Пояснения к пределам очередей и сбросу выходных данных для платформ ПО Cisco IOS

Содержание

Введение

Этот документ применим к платформам программного обеспечения Cisco IOS только, который обычно включает Cisco 7200VXR и Cisco ISR 3800, 2800, маршрутизаторы серии 1800, а также устаревшие Маршрутизаторы доступа Cisco включая 3700, 3600, 2600, и маршрутизаторы серии "1700".

Предварительные условия

Требования

Компания Cisco рекомендует предварительно ознакомиться со следующими предметами:

- Решения качества обслуживания Cisco IOS
- QoS — иерархическая структура очередей (HQF)

Используемые компоненты

Сведения, содержащиеся в этом документе, касаются следующих версий программного обеспечения:

- Для Pre-HQF: Маршрутизаторы, которые работают под управлением ПО Cisco IOS выпуска 12.3(26)

- Для HQF: Маршрутизаторы, которые работают под управлением ПО Cisco IOS выпуска 12.4(22)T

Сведения, представленные в этом документе, были получены от устройств, работающих в специальной лабораторной среде. Все устройства, описанные в этом документе, были запущены с чистой (стандартной) конфигурацией. В рабочей сети необходимо изучить потенциальное воздействие всех команд до их использования.

Условные обозначения

Дополнительные сведения об условных обозначениях см. в документе Условные обозначения технических терминов Cisco.

Взвешенная организация справедливых очередей на основе классов

В образах Pre-HQF IOS любой класс с командой bandwidth, как правило, имеет приоритет относительно классов без команд bandwidth или priority, основанных на весах класса Weight. Для понимания алгоритма планирования Class-Based Weighted Fair Queueing (CBWFQ) сначала необходимо ознакомиться с концепцией веса, который зависит от потока для очередей Flow Based Fair Queue и от класса для отдельных очередей Class Based Queue внутри справедливой взвешенной очереди Class Based Weighted Fair Queue.

Формула для определения веса для очереди, основанной на потоке, Flow Based Fair Queue:

```
32384 / (IP Prec + 1)
```

Определенные пользователем классы в основанной на классах взвешенной справедливой очереди получают соответствующие веса как функцию команды bandwidth, настроенной в классе как пропорция суммы всех классов bandwidth в очереди Class Based Queue. Точная формула является собственностью компании.

В образах HQF основанные на потоке справедливые очереди, настраиваемые как в определяемых пользователем классах, так и в классах по умолчанию с fair-queue, размещаются равноправно (а не по весам). Кроме того, в HQF расписание приоритезации очереди, основанной на классах, определяется планировщиком HQF, а не по формуле унаследованных весов классов.

Примечание: Этот раздел не предназначен, чтобы быть всесторонним анализом на поведенческом уровне для Класса Базирующиеся операции Организации очереди. Здесь представлены лишь краткие сведения в тех пределах, в каких CBWFQ имеет отношение к пониманию пределов очередей и сброса выходных данных.

Пояснения к пределам очередей и сбросу выходных данных

Определяемые пользователем классы, сконфигурированные с командой priority

Применимо для определяемых пользователем классов MQC, настроенных с любыми вариантами команды priority, включая priority, priority <kbps> и priority percent.

Поведение Pre-HQF

Технически, даже если нет интерфейса командной строки, который может определить ее, и если она не настраивается, скрытая очередь системы существует и используется всеми данными приоритетного класса. Она действует как область хранения для приоритетных данных после того, как она была классифицирована и после того, как она была разрешена осведомленным о перегрузке определителем политики. Пакеты LLQ помещаются в эту скрытую очередь системы, если они не могут быть размещены непосредственно на выходе кольца передачи выходного интерфейса во время прерывания приема, что иначе называют функциональной перегрузкой. В этой ситуации, поскольку существует функциональная перегрузка, пакет оценивается по условному определителю политики LLQ во время прерывания приема, в то время как им все еще владеет драйвер приемного интерфейса. Если пакет не сброшен условным определителем политики LLQ, он помещается в эту скрытую очередь системы LLQ и прерывание приема устраняется. Таким образом, все пакеты, помещенные в эту скрытую очередь системы, адаптируются к осведомленному о перегрузке определителю политики LLQ и будут немедленно переданы из очереди в кольцо передачи выходного интерфейса планировщиком LLQ/CBWFQ.

Несмотря на существование этой очереди, ее поведение не похоже на очереди IOS, созданные для данных, не относящихся к типу LLQ (например, справедливой очереди и очередей полосы пропускания) в том, что никакое дополнительное время ожидания в очереди (выше времени ожидания кольца передачи) не будет внесено, так как пакеты в этой очереди будут немедленно передаваться в кольцо передачи планировщиком LLQ/CBWFQ. Если пакет не сброшен условным определителем политики при прерывании приема, то этот пакет LLQ будет находиться в этой скрытой очереди системы кратковременно перед выходом из очереди в кольцо передачи выходного интерфейса. В этом случае планировщик LLQ/CBWFQ немедленно передает пакет в кольцо передачи выходного интерфейса. Условный определитель политики работает перед допуском пакета в LLQ/CBWFQ, ограничивая тем самым LLQ в соответствии с установленной нормой приоритетности.

В итоге рекомендуется сбросить данные LLQ, которые превышают норму приоритетности в течение времени перегрузки, вместо того, чтобы увеличивать время ожидания в очереди, которое является основным компонентом LLQ. Этот механизм условной политики разрешает использовать строго приоритетную очередь, не разрешая ей монополизировать весь интерфейс PLIM, что является шагом вперед по сравнению с функцией IOS legacy Priority Queueing (очередность с унаследованными приоритетами).

- **Предел очереди Pre-HQF:** NA
- **Поведение Pre-HQF “priority” + “random-detect”:** НП, взвешенное упредительное обнаружение с псевдослучайной логикой (WRED) не разрешено в LLQ.
- **Поведение Pre-HQF “priority” + “fair-queue”:** НП, справедливая очередность (fair-queue) не разрешена в LLQ.
- **Поведение Pre-HQF “priority” + “random-detect” + “fair-queue”:** НП, ни справедливая очередность (fair-queue), ни случайное обнаружение (random-detect) не поддерживаются в LLQ.

Поведение HQF

Аналогично поведению Pre-HQF за исключением того, что скрытая очередь больше не является скрытой, а предел очереди может настраиваться и по умолчанию составляет 64

пакета.

- **Предел очереди HQF:** 64 пакета
- **Поведение HQF “priority” + “random-detect”:** НП, взвешенное упредительное обнаружение с псевдослучайной логикой (WRED) не разрешено в LLQ.
- **Поведение HQF “priority” + “fair-queue”:** НП, справедливая очередность (fair-queue) не разрешена в LLQ.
- **Поведение HQF “priority” + “random-detect” + “fair-queue”:** НП, ни справедливая очередность (fair-queue), ни случайное обнаружение (random-detect) не поддерживаются в LLQ.

Определяемые пользователем классы, сконфигурированные с командой bandwidth

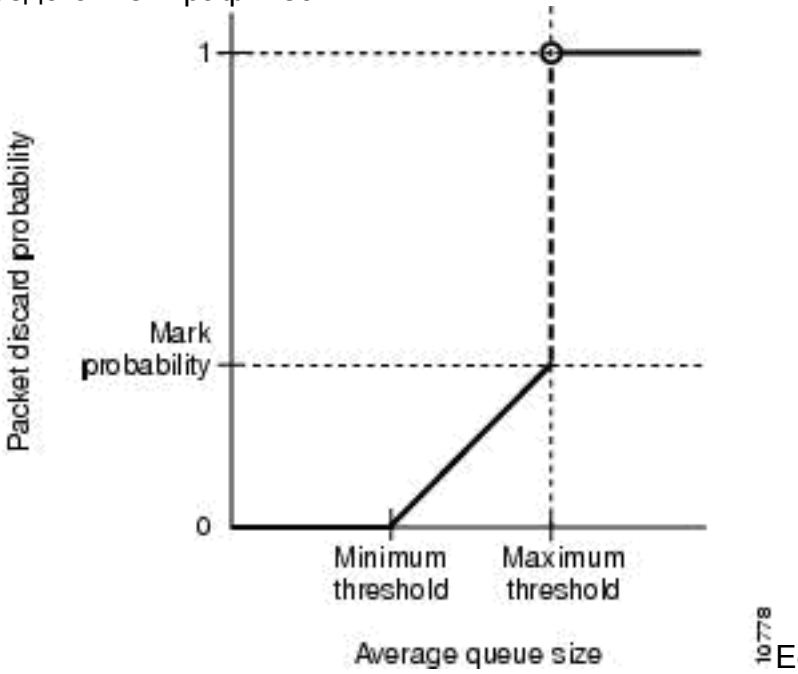
Применимо для определяемых пользователем классов MQC, настроенных с любыми вариантами команды bandwidth, включая bandwidth <kbps>, bandwidth percent и bandwidth remaining percent.

Поведение Pre-HQF

Предел очереди может настраиваться и по умолчанию составляет 64 пакета. Если во время прерывания приема требуется поставить в очередь пакет и при этом число пакетов в очереди превысит 64, последний пакет будет отброшен.

- **Предел очереди Pre-HQF:** 64 пакета, возможна настройка с помощью команды queue-limit.
- **Поведение Pre-HQF “bandwidth” + “random-detect”:**Пример:`policy-map PRE_HQF_BANDWIDTH_WRED  class 1    bandwidth 32    random-detect`Если любой вариант полосы пропускания настроен наряду с вариантом случайного обнаружения, игнорируются все пределы очереди, что эффективно удаляет любое ограничение буфера в классе. Другими словами, случайное обнаружение и ограничение очереди взаимно исключают друг друга в образах Pre-HQF. При использовании случайного обнаружения как стратегии сбрасывания текущий размер очереди не ограничивается и теоретически допускается занятие каждого буфера, выделенного для справедливой очереди, основанной на классах, причем число этих буферов определяется по точке назначения политики обслуживания:Физический интерфейс: 1000 пакетов, возможна настройка с помощью команды CLI-интерфейса: hold-queue outATM PVC: 500 пакетов, возможна настройка с помощью команды CLI-интерфейса PVC: vc-hold-queueFrame-Relay map-class: 600 пакетов, возможна настройка с помощью команды frame-relay раздела map-class CLI-интерфейса: frame-relay holdqПолитика формирования трафика на основе классов, примененная к (суб)интерфейсу (pre-HQF): 1000 пакетов, возможна настройка с помощью команды CLI-интерфейса класса MQC: shape max-buffers.**Примечание:** Весь Frame-Relay и Класс, Базирующиеся примеры формирования принимают сумму формирователей, не превышают интерфейсную тактовую частоту.**Примечание:** В образах перед HQF, когда Класс Базирующаяся Политика формирования применена к (sub) интерфейс, остерегаются скорости базового физического интерфейса, как интерфейсы <2 Мбит/с

примут значение по умолчанию к Взвешенной Справедливой очереди и интерфейсам>, 2 Мбит/с примут значение по умолчанию к FIFO. В pre-HQF формирующая очередь подпитывает очередь ожидания интерфейса либо при применении политики формирования, либо на уровне физического интерфейса.Во время прерывания приема, каждый раз, когда пакет становится кандидатом для выходной очереди интерфейса, размер средней очереди WRED вычисляется по формуле:`Average Queue Size = (old_average * (1-1/2^n)) + (current_queue_size * 1/2^n)`Если результирующий средний размер очереди:Меньше, чем минимальный порог WRED, пакет вставляется в очередь и прерывание передачи прекращается.Между минимальным и максимальным порогом WRED возможен сброс пакета, причем вероятность тем выше, чем ближе средний размер очереди к максимальному порогу WRED. Если средний размер очереди точно равен максимальному порогу WRED, пакет сбрасывается согласно знаменателю метки вероятности. Знаменатель метки вероятности также служит базовой линией для определения, какой процент пакетов сбрасывается, когда средний предел очереди не равен точно максимальному порогу WRED, но превышает минимальный порог WRED. Ниже представлен графический



пример:Если пакет удален, прерывание приема прекращается, а случайное сбрасывание увеличивается. Если пакет не удален, он ставится в очередь, а прерывание приема прекращается.Больше, чем минимальный порог WRED, пакет сбрасывается, прерывание передачи прекращается, а отбрасывание остатка увеличивается.

- **Примечание:** Приоритет IP-трафика базировался и основанный на DSCP WRED (по умолчанию) позволяют min-threshold, max-threshold, и отмечают знаменатель вероятности, который будет определен по-другому для других значений. Здесь взвешенный компонент случайного раннего обнаружения очевиден. Можно защитить определенные значения ToS относительно других значений посредством настройки их относительных порогов и знаменателей меток вероятности.Когда случайное обнаружение и полоса пропускания настроены вместе, текущий размер очереди может быть больше максимального порога WRED в любой момент времени. Это происходит из-за того, что минимальные и максимальные пороги WRED действуют только на основе среднего (а не текущего) размера очереди. Это предоставляет возможность освободить все буферы, выделенные для очереди, основанной на классах, что может закончиться небуферными сбрасываниями, происходящими в пределах справедливой очереди,

основанной на классах (см. идентификатор ошибки Cisco CSCsm94757).

- **Поведение Pre-HQF “bandwidth” + “fair-queue”:** НП, справедливая очередь (fair-queue) не разрешена в классе пропускной способности (bandwidth).
- **Поведение Pre-HQF “bandwidth” + “random-detect” + “fair-queue”:** НП, справедливая очередь (fair-queue) не разрешена в классе пропускной способности (bandwidth)

Поведение HQF

Поведение аналогично описанному в разделе Pre-HQF.

- **Предел очереди HQF:** 64 пакета, возможна настройка с помощью команды queue-limit.То же самое, что и для pre-HQF.
- **Поведение HQF “bandwidth” + “random-detect”:**Пример:
```
policy-map HQF_BANDWIDTH_WRED
 class 1
   bandwidth 32
   queue-limit 512
   random-detect
```
Примечание: Ограничение очереди по умолчанию является 64 пакетами.Поведение аналогично описанному в разделе Pre-HQF с одним важным исключением. В образах HQF случайное обнаружение и предел очереди могут сосуществовать в том же самом классе, определяемом пользователем (или в классе по умолчанию), и предел очереди будет включен и установлен на 64 пакета в заданной по умолчанию конфигурации. В этом качестве предел очереди будет определять максимальный текущий размер очереди в классе случайного обнаружения, предоставляя механизм для ограничения небуферных сбрасываний, описанный в разделе Pre-HQF. Из-за этого добавления настроенный предел очереди должен быть по крайней мере такого же размера, что и максимальный порог случайного обнаружения, который по умолчанию должен быть равен 40 пакетам, или же синтаксический анализатор отклонит конфигурацию.Для этого введена проверка текущего предела очереди в классах WRED, посредством чего, даже если среднее вычисление глубины очереди меньше, чем максимальный порог, а текущий (но не средний) размер очереди больше, чем предел очереди, пакет будет сброшен, прерывание приема прекращено, а отбрасывание остатка зарегистрировано. Обратите внимание, что, если предел очереди установлен достаточно высоким, чтобы исчерпать совокупные буфера основанной на классе очереди, небуферные сбрасывания могут все еще происходить. Совокупные буферы для HQF определены ниже:Физический интерфейс: 1000 пакетов, возможна настройка с помощью команды CLI-интерфейса: hold-queue outATM PVC: 500 пакетов, возможна настройка с помощью команды CLI-интерфейса PVC: vc-hold-queueFrame-Relay map-class: 600 пакетов, возможна настройка с помощью команды frame-relay раздела map-class CLI-интерфейса: frame-relay holdqПолитика формирования трафика на основе классов, примененная к физическому интерфейсу в коде HQF: 1000 пакетов, настраиваемых с комбинацией команд CLI-интерфейса hold-queue out и child policy queue-limit, где child policy queue-limit является верхней границей команды интерфейса hold-queue out.Политика формирования трафика на основе классов, примененная к субинтерфейсу в коде HQF: 512 пакета, без настройки (исследуйте группу платформы NSSTG QoS, возможна ли настройка)**Примечание:** Весь Frame-Relay и Класс, Базирующиеся примеры формирования принимают сумму формирователей, не превышают интерфейсную тактовую частоту.Ниже представлен практический пример:
```
policy-map JACKLYN
 class 1
```

```
bandwidth 64
queue-limit 500 packets
 random-detect
 random-detect precedence 1 22 300
```
В процессе этого вывода трафик не пропускался через интерфейс:
```
F340.11.25-7200-5_LAC#show policy-map interface | i queue
      queue limit 500 packets
      (queue depth/total drops/no-buffer drops) 0/387595/0
!--- Current_q_depth is 0 Mean queue depth: 107 packets !--- last calculation of
Average_queue_depth
```
В этом пункте начался трафик. Это поток без всплесков со скоростью 400 пакетов в секунду, состоящий из кадров по 1000 байт:
```
F340.11.25-7200-5_LAC#show policy-map interface | i queue
      queue limit 500 packets
      (queue depth/total drops/no-buffer drops) 461/387641/0
!--- 461 is Current_q_depth > Prec 1 max-thresh of 300 !--- but < "queue-limit 500 packets".
Mean queue depth: 274 packets !--- Avg_q_depth is rising, Mark Prob Denom is being used.
F340.11.25-7200-5_LAC#show policy-map interface | i queue queue limit 500 packets (queue
depth/total drops/no-buffer drops) 363/387919/0 !--- 363 is Current_q_depth and it is
falling compared to last !--- iteration because WRED is random dropping packets. Mean queue
depth: 351 packets !--- Avg_q_depth is now above max_thresh, WRED starts to tail-drop !---
in addition to random-drop. F340.11.25-7200-5_LAC#show policy-map interface | i queue queue
limit 500 packets (queue depth/total drops/no-buffer drops) 199/388263/0 Mean queue depth:
312 packets F340.11.25-7200-5_LAC#show policy-map interface | i queue queue limit 500
packets (queue depth/total drops/no-buffer drops) 303/388339/0 Mean queue depth: 276 packets
F340.11.25-7200-5_LAC#show policy-map interface | i queue queue limit 500 packets (queue
depth/total drops/no-buffer drops) 325/388498/0 Mean queue depth: 314 packets F340.11.25-
7200-5_LAC#show policy-map interface | i queue queue limit 500 packets (queue depth/total
drops/no-buffer drops) 298/390075/0 Mean queue depth: 300 packets
```
Обратите внимание, как в конечном счете для потока без всплесков средняя глубина очереди WRED будет равняться текущей глубине очереди, что является ожидаемым поведением.

- **Поведение HQF “bandwidth” + “fair-queue”:**Когда полоса пропускания и справедливая очередь применены вместе к определяемому пользователем классу HQF, каждой основанной на потоке очереди выделен предел, равный .25 * queue-limit. Поскольку заданный по умолчанию предел очереди — 64 пакета, для каждой очереди на основе потока в справедливой очереди будет выделено 16 пакетов. Если бы четыре потока пересекали этот класс, то по умолчанию у каждой очереди потока было бы 16 пакетов, поэтому никогда полное число пакетов в очереди не будет превышать 64 (4 *16). Все отбрасывания остатка в индивидуальной очереди потока регистрируются как flowdrop. Если число очередей потока достаточно высоко, как и предел очереди, то возникает другая возможность небуферных сбрасываний. Например, точка присоединения политики является физическим интерфейсом, для которого выделены 1000 составных буферов:
```
policy-map TEST
 class 1
  bandwidth 32
  fair-queue 1024
  queue-limit 128
```
В этой конфигурации оценочный трафик во всех очередях потока может исчерпать составные буфера интерфейса и привести к небуферным отбрасываниям в других определяемых пользователем классах (см. идентификатор ошибки Cisco CSCsw98427). Это происходит из-за того, что 1024 очереди потока, каждая с пределом 32 пакета, могут легко превысить выделенные ресурсы 1000 составных буферов интерфейса очереди, основанной на классах.
- **Поведение HQF “bandwidth” + “random-detect” + “fair-queue”:**Пример:
```
policy-map TEST
 class 1
  bandwidth 32
  fair-queue 1024
  queue-limit 128
```

`random-detect`Так же, как полоса пропускания и справедливая очередь в разделе, за исключением WRED, средний размер очереди вычисляется каждый раз, когда прибывает пакет, чтобы решить, должен ли пакет быть отброшен случайным образом или по переполнению очереди. Как и с Pre-HQF, все очереди потока совместно используют один экземпляр порогов WRED, а это означает, что все пакеты, направленные во все очереди потока, используются для вычисления средней глубины очереди WRED. Затем при решении об отбрасывании минимальные и максимальные пороги WRED сравниваются с совокупными пакетами во всех очередях потока. Однако другое отклонение от полосы пропускания и справедливой очереди в разделе (связанное с тем, что один экземпляр порогов WRED применяется ко всем основанным на потоке очередям) – индивидуальный предел очереди потока (.25 * "предел очереди") игнорируется, а вместо него для проверки текущего предела очереди используется совокупный предел очереди классов.

Поведение класса по умолчанию

Поведение Pre-HQF

В pre-HQF в значениях по умолчанию класса используется справедливая очередь, все очереди потока совместно используют предел очереди для класса (значение по умолчанию 64), и резервирование полосы пропускания отсутствует. Другими словами, общее количество пакетов, находящихся во всех очередях потока, никогда не будет превышать предел очереди. Количество пакетов, находящихся в каждой очереди потока, будет зависеть от расчетного веса очереди потока. И наоборот, если справедливая очередь и случайное обнаружение используются вместе в классе по умолчанию, предел очереди будет проигнорирован, а все очереди потока будут совместно использовать те же самые пороги WRED. По существу, все пакеты, находящиеся в настоящий момент во всех очередях потока, будут использоваться для вычисления среднего размера очереди WRED. Поскольку у текущего размера очереди нет верхнего предела в этой конфигурации, возможность небуферных сбрасываний будет высока (см. идентификатор ошибки Cisco CSCsm94757).

- Добавление полосы пропускания к классу по умолчанию описано в разделе Поведение Pre-HQF — определяемые пользователем классы, настроенные командой bandwidth.
- *Добавление полосы пропускания и случайного обнаружения к классу по умолчанию описано в разделе Поведение Pre-HQF “bandwidth” + “random-detect” раздела Поведение Pre-HQF — определяемые пользователем классы, настроенные командой bandwidth.*

Примечание: В предварительном HQF пропускная способность не может сосуществовать с fair-queue в class-default.

Поведение HQF

В HQF в значениях по умолчанию класса используется очередь в порядке поступления (FIFO) с выделением резервирования псевдополосы пропускания на основании оставшихся выделенных ресурсов классов, определяемых пользователем. Поведение по умолчанию для класса class-default HQF описано в разделе Поведение HQF — определяемые пользователем классы, настроенные командой bandwidth. Всегда, независимо от конфигурации, у класса по умолчанию (class-default) в образах HQF будет неявное

резервирование полосы пропускания, эквивалентное неиспользованной полосе пропускания интерфейса классов, определяемых пользователем. По умолчанию класс class-default получает минимум 1 % интерфейса или полосы пропускания родительского формирования. Можно также явно настроить полосу пропускания интерфейса командной строки в классе по умолчанию.

- *Если справедливая очередь (fair-queue) настроена в классе по умолчанию (Class-Default), поведение соответствует Поведению HQF “bandwidth” + “fair-queue” , описанному в разделе Поведение HQF — определяемые пользователем классы, настроенные командой bandwidth.*
- *Если справедливая очередь (fair-queue) и случайное обнаружение (random-detect) настроены вместе в классе по умолчанию (Class-Default), поведение соответствует Поведению HQF “bandwidth” + “random-detect” + “fair-queue” , описанному в разделе Поведение HQF — определяемые пользователем классы, настроенные командой bandwidth.*

Дополнительные сведения

- Руководство по конфигурации решений для качества сервиса Cisco IOS, выпуск 12.4T
- QoS — иерархическая структура очередей (HQF)
- Поддержка технологии QoS
- Поддержка продуктов маршрутизаторов
- Cisco Systems – техническая поддержка и документация