You make **possible**

IOS-XE Troubleshooting Hands-on Lab

Olivier Pelerin, Technical Leader
Michal Stanczyk, Technical Leader
Wen Zhang, Technical Leader

LTRARC-3500

Barcelona | January 27-31, 2020

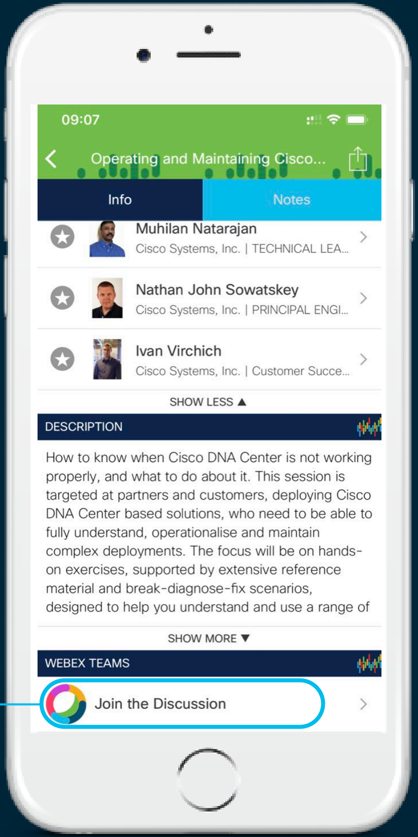
Cisco Webex Teams

Questions?

Use Cisco Webex Teams 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 Webex Teams or go directly to the team space
4. Enter messages/questions in the team space



09:07

Operating and Maintaining Cisco...

Info | Notes

Muhilan Natarajan
Cisco Systems, Inc. | TECHNICAL LEA...

Nathan John Sowatskey
Cisco Systems, Inc. | PRINCIPAL ENGI...

Ivan Virchich
Cisco Systems, Inc. | Customer Succe...

SHOW LESS ▲

DESCRIPTION

How to know when Cisco DNA Center is not working properly, and what to do about it. This session is targeted at partners and customers, deploying Cisco DNA Center based solutions, who need to be able to fully understand, operationalise and maintain complex deployments. The focus will be on hands-on exercises, supported by extensive reference material and break-diagnose-fix scenarios, designed to help you understand and use a range of

SHOW MORE ▼

WEBEX TEAMS

Join the Discussion

Agenda

- Introduction to IOS-XE Platform Software/Hardware Architecture
  - Resource Consumption Monitoring
- Day in the Life of a Packet
  - Data Plane Packet Tracing
- Troubleshooting strategy and Tools
  - Embedded Packet Capture
  - Understanding and Extracting Platform Logs
- Hands-on Lab exercise
- Wrapping up...

Session Objectives

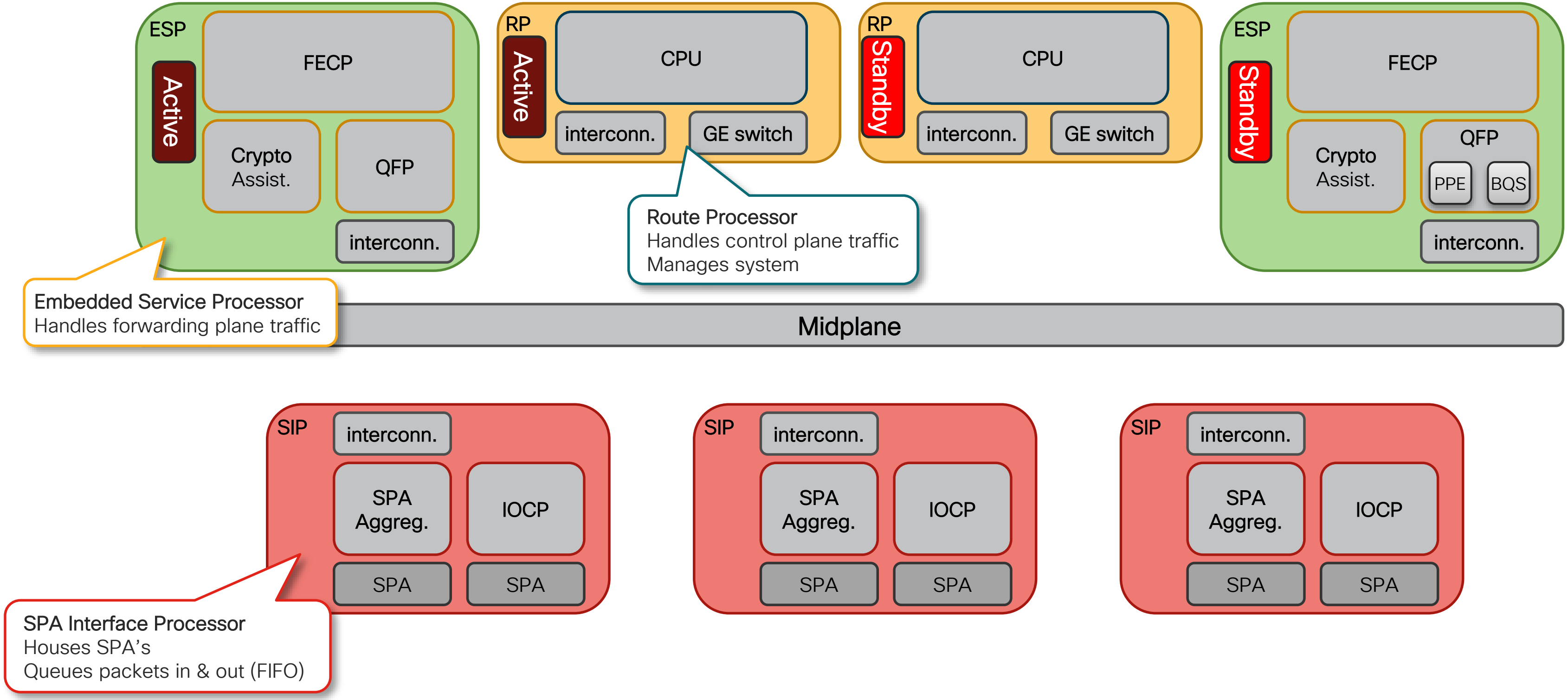
- To understand IOS-XE (ASR1k, ISR4k, CSR1Kv) Platform Architecture
  - Software
  - Hardware
  - Feature implementations
- Understand how features process packets through IOS-XE
- To demonstrate a systematic Troubleshooting Strategy
- To showcase various troubleshooting Tools and Capabilities
- To provide a Hands-on experience on how to effectively troubleshoot the platform using these tools

Related Sessions

- BRKCRS-3147
  Advanced troubleshooting of the ASR1K and ISR (IOS-XE) made easy
  - Olivier Pelerin – Technical Leader, Services
  - Frederic Detienne – Distinguished Engineer, Services
- LABRST-2400
  Packet Capturing Tools in Routing Environments WISP Lab

ASR Series Hardware Architecture

ASR1000 Building Blocks

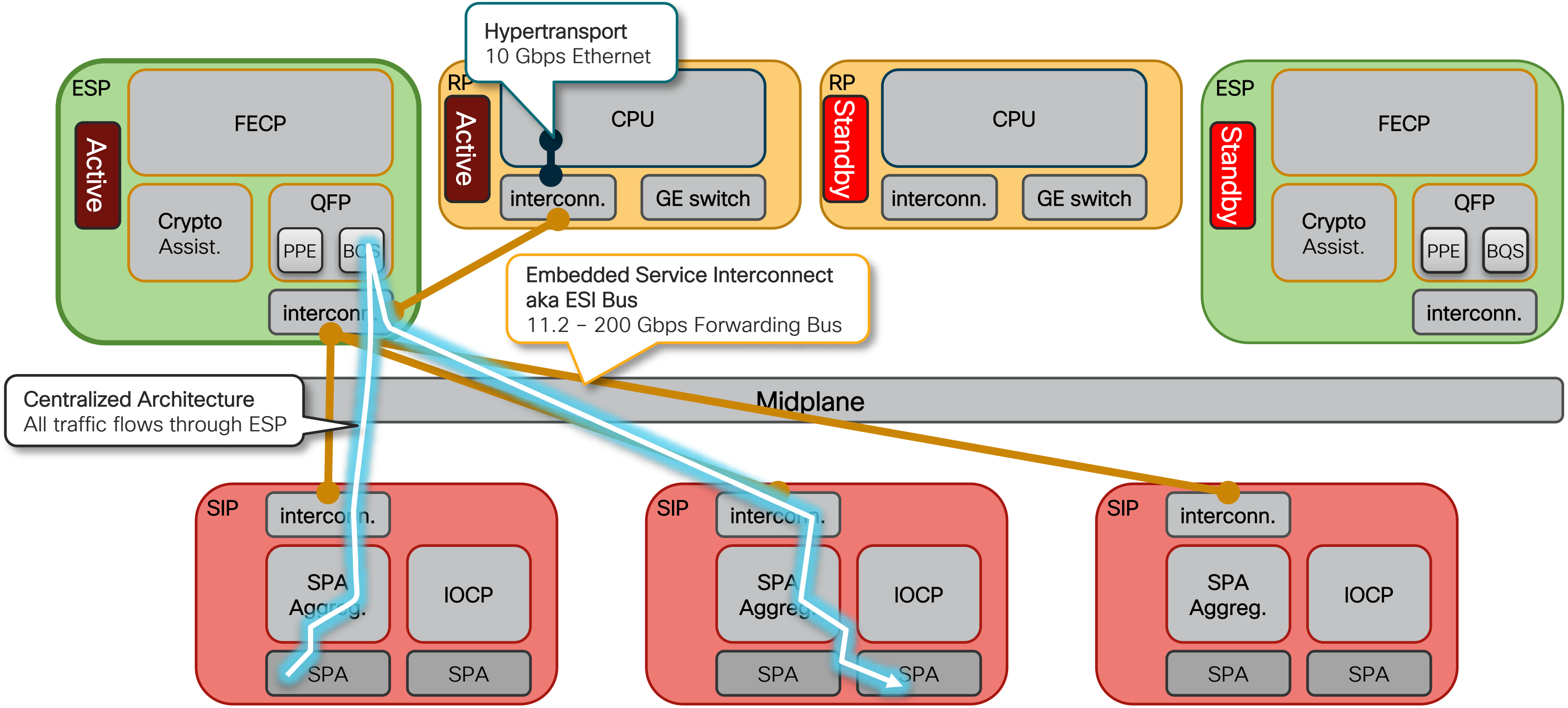


Route Processor
Handles control plane traffic
Manages system

Embedded Service Processor
Handles forwarding plane traffic

SPA Interface Processor
Houses SPA's
Queues packets in & out (FIFO)

System Architecture Forwarding Plane

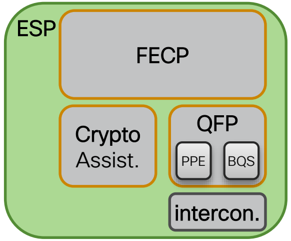


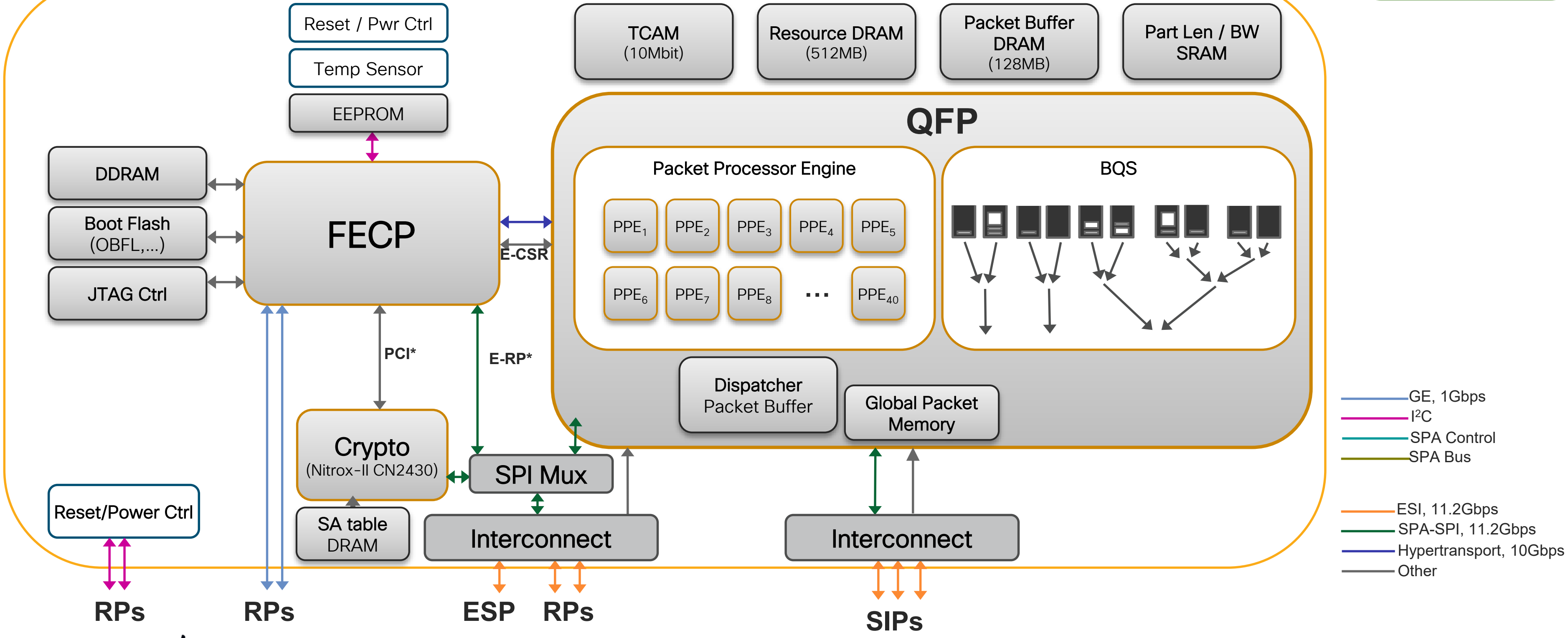
Hypertransport
10 Gbps Ethernet

Embedded Service Interconnect
aka ESI Bus
11.2 – 200 Gbps Forwarding Bus

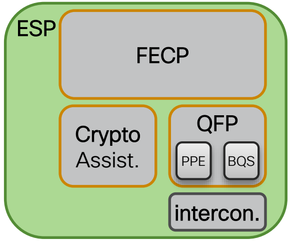
Centralized Architecture
All traffic flows through ESP

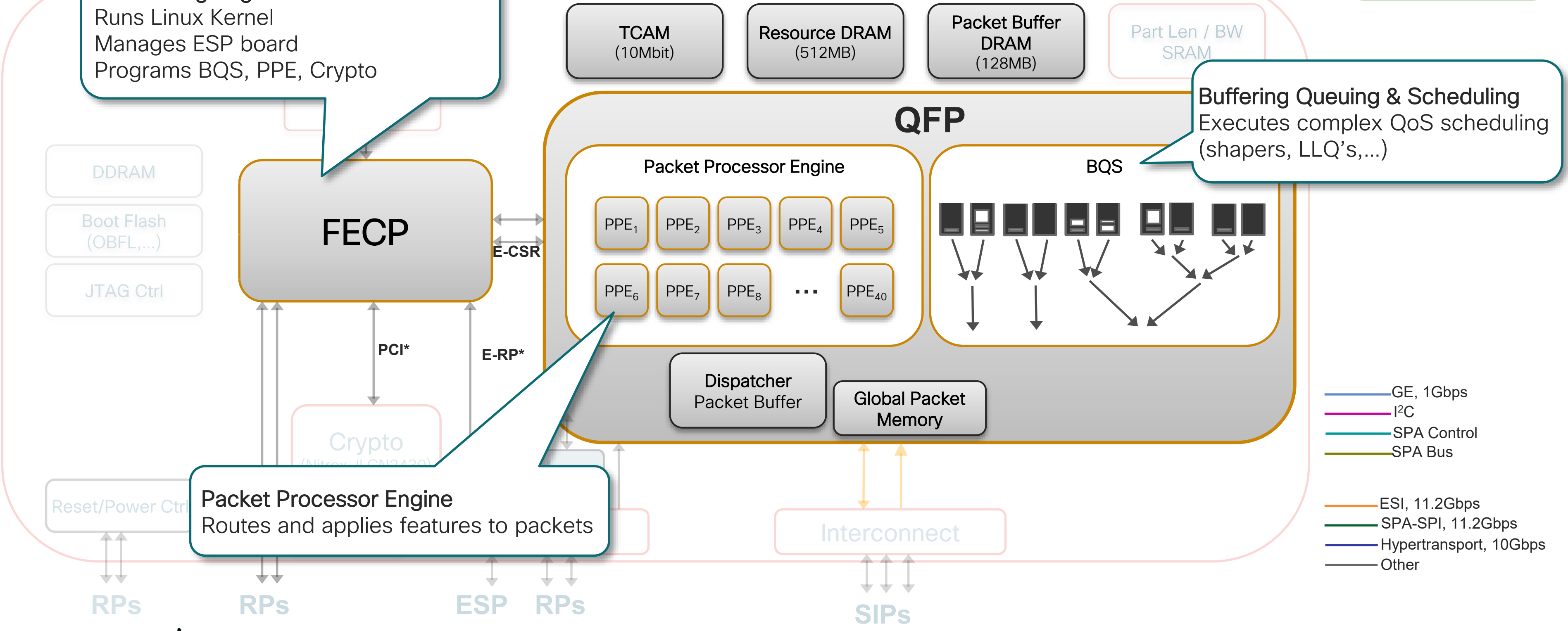
ESP-xx Block Diagram





ESP-xx Block Diagram





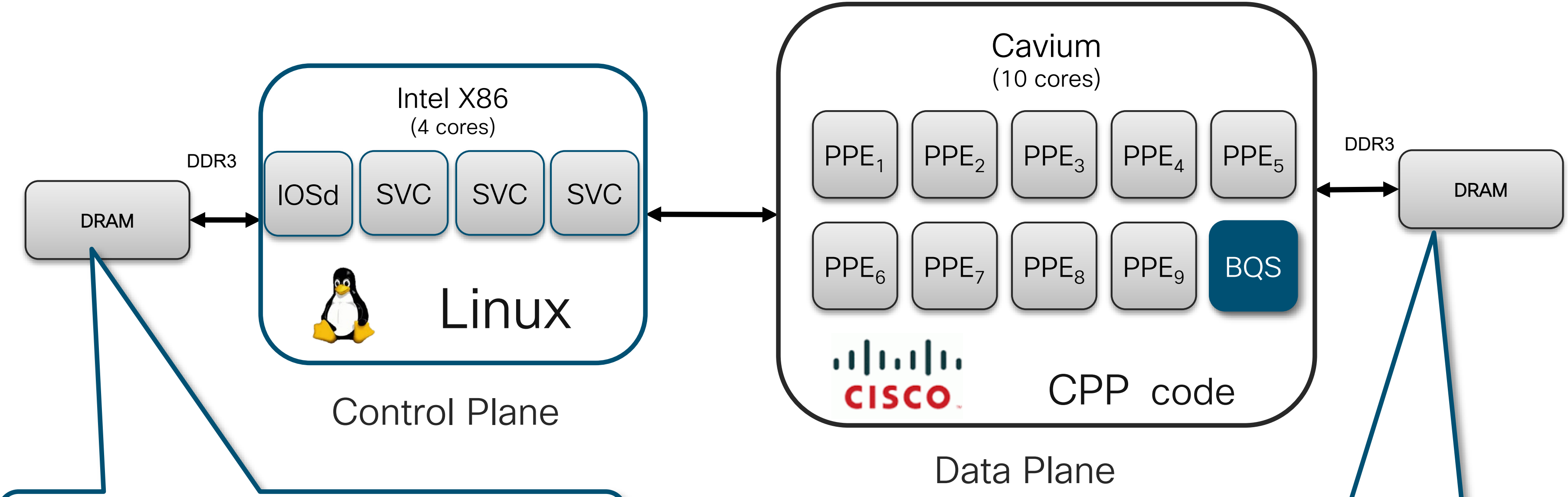
Forwarding Engine Control Processor
Runs Linux Kernel
Manages ESP board
Programs BQS, PPE, Crypto

Buffering Queuing & Scheduling
Executes complex QoS scheduling (shapers, LLQ’s,...)

Packet Processor Engine
Routes and applies features to packets

ISR4000 Series Hardware Architecture

ISR 4451-X (ISR4451)



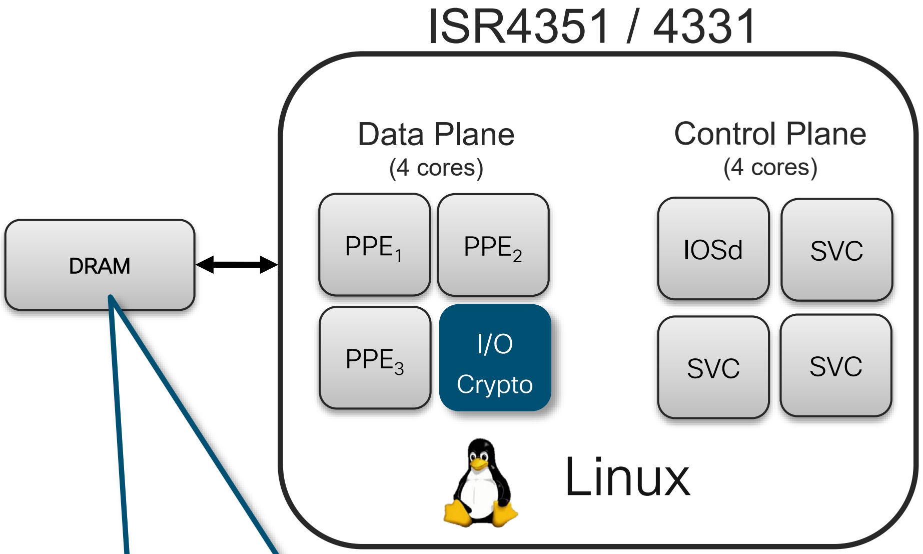
4GB by default, can be upgraded

- Used for IOS daemon
- Used for Linux (including service containers)

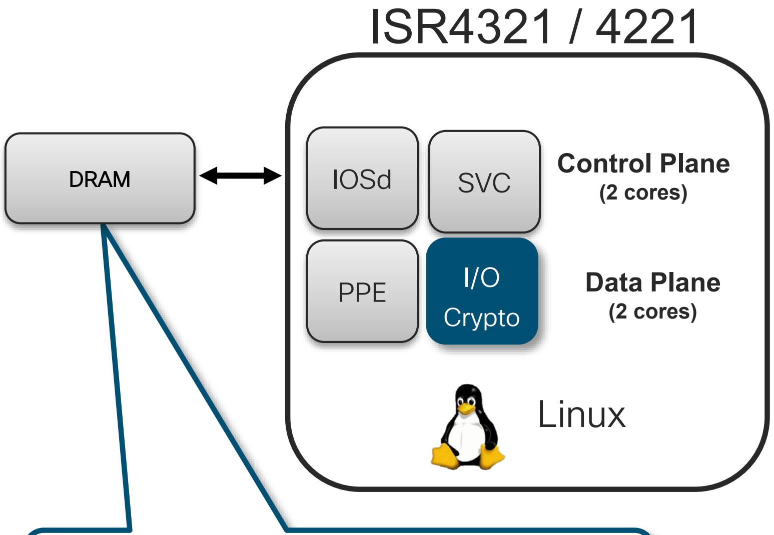
Fixed 2GB for dataplane only:

- Packet Buffering (750 MB)
- QFP internal microcode (750 MB)
- EXMEM (512 MB)

ISR 4300 & 4200

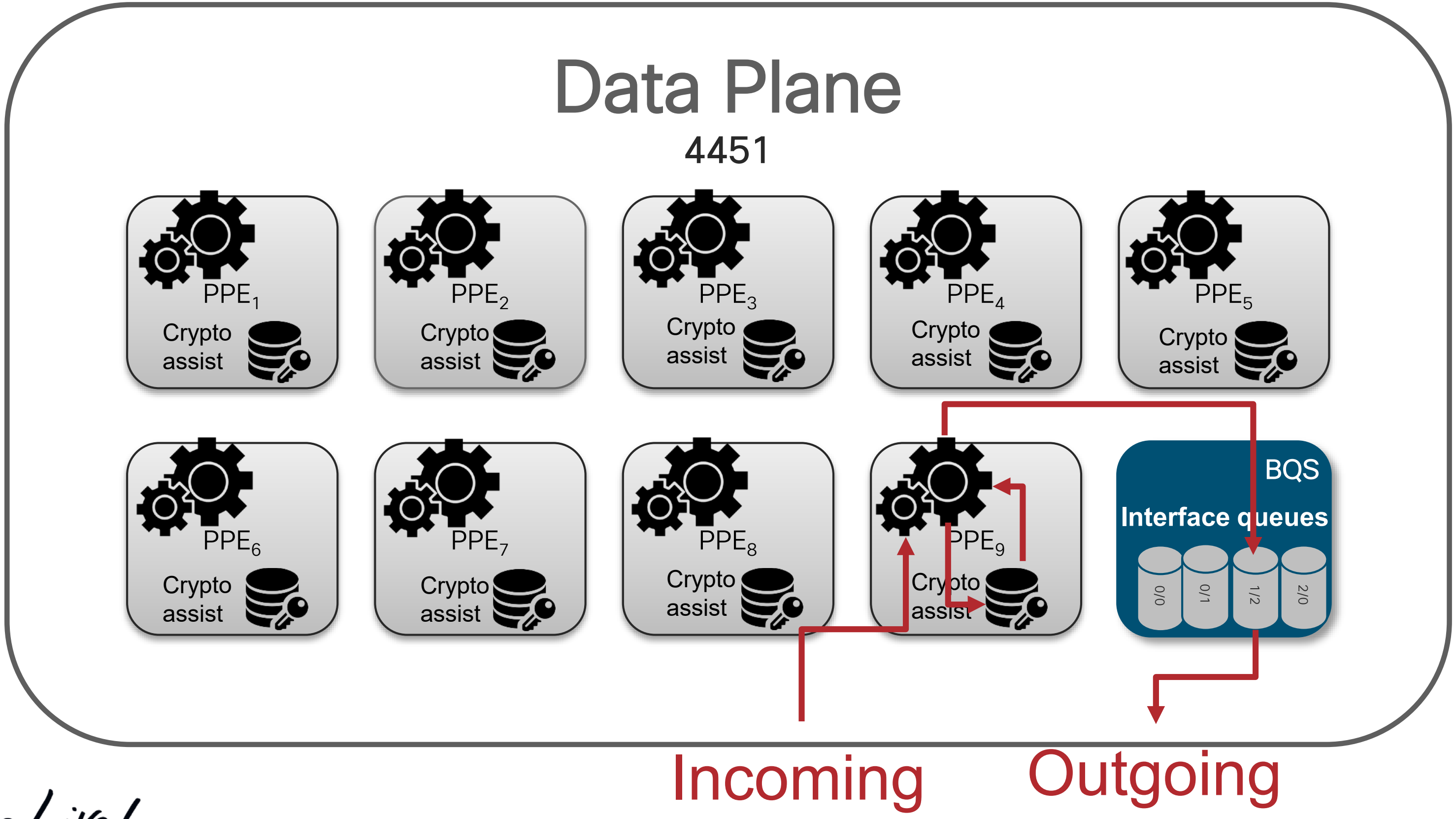


Shared between dataplane and control plane



Shared between dataplane and control plane

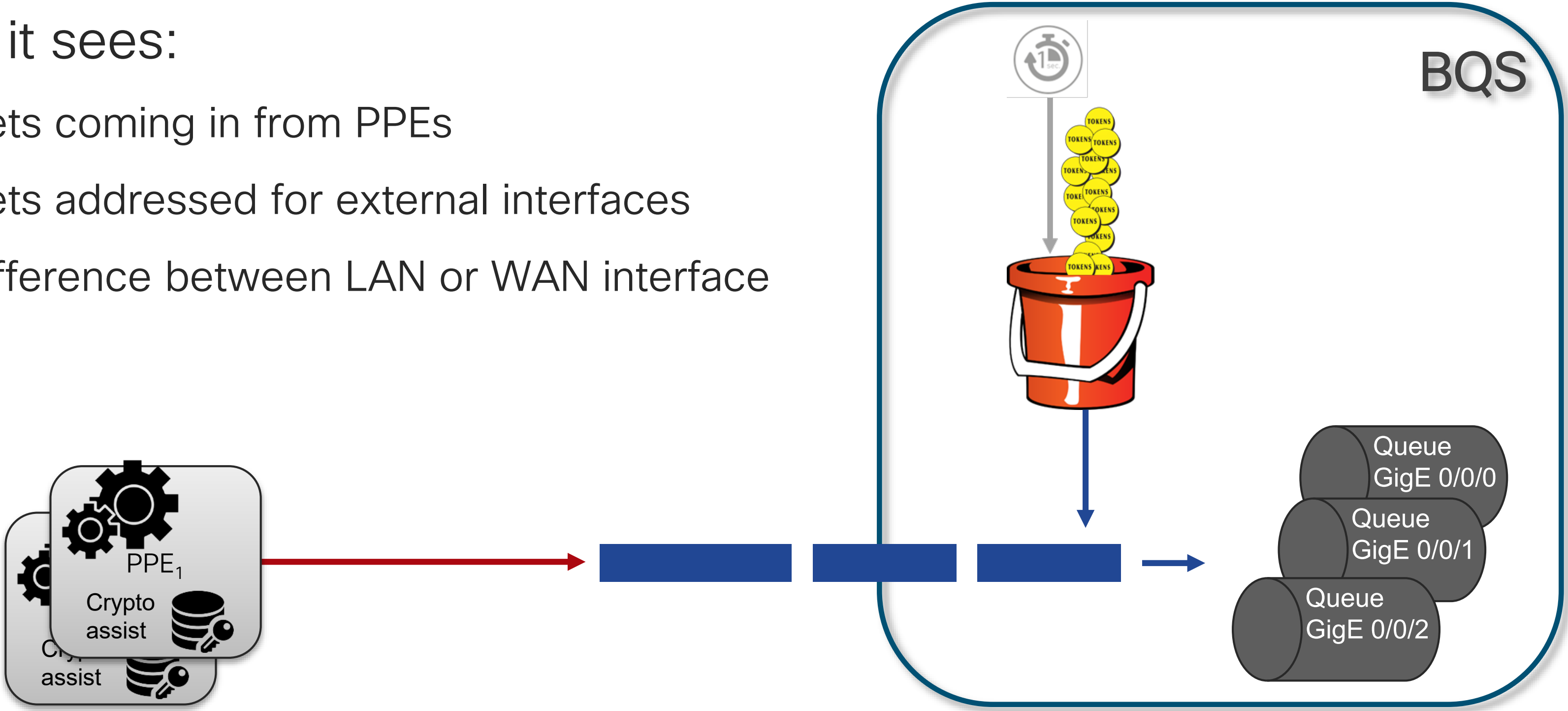
BQS - Where the Performance Shaper lives



Performance License bit counter view

What it sees:

- Packets coming in from PPEs
- Packets addressed for external interfaces
- No difference between LAN or WAN interface



Looking for indications of exceeding license

- Oversubscribed ISR4k lab router

```
#show plat hard qfp active datapath utilization
  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)        18027        17536        17493        17740
                 (bps)    101806904       184352       195272       204816
           Total (pps)        18207        17536        17493        17740
                 (bps)    101806904       184352       195272       204816
Output: Priority (pps)            0            0            0            0
                 (bps)            0            0            0            0
    Non-Priority (pps)        17916        17400        17361        17578
                 (bps)     99956512       198024       209024       218568
           Total (pps)        17916        17400        17361        17578
                 (bps)     99956512     97592394     98694332     94902000
Processing: Load (pct)            7            7            7            7
```

100Mbps on egress at the QFP level

```
#show plat hard qfp active statistics drop
-------------------------------------------------------------------------
Global Drop Stats                         Packets                  Octets
-------------------------------------------------------------------------
TailDrop                                     4395                 6634970
```

Looking for indications of exceeding license

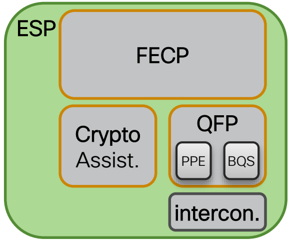
Oversubscribed ISR4k lab router - showing oversubscribed interfaces

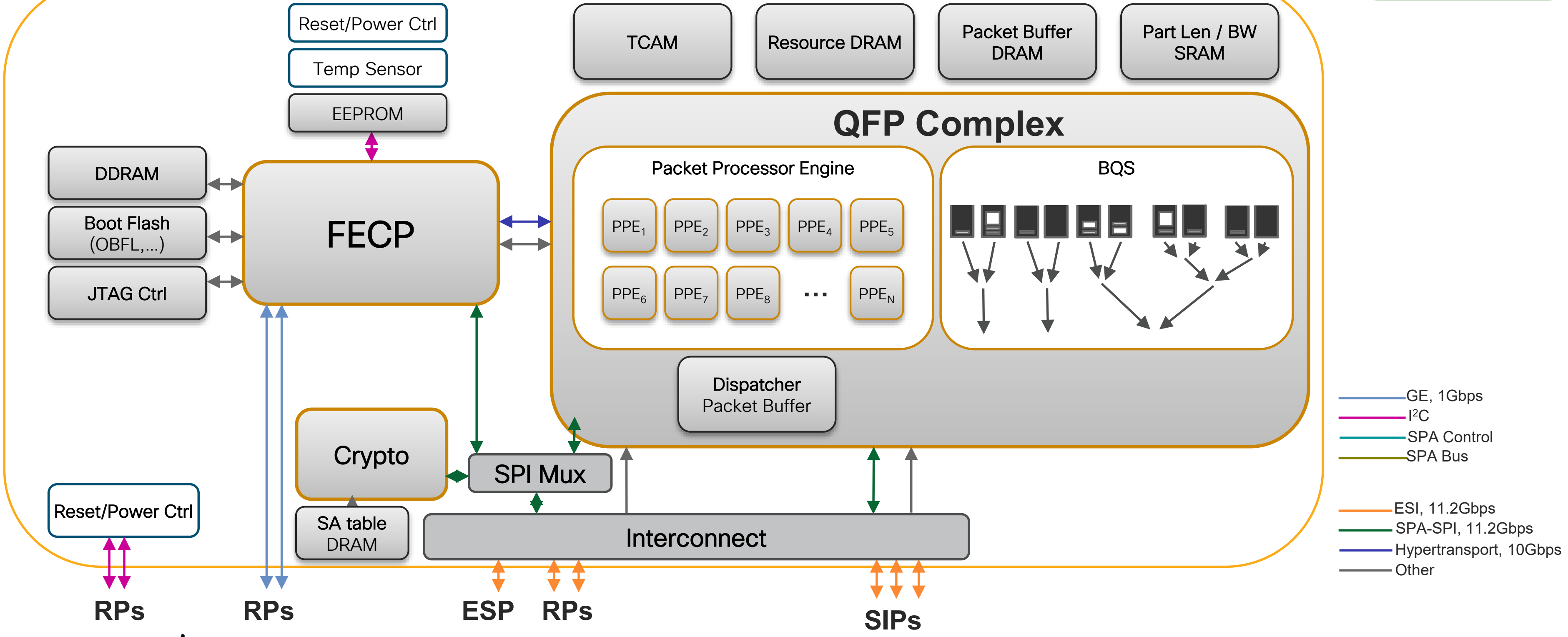
```
#show plat hard qfp active feature lic-bw oversubscription
Interface: GigabitEthernet0/0/0, QFP interface: 7

  Overall Traffic:
      enqueued   (bytes):          7188433, (packets):        75926
      tail_drops (bytes):                0, (packets):            0
      total      (bytes):          7188433, (packets):        75926

Interface: GigabitEthernet0/0/1, QFP interface: 8

  Overall Traffic:
      enqueued   (bytes):      10492353355, (packets):    236972715
      tail_drops (bytes):         18809589, (packets):        56020
      total      (bytes):      10511162944, (packets):    237028735

Interface: GigabitEthernet0/0/2, QFP interface: 9

  Overall Traffic:
      enqueued   (bytes):          9544293, (packets):        57041
      tail_drops (bytes):                0, (packets):            0
      total      (bytes):          9544293, (packets):        57041
```

Look for signs of evenly distributed buffering on interfaces

Look for drops on busy interfaces

Generic ESP Block Diagram





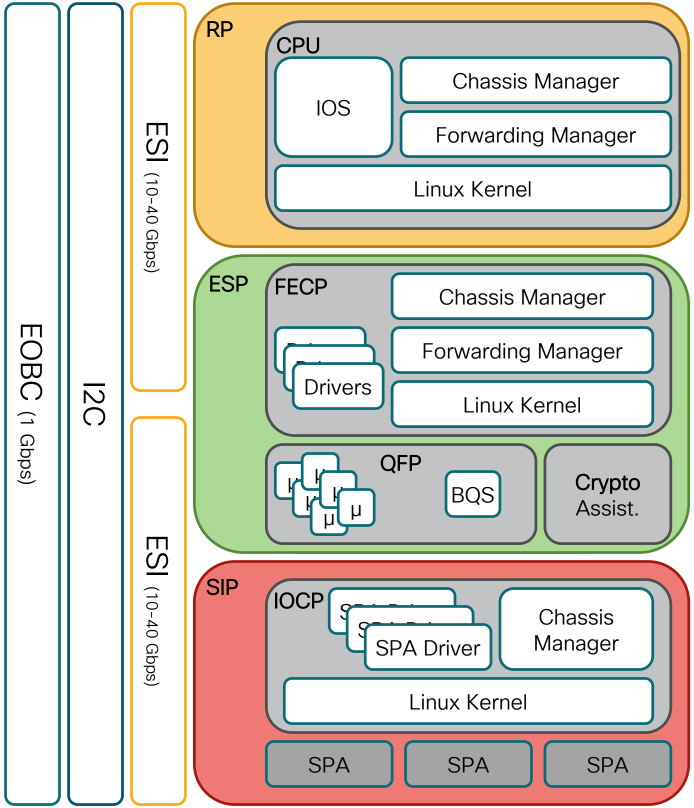
Acronyms

For Your Reference

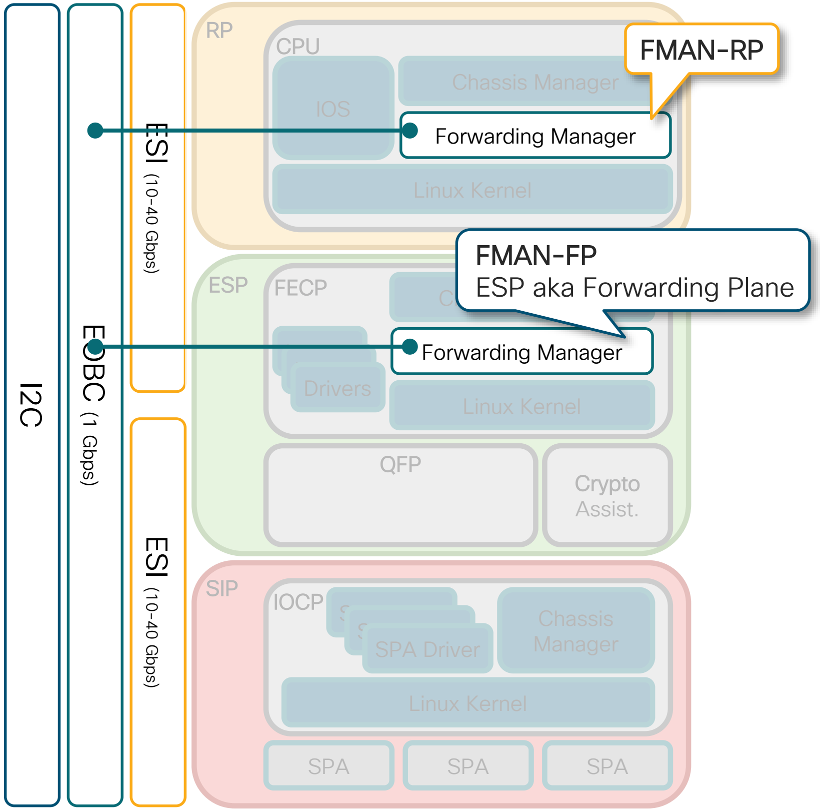
- RP – Route Processor
- FP – Forwarding Processor = ESP (Embedded Service Processor)
- CPP – Cisco Packet Processor Complex= QFP (Quantum Flow Processor)
- PPE – Packet Processing Engine
- IOCP – I/O Control Processor
- FECP – Forwarding Engine Control Processor
- SPA – Shared Port Adapter
- SIP – SPA Interface Processor
- IOSd – IOS image that runs as a process on the RP
- FMAN – Forwarding manager (FMAN-RP, FMAN-FP)
- EOBC = Ethernet Out of Band Channels – Packet Interface for Card to Card Control Traffic
- IOS-XE (BinOS) = Linux Based Software Infrastructure for IOS-XE

ASR1000 Software Architecture

ASR1K Software Architecture

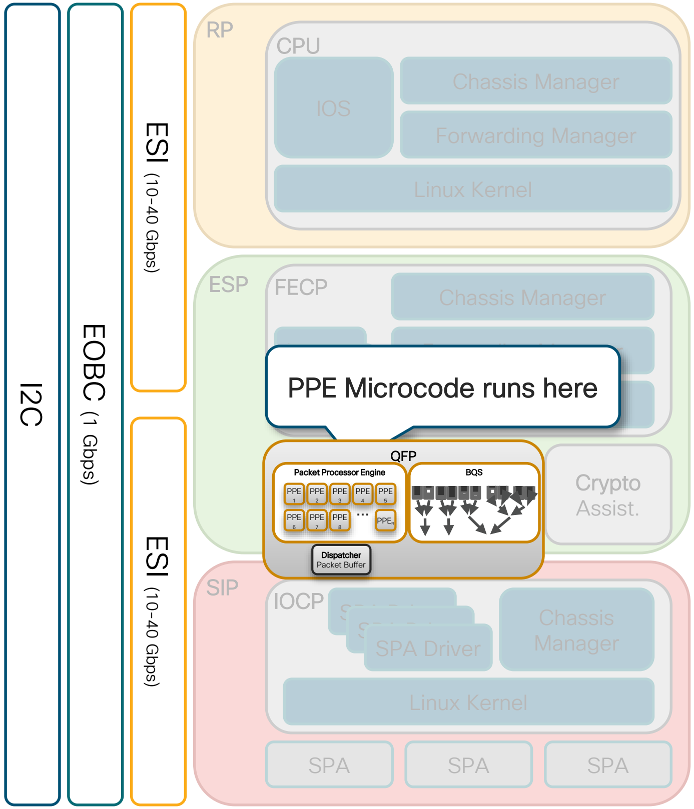


Forwarding Manager (FMAN)



- FMAN on RP communicates with FMAN process on ESP
  - Distributed function
- Propagates control plane ops. to ESP
  - CEF tables, ACL's, NAT, SA's,...
- FMAN-FP communicates information back to FMAN-RP
  - e.g. statistics
  - FMAN-RP pushes info back to IOS
- FMAN on active RP maintains state for both active & standby ESP's
  - Facilitates NSF after re-start with bulk download of state information

PPE Microcode

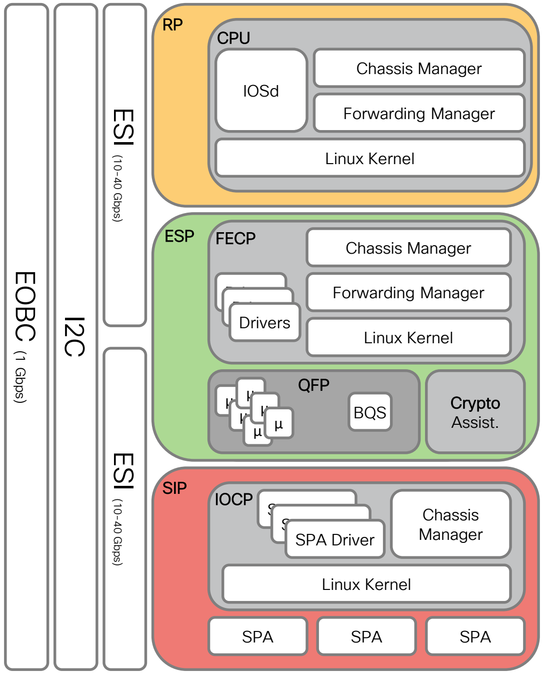


PPE Microcode runs here

- Written in C
  - Proper features, no hack
- Runs on each thread of the PPE
- Processes packets
  - Run to completion
  - Assisted by various memories
  - TCAM, DRAM,... various speeds
- Features applied via FIA
  - Feature Invocation Array
- FIA per interface
  - Input FIA, output FIA
  - Drop FIA (Null interface)

ASR1000 vs ISR4000

ASR1000



EOBC (1 Gbps)

I2C

ESI (10-40 Gbps)

ESI (10-40 Gbps)

RP

CPU

IOSd

Chassis Manager

Forwarding Manager

Linux Kernel

ESP

FECP

Chassis Manager

Forwarding Manager

Drivers

Linux Kernel

QFP

µ

BQS

Crypto Assist.

SIP

IOCP

SPA Driver

Chassis Manager

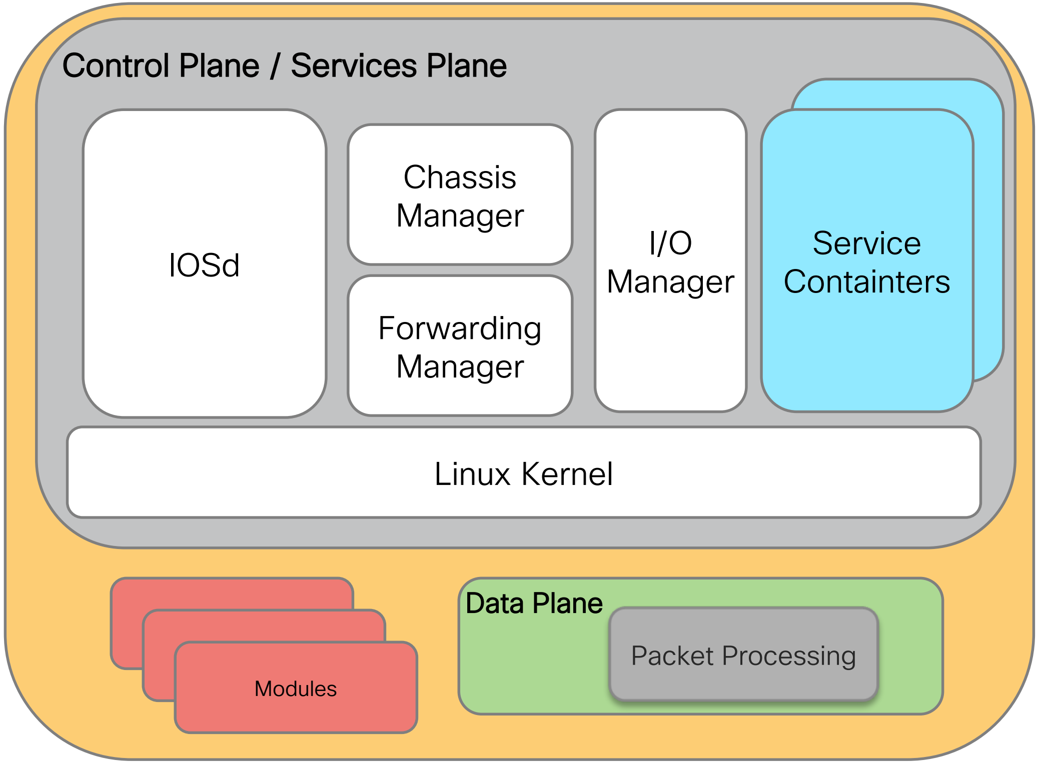
Linux Kernel

SPA

SPA

SPA

ISR4000



Control Plane / Services Plane

IOSd

Chassis Manager

Forwarding Manager

I/O Manager

Service Containters

Linux Kernel

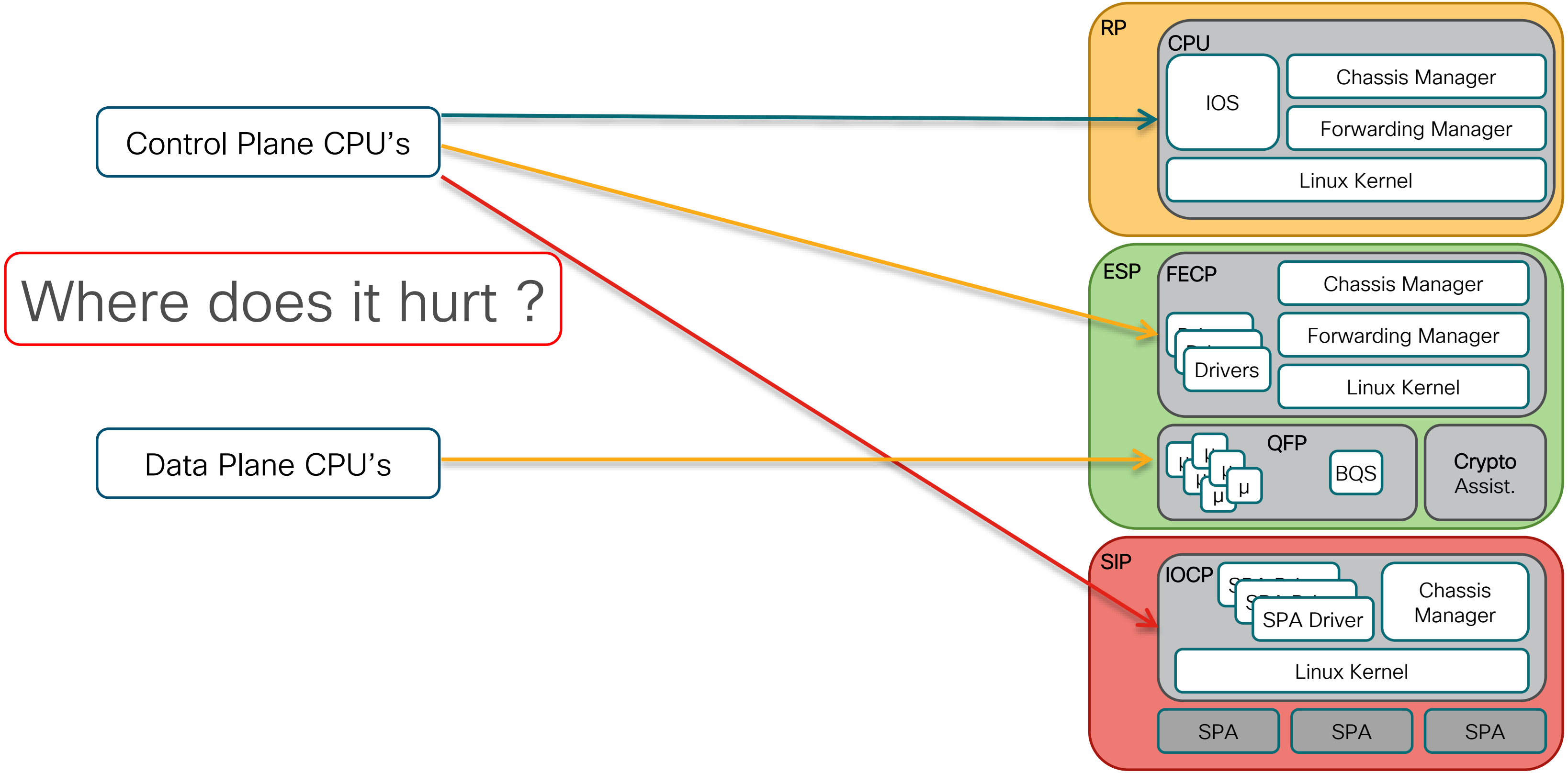
Modules

Data Plane

Packet Processing

Resource Monitoring

The Vital Signs...



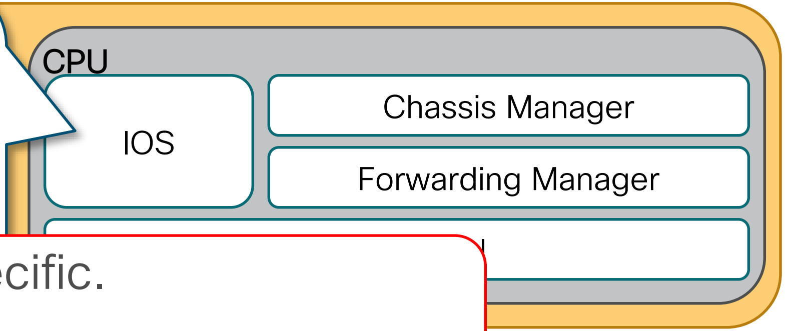
Control Plane CPU's

Where does it hurt ?

Data Plane CPU's

Example: IOS Memory vs RP Memory Utilization

```
asr-1k#show memory statistic
Load for five secs: 6%/1%; one minute: 5%; five minutes: 3%
Time source is NTP, 22:18:08.111 EDT Sat Apr 19 2014

                Head     Total(b)      Used(b)      Free(b)    Lowest(b)  Largest(b)
Processor   300AE008   1713127140    564269356   1148857784   1066242316   992444168
 lsmpi_io   963791D0      6295088      6294120          968          968         968

asr-1k#show process mem | inc BGP
523    0

asr-1k#sh
…
```



Complex CLI, platform specific.

Additional information require connecting to the Linux shell

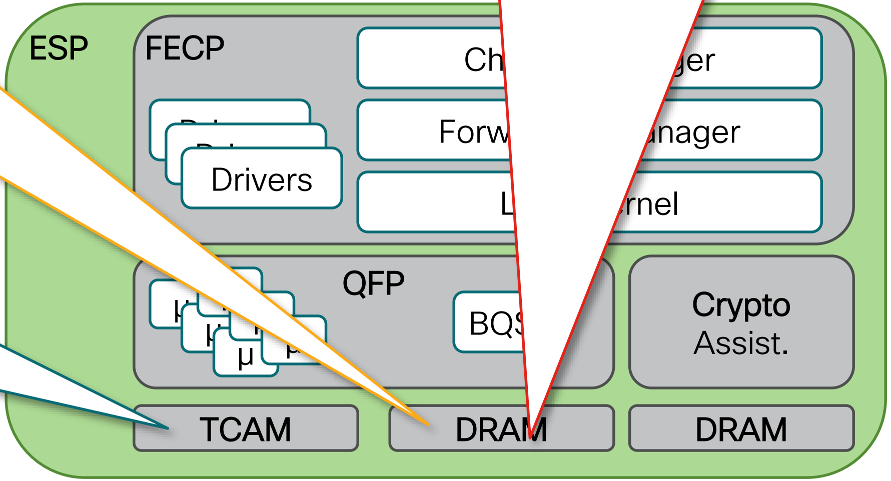
```
asr-1k#show platform software process list RP active summary
…
 Architecture     : ppc
  Memory (kB)
    Physical       : 4127744
    Total          : 3874516
    Used           : 2095636
    Free           : 1778880

asr-1k#show platform software process list RP active | inc fman
fman_rp                  29015   27992     29015   S          20   136847360
```

QFP Memory Utilization
It gets worse…

```
asr-1k#show platform hardware qfp active infrastructure exmem statistics
QFP exmem statistics

Type: Name: DRAM, QFP: 0
  Total: 1073741824
  InUse: 219466752
  Free: 854275072
  Lowest free water mark: 854005760
Type: Name: IRAM, QFP: 0
  Total: 134217728
  InUse: 8728576
  Free: 125489152
  Lowest free water mark: 125489152
Type: Name: SRAM, QFP: 0
  Total: 32768
  InUse: 15088
  Free: 17680
  Lowest free water mark: 17680
```

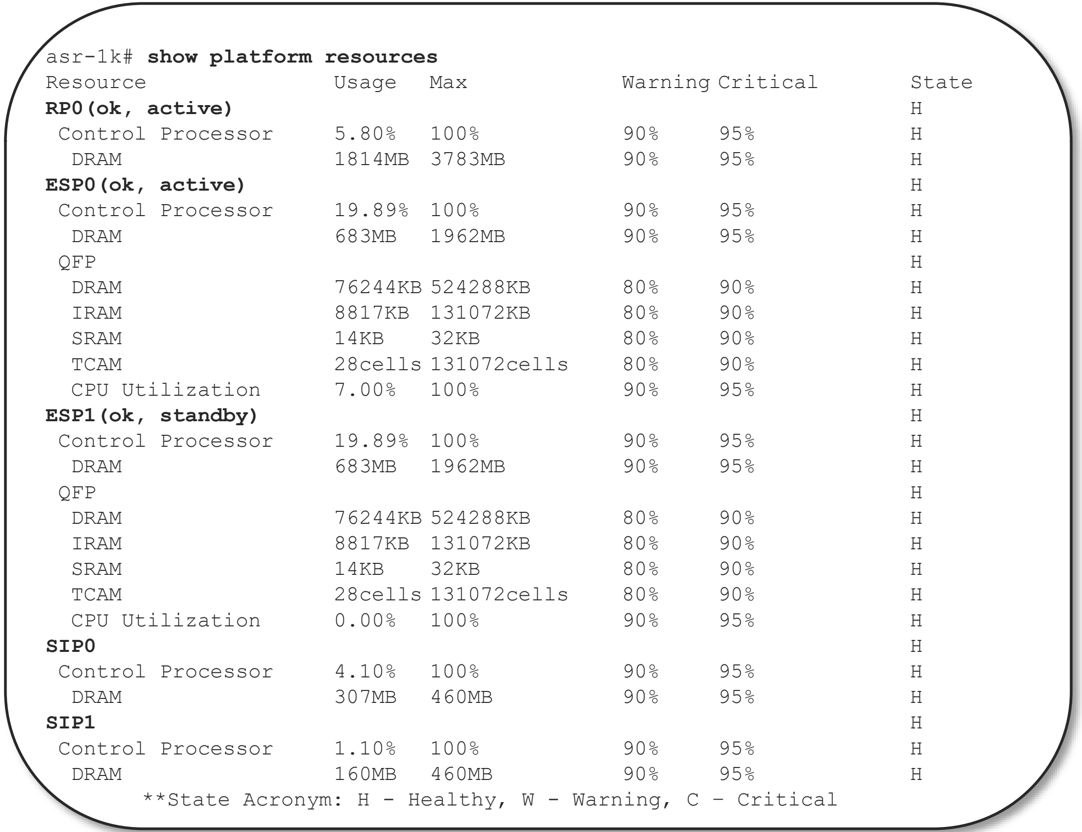
```
asr-1k#show platform hardware qfp active infrastructure exmem statistics user
…
10            279092        284672        CEF
40            36441494      36458496      NAT
```

```
asr-1k#show platform hardware qfp active tcam resource-manager usage
Load for five secs: 0%/0%; one minute: 1%; five minutes: 1%
Time source is NTP, 09:43:55.075 EDT Fri Apr 25 2014

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        : 28
Total tcam free cell entries        : 524260
Threshold status                    : below critical limit
```

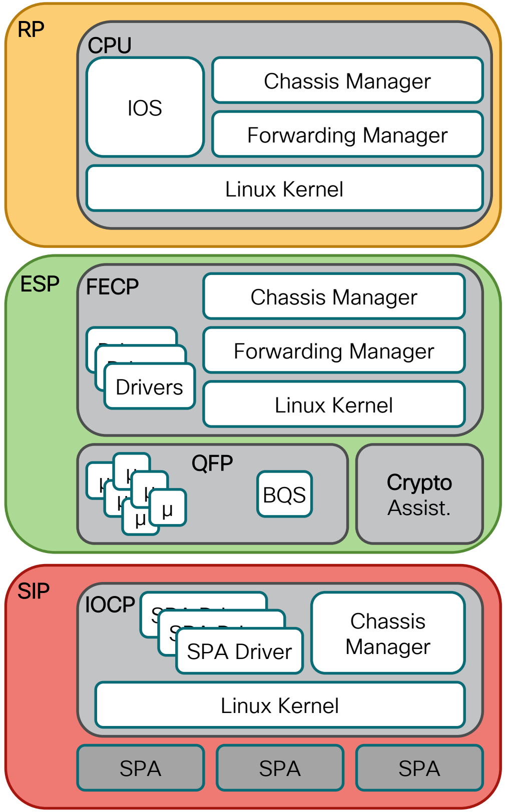


Resources - A Simplified View

Introduced in IOS-XE 3.14



```
asr-1k# show platform resources
Resource               Usage   Max             Warning Critical       State
RP0(ok, active)                                                       H
 Control Processor     5.80%   100%            90%     95%            H
  DRAM                 1814MB  3783MB          90%     95%            H
ESP0(ok, active)                                                      H
 Control Processor     19.89%  100%            90%     95%            H
  DRAM                 683MB   1962MB          90%     95%            H
 QFP                                                                  H
  DRAM                 76244KB 524288KB        80%     90%            H
  IRAM                 8817KB  131072KB        80%     90%            H
  SRAM                 14KB    32KB            80%     90%            H
  TCAM                 28cells 131072cells     80%     90%            H
  CPU Utilization      7.00%   100%            90%     95%            H
ESP1(ok, standby)                                                     H
 Control Processor     19.89%  100%            90%     95%            H
  DRAM                 683MB   1962MB          90%     95%            H
 QFP                                                                  H
  DRAM                 76244KB 524288KB        80%     90%            H
  IRAM                 8817KB  131072KB        80%     90%            H
  SRAM                 14KB    32KB            80%     90%            H
  TCAM                 28cells 131072cells     80%     90%            H
  CPU Utilization      0.00%   100%            90%     95%            H
SIP0                                                                  H
 Control Processor     4.10%   100%            90%     95%            H
  DRAM                 307MB   460MB           90%     95%            H
SIP1                                                                  H
 Control Processor     1.10%   100%            90%     95%            H
  DRAM                 160MB   460MB           90%     95%            H
        **State Acronym: H - Healthy, W - Warning, C - Critical
```



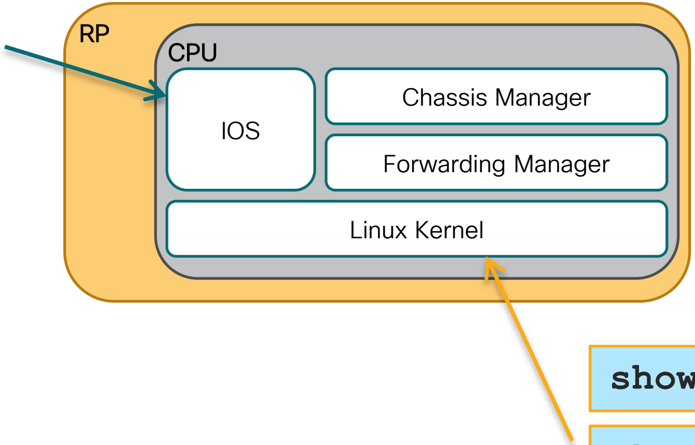
Other Show Commands
Improvements Improves interaction with TAC

Introduced in IOS-XE 3.14

```
show memory
```

```
show processes memory
```

```
show processes cpu
```



```
show memory platform
```

```
show processes memory platform
```

```
show processes cpu platform
```

Lab Access

1. Use AnyConnect and log in to the dCloud environment.
2. Open the Cisco CLI Analyzer Telnet/SSH Client and log in

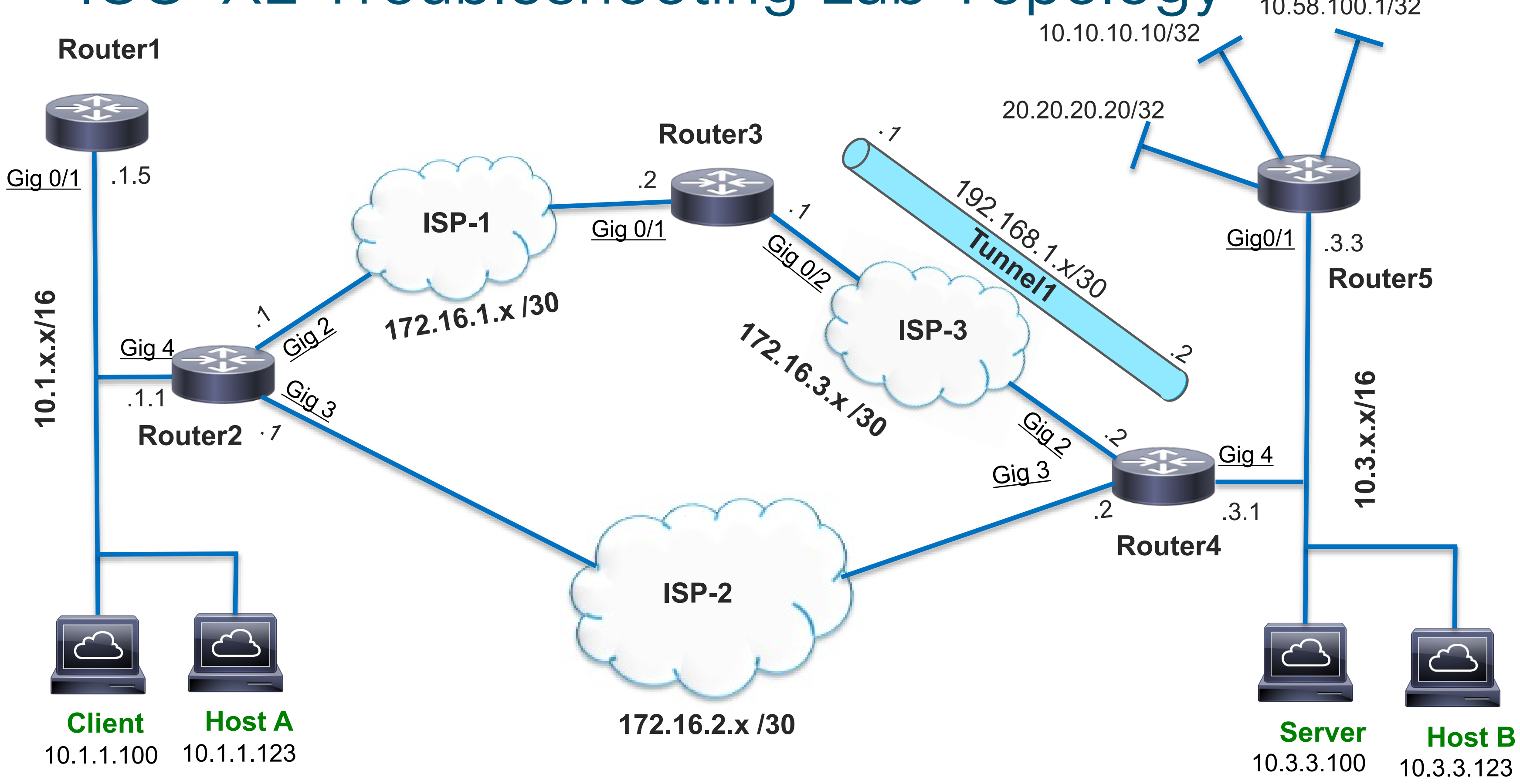
   Master Password: cisco!123
3. Create a new session for each of the devices in your POD

- Click on "Devices"
- Enter the search term "LTRARC-3500" and press Enter
- Click on the device name to connect, use the below credentials:

  Username: cisco

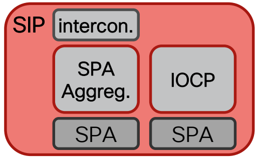
  Password: cisco
- Click on "Devices" and connect to the remaining devices

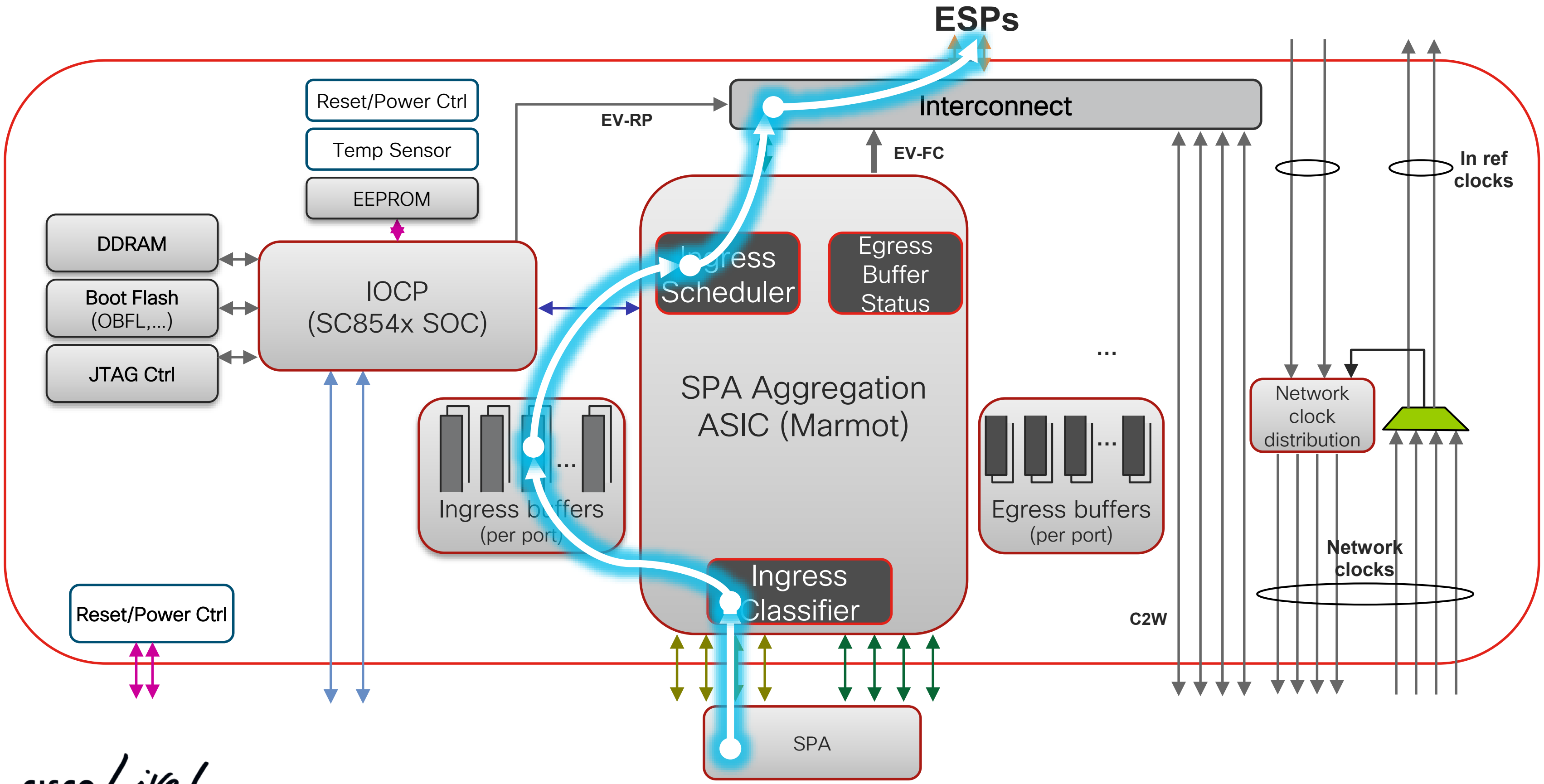
IOS-XE Troubleshooting Lab Topology



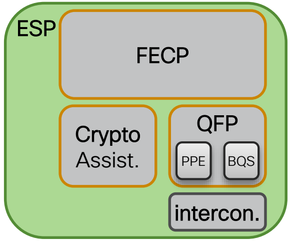
Day in the Life of a Normal Packet

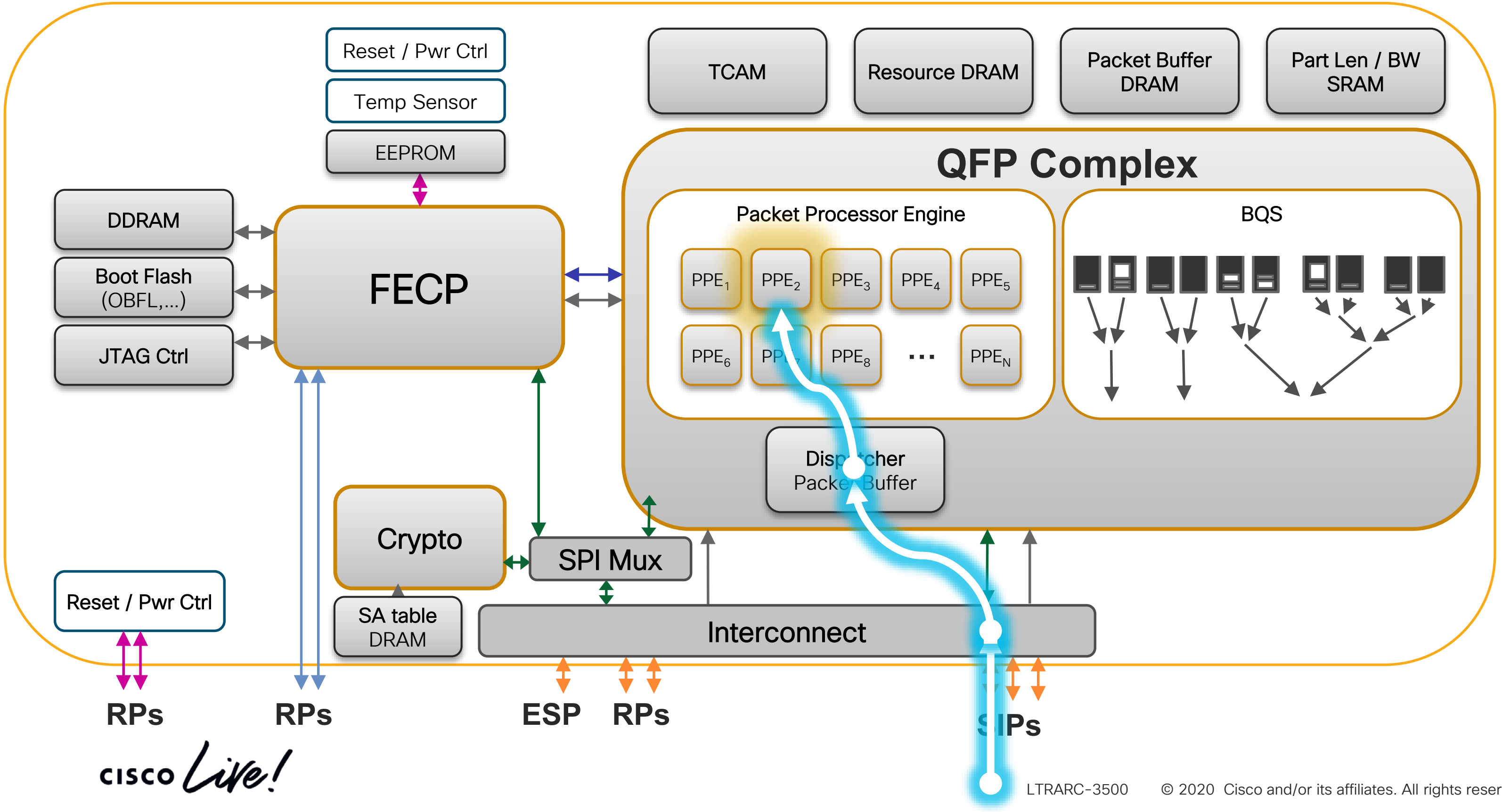
Ingress Packet Through SIP



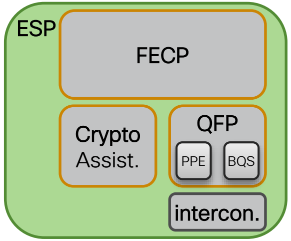


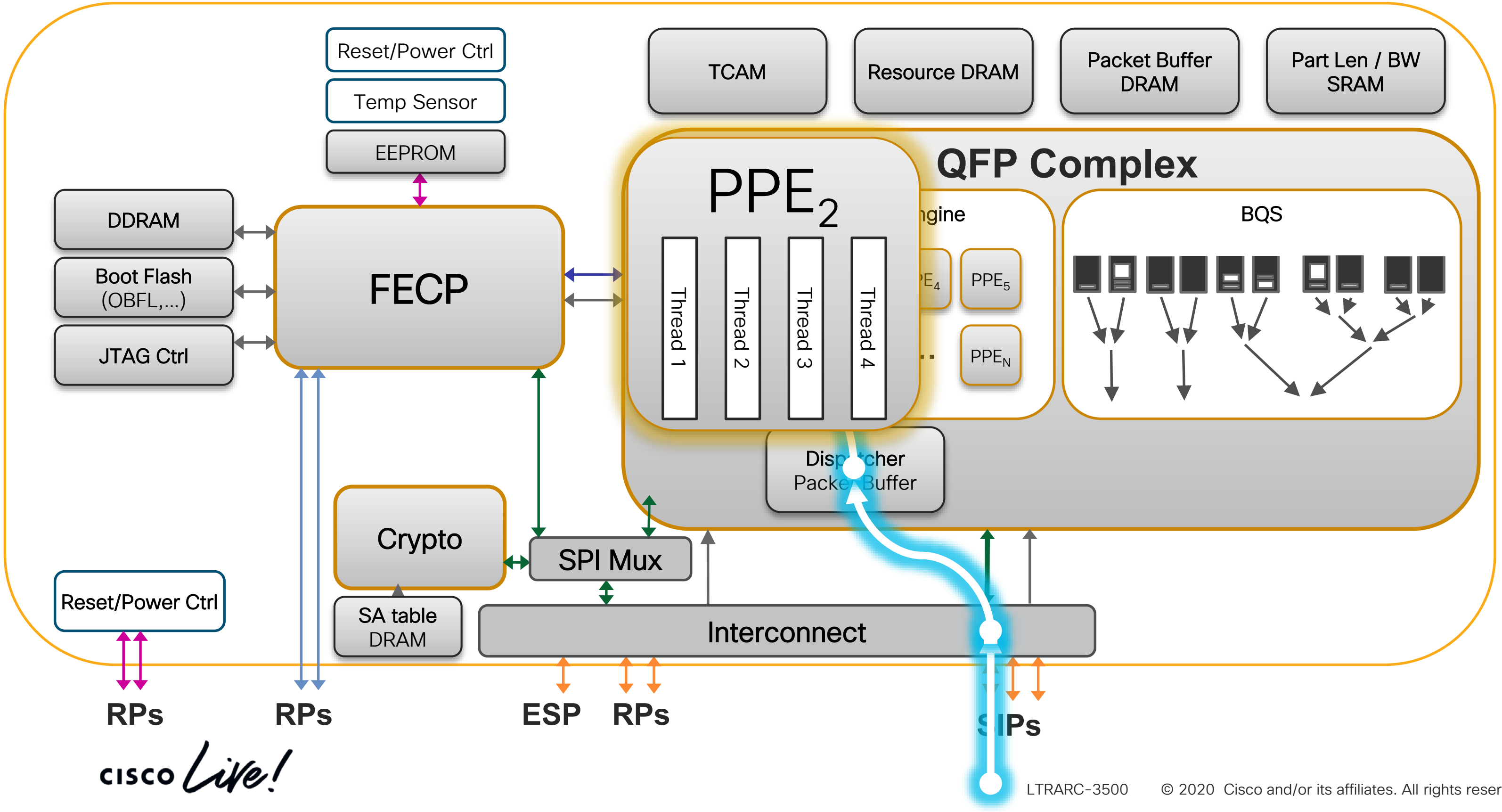
Ingress Packet Through ESP



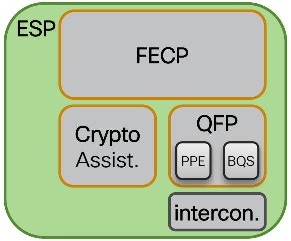


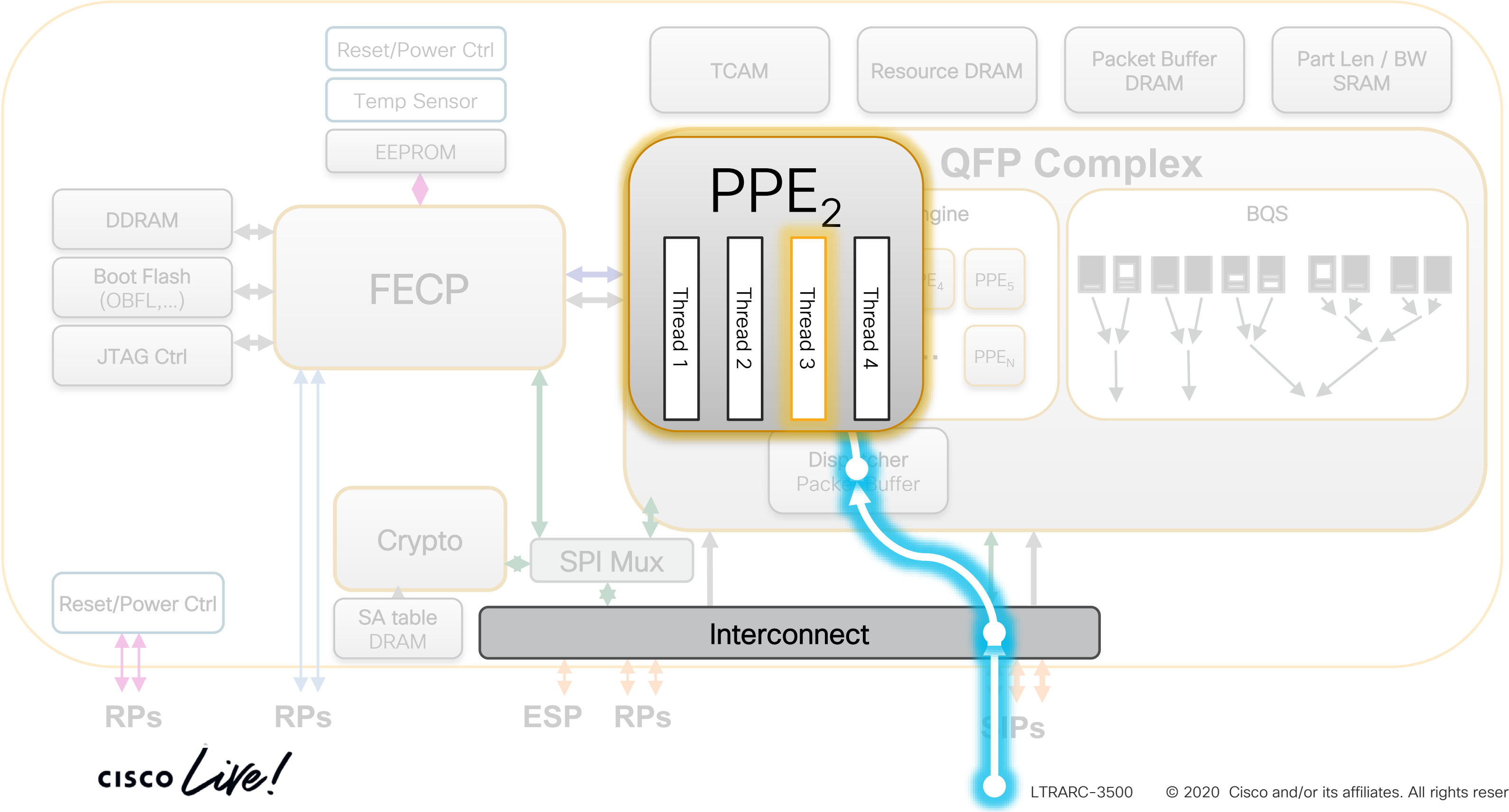
Packet Dispatched to PPE Core



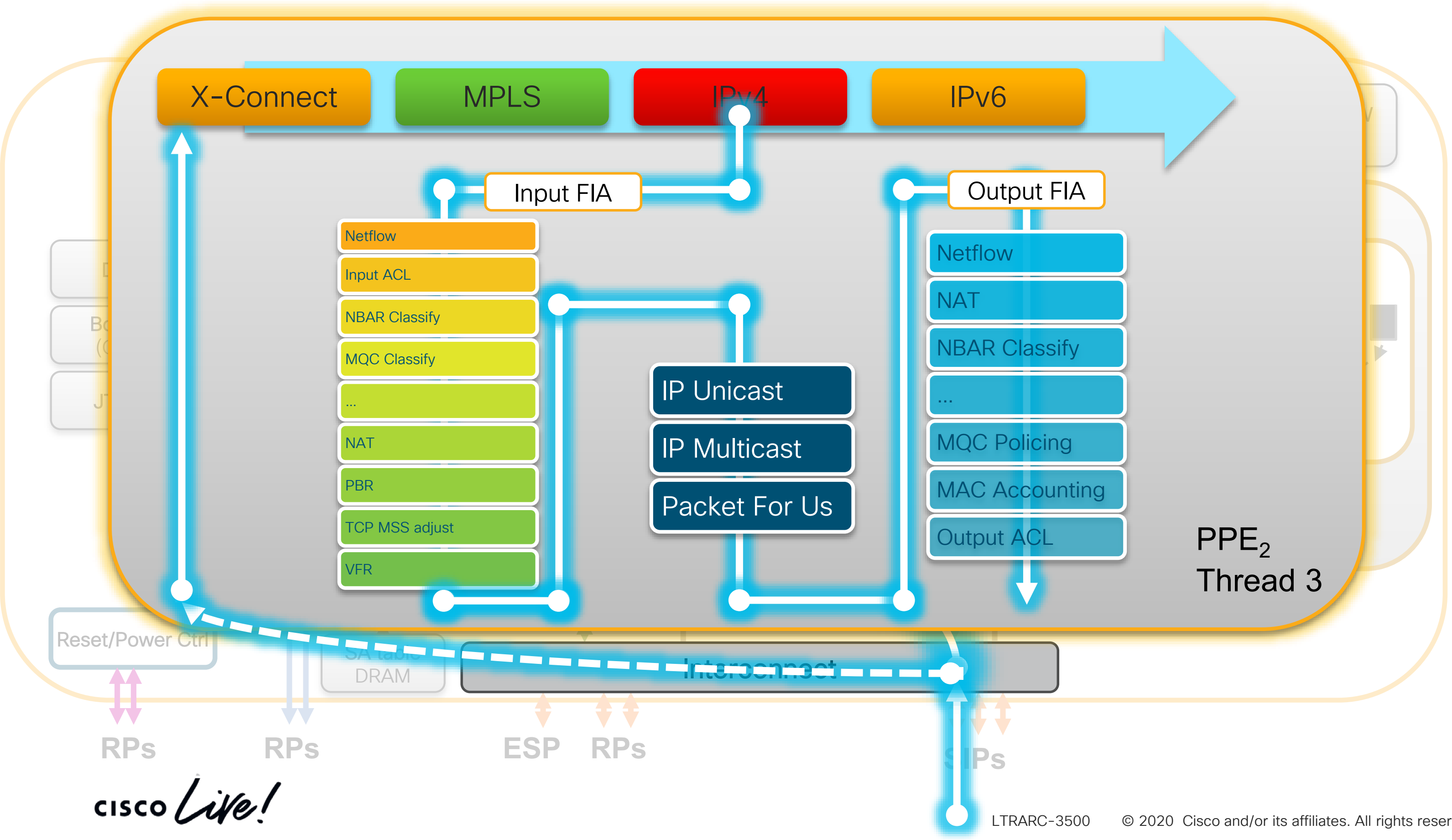


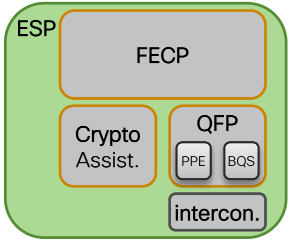
Packet Dispatched to PPE Thread



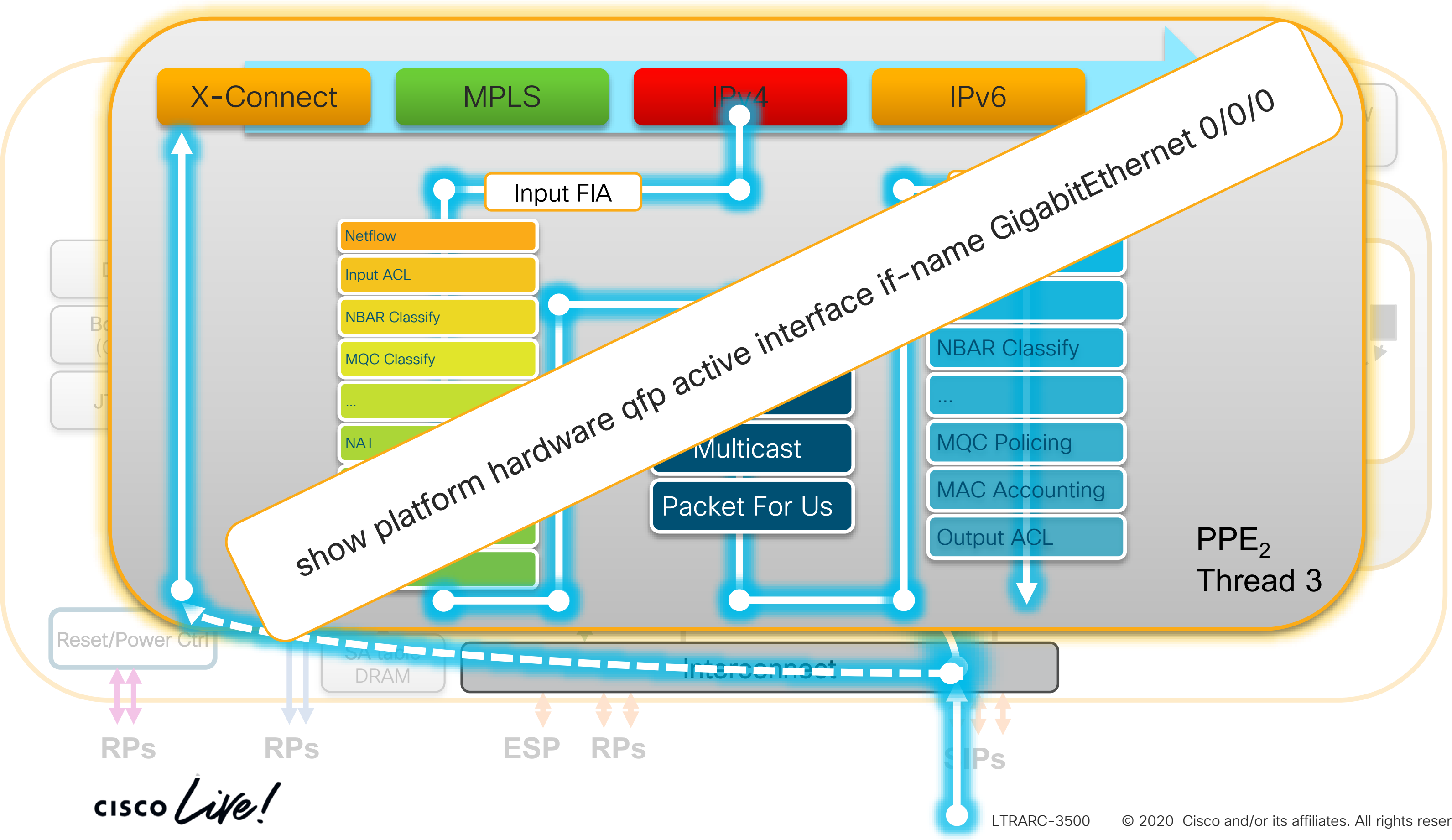


FIA's Applied on Packet by PPE Thread

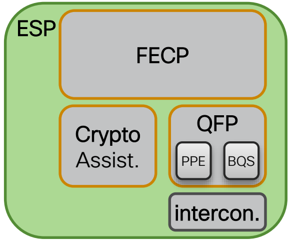




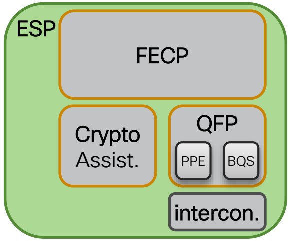
FIA’s Applied on Packet by PPE Thread

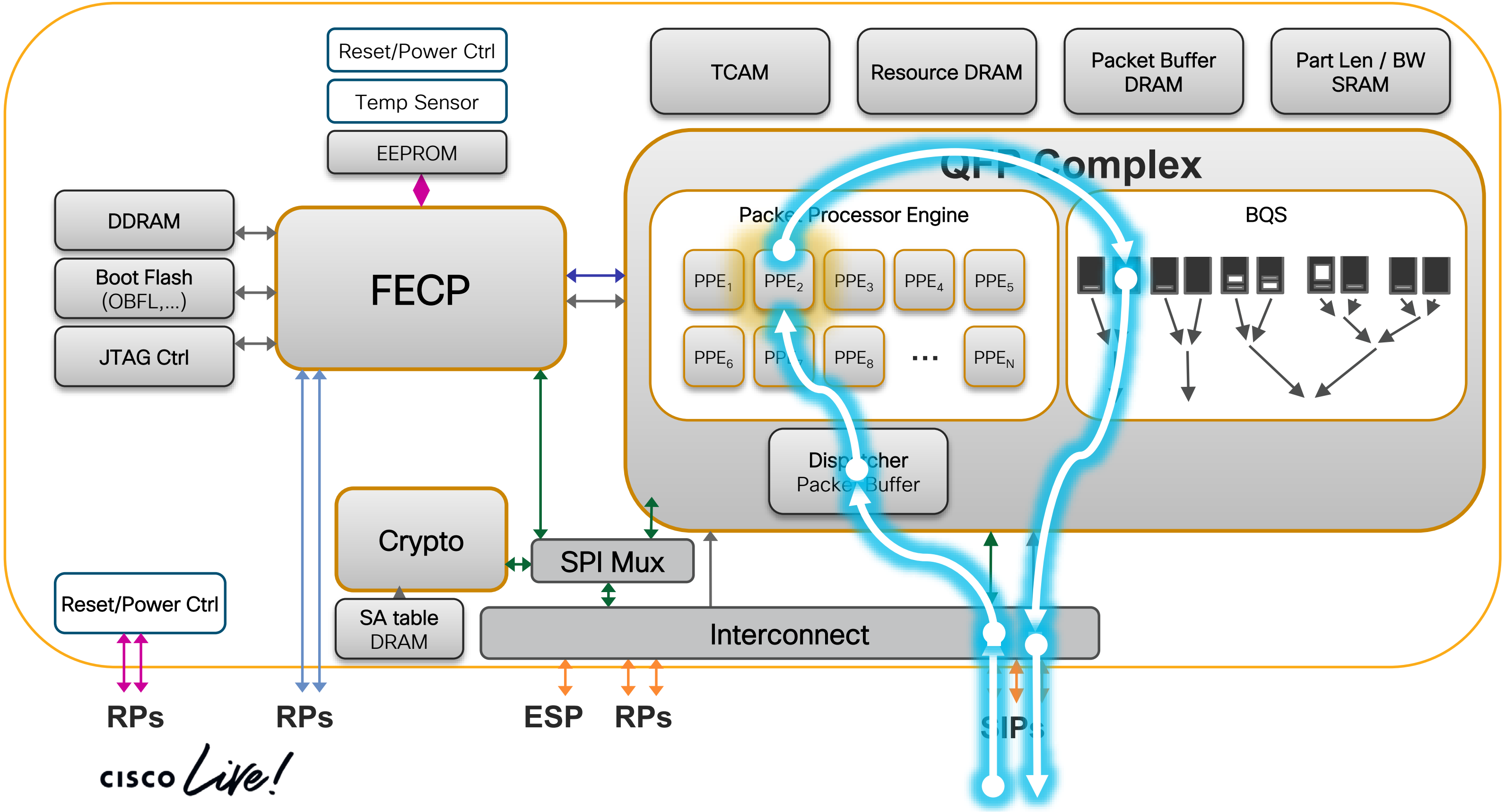


show platform hardware qfp active interface if-name GigabitEthernet 0/0/0

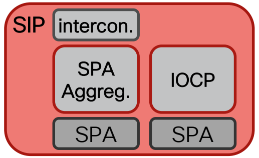


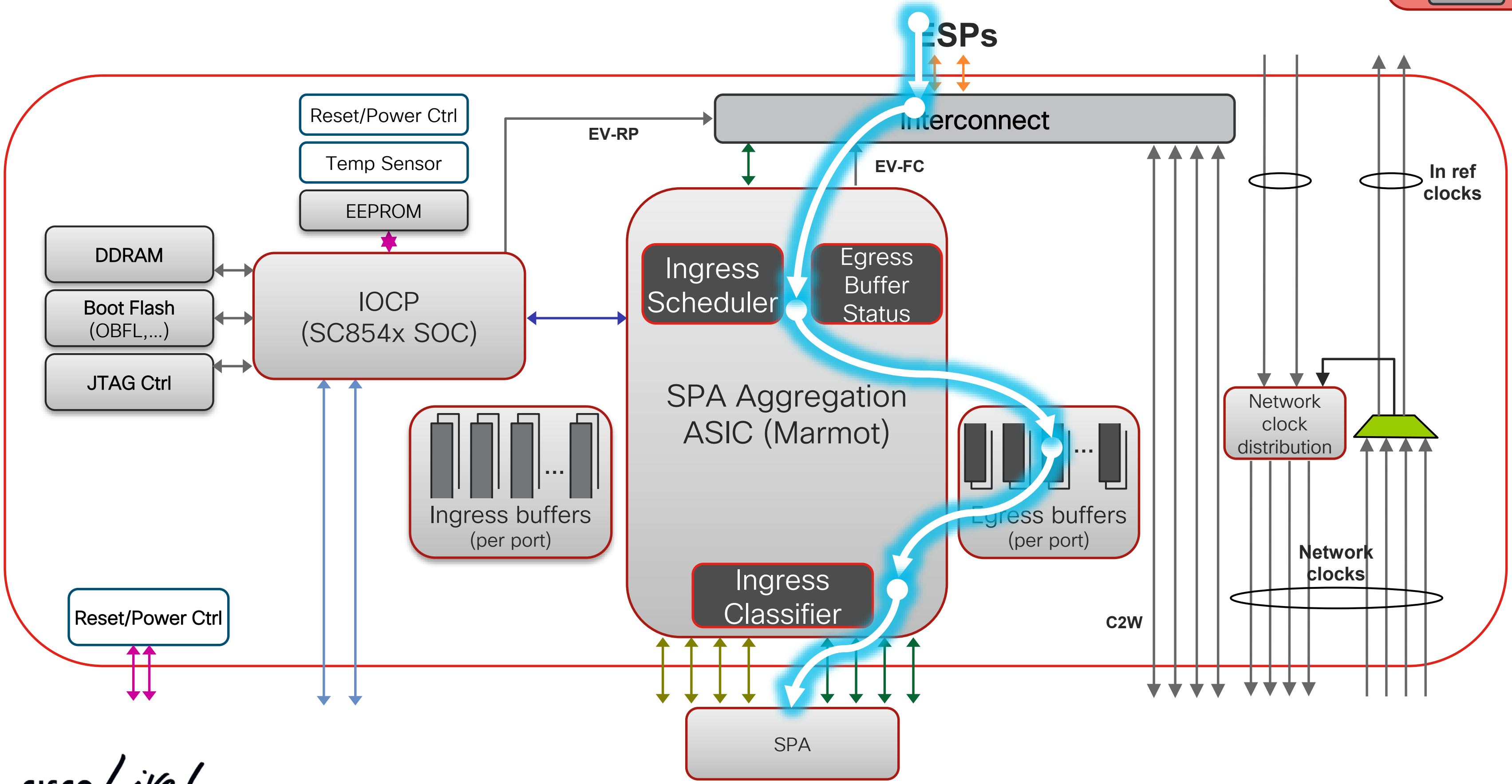
Packet Proceeding to BQS then SIP



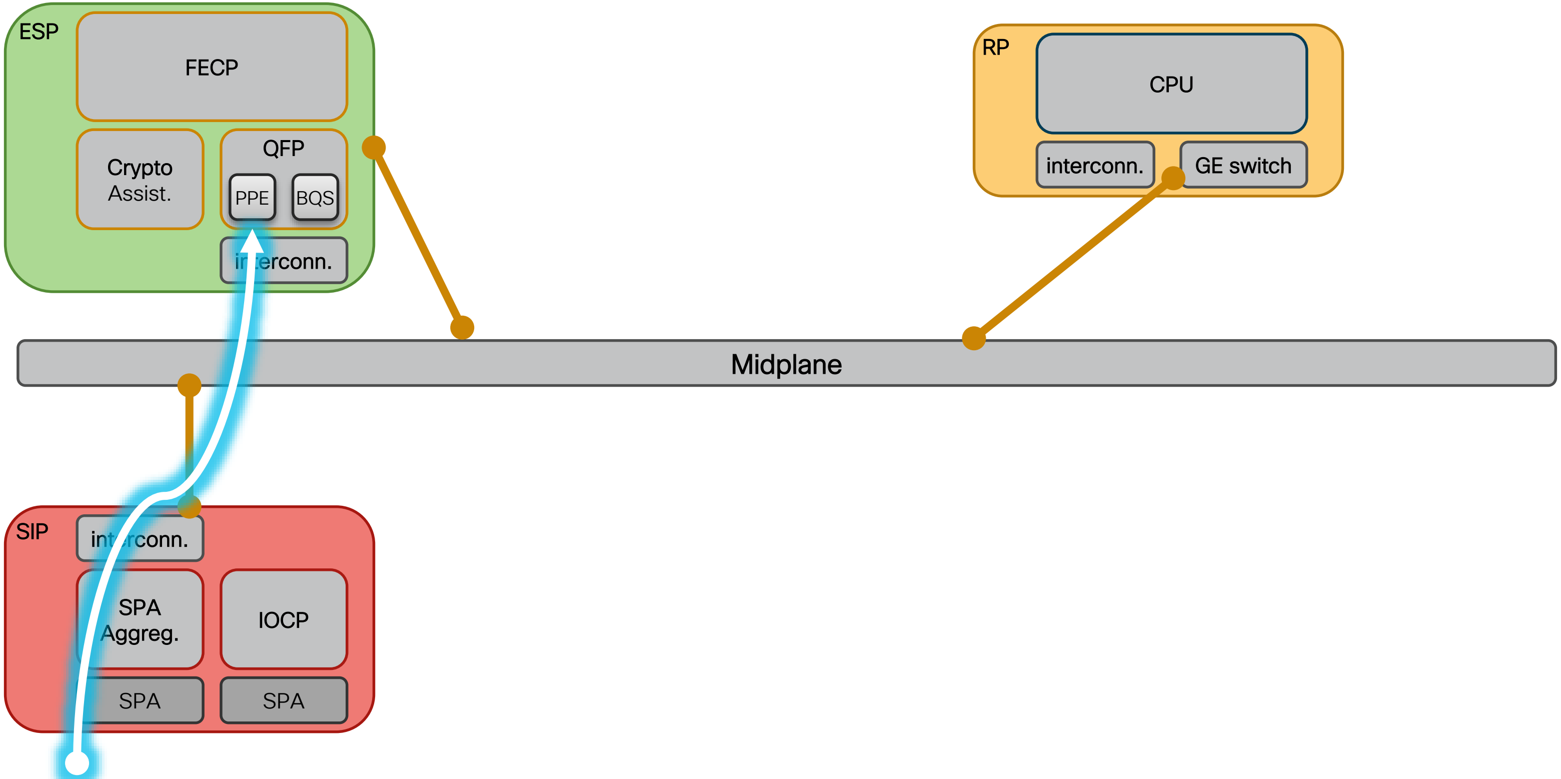


Egress Packet Through SIP

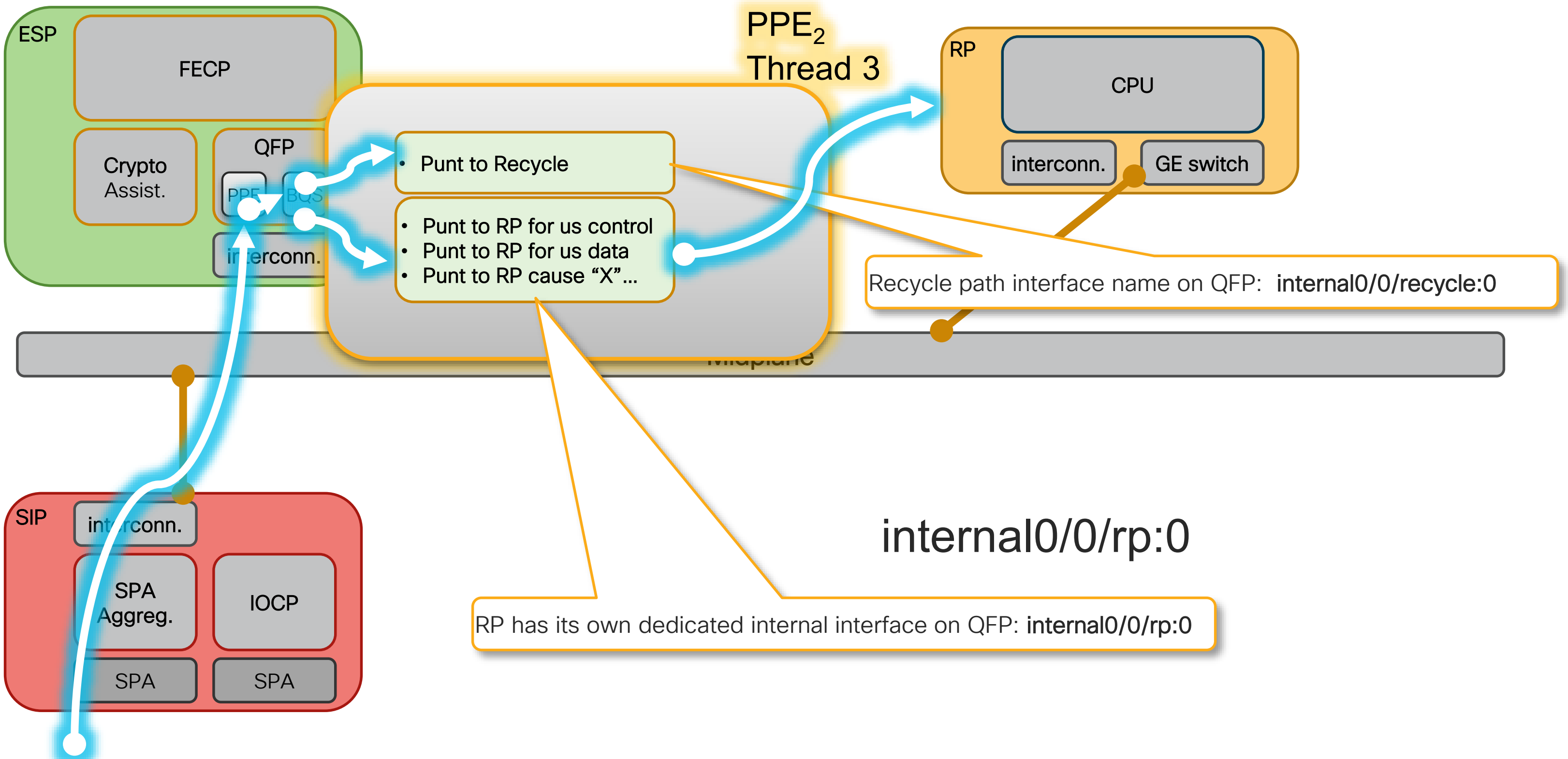




Punt Path: From QFP to Internal Destination



Punt Path: From QFP to Internal Destination



- Punt to Recycle

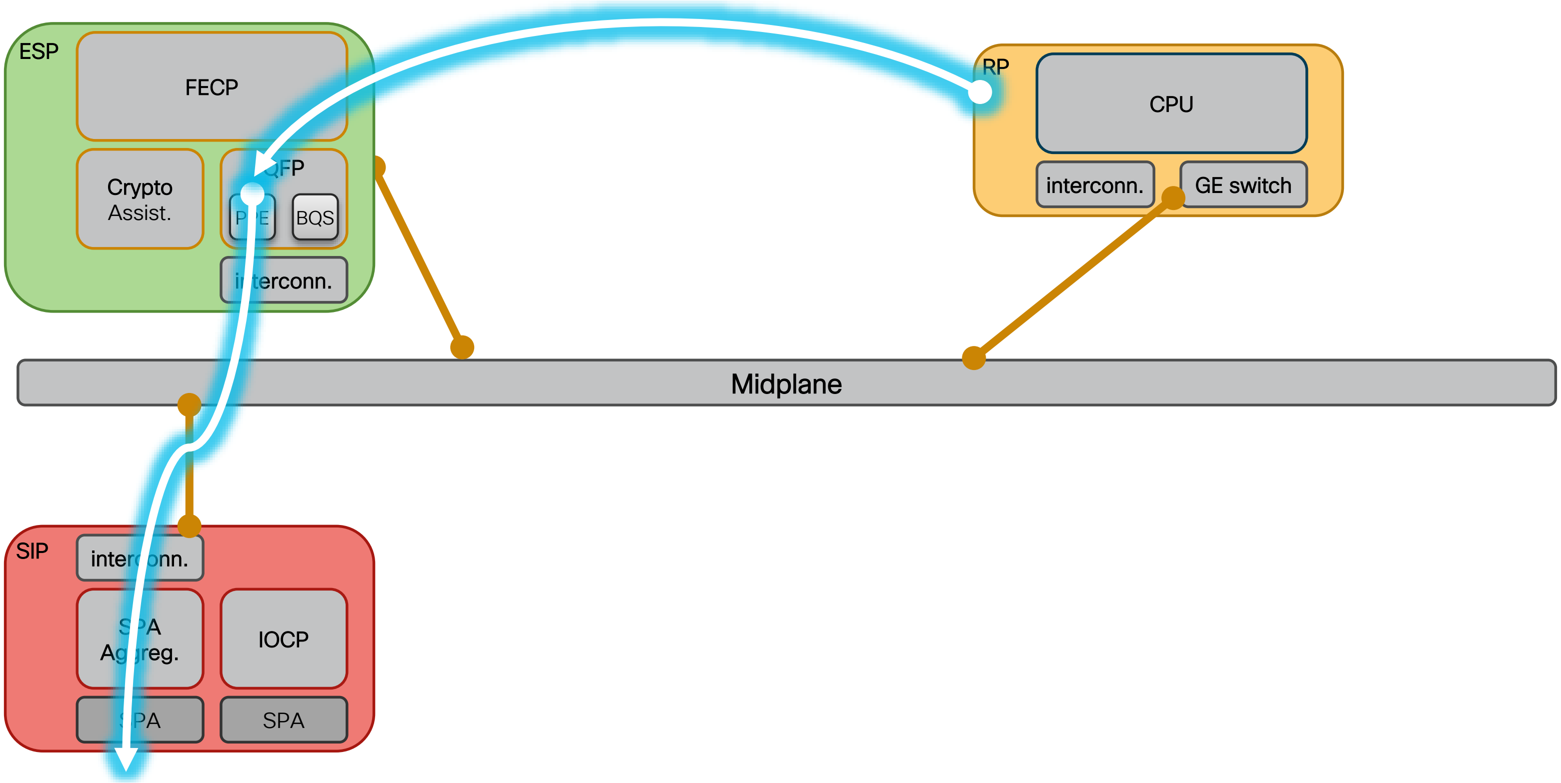
- Punt to RP for us control
- Punt to RP for us data
- Punt to RP cause “X”…

Recycle path interface name on QFP: internal0/0/recycle:0

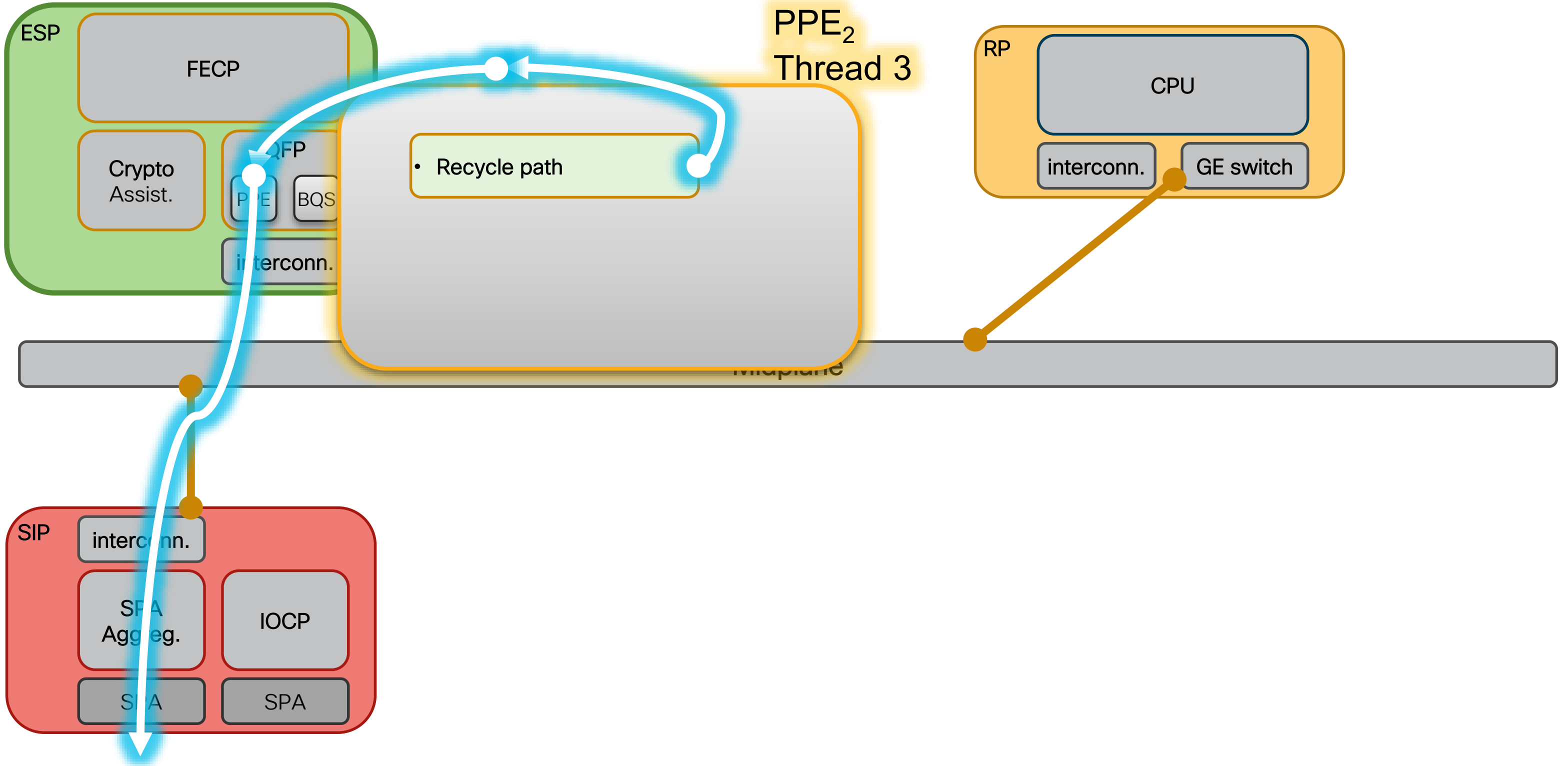
internal0/0/rp:0

RP has its own dedicated internal interface on QFP: internal0/0/rp:0

Inject Path: From RP via QFP to the network



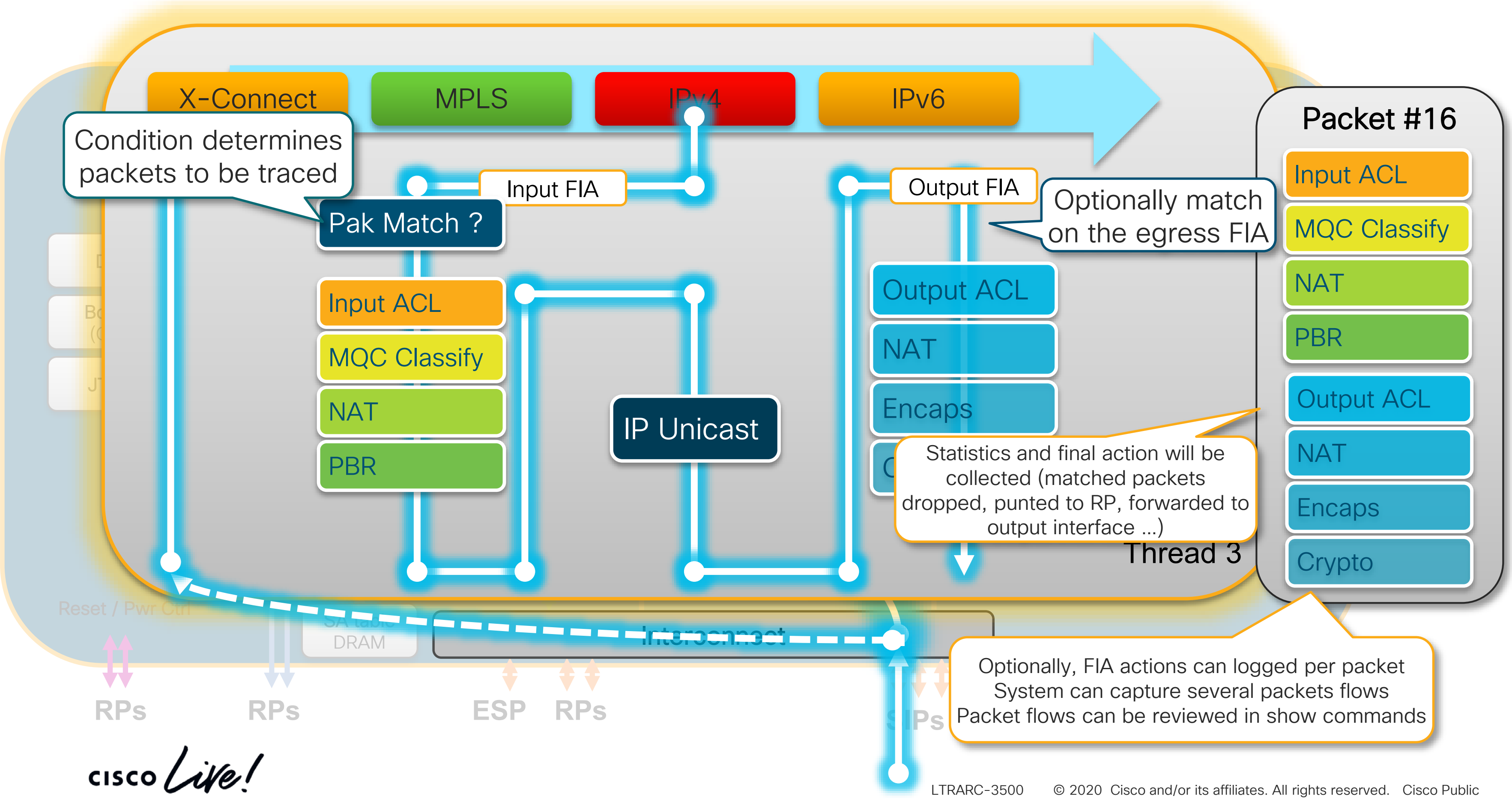
Inject Path: Recycling packet via QFP to network



Packet-tracer and FIA Debugger

The Packet Tracer and FIA Debugger

Introduced in IOS-XE 3.10



Condition determines packets to be traced

Optionally match on the egress FIA

Statistics and final action will be collected (matched packets dropped, punted to RP, forwarded to output interface …)

Optionally, FIA actions can logged per packet
System can capture several packets flows
Packet flows can be reviewed in show commands

Packet-Trace: Accounting

- Accounting keeps a count of all packet-trace interesting packets that enter and leave the “packet processor”.
- Three basic count groups.
  - Summary Counts
    - Packets Matched –packets that matched conditions
    - Packets Traced – packets that were traced
  - Arrival Counts
    - Ingress – packets entering via external interfaces
    - Inject* – number of packets seen as injected from control plane
  - Departure Counts
    - Forward – number of packets scheduled/queued for delivery
    - Punt* – number of packets punted to control plane
    - Drop* – number of packets specifically dropped by packet processing
    - Consume – number of packets consumed during packet process (e.g. ping request)

Packet-Trace: Summary Data

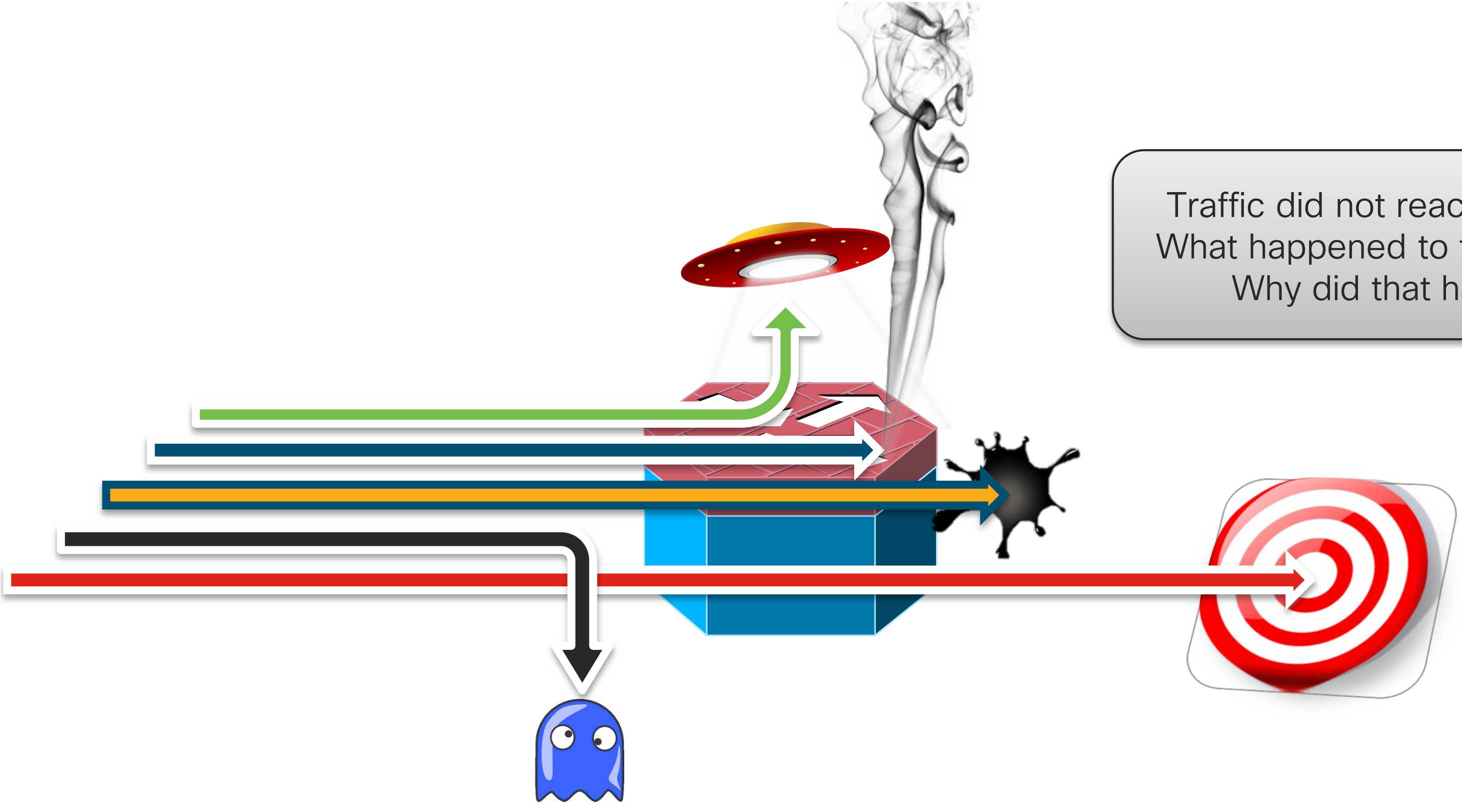
- Summary data is collected for a specified number of packets and includes:
  - Packet number (packet-trace specific packet number)
  - Input interface
  - Output interface
  - Final packet state and any punt/drop/inject codes
- Collecting summary data uses little performance over the normal packet processing
- Example usage:
  - To isolate which interfaces are dropping traffic so more detailed inspection can be used after applying interface specific conditions.

Packet-Trace: Path Data

- Path data may be collected per packet and is made up of different types of data:
  - Common path data (e.g. IP tuple)
  - Feature specific data (e.g. NAT)
  - Feature Invocation Array (FIA) trace – optionally enabled
  - Copy of all or part of the incoming and/or outgoing packet – optionally enabled
- Capturing path data with FIA trace and packet copy has the greatest impact on packet processing
  - FIA tracing creates many path data entries costing instructions and DRAM writes
  - Packet copy creates many DRAM read/writes
- Packet-trace will only affect the performance of packets traced (i.e. those matched by the user provided conditions)

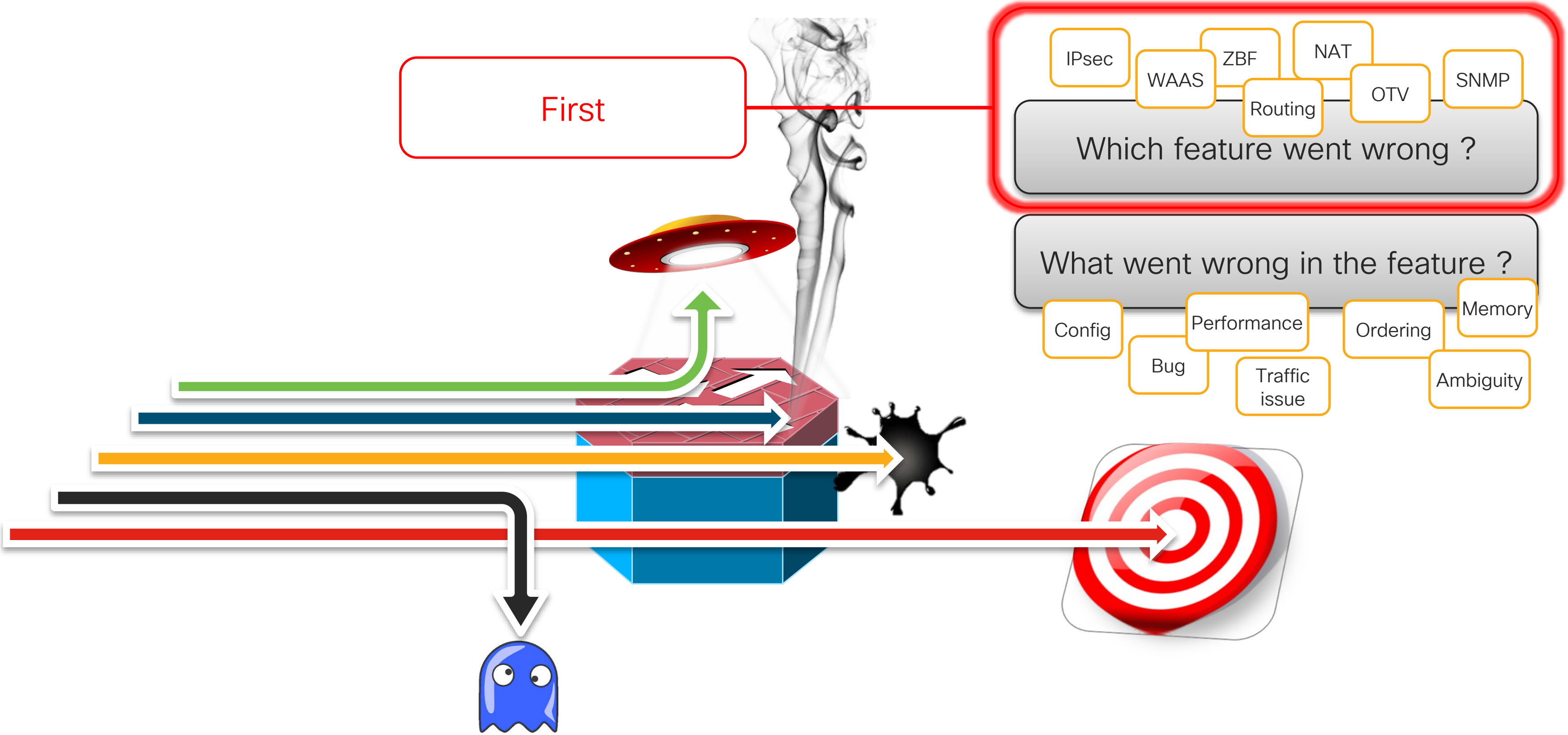
Debugging Strategies

Everyday Situations



Traffic did not reach its target!
What happened to that packet?
Why did that happen?

Everyday Situations



First

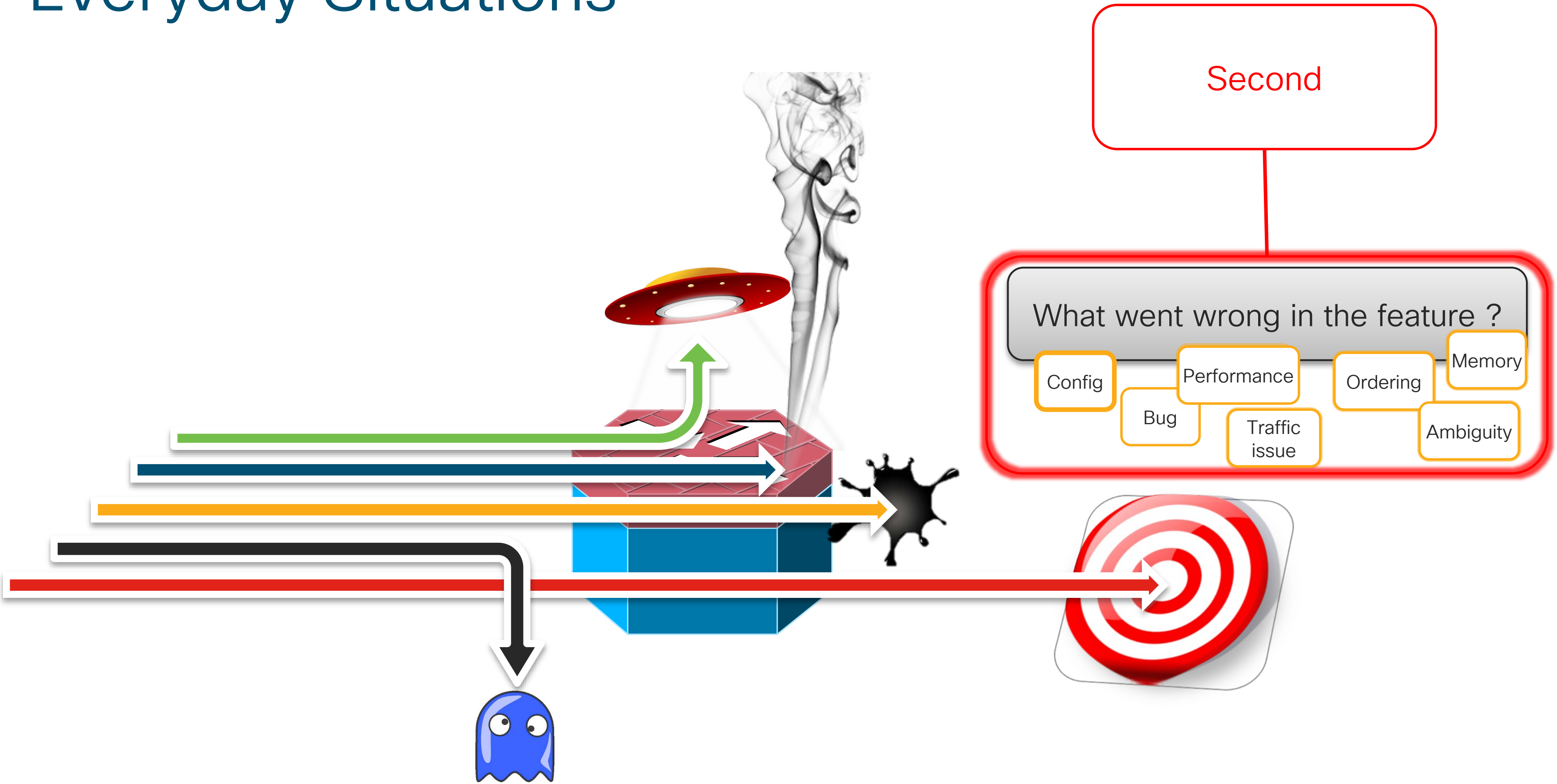
Which feature went wrong ?

IPsec WAAS ZBF NAT Routing OTV SNMP

What went wrong in the feature ?

Config Bug Performance Traffic issue Ordering Memory Ambiguity

Everyday Situations



Second

What went wrong in the feature ?

Config

Bug

Performance

Traffic issue

Ordering

Memory

Ambiguity

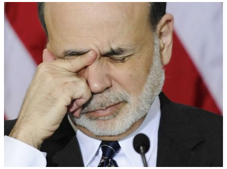
Debugging Strategies to Date

IOSd Control Plane

- ACL + show access-list,…
- show interface / ip route / bgp …

Top Down

Rock bottom



Platform Control Plane

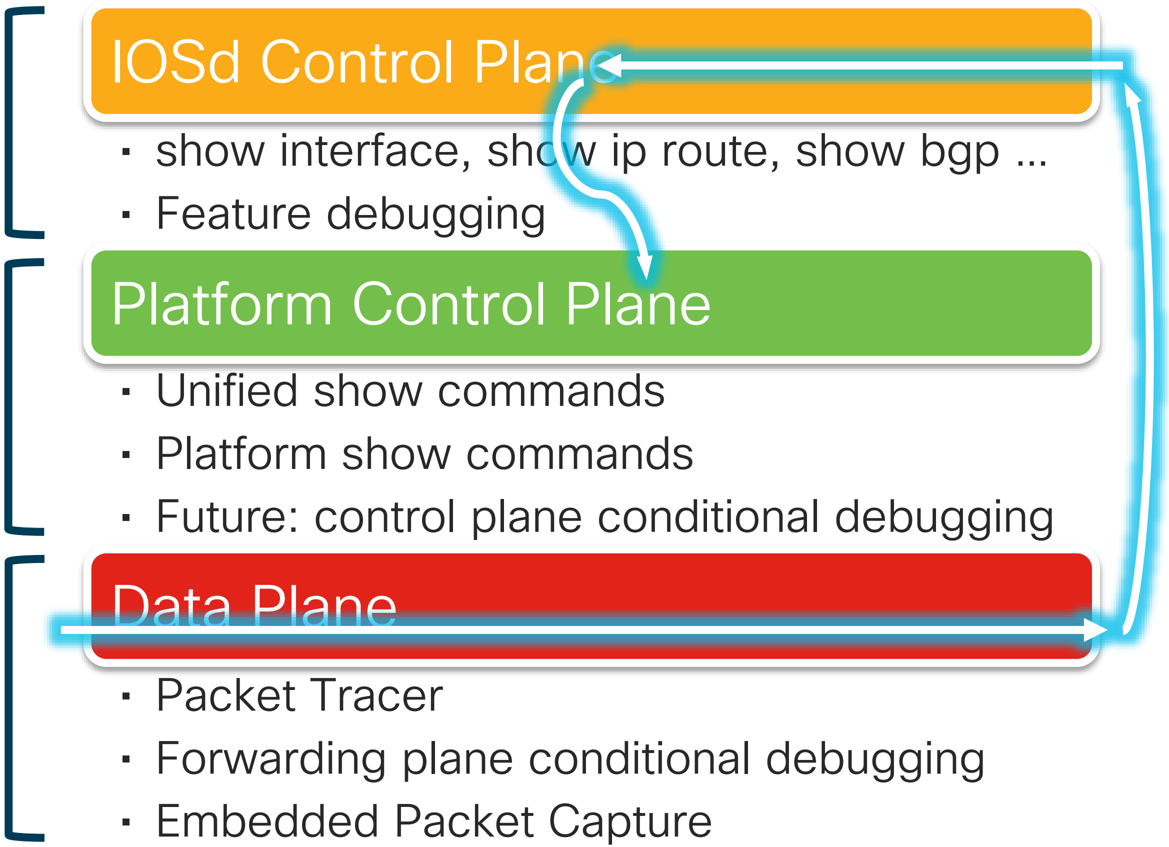
- ESP “stuff”
- e.g. show platform … hard to remember

Let’s change that!!

Data Plane

- ESP “stuff”
- More arcane show platform …

New Debugging Strategy



IOSd Control Plane

- show interface, show ip route, show bgp …
- Feature debugging

Platform Control Plane

- Unified show commands
- Platform show commands
- Future: control plane conditional debugging

Data Plane

- Packet Tracer
- Forwarding plane conditional debugging
- Embedded Packet Capture

Troubleshooting Tools and Capabilities

Embedded Packet Capture

The Embedded Packet Capture

One way of capturing packets…

Introduced in IOS-XE 3.7

- Shows whether packets have been received or sent
- Shows what packets look like
- Excellent tool but insufficient
- Requires export to decoder
- Config and decode made easy with

```
Device# monitor capture mycap start
Device# monitor capture mycap access-list v4acl
Device# monitor capture mycap limit duration 1000
Device# monitor capture mycap interface GigabitEthernet 0/0/1 both
Device# monitor capture mycap buffer circular size 10
Device# monitor capture mycap start
Device# monitor capture mycap export tftp://10.1.88.9/mycap.pcap
Device# monitor capture mycap stop
```

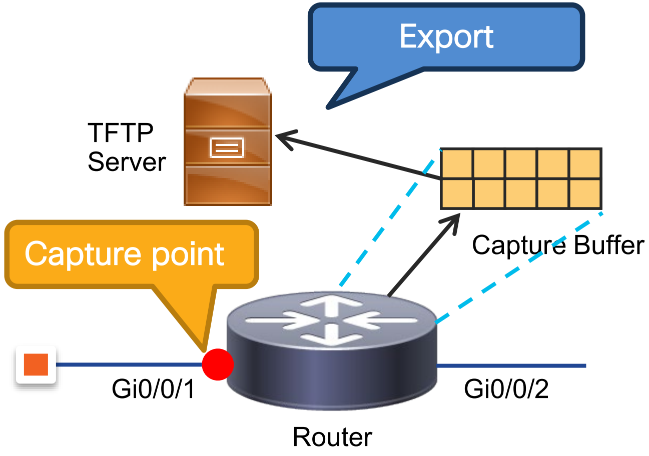
```
Device# show monitor capture mycap buffer dump

0
 0000:   01005E00 00020000 0C07AC1D 080045C0    ..^...........E.
 0010:   00300000 00000111 CFDC091D 0002E000    .0..............
 0020:   000207C1 07C1001C 802A0000 10030AFA    .........*......
 0030:   1D006369 73636F00 0000091D 0001        ..example.......

1
 0000:   01005E00 0002001B 2BF69280 080046C0    ..^.....+.....F.
 0010:   00200000 00000102 44170000 0000E000    .  ......D.......
```

https://cway.cisco.com/tools/CaptureGenAndAnalyse/

```
 0020:   000207C1 07C1001C 88B50000 08030A6E    ...............n
 0030:   1D006369 73636F00 0000091D 0001        ..example.......
```

Embedded Packet Capture

- EPC added to FIA
  - Beginning of ingress FIA
  - End of egress FIA
- Matched packets are copied
- Copied packets get punted to RP
- Original packets processed as usual
- Capture buffer on RP can be exported to .pcap file



Use EPC to Troubleshoot Packet Corruptions

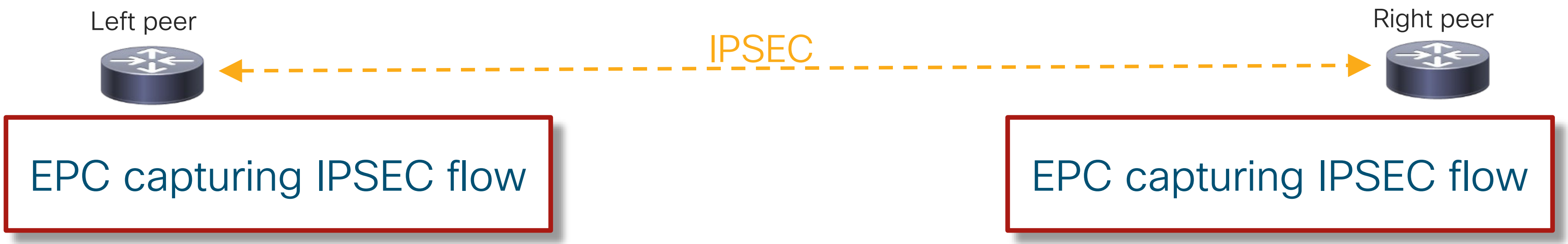
An Use Case Study of Data Collection Automation

- IPSec integrity check makes it sensitive to packet corruption in the network

```
%CRYPTO-4-RECVD_PKT_MAC_ERR: decrypt: mac verify failed for connection id=695
local=192.168.14.2 remote=192.168.13.2 spi=7C4E759F seqno=00000001
```

- Problem Challenges:
  - Highly intermittent
  - Requires Packet Capture on both ends to prove network corruption
- Solution
  - Run continuous EPC with a circular buffer on both tunnel end points
  - Use EEM with SNMP to synchronize and stop capture on both sides
  - Notify the network administration by email
  - Upload and examine both captures for evidence of corruption

EPC in real life: troubleshooting packet corruption



Left peer

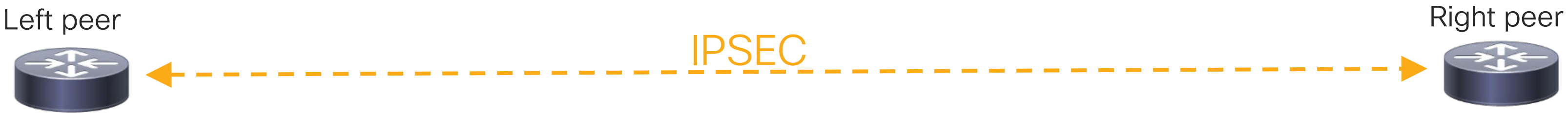
IPSEC

Right peer

EPC capturing IPSEC flow

EPC capturing IPSEC flow

EPC in real life: troubleshooting packet corruption

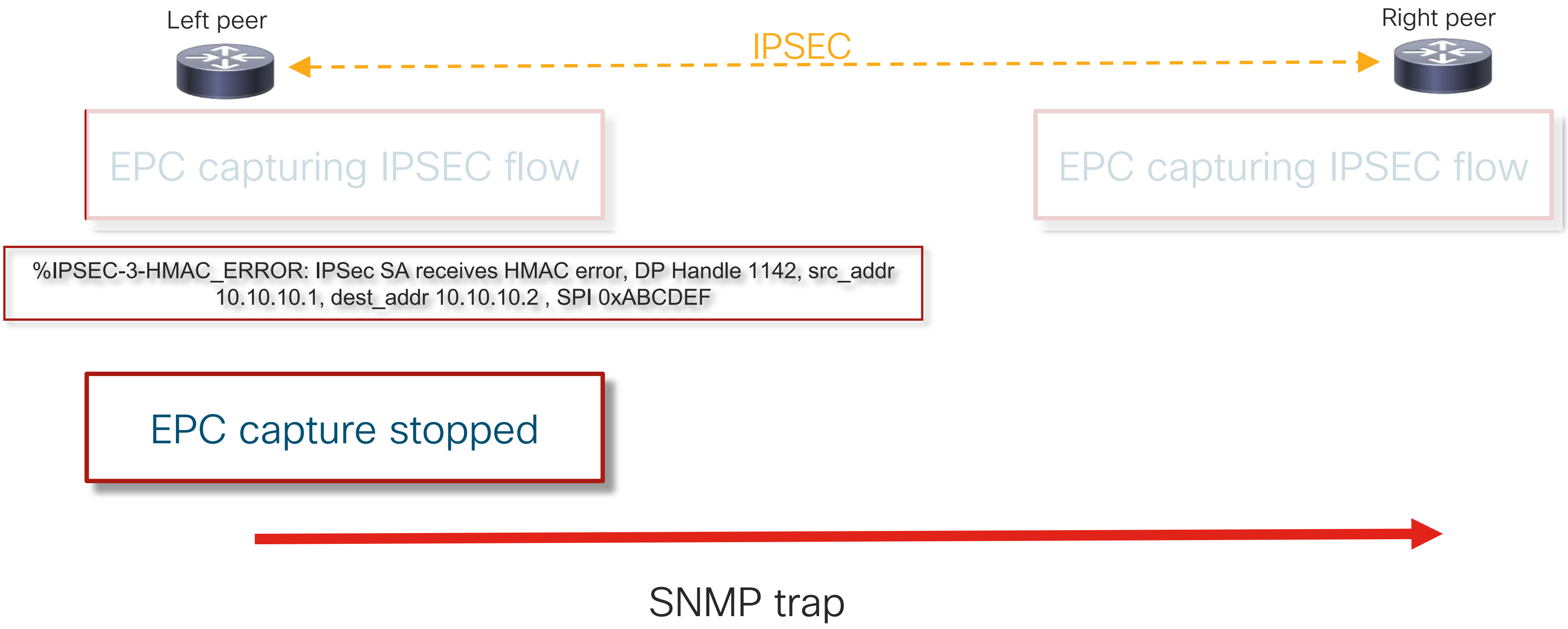


EPC capturing IPSEC flow

EPC capturing IPSEC flow

```
%IPSEC-3-HMAC_ERROR: IPSec SA receives HMAC error, DP Handle 1142, src_addr
10.10.10.1, dest_addr 10.10.10.2 , SPI 0xABCDEF
```

EPC in real life: troubleshooting packet corruption



EPC capturing IPSEC flow

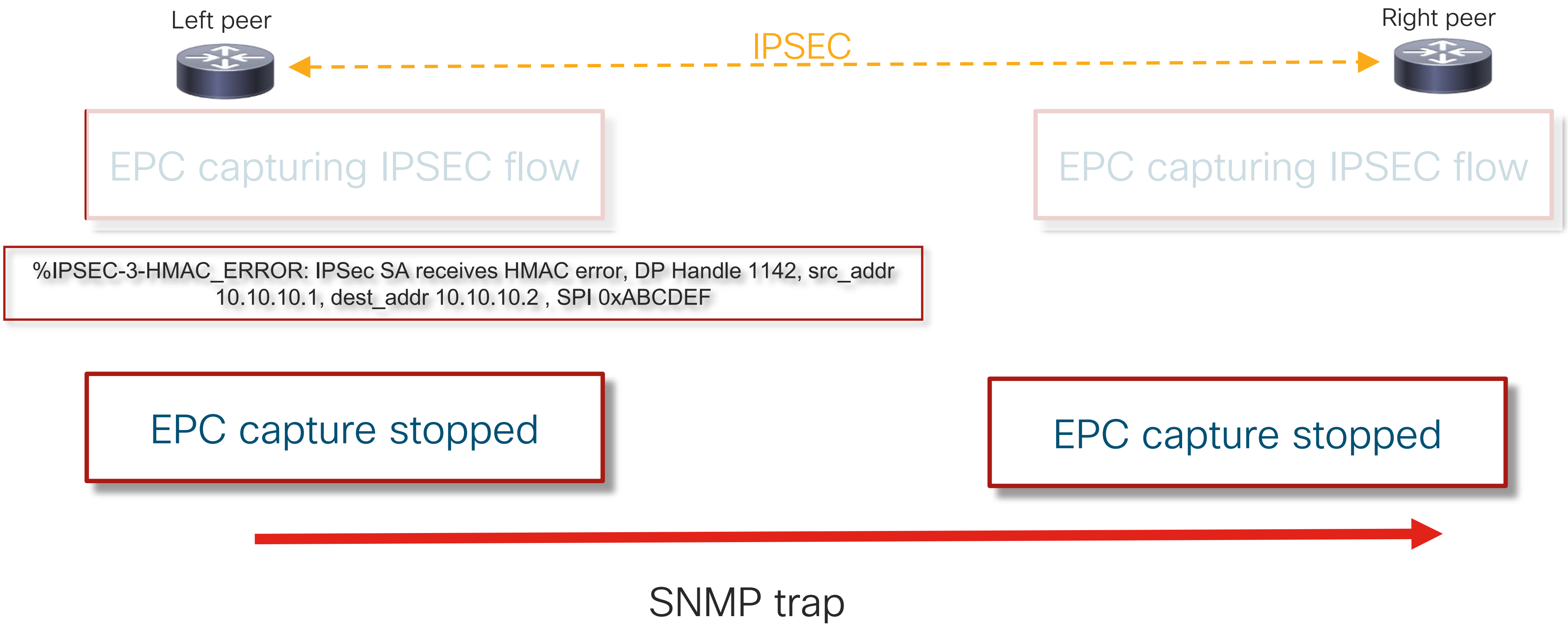
EPC capturing IPSEC flow

%IPSEC-3-HMAC_ERROR: IPSec SA receives HMAC error, DP Handle 1142, src_addr 10.10.10.1, dest_addr 10.10.10.2 , SPI 0xABCDEF

EPC capture stopped

SNMP trap

EPC in real life: troubleshooting packet corruption



Left peer

IPSEC

Right peer

EPC capturing IPSEC flow

EPC capturing IPSEC flow

```
%IPSEC-3-HMAC_ERROR: IPSec SA receives HMAC error, DP Handle 1142, src_addr
10.10.10.1, dest_addr 10.10.10.2 , SPI 0xABCDEF
```

EPC capture stopped

EPC capture stopped

SNMP trap

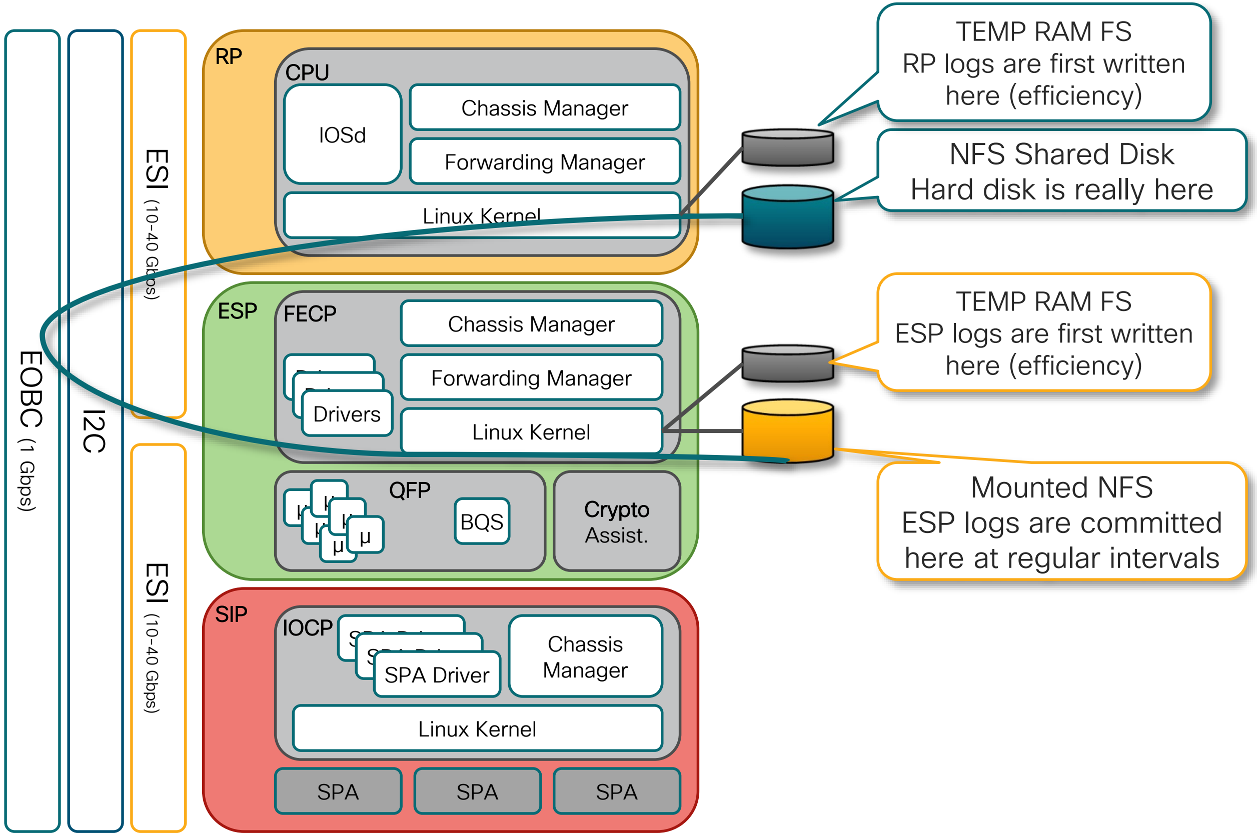
EPC in real life: configs

```
left-peer#show run | se event
snmp-server enable traps event-manager
snmp-server host 10.10.10.1 public  event-manager
event manager applet detect_bad_packet
event syslog pattern " IPSEC-3-HMAC_ERROR "
action 1.0 cli command "enable"
action 2.0 cli command "monitor capture stop test"
action 3.0 syslog msg "Packet corruption detected and
capture stopped!"
action 4.0 snmp-trap intdata1 123456 strdata "”

*Jan 14 21:34:51.639: %IPSEC-3-HMAC_ERROR: IPSec SA
receives HMAC error, DP Handle 1142, src_addr 10.10.10.1,
dest_addr 10.10.10.2 X, SPI 0xABCDEF
*Jan 14 21:34:51.858: %BUFCAP-6-DISABLE: Capture Point
test disabled.
left-peer#
*Jan 14 21:34:51.860: %HA_EM-6-LOG:
detect_bad_packet: Packet corruption detected and capture
stopped!
```

```
right-peer#show run | se event
event manager applet detect_bad_packet
event snmp-notification oid 1.3.6.1.4.1.9.10.91.1.2.3.1.9.
oid-val "123456" op eq src-ip-address 10.10.10.2
action 1.0 cli command "enable"
action 2.0 cli command "monitor capture stop test"
action 3.0 syslog msg "Packet corruption detected and
capture stopped!"
right-peer#
right-peer#
*Jan 14 21:34:52.337: %HA_EM-6-LOG:
detect_bad_packet: Packet corruption detected and capture
stopped!
right-peer#
```

Understanding and Extracting Platform Tracelogs

Platform Tracing and Logging



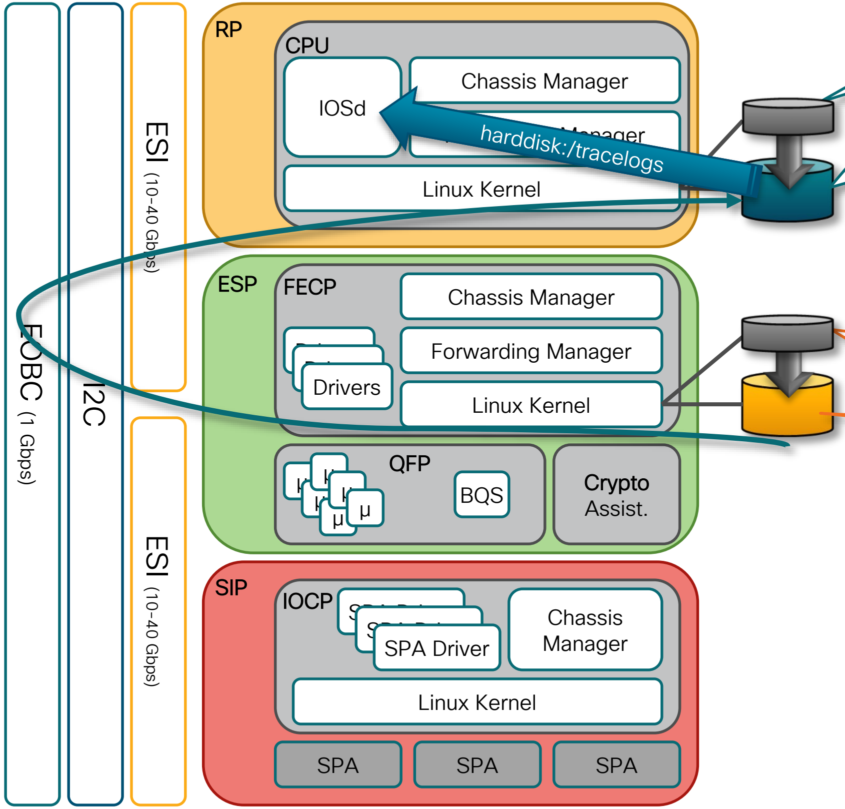
TEMP RAM FS
RP logs are first written here (efficiency)

NFS Shared Disk
Hard disk is really here

TEMP RAM FS
ESP logs are first written here (efficiency)

Mounted NFS
ESP logs are committed here at regular intervals

Important Logs



fman_rp_R[0|1]-0.log

Under **/harddisk/tracelogs**
fman_rp_R[0|1]-0.log.<timestamp>
fman-fp_R0.log.<timestamp>
cpp_cp_F[0|1]-0.log.<timestamp>

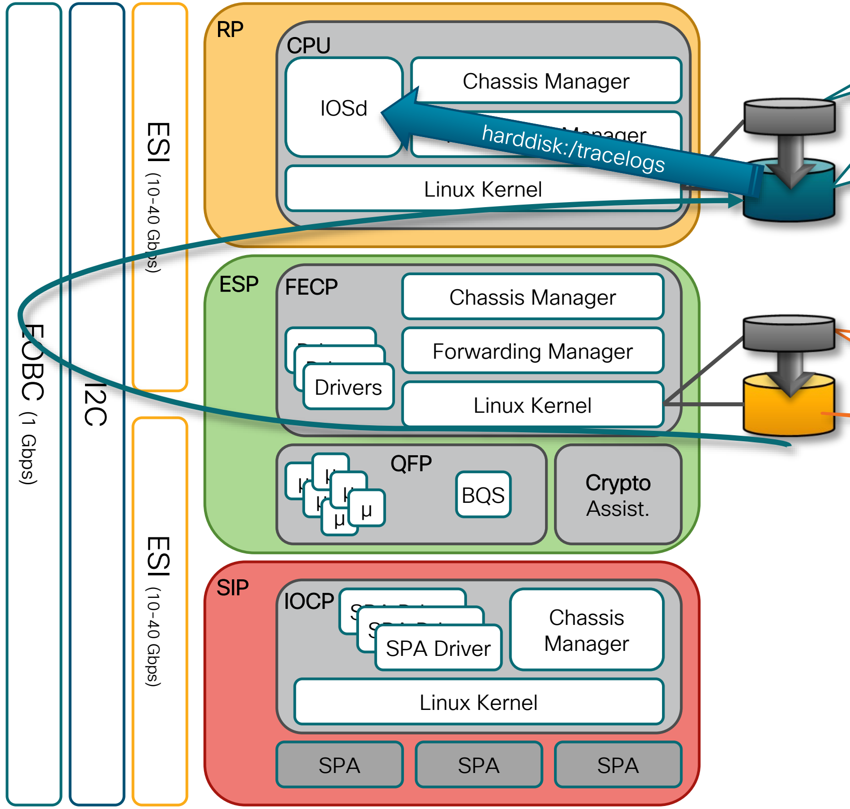
fman_fp_F[0|1]-0.log
cpp_cp_F[0|1]-0.log

Under **/harddisk/tracelogs/**
fman-fp_R0.log.<timestamp>
cpp_cp_F[0|1]-0.log.<timestamp>

What log files are important?

- Important log files to get for security issues:
  - fman_rp_R[0|1].log (under /tmp/rp/trace directory on RP)
  - fman-fp_F[0|1]-0.log (under /tmp/fp/trace directory on ESP
  - cpp_cp_F[0|1]-0.log (under /tmp/fp/trace directory on ESP)
- All these logs get rotated and are copied to /harddisk/tracelogs directory on active RP.
- Look for the relevant log files depending on the time of the failure
- By default, all ERR messages are logged ➔ should be the first things to look for

Important Logs



fman_rp_R[0|1]-0.log

Under **/harddisk/tracelogs**
fman_rp_R[0|1]-0.log.<timestamp>
fman-fp_R0.log.<timestamp>
cpp_cp_F[0|1]-0.log.<timestamp>

fman_fp_F[0|1]-0.log
cpp_cp_F[0|1]-0.log

Under **/harddisk/tracelogs/**
fman-fp_R0.log.<timestamp>
cpp_cp_F[0|1]-0.log.<timestamp>

Introduced in IOS-XE 16.7

New Logging Framework: Show logging process

Show logging process <process name> internal

```
#csr1000v-1# show logging process fman internal
excuting cmd on chassis local ...
Collecting files on current[local] chassis.
Total # of files collected = 4
Decoding files:
/harddisk/tracelogs/tmp_trace/fman_fp_F0-0.21047_0.20180109071524.bin: DECODE(592:0:592:10)
/harddisk/tracelogs/tmp_trace/fman_rp_R0-0.14852_0.20180109071523.bin: DECODE(21:0:21:11)
/harddisk/tracelogs/tmp_trace/fman_rp_pmanlog_R0-0.14682_0.20180109071455.bin: DECODE(25:0:25:1)
/harddisk/tracelogs/tmp_trace/fman_fp_image_pmanlog_F0-0.20738_0.20180109071508.bin:
DECODE(28:0:28:1)
<……decoded files>
```

Introduced in IOS-XE 16.7

New Logging Framework: Show logging profile

Show logging profile <profile name> internal

```
csr1000v-1# show logging profile iwan internal
executing cmd on chassis local ...
Collecting files on current[local] chassis.
Total # of files collected = 16
Decoding files:
2018/01/09 07:14:55.770 {fman_rp_pmanlog_R0-0}{1}: [fman_rp_pmanlog] [14682]: (note):  gdb port 9905 allocated
2018/01/09 07:14:55.812 {fman_rp_pmanlog_R0-0}{1}: [fman_rp_pmanlog] [14682]: (note):  swift_repl port 8005
allocated
2018/01/09 07:14:55.882 {fman_rp_pmanlog_R0-0}{1}: [fman_rp_pmanlog] [14682]: (info): (std):
/tmp/sw/rp/0/0/rp_security/mount/usr/binos/conf/pman.sh: line 424: sigusr1_func: readonly function
2018/01/09 07:14:55.902 {fman_rp_pmanlog_R0-0}{1}: [fman_rp_pmanlog] [14682]: (note):  process scoreboard
/tmp/rp/process/fman_rp%rp_0_0%0 fman_rp%rp_0_0%0.pid is 1458
22018/01/09 07:14:55.902 {fman_rp_pmanlog_R0-0}{1}: [fman_rp_pmanlog] [14682]: (note):
fman_rp%rp_0_0%0.gdbport is 9905
2018/01/09 07:14:55.902 {fman_rp_pmanlog_R0-0}{1}: [fman_rp_pmanlog] [14682]: (note):
fman_rp%rp_0_0%0.swift_replport is 8005
```

Wrapping up...

Key Session Takeaways

- IOS-XE Platforms are complex but troubleshooting doesn’t have to be
  - Use Resource Monitoring for consolidated view of system health
  - Use the platform CPU/memory command variant for in-depth resource check
- Detailed Discussion on Packet Forwarding
  - Data plane Packet Tracing is your friend!
  - Use the right tool for the job!
- Discussed Troubleshooting Strategy and Tools
  - Control vs. Data Plane
  - Embedded Packet Capture
  - Leverage Platform Logs for in-depth troubleshooting
  - End-to-end platform debugging workflow and strategies

Complete your online session survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (starting on 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 Content Catalog on ciscolive.com/emea.

Cisco Live sessions will be available for viewing on demand after the event at ciscolive.com.

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

cisco Live!

Thank you

You make **possible**