TOPICS COVERED:

NETWORK ASURANCE

- SPAN / RSPAN
- ERSPAN
- PING
- TRACEROUTE
- CONDITIONAL DEBUG
- IP SLA
- CISCO NETFLOW

SPAN / RSPAN / ERSPAN

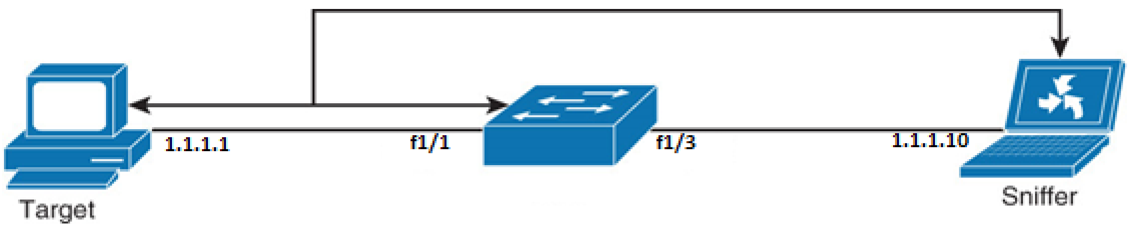
SPAN

Cisco Catalyst Switches have a feature called SPAN (Switch Port Analyzer) that lets you **copy all traffic from a source port or source VLAN** to a destination interface. This is very useful for a number of reasons:

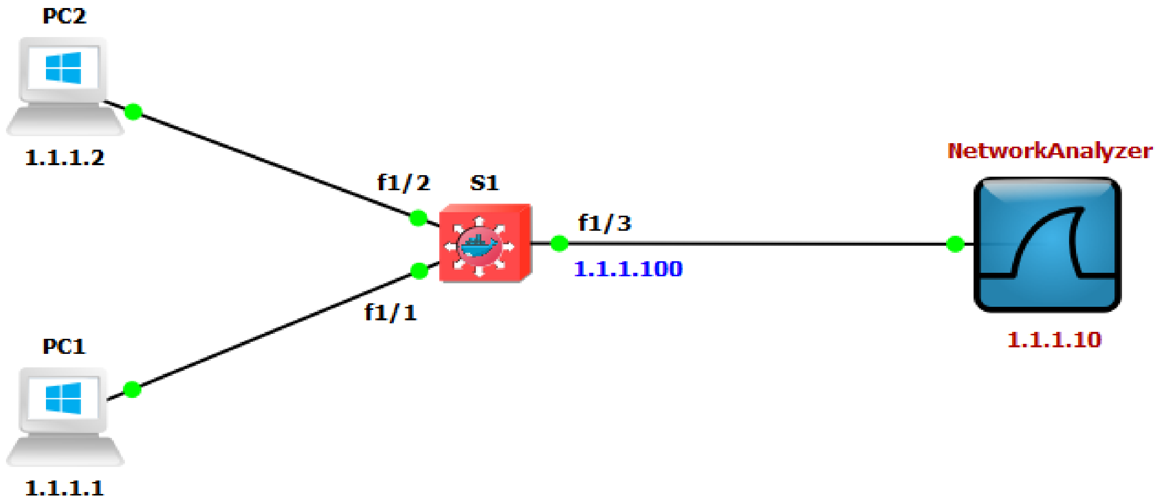
- If you want to use wireshark to capture traffic from an interface that is connected to a workstation, server, phone or anything else you want to sniff.
- Redirect all traffic from a VLAN to an IDS / IPS.
- Redirect all VoIP calls from a VLAN so you can record the calls.

The source can be an interface or a VLAN, the destination is an interface. You can choose if you want to forward **transmitted, received or both directions** to the destination interface.

EXAMPLE 1: SPAN



EXAMPLE 2: SPAN

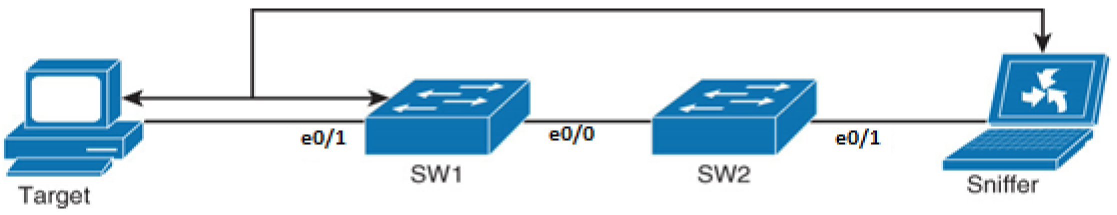


When you use a destination interface on the same switch as your switch, we call it SPAN, when the destination is a remote interface on another switch we call it **RSPAN** (Remote SPAN). When using RSPAN you need to use a VLAN for your RSPAN traffic so that traffic can travel from the source switch to the destination switch.

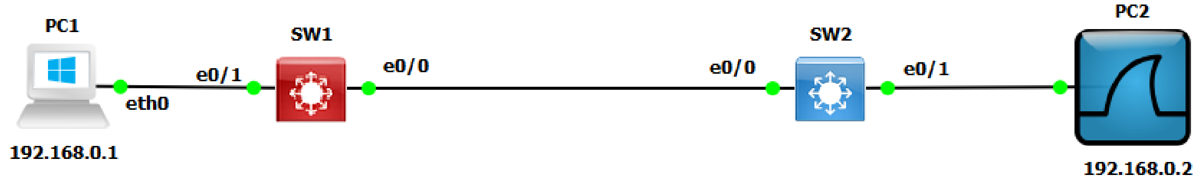
RSPAN

When you use RSPAN you need to use a VLAN that carries the traffic that you are copying. In the picture above you see SW1 which will copy the traffic from the computer onto a "RSPAN VLAN". SW2 receives the traffic and forwards it to a computer that has wireshark running. Make sure the trunks between the switches **allow the RSPAN VLAN**.

EXAMPLE 1: RSPAN



EXAMPLE 2: RSPAN



SPAN and RSPAN are great but there are a couple of things you need to keep in mind…

RESTRICTIONS:

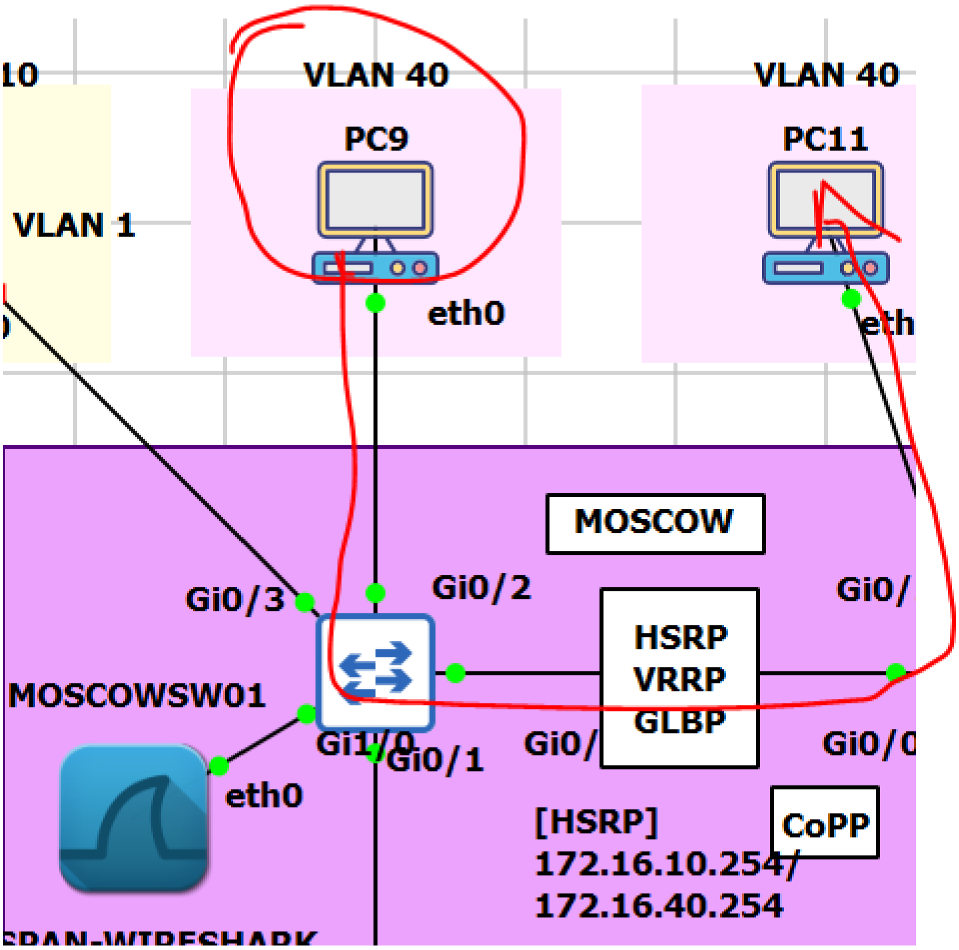
Both SPAN and RSPAN have some restrictions, I'll give you an overview of the most important ones:

- The source interface can be anything…switchport, routed port, access port, trunk port, etherchannel, etc.
- When you configure a trunk as the source interface it will copy traffic from all VLANs, however there is an option to filter this.
- You can use multiple source interfaces or multiple VLANs, but you can't mix interfaces and VLANs.
- It's very simple to overload an interface. When you select an entire VLAN as the source and use a 100Mbit destination interface…it might be too much.
- When you configure a destination port you will "lose" its configuration. By default, the destination interface will only be used to forward SPAN traffic to. However, it can be configured to permit incoming traffic from a device that is connected to the destination interface.
- Layer 2 frames like CDP, VTP, DTP and spanning-tree BPDUs are not copied by default but you can tell SPAN/RSPAN to copy them anyway.

This should give you an idea of what SPAN / RSPAN are capable of. The configuration is pretty straight-forward so let me give you some examples…

GNS3

LAB#1: SPAN CONFIGURATIONS:



MOSCOWSW01 (config)#
monitor session 1 source interface gi0/2
monitor session 1 destination interface gi1/0

```
MOSCOWSW01#sh monitor session 1
Session 1
---------
Type                  : Local Session
Source Ports          :
    Both              : Gi0/2
Destination Ports     : Gi1/0
    Encapsulation     : Native

sh monitor session 1 detail
```

As you can see, by default it will copy traffic that is transmitted and received (both) to the destination port. If you only want the capture the traffic going in one direction you have to specify it like this:

```
MOSCOWSW01 (config)#monitor session 1 source interface gi0/2 ?
 ,    Specify another range of interfaces
 -    Specify a range of interfaces
 both  Monitor received and transmitted traffic
 rx    Monitor received traffic only
 tx    Monitor transmitted traffic only
```

Just add rx or tx and you are ready to go. If interface FastEthernet 0/1 were a trunk you could add a filter to select the VLANs you want to forward:

```
monitor session 1 source interface Gi0/2 rx

**rx and tx from switch point of view
**rx will be src: .10 dst:.20 bcaz traffic is received on to switch
```

```
MOSCOWSW01 (config)#monitor session 1 filter vlan 1 - 100
```

This filter above will only forward VLAN 1 – 100 to the destination. If you don't want to use an interface as the source but a VLAN, you can do it like this:

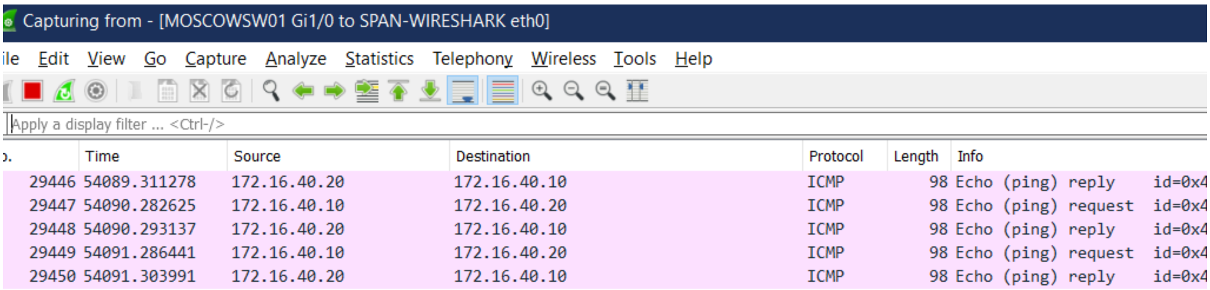
```
MOSCOWSW01 (config)#monitor session 2 source vlan 1
Switch(config)#monitor session 2 destination interface Gi0/3
```

I am unable to use session 1 for this because I am already using source interfaces for that session. It's also impossible to use the same destination interface for another session. This is why I created another session number and picked Gi 0/3 as a destination.

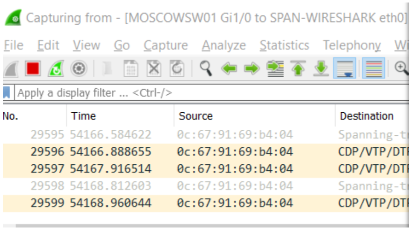
VALIDATION:

```
MOSCOWSW01(config)#do sh int gi1/0
GigabitEthernet1/0 is up, line protocol is down (monitoring)
  Hardware is iGbE, address is 0c67.9181.a704 (bia 0c67.9181.a704)
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
```

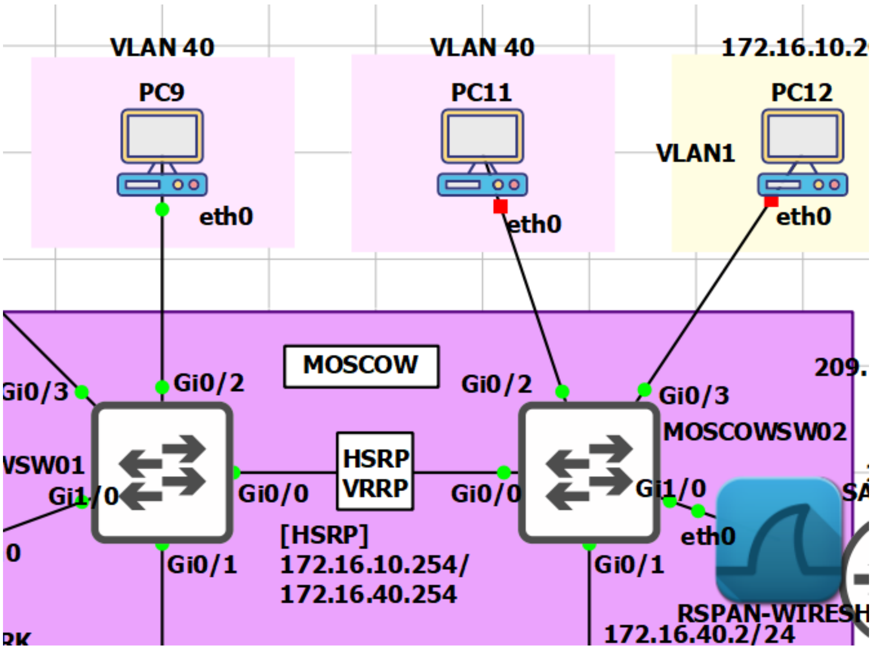
SPAN capture working in right way.
I can see packet capture on my Wireshark



When I stop monitor session 1 or 2, span traffic stops coming to wireshark.



<u>LAB#2: RSPAN</u> CONFIGURATIONS:



The idea is to forward traffic from Gi0/2 on MOSCOWSW01 to Gi 1/0 on MOSCOWSW02. There are a couple of things we have to configure here:

MOSCOWSW01
vlan 100
remote-span

MOSCOWSW02 (config)#
vlan 100
remote-span

First, we need to create the VLAN and tell the switches that it's a RSPAN vlan. This is something that is easily forgotten. Secondly, we will configure the link between the two switches as a trunk:

MOSCOWSW01(config)#
interface Gi 0/0
switchport trunk encapsulation dot1q
switchport mode trunk

MOSCOWSW02(config)#
interface Gi 0/0
switchport trunk encapsulation dot1q
switchport mode trunk

Now we can configure RSPAN:
MOSCOWSW01(config)#
monitor session 1 source interface Gi 0/2
monitor session 1 destination remote vlan 100
This selects Gi 0/1 as the source and VLAN 100 as the destination…

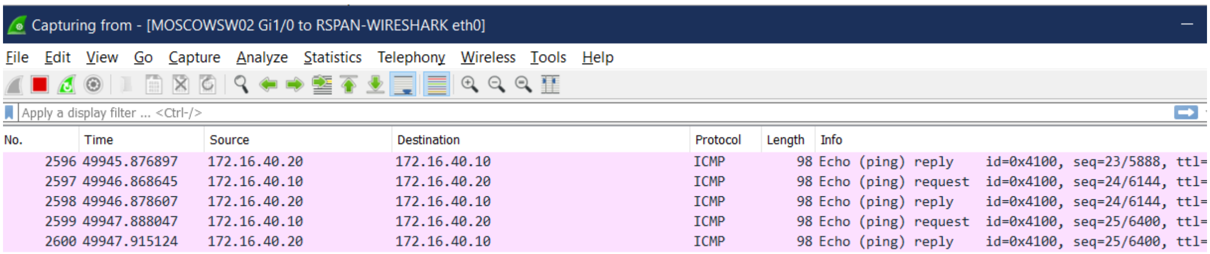
MOSCOWSW02(config)#
monitor session 1 source remote vlan 100
monitor session 1 destination interface Gi 1/0

And on MOSCOWSW02, we select VLAN 100 as the source and Gi 0/1 as its destination. Here's the output of the show monitor session command:

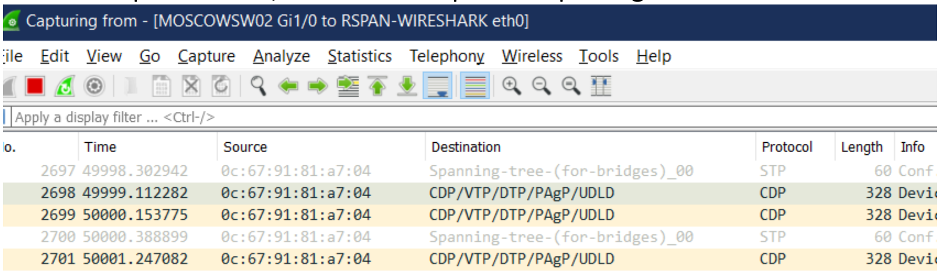
```
MOSCOWSW01# show monitor session 1
Session 1
---------
Type              : Remote Source Session
Source Ports      :
   Both           : Gi 0/2
Dest RSPAN VLAN   : 40

MOSCOWSW02#show monitor session 1
Session 1
---------
Type              : Remote Destination Session
Source RSPAN VLAN    : 100
Destination Ports    : Gi 1/0
   Encapsulation     : Native
        Ingress      : Disabled
```

```
MOSCOWSW02(config)#do sh int gi1/0
GigabitEthernet1/0 is up, line protocol is down (monitoring)
  Hardware is iGbE, address is 0c67.9181.a704 (bia 0c67.9181.a704)
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
```

Wireshark Capture Applied between MOSCOWSW02_Gi1/0 to Wireshark host can see RSPAN captured traffic:

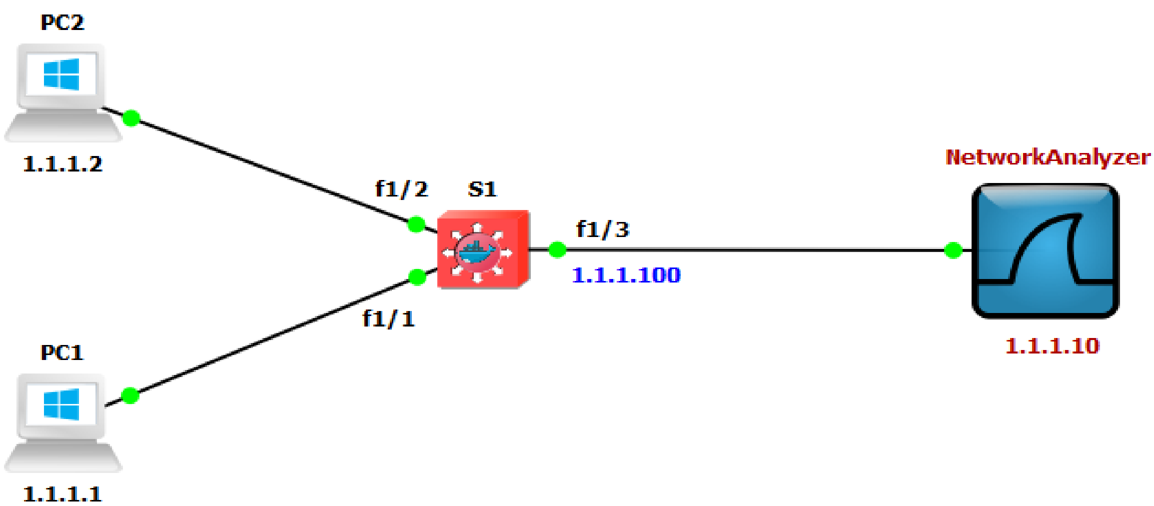


When I stop the traffic, Wireshark stops the capturing too:



ASSIGNMENTS - (FOR STUDENTS)

LAB#3 SPAN



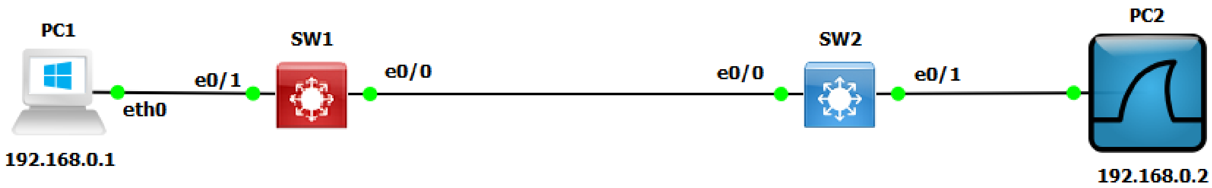
S1(config)#monitor session 1 source interface FastEthernet 1/1
S1(config)#monitor session 1 destination interface FastEthernet 1/3
S1#show monitor session 1
S1#show interfaces FastEthernet 1/3

```
S1#show interfaces fastEthernet 1/3
FastEthernet1/3 is up, line protocol is down (monitoring)
  Hardware is Fast Ethernet, address is c201.3ec8.f103 (bia c201.3ec8.f103)
  MTU 1500 bytes, BW 100000 Kbit, DLY 100 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
```

	Time	Source	Destination	Protocol	Length	Info
32	27.236167	1.1.1.2	1.1.1.1	ICMP	98	Echo (ping) reply
34	28.251150	1.1.1.1	1.1.1.2	ICMP	98	Echo (ping) request
35	28.251515	1.1.1.2	1.1.1.1	ICMP	98	Echo (ping) reply
37	29.267079	1.1.1.1	1.1.1.2	ICMP	98	Echo (ping) request
38	29.267404	1.1.1.2	1.1.1.1	ICMP	98	Echo (ping) reply
39	30.281034	1.1.1.1	1.1.1.2	ICMP	98	Echo (ping) request
40	30.281375	1.1.1.2	1.1.1.1	ICMP	98	Echo (ping) reply
46	31.297370	1.1.1.1	1.1.1.2	ICMP	98	Echo (ping) request

ASSIGNMENTS - (FOR STUDENTS)

LAB#4 RSPAN



SW1 Configuration
SW1(config)#vlan 100 SW1(config-vlan)#remote-span
SW1(config)#interface Ethernet 0/0 SW1(config-if)#switchport trunk encapsulation dot1q SW1(config-if)#switchport mode trunk
SW1(config)#monitor session 1 source interface Ethernet 0/1 SW1(config)#monitor session 1 destination remote vlan 100
SW1#show monitor session all
SW2 Configuration
SW2(config)#vlan 100 SW2(config-vlan)#remote-span
SW2(config)#interface Ethernet 0/0 SW2(config-if)#switchport trunk encapsulation dot1q SW2(config-if)#switchport mode trunk
SW2(config)#monitor session 1 source remote vlan 100 SW2(config)#monitor session 1 destination interface Ethernet 0/1
SW2#show monitor session all

VALIDATIONS:

```
MOSCOWSW01# show monitor session 1
Session 1
---------
Type              : Remote Source Session
Source Ports          :
    Both            : Gi0/2
Dest RSPAN VLAN       : 100
```

```
MOSCOWSW02#show monitor session 1
Session 1
---------
Type                : Remote Destination Session
Source RSPAN VLAN      : 100
Destination Ports      : Gi1/0
    Encapsulation      : Native
```

ERSPAN Configuration on Cisco IOS XE

SPAN and RSPAN allow us to copy traffic from one interface to another. This is great if you want to send traffic to a sensor or if you want to take a closer look at it with a packet analyzer like Wireshark. SPAN is however limited to one switch, RSPAN is able to send traffic between switches but this traffic can't be routed.

ERSPAN (Encapsulated Remote Switched Port Analyzer) solves this issue! It uses GRE encapsulation, this allows us to route SPAN traffic from a source to a destination. You can use ERSPAN on IOS XE, NX-OS and the Catalyst 6500/7600 switches. Unfortunately, It's not supported on the "smaller" IOS switches and routers.

When you want to configure ERSPAN, there's a couple of things you have to keep in mind. For the source session, we have to configure:

- Unique session ID.
- List of source interfaces or source VLANs that you want to monitor. Not all platforms support every possible source.
- What traffic we want to capture: tx, rx or both.
- Destination IP address for the GRE tunnel.
- Origin IP address which is used as the source for the GRE tunnel.
- Unique ERSPAN flow ID.
- Optional: you can specify attributes like the ToS (Type of Service), TTL, etc.

For the destination we have to specify:

- Unique session ID, doesn't have to match with the source session.
- Destination interface(s) where you want to forward the traffic to.
- Source IP address, which is the same as the destination IP address of the corresponding source session
- Unique ERSPAN flow ID, has to match with the source session.

- KEY POINTS:
  ERSPAN is term stand for Encapsulated Remote Switched Port Analyzer.
- Feature present on new IOS-XE on ASR1000 also available on Catalyst 6500.
- ERSPAN brings generic routing encapsulation (GRE) for all captured traffic.
- ERSPAN is used to send traffic for sniffing over L3 networks using GRE tunnel.
- ERSPAN on Cisco ASR 1000 Series Routers supports only The Layer 3 interfaces.

- Ethernet interfaces are not supported on ERSPAN configured as Layer 2 interfaces.

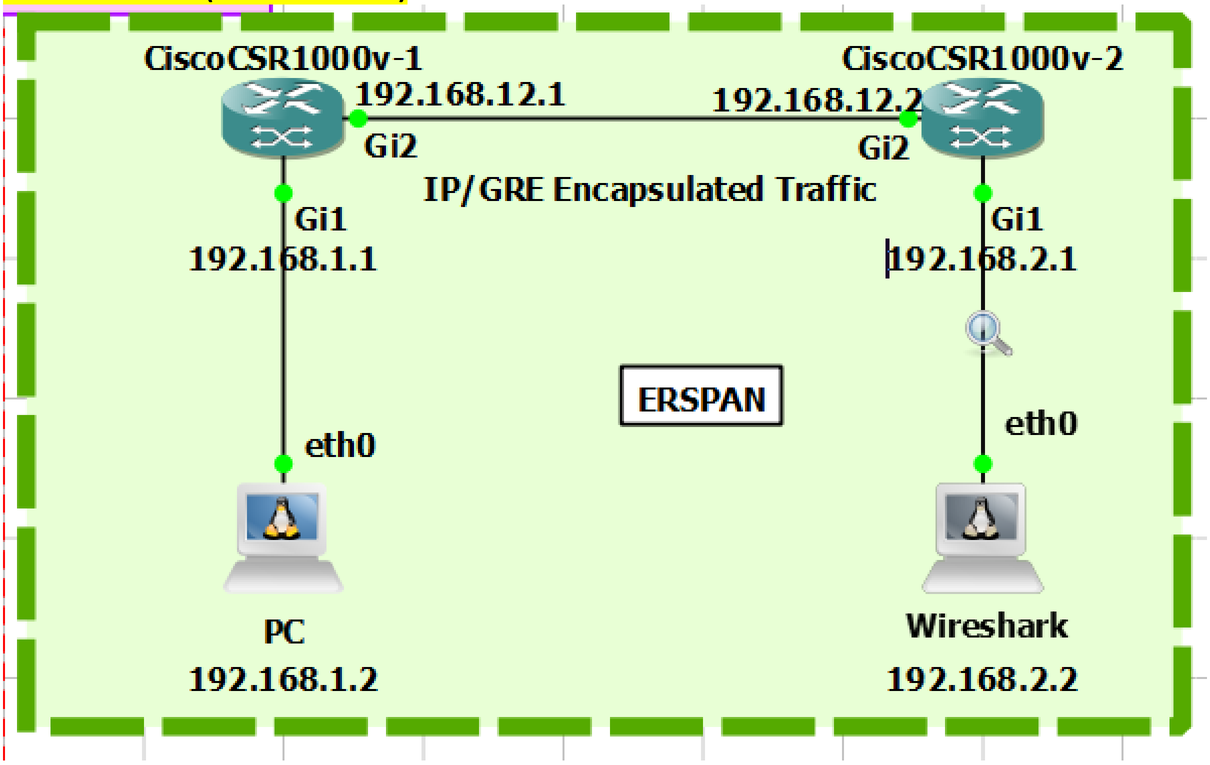
For the Source session, need to Configure:

- To configure ERSPAN it require Unique session ID, List of source interfaces or VLANs.
- What is the traffic we want to capture tx (Transmit Only), rx (Receive Only) or both.
- ERSPAN configuration require Destination IP address for the GRE tunnel to connect.
- Origin IP address which is used as source for generic routing encapsulation tunnel.
- Unique Encapsulated Remote Switched Port Analyzer (ERSPAN) flow ID (Identity).

For the Destination need to Specify:

- For the Destination Unique session ID doesn't have to match with source session.
- ERSPAN require Destination interface(s) where you want to forward the traffic to.
- Source IP address has to match with the origin IP address of the source session.
- ERSPAN require Unique ERSPAN flow ID, has to match with the source session.

GNS3
ASSIGNMENTS - (FOR STUDENTS)



PC1 IP Address Configuration:

CSR1 Basic IP Configuration
CiscoCSR1000v-1(config)# interface gigabitEthernet 2 ip address 192.168.12.1 255.255.255.0 no shutdown exit interface gigabitEthernet 1 ip add 192.168.1.1 255.255.255.0 no shutdown exit ip route 0.0.0.0 0.0.0.0 192.168.12.2

CSR2 Basic IP Configuration
CiscoCSR1000v-2(config)# interface gigabitEthernet 2 ip address 192.168.12.2 255.255.255.0 no shutdown exit interface gigabitEthernet 1 ip add 192.168.2.1 255.255.255.0 no shutdown exit ip route 0.0.0.0 0.0.0.0 192.168.12.1

CSR1 ERSPAN Configuration:
CSR1(config)# monitor session 1 type erspan-source source interface GigabitEthernet 1 rx no shutdown destination erspan-id 100 ip address 192.168.2.2 origin ip address 192.168.12.1
CSR1#show monitor session 1

CSR2 ERSPAN Configuration:

CSR2(config)# monitor session 1 type erspan-destination no shutdown destination interface GigabitEthernet 2 source erspan-id 100 ip address 192.168.2.2
CSR2#show monitor session 1

Verification:

Let's ping from PC to CSR1 local interface IP which is 192.168.1.1

```
root@PC:~# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.2  netmask 255.255.255.0  broadcast 0.0.0.0
        inet6 fe80::a029:88ff:fe20:d327  prefixlen 64  scopeid 0x20<link>
        ether a2:29:88:20:d3:27  txqueuelen 1000  (Ethernet)
        RX packets 70  bytes 16072 (16.0 KB)
        RX errors 0  dropped 4  overruns 0  frame 0
        TX packets 42  bytes 3624 (3.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@PC:~# ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=255 time=1.97 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=255 time=2.01 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=255 time=1.43 ms
```

```
CSR2#show monitor session 1
Session 1
---------
Type                      : ERSPAN Destination Session
Status                    : Admin Enabled
Destination Ports         : Gi2
Source IP Address         : 192.168.2.2
Source ERSPAN ID          : 100
```

```
CSR1#show monitor session 1
Session 1
---------
Type                      : ERSPAN Source Session
Status                    : Admin Enabled
Source Ports              :
    RX Only               : Gi2
Destination IP Address    : 192.168.2.2
MTU                       : 1464
Destination ERSPAN ID     : 100
Origin IP Address         : 192.168.12.1
```

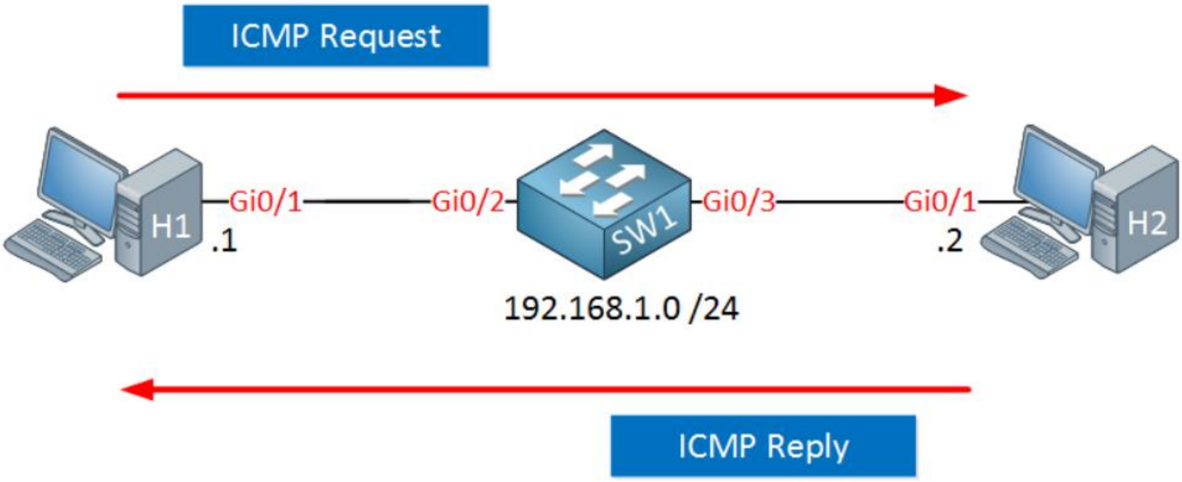
After ping from PC to CSR1 local interface CSR1, encapsulate the traffic and send to Sniffer.

No.	Time	Source	Destination	Protocol	Length	Info
22	173.528901	192.168.1.2	192.168.1.1	ICMP	148	Echo (ping) request id=0x0268, seq=1,
23	174.513989	192.168.1.2	192.168.1.1	ICMP	148	Echo (ping) request id=0x0268, seq=2,
24	175.515633	192.168.1.2	192.168.1.1	ICMP	148	Echo (ping) request id=0x0268, seq=3,
25	176.517243	192.168.1.2	192.168.1.1	ICMP	148	Echo (ping) request id=0x0268, seq=4,
26	177.517885	192.168.1.2	192.168.1.1	ICMP	148	Echo (ping) request id=0x0268, seq=5,
27	178.517311	192.168.1.2	192.168.1.1	ICMP	148	Echo (ping) request id=0x0268, seq=6,
28	178.629621	a2:29:88:20:d3:27	0c:06:07:92:2e:01	ARP	110	Who has 192.168.1.1? Tell 192.168.1.2
29	179.517905	192.168.1.2	192.168.1.1	ICMP	148	Echo (ping) request id=0x0268, seq=7,

```
> Frame 22: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits) on interface -, id 0
> Ethernet II, Src: 0c:06:07:be:03:01 (0c:06:07:be:03:01), Dst: c2:01:24:ec:00:00 (c2:01:24:ec:00:00)
> Internet Protocol Version 4, Src: 192.168.12.1, Dst: 192.168.2.2
v Generic Routing Encapsulation (ERSPAN)
  > Flags and Version: 0x1000
    Protocol Type: ERSPAN (0x88be)
    Sequence Number: 0
> Encapsulated Remote Switch Packet ANalysis Type II
> Ethernet II, Src: a2:29:88:20:d3:27 (a2:29:88:20:d3:27), Dst: 0c:06:07:92:2e:01 (0c:06:07:92:2e:01)
> Internet Protocol Version 4, Src: 192.168.1.2, Dst: 192.168.1.1
> Internet Control Message Protocol
```

PING

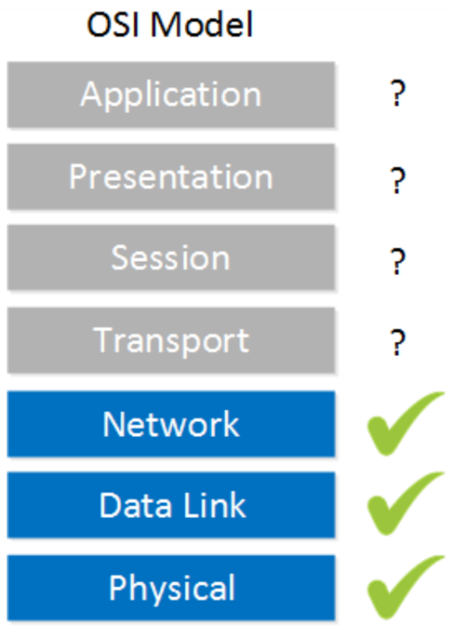
The ping command can be used to quickly check if a remote device is reachable or not. In this lesson, I'll show you how you can use it to troubleshoot issues in your network.



When this ping from H1 to H2 is successful, what does it tell us?

```
C:\Users\H1>ping 192.168.1.2



Pinging 192.168.1.2 with 32 bytes of data:

Reply from 192.168.1.2: bytes=32 time<1ms TTL=255

Reply from 192.168.1.2: bytes=32 time=1ms TTL=255

Reply from 192.168.1.2: bytes=32 time<1ms TTL=255

Reply from 192.168.1.2: bytes=32 time<1ms TTL=255



Ping statistics for 192.168.1.2:

    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),

Approximate round trip times in milli-seconds:

    Minimum = 0ms, Maximum = 1ms, Average = 0ms
```

It does tell us quite some things. Think of the OSI model:



We don't have to worry about things like cabling, connectors, interfaces, VLANs, MAC addresses, ARP, IP addresses and subnet masks. If any of these were configured incorrectly, the upper layers wouldn't work.

It doesn't tell me much about the transport, session, presentation and application layers though.

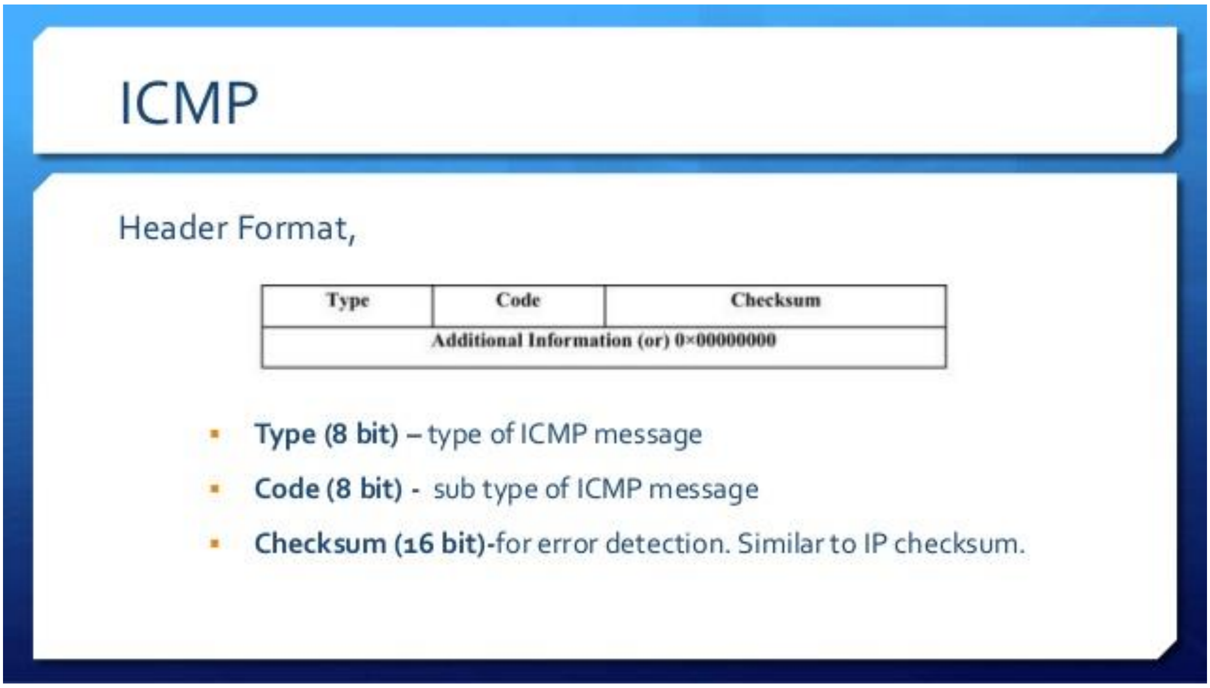
If you think an access-list is blocking ICMP packets, you can always try the telnet command to different port numbers. This is a quick way to test if the remote device is blocking traffic to certain TCP port numbers. For example, on Cisco IOS, you can try to telnet to 192.168.1.1 80 to see if you can reach TCP port 80 on 192.168.1.1. On your desktop, nmap is a great tool to try.

The ping command can also be used to isolate MTU problems

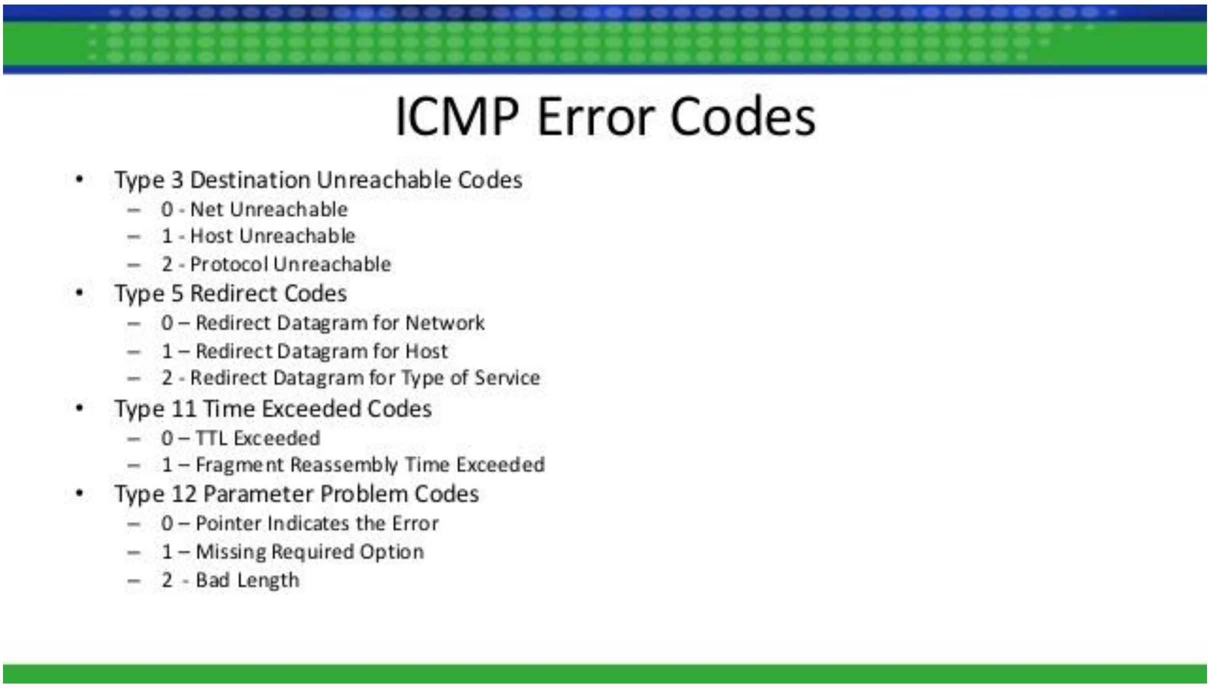
ICMP (INTERNET CONTROL MESSAGE PROTOCOL)

ICMP (Internet Control Message Protocol) is a network protocol used for diagnostics and network management. A good example is the “ping” utility which uses an ICMP request and ICMP reply message. When a certain host of port is unreachable, ICMP might send an error message to the source. Another example of an application that uses ICMP is traceroute.

ICMP messages are encapsulated in IP packets so most people would say that it’s a layer 4 protocol like UDP or TCP. However, since ICMP is a vital part of the IP protocol it is typically considered a layer 3 protocol.



ICMP ERROR CODES:



[refer next screenshot for all error codes]

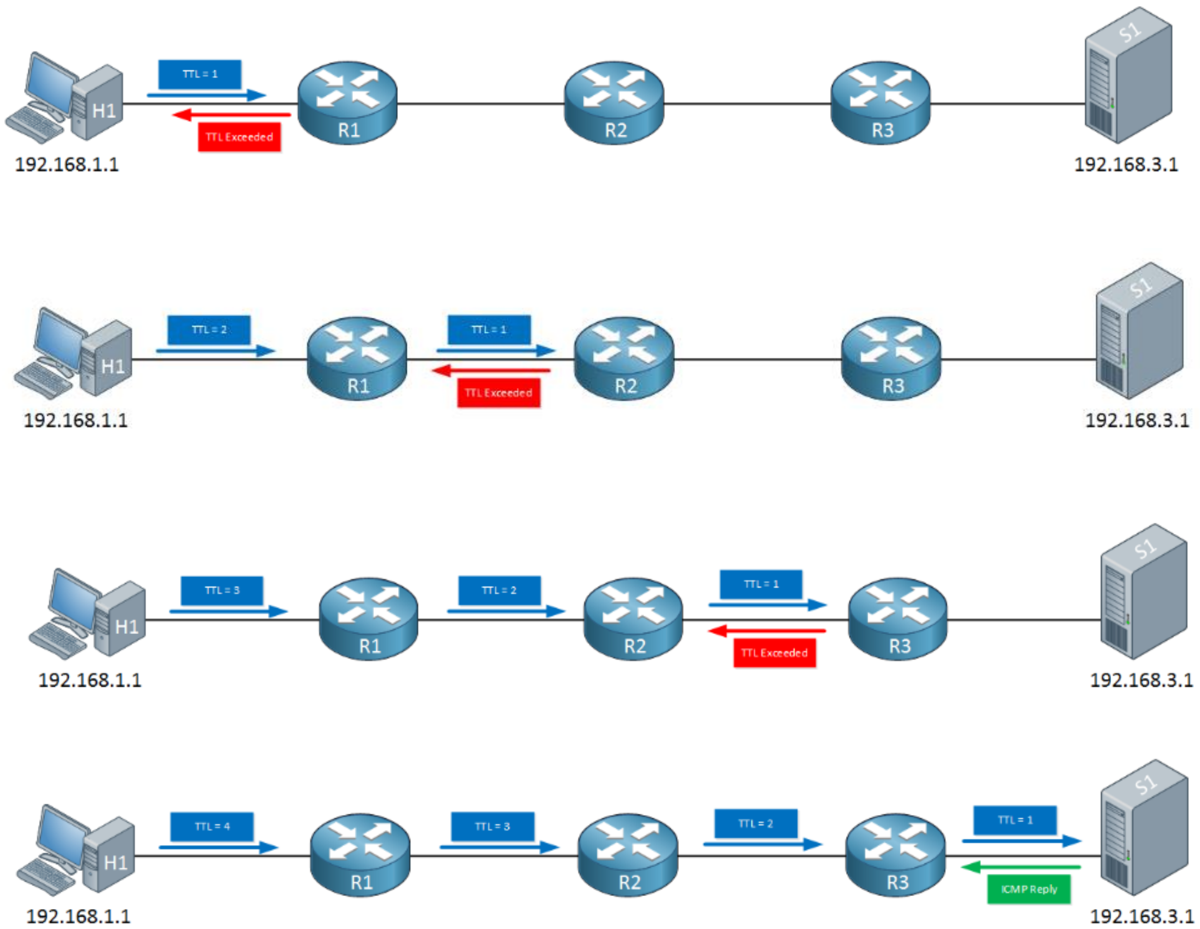
ICMP DETAILED CODES:

Type	Code	Description
0 – Echo Reply	0	Echo reply
3 – Destination Unreachable	0	Destination network unreachable
	1	Destination host unreachable
	2	Destination protocol unreachable
	3	Destination port unreachable
	4	Fragmentation needed and DF flag set
	5	Source route failed
5 – Redirect Message	0	Redirect datagram for the Network
	1	Redirect datagram for the host
	2	Redirect datagram for the Type of Service and Network
	3	Redirect datagram for the Service and Host
8 – Echo Request	0	Echo request
9 – Router Advertisement	0	Use to discover the addresses of operational routers
10 – Router Solicitation	0	
11 – Time Exceeded	0	Time to live exceeded in transit
	1	Fragment reassembly time exceeded
12 – Parameter Problem	0	Pointer indicates error
	1	Missing required option
	2	Bad length
13 – Timestamp	0	Used for time synchronization
14 – Timestamp Reply	0	Reply to Timestamp message

TRACEROUTE

So, how does traceroute work?

Traceroute uses the TTL (Time to Live) field in the IP packet header. Normally, TTL is used to prevent packets from being forwarded forever when there is a routing loop. Whenever an IP packet is forwarded by a router, the **TTL is decreased by one**. When the **TTL is zero, the IP packet will be discarded**.



Each IP packet that we send is called a **probe**.
Traceroute can be used with ICMP, UDP and TCP, depending on your operating system.

TRACEROUTE COMMAND:

1. Windows:
C:\Users\vmware>tracert 192.168.3.1

2. Linux:
traceroute -N 1 -q 1 192.168.3.1

3. Cisco Routers:
traceroute 192.11.1.11

Traceroute proceeds unless all (usually three) sent packets are lost more than twice; then the connection is lost and the route cannot be evaluated. Ping, on the other hand, only computes the final round-trip times from the destination point.

LIMITATION OF TRACEROUTE:
1. For example, traceroute does not discover paths at the router level, but at the interface level.
2. Another limitation appears when routers do not respond to probes or when routers have a limit for ICMP responses, example: firewalls
3. In the presence of traffic load balancing, traceroute may indicate a path that does not actually exist; to minimize this problem there is a traceroute modification called Paris-traceroute, which maintains the flow identifier of the probes to avoid load balancing.

Traceroute on Cisco IOS might be very slow. This is because it will attempt a DNS lookup for each IP address. To make it faster, make sure these lookups can be resolved or disable DNS lookups with the **no ip** domain-lookup command.

If you want to cancel traceroute, hit SHIFT+CTRL+6, let go then press X.

If you see some asterisks (timeouts) in your trace for some routers, then this router (or firewall) is probably configured with an access-list and configure not to respond with any TTL expired messages.

CONDITIONAL DEBUG

Conditional debug is very useful to filter out some of the debug information that you see on a (busy) router. It allows us to only show debug information that matches a certain interface, MAC address, username and some other items.

GNS3:

MUMBAIR3#debu condition interface e0/1
Condition 1 set

Be careful…using **no debug all** or **undebug all** doesn't remove the condition. You need to remove it using the command that I just showed you!

MUMBAIR3#**undebug condition interface e0/1**
This condition is the last interface condition set.
Removing all conditions may cause a flood of debugging
messages to result, unless specific debugging flags
are first removed.

Proceed with removal? [yes/no]: yes
Condition 1 has been removed

IP SLA

IP SLA (Service-Level Agreement) is a great feature on Cisco IOS devices that can be used to "measure" network performance.

This can be something simple like a ping where we check the round-trip time or something more advanced like a VoIP RTP packet where we check the delay, jitter and calculate a Mean opinion score (MOS) score that gives you an indication what the voice quality will be like.

Measuring network performance is pretty cool but what makes IP SLA even more powerful is that you can combine it with static routes, policy-based routing and routing protocols like OSPF or EIGRP.

Let us assume that we have two ISPs that we can use to reach a remote branch router. Instead of a simple ping, we can send RTP packets and check these for a certain delay, jitter and calculate a MOS score. When we get below a certain threshold, we will switch from ISP1 to ISP2.

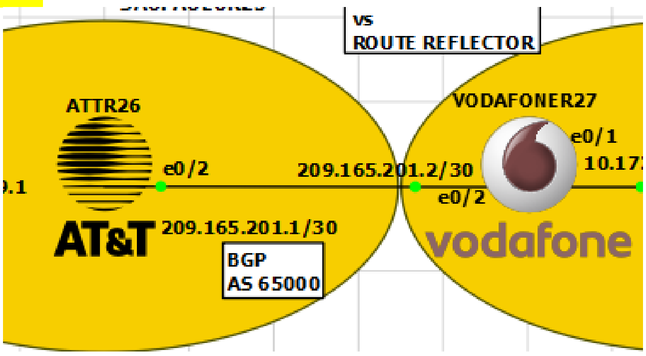
Each measurement that we do with IP SLA is called an **operation**. For each operation we have to configure the type of traffic, source IP, destination IP, port numbers, etc. We can then configure when to run the operation…24/7, 9-to-5, etc.

When you use IP SLA for a simple ping then you only have to configure your local router. However when you want to use it for some more advanced things like sending RTP packets then you have to configure the remote router to **respond** to your IP SLA traffic.

Besides pings and RTP, there are a lot of different operations we can use:

- TCP Connections
- UDP
- DNS
- DHCP
- HTTP
- FTP

GNS3
1. ICMP ECHO OPERATION



```
hostname ATT26
!
!
ip sla 1
 icmp-echo 209.165.201.2
 frequency 10
!
ip sla schedule 1 start-time now life forever
!
end
```

```
show ip sla configuration
show ip sla statistics
```

2. UDP JITTER OPERATION

```
hostname ATT26
ip sla 2
udp-jitter 209.165.201.2 16384 codec g711alaw
frequency 60
tos 184
```

tos = type of service

```
ip sla schedule 2 life forever start-time now
```

VODAFONER27 (config)#ip sla responder

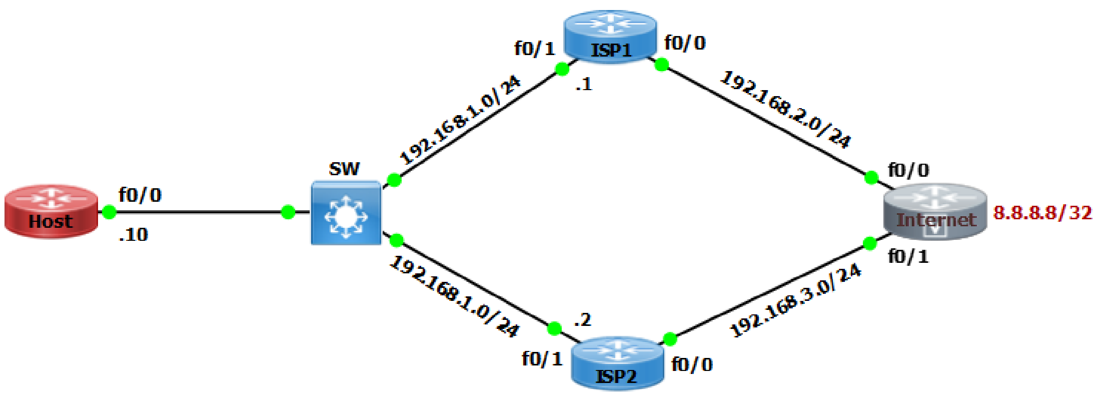
The ip sla responder command is required on VODAFONER27 otherwise it will drop our UDP packets. Let's verify our work:

```
show ip sla configuration 2
show ip sla statistics 2
```

Above you can see our results, how often these probes have been sent and at the bottom you can see the MOS score which was calculated. This is **based on a scale from 1 – 5 so 4.34 is pretty good**.

GNS3

ASSIGNMENTS - (FOR STUDENTS)



ISP1 Configuration
ISP1(config)#interface f0/1 ISP1(config-if)#ip add 192.168.1.1 255.255.255.0 ISP1(config-if)#no shutdown
ISP1(config)#interface f0/0 ISP1(config-if)#ip add 192.168.2.1 255.255.255.0 ISP1(config-if)#no shutdown
ISP1(config)#router eigrp 1 ISP1(config-router)#network 192.168.2.0 ISP1(config-router)#network 192.168.1.0 ISP1(config-router)#no auto-summary
ISP2 Configuration
ISP2(config)#interface f0/1 ISP2(config-if)#ip add 192.168.1.2 255.255.255.0 ISP2(config-if)#no shutdown
ISP2(config)#interface f0/0 ISP2(config-if)#ip add 192.168.3.2 255.255.255.0 ISP2(config-if)#no shutdown
ISP2(config)#router eigrp 1 ISP2(config-router)#network 192.168.3.0 ISP2(config-router)#network 192.168.1.0 ISP2(config-router)#no auto-summary
Internet Configuration
Internet(config)#interface f0/1 Internet(config-if)#ip add 192.168.3.3 255.255.255.0 Internet(config-if)#no shutdown
Internet(config)#interface f0/0 Internet(config-if)#ip add 192.168.2.3 255.255.255.0 Internet(config-if)#no shutdown
Internet(config)#interface loopback 8 Internet(config-if)#ip address 8.8.8.8 255.255.255.255
Internet(config)#router eigrp 1

Internet(config-router)#network 0.0.0.0 Internet(config-router)#no auto-summary
Host IP Configuration
Host(config)#interface f0/0 Host(config-if)#ip address 192.168.1.10 255.255.255.0 Host(config-if)#no shutdown

IP SLA Configuration in Host Router
Host(config)#ip sla 1 Host(config-ip-sla)#icmp-echo 192.168.1.1 Host(config-ip-sla-echo)#frequency 10 Host(config-ip-sla-echo)#timeout 5000 Host(config-ip-sla-echo)#exit
Host(config)#ip sla schedule 1 start-time now life forever
Track Configuration to bind to IP SLA
Host(config)#track 10 ip sla 1 reachability
Host(config)#ip route 0.0.0.0 0.0.0.0 192.168.1.1 track 10 Host(config)#ip route 0.0.0.0 0.0.0.0 192.168.1.2 20
Host# show track
Host# show ip sla statistics
Host# show ip route

Show ip route display best route is 192.168.1.1 for all traffic.

```
Host#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, + - replicated route

Gateway of last resort is 192.168.1.1 to network 0.0.0.0

S*    0.0.0.0/0 [1/0] via 192.168.1.1
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.0/24 is directly connected, FastEthernet0/0
L        192.168.1.10/32 is directly connected, FastEthernet0/0
```

IP SLA operation is ok and working.

```
Host#show ip sla statistics
IPSLAs Latest Operation Statistics

IPSLA operation id: 1
Type of operation: icmp-echo
        Latest RTT: 47 milliseconds
Latest operation start time: *09:45:37.811 UTC Mon Apr 8 2019
Latest operation return code: OK
Number of successes: 36
Number of failures: 0
Operation time to live: Forever
```

Show track is monitoring IP SLA reachability its working and return code is OK.

```
Host#show track
Track 10
  IP SLA 1 reachability
  Reachability is Up
    1 change, last change 00:05:34
  Latest operation return code: OK
  Latest RTT (millisecs) 61
  Tracked by:
    STATIC-IP-ROUTING 0
```

Traceroute show best route is 192.168.1.1 which is ISP1.

```
Host#traceroute 8.8.8.8 numeric

Type escape sequence to abort.
Tracing the route to 8.8.8.8

  1 192.168.1.1 56 msec 56 msec 60 msec
  2 192.168.2.3 80 msec 56 msec 60 msec
```

After shutdown ISP1 interface F0/1 track IP SLA messages is generated.

```
Host#
*Apr  8 09:50:24.063: %TRACKING-5-STATE: 10 ip sla 1 reachability Up->Down
```

Now second backup route with AD 20 has been installed automatically.

```
Host#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, + - replicated route

Gateway of last resort is 192.168.1.2 to network 0.0.0.0

S*    0.0.0.0/0 [20/0] via 192.168.1.2
      192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C        192.168.1.0/24 is directly connected, FastEthernet0/0
L        192.168.1.10/32 is directly connected, FastEthernet0/0
```

Now traceroute again second backup route 192.168.1.2 ISP2.

```
Host#traceroute 8.8.8.8 numeric

Type escape sequence to abort.
Tracing the route to 8.8.8.8

  1 192.168.1.2 56 msec 52 msec 56 msec
  2 192.168.3.3 76 msec 72 msec 84 msec
```

CISCO NETFLOW

- NetFlow is an application for collecting IP traffic information.
- NetFlow is a protocol developed by Cisco Systems to record all IP traffic flows.
- Flow is unidirectional stream of packets from a source to destination.
- Flow is a stream of packets that share the same characteristics.
- Flow characteristics are like source, destination port, address, protocol, type etc.
- NetFlow allows tracking these flows on in the network.
- NetFlow track the number of packets sent, bytes sent, packet sizes and more.
- Configure router to keep track of all flows and then export them to central server.
- NetFlow is a great protocol to get an insight in the network traffic.
- NetFlow allows seeing real time data on who/what is eating the bandwidth.
- NetFlow is used to collect data flows from routers & switches interfaces.
- NetFlow is an application that provides statistics on packets flowing through routers.
- NetFlow captures statistics on IP flows through a device.
- NetFlow allows collecting traffic and analyzing it through a program.
- NetFlow is configured in the interface configuration mode on a router.
- NetFlow monitor of ingress traffic, egress traffic, or both ingress or egress traffic.
- Specify the IP address of the NetFlow collector & UDP port of collector is listening.
- Provides statistics on packets flowing through a router or a switch.
- NetFlow collect and export the data to enable network & security monitoring.
- NetFlow collect & export data for network planning, traffic analysis & IP accounting.
- NetFlow records are exported to a NetFlow collector using UDP.
- The standard value is **UDP port 2055**, but other values can be set **9555** or **9995**.
- NetFlow can be used for network accounting and security auditing.
- NetFlow consumes additional memory of devices to process.

Network management protocols like SNMP allow us to monitor our network. We can check things like cpu load, memory usage, interface status and even the load of an interface. Other tools like NBAR allow us to see what kind of protocols are used.

One of the things we can't do with those tools is tracking all flows in our network. A flow is a stream of packets that share the same characteristics like source/destination port, source/destination address, protocol, type, service marking, etc.

NetFlow allows us to track these flows on our network. We can use this information to solve problems like bottlenecks, identify what applications are used, how much bandwidth they use etc.

For each of the flows, NetFlow will track the number of packets sent, bytes sent, packet sizes and more. You can configure your router to keep track of all flows and then export them to a central server where we analyse our traffic.

NetFlow on the other hand is a feature on Cisco Layer 3 devices (routers and L3 switches) that captures flows and exports them to an external server for analysis. Unlike SPAN which simply dumps everything it sees on specific ports to the monitoring port, NetFlow will provide more structured information.

NetFlow Versions:

Version 1:

- First implementation, now obsolete, and restricted to IPV4 only.

Versions 2:

- Cisco internal version never released.

Version 3:

- Cisco internal version never released.

Version 4:

- Cisco internal version never released.

Version 6:

- No longer supported by Cisco Encapsulation information.
- Version 6 is not compatible with Cisco routers.

Version 7:

- Cisco-specific version for Catalyst 5000 series switches.
- Version 7 is not compatible with Cisco routers.

Version 8:

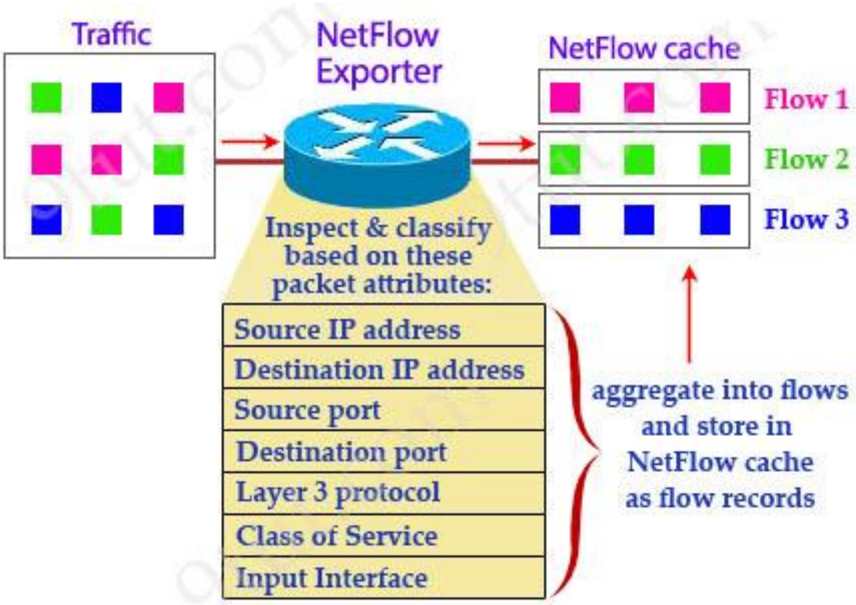
- Choice of aggregation schemes in order to reduce resource usage.

Version 5:

- Fixed format that cannot be added or extended.
- NetFlow version 5 only support IPv4.
- NetFlow version 5 added BGP support.
- Export data from main cache only.
- No real concept of ingress & egress flows.
- NetFlow added flow sequence numbers & additional fields.
- NetFlow Version 5 is standard & most common NetFlow version.

Version 9:

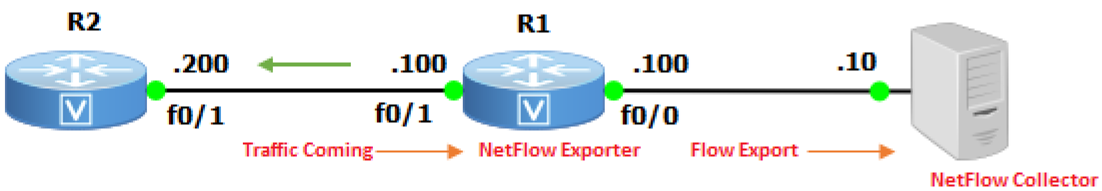
- NetFlow Version 9 support IPV4 and IPv6.
- Not backwards compatible with any previous version.
- Added additional information to flows & template based.
- Exports data from main & aggregation cache.
- NetFlow Version 9 support for MPLS.
- Support flow-record format known as Flexible NetFlow technology.
- NetFlow Version 9 is most important NetFlow version.



NetFlow uses the following fields to identify a unique flow:

- Source IP address
- Source port number
- Destination IP address
- Destination port number
- Layer 3 Protocol Type
- Type of service
- Logical input interface

GNS3:



R1 Basic Configuration	
R1(config)#interface f0/0 R1(config-if)#ip address 192.168.169.100 255.255.255.0 R1(config-if)#no shutdown	R1(config)#interface f0/1 R1(config-if)#ip add 192.168.12.100 255.255.255.0 R1(config-if)#no shutdown
R1(config)#router rip R1(config-router)#network 0.0.0.0	
R2 Basic Configuration	
R2(config)#interface f0/1 R2(config-if)#ip add 192.168.12.200 255.255.255.0 R2(config-if)#no shutdown	R2(config)#router rip R2(config-router)#network 0.0.0.0

NetFlow V5 Configuration On R1
R1(config)#interface f0/0 R1(config-if)#ip flow ingress R1(config-if)#ip route-cache flow //same as ip flow ingress
R1(config)#ip flow-export destination 1.1.1.10 2055
R1(config)#ip flow-export source f0/0
R1(config)#ip flow-export version 5
R1(config)# ip flow-cache timeout active 1 //export flow records every minute
R1(config)#ip flow-cache timeout inactive 10
R1#show ip flow export
R1# show ip flow export template
R1# show flow exporter
R1# show ip cache flow
NetFlow V9 Configuration On R1
R1(config)#interface f0/0 R1(config-if)#ip flow ingress R1(config-if)#ip flow egress
R1(config)#ip flow-export destination 1.1.1.10 2055
R1(config)#ip flow-export source f0/0
R1(config)#ip flow-export version 9
R1(config)# ip flow-cache timeout active 1
R1(config)#ip flow-cache timeout inactive 10
R1#show ip flow export
R1# show ip flow export template
R1# show flow exporter
R1# show ip cache flow

NetFlow Top Talkers is a feature supported on all versions of NetFlow that provide top talkers for performance debugging purposes. It can be useful if there is no collector. NetFlow information can be seen from either the collector or the CLI.

NetFlow Top Taker Configuration on R1
R1(config)# ip flow-top-talkers R1(config-flow-top-talkers)# top 10 R1(config-flow-top-talkers)# sort-by packets R1(config-flow-top-talkers)# sort-by bytes
R1# show ip flow top-talkers