Best Practices for Troubleshooting Cisco Catalyst 8000 Edge Platforms

Michal Stanczyk - Technical Leader, Cisco TAC

BRKTRS-2572

Disclaimer

This session IS NOT about:

- Sales pitch
- Troubleshooting using mouse
- Cisco 8000 Series (IOS-XR)

This session IS about:

- Defining and diagnosing problems
- Troubleshooting using keyboard
- Catalyst 8000 (IOS-XE) and its tools

Session Goal

Boost your troubleshooting proficiency and confidence in tackling Catalyst 8000 platform issues either independently or with support of Cisco TAC.

Agenda

- Introduction
- Packet Walk Through Catalyst 8000
- Troubleshooting Packet Loss
- Platform Resources Verification
- Conclusion

Webex App

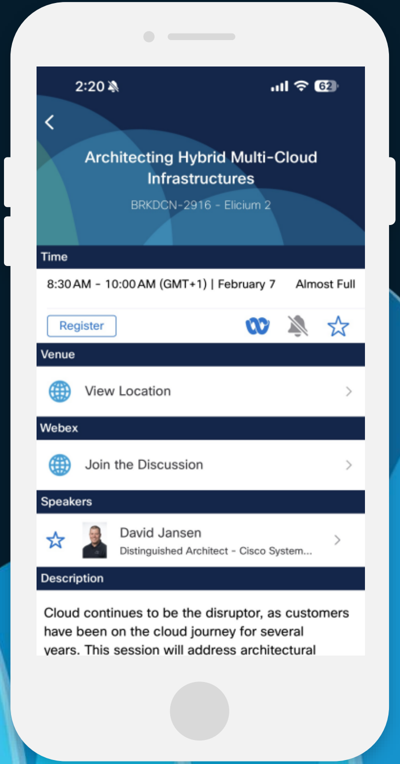
Questions?

Use the Webex app to chat with the speaker after the session

How

1. Find this session in the Cisco Events 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 by the speaker until February 28, 2025.



2:20

Architecting Hybrid Multi-Cloud Infrastructures

BRKDCN-2916 - Elicium 2

Time

8:30 AM - 10:00 AM (GMT+1) | February 7 Almost Full

Venue

Webex

Speakers

David Jansen
Distinguished Architect - Cisco System...

Description

Cloud continues to be the disruptor, as customers have been on the cloud journey for several years. This session will address architectural

Introduction

Glossary

For Your Reference

RP – Route Processor

FP – Forwarding Processor

QFP – Quantum Flow Processor

PPE – Packet Processing Engine

BQS – Buffering, Queueing, Scheduling

SoC – System on Chip

DPDK – Data Plane Development Kit

QAT – Quick Assist Technology

LBD – Load Based Distribution

NSFBD – Non-strict Flow Based Distribution

SFBD – Strick Flow Based Distribution

COFF – Crypto Offload

TM – Traffic Manager

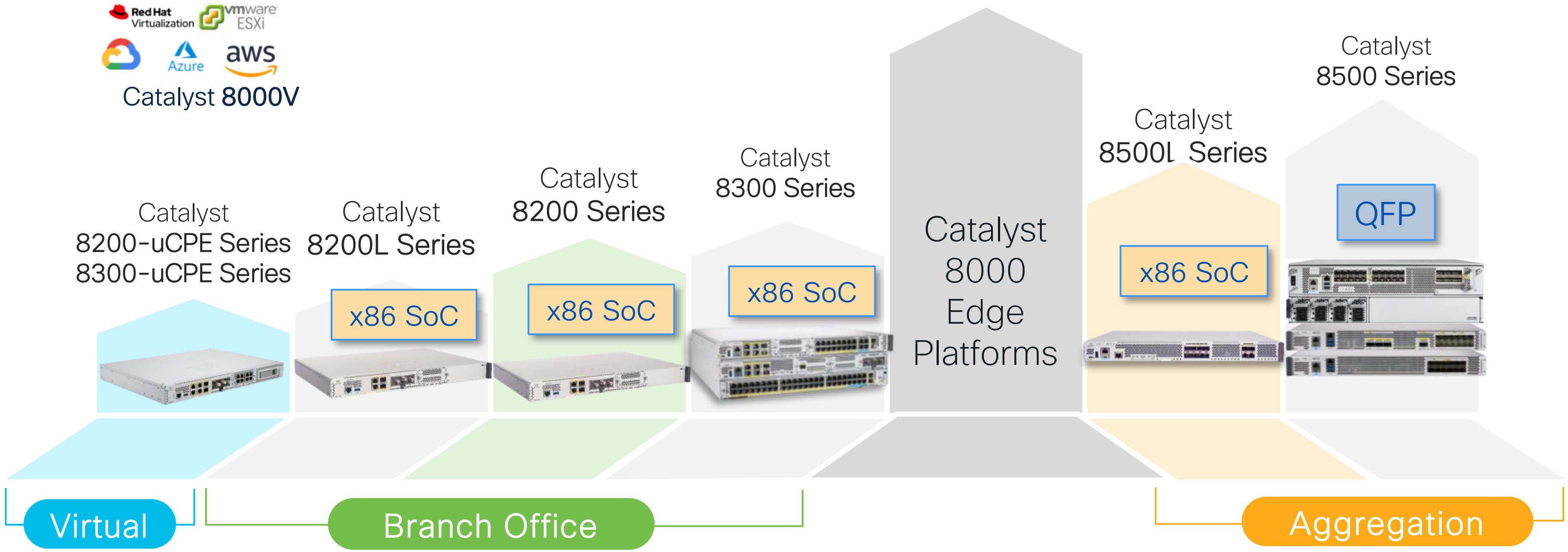
EEM – Embedded Event Manager

RSS – Resident Set Size

CACE – Common Adaptive Classification Engine

Cisco Catalyst 8000 Edge Family

* QFP = Quantum Flow Processor
* SoC = System on Chip



Red Hat Virtualization
vmware ESXi
Azure
aws
Catalyst 8000V

Catalyst
8200-uCPE Series
8300-uCPE Series

Catalyst
8200L Series
x86 SoC

Catalyst
8200 Series
x86 SoC

Catalyst
8300 Series
x86 SoC

Catalyst
8000
Edge
Platforms

Catalyst
8500L Series
x86 SoC

Catalyst
8500 Series
QFP

Virtual

Branch Office

Aggregation

cisco Live!

What’s different?

Main architectural differences between Catalyst 8000 physical platforms

QFP-based platforms (successors of ASR1000)

- ❖ C8500-20X6C
- ❖ C8500-12X4QC
- ❖ C8500-12X

x86 SoC (System on Chip) platforms (successors of ISR4000)

- ❖ C8200(L)
- ❖ C8300
- ❖ C8500L

- ➢ Physical dataplane CPU (QFP 3.0)
- ➢ Hundreds of packet processing cores/threads
- ➢ Hardware accelerated crypto (16 crypto engines)
- ➢ Physical TCAM for classification lookups

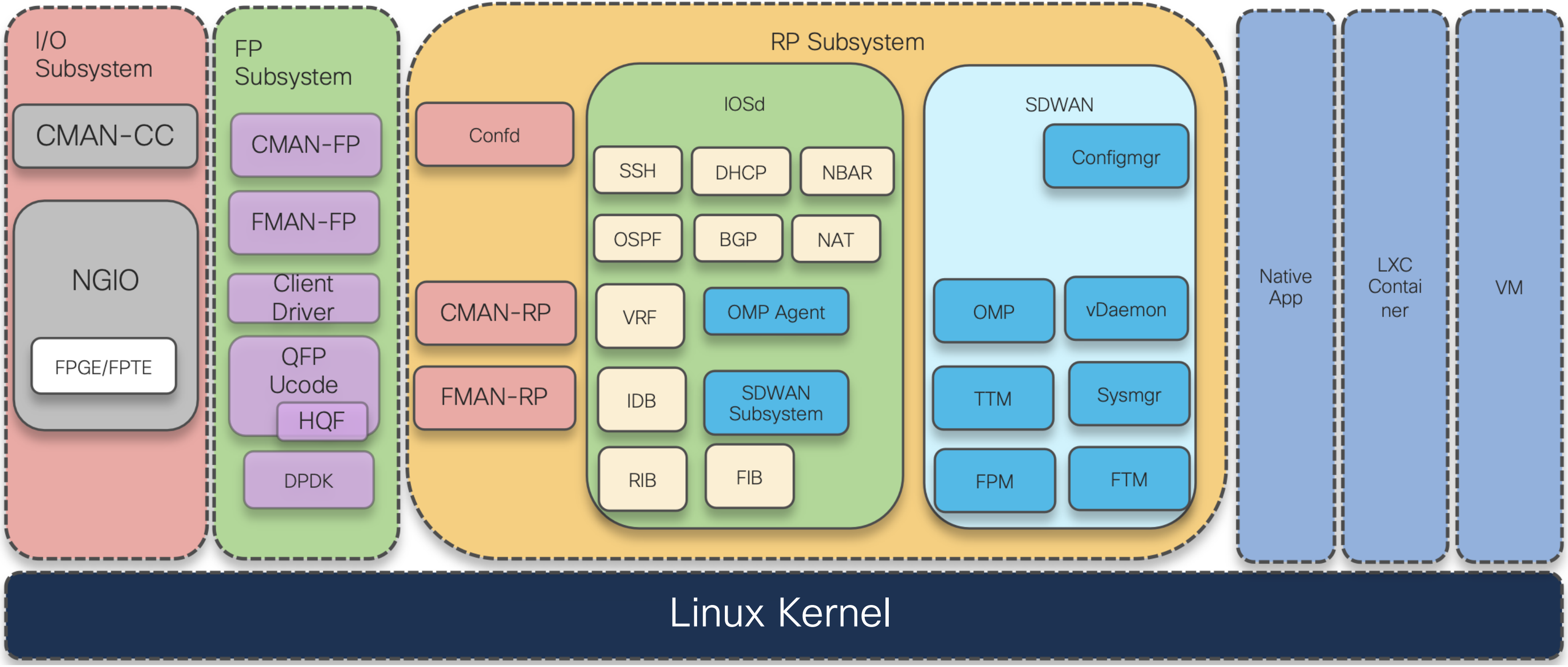
- ➢ x86 CPU with DPDK for dataplane
- ➢ Up to ~20 CPU threads (Dynamic Core Allocation)
- ➢ QAT for in-line crypto acceleration
- ➢ QFP Resource Memory for classification lookups

What’s common?

Common areas across Catalyst 8000 platforms

- IOS-XE software architecture
- Logging infrastructure (binary tracing/unified tracing)
- QFP datapath troubleshooting workflow and tools
  - Packet Trace
  - Embedded Packet Capture
  - Conditional Debugging

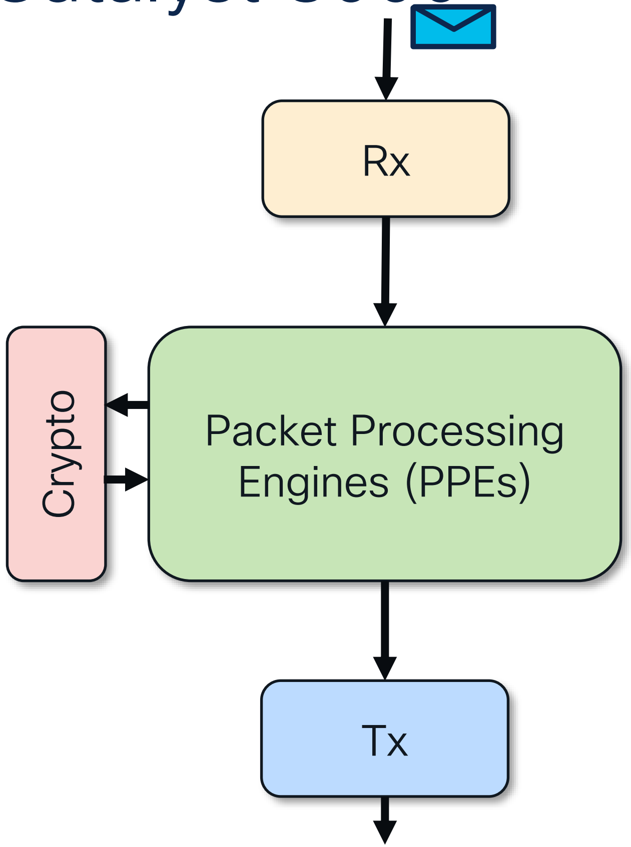
IOS-XE Software Architecture



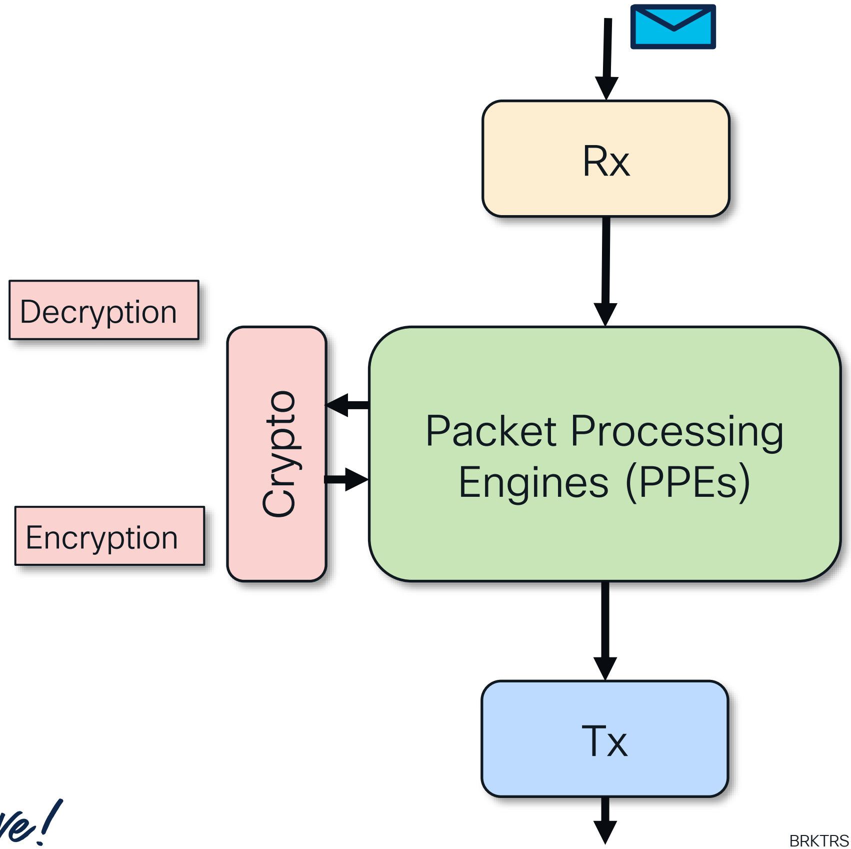
Packet Walk Through Catalyst 8000

General data plane functions on Catalyst 8000

- Receiving traffic (Rx)
- Distributing traffic (Rx)
  - Load based distribution (LBD)
  - Non-strict flow-based distribution (NSFBD)
  - Strict flow-based distribution (SFBD)
- Crypto processing
- Forwarding/Feature Processing (PPE)
- Queuing and scheduling (Tx)



General data plane functions on Catalyst 8000



Parse outer header then send to crypto **if** needed

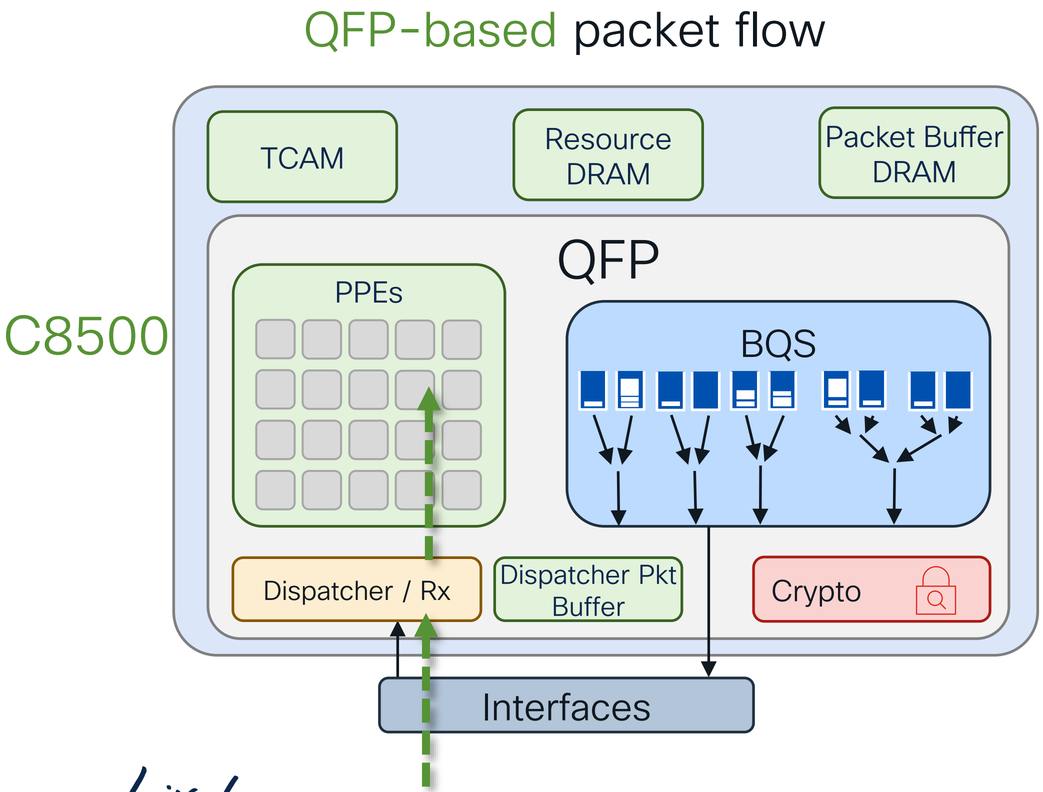
Remove outer headers
Apply ingress features

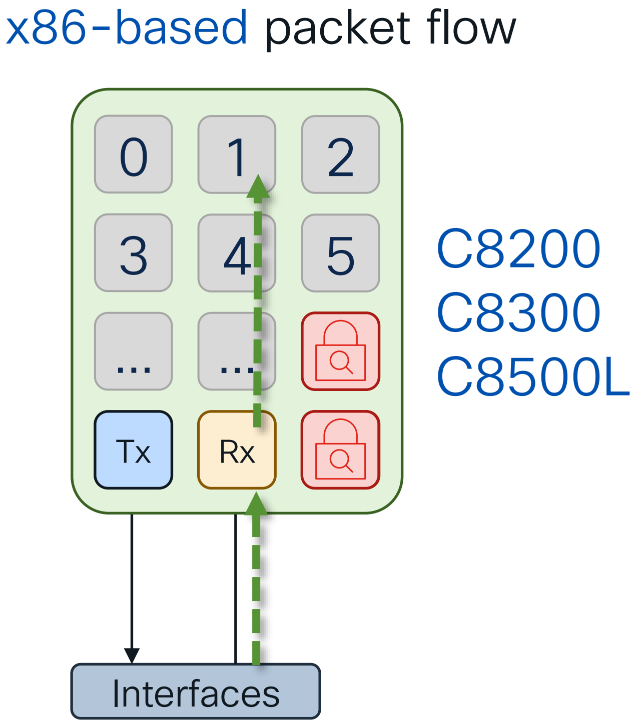
Determine next hop and output interface

Apply egress features
Encapsulate packet

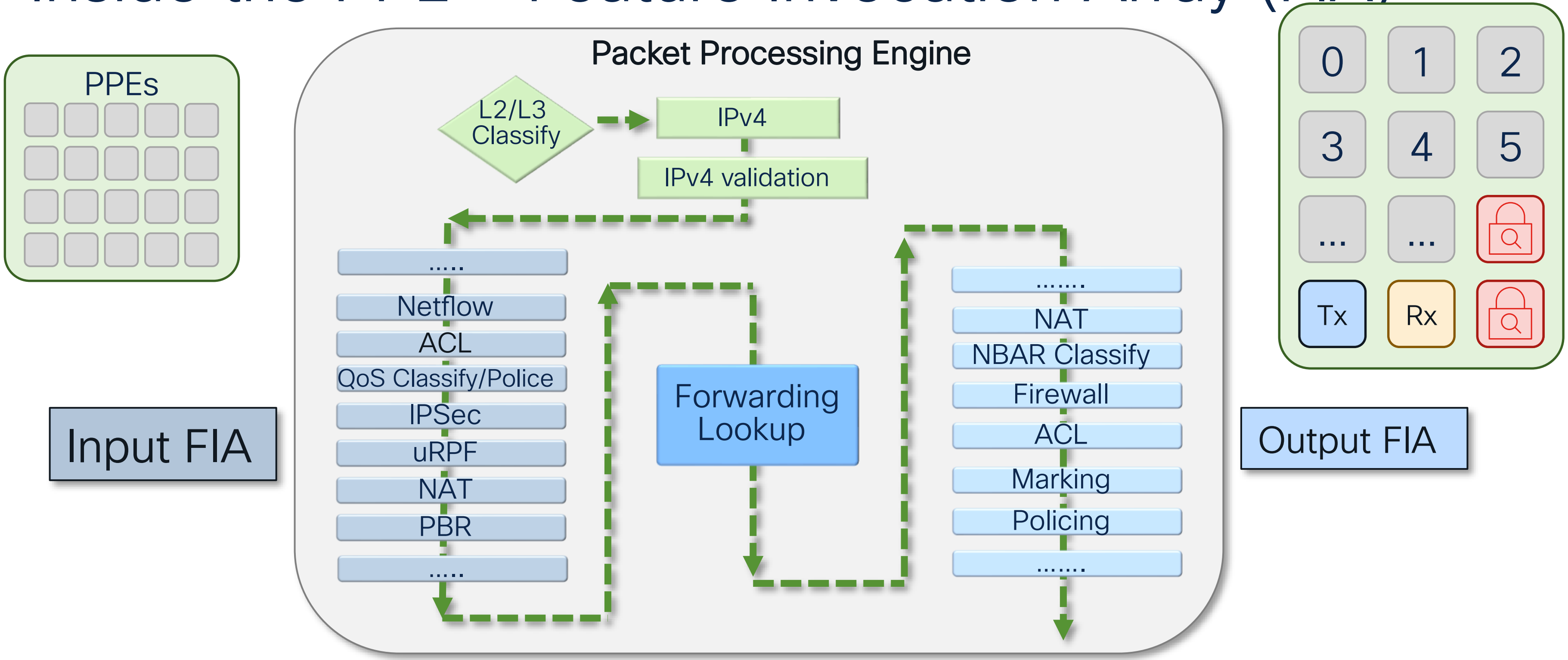
Queueing/Scheduling

Packet Walk Through Catalyst 8000





Inside the PPE – Feature Invocation Array (FIA)



PPEs

Packet Processing Engine

L2/L3 Classify

IPv4

IPv4 validation

.....

Netflow

ACL

QoS Classify/Police

IPSec

uRPF

NAT

PBR

.....

Forwarding Lookup

....

NAT

NBAR Classify

Firewall

ACL

Marking

Policing

....

Input FIA

Output FIA

0

1

2

3

4

5

...

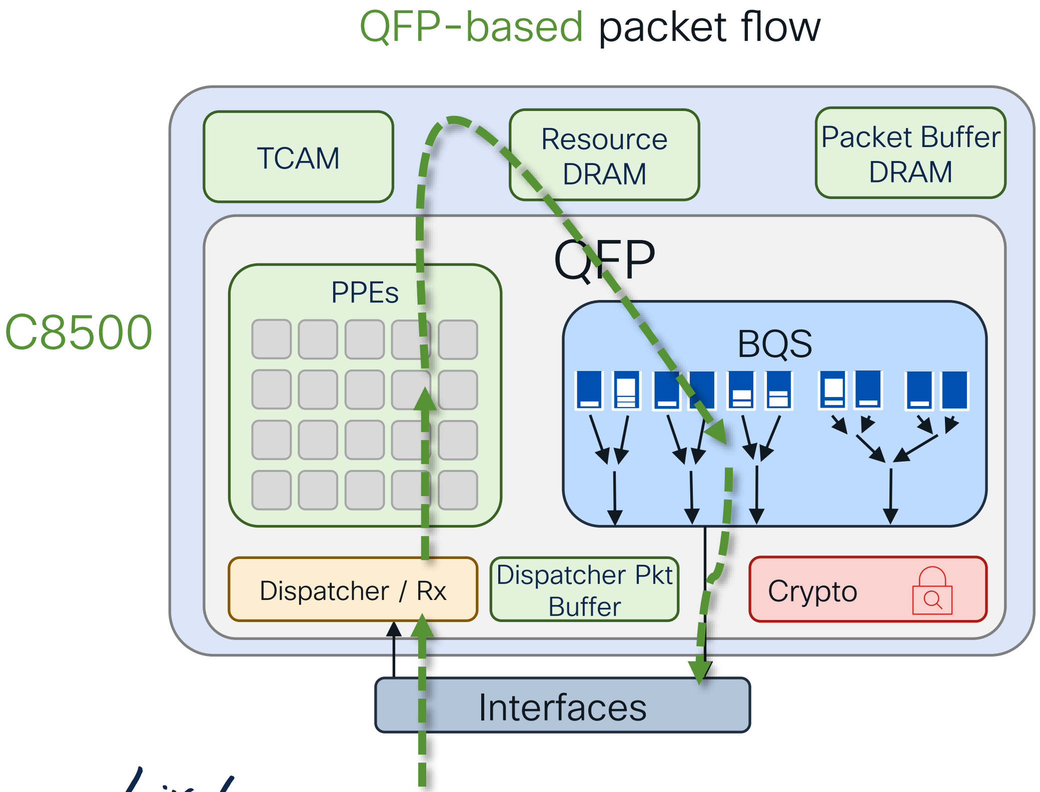
...

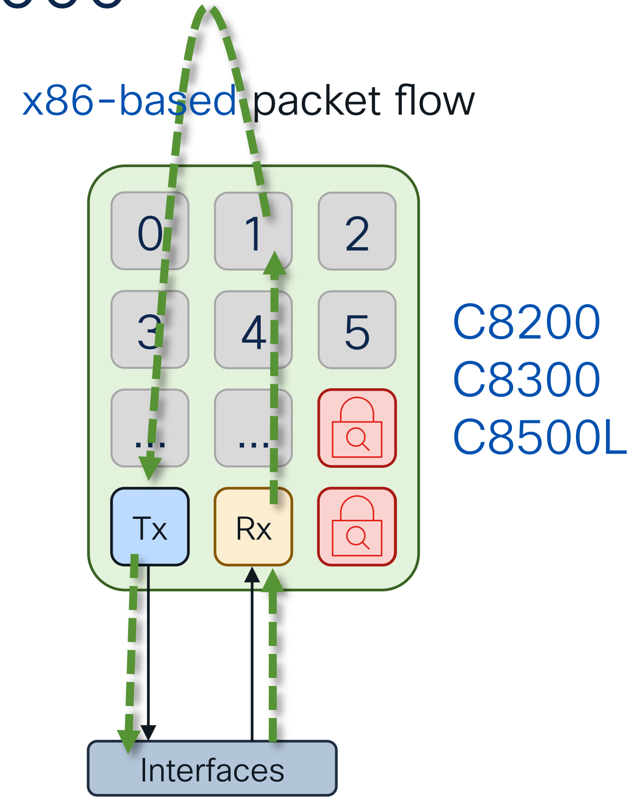
Tx

Rx

```
show platform hardware qfp active interface if-name <name>
```

Packet Walk Through Catalyst 8000





Dynamic Core Allocation (x86 based platforms)

- SoC platforms use multi-core CPUs

data plane   control plane   service plane

- HyperTreading enabled on some cores (~30% performance gain)

Core allocation templates

Data Plane Heavy (DPH)	Optimized for throughput
Service Plane Heavy (SPH)	Optimized for app hosting
Application Heavy (APH)	Optimized for app hosting (extra CPUs) C8500L only

Default in SD-WAN mode

C8500L-8S4X – SP heavy

BRKENT-2653

	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7	Core 8	Core 9	Core 10	Core 11
CPU Thread 0	🟦 0	🟦 1	🟪 2	🟪 3	🟪 4	🟪 5	🟩 6	🟩 7	🟩 8	🟩 9	🟧 10	🟥 11
CPU Thread 1	🟦 12	🟦 13	🟪 14	🟪 15	🟪 16	🟪 17	🟩 18	🟩 19	🟩 20	🟩 21	⬜ 22	⬜ 23

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

- 🟪 Service plane
- 🟩 Data plane
- 🟦 Control plane
- 🟧 I/O actions
- 🟥 Crypto
- ⬜ Idle

Default in autonomous mode

C8500L-8S4X – DP heavy

BRKENT-2653

	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7	Core 8	Core 9	Core 10	Core 11
CPU Thread 0	🟦 0	🟦 1	🟩 2	🟩 3	🟩 4	🟩 5	🟩 6	🟩 7	🟧 8	🟧 9	🟥 10	🟥 11
CPU Thread 1	🟦 12	🟦 13	🟩 14	🟩 15	🟩 16	🟩 17	🟩 18	🟩 19	⬜ 20	⬜ 21	⬜ 22	⬜ 23

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

- 🟪 Service plane
- 🟩 Data plane
- 🟦 Control plane
- 🟧 I/O actions
- 🟥 Crypto
- ⬜ Idle

For Your Reference

BRKENT-2653

C8500L-8S4X - App heavy

	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7	Core 8	Core 9	Core 10	Core 11
CPU Thread 0	🟦 0	🟪 1	🟪 2	🟪 3	🟪 4	🟪 5	🟪 6	🟪 7	🟩 8	🟩 9	🟧 10	🟥 11
CPU Thread 1	🟦 12	🟪 13	🟪 14	🟪 15	🟪 16	🟪 17	🟪 18	🟪 19	🟩 20	🟩 21	⬜ 22	⬜ 23

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

🟪 Service plane

🟩 Data plane

🟦 Control plane

🟧 I/O actions

🟥 Crypto

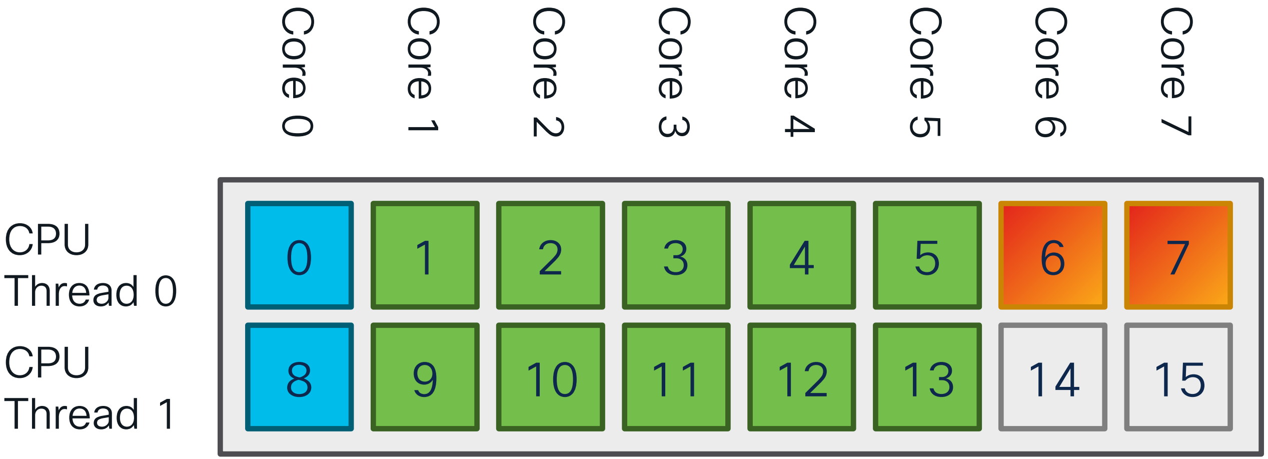
⬜ Idle

Default in autonomous mode

For Your Reference

BRKENT-2653

C8300-1N1S-4T2X – DP heavy



	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7
CPU Thread 0	🟦 0	🟩 1	🟩 2	🟩 3	🟩 4	🟩 5	🟥🟧 6	🟥🟧 7
CPU Thread 1	🟦 8	🟩 9	🟩 10	🟩 11	🟩 12	🟩 13	⬜ 14	⬜ 15

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

🟪 Service plane

🟩 Data plane

🟦 Control plane

🟧 I/O actions

🟥 Crypto

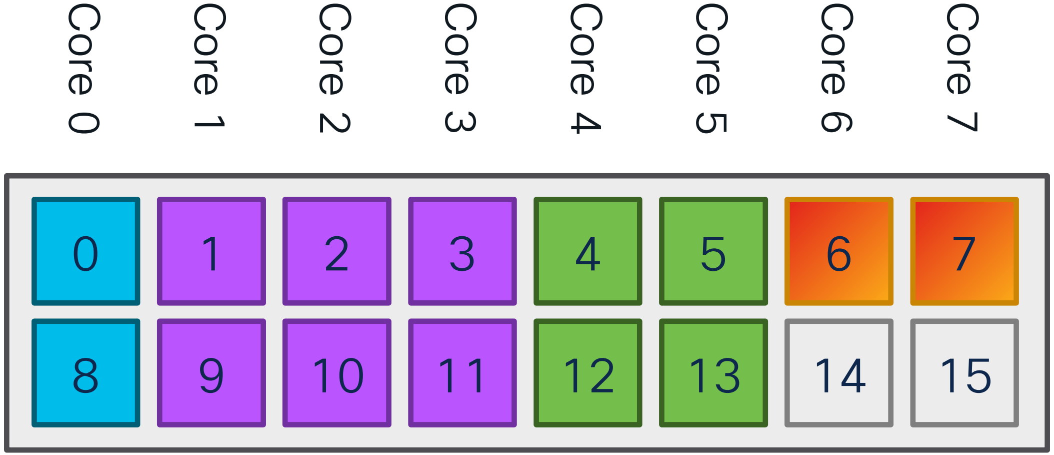
⬜ Idle

Default in SD-WAN mode

For Your Reference

BRKENT-2653

C8300-1N1S-4T2X – SP heavy



	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7
CPU Thread 0	🟦 0	🟪 1	🟪 2	🟪 3	🟩 4	🟩 5	🟥🟧 6	🟥🟧 7
CPU Thread 1	🟦 8	🟪 9	🟪 10	🟪 11	🟩 12	🟩 13	⬜ 14	⬜ 15

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

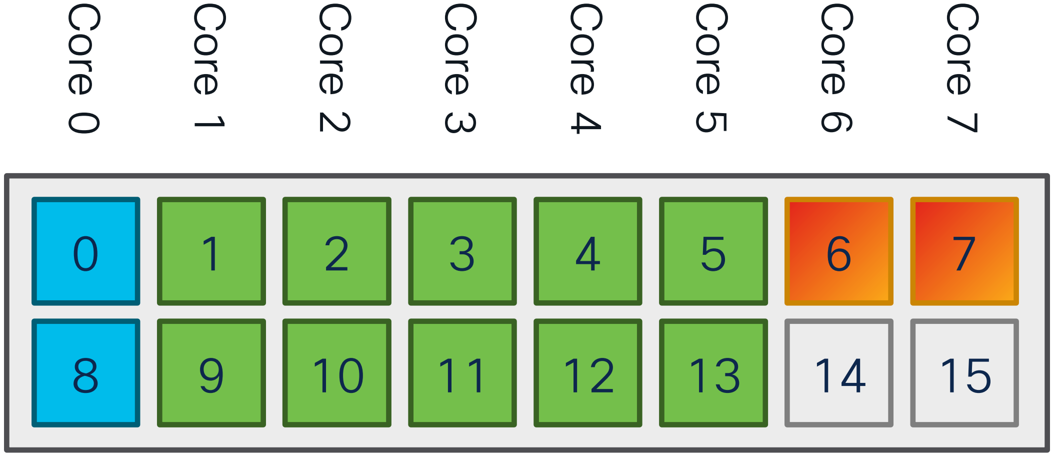
- 🟪 Service plane
- 🟩 Data plane
- 🟦 Control plane
- 🟧 I/O actions
- 🟥 Crypto
- ⬜ Idle

Default in autonomous mode

For Your Reference

BRKENT-2653

C8300-1N1S-4T2X – SP heavy



	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7
CPU Thread 0	🟦 0	🟩 1	🟩 2	🟩 3	🟩 4	🟩 5	🟥🟧 6	🟥🟧 7
CPU Thread 1	🟦 8	🟩 9	🟩 10	🟩 11	🟩 12	🟩 13	⬜ 14	⬜ 15

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

🟪 Service plane

🟩 Data plane

🟦 Control plane

🟧 I/O actions

🟥 Crypto

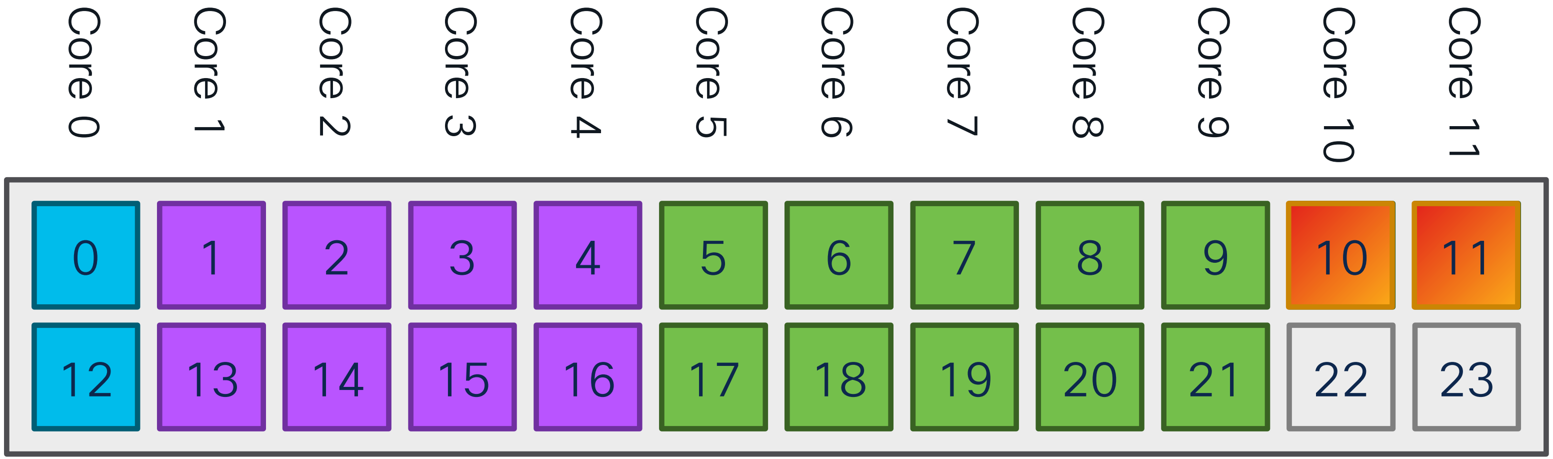
⬜ Idle

Default in autonomous mode

For Your Reference

BRKENT-2653

C8300-2N2S-4T2X - DP heavy



	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7	Core 8	Core 9	Core 10	Core 11
CPU Thread 0	🟦 0	🟪 1	🟪 2	🟪 3	🟪 4	🟩 5	🟩 6	🟩 7	🟩 8	🟩 9	🟥🟧 10	🟥🟧 11
CPU Thread 1	🟦 12	🟪 13	🟪 14	🟪 15	🟪 16	🟩 17	🟩 18	🟩 19	🟩 20	🟩 21	⬜ 22	⬜ 23

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

🟪	🟩	🟦	🟧	🟥	⬜
Service plane	Data plane	Control plane	I/O actions	Crypto	Idle

Default in autonomous mode

For Your Reference

BRKENT-2653

C8300-2N2S-4T2X - SP heavy

	Core 0	Core 1	Core 2	Core 3	Core 4	Core 5	Core 6	Core 7	Core 8	Core 9	Core 10	Core 11
CPU Thread 0	🟦 0	🟪 1	🟪 2	🟪 3	🟪 4	🟪 5	🟩 6	🟩 7	🟩 8	🟩 9	🟥🟧 10	🟥🟧 11
CPU Thread 1	🟦 12	🟪 13	🟪 14	🟪 15	🟪 16	🟪 17	🟩 18	🟩 19	🟩 20	🟩 21	⬜ 22	⬜ 23

This is the allocation for IOS XE 17.9 software. The allocations may vary per software versions.

🟪 Service plane

🟩 Data plane

🟦 Control plane

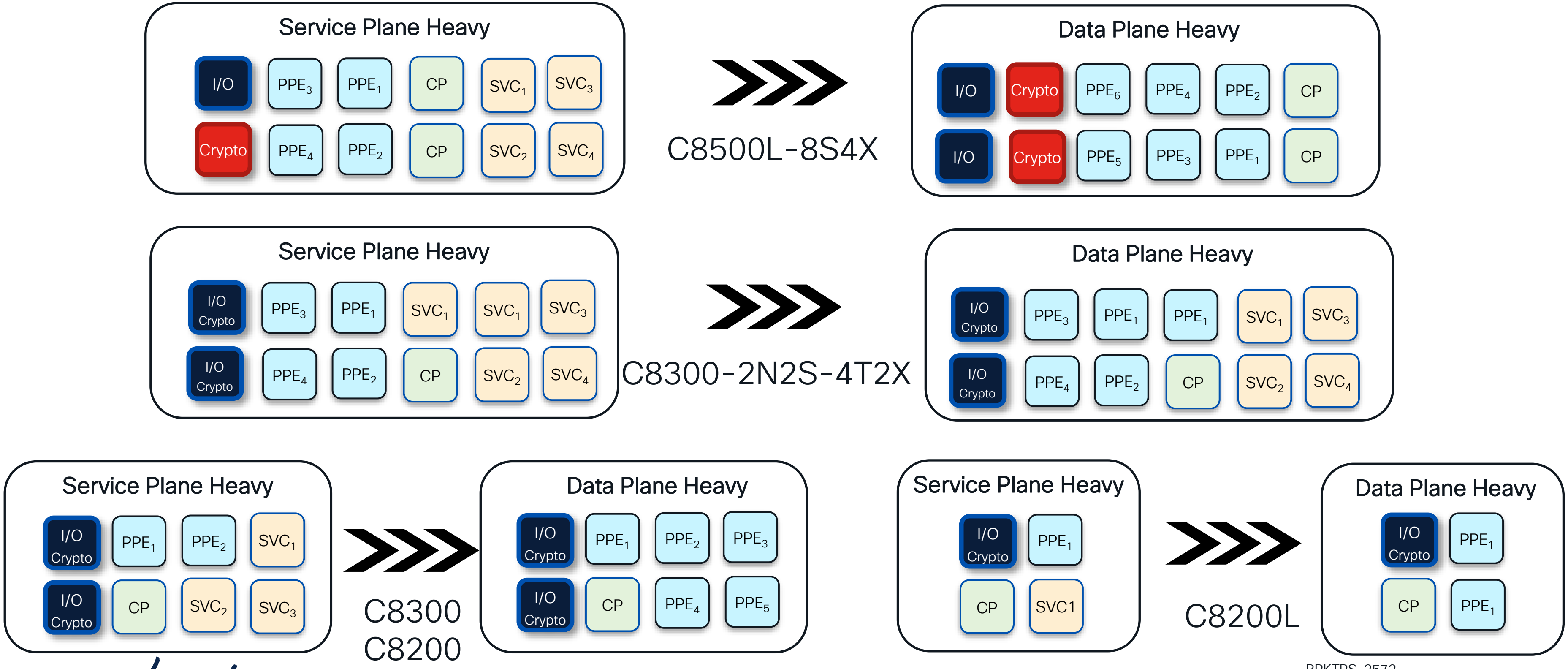
🟧 I/O actions

🟥 Crypto

⬜ Idle

For Your Reference

Dynamic Core Allocation (x86 based platforms)



C8500L-8S4X

Service Plane Heavy

I/O, PPE$_3$, PPE$_1$, CP, SVC$_1$, SVC$_3$

Crypto, PPE$_4$, PPE$_2$, CP, SVC$_2$, SVC$_4$

Data Plane Heavy

I/O, Crypto, PPE$_6$, PPE$_4$, PPE$_2$, CP

I/O, Crypto, PPE$_5$, PPE$_3$, PPE$_1$, CP

C8300-2N2S-4T2X

Service Plane Heavy

I/O Crypto, PPE$_3$, PPE$_1$, SVC$_1$, SVC$_1$, SVC$_3$

I/O Crypto, PPE$_4$, PPE$_2$, CP, SVC$_2$, SVC$_4$

Data Plane Heavy

I/O Crypto, PPE$_3$, PPE$_1$, PPE$_1$, SVC$_1$, SVC$_3$

I/O Crypto, PPE$_4$, PPE$_2$, CP, SVC$_2$, SVC$_4$

C8300
C8200

Service Plane Heavy

I/O Crypto, PPE$_1$, PPE$_2$, SVC$_1$

I/O Crypto, CP, SVC$_2$, SVC$_3$

Data Plane Heavy

I/O Crypto, PPE$_1$, PPE$_2$, PPE$_3$

I/O Crypto, CP, PPE$_4$, PPE$_5$

C8200L

Service Plane Heavy

I/O Crypto, PPE$_1$

CP, SVC1

Data Plane Heavy

I/O Crypto, PPE$_1$

CP, PPE$_1$

BRKTRS-2572

Traffic distribution models

Load based (LBD)

Packets are handled by any available data plane thread.

Platforms: C8500 (QFP), C8000V (x86)

Strict Flow based (S-FBD)

Packets are **strictly** distributed to PPEs based on flow hashing.

Platform: C8500L

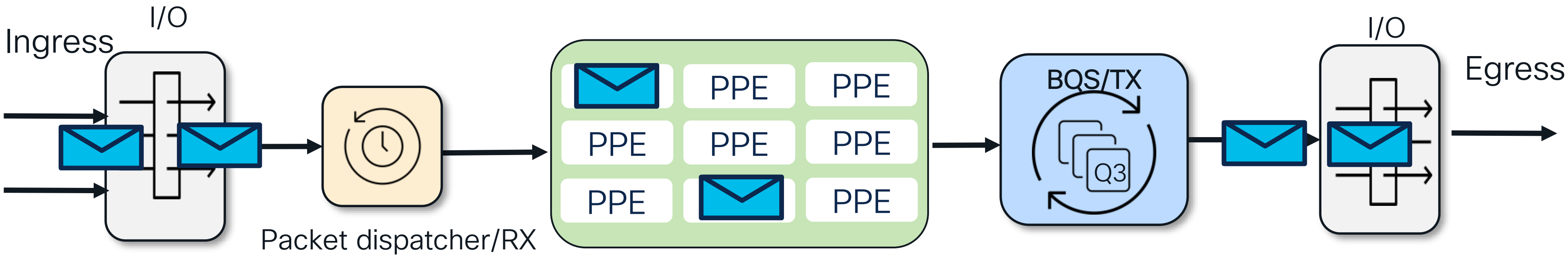
Non-strict Flow based (NS-FBD)

Trying to keep packets from a given flow on a given thread, in some conditions idle threads may assist.

Platforms: C8200L, C8200, C8300

C8500
C8000v

Load Based Distribution (LBD)



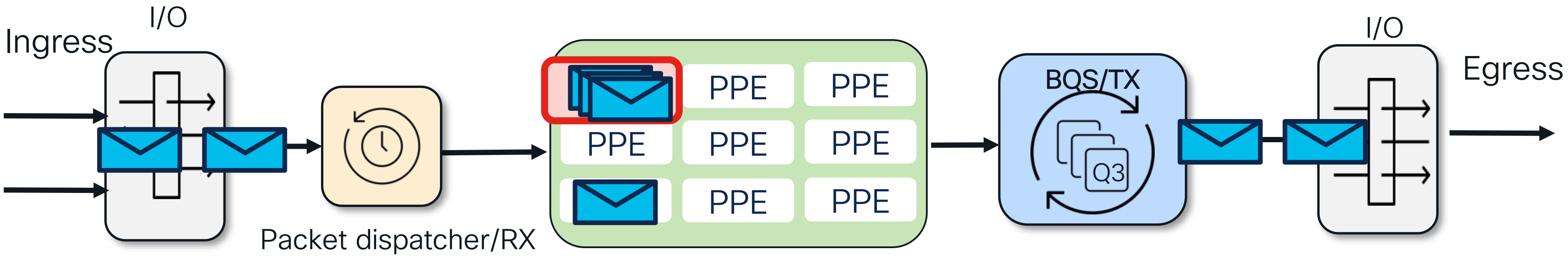
- Packets from the same flow can be processed by any available core
- State of the flow must be available to all core at any given time
- Packets are distributed strictly based on availability of QFP cores either via hardware dispatcher (QFP) or software Rx function (x86)
- QFP have optimized design which makes these challenges less of a concern

Challenges

- Packet ordering
- Memory access

C8200L
C8200
C8300

Non-Strict Flow Based Distribution (NS-FBD)



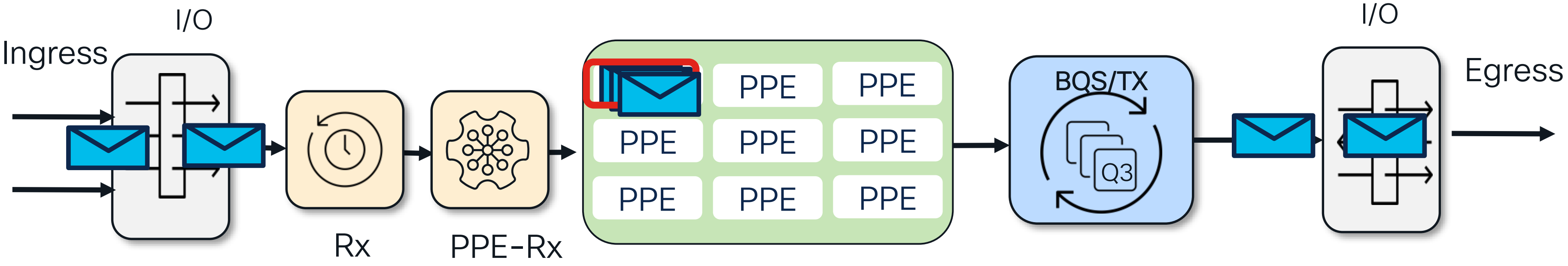
- There are efforts to keep same flow on same core to optimize forwarding
- State of the flow must be available to all cores at any given time
- Packets are classified to given cores based on the outer encapsulation of packets. If targeted core is busy, packet can be processed by different core.

Challenges

- Packet ordering
- Memory access

C8500L

Strict Flow Based Distribution (S-FBD)

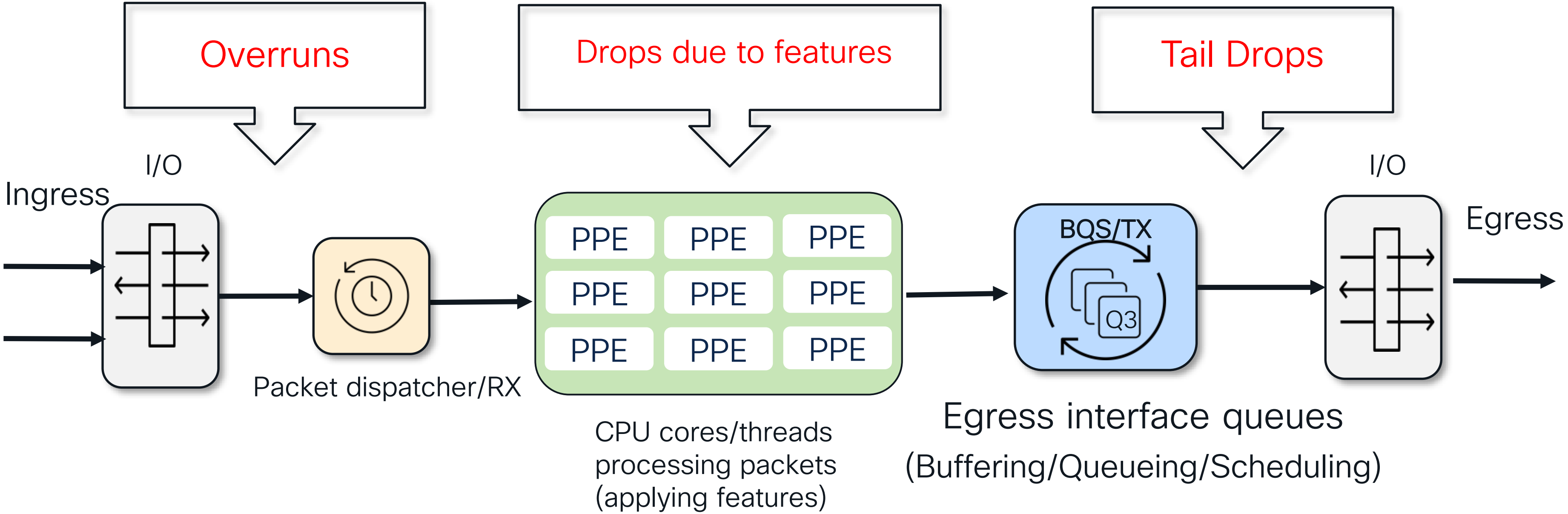


- Packets from same flow are always processed by same core
- State of the flow must be available only to single CPU core
- Packets are initially hashed and handled by Rx core and later passed to PPE-Rx function, so the CPU core handling specific flow can be found
- Suitable for environment with huge number of flows, **elephant flow might be concern**

x86 optimized

Troubleshooting Packet Drops

Packet drops – most common scenarios



Overruns

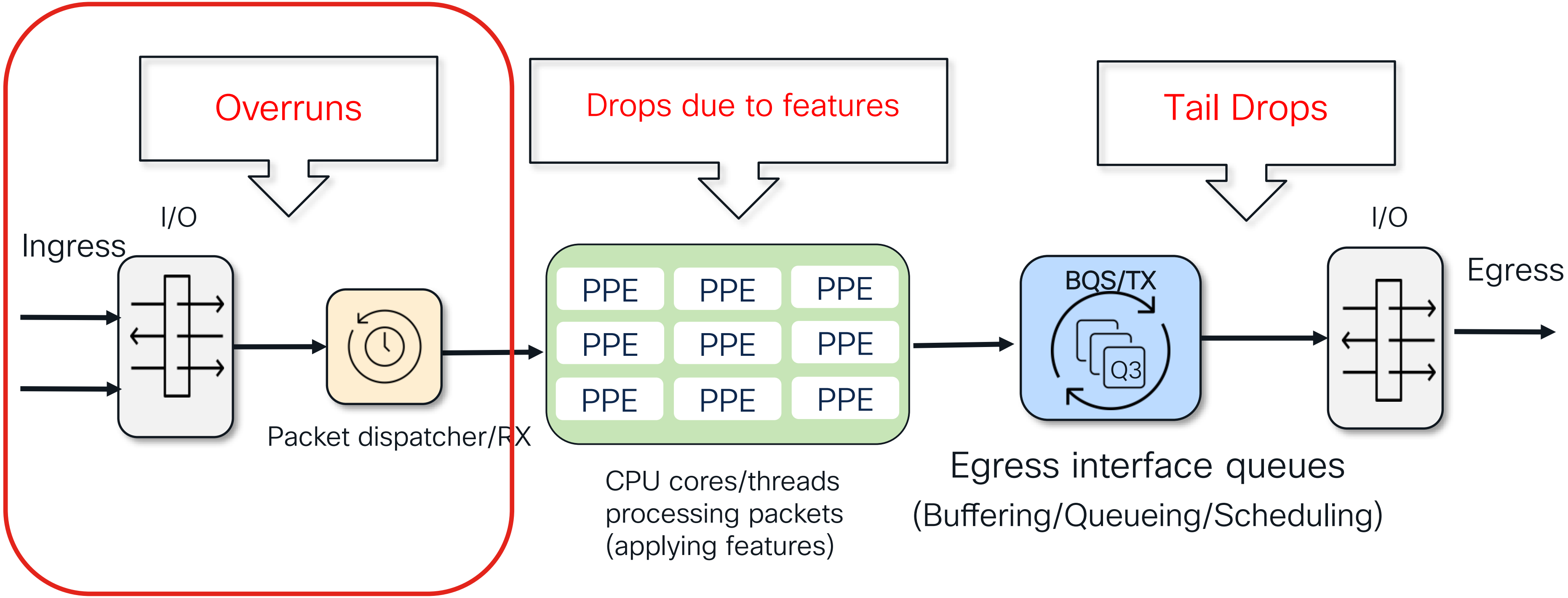
Drops due to features

Tail Drops

CPU cores/threads
processing packets
(applying features)

Egress interface queues
(Buffering/Queueing/Scheduling)

Packet drops – most common scenarios

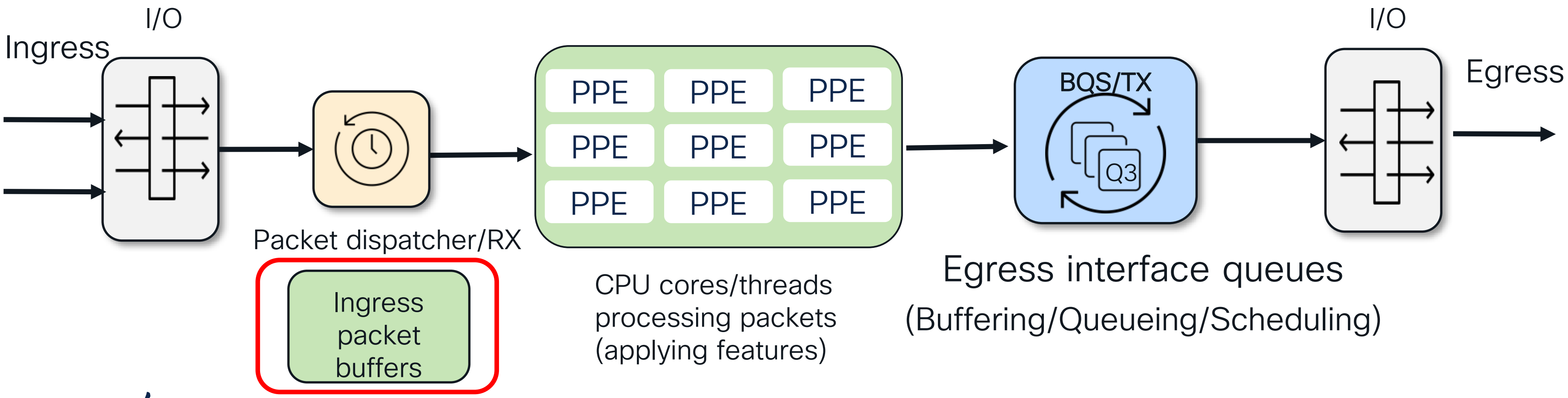


Overruns

```
GigabitEthernet0/0/0 is up, line protocol is up
[…]
     13464 input errors, 0 CRC, 0 frame, 13464 overrun, 0 ignored
```

Input drops due to no available resources to handle incoming traffic:

1) PPEs/CPU cores are busy processing packets
2) Ingress buffers are already occupied and cannot store new incoming packets



Troubleshooting Overruns

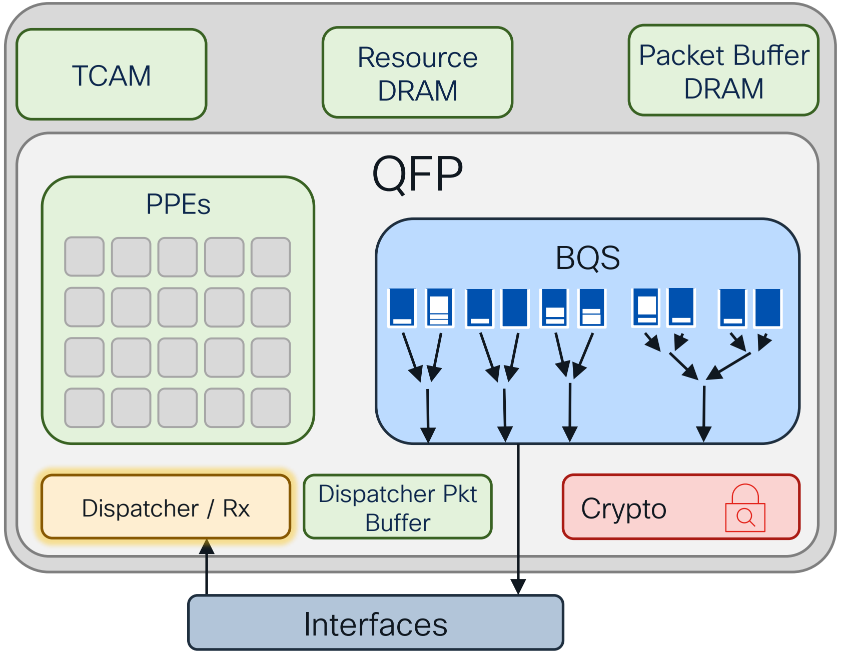
- ➢ QFP-based platforms
- ➢ x86 SoC platforms

Troubleshooting overruns on QFP based platforms

- All PPE threads are busy processing packets
- New packets need to wait for Dispatcher to find an available PPE thread
- Often times, QFP usage is very high (90%+)

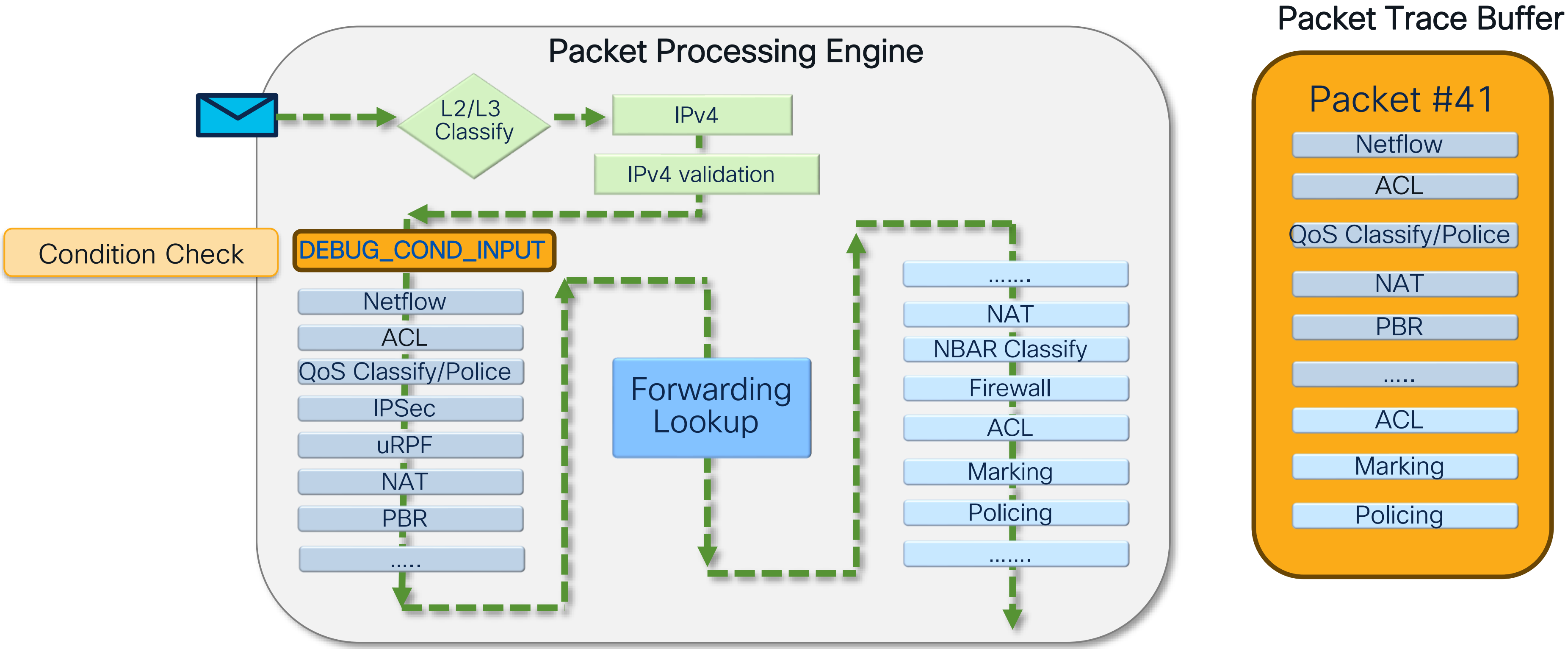
```
C8500#show plat hard qfp active datapath util summary
  CPP 0: Subdev 0                5 secs        1 min    …
Input:     Total (pps)           108837       111107    …
                 (bps)        711833352    713699424    …
Output:    Total (pps)           108332       109962    …
                 (bps)        722352024    723511272    …
Processing: Load (pct)               99           95
```

- Are PPEs taking more time than usual to process packets?
- **Next step:** QFP Profiling using **Packet Trace**



BRKTRS-3475

Packet Trace and FIA Debugging



For Your Reference

Packet Trace

- True inspection of IOS-XE packet forwarding flow
- Designed to be used in production, even in scaled setup
- Conditions define what the filters are and when the filters are applied to a packet
- Detailed report of what each configured feature did to packets matching the filter
- Can be used to trace dropped and punted packets as well

FIA Trace example

```
Packet: 0           CBUG ID: 0
Summary
  Input     : Port-channel1
  Output    : BD-VIF5086
  Timestamp
    Start   : 4423148105825975 ns (12/07/2020 11:00:46.156544 UTC)
    Stop    : 4423148105904766 ns (12/07/2020 11:00:46.156622 UTC)
Path Trace
 Feature: IPV4(Input)
    Input       : Port-channel1
    Output      : <unknown>
    Source      : 10.250.0.2
    Destination : 142.250.71.110
    Protocol    : 6 (TCP)
      SrcPort   : 41510
      DstPort   : 443
  Feature: DEBUG_COND_INPUT_PKT
    Entry       : Input - 0x800164e8
    Input       : Port-channel1
    Output      : <unknown>
    Lapsed time : 2336 ns
Feature: LAYER2_INPUT_VLAN_TAG_MANIPULATION
    Entry       : Input - 0x8001677c
    Input       : Port-channel1.EFP2115
    Output      : <unknown>
    Lapsed time : 2640 ns
…
```

Total time spent in PPE

Packet details

```
  …
Feature: IPV4_INPUT_VFR
    Entry       : Input - 0x80016a74
    Input       : BD-VIF7509
    Output      : <unknown>
    Lapsed time : 224 ns
Feature: Policy Based Routing
    PBR feature
    Route-map name: pbr-9297
    Seq number: 15
    Set precedence: 0
    Stats_addr: 0x424bf940
  Feature: IPV4_INPUT_PBR
    Entry       : Input - 0x80016adc
    Input       : BD-VIF7509
    Output      : <unknown>
    Lapsed time : 8640 ns
  Feature: IPV4_INPUT_LOOKUP_PROCESS
    Entry       : Input - 0x8001645c
    Input       : BD-VIF7509
    Output      : BD-VIF5086
    Lapsed time : 1232 ns
  Feature: IPV4_INPUT_IPOPTIONS_PROCESS
    Entry       : Input - 0x80016b38
    Input       : BD-VIF7509
    Output      : BD-VIF5086
    Lapsed time : 224 ns
  Feature: IPV4_INPUT_GOTO_OUTPUT_FEATURE
    Entry       : Input - 0x80016b5c
    Input       : BD-VIF7509
    Output      : BD-VIF5086
    Lapsed time : 736 ns
….
```

Feature applied

Time spent on this feature

Enabling Packet-trace

Packet Trace Configuration

```
Cat8k# debug platform condition ipv4 [interface] | [access-list]  | [ip_address] ingress

Cat8k# debug platform packet-trace packet <number of packets> fia-trace

Cat8k# debug platform condition start
```

For production use, also in scaled deployments.

Optionally:

```
Cat8k# debug platform packet-trace copy packet both size <…>
```

To dump L2/L3/L4 packet headers on ingress and egress

Packet Trace buffer:

```
Cat8k# show platform packet-trace summary
0     Gi0/0/2.25        Gi0/0/3          FWD
1     Gi0/0/2.25        Gi0/0/3          FWD
2     Tu1               Gi0/0/2.35       FWD
3     Gi0/0/2.21        Gi0/0/3          DROP   20  (QosPolicing)
4     Tu1               Gi0/0/2.35       FWD

Cat8k# show platform packet-trace packet <packet number>
```

Detailed information of specific packet handling within QFP

Case Study: Overruns with low traffic rate

- QFP usage exceeds 80% threshold, overruns are reported in ”show interface”

```
%IOSXE_QFP-2-LOAD_EXCEED: Slot: 0, QFP:0, Load 96% exceeds the setting threshold 80%.
5 secs traffic rate on QFP: Total Input: 100768 pps (100.8 kpps), 637917984 bps (637.9 mbps), Total Output:
99780 pps (99.8 kpps), 643689256 bps (643.7 mbps).
```

```
254829 input errors, 0 CRC, 0 frame, 254829 overrun, 0 ignored
```

- Next step: Use Packet Trace to collect a sample of traffic for analysis

```
C8k-Edge1#debug platform condition ingress

C8k-Edge1#debug platform packet-trace packet 8192 data-size 4096 fia-trace

C8k-Edge1#debug platform condition start


C8k-Edge1#show platform packet-trace statistics
Packets Summary
  Matched  134220
  Traced   8192
```

Match all incoming traffic

Capture entire FIA

Up to 8192 packets can be analyzed

QFP Profiling using Packet Trace

Sample packet:

```
Feature: IPV4_NAT_INPUT_FIA
    Entry        : Input - 0x80018204
    Input        : TenGigabitEthernet0/0/0
    Output       : <unknown>
    Lapsed time : 21468880 ns
```

Packet spent 21ms processed by NAT Input

```
Cat8k#show platform packet-trace fia-statistics
Feature                                      Count    Min(ns)   Max(ns)     Avg(ns)
----------------------------------------------------- ------------------------------
 IPV4_NAT_INPUT_FIA                            17      19408   27833968    8195994
 IPV4_NAT_OUTPUT_FIA                           85      17920      85824      35082
 IPV4_INPUT_QOS                                 9       6448      22592      15320
 ESI_BAF_TRANSMIT_PKT                         136      10160      33200      13867
 RELOOKUP_NOTIFY                                9       3376       4672       3909
 IPV4_OUTPUT_DROP_POLICY                      136       2192       3088       2731
 IPV4_INPUT_LOOKUP_PROCESS                    102       1920       2752       2354
<snip>
```

New CLI
in IOS-XE 17.11

Ingress NAT consuming significant amount of CPU time

Observation: There’s a lot of non-NATed traffic received on NAT-enabled interface.
Solution: increase NAT gatekeeper cache size to avoid having such traffic being processed by NAT.

```
ip nat settings gatekeeper-size 65536
```

Overruns on C8500-12X or C8500-12X4QC

- Slow increase of overruns might be observed in micro-bursty conditions
- Adjustments to **ingress buffers allocations** applied in newer software
- These changes were implemented in IOS-XE versions:
  - 17.9.6 and newer
  - 17.12.4 and newer
  - 17.15.1 and newer

Mitigating overruns

- Make sure output flow control is enabled:

```
(config-if)# plim qos input queue 0 pause enable
```

- PAUSE frame will be sent to the peer, this is a request to slow down with sending further traffic

- If flow control is working properly on **both** ends of the link the PAUSE frames should stop the overruns (other end could start tail dropping if the backpressure lasts long enough).

- If both PAUSE output and overrun counters increase, make sure that the connected device is properly respecting and responding to flow control.

Troubleshooting Overruns

- ➢ QFP-based platforms
- ➢ x86 SoC platforms

Troubleshooting overruns on x86 based platforms

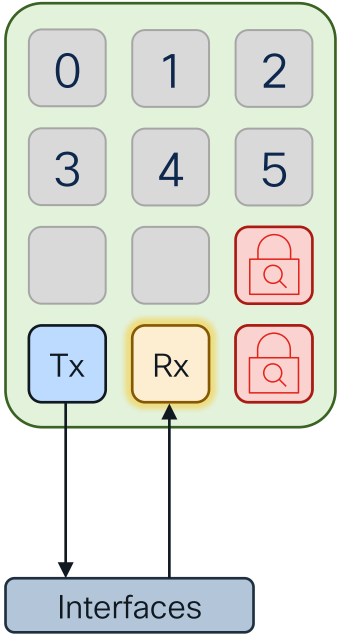
- Rx thread unable to distribute incoming packets to the relevant PP thread and ingress buffers are already full.

C8200/8300:

- All PP threads busy
- Rx thread is congested

C8500L:

- PP thread handling **this traffic** flow is busy
- Rx thread is congested



Case Study: Overruns on C8500L

- Customer migrated to C8500L and started to observe overruns on TenGigabit0/1/0

```
C8500L# show int Te0/1/0 | i overrun
     254829303 input errors, 0 CRC, 0 frame, 254829303 overrun, 0 ignored
```

General troubleshooting steps:

1. Verify the core allocation template in use
2. Determine if any CPU core/thread is reporting high utilization
3. Confirm which Rx thread/worker is assigned to the interface reporting overruns
4. Check Credit Errors

Dynamic Core Allocation (x86 based platforms)

1 Determine CPU allocation scheme:

```
C8500L# show platform software cpu alloc
CPU alloc information:

  Control plane cpu alloc: 0-1,12-13
  Data plane cpu alloc: 2-11,14-19
  Service plane cpu alloc: 0

  Slow control plane cpu alloc:
  Template used: default-data_plane_heavy
```

HyperThreading enabled
on some CPU cores
1 core = 2 threads

- The default mapping can be adjusted, if needed:

```
C8500L(config)# platform resource ?
app-heavy               Use App Heavy template
data-plane-heavy        Use Data Plane Heavy template
service-plane-heavy     Use Service Plane Heavy template
```

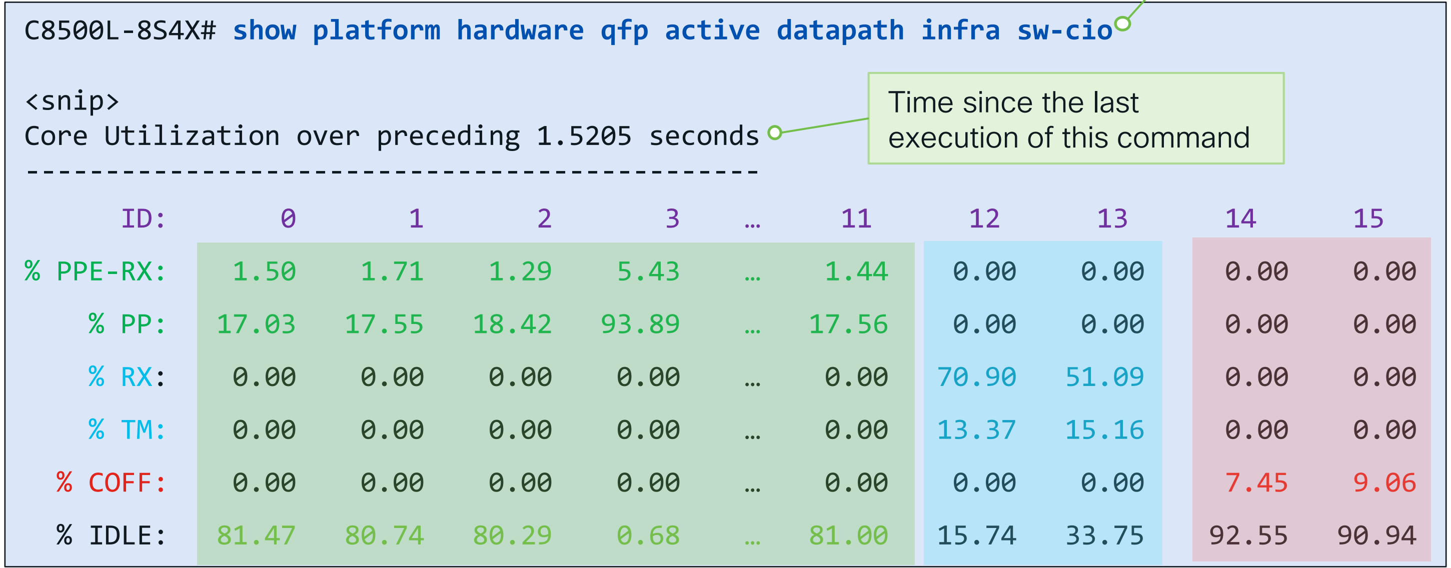
System default template
default-data_plane_heavy

User configured template
CLI-service_plane_heavy

Datapath CPU core/thread utilization

2 Determine % of CPU cycles spent on feature processing/Rx/Tx/Crypto

The goal is to identify potential bottleneck.



```
C8500L-8S4X# show platform hardware qfp active datapath infra sw-cio

<snip>
Core Utilization over preceding 1.5205 seconds
----------------------------------------------

      ID:       0       1       2       3    …     11      12      13      14      15
% PPE-RX:    1.50    1.71    1.29    5.43    …   1.44    0.00    0.00    0.00    0.00
    % PP:   17.03   17.55   18.42   93.89    …  17.56    0.00    0.00    0.00    0.00
    % RX:    0.00    0.00    0.00    0.00    …   0.00   70.90   51.09    0.00    0.00
    % TM:    0.00    0.00    0.00    0.00    …   0.00   13.37   15.16    0.00    0.00
  % COFF:    0.00    0.00    0.00    0.00    …   0.00    0.00    0.00    7.45    9.06
  % IDLE:   81.47   80.74   80.29    0.68    …  81.00   15.74   33.75   92.55   90.94
```

This command needs to be **executed at least twice**!

Time since the last execution of this command

CPU Thread/Worker IDs

Hashing/Distribution (C8500L)

Feature Processing

Rx functions

Traffic Manager (Tx functions)

Crypto functions

Rx/Tx thread mapping per interface

3 Confirm Rx thread ID assigned to the interface reporting overruns:

```
C8500L-8S4X# show platform hardware qfp active datapath infra binding
Port Instance Bindings:

  ID       Port                IOS Port   WRKR12 WRKR13
   1       rcl0                    rcl0     Rx      Tx
   2        ipc                     ipc     Tx      Rx
   3  vxe_punti              vxe_puntif     Tx      Rx
   4       fpe0      GigabitEthernet0/0/0     Tx      Rx
                                ......

                                ......
   8       fpe4      GigabitEthernet0/0/4     Rx      Tx
   9       fpe5      GigabitEthernet0/0/5     Tx      Rx
  10       fpe6      GigabitEthernet0/0/6     Rx      Tx
  11       fpe7      GigabitEthernet0/0/7     Tx      Rx
  12       fpe8   TenGigabitEthernet0/1/0     Rx      Tx
  13       fpe9   TenGigabitEthernet0/1/1     Tx      Rx
  14      fpe10   TenGigabitEthernet0/1/2     Rx      Tx
  15      fpe11   TenGigabitEthernet0/1/3     Tx      Rx
```

Rx/Tx mapping may vary across IOS-XE versions/platforms.

Credits system

- Each interface gets assigned a limited pool of credits (prevents a busy interface overloading the system resources).
- Each time a new packet arrives into the dataplane a credit is required.
- When packet processing is done, the credit is returned so Rx thread can use it again.

```
C8500L-8S4X# #show platform hardware qfp active datapath infrastructure sw-cio
Credits Usage:
  ID       Port  Wght  Global WRKR0  WRKR1  WRKR2  ... WRKR10  WRKR11  WRKR12 WRKR13  WRKR14 WRKR15 Total
   1       rcl0    1:   5849     0      0      0  ...      0       0     96     56       0      0   6029
   1       rcl0  128:   6048     0      0      0  ...      0       0     96      0       0      0  6144
   2        ipc    1:      0     0      0      0  ...      0       0      0      0       0      0     0
            …                                     ...
  11       fpe7    1:   1952     0      0      0  ...      0       0      0     96       0      0   2048
  11       fpe7    2:   1952     0      0      0  ...      0       0      0     96       0      0   2048
  12       fpe8    1:      0     0      0      0  ...      0       0      0      0       0      0      0
  12       fpe8    2:   1952     0      0      0  ...      0       0     96      0       0      0   2048
  13       fpe9    1:   1952     0      0      0  ...      0       0      0     96       0      0   2048
  13       fpe9    2:   1952     0      0      0  ...      0       0      0     96       0      0   2048
  14      fpe10    1:      0     0      0      0  ...      0       0     37      0       0      0     43
  14      fpe10    2:   1952     0      0      0  ...      0       0     96      0       0      0   2048
  15      fpe11    1:   1952     0      0      0  ...      0       0      0     96       0      0   2048
  15      fpe11    2:   1952     0      0      0  ...      0       0      0     96       0      0   2048
```

fpe8 ran out of credits

Credit Err counter

- If there’s no available credit for the interface the packet will need to wait in the interface Rx ring and Credit Err counter is incremented.

4

```
C8500L-2#show platform hardware qfp active datapath infrastructure sw-distrib
<snip>

Port 12, fpe8/TenGigabitEthernet0/1/0: Classifier: L4TUPLE, uidb:1015, Credit Err: 153838010
                 Flushes        Flushed         SW Hash          Total
  PP  0:           17998          25879           25879          25879
  PP  1:          592718         602277          602277         602277
  PP  2:           34366          44057           44057          44057
  PP  3:          211671         222721          222721         222721
  PP  4:           22707          34099           34099          34099
  ……
  PP 10:           16657          27015           27015          27015
  PP 11:          209707         216012          216012         216012
COFF  0:               -              -         9043333        9043333
```

- Rx is being blocked from pulling new packets into the system.
- If it is blocked long enough, the interface Rx rings will overflow resulting in input **overruns.**

Are we dealing with Elephant Flows?

```
show interface
show plat hard qfp active datapath infra sw-distrib
show plat hard qfp active datapath infra sw-cio
```

Collecting outputs periodically:

```
  254829303 input errors, 0 CRC, 0 frame, 254829303 overrun, 0 ignored

  Port 12, fpe8/TenGigabitEthernet0/1/0: Classifier: L4TUPLE, uidb:1015, Credit Err: 5451656

 ID:       0      1      2      3      4      5      6      7      8      9     10     11     12     13     14     15
%IDLE:   26.23  15.27  13.21  0.00  10.02  7.99  15.51  14.47  16.80  16.49  16.60  16.81  74.18  92.05  99.75  99.76
```

```
  462946846 input errors, 0 CRC, 0 frame, 462946846 overrun, 0 ignored

  Port 12, fpe8/TenGigabitEthernet0/1/0: Classifier: L4TUPLE, uidb:1015, Credit Err: 9457268

 ID:       0      1      2      3      4      5      6      7      8      9     10     11     12     13     14     15
%IDLE:   26.02  15.24  13.63  0.00  10.10  6.94  15.61  14.70  16.26  16.02  16.11  16.16  74.23  91.95  99.75  99.76
```

```
  565131966 input errors, 0 CRC, 0 frame, 565131966 overrun, 0 ignored

  Port 12, fpe8/TenGigabitEthernet0/1/0: Classifier: L4TUPLE, uidb:1015, Credit Err: 11576871

 ID:       0      1      2      3      4      5      6      7      8      9     10     11     12     13     14     15
%IDLE:   26.46  14.92  13.16  0.00  10.52  5.42  15.56  15.01  17.04  16.38  16.82  16.64  74.20  91.92  99.75   99.76
```

Observations: Credit Err counter increases along with overruns, PP #3 constantly fully utilized (Idle = 0%)

Are we dealing with Elephant Flows?

```
show interface
show plat hard qfp active datapath infra sw-distrib
show plat hard qfp active datapath infra sw-cio
```

Collecting outputs periodically:

```
  254829303 input errors, 0 CRC, 0 frame, 254829303 overrun, 0 ignored

  Port 12, fpe8/TenGigabitEthernet0/1/0: Classifier: L4TUPLE, uidb:1015, Credit Err: 5451656

 ID:       0      1      2      3      4      5      6      7      8      9      10     11     12     13     14     15
%IDLE:   26.23  15.27  13.21  0.00  10.02  7.99  15.51  14.47  16.80  16.49  16.60  16.81  74.18  92.05  99.75  99.76
```

```
  462946846 input errors, 0 CRC, 0 frame, 462946846 overrun, 0 ignored

  Port 12, fpe8/TenGigabitEthernet0/1/0: Classifier: L4TUPLE, uidb:1015, Credit Err: 9457268

 ID:       0      1      2      3      4      5      6      7      8      9      10     11     12     13     14     15
%IDLE:   26.02  15.24  13.63  0.00  10.10   6.94  15.61  14.70  16.26  16.02  16.11  16.16  74.23  91.95  99.75  99.76
```

```
  565131966 input errors, 0 CRC, 0 frame, 565131966 overrun, 0 ignored

  Port 12, fpe8/TenGigabitEthernet0/1/0: Classifier: L4TUPLE, uidb:1015, Credit Err: 11576871

 ID:       0      1      2      3      4      5      6      7      8      9      10     11     12     13     14     15
%IDLE:   26.46  14.92  13.16  0.00  10.52   5.42  15.56  15.01  17.04  16.38  16.82  16.64  74.20  91.92  99.75   99.76
```

Observations: Credit Err counter increases along with overruns, PP #3 constantly fully utilized (Idle = 0%)

C8500L placement guidance

- C8500L is best suited for deployments where the system is exposed to **high flow count.**
- The PPE-Rx function performs hashing based on internal header (2nd pass)

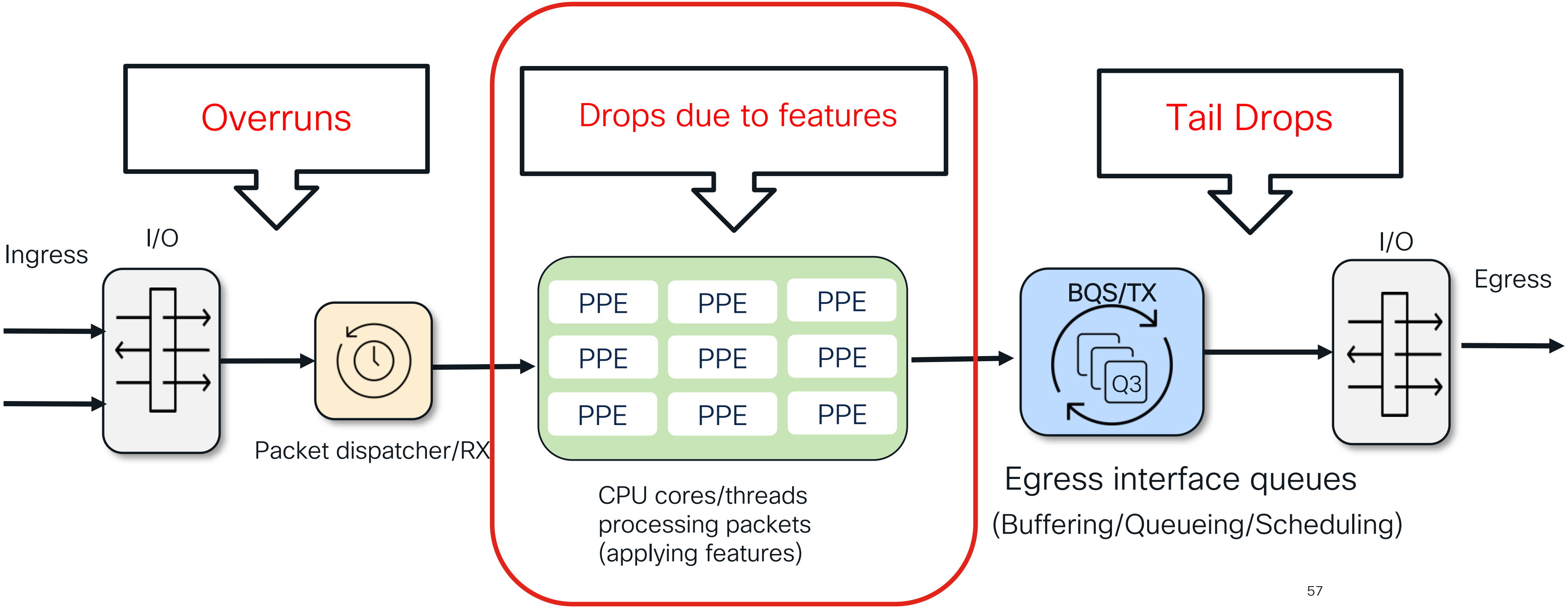
```
C8500L-8S4X#show platform hardware qfp active fbd-flowdb balance distribution

PP Flow Distribution
                             Flows
        PP      0:           19010
        PP      1:           21085
        PP      2:           21043
        PP      3:           21337
        PP      4:           21495
        PP      5:           21051
        PP      6:           20242
        PP      7:           20298
        PP      8:           20216
        PP      9:           20330
        PP      10:          20180
        PP      11:          20065
```

This command is available on C8500L only.

Most optimal performance with even distribution of traffic amongst all PP threads

Troubleshooting Packet Drops in PPE

Packet drops – most common scenarios



Packet Drops in PPEs

- Packets that arrive to the PPE may be dropped with specific QFP drop reason.

```
C8300#show platform hardware qfp active statistics drop
Last clearing of QFP drops statistics : never


-------------------------------------------------------------------------
Global Drop Stats                         Packets                  Octets
-------------------------------------------------------------------------
QosPolicing                                  4230                  177792
IpsecInput                                      5                     790
Ipv4NoRoute                                   334                   58502
```

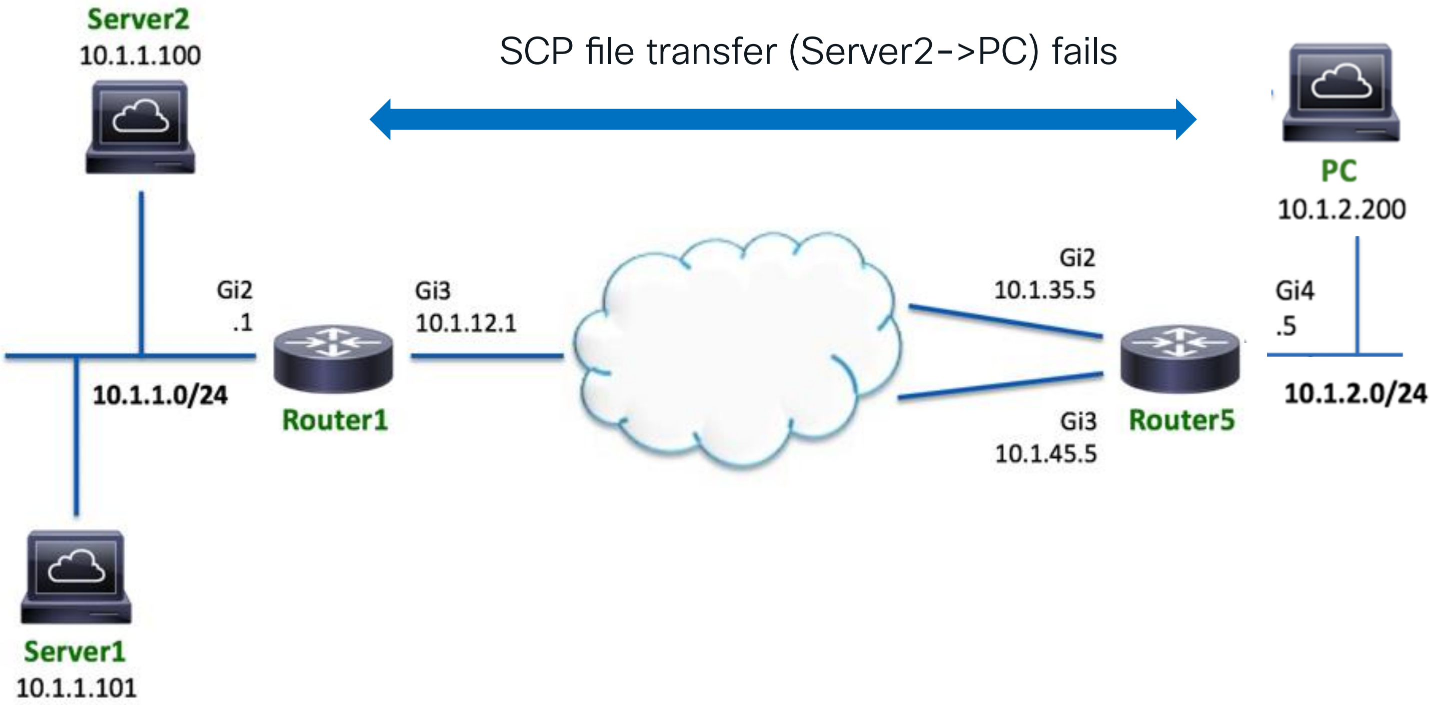
- Clear the accumulated drop counters to begin with:

```
C8300#show platform hardware qfp active statistics drop clear
```

- In IOS-XE 17.9 a simplified CLI is available:

```
C8500#show drops [options]
```

Case Study: File transfer getting stuck



Troubleshooting QFP drops with Packet Trace

1 Define condition, tracing level and buffer size on Cat8k router

```
Cat8k# debug platform condition ipv4 access-list ACL_SCP ingress

Cat8k# debug platform packet-trace packet 512 fia-trace

Cat8k# debug platform condition start
```

Trace packets matching this ACL

Trace 512 packets and stop, capture FIA details

2 Review the packet-trace summary

```
Cat8k# show platform packet-trace statistics

Cat8k# show platform packet-trace summary
```

3 Inspect individual packets

```
Cat8k# show platform packet-trace packet <packet#>
```

Packet Trace outputs

```
# show platform packet-trace statistics
Packets Summary
  Matched   18
  Traced    18
Packets Received
  Ingress   18
  Inject    0
Packets Processed
  Forward   6
  Punt      0
  Drop      12
    Count        Code  Cause
    12           187   FirewallPolicy
  Consume   0
```

```
# show platform packet-trace summary
0      Gi4         Gi3          FWD
1      Gi3         Gi4          DROP   187 (FirewallPolicy)
2      Gi4         Gi3          FWD
3      Gi3         Gi4          DROP   187 (FirewallPolicy)
4      Gi3         Gi4          DROP   187 (FirewallPolicy)
```

```
# show platform packet-trace packet 1

Path Trace
  Feature: IPV4(Input)
    Input       : GigabitEthernet3
    Output      : <unknown>
    Source      : 10.1.1.100
    Destination : 10.1.2.200
    Protocol    : 6 (TCP)
      SrcPort   : 22
      DstPort   : 60202

<…>

Feature: ZBFW
    Action   : Drop
    Reason   : Policy drop:classify result
    Zone-pair name            : WAN2_Inside
    Class-map name            : class-default
    Input interface           : GigabitEthernet3
    Egress interface          : GigabitEthernet4
```

This config needs to be verified

Serviceability enhancements: QFP drops history

Tracking QFP drops every 1 minute to determine trends:

```
Cat8000-1#show drops history

           or

Cat8000-1#show platform hardware qfp active statistics drop history
```

New CLI
in IOS-XE 17.13

```
Last clearing of QFP drops statistics : never
Last history counters update : Mon Jan 15 18:52:41 2025
 (47s ago)

-------------------------------------------------------------------------------
Global Drop Stats                1-Min            5-Min            30-Min            All
-------------------------------------------------------------------------------
TailDrops                          254             2441            532422        2552143
IpTtlExceeded                        1                1                 4            509
Ipv4Null0                          433             2171             13007        2129165
```

Serviceability enhancements: QFP drops thresholds

Syslog alert triggered when QFP drops threshold exceeded

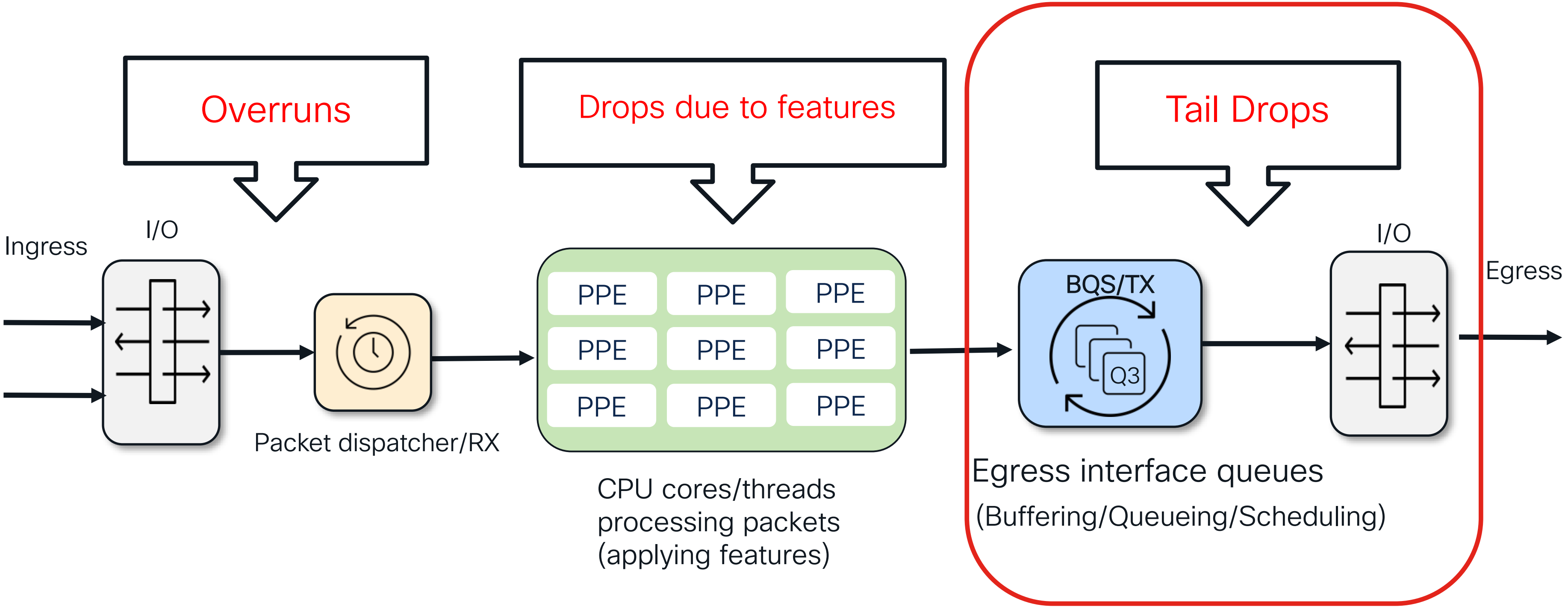
```
Cat8000-1(config)#platform qfp drops threshold ?
  per-cause   Set warning threshold for per cause QFP drops
  total       Set warning threshold for total QFP drops

Cat8000-1#show platform hardware qfp active statistics drop threshold
```

New CLI
in IOS-XE 17.14

```
%CPP_GIC_SVR-3-PERCAUSE_DROP_EXCEEDED: F0/0: cpp_cp_svr: Exceeded the drop threshold of 100
pps for Ipv4Null0 (drop code: 95) during the last 60-second measurement period. Packets
dropped due to Ipv4Null0 in last 1 minute: 439, last 5 minutes: 2171, last 30 minutes: 13007.


%CPP_GIC_SVR-3-TOTAL_DROP_EXCEEDED: F0/0: cpp_cp_svr: Exceeded the total drop threshold of
2500 pps during the last 60-second measurement period. Top 3 drop causes: Ipv4Null0,
QoSPolicing, IpTtlExceeded. Packets dropped in last 1 minute: 439, last 5 minutes: 2171, last
30 minutes: 13019.
```

Troubleshooting Tail Drops

Packet drops – most common scenarios



CPU cores/threads processing packets (applying features)

Egress interface queues (Buffering/Queueing/Scheduling)

Tail drops reasons

Tail drops indicate congestion on egress datapath

```
C8500#show platform hardware qfp active statistics drop
Last clearing of QFP drops statistics : never

--------------------------------------------------------------------------
Global Drop Stats                         Packets                   Octets
--------------------------------------------------------------------------
TailDrop                                   14230                   1277792
```

Congestion may occur due to:

- oversubscribing a shaper (e.g. class-default shaper setting)
- oversubscribing a physical interface
- backpressure (e.g. pause frames) sent by a peer device

Tail drops due to oversubscribed interface

- Tail drops occur when the internal queue limit for the egress interface is exceeded.

```
C8500L#show platform hardware qfp active infrastructure bqs interface GigabitEthernet 0/0/0 detail
Interface: GigabitEthernet0/0/0 QFP: 0.0 if_h: 10 Num Queues/Schedules: 1
  Queue specifics:
    Index 0 (Queue ID:0x70, Name: GigabitEthernet0/0/0)
    PARQ Software Control Info:
      (cache) queue id: 0x00000070, wred: 0xc6f6ebc0, qlimit (pkts ): 4210
     <snip>
    Statistics:
      tail drops  (bytes): 770040065195                  (packets): 520842994
      total enqs  (bytes): 20039977313838                (packets): 13713020916
      queue_depth (pkts ): 939
```

Size of the egress queue

Couldn't fit within the queue limit

Packets currently in the equeue

- The default queue limit depends on the bandwidth of an interface - can be overridden in configuration to reduce tail drops during brief periods of congestion.

- Increased queue limit will also increase latency of transmitted packets during periods of congestion.

TailDrops due to backpressure from peer

The pause inputs indicate the physical interface congestion is the result of back pressure from the directly connected peer device:

```
C8500L#sh int GigabitEthernet 0/0/0
GigabitEthernet0/0/0 is up, line protocol is up
<snip>
  output flow-control is on, input flow-control is on
<snip>
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 428856328
  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  30 second input rate 17000 bits/sec, 12 packets/sec
  30 second output rate 406106000 bits/sec, 214854 packets/sec
     651119 packets input, 117161693 bytes, 0 no buffer
     Received 1 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
     0 watchdog, 2663 multicast, 1602256 pause input
```

Pause frames from directly connected peer device

Traffic Manager (TM) utilization

```
C8500L#show platform harware qfp active datapath infrastructure sw-cio | begin Core

Core Utilization over preceding 7.1235 seconds
----------------------------------------------
      ID:      0       1       2   ……      10      11      12      13      14      15
% PPE-RX:   0.00    0.00    0.00   ……    0.00    0.00    0.00    0.00    0.00    0.00
    % PP:   0.28    0.54    0.57   ……    0.00    0.00    0.00    0.00    0.00    0.00
    % RX:   0.00    0.00    0.00   ……    0.00    0.00    0.00    0.78    0.00    0.00
    % TM:   0.00    0.00    0.00   ……    0.00    0.00  100.00    9.71    0.00    0.00
  % COFF:   0.00    0.00    0.00   ……    0.00    0.00    0.00    0.00    0.00    0.23
  % IDLE:  99.72   99.46   99.43   ……   99.85   99.85    0.00   89.51   99.75   99.77
```

TM (Tx) thread operating at 100%

```
C8500L#show platform harware qfp active datapath infrastructure sw-hqf

Name                   : Pri1 Pri2 None / Inflight pkts
GigabitEthernet0/0/0   : XON  XON  XOFF / 4175

HQF[0] IPC: send 14648 fc 0 congested_cnt 0
HQF[0] pkt: send hi 0 send lo 2761440507
          fc/full hi 0 fc/full lo 2758656
          cong hi 0 cong lo 1396909120
```

Packets accumulated in egress buffer

Congestion observed

For Your Reference

How to interpret TM utilization of 100%

Up until IOS-XE 17.16.x the TM thread utilization includes the cycles spent by TM polling the congested network interface until the congestion clears.

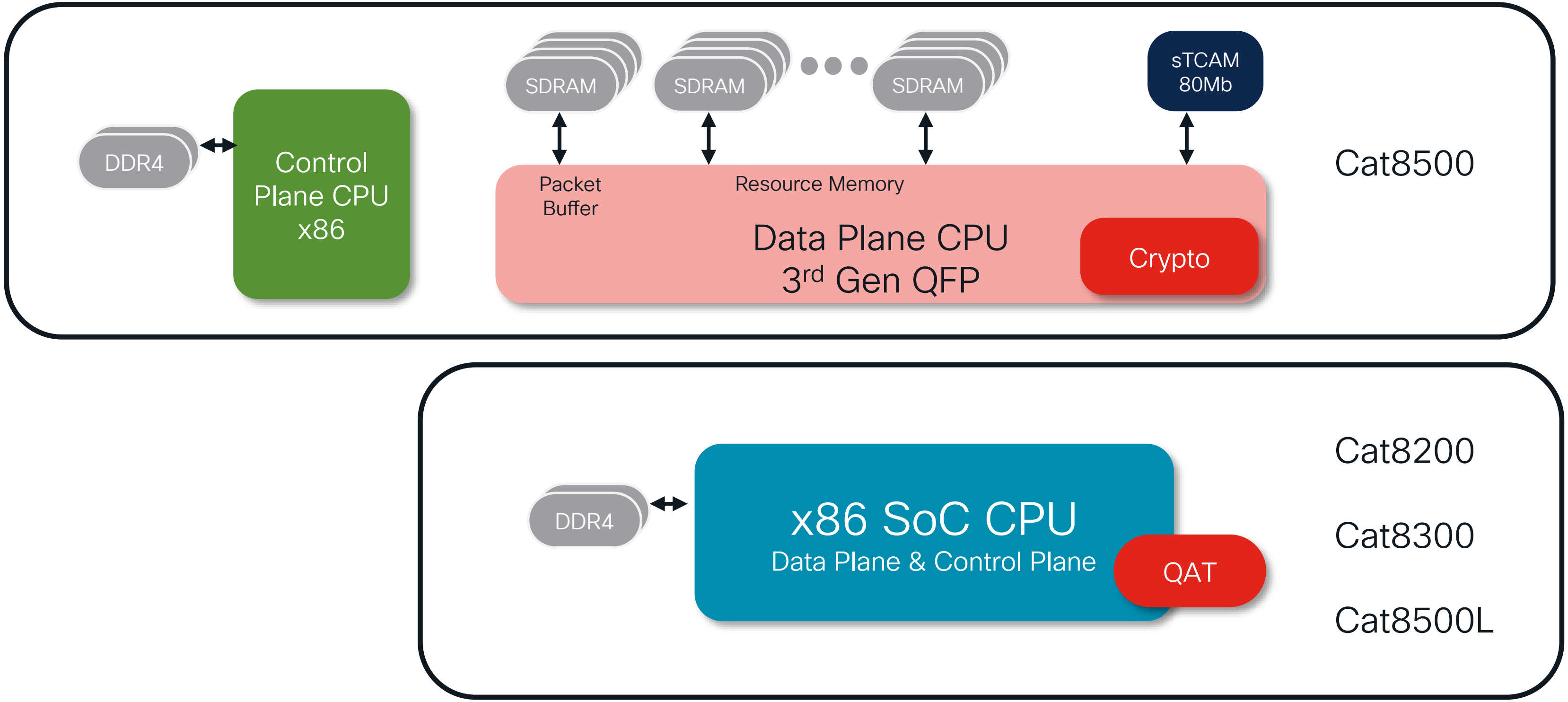
- 100% TM utilization indicates there's congestion on the physical network port. After servicing other ports the TM is dedicating the remaining cycles to polling the congested port to empty the queue.
- In this case it's "normal" for TM to reach 100%

In IOS-XE 17.17.1 onwards the TM CPU utilization calculation excludes the cycles spent on polling a congested network port.

- In this case if TM reaches 100% it indicates the TM got overwhelmed with work.

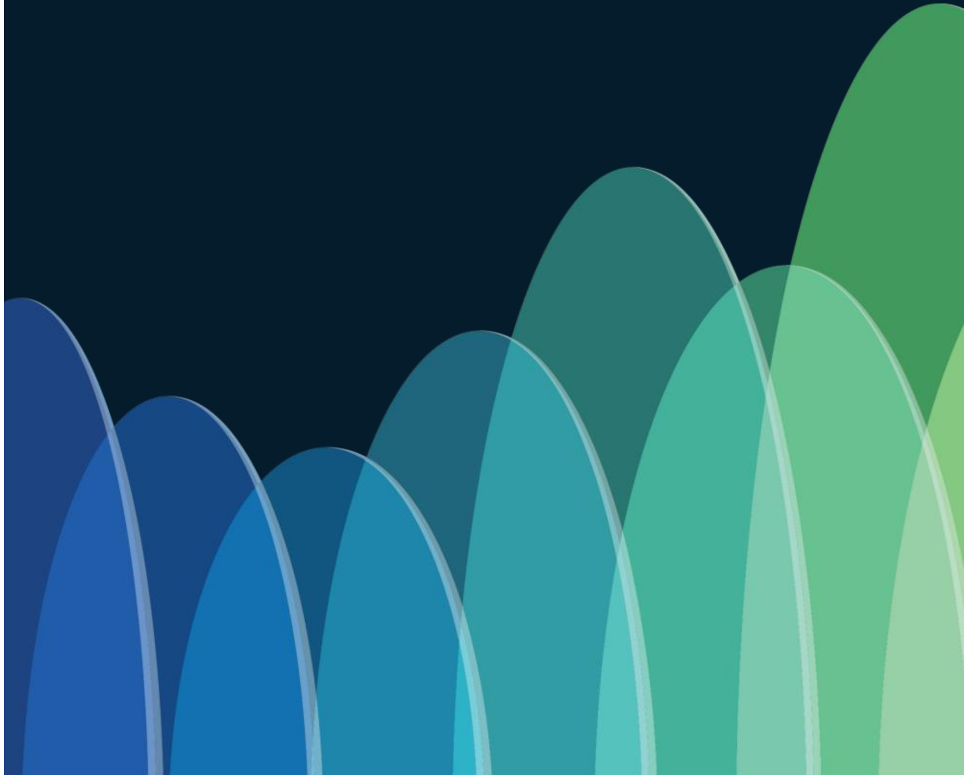
Platform Resources Verification

Control Plane vs Data Plane Resources



Resource Utilization

- ➢ Control Plane CPU
- ➢ Control Plane Memory
- ➢ Data Plane CPU
- ➢ Data Plane Memory



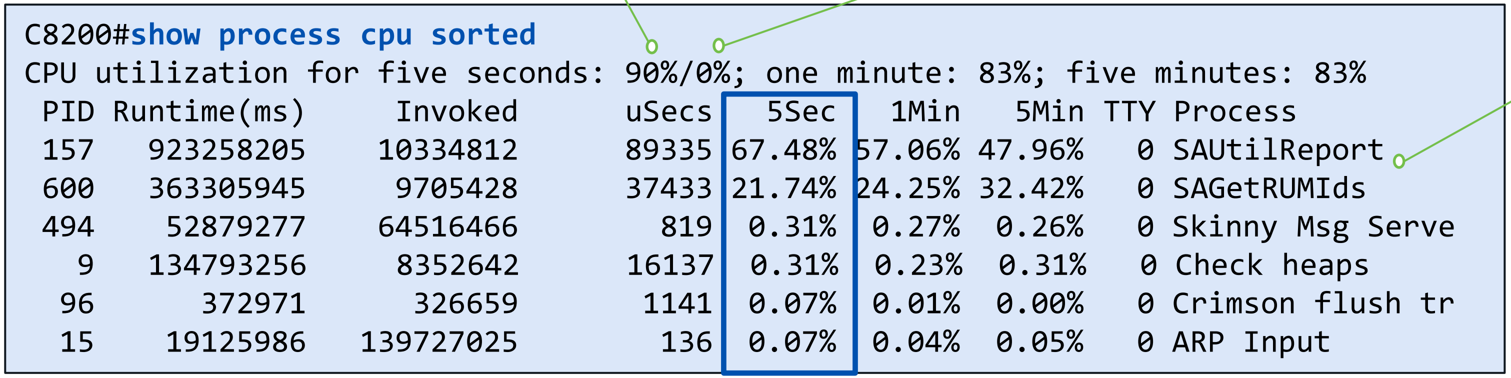
CPU utilization: IOS vs IOS-XE

IOS perspective

High CPU usage

CPU usage due to interrupts

SNMP OID: .1.3.6.1.4.1.9.2.1.56



```
C8200#show process cpu sorted
CPU utilization for five seconds: 90%/0%; one minute: 83%; five minutes: 83%
 PID Runtime(ms)     Invoked      uSecs   5Sec   1Min   5Min TTY Process
 157   923258205    10334812      89335 67.48% 57.06% 47.96%   0 SAUtilReport
 600   363305945     9705428      37433 21.74% 24.25% 32.42%   0 SAGetRUMIds
 494    52879277    64516466        819  0.31%  0.27%  0.26%   0 Skinny Msg Serve
   9   134793256     8352642      16137  0.31%  0.23%  0.31%   0 Check heaps
  96      372971      326659       1141  0.07%  0.01%  0.00%   0 Crimson flush tr
  15    19125986   139727025        136  0.07%  0.04%  0.05%   0 ARP Input
```

Processes consuming most IOSd CPU cycles

- Collect “show process cpu sorted” output periodically to identify the IOS process(es) consuming most CPU cycles during high CPU periods
- Look for patterns in historical CPU usage stats

CPU utilization: IOS vs IOS-XE

IOS perspective

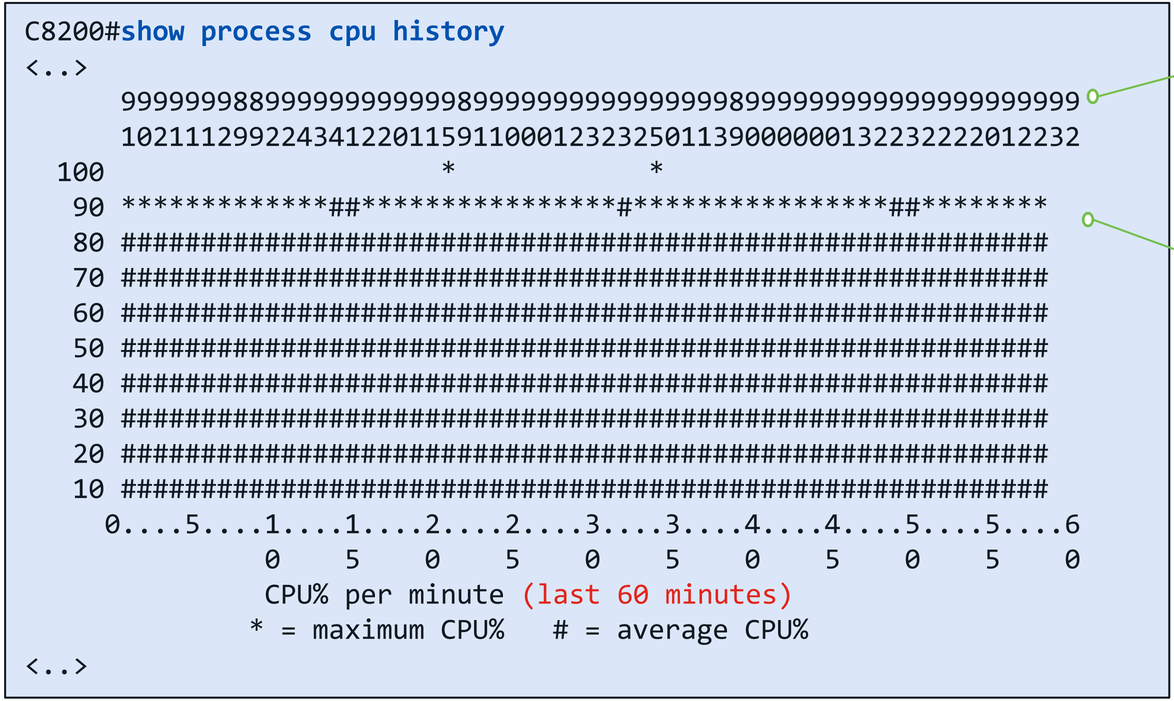
```
C8200#show process cpu history

      668888999999999888888888887777777778888888888888886666677777
      668888000033333333333333337777999998888886666666669999988888
  100
   90   *************                    ***************
   80   ************************************************     ***
   70 **********************************************************
   60 **********************************************************
   50 **********************************************************
   40 **********************************************************
   30 **********************************************************
   20 **********************************************************
   10 **********************************************************
     0....5....1....1....2....2....3....3....4....4....5....5....6
               0    5    0    5    0    5    0    5    0    5    0
               CPU% per second (last 60 seconds)
<..>
```

Max CPU usage captured in 1-second intervals

80% of CPU cycles for IOS constantly consumed

CPU utilization: IOS vs IOS-XE

IOS perspective



```
C8200#show process cpu history
<..>
      999999889999999999998999999999999999899999999999999999999999
      102111299224341220115911000123232501139000000132232222012232
  100                     *            *
   90 *************##****************#****************##********
   80 ##########################################################
   70 ##########################################################
   60 ##########################################################
   50 ##########################################################
   40 ##########################################################
   30 ##########################################################
   20 ##########################################################
   10 ##########################################################
     0....5....1....1....2....2....3....3....4....4....5....5....6
               0    5    0    5    0    5    0    5    0    5    0
               CPU% per minute (last 60 minutes)
              * = maximum CPU%   # = average CPU%
<..>
```

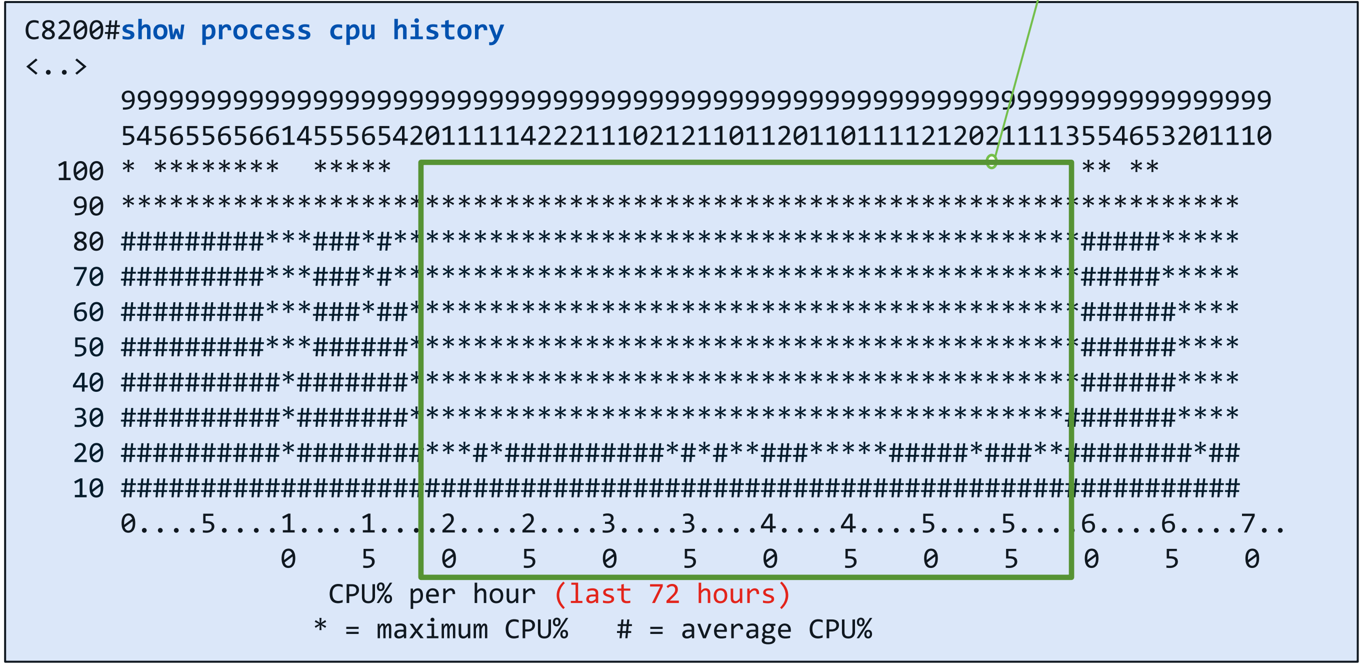
Max CPU usage within each 1-minute interval

Over past 60 minutes the average CPU usage on IOS side remained at 80%

CPU utilization: IOS vs IOS-XE

IOS perspective

In this period the average CPU usage remains low (~20%), occasional CPU spikes (up to 90+%) are not a concern



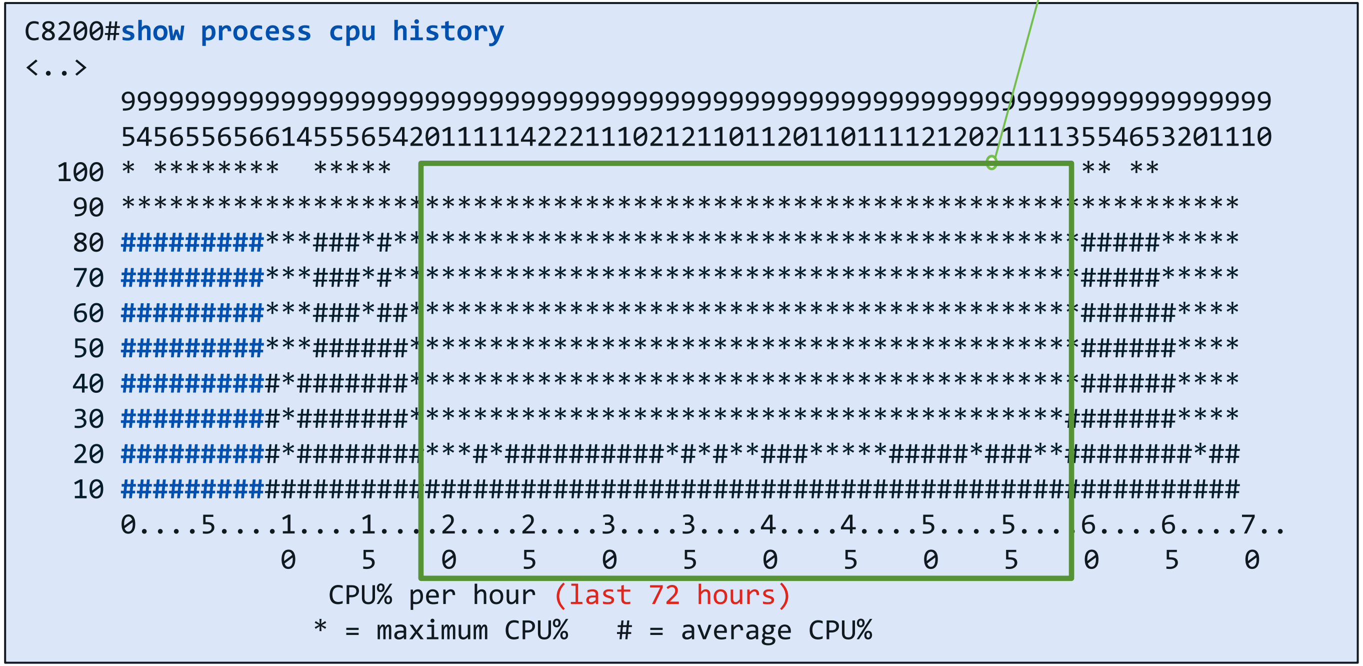
```
C8200#show process cpu history
<..>
      999999999999999999999999999999999999999999999999999999999999999999999999
      545655656614555654201111142221110212110112011011112120211113554653201110
  100 * ********  *****                                           ** **
   90 **********************************************************************
   80 #########***###*#*******************************************#####*****
   70 #########***###*#*******************************************#####*****
   60 #########***###*##******************************************######****
   50 #########***######******************************************######****
   40 ##########*#######******************************************######****
   30 ##########*#######*****************************************#######****
   20 ##########*########***#*##########*#*#**###*****#####*###**########*##
   10 ######################################################################
      0....5....1....1....2....2....3....3....4....4....5....5....6....6....7..
                0    5    0    5    0    5    0    5    0    5    0    5    0
                   CPU% per hour (last 72 hours)
                  * = maximum CPU%   # = average CPU%
```

CPU utilization: IOS vs IOS-XE

IOS perspective

In this period the average CPU usage remains low (~20%), occasional CPU spikes (up to 90+%) are not a concern

For the past 8 hours IOS was consuming 80% CPU cycles (on average)



```
C8200#show process cpu history
<..>
      999999999999999999999999999999999999999999999999999999999999999999999999
      545655656614555654201111142221110212110112011011112120211113554653201110
  100 * ********  *****                                           ** **
   90 **********************************************************************
   80 #########***###*#*******************************************#####*****
   70 #########***###*#*******************************************#####*****
   60 #########***###*##*******************************************######****
   50 #########***######*******************************************######****
   40 ##########*#######*******************************************######****
   30 ##########*#######*****************************************#######****
   20 ##########*########***#*##########*#*#**###*****#####*###**########*##
   10 ######################################################################
      0....5....1....1....2....2....3....3....4....4....5....5....6....6....7..
                0    5    0    5    0    5    0    5    0    5    0    5    0
                   CPU% per hour (last 72 hours)
                  * = maximum CPU%   # = average CPU%
```

High CPU utilization (IOS) investigation

- Define CPU threshold to produce syslog alert, for example:

```
(config)# process cpu threshold type total rising 80 interval 5
```

Example alert:

```
Jul 25 22:43:52: %SYS-1-CPURISINGTHRESHOLD: Threshold: Total CPU Utilization(Total/Intr):
93%/2%, Top 3 processes(Pid/Util):  747/76%, 325/3%, 573/2%
```

Top 3 IOS processes consuming most CPU cycles

```
CPU utilization for five seconds: 93%/2%; one minute: 27%; five minutes: 22%
 PID Runtime(ms)     Invoked      uSecs    5Sec    1Min    5Min TTY Process
 747     19467984    13889967      1401 76.33%  7.33%  1.64%   0 BGP Task
 325      3623133     6540482       553  3.86%  1.69%  0.41%   0 IP RIB Update
 573     38913959   622896760        62  2.41%  1.71%  1.44%   0 BGP Router
```

High CPU utilization (IOS) investigation

General Procedure

- During the high CPU usage period:
  - Identify processes (features) that consume most CPU cycles
  - Collect IOS tracelogs and feature-specific debugs/outputs
  - `show stack <PID>` will capture the call trace of PCs (functions) executed at that moment

```
#show stack <PID>

Process 761:  SNMP ENGINE
Tracekey : 1#4c4803d2767f5c964caa60fcdc63d5a3
  Stack segment 0x7FA50FDBA000 - 0x7FA50FDD1700
        RSP: 0x7FA50FDD0F80, PC: :5584BCAB1000+9FB7810
        RSP: 0x7FA50FDD0FC0, PC: :5584BCAB1000+8C15D9D
        RSP: 0x7FA50FDD1090, PC: :5584BCAB1000+6C46949
        RSP: 0x7FA50FDD1340, PC: :5584BCAB1000+6C46635
        RSP: 0x7FA50FDD1410, PC: :5584BCAB1000+87B0514
        RSP: 0x7FA50FDD14D0, PC: :5584BCAB1000+876C5AA
        RSP: 0x7FA50FDD15A0, PC: :5584BCAB1000+8753022
```

Collect a few instances of this output for better accuracy

High CPU utilization (IOS) investigation

Automated data collection via EEM

- Embedded Event Manager (EEM) applet can by triggered by:
  - Syslog message (after applying `"process cpu threshold…"` config)

```
event manager applet CPUMON authorization bypass
 event syslog pattern "%SYS-1-CPURISINGTHRESHOLD" ratelimit 300
 action 1.0 syslog msg "Collecting Diagnostics Data for High CPU usage"
 action 1.1 cli command "enable"
 action 1.2 cli command "terminal exec prompt timestamp”
 action 1.3 cli command ”show process cpu sorted | append bootflash:cpumon.txt”
 …
```

  - SNMP OID

```
event manager applet CPUMON_OID authorization bypass
 event snmp oid 1.3.6.1.4.1.9.2.1.56 get-type exact entry-op ge entry val 85 poll-interval 10
 action 1.1 cli command …
```

SoC platforms

CPU utilization: IOS vs IOS-XE

IOS-XE perspective (Linux kernel view)

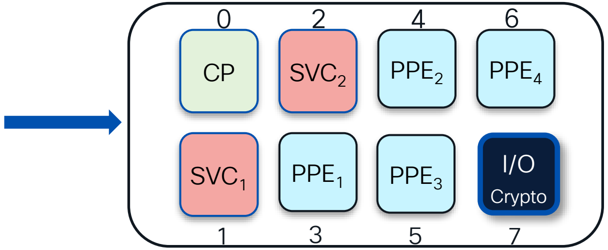
```
C8200-2#show process cpu platform sorted
CPU utilization for five seconds:  5%, one minute:  5%, five minutes: 10%
Core 0: CPU utilization for five seconds:  7%, one minute: 10%, five minutes: 10%
Core 1: CPU utilization for five seconds:  0%, one minute:  0%, five minutes:  0%
Core 2: CPU utilization for five seconds:  0%, one minute:  0%, five minutes:  0%
Core 3: CPU utilization for five seconds:  2%, one minute:  2%, five minutes:  2%
Core 4: CPU utilization for five seconds:  3%, one minute:  3%, five minutes:  4%
Core 5: CPU utilization for five seconds: 11%, one minute: 11%, five minutes: 12%
Core 6: CPU utilization for five seconds:  2%, one minute:  2%, five minutes:  2%
Core 7: CPU utilization for five seconds: 16%, one minute: 15%, five minutes: 53%
   Pid    PPid    5Sec    1Min    5Min  Status        Size  Name
--------------------------------------------------------------------------------
 19113   19100     67%     67%     67%  S           205272  ucode_pkt_PPE0
  3861    3845      3%      2%      2%  S           659528  linux_iosd-imag
```

Control Plane on Core 0 (used by IOSd)

Service Plane CPUs remain idle

This dataplane process consumes CPU cycles polling for new packets, it's expected to see high CPU usage here

```
C8200-2#show platform software cpu alloc
CPU alloc information:

 Control plane cpu alloc: 0

 Data plane cpu alloc: 3-7

 Service plane cpu alloc: 1-2
```



SoC platforms

CPU utilization: IOS vs IOS-XE

IOS-XE perspective – **SoC platforms**

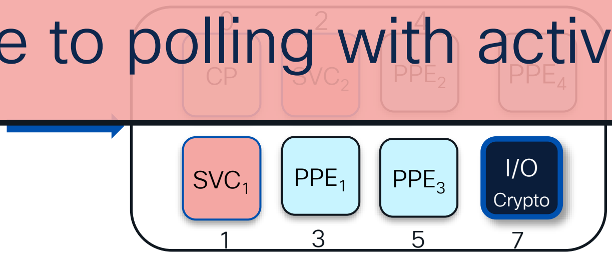
```
C8200#show process cpu platform sorted
CPU utilization for five seconds:  5%, one minute:  5%, five minutes: 10%
Core 0: CPU utilization for five seconds:  7%, one minute: 10%, five minutes: 10%
Core 1: CPU utilization for five seconds:  0%, one minute:  0%, five minutes:  0%
Core 2: CPU utilization for five seconds:  0%, one minute:  0%, five minutes:  0%
Core 3: CPU utilization for five seconds:  2%, one minute:  2%, five minutes:  2%
Core 4: CPU utilization for five seconds:  3%, one minute:  3%, five minutes:  4%
Core 5: CPU utilization for five seconds: [illegible], one minute: [illegible], five minutes: [illegible]
Core 6: CPU utilization for five seconds:  2%, one minute:  2%, five minutes:  2%
Core 7: CPU utilization for five seconds: 10%, one minute: [illegible], five minutes: [illegible]
   Pid    PPid    5Sec    1Min    5Min  Status        Size  Name
--------------------------------------------------------------------------------
 19113   19100     67%     67%     67%  S           205272  ucode_pkt_PPE0
  3861    3845      2%      2%      2%  S           659528  linux_iosd-imag
```

Control Plane on Core 0 (used by IOSd)

Service Plane cores remain idle

Data Plane cores will be polling for new work

```
C8200#show platform software cpu alloc
CPU alloc information:

  Control plane cpu alloc: 0
  Data plane cpu alloc: 3-7
  Service plane cpu alloc: 1-2
```



On SoC platforms the “show process cpu platform sorted” command is **not** really useful for CPU usage monitoring, due to DPDK characteristics.

The underlying Linux OS cannot distinguish between CPU core being busy due to polling with active or idle results.

For Your Reference

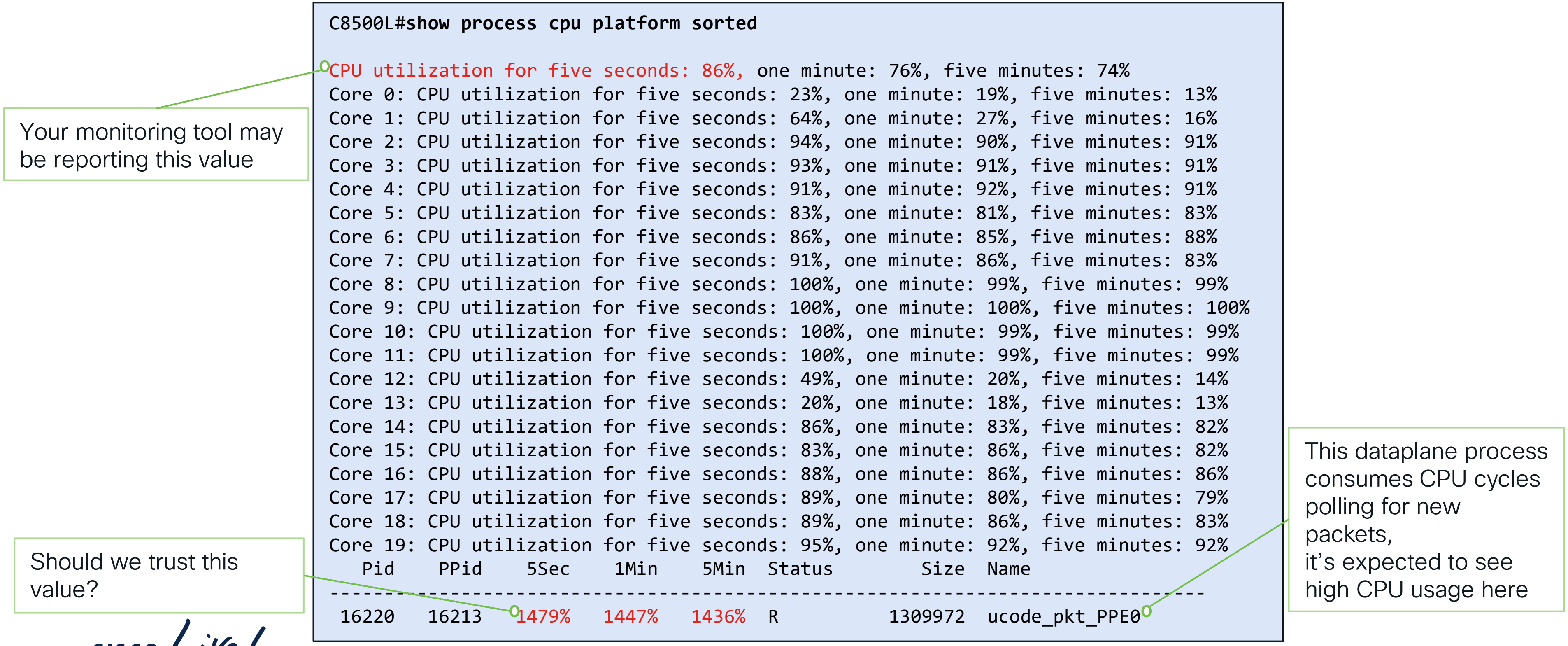
DPDK (Data Plane Development Kit) overview

- Set of libraries and drivers to accelerate packet processing on general purpose CPUs.
  - Packet processing is pushed to user space of the operating system
  - Applications can directly access network interface cards (NICs)
- Polling Mode Drivers (PMDs) constantly check for new packets (CPU doesn’t wait for interrupt signals)
- Core Affinity – specific CPU cores assigned to handle packet processing

CPU utilization: IOS vs IOS-XE

SoC platforms

Confusing CPU usage statistics



```
C8500L#show process cpu platform sorted

CPU utilization for five seconds: 86%, one minute: 76%, five minutes: 74%
Core 0: CPU utilization for five seconds: 23%, one minute: 19%, five minutes: 13%
Core 1: CPU utilization for five seconds: 64%, one minute: 27%, five minutes: 16%
Core 2: CPU utilization for five seconds: 94%, one minute: 90%, five minutes: 91%
Core 3: CPU utilization for five seconds: 93%, one minute: 91%, five minutes: 91%
Core 4: CPU utilization for five seconds: 91%, one minute: 92%, five minutes: 91%
Core 5: CPU utilization for five seconds: 83%, one minute: 81%, five minutes: 83%
Core 6: CPU utilization for five seconds: 86%, one minute: 85%, five minutes: 88%
Core 7: CPU utilization for five seconds: 91%, one minute: 86%, five minutes: 83%
Core 8: CPU utilization for five seconds: 100%, one minute: 99%, five minutes: 99%
Core 9: CPU utilization for five seconds: 100%, one minute: 100%, five minutes: 100%
Core 10: CPU utilization for five seconds: 100%, one minute: 99%, five minutes: 99%
Core 11: CPU utilization for five seconds: 100%, one minute: 99%, five minutes: 99%
Core 12: CPU utilization for five seconds: 49%, one minute: 20%, five minutes: 14%
Core 13: CPU utilization for five seconds: 20%, one minute: 18%, five minutes: 13%
Core 14: CPU utilization for five seconds: 86%, one minute: 83%, five minutes: 82%
Core 15: CPU utilization for five seconds: 83%, one minute: 86%, five minutes: 82%
Core 16: CPU utilization for five seconds: 88%, one minute: 86%, five minutes: 86%
Core 17: CPU utilization for five seconds: 89%, one minute: 80%, five minutes: 79%
Core 18: CPU utilization for five seconds: 89%, one minute: 86%, five minutes: 83%
Core 19: CPU utilization for five seconds: 95%, one minute: 92%, five minutes: 92%
   Pid    PPid    5Sec    1Min    5Min  Status        Size  Name
--------------------------------------------------------------------------------
 16220   16213   1479%   1447%   1436%  R          1309972  ucode_pkt_PPE0
```

Your monitoring tool may be reporting this value

Should we trust this value?

This dataplane process consumes CPU cycles polling for new packets,
it's expected to see high CPU usage here

CPU utilization: IOS vs IOS-XE

IOS-XE perspective – **SoC platforms**

SoC platforms

- In IOS-XE 17.13.1 onwards - enhanced CLI to avoid confusion when monitoring CPU usage

```
C8200#show process cpu platform sorted profile ?
  CP   Show CPU usage for Control Plane
  DP   Show CPU usage for Data Plane
  SP   Show CPU usage for Service Plane

C8200#show process cpu platform sorted profile cp
CPU utilization for five seconds:  6%, one minute: 13%, five minutes: 12%
Core 0: CPU utilization for five seconds:  6%, one minute: 13%, five minutes: 12%
Control plane process utilization for five seconds:  8%, one minute: 15%, five
minutes: 14%
   Pid    PPid    5Sec    1Min    5Min  Status        Size  Name
--------------------------------------------------------------------------------
  3972    3960      2%      3%      2%  R           730220  linux_iosd-imag
 18439   18417      1%      1%      1%  S           178256  fman_fp_image
 ...
```

New CLI
in IOS-XE 17.13

Only CPU core(s) involved in control plane processing are displayed

- For dataplane CPU utilization there are better ways to monitor performance (will be covered in the next section).

CPU utilization: IOS vs IOS-XE

IOS-XE perspective – **QFP based platforms**

QFP-based platforms

```
C8500-1#show process cpu platform sorted
CPU utilization for five seconds:  3%, one minute:  3%, five minutes:  4%
Core 0: CPU utilization for five seconds: 12%, one minute:  4%, five minutes:  2%
Core 1: CPU utilization for five seconds: 11%, one minute:  2%, five minutes: 11%
Core 2: CPU utilization for five seconds:  8%, one minute: 16%, five minutes:  7%
Core 3: CPU utilization for five seconds:  8%, one minute:  1%, five minutes: 18%
Core 4: CPU utilization for five seconds:  5%, one minute:  5%, five minutes:  6%
Core 5: CPU utilization for five seconds:  3%, one minute:  2%, five minutes:  1%
Core 6: CPU utilization for five seconds:  8%, one minute:  2%, five minutes:  2%
Core 7: CPU utilization for five seconds:  6%, one minute:  5%, five minutes:  5%
Core 8: CPU utilization for five seconds:  7%, one minute:  3%, five minutes:  1%
Core 9: CPU utilization for five seconds: 47%, one minute:  4%, five minutes:  1%
Core 10: CPU utilization for five seconds:  2%, one minute:  0%, five minutes:  0%
Core 11: CPU utilization for five seconds:  8%, one minute:  1%, five minutes:  3%
Core 12: CPU utilization for five seconds:  3%, one minute:  3%, five minutes:  3%
Core 13: CPU utilization for five seconds:  7%, one minute:  1%, five minutes:  1%
Core 14: CPU utilization for five seconds:  3%, one minute:  0%, five minutes:  0%
Core 15: CPU utilization for five seconds: 20%, one minute:  9%, five minutes:  2%
   Pid    PPid    5Sec    1Min    5Min  Status        Size  Name
--------------------------------------------------------------------------------
  4490    4447     69%     37%     47%  S          3120392  linux_iosd-imag
 23472   23465     15%      9%      9%  S           154132  mcpcc-lc-ms
 19746   19734     10%      3%      3%  S           973448  fman_fp_image
```

None of these CPU cores is involved in datapath/forwarding functions (packet processing handled by QFP).

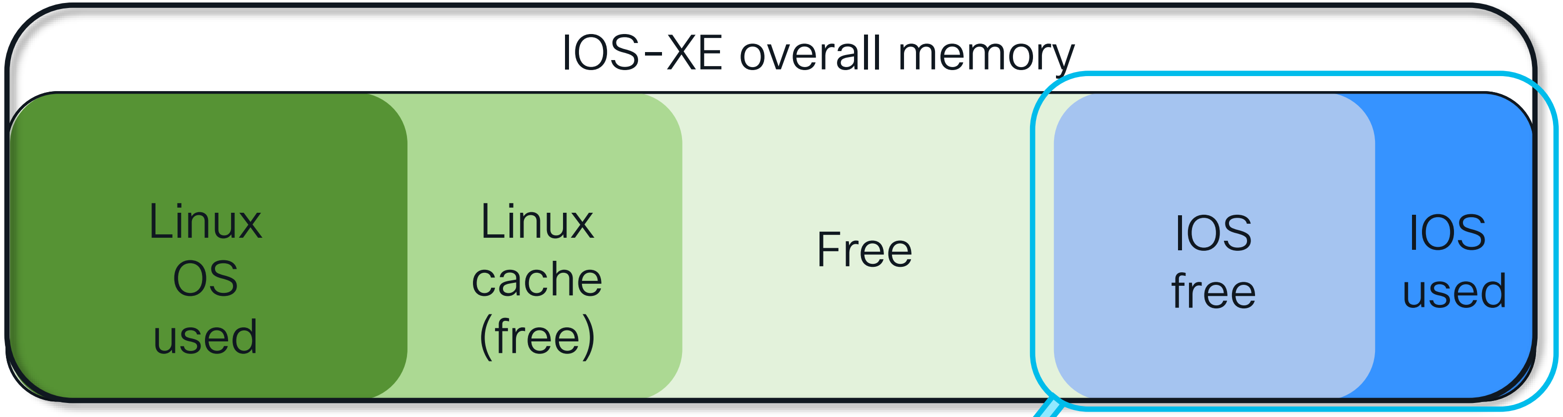
SNMP OID: .1.3.6.1.4.1.9.9.109.1.1.2.1.3

Resource Utilization

- ➢ Control Plane CPU
- ➢ Control Plane Memory
- ➢ Data Plane CPU
- ➢ Data Plane Memory

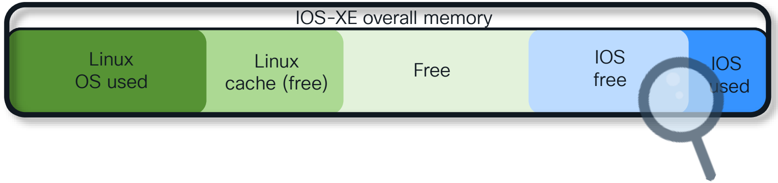
IOS-XE vs IOS memory usage

Control Plane + Management Plane



```
#show version
…
cisco C8500-12X4QC (1GD) processor (revision 1GD) with 6755559K/6147K bytes of memory.
…
16777216K bytes of physical memory.
```

IOS memory usage



```
#show memory statistics

Tracekey : 1#cc3dd7de68a09bce3a76a3e96c1758af

               Head       Total(b)     Used(b)     Free(b)    Lowest(b)   Largest(b)
Processor  76D24DB72048  6917553548   499355388  6418198160  5416976620  6417925616
reserve P  76D24DB720A0      102404          92      102312      102312      102312
 lsmpi_io  76D23726B1A8     6295128     6294304         824         824         412
```

Lowest free memory since last boot

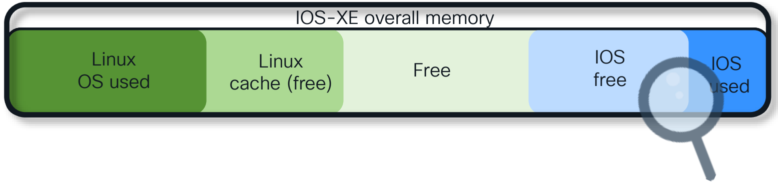
Largest available free memory block

```
#show process memory sorted

Processor Pool Total: 6917553548 Used:   499414136 Free: 6418139412
reserve P Pool Total:     102404 Used:          88 Free:     102316
 lsmpi_io Pool Total:    6295128 Used:     6294296 Free:        832

 PID TTY  Allocated      Freed    Holding    Getbufs    Retbufs Process
   0   0  375993784   40532056  309544720          0          0 *Init*
 735   0   51421976        592   51343568          0          0 PPPoE Background
 699   0   33989568      51720   34313848          0          0 SBC main process
```

IOS processes only

IOS memory usage

Top memory allocators



```
#show memory allocating totals

Tracekey : 1#cc3dd7de68a09bce3a76a3e96c1758af

Allocator PC Summary for: Processor

   Total     Count    Name                    PC
 33554528        1   Init                    :5ACE3CFFA000+9AA7431
 29691840      751   *Init*                  :5ACE3CFFA000+9A43778
 29069568     9768   *Packet Header*         :5ACE3CFFA000+CE54937
 28063240     9528   *Packet Data*           :5ACE3CFFA000+CE5498E
…
```

“Tracekey” encodes the IOS-XE process and IOS-XE version

Alloc PC represents a specific function in the source code. It can be decoded by Cisco TAC

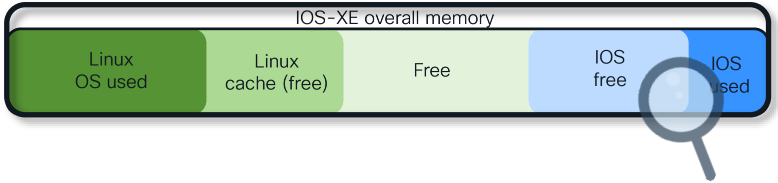
Total amount of memory allocated by given PC

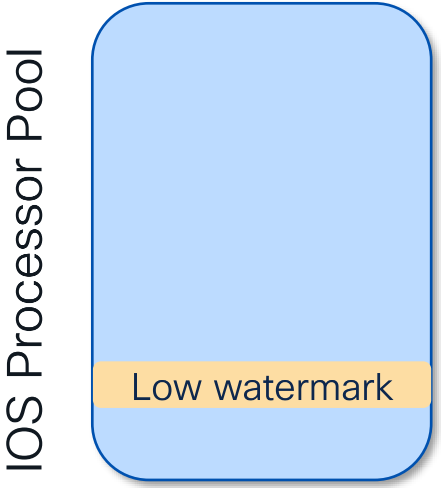
“Name” might give us a clue about top IOS memory consumer

For Your Reference

IOS memory usage

Low memory watermark





- IOS syslog generated when IOS free memory drops below the pre-configured low memory watermark

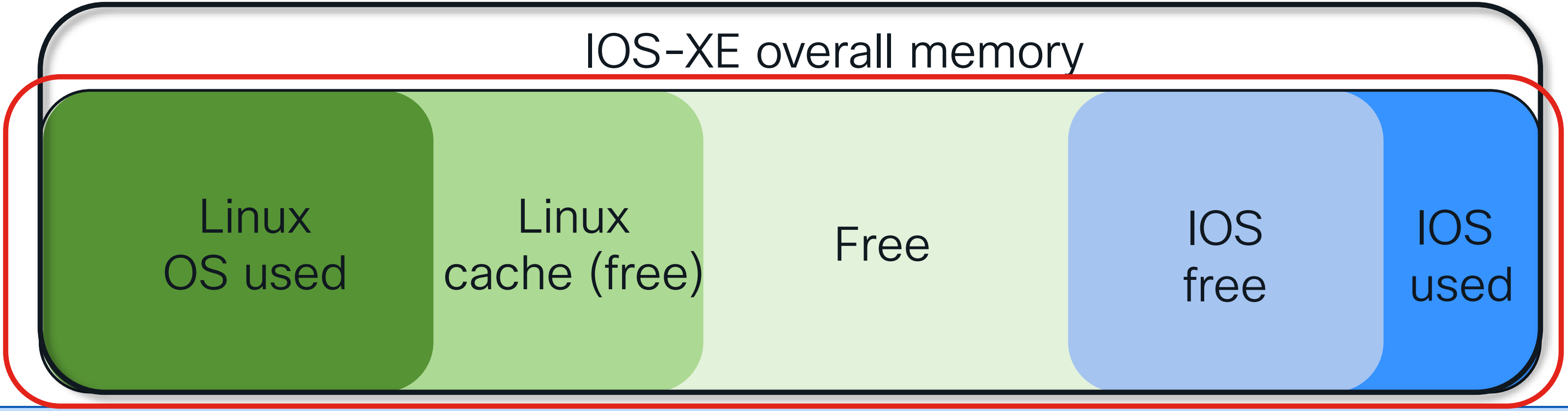
```
%SYS-4-THRESHOLD_TK: Free Memory has dropped below low watermark.
Pool: Processor Free: 52181492 Threshold: 134870705 Tracekey:
1#09f7811786f1de5ddfa0f5542a69f593

%SYS-4-FREEMEMLOW: Top Allocator Name: HTTP CORE, PC:
:55B1DF50A000+B6BE3ED, Size: 346275328, Count: 789749
```

- IOS memory usage outputs stored in:

*bootflash:**threshold_lowmem_info_<timestamp>***

IOS-XE memory usage

Control Plane + Management Plane

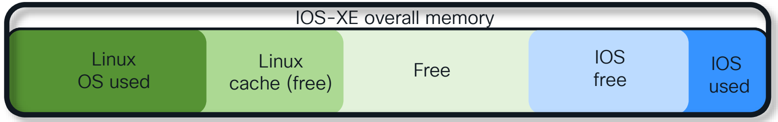


```
# show platform resources

**State Acronym: H - Healthy, W - Warning, C - Critical
Resource                 Usage                Max             Warning         Critical        State
----------------------------------------------------------------------------------------------------
RP0 (ok, active)                                                                              H
 Control Processor       4.13%                100%            80%             90%             H
  DRAM                   4321MB(27%)          15449MB         88%             93%             H
```

IOS-XE (RP) usage

IOS-XE memory usage

Top memory consumer processes

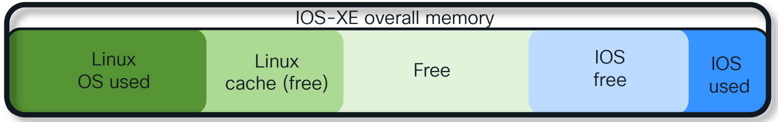


```
#show process memory platform sorted

System memory: 15820156K total, 4424620K used, 11395536K free,
Lowest: 11349604K

   Pid     Text         Data    Stack   Dynamic       RSS              Name
----------------------------------------------------------------------------
  3406   403647      1182620      136       456   1182620   linux_iosd-imag
 23366     3389       262000      136      1372    262000         confd.smp
 18841      280       242584      132      1448    242584        cpp_cp_svr
 19055    11767       209060      136      3216    209060     fman_fp_image
 22437    40710       147708      136       392    147708       mcpcc-lc-ms
```

IOS-XE processes

All counters in kilobytes

IOS-XE memory usage



Memory allocation tracking mechanism based on tags (callsites).

Callsite ID can be decoded by Cisco TAC

```
#show process memory platform accounting

  Hourly Stats

  process                 callsite_ID(bytes)  max_diff_bytes  callsite_ID(calls)  max_diff_calls
-------------------------------------------------------------------------------------------------
  fman_rp_rp_0            2085458948          3607550816      2085458948          1943922
  sessmgrd_rp_0           1824349186          6428784         1823428608          12260
  cli_agent_rp_0          2085458948          1268440         2427634691          3050
  smand_rp_0              2083290122          1205064         3427598337          156
  …
```

For each IOS-XE process the top memory allocator (represented by callsite ID) is displayed based on:

- the number of bytes allocated but not freed
- the number of memory allocations without corresponding memory free request

For Your Reference

IOS-XE memory usage warnings

Syslog alerts when warning/critical IOS-XE memory usage threshold is reached.

```
Mar 13 17:12:47.422 UTC: %PLATFORM-4-ELEMENT_WARNING: R0/0: smand: RP/0: Used Memory value 89% exceeds
warning level 88%. Top memory allocators are: Process: fman_fp_image_fp_0. Tracekey:
1#41501ff8e9f8c5348c0d01317ac6e775 Callsite ID: 1952442373 (diff_call: 1033957). Process:
sessmgrd_rp_0. Tracekey: 1#da8dfbbe9dfb910b99693a33a0353a58 Callsite ID: 1950538752 (diff_call: 12260).
Process: linux_iosd-imag_rp_0. Tracekey: 1#71c88d7e1b9cf87e65b91ce4dcbb60d6 Callsite ID: 1956637699
(diff_call: 4444)
```

Pay attention to timestamps of consecutive alerts (rapid memory spike vs slow increase).

Top 3 memory allocators are displayed, based on the memory accounting analysis.

Memory/CPU usage captured in tracelogs

The Host Manager (HMAN) will periodically (**hourly**) capture the per-process CPU and memory utilization. This information is captured in HMAN tracelogs.

- overall memory usage
- per process RSS/top callsite memory usage
- per process cpu usage

```
#show logging process hman internal start last boot

2020/05/18 07:54:45.205759 {hman_R0-0}{1}: [ov-mem] [24289]: UUID: 0, ra: 0, TID: 0 (note): Memory summary -
Total: 65251644, Used: 4597820, Available: 60653824, Low free: 60542820
<snip>
2020/05/18 07:54:45.208414 {hman_R0-0}{1}: [proc_data] [24289]: UUID: 0, ra: 0, TID: 0 (note): FRU: CC,
Proc: ezman, RSS: 29237, VSS: 539713536, CPU utilization for 5 sec: 1%, 1 min: 1%, 5 min: 1%, Num of open
file descriptors: 20
<snip>
2020/05/18 07:54:45.386709 {hman_R0-0}{1}: [maroon_malloc] [18302]: UUID: 0, ra: 0, TID: 0 (note): Proc:
pubd_rp_0, CS calls: 1748560898, Diff calls: 18446740u, CS bytes: 1612103692, Diff bytes: 353930
```

For Your Reference

Memory Monitoring service & logs

```
/bootflash/tracelogs/memmon_log_20241128_153115_JST_1732775475.tar.gz
/bootflash/tracelogs/memmon_log_20250116_090614_JST_1736985974.tar.gz
/bootflash/tracelogs/memmon_log_20241029_153114_JST_1730183474.tar.gz
/bootflash/tracelogs/memmon_log_20250117_090614_JST_1737072374.tar.gz
```

- Memory usage data is captured periodically into memmon_log files stored in tracelogs folder
- Each tar.gz includes:
  - Memauditlog.txt – stores Linux outputs (IOS-XE system memory)
  - Meminfo.txt - stores IOS memory outputs
  - Ts.txt – stores Linux epoch time
- Implemented on:
  - Catalyst 8200/8200L/8300/8500L – all releases
  - Catalyst 8500 – in IOS-XE 17.8 onwards

For Your Reference

show tech memory

- Contains relevant outputs to be collected in a single shot

```
Router#show tech memory | include -- show
------------------ show clock ------------------
------------------ show version ------------------
------------------ show running-config ------------------
------------------ show platform ------------------
------------------ show platform software status control-processor brief ------------------
------------------ show platform resources ------------------
------------------ show memory statistics history ------------------
------------------ show memory allocating-process total ------------------
------------------ show process memory sorted ------------------
------------------ show process memory platform sorted ------------------
------------------ show memory lite-chunks totals ------------------
------------------ show buffer ------------------
------------------ show buffer usage ------------------
------------------ show region ------------------
------------------ show memory dead totals ------------------
------------------ show chunk brief ------------------
<snip>
------------------ show platform software memory backplaneswitch-manager rp active brief ------------------
------------------ show platform software memory messaging backplaneswitch-manager rp active ------------------
------------------ show processes memory platform accounting ------------------
```

Resource Utilization

- Control Plane CPU
- Control Plane Memory
- **Data Plane CPU**
- Data Plane Memory

Dataplane CPU utilization

Overall processing load

```
C8500#show platform resources

**State Acronym: H - Healthy, W - Warning, C - Critical
Resource                 Usage                 Max             Warning         Critical      State
--------------------------------------------------------------------------------------------------
 ……
 ESP0(ok, active)                                                                              H
  QFP                                                                                          H
   …
   CPU Utilization     46.00%                 100%             90%             95%             H
```

```
C8500#show platform hardware qfp active datapath utilization summary

CPP 0: Subdev 0            5 secs        1 min        5 min       60 min
Input: Total  (pps)        1178722      1231063      1232043      1214378
              (bps)     6293516608   6690041264   6714960600   6634462072
Output: Total (pps)        1169061      1220916      1220224      1203170
              (bps)     6450486808   6853071560   6874761352   6794245080
Processing: Load (pct)          46           36           33           36
```

Total amount of traffic received by QFP

Total amount of traffic leaving QFP

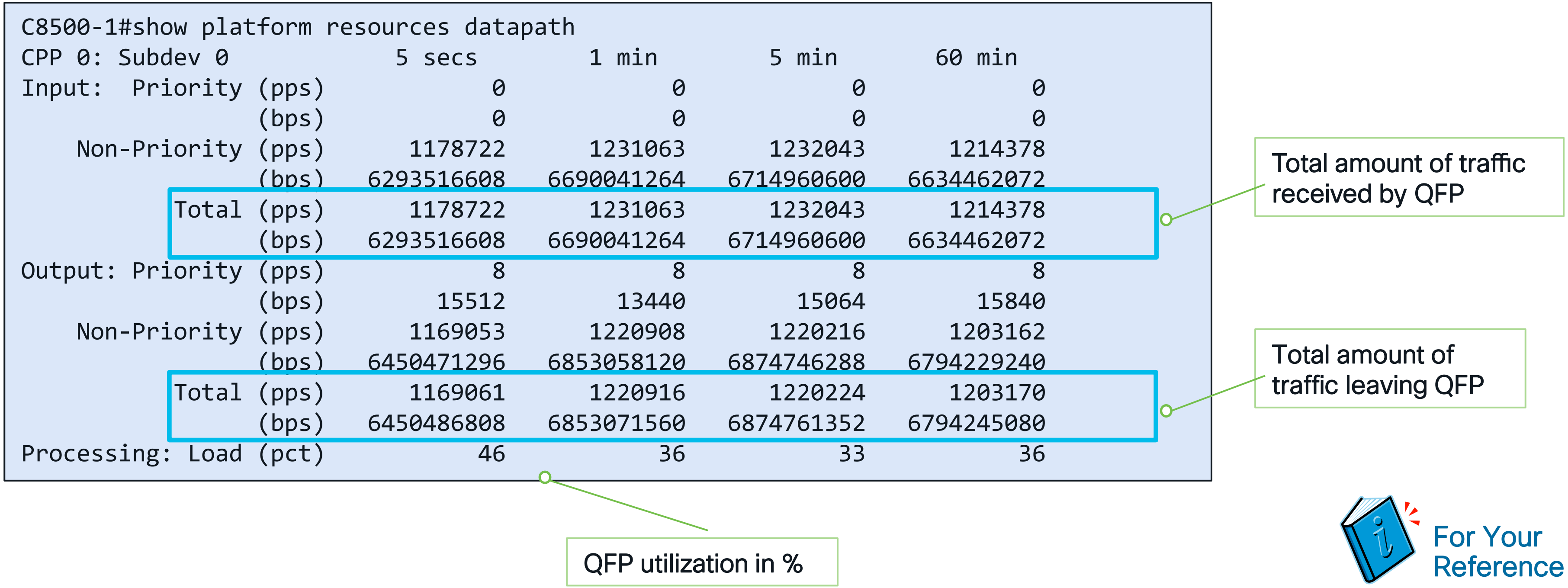
QFP utilization in %

SNMP OID: .1.3.6.1.4.1.9.9.715.1.1.6.1.14

Dataplane CPU utilization

Priority vs non-priority traffic

QFP-based platforms



```
C8500-1#show platform resources datapath
CPP 0: Subdev 0            5 secs        1 min        5 min        60 min
Input:  Priority (pps)            0            0            0            0
                 (bps)            0            0            0            0
    Non-Priority (pps)      1178722      1231063      1232043      1214378
                 (bps)   6293516608   6690041264   6714960600   6634462072
           Total (pps)      1178722      1231063      1232043      1214378
                 (bps)   6293516608   6690041264   6714960600   6634462072
Output: Priority (pps)            8            8            8            8
                 (bps)        15512        13440        15064        15840
    Non-Priority (pps)      1169053      1220908      1220216      1203162
                 (bps)   6450471296   6853058120   6874746288   6794229240
           Total (pps)      1169061      1220916      1220224      1203170
                 (bps)   6450486808   6853071560   6874761352   6794245080
Processing: Load (pct)           46           36           33           36
```

Total amount of traffic received by QFP

Total amount of traffic leaving QFP

QFP utilization in %

For Your Reference

Dataplane CPU utilization

High QFP utilization alerts

```
%IOSXE_QFP-2-LOAD_EXCEED: Slot: 0, QFP:0, Load 88% exceeds the setting threshold 80%.
5 secs traffic rate on QFP: Total Input: 2940667 pps (2940.7 kpps), 9039935768 bps (9039.9 mbps), Total
Output: 2943211 pps (2943.2 kpps), 9365649048 bps (9365.6 mbps).
```

Syslog alerts in newer code versions include the traffic rate information.

- Potential causes of high QFP utilization:
  - Amount of traffic received by the router exceeds the platform limits
  - Low traffic rate but CPU-intensive features configured
  - Sub-optimal router configuration

Next step: Perform **QFP Profiling** with **Packet Trace** (see: Overruns troubleshooting section)

Catalyst 8000 Throughput Considerations

For Your Reference

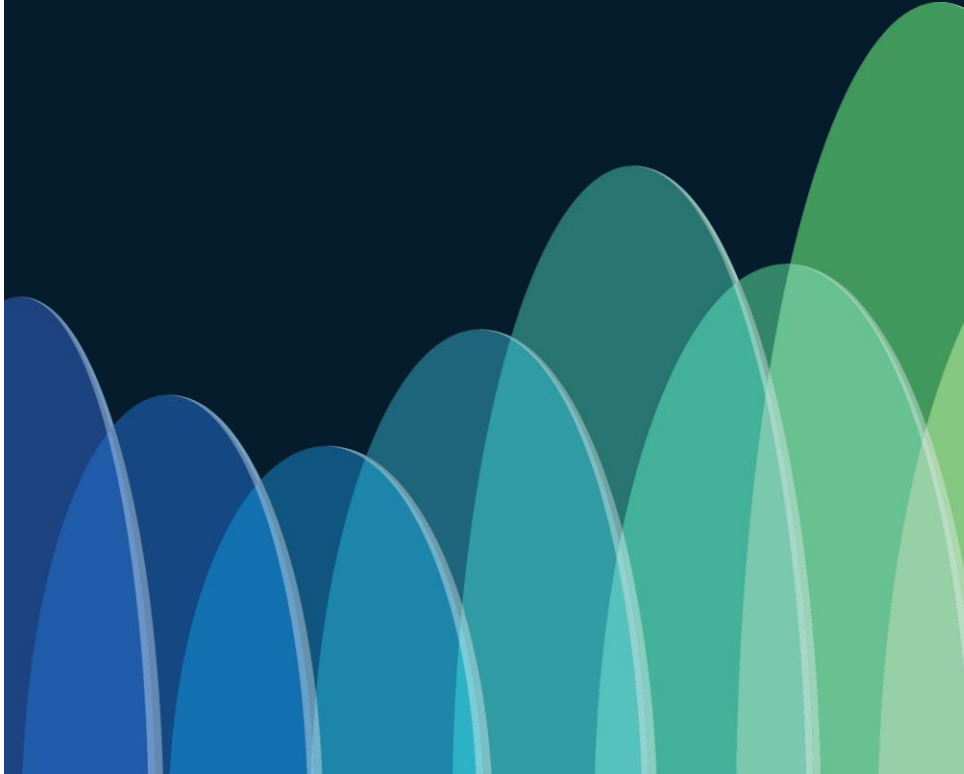
- On physical Catalyst 8000 platforms:
  - Max CEF throughput not restricted (up to platform dataplane limits)
  - Max **crypto** throughput enforced by licensing (DNA Tier + HSEC)
    - Aggregate throughput throttling, no restrictions to input/output ratio
  - The highest DNA Tier unlocks the max platform performance (e.g. Tier 3 on C8500L-8S4X, C8500-12X, C8500-12X4QC)

- On Catalyst 8000v:
  - Max **CEF and crypto throughput (combined)** enforced by licensing

Cisco DNA Subscription Software for SD-WAN and Routing FAQ

Resource Utilization

- ➢ Control Plane CPU
- ➢ Control Plane Memory
- ➢ Data Plane CPU
- ➢ Data Plane Memory

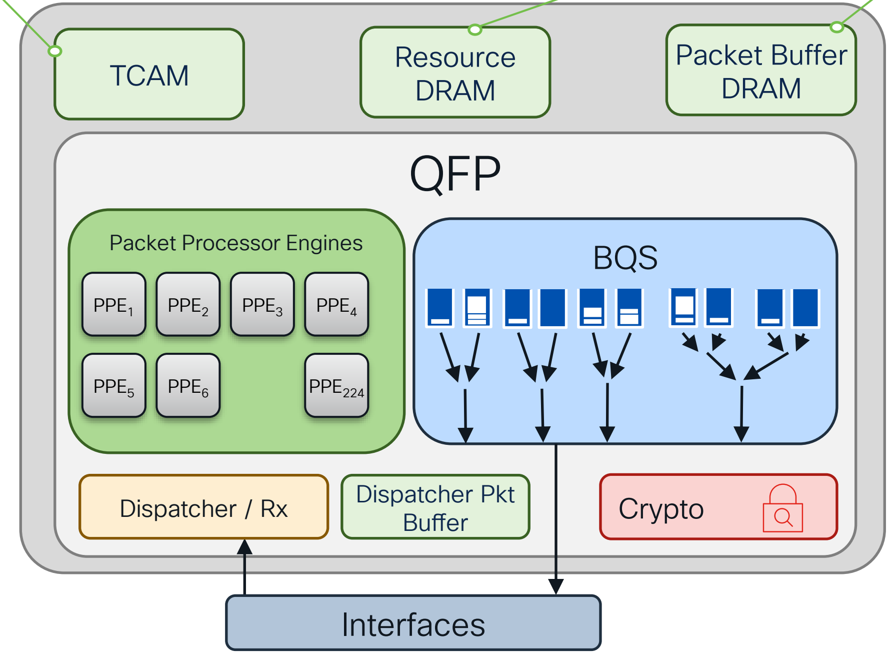


QFP Dataplane Memory

- Class/Policy Maps: QoS, DPI, FW
- ACL/ACE, Route-maps
- IPSec Security Association class groups, classes, rules

- QoS Marking/Policing
- NAT Sessions
- IPsec SA
- Netflow Cache
- FW hash tables

- QoS Queuing
- NAT VFR re-assembly
- IPsec headers



QFP Resources Monitoring

```
C8200#show platform resources
**State Acronym: H - Healthy, W - Warning, C - Critical
Resource                 Usage                 Max             Warning         Critical        State
----------------------------------------------------------------------------------------------------
 <snip>
ESP0(ok, active)                                                                               H
 QFP                                                                                           H
  DRAM                   25225KB(3%)           786432KB        85%             95%             H
  IRAM                   207KB(10%)            2048KB          85%             95%             H
  CPU Utilization        12.00%                100%            90%             95%             H
   …
```

QFP EXMEM

```
C8200#show plat hard qfp active infra exmem statistics
user
<snip>
Type: Name: GLOBAL, QFP: 0
  Allocations  Bytes-Alloc  Bytes-Total  User-Name
  -------------------------------------------------
  8            57236        61440        P/I
  1            65536        65536        EPBR
  1            4384         5120         DPSS
  1            544          1024         CONF_SW
  1            16384        16384        FHS
  …
```

QFP EXMEM monitoring alerts

EXMEM usage exceeds a warning (85%) or critical (95%) threshold:

```
*Aug 10 22:49:56.271: %QFPOOR-4-LOWRSRC_PERCENT_WARN: R0/0: cpp_ha_top_level_server: QFP 0 DRAM (EXMEM) at 86
percent, exceeds warning level 85
*Aug 10 22:49:56.271: %QFPOOR-4-TOP_EXMEM_USER: R0/0: cpp_ha_top_level_server: User: FNF, Allocations: 16,
Bytes-Alloc: 96606508, Bytes-Total: 96617472
*Aug 10 22:49:56.271: %QFPOOR-4-TOP_EXMEM_USER: R0/0: cpp_ha_top_level_server: User: NAT, Allocations: 50,
Bytes-Alloc: 82027184, Bytes-Total: 82048000
```

Not enough QFP EXMEM available to download/update some dataplane structures:

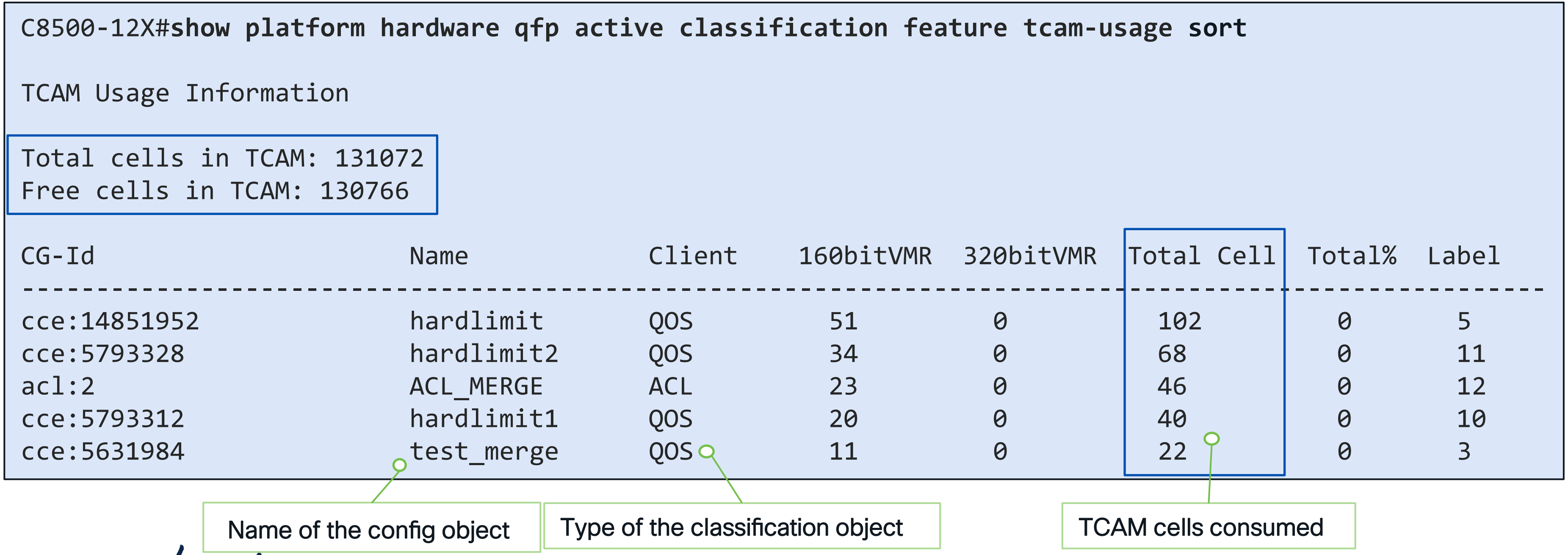
```
*Jul 25 14:57:46.666: %CPPEXMEM-3-NOMEM: R0/0: cpp_cp_svr: QFP: 0, GLOBAL memory allocation of 7130624 bytes
by NAT failed
*Jul 25 14:58:38.787: %CPPEXMEM-3-TOPUSER: R0/0: cpp_cp_svr: QFP: 0, Top User: NAT, Allocations: 52, Type:
GLOBAL
*Jul 25 14:58:38.787: %CPPEXMEM-3-TOPUSER: R0/0: cpp_cp_svr: QFP: 0, Top User: NAT, Bytes Allocated:
96310272, Type: GLOBAL
```

In both scenarios top 2 EXMEM users along with the amount of memory they consume are displayed.

TCAM usage monitoring

QFP-based platforms only

- Display the top 25 class-groups based on the TCAM usage



```
C8500-12X#show platform hardware qfp active classification feature tcam-usage sort

TCAM Usage Information

Total cells in TCAM: 131072
Free cells in TCAM: 130766

CG-Id                      Name            Client     160bitVMR  320bitVMR  Total Cell  Total%  Label
---------------------------------------------------------------------------------------------------------
cce:14851952               hardlimit       QOS          51          0         102         0       5
cce:5793328                hardlimit2      QOS          34          0         68          0       11
acl:2                      ACL_MERGE       ACL          23          0         46          0       12
cce:5793312                hardlimit1      QOS          20          0         40          0       10
cce:5631984                test_merge      QOS          11          0         22          0       3
```

Name of the config object

Type of the classification object

TCAM cells consumed

TCAM limit exceeded alert

QFP-based platforms

When configuration update involves adding/modifying the classification object (e.g. ACL, Class-map, etc.) the structure in TCAM needs to be reprogrammed.

```
%CPP_FM-3-CPP_FM_TCAM_WARNING: R0/0: cpp_sp_svr: TCAM limit exceeded: HW TCAM cannot hold class
group [acl:7] test1. Fail to allocate  160006 TCAM cell entries. Free TCAM cell: 131040 Total TCAM
cell: 131072. Use SW TCAM instead.

%CPP_FM-4-CPP_FM_TCAM_MORE_INFO_WARNING: R0/0: cpp_sp_svr: TCAM limit exceeded:
 Top TCAM users: [acl:2 ACL_MERGE 46] [cce:5631984 test_merge 22] [cce:5551168 test_match_all 2]
```

How to interpret the alert:

- TCAM utilization at the time of error
- Class-group NAME and ID
- Number of TCAM entries that were needed to add the class-group
- Dumps 3 top TCAM using CGs (format: CG-ID, CG-NAME , total VMR entries)

SW TCAM (CACE) limit exceeded alert

SoC platforms

On x86-based platforms there is no physical TCAM present.

For classification objects the QFP EXMEM is utilized by CACE (Common Adaptive Classification Engine), also referred as SW TCAM, with the limit of 64k entries per object.

When new/updated classification object can’t be installed into dataplane due to CACE limit exceeded the syslog alert will be displayed:

```
%CPP_FM-3-CPP_FM_TCAM_WARNING: R0/0: cpp_sp_svr: TCAM limit exceeded: The size of [acl:7] FLR_ND41
config (80003) exceeds the CACE limit (65535 entries).
```

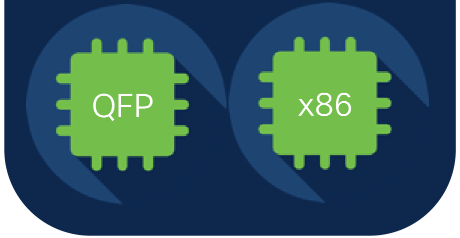
Max number of entries supported in SW TCAM (i.e. 64K) for a single object.

Config object that was getting installed at the time of failure

Conclusions

Key Takeaways

Recognize the importance of platform architecture



Understand traffic distribution model

Utilize the power of Packet Trace

Know where it hurts: Control Plane vs Data Plane

Webex App

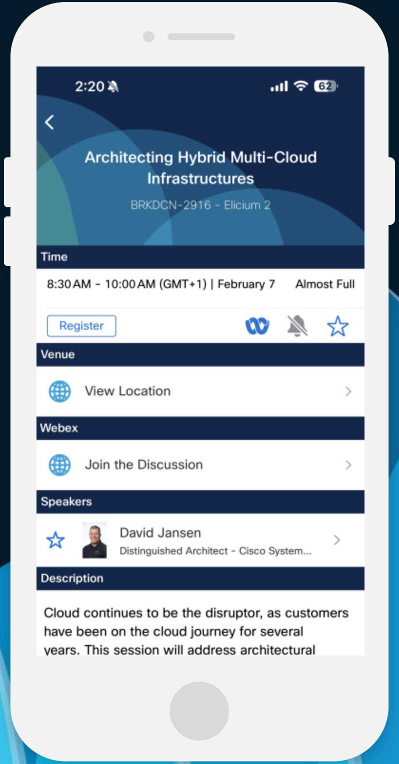
Questions?

Use the Webex app to chat with the speaker after the session

How

1. Find this session in the Cisco Events 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 by the speaker until February 28, 2025.



Architecting Hybrid Multi-Cloud Infrastructures

BRKDCN-2916 - Elicium 2

Time

8:30 AM - 10:00 AM (GMT+1) | February 7 Almost Full

Register

Venue

View Location

Webex

Join the Discussion

Speakers

David Jansen
Distinguished Architect - Cisco System...

Description

Cloud continues to be the disruptor, as customers have been on the cloud journey for several years. This session will address architectural

Fill Out Your Session Surveys

Participants who fill out a minimum of 4 session surveys and the overall event survey will get a unique Cisco Live t-shirt.

(from 11:30 on Thursday, while supplies last)

All surveys can be taken in the Cisco Events mobile app or by logging in to the Session Catalog and clicking the ‘Participant Dashboard’

Content Catalog

Continue your education

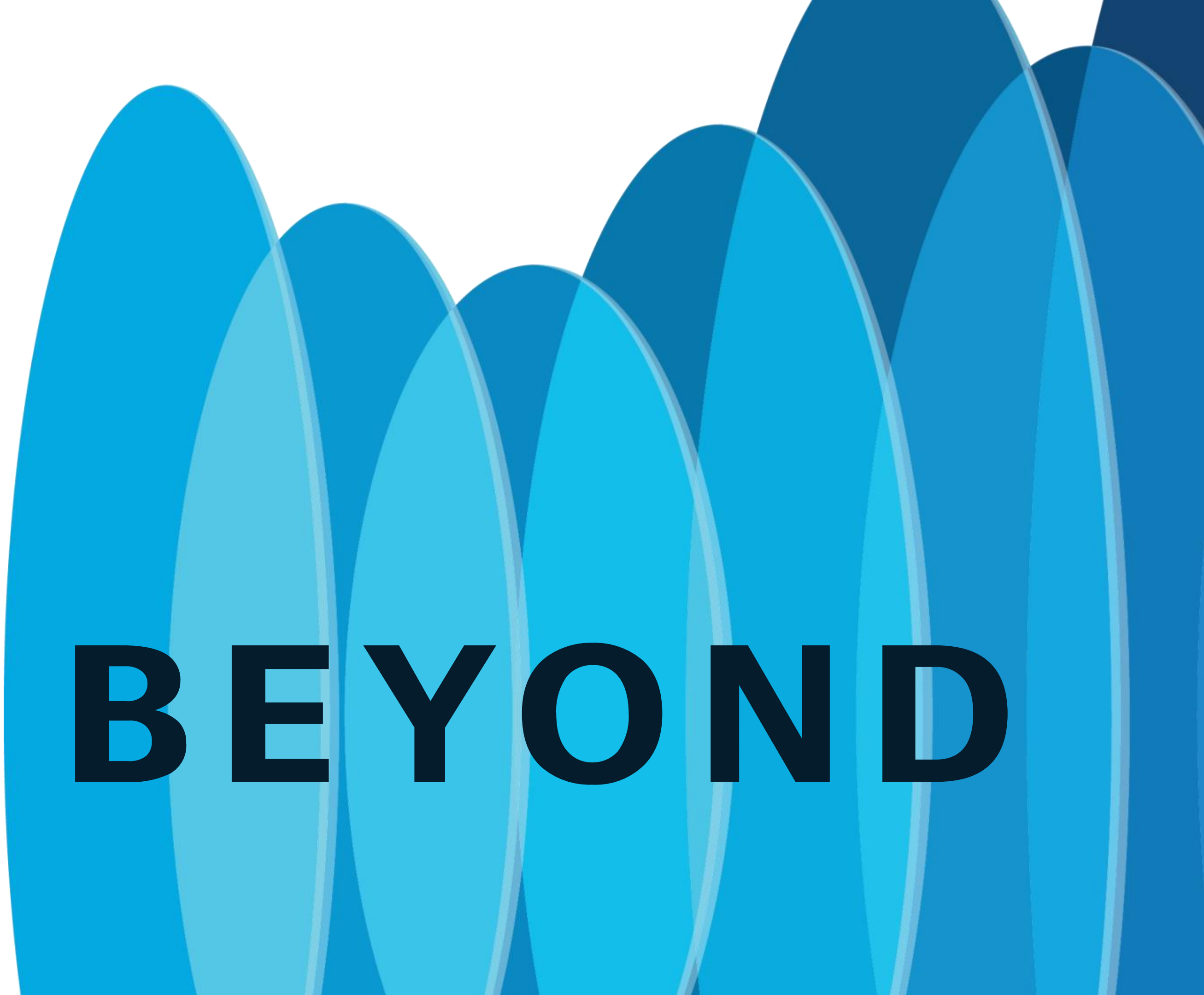
Related Breakout Sessions

- **BRKTRS-3475** [Thursday 3:00 PM]
  Automation and In-Depth Troubleshooting of Cisco Catalyst 8000, ASR1000, ISR4000 and SD-WAN Edge
- **BRKARC-2885** [Thursday 3:45 PM]
  Cisco Catalyst 8500 Series Edge Platform Deep Dive
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library
  for more sessions at ciscolive.com/on-demand.
  Sessions from this event will be available from March 3.

Contact me at: mstanczy@cisco.com

Thank you

cisco Live!



GO **BEYOND**