了解在Cisco IOS软件平台的队列限制和输出丢弃

目录

简介

本文档仅适用于 Cisco IOS® 软件平台，此类平台通常包括 Cisco 7200VXR 和 Cisco ISR 3800、2800、1800 系列路由器，以及包括 3700、3600、2600 和 1700 系列路由器在内的传统 Cisco 接入路由器。

先决条件

要求

Cisco 建议您了解以下主题：

- Cisco IOS 服务质量解决方案
- QoS---分层排队框架 (HQF)

使用的组件

本文档中的信息基于以下软件版本：

- 对于 Pre-HQF：运行 Cisco IOS 软件版本 12.3(26) 的 Cisco 路由器
- 对于 HQF：运行 Cisco IOS 软件版本 12.4(22)T 的 Cisco 路由器

本文档中的信息都是基于特定实验室环境中的设备编写的。本文档中使用的所有设备最初均采用原始（默认）配置。如果您使用的是真实网络，请确保您已经了解所有命令的潜在影响。

规则

有关文档规则的详细信息，请参阅 Cisco 技术提示规则。

Class-Based Weighted Fair Queueing 入门

在 pre-HQF IOS 映像中，一般来说，根据类的*权重*，任何使用 **bandwidth** 命令的类将优先于不使用 *bandwidth* 或 *priority* 的类。要了解 Class-Based Weighted Fair Queueing (CBWFQ) 调度算法，您必须首先了解权重的概念：对于基于流的公平队列，它特定于流；对于基于类的加权公平队列中的单个基于类的队列，它特定于类。

计算基于流的公平队列的权重的公式为：

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

基于类的加权公平队列内的用户定义的类将通过 **bandwidth** 命令函数（已在类中配置为基于类的队列中所有*带宽类*总和的一部分）获取各自的权重。具体公式是专有的。

在 HQF 映像中，可在用户定义的类和默认使用 *fair-queue* 的类中配置的基于流的 fair-queue 将平等地进行调度（而不是按权重进行调度）。此外，在 HQF 中，基于类的队列的调度优先级将根据 HQF 调度程序而非类的传统权重公式来确定。

注意： 本部分不会对基于类的排队操作进行全面的行为分析。本部分的目的是进行简单的介绍，因为 CBWFQ 可用于了解队列限制和输出丢弃。

了解队列限制和输出丢弃

用“priority”命令配置的用户定义的类

适用于使用 **priority** 命令的任何变体（包括 **priority**、**priority<kbps>** 和 **priority percent**）配置的 MQC 用户定义的类。

Pre-HQF 行为

从技术上来说，会有一个供所有优先级数据共享的“隐藏”系统队列存在，即使不存在可查看它的 CLI，且不可对其进行配置。在对优先级数据进行分类并由能够感知拥塞的监察器对优先级数据进行许可后，该队列将作为这些数据的保存区域。如果在接收中断期间无法将 LLQ 数据包直接放置在输出接口的传输环路上（又称为功能性拥塞），则会将其放入此“隐藏”系统队列。在这种情况下，因为存在功能性拥塞，数据包将在接收中断期间根据 LLQ 条件监察器进行评估，同时仍然归接收接口的驱动程序所有。如果数据包未被 LLQ 条件监察器丢弃，则会将其放入此“隐藏”LLQ 系统队列，同时释放接收中断。因此，所有放入此“隐藏”系统队列中的数据包均符合 LLQ 的能够感知拥塞的监察器的标准，并且会立即由 LLQ/CBWFQ 调度程序出列到输出接口的传输环路。

尽管存在此队列，但其行为仍不同于为非 LLQ 数据创建的 IOS 队列（如 fair-queue 和带宽队列），这是因为此队列中的数据包会立即被 LLQ/CBWFQ 调度程序清空到传输环路，因而不会导致其他排队延迟（传输环路延迟除外）。如果优先级数据包在接收中断期间未被条件监察器丢弃，则该 LLQ 数据包将会在此“隐藏”系统队列中短暂存在，然后出列到输出接口的传输环路。在这种情况下，LLQ/CBWFQ 调度程序会立即将数据包提供给输出接口的传输环路。在将数据包提交给 LLQ/CBWFQ 之前，条件监察器已经运行，因此可将 LLQ 限制为配置的优先级速率。

总之，建议在拥塞期间丢弃超出优先级速率的 LLQ 数据，以免导致其他排队延迟，这是 LLQ 的基本组成部分。此条件管制机制允许采用严格的优先级队列，而不允许优先级队列垄断整个接口 PLIM，这是对 IOS 的传统优先级排队功能的一项改进。

- **Pre-HQF 队列限制**：NA
- **Pre-HQF“priority”+“random-detect”行为**：不可用，LLQ 中不允许 WRED。
- **Pre-HQF“priority”+“fair-queue”行为**：不可用，LLQ 中不允许 fair-queue。
- **Pre-HQF“priority”+“random-detect”+“fair-queue”行为**：不可用，LLQ 中不支持 fair-queue 或 random-detect。

HQF 行为

和 Pre-HQF 一样，只是隐藏队列不再能够隐藏，队列限制现在可以进行配置，默认值为 64 个数据包。

- **HQF 队列限制**：64 个数据包
- **HQF“priority”+“random-detect”行为**：不可用，LLQ 中不允许 WRED。
- **HQF“priority”+“fair-queue”行为**：不可用，LLQ 中不允许 fair-queue。
- **HQF“priority”+“random-detect”+“fair-queue”行为**：不可用，LLQ 中不支持 fair-queue 或 random-detect。

用“bandwidth”命令配置的用户定义的类

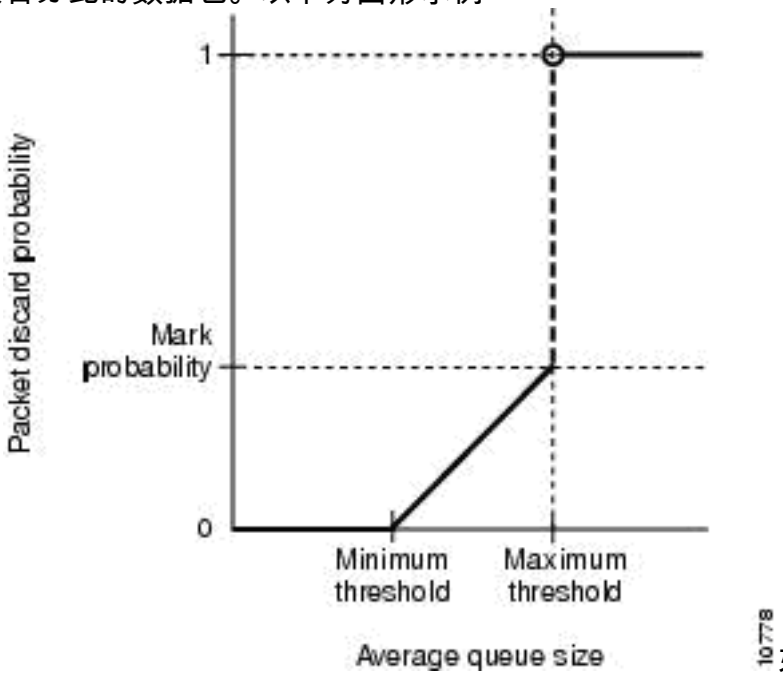
适用于使用 **bandwidth** 命令的任何变体（包括 **bandwidth <kbps>**、**bandwidth percent** 和 **bandwidth remaining percent**）配置的 MQC 用户定义的类。

Pre-HQF 行为

默认队列限制为 64 个数据包，可调整。在接收中断期间，如果需要将某个数据包排入队列中，从而导致队列中的数据包超出 64 个，则会对该数据包执行尾部丢弃。

- **Pre-HQF 队列限制**：64 个数据包，可通过 queue-limit 进行调整。
- **Pre-HQF“bandwidth”+“random-detect”行为**：**示例：**
```
policy-map PRE_HQF_BANDWIDTH_WRED
 class 1
   bandwidth 32
   random-detect
```
如果将 bandwidth 的任意变体与 random-detect 的任意变体一起进行配置，请忽略任何 queue-limit CLI，即可有效去除类中的任何缓冲限制。换言之，在 Pre-HQF 映像中，random-detect 和 queue-limit 是互斥的。使用 random-detect 作为丢弃策略，当前队列大小没有限制，并且在理论上可以占用分配给基于类的 fair-queue 的每个缓冲区，分配给基于类的 fair-queue 的此数量的缓冲区在此处基于服务策略附加点派生：物理接口：1000 个数据包，可使用接口 CLI hold-queue out 进行调整ATM PVC：500 个数据包，可使用 PVC CLI vc-hold-queue 进行调整帧中继 map-class：600 个数据包，可使用 frame-relay map-class CLI frame-relay holdq 进行调整基于类的整形策略应用于（子）接口 (Pre-HQF)：1000 个数据包，可使用 MQC 类 CLI shape max-buffers 进行调整。**注意：** 所有帧中继和基于类的整形示例均假设整形器的总和不超过接口时钟频率。**注意：** 在 Pre-HQF 映像中，当基于类的整形策略应用于（子）接口时，请注意基础物理接口的速度，因为小于 2Mbps 的接口将默认为加权公平队列，大于 2Mbps 的接口将默认为 FIFO。在 Pre-HQF 中，无论整形策略是在子接口级别附加的，还是在物理接口级别附加的，整形队列都将提供接口保持队列。在接收中断期间，每次有数据包变为接口的输出队列的候选者时，都会使用此公式计算 WRED 平均队列大小：`Average Queue Size = (old_average * (1-1/2^n)) + (current_queue_size * 1/2^n)`如果得到的平均队列大小：小于 WRED 最小阈值，则会将数据包排入队列并释放接收中断。处于 WRED 最小阈值和 WRED 最大阈值之间，则平均队列大小越接近 WRED 最大阈值，丢弃数据包的可能性越大。如果平均队

列大小正好等于 WRED 最大阈值，则根据标记概率分母丢弃数据包。在平均队列大小不等于 WRED 最大阈值但大于 WRED 最小阈值时，还可使用标记概率分母作为基准来确定要丢弃多大百分比的数据包。以下为图形示例



：如果数据包已丢弃，则释放接收中断，并增加一次“随机丢弃”。如果数据包未丢弃，则将数据包排入队列，并释放接收中断。大于 WRED 最大阈值，则丢弃数据包，释放接收中断，并增加一次“尾部丢弃”。

- **注意：** 基于 Ip Precedence（默认）和基于 DSCP 的 WRED 允许为不同的值定义不同的最小阈值、最大阈值和标记概率分母。这是随机早期检测的加权组件最突出的特点。通过调整某些 Tos 值的相对阈值和标记概率分母，您可以为这些 Tos 值提供保护（相对于其他值而言）。随机检测和带宽在一起配置时，任何时候，当前队列大小均可大于 WRED 最大阈值。这是因为 WRED 最小阈值和最大阈值只能根据平均（而不是当前）队列大小进行操作。这可能导致分配给基于类的队列的所有缓冲区过期，以及基于类的公平队列中的任何位置发生“无缓冲区丢弃”（请参阅 Cisco Bug ID CSCsm94757）。
- **Pre-HQF“bandwidth”+“fair-queue”行为：**不可用，带宽类中不允许 fair-queue。
- **Pre-HQF“bandwidth”+“random-detect”+“fair-queue”行为**：不可用，带宽类中不允许 fair-queue

HQF 行为

其行为与 Pre-HQF 部分描述的行为相同。

- **HQF 队列限制**：64 个数据包，可通过 queue-limit 进行调整。这与 Pre-HQF 中的队列限制相同。
- **HQF“bandwidth”+“random-detect”行为**：**示例：**

```
policy-map HQF_BANDWIDTH_WRED
 class 1
   bandwidth 32
   queue-limit 512
   random-detect
```

注意： 默认队列限制为 64 个数据包。此行为与对应的 Pre-HQF 部分相同，但有一个重要例外。在 HQF 映像中，random-detect 和队列限制可在用户定义的同一类（或 class class-default）中共存，而且将在默认配置中启用队列限制并将其调整为 64 个数据包。同样地，队列限制将在 random-detect 类中用作最大当前队列大小，因此可以作为一种机制来限制“无缓冲区丢弃”（已在相应的 Pre-HQF 部分讨论过）。由于此新增内容，配置的队列限制必须至少和 random-detect 最大阈值一样大，此时 random-detect 最大阈值将默认为 40 个数据包，否则分析程序将拒绝该配置。这样就可以在 WRED 类中引入“当前队列限制”检查，因此，即使平均队列深度计算小于最大阈值，但如果当前（而不是平均）队列大小大于队列限制

，则也会丢弃数据包，释放接收中断并记录尾部丢弃。切记，如果将队列限制调整得足够高，耗尽了基于类的队列的聚合排队缓冲区，此时仍可能发生“无缓冲区丢弃”。HQF 的聚集排队缓冲区定义如下：物理接口：1000 个数据包，可使用接口 CLI hold-queue out 进行调整ATM PVC：500 个数据包，可使用 PVC CLI vc-hold-queue 进行调整帧中继 map-class：600 个数据包，可使用 frame-relay map-class CLI frame-relay holdq 进行调整基于类的整形策略以 HQF 代码方式应用于物理接口：1000 个数据包，可使用接口 CLI hold-queue out 和子策略 queue-limit 的组合进行调整（如果子策略 queue-limit 的上限为接口 hold-queue out）。基于类的整形策略以 HQF 代码方式应用于子接口：512 个数据包，不可调（如果可以调整，请配合 NSSTG QoS 平台团队查明原因）**注意：** 所有帧中继和基于类的整形示例均假设整形器的总和不超过接口时钟频率。以下为实际示例：`policy-map JACKLYN`

```
 class 1
    bandwidth 64
    queue-limit 500 packets
     random-detect
     random-detect precedence 1 22 300
```

在此输出期间，没有流量通过接口生成：`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
```

这时，流量开始生成。数据流是非突发性的，速度为 400PPS，包括 1000 字节的帧：`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
```

请注意，对于非突发性数据流，如何才能最终让 WRED 平均队列深度与当前队列深度保持一致（这是预期行为）。

- **HQF“bandwidth”+“fair-queue”行为：**将带宽和公平队列一起应用于 HQF 用户定义的类时，将会为每个基于流的队列分配一个队列限制（等于 .25 * 队列限制）。由于默认队列限制为 64 个数据包，公平队列中每个基于流的队列将分配到 16 个数据包。如果有四个流经过此类，默认情况下，每个流队列将有 16 个数据包，因此您绝不会看到已排队数据包的总和大于 64 (4*16) 的情况。来自单个流队列的所有尾部丢弃都将记录为“流丢弃”。如果流队列的数量远大于队列限制的数量，则这时也可能会发生“无缓冲区丢弃”。例如，假设策略附加点是物理接口，并在该接口处分配了 1000 个聚集缓冲区：`policy-map TEST`

```
 class 1
  bandwidth 32
  fair-queue 1024
  queue-limit 128
```

在此配置中，所有流队列中的可感知流量将耗尽聚集接口缓冲区，并导致其他用户定义的类中发生“无缓冲区丢弃”（请参阅 Cisco Bug ID CSCsw98427）。这是因为，1024 个流队列（每个流队列的队列限制为 32 个数据包）能够轻易使基于类的排队缓冲区（具有 1000 个聚集接口）发生分配超载。

- **HQF“bandwidth”+“random-detect”+“fair-queue”行为：示例：`policy-map TEST`**

```
 class 1
  bandwidth 32
```

```
fair-queue 1024
queue-limit 128
```

`random-detect`除了每次在数据包到达时通过计算 WRED 平均队列大小来决定对数据包执行随机丢弃或尾部丢弃外，其他行为与该部分 bandwidth 和 fair-queue 的行为相同。和 Pre-HQF 一样，所有流队列将共享 WRED 阈值的一个实例，这意味着将使用已排入所有流队列中的全部信息包来计算 WRED 平均队列深度，然后通过丢弃决策将 WRED 最小阈值和最大阈值应用于所有流队列中的聚集数据包。但是，另一个与该部分的 bandwidth 和 fair-queue 不同的地方在于，由于已将 WRED 阈值的一个实例应用于所有基于流的队列，因此不会采用各个流队列的队列限制（.25 * 队列限制），而是采用当前队列限制检查的类聚集队列限制。

类的默认行为

Pre-HQF 行为

在 Pre-HQF 中，Class Default 默认为 fair-queue，所有流队列具有相同的类队列限制（默认为 64），并且没有带宽预留。换句话说，所有流队列中已排列数据包的总数绝不会超出队列限制。每个流队列中已排列数据包的数量将取决于流队列的计算权重。相反，如果在 class-default 中同时使用 fair-queue 和 random-detect，将会忽略队列限制，所有流队列将具有相同的 WRED 阈值。同样，所有流队列中当前已排列的所有数据包将用于计算 WRED 平均队列大小。由于当前队列大小在此配置中没有上限，因此很有可能会发生“无缓冲区丢弃”（请参阅 Cisco Bug ID CSCsm94757）。

- 将 bandwidth 添加到 class-default 时，会出现 Pre-HQF 行为 - 用“bandwidth”命令配置的用户定义的类部分所述行为。
- 将 bandwidth 和 random-detect 添加到 class class-default 时，会出现 Pre-HQF 行为 - 用“bandwidth”命令配置的用户定义的类中 *Pre-HQF“bandwidth”+“random-detect”行为*部分所述行为。

注意： 在 Pre-HQF 中，bandwidth 与 fair-queue 不能在 class-default 中共存。

HQF 行为

在 HQF 中，Class Default 的默认值为 FIFO 队列，并已根据用户定义的类中的残余分配分配了伪带宽预留。同样地，关于 HQF class class-default 的默认行为，请参阅 HQF 行为 - 用“bandwidth”命令配置的用户定义的类部分。任何时候，无论何种配置，HQF 映像中的 class class-default 将始终具有一个隐式带宽预留，该带宽等于未使用接口带宽（即未被用户定义的类占用）。默认情况下，class-default 类的带宽至少为接口或父整形带宽的 1%。此外，还可以在 class default 中显式配置带宽 CLI。

- 如果在 class class-default 中配置了 fair-queue，则行为与 HQF 行为 - 用“bandwidth”命令配置的用户定义的类中 *HQF“bandwidth”+“fair-queue”行为*部分所述相符。
- 如果在 Class-Default 中同时配置了 fair-queue 和 random-detect，则行为与 HQF 行为 - 用“bandwidth”命令配置的用户定义的类中 *HQF“bandwidth”+“random-detect”+“fair-queue”行为*部分所述相符。

相关信息

- **Cisco IOS 服务质量解决方案配置指南，版本 12.4T**
- **QoS---分层排队框架 (HQF)**
- **QoS 技术支持**

- 路由器产品支持
- 技术支持和文档 - Cisco Systems