**QOS 介绍及SAI接口说明**

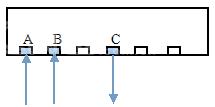
# 一、QOS概述

QoS（Quality of Service，[服务质量](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E8%B4%A8%E9%87%8F/9401950" \t "https://baike.baidu.com/item/qos/_blank)）指一个网络能够利用各种基础技术，为指定的[网络通信](https://baike.baidu.com/item/%E7%BD%91%E7%BB%9C%E9%80%9A%E4%BF%A1" \t "https://baike.baidu.com/item/qos/_blank)提供更好的服务能力, 是网络的一种安全机制， 是用来解决网络延迟和阻塞等问题的一种技术。 在正常情况下，如果网络只用于特定的无时间限制的应用系统，并不需要QoS，比如[Web](https://baike.baidu.com/item/Web" \t "https://baike.baidu.com/item/qos/_blank)应用，或[E-mail](https://baike.baidu.com/item/E-mail" \t "https://baike.baidu.com/item/qos/_blank)设置等。但是对关键应用和多媒体应用就十分必要。当网络过载或拥塞时，QoS 能确保重要业务量不受[延迟](https://baike.baidu.com/item/%E5%BB%B6%E8%BF%9F/5950826" \t "https://baike.baidu.com/item/qos/_blank)或[丢弃](https://baike.baidu.com/item/%E4%B8%A2%E5%BC%83/10248074" \t "https://baike.baidu.com/item/qos/_blank)，同时保证网络的高效运行。

# 二、流控

### ****基本知识****

流量控制简称流控，是以太网的一项基本功能，可以防止在端口拥塞的情况下出现丢帧。一般交换芯片常见的实现方法是基于pause帧的流控。以下是一个简单的应用场景：



端口A和B接收报文，端口C向外转发报文。如果端口A和B的收包速率之和大于端口C的带宽，那么部分报文就会缓存在设端口A和B就会向外发送时间参数不为0的PAUSE帧，通知对端暂停发送一段时间。当buffer的占用率降低到低水线XON以下时，端口A和B就会向外发送时间参数等于0的PAUSE帧，表示收到PAUSE帧的一方可以马上发送数据包。

以上的描述有个先决条件，那就是端口A和B及其对端设备都支持备内部的报文buffer中。当buffer的占用率达到高水线XOFF时，流控并使能了流控功能。

### ****PFC****

全称Priority-based Flow Control，跟流控原理相同，它是基于优先级的流控。

SONiC的PFC使能方法

"PORT\_QOS\_MAP": {

"Ethernet0,Ethernet1,Ethernet2,Ethernet3,Ethernet48": {

"pfc\_enable": "3,4"

}

}

### ****流控相关SAI接口****

* + **PFC使能**

接口名：set\_port\_attribute

参数1：sai\_object\_id\_t port

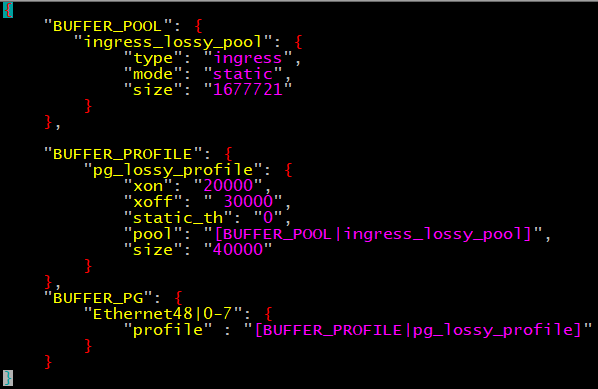
参数2：const sai\_attribute\_t \*attr 见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_PORT\_ATTR\_PRIORITY\_FLOW\_CONTROL | u8  使能队列的bitmap |

xp70不支持单独使能某个队列的pfc,如果任何一个队列的pfc使能了，则全部队列都使能

* + **水线配置**

水线配置涉及BUFFER\_PG -> BUFFER\_PROFILE -> BUFFER\_POOL的多级引用关系，可以通过以下json文件理解



交换芯片的入口buffer是所有端口共享的，但在流控场景下，可以进行分配。可以将若干个队列指定一个BUFFER\_POOL，BUFFER\_POOL的大小不能超过XPSAI\_BUFFER\_TOTAL\_BUFFER\_BYTES，xp70是64k\*256字节。

当前设备对几种拓扑的buffer占用情况进行了测试：

10G到10G线速打流：3840

40G到10G线速打流：154624

40G到40G线速打流：15360

SAI层接口也分成了三个：

1、

接口名：create\_buffer\_pool

功能： 创建buffer pool

参数1：sai\_object\_id\_t \*pool\_id 返回申请的pool\_id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list attr的列表，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_BUFFER\_POOL\_ATTR\_TYPE | u32  SAI\_BUFFER\_POOL\_TYPE\_INGRESS |
| SAI\_BUFFER\_POOL\_ATTR\_THRESHOLD\_MODE | u32  SAI\_BUFFER\_POOL\_THRESHOLD\_MODE\_STATIC |
| SAI\_BUFFER\_POOL\_ATTR\_SIZE | u32  buffer pool大小，单位字节，要保证所有配置buffer的队列buffer之和小于该值 |

2、

接口名：xpSaiCreateBufferProfile

功能：创建buffer profile

参数1：sai\_object\_id\_t \*buffer\_profile\_id 返回申请的buffer\_profile\_id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list attr的列表，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_BUFFER\_PROFILE\_ATTR\_BUFFER\_SIZE | u32  Buffer profile的buffer大小，单位字节 |
| SAI\_BUFFER\_PROFILE\_ATTR\_XOFF\_TH | u32  xoff的值，单位字节 |
| SAI\_BUFFER\_PROFILE\_ATTR\_XON\_TH | u32  xon的值，单位字节 |

要保证：BUFFER\_SIZE > XOFF\_TH > XON\_TH

3、

接口名：xpSaiSetIngressPriorityGroupAttr

功能：ingress\_pg\_id与buffer profile的绑定

参数1：sai\_object\_id\_t ingress\_pg\_id 端口+优先级与ingress\_pg\_id一一对应

参数2：const sai\_attribute\_t \*attr\_list attr的列表，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_INGRESS\_PRIORITY\_GROUP\_ATTR\_BUFFER\_PROFILE | oid  Buffer profile的id |

# 三、队列与优先级映射

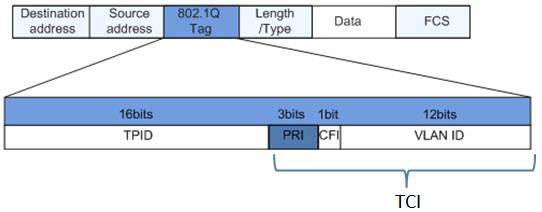
### ****基本知识****

不同的报文使用不同的QoS优先级，例如VLAN报文使用802.1p，IP报文使用DSCP。为了保证不同报文的服务质量，在报文进设备时，需要将报文携带的QoS优先级统一映射到设备内部的服务等级Service Class（也叫做调度优先级PHB）和丢弃优先级（也叫颜色Color），在设备内部，根据报文的服务等级进行拥塞管理，根据报文的颜色进行拥塞避免；在报文出设备时，需要将内部的服务等级和颜色映射为QoS优先级，以便后续网络设备能够根据QoS优先级提供相应的服务质量。

一般PHB的范围是0~7，或者分别用BE、AF1、AF2、AF3、AF4、CS6、CS7表示，Color有绿黄红三种。

### ****二层VLAN帧中的优先级****

二层帧中的优先级是专门针对VLAN帧的，因为普通二层帧中是不携带有优先级字段的。VLAN帧中的优先级那就是我们通常所说的802.1p优先级（由IEEE 802.1p协议定义），位于VLAN帧中的“802.1Q Tag”字段的“PRI”子字段中，如图所示。

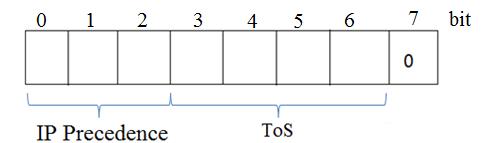


### ****三层IP报文中的优先级****

上面介绍的二层VLAN帧优先级比较简单，就是由PRI子字段的三位来标识，共有8种优先级，但在三层IP报文中，优先级的描述就要复杂许多，并且在不同时期出现了两种不同的优先级类型和不同的标识方法。

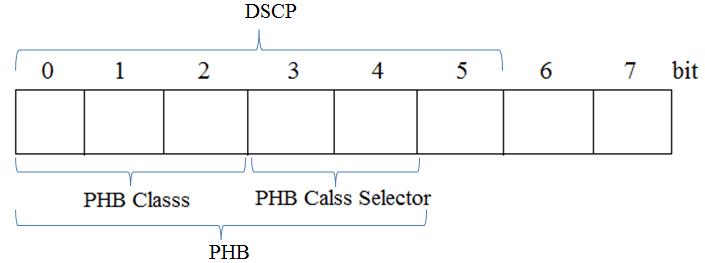
* + **ToS**

在早期的RFC 791标准中，IP数据包是依赖ToS（Type of Service，服务类型）字段来标识数据优先级值的。ToS是IP数据包中的IP报头中的一个字段（共1个字节），用来指定IP包的优先级，设备会优先转发ToS值高的数据包。ToS字段共一个字节（8位），包括三个部分：0~2共三位用来定义数据包的IP优先级（IP Precedence）、ToS和最后一个固定为0的位，如图所示。



* + **dscp**

在后来新的RFC 2474标准中，重新定义了原来IP包头部的ToS字段，并改称之为DS（Differentiated Services，差分服务）字段，也是共一个字节（8位）。总的来说，第0~5位（共六位）用来表示DSCP（Differentiated Services Code Point，差分服务代码点）优先级，取值范围为0~63，共能标识出64个优先级值（值越大，优先级越高），最后两位保留，用于显示拥塞通知（Explicit Congestion Notification，ECN）



### ****优先级映射相关SAI接口****

* + **映射表配置**

接口名：xpSaiCreateQosMap

功能：创建qos map

参数1：sai\_object\_id\_t \*qos\_map\_id 返回映射表id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list，一般为两个属性，一个id = SAI\_QOS\_MAP\_ATTR\_TYPE，value.s32为具体的type类型；一个id = SAI\_QOS\_MAP\_ATTR\_MAP\_TO\_VALUE\_LIST，value.qosmap为具体的映射表。定义见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_QOS\_MAP\_ATTR\_TYPE | s32  可选类型  SAI\_QOS\_MAP\_TYPE\_DOT1P\_TO\_TC  SAI\_QOS\_MAP\_TYPE\_DOT1P\_TO\_COLOR  SAI\_QOS\_MAP\_TYPE\_DSCP\_TO\_TC  SAI\_QOS\_MAP\_TYPE\_DSCP\_TO\_COLOR  SAI\_QOS\_MAP\_TYPE\_TC\_TO\_QUEUE  SAI\_QOS\_MAP\_TYPE\_TC\_AND\_COLOR\_TO\_DSCP  SAI\_QOS\_MAP\_TYPE\_TC\_AND\_COLOR\_TO\_DOT1P |
| SAI\_QOS\_MAP\_ATTR\_MAP\_TO\_VALUE\_LIST | qosmap  优先级和映射后的优先级的键值对 |

qosmap的类型sai\_qos\_map\_list\_t定义见以下代码

typedef enum \_sai\_packet\_color\_t

{

/\*\*

\* @brief Color Green

\*/

SAI\_PACKET\_COLOR\_GREEN,

/\*\*

\* @brief Color Yellow

\*/

SAI\_PACKET\_COLOR\_YELLOW,

/\*\*

\* @brief Color Red

\*/

SAI\_PACKET\_COLOR\_RED,

} sai\_packet\_color\_t;

/\*\*

\* @brief Defines QOS map types.

\*

\* @par Examples:

\*

\* dot1p/DSCP --> TC

\* dot1p/DSCP --> Color

\* dot1p/DSCP --> TC + Color

\* TC --> dot1p/DSCP.

\* TC + color --> dot1p/DSCP.

\* TC --> Egress Queue.

\*/

typedef struct \_sai\_qos\_map\_params\_t

{

/\*\* Traffic class \*/

sai\_cos\_t tc;

/\*\* DSCP value \*/

sai\_uint8\_t dscp;

/\*\* Dot1p value \*/

sai\_uint8\_t dot1p;

/\*\* PFC priority value \*/

sai\_uint8\_t prio;

/\*\* Priority group value \*/

sai\_uint8\_t pg;

/\*\*

\* @brief Egress port queue OID is not known at the time of map creation.

\* Using queue index for maps.

\*/

sai\_queue\_index\_t queue\_index;

/\*\* Color of the packet \*/

sai\_packet\_color\_t color;

} sai\_qos\_map\_params\_t;

typedef struct \_sai\_qos\_map\_t

{

/\*\* Input parameters to match \*/

sai\_qos\_map\_params\_t key;

/\*\* Output map parameters \*/

sai\_qos\_map\_params\_t value;

} sai\_qos\_map\_t;

typedef struct \_sai\_qos\_map\_list\_t

{

/\*\* Number of entries in the map \*/

uint32\_t count;

/\*\* Map list \*/

sai\_qos\_map\_t \*list;

} sai\_qos\_map\_list\_t;

* + **端口绑定映射表**

接口名：set\_port\_attribute

参数1：sai\_object\_id\_t port

参数2：const sai\_attribute\_t \*attr 见下表

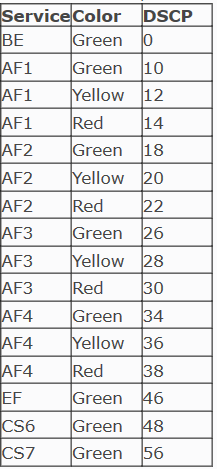
|  |  |
| --- | --- |
| Id | Value |
| SAI\_PORT\_ATTR\_QOS\_DOT1P\_TO\_TC\_MAP、  SAI\_PORT\_ATTR\_QOS\_DOT1P\_TO\_COLOR\_MAP、  SAI\_PORT\_ATTR\_QOS\_DSCP\_TO\_TC\_MAP、  SAI\_PORT\_ATTR\_QOS\_DSCP\_TO\_COLOR\_MAP、  SAI\_PORT\_ATTR\_QOS\_TC\_TO\_QUEUE\_MAP、  SAI\_PORT\_ATTR\_QOS\_TC\_AND\_COLOR\_TO\_DOT1P\_MAP、  SAI\_PORT\_ATTR\_QOS\_TC\_AND\_COLOR\_TO\_DSCP\_MAP、 | oid  绑定的映射表id |

### ****优先级映射****

优先级映射用于在入口的时候将报文的vlan pri或tos(dscp)映射成phb+color，在出口的时候，将phb+color映射成报文的vlan pri或tos(dscp)。表1 华为NE40E的缺省dscp到phb+color的映射表



表2 华为NE40E的缺省phb+color到dscp的映射表



# 四、流量监管与流量整形

### ****两种令牌桶算法****

IETF建议采用srTCM（A Single Rate Three Color Marker，RFC2697）算法或trTCM（A Two Rate Three Color Marker，RFC2698）算法对流量进行测评，根据评估结果为报文打颜色标记，即绿色、黄色和红色。

srTCM与trTCM算法均采用两个令牌桶对到达的报文进行评估，他们允许流量在某种级别上突发——但关注的侧重点不同，srTCM更关注报文尺寸的突发，trTCM则关注速率上的突发。srTCM与trTCM算法有两种工作模式，色盲模式（Color-Blind）与色感知模式（Color-Aware），其中色盲模式是较常用的。

* + **srTCM**

RFC 2697 srTCM 单速率三色标记指定3个参数：承诺信息速率CIR 、承诺突发尺寸CBS 和超额突发尺寸EBS。

为方便，将两个令牌桶称为C桶和E桶，用Tc和Te表示桶中的令牌数量，Tc和Te初始化等于CBS和EBS。

Tc和Te在每秒钟内更新CIR次，每次更新时遵循以下规则:

如果Tc < CBS，则Tc增加1，否则

如果Te < EBS，则Te增加1，否则

Tc和Te都不增加

色盲模式下，在对到达报文（假设报文大小为B）进行评估时，遵循以下规则：

如果Tc-B >= 0，则报文被标记为绿色，且Tc降低B，否则

如果Te-B >= 0，则报文被标记为黄色，且Te降低B，否则

报文被标记为红色且Tc和Te都不降低。

非色盲模式下，在对到达报文（假设报文大小为B）进行评估时，遵循以下规则：

如果报文已被标记为绿色且Tc-B >= 0，则报文被标记为绿色，且Tc降低B，否则

如果报文已被标记为绿色或黄色且Te-B >= 0，则报文被标记为黄色，且Te降低B，否则报文被标记为红色且Tc和Te都不降低。

ebs

Te

cbs

Tc

* + **trTCM**

RFC 2698 trTCM算法 双速率三色标记指定4个参数：承诺信息速率CIR 、峰值信息速率PIR、承诺突发尺寸CBS 和峰值突发尺寸PBS。

trTCM算法中两个令牌桶的填充令牌的速率不同，分别为承诺的平均速率CIR（Committed Information Rate）和峰值速率PIR（Peak Information Rate）。

为方便将这两个令牌桶称为C桶和P桶，这两个桶的尺寸分别为承诺突发尺寸CBS（Committed Burst Size）和峰值突发尺寸PBS（Peak Burst Size）。

用Tc和Tp表示桶中的令牌数量，Tc和Tp初始化等于CBS和PBS。Tc和Tp在每秒钟内分别更新CIR和PIR次，每次更新增加一个令牌（除非桶满）。

在色盲模式下，在对到达报文（假设报文大小为B）进行评估时，遵循以下规则：

如果Tp-B < 0，则报文被标记为红色，否则

如果Tc-B < 0，则报文被标记为黄色，且Tp降低B，否则

报文被标记为绿色且Tc和Tp都降低B。

在非色盲模式下，在对到达报文（假设报文大小为B）进行评估时，遵循以下规则：

如果报文已被标记为红色或者Tp-B < 0，则报文被标记为红色，否则

如果报文已被标记为黄色或者Tc-B < 0，则报文被标记为黄色，且Tp降低B，否则

报文被标记为绿色且Tc和Tp都降低B。

pbs

cbs

Tc

Tp

### ****流量监管****

从“流量监管”（TrafficPolicing）就是对流量进行控制，通过监督进入交换机端口的流量速率，对超出部分的流量进行丢弃或重新分配优先级。

### ****流量整形****

“流量整形”为控制最大输出通信速率提供可能，以确保通信符合配置的最大传输速率规定。符合某种配置的通信可能被整形，以使它符合下游设备的通信速率需求，处理任何失配的数据传输速率。流量整形通常使用缓存区和令牌桶来完成，当报文的发送速率过快时，首先在缓冲区进行缓存，在令牌桶的控制下再均匀地发送这些被缓冲的报文。

当下游设备的接口速率小于上游设备的端口速率或发生突发流量时，在下游设备入端口处可能出现流量拥塞的情况。此时用户可以通过在上游设备的出端口配置流量整形，将上游不规整的流量进行削峰填谷，输出一条比较平整的流量，从而解决下游设备的拥塞问题。

### ****流量整形和流量监管的区别****

流量整形和流量监管都是作用于网络边缘，对进入设备端口的流量进行的一种处理方式。它们的主要区别在于：流量监管直接丢弃不符合速率要求的报文，丢弃的报文比较多，可能引发重传；而流量整形是将不符合速率要求的报文先行缓存，当令牌桶有足够的令牌时再均匀地向外发送这些被缓存的报文，较少丢弃报文，但引入时延和抖动，需要较多的缓存资源缓存报文。所以这两种功能的应用领域也不尽相同，流量监管适用于对丢弃率不敏感，而对时延和抖动比较敏感的网络应用，如一些普通的话音和视频通信；流量整形适用于对时延和抖动不敏感的网络应用，如数据传输、WWW访问等。

# 五、拥塞管理与拥塞避免

### ****拥塞管理****

拥塞管理是指在网络间歇性出现拥塞，时延敏感业务要求得到比其它业务更高质量的QoS服务时，通过调整报文的调度次序来满足时延敏感业务高QoS服务的一种流量控制机制。

* + **PQ调度**

PQ（Priority Queuing）调度，也叫SP(Strict Priority)就是严格按照队列优先级的高低顺序进行调度。只有高优先级队列中的报文全部调度完毕后，低优先级队列才有调度机会。

采用PQ调度方式，将延迟敏感的关键业务放入高优先级队列，将非关键业务放入低优先级队列，从而确保关键业务被优先发送。

PQ调度的缺点是：拥塞发生时，如果较高优先级队列中长时间被占用，那么低优先级队列中的报文就会得不到调度机会。

* + **WRR调度**

WRR（Weighted Round Robin）调度即加权轮询调度。WRR在队列之间进行轮流调度，保证每个队列都得到一定的服务时间。

以端口有8个输出队列为例，WRR可为每个队列配置一个加权值（依次为w7、w6、w5、w4、w3、w2、w1、w0），加权值表示获取资源的比重。例如：一个100M的端口，配置它的WRR队列调度算法的加权值为50、50、30、30、10、10、10、10（依次对应w7、w6、w5、w4、w3、w2、w1、w0），这样可以保证最低优先级队列至少获得5Mbit/s带宽，避免了采用PQ调度时低优先级队列中的报文可能长时间得不到服务的缺点。

* + **DWRR调度**

DWRR（Deficit Weighted Round Robin）调度即差分加权轮询，实现原理与WRR调度基本相同。

DWRR与WRR的区别是：WRR调度是按照报文个数进行调度，而DWRR是按照报文长度进行调度。

* + **混合调度**

PQ调度和WRR/DWRR调度各有优缺点。单纯采用PQ调度时，低优先级队列中的报文可能长期得不到调度，而单纯采用WRR/DWRR调度时低延时需求业务得不到优先调度，“PQ+WRR/PQ+DWRR”调度方式则将两种调度方式结合起来。按照标准，所有所有WRR或DWRR的队列优先级应小于所有PQ队列的优先级。

### ****拥塞避免****

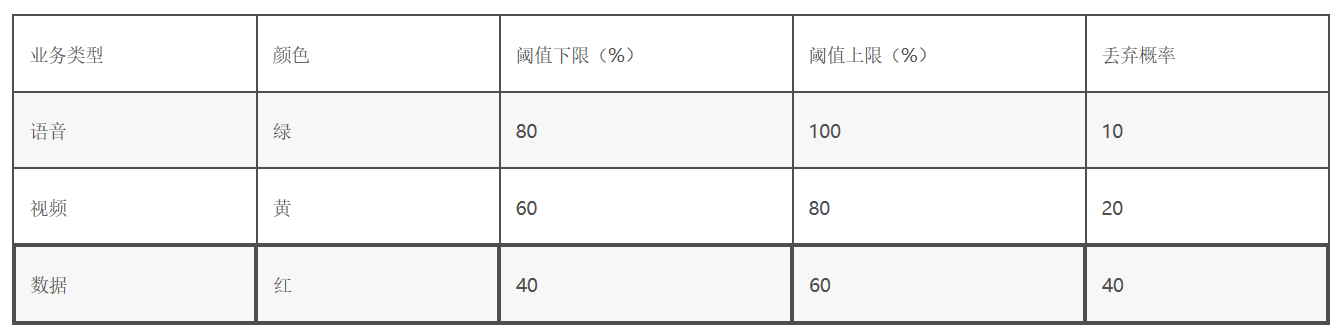
为了解决网络拥塞，可以通过拥塞避免在网络出现拥塞时主动丢弃一些报文，解除网络过载；为了使用户得到更好的服务质量，可以通过拥塞管理对关键业务优先调度，使得这些业务得到更高的QoS服务。

* + **1、尾部丢弃**

传统的丢包策略采用尾部丢弃的方法，且是同等的对待所有报文，不区分报文的服务等级。这时在拥塞发生期间，队列尾部的数据报文将被丢弃，直到拥塞解除。

* + **2、WRED**

在网络发生拥塞的时候，默认的是尾丢弃，我们不希望这样，我们总是希望先丢弃优先级低的数据包，而保证优先级高的数据包的发送，WRED是基于weight的随机早侦测，依靠流量的优先级来分配相应的丢弃几率的。当网络中有多种数据时，在发生拥塞之后，人们总是希望先将优先级较低的相对不重要的数据丢弃，而优先保证重要数据的传递。WRED正是迎合了人们的这种期望，在网络发生拥塞之后，总是先保证高优先级的重要数据的传递，而先丢弃普通的数据。 WRED在网络发生拥塞之后，可根据数据包的DSCP或IP优先级来丢弃数据包，低优先级的数据总是比高优先级的数据先丢，从而保证重要数据的传递。举一个具体的wred配置实例：

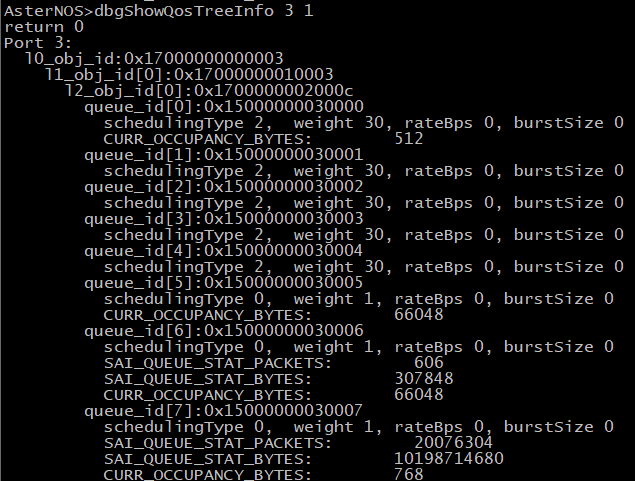


# 六、QOS调度相关SAI接口

qos调度的配置是QUEUE对SCHEDULER和WRED\_PROFILE这两个profile的引用，以下json文件展示的比较清晰



很多网络设备配置wred阈值上限和下限都是通过百分比配置的，SONiC是通过配置buffer的字节数实现的，并不是太友好。针对这个情况，可有通过进行实际打流，用调试命令dbgShowQosTreeInfo查看各队列的buffer占用情况。比如：3个10G口打一个10G口，这个3个10G口的流量的优先级分别是7、6、5，出口队列配置成SP的调度方式，不开wred。那么理论上只有优先级7的流量能通过，优先级6和5的buffer会占用的最大。用调试命令查看计数如下：



### ****SCHEDULER****

接口名：create\_scheduler

功能： 创建调度profile

参数1：sai\_object\_id\_t \*scheduler\_id 返回映射表id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_SCHEDULER\_ATTR\_SCHEDULING\_TYPE | s32  SAI\_SCHEDULING\_TYPE\_STRICT/SAI\_SCHEDULING\_TYPE\_DWRR  选择严格优先级或权重调度方式 |
| SAI\_SCHEDULER\_ATTR\_SCHEDULING\_WEIGHT | u8  DWRR的权重值 |
| SAI\_SCHEDULER\_ATTR\_MAX\_BANDWIDTH\_RATE | u64  队列限速值，单位Byte/s（xp70的最小粒度为10Mb/s） |
| SAI\_SCHEDULER\_ATTR\_MAX\_BANDWIDTH\_BURST\_RATE | s32  队列限速令牌桶大小 |

### ****WRED\_PROFILE****

接口名：create\_wred

功能： 创建wred profile

参数1：sai\_object\_id\_t \*wred\_id 返回wred\_id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_WRED\_ATTR\_GREEN\_ENABLE | booldata 绿包wred使能 |
| SAI\_WRED\_ATTR\_GREEN\_MIN\_THRESHOLD | u32 绿包随机丢弃低门限，单位字节 |
| SAI\_WRED\_ATTR\_GREEN\_MAX\_THRESHOLD | u32 绿包随机丢弃高门限 |
| SAI\_WRED\_ATTR\_GREEN\_DROP\_PROBABILITY | u32 绿包随机丢弃最高百分比，取值0~100 |
| SAI\_WRED\_ATTR\_YELLOW\_ENABLE | booldata 黄包wred使能 |
| SAI\_WRED\_ATTR\_YELLOW\_MIN\_THRESHOLD | u32 黄包随机丢弃低门限 |
| SAI\_WRED\_ATTR\_YELLOW\_MAX\_THRESHOLD | u32 黄包随机丢弃高门限 |
| SAI\_WRED\_ATTR\_YELLOW\_DROP\_PROBABILITY | u32 黄包随机丢弃最高百分比，取值0~100 |
| SAI\_WRED\_ATTR\_RED\_ENABLE | booldata 红包wred使能 |
| SAI\_WRED\_ATTR\_RED\_MIN\_THRESHOLD | u32 红包随机丢弃低门限 |
| SAI\_WRED\_ATTR\_RED\_MAX\_THRESHOLD | u32 红包随机丢弃高门限 |
| SAI\_WRED\_ATTR\_RED\_DROP\_PROBABILITY | u32 红包随机丢弃最高百分比，取值0~100 |
| SAI\_WRED\_ATTR\_WEIGHT | u8 平均队列长度算法滤波系数。值越大，取的样本越大，平均队列长度曲线越平缓；值越小，更能反应短期的波动，平均队列长度曲线会更剧烈。取值0~15 |
| SAI\_WRED\_ATTR\_ECN\_MARK\_MODE | sai\_ecn\_mark\_mode\_t  明确拥塞通告模式 |

typedef enum \_sai\_ecn\_mark\_mode\_t

{

/\*\* Disable ECN marking for all colors \*/

SAI\_ECN\_MARK\_MODE\_NONE,

/\*\* Enable ECN marking for green color. Yellow and red are disabled \*/

SAI\_ECN\_MARK\_MODE\_GREEN,

/\*\* Enable ECN marking for yellow color. Green and red are disabled \*/

SAI\_ECN\_MARK\_MODE\_YELLOW,

/\*\* Enable ECN marking for red color. Green and yellow are disabled \*/

SAI\_ECN\_MARK\_MODE\_RED,

/\*\* Enable ECN marking for green and yellow colors. Red is disabled \*/

SAI\_ECN\_MARK\_MODE\_GREEN\_YELLOW,

/\*\* Enable ECN marking for green and red colors. Yellow is disabled \*/

SAI\_ECN\_MARK\_MODE\_GREEN\_RED,

/\*\* Enable ECN marking for yellow and red colors. Green is disabled \*/

SAI\_ECN\_MARK\_MODE\_YELLOW\_RED,

/\*\* Enable ECN marking for all colors \*/

SAI\_ECN\_MARK\_MODE\_ALL,

} sai\_ecn\_mark\_mode\_t;

### ****QUEUE****

* + **queue与scheduler profile或wred profile的绑定**

接口名：set\_queue\_attribute

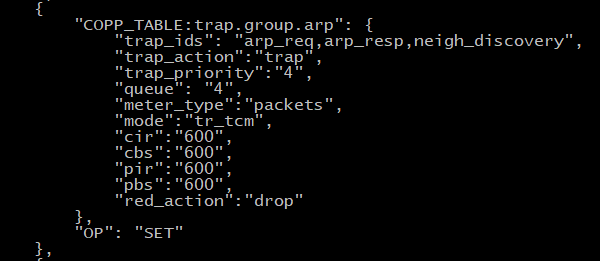
参数1：sai\_object\_id\_t queue\_id

参数2：const sai\_attribute\_t \*attr 见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_QUEUE\_ATTR\_SCHEDULER\_PROFILE\_ID | oid  scheduler profile id |
| SAI\_QUEUE\_ATTR\_WRED\_PROFILE\_ID | oid  wred profile id |

# COPP

Control Plane Policing (CoPP)被称为控制面板策略，控制面板策略这个特性让用户通过配置配置qos策略来控制上送cpu的数据包，从而保护cpu免受攻击。



### ****相关SAI接口****

* + **创建host interface trap group**

接口名：create\_hostif\_trap\_group

参数1：ai\_object\_id\_t \*hostif\_trap\_group\_id 返回申请的id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list attr的列表，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_HOSTIF\_TRAP\_GROUP\_ATTR\_QUEUE | u32  绑定的cpu queue(xdk对queue的范围没有判断) |

* + **创建policer**

接口名：create\_policer

参数1：sai\_object\_id\_t \*policer\_id 返回申请的id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list attr的列表，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_POLICER\_ATTR\_METER\_TYPE | s32  SAI\_METER\_TYPE\_PACKETS  SAI\_METER\_TYPE\_BYTES |
| SAI\_POLICER\_ATTR\_MODE | s32  SAI\_POLICER\_MODE\_SR\_TCM  SAI\_POLICER\_MODE\_TR\_TCM  目前xdk只支持TR\_TCM |
| SAI\_POLICER\_ATTR\_COLOR\_SOURCE | s32  SAI\_POLICER\_COLOR\_SOURCE\_AWARE  SAI\_POLICER\_COLOR\_SOURCE\_BLIND |
| SAI\_POLICER\_ATTR\_CBS | u64 CBS |
| SAI\_POLICER\_ATTR\_CIR | u64 CIR |
| SAI\_POLICER\_ATTR\_PBS | u64 PBS |
| SAI\_POLICER\_ATTR\_PIR | u64 PIR |
| SAI\_POLICER\_ATTR\_GREEN\_PACKET\_ACTION | s32  SAI\_PACKET\_ACTION\_DROP  SAI\_PACKET\_ACTION\_TRAP(默认) |
| SAI\_POLICER\_ATTR\_YELLOW\_PACKET\_ACTION | s32  SAI\_PACKET\_ACTION\_DROP  SAI\_PACKET\_ACTION\_TRAP(默认) |
| SAI\_POLICER\_ATTR\_RED\_PACKET\_ACTION | s32  SAI\_PACKET\_ACTION\_DROP  SAI\_PACKET\_ACTION\_TRAP(默认) |

注：当SAI\_POLICER\_ATTR\_METER\_TYPE为SAI\_METER\_TYPE\_PACKETS时，cir、pir单位为packets/s cbs、pbs单位为packets，当SAI\_POLICER\_ATTR\_METER\_TYPE为SAI\_METER\_TYPE\_BYTES时，cir、pir的单位为bytes/s，cbs、pbs单位为bytes

* + **trap group与policer的绑定**

接口名：set\_hostif\_trap\_group\_attribute

参数1：sai\_object\_id\_t hostif\_trap\_group\_id ，创建hostif\_trap\_group时返回的id

参数2：const sai\_attribute\_t \*attr 属性

|  |  |
| --- | --- |
| Id | Value |
| SAI\_HOSTIF\_TRAP\_GROUP\_ATTR\_POLICER | oid  policer\_id |

* + **创建trap，并与trap group绑定**

接口名：xpSaiCreateHostInterfaceTrap

参数1：sai\_object\_id\_t \*hostif\_trap\_id ，创建hostif\_trapp时返回的id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_HOSTIF\_TRAP\_ATTR\_TRAP\_GROUP | oid  Trap group id |
| SAI\_HOSTIF\_TRAP\_ATTR\_TRAP\_TYPE | s32  SAI\_HOSTIF\_TRAP\_TYPE\_BGP、SAI\_HOSTIF\_TRAP\_TYPE\_ARP\_REQUEST等  见trap\_id\_map |
| SAI\_HOSTIF\_TRAP\_ATTR\_PACKET\_ACTION | s32  SAI\_PACKET\_ACTION\_TRAP |
| SAI\_HOSTIF\_TRAP\_ATTR\_TRAP\_PRIORITY | u32  优先级(目前xdk没有处理) |

* + **创建hostif\_table\_entry,并与trap绑定**

接口名：create\_hostif\_table\_entry

参数1：ai\_object\_id\_t \*hif\_table\_entry\_id 返回hostif\_table\_entry表id

参数2：sai\_object\_id\_t switch\_id

参数3：uint32\_t attr\_count attr个数

参数4：const sai\_attribute\_t \*attr\_list，见下表

|  |  |
| --- | --- |
| Id | Value |
| SAI\_HOSTIF\_TABLE\_ENTRY\_ATTR\_TRAP\_ID | oid  hostif\_trap\_id |
| SAI\_HOSTIF\_TABLE\_ENTRY\_ATTR\_TYPE | s32  SAI\_HOSTIF\_TABLE\_ENTRY\_TYPE\_TRAP\_ID |
| SAI\_HOSTIF\_TABLE\_ENTRY\_ATTR\_CHANNEL\_TYPE | s32  SAI\_HOSTIF\_TABLE\_ENTRY\_CHANNEL\_TYPE\_NETDEV\_PHYSICAL\_PORT |