Internet Engineering Task Force (IETF)
Request for Comments: 9406
Category: Standards Track
ISSN: 2070-1721

P. Balasubramanian
Confluent
Y. Huang
M. Olson
Microsoft

May 2023

HyStart++: Modified Slow Start for TCP

HyStart++ - изменённый алгоритм Slow Start для TCP

Аннотация

В этом документе описывается HyStart++ - простое изменение фазы замедленного старта для алгоритмов контроля перегрузки. Замедленный старт во многих случаях может приводить к превышению идеальной скорости передачи, вызывающему потерю пакетов и снижение производительности. В HyStart++ используется увеличение времени кругового обхода (round-trip delay) в качестве эвристических данных для нахождения точки выхода до возможного превышения скорости. Это также смягчает возникновение вариаций задержки (jitter), приводящих к преждевременному выходу из замедленного старта.

Статус документа

Документ относится к категории Internet Standards Track.

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 стандартах Internet 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/rfc9406.

Оглавление

1. Введение

В [RFC5681] описан алгоритм замедленного старта для контроля перегрузок в протоколе TCP. Алгоритм замедленного старта применяется в случаях, когда окно перегрузок (congestion window или cwnd) становится меньше порога замедленного старта (slow start threshold или ssthresh). В процессе замедленного старта при отсутствии потерь пакетов протокол TCP экспоненциально увеличивает значение cwnd для определения пропускной способности сети (network capacity). Такой быстрый рост может приводить к тому, что скорость передачи станет выше идеальной и это приведёт к значительным потерям пакетов, которые не всегда можно восстановить.

HyStart++ строится на основе алгоритма Hybrid Start (HyStart), исходно описанного в [HyStart]. HyStart++ использует увеличение времени кругового обхода (round-trip delay) как сигнал для выхода из процедуры замедленного старта до того, как станут возможны потери в результате превышения скорости. Это один из двух алгоритмов, заданных в [HyStart] для поиска безопасной точки выхода из процедуры замедленного старта. После выхода из процедуры замедленного старта используется новая фаза консервативного замедленного старта (Conservative Slow Start или CSS)

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

для решения вопроса, не был ли этот выход преждевременным и возобновления замедленного старта при преждевременному выходе. Такое смягчение повышает производительность при наличии вариаций задержки. Механизм HyStart++ снижает потери пакетов и их повторную передачу, повышая производительность в лабораторных условиях и работающих системах.

Хотя документ описывает HyStart++ для протокола TCP, механизм можно использовать и с другими транспортными протоколами, применяющими замедленный старт, такими как QUIC [RFC9002] или SCTP[1] [RFC9260].

2. Уровни требований

Ключевые слова **необходимо** (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.

3. Определения

Чтобы помочь читателям, ниже приведены некоторые определения из [RFC5681].

SENDER MAXIMUM SEGMENT SIZE (SMSS) - максимальный размер сегмента для отправителя

Размер самого большого сегмента, который может быть передан отправителем. Это значение может определяться на основе максимального блока данных, передаваемого через сеть, алгоритма определения Path MTU [RFC1191, RFC4821], RMSS (см. следующее определение) и других факторов. Значение не включает размеры заголовков и опций TCP/IP.

RECEIVER MAXIMUM SEGMENT SIZE (RMSS) - максимальный размер сегмента для получателя

RMSS – размер максимального сегмента, который желает принимать получатель. Это значение задаётся в опции MSS, передаваемой хостом при организации соединения. Если опция MSS при организации соединения не была задана, используется значение 536 байтов [RFC1122]. Значение не включает размеры заголовков и опций TCP/IP.

RECEIVER WINDOW (rwnd) - размер окна принимающей стороны

Последнее анонсированное значение размера окна принимающей стороны.

CONGESTION WINDOW (cwnd) - размер окна насыщения (перегрузки)

Переменная состояния TCP, которая ограничивает количество данных, разрешённых протоколу для передачи. В любой момент для TCP **недопустима** передача данных с порядковыми номерами, превышающими значение суммы наибольшего из подтверждённых порядковых номеров и меньшего из двух значений cwnd и rwnd.

4. Алгоритм HyStart++

4.1. Обзор

В [HyStart] заданы два алгоритма (Delay Increase и Inter-Packet Arrival), которые работают параллельно для определения момента, когда скорость передачи достигает предела возможностей сети (capacity). На практике алгоритм Inter-Packet Arrival не работает достаточно хорошо и не способен обнаруживать перегрузку заранее прежде всего в результате сжатия ACK. Идея алгоритма Delay Increase состоит в поиске скачков времени кругового обхода (round-trip time или RTT), которые указывают заполнение буферов в узком месте сети (bottleneck).

В HyStart++ отправитель TCP использует стандартный замедленный старт и алгоритм увеличения задержки (Delay Increase) для инициирования выхода из процедуры замедленного старта. Но вместо незамедлительного перехода от механизма slow start к предотвращению перегрузки (congestion avoidance), отправитель в течение нескольких интервалов RTT в фазе консервативного замедленного старта (CSS) для решения вопроса о своевременности выхода из процедуры замедленного старта. В процессе CSS окно перегрузок увеличивается экспоненциально, подобно обычной процедуре замедленного старта, но с меньшим показателем, что ведёт к менее агрессивному росту. Если RTT снижается в процессе CSS, считается что скачок RTT не был вызван перегрузкой, связанной с ростом скорости передачи сверх идеальной, и для соединения возобновляется замедленный старт. Если рост RTT продолжается в процессе CSS, соединение переходит в режим предотвращения перегрузки.

4.2. Детали алгоритма

В приведённом ниже псевдокоде предел L служит для управления энергичностью (aggressiveness) увеличения cwnd в стандартном механизме замедленного старта и CSS. Хотя прибытие ACK может заново подтверждать доставку произвольного числа байтов, алгоритм HyStart++ ограничивает число байтов, применяемых для увеличения cwnd значением L*SMSS.

При инициализации для lastRoundMinRTT и currentRoundMinRTT устанавливаются бесконечные значения, currRTT указывает значение RTT, полученное из входящего ACK, и при инициализации также устанавливается бесконечным.

```
lastRoundMinRTT = infinity
currentRoundMinRTT = infinity
currRTT = infinity
```

HyStart++ определяет интервал (round) на основе порядковых номеров:

- определяется windowEnd как порядковый номер, инициализируемый значением SND.NXT;
- при подтверждении номера windowEnd (ACK) текущий интервал считается завершённым и снова устанавливается windowEnd = SND.NXT.

В начале каждого интервала при стандартном замедленном старте [RFC5681] и CSS инициализируются переменные, применяемые для расчёта минимального значения RTT в текущем интервале

```
lastRoundMinRTT = currentRoundMinRTT
currentRoundMinRTT = infinity
rttSampleCount = 0
```

[1]Stream Control Transmission Protocol - протокол управления потоковой передачей.

Для каждого прибывающего ACK при замедленном старте (N - число не подтверждённых ранее байтов, подтверждаемых данным ACK)

- обновляется cwnd

```
cwnd = cwnd + min(N, L * SMSS)
```

- сохраняется минимальное наблюдавшееся значение RTT

```
currentRoundMinRTT = min(currentRoundMinRTT, currRTT)
rttSampleCount += 1
```

Для интервалов, в которых получено хотя бы N_RTT_SAMPLE значений RTT и текущие значения currentRoundMinRTT и lastRoundMinRTT действительны, проверяется, вызывает ли рост задержки выход из процедуры замедленного старта

```
if ((rttSampleCount >= N_RTT_SAMPLE) AND
    (currentRoundMinRTT != infinity) AND
    (lastRoundMinRTT != infinity))
  RttThresh = max(MIN_RTT_THRESH,
    min(lastRoundMinRTT / MIN_RTT_DIVISOR, MAX_RTT_THRESH))
  if (currentRoundMinRTT >= (lastRoundMinRTT + RttThresh))
    cssBaselineMinRtt = currentRoundMinRTT
    выход из slow start и запуск CSS
```

Для каждого прибывающего ACK в CSS (N - число не подтверждённых ранее байтов, подтверждаемых данным ACK)

- обновляется cwnd

```
cwnd = cwnd + (min(N, L * SMSS) / CSS_GROWTH_DIVISOR)
```

- сохраняется минимальное наблюдавшееся значение RTT

```
currentRoundMinRTT = min(currentRoundMinRTT, currRTT)
rttSampleCount += 1
```

Для интервалов CSS , в которых получено хотя бы N_RTT_SAMPLE значений RTT, проверяется, не стало ли значение minRTT текущего интервала ниже базового (cssBaselineMinRtt), что указывает на ложный выход из процедуры замедленного старта

```
if (currentRoundMinRTT < cssBaselineMinRtt)
  cssBaselineMinRtt = infinity
  возобновление slow start, включая HyStart++
```

CSS продолжается не более CSS_ROUNDS интервалов. Если переход в CSS произошёл в середине интервала, неполный интервал учитывается.

По истечении CSS_ROUNDS интервалов запускается алгоритм предотвращения перегрузки установкой

```
ssthresh = cwnd
```

При наличии потерь или маркеров явного уведомления о перегрузке (Explicit Congestion Notification или ECN) во время стандартной процедуры замедленного старта или CSS запускается механизм предотвращения перегрузки установкой

```
ssthresh = cwnd
```

4.3. Настройка констант и другие соображения

Реализациям HyStart++ **рекомендуется** устанавливать приведённые ниже значения констант.

```
MIN_RTT_THRESH = 4 мсек
MAX_RTT_THRESH = 16 мсем
MIN_RTT_DIVISOR = 8
N_RTT_SAMPLE = 8
CSS_GROWTH_DIVISOR = 4
CSS_ROUNDS = 5
L = бесконечность в режиме с «прореживанием», L = 8 в ином случае
```

Эти константы были определены в лабораторных условиях и рабочих системах. Реализация **может** подстраивать эти значения в соответствии с характеристиками сети.

Чувствительность к росту задержки определяется константами MIN_RTT_THRESH и MAX_RTT_THRESH. Меньшие значения MIN_RTT_THRESH могут вызывать необоснованный выход из процедуры slow start, большие значения MAX_RTT_THRESH - препятствовать выходу из slow start до момента возникновения потерь на путях с большим RTT.

MIN_RTT_DIVISOR - это доля[1] RTT, применяемая для расчёта порога задержки. Меньшее значение повышает порог, снижая чувствительность к росту задержки.

Хотя от всех реализаций TCP **требуется** делать хотя бы 1 выборку RTT за каждый интервал, реализациям HyStart++ **рекомендуется** делать не меньше N_RTT_SAMPLE выборок RTT. Уменьшение N_RTT_SAMPLE будет снижать точность измерения RTT, а большие значения повышают точность за счёт роста издержек на обработку.

Для CSS_GROWTH_DIVISOR **должно** устанавливаться значение не меньше 2. Значение 1 приведёт к такому же поведению, как при обычном замедленном старте, значения больше 4 снижают энергичность алгоритма и могут снижать производительность.

Малые значения CSS_ROUNDS могут препятствовать обнаружению вариаций задержки, большие - ограничивать производительность.

«Прореживание» (pacing) пакетов [ASA00] является одним из возможных механизмов предотвращения значительных пиков трафика и связанного с этим ущерба. В реализации TCP с прореживанием следует устанавливать бесконечное значение L. Прореживание смягчает проблемы, связанные с пиками трафика и такая установка обеспечивает оптимальный рост cwnd в современной сети.

В реализациях TCP с прореживанием для смягчения влияния пиков трафика конечные значения L могут приводить к существенным проблемам производительности в высокоскоростных сетях по причине медленного роста cwnd. Для

[1]Результат деления. *Прим. перев.*

реализация TCP без прореживания значение L меньше 8 могут приводить к существенным проблемам производительности в высокоскоростных сетях по причине медленного роста cwnd, а значения больше 8 могут способствовать возникновению пиков трафика и соответствующему снижению производительности.

Реализациям **следует** использовать HyStart++ только в начальной процедуре slow start (когда ssthresh имеет начальное значение, произвольное в соответствии с [RFC5681]) и возвращаться к стандартной процедуре slow start в течение остального времени действия соединения. Это приемлемо, поскольку в последующих процедурах slow start используется обнаруженное значение ssthresh для выхода из процедуры замедленного старта и предотвращения проблемы превышения скорости. Реализация **может** использовать HyStart++ для увеличения restart window [RFC5681] после длительного бездействия.

В ограниченных приложениями сценариях, где объем находящихся в сети данных (in flight) может быть меньше произведения задержки на пропускную способность (bandwidth-delay product или BDP), снижая число выборок RTT, что может приводить к возврату в slow start. В таких случаях ожидается «колебание» между механизмами CSS и slow start, но такое поведение не будет приводить к преждевременному переходу в режим предотвращения перегрузки или превышению скорости по сравнению с обычным механизмом slow start.

5. Развёртывание и оценка производительности

Во время подготовки этого документа описанный здесь механизм HyStart++ был по умолчанию включён для всех соединений TCP в операционной системе Windows уже более 2 лет с отключённым прореживанием и L = 8.

В лабораторных измерениях с Windows TCP механизм HyStart++ показал улучшение пропускной способности, а также сокращение потери пакетов и повтора передачи по сравнению со стандартным механизмом slow start. Например, различные тесты на каналах 100 Мбит/с размером буфера, ограниченным произведением задержки на пропускную способность, механизм HyStart++ снижает объем повторно передаваемых данных на 50% (в байтах) и тайм-аут повтора (retransmission timeout или RTO) на 36%.

В тесте A/B, где сравнивалась реализация HyStart++ (на основе предварительной версии этого документа) со стандартным механизмом slow start в среде с большим числом устройств Windows из 52 миллиардов соединений TCP лишь 0,7% соединений переходили от 1 RTO к 0 RTO и ещё 0,7% - от 2 RTO к 1 RTO при использовании HyStart++. Этот тест не был сфокусирован на соединениях с отправкой большого объёма данных, где влияние может оказаться существенно сильнее. Планируется проведение дополнительных экспериментов для сбора данных.

6. Вопросы безопасности

HyStart++ улучшает механизм slow start и наследует соображения безопасности, отмеченные в [RFC5681].

Атакующий может вызвать преждевременный выход HyStart++ из процедуры замедленного старта и снизить производительность соединения TCP, например, путём отбрасывания пакетов данных или их подтверждения.

Атака ACK division, описанная в [SCWA99]Ю не влияет на HyStart++, поскольку расширение окна перегрузки для HyStart++ основано на числе вновь подтверждённых данных в каждом прибывающем ACK, а не на увеличении на конкретную константу при получении каждого ACK.

7. Взаимодействие с IANA

Этот документ не требует действий IANA.

8. Литература

8.1. Нормативные документы

[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>.

[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>.

[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>.

8.2. Дополнительная литература

[ASA00] Aggarwal, A., Savage, S., and T. Anderson, "Understanding the performance of TCP pacing", Proceedings IEEE INFOCOM 2000, DOI 10.1109/INFCOM.2000.832483, March 2000, <https://doi.org/10.1109/INFCOM.2000.832483>.

[HyStart] Ha, S. and I. Rhee, "Taming the elephants: New TCP slow start", Computer Networks vol. 55, no. 9, pp. 2092-2110, DOI 10.1016/j.comnet.2011.01.014, June 2011, <https://doi.org/10.1016/j.comnet.2011.01.014>.

[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>.

[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>.

[RFC9002] Iyengar, J., Ed. and I. Swett, Ed., "QUIC Loss Detection and Congestion Control", RFC 9002, DOI 10.17487/RFC9002, May 2021, <https://www.rfc-editor.org/info/rfc9002>.

[RFC9260] Stewart, R., Tüxen, M., and K. Nielsen, "Stream Control Transmission Protocol", RFC 9260, DOI 10.17487/RFC9260, June 2022, <https://www.rfc-editor.org/info/rfc9260>.

[SCWA99] Savage, S., Cardwell, N., Wetherall, D., and T. Anderson, "TCP congestion control with a misbehaving receiver", ACM SIGCOMM Computer Communication Review, vol. 29, issue 5, pp. 71-78, DOI 10.1145/505696.505704, October 1999, <https://doi.org/10.1145/505696.505704>.

Благодарности

В процессе обсуждения этой работы в почтовой конференции TCPM и на встречах рабочей группы были получены полезные комментарии и рецензии от (в алфавитном порядке фамилий) Mark Allman, Bob Briscoe, Neal Cardwell, Yuchung Cheng, Junho Choi, Martin Duke, Reese Enghardt, Christian Huitema, Ilpo Järvinen, Yoshifumi Nishida, Randall Stewart, Michael Tüxen.

Адреса авторов

Praveen Balasubramanian
Confluent
899 West Evelyn Ave
Mountain View, CA 94041
United States of America
Email: pravb.ietf@gmail.com

Yi Huang
Microsoft
One Microsoft Way
Redmond, WA 98052
United States of America
Phone: +1 425 703 0447
Email: huanyi@microsoft.com

Matt Olson
Microsoft
One Microsoft Way
Redmond, WA 98052
United States of America
Phone: +1 425 538 8598
Email: maolson@microsoft.com

Перевод на русский язык

Николай Малых

nmalykh@protokols.ru