
TOMORROW starts here.
cisco
Cisco live!

Overview of Troubleshooting Tools in Cisco Switches and Routers

BRKARC-2011

Ed Kim

High Touch Engineer, Cisco Services

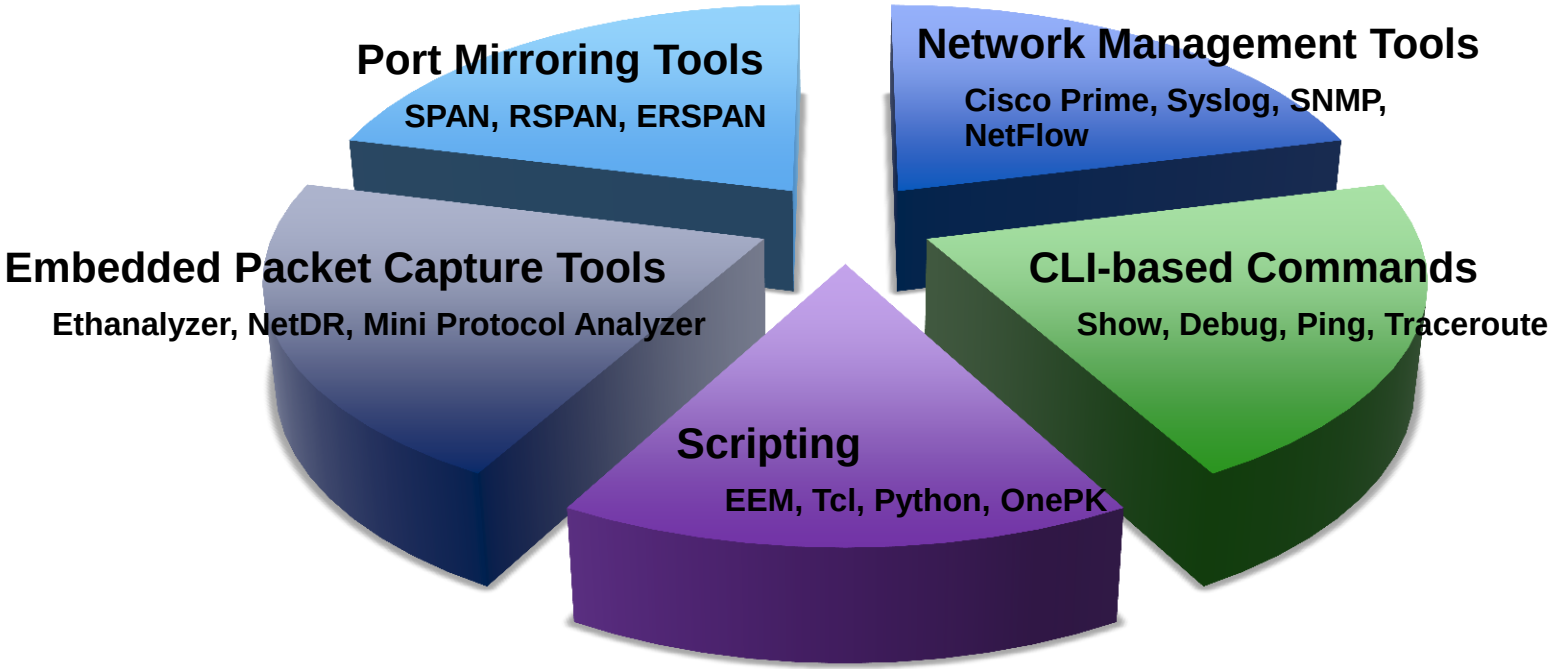
Cisco live!

Agenda

- The Value of Good Incident Management
- Flexible NetFlow (FnF)
- Embedded Packet Capture Tools
- SPAN / RSPAN / ERSPAN
- Ethanalyzer and NetDR
- VACL and ACL Capture
- ELAM
- Putting It All Together

"Nothing has changed, and anything we did change we changed back!"

Taxonomy of the Troubleshooting Tools

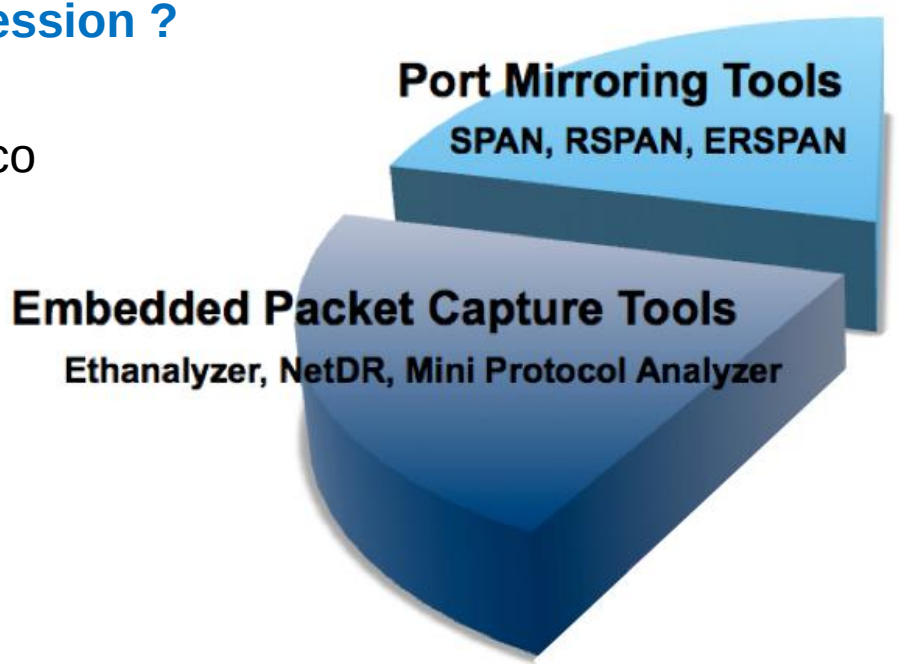


Our Focus ….

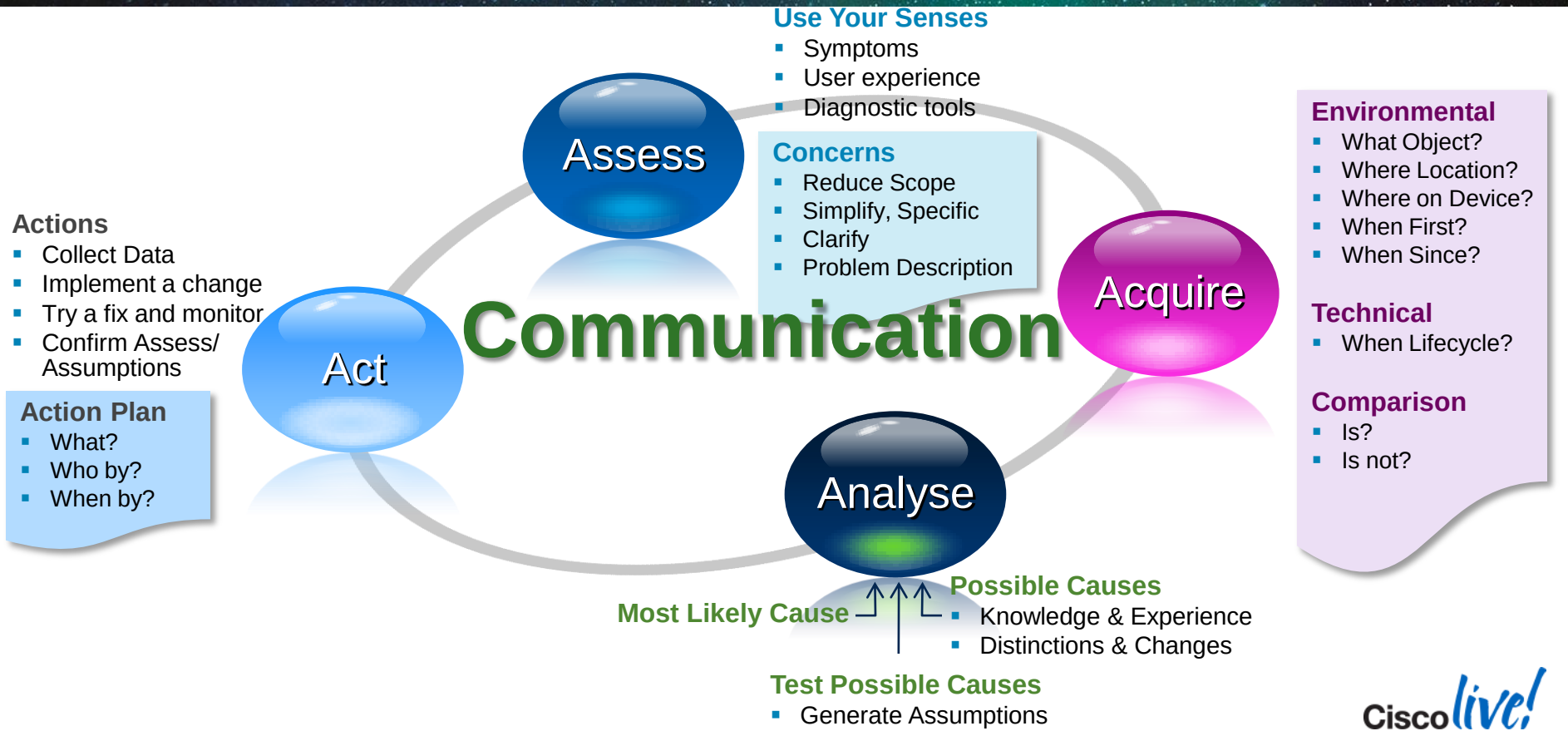
What I am going to get out of this session ?

- Good idea of the capabilities of Cisco switches/routers in terms of various troubleshooting tools, especially packet capture tools.

- Ability to choose the right tool to troubleshoot, which helps for timely resolution of the problem.



Cisco TAC Discipline



Acronyms / Definitions

Acronyms	Definitions	Acronyms	Definitions
FnF	Flexible NetFlow	SP	Switch Processor
EPC	Embedded Packet Capture	RP	Route Processor
MPA	Mini Protocol Analyzer	VDC	Virtual Device Context
SPAN	Switch Port Analyzer	FE	Forwarding Engine
RSPAN	Remote SPAN	CFC	Centralized Forwarding Card
ERSPAN	Encapsulated RSPAN	DFC	Distributed Forwarding Card
CEF	Cisco Express Forwarding	LTL	Local Target Logic
ACL	Access Control List	DBUS / RBUS	Data Bus / Result Bus
VACL	Vlan-based ACL	VSS	Virtual Switching System
RACL	Router-based ACL	ASIC	Application Specific Integrated Circuit
PACL	Port-based ACL	ELAM	Embedded Logic Analyzer Module
NetDR	Net Driver	CoPP	Control Plane Policing

➔ Reference Slide

Flexible NetFlow (FnF)

Flexible NetFlow

The challenge and the need for Flexible NetFlow

- Monitoring IP traffic flows facilitates:
  - More accurate capacity planning
  - Optimising network, ranging from traffic engineering and understanding network detailed behaviour
  - Implementing new IP services and applications with confidence

- The challenge, however, is finding a scalable, manageable, and reliable solution to provide the necessary data to support these opportunities

Flexible NetFlow addresses the challenge!

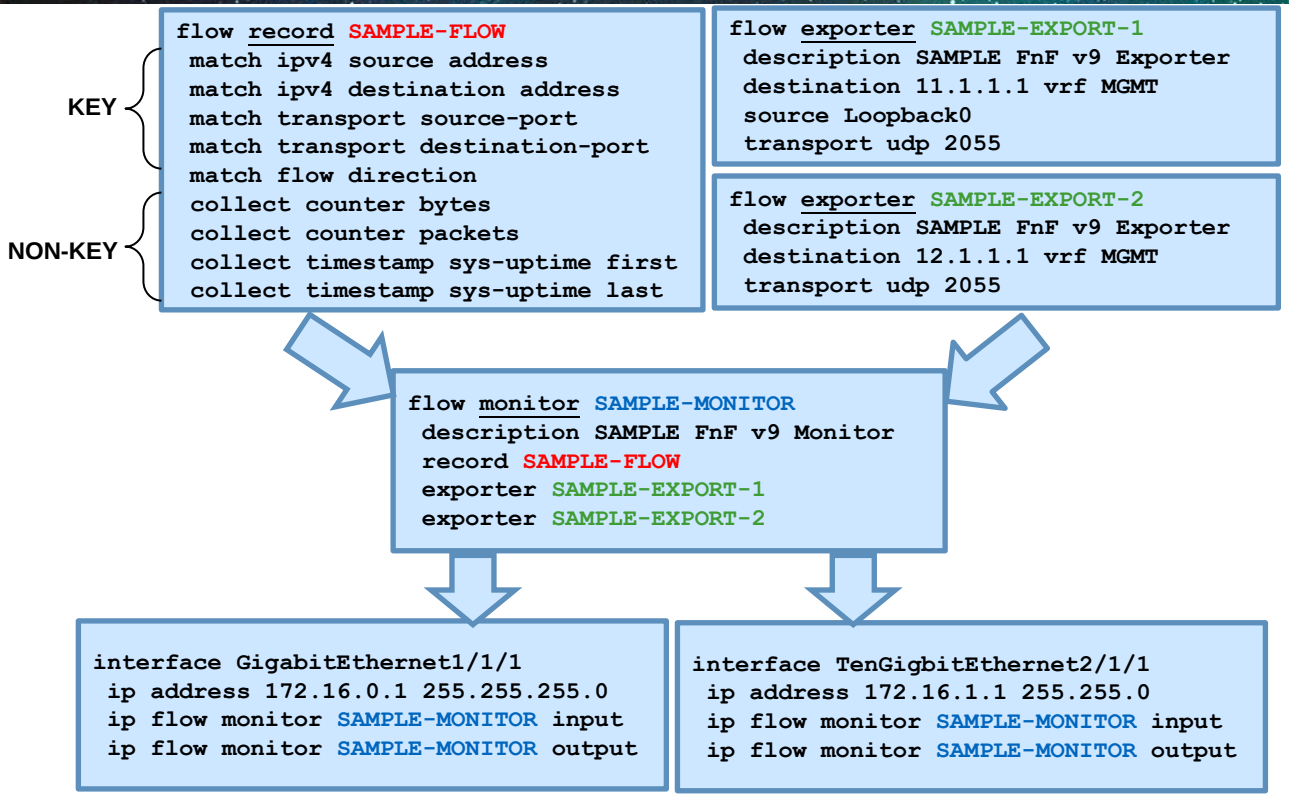
Flexible NetFlow

Key Advantages to using Flexible NetFlow

- Flexibility and scalability of flow data beyond traditional NetFlow
- The ability to monitor a wider range of packet information producing new information about network behaviour not available today
- Enhanced network anomaly and security detection
- User configurable flow information to perform customised traffic identification and the ability to focus and monitor specific network behaviour

Flexible NetFlow

Configuration

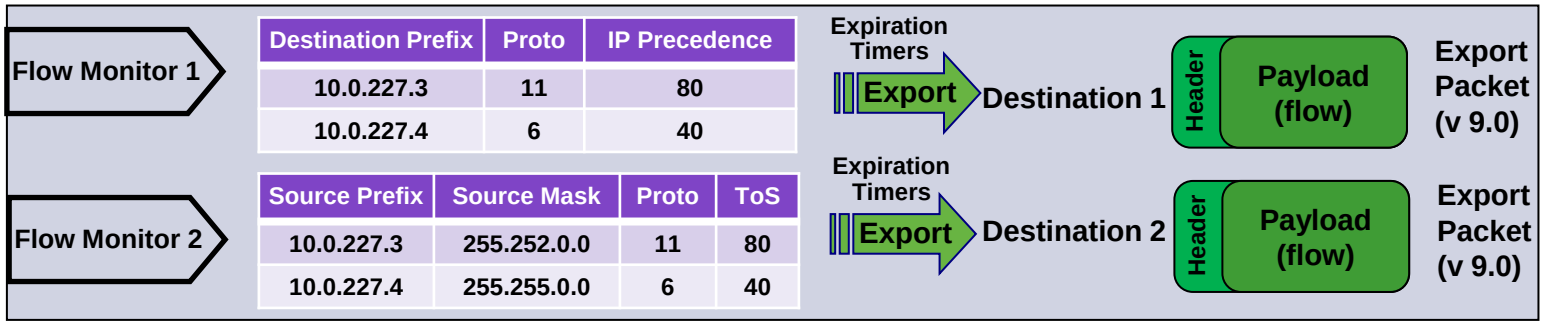


Steps:

1. Create Flow Record
2. Create Flow Exporter
3. Associate Record and Exporter to a Flow Monitor
4. Apply to the interfaces

Flexible NetFlow

Data Export



- The export versions are well documented (data export) formats including version 5 (most commonly used), 7 and 9
- NetFlow export version 9 is the latest Cisco invented format and has some advantages (compared to other versions) for key technologies such as security, traffic analysis and multicast

Without version 9 export format, Flexible NetFlow would not be possible !!

Flexible NetFlow (FnF)

Supported Platforms and Software Releases

IOS-XE

- ISR4451-X ➔ 3.8.0S
- ASR100x ➔ 3.8.0S
- Cat3850 ➔ 3.2.0SE

NX-OS

- Nexus7000 ➔ 4.x / 5.x / 6.x

IOS-XR

- ASR9000 ➔ 4.1 / 4.2 / 4.3

IOS

- Cisco 800 series ➔ 12.4T / 15.1T / 15.0M releases
- Cisco 1800/1900 series ➔ 15.0M / 15.1T
- Cisco 2800/2900 series ➔ 15.0M / 15.1M / 15.1T / 15.2T
- Cisco 3800/3900 series ➔ 15.0M / 15.1M / 15.1T / 15.2T / 12.4T / 12.4XY / 12.4XW
- Cat6K-Sup2T ➔12.2(50)SY / 15.0(1)SY releases

For more details on platforms and software releases supporting FnF, please use Feature Navigator tool available at CCO: http://www.cisco.com/go/fn

Real World Example

Resolving High CPU using FnF

```
Sup2T# show process cpu sorted
CPU utilization for five seconds:  65%/8%; one minute: 63%; five minutes: 61%
 PID  Runtime(ms)   Invoked  uSecs     5Sec    1Min    5Min TTY Process
 310        30544    189234     81   47.12%  45.11% 45.23%  0  IP Input
```

High CPU due to process “IP Input”

```
Sup2T(config)#flow RECORD RP-FnF-CEF-Receive-record
Sup2T(config-flow-record)#match ipv4 protocol
Sup2T(config-flow-record)#match ipv4 source address
Sup2T(config-flow-record)#match ipv4 destination address
Sup2T(config-flow-record)#match transport source-port
Sup2T(config-flow-record)#match transport destination-port
Sup2T(config-flow-record)#collect interface input
Sup2T(config-flow-record)#collect counter packets
Sup2T(config-flow-record)#exit
```

Building a FnF record, matching L3 and L4 parameters (key fields) and collecting details on Input interface and packet count (non-key fields)

```
Sup2T(config)#flow MONITOR RP-FnF-CEF-Receive
Sup2T(config-flow-monitor)#record RP-FnF-CEF-Receive-record
Sup2T(config-flow-monitor)#exit
```

Associating the FnF record to a monitor. Here, you can add an option (not enabled here) to export the data to the collector

```
Sup2T(config)#control-plane
Sup2T(config-cp)#ip flow monitor RP-FnF-CEF-Receive input
Sup2T(config-cp)#exit
```

Applying to the control-plane interface

Real World Example

Monitoring Control-Plane traffic using FnF

```
Sup2T# show flow monitor copp-fnf-cef-receive cache sort counter packet
Processed 5 flows
Aggregated to 5 flows
Showing the top 5 flows

IPV4 SOURCE ADDRESS:         192.107.16.40
IPV4 DESTINATION ADDRESS:    192.168.88.1
TRNS SOURCE PORT:            48827
TRNS DESTINATION PORT:       63
IP PROTOCOL:                 17
interface input:             Vl40
counter packets:             460983
<snip>
```

First flow with high number of packets hitting the CPU

Results sorted according to the number of packets per flow.
Note: Some of the router platforms may not support "sort" option. If so, use sorting functions available in the NetFlow Data Collector applications.

After few seconds…

```
Sup2T# show flow mon copp-fnf-cef-receive cache sort count pack
<snip>
IPV4 SOURCE ADDRESS:         192.107.16.40
IPV4 DESTINATION ADDRESS:    192.168.88.1
TRNS SOURCE PORT:            48827
TRNS DESTINATION PORT:       63
IP PROTOCOL:                 17
interface input:             Vl40
counter packets:             461181
```

Make sure the counters are increasing

Once the flow is identified, further action could be (1) blocking the flow with an Access List (ACL) or (2) rate-limiting it using Control Plane Policing (CoPP) depending on the criticality of the flow to the production.

Embedded Packet Capture Tools

Embedded Packet Capture Tools

- Embedded Packet Capture (EPC)
- Mini Protocol Analyzer (MPA)
- Wireshark

Embedded Packet Capture Tools

Overview

- Embedded Packet Capture (EPC):
  - Allows for packet data to be captured at various points in the CEF packet-processing path; flowing through, to and from a Cisco router
  - Supported in Cisco Routers like Cisco 900, 1900, ASR 1000 etc.
- Mini-Protocol Analyzer (MPA) – An extension of EPC:
  - Uses a SPAN session to capture the traffic
  - Allows for packet data to be captured at various points in a hardware-forwarding device like Cisco 7600, Catalyst 6500 and ME6500 platforms
- Wireshark:
  - Allows for packet data to be captured at various points in the packet-processing path; flowing through, to and from a Catalyst 4500 switch, with a Sup7E running 3.3SG or 3.4SG

Embedded Packet Capture Tools

Key Advantages and Benefits – EPC and MPA

- Exec-level commands to start and stop the capture, define buffer size, buffer type (linear or circular) and packet size to capture
- Facility to export the packet capture in PCAP format suitable for analysis
- Useful when it is not possible to tap into the network using a stand-alone packet-sniffing tool, or when need arises to remotely debug and troubleshoot issues
- Capture rate can be throttled using further administrative controls. For example, using an Access Control List (ACL), specify maximum packet capture rate or specify a sampling interval
- Show commands to display packet contents on the device itself

Embedded Packet Capture Tools

Key Advantages and Benefits - Wireshark

- Catalyst 4500 series switch supports **SPAN** and **debug platform packet**. SPAN provides no local display or analysis support, as it exports the captured packets to some specified local or remote destination. The debug platform packet command works on packets that are software-processed, with no analysis support. Wireshark provides strong packet capture, display and analysis support.
- Most of the advantages and benefits mentioned for EPC / MPA

Embedded Packet Capture (EPC)

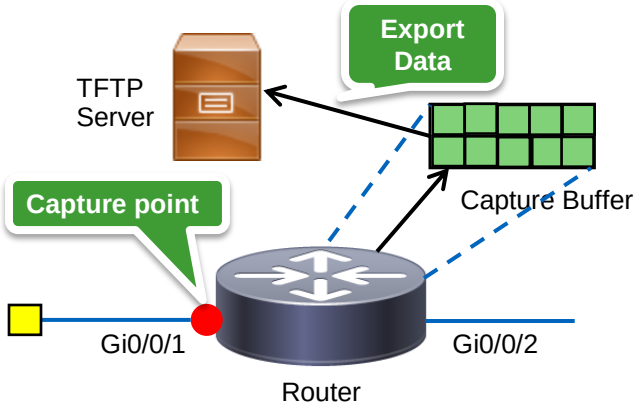
Configuration Steps

In ASR 1002 running IOS-XE 15.3(2)S / 3.9(0)S release:

```
Router# monitor capture MYCAP buffer circular packets 10000
Router# monitor capture MYCAP buffer size 10
Router# monitor capture MYCAP interface Gig0/0/1 in
Router# monitor capture MYCAP access-list MYACL
Router# monitor capture MYCAP start
Router# monitor capture MYCAP stop
Router# monitor capture MYCAP export bootflash:EPC1.pcap
```

In Cisco 7206VXR running 12.4(4)XD

```
Router# mon cap buffer MYBUF size 128 max-size 256 circular
Router# mon cap buffer MYBUF filter access-list MYACL
Router# mon cap point ip cef IPCEFCAP Gig0/0/1 both
Router# mon cap point associate IPCEFCAP MYBUF
Router# mon cap point start IPCEFCAP
Router# mon cap point stop IPCEFCAP
Router# mon cap buffer MYBUF export tftp://1.1.1.1/EPC1.pcap
```



Steps to Configure:

1. Define capture buffer
2. Define capture point
3. Associate capture buffer and point (depends on the platform and OS ver)
4. Capture data
5. Export / display captured data

Embedded Packet Capture (EPC)

Analysing the traffic on the device

```
ASR# show monitor capture CAP parameter
   monitor capture CAP interface Gig0/0/2 both
   monitor capture CAP access-list test
   monitor capture CAP buffer size 10
   monitor capture CAP limit pps 1000

ASR# show mon cap CAP buffer
 buffer size (KB) : 10240
 buffer used (KB) : 128
 packets in buf   : 5
 packets dropped  : 0
 packets per sec  : 1
```

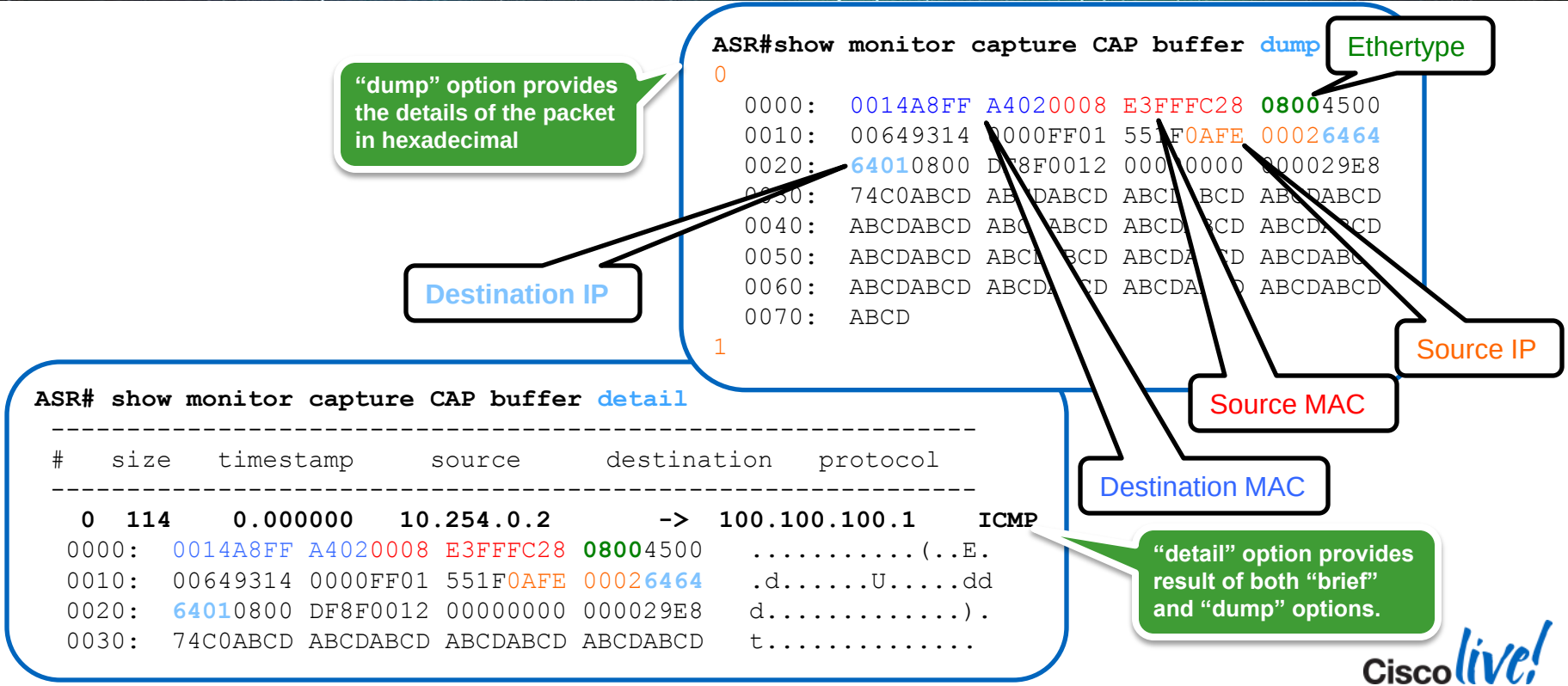
Indicates total number of packets in the capture buffer

```
ASR# show monitor capture CAP buffer ?
  brief     brief display
  detailed  detailed display
  dump      for dump
  |         Output modifiers
  <cr>
ASR# show monitor capture CAP buffer brief ----------------------
-----------------------------------------
 #   size   timestamp     source              destination   protocol
-----------------------------------------------------------------------
   0  114    0.000000   10.254.0.2        ->  100.100.100.1    ICMP
   1  114    0.000992   10.254.0.2        ->  100.100.100.1    ICMP
   2  114    2.000992   10.254.0.2        ->  100.100.100.1    ICMP
```

"brief" option provides basic information of the traffic like source/destination IP address, Protocol type, packet length.

Embedded Packet Capture (EPC)

Analysing the traffic on the device



Embedded Packet Capture (EPC)

Supported Platforms and Software Releases

IOS-XE

ASR1000 ➔ 3.7S / 3.8S / 3.9S / 15.2S / 15.4T

IOS

Cisco 800/1800/1900/2800/2900 series ➔ 15.0M / 15.1M / 15.2M / 15.1T / 15.2T

Cisco 3800 series ➔ 12.4T / 12.4YA / 15.1T / 15.0M / 15.1M / 15.1XB

Cisco 3900 series ➔ 12.4T / 15.1T / 15.2T / 15.3T / 15.1M / 15.2M

Cisco 7200/7300 series ➔ 12.4T / 15.0M / 15.1M / 15.2M

Mini Protocol Analyzer (MPA)

Configuration Steps

In a Catalyst 6500 switch:

```
C6K# config t
C6K(config)# monitor session 1 type capture
C6K(config-mon-capture)#source vlan 10, 12-13
C6K(config-mon-capture)#filter access-group 99
C6K(config-mon-capture)#end
C6K#
C6K# monitor capture buffer size 1000 length 128 linear
C6K# monitor capture start for 10000 packet
C6K# monitor capture stop
```

Configure a SPAN session, with type “capture”

Identify the source (either an interface, port-channel or Vlan). By default, it capture traffic in both directions.

Configure a filter with an ACL, mac-address, VLAN, Ether-Type or packet length with range.

Configure buffer size and packet length, and capture type. Default buffer size is 2048 KB, default 68 Bytes and type is linear

Start the capture. Capture is set to capture 10,000 packets

Stop the capture manually, if needed

Mini Protocol Analyzer (MPA)

Analysing the traffic on the device

```
C6K# show monitor capture buffer ?
  <1-4294967295>  start index
  acl             filter output of captured Packets
  brief           Brief output of captured Packets
  detail          Detailed output of captured Packets
  dump            Hex Dump of captured Packets
  |               Output modifiers
  <cr>

C6K# show monitor capture buffer brief
 1      IP: s=10.0.10.72 , d=10.62.12.10, len 46
 2      IP: s=10.0.10.72 , d=10.62.12.10, len 46
 <snip>

C6K# show monitor capture buffer detail
 1       Arrival time : 14:19:04.520 UTC Mon Apr 15 2013
            Packet Length : 60 , Capture Length : 60
            Ethernet II :  0000.0c9f.f00a  0000.0000.000a 0800
            IP: s=10.0.10.72 , d=10.62.12.10, len 46
            TCP src=8963, dst=2000, seq=44303, ack=16301, win=0
```

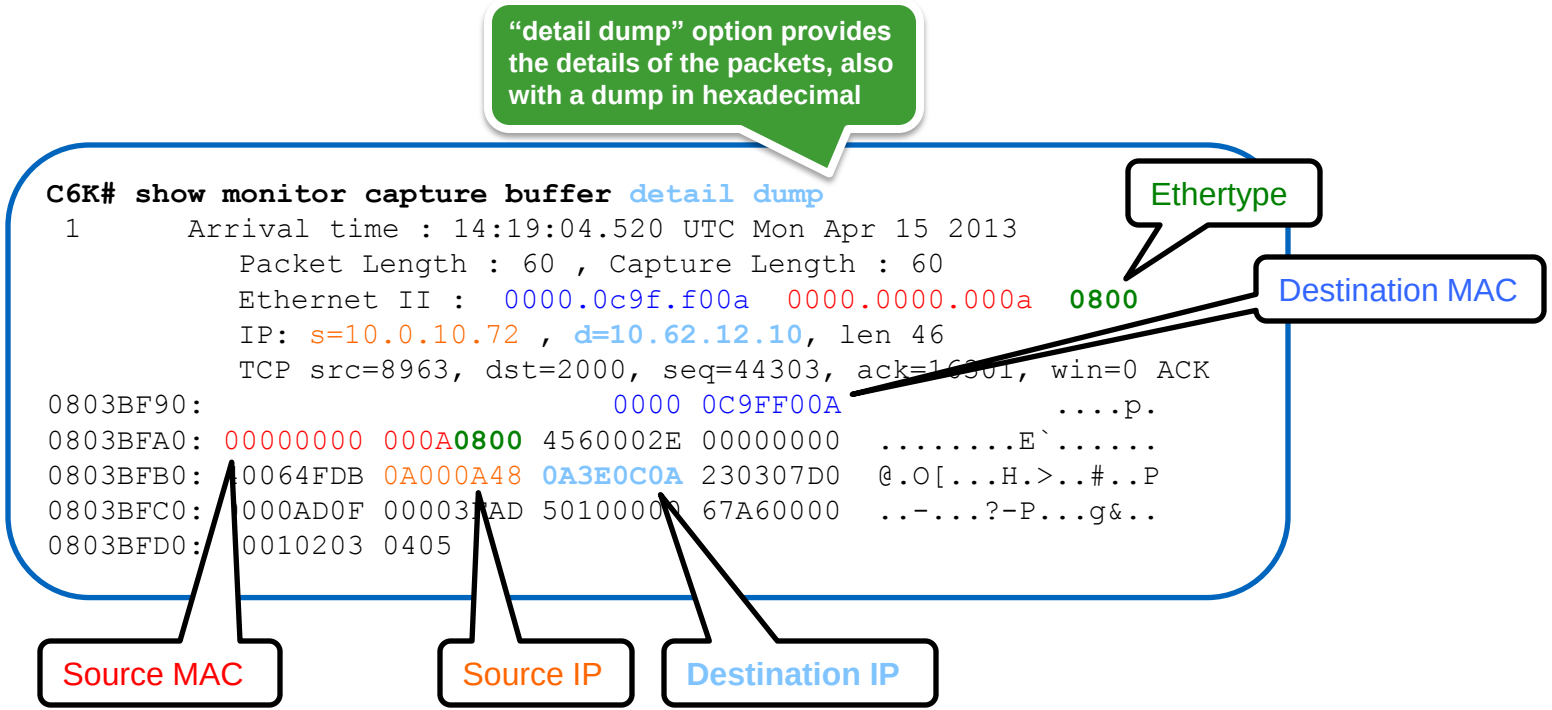
"acl" helps to filter the packets displayed by the command

"brief" option provides the basic info of the packets in buffer

"detail" option provides the detailed info of the packets such as packet length, MAC addresses, IP addresses and TCP/UDP info.

Mini Protocol Analyzer (MPA)

Analysing the traffic on the device



Mini Protocol Analyzer (MPA)

Supported Platforms and Software Releases

IOS

- ME6500 ➔ 12.2SXI
- Cat6K-Sup720 / VS-S720 ➔ 12.2SXI
- Cat6K-Sup2T ➔12.2SY / 15.0SY
- Cisco 7600 series ➔ 12.2 SRD / 12.2 SRE / 15.0S / 15.1S / 15.2S

Wireshark

Configuration Steps and traffic Analysis

```
C4500X# monitor capture TESTCAP vlan 100 both
C4500X# monitor capture TESTCAP file location
bootdisk:testcap.pcap
C4500X# monitor capture TESTCAP buffer size 100
C4500X# monitor capture TESTCAP match ipv4 proto tcp eq 80
C4500X# monitor capture TESTCAP start
C4500X# monitor capture TESTCAP stop
```

Define a capture point. It could be an interface, Vlan or control-plane

Define a location to build a .pcap file

Buffer size (in MB) to dump the captured data

Start / stop the capture

Inline-filter matching with protocol type, L4 port number

Start a capture with capture or display filter:

```
C4500X# monitor capture TESTCAP start capture-filter "host 10.20.30.1 and port 80"
C4500X# monitor capture TESTCAP start display display-filter "host 10.20.30.1 and port 80"
```

Display options ….

```
C4500X#show monitor capture TESTCAP buffer ?
  brief           brief display
  detailed        detailed display
  display-filter  Display filter
  dump            for dump
  |               Output modifiers
```

Just like in EPC / MPA display options, Wireshark supports "brief", "detail" and "dump" options, with similar results.

Embedded Packet Capture Tools

Limitations and Restrictions – EPC / MPA

EPC:

- This feature captures multicast traffic only on ingress, not the replicated packets on egress
- In some of the Cisco platforms, EPC can be done only on one interface at a given time.

Mini Protocol Analyzer:

- Only one session possible at any given time, and uses a SPAN session
- To control the CPU usage, it is strongly recommended to use filters to limit the traffic to the CPU

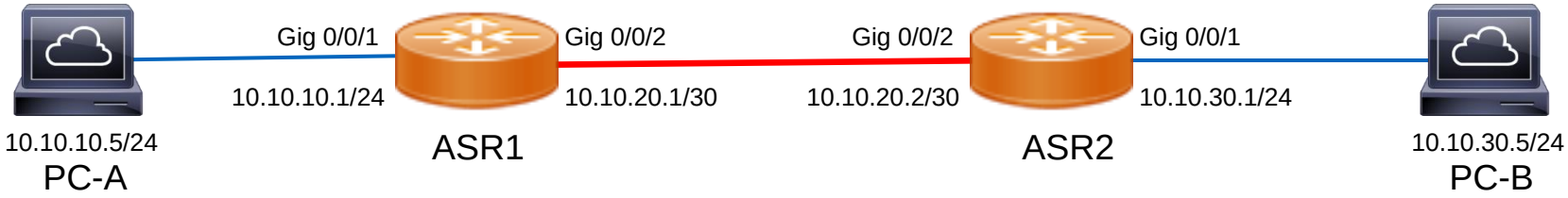
Embedded Packet Capture Tools

Limitations and Restrictions - Wireshark

- When packet capture is enabled in the input direction, the matching packets undergo software-based lookup in CPU for the first 15 seconds.
- When packet capture is enabled in the output direction, packets are not captured in the first 15 seconds, and the captured packets may not reflect the changes made by the rewrite process (e.g., TTL, VLAN tag, MAC addresses).
- Wireshark cannot capture IPv6 packets if the capture point's class-map filter is attempting to match one of the following: Extension headers followed by Hop-by-hop header and DSCP values.
- Wireshark is not supported for Management interface (e.g., FastEthernet 1) or on an interface that belongs to a logical group (i.e., capturing at a physical port that is member of a port-channel)

Real World Example

Isolating the device causing packet loss

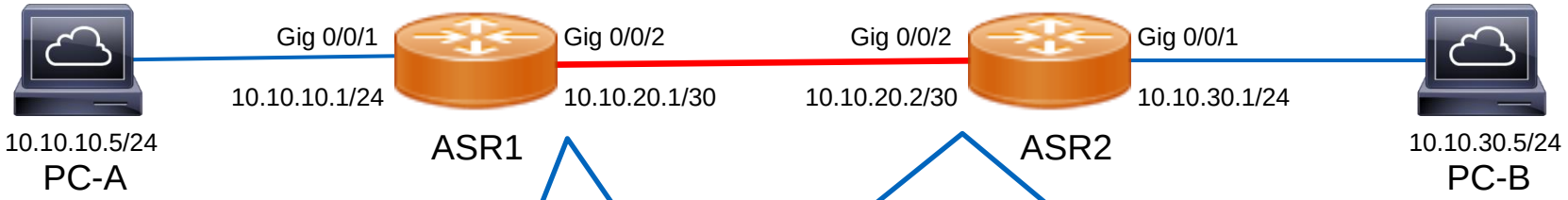



```
C:\>ping 10.10.30.5 -n 5 -l 100
Pinging 10.10.30.5 with 100 bytes of data:
Reply from 10.10.30.5: bytes=100 time<1ms TTL=126
Reply from 10.10.30.5: bytes=100 time<1ms TTL=126
Reply from 10.10.30.5: bytes=100 time<1ms TTL=126
Reply from 10.10.30.5: bytes=100 time<1ms TTL=126
Reply from 10.10.30.5: bytes=100 time<1ms TTL=126
Ping statistics for 10.10.30.5:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss)
```

```
C:\>ping 10.10.30.5 -n 5 -l 1000
Pinging 10.10.30.5 with 1000 bytes of data:
Reply from 10.10.30.5: bytes=1000 time<1ms TTL=126
Reply from 10.10.30.5: bytes=1000 time<1ms TTL=126
Request timed out.
Request timed out.
Reply from 10.10.30.5: bytes=1000 time<1ms TTL=126
Ping statistics for 10.10.30.5:
    Packets: Sent = 5, Received = 3, Lost = 2 (40% loss)
```

Real World Example

Isolating the device causing packet loss




```
! Filter ICMP traffic between hosts
ip access-list extended ICMP1
 permit icmp host 10.10.10.5 host 10.10.30.5
 permit icmp host 10.10.30.5 host 10.10.10.5

! EPC on ASR1
monitor capture CAP1 int Gig0/0/2 both access-list ICMP1
monitor capture CAP1 start
monitor capture CAP1 stop
```

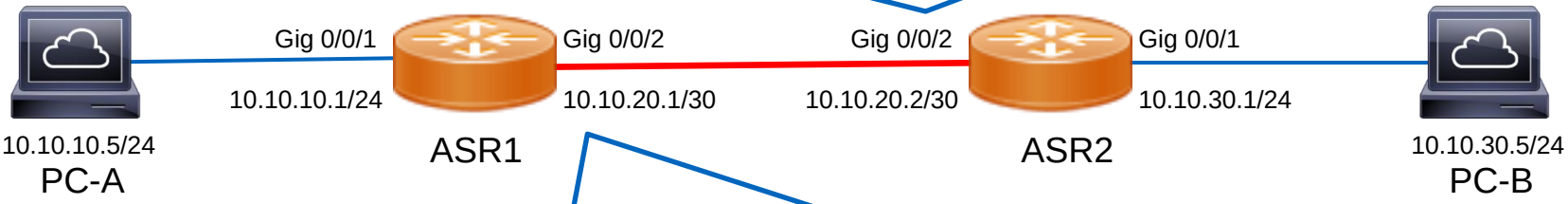
```
! Filter ICMP traffic between hosts
ip access-list extended ICMP2
 permit icmp host 10.10.10.5 host 10.10.30.5
 permit icmp host 10.10.30.5 host 10.10.10.5

! EPC on ASR2
monitor capture CAP2 int Gig0/0/2 both access-list ICMP2
monitor capture CAP2 start
monitor capture CAP2 stop
```

Real World Example

Isolating the device causing pa

```
ASR2#show monitor capture CAP2 buffer detail | inc ICMP|#
 #    size    timestamp      source           destination     protocol
 0  1014      0.000000     10.10.10.5      ->  10.10.30.5       ICMP
 1  1014      0.003997     10.10.30.5      ->  10.10.10.5        ICMP
 2  1014      0.004013     10.10.10.5      ->  10.10.30.5       ICMP
 3  1014      0.004976     10.10.30.5      ->  10.10.10.5        ICMP
 4  1014      0.005033     10.10.10.5      ->  10.10.30.5       ICMP
 5  1014      2.006834     10.10.10.5      ->  10.10.30.5       ICMP
 6  1014      4.007125     10.10.10.5      ->  10.10.30.5       ICMP
 7  1014      4.008003     10.10.30.5      ->  10.10.10.5        ICMP
```

5 Echo Requests from PC-A to PC-B.
Only 3 Echo Replies from PC-B to PC-A.



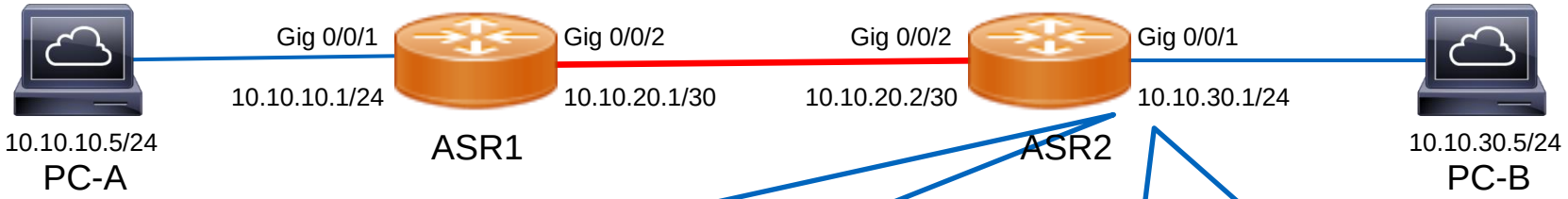

```
ASR1#show monitor capture CAP1 buffer detail | inc ICMP|#
 #    size    timestamp      source           destination     protocol
 0  1014      0.000000     10.10.10.5      ->  10.10.30.5       ICMP
 1  1014      0.003998     10.10.30.5      ->  10.10.10.5        ICMP
 2  1014      0.004012     10.10.10.5      ->  10.10.30.5       ICMP
 3  1014      0.004978     10.10.30.5      ->  10.10.10.5        ICMP
 4  1014      0.005031     10.10.10.5      ->  10.10.30.5       ICMP
 5  1014      2.006832     10.10.10.5      ->  10.10.30.5       ICMP
 6  1014      4.007124     10.10.10.5      ->  10.10.30.5       ICMP
 7  1014      4.008006     10.10.30.5      ->  10.10.10.5        ICMP
```

ASR1 has sent out 5 ICMP Echo Requests on Gig0/0/2 and received by ASR2 on Gig0/0/2.

Did all the ICMP Echo Requests sent out on Gig0/0/1 by ASR2 ? Let's find out.

Real World Example

Isolating the device causing packet loss




```
! EPC on ASR2
monitor capture CAP2 int Gig0/0/1 both access-list ICMP2
monitor capture CAP2 start
monitor capture CAP2 stop
```

```
ASR2#show monitor capture CAP2 buffer detail | inc ICMP|#
 #   size   timestamp     source           destination     protocol
   0 1014   0.000000    10.10.10.5       ->  10.10.30.5       ICMP
   1 1014   0.003814    10.10.30.5       ->  10.10.10.5       ICMP
   2 1014   0.003838    10.10.10.5       ->  10.10.30.5       ICMP
   3 1014   0.004612    10.10.30.5       ->  10.10.10.5       ICMP
   4 1014   4.008062    10.10.10.5       ->  10.10.30.5       ICMP
   5 1014   4.008822    10.10.30.5       ->  10.10.10.5       ICMP
```

EPC capture on Gig0/0/1 confirms that only 3 Echo Request from PC-A to PC-B are sent out by ASR2.

The device dropping the packets is ASR2 !!!

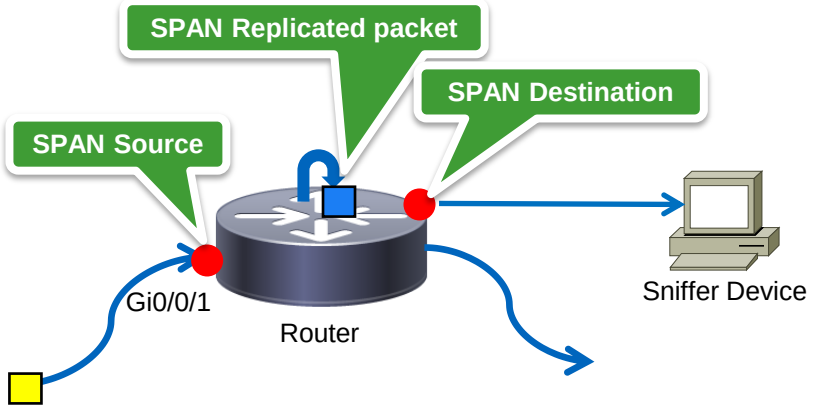



SPAN

Switch Port Analyzer (SPAN)

Overview

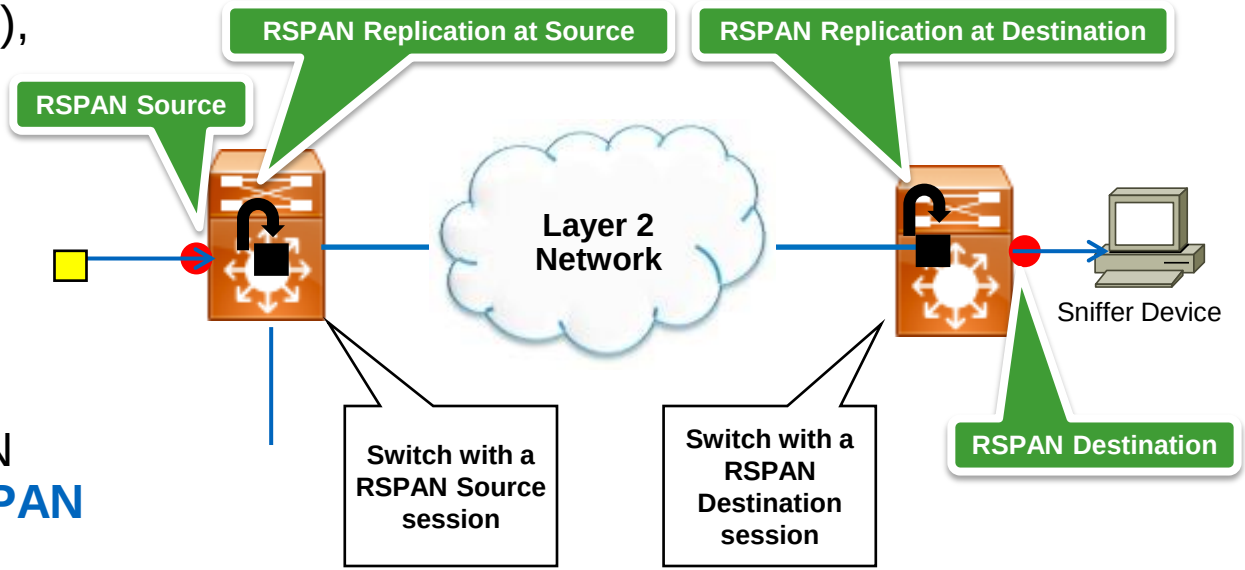
- A SPAN session (also known as port mirroring or monitoring) is an association of source ports/vlans to one or more destination ports.
- Once the traffic is identified for replication, Cisco switch/router replicates the traffic to the destination port(s).



Remote Switch Port Analyzer (RSPAN)

Overview

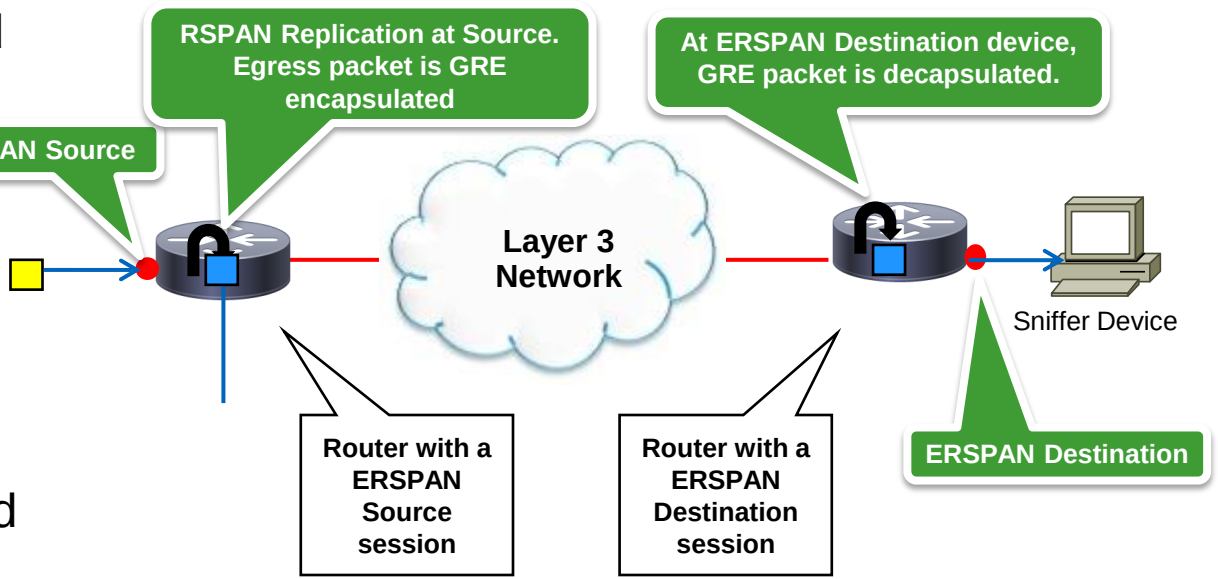
- RSPAN supports source ports (or source VLANs), and destinations on different switches,
- It uses a user-specified Layer 2 VLAN to carry SPAN traffic between switches.
- It consists of an RSPAN source session, **an RSPAN VLAN,** and an RSPAN destination session.



Encapsulated Remote SPAN (ERSPAN)

Overview

- ERSPAN supports source ports, source VLANs, and destinations on different switches
- It uses a GRE tunnel to carry traffic
- ERSPAN consists of an ERSPAN source session, **routable ERSPAN GRE-encapsulated traffic**, and an ERSPAN destination session



SPAN / RPSAN / ERSPAN

Supported Platforms and Software Releases

SPAN / RSPAN

IOS-XE:

Cat4500-Sup7E/L-E ➔ 3.2XO

Cat4500X ➔ 3.4SG

Cat3850 / 5760 WC ➔ 3.2SE

IOS:

Supported in most of the Cisco Router platforms, Catalyst switches, Blade Servers and Metro Ethernet switches.

….. and in many more platforms and OS versions

ERSPAN

IOS-XE:

ISR4451 ➔ 3.8

ASR100x ➔ 3.2S. 3.8S,

IOS:

ME6500 ➔ 12.2SX

Switch ➔ 12.2SX, 15.0SY, 15.1SY

Cisco7600 ➔ 12.2SX, 12.2SR, 15.0S, 15.1S

SPAN / RSPAN / ERSPAN

Limitations and Restrictions

- Limited number of Egress-SPAN (also called as Transmit SPAN) supported. E.g., Catalyst 6500 supports only 2 Egress-SPAN sessions at max.
- Exercise all possible care when enabling and configuring ER/RSPAN. The traffic copied by ER/RSPAN can impose a significant load on the switch and the network. To minimise the load, configure filters to copy only the specific traffic that you want to analyse.
- Please be aware that SPAN adds load to the switch fabric and forwarding engine. So oversubscribing at any of these points cause network disruption.
- The supervisor engine handles the entire load imposed by Egress-SPAN, when the switch is in the centralised replication mode. In Catalyst 6500, 12.2(33)SXH and later releases support **distributed replication** (egress modules replicate the traffic locally). In Cisco 7600 routers15.2(2)S and later releases support distributed replication.
- Please be aware that some of the features are incompatible with SPAN Destination port. E.g., Private VLANs(PVLANs), Port Security, 802.1Q tunnelling, 802.1X, DTP, VTP etc.

Ethanalyzer and NetDR

Ethanalyzer and NetDR

Overview

- Tool to see traffic to/from CPU
- **Ethanalyzer** is implementation of TShark on **NxOS** to capture inband and management traffic
- **NetDR** (NetDriver) is software on **Catalyst 6500 / Cisco 7600 platforms** responsible for handling packets on CPU inband. Non-intrusive debug allows user to capture traffic

Ethanalyzer

Configuration

Nexus1000 | Nexus3000 | Nexus4000 | Nexus5000 | Nexus6000 | Nexus7000

Ethanalyzer

Configuration - Capture Interface

Nexus devices have multiple capture interfaces depending on platform

1. **Mgmt** – Captures traffic on the mgmt0 interface of the switch
2. **Inbound-Hi** – Captures high-priority control packets on the inband such as STP, LACP, CDP, Data Center Bridging Exchange (DCBX), Fiber Channel, and FCOE
3. **Inbound-Lo** – Captures low-priority control packets on the inband such as IGMP, TCP, UDP, IP, and ARP traffic.

- Note, Nexus7000 and Nexus4000 each have only a single **inband** interface that captures all inband traffic.

Ethanalyzer

Configuration - Filters

- There are two filtering approaches for configuring a packet capture
  http://wiki.wireshark.org/DisplayFilters
  http://wiki.wireshark.org/CaptureFilters

Display Filter Example	Capture Filter Example
“eth.addr==00:00:0c:07:ac:01”	“ether host 00:00:0c:07:ac:01”
“ip.src==10.1.1.1 && ip.dst==10.1.1.2”	“src host 10.1.1.1 and dst host 10.1.1.2”
“snmp”	"udp port 161”
“ospf”	“ip proto 89”

Ethanalyzer

Configuration - Filters

- Which filter is best to use?
- More robust filtering available via **display-filter** and any proprietary shim headers do not interfere with filter. **This is generally the best filter to use**.
- On Nexus7000, frames not matching display-filter are still captured but not displayed. The capture stops once limit-capture-frames threshold is met (default of 10 frames) which means Ethanalyzer may end before matching any frames in display-filter.
- On Nexus7000, use **capture-filter** and only frames matching the filter are captured. Alternatively, use **display-filter with high limit-capture-frames** threshold.

Ethanalyzer

Configuration - Stop Criteria

- By default, Ethanalyzer stops after capturing 10 frames. This can be changed by updating **limit-captured-frames** (0 means no limit).
- This can be used in conjunction with a **capture-ring-buffer** to create multiple files. New files can be created based on **duration** or **filesize**. The total number of files written can be control with the **files** parameter.
- **autostop** can be used to stop the capture after a certain **duration**, **filesize**, or total number of **files**.

Ethanalyzer

Putting it all together

```
NxOS# ethanalyzer local interface inbound-hi display-filter "stp" limit-captured-frames 0 capture-ring-buffer filesize 200 write bootflash:stp_ring.pcap autostop files 5
```

- Captures on **inbound-hi** interface
- Uses a **display-filter** searching for **“stp”** frames
- Sets **limit-captured-frames** to **zero** to allow continuous capturing of frames
- Uses a **capture-ring-buffer** to create a new file every **200 KB**
- **autostop** after **5** files have been created

NetDR

Overview

- Supported on Catalyst 6500 and Cisco 7600 platforms starting in 12.2(18)SXF
- **Non-Intrusive Debug** that can be used for troubleshooting high CPU
- Available on both Switch Processor (SP) and Route Processor (RP)
- Captures up to 4096 frames (wrap with **continuous** option)

Direction

From CPU's Perspective

- Receive (Rx)
- Transmit (Tx)
- Both

Filters

- Interface
- Source/Destination Index
- Ingress VLAN
- Ethertype
- Source/Destination MAC
- Source/Destination IP Address

NetDR - Example

```
Sup2T# debug netdr capture rx
Sup2T# show netdr captured-packets
A total of 111 packets have been captured
The capture buffer wrapped 0 times
Total capture capacity: 4096 packets

------- dump of incoming inband packet -------
l2idb Gi6/3, l3idb vl576, routine inband_process_rx_packet, timestamp 21:33:37.779
dbus info: src_vlan 0x240(576), src_indx 0x142(322), len 0x82(130)
  bpdu 0, index_dir 0, flood 0, dont_lrn 0, dest_index 0x
  cap1 0, cap2 0
  D0020900 02400400 01420000 82000000 11000424 260
destmac B4.14.89.61.37.80, srcmac 68.D0.9F
earl 8 shim header IS present:
  version 0, control 64(0x40), rx_vlan 576(0x240), mark_enab
  feature_index 0, group_id 0(0x0), acos 0(0x0), ttl 14,
  dti 4, dti_value 0(0x0)
ethertype 0800
protocol ip: version 0x04, hlen 0x05, tos 0x00, totlen 100, identifier 5
  df 0, mf 0, fo 0, ttl 255, src 14.2.36.1, dst 14.2.36.11
    icmp type 8, code 0
```

Ingress on port Gi6/3 on VLAN 576

Source and Destination MAC Address

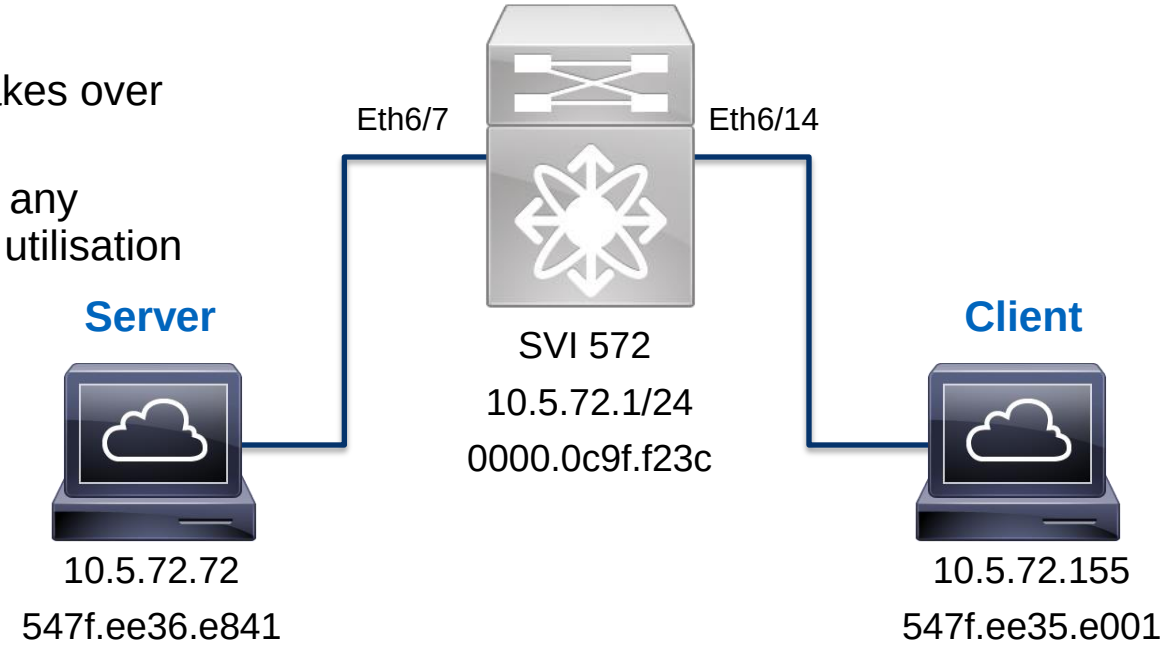
Ethertype is 0x0800 for IPv4

Source and Destination IPv4 Address

Real World Example

Slow Download Rate

- Two hosts in VLAN 572
- Download of 200MB file takes over 15 minutes to complete.
- No incrementing errors on any interface and low average utilisation

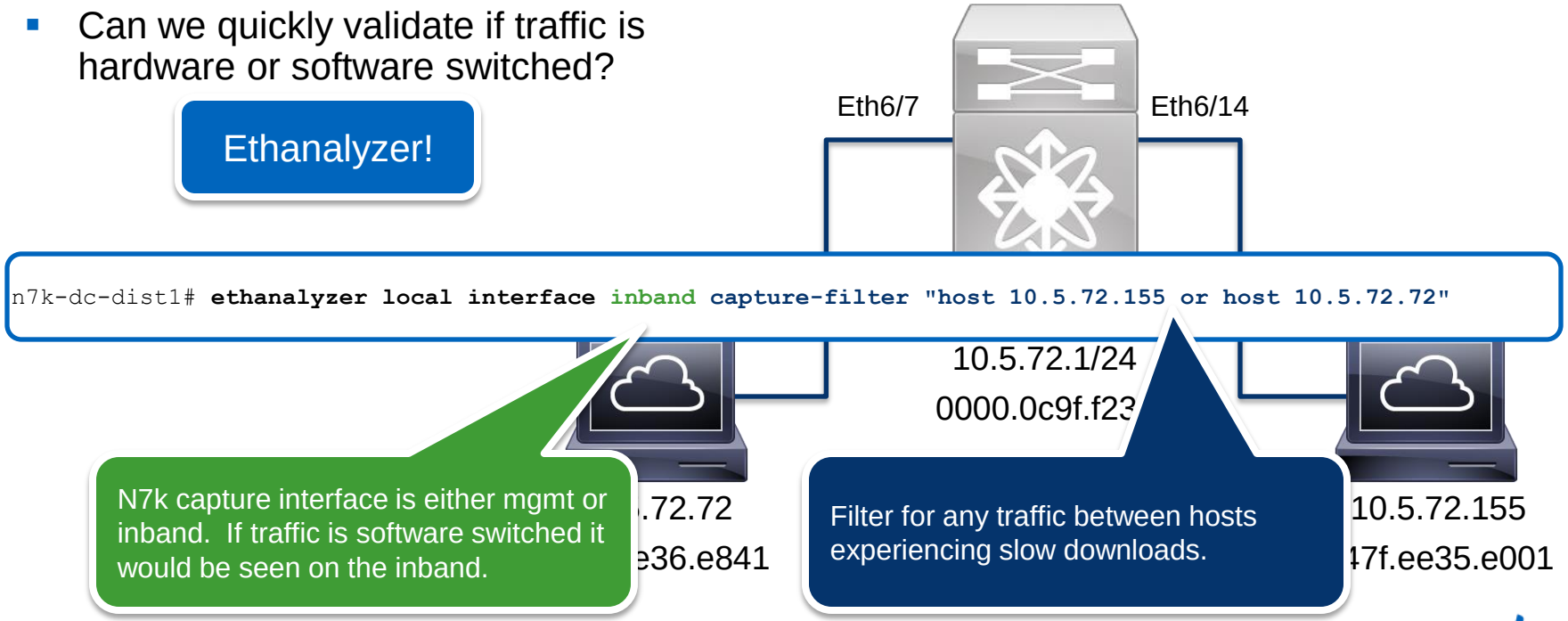


Real World Example

Slow Download Rate

- Can we quickly validate if traffic is hardware or software switched?

Ethanalyzer!



Real World Example

Slow Download Rate

- Can we quickly validate if traffic is hardware or software switched?

Ethanalyzer!

All traffic from Server (10.5.72.72) to the client (10.5.72.155) is being software switched

ICMP redirects sent back toward Server (10.5.72.72)

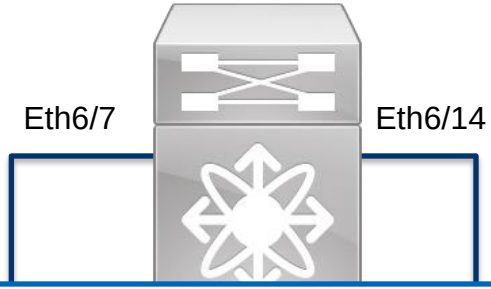
```
n7k-dc-dist1# ethanalyzer local interface inband capture-f          st 10.5.72.155 or host 10.5.72.72"
Capturing on inband
2013-03-17 13:10:24.777838  10.5.72.254 -> 10.5.72.72          ICMP Redirect
2013-03-17 13:10:24.777898   10.5.72.72 -> 10.5.72.155         SSH Encrypted
2013-03-17 13:10:24.778259   10.5.72.72 -> 10.5.72.155         SSH Encrypted
2013-03-17 13:10:24.778447  10.5.72.254 -> 10.5.72.72          ICMP Redirect
2013-03-17 13:10:24.778519   10.5.72.72 -> 10.5.72.155         SSH [TCP Out-
len=524
2013-03-17 13:10:24.778757   10.5.72.72 -> 10.5.72.155         SSH Encrypted re          nse packet len=524
2013-03-17 13:10:24.778927  10.5.72.254 -> 10.5.72.72          ICMP Redirect (Redirect for host)
etc...
```

Real World Example

Slow Download Rate

- Can we quickly validate if traffic is hardware or software switched?

Ethanalyzer!




```
n7k-dc-dist1# ethanalyzer local interface inband capture-filter "host 10.5.72.72" limit-captured-frames 1
detail | i Ethernet|Internet
Capturing on inband
1 packet captured
Ethernet II, Src: 54:7f:ee:36:e8:41 (54:7f:ee:36:e8:41), Dst: 00:00:0c:9f:f2:3c (00:00:0c:9f:f2:3c)
Internet Protocol, Src: 10.5.72.72 (10.5.72.72), Dst: 10.5.72.155 (10.5.72.155)
```

10.5.72.72

547f.ee36.e841

72.155

35.e001

These hosts are in the same VLAN yet the Server (10.5.72.72) is sending traffic destined to the gateway's MAC address

Real World Example

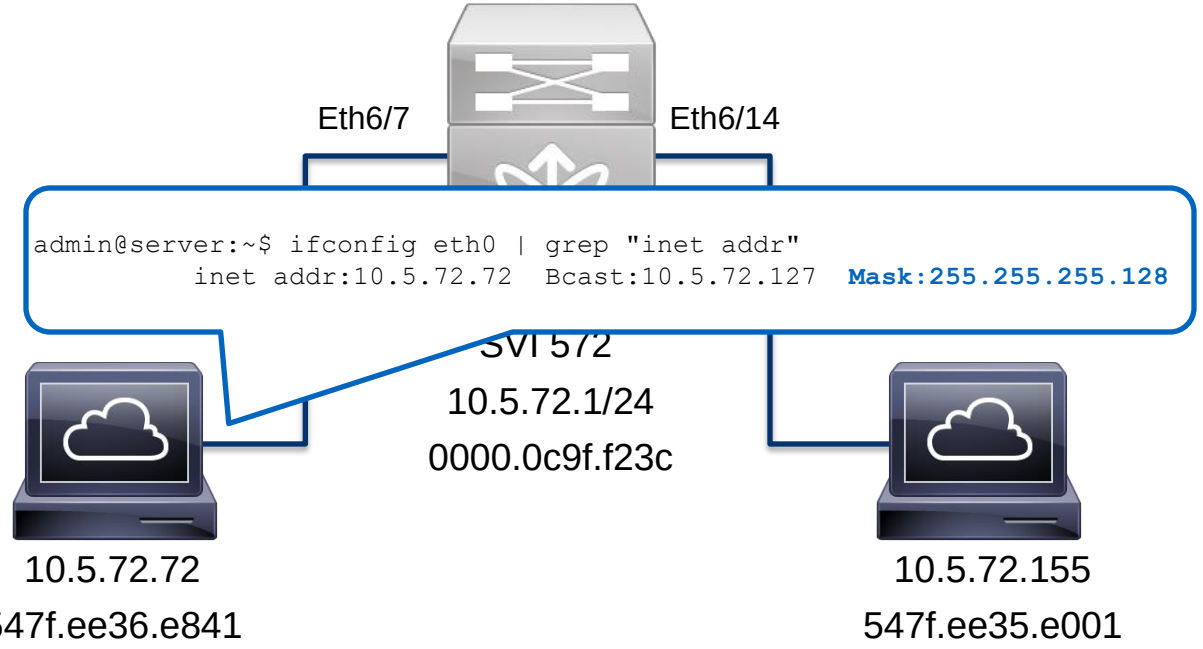
Slow Download Rate

Root cause:

- Server subnet mask incorrectly set to /25 instead of /24

Fix

1. Update server subnet mask
2. Configure “no ip redirects” on the gateway N7k



VACL and ACL Capture

VACL and ACL Capture

Overview

- Both of these features allow network administrators to replicate network traffic for monitoring purposes
- Different from traditional SPAN/RSPAN/ERSPAN, the capture feature provides the ability to selectively monitor traffic of interest via the use of an access-list
- This is extremely useful in scenarios where a subset of traffic needs to be monitored on high bandwidth links and it is not practical or possible to capture all traffic.

VACL Capture

Overview

- VLAN ACLs (VACLs) can provide access control for all packets that are bridged within a VLAN or that are routed into or out of a VLAN. VACL options include:
  - Drop
  - Forward [capture]
  - Redirect
- The **capture** action sets the capture bit for the forwarded packets so that ports with the capture function enabled can receive the packets.

VACL Capture is only supported on Catalyst 6500 / Cisco 7600.

VACL Capture

Configuration

```
ip access-list extended INTERESTING_TRAFFIC
 permit ip host 14.1.100.1 any
 permit ip any host 14.1.100.1
ip access-list extended PERMIT_ALL
 permit ip any any
!
vlan access-map VACL_CAPTURE 10
 match ip address INTERESTING_TRAFFIC
 action forward capture
vlan access-map VACL_CAPTURE 20
 match ip address PERMIT_ALL
 action forward
!
vlan filter VACL_CAPTURE vlan-list  10
!
interface GigabitEthernet1/9
 switchport
 switchport trunk encapsulation dot1q
 switchport mode trunk
 switchport capture
```

Define an ACL list that matches interesting traffic that should be sent to capture port

ACL to allow all remaining traffic

Use the '**forward capture**' keyword to forward the traffic and copy to capture port

Forward all remaining traffic

Apply VACL to vlan

Configure '**switchport capture'** on capture interface that will receive the copied frames

ACL Capture

Overview

- ACL capture allows user to selectively monitor traffic on an interface or VLAN.
- ACL capture requires a **monitor session** of type **acl-capture**
- ACE's (for VACL/RACL/PACL) can include a **capture** keyword to copy traffic matching the ACE to the monitor session.

ACL Capture is only supported on Nexus7000 M-series modules in 5.2(1) and above

ACL Capture

Configuration

```
hardware access-list capture
!
monitor session 1 type acl-capture
  destination interface Ethernet7/3
  no shut
!
interface Ethernet7/3
  switchport
  switchport mode trunk
  switchport monitor
  no shutdown
!
ip access-list ACL_CAPTURE
  10 permit ip 10.5.72.72/32 any capture session 1
  20 permit ip any any
!
interface Ethernet6/7
  ip port access-group ACL_CAPTURE in
  switchport
  switchport mode trunk
  no shutdown
```

Must be configured on Default VDC to enable ACL capture

Configure local monitor session as type acl-capture

Destination interface is configured the same as any monitor session

Configure **"capture session"** under any entry to forward capture traffic for that ACE.
Can alternatively configure **"capture session"** at the beginning of the ACL to capture all traffic

Apply as PACL, VACL, or RACL as needed
This example uses a PACL

ACL Capture

Limitations and Restrictions

- ACL Capture is only supported on Nexus7000 M-series modules in 5.2(1) and above
- Enabling ACL Capture disables ACL Logging
- Multiple ACL Capture SPAN sessions can be configured but only one ACL Capture session will be active at a time on the system across all VDCs.
- The ACL policy with capture rules can be applied in ingress direction on all interfaces
- The ACL policy with capture rules can be applied in egress direction on all L3 interfaces




ELAM

ELAM

Overview and Challenges

- Embedded Logic Analyzer Module (ELAM) is an engineering tool that is used to look inside Cisco ASICs.
- ELAM is architecture specific and therefore will have different capabilities and different CLI syntax across different forwarding engines (FE).
- Identifying the appropriate FE, creating triggers, and interpreting ELAM data for complex flows requires full architectural and forwarding knowledge

ELAM is **NOT** a supported feature. It is a diagnostic tool designed for internal use. Anything and everything about it may change from version to version without any notice

ELAM

Key Advantages and Benefits

- It is possible to use ELAM as a **capturing tool** to validate:
  1. Was the packet received
  2. On which interface/VLAN did the packet arrive
  3. What did the packet look like
  4. How was the packet altered and where was it sent
- It is **not intrusive**
- It can be used at a very granular level to **troubleshoot a single traffic flow** which can be an **invaluable tool** to network administrators.
- In this section we will review ELAM on Catalyst 6500 / Cisco 7600 and Nexus7000

The purpose of this section is to give you enough information to perform basic capturing tasks. It is not meant to be a deep dive into all capabilities of ELAM.

ELAM

Basics to know before performing an ELAM

- Data Bus (DBUS) and Result Bus (RBUS)

  The **DBUS** contains several platform specific internal fields along with the header information from a frame required to make the forwarding decision. We use the DBUS information to validate where the frame was received and basic data about the frame.

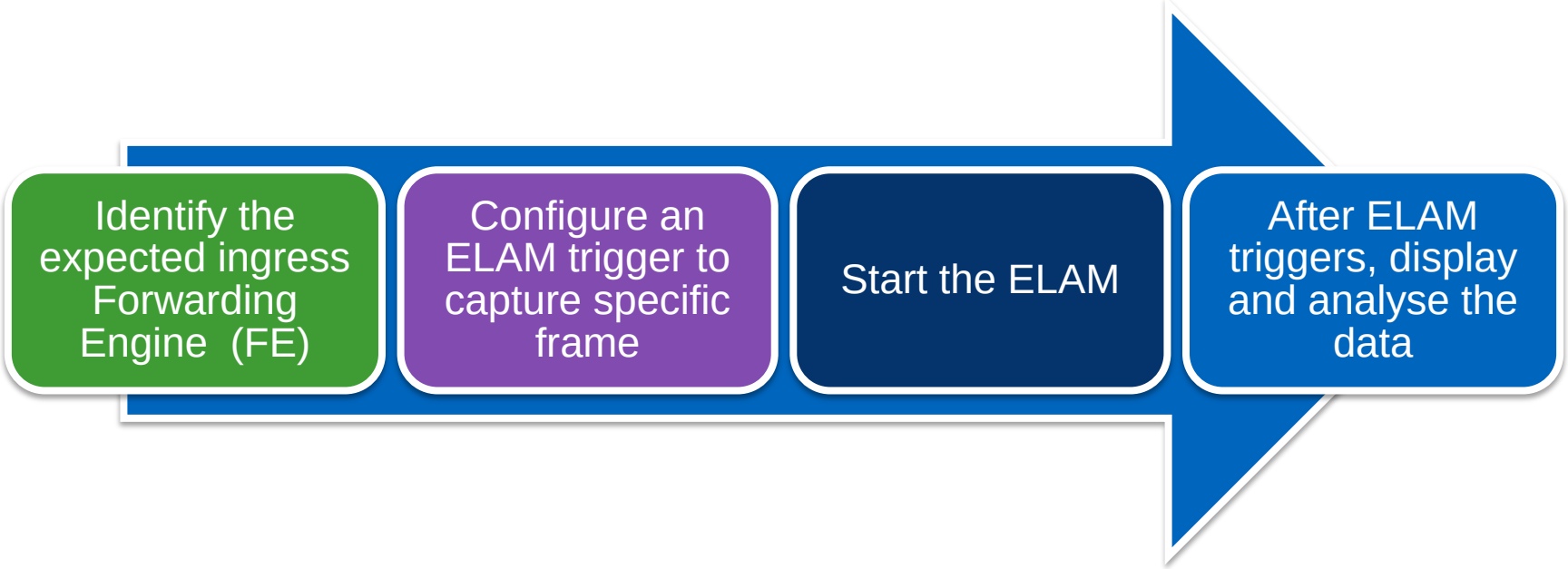
  The **RBUS** will contain information about the forwarding decision to help determine if the frame was altered and where it was sent.

- Local Target Logic (LTL)

  The **LTL** is an index used to represent a port or group of ports. The source LTL index and the destination LTL index tell us which port the frame was received and where it was sent.

ELAM

Workflow



ELAM

Catalyst 6500 / Cisco 7600 Overview

- Centralised Forwarding by the supervisor
- Distributed Forwarding on DFC enabled line cards
- When performing an ELAM, you want to ensure that you are capturing on the ingress forwarding engine. For traffic that ingresses a CFC-enabled line card, or a classical line card, the PFC on active supervisor in the chassis will be the ingress forwarding engine.
- For traffic that ingresses a DFC enabled line card, the local DFC will be the ingress forwarding engine.

ELAM

Catalyst 6500 / Cisco 7600 Basic Syntax PFC4/DFC4

```
Sup2T(config)#service internal
Sup2T# show platform capture elam asic eureka slot 5
Assigned asic_desc=eu50
Sup2T# show platform capture elam trigger master eu50 dbus dbi ingress <trigger>
Sup2T# show platform capture elam start
cap_commands: Default ELAM RBI PB1 added to list
Sup2T# show platform capture elam status
ID#    Role  ASIC     Slot  Inst  Ver  ELAM        Status
-----  ----  -------  ----  ----  ---  ---------   ------
eu50   M     EUREKA   5     0     1.3  DBI_ING     Capture Completed
eu50   s     EUREKA   5     0     1.3  RBI_PB1     Capture Completed

ID#    ELAM        Trigger
-----  ---------   ----------
eu50   DBI_ING     <trigger displayed here>
eu50   RBI_PB1     TRIG=1
Sup2T# show platform capture elam data
DBUS data:
(output omitted)
RBUS data:
(output omitted)
```

Sup2T/PFC4/DFC4 **eureka** ASIC

RBUS Trigger automatically set. It can manually be set via:
`show platform capture elam trigger slave eu50 rbus rbi pb1`

Use the following command to map source index/destination index to physical ports on PFC4/DFC4:

show platform hardware ltl index <index>

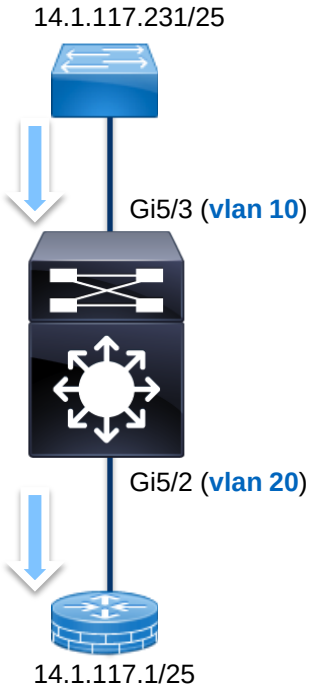
ELAM

IPv4 Example on Sup2T (PFC4)

```
Sup2T#show mod 5
Mod Ports Card Type                                   Model              Serial No.
--- ----- ------------------------------------------ ------------------ -----------
  5     5  Supervisor Engine 2T 10GE w/ CTS (Acti VS-SUP2T-10G          SAL15056BKR
```

- Traffic ingresses on module **5** which is the active supervisor and will therefore be the ingress FE
- The traffic flow is from host **14.1.117.231** toward host **14.1.117.1** so the trigger will be:

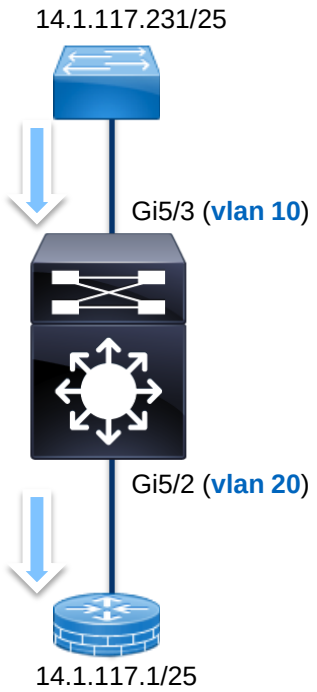
IPv4 if IP_SA=14.1.117.231 IP_DA=14.1.117.1



ELAM

IPv4 Example on Sup2T (PFC4)

```
Sup2T(config)#service internal
Sup2T# show platform capture elam asic eureka slot 5
Assigned asic_desc=eu50
Sup2T# show platform capture elam trigger master eu50 dbus dbi ingress ipv4 if
ip_sa=14.1.117.231 ip_da=14.1.117.1

Sup2T# show platform capture elam start
cap_commands: Default ELAM RBI PB1 added to list
Sup2T# show platform capture elam status
ID#    Role  ASIC     Slot  Inst  Ver  ELAM       Status
-----  ----  -------  ----  ----  ---  ---------  ------
eu50   M     EUREKA   5     0     1.3  DBI_ING    Capture Completed
eu50   s     EUREKA   5     0     1.3  RBI_PB1    Capture Completed

ID#    ELAM       Trigger
-----  ---------  ----------
eu50   DBI_ING    FORMAT=IP L3_PROTOCOL=IPV4 IP_SA=14.1.117.231 IP_DA=14.1.117.1
eu50   RBI_PB1    TRIG=1
```



ELAM

IPv4 Example on Sup2T (PFC4)

```
Sup2T#show platform capture elam data
(some output omitted)
DBUS data:
VLAN ............................ [12] = 10
SRC_INDEX ....................... [19] = 0x102
L3_PROTOCOL ..................... [4] = 0 [IPV4]
L3_PT ........................... [8] = 1 [ICMP]
IP_TTL .......................... [8] = 255
IP_SA ........................... = 14.1.117.231
IP_DA ........................... = 14.1.117.1

RBUS data:
FLOOD ........................... [1] = 0
DEST_INDEX ...................... [19] = 0x101
VLAN ............................ [12] = 20
IP_TTL .......................... [8] = 254
REWRITE_INFO
 i0  - replace bytes from ofs 0 to ofs 11 with seq '00 00 0C 07 AC CA B4 14 89 61 37 80'.
```

SRC_INDEX is ingress port (0x102) = Gi5/3

```
Sup2T#show platform hardware ltl index 0x102
LTL index 0x102 contain ports :
=======================================
Gi5/3
```

DEST_INDEX is egress port (0x101) = Gi5/2

```
Sup2T#show platform hardware ltl index 0x101
LTL index 0x101 contain ports :
=======================================
Gi5/2
```

Packet received on VLAN 10 with a TTL of 255 and routed out VLAN 20 with a TTL of 254

Rewrite information on the packet contains destination MAC (0000.0c07.acca) and source MAC (b414.8961.3780)

2 (**vlan 20**)

14.1.117.1/25

ELAM

Nexus7000 Overview

- Fully Distributed Forwarding on all line cards
- Most line cards contain multiple Forwarding Engines
- Different ASICs and therefore different types of Forwarding Engines between modules each have a unique set of capabilities and CLI
- Need to validate which forwarding engine instance on a particular module maps to the front-panel port for the ingress traffic

ELAM

Nexus7000 Overview

M1/M2	F1	F2
Eureka	Orion	Clipper

- Each of the above ASICs has the **L2LKP/L2LU** role. The idea is to find the instance number of the L2LKP/L2LU ASIC for the ingress port so the ELAM can be performed on the correct ASIC. Attach to the module and issue **show hardware internal dev-port-map** to perform this task.
- To map source LTL index and destination LTL index to a port(s), use **show system internal pixm info ltl <index>**

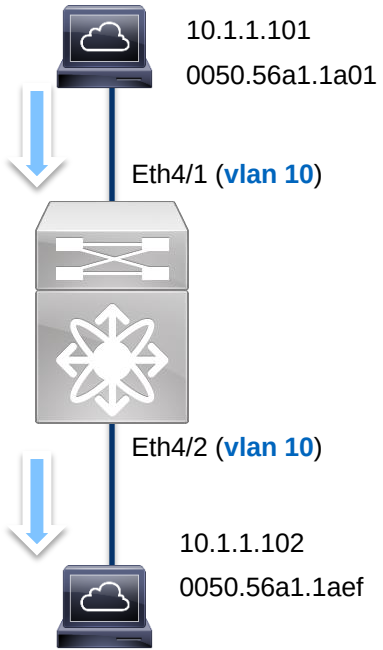
ELAM

Nexus7000 – M1 Example

```
N7k# attach module 4
Attaching to module 4 ...
To exit type 'exit', to abort type '$.'
module-4# show hardware internal dev-port-map
--------------------------------------------------------------
CARD_TYPE:         48 port 1G
>Front Panel ports:48
--------------------------------------------------------------
 Device name              Dev role              Abbr num_ins
--------------------------------------------------------------
> Eureka                 DEV_LAYER_2_LOOKUP     L2LKP  1
+-------------------------------------------------------------+
+-----------+++FRONT PANEL PORT TO ASIC INSTANCE MAP+++--------+
+-------------------------------------------------------------+
FP port|PHYS |SECUR |MAC_0 |RWR_0 |L2LKP |L3LKP |QUEUE |SWICHF
   1      0      0      0      0      0      0      0      0
   2      0      0      0      0      0      0      0      0
```

There is only one Eureka ASIC for the entire card, therefore, specifying the instance is not necessary

L2LKP for M1 is Eureka ASIC. Ingress port (Eth4/1) is on instance 0



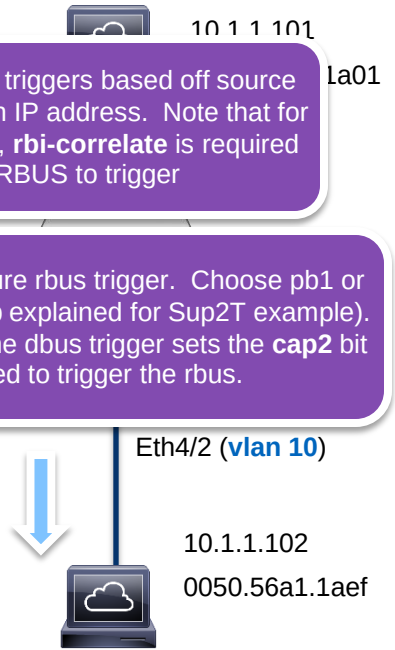
ELAM

Nexus7000 – M1 Example

```
module-4# elam asic eureka
module-4(eureka-elam)# trigger dbus dbi ingress ipv4 if source-ipv4-address
10.1.1.101 destination-ipv4-address 10.1.1.102 rbi-corelate
module-4(eureka-elam)# trigger rbus rbi pb1 ip if cap2 1
module-4(eureka-elam)# start
module-4(eureka-elam)# status
Instance: 1

EU-DBUS: Triggered
trigger dbus dbi ingress ipv4 if source-ipv4-address 10.1.1.101 destina
address 10.1.1.102 rbi-corelate

EU-RBUS: Triggered
trigger rbus rbi pb1 ip if cap2 1
```

This example triggers based off source and destination IP address. Note that for M1/M2 ELAM, **rbi-correlate** is required for RBUS to trigger

Must manually configure rbus trigger. Choose pb1 or pb2 (use rule-of-thumb explained for Sup2T example). The **rbi-correlate** in the dbus trigger sets the **cap2** bit which is used to trigger the rbus.



ELAM

Nexus7000 – M1 Example

```
(some output omitted)
module-4(eureka-elam)# show dbus
seq = 0x17
vlan = 10
source_index = 0x00a21
l3_protocol = 0x0    (0:IPv4, 6:IPv6)
l3_protocol_type = 0x06, (1:ICMP, 2:IGMP, 4:IP, 6:TCP,
dmac = 00.50.56.a1.1a.ef
smac = 00.50.56.a1.1a.01
ip_ttl = 0x40
ip_source = 010.001.001.101
ip_destination = 010.001.001.102
tcp source port = 0x0050
tcp dest port = 0x2309
tcp sequence no = 0x00af5322
tcp acknoledgement no = 0x0e8aff20

module-4(eureka-elam)# show rbus
seq = 0x17
flood = 0x1
dest_index = 0x00048
vlan = 10
ttl = 0x40
```

Ensure that sequence number in dbus and rbus match

10.1.1.101

0050.56a1.1a01

```
N7k# show system internal pixm info ltl 0xa21
Type              LTL
--------------------------------
PHY_PORT        Eth4/1
```

The **flood bit** is set in the rbus result which means this frame is flooded on vlan 10. The flood-bit is the most significant bit in the destination index. Therefore destination index changes from 0x0048 to 0x8048:

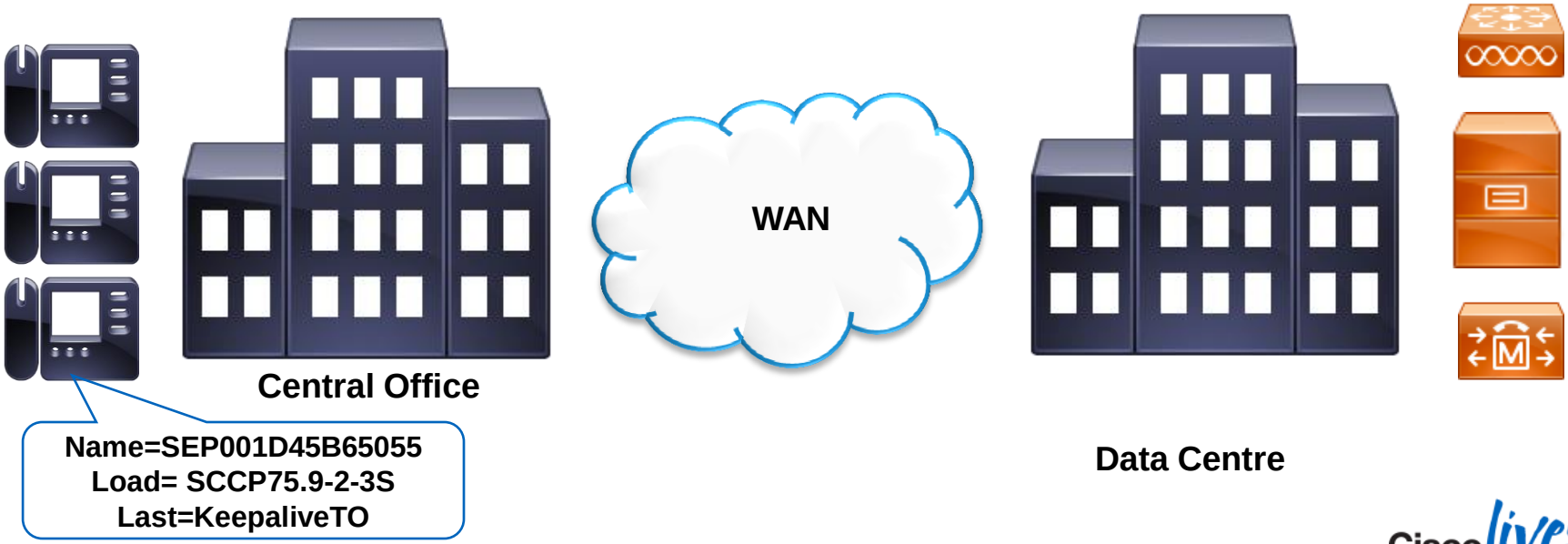
```
N7k# show system internal pixm info ltl 0x8048
IFIDX             LTL
--------------------------------
Eth4/1              0x0a21
Eth4/2              0x0a20
```




Putting It All Together

Real World Example

VoIP phone reset



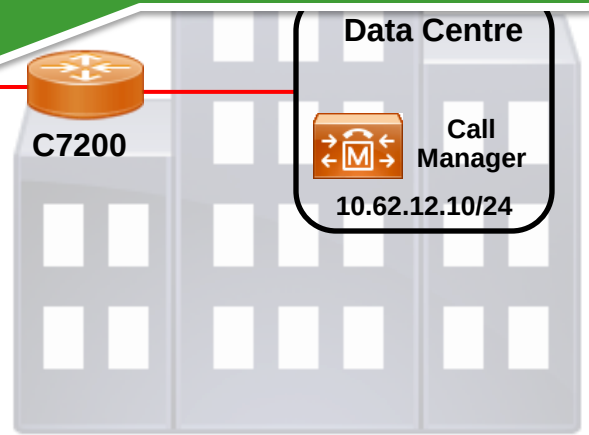
Real World Example

VoIP phone reset

Assess Phase

```
! ACL to filter ICMP traffic between hosts
ip access-list extended icmp-acl
 permit icmp host 10.0.101.22 host 10.62.12.10
 permit icmp host 10.62.12.10 host 10.0.101.22

! EPC on ASR
monitor capture icmpCap interface <WAN> both access-list icmp-acl
monitor capture icmpCap start

! EPC on 7200
monitor capture buffer icmpBuf
monitor capture buffer icmpBuf filter access-list icmp-acl
monitor capture point ip cef icmpCap <WAN> both
monitor capture point associate icmpCap icmpBuf
monitor capture point start icmpCap
```

In addition to VoIP issues, we've confirmed that we also see ping loss between sites. Next step is to configure EPC on each WAN router to isolate which site is dropping packets.



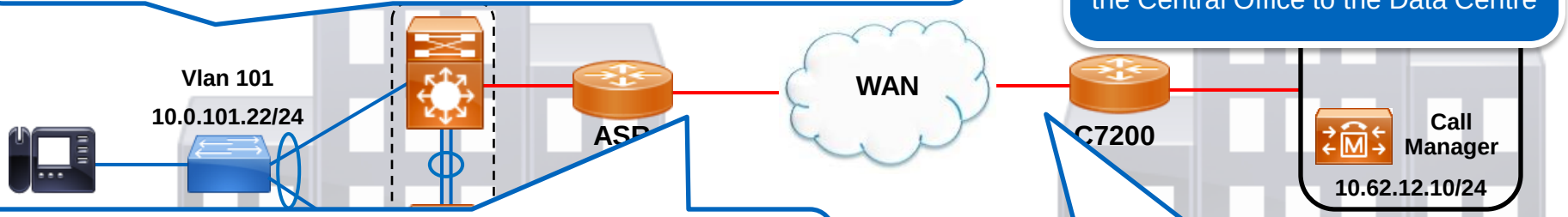
Real World Example

VoIP phone reset

Assess Phase

```
C3560#ping 10.62.12.10 repeat 25
Sending 25, 100-byte ICMP Echos to 10.62.12.10, timeout is 2 seconds:
!!!.!!!!!!!.!!!.!!..!.!..
Success rate is 68 percent (17/25), round-trip min/avg/max = 1/1/8 ms
```

The difference in the packet count for this flow confirms there is traffic loss through the WAN in the direction from the Central Office to the Data Centre




```
ASR#show monitor capture icmpCap buffer | i packets in buf
 packets in buf    : 42

ASR#monitor capture icmpCap export bootflash:asr_icmpCap.pcap
```

```
C7200#show monitor capture buffer icmpBuf parameters | i Packets
Buffer Size : 1048576 bytes, Max Element Size : 68 bytes, Packets : 34

C7200#monitor capture buffer icmpBuf export bootflash:7200_icmpBuf.pcap
```

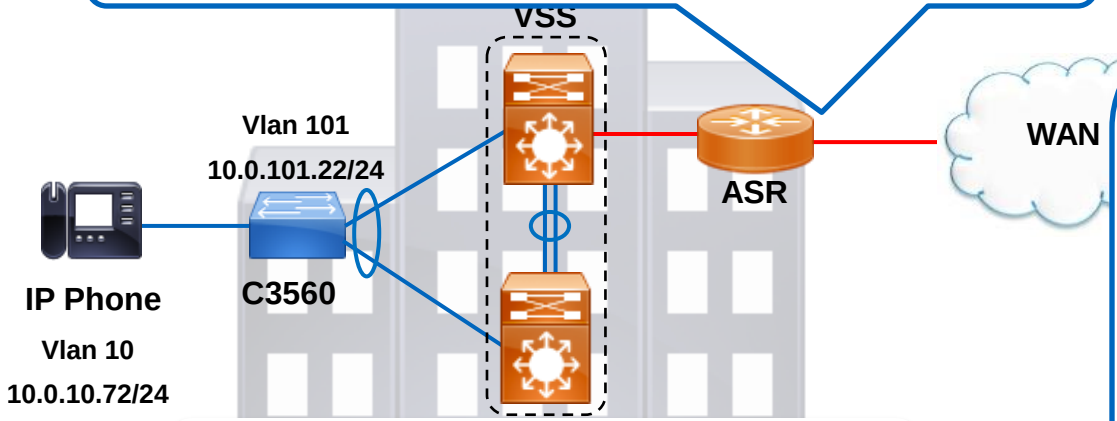
Real World Example

VoIP phone reset

Assess Phase

```
ASR#show interfaces <WAN> | i output rate
  30 second output rate 99404000 bits/sec, 8983 packets/sec
```

ISP Committed Information Rate (CIR) is only 100 Mbps . Traffic at the Central Office is exceeding that rate!

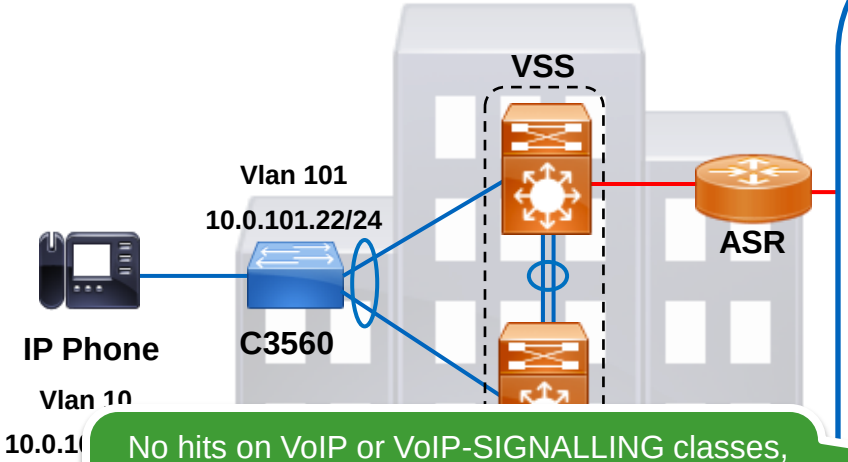


Let's configure FnF to see what is utilising all the bandwidth on the WAN uplink at the Central Office.

```
flow record RECORD
 match ipv4 source address
 match ipv4 destination address
 match transport source-port
 match transport destination-port
 match flow direction
 collect interface input
 collect counter packets
!
flow exporter EXPORT
 destination <collector> vrf netflow-mgmt
!
flow monitor MONITOR
 exporter EXPORT
 record RECORD
!
interface <WAN>
 ip flow monitor MONITOR output
```

Real World Example

VoIP phone reset



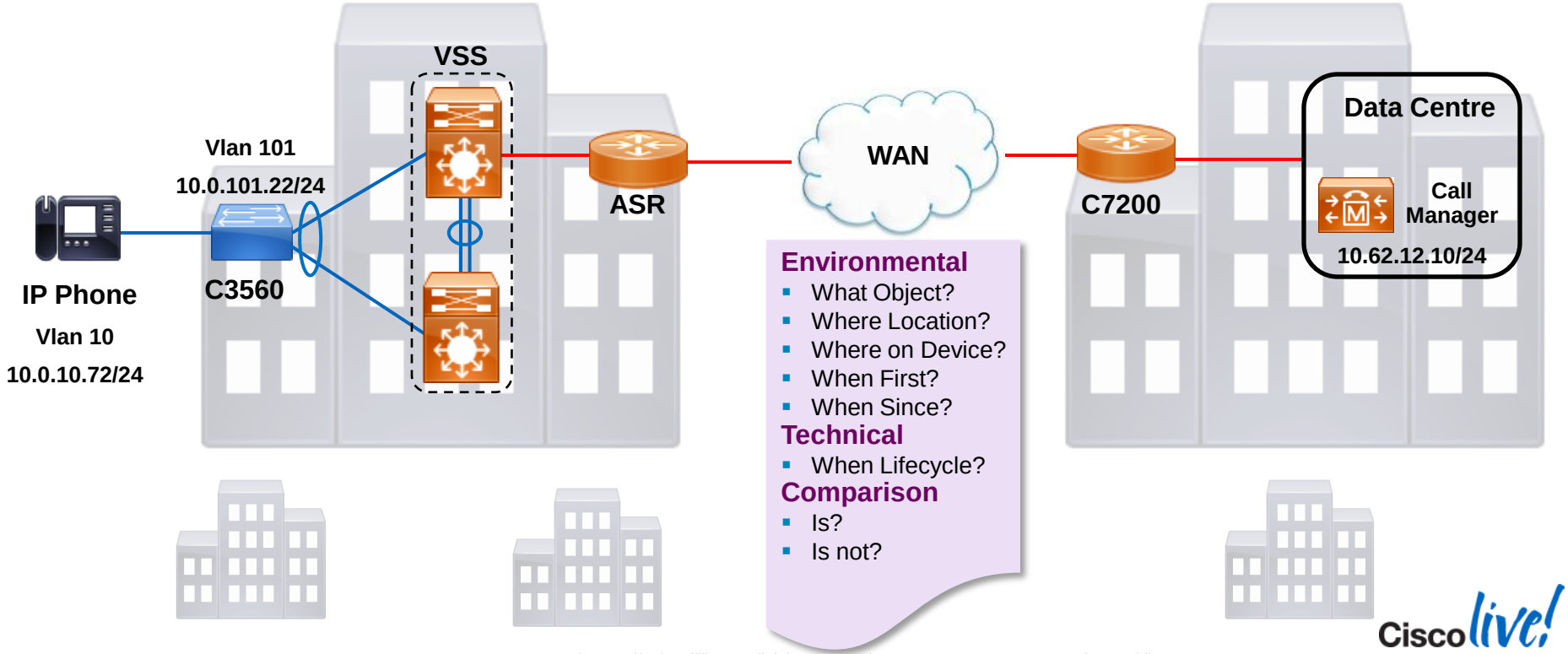
No hits on VoIP or VoIP-SIGNALLING classes, therefore VoIP traffic is not prioritised across the WAN. This will impact voice quality and possibly cause phones to reset.

```
ASR#show policy-map interface <WAN>

Service-policy output: SHAPE_100
    Class-map: class-default (match-any)
      113588941 packets, 156012353154 bytes
      (pkts output/bytes output) 113587316/156012107457
      shape (average) cir 100000000, bc 400000, be 400000

      Service-policy : VOIP
        Class-map: VOIP-CALL (match-all)
          0 packets, 0 bytes
          Match:  dscp ef (46)
          Priority: 25% (25000 kbps)

        Class-map: VOIP-SIGNALING (match-all)
          0 packets, 0 bytes
          Match:  dscp cs3 (24) af31 (26)
          bandwidth remaining 10%

        Class-map: class-default (match-any)
          113588922 packets, 156012327106 bytes
```

Real World Example

VoIP phone reset

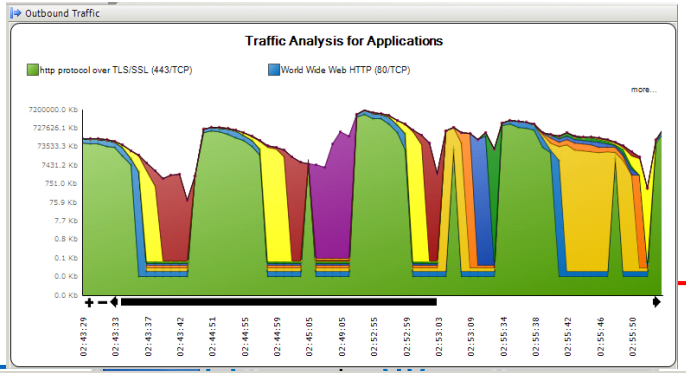
Acquire Phase



Real World Example

VoIP phone reset

Analyse Phase



From the Netflow collector analysis we found that most of the traffic flows are Web traffic combined with business critical applications, from multiple hosts.

Most Likely Cause / Test Possible Causes / Possible Causes

So rather than policing any particular flow, let's create a QoS policy to prioritise the VoIP Call and Signalling traffic.

C7200

Call Manager

2.10/24

IP P

Vla

10.0.1

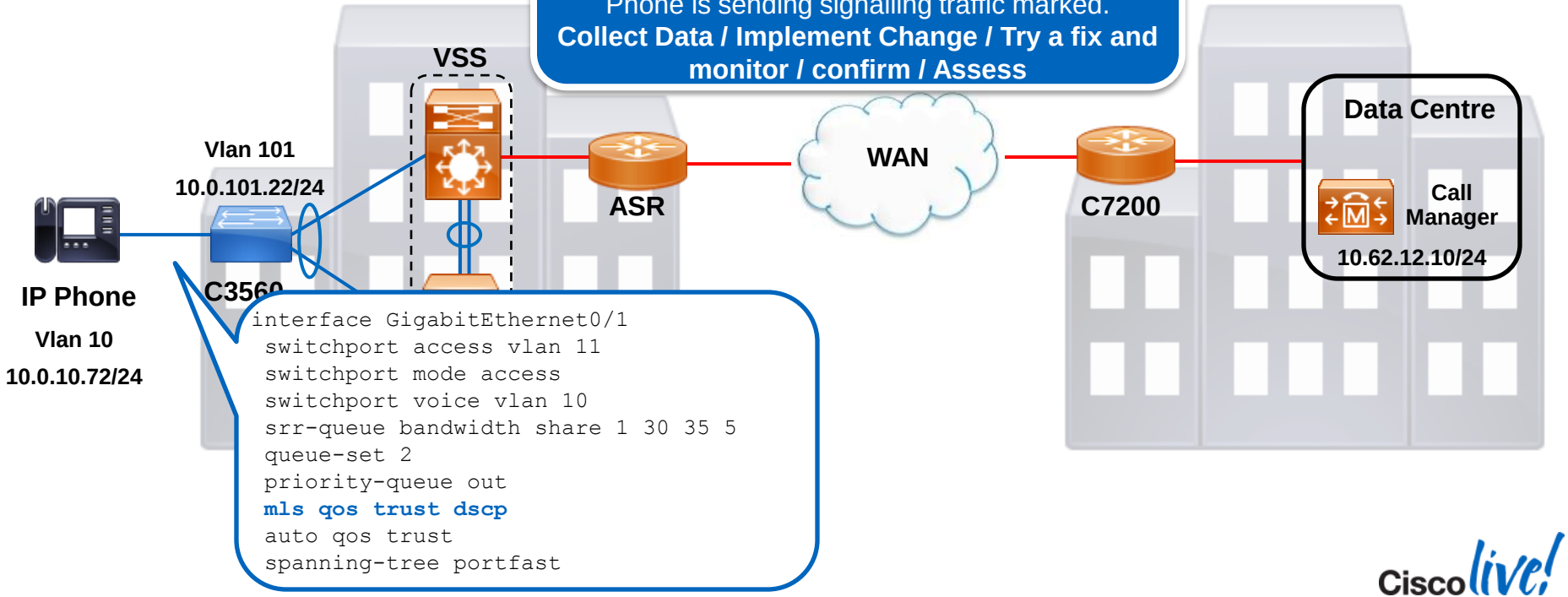
Application	Protocol	Total Traffic	Total Packets	Traffic Percentage
http protocol over TLS/SSL (443)	TCP	41350174.	112364604	36%
World Wide Web HTTP (80)	TCP	30113087.	81829042	26%
Internet Message Access Protocol (143)	TCP	8606632.0	23387587	8%
NETBIOS Name Service (137)	TCP	7950104.9	21603546	7%
Lightweight Directory Access Protocol (389)	TCP	7710322.4	20951963	7%
Network News Transfer Protocol (119)	TCP	7453306.8	20253551	7%
SSH Remote Login Protocol (22)	TCP	5277258.4	14340376	5%
MySQL (3306)	TCP	3739078.4	10160539	3%

Real World Example

VoIP phone reset

Act Phase

We need to determine where the packets are getting remarked. First step is to validate the configuration on the access port. Next is to validate if the IP Phone is sending signalling traffic marked.
Collect Data / Implement Change / Try a fix and monitor / confirm / Assess



Real World Example

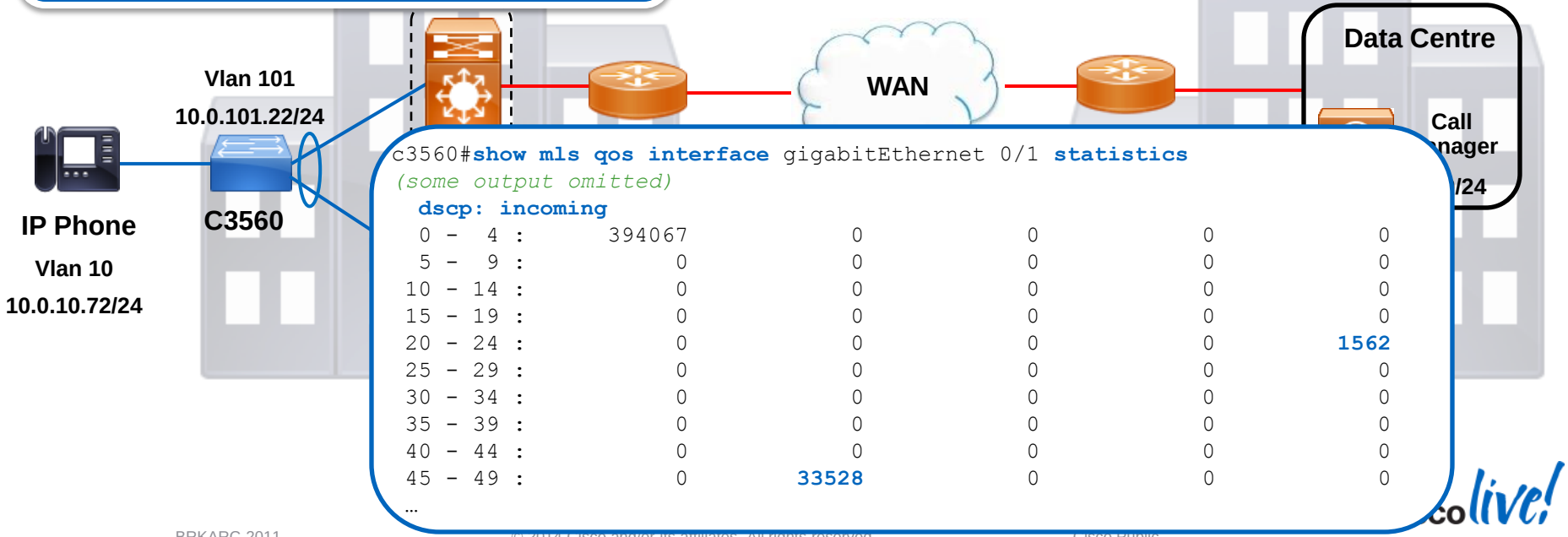
VoIP phone reset

Act Phase

We have two options:

1. SPAN/RSPAN on C3560 or IP Phone to validate marking of VoIP traffic
2. CLI on C3560 to check received QoS statistics

Easier!




```
c3560#show mls qos interface gigabitEthernet 0/1 statistics
(some output omitted)
  dscp: incoming
  0 -  4 :      394067          0          0          0          0
  5 -  9 :           0          0          0          0          0
 10 - 14 :           0          0          0          0          0
 15 - 19 :           0          0          0          0          0
 20 - 24 :           0          0          0          0       1562
 25 - 29 :           0          0          0          0          0
 30 - 34 :           0          0          0          0          0
 35 - 39 :           0          0          0          0          0
 40 - 44 :           0          0          0          0          0
 45 - 49 :           0      33528          0          0          0
...
```

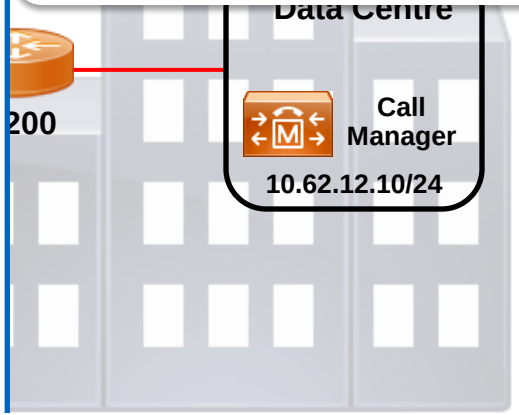
Real World Example

VoIP phone reset

Act Phase

```
VSS# show plat cap elam asic superman slot 5
VSS# show plat cap elam trigger dbus ipv4 if ip_sa=10.0.10.72 ip_da=10.62.12.10
VSS# show plat cap elam start
VSS# show plat cap elam status
Active ELAM info:
Slot Cpu   Asic    Inst Ver PB Elam
---- --- -------- ---- --- -- ----
5    0   ST_SMAN  0    3.2    Y
DBUS trigger: FORMAT=IP L3_PROTOCOL=IPV4 IP_SA=10.0.10.72 IP_DA=10.62.12.10
Elam capture completed

VSS# show plat cap elam data
DBUS data:
IP_TOS                              [8] = 0x60
IP_SA                               = 10.0.10.72
IP_DA                               = 10.62.12.10

RBUS data:
IP_TOS                              [8] = 0x0
```

Let's use ELAM to see what the VSS is doing with the Frame.



VSS is remarking the ToS field from 0x60 (cs3) to 0x0 (default)

Real World Example

VoIP phone reset

Act Phase

Port-channel QoS state is untrusted.
Let's set it to **trust dscp.**

```
VSS# show run interface port-channel <pc>
!
interface Port-channel13
 switchport
 switchport trunk encapsulation dot1q
 switchport mode trunk

VSS#conf t
VSS(config)#interface port-channel <pc>
VSS(config-if)# mls qos trust dscp
```

```
ASR#show policy-map interface <WAN>

Service-policy output: SHAPE_100
    Class-map: class-default (match-any)
      113588941 packets, 156012353154 bytes
      (pkts output/bytes output) 113587316/156012107457
      shape (average) cir 100000000, bc 400000, be 400000

      Service-policy : VOIP
        Class-map: VOIP-CALL (match-all)
          5821 packets, 0 bytes
          Match:  dscp ef (46)
          Priority: 25% (25000 kbps)

        Class-map: VOIP-SIGNALING (match-all)
          392 packets, 0 bytes
          Match:  dscp [illegible] (24) af31 (26)
          bandwidth remaini[illegible]0%

        Class-map: class-[illegible]
          113588922 packet[illegible]
```

VOIP Traffic is now prioritised!

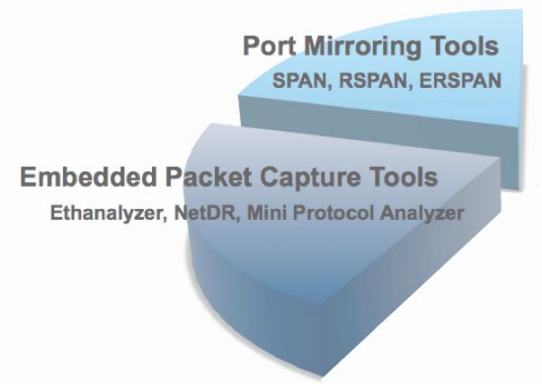



Summary and Take Away

Overview of Troubleshooting Tools

Summary and Take Away ….

- Capture Control-Plane traffic:
  - EthAnalyzer, NetDr and Flexible NetFlow (FnF)
- Copy Data-Plane traffic to external device:
  - SPAN/RSPAN/ERSPAN, VACL and ACL capture
- Copy Data-Plane traffic to internal buffer:
  - FnF, Embedded Packet Capture (EPC), Mini Protocol Analyzer (MPA), and Wireshark
- Capture Data-Plane Frame and Forwarding Headers:
  - ELAM



Overview of Troubleshooting Tools

Summary and Take Away ….

- Cisco Routers and Switches are advanced and feature-rich, built with keeping end-users in mind and also network engineers.
- Cisco provides a rich set of troubleshooting and capturing tools embedded and supported across the spectrum of its products. These tools give visibility into the products helping to validate the path-of-the-packet and isolate problems.
- Knowing the tools and capabilities available on each platform will reduce the time to resolution of network issues.

We have a tool for you!




Q & A

Complete Your Online Session Evaluation

Give us your feedback and receive a Cisco Live 2014 Polo Shirt!

Complete your Overall Event Survey and 5 Session Evaluations.

- Directly from your mobile device on the Cisco Live Mobile App
- By visiting the Cisco Live Mobile Site www.ciscoliveaustralia.com/mobile
- Visit any Cisco Live Internet Station located throughout the venue

Polo Shirts can be collected in the World of Solutions on Friday 21 March 12:00pm - 2:00pm

Learn online with Cisco Live!

Visit us online after the conference for full access to session videos and presentations.
www.CiscoLiveAPAC.com


CISCO
TM