ACI SPAN Guide

Contents

Introduction

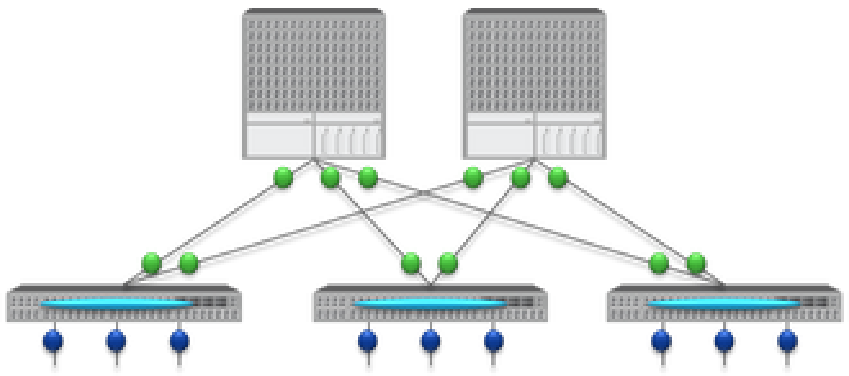
This document describes how to configure Switched Port Analyzer (SPAN) on Cisco Application Centric Infrastructure (ACI).

Background Information

In general, there are three types of SPAN. Local SPAN, Remote SPAN (RSPAN) and Encapsulated Remote SPAN (ERSPAN). The differences between these SPANs are mainly the destination of copy packets. Cisco ACI supports Local SPAN and ERSPAN.

Note: This document assumes that readers are already familiar with SPAN in general such as Local SPAN and ERSPAN differences.

SPAN Type in Cisco ACI



	== TYPE ==	== SRC ==		== DST ==
●	Fabric SPAN	SPAN on Fabric ports on Spine or Leaf	→	ERSPAN (remote IP)
●	Tenant SPAN	SPAN on EPG(=VLAN) on Leaf	→	ERSPAN (remote IP)
●	Access SPAN	SPAN on Access ports on Leaf	→	ERSPAN (remote IP)
			→	Local SPAN (Local port)

※ Infra SPAN = Access SPAN

Cisco ACI has three types of SPAN; Fabric SPAN, Tenant SPAN and Access SPAN. The difference between each SPANs is the source of copy packets.

As mentioned previously,

- **Fabric SPAN** is to capture packets that come in and go out from **interfaces between Leaf and Spine switches**.
- Access SPAN is to capture packets that come in and go out from interfaces between Leaf switches and external devices.
- Tenant SPAN is to capture packets that come in and go out from EndPoint Group (EPG) on ACI Leaf switches.

This SPAN name corresponds to where to be configured on Cisco ACI GUI.

- Fabric SPAN is configured under Fabric > Fabric Policies
- Access SPAN is configured under Fabric > Access Policies
- Tenant SPAN is configured under Tenants > {each tenant}

As for the destination of each SPAN, only Access SPAN is capable of both Local SPAN and ERSPAN. The other two SPAN (Fabric and Tenant) are only capable of ERSPAN.

Limitations and Guidelines

Please review the Limitations & Guidelines from Cisco APIC Troubleshooting Guide. It is mentioned in Troubleshooting Tools and Methodology > Using SPAN.

Configuration

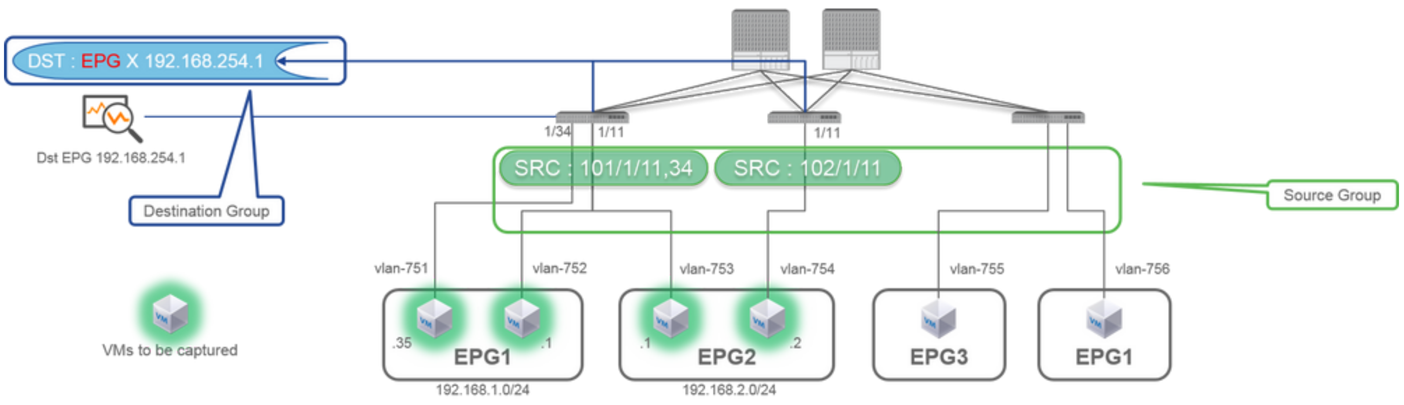
This section introduces brief examples that relate to the configuration for each SPAN Type. There are specific sample cases on how to select the span type in the later section.

SPAN Configuration is also described in [Cisco APIC Troubleshooting Guide: Troubleshooting Tools and methodology > Using SPAN]().

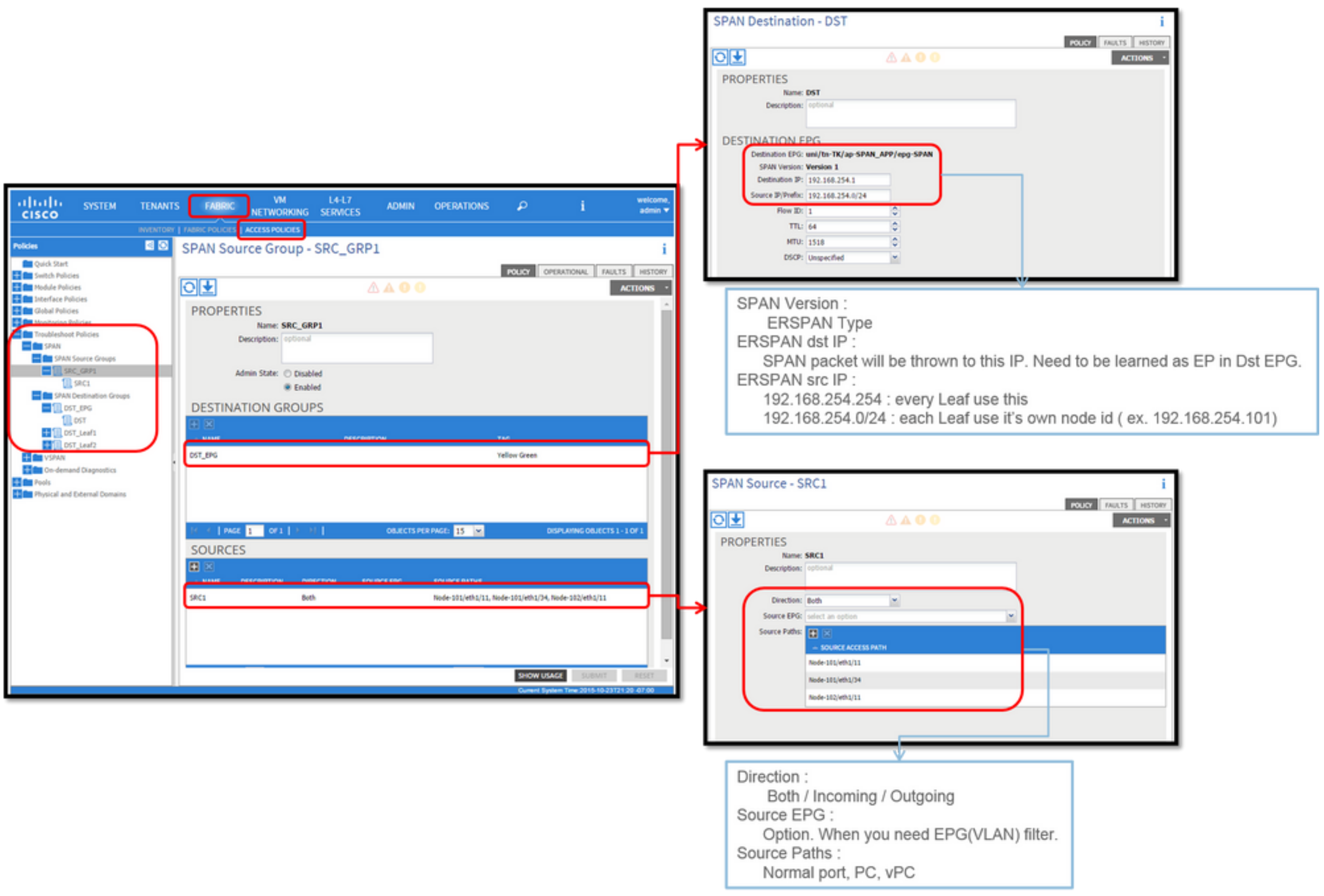
The UI can appear different than the current versions but the config approach is the same.

Access SPAN (ERSPAN)

Sample Topology



Configuration Example



SPAN Source Group - SRC_GRP1

PROPERTIES

Name: SRC_GRP1
Description: optional
Admin State: Disabled [✓ Enabled]

DESTINATION GROUPS

NAME	DESCRIPTION	TAG
DST_EPG		Yellow Green

PAGE 1 OF 1 | OBJECTS PER PAGE: 15 | DISPLAYING OBJECTS 1 - 1 OF 1

SOURCES

NAME	DESCRIPTION	DIRECTION	SOURCE EPG	SOURCE PATHS
SRC1		Both		Node-101/eth1/11, Node-101/eth1/34, Node-102/eth1/11

Current System Time 2015-10-23T21:20 -07:00

SPAN Destination - DST

PROPERTIES

Name: DST
Description: optional

DESTINATION EPG

Destination EPG: uni/tn-TK/ap-SPAN_APP/epg-SPAN
SPAN Version: Version 1
Destination IP: 192.168.254.1
Source IP/Prefix: 192.168.254.0/24
Flow ID: 1
TTL: 64
MTU: 1518
DSCP: Unspecified

SPAN Version :
    ERSPAN Type
ERSPAN dst IP :
    SPAN packet will be thrown to this IP. Need to be learned as EP in Dst EPG.
ERSPAN src IP :
    192.168.254.254 : every Leaf use this
    192.168.254.0/24 : each Leaf use it's own node id (ex. 192.168.254.101)

SPAN Source - SRC1

PROPERTIES

Name: SRC1
Description: optional
Direction: Both
Source EPG: select an option
Source Paths:

SOURCE ACCESS PATH
Node-101/eth1/11
Node-101/eth1/34
Node-102/eth1/11

Direction :
    Both / Incoming / Outgoing
Source EPG :
    Option. When you need EPG(VLAN) filter.
Source Paths :
    Normal port, PC, vPC

Where:

Navigate to FABRIC > ACCESS POLICIES > Troubleshoot Policies > SPAN.

- SPAN Source Groups
- SPAN Destination Groups

SPAN Source Group ties Destination and Sources.

How:

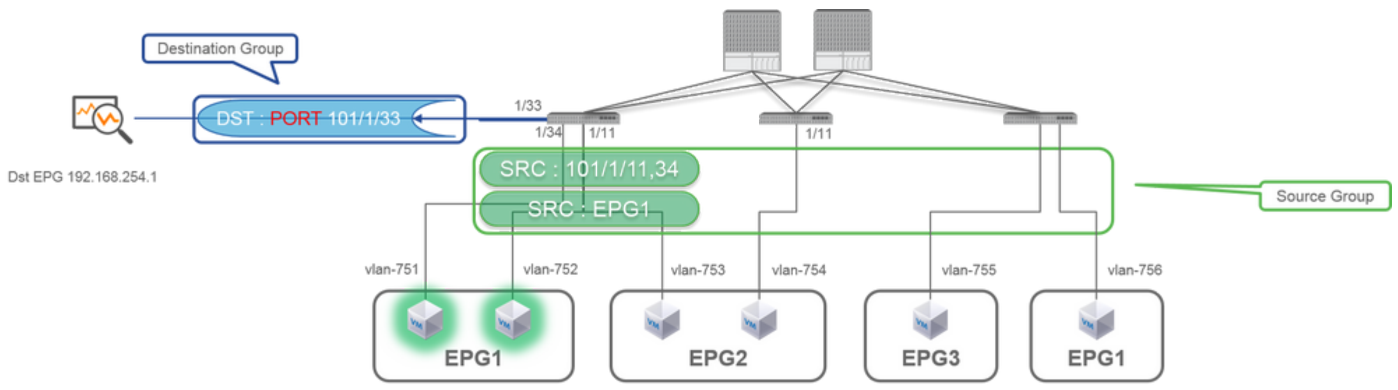
1. Create SPAN Source Group (SRC_GRP1).
2. Create SPAN Source (SRC1) under SPAN Source Group (SRC_GRP1).
3. Configure these parameters for SPAN Source (SRC1).
   - Direction - Source EPG (option)
   - Source Paths (could be multiple interfaces)

4. Create SPAN Destination Group (DST_EPG).
5. Create SPAN Destination (DST).
6. Configure these parameters for SPAN Destination (DST)
   - Destination EPG
   - Destination IP
   - Source IP/Prefix   (This can be any IP. If the prefix is used, node-id of the source node is used for the undefined bits. For example, prefix: 1.0.0.0/8 on node-101 => src IP 1.0.0.101)
   - Other parameters can be left as default
7. Ensure the SPAN Destination Group is tied to an appropriate SPAN Source Group.
8. Ensure Admin Stateis Enabled.

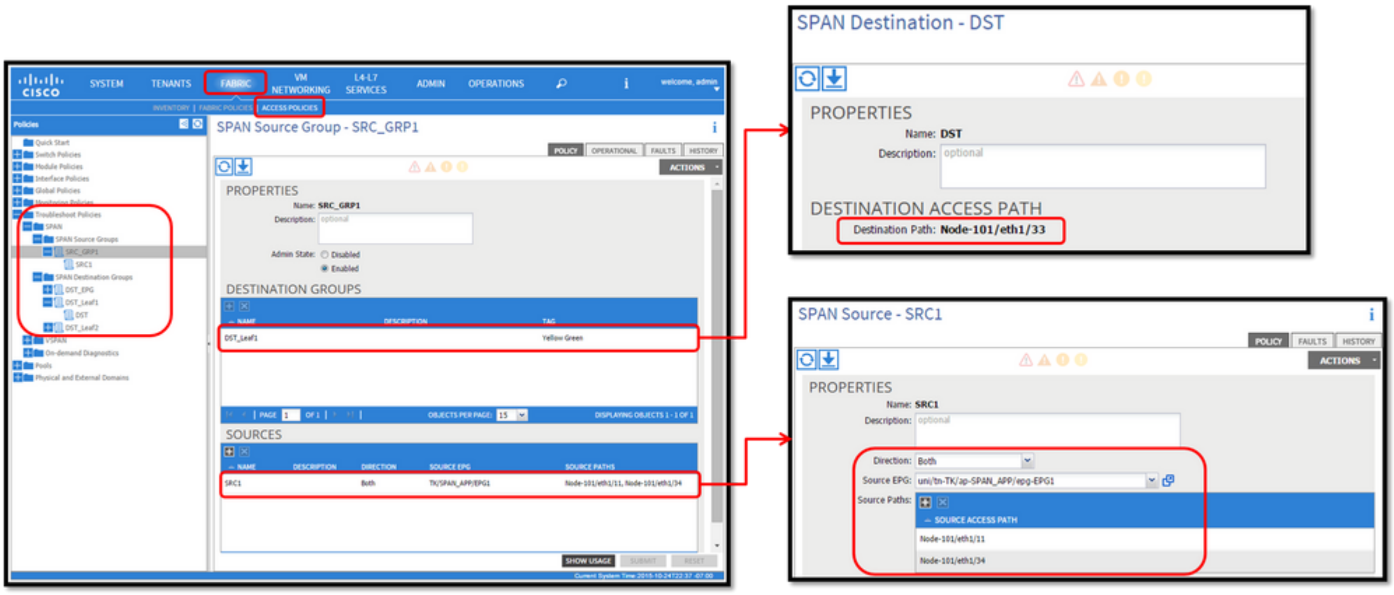
Also please ensure that the destination IP for ERSPAN is learned as an endpoint under the specified destination EPG. In the previously mentioned example, 192.168.254.1 has to be learned under Tenant TK > Application profile SPAN_APP > EPG SPAN. Or the destination IP can be configured as a static EndPoint under this EPG if the destination device is a silent host.

Access SPAN (Local)

Sample Topology



Configuration Example


SPAN Source Group - SRC_GRP1

PROPERTIES

Name: SRC_GRP1

Description: optional

Admin State: Disabled [✓ Enabled]

DESTINATION GROUPS

NAME	DESCRIPTION	TAG
DST_Leaf1		Yellow Green

SOURCES

NAME	DESCRIPTION	DIRECTION	SOURCE EPG	SOURCE PATHS
SRC1		Both	TK/SPAN_APP/EPG1	Node-101/eth1/11, Node-101/eth1/34

SPAN Destination - DST

PROPERTIES

Name: DST

Description: optional

DESTINATION ACCESS PATH

Destination Path: Node-101/eth1/33

SPAN Source - SRC1

PROPERTIES

Name: SRC1

Description: optional

Direction: Both

Source EPG: uni/tn-TK/ap-SPAN_APP/epg-EPG1

Source Paths:

SOURCE ACCESS PATH
Node-101/eth1/11
Node-101/eth1/34

- Where:
  Fabric > ACCESS POLICIES > Troubleshoot Policies > SPAN

- SPAN Source Groups

- SPAN Destination Groups

SPAN Source Group ties Destination and Sources.

- How :

1. Create SPAN Source Group (SRC_GRP1)
2. Create SPAN Source(SRC1) under SPAN Source Group (SRC_GRP1)
3. Configure these parameters for SPAN Source (SRC1)
   - Direction
   - Source EPG   (option)
   - Source Paths   (could be multiple interfaces)
   please refer to the picture for details of each parameter.
4. Create SPAN Destination Group(DST_Leaf1)
5. Create **SPAN Destination**(DST)
6. Configure these parameters for SPAN Destination (DST)
   - Destination interface and node.
7. Ensure the SPAN Destination Group is tied to an appropriate SPAN Source Group.
8. Ensure Admin State is Enabled.
   SPAN stops when you select Disabled on this Admin State. There is no need to delete all policies if you re-use them later.

The destination interface does not require any configuration by Interface Policy Groups. It works when you plug a cable into the interface on ACI Leaf.

Limitations:

- For Local SPAN, a destination interface and source interfaces must be configured on the same Leaf.
- The destination interface does not require it to be on an EPG as long as it is UP.

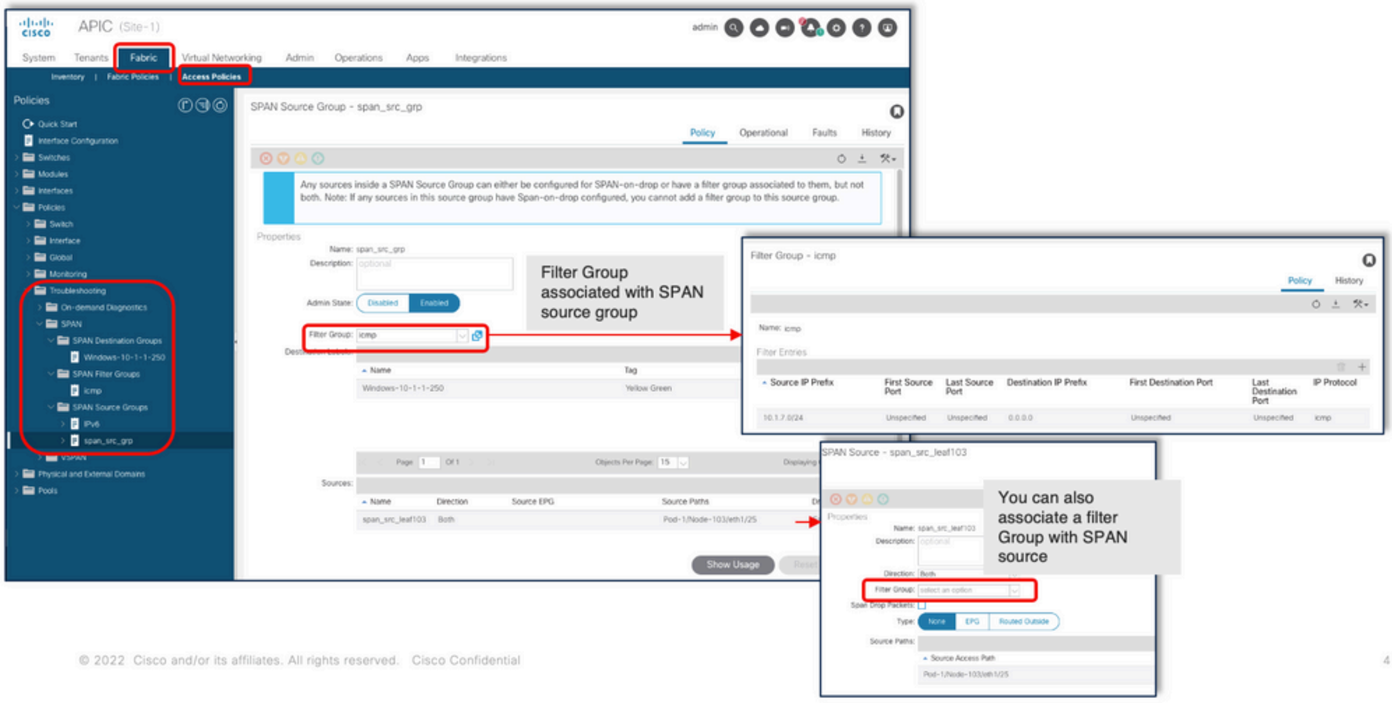
- When the virtual Port-Channel (vPC) interface is specified as a source port, Local SPAN cannot be used
  However, there is a workaround. On a first-generation leaf, an individual physical port that is a member of vPC or PC can be configured as a SPAN source. With this Local SPAN can be used for traffic on vPC ports.
  This option, however, is not available on a second-generation leaf (CSCvc11053). Instead, support for SPAN on "VPC component PC" was added viaCSCvc44643in 2.1(2e), 2.2(2e) and forward. With this, any generation leaf can configure a port channel, which is a member of vPC, as a SPAN source. This allows any generation leaf to use Local SPAN for traffic on vPC ports.
- Specifying the individual ports of a port channel on second-generation leaves cause only a subset of the packets to be spanned (also due toCSCvc11053).
- PC and vPC cannot be used as the destination port for Local SPAN. From 4.1(1), the PC can be used as a destination port for Local SPAN.

Access SPAN - With ACL Filters

You can use ACL filters on access span sources. This feature provides the ability to SPAN a particular flow or flow of traffic in/out of a SPAN source.
Users can apply the SPAN Acl(s) to a source when there is a need to SPAN flow specific traffic.
It is not supported in Fabric SPAN and Tenant Span source groups/sources.
Care must be taken when you add filter entries in a filter group since it could add tcam entries for every source that currently uses the filter group.

A Filter Group can be associated to:
-Span Source: the filter group is used to filter traffic on ALL interfaces defined under this Span Source.
-Span Source Group: the filter group (say x) is used to filter traffic on ALL interfaces defined under each of the Span Source(s) of this Span Source Group.

In this configuration snapshot, the filter group is applied to the Span source group.



SPAN Source Group - span_src_grp

Any sources inside a SPAN Source Group can either be configured for SPAN-on-drop or have a filter group associated to them, but not both. Note: If any sources in this source group have Span-on-drop configured, you cannot add a filter group to this source group.

Properties
Name: span_src_grp
Description: optional
Admin State: Disabled [✓ Enabled]
Filter Group: icmp
Destination Labels:

Name	Tag
Windows-10-1-1-250	Yellow Green

Page 1 Of 1 Objects Per Page: 15 Displaying

Sources:

Name	Direction	Source EPG	Source Paths
span_src_leaf103	Both		Pod-1/Node-103/eth1/25

Filter Group - icmp

Name: icmp

Filter Entries

Source IP Prefix	First Source Port	Last Source Port	Destination IP Prefix	First Destination Port	Last Destination Port	IP Protocol
10.1.7.0/24	Unspecified	Unspecified	0.0.0.0	Unspecified	Unspecified	icmp

SPAN Source - span_src_leaf103

Properties
Name: span_src_leaf103
Description: optional
Direction: Both
Filter Group: select an option
Span Drop Packets: ☐
Type: [✓ None] EPG Routed Outside
Source Paths:

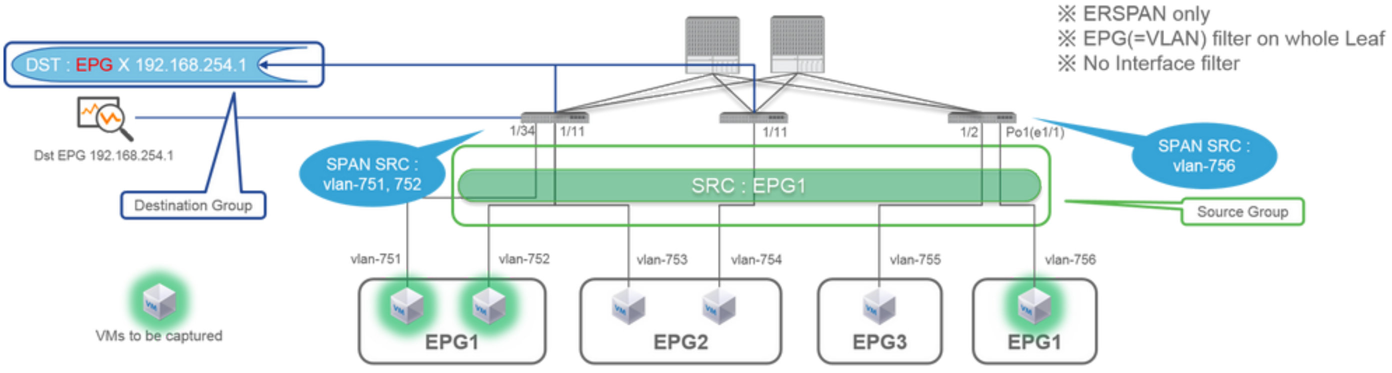
Source Access Path
Pod-1/Node-103/eth1/25

In the case where a particular Span Source already associates with a Filter Group (say y), that filter group (y) is used instead to filter group on all interfaces under this specific Span Source

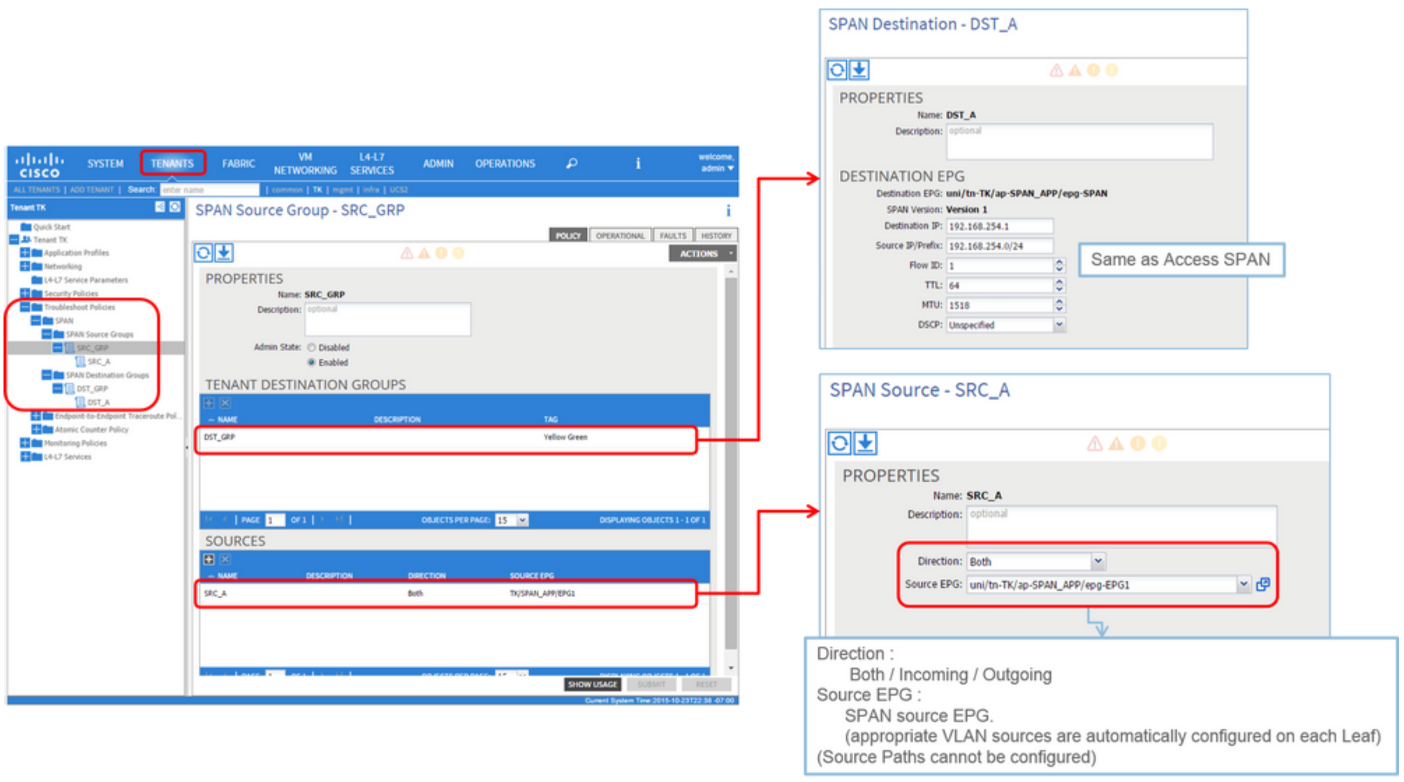
- A Filter group that is applied at a source group automatically applies to all sources in that source group.
- A Filter group that is applied at a source is applicable to that source only.
- A filter group is applied at both the source group and a source in that source group, the filter group applied at the source takes precedence.
- A filter group applied at a source is deleted, filter group applied at the parent source group is automatically applied.
- A filter group applied at a source group is deleted, it is deleted from all sources currently that inherit in that source group.

Tenant SPAN (ERSPAN)

Sample Topology



Configuration Example



SPAN Source Group - SRC_GRP

PROPERTIES

Name: SRC_GRP
Description: optional
Admin State: Disabled [✓ Enabled]

TENANT DESTINATION GROUPS

NAME	DESCRIPTION	TAG
DST_GRP		Yellow Green

PAGE 1 OF 1 | OBJECTS PER PAGE: 15 | DISPLAYING OBJECTS 1 - 1 OF 1

SOURCES

NAME	DESCRIPTION	DIRECTION	SOURCE EPG
SRC_A		Both	TK/SPAN_APP/EPG1

Current System Time 2015-10-23T22:38 -07:00

SPAN Destination - DST_A

PROPERTIES

Name: DST_A
Description: optional

DESTINATION EPG

Destination EPG: uni/tn-TK/ap-SPAN_APP/epg-SPAN
SPAN Version: Version 1
Destination IP: 192.168.254.1
Source IP/Prefix: 192.168.254.0/24
Flow ID: 1
TTL: 64
MTU: 1518
DSCP: Unspecified

Same as Access SPAN

SPAN Source - SRC_A

PROPERTIES

Name: SRC_A
Description: optional
Direction: Both
Source EPG: uni/tn-TK/ap-SPAN_APP/epg-EPG1

Direction :
Both / Incoming / Outgoing
Source EPG :
SPAN source EPG.
(appropriate VLAN sources are automatically configured on each Leaf)
(Source Paths cannot be configured)

- Where:

Tenants > {tenant name} > Troubleshoot Policies > SPAN

- SPAN Source Groups

- SPAN Destination Groups

SPAN Source Group ties Destination and Sources.

- How:

1. Create SPAN Source Group (SRC_GRP)
2. Create SPAN Source (SRC_A) under SPAN Source Group (SRC_GRP)
3. Configure these parameters for SPAN Source (SRC_A)
   - Direction
   - Source EPG

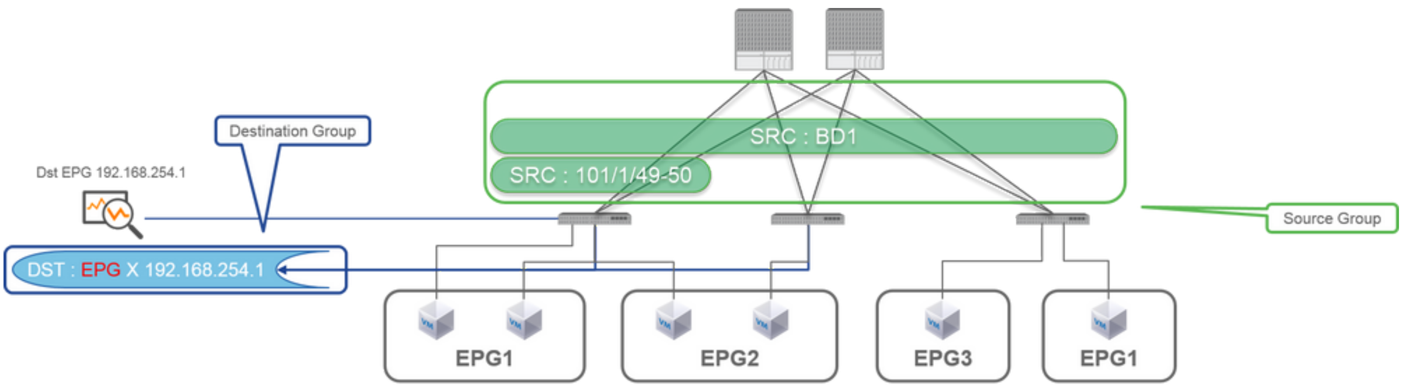
   Please refer to the picture for details of each parameter.
4. Create SPAN Destination Group (DST_GRP)
5. Create SPAN Destination (DST_A)
6. Configure these parameters for SPAN Destination(DST_A)
   - Destination EPG
   - Destination IP
   - Source IP/Prefix
   - Other parameters can be left as default

   Please refer to the picture for details of each parameter.
7. Ensure SPAN Destination Group is tied to an appropriate SPAN Source Group.
8. Ensure Admin State is Enabled.

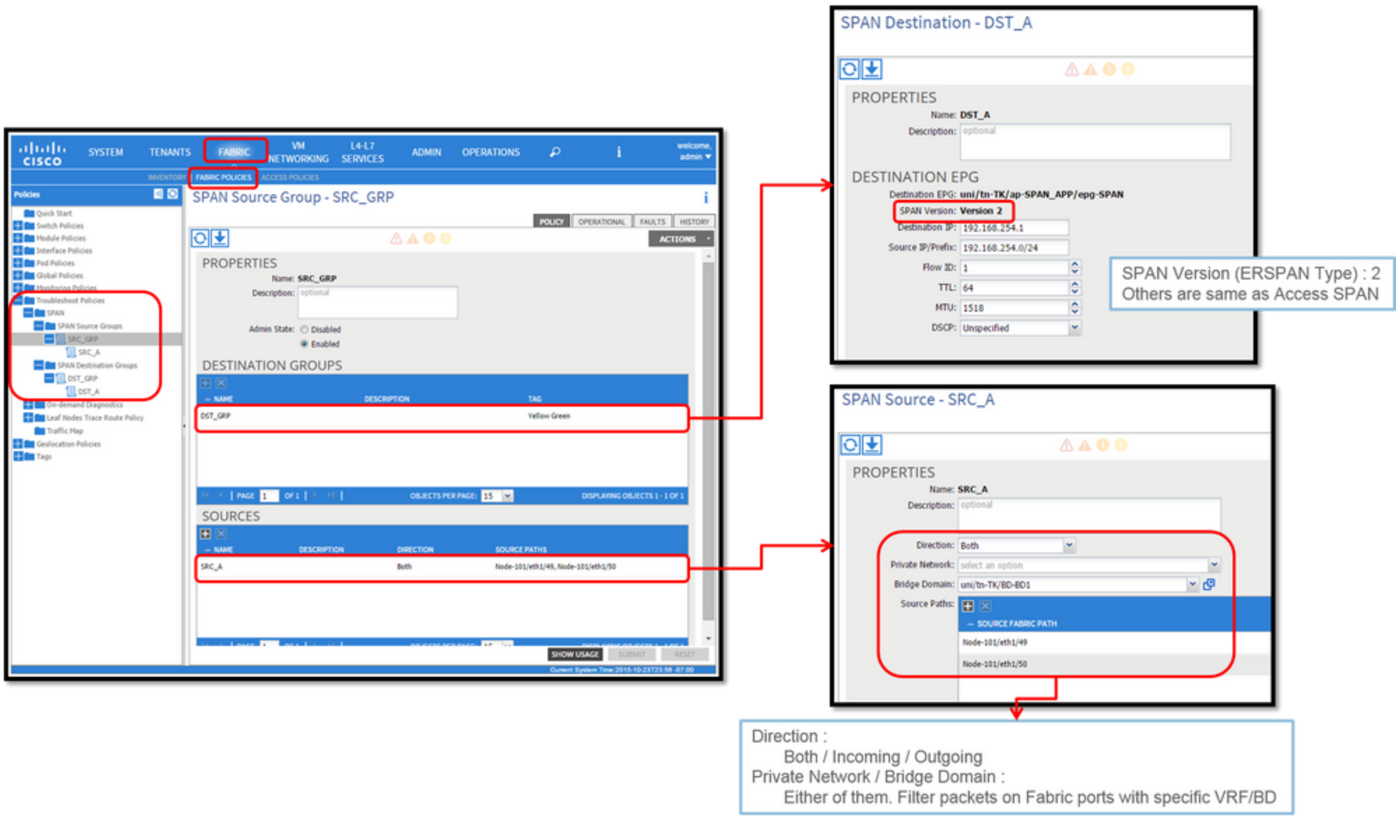
   SPAN stops when you select Disabled on this Admin State. There is no need to delete all policies if you re-use them later.

Fabric SPAN (ERSPAN)

Sample Topology



Configuration Example



SPAN Source Group - SRC_GRP

PROPERTIES
Name: SRC_GRP
Description: optional
Admin State: Disabled [✓ Enabled]

DESTINATION GROUPS

NAME	DESCRIPTION	TAG
DST_GRP		Yellow Green

SOURCES

NAME	DESCRIPTION	DIRECTION	SOURCE PATHS
SRC_A		Both	Node-101/eth1/49, Node-101/eth1/50

SPAN Destination - DST_A

PROPERTIES
Name: DST_A
Description: optional

DESTINATION EPG
Destination EPG: uni/tn-TK/ap-SPAN_APP/epg-SPAN
SPAN Version: Version 2
Destination IP: 192.168.254.1
Source IP/Prefix: 192.168.254.0/24
Flow ID: 1
TTL: 64
MTU: 1518
DSCP: Unspecified

SPAN Version (ERSPAN Type) : 2
Others are same as Access SPAN

SPAN Source - SRC_A

PROPERTIES
Name: SRC_A
Description: optional
Direction: Both
Private Network: select an option
Bridge Domain: uni/tn-TK/BD-BD1
Source Paths:

SOURCE FABRIC PATH
Node-101/eth1/49
Node-101/eth1/50

Direction :
Both / Incoming / Outgoing
Private Network / Bridge Domain :
Either of them. Filter packets on Fabric ports with specific VRF/BD

- Where:

Fabric > FABRIC POLICIES > Troubleshoot Policies > SPAN

- Fabric

- SPAN Destination Groups

SPAN Source Group ties Destination and Sources

- How:

1. Create SPAN Source Group (SRC_GRP)
2. Create SPAN Source (SRC_A) under SPAN Source Group (SRC_GRP)
3. Configure these parameters for SPAN Source (SRC_A)
   - Direction
   - Private Network (option)
   - Bridge Domain (option)
   - Source Paths (could be multiple interfaces)

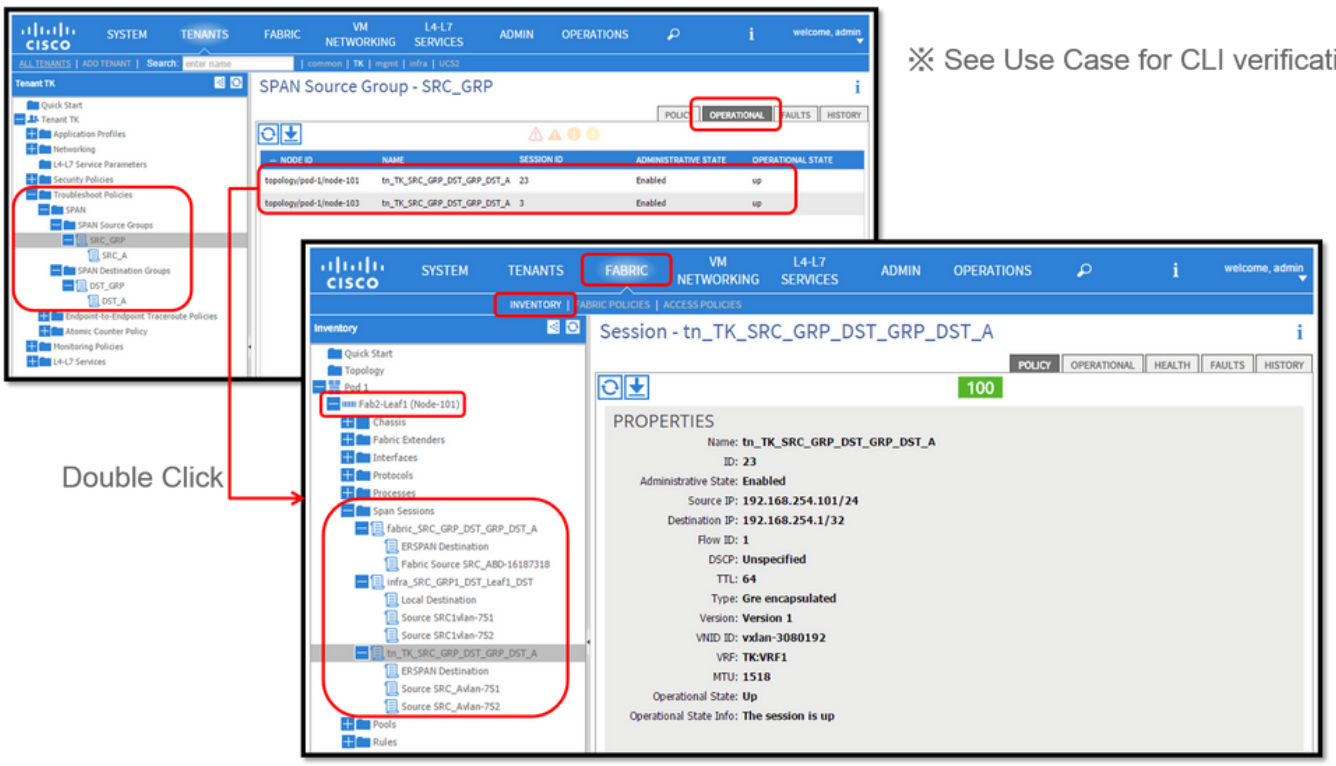
   please refer to the picture for details of each parameter.
4. Create SPAN Destination Group (DST_GRP)
5. Create SPAN Destination (DST_A)
6. Configure these parameters for SPAN Destination (DST_A)
   - Destination EPG
   - Destination IP
   - Source IP/Prefix
   - Other parameters can be left as default

   please refer to the picture for details of each parameter.
7. Ensure SPAN Destination Group is tied to an appropriate SPAN Source Group.
8. Ensure Admin State is Enabled.

SPAN stops when you select Disabled on this Admin State. There is no need to delete all policies if you re-use them later.

Although it is described in a later section "ERSPAN Version (type)", you can tell ERSPAN version II is used for Fabric SPAN and version I is used for Tenant and Access SPAN.

GUI Verification



SPAN Source Group - SRC_GRP

NODE ID	NAME	SESSION ID	ADMINISTRATIVE STATE	OPERATIONAL STATE
topology/pod-1/node-101	tn_TK_SRC_GRP_DST_GRP_DST_A	23	Enabled	up
topology/pod-1/node-103	tn_TK_SRC_GRP_DST_GRP_DST_A	3	Enabled	up

Session - tn_TK_SRC_GRP_DST_GRP_DST_A

PROPERTIES

```
                     Name: tn_TK_SRC_GRP_DST_GRP_DST_A
                       ID: 23
     Administrative State: Enabled
                Source IP: 192.168.254.101/24
           Destination IP: 192.168.254.1/32
                  Flow ID: 1
                     DSCP: Unspecified
                      TTL: 64
                     Type: Gre encapsulated
                  Version: Version 1
                  VNID ID: vxlan-3080192
                      VRF: TK:VRF1
                      MTU: 1518
        Operational State: Up
   Operational State Info: The session is up
```

- Verification of SPAN Configuration Policy

1. Fabric > ACCESS POLICIES > Troubleshoot Policies > SPAN > SPAN Source Groups > Operational tab
2. Fabric > FABRIC POLICIES > Troubleshoot Policies > SPAN > SPAN Source Groups > Operational tab
3. Tenants > {tenant name} > Troubleshoot Policies > SPAN > SPAN Source Groups > Operational tab

Please ensure Operational State is up.

- Verification on SPAN Session on the node itself

1. Double-click on each session from SPAN Configuration Policyor Fabric > INVENTORY > Node > Span Sessions > { SPAN session name }

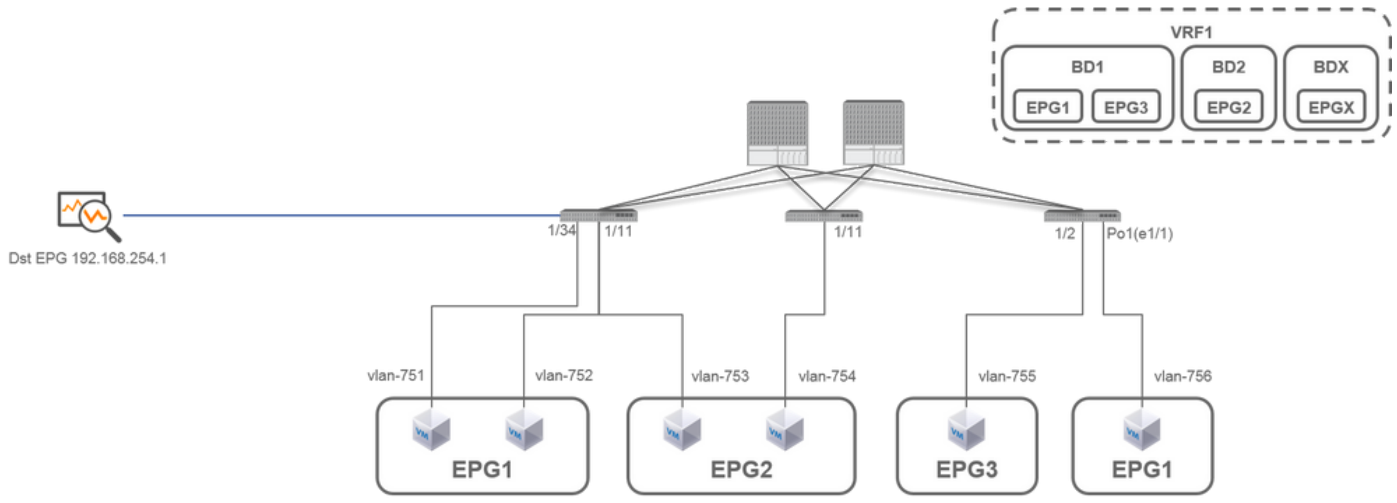
Please ensure Operational State is up.

SPAN Session naming convention:

- Fabric SPAN: fabric_xxxx

- Access SPAN: infra_xxxx
- Tenant SPAN: tn_xxxx

Select the ACI SPAN Type



In this section, detailed scenarios are described for each ACI SPAN type (Access, Tenant, Fabric) The base topology for each scenario is mentioned in the previous section.

If you understand these scenarios, you can select the appropriate ACI SPAN type for your requirement such as packets on only specific interfaces must be captured or all packets on a specific EPG regardless of interfaces must be captured, and more.

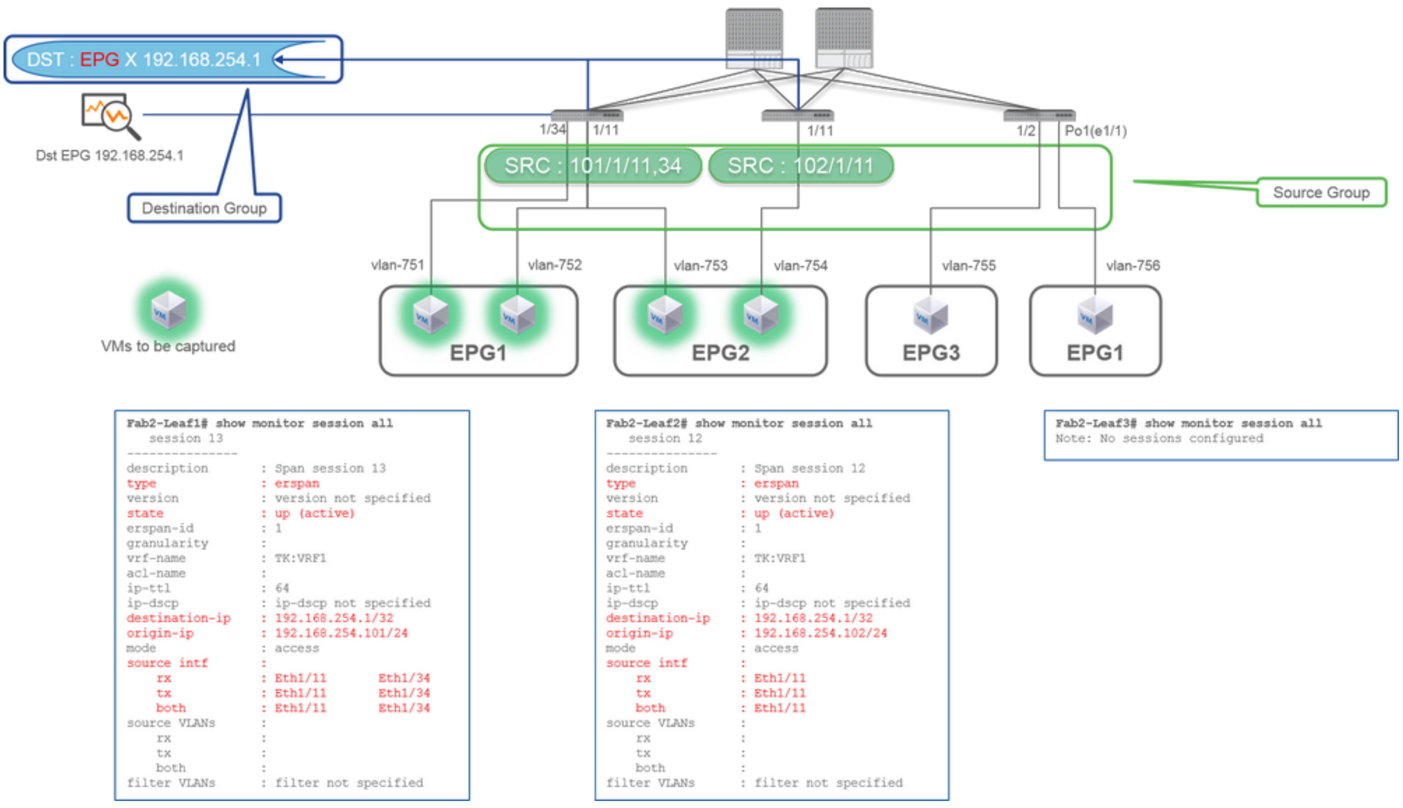
In Cisco ACI, SPAN is configured with the source group and destination group. The Source group contains multiple source factors such as interfaces or EPG. The destination group contains destination information such as the destination interface for Local SPAN or destination IP for ESPAN.

After packets are captured, please see the section "How to Read SPAN Data" to decode captured packets.

Note: Please focus on VMs highlighted with a green light in each topology. Each scenario is to capture packets from these highlighted VMs.

Access SPAN (ERSPAN)

Case 1. Src "Leaf1 e1/11 e1/34 & Leaf2 e1/11" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 13
---------------
description          : Span session 13
type                 : erspan
version              : version not specified
state                : up (active)
erspan-id            : 1
granularity          :
vrf-name             : TK:VRF1
acl-name             :
ip-ttl               : 64
ip-dscp              : ip-dscp not specified
destination-ip       : 192.168.254.1/32
origin-ip            : 192.168.254.101/24
mode                 : access
source intf          :
    rx               : Eth1/11       Eth1/34
    tx               : Eth1/11       Eth1/34
    both             : Eth1/11       Eth1/34
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : filter not specified
```

```
Fab2-Leaf2# show monitor session all
   session 12
---------------
description          : Span session 12
type                 : erspan
version              : version not specified
state                : up (active)
erspan-id            : 1
granularity          :
vrf-name             : TK:VRF1
acl-name             :
ip-ttl               : 64
ip-dscp              : ip-dscp not specified
destination-ip       : 192.168.254.1/32
origin-ip            : 192.168.254.102/24
mode                 : access
source intf          :
    rx               : Eth1/11
    tx               : Eth1/11
    both             : Eth1/11
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : filter not specified
```

```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

- Source Group Leaf1 e1/11Leaf1 e1/34Leaf2 e1/11
- Destination Group 192.168.254.1 on EPG X

Access SPAN can specify multiple interfaces for a single SPAN session. It can capture all packets that come in or go out from specified interfaces regardless of their EPG.

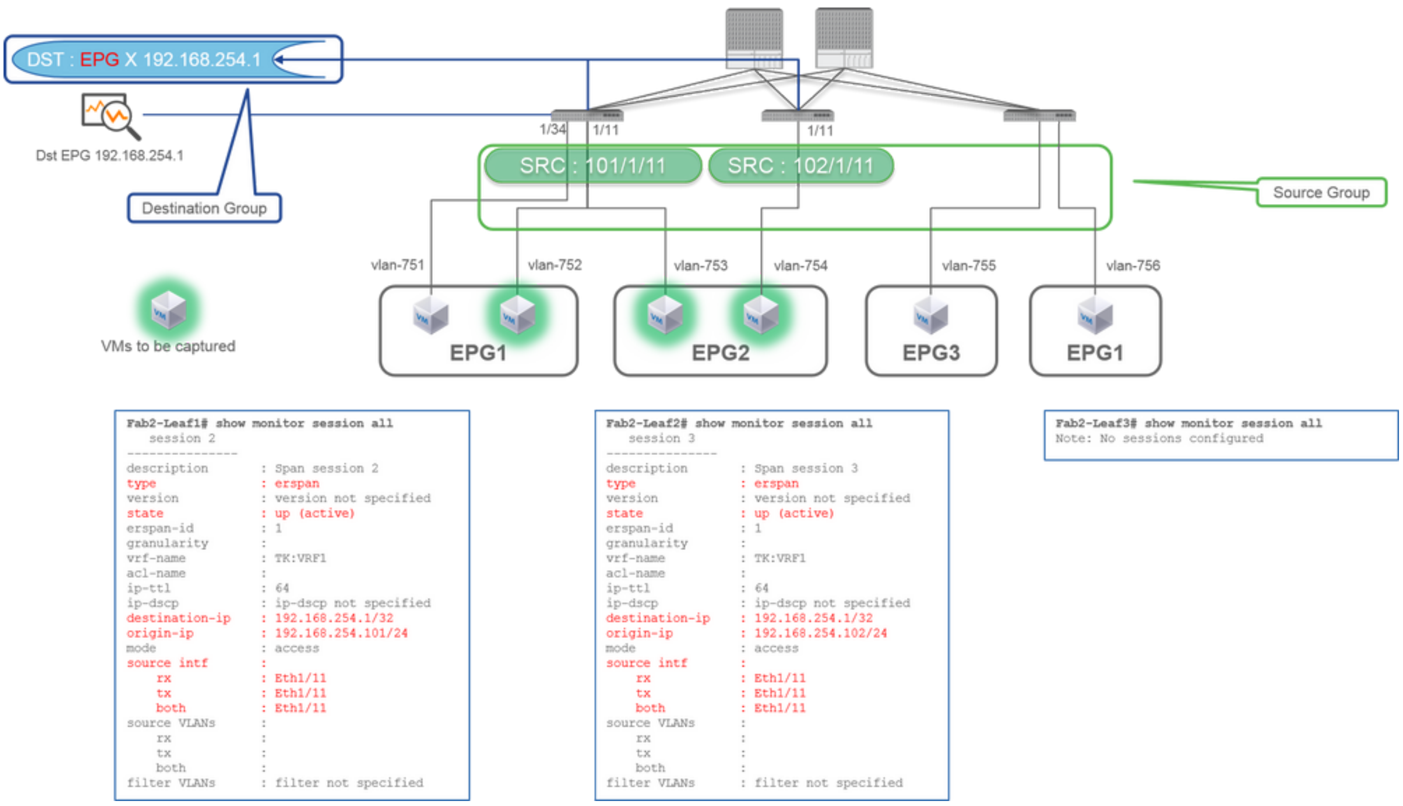
When multiple interfaces are specified as a source group from multiple Leaf switches, the destination group must be ERSPAN, not Local SPAN.

In this example, it copies packets from all VMs on EPG1 and EPG2.

CLI Check Point

- Please make sure the status is "up (active)"
- "destination-ip" is destination IP for ERSPAN
- "origin-ip" is source IP for ERSPAN

Case 2. Src "Leaf1 e1/11 & Leaf2 e1/11" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 2
---------------
description        : Span session 2
type               : erspan
version            : version not specified
state              : up (active)
erspan-id          : 1
granularity        :
vrf-name           : TK:VRF1
acl-name           :
ip-ttl             : 64
ip-dscp            : ip-dscp not specified
destination-ip     : 192.168.254.1/32
origin-ip          : 192.168.254.101/24
mode               : access
source intf        :
    rx             : Eth1/11
    tx             : Eth1/11
    both           : Eth1/11
source VLANs       :
    rx             :
    tx             :
    both           :
filter VLANs       : filter not specified
```

```
Fab2-Leaf2# show monitor session all
   session 3
---------------
description        : Span session 3
type               : erspan
version            : version not specified
state              : up (active)
erspan-id          : 1
granularity        :
vrf-name           : TK:VRF1
acl-name           :
ip-ttl             : 64
ip-dscp            : ip-dscp not specified
destination-ip     : 192.168.254.1/32
origin-ip          : 192.168.254.102/24
mode               : access
source intf        :
    rx             : Eth1/11
    tx             : Eth1/11
    both           : Eth1/11
source VLANs       :
    rx             :
    tx             :
    both           :
filter VLANs       : filter not specified
```

```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

- **Source Group** Leaf1 e1/11Leaf2 e1/11
- **Destination Group** 192.168.254.1 on EPG X

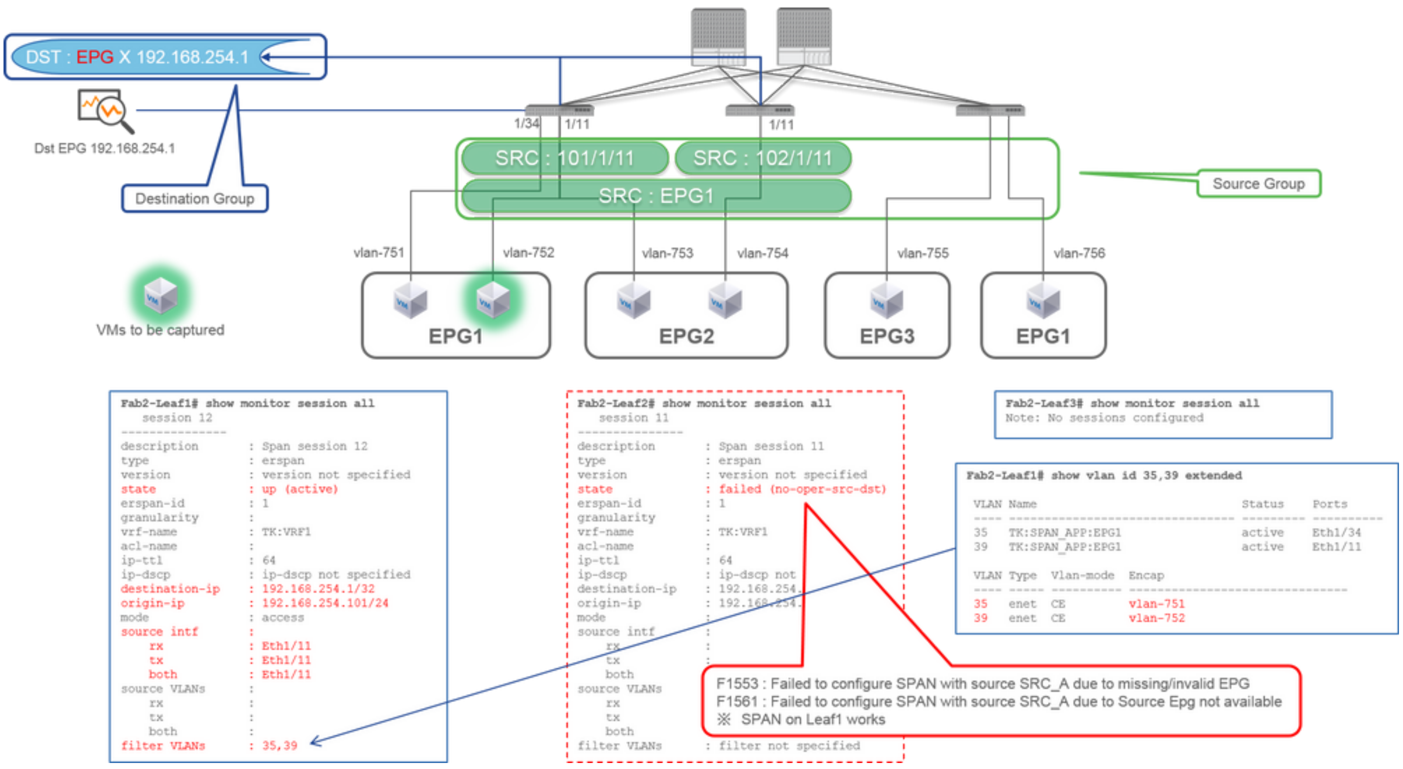
In this example, Leaf1 e1/34 is removed from the SPAN Source Group configured at previous Case1.

The key point in this example is that Access SPAN can specify source interfaces regardless of EPG.

CLI Check Point

- source interface on Leaf1 is changed to "Eth1/11" from "Eth1/11   Eth1/34"

Case 3. Src "Leaf1 e1/11 & Leaf2 e1/11 & EPG1 filter" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 12
---------------
description         : Span session 12
type                : erspan
version             : version not specified
state               : up (active)
erspan-id           : 1
granularity         :
vrf-name            : TK:VRF1
acl-name            :
ip-ttl              : 64
ip-dscp             : ip-dscp not specified
destination-ip      : 192.168.254.1/32
origin-ip           : 192.168.254.101/24
mode                : access
source intf         :
    rx              : Eth1/11
    tx              : Eth1/11
    both            : Eth1/11
source VLANs        :
    rx              :
    tx              :
    both            :
filter VLANs        : 35,39
```

```
Fab2-Leaf2# show monitor session all
   session 11
---------------
description         : Span session 11
type                : erspan
version             : version not specified
state               : failed (no-oper-src-dst)
erspan-id           : 1
granularity         :
vrf-name            : TK:VRF1
acl-name            :
ip-ttl              : 64
ip-dscp             : ip-dscp not
destination-ip      : 192.168.254.
origin-ip           : 192.168.254.
mode                :
source intf         :
    rx              :
    tx              :
    both
source VLANs
    rx
    tx
    both
filter VLANs        : filter not specified
```

```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf1# show vlan id 35,39 extended

 VLAN Name                             Status    Ports
 ---- -------------------------------- --------- ----------
 35   TK:SPAN_APP:EPG1                 active    Eth1/34
 39   TK:SPAN_APP:EPG1                 active    Eth1/11

 VLAN Type  Vlan-mode  Encap
 ---- ----- ---------- -------------------------------
 35   enet  CE         vlan-751
 39   enet  CE         vlan-752
```

- **Source Group** Leaf1 e1/11Leaf2 e1/11Filter EPG1
- **Destination Group** 192.168.254.1 on EPG X

This example shows that Access SPAN also can specify a specific EPG on the source ports. This is useful when multiple EPGs flow on a single interface and it is required to capture traffic only for EPG1 on this interface.

Since EPG1 is not deployed on Leaf2, SPAN for Leaf2 fails with faults F1553 and F1561. However, SPAN on Leaf1 still works.

Also, two VLAN filters are automatically added for the SPAN session on Leaf1 because EPG1 uses two VLANs (VLAN-751,752) on Leaf1.

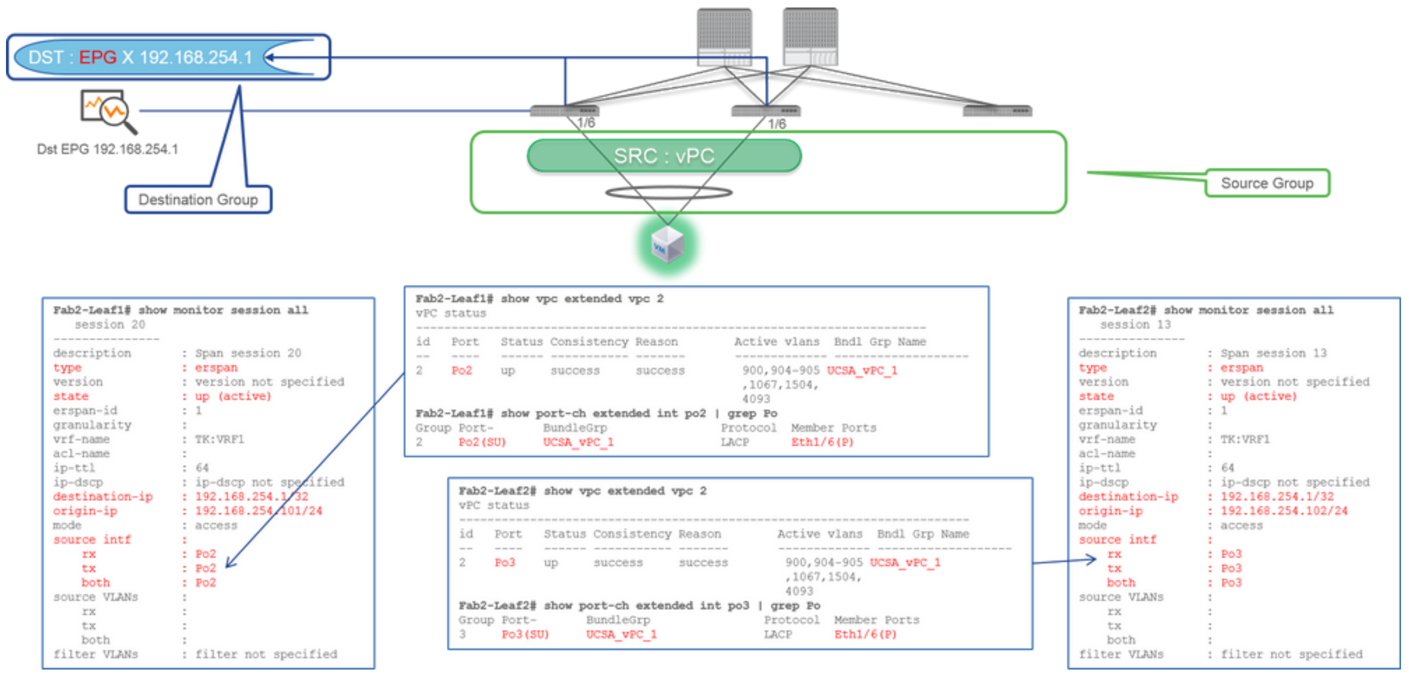
Please be noted that the VLAN ID on CLI (35, 39) is the internal VLAN so-called PI-VLAN(Platform Independent VLAN) which is not the actual ID on the wire. As shown in the picture, **show vlan extended** command shows the mapping of the actual encap VLAN ID and PI-VLAN.

This SPAN session allows us to capture packets only for EPG1(VLAN-752) on Leaf1 e1/11 even though EPG2 (VLAN-753) flows on the same interface.

CLI Check Point

- Filter VLANs are added as per the EPGs that are used for the filter.
- If there are no corresponding EPGs on Leaf, the SPAN session on that Leaf fails.

Case 4. Src "Leaf1-Leaf2 vPC" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 20
---------------
description          : Span session 20
type                 : erspan
version              : version not specified
state                : up (active)
erspan-id            : 1
granularity          :
vrf-name             : TK:VRF1
acl-name             :
ip-ttl               : 64
ip-dscp              : ip-dscp not specified
destination-ip       : 192.168.254.1/32
origin-ip            : 192.168.254.101/24
mode                 : access
source intf          :
    rx               : Po2
    tx               : Po2
    both             : Po2
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : filter not specified
```

```
Fab2-Leaf1# show vpc extended vpc 2
vPC status
----------------------------------------------------------------------------
id   Port   Status Consistency Reason        Active vlans  Bndl Grp Name
--   ----   ------ ----------- -------       ------------  ------------------
2    Po2    up     success     success        900,904-905 UCSA_vPC_1
                                              ,1067,1504,
                                              4093
Fab2-Leaf1# show port-ch extended int po2 | grep Po
Group Port-        BundleGrp                 Protocol  Member Ports
2     Po2(SU)      UCSA_vPC_1                LACP      Eth1/6(P)
```

```
Fab2-Leaf2# show vpc extended vpc 2
vPC status
----------------------------------------------------------------------------
id   Port   Status Consistency Reason        Active vlans  Bndl Grp Name
--   ----   ------ ----------- -------       ------------  ------------------
2    Po3    up     success     success        900,904-905 UCSA_vPC_1
                                              ,1067,1504,
                                              4093
Fab2-Leaf2# show port-ch extended int po3 | grep Po
Group Port-        BundleGrp                 Protocol  Member Ports
3     Po3(SU)      UCSA_vPC_1                LACP      Eth1/6(P)
```

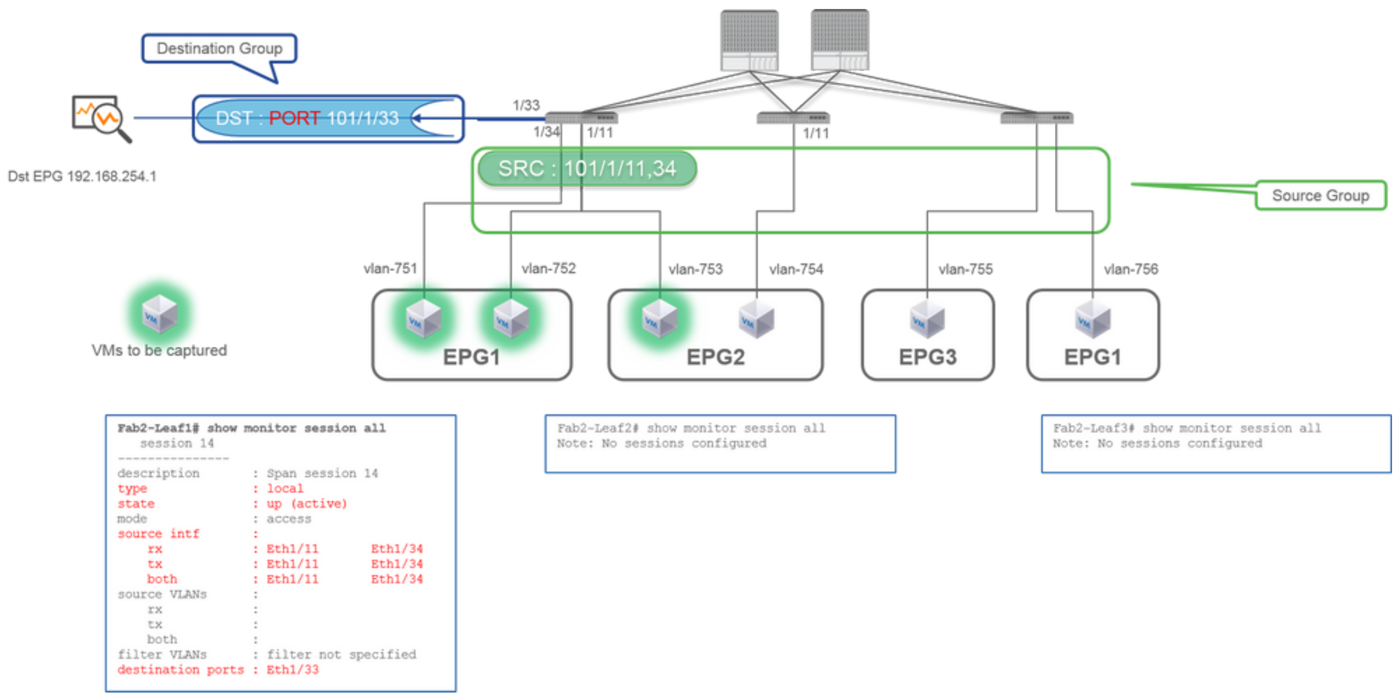
```
Fab2-Leaf2# show monitor session all
   session 13
---------------
description          : Span session 13
type                 : erspan
version              : version not specified
state                : up (active)
erspan-id            : 1
granularity          :
vrf-name             : TK:VRF1
acl-name             :
ip-ttl               : 64
ip-dscp              : ip-dscp not specified
destination-ip       : 192.168.254.1/32
origin-ip            : 192.168.254.102/24
mode                 : access
source intf          :
    rx               : Po3
    tx               : Po3
    both             : Po3
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : filter not specified
```

- **Source Group** Leaf1 - 2e1/11
- **Destination Group** 192.168.254.1 on EPG X

When the vPC interface is configured as a source, a destination must be remote IP (ERSPAN) not the interface (Local SPAN)

Access SPAN (Local SPAN)

Case 1. Src "Leaf1 e1/11 e1/34" | Dst "Leaf1 e1/33"



```
Fab2-Leaf1# show monitor session all
   session 14
---------------
description          : Span session 14
type                 : local
state                : up (active)
mode                 : access
source intf          :
    rx               : Eth1/11       Eth1/34
    tx               : Eth1/11       Eth1/34
    both             : Eth1/11       Eth1/34
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : filter not specified
destination ports : Eth1/33
```

```
Fab2-Leaf2# show monitor session all
Note: No sessions configured
```

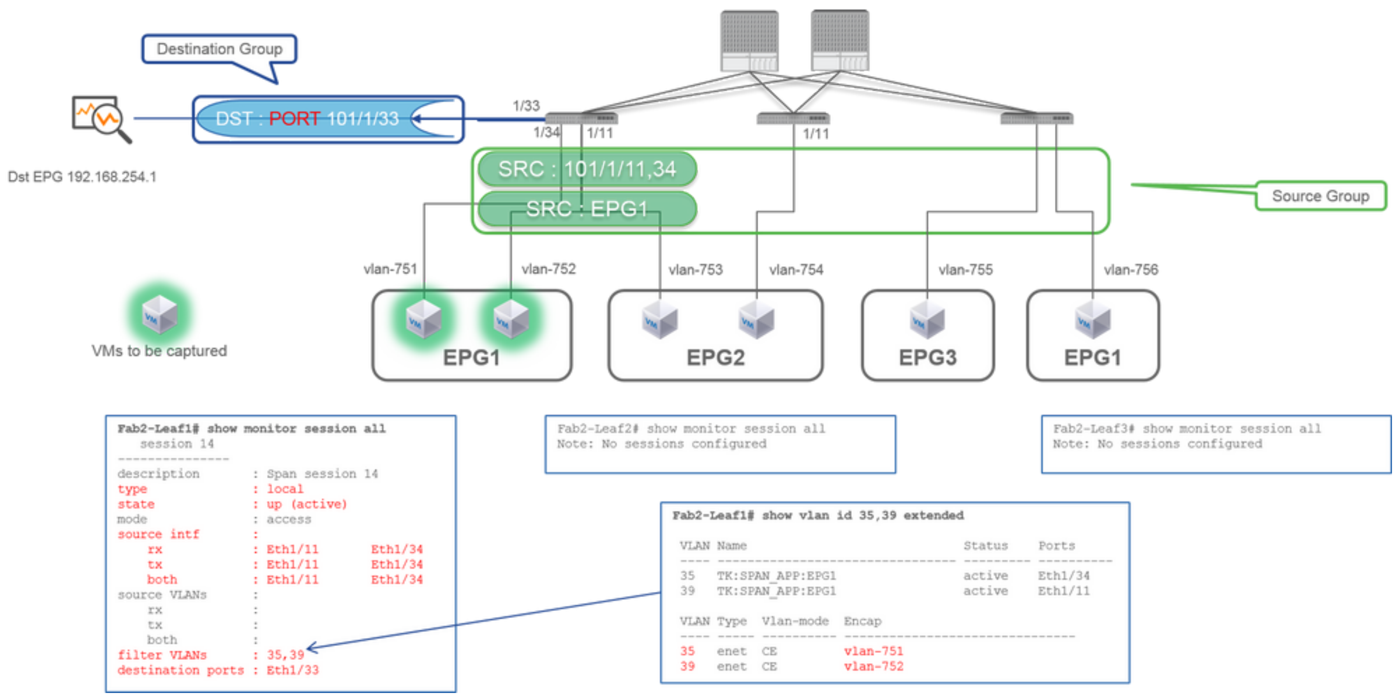
```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

- **Source Group** Leaf1 e1/11Leaf1 e1/34
- **Destination Group** Leaf1 e1/33

Access SPAN can also use Local SPAN (that is a specific interface as a destination)

However, in this case, source interfaces must be on the same Leaf as the destination interface.

Case 2. Src "Leaf1 e1/11 e1/34 & EPG1 filter | Dst " Leaf1 e1/33"



```
Fab2-Leaf1# show monitor session all
   session 14
---------------
description          : Span session 14
type                 : local
state                : up (active)
mode                 : access
source intf          :
    rx               : Eth1/11       Eth1/34
    tx               : Eth1/11       Eth1/34
    both             : Eth1/11       Eth1/34
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : 35,39
destination ports : Eth1/33
```

```
Fab2-Leaf2# show monitor session all
Note: No sessions configured
```

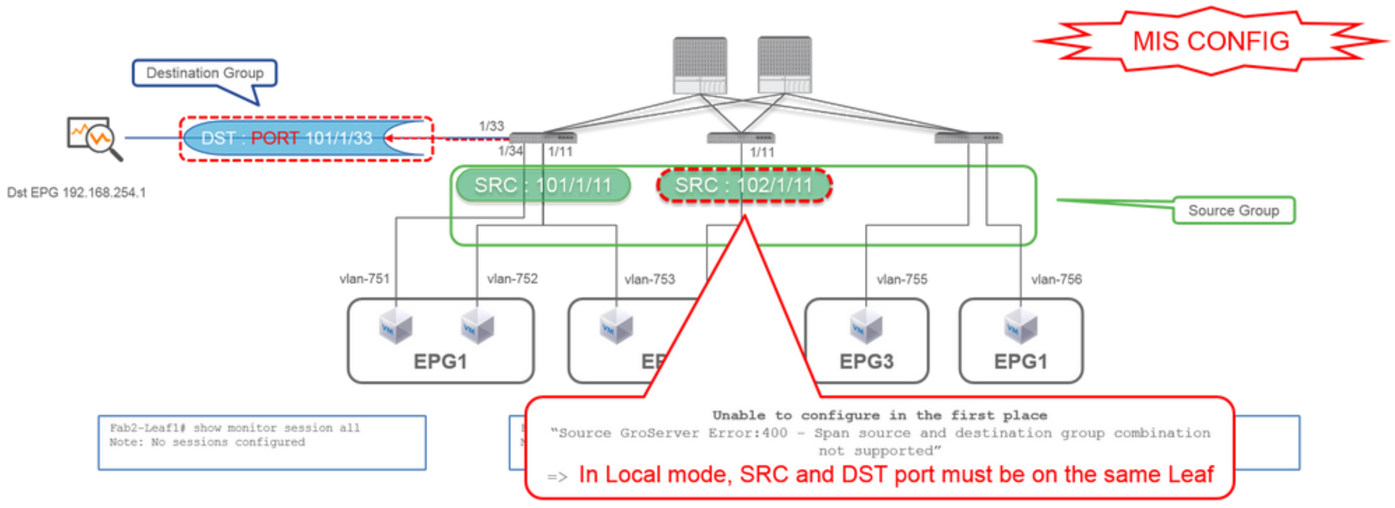
```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf1# show vlan id 35,39 extended

 VLAN Name                             Status    Ports
 ---- -------------------------------- --------- ----------
 35   TK:SPAN_APP:EPG1                 active    Eth1/34
 39   TK:SPAN_APP:EPG1                 active    Eth1/11

 VLAN Type  Vlan-mode  Encap
 ---- ----- ---------- -------------------------------
 35   enet  CE         vlan-751
 39   enet  CE         vlan-752
```

- **Source Group** Leaf1 e1/11Leaf1 e1/34EPG1 Filter
- **Destination Group** Leaf1 e1/33

Access SPAN with Local SPAN can also use EPG Filter as well as ERSPAN.

Case 3. Src "Leaf1 e1/11 & Leaf2 e/11" | Dst "Leaf1 e1/33" (bad case)



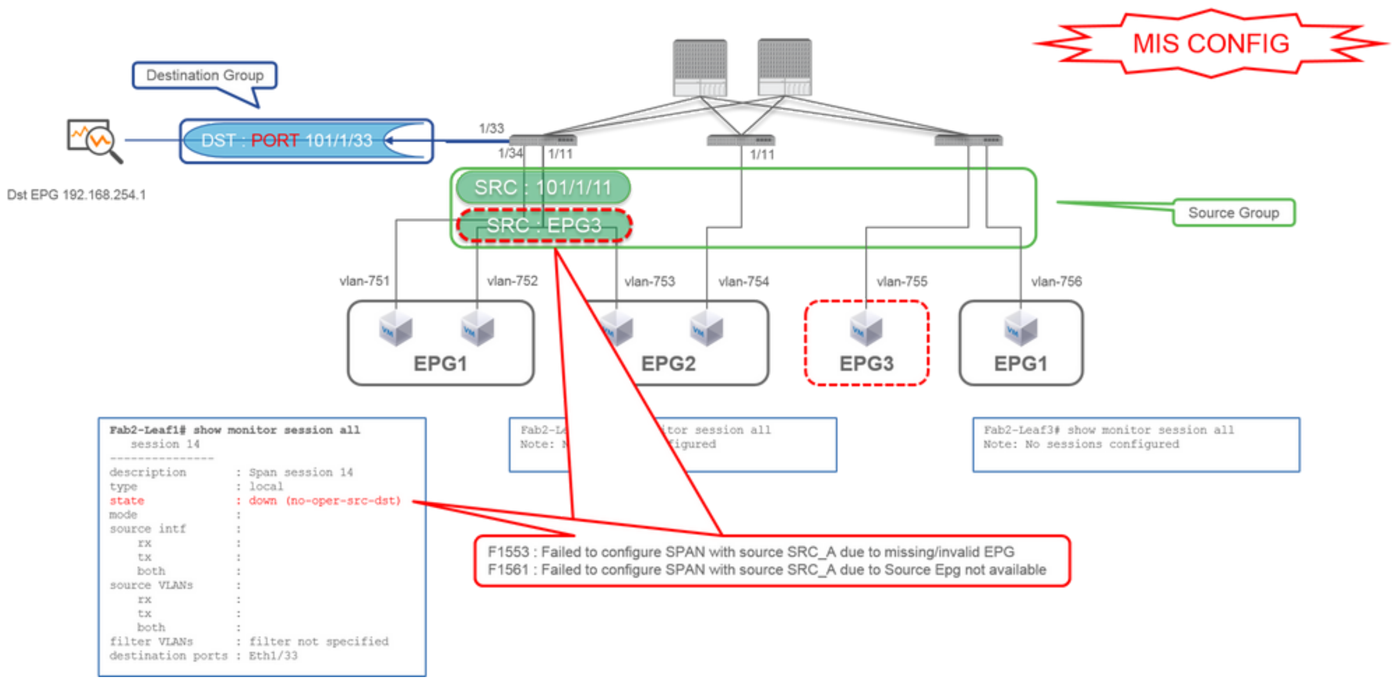
```
Fab2-Leaf1# show monitor session all
Note: No sessions configured
```

```
Unable to configure in the first place
"Source GroServer Error:400 - Span source and destination group combination
not supported"
```

=> In Local mode, SRC and DST port must be on the same Leaf

- **Source Group** Leaf1 e1/11Leaf2 e1/11
- **Destination Group** Leaf1 e1/33

Case 4. Src "Leaf1 e1/11 & EPG3 filter" | Dst "Leaf1 e1/33" (bad case)



```
Fab2-Leaf1# show monitor session all
   session 14
---------------
description       : Span session 14
type              : local
state             : down (no-oper-src-dst)
mode              :
source intf       :
    rx            :
    tx            :
    both          :
source VLANs      :
    rx            :
    tx            :
    both          :
filter VLANs      : filter not specified
destination ports : Eth1/33
```

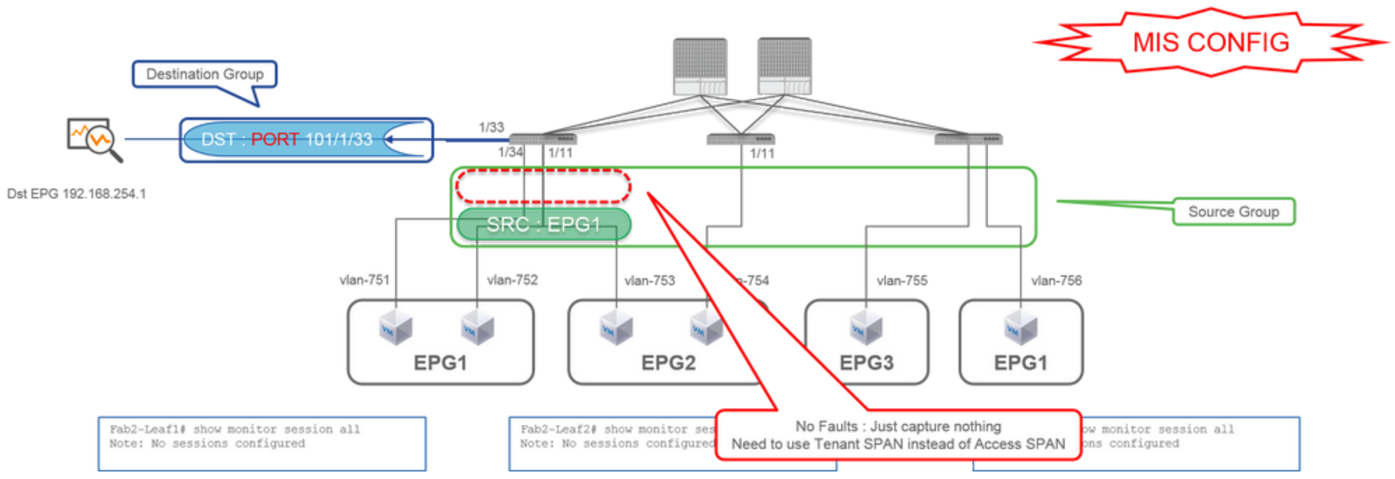
```
Fab2-L          itor session all
Note: N          figured
```

```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

- **Source Group** Leaf1 e1/11EPG3 Filter
- **Destination Group** Leaf1 e1/33

It is similar to case 3 on Access SPAN (ERSPAN) but in this example, the one and only SPAN session on Leaf1 fails because EPG3 does not exist on Leaf1. So SPAN does not work at all.

Case 5 : Src "EPG1 filter" | Dst "Leaf1 e1/33" (bad case)



```
Fab2-Leaf1# show monitor session all
Note: No sessions configured
```

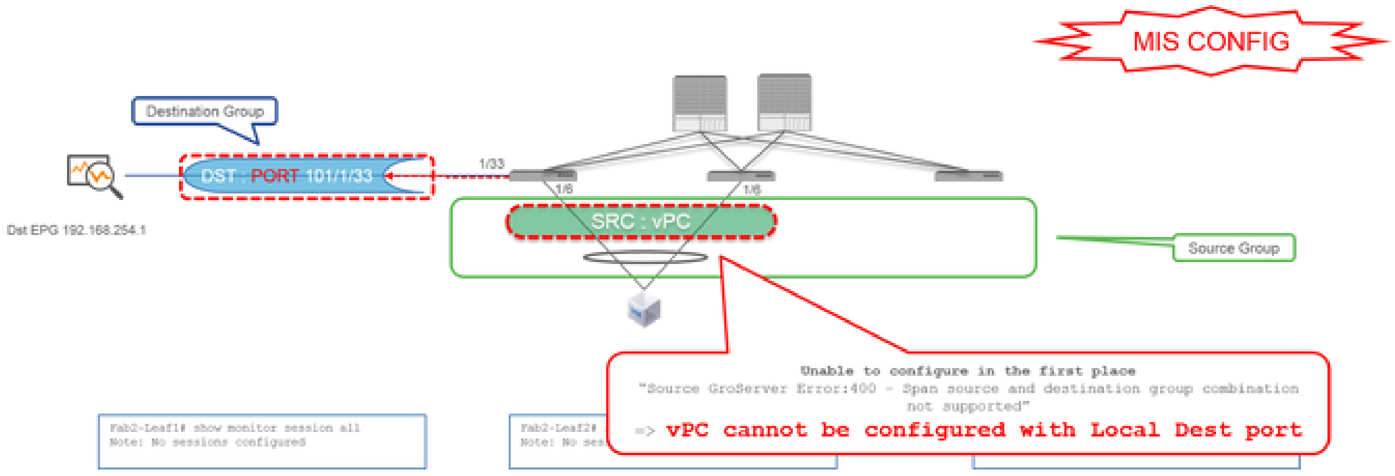
```
Fab2-Leaf2# show monitor ses
Note: No sessions configured
```

```
w monitor session all
ons configured
```

- **Source Group**
  EPG1 Filter
- **Destination Group** Leaf1 e1/33

EPG filter on Access SPAN works only when source ports are configured. If EPG is the only source to be specified, Tenant SPAN must be used instead of Access SPAN.

Case 6. Src "Leaf1 - Leaf2 vPC" | Dst "Leaf1 e1/33" (bad case)



```
Fab2-Leaf1# show monitor session all
Note: No sessions configured
```

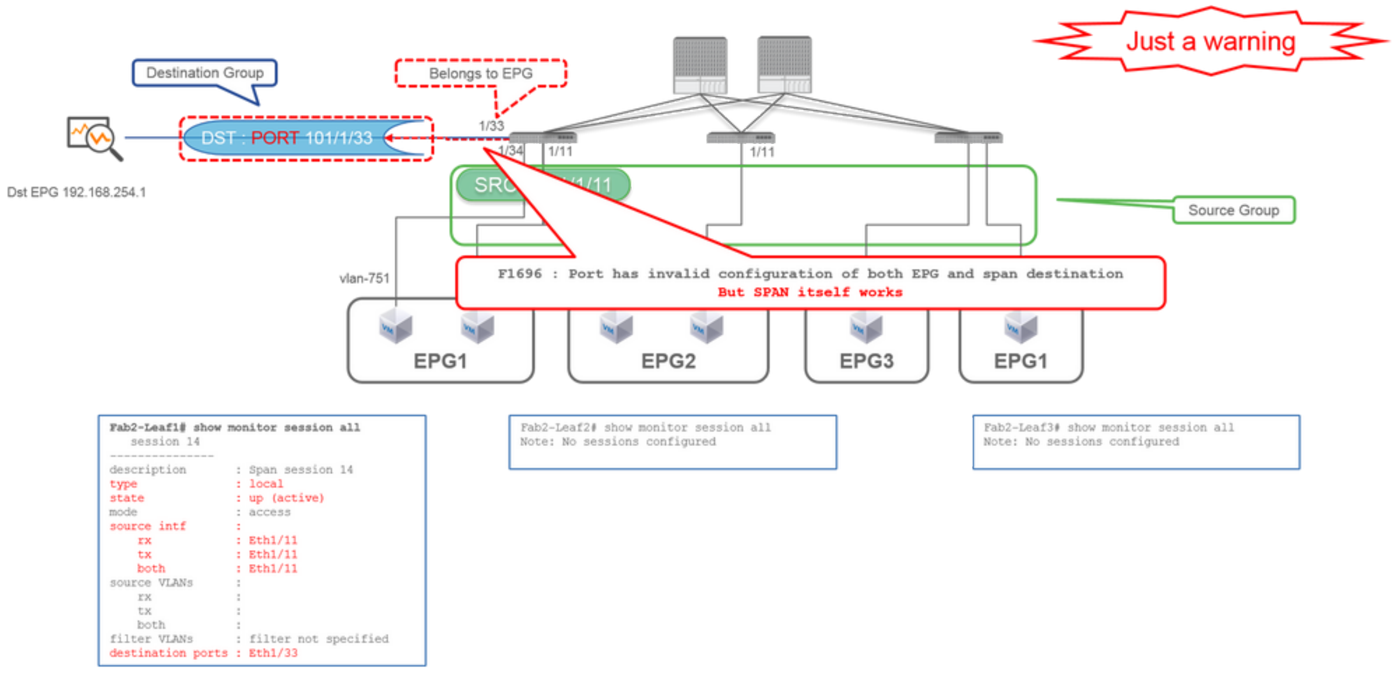
```
Fab2-Leaf2# 
Note: No ses
```

```
Unable to configure in the first place
"Source GroServer Error:400 - Span source and destination group combination
not supported"
=> vPC cannot be configured with Local Dest port
```

- **Source Group**
  Leaf1-2 vPC
- **Destination Group** Leaf1 e1/33

A vPC interface cannot be configured as a source with Local SPAN. Please use ERSPAN. Please refer to case4 for Access SPAN (ERSPAN).

Case 7. Src "Leaf1 e1/11 | Dst "Leaf1 e1/33 & e1/33 belongs to EPG" (works with fault)



```
F1696 : Port has invalid configuration of both EPG and span destination
But SPAN itself works
```

```
Fab2-Leaf1# show monitor session all
   session 14
---------------
description          : Span session 14
type                 : local
state                : up (active)
mode                 : access
source intf          :
    rx               : Eth1/11
    tx               : Eth1/11
    both             : Eth1/11
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : filter not specified
destination ports : Eth1/33
```

```
Fab2-Leaf2# show monitor session all
Note: No sessions configured
```

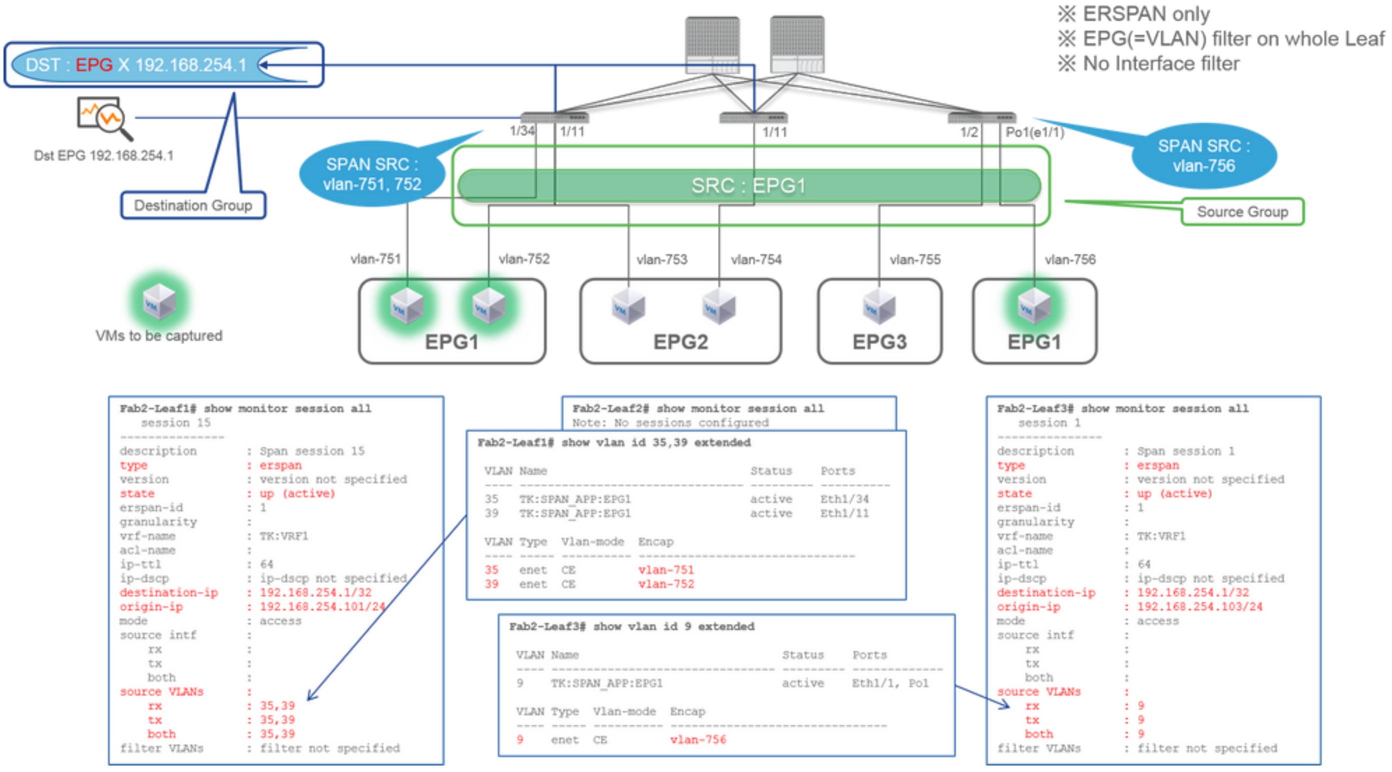
```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

If a destination I/F for SPAN already belongs to EPG, a fault "F1696 : Port has an invalid configuration of both EPG and span destination" is raised under the physical I/F.

But even with this fault, SPAN works without any problems. This fault is just a warning about extra traffic caused by SPAN since it can impact customers' normal EPG traffic on the same I/F.

Tenant SPAN (ERSPAN)

Case 1. Src "EPG1" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 15
---------------
description         : Span session 15
type                : erspan
version             : version not specified
state               : up (active)
erspan-id           : 1
granularity         :
vrf-name            : TK:VRF1
acl-name            :
ip-ttl              : 64
ip-dscp             : ip-dscp not specified
destination-ip      : 192.168.254.1/32
origin-ip           : 192.168.254.101/24
mode                : access
source intf         :
    rx              :
    tx              :
    both            :
source VLANs        :
    rx              : 35,39
    tx              : 35,39
    both            : 35,39
filter VLANs        : filter not specified
```

```
Fab2-Leaf2# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf1# show vlan id 35,39 extended

 VLAN Name                             Status    Ports
 ---- -------------------------------- --------- ----------
 35   TK:SPAN_APP:EPG1                 active    Eth1/34
 39   TK:SPAN_APP:EPG1                 active    Eth1/11

 VLAN Type  Vlan-mode  Encap
 ---- ----- ---------- -------------------------------
 35   enet  CE         vlan-751
 39   enet  CE         vlan-752
```

```
Fab2-Leaf3# show vlan id 9 extended

 VLAN Name                             Status    Ports
 ---- -------------------------------- --------- -------------
 9    TK:SPAN_APP:EPG1                 active    Eth1/1, Po1

 VLAN Type  Vlan-mode  Encap
 ---- ----- ---------- -------------------------------
 9    enet  CE         vlan-756
```

```
Fab2-Leaf3# show monitor session all
   session 1
---------------
description         : Span session 1
type                : erspan
version             : version not specified
state               : up (active)
erspan-id           : 1
granularity         :
vrf-name            : TK:VRF1
acl-name            :
ip-ttl              : 64
ip-dscp             : ip-dscp not specified
destination-ip      : 192.168.254.1/32
origin-ip           : 192.168.254.103/24
mode                : access
source intf         :
    rx              :
    tx              :
    both            :
source VLANs        :
    rx              : 9
    tx              : 9
    both            : 9
filter VLANs        : filter not specified
```

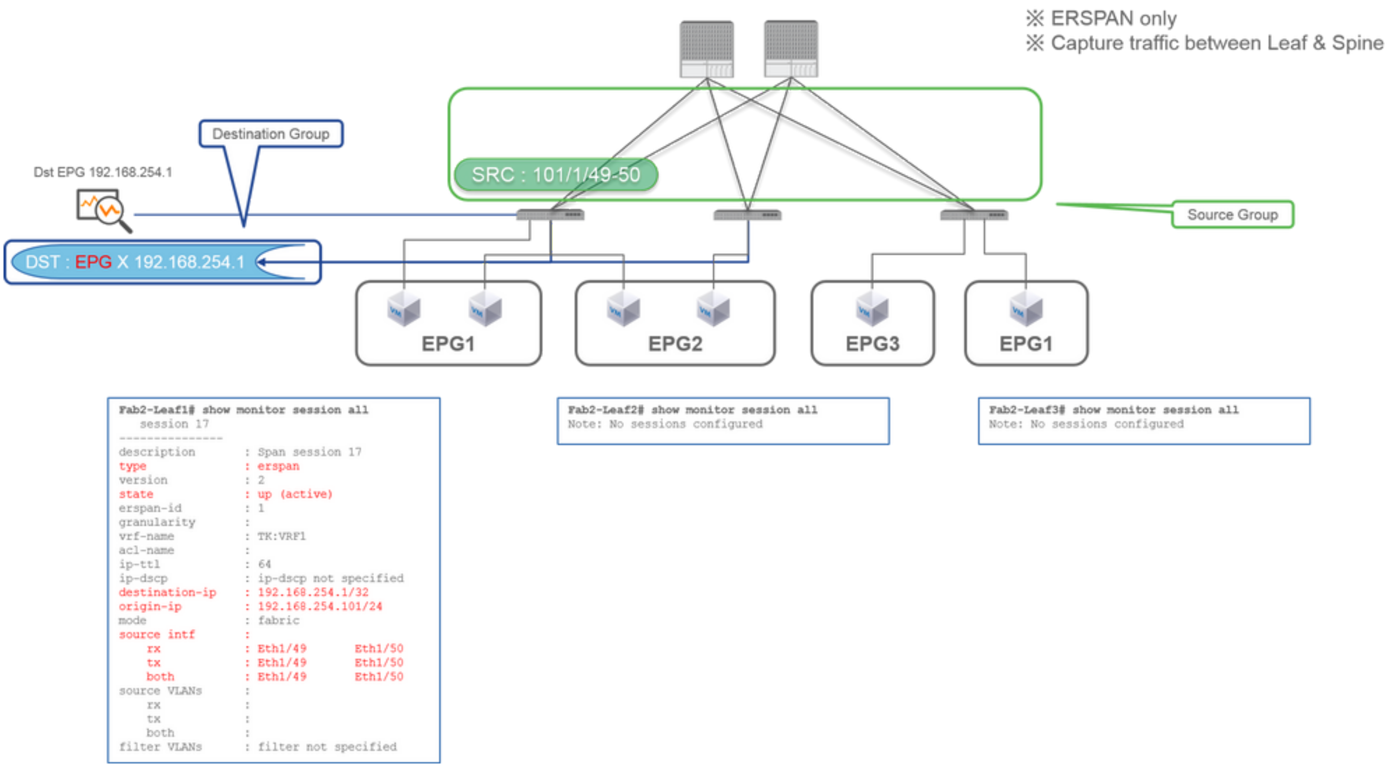
- **Source Group**
  EPG1 (no filter)
- **Destination Group** 192.168.254.1 on EPG X

Tenant SPAN uses EPG itself as a source while Access SPAN use EPG just for a filter.

The key point of Tenant SPAN is that you do not have to specify each individual port and ACI automatically detects appropriate VLANs on each Leaf switch. So this would be useful when all packets for specific EPG must be monitored and EndPoints for that EPG belong to multiple interfaces across Leaf switches.

Fabric SPAN (ERSPAN)

Case 1. Src "Leaf1 e1/49-50" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 17
---------------
description          : Span session 17
type                 : erspan
version              : 2
state                : up (active)
erspan-id            : 1
granularity          :
vrf-name             : TK:VRF1
acl-name             :
ip-ttl               : 64
ip-dscp              : ip-dscp not specified
destination-ip       : 192.168.254.1/32
origin-ip            : 192.168.254.101/24
mode                 : fabric
source intf          :
    rx               : Eth1/49       Eth1/50
    tx               : Eth1/49       Eth1/50
    both             : Eth1/49       Eth1/50
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : filter not specified
```

```
Fab2-Leaf2# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

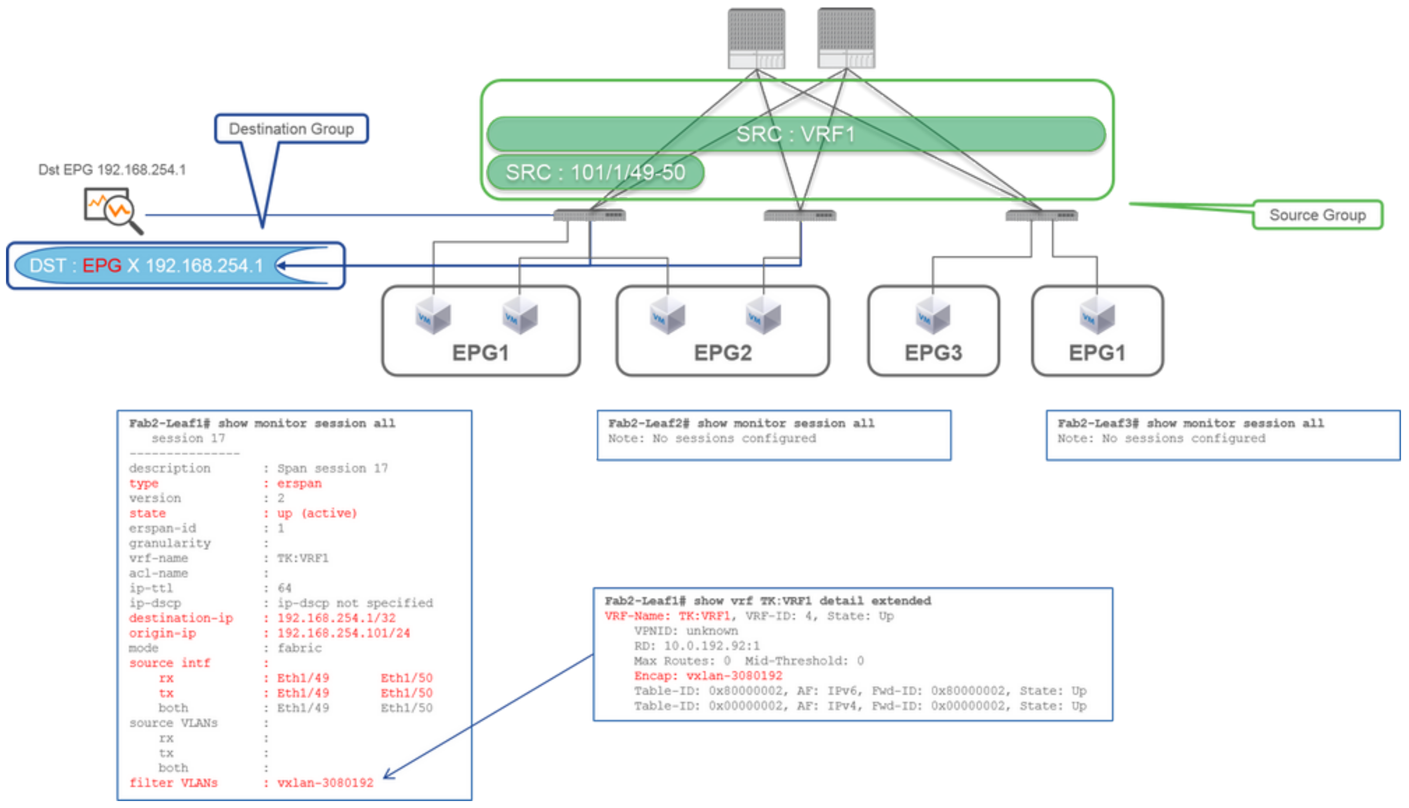
- **Source Group**
  Leaf1 e1/49-50
- **Destination Group** 192.168.254.1 on EPG X

Fabric SPAN specifies Fabric ports as a source where Fabric ports are interfaces between Leaf and Spine switches.

This SPAN is useful when it is required to copy packets between Leaf and Spine switches. However, packets between Leaf and Spine switches are encapsulated with iVxLAN header. So it is required a bit of a trick to read it. Please refer to "How to Read SPAN Data".

Note: iVxLAN header is an enhanced VxLAN header only for ACI Fabric internal use.

Case 2. Src "Leaf1 e1/49-50 & VRF filter" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 17
---------------
description          : Span session 17
type                 : erspan
version              : 2
state                : up (active)
erspan-id            : 1
granularity          :
vrf-name             : TK:VRF1
acl-name             :
ip-ttl               : 64
ip-dscp              : ip-dscp not specified
destination-ip       : 192.168.254.1/32
origin-ip            : 192.168.254.101/24
mode                 : fabric
source intf          :
    rx               : Eth1/49       Eth1/50
    tx               : Eth1/49       Eth1/50
    both             : Eth1/49       Eth1/50
source VLANs         :
    rx               :
    tx               :
    both             :
filter VLANs         : vxlan-3080192
```

```
Fab2-Leaf2# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf1# show vrf TK:VRF1 detail extended
VRF-Name: TK:VRF1, VRF-ID: 4, State: Up
    VPNID: unknown
    RD: 10.0.192.92:1
    Max Routes: 0  Mid-Threshold: 0
    Encap: vxlan-3080192
    Table-ID: 0x80000002, AF: IPv6, Fwd-ID: 0x80000002, State: Up
    Table-ID: 0x00000002, AF: IPv4, Fwd-ID: 0x00000002, State: Up
```

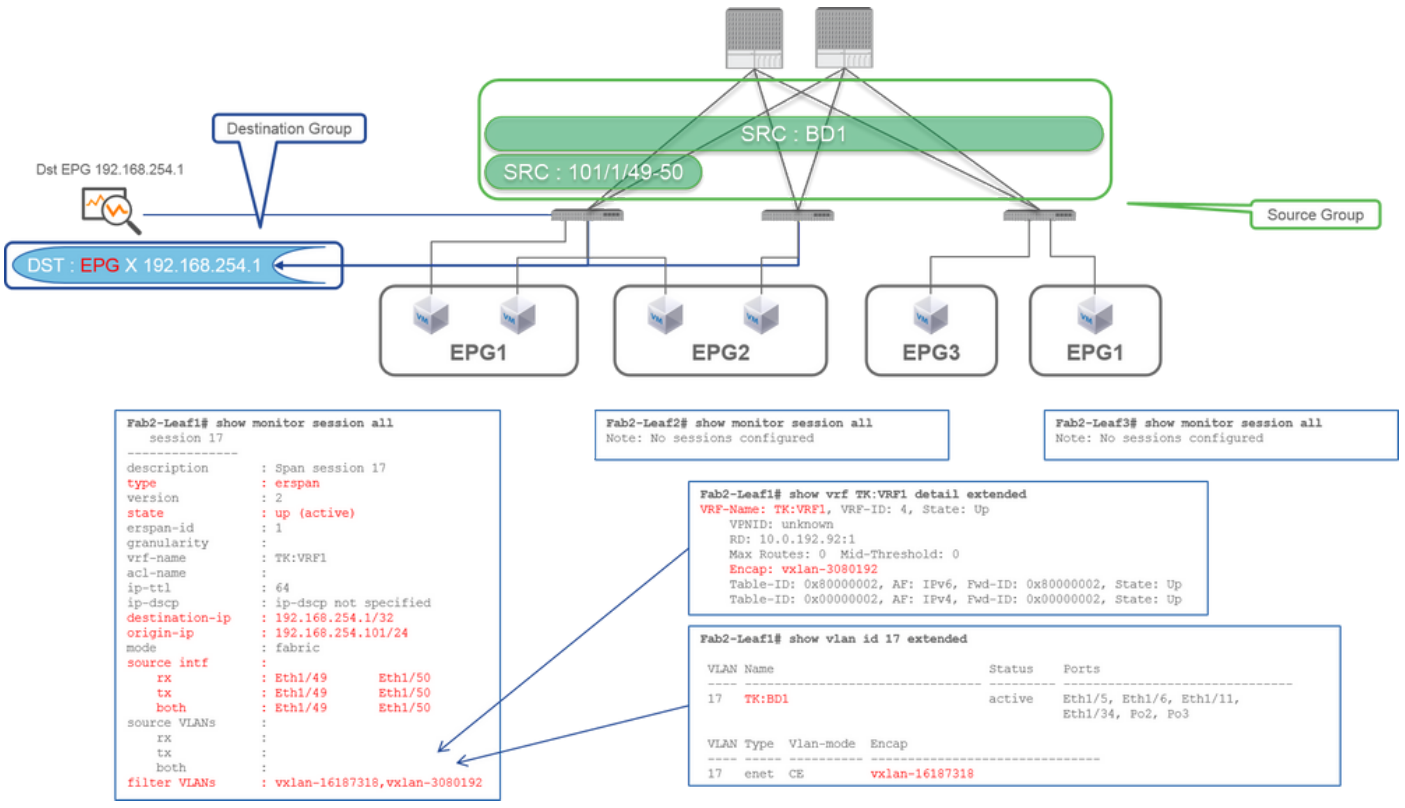
- **Source Group**
  Leaf1 e1/49-50VRF Filter
- **Destination Group** 192.168.254.1 on EPG X

Fabric SPAN can use filters as well as Access SPAN. But the filter type is different. Fabric SPAN uses Virtual Routing and Forwarding (VRF) or BD as a filter.

In Cisco ACI, as described before, packets that go through Fabric ports are encapsulated with iVxLAN header. This iVxLAN header has VRF or BD information as Virtual Network Identifier (VNID). When packets are forwarded as Layer2 (L2), iVxLAN VNID stands for BD. When packets are forwarded as Layer3 (L3), iVxLAN VNID stands for VRF.

So when it is required to capture routed traffic on Fabric ports, use VRF as a filter.

Case 3. Src "Leaf1 e1/49-50 & BD filter" | Dst "192.168.254.1"



```
Fab2-Leaf1# show monitor session all
   session 17
---------------
description         : Span session 17
type                : erspan
version             : 2
state               : up (active)
erspan-id           : 1
granularity         :
vrf-name            : TK:VRF1
acl-name            :
ip-ttl              : 64
ip-dscp             : ip-dscp not specified
destination-ip      : 192.168.254.1/32
origin-ip           : 192.168.254.101/24
mode                : fabric
source intf         :
    rx              : Eth1/49       Eth1/50
    tx              : Eth1/49       Eth1/50
    both            : Eth1/49       Eth1/50
source VLANs        :
    rx              :
    tx              :
    both            :
filter VLANs        : vxlan-16187318,vxlan-3080192
```

```
Fab2-Leaf2# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf3# show monitor session all
Note: No sessions configured
```

```
Fab2-Leaf1# show vrf TK:VRF1 detail extended
VRF-Name: TK:VRF1, VRF-ID: 4, State: Up
    VPNID: unknown
    RD: 10.0.192.92:1
    Max Routes: 0  Mid-Threshold: 0
    Encap: vxlan-3080192
    Table-ID: 0x80000002, AF: IPv6, Fwd-ID: 0x80000002, State: Up
    Table-ID: 0x00000002, AF: IPv4, Fwd-ID: 0x00000002, State: Up
```

```
Fab2-Leaf1# show vlan id 17 extended

 VLAN Name                             Status    Ports
 ---- -------------------------------- --------- -------------------------------
 17   TK:BD1                           active    Eth1/5, Eth1/6, Eth1/11,
                                                 Eth1/34, Po2, Po3

 VLAN Type  Vlan-mode  Encap
 ---- ----- ---------- -------------------------------
 17   enet  CE         vxlan-16187318
```

- **Source Group**
  Leaf1 e1/49-50BD Filter
- **Destination Group** 192.168.254.1 on EPG X

As described in previous case 2, Fabric SPAN can use BD as a filter.

When it is required to capture bridged traffic on Fabric ports, use BD as a filter.

Note: Only a single filter of either BD or VRF can be configured at a time.

What do you need on the SPAN destination device?

Just run a packet capture application such as tcpdump, wireshark on it. It is not required to configure the ERSPAN destination session or anything.

For ERSPAN

Please ensure to run a capture tool on the interface with the destination IP for ERSPAN since SPAN packets are forwarded to the destination IP.

The received packet is encapsulated with a GRE header. Please refer to this section "How to Read ERSPAN Data" on how to decode the ERSPAN GRE header.

For Local SPAN

Please ensure to run a capture tool on the interface that connects to the SPAN Destination interface on ACI Leaf.

Raw packets are received in this interface. it is not required to deal with the ERSPAN header.

How to Read ERSPAN Data

ERSPAN Version (type)

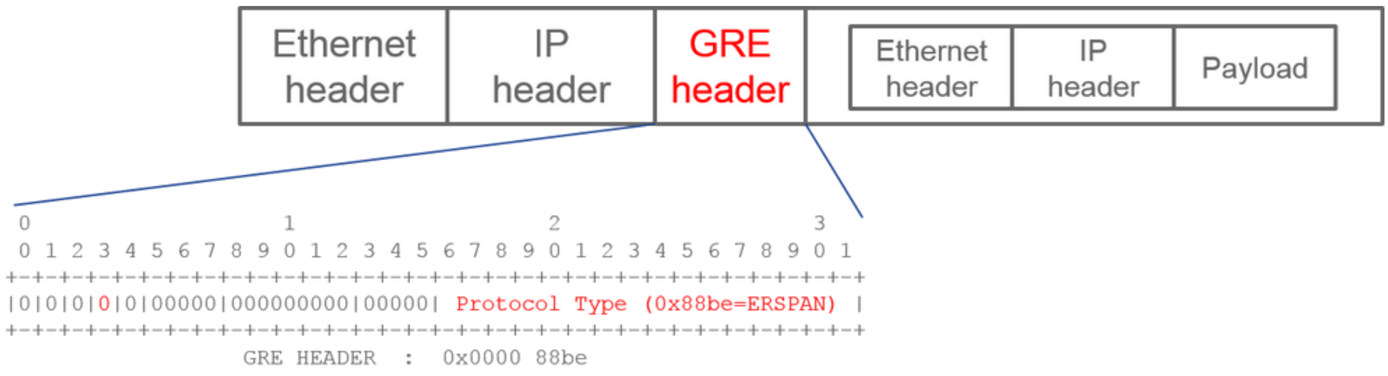
ERSPAN encapsulates copied packets to forward them to the remote destination. GRE is used for this encapsulation. The protocol type for ERSPAN on the GRE header is 0x88be.

In Internet Engineering Task Force (IETF) document, the ERSPAN version is described as type instead of version.

There are three types of ERSPAN. I, II and III. ERSPAN Type is mentioned in this RFC draft. Also, this GRE RFC1701 can be helpful to understand each ERSPAN type as well.

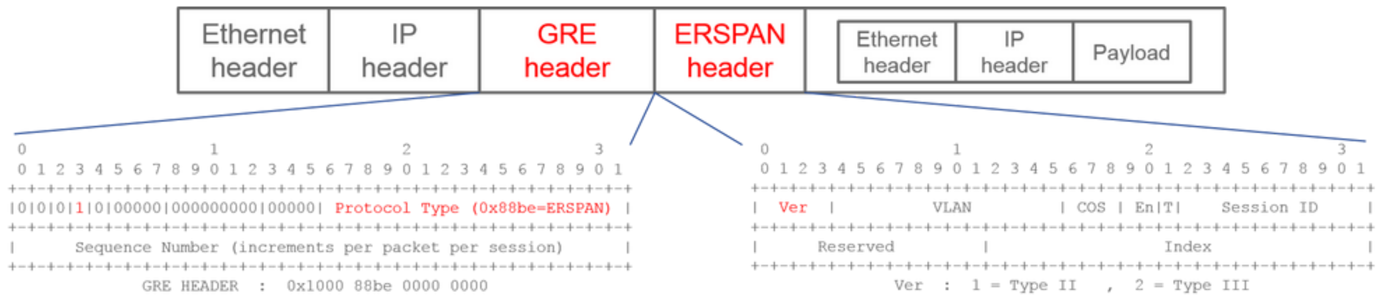
Here is the packet format of each type:

ERSPAN Type I (used by Broadcom Trident 2)



Type I does not use the sequence field on the GRE header. It does not even use the ERSPAN header which must follow the GRE header if it was ERSPAN type II and III. Broadcom Trident 2 only supports this ERSPAN type I.

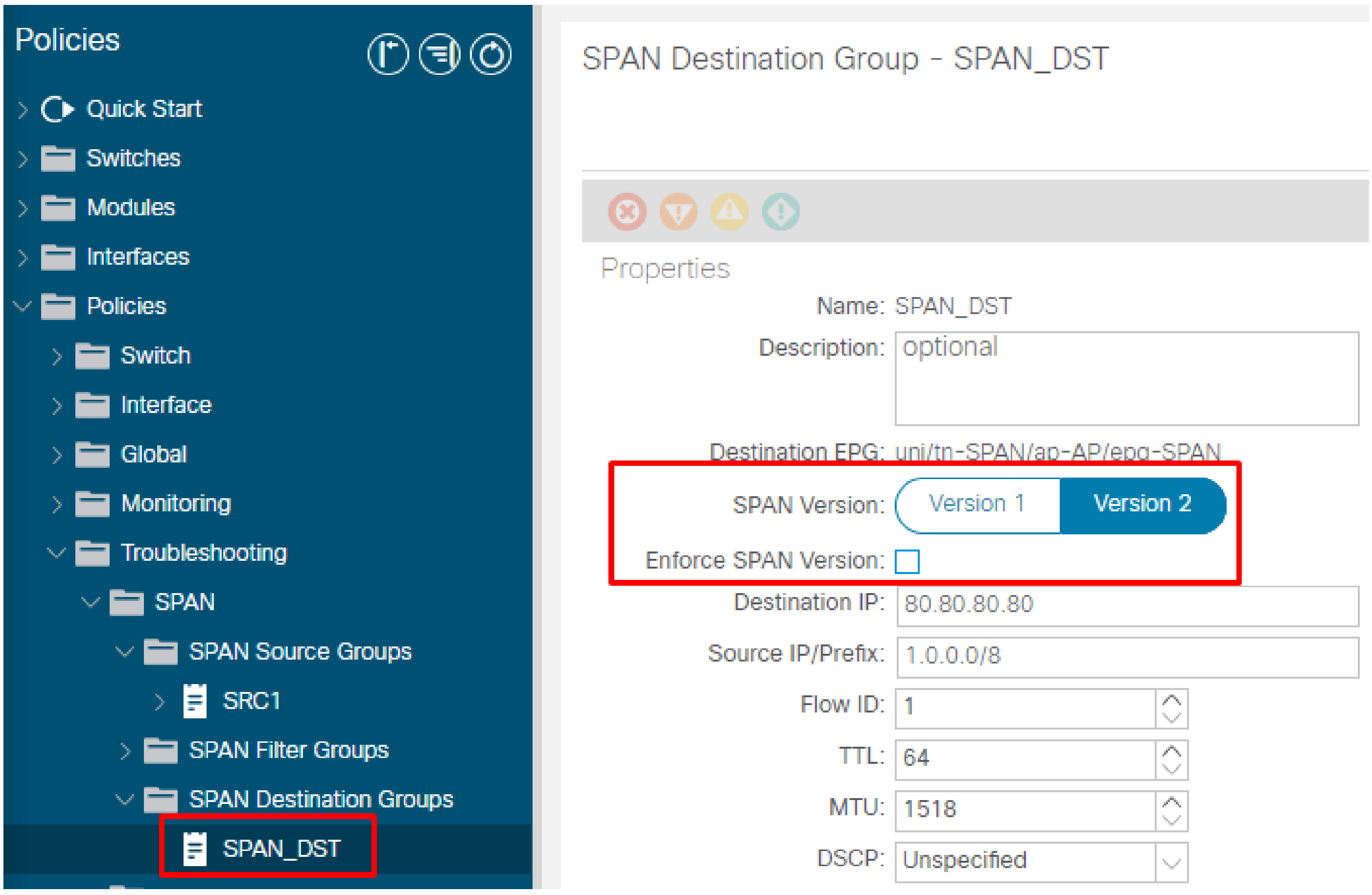
ERSPAN Type II or III



If the sequence field is activated by the S bit, this must be ERSPAN type II or III. The version field on the ERSPAN header identifies the ERSPAN type. In ACI, type III is not supported as of 03/20/2016.

If a SPAN source group for Access or Tenant SPAN has sources on both 1st-gen and 2nd-gen nodes, the ERSPAN destination receives both ERSPAN Type I and II packets from each generation of nodes. However, Wireshark can decode only one of the ERSPAN Types at a time. By default, it only decodes ERSPAN Type II. If you enable the decode of ERSPAN Type I, Wireshark does not decode ERSPAN Type II. See the later section on how to decode ERSPAN Type I on Wireshark.

To avoid this type of issue, you can configure ERSPAN Type on a SPAN destination group.



SPAN Destination Group - SPAN_DST

Properties

Name: SPAN_DST
Description: optional
Destination EPG: uni/tn-SPAN/ap-AP/epg-SPAN
SPAN Version: Version 1 [✓ Version 2]
Enforce SPAN Version: ☐
Destination IP: 80.80.80.80
Source IP/Prefix: 1.0.0.0/8
Flow ID: 1
TTL: 64
MTU: 1518
DSCP: Unspecified

- SPAN Version (Version 1 or Version 2): This refers to the ERSPAN Type I or II
- Enforce SPAN Version (checked or unchecked): This decides if the SPAN session must fail in case the configured ERSPAN Type is not supported on the source node hardware.

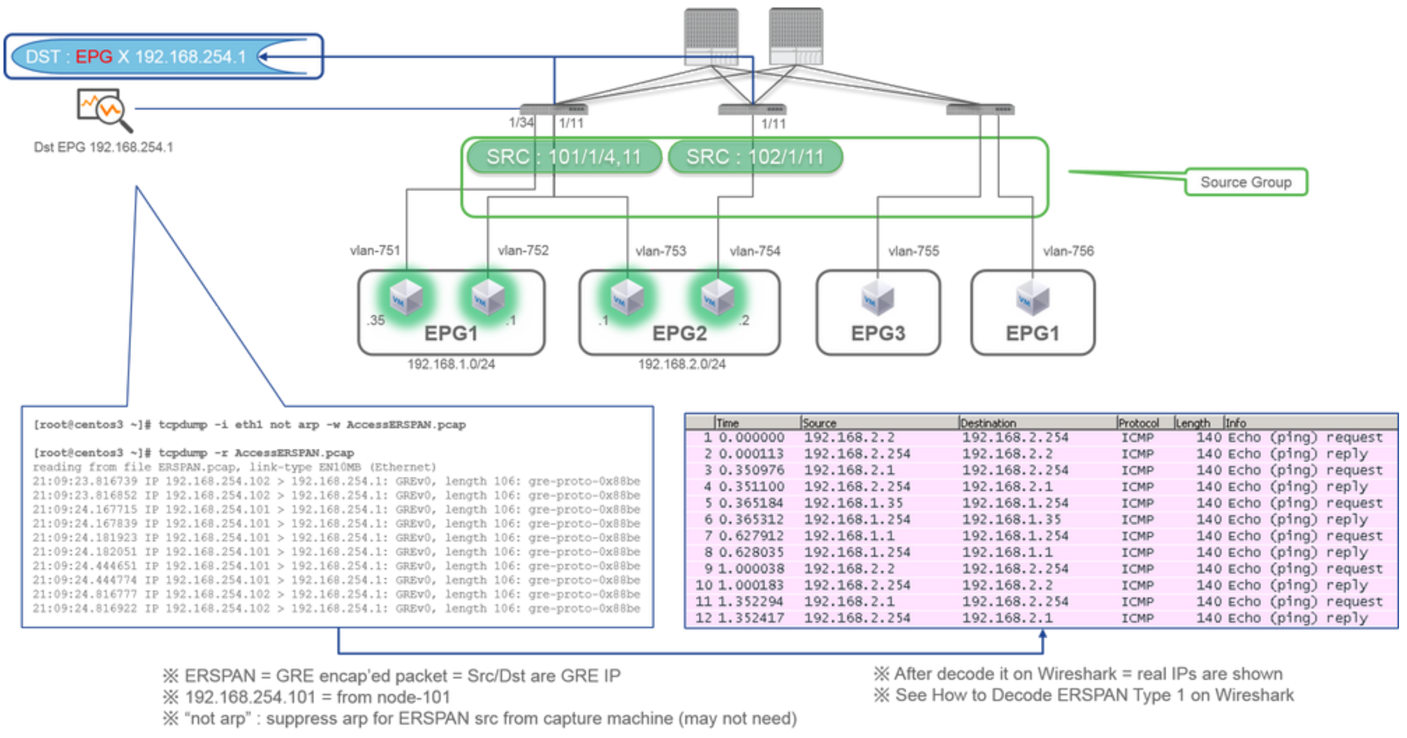
By default, SPAN Version is Version 2 and Enforce SPAN Version is unchecked. This means that if the source node is 2nd gen or later which supports ERSPAN Type II, it generates ERSPAN with Type II. If the source node is 1st gen which does not support ERSPAN Type II (except for Fabric SPAN), it falls back to Type I since the Enforce SPAN Version is not checked. As a result, the ERSPAN destination receives a mixed type of ERSPAN.

This table explains each combination for Access and Tenant SPAN.

SPAN Version	**Enforce SPAN Version**	**1st gen source node**	**2nd gen source node**
Version 2	Unchecked	Uses Type I	Uses Type II
Version 2	Checked	Fails	Uses Type II
Version 1	Unchecked	Uses Type I	Uses Type I
Version 1	Checked	Uses Type I	Uses Type I

ERSPAN Data Example

Tenant SPAN/Access SPAN (ERSPAN)



```
[root@centos3 ~]# tcpdump -i eth1 not arp -w AccessERSPAN.pcap

[root@centos3 ~]# tcpdump -r AccessERSPAN.pcap
reading from file ERSPAN.pcap, link-type EN10MB (Ethernet)
21:09:23.816739 IP 192.168.254.102 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:23.816852 IP 192.168.254.102 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.167715 IP 192.168.254.101 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.167839 IP 192.168.254.101 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.181923 IP 192.168.254.101 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.182051 IP 192.168.254.101 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.444651 IP 192.168.254.101 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.444774 IP 192.168.254.101 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.816777 IP 192.168.254.102 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
21:09:24.816922 IP 192.168.254.102 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
```

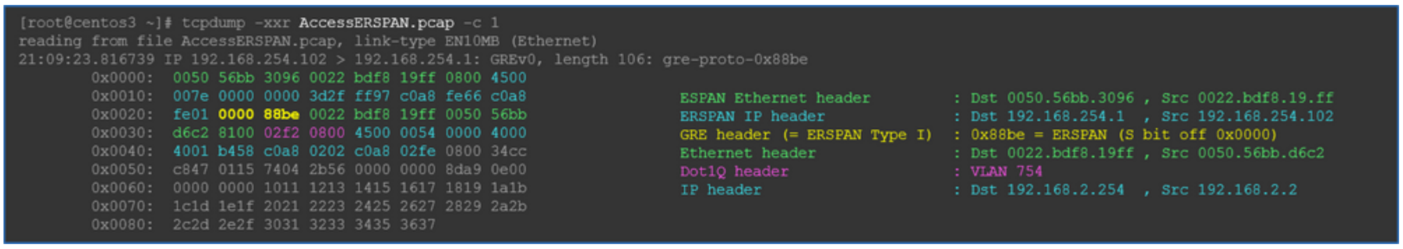
	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.2.2	192.168.2.254	ICMP	140	Echo (ping) request
2	0.000113	192.168.2.254	192.168.2.2	ICMP	140	Echo (ping) reply
3	0.350976	192.168.2.1	192.168.2.254	ICMP	140	Echo (ping) request
4	0.351100	192.168.2.254	192.168.2.1	ICMP	140	Echo (ping) reply
5	0.365184	192.168.1.35	192.168.1.254	ICMP	140	Echo (ping) request
6	0.365312	192.168.1.254	192.168.1.35	ICMP	140	Echo (ping) reply
7	0.627912	192.168.1.1	192.168.1.254	ICMP	140	Echo (ping) request
8	0.628035	192.168.1.254	192.168.1.1	ICMP	140	Echo (ping) reply
9	1.000038	192.168.2.2	192.168.2.254	ICMP	140	Echo (ping) request
10	1.000183	192.168.2.254	192.168.2.2	ICMP	140	Echo (ping) reply
11	1.352294	192.168.2.1	192.168.2.254	ICMP	140	Echo (ping) request
12	1.352417	192.168.2.254	192.168.2.1	ICMP	140	Echo (ping) reply

※ ERSPAN = GRE encap'ed packet = Src/Dst are GRE IP
※ 192.168.254.101 = from node-101
※ "not arp" : suppress arp for ERSPAN src from capture machine (may not need)

※ After decode it on Wireshark = real IPs are shown
※ See How to Decode ERSPAN Type 1 on Wireshark

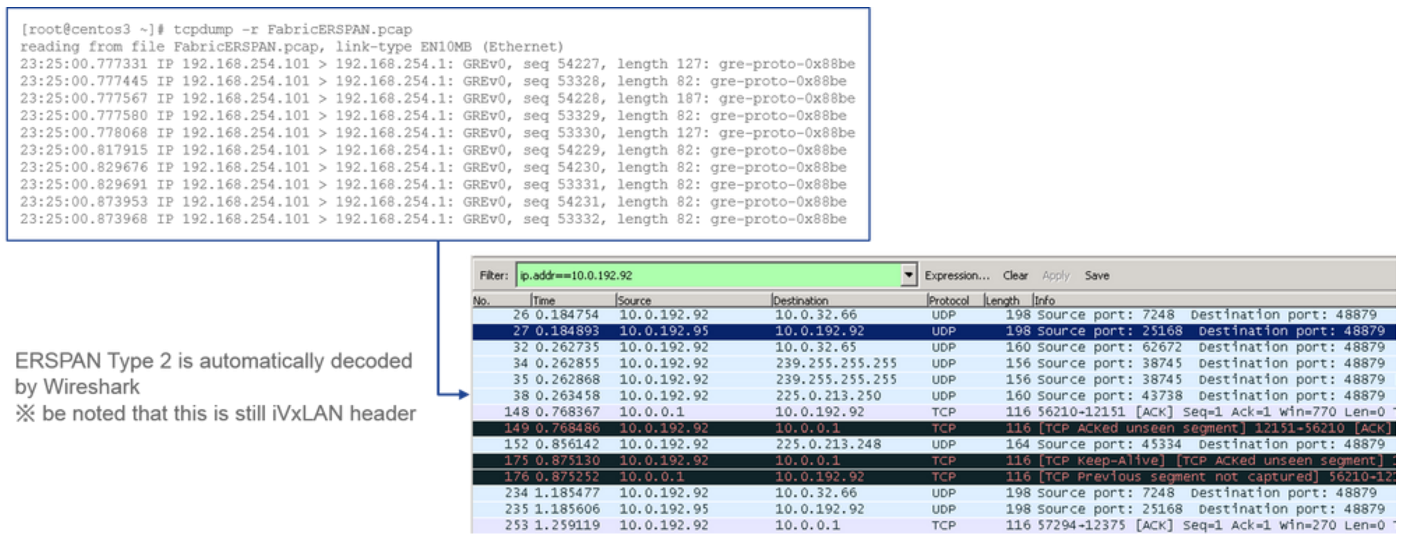
Packets need to be decoded since it is encapsulated by ERSPAN Type I. This can be done with Wireshark. Please refer to the section "How to Decode ERSPAN Type 1".

Details of Captured Packet (ERSPAN Type I)



```
[root@centos3 ~]# tcpdump -xxr AccessERSPAN.pcap -c 1
reading from file AccessERSPAN.pcap, link-type EN10MB (Ethernet)
21:09:23.816739 IP 192.168.254.102 > 192.168.254.1: GREv0, length 106: gre-proto-0x88be
        0x0000:  0050 56bb 3096 0022 bdf8 19ff 0800 4500
        0x0010:  007e 0000 0000 3d2f ff97 c0a8 fe66 c0a8     ESPAN Ethernet header          : Dst 0050.56bb.3096 , Src 0022.bdf8.19.ff
        0x0020:  fe01 0000 88be 0022 bdf8 19ff 0050 56bb     ERSPAN IP header               : Dst 192.168.254.1  , Src 192.168.254.102
        0x0030:  d6c2 8100 02f2 0800 4500 0054 0000 4000     GRE header (= ERSPAN Type I)   : 0x88be = ERSPAN (S bit off 0x0000)
        0x0040:  4001 b458 c0a8 0202 c0a8 02fe 0800 34cc     Ethernet header                : Dst 0022.bdf8.19ff , Src 0050.56bb.d6c2
        0x0050:  c847 0115 7404 2b56 0000 0000 8da9 0e00     Dot1Q header                   : VLAN 754
        0x0060:  0000 0000 1011 1213 1415 1617 1819 1a1b     IP header                      : Dst 192.168.2.254  , Src 192.168.2.2
        0x0070:  1c1d 1e1f 2021 2223 2425 2627 2829 2a2b
        0x0080:  2c2d 2e2f 3031 3233 3435 3637
```

Fabric SPAN (ERSPAN)



```
[root@centos3 ~]# tcpdump -r FabricERSPAN.pcap
reading from file FabricERSPAN.pcap, link-type EN10MB (Ethernet)
23:25:00.777331 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 54227, length 127: gre-proto-0x88be
23:25:00.777445 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 53328, length 82: gre-proto-0x88be
23:25:00.777567 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 54228, length 187: gre-proto-0x88be
23:25:00.777580 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 53329, length 82: gre-proto-0x88be
23:25:00.778068 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 53330, length 127: gre-proto-0x88be
23:25:00.817915 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 54229, length 82: gre-proto-0x88be
23:25:00.829676 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 54230, length 82: gre-proto-0x88be
23:25:00.829691 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 53331, length 82: gre-proto-0x88be
23:25:00.873953 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 54231, length 82: gre-proto-0x88be
23:25:00.873968 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 53332, length 82: gre-proto-0x88be
```

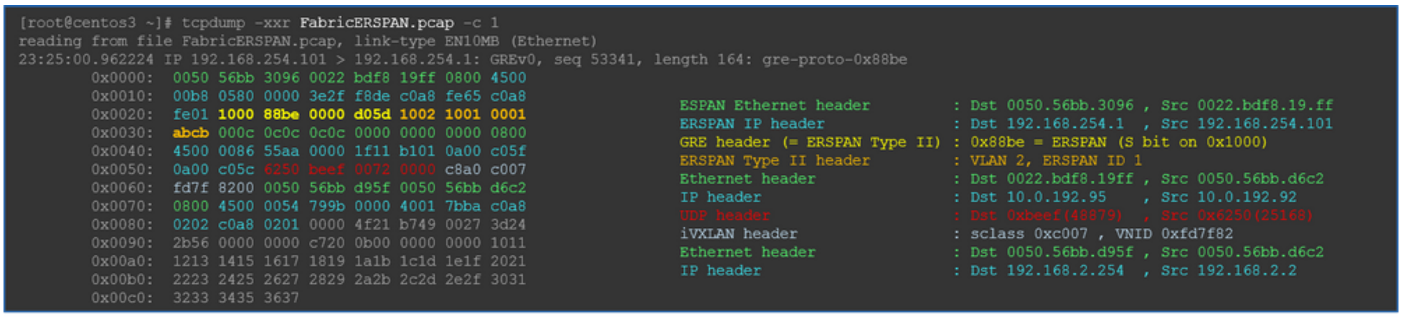
ERSPAN Type 2 is automatically decoded by Wireshark
※ be noted that this is still iVxLAN header

No.	Time	Source	Destination	Protocol	Length	Info
26	0.184754	10.0.192.92	10.0.32.66	UDP	198	Source port: 7248 Destination port: 48879
27	0.184893	10.0.192.95	10.0.192.92	UDP	198	Source port: 25168 Destination port: 48879
32	0.262735	10.0.192.92	10.0.32.65	UDP	160	Source port: 62672 Destination port: 48879
34	0.262855	10.0.192.92	239.255.255.255	UDP	156	Source port: 38745 Destination port: 48879
35	0.262868	10.0.192.92	239.255.255.255	UDP	156	Source port: 38745 Destination port: 48879
38	0.263458	10.0.192.92	225.0.213.250	UDP	160	Source port: 43738 Destination port: 48879
148	0.768367	10.0.0.1	10.0.192.92	TCP	116	56210→12151 [ACK] Seq=1 Ack=1 Win=770 Len=0
149	0.768486	10.0.192.92	10.0.0.1	TCP	116	[TCP ACKed unseen segment] 12151→56210 [ACK]
152	0.856142	10.0.192.92	225.0.213.248	UDP	164	Source port: 45334 Destination port: 48879
175	0.875130	10.0.192.92	10.0.0.1	TCP	116	[TCP Keep-Alive] [TCP ACKed unseen segment]
176	0.875252	10.0.0.1	10.0.192.92	TCP	116	[TCP Previous segment not captured] 56210→12
234	1.185477	10.0.192.92	10.0.32.66	UDP	198	Source port: 7248 Destination port: 48879
235	1.185606	10.0.192.95	10.0.192.92	UDP	198	Source port: 25168 Destination port: 48879
253	1.259119	10.0.192.92	10.0.0.1	TCP	116	57294→12375 [ACK] Seq=1 Ack=1 Win=270 Len=0

Wireshark automatically decodes ERSPAN Type II. However, it is still encapsulated by iVxLAN header.

By default, Wireshark does not understand the iVxLAN header since it is ACI internal header. Please refer to "How to Decode iVxLAN Header".

Details of the Captured Packet (ERSPAN Type II)

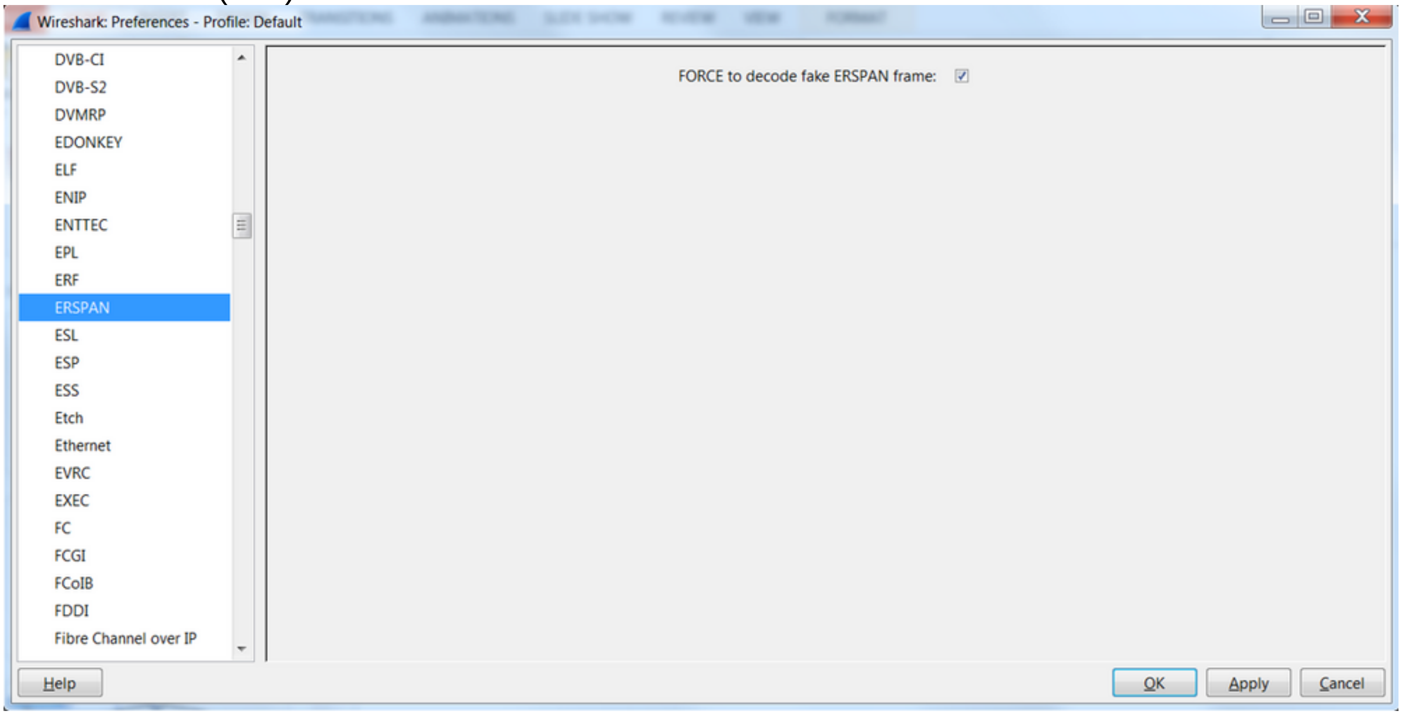


```
[root@centos3 ~]# tcpdump -xxr FabricERSPAN.pcap -c 1
reading from file FabricERSPAN.pcap, link-type EN10MB (Ethernet)
23:25:00.962224 IP 192.168.254.101 > 192.168.254.1: GREv0, seq 53341, length 164: gre-proto-0x88be
        0x0000:  0050 56bb 3096 0022 bdf8 19ff 0800 4500
        0x0010:  00b8 0580 0000 3e2f f8de c0a8 fe65 c0a8      ESPAN Ethernet header         : Dst 0050.56bb.3096 , Src 0022.bdf8.19.ff
        0x0020:  fe01 1000 88be 0000 d05d 1002 1001 0001      ERSPAN IP header              : Dst 192.168.254.1  , Src 192.168.254.101
        0x0030:  abcb 000c 0c0c 0c0c 0000 0000 0000 0800      GRE header (= ERSPAN Type II) : 0x88be = ERSPAN (S bit on 0x1000)
        0x0040:  4500 0086 55aa 0000 1f11 b101 0a00 c05f      ERSPAN Type II header         : VLAN 2, ERSPAN ID 1
        0x0050:  0a00 c05c 6250 beef 0072 0000 c8a0 c007      Ethernet header               : Dst 0022.bdf8.19ff , Src 0050.56bb.d6c2
        0x0060:  fd7f 8200 0050 56bb d95f 0050 56bb d6c2      IP header                     : Dst 10.0.192.95    , Src 10.0.192.92
        0x0070:  0800 4500 0054 799b 0000 4001 7bba c0a8      UDP header                    : Dst 0xbeef(48879)  , Src 0x6250(25168)
        0x0080:  0202 c0a8 0201 0000 4f21 b749 0027 3d24      iVXLAN header                 : sclass 0xc007 , VNID 0xfd7f82
        0x0090:  2b56 0000 0000 c720 0b00 0000 0000 1011      Ethernet header               : Dst 0050.56bb.d95f , Src 0050.56bb.d6c2
        0x00a0:  1213 1415 1617 1819 1a1b 1c1d 1e1f 2021      IP header                     : Dst 192.168.2.254  , Src 192.168.2.2
        0x00b0:  2223 2425 2627 2829 2a2b 2c2d 2e2f 3031
        0x00c0:  3233 3435 3637
```

How to Decode ERSPAN Type I

Option 1. Navigate to Edit > Preference > Protocols > ERSPAN and check FORCE to decode the fake ERSPAN frame.

- Wireshark (GUI)



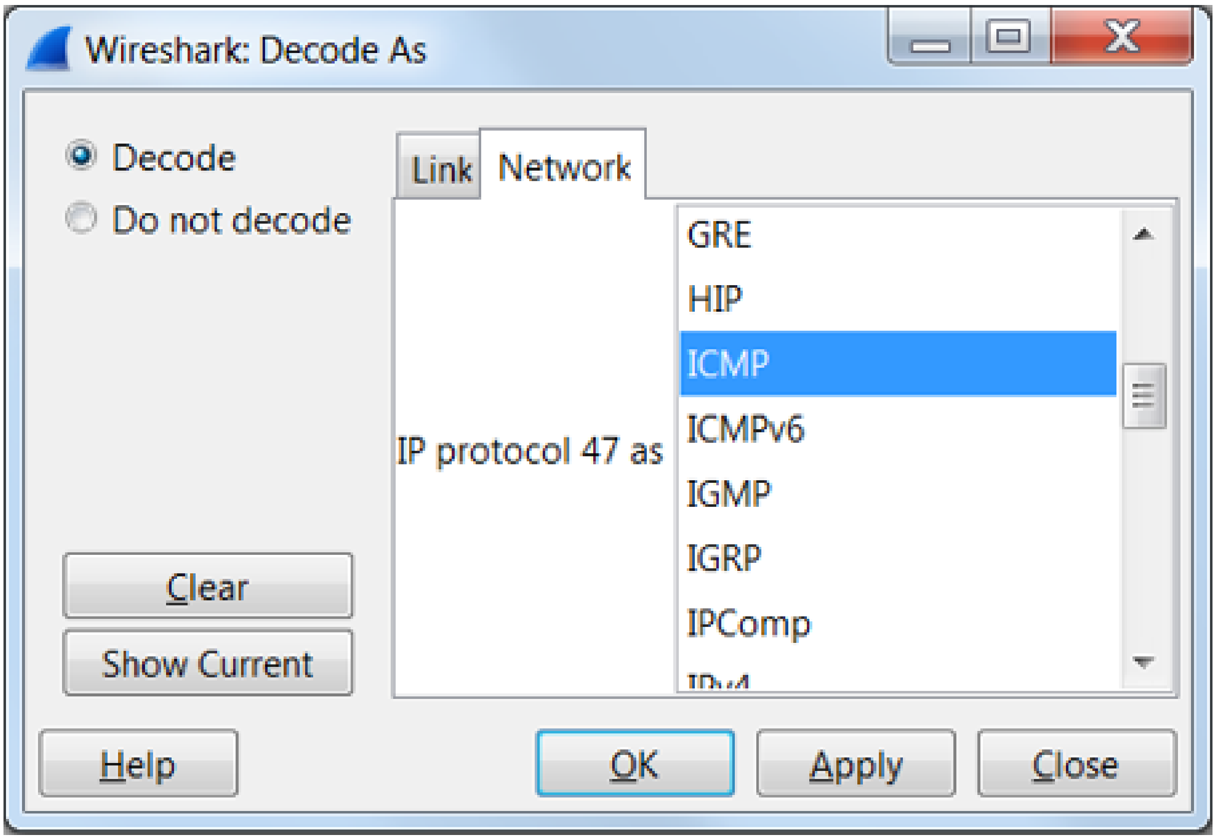
FORCE to decode fake ERSPAN frame: ☑

- Tshark (CLI version of Wireshark):

```
user1@linux# tshark -f 'proto GRE' -nV -i eth0 -o erspan.fake_erspan:true
```

Note: Please ensure to disable this option when you read ERSPAN type II or III.

Option 2. Navigate to Decode As > Network > ICMP (if it’s ICMP).

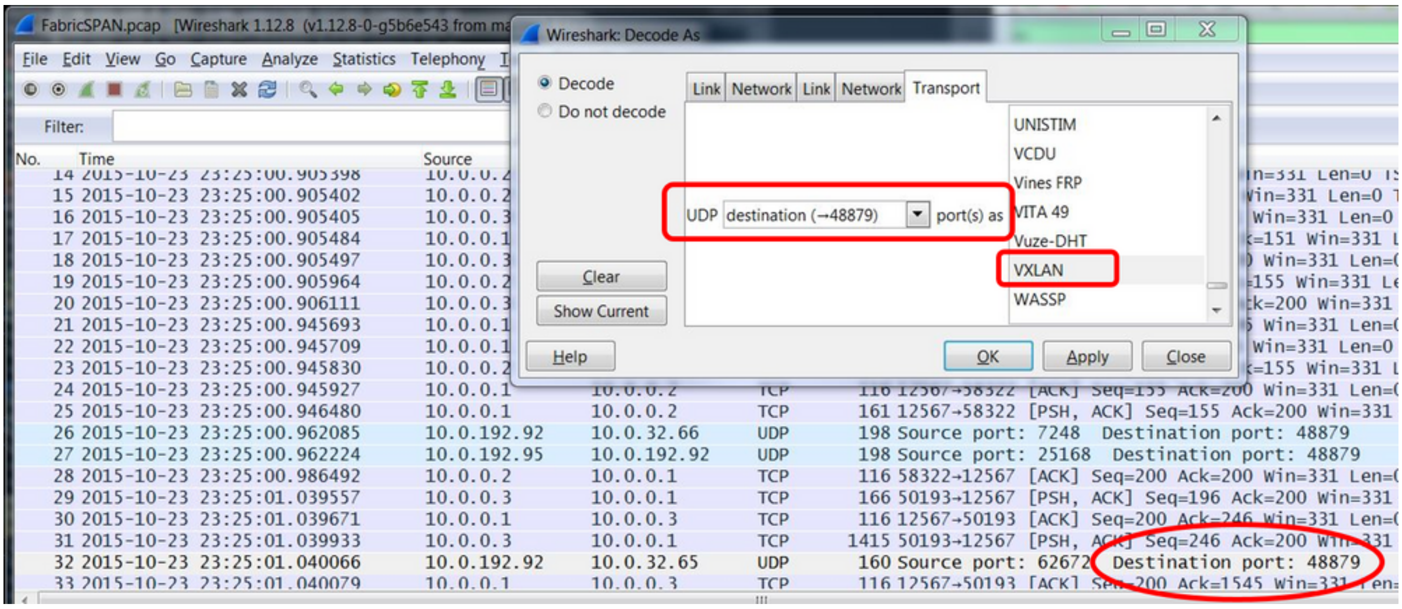


Wireshark: Decode As

Decode [✓] / Do not decode

IP protocol 47 as: GRE, HIP, ICMP [✓], ICMPv6, IGMP, IGRP, IPComp

How to Decode iVxLAN Header



FabricSPAN.pcap [Wireshark 1.12.8 (v1.12.8-0-g5b6e543 from ma

Wireshark: Decode As

Decode [✓] / Do not decode

UDP destination (→48879) port(s) as: UNISTIM, VCDU, Vines FRP, VITA 49, Vuze-DHT, VXLAN [✓], WASSP

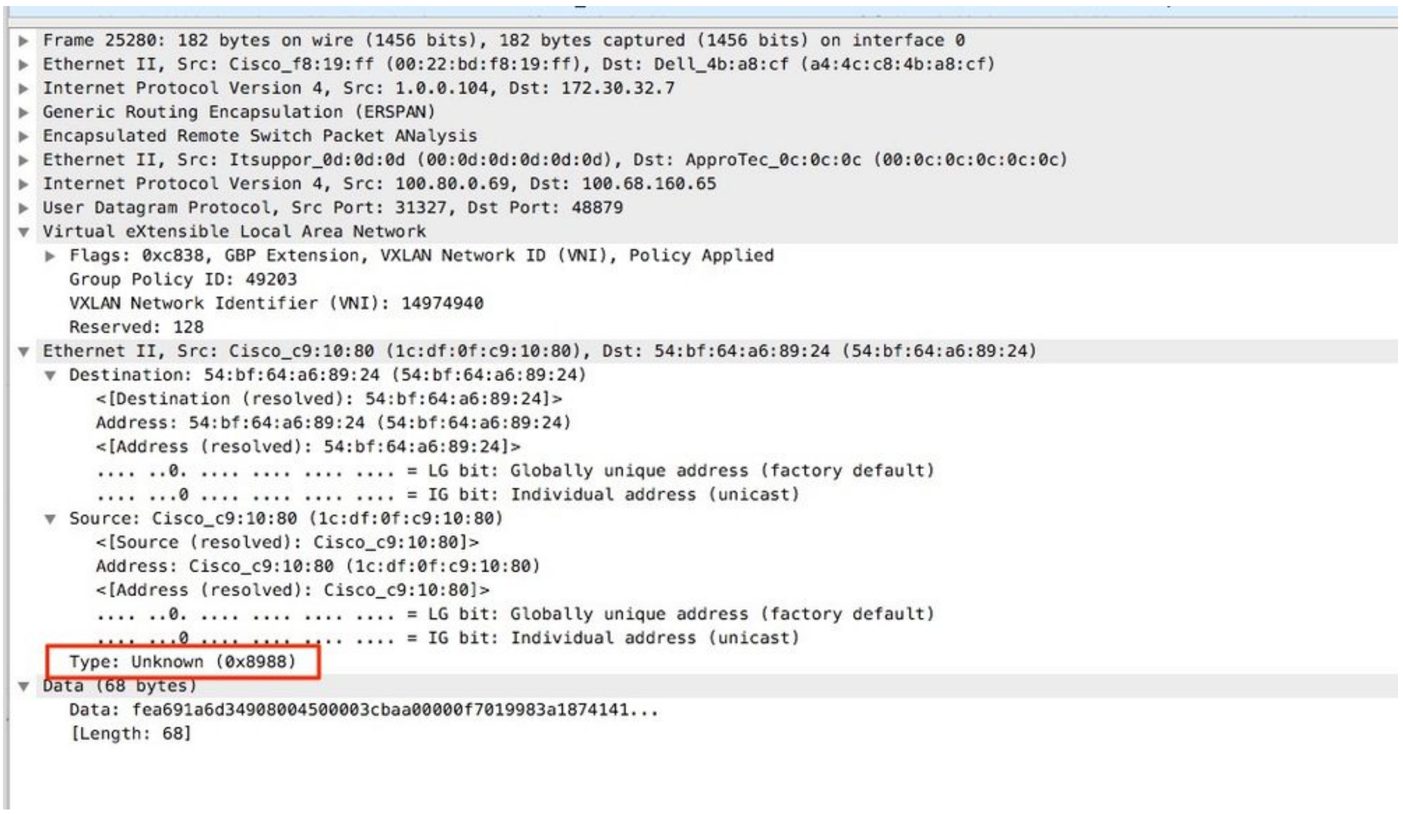
```
No.  Time                          Source        Destination   Protocol
 25 2015-10-23 23:25:00.946480  10.0.0.1      10.0.0.2      TCP   161 12567→58322 [PSH, ACK] Seq=155 Ack=200 Win=331
 26 2015-10-23 23:25:00.962085  10.0.192.92   10.0.32.66    UDP   198 Source port: 7248  Destination port: 48879
 27 2015-10-23 23:25:00.962224  10.0.192.95   10.0.192.92   UDP   198 Source port: 25168  Destination port: 48879
 28 2015-10-23 23:25:00.986492  10.0.0.2      10.0.0.1      TCP   116 58322→12567 [ACK] Seq=200 Ack=200 Win=331 Len=
 29 2015-10-23 23:25:01.039557  10.0.0.3      10.0.0.1      TCP   166 50193→12567 [PSH, ACK] Seq=196 Ack=200 Win=331
 30 2015-10-23 23:25:01.039671  10.0.0.1      10.0.0.3      TCP   116 12567→50193 [ACK] Seq=200 Ack=246 Win=331 Len=
 31 2015-10-23 23:25:01.039933  10.0.0.3      10.0.0.1      TCP  1415 50193→12567 [PSH, ACK] Seq=246 Ack=200 Win=331
 32 2015-10-23 23:25:01.040066  10.0.192.92   10.0.32.65    UDP   160 Source port: 62672  Destination port: 48879
 33 2015-10-23 23:25:01.040079  10.0.0.1      10.0.0.3      TCP   116 12567→50193 [ACK] Seq=200 Ack=1545 Win=331 Len=
```

iVxLAN header uses destination port 48879. So, you can decode iVxLAN header as well as VxLAN if you configure UDP destination port 48879 as VxLAN on Wireshark.

1. Please ensure that you select iVxLAN encapsulated packets first.
2. Navigate to Analyze > Decode As > Transport > UDP destination (48879) > VxLAN.
3. And then Apply.

Note: There are communication packets between APICs on Fabric ports. Those packets are not encapsulated by iVxLAN header.

When you take an erspan capture on a user network that runs Precision Time Protocol (PTP) sometimes it is seen that Wireshark does not interpret the data due to an unknown ethertype within the GRE encap (0x8988). 0x8988 is the ethertype for the time tag that is inserted into dataplane packets when PTP is enabled. Decode the ethertype 0x8988 as "Cisco ttag" to expose the details of the packet.



```
▶ Frame 25280: 182 bytes on wire (1456 bits), 182 bytes captured (1456 bits) on interface 0
▶ Ethernet II, Src: Cisco_f8:19:ff (00:22:bd:f8:19:ff), Dst: Dell_4b:a8:cf (a4:4c:c8:4b:a8:cf)
▶ Internet Protocol Version 4, Src: 1.0.0.104, Dst: 172.30.32.7
▶ Generic Routing Encapsulation (ERSPAN)
▶ Encapsulated Remote Switch Packet ANalysis
▶ Ethernet II, Src: Itsuppor_0d:0d:0d (00:0d:0d:0d:0d:0d), Dst: ApproTec_0c:0c:0c (00:0c:0c:0c:0c:0c)
▶ Internet Protocol Version 4, Src: 100.80.0.69, Dst: 100.68.160.65
▶ User Datagram Protocol, Src Port: 31327, Dst Port: 48879
▼ Virtual eXtensible Local Area Network
  ▶ Flags: 0xc838, GBP Extension, VXLAN Network ID (VNI), Policy Applied
    Group Policy ID: 49203
    VXLAN Network Identifier (VNI): 14974940
    Reserved: 128
▼ Ethernet II, Src: Cisco_c9:10:80 (1c:df:0f:c9:10:80), Dst: 54:bf:64:a6:89:24 (54:bf:64:a6:89:24)
  ▼ Destination: 54:bf:64:a6:89:24 (54:bf:64:a6:89:24)
      <[Destination (resolved): 54:bf:64:a6:89:24]>
      Address: 54:bf:64:a6:89:24 (54:bf:64:a6:89:24)
      <[Address (resolved): 54:bf:64:a6:89:24]>
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  ▼ Source: Cisco_c9:10:80 (1c:df:0f:c9:10:80)
      <[Source (resolved): Cisco_c9:10:80]>
      Address: Cisco_c9:10:80 (1c:df:0f:c9:10:80)
      <[Address (resolved): Cisco_c9:10:80]>
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: Unknown (0x8988)
▼ Data (68 bytes)
    Data: fea691a6d34908004500003cbaa00000f7019983a1874141...
    [Length: 68]
```