
cisco Live!
Let's go
#CiscoLive


2
HOLLY SPRINGS FIRE-RESCUE
HOLLY SPRINGS
HOLLY SPRINGS
2
SAFE PLACE

The bridge to possible

Overview of Packet Capturing Tools in Cisco Switches and Routers

Nick Oliver, Technical Leader
@RTPCCIE
Oscar Silva, Technical Leader
@CiscoPerson

BRKTRS-2811

Cisco Webex App

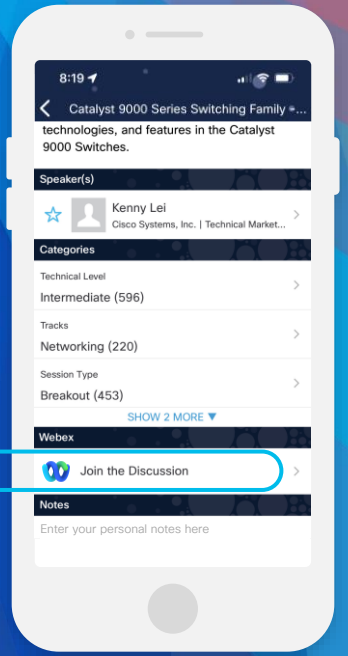
Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until June 9, 2023.



https://ciscolive.ciscoevents.com/ciscolivebot/#BRKTRS-2811

Additional Breakout Sessions and Labs

For more info...

LABTRS-2456 Packet Capturing Tools in Routing Environments
LABTRS-2391 Packet Capturing Tools in Enterprise Switching Environments
LABTRS-2048 Packet Trace and Conditional Debugger on IOS-XE Routers

LABCRT-2452 CCNP ENCOR – Core Enterprise Network Technologies Practice Lab
LABCRT-2460 CCNP ENARSI – Implementing Cisco Enterprise Advanced Routing and Services Practice Lab
LABCRT-2464 Troubleshoot like a CCNP Practice Lab

BRKTRS-3475 Advanced Troubleshooting of the cat8k, asr1k, ISR and SDWAN Edge Made Easy
BRKXAR-2003 Extending Enterprise Network into Public Cloud with Cisco Catalyst 8000V Edge Software
BRKTRS-3090 Troubleshooting Cisco Catalyst 9000 Series Switches

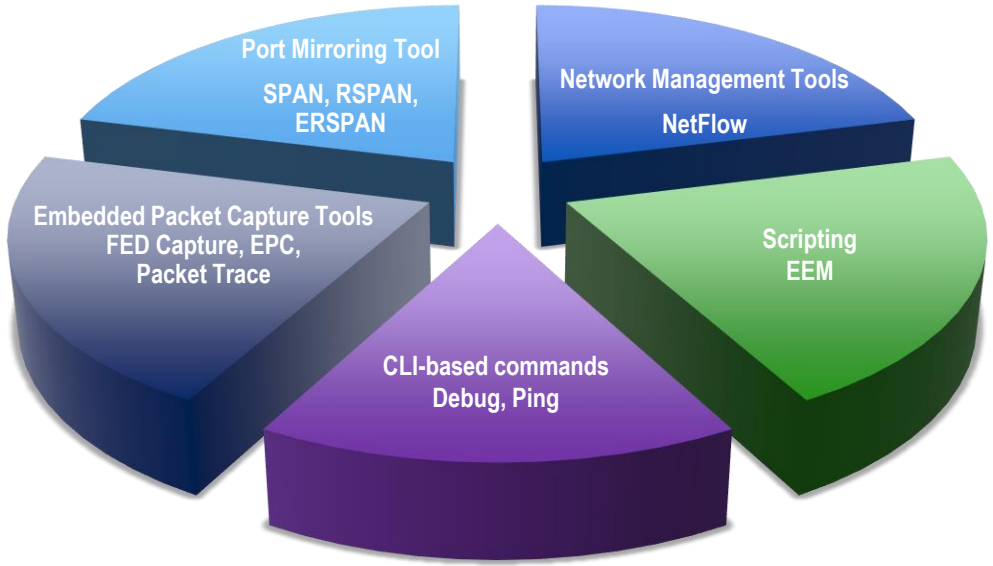
Agenda

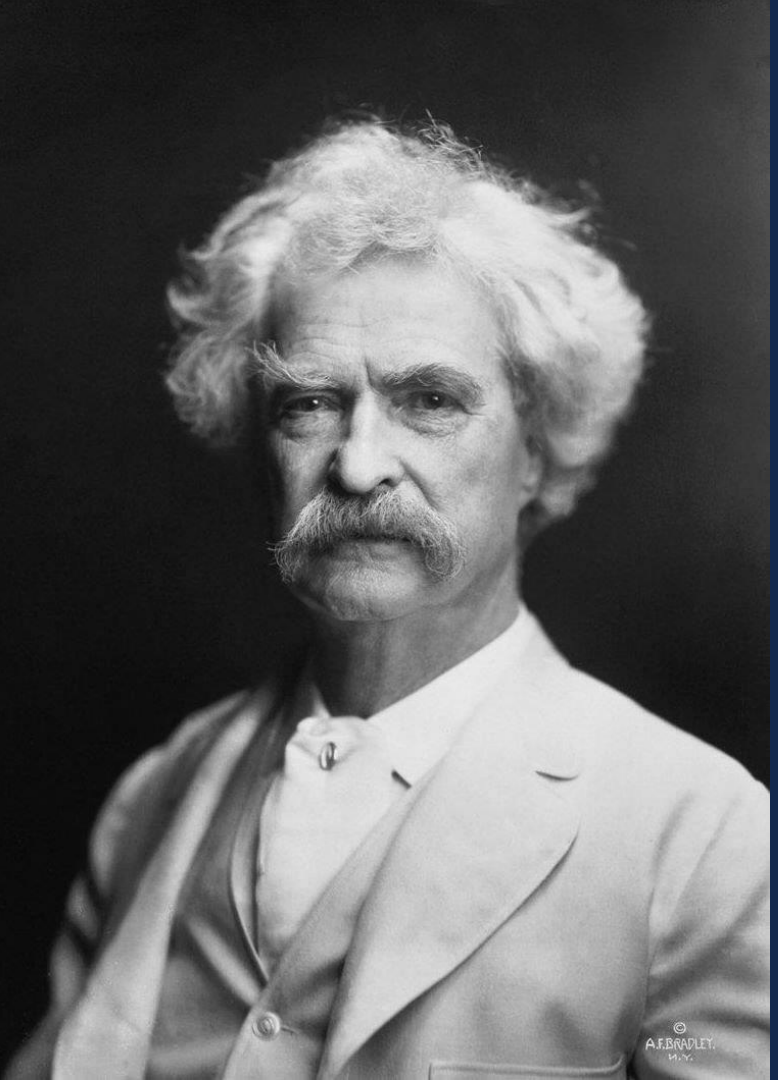
- Introduction
- Port Mirror Tool
- Embedded Packet Capture
- FED Capture
- Packet Trace
- Event Triggered Captures
- Putting it All Together

Introduction

Goal of this Session…

- Create an **awareness of the packet capturing** tools that are available
- Learn how to use the tools through **real-world examples**
- Help you understand the **capabilities and features** of each tool



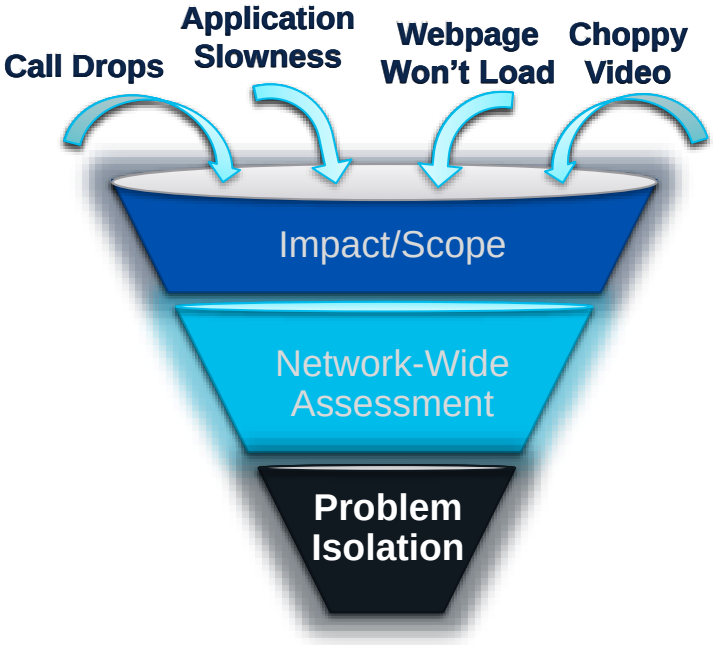


“What gets us into trouble is not what we don't know. It's what we know for sure that just ain't so.”

Mark Twain

A Troubleshooting Methodology

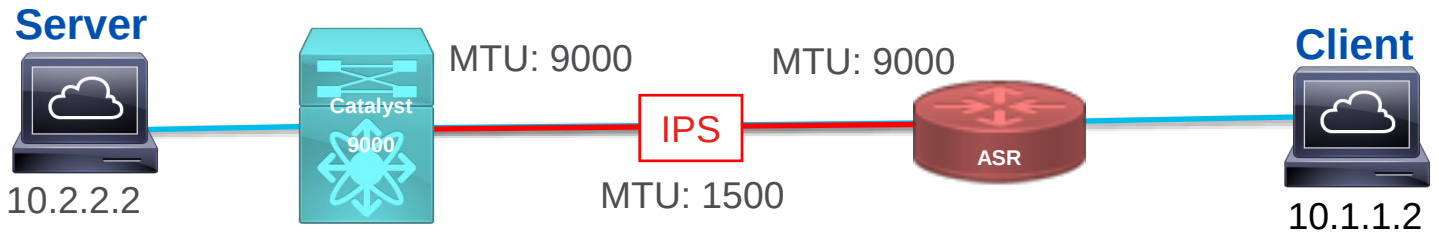
- Define the problem, impact, and scope based on facts, user reports, and considering recent changes
  - E.g., Application slowness after recent migration, When does the issue occur? Are all users impacted?
- Assess what you know from network monitoring
  - E.g., SNMP, Syslogs, NetFlow Data, Real-time performance monitoring, IP SLA. What is outside normal bounds?
- Select the right tool to isolate the problem
  - E.g., Are you on a router or switch? What type of data do you need (flow visibility, pcap, forwarding data, etc)



Having these details helps us make progress towards a resolution.

Data Transfers Are Broken

A Problem with MTU?



Sometimes a cable is more than just a cable.

Acronyms / Definitions

➔ Reference Slide

Acronyms	Definitions	Acronyms	Definitions
FNF	Flexible NetFlow	SP	Switch Processor
EPC	Embedded Packet Capture	RP	Route Processor
PSV	Packet State Vector	ASIC	Application Specific Integrated Circuit
SPF	Show Platform Forward	ELAM	Embedded Logic Analyzer Module
SPAN	Switch Port Analyzer	CoPP	Control Plane Policing
RSPAN	Remote SPAN	ACL	Access Control List
ERSPAN	Encapsulated RSPAN	FED	Forwarding Engine Driver
UADP	Unified Access Data Plane	RACL	Router-based ACL
		VACL	VLAN-based ACL

Join at
slido.com
#06 07 2023

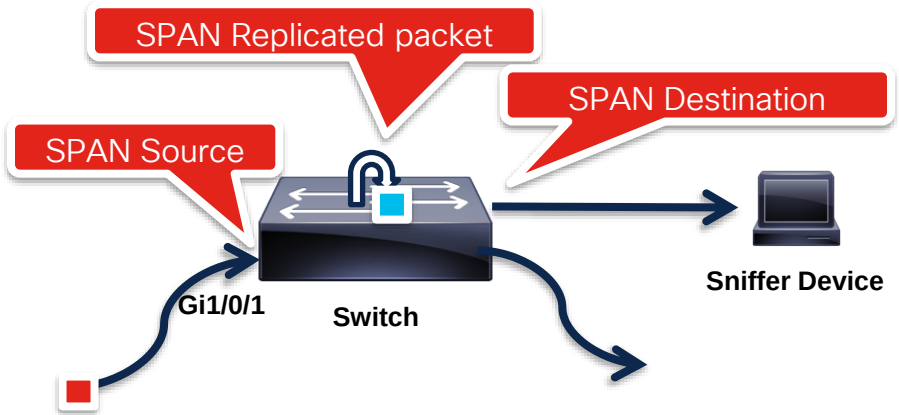
What problems have you solved using packet capturing tools?

Port Mirroring Tools

Switch Port Analyzer (SPAN)

Overview

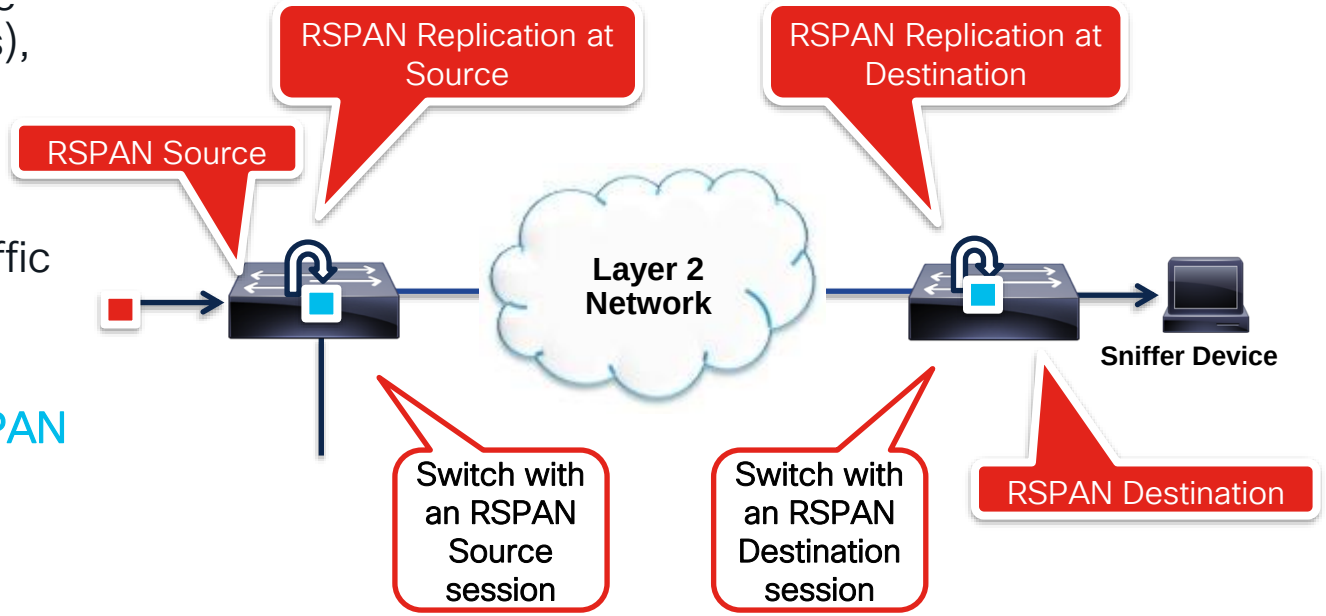
- A SPAN session (also known as port mirroring or monitoring) is an association of source ports/VLANs to one or more destination ports.
- Once the traffic is identified for replication, Cisco switch/router replicates the traffic to the destination port(s).



Remote Switch Port Analyzer (RSPAN)

Overview

- RSPAN supports source ports (or source VLANs), and destinations on different switches
- User-specified **Layer 2 VLAN** carries SPAN traffic between switches
- Consists of an RSPAN source session, an RSPAN VLAN, and an RSPAN destination session



SPAN/RSPAN

Configuration Example

Configuration syntax may differ depending on platform and version of code.

Local SPAN

```
monitor session 1 source <interface/vlan>
monitor session 1 destination interface <interface>
```

```
monitor session 1 type local
 source <interface/vlan>
 destination interface <interface>
 no shutdown
```

RSPAN Source

```
vlan 999
  remote-span
!
monitor session 1 source <interface/vlan>
monitor session 1 destination remote vlan 999
```

```
vlan 999
  remote-span
!
monitor session 1 type rspan-source
 source <interface/vlan>
 destination remote vlan 999
 no shutdown
```

RSPAN Destination

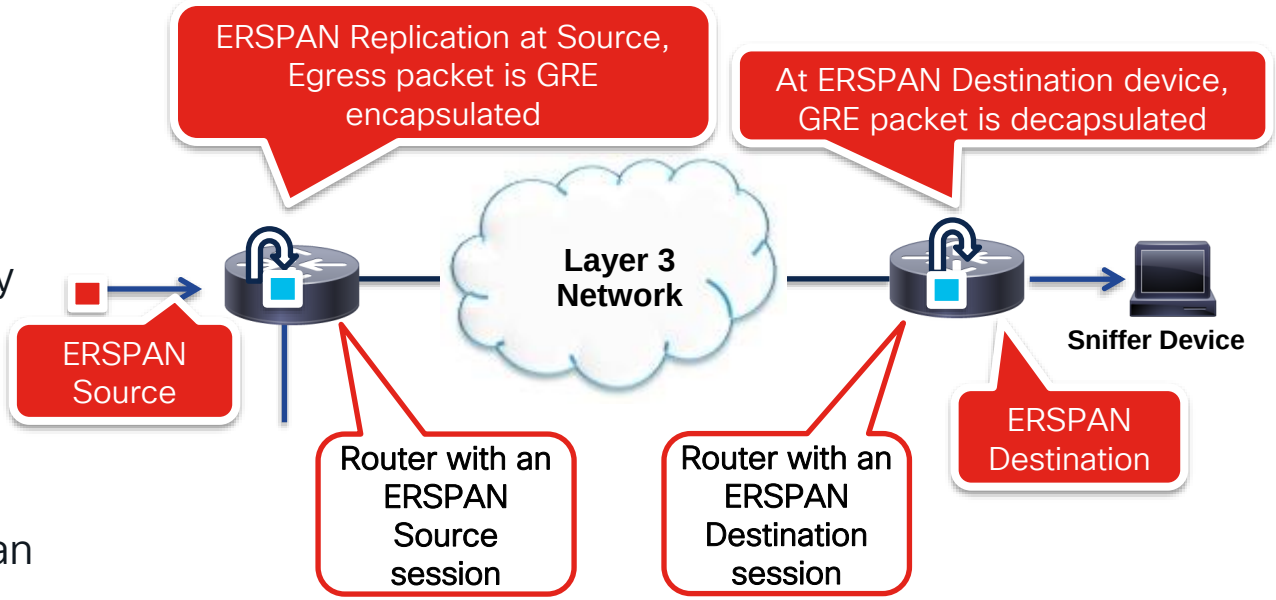
```
vlan 999
  remote-span
!
monitor session 1 source remote vlan 999
monitor session 1 destination interface <interface>
```

```
vlan 999
  remote-span
!
monitor session 1 type rspan-destination
 source remote vlan 999
 destination interface <interface>
 no shutdown
```

Encapsulated Remote SPAN (ERSPAN)

Overview

- ERSPAN supports source ports, source VLANs, and destinations on different devices
- Uses a **Layer 3 Transport**
- Uses a **GRE tunnel** to carry traffic
- ERSPAN consists of an ERSPAN source session, routable ERSPAN GRE-encapsulated traffic, and an ERSPAN destination session



ERSPAN

Configuration Example

```
! Router-1
monitor session 1 type erspan-source
 source interface Gi0/0/1
 destination
  erspan-id 1
  ip address 2.2.2.2
  origin ip address 1.1.1.1
```

ERSPAN Source

```
! Router-2
monitor session 1 type erspan-destination
 destination interface Gi0/0/0
 source
  erspan-id 1
  ip address 2.2.2.2
```

ERSPAN Destination

Using ERSPAN for Local SPAN on an ASR

Overview

- ERSPAN can be used to emulate Local SPAN on an ASR
- Builds an internal tunnel that both encapsulates and de-encapsulates traffic to send it out a local physical port
- Reduces administration and bandwidth overhead of having to transport a tunnel with SPAN traffic across your network
- Useful in situations where another ERSPAN-compatible device is not be present in the network

Using ERSPAN for Local SPAN on an ASR

```
ASR1000(config)# interface lo10
ASR1000(config-if)# ip address 10.1.1.1 255.255.255.255

ASR1000(config)# monitor session 1 type erspan-source
ASR1000(config-mon-erspan-src)# source interface gigabitEthernet 1/2/0 both
ASR1000(config-mon-erspan-src)# destination
ASR1000(config-mon-erspan-src-dst)# erspan-id 100
ASR1000(config-mon-erspan-src-dst)# ip address 10.1.1.1
ASR1000(config-mon-erspan-src-dst)# origin ip address 10.1.1.1
ASR1000(config-mon-erspan-src-dst)# exit
ASR1000(config-mon-erspan-src)# no shut

ASR1000(config)# monitor session 2 type erspan-destination
ASR1000(config-mon-erspan-dst)# destination interface gigabitEthernet 1/2/1
ASR1000(config-mon-erspan-dst)# source
ASR1000(config-mon-erspan-dst-src)# erspan-id 100
ASR1000(config-mon-erspan-dst-src)# ip address 10.1.1.1
ASR1000(config-mon-erspan-dst-src)# exit
ASR1000(config-mon-erspan-dst)# no shut
```

Use a ‘dummy’ loopback interface/address for the tunnel

Using ERSPAN for Local SPAN on an ASR

Verification

```
ASR1000# show monitor session 1

Session 1
---------
Type                      : ERSPAN Source
Session
Status                    : Admin Enabled
Source Ports              :
    Both                  : Gi1/2/0
Destination IP Address    : 10.1.1.1
MTU                       : 1464
Destination ERSPAN ID     : 100
Origin IP Address         : 10.1.1.1
```

```
ASR1000# show monitor session 2

Session 2
---------
Type                      : ERSPAN Destination
Session
Status                    : Admin Enabled
Destination Ports         : Gi1/2/1
Source IP Address         : 10.1.1.1
Source ERSPAN ID          : 100
MTU                       : 1464
```

SPAN / RSPAN / ERSPAN

Limitations and Restrictions

- Limited number of Ingress-SPAN sessions supported. E.g., Catalyst 6500 supports only 2 Ingress-SPAN sessions at max (Egress-Only SPAN supports up to 14 sessions)
- Exercise care when enabling and configuring ER/R/SPAN. The traffic copied by ER/R/SPAN can impose a significant load on the switch and the network. To minimize the load, configure filters to copy only the specific traffic that you want to analyze.
- SPAN adds load to the switch fabric and forwarding engine. Oversubscription at any of these points can cause network disruption.
- The supervisor engine handles the entire load imposed by Egress-SPAN, when the switch is in centralized replication mode. In the Catalyst 6500, 12.2(33)SXH and later releases support **distributed replication** (ingress modules replicate the traffic locally). In the Cisco 7600 routers, 15.2(2)S and later releases support **distributed replication**.
- Some features are incompatible with SPAN Destination ports. E.g., Private VLANs(PVLANs), Port Security, 802.1Q tunneling, 802.1X. DTP. VTP etc.

Please check configuration guide for more details on platforms and software restrictions

Advanced SPAN Filtering

Flow-Based SPAN, VACL and ACL Capture

Overview

- These features allow network administrators to replicate network traffic for monitoring purposes
- Different from traditional SPAN, these techniques provide the ability to selectively monitor traffic of interest via the use of an access-list
- 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

Flow-Based SPAN

Overview

- Apply an IPv4, IPv6, or MAC ACL to filter traffic on a SPAN session

```
ip access-list extended INTERESTING_TRAFFIC
 permit ip host 10.1.100.1 any
 permit ip any host 10.1.100.1
!
monitor session 1 source interface Gi4/0/1
monitor session 1 destination interface Gi4/0/2
monitor session 1 filter ip access-group INTERESTING_TRAFFIC

Switch# show monitor session 1
Session 1
---------
Type                   : Local Session
Source Ports           :
    Both               : Gi4/0/1
Destination Ports      : Gi4/0/2
    Encapsulation      : Native
          Ingress      : Disabled
IP Access-group        : INTERESTING_TRAFFIC
```

Flow-Based SPAN is supported on multiple switching platforms:

- 3560-X, 3750-X >= 12.2(44)SE
- 2960-S, 2960-X
- 3850, 3650
- Catalyst 4000
- Catalyst 9000

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 6500/6800 & Cisco 7600 platforms

VACL Capture

Configuration

```
ip access-list extended INTERESTING_TRAFFIC
 permit ip host 10.1.100.1 any
 permit ip any host 10.1.100.1
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 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

Join at
slido.com
#06 07 2023

True or False

SPAN tools always require physical access to a device.

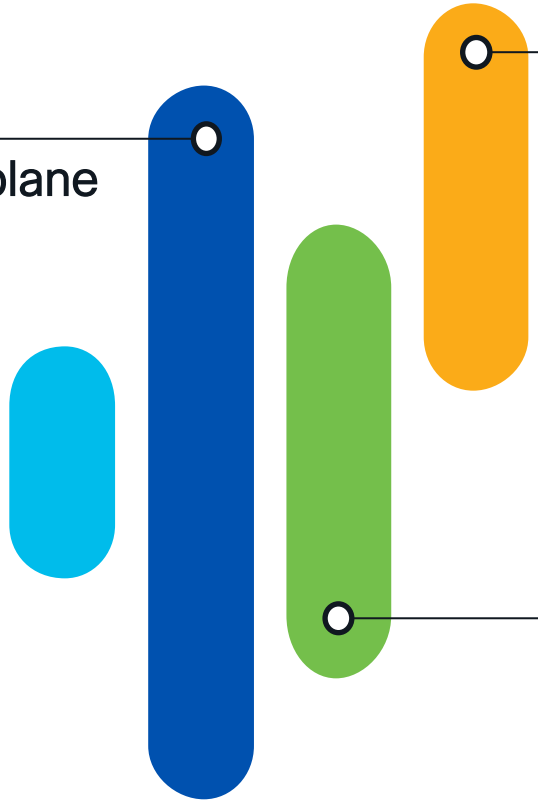
Embedded Packet Capture

Embedded Packet Capture Tools

What are they?

Built-in **control and data plane** capturing

Local buffer for storage



What do they provide?

ACL filtering (**L3/L4 parameters**)

Export to a **PCAP**

On-box analysis

Platforms

IOS-XE Routers and Switches

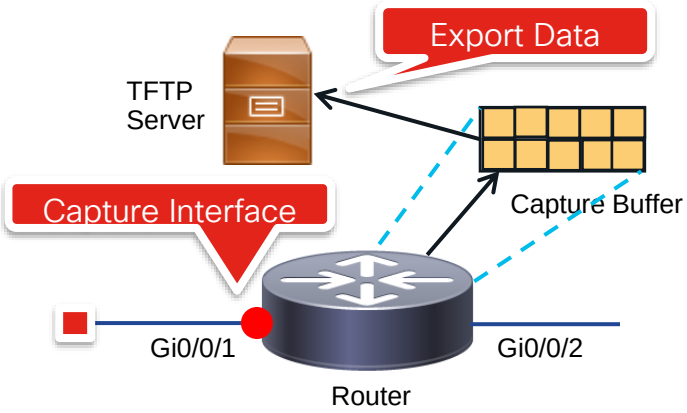
Embedded Packet Capture Tools

Key Advantages and Benefits

- 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
- Capture rate can be throttled using further administrative controls (i.e. Access Control Lists)
- Show commands to display packet contents on the device itself
- Captures multicast traffic only on ingress, not replicated packets on egress
- Some Cisco platforms, only allow EPC on one interface at a time

Embedded Packet Capture (EPC)

Workflow



Define Buffer Criteria → Determine interface and filter → Export/display captured data

```
Router# monitor capture MYCAP interface Gig0/0/1 in
Router# monitor capture MYCAP access-list MYACL
Router# monitor capture MYCAP start
```

Embedded Packet Capture

Configuration Steps

Define Buffer Criteria

```
monitor capture MYCAP buffer [circular] [limit packets <1-10000>]
monitor capture MYCAP buffer size <1-10>
```

Determine Interface and Filter

```
monitor capture MYCAP [interface INTERFACE | control-plane] <in|out|both>
monitor capture MYCAP access-list MYACL
monitor capture MYCAP match [ any | mac <MAC> | <ipv4|ipv6> <any|host IP> <any|host IP> ]
monitor capture MYCAP start
```

Export/Display Captured Data

```
show monitor capture MYCAP parameter
show monitor capture MYCAP buffer [brief|detailed|dump]

show monitor capture file bootflash:MY_CAP.pcap
show monitor capture MYCAP capture-statistics

monitor capture MYCAP export <bootflash:|ftp:|tftp:|sftp:|scp:>
```

*Only available on switches

Embedded Packet Capture (EPC)

Analyzing the Traffic on the Device

```
ASR# show monitor capture CAP parameter
   monitor capture CAP interface Gig0/0/2 both
   monitor capture CAP access-list MYACL
   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)

Analyzing the Traffic on the Device

"detail" option provides result of both "brief" and "dump" options

```
ASR# show monitor capture CAP buffer detail
 ------------------------------------------------------------------
 #   size   timestamp     source      destination   protocol
 ------------------------------------------------------------------
   0  114     0.000000    10.254.0.2       ->  100.100.100.1    ICMP
 0000:  0014A8FF A4020008 E3FFFC28 08004500   ...........(..E.
 0010:  00649314 0000FF01 551F0AFE 00026464   .d......U.....dd
 0020:  64010800 DF8F0017 00000000 000029E8   d.............).
 0030:  74C0ABCD ABCDABCD ABCDABCD ABCDABCD   t...............
```

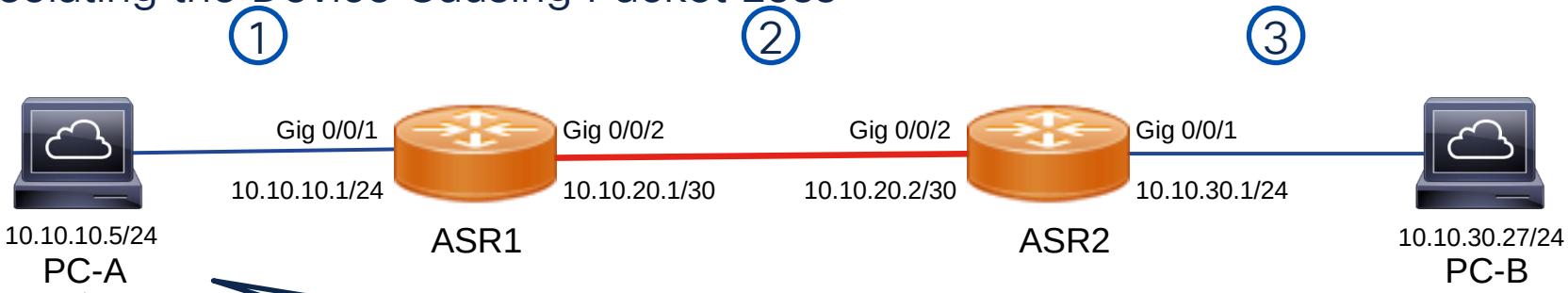
Destination MAC | Source MAC | Source IP | Destination IP

```
ASR# monitor capture CAP export bootflash:my_capture.pcap
Exported Successfully
```

Save capture in standard PCAP format

Real World Example

Isolating the Device Causing Packet Loss

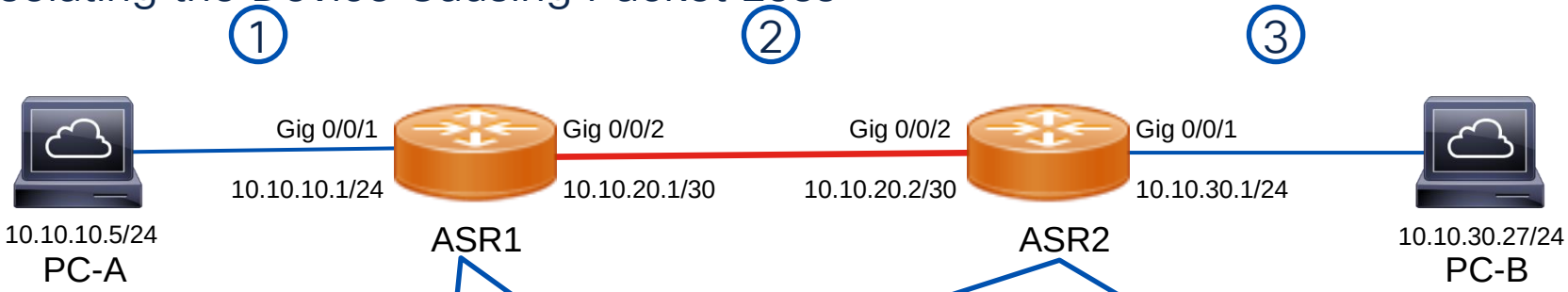



```
C:\>ping 10.10.30.27 -n 5 -l 100
Pinging 10.10.30.27 with 100 bytes of data:
Reply from 10.10.30.27: bytes=100 time<1ms TTL=126
Reply from 10.10.30.27: bytes=100 time<1ms TTL=126
Reply from 10.10.30.27: bytes=100 time<1ms TTL=126
Reply from 10.10.30.27: bytes=100 time<1ms TTL=126
Reply from 10.10.30.27: bytes=100 time<1ms TTL=126
Ping statistics for 10.10.30.27:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss)
```

```
C:\>ping 10.10.30.27 -n 5 -l 1000
Pinging 10.10.30.27 with 1000 bytes of data:
Reply from 10.10.30.27: bytes=1000 time<1ms TTL=126
Reply from 10.10.30.27: bytes=1000 time<1ms TTL=126
Request timed out.
Request timed out.
Reply from 10.10.30.27: bytes=1000 time<1ms TTL=126
Ping statistics for 10.10.30.27:
    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 10.10.30.27
 permit icmp host 10.10.30.27 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
```

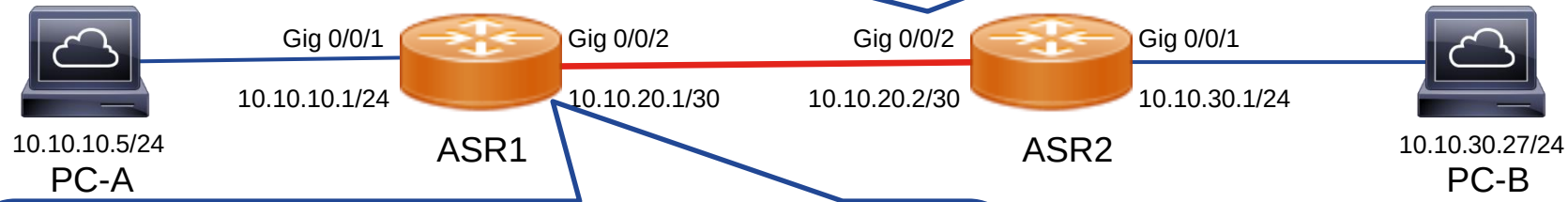
```
! Filter ICMP traffic between hosts
ip access-list extended ICMP1
 permit icmp host 10.10.10.5 host 10.10.30.27
 permit icmp host 10.10.30.27 host 10.10.10.5

! EPC on ASR2
monitor capture CAP2 int Gig0/0/2 both access-list ICMP1
monitor capture CAP2 start
monitor capture CAP2 stop
```

Real World Example

Isolating the Device Causing Packet Loss

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

```
ASR2# show monitor capture CAP2 buffer brief | inc ICMP|#
#    size    timestamp     source              destination     protocol
   0 1014     0.000000    10.10.10.5        ->  10.10.30.27       ICMP
   1 1014     0.003997    10.10.30.27       ->  10.10.10.5        ICMP
   2 1014     0.004013    10.10.10.5        ->  10.10.30.27       ICMP
   3 1014     0.004976    10.10.30.27       ->  10.10.10.5        ICMP
   4 1014     0.005033    10.10.10.5        ->  10.10.30.27       ICMP
   5 1014     2.006834    10.10.10.5        ->  10.10.30.27       ICMP
   6 1014     4.007125    10.10.10.5        ->  10.10.30.27       ICMP
   7 1014     4.008003    10.10.30.27       ->  10.10.10.5        ICMP
```



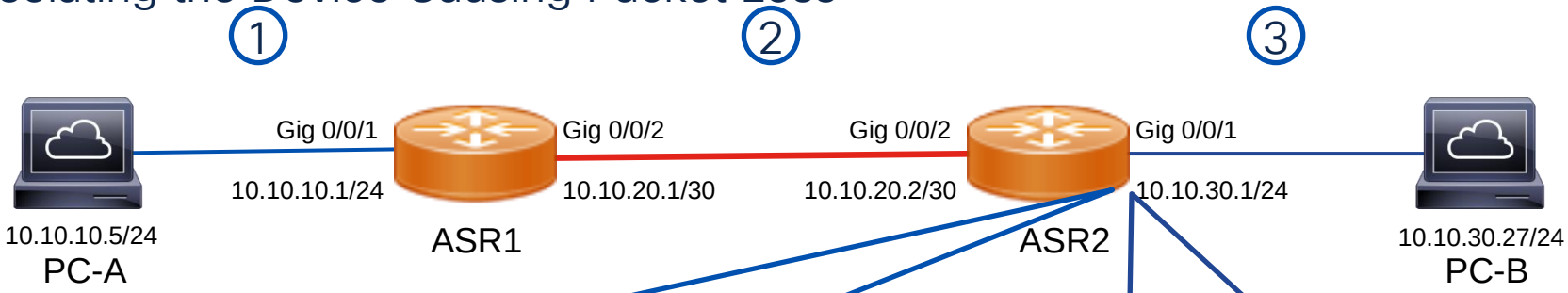

```
ASR1# show monitor capture CAP1 buffer brief | inc ICMP|#
#    size    timestamp     source              destination     protocol
   0 1014     0.000000    10.10.10.5        ->  10.10.30.27       ICMP
   1 1014     0.003998    10.10.30.27       ->  10.10.10.5        ICMP
   2 1014     0.004012    10.10.10.5        ->  10.10.30.27       ICMP
   3 1014     0.004978    10.10.30.27       ->  10.10.10.5        ICMP
   4 1014     0.005031    10.10.10.5        ->  10.10.30.27       ICMP
   5 1014     2.006832    10.10.10.5        ->  10.10.30.27       ICMP
   6 1014     4.007124    10.10.10.5        ->  10.10.30.27       ICMP
   7 1014     4.008006    10.10.30.27       ->  10.10.10.5        ICMP
```

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

Were all of 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 ICMP1
monitor capture CAP2 start
monitor capture CAP2 stop
```

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!

```
ASR2# show monitor capture CAP2 buffer brief | inc ICMP|#
 #   size   timestamp     source           destination     protocol
   0 1014     0.000000    10.10.10.5       ->  10.10.30.27      ICMP
   1 1014     0.003814    10.10.30.27      ->  10.10.10.5       ICMP
   2 1014     0.003838    10.10.10.5       ->  10.10.30.27      ICMP
   3 1014     0.004612    10.10.30.27      ->  10.10.10.5       ICMP
   4 1014     4.008062    10.10.10.5       ->  10.10.30.27      ICMP
   5 1014     4.008822    10.10.30.27      ->  10.10.10.5       ICMP
```

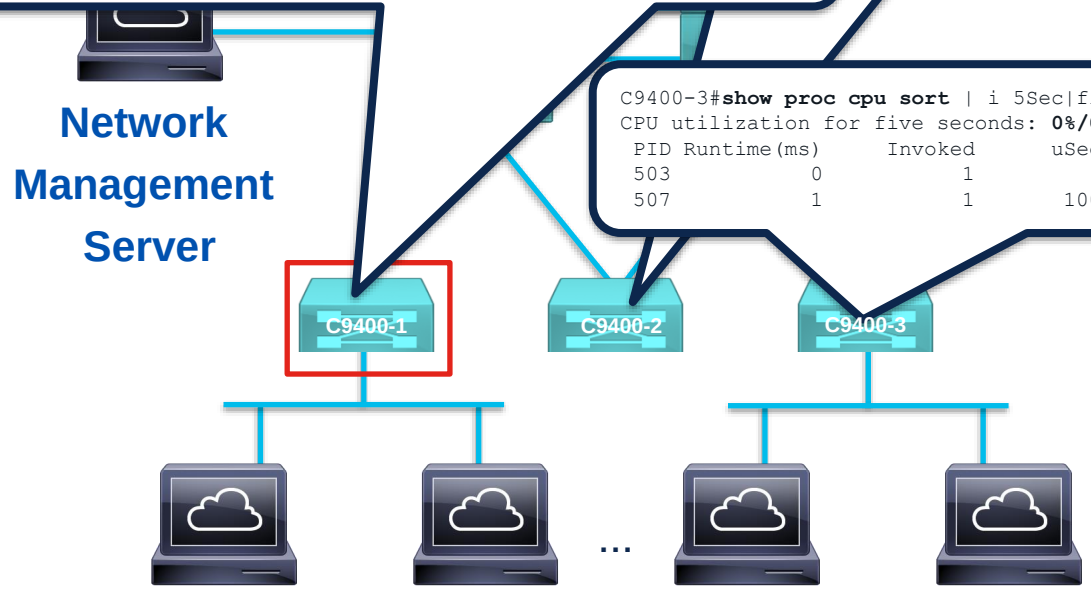
Real World Example

Access Switch with High CPU Utilization in SNMP Engine

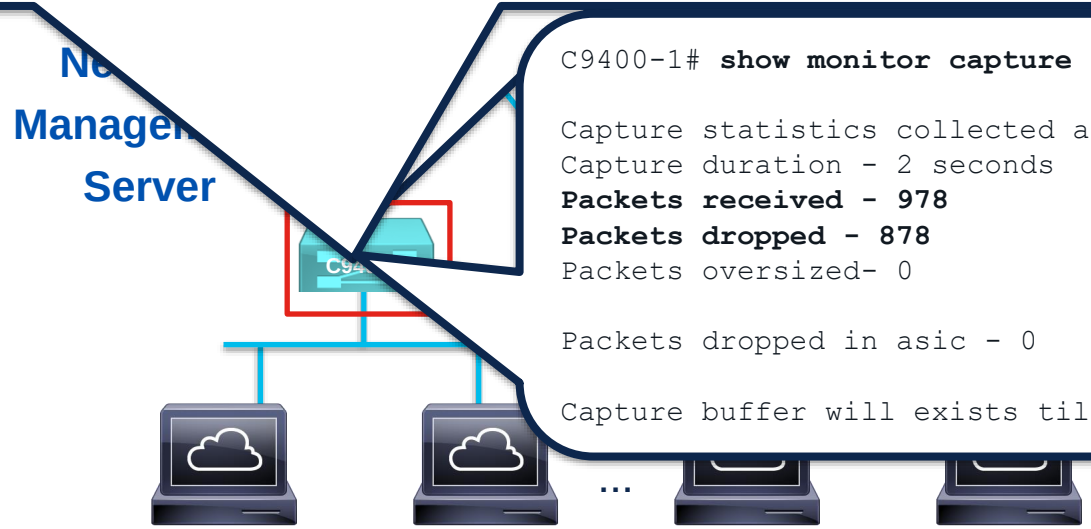
```
C9400-2#show proc cpu sort | i 5Sec|five seconds|SNMP ENGINE|IP SNMP
CPU utilization for five seconds: 0%/0%; one minute: 0%; five minutes: 0%
 PID Runtime(ms)     Invoked      uSecs   5Sec   1Min   5Min TTY Process
 503           0           1          0  0.00%  0.00%  0.00%   0 IP SNMP
 507           1           1       1000  0.00%  0.00%  0.00%   0 SNMP ENGINE
```

```
C9400-1#show proc cpu sort | i 5Sec|five seconds|SNMP ENGINE|IP SNMP
CPU utilization for five seconds: 18%/6%; one minute: 17%; five minutes: 16%
 PID Runtime(ms)     Invoked      uSecs   5Sec   1Min   5Min TTY Process
 464      601422    17193042         34  5.19%  5.11%  4.71%   0 SNMP ENGINE
 432      386396    25797588         14  3.27%  3.04%  2.93%   0 IP SNMP
```

```
C9400-3#show proc cpu sort | i 5Sec|five seconds|SNMP ENGINE|IP SNMP
CPU utilization for five seconds: 0%/0%; one minute: 0%; five minutes: 0%
 PID Runtime(ms)     Invoked      uSecs   5Sec   1Min   5Min TTY Process
 503           0           1          0  0.00%  0.00%  0.00%   0 IP SNMP
 507           1           1       1000  0.00%  0.00%  0.00%   0 SNMP ENGINE
```



```
C9400-1# monitor capture snmp_cap control-plane in limit packets 100
C9400-1# monitor capture snmp_cap match ipv4 protocol udp any any eq 161

C9400-1# monitor capture snmp_cap start

Enabling Control plane capture may seriously impact system performance. Do you want to continue?
[yes/no]: yes
Started capture point : snmp_cap

%BUFCAP-6-ENABLE: Capture Point snmp_cap enabled.

%BUFCAP-6-ENABLE: Capture Point snmp_cap disabled.
```




```
C9400-1# show monitor capture snmp_cap capture-statistics

Capture statistics collected at software:
Capture duration - 2 seconds
Packets received - 978
Packets dropped - 878
Packets oversized- 0

Packets dropped in asic - 0

Capture buffer will exists till exported or cleared
```

Real World Example

Access Switch with High CPU utilization in SNMP Engine

```
C9400-1# show monitor capture snmp_cap buffer brief
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

  1    0.000000    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  2    0.000003    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  3    0.000005    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  4    0.000007    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  5    0.000009    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  6    0.000010    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  7    0.000012    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  8    0.000019    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
  9    0.000021    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0
 10    0.000023    172.16.1.25 -> 172.16.1.11    SNMP 90 get-next-request 1.3.6.1.4.1.9.2.1.1.0

C9400-1# show monitor capture snmp_cap buffer brief | count 1.3.6.1.4.1.9.2.1.1.0
Number of lines which match regexp = 97

C9400-1# monitor capture snmp_cap clear
```

FED Capture

FED Packet Capture

Catalyst 9000

Forwarding Engine Driver (FED) is the heart of Cisco Unified Access switching platforms and is responsible for hardware programming and forwarding.

- **Non-Intrusive Debug**
- Linear buffer [circular], 4096 frames [256-16384]

Direction

From CPU's Perspective

- Punt (Rx)
- Inject (Tx)

Wireshark Based Filters

- Display Filters:
  http://wiki.wireshark.org/DisplayFilters
  `"eth.addr==00:00:0c:07:ac:01"`
  `"ip.src==10.1.1.1 && ip.dst==10.1.1.2"`

- Capture Filters:
  http://wiki.wireshark.org/CaptureFilters
  `"ether host 00:00:0c:07:ac:01"`
  `"src host 10.1.1.1 and dst host 10.1.1.2"`

FED Packet Capture

Filter Types

Capture Filters:

```
CAT9300-1# debug platform software fed switch active [punt|inject] packet-capture  ?
  buffer         Configure packet capture buffer
  clear-filter   Clear punt PCAP filter
  set-filter     Specify wireshark like filter (Punt PCAP)
  start          Start punt packet capturing
  stop           Stop punt packet capturing
```

To clear the filter, use the 'clear-filter' option.

Capture filters are configured with the 'set-filter' option on the 'debug' command before the capture is started.

Display Filters:

```
CAT9300-1# show platform software fed switch active [punt|inject] packet-capture ?
  brief                Display captured packets
  detailed             Display in detailed format
  display-filter       Specify a wireshark like display filter
  display-filter-help  List all filters supported here
  status               Show packet capturing status
```

Display filters are issued with the 'display-filter' option on the 'show' command after capture.

Using 'display-filter-help' option will provide a long list of platform specific display filters.

FED Packet Capture
Filter Examples

Capture Filters:

Capture Only Spanning-Tree Packets:

```
CAT9300-1# debug platform software fed switch active [punt|inject] packet-capture set-filter "stp"
```

Capture Only Packets to/from 10.1.1.1 or from 192.168.1.1:

```
CAT9300-1# debug platform software fed switch active [punt|inject] packet-capture set-filter
"ip.addr == 10.1.1.1 or ip.src == 192.168.1.1"
```

Display Filters:

Display Only Packets with a Source or Destination of 3c51.0e7c.0182:

```
CAT9300-1# show platform software fed switch active [punt|inject] packet-capture display-filter
"eth.addr==3c51.0e7c.0182" brief
```

Display Only ARP Packets:

```
CAT9300-1# show platform software fed switch active [punt|inject] packet-capture display-filter
"arp" brief
```

Remember: FED Packet Capture is in the control-plane.

FED Packet Capture - Example

```
CAT9300-1# debug platform software fed switch active punt packet-capture start
Punt packet capturing started.

CAT9300-1# show platform software fed switch active punt  packet-capture status
Punt packet capturing: enabled. Buffer wrapping: disabled
Total captured so far: 15 packets. Capture capacity : 4096 packets

CAT9300-1# debug platform software fed switch active punt packet-capture stop
Punt packet capturing stopped. Captured 24 packet(s)

CAT9300-1# show platform software fed switch active punt packet-capture brief
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 24 packets. Capture capacity : 4096 packets

------ Punt Packet Number: 1, Timestamp: 2023/01/02 17:52:50.461 ------
 interface : physical: GigabitEthernet1/0/1[if-id: 0x00000009], pal: GigabitEthernet1/0/1
[if-id: 0x00000009]
 metadata  : cause: 58 [Layer2 bridge domain data packet], sub-cause: 11, q-no: 4, linktype:
MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 0100.5e00.000d, src mac: 001b.53bb.bbc5
 ether hdr : ethertype: 0x0800 (IPv4)
 ipv4  hdr : dest ip: 224.0.0.13, src ip: 10.122.164.3
 ipv4  hdr : packet len: 58, ttl: 1, protocol: 103
```

FED Packet Capture - Example

```
CAT9300-1# debug platform software fed switch active [punt|inject] packet-capture [start|stop]
Punt packet capturing started.

CAT9300-1# show platform software fed switch active punt packet-capture status
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 4096 packets. Capture capacity : 4096 packets

CAT9300-1# show platform software fed switch active punt packet-capture brief
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 4096 packets. Capture capacity : 4096 packets

------ Punt Packet Number: 1, Timestamp: 2023/01/02 18:22:51.757 ------
 interface : physical: GigabitEthernet1/0/2[if-id: 0x0000000a], pal: Vlan101 [i
0x00000042]
 metadata  : cause: 11 [For-us data], sub-cause: 0, q-no: 2, linktyp
 ether hdr : dest mac: 3c51.0e7c.01c1, src mac: 0016.c81c.2f81
 ether hdr : ethertype: 0x0800 (IPv4)
 ipv4  hdr : dest ip: 10.1.1.2, src ip: 10.1.1.1
 ipv4  hdr : packet len: 100, ttl: 255, protocol: 1 (ICMP)
 icmp  hdr : icmp type: 8, code: 0
```

Capture Complete!
If the capture is still running use the 'stop' command.

Ingress on port Gi1/0/2 on VLAN 101

Source and Destination MAC Address

Ethertype is 0x0800 for IPv4

Source and Destination IPv4 Address

FED Packet Capture - Example

```
CAT9300-1# show platform software fed switch active punt packet-capture detailed
<snip>
------ Punt Packet Number: 2, Timestamp: 2023/01/02 18:22:51.757 ------
 interface : physical: GigabitEthernet1/0/2[if-id: 0x0000000a], pal: Vlan101 [if-id:
0x00000042]
 metadata  : cause: 11 [For-us data], sub-cause: 0, q-no: 2, linktype:
MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: 3c51.0e7c.01c1, src mac: 0016.c81c.2f81
 ether hdr : ethertype: 0x0800 (IPv4)
 ipv4  hdr : dest ip: 10.1.1.2, src ip: 10.1.1.1
 ipv4  hdr : packet len: 100, ttl: 255, protocol: 1 (ICMP)
 icmp  hdr : icmp type: 8, code: 0

 Packet Data Hex-Dump (length: 118 bytes) :
   3C510E7C01C10016  C81C2F8108004500  0064A2BF0000FF01  02D50A0101010A01
   0102[illegible]00CEEE000[illegible]  17A80000000000FD  96A8ABCDABCDABCD  ABCDABCDABCDABCD
   ABCD[illegible]BCDABCDABC[illegible]  ABCDABCDABC[illegible]ABCD  ABCDABCDABCDABCD  [illegible]CDABCDABCD[illegible]CD
   AB[illegible]BCDABCDAB[illegible]  ABCDABCDABC[illegible]CD  ABCD1FECA2[illegible]
```

Punt Cause

ICMP Packet

Dest MAC

Source MAC

Ethertype 0x0800

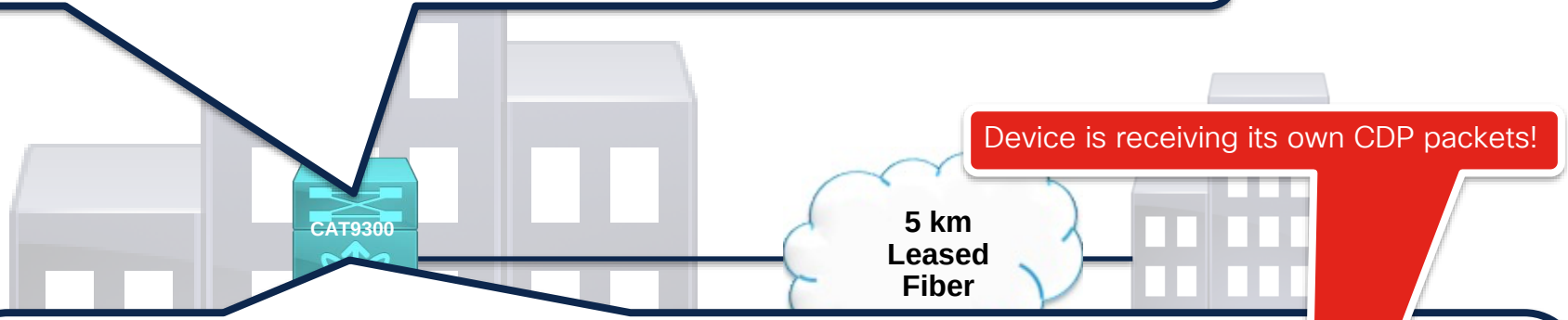
Source IP
10.1.1.1

Dest IP
10.1.1.2

Real World Example

High CPU due to CDP Protocol

```
CAT9300-1# show proc cpu sort
CPU utilization for five seconds: 27%/7%; one minute: 28%; five minutes: 20%
 PID Runtime(ms)     Invoked      uSecs   5Sec   1Min   5Min TTY Process
 227     1196642     9627692        124 16.87% 17.20% 12.43%   0 CDP Protocol
```




```
CAT9300-1# show cdp neighbors
Capability Codes: R - Router, T - Trans Bridge, B - Source Route Bridge
                  S - Switch, H - Host, I - IGMP, r - Repeater, P - Phone,
                  D - Remote, C - CVTA, M - Two-port Mac Relay
Device ID        Local Intrfce     Holdtme    Capability  Platform  Port ID
ASR1000.cisco.com   Gig 1/0/3        158             R I   ASR1006   Gig 0
ISR4K-1.cisco.com   Gig 1/0/1        155           R S I   ISR4451-X Gig 0/0/3
ISR4K-2.cisco.com   Gig 1/0/4        136           R S I   ISR4351/K Gig 0/0/0
CAT9300-1.cisco.com Gig 1/0/2        179             S I   C9300L-48 Gig 1/0/2
Total cdp entries displayed : 4
```

Real World Example

High CPU due to CDP Protocol

```
CAT9300-1# debug platform software fed switch active punt packet-capture start
Punt packet capturing started.

CAT9300-1# show platform software fed switch active punt packet-capture status
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 4096 packets. Capture capacity : 4096 packets

CAT9300-1# show platform software fed switch active punt packet-capture brief
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 4096 packets. Capture capacity : 4096 packets
------ Punt Packet Number: 1, Timestamp: 2023/01/02 19:22:1[illegible]
 interface : physical: GigabitEthernet1/0/2[if-id: 0x0000000a],
   0x0000000a]
 metadata  : cause: 96 [Layer2 control protocols], sub-cause: 0,[illegible]
   MCP_LINK_TYPE_LAYER2 [10]
 ether hdr : dest mac: 0100.0ccc.cccc, [illegible] mac: 3c51.0e7c.0182
 ether hdr : length: 449

CAT9300-1# debug plat soft fed switch act punt packet-capture set-filter "eth.addr == 0100.0ccc.cccc"
CAT9300-1# debug platform software fed switch active punt packet-capture start
```

After identifying the flow a short-term solution would be to disable CDP on G1/0/2 until the link issue is resolved:

```
CAT9300-1(config)# int G1/0/2
CAT9300-1(config-if)# no cdp enable
```

Leased Fiber link for the ingress interface

CDP Destination MAC for filter to narrow results

Packet Trace
Routing Platforms

ASR1000 Acronyms

- MCP – Midrange Converged Platform (codename for ASR1000 during development)
- RP – Route Processor
- FP – Forwarding Processor = ESP (Embedded Service Processor)
- CPP – Cisco Packet Processor Complex= QFP (Quantum Flow Processor)
- PPE – Packet Processing Engine
- IOCP – I/O Control Processor
- FECP – Forwarding Engine Control Processor
- SPA – Shared Port Adapter
- SIP – SPA Interface Processor
- IOSd – IOS image that runs as a process on the RP
- FMAN – Forwarding manager (FMAN-RP, FMAN-FP)
- Scbac – FW Session Control Block
- **EOBC** = Ethernet Out of Band Channels – Packet Interface for Card to Card Control Traffic
- **IOS-XE (BinOS)** = Linux Based Software Infrastructure That Executes on MCP

Packet Trace Details

- Gain a deep understanding of the actions taken on a packet during packet processing
- Integrates with **`debug platform condition`** for filtering
- For **control** and **data plane** traffic in the datapath

Packet Trace is supported on the CAT8000, ASR1000, ISR1000, ISR4000, and CSR1000V

Packet Trace: Accounting

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

Packet Trace: Summary Data

- When enabled, summary data is collected for a specified number of packets and includes:
  - Packet number (pactrac specific packet number)
  - Input interface
  - Output interface
  - Final packet state and any punt/drop/inject codes
- Summary data collection incurs minimal overhead over normal packet processing
- Often used to isolate specific drop conditions so more detailed inspection can be used after applying specific conditions

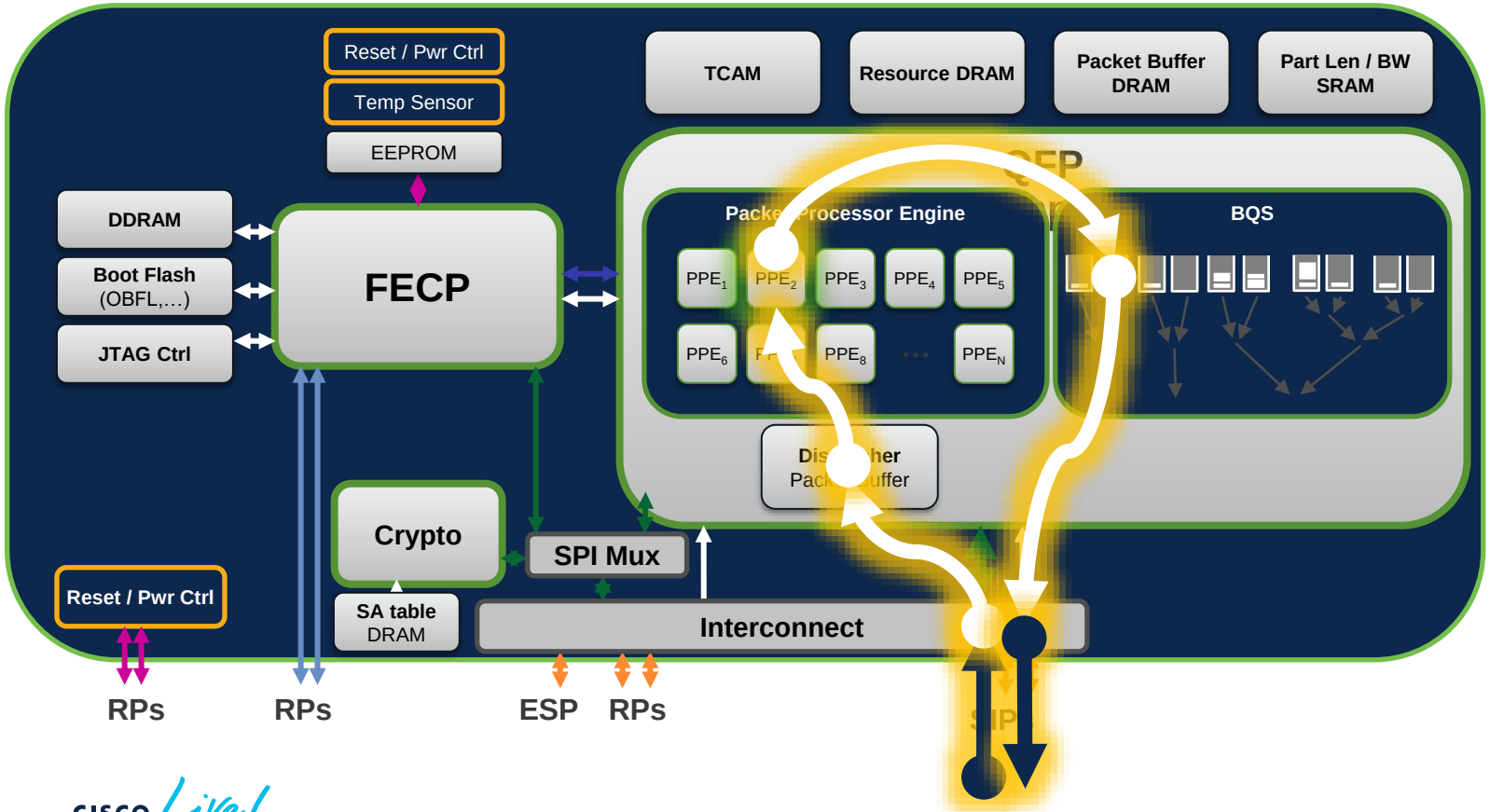
Packet Trace: Forwarding Data

Three levels of forwarding data:

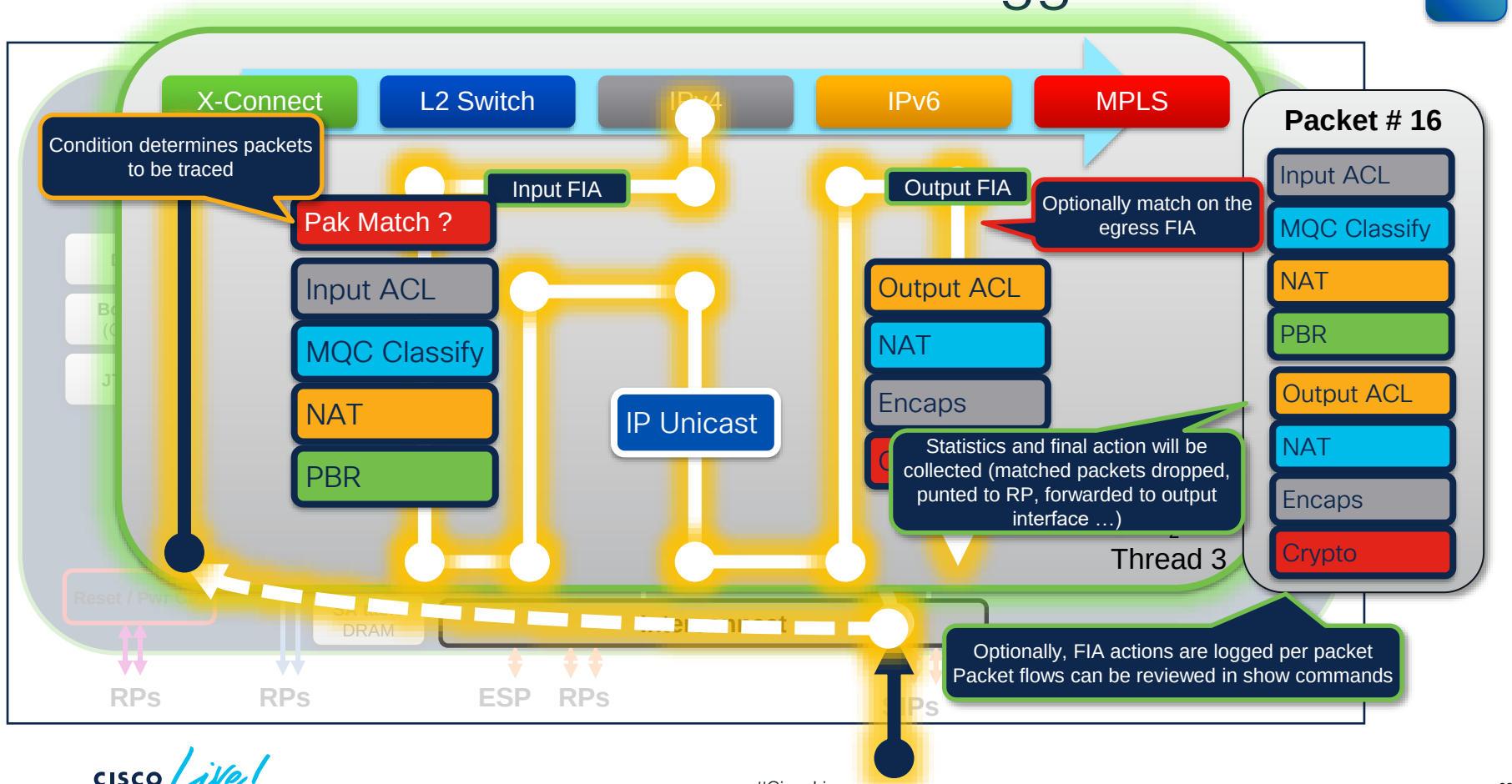
- Common forwarding data (e.g. IP tuple)
- Feature specific data (e.g. NAT)
- Feature Invocation Array (FIA) trace – optionally enabled (e.g. fia-trace)

- Copy all or part of the incoming and/or outgoing packet – optionally enabled
- Safe to use with **appropriate filters** in place
- Potential delay for traced packets only (with **fia-trace** and **packet copy**)

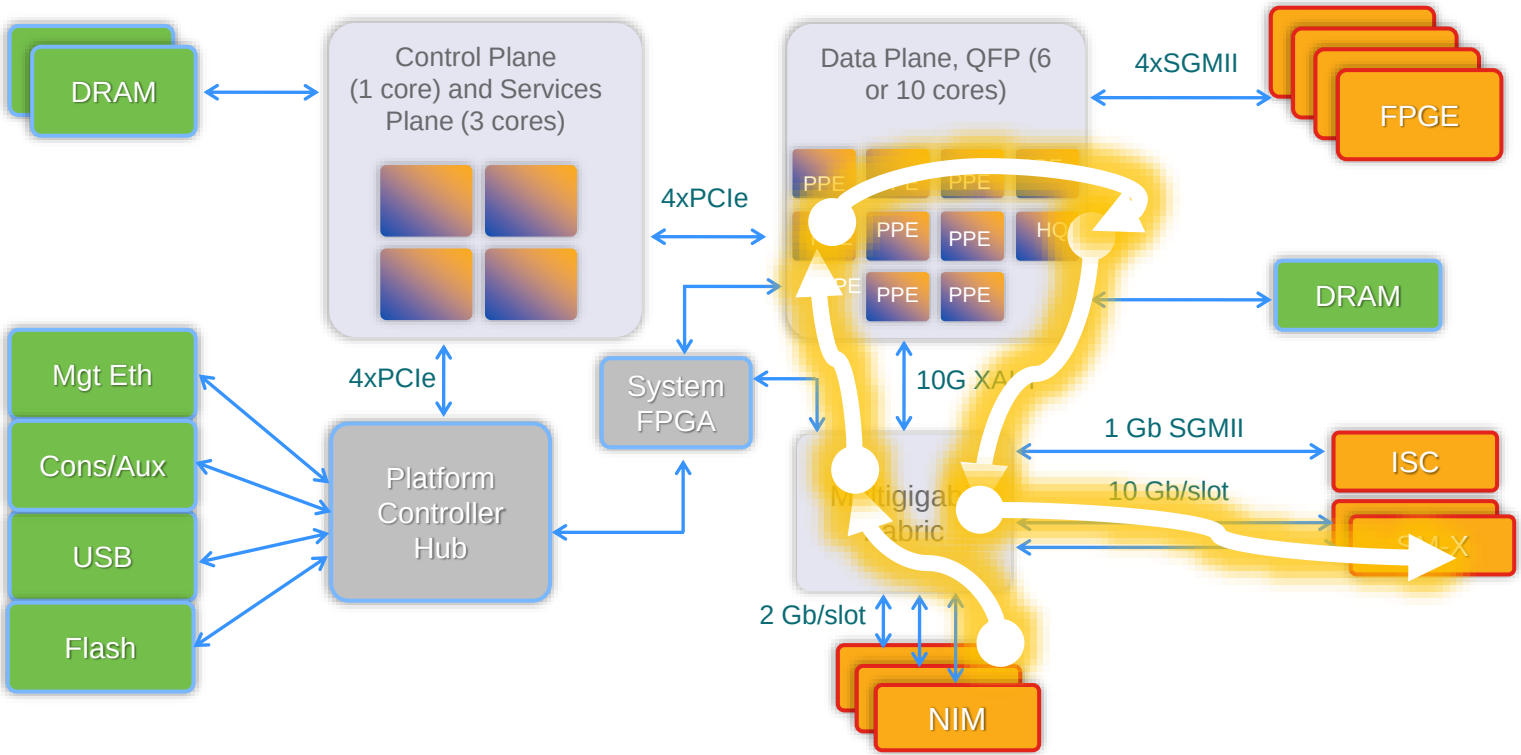
Cisco ASR1000 Packet Flow



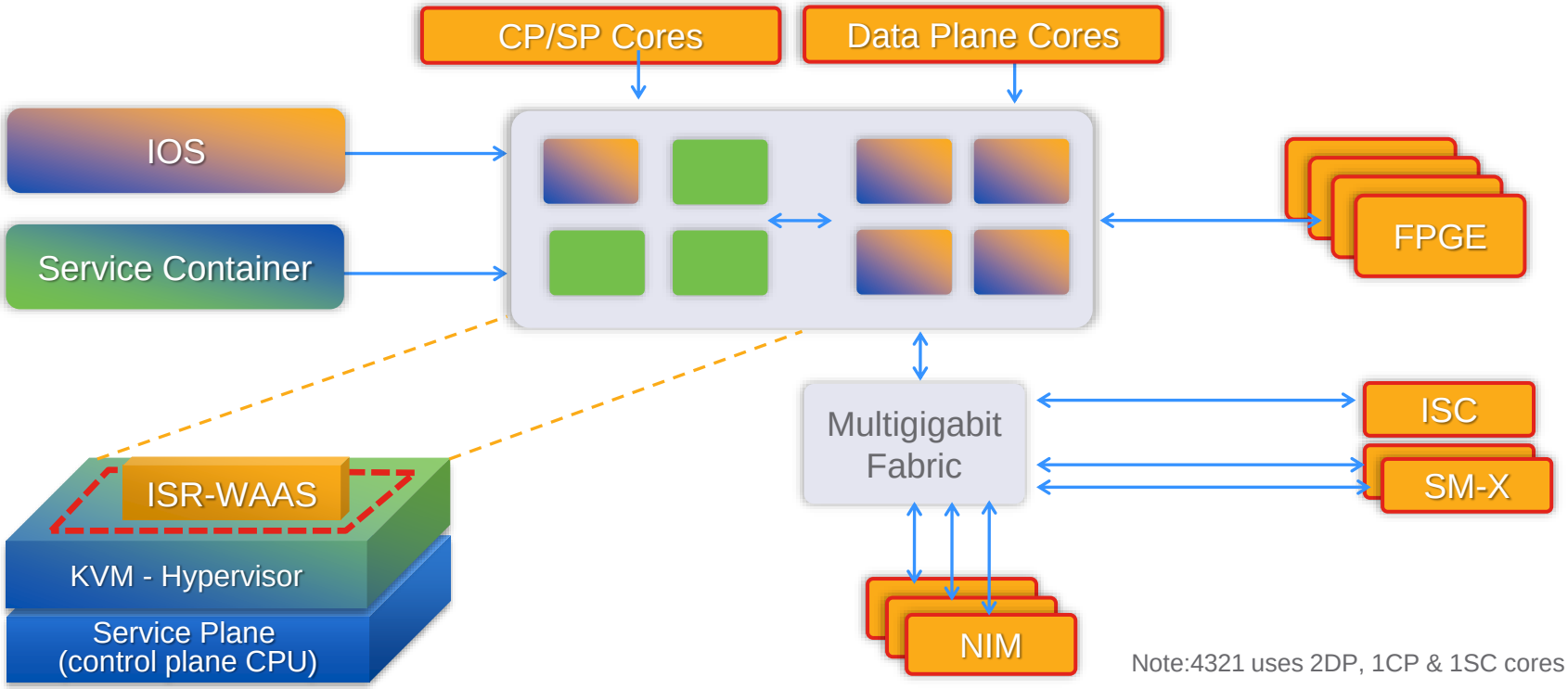
The Packet Trace and FIA Debugger



Cisco ISR 4400 Packet Flow



Cisco ISR 4300 Series Architecture



Configuring Packet Trace (Part 1)

- **debug platform packet-trace copy packet {in | out | both} [L2 | L3 | L4] [size** *num-bytes* **]**
  - Needs to be done first to enable packet tracing

- **debug platform packet-trace packet** *pkt-num* **[fia-trace | summary-only] [circular] [data-size** *data-size* **]**
  - *pkt-num* –Specifies the maximum number of packets maintained at a given time.
  - **fia-trace** –Provides detailed level of data capture, including summary data, feature-spec
  - **summary-only** –Enables the capture of summary data with minimal details.
  - **circular** –Saves the data of the most recently traced packets.
  - *data-size* –Specifies the size of data buffers for storing feature

Configuring Packet Trace (Part 2)

- **debug platform packet-trace copy packet {in | out | both} [L2 | L3 | L4] [size** *num-bytes* **]**
  - Enables copying of the packet
  - Used to specify ingress, egress or both
  - Optionally, allows specifying the layer of the packet in which the packet copy should start (default L2)

- **debug platform packet-trace drop [code** *code-num* **]**
  - Limits the packet trace to only dropped packets
  - Optionally **code** specifies limiting to a specific drop code

Configuring Packet Trace (Part 3)

- debug platform condition start
  - Enables the specified matching criteria and starts packet tracing

- debug platform condition stop
  - Deactivates the condition and stops packet tracing.

Configuring Packet Trace (Part 4)

- **show platform packet-trace {configuration | statistics | summary | packet {all | *pkt-num* } [decode] }**
  - Displays packet-trace data
  - **configuration** – Displays packet trace configuration, including any defaults.
  - **statistics** – Displays accounting data for all the traced packets.
  - **summary** – Displays summary data for the number of packets specified.
  - **{all | *pkt-num* } [decode]** – Displays the path data for all the packets or the packet specified. The **decode** option attempts to decode the binary packet into a more human- readable form.
- **clear platform condition all**
  - Removes the configurations provided by the **debug platform condition** and **debug platform packet-trace** command

Packet Trace

Workflow


```
debug platform packet-trace packet <16-8192> [circular] fia-trace
debug platform packet-trace copy packet <input|output|both> [L2|L3|L4] [size <16-2048>]
```

```
show platform conditions
show platform packet-trace configuration
clear platform condition all
```

8000

Packet Trace

Configuration Steps

Define Buffer Criteria

```
debug platform packet-trace packet <16-8192> [circular] fia-trace
debug platform packet-trace copy packet <both|input|output> [L2|L3|L4] [size <16-2048>]
```

Define Condition/Start/Stop

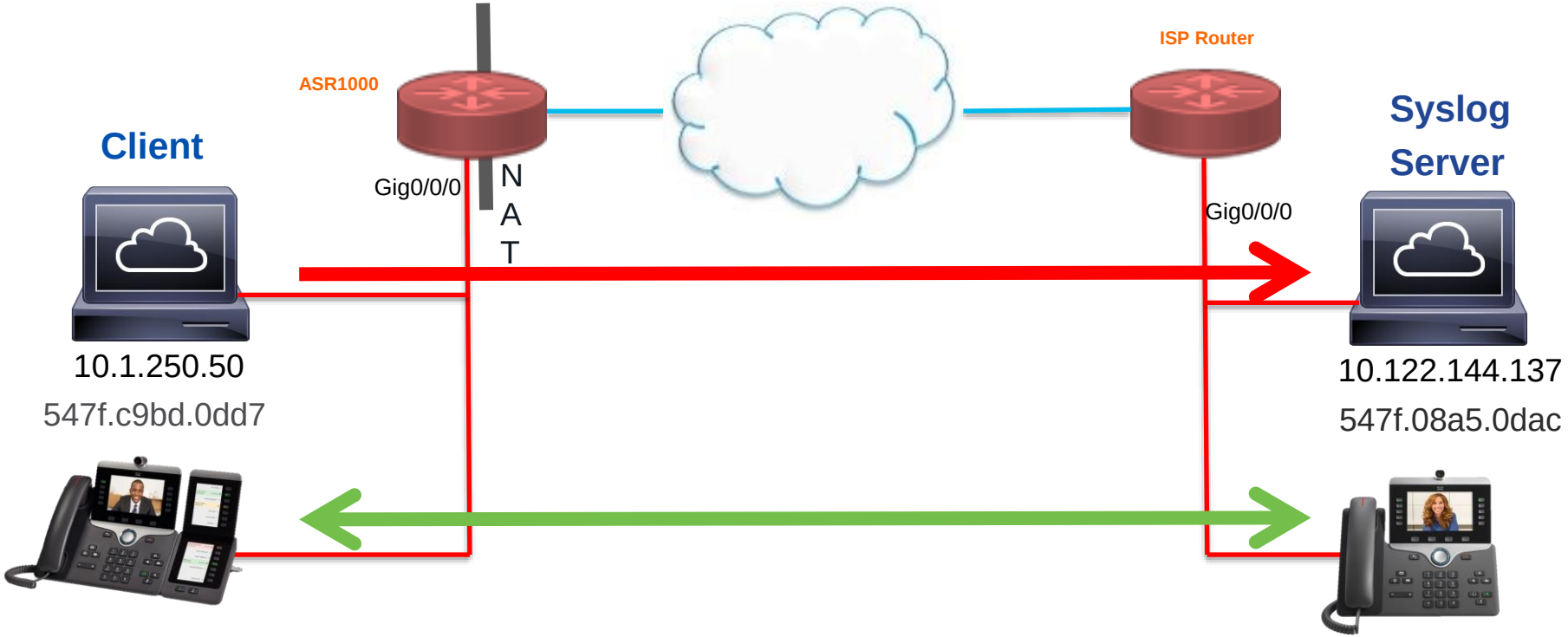
```
debug platform condition <ipv4|ipv6> access-list <#> <both|ingress|egress>
debug platform condition start
debug platform condition stop
```

Review Data/Clear Buffer

```
show platform packet-trace summary
show platform packet-trace packet all
show platform packet-trace packet 5

show platform packet-trace configuration
clear platform condition all
```

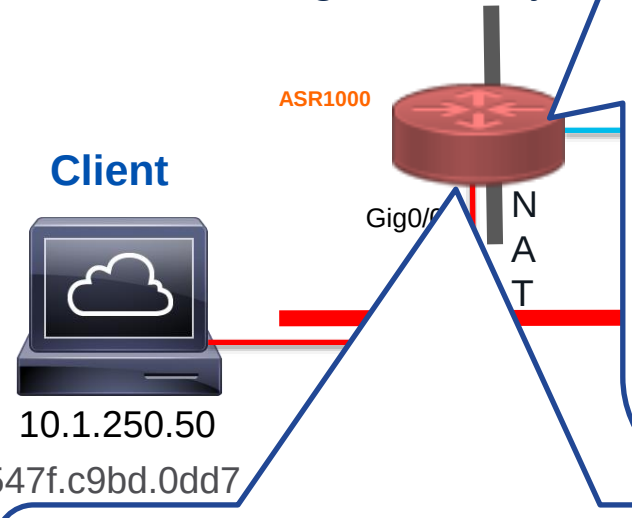
Real World Example

Voice and Ping Work, Syslog Does Not



Real World Examp

Voice and Ping Work, Syslo




```
ASR1000# show platform packet-trace summary
Pkt    Input                Output               State  Reason
0      Gi0/0/3              Gi0/0/0              DROP   187 (FirewallPolicy)
1      Gi0/0/3              Gi0/0/0              DROP   187 (FirewallPolicy)
2      Gi0/0/3              Gi0/0/0              DROP   187 (FirewallPolicy)
3      Gi0/0/3              Gi0/0/0              FWD
4      Gi0/0/3              Gi0/0/0              FWD
5      Gi0/0/3              Gi0/0/0              FWD
6      Gi0/0/3              Gi0/0/0              FWD
7      Gi0/0/3              Gi0/0/0              FWD
8      Gi0/0/3              Gi0/0/0              DROP   187 (FirewallPolicy)
9      Gi0/0/3              Gi0/0/0              DROP   187 (FirewallPolicy)
10     Gi0/0/3              Gi0/0/0              DROP   187 (FirewallPolicy)
11     Gi0/0/3              Gi0/0/0              DROP   187 (FirewallPolicy)
12     Gi0/0/3              Gi0/0/0              FWD
13     Gi0/0/3              Gi0/0/0              FWD
14     Gi0/0/3              Gi0/0/0              FWD
```

```
ASR1000# debug platform packet-trace packet 50 fia-trace data-size 2048
ASR1000# debug platform  packet-trace copy packet output size 128 L2
ASR1000# debug platform condition ipv4 10.1.250.50/32  egress
ASR1000# debug platform condition start
! Perform testing to generate traffic
ASR1000# debug platform condition stop
```

```
ASR1000# show platform packet-trace packet 3
acket: 3            CBUG ID: 137
Summary
  Input     : GigabitEthernet0/0/3
  Output    : GigabitEthernet0/0/0
  State     : FWD
  Timestamp
    Start   : 2417005427340888 ns (04/19/2023  9:02:
    Stop    : 2417005427446059 ns (04/19/2023  9:02:
Path Trace
  Feature: IPV4(Input)
    Input       : GigabitEthernet0/0/3
    Output      : <unknown>
    Source      : 10.1.250.50
    Destination : 10.122.144.137
    Protocol    : 1 (ICMP)
  Feature: DEBUG_COND_INPUT_PKT
    Entry       : Input - 0x11460100
    Input       : GigabitEthernet0/0/3
    Output      : <unknown>
    Lapsed time : 2560 ns
```

```
Feature: IPV4_OUTPUT_INSPECT
   Entry       : Output - 0x1145fb8c
   Input       : GigabitEthernet0/0/3
   Output      : GigabitEthernet0/0/0
   Lapsed time : 1936800 ns
 Feature: NAT
   Direction   : IN to OUT
   Action      : Translate Source
   Steps       : SESS-FD
   Match id    : 2
   Old Address : 10.1.250.50
   New Address : 10.122.144.136
 Feature: IPV4_NAT_OUTPUT_FIA
   Entry       : Output - 0x11460298
   Input       : GigabitEthernet0/0/3
   Output      : GigabitEthernet0/0/0
   Lapsed time : 113200 ns
 Feature: IPV4_OUTPUT_THREAT_DEFENSE
   Entry       : Output - 0x1145fb00
   Input       : GigabitEthernet0/0/3
   Output      : GigabitEthernet0/0/0
   Lapsed time : 5680 ns
 <snip>
   Lapsed time : 7220 ns
 Feature: BM_TRANSMIT_PKT
   Entry       : Output - 0x114601c0
   Input       : GigabitEthernet0/0/3
   Output      : GigabitEthernet0/0/0
   Lapsed time : 206040 ns
```

```
ASR1000# show platform packet-trace packet 8
Packet: 8           CBUG ID: 142
Summary
  Input     : GigabitEthernet0/0/3
  Output    : GigabitEthernet0/0/0
  State     : DROP 187 (FirewallPolicy)
  Timestamp
    Start   : 2417018445350231 ns (04/19/2023 19:02
    Stop    : 2417018445407738 ns (04/19/2023 19:02
Path Trace
  Feature: IPV4(Input)
    Input       : GigabitEthernet0/0/3
    Output      : <unknown>
    Source      : 10.1.250.50
    Destination : 10.122.144.137
    Protocol    : 17 (UDP)
      SrcPort   : 63478
      DstPort   : 514
  Feature: DEBUG_COND_INPUT_PKT
    Entry       : Input - 0x11460100
    Input       : GigabitEthernet0/0/3
    Output      : <unknown>
    Lapsed time : 5140 ns
  Feature: IPV4_INPUT_DST_LOOKUP_CONSUME
    Entry       : Input - 0x1145fe74
    Input       : GigabitEthernet0/0/3
    Output      : <unknown>
    Lapsed time : 9380 ns
ASR1000# debug platform condition stop
```

```
Feature: ZBFW
    Action  : Drop
    Reason  : ICMP policy drop:classify result
    Zone-pair name          : IN-TO-OUT
    Class-map name          : class-default
    Input interface         : GigabitEthernet0/0/3
    Egress interface        : GigabitEthernet0/0/0
    AVC Classification ID   : 0
    AVC Classification name: N/A
  Feature: OUTPUT_DROP
    Entry       : Output - 0x1145f484
    Input       : GigabitEthernet0/0/3
    Output      : GigabitEthernet0/0/0
    Lapsed time : 1500 ns
  Feature: IPV4_OUTPUT_INSPECT
    Entry       : Output - 0x1145fb8c
    Input       : GigabitEthernet0/0/3
    Output      : GigabitEthernet0/0/0
    Lapsed time : 1134760 ns
Packet Copy In
  0022bdf9 a993848a 8d48cd51 08004500 00820005 0000ff11
  1c2f0a01 fa320a7a 9089f7f6 0202006e 8b403c31 38373e39
  39373a20 41707220 31392030 313a3333
```

Join at
slido.com
#06 07 2023

True or False:

Packet Trace can only capture data-plane traffic.

Packet Trace
Catalyst 9000 Series

cisco Live!

Switch Capturing Tool Availability

RSPAN

FED Capture

Embedded Packet Capture (EPC)

Packet Trace

Show Platform Forward (SPF)

UADP 2.0/UADP 2.0 mini

Catalyst 9400

Catalyst 9300

Catalyst 9200

Catalyst 9500

Packet State Vector (PSV)

UADP 3.0

Catalyst 9600

Catalyst 9500H

Packet Trace

Catalyst 9000 Series
Show Platform Forward

Show Platform Forward
Catalyst 9000 Series UADP 2.0, 2.0 mini, 3.0

- **Available starting in 16.3.1**
- **User defined L2/L3/L4** packet data for a specific packet
- **Import** a specific packet **from a PCAP**
- **Switch generates 150-200 packets** to determine ingress and egress forwarding decisions
  NOTE: generated **packets do not leave the switch**
- Determine handling of the packet:
  - Ingress & Egress decisions
  - Forwarding Interfaces
  - Rewrite Type

Show Platform Forward

Show Platform Forward Syntax in 16.3.1 onwards

- Define the ingress port, vlan, etc.
- Provides the L2 Packet Data for the generated traffic
- Specify the L3 Protocol and Addressing for the traffic
- Configure L4 Protocol, Ports, and Optional Flags

```
CAT9300-1# show platform hardware fed switch <#> forward [interface|vlan|control-plane] <SMAC>
<DMAC> <L3 PROTOCOL> <L3 SRC> <L3 DEST> <L4 PROTOCOL> <L4 SRC> <L4 DEST> <OPT FLAGS>
```

```
CAT9300-1# show platform hardware fed switch active forward interface Te1/0/1 0cd0.f852.8042
c014.fe84.cc40 ipv4 192.168.1.37 8.8.8.8 udp 5193 53
```

Show Platform Forward

"Debug Platform Packet-Trace Feature Simulate" Syntax in 17.3.1 onwards

- Provides the L2 Packet Data for the generated traffic
- Specify the L3/L4 Protocol, Addressing, Ports, and Flags for the traffic
- Define the ingress port, vlan, etc.

```
mac access-list extended DNS-PACKET-L2
 permit host 0cd0.f852.8042 host c014.f384.cc40

CAT9300-1# debug platform condition feature simulate mac DNS-PACKET-L2

ip access-list extended DNS-PACKET
 10 permit udp host 192.168.1.37 eq 5193 host 8.8.8.8 eq domain

CAT9300-1# debug platform condition feature simulate ipv4 DNS-PACKET

CAT9300-1# debug platform condition feature simulate interface Ten1/0/1

CAT9300-1# debug platform condition start
CAT9300-1# debug platform packet-trace simulation start
```

Show Platform Forward

"Debug Platform Packet-Trace Feature Simulate" Syntax in 17.3.1 onwards

- Define the ingress port, vlan, etc.
- Provides the L2 Packet Data for the generated traffic
- Specify the L3 Protocol and Addressing for the traffic
- Configure L4 Protocol, Ports, and Optional Flags

```
CAT9300-1# show platform conditions

Conditional Debug Global State: Start
<SNIP>
Feature Condition        Type                        Value
-----------------------|--------------------------|-----------------------------------------
simulate                 ipv4 acl name               DNS-PACKET
simulate                 mac acl name                DNS-PACKET-L2
simulate                 interface name              TenGigabitEthernet1/0/1
```

Show Platform Forward

Verify the Results in 16.3.1 onward

After 2-5 minutes

%SHFWD-6-PACKET_TRACE_DONE: Switch 1 F0/0: fed: Packet Trace Complete: Execute (show platform hardware fed switch <> forward last summary|detail)
%SHFWD-6-PACKET_TRACE_FLOW_ID: Switch 1 F0/0: fed: Packet Trace Flow id is 65537

```
show platform hardware fed switch active forward last [summary|detail|flowid <#> (summary|detail)]
```

Input Packet

```
Input Packet Details:
###[ Ethernet ]###
  dst       = c0:14:fe:84:cc:40
  src       = 0c:d0:f8:52:80:42
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xa8f3
     src       = 192.168.1.37
     dst       = 8.8.8.8
     options   = ''
###[ UDP ]###
        sport     = 5193
        dport     = domain
        len       = 8
        chksum    = 0x1983
```

Ingress Details

```
Ingress:
   Port                : Ten1/0/1
   Global Port Number  : 1
   Local Port Number   : 1
   Asic Port Number    : 0
   Asic Instance       : 3
   Vlan                : 1
   Mapped Vlan ID      : 4
   STP Instance        : 2
   L3 Interface        : 37
    IPv4 Routing   : enabled
    IPv6 Routing   : enabled
    Vrf Id         : 0
   Adjacency:
    Station Index      : 180
    Destination Index  : 16402
    Rewrite Index      : 2
 Decision:
   Forwarding Mode    : 0 [Bridging]
   Replication Bit Map: ['localData']
   Winner : L2DESTMACVLAN LOOKUP
No exceptions occured.
CMI-2 Catch-all. Do not punt to CPU
```

Egress Details

```
Egress:
   Possible Replication :
       Port              : Ten1/0/2
       Port              : Ten1/0/3
   Output Port Data      :
       Port              : Ten1/0/2
    Global Port Number   : 1536
    Local Port Number    : 2
    Asic Port Number     : 1
    Asic Instance        : 3
    Unique RI            : 0
    Rewrite Type         : 1
                [L2_BRIDGE]
    Mapped Rewrite Type  : 4
                [L2_BRIDGE_INNER_IPv4]
    Vlan                 : 1
    Mapped Vlan ID       : 4
```

Output Packet

```
Output Packet Details:
    Port         : Ten1/0/2
###[ Ethernet ]###
  dst       = c0:14:fe:84:cc:40
  src       = 0c:d0:f8:52:80:42
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xa8f3
     src       = 192.168.1.37
     dst       = 8.8.8.8
     options   = ''
###[ UDP ]###
        sport     = 5193
        dport     = domain
        len       = 8
        chksum    = 0x1983
```

Show Platform Forward

Verify the Results in 17.3.1 onward

After 2-5 minutes

```
%SHFWD-6-PACKET_TRACE_DONE: Switch 1 F0/0: fed: Packet Trace Complete:
     Execute (show platform hardware fed switch <> forward last summary|detail)
%SHFWD-6-PACKET_TRACE_FLOW_ID: Switch 1 F0/0: fed: Packet Trace Flow id is 65537
```

show platform packet-trace simulation [summary|detail|status]

Input Packet

```
Input Packet Details:
###[ Ethernet ]###
  dst       = c0:14:fe:84:cc:40
  src       = 0c:d0:f8:52:80:42
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xa8f3
     src       = 192.168.1.37
     dst       = 8.8.8.8
     options   = ''
###[ UDP ]###
        sport     = 5193
        dport     = domain
        len       = 8
        chksum    = 0x1983
```

Ingress Details

```
Ingress:
   Port               : Ten1/0/1
   Global Port Number : 1
   Local Port Number  : 1
   Asic Port Number   : 0
   Asic Instance      : 3
   Vlan               : 1
   Mapped Vlan ID     : 4
   STP Instance       : 2
   L3 Interface       : 37
    IPv4 Routing   : enabled
    IPv6 Routing   : enabled
    Vrf Id         : 0
   Adjacency:
    Station Index      : 180
    Destination Index  : 16402
    Rewrite Index      : 2
 Decision:
   Forwarding Mode    : 0 [Bridging]
   Replication Bit Map: ['localData']
   Winner : L2DESTMACVLAN LOOKUP
No exceptions occured.
CMI-2 Catch-all. Do not punt to CPU
```

Egress Details

```
Egress:
 Possible Replication   :
       Port             : Ten1/0/1
       Port             : Ten1/0/2
       Port             : Ten1/0/3
       Port             : Ten1/0/4
Output Port Data      :
       Port             : Ten1/0/2
    Global Port Number  : 1536
    Local Port Number   : 2
    Asic Port Number    : 1
    Asic Instance       : 3
    Unique RI           : 0
    Rewrite Type        : 1
                [L2_BRIDGE]
    Mapped Rewrite Type : 4
                [L2_BRIDGE_INNER_IPv4]
    Vlan                : 1
    Mapped Vlan ID      : 4
```

Output Packet

```
Output Packet Details:
    Port         : Ten1/0/2
###[ Ethernet ]###
  dst       = c0:14:fe:84:cc:40
  src       = 0c:d0:f8:52:80:42
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xa8f3
     src       = 192.168.1.37
     dst       = 8.8.8.8
     options   = ''
###[ UDP ]###
        sport     = 5193
        dport     = domain
        len       = 8
        chksum    = 0x1983
```

Show Platform Forward

Verify the Results in 17.3.1 onward

After 2-5 min

```
show platform packet-trace simulation status
========================
Switch 1:
========================
Available Flows in Switch: 1
65537     Progress
```

```
show platform packet-trace simulation summary
```

Input Packet

```
Input Packet Details:
###[ Ethernet ]###
  dst       = c0:14:fe:84:cc:40
  src       = 0c:d0:f8:52:80:42
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xa8f3
     src       = 192.168.1.37
     dst       = 8.8.8.8
     options   = ''
###[ UDP ]###
        sport     = 5193
        dport     = domain
        len       = 8
        chksum    = 0x1983
```

Ingress Details

```
Ingress:
   Port               : Ten1/0/1
   Global Port Number : 1
   Local Port Number  : 1
   Asic Port Number   : 0
   Asic Instance      : 3
   Vlan               : 1
   Mapped Vlan ID     : 4
   STP Instance       : 2
   L3 Interface       : 37
    IPv4 Routing   : enabled
    IPv6 Routing   : enabled
    Vrf Id         : 0
   Adjacency:
    Station Index      : 180
    Destination Index  : 16402
    Rewrite Index      : 2
 Decision:
   Forwarding Mode    : 0 [Bridging]
   Replication Bit Map: ['localData']
   Winner : L2DESTMACVLAN LOOKUP
No exceptions occured.
CMI-2 Catch-all. Do not punt to CPU
```

Egress Details

```
Egress:
 Possible Replication   :
       Port             : Ten1/0/1
       Port             : Ten1/0/2
       Port             : Ten1/0/3
       Port             : Ten1/0/4
Output Port Data      :
       Port             : Ten1/0/2
    Global Port Number  : 1536
    Local Port Number   : 2
    Asic Port Number    : 1
    Asic Instance       : 3
    Unique RI           : 0
    Rewrite Type        : 1
                [L2_BRIDGE]
    Mapped Rewrite Type : 4
                [L2_BRIDGE_INNER_IPv4]
    Vlan                : 1
    Mapped Vlan ID      : 4
```

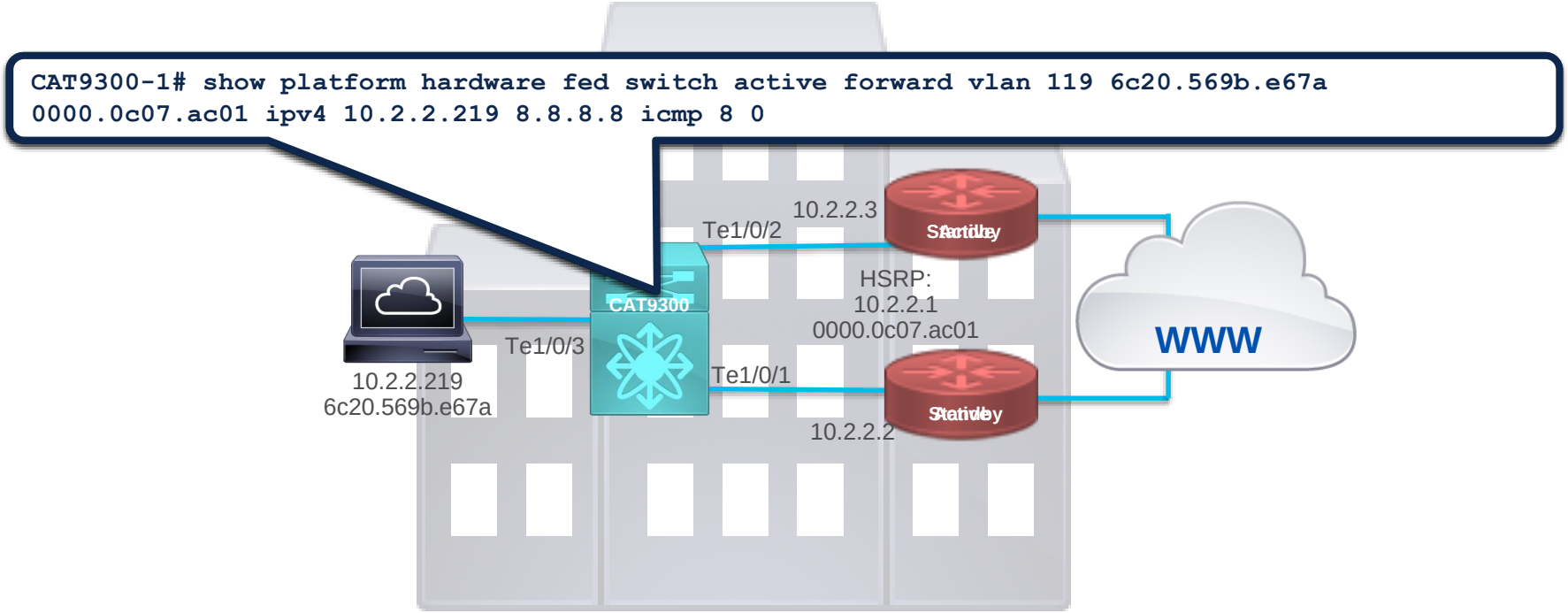
Output Packet

```
Output Packet Details:
    Port         : Ten1/0/2
###[ Ethernet ]###
  dst       = c0:14:fe:84:cc:40
  src       = 0c:d0:f8:52:80:42
  type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 28
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0xa8f3
     src       = 192.168.1.37
     dst       = 8.8.8.8
     options   = ''
###[ UDP ]###
        sport     = 5193
        dport     = domain
        len       = 8
        chksum    = 0x1983
```

Real World Example

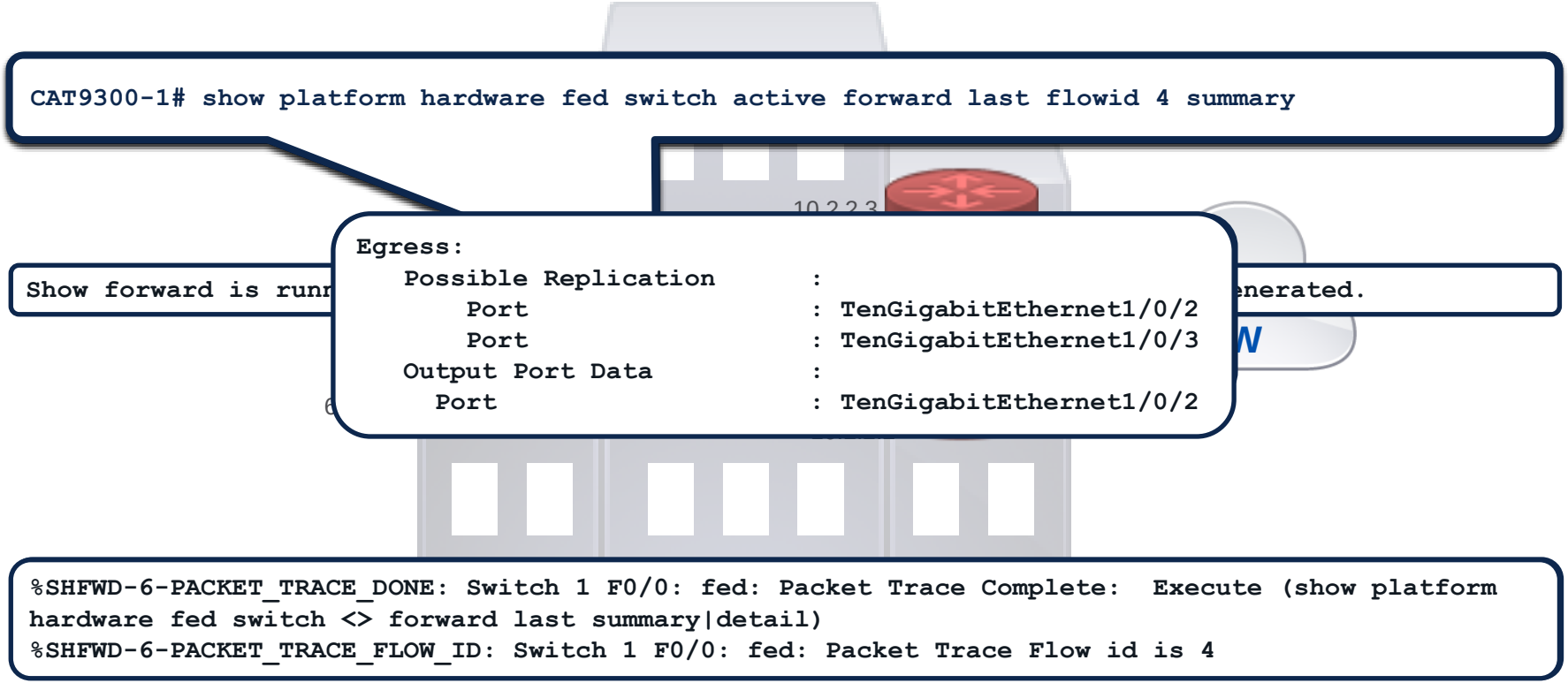
HSRP Forwarding Concerns – 16.3.1 Syntax

```
CAT9300-1# show platform hardware fed switch active forward vlan 119 6c20.569b.e67a
0000.0c07.ac01 ipv4 10.2.2.219 8.8.8.8 icmp 8 0
```



Real World Example

HSRP Forwarding Concerns – 16.3.1 Syntax



```
CAT9300-1# show platform hardware fed switch active forward last flowid 4 summary
```

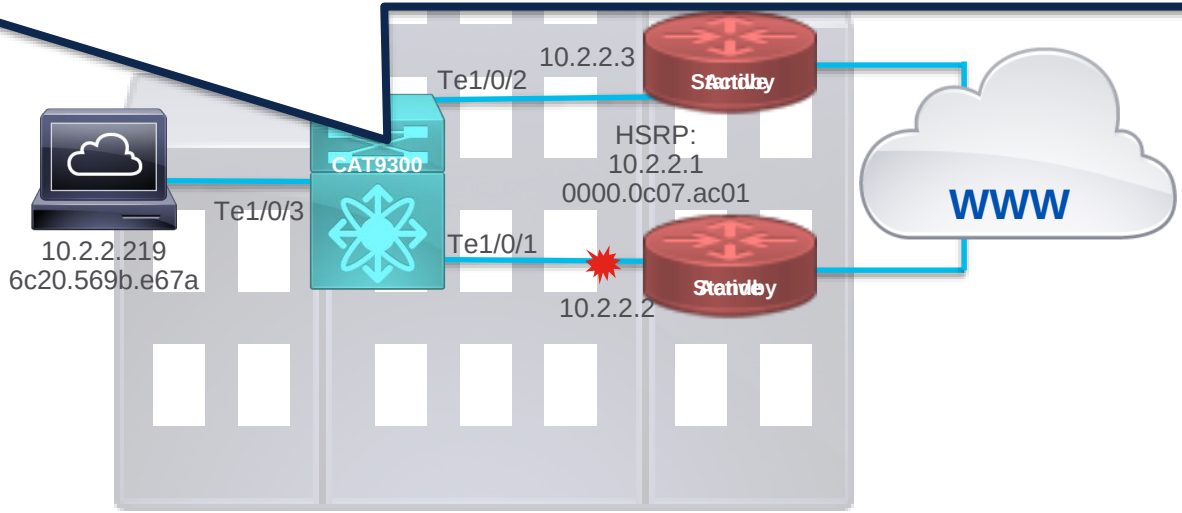
```
Egress:
    Possible Replication       :
        Port                   : TenGigabitEthernet1/0/2
        Port                   : TenGigabitEthernet1/0/3
    Output Port Data           :
      Port                     : TenGigabitEthernet1/0/2
```

```
%SHFWD-6-PACKET_TRACE_DONE: Switch 1 F0/0: fed: Packet Trace Complete:  Execute (show platform
hardware fed switch <> forward last summary|detail)
%SHFWD-6-PACKET_TRACE_FLOW_ID: Switch 1 F0/0: fed: Packet Trace Flow id is 4
```

Real World Example

HSRP Forwarding Concerns

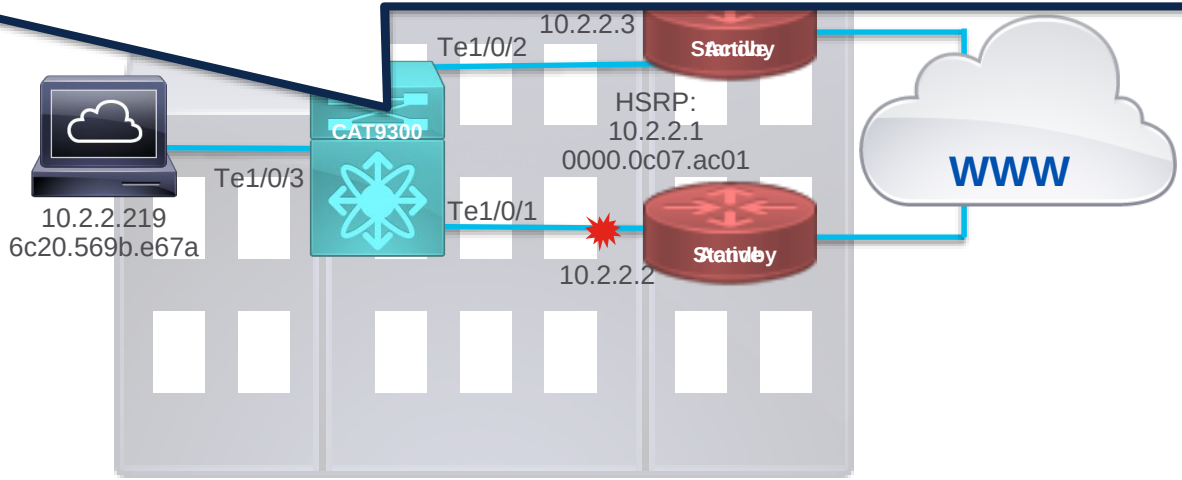
```
CAT9300-1# config t
CAT9300-1(config)# mac access-list extended HSRP-L2
CAT9300-1(config)# permit host 6c20.569b.e67a host 0000.0c07.ac01
CAT9300-1(config)# ip access-list extended HSRP-L3
CAT9300-1(config)# permit icmp host 10.2.2.219 host 8.8.8.8 echo
```



Real World Example

HSRP Forwarding Concerns

```
CAT9300-1# debug platform condition feature simulate interface T1/0/3
CAT9300-1# debug platform condition feature simulate mac HSRP-L2
CAT9300-1# debug platform condition feature simulate ipv4 HSRP-L3

CAT9300-1# debug platform condition start
CAT9300-1# debug platform packet-trace simulation start
```



Real World Example

HSRP Forwarding Concerns

```
CAT9300-1# debug platform condition feature simulate mac HSRP-L2
CAT9300-1# debug platform condition feature simulate ipv4 HSRP-L3
```

```
CAT9300-1# debug platform packet-trace simulation start
```

```
CAT9300-1# debug p
CAT9300-1# debug plat
```

```
CAT9300-1# show platform pac
<SNIP>
Egress:
   Possible Replication
       Port
Output Port Data          :
     Port                 :
```

```
CAT9300-1# show platform packet-trace simulation status
=========================
Switch 1:
=========================
Available Flows in Switch: 1
100728840    Complete
100728841    Complete
=========================
```

```
CAT9300-1# show platform packet-trace simulation status
=========================
Switch 1:
=========================
Available Flows in Switch: 1
100728840    Complete
=========================
```

Packet Trace

Catalyst 9500H & 9600
Packet State Vector

Packet Trace
Catalyst 9500H & 9600- UADP 3.0

- **UADP 3.0** feature that is only available on 9600 & 9500H
- "Packet State Vector" first introduced in 16.8.1 **(covered in LABTRS-2391)**
- New **"Packet Trace" syntax starting in 17.3.1**
- **Live capture** of a single packet
- **Visibility into the ASIC level** forwarding details for the captured packet

Catalyst 9600

Catalyst 9500H

Packet Trace

UADP 3.0

Catalyst 9600

Catalyst 9500H

Define Trigger & Start Capture

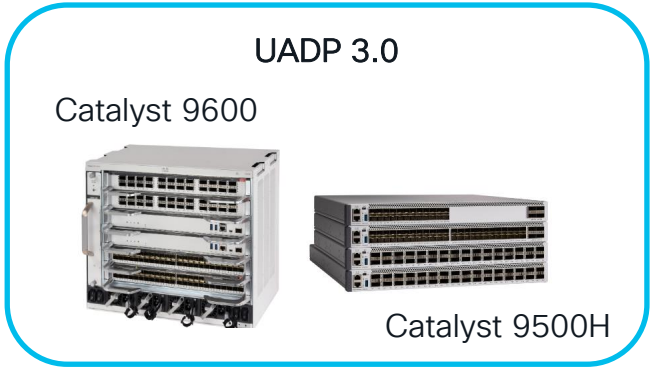
Review Configuration/Status & Clear

View Packet Forwarding Decision

```
debug platform condition interface <#> match ipv4 protocol udp host 192.168.1.37 host 8.8.8.8 eq 53 ingress
debug platform condition start
debug platform packet-trace start
```

```
show platform conditions
show platform packet-trace status
clear platform condition all
clear platform packet-trace configuration
```

Packet Trace



Define Trigger & Start Capture → Review Configuration/Status & Clear → View Packet Forwarding Decision

```
show platform packet-trace summary                 <--- PSV Summary results
show platform packet-trace packet all              <--- Packet details
show platform packet-trace detailed ingress        <--- PSV FIR (Forwarding Ingress Resolution)
show platform packet-trace detailed egress         <--- PSV FER (Forwarding Egress Resolution)
```

Packet Trace

Packet State Vector Configuration Syntax in 16.8.1 onward

Define the Trigger & Start Capture (lots of options, use ? to explore)

```
debug platform hardware fed switch active capture trigger ipv4 192.168.29.132 192.168.32.2 icmp
debug platform hardware fed switch active capture trigger interface twentyFiveGigE2/5/0/5 ingress
debug platform hardware fed switch active capture start
```

Review Configuration/Status & Clear

```
show platform hardware fed switch active capture trigger
show platform hardware fed switch active capture status
clear platform hardware fed switch active capture trigger
```

View Packet Forwarding Decision

```
show platform hardware fed switch active capture summary
show platform hardware fed switch active capture packet
show platform hardware fed switch active capture detailed [ingressFc|egressFc]
```

Packet Trace

Configuration Syntax in 17.3.1 onward

Define the Trigger & Start Capture

```
debug platform condition interface <#> match ipv4 protocol udp host 192.168.1.37 host 8.8.8.8 eq 53 ingress
debug platform condition start
debug platform packet-trace start
```

Review Configuration/Status & Clear

```
show platform conditions
show platform packet-trace status
clear platform condition all
clear platform packet-trace configuration
```

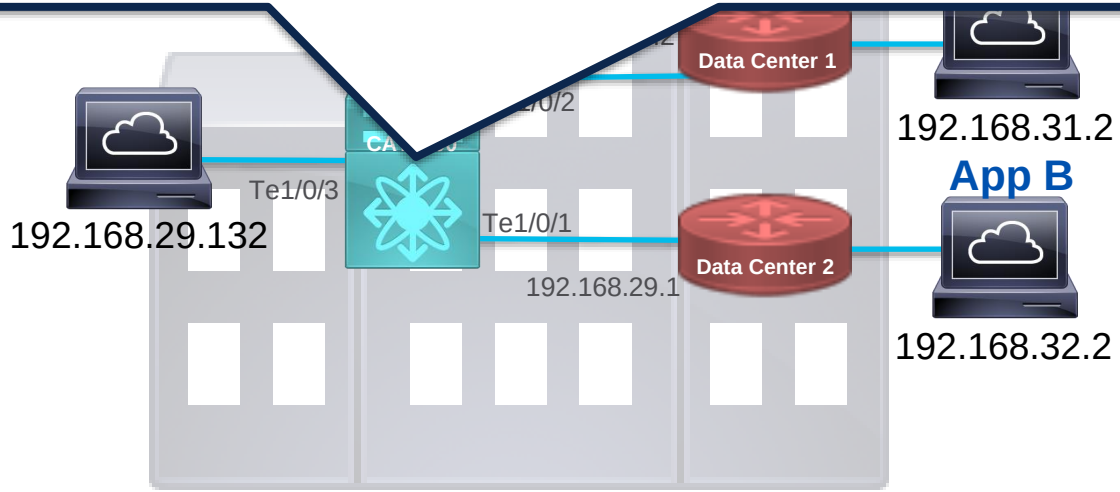
View Packet Forwarding Decision

```
show platform packet-trace summary              <--- PSV Summary results
show platform packet-trace packet all           <--- Packet details
show platform packet-trace detailed ingress     <--- PSV FIR (Forwarding Ingress Resolution)
show platform packet-trace detailed egress      <--- PSV FER (Forwarding Egress Resolution)
```

Packet State Vector Demo

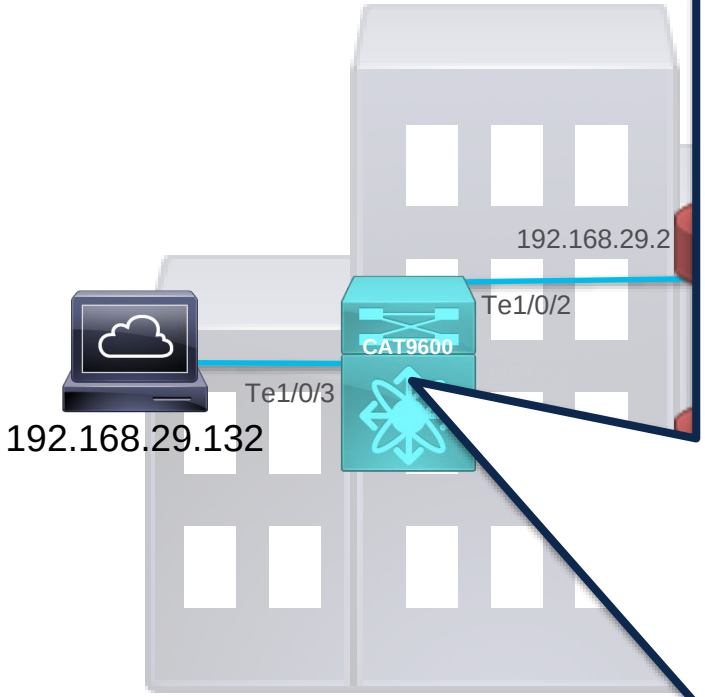
Application Traffic Forwarding Decision

```
CAT9600-1# debug platform condition interface Ten1/0/3 match ipv4 host 192.168.29.132 host 192.168.32.2 ingress
CAT9600-1# debug platform condition start
CAT9600-1# debug platform packet-trace start
```



Packet State Vector D

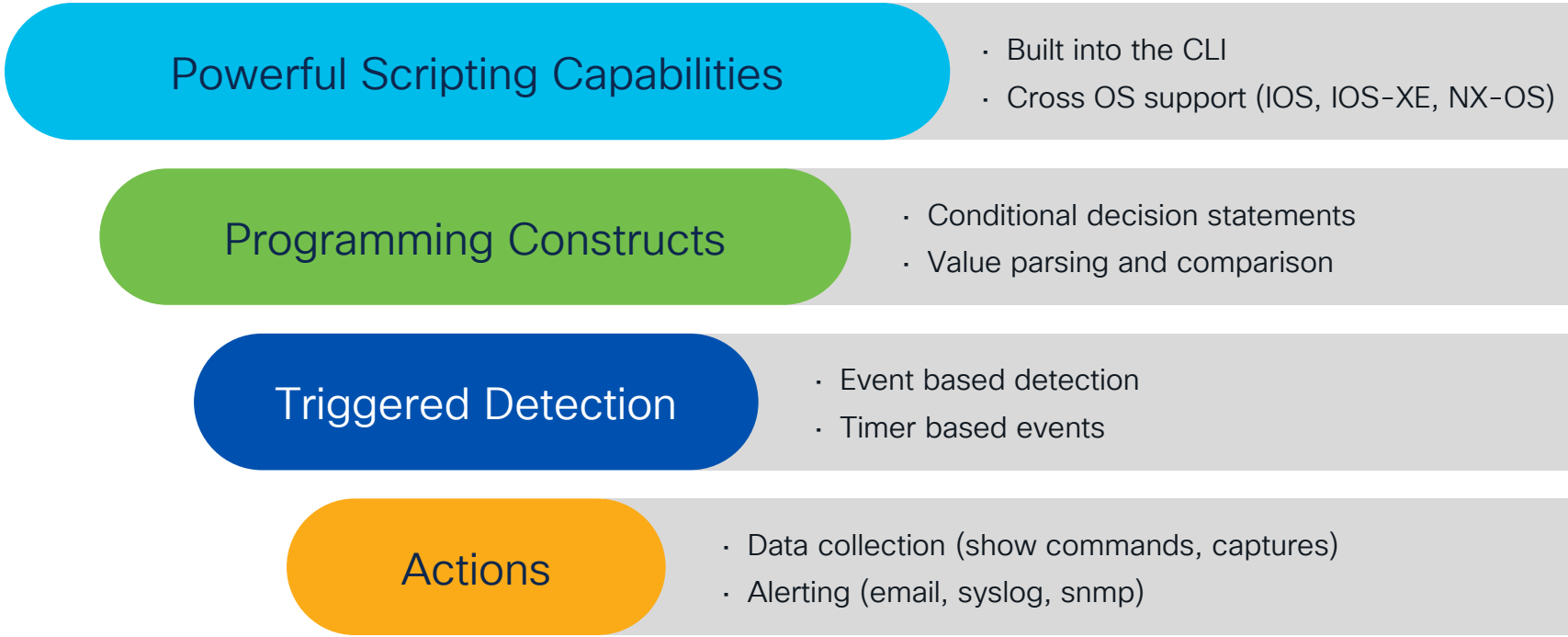
Application Traffic Forwarding D




```
CAT9600-1# show platform packet-trace packet all
<SNIP>
Trigger: Ingress Interface:TenGigabitEthernet1/0/3
      IP:192.168.32.2 Src IP:192.168.29.132
SUMMARY:
 Input      : Interface Te1/0/3
 Output     : Interface Te1/0/1
 Ingress Port: Te1/0/3
  GPN: 965 ASIC: 0 CORE: 0 SLICE: 1 ContextId: 4
 Ingress Packet:
  DMAC: cc:db:93:a2:18:70 SMAC: 6c:20:56:9b:e6:79 ETYPE:
       0x0800 LEN: 118
  SRC_IP: 192.168.29.132 DST_IP: 192.168.32.2 ToS: 0 PROTO:
       1 TTL: 255 Payload Length: 100
  ICMP TYPE: 0 CODE: 8 SEQ: 0
 Classification & Look up:
  Ingress:
  Replication Bitmap:
   LocalData : 1 LocalCpu  : 0 LocalDpu  : 0
   CoreData  : 0 CoreCpu   : 0 CoreDpu   : 0
   RemoteData: 0 RemoteCpu : 0 RemoteDpu : 0
 Egress Port: Te1/0/1
  GPN: 1344 ASIC: 0 CORE: 0 SLICE: 0 ContextId: 5
 Egress Packet:
  DMAC: cc:db:93:a2:18:70 SMAC: 6c:20:56:9b:e6:79 ETYPE:
       0x0800 LEN: 122
  SRC_IP: 192.168.29.132 DST_IP: 192.168.32.2 ToS: 0 PROTO:
       1 TTL: 255 Payload Length: 100
  ICMP TYPE: 0 CODE: 8 SEQ: 0
 Classification & Look up:
  Egress:
  Replication Bitmap:
   LocalData : 0 LocalCpu  : 0 LocalDpu  : 0
   CoreData  : 0 CoreCpu   : 0 CoreDpu   : 0
   RemoteData: 0 RemoteCpu : 0 RemoteDpu : 0
```

Event Triggered Captures

Embedded Event Manager (EEM)



Event Triggered Captures

Workflow

Match the event → Set the actions → Review Data

Syslog, timer, counter, OID

CLI, email, SNMP trap

PCAP, Appended outputs

```
*Jan 10 19:13:31.512: %DUAL-5-NBRCHANGE: EIGRP-IPv4 10: Neighbor 10.10.20.3 (GigabitEthernet0/0/2) is
down: holding time expired
```

Router A

Router B

EIGRP dropped! Trigger EEM script to run!

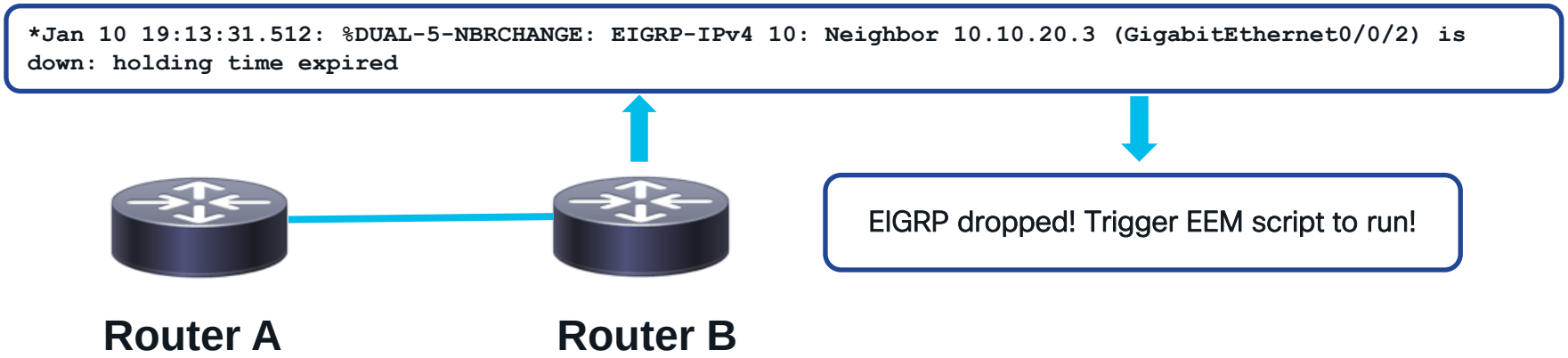
Stop capture and append commands to file!

Embedded Event Manager (EEM)

- EEM is a **powerful scripting** tool built-in to IOS, IOS-XE, and NX-OS that allows for **scripting detectable events** and actions and responding with other actions, including data collection, e-mail generation, and other automation.
- Provides programming constructs, including **conditional statements**, that allow for **intelligent event detection** and response.
- Useful in capturing **events that may be transient** or otherwise hard to detect or troubleshoot due to the limited timing of the event
- Reference the configuration guides and command references for more details on all options for actions and events:
  - http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/eem/configuration/15-mt/eem-15-mt-book.html
  - http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/eem/command/eem-cr-book.html

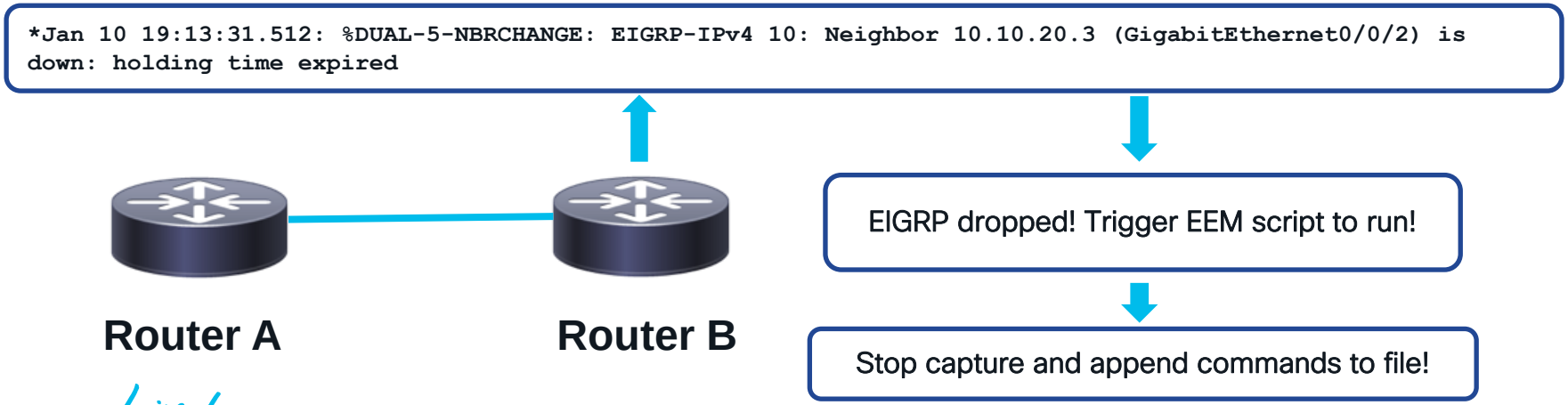
Matching the Event

- Each script is configured with an event which triggers it to run
- Often a **time-based event** (i.e. run every “x” seconds or run at a given time of day using CRON) or **syslog-based** (i.e. message generated in the logs is detected)
- It is important that the event be carefully considered. If the wrong message is defined or the time-based event doesn’t run often enough, the nature of the event may dictate the EEM won’t catch it in time to obtain relevant outputs.



Setting the Actions

- **Actions** are defined after the event and **run once the script is triggered**.
- The most common actions are **CLI commands** or output **string comparisons** based on outputs collected previously in the script. You can also **log a message** stating that a condition was detected.
- Output can be **appended to a file** for review so that data is stored in the event the user cannot troubleshoot or catch the issue in real-time.



Scripting Captures Example

- Before the script can run, careful preparation and consideration should be made to ensure the script triggers correctly and runs the appropriate commands to capture relevant information.

```
monitor capture EIGRP-CAP match ipv4 any any control-plane in buffer size 10 circular
monitor capture EIGRP-CAP start
```

Setup and start the capture for all IPv4 traffic ingressing the control plane. Circular buffer used for continuous capture.

```
event manager applet EIGRP-FLAP
 event syslog pattern "holding time expired"
 action 1.0 cli command "enable"
 action 1.1 syslog msg "EIGRP Flap Detected! Running EEM Script!"
 action 1.2 cli command "monitor capture EIGRP-CAP stop"
 action 2.1 cli command "show ip eigrp neighbor | append bootflash:EIGRP_flap.txt"
 action 2.2 cli command "show proc cpu sorted | append bootflash:EIGRP_flap.
 action 2.3 cli command "show proc cpu history | append bootflash:EIGRP_flap.tx
 ...
 action 3.1 cli command "monitor capture EIGRP-CAP export bootflash:EIGRP-CAP.pcap
```

Logging message pattern that will be matched to trigger the script.

Make sure **enable** is defined before any other CLI command action!

Stop the capture once the flap occurs.

Router A

Router B

Note: If using AAA command authorization

```
event manager session cli username <NAME>
```
OR
```
event manager applet EIGRP-FLAP authorization bypass
```

Scripting Captures Example (Continued)

- Once the event is detected, EEM will log a message stating that it has been triggered and will run the commands defined.

```
Router_B#
*Jan 16 19:27:59.672: %DUAL-5-NBRCHANGE: EIGRP-IPv4 10: Neighbor 10.10.20.3 (GigabitEthernet0/0/2) is down:
holding time expired
*Jan 16 19:27:59.686: %HA_EM-6-LOG: EIGRP-FLAP: EIGRP Flap Detected! Running EEM Script!
```


```
SCRIPT IS RUNNING...
```

The user-defined message in the EEM script is generated alerting that the script is running.

Files have been created and can be copied off for external review.

```
Router_B# show bootflash: | in EIGRP
! PCAP FILE FROM THE CAPTURE
135          24 Jan 16 2023 19:27:59 +00:00 /bootflash/EIGRP-CAP.pcap
! FILE WITH CAPTURED COMMANDS
136       56257 Jan 16 2023 19:28:00 +00:00 /bootflash/EIGRP_flap.txt
```

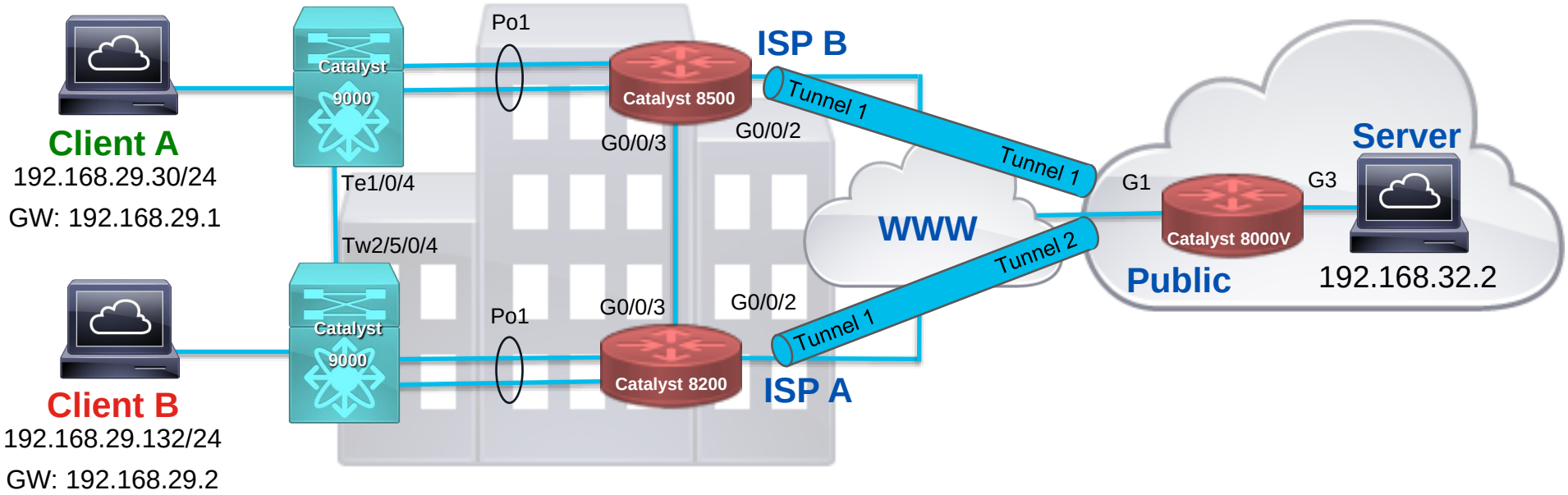
Join at
slido.com
#06 07 2023

What set of tools will you use the most in your network?

Putting it All Together

Real World Example

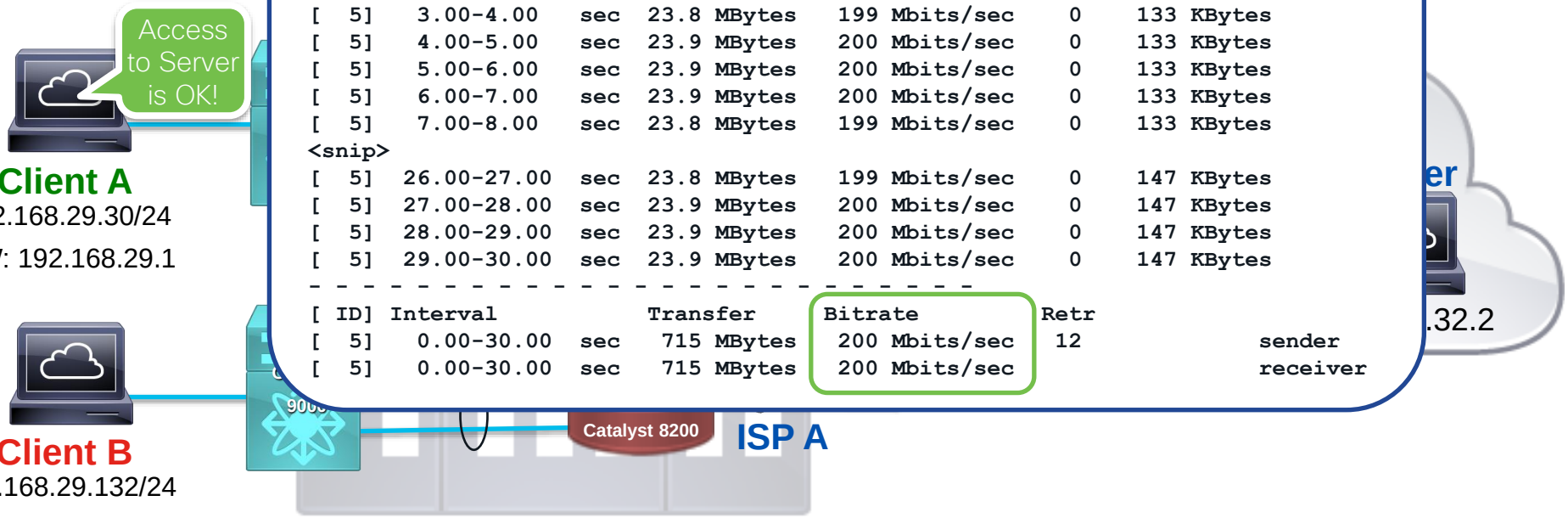
Degraded Performance



Real Wo

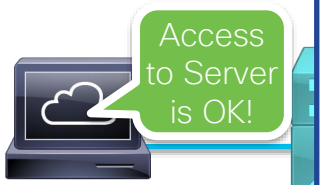
Degraded Per

```
cisco@linux1:~$ iperf3 -b 200M -c 192.168.32.2 -t 30
Connecting to host 192.168.32.2, port 5201
[  5] local 192.168.29.30 port 58538 connected to 192.168.32.2 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec  23.9 MBytes   200 Mbits/sec   12    133 KBytes
[  5]   1.00-2.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   2.00-3.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   3.00-4.00   sec  23.8 MBytes   199 Mbits/sec    0    133 KBytes
[  5]   4.00-5.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   5.00-6.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   6.00-7.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   7.00-8.00   sec  23.8 MBytes   199 Mbits/sec    0    133 KBytes
<snip>
[  5]  26.00-27.00  sec  23.8 MBytes   199 Mbits/sec    0    147 KBytes
[  5]  27.00-28.00  sec  23.9 MBytes   200 Mbits/sec    0    147 KBytes
[  5]  28.00-29.00  sec  23.9 MBytes   200 Mbits/sec    0    147 KBytes
[  5]  29.00-30.00  sec  23.9 MBytes   200 Mbits/sec    0    147 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-30.00  sec   715 MBytes   200 Mbits/sec   12             sender
[  5]   0.00-30.00  sec   715 MBytes   200 Mbits/sec                  receiver
```

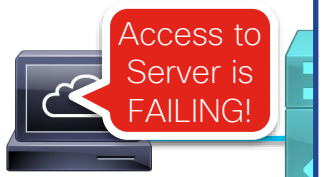


Real World Example

Degraded Perf



Client A
192.168.29.30/24
GW: 192.168.29.1

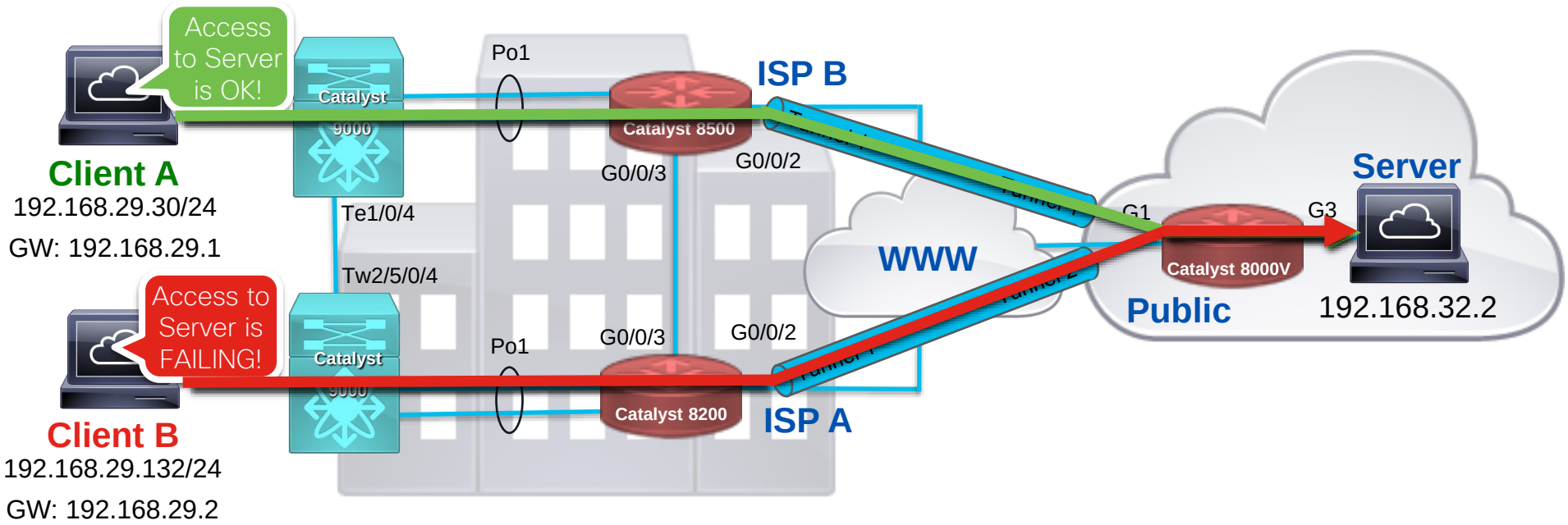


Client B
192.168.29.132/24
GW: 192.168.29.2

```
cisco@linux2:~$ iperf3 -b 200M -c 192.168.32.2 -t 30
Connecting to host 192.168.32.2, port 5201
[  5] local 192.168.29.132 port 39190 connected to 192.168.32.2 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec  5.57 MBytes  46.7 Mbits/sec    7   2.19 MBytes
[  5]   1.00-2.00   sec  12.8 MBytes   107 Mbits/sec  385   1.50 MBytes
[  5]   2.00-3.00   sec  15.2 MBytes   128 Mbits/sec    0   1.59 MBytes
[  5]   3.00-4.00   sec  16.1 MBytes   135 Mbits/sec    0   1.66 MBytes
[  5]   4.00-5.00   sec  16.8 MBytes   141 Mbits/sec    0   1.70 MBytes
[  5]   5.00-6.00   sec  16.4 MBytes   137 Mbits/sec    0   1.73 MBytes
[  5]   6.00-7.00   sec  16.8 MBytes   141 Mbits/sec    0   1.75 MBytes
[  5]   7.00-8.00   sec  17.1 MBytes   144 Mbits/sec    0   1.76 MBytes
[  5]   8.00-9.00   sec  17.2 MBytes   145 Mbits/sec   19   1.61 MBytes
[  5]   9.00-10.00  sec  6.00 MBytes  50.3 Mbits/sec  1843    622 KBytes
[  5]  10.00-11.00  sec  6.25 MBytes  52.4 Mbits/sec    0    675 KBytes
[  5]  11.00-12.00  sec  6.75 MBytes  56.6 Mbits/sec    0    712 KBytes
[  5]  12.00-13.00  sec  6.88 MBytes  57.7 Mbits/sec    0    734 KBytes
[  5]  13.00-14.00  sec  7.12 MBytes  59.8 Mbits/sec    0    746 KBytes
[  5]  14.00-15.00  sec  7.25 MBytes  60.8 Mbits/sec    0    750 KBytes
<snip>
[  5]  26.00-27.00  sec  12.6 MBytes   106 Mbits/sec    0   1.40 MBytes
[  5]  27.00-28.00  sec  14.9 MBytes   125 Mbits/sec    0   1.61 MBytes
[  5]  28.00-29.00  sec  6.12 MBytes  51.4 Mbits/sec  1123    585 KBytes
[  5]  29.00-30.00  sec  6.00 MBytes  50.3 Mbits/sec    0    632 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-30.00  sec   305 MBytes  85.2 Mbits/sec  3377             sender
[  5]   0.00-30.10  sec   303 MBytes  84.4 Mbits/sec                   receiver
```

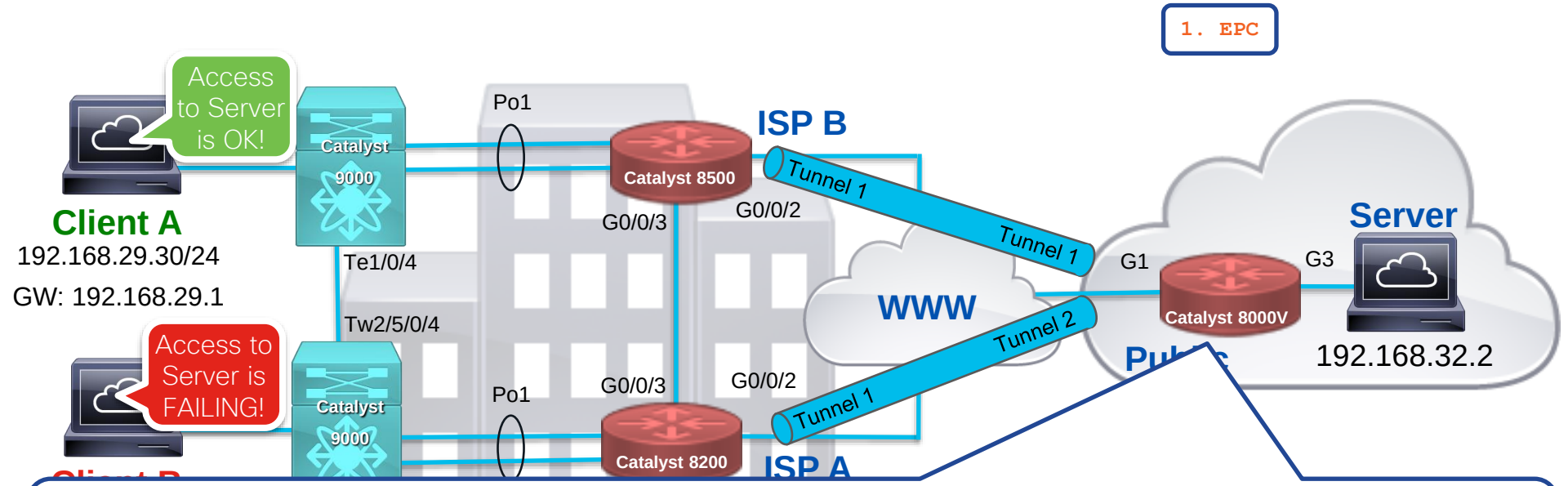
Real World Example

Degraded Performance



Real World Example

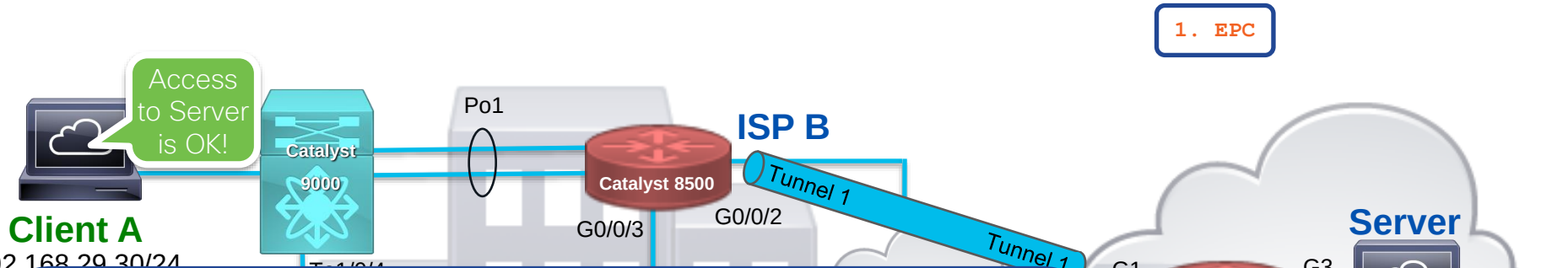
Degraded Performance




```
c8000v# monitor capture clienta buffer size 100 match ipv4 host 192.168.29.30 host 192.168.32.2 interface gi3 both
c8000v# monitor capture clienta start
c8000v# monitor capture clientb buffer size 100 match ipv4 host 192.168.29.132 host 192.168.32.2 interface gi3 both
c8000v# monitor capture clienta start
```

Real World Example

Degraded Performance




```
c8000v# monitor capture clienta stop
Stopped capture point : clienta
c8000v# show monitor capture clienta buffer brief
 ------------------------------------------------------------------------------
 #   size   timestamp     source              destination      dscp    protocol
 ------------------------------------------------------------------------------
   0   74    0.000000    192.168.29.30     ->  192.168.32.2     0  BE   TCP
   1   66    0.001007    192.168.29.30     ->  192.168.32.2     0  BE   TCP
   2  103    0.001007    192.168.29.30     ->  192.168.32.2     0  BE   TCP
<snip>
c8000v# monitor capture clienta export bootflash:ClientA-c8000v-30sec-200MB.pcap
Exported Successfully
```

Real W

Degraded P

Access
to Server

ClientA-c8000v-30sec-200MB.pcap

tcp.stream eq 1

Expression...

No.	Time	Source	Destination	Protocol	Length	Info
52	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [PSH, ACK] Seq=54404 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
53	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=55798 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
54	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=57192 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
55	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=58586 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
56	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=59980 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
57	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=61374 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
58	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=62768 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
59	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=64162 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006
60	15:59:04…	192.168.29.30	192.168.32.2	TCP	1460	58542 → 5201 [ACK] Seq=65556 Ack=1 Win=64768 Len=1394 TSval=1402995439 TSecr=1637464006

ClientB-c8000v-30sec-200MB.pcap

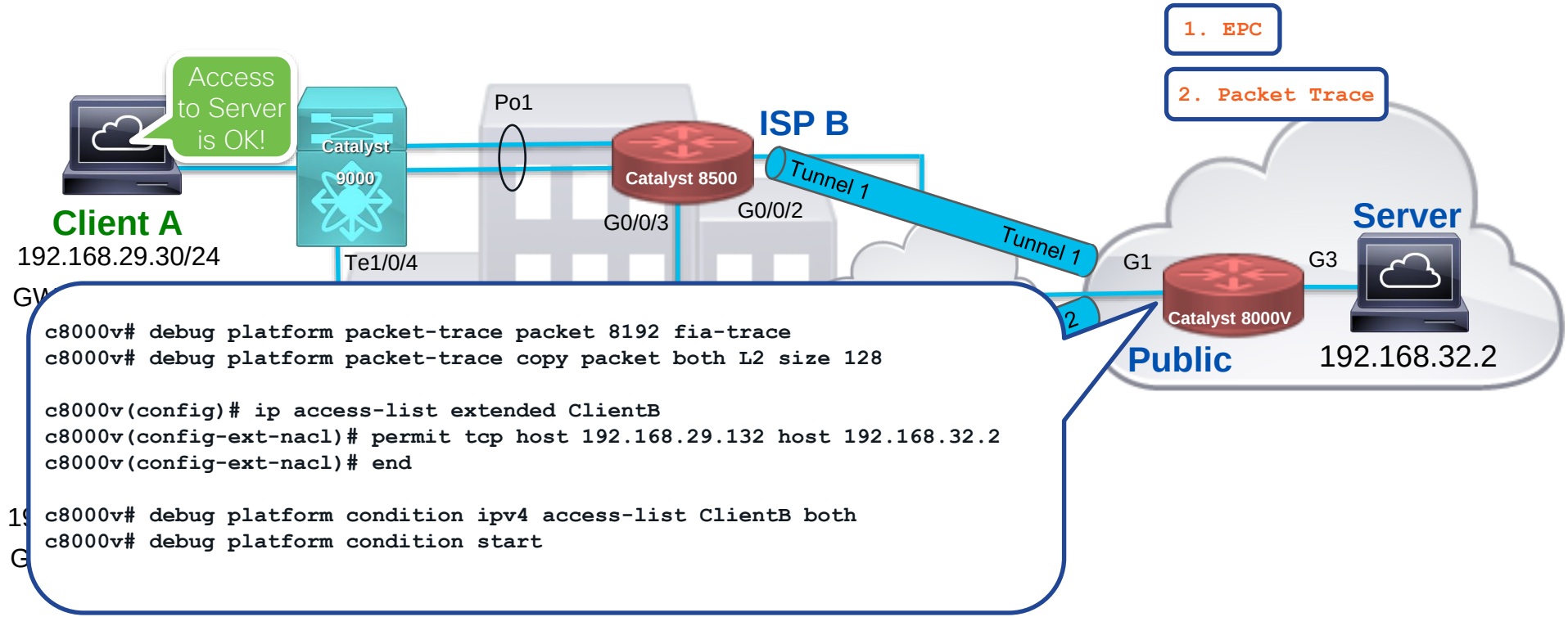
tcp.stream eq 1

Expression...

No.	Time	Source	Destination	Protocol	Length	Info
1535	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Previous segment not captured] 39190 → 5201 [ACK] Seq=5237296 Ack=1 Win=64768 Len=1394 T
1536	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Previous segment not captured] 39190 → 5201 [ACK] Seq=5350210 Ack=1 Win=64768 Len=1394 T
1537	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Previous segment not captured] 39190 → 5201 [ACK] Seq=5461776 Ack=1 Win=64768 Len=1394 T
1538	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Previous segment not captured] 39190 → 5201 [ACK] Seq=5576084 Ack=1 Win=64768 Len=1394 T
1539	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Previous segment not captured] 39190 → 5201 [ACK] Seq=5687604 Ack=1 Win=64768 Len=1394 T
1540	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Previous segment not captured] 39190 → 5201 [ACK] Seq=5806094 Ack=1 Win=64768 Len=1394 T
1541	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Out-Of-Order] 39190 → 5201 [ACK] Seq=3058474 Ack=1 Win=64768 Len=1394 TSval=651312714 TS
1542	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Out-Of-Order] 39190 → 5201 [PSH, ACK] Seq=3059868 Ack=1 Win=64768 Len=1394 TSval=6513127
1543	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Out-Of-Order] 39190 → 5201 [ACK] Seq=3061262 Ack=1 Win=64768 Len=1394 TSval=651312714 TS
1544	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Out-Of-Order] 39190 → 5201 [ACK] Seq=3062656 Ack=1 Win=64768 Len=1394 TSval=651312714 TS
1545	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Out-Of-Order] 39190 → 5201 [ACK] Seq=3064050 Ack=1 Win=64768 Len=1394 TSval=651312714 TS
1546	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Out-Of-Order] 39190 → 5201 [ACK] Seq=3065444 Ack=1 Win=64768 Len=1394 TSval=651312714 TS
1547	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Out-Of-Order] 39190 → 5201 [PSH, ACK] Seq=3066838 Ack=1 Win=64768 Len=1394 TSval=6513127
1548	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3068232 Ack=1 Win=64768 Len=1394 TSval=651312880
1549	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3069626 Ack=1 Win=64768 Len=1394 TSval=651312880
1550	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3071020 Ack=1 Win=64768 Len=1394 TSval=651312880
1551	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3072414 Ack=1 Win=64768 Len=1394 TSval=651312880
1552	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3073808 Ack=1 Win=64768 Len=1394 TSval=651312880
1553	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3075202 Ack=1 Win=64768 Len=1394 TSval=651312880
1554	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3076596 Ack=1 Win=64768 Len=1394 TSval=651312880
1555	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3077990 Ack=1 Win=64768 Len=1394 TSval=651312880
1556	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3079384 Ack=1 Win=64768 Len=1394 TSval=651312880
1557	16:01:37…	192.168.29.132	192.168.32.2	TCP	1460	[TCP Retransmission] 39190 → 5201 [ACK] Seq=3080778 Ack=1 Win=64768 Len=1394 TSval=651312880

Real World Example

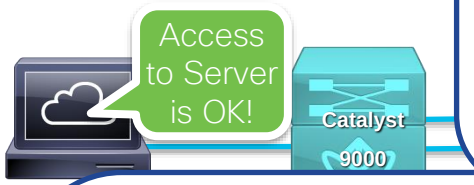
Degraded Performance



Real World E

Degraded Performan

```
c8000v#show platform packet-trace statistics
Packets Summary
  Matched   279984
  Traced    8192
Packets Received
  Ingress   279984
  Inject    0
Packets Processed
  Forward   279984
  Punt      0
  Drop      0
  Consume   0
```



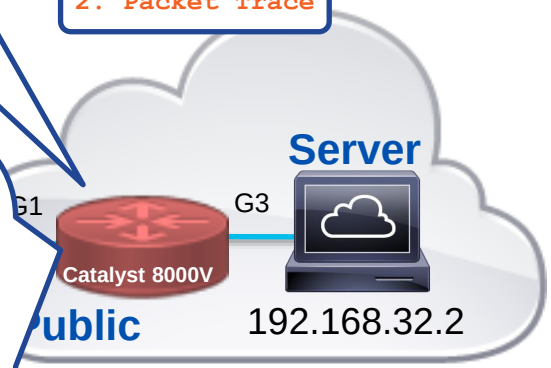
Cli
192.1
GW:

C
192.1
GW:

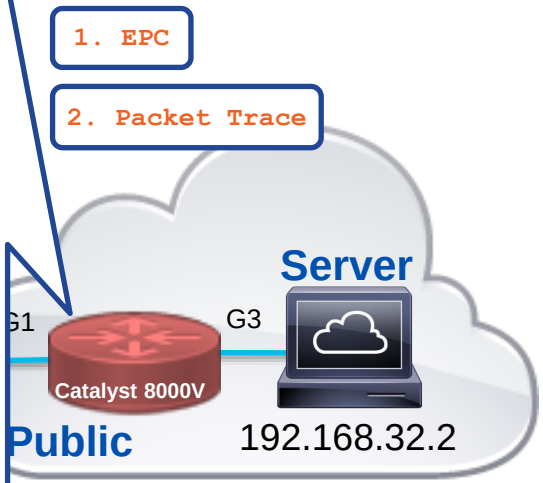
```
c8000v#show platform packet-trace summary
Pkt    Input                Output               State  Reason
0      Tu2                  Gi3                  FWD
1      Tu2                  Gi3                  FWD
2      Tu2                  Gi3                  FWD
3      Tu2                  Gi3                  FWD
4      Tu2                  Gi3                  FWD
5      Tu2                  Gi3                  FWD
6      Tu2                  Gi3                  FWD
7      Tu2                  Gi3                  FWD
8      Tu1                  Gi3                  FWD
9      Tu1                  Gi3                  FWD
10     Tu1                  Gi3                  FWD
11     Tu1                  Gi3                  FWD
12     Tu1                  Gi3                  FWD
13     Tu1                  Gi3                  FWD
14     Tu1                  Gi3                  FWD
```

1. EPC

2. Packet Trace

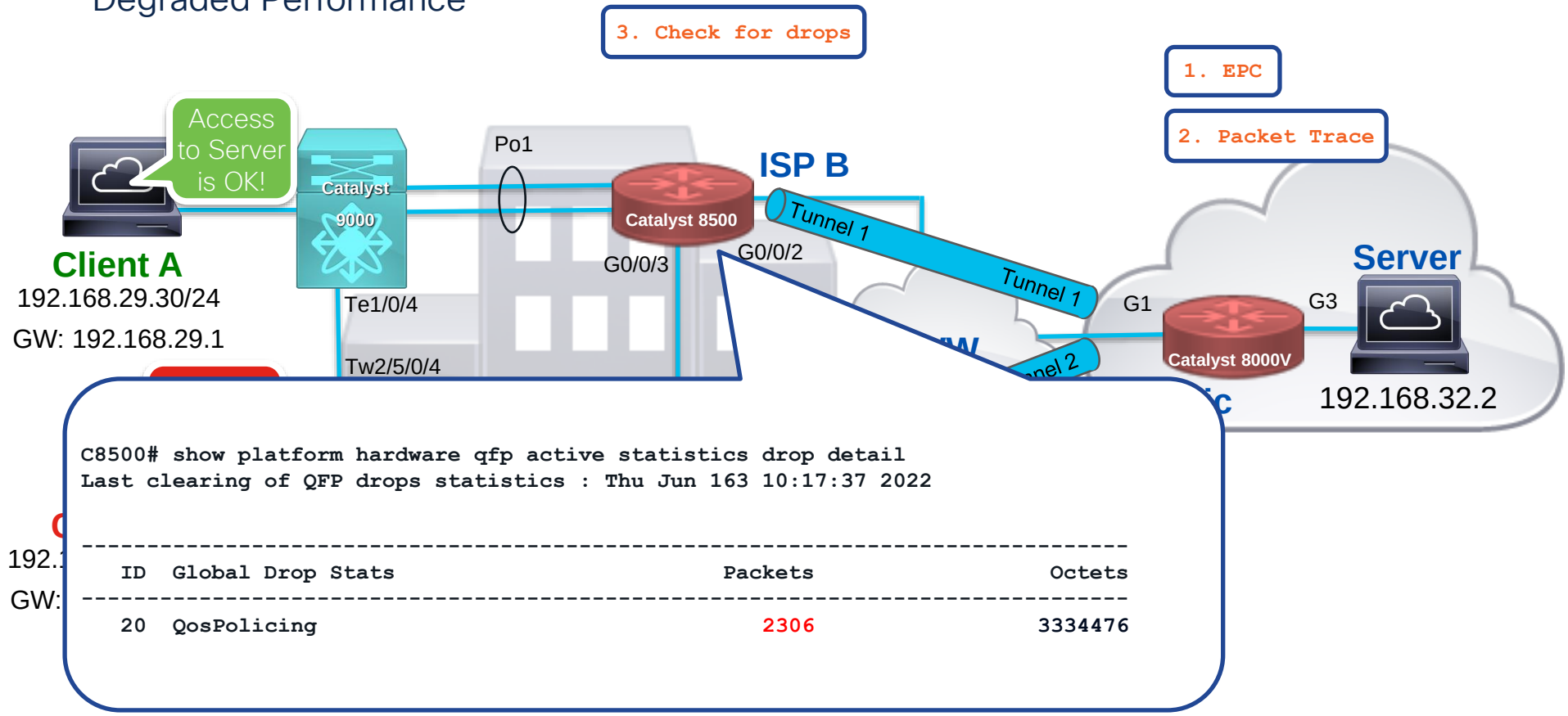


```
c8000v#show platform packet-trace packet 8
Packet: 8            CBUG ID: 452579
Summary
  Input      : Tunnel1
  Output     : GigabitEthernet3
  State      : FWD
  Timestamp
    Start    : 744635101329389 ns (04/13/2023 16:27:49.562588 UTC)
    Stop     : 744635101347589 ns (04/13/2023 16:27:49.562606 UTC)
Path Trace
  Feature: IPV4(Input)
    Input        : Tunnel1
    Output       : <unknown>
    Source       : 192.168.29.132
    Destination  : 192.168.32.2
    Protocol     : 6 (TCP)
      SrcPort    : 39202
      DstPort    : 5201
<snip>
 Feature: DEBUG_COND_OUTPUT_PKT
    Entry        : Output - 0x8143e18c
    Input        : Tunnel1
    Output       : GigabitEthernet3
    Lapsed time  : 155 ns
  Feature: MARMOT_SPA_D_TRANSMIT_PKT
    Entry        : Output - 0x81481154
    Input        : Tunnel1
    Output       : GigabitEthernet3
    Lapsed time  : 4847 ns
```



Real World Example

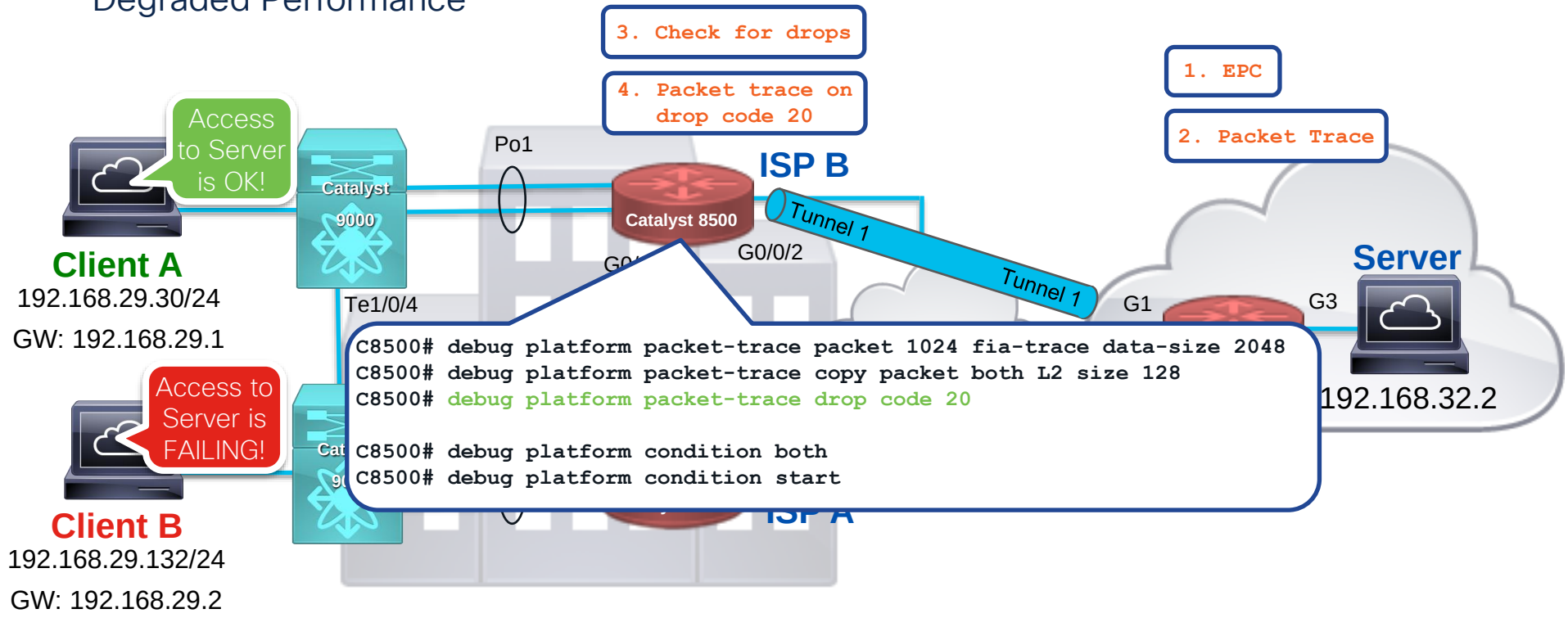
Degraded Performance




```
C8500# show platform hardware qfp active statistics drop detail
Last clearing of QFP drops statistics : Thu Jun 163 10:17:37 2022

-------------------------------------------------------------------------------
   ID  Global Drop Stats                                 Packets              Octets
-------------------------------------------------------------------------------
   20  QosPolicing                                        2306              3334476
```

Real World Example

Degraded Performance



```
C8500# show platform packet-trace packet 0
Packet: 0            CBUG ID: 20486
Summary
  Input     : GigabitEthernet0/0/3
  Output    : Tunnel1
  State     : DROP 20  (QosPolicing)
  Timestamp
    Start   : 876257676515945 ns (04/13/202
    Stop    : 876257676522704 ns (04/13/20
Path Trace
  Feature: IPV4(Input)
    Input       : GigabitEth rnet0/0/3
    Output      : <unknown>
    Source      : 192.168.29 132
    Destination : 192.168.32 2
    Protocol    : 6 (TCP)
      SrcPort   : 39226
      DstPort   : 5201
<snip>
```

```
<snip>
  Feature: QOS
    Direction
    Action
    Drop Cause
    Policy name
    Class name
Feature: OUTPUT_FNF_DROP_SDWAN
  Entry        : Output - 0x814f4700
  Input        : GigabitEthernet0/0/3
  Output       : Tunnel1
  Lapsed time  : 2406 ns
Feature: OUTPUT_DROP
  Entry        : Output - 0x814e1798
  Input        : GigabitEthernet0/0/3
  Output       : Tunnel1
  Lapsed time  : 738 ns
Feature: IPV4_OUTPUT_QOS
  Entry        : Output - 0x8151e720
  Input        : GigabitEthernet0/0/3
  Output       : Tunnel1
  Lapsed time  : 24400 ns
```

```
interface Tunnel1
 ip address 192.168.30.1 255.255.255.0
 tunnel source GigabitEthernet0/0/2
 tunnel mode ipsec ipv4
 tunnel destination 14.2.56.124
 tunnel protection ipsec profile PROFILE_IPSEC
 service-policy output POLICELOWER
end
!
policy-map POLICELOWER
 class LOWER
  police cir 50000000
   conform-action transmit
   exceed-action drop
```

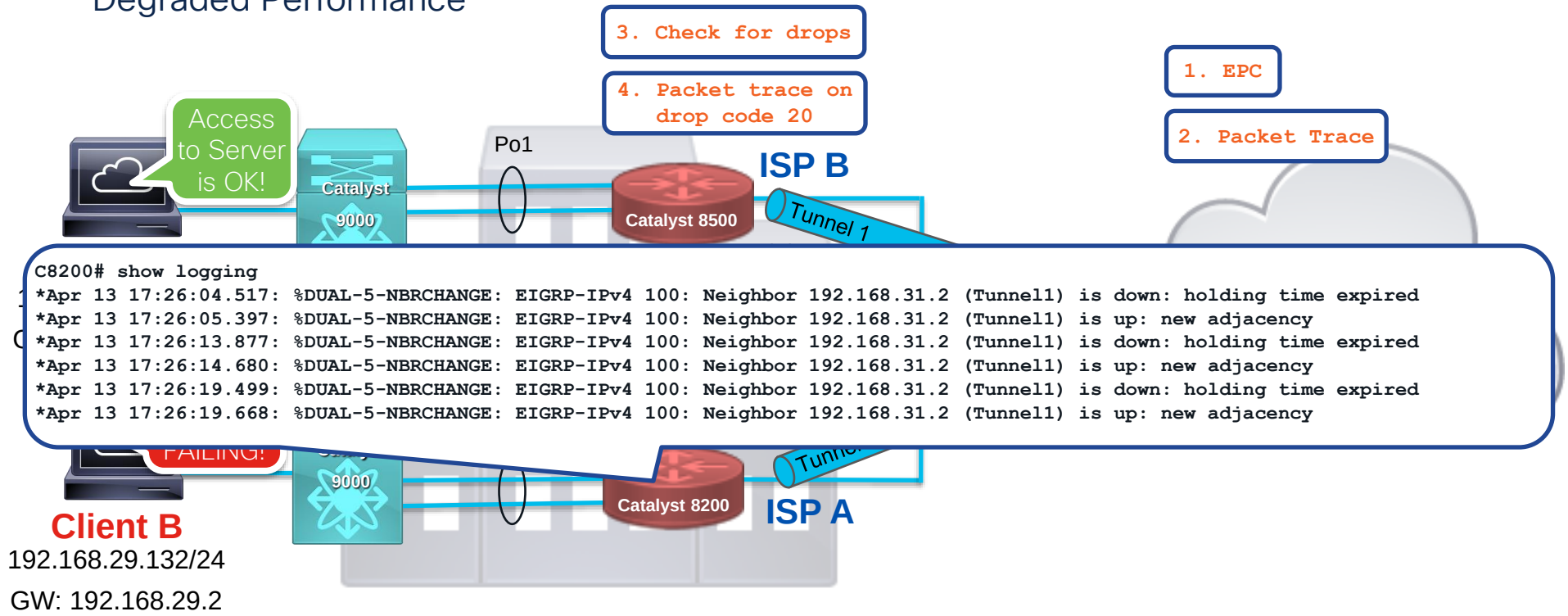
Client B

192.168.29.132/24

GW: 192.168.29.2

Real World Example

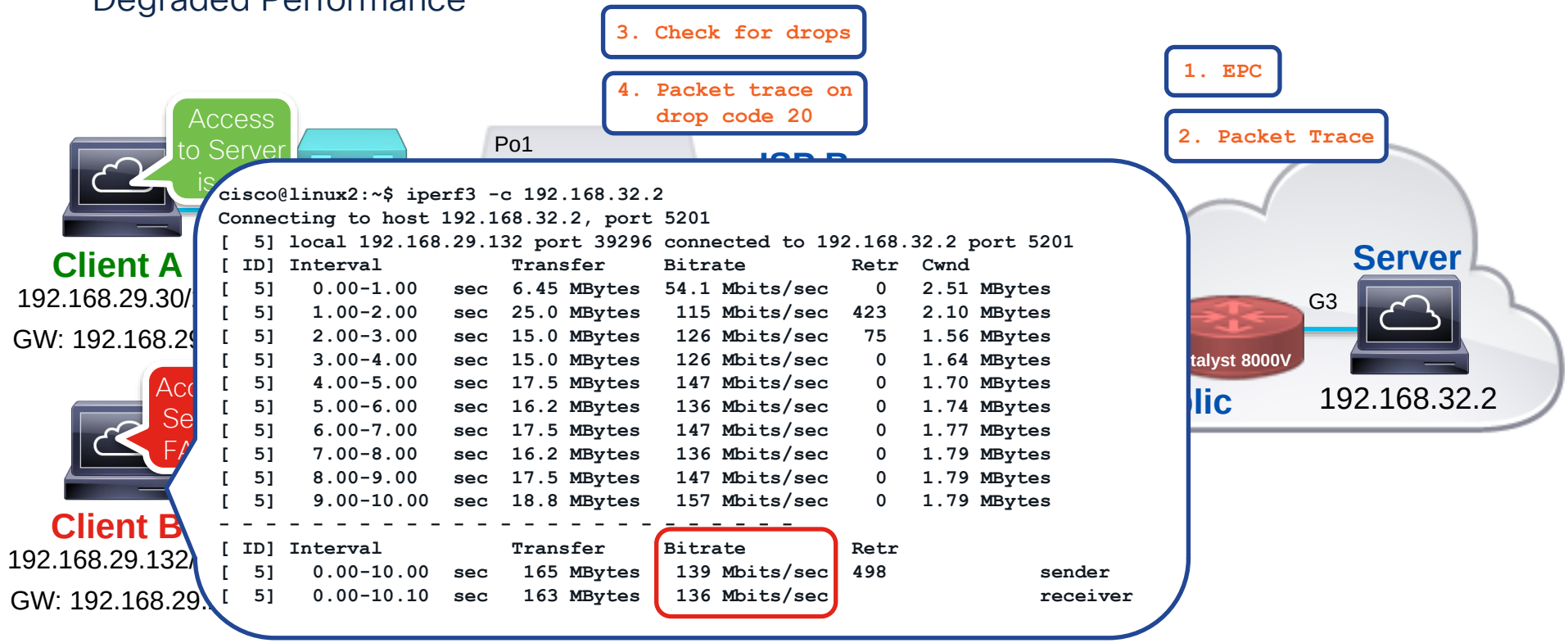
Degraded Performance



Survey Results

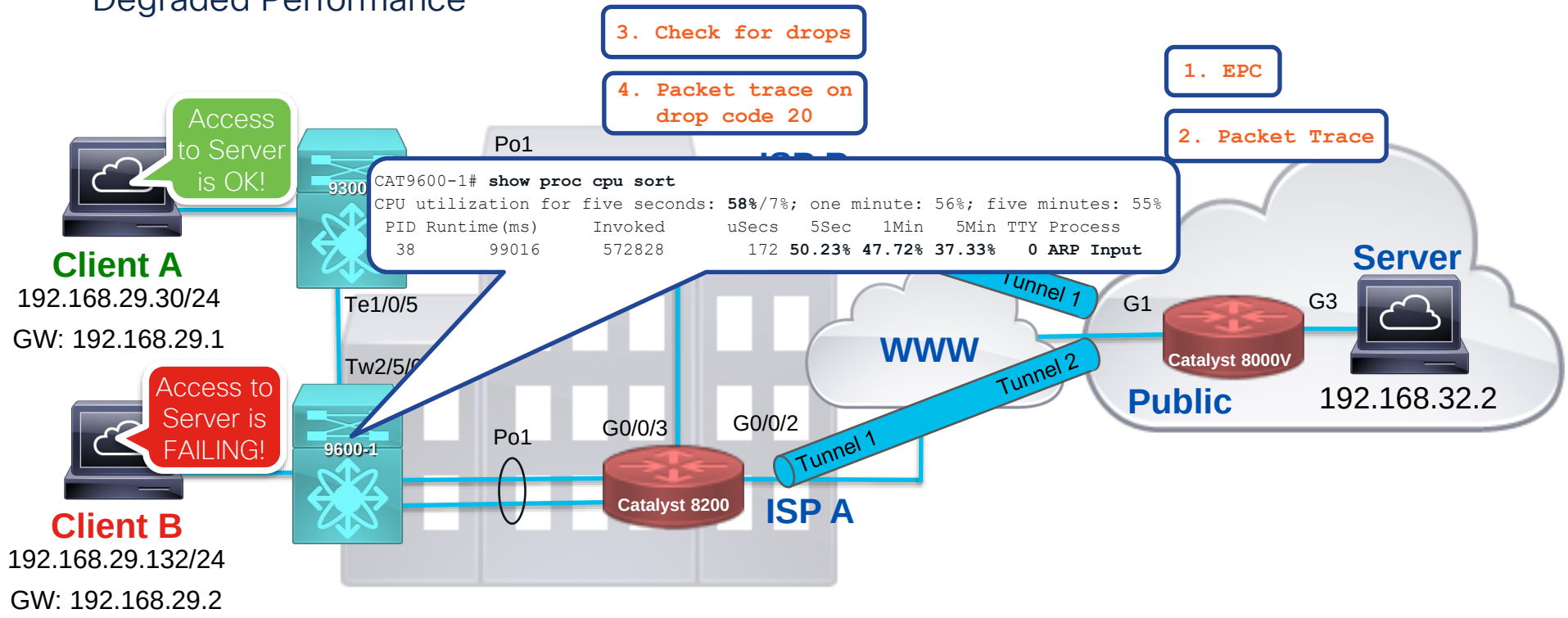
Real World Example

Degraded Performance



Real World Example

Degraded Performance



Real World

Degraded Perform

Client A
192.168.29.30/24
GW: 192.168.29.1

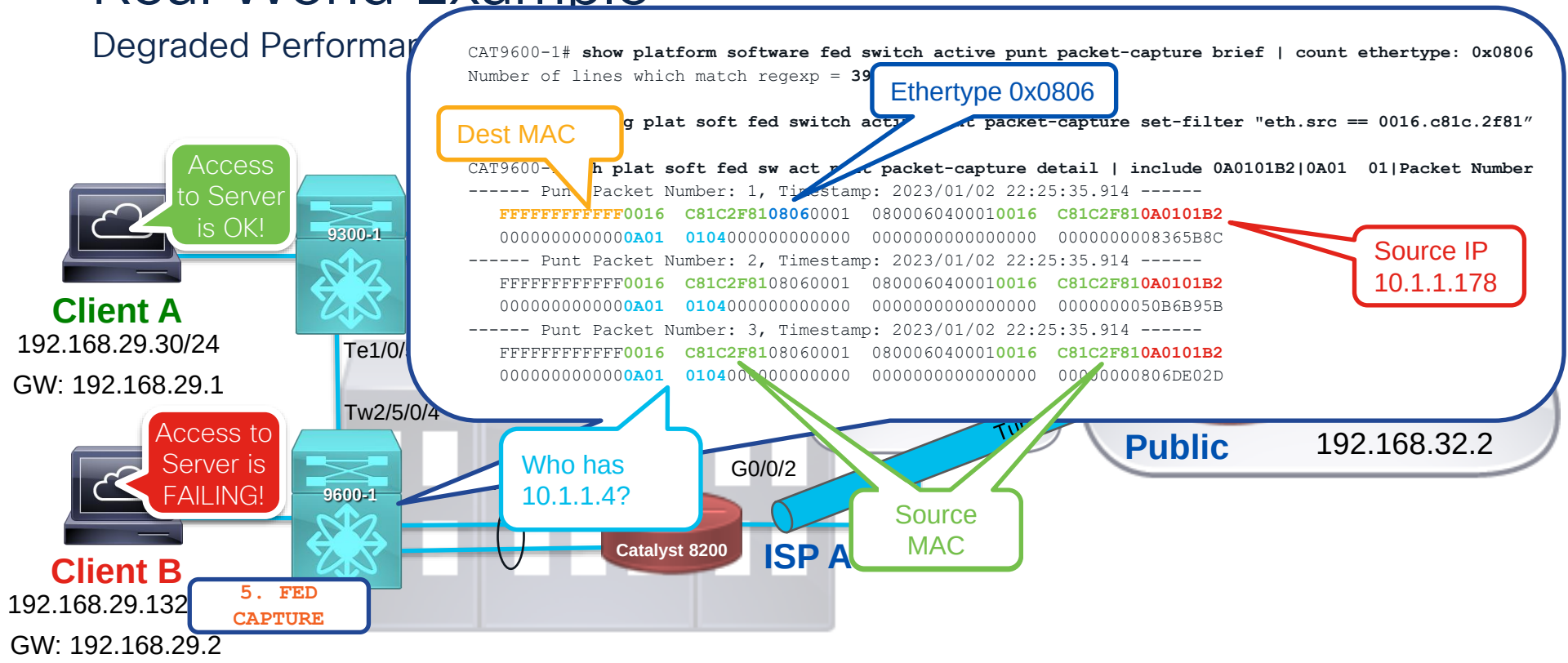
Client B
192.168.29.132
GW: 192.168.29.2

```
CAT9600-1# debug platform software fed switch active punt packet-capture start
Punt packet capturing started.

CAT9600-1# show platform software fed switch active punt packet-capture status
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 4096 packets. Capture capacity : 4096 packets

CAT9600-1# show platform software fed switch active punt packet-capture brief
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 4096 packets. Capture capacity : 4096 packets

------ Punt Packet Number: 1, Timestamp: 2023/01/02 22:02:32.660 ------
 interface : physical: GigabitEthernet1/0/2[if-id: 0x0000000a], pal: Vlan500 [if-id: 0x00000042]
 metadata  : cause: 7 [ARP request or response], sub-cause: 1, q-no: 5, linktype: MCP_LINK_TYPE_IP [1]
 ether hdr : dest mac: ffff.ffff.ffff, src mac: 0016.c81c.2f81
 ether hdr : ethertype: 0x0806 (ARP)

CAT9600-1# show platform software fed switch active punt packet-capture cpu-top-talker summary
Punt packet capturing: disabled. Buffer wrapping: disabled
Total captured so far: 4096 packets. Capture capacity : 4096 packets

L2 Top Talkers:
3937    Source mac      00:16:c8:1c:2f:81
3946    Dest mac        ff:ff:ff:ff:ff:ff
3937    Vlan    500

L3 Top Talkers:
3937    Source IPv4     10.1.1.178
15      Dest IPv4       10.1.1.4
131     TTL     255

L4 Top Talkers:
104     Protocol Num    (TCP)
104     L4 Source Port  5427
104     L4 Dest Port    23

Internal Top Talkers:
3937    Interface       Vlan500
3946    CPU Queue       ARP request or response
```

NOTE:
packet-capture cpu-top-talker summary
is available in 17.6.1 onwards

Real World Example

Degraded Performar



Real Wo

Degraded Per

```
cisco@linux2:~$ ping -c 100 192.168.29.2
PING 192.168.29.2 (192.168.29.2): 56 data bytes
64 bytes from 192.168.29.2: icmp_seq=0 ttl=239 time=33.818 ms
64 bytes from 192.168.29.2: icmp_seq=1 ttl=239 time=27.150 ms
64 bytes from 192.168.29.2: icmp_seq=2 ttl=239 time=26.890 ms
<snip>
64 bytes from 192.168.29.2: icmp_seq=97 ttl=239 time=26.538 ms
64 bytes from 192.168.29.2: icmp_seq=98 ttl=239 time=27.251 ms
64 bytes from 192.168.29.2: icmp_seq=99 ttl=239 time=26.676 ms

--- 192.168.29.2 ping statistics ---
100 packets transmitted, 100 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 25.932/26.959/33.818/0.864 ms
cisco@linux2:~$
```

Access to Server is OK!

Client A

192.168.29.30/24

GW: 192.168.29.1

```
cisco@linux2:~$ ping -c 100 192.168.32.2
PING 192.168.32.2 (192.168.32.2): 56 data bytes
64 bytes from 192.168.32.2: icmp_seq=0 ttl=239 time=33.354 ms
64 bytes from 192.168.32.2: icmp_seq=1 ttl=239 time=26.603 ms
Request timeout for icmp_seq 2
Request timeout for icmp_seq 3
64 bytes from 192.168.32.2: icmp_seq=4 ttl=239 time=26.551 ms
<snip>
64 bytes from 192.168.32.2: icmp_seq=96 ttl=239 time=26.710 ms
Request timeout for icmp_seq 97
Request timeout for icmp_seq 98
64 bytes from 192.168.32.2: icmp_seq=99 ttl=239 time=26.520 ms

--- 192.168.32.2 ping statistics ---
100 packets transmitted, 80 packets received, 20.0% packet loss
round-trip min/avg/max/stddev = 25.621/26.990/33.354/0.883 ms
cisco@linux2:~$
```

Access to Server is FAILING!

Client B

192.168.29.132

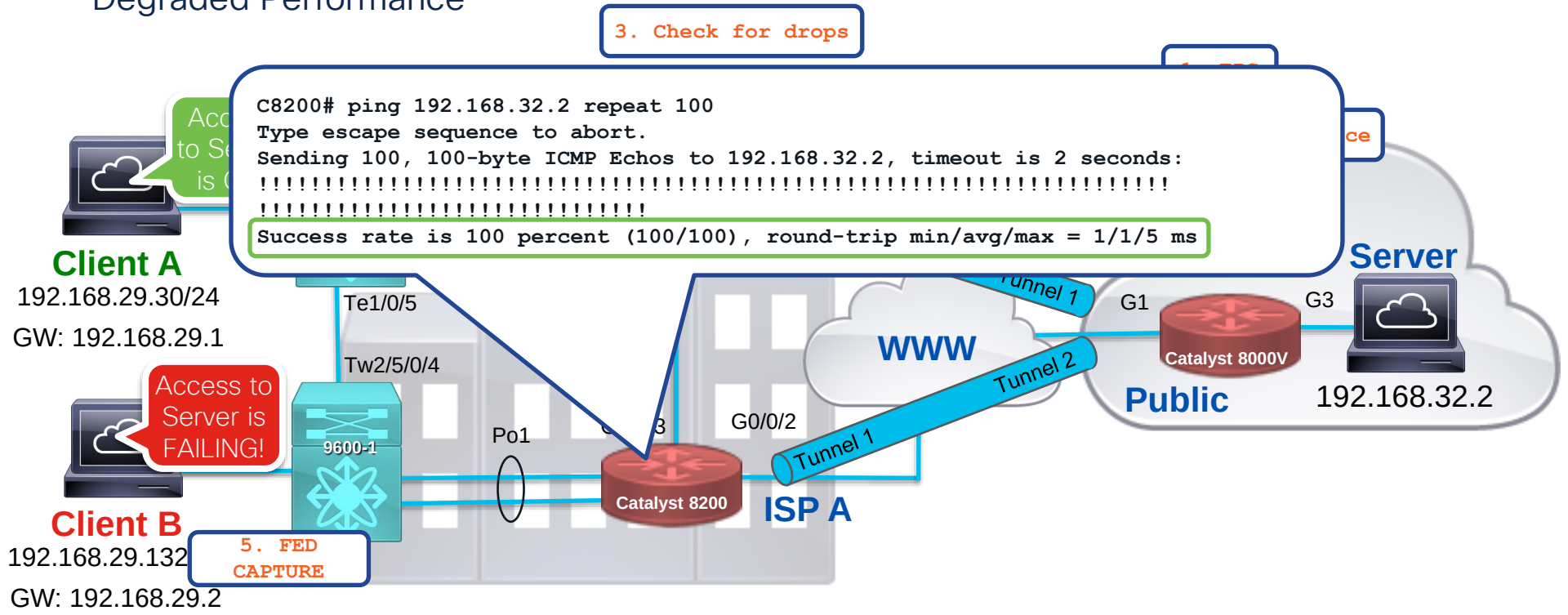
GW: 192.168.29.2

Server

.32.2

Real World Example

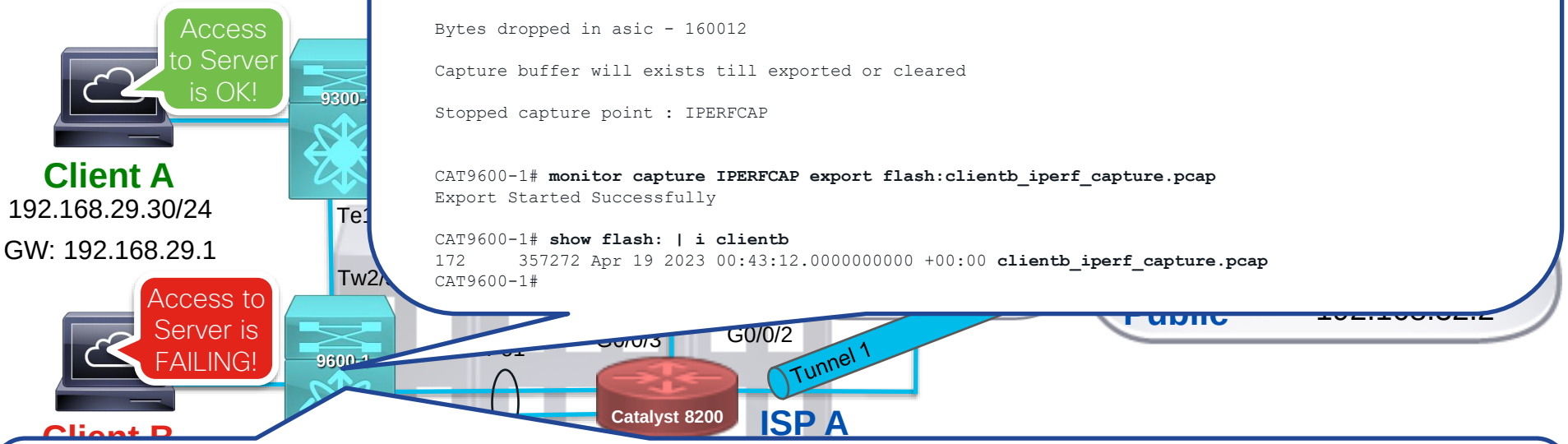
Degraded Performance



Real World

Degraded Perform

```
CAT9600-1# monitor capture IPERFCAP vlan 119 both match ipv4 host 192.168.29.132 host 192.168.32.2
CAT9600-1# monitor capture IPERFCAP start

CAT9600-1# monitor capture IPERFCAP stop
Capture statistics collected at software:
        Capture duration - 22 seconds
        Packets received - 268
        Packets dropped - 0
        Packets oversized - 0

Bytes dropped in asic - 160012

Capture buffer will exists till exported or cleared

Stopped capture point : IPERFCAP


CAT9600-1# monitor capture IPERFCAP export flash:clientb_iperf_capture.pcap
Export Started Successfully

CAT9600-1# show flash: | i clientb
172     357272 Apr 19 2023 00:43:12.0000000000 +00:00 clientb_iperf_capture.pcap
CAT9600-1#
```




```
CAT9600-1# show monitor capture file flash:clientb_iperf_capture.pcap brief
Starting the packet display ........ Press Ctrl + Shift + 6 to exit

    1   0.000000 192.168.29.132 -> 192.168.32.2 TCP 78 39488 -> 5201 [SYN] Seq=0 Win=64676 Len=0 MSS=1406 SACK_PERM=1 TSval=1112716044 TSecr=0 WS=128
    2   0.001182 192.168.29.132 -> 192.168.32.2 TCP 70 39488 -> 5201 [ACK] Seq=1 Ack=1 Win=64768 Len=0 TSval=1112716045 TSecr=2858875194
    3   0.001217 192.168.29.132 -> 192.168.32.2 TCP 107 39488 -> 5201 [PSH, ACK] Seq=1 Ack=1 Win=64768 Len=37 TSval=1112716045 TSecr=2858875194
    4   0.002761 192.168.29.132 -> 192.168.32.2 TCP 70 39488 -> 5201 [ACK] Seq=38 Ack=2 Win=64768 Len=0 TSval=1112716047 TSecr=2858875196
```

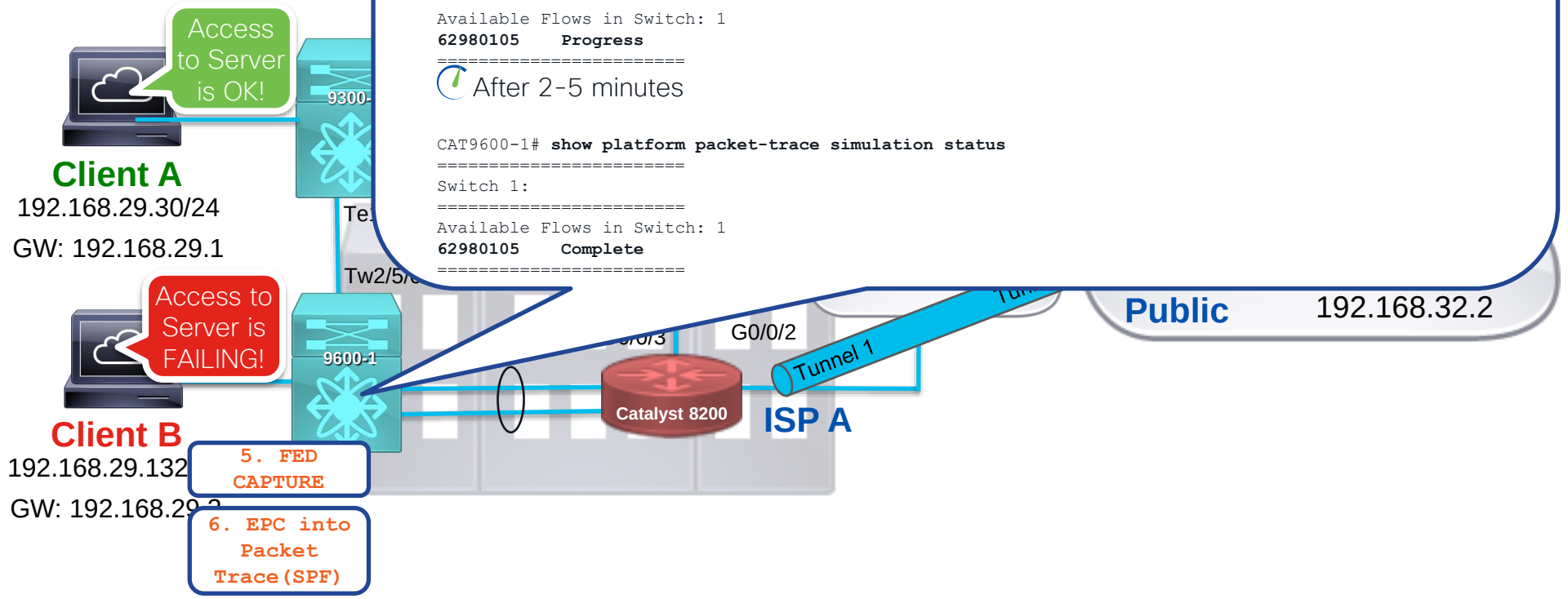
Real World

Degraded Perform

```
CAT9600-1# debug platform condition feature simulate pcap flash:clientb_iperf_capture.pcap number 1
CAT9600-1# debug platform condition feature simulate interface tw2/5/0/1

CAT9600-1# debug platform condition start
CAT9600-1# debug platform packet-trace simulation start

CAT9600-1# show platform packet-trace simulation status
==========================
Switch 1:
==========================
Available Flows in Switch: 1
62980105     Progress
==========================
```

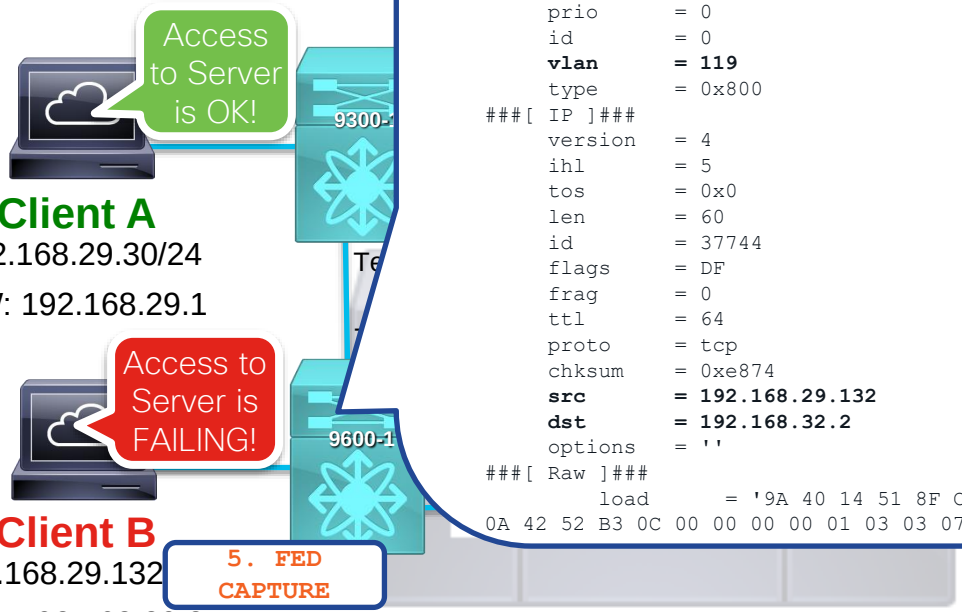
After 2-5 minutes

```
CAT9600-1# show platform packet-trace simulation status
==========================
Switch 1:
==========================
Available Flows in Switch: 1
62980105     Complete
==========================
```



Real World

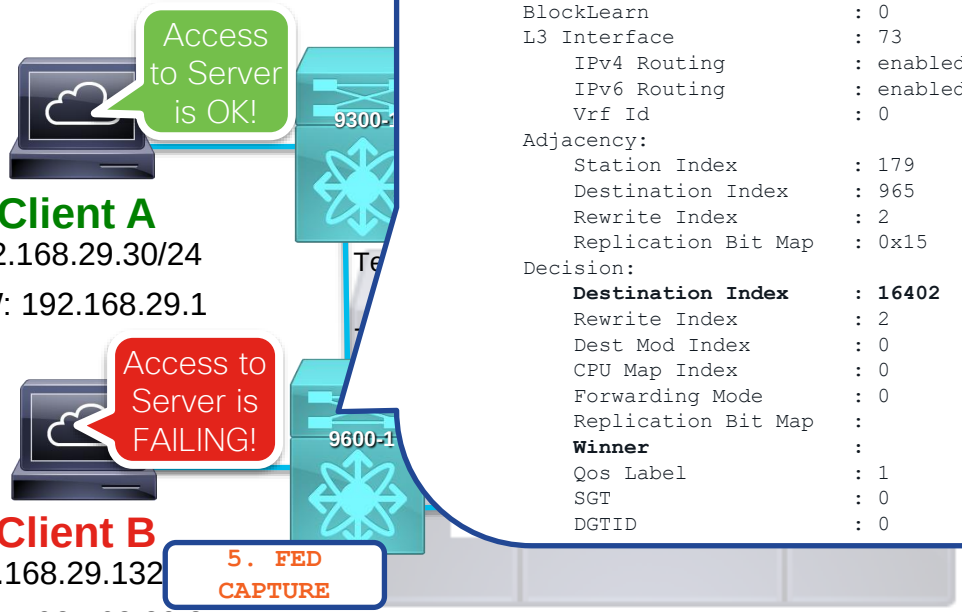
Degraded Perform




```
CAT9600-1# show platform packet-trace simulation flowid 62980105 summary
========================
Switch Active:
========================
Input Packet Details:
###[ Ethernet ]###
  dst       = 00:00:0c:07:ac:02
  src       = 00:50:56:b4:23:4f
  type      = 0x8100
###[ 802.1Q ]###
     prio      = 0
     id        = 0
     vlan      = 119
     type      = 0x800
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 60
     id        = 37744
     flags     = DF
     frag      = 0
     ttl       = 64
     proto     = tcp
     chksum    = 0xe874
     src       = 192.168.29.132
     dst       = 192.168.32.2
     options   = ''
###[ Raw ]###
        load      = '9A 40 14 51 8F CB 40 06 00 00 00 00 A0 02 FC A4 18 F8 00 00 02 04 05 7E 04 02 08
0A 42 52 B3 0C 00 00 00 00 01 03 03 07'
```

Real World

Degraded Perform




```
Ingress:
   Port                          : TwentyFiveGigE2/5/0/1
   Global Port Number            : 961
   Local Port Number             : 1
   Asic Port Number              : 1
   Asic Instance                 : 0
   Vlan                          : 119
   Mapped Vlan ID                : 5
   STP Instance                  : 4
   BlockForward                  : 0
   BlockLearn                    : 0
   L3 Interface                  : 73
       IPv4 Routing              : enabled
       IPv6 Routing              : enabled
       Vrf Id                    : 0
   Adjacency:
       Station Index             : 179
       Destination Index         : 965
       Rewrite Index             : 2
       Replication Bit Map       : 0x15    ['localData', 'remoteData', 'coreData']
   Decision:
       Destination Index         : 16402   [DI_ETHER_CHANNEL]
       Rewrite Index             : 2       [RI_L2]
       Dest Mod Index            : 0       [IGR_FIXED_DMI_NULL_VALUE]
       CPU Map Index             : 0       [CMI_NULL]
       Forwarding Mode           : 0       [Bridging]
       Replication Bit Map       :         ['localData', 'remoteData', 'coreData']
       Winner                    :         L2DESTMACVLAN LOOKUP
       Qos Label                 : 1
       SGT                       : 0
       DGTID                     : 0
```

```
Egress:
   Possible Replication       :
       Port                   : TwentyFiveGigE2/5/0/2
       Port                   : TwentyFiveGigE2/5/0/3
   Output Port Data           :
     Port                     : TwentyFiveGigE2/5/0/3
       Global Port Number     : 1285
       Local Port Number      : 25
       Asic Port Number       : 11
       Asic Instance          : 3
       Unique RI              : 0
       Rewrite Type           : 0       [Unknown]
       Mapped Rewrite Type    : 4       [L2_BRIDGE_INNER_IPv4]
       Vlan                   : 119
       Mapped Vlan ID         : 5

Output Packet Details:
   Port                       : TwentyFiveGigE2/5/0/3
###[ Ethernet ]###
  dst       = dc:77:4c:6f:a7:1f
  src       = 00:50:56:b4:23:4f
  type      = 0x8100
###[ 802.1Q ]###
     prio      = 0
     id        = 0
     vlan      = 119
     type      = 0x800
###[ IP ]###
        version   = 4
        ihl       = 5
        tos       = 0x0
        len       = 84
        id        = 4258
        flags     =
        frag      = 0
        ttl       = 255
        proto     = icmp
        chksum    = 0x9502
        src       = 192.168.29.132
        dst       = 192.168.29.2
        options   = ''
```

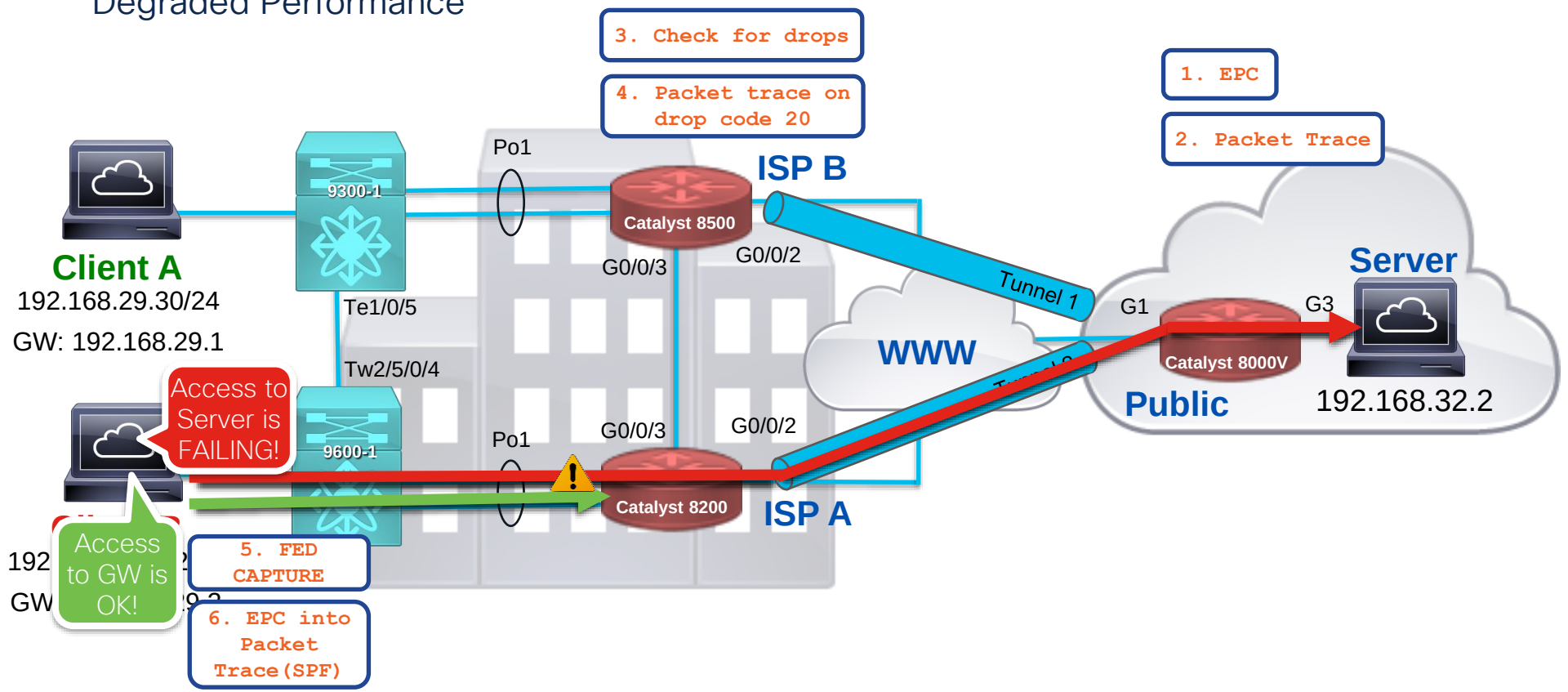
ICMP TO GATEWAY

```
Egress:
   Possible Replication       :
       Port                   : TwentyFiveGigE2/5/0/2
       Port                   : TwentyFiveGigE2/5/0/3
   Output Port Data           :
     Port                     : TwentyFiveGigE2/5/0/2
       Global Port Number     : 1346
       Local Port Number      : 25
       Asic Port Number       : 11
       Asic Instance          : 3
       Unique RI              : 0
       Rewrite Type           : 0       [Unknown]
       Mapped Rewrite Type    : 4       [L2_BRIDGE_INNER_IPv4]
       Vlan                   : 119
       Mapped Vlan ID         : 5

Output Packet Details:
   Port                       : TwentyFiveGigE2/5/0/2
###[ Ethernet ]###
  dst       = 00:00:0c:07:ac:02
  src       = 00:50:56:b4:23:4f
  type      = 0x8100
###[ 802.1Q ]###
     prio      = 0
     id        = 0
     vlan      = 119
     type      = 0x800
###[ IP ]###
        version   = 4
        ihl       = 5
        tos       = 0x0
        len       = 60
        id        = 37744
        flags     = DF
        frag      = 0
        ttl       = 64
        proto     = tcp
        chksum    = 0xe874
        src       = 192.168.29.132
        dst       = 192.168.32.2
        options   = ''
```

TCP TO SERVER

Real World Example

Degraded Performance



Re

Deg

```
cisco@linux2:~$ iperf3 -b 200M -c 192.168.32.2
Connecting to host 192.168.32.2, port 5201
[  5] local 192.168.29.132 port 37510 connected to 192.168.32.2 port 5201
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec  23.8 MBytes   199 Mbits/sec   12    133 KBytes
[  5]   1.00-2.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   2.00-3.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   3.00-4.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   4.00-5.00   sec  23.8 MBytes   199 Mbits/sec    0    133 KBytes
[  5]   5.00-6.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   6.00-7.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   7.00-8.00   sec  23.9 MBytes   200 Mbits/sec    0    133 KBytes
[  5]   8.00-9.00   sec  23.8 MBytes   199 Mbits/sec    0    147 KBytes
[  5]   9.00-10.00  sec  23.9 MBytes   200 Mbits/sec    0    147 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Retr
[  5]   0.00-10.00  sec   312 MBytes   200 Mbits/sec   12             sender
[  5]   0.00-10.00  sec   311 MBytes   200 Mbits/sec                  receiver
iperf Done.
cisco@linux2:~$
```

1. EPC

2. Packet Trace

Clien

192.168.

GW: 192.168.

Access to Server is OK!

Access to GW is OK!

5. FED CAPTURE

6. EPC into Packet Trace (SPF)

TW2/5/0/4

9600-1

Po1

G0/0/3

G0/0/2

Catalyst 8200

ISP A

1

G3

Catalyst 8000V

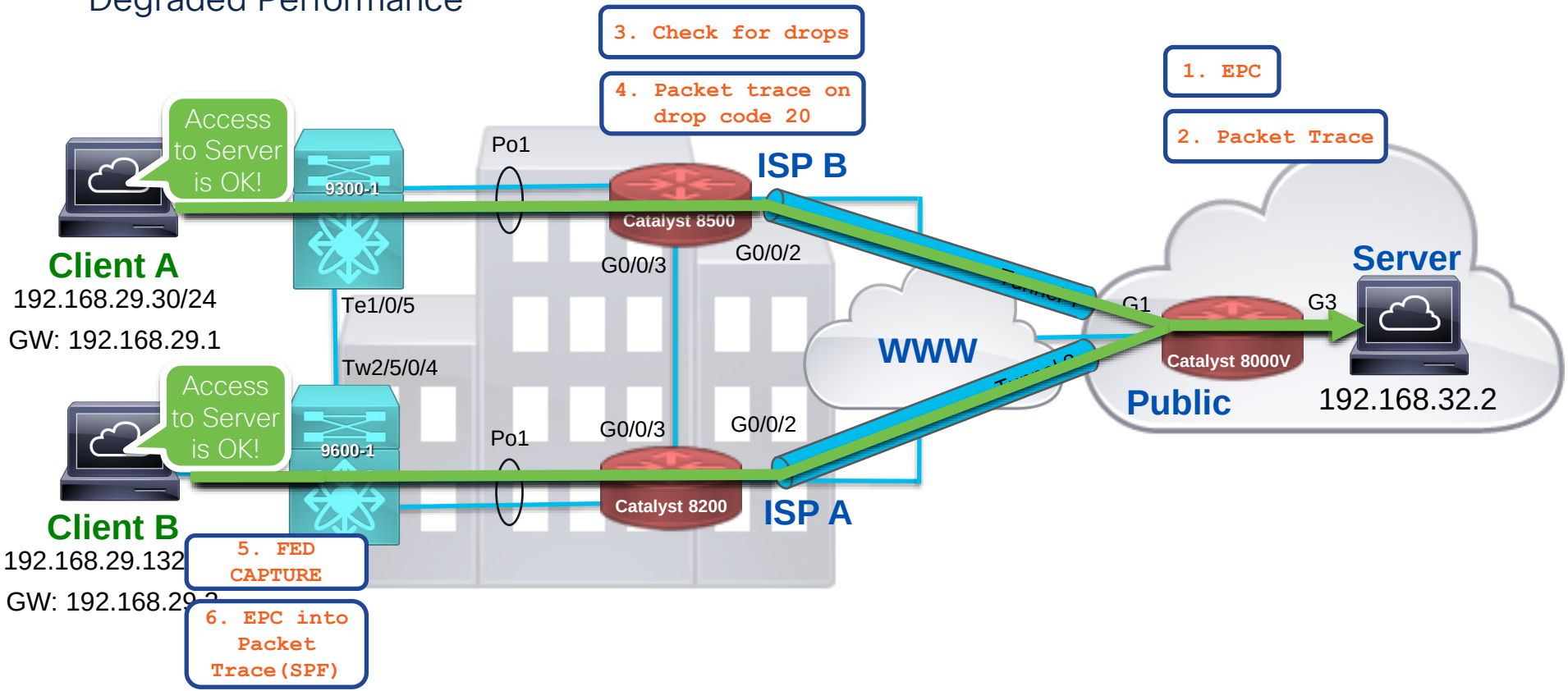
Public

Server

192.168.32.2

Real World Example

Degraded Performance



Summary & Take Away

Packet Capturing Tools

Usage Considerations

Tool	Impact	Comments
Show commands		This command shows detail of the packets in the system buffer
Catalyst 3650/3850/9000 FED Tracing/Packet Capture		Uses limited CPU/memory resources and can be run during high CPU utilization
Flexible NetFlow		For software-based forwarding platforms, this feature utilizes memory/buffer and CPU cycles. For hardware-based forwarding platforms, number of flows is limited by the hardware capacity.
SPAN / RPSAN / ERSPAN		Packet replication is performed by a specific ASIC. With oversubscription, this could cause adverse effects. With RSPAN, the replicated traffic may get flooded throughout the network. ERSPAN may require CPU cycles for decapsulation.
Embedded Packet Capture		The traffic captured by these tools is saved in the system memory/buffer. It is recommended to fine-tune the capture filters/ACLs to reduce the number of packets captured, size of the packets, etc.
Packet Trace (Switches)		Captures a single packet (PSV) or mimics packet forwarding decision (SPF). Perfectly safe to run with any packet type or conditions.
Packet Trace (Routers)		Similar concerns to Embedded Packet Capture (above)
Debug Commands		Use caution, can increases CPU utilization, filters reduce impact

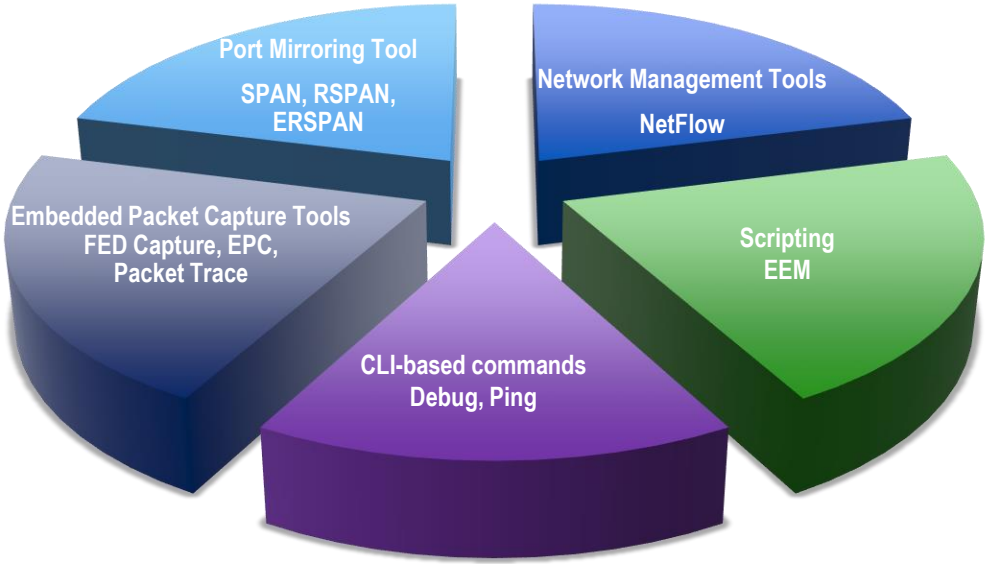
Packet Capturing Tool	Control Plane	Data Plane	PCAP	Header Info	Full Packet	Local Viewing	Remote Viewing	Filtering	Single Packet	Forwarding Information	CLI Analyzer Support	Platform
Flexible NetFlow	●	●		●		●	●			●		All
FED Tracing/Packet Capture	●			●	●	●		●				3650/3850 & 9000
SPAN/RSPAN/ERSPAN		●	●	●	●		●					Switches & IOS-XE Routers
Embedded Packet Capture	●	●	●	●	●	●	●	●			●	All Routers /IOS-XE Switches
Packet Trace (Routers)	●*	●		●	●	●		●		●		IOS-XE Routers
Packet Trace (Switches-SPF)	●	●		●		●			●	●		Catalyst 9000 Series UADP2.0, 2.0 mini, & 3.0
Packet Trace (Switches-PSV)		●		●		●		●	●	●		Catalyst 9600 & 9500H UADP 3.0

*Cannot capture on the management port

Overview of Troubleshooting Tools

Summary and Take Away ....

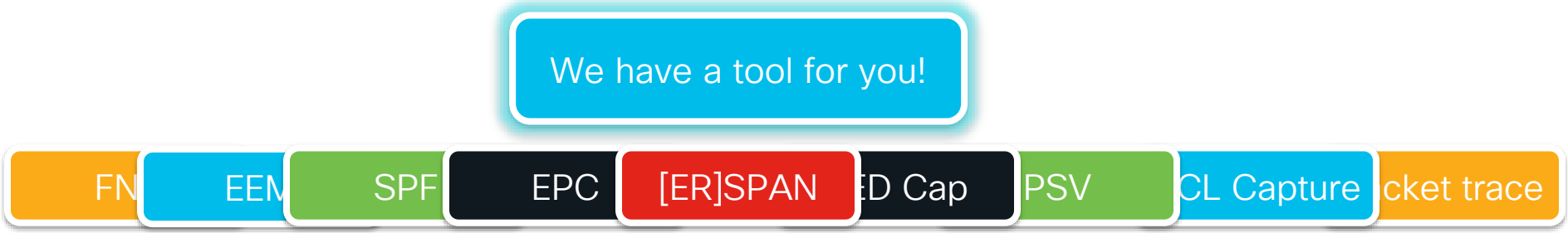
- Capture Control Plane traffic:
  - Show/Debug Commands, FED Tracing,
- Copy Data Plane traffic to external device:
  - SPAN/RSPAN/ERSPAN
- Copy Data Plane traffic to internal buffer:
  - Flexible NetFlow (FNF), Embedded Packet Capture (EPC), and Packet Trace (Routers)
- Capture Data Plane Frame and Forwarding Headers:
  - Packet Trace (Switches)



Overview of Troubleshooting Tools

Summary and Take Away ….

- Cisco Routers and Switches are advanced and feature-rich, built with keeping end-users and network engineers in mind.
- Cisco provides a rich set of packet capturing tools embedded and supported across the spectrum of our 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.



References

ELAM, NetDR, FED Tracing and CPU Queue Debug

- ELAM Overview: (has links to ELAM examples for many different platforms and modules) http://www.cisco.com/c/en/us/support/docs/switches/nexus-7000-series-switches/116648-technote-product-00.html
- Catalyst 6500 Series Switches NetDR Tool for CPU-Bound Packet Captures: http://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/116475-technote-product-00.html
- High CPU Troubleshooting in Catalyst 3850 with “FED Tracing”: http://www.cisco.com/c/en/us/support/docs/switches/catalyst-3850-series-switches/117594-technote-hicpu3850-00.html
- Troubleshooting High CPU in Catalyst 4500 switches – with CPU Queue Debug:http://www.cisco.com/c/en/us/support/docs/switches/catalyst-4000-series-switches/65591-cat4500-high-cpu.html

References

Flexible NetFlow

- Flexible NetFlow - Data Sheets, Q&A and White Paper
  http://www.cisco.com/en/US/products/ps6965/products_ios_protocol_option_home.html
- Flexible NetFlow Configuration Guide
  https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/fnetflow/configuration/15-mt/fnf-15-mt-book/fnf-fnetflow.html
- NetFlow Version 9
  http://www.cisco.com/en/US/products/ps6645/products_ios_protocol_option_home.html
- Flexible NetFlow - Data Sheets and Literature
  http://www.cisco.com/en/US/products/ps6601/prod_literature.html

References

Embedded Packet Capture Tools

- Cisco IOS-XE Embedded Packet Capture: https://www.cisco.com/c/en/us/support/docs/ios-nx-os-software/ios-embedded-packet-capture/116045-productconfig-epc-00.html
- Cisco IOS Embedded Packet Capture – Data Sheet: http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps9913/datasheet_c78-502727.html
- Utilizing the New Packet Capture Feature – Cisco Support Community: https://supportforums.cisco.com/docs/DOC-5799
- Embedded Wireshark – Configuration Guide: http://www.cisco.com/en/US/docs/switches/lan/catalyst4500/15.1/XE_330SG/configuration/guide/wireshrk.html
- IOS-XE Packet Trace:

  http://www.cisco.com/c/en/us/support/docs/content-networking/adaptive-session-redundancy-asr/117858-technote-asr-00.html

Overview of Troubleshooting Tools

More to Know and Learn…

- Cisco IOS Embedded Event Manager (EEM) provides a real-time network event detection and onboard automation. http://www.cisco.com/en/US/products/ps6815/products_ios_protocol_group_home.html
- Cisco IP Service Level Agreements (SLAs) assures network service levels and proactively monitors network health and performance. http://www.cisco.com/en/US/products/ps6602/products_ios_protocol_group_home.html

Overview of Troubleshooting Tools

More to Know and Learn…

- Cisco ELAM allows capturing of a single packet during the forwarding decision on hardware forwarded platforms.
  http://www.cisco.com/c/en/us/support/docs/switches/nexus-7000-series-switches/116648-technote-product-00.html

  https://www.cisco.com/c/en/us/support/docs/switches/nexus-9000-series-switches/213848-nexus-9000-cloud-scale-asic-tahoe-nx-o.html

- Cisco Packet Tracer on the Nexus 9000 allows for visibility into the forwarding path of traffic through the switch.
  https://www.cisco.com/c/en/us/support/docs/switches/nexus-9000-series-switches/210592-Nexus-9000-Packet-Tracer-tool-explained.html

Continue Your Education

- Walk-in Labs (45-60 minutes, self paced labs)
  - **LABTRS-2456 – Packet Capturing Tools in Routing Environments**
    Hands-on scenarios with Embedded Packet Capture, Packet Trace, and NetFlow
  - **LABTRS-2391 – Packet Capturing Tools in Enterprise Switching Environments**
    Hands-on scenarios focused on Embedded Packet Capture, FED Packet Capture, and Show Platform Forward on the Catalyst 9000 Series Switches
  - **LABTRS-2048 – Packet Trace and Conditional Debugger on IOS-XE Routers**
    Hands-on scenarios with Packet Trace and Conditional Debugger to answer questions about specific traffic flows and the features configured.

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Additional Breakout Sessions and Labs

For more info...

LABTRS-2456 Packet Capturing Tools in Routing Environments
LABTRS-2391 Packet Capturing Tools in Enterprise Switching Environments
LABTRS-2048 Packet Trace and Conditional Debugger on IOS-XE Routers

LABCRT-2452 CCNP ENCOR – Core Enterprise Network Technologies Practice Lab
LABCRT-2460 CCNP ENARSI – Implementing Cisco Enterprise Advanced Routing and Services Practice Lab
LABCRT-2464 Troubleshoot like a CCNP Practice Lab

BRKTRS-3475 Advanced Troubleshooting of the cat8k, asr1k, ISR and SDWAN Edge Made Easy
BRKXAR-2003 Extending Enterprise Network into Public Cloud with Cisco Catalyst 8000V Edge Software
BRKTRS-3090 Troubleshooting Cisco Catalyst 9000 Series Switches

Fill out your session surveys!

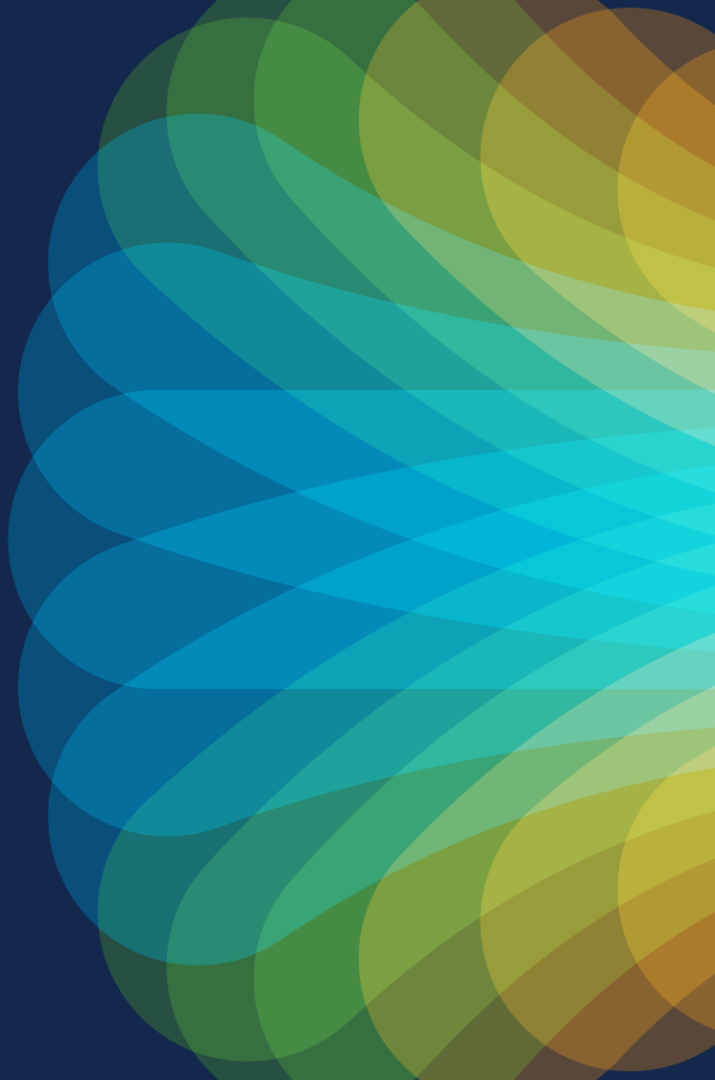
Attendees who fill out a minimum of four session surveys and the overall event survey will get **Cisco Live-branded socks** (while supplies last)!

Attendees will also earn 100 points in the **Cisco Live Game** for every survey completed.

These points help you get on the leaderboard and increase your chances of winning daily and grand prizes

The bridge to possible

Thank you



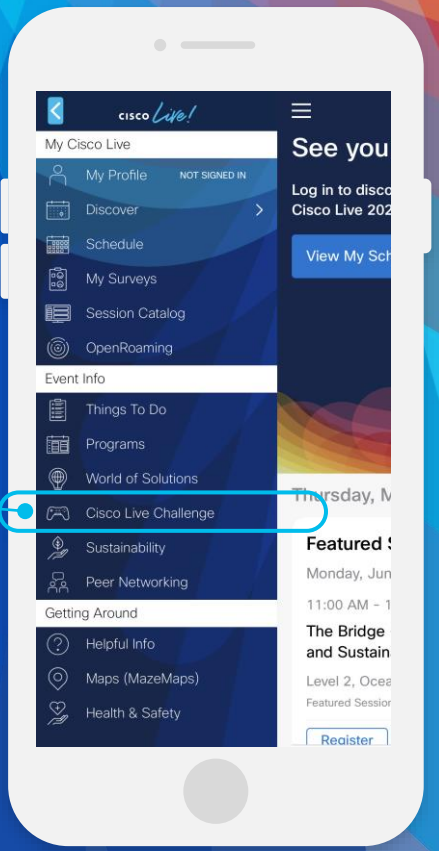
Cisco Live
Challenge

Gamify your Cisco Live experience!

Get points for attending this session!

How:

1. Open the Cisco Events App.
2. Click on 'Cisco Live Challenge' in the side menu.
3. Click on View Your Badges at the top.
4. Click the + at the bottom of the screen and scan the QR code:




cisco Live!
Let's go
#CiscoLive