
cisco
You make possible

Advanced troubleshooting of the NCS5500 (IOS-XR) made easy

Andrzej Mieczkowski, Technical Consulting Engineer
Vadim Zhovtanyuk, Technical Leader

BRKSPG-2165

cisco Live!
Barcelona | January 27-31, 2020

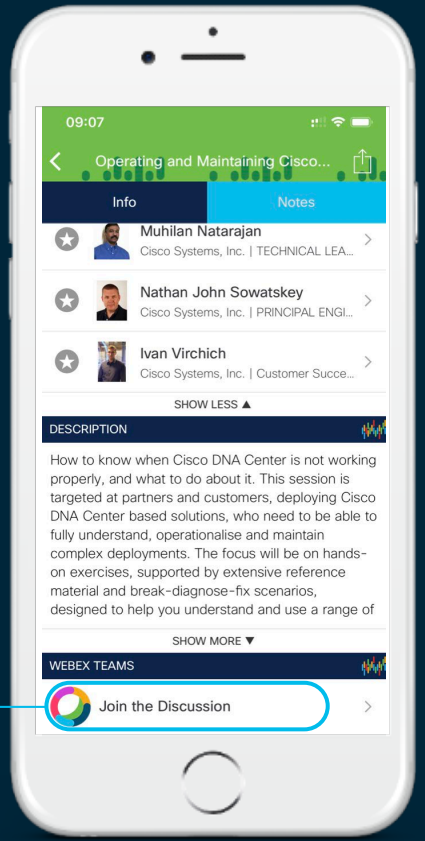
Cisco Webex Teams

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

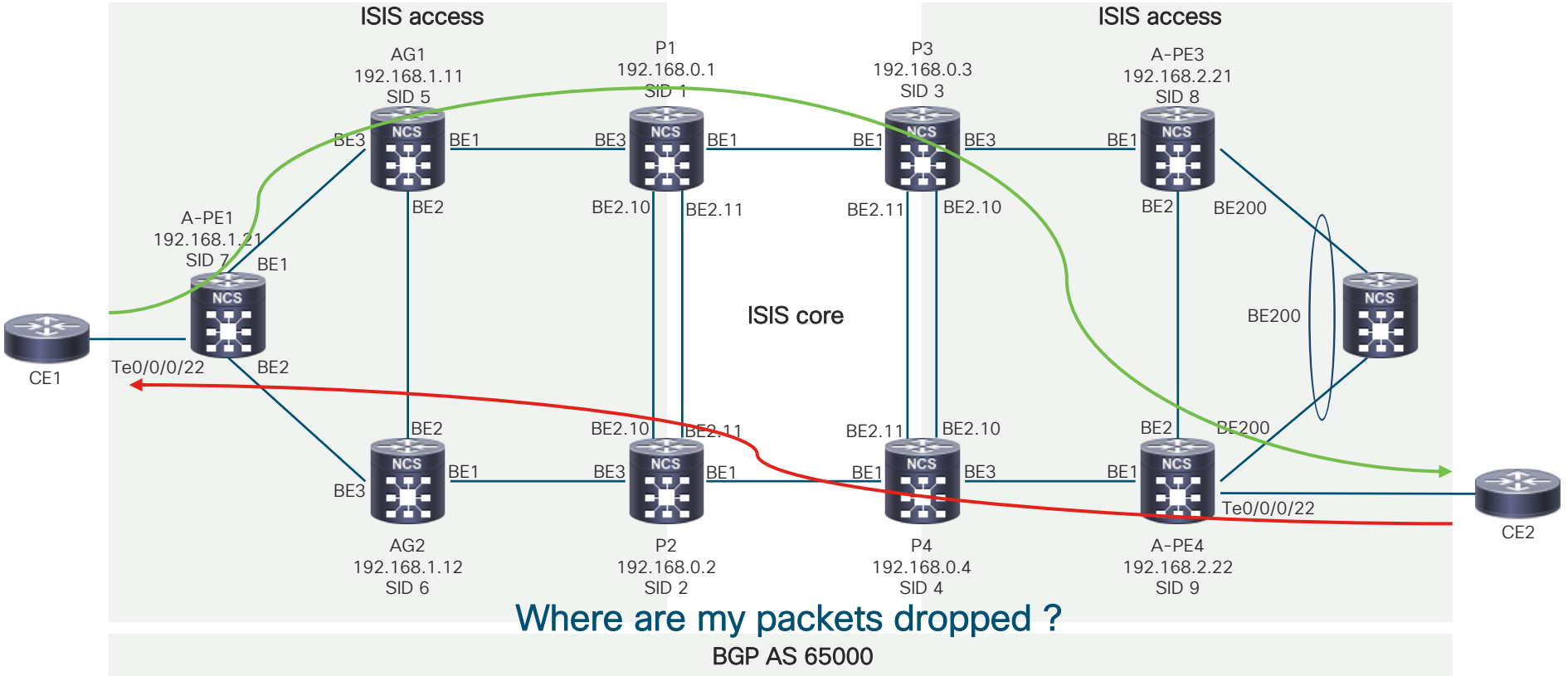


Agenda

- Unicast forwarding troubleshooting
- Unicast forwarding troubleshooting Demo
- Multicast forwarding troubleshooting
- Multicast forwarding troubleshooting Demo
- Netflow troubleshooting on NCS5500
- Netflow troubleshooting Demo

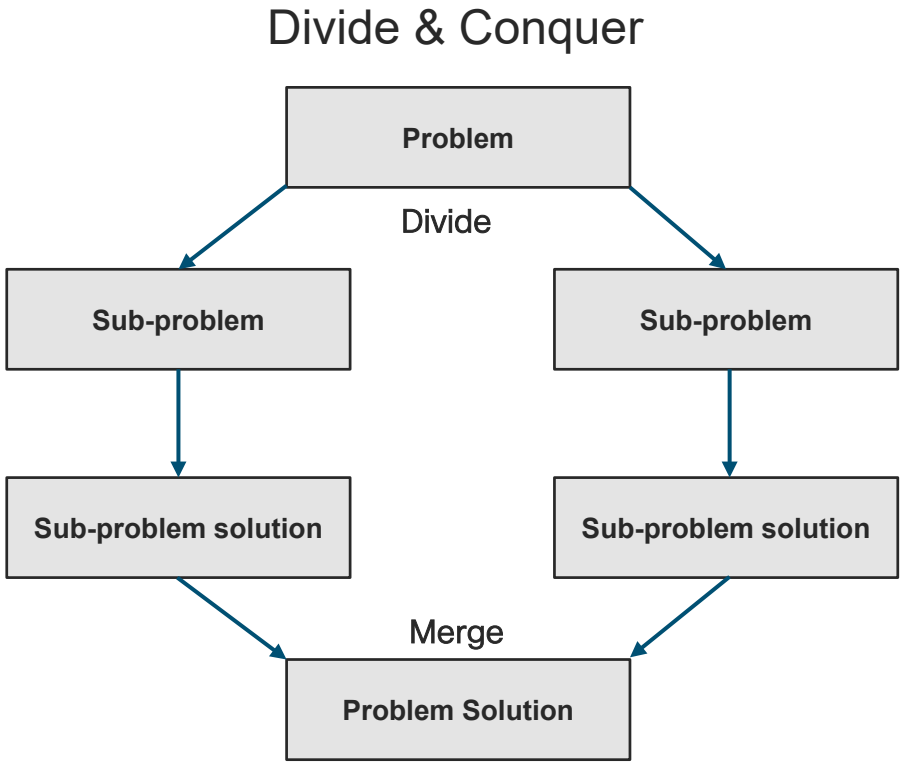
Unicast forwarding troubleshooting

Network topology and legend



How to find the device which is dropping the traffic?

- Use Divide & Conquer approach
- Examinee ingress & egress path separately
- Check if traffic from the source is reaching destination node
  - If not, troubleshoot ingress path
  - If yes, then troubleshoot egress path
- Divide forwarding path in half
- Use ACLs, Netflow etc. to match the traffic
- Move to the next hop until match is not found



Using ACLs to match the unlabeled traffic

- ACLs are one of the methods to confirm that **unlabeled** traffic is reaching the node
- With the **hardware** keyword in the command syntax we can check if traffic is matching ACL line

```
RP/0/RP0/CPU0:A-PE4#show access-lists ICMP_IN hardware ingress location 0/0/CPU0
ipv4 access-list ICMP_IN
 10 permit icmp host 192.168.2.2 any (3575352 matches)
 20 permit ipv4 any any (1261642975 matches)

RP/0/RP0/CPU0:A-PE4#sh run int Te0/0/0/22.500
interface TenGigE0/0/0/22.500
…
 ipv4 access-group ICMP_IN ingress
```

- By default on NCS5500 we provide only statistics for the dropped packets, to enable statistics on the permitted traffic, we need to configure following command (LC reload is required)

```
hw-module profile stats acl-permit
```

- **Note**: This profile cannot be used with the qos-enhanced

```
hw-module profile stats qos-enhanced
```

Using Netflow to match the traffic

- Another option which will help us to check if the traffic has reached the node is Netflow
- We don't need external collector, we only need a local cache
- With the Netflow we can capture IPv4/IPv6 and MPLS traffic
- Thanks to the MPLS Netflow, we can capture imposed label stack and ingress/egress interface:

```
RP/0/RP0/CPU0:PE1#show flow monitor monitor1 cache location 0/0/CPU0

LabelType Prefix/Length      Label1-EXP-S    Label2-EXP-S    Label3-EXP-S    Label4-EXP-S    Label5-EXP-S    Label6-EXP-S    InputInterface
      LDP 192.168.2.22/32      16009-3-0       32006-3-1       -               -               -               -               BE3

OutputInterface ForwardStatus        FirstSwitched   LastSwitched    ByteCount    PacketCount  Dir SamplerID  IPV4SrcAddr     IPV4DstAddr     IPV4TOS
BE1             Fwd                  03 23:55:41:368 03 23:55:56:223 1065674      2534         Ing 1          192.168.1.64    192.168.2.64    0x60

IPV4Prot L4SrcPort  L4DestPort L4TCPFlags   InputVRFID                      OutputVRFID
udp      63         63         0            default                         default
```

* This output has been modified to be more human readable

Interface drop counters

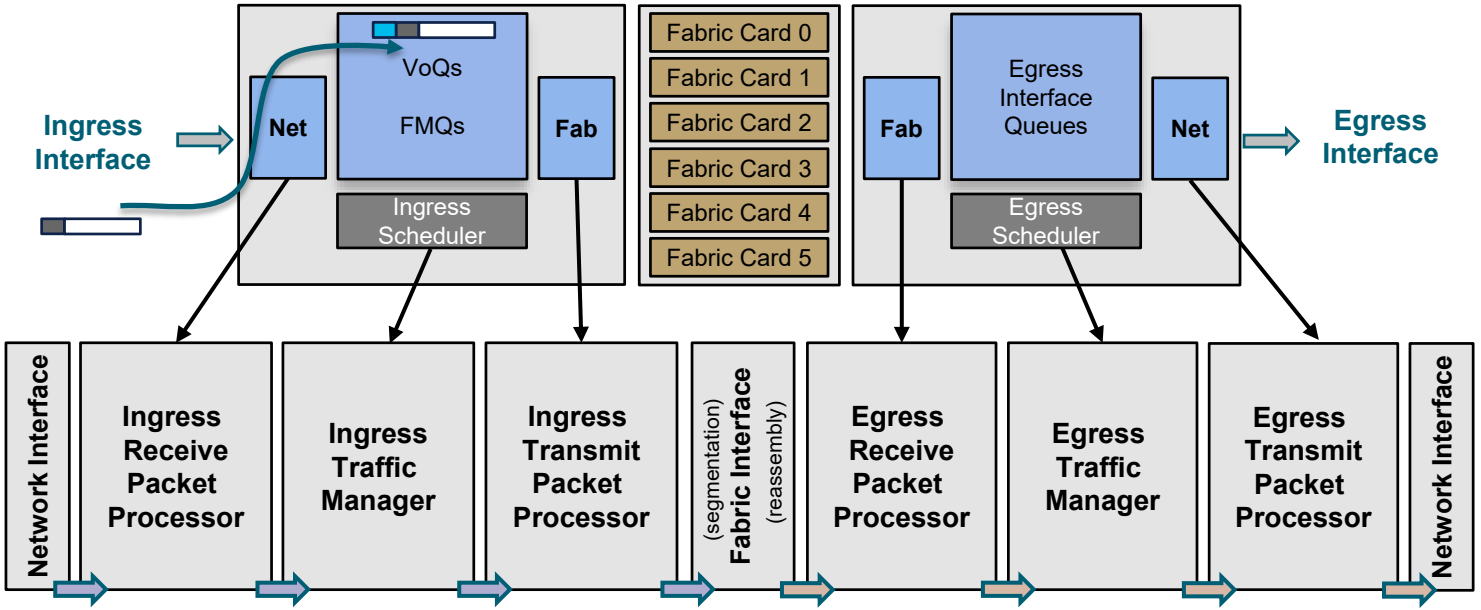
```
RP/0/RP0/CPU0:A-PE4#show int HundredGigE0/0/2/2

HundredGigE0/0/2/2 is up, line protocol is up
  Interface state transitions: 1
  Hardware is HundredGigE, address is 00bc.6047.b060 (bia 00bc.6047.b060)
  Description: PE4 Hu0/0/0/3
  Internet address is Unknown
  MTU 9216 bytes, BW 100000000 Kbit (Max: 100000000 Kbit)
     reliability 255/255, txload 12/255, rxload 7/255
  Encapsulation ARPA,
  Full-duplex, 100000Mb/s, 100GBASE-SR4, link type is force-up
  output flow control is off, input flow control is off
  Carrier delay (up) is 10 msec
  loopback not set,
  Last link flapped 3d22h
  Last input 00:00:00, output 00:00:00
  Last clearing of "show interface" counters 3d18h
  5 minute input rate 3090402000 bits/sec, 1467261 packets/sec
  5 minute output rate 4747164000 bits/sec, 1598464 packets/sec
     229081981434 packets input, 60313304464901 bytes, 0 total input drops
     0 drops for unrecognized upper-level protocol
     Received 0 broadcast packets, 6589365 multicast packets
              0 runs, 0 giants, 0 throttles, 0 parity
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     249570539558 packets output, 92644183417443 bytes, 0 total output drops
     Output 0 broadcast packets, 6589410 multicast packets
     0 output errors, 0 underruns, 0 applique, 0 resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions
```

Controller drop counters

```
RP/0/RP0/CPU0:A-PE4#show controllers HundredGigE 0/0/2/2
stats

Statistics for interface HundredGigE0/0/2/2 (cached values):

Ingress:
…

    Input drop overrun          = 0
    Input drop abort            = 0
    Input drop invalid VLAN     = 0
    Input drop invalid DMAC     = 0
    Input drop invalid encap    = 0
    Input drop other            = 0

    Input error giant           = 0
    Input error runt            = 0
    Input error jabbers         = 0
    Input error fragments       = 0
    Input error CRC             = 0
    Input error collisions      = 0
    Input error symbol          = 0
    Input error other           = 0

    Input MIB giant             = 0
    Input MIB jabber            = 0
    Input MIB CRC               = 0
```

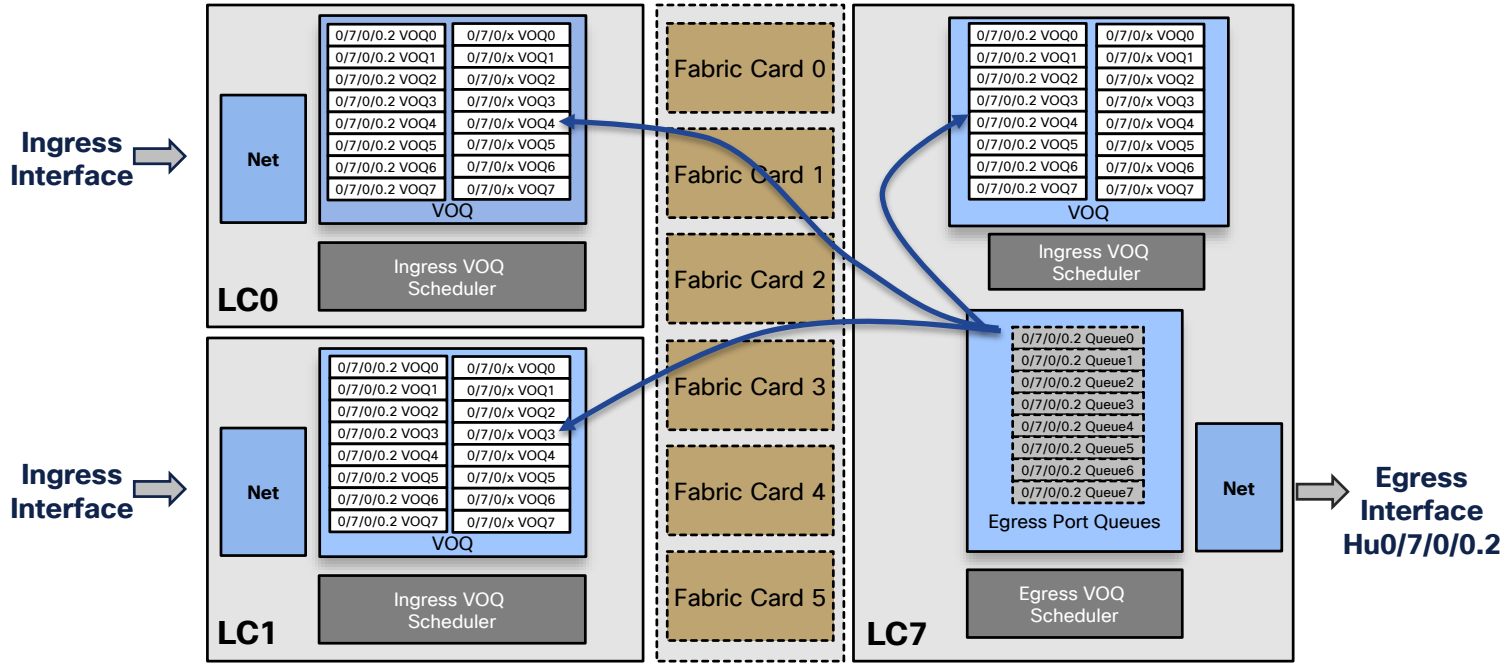
```
Egress:
    Output total bytes          = 92730300788456
    Output good bytes           = 92730300788456

    Output total packets        = 249802525333
    Output 802.1Q frames        = 0
    Output pause frames         = 0
    Output pkts 64 bytes        = 32766
    Output pkts 65-127 bytes    = 145720161036
    Output pkts 128-255 bytes   = 73005
    Output pkts 256-511 bytes   = 5
    Output pkts 512-1023 bytes  = 83265928020
    Output pkts 1024-1518 bytes = 0
    Output pkts 1519-Max bytes  = 20816293892

    Output good pkts            = 249802525333
    Output unicast pkts         = 249795933636
    Output multicast pkts       = 6592331
    Output broadcast pkts       = 0

    Output drop underrun        = 0
    Output drop abort           = 0
    Output drop other           = 0

    Output error other          = 0
```

Reminder on VOQ-only Architecture



1 lookup → VoQ Sch poll → VoQ Sch grant → spray cells → reassemble packet → transmit

Virtual Output Queues (VOQ) – reminder



More details:
BRKSPG-2900 Deepdive in the Merchant Silicon High-end SP Routers

Virtual Output Queues (VOQ) mapping

- CLI illustration: Local and Remote visibility of the Output Queues

```
RP/0/RP0/CPU0:ios#sh contr npu voq-usage interface all instance 0 location 0/0/CPU0

--------------------------------------------------------------------
Node ID: 0/0/CPU0
Intf          Intf      NPU NPU   PP    Sys    VOQ    Flow    VOQ     Port
name          handle     #  core Port  Port   base   base    port    speed
              (hex)                                      type    (Gbps)
--------------------------------------------------------------------------
Hu0/3/0/5     1800100    0   0     1   1537   1072   10280 remote  100
Hu0/0/0/26    200        4   1    17    273   1424    4136 local   100
Hu0/3/0/6     1800108    1   1    21   1621   1080    1064 remote  100
Hu0/0/0/27    208        4   0     9    265   1432    5416 local   100
Hu0/3/0/7     1800110    1   1    13   1613   1088    2344 remote  100
Hu0/0/0/28    210        4   0     5    261   1440    7208 local   100
Hu0/3/0/8     1800118    1   1    17   1617   1096    4136 remote  100
Hu0/0/0/29    218        4   0     1    257   1448    8488 local   100
Hu0/3/0/9     1800120    1   0     9   1609   1104    5416 remote  100
Hu0/0/0/30    220        5   1    21    341   1456    2344 local   100
```

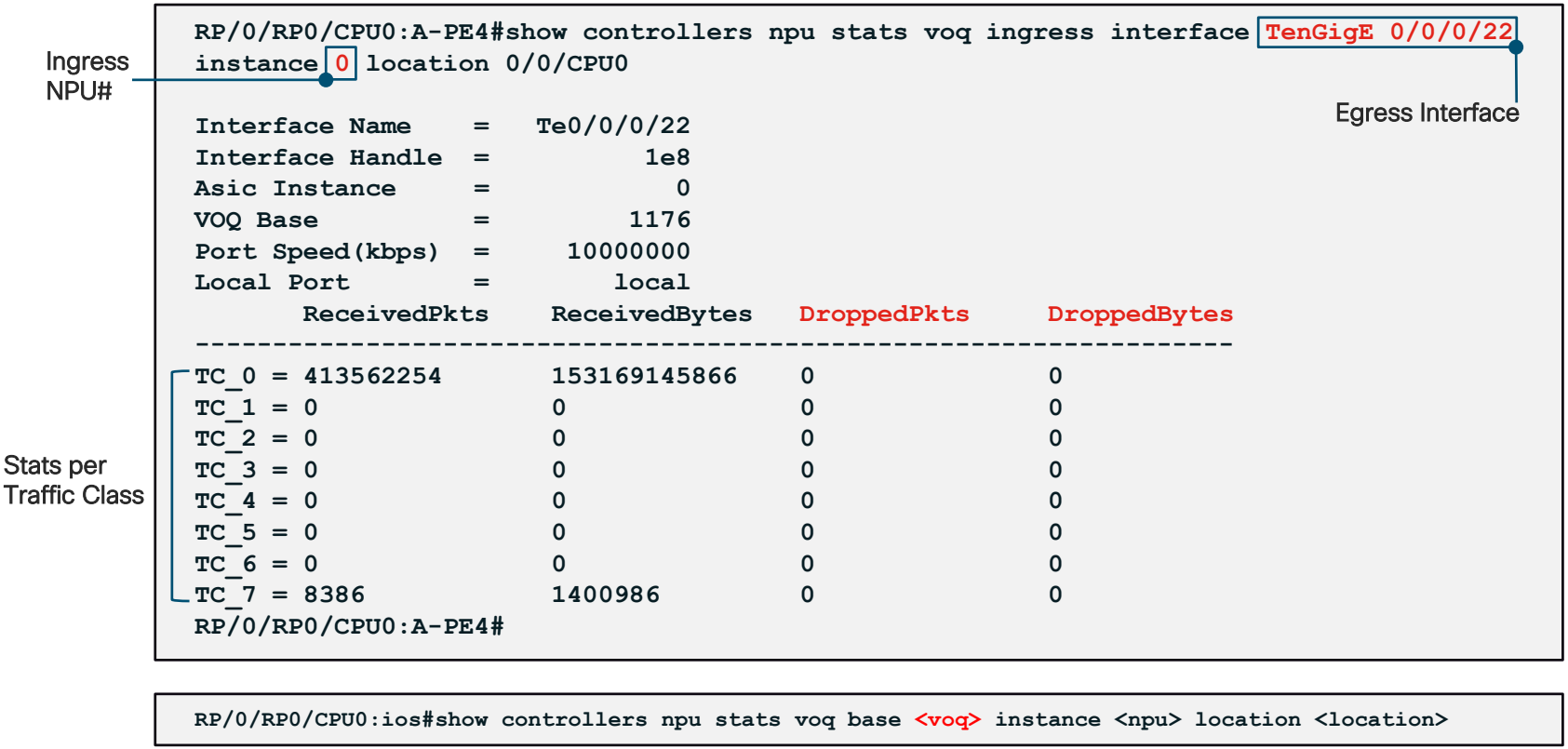
LC 0/0 point of view

VOQ number

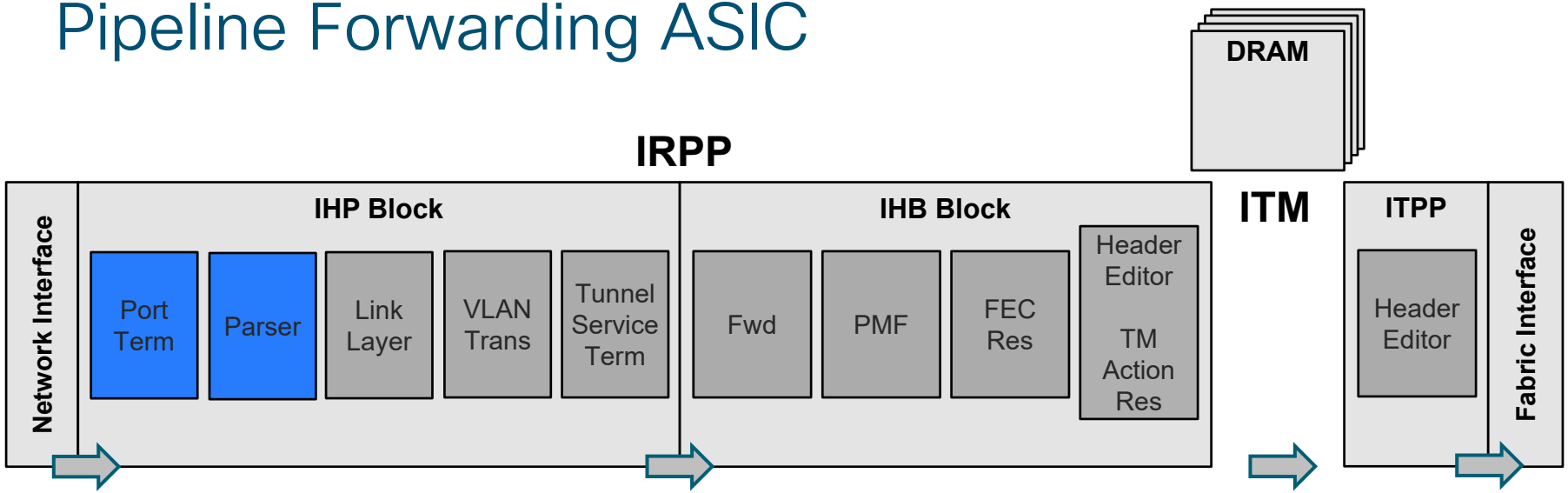
Local for LC 0/0

Remote for LC 0/0

VOQ Stats



Pipeline Forwarding ASIC



- Port Termination: packet received from network interface / CPU / Recirculation
- Parser: extract ethertype, MAC addresses, determine offset (where network header starts) for next stages in the pipeline

Port Mapping

- Mapping Physical ports to NPU, NPU core, PP port and source system port
  - PP port: Port Termination ID
  - Flow base ID: connector ID linking VOQ to egress scheduling elements
  - System port ID: source port (used in egress)

```
RP/0/RP0/CPU0:A-PE4#show controllers npu voq-usage interface TenGigE 0/0/0/22 instance
all location 0/0/CPU0

------------------------------------------------------------------
Node ID: 0/0/CPU0
Intf          Intf      NPU  NPU   PP    Sys   VOQ   Flow   VOQ     Port
name          handle    #    core  Port  Port  base  base   port    speed
              (hex)                                        type
------------------------------------------------------------------------------
Te0/0/0/22    1e8       0    1     48     48   1176   5456 local     10G
```

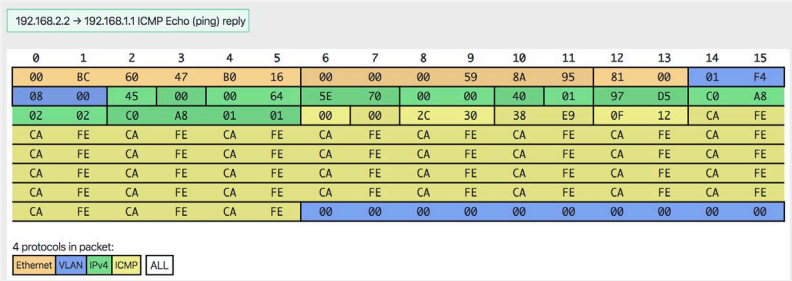
NCS5500 Last packet dump

NPU number

LC

```
RP/0/RP0/CPU0:ios#show controller npu diag last instance 0 location 0/0/CPU0

Core 0:
Last packet information: is_valid=1  tm_port=17
pp_port=17  src_syst_port=49152  port_header_type=eth packet_size=118
Packet start, offset in bytes:
00: 00bc6047 b0dc008a 96cc50da 884707d1 41fb4500 00640f11 0000ff01 2834c0a8
20: 0101c0a8 02020800 243138e9 0f11cafe cafecafe cafecafe cafecafe cafecafe
40: cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe
60: cafecafe cafecafe cafecafe cafecafe cafecafe cafe0000 00000000 00000000
Core 1:
Last packet information: is_valid=1  tm_port=48
pp_port=48  src_syst_port=48  port_header_type=eth packet_size=118
Packet start, offset in bytes:
00: 00bc6047 b0160000 00598a95 810001f4 08004500 00645e70 00004001 97d5c0a8
20: 0202c0a8 01010000 2c3038e9 0f12cafe cafecafe cafecafe cafecafe cafecafe
40: cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe
60: cafecafe cafecafe cafecafe cafecafe cafecafe cafe0000 00000000 00000000
:
RP/0/RP0/CPU0:ios#
```

Decoded packet



pp_port will help to identify ingress interface under `show controllers npu voq-usage interface` command output.
Make sure that to check on the right NPU # & NPU core.

NCS5500 Last packet dump – hack

- Generate ICMP traffic with the pattern on the remote node

```
RP/0/RP0/CPU0:remote#ping 192.168.0.1 repeat 1000000 pattern cafe
```

- Use describe command to get shell syntax for the npu diag last with the pattern

```
RP/0/RP0/CPU0:ios#describe show controllers npu  diag last instance 1 location 0/0/CPU0 | inc cafe
The command is defined in parsercmds.parser


User needs ALL of the following taskids:

              sonet-sdh (READ) or dwdm (READ) or interface (READ)          cisco-support (READ)

It will take the following actions:
Mon Dec 16 14:02:51.067 CET
  Spawn the process:
            ofa_npu_diag_stats_show -v 0xf -i 0x1 -n 0 | grep -E cafe
```

- Go to the shell and use a while loop to catch the packet

```
RP/0/RP0/CPU0:iso#run
Mon Dec 16 14:02:57.848 CET
[xr-vm_node0_RP0_CPU0:~]$while :; do ofa_npu_diag_stats_show -v 0xf -i 0x1 -n 0 | grep -E cafe; done
20: 00010800 f6bb69bb 0bb4cafe cafecafe cafecafe cafecafe cafecafe cafecafe
40: cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe cafecafe
60: cafecafe cafecafe cafecafe cafecafe cafe0000 00000000 00000000 00000000
40: 00ea7269 bb1ffdca fecafeca fecafeca fecafeca fecafeca fecafeca fecafeca
```

Parsing Info for the last packet

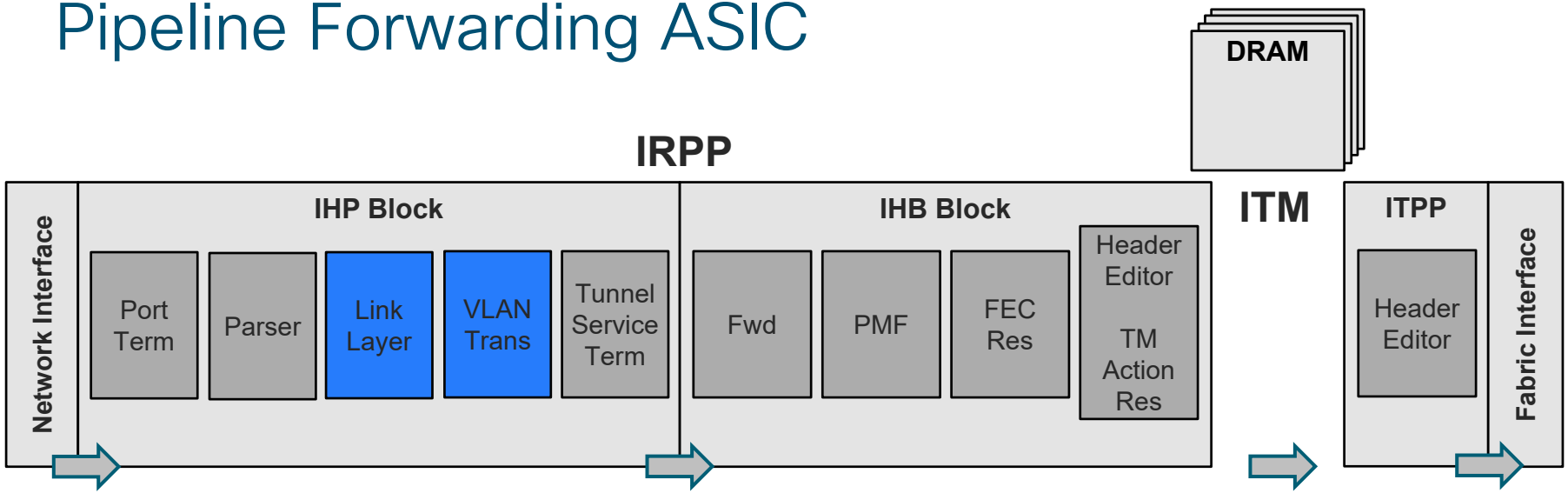
```
RP/0/RP0/CPU0:A-PE4#show controllers npu diag pp ParsingInfo
instance all loc 0/0/CPU0

Core 0:
hdr_type IPv4oMPLSx1oETH

 header[0]:
 hdr_type eth hdr_offset: 0 bytes
 encap_type eth2 tag_fromat:outer_tpid none inner_tpid none
is_outer_prio: 0
 vlan_tags:
 vlan_tag[0]:tpid: 00000000  vid: 0
 vlan_tag[1]:tpid: 00000000  vid: 0
 vlan_tag_format: 0 (none)
 next_prtcl mpls

 header[1]:
 hdr_type mpls hdr_offset: 14 bytes
 bos: 1

 header[2]:
 hdr_type ipv4 hdr_offset: 18 bytes
 next_prtcl icmp is_mc: 0
 is_fragmented: 0
 hdr_err: 0

 initial_vid 4095
```

Incoming ICMP traffic from the MPLS cloud

```
 Core 1:
hdr_type IPv4oETH

 header[0]:
 hdr_type eth hdr_offset: 0 bytes
 encap_type eth2 tag_fromat:outer_tpid tpid2 inner_tpid none
is_outer_prio: 0
 vlan_tags:
 vlan_tag[0]:tpid: 00008100  vid: 500
 vlan_tag[1]:tpid: 00000000  vid: 0
 vlan_tag_format: 8 ( Unknown)
 next_prtcl ipv4

 header[1]:
 hdr_type ipv4 hdr_offset: 18 bytes
 next_prtcl icmp is_mc: 0
 is_fragmented: 0
 hdr_err: 0

 initial_vid 500
 :
```

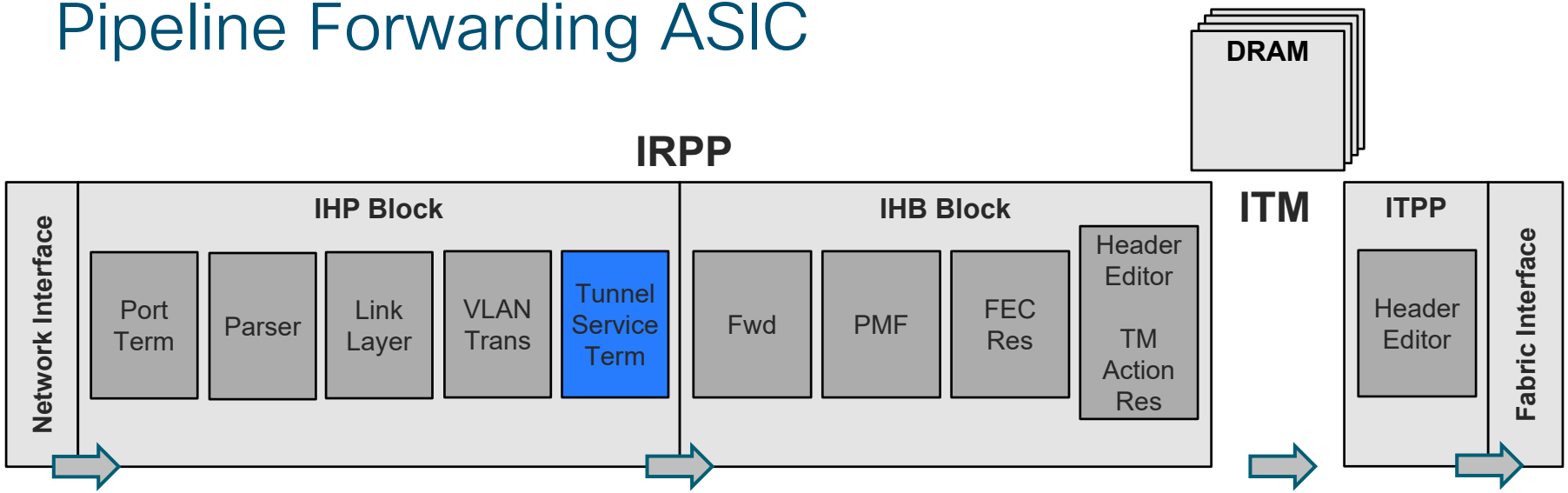
ICMP packet from vlan 500

Pipeline Forwarding ASIC



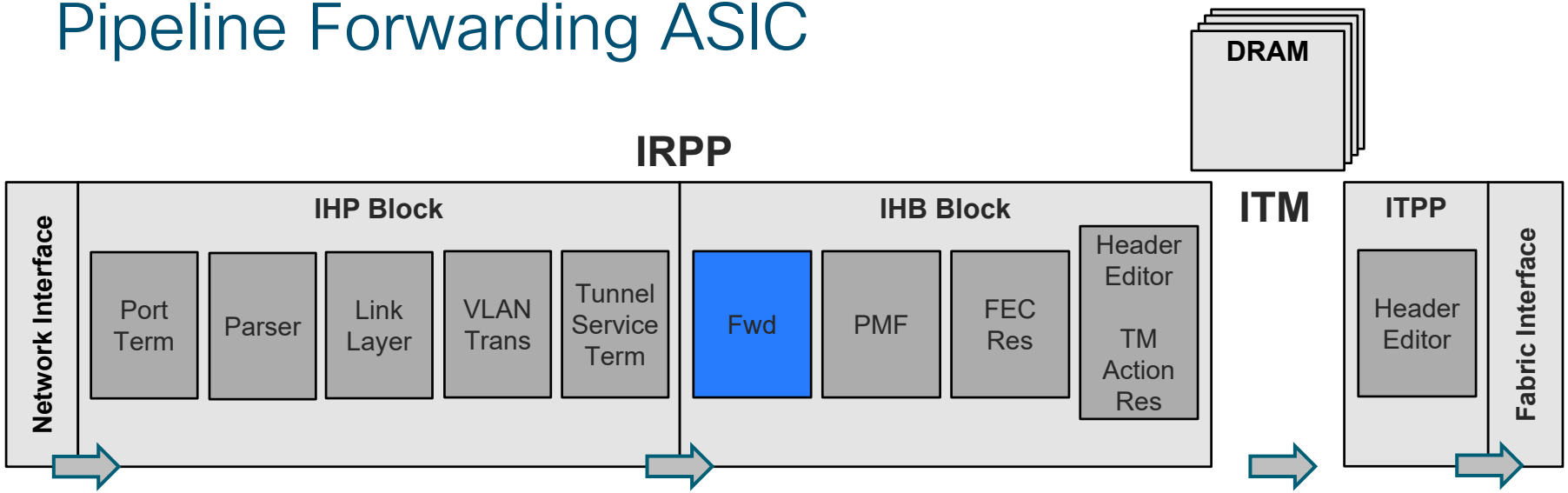
- Link-layer: **filtering on L2** (ARP, …), source address authentication, **Mirroring** and **Netflow**
- VLAN translation: mapping of the logical interface of the packet

Pipeline Forwarding ASIC



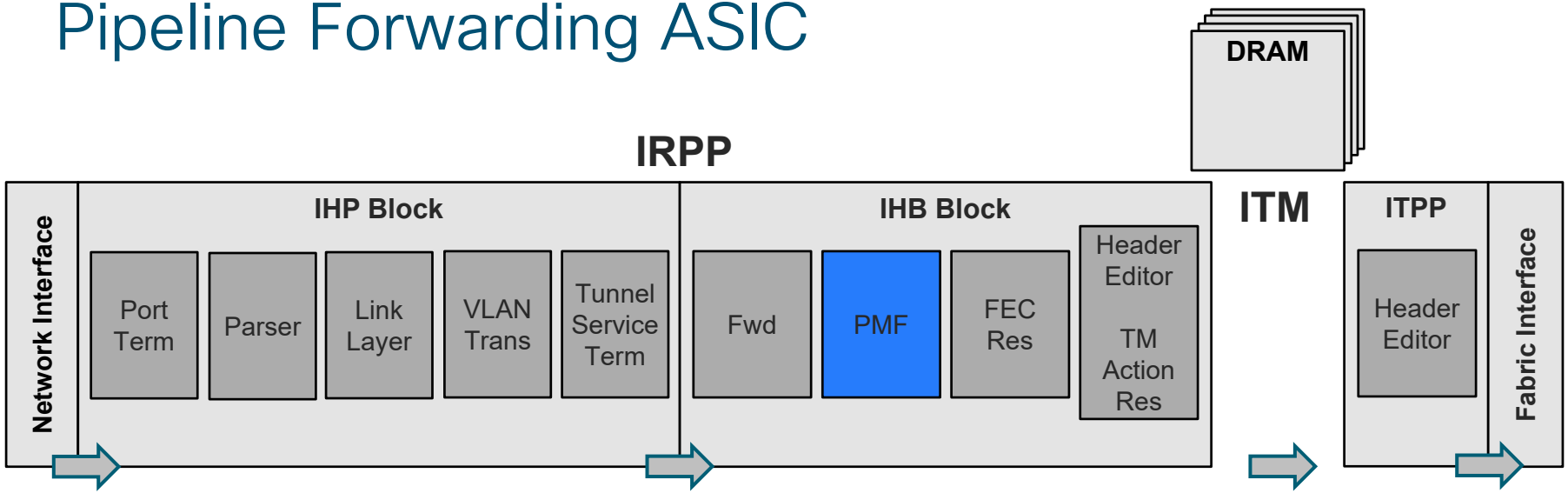
- Tunnel Service Termination: **tunnels termination (GRE/MPLS/...)**
  - If the destination header match to my@MAC, we terminate and try to do routing
  - In output, it will decide what is my forwarding header (am I going to do bridging, routing, MPLS, IP etc)

Pipeline Forwarding ASIC



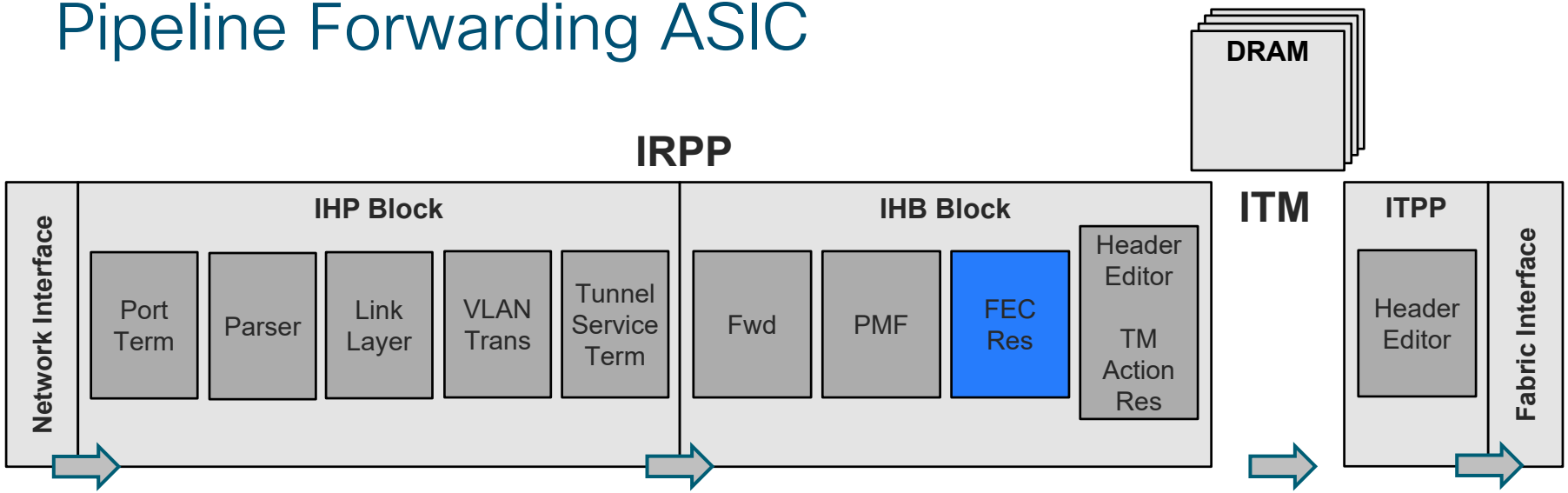
- Forwarding lookup
  - Depending on the forwarding header different lookup action (using different database: LEM, TCAM, KAPS, ...)
  - If external memory is present, ROP (record over packet) is generated and sent to the external device.
  - OAM classification
  - The result will be a destination and editing information

Pipeline Forwarding ASIC



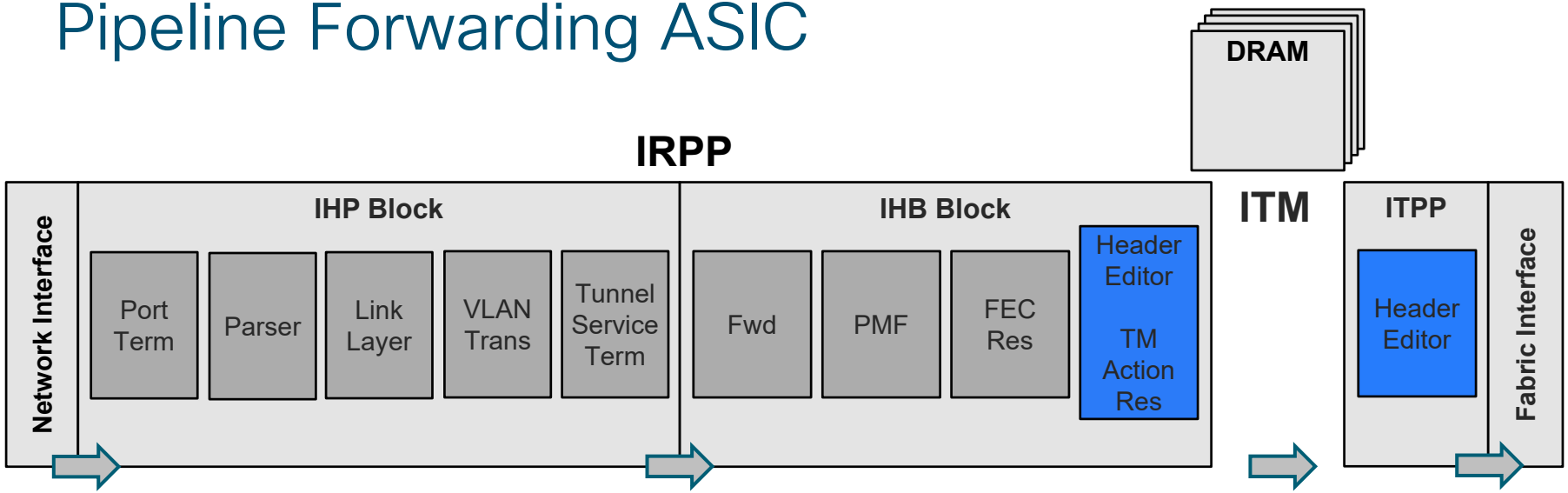
- PMF is where we **apply ACLs**
  - It has all the history of the packet from other blocks (incoming port, lookup results, etc)
  - We can override here every decision taken along the pipe
  - Here we do **ACL, QOS, LPTS**, etc classifications and set actions (counters, policers, TC).

Pipeline Forwarding ASIC



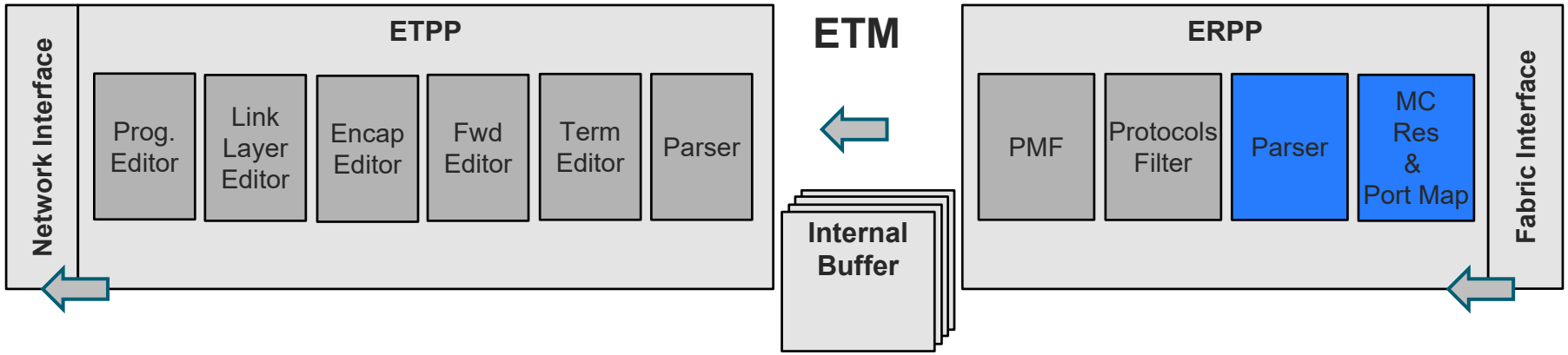
- FEC Resolution: final destination of the packet is calculated here
  - Port protection
  - ECMP
  - RPF check
  - LAG resolution

Pipeline Forwarding ASIC



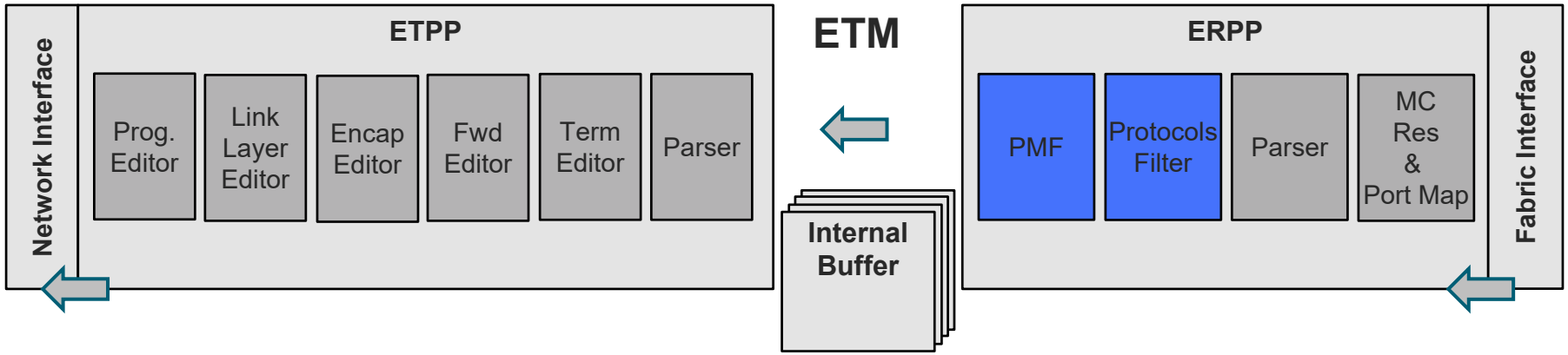
- Header Editor is the last block IRPP/IHB
  - Where we build the system header and build the TM command
- ITPP Header Editor

Pipeline Forwarding ASIC



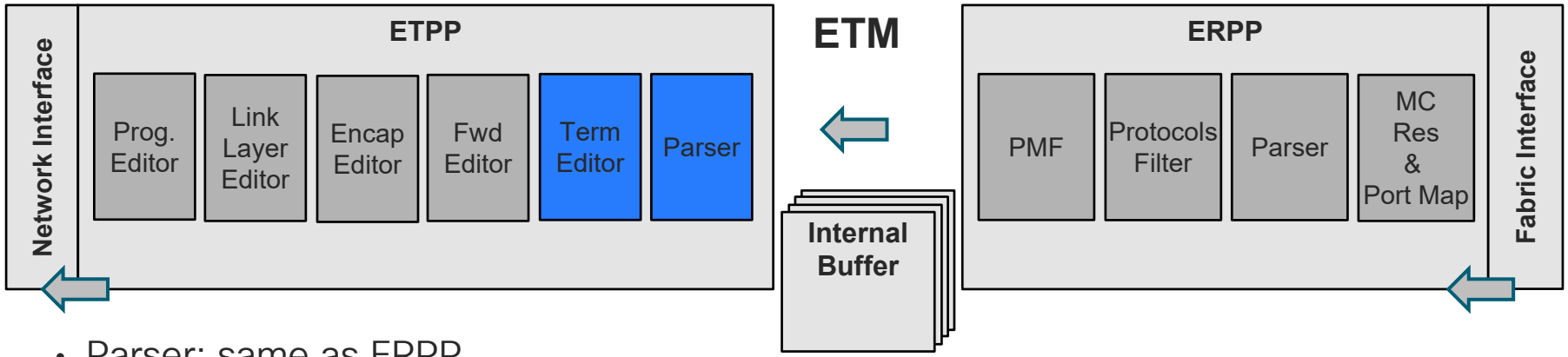
- Basic parsing operation on the forwarding header: Ethernet, IP, MPLS, …
- System headers from the ingress side (FTMH, PPH, …) are available for processing

Pipeline Forwarding ASIC



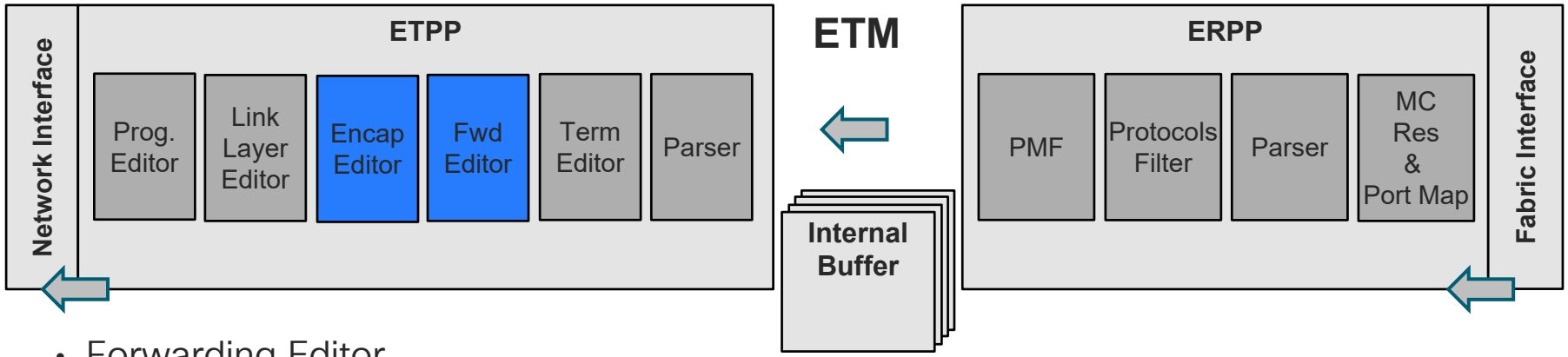
- Egress PMF capable of doing internal TCAM lookup for **egress ACL**

Pipeline Forwarding ASIC



- Parser: same as ERPP
- Termination Editor, where we terminated any header we want
  - remove system headers
  - remove link layer header, or any additional tunnel header.
  - If we terminate the link layer header, it applies the egress VLAN editing command
  - egress OAM

Pipeline Forwarding ASIC



- Forwarding Editor
  - where we update the forwarding header. Remark TTL, TOS, EXP etc
  - In case of MPLS, we can encapsulate/manipulate (swap) one MPLS header
- Encapsulation Editor
  - where we build tunnels (GRE, MPLS up to 2 labels) if needed (based on encapsulation database info)

Encapsulation Info for the last packet

```
RP/0/RP0/CPU0:A-PE4# show controllers npu diag pp
EncapsulationInfo instance all location 0/0/CPU0

Core 0:
eep[0]: 49170
 encap_info[0]:MPLS Encapsulation:
 tunnel 1:
 tunnel_label: 32010
 push_profile: 3
 tunnel 2:
 tunnel_label: 16007
 push_profile: 3
 nof_tunnels: 2
 orientation NA
 out_vsi: 0
 oam_lif_set: 0
 outlif_profile: 0x11
```

MPLS encapsulation with the labels

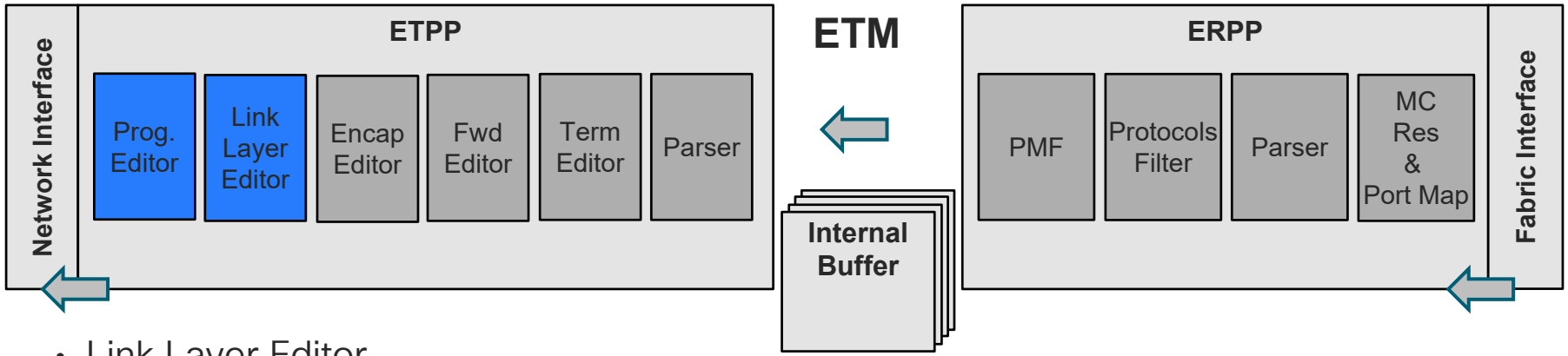
```
 eep[1]: 24582
 encap_info[1]:LL Encapsulation:
 dest_mac:00:8a:96:cc:50:da
 out_vid_valid: 1
 out_vid: 1598
 pcp_dei_valid: 0
 pcp_dei: 0
 tpid_index: 0
 ll_remark_profile: 0
 out_ac_valid: 0
 out_ac_lsb: 0
 oam_lif_set: 0
 outlif_profile: 0x10
 is native: 0
```

ETH encapsulation with the NH mac address

```
 ll_vsi: 1598
 out_ac: 8192
 pp_port: 17
 tm_port: 17
 OutLIF Profile
 ==============
        Bit 0 : Split Horizon Orientation
        Bit 1 : Split Horizon Orientation
        Bit 2 : Unreserved
        Bit 3 : Mpls encapsulation extended label
        Bit 4 : DSCP remark preserve
        Bit 5 : EVPN
```

```
RP/0/RP0/CPU0:A-PE4#show controllers npu voq-usage interface all
instance all location 0/0/CPU0

--------------------------------------------------------------------
Node ID: 0/0/CPU0
Intf          Intf      NPU NPU  PP    Sys   VOQ   Flow   VOQ     Port
name          handle     #  core Port  Port  base  base   port    speed
              (hex)                                        type
--------------------------------------------------------------------
<...>
Hu0/0/2/2     388        0   0    17    17    1456   5584 local   100G
<...>
```

Pipeline Forwarding ASIC



- Link Layer Editor
  - out AC for bridging or Ethernet header re-write if the packet is routed
  - egress VLAN editing on the packet
  - Link Layer filtering
- Programmable Editor
  - allows to reconstruct or change any header and offers a bit more flexibility than the other configurable blocks of the pipe

NCS5500 Forwarding ASIC Statistics

```
RP/0/RP0/CPU0:ios#sh controllers npu stats counters-all instance 0 location 0/0/CPU0

FIA Statistics Rack: 0, Slot: 0, Asic instance: 0

Per Block Statistics:

Ingress:

NBI RX:
   RX_TOTAL_BYTE_COUNTER          = 0
   RX_TOTAL_PKT_COUNTER           = 0

IRE:
   CPU_PACKET_COUNTER             = 0
   NIF_PACKET_COUNTER             = 0
   OAMP_PACKET_COUNTER            = 47
   OLP_PACKET_COUNTER             = 0
   RCY_PACKET_COUNTER             = 0
   IRE_FDT_INTRFACE_CNT           = 128

IDR:
   MMU_IDR_PACKET_COUNTER         = 32
   IDR_OCB_PACKET_COUNTER         = 1

IQM:
   ENQUEUE_PKT_CNT                = 47
   DEQUEUE_PKT_CNT                = 47
   DELETED_PKT_CNT                = 0
   ENQ_DISCARDED_PACKET_COUNTER   = 0

IPT:
   EGQ_PKT_CNT                    = 47
   ENQ_PKT_CNT                    = 47
   FDT_PKT_CNT                    = 0
   CFG_EVENT_CNT                  = 47
   CFG_BYTE_CNT                   = 3611

FDT:
   IPT_DESC_CELL_COUNTER          = 0
   IRE_DESC_CELL_COUNTER          = 0
   TRANSMITTED_DATA_CELLS_COUNTER = 0

Egress:
```

Network Int. counters

Network buffer and Interlaken block
It manages the RX buffers for the interface.

IRPP counters

Ingress Receive Editor
Receives the packet segments from the packet interface.

Ingress DRAM Assembly
Reassembles the data segments into full packets to be sent to DRAM or to OCB.

ITM counters

Ingress Queue Manager
Handles en-queue and de-queue commands from IRR (Ingress Replicator) and IPS (Ingress Packet Scheduler).

ITPP counters

Ingress Packet Transmit
Receives packet context from IQM.

Fabric Int. counters

Fabric Data transmit
Receives data packets from IPT and TDM packets from IRE.

* Drop counters in red.

NCS5500 Forwarding ASIC Statistics

For Reference

```
FDR:
   P1_CELL_IN_CNT                = 0
   P2_CELL_IN_CNT                = 0
   P3_CELL_IN_CNT                = 0
   CELL_IN_CNT_TOTAL             = 0

FDA:
   CELLS_IN_CNT_P1               = 0
   CELLS_IN_CNT_P2               = 0
   CELLS_IN_CNT_P3               = 0
   CELLS_IN_TDM_CNT              = 0
   CELLS_IN_MESHMC_CNT           = 0
   CELLS_IN_IPT_CNT              = 47
   CELLS_OUT_CNT_P1              = 0
   CELLS_OUT_CNT_P2              = 0
   CELLS_OUT_CNT_P3              = 0
   CELLS_OUT_TDM_CNT             = 0
   CELLS_OUT_MESHMC_CNT          = 0
   CELLS_OUT_IPT_CNT             = 47
   EGQ_DROP_CNT                  = 0
   EGQ_MESHMC_DROP_CNT           = 0
```

Fabric Int. counters

Fabric Data Receive

Receives data packets from IPT and FDT and data cells from fabric. Maps cells to one of the cores of EGQ (egress queue).

```
EGQ:
   FQP_PACKET_COUNTER            = 47
   PQP_UNICAST_PKT_CNT           = 47
   PQP_DSCRD_UC_PKT_CNT          = 0
   PQP_UC_BYTES_CNT              = 3611
   PQP_MC_PKT_CNT                = 0
   PQP_DSCRD_MC_PKT_CNT          = 0
   PQP_MC_BYTES_CNT              = 0
   EHP_UNICAST_PKT_CNT           = 47
   EHP_MC_HIGH_PKT_CNT           = 0
   EHP_MC_LOW_PKT_CNT            = 0
   DELETED_PKT_CNT               = 0
   EHP_MC_HIGH_DSCRD_CNT         = 0
   EHP_MC_LOW_DSCRD_CNT          = 0
   ERPP_LAG_PRUNING_DSCRD_CNT    = 0
   ERPP_PMF_DISCARDS_CNT         = 0
   ERPP_VLAN_MBR_DSCRD_CNT       = 0
```

ERPP / ETM counters

Egress Queue

Includes the ERPP and ETM logic (packet reassembly, egress acl's, scheduling, etc).

```
EPNI:
   EPE_BYTES_COUNTER             = 4363
   EPE_PKT_COUNTER               = 47
   EPE_DSCRD_PKT_CNT             = 0
```

ETPP counters

Egress process network interface

Manages all packet editing.

```
NBI TX:
   TX_TOTAL_BYTE_COUNTER         = 0
   TX_TOTAL_PKT_COUNTER          = 0
```

Network Int. counters

Network buffer and Interlaken block

It manages the TX buffers for the interface.

* Drop counters in red.

For Reference

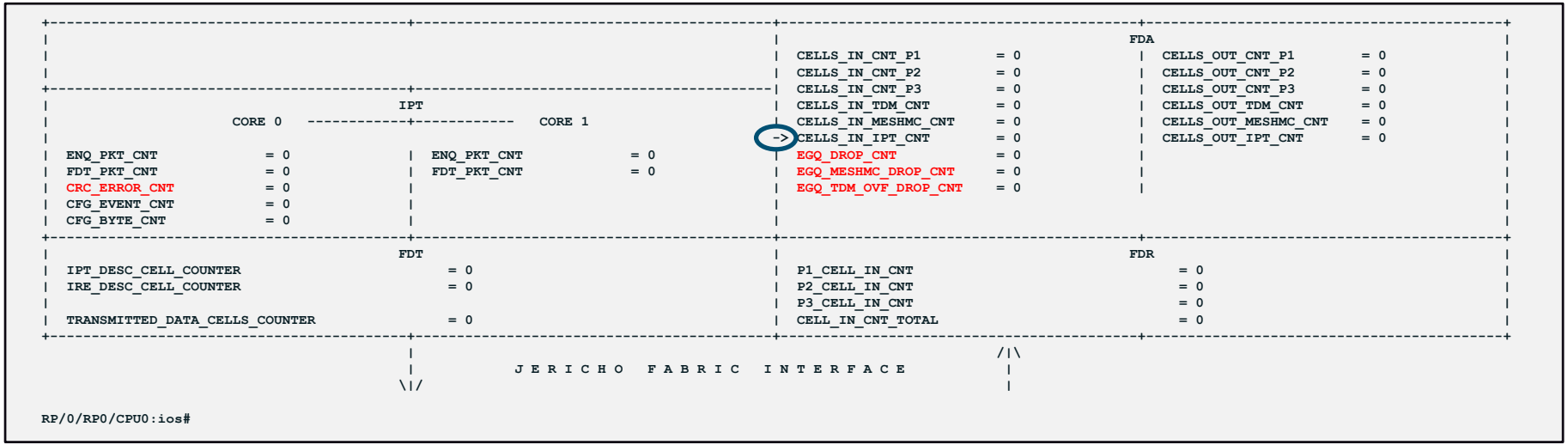
NCS5500 Forwarding ASIC Counters

```
RP/0/RP0/CPU0:ios# show controllers np diag counters graphical cdsp instance 0 location 0/0/CPU0
                                            |                                                         /|\
                                            |         J E R I C H O   N E T W O R K   I N T E R F A C E |
                                           \|/                                                          |
+-------------------------------------------+-------------------------------------------+-------------------------------------------+
|                                                                   NBI                                                            |
|  RX_TOTAL_BYTE_COUNTER                         = 0                |  TX_TOTAL_BYTE_COUNTER                         = 0           |
|  RX_TOTAL_PKT_COUNTER                          = 0                |  TX_TOTAL_PKT_COUNTER                          = 0           |
|  RX_TOTAL_DROPPED_EOPS                         = 0                |                                                              |
+-------------------------------------------+-------------------------------------------+-------------------------------------------+
|                                          IRE                      |                                          EPNI                 |
|  CPU_PACKET_COUNTER      = 0              |                       |                                                              |
|  NIF_PACKET_COUNTER_0    = 0              |  NIF_PACKET_COUNTER_1    = 0  |  EPE_BYTES_COUNTER     = 0  |  EPE_BYTES_COUNTER     = 0  |
|  OAMP_PACKET_COUNTER     = 0              |                       |  EPE_PKT_COUNTER       = 0  |  EPE_PKT_COUNTER       = 0      |
|  OLP_PACKET_COUNTER      = 0              |                       |  EPE_DSCRD_PKT_CNT     = 0  |  EPE_DSCRD_PKT_CNT     = 0      |
|  RCY_PACKET_CNT_0_0      = 0              |  RCY_PACKET_CNT_1_0      = 0  |                                                      |
|  RCY_PACKET_CNT_0_1      = 0              |  RCY_PACKET_CNT_1_1      = 0  |                                                      |
|  IRE_FDT_INTRFACE_CNT    = 0              |                       |                                                              |
+-------------------------------------------+-------------------------------------------+-------------------------------------------+
|                                          IDR                      |                                          EGQ                  |
|                                                                   |                  CORE 0   ------------+-----------   CORE 1 |
|  MMU_IDR_PACKET_COUNTER                        = 0                |  FQP_PACKET_COUNTER      = 0  |  FQP_PACKET_COUNTER      = 0  |
|  IDR_OCB_INTERFACE_COUNTER                     = 0                |  PQP_UNICAST_PKT_CNT     = 0  |  PQP_UNICAST_PKT_CNT     = 0  |
|                                                                   |  PQP_DSCRD_UC_PKT_CNT    = 0  |  PQP_DSCRD_UC_PKT_CNT    = 0  |
|                                                                   |  PQP_UC_BYTES_CNT        = 0  |  PQP_UC_BYTES_CNT        = 0  |
+-------------------------------------------+-----------------------|  PQP_MC_PKT_CNT          = 0  |  PQP_MC_PKT_CNT          = 0  |
|                                          IQM                      |  PQP_DSCRD_MC_PKT_CNT    = 0  |  PQP_DSCRD_MC_PKT_CNT    = 0  |
|                     CORE 0   ------------+-----------   CORE 1    |  PQP_MC_BYTES_CNT        = 0  |  PQP_MC_BYTES_CNT        = 0  |
|  ENQUEUE_PKT_CNT         = 0              |  ENQUEUE_PKT_CNT         = 0  |  EHP_UNICAST_PKT_CNT     = 0  |  EHP_UNICAST_PKT_CNT     = 0  |
|  DEQUEUE_PKT_CNT         = 0              |  DEQUEUE_PKT_CNT         = 0  |  EHP_MC_HI_PKT_CNT       = 0  |  EHP_MC_HI_PKT_CNT       = 0  |
|  DELETED_PKT_CNT         = 0              |  DELETED_PKT_CNT         = 0  |  EHP_MC_LOW_PKT_CNT      = 0  |  EHP_MC_LOW_PKT_CNT      = 0  |
|  ENQ_DSCRD_PKT_CNT       = 0              |  ENQ_DSCRD_PKT_CNT       = 0  |  DELETED_PKT_CNT         = 0  |  DELETED_PKT_CNT         = 0  |
|                                                                   |  RQP_PKT_CNT             = 47 |  RQP_PKT_CNT             = 0  |
|                                                                   |  RQP_DSCRD_PKT_CNT       = 0  |  RQP_DSCRD_PKT_CNT       = 0  |
|                                                                   |  PRP_PKT_DSCRD_TDM_CNT   = 0  |  PRP_PKT_DSCRD_TDM_CNT   = 0  |
|                                                                   |  PRP_SOP_DSCRD_UC_CNT    = 0  |  PRP_SOP_DSCRD_UC_CNT    = 0  |
|                                                                   |  PRP_SOP_DSCRD_MC_CNT    = 0  |  PRP_SOP_DSCRD_MC_CNT    = 0  |
|                                                                   |  PRP_SOP_DSCRD_TDM_CNT   = 0  |  PRP_SOP_DSCRD_TDM_CNT   = 0  |
|                                                                   |  EHP_MC_HIGH_DSCRD_CNT   = 0  |  EHP_MC_HIGH_DSCRD_CNT   = 0  |
|                                                                   |  EHP_MC_LOW_DSCRD_CNT    = 0  |  EHP_MC_LOW_DSCRD_CNT    = 0  |
|                                                                   |  EHP_MC_LOW_DSCRD_CNT    = 0  |  EHP_MC_LOW_DSCRD_CNT    = 0  |
|                                                                   |  ERPP_PMF_DSCRD_CNT      = 0  |  ERPP_PMF_DSCRD_CNT      = 0  |
|                                                                   |  ERPP_VLAN_MBR_DSCRD_CNT = 0  |  ERPP_VLAN_MBR_DSCRD_CNT = 0  |
+-------------------------------------------+-------------------------------------------+-------------------------------------------+
```

* Drop counters in red.

For Reference

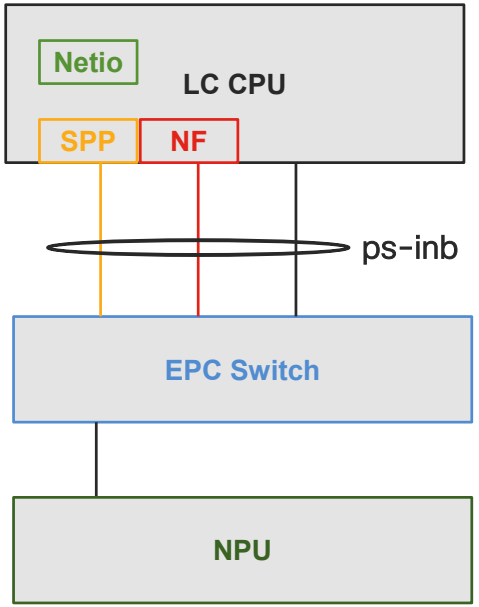
NCS5500 Forwarding ASIC Counters



* Drop counters in red.

Special Packet Handling

- Punt to the CPU
  - Routing protocols, SSH, SNMP etc. (RP CPU)
  - E-OAM, BFD, ICMP, Netflow etc. (LC CPU)
- Punted packets are handled by LPTS in 2 ways
  - Per FlowType
  - Per TRAP
- Fragmentation
  - IPv4 packets requiring fragmentation are punted to the LC CPU via SPP
  - MPLS packets requiring fragmentation are dropped
  - IPv6 doesn't support router fragmentation (per standard)



Special Packet Handling

Traps - Non-zero counters example

- Traps are used to punt the traffic to the LC CPU over LPTS

```
RP/0/RP0/CPU0:PE4#show controller npu stats traps-all instance 0 loc 0/0/CPU0 | exc "0                    0"


Trap Type                                 NPU   Trap TrapStats   Policer Packet              Packet
                                          ID     ID      ID                Accepted           Dropped
=================================================================================================================
RxTrapMimTeSaMove(SPIO)                   0     4     0x4         32041   223224             0
RxTrapAuthSaVlanFail (L3 unknown-MC/BC)   0     10    0xa         32020   1266022            8
RxTrapArp                                 0     15    0xf         32001   10921              0
RxTrapL2Cache_CDP                         0     31    0x1f        32002   190383             0
RxTrapArpReply                            0     40    0x28        32001   669                0
RxTrapFibDrop                             0     43    0x2b        32020   568                60597
RxTrapMTU                                 0     44    0x2c        32022   817528             15218165
RxTrapMiscDrop                            0     45    0x2d        32020   736                8248
RxTrapMplsTtl0                            0     139   0x8b        32014   6                  0
RxTrapMplsTtl1                            0     140   0x8c        32014   15                 0
RxTrapReceive                             0     167   0xa7        32019   89560189           0
RxTrapUserDefine_FIB_IPV4_NULL0           0     168   0xa8        32020   99                 2
RxTrapUserDefine_SR_Unknown_Label         0     169   0xa9        32020   814                12966725
RxTrapUserDefine_FIB_IPV4_GLEAN           0     171   0xab        32018   83                 215851
RxTrapUserDefine_RECEIVE_L2               0     178   0xb2        32019   1022181            0
RxTrapUserDefine_BFD                      0     183   0xb7        32030   15                 0
```

Special Packet Handling
Fragmentation example

- Fragment created in the router
  - MTU fragmentation is handled as trap and stats can be verified using

```
RP/0/RP0/CPU0:PE4#show controller npu stats traps-all instance 0 loc 0/0/CPU0
Sat Dec 28 20:27:21.092 CET


Trap Type                                NPU  Trap TrapStats   Policer Packet              Packet
                                         ID    ID      ID              Accepted            Dropped
=============================================================================================================
...
RxTrapMTU                                0    44    0x2c        32022   817528              15218165
...
```

Policer id

  - Default policer rate can be verified with the following command

```
RP/0/RP0/CPU0:PE4#attach location 0/0/CPU0
#dpa_show puntpolicer
            Def CIR Rate  Conf CIR Rate  CIR Burst         ID
...
 22 -  0:          1000              0        100      32022
       1:          1000              0        100      32022
```

Default CIR rate

Policer id

Special Packet Handling
Fragmentation example

- Fragments received on the router
  - Handled with LPTS

```
RP/0/RP0/CPU0:PE4#sh lpts pifib hardware police location 0/0/CPU0

-------------------------------------------------------------
                Node 0/0/CPU0:
-------------------------------------------------------------
FlowType               Policer Type    Cur. Rate Burst     npu
---------------------- ------- ------- --------- --------- ---------
Fragment               32102   np      1000      100       0
Fragment               32102   np      1000      100       1
Fragment               32102   np      1000      100       2
Fragment               32102   np      1000      100       3
```

Current Policer rate

  - Default policer rate is 1000 PPS and can be configured via cli:

```
RP/0/RP0/CPU0:PE4(config)#lpts punt police location 0/x/CPU0 exception ipv4 fragment rate ?
  <0-4294967295>  Packets Per Second
```

SPP (Software Packet Path)

- SPP block directly interface with the Linux interface to receive or transmit the packet. It creates packet socket and bind it to the RP and LC CPU interfaces.
- Packets are either forwarded to the NetIO or SPIO.
- Packets which are traversing over SPP interface could be captured with tcpdump:

```
attach location <LC>
tcpdump -xxxi ps-inb.1538
tcpdump -xxxi eth-vf2 (SoC)
```

```
RP/0/RP0/CPU0:PE1#show spp node-counters location
0/0/CPU0

socket/rx
                  ether raw pkts:        159548873
--------------------------------
socket/tx
                        ce pkts:        315456791
--------------------------------
cfm_off_tx_node
                Hostname updated:                1
--------------------------------
fretta/classify
       forwarded to spp clients:        159548667
  forwarded NPU packet to NetIO:        159278931
   forwarded NPU packet to SPIO:           269736
        dropped in classify node:                5
           Fwded to CoPP sampler:              201
              PUNT DROP PACKET:                324
Packets dropped on SPP  PUNT CDP:           125975
                        PUNT ARP:              201
                       PUNT LLDP:           202577
                   PUNT FRAG REQ:        159152620
Fragmented packets      PUNT IPV4_BFD:          12
                      PUNT PUNT_SPIO:        67159
                      SPIO_PROT LACP:        67159
<...>
```

SPP packet capture

```
RP/0/RP0/CPU0:PE1#attach location 0/0/CPU0

#tcpdump -xxx -i eth-vf2

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth-vf2, link-type EN10MB (Ethernet), capture size 262144 bytes
14:17:15.734437 4e:41:50:00:00:11 (oui Unknown) > 4e:41:50:00:00:01 (oui
Unknown), ethertype Unknown (0x876f), length 1696:
        0x0000:  4e41 5000 0001 4e41 5000 0011 876f 1a50
        0x0010:  8000 7e82 1180 5801 0018 8157 170b 0000
        0x0020:  0040 0000 003b 0000 0800 0019 dc60 0120
        0x0030:  8204 00c8 79c0 00c1 4125 2600 18a8 062a
        0x0040:  03e8 9010 fe00 0000 0000 008a 9671 60da
        0x0050:  008a 96ea f8dc 8847 03e8 90fe 07d0 61ff
        0x0060:  4500 0640 0004 0000 ff01 3165 c0a8 0101
        0x0070:  c0a8 0202 0800 4d3f 6871 0004 cafe cafe
        0x0080:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x0090:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x00a0:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x00b0:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x00c0:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x00d0:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x00e0:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x00f0:  cafe cafe cafe cafe cafe cafe cafe cafe
        0x0100:  cafe cafe cafe cafe cafe cafe cafe cafe
```

Internal headers

Src MAC

MPLS ethertype

Dst MAC

192.168.1.1 → 192.168.2.2 ICMP Echo (ping) request

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
00	8A	96	71	60	DA	00	8A	96	EA	F8	DC	88	47	03	E8
90	FE	07	D0	61	FF	45	00	06	40	00	04	00	00	FF	01
31	65	C0	A8	01	01	C0	A8	02	02	08	00	4D	3F	68	71
00	04	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE
CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE
CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE
CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE	CA	FE

5 protocols in packet:

Ethernet | MPLS | MPLS | IPv4 | ICMP | ALL

[+] [-]

- Frame 1: 1622 bytes on wire (12976 bits)
- Ethernet II
- MultiProtocol Label Switching Header
- MultiProtocol Label Switching Header
- Internet Protocol Version 4
- Internet Control Message Protocol

Packets dropped in HW

- Packets dropped in hardware can be captured with the following command:
  `show captured packets[ingress|egress] location <>`

```
RP/0/RP0/CPU0:PE3#show captured packets ingress hex location 0/0/CPU0

------------------------------------------------------
packets dropped in hardware in the ingress direction
buffer overflow pkt drops:219717, current: 200, maximum: 200
------------------------------------------------------
          Wrapping entries
------------------------------------------------------
[1] Dec  28 20:54:42.146, len: 78, hits: 1, buffhdr type: 1
    i/p i/f: HundredGigE0/0/0/4
    punt reason: DROP_PACKET
    Ingress Headers:
    port_ifh: 0xf0, sub_ifh: 0x0, bundle_ifh: 0x800001c
    logical_port: 0x5, pd_pkt_type: 3
    punt_reason: DROP_PACKET (0)
    payload_offset: 29, l3_offset: 29
    FTMH:
    pkt_size: 0x6d, tc: 0, tm_act_type: 0, ssp: 0xc000
    PPH:
    pph_fwd_code: CPU Trap (7), fwd_hdr_offset: 0
    inlif: 0x0, vrf: 0x0, rif: 0x0
    FHEI:
    trap_code: FLP_USER_DEFINE1 (SR_Unknown_Label) (150), trap_qual: 0

    008a9684 44dc008a 967160dc 884703e8 903d07d1 413f4500 00380000 00003f11
    f734c0a8 0118c0a8 0218003f 003f0024 5c04fa82 6d082132 20044978 69600000
    00421011 1213b01e e2690000 0e1a0000 00000000 00000000 00000000 00000000
    00000000 00000000 000000
```

Interface

Punt reason

Trap code from the **`show controllers npu stats trap-all`**

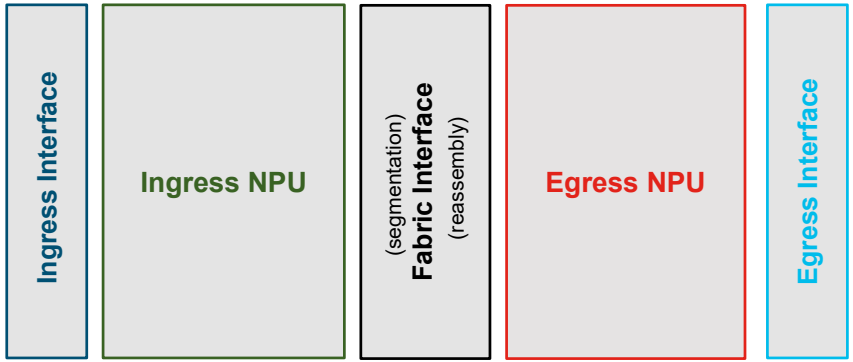
Dropped packet in the hex format

For Reference

Commands summary

```
show controllers npu stats counters-all instance <> loc <>
show controllers np diag counters graphical cdsp instance <> loc <>

show controllers npu stats traps-all instance <> loc <>
show interfaces <> accounting
show captured packets ingress loc <>

show access-lists <> hardware ingress interface <> loc <>
show flow monitor <> cache match <> loc <>

show controllers npu diag last instance <> location <>
show controllers npu diag pp ParsingInfo instance <> loc <>
```

```
show controllers npu stats counters-all instance <> loc <>
show controllers np diag counters graphical cdsp instance <> loc <>

show controllers npu stats traps-all instance <> loc <>
show interfaces <> accounting
show captured packets egress loc <>

show controllers npu diag pp EncapsulationInfo instance <> loc <>
```

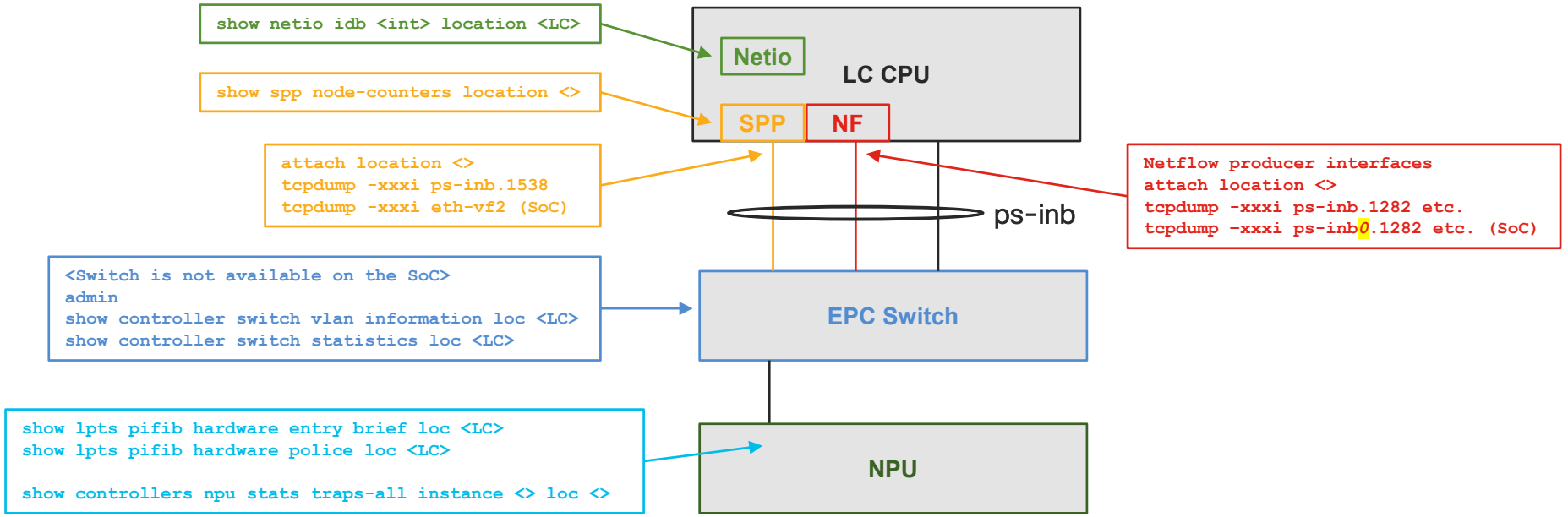



```
show interfaces <interface>
show controllers <interface> stats
show controllers <interface> phy
```

```
show interfaces <interface>
show controllers <interface> stats
show controllers <interface> phy
```

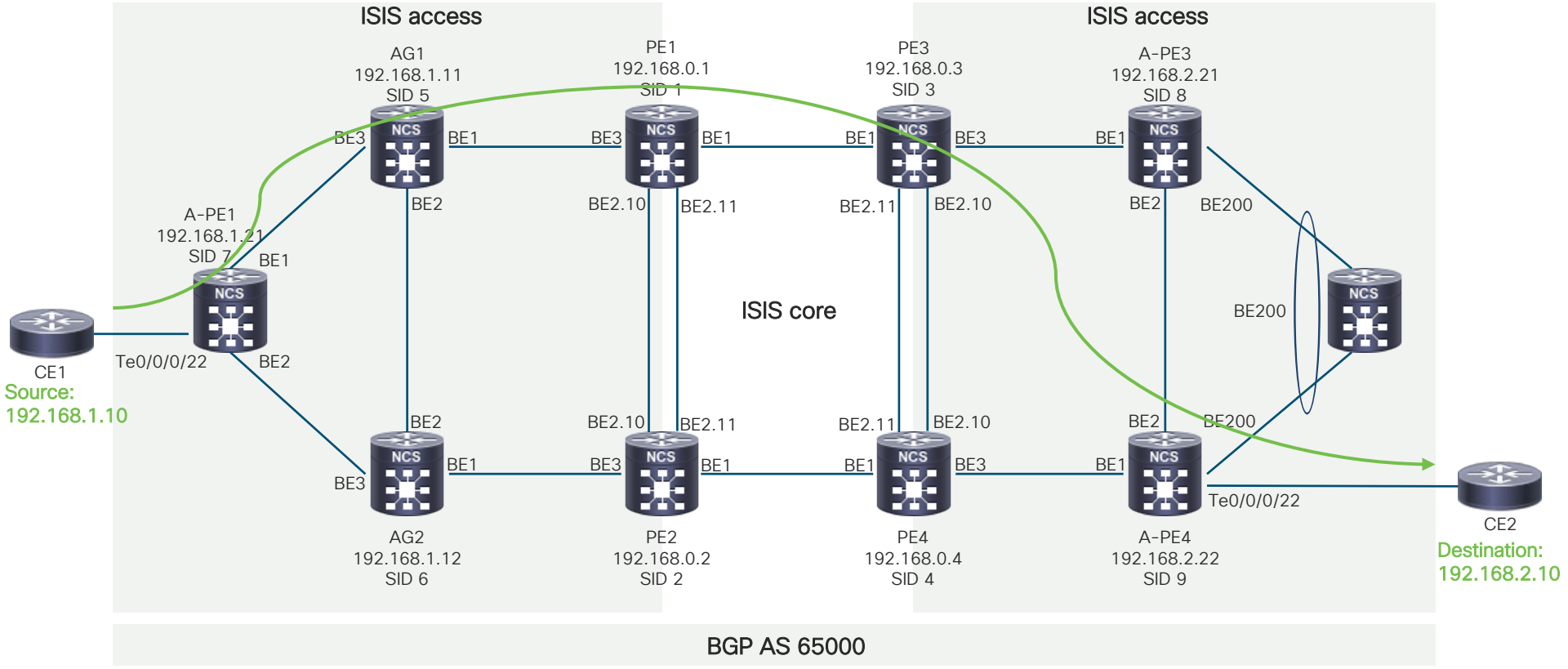
```
show controllers npu voq-usage inteface <egr-interface> instance all loc <>
show controllers npu stats voq ingress <egr-interface> instance all loc <>
show controllers npu stats voq base <voq-base> instance <> loc <>
```

Commands summary – punted packets



Unicast forwarding troubleshooting demo

Network topology and legend



https://youtu.be/0aSTriVj86I

Reported issue

- End customer reported an issue with L3VPN between CE1 & CE2
- 192.168.1.0/24 subnet is connected on CE1
- 192.168.2.0/24 subnet is connected on CE2
- As per end customer, RTP traffic for the 192.168.1.10 -> 192.168.2.10 IP pair is dropped, all other flows are working fine.

For Reference

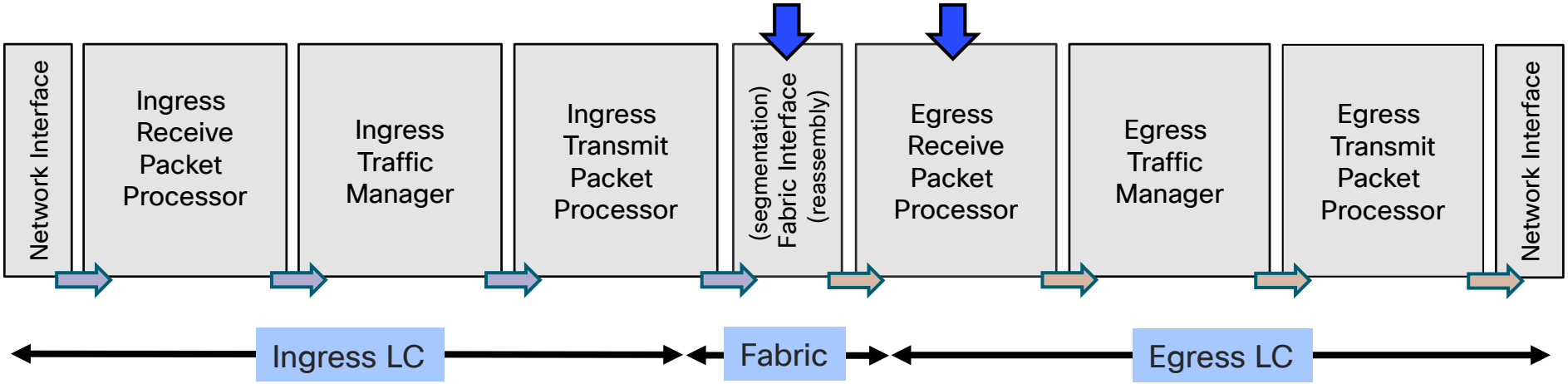
Troubleshooting flow

1. Divide & conquer – find the device which is dropping the packets
   - Check on the destination device (CE) if traffic is arriving from the source, if not, we will need to troubleshoot sender to the receiver path, if traffic arrives, then we will need to troubleshoot reverse path
2. Generate ICMP traffic with the pattern or match already flowing traffic
   - Use Ingress ACL's (for unlabeled traffic) or Ingress Netflow to check if traffic is arriving on the node
   - show controllers npu diag last command might be helpful here as well
3. After device which is dropping the traffic is found, check following:
   - Drops on the ingress|egress interface / controller level
   - Identify NPU where interfaces are hosted
   - Check for the drops under **`show controllers np diag counters graphical cdsp instance <> location <>`**
   - Check for the drops under **`sh controllers npu stats counters-all instance <> location <>`**
   - Check VOQ stats
   - Refer to the trap stats if a packet is dropped or sent to the CPU
   - If traffic is punted to the CPU, however we don't see any trap drops, check SPP counters for any drops
   - If packet is seen in trap counters and not in SPP, run tcpdump on the node (LC or RP) on the SPP interface
   - Check for the drops under **`show captured packets [ingress|egress]`**

Multicast forwarding troubleshooting

Multicast in NCS 5500

- Multicast support introduced in IOS XR 6.1.2
- Initially limited to Source Specific Protocols and IPv4 only (IGMPv3 and PIM SSM)
  - (S,G) information for v4/v6 stored in LPM
- 2-level replication architecture (fabric and egress NPU)
- No ingress replication



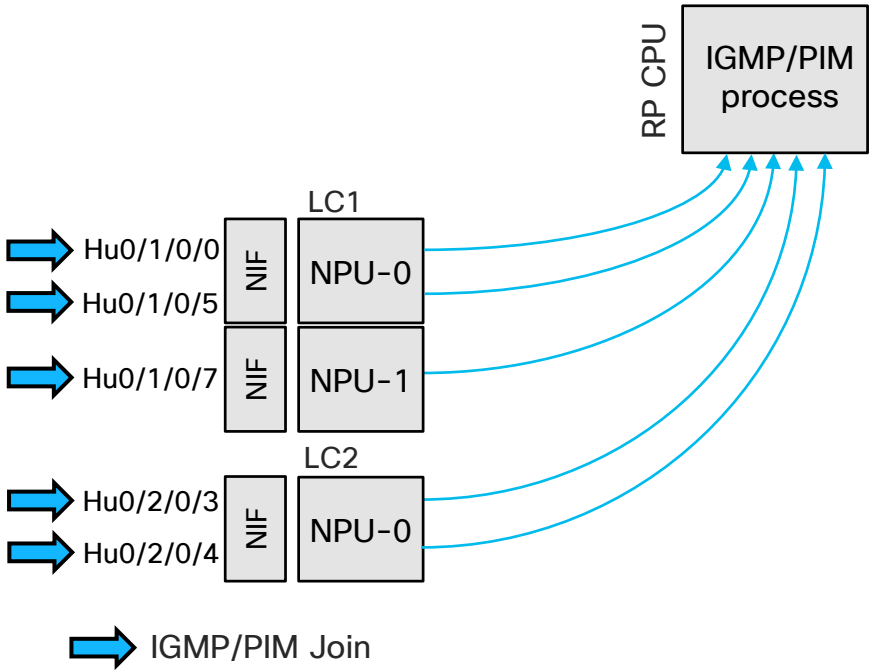
Multicast in NCS 5500

- IPv4 and IPv6 multicast routes take 1 entry each in LPM
  - IPv4 key (VRF, S, G) stored in LPM
  - IPv6 key (VRF, G) stored in LPM
- LPM lookup does two things
  - Performs RPF check
  - Gives a FEC-ID which points to MCID
- Note for IPv6:
  - The key used for lookup is comprised of (VRF, G), we ignore the source
  - A key w/ (VRF, S, G) would have taken more entries in LPM, impacting the scale
  - → We only support 1 source per Group for IPv6

NCS 5500 Multicast

Control Plane

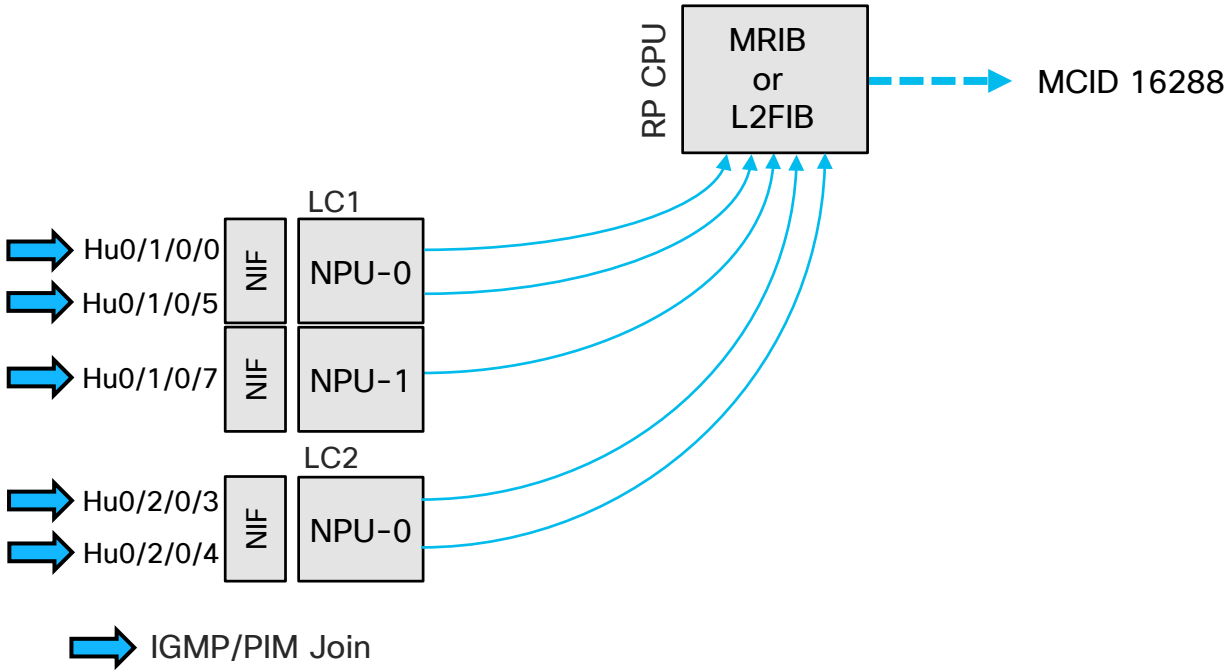
- IGMP and PIM joins are punted to RP CPU process (igmp/pim)
- Packets are using EPC internal network to reach the process executed on RP LXC



NCS 5500 Multicast

Control Plane

- If it's a new group, the process (MRIB or L2FIB) will allocation a Multicast ID (MCID)
- If a MCID is already allocated, information will be updated based on join/leave



NCS 5500 Multicast

Control Plane: Identifying MCID

- MCID is often referred as FGID internally
- You can find the MCID associated to a (*,G) or (S,G) pair with the following CLI:

```
RP/0/RP0/CPU0:ios#show mrib route <group> detail
```

```
RP/0/RP0/CPU0:PE4#show mrib route 239.255.1.1 det
<…>
(*,239.255.1.1) Ver: 0xcbf2 RPF nbr: 192.168.4.9 Flags: C RPF, FGID: 16356, Statistics enabled: FALSE
  Up: 00:47:53
  Incoming Interface List
    Bundle-Ether1 Flags: A, Up: 00:47:53
  Outgoing Interface List
    Bundle-Ether3 (0/0/0) Flags: F NS, Up: 00:39:58
    Bundle-Ether2.11 (0/0/0) Flags: F NS, Up: 00:39:58

(192.168.10.1,239.255.1.1) Ver: 0x823c RPF nbr: 192.168.4.13 Flags: RPF, FGID: 16288, Statistics enabled: FALSE
  Up: 00:39:02
  Incoming Interface List
    Bundle-Ether2.11 Flags: A, Up: 00:39:02
  Outgoing Interface List
    Bundle-Ether3 (0/0/0) Flags: F NS, Up: 00:39:02
```

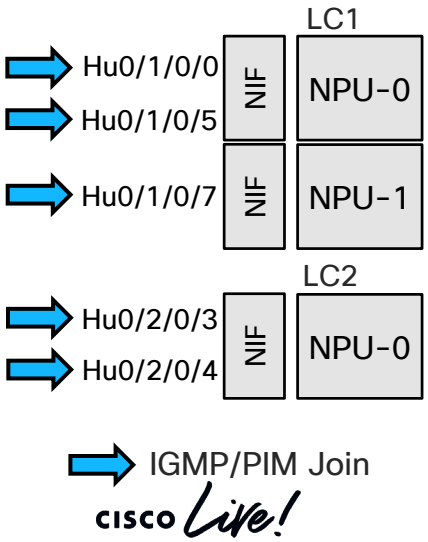
Incoming interface list

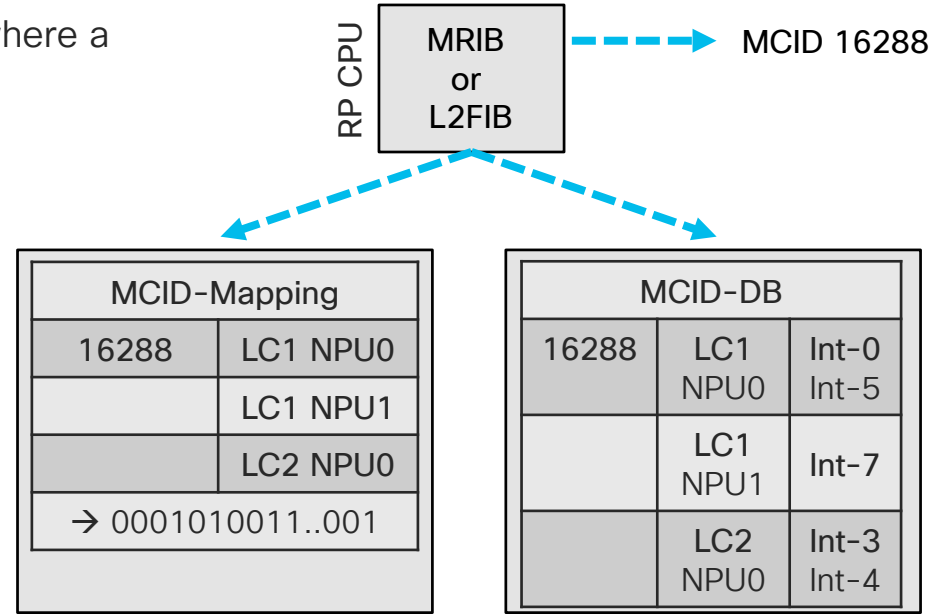
Outgoing interface list

NCS 5500 Multicast

Control Plane Tables

- The process running on RP CPU will dynamically compute two tables for each MCID
- MCID mapping is a 128 bitmap mask where Ones represent the NPUs who received a join and who expect a copy of the packet from the fabric
- MCID-DB associates ports where a replication is expected





NCS 5500 Multicast

MCID Bitmap – fabric side (admin mode)

- Each LC has 6 bits corresponding to its 6 NPUs. The bit which is set will tell us the LC and the NPU to which the FGID belongs. The same thing is also displayed using the **show mrib fgid info <fgid number>** command in the hex format.

```
sysadmin-vm:0_RP0# show controller fabric fgid information id 16288 detail

Displaying FGID: 16288
FGID Information:


FGID number:           16288

FGID Hex bitmap:
        0x0000000000000001-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000
        0x0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000
        0x0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000
        0x0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000
        0x0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000 0000000000000000-0000000000000000

FGID Binary bitmap:                                                                                                    First bit is set, which means LC0/0 NPU #0
        00000000000000000000000000000000000000000000000000000000000000000000000000000001-00000000000000000000000000000000000000000000000000000000000000000000000000000000
        00000000000000000000000000000000000000000000000000000000000000000000000000000000-00000000000000000000000000000000000000000000000000000000000000000000000000000000
        00000000000000000000000000000000000000000000000000000000000000000000000000000000-00000000000000000000000000000000000000000000000000000000000000000000000000000000
        00000000000000000000000000000000000000000000000000000000000000000000000000000000-00000000000000000000000000000000000000000000000000000000000000000000000000000000
<…>


FGID associated fabricq Ids:
[1] :=
        0/0/0,                                                                                                         LC 0/0 NPU# 0

FGID associated client application:
client id = 1, client name = MRIB-ipv4-default, SDR name = default-sdr
```

NCS 5500 Multicast

MCID Bitmap

```
RP/0/RP0/CPU0:PE4#show mrib fgid info 16288

FGID information
----------------

FGID (type)      : 16288 (Primary)
Context          : IP (0xe0000000, 192.168.10.1, 239.255.1.1/32)
Members[ref]     :  0/0/0[2]
LineCard Slot    :  0 ::  Npu Instance 0
FGID bitmap    :
0x0000000000000001  0x0000000000000000  0x0000000000000000  0x0000000000000000
0x0000000000000000  0x0000000000000000  0x0000000000000000  0x0000000000000000
0x0000000000000000  0x0000000000000000  0x0000000000000000  0x0000000000000000
0x0000000000000000  0x0000000000000000  0x0000000000000000  0x0000000000000000

FGID chkpt context valid : TRUE
FGID chkpt context :
                table_id 0xe0000000 group 0xefff0101/32 source 0xc0a80a01
FGID chkpt info : 0x23000000

Fgid in batch          : NO
Secondary node count   : 0
```

Decoded LC/NPU# from the FGID bitmap

Should be set to TRUE

Should be set to NO

NCS 5500 Multicast

What is the command to see MCID-DB per NPU mapping?

- From IOS-XR 6.6.3 following command will show MCID-BD per NPU

```
RP/0/RP0/CPU0:ios#show mfib hardware egress mcid <FGID> npu all location all
```

- Example output:

```
RP/0/RP0/CPU0:PE4#show mfib hardware egress mcid 16288 npu all location all

0/0/CPU0 npu:0 pp_port:0x4000009 ecap1:0x61c ecap2:0x0 sysport:9 interface: Hu0/0/0/3
```

Pp_port in hex (0x9 = 9)

Outgoing interface (BE3 member)

```
RP/0/RP0/CPU0:PE4#show controllers npu voq-usage interface HundredGigE0/0/0/3 instance all loc 0/0/CPU0

--------------------------------------------------------------------
Node ID: 0/0/CPU0
Intf          Intf      NPU NPU  PP    Sys   VOQ   Flow   VOQ     Port
name          handle     #  core Port  Port  base  base   port    speed
              (hex)                                        type
--------------------------------------------------------------------------
Hu0/0/0/3     f8         0   0     9     9   1064   5416 local   100G
```

NCS 5500 Multicast

Ecap ID programming verification

- Besides of the pp_port / sysport programming verification, also ecap ID should be checked:

```
RP/0/RP0/CPU0:PE4#show mfib hardware egress mcid 16288 npu all location all

0/0/CPU0 npu:0 pp_port:0x4000009 ecap1:0x61c ecap2:0x0 sysport:9 interface: Hu0/0/0/3
```

- Collect ifhandle for the egress interface

```
RP/0/RP0/CPU0:PE4#show im database interface Bundle-Ether 3 | inc ifh
<…>
Interface Bundle-Ether3, ifh 0x0800002c (up, 9216)
```

- Verify if rif is equal to the Ecap ID

```
RP/0/RP0/CPU0:PE4#attach location 0/0/CPU0
<…>
#dnx_pidb_test -s ifh=0x0800002c | grep rif
rif                          : 0x61c
```

In the Bundle, all members will point to this value

NCS 5500 Multicast

What is the command to see MCID-DB per NPU mapping?

- Old command syntax (prior 6.6.3)

```
RP/0/RP0/CPU0:PE4#sh controllers fia diag 0 "diag multicast egress 16288 members " loc 0/0/cpu0

Node ID: 0/0/CPU0
Multicast ID: 16288
Ingress/Egress: Egress-list
Members count: 1
 ------------------------------------------------------------------------------
    Member    |  Gport Port  |  Gport Type  | 1st Encap ID/CUD | 2nd Encap ID/CUD |
 ------------------------------------------------------------------------------
       0      | 0x4000009    |   Local_Port |      0x61c       |      0x0         |
```

If multiple egress interfaces are preset, collect this output for each NPU# and LC and verify all interfaces

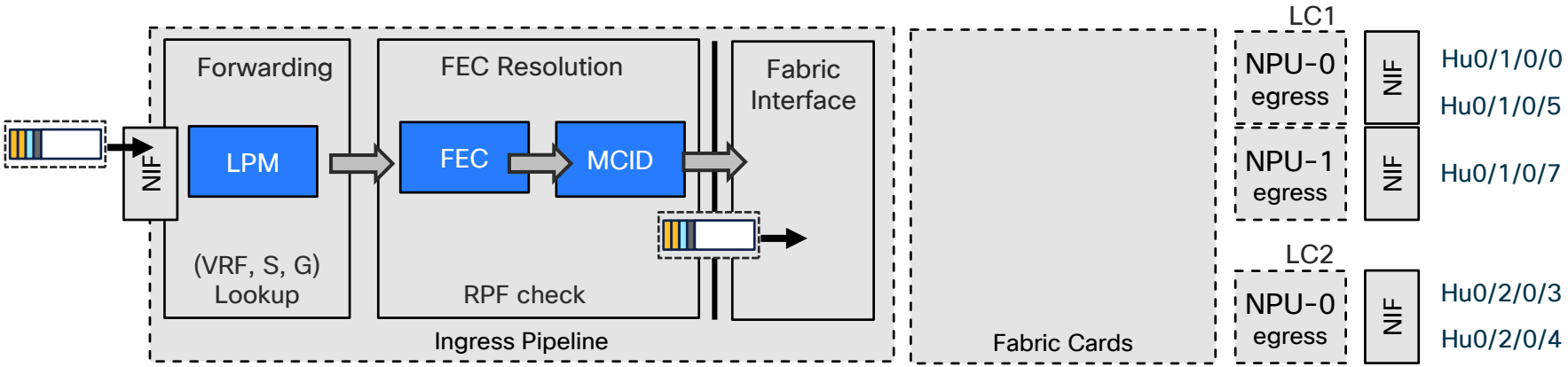
pp_port in hex

```
RP/0/RP0/CPU0:PE4#show controllers npu voq-usage interface HundredGigE0/0/0/3 instance all loc 0/0/CPU0

--------------------------------------------------------------------
Node ID: 0/0/CPU0
Intf          Intf      NPU  NPU   PP    Sys   VOQ   Flow   VOQ     Port
name          handle     #   core  Port  Port  base  base   port    speed
              (hex)                                       type
--------------------------------------------------------------------
Hu0/0/0/3     f8         0    0     9     9   1064   5416 local   100G
```

NCS 5500 Multicast

Data Plane

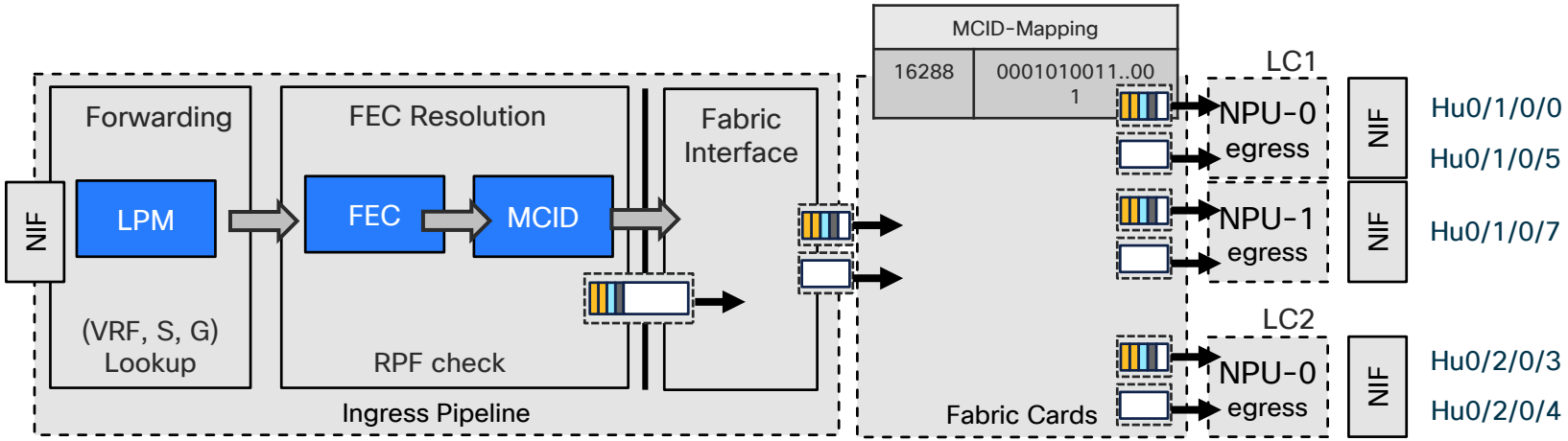
- Multicast Packet is received on ingress interface
- Lookup provides a FEC-ID itself pointing to MCID
  - In LPM for L3 packets (we will use it as an example)
  - In iTCAM for L2 packets (future plans to move them to LPM too)
- RPF check is performed in FEC block



NCS 5500 Multicast

Data Plane

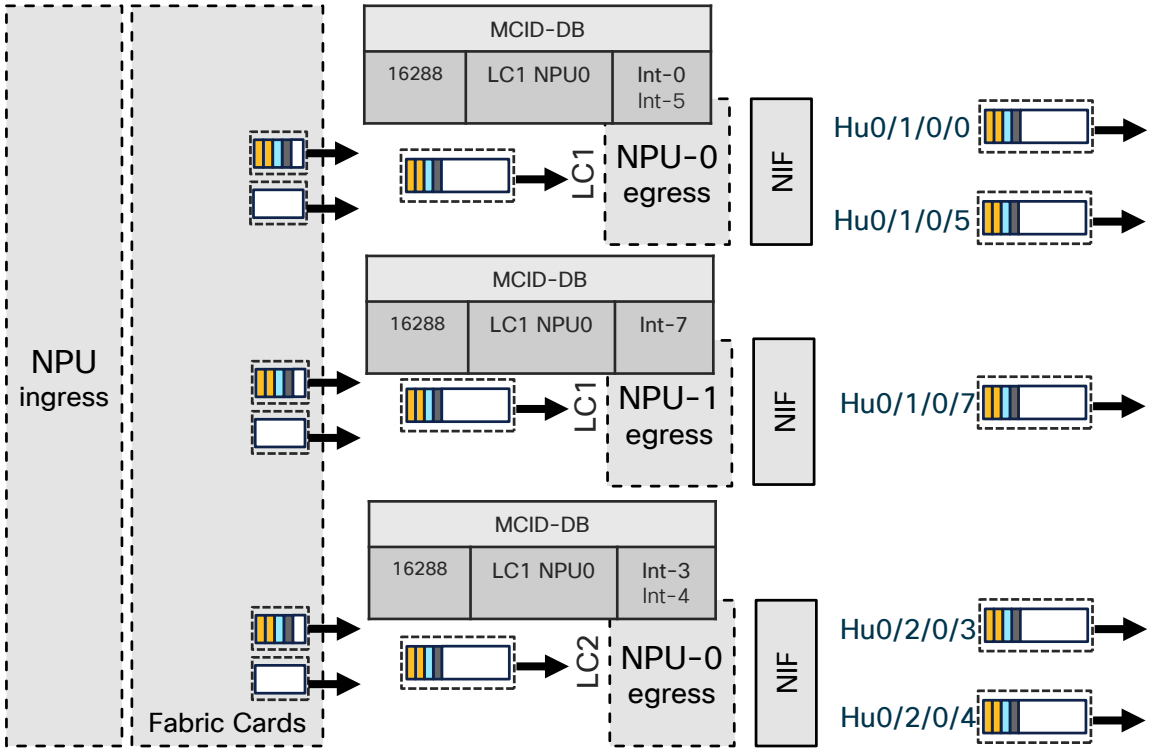
- Internal Header has been marked with MCID
- Packet is passed to the fabric interface and split in cells
- Based on MCID-Mapping bitmap, the cells are replicated in the fabric to the NPUs where they are re-assembled by fabric interfaces



NCS 5500 Multicast

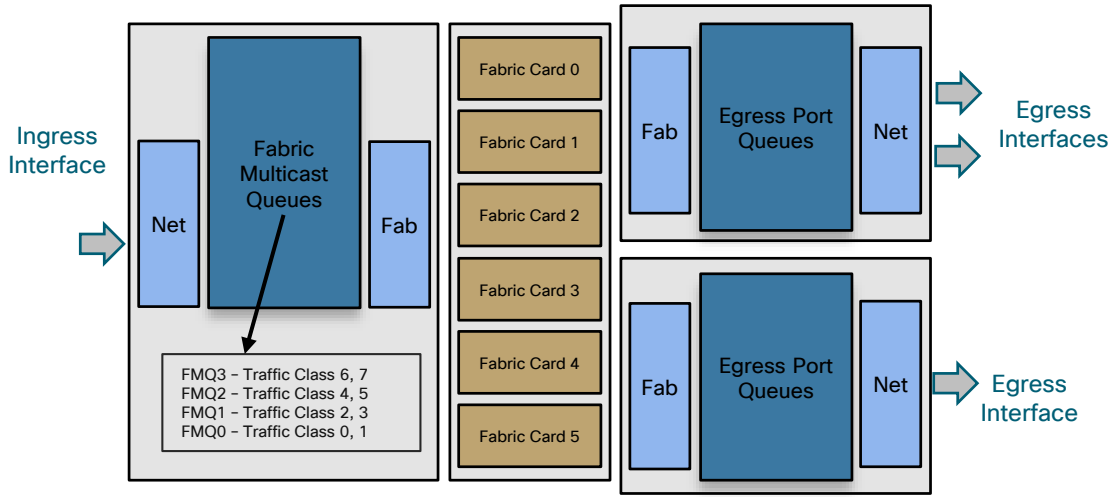
Data Plane

- Re-assembled packets will be replicated on egress NPU based on MCID-DB information
- It's the second level of replication



FMQ Definition

- Mcast packets are not classified in VOQs, but in 4 Fabric Multicast Queues
  - FMQ0 to FMQ2 will be Low Priority
  - FMQ3 is treated as High Priority in the Egress port queues
- Mapping is done in ingress between Traffic Class pairs are FMQ
  - Without policy-map, all mcast traffic is TC=0, which maps to FMQ0



VOQ Credit Mechanism for multicast

- Unicast packets (descriptors) are stored in various VOQs
- These queues are constantly monitored
  - Credit allocation is reduced or stopped based on thresholds
- Multicast packets (descriptors) are stored in Fabric Multicast Queue (FMQs) !
- No scheduling / credit allocation, hence, we don’t enforce flow control !
  - Because it could generate head of line blocking situations
  - We just drop the traffic is thresholds are exceeded
- In case of egress interface congestion
  - If one traffic kind (unicast or multicast) is high priority, it will take full precedence over the other
  - If both traffic kinds are of same priority, then the forwarding will be 80% unicast / 20% multicast

FMQ statistics

- To display FMQ statistics, following command should be used:

```
RP/0/RP0/CPU0:ios#show controllers npu stats voq base 0 instance all location <>
```

- Example output:

```
RP/0/RP0/CPU0:A-PE4#show controllers npu stats voq base 0 instance all location all

Asic Instance     =            0
VOQ Base          =            0
       ReceivedPkts    ReceivedBytes   DroppedPkts     DroppedBytes
-----------------------------------------------------------------
TC_0 = 7772735752      10850739109792  6427314         2384335486
TC_1 = 0               0               0               0
TC_2 = 0               0               0               0
TC_3 = 0               0               0               0
TC_4 = 0               0               0               0
TC_5 = 0               0               0               0
TC_6 = 0               0               0               0
TC_7 = 0               0               0               0
```

NCS 5500 Multicast

MFIB global counters

- Following command will show all global mcast drop counters on the LC

```
RP/0/RP0/CPU0:PE4#show mfib ipv4 counter location 0/0/CPU0

MFIB global counters are :
* Packets [no input idb]                              : 0
* Packets [failed route lookup]                       : 0
* Packets [Failed idb lookup]                         : 0
* Packets [Mcast disabled on input I/F]               : 0
* Packets [encap drops due to ratelimit]              : 0
* Packets [MC disabled on input I/F (iarm nfn)]       : 0
* Packets [IC ignored due to IDB unlinked from VRF]   : 0
* Packets [MVPN TTL 1 decapped]                       : 0
* Packets [Failed TTL check]                          : 0
* Packets [Outgoing list error]                       : 0
* Packets [Failed FSV get]                            : 0
* Packets [Failed fint idb lookup]                    : 0
* Packets [Failed to send pkt to LC]                  : 0
* Packets [Already delivered by HW]                   : 0
* Packets [Miscellaneous Failure]                     : 319
* Packets [Invalid interface handle]                  : 0
* Packets [Null route]                                : 0
* Packets [Drop preserved packets]                    : 0
* Packets [Send to fabric failed]                     : 217
* Packets [Preserved packet is stale]                 : 0
```

NCS 5500 Multicast

HW multicast route statistics

- By default on NCS5500 HW multicast route statistics are not available
- To enable them, we need to create an ACL to match (S,G) routes and in the next step, enable route-stats for l3mcast

```
RP0/0/RP0/CPU0:PE4# configure

/* Configure an ACL matching the (S,G) routes for which statistics have to be captured:*/
RP0/0/RP0/CPU0:router(config)# ipv4 access-list mcast-counter
RP0/0/RP0/CPU0:router(config-acl)# 10 permit ipv4 host 192.168.10.1 239.255.1.0/24
RP0/0/RP0/CPU0:router(config-acl)#commit
RP0/0/RP0/CPU0:router(config-acl)#exit

/* Enable multicast route statistics for the configured ACL on the default VRF. */
RP0/0/RP0/CPU0:router(config)# hw-module route-stats l3mcast vrf default ipv4 mcast-counter
```

NCS 5500 Multicast

HW multicast route statistics

- To display statistics, following command should be used:

```
RP/0/RP0/CPU0:PE4#show mfib route statistics 239.255.1.1 192.168.10.1 location 0/0/CPU0

IP Multicast Forwarding Information Base
<…>
SW/HW Forwarding/Replication Counts: Packets in/Packets out/Bytes out
SW Failure Counts: RPF / TTL / Empty Olist / Encap RL / Other
HW Drop Counts: Ingress / Egress
HW Forwarding Rates: bps In/pps In/bps Out/pps Out

(192.168.10.1,239.255.1.1),  Flags:
  Up: 00:57:50
  Last Used: 00:42:32
  SW Forwarding Counts: 56/0/0
  SW Replication Counts: 56/0/0
  SW Failure Counts: 2/0/0/0/0
  HW Forwarding Counts: 2418250/N/A /N/A
  HW Replication Counts: 2418250/N/A /N/A
  HW Drop Counts: 0/N/A
  HW Forwarding Rates: N/A /N/A /N/A /N/A
  Bundle-Ether3 Flags:  NS EG, Up:00:57:50
  Bundle-Ether2.11 Flags:  A, Up:00:57:50
```

Mcast traffic handled by software switching

Mcast traffic handled in hardware

NCS 5500 Multicast

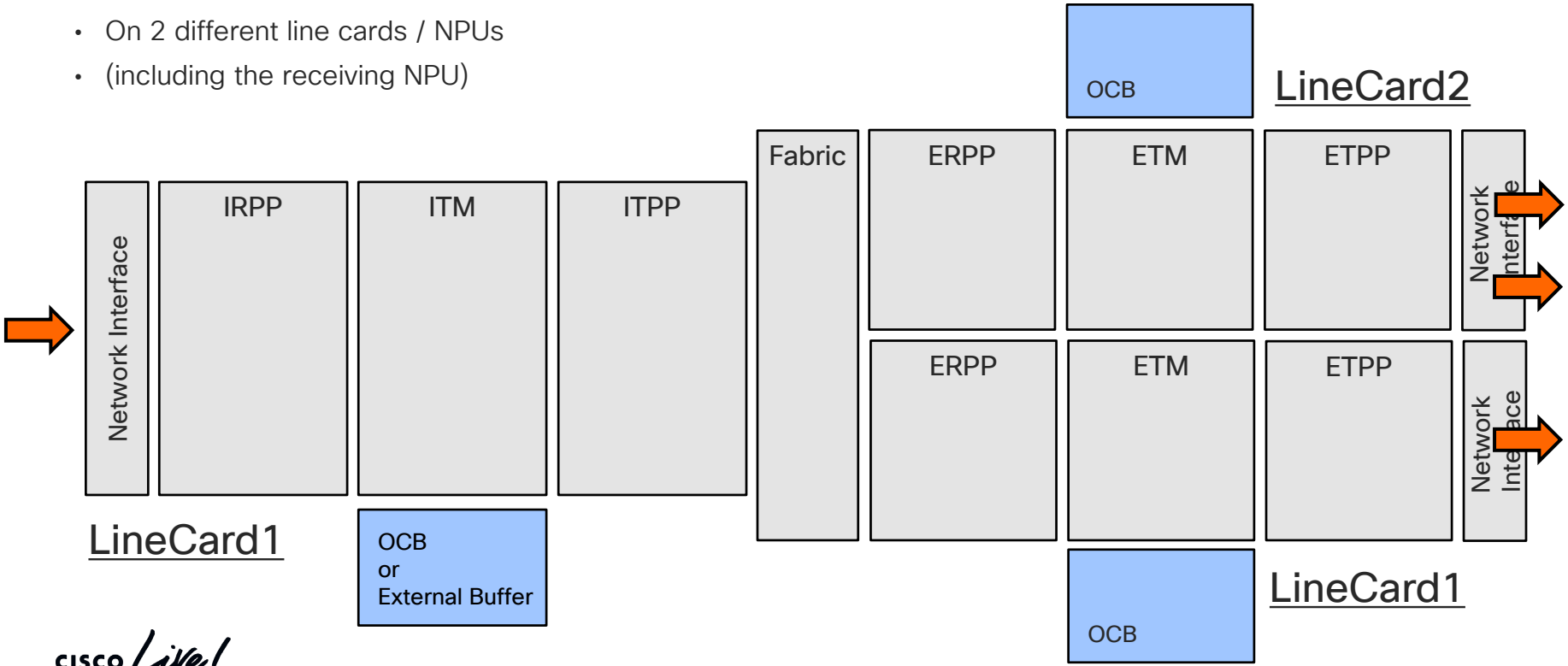
HW multicast route statistics

```
RP/0/RP0/CPU0:PE4#show mfib route rate 192.168.10.1 239.255.1.1 detail

IP Multicast Forwarding Rates
(Source Address, Group Address)
  Incoming rate: (Incoming interface)
    Node: (Incoming node) : pps/bps
      HW Incoming count : (in packets)
      HW Drop count : (in packets)
  Outgoing rate:
    Node: (Outgoing node) : pps/bps
      HW Forwarding count: (in packets)
      HW Drop count: (in packets)
        Interfaces: (Outgoing interface list)

(192.168.10.1,239.255.1.1)
  Incoming rate :  BE2.11
    Node : 0/0/CPU0 : 0 / 0
      HW Incoming count : 6413929 packets
      HW Drop count : 0 packets
  Outgoing rate :
    Node : 0/0/CPU0 : 0 / 0
      HW Forwarding count: 0 packets
      HW Drop count: 0 packets
        Interfaces: BE3
```

Only Ingress counters

Life of a Multicast Packet in Chassis

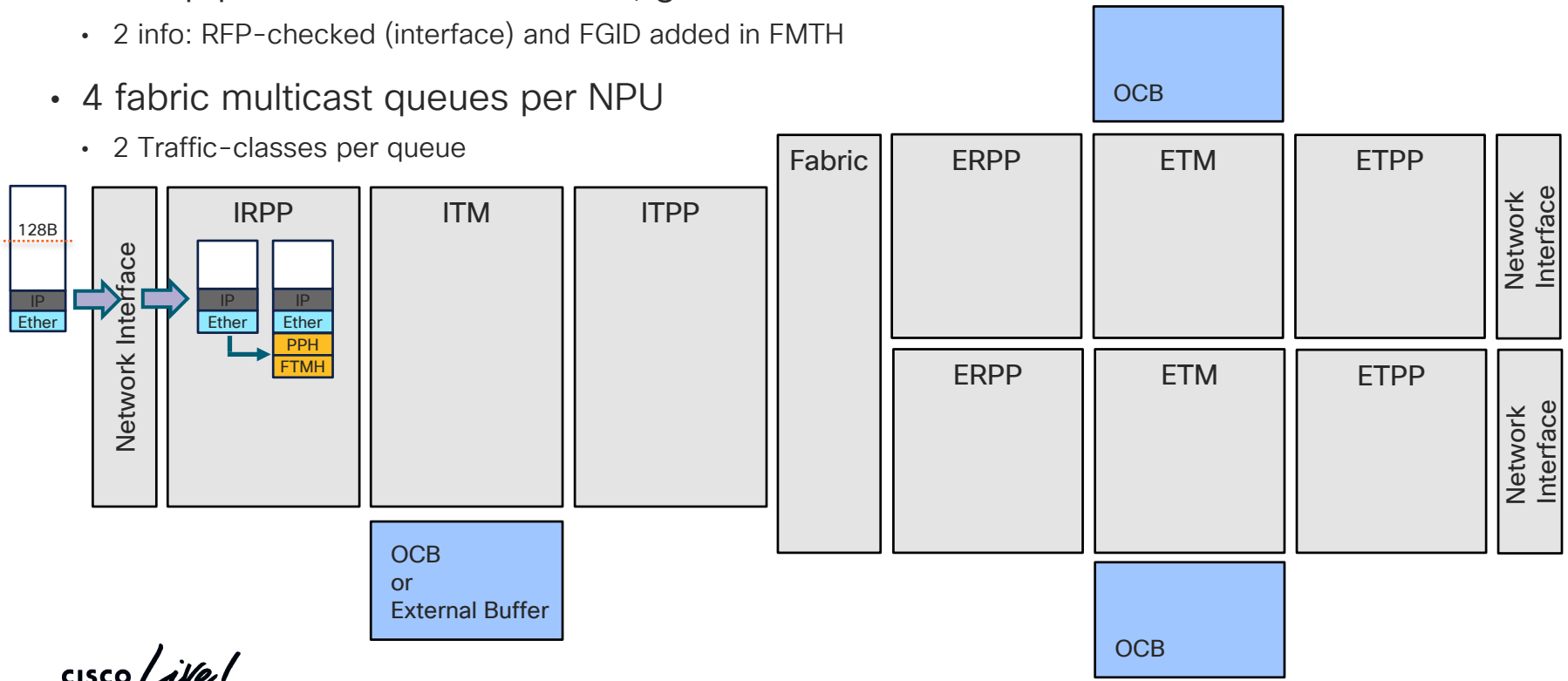
- Multicast packet is received on LC1
- And replicated to three ports:
  - On 2 different line cards / NPUs
  - (including the receiving NPU)



Life of a Multicast Packet in Chassis

Ingress Receive Packet Processor

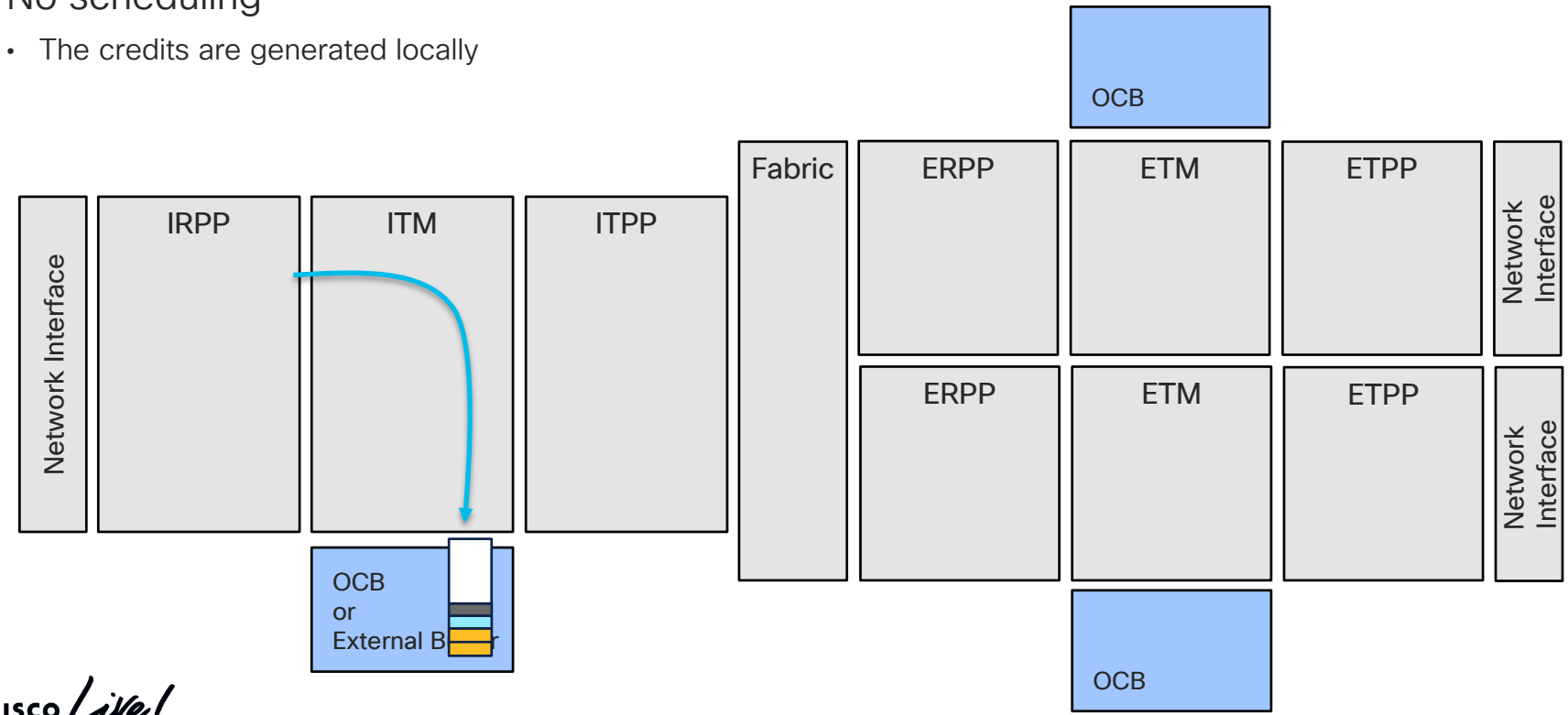
- Lookup performed in LPM/KAPS, gives a FEC-ID
  - 2 info: RFP-checked (interface) and FGID added in FMTH
- 4 fabric multicast queues per NPU
  - 2 Traffic-classes per queue



Life of a Multicast Packet in Chassis

Ingress Traffic Manager

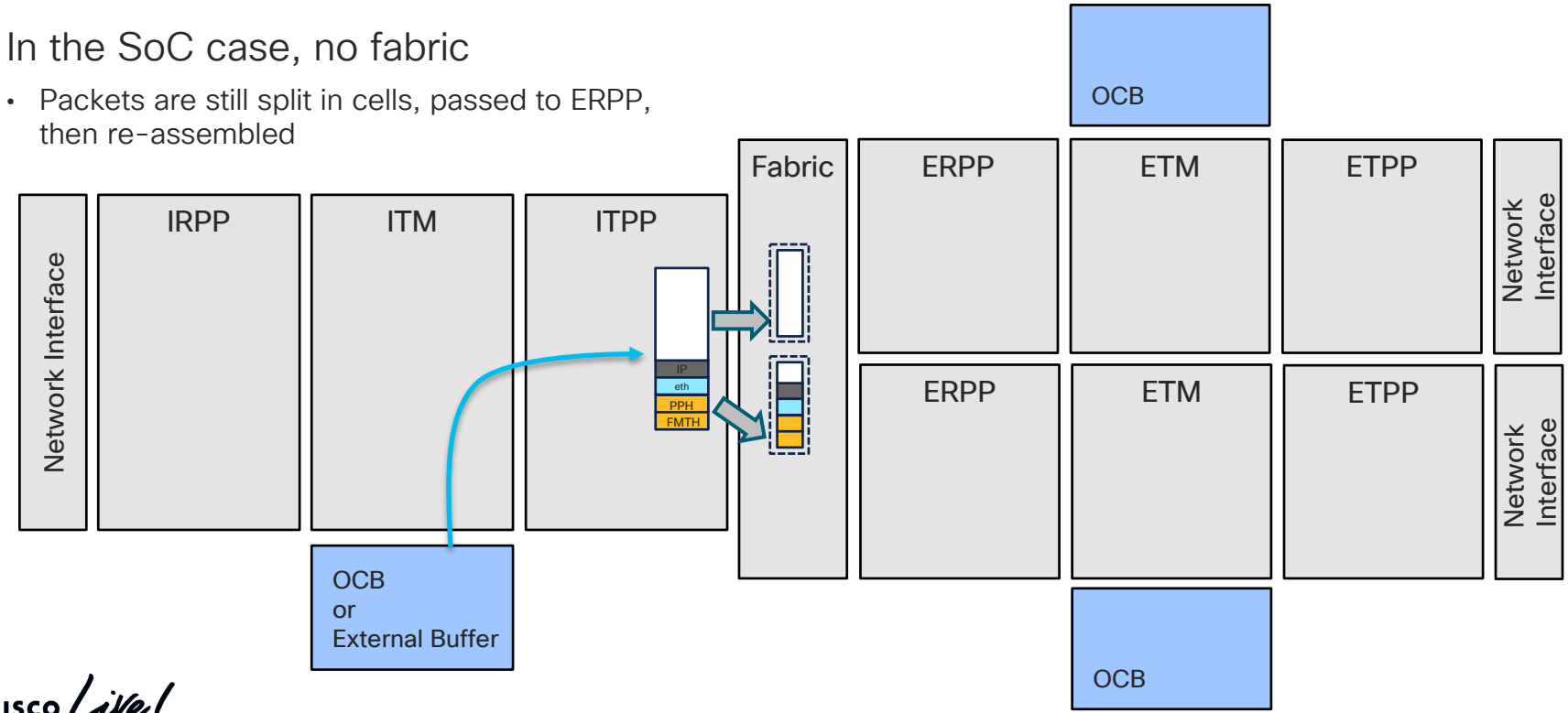
- No scheduling
  - The credits are generated locally



Life of a Multicast Packet in Chassis

Ingress Transmit Packet Process

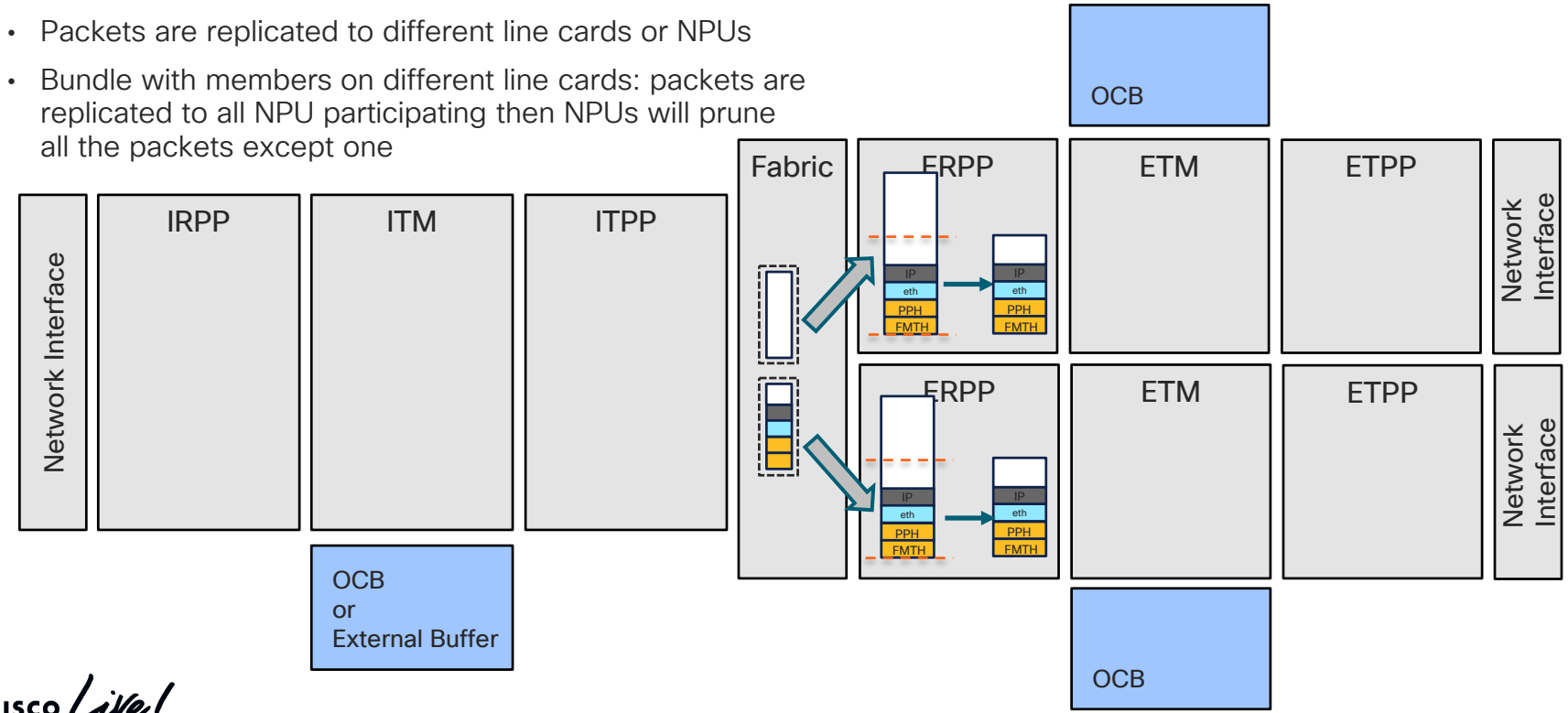
- Packets are going to fabric even for local routing
- In the SoC case, no fabric
  - Packets are still split in cells, passed to ERPP, then re-assembled



Life of a Multicast Packet in Chassis

Egress Receive Packet Processor

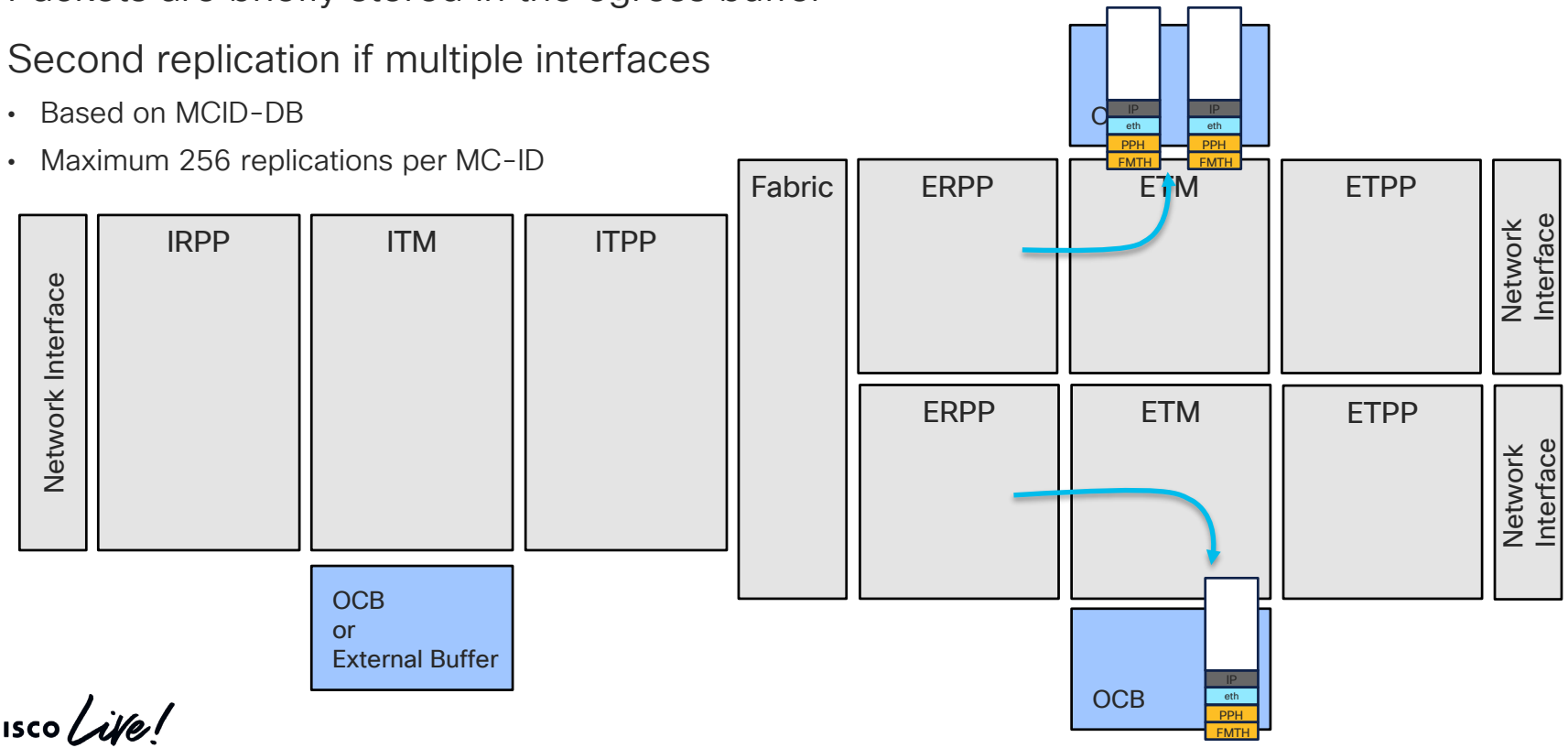
- First replication level based on FGID
  - Packets are replicated to different line cards or NPUs
  - Bundle with members on different line cards: packets are replicated to all NPU participating then NPUs will prune all the packets except one



Life of a Multicast Packet in Chassis

Egress Traffic Manager

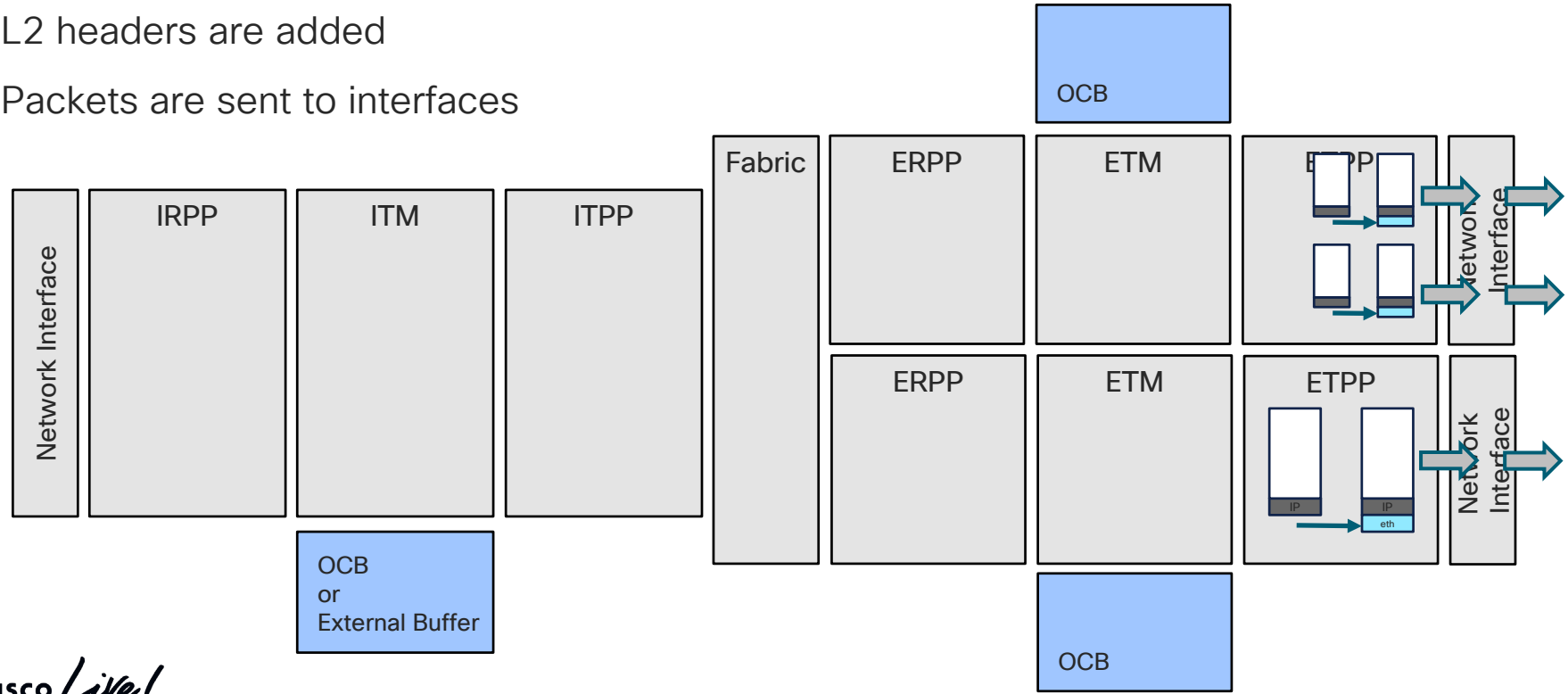
- Packets are briefly stored in the egress buffer
- Second replication if multiple interfaces
  - Based on MCID-DB
  - Maximum 256 replications per MC-ID



Life of a Multicast Packet in Chassis

Egress Transmit Packet Processor

- L2 headers are added
- Packets are sent to interfaces



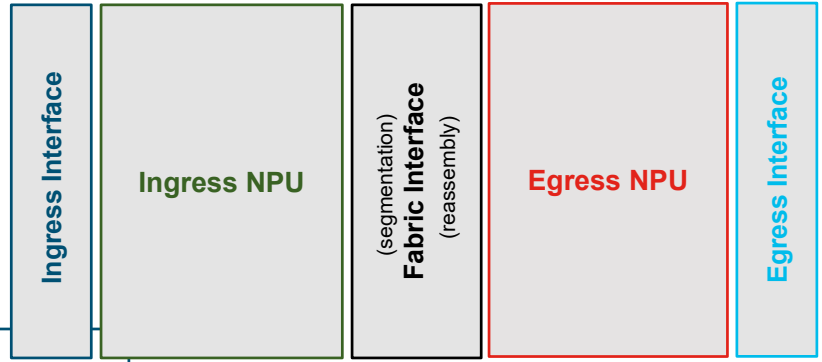
For Reference

Commands summary

```
show controllers npu stats counters-all instance <> loc <>
show controllers np diag counters graphical cdsp instance <> loc <>

show controllers npu stats traps-all instance <> loc <>
show interfaces <> accounting
show captured packets ingress loc <>

show access-lists <> hardware ingress interface <> loc <>
show flow monitor <> cache match <> loc <>

show controllers npu diag last instance <> location <>
show controllers npu diag pp ParsingInfo instance <> loc <>
```

```
show controllers npu stats counters-all instance <> loc <>
show controllers np diag counters graphical cdsp instance <> loc <>

show controllers npu stats traps-all instance <> loc <>
show interfaces <> accounting
show captured packets egress loc <>

show controllers npu diag pp EncapsulationInfo instance <> loc <>
```

```
admin show controller fabric fgid information id <fgid> detail
```




```
show interfaces <interface>
show interfaces <interface> accounting
show controllers <interface> stats
show controllers <interface> phy
```

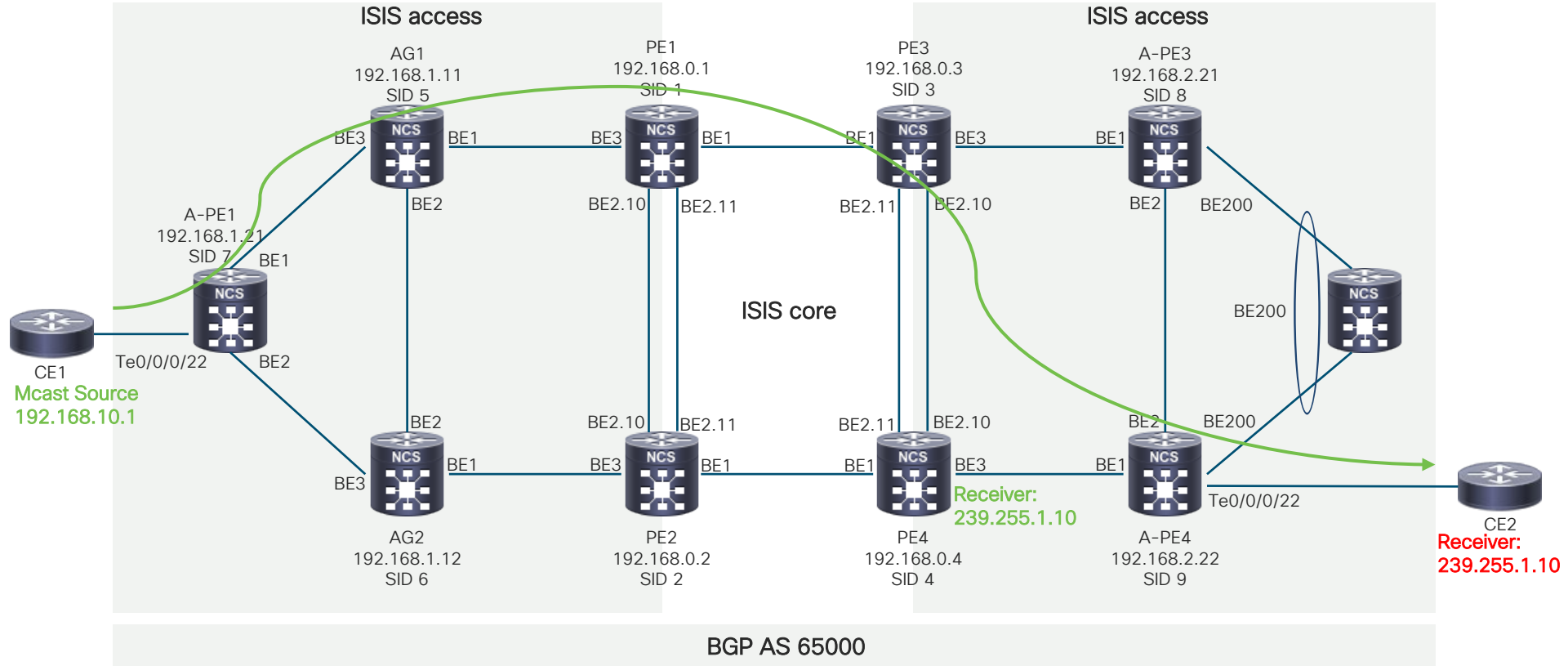
```
show interfaces <interface>
show interfaces <interface> acc
show controllers <interface> stats
show controllers <interface> phy
```

```
show mrib route <group> detail
show mfib route <group> location <loc>
show mfib ipv4 counter location <loc>
show mfib route rate <source> <group> detail
show mrib fgid info <FGID>
show mfib hardware egress mcid <FGID> npu <> loc <loc>
```

```
show controllers npu voq-usage inteface <egr-interface> instance all loc <>
show controllers npu stats voq base 0 instance all location <>
```

Multicast forwarding troubleshooting demo

Network topology and legend



https://youtu.be/yi3CnldRODk

Reported issue

- Administrator enabled a new mcast group in the network – 239.255.1.10, however for some reason it's not working on CE2 router
- There is also another receiver connected on PE4, and on this one there is no issue
- All other groups (239.255.1.1 -> 239.255.1.9) are working fine on CE2

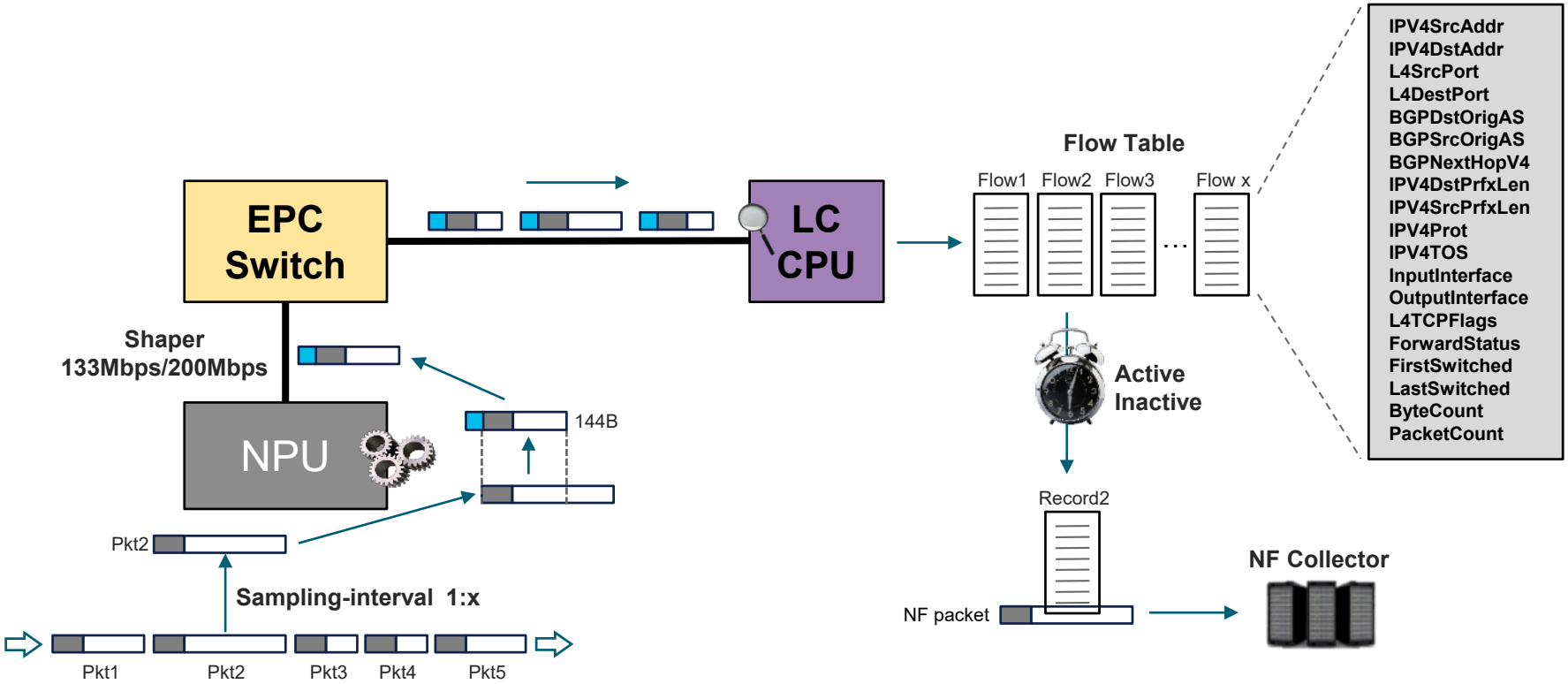
Troubleshooting Flow

- Following info should be determined when you start troubleshooting:
  - Number of S,G flows expected on your router.
  - Expected Incoming and Outgoing interfaces for an (S,G).
  - RPF addresses
  - Integrity of the routing table and any changes that may have occurred that could have impacted RPF or Source
  - Know the interfaces that have QoS configs
- Find the last device on which mcast stream is received
- Examinee **`show mrib route <group> detail`** for the FGID & incoming / outgoing interfaces
- Check ingress / egress interface / controller counters for drops
- Check interface accounting if multicast traffic is leaving the node
- Check multicast HW counters
- Check global drop counters
- Check FMQ stats for congestion
- Check if MCID mapping / MCID-DB programming is correct
- Follow unicast packet troubleshooting flow from this presentation

Netflow troubleshooting on NCS 5500

Netflow Principles

- Netflow Architecture Basics



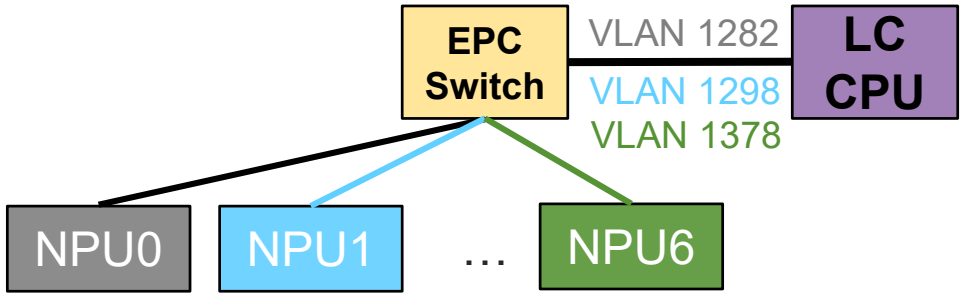
How Does Flow Monitoring Work – summary

- Router generates flow records from a packet stream
- Records created by extracting fields from sampled packets
- Database (aka NetFlow cache) stores current flows and their accounting information
- Flows expired from the database based on various criteria (aka aging)
- Expired flows exported from the router to external collector
- Various applications on or behind collector analyze data

VLANs used in Line Cards and NCS 5502

```
sysadmin-vm:0_RP0# show controller switch vlan information location 0/LC0/LC-SW

SDR
Identifier  SDR Name     VLAN           VLAN Use
-------------------------------------------------------------------------
1           sysadmin-vm  1     (0x001)  Platform EMON
                         17    (0x011)  Platform HOST
                         3073  (0xC01)  Calvados IPC
2           default-sdr  1282  (0x502)  SDR 2 Platform Netflow 1
                         1298  (0x512)  SDR 2 Platform Netflow 2
                         1314  (0x522)  SDR 2 Platform Netflow 3
                         1330  (0x532)  SDR 2 Platform Netflow 4
                         1346  (0x542)  SDR 2 Platform Netflow 5
                         1362  (0x552)  SDR 2 Platform Netflow 6
                         1538  (0x602)  SDR 2 Platform SPP
                         1554  (0x612)  SDR 2 Platform BFD
                         1570  (0x622)  SDR 2 Platform MAC learning
                         1794  (0x702)  SDR 2 Third Party Applications
                         3074  (0xC02)  SDR 2 IPC
```

```
sysadmin-vm:0_RP0# show controller switch vlan info

SDR
Identifier  SDR Name     VLAN           VLAN Use
-------------------------------------------------------------------------
1           sysadmin-vm  1     (0x001)  Platform EMON
2           default-sdr  1282  (0x502)  SDR 2 Platform Netflow 1
                         1298  (0x512)  SDR 2 Platform Netflow 2
                         1314  (0x522)  SDR 2 Platform Netflow 3
                         1330  (0x532)  SDR 2 Platform Netflow 4
                         1346  (0x542)  SDR 2 Platform Netflow 5
                         1362  (0x552)  SDR 2 Platform Netflow 6
                         1378  (0x562)  SDR 2 Platform Netflow 7
                         1394  (0x572)  SDR 2 Platform Netflow 8
                         1538  (0x602)  SDR 2 Platform SPP
                         1554  (0x612)  SDR 2 Platform BFD
                         1794  (0x702)  SDR 2 Third Party Applications
```



Packet capture on EPC VLANs

- EPC switch is ”visible” to the host OS i.e. Linux
- As a result, it is possible to use standard Linux utilities like tcpdump to capture packet on EPC switch VLANs.

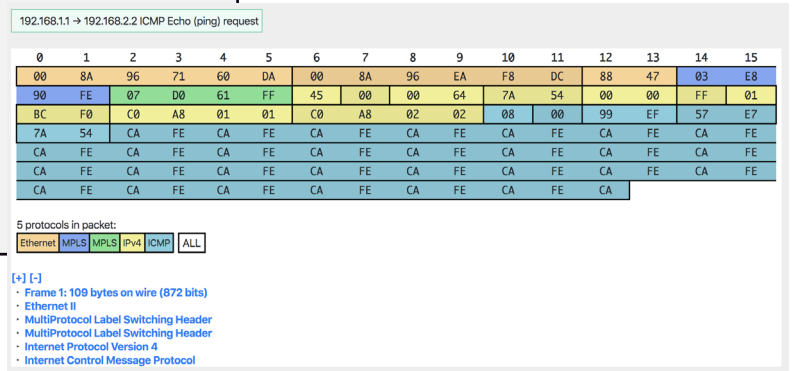
```
RP/0/RP0/CPU0:P1#attach location 0/0/cpu0

#ifconfig | grep ps
ps-inb0.1282 Link encap:Ethernet  HWaddr 4e:41:50:00:00:01
ps-inb1.1298 Link encap:Ethernet  HWaddr 4e:41:50:00:00:01
ps-inb2.1314 Link encap:Ethernet  HWaddr 4e:41:50:00:00:01
ps-inb3.1330 Link encap:Ethernet  HWaddr 4e:41:50:00:00:01

#ifconfig ps-inb1.1298
ps-inb1.1298 Link encap:Ethernet  HWaddr 4e:41:50:00:00:01
          inet6 addr: fe80::4c41:50ff:fe00:1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9702  Metric:1
          RX packets:257081718 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:112222067076 (104.5 GiB)  TX bytes:648 (648.0 B)    …
```

Netflow packet capture

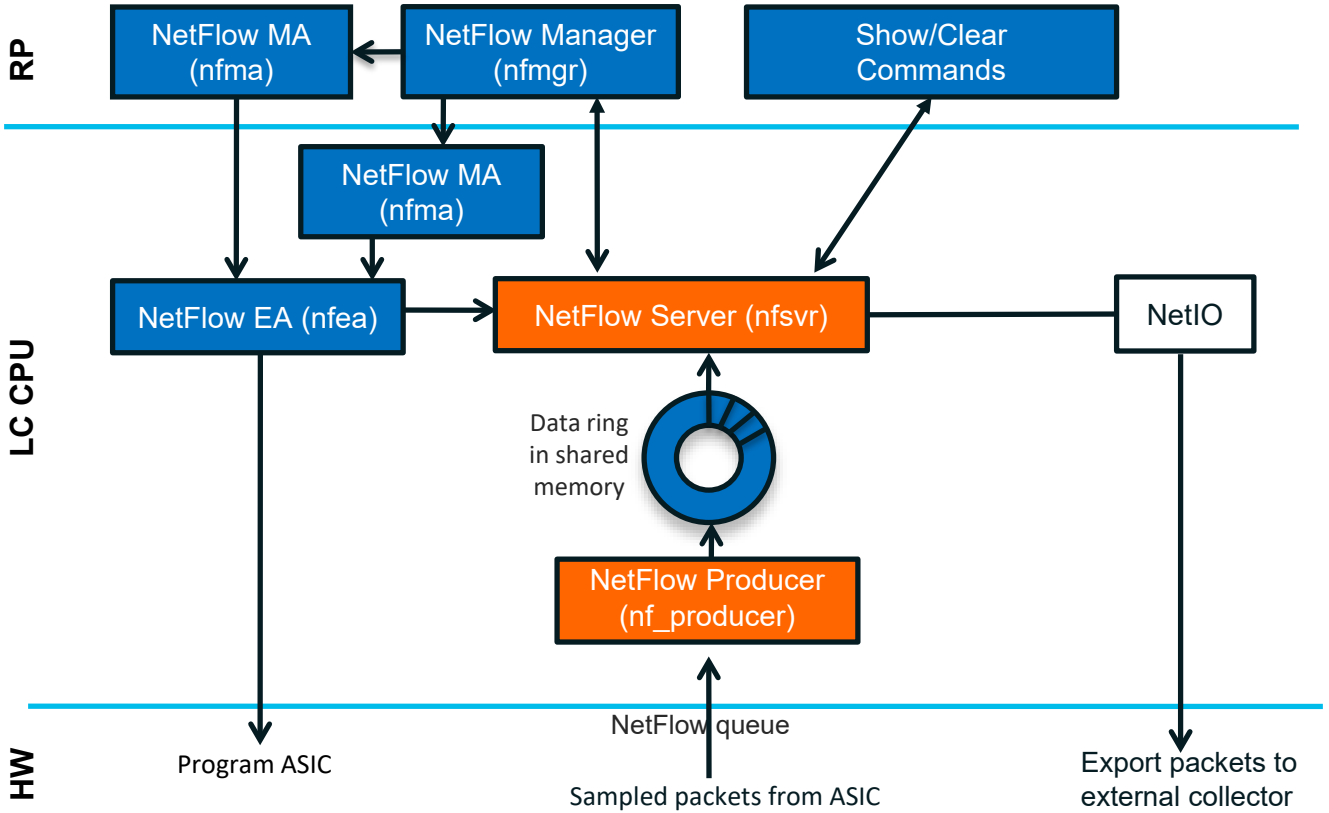
```
RP/0/RP0/CPU0:PE1#attach location 0/0/CPU0

#tcpdump -xxx -i ps-inb1.1298

10:08:53.631457 4e:41:50:00:00:12 (oui Unknown) > 4e:41:50:00:00:01 (oui
Unknown), ethertype Unknown (0x876f), length 167:
        0x0000:  4e41 5000 0001 4e41 5000 0012 876f 026c
        0x0010:  6001 7e8a 1180 7879 c000 c141 2526 0018
        0x0020:  a806 2a03 e890 10fe 0000 0000 0000 8a96
        0x0030:  7160 da00 8a96 eaf8 dc88 4703 e890 fe07
        0x0040:  d061 ff45 0000 647a 5400 00ff 01bc f0c0
        0x0050:  a801 01c0 a802 0208 0099 ef57 e77a 54ca
        0x0060:  feca feca feca feca feca feca feca feca
        0x0070:  feca feca feca feca feca feca feca feca
        0x0080:  feca feca feca feca feca feca feca feca
        0x0090:  feca feca feca feca feca feca feca feca
        0x00a0:  feca feca feca fe
```

Internal headers

Src MAC
Dst MAC
MPLS ethertype



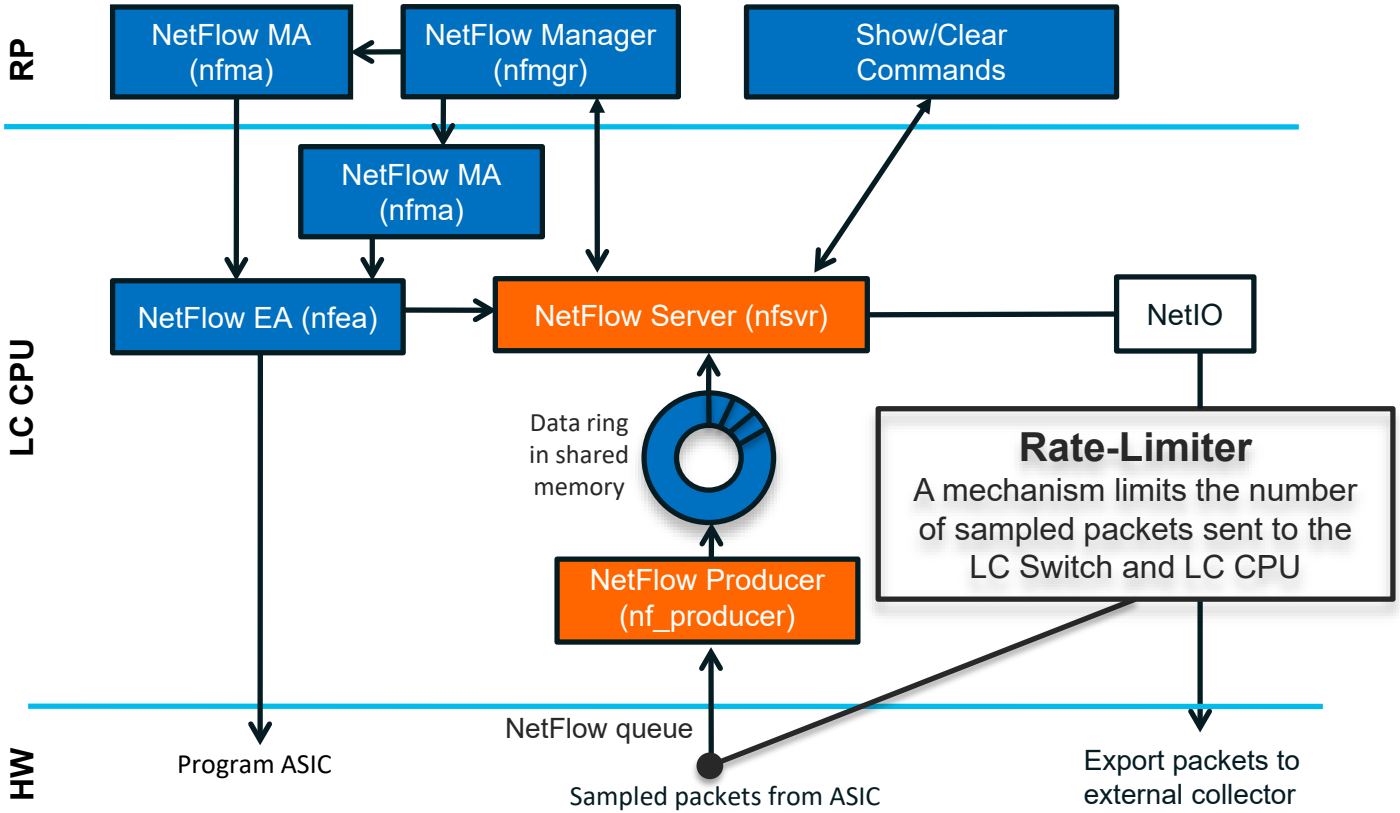
Processes Involved in Netflow Operation



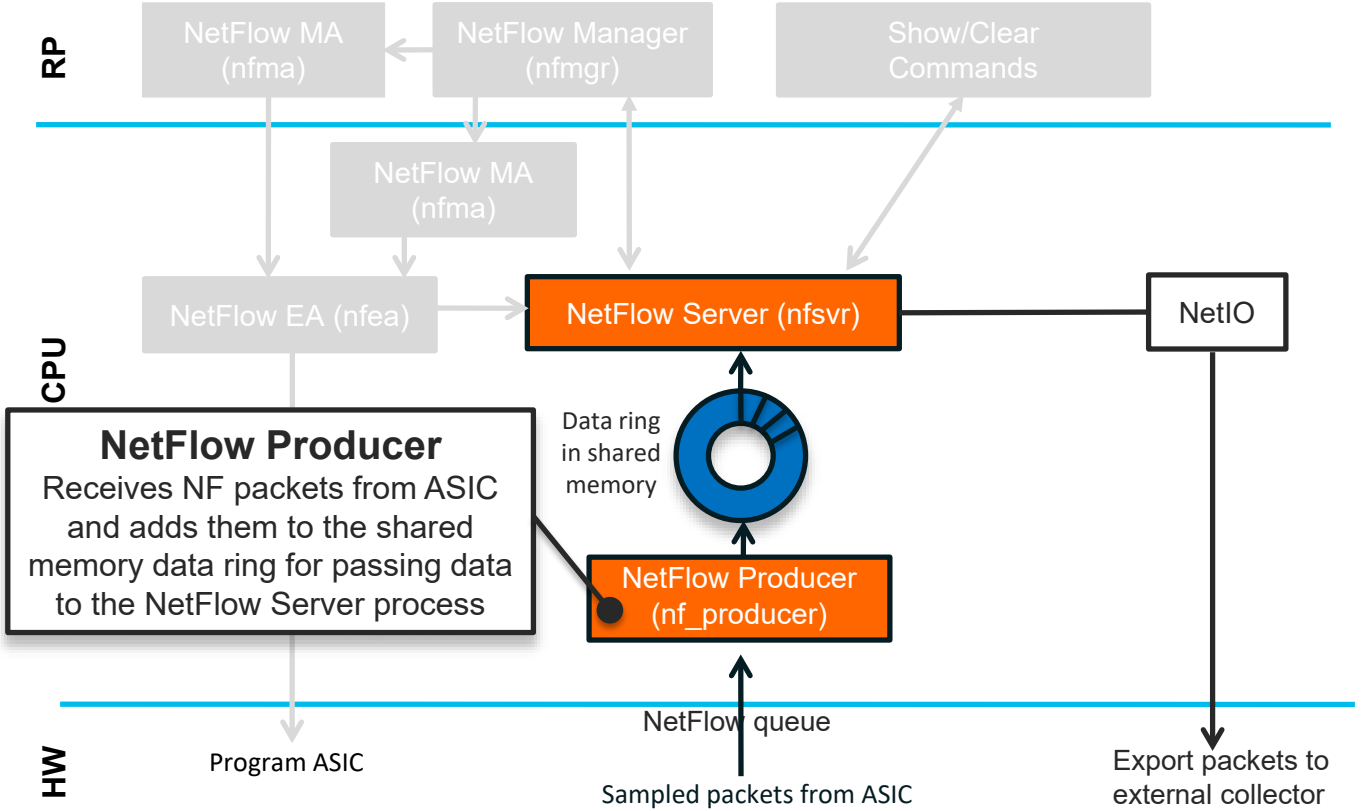
Processes Involved in Netflow Operation

```
RP/0/RP0/CPU0:P1#sh processes cpu location 0/rp0/cpu0 | i " nf"

4666      0%       0%       0% nfmgr
4769      0%       0%       0% nfma


RP/0/RP0/CPU0:P1#sh processes cpu location 0/0/cpu0 | i " nf"

4488      0%       0%       0% nfma
14343     0%       0%       0% nfea
14348     0%       0%       0% nfsvr
14358     0%       0%       0% nf_producer
```

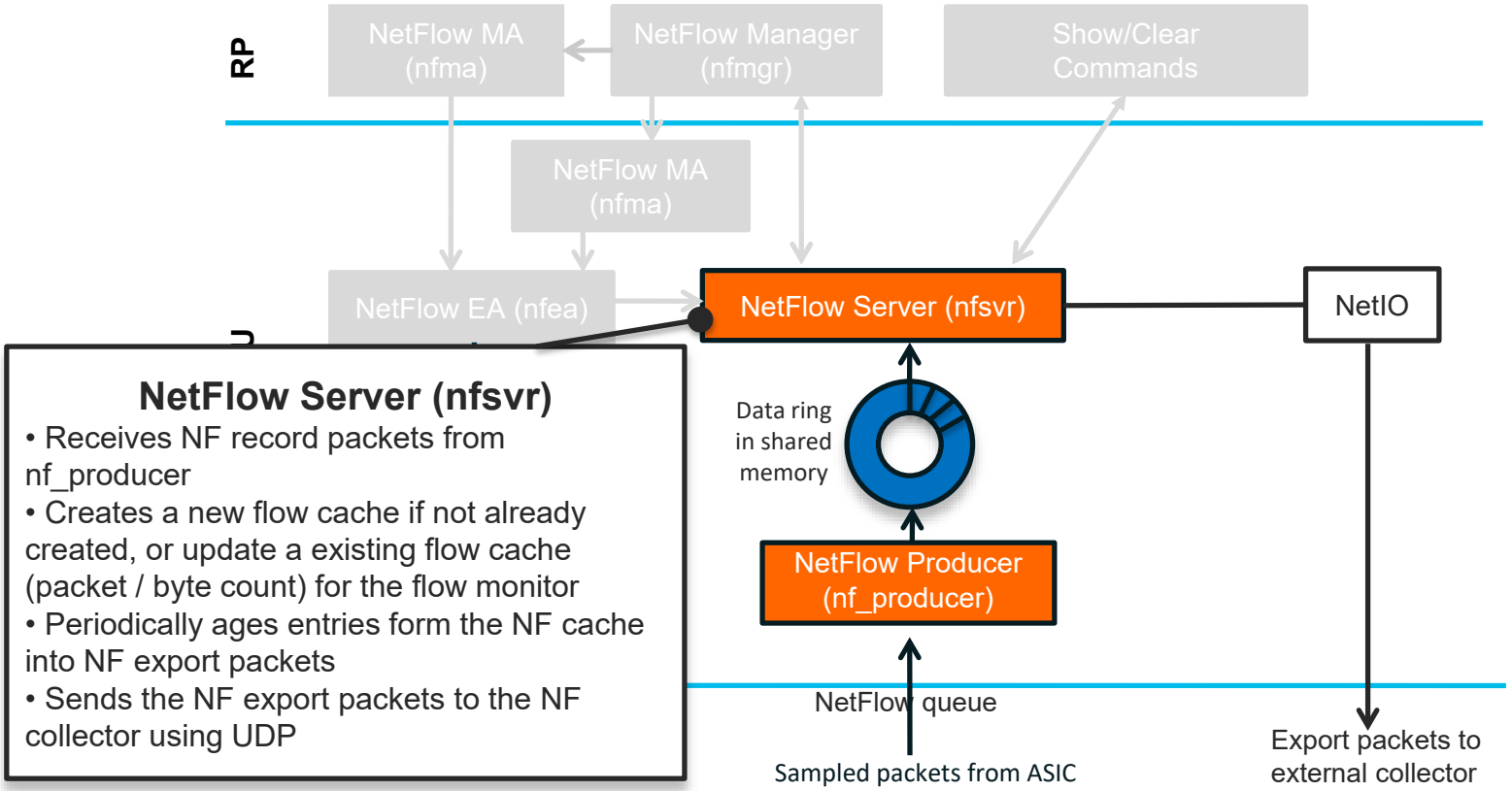
Processes Involved in Netflow Operation



Processes Involved in Netflow Operation



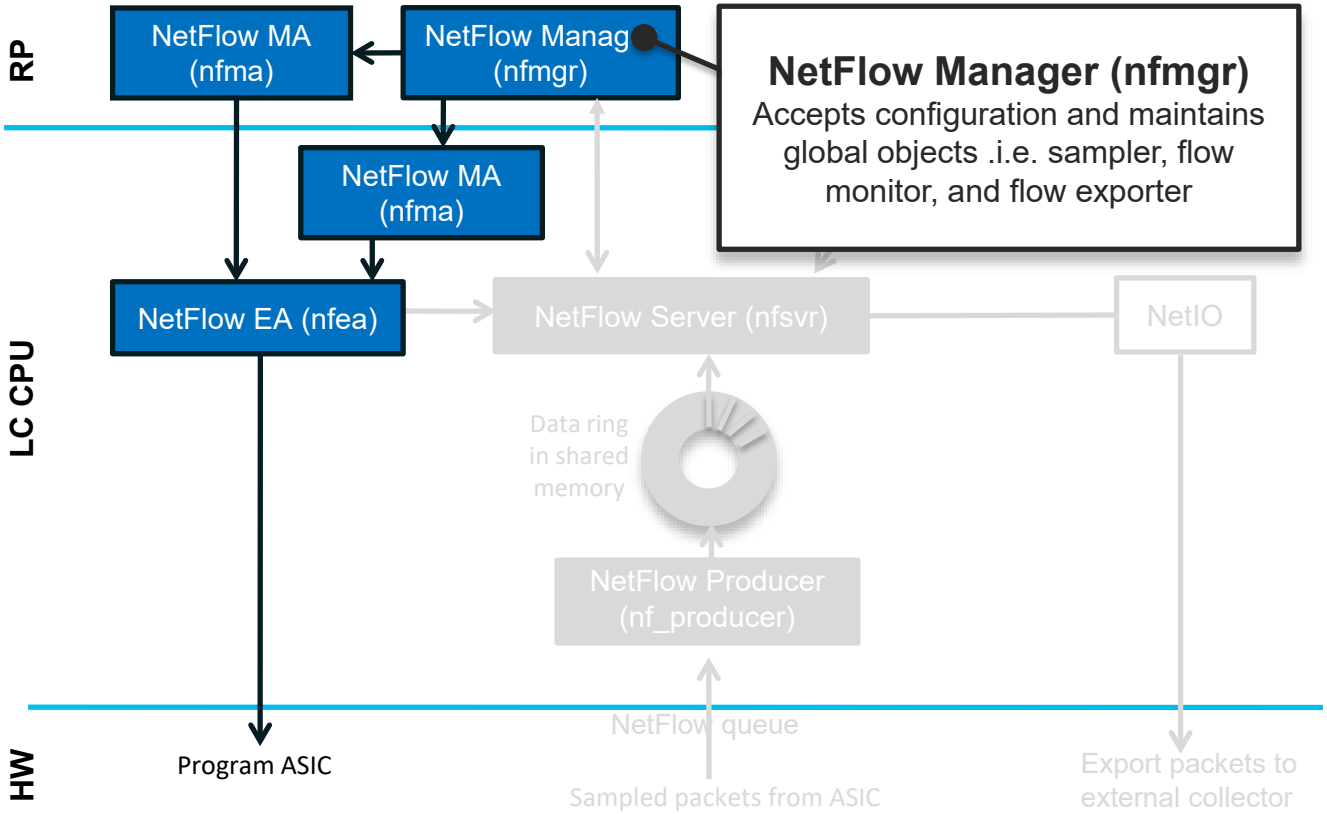
Processes Involved in Netflow Operation



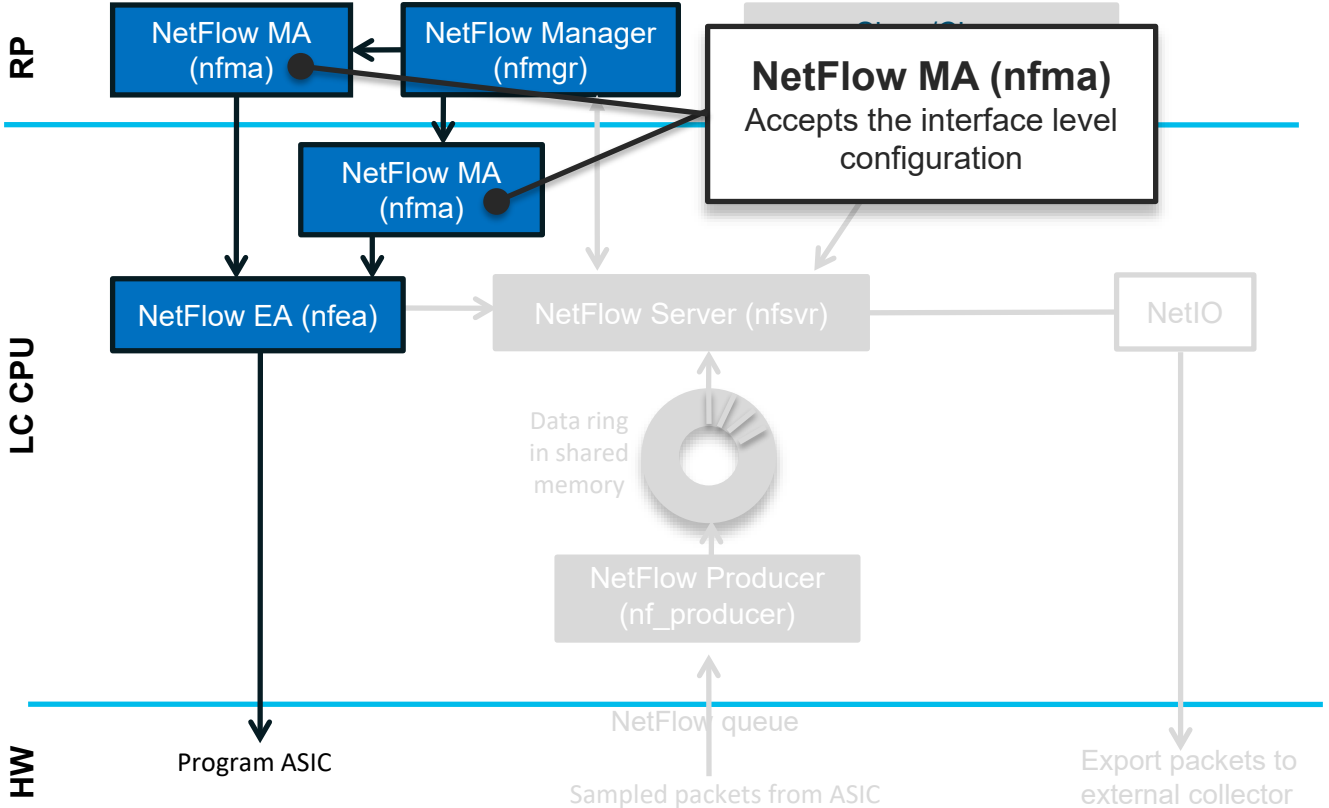
NetFlow Server (nfsvr)

- Receives NF record packets from nf_producer
- Creates a new flow cache if not already created, or update a existing flow cache (packet / byte count) for the flow monitor
- Periodically ages entries form the NF cache into NF export packets
- Sends the NF export packets to the NF collector using UDP

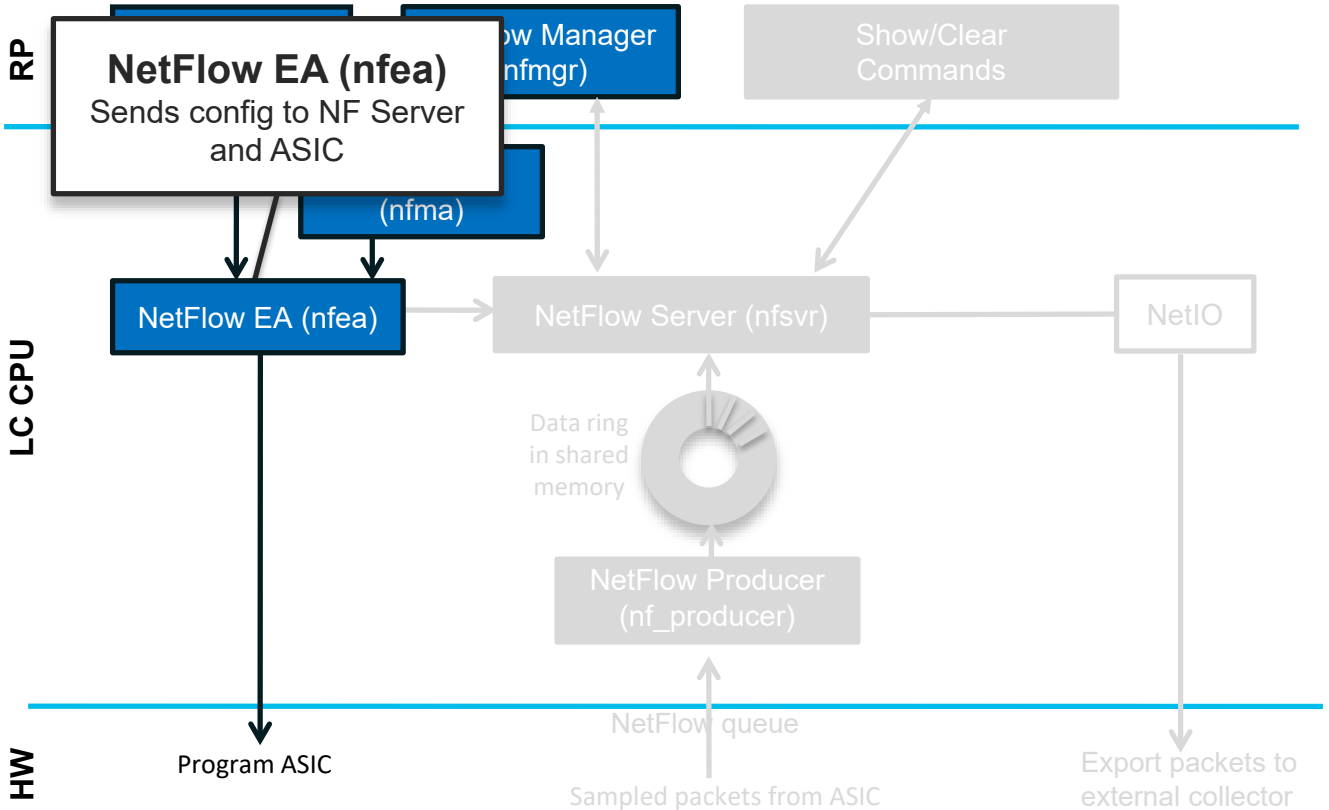
Processes Involved in Netflow Operation



Processes Involved in Netflow Operation



Processes Involved in Netflow Operation



Netflow Configuration Example

```
flow exporter-map export1
 version v9
  options interface-table
  options sampler-table
 !
 transport udp 9951
 source Loopback0
 destination 192.168.0.3
!
flow monitor-map monitor1
 record mpls ipv4-fields
 exporter export1
 cache entries 1000000
 cache timeout active 15
 cache timeout inactive 2
 cache timeout rate-limit 2000
!
sampler-map sampler1
 random 1 out-of 500
!
interface Bundle-Ether3
 flow mpls monitor monitor1 sampler sampler1 ingress
```

exporter-map config

monitor-map config

sampler-map config

Show Commands

```
RP/0/RP0/CPU0:P1#show flow monitor-map monitor1

Flow Monitor Map : monitor1
-----------------------------------------------
Id:                 2
RecordMapName:      mpls-ipv4 (6 labels)
ExportMapName:      export1
CacheAgingMode:     Normal
CacheMaxEntries:    1000000
CacheActiveTout:    15 seconds
CacheInactiveTout:  2 seconds
CacheUpdateTout:    N/A
CacheRateLimit:     2000
```

Show Commands

```
RP/0/RP0/CPU0:P1#show flow platform producer statistics location 0/0/cpu0

Netflow Platform Producer Counters:
IPv4 Ingress Packets:                    1999
IPv4 Egress Packets:                        0
IPv6 Ingress Packets:                       0
IPv6 Egress Packets:                        0
MPLS Ingress Packets:                53824976
MPLS Egress Packets:                        0
IPFIX315 Ingress Packets:                   0
IPFIX315 Egress Packets:                    0
Drops (no space):                           0
Drops (other):                              9
Unknown Ingress Packets:                    0
Unknown Egress Packets:                     0
Worker waiting:                        205359
```

Recognized and parsed IPv4/IPv6/MPLS packets

Dropped packets on the NF producer level

Show Commands

```
RP/0/RP0/CPU0:P1#show flow monitor monitor1 cache summary location 0/0/CPU0

Cache summary for Flow Monitor monitor1:
Cache size:                          1000000
Current entries:                         295
Flows added:                          184409
Flows not added:                           0
Ager Polls:                             9824
  - Active timeout                    183855
  - Inactive timeout                     259
  - Immediate                              0
  - TCP FIN flag                           0
  - Emergency aged                         0
  - Counter wrap aged                      0
  - Total                             184114
Periodic export:
  - Counter wrap                           0
  - TCP FIN flag                           0
Flows exported                        184114
```

Flows not added due to the cache size

Show Commands

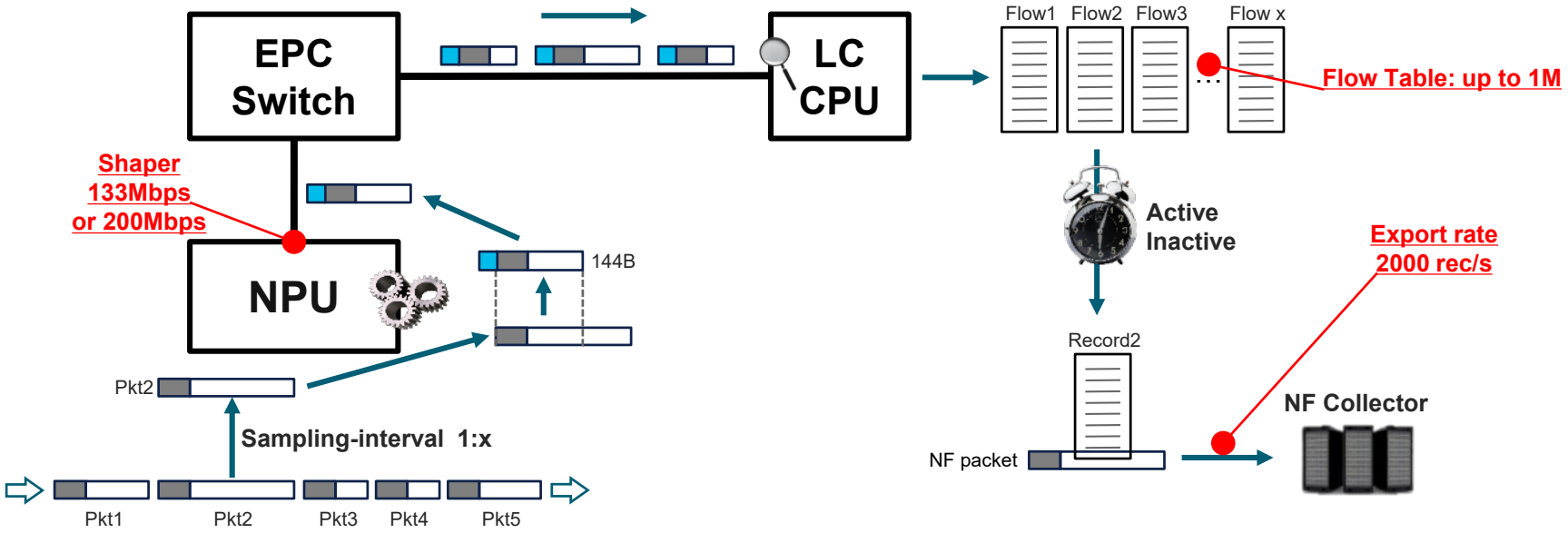
```
RP/0/RP0/CPU0:P1#show flow monitor monitor1 cache brie loc 0/0/cpu0

Cache summary for Flow Monitor monitor1:
Cache size:                         1000000
Current entries:                        295
Flows added:                         185241
Flows not added:                          0
Ager Polls:                            9868
  - Active timeout                   184687
  - Inactive timeout                    259
  - Immediate                             0
  - TCP FIN flag                          0
  - Emergency aged                        0
  - Counter wrap aged                     0
  - Total                            184946
Periodic export:
  - Counter wrap                          0
  - TCP FIN flag                          0
Flows exported                       184946

LabelType Prefix/Length      Label1-EXP-S     Label2-EXP-S     Label3-EXP-S     Label4-EXP-S     Label5-EXP-S     Label6-EXP-S
      LDP 192.168.2.22/32       16009-0-0        32003-0-1                -                -                -                -
InputInterface  OutputInterface ForwardStatus        ByteCount    PacketCount  Dir
BE3             BE1             Fwd                  87212        230          Ing

LabelType Prefix/Length      Label1-EXP-S     Label2-EXP-S     Label3-EXP-S     Label4-EXP-S     Label5-EXP-S     Label6-EXP-S
      LDP 192.168.2.22/32       16009-0-0        32020-0-1                -                -                -                -
InputInterface  OutputInterface ForwardStatus        ByteCount    PacketCount  Dir IPV4SrcAddr      IPV4DstAddr      IPV4TOS  IPV4Prot
BE3             BE1             Fwd                  45874        124          Ing 192.168.1.90     192.168.2.90     0        udp
L4SrcPort  L4DestPort
63         63
```


Show Commands

```
RP/0/RP0/CPU0:P1#show flow exporter export1 location 0/0/cpu0

Flow Exporter: export1
Flow Exporter memory usage: 5247512
Used by flow monitors: monitor1

Status: Normal
Transport:   UDP
Destination: 192.168.0.3     (9951) VRF default
Source:      192.168.0.1     (10743)
Flows exported:                              189011 (14175825 bytes)
Flows dropped:                                    0 (0 bytes)

Templates exported:                             194 (22504 bytes)
Templates dropped:                                0 (0 bytes)

Option data exported:                            16 (1066 bytes)
Option data dropped:                              0 (0 bytes)

Option templates exported:                       18 (504 bytes)
Option templates dropped:                         0 (0 bytes)

Packets exported:                            15477 (14284556 bytes)
Packets dropped:                                  0 (0 bytes)

Total export over last interval of:
  1 hour:                                     5558 pkts
                                           5124744 bytes
                                             67807 flows
  1 minute:                                    92 pkts
                                             86272 bytes
                                              1142 flows
  1 second:                                     2 pkts
                                              2412 bytes
                                                32 flows
```

Netflow Performance

- Three bottlenecks could impact the performance:
  - 133 Mbps or 200 Mbps shaper (not configurable)
  - Flow table size (up to 1M per monitor-map, configurable)
  - Export rate-limiter (default 2000 records / sec, configurable)



Netflow Shaper Per NPU

Platform / LC	#NPU / LC	Shaper / NPU	Total / LC
NCS 5501(-SE)	1xQmx	200 Mbps	Min 174Kpps
NCS 5502(-SE)	8x	200 Mbps	Min 174x8= 1392Kpps
NCS 55A1-36H(-SE)	4	200 Mbps	Min 174x4= 696Kpps
NCS 55A1-24H	2	200 Mbps	Min 174x2= 348Kpps
36x100G w/o eTCAM	6	133 Mbps	Min 115x6= 690Kpps
24x 100G w/ eTCAM	4	133 Mbps	Min 115x4= 462Kpps
18H18F w/o eTCAM	3	133 Mbps	Min 115x3= 345Kpps
24H12F w/ eTCAM	4	133 Mbps	Min 115x4= 462Kpps
36x 100G MACSEC w/o eTCAM	6	133 Mbps	Min 115x6= 692Kpps
6x 200G Coherent w/o eTCAM	2	133 Mbps	Min 115x2= 230Kpps
36x100G-SE J+ w/ eTCAM	4	133 Mbps	Min 115x4= 462Kpps

“Min”: because sampled packets can be <128B

Netflow Shaper Per NPU

- The shaper rate might be checked with the following CLI

```
RP/0/RP0/CPU0:Router#show flow platform pse policer-rate location <location>
```

- Example output:

```
RP/0/RP0/CPU0:PE3#show flow platform pse policer-rate location 0/0/CPU0

Npu id :0
Netflow Platform Pse Policer Rate:
Ingress Policer Rate:                    199 Mbps
Npu id :1
Netflow Platform Pse Policer Rate:
Ingress Policer Rate:                    199 Mbps

RP/0/RP0/CPU0:NCS5508#show flow platform pse policer-rate location 0/0/CPU0
Mon Jan 27 15:42:47.664 CET
Npu id :0
Netflow Platform Pse Policer Rate:
Ingress Policer Rate:                    133 Mbps
Npu id :1
Netflow Platform Pse Policer Rate:
Ingress Policer Rate:                    133 Mbps
```

Netflow Shaper Per NPU

- To check if shaper is dropping NetFlow samples, drop counters should be checked under VOQ 32 / COS3, which is used for a Netflow traffic punt:

```
RP/0/RP0/CPU0:PE3#show controllers npu stats voq base 32 instance 0 location 0/0/CPU0

Asic Instance     =            0
VOQ Base          =           32
       ReceivedPkts    ReceivedBytes   DroppedPkts     DroppedBytes
-------------------------------------------------------------------
TC_0 = 878025          228648068       0               0
TC_1 = 0               0               0               0
TC_2 = 0               0               0               0
TC_3 = 244205          27635546        0               0
TC_4 = 666393          4520523722      0               0
TC_5 = 0               0               0               0
TC_6 = 0               0               0               0
TC_7 = 0               0               0               0
```

- Note: Prior to the XR 6.3.x VOQ 24 & COS2 were used.

Sampling-interval ?

- There is always discussion about sampling rate
- Sample traffic bandwidth is more useful
- Multi-parameter equation
  - Number of ports per NPU
  - Average bandwidth per link
  - Average packet size
  - Sampling-interval
- Each sample is 128B long, we add internal headers and we transport 144B
  - 133Mbps / 144B = 115KPPS per NPU
  - 200Mbps / 144B = 174KPPS per NPU
  - It's possible to predict the BW of sampled traffic between NPU and EPC switch
  - Some packets may be smaller than 128B but let's consider the worst-case scenario

Sampling-interval ?

- Sampled packet size: 144 Bytes (128 + headers)
- Example 1:
  - Number of ports per NPU: 6 x100G ports
  - Average bandwidth per link: 100G line rate
  - Average packet size: 350B (it's a worst case, since some packets sampled will be smaller than 128B, they will use less
  - Sampling-rate: 1:4000
  - 600,000,000,000 bps / (350 x 8) / 4000 x (144 x 8) = 61.7Mbps < 133Mbps
- Example 2:
  - Same parameters with sampling-rate: 1:1000
  - 600,000,000,000 bps / (350 x 8) / 1000 x (144 x 8) = 246.8Mbps > 133Mbps

https://xrdocs.io/ncs5500/tutorials/2018-02-19-netflow-sampling-interval-and-the-mythical-internet-packet-size/

Netflow support in 6.5.x

- In 6.5.x, we support:
  - Ingress Netflow on L3 interfaces (Physical or Bundled)
  - Ingress Netflow on L3 sub-interfaces
  - Ingress Netflow on L2 interfaces (Physical or Bundled)
  - Ingress Netflow on L2 sub-interfaces

    Note: it needs to be IPv4 / IPv6 / MPLS traffic over L2 interfaces. Pure L2 traffic is not captured.
  - IPFIX

- We DON'T support:
  - Egress Netflow (in roadmap)

Increasing NF Records Rate-limiter

- We may would like to extend the number of records generated
  - Default of 2000 pps is not satisfactory

```
flow monitor-map monitor1
 cache timeout rate-limit 20000
```

- Misconception:
  - Cache timeout RL is driving the speed the cache is emptied and not the opposite
  - If the RL is not fast enough, active/inactive entries may stay longer than their timer
  - It could lead to a cache table full, reflected by the ”Flow dropped” counter

```
RP/0/RP0/CPU0:P1#show flow exporter export1 location 0/0/cpu0
…
Source:       192.168.0.1      (10743)
Flows exported:                                  189011 (14175825 bytes)
Flows dropped:                                        12 (1392 bytes)
…
```

Increasing NF Records Rate-limiter

- Impact on system
  - Process nf_producer and nfser executed in Line Card CPU will be more solicited
  - Example below shows nfsvr at 12% CPU load with 50.000 records per second

```
RP/0/RP0/CPU0:P1#sh processes cpu location 0/0/cpu0 | i " nf"

…
14348     12%      12%       12% nfsvr
…
```

Netflow Full Packet Capture

- Starting from XR release 7.0 we can capture full packet
- This feature captures the exact packet size of the ingress Netflow packet
- Usually Netflow only reports MPLS packets with IPv4 and IPv6 payloads. Netflow will not report the MPLS packets with L2 payload (e.g. L2VPN packets). When a MPLS packet with underlying L2 payload and a payload destination MAC address starting with the number 6 or 4 is received without any control word, the packet can get wrongly decoded as MPLS + IP packet, and inaccurate packet size can get reported to the collector.
- When this feature is enabled, the MPLS packets with all payload types will be reported to the collector, though only the IP payload be decoded. The packet size will be reported correctly in all cases. The use of control word is still recommended to avoid misclassifying L2 payload as IPv4 or IPv6 payload.
- **Caveat:** There may a netflow performance impact due to this feature.
- In order to use this feature, the command “hw-module profile netflow fpc-enable location <loc>” has to be applied to the LC and card must be reloaded.

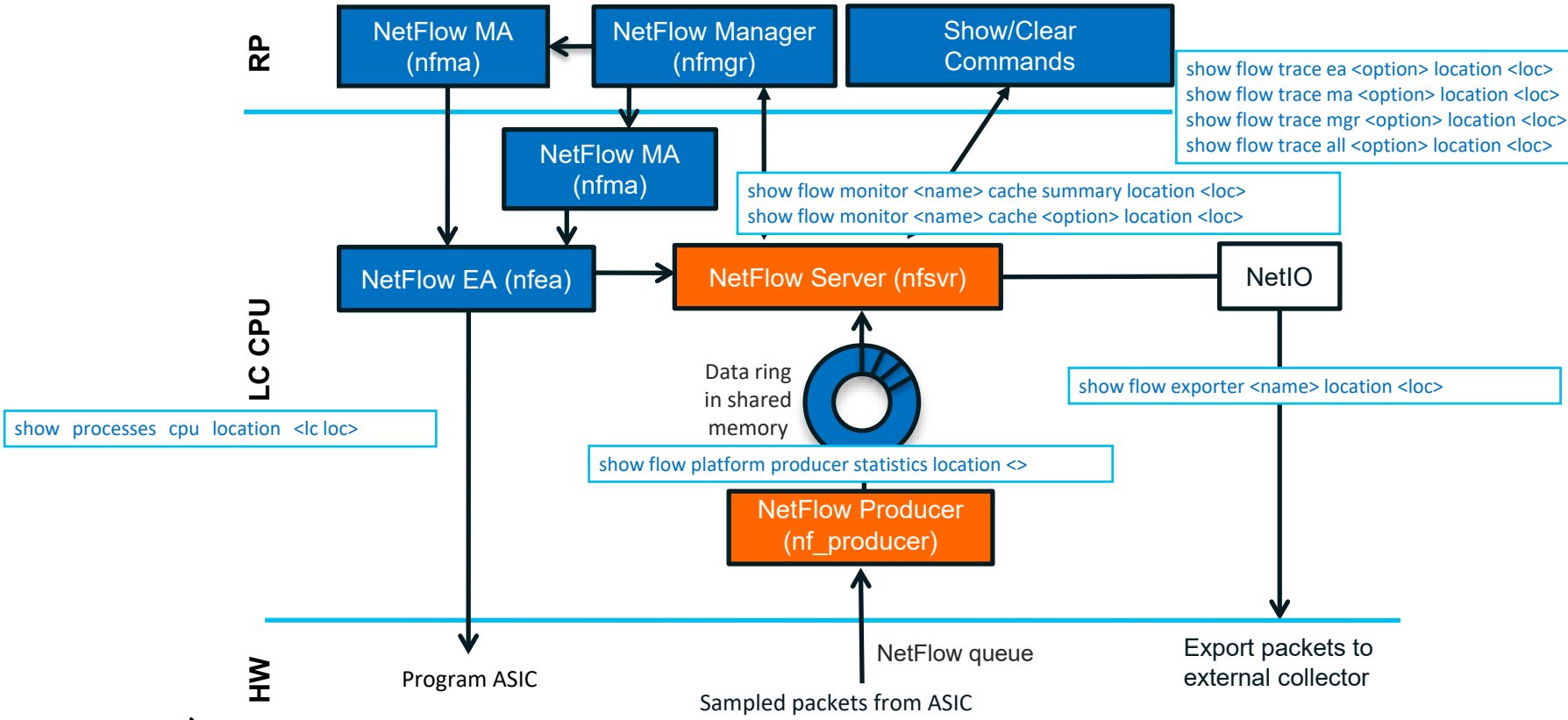
```
hw-module profile netflow fpc-enable location <location>
```


Commands summary

For Reference

show run flow monitor-map <>
show run flow exporter-map <>
show run sampler-map <>
show run interface <>

show processes cpu location <rp loc>



show flow trace ea <option> location <loc>
show flow trace ma <option> location <loc>
show flow trace mgr <option> location <loc>
show flow trace all <option> location <loc>

show flow monitor <name> cache summary location <loc>
show flow monitor <name> cache <option> location <loc>

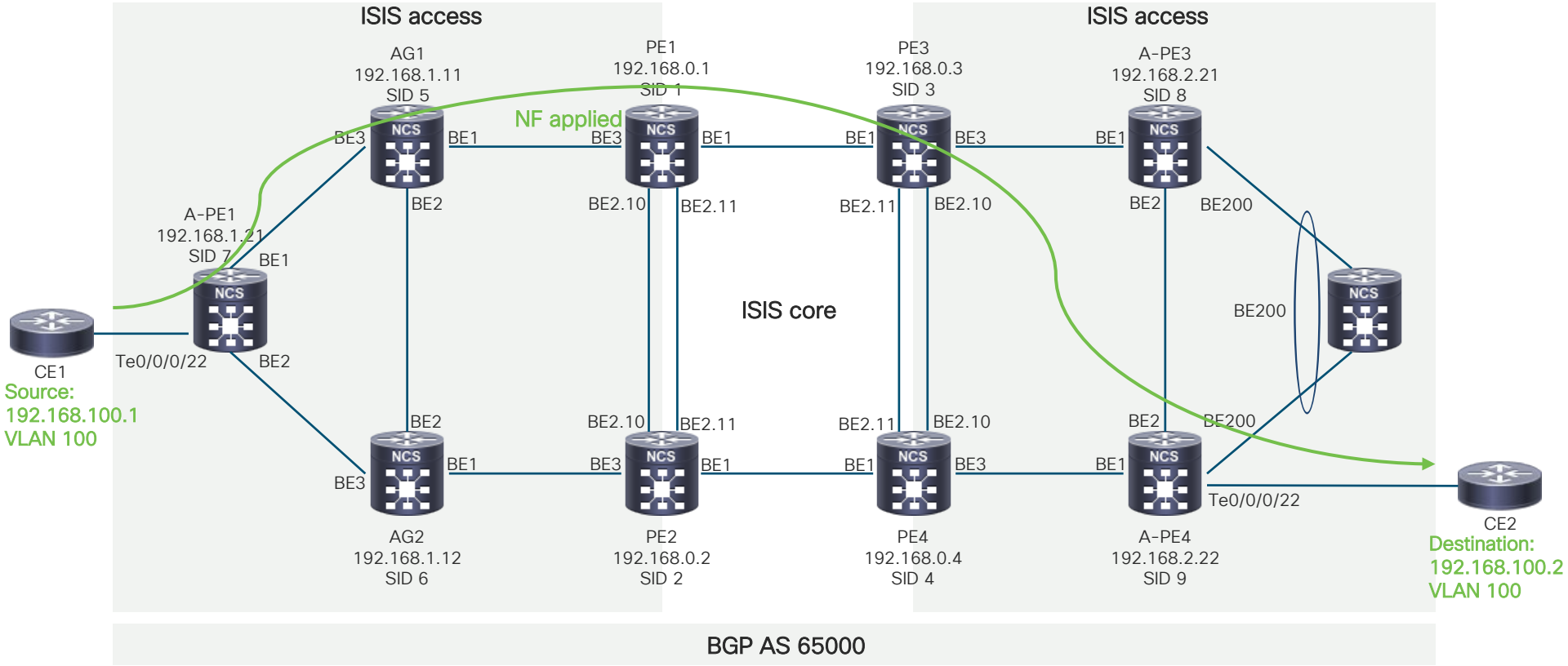
show flow exporter <name> location <loc>

show processes cpu location <lc loc>

show flow platform producer statistics location <>

Netflow troubleshooting demo

Network topology and legend



https://youtu.be/plS-Q-NsZIE

Reported issue

- There is L2 traffic with IPv4 payload being sent from source to the receiver and, hence L2VPN configured between A-PE1 and A-PE4. Netflow is configured in ingress direction on interface BE3 on P1 to record mpls ipv4-field. There is NF collector in the network reachable over IPv4 address 192.168.0.3. However, there is no single entry for the L2 traffic from CE1 towards CE2 we suppose to account for.
- We must troubleshoot this issue and find out the root cause of the issue and provide solution.
- We must isolate if the issue related to Netflow collector, router P1 configuration, the network itself or anything else.

Summary

- Unicast forwarding troubleshooting
- Multicast forwarding troubleshooting
- Netflow troubleshooting on NCS5500

Please remember:

Structure is the key to successful troubleshooting...

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.

cisco Live!

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

Thank you

You make **possible**