

Advanced Troubleshooting Nexus 7000 Series Switches

Presented by:

Alex Lv - ENGINEER.CUSTOMER SUPPORT, Data-Center

Aug 26th, 2015

Open Day Session Goal

- **Who For:**
 - Those implementing or have existing Nexus 7/77K platforms in their network.
 - New to, or already familiar with NXOS.
 - Anyone else who just wants to hang out and learn 😊
- **Session Goal**
 - Troubleshooting methodology is the same, no matter the platform...
 - **Where and What** (where to look, and what to use)
 - Quick and easy data collection **Cheat Sheets**
 - **Faster root cause** = decreased impact + eliminate re-occurrence

Agenda

- Strategy & Tools
- Control Plane Troubleshooting
- Data Plane Troubleshooting
- Q&A / Conclusion

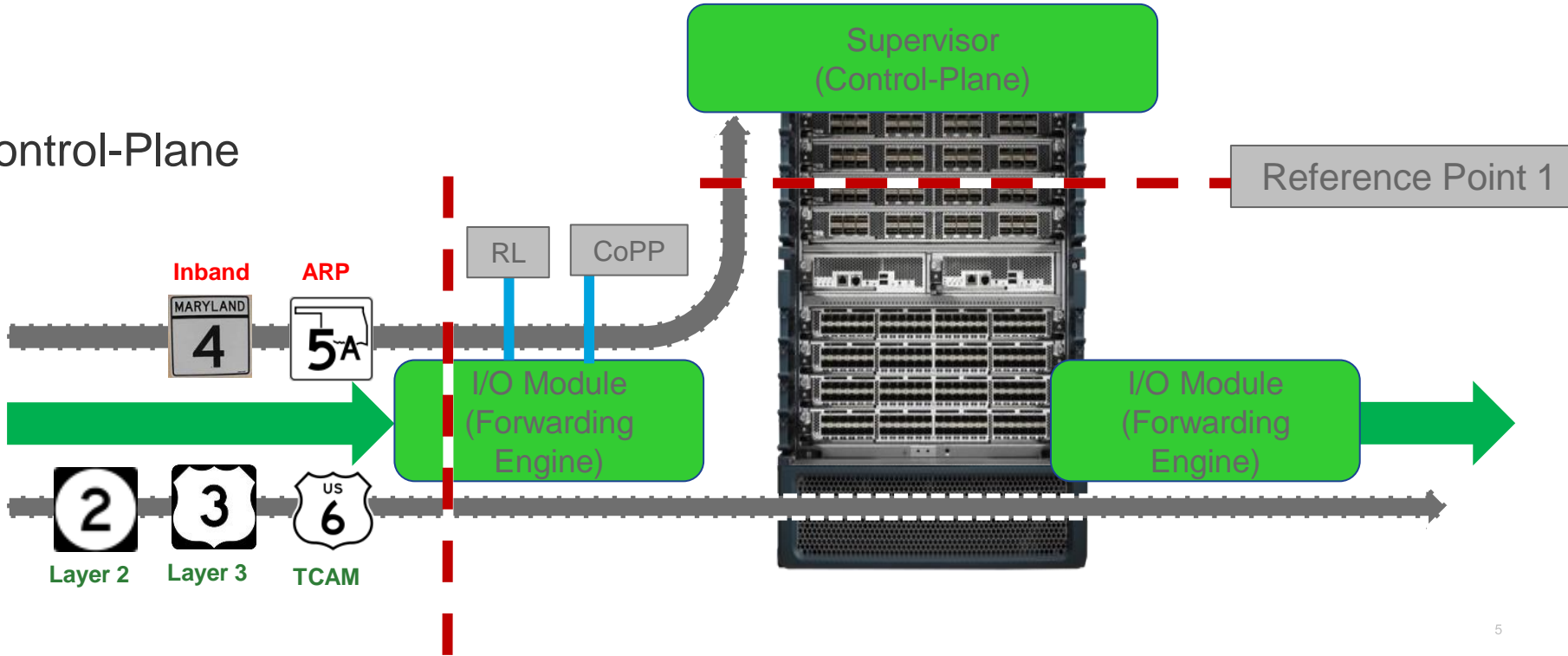
Agenda

- Strategy & Tools
- Control Plane Troubleshooting
- Data Plane Troubleshooting
- Q&A / Conclusion

Strategy

System, Data-Plane, and Control-Plane

Control-Plane



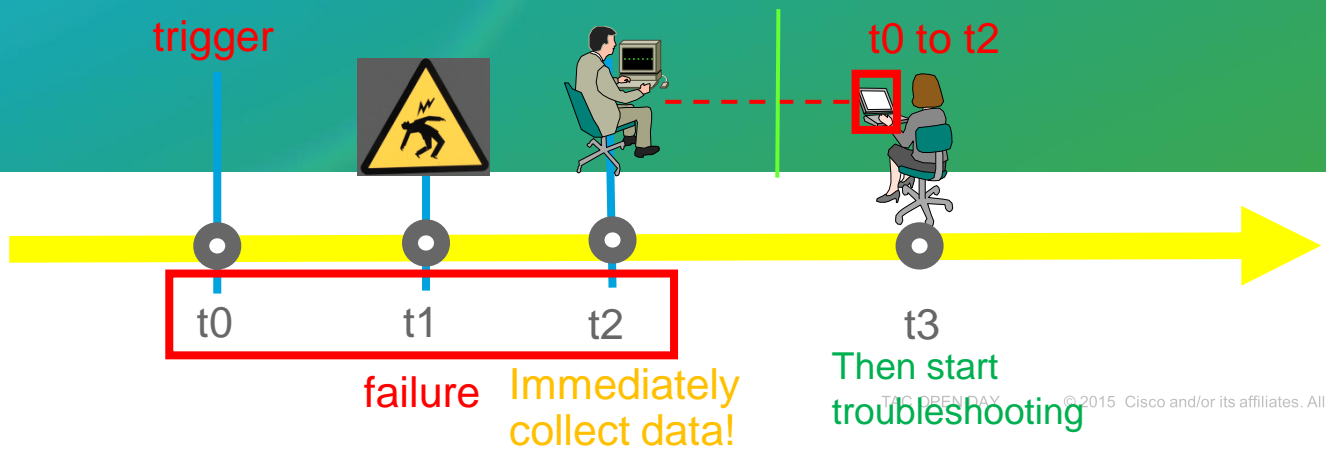
Data-Plane

Reference Point 2

NX-OS Troubleshooting Tools

- **Granular show tech-support and CLI Filtering**
- **Logging Capabilities**
- **Debug Capabilities**
- **Core Files (Process Restarts)**
- **Module Troubleshooting (OBFL & GOLD)**
- **Scripting (Scheduler & EEM & Python)**
- **Packet Capture Tools (SPAN / ELAM / Ethalyzer)**

Granular Show Tech-Support and CLI Filtering



Granular Show-Techs

- Every feature has its own `show tech <feature>`
- Issues frequently involve multiple features so good practice to collect generic `show tech detail` + `show tech <feature>`
- Collect show techs as early as possible

```
Nexus# show tech ?
aclmgr          ACL commands
aclqos          Show information for aclqos technical support
adjmgr          Display Adjmgr information
```

- `sh tech` can be over **150MB+** and can take several minutes...

```
Nexus# show tech details
---- show tech-support details ----
.....
```

Alternatives... ?

Granular Show-Techs cont...

Alternatives:

1. Tac-Pac: Still takes 4-6 minutes, but automatically zips file reducing size
2. Binary-Tech: **Done in seconds** and compresses into .tar – faster and reduces file size
 - CLI Issued from default VDC, and takes tech-support for ALL VDCs

```
Nexus# tac-pac bootflash:tac-pac
Show tech details will take 4-6 minutes to complete. Please Wait ...
Nexus# dir bootflash: | eg -i tac
24521266 Mar 14 04:27:07 2015 tac-pac
```

Compressed tech,
but still 4-6 minutes

```
Nexus# show tech-support all binary bootflash:
Nexus# dir bootflash: | eg -i binary
65208320 Mar 14 03:13:02 2015 binary_show_tech_all_03_14_2015_03_12_33HRS.tar
```

Binary format so
must be parsed by
TAC

CLI Filtering and Redirection

- Re-direct show commands to a file (can also append multiple cmds)

```
Nexus# show ip route > bootflash:ip-route.txt
```

```
Nexus# dir bootflash: | eg -i route
```

```
4873 Jan 26 21:09:29 2015 ip-route.txt
```

```
Nexus# show ip route vrf all >> bootflash:ip-route.txt
```

```
Nexus# dir bootflash: | eg -i route
```

```
10385 Jan 26 21:17:11 2015 ip-route.txt
```

“>” Creates a new file

“>>” Appends to existing file

- NXOS provides unix CLI & supports regex (to filter specific output)

```
Nexus# sh ver | ?
```

```
egrep Egrep - print lines matching a pattern
```

```
grep Grep - print lines matching a pattern
```

```
head Display first lines
```

```
last Display last lines
```

```
Nexus# show logging last 1
```

```
Mar 22 15:40:13 N7K-1 %BGP-5-MEMALERT: bgp-1 [3439] BGP memory status changed from OK to Minor Alert
```

What commands are available



- Quick way to find what commands are avail for what features.

```
Nexus# show tech ospf | in `show`  
`show running-config ospf`  
`show running-config rpm`  
`show ip ospf internal event-history cli`  
`show ip ospf vrf all`  
`show ip ospf ha vrf all`
```

```
Nexus# sh cli syntax | in ospf  
(2511) show ip ospf [ <tag> ] interface brief [ vrf { <vrf-name> | <vrf-known-name> |  
(2512) show ip ospf [ <tag> ] interface [ <interface> | vrf { <vrf-name> | <vrf-known-name>  
(2513) show ip ospf [ <tag> ] virtual-links brief [ vrf { <vrf-name> | <vrf-known-name> |  
(2514) show ip ospf [ <tag> ] virtual-links [ vrf { <vrf-name> | <vrf-known-name> |  
(2515) show ip ospf [ <tag> ] sham-links [ brief ] [ vrf { <vrf-name> | <vrf-known-name> |
```

Logging Capabilities



Logging

NX-OS:
Build in
Flight Recorder

Logging Capabilities: Log Buffer

Common Mistake About Logging:

- ``logging level <feature> <level>`` Does not cause feature to print messages `at` that level.
- Tells system to print only messages `of` that level. Think of it as a “print threshold”
- Example below (ETHPM default logging level = “5”)

```
Nexus(config)# int e 3/1
Nexus(config-if)# shut
2015 Jan 25 21:42:07 Nexus %ETHPORT-5-IF_DOWN_ADMIN_DOWN: Interface Ethernet3/1 is down (Administratively
down)
```

Change the level to 3, messages are no longer printed:

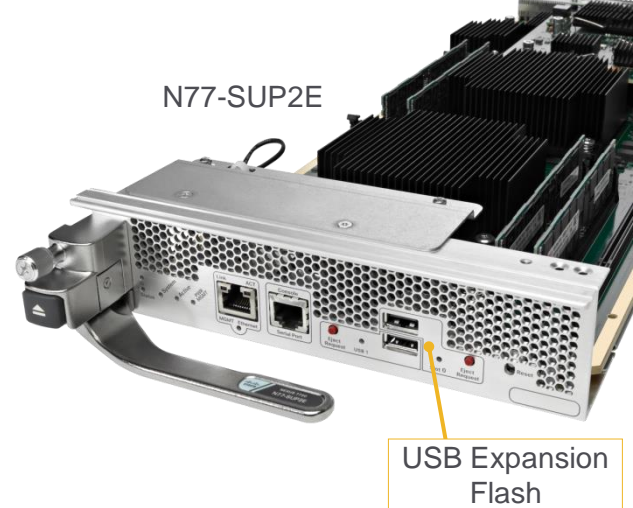
Can severely hinder troubleshooting

```
Nexus(config-if)# logging level ethpm 3
Nexus(config)# int e 3/1
Nexus(config-if)# no shut
Nexus(config-if)# sh log last 1
2015 Jan 25 21:42:07 Nexus %ETHPORT-5-IF_DOWN_ADMIN_DOWN: Interface Ethernet3/1 is down (Administratively
down)
```

Same timestamp as previous message. No new message logged

Logging Capabilities: Logflash

- Logflash is an 8GB **persistent** storage location.
- Provides storage for info such as: syslog messages, debug output, core file
- Sup1 = compact flash
- Sup2/2E = USB flash drive
- Should **ALWAYS** be inserted into supervisor. Syslog alerts when it's not



```
2014 Sep 1 13:08:56 N7K-18 %USBHSD-1-ABSENT_ALERT: logflash: not present, Please ensure logflash is inserted to save debug logs
```

- If not recognized, try formatting

```
Nexus# format logflash:
```

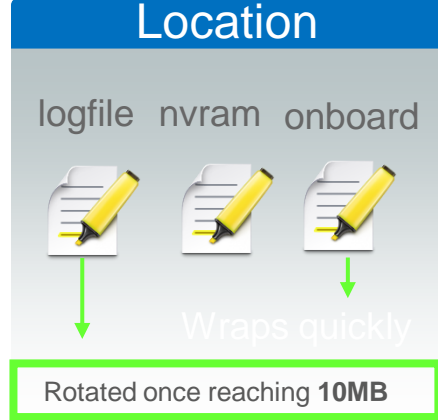
```
=====
NOTE: Formatting logflash can take over 60s. Please be patient.
=====
```

N7K-SUP2/N7K-SUP2E



Logging Capabilities: Logflash cont...

- `show logging log-file` standard logging messages but **lost on reload**
- Logflash allows for persistent storage – **survives reloads**



```
Nexus# sh clock
21:19:03.878 UTC Fri Jan 23 2015
Nexus# sh ver | in uptime
Kernel uptime is 16 day(s), 2 hour(s), 45 minute(s), 59 second(s)

Nexus# show file logflash://sup-active//log/messages
2010 Jan 1 14:05:54 %IDEHSD-2-MOUNT: logflash: online
2010 Jan 1 14:06:07 %MODULE-5-ACTIVE_SUP_OK: Supervisor 6 is active
2010 Jan 1 14:06:07 %PLATFORM-5-MOD_STATUS: Module 6 current-stat

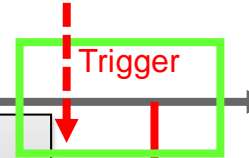
Nexus# dir logflash://sup-standby/vdc_3/log/messages
219040 Jul 16 20:51:25 2012 vdc_3/log/messages
```

System rebooted
~Jan 2015

We can still see
messages from Jan
2010

Can read non-default
vdc's and standby sup
logs

Logging Capabilities: Accounting Log



- `show accounting log` allows us to see all **'config' commands**

```
Nexus# show accounting log
Fri Nov 16 16:51:20 2012:type=update:id=172.18.118.33@pts/0:user=mesau:cmd=switchto ; configure terminal ;
interface Vlan143 (SUCCESS)
Fri Nov 16 16:51:20 2012:type=update:id=172.18.118.33@pts/0:user=mesau:cmd=switchto ; configure terminal ;
interface Vlan143 ; no shutdown (SUCCESS)
```

- `terminal log-all` adds logging of **'show' commands** as well

```
Nexus(config)# terminal log-all
Nexus(config)# show accounting log all
2:user=mesau:cmd=switchto ; show version internal build-identifier (SUCCESS)
Sat Jan 26 21:41:28 2015:type=update:id=10.82.245.163@pts/2:user=mesau:cmd=switchto ;
show accounting log all | eg build- (SUCCESS)
```

- Accounting logs also have **persistent** storage on Logflash:

```
Nexus# dir logflash://sup-active/vdc_1
130557 Jan 26 21:46:12 2015 accounting_log
```


Debug Capabilities



Debug Capabilities: Event-Histories

- Event-histories - are **“running debugs”** on by default (per vdc / per feature)
- No more waiting for maintenance windows to debug 😊
- No CPU Impact

```
Nexus(config)# show ip pim event-history join-prune
```

```
5) Event:E_DEBUG, length:61, at 61372 usecs after Sat Mar 14 20:18:24 2015  
  [116] : Received Hello from 10.0.104.2 on Vlan104, length: 30
```

```
6) Event:E_DEBUG, length:113, at 613090 usecs after Sat Mar 14 20:18:22 2015  
  [116] : iod = 49 - Send Hello on Vlan1510 from 10.15.10.252, holdtime: 105 secs, genID: 0xdc8661f, dr-  
priority: 1
```

```
Nexus(config)# sh ip eigrp event-history packet
```

```
IP-EIGRP packet events for AS 50  
2015 Mar 14 18:11:50.408038 eigrp 50 [7816]: : Send ACK 10.0.102.252->10.0.102.2 Vlan102 len:20 seq:0  
ack:6624 flg:0
```

Debug Capabilities: Debug Log-File

- Debugs also still available
- No more concern about printing debugs to terminal and CPU impact
- Can be directly logged to a logfile and viewed immediately

```
Nexus# debug logfile ospfl
Nexus# debug .....
Nexus# dir log:
 1261623   Jan 28 16:24:16 2015  messages
   18628   Jan 24 19:12:22 2015  ospfl
   16939   Jan 24 19:07:23 2015  startupdebug

Nexus# show debug logfile ospfl
2015 Jan 24 19:10:12.801652 ospf: 1 [12275] (default) Nbr 10.10.10.4 FSM start: old state INIT, event
HELLORCVD
```

- ‘`undebug all`’ command disables all existing debugs.

Debug Capabilities: Debug Filter

- Debug-filter allows more granularity
- Helpful when debugging specific packets, protocols, subnets, etc.
- Able to apply multiple filters at once.

```
Nexus# debug-filter pktmgr interface e4/1
Nexus# debug-filter pktmgr dest-mac 0100.5e00.000d
Nexus# debug pktmgr frame
```

I've set up filters for PIM pkts on a specific interface

```
Nexus# show debug-filter all
debug-filter pktmgr dest-mac 0100.5E00.000D
debug-filter pktmgr interface Ethernet4/1
```

```
2013 Jan 27 20:12:19.447962 netstack: In 0x0800 72 7 001f.27c8.8c00 -> 0100.5e00.000d Eth4/1
2013 Jan 27 20:12:24.996482 netstack: Out 0x0800 64 6 64a0.e744.3942 -> 0100.5e00.000d Eth4/1
2013 Jan 27 20:12:48.607940 netstack: In 0x0800 72 7 001f.27c8.8c00 -> 0100.5e00.000d Eth4/1
```

```
Nexus# show ip pim int b | count
25
```

25 PIM interfaces, but only printing debugs for e4/1

Debugging Tools Summary: What & When



Debugging Tool	Status	Storage	When Most Useful
Event-Histories	Default	non-persistent memory <code>show ip pim event-history</code>	Readily available: quick and easy access to debug level information
Debug Logfile + Debug-Filter	Must be configured	Persistent Storage <code>dir log:</code>	Know what you're looking for: Want to make data-collection more granular (common in large scale setups).

Core Files (*Process Restarts*)



Core Files

- Core Files are created when a process restarts
- Knowing **what, where & when** is key

```
2015 Jan 17 22:13:26 Nexus %SYSMGR-2-SERVICE_CRASHED: Service "__inst_001__ospf" (PID 6219) hasn't caught signal 6 (core will be saved)
```

```
2015 Jan 27 00:30:18 Nexus VDC-1 %$ %SYSMGR-SLOT8-2-SERVICE_CRASHED: Service "mtm" (PID 1600) hasn't caught signal 6 (core will be saved).
```

- Processes run on Supervisors and Linecards
- Helpful to then gather `show-tech + show tech ospf`

Core Files cont...

- Copy the core to remote server. Keeping in mind what slot the core is on.

```
Nexus# sh cores vdc-all
```

VDC	Module	Instance	Process-name	PID	Date (Year-Month-Day Time)
2	5	1	ospf-1	6219	2015-01-27 22:13:26
3	5	1	netstack	6919	2015-01-27 23:32:46

```
Nexus# dir logflash://module-5/vdc_3/core
```

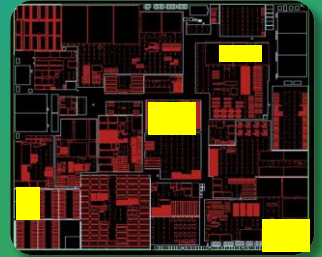
```
10052989 Jan 27 23:33:18 2015 1359329598_0x509_netstack_log.6919.tar.gz
```

```
Nexus# copy core: ?
```

```
core: Enter URL "core://<module-number>/<process-id>[/instance-num] "
```

If for some reason
show cores is empty,
always check
persistent storage
logflash

OBFL & GOLD (Module Troubleshooting)



OBFL (On-Board Failure Logging)

- On-Board Failure Logging provides **persistent storage**
- Primarily used for Module troubleshooting
 - Insertion & Removal times
 - Reset Reasons
 - Error counters & packet drops,
 - ISSU Info & LC CPU info/issues
- Info is tied to LC, so move LC to new chassis = still access OBFL info
- ``show logging onboard`` & ``show tech mod <x|all>``
- Let's look at a few examples:

OBFL – Practical Example 1

```
Nexus# show logging onboard mod 3 exception-log  
2015-03-16 20:02:15
```

Timestamp command
is run – Mar 2015

```
Exception Log Record : Thu Aug 2 20:55:32 2012 (790410 us)
```

Still has info from
event in 2012

```
Device Id       : 34304  
Device Name     : 0x8600  
Device Error Code : 6e010000(H)  
Device Error Type : NULL  
Device Error Name : NULL  
Device Instance : 0  
Sys Error       : (null)  
Errtype        : CATASTROPHIC  
PhyPortLayer   : 0x0  
Port(s) Affected :  
Error Description : aclqos hap reset  
DSAP           : 0  
UUID           : 16777216  
Time           : Thu Aug 2 20:55:32 2012  
                (790409 usecs 501AE944(H) jiffies)
```

Gives a hint that may be
related to ACL / QoS

Timestamp of issue

OBFL – Practical Example 2

Log msg indicating issue with Mod 3

```
%MODULE-4-MOD_WARNING: Module 3 (serial: xxxxxxxx) reported warning due to X-bar Interface ASIC Error in device 70
```

Mod 3 “exceptions” indicating issue with “Sync Loss” with Fabric

```
Nexus# show logging onboard mod 3 exception-log
Device Id      : 70
Device Name    : Santa-Cruz-Module
Sys Error      : X-bar Interface ASIC Error
Errrype       : WARNING
Error Description : Lost Sync on slot:1 asic:0 port:16 to Fabric:1
Time           : Thu Jan 26 13:50:08 2012
                (414667 usecs 4F215A10(H) jiffies)
```

Specify time-range you want to see any “interrupts” or “counters” that occurred – Helps TAC narrow down issue

```
Nexus# show logging onboard mod 3 starttime 01/26/12-13:49:00 endtime 01/26/12-13:50:30
```

```
Thu Jan 26 13:50:08 2012 (464621 us)
[sc_dc3_check_chico_intr]SCZ Chico Interrupt Detected slot: 1 asic:0 port:11-->connected to -->Fabric-Slot:1
Thu Jan 26 13:50:08 2012 (454727 us)
[sc_dc3_check_chico_intr]SCZ Chico Interrupt Detected slot: 1 asic:0 port:7-->connected to -->Fabric-Slot:1
```

GOLD: Generic On-Line Diagnostics

- Diagnostic test suite to detect hw, sw, or even traffic related issues.
- **Common mistake** = GOLD Failure does not always indicate HW failure & RMA
- Corrective actions are controlled via EEM (Embedded Event Manager)
- Tests run on Supervisors (active & standby) and LCs
 - Bootup
 - Health Monitoring (run while system is live)
 - On-demand
- **NOTE:** Show commands used in Default-VDC, but tests run on all hardware regardless of VDC

Commonly used for
“Soak” tests before
putting system in
production

GOLD: Generic On-Line Diagnostics

- What tests are available ?

```
Nexus# show diagnostic content module 1
```

ID	Name	Attributes	Testing Interval (hh:mm:ss)
1)	ASICRegisterCheck----->	***N*****A	00:01:00
2)	PrimaryBootROM----->	***N*****A	00:30:00
<snip>			
11)	BootupPortLoopback----->	CP*N**XE*T*	-NA-

Intervals are configurable

Legend explaining attributes included in this command (omitted here)

- What does each test do?

```
Nexus# show diagnostic description module 1 test PortLoopback
```

```
PortLoopback :
```

A health monitoring test that will test the packet path from the Supervisor card to the physical port in ADMIN DOWN state on Linecards.

GOLD: Generic On-Line Diagnostics

- GOLD can take corrective action upon failure - 6.2(8) and later
 - PortLoopback, RewriteEngineLoopback, SnakeLoopback test , and StandbyFabricLoopback.

- Simulate test failures

Useful to see syslog msgs, test corrective actions, etc

```
Nexus(config)# diagnostic test simulation mod 2 test 7 fail
Nexus(config)# show log log
2015 Mar 17 03:34:08 %DIAG_PORT_LB-2-REWRITE_ENGINE_LOOPBACK_TEST_FAIL: Module:2 Test:RewriteEngine Loopback failed 10
consecutive times due to faulty port(s):1-24 Error:Simulated Error for Forwarding Asic ports. Error disabling ports
```

GOLD: Generic On-Line Diagnostics

- View diagnostic result details and statistics

```
Nexus(config)# show diagnostic result mod 2 statistics
```

```
6) PortLoopback
```

Port No	Packet TX	Packet RX	Packet Loss
1	15624	15624	0
----- SNIP -----			
24	15628	15625	3

Remember...

Diag drops / failures do not mean 100% hardware failure.

Packet Loss could be due to congestion as well...

```
Nexus(config)# show diagnostic result mod 2 test 7 detail
```

```
7) RewriteEngineLoopback:
```

```
Error code -----> DIAG TEST ERR DISABLE
Total run count -----> 4223
Last test execution time ----> Tue Mar 17 03:35:08 2015
First test failure time ----> Tue Mar 17 03:25:08 2015
Last test failure time ----> Tue Mar 17 03:34:08 2015
Last test pass time -----> Tue Mar 17 03:24:08 2015
Total failure count -----> 10
Consecutive failure count ---> 10
Last failure reason -----> Simulated Test
```


GOLD: Generic On-Line Diagnostics



Test Name	Test Type	Description
ASICRegisterCheck	Health Monitoring	Checks read/write access to scratch registers on ASICs on the module. Runs on all modules. (enabled by default)
CryptoDevice	Bootup	Checks the CTS device initialization on the module. Run on the Supervisor only.
NVRAM	Health Monitoring	Checks the sanity of the NVRAM device on the module. Runs on the Supervisor only. (enabled by default)
RealTimeClock	Health Monitoring	Verifies the real time clock on the module. Runs on the Supervisor only. (enabled by default)
PrimaryBootROM	Health Monitoring	Verifies the primary BootROM state. Runs on all modules.
SecondaryBootROM	Health Monitoring	Verifies the secondary BootROM state. Runs on all modules.
CompactFlash	Health Monitoring	Verifies access to the internal and external compactflash devices. Runs on the Supervisor only. (enabled by default)
PwrMgmtBus	Health Monitoring	Verifies the redundant Power Management Control Bus. Runs on the Supervisor only. (enabled by default)

GOLD: Generic On-Line Diagnostics



Test Name	Test Type	Description
SpineControlBus	Health Monitoring	Verifies the redundant Spine Card Control Bus. Runs on the Supervisor only. (enabled by default)
SystemMgmtBus	Health Monitoring	Verifies the redundant System Management Bus that controls the Power Supplies and the Fans. Runs on the Supervisor only. (enabled by default)
MgmtPortLoopback	Bootup	Tests loopback on the management port of the module. Runs on the Supervisor only.
EOBCPortLoopback	Bootup	Tests loopback on the EOBC interface of the module. Runs on all modules.
RewriteEngineLoopback	Health Monitoring	Performs non-disruptive loopback for all LineCard ports up to the Rewrite Engine ASIC. (enabled by default)
PortLoopback	Health Monitoring	Verifies the packet path loopback till the port physical device for admin down ports. Runs on LineCards only (enabled by default)
SpineControlBus	Health Monitoring	Verifies the redundant Spine Card Control Bus. Runs on the Supervisor only. (enabled by default)
SystemMgmtBus	Health Monitoring	Verifies the redundant System Management Bus that controls the Power Supplies and the Fans. Runs on the Supervisor only. (enabled by default)

GOLD: Generic On-Line Diagnostics



Test Name	Test Type	Description
OBFL	Bootup	Checks the onboard flash used for failure logging (OBFL) device initialization on the module (enabled by default)
SnakeLoopback	Health Monitoring	Does SnakeTest for SoC based ASICs (F1, F2, and F2e modules)
InternalPortLoopback	Health Monitoring	Nondisruptive per-port loopback test, and hence can run on ports that are up as well (Supported on M2, F2, F2E)
StandbyFabricLoopback	Health Monitoring	Verifies packet path from the Standby supervisor to the Fabric (Supported on Sup1, Sup2, Sup2E)

GOLD: Generic On-Line Diagnostics



- GOLD tests & corrective actions are run via **EEM** as system policies

```
Nexus(config)# show event manager system-policy
      Name : __PortLoopback
Description : Do CallHome, log error and disable further HM testing on affected ports after 10 consecutive failures of GOLD
"PortLoopback" test
Overridable : Yes
```

Original portloopback
policy required 10
consecutive failures

- Can override various GOLD test's parameters

```
Nexus(config)# event manager applet custom-PortLoopback override __PortLoopback
Nexus(config-applet)# event gold mod 2 test PortLoopback testing-type monitoring consecutive-failure 5
Nexus(config-applet)# action 1 policy-default
```

```
Nexus(config-applet)# show event manager policy internal
Name : custom-PortLoopback (overrides __PortLoopback)
      Policy Type : applet
      Event Specification : event gold module 2 test PortLoopback testing-type monitoring consecutive-failure 5
      Action : 1, sup:service:default
Event Specification active on : Active
```

After overriding policy,
now requires only 5.

Putting it all Together

Log Msg indicating Mod 2 failure

```
Nexus# show logging log
2014 Jun 6 14:38:49 %MODULE-2-MOD_DIAG_FAIL: Module 2 reported failure on ports 2/17-2/17 (Ethernet)
```

```
Nexus# show logging onboard exception-log
<snip> - some irrelevant output omitted
Device Id       : 79
Device Name     : Metropolis
Sys Error       : Metro fatal interrupt
Errrtye        : CATASTROPHIC
PhyPortLayer    : Ethernet
Port(s) Affected : 17, 19, 21, 23, 25, 27, 29, 31
Time            : Fri Jun 6 14:38:46 2014
```

Gives specific ASIC on LC that had the problem

Found config changes being made just prior to issue – led us to be able to reproduce and RCA issue

```
Nexus# show accounting log
Fri Jun 6 14:02:38 2014 cmd=configure terminal ; monitor session 1 ; source vlan 2909 both (SUCCESS)
Fri Jun 6 14:12:42 2014 cmd=configure terminal ; monitor session 1 (SUCCESS)
Fri Jun 6 14:12:43 2014 cmd=configure terminal ; monitor session 1 ; filter vlan 1-3967 include-untagged (SUCCESS)
```

Module Troubleshooting Cheat Sheet

- tac-pac
- show tech gold
- show tech mod all
- show tech platform (hidden)



Scripting (EEM, Scheduler, & Python)



Scripting: EEM

- EEM scripts will have an “*event*” that triggers an “*action*”

Event based on SNMP
OID (many other events
also avail)

```
event manager applet HIGH-CPU
  event snmp oid 1.3.6.1.4.1.9.9.109.1.1.1.1.6.1 get-type exact entry-op ge entry-val 60 poll-interval 1
  action 1.0 syslog msg High CPU hit $_event_pub_time
  action 2.0 cli enable
  action 3.0 cli show clock >> bootflash:high-cpu.txt
  action 4.0 cli show proc cpu sort >> bootflash:high-cpu.txt
```

Creating a syslog message
and running show
commands when event
triggers

```
event manager applet clear_hw_counter
  action 1.0 cli show hardware rate-limiter > bootflash:hw_rl.out
  action 2.0 cli clear hardware rate-limit all
  action 3.0 cli show policy-map interface control > bootflash:cop
  action 4.0 cli clear copp stat
```

Notice no “*event*”.

So how do we trigger?

- Can also be **scheduled** at a specific time or intervals...

Scripting: Scheduler + EEM

- Scheduler – per vdc feature allows you to schedule specific tasks
 - Save configs, run commands, call EEM scripts etc...

Enable feature then
Create “job name”

```
Nexus(config)# feature scheduler
Nexus(config)# scheduler job name check-inband
Nexus(config-job)# event manager run clear_hw_counter
```

Job is to “run” EEM script
“clear_hw_counter”

```
Nexus(config)# scheduler schedule name check-inband
Nexus(config)# time start now repeat 00:00:2
Nexus(config)# job name check-inband
Nexus(config)# email-addr mesau@cisco.com
```

Schedules above job to
run every two minutes

```
Nexus(config-applet)# sh scheduler schedule
Schedule Name      : check-inband
-----
User Name          : mesau
Schedule Type      : Run every 0 Days 0 Hrs 2 Mins
Start Time         : Wed Mar 18 04:25:53 2015
Last Execution Time : Wed Mar 18 04:35:53 2015
Last Completion Time: Wed Mar 18 04:35:53 2015
Execution count    : 6
```

Scheduler timestamp matches
file created by EEM script

```
Nexus(config-applet) dir bootflash:
33756 Feb 16 06:24:24 2014 bootflash
15014 Mar 18 04:35:54 2015 hw_rl.out
```

Scripting: Scheduler + Python

Scripts must be stored in /scripts directory

- Scheduler can also call .py scripts (starting in 6.2.8a)

```
Nexus(config)# dir bootflash:scripts
 130   Mar 30 20:17:47 2015  switch_info_file2.py
4136   Mar 30 19:59:56  urib-mrib-rpf-check.py
```

```
Nexus(config)# scheduler job name rpf-check
Nexus(config-job)# source urib-mrib-rpf-check.py
```

Calls specified python script.

```
Nexus(config-schedule)# show scheduler logfile
```

```
Job Name       : rpf-check           Job Status: Success (0)
Schedule Name  : rpf-check-schedule  User Name  : mesau
Completion time: Mon Mar 30 20:55:42 2015
```

You can see the results of the called .py script

```
----- Job Output -----
`source urib-mrib-rpf-check.py`
```

```
(200.5.1.200/32, 235.50.1.100/32), uptime: 2d18h, ip mrib pim
Incoming interface: Ethernet4/1, RPF nbr: 172.16.1.42
Outgoing interface list: (count: 1)
  Ethernet2/23, uptime: 2d18h, mrib
```

Of course, you can just have .py redirect CLI output to file as well 😊

-----SNIP-----

Scripting: Reference Info



- **EEM:**

http://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus7000/sw/system-management/guide/b_Cisco_Nexus_7000_Series_NX-OS_System_Management_Configuration_Guide/b_Cisco_Nexus_7000_Series_NX-OS_System_Management_Configuration_Guide_appendix_010101.html

- **Scheduler:**

http://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus7000/sw/system-management/guide/b_Cisco_Nexus_7000_Series_NX-OS_System_Management_Configuration_Guide/b_Cisco_Nexus_7000_Series_NX-OS_System_Management_Configuration_Guide_chapter_01110.html

- **Python:**

http://www.cisco.com/c/en/us/td/docs/switches/datacenter/sw/6_x/nx-os/fundamentals/configuration/guide/b_Cisco_Nexus_7000_Series_NX-OS_Fundamentals_Configuration_Guide_Release_6-x/b_Cisco_Nexus_7000_Series_NX-OS_Fundamentals_Configuration_Guide_Release_6-x_chapter_01011.html

Packet Capture Tools (SPAN, ELAM, Ethalyzer)



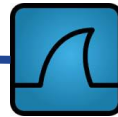
Packet Capture Tools: SPAN Options

- Allows us to capture any packet Rx/Tx the switch
- Even allows us to capture specific exception packets being dropped
- Copy of original packet
- Requires external sniffer

Packet Capture Tools: SPAN Options cont...

Local SPAN

```
monitor session 2 type local
source interface port-channel1995 both
source interface Ethernet 4/1 both
source interface Ethernet 7/1-7/5 both
source vlan 500-510 both
destination interface Ethernet2/23
```



Sniffer Device

ERSPAN (Encapsulated Remote)

```
monitor session 1 type erspan-source
erspan-id 41
vrf default
destination ip 100.3.20.101
source interface port-channel11 both
source interface Ethernet4/1 both
```

Exception SPAN (Packet Drops)

```
monitor session 2
source exception all
destination interface Ethernet2/23
```

Nexus# show hardware ip verify

IPv4 IDS Checks	Status	Packets Failed
checksum	Enabled	3671
protocol	Enabled	183
address source multicast	Enabled	851

Now we can Identify source of dropped packets



Sniffer Device

Layer 3 Network

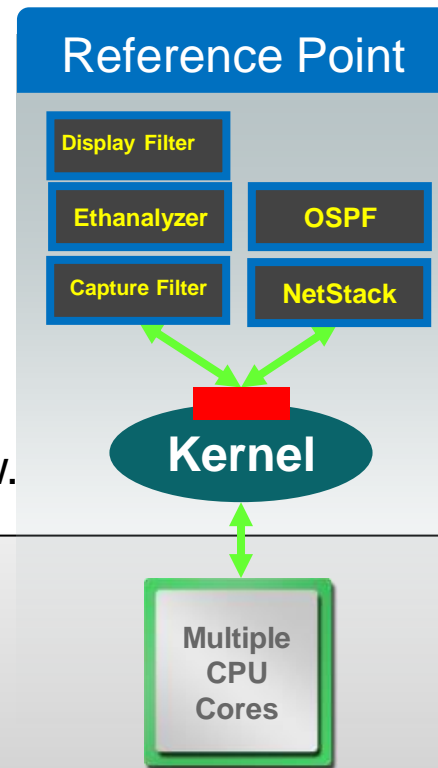


Sniffer Device

Packet Capture Tools: Ethalyzer

- Built-in sniffer that captures packets **Tx/Rx CPU**
 - Can also capture data-plane packets with use of ACL (See Appendix I)
- No longer required to have external sniffer
- View capture immediately via CLI, or write to a file for export view.

```
Nexus# ethalyzer local interface inband ?
capture-filter      Filter on ethalyzer capture
capture-ring-buffer Capture ring buffer option
decode-internal     Include internal system header decoding
detail             Display detailed protocol information
display            Display packets even when writing to a file
display-filter      Display filter on frames captured
limit-captured-frames Maximum number of frames to be captured (default is 10)
limit-frame-size    Capture only a subset of a frame
raw               Hex/Ascii dump the packet with possibly one line summary
write             Filename to save capture to
```



Packet Capture Tools: Ethalyzer cont...

- “Brief” decode capture, filter examples, and write to file

```
Nexus# ethalyzer local interface inband capture-filter "host 10.10.10.2" limit-captured-frames 5
Capturing on inband
2015-02-10 12:51:52.150404 10.10.10.1 -> 10.10.10.2 UDP Source port: 3200 Destination port: 3200
2015-02-10 12:51:52.150480 10.10.10.2 -> 10.10.10.1 UDP Source port: 3200 Destination port: 3200
2015-02-10 12:51:52.496447 10.10.10.2 -> 10.10.10.1 UDP Source port: 3200 Destination port: 3200
2015-02-10 12:51:52.497201 10.10.10.1 -> 10.10.10.2 UDP Source port: 3200 Destination port: 3200
2015-02-10 12:51:53.149831 10.10.10.1 -> 10.10.10.2 UDP Source port: 3200 Destination port: 3200
5 packets captured
```

```
Nexus# ethalyzer local interface inband display-filter "ip.addr==10.10.10.2" limit-captured-frames 5
Capturing on inband
2015-02-10 12:53:54.217462 10.10.10.1 -> 10.10.10.2 UDP Source port: 3200 Destination port: 3200
2015-02-10 12:53:54.217819 10.10.10.2 -> 10.10.10.1 UDP Source port: 3200 Destination port: 3200
2 packets captured
```

```
Nexus# ethalyzer local interface inband capture-filter "tcp port 5000" write bootflash:test.cap
```

```
Nexus# dir test.cap
26224 Jan12 18:40:08 2015 test.cap
```


Packet Capture Tools: Ethalyzer cont...

- Detailed packet capture example:

```
Nexus# ethalyzer local interface inband detail
Capturing on inband
Frame 1 (106 bytes on wire, 74 bytes captured)
  Arrival Time: Feb 10, 2013 23:00:24.253088000
<snip>
Ethernet II, Src: 00:26:51:ce:0f:44 (00:26:51:ce:0f:44), Dst: 01:00:5e:00:00:0a (01:00:5e:00:00:0a)
  Destination: 01:00:5e:00:00:0a (01:00:5e:00:00:0a)
    Address: 01:00:5e:00:00:0a (01:00:5e:00:00:0a)
      .... 1 .... = IG bit: Group address (multicast/broadcast)
      .... 0 .... = LG bit: Globally unique address (factory default)
  Source: 00:26:51:ce:0f:44 (00:26:51:ce:0f:44)
    Address: 00:26:51:ce:0f:44 (00:26:51:ce:0f:44)
      .... 0 .... = IG bit: Individual address (unicast)
      .... 0 .... = LG bit: Globally unique address (factory default)
  Type: IP (0x0800)
Internet Protocol, Src: 10.10.18.6 (10.10.18.6), Dst: 224.0.0.10 (224.0.0.10)
-----SNIP-----
```

Packet Capture Tools: Cheat Sheet



Packet Capture Tool	When Useful	Comments
SPAN	<ul style="list-style-type: none">• External Sniffer is available.• Capture many packets (look at traffic rate)	<ul style="list-style-type: none">• Local SPAN (external sniffer local to switch)• ERSPAN (centralized sniffer, no need for directly connected sniffer)• Exception SPAN: Captures dropped packets and identifies offending source/dst IPs
Ethalyzer	<ul style="list-style-type: none">• No external sniffer• Capture packets Rx/Tx CPU• Can be used with ACL to capture data-plane packets	<ul style="list-style-type: none">• Run in default vdc• Can be viewed immediately or directed to file• Capture and display filters

Packet Capture Tools: Reference Info



- Ethalyzer:

http://www.cisco.com/c/en/us/products/collateral/switches/nexus-3000-series-switches/white_paper_c11-673817.html

- SPAN:

http://www.cisco.com/c/en/us/td/docs/switches/datacenter/sw/6_x/nx-os/system_management/configuration/guide/sm_nx_os_cg/sm_14span.html

- http://www.cisco.com/c/en/us/td/docs/switches/datacenter/sw/6_x/nx-os/system_management/configuration/guide/sm_nx_os_cg/sm_erspan.html

Agenda

- ✓ Strategy & Tools
- Control-Plane Troubleshooting
- Data Plane Troubleshooting
- Q&A / Conclusion

Control Plane Troubleshooting

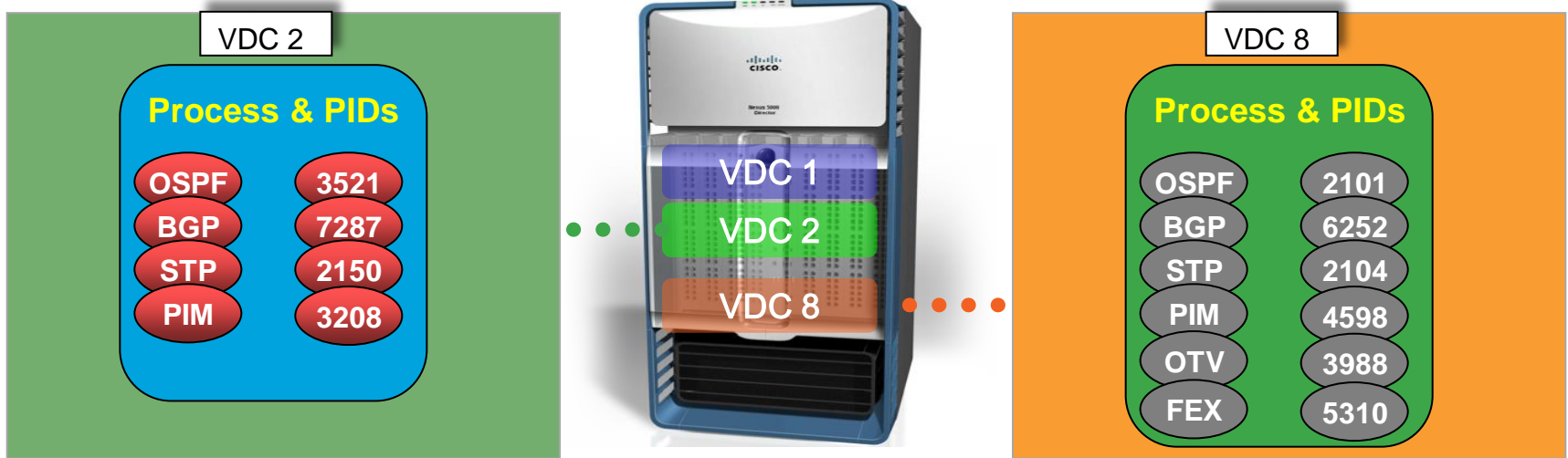


Control-Plane Troubleshooting: Architecture

- N7K supports VDCs (Virtual Device Context)
- Control-Plane processes are separated on a per VDC basis
- VDCs still **share CPU and Inband** (path to CPU) resources
- Pre-emptive multitasking to prevent single process CPUHOGs
- Sup2/Sup2E add **“cpu-shares”** allowing priority to specific VDCs

	Sup1	Sup2	Sup2E
CPU	Dual-Core Xeon	Quad-Core Xeon	2 x Quad-Core Xeon
VDCs	4	4+1	8+1
CPU-Shares	Not Supported	Supported	Supported

Control-Plane Troubleshooting: Architecture

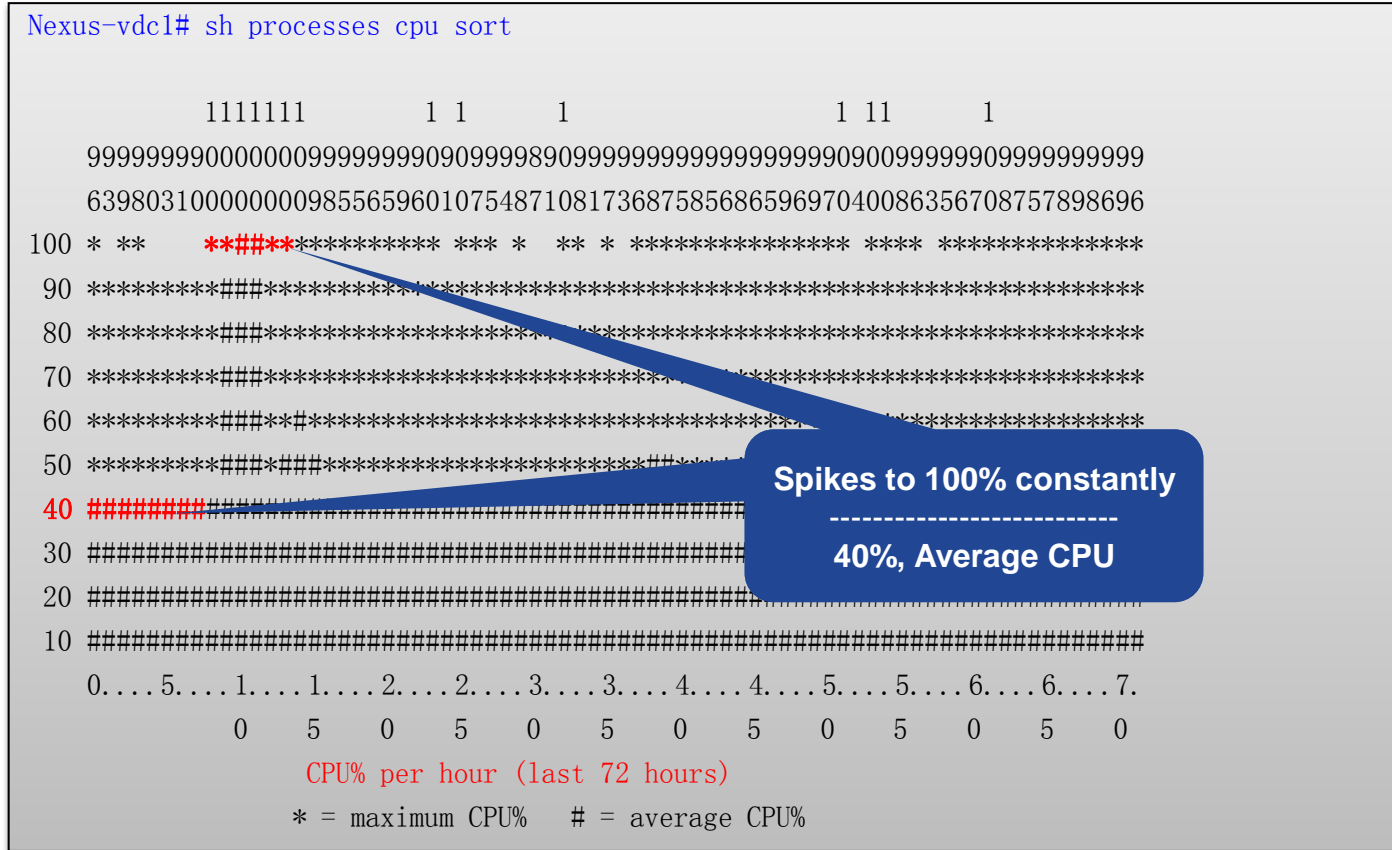


- Processes in one VDC are independent of other VDCs

Control-Plane Troubleshooting: Processes CPU

- Always understand CPU history and **baseline** for your system

- CPU history can be observed for past **60 sec**, **60 min**, and **72 hrs**



Control-Plane Troubleshooting: Processes CPU

Process in question
OTV PID = 8273

```
Nexus-vdcl# # sh processes cpu sort
CPU utilization for five seconds: 2%/0%; one minute: 1%; five minutes: 1%
PID      Runtime(ms)   Invoked    uSecs  5Sec   1Min   5Min   TTY  Process
-----
8273     1088950      2566908   2565   60.13% 60.31  30.5%  -   otv
```

Processes sorted
high to low

```
Corrochio-otv# show process cpu detailed 8273
```

```
CPU utilization for five seconds: 1%/0%; one minute: 1%; five minutes: 1%
PID      Runtime(ms)   Invoked    uSecs  5Sec   1Min   5Min   TTY  Process
-----
8273     200          20         10     60.63% 58.31% 0.21%  -   otv
8330    278710      716662     0      0.00%  0.00%  0.00%  -   otv:active-time
8333    158660      406757     0      48.21% 42.61% 0.01%  -   otv:otv-ip-udp
```

PIDs have sub-
processes as well
8273 → 8333

```
Nexus-vdcl# show otv internal event-history events
```

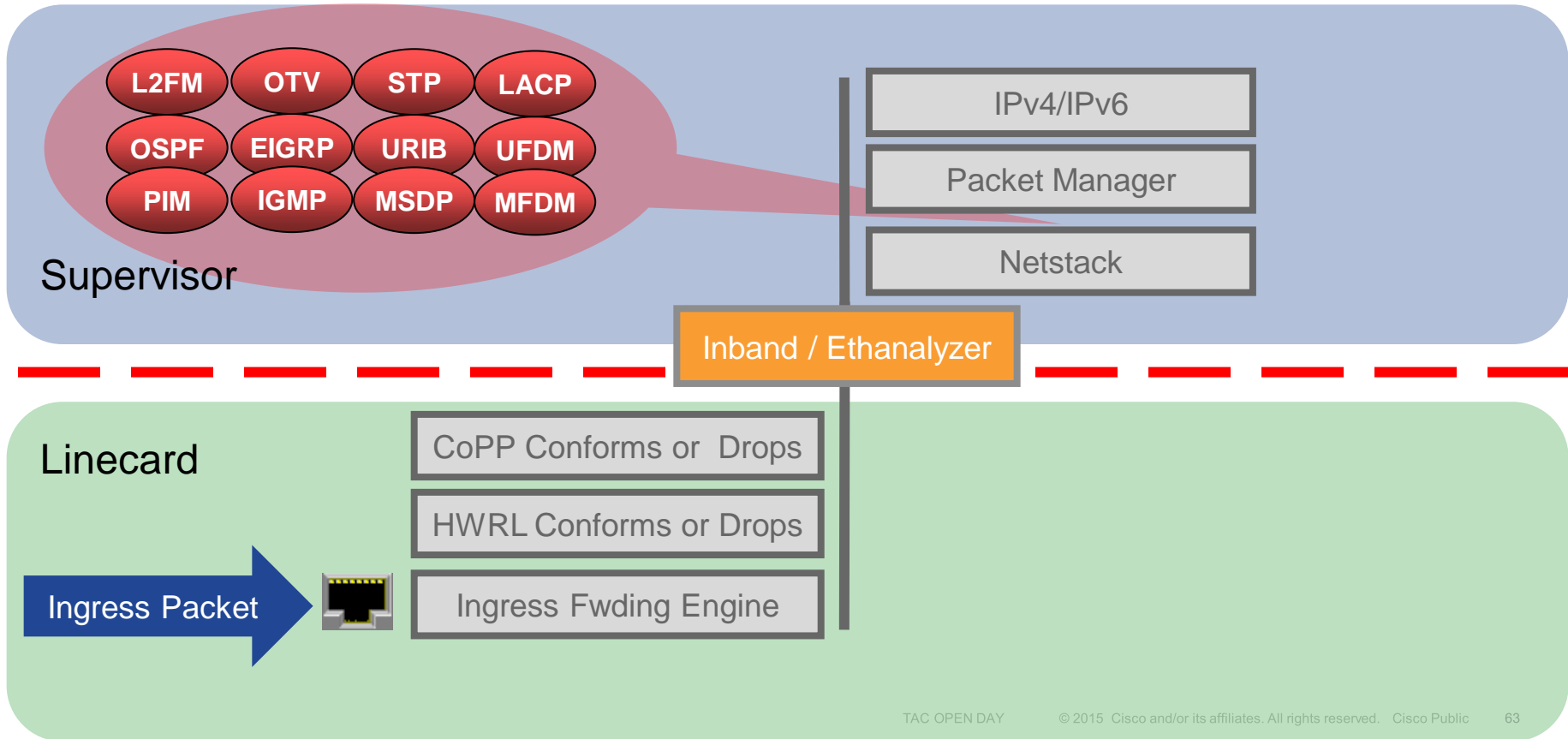
```
2015 Apr 4 03:36:57.236187 otv [8273]: [8333]: otv_ha_send_vd_sdb_batched_updates:
2015 Apr 4 03:36:27.203105 otv [8273]: [8333]: otv_ha_send_vd_sdb_batched_updates:
2015 Apr 4 03:35:57.163145 otv [8273]: [8333]: otv_ha_send_vd_sdb_batched_updates:
2015 Apr 4 03:35:27.133090 otv [8273]: [8333]: otv_ha_send_vd_sdb_batched_updates:
```

Event history gives
further indication of
what's occurring.

Control-Plane Troubleshooting: Traffic

- Common traffic types driving CPU
 - ARP destined to CPU
 - Fragmented traffic
 - Glean
 - Broadcast / Link-Local mcast
- Identifying offending traffic
 - Control-Plane Policing (CoPP) & Hardware Rate-Limiters (HWRL)
 - Both implemented on **`per-module`** basis and CLI available in **default vdc**
 - Inband counters
 - Ethalyzer

Control-Plane Troubleshooting: Traffic Path



Control-Plane Troubleshooting: HWRL Traffic

- Hardware Rate-Limiters are on by default & CLI available in **default vdc**
- Implemented on a **'per-module'** basis

```
Nexus# show hardware rate-limiter module 9
Module: 9
```

R-L Class	Config	Allowed	Total	
L3 mtu	500	3012591	3030853	6043444
<snip>				
receive	30000	3011594	0	3011594
L2 port-sec	500	0	0	0
L2 mcast-snoop	10000	0	0	0

Type of Traffic

Configured Rate = pps

- Clear these counters via CLI: `clear hardware rate-limiter <options>`

Control-Plane Troubleshooting: CoPP Traffic

- CoPP: more customizable and granular than HWRL
- 4 default profiles (dense, lenient, moderate, strict), but also **customizable**
 - Cannot modify default profiles. Copy default profile and then customize

```
Nexus# copp copy profile moderate suffix mesau
Nexus#(config)# show run copp
class-map type control-plane match-any copp-class-critical-mesau
  match access-group name copp-acl-bgp-mesau
  match access-group name copp-acl-rip-mesau
----- SNIP -----
class-map type control-plane match-any copp-class-monitoring-mesau
class-map type control-plane match-any copp-class-multicast-host-mesau
class-map type control-plane match-any copp-class-multicast-router-mesau
```

- Customize existing classes and policers, or create your own
 - **Match** (acl, protocol, etc) → **Classify** (class-map) → **Police** (policer)

Control-Plane Troubleshooting: CoPP Traffic cont...

- Pay attention to violate packets AND conform packets

```
Nexus# show policy-map interface control-plane | eg -i "drop|class|mod|violate|conform" | exc "violated 0"
class-map copp-mcast-data-mesau (match-any)
  violate action: drop
  module 1:
    conformed 23085272 bytes,
  module 2:
    conformed 10681168 bytes,
class-map class-default (match-any)
  violate action: drop
```

Conformed packets can be just as important as dropped packets, depending on the class (ex: TTL expiring classes)

Control-Plane Troubleshooting: CoPP Traffic cont...

- Ethalyzer will see packets that have **NOT** been dropped by CoPP/HWRL

```
Nexus# ethalyzer local interface inband limit-captured-frames 0
Capturing on inband
Corrochio# ethalyzer local interface inband capture-filter "ether proto \ip" limit-captured-frames 0
Capturing on inband
2015-04-09 21:10:42.444847 150.10.2.253 -> 224.0.0.2 HSRP 62 Hello (state Active)
2015-04-09 21:10:42.450939 10.0.103.252 -> 224.0.0.10 EIGRP 74 Hello
2015-04-09 21:10:42.493594 200.5.0.252 -> 224.0.0.5 OSPF 78 Hello Packet
2015-04-09 21:10:42.573178 150.10.3.252 -> 224.0.0.2 HSRP 62 Hello (state Standby)
2015-04-09 21:10:42.585122 150.10.1.253 -> 224.0.0.2 HSRP 62 Hello (state Active)
2015-04-09 21:10:42.603590 100.4.10.252 -> 100.4.10.253 OSPF 2814 LS Update
2015-04-09 21:10:42.614942 150.10.1.252 -> 224.0.0.5 OSPF 82 Hello Packet
2015-04-09 21:10:42.623505 100.2.10.252 -> 224.0.0.5 OSPF 82 Hello Packet
2015-04-09 21:10:42.803436 10.15.11.252 -> 224.0.0.2 HSRP 62 Hello (state Standby)
2015-04-09 21:10:42.823577 150.1.1.1 -> 150.1.1.2 UDP 78 Source port: tick-port Destination port: tick-port
2015-04-09 21:10:42.835838 100.4.12.252 -> 224.0.0.13 PIMv2 70 Hello
12 2015-04-09 21:10:42.975603 100.1.20.253 -> 224.0.0.2 HSRP 62 Hello (state Standby)
2015-04-09 21:10:43.002977 10.0.104.252 -> 224.0.0.2 HSRP 62 Hello (state Active)
```

Control-Plane Troubleshooting: Inband Traffic

- Key statistics when looking at the traffic hitting In-Band, destined for CPU

```
Nexus# show hardware internal cpu-mac inband stats
<snip>
Packet rate limit ..... 32000 pps
Rx packet rate (current/max) 1173 / 2107 pps
Tx packet rate (current/max) 4558 / 1765 pps
...
Rate limit reached counter .. 0
...
Rx no buffers ..... 0
```

Max rx pps
&
of times reached pps rx
limit for Inband

Indicates when any new
maximum Rx pkts occurred

```
Nexus# show hardware internal cpu-mac inband events
```

- ```
6) Event:RX_PPS_MAX, length:4, at 387644 usecs after Thu Mar 26 14:22:00 2015
 new maximum = 2107

29) Event:RX_PPS_MAX, length:4, at 357606 usecs after Sat Mar 14 05:14:57 2015
 new maximum = 1615
```

So how do we know where the traffic is coming from ?



# Control-Plane Troubleshooting: Inband Traffic

- We can look at various commands to identify offending interfaces...

```
Nexus-vdc1# show system internal pktmgr interface
```

```
Ethernet2/23, ordinal: 201 Hash_type: 1
SUP-traffic statistics: (sent/received)
Packets: 842445 / 11147
Bytes: 71194935 / 2784182
Instant packet rate: 0 pps / 8 pps
Packet rate limiter (Out/In): 0 pps / 0 pps
Average packet rates(1min/5min/15min/EWMA):
Packet statistics:
Tx: Unicast 523, Multicast 841905, Broadcast 17
Rx: Unicast 521, Multicast 10615, Broadcast 11
```

Interface and VDC  
Granularity for CPU bound  
pkts

```
Nexus-vdc1# show system internal pktmgr internal vdc inband | exclude "In VDC"
```

| Interface    | Src Index | VDC ID | Packet rcvd |
|--------------|-----------|--------|-------------|
| Ethernet8/1  | 0x30      | 2      | 42557       |
| Ethernet8/30 | 0x4d      | 2      | 1552046     |
| Ethernet8/48 | 0x5f      | 2      | 6103038     |
| Ethernet2/23 | 0x17e     | 2      | 13352       |
| Ethernet4/1  | 0x180     | 2      | 642228      |

Quick Snapshot of all  
interfaces and packets  
destined for CPU

# Control-Plane Troubleshooting: Cheat-Sheet



- ethalyzer local interface inband ...
- show tech / tac pac
- show tech <feature> <<<< feature driving cpu from show proc cpu
- show proc cpu history
- show proc cpu sort | exc 0.00
- show policy-map interface control-plane | eg "drop|class-map|module" | ex "violated 0| conform"
- show hardware rate-limiter
- show hardware internal cpu-mac inband stats
- show hardware internal cpu-mac inband events
- show system internal pktmgr interface
- show system internal pktmgr internal vdc inband

# Agenda

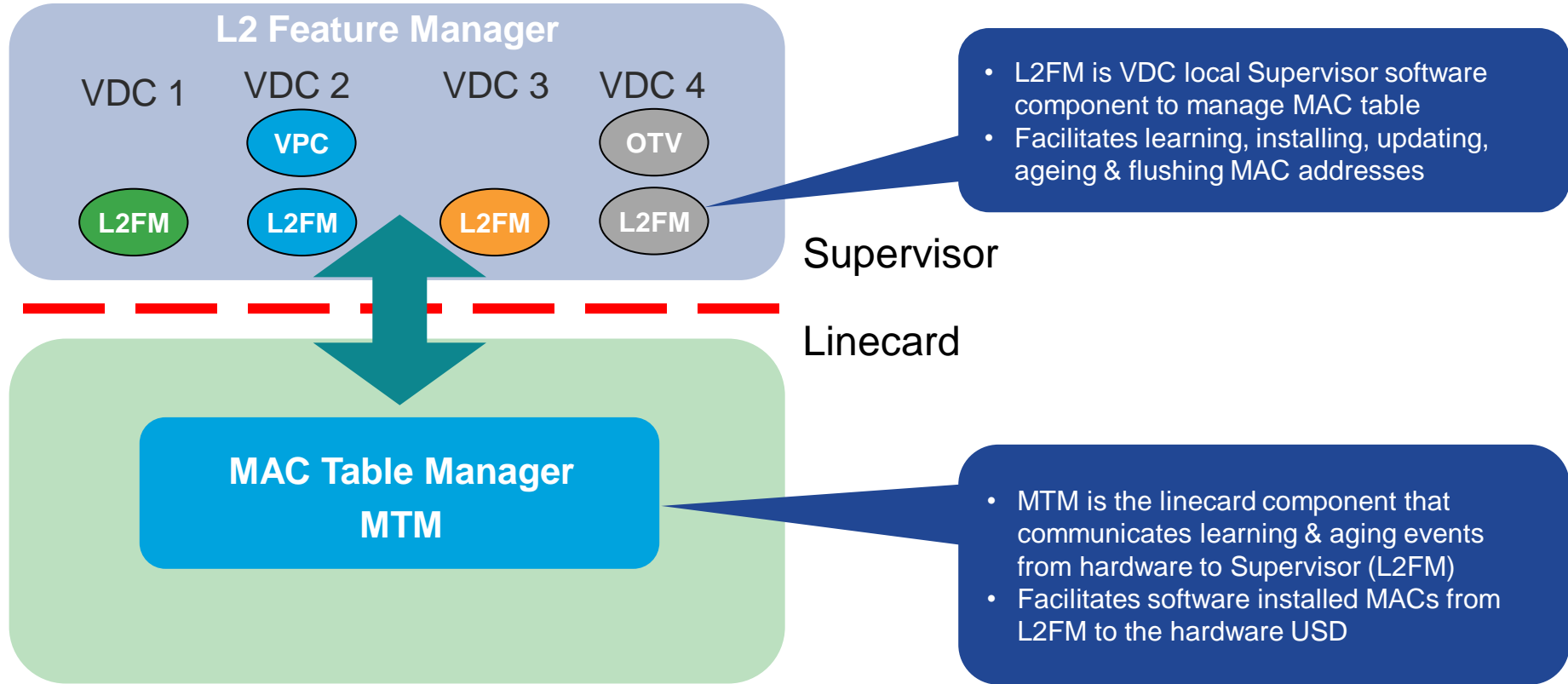
- ✓ Strategy & Tools
- ✓ Control-Plane Troubleshooting
- Data-Plane Troubleshooting
- Q&A / Conclusion

# *Data-Plane Troubleshooting: L2 Programming Verification*

# Layer 2 Data Plane Troubleshooting: Common Symptoms

- Unicast flooding
  - Bandwidth spikes
  - Unexpected traffic on interfaces
  - Congestion drops
- Blackholing traffic
  - Traffic destined out wrong interface

# Layer 2 Data Plane Troubleshooting:



# Layer 2 Data Plane Troubleshooting: Verifying SW

- L2FM is the Supervisor's perspective of the MAC table
- L2FM can pass this information to other features such as OTV

```
Nexus# show mac address-table vlan 500 address 0011.2233.4455
Note: MAC table entries displayed are getting read from software.
Use the 'hardware-age' keyword to get information related to 'Age'
```

This command is looking at Supervisor perspective

Legend:

\* - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC  
age - seconds since last seen, + - primary entry using vPC Peer-Link,  
(T) - True, (F) - False, ~~~ - use 'hardware-age' keyword to retrieve

```
age info
VLAN MAC Address Type age Secure NTFY Ports/SWID. SSID. LID
-----+-----+-----+-----+-----+-----+-----
* 500 0011.2233.4455 dynamic ~~~ F F Po1995
```

Was this Mac learned in HW ?

```
Nexus# show system internal l2fm event-history debugs | include 0011.2233.4455
```

```
[snip]
[104] l2fm_macdb_insert(5817): slot 1 fe 1 mac 0011.2233.4455 vlan 500 flags 0x107 hints 0 E8 NL lc : 0x160007ca
```

Front Panel Port represented in ifIndex

# Layer 2 Data Plane Troubleshooting: Verifying SW

- Debug MACDB tracks current and **historical state of MAC address** learned, updated or flushed

```
Nexus# show system internal 12fm 12dbg macdb address 0011.2233.4455
```

## Legend

```

Db: 0-MACDB, 1-GWMACDB, 2-SMACDB, 3-RMDB, 4-SECMACDB
Src: 0-UNKNOWN, 1-L2FM, 2-PEER, 3-LC, 4-HSRP
 5-GLBP, 6-VRRP, 7-STP, 8-DOTX, 9-PSEC 10-CLI 11-PVLAN
 12-ETHPM, 13-ALW_LRN, 14-Non_PI_MOD, 15-MCT_DOWN, 16 - SDB
 17-OTV, 18-Debounce Timer, 19-AM, 20-PCM_DOWN, 21 - MCT_UP
 22-L2VPN, 23-EFP, 24-DRV
Slot:0 based for LCS 19-MCEC 20-OTV/ORIB
```

```
VLAN: 500 MAC: 0011.2233.4455
```

| Time                            | If/swid    | Db | Op     | Src | Slot | FE |
|---------------------------------|------------|----|--------|-----|------|----|
| Mon Apr 13 01:21:00 2015        | 0x160007ca | 0  | INSERT | 3   | 1    | 1  |
| Mon Apr 13 01:45:39 2015        | 0x160007ca | 0  | FLUSH  | 10  | 0    | 15 |
| Mon Apr 13 01:45:39 2015        | 0x160007ca | 0  | DELETE | 0   | 0    | 15 |
| <b>Mon Apr 13 01:45:40 2015</b> | 0x160007ca | 0  | INSERT | 3   | 1    | 1  |

Use the Legend to determine where the mac was installed from

“how” and “where” MAC was learned

Installed from LC 2 FE 2  
(zero based)

```
Nexus# show system internal pixm info | grep 0x160007ca
```

| PC_TYPE | PORT   | LTL    | RES_ID     | LTL_FLAG   | CB_FLAG    | MEMB_CNT |
|---------|--------|--------|------------|------------|------------|----------|
| Normal  | Po1995 | 0x041e | 0x160007ca | 0x00000000 | 0x00000002 | 1        |



# Layer 2 Data Plane Troubleshooting: Verifying HW

- Everything you need to know about hardware programming is in MTM
- MTM informs L2FM of a mac learn updating software.

```
Nexus# show hardware mac address-table 2 vlan 500 address 0011.2233.4455
```

| FE | Valid | PI | BD | MAC            | Index   | Stat | SW    | Modi | Age  | Tmr | GM | Sec | TR | NT | RM | RMA  | Cap | Fld | Always |
|----|-------|----|----|----------------|---------|------|-------|------|------|-----|----|-----|----|----|----|------|-----|-----|--------|
|    |       |    |    |                |         | ic   |       | fied | Byte | Sel |    | ure | AP | FY |    | TURE |     |     | Learn  |
| 0  | 1     | 0  | 70 | 0011.2233.4455 | 0x0041e | 0    | 0x003 | 0    | 6    | 1   | 0  | 0   | 0  | 0  | 0  | 0    | 0   | 0   | 0      |
| 1  | 1     | 1  | 70 | 0011.2233.4455 | 0x0041e | 0    | 0x003 | 0    | 0    | 1   | 0  | 0   | 0  | 0  | 0  | 0    | 0   | 0   | 0      |

Forwarding Engine

"Primary Input"

MAC Address

LTL Index - Maps to port

```
Nexus# show system internal pixm info | grep 0x160007ca
```

```
Normal Po1995 0x041e 0x160007ca 0x00000000 0x00000002 1
```

# Layer 2 Data Plane Troubleshooting: Verifying HW

- PIXM and PIXMc is responsible for LTL, FPOE and Multicast Index management
- PIXM manages VLAN, BD & CBL Forwarding state for each interface
- Maintains LTL to IF-Index mapping

```
Nexus# sh system internal pixm info ltl 0x41e
```

| PC_TYPE | PORT   | LTL    | RES_ID     | LTL_FLAG   | CB_FLAG    | MEMB_CNT |
|---------|--------|--------|------------|------------|------------|----------|
| Normal  | Po1995 | 0x041e | 0x160007ca | 0x00000000 | 0x00000002 | 1        |

```
Member rbh rbh_cnt
Eth2/24 0x000000ff 0x08
```

```
CBL Check States: Ingress: Enabled; Egress: Enabled
```

```
VLAN| BD | BD-St | CBL St & Direction:
```

|     |      |                    |                   |
|-----|------|--------------------|-------------------|
| 1   | 0x0  | EXCLUDE_IF_FROM_BD | BLOCKING (Both)   |
| 500 | 0x46 | INCLUDE_IF_IN_BD   | FORWARDING (Both) |
| 502 | 0x48 | INCLUDE_IF_IN_BD   | FORWARDING (Both) |

```
Member info
```

```
Type LTL
```

|              |        |
|--------------|--------|
| PORT_CHANNEL | Po1995 |
| MCAST_GROUP  | 0x7fe8 |
| MCAST_GROUP  | 0x7bdc |
| FLOOD_W_FPOE | 0x8046 |
| FLOOD_W_FPOE | 0x8048 |

Port <-> LTL mapping &  
If-Index information

**Color Blocking Logic** is the HW  
version of STP State

PIXM includes interfaces into the  
Flood List for BD if in a *forwarding*  
state

# Layer 2 Data Plane Troubleshooting: Verifying HW

- All in one command to display discrepant, missing, or extra MAC addresses between the supervisor and the module.

```
Nexus# show forwarding consistency 12 2
```

```
Note: MAC table entries displayed are getting read from software.
Use the 'hardware-age' keyword to get information related to 'Age'
```

Legend:

```
* - primary entry, G - Gateway MAC, (R) - Routed MAC, O - Overlay MAC
age - seconds since last seen, + - primary entry using vPC Peer-Link,
(T) - True, (F) - False, - use 'hardware-age' keyword to retrieve age info
```

## Missing entries in the MAC Table

| VLAN | MAC Address | Type           | age    | Secure | NTFY | Ports/SWID. | SSID.       | LID       |
|------|-------------|----------------|--------|--------|------|-------------|-------------|-----------|
| G    | -           | 0022.557a.7446 | static | -      | F    | F           | sup-eth1(R) | (Eth2/23) |

## Extra and Discrepant entries in the MAC Table

| VLAN | MAC Address | Type | age | Secure | NTFY | Ports |
|------|-------------|------|-----|--------|------|-------|
|------|-------------|------|-----|--------|------|-------|

# Layer 2 Data-Plane Troubleshooting: Cheat-Sheet



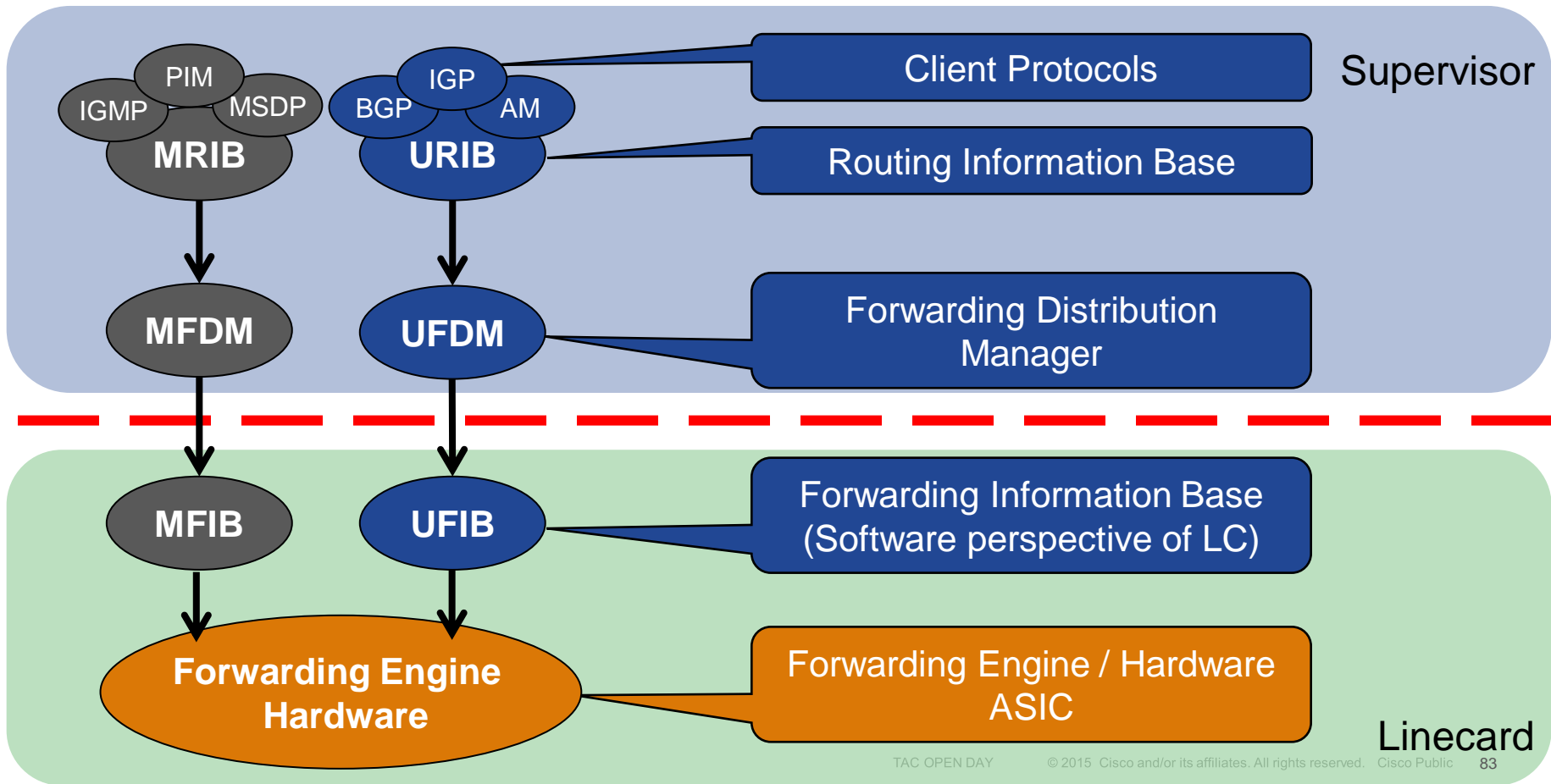
- Show mac address-table
- Show system internal l2fm l2dbg macdb address [mac address]
- show hardware mac address-table [LC#] [mac address]
- show forwarding consistency l2 <mod>
- show tech-support L2FM detail (Always go for the detailed output)
  - Includes MTM information
  - Includes all MACDB L2 debug outputs
- Show tech-support forwarding l2 unicast
- Show tech-support vlan

# *Data-Plane Troubleshooting: L3 Programming Verification*

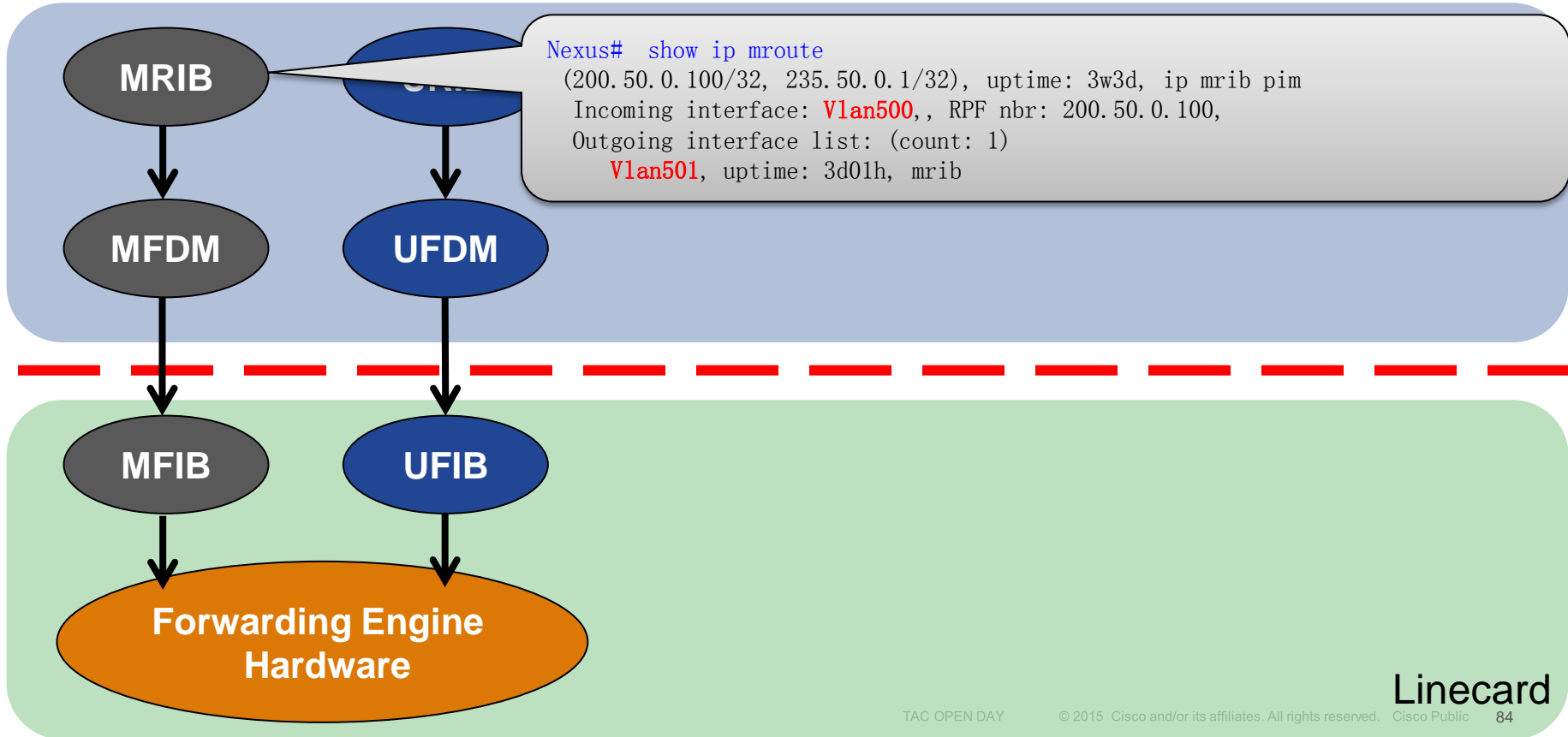
# Layer 3 Data Plane Troubleshooting: Common Symptoms

- Complete Blackholing traffic (for flows)
  - **Not Intermittent traffic loss**
  - Usually a pattern
    - Same next-hop (unicast)
    - Same mroute or OIF (multicast)
    - Group of physical interfaces all mapping to the same forwarding engine (FE)

# Layer 3 Data Plane Troubleshooting: Components

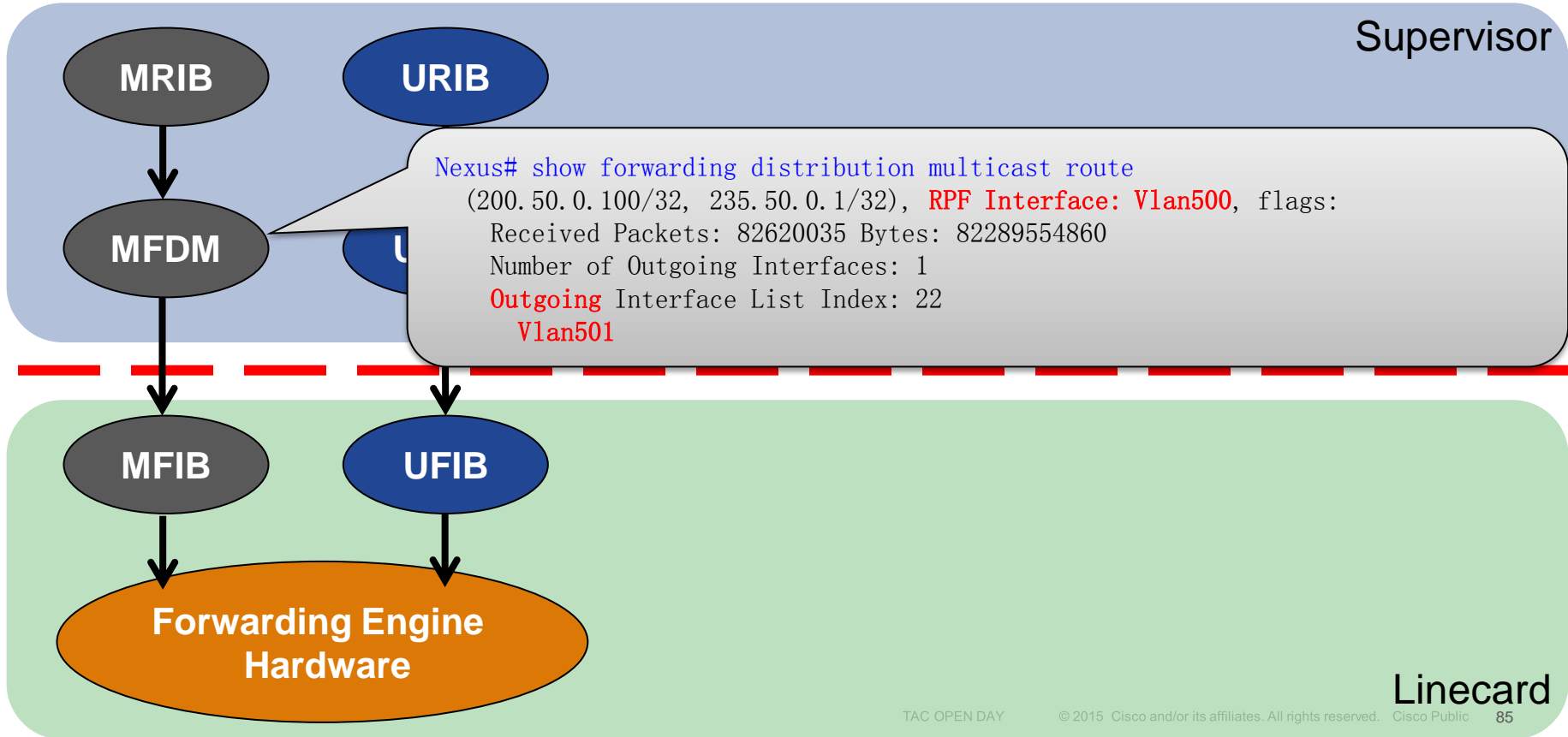


# Layer 3 Data Plane Troubleshooting: Multicast

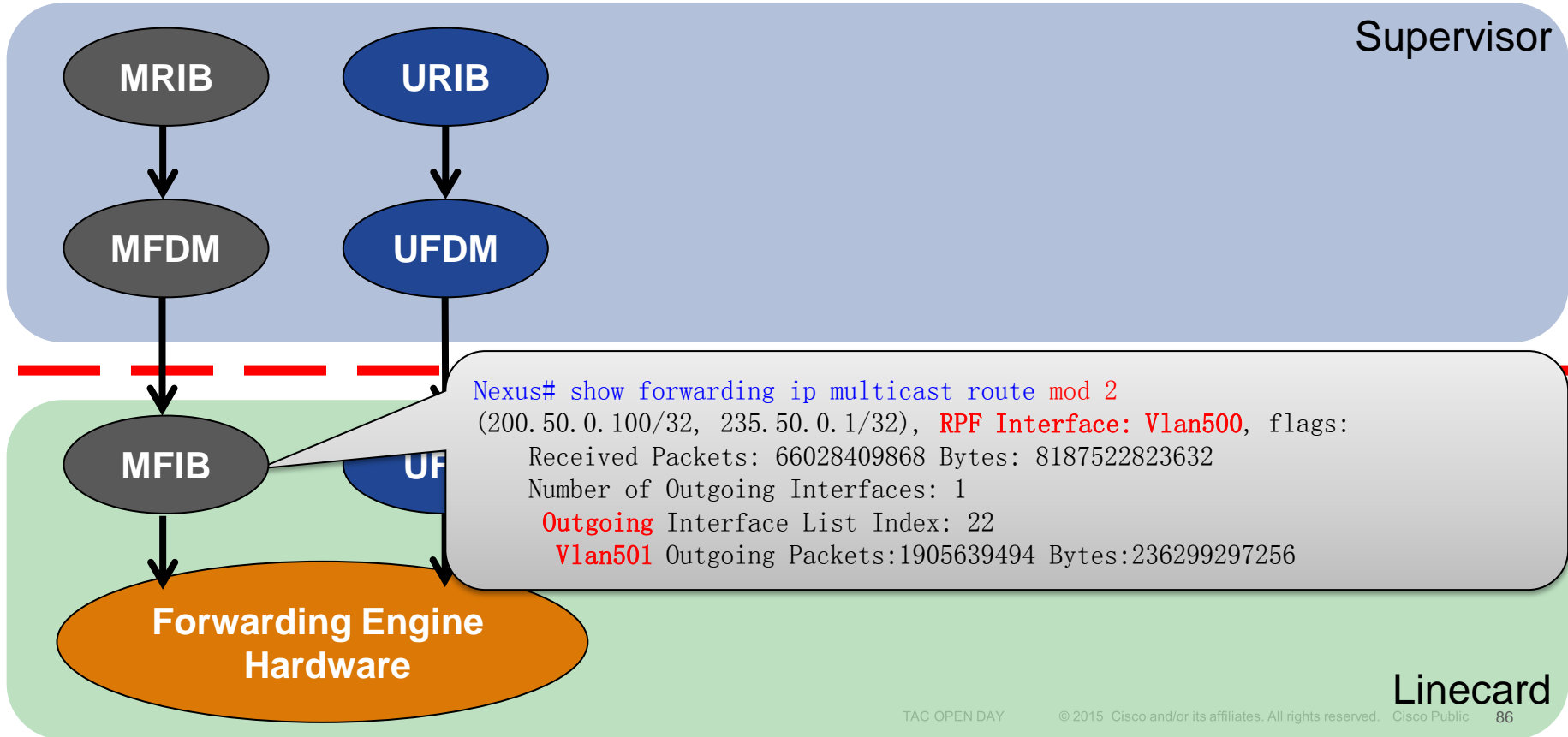




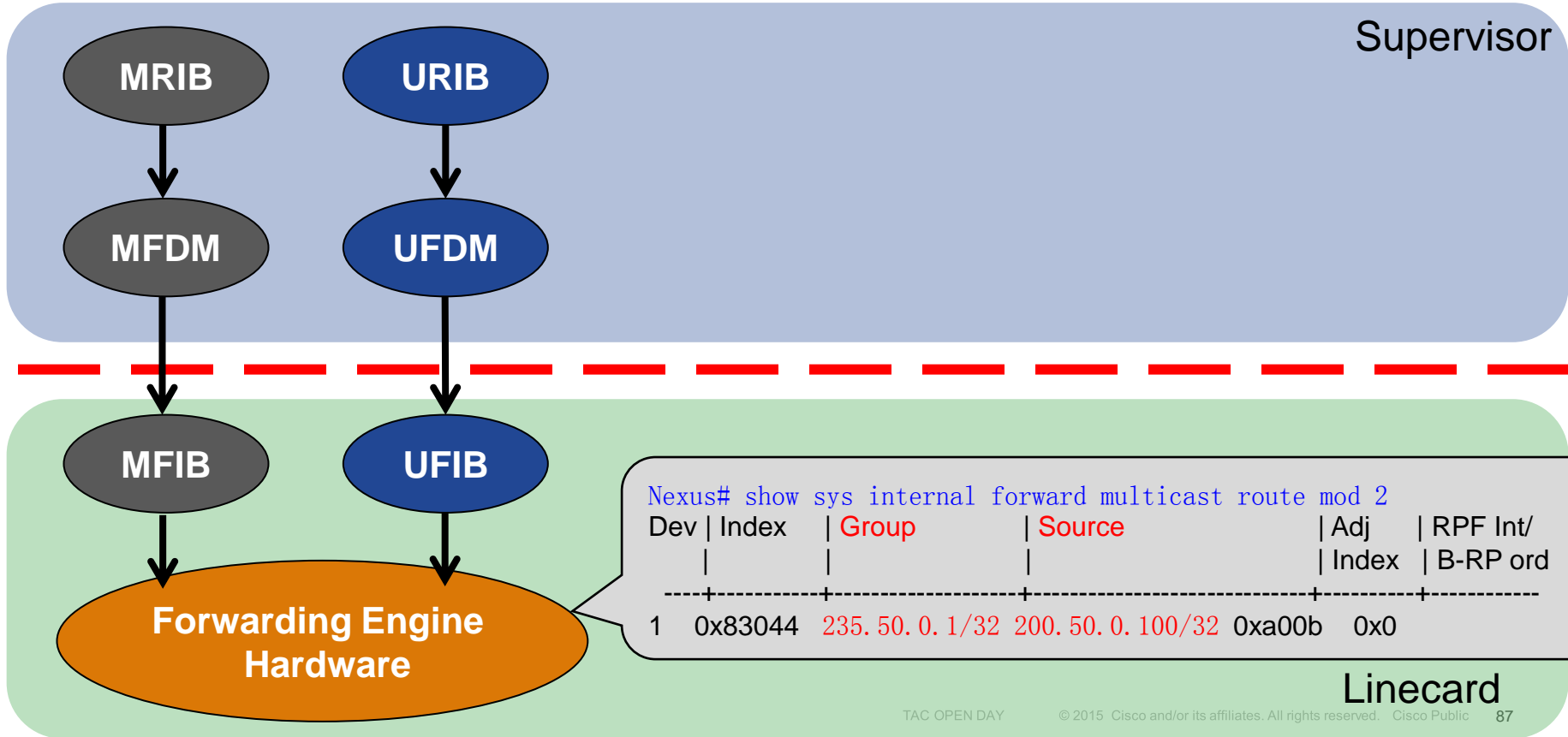
# Layer 3 Data Plane Troubleshooting: Multicast



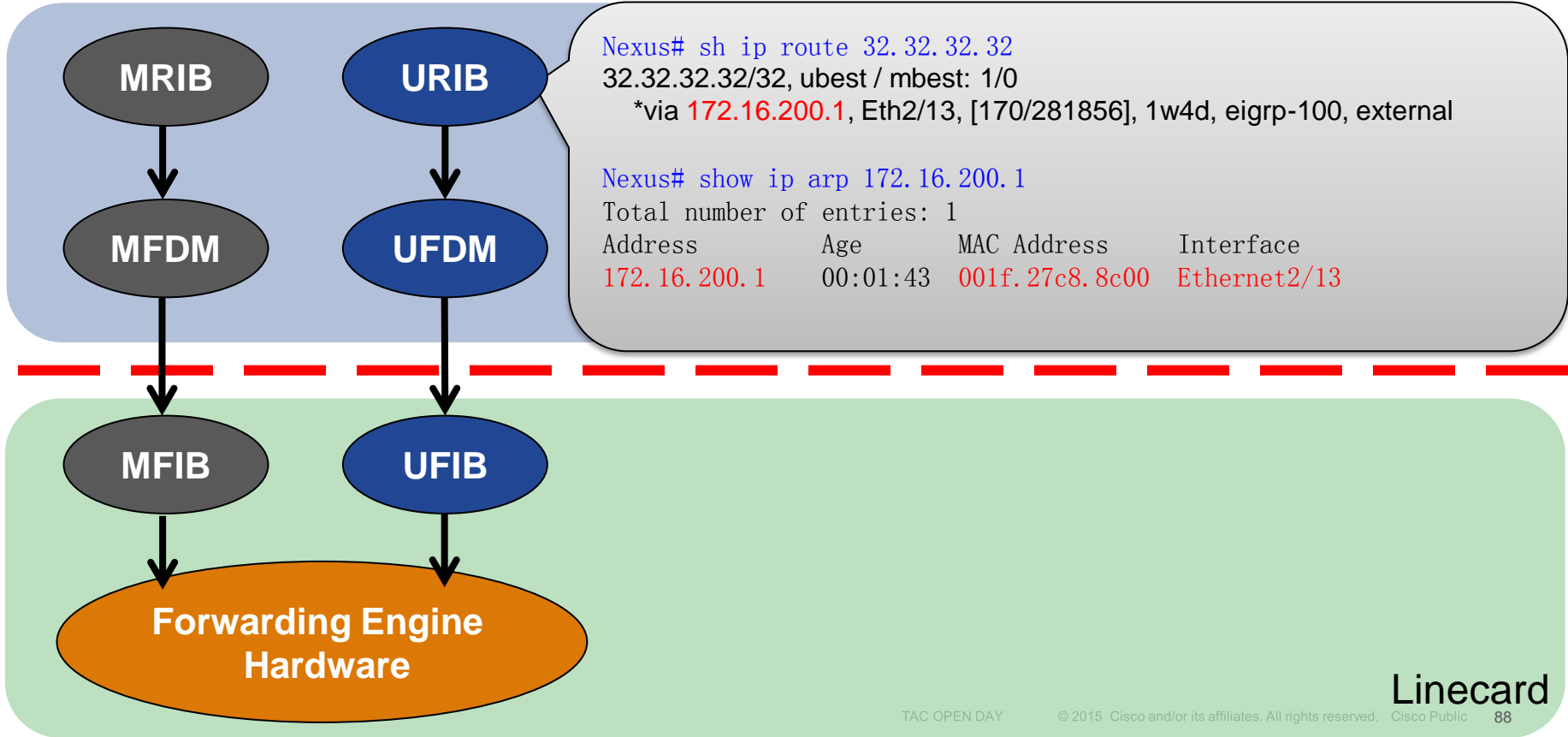
# Layer 3 Data Plane Troubleshooting: Multicast



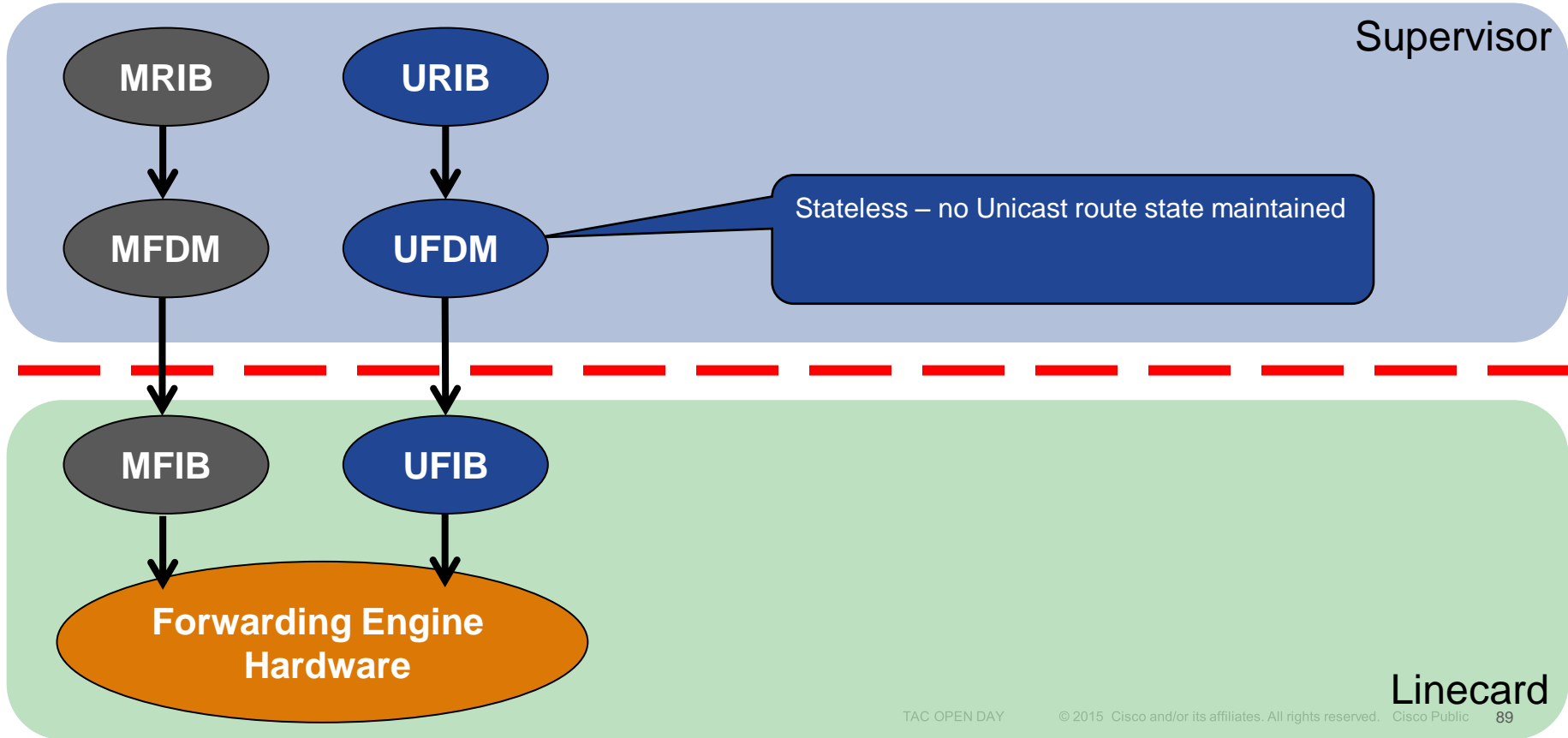
# Layer 3 Data Plane Troubleshooting: Multicast



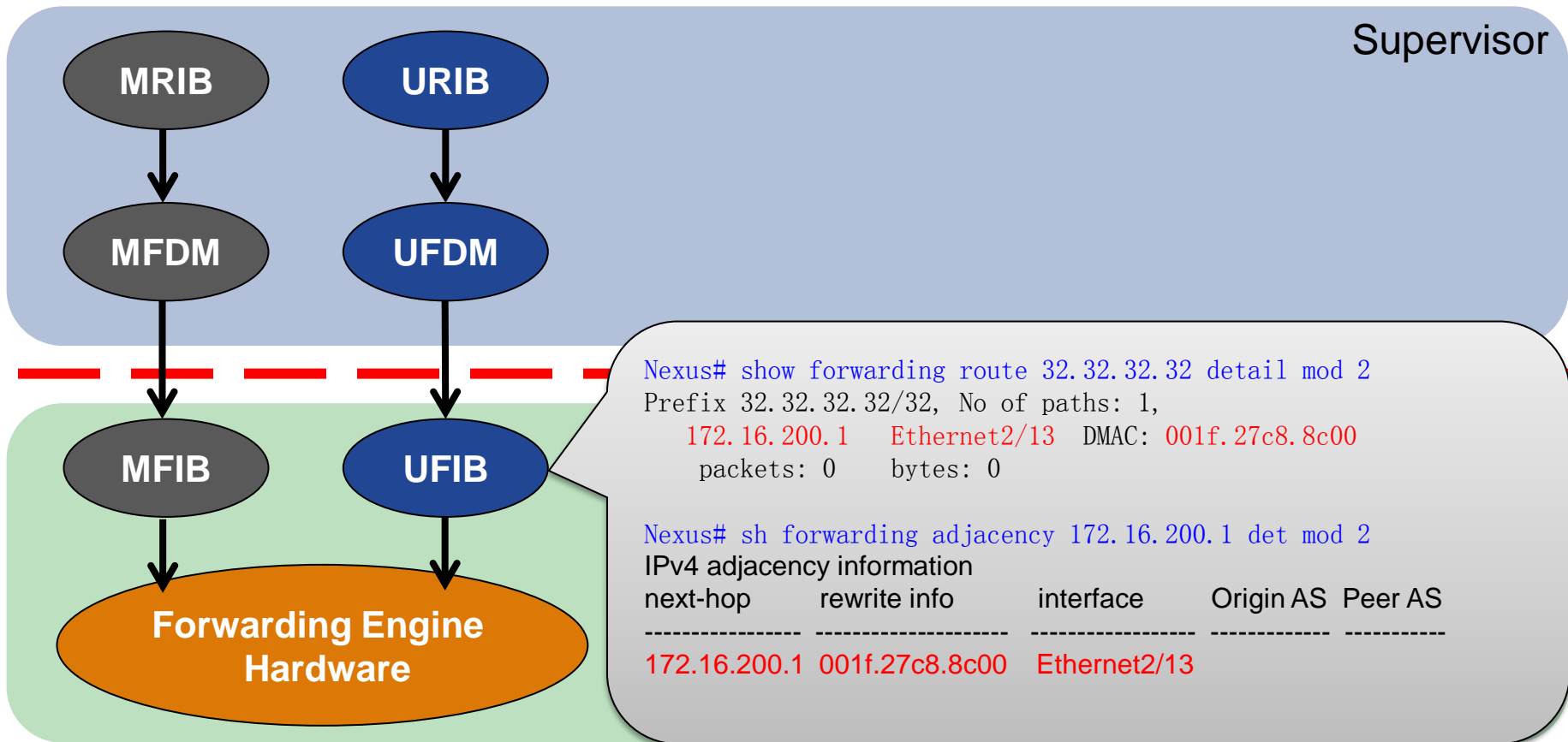
# Layer 3 Data Plane Troubleshooting: Unicast



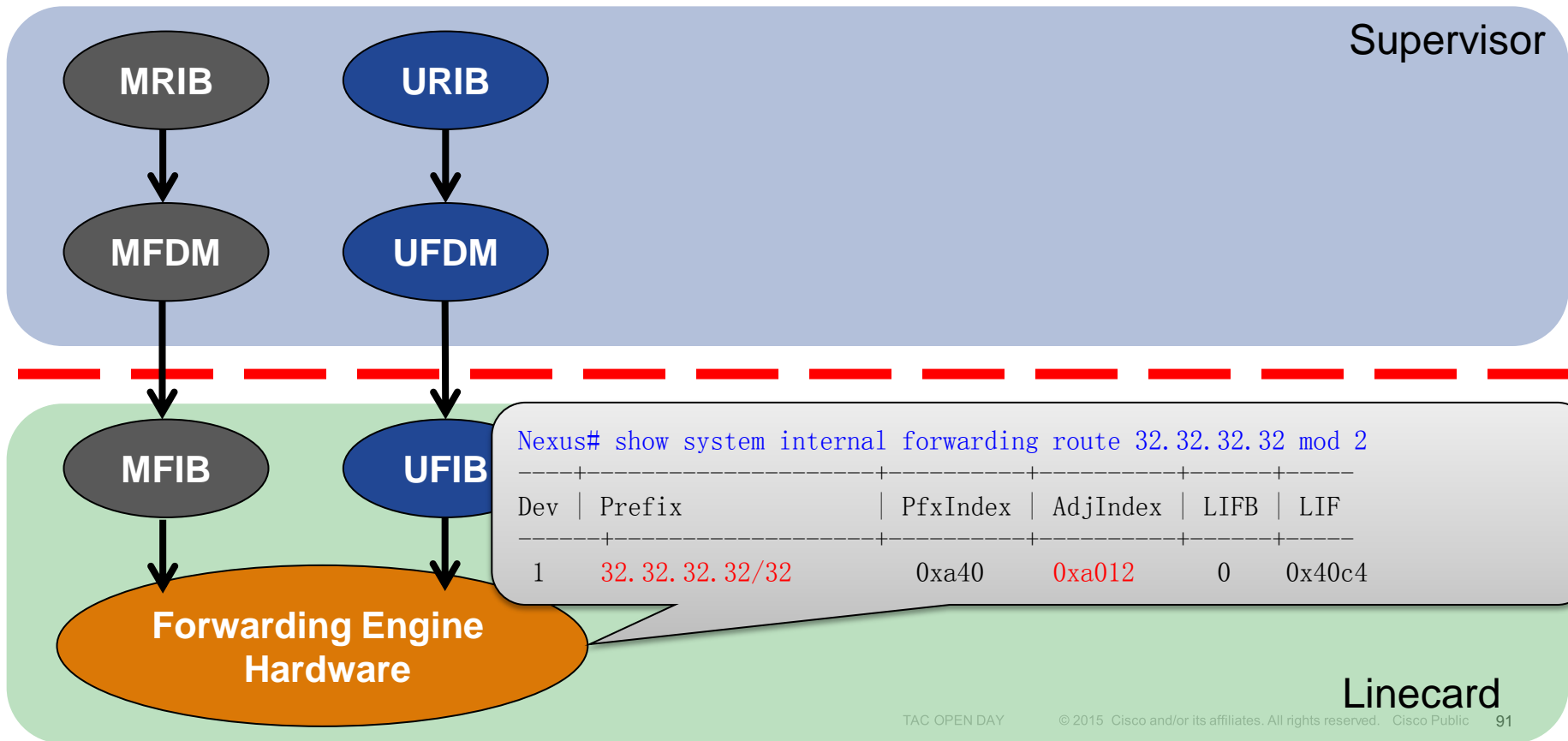
# Layer 3 Data Plane Troubleshooting: Unicast



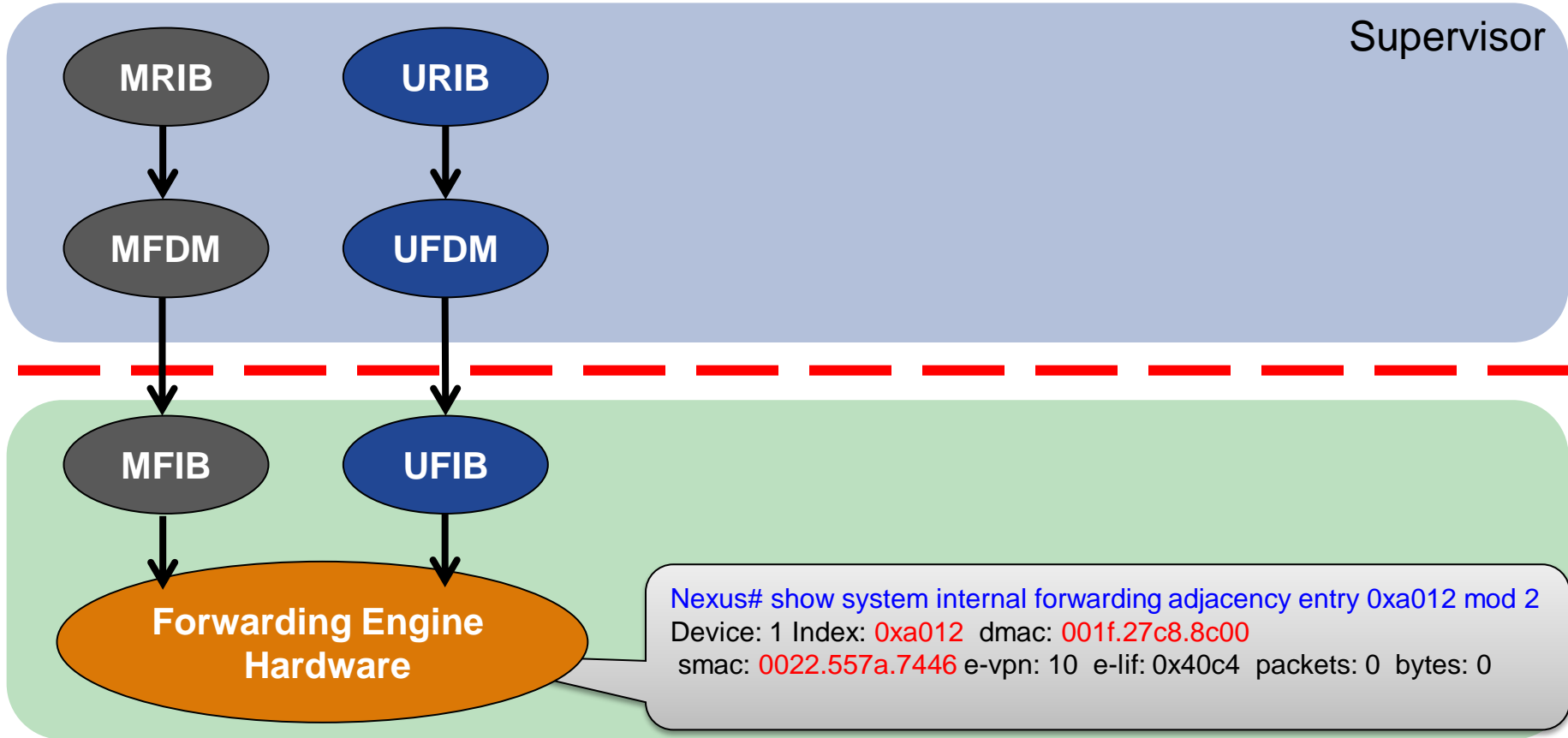
# Layer 3 Data Plane Troubleshooting: Unicast



# Layer 3 Data Plane Troubleshooting: Unicast

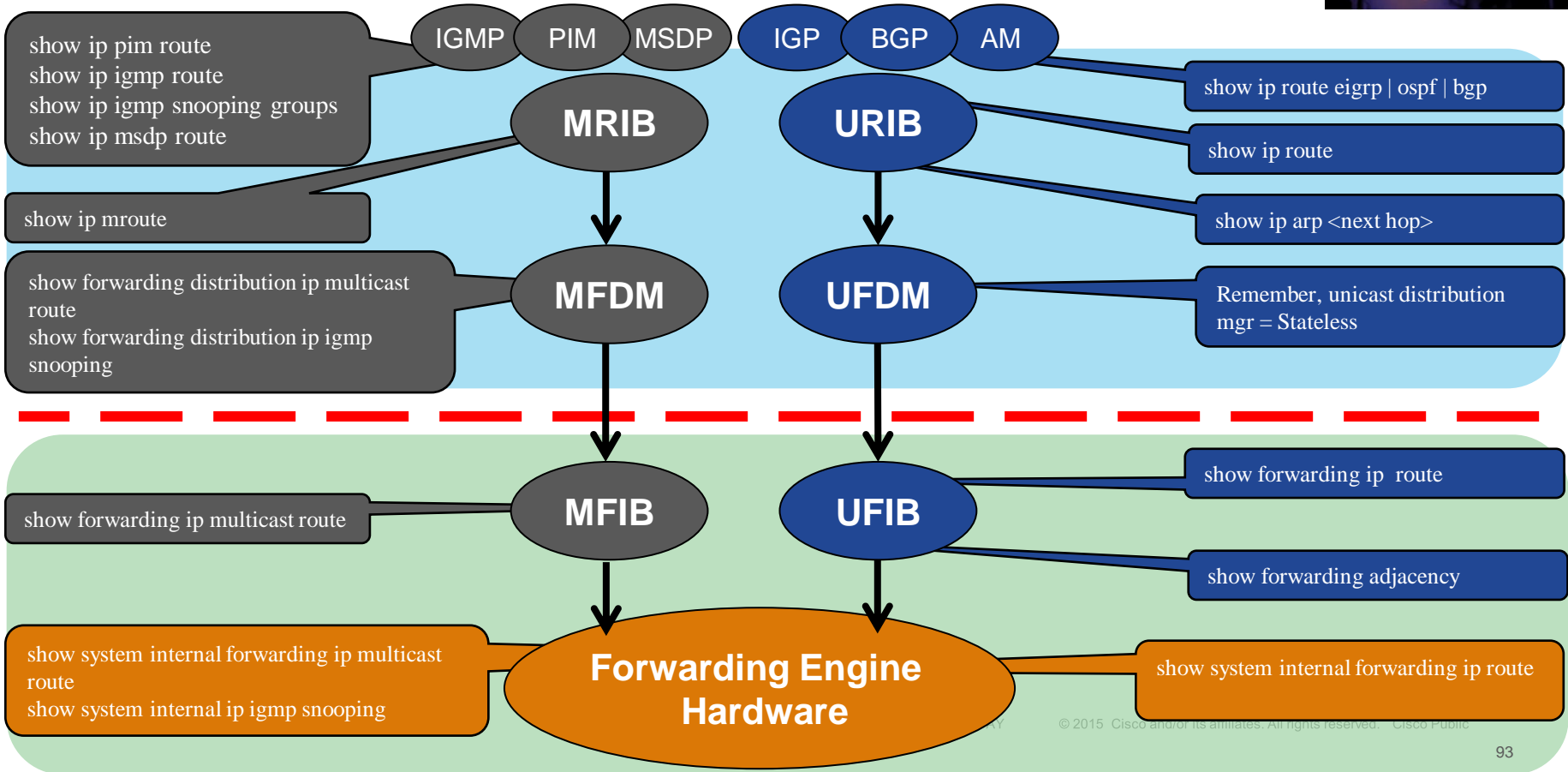


# Layer 3 Data Plane Troubleshooting: Unicast





# Layer 3 Data Plane Troubleshooting: Cheat-Sheet



# Layer 3 Data-Plane Troubleshooting: Cheat-Sheet



## Multicast

- Show tech-support ipv4 multicast  
(show techs included)
  - Show tech-support msdp
  - Show tech-support mfw
  - Show tech-support routing ip multicast
  - Show tech-support ip pim
  - Show tech-support routing ip unicast
- Show tech-support m2rib
- Show tech-support m2fib
- Show tech-support forwarding I2 multicast
- Show tech-support forwarding I3 multicast

## Unicast

- Show tech-support (client protocol) ex. OSPF
- Show tech-support arp
- Show tech-support adjmgr
- Show tech-support u2rib
- Show tech-support u2fib
- Show tech-support forwarding I3 unicast detail

# *Data-Plane Troubleshooting: Intermittent Packet Loss*

# Data-Plane Troubleshooting Intermittent Packet Loss: CoPP and HWRLs

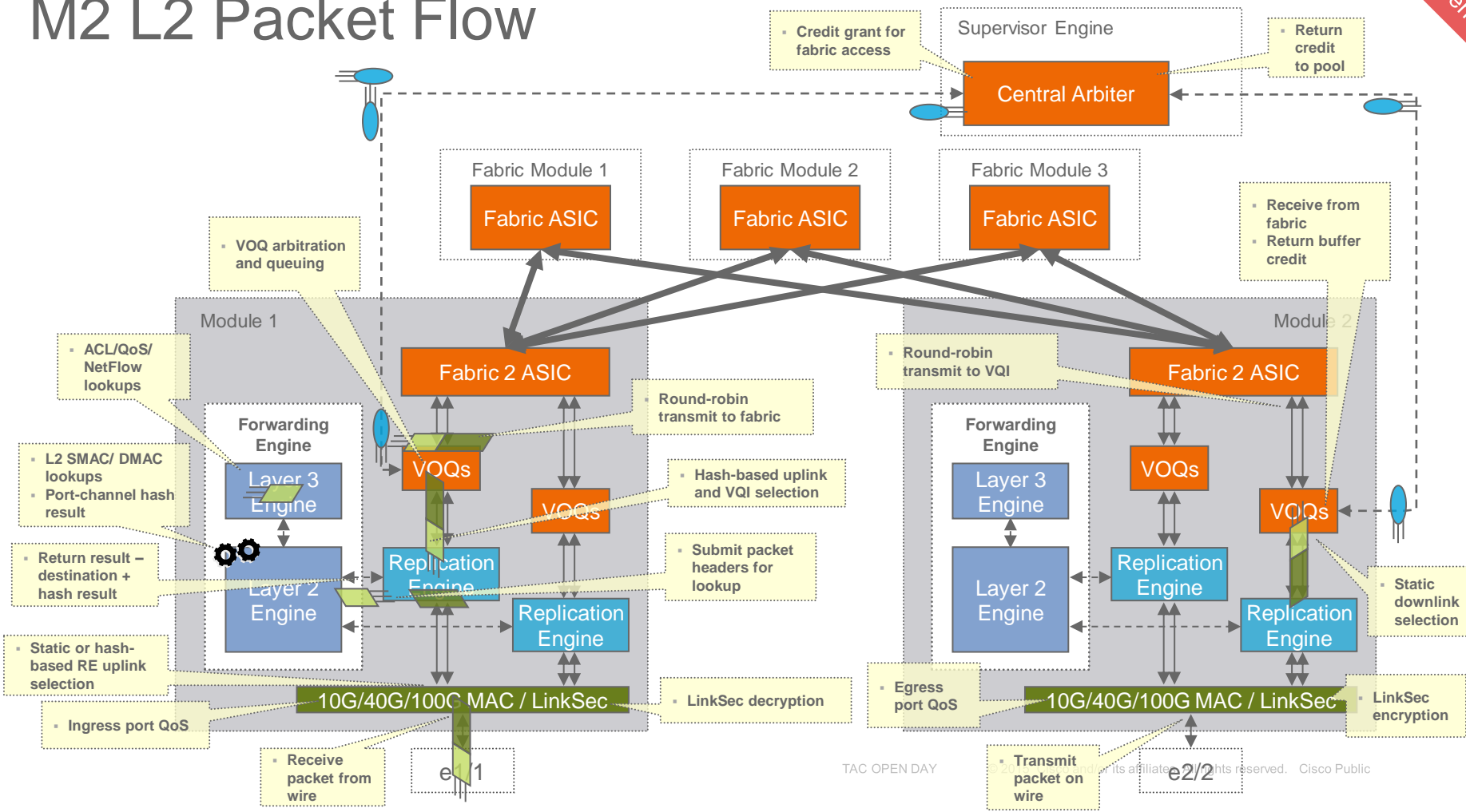
- Control Plane Policing (CoPP) and Hardware Rate-Limiters (HWRL) are **common, but often overlooked** areas for packet drops.
- Packets to the CPU are typically Control Plane packets, but **can be Data Packets too**.
- Both CoPP and HWRL counters are on a **“per-FE”** basis.
- Don't forget Ethalyzer can help to identify CPU bound packets



# M2 L2 Packet Flow

**HDR** = Packet Headers    **DATA** = Packet Data    **CTRL** = Internal Signaling

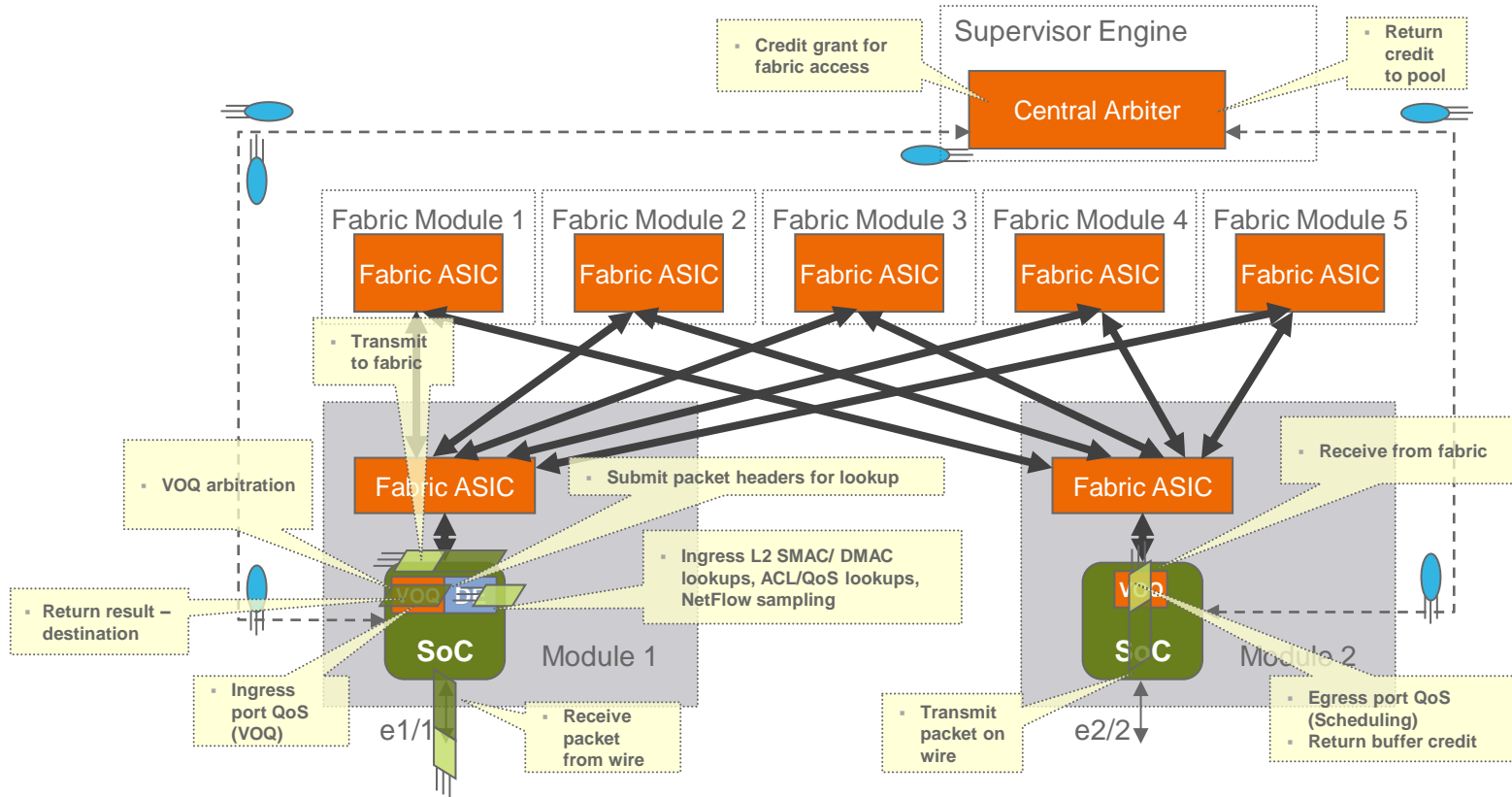
M Series



# F2E / F3 L2 Packet Flow

HDR = Packet Headers DATA = Packet Data CTRL = Internal Signaling

F Series

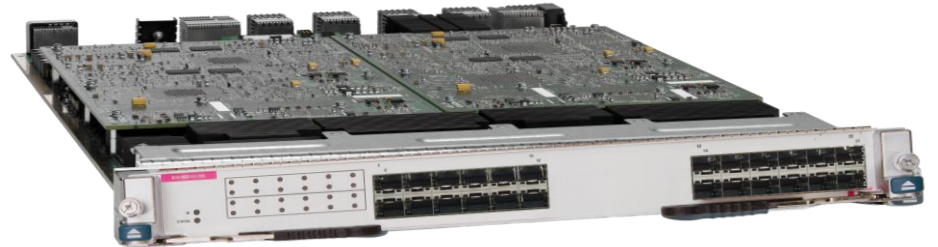


# Data-Plane Troubleshooting Intermittent Packet Loss: Linecard Drop Counters

## Key terms when discussing N7K Linecard Drop Counters

- **Credited traffic:** refers to unicast traffic that is centrally arbitrated by the system.
  - Ingress card will drop packet if no credit is available at egress port.
- **Uncredited traffic:** refers to multi-destination traffic (mcast, bcast, unk ucast).
  - NOT centrally arbitrated, so ingress cards will continue to send, resulting in egress drops
- F-Series is pure ingress-buffered architecture, using VOQ
  - Ingress discards for credited traffic, output drops for uncredited (not through VOQ)
- M-Series is a hybrid ingress & egress buffered I/O module architecture, using VOQ plus ingress/egress port buffers
  - Output Drops for credited / Uncredited, VOQ drops for Credited

# M1 & M2 Packet Counters





# Data-Plane Troubleshooting Intermittent Packet Loss: M-Series Per-Queue Counters

- Per-Interface and Per-Queue counters for Ucast & Mcast...

```
Nexus-1# show policy-map interface ethernet x/y | eg -i class|drop
Class-map (queuing): in-q1 (match-any)
 queue dropped pkts : 0
Class-map (queuing): in-q-default (match-any)
 queue dropped pkts : 0
Class-map (queuing): out-pq1 (match-any)
 queue dropped pkts : 0
Class-map (queuing): out-q2 (match-any)
```

Per-queue drops help identify which CoS traffic is affected

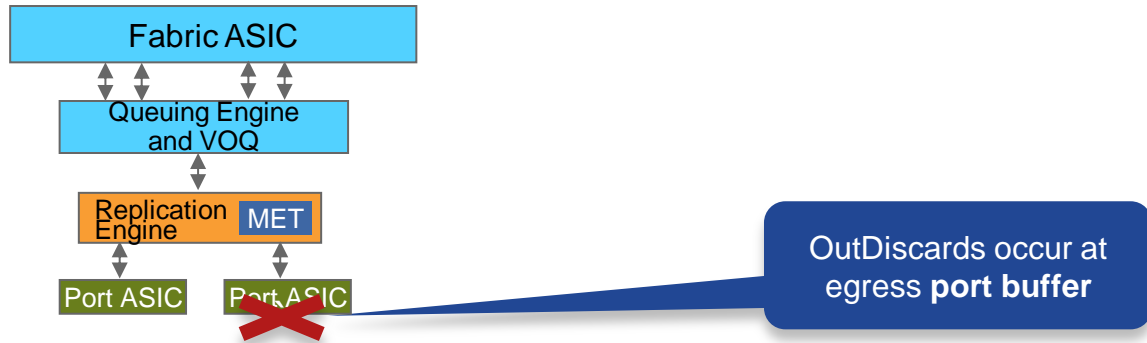
```
Nexus-1# show system internal qos queuing stats interface x/y
Receive queues

Queue 2q4t-in-q-default
 Total packets 0
 WRED drops 0
 Taildrop drops 0
Transmit queues

Queue 1p3q4t-out-q-default
 Total packets 0
 WRED drops 0
 Taildrop drops 0
```

Shows more detailed queue counter/drop info

# Data-Plane Troubleshooting Intermittent Packet Loss: M-Series Egress Port Drops (Ucast & Mcast)

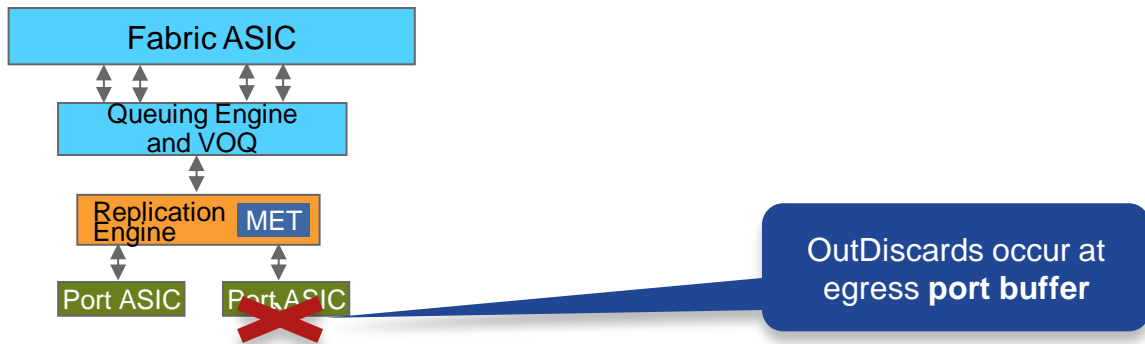


- Oversubscription of the **egress port** is identifiable as **OutDiscards** for Ucast & Mcast

```
Nexus-1# show interface e8/48 counters errors
Sacamano(config-if)# show interface e8/48 counters errors
```

| Port    | Align-Err | FCS-Err | Xmit-Err | Rcv-Err | UnderSize | OutDiscards |
|---------|-----------|---------|----------|---------|-----------|-------------|
| Eth8/48 | 0         | 0       | 0        | 0       | 0         | 330000884   |

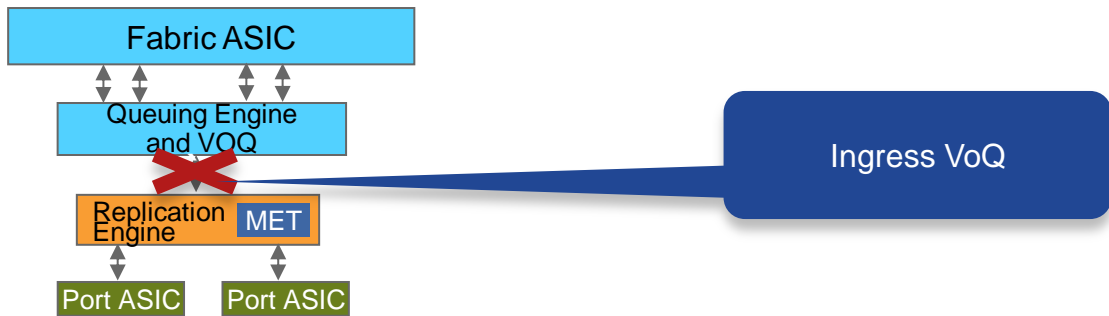
# Data-Plane Troubleshooting Intermittent Packet Loss: M-Series Egress Port Drops (Ucast & Mcast)



- Also viewable via **policy-map counters**

```
Nexus-1# show policy-map interface ethernet 2/24 output
<snip>
Class-map (queuing): out-q-default (match-any)
 queue-limit percent 82
 bandwidth remaining percent 25
 queue dropped pkts : 326893332
```

# Data-Plane Troubleshooting Intermittent Packet Loss: M-Series Unicast Ingress VoQ Drops



- Unicast also utilizes **Ingress Buffers**, and can drop at **Ingress VoQ**

```
Nexus# slot 3 show hardware internal statistics device qengine congestion port 2
83 BA hard drop count for QOS3 0000000074393315 1-4 I1
87 BA WRED drop count for QOS3 0000000002390602 1-4 I1
```

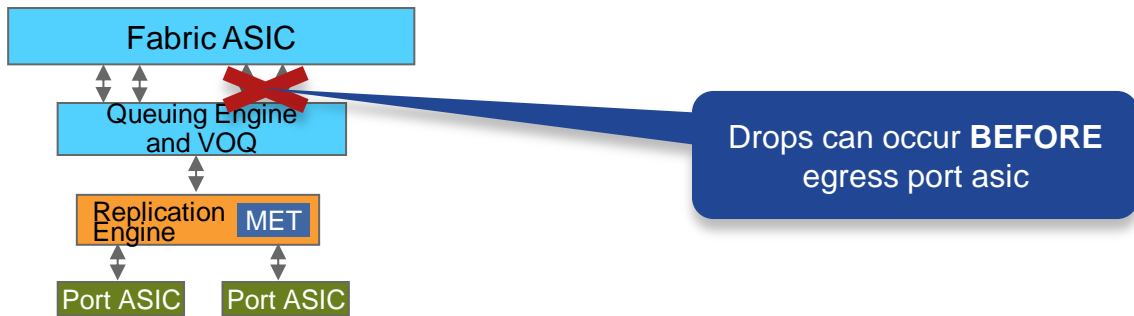
M1

```
Nexus# slot 3 show hardware skytrain queuing drops
Source Intf Traffic Type Drop Reason Count

eth 2/23 Unicast VOQ tail-drop 94980
```

M2

# Data-Plane Troubleshooting Intermittent Packet Loss: M-Series Multicast Egress Drops



- If Mcast traffic is great enough, **may not see drops in policy-map or as outdiscards.**

```
Nexus-1# slot 3 show hardware internal statistics device qengine congestion port 32 | grep MB
320 MB FI0 packet drop count 0000000005115220 1-48 I1
321 MB FI1 packet drop count 0000000000212321 1-48 I1
322 MB FI2 packet drop count 0000000005541641 1-48 I1
323 MB FI3 packet drop count 0000000010050874 1-48 I1
```

**MB = Multicast Buffer  
FI = Fabric Interface**

# M1+M2 Drop Counter Cheat Sheet:



| CLI                                                                                               | Module | Description                                                                          |
|---------------------------------------------------------------------------------------------------|--------|--------------------------------------------------------------------------------------|
| <code>show interface counters errors</code>                                                       | Both   | Basic in/outdiscards, CRC, FCS, etc errors                                           |
| <code>show policy-map interface</code>                                                            | Both   | Per-queue drops help identify which CoS traffic is affected                          |
| <code>show system internal qos queuing stats interface</code>                                     | Both   | Per-queue WRED and Taildrop counters                                                 |
| <code>slot &lt;x&gt; show hardware internal statistics device qengine congestion</code>           | M1     | Ingress Unicast VoQ drops due to no credits available b/c of egress congestion       |
| <code>slot &lt;x&gt; show hardware skytrain queuing drops</code>                                  | M2     | Ingress Unicast VoQ drops due to no credits available b/c of egress congestion       |
| <code>slot &lt;x&gt; show hardware internal statistics device qengine congestion   grep MB</code> | M1     | Egress Multicast drops due to upper ASIC congestion prior to port-ASIC.              |
| <code>clear statistics module-all device all</code>                                               | All    | Clear all hardware counters registers for monitoring purposes                        |
| <code>show tech module</code><br><code>show tech module all</code>                                | All    | Collect all hardware statistics for a particular module or all modules in the switch |

# F2/F2E/F3 Packet Counters



# Data-Plane Troubleshooting Intermittent Packet Loss: F-Series Per-Queue Counters

- Per-Interface and Per-Queue counters for Ucast & Mcast...
- Ingress discards for credited traffic, output drops for uncredited (**Different than M-Series**)

```
Nexus-1# show policy-map interface ethernet x/y | eg -i class|drop
Class-map (queuing): 2q4t-8e-in-q1 (match-any)
 queue dropped pkts : 0
Class-map (queuing): 2q4t-8e-in-q-default (match-any)
 queue dropped pkts : 0
Class-map (queuing): 1p3q1t-8e-out-pq1 (match-any)
 queue dropped pkts : 0
```

Per-queue drops help identify which CoS traffic is affected

```
Nexus-1# show system internal qos queuing stats interface x/y
Receive queues
```

```

Queue 2q4t-in-q-default
 Total packets 0
 WRED drops 0
 Taildrop drops 0
```

```
Transmit queues
```

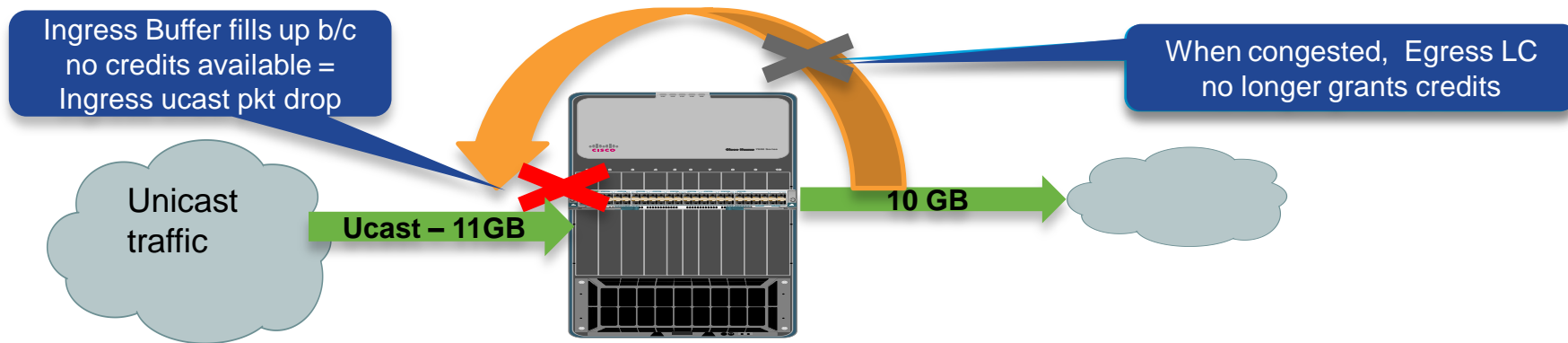
```

Queue 1p3q4t-out-q-default
 Total packets 0
 WRED drops 0
 Taildrop drops 0
```

Shows more detailed queue counter/drop info

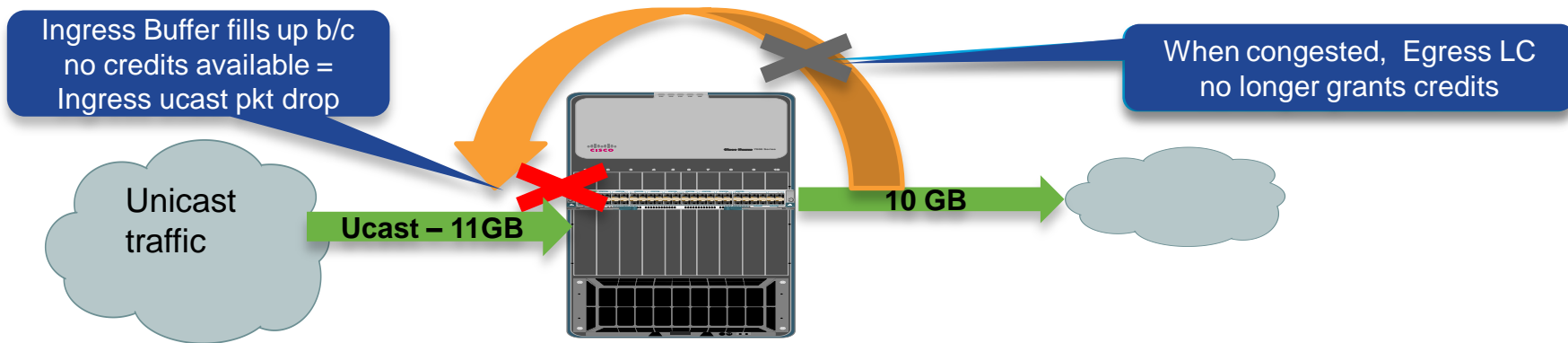


# Data-Plane Troubleshooting Intermittent Packet Loss: F-Series Ucast Congestion



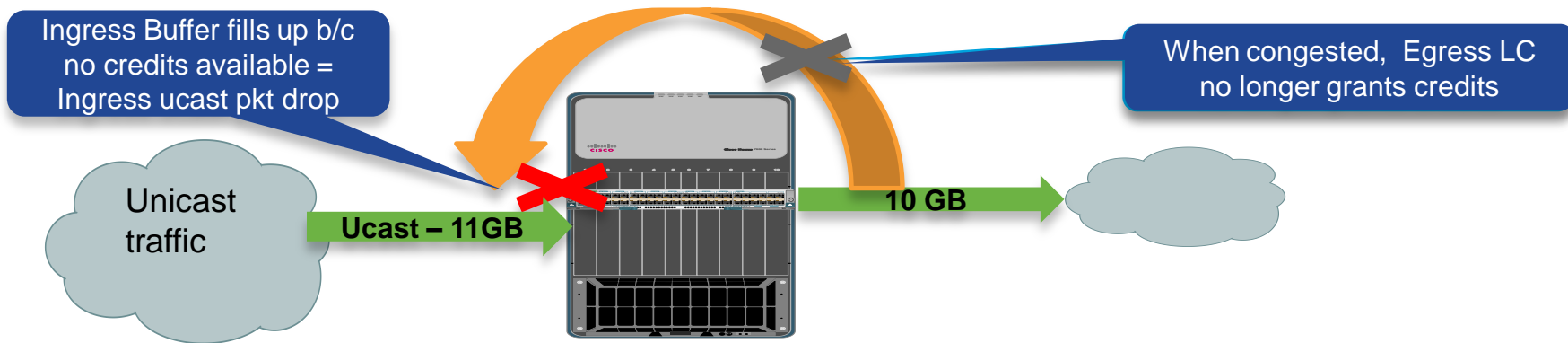
- **Unicast** (credited) traffic egress port congestion = no credits given
- **Ingress buffers** will fill up when **no credits** are given causing packet drops.

# Data-Plane Troubleshooting Intermittent Packet Loss: F-Series Ucast Congestion



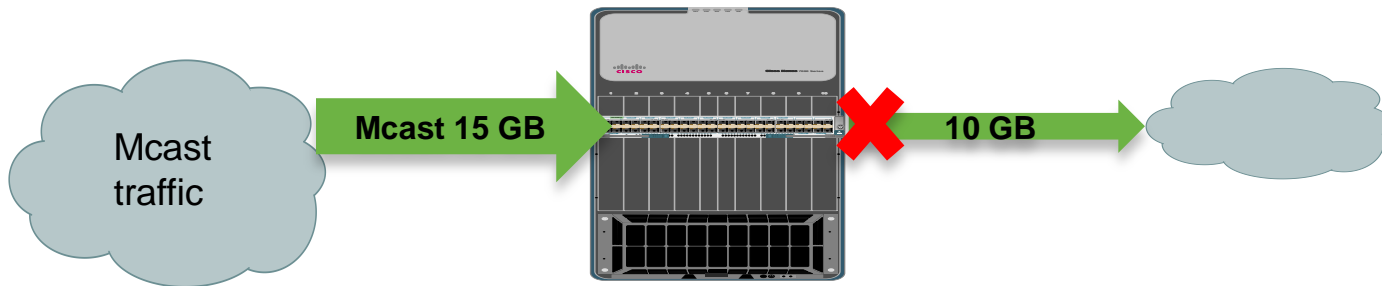
```
Nexus-7K# show int e 10/30 | eg -i discard
0 input with dribble 974183 input discard
```

# Data-Plane Troubleshooting Intermittent Packet Loss: F-Series Ucast Congestion



```
Nexus-7K# show policy-map interface ethernet 10/30 input | eg -i que|drop
Service-policy (queuing) input: default-4q-8e-in-policy
Class-map (queuing): 2q4t-8e-in-q1 (match-any)
queue dropped pkts : 0
Class-map (queuing): 2q4t-8e-in-q-default (match-any)
queue dropped pkts : 974183
```

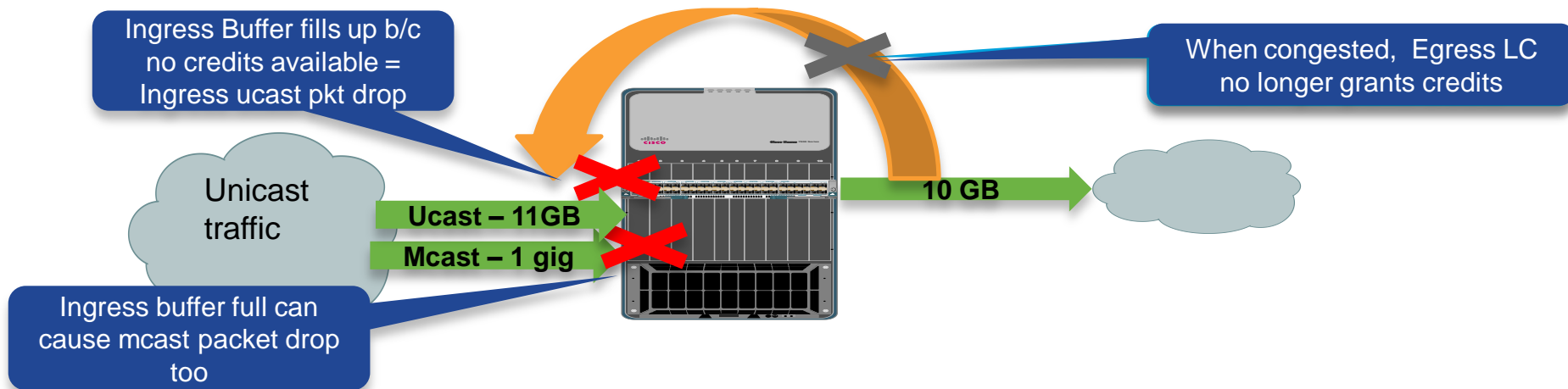
# Data-Plane Troubleshooting Intermittent Packet Loss: F-Series Multicast Congestion



- Oversubscription by **multicast / uncredited** packets, dropped on **egress** buffer
- Per-interface / Per-queue counters available

```
Nexus# show policy-map interface e10/1 output | eg -i drop|queu
Service-policy (queuing) output: default-4q-8e-out-policy
<snip>
Class-map (queuing): 1p3q1t-8e-out-q-default (match-any)
queue dropped pkts : 29851887
```

# Data-Plane Troubleshooting Intermittent Packet Loss: F-Series Ucast Congestion affects Mcast



- **Unicast** (credited) traffic egress port congestion = no credits given
- **Ingress buffers** will fill up when **no credits** are given causing packet drops.
- Mcast Packets are also now dropped on ingress instead of egress
- **May cause red herring if you don't see egress drops**

# Data-Plane Troubleshooting Intermittent Packet Loss: F-Series Ucast Congestion

- No drops at egress direction may be misleading. Don't stop there...

```
Nexus-7K# show policy-map interface e10/1 output | eg -i drop
 queue dropped pkts : 0
 queue dropped pkts : 0
```

- We also need to look at **ingress counters**, considering ucast congestion scenario

```
Nexus-7K# show int e 10/30 | eg -i discard
 0 input with dribble 101593 input discard
```

```
Nexus-7K# show policy-map interface ethernet 10/30 input | eg -i que|drop
Service-policy (queuing) input: default-4q-8e-in-policy
Class-map (queuing): 2q4t-8e-in-q1 (match-any)
 queue dropped pkts : 0
Class-map (queuing): 2q4t-8e-in-q-default (match-any)
 queue dropped pkts : 107543
```

- F2 May be difficult to identify congested egress port (**improved detection on F2E & F3**)

# Data-Plane Troubleshooting Intermittent Packet Loss: F2E/F3 Optimizations

Where F2 and F2e/F3 Differ...

# Data-Plane Troubleshooting Intermittent Packet Loss: F2E Egress Port Congestion

- F2 HW limitation cannot define VOQ drops
- F2E/F3 Provides ability to look deeper into congestion counters
  - Help to **identify egress buffer congestion backing up ingress port.**
  - **Only available in Admin VDC**

```
Sacamano-Default-VDC# show hardware queuing drops egress
```

## VQ Drops

| Output Interface | VQ Drops         | VQ Congestion    | Src Mod | Src Inst | Input Interface |
|------------------|------------------|------------------|---------|----------|-----------------|
| Eth1/1           | 0000000000393210 | 0000000000000000 | 1       | 7        | Eth1/29-32      |

VOQ drops are due to unicast /credited traffic

## Egress Buffer Drops

| Output Interface | EB Drops         |
|------------------|------------------|
| Eth3/21-24       | 0000000000001200 |

EB Drops are due to mcast / uncredited traffic



# F2 / F2e / F3 Drop Counter Cheat Sheet:



| CLI                                                                | Module | Description                                                                          |
|--------------------------------------------------------------------|--------|--------------------------------------------------------------------------------------|
| <code>show interface counters errors</code>                        | All    | Basic in/out discards, CRC, FCS, etc errors                                          |
| <code>show policy-map interface</code>                             | All    | Per-queue drops help identify which CoS traffic is affected                          |
| <code>show system internal qos queuing stats interface</code>      | All    | Per-queue WRED and Taildrop counters                                                 |
| <code>show hardware queuing drops egress module</code>             | F2e/F3 | Ingress Unicast VoQ drops due to no credits available b/c of egress congestion       |
| <code>clear statistics module-all device all</code>                | All    | Clear all hardware counters registers for monitoring purposes                        |
| <code>show tech module</code><br><code>show tech module all</code> | All    | Collect all hardware statistics for a particular module or all modules in the switch |

# Data-Plane Troubleshooting Performance Drops: Cheat Sheet



## N7K Counter Commands

- show interface x/y counters errors
- show interface x/y counters detail all
- show system internal qos queuing stats interface x/y
- show policy-map interface ethernet x/y
- slot (x) show hardware internal statist dev qengine congest port x | grep MB (M1 Only)
- slot (x) show hardware internal skytrain queuing drops (M2 Only)
- show hardware queuing drops egress (F2E / F3 only)
- Slot (X) show hardware internal statistics pktflow dropped congestion
- Clear statistics module-all device all (clear all statistics counters)

## N7K Relevant Show Techs

- Show tech module (X)
- Show tech module all

# *Summary*

# What we have covered today

## Summary

- Tools (CLI, Logging, and Scripts, Packet Capturing Tools)
- Control Plane (Processes, Inband-Path, RL, CoPP)
- Layer 2 Data Plane (L2FM, PIXM, MAC-Addresses, LTL and CBL)
- Layer 3 Data Plane (URIB and UFDM)
- Traffic Filter (CL-TCAM, LOU and Result Types and Bank Mapping)

Thank you.

