The bridge to possible

What QoS can do for your network with Catalyst 8000 and other IOS XE routers

David Roten
TME, Technical Leader

BRKENT-2731

Cisco Webex App

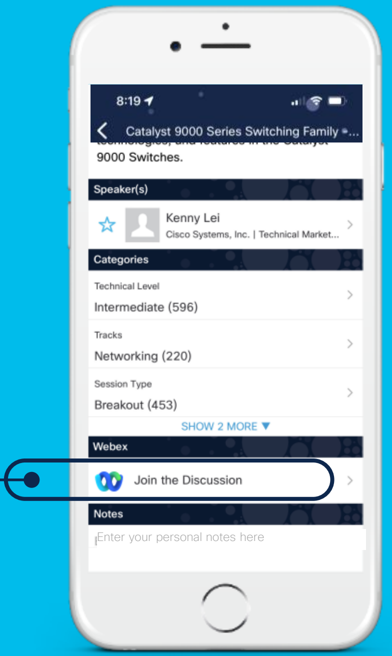
Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated
until February 24, 2023.



8:19

Catalyst 9000 Series Switching Family ...

9000 Switches.

Speaker(s)

Kenny Lei
Cisco Systems, Inc. | Technical Market...

Categories

Technical Level
Intermediate (596)

Tracks
Networking (220)

Session Type
Breakout (453)

SHOW 2 MORE ▼

Webex

Join the Discussion

Notes

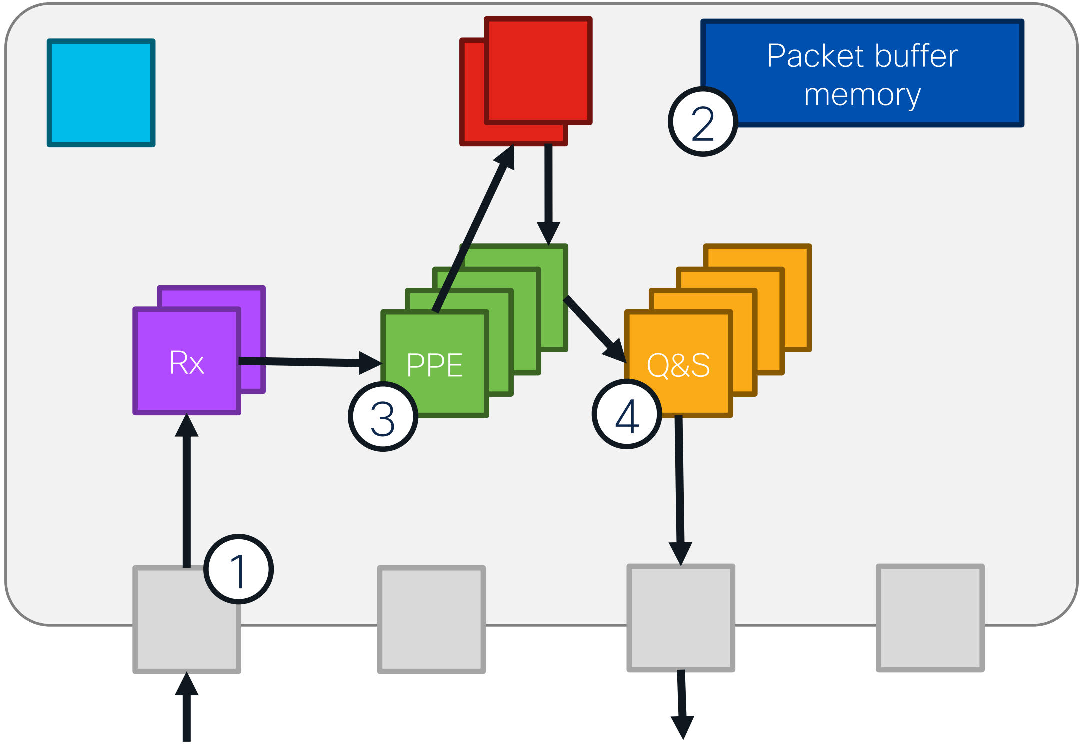
Enter your personal notes here

Agenda

- Platform QoS implementation
- IOS XE routing QoS capabilities
  - 3 parameters
  - Queue-limits
  - Aggregate Etherchannel
  - PAK_PRI
  - Service Fragments
  - Service-groups
  - Tunnel QoS
- Troublehsooting
- Conclusion

Platform QoS Implementation

Generalized IOS XE router datapath



🟦 Control plane

🟪 Receive function

🟩 Data plane

🟧 Queuing & scheduling

🟥 Crypto

⬜ Physical interface

🟦 Packet buffer memory

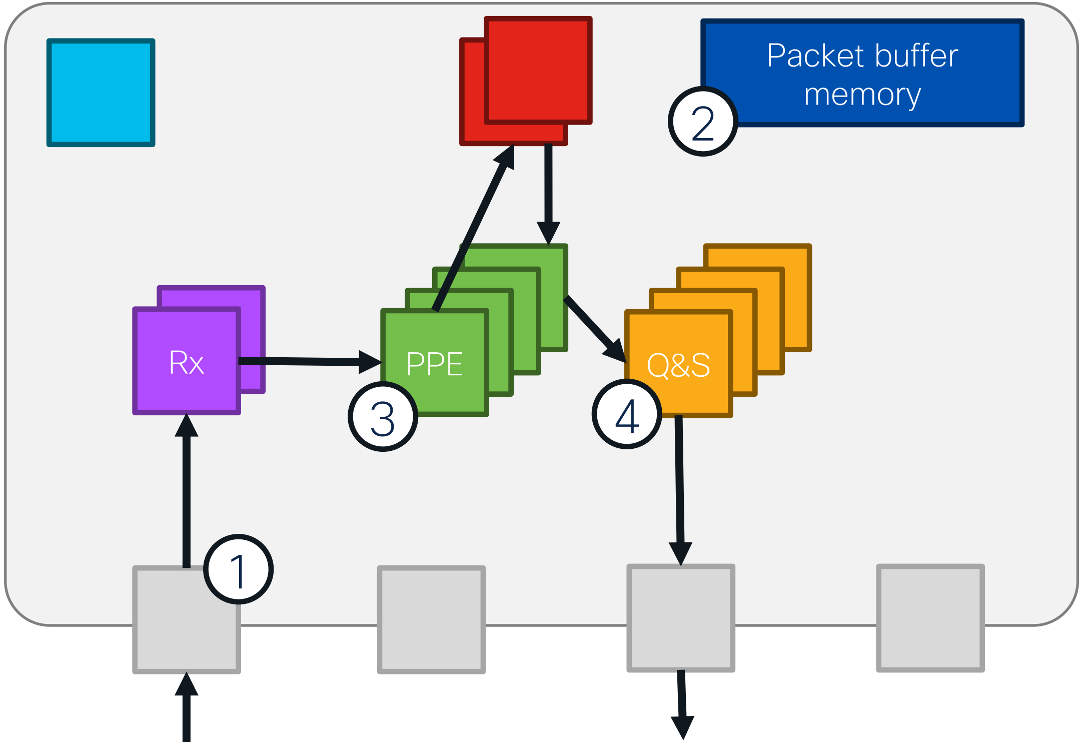
① Initial prioritization

② Store packets during processing

③ Non-queuing functions

④ Queuing functions

Generalized IOS XE router datapath



🟩 Data plane

🟧 Queuing & scheduling

⬜ Physical interface

🟦 Packet buffer memory

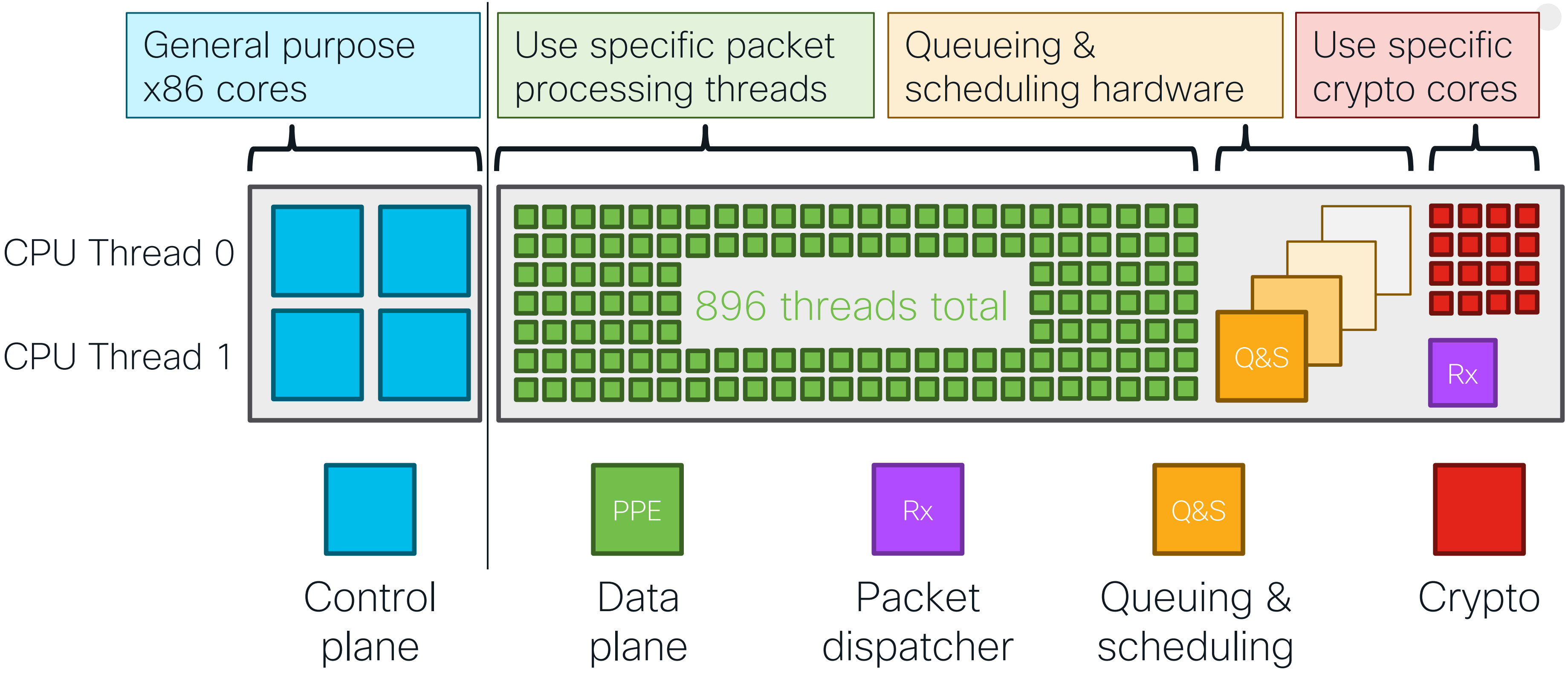
1 ASR1000 / C8500 initial prioritization

2 Store packets during processing

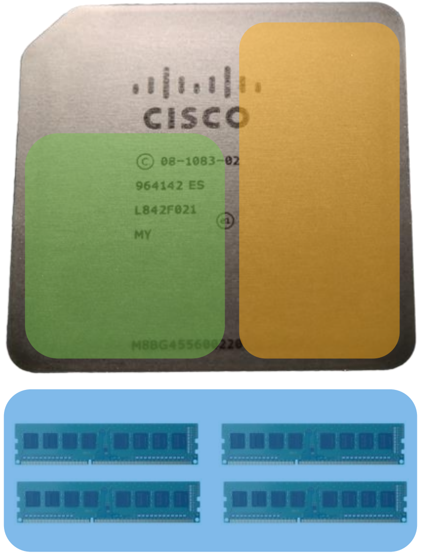
3 Classify Mark Police WRED

4 Priority Shape Bandwidth Bandwidth remaining Fair queuing

Core distribution – C8500-12X

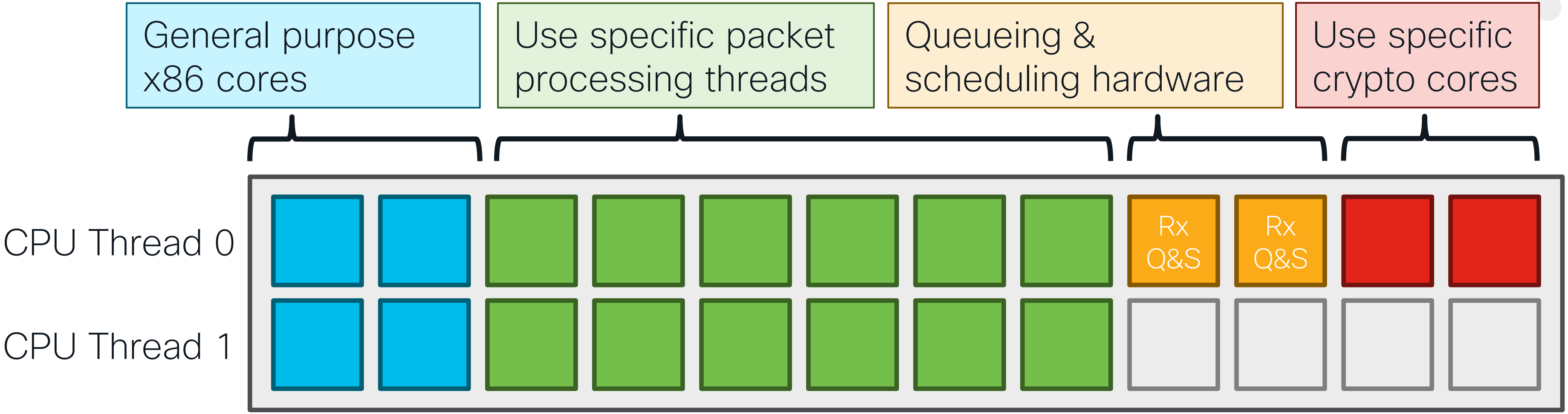


QoS functions on QFP platform architecture



- Programable packet processing engines (PPE) handle all non-queuing functions (224 cores with total of 896 threads)
  - Software based feature
- Dedicated queuing hardware handles queuing and scheduling of all traffic through the ASIC
  - ASIC hardware execution, not software or programable PPEs
- Dedicated packet buffer memory stores packets during PPE processing and scheduling

Core distribution – C8500L-8S4X – DP heavy



	General purpose x86 cores		Use specific packet processing threads						Queueing & scheduling hardware		Use specific crypto cores	
CPU Thread 0	🟦	🟦	🟩	🟩	🟩	🟩	🟩	🟩	🟧 Rx Q&S	🟧 Rx Q&S	🟥	🟥
CPU Thread 1	🟦	🟦	🟩	🟩	🟩	🟩	🟩	🟩	⬜	⬜	⬜	⬜

General purpose x86 cores

Rx + Q&S on shared cores

🟦 Control plane

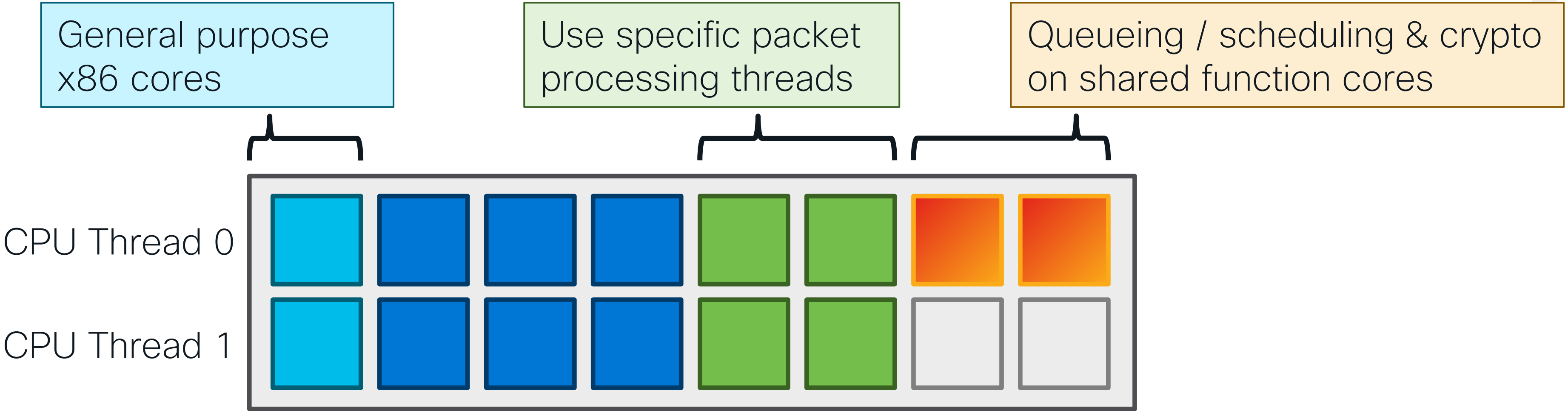
🟩 Data plane

🟧 **Rx** with queuing & scheduling

🟥 Crypto

⬜ Idle

Core distribution – C8300-1N1S-4T2X – SP heavy



General purpose x86 cores

Use specific packet processing threads

Queueing / scheduling & crypto on shared function cores

General purpose x86 cores

Rx + Q&S + crypto on shared cores

🟦 Control plane

🟩 Data plane

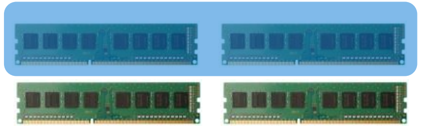
🟦 Service plane

🟧 Rx + Queue/Schedule + Crypto

⬜ Idle

QoS functions x86 platform architecture

- General cores do all non-queuing functions (data plane core count varies between platforms)
  - Same source code that runs on the QFP platforms
- General cores handle queuing and scheduling of all traffic through the ASIC
  - Software emulates the QFP hardware scheduler



- Shared memory pool has packet buffer memory and stores packets during PPE processing and scheduling

IOS XE routing QoS capabilities

3 parameter scheduling behavior

IOS XE MQC based QoS – 3 parameter scheduler

- IOS XE provides an advanced 3 parameter scheduler
  - Minimum - `bandwidth`
  - Excess - `bandwidth remaining`
  - Maximum - `shape`
- 3 parameter schedulers share excess bandwidth equally in default configuration
- bandwidth and bandwidth remaining may not be configured in the same policy-map

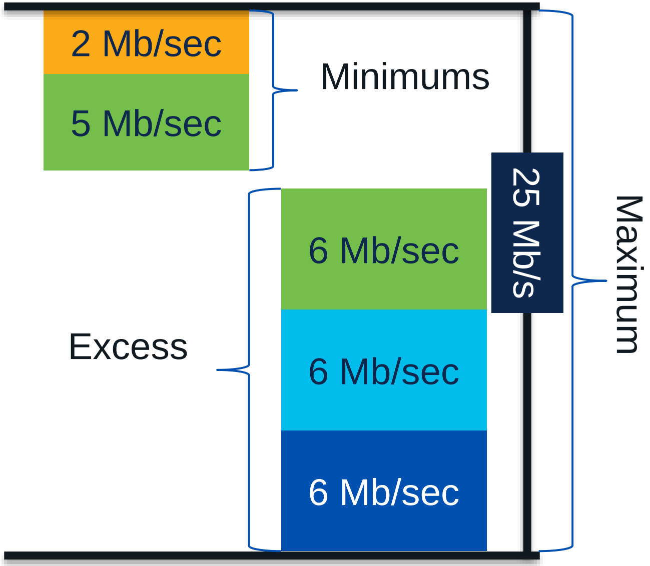
What are the three parameters?

```
policy-map child
  class voice
    priority level 1
    police cir 2000000  (bit/sec)
  class critical_services
    bandwidth 5000
  class internal_services
    shape average percent 100
  class class-default
!
policy-map parent
  class class-default
    shape average 25000000
    service-policy child
```

Minimum is defined by the **bandwidth** command or priority classes with policers. Classes with these directives are guaranteed to receive at least and maybe more bandwidth.

Excess is defined by the **bandwidth remaining** command. Excess is the amount of bandwidth available once all the minimums guarantees are satisfied. By default, classes have a remaining ratio of 1 even if bandwidth remaining is not configured.

Maximum is implemented by shapers. Traffic rates beyond the shaper rates will be held in queues.

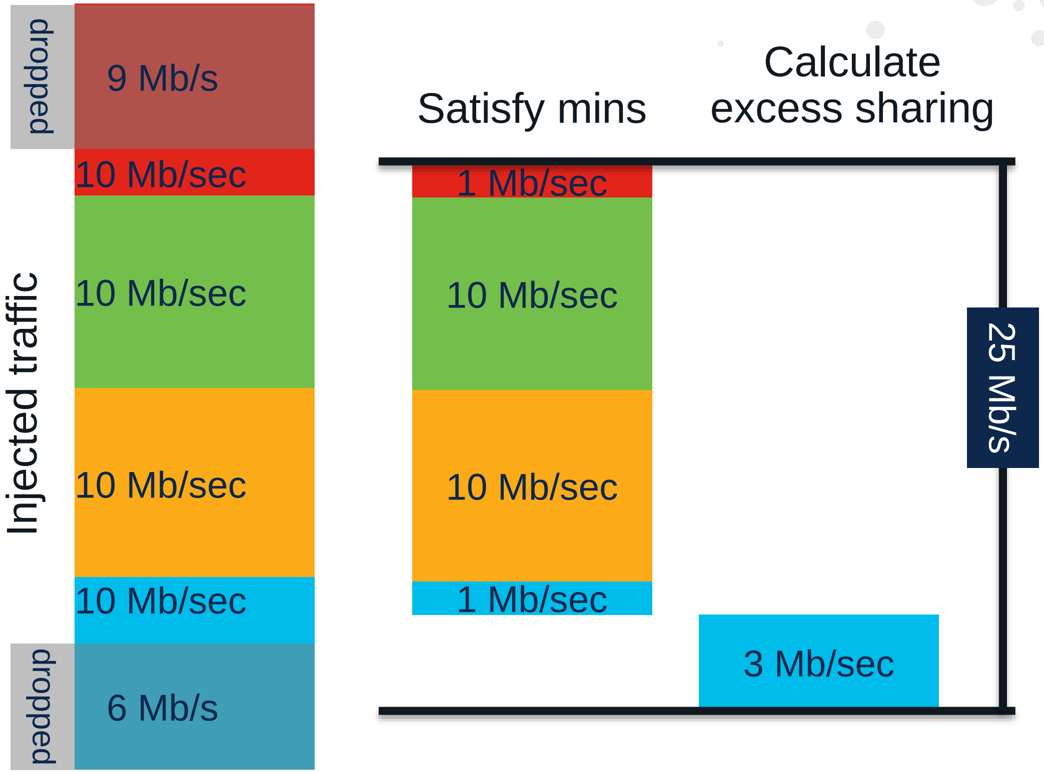


2 ways to manage bandwidth remaining (excess)

- The `bandwidth remaining` (BR) adjust sharing of excess bandwidth
- Two options are available:
  - `bandwidth remaining ratio X`
    where X ranges from 1 to 1000, with variable base
  - `bandwidth remaining percent Y`
    where Y ranges from 1 to 100, with fixed base of 100
- bandwidth remaining percent (BR%) based allocations remain the same as classes are added to a configuration
- bandwidth remaining ratio (BRR) based allocations adjust as more queuing classes are added to a configuration with or without BRR configured
  - base changes as new classes are added with their own ratios defined or with a default of 1
- By default, all classes have a bandwidth remaining ratio or percent value of 1

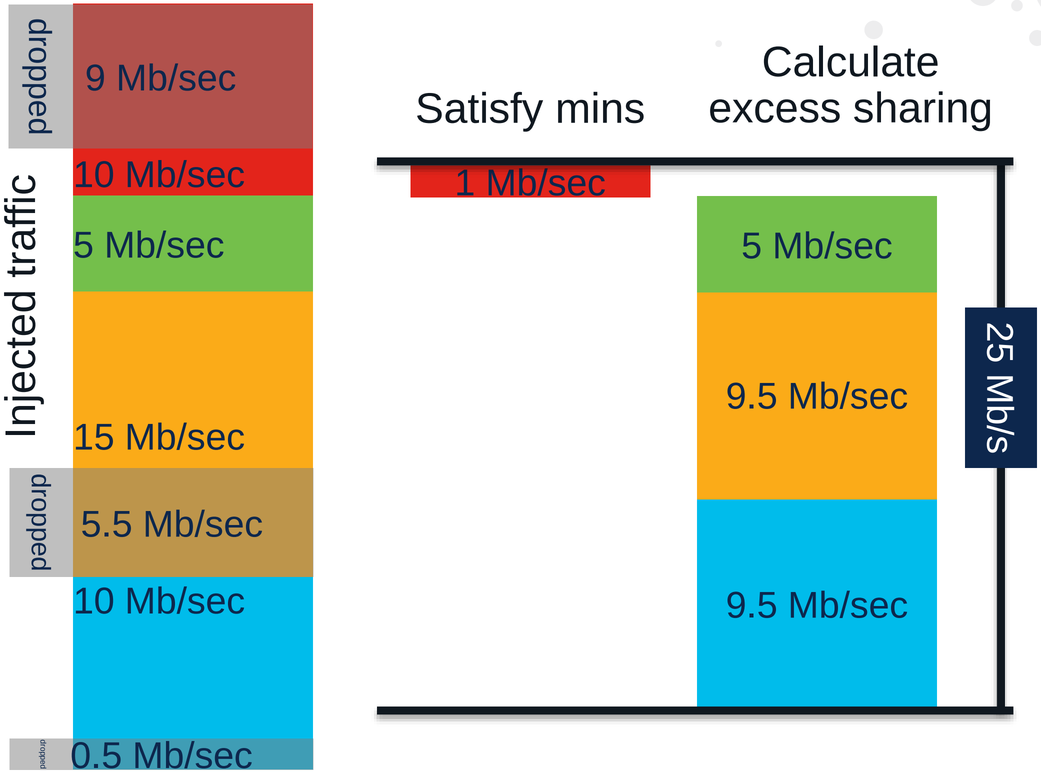
QoS 3 parameter scheduler – minimum

```
policy-map child
  class voice
    priority level 1
    police cir 1000000 (bit/sec)
  class critical_services
    bandwidth 10000 (kbit/sec)
  class internal_services
    bandwidth 10000 (kbit/sec)
  class class-default
    bandwidth 1000 (kbit/sec)
!
policy-map parent
  class class-default
    shape average 25000000
    service-policy child
```



QoS 3 parameter scheduler – excess ratio

```
policy-map child
  class voice
    priority level 1
    police cir 1000000 (bit/sec)
  class critical_services
    bandwidth remaining ratio 4
  class internal_services
    bandwidth remaining ratio 1
  class class-default
    bandwidth remaining ratio 1
!
policy-map parent
  class class-default
    shape average 25000000
    service-policy child
```



dropped

9 Mb/sec

10 Mb/sec

5 Mb/sec

15 Mb/sec

dropped

5.5 Mb/sec

10 Mb/sec

dropped

0.5 Mb/sec

Injected traffic

Satisfy mins

Calculate excess sharing

1 Mb/sec

5 Mb/sec

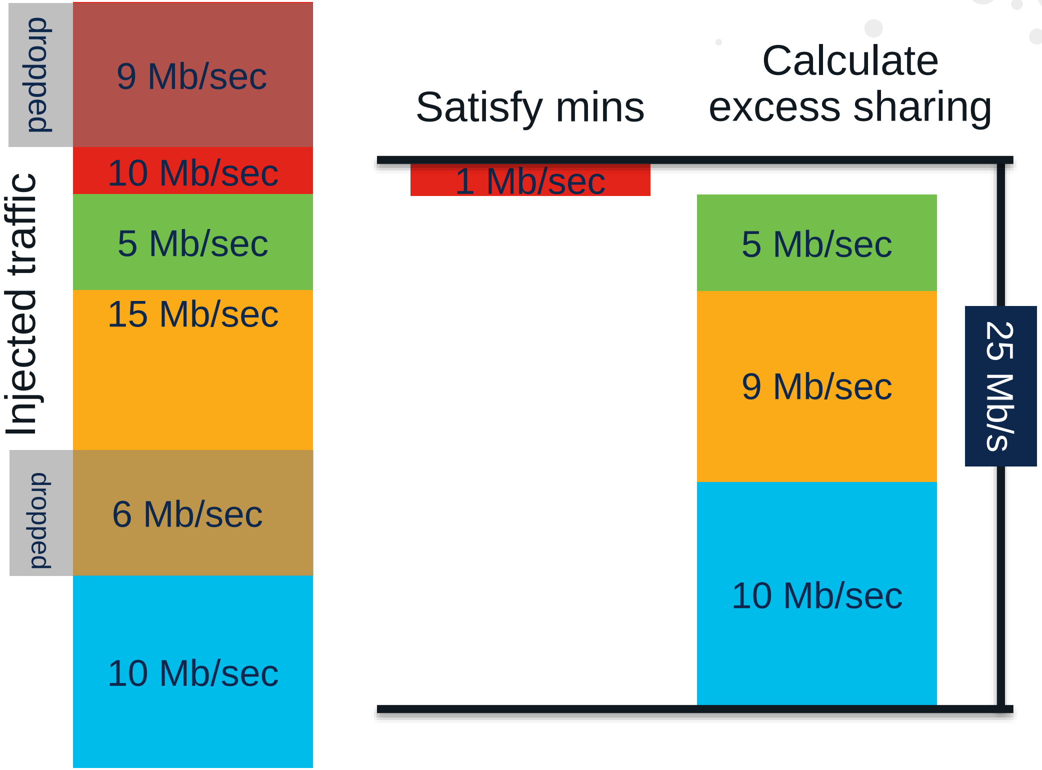
9.5 Mb/sec

9.5 Mb/sec

25 Mb/s

QoS 3 parameter scheduler – excess %

```
policy-map child
  class voice
    priority level 1
    police cir 1000000 (bit/sec)
  class critical_services
    bandwidth remain percent 40
  class internal_services
    bandwidth remain percent 10
  class class-default
!
!
policy-map parent
  class class-default
    shape average 25000000
    service-policy child
```



BR% vs BRR behavior

```
policy-map BR-precent
 class D1
  bandwidth remaining percent 12
 class D2
  bandwidth remaining percent 6
 class D3
  bandwidth remaining percent 2
```

	D1	D2	D3	c-d
BR-percent	12%	6%	2%	80%
BR-ratio	57%	29%	9.5%	4.5%
	${}^{12}/_{21}$	${}^{6}/_{21}$	${}^{2}/_{21}$	${}^{1}/_{21}$

```
policy-map BR-ratio
 class D1
  bandwidth remaining ratio 12
 class D2
  bandwidth remaining ratio 6
 class D3
  bandwidth remaining ratio 2
 class class-default
  bandwidth remaining ratio 1
```

The sum of all values is 21.

20 are defined in class-maps plus 1 from class-default (which can be overridden by the user.)

Queue limit options

Default queue-limit

- IOS XE uses the following defaults:
  - 512 packets for priority queues
  - 50ms of MTU sized packets for all other queues (“`mtu`“ not “`ip mtu`”)
  - strict minimum of 64 packets
- Do not adjust priority queue limits
- IOS XE platforms **ignore** the following buffering parameters for interface QoS
  - interface hold-queue
  - global IOS buffers configuration
  - shaper `bc` and `be` values (policer `bc` and `be` values are recognized and used)

IOS XE – queue limit management

- If `bandwidth` parameter is configured, then that rate is used as the speed value
- If `bandwidth percent` parameter is configured, then that rate based on the percentage from the parent is used as the speed value
- If a `shape` parameter is configured, then that rate is used as the speed value
- If only `bandwidth remaining` is configured, then the parent's speed value is used

$$queue\_limit_{packets} = \frac{speed_{bits/sec} \times 0.050_{sec}}{interface\_mtu_{bytes/packet} \times 8\,\frac{bits}{byte}}$$

IOS XE – queue limit management

```
policy-map child
 class p7
  priority
  police cir percent 2
 class p6
  bandwidth 10000
  shape average 200,000,000
 class p5
  shape average 200,000,000
 class class-default
!
policy-map parent
  class class-default
    shape average 800000000
    service-policy child
```

`priority` classes always have a default queue depth of 512 packets

`bandwidth` classes use the configured bandwidth value to come up with $^{10E6\times 0.050}/_{1500\times 8} = 41 \rightarrow 64\ packets$

`shape` only classes use the configured shape value to come up with $^{200E6\times 0.050}/_{1500\times 8} = 832\ packets$

$$queue\_limit_{packets} = \frac{speed_{bits/sec} \times 0.050_{sec}}{interface\_mtu_{bytes/packet} \times 8\,\frac{bits}{byte}}$$

IOS XE – queue limit management

```
policy-map child
 class p7
  priority
  police cir percent 2
 class p6
  bandwidth remaining ratio 20
  shape aver 30000000
 class p5
  bandwidth remaining ratio 10
 class class-default
!
policy-map parent
  class class-default
    shape average 800000000
    service-policy child
```

`priority` classes always have a default queue depth of 512 packets

Classes with `shape` use the configured shape value as follows: $^{30E6\times0.050}/_{1500\times8} = 125\ packets$

`bandwidth remaining` only classes use the parent shape value 8 Mbit/sec as follows: $^{8E6\times0.050}/_{1500\times8} = 33{,}333\ packets$

$$queue\_limit_{packets} = \frac{speed_{bits/sec} \times 0.050_{sec}}{interface\_mtu_{bytes/packet} \times 8\,\frac{bits}{byte}}$$

Queue-limit options

- IOS XE defaults to **packet based** configs but also supports time and byte based configuration
  - Supported on ASR1000 / C8500 / C8300 / C8200 / ISR1000 / C8000v
- Time and byte based are essentially the same thing
- Time is simply converted into bytes based on the speed of the interface (or parent shaper in a hierarchical policy.)

GigabitEthernet: $0.150\ sec \times \frac{1E9\ bits}{sec} \times \frac{byte}{8\ bits} = 18.75\ Mbytes$

TenGigabitEthernet: $0.150\ sec \times \frac{10E9\ bits}{sec} \times \frac{byte}{8\ bits} = 187.5\ Mbytes$

Queue-limit options

$$0.150\ sec \times \frac{1E9\ bits}{sec} \times \frac{byte}{8\ bits} = 18.75\ Mbytes$$

$$0.150\ sec \times \frac{10E9\ bits}{sec} \times \frac{byte}{8\ bits} = 187.5\ Mbytes$$

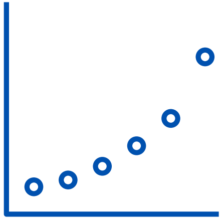
```
policy-map queue-limit
 class class-default
  queue-limit 150 ms
```

```
Rtr#show policy-map int GigabitEthernet0/0/0
 GigabitEthernet0/0/0

  Service-policy output: queue-limit

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any

      queue limit 150 ms/ 18,750,000 bytes
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
```

```
Rtr#show policy-map int TenGigabitEthernet0/1/0
 TenGigabitEthernet0/1/0

  Service-policy output: queue-limit

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any

      queue limit 150 ms/ 187,500,000 bytes
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
```

Why use the time based queue-limits?

- Flexibility to have a single policy-map work for multiple interfaces instead of needing multiple variations of a single policy-map
- Consistent latency profile
  - The latency associated with 4000 packets is much different if those packets are 64 or 1500 bytes
  - With a GigE interface the difference is 2 msec versus 48 msec 64 bytes versus 1500 byte packets

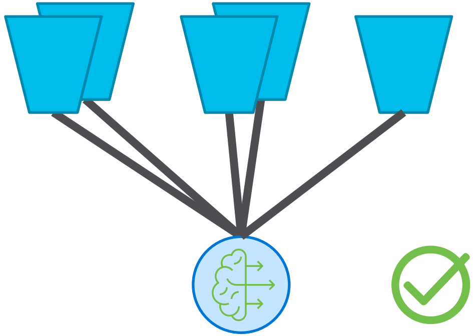


- Restrictions
  - All queue-limit units in a policy and its related hierarchy must be the same units
  - Precludes modification on the fly, must remove, modify and reapply

Aggregate Etherchannel QoS

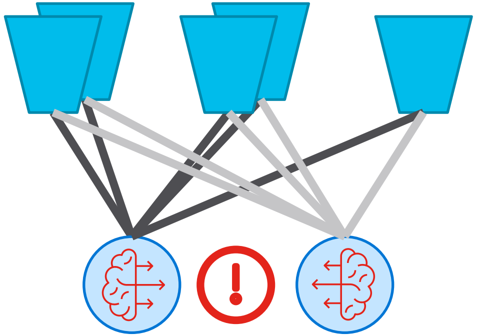
Etherchannel challenges

- Etherchannel does not blend well with QoS
  - Interface bandwidth can change under guarantees
  - IOS XE queuing expects to have the hierarchy rooted onto a physical interface (not multiple interfaces)



QoS is configured at this level

Scheduling decisions at this lower level need visibility to all traffic subjected to config at level above



Neither interface schedule knows about traffic on the other interface for scheduling decisions!

IOS XE Aggregate GEC QoS

- Aggregate GEC QoS creates a new hierarchy that is rooted on a logical recycle interface

```
platform qos port-channel-aggregate <#>
```

After packets run through the aggregate schedule, they are recycled for an abbreviated run through the PPEs for Etherchannel processing. Queued again through internally defined member-link hierarchies.

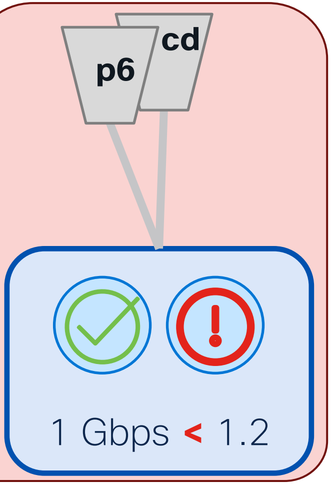
IOS XE Aggregate GEC QoS

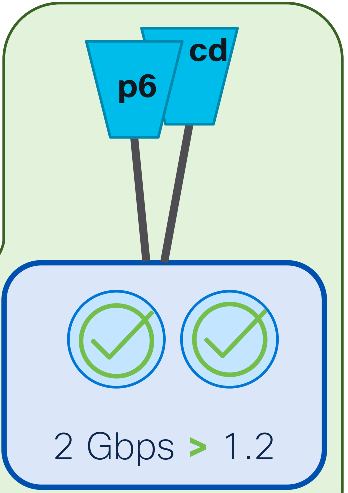
- Policy-maps can be attached to Port-channel main-interface, sub-interfaces, and service-groups on the Port-channel for input and output directions
- No restrictions for contents of the policy-map other than what exist already for GigabitEthernet interfaces
- For example, all of the following are supported:
  - 3 levels of hierarchy (queuing + nonqueuing)
  - 2 levels of priority
  - WRED
  - fair-queue, plus other features

IOS XE Aggregate GEC QoS

```
policy-map WAN
  class p6
    bandwidth 600,000 (Kbps)
  class class-default
    bandwidth 600,000 (Kbps)
```

- If Port-channel does not have enough bandwidth to service all configured minimums, policy-map will go into suspended mode
  - For example, there are 1.2 Gb/sec of minimums and a Port-channel with 2 GigE interfaces has one of the member links go down
  - Once enough member links are available to provide the guaranteed minimums, service-policy will leave suspended mode and go into effect.



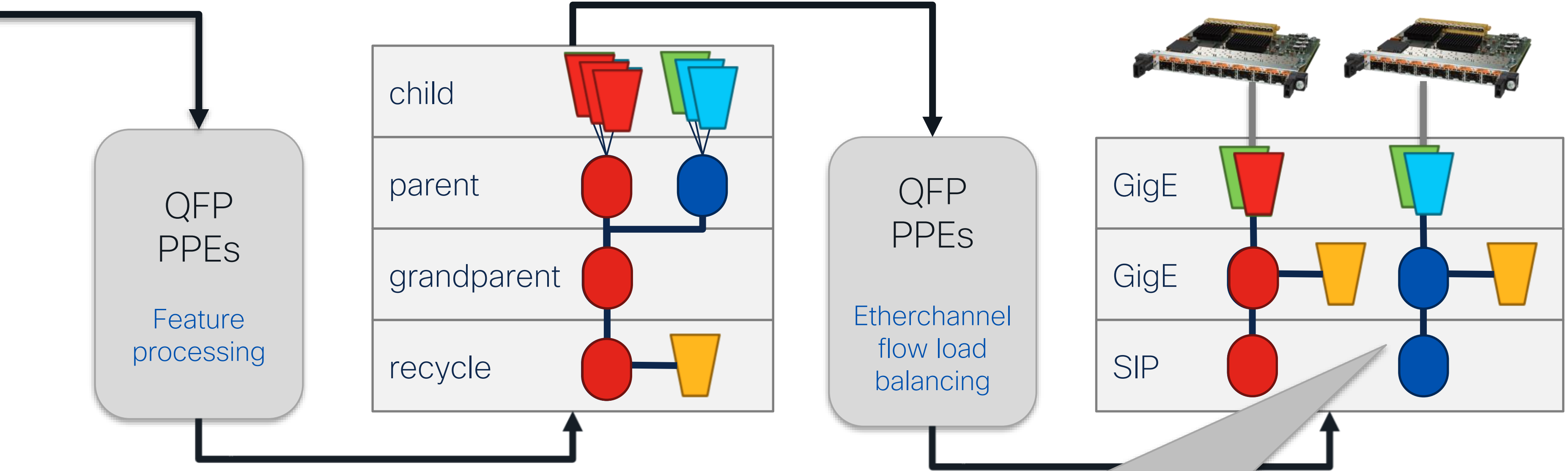


IOS XE Aggregate GEC QoS

- If one of the member links is overdriven, it will exert back pressure on the aggregate hierarchy
  - Flow based load balancing could be lopsided and one of the member links is utilized at 100% and its egress queues fill up
  - When this happens, backpressure is exerted on the aggregate hierarchy and all traffic will be queued even if it is destined for an underutilized member link

Aggregate GEC QoS backpressure

Feedback from Ethernet to grandparent happens fast enough that the grandparent can slow down before Ethernet starts tail dropping.



Available bandwidth, but not available due to recycle queue under backpressure.

Priority versus PAK_PRI

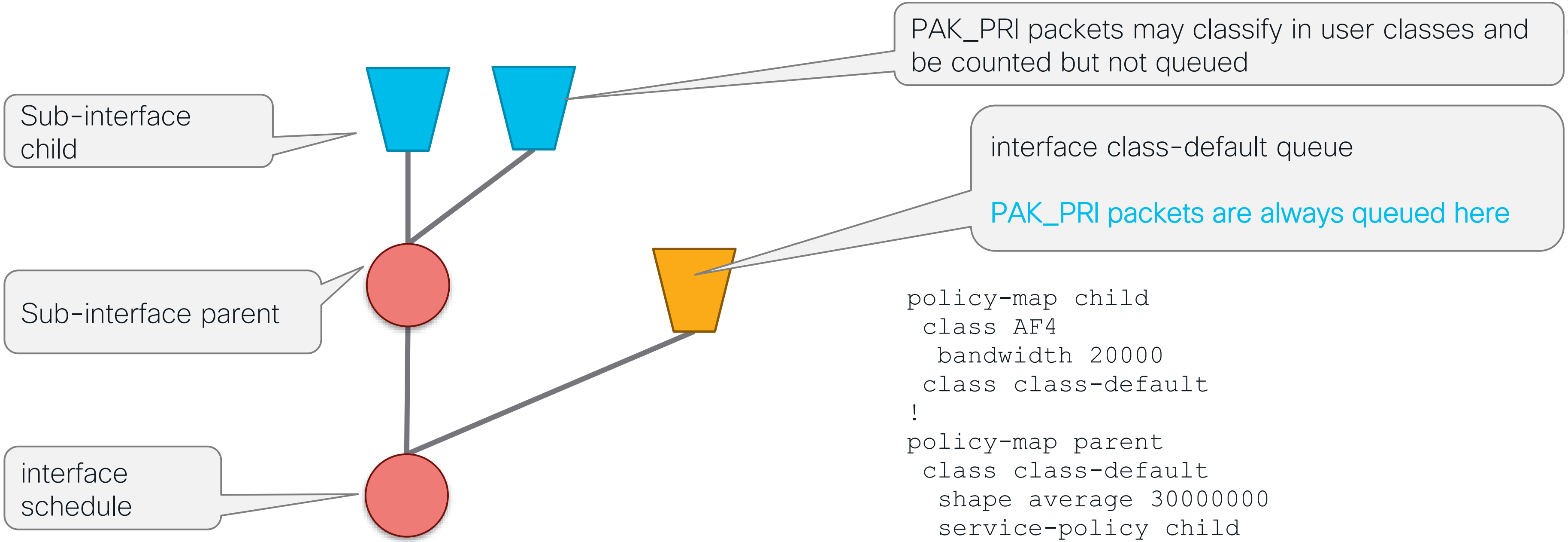
What is PAK_PRI

- Some internally generated packets considered so important that they are always “no drop”
- Typically these are associated with protocols where reliable delivery is considered highly desirable
- Not all packets for a given protocol are considered PAK_PRI

How is PAK_PRI handled

- PAK_PRI packets will show up in QoS class classification stats but will be queued in the interface default queues
  - Classification and queuing stats won’t match
- PAK_PRI packets are never subject to dropping
  - Only if physical packet memory is exhausted will PAK_PRI be dropped
  - 5% of packet memory is reserved for PAK_PRI packet only
- PAK_PRI packets are not treated with LLQ unless classified into that class, and they will actually move through a low latency queue

Interface default queue used for PAK_PRI



Sub-interface child

Sub-interface parent

interface schedule

PAK_PRI packets may classify in user classes and be counted but not queued

interface class-default queue

PAK_PRI packets are always queued here

```
policy-map child
 class AF4
  bandwidth 20000
 class class-default
!
policy-map parent
 class class-default
  shape average 30000000
  service-policy child
!
interface GigabitEthernet0/0/0
!
interface GigabitEthernet0/0/0.2
 service-policy output parent
```

PAK_PRI protocols

Layers 1 and 2

- ATM Address Resolution Protocol Negative Acknowledgement (ARP NAK)
- ATM ARP requests
- ATM host ping operations, administration and management cell(OA&M)
- ATM Interim Local Management Interface (ILMI)
- ATM OA&M
- ATM ARP reply
- Cisco Discovery Protocol
- Dynamic Trunking Protocol (DTP)
- Ethernet loopback packet
- Frame Relay End2End Keepalive
- Frame Relay inverse ARP
- Frame Relay Link Access Procedure (LAPF)
- Frame Relay Local Management Interface (LMI)
- Hot standby Connection-to-Connection Control packets (HCCP)
- High-Level Data Link Control (HDLC) keepalives
- Link Aggregation Control Protocol (LACP) (802.3ad)
- Port Aggregation Protocol (PAgP)
- PPP keepalives
- Link Control Protocol (LCP) Messages
- PPP LZS-DCP
- Serial Line Address Resolution Protocol (SLARP)
- Some Multilink Point-to-Point Protocol (MLPP) control packets (LCP)

IPv4 Layer 3

- Protocol Independent Multicast (PIM) hellos
- Interior Gateway Routing Protocol (IGRP) hellos
- OSPF hellos
- EIGRP hellos
- Intermediate System-to-Intermediate System (IS-IS) hellos, complete sequence number PDU (CSNP), PSNP, and label switched paths (LSPs)
- ISIS hellos
- Triggered Routing Information Protocol (RIP) Ack
- TDP and LDP hellos
- Resource Reservation Protocol (RSVP)
- Some L2TP control packets
- Some L2F control packets
- GRE IP Keepalive
- IGRP CLNS
- Bidirectional Forwarding Protocol (BFD)

IPv6 Layer 3

- Miscellaneous protocols

This list is not considered to be complete nor exhaustive and is subject to change without notice

Service fragments

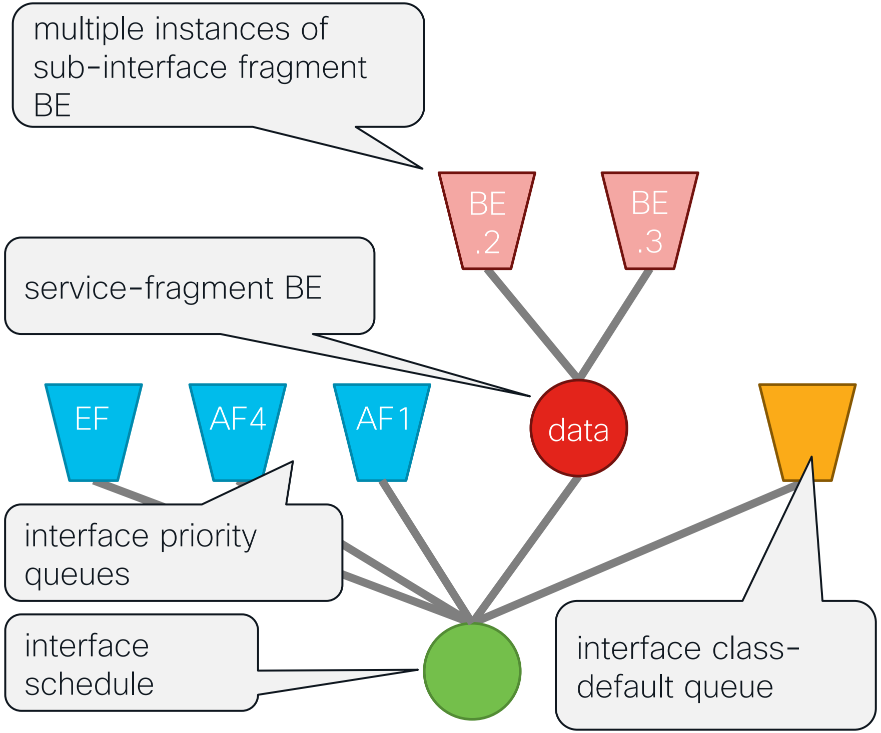
ASR 1000 QoS service-fragments

- Typically hierarchies are very rigid with strict parent – child relationships
- Service-fragments allow queues to be parented by an schedules outside of the strict hierarchy
- Model 4

  Allows queue conservation in scaled broadband configs
- Model 3

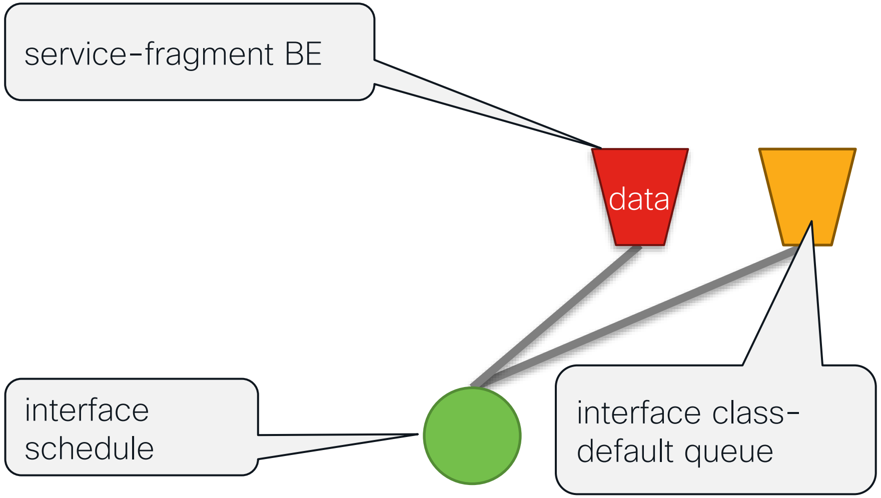
  Allows aggregated shaping of selected traffic with per session / sub-interface prioritization of other traffic outside the session limits

QoS service-fragments model 4


multiple instances of sub-interface fragment BE
BE .2
BE .3
service-fragment BE
EF
AF4
AF1
data
interface priority queues
interface schedule
interface class-default queue

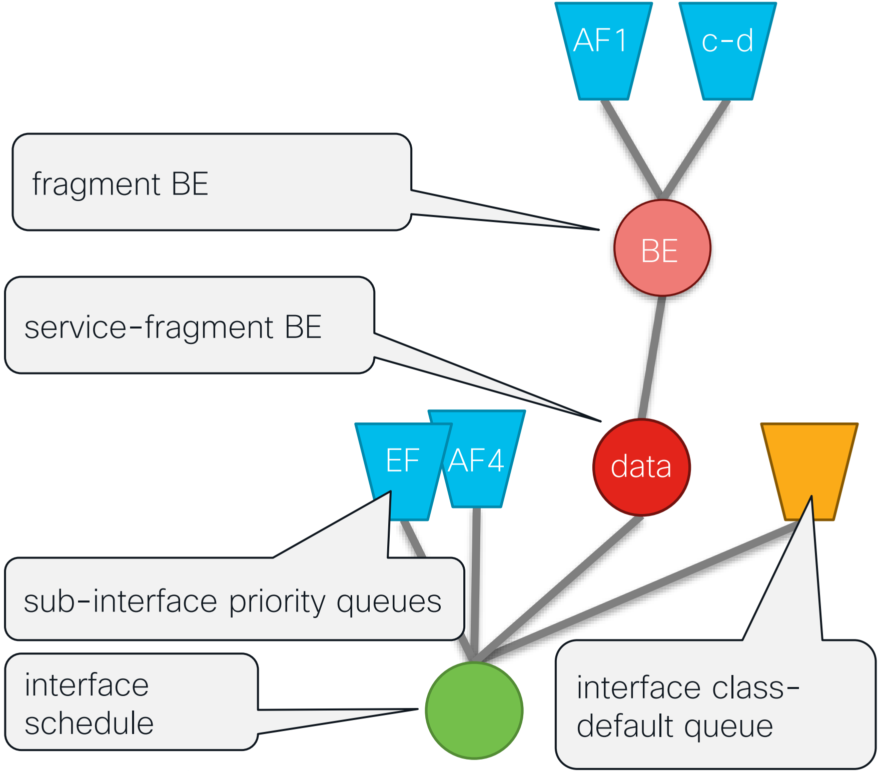
```
policy-map sub-int-mod4
 class EF
  police 1000000
 class AF4
  police 2000000
 class class-default fragment BE
  shape average 75000000
!
policy-map int-mod4
 class data service-fragment BE
  shape average 128000000
 class EF
  priority level 1
 class AF4
  priority level 2
 class AF1
  shape average 50000000
  random-detect
!
interface GigabitEthernet0/0/0
 service-policy output int-mod4
!
interface GigabitEthernet0/0/0.2
 service-policy output sub-int-mod4
!
interface GigabitEthernet0/0/0.2
 service-policy output sub-int-mod4
```

QoS service-fragments model 3



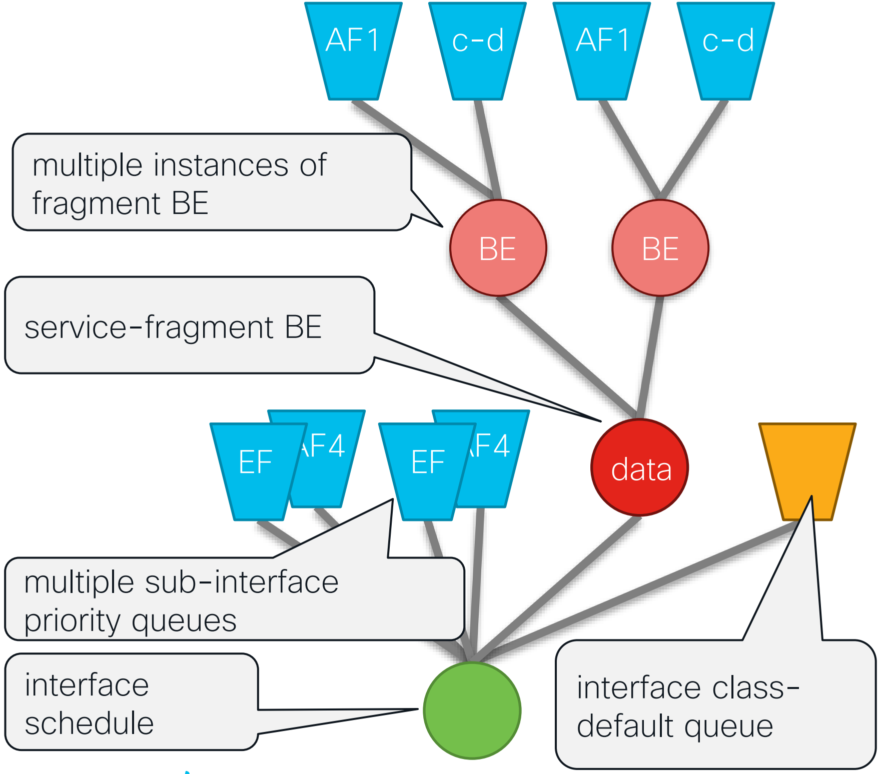
```
policy-map int-mod3
 class data service-fragment BE
  shape average 128000000
 class class-default
  shape average 100000000
!
policy-map sub-int-mod3
 class EF
  police 1000000
  priority level 1
 class AF4
  police 2000000
  priority level 2
 class class-default fragment BE
  shape average 115000000
  service-policy sub-int-child-mod3
!
policy-map sub-int-child-mod3
 class AF1
  shape average 100000000
 class class-default
  shape average 25000000
```

QoS service-fragments model 3



```
policy-map int-mod3
 class data service-fragment BE
  shape average 128000000
 class class-default
  shape average 100000000
!
policy-map sub-int-mod3
 class EF
  police 1000000
  priority level 1
 class AF4
  police 2000000
  priority level 2
 class class-default fragment BE
  shape average 115000000
  service-policy sub-int-child-mod3
!
policy-map sub-int-child-mod3
 class AF1
  shape average 100000000
 class class-default
  shape average 25000000
```

QoS service-fragments model 3



AF1
c-d
AF1
c-d
multiple instances of fragment BE
BE
BE
service-fragment BE
EF
AF4
EF
AF4
data
multiple sub-interface priority queues
interface schedule
interface class-default queue

```
policy-map int-mod3
 class data service-fragment BE
  shape average 128000000
 class class-default
  shape average 100000000
!
policy-map sub-int-mod3
 class EF
  police 1000000
  priority level 1
 class AF4
  police 2000000
  priority level 2
 class class-default fragment BE
  shape average 115000000
  service-policy sub-int-child-mod3
!
policy-map sub-int-child-mod3
 class AF1
  shape average 100000000
 class class-default
  shape average 25000000
```

Service groups

What are service-groups?

- Service-groups allow linking multiple L3 sub-interfaces and L2 service instances together for the purpose of aggregated QoS
- Before service-groups
  - QoS policies could be applied to individual L3 sub-interfaces, individual L2 service instances, or to ethernet main interfaces
  - In order to group multiple L3 or L2 entities together for QoS, a “mega-policy” on the main interface which classified multiple vlans in the topmost layer was required.
  - If various groups of vlans on the same physical interface required QoS, the configuration quickly became unmanageable.

New configuration commands for service-groups

```
policy-map alpha
  class-default
    shape average 10000000
!
interface GigabitEthernet0/0/0
  service instance 11 ethernet
    encapsulation dot1q 11
    group 10
 service instance 12 ethernet
    encapsulation dot1q 12
    group 10
!
interface GigabitEthernet0/0/0.13
  encapsulation dot1q 13
  group 10
!
interface GigabitEthernet0/0/0.14
  encapsulation dot1q 14
  group 10
!
service-group 10
  service-policy alpha
```

Use the `group` keyword to put service instances and sub-interfaces into a service-group.

Use the `service-group` command as the application point for QoS policies.

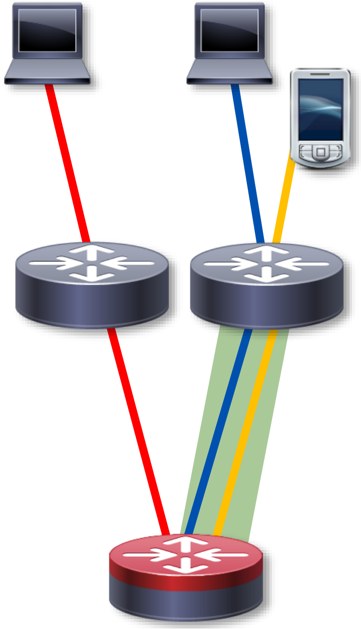
Service-group configuration

- Ingress and egress policy-maps are supported on service-groups
- Up to three levels in policy-maps (ingress and egress)
    - Same hierarchy restrictions as policy-maps applied to main-interfaces
- Support for all Ethernet interfaces and Aggregate Port-channels
- No support for non-ethernet interface types
- Statistics are collected on a per service-group level and will not be available per service-group member unless explicitly classified as part of the service policy

Restrictions

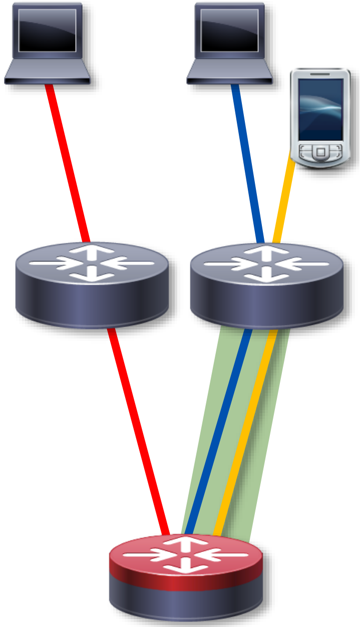
- All members of a given service-group must be on the same physical interface
- A sub-interface or service instance can belong to only one service-group at time
- Sub-interfaces and service instances in a service-group can not have a policy-map applied other than on the service-group
- Tunnels with QoS egressing through service-group not supported

Service groups use case

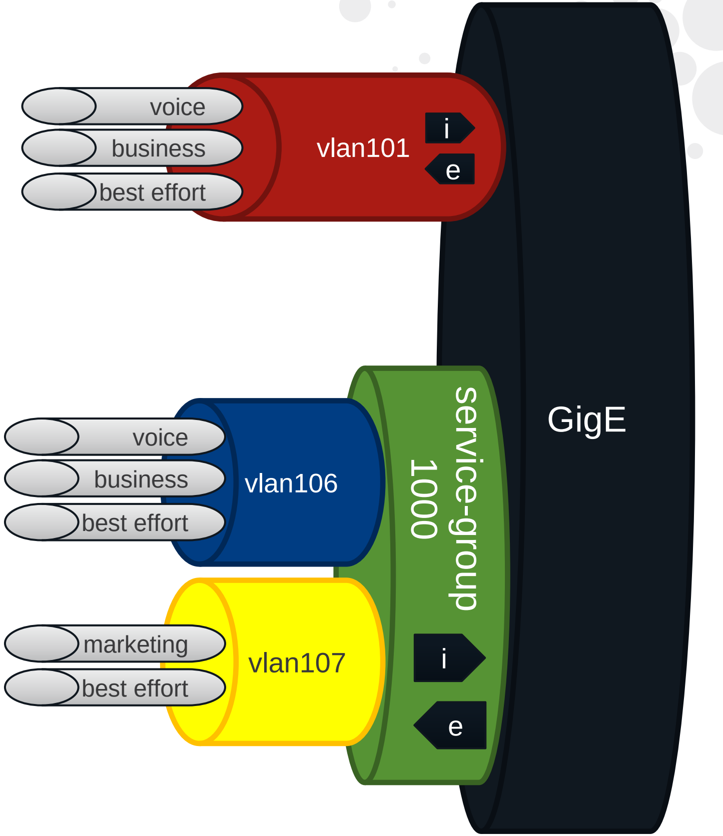


- Left side branch
  - Traditional deployment with a single VLAN servicing the entire branch
- Right side branch
  - serviced by a single downlink across the WAN but uses VLANs (blue and yellow) to differentiate business traffic and customer BYOD traffic.
  - From the headend C8500, it is necessary to rate limit traffic to the CPE as a whole but also make guarantees to the business class traffic over the BYOD traffic via vlan classification

Service groups use case



```
policy-map service-group-1000
  class-default
    shape average 10000000
    service-policy right-branch
!
policy-map right-branch
  class-map vlan106
    bandwidth remaining ratio 20
    service-policy workers
  class-map vlan107
    bandwidth remaining ratio 1
    service-policy guests
!
policy-map workers
  class-map voice
    priority
    police cir 1000000
  class-map business
    bandwidth 8000
  class-map class-default
!
policy-map guests
  class-map marketing
    bandwidth remaining ratio 10
  class-map class-default
    bandwidth remaining ratio 1
```



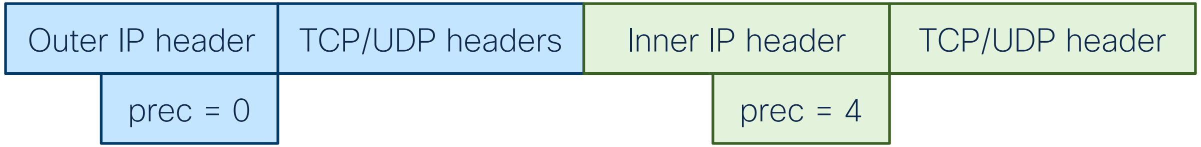
Dealing with Tunnels

Use QoS-group marking

- All packets can be marked with QoS-group
  - invisible marking that will follow the packet through processing
  - does not modify that packet on the wire in any way
  - survives any transformation
    - tunnel, imposition, crypto, fragmentation)
- Very useful to mark packets on ingress when they are unobscured and then classify on egress once they have been manipulated
- Helps work around some restrictions on logical versus physical interfaces

Tunnel “qos pre-classify”

- Set this command on tunnels to use the encapsulated packet for classification later.



- Policy-map on egress physical interface will see the outer headers normally
- With ”qos pre-classify” on the tunnel, egress physical interface will use the encrypted inner / green headers for classification.

Order of operations with tunnels

	physical interface QoS policy with queuing actions	physical interface QoS policy without queuing actions
tunnel policy-map with queuing actions	Class-default only policy-map on physical interface supported. Packets will be managed by tunnel policy then rate limited by the interface policy (class-default only).	Tunnel packets bypass interface policy-map
tunnel policy-map without queuing actions	Tunnel packets go through tunnel policy-map fully and then through interface policy-map (class-default only)	Tunnel packets go through tunnel policy-map fully and then through interface policy-map (interface default queue). Interface policy-map has no effect. Traffic is only classified once in IOS XE.

- Maximum of two levels in policy-map hierarchies allowed on tunnels.
- Encryption is executed prior to egress queuing.

Troubleshooting

Packet buffer memory utilization

```
C8500-12X#show platform hard qfp active bqs 0 packet-buffer utilization
Packet buffer memory utilization details:
  QFP.0:
    Total:     161.0    0 MB
         :     963.00 MB cblk
    Used :    1152.00 KB
         :   14544.00 KB cblk
    Free :     159.88 MB
         :     948.80 MB cblk

    Utilization:    0 %
               :    1 % cblk

    Threshold Values:
      Vital                :    160.94 MB, Status: False
                           :    962.91 MB cblk
      Packet Priority      :    159.44 MB, Status: False
                           :    953.39 MB cblk
      Priority             :    152.94 MB, Status: False
                           :    914.81 MB cblk
      Non-Priority         :    136.81 MB, Status: False
                           :    818.44 MB cblk
```

Non-priority user data dropped at 85% packet memory utilization

Priority user data dropped at 97% packet memory utilization

PAK_PRI and internal control packets only dropped at 100% memory utilization

BQS queue and schedule scale

```
C8500-12X#show platform hardware qfp active infrastructure bqs status
BQS-RM Status :
=============================================
Object Counts:
  Recycle Object Count:               149
  Recycle Schedule Count:              26
  Recycle Queue Count:                 89
  # of Active Queues:                 501
  # of Active Schedules:              518
  # of Active Roots:                    9
  # of Active Min Profiles:            10
  # of Active Max Profiles:             4
  # of Active Exs Profiles:             5
```

TCAM usage

```
C8500-12X#show platform hardware qfp active tcam resource-manager usage
QFP TCAM Usage Information

--snip--

Total TCAM Cell Usage Information
----------------------------------
Name                                : TCAM #0 on CPP #0
Total number of regions             : 3
Total tcam used cell entries        : 44
Total tcam free cell entries        : 131028
Threshold status                    : below critical limit
```

Pending objects – unfinished data plane work

```
C8500-12X#show platform software object-manager f0 statistics
Forwarding Manager Asynchronous Object Manager Statistics

Object update: Pending-issue: 0, Pending-acknowledgement: 0
Batch begin:   Pending-issue: 0, Pending-acknowledgement: 0
Batch end:     Pending-issue: 0, Pending-acknowledgement: 0
Command:       Pending-acknowledgement: 0
Total-objects: 1315
Stale-objects: 0
Resolve-objects: 0
Childless-delete-objects: 0
Backplane-objects: 0
Error-objects: 0
Number of bundles: 0
Paused-types: 3
```

Details from show policy-map interface

```
c8000v#show policy-map interface
 GigabitEthernet1

  Service-policy output: Output-250Gb

    Class-map: class-default (match-any)
      4056967 packets, 3064214398 bytes
      30 second offered rate 41000 bps, drop rate 0000 bps
      Match: any
      Queueing
      queue limit 1041 packets
      (queue depth/total drops/no-buffer drops) 0/27384/0
      (pkts output/bytes output) 3856784/3033469569
      shape (average) cir 250000000, bc 1000000, be 1000000
      target shape rate 250000000
       Overhead Accounting Enabled
```

Drops from platform statistics

```
c8000v#show platform hardware qfp active statistics drop
Last clearing of QFP drops statistics : never

-------------------------------------------------------------------------
Global Drop Stats                         Packets                  Octets
-------------------------------------------------------------------------
TailDrop                                 72374984                  483790
QosPolicing                               1504778              1268527854
BqsOor                                          0                       0
BqsOorPakPri                                    0                       0
BqsOorPri                                       0                       0
BqsOorVital                                     0                       0
Wred                                            0                       0
```

Conclusion

Conclusion

- 3 parameter scheduler – different from Classic IOS platforms
- Remember that you don’t always have to classify were you deploy QoS
  - Use QoS groups, use tunnel pre-classify
- Remember that you don’t have to have strict hierarchies on singlt targets
  - Service-fragments, service-groups
- Remember to manage queue-limits appropriately
  - Time and byte based configurations

Complete your Session Survey

- Please complete your session survey after each session. Your feedback is important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (open from Thursday) to receive your Cisco Live t-shirt.

- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Session Catalog and clicking the "Attendee Dashboard” at https://www.ciscolive.com/emea/learn/sessions/session-catalog.html

Continue Your Education

Visit the Cisco Showcase for related demos.

Book your one-on-one Meet the Engineer meeting.

Attend any of the related sessions at the DevNet, Capture the Flag, and Walk-in Labs zones.

Visit the On-Demand Library for more sessions at ciscolive.com/on-demand.

The bridge to possible

Thank you

cisco Live!

ALL IN