
+TFO_cookie
TCP Fast Open cookie - состояние, используемое как часть механизма TFO при поддержке TFO [RFC7413].
+TFO_failure
индикация отказа при согласовании опции TFO в случае поддержки TFO.
+TFOinfo
Информация, кэшируемая при организации соединения TFO, которая включает TFO_cookie [RFC7413].

4. Блок управления TCP (TCB)

TCB описывает данные, связанные с каждым соединением, т. е. взаимодействием пары приложений через сеть. В состав TCB входят по меньшей мере указанные ниже сведения [RFC0793]:

Состояние локального процесса
указатели на буферы приёма и передачи
указатели на очередь повторной передачи и текущий сегмент
указатели на IP PCB
Обобществлённое состояние на уровне соединения
macro-state
состояние соединения
таймеры
флаги
адреса и номера портов локального и удалённого хоста
состояние опций TCP
micro-state
состояние окна приёма и передачи (size*, текущий номер)
размер окна перегрузки (sendcwnd)*
пороговый размер окна перегрузки (ssthresh)*
максимальный наблюдаемый размер окна*
sendMSS#
MMS_S#
MMS_R#
PMTU#
время кругового обхода и его вариации#

Сведения по соединениям разделены на макро (macro-state) и микросостояние (micro-state), как принято в [Co91]. Макросостояние описывает протокол для организации начального общего состояния соединения - адреса и номера портов конечных точек, а также компоненты (таймеры, флаги), требуемые при организации и затем используемые для поддержки состояния. Микросостояние описывает протокол после организации соединения для поддержки надёжности и контроля перегрузок при передаче данных через соединение.

Выделены два других класса общего микросостояния, связанные больше с парой хостов, чем с парой приложений. Один класс явно зависит от пары хостов (суффикс #, например, sendMSS, MMS_R, MMS_S, PMTU, RTT), поскольку эти параметры определяются конечной точкой или парой конечных точек (в данном примере sendMSS, MMS_R, MMS_S, RTT) или уже кэшированы и обобществлены на основе этого (в данном примере PMTU [RFC1191] [RFC4821]). Другой класс зависит от пары хостов в совокупности (суффикс *, например, сведения об окне перегрузки, размеры текущих окон и т. п.), поскольку параметры определяются общей пропускной способностью между двумя конечными точками.

Не все данные состояния в TCB обязательно обобществляются. В частности, некоторые опции TCP согласуются лишь по запросу прикладного уровня, поэтому их использование может быть разным в соединениях. Другие опции согласуют зависящие от соединения параметры, которые также не являются общими. Это дополнительно рассматривается в Приложении B.

Параметр rwnd далее не рассматривается, поскольку его значению следует зависеть от обобществляемого размера окна передачи, поэтому оно уже учтено и не используется независимо.

5. Взаимозависимость TCB

Существует два случая взаимозависимости TCB. Временное обобществление происходит, когда TCB прежнего (сейчас CLOSED) соединения с хостом служит для инициализации некоторых параметров нового (последующего) соединения с тем же хостом. Обобществление в ансамбле происходит когда текущее активное соединение с хостом применяется для инициализации нового (одновременного) соединения с тем же хостом.

6. Временное обобществление

Доступ к кэшу данных TCB осуществляется двумя способами - чтение при инициализации новых TCB и запись при наличии более актуального состояния на уровне хоста.

6.1. Инициализация нового TCB

TCB для новых соединений могут инициализироваться с использованием кэшированного содержимого прежних соединений, как показано в таблице 1.

Таблица 1. Временное обобществление - инициализация TCB.

Кэшированный TCB	***Новый TCB***
old_MMS_S	old_MMS_S или без кэширования[1]
old_MMS_R	old_MMS_S или без кэширования[1]
old_sendMSS	old_sendMSS
old_PMTU	old_PMTU[2]
old_RTT	old_RTT
old_RTTVAR	old_RTTVAR
old_option	(зависит от опции)
old_ssthresh	old_ssthresh
old_sendcwnd	old_sendcwnd

В таблице 2 приведён обзор связанных с опциями сведений, которые могут обобществляться. Дополнительные сведения о обобществлении конкретных опций TCP приведены в Приложении B.

Таблица 2. Временное обобществление - инициализация опций.

Кэшированная	***Новая***
old_TFO_cookie	old_TFO_cookie
old_TFO_failure	old_TFO_failure

6.2. Обновления кэша TCB

В процессе работы соединения кэш TCB может обновляться по событиям в текущих соединениях и по мере изменения их TCB с течением времени, как показано в таблице 3.

Таблица 3. Временное обобществление - обновление кэша.

Кэшированный TCB	***Текущий TCB***	***Когда?***	***Новый кэшированный TCB***
old_MMS_S	curr_MMS_S	OPEN	curr_MMS_S
old_MMS_R	curr_MMS_R	OPEN	curr_MMS_R
old_sendMSS	curr_sendMSS	MSSopt	curr_sendMSS
old_PMTU	curr_PMTU	PMTUD/PLPMTUD[1]	curr_PMTU
old_RTT	curr_RTT	CLOSE	merge(curr,old)
old_RTTVAR	curr_RTTVAR	CLOSE	merge(curr,old)
old_option	curr_option	ESTAB	(зависит от опции)
old_ssthresh	curr_ssthresh	CLOSE	merge(curr,old)
old_sendcwnd	curr_sendcwnd	CLOSE	merge(curr,old)

Функция merge() объединяет текущее и прежнее (старое) значение и может различаться для каждого параметра в кэше. Документ не задаёт конкретную функцию. Например, может использоваться усреднение в окне (среднее из N прошлых значений для некоторого N) или разложение вида new = (1-alpha)*old + alpha*new, где alpha имеет значение из диапазона [0..1]).

В таблице 4 приведён обзор зависящей от опции информации, которая может обобщестляться аналогичным способом. значение TFO cookie поддерживается, пока клиент не запросит явное обновление как отдельное событие.

Таблица 4. Временное обобществление - обновление опций.

Кэшированное	***Текущее***	***Когда?***	***Новое кэшированное***
old_TFO_cookie	old_TFO_cookie	ESTAB	old_TFO_cookie
old_TFO_failure	old_TFO_failure	ESTAB	old_TFO_failure

6.3. Обсуждение

Как уже отмечалось, особой пользу от кэширования MMS_S и MMS_R нет, поскольку их сообщает локальный стек IP. Кэширование sendMSS и PMTU тривиально - сообщаемые значения кэшируются (PMTU на уровне IP) и спользуется самое свежее значение. Кэш обновляется при получении опции MSS в SYN или после PMTUD (т. е при получении сообщения ICMPv4 Fragmentation Needed [RFC1191] или ICMPv6 Packet Too Big [RFC8201], а также эквивалентных производных сообщений, например, от PLPMTUD [RFC4821]),поэтому в кэше всегда хранится самое свежее значение для соединения. Для sendMSS обращение к кэшу происходит только при организации соединения, а в других случаях кэш не обновляется, поэтому опции MSS не влияют на текущие соединения. Принятое по умолчанию значение sendMSS никогда не сохраняется и кэш одновляют лишь сообщеннные значения MSS, поэтому для снижения sendMSS требуется явное переопределение. Кэшированное значение sendMSS влияет только на данные в сегментах SYN, т. е. при организации соединения клиентом или одновременном открытии, MSS для остальных сегментов ограничивается значением, обновлённым из SYN.

Значения RTT обновляются по старым и новым значениям, как указано в параграфе 6.2. Для динамической оценки RTT нужны последовательные измерения RTT. К результате кэшированное значение RTT (и вариации) является средним между предшествующим значением и содержимым активного в настоящее время TCB для данного хоста, когда TCB закрывается. Значения RTT обновляются при закрытии соединения. Метод слияния старых и новых значений должен пытаться снизить переходные эффекты новых соединений.

Обновления RTT, RTTVAR и ssthresh основываются лишь на имеющихся сведениях (на старых значениях). Если старых значений нет, кэшируются текущие значения.

[1]PMTU кэшируется на уровне IP [RFC1191] [RFC4821].
[2]Некоторые значения не кэшируются при локальном расчёте (MMS_R) или указании в самом соединении (MMS_S в SYN).

Опции TCP копируются или объединяются в зависимости от деталей каждой опции. Например, состояние TFO обновляется при организации соединения и считывается перед организацией нового соединения.

В разделах 8 и 9 обсуждаются вопросы совместимости и последствия совместного использования укаазнных выше сведений, в разделе 10 представлен обзор известных реализаций.

Большинство кэшированных значений TCB обновляется при закрытии соединения. Исключениями являются MMS_R и MMS_S, которые ссобщает IP [RFC1122], PMTU, обновляемое после Path MTU Discovery и также приходящее от IP [RFC1191] [RFC4821] [RFC8201], и sendMSS, которое обновляется при получении опции MSS в заголовке TCP SYN.

Совместное использование сведений sendMSS влияет лишь на данные в SYN следующего соединения, поскольку информация sendMSS обычно включается в большинство сегментов TCP SYN. Кэширование PMTU может повысить эффективность PMTUD, но может привести к возникновению «черной дыры», пока ошибка не будет исправлена. Кэширование MMS_R и MMS_S может оказывать лишь незначительное прямое влияние, поскольку эти значения все равно сообщает локальный стек IP.

Способ обоществления состояния, связанного с другими опциями TCP, зависит от каждой опции. Например, состояние TFO включает TCP Fast Open cookie [RFC7413], а при отказе TFO - негативный отклик TCP Fast Open. В RFC 7413 сказано:

> Клиент **должен** кэшировать негативные отклики сервера для предотвращения возможных отказов при соединениях. Негативные отклики включают отсутствие подтверждения от сервера данных в пакете SYN, сообщения ICMP об ошибках и (наиболее важно) полное отсутствие отклика SYN-ACK от сервера, т. е. тайм-аут в соединении.

TFOinfo кэшируется при организации соединения.

Состояние, связанное с другими опциями TCP, может кэшироваться не так легко. Например, отказ или успех TCP-AO [RFC5925] между парой хостов для одного порта получателя SYN может быть полезно кэшировать. Успех или отказ TCP-AO для других портов получателя SYN той же пары хостов не будет полезен для кэширования,поскольку параметры защиты TCP-AO могут меняться в зависимости от службы.

7. Обобществление в ансамбле

Обобществление кэшированных данных TCB несколькими одновременными соединениями требует внимания к агрегированной природе некоторых обобществлённых состояний. Например, хотя значения MSS и RTT можно обобществлять путём копирования, простое копирование окна перегрузок и ssthresh может оказаться нецелесообразным и новые значения могут быть функцией (f) от совокупных значения и числа соединений (N).

7.1. Инициализация нового TCB

Блоки TCB для новых соединений можно инициализировать на основе контекста параллельных соединений, как показано в таблице 5.

Таблица 5. Обобществление в ансамбле - инициализация TCB.

Кэшированный TCB	***Новый TCB***
old_MMS_S	old_MMS_S
old_MMS_R	old_MMS_R
old_sendMSS	old_sendMSS
old_PMTU	old_PMTU[1]
old_RTT	old_RTT
old_RTTVAR	old_RTTVAR
sum(old_ssthresh)	f(sum(old_ssthresh), N)
sum(old_sendcwnd)	f(sum(old_sendcwnd), N)
old_option	(зависит от опции)

В таблице 5 сумма кэшированных значений sum() определяется для всех активных соединений, поскольку параметр является агрегатным. Функция f() обновляет сумму на основе значений нового соединения, представленного как N.

В таблице 6 приведён обзор зависящей от опции информации, которая может обобщестляться аналогичным способом. Значение TFO cookie поддерживается, пока клиент не запросит явное обновление как отдельное событие.

Таблица 6. Обобществление в ансамбле - инициализация опций.

Кэшированное	***Новое***
old_TFO_cookie	old_TFO_cookie
old_TFO_failure	old_TFO_failure

7.2. Обновление кэша TCB

При работе соединения кэш TCB может обновляться по изменениям параллельных соединений и их TCB, как показано в таблице 7.

Таблица 7. Обобществление в ансамбле - обновление кэша.

Кэшированный TCB	***Текущий TCB***	***Когда?***	***Новый кэшированный TCB***
old_MMS_S	curr_MMS_S	OPEN	curr_MMS_S
old_MMS_R	curr_MMS_R	OPEN	curr_MMS_R
old_sendMSS	curr_sendMSS	MSSopt	curr_sendMSS
old_PMTU	curr_PMTU	PMTUD/PLPMTUD[1]	curr_PMTU
old_RTT	curr_RTT	update	rtt_update(old, curr)
old_RTTVAR	curr_RTTVAR	update	rtt_update(old, curr)
old_ssthresh	curr_ssthresh	update	корректировка суммы при необходимости
old_sendcwnd	curr_sendcwnd	update	корректировка суммы при необходимости
old_option	curr_option	(зависимость)	(зависит от опции)

[1]PMTU кэшируется на уровне IP [RFC1191] [RFC4821].

В таблице 7 функция rtt_update() служит для комбинирования старых и новых значений, например, в форме среднего по окну или экспоненциально снижающегося среднего значения.

В таблице 8 приведена сводна зависящих от опции сведений, которые можно обобществить аналогичным способом.

Таблица 8. Обобществление в ансамбле - обновление опций.

Кэшированное	***Текущее***	***Когда?***	***Новое кэшированное***
old_TFO_cookie	old_TFO_cookie	ESTAB	old_TFO_cookie
old_TFO_failure	old_TFO_failure	ESTAB	old_TFO_failure

7.3. Обсуждение

Для обобществления в ансамбле сведения TCB следует кэшировать как можно раньше, иногда до закрытия соединения. В противном случае создание множества одновременных соединений может не привести к обобществлению данных TCB, если перед созданием соединения не было закрыто другое соединение. Объем работы, связанной с обновлением агрегированных средних значений, следует минимизировать так, чтобы результат был эквивалентен измерению всех значений в рамках одного соединения. Функция rtt_update в таблице 7 указывает эту операцию, которая выполняется всякий раз, когда значение RTT будет обновляться в отдельном соединении TCP. В результате кэш содержит обобществленные переменные RTT, которые больше не нужно хранить в TCB.

Агрегирование размера окна перегрузки и ssthresh при одновременных соединениях сложнее. При наличии ансамбля соединений нужно рашить, как эти переменные будут обобществляться в нем, чтобы вывести начальные значения для новых TCB.

В разделах 8 и 9 рассматриваются вопросы совместимости и влияние обобществления указанных выше сведений.

Имеется несколько способов инициализации окна перегрузки в новом TCB среди ансамбля текущих соединений с хостом. Современные реализации TCP инициализируют его 4 сегментами в соответствии со стандартом [RFC3390] и 10 - в экспериментах [RFC6928]. Эти подходы предполагают, что новым соединениям следует быть как можно более консервативными. Описанный в [Ba12] алгоритм корректирует начальное значение cwnd в зависимости от cwnd имеющихся соединений. Можно также использовать механизмы обобществления в долгосрочных интервалах для автоматической адаптации начального окна TCP, как описано в Приложении C.

8. Проблемы при совместном использовании сведений TCB

В этом разделе рассматриваются различные проблемы, которые могут возникать при обобществлении данных TCB. Для окна перегрузки и текущего окна начальные значения, определяемые взаимозависимостью TCB, могут не соответствовать долговременному поведению агрегата одновременных соединений между теми же конечными точками. Если при традиционном контроле перегрузок TCP окно перегрузки одного имеющегося соединения сходится к 40 сегментам, два недавно добавившихся соединения будут иметь начальные окна в 10 сегментов [RFC6928]A окно имеющегося соединения не будет сокращаться, чтобы воспринять дополнительную нагрузку. В результате три соединения могут мешать друг другу. Один из примеров этого можно видеть на узкополосных каналах с высокой задержкой, где одновременные соединения, поддерживающие трафик Web, могут конфликтовать из-за слишком большого начального окна, даже если это 1 сегмент.

Авторы работы [Hu12] рекомендуют кэшировать ssthresh для временного обобществления толь при длительных потоках. Некоторые исследования показываю, что обобществление ssthresh для коротких потоков может снижать производительность отдельных соединений [Hu12] [Du16], хотя это может повышать совокупную производительность.

8.1. Путь прохождения через сеть

TCP иногда используется в ситуациях, когда пакеты между одной парой хостов не всегда идут по одному пути, например, при использовании в маршрутизации параметров, зависящих от соединения (например, для распределения нагрузки). Маршрутизация по нескольким путям, основанная на транспортных заголовках, такая как ECMP и группы агрегирования каналов (Link Aggregation Group или LAG) [RFC7424] может не приводить к повторяющемуся выбору пути, когда сегменты TCP инкапсулируются, шифруются или изменяются, например, в туннеле виртуальной частной сети (Virtual Private Network или VPN), использующем фирменную (нестандартную) инкапсуляцию. Такие подходы не могут быть детерминированными при шифровании заголовков TCP, например, в случае применения инкапсуляции защищённых данных IPsec (Encapsulating Security Payload или ESP), хотя взаимозависимость TCB для всего набора с одинаковыми IP-адресами конечных точек должна работать без проблем при зашифрованных заголовках TCP. Можно применить меры для повышения вероятности использования соединениями одного пути, например, присваивать им одну метку потока IPv6 [RFC6437]. Взаимозависимость TCB можно распространить на наборы пар адресов IP, использующих одинаковые условия для сетевого пути, например, при размещении хостов в одной ЛВС (см. раздел 9).

Прохождение по одному пути не имеет значения для информации, связанной с хостом (например, rwnd), состояния опций TCP (например, TFOinfo) или сведений, уже кэшированных по хостам (например, path MTU). При обобществении данных TCB между различными портами получателя SYN, связанные с путём сведения могут быть некорректными, однако влияние таких ошибок снижается, если (как отмечено здесь) обобществление TCB влияет лишь на временное событие организации соединения или сведения TCB сообща применяюичся лишь соединениями с одним портом получателя SYN.

При временном обобществлении сведения TCB могут стать недействительными через некоторое время, указывая, что путь остался прежним, однако его свойства изменились. Поскольку это похоже на случай с простоем соединения, механизмы, применяемые в случае бездействия соединений TCP (например, [RFC7661]) могут применяться и для управления кэшем TCB, особенно при использовании TCP Fast Open [RFC7413].

8.2. Зависимость состояний

Могут быть дополнительные соображения относительно способа перераспределения взаимозависимостями TCB откликов о перегрузке между текущими соединениями. Например, может быть целесообразно рассмотреть влияние соединения в режиме Fast Recovery [RFC5681] или ином необычно режиме обратной связи, которые могут препятствовать или видять на описанные здесь расчёты.

8.3. Проблемы при обобществлении по адресу IP

Обобществление данных TCB между соединениями TCP одного хоста, идентифицируемого адресом IP, может оказаться ошибочным, если IP-адрес назначен новому хосту (например, при использовании провайдером IP address для подавления нежелательных серверов). Это может быть ошибкой при использовании трансляции сетевых адресов (Network Address Translation или NAT) [RFC2663], адресов и портов (Network Address and Port Translation или NAPT) [RFC2663] и других механизмов совместного использования адресов IP. В IPv6 такие механизмы применяются реже. Можно рассмотреть другие методы идентификации хоста для повышения вероятности коорректного обобществления данных TCB. Кроме того, некоторые сведения TCB относятся к доминирующим свойствам пути, а не к конкретному хосту - адреса IP могут различаться, но соответствующая часть пути может совпадать.

9. Последствия

Имеется несколько последствий встраивания взаимозависимости TCB в реализации TCP. Это может снижать потребность в мультиплексировании на прикладном уровне для повышения производительности [RFC7231]. Такие протоколы, как HTTP/2 [RFC7540], позволяют избежать расходов, связанных с повторной организацией соединений, сериализуя и мультиплексируя набор соединений хоста через одно соединение TCP. Это позволяет избежать на уровне соединений согласования TCP OPEN, а также повторного расчёта MSS, RTT и размера окна перегрузки. Избегая медленного перезапуска замедленного старта (slow-start restart), можно оптимизировать производительность [Hu01]. Взаимозависимость TCB может обеспечит такое предотвращение для мультиплексирования без необходимости поддержки мультиплексирования на уровне приложения.

Как и в исходной версии этого документа [RFC2140], подход к взаимозависимости TCB сосредоточен на обобществлении набора TCB путём обновления состояния TCB для сокращения влияния переходных процессов а начале соединения, при его завершении или ином существенном изменении состояния. Были предложены другие механизмы непрерывного обмена информацией между всеми текущими взаимодействиями (включая протоколы без организации явных соединений) и обновления состояния перегрузки по любым событиям, связанным с перегрузкой (например, тайм-аут, подтверждение потери и т. п.) [RFC3124]. Будучи связанным исключительно с переходными процессами, предложенный здесь подход, с большей вероятностью будет демонстрировать поведение в установившемся состоянии как у неизменяемых независимых соединений TCP.

9.1. Уровни

Взаимозависимость TCB выталкивая часть реализации TCP с обычного транспортного уровня (модель ISO) на сетевой. Это означает, что некоторые компоненты состояния относятся, по сути, к парам хостов и могут относиться к путям, заданным лишь такой парой хостов. Транспортные протоколы обычно управляют ассоциациями между парой хостов (потоки), а сетевые протоколы - ассоциациями между парами хостов и путями (маршрутизация). Время кругового обхода, MSS и сведения о перегрузке могут быть более подходящими для обработки на сетевом уровне, агрегироваться для одновременных соединений и обобществляться экземплярами соединений [RFC3124].

В более ранней версии обобществления RTT предлагалась реализация состояния RTT на уровне IP, а не TCP. В наблюдениях описано обобществление состояния соединениями TCP, позволяющее избежать некоторых сложностей решения на уровне IP. Одной из проблем решения IP является определение соответствия между обменами на основе лишь данных из заголловков IP при необходимости найти такое соответствие для расчёта RTT. Поскольку при обобществлении TCB расчёт RTT происходит внутри уровня TCP с использованием данных заголовка TCP, его можно реализовать более непосредственно и просто, нежели на уровне IP. Это тот случай, когда информацию следует рассчитывать на транспортном уровне, а обобществлять на сетевом.

9.2. Другие возможности

Попарные ассоциации хостов не являются переделом возможностей описанных методов. Вполне возможно аналогичное обобществление TCB между хостами в подсети или в кластере, поскольку преобладаюим типом будут соединения между подсетями, а не между хостами. Кроме того, взаимозависимость TCB может работать в любом протоколе с поддержкой состояния перегрузок, включая SCTP [RFC4960], DCCP [RFC4340], а также отдельные субпотоки Multipath TCP [RFC8684].

Между параллельными соединениями может обобществляться и другая информация. Например, зная о неудачной попытке соединения расширить окно, другое соединение может отказаться на некоторое время от попыток сделать то же самое. Идея заключается в том, что существующие реализации TCP учитывают поведение всех одновременных соединений, включая относящиеся к одному хосту или подсети. Одна из возможных оптимизаций заключается в том, чтобы сделать неявные обратные связи явными через расширение сведений, связанных с IP-адресом и реализацией TCP каждой конечной точки, а не состоянием TCB на уровне соединения.

В этом документе основное внимание уделено обобществлению сведений TCB при инициализации соединения. С момента публикации RFC 2140 было предложено множество подходов, пытающихся координировать текущие состояния одновременных соединений как в TCP, так и в других протоколах с реакцией на перегрузку. Обзор этих подходов приведён в [Is18]. Эти подходы сложнее в реализации и их сравнение с эквивалентностью установившихся состояний TCP может оказаться более сложным, иногда немеренно (т. е. они ингда стремяться обеспечить иной вариант «беспристрастности», нежели в TCP).

10. Имеющиеся реализации

Наблюдение о зависимости некоторых состояний TCB от пары хостов, а не пары предложений, не является новым и представляет обычное инженерное решание в многоуровневых реализациях протоколов. В T/TCP [RFC1644] (признан устаревшим) было впервые предложено использовать кэш для поддержки состояний TCB (см. Приложение A).

В таблице 9 показаны состояния реализации временного обобществления TCB в Windows на декабрь 2020 г., вариантах Apple (macOS, iOS, iPadOS, tvOS, watchOS) на явнварь 2021 г., ядре Linux версии 5.10.3 и FreeBSD 12. Обобществление для ансамблей ещё не реализовано.

Таблица 9. Известные состояния реализаций.

Данные TCB	*Статус*
old_MMS_S	Не обобществляется
old_MMS_R	Не обобществляется
old_sendMSS	Кэшируется и обобществляется в Apple, Linux (MSS)
old_PMTU	Кэшируется и обобществляется в Apple, FreeBSD, Windows (PMTU)
old_RTT	Кэшируется и обобществляется в Apple, FreeBSD, Linux, Windows
old_RTTVAR	Кэшируется и обобществляется в Apple, FreeBSD, Windows
old_TFOinfo	Кэшируется и обобществляется в Apple, Linux, Windows
old_sendcwnd	Не обобществляется
old_ssthresh	Кэшируется и обобществляется в Apple, FreeBSD, Linux[1]
TFO failure	Кэшируется и обобществляется в Apple

Apple в таблице 9 указывает операционные системы macOS (настольные и переносные ПК), iOS (телефоны), iPadOS (планшеты), tvOS (видео-проигрыватели), watchOS (часы), в которых применяется один стек протоколов Internet.

11. Отличия от RFC 2140

Этот документ обновляет описание обобществления TCB в RFC 2140 и его влияние на состояния имеющихся и новых соединений, полностью заменяя [RFC2140]. Разъяснены описания и термины, расширен механизм с учётом влияния новых протоколов и механизмов, включая Multipath TCP, Fast Open, PLPMTUD, NAT, TCP Authentication Option.

В подробном описании влияния на состояние TCB параметры TCB рассмотрены более конкретно. Разделены способы использовани MSS для приёма и передачи, ихи отличия от sendMSS, добавлены параметры Path MTU и ssthresh, а также рассмотрено влияние на состояние, связанное с опциями TCP. Добавлены сведения о проблемах совместимости и обзор реализаций. Связь этой работы с T/TCP перенесена в Приложение A (история обобществления TCB), отражая признание документа устаревшим. Добавлено Приложение C, где обсуждается возможное использование временного обобществления в долгосрочном масштабе для автомтическогй адаптации начального окна TCP, позволяющего обойтись без переиодического пересмотра значения глобального параметра. Кроме того, в документе обновлён и значительно расширен список литературы.

12. Вопросы безопасности

Представленные результаты не имеют дополнительных последствий в случаях прямых атак на отдельные соединения (разрыв соединений или внедрение данных). Отдельные соединения, независимо от использования обобществления, могут быть подвержены атакам с отказом в обслуживании, снижающим производительность или делающим соединения полностью неработоспособными при отсутствии должной защиты.

Обобществление TCB может приводить к дополнительным DoS-атакам, влияющим на производительность других соединений путём загрязнения обобществленных сведений. Это может происходить в любом наборе соединений с обобществлением TCB, соединений одного хоста или соединений между хостами при реализации обобществления внутри подсети (см. раздел 9). Некоторые обобществляемые параметры TCB используются лишь при создании новых TCB, а другие применяются всеми действующими соединениями. Новые соединения могут добавляться в имеющийся набор, например, для оптимизации окна передачи в соединениях с одним хостом. Значение PMTU задано как обобществляемое на уровне IP и уже подверженное таким атакам.

Опции в SYN от клиента подменить проще, чем в полном двухстороннем соединении. Поэтому такие значения могут безопасно включаться в обобществленные лишь по завершении трехэтапного согласования.

Атаки на параметры, применяемые только для инициализации, воздействуют на производительность соединения TCP лишь временно. Для краткосрочных соединений последствия могут быть близки к результатам DoS-атак. Например, если прилоюение изменяет свой блок TCB с указанием ложного или малого размера окна, производительность последующих соединений будет сокращаться, пока размер окна не возрастёт должным образом.

При обобществлении TCB повторно используется и смешивается информация из прошлых и текущих соединений. Хотя неоднократное использование информации может открывать возможность идентифицировать хосты по их отпечаткам (fingerprint) смешивание сокращает такие возможности. Не было замечено никаких подтверждений использования таких отпечатков и в настоящее время это считается безопасным. Сведения о производительности соединений TCP не считаются конфиденциальными.

13. Взаимодействие с IANA

Этот документ не запрашивает действий IANA.

14. Литература

14.1. Нормативные документы

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://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, <https://www.rfc-editor.org/info/rfc1122>.

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

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

[1]Для FreeBSD новое значение ssthresh является средним между curr_ssthresh и текущим значением, для Linux расчёт в большинстве случаев зависит от состояния и max(curr_cwnd/2, old_ssthresh).

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

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

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

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <https://www.rfc-editor.org/info/rfc7413>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

14.2. Дополнительная литература

[Al10] Allman, M., "Initial Congestion Window Specification", Work in Progress, Internet-Draft, draft-allman-tcpm-bump-initcwnd-00, 15 November 2010, <https://datatracker.ietf.org/doc/html/draft-allman-tcpm-bump-initcwnd-00>.

[Ba12] Barik, R., Welzl, M., Ferlin, S., and O. Alay, "LISA: A linked slow-start algorithm for MPTCP", IEEE ICC, DOI 10.1109/ICC.2016.7510786, May 2016, <https://doi.org/10.1109/ICC.2016.7510786>.

[Ba20] Bagnulo, M. and B. Briscoe, "ECN++: Adding Explicit Congestion Notification (ECN) to TCP Control Packets", Work in Progress, Internet-Draft, draft-ietf-tcpm-generalized-ecn-07, 16 February 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-tcpm-generalized-ecn-07>.

[Be94] Berners-Lee, T., Cailliau, C., Luotonen, A., Nielsen, H., and A. Secret, "The World-Wide Web", Communications of the ACM V37, pp. 76-82, DOI 10.1145/179606.179671, August 1994, <https://doi.org/10.1145/179606.179671>.

[Br02] Brownlee, N. and KC. Claffy, "Understanding Internet traffic streams: dragonflies and tortoises", IEEE Communications Magazine, pp. 110-117, DOI 10.1109/MCOM.2002.1039865, 2002, <https://doi.org/10.1109/MCOM.2002.1039865>.

[Br94] Braden, B., "T/TCP -- Transaction TCP: Source Changes for Sun OS 4.1.3", USC/ISI Release 1.0, September 1994.

[Co91] Comer, D. and D. Stevens, "Internetworking with TCP/IP", ISBN 10: 0134685059, ISBN 13: 9780134685052, 1991.

[Du16] Dukkipati, N., Cheng, Y., and A. Vahdat, "Research Impacting the Practice of Congestion Control", Computer Communication Review, The ACM SIGCOMM newsletter, July 2016.

[FreeBSD] FreeBSD, "The FreeBSD Project", <https://www.freebsd.org/>.

[Hu01] Hughes, A., Touch, J., and J. Heidemann, "Issues in TCP Slow-Start Restart After Idle", Work in Progress, Internet-Draft, draft-hughes-restart-00, December 2001, <https://datatracker.ietf.org/doc/html/draft-hughes-restart-00>.

[Hu12] Hurtig, P. and A. Brunstrom, "Enhanced metric caching for short TCP flows", IEEE International Conference on Communications, DOI 10.1109/ICC.2012.6364516, 2012, <https://doi.org/10.1109/ICC.2012.6364516>.

[IANA] IANA, "Transmission Control Protocol (TCP) Parameters", <https://www.iana.org/assignments/tcp-parameters>.

[Is18] Islam, S., Welzl, M., Hiorth, K., Hayes, D., Armitage, G., and S. Gjessing, "ctrlTCP: Reducing latency through coupled, heterogeneous multi-flow TCP congestion control", IEEE INFOCOM 2018 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), DOI 10.1109/INFCOMW.2018.8406887, April 2018, <https://doi.org/10.1109/INFCOMW.2018.8406887>.

[Ja88] Jacobson, V. and M. Karels, "Congestion Avoidance and Control", SIGCOMM Symposium proceedings on Communications architectures and protocols, November 1988.

[RFC1379] Braden, R., "Extending TCP for Transactions - Concepts", RFC 1379, DOI 10.17487/RFC1379, November 1992, <https://www.rfc-editor.org/info/rfc1379>.

[RFC1644] Braden, R., "T/TCP -- TCP Extensions for Transactions Functional Specification", RFC 1644, DOI 10.17487/RFC1644, July 1994, <https://www.rfc-editor.org/info/rfc1644>.

[RFC2001] Stevens, W., "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", RFC 2001, DOI 10.17487/RFC2001, January 1997, <https://www.rfc-editor.org/info/rfc2001>.

[RFC2140] Touch, J., "TCP Control Block Interdependence", RFC 2140, DOI 10.17487/RFC2140, April 1997, <https://www.rfc-editor.org/info/rfc2140>.

[RFC2414] Allman, M., Floyd, S., and C. Partridge, "Increasing TCP's Initial Window", RFC 2414, DOI 10.17487/RFC2414, September 1998, <https://www.rfc-editor.org/info/rfc2414>.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, DOI 10.17487/RFC2663, August 1999, <https://www.rfc-editor.org/info/rfc2663>.

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

[RFC3390] Allman, M., Floyd, S., and C. Partridge, "Increasing TCP's Initial Window", RFC 3390, DOI 10.17487/RFC3390, October 2002, <https://www.rfc-editor.org/info/rfc3390>.

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

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

[RFC5925] Touch, J., Mankin, A., and R. Bonica, "The TCP Authentication Option", RFC 5925, DOI 10.17487/RFC5925, June 2010, <https://www.rfc-editor.org/info/rfc5925>.

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

[RFC6691] Borman, D., "TCP Options and Maximum Segment Size (MSS)", RFC 6691, DOI 10.17487/RFC6691, July 2012, <https://www.rfc-editor.org/info/rfc6691>.

[RFC6928] Chu, J., Dukkipati, N., Cheng, Y., and M. Mathis, "Increasing TCP's Initial Window", RFC 6928, DOI 10.17487/RFC6928, April 2013, <https://www.rfc-editor.org/info/rfc6928>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

[RFC7323] Borman, D., Braden, B., Jacobson, V., and R. Scheffenegger, Ed., "TCP Extensions for High Performance", RFC 7323, DOI 10.17487/RFC7323, September 2014, <https://www.rfc-editor.org/info/rfc7323>.

[RFC7424] Krishnan, R., Yong, L., Ghanwani, A., So, N., and B. Khasnabish, "Mechanisms for Optimizing Link Aggregation Group (LAG) and Equal-Cost Multipath (ECMP) Component Link Utilization in Networks", RFC 7424, DOI 10.17487/RFC7424, January 2015, <https://www.rfc-editor.org/info/rfc7424>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/info/rfc7540>.

[RFC7661] Fairhurst, G., Sathiaseelan, A., and R. Secchi, "Updating TCP to Support Rate-Limited Traffic", RFC 7661, DOI 10.17487/RFC7661, October 2015, <https://www.rfc-editor.org/info/rfc7661>.

[RFC8684] Ford, A., Raiciu, C., Handley, M., Bonaventure, O., and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 8684, DOI 10.17487/RFC8684, March 2020, <https://www.rfc-editor.org/info/rfc8684>.

Приложение A. История обобществления TCB

В T/TCP предложено использовать кэш для поддержки использования данных TCB (временное обобществление), например, сглаженного значения RTT, вариаций RTT, порога предотвращения перегрузки и MSS [RFC1644]. Эти значения были добавлены к подсчёту соединений, используемому T/TCP для ускорения доставки данных до завершения трехэтапного согласования при OPEN. Цель заключалась о объединении компонентов TCB, когда они отражают одну ассоциацию пары хостов вместо искусственного разделения этих компонентов по соединениям.

По меньшей мере одна из реализаций T/TCP сохраняла MSS и агрегировала параметры RTT среди множества соединений, но не кэшировала сведения об окне перегрузки [Br94], как было изначально указано в [RFC1379]. Некоторые реализации T/TCP обновляли MSS сразу по получении опции заголовка TCP MSS [Br94], хотя это не было оговорено в концепциях и функциональной спецификации [RFC1379] [RFC1644]. В последующих реализациях T/TCP значения RTT обновлялись лишь после CLOSE, что не шло на пользу одновременным сессиям.

Временное обобществление кэшированных данных TCB было реализовано сначала в расширениях T/TCP Sun OS 4.1.3 [Br94] и варианте (port) FreeBSD [FreeBSD]. Как отмечено выше, кэшировались лишь MSS и параметры RTT, как исходно указано в [RFC1379]. В последующем обсуждении T/TCP было предложено включить в этот кэш параметры контроля перегрузок, например, параграф 3.1 в [RFC1644] советует инициализировать окно перегрузок прежним значением.

Приложение B. Обобществление и кэширование TCP Option

В дополнение к опциям, которые могут быть кэшированы или обобществлены, этот документ также указывает опции TCP [IANA] для которых небезопасно сохранять состояние. Список не является полномочным и исчерпывающим.

Устарели (сохранять состояние небезопасно)

- Echo
- Echo Reply
- Разрешены соединения с частичным упорядочением (Partial Order Connection Permitted)
- Профиль службы с частичным упорядочением (Partial Order Service Profile)
- CC
- CC.NEW
- CC.ECHO
- Запрос дополнительной контрольной суммы TCP (Alternate Checksum Request)
- Данные дополнительной контрольной суммы TCP (Alternate Checksum Data)

Не сохраняется состояние

- Конец списка опций (End of Option List или EOL)
- Нет операции (No-Operation или NOP)
- Масштабирование окна (Window Scale или WS)
- SACK
- Временные метки (TS)
- Подпись MD5 (Signature Option)
- Опция аутентификации TCP (TCP Authentication Option или TCP-AO)
- Эксперимент 1 в стиле RFC3692
- Эксперимент 2 в стиле RFC3692

Небезопасно сохранять состояние

Skeeter (известно, что обмен DH уязвим)
Bubba (известно, что обмен DH уязвим)
Контрольная сумма трейлера (Trailer Checksum Option)
Возможности SCPS
Селективные негативные подтверждения (Selective Negative Acknowledgements или S-NACK)
Границы записей
Позникло повреждение
SNAP
Фильтр сжатия TCP
Откли Quick-Start
Опция пользовательского тайм-аута (User Timeout Option или UTO)
Успех согласования Multipath TCP (MPTCP), об отказе см. ниже
Успех согласования TCP Fast Open (TFO), об отказе см. ниже

Safe but optional to keep state:

Отказ согласования Multipath TCP (MPTCP), об успехе см. выше
Максимальный размер сегмента (Maximum Segment Size или MSS)
Отказ согласования TCP Fast Open (TFO), об отказе см. ниже

Safe and necessary to keep state:

TCP Fast Open (TFO) Cookie (при успехе TFO в прошлом)

Приложение C. Автоматизация TCP IW в долгосрочном масштабе

C.1. Введение

Временное обоществление основано на предположении о том, что несколько одновременных соединений между одной парой хостов с некоторой вероятностью будут сталкиваться со схожими характеристиками сети. Сохранённые сведения со временем могут терять точность и нужны соответствующие меры учёта этого (см. параграф 8.1). Однако имеются случаи, когда осмысленно отслеживать эти значения в течение более длительных периодов, наблюдая свойства соединений TCP для постепенного учёта тенденций параметров TCP. В этом приложении рассматривается такой случай.

Алгоритм контроля перегрузок TCP использует начальный размер окна (initial window или IW) как при старте новых соединений, так в качестве верхнего предела при перезапуке после простоя [RFC5681] [RFC7661]. Это значение меняется с течение времени - изначально устанавливается окно размером в 1 максимальный сегмент (MSS), которое постепенно увеличивается до меньшего за значений 4 MSS или 4380 байтов [RFC3390] [RFC5681]. Для типичного соединения Internet с MTU 1500 байтов это позволяет использовать 3 сегмента по 1460 байтов.

Значение IW предложено в исходном описании контроля перегрузки TCP и документировано как стандарт в 1997 г. [RFC2001] [Ja88]. Значение было обновлено в 1998 (экспериментально) и опубликовано как Standards Track в 2002 г. [RFC2414] [RFC3390]. В 2013 г. оно было экспериментально повышено до 10 [RFC6928].

В этом приложении рассматривается, как протокол TCP может объективно определить, не является ли IW излишне большим, и как следует использовать обратную связь для автоматической корректировки IW в продолжительном соединении. Результату следует быть безопасным для внедрения и он, возможно, избавит от необходимости неоднократного пересмотра IW с течением времени.

Отметим, что этот механизм пытается сделать IW более адптивным с течением времени. Он может увеличивать IW сверх рекомендуемого для широкого внедоения значения, поэтому размер окна следует внимательно отслеживать.

C.2. Устройство

Значение TCP IW применяется статически уже более 20 лет и любому решению по динамической корректировке IW следует обеспечивать стабильное неразрушающее воздействие на производительность и сложность TCP. Для обеспечения беспристрастности значение IW следует делать близким для большинства машин в общедоступной сети Internet. Желательная также разработка самокорректирующего алгоритма, чтобы избегать значений IW, вызывающих проблемы в сети. В соответствии с этим предлагаются указанные ниже цели разработки.

- Практически полное отсутствие влияния на TCP при отсутствии потерь, т. е. не следует повышать сложность принятой по умолчанию обработки пакетов в обычных случаях.
- Адаптация к обратной связи из сети в долгосрочном масштабе с исключением значений, постоянно вызывающих проблемы в сети.
- Сокращение IW при наличии устойчивых потерь сегментов IW во множестве разных соединений.
- Увеличение IW в отсутствие устойчивых потерь сегментом IW во множестве разных соединений.
- Консервативные действия для сохранения IW при отсутствии достаточных сведений и более внимательное отношение к потере сегментов IW, нежели успешной доставке.

Предполагается, что в отсутствие друго контекста хороший алгоритм IW будет сходиться к одному значению, хотя это и не требуется. Кончная точка с дополнительными сведениями о контексте или развернутая в среде с ограничениями всегда может использовать другое значение. В частности, сведения из предыдущих соединений или набора соединений с походим путём могут служить в качестве контекста для таких решений (см. выше).

Если же данное значение IW постоянно ведёт к потерям в течение начального пика пакетов, оно явно не пригодно и может приводить к дополнительным потерям в одновременных соединениях. Это может происходить на сайтах, расположенных за очень медленными устройствами с малыми буферами, которые могут (но не обязательно) первым узлом пересылки (hop).

C.3. Предложенный алгоритм IW

Ниже приведено краткое описание предлагаемого алгоритма IW, основанного на указанных далее параметрах.

- MinIW = 3 MSS или 4380 байтов (в соответствии с [RFC3390]).
- MaxIW = 10 MSS (в соответствии с [RFC6928]).
- MulDecr = 0,5.
- AddIncr = 2 MSS.
- Threshold = 0,05.

Предполагается, что минимальное значение IW (MinIW) следует устанавливать в соответствии с текущим стандартом [RFC3390]. Для максимального IW (MaxIW) можно задать фиксированное значение (предполагается использовать экспериментальный документ [RFC6928], уже ставший фактическим стандартом) или основывать его на планировании, если доступны надёжные временные ссылки [AI10]. Здесь выбран вариант с фиксированным значением. Предлагается также использовать алгоритм адаптивного увеличения и эспоненциального снижения (Additive Increase Multiplicative Decrease или AIMD) с указанными параметрами.

Хотя приведённые выше параметры в той или иной мере спорны, их исходные значения не важны, за исключением того, что алгоритму AIMD и параметру MaxIW не следует выходить за пределы, рекомендуемые для других систем Internet (здесь выбран фактический, а не формальный стандарт). Текущие предложения, включая принятые по умолчанию операции, являются вырожденными вариантами приведённого ниже алгоритма для заданных параметров (в частности, MulDec = 1,0 и AddIncr = 0 MSS, отключающих автоматическую часть алгоритма).

Предлагаемый алгоритм указан ниже

1. При загрузке.

```
IW = MaxIW;    # предполагается, что это целое число байтов, кратное 2 MSS
               # (чётное для поддержки сжатия ACK)
```

2. При старте нового соединения

```
CWND = IW;
conncount++;
IWnotchecked = 1; # true
```

3. При обработке SYN-ACK в соединении наличие ECN (как указано для ECN++ в разделе 5 [Ba20] для TCP), считается индикацией слишком большого IW

```
if (IWnotchecked && (synackecn == 1)) {
    losscount++;
    IWnotchecked = 0; # больше не повторять
}
```

4. При повторе передачи в соединении проверяется порядковый номер (seqno) исходящего пакета (в байтах) для обнаружения потери сегмента IW

```
if (Retransmitting && IWnotchecked && ((seqno - ISN) < IW))) {
    losscount++;
    IWnotchecked = 0; # не повторять впредь при соблюдении условия
} else {
    IWnotchecked = 0; # выход за предел IW, остановка проверки
}
```

5. На каждую 1000 соединений выполняется отдельный процесс, не являющийся частью обработки данного соединения

```
if (conncount > 1000) {
    if (losscount/conncount > threshold) {
        # слишком много соединений с ошибками
        IW = IW * MulDecr;
    } else {
        IW = IW + AddIncr;
    }
}
```

В представленном виде алгоритм может давать ложные срабатывания при прохождении порядкового номера через максимум, например, код может увеличить losscount на этапе 4 при отсутствии потеррь или не увеличить при их наличии. Этого можно избежать, применяя контекст защиты от прохождения номер через максимум (Protection Against Wrapped Sequences или PAWS) [RFC7323] или расширяя внутреннее пространство порядковых номеров (как в TCP Authentication Option (TCP-AO) [RFC5925]). Можно допустить ложные срабатывания, считая из нечастыми и не оказывающими существенного влияния на алгоритм.

При реализации алгоритма нужно внести ряд дополнительных ограничений, чтобы по умолчанию принимались значения, соответствующие текущим стандартам Internet, обеспечивалось консервативное увеличесние значений и возврат при отсутствии положительных откликов (успех). Для этого рекомендуются указанные ниже ограничения.

- Автоматический алгоритм для IW **должен** инициализировать MaxIW значением не больше рекомендуемого в настоящее время по умолчанию для Internet при отсутствии других сведений о контексте. Таким образом, при слишком малом для принятия решения числе соединений или недостаточности иных сведений для увеличения IW параметр the MaxIW принимает текущее рекомендованное значение.
- Реализация **может** разрешить рост MaxIW сверх рекомендуемого по умолчанию значения для Internet, но не более, чем на 2 сегмента за календарный год. Таким образом, при наличии у конечной точки постоянной истории успешной передачи сегментов IW без потерь ей разрешается исследовать Internet для проверки аналогичной успешности использования больших значений IW. Такая проверка ограничена и требует доверенного источника точного времни, иначе значение MaxIW сохранится постоянным.

- Реализация **должна** корректировать IW на основе статистики не реже 1 раза на 1000 соединений.

  Конечная точка должна подобающим образом реагировать на потери сегментов IW.

- Реализация **должна** снижать IW не менее, чем на 1 MSS, когда снижение указано в интервале оценки.

  Конечной точке, обнаружившей потерю необходимо сократить IW хотя бы на 1 MSS, иначе она не будет участвовать в алгоритме автоматического реагирования.

- Реализация **должна** увеличивать значение не более, чем на 2 MSS за интервал оценки.

  Конечная точка, не сталкивающаяся с потерями IW должна инкрементально зондировать сеть.

- Реализации **следует** использовать для IW целые значения, кратные 2 MSS.

  Значение IW следует сохранять кратным сегментам 2 MSS для эффективного сжатия ACK без возникновения излишних тайм-аутов.

- Реализация **должна** снижать IW, если потери IW наблюдаются более, чем в 95% соединений.

  Это требуется для достаточн эффективной реакции реализации.

- Реализация **может** группировать значения и статистику IW для подмножества соединений. Для группировки **можно** использовать любую информацию о соединениях, кроме статистики потерь.

Существуют соединения TCP, которые могут не учитываться совсем, например, соединения через шлейфовые интерфейсы или внутри одной подсети с локальным интерфейсом (где контроль перегрузок иногда просто отключают). Сюдя можно отнести также соединения, прерванные до заполнения IW (в качестве отдельной проверки при закрытии).

Период, в течение которого обновляется IW, должен быть долгим, например, месяц или 1000 соединений (что больше). Реализация может проверять IW один раз в месяц и просто не обновлять IW или очищать счётчики соединений в месяцы, когда число соединений слишком мало.

C.4. Обсуждение

Существует множество параметров указанного выше алгоритма, которые соответствуют приведённым требованиям. Это сделано для того, чтобы можно было варьировать конфигурации и реализации, гарантируя, что все такие алгоритмы обеспечивают нужную реакцию и безопасны.

В алгоритме рассматриваются сегменты, поскольку они являются основой для большинства реализаций TCP. Возможно, полезно было бы пересмотреть спецификации, чтобы разрешить контроль перегрузки по подсчёту байтов при наличии достаточного опыта.

Алгоритм проверяет потери IW только для первого IW после старта, а затем проверка потерь IW не применяется (например, при использовании замедленного старта при перезапуске).

- Реализация **может** обнаружить потери IW при замедленном старте в процессе перезапуска в дополнение к потерям в первом IW соединения. В этом случае реализация **должна** считать каждый перезапуск «соединением» для подсчёта соединений и переиодической проверки значения IW.

При некоторых вариантах нарушения порядка сегментов могут возникать ложные срабатывания, вызывающие например, ложный повтор передачи при отсутствии потерь. Предполагается, что таким события не будут достаточно частыми, чтобы влиять на алгоритм и его заключения.

Этот механизм не требует дополнительного состояния на соединение, что в настоящее время распространено в некоторых реализациях и полезно по иным причинам (например, ISN в TCP-AO [RFC5925]). Механизм из этого приложения выигрывает от сохранения состояния при перезагрузке, что полезно и для других механизмов обобществления состояния (например, TCP Control Block Sharing, как описано выше).

Окно приёма (rwnd) не учитывается в расчёте, его размер определяется ресурсами получателя и обеспечивает пространство для восстановления порядка сегментов. Значение rwnd не участвует и в контроле перегрузки, который является основным способом управления IW в этом приложении.

C.5. Наблюдения

IW может не сходиться к одному глобальному значению и даже не сходиться совсем, осциллируя между несколькими значениями MSS при периодическом зондировании Internet для больших IW с отказами. Оба эти свойства согласуются с поведением TCP в каждом отдельном соединении.

Этот механизм предполагает, что потери в IW обусловлены размером IW. Постоянные ошибки с отбрасыванием пакетов по иным причинам, например, из-за ошибок ОС, могут вызывать ложные срабатывания. Это согласуется с базовым допущением TCP о потерях, вызываемых перегрузкой и требующих сокращения окна (backoff). Алгоритм считает IW новых соединений системов отката в длительных интервалах времени.

Благодарности

Авторы благодарны Praveen Balasubramanian за сведения об обобществлении TCB в Windows, Christoph Paasch - за сведения для Apple OS, Yuchung Cheng, Lars Eggert, Ilpo Jarvinen и Michael Scharf за комментарии к ранним версиям документа. Спасибо также членам рабочей группы TCPM WG. Предварительные выпуски этой работы финансировались в рамках совместного исследовательского проекта University of Oslo и Huawei Technologies Co., Ltd., а также частично поддерживались USC/ISI Postel Center.

Адреса авторов

Joe Touch
Manhattan Beach, CA 90266

United States of America
Phone: +1 (310) 560-0334
Email: touch@strayalpha.com

Michael Welzl
University of Oslo
PO Box 1080 Blindern
N-0316 Oslo
Norway
Phone: +47 22 85 24 20
Email: michawe@ifi.uio.no

Safiqul Islam
University of Oslo
PO Box 1080 Blindern
Oslo N-0316
Norway
Phone: +47 22 84 08 37
Email: safiquli@ifi.uio.no

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru