



*TOMORROW  
starts here.*

Cisco *live!*



# Overview of Troubleshooting and Packet Capturing Tools in Cisco Switches and Routers

BRKARC-2011

Jeremy Moschner  
High Touch Engineer, Cisco Services

#clmel

Cisco *live!*



```
Path: C:\DOOM2 14-02-07 7:48:53 pm

c:\N
├── doom2
├── skyroads
├── telix
├── windows
│   ├── system
│   └── temp
├── wolf3d
├── word55
└── xtgold

default .cfg      serssetup.exe
modem .cfg        setup .exe
doom2 .exe        modem .num
ipxsetup.exe      modem .str

DIR Avail Branch Compare Dele
COMMANDS Oops! Print Rename Showal
└─ file F7 autoview F8 split F9
```

FILE \*.\*

DISK C:C\_DRIVE  
Available 14,235  
Bytes 110,540,800

Evaluation Copy - Not for Resale

XTREME™

<c> Executive Systems, Inc. 1985, 1989

All Rights Reserved Worldwide

Version 2.00E Serial # 999999

FILE: \*.\*

DISK: H: PROG1  
Available  
Bytes: 61,210,624

DISK Statistics

Total	
Files:	301
Bytes:	0
Matching	
Files:	0
Bytes:	0
Tagged	
Files:	0
Bytes:	0

Installed For IBM PC or Compatible <Quick Display>

ALL OF THESE NUMBERS  
ARE SOMEWHERE...



CHANGE ALL THE NUMBERS!

00000000	5241	5354	494E	0045	0008	0000	0000	0000	.....HEX.....
00000010	0100	0001	0009	0011	0010	0012	0010	0011	.....EDITOR...
00000020	0011	0010	0012	0010	0011	00F7	FF2E	012E	.....
00000030	0132	0132	0132	803C	802F	8044	8041	805C	.FOR.....
00000040	8000	0000	00B0	DA13	00F9	E104	000D	000D	.....THE.....
00000050	0007	0707	0700	0000	0000	0000	0000	0000	.....WIN..
00000060	0000	0000	0000	0000	00AE	0600	0000	0000	.!!!.....

# Agenda

- AAAA  
The Game Plan
- Packets in Flight  
Flexible Netflow, EPC and ERSPAN
- Where is my Packet?  
ELAM Enhanced
- Taming High CPU  
Ethanalyser and NetDR



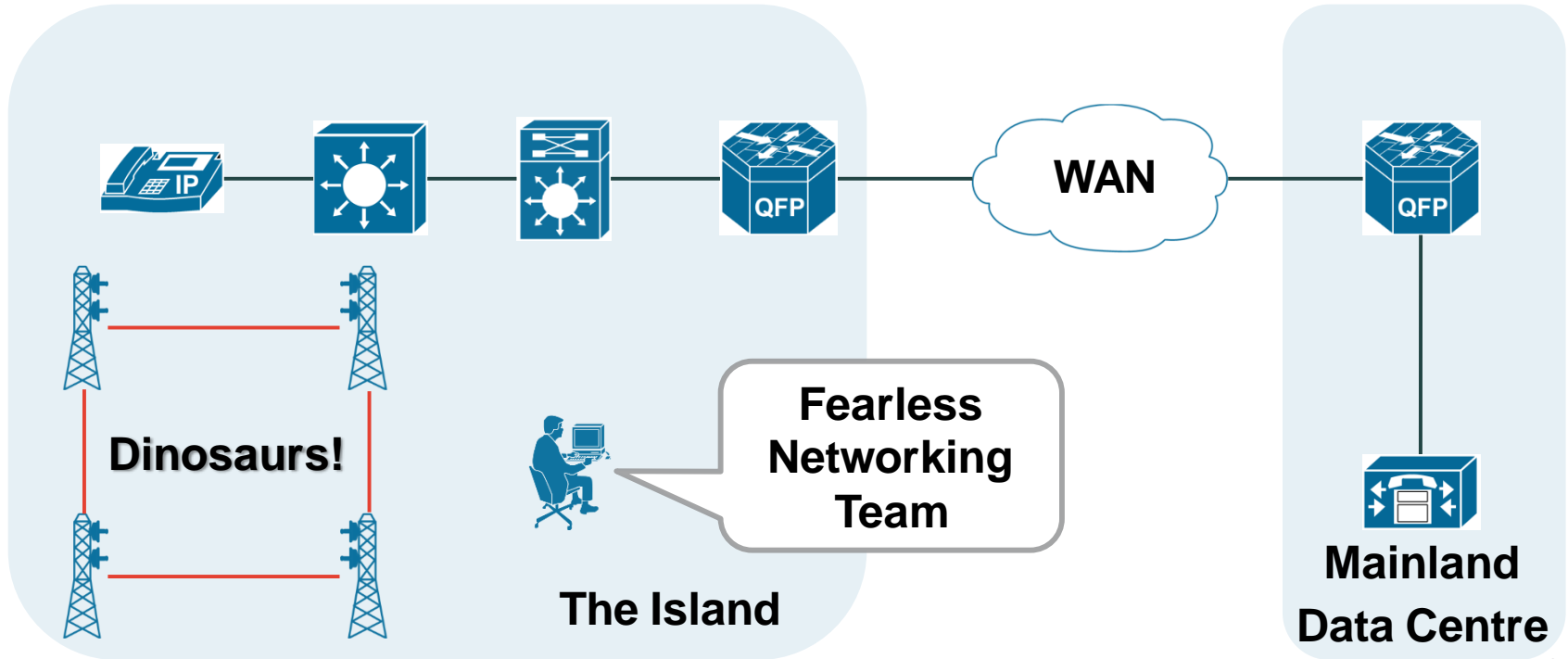
# Our Objectives

1. Know your problem
2. Choose a Tool ...
3. ...and use it!



# A “Real-world” Example

## Dino Farms – Your Friendly Dinosaur Park



# Acronyms and Abbreviations



<b>AAAA</b>	Assess, Acquire, Analyse and Act	<b>DBUS</b>	Data Bus	<b>FNF</b>	Flexible Netflow
<b>ACL</b>	Access Control List	<b>DDoS</b>	Distributed Denial of Service Attack	<b>FSPAN</b>	Flow-based SPAN
<b>ASIC</b>	Application-specific Integrated Circuit	<b>ELAM</b>	Embedded Logic Analyser Module	<b>LIF</b>	Logical Interface
<b>CEF</b>	Cisco Express Forwarding	<b>ELAM-E</b>	ELAM Enhanced	<b>GRE</b>	Generic Routing Encapsulation
<b>CoPP</b>	Control Plane Policing	<b>EPC</b>	Embedded Packet Capture	<b>IFE</b>	Input Forwarding Engine
<b>CoS</b>	Class of Service	<b>ERSPAN</b>	Encapsulated Remote Switched Port Analyser	<b>LTL</b>	Local Target Logic
<b>CPU</b>	Central Processing Unit	<b>FE</b>	Forwarding Engine	<b>MPA</b>	Mini-protocol Analyser
<b>CUCM</b>	Cisco Unified Communications Manager	<b>FIFO</b>	First In, First Out	<b>NetDR</b>	Net Driver

# Acronyms and Abbreviations

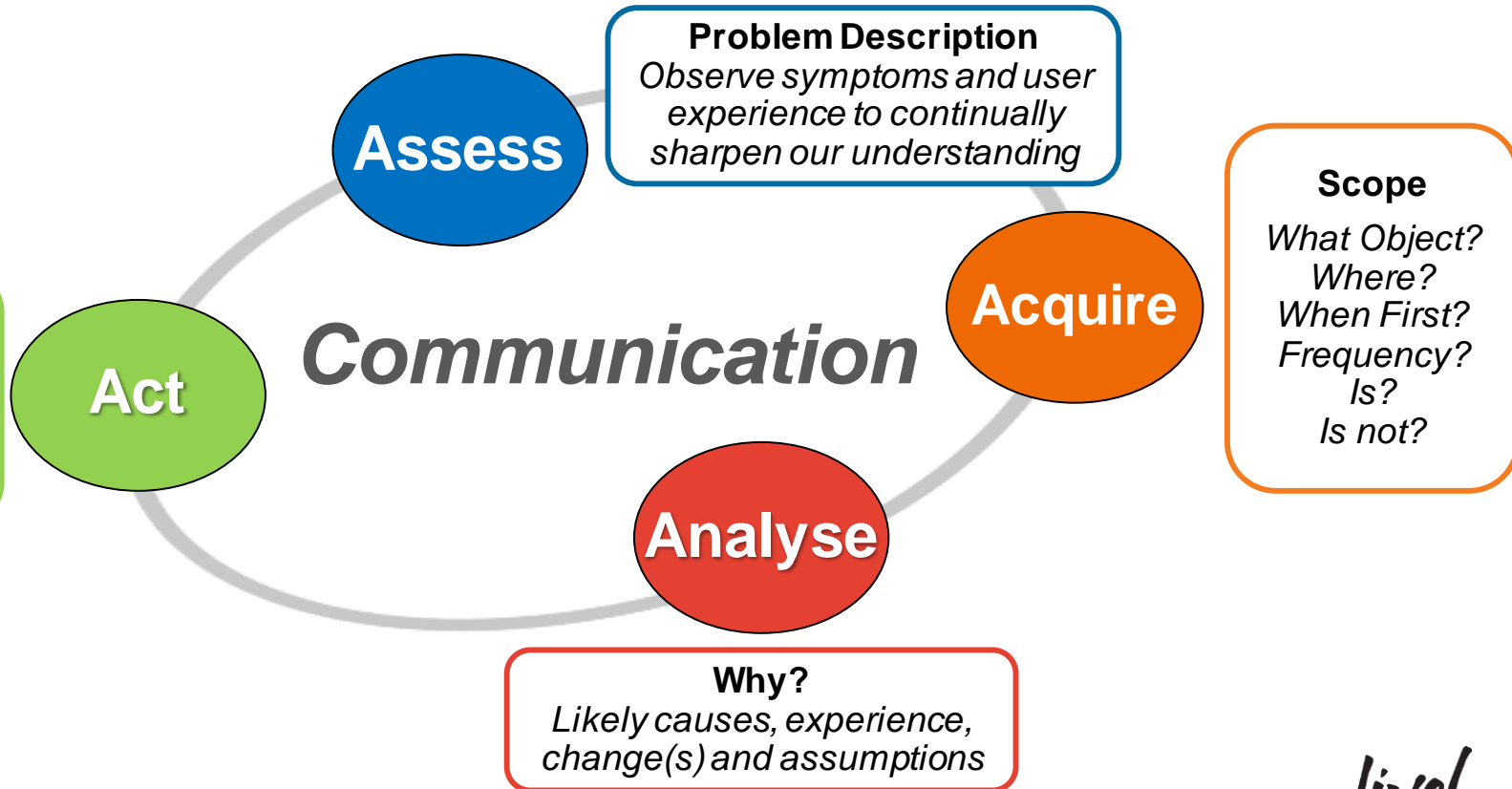


<b>OFE</b>	Output Forwarding Engine	<b>RFC</b>	Request for Comment (IETF Standards)	<b>ToS</b>	Type of Service
<b>PCAP</b>	Packet Capture File	<b>RP</b>	Route Processor	<b>TTL</b>	Time to Live
<b>PTP</b>	Precision Time Protocol (IEEE 1588)	<b>RSPAN</b>	Remote SPAN	<b>VDC</b>	Virtual Device Context
<b>QFP</b>	Quantum Flow Processor (Used on ASR 1000)	<b>SCTP</b>	Stream Control Transmission Protocol	<b>WAN</b>	Wide-area Network
<b>QoS</b>	Quality of Service	<b>SP</b>	Switch Processor		
<b>RBUS</b>	Result Bus	<b>SPAN</b>	Switched Port Analyser		

A long-exposure photograph of a city street at night. The foreground is filled with vibrant, multi-colored light trails from moving vehicles, creating a sense of motion. In the background, a modern pedestrian bridge with blue lighting spans the street. Tall buildings with illuminated windows and storefronts line the street, and several flags are visible on the left side.

# 1. AAAA – The Game Plan

# AAAA – The Game Plan



# AAAA – The Game Plan



## Action Plans, Questions and Problem Description

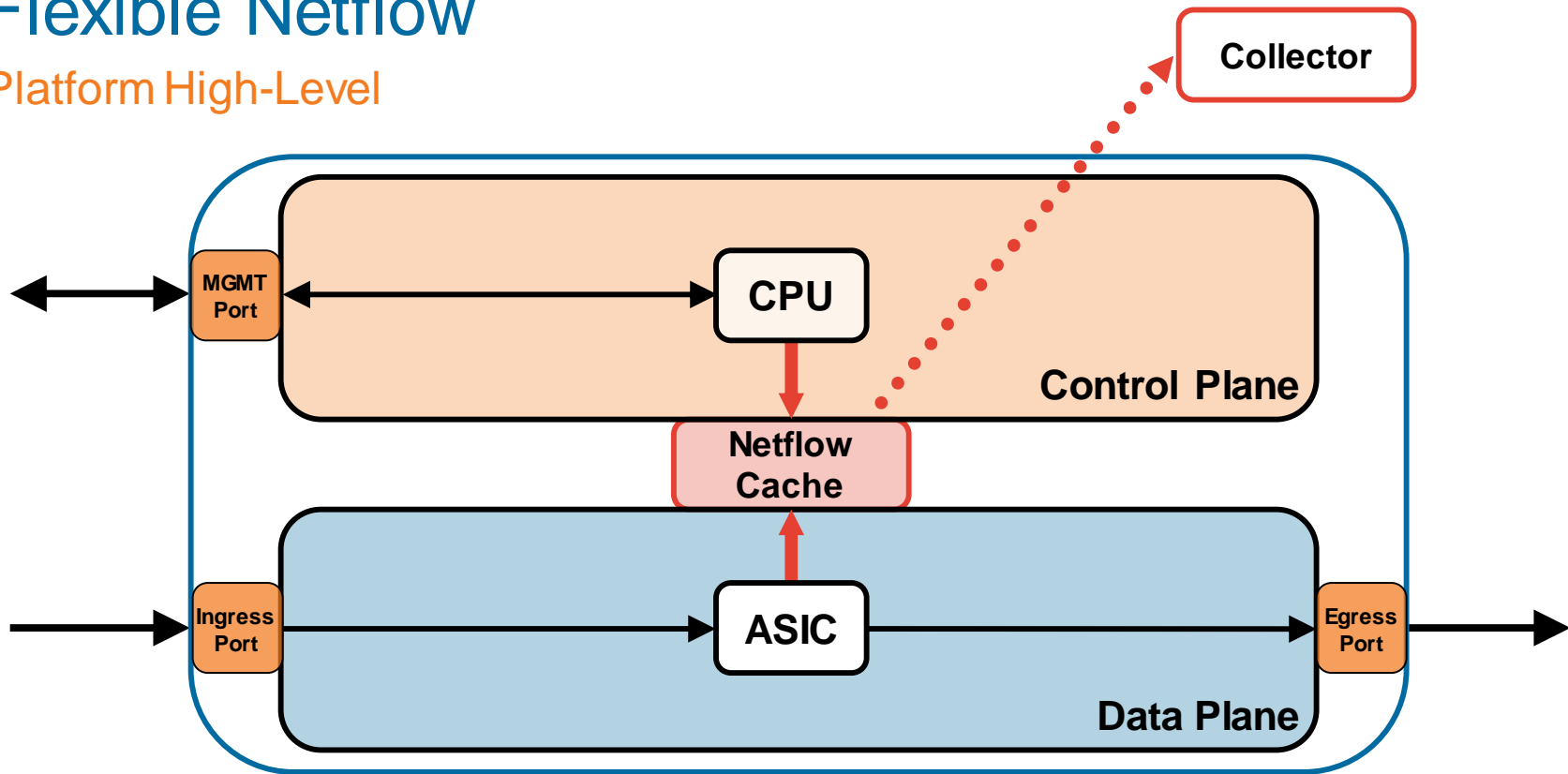
- An action plan is key to a structured troubleshooting process:
  - What action?
  - Who will do it?
  - By when?
- Problems don't exist in a vacuum – taking these four steps produces a specific (sometimes very detailed) problem description.
- Reveal assumptions by clarification (eg, “what do you mean by that?”)
- Always refine the action plan and problem description as you go.



## 2. Flexible Netflow

# Flexible Netflow

## Platform High-Level



# Flexible Netflow



## Overview and Advantages

- Traditional Netflow defines a “flow” based on a narrow range of criteria.
- Flexible Netflow (**FnF**) allows the user to select “keys” to define a “flow record”. This record then matches and tracks these user-defined flows.
- There are numerous records in FnF, matching information from layers 2 to 7.
- Exporting flow data to an external collector is optional, but a good idea if you want to retain relevant data for planning and monitoring. Cached results can be seen directly on the router.
- FnF caches flow entries in memory or in a hardware cache on higher-end platforms. See individual product datasheets for Cisco’s officially supported netflow cache entries.

# FnF Cache and Export

## Fields

### “Traditional” Netflow Fields

- *Source / Destination IPv4 Address*
- *Source / Destination Port*
- *IP Protocol Type and ToS*
- *Input Interface*

### Flexible Netflow Fields

- *Extensive Layer 2 to 7 Fields  
(eg. IOS-XE 3.10S supports 40 fields)*
- *IPv4 and IPv6 Ingress or Egress*
- *Unicast, Multicast and MPLS.*



Netflow  
Cache



Cisco *live!*

# FnF Cache and Export

## Export

### “Traditional” Netflow Export

- *Typically Version 5*
- *Required to view Netflow Data*
- *Commonly IPv4 over UDP*

### Flexible Netflow Export

- *Version 9 and IPFIX IETF Standards (RFCs 7011 and 7015)*
- *Export is Optional*
- *IPv4 and IPv6 over independent transport (UDP or SCTP).*



**Netflow  
Cache**

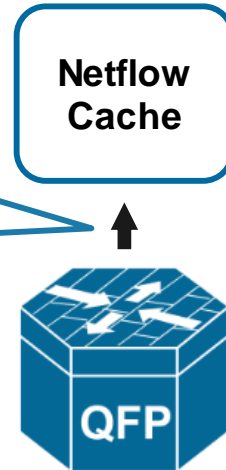


**Cisco** *live!*

# FnF Configuration

## Step 1 – Flow Record

```
flow record MYRECORD  
description into_fnf_cache  
match ipv4 source address  
match ipv4 destination address  
match transport source-port  
match transport destination-port  
match flow direction  
collect counter bytes  
collect counter packets  
collect timestamp sys-uptime first  
collect timestamp sys-uptime last
```



# FnF Configuration

## Step 2 – Optional Flow Export

```
flow exporter MYEXPORT  
description to_netflow_collector  
destination 10.5.5.24 vrf netflow_vrf  
export-protocol netflow-v9  
source Loopback10  
transport udp 2055
```



Netflow  
Cache



# FnF Configuration

## Step 3 – Linking a Record and Flow Monitor

```
flow monitor MYMONITOR  
description link_flow_monitor_and_record  
record MYRECORD  
exporter MYEXPORT
```



Netflow  
Cache



# FnF Configuration

## Step 4 – Apply and Go!

```
Interface GigabitEthernet 0/3/0
...
ip flow monitor MYMONITOR input
ip flow monitor MYMONITOR output
ip flow monitor MYMONITOR unicast
```



Netflow  
Cache

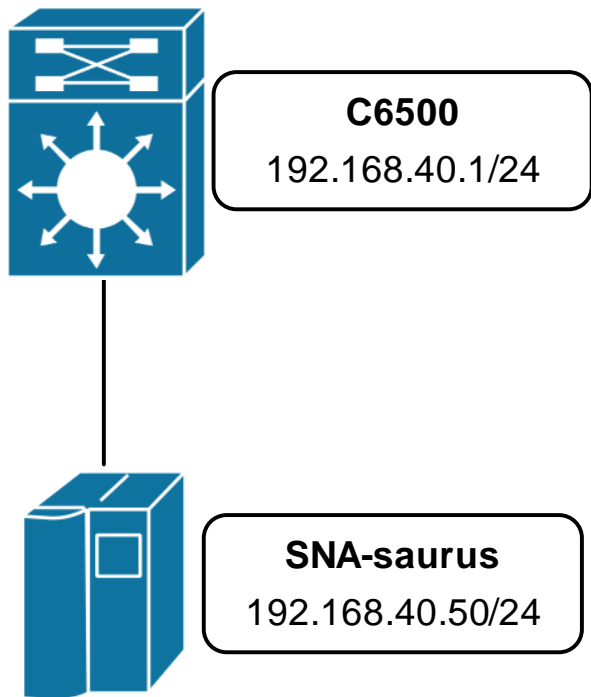


Cisco *live!*

# Example – “Top N Talkers Feature”

Assess

## 1. Assess



**Management**  
*“Help, there’s a DDoS attack!”*



# Example – “Top N Talkers Feature”



## 2. Acquire



**C6500**  
192.168.40.1/24



**SNA-saurus**  
192.168.40.50/24

C6500# show process cpu sorted

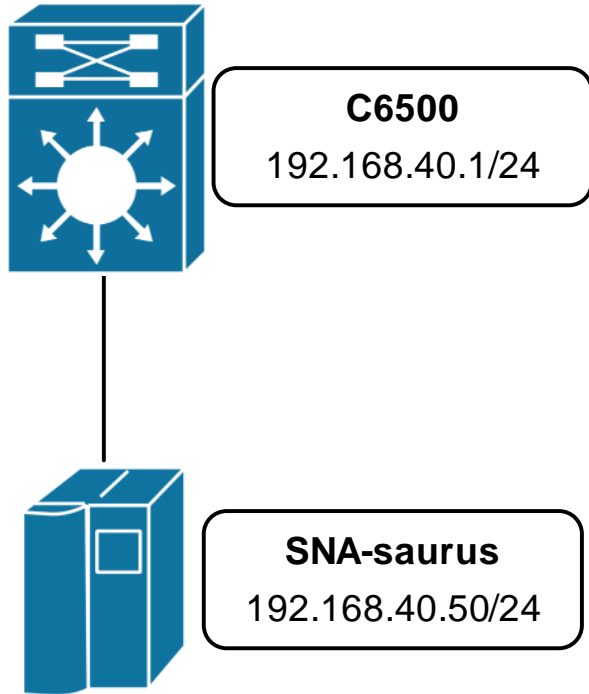
CPU utilization for five seconds: **65%/8%**; one minute: 63%; five minutes: 61%

PID	Runtime(ms)	Invoked	uSecs	5Sec	1Min	5Min	TTY	Process
<b>310</b>	<b>30544</b>	<b>189234</b>	<b>81</b>	<b>47.12%</b>	<b>45.11%</b>	<b>45.23%</b>	<b>0</b>	<b>IP Input</b>

# Example – “Top N Talkers Feature”



## 3. Analyse



```
flow RECORD copp-fnf-cef-in-rec
match ipv4 protocol
match ipv4 source address
match ipv4 destination address
match transport source-port
match transport destination-port
collect interface input
collect counter packets
```

```
flow MONITOR copp-fnf-cef-in
record copp-fnf-cef-in-rec
```

```
control plane
ip flow monitor copp-fnf-cef-in input
```

# Example – “Top N Talkers Feature”



## 3. Analyse

```
C6500# show flow monitor copp-fnf-cef-in cache  
sort counter packet
```

```
Processed 5 flows
```

```
Aggregated to 5 flows
```

```
Showing the top 5 flows
```

```
IPV4 SOURCE ADDRESS:      192.168.40.5
```

```
IPV4 DESTINATION ADDRESS: 192.168.40.1
```

```
TRNS SOURCE PORT:         48827
```

```
TRNS DESTINATION PORT:    63
```

```
IP PROTOCOL:              17
```

```
interface input:          V140
```

```
counter packets:         460983
```

# Example – “Top N Talkers Feature”



## 3. Analyse

```
C6500# show flow monitor copp-fnf-cef-in cache  
sort counter packet
```

```
...
```

```
IPV4 SOURCE ADDRESS: 192.168.40.5
```

```
IPV4 DESTINATION ADDRESS: 192.168.40.1
```

```
TRNS SOURCE PORT: 48827
```

```
TRNS DESTINATION PORT: 63
```

```
IP PROTOCOL: 17
```

```
interface input: V140
```

```
counter packets: 461181
```

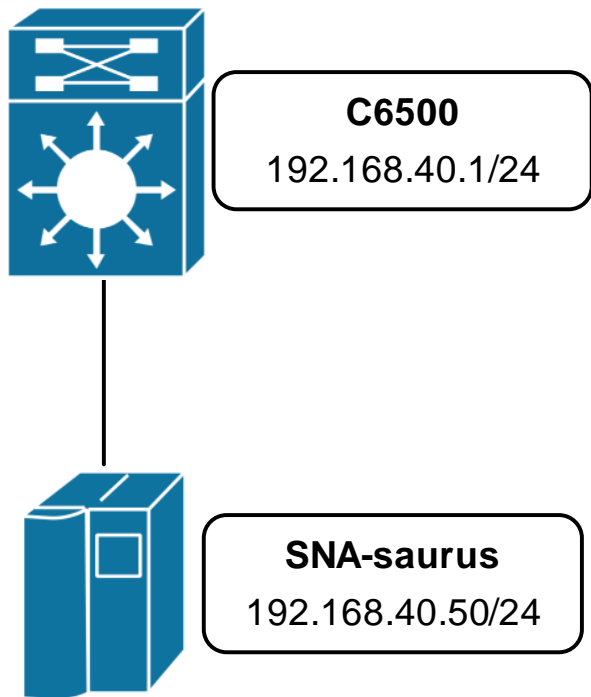
```
...
```

**This is host is the “top talker”**  
*High number of packets sent to  
UDP (IP Protocol 17) port 63  
(Whois++) on C6500.*

# Example – “Top N Talkers Feature”



## 4. Act



### ACL to drop this traffic

*Could some of this traffic be legitimate or expected?*

### Is CoPP configured on the device?

- *What rate of policing is appropriate?*
- *Should this be evaluated for the whole network?*

# Choose and Use

## Flexible Netflow (FnF)

- FnF can act as a quick way to identify flows and patterns
- Can also form part of a long-term visibility solution
- Have your favourite FnF monitors handy (or even better, preconfigured).



# Flexible Netflow – General Feature Support



- **IOS-XE**

- 3.2SE +
  - Catalyst 3850
- 3.3SE +
  - Catalyst 3650
- 3.8S +
  - ASR 1000 Family
  - ISR 4451-X
- 3.9S +
  - CSR 1000V
- 3.13S +
  - ISR 4300 and 4400 Series

- **IOS**

- 12.2SY +
  - Catalyst 6500, 6800 (with Supervisor 2T)
- 15.1SY +
  - Catalyst 6880-X



- **NX-OS**

- 4.0 +
  - Nexus 7010
- 4.1 +
  - Nexus 7018
- 5.2 +
  - Nexus 7009
- 6.1 +
  - Nexus 7004
- 6.2 +
  - Nexus 7700 Series

For more details on this feature across different platforms and software releases, please use Cisco's Feature Navigator tool available at:

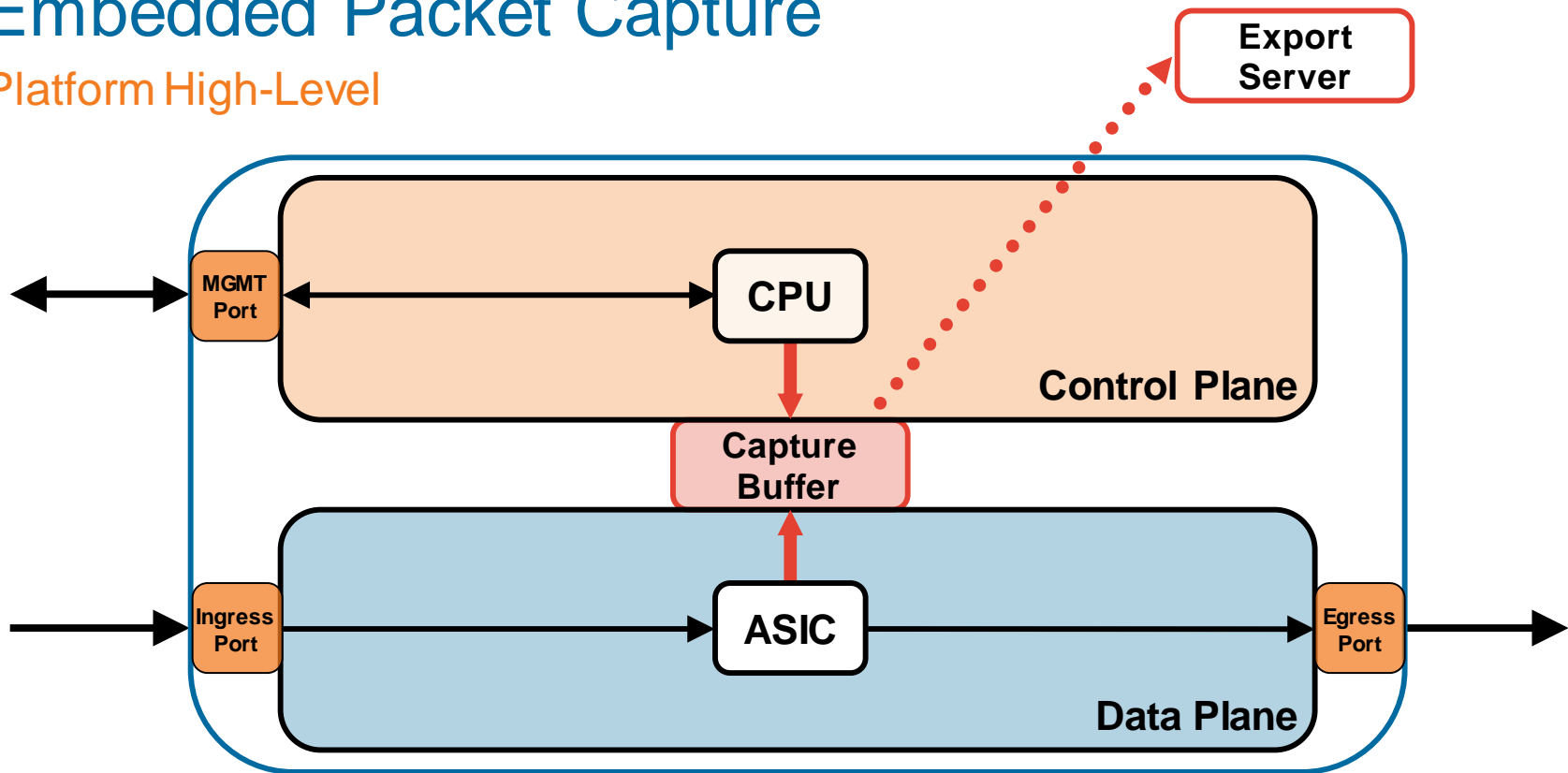
<http://www.cisco.com/go/fn>

A long-exposure photograph of a city street at night. The foreground is filled with vibrant, multi-colored light trails from moving vehicles, creating a sense of motion. In the background, a pedestrian bridge spans the street, and modern buildings with lit windows and signage line the street. Traffic lights are visible in the distance.

# 3. Embedded Packet Capture

# Embedded Packet Capture

## Platform High-Level



# Embedded Packet Capture (EPC)

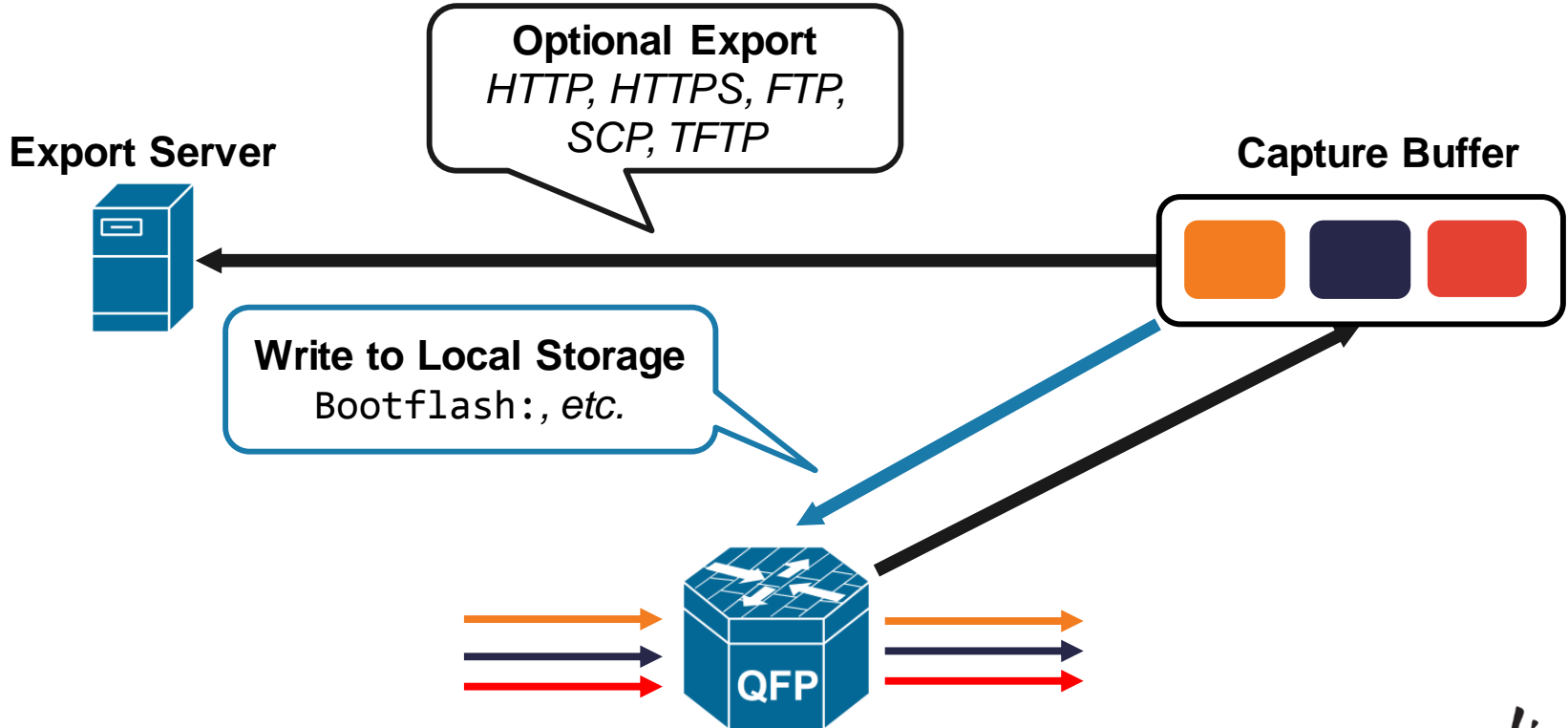


## Key Points

- Embedded Packet Capture (**EPC**) allows for raw packets to be captured at various points in the CEF packet-processing path (transit and “for-us” (punted) traffic).
- Exec-level command, with no configuration required (although an access-list is very useful).
- Supports exporting a PCAP file for external analysis.
- In hardware-based forwarding platforms (such as the Catalyst 6500) this is known as the Mini-Protocol Analyser (**MPA**). MPA utilises a SPAN session to capture traffic.
- Our focus here is on EPC for the ASR 1000.

# EPC Concept

ASR 1000



# EPC Configuration

## Step 1 – Define Capture Buffer (Linear vs Circular)

```
ASR# monitor capture MYCAP buffer circular
ASR# monitor capture MYCAP buffer size 5000
```

Capture Buffer



# EPC Configuration

## Step 2 – Define Capture Point

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



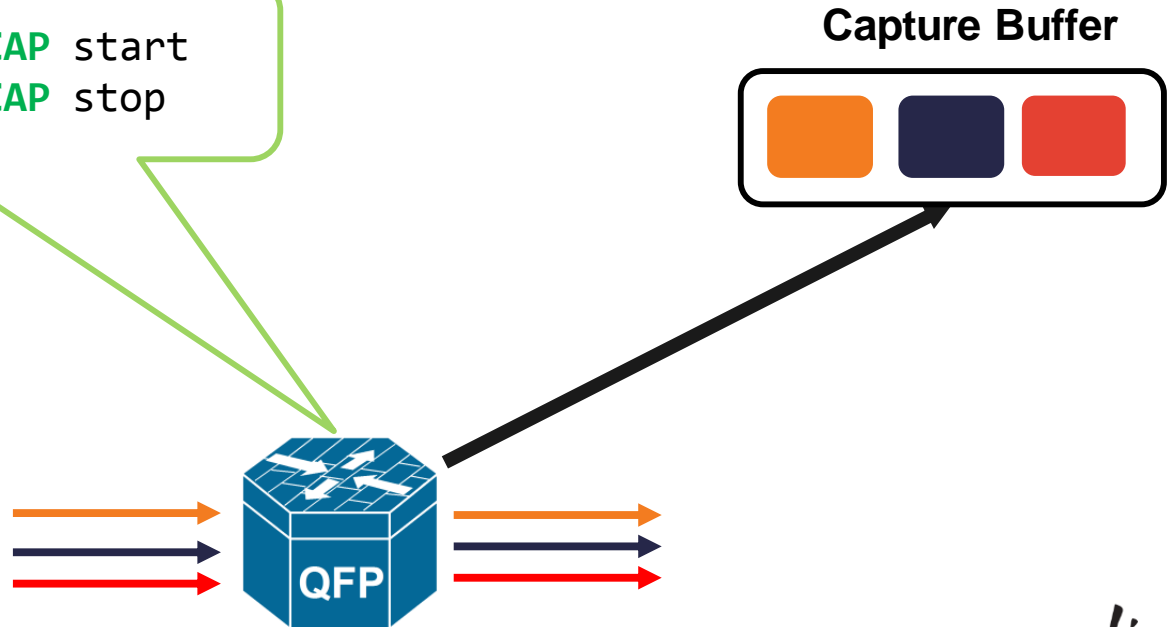
**Capture Buffer**



# EPC Configuration

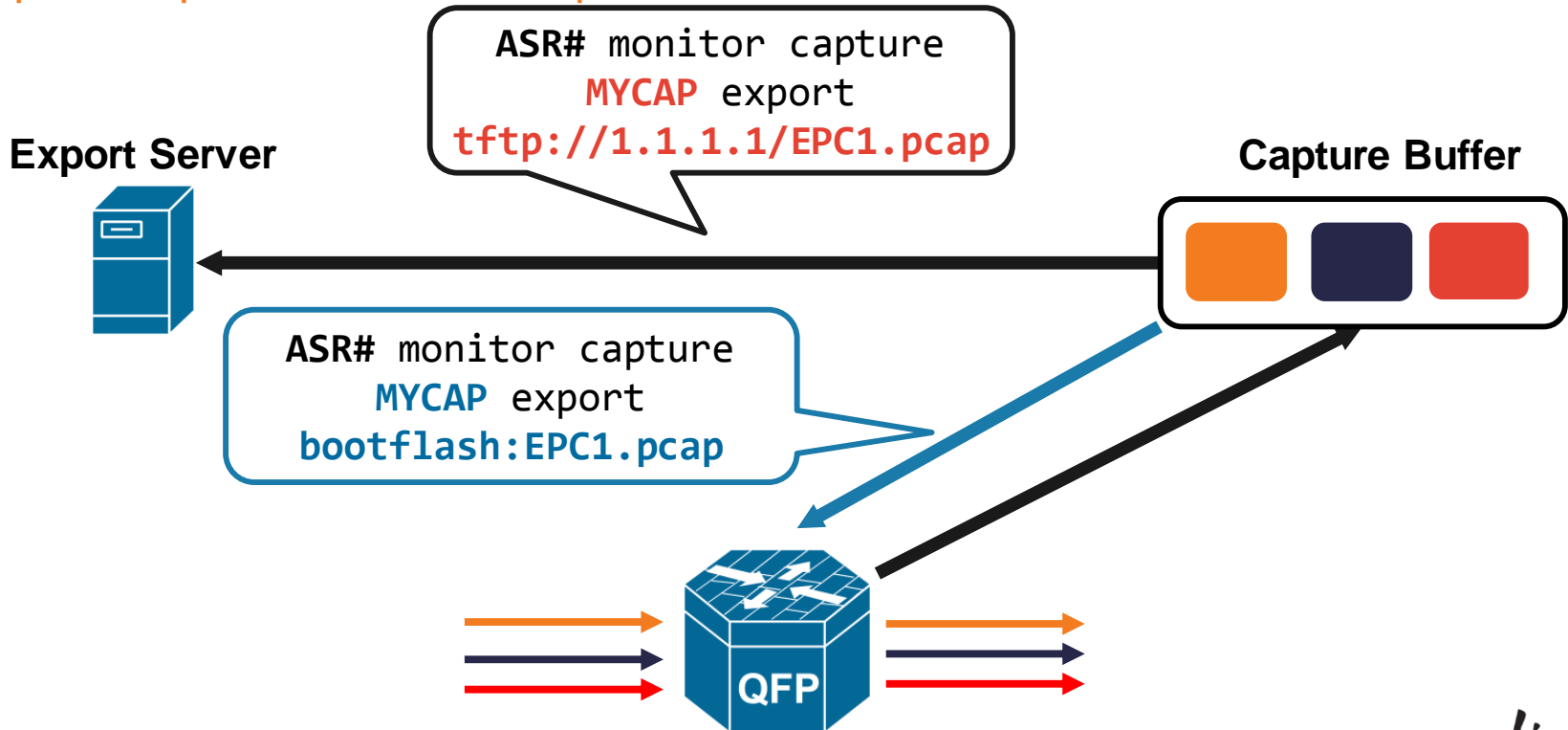
## Step 3 – Run the Capture

```
ASR# monitor capture MYCAP start  
ASR# monitor capture MYCAP stop
```



# EPC Configuration

## Step 4 – Export or View the Capture



# EPC Configuration

## Step 4 – Export or View the Capture



**ASR# show monitor capture CAP parameter**

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

**ASR# show monitor capture CAP buffer**

```
buffer size (KB) : 10240
buffer used (KB) : 128
packets in buf : 5
packets dropped : 0
packets per sec : 1
```

# EPC Configuration

## Step 4 – Export or View the Capture

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



# EPC Configuration

## Step 4 – Export or View the Capture

“show monitor capture **CAP** **buffer dump**” will also display the hexadecimal details only

```
ASR# show monitor capture CAP buffer detail
```

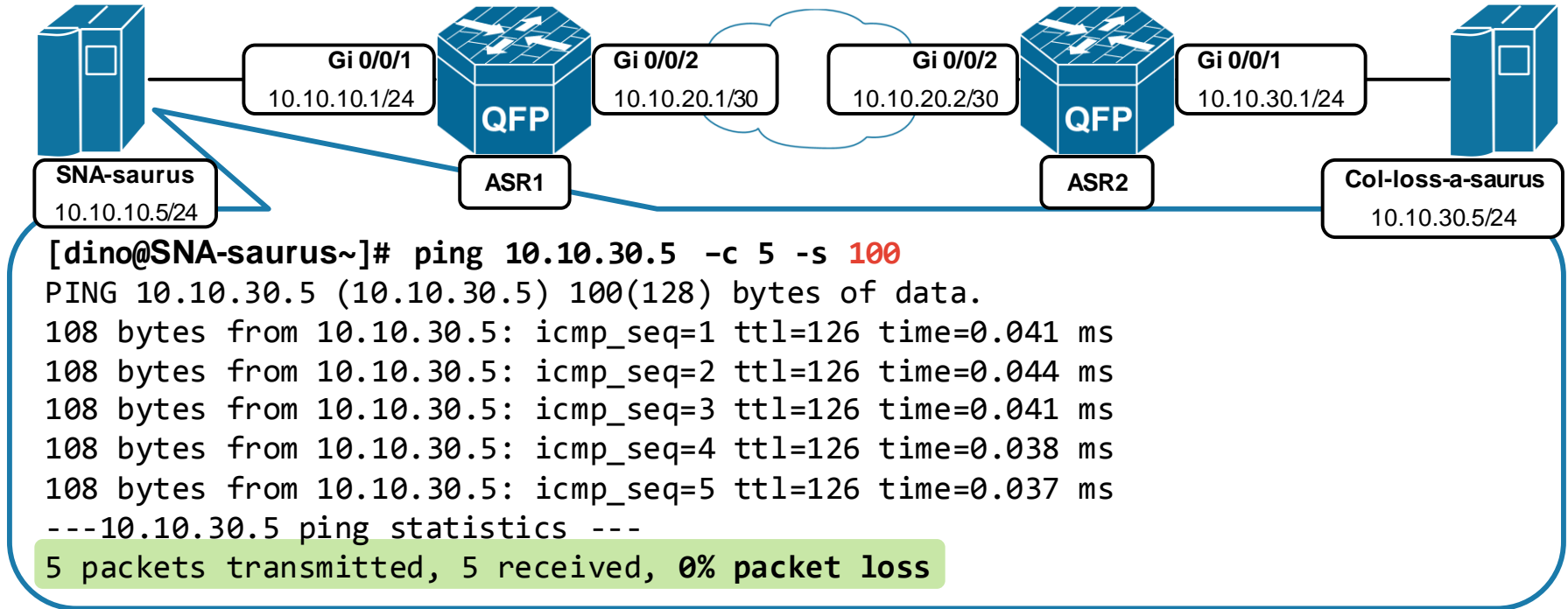
#	size	timestamp	source	destination	protocol
0	114	0.000000	10.254.0.2->	100.100.100.1	ICMP
0000:	0014A8FF	A4020008	E3FFFC28	08004500	.....(..E.
0010:	00649314	0000FF01	551F0AFE	00026464	.d.....U.....dd
0020:	64010800	DF8F0012	00000000	000029E8	d.....).
0030:	74C0ABCD	ABCDABCD	ABCDABCD	ABCDABCD	t.....



# Example – “Isolating Packet Loss”

Assess

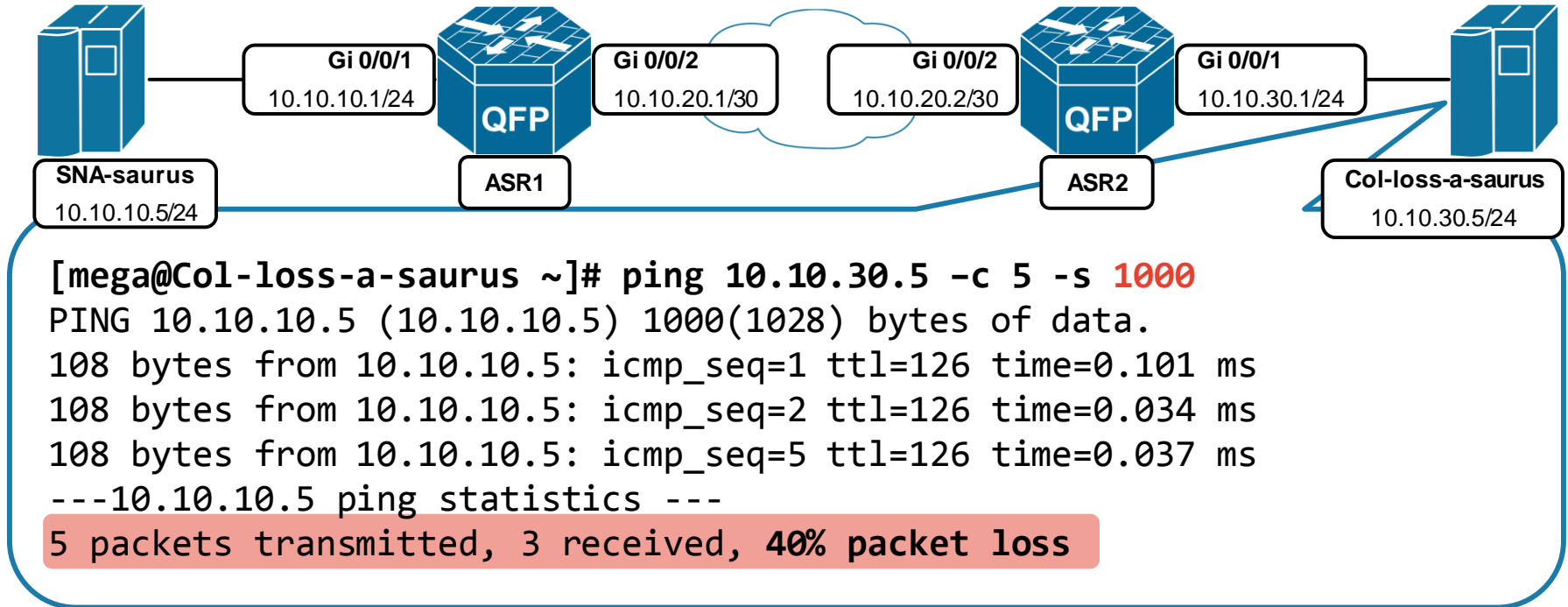
## 1. Assess



# Example – “Isolating Packet Loss”

Assess

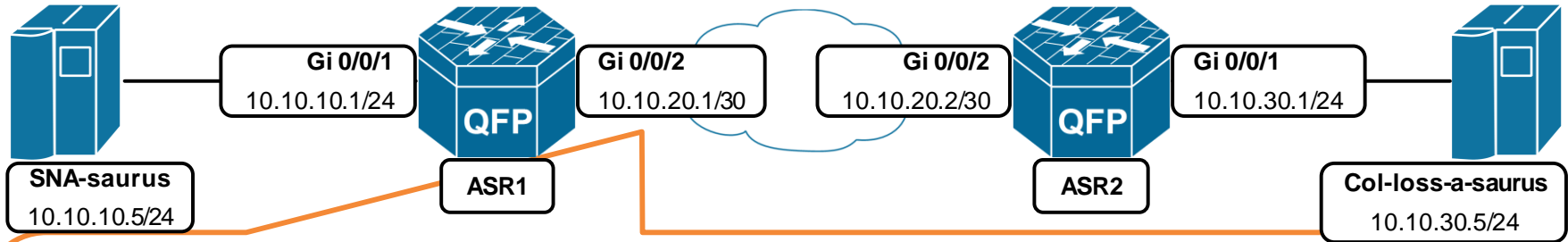
## 1. Assess



# Example – “Isolating Packet Loss”



## 2. Acquire



### Filter ICMP traffic between hosts

```
ip access-list extended ICMP1
permit icmp host 10.10.10.5 host 10.10.30.5
permit icmp host 10.10.30.5 host 10.10.10.5
```

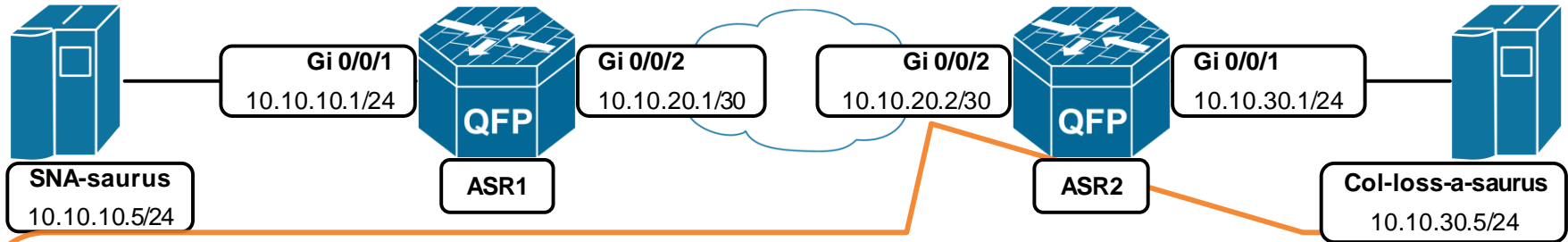
### EPC on ASR1

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

# Example – “Isolating Packet Loss”



## 2. Acquire



### Filter ICMP traffic between hosts

```
ip access-list extended ICMP2
permit icmp host 10.10.10.5 host 10.10.30.5
permit icmp host 10.10.30.5 host 10.10.10.5
```

### EPC on ASR2

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

# Example – “Isolating Packet Loss”



## 3. Analyse

```
ASR1#show monitor capture CAP1 buffer detail | include ICMP|#
```

#	size	timestamp	source	destination	protocol
0	1014	0.000000	10.10.10.5	-> 10.10.30.5	ICMP
1	1014	0.003998	10.10.30.5	-> 10.10.10.5	ICMP
2	1014	0.004012	10.10.10.5	-> 10.10.30.5	ICMP
3	1014	0.004978	10.10.30.5	-> 10.10.10.5	ICMP
4	1014	0.005031	10.10.10.5	-> 10.10.30.5	ICMP
5	1014	2.006832	10.10.10.5	-> 10.10.30.5	ICMP
6	1014	4.007124	10.10.10.5	-> 10.10.30.5	ICMP
7	1014	4.008006	10.10.30.5	-> 10.10.10.5	ICMP

# Example – “Isolating Packet Loss”



## 3. Analyse

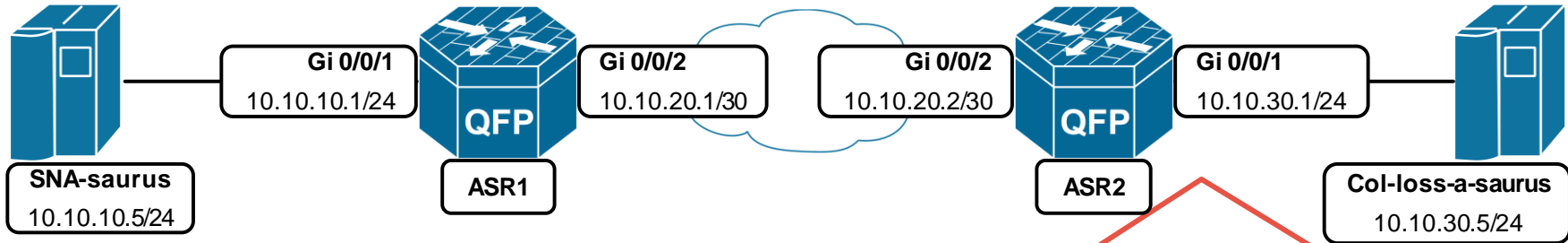
```
ASR2#show monitor capture CAP2 buffer detail | include ICMP|#
```

#	size	timestamp	source	destination	protocol
0	1014	0.000000	10.10.10.5	-> 10.10.30.5	ICMP
1	1014	0.003997	10.10.30.5	-> 10.10.10.5	ICMP
2	1014	0.004013	10.10.10.5	-> 10.10.30.5	ICMP
3	1014	0.004976	10.10.30.5	-> 10.10.10.5	ICMP
4	1014	0.005033	10.10.10.5	-> 10.10.30.5	ICMP
5	1014	2.006834	10.10.10.5	-> 10.10.30.5	ICMP
6	1014	4.007125	10.10.10.5	-> 10.10.30.5	ICMP
7	1014	4.008003	10.10.30.5	-> 10.10.10.5	ICMP

# Example – “Isolating Packet Loss”



## 3. Analyse



### EPC on ASR2

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

# Example – “Isolating Packet Loss”



## 3. Analyse

```
ASR2#show monitor capture CAP2 buffer detail | include ICMP|#
```

#	size	timestamp	source	destination	protocol
0	1014	0.000000	10.10.10.5	-> 10.10.30.5	ICMP
1	1014	0.003814	10.10.30.5	-> 10.10.10.5	ICMP
2	1014	0.003838	10.10.10.5	-> 10.10.30.5	ICMP
3	1014	0.004612	10.10.30.5	-> 10.10.10.5	ICMP
4	1014	4.008062	10.10.10.5	-> 10.10.30.5	ICMP
5	1014	4.008822	10.10.30.5	-> 10.10.10.5	ICMP

**ASR2 is dropping packets**

*Gig 0/0/1 never forwards 2 of 5 ICMP Echo Requests,  
so Col-loss-a-saurus never generates a response.*

# Example – “Isolating Packet Loss”



## 3. Analyse

```
ASR1#ping
Protocol [ip]:
Target IP address: 10.10.20.2
Repeat count [5]: 1
Datagram size [100]:
Timeout in seconds [2]:
Extended commands [n]: y
Source address or interface: 10.10.20.1
Type of service [0]:
Set DF bit in IP header? [no]:
Validate reply data? [no]:
Data pattern [0xABCD]:
Loose, Strict, Record, Timestamp, Verbose[none]:
Sweep range of sizes [n]: y
Sweep min size [36]:
Sweep max size [18024]: 1200
Sweep interval [1]:
```

## Example – “Isolating Packet Loss”

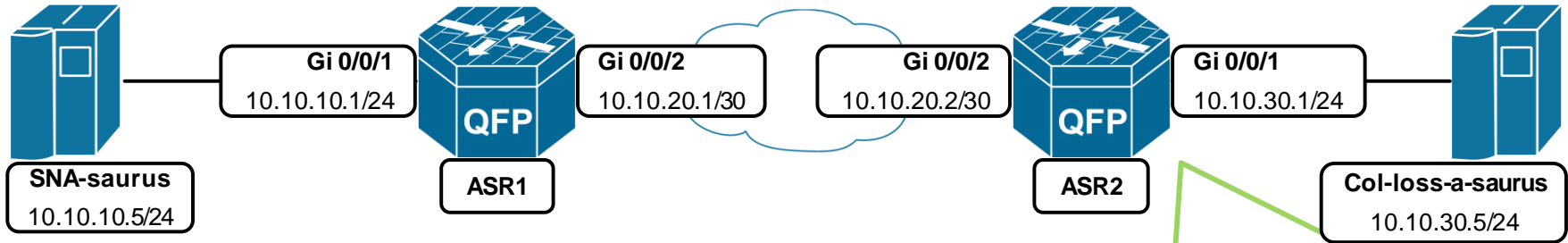
Type escape sequence to abort.

[illegible]

# Example – “Isolating Packet Loss”



## 4. Act



**New focus for troubleshooting**

*Continue to follow AAAA to further isolate root cause.*

# Choose and Use

## EPC

- Packet captures “almost” never lie
- Is this a situation where another tool is better?
- Key advantage of EPC is the “brief” output
- Sometimes you’ll need to see the whole payload

***“A packet capture is like a microscope ...  
we don’t hunt big game with a microscope!”***



# EPC and MPA – General Feature Support



- **IOS-XE**

- 3.7S +
  - ASR 1000 (RP1 or RP2)
  - ASR 1001
  - ASR 1002-X
- 3.8S +
  - ISR 4451-X
- 3.9S +
  - CSR 1000V
- 3.13S +
  - ASR 1001-X

- **IOS**

- 12.2SY +
  - Catalyst 6500, 6800 (with Supervisor 2T)
- 15.1SY +
  - Catalyst 6880-X



For more details on this feature across different platforms and software releases, please use Cisco's Feature Navigator tool available at:

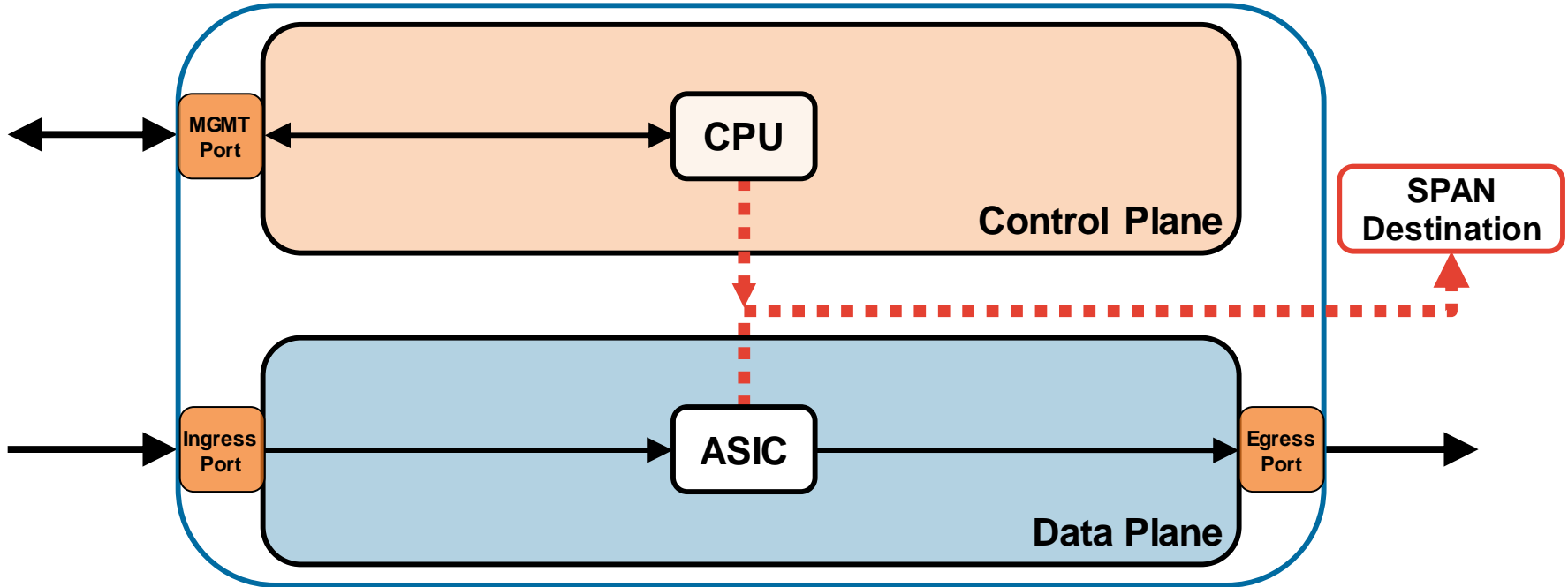
<http://www.cisco.com/go/fn>



## 4. ERSPAN

# ERSPAN / RSPAN / SPAN

## Platform High-Level



# ERSPAN



## Why use ERSPAN?

- SPANs are key in application-level troubleshooting (especially when EPC or other tools are not available)
- Also useful in situations where you have a need have a full traffic “record”
- ERSPAN has significant advantages over SPAN or RSPAN:
  - **Flexibility** - Routable L3 GRE encapsulation (reduction in need for L2 infrastructure to support RSPAN and you don't need to be on-site)
  - **Accuracy** – ERSPAN Type III can support hardware time stamping at point of encapsulation at the SPAN source
  - **Granularity** – ACLs, FSPAN and a number of NX-OS features address how to span high bandwidth interfaces or vlans (10G, 40G etc)
- Be aware of the platform's limitations for SPAN sessions.

# ERSPAN

## Flexibility – Topology Challenges with SPAN and RSPAN



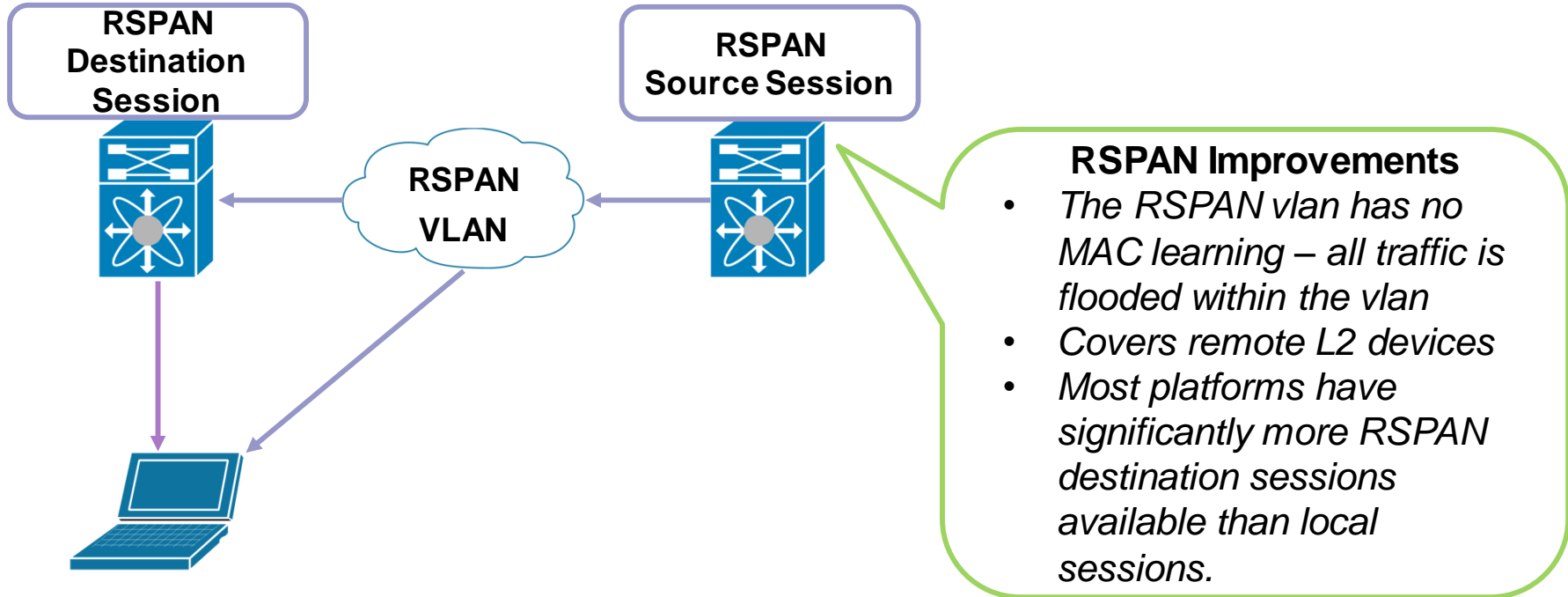
**SPAN  
Source**

### **SPAN Limitations**

- *Destination must be “local” to the SPAN source*
- *Chances are you don’t have a sniffer ready to accept spanned traffic.*

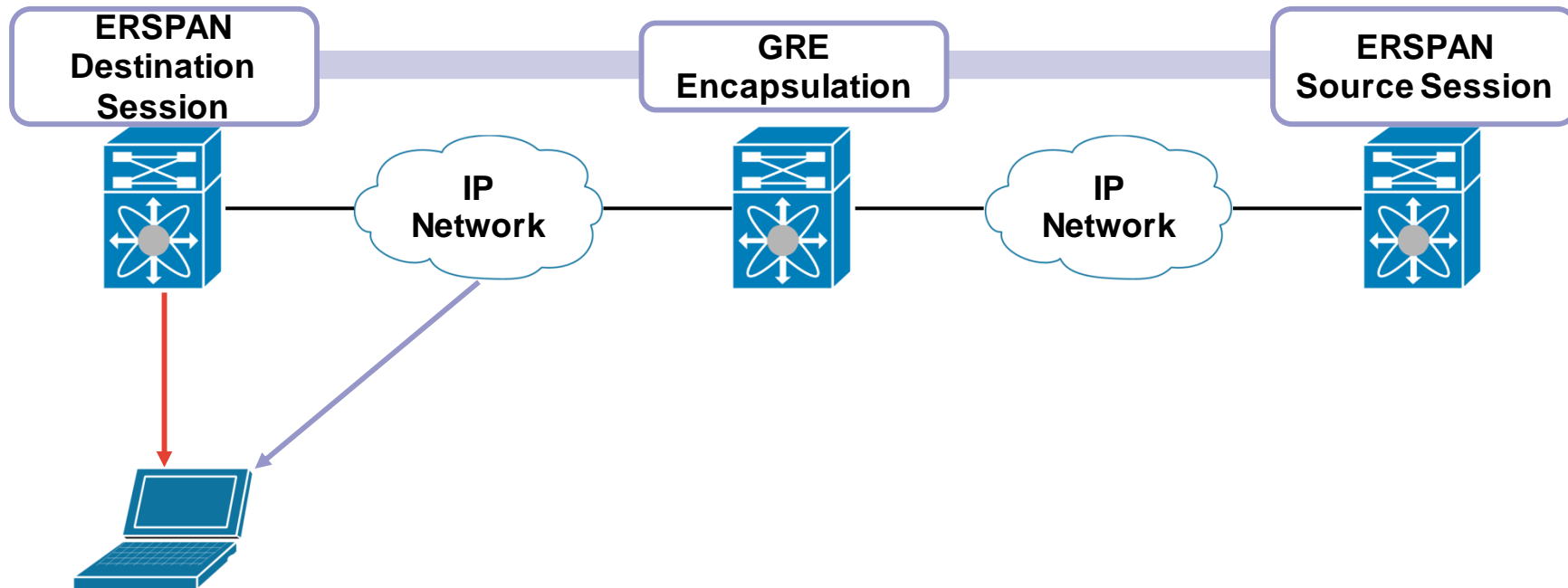
# ERSPAN

## Flexibility – Topology Challenges with SPAN and RSPAN



# ERSPAN

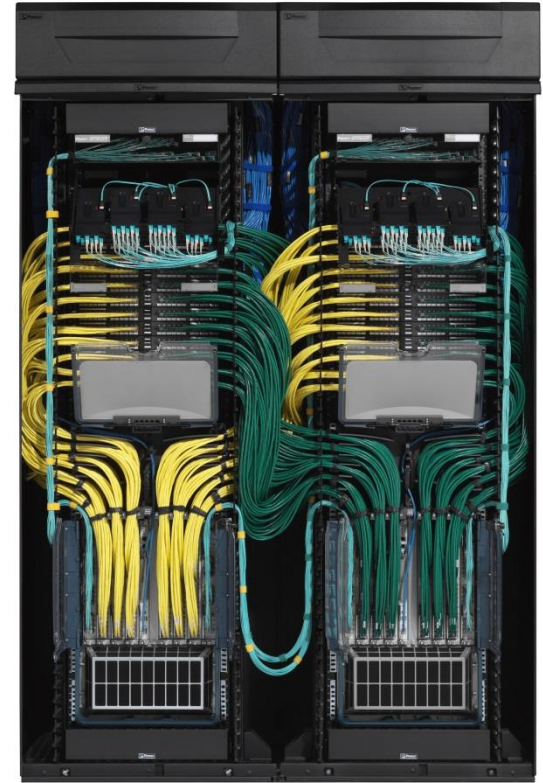
## Flexibility – ERSPAN to the Rescue



# ERSPAN Advantages

## 1. Flexibility

- ERSPAN encapsulation format is GRE-based
  - See [draft-faschiano-erspan-00](#) for ERSPAN header type differences
  - Wireshark understands this encapsulation
- No need to run traditional GRE to enable ERSPAN.



# ERSPAN Advantages

## 2. Accuracy

- ERSPAN Type III supports preservation of hardware time stamping (Precision Time Protocol (**PTP**) level accuracy)
- Very useful for analysing issues where latency or jitter are important
- ERSPAN output is not affected by delay in transporting it to the sniffer.



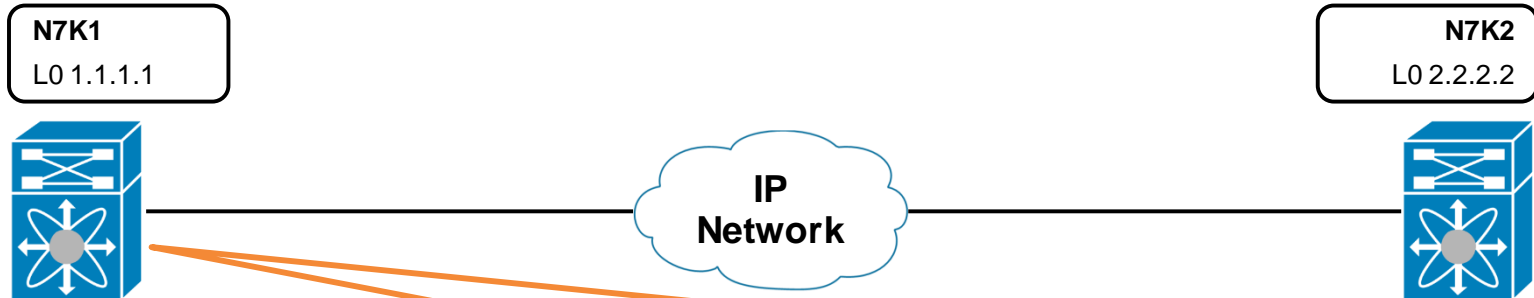
# ERSPAN Advantages

## 3. Granularity

- Flow-based SPAN (**FSPAN**) attaches an ACL directly to a monitor session
- ERSPAN can (and should) be filtered by vlan
- Rule-based SPAN allows extremely granular control over when a packet is spanned:
  - NX-OS 6.2(2)+ supports over 70 L2 to L4 fields
  - Only span when all requirements are met.

# ERSPAN Configuration

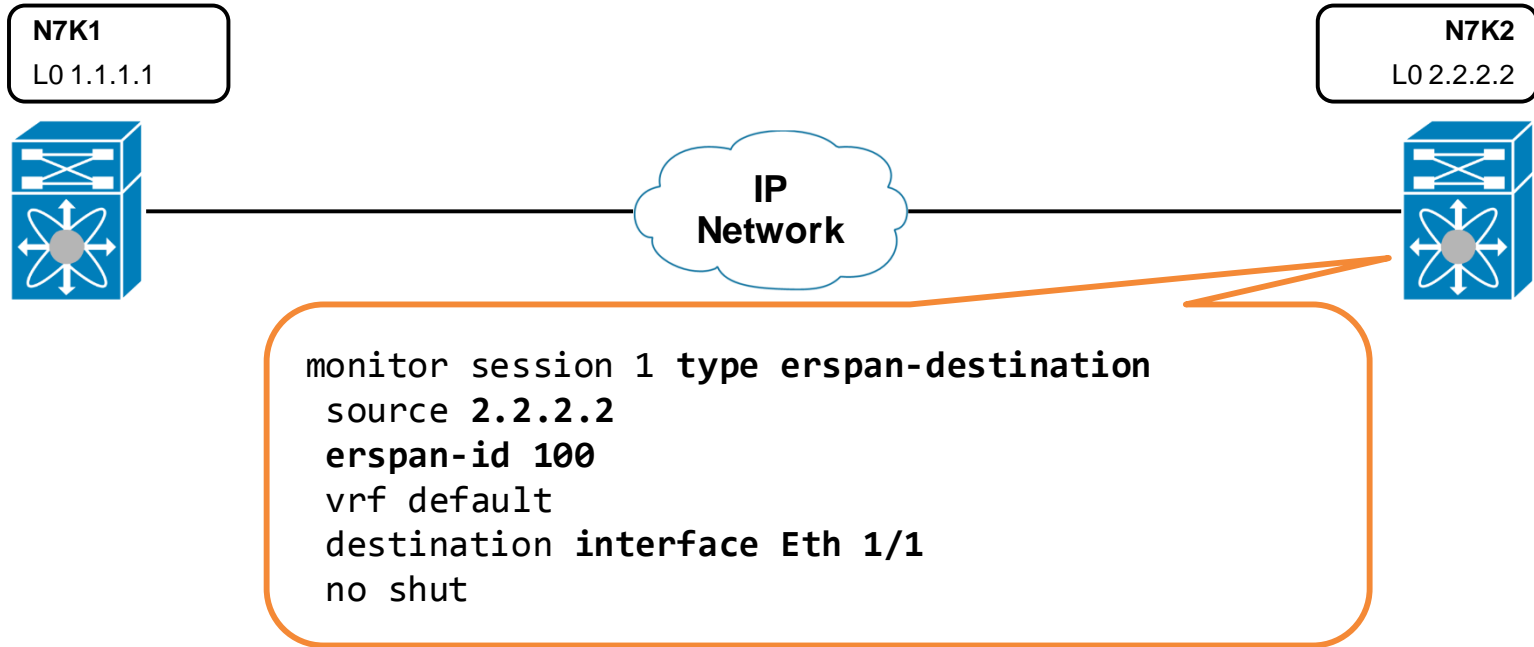
## 1. ERSPAN Source Session



```
monitor session 1 type erspan-source  
  erspan-id 100  
  vrf default  
  destination ip 2.2.2.2  
  source interface Eth 4/2 both  
  filter vlan 20  
  no shut  
monitor erspan origin ip-address 1.1.1.1 global
```

# ERSPAN Configuration

## 2. ERSPAN Destination Session



# ERSPAN Configuration

## 3. Verify

N7K1

L0 1.1.1.1



```
N7K1# show monitor session 1
      session 1
-----
type           : erspan-source
state          : up
erspan-id      : 100
vrf-name       : default
acl-name       : acl-name not specified
ip-ttl         : 255
ip-dscp        : 0
destination-ip : 2.2.2.2
origin-ip      : 1.1.1.1 (global)
source intf    :
  rx           : Eth4/2
  tx           : Eth4/2
  both         : Eth4/2
source VLANs   :
  rx           :
  tx           :
  both         :
filter VLANs   : 20
```

# Example – “Security Audit”

Assess

## 1. Assess

### Security Auditors

*“We need an exact copy of all traffic received by an IP camera in vlan 93 taken from the closest layer 2 device.”*



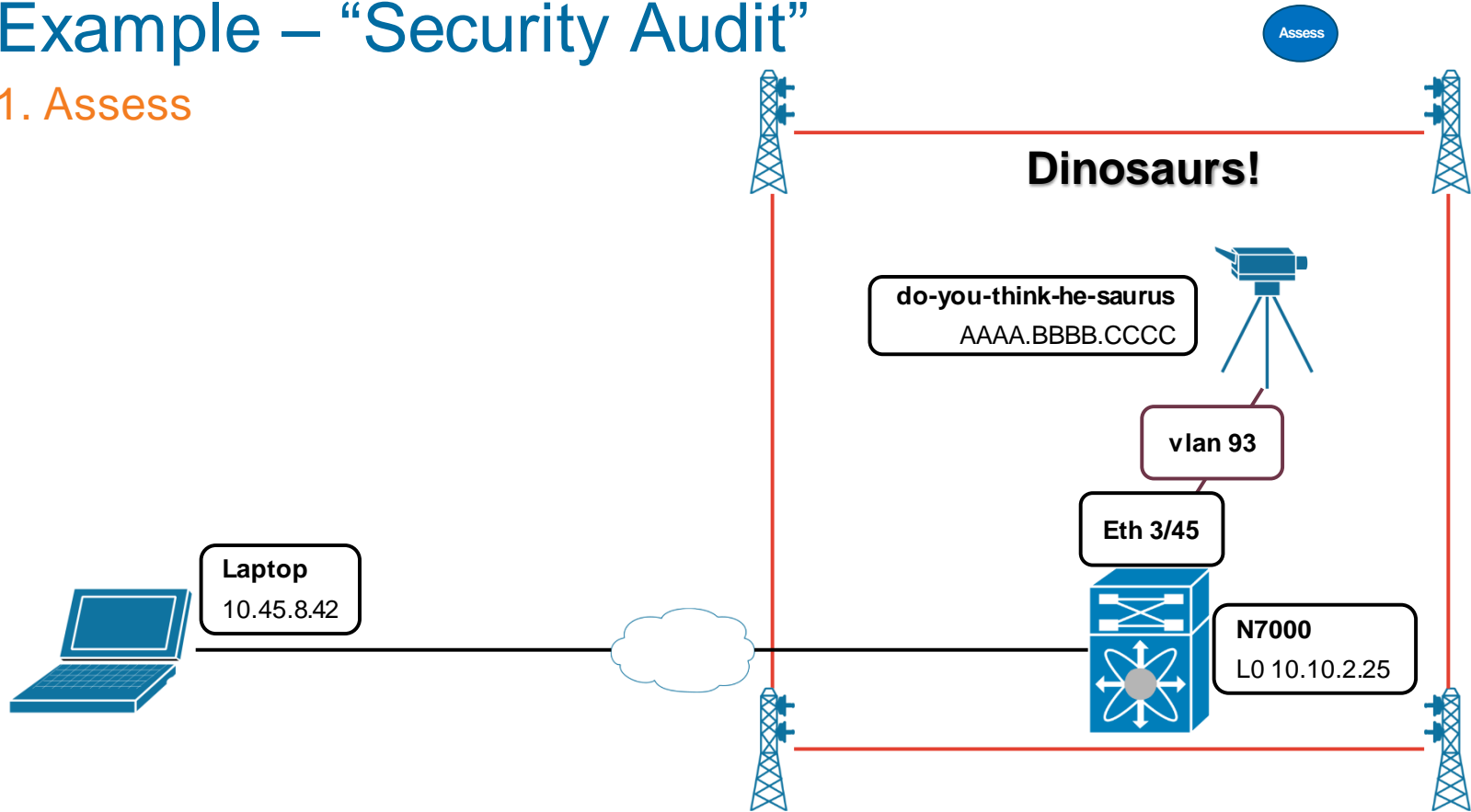
### Management

*“Oh, and they need it this afternoon ...”*



# Example – “Security Audit”

## 1. Assess



# Example – “Security Audit”



## 2. Acquire

### Security Auditors

*“Oh, yeah – yes please !!” :) :)*



### Network Team

*“When you say “exact copy”, does that include the 802.1Q vlan tags?”*



**Management**  
*Huh?*



# Example – “Security Audit”



## 3. Analyse

**SPAN?**

*Have you seen where that switch is?*

**RSPAN?**

*We need this quickly and we have to keep the vlan tags*



**ERSPAN?**

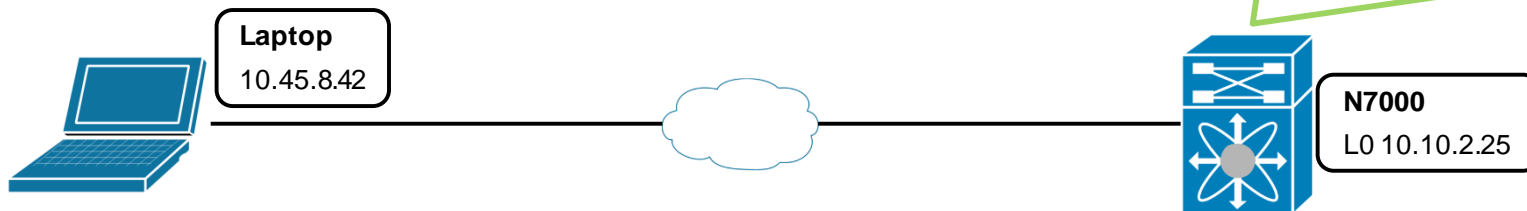
*Let's go!*

# Example – “Security Audit”



## 4. Act

```
monitor session 1 type erspan-source  
erspan-id 100  
vrf default  
destination ip 10.45.8.42  
source interface Eth 3/45 tx  
filter vlan 93  
filter dest-mac AAAA.BBBB.CCCC  
no shut  
monitor erspan origin ip-address 10.10.2.25 global
```



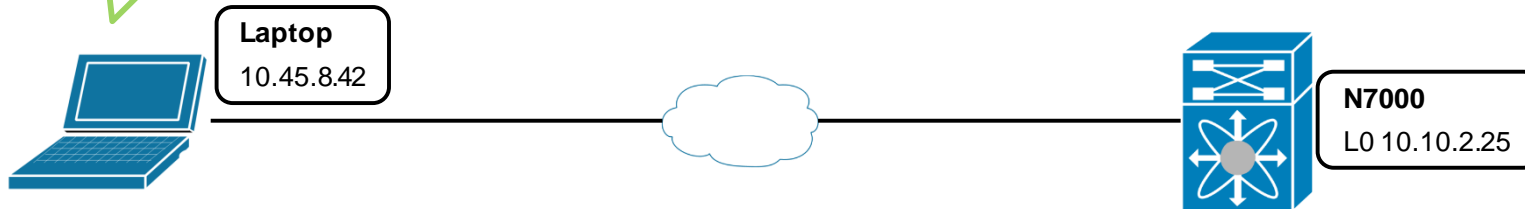
# Example – “Security Audit”



## 4. Act

### Packet Capture Filter

*Capture or display only GRE packets  
(IP Protocol 47, 0x2F)*



# ERSPAN

## Taking it Further

- **Open source tools for processing ERSPAN packet streams:**
  - RDCAP - <http://sourceforge.net/projects/rcdcap/>
  - GULP - <http://staff.washington.edu/corey/gulp/>
- **Ludicrous Speed!**
  - Libpcap is widely used – but has some limitations (drops at high input and low packet size)
  - PF\_Ring – Kernel Module to capture beyond 1 Gbps
- **Use in Virtualisation environments:**
  - Support in Nexus 1000v
  - Open vSwitch emulates this through SPAN over GRE

# Choose and Use

## ERSPAN

- Sometimes span is the only way; but most of the time, it is not
- SPAN sessions are a limited resource – use them wisely
- ERSPAN on NX-OS has a lot of interesting features that make spanning relevant traffic on 40G+ interfaces practical.



# ERSPAN – General Feature Support



- **IOS-XE**

- 2.1+
  - ASR 1000 (RP1 or RP2)
- 3.2S+
- ASR 1001
- 3.7S+
- ASR 1002-X
- 3.8S +
- ISR 4451-X
- 3.9S +
- CSR 1000V
- 3.13S +
- ISR 4300 and 4400 Series

- **IOS**

- 12.2SY +
  - Catalyst 6500, 6800 (with Supervisor 2T)
- 15.1SY +
  - Catalyst 6880-X



- **NX-OS**

- 5.1 +
  - Nexus 7010, 7018
- 5.2 +
  - Nexus 7009
- 6.1 +
  - Nexus 7004
- 6.2 +
  - Nexus 7700 Series

For more details on this feature across different platforms and software releases, please use Cisco's Feature Navigator tool available at:

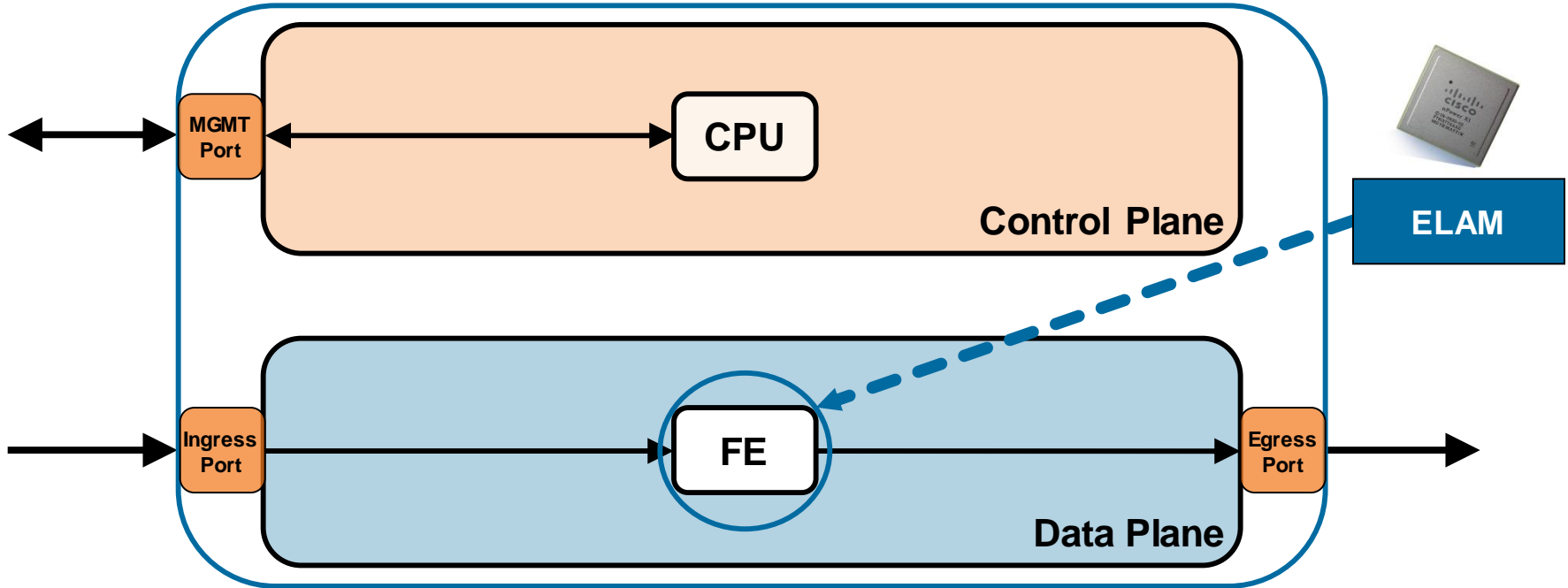
<http://www.cisco.com/go/fn>

A long-exposure photograph of a city street at night. The foreground is filled with vibrant, multi-colored light trails from moving vehicles, creating a sense of motion. In the background, a pedestrian bridge spans the street, and modern buildings with lit windows and signage line the street. The overall scene is a dynamic urban environment.

## 5. ELAM Enhanced

# ELAM

## Platform High-Level



# ELAM Enhanced



## Overview

- Embedded Logic Analyser Module (ELAM) is an engineering tool that is used to look inside Cisco ASICs. The main drawbacks for use are a lack of official support and the high degree of required architectural knowledge.
- ELAM Enhanced abstracts away the need to understand the specific forwarding engine (**FE**) details and presents the detail on:
  1. Whether the packet was received?
  2. Which interface did it arrive?
  3. What were the contents of the packet's header?
  4. How did the device make a forwarding decision based on that packet?
  5. Whether any alterations occurred?
  6. Where was the packet sent?

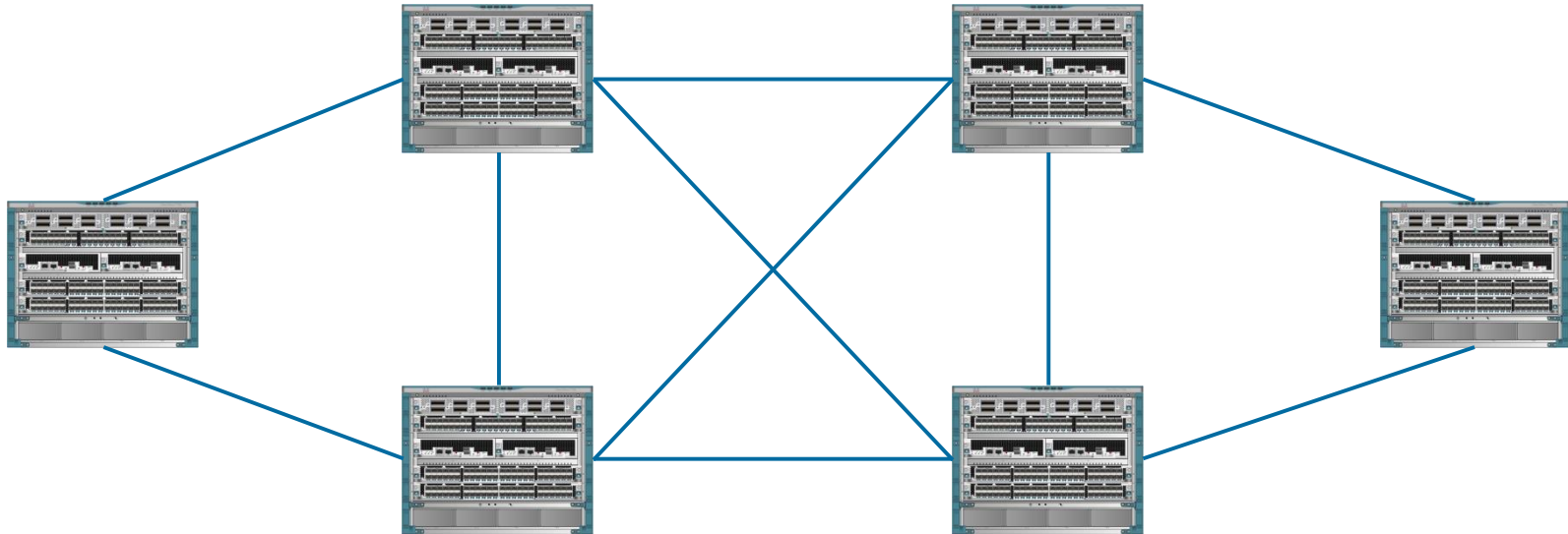
# ELAM Enhanced

## Key Use Cases

### Extent of Failure Domain

*Flooding or replication*

**Forwarding Decisions**  
*Identifying drops, egress interfaces and rewrites.*



# ELAM Enhanced

## Usage

- ELAM looks at the forwarding path of a single packet in a non-intrusive way.
- From NX-OS 6.2(2), it can be invoked for IPv4 packets on F2 and M-series modules using “**source sys/elame.tcl**”
- From 12.2(50)SY on the Catalyst 6500 with a Supervisor 2T, use “**show platform datapath**”.
- Refer to the reference slides from previous presentations of BRKARC 2011 for more detail on the engineering syntax used in full ELAM.

# ELAM



## Full Engineering Capability

- General Discussion

- <https://supportforums.cisco.com/document/9880486/understanding-elam>

- 6500 Platform

- <http://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/116643-technote-product-00.html>
- <http://www.cisco.com/c/en/us/support/docs/switches/catalyst-6500-series-switches/116644-technote-product-00.html>

- Nexus Platform

- <http://www.cisco.com/c/en/us/support/docs/switches/nexus-7000-series-switches/116648-technote-product-00.html>
- <http://www.cisco.com/c/en/us/support/docs/switches/nexus-7000-series-switches/116645-technote-product-00.html>
- <http://www.cisco.com/c/en/us/support/docs/switches/nexus-7000-series-switches/116647-technote-product-00.html>

# ELAM Enhanced

## Concepts

- Forwarding Engine (**FE**) – The ASIC responsible on a module / line card for processing the packet
- Local Target Logic (**LTL**) – An index used to represent a port or group of ports. The source LTL index and the destination LTL index tell us which port the frame was received and where it was sent
- Logical Interface (**LIF**) – An index representation for a logical interface (eg. an SVI)



# ELAM Enhanced

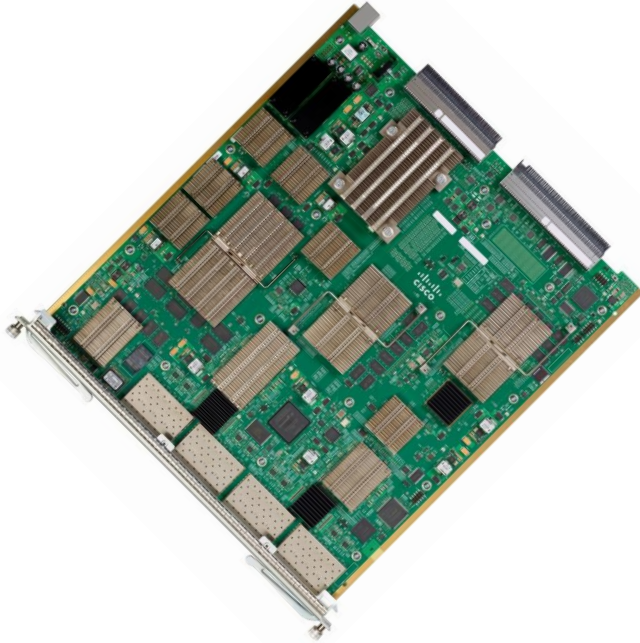
## Concepts

- Data Bus (**DBUS**) – Contains several platform specific internal fields along with the header information from a frame required to make the forwarding decision
- Result Bus (**RBUS**) – Information about the forwarding decision to help determine if the frame was altered and where it was sent.



# ELAM Enhanced

## Concepts



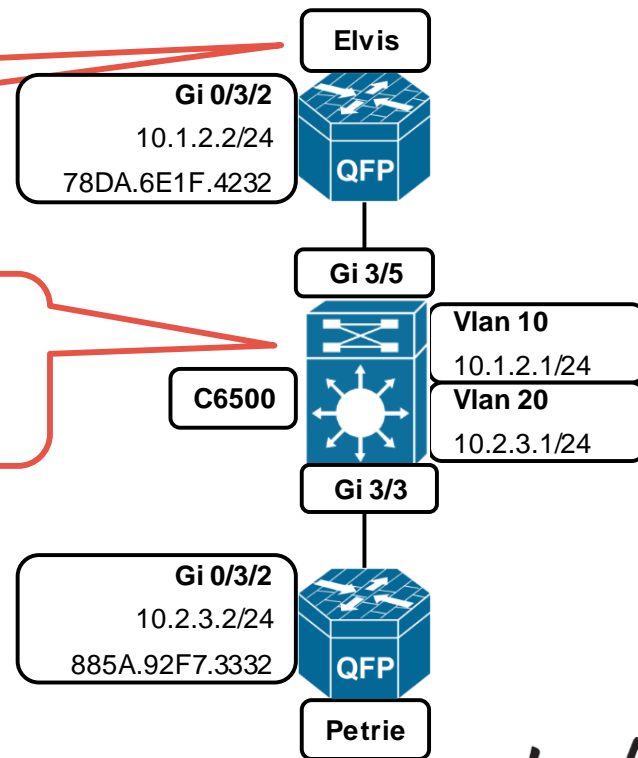
- Input Forwarding Engine (**IFE**) – The ingress ASIC (or “pipeline”) which receives the packet and generates the DBUS query
- Output Forwarding Engine (**OFE**) – The destination that the IFE sends the packet, based on the RBUS response.

# Datapath Trace Example

Catalyst 6500 with Supervisor 2T, DFC4

**Elvis#** telnet 10.2.3.2

**C6500#** show platform datapath vty ingress-  
interface gigabitEthernet 3/5 packet-data  
ipv4 src-address 10.1.2.2



# Datapath Trace Example

## Catalyst 6500 with Supervisor 2T, DFC4

Capturing from **GigabitEthernet3/5 src\_index 132[0x84]**

LTL

Basic Packet Flow

TELNET(6)[len=64]R: 10.1.2.2 -> 10.2.3.2  
Ports: 41986 -> 23 [ACK PSH 0x18] Dscp/Tos 48/0xC0

RouterMAC 6073.5cb3.d180 SMAC 78da.6e1f.4232  
Vlan 10 CoS 0 1q 0

Ingress Lif 0xA Vlan 10  
ILM 0xA Lif\_Sel 1 Lif\_Base 0x80000  
Cpp\_en

Ingress  
LIF

L2

L4, L3 and  
QoS Info

ACL: Permit (Default)  
QoS: Default (Tcam\_Lkup\_Disabled\_by\_Port\_Map)

Packet  
|  
Ttl 255  
|  
V  
Gi3/5[84]  
|  
V  
Ingress  
Features  
V

Flow

# Datapath Trace Example

Catalyst 6500 with Supervisor 2T, DFC4

FIB-L3

|

V

Adjacency

V

EgressLIF

V

Egress

Features

V

Rewrite

|

|

V

Key: 10.2.3.2 [No VPN]  
TCAM[5637] Adj 0x5C000 dgt 0

[FIB] L3\_Enable Dec\_Tt1 ADJ[IP][0x5C000]

0x14 Vlan 20 IpMtu 1518[16] Base 0x0

ACL: Permit (Default)  
QoS: Default (Tcam\_Lkup\_Disabled)

[FIB] L2\_RW[0]: 6073.5cb3.d180 -> 885a.92f7.3332 Dec\_Tt1  
CCC 4  
RIT[0x5C000]

*L3 RBUS /  
Lookup*

*Egress LIF*

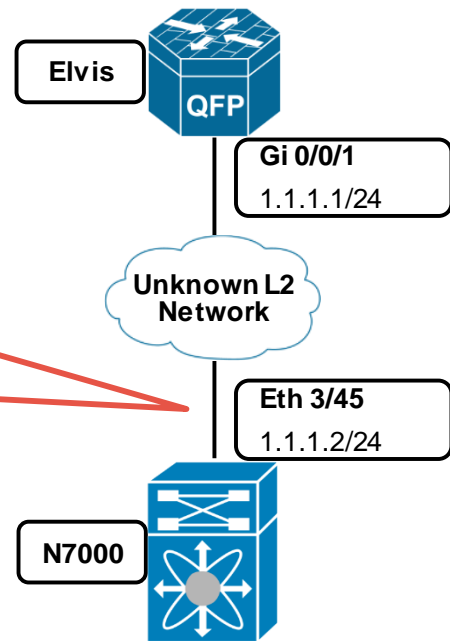
*Rewrite*

# ELAM Enhanced Example

Nexus 7000 (F2 and M1 Series Modules)

*Did we receive OSPF Hellos from Elvis?*

```
N7000# source sys/elame.tcl 1.1.1.1 224.0.0.5
```



# ELAM Enhanced Example

## Nexus 7000 (F2 and M1 Series Modules)

```
N7000# source sys/elame.tcl 1.1.1.1 224.0.0.5 vrf default
elam helper, version 1.015
... source 1.1.1.1, destination 224.0.0.5
```

```
... getting current vdc ... 1
```

```
... vrf default
```

```
... ingress interface derived from source address
```

```
... ingress interface list is Ethernet3/45
```

```
... expanded ingress interface list is Ethernet3/45
```

```
... FE instance list is 3/1/1
```

```
... setting trigger...
```

```
... elam trigger set
```

```
... starting capture...
```

```
... elam capture started
```

```
... no packet captured so far
```

```
press [enter] when packets in question are known to have been sent (ctrl-c
to exit)
```

*Automatically  
calculated from  
source and  
destination IPv4  
address*

*Trigger Set*

*Capture Started*

# ELAM Enhanced Example

## Nexus 7000 (F2 and M1 Series Modules)

... packet captured at FE: 3/1/1

*Captured on  
this IFE*

... capture instance 3/1/1 (slot/type/instance)

+++ IPv4 packet: 98 bytes from MAC 78da.6e1f.4234 / IP 1.1.1.1 to MAC 0100.5e00.0005 / IP 224.0.0.5 TTL 1

+++ protocol OSPF

+++ packet received on interface Eth3/45 vlan 0 (source index 0x0005c)

... rbus: ccc 0x0 cap1 0x1 cap2 0x1 flood 0x1 dest\_vlan 0 dest\_index 0x00002  
l2\_fwd 0x0

+++ packet is flooded to BD 2 / vlan 0

... destination index is NOT from L2 table lookup

+++ copy of the packet is sent to CPU

... lamira OFE: rdt 0x0 dest\_index 0x010c7 flood 0x0 l2fwd 0x0 ofe\_drop 0x0

+++ lamira OFE exception(s): CPP\_LIF (0x200000000)

... FE instance 3/1/1 context after analysis: pb2 retrieved

... done

*RBUS*

*DBUS*

*Forwarding and  
Punting to CPU*

# Choose and Use

## ELAM Enhanced

- ELAM can save you time in very complicated setups
- There is a learning curve
- Proving that a packet is received and forwarded
- “show platform datapath” is significantly easier to read than the full ELAM output.



# ELAM Enhanced

## General Platform Availability



- **IOS**

- Catalyst 6500
- Catalyst 6800
- Catalyst 6880-X
- Cisco 7600 Series

- **NX-OS**

- Nexus 6000
- Nexus 7000 and 7700

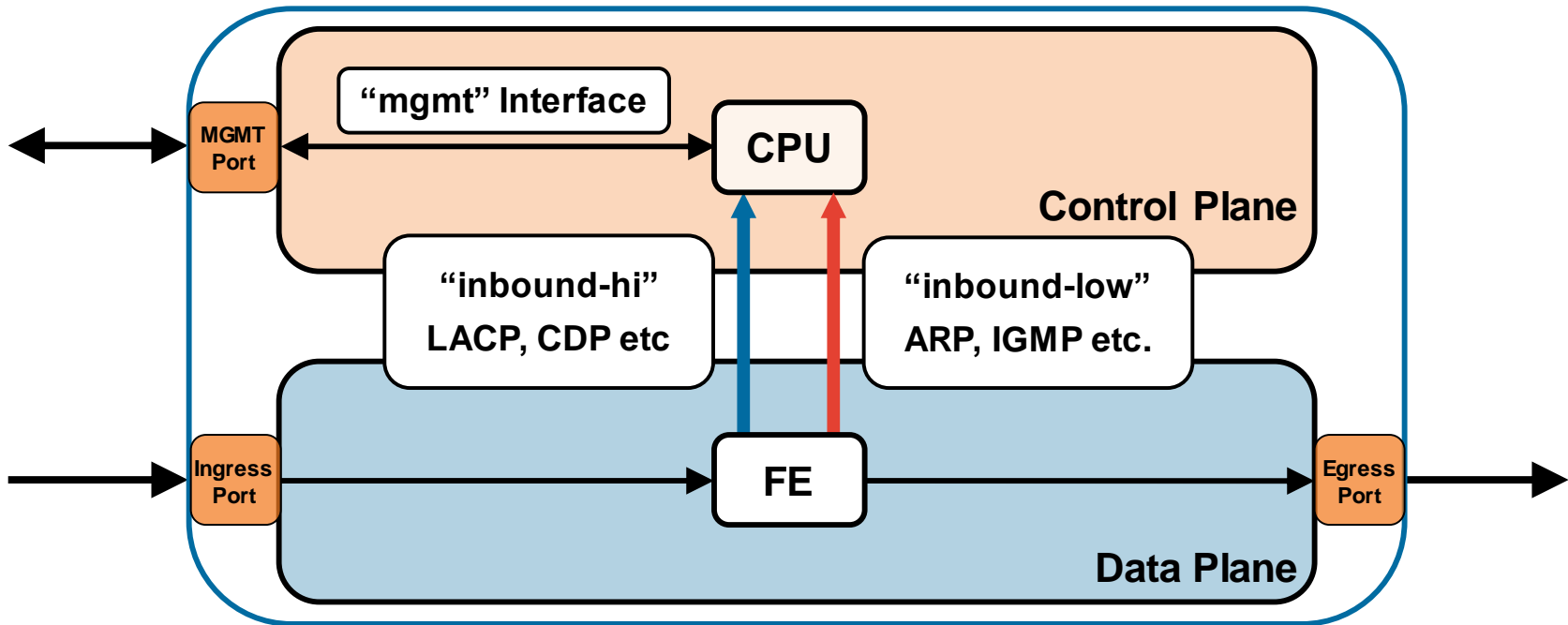




## 5. Ethanalyser and NetDR

# Ethanalyser

## Nexus Platform High-Level



# Ethanalyser and NetDR



## Overview

- Both tools provide visibility into traffic destined for (or “punted to”) the router’s CPU.
- In hardware-based forwarding platforms, the aim is to process the majority of common traffic in hardware to leave the CPU free. However, control plane protocols (eg. OSPF, STP, CDP, LLDP) require the CPU’s involvement.
- **Ethanalyser** – A Tshark implementation for NX-OS.
- **NetDR** (NetDriver) – Supports a non-intrusive debug on the Catalyst 6500 and Cisco 7600 platforms of how CPU-destined traffic is handled.

# Ethanalyser



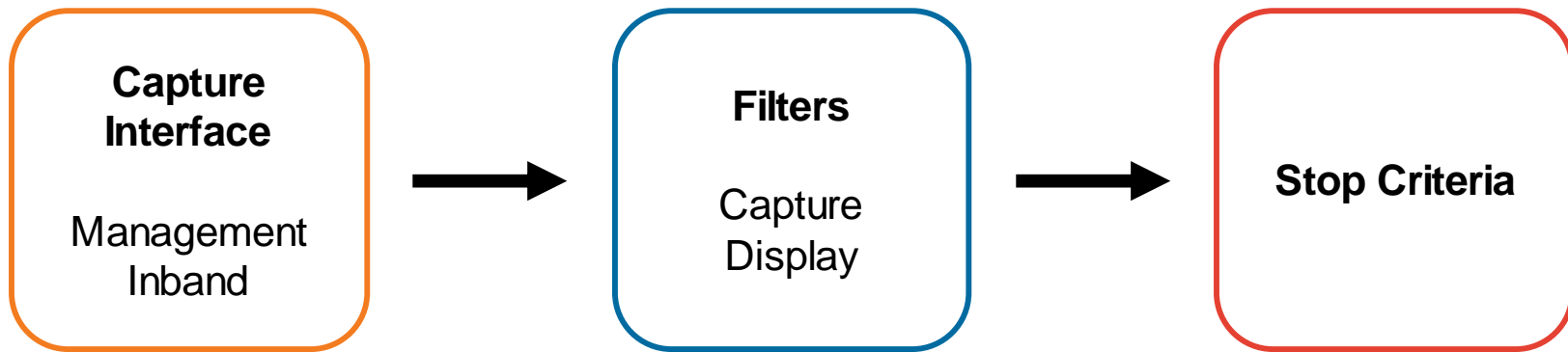
## Nexus Capture Interfaces

- Not all Nexus platforms have two inband interfaces (Nexus 7000 and 9000 see all traffic here from an interface named “**inband**”, with separate queues)
- Internal interface naming also differs between platforms:

Ethanalyser Capture Interface	Internal Physical Interface Name		
	Nexus 3000, 5000	Nexus 7000	Nexus 9000
mgmt	Eth1	Eth0	mgmt
inbound-hi	Eth4	Eth1	inband
inbound-low	Eth3		

# Ethanalyser

## Configuration Flow



- Capture filter applies to the buffer first
- Display filter limits the PCAP contents when viewed locally on the device.

# Ethanalyser

## Filters

- Ideally, use:
  1. A wide **capture filter** (and an appropriately sized threshold) first; and then
  2. Narrow in with a **display filter** (much higher granularity available).
- Applying the reverse can work but you need to have a strong idea of what you are expecting to see.
- Additionally, a capture filter may inadvertently exclude frames that have an internal header applied.
- Filter syntaxes are well documented:
  - <http://wiki.wireshark.org/CaptureFilters>
  - <http://wiki.wireshark.org/DisplayFilters>

# Ethanalyser

## Command Breakdown

**Interface**  
inbound-hi interface

**Display Filter**  
Spanning Tree

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

### Stop Criteria

- *No frame limit (default is 10 if not specified)*
- *Use a circular buffer that writes out 5 x 200KB files to the bootflash:*
- *Stop when five of these files are written to disk.*

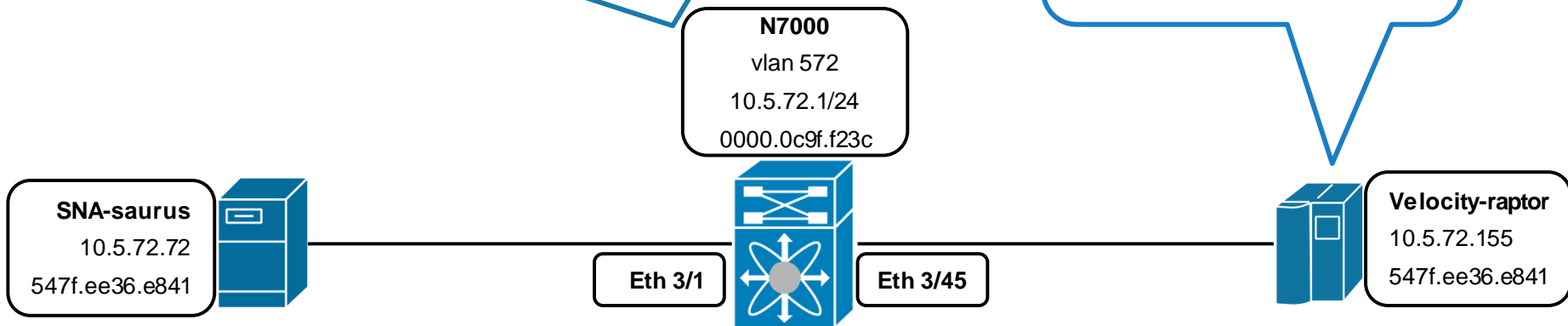
# Nexus Example – Slow File Transfers

Assess

## 1. Assess

**Interface speeds and error rates seem normal**  
*No immediate evidence of traffic loss.*

**scp transfer is slow!**  
*200MB takes over 15 minutes*



# Nexus Example – Slow File Transfers



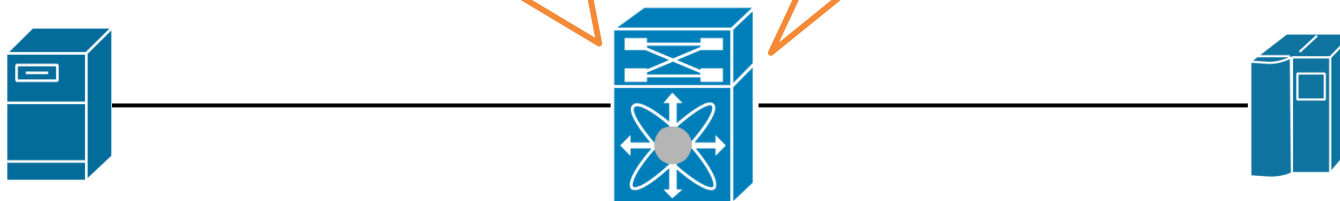
## 2. Acquire

**“Slowness” could suggest software switching**

*The supervisor’s CPU could be processing packets instead of the hardware forwarding engines.*

**Further Verification?**

- *SNMP Graph of CPU Utilisation*
- *CLI Show commands.*



# Nexus Example – Slow File Transfers



## 2. Acquire

```
N7000# show system internal pktmgr internal vdc inband | egrep -v " 0$"
```

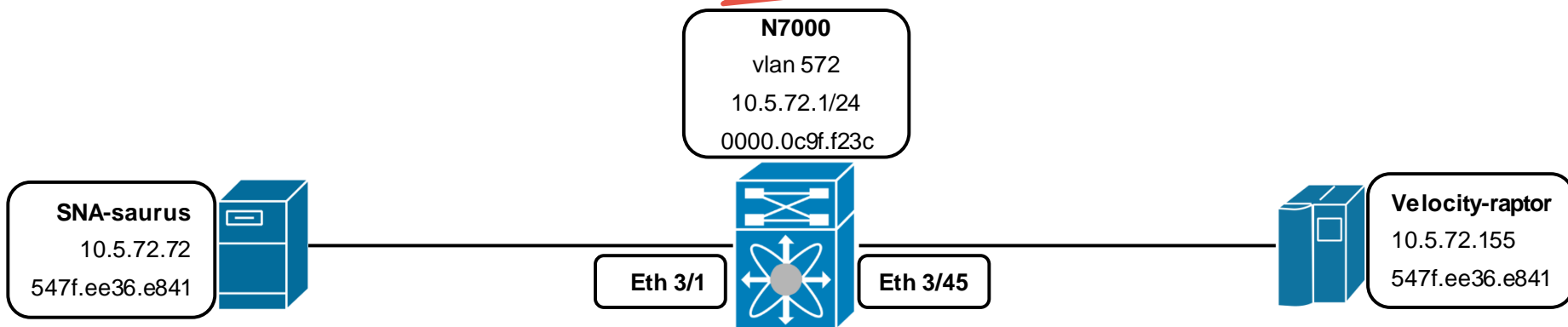
Interface	Src Index	VDC ID	Packet rcvd
Ethernet4/1	0x0	1	15007
Ethernet4/45	0x2c	1	2312
Ethernet4/47	0x2e	1	15001
<b>Ethernet3/1</b>	<b>0x30</b>	<b>1</b>	<b>474567</b>
Ethernet3/45	0x5c	1	2533
Ethernet3/47	0x5e	1	16351

# Nexus Example – Slow File Transfers



## 3. Analyse

```
N7000# ethanalyzer local interface inband capture-filter  
"host 10.5.72.155 or host 10.5.72.72"
```



# Nexus Example – Slow File Transfers



## 3. Analyse

```
N7000# ethanalyzer local interface inband capture-filter "host 10.5.72.155 or  
host 10.5.72.72"
```

Capturing on inband

```
2015-01-26 13:10:24.777838 10.5.72.254 -> 10.5.72.72 ICMP Redirect (Redirect  
for host)
```

```
2015-01-26 13:10:24.777898 10.5.72.72 -> 10.5.72.155 SSH Encrypted response  
packet len=524
```

...

**Why is this traffic being punted at all?**

*This flow is being software switched*

**Why is the supervisor sending ICMP redirect responses?**

*These are going back to SNA-saurus.*

# Nexus Example – Slow File Transfers



## 3. Analyse

```
N7000# ethanalyzer local interface inband capture-filter "host 10.5.72.72"
limit-captured-frames 1 detail | include Ethernet|Internet
Capturing on inband
1 packet captured
Ethernet II, Src: 54:7f:ee:36:e8:41 (54:7f:ee:36:e8:41), Dst: 00:00:0c:9f:f2:3c
(00:00:0c:9f:f2:3c)
Internet Protocol, Src: 10.5.72.72 (10.5.72.72), Dst: 10.5.72.155 (10.5.72.155)
```

### Something isn't right here

*SNA-saurus and Velocity-raptor are in the same vlan, yet SNA-saurus sends frames to the MAC address of N7000's SVI (vlan 572).*

# Nexus Example – Slow File Transfers



## 3. Analyse

```
[dino@SNA-saurus ~]$ nmcli -p -f IP4.ADDRESS device show ens32
```

```
=====
                        Device details (ens32)
=====
IP4.ADDRESS[1]:                ip = 10.5.72.72/25, gw = 10.5.72.1
-----
```

SNA-saurus

10.5.72.72

547f.ee36.e841

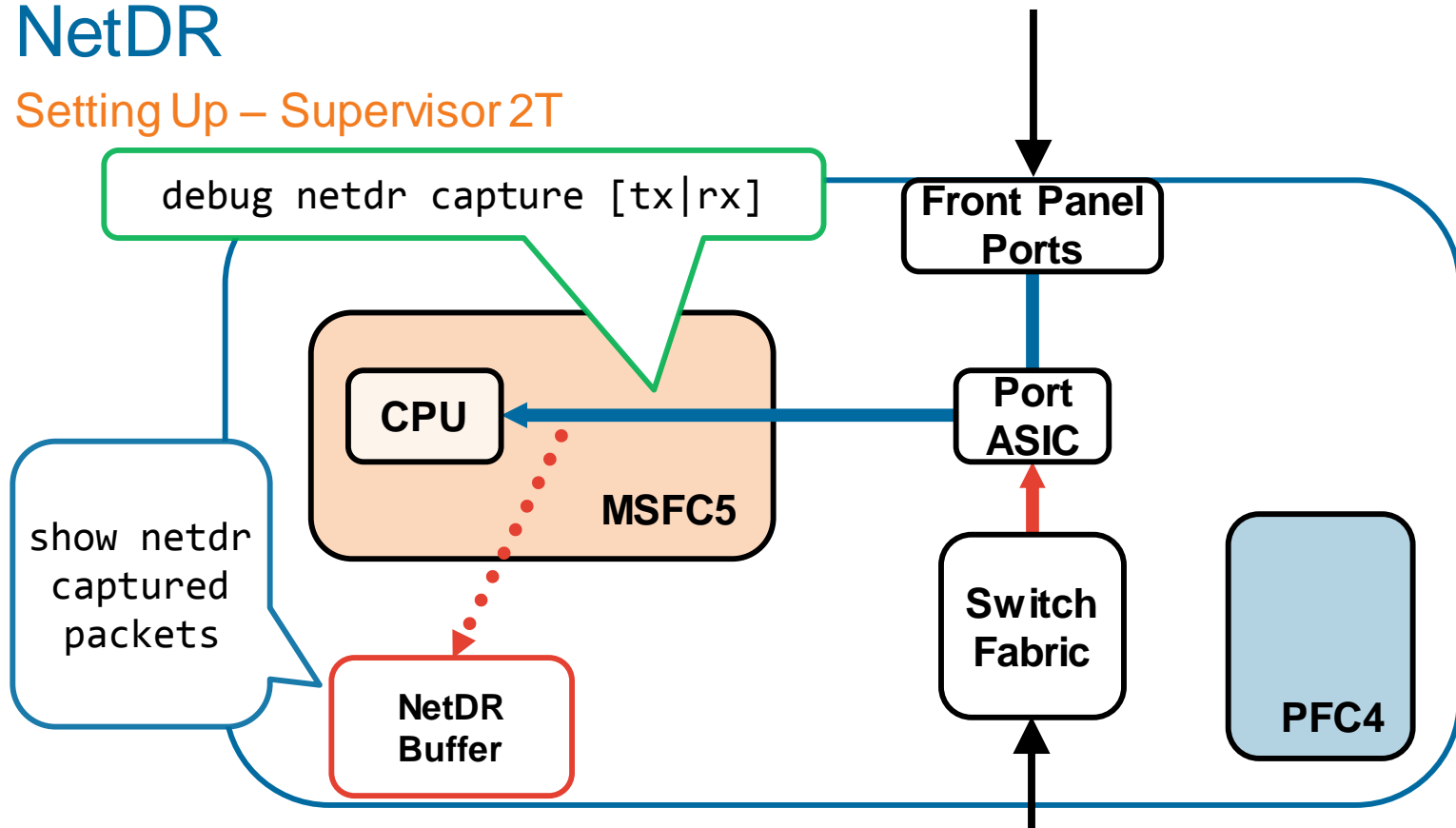


**Incorrect IPv4 subnet mask is the root cause.**

- *Should be changed to /24.*
- *Disable ICMP redirects on N7000?*
- *Would there be unanticipated consequences?*

# NetDR

## Setting Up – Supervisor 2T



# NetDR



## Overview

- NetDR is a debug feature built into the 6500, 6800 and 7600 platforms to capture packets punted to the CPU (includes both RP and SP on older supervisors).
- NetDR can be used even in situations very high CPU from IOS 12.2(33)SXH onwards. The use of the NetDR feature itself though can be expected to add a small amount of CPU consumption.
- It is, however, present from 12.2(18)SXF and might utilise an additional amount of CPU in these much older releases.
- NetDR is a debug feature in IOS, so don't forget to turn it off when you are finished troubleshooting.

# NetDR Output

```
6500#show netdr captured-packets
```

A total of 112 packets have been captured  
The capture buffer wrapped 0 times  
Total capture capacity: 4096 packets

## Buffer Details

*4096 Packet FIFO Buffer*

## Forwarding Information

*L2 / L3 Ingress / Egress  
vlan (Includes Internal)*

```
----- dump of outgoing inband packet -----
```

```
12idb NULL, 13idb Gi1/3, routine etsec_tx_pak, timestamp 08:43:37.798  
dbus info: src_vlan 0x3F4(1012), src_indx 0x380(896), len 0x7A(122)  
  bpu 0, index_dir 0, flood 0, dont_lrn 0, dest_indx 0x0(0), CoS 6  
  cap1 0, cap2 0
```

# NetDR Output

## L2 Information

*Source / Destination MAC*

```
06020000 03F4A800 03800000 7A000000 00000000 00000000 00000000 00000000  
destmac 00.00.0C.07.AC.0A, srcmac 60.73.5C.B3.D1.80, shim ethertype CCF0
```

...

```
ethertype 0800
```

```
protocol ip: version 0x04, hlen 0x05, tos 0xC0, totlen 92, identifier 34548
```

```
df 1, mf 0, fo 0, ttl 255, src 10.66.91.151, dst 10.66.254.33
```

```
tcp src 22, dst 31783, seq 769146961, ack 2924200440, win 3920 off 5  
checksum 0xB0F0 ack psh
```

## L3 / L4 Information

*Source / Destination Port,  
IP Address, TTL, etc.*

# NetDR

## NetDR Parser Tool

- New tool to identify rich detail from NetDR output.
- Features include:
  - Identify “Top Talkers”;
  - Convert Text to PCAP file; and
  - Flow counters.
- Available at:
  - <http://www.cisco.com/c/en/us/support/web/tools-catalog.html>


# NetDR

## NetDR Parser Tool

### PCAP Generation

As an example, use “debug netdr copy-captured disk0:netdr.txt” to copy the entire buffer.

Start Over

 Convert to pcap

#### Top Talkers

##### L2

Source mac	0000.0000.0202	2
Dest mac	FFFF.FFFF.FFFF	2
Protocol	0800	3
L2idb	NULL	6
L3idb	Gi1/3	3
Source vlan	0x3F4(1012)	4
Source index	0x380(896)	3
Dest index	0x808(2056)	2

##### L3

ipv4 source	10.66.91.3	1
ipv4 dest	224.0.0.2	1
ipv4 ttl	1	1
ipv6 source	-	0
ipv6 dest	-	0
ipv6 hoplt	-	0
ipv6 flow	-	0

##### L4

udp	-	1
tcp	-	2
Source port	1985	1
Dest port	1985	1

Clear

#### Flow Results

Click any field in the Top Talkers or Detailed Counters to show all flows that match the selected field.

Max Results: 50 ▼

#### Detailed Counters

Successfully parsed 6 frame(s)

Failed to parse 1 frame(s)

L2 Source Mac (5)

Value	Count
-------	-------

# Ethanalyser and NetDR

## General Platform Availability



- **IOS (NetDR)**
  - 12.2(18)SXF +
    - Catalyst 6500, 6800
    - Cisco 7600 Series
  - 15.1SY +
    - Catalyst 6880-X
- **NX-OS (Ethanalyser)**
  - Present in the first supported NX-OS release for that platform.



For more details on this feature across different platforms and software releases, please use Cisco's Feature Navigator tool available at:

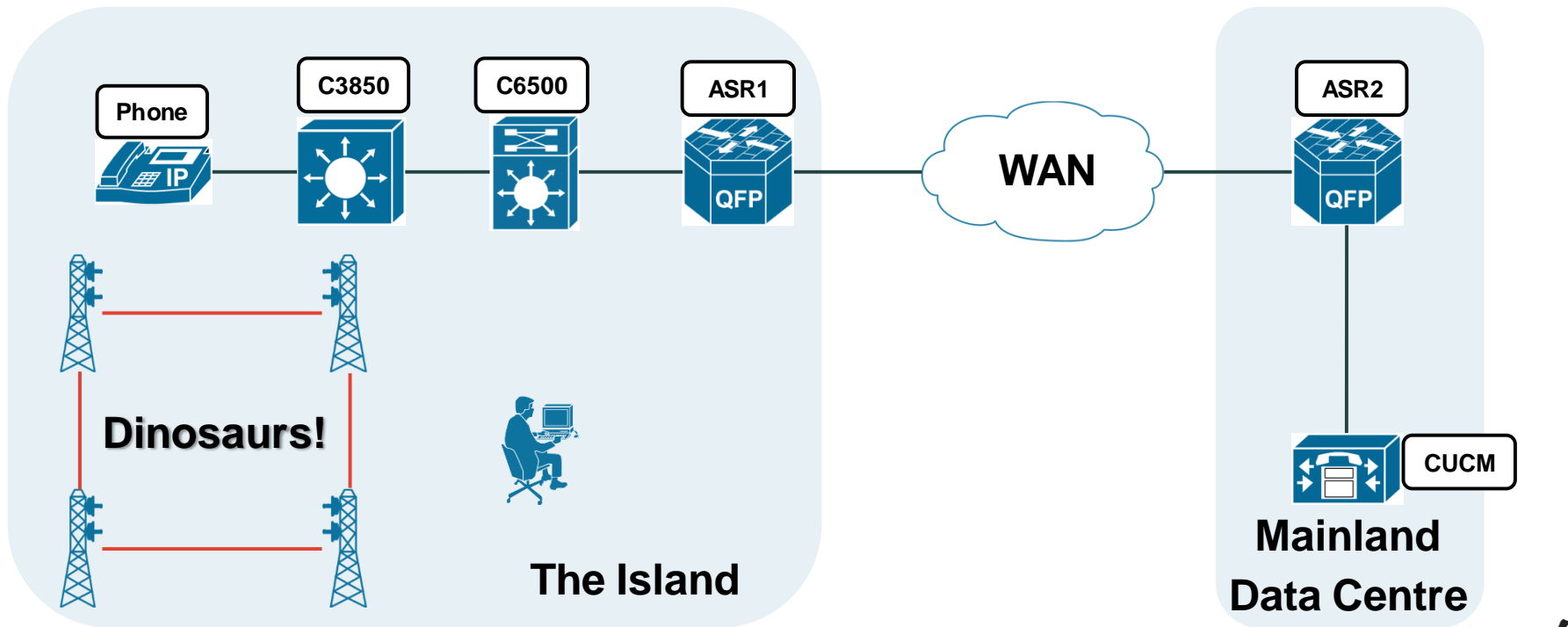
<http://www.cisco.com/go/fn>



6. It's a Jungle Out There!

# It's a Jungle Out There!

“Undocumented Features”



# It's a Jungle Out There!

“Undocumented Features”

## Dinosaurs!



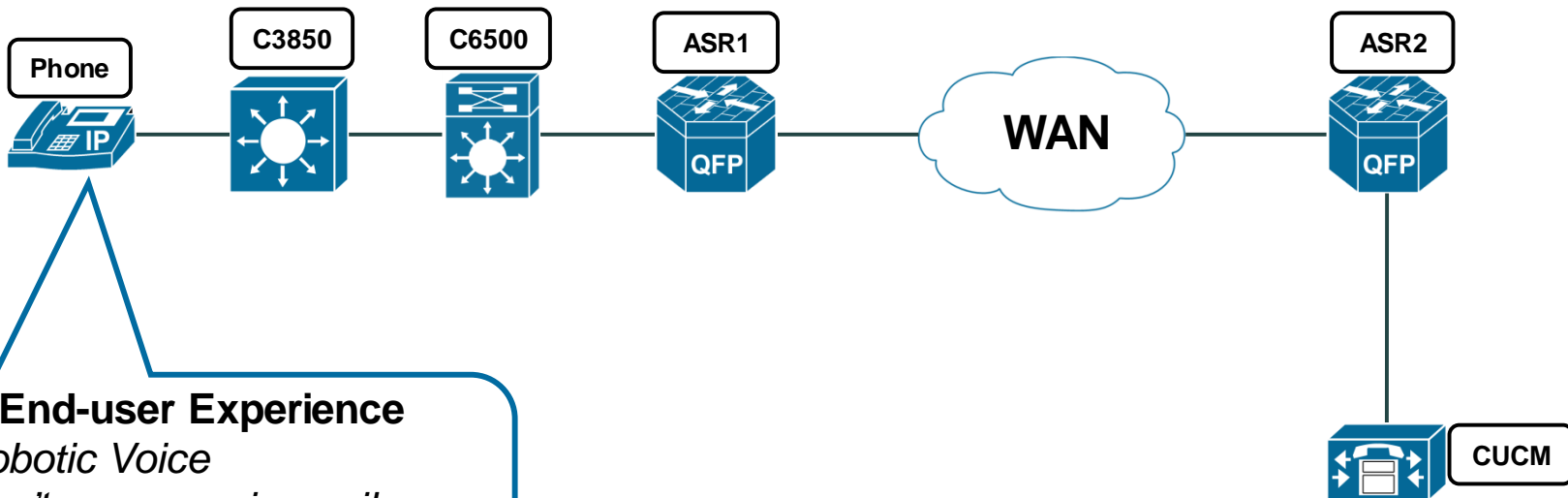
**Application Team**  
*Oh, yeah ... umm ..*  
*We didn't expect that!*



# It's a Jungle Out There!

Assess

## 1. Assess

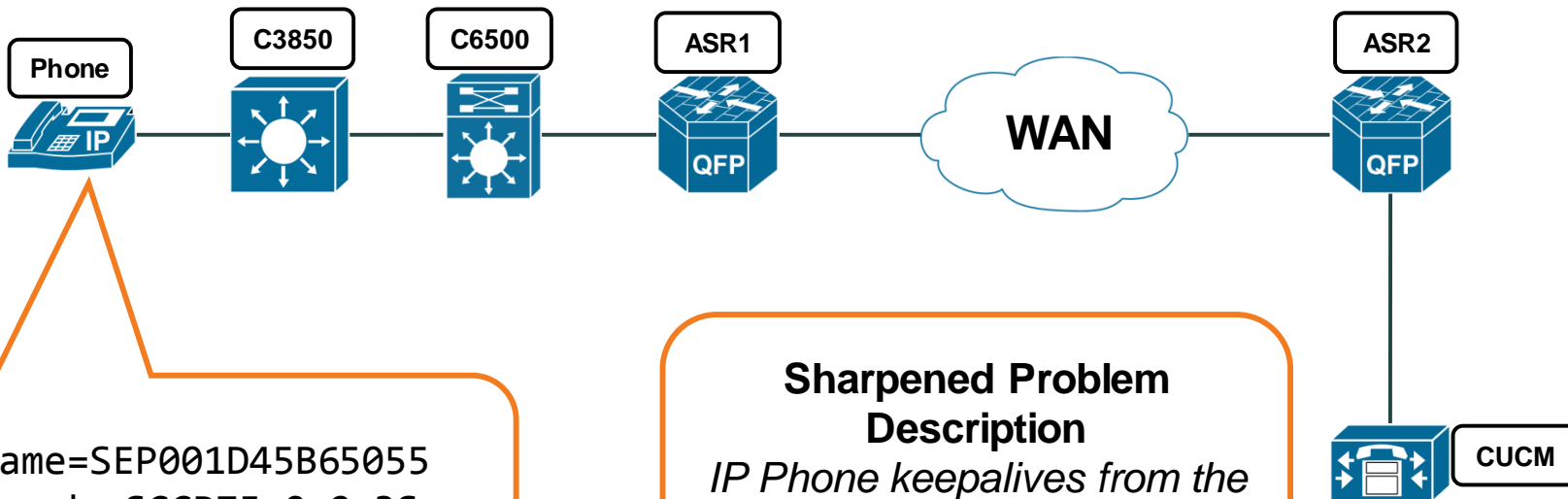


### End-user Experience

- *Robotic Voice*
- *Can't access voicemail*
- *Call Drops*
- *Phone keeps rebooting*

# It's a Jungle Out There!

## 2. Acquire



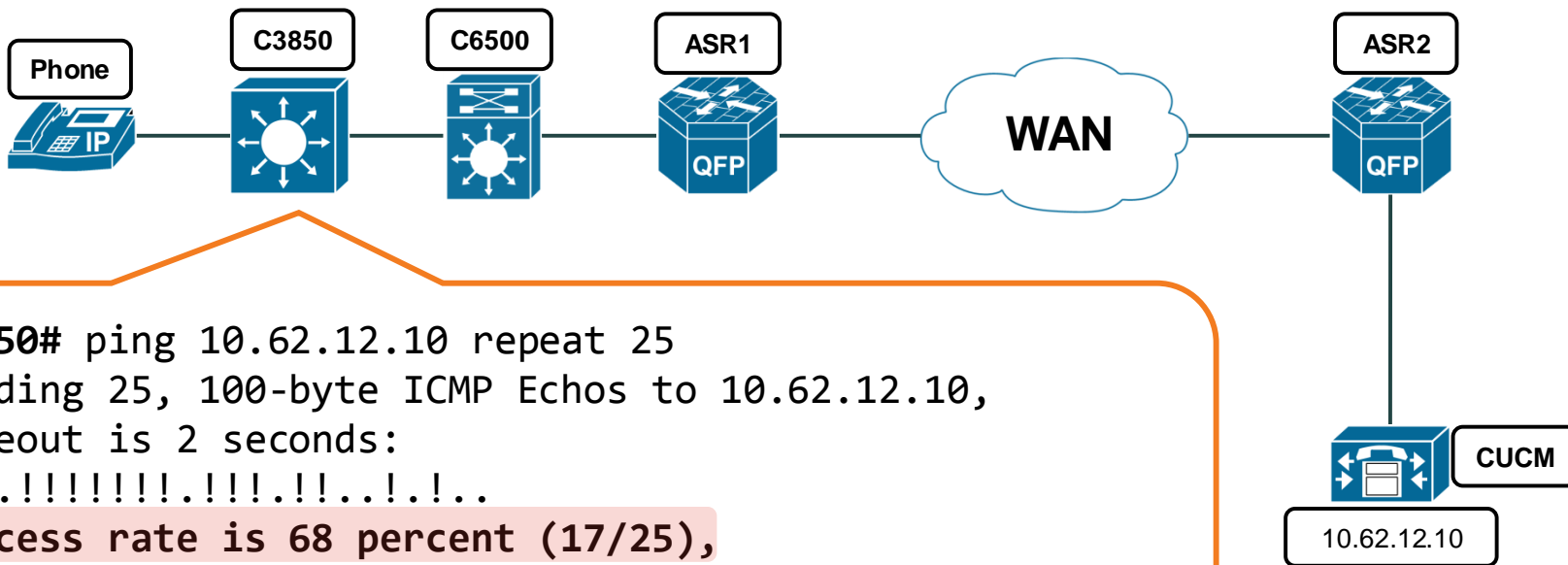
Name=SEP001D45B65055  
Load= SCCP75.9-2-3S  
Last=KeepaliveT0

### Sharpened Problem Description

*IP Phone keepalives from the island to CUCM in the data centre are failing.*

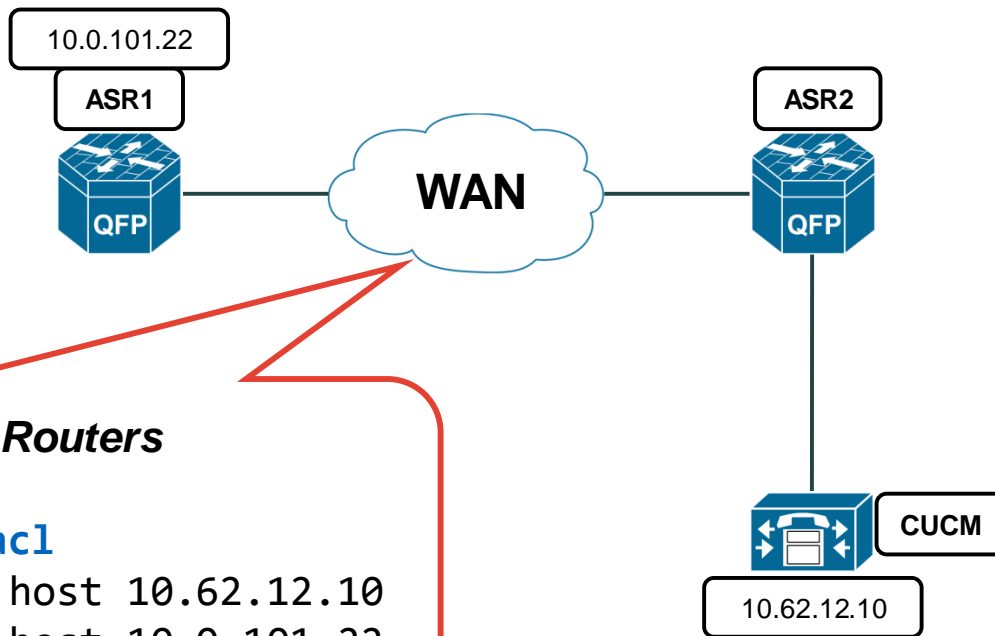
# It's a Jungle Out There!

## 2. Acquire



# It's a Jungle Out There!

## 3. Analyse

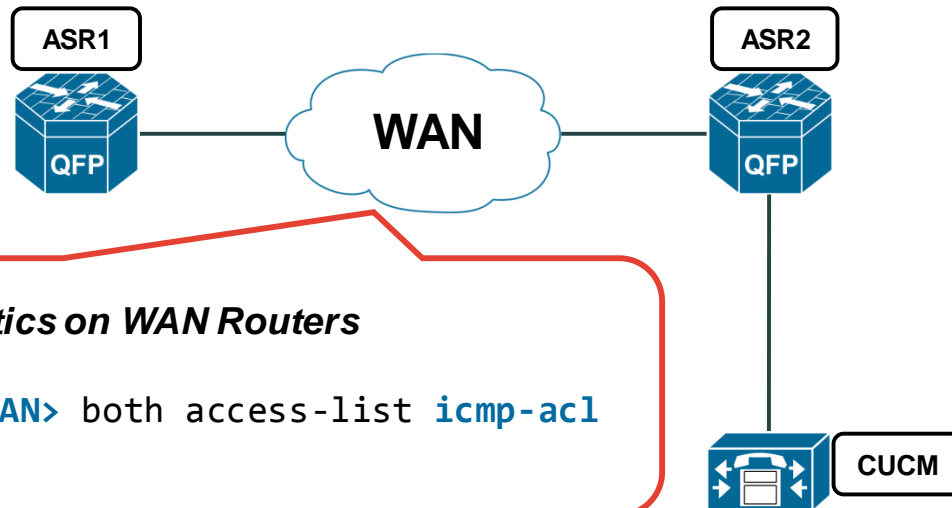


### *Access Lists for WAN Routers*

```
ip access-list extended icmp-acl
 permit icmp host 10.0.101.22 host 10.62.12.10
 permit icmp host 10.62.12.10 host 10.0.101.22
```

# It's a Jungle Out There!

## 3. Analyse



### *EPC to Gather ICMP Statistics on WAN Routers*

```
monitor capture icmpCap interface <WAN> both access-list icmp-acl  
monitor capture icmpCap start
```

# It's a Jungle Out There!



## 3. Analyse

```
ASR1# showmonitor capture icmpCapbuffer | include "packets in buf"  
packets in buf: 42
```

```
ASR2# showmonitor capture icmpCapbuffer | include "packets in buf"  
packets in buf: 34
```

***ASR2 looks to be dropping traffic!***

# It's a Jungle Out There!



## 2. Assess

### Sharpened Problem Description

*Congestion on the WAN link leads to traffic drops.*



```
ASR1# show interfaces <WAN> | include output rate
30 second output rate 99404000 bits/sec, 8983 packets/sec
```

***The WAN's CIR is 100 Mbps, so this is leading to traffic drops***

# It's a Jungle Out There!



## 3. Analyse

### flow record **RECORD**

```
match ipv4 source address  
match ipv4 destination address  
match transport source-port  
match transport destination-port  
match flow direction  
collect interface input  
collect counter packets
```

### flow exporter **EXPORT**

```
destination <collector>  
vrf <vrf>
```



### flow monitor **MONITOR**

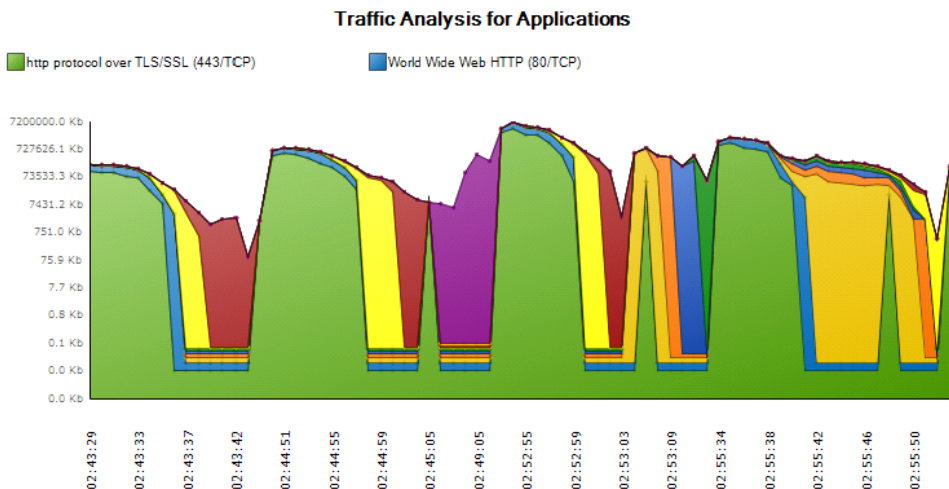
```
exporter EXPORT  
record RECORD
```

### interface <WAN>

```
ipflow monitor MONITOR output
```

# It's a Jungle Out There!

## 3. Analyse

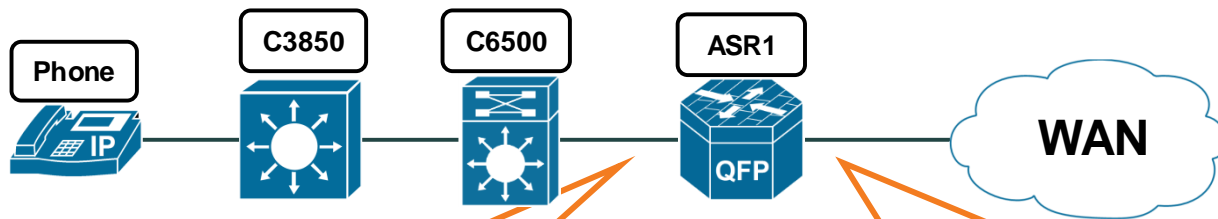


Protocol	Traffic Share
HTTP over SSL (TCP 443)	36%
HTTP (TCP 80)	26%
IMAP (TCP 143)	8%
NETBIOS Name (TCP 137)	7%
LDAP (TCP 389)	7%
NNTP (TCP 119)	7%
SSH (TCP 22)	5%
MySQL (TCP 3306)	3%

# It's a Jungle Out There!



## 2. Assess



### **Congestion is Expected**

- *Verify QoS Policies*

### **Voice QoS on <WAN>**

- *SIGNALLING Class*
- *RTP\_MEDIA Class*

# It's a Jungle Out There!



## 3. Analyse

```
ASR1# show policy-map interface <WAN>
```

```
Service-policy output: OUTBOUND_ISP_100
```

```
Class-map: class-default (match-any)
```

```
113588941 packets, 156012353154 bytes
```

```
(pkts output/bytes output) 113587316/156012107457
```

```
shape (average) cir 100000000, bc 400000, be 400000
```

### CIR Shaping

*This looks ok, what about  
other QoS polices?*

# It's a Jungle Out There!



## 3. Analyse

...

Service-policy : VOIP

Class-map: RTP\_MEDIA (match-all)

0 packets, 0 bytes

Match: dscp ef (46)

Priority: 25% (25000 kbps)

Class-map: SIGNALING (match-all)

0 packets, 0 bytes

Match: dscp cs3 (24) af31 (26)

bandwidth remaining 10%

Class-map: class-default (match-any)

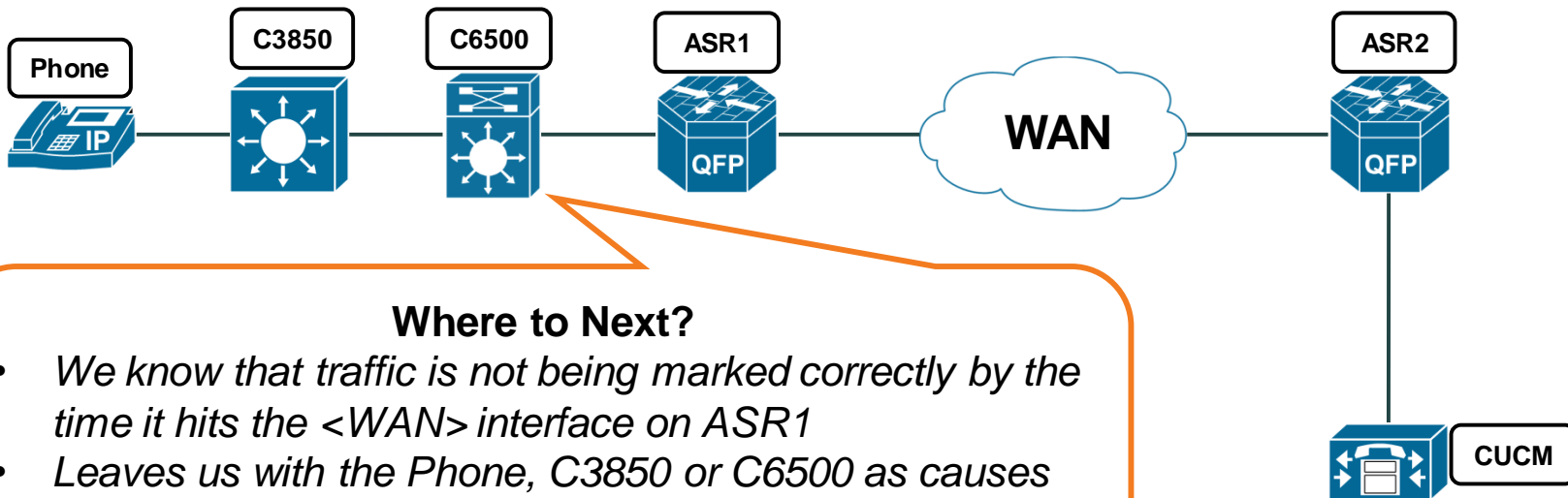
113588922 packets, 156012327106 bytes

### Sharpened Problem Description

*In situations of expected WAN link congestion, high-priority traffic is not being marked correctly into the RTP\_MEDIA or SIGNALLING classes.*

# It's a Jungle Out There!

## 2. Assess

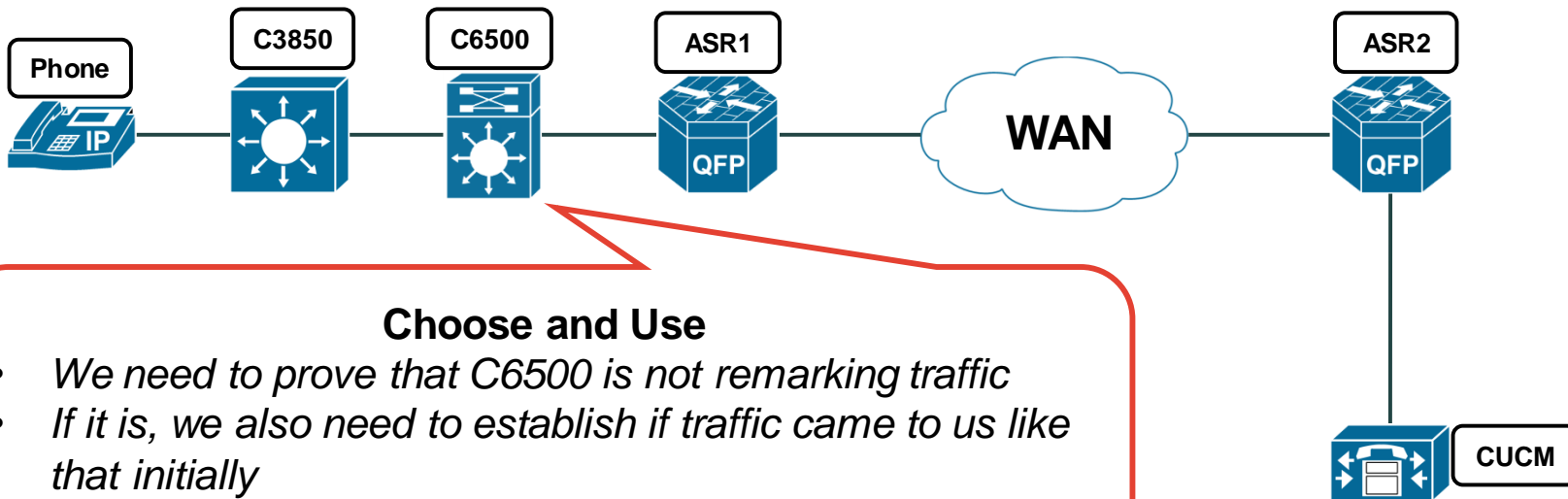


### Where to Next?

- We know that traffic is not being marked correctly by the time it hits the <WAN> interface on ASR1
- Leaves us with the Phone, C3850 or C6500 as causes
- Start with C6500 – Phone is unlikely to operating incorrectly and this helps eliminate a complex element.

# It's a Jungle Out There!

## 3. Analyse

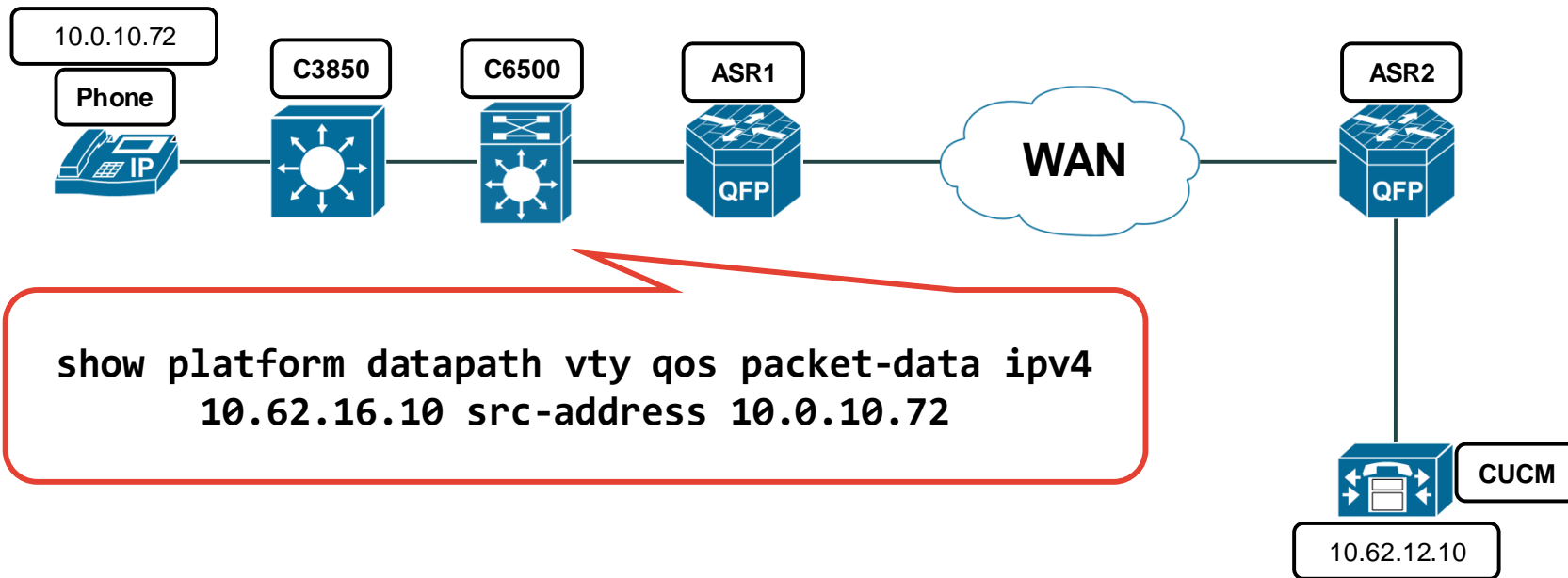


### Choose and Use

- *We need to prove that C6500 is not remarking traffic*
- *If it is, we also need to establish if traffic came to us like that initially*
- *Let's trace the datapath to confirm.*

# It's a Jungle Out There!

## 3. Analyse



# It's a Jungle Out There!



## 3. Analyse

Capturing from GigabitEthernet3/5 src\_index 132[0x84]

```
QoS Packet Flow
-----
Packet  ICMP(1)[len=118]B: 10.0.10.72 -> 10.62.12.10
|
|      Dscp/Tos 24/0x60 Ttl 255
|
|      DMAC 885a.92f7.3333 SMAC 78da.6e1f.4232
|
|      Vlan 101 CoS 0 1q 0
V
...
RIT  Cos 0 Acos 0 Dscp 0
...
```

**C6500 is remarking the traffic!**

# It's a Jungle Out There!



## 3. Analyse

# Dinosaurs!



### Network Team

The default behaviour for a DFC4 module is to trust DSCP (unlike DFC3 and earlier where *"mls qos trust dscp"* had to be explicitly configured).

# It's a Jungle Out There!



## 3. Analyse



C6500



### Fancy Consultant

*I liked the old way  
better ... I'm sure they'll  
just throw bandwidth at  
the problem later.*

### But never make assumptions...

```
C6500# show run interface gigabitEthernet 3/5  
Building configuration...
```

```
Current configuration : 108 bytes
```

```
!
```

```
interface GigabitEthernet3/5
```

```
switchport
```

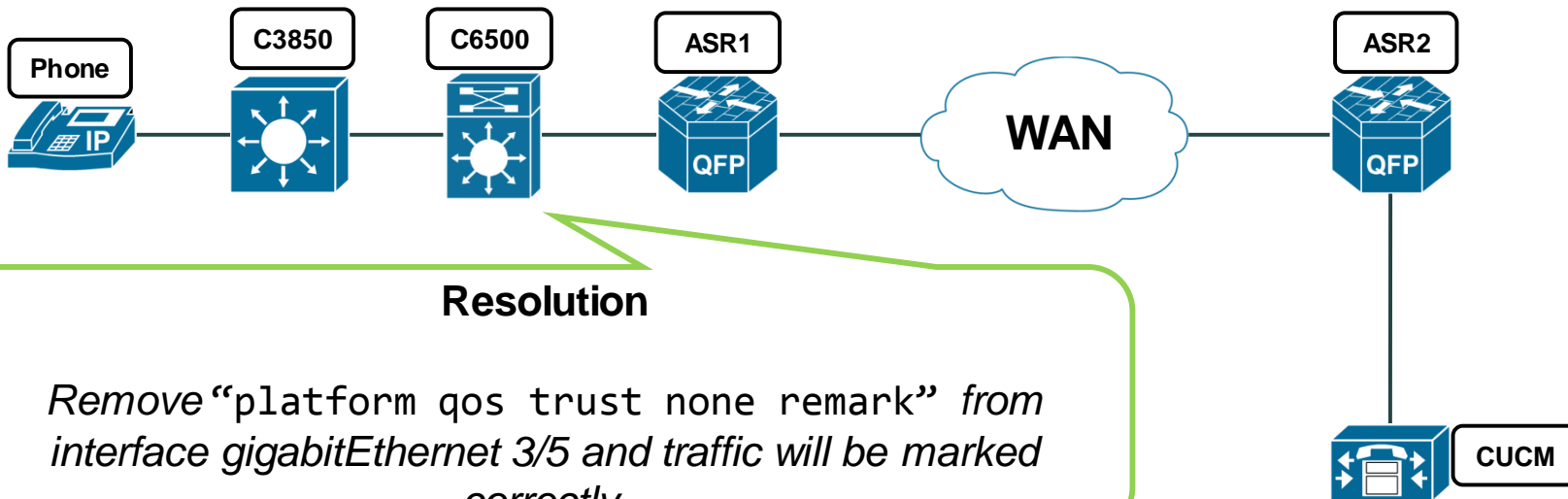
```
switchport access vlan 101
```

```
platform qos trust none remark
```

```
end
```

# It's a Jungle Out There!

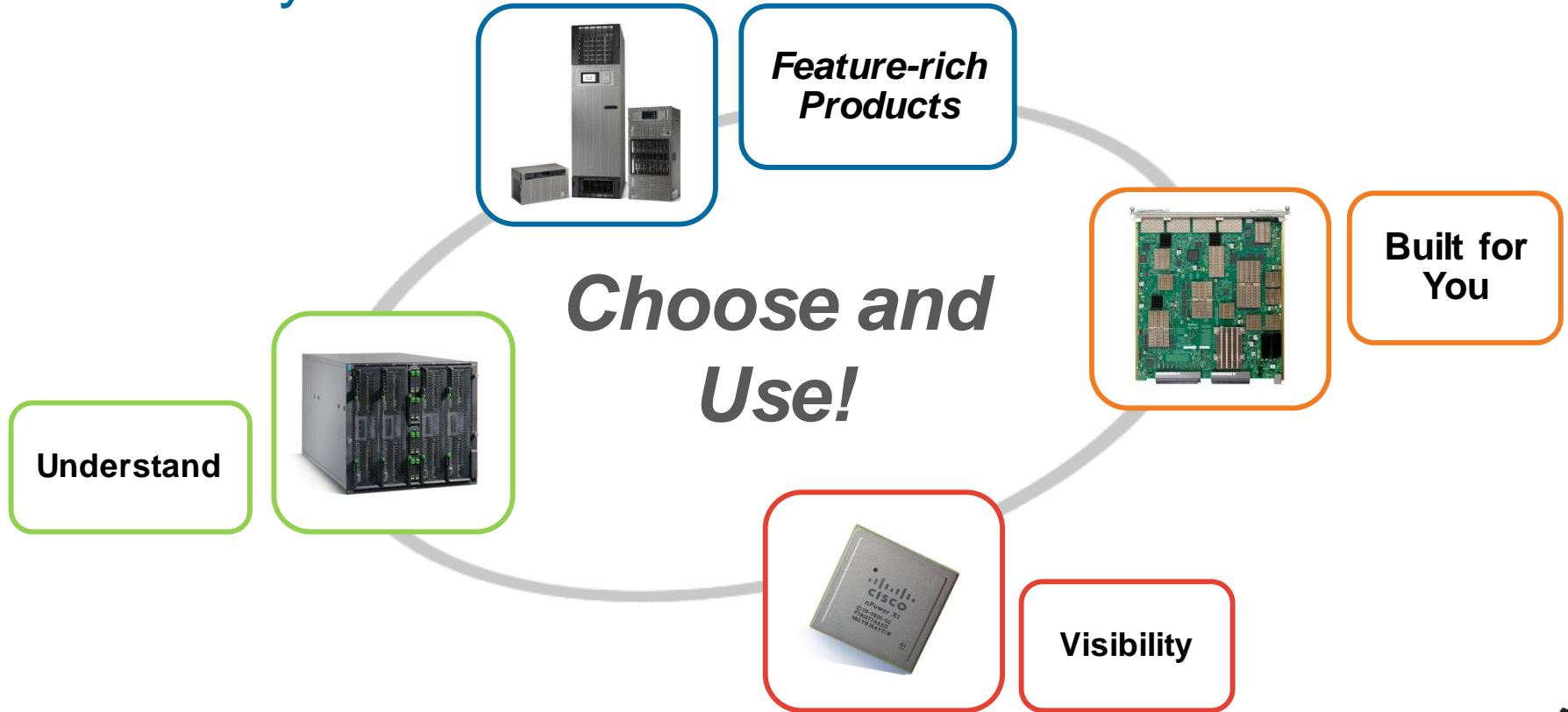
## 4. Act





## 7. Summary

# Summary



# Summary



## Key Points

- Cisco Routers and Switches are advanced and feature-rich, built with keeping end-users in mind and also network engineers.
- Cisco provides a rich set of troubleshooting and capturing tools embedded and supported across the spectrum of its products. These tools give visibility into the products helping to validate the path-of-the-packet and isolate problems.
- Knowing the tools and capabilities available on each platform will reduce the time to resolution of network issues.
- Knowing when to use a tool is just as important as having it. Remember to use the AAAA process to sharpen your understanding of the issue.

# Summary



## Comparison of Troubleshooting and Packet Capture Tools

Tool	Platform Impact	Initial Configuration? Command Context?	Exportable Data?	Key Use Case
Flexible Netflow	Platform Dependent– Very low on modern Cisco Devices	Yes  Exec Context	Netflow v9 / IPFIX	Examination of flows and repeated behaviour
Embedded Packet Capture	Minor CPU consumption, avoid in very high (~90%) CPU scenarios	No (an ACL does)  Exec Context	.PCAP File	Analysis of raw packets on the wire and application-level troubleshooting
ERSPAN	Platform Dependent– Low on modern Cisco Devices	Yes  Exec Context	N/A – Export Client creates .PCAP	Complex application-level troubleshooting
ELAM Enhanced	None	No  Exec Context	TTY output	Examining hardware forwarding behaviour
Ethalyzer	None	No  Exec Context	.PCAP File	Identifying cause of high CPU or software switching
NetDR	12.2(33)SXH+, None Pre-12.2(33)SXH, Minor	No  Debug	Decoded Headers (TTY/TXT) Use NetDR Parser Tool to generate PCAP offline	

A long-exposure photograph of a city street at night. The foreground is filled with vibrant, curved light trails from car headlights and taillights in shades of yellow, orange, and red. In the background, a pedestrian bridge spans the street, and tall buildings with lit windows and colorful neon signs (blue, purple, red) line the street. Traffic lights are visible in the distance.

Q & A

Cisco *live!*

# Complete Your Online Session Evaluation

**Give us your feedback and receive a Cisco Live 2015 T-Shirt!**

Complete your Overall Event Survey and 5 Session Evaluations.

- Directly from your mobile device on the Cisco Live Mobile App
- By visiting the Cisco Live Mobile Site  
<http://showcase.genie-connect.com/clmelbourne2015>
- Visit any Cisco Live Internet Station located throughout the venue

T-Shirts can be collected in the World of Solutions on Friday 20 March 12:00pm - 2:00pm



**Learn online with Cisco Live!**

Visit us online after the conference for full access to session videos and presentations. [www.CiscoLiveAPAC.com](http://www.CiscoLiveAPAC.com)

**Cisco** *live!*



Thank you.

Cisco *live!*



**CISCO**