Troubleshoot Firepower Threat Defense (FTD) Cluster

Contents

Introduction

This document describes the troubleshooting of a cluster setup on the Firepower Next-Generation Firewall (NGFW).

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics (see Related Information section for links):

- Firepower platform architecture
- Firepower Cluster configuration and operation
- Familiarity with the FTD and Firepower eXtensible Operating System (FXOS) CLI
- NGFW/data plane logs
- NGFW/data plane packet-tracer
- FXOS/data plane captures

Components Used

- HW: Firepower 4125
- SW: 6.7.0 (Build 65) â€“ data plane 9.15(1)

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

Most of the items covered in this document are also fully applicable to the Adaptive Security Appliance (ASA) cluster troubleshooting.

Configure

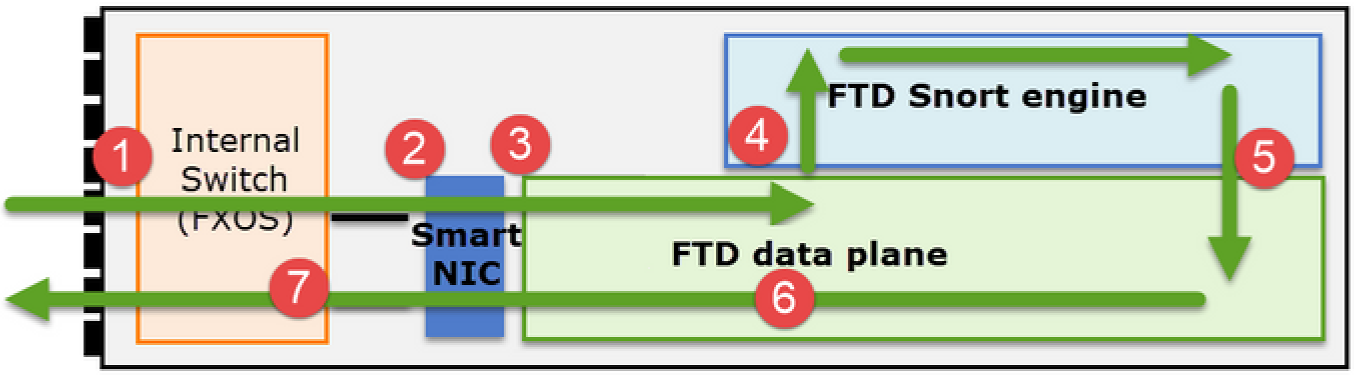
The configuration part of a cluster deployment is covered in the FMC and FXOS configuration guides:

- Clustering for the Firepower Threat Defense
- Deploying a Cluster for Firepower Threat Defense for Scalability and High Availability

Cluster Basics

NGFW Architecture

It is important to understand how a Firepower 41xx or 93xx series handles transit packets:



1. A packet enters the ingress interface, and it is handled by the chassis internal switch.
2. The packet goes through the Smart NIC. If the flow is offloaded (HW acceleration), then the packet is handled solely by the Smart NIC, and then is sent back to the network.
3. If the packet is not offloaded, it enters the FTD data plane which does mainly L3/L4 checks.
4. If the policy requires it, the packet is inspected by the Snort engine (mainly L7 inspection).
5. The Snort engine returns a verdict (for example, allow or block) for the packet.

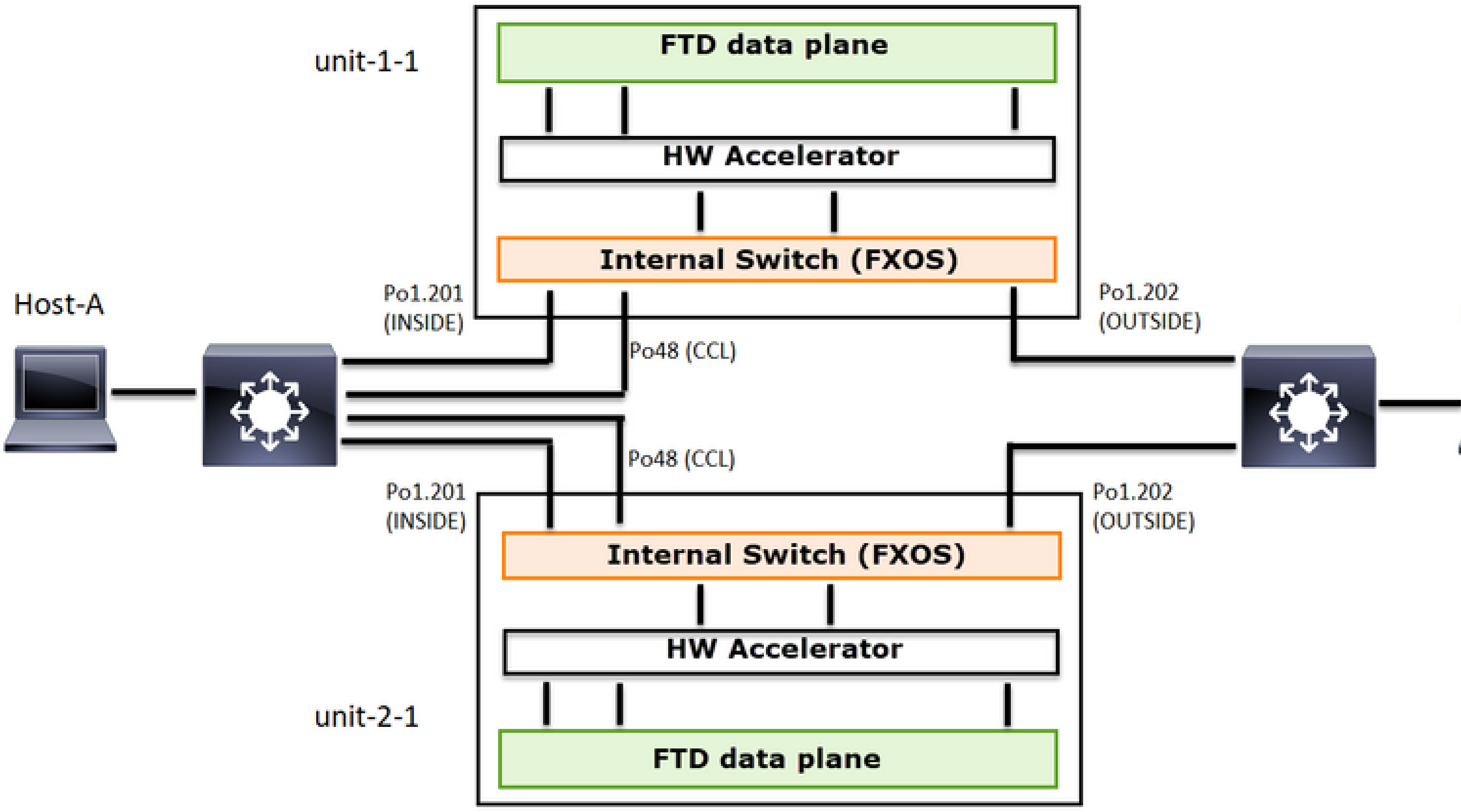
6. The data plane drops or forwards the packet based on Snortâ€™s verdict.
7. The packet egresses the chassis through the internal chassis switch.

Cluster Captures

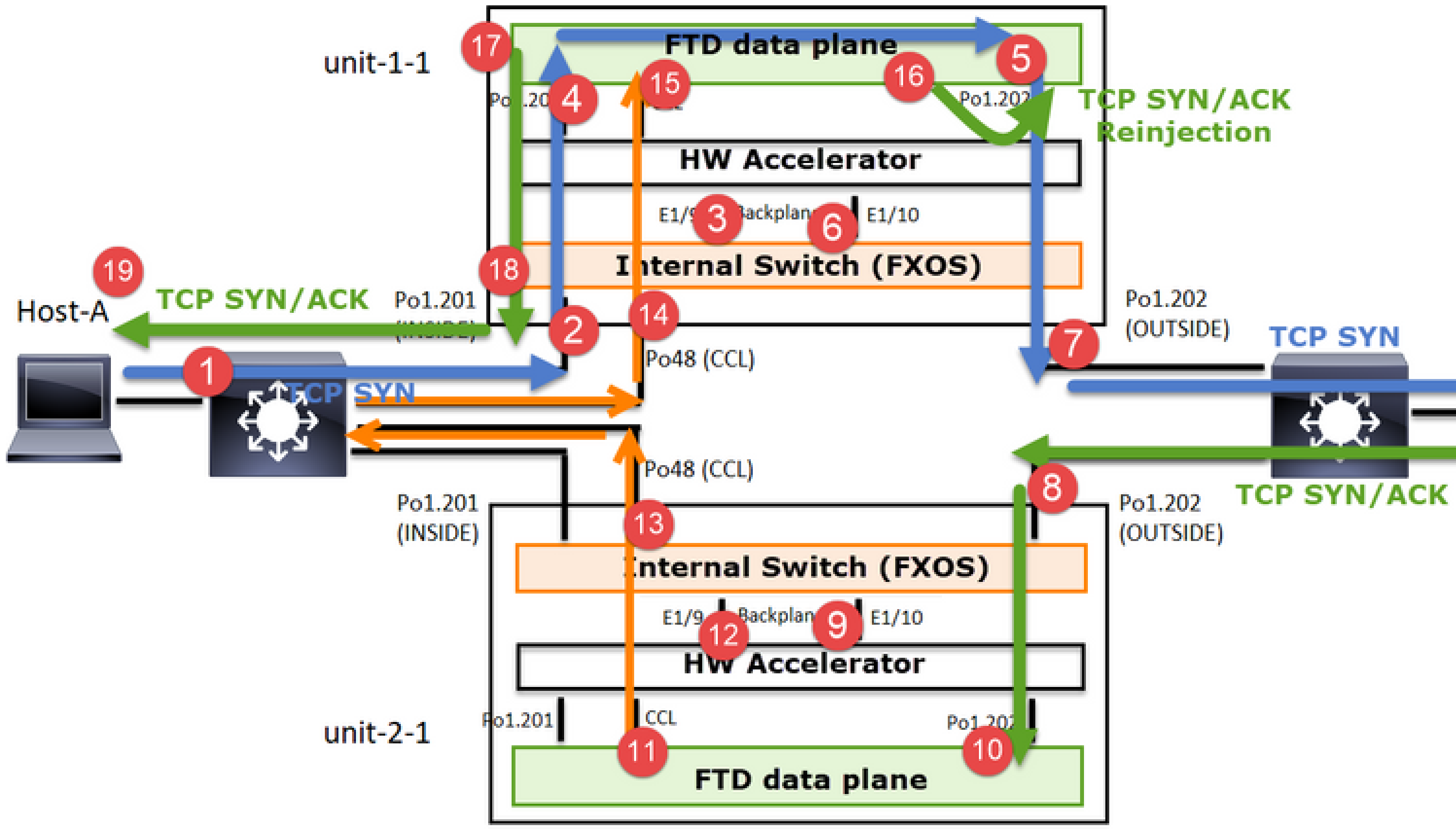
Firepower appliances provide multiple capture points that provide visibility into the transit flows. When you troubleshoot and enable cluster captures the main challenges are:

- The number of captures increases as the number of units in the cluster increase.
- You need to be aware of the way the cluster handles a specific flow to be able to track the packet through the cluster.

This diagram shows a 2-unit cluster (for example, FP941xx/FP9300):



In case of an asymmetric TCP connection establishment, a TCP SYN, SYN/ACK exchange looks like this:



Forward traffic

1. TCP SYN is sent from Host-A to Host-B.
2. TCP SYN arrives on the chassis (one of the members of Po1).
3. TCP SYN is sent through one of the chassis backplane interfaces (for example, E1/9, E1/10, etc) to the data plane.
4. TCP SYN arrives on the data plane ingress interface (Po1.201/INSIDE). In this example, unit1-1 takes ownership of the flow, does Initial Sequence Number (ISN) randomization, and encodes the ownership (cookie) info in Seq number.
5. TCP SYN is sent out of Po1.202/OUTSIDE (data plane egress interface).
6. TCP SYN arrives on one of the chassis backplane interfaces (for example, E1/9, E1/10, etc).
7. TCP SYN is sent out of the chassis physical interface (one of the members of Po1) towards Host-B.

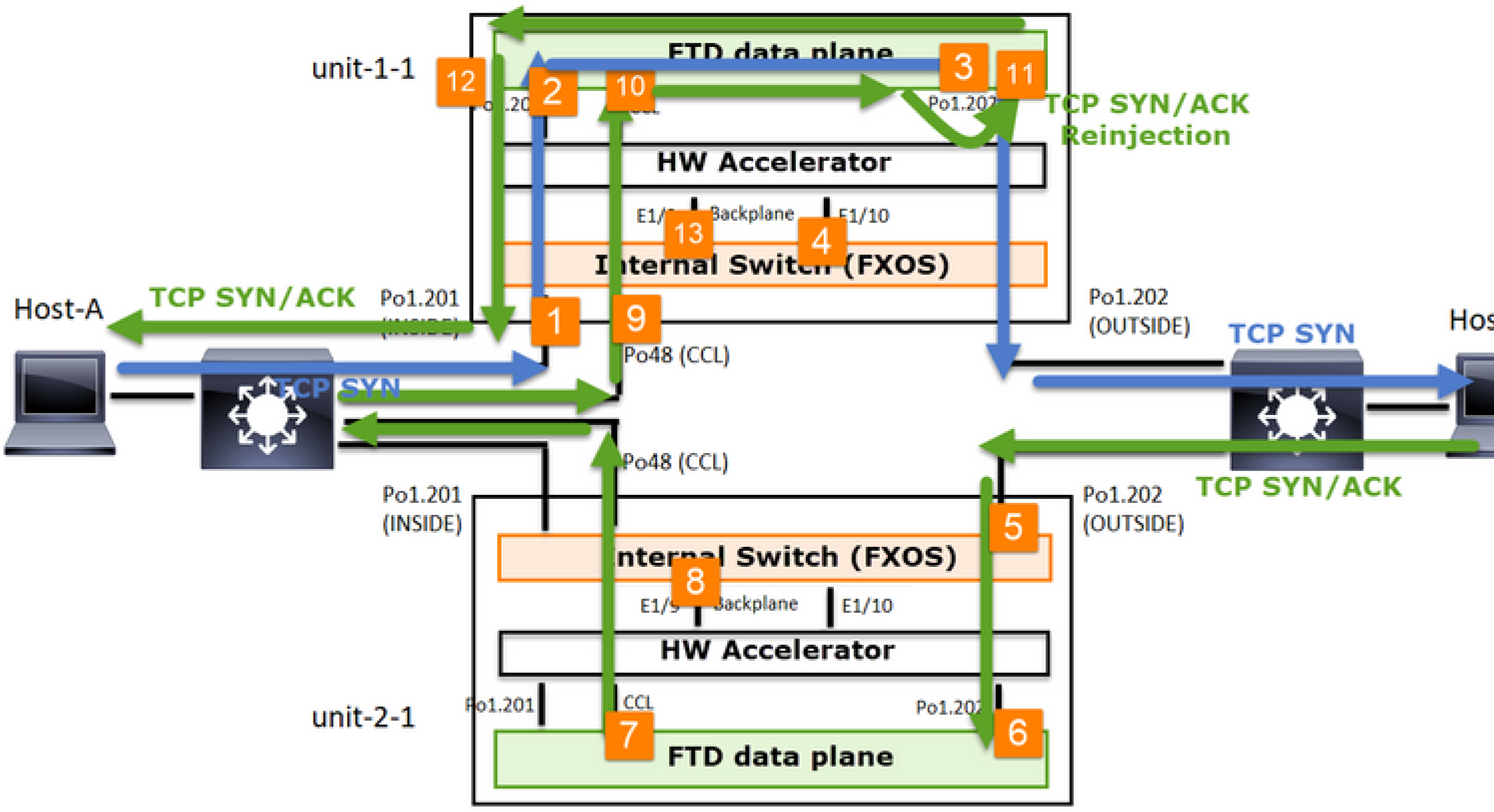
Return traffic

8. TCP SYN/ACK is sent from Host-B and arrives on unit-2-1 (one of the members of Po1).
9. TCP SYN/ACK is sent through one of the chassis backplane interfaces (for example, E1/9, E1/10, etc) to the data plane.
10. TCP SYN/ACK arrives on the data plane ingress interface (Po1.202/OUTSIDE).
11. TCP SYN/ACK is sent out of Cluster Control Link (CCL) towards unit-1-1. By default, ISN is enabled. Thus, the forwarder finds the owner info for TCP SYN+ACKs without the involvement of the director. For other packets or when ISN is disabled, the director is queried.
12. TCP SYN/ACK arrives on one of the chassis backplane interfaces (for example, E1/9, E1/10, and so on).
13. TCP SYN/ACK is sent out of the chassis physical interface (one of the members of Po48) towards unit-1-1.
14. TCP SYN/ACK arrives on unit-1-1 (one of the members of Po48).
15. TCP SYN/ACK is forwarded through one of the chassis backplane interfaces to the data plane CCL port-channel interface (nameif cluster).
16. The data plane reinjects the TCP SYN/ACK packet to the data plane interface Po1.202/OUTSIDE.

17. TCP SYN/ACK is sent out of Po1.201/INSIDE (data plane egress interface) towards HOST-A.
18. The TCP SYN/ACK traverses one of the chassis backplane interfaces (for example, E1/9, E1/10, etc) and egresses one of the members of Po1.
19. TCP SYN/ACK arrives on Host-A.

For more details about this scenario, read the related section in the Cluster Connection Establishment Case Studies.

Based on this packet exchange, all the possible cluster capture points are:



For the forward traffic (for example, TCP SYN) capture on:

1. The chassis physical interface (for example, Po1 members). This capture is configured from the Chassis Manager (CM) UI or the CM CLI.
2. Data plane ingress interface (for example, Po1.201 INSIDE).
3. Data plane egress interface (for example, Po1.202 OUTSIDE).
4. Chassis backplane interfaces. On FP4100 there are 2 backplane interfaces. On FP9300 there are a total of 6 (2 per module). Since you do not know in which interface the packet arrives, you must enable capture on all interfaces.

For the return traffic (for example, TCP SYN/ACK) capture on:

5. The chassis physical interface (for example, Po1 members). This capture is configured from the Chassis Manager (CM) UI or the CM CLI.
6. Data plane ingress interface (for example, Po1.202 OUTSIDE).
7. Since the packet is redirected the next capture point is the data plane CCL.
8. Chassis backplane interfaces. Again, you must enable capture on both interfaces.
9. Unit-1-1 chassis CCL member interfaces.
10. Data plane CCL interface (nameif cluster).
11. Ingress interface (Po1.202 OUTSIDE). This is the reinjected packet from CCL to the data plane.
12. Data plane egress interface (for example, Po1.201 INSIDE).

13. Chassis backplane interfaces.

How to Enable the Cluster Captures

FXOS Captures

The process is described in the FXOS Config Guide: Packet Capture

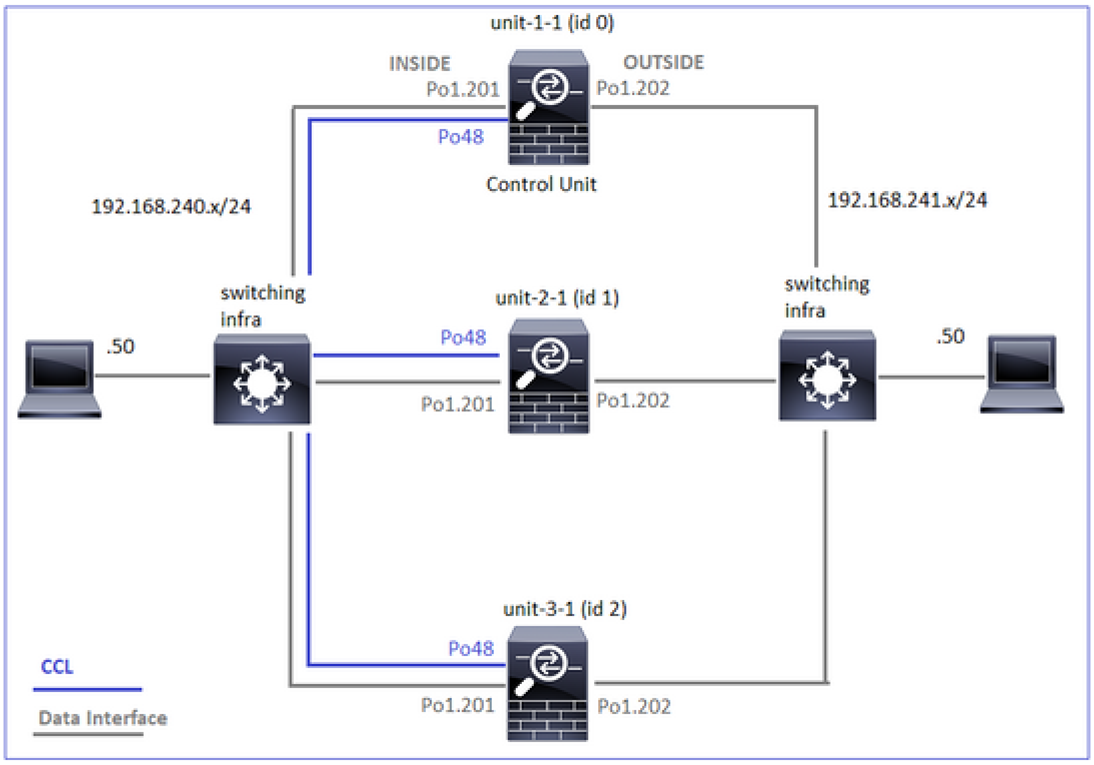
---

Note: FXOS captures can be only taken in the ingress direction from the internal switch point of view.

Data Plane Captures

The recommended way to enable capture on all cluster members is with the **cluster exec** command.

Consider a 3-unit cluster:



To verify if there are active captures in all cluster units, use this command:

<#root>

```
firepower#
```

`cluster exec show capture`

```
unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
firepower#
```

To enable a data plane capture on all units on Po1.201 (INSIDE):

<#root>

```
firepower#

cluster exec capture CAPI interface INSIDE
```

It is highly recommended to specify a capture filter, and in case you expect a lot of traffic, to increase the capture buffer:

<#root>

```
firepower#

cluster exec capture CAPI buffer 33554432 interface INSIDE match tcp host 192.168.240.50 host 192.168.24
```

Verification

<#root>

```
firepower#

cluster exec show capture

unit-1-1(LOCAL):******************************************************
capture CAPI type raw-data buffer 33554432 interface INSIDE [Capturing - 5140 bytes]
  match tcp host 192.168.240.50 host 192.168.241.50 eq www

unit-2-1:*************************************************************
capture CAPI type raw-data buffer 33554432 interface INSIDE [Capturing - 260 bytes]
  match tcp host 192.168.240.50 host 192.168.241.50 eq www

unit-3-1:*************************************************************
capture CAPI type raw-data buffer 33554432 interface INSIDE [Capturing - 0 bytes]
  match tcp host 192.168.240.50 host 192.168.241.50 eq www
```

To see the contents of all captures (this output can be very long):

<#root>

```
firepower#

terminal pager 24

firepower#

cluster exec show capture CAPI

unit-1-1(LOCAL):******************************************************
```

```
21 packets captured

1: 11:33:09.879226 802.1Q vlan#201 P0 192.168.240.50.45456 > 192.168.241.50.80: S 2225395909:2225395909(
2: 11:33:09.880401 802.1Q vlan#201 P0 192.168.241.50.80 > 192.168.240.50.45456: S 719653963:719653963(0)
3: 11:33:09.880691 802.1Q vlan#201 P0 192.168.240.50.45456 > 192.168.241.50.80: . ack 719653964 win 229
4: 11:33:09.880783 802.1Q vlan#201 P0 192.168.240.50.45456 > 192.168.241.50.80: P 2225395910:2225396054(

unit-2-1:*************************************************************
0 packet captured
0 packet shown

unit-3-1:*************************************************************
0 packet captured
0 packet shown
```

Capture Traces

If you want to see how the ingress packets are handled by the data plane on each unit, use the **trace** keyword. This traces the first 50 ingress packets. You can trace up to 1000 ingress packets.

> **Note**: In case you have multiple captures applied on an interface, you can trace a single packet only once.

To trace the first 1000 ingress packets on interface OUTSIDE on all cluster units:

<#root>

```
firepower#
```

`cluster exec cap CAPO int OUTSIDE buff 33554432 trace trace-count 1000 match tcp host 192.168.240.50 hos`

Once you capture the flow of interest there is a need to ensure that you trace the packets of interest on each unit. The important thing to remember is that a specific packet can be #1 on unit-1-1, but #2 on another unit, and so on.

In this example, you can see that the SYN/ACK is packet #2 on unit-2-1, but packet #1 on unit-3-1:

<#root>

```
firepower#
```

`cluster exec show capture CAPO | include S.*ack`

```
unit-1-1(LOCAL):******************************************************
1: 12:58:31.117700 802.1Q vlan#202 P0 192.168.240.50.45468 > 192.168.241.50.80: S 441626016:441626016(0)
2: 12:58:31.118341 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.45468:
```

`S`

```
 301658077:301658077(0)
```

`ack`

```
 441626017 win 28960 <mss 1460,sackOK,timestamp 1125686319 1115330849,nop,wscale 7>

unit-2-1:*************************************************************

unit-3-1:*************************************************************
1: 12:58:31.111429 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.45468:
```

S

```
 301658077:301658077(0)
```

ack

```
 441626017 win 28960 <mss 1460,sackOK,timestamp 1125686319 1115330849,nop,wscale 7>
```

To trace packet #2 (SYN/ACK) on the local unit:

<#root>

```
firepower#
```

cluster exec show cap CAPO packet-number 2 trace

```
unit-1-1(LOCAL):******************************************************

2: 12:58:31.118341 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.45468:
```

S

```
 301658077:301658077(0)
```

ack

```
 441626017 win 28960 <mss 1460,sackOK,timestamp 1125686319 1115330849,nop,wscale 7>
Phase: 1
Type: CAPTURE
Subtype:
Result: ALLOW
Config:
Additional Information:
MAC Access list
...
```

To trace the same packet (SYN/ACK) on the remote unit:

<#root>

```
firepower#
```

cluster exec unit unit-3-1 show cap CAPO packet-number 1 trace

```
1: 12:58:31.111429 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.45468:
```

S

```
 301658077:301658077(0)
```

```
ack

 441626017 win 28960 <mss 1460,sackOK,timestamp 1125686319 1115330849,nop,wscale 7>
Phase: 1
Type: CAPTURE
Subtype:
Result: ALLOW
Config:
Additional Information:
MAC Access list
...
```

CCL Capture

To enable capture on the CCL link (on all units):

<#root>

```
firepower#

cluster exec capture CCL interface cluster


unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

Reinject Hide

By default, a capture enabled on a data plane data interface shows all packets:

- The ones that arrive from the physical network
- The ones that are reinjected from the CCL

If you do not want to see the reinjected packets, use the **reinject-hide** option. This can be helpful if you want to verify if a flow is asymmetric:

<#root>

```
firepower#

cluster exec capture CAPI_RH reinject-hide interface INSIDE match tcp host 192.168.240.50 host 192.168.2
```

This capture only shows you what the local unit actually receives on the specific interface directly from the physical network, and not from the other cluster units.

ASP drops

If you want to check for software drops for a specific flow, you can enable **asp-drop** capture. If you do not know which drop reason to focus on, use the keyword **all**. Additionally, if you are not interested in the packet payload, you can specify the **headers-only** keyword. This allows you to capture 20-30 times more packets:

```
<#root>

firepower#

cluster exec cap ASP type asp-drop all buffer 33554432 headers-only

unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

Additionally, you can specify the IPs of interest in the ASP capture:

```
<#root>

firepower#

cluster exec cap ASP type asp-drop all buffer 33554432 headers-only


match ip host 192.0.2.100 any
```

Clear a Capture

To clear the buffer of any capture that runs in all cluster units. This does not stop the captures, but only clears the buffers:

```
<#root>

firepower#

cluster exec clear capture /all

unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

Stop a Capture

There are 2 ways to stop an active capture on all cluster units. Later you can resume.

Way 1

```
<#root>
```

```
firepower#
cluster exec cap CAPI stop

unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

To resume

<#root>

```
firepower#
cluster exec no capture CAPI stop

unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

Way 2

<#root>

```
firepower#
cluster exec no capture CAPI interface INSIDE

unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

To resume

<#root>

```
firepower#
cluster exec capture CAPI interface INSIDE

unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

Collect a Capture

There are multiple ways to export a capture.

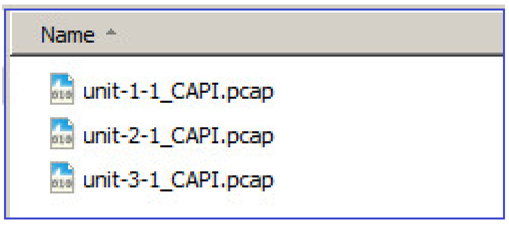
Way 1 â€“ To a remote server

This allows you to upload a capture from the data plane to a remote server (for example, TFTP). The capture names are automatically changed to reflect the source unit:

<#root>

```
firepower#
cluster exec copy /pcap capture:CAPI tftp://192.168.240.55/CAPI.pcap

unit-1-1(LOCAL):******************************************************

Source capture name [CAPI]?

Address or name of remote host [192.168.240.55]?

Destination filename [CAPI.pcap]?

INFO: Destination filename is changed to unit-1-1_CAPI.pcap !!!!!!!

81 packets copied in 0.40 secs

unit-2-1:*************************************************************

INFO: Destination filename is changed to unit-2-1_CAPI.pcap !


unit-3-1:*************************************************************

INFO: Destination filename is changed to unit-3-1_CAPI.pcap !
```

The uploaded pcap files:


Name

unit-1-1_CAPI.pcap

unit-2-1_CAPI.pcap

unit-3-1_CAPI.pcap

Way 2 - Fetch the captures from the FMC

This way is only applicable to FTD. First, you copy the capture to the FTD disk:

<#root>

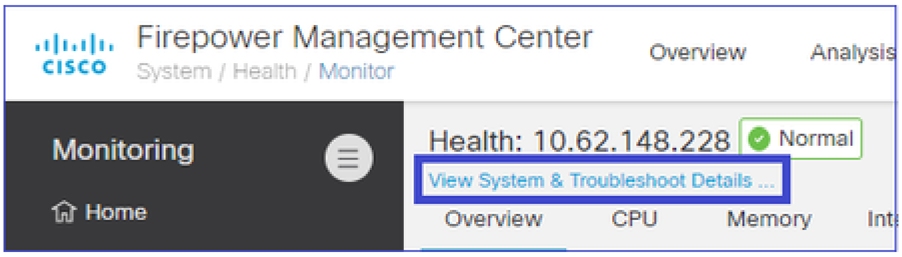
```
firepower#
cluster exec copy /pcap capture:CAPI disk0:CAPI.pcap

unit-1-1(LOCAL):******************************************************

Source capture name [CAPI]?

Destination filename [CAPI.pcap]?
!!!!!
62 packets copied in 0.0 secs
```

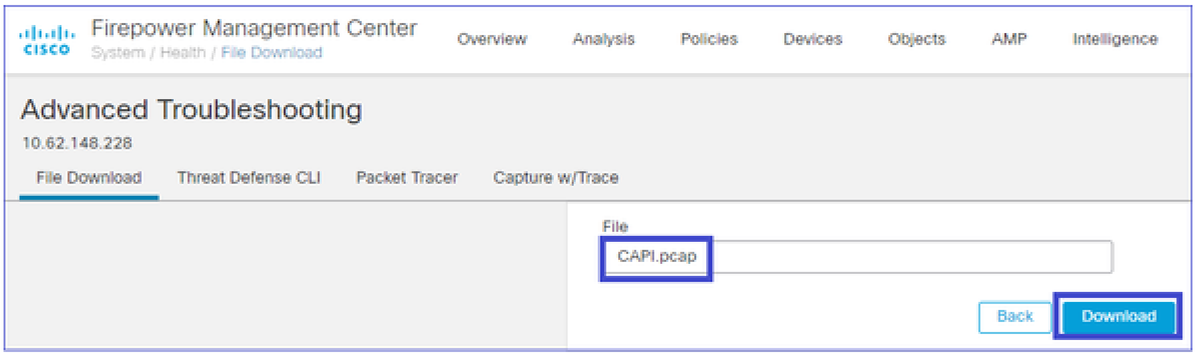
From expert mode, copy the file from /mnt/disk0/ to /ngfw/var/common/ directory:

<#root>

```
>
expert

admin@firepower:~$
cd /mnt/disk0

admin@firepower:/mnt/disk0$
sudo cp CAPI.pcap /ngfw/var/common
```

Finally, on FMC navigate to **System** > **Health** > **Monitor** section. Choose **View System & Troubleshoot Details** > **Advanced Troubleshooting** and fetch the capture file:


Firepower Management Center
System / Health / Monitor
Overview
Analysis
Monitoring
Home
Health: 10.62.148.228
Normal
View System & Troubleshoot Details ...
Overview
CPU
Memory



Delete a Capture

To remove a capture from all cluster units, use this command:

<#root>

```
firepower#
cluster exec no capture CAPI

unit-1-1(LOCAL):******************************************************

unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

Offloaded flows

On FP41xx/FP9300 flows can be offloaded to HW Accelerator either statically (for example, Fastpath rules) or dynamically. For more details about flow-offload, check this document:

https://www.cisco.com/c/en/us/support/docs/security/firepower-ngfw/212321-clarify-the-firepower-threat-defense-acc.html#anc22

If a flow is offloaded, then only a few packets go through the FTD data plane. The rest is handled by the HW accelerator (Smart NIC).

From a capture point of view, this means that if you only enable FTD data plane-level captures you donâ€™t see all the packets that go through the device. In this case, you also need to enable FXOS chassis-level captures.

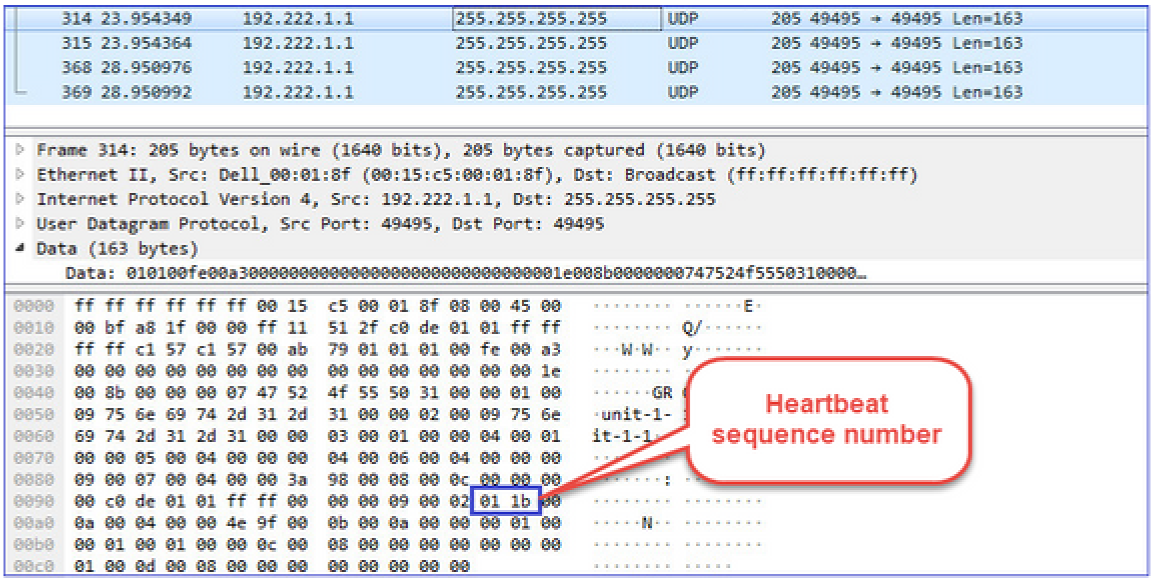
Cluster Control Link (CCL) Messages

If you take a capture on the CCL, you notice that the cluster units exchange different types of messages. The ones of interest are:

Protocol	**Description**
UDP 49495	Cluster heartbeats (keepalives) · L3 broadcast (255.255.255.255) · These packets are sent by every cluster unit at 1/3 of the health-check hold time value. · Note that not all UDP 49495 packets seen in the capture are heartbeats · The heartbeats contain a sequence number.
UDP 4193	Cluster Control Protocol Data Path Messages · Unicast · These packets contain information (metadata) about the flow owner, director, backup owner, and so on. Examples are:

	· A â€˜cluster addâ€™ message is sent from the owner to the director when a new flow is created · A â€˜cluster deleteâ€™ message is sent from the owner to the director when a flow is terminated
Data packets	Data packets that belong to the various traffic flows that traverse the cluster

Cluster Heartbeat



```
314 23.954349    192.222.1.1    255.255.255.255    UDP    205 49495 → 49495 Len=163
315 23.954364    192.222.1.1    255.255.255.255    UDP    205 49495 → 49495 Len=163
368 28.950976    192.222.1.1    255.255.255.255    UDP    205 49495 → 49495 Len=163
369 28.950992    192.222.1.1    255.255.255.255    UDP    205 49495 → 49495 Len=163

▷ Frame 314: 205 bytes on wire (1640 bits), 205 bytes captured (1640 bits)
▷ Ethernet II, Src: Dell_00:01:8f (00:15:c5:00:01:8f), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▷ Internet Protocol Version 4, Src: 192.222.1.1, Dst: 255.255.255.255
▷ User Datagram Protocol, Src Port: 49495, Dst Port: 49495
◢ Data (163 bytes)
    Data: 010100fe00a30000000000000000000000000000001e008b00000007475245f5550310000…

0000  ff ff ff ff ff ff 00 15  c5 00 01 8f 08 00 45 00   ········ ······E·
0010  00 bf a8 1f 00 00 ff 11  51 2f c0 de 01 01 ff ff   ········ Q/······
0020  ff ff c1 57 c1 57 00 ab  79 01 01 01 00 fe 00 a3   ···W·W·· y·······
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 1e   ········
0040  00 8b 00 00 00 07 47 52  4f 55 50 31 00 00 01 00   ······GR
0050  09 75 6e 69 74 2d 31 2d  31 00 00 02 00 09 75 6e   ·unit-1-
0060  69 74 2d 31 2d 31 00 00  03 00 01 00 00 04 00 01   it-1-1
0070  00 00 05 00 04 00 00 00  04 00 06 00 04 00 00 00
0080  09 00 07 00 04 00 00 3a  98 00 08 00 0c 00 00 00   ·······:
0090  00 c0 de 01 01 ff ff 00  00 00 09 00 02 01 1b 00   ········ ········
00a0  0a 00 04 00 00 4e 9f 00  0b 00 0a 00 00 00 01 00   ·····N·· ········
00b0  00 01 00 01 00 00 0c 00  08 00 00 00 00 00 00 00   ········ ········
00c0  01 00 0d 00 08 00 00 00  00 00 00 00 00            ········ ·····
```

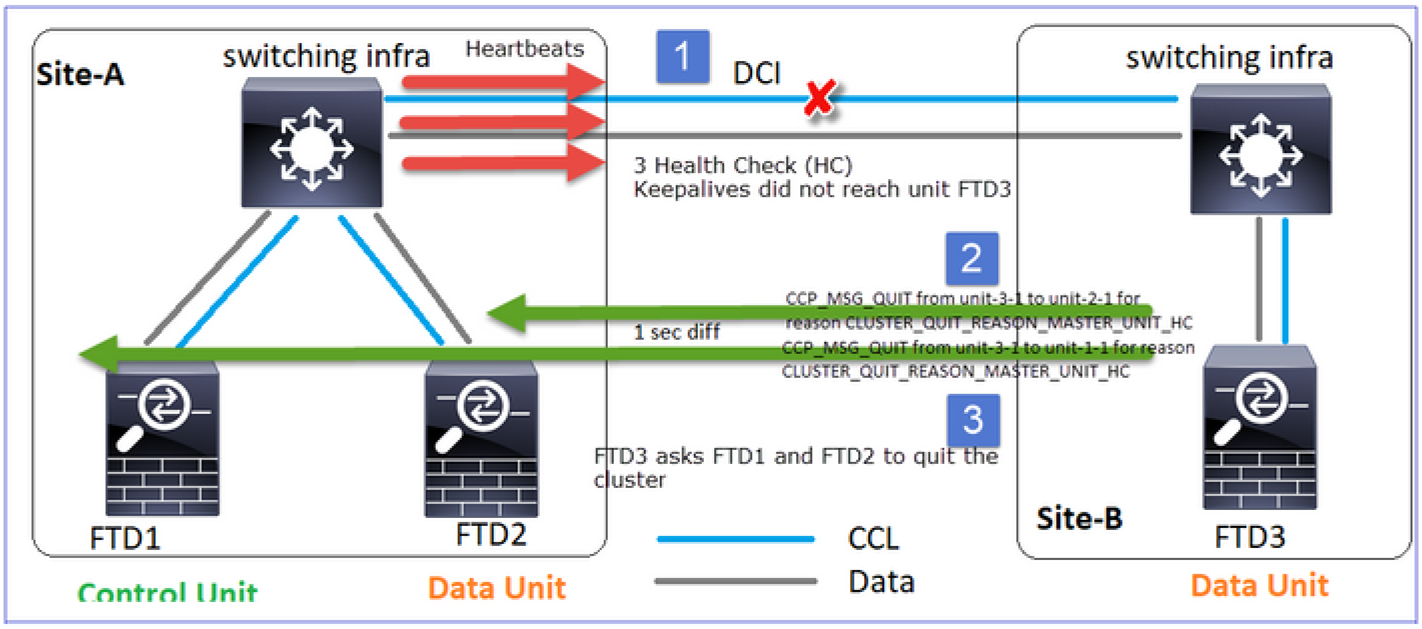
Cluster Control Point (CCP) Messages

In addition to the heartbeat messages, there are a number of cluster control messages that are exchanged through the CCL in specific scenarios. Some of them are unicast messages while others are broadcasts.

CLUSTER_QUIT_REASON_PRIMARY_UNIT_HC

Whenever a unit loses 3 consecutive heartbeat messages from the control node it generates a CLUSTER_QUIT_REASON_PRIMARY_UNIT_HC message over the CCL. This message:

- Is a unicast.
- It is sent to each of the units with a 1-sec interval.
- When a unit receives this message, quits the cluster (DISABLED) and re-joins.



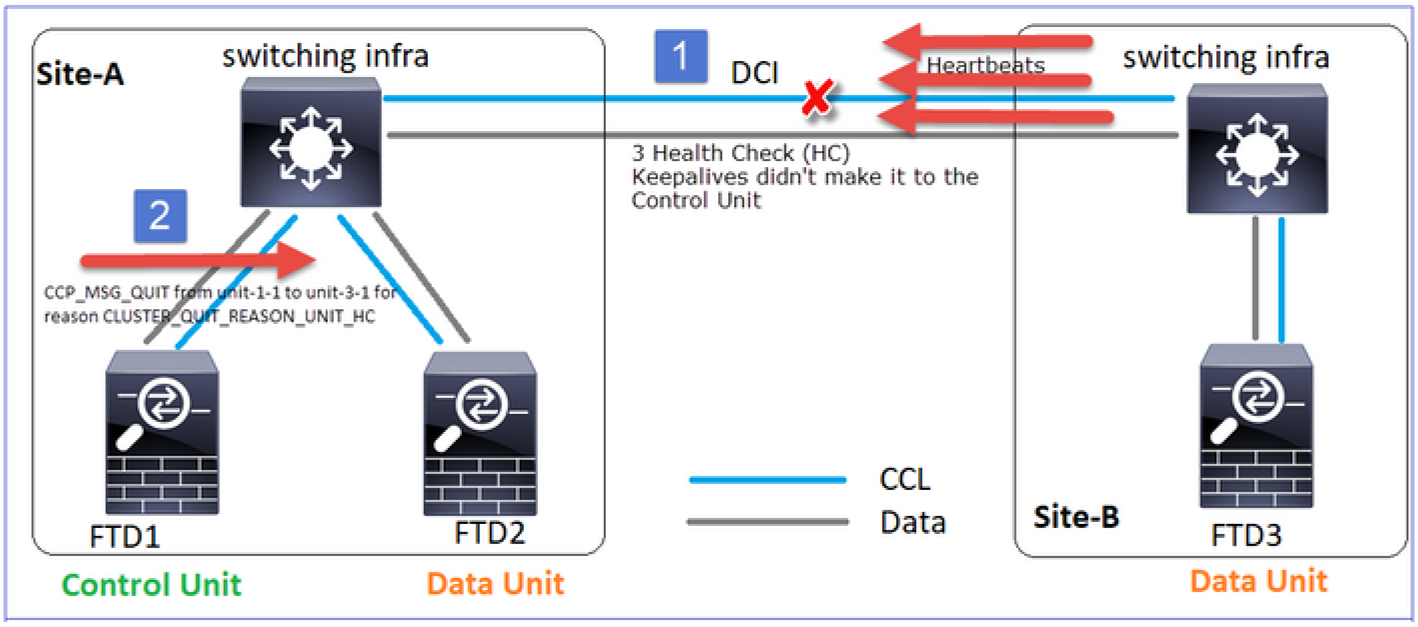
Site-A
switching infra
Heartbeats
1
DCI
3 Health Check (HC) Keepalives did not reach unit FTD3
switching infra
2
CCP_MSG_QUIT from unit-3-1 to unit-2-1 for reason CLUSTER_QUIT_REASON_MASTER_UNIT_HC
1 sec diff
CCP_MSG_QUIT from unit-3-1 to unit-1-1 for reason CLUSTER_QUIT_REASON_MASTER_UNIT_HC
3
FTD3 asks FTD1 and FTD2 to quit the cluster
FTD1
FTD2
CCL
Data
Site-B
FTD3
Control Unit
Data Unit
Data Unit

Q. What is the purpose of the CLUSTER_QUIT_REASON_PRIMARY_UNIT_HC?

A. From unit-3-1â€™s (Site-B) point of view, it loses connection to both unit-1-1 and unit-2-1 from site A, so it needs to remove them from its member list as soon as possible, otherwise, it can have packet lost if unit-2-1 is still in its member list and unit-2-1 happens to be a director of a connection, and flow query to unit-2-1 fails.

CLUSTER_QUIT_REASON_UNIT_HC

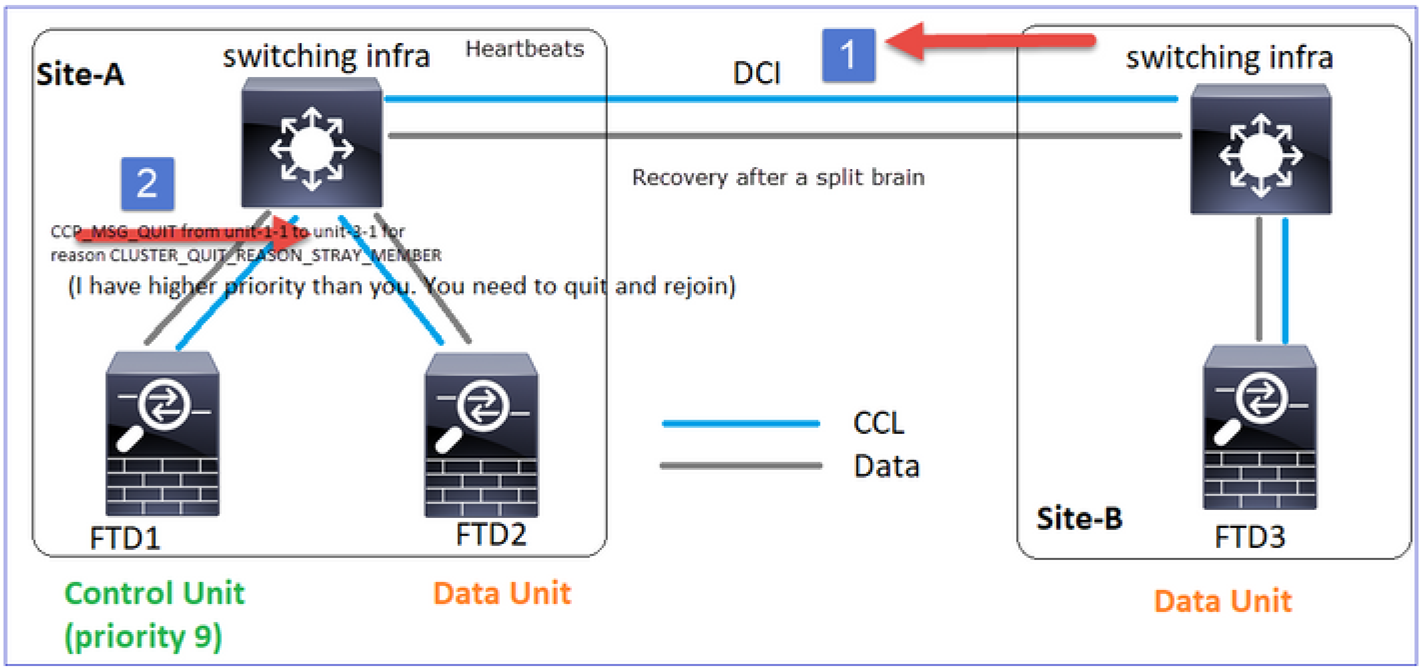
Whenever the control node loses 3 consecutive heartbeat messages from a data node, it sends CLUSTER_QUIT_REASON_UNIT_HC message over the CCL. This message is unicast.



Site-A
switching infra
1
DCI
Heartbeats
switching infra
3 Health Check (HC) Keepalives didn't make it to the Control Unit
2
CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER_QUIT_REASON_UNIT_HC
FTD1
FTD2
CCL
Data
Site-B
FTD3
Control Unit
Data Unit
Data Unit

CLUSTER_QUIT_REASON_STRAY_MEMBER

When a split-partition reconnects with a peer partition, the new data node is treated as a stray member by the dominant control unit and receives a CCP quit message with the reason of CLUSTER_QUIT_REASON_STRAY_MEMBER.


Site-A
switching infra
Heartbeats
DCI
1
switching infra
2
Recovery after a split brain
CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER_QUIT_REASON_STRAY_MEMBER
(I have higher priority than you. You need to quit and rejoin)
CCL
Data
FTD1
FTD2
Site-B
FTD3
Control Unit (priority 9)
Data Unit
Data Unit

CLUSTER_QUIT_MEMBER_DROPOUT

A broadcast message that is generated by a data node, and is sent as a broadcast. Once a unit receives this message, moves to DISABLED status. Additionally, auto-rejoin does not kick-off:

<#root>

```
firepower#

show cluster info trace | include DROPOUT

Nov 04 00:22:54.699 [DBUG]Receive CCP message: CCP_MSG_QUIT from unit-3-1 to unit-1-1 for reason

CLUSTER_QUIT_MEMBER_DROPOUT

Nov 04 00:22:53.699 [DBUG]Receive CCP message: CCP_MSG_QUIT from unit-3-1 to unit-2-1 for reason

CLUSTER_QUIT_MEMBER_DROPOUT
```

The cluster history shows:

<#root>

```
PRIMARY       DISABLED      Received control message DISABLE (

member dropout announcement

)
```

Cluster Health-Check (HC) Mechanism

Main points

- Each cluster unit sends a heartbeat every 1/3 of the health-check hold time value to all other units (broadcast 255.255.255.255) and uses UDP port 49495 as transport over the CCL.
- Each cluster unit tracks independently every other unit with a Poll timer and a Poll count value.
- If a cluster unit does not receive any packet (heartbeat or data packet) from a cluster peer unit within a heartbeat interval, it increases the Poll count value.
- When the Poll count value for a cluster peer unit becomes 3 the peer is considered down.
- Whenever a heartbeat is received, its sequence number is checked and in case the diff with the previously received heartbeat is different than 1, the heartbeat drop counter increases accordingly.
- If the Poll count counter for a cluster peer is different than 0, and a packet is received by the peer, the counter is reset to a 0 value.

Use this command to check the cluster health counters:

<#root>

```
firepower#

show cluster info health details


----------------------------------------------------------------------------------
|               Unit (ID)| Heartbeat| Heartbeat|   Average|   Maximum|      Poll|
|                        |     count|     drops|  gap (ms)| slip (ms)|     count|
----------------------------------------------------------------------------------
|           unit-2-1 ( 1)|       650|         0|      4999|         1|         0|
|           unit-3-1 ( 2)|       650|         0|      4999|         1|         0|
----------------------------------------------------------------------------------
```

Description of the main columns

Column	**Description**
Unit (ID)	The ID of the remote cluster peer.
Heartbeat count	The number of heartbeats received from the remote peer over the CCL.
Heartbeat drops	The number of missed heartbeats. This counter is calculated based on the received heartbeat sequence number.
Average gap	The average time interval of the received heartbeats.
Poll count	When this counter becomes 3 the unit is removed from the cluster. The poll query interval is the same as the heartbeat interval but runs independently.

To reset the counters use this command:

<#root>

```
firepower#

clear cluster info health details
```

Q. How to verify the heartbeat frequency?

A. Check the average gap value:

<#root>

```
firepower#

show cluster info health details

----------------------------------------------------------------------------------
|                Unit (ID)| Heartbeat| Heartbeat|

Average

|   Maximum|      Poll|
|                         |     count|     drops|

gap (ms)

| slip (ms)|     count|
----------------------------------------------------------------------------------
|            unit-2-1 ( 1)|      3036|         0|

999

|         1|         0|
----------------------------------------------------------------------------------
```

Q. How can you change the cluster hold time on FTD?

A. Use FlexConfig

Q. Who becomes the control node after a split-brain?

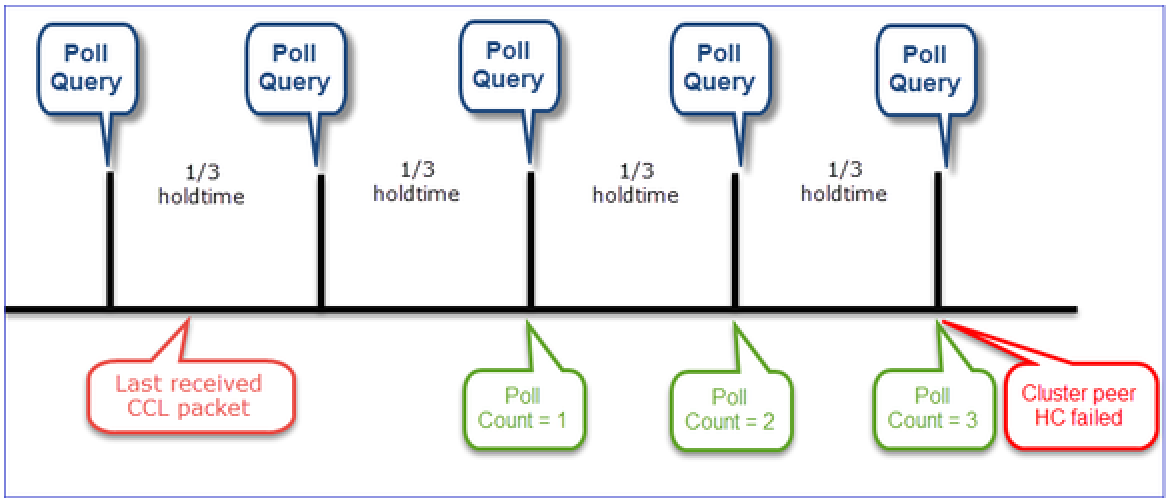
A. The unit with the highest priority (lowest number):

<#root>

```
firepower#

show run cluster | include priority



priority 9
```

Check HC failure scenario 1 for more details.

The cluster HC mechanism visualization



Indicative timers: The min and max depend on the last received CCL packet arrival.

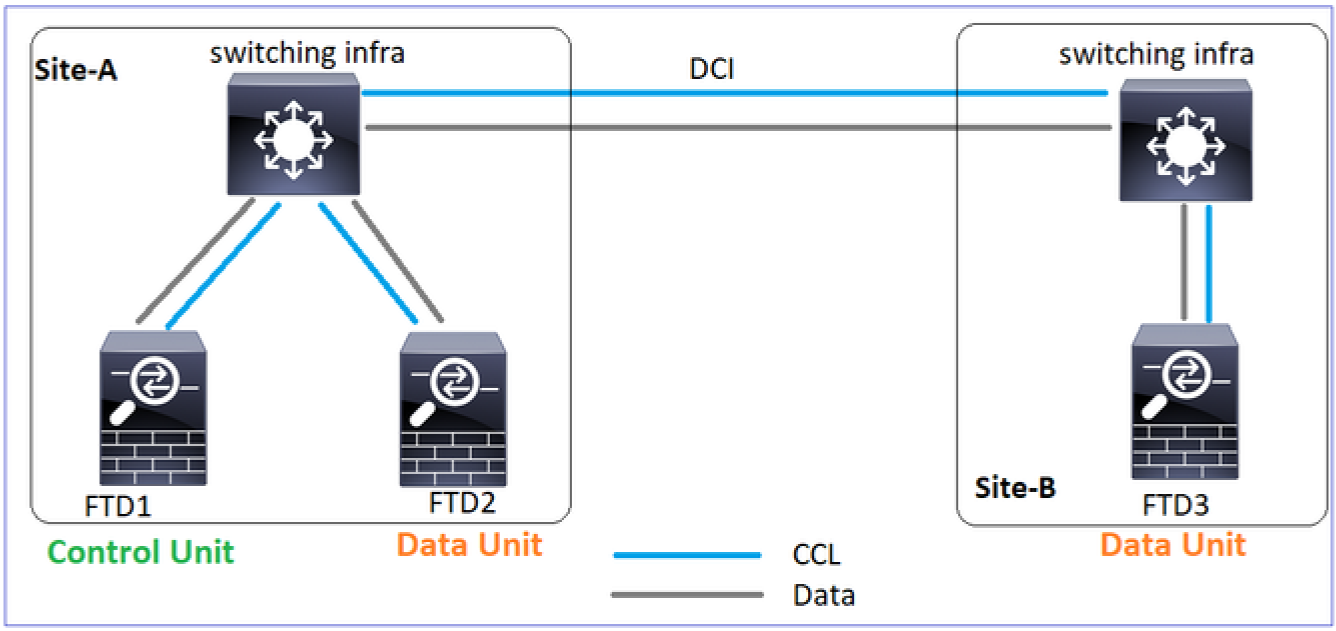
Hold time	**Poll query check (frequency)**	**Min detection time**	**Max detection time**
3 sec (default)	~1 sec	~3.01 sec	~3.99 sec
4 sec	~1.33 sec	~4.01 sec	~5.32 sec
5 sec	~1.66 sec	~5.01 sec	~6.65 sec
6 sec	~2 sec	~6.01 sec	~7.99 sec
7 sec	~2.33 sec	~7.01 sec	~9.32 sec
8 sec	~2.66 sec	~8.01 sec	~10.65 sec

Cluster HC Failure Scenarios

The goals of this section are to demonstrate:

- Different cluster HC failure scenarios.
- How the different logs and command outputs can be correlated.

Topology



Cluster configuration

Unit-1-1	**Unit-2-1**

`cluster group GROUP1`
` key *****`
` local-unit unit-1-1`
` cluster-interface Port-channel48 ip 10.17.1.1 255.255.0.0`
` priority 9`
` health-check holdtime 3`
` health-check data-interface auto-rejoin 3 5 2`
` health-check cluster-interface auto-rejoin unlimited 5 1`
` health-check system auto-rejoin 3 5 2`
` health-check monitor-interface debounce-time 500`
` site-id 1`
` enable`
	`cluster group GROUF`
	` key *****`
	` local-unit unit-2-`
	` cluster-interface`
	` priority 17`
	` health-check holdt`
	` health-check data-`
	` health-check clust`
	` health-check syste`
	` health-check monit`
	` site-id 1`
	` enable`

Cluster status

Unit-1-1	**Unit-2-1**

<#root>
`firepower#`
`show cluster info`
`Cluster GROUP1: On`
`    Interface mode: spanned`
	<#root>
	`firepower#`
	`show cluster info`
	`Cluster GROUP1: On`
	`    Interface mode: spanned`

```
This is "unit-1-1" in state PRIMARY

        ID         : 0
        Site ID    : 1
        Version    : 9.12(2)33
        Serial No.: FCH22247LNK
        CCL IP     : 10.17.1.1
        CCL MAC    : 0015.c500.018f
        Last join : 20:25:36 UTC Nov 1 2020
        Last leave: 20:25:28 UTC Nov 1 2020
Other members in the cluster:

Unit "unit-3-1" in state secondary

        ID         : 1
        Site ID    : 2
        Version    : 9.12(2)33
        Serial No.: FCH22247MKJ
        CCL IP     : 10.17.3.1
        CCL MAC    : 0015.c500.038f
        Last join : 20:58:45 UTC Nov 1 2020
        Last leave: 20:58:37 UTC Nov 1 2020

Unit "unit-2-1" in state SECONDARY

        ID         : 2
        Site ID    : 1
        Version    : 9.12(2)33
        Serial No.: FCH23157Y9N
        CCL IP     : 10.17.2.1
        CCL MAC    : 0015.c500.028f
        Last join : 20:44:45 UTC Nov 1 2020
        Last leave: 20:44:38 UTC Nov 1 2020
```

```
This is "unit-2-1" in state SECONDARY

        ID         : 2
        Site ID    : 1
        Version    : 9.12(2)33
        Serial No.: FCH23157Y9N
        CCL IP     : 10.17.2.1
        CCL MAC    : 0015.c500.028f
        Last join : 20:44:46 UTC N
        Last leave: 20:44:38 UTC N
Other members in the cluster:

 Unit "unit-1-1" in state PRIMARY

        ID         : 0
        Site ID    : 1
        Version    : 9.12(2)33
        Serial No.: FCH22247LNK
        CCL IP     : 10.17.1.1
        CCL MAC    : 0015.c500.018f
        Last join : 20:25:36 UTC N
        Last leave: 20:25:28 UTC N

Unit "unit-3-1" in state SECONDARY

        ID         : 1
        Site ID    : 2
        Version    : 9.12(2)33
        Serial No.: FCH22247MKJ
        CCL IP     : 10.17.3.1
        CCL MAC    : 0015.c500.038f
        Last join : 20:58:45 UTC N
        Last leave: 20:58:37 UTC N
```

Scenario 1

CCL communication loss for ~4+ sec in both directions.

Before the failure

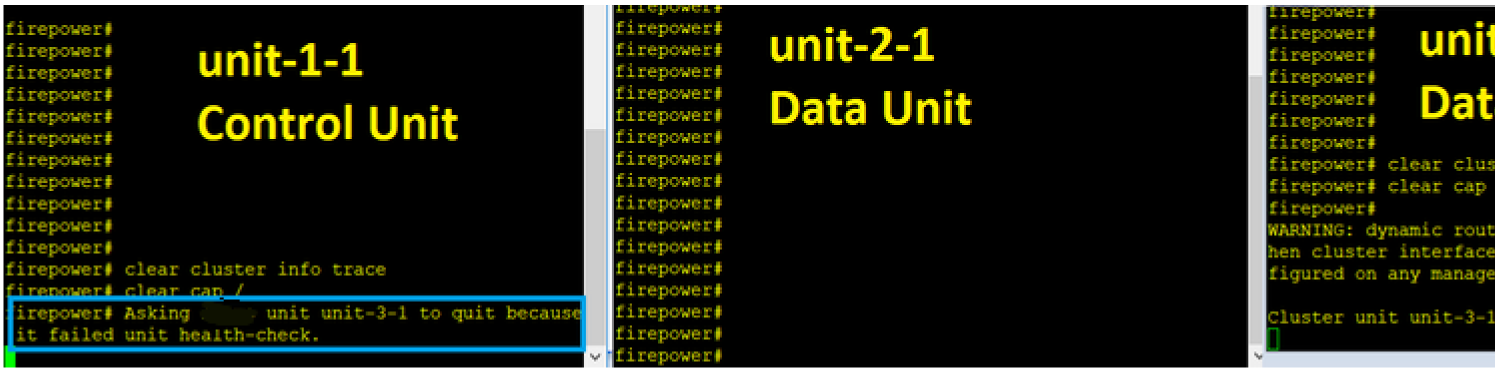
FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Control node	Data node	Data node

After the recovery (no changes in the unit roles)

FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Control node	Data node	Data node

Analysis

The failure (CCL communication was lost).



```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# clear cluster info trace
firepower# clear cap /
firepower# Asking [redacted] unit unit-3-1 to quit because
it failed unit health-check.
```

```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
```

```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# clear clus
firepower# clear cap
firepower#
WARNING: dynamic rout
hen cluster interface
figured on any manage

Cluster unit unit-3-1
```

The data plane console message on unit-3-1:

<#root>

```
firepower#
WARNING: dynamic routing is not supported on management interface when cluster interface-mode is 'spanne
If dynamic routing is configured on any management interface, please remove it.



Cluster unit unit-3-1 transitioned from SECONDARY to PRIMARY


Cluster disable is performing cleanup..done.
All data interfaces have been shutdown due to clustering being disabled.
To recover either enable clustering or remove cluster group configuration.
```

Unit-1-1 cluster trace logs:

<#root>

```
firepower#

show cluster info trace | include unit-3-1


Nov 02 09:38:14.239 [INFO]Notify chassis de-bundle port for blade unit-3-1, stack 0x000055a8918307fb 0x0
Nov 02 09:38:14.239 [INFO]FTD - CD proxy received state notification (DISABLED) from unit unit-3-1
Nov 02 09:38:14.239
```

[DBUG]Send CCP message to all: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER_QUIT_MEMBER_DRO

```
Nov 02 09:38:14.239 [INFO]Notify chassis de-bundle port for blade unit-3-1, stack 0x000055a8917eb596 0x0
Nov 02 09:38:14.239
```

[DBUG]Send CCP message to id 1: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER_QUIT_REASON_UN

```
Nov 02 09:38:14.239 [CRIT]Received heartbeat event 'SECONDARY heartbeat failure' for member unit-3-1 (ID
```

Split-brain

Unit-1-1	**Unit-2-1**

<#root>
`firepower#`
`show cluster info`
 	
`Cluster GROUP1: On`
`    Interface mode: spanned`
 	
`This is "unit-1-1" in state PRIMARY`
 	
`        ID        : 0`
`        Site ID   : 1`
`        Version   : 9.12(2)33`
`        Serial No.: FCH22247LNK`
`        CCL IP    : 10.17.1.1`
`        CCL MAC   : 0015.c500.018f`
`        Last join : 20:25:36 UTC Nov 1 2020`
`        Last leave: 20:25:28 UTC Nov 1 2020`
`Other members in the cluster:`
`    Unit "unit-2-1" in state SECONDARY`
`        ID        : 2`
`        Site ID   : 1`
`        Version   : 9.12(2)33`
`        Serial No.: FCH23157Y9N`
`        CCL IP    : 10.17.2.1`
`        CCL MAC   : 0015.c500.028f`
`        Last join : 20:44:45 UTC Nov 1 2020`
`        Last leave: 20:44:38 UTC Nov 1 2020`
	<#root>
	`firepower#`
	`show cluster info`
	
	`Cluster GROUP1: On`
	`    Interface mode: spanned`
	`    This is "unit-2-1" in state SE`
	`        ID        : 2`
	`        Site ID   : 1`
	`        Version   : 9.12(2)33`
	`        Serial No.: FCH23157Y9N`
	`        CCL IP    : 10.17.2.1`
	`        CCL MAC   : 0015.c500.028f`
	`        Last join : 20:44:46 UTC N`
	`        Last leave: 20:44:38 UTC N`
	`Other members in the cluster:`
	
	`Unit "unit-1-1" in state PRIMARY`
	
	`        ID        : 0`
	`        Site ID   : 1`
	`        Version   : 9.12(2)33`
	`        Serial No.: FCH22247LNK`
	`        CCL IP    : 10.17.1.1`
	`        CCL MAC   : 0015.c500.018f`
	`        Last join : 20:25:36 UTC N`
	`        Last leave: 20:25:28 UTC N`

Cluster history

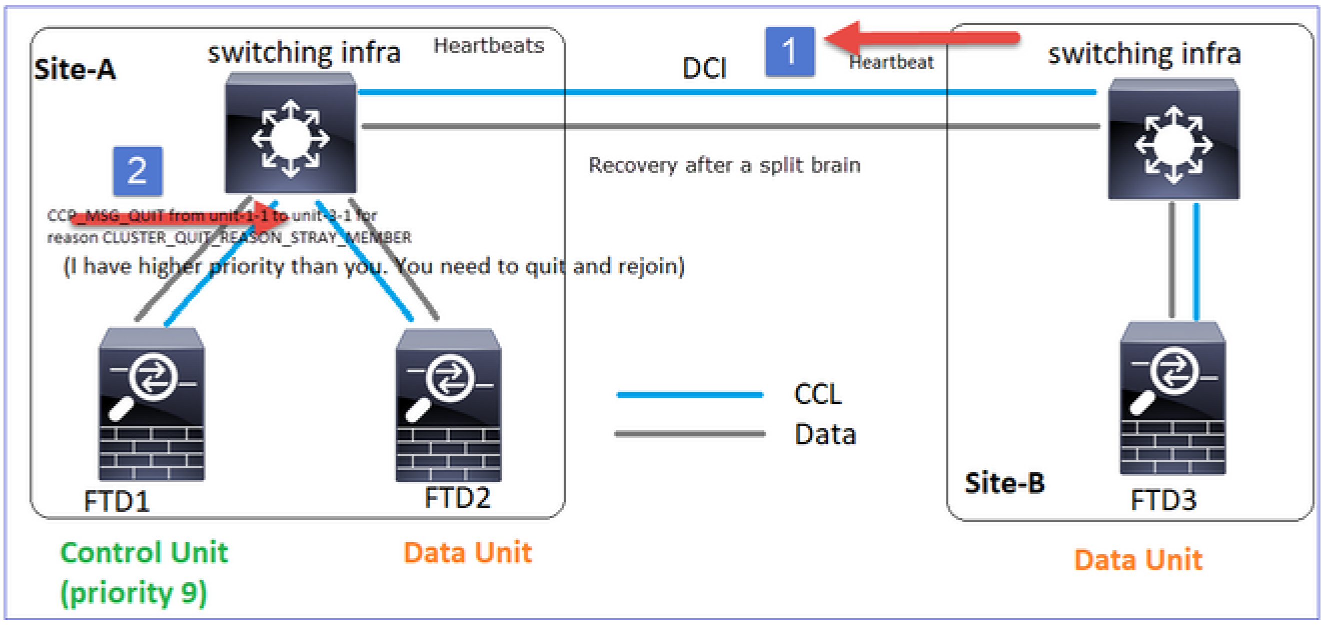
Unit-1-1	Unit-2-1	Unit-3-1

No events	No events	<#root>
		`09:38:16 UTC Nov 2 2020`
		`SECONDARY                  PRIMARY_POST_CONFIG    Primary relinquish`
		
		`09:38:17 UTC Nov 2 2020`
		`PRIMARY_POST_CONFIG    Primary                Primary post config do`

CCL communication restoration

Unit-1-1 detects the current control node and since unit-1-1 has higher priority sends to unit-3-1 a CLUSTER_QUIT_REASON_STRAY_MEMBER message to trigger a new election process. In the end, unit-3-1 re-joins as a data node.

When a split-partition reconnects with a peer partition, the data node is treated as a stray member by the dominant control node and receives a CCP quit msg with a reason of CLUSTER_QUIT_REASON_STRAY_MEMBER.


Site-A
switching infra
Heartbeats
DCI
1
Heartbeat
switching infra
Recovery after a split brain
2
CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER_QUIT_REASON_STRAY_MEMBER
(I have higher priority than you. You need to quit and rejoin)
CCL
Data
FTD1
FTD2
Site-B
FTD3
Control Unit (priority 9)
Data Unit
Data Unit

<#root>

```
Unit-3-1 console logs show:


Cluster unit unit-3-1 transitioned from PRIMARY to DISABLED


The 3DES/AES algorithms require a Encryption-3DES-AES activation key.


Detected Cluster Primart.
```

```
Beginning configuration replication from Primary.
WARNING: Local user database is empty and there are still 'aaa' commands for 'LOCAL'.
..
Cryptochecksum (changed): a9ed686f 8e2e689c 2553a104 7a2bd33a
End configuration replication from Primary.
```

`Cluster unit unit-3-1 transitioned from DISABLED to SECONDARY`

Both units (unit-1-1 and unit-3-1) show in their cluster logs:

<#root>

`firepower#`

`show cluster info trace | include retain`

```
Nov 03 21:20:23.019 [CRIT]Found a split cluster with both unit-1-1 and unit-3-1 as primary units. Primar
Nov 03 21:20:23.019 [CRIT]Found a split cluster with both unit-1-1 and unit-3-1 as primary units. Primar
```

There are also syslog messages generated for the split-brain:

<#root>

`firepower#`

`show log | include 747016`

```
Nov 03 2020 21:20:23: %FTD-4-747016: Clustering: Found a split cluster with both unit-1-1 and unit-3-1 a
Nov 03 2020 21:20:23: %FTD-4-747016: Clustering: Found a split cluster with both unit-1-1 and unit-3-1 a
```

Cluster history

Unit-1-1	Unit-2-1	Unit-3-1
No events	No events	<#root> 09:47:33 UTC Nov 2 2020 **Primary DISABLED Detected a splitted cluster** 09:47:38 UTC Nov 2 2020 DISABLED ELECTION Enabled from CLI 09:47:38 UTC Nov 2 2020 ELECTION SECONDARY_COLD Received cluster control r

```
09:47:38 UTC Nov 2 2020
SECONDARY_COLD             SECONDARY_APP_SYNC  Client progression do
09:48:18 UTC Nov 2 2020
SECONDARY_APP_SYNC         SECONDARY_CONFIG    SECONDARY application
09:48:29 UTC Nov 2 2020
SECONDARY_CONFIG           SECONDARY_FILESYS   Configuration replica
09:48:30 UTC Nov 2 2020
SECONDARY_FILESYS          SECONDARY_BULK_SYNC Client progression do
09:48:54 UTC Nov 2 2020
SECONDARY_BULK_SYNC

SECONDARY

 Client progression done
```

Scenario 2

CCL communication loss for ~3-4 sec in both directions.

Before the failure

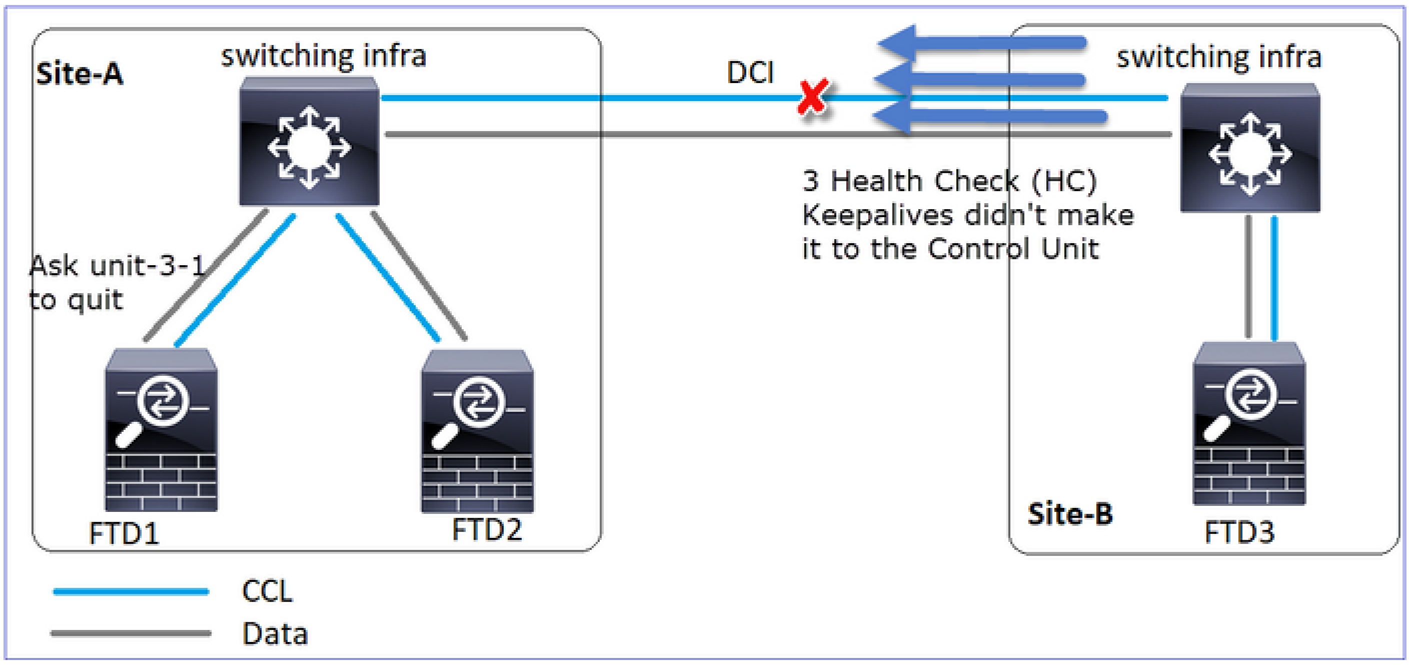
FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Control node	Data node	Data node

After the recovery (no changes in the unit roles)

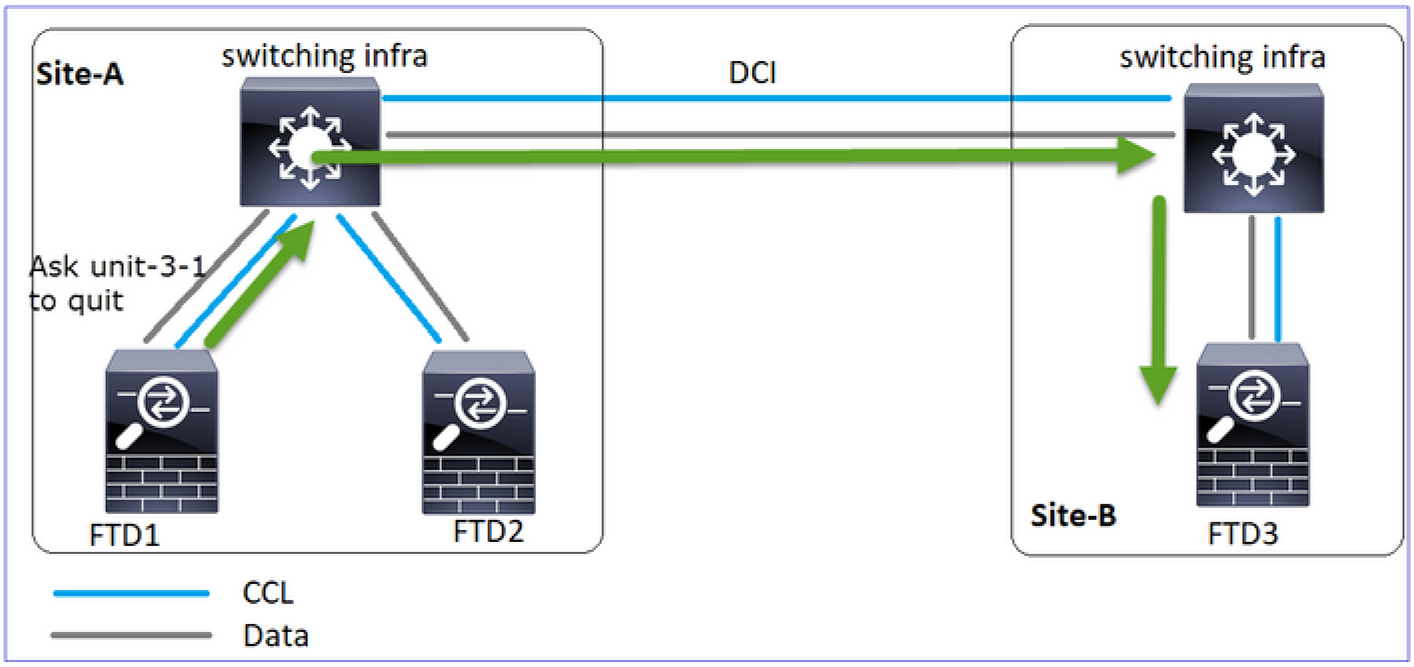
FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Control node	Data node	Data node

Analysis

Event 1: The control node loses 3 HCs from the unit-3-1 and sends a message to unit-3-1 to leave the cluster.



Event 2: The CCL recovered very fast and the CLUSTER_QUIT_REASON_STRAY_MEMBER message from the control node made it to the remote side. Unit-3-1 goes directly to DISABLED mode and there is no split-brain



On unit-1-1 (control) you see:

<#root>

```
firepower#
Asking SECONDARY unit unit-3-1 to quit because it failed unit health-check.


**Forcing stray member unit-3-1 to leave the cluster**
```

On unit-3-1 (data node) you see:

<#root>

```
firepower#


Cluster disable

 is performing cleanup..done.
All data interfaces have been shutdown due to clustering being disabled. To recover either enable cluste

Cluster unit unit-3-1 transitioned from SECONDARY to DISABLED
```

Cluster unit unit-3-1 transitioned to a DISABLED state and once the CCL communication is restored it re-joins as a data node:

<#root>

```
firepower#

show cluster history


20:58:40 UTC Nov 1 2020


SECONDARY                 DISABLED              Received control message DISABLE (stray member)


20:58:45 UTC Nov 1 2020
DISABLED              ELECTION              Enabled from CLI
20:58:45 UTC Nov 1 2020
ELECTION              SECONDARY_COLD            Received cluster control message
20:58:45 UTC Nov 1 2020
SECONDARY_COLD            SECONDARY_APP_SYNC        Client progression done
20:59:33 UTC Nov 1 2020
SECONDARY_APP_SYNC        SECONDARY_CONFIG          SECONDARY application configuration sync done
20:59:44 UTC Nov 1 2020
SECONDARY_CONFIG          SECONDARY_FILESYS         Configuration replication finished
20:59:45 UTC Nov 1 2020
SECONDARY_FILESYS         SECONDARY_BULK_SYNC       Client progression done
21:00:09 UTC Nov 1 2020


SECONDARY_BULK_SYNC       SECONDARY

                Client progression done
```

Scenario 3

CCL communication loss for ~3-4 sec in both directions.

Before the failure.

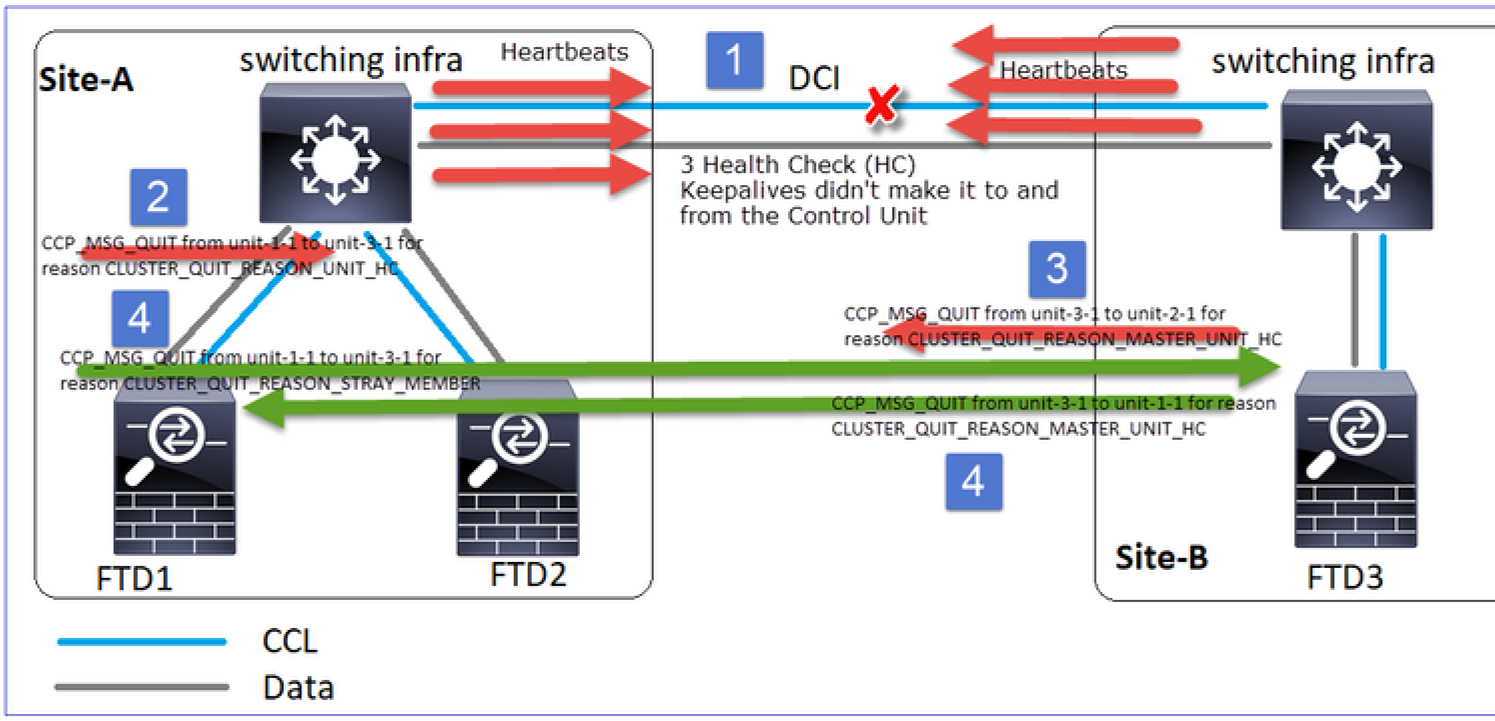
FTD1	**FTD2**	**FTD3**

Site-A	Site-A	Site-B
Control node	Data node	Data node

After the recovery (the control node was changed).

FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Data node	**Control node**	Data node

Analysis



Site-A
switching infra
Heartbeats
1
DCI
Heartbeats
switching infra
3 Health Check (HC) Keepalives didn't make it to and from the Control Unit
2
CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER_QUIT_REASON_UNIT_HC
3
CCP_MSG_QUIT from unit-3-1 to unit-2-1 for reason CLUSTER_QUIT_REASON_MASTER_UNIT_HC
4
CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER_QUIT_REASON_STRAY_MEMBER
CCP_MSG_QUIT from unit-3-1 to unit-1-1 for reason CLUSTER_QUIT_REASON_MASTER_UNIT_HC
4
FTD1
FTD2
Site-B
FTD3
CCL
Data

1. CCL goes down.
2. Unit-1-1 does not get 3 HC messages from unit-3-1 and sends a QUIT message to unit-3-1. This message never reaches unit-3-1.
3. Unit-3-1 sends a QUIT message to unit-2-1. This message never reaches unit-2-1.

CCL recovers.

4. Unit-1-1 sees that unit-3-1 advertised itself as a control node and sends QUIT_REASON_STRAY_MEMBER message to unit-3-1. Once unit-3-1 gets this message goes to a DISABLED state. At the same time, unit-3-1 sends a QUIT_REASON_PRIMARY_UNIT_HC message to unit-1-1 and asks it to quit. Once unit-1-1 gets this message goes to a DISABLED state.

Cluster history

Unit-1-1

<#root>

```
19:53:09 UTC Nov 2 2020

PRIMARY         DISABLED

       Received control message DISABLE
                                   (primary unit health check failure)
19:53:13 UTC Nov 2 2020
DISABLED              ELECTION         Enabled from CLI
19:53:13 UTC Nov 2 2020
ELECTION              SECONDARY_COLD       Received cluster control message
19:53:13 UTC Nov 2 2020
SECONDARY_COLD            SECONDARY_APP_SYNC   Client progression done
19:54:01 UTC Nov 2 2020
SECONDARY_APP_SYNC        SECONDARY_CONFIG     SECONDARY application configur
19:54:12 UTC Nov 2 2020
SECONDARY_CONFIG          SECONDARY_FILESYS    Configuration replication fini
19:54:13 UTC Nov 2 2020
SECONDARY_FILESYS         SECONDARY_BULK_SYNC  Client progression done
19:54:37 UTC Nov 2 2020
SECONDARY_BULK_SYNC

      SECONDARY

           Client progression done
```

Scenario 4

CCL communication loss for ~3-4 sec

Before the failure

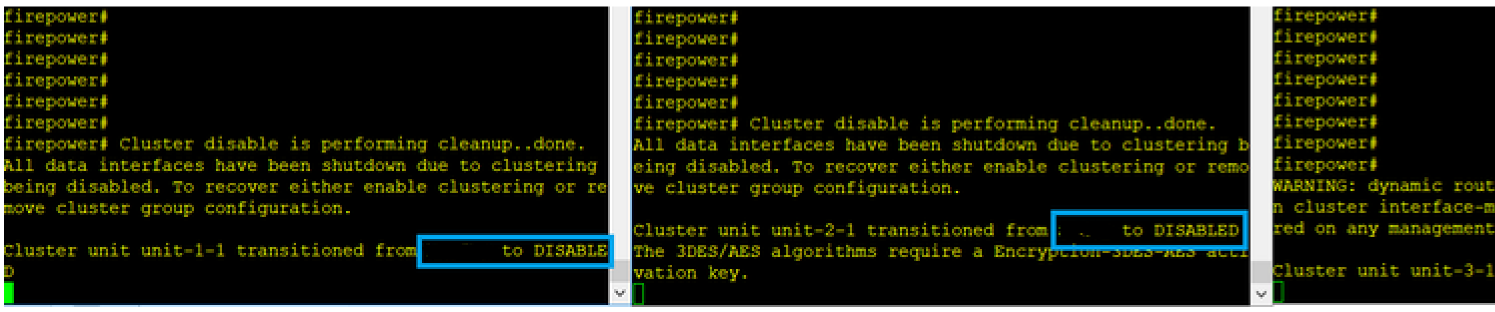
FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Control node	Data node	Data node

After the recovery (the control node changed sites)

FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Data node	Data node	**Control node**

Analysis

The failure

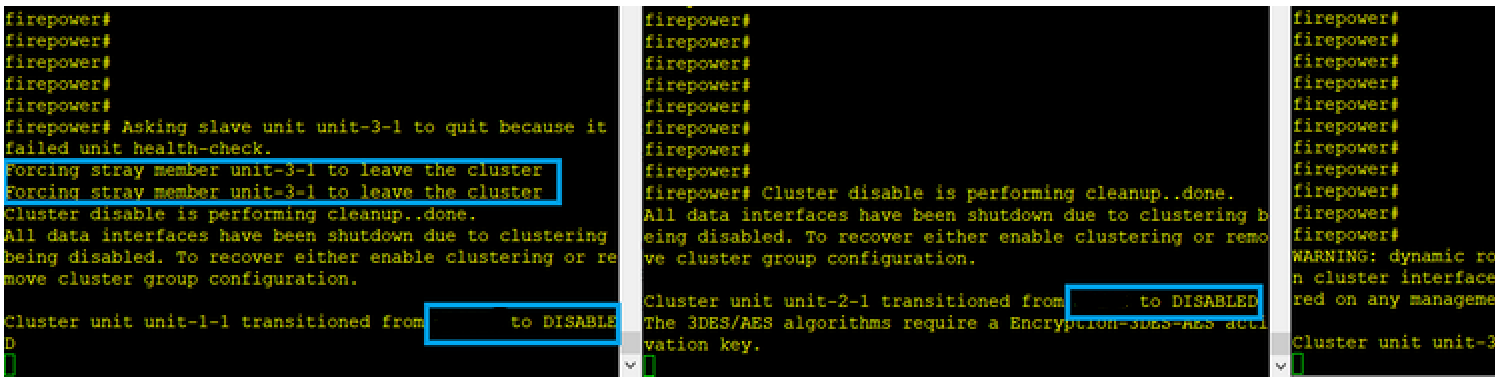


```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# Cluster disable is performing cleanup..done.
All data interfaces have been shutdown due to clustering
being disabled. To recover either enable clustering or re
move cluster group configuration.

Cluster unit unit-1-1 transitioned from [redacted] to DISABLE
D
```

```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# Cluster disable is performing cleanup..done.
All data interfaces have been shutdown due to clustering b
eing disabled. To recover either enable clustering or remo
ve cluster group configuration.

Cluster unit unit-2-1 transitioned from [redacted] to DISABLED
The 3DES/AES algorithms require a Encryption-3DES-AES acti
vation key.
```

```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
WARNING: dynamic rout
n cluster interface-m
red on any management

Cluster unit unit-3-1
```

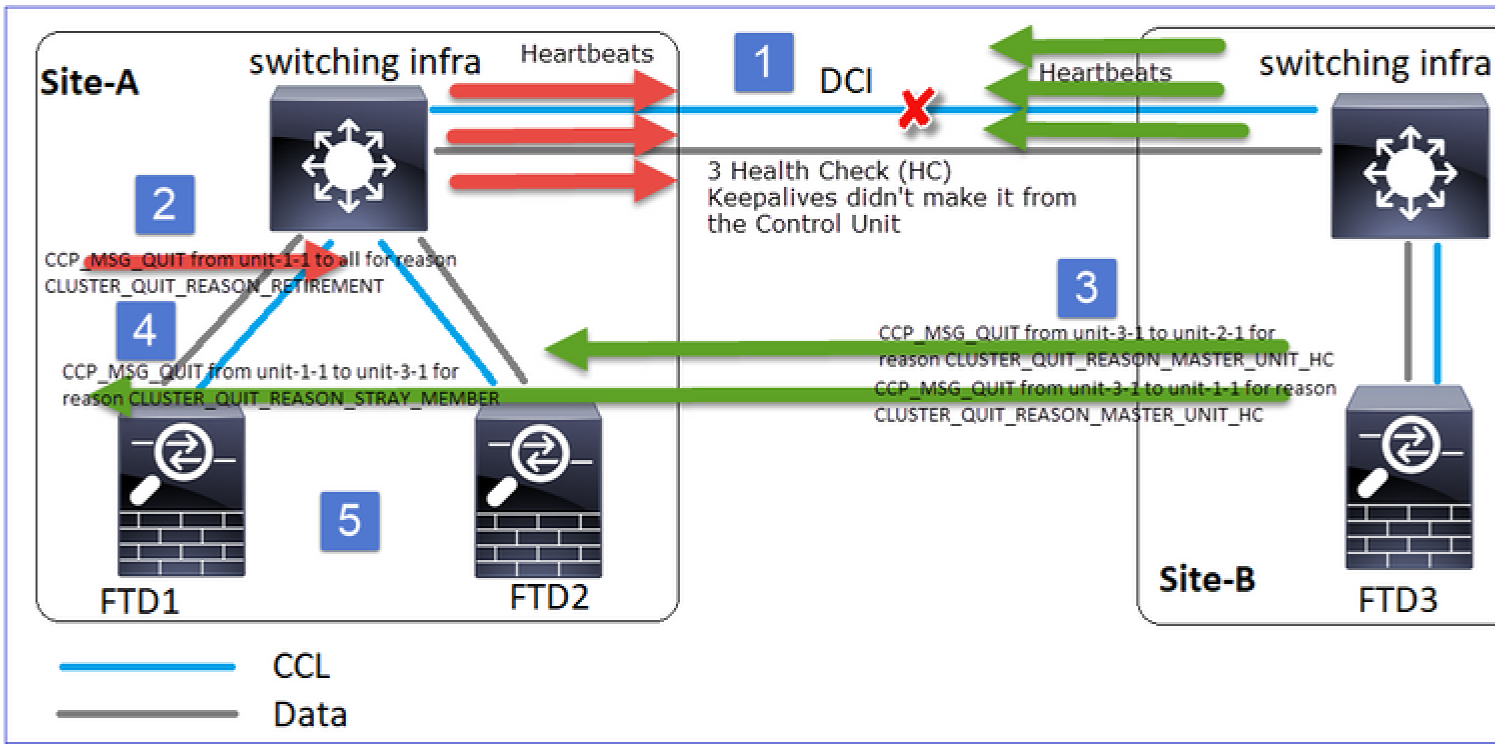
A different flavor of the same failure. In this case, the unit-1-1 also didnâ€™t get 3 HC messages from the unit-3-1, and once it got a new keepalive, tried to kick out the unit-3-1 with the use of a STRAY message, but the message never made it to the unit-3-1:



```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# Asking slave unit unit-3-1 to quit because it
failed unit health-check.
Forcing stray member unit-3-1 to leave the cluster
Forcing stray member unit-3-1 to leave the cluster
Cluster disable is performing cleanup..done.
All data interfaces have been shutdown due to clustering
being disabled. To recover either enable clustering or re
move cluster group configuration.

Cluster unit unit-1-1 transitioned from [redacted] to DISABLE
D
```

```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# Cluster disable is performing cleanup..done.
All data interfaces have been shutdown due to clustering b
eing disabled. To recover either enable clustering or remo
ve cluster group configuration.

Cluster unit unit-2-1 transitioned from [redacted] to DISABLED
The 3DES/AES algorithms require a Encryption-3DES-AES acti
vation key.
```

```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
WARNING: dynamic ro
n cluster interface
red on any manageme

Cluster unit unit-3
```


Site-A
switching infra
Heartbeats
1
DCI
Heartbeats
switching infra
3 Health Check (HC)
Keepalives didn't make it from
the Control Unit
2
CCP_MSG_QUIT from unit-1-1 to all for reason
CLUSTER_QUIT_REASON_RETIREMENT
3
CCP_MSG_QUIT from unit-3-1 to unit-2-1 for
reason CLUSTER_QUIT_REASON_MASTER_UNIT_HC
CCP_MSG_QUIT from unit-3-1 to unit-1-1 for reason
CLUSTER_QUIT_REASON_MASTER_UNIT_HC
4
CCP_MSG_QUIT from unit-1-1 to unit-3-1 for
reason CLUSTER_QUIT_REASON_STRAY_MEMBER
5
FTD1
FTD2
Site-B
FTD3
CCL
Data

1. CCL becomes unidirectional for a few seconds. Unit-3-1 does not receive 3 HC messages from the unit-1-1 and becomes a control node.
2. Unit-2-1 sends a CLUSTER_QUIT_REASON_RETIREMENT message (broadcast).
3. Unit-3-1 sends a QUIT_REASON_PRIMARY_UNIT_HC message to unit-2-1. Unit-2-1 receives it and quits the cluster.
4. Unit-3-1 sends a QUIT_REASON_PRIMARY_UNIT_HC message to unit-1-1. Unit-1-1 receives it and quits the cluster. CCL recovers.
5. Units-1-1 and 2-1 rejoin the cluster as data nodes.

Note: If in step 5 the CCL does not recover, then in site-A the FTD1 becomes the new control node, and after the CCL recovery, it wins the new election.

Syslog messages on unit-1-1:

<#root>

```
firepower#
```

`show log | include 747`

```
Nov 03 2020 23:13:08: %FTD-7-747005: Clustering: State machine notify event CLUSTER_EVENT_MEMBER_STATE (
Nov 03 2020 23:13:09: %FTD-4-747015: Clustering: Forcing stray member unit-3-1 to leave the cluster
Nov 03 2020 23:13:09: %FTD-7-747005: Clustering: State machine notify event CLUSTER_EVENT_MEMBER_STATE (
Nov 03 2020 23:13:10: %FTD-4-747015: Clustering: Forcing stray member unit-3-1 to leave the cluster
Nov 03 2020 23:13:10: %FTD-6-747004: Clustering:
```

`State machine changed from state PRIMARY to DISABLED`

```
Nov 03 2020 23:13:12: %FTD-7-747006: Clustering: State machine is at state DISABLED
Nov 03 2020 23:13:12: %FTD-7-747005: Clustering: State machine notify event CLUSTER_EVENT_MY_STATE (stat
Nov 03 2020 23:13:18: %FTD-6-747004: Clustering: State machine changed from state ELECTION to ONCALL
```

Cluster trace logs on unit-1-1:

<#root>

```
firepower#
```

```
show cluster info trace | include QUIT
```

```
Nov 03 23:13:10.789 [DBUG]Send CCP message to all: CCP_MSG_QUIT from unit-1-1 for reason CLUSTER_QUIT_RE
Nov 03 23:13:10.769 [DBUG]
```

`Receive CCP message: CCP_MSG_QUIT from unit-3-1 to unit-1-1 for reason CLUSTER_QUIT_REASON_PRIMARY_UNIT_`

```
Nov 03 23:13:10.769 [DBUG]Send CCP message to id 1: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CL
Nov 03 23:13:09.789 [DBUG]Receive CCP message: CCP_MSG_QUIT from unit-2-1 for reason CLUSTER_QUIT_REASON
Nov 03 23:13:09.769 [DBUG]Send CCP message to id 1: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CL
Nov 03 23:13:08.559 [DBUG]Send CCP message to all: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLU
Nov 03 23:13:08.559 [DBUG]Send CCP message to id 1: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CL
```

Syslog messages on unit-3-1:

<#root>

```
firepower#
```

```
show log | include 747
```

```
Nov 03 2020 23:13:09: %FTD-7-747005: Clustering: State machine notify event CLUSTER_EVENT_MEMBER_STATE (
Nov 03 2020 23:13:10: %FTD-7-747005: Clustering: State machine notify event CLUSTER_EVENT_MEMBER_STATE (
Nov 03 2020 23:13:10: %FTD-6-747004: Clustering:
```

`State machine changed from state SECONDARY to PRIMARY`

```
Nov 03 2020 23:13:10: %FTD-6-747004: Clustering: State machine changed from state PRIMARY_FAST to PRIMAR
Nov 03 2020 23:13:10: %FTD-6-747004: Clustering: State machine changed from state PRIMARY_DRAIN to PRIMA
Nov 03 2020 23:13:10: %FTD-6-747004: Clustering: State machine changed from state PRIMARY_CONFIG to PRIM
Nov 03 2020 23:13:10: %FTD-7-747006: Clustering: State machine is at state PRIMARY_POST_CONFIG
Nov 03 2020 23:13:10: %FTD-6-747004: Clustering: State machine changed from state PRIMARY_POST_CONFIG to
Nov 03 2020 23:13:10: %FTD-7-747006: Clustering:
```

`State machine is at state PRIMARY`

Cluster history

Unit-1-1	**U**
<#root> `23:13:13 UTC Nov 3 2020`	< 2

```
PRIMARY       DISABLED        Received control message DISABLE
                                (primary unit health check failure)


23:13:18 UTC Nov 3 2020
DISABLED        ELECTION        Enabled from CLI
23:13:18 UTC Nov 3 2020
ELECTION        ONCALL          Received cluster control message
23:13:23 UTC Nov 3 2020
ONCALL          ELECTION        Received cluster control message
â€¦
23:14:48 UTC Nov 3 2020
ONCALL          ELECTION        Received cluster control message
23:14:48 UTC Nov 3 2020
ELECTION        SECONDARY_COLD      Received cluster control message
23:14:48 UTC Nov 3 2020
SECONDARY_COLD      SECONDARY_APP_SYNC  Client progression done
23:15:36 UTC Nov 3 2020
SECONDARY_APP_SYNC  SECONDARY_CONFIG    SECONDARY application configuration
                                sync done
23:15:48 UTC Nov 3 2020
SECONDARY_CONFIG    SECONDARY_FILESYS   Configuration replication finished
23:15:49 UTC Nov 3 2020
SECONDARY_FILESYS   SECONDARY_BULK_SYNC Client progression done
23:16:13 UTC Nov 3 2020
SECONDARY_BULK_SYNC

SECONDARY

        Client progression done
```

Scenario 5

Before the failure

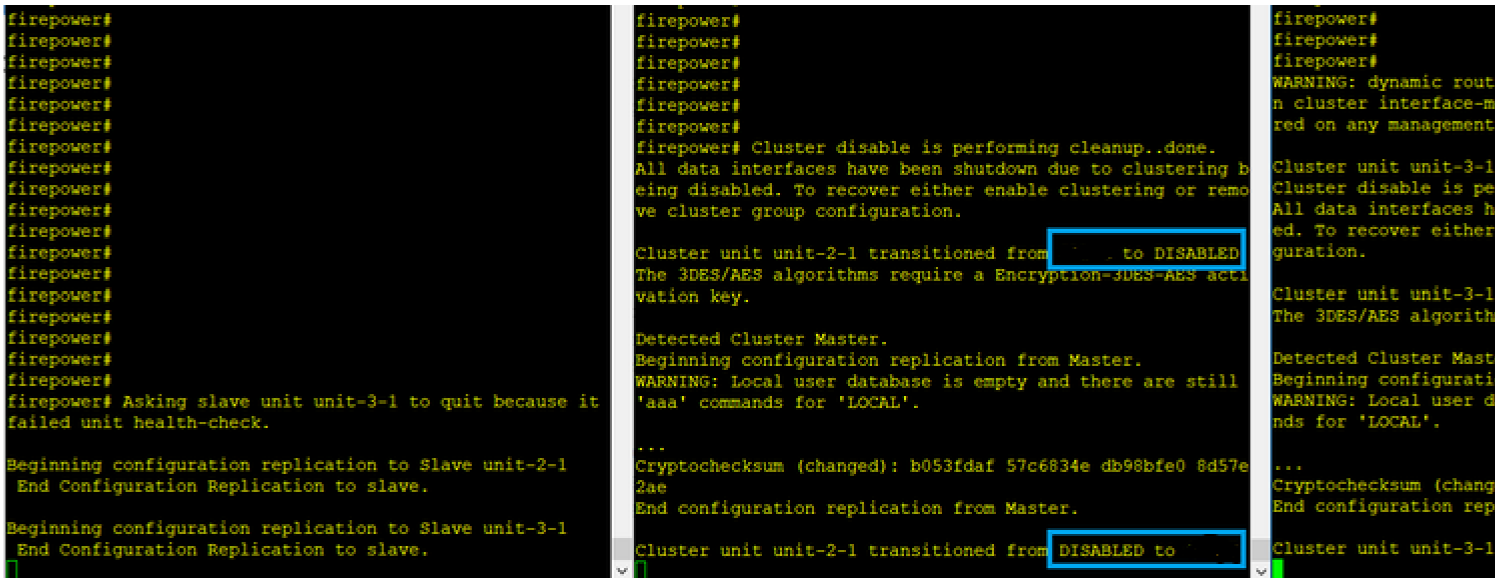
FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B
Control node	Data node	Data node

After the recovery (no changes)

FTD1	**FTD2**	**FTD3**
Site-A	Site-A	Site-B

Control node	Data node	Data node

The failure



```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# Asking slave unit unit-3-1 to quit because it
failed unit health-check.

Beginning configuration replication to Slave unit-2-1
 End Configuration Replication to slave.

Beginning configuration replication to Slave unit-3-1
 End Configuration Replication to slave.
```

```
firepower#
firepower#
firepower#
firepower#
firepower#
firepower#
firepower# Cluster disable is performing cleanup..done.
All data interfaces have been shutdown due to clustering b
eing disabled. To recover either enable clustering or remo
ve cluster group configuration.

Cluster unit unit-2-1 transitioned from [redacted] to DISABLED
The 3DES/AES algorithms require a Encryption-3DES-AES acti
vation key.

Detected Cluster Master.
Beginning configuration replication from Master.
WARNING: Local user database is empty and there are still
'aaa' commands for 'LOCAL'.

...
Cryptochecksum (changed): b053fdaf 57c6834e db98bfe0 8d57e
2ae
End configuration replication from Master.

Cluster unit unit-2-1 transitioned from DISABLED to [redacted]
```

```
firepower#
firepower#
firepower#
WARNING: dynamic rout
n cluster interface-m
red on any management

Cluster unit unit-3-1
Cluster disable is pe
All data interfaces h
ed. To recover either
guration.

Cluster unit unit-3-1
The 3DES/AES algorith

Detected Cluster Mast
Beginning configurati
WARNING: Local user d
nds for 'LOCAL'.

...
Cryptochecksum (chang
End configuration rep

Cluster unit unit-3-1
```

Unit-3-1 sent QUIT messages to both unit-1-1 and unit-2-1, but due to connectivity issues only unit-2-1 received the QUIT message.

Unit-1-1 cluster trace logs:

<#root>

```
firepower#
```

`show cluster info trace | include QUIT`

```
Nov 04 00:52:10.429 [DBUG]Receive CCP message: CCP_MSG_QUIT from unit-3-1 for reason CLUSTER_QUIT_REASON
Nov 04 00:51:47.059 [DBUG]Receive CCP message: CCP_MSG_QUIT from unit-2-1 for reason CLUSTER_QUIT_REASON
Nov 04 00:51:45.429 [DBUG]Send CCP message to all: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLU
Nov 04 00:51:45.429 [DBUG]Send CCP message to unit-3-1(1): CCP_MSG_QUIT from unit-1-1 to unit-3-1 for re
```

Unit-2-1 cluster trace logs:

<#root>

```
firepower#
```

`show cluster info trace | include QUIT`

```
Nov 04 00:52:10.389 [DBUG]Receive CCP message: CCP_MSG_QUIT from unit-3-1 for reason CLUSTER_QUIT_REASON
Nov 04 00:51:47.019 [DBUG]Send CCP message to all: CCP_MSG_QUIT from unit-2-1 for reason CLUSTER_QUIT_RE
Nov 04 00:51:46.999 [DBUG]
```

`Receive CCP message: CCP_MSG_QUIT from unit-3-1 to unit-2-1 for reason CLUSTER_QUIT_REASON_PRIMARY_UNIT_`

```
Nov 04 00:51:45.389 [DBUG]Receive CCP message: CCP_MSG_QUIT from unit-1-1 to unit-3-1 for reason CLUSTER
```

Cluster history

Unit-1-1	Unit-2-1

No events	<#root>
	`00:51:50 UTC Nov 4 2020`
	`SECONDARY           DISABLED        Received control message DISABLE`
	`                                  (primary unit health check failure)`
	`00:51:54 UTC Nov 4 2020`
	`DISABLED        ELECTION        Enabled from CLI`
	`00:51:54 UTC Nov 4 2020`
	`ELECTION        SECONDARY_COLD      Received cluster control message`
	`00:51:54 UTC Nov 4 2020`
	`SECONDARY_COLD      SECONDARY_APP_SYNC   Client progression done`
	`00:52:42 UTC Nov 4 2020`
	`SECONDARY_APP_SYNC   SECONDARY_CONFIG     SECONDARY application configura`
	`                                  sync done`
	`00:52:54 UTC Nov 4 2020`
	`SECONDARY_CONFIG     SECONDARY_FILESYS    Configuration replication finis`
	`00:52:55 UTC Nov 4 2020`
	`SECONDARY_FILESYS    SECONDARY_BULK_SYNC  Client progression done`
	`00:53:19 UTC Nov 4 2020`
	`SECONDARY_BULK_SYNC`
	`SECONDARY`
	`         Client progression done`

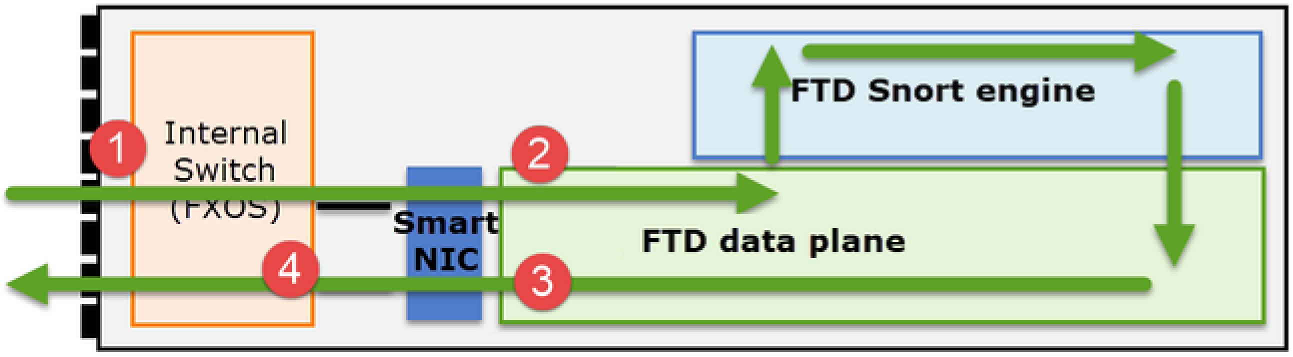
Cluster Data Plane Connection Establishment

NGFW Capture Points

The NGFW provides capture capabilities on these points:

- Chassis internal switch (FXOS)
- FTD data plane engine
- FTD Snort engine

When you troubleshoot data-path issues on a cluster, the capture points used in most cases are the FXOS and FTD data plane engine captures.



1. FXOS ingress capture on the physical interface
2. FTD ingress capture in data plane engine
3. FTD egress capture in data plane engine
4. FXOS ingress capture on backplane interface

For additional details about NGFW captures check this document:

Cluster Unit Flow Roles Basics

Connections can be established through a cluster in multiple ways which depend on factors like:

- Type of traffic (TCP, UDP, etc)
- Load balancing algorithm configured on the adjacent switch
- Features configured on the firewall
- Network conditions (for example, IP fragmentation, network delays, and so on)

Flow role	**Description**	**Flag(s)**
Owner	Usually, the unit that initially receives the connection	UIO
Director	The unit that handles owner lookup requests from forwarders.	Y
Backup owner	As long as the director is not the same unit as the owner, then the director is also the backup owner. If the owner chooses itself as the director, then a separate backup owner is chosen.	Y (if the director is also the backup owner) y (if the director is not the backup owner)
Forwarder	A unit that forwards packets to the owner	z
Fragment owner	The unit that handles the fragmented	-

	traffic	
Chassis backup	In an inter-chassis cluster when both director/backup and owner flows are owned by the units of the same chassis, a unit in one of the other chassis becomes a secondary backup/director. This role is specific to inter-chassis clusters of the Firepower 9300 series with more than 1 blade.	w

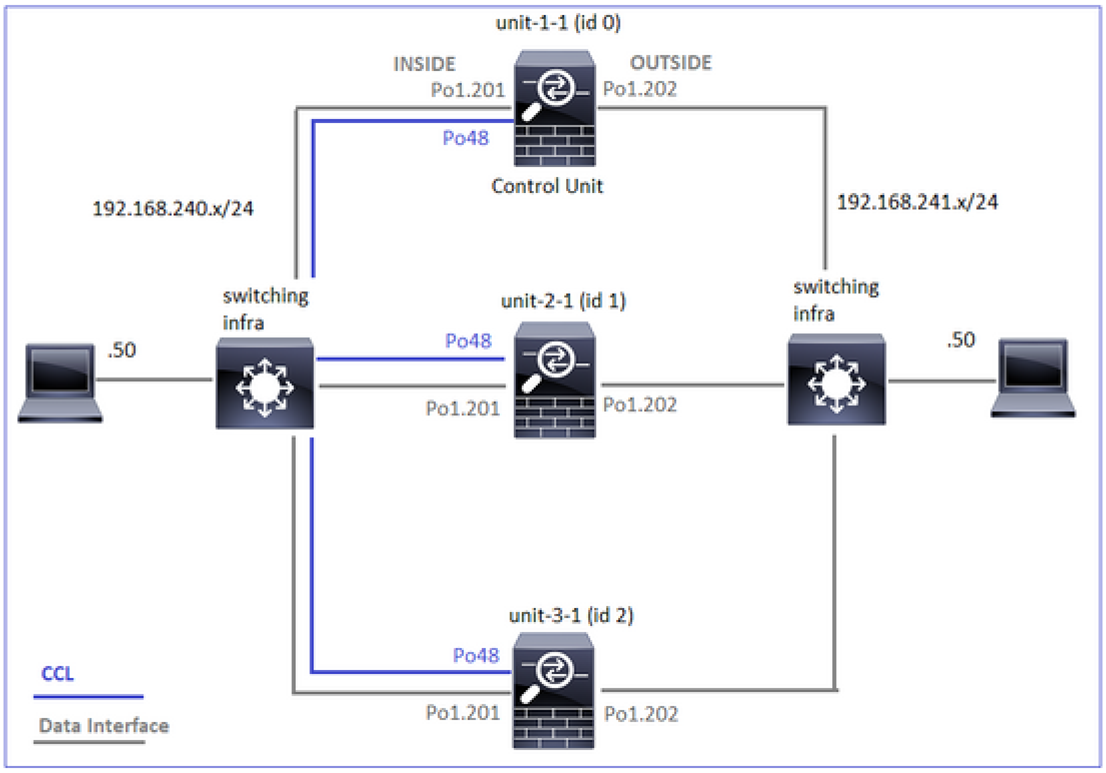
- For additional details check the related section in the Configuration Guide (see links in the Related Information)
- In specific scenarios (see case studies section) some flag(s) are not always shown..

Cluster Connection Establishment Case Studies

The next section covers various case studies that demonstrate some of the ways a connection can be established through a cluster. The goals are to:

- Familiarize you with the different unit roles.
- Demonstrate how the various command outputs can be correlated.

Topology



Cluster units and IDs:

Unit-1-1	**Unit-2-1**

<#root>
 	
`Cluster GROUP1: On`
`    Interface mode: spanned`
 	
`This is "unit-1-1" in state PRIMARY`
 	
`ID        : 0`
 	
`Site ID   : 1`
`Version   : 9.15(1)`
`Serial No.: FCH22247LNK`
`CCL IP    : 10.17.1.1`
`CCL MAC   : 0015.c500.018f`
`Last join : 02:24:43 UTC Nov 27 2020`
`Last leave: N/A`
	<#root>
	
	`Unit "unit-2-1" in state SECO`
	
	`ID        : 1`
	
	`Site ID   : 1`
	`Version   : 9.15(1)`
	`Serial No.: FCH23157Y9N`
	`CCL IP    : 10.17.2.1`
	`CCL MAC   : 0015.c500.028`
	`Last join : 02:04:19 UTC`
	`Last leave: N/A`

Cluster captures enabled:

```
cluster exec cap CAPI int INSIDE buffer 33554432 match tcp host 192.168.240.50 host 192.168.241.50 eq 80
cluster exec cap CAPO int OUTSIDE buffer 33554432 match tcp host 192.168.240.50 host 192.168.241.50 eq 8
cluster exec cap CAPI_RH reinject-hide int INSIDE buffer 33554432 match tcp host 192.168.240.50 host 192
cluster exec cap CAPO_RH reinject-hide int OUTSIDE buffer 33554432 match tcp host 192.168.240.50 host 19
cluster exec cap CCL int cluster buffer 33554432
```

Note: These tests were run in a lab environment with minimal traffic through the cluster. In production try to use as specific capture filters as possible (for example, destination port and whenever possible the source port) to minimize the â€˜noiseâ€™ in the captures.

Case Study 1. Symmetric Traffic (owner is also the director)

Observation 1. The reinject-hide captures show packets only on unit-1-1. This means that the flow in both directions went through unit-1-1 (symmetric traffic):

<#root>

```
firepower#
```

`cluster exec show cap`

```
unit-1-1(LOCAL):******************************************************
capture CCL type raw-data interface cluster [Capturing - 33513 bytes]
capture CAPI type raw-data buffer 33554432 trace interface INSIDE [Buffer Full - 33553914 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPO type raw-data buffer 33554432 trace interface OUTSIDE [Buffer Full - 33553914 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPI_RH type raw-data
```

reinject-hide

```
 buffer 33554432 interface INSIDE [Buffer Full -
```

33553914 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 33554432 interface OUTSIDE [Buffer Full -
```

33553914 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80

unit-2-1:*************************************************************
capture CCL type raw-data interface cluster [Capturing - 23245 bytes]
capture CAPI type raw-data buffer 33554432 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPO type raw-data buffer 33554432 trace interface OUTSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPI_RH type raw-data
```

reinject-hide

```
 buffer 33554432 interface INSIDE [Capturing -
```

0 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 33554432 interface OUTSIDE [Capturing -
```

0 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80

unit-3-1:*************************************************************
capture CCL type raw-data interface cluster [Capturing - 24815 bytes]
capture CAPI type raw-data buffer 33554432 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPO type raw-data buffer 33554432 trace interface OUTSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPI_RH type raw-data
```

reinject-hide

```
 buffer 33554432 interface INSIDE [Capturing -
```

0 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 33554432 interface OUTSIDE [Capturing -
```

```
0 bytes

]
match tcp host 192.168.240.50 host 192.168.241.50 eq 80
```

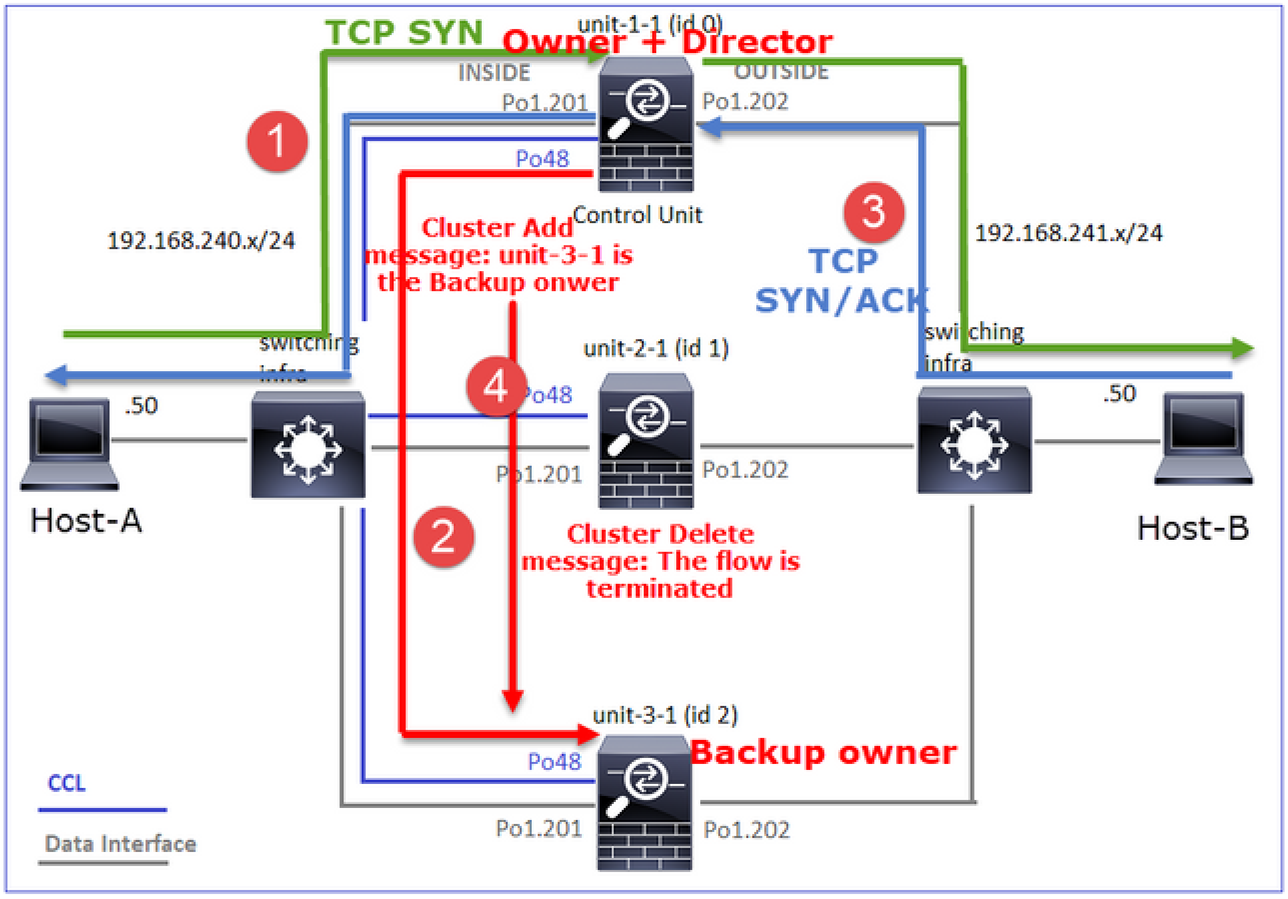
Observation 2. Connection flag analysis for flow with source port 45954

<#root>

```
firepower#

cluster exec show conn


unit-1-1(LOCAL):******************************************************
22 in use, 25 most used
Cluster:
fwd connections: 0 in use, 1 most used
dir connections: 0 in use, 122 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 1 enabled, 0 in effect, 2 most enabled, 1 most in effect

TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:

45954

, idle 0:00:00, bytes 487413076,

flags UIO N1



unit-2-1:*************************************************************
22 in use, 271 most used
Cluster:
fwd connections: 0 in use, 2 most used
dir connections: 0 in use, 2 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 1 enabled, 0 in effect, 249 most enabled, 0 most in effect

unit-3-1:*************************************************************
17 in use, 20 most used
Cluster:
fwd connections: 1 in use, 2 most used
dir connections: 1 in use, 127 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 1 most enabled, 0 most in effect

TCP OUTSIDE 192.168.241.50:443 NP Identity Ifc 192.168.240.50:39698, idle 0:00:23, bytes 0, flags z
TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:

45954

, idle 0:00:06, bytes 0,

flags y
```

Unit	**Flag**	**Note**
Unit-1-1	UIO	· Flow Owner â€“ The unit handles the flow · Director â€“ Since unit-3-1 has â€˜yâ€™ and not â€˜Yâ€™ this implies that unit-1-1 was chosen as the director for this flow. Thus, since it is also the owner, another unit (unit-3-1 in this case) was elected as the backup owner
Unit-2-1	-	-
Unit-3-1	y	The unit is a Backup owner

This can be visualized as:



1. TCP SYN packet arrives from Host-A to unit-1-1. Unit-1-1 becomes the flow owner.
2. Unit-1-1 is also elected the flow director. Thus, it also elects unit-3-1 as the backup owner (cluster add message).
3. TCP SYN/ACK packet arrives from Host-B to unit-3-1. The flow is symmetric.
4. Once the connection is terminated, the owner sends a cluster delete message to remove the flow info from the backup owner.

Observation 3. Capture with trace shows that both directions go only through unit-1-1.

Step 1. Identify the flow and packets of interest in all cluster units based on the source port:

<#root>

```
firepower#
```

cluster exec show capture CAPI | i 45954

```
unit-1-1(LOCAL):******************************************************
1: 08:42:09.362697 802.1Q vlan#201 P0 192.168.240.50.45954 > 192.168.241.50.80: S 992089269:992089269(0)
2: 08:42:09.363521 802.1Q vlan#201 P0 192.168.241.50.80 > 192.168.240.50.45954: S 4042762409:4042762409(
3: 08:42:09.363827 802.1Q vlan#201 P0 192.168.240.50.45954 > 192.168.241.50.80: . ack 4042762410 win 229
â€¦
unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

<#root>

```
firepower#
```

cluster exec show capture CAPO | i 45954

```
unit-1-1(LOCAL):******************************************************
1: 08:42:09.362987 802.1Q vlan#202 P0 192.168.240.50.45954 > 192.168.241.50.80: S 2732339016:2732339016(
2: 08:42:09.363415 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.45954: S 3603655982:3603655982(
3: 08:42:09.363903 802.1Q vlan#202 P0 192.168.240.50.45954 > 192.168.241.50.80: . ack 3603655983 win 229
â€¦
unit-2-1:*************************************************************

unit-3-1:*************************************************************
```

Step 2. Since this is a TCP flow trace the 3-way handshake packets. As it can be seen in this output, unit-1-1 is the owner. For simplicity, the non-relevant trace phases are omitted:

<#root>

```
firepower#
```

show cap CAPI packet-number 1 trace

```
25985 packets captured
1: 08:42:09.362697 802.1Q vlan#201 P0 192.168.240.50.
```

45954

```
 > 192.168.241.50.80:
```

S

```
 992089269:992089269(0) win 29200 <mss 1460,sackOK,timestamp 495153655 0,nop,wscale 7>
...
```

```
Phase: 4
```

Type: CLUSTER-EVENT

```
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
```

Flow type: NO FLOW

I (0) got initial, attempting ownership.

```
Phase: 5
```

Type: CLUSTER-EVENT

```
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

I (0) am becoming owner

```
...
```

The return traffic (TCP SYN/ACK):

<#root>

```
firepower#
```

show capture CAPO packet-number 2 trace

```
25985 packets captured
2: 08:42:09.363415 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.45954:
```

S

```
 3603655982:3603655982(0)
```

ack

```
 2732339017 win 28960 <mss 1460,sackOK,timestamp 505509125 495153655,nop,wscale 7>
...
Phase: 3
```

```
Type: FLOW-LOOKUP

Subtype:
Result: ALLOW
Config:
Additional Information:

Found flow with id 9364, using existing flow
```

Observation 4. FTD data plane syslogs show the connection creation and termination on all units:

<#root>

```
firepower#

cluster exec show log | include 45954



unit-1-1

(LOCAL):******************************************************
Dec 01 2020 08:42:09: %FTD-6-302013:

Built inbound TCP connection 9364

 for INSIDE:192.168.240.50/45954 (192.168.240.50/45954) to OUTSIDE:192.168.241.50/80 (192.168.241.50/80)
Dec 01 2020 08:42:18: %FTD-6-302014:

Teardown TCP connection 9364

 for INSIDE:192.168.240.50/45954 to OUTSIDE:192.168.241.50/80 duration 0:00:08 bytes 1024000440 TCP FINs

unit-2-1:*************************************************************


unit-3-1

:************************************************************
Dec 01 2020 08:42:09: %FTD-6-302022:

Built backup stub TCP connection

 for INSIDE:192.168.240.50/45954 (192.168.240.50/45954) to OUTSIDE:192.168.241.50/80 (192.168.241.50/80)
Dec 01 2020 08:42:18: %FTD-6-302023:

Teardown backup TCP connection

 for INSIDE:192.168.240.50/45954 to OUTSIDE:192.168.241.50/80 duration 0:00:08 forwarded bytes 0 Cluster
```

Case Study 2. Symmetric Traffic (owner different than the director)

- Same as case study #1, but in this case study, a flow owner is a different unit than the director.
- All the outputs are similar to case study #1. The main difference compared to case study #1 is the â€˜Yâ€™ flag which substitutes the â€˜yâ€™ flag of scenario 1.

Observation 1. The owner is different than the director.

Connection flag analysis for flow with source port 46278.

<#root>

```
firepower#
```

```
cluster exec show conn
```

```
unit-1-1(LOCAL):******************************************************
23 in use, 25 most used
Cluster:
fwd connections: 0 in use, 1 most used
dir connections: 0 in use, 122 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 2 enabled, 0 in effect, 4 most enabled, 1 most in effect

TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:
```

46278

```
, idle 0:00:00, bytes 508848268, flags
```

UIO N1

```
TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:46276, idle 0:00:03, bytes 0, flags aA N1

unit-2-1:*************************************************************
21 in use, 271 most used
Cluster:
fwd connections: 0 in use, 2 most used
dir connections: 0 in use, 2 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 249 most enabled, 0 most in effect

unit-3-1:*************************************************************
17 in use, 20 most used
Cluster:
fwd connections: 1 in use, 5 most used
dir connections: 1 in use, 127 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 1 most enabled, 0 most in effect

TCP OUTSIDE 192.168.241.50:80 NP Identity Ifc 192.168.240.50:46276, idle 0:00:02, bytes 0, flags z
TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:
```

46278

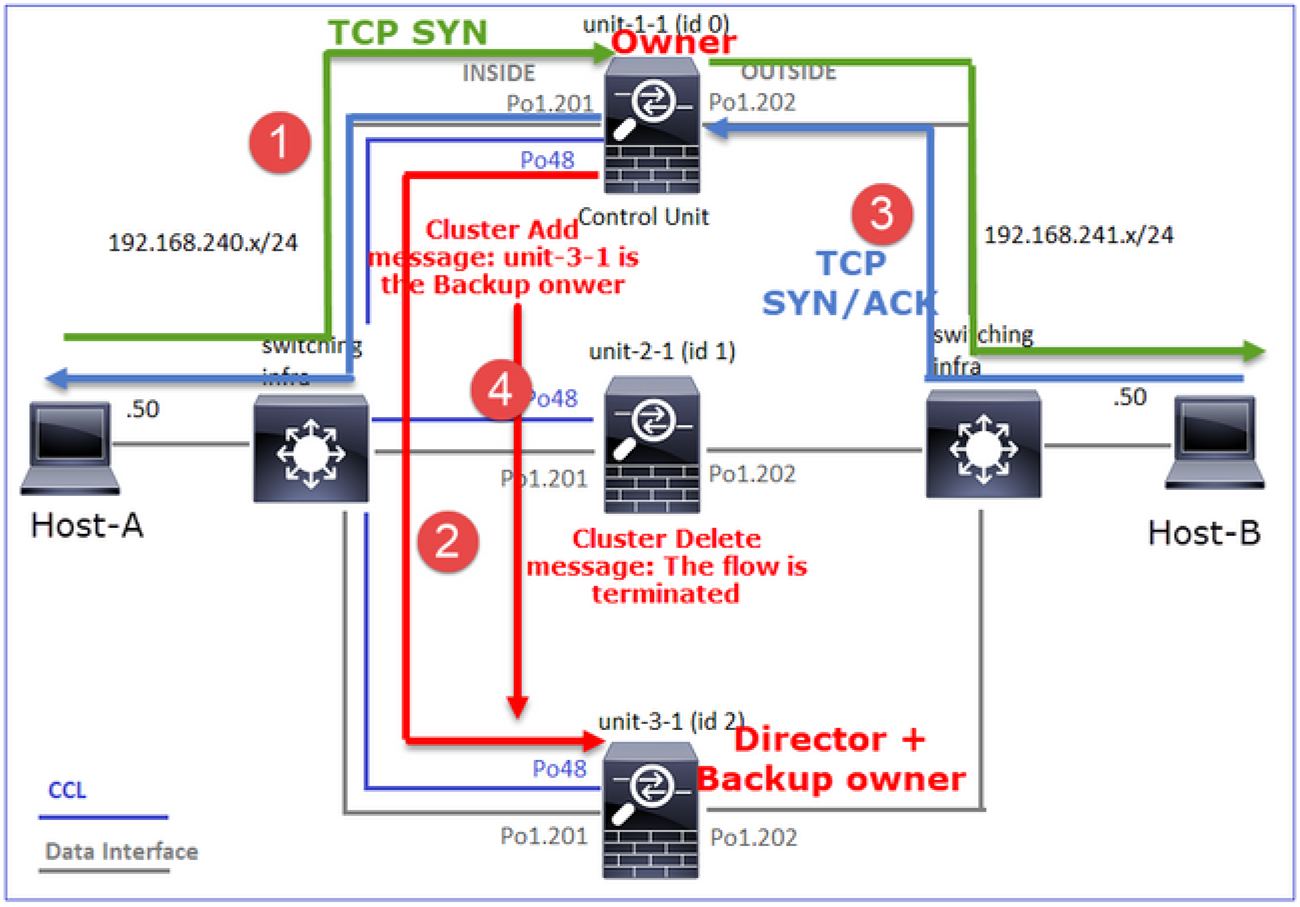
```
, idle 0:00:06, bytes 0,
```

flags Y

Unit	Flag	Note

Unit-1-1	UIO	· Flow Owner â€“ The unit handles the flow
Unit-2-1	-	-
Unit-3-1	Y	· Director and Backup owner â€“ Unit 3-1 has the flag Y (Director).

This can be visualized as:



1. TCP SYN packet arrives from Host-A to unit-1-1. Unit-1-1 becomes the flow owner.
2. Unit-3-1 is elected the flow director. Unit-3-1 is also the backup owner ('cluster add' message on UDP 4193 over the CCL).
3. TCP SYN/ACK packet arrives from Host-B to unit-3-1. The flow is symmetric.
4. Once the connection is terminated, the owner sends over the CCL a 'cluster delete' message on UDP 4193 to remove the flow info from the backup owner.

Observation 2. Capture with trace shows that both directions go only through unit-1-1

Step 1. Use the same approach as in case study 1 to identify the flow and packets of interest in all cluster units based on the source port:

<#root>

```
firepower#

cluster exec show cap CAPI | include 46278
```

```
unit-1-1

(LOCAL):******************************************************
3: 11:01:44.841631 802.1Q vlan#201 P0 192.168.240.50.46278 > 192.168.241.50.80:

S

 1972783998:1972783998(0) win 29200 <mss 1460,sackOK,timestamp 503529072 0,nop,wscale 7>
4: 11:01:44.842317 802.1Q vlan#201 P0 192.168.241.50.80 > 192.168.240.50.46278:

S

 3524167695:3524167695(0)

ack

 1972783999 win 28960 <mss 1380,sackOK,timestamp 513884542 503529072,nop,wscale 7>
5: 11:01:44.842592 802.1Q vlan#201 P0 192.168.240.50.46278 > 192.168.241.50.80: . ack 3524167696 win 229
â€¦
unit-2-1:*************************************************************

unit-3-1:*************************************************************
firepower#
```

Capture on the OUTSIDE interface:

<#root>

```
firepower#

cluster exec show cap CAPO | include 46278


unit-1-1

(LOCAL):******************************************************
3: 11:01:44.841921 802.1Q vlan#202 P0 192.168.240.50.46278 > 192.168.241.50.80:

S

 2153055699:2153055699(0) win 29200 <mss 1380,sackOK,timestamp 503529072 0,nop,wscale 7>
4: 11:01:44.842226 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.46278:

S

 3382481337:3382481337(0)

ack

 2153055700 win 28960 <mss 1460,sackOK,timestamp 513884542 503529072,nop,wscale 7>
5: 11:01:44.842638 802.1Q vlan#202 P0 192.168.240.50.46278 > 192.168.241.50.80: . ack 3382481338 win 229

unit-2-1:*************************************************************

unit-3-1:*************************************************************
firepower#
```

Step 2. Focus on the ingress packets (TCP SYN and TCP SYN/ACK):

<#root>

```
firepower#
```

`cluster exec show cap CAPI packet-number 3 trace`

```
unit-1-1(LOCAL):******************************************************

824 packets captured

3: 11:01:44.841631 802.1Q vlan#201 P0 192.168.240.50.46278 > 192.168.241.50.80:
```

`S`

```
 1972783998:1972783998(0) win 29200 <mss 1460,sackOK,timestamp 503529072 0,nop,wscale 7>
â€¦

Phase: 4
```

`Type: CLUSTER-EVENT`

```
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

`I (0) got initial, attempting ownership.`

```
Phase: 5
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

`I (0) am becoming owner`

Trace the SYN/ACK on unit-1-1:

<#root>

```
firepower#
```

`cluster exec show cap CAPO packet-number 4 trace`

```
unit-1-1(LOCAL):******************************************************

4: 11:01:44.842226 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.
```

46278

```
:
```

S

```
 3382481337:3382481337(0)
```

ack

```
 2153055700 win 28960 <mss 1460,sackOK,timestamp 513884542 503529072,nop,wscale 7>
Phase: 3
Type: FLOW-LOOKUP
Subtype:
Result: ALLOW
Config:
Additional Information:
```

Found flow with id 9583, using existing flow

Observation 3. FTD data plane syslogs show the connection creation and termination on owner and backup owner:

<#root>

```
firepower#
```

cluster exec show log | include 46278

```
unit-1-1(LOCAL):******************************************************
Dec 01 2020 11:01:44: %FTD-6-302013:
```

Built inbound TCP connection

```
 9583 for INSIDE:192.168.240.50/46278 (192.168.240.50/46278) to OUTSIDE:192.168.241.50/80 (192.168.241.5
Dec 01 2020 11:01:53: %FTD-6-302014:
```

Teardown TCP connection

```
 9583 for INSIDE:192.168.240.50/46278 to OUTSIDE:192.168.241.50/80 duration 0:00:08 bytes 1024001808 TCP

unit-2-1:*************************************************************

unit-3-1:*************************************************************
Dec 01 2020 11:01:44: %FTD-6-302022:
```

Built director stub TCP connection

```
 for INSIDE:192.168.240.50/46278 (192.168.240.50/46278) to OUTSIDE:192.168.241.50/80 (192.168.241.50/80)
Dec 01 2020 11:01:53: %FTD-6-302023:
```

Teardown director TCP connection

```
 for INSIDE:192.168.240.50/46278 to OUTSIDE:192.168.241.50/80 duration 0:00:08 forwarded bytes 0 Cluster
```

Case Study 3. Asymmetric Traffic (director forwards the traffic).

Observation 1. The reinject-hide captures show packets on unit-1-1 and unit-2-1 (asymmetric flow):

<#root>

```
firepower#
```

```
cluster exec show cap
```

```
unit-1-1(LOCAL):******************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33554320 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Buffer Full - 98552 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Buffer Full - 98552 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface
```

INSIDE

```
 [Buffer Full -
```

98552 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface
```

OUTSIDE

```
 [Buffer Full -
```

99932 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www

unit-2-1:************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33553268 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Buffer Full - 99052 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data reinject-hide buffer 100000 interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface
```

OUTSIDE

```
 [Buffer Full -
```

99052 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www

unit-3-1:*************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Capturing - 53815 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Capturing - 658 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data reinject-hide buffer 100000 interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data reinject-hide buffer 100000 interface OUTSIDE [Capturing - 658 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
```

Observation 2. Connection flag analysis for flow with source port 46502.

<#root>

```
firepower#
```

cluster exec show conn

unit-1-1

```
(LOCAL):*****************************************************
23 in use, 25 most used
Cluster:
fwd connections: 0 in use, 1 most used
dir connections: 0 in use, 122 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 2 enabled, 0 in effect, 4 most enabled, 1 most in effect

TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:
```

46502

```
, idle 0:00:00, bytes 448760236,
```

flags UIO N1

```
TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:46500, idle 0:00:06, bytes 0, flags aA N1
```

unit-2-1

```
:************************************************************
21 in use, 271 most used
Cluster:
fwd connections: 0 in use, 2 most used
dir connections: 1 in use, 2 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 249 most enabled, 0 most in effect
```

```
TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:
```

46502

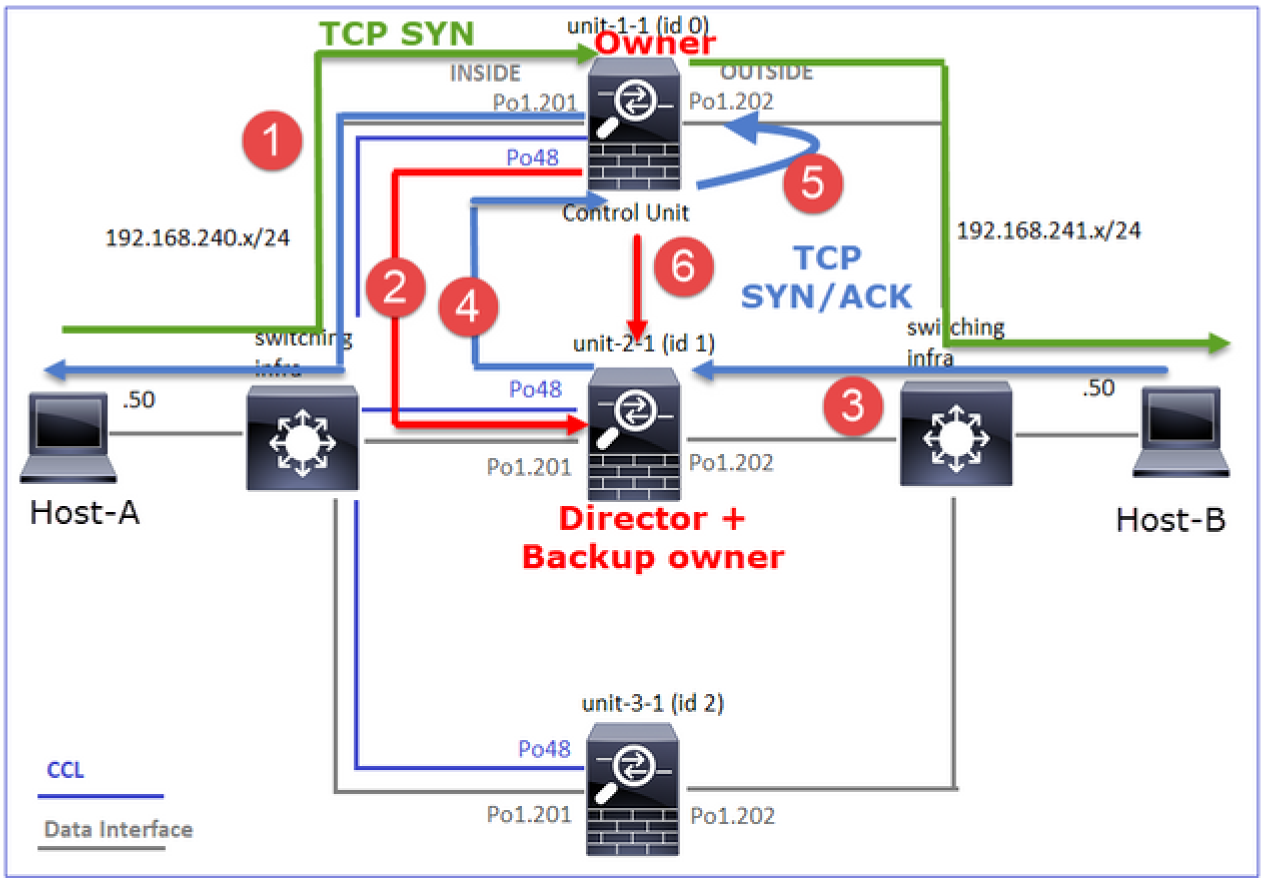
```
, idle 0:00:00, bytes 0,
```

flags Y

```
unit-3-1:*************************************************************
17 in use, 20 most used
Cluster:
fwd connections: 1 in use, 5 most used
dir connections: 0 in use, 127 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 1 most enabled, 0 most in effect
```

Unit	**Flag**	**Note**
Unit-1-1	UIO	· Flow Owner â€“ The unit handles the flow.
Unit-2-1	Y	· Director â€“ Since unit-2-1 has flag â€˜Yâ€™ this implies that unit-2-1 was chosen as the director for this flow. · Backup owner · Finally, although it is not obvious from this output, from the show capture and show log outputs it is evident that unit-2-1 forwards this flow to the owner (although technically it is not considered a forwarder in this scenario). Note: A unit cannot be both director (Y flow) and forwarder (z flow), these 2 roles are mutually exclusive. Directors (Y flow) can still forward traffic. See the show log output later in this case study.
Unit-3-1	-	-

This can be visualized as:



1. TCP SYN packet arrives from Host-A to unit-1-1. Unit-1-1 becomes the flow owner.
2. Unit-2-1 is elected the flow director and backup owner. The flow owner sends a 'cluster add' unicast message on UDP 4193 to inform the backup owner about the flow.
3. TCP SYN/ACK packet arrives from Host-B to unit-2-1. The flow is asymmetric.
4. Unit-2-1 forwards the packet through the CCL to the owner (due to TCP SYN Cookie).
5. The owner reinjects the packet on the interface OUTSIDE and then forwards the packet towards Host-A.
6. Once the connection is terminated, the owner sends a cluster delete message to remove the flow info from the backup owner.

Observation 3. Capture with trace shows the asymmetric traffic and the redirection from unit-2-1 to unit-1-1.

Step 1. Identify the packets that belong to the flow of interest (port 46502):

<#root>

```
firepower#
```

```
cluster exec show capture CAPI | include 46502
```

```
unit-1-1(LOCAL):******************************************************
3: 12:58:33.356121 802.1Q vlan#201 P0 192.168.240.50.46502 > 192.168.241.50.80: S 4124514680:4124514680(
4: 12:58:33.357037 802.1Q vlan#201 P0 192.168.241.50.80 > 192.168.240.50.46502: S 883000451:883000451(0)
5: 12:58:33.357357 802.1Q vlan#201 P0 192.168.240.50.46502 > 192.168.241.50.80: . ack 883000452 win 229
unit-2-1:************************************************************

unit-3-1:************************************************************
```

The return direction:

<#root>

```
firepower#
```

`cluster exec show capture CAPO | include 46502`

```
unit-1-1(LOCAL):******************************************************
3: 12:58:33.356426 802.1Q vlan#202 P0 192.168.240.50.46502 > 192.168.241.50.80: S 1434968587:1434968587(
4: 12:58:33.356915 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.46502: S 4257314722:4257314722(
5: 12:58:33.357403 802.1Q vlan#202 P0 192.168.240.50.46502 > 192.168.241.50.80: . ack 4257314723 win 229

unit-2-1:*************************************************************
1: 12:58:33.359249 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.46502: S 4257314722:4257314722(
2: 12:58:33.360302 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.46502: . ack 1434968736 win 235
3: 12:58:33.361004 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.46502: . 4257314723:4257316091(
â€¦

unit-3-1:*************************************************************
```

Step 2. Trace the packets. By default, only the first 50 ingress packets are traced. For simplicity, the non-relevant trace phases are omitted.

Unit-1-1 (owner):

<#root>

```
firepower#
```

`cluster exec show capture CAPI packet-number 3 trace`

```
unit-1-1(LOCAL):******************************************************
3: 12:58:33.356121 802.1Q vlan#201 P0 192.168.240.50.
```

`46502`

```
 > 192.168.241.50.80:
```

`S`

```
 4124514680:4124514680(0) win 29200 <mss 1460,sackOK,timestamp 510537534 0,nop,wscale 7>
...
Phase: 4
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

`I (0) got initial, attempting ownership.`

```
Phase: 5
```

```
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

`I (0) am becoming owner`

Unit-2-1 (forwarder)

The return traffic (TCP SYN/ACK). The unit of interest is unit-2-1 which is the director/backup owner and forwards the traffic to the owner:

<#root>

```
firepower#
```

`cluster exec unit unit-2-1 show capture CAPO packet-number 1 trace`

```
1: 12:58:33.359249 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.
```

`46502`

```
: S 4257314722:4257314722(0) ack 1434968588 win 28960 <mss 1460,sackOK,timestamp 520893004 510537534,nop
...
Phase: 4
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW
```

`I (1) got initial, attempting ownership.`

```
Phase: 5
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW
```

`I (1) am early redirecting to (0) due to matching action (-1).`

Observation 4. FTD data plane syslogs show the connection creation and termination on all units:

<#root>

```
firepower#
```

cluster exec show log | i 46502

```
unit-1-1(LOCAL):******************************************************
Dec 01 2020 12:58:33: %FTD-6-302013:
```

B

uilt inbound TCP connection

```
 9742 for INSIDE:192.168.240.50/46502 (192.168.240.50/46502) to OUTSIDE:192.168.241.50/80 (192.168.241.5
Dec 01 2020 12:59:02: %FTD-6-302014:
```

Teardown TCP connection

```
 9742 for INSIDE:192.168.240.50/46502 to OUTSIDE:192.168.241.50/80 duration 0:00:28 bytes 2048000440 TCP

unit-2-1:*************************************************************
Dec 01 2020 12:58:33: %FTD-6-302022:
```

Built forwarder stub TCP connection

```
 for OUTSIDE:192.168.241.50/80 (192.168.241.50/80) to unknown:192.168.240.50/46502 (192.168.240.50/46502
Dec 01 2020 12:58:33: %FTD-6-302023:
```

Teardown forwarder TCP connection

```
 for OUTSIDE:192.168.241.50/80 to unknown:192.168.240.50/46502 duration 0:00:00 forwarded bytes 0 Forwar
Dec 01 2020 12:58:33: %FTD-6-302022:
```

Built director stub TCP connection

```
 for INSIDE:192.168.240.50/46502 (192.168.240.50/46502) to OUTSIDE:192.168.241.50/80 (192.168.241.50/80)
Dec 01 2020 12:59:02: %FTD-6-302023:
```

Teardown director TCP connection

```
 for INSIDE:192.168.240.50/46502 to OUTSIDE:192.168.241.50/80 duration 0:00:28 forwarded bytes 204831630

unit-3-1:*************************************************************
firepower#
```

Case Study 4. Asymmetric Traffic (owner is the director)

Observation 1. The reinject-hide captures show packets on unit-1-1 and unit-2-1 (asymmetric flow):

<#root>

```
firepower#
```

cluster exec show cap

```
unit-1-1(LOCAL):******************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33554229 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Buffer Full - 98974 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Buffer Full - 98974 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface
```

INSIDE

```
 [Buffer Full -
```

98974 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface
```

OUTSIDE

```
 [Buffer Full -
```

99924 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www

unit-2-1:************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33552925 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Buffer Full - 99052 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data reinject-hide buffer 100000 interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface OUTSIDE [Buffer Full -
```

99052 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www

unit-3-1:************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Capturing - 227690 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Capturing - 4754 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data reinject-hide buffer 100000 interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data reinject-hide buffer 100000 interface OUTSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
```

Observation 2. Connection flag analysis for flow with source port 46916.

<#root>

```
firepower#
```

cluster exec show conn

unit-1-1

```
(LOCAL):******************************************************
23 in use, 25 most used
Cluster:
fwd connections: 0 in use, 1 most used
dir connections: 0 in use, 122 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 1 enabled, 0 in effect, 4 most enabled, 1 most in effect

TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:
```

46916

```
, idle 0:00:00, bytes 414682616,
```

flags UIO N1

unit-2-1

```
:*************************************************************
21 in use, 271 most used
Cluster:
fwd connections: 1 in use, 2 most used
dir connections: 0 in use, 2 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 249 most enabled, 0 most in effect

TCP OUTSIDE 192.168.241.50:80 NP Identity Ifc 192.168.240.50:
```

46916

```
, idle 0:00:00, bytes 0,
```

flags z

unit-3-1

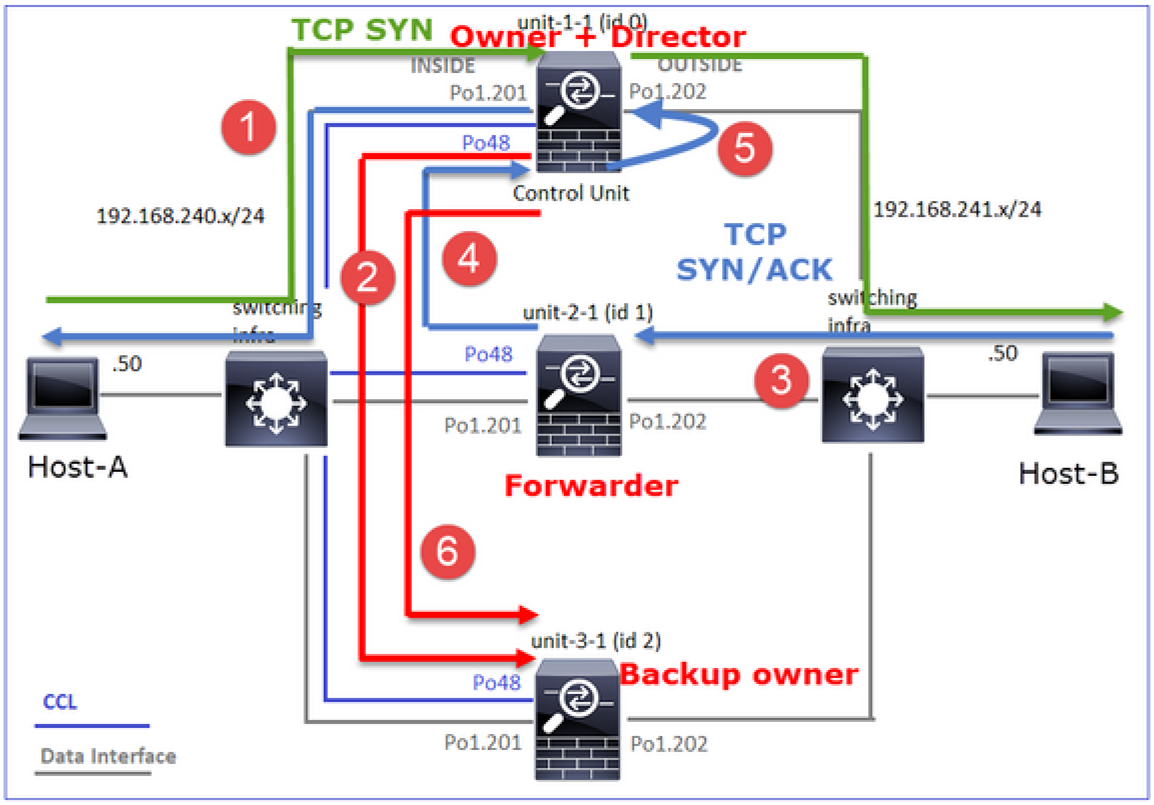
```
:*************************************************************
17 in use, 20 most used
Cluster:
fwd connections: 0 in use, 5 most used
dir connections: 1 in use, 127 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 1 most enabled, 0 most in effect

TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:
```

46916

```
, idle 0:00:04, bytes 0,
flags y
```

Unit	**Flag**	**Note**
Unit-1-1	UIO	· Flow Owner â€“ The unit handles the flow · Director â€“ Since unit-3-1 has â€˜yâ€™ and not â€˜Yâ€™ this implies that unit-1-1 was chosen as the director for this flow. Thus, since it is also the owner, another unit (unit-3-1 in this case) was elected as the backup owner
Unit-2-1	z	· Forwarder
Unit-3-1	y	- Backup owner

This can be visualized as:



1. TCP SYN packet arrives from Host-A to unit-1-1. Unit-1-1 becomes the flow owner and it is elected as a director.
2. Unit-3-1 is elected as a backup owner. The flow owner sends a unicast 'cluster add' message on UDP 4193 to inform the backup owner about the flow.
3. TCP SYN/ACK packet arrives from Host-B to unit-2-1. The flow is asymmetric.
4. Unit-2-1 forwards the packet through the CCL to the owner (due to TCP SYN Cookie).
5. The owner reinjects the packet on the interface OUTSIDE and then forwards the packet towards Host-A.

6. Once the connection is terminated, the owner sends a cluster delete message to remove the flow info from the backup owner.

Observation 3. Capture with trace shows the asymmetric traffic and the redirection from unit-2-1 to unit-1-1.

Unit-2-1 (forwarder)

<#root>

```
firepower#

cluster exec unit unit-2-1 show capture CAPO packet-number 1 trace


1: 16:11:33.653164 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.

46916

:

S

 1331019196:1331019196(0)

ack

 3089755618 win 28960 <mss 1460,sackOK,timestamp 532473211 522117741,nop,wscale 7>
...
Phase: 4
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW


I (1) got initial, attempting ownership.



Phase: 5
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW


I (1) am early redirecting to (0) due to matching action (-1).
```

Observation 4. FTD data plane syslogs show the connection creation and termination on all units:

- Unit-1-1 (owner)
- Unit-2-1 (forwarder)
- Unit-3-1 (backup owner)

<#root>

```
firepower#

cluster exec show log | i 46916


unit-1-1(LOCAL):******************************************************
Dec 01 2020 16:11:33: %FTD-6-302013:

Built inbound TCP connection

 10023 for INSIDE:192.168.240.50/46916 (192.168.240.50/46916) to OUTSIDE:192.168.241.50/80 (192.168.241.
Dec 01 2020 16:11:42: %FTD-6-302014:

Teardown TCP connection

 10023 for INSIDE:192.168.240.50/46916 to OUTSIDE:192.168.241.50/80 duration 0:00:09 bytes 1024010016 TC

unit-2-1:*************************************************************
Dec 01 2020 16:11:33: %FTD-6-302022:

Built forwarder stub TCP connection

 for OUTSIDE:192.168.241.50/80 (192.168.241.50/80) to unknown:192.168.240.50/46916 (192.168.240.50/4691
Dec 01 2020 16:11:42: %FTD-6-302023:

Teardown forwarder TCP connection

 for OUTSIDE:192.168.241.50/80 to unknown:192.168.240.50/46916 duration 0:00:09 forwarded bytes 10240098

unit-3-1:*************************************************************
Dec 01 2020 16:11:33: %FTD-6-302022:

Built backup stub TCP connection

 for INSIDE:192.168.240.50/46916 (192.168.240.50/46916) to OUTSIDE:192.168.241.50/80 (192.168.241.50/80)
Dec 01 2020 16:11:42: %FTD-6-302023:

Teardown backup TCP connection

 for INSIDE:192.168.240.50/46916 to OUTSIDE:192.168.241.50/80 duration 0:00:09 forwarded bytes 0 Cluste
```

Case Study 5. Asymmetric Traffic (owner is different than the director).

Observation 1. The reinject-hide captures show packets on unit-1-1 and unit-2-1 (asymmetric flow):

<#root>

```
firepower#

cluster exec show cap



unit-1-1

(LOCAL):******************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33553207 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Buffer Full - 99396 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Buffer Full - 99224 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
```

```
capture CAPI_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface
```

INSIDE

```
 [Buffer Full -
```

99396 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data
```

reinject-hid

```
e buffer 100000 interface
```

OUTSIDE

```
 [Buffer Full -
```

99928 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
```

unit-2-1

```
:*************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33554251 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Buffer Full - 99052 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data reinject-hide buffer 100000 interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data
```

reinject-hide

```
 buffer 100000 interface
```

OUTSIDE

```
 [Buffer Full -
```

99052 bytes

```
]
match tcp host 192.168.240.50 host 192.168.241.50 eq www

unit-3-1:************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Capturing - 131925 bytes]
capture CAPI type raw-data buffer 100000 trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO type raw-data buffer 100000 trace interface OUTSIDE [Capturing - 2592 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPI_RH type raw-data reinject-hide buffer 100000 interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
capture CAPO_RH type raw-data reinject-hide buffer 100000 interface OUTSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.241.50 eq www
```

Observation 2. Connection flag analysis for flow with source port 46994**:**

<#root>

```
firepower#

cluster exec show conn




unit-1-1

(LOCAL):******************************************************
23 in use, 25 most used
Cluster:
fwd connections: 0 in use, 1 most used
dir connections: 0 in use, 122 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 1 enabled, 0 in effect, 4 most enabled, 1 most in effect

TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:

46994

, idle 0:00:00, bytes 406028640,

flags UIO N1




unit-2-1

:*************************************************************
22 in use, 271 most used
Cluster:
fwd connections: 1 in use, 2 most used
dir connections: 0 in use, 2 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 249 most enabled, 0 most in effect

TCP OUTSIDE 192.168.241.50:80 NP Identity Ifc 192.168.240.50:

46994

, idle 0:00:00, bytes 0,

flags z




unit-3-1

:*************************************************************
17 in use, 20 most used
Cluster:
fwd connections: 2 in use, 5 most used
```

```
dir connections: 1 in use, 127 most used
centralized connections: 0 in use, 0 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 1 most enabled, 0 most in effect

TCP OUTSIDE 192.168.241.50:80 INSIDE 192.168.240.50:
```

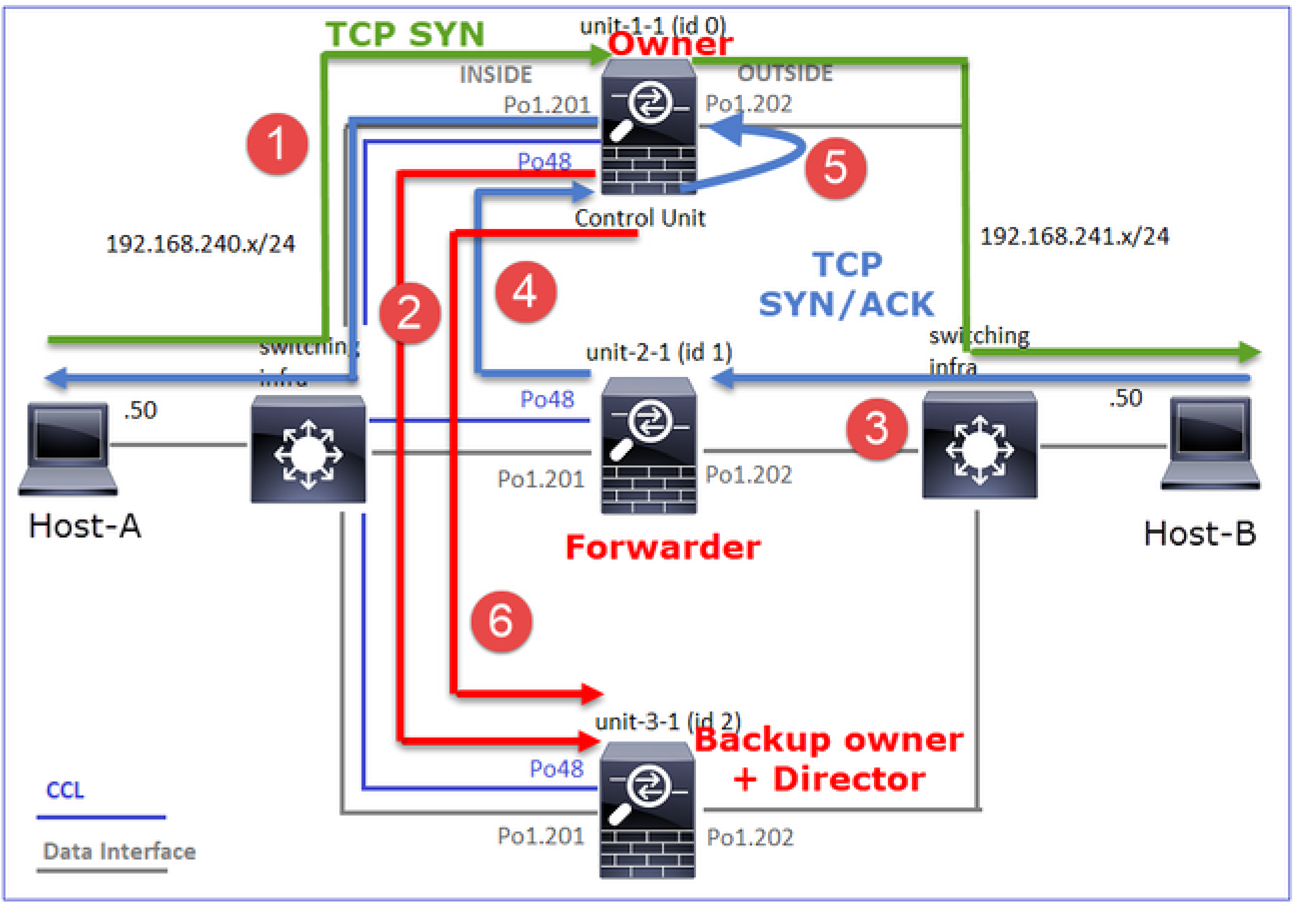
46994

```
, idle 0:00:05, bytes 0,
```

flags Y

Unit	**Flag**	**Note**
Unit-1-1	UIO	· Flow Owner â€“ The unit handles the flow
Unit-2-1	z	· Forwarder
Unit-3-1	Y	· Backup owner · Director

This can be visualized as:



1. TCP SYN packet arrives from Host-A to unit-1-1. Unit-1-1 becomes the flow owner.
2. Unit-3-1 is elected as a director and backup owner. The flow owner sends a 'cluster add' unicast message on UDP 4193 to inform the backup owner about the flow.
3. TCP SYN/ACK packet arrives from Host-B to unit-2-1. The flow is asymmetric
4. Unit-2-1 forwards the packet through the CCL to the owner (due to TCP SYN Cookie).
5. The owner reinjects the packet on the interface OUTSIDE and then forwards the packet towards Host-A.
6. Once the connection is terminated, the owner sends a cluster delete message to remove the flow info from the backup owner.

Observation 3. Capture with trace shows the asymmetric traffic and the redirection from unit-2-1 to unit-1-1.

Unit-1-1 (owner)

<#root>

```
firepower#
```

```
cluster exec show cap CAPI packet-number 1 trace
```

```
unit-1-1(LOCAL):******************************************************
â€¦
Phase: 4
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW


I (0) got initial, attempting ownership.



Phase: 5
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW


I (0) am becoming owner
```

Unit-2-1 (forwarder)

<#root>

```
firepower#
```

```
cluster exec unit unit-2-1 show cap CAPO packet-number 1 trace
```

```
1: 16:46:44.232074 802.1Q vlan#202 P0 192.168.241.50.80 > 192.168.240.50.
```

46994

```
: S 2863659376:2863659376(0) ack 2879616990 win 28960 <mss 1460,sackOK,timestamp 534583774 524228304,nop
â€¦
Phase: 4
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW
```

I (1) got initial, attempting ownership.

```
Phase: 5
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW
```

I (1) am early redirecting to (0) due to matching action (-1).

Observation 4. FTD data plane syslogs show the connection creation and termination on all units:

- Unit-1-1 (owner)
- Unit-2-1 (forwarder)
- Unit-3-1 (backup owner/director)

<#root>

```
firepower#
```

cluster exec show log | i 46994

```
unit-1-1(LOCAL):******************************************************
Dec 01 2020 16:46:44: %FTD-6-302013:
```

Built inbound TCP connection

```
 10080 for INSIDE:192.168.240.50/46994 (192.168.240.50/46994) to OUTSIDE:192.168.241.50/80 (192.168.241.
Dec 01 2020 16:46:53: %FTD-6-302014:
```

Teardown TCP connection

```
 10080 for INSIDE:192.168.240.50/46994 to OUTSIDE:192.168.241.50/80 duration 0:00:09 bytes 1024000440 TC

unit-2-1:*************************************************************
Dec 01 2020 16:46:44: %FTD-6-302022:
```

Built forwarder stub TCP connection

```
 for OUTSIDE:192.168.241.50/80 (192.168.241.50/80) to unknown:192.168.240.50/46994 (192.168.240.50/4699
Dec 01 2020 16:46:53: %FTD-6-302023:
```

`Teardown forwarder TCP connection`

```
 for OUTSIDE:192.168.241.50/80 to unknown:192.168.240.50/46994 duration 0:00:09 forwarded bytes 1024000

unit-3-1:************************************************************
Dec 01 2020 16:46:44: %FTD-6-302022:
```

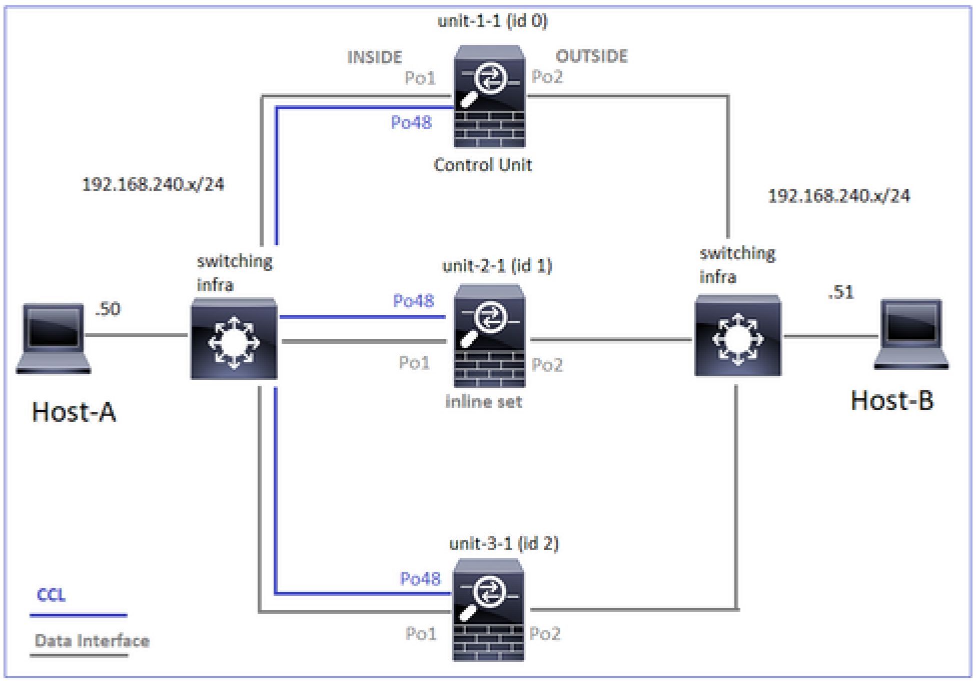
`Built director stub TCP connection`

```
 for INSIDE:192.168.240.50/46994 (192.168.240.50/46994) to OUTSIDE:192.168.241.50/80 (192.168.241.50/80
Dec 01 2020 16:46:53: %FTD-6-302023:
```

`Teardown director TCP connection`

```
 for INSIDE:192.168.240.50/46994 to OUTSIDE:192.168.241.50/80 duration 0:00:09 forwarded bytes 0 Cluste
```

For the next case studies, the topology used is based on a cluster with inline sets:



Case Study 6. Asymmetric Traffic (Inline-set, the owner is the director)

Observation 1. The reinject-hide captures show packets on unit-1-1 and unit-2-1 (asymmetric flow). Additionally, the owner is unit-2-1 (there are packets on both, INSIDE and OUTSIDE interfaces for the reinject-hide captures, while unit-1-1 has only on OUTSIDE):

<#root>

```
firepower#
```

`cluster exec show cap`

`unit-1-1`

```
(LOCAL):******************************************************
```

```
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33553253 bytes]
capture CAPO type raw-data trace interface OUTSIDE [Buffer Full - 523432 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI type raw-data trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPO_RH type raw-data
```

reinject-hide

```
 interface
```

OUTSIDE

```
 [Buffer Full -
```

523432 bytes

```
]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI_RH type raw-data reinject-hide interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
```

unit-2-1

```
:*************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33554312 bytes]
capture CAPO type raw-data trace interface OUTSIDE [Buffer Full - 523782 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI type raw-data trace interface INSIDE [Buffer Full - 523782 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPO_RH type raw-data
```

reinject-hide

```
 interface
```

OUTSIDE

```
 [Buffer Full -
```

524218 bytes

```
]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI_RH type raw-data
```

reinject-hide

```
 interface
```

INSIDE

```
 [Buffer Full -
```

523782 bytes]

```
match tcp host 192.168.240.50 host 192.168.240.51 eq www

unit-3-1:*************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Capturing - 53118 bytes]
capture CAPO type raw-data trace interface OUTSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI type raw-data trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
```

```
capture CAPO_RH type raw-data reinject-hide interface OUTSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI_RH type raw-data reinject-hide interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
```

Observation 2. Connection flag analysis for flow with source port 51844.

<#root>

```
firepower#
```

`cluster exec show conn addr 192.168.240.51`

`unit-1-1`

```
(LOCAL):******************************************************
30 in use, 102 most used
Cluster:
fwd connections: 1 in use, 1 most used
dir connections: 2 in use, 122 most used
centralized connections: 3 in use, 39 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 4 most enabled, 1 most in effect

TCP OUTSIDE 192.168.240.51:80 NP Identity Ifc 192.168.240.50:
```

`51844`

```
, idle 0:00:00, bytes 0,
```

`flags z`

`unit-2-1`

```
:*************************************************************
23 in use, 271 most used
Cluster:
fwd connections: 0 in use, 2 most used
dir connections: 4 in use, 26 most used
centralized connections: 0 in use, 14 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 249 most enabled, 0 most in effect

TCP OUTSIDE 192.168.240.51:80 INSIDE 192.168.240.50:
```

`51844`

```
, idle 0:00:00, bytes 231214400,
```

`flags b N`

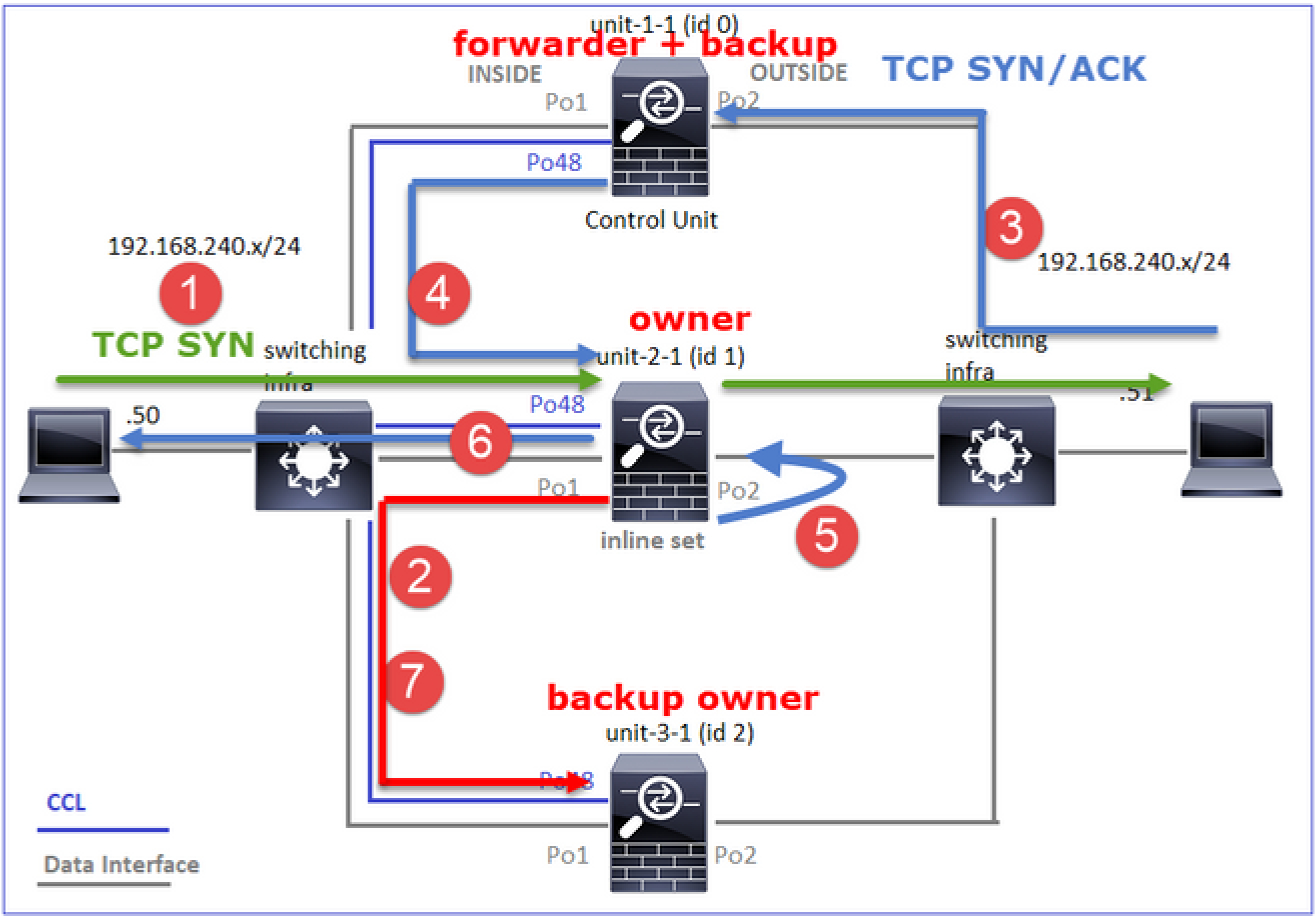
`unit-3-1`

```
:*************************************************************
20 in use, 55 most used
Cluster:
fwd connections: 0 in use, 5 most used
dir connections: 1 in use, 127 most used
centralized connections: 0 in use, 24 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 1 most enabled, 0 most in effect

TCP OUTSIDE 192.168.240.51:80 INSIDE 192.168.240.50:51844, idle 0:00:01, bytes 0,
```

`flags y`

Unit	**Flag**	**Note**
Unit-1-1	z	· Forwarder
Unit-2-1	b N	· Flow Owner â€“ The unit handles the flow
Unit-3-1	y	· Backup owner

This can be visualized as:



1. TCP SYN packet arrives from Host-A to unit-2-1. Unit-2-1 becomes the flow owner and is elected as the director.
2. Unit-3-1 is elected the backup owner. The flow owner sends a 'cluster add' unicast message on UDP 4193 to inform the backup owner about the flow.
3. TCP SYN/ACK packet arrives from Host-B to unit-1-1. The flow is asymmetric.
4. Unit-1-1 forwards the packet through the CCL to the director (unit-2-1).
5. Unit-2-1 is also the owner and reinjects the packet on the interface OUTSIDE.
6. Unit-2-1 forwards the packet towards Host-A.
7. Once the connection is terminated, the owner sends a cluster delete message to remove the flow info from the backup owner.

Observation 3. Capture with trace shows the asymmetric traffic and the redirection from unit-1-1 to unit-2-1.

Unit-2-1 (owner/director)

<#root>

```
firepower#
```

```
cluster exec unit unit-2-1 show cap CAPI packet-number 1 trace
```

```
1: 18:10:12.842912 192.168.240.50.51844 > 192.168.240.51.80:
```

```
S
```

```
 4082593463:4082593463(0) win 29200 <mss 1460,sackOK,timestamp 76258053 0,nop,wscale 7>
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

```
I (1) got initial, attempting ownership.
```

```
Phase: 2
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

```
I (1) am becoming owner
```

Unit-1-1 (forwarder)

<#root>

```
firepower#
```

```
cluster exec show cap CAPO packet-number 1 trace


unit-1-1(LOCAL):******************************************************

1: 18:10:12.842317 192.168.240.51.80 > 192.168.240.50.51844: S 2339579109:2339579109(0) ack 4082593464 w
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW


I (0) am asking director (1).
```

Return traffic (TCP SYN/ACK)

Unit-2-1 (owner/director)

<#root>

```
firepower#

cluster exec unit unit-2-1 show cap CAPO packet-number 2 trace



2: 18:10:12.843660 192.168.240.51.80 > 192.168.240.50.51844: S 2339579109:2339579109(0) ack 4082593464 w
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: FULL


I (1) am owner, update sender (0).



Phase: 2
Type: FLOW-LOOKUP
Subtype:
Result: ALLOW
Config:
Additional Information:


Found flow with id 7109, using existing flow
```

Observation 4. FTD data plane syslogs show the connection creation and termination on all units:

- Unit-1-1 (owner)

- Unit-2-1 (forwarder)
- Unit-3-1 (backup owner/director)

<#root>

```
firepower#

cluster exec show log | include 51844


unit-1-1(LOCAL):******************************************************
Dec 02 2020 18:10:12: %FTD-6-302022:

Built forwarder stub TCP connection

 for OUTSIDE:192.168.240.51/80 (192.168.240.51/80) to unknown:192.168.240.50/51844 (192.168.240.50/5184
Dec 02 2020 18:10:22: %FTD-6-302023:

Teardown forwarder TCP connection

 for OUTSIDE:192.168.240.51/80 to unknown:192.168.240.50/51844 duration 0:00:09 forwarded bytes 10240017

unit-2-1:*************************************************************
Dec 02 2020 18:10:12: %FTD-6-302303:

Built TCP state-bypass connection

 7109 from INSIDE:192.168.240.50/51844 (192.168.240.50/51844) to OUTSIDE:192.168.240.51/80 (192.168.240.
Dec 02 2020 18:10:22: %FTD-6-302304:

Teardown TCP state-bypass connection

 7109 from INSIDE:192.168.240.50/51844 to OUTSIDE:192.168.240.51/80 duration 0:00:09 bytes 1024001888 TC

unit-3-1:*************************************************************
Dec 02 2020 18:10:12: %FTD-6-302022:

Built backup stub TCP connection

 for INSIDE:192.168.240.50/51844 (192.168.240.50/51844) to OUTSIDE:192.168.240.51/80 (192.168.240.51/80)
Dec 02 2020 18:10:22: %FTD-6-302023:

Teardown backup TCP connection

 for INSIDE:192.168.240.50/51844 to OUTSIDE:192.168.240.51/80 duration 0:00:09 forwarded bytes 0 Cluste
```

Case Study 7. Asymmetric Traffic (Inline-set, the owner is different than the director)

The owner is unit-2-1 (there are packets on both, INSIDE and OUTSIDE interfaces for the reinject-hide captures, while unit-3-1 has only on OUTSIDE):

<#root>

```
firepower#

cluster exec show cap


unit-1-1(LOCAL):******************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Capturing - 13902 bytes]
capture CAPO type raw-data trace interface OUTSIDE [Capturing - 90 bytes]
```

```
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI type raw-data trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPO_RH type raw-data reinject-hide interface OUTSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI_RH type raw-data reinject-hide interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
```

unit-2-1

```
:****************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33553936 bytes]
capture CAPO type raw-data trace interface OUTSIDE [Buffer Full - 523126 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI type raw-data trace interface INSIDE [Buffer Full - 523126 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPO_RH type raw-data
```

reinject-hid

e

 interface

OUTSIDE

 [Buffer Full -

524230 bytes

```
]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI_RH type raw-data
```

reinject-hide

 interface

INSIDE

 [Buffer Full -

523126 bytes

```
]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
```

unit-3-1

```
:****************************************************************
capture CCL type raw-data buffer 33554432 interface cluster [Buffer Full - 33553566 bytes]
capture CAPO type raw-data trace interface OUTSIDE [Buffer Full - 523522 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI type raw-data trace interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPO_RH type raw-data
```

reinject-hide

 interface

OUTSIDE

```
 [Buffer Full -
```

523432 bytes

```
]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
capture CAPI_RH type raw-data reinject-hide interface INSIDE [Capturing - 0 bytes]
match tcp host 192.168.240.50 host 192.168.240.51 eq www
```

Observation 2. Connection flag analysis for flow with source port 59210.

<#root>

```
firepower#
```

cluster exec show conn addr 192.168.240.51

unit-1-1

```
(LOCAL):******************************************************
25 in use, 102 most used
Cluster:
fwd connections: 0 in use, 1 most used
dir connections: 2 in use, 122 most used
centralized connections: 0 in use, 39 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 4 most enabled, 1 most in effect

TCP OUTSIDE 192.168.240.51:80 INSIDE 192.168.240.50:
```

59210

```
, idle 0:00:03, bytes 0,
```

flags Y

unit-2-1

```
:*************************************************************
21 in use, 271 most used
Cluster:
fwd connections: 0 in use, 2 most used
dir connections: 0 in use, 28 most used
centralized connections: 0 in use, 14 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 249 most enabled, 0 most in effect

TCP OUTSIDE 192.168.240.51:80 INSIDE 192.168.240.50:
```

59210

```
, idle 0:00:00, bytes 610132872,
```

flags b N

unit-3-1

```
:*************************************************************
19 in use, 55 most used
Cluster:
fwd connections: 1 in use, 5 most used
dir connections: 0 in use, 127 most used
centralized connections: 0 in use, 24 most used
VPN redirect connections: 0 in use, 0 most used
Inspect Snort:
preserve-connection: 0 enabled, 0 in effect, 1 most enabled, 0 most in effect

TCP OUTSIDE 192.168.240.51:80 NP Identity Ifc 192.168.240.50:
```

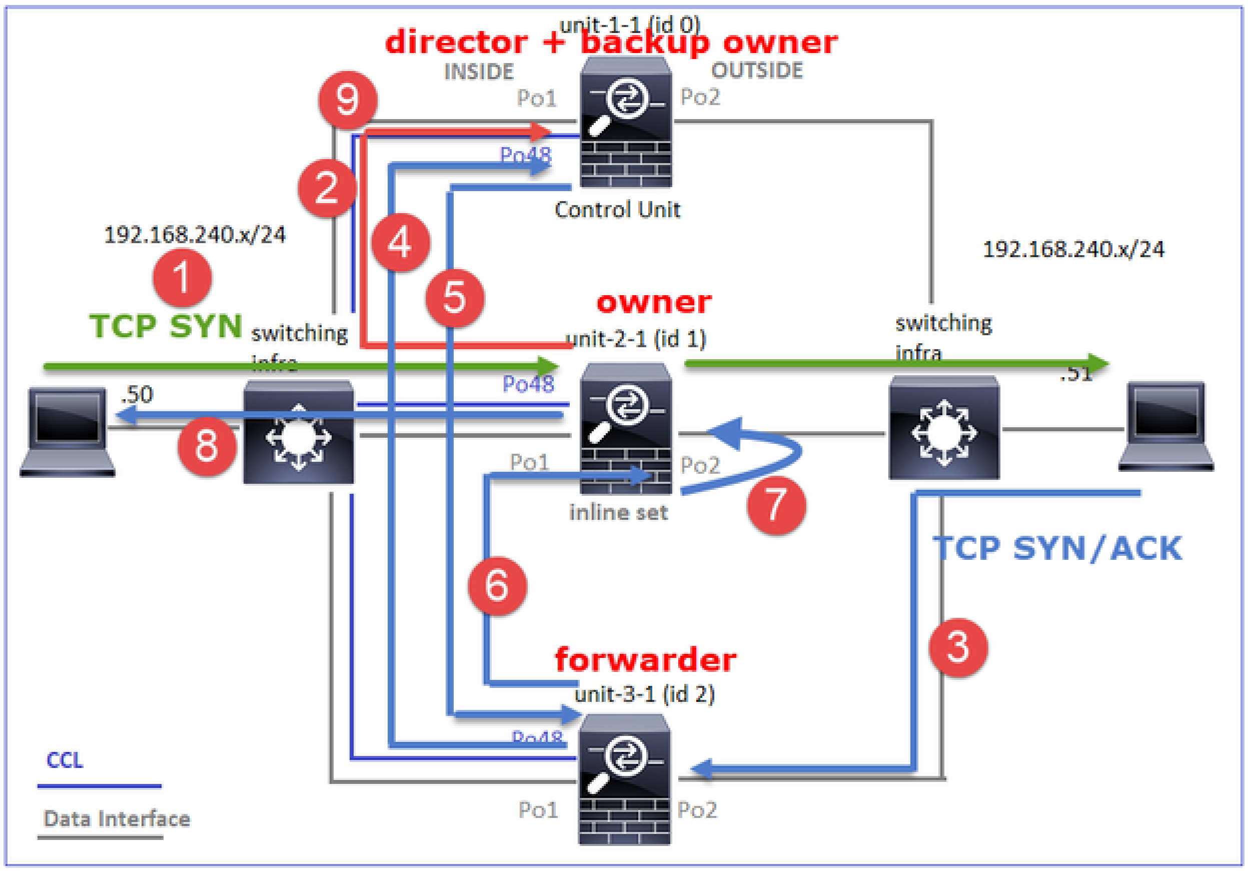
59210

```
, idle 0:00:00, bytes 0,
```

flags z

Unit	**Flag**	**Note**
Unit-1-1	Y	· Director/Backup owner
Unit-2-1	b N	· Flow Owner â€“ The unit handles the flow
Unit-3-1	z	· Forwarder

This can be visualized as:



1. TCP SYN packet arrives from Host-A to unit-2-1. Unit-2-1 becomes the flow owner and unit-1-1 is elected as the director
2. Unit-1-1 is elected the backup owner (since it is the director). The flow owner sends a 'cluster add' unicast message on UDP 4193 to. inform the backup owner about the flow.
3. TCP SYN/ACK packet arrives from Host-B to unit-3-1. The flow is asymmetric.
4. Unit-3-1 forwards the packet through the CCL to the director (unit-1-1).
5. Unit-1-1 (director) knows that the owner is unit-2-1, sends the packet back to the forwarder (unit-3-1), and notifies him that the owner is unit-2-1.
6. Unit-3-1 sends the packet to unit-2-1 (owner).
7. Unit-2-1 reinjects the packet on the interface OUTSIDE.
8. Unit-2-1 forwards the packet towards Host-A.
9. Once the connection is terminated, the owner sends a cluster delete message to remove the flow info from the backup owner.

Note: It is important for step 2 (packet through the CCL) to occur before step 4 (data traffic). In a different case (for example, race condition), the director is not aware of the flow. Thus, since it is an inline set, forwards the packet toward the destination. If the interfaces are not in an inline set, the data packet is dropped.

Observation 3. Capture with trace shows the asymmetric traffic and the exchanges over the CCL:

Forward traffic (TCP SYN)

Unit-2-1 (owner)

<#root>

```
firepower#
```

cluster exec unit unit-2-1 show cap CAPI packet-number 1 trace

```
1: 09:19:49.760702 192.168.240.50.59210 > 192.168.240.51.80: S 4110299695:4110299695(0) win 29200 <mss 1
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

I (1) got initial, attempting ownership.

```
Phase: 2
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'INSIDE'
Flow type: NO FLOW
```

I (1) am becoming owner

Return traffic (TCP SYN/ACK)

Unit-3-1 (ID 2 - forwarder) sends the packet through the CCL to unit-1-1 (ID 0 - director).

<#root>

firepower#

cluster exec unit unit-3-1 show cap CAPO packet-number 1 trace

```
1: 09:19:49.760336 192.168.240.51.80 > 192.168.240.50.59210:
```

S

```
 4209225081:4209225081(0)
```

ack

```
 4110299696 win 28960 <mss 1460,sackOK,timestamp 567715984 130834570,nop,wscale 7>
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: NO FLOW
```

```
I (2) am asking director (0).
```

Unit-1-1 (director) â€“ Unit-1-1 (ID 0) knows that the flow owner is unit-2-1 (ID 1) and sends the packet over the CCL back to unit-3-1 (ID 2 - forwarder).

<#root>

```
firepower#

cluster exec show cap CAPO packet-number 1 trace


unit-1-1(LOCAL):******************************************************

1: 09:19:49.761038 192.168.240.51.80 > 192.168.240.50.59210:

S

 4209225081:4209225081(0)

ack

 4110299696 win 28960 <mss 1460,sackOK,timestamp 567715984 130834570,nop,wscale 7>
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: STUB


I (0) am director, valid owner (1), update sender (2).
```

Unit-3-1 (ID 2 - forwarder) gets the packet through the CCL and sends it to unit-2-1 (ID 1 - owner).

<#root>

```
firepower#

cluster exec unit unit-3-1 show cap CAPO packet-number 2 trace


...
2: 09:19:49.761008 192.168.240.51.80 > 192.168.240.50.59210:

S

 4209225081:4209225081(0) ack 4110299696 win 28960 <mss 1460,sackOK,timestamp 567715984 130834570,nop,ws
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: STUB
```

```
I (2) am becoming forwarder to (1), sender (0).
```

The owner reinjects and forwards the packet towards the destination:

<#root>

```
firepower#

cluster exec unit unit-2-1 show cap CAPO packet-number 2 trace



2: 09:19:49.775701 192.168.240.51.80 > 192.168.240.50.59210:

S

 4209225081:4209225081(0)

ack

 4110299696 win 28960 <mss 1460,sackOK,timestamp 567715984 130834570,nop,wscale 7>
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'OUTSIDE'
Flow type: FULL


I (1) am owner, sender (2).
```

Observation 4. FTD data plane syslogs show the connection creation and termination on all units:

- Unit-1-1 (director/backup owner)
- Unit-2-1 (owner)
- Unit-3-1 (forwarder)

<#root>

```
firepower#

 cluster exec show log | i 59210


unit-1-1(LOCAL):******************************************************
Dec 03 2020 09:19:49: %FTD-6-302022:

Built director stub TCP connection

 for INSIDE:192.168.240.50/59210 (192.168.240.50/59210) to OUTSIDE:192.168.240.51/80 (192.168.240.51/80)
Dec 03 2020 09:19:59: %FTD-6-302023:

Teardown director TCP connection

 for INSIDE:192.168.240.50/59210 to OUTSIDE:192.168.240.51/80 duration 0:00:09 forwarded bytes 0 Cluster

unit-2-1:*************************************************************
```

```
Dec 03 2020 09:19:49: %FTD-6-302303:
```

Built TCP state-bypass connection

```
 14483 from INSIDE:192.168.240.50/59210 (192.168.240.50/59210) to OUTSIDE:192.168.240.51/80 (192.168.240
Dec 03 2020 09:19:59: %FTD-6-302304:
```

Teardown TCP state-bypass connection

```
 14483 from INSIDE:192.168.240.50/59210 to OUTSIDE:192.168.240.51/80 duration 0:00:09 bytes 1024003336 T

unit-3-1:************************************************************
Dec 03 2020 09:19:49: %FTD-6-302022:
```

Built forwarder stub TCP connection

```
 for OUTSIDE:192.168.240.51/80 (192.168.240.51/80) to unknown:192.168.240.50/59210 (192.168.240.50/59210
Dec 03 2020 09:19:59: %FTD-6-302023:
```

Teardown forwarder TCP connection

```
 for OUTSIDE:192.168.240.51/80 to unknown:192.168.240.50/59210 duration 0:00:09 forwarded bytes 10240031
```

Troubleshoot

Cluster Troubleshooting Introduction

The cluster problems can be categorized in:

- Control Plane Issues (issues related to the cluster stability)
- Data Plane Issues (issues related to the transit traffic)

Cluster Data Plane Issues

NAT/PAT Common Issues

Important Configuration Considerations

- Port Address Translation (PAT) pools must have at least as many IPs available as the number of units in the cluster, preferably more IPs than cluster nodes.
- The default **xlate per-session** commands must be left in place unless there is a specific reason to disable them. Any PAT xlate built for a connection that has xlate per-session disabled is always handled by the control node unit in the cluster, which can cause performance degradation.

High PAT pool range usage due to traffic sourced from low-ports that causes cluster IP imbalance

The FTD divides up a PAT IP into ranges and tries to maintain the xlate in the same source range. This table shows how a source port is translated to a global port within the same source range.

Original Src Port	**Translated Src Port**
1-511	1-511

512-1023	512-1023
1024-65535	1024-65535

When a source port range is full and a new PAT xlate needs to be allocated from that range, FTD moves to the next IP to allocate new translations for that source port range.

Symptoms

Connectivity issues for NATed traffic that traverses the cluster

Verification

<#root>

```
#
show nat pool
```

FTD data plane logs show PAT pool exhaustion:

<#root>

```
Dec 9 09:00:00 192.0.2.10 FTD-FW %ASA-3-202010:
PAT pool exhausted. Unable to create TCP connection
 from Inside:192.0.2.150/49464 to Outside:192.0.2.250/20015
Dec 9 09:00:00 192.0.2.10 FTD-FW %ASA-3-202010:
PAT pool exhausted. Unable to create TCP connection
 from Inside:192.0.2.148/54141 to Outside:192.0.2.251/443
```

Mitigation

Configure NAT Flat Port Range and Include Reserve Ports.

Additionally, in post-6.7/9.15.1 you can end up with imbalanced port block distribution only when nodes leave/join the cluster with huge background traffic that is subject to PAT. The only way it recovers by itself is when port blocks are freed up to be redistributed across nodes.

With port block-based distribution, when a node is allocated with say 10 port blocks such as pb-1, pb-2 ... pb-10. The node always starts with the first available port block and allocates a random port from it until it exhausts. The allocation moves to the next port block only when all the port blocks up to that point are exhausted.

For example, if a host establishes 512 connections, the unit allocates mapped ports for all those 512 connections from pb-1 randomly. Now, with all these 512 connections active, when the host establishes the 513th connection since pb-1 is exhausted, it moves to pb-2 and allocates a random port from it. Now again, out of 513 connections, assume the 10th connection finished and cleared one port available in pb-1. At this point, if the host establishes the 514th connection, the cluster unit allocates a mapped port from pb-1 and not pb-2 because pb-1 now has a free port (which was released as part of the 10th connection removal).

The important part to keep in mind is that the allocation happens from the first available port block with free ports so that the last port blocks are always available for redistribution in a normally loaded system. Additionally, PAT is typically used for short-lived connections. The probability of a port block to become available in a shorter time is very high. So, the time required for pool distribution to become balanced can improve with port block-based pool distribution.

However, in case all port blocks, from pb-1 to pb-10, are exhausted or each port block holds a port for a long-lived connection, the port blocks never get freed up quickly and get redistributed. In such a case, the least disruptive approach is to:

1. Identify nodes with excessive port blocks (show nat pool cluster summary).
2. Identify the least used port blocks on that node (show nat pool ip <addr> detail).
3. Clear xlates for such port blocks (clear xlate global <addr> gport 'start-end') to make them available for redistribution.

Warning: This disrupts the relevant connections.

Unable to browse to dual-channel websites (like webmail, banking, etc), or to SSO websites when redirection to a different destination happens.

Symptoms

Unable to browse to dual-channel websites (like webmail, bank websites, and so on). When a user connects to a website that requires the client to open a second socket/connection and the second connection is hashed to a cluster member different from the one that got the first connection hashed to, and traffic uses an IP PAT pool, traffic is reset by the server as it receives the connection from a different public IP address.

Verification

Take data plane cluster captures to see how the affected transit flow is handled. In this case, a TCP reset comes from the destination website.

Mitigation (pre-6.7/9.15.1)

- Observe if any multi-session applications use multiple mapped IP addresses.
- Use the **show nat pool cluster summary** command to check if the pool is distributed evenly.
- Use the **cluster exec show conn** command to check if traffic is load balanced properly.
- Use the **show nat pool cluster ip <address> detail** command to check the pool usage of sticky IP.
- Enable syslog 305021 (6.7/9.15) to see which connections have failed to use sticky IP.
- To solve add more IPs to the PAT pool or fine-tune the load balance algorithm on connected switches.

About the ether-channel load balancing algorithm:

- For non-FP9300 and if Authentication occurs via one server: Adjust the ether-channel load balancing algorithm on the adjacent switch from Source IP/Port and Destination IP/Port to Source IP and Destination IP.
- For non-FP9300 and if Authentication occurs via multiple servers: Adjust the ether-channel load balancing algorithm on the adjacent switch from Source IP/Port and Destination IP/Port to Source IP.
- For FP9300: On the FP9300 chassis the load balancing algorithm is fixed as source-dest-port source-dest-ip source-dest-mac and cannot be changed. The workaround, in this case, is to use FlexConfig to add **xlate per-session deny** commands to the FTD configuration to force traffic for certain destination

IP addresses (for the problematic/incompatible applications) to be handled by only the control node in the intra-chassis cluster. The workaround comes with these side effects:
    - No load balancing of the differently-translated traffic (everything goes to the control node).
    - Potential for xlate slots to run out (and adversely affect NAT translation for other traffic on the control node).
    - Reduced scalability of the intra-chassis cluster.

Low cluster performance due to all traffic sent to the control node because of not enough PAT IPs in the pools.

Symptoms

There are not enough PAT IPs in the cluster to allocate a free IP to the data nodes, and therefore, all traffic subject to the PAT config is forwarded to the control node for processing.

Verification

Use the **show nat pool cluster** command to see the allocations for each unit and confirm that they all own at least one IP in the pool.

Mitigation

For pre-6.7/9.15.1 ensure you have a PAT pool of size at least equal to the number of nodes in the cluster. In post-6.7/9.15.1 with PAT pool, you allocate port blocks from all the PAT pool IPs. If the PAT pool usage is really high which leads to frequent exhaustion of the pool you need to increase the PAT pool size (see the FAQ section).

Low performance due to all traffic sent to the control node because xlates are not per-session enabled.

Symptoms

Lots of high-speed UDP backup flows are processed through the cluster control node, which can impact the performance.

Background

Only connections that use xlates that are per-session enabled can be processed by a data node that uses PAT. Use the command **show run all xlate** to see the xlate per-session config.

Per-session enabled means that the xlate is torn down immediately when the associated connection is torn down. This helps improve connection per second performance when the connections are subjected to PAT. Non-per-session xlates live for another 30 seconds after the associated connection is torn down, and if the connection rate is high enough, the available 65k TCP/UDP ports on each global IP can be used up in a short amount of time.

By default, all TCP traffic is per-xlate enabled and only UDP DNS traffic is per-session enabled. This means all non-DNS UDP traffic is forwarded to the control node for processing.

Verification

Use this command to check the connection and packet distribution among the cluster units:

<#root>

```
firepower#
```

```
show cluster info conn-distribution
```

```
firepower#

show cluster info packet-distribution
```

```
firepower#

show cluster info load-monitor
```

Use the **cluster exec show conn** command to see which cluster nodes own the UDP connections.

<#root>

```
firepower#

cluster exec show conn
```

Use this command to understand the pool usage across cluster nodes.

<#root>

```
firepower#

cluster exec show nat pool ip <addr> | in UDP
```

Mitigation

Configure per-session PAT (**per-session permit udp** command) for the traffic of interest (for example, UDP). For ICMP, you cannot change from the default multi-session PAT, thus ICMP traffic is handled always by the control node when there is PAT configured.

PAT pool distribution becomes imbalanced as nodes leave/join the cluster.

Symptoms

- Connectivity issues since PAT IP allocation can become imbalanced over time due to units who leave and join the cluster.
- In post-6.7/9.15.1, there can be cases where the newly joined node cannot get enough port blocks. A node that does not have any port block redirects traffic to the control node. A node that has at least one port block handles the traffic and drops it once the pool is exhausted.

Verification

- The data plane syslogs show messages like:

<#root>

```
%ASA-3-202010:

NAT pool exhausted. Unable to create TCP connection
```

```
 from inside:192.0.2.1/2239 to outside:192.0.2.150/80
```

- Use the **show nat pool cluster summary** command to identify the pool distribution.
- Use the **cluster exec show nat pool ip <addr> detail** command to understand the pool usage across cluster nodes.

Mitigation

- For pre-6.7/9.15.1 a few workarounds are described in Cisco bug ID CSCvd10530 .
- In post-6.7/9.15.1, use the **clear xlate global <ip> gport <start-end>** command to manually clear some of the port blocks on other nodes for redistribution to the required nodes.

Symptoms

Major connectivity issues for traffic that is PATed by the cluster. This is because the FTD data plane, per design, does not send GARP for global NAT addresses.

Verification

The ARP table of the directly connected devices shows different the MAC address of the cluster data interface after a change of the control node:

<#root>

```
root@kali2:~/tests#

arp -a


? (192.168.240.1) at f4:db:e6:

33:44:2e

 [ether] on eth0
root@kali2:~/tests#

arp -a


? (192.168.240.1) at f4:db:e6:

9e:3d:0e

 [ether] on eth0
```

Mitigation

Configure static (virtual) MAC on cluster data interfaces.

Connections subjected to PAT fail

Symptoms

Connectivity issues for traffic that is PATed by the cluster.

Verification/Mitigation

- Ensure that the configuration is replicated properly.
- Ensure that the pool is distributed evenly.
- Ensure that pool ownership is valid.
- No failure counter increments in show asp cluster counter.
- Ensure director/forwarder flows are created with proper information.
- Validate if backup xlates are created, updated, and cleaned up as expected.
- Validate if xlates are created and terminated as per â€œper-sessionâ€� behavior.
- Enable â€œdebug nat 2â€� for an indication of any errors. Note, this output can be very noisy, for example:

<#root>

```
firepower#
```

`debug nat 2`

```
nat:
```

`no free blocks available to reserve for 192.168.241.59, proto 17`

```
nat: no free blocks available to reserve for 192.168.241.59, proto 17
nat: no free blocks available to reserve for 192.168.241.58, proto 17
nat: no free blocks available to reserve for 192.168.241.58, proto 17
nat: no free blocks available to reserve for 192.168.241.57, proto 17
```

To stop the debug:

<#root>

```
firepower#
```

`un all`

- Enable connection and NAT-related syslogs to correlate the information to a failed connection.

ASA and FTD Clustering PAT Improvements (post-9.15 and 6.7)

What has changed?

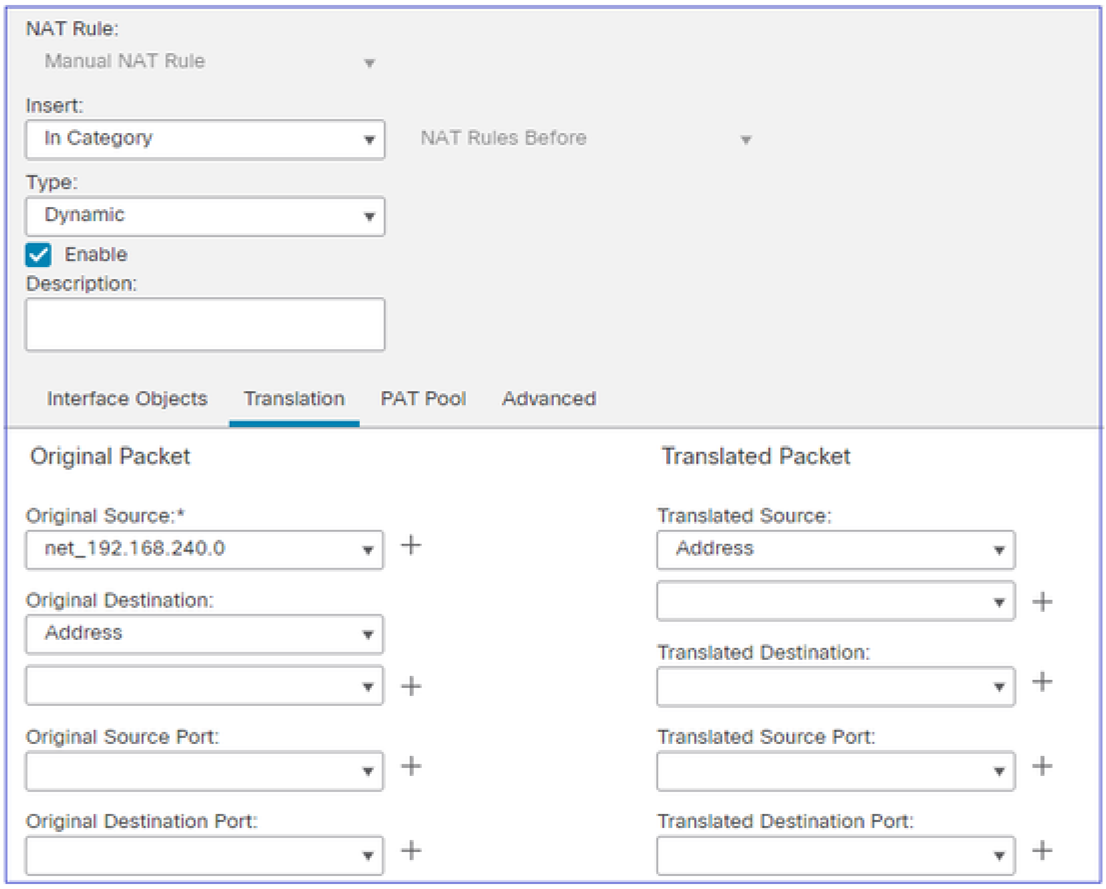
The PAT operation was redesigned. Individual IPs are not any more distributed to each of the cluster members. Instead, the PAT IP(s) are split into port blocks and distributed those port blocks evenly (as much as possible) between the cluster members, in combination with IP stickiness operation.

The new design addresses these limitations (see the previous section):

- Multi-session applications are affected due to a lack of cluster-wide IP stickiness.
- The requirement is to have a PAT pool of size at least equal to the number of nodes in the cluster.
- PAT pool distribution becomes imbalanced as nodes leave/join the cluster.
- No syslogs to indicate PAT pool imbalance.

Technically, Instead of the default 1-511, 512-1023, and 1024-65535 port ranges, now there is 1024-65535 as the default port range for PAT. This default range can be extended to include privileged port range 1-1023 for regular PAT (â€˜include-reserveâ€™ option).

This is an example of a PAT pool configuration on FTD 6.7. For additional details check the related section in the Configuration Guide:



NAT Rule: Manual NAT Rule

Insert: In Category

NAT Rules Before

Type: Dynamic

[✓] Enable

Description:

Interface Objects | Translation | PAT Pool | Advanced

Original Packet

Original Source:* net_192.168.240.0

Original Destination: Address

Original Source Port:

Original Destination Port:

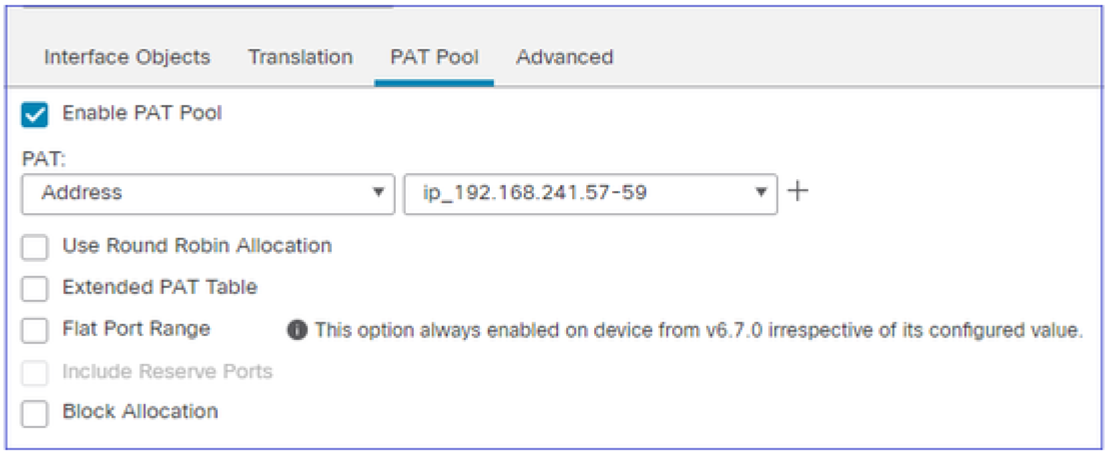
Translated Packet

Translated Source: Address

Translated Destination:

Translated Source Port:

Translated Destination Port:



Interface Objects | Translation | PAT Pool | Advanced

[✓] Enable PAT Pool

PAT: Address | ip_192.168.241.57-59

[ ] Use Round Robin Allocation

[ ] Extended PAT Table

[ ] Flat Port Range — This option always enabled on device from v6.7.0 irrespective of its configured value.

[ ] Include Reserve Ports

[ ] Block Allocation

Additional Troubleshooting information about PAT

FTD data plane syslogs (post-6.7/9.15.1)

A stickiness invalidation syslog is generated when all ports are exhausted in the sticky IP on a cluster node, and allocation moves to the next available IP with free ports, for example:

```
%ASA-4-305021: Ports exhausted in pre-allocated PAT pool IP 192.0.2.100 for host 198.51.100.100 Allocati
```

A Pool imbalance syslog is generated on a node when it joins the cluster and does not get any or unequal share of port blocks, for example:

```
%ASA-4-305022: Cluster unit ASA-4 has been allocated 0 port blocks for PAT usage. All units should have
%ASA-4-305022: Cluster unit ASA-4 has been allocated 12 port blocks for PAT usage. All units should have
```

Show commands

Pool distribution status

In the show nat pool cluster summary output, for each PAT IP address, there must not be a difference of more than 1 port block across the nodes in a balanced distribution scenario. Examples of a balanced and imbalanced port block distribution.

<#root>

```
firepower#
```

`show nat pool cluster summary`

```
port-blocks count display order: total, unit-1-1, unit-2-1, unit-3-1
IP OUTSIDE:ip_192.168.241.57-59 192.168.241.57 (126 -
```

`42 / 42 / 42`

```
)
IP OUTSIDE:ip_192.168.241.57-59 192.168.241.58 (126 - 42 / 42 / 42)
IP OUTSIDE:ip_192.168.241.57-59 192.168.241.59 (126 - 42 / 42 / 42)
```

Imbalanced distribution:

<#root>

```
firepower#
```

`show nat pool cluster summary`

```
port-blocks count display order: total, unit-1-1, unit-4-1, unit-2-1, unit-3-1
IP outside:src_map 192.0.2.100 (128 - 32 /
```

`22 / 38`

```
 / 36)
```

Pool ownership status

In the show nat pool cluster output, there must not be a single port block with either owner or backup as UNKNOWN. If there is one, it indicates a problem with pool ownership communication. Example:

<#root>

```
firepower#
```

```
show nat pool cluster | in <UNKNOWN>
```

```
[3072-3583], owner unit-4-1, backup <
```

```
UNKNOWN
```

```
>
[56832-57343], owner <UNKNOWN>, backup <UNKNOWN>
[10240-10751], owner unit-2-1, backup <UNKNOWN>
```

Accounting of port allocations in port blocks

The **show nat pool** command is enhanced with additional options to display detailed information as well as filtered output. Example:

<#root>

```
firepower#
```

```
show nat pool detail
```

```
TCP PAT pool INSIDE, address 192.168.240.1, range 1-1023, allocated 0
TCP PAT pool INSIDE, address 192.168.240.1, range 1024-65535, allocated 18
UDP PAT pool INSIDE, address 192.168.240.1, range 1-1023, allocated 0
UDP PAT pool INSIDE, address 192.168.240.1, range 1024-65535, allocated 20
TCP PAT pool OUTSIDE, address 192.168.241.1, range 1-1023, allocated 0
TCP PAT pool OUTSIDE, address 192.168.241.1, range 1024-65535, allocated 18
UDP PAT pool OUTSIDE, address 192.168.241.1, range 1-1023, allocated 0
UDP PAT pool OUTSIDE, address 192.168.241.1, range 1024-65535, allocated 20
UDP PAT pool OUTSIDE, address 192.168.241.58
range 1024-1535, allocated 512
range 1536-2047, allocated 512
range 2048-2559, allocated 512
range 2560-3071, allocated 512
...
unit-2-1:************************************************************
UDP PAT pool OUTSIDE, address 192.168.241.57
range 1024-1535, allocated 512 *
range 1536-2047, allocated 512 *
range 2048-2559, allocated 512 *
```

â€˜*â€™ indicates that it is a backed-up port block

To resolve this use the **clear xlate global <ip> gport <start-end>** command to manually clear some of the port blocks on other nodes for redistribution to the required nodes.

Manually triggered redistribution of port blocks

- In a production network with constant traffic, when a node leaves and rejoins the cluster (probably due to a traceback), there can be cases where it cannot get an equal share of the pool or, in the worst case, it cannot get any port block.
- Use the **show nat pool cluster summary** command to identify which node owns more port blocks than required.

- On the nodes that own more port blocks, use the **show nat pool ip <addr> detail** command to figure out the port blocks with the least number of allocations.
- Use the **clear xlate global <address> gport <start-end>** command to clear translations created out of those port blocks so that they become available for redistribution to the required nodes, for example:

<#root>

```
firepower#

show nat pool detail | i 19968


        range 19968-20479, allocated 512
        range 19968-20479, allocated 512
        range 19968-20479, allocated 512

firepower#

clear xlate global 192.168.241.57 gport 19968-20479


INFO: 1074 xlates deleted
```

Frequently Asked Questions (FAQ) for post-6.7/9.15.1 PAT

Q. In case you have the number of IPs available for the number of available units in the cluster, can you still use 1 IP per unit as an option?

A. Not anymore, and there is no toggle to switch between IP addresses-based vs port block-based pool distribution schemes.

The older scheme of IP address-based pool distribution resulted in multi-session application failures where multiple connections (which are part of a single application transaction) from a host are load-balanced onto different nodes of the cluster and thus translated by different mapped IP addresses which lead to the destination server to see them as sourced from different entities.

And, with the new port block-based distribution scheme, even though you now can work with as low as a single PAT IP address, it is always recommended to have enough PAT IP addresses based on the number of connections that are required to be PATed.

Q. Can you still have a pool of IP Addresses for the PAT pool for the cluster?

A. Yes, you can. Port blocks from all the PAT pool IPs get distributed across the cluster nodes.

Q. If you use a number of IP addresses for the PAT pool, is the same block of ports given out to each member per each IP address?

A. No, each IP is distributed independently.

Q. All cluster nodes have all the public IPs, but just a subset of ports? If this is the case, is it then guaranteed that every time the source IP uses the same public IP?

A. That is correct, each PAT IP is partially owned by each node. If a chosen public IP is exhausted on a node, a syslog is generated that indicates that sticky IP cannot be preserved, and the allocation moves to the next available public IP. Be it a standalone, HA, or Cluster deployment, IP stickiness is always on a best effort basis subject to the pool availability.

Q. Is everything based on a single IP address in the PAT pool, but does not apply if more than one IP address in the PAT pool is used?

A. It applies to multiple IP addresses in PAT Pool as well. Port blocks from every IP in the PAT Pool are distributed across cluster nodes. Every IP address in the PAT pool gets split across all members in the cluster. So, if you have, a class C of addresses in the PAT pool, every cluster member has port pools from every one of the PAT pool addresses.

Q. Does it work with CGNAT?

A. Yes, CGNAT is as well supported. CGNAT, also known as block-allocation PAT has a default block size of '512' which can be modified through xlate block-allocation size CLI. In the case of regular dynamic PAT (non-CGNAT), the block size is always '512' which is fixed and non-configurable.

Q. If the unit leaves the cluster, does the control node assign the port block range to other units or keep it to itself?

A. Each port block has an owner and backup. Every time an xlate is created from a port block, it is also replicated to the port block backup node. When a node leaves the cluster, the backup node owns all the port blocks and all the current connections. The backup node, since it has become the owner of these additional port blocks, elects a new backup for them and replicates all current xlates to that node to handle failure scenarios.

Q. What action can be taken on basis of that alert to enforce stickiness?

A. There are two possible reasons why stickiness cannot be preserved.

Reason-1: The traffic is incorrectly load-balanced due to which one of the nodes sees a higher number of connections than others which leads to the particular sticky IP exhaustion. This can be addressed if you ensure that traffic is evenly distributed across cluster nodes. For example, on an FPR41xx cluster, tweak the load balancing algorithm on connected switches. On an FPR9300 cluster, ensure an equal number of blades across the chassis.

Reason-2: PAT pool usage is really high which leads to frequent exhaustion of the pool. To address this increase the PAT pool size.

Q. How is the support for the extended keyword handled? Does it show an error, and prevents the whole NAT command to be added during the upgrade, or does it remove the extended keyword, and shows a warning?

A. PAT extended option is not supported in Cluster from ASA 9.15.1/FP 6.7 onwards. The configuration option is not removed from any of CLI/ASDM/CSM/FMC. When configured (directly or indirectly through an upgrade), you are notified with a warning message, and the configuration is accepted but you do not see the extended functionality of PAT in action.

Q. Is it the same number of translations as concurrent connections?

A. In pre-6.7/9.15.1, although it was 1-65535, as the source ports are never be used much in the range 1-1024, it effectively makes it 1024-65535 (64512 conns). In the post-6.7/9.15.1 implementation with 'flat' as default behavior, it is 1024-65535. But if you want to use the 1-1024, you can with the "include-reserve" option.

Q. If the node joins the cluster back, it has the old backup node as backup and that backup node gives its old port block to it?

A. It depends on the availability of port blocks at that time. When a node leaves the cluster, all its port blocks are moved to the backup node. It is then the control node that accumulates free port blocks and distributes them to the required nodes.

Q. If there is a change in the control nodeâ€™s state, a new control node elected, is the PAT block allocation be maintained, or the port blocks are reallocated based on the new control node?

A. The new control node has an understanding of what blocks have been allocated and which are free and starts from there.

Q. Is the maximum number of xlates the same as the max number of concurrent connections with this new behavior?

A. Yes. The max number of xlates is dependent on the availability of PAT ports. It has nothing to do with the max number of concurrent connections. If you only allow 1 address, you have 65535 possible connections. If you need more, you have to allocate more IP addresses. If there are enough addresses/ports, you can reach max concurrent connections.

Q. What is the process of the port block allocation when a new cluster member is added? What happens if a cluster member is added due to reboot?

A. Port blocks are always distributed by the control node. Port blocks are allocated to a new node only when there are free port blocks. Free port blocks mean that no connection is served through any mapped port within the port block.

Further, upon rejoin, each node recalculates the number of blocks it can own. If a node holds more blocks than it is supposed to, it releases such additional port blocks to the control node as and when they become available. The control node then allocates them to the newly joined data node.

Q. Is it supported only TCP and UDP protocols or SCTP as well?

A. SCTP was never supported with dynamic PAT. For SCTP traffic, the recommendation is to use a static network object NAT only.

Q. If a node runs out of block ports, does it drop packets and not use the next available IP block?

A. No, it does not drop immediately. It uses available port blocks from the next PAT IP. If all the port blocks across all the PAT IPs are exhausted, then it drops traffic.

Q. To avoid the overload of the control node in a cluster upgrade window, is it be better to elect a new control manually earlier (for example, halfway through a 4-unit cluster upgrade), rather than wait for all connections to be handled on the control node?

A. The control must be updated last. This is because, when the control node runs the newer version it does not initiate pool distribution unless all nodes run the newer version. Additionally, when an upgrade runs, all data nodes with a newer version ignore pool distribution messages from a control node if it runs an older version.

To explain this in detail, consider a cluster deployment with 4 nodes A, B, C, and D with A as control. Here are the typical hitless upgrade steps:

1. Download a new version onto each of the nodes.
2. Reload unit â€˜Dâ€™. All connections, xlates are moved to the backup node.
3. Unit â€˜Dâ€™ comes up and:

a. Processes PAT configuration

b. Breaks each PAT IP into port blocks

c. Has all port blocks in unassigned state

d. Ignores older version of cluster PAT messages received from control

e. Redirects all PAT connections to Primary.

4. Similarly, bring up other nodes with the new version.

5. Reload unit â€˜Aâ€™ control. Since there is no backup for control, all existing connections are dropped

6. The new control starts the distribution of port blocks in the newer format

7. Unit â€˜Aâ€™ rejoins and is able to accept and act on port-block distribution messages

Fragment Handling

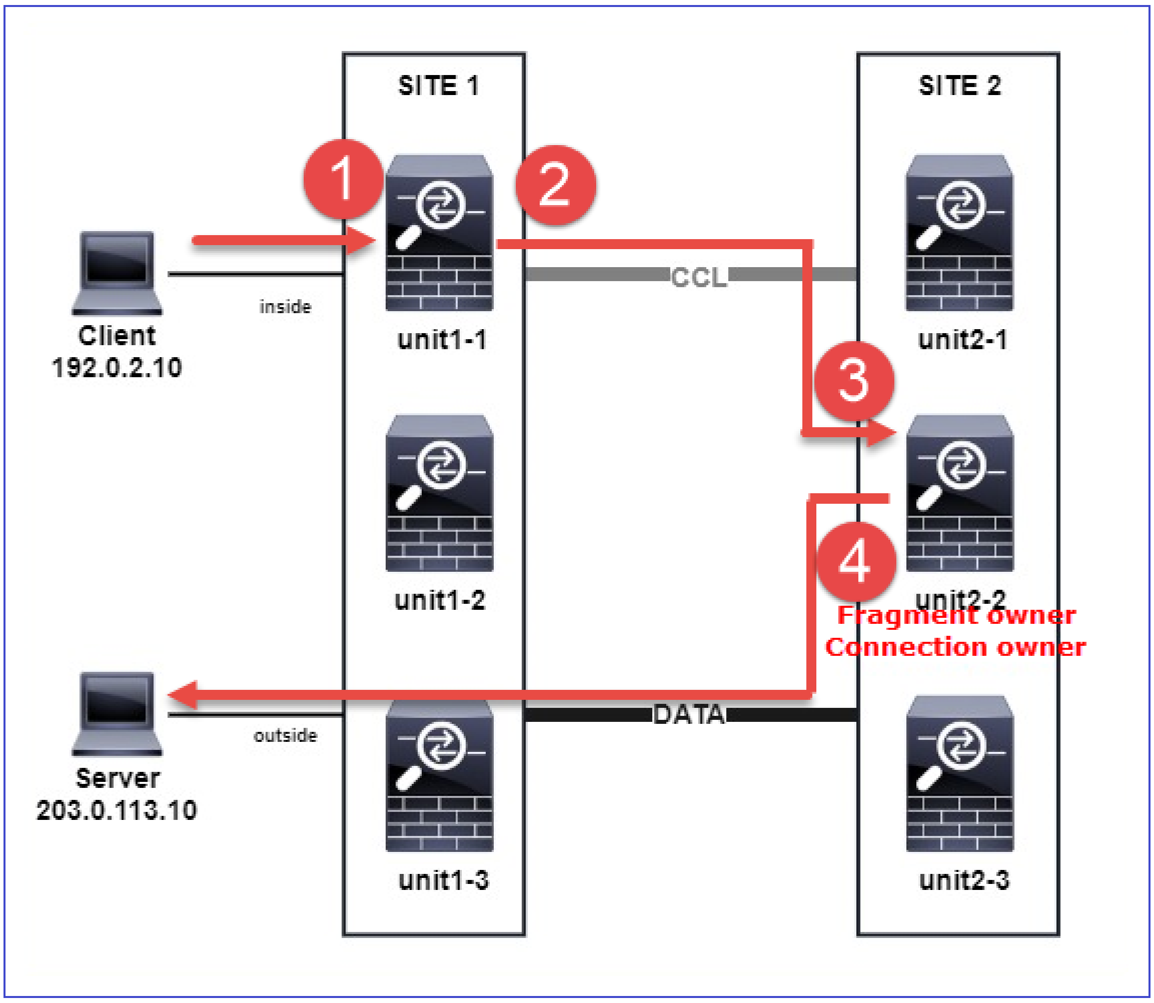
Symptom

In inter-site cluster deployments fragmented packets that must be handled in 1 specific site (site-local traffic), can still be sent to the units in other sites, as one of these sites can have the fragment owner.

In cluster logic, there is an additional role defined for connections with fragmented packets: fragment owner.

For fragmented packets, cluster units that receive a fragment determine a fragment owner based on a hash of the fragment source IP address, destination IP address, and the packet ID. All fragments are then forwarded to the fragment owner over the cluster control link. Fragments can be load-balanced to different cluster units because only the first fragment includes the 5-tuple used in the switch load-balance hash. Other fragments do not contain the source and destination ports and can be load-balanced to other cluster units. The fragment owner temporarily reassembles the packet so it can determine the director based on a hash of the source/destination IP address and ports. If it is a new connection, the fragment owner becomes the connection owner. If it is an existing connection, the fragment owner forwards all fragments to the connection owner over the cluster control link. The connection owner then reassembles all fragments.

Consider this topology with the flow of a fragmented ICMP echo request from the client to the server:



In order to understand the order of operations, there are cluster-wide packet captures on the inside, outside, and cluster control link interfaces configured with the trace option. Additionally, a packet capture with the reinject-hide option is configured on the inside interface.

```
<#root>

firepower#

cluster exec capture capi interface inside trace match icmp any any


firepower#

cluster exec capture capir interface inside reinject-hide trace match icmp any any


firepower#

cluster exec capture capo interface outside trace match icmp any any


firepower#

cluster exec capture capccl interface cluster trace match icmp any any
```

Order of operations within the cluster:

1. unit-1-1 in site 1 receives the fragmented ICMP echo request packets.

<#root>

```
firepower#

cluster exec show cap capir


unit-1-1(LOCAL)

:******************************************************

2 packets captured


1: 20:13:58.227801 802.1Q vlan#10 P0 192.0.2.10 > 203.0.113.10 icmp: echo request




2: 20:13:58.227832 802.1Q vlan#10 P0


2 packets shown
```

2. unit-1-1 selects unit-2-2 in site 2 as the fragment owner and sends fragmented packets to it.
The destination MAC address of the packets sent from unit-1-1 to unit-2-2 is the MAC address of the CCL link in unit-2-2.

<#root>

```
firepower#

show cap capccl packet-number 1 detail



7 packets captured

1: 20:13:58.227817

0015.c500.018f 0015.c500.029f

 0x0800 Length: 1509


192.0.2.10 > 203.0.113.10

 icmp: echo request (wrong icmp csum) (frag 46772:1475@0+) (ttl 3)
1 packet shown

firepower#

show cap capccl packet-number 2 detail



7 packets captured
```

```
2: 20:13:58.227832
```

0015.c500.018f 0015.c500.029f

```
 0x0800 Length: 637
```

192.0.2.10 > 203.0.113.10

```
 (
```

frag 46772

```
:603@1480) (ttl 3)
1 packet shown
```

```
firepower#
```

cluster exec show interface po48 | i MAC

```
unit-1-1(LOCAL):******************************************************
MAC address 0015.c500.018f, MTU 1500
unit-1-2:*************************************************************
MAC address 0015.c500.019f, MTU 1500
```

unit-2-2

```
:*************************************************************
```

MAC address 0015.c500.029f, MTU 1500

```
unit-1-3:*************************************************************
MAC address 0015.c500.016f, MTU 1500
unit-2-1:*************************************************************
MAC address 0015.c500.028f, MTU 1500
unit-2-3:*************************************************************
MAC address 0015.c500.026f, MTU 1500
```

3. unit-2-2 receives, reassembles the fragmented packets, and becomes the owner of the flow.

<#root>

```
firepower#
```

cluster exec unit unit-2-2 show capture capccl packet-number 1 trace

```
11 packets captured

1: 20:13:58.231845 192.0.2.10 > 203.0.113.10 icmp: echo request
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
```

```
Input interface: 'inside'
Flow type: NO FLOW
```

```
I (2) received a FWD_FRAG_TO_FRAG_OWNER from (0).
```

```
Phase: 2
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'inside'
```

```
Flow type: NO FLOW
```

```
I (2) have reassembled a packet and am processing it.
```

```
Phase: 3
Type: CAPTURE
Subtype:
Result: ALLOW
Config:
Additional Information:
MAC Access list

Phase: 4
Type: ACCESS-LIST
Subtype:
Result: ALLOW
Config:
Implicit Rule
Additional Information:
MAC Access list

Phase: 5
Type: ROUTE-LOOKUP
Subtype: No ECMP load balancing
Result: ALLOW
Config:
Additional Information:
Destination is locally connected. No ECMP load balancing.
Found next-hop 203.0.113.10 using egress ifc outside(vrfid:0)

Phase: 6
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'inside'
```

```
Flow type: NO FLOW
```

```
I (2) am becoming owner



Phase: 7
Type: ACCESS-LIST
Subtype: log
Result: ALLOW
Config:
access-group CSM_FW_ACL_ global
access-list CSM_FW_ACL_ advanced trust ip any any rule-id 268435460 event-log flow-end
access-list CSM_FW_ACL_ remark rule-id 268435460: PREFILTER POLICY: igasimov_prefilter1
access-list CSM_FW_ACL_ remark rule-id 268435460: RULE: r1
Additional Information:

...

Phase: 19
Type: FLOW-CREATION
Subtype:
Result: ALLOW
Config:
Additional Information:
New flow created with id 1719, packet dispatched to next module

...

Result:
input-interface: cluster(vrfid:0)
input-status: up
input-line-status: up
output-interface: outside(vrfid:0)
output-status: up
output-line-status: up


Action: allow




1 packet shown
firepower#

cluster exec unit unit-2-2 show capture capccl packet-number 2 trace



11 packets captured

2: 20:13:58.231875
Phase: 1
Type: CLUSTER-EVENT
Subtype:
Result: ALLOW
Config:
Additional Information:
Input interface: 'inside'


Flow type: NO FLOW
```

```
I (2) received a FWD_FRAG_TO_FRAG_OWNER from (0).



Result:
input-interface: cluster(vrfid:0)
input-status: up
input-line-status: up
Action: allow

1 packet shown
```

4. unit-2-2 allows the packets based on the security policy and sends them, via the outside interface, from site 2 to site 1.

<#root>

```
firepower#

cluster exec unit unit-2-2 show cap capo



2 packets captured



1: 20:13:58.232058 802.1Q vlan#20 P0 192.0.2.10 > 203.0.113.10 icmp: echo request




2: 20:13:58.232058 802.1Q vlan#20 P0
```

Observations/Caveats

- Unlike the director role, the fragment owner cannot be localized within a particular site. The fragment owner is determined by the unit that originally receives the fragmented packets of a new connection and can be located at any site.
- Since a fragment owner can also become the connection owner, then in order to forward the packets to the destination host, it must be able to resolve the egress interface, and find the IP and MAC addresses of the destination host or the next hop. This presumes that the next-hop(s) must also have the reachability to the destination host.
- To reassemble the fragmented packets the ASA/FTD maintains an IP fragment reassembly module for each named interface. To display the operational data of the IP fragment reassembly module, use the **show fragment** command:
  <#root>

  ```
  Interface: inside
  Configuration:

  Size: 200

  , Chain: 24, Timeout: 5, Reassembly: virtual
  ```

```
    Run-time stats: Queue: 0, Full assembly: 0
    Drops: Size overflow: 0, Timeout: 0,
    Chain overflow: 0, Fragment queue threshold exceeded: 0,
    Small fragments: 0, Invalid IP len: 0,
    Reassembly overlap: 0, Fraghead alloc failed: 0,
    SGT mismatch: 0, Block alloc failed: 0,
    Invalid IPV6 header: 0, Passenger flow assembly failed: 0
```

In cluster deployments, the fragment owner or the connection owner put the fragmented packets into the fragment queue. The fragment queue size is limited by the value of the Size counter (by default 200) that is configured with the **fragment size <size> <nameif>** command. When the fragment queue size reaches 2/3 of the Size, the fragment queue threshold is considered to be exceeded, and any new fragments that are not part of the current fragment chain are dropped. In this case, the Fragment queue threshold exceeded is incremented, and syslog message FTD-3-209006 is generated.

<#root>

```
firepower#

show fragment inside


Interface: inside

    Configuration:

Size: 200

, Chain: 24, Timeout: 5, Reassembly: virtual
    Run-time stats:

Queue: 133

, Full assembly: 0
    Drops: Size overflow: 0, Timeout: 8178,
           Chain overflow: 0,

Fragment queue threshold exceeded: 40802

,
           Small fragments: 0, Invalid IP len: 0,
           Reassembly overlap: 9673, Fraghead alloc failed: 0,
           SGT mismatch: 0, Block alloc failed: 0,
           Invalid IPV6 header: 0, Passenger flow assembly failed: 0



%FTD-3-209006: Fragment queue threshold exceeded, dropped TCP fragment from 192.0.2.10/21456 to 203.0.11
```

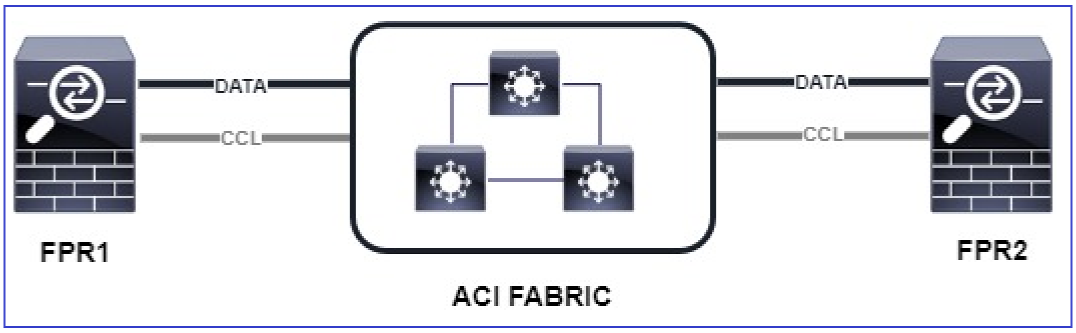
As a workaround, increase the size in **Firepower Management Center** > **Devices** > **Device Management** > **[Edit Device]** > **Interfaces** > **[Interface]** > **Advanced** > **Security Configuration** > **Override Default Fragment Setting**, **save** configuration and deploy policies. Then monitor the Queue counter in the **show fragment** command output and the occurrence of syslog message FTD-3-209006.

ACI Issues

Intermittent connectivity issues through the cluster due to active L4 checksum verification in ACI Pod

Symptom

- Intermittent connectivity issues through the ASA/FTD cluster deployed in an ACI Pod.
- If there is only 1 unit in the cluster, the connectivity issues are not observed.
- Packets sent from one cluster unit to one or more other units in the cluster are not visible in the FXOS and data plane captures of the target units.



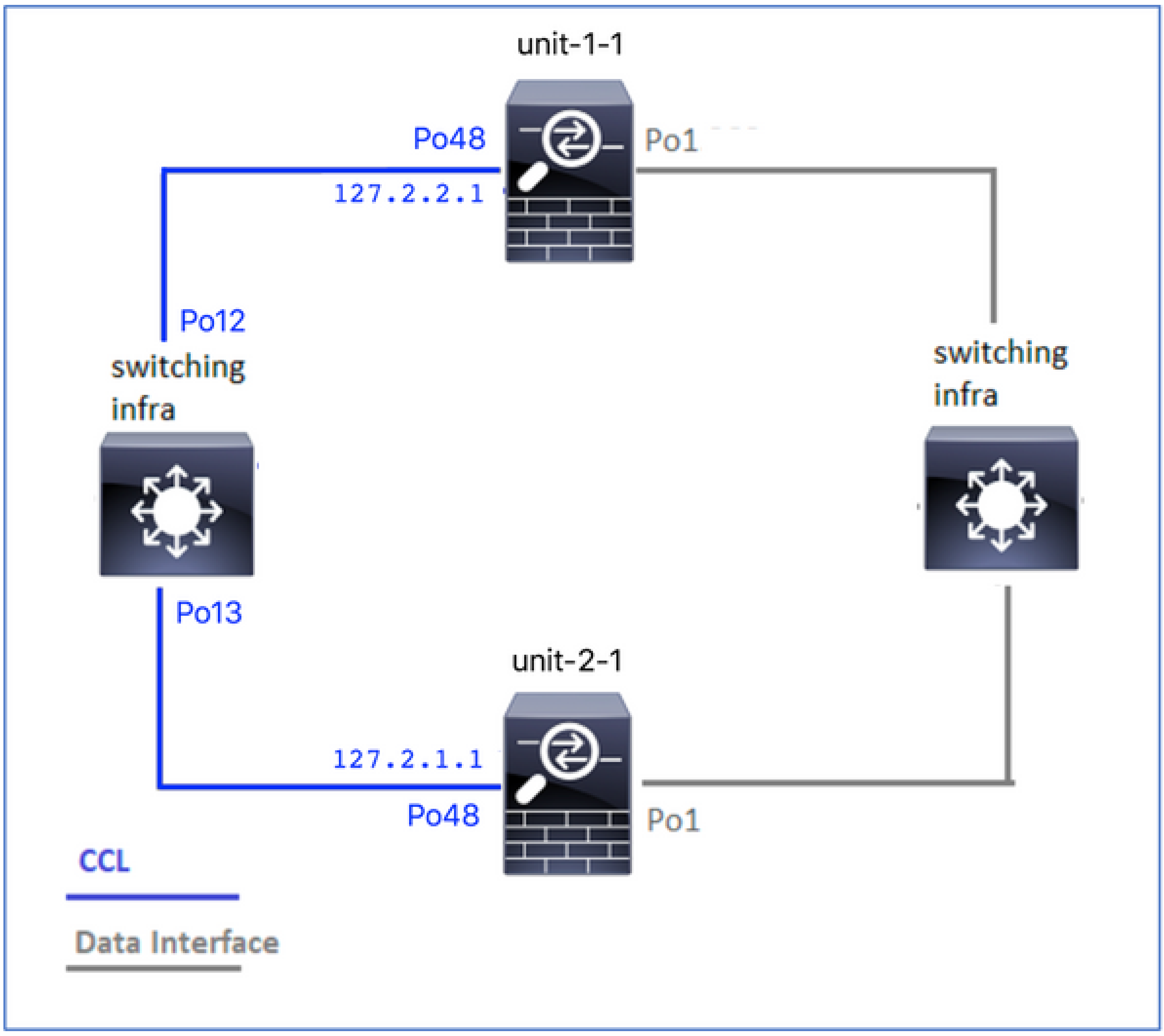
Mitigation

- Redirected traffic over the cluster control link does not have a correct L4 checksum and this is expected behavior. Switches on the cluster control link path must not verify the L4 checksum. Switches that verify the L4 checksum can cause traffic to be dropped. Check the ACI fabric switch configuration and ensure no L4 checksum is performed on the received or sent packets via the cluster control link.

Cluster Control Plane Issues

Unit Cannot Join the Cluster

MTU Size on CCL



Symptoms

The unit cannot join the cluster and this message is shown:

```
The SECONDARY has left the cluster because application configuration sync is timed out on this unit. Dis
Cluster disable is performing cleanup..done.
Unit unit-2-1 is quitting due to system failure for 1 time(s) (last failure is SECONDARY application con
All data interfaces have been shutdown due to clustering being disabled. To recover either enable cluste
```

Verification/Mitigation

- Use the **show interface** command on the FTD, to verify that the MTU on the cluster control link interface is at least 100 bytes higher than the data interface MTU:

<#root>

```
firepower#

show interface


Interface
```

```
Port-channel1

 "

Inside

", is up, line protocol is up
  Hardware is EtherSVI, BW 40000 Mbps, DLY 10 usec
    MAC address 3890.a5f1.aa5e,

MTU 9084


Interface

 Port-channel48

 "

cluster

", is up, line protocol is up
  Hardware is EtherSVI, BW 40000 Mbps, DLY 10 usec
    Description: Clustering Interface
    MAC address 0015.c500.028f,

MTU 9184


    IP address 127.2.2.1, subnet mask 255.255.0.
```

- Do a ping through the CCL, with the size option, to verify if configured on the CCL MTU is correctly configured on all devices in the path.

<#root>

```
firepower#

ping 127.2.1.1 size 9184
```

- Use the **show interface** command on the switch to verify the MTU configuration

<#root>

```
Switch#

show interface



port-channel12

 is up
admin state is up,
  Hardware: Port-Channel, address: 7069.5a3a.7976 (bia 7069.5a3a.7976)


MTU 9084
```

```
bytes, BW 40000000 Kbit , DLY 10 usec
```

```
port-channel13
```

```
 is up
admin state is up,
  Hardware: Port-Channel, address: 7069.5a3a.7967 (bia 7069.5a3a.7967)
```

```
MTU 9084
```

```
bytes, BW 40000000 Kbit , DLY 10 use
```

Interface Mismatch Between Cluster Units

Symptoms

The unit cannot join the cluster and this message is shown:

```
Interface mismatch between cluster primary and joining unit unit-2-1. unit-2-1 aborting cluster join.
Cluster disable is performing cleanup..done.
Unit unit-2-1 is quitting due to system failure for 1 time(s) (last failure is Internal clustering error
All data interfaces have been shutdown due to clustering being disabled. To recover either enable cluste
```

Verification/Mitigation

Log in to the FCM GUI on each chassis, navigate to the **Interfaces** tab, and verify if all cluster members have the same interface configuration:

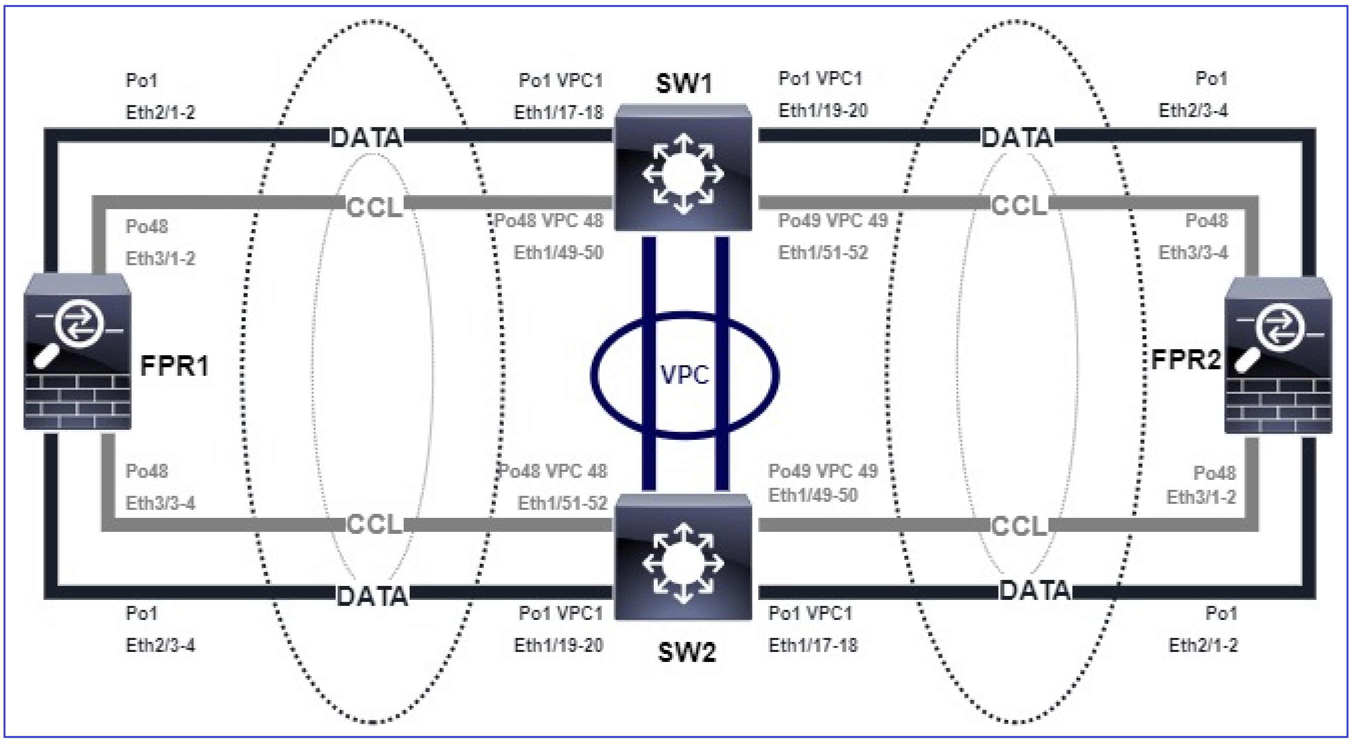
- Interfaces assigned to the logical device
- Admin speed of the interfaces
- Admin duplex of the interfaces
- Interface status

Data/Port-Channel Interface Issue

Split-brain due to Reachability Issues Over the CCL

Symptom

There are multiple control units in the cluster. Consider this topology:



Chassis 1:

<#root>

```
firepower# show cluster info


Cluster ftd_cluster1: On
Interface mode: spanned


This is "unit-1-1" in state PRIMARY


ID : 0
Site ID : 1
Version : 9.15(1)
Serial No.: FLM2103TU5H
CCL IP : 127.2.1.1
CCL MAC : 0015.c500.018f
Last join : 07:30:25 UTC Dec 14 2020
Last leave: N/A
Other members in the cluster:
Unit "unit-1-2" in state SECONDARY
ID : 1
Site ID : 1
Version : 9.15(1)
Serial No.: FLM2103TU4D
CCL IP : 127.2.1.2
CCL MAC : 0015.c500.019f
Last join : 07:30:26 UTC Dec 14 2020
Last leave: N/A
Unit "unit-1-3" in state SECONDARY
ID : 3
Site ID : 1
Version : 9.15(1)
Serial No.: FLM2102THJT
```

```
CCL IP : 127.2.1.3
CCL MAC : 0015.c500.016f
Last join : 07:31:49 UTC Dec 14 2020
Last leave: N/A
```

Chassis 2:

<#root>

```
firepower# show cluster info
```

```
Cluster ftd_cluster1: On
Interface mode: spanned
```

```
This is "unit-2-1" in state PRIMARY
```

```
ID : 4
Site ID : 1
Version : 9.15(1)
Serial No.: FLM2103TUN1
CCL IP : 127.2.2.1
CCL MAC : 0015.c500.028f
Last join : 11:21:56 UTC Dec 23 2020
Last leave: 11:18:51 UTC Dec 23 2020
Other members in the cluster:
Unit "unit-2-2" in state SECONDARY
ID : 2
Site ID : 1
Version : 9.15(1)
Serial No.: FLM2102THR9
CCL IP : 127.2.2.2
CCL MAC : 0015.c500.029f
Last join : 11:18:58 UTC Dec 23 2020
Last leave: 22:28:01 UTC Dec 22 2020
Unit "unit-2-3" in state SECONDARY
ID : 5
Site ID : 1
Version : 9.15(1)
Serial No.: FLM2103TUML
CCL IP : 127.2.2.3
CCL MAC : 0015.c500.026f
Last join : 11:20:26 UTC Dec 23 2020
Last leave: 22:28:00 UTC Dec 22 2020
```

Verification

- Use the **ping** command to verify connectivity between the cluster control link (CCL) IP addresses of the control units:

<#root>

```
firepower# ping 127.2.1.1
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 127.2.1.1, timeout is 2 seconds:
```

```
?????
Success rate is 0 percent (0/5)
```

- Check the ARP table:

<#root>

```
firepower# show arp
```

```
cluster 127.2.2.3 0015.c500.026f 1
cluster 127.2.2.2 0015.c500.029f 1
```

- In control units configure and check captures on the CCL interfaces:

<#root>

```
firepower# capture capccl interface cluster
firepower# show capture capccl | i 127.2.1.1
```

```
2: 12:10:57.652310 arp who-has 127.2.1.1 tell 127.2.2.1
41: 12:11:02.652859 arp who-has 127.2.1.1 tell 127.2.2.1
74: 12:11:07.653439 arp who-has 127.2.1.1 tell 127.2.2.1
97: 12:11:12.654018 arp who-has 127.2.1.1 tell 127.2.2.1
126: 12:11:17.654568 arp who-has 127.2.1.1 tell 127.2.2.1
151: 12:11:22.655148 arp who-has 127.2.1.1 tell 127.2.2.1
174: 12:11:27.655697 arp who-has 127.2.1.1 tell 127.2.2.1
```

Mitigation

- Ensure that the CCL port-channel interfaces are connected to separate port-channel interfaces on the switch.
- When virtual port-channels (vPC) are used on Nexus switches, ensure that CCL port-channel interfaces are connected to different vPC and that vPC configuration does not have failed consistency status.
- Ensure that the CCL port-channel interfaces are in the same broadcast domain and that the CCL VLAN is created and allowed on the interfaces.

This is a sample switch configuration:

<#root>

```
Nexus#
```

```
show run int po48-49
```

```
interface port-channel48
description FPR1

switchport access vlan 48



vpc 48



interface port-channel49
description FPR2

switchport access vlan 48



vpc 49




Nexus#
 show vlan id 48



VLAN Name Status Ports
---- ----------- --------- -------------------------------


48   CCL active Po48, Po49, Po100, Eth1/53, Eth1/54



VLAN Type Vlan-mode
---- ----- ----------
48 enet CE

1  Po1  up success success 10,20

48 Po48 up success success 48



49 Po49 up success success 48
```

<#root>

```
Nexus1#

show vpc brief


Legend:
(*) - local vPC is down, forwarding via vPC peer-link
```

```
vPC domain id : 1
Peer status : peer adjacency formed ok
vPC keep-alive status : peer is alive
Configuration consistency status : success
```

```
Per-vlan consistency status : success
```

```
Type-2 consistency status : success
```

```
vPC role : primary
Number of vPCs configured : 3
Peer Gateway : Disabled
Dual-active excluded VLANs : -
Graceful Consistency Check : Enabled
Auto-recovery status : Disabled
Delay-restore status : Timer is off.(timeout = 30s)
Delay-restore SVI status : Timer is off.(timeout = 10s)

vPC Peer-link status
---------------------------------------------------------------------
id Port Status Active vlans
-- ---- ------ --------------------------------------------------
1 Po100 up 1,10,20,48-49,148

vPC status
----------------------------------------------------------------------
id Port Status Consistency Reason Active vlans
-- ---- ------ ----------- ------ ------------
1 Po1 up success success 10,20
```

```
48 Po48 up success success 48
```

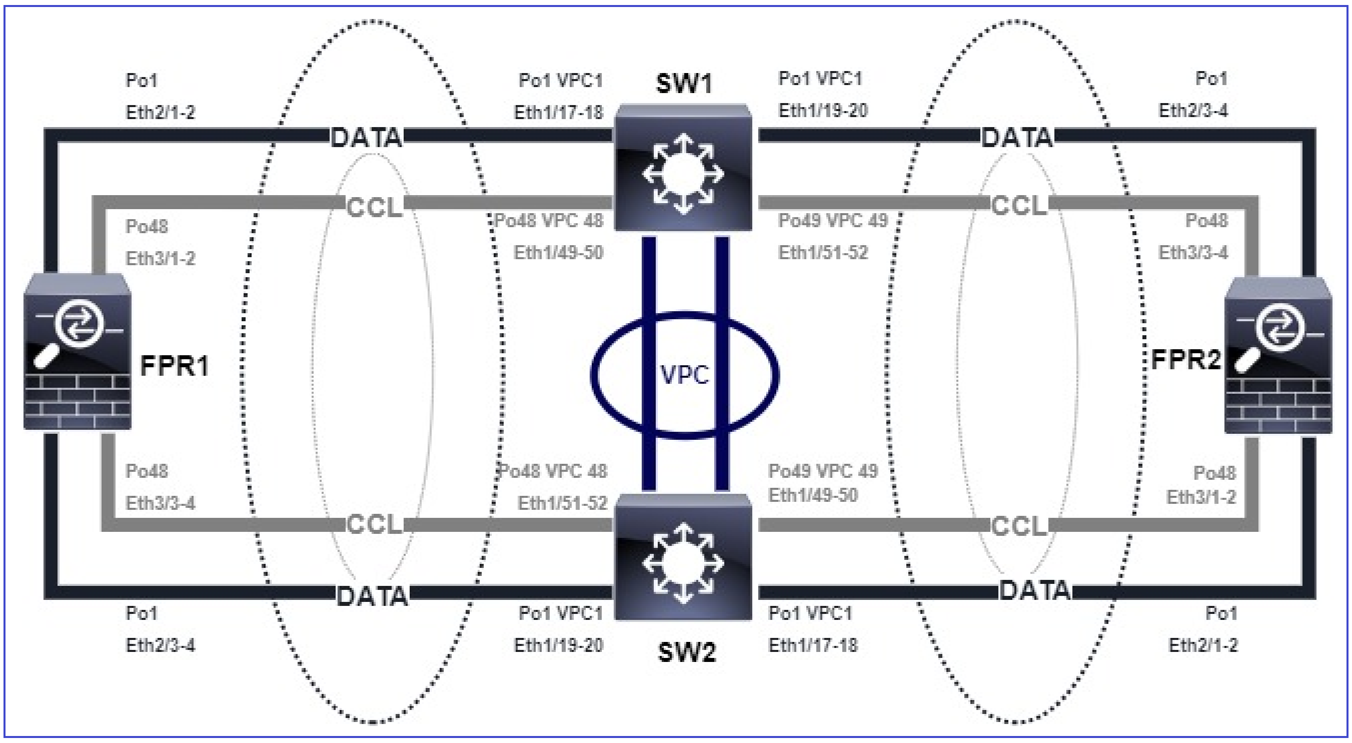
```
49 Po49 up success success 48
```

Disabled Cluster due to Suspended Data Port-channel Interfaces

Symptom

One or more data port-channel interfaces are suspended. When an administratively enabled data interface is suspended, all the cluster units in the same chassis are kicked out of the cluster because of interface health check failure.

Consider this topology:



Verification

- Check the control unit console:

<#root>

```
firepower#
Beginning configuration replication to
```

```
SECONDARY unit-2-2
```

```
End Configuration Replication to SECONDARY.
Asking SECONDARY unit
```

```
unit-2-2
```

```
 to quit because it
```

```
failed interface health
```

```
 check 4 times (last failure on
```

```
Port-channel1
```

```
). Clustering must be manually enabled on the unit to rejoin.
```

- Check the output of the **show cluster history** and the **show cluster info trace module hc** commands in the affected unit(s):

<#root>

```
firepower# Unit is kicked out from cluster because of interface health check failure.
Cluster disable is performing cleanup..done.
All data interfaces have been shutdown due to clustering being disabled. To recover either enable cluste
```

```
Cluster unit unit-2-1 transitioned from SECONDARY to DISABLED
```

```
firepower#
 show cluster history

==========================================================================
From State To State Reason
==========================================================================

12:59:37 UTC Dec 23 2020
ONCALL SECONDARY_COLD Received cluster control message

12:59:37 UTC Dec 23 2020
SECONDARY_COLD SECONDARY_APP_SYNC Client progression done

13:00:23 UTC Dec 23 2020
SECONDARY_APP_SYNC SECONDARY_CONFIG SECONDARY application configuration sync done

13:00:35 UTC Dec 23 2020
SECONDARY_CONFIG SECONDARY_FILESYS Configuration replication finished

13:00:36 UTC Dec 23 2020
SECONDARY_FILESYS SECONDARY_BULK_SYNC Client progression done


13:01:35 UTC Dec 23 2020


SECONDARY_BULK_SYNC DISABLED Received control message DISABLE (interface health check failure)
```

<#root>

```
firepower#
 show cluster info trace module hc

Dec 23 13:01:36.636 [INFO]cluster_fsm_clear_np_flows: The clustering re-enable timer is started to expir
Dec 23 13:01:32.115 [INFO]cluster_fsm_disable: The clustering re-enable timer is stopped.

Dec 23 13:01:32.115 [INFO]Interface Port-channel1 is down
```

- Check the output of the **show port-channel summary** command in the **fxos** command shell:

<#root>

```
FPR2(fxos)#
 show port-channel summary

Flags: D - Down P - Up in port-channel (members)
```

```
I - Individual H - Hot-standby (LACP only)
s - Suspended r - Module-removed
S - Switched R - Routed
U - Up (port-channel)
M - Not in use. Min-links not met
--------------------------------------------------------------------------
Group Port-Channel Type Protocol Member Ports

--------------------------------------------------------------------------


1 Po1(SD) Eth LACP Eth2/1(s) Eth2/2(s) Eth2/3(s) Eth2/4(s)


48 Po48(SU) Eth LACP Eth3/1(P) Eth3/2(P) Eth3/3(P) Eth3/4(P)
```

Mitigation

- Ensure that all chassis have the same cluster group name and password.
- Ensure that the port-channel interfaces have administratively enabled physical member interfaces with the same duplex/speed configuration in all chassis and switches.
- In intra-site clusters ensure that the same data port-channel interface in all chassis is connected to the same port-channel interface on the switch.
- When virtual port-channels (vPC) are used in Nexus switches, ensure that vPC configuration does not have failed consistency status.
- In intra-site clusters ensure that the same data port-channel interface in all chassis is connected to the same vPC.

Cluster Stability Issues

FXOS Traceback

Symptom

Unit leaves the cluster.

Verification/Mitigation

- Use the **show cluster history** command to see when the unit left the cluster

<#root>

```
firepower#

show cluster history
```

- Use these commands to check if the FXOS had a traceback

<#root>

```
FPR4150#

connect local-mgmt
```

```
FPR4150 (local-mgmt)#
```

```
dir cores
```

- Collect the core file generated around the time when the unit left the cluster and provide it to TAC.

Disk Full

In case the disk utilization in the /ngfw partition of a cluster unit reaches 94% the unit quits the cluster. The disk utilization check occurs every 3 seconds:

<#root>

```
> show disk


Filesystem Size Used Avail Use% Mounted on
rootfs 81G 421M 80G 1% /
devtmpfs 81G 1.9G 79G 3% /dev
tmpfs 94G 1.8M 94G 1% /run
tmpfs 94G 2.2M 94G 1% /var/volatile
/dev/sda1 1.5G 156M 1.4G 11% /mnt/boot
/dev/sda2 978M 28M 900M 3% /opt/cisco/config
/dev/sda3 4.6G 88M 4.2G 3% /opt/cisco/platform/logs
/dev/sda5 50G 52M 47G 1% /var/data/cores
/dev/sda6 191G 191G 13M

100% /ngfw


cgroup_root 94G 0 94G 0% /dev/cgroups
```

In this case, the show cluster history output shows:

<#root>

```
15:36:10 UTC May 19 2021
PRIMARY Event: Primary unit unit-1-1 is quitting
                    due to

diskstatus

 Application health check failure, and
                    primary's application state is down
```

or

```
14:07:26 CEST May 18 2021
SECONDARY DISABLED Received control message DISABLE (application health check failure)
```

Another way to verify the failure is:

<#root>

```
firepower#

show cluster info health


Member ID to name mapping:
0 - unit-1-1(myself) 1 - unit-2-1

               0   1
Port-channel48 up up
Ethernet1/1 up up
Port-channel12 up up
Port-channel13 up up

Unit overall            healthy healthy
Service health status:
                        0       1


diskstatus (monitor on) down    down


snort (monitor on)      up      up
Cluster overall         healthy
```

Additionally, if the disk is ~100% the unit can have difficulties to join back the cluster until there is some disk space released.

Overflow Protection

Every 5 minutes each cluster unit checks the local and the peer unit for CPU and memory utilization. If the utilization is higher than the system thresholds (LINA CPU 50% or LINA memory 59%) an informational message is shown in:

- Syslogs (FTD-6-748008)
- File log/cluster_trace.log, for example:

<#root>

```
firepower#

more log/cluster_trace.log | i CPU


May 20 16:18:06.614 [INFO][

CPU load 87%

 | memory load 37%] of module 1 in chassis 1 (unit-1-1) exceeds overflow protection threshold [

CPU 50% | Memory 59%

]. System may be oversubscribed on member failure.
May 20 16:18:06.614 [INFO][CPU load 87% | memory load 37%] of chassis 1 exceeds overflow protection thre
May 20 16:23:06.644 [INFO][CPU load 84% | memory load 35%] of module 1 in chassis 1 (unit-1-1) exceeds o
```

The message denotes that in case of a unit failure the other unit(s) resources can be oversubscribed.

Simplified Mode

Behavior on pre-6.3 FMC releases

- You register each cluster node individually on FMC.
- Then you form a logical cluster in FMC.
- For every new cluster node addition, you must manually register the node.

Post-6.3 FMC

- The simplified mode feature allows you to register the whole cluster on FMC in one step (just register any one node of the cluster).

Minimum Supported Manager	**Managed Devices**	**Min Supported Managed Device Version Required**	**Notes**
FMC 6.3	FTD clusters on FP9300 and FP4100 only	6.2.0	This is an FMC feature only

Warning: Once the cluster is formed on FTD you need to wait for auto-registration to kick off. You must not try to register the cluster nodes manually (Add Device) but use the Reconcile option.

Symptom

Node registration failures

- If the control node registration fails for any reason, then the cluster is deleted from FMC.

Mitigation

If data node registration fails for any reason, there are 2 options:

1. With every deployment to the cluster, FMC checks if there are cluster nodes that need to be registered and then starts the auto-registration for these nodes.
2. There is a **Reconcile** option available under the cluster summary tab (**Devices** > **Device Management** > **Cluster tab** > **View Cluster Status** link). Once the Reconcile action is triggered, FMC starts auto-registration of the nodes that need to be registered.

Related Information

- Clustering for the Firepower Threat Defense
- ASA Cluster for the Firepower 4100/9300 Chassis
- About Clustering on the Firepower 4100/9300 Chassis
- Firepower NGFW Clustering Deep Dive - BRKSEC-3032

- Analyze Firepower Firewall Captures to Effectively Troubleshoot Network Issues