

# Troubleshoot vEdge Bidirectional Forwarding Detection and Data Plane Connections Issues

## Contents

---

### [Introduction](#)

### [Prerequisites](#)

[Requirements](#)

[Components Used](#)

### [Control Plane Information](#)

[Check Control Local Properties](#)

[Check Control Connections](#)

### [Overlay Management Protocol](#)

[Verify OMP TLOCs are Advertised from the vEdges](#)

[Verify vSmart Receives and Advertises the TLOCs](#)

### [Bidirectional Forwarding Detection](#)

[Understand the show bfd sessions command](#)

[Command show tunnel statistics](#)

### [Access List](#)

### [Network Address Translation](#)

[How to Use tools stun-client to Detect NAT Maps and Filters.](#)

[Supported NAT Types for Data Plane Tunnels“Sending” used in CLI](#)

### [Firewalls](#)

### [Security](#)

### [ISP Problems with DSCP Marked Traffic](#)

### [Debug BFD](#)

### [Related Information](#)

---

## Introduction

This document describes vEdge data plane connection problems after a control plane connection however no data plane connectivity between sites.

## Prerequisites

### Requirements

Cisco recommends knowledge of **Cisco Software Defined Wide Area Network (SDWAN)** solution.

### Components Used

This document is not restricted to specific software and hardware versions. This document is focused on vEdge platforms.




```
-----
vsmart dtls 10.1.0.3      3      1      203.0.113.13      12446 203.0.113.1
vbond dtls -            0      0      203.0.113.12      12346 203.0.113.1
vmanage dtls 10.1.0.1    1      0      203.0.113.14      12646 203.0.113.1
```

If the interface (that does not form data tunnels) tries to connect, solve it with a successful start-up of the control connections via that color.

Or, set the **max-control-connections 0** in the selected interface under the tunnel interface section.

```
vpn 0
interface ge0/1
 ip address 10.20.67.10/24
 tunnel-interface
  encapsulation ipsec
  color mpls restrict
  max-control-connections 0
  no allow-service bgp
  allow-service dhcp
  allow-service dns
  allow-service icmp
  no allow-service sshd
  no allow-service netconf
  no allow-service ntp
  no allow-service ospf
  no allow-service stun
!
no shutdown
!
```

---

 **Note:** Sometimes, you can use the **no control-connections** command to achieve the same goal. However, that command does not establish a maximum number of control connections. This command is deprecated from version 15.4 and is not used on newer software.

---

## Overlay Management Protocol

### Verify OMP TLOCs are Advertised from the vEdges

OMP TLOCs cannot be sent because the interface tries to form control connections via that color and is not able to reach the controllers.

Check if the color (that the data tunnels) sends the TLOC for that particular color to the vSmarts.

Use the command **show omp tlocs advertised** in order to check the TLOCs that are sent to the OMP peers.

Example: Colors **mpls** and **gold**. No TLOC is sent to vSmart for color mpls.

```
vEdge1# show omp tlocs advertised
C -> chosen
I -> installed
```

Red -> redistributed  
 Rej -> rejected  
 L -> looped  
 R -> resolved  
 S -> stale  
 Ext -> extranet  
 Stg -> staged  
 Inv -> invalid


| ADDRESS FAMILY | TLOC IP   | COLOR | ENCAP | FROM PEER | STATUS  | PSEUDO KEY | PUBLIC IP      | P |
|----------------|-----------|-------|-------|-----------|---------|------------|----------------|---|
| ipv4           | 10.1.0.5  | gold  | ipsec | 0.0.0.0   | C,Red,R | 1          | 203.0.113.225  | 4 |
|                | 10.1.0.2  | mpls  | ipsec | 10.1.0.3  | C,I,R   | 1          | 10.20.67.20    |   |
|                | 10.1.0.2  | blue  | ipsec | 10.1.0.3  | C,I,R   | 1          | 198.51.100.187 |   |
|                | 10.1.0.30 | mpls  | ipsec | 10.1.0.3  | C,I,R   | 1          | 10.20.67.30    |   |
|                | 10.1.0.30 | gold  | ipsec | 10.1.0.3  | C,I,R   | 1          | 192.0.2.129    |   |
|                | 10.1.0.4  | mpls  | ipsec | 10.1.0.3  | C,I,R   | 1          | 10.20.67.40    |   |
|                | 10.1.0.4  | gold  | ipsec | 10.1.0.3  | C,I,R   | 1          | 203.0.113.226  |   |

Example: Colors **mpls** and **gold**. TLOC is sent for both the colors.

```

vEdge2# show omp tlocs advertised
C -> chosen
I -> installed
Red -> redistributed
Rej -> rejected
L -> looped
R -> resolved
S -> stale
Ext -> extranet
Stg -> staged
Inv -> invalid
  
```

| ADDRESS FAMILY | TLOC IP   | COLOR | ENCAP | FROM PEER | STATUS  | PSEUDO KEY | PUBLIC IP      | P |
|----------------|-----------|-------|-------|-----------|---------|------------|----------------|---|
| ipv4           | 10.1.0.5  | gold  | ipsec | 10.1.0.3  | C,I,R   | 1          | 203.0.113.225  | 4 |
|                | 10.1.0.2  | mpls  | ipsec | 0.0.0.0   | C,Red,R | 1          | 10.20.67.20    | 1 |
|                | 10.1.0.2  | blue  | ipsec | 0.0.0.0   | C,Red,R | 1          | 198.51.100.187 | 1 |
|                | 10.1.0.30 | mpls  | ipsec | 10.1.0.3  | C,I,R   | 1          | 10.20.67.30    |   |
|                | 10.1.0.30 | gold  | ipsec | 10.1.0.3  | C,I,R   | 1          | 192.0.2.129    |   |
|                | 10.1.0.4  | mpls  | ipsec | 10.1.0.3  | C,I,R   | 1          | 10.20.67.40    |   |
|                | 10.1.0.4  | gold  | ipsec | 10.1.0.3  | C,I,R   | 1          | 203.0.113.226  |   |

 **Note:** For any locally generated control plane information, "FROM PEER" field is set to 0.0.0.0. When you look for locally originated information, ensure to match based on this value.

## Verify vSmart Receives and Advertises the TLOCs

TLOCs are now advertised out to the vSmart. Confirm that it receives TLOCs from the correct peer and advertises it to the other vEdge.

Example: vSmart receives the TLOCs from 10.1.0.2 vEdge1.

<#root>

vSmart1# show omp tlocs received

C -> chosen  
I -> installed  
  
Red -> redistributed  
Rej -> rejected  
L -> looped  
  
R -> resolved  
  
S -> stale  
Ext -> extranet  
Stg -> staged  
Inv -> invalid

| ADDRESS FAMILY | TLOC IP   | COLOR | ENCAP | FROM PEER | STATUS | PSEUDO KEY | PUBLIC IP      | P   |
|----------------|-----------|-------|-------|-----------|--------|------------|----------------|-----|
| ipv4           | 10.1.0.5  | gold  | ipsec | 10.1.0.5  | C,I,R  | 1          | 203.0.113.225  | 450 |
|                | 10.1.0.2  | mpls  | ipsec | 10.1.0.2  | C,I,R  | 1          | 10.20.67.20    | 12  |
|                | 10.1.0.2  | blue  | ipsec | 10.1.0.2  | C,I,R  | 1          | 198.51.100.187 | 12  |
|                | 10.1.0.30 | mpls  | ipsec | 10.1.0.30 | C,I,R  | 1          | 10.20.67.30    |     |
|                | 10.1.0.30 | gold  | ipsec | 10.1.0.30 | C,I,R  | 1          | 192.0.2.129    |     |
|                | 10.1.0.4  | mpls  | ipsec | 10.1.0.4  | C,I,R  | 1          | 10.20.67.40    | 1   |
|                | 10.1.0.4  | gold  | ipsec | 10.1.0.4  | C,I,R  | 1          | 203.0.113.226  | 1   |

If you do not see the TLOCs or you see any other codes here, check these:

<#root>

vSmart-vIPtela-MEX# show omp tlocs received

C -> chosen  
I -> installed  
  
Red -> redistributed  
Rej -> rejected  
L -> looped  
  
R -> resolved  
  
S -> stale  
Ext -> extranet  
Stg -> staged  
Inv -> invalid

| ADDRESS FAMILY | TLOC IP   | COLOR | ENCAP | FROM PEER | STATUS    | PSEUDO KEY | PUBLIC IP      | P  |
|----------------|-----------|-------|-------|-----------|-----------|------------|----------------|----|
| ipv4           | 10.1.0.5  | gold  | ipsec | 10.1.0.5  | C,I,R     | 1          | 203.0.113.225  | 4  |
|                | 10.1.0.2  | mpls  | ipsec | 10.1.0.2  | C,I,R     | 1          | 10.20.67.20    | 12 |
|                | 10.1.0.2  | blue  | ipsec | 10.1.0.2  | Rej,R,Inv | 1          | 198.51.100.187 | 12 |
|                | 10.1.0.30 | mpls  | ipsec | 10.1.0.30 | C,I,R     | 1          | 10.20.67.30    |    |
|                | 10.1.0.30 | gold  | ipsec | 10.1.0.30 | C,I,R     | 1          | 192.0.2.129    |    |
|                | 10.1.0.4  | mpls  | ipsec | 10.1.0.4  | C,I,R     | 1          | 10.20.67.40    | 1  |
|                | 10.1.0.4  | gold  | ipsec | 10.1.0.4  | C,I,R     | 1          | 203.0.113.226  | 1  |

Verify that there is no policy that blocks the TLOCs.

`show run policy control-policy` - look for any tloc-list that rejects your TLOCs as advertised or received in the vSmart.

<#root>


```
vSmart1(config-policy)# sh config
policy
lists
  tloc-list SITE20
    tloc 10.1.0.2 color blue encap ipsec
  !
!
control-policy SDWAN
sequence 10
  match tloc
    tloc-list SITE20
  !
  action reject ---->
```

here we are rejecting the TLOC 10.1.0.2,blue,ipsec

```
!
!
default-action accept
!
apply-policy
site-list SITE20
control-policy SDWAN in ---->
```

the policy is applied to control traffic coming IN the vSmart, it will filter the tlocs before adding i

---

 **Note:** If a TLOC is **Rejected** or **Invalid**, it is not advertised to the other vEdges.

---

Ensure that a policy does not filter the TLOC when it is advertised from the vSmart. You can see that the

TLOC is received on the vSmart, but you do not see it on the other vEdge.

Example 1: vSmart with TLOC in C,I,R.

<#root>

vSmart1# show omp tlocs

C -> chosen  
I -> installed  
  
Red -> redistributed  
Rej -> rejected  
L -> looped  
  
R -> resolved  
  
S -> stale  
Ext -> extranet  
Stg -> staged  
Inv -> invalid

| ADDRESS FAMILY | TLOC IP  | COLOR | ENCAP     | FROM PEER | STATUS | PSEUDO KEY    | PUBLIC IP      | P  |
|----------------|----------|-------|-----------|-----------|--------|---------------|----------------|----|
| ipv4           | 10.1.0.5 | mpls  | ipsec     | 10.1.0.5  | C,I,R  | 1             | 10.20.67.10    | 1  |
|                | 10.1.0.5 | gold  | ipsec     | 10.1.0.5  | C,I,R  | 1             | 203.0.113.225  | 4  |
| 10.1.0.2       | mpls     | ipsec | 10.1.0.2  | C,I,R     | 1      | 10.20.67.20   | 12386          | 10 |
|                | 10.1.0.2 | blue  | ipsec     | 10.1.0.2  | C,I,R  | 1             | 198.51.100.187 | 12 |
| 10.1.0.30      | mpls     | ipsec | 10.1.0.30 | C,I,R     | 1      | 10.20.67.30   |                |    |
|                | gold     | ipsec | 10.1.0.30 | C,I,R     | 1      | 192.0.2.129   |                |    |
| 10.1.0.4       | mpls     | ipsec | 10.1.0.4  | C,I,R     | 1      | 10.20.67.40   |                | 1  |
|                | gold     | ipsec | 10.1.0.4  | C,I,R     | 1      | 203.0.113.226 |                | 1  |

Example 2: vEdge1 does not see the TLOC from color blue that comes of vEdge2. It only sees MPLS TLOC.

<#root>

vEdge1# show omp tlocs

C -> chosen  
I -> installed  
Red -> redistributed  
Rej -> rejected  
L -> looped  
R -> resolved  
S -> stale  
Ext -> extranet  
Stg -> staged  
Inv -> invalid

| ADDRESS FAMILY | TLOC IP | COLOR | ENCAP | FROM PEER | STATUS | PSEUDO KEY | PUBLIC IP | P |
|----------------|---------|-------|-------|-----------|--------|------------|-----------|---|
|----------------|---------|-------|-------|-----------|--------|------------|-----------|---|

|      |           |      |       |          |         |   |               |   |
|------|-----------|------|-------|----------|---------|---|---------------|---|
| ipv4 | 10.1.0.5  | mpls | ipsec | 0.0.0.0  | C,Red,R | 1 | 10.20.67.10   | 1 |
|      | 10.1.0.5  | gold | ipsec | 0.0.0.0  | C,Red,R | 1 | 203.0.113.225 | 4 |
|      | 10.1.0.2  | mpls | ipsec | 10.1.0.3 | C,I,R   | 1 | 10.20.67.20   | 1 |
|      | 10.1.0.30 | mpls | ipsec | 10.1.0.3 | C,I,R   | 1 | 10.20.67.30   |   |
|      | 10.1.0.30 | gold | ipsec | 10.1.0.3 | C,I,R   | 1 | 192.0.2.129   |   |
|      | 10.1.0.4  | mpls | ipsec | 10.1.0.3 | C,I,R   | 1 | 10.20.67.40   |   |
|      | 10.1.0.4  | gold | ipsec | 10.1.0.3 | C,I,R   | 1 | 203.0.113.226 |   |
|      |           |      |       |          |         |   |               |   |
|      |           |      |       |          |         |   |               |   |
|      |           |      |       |          |         |   |               |   |

When you check the policy, you can see why the TLOC does not appear on the vEdge1.

```
<#root>
```

```
vSmart1# show running-config policy
policy
lists
  tloc-list SITE20
    tloc 10.1.0.2 color blue encaps ipsec
  !
  site-list SITE10
  site-id 10
  !
  !
  control-policy SDWAN
  sequence 10
  match tloc
    tloc-list SITE20
  !
  action reject
  !
  !
  default-action accept
  !
  apply-policy
  site-list SITE10
    control-policy SDWAN out
  !
  !
```

## Bidirectional Forwarding Detection

### Understand the show bfd sessions command

These are the key things to look for in the output:

```
<#root>
```



```
vEdge-2# show bfd sessions
```

| SYSTEM IP | SITE ID | STATE | SOURCE TLOC COLOR | REMOTE TLOC COLOR | SOURCE IP   |
|-----------|---------|-------|-------------------|-------------------|-------------|
| 10.1.0.5  | 10      | down  | blue              | gold              | 10.19.146.2 |
| 10.1.0.30 | 30      | up    | blue              | gold              | 10.19.146.2 |
| 10.1.0.4  | 40      | up    | blue              | gold              | 10.19.146.2 |
| 10.1.0.4  | 40      | up    | mpls              | mpls              | 10.20.67.10 |

- **SYSTEM IP:** Peers system-ip
- **SOURCE and REMOTE TLOC COLOR:** This is useful to know what TLOC is expected to receive and send.
- **SOURCE IP:** It is the **private** source IP. If you are behind a NAT, this information is displayed here (it can be seen with the use of `show control local-properties <wan-interface-list>`).
- **DST PUBLIC IP:** It is the destination that the vEdge uses to form the **Data Plane** tunnel, whether or not it is behind NAT. (Example: vEdges directly attached to the Internet, or **Multi-Protocol Label Switching (MPLS)** links)
- **DST PUBLIC PORT** Public NAT-ed port that the vEdge uses in order to form the **Data Plane** tunnel to the remote vEdge.
- **TRANSITIONS:** Number of times the BFD session has changed its status, from NA to UP and vice versa.


## Command show tunnel statistics

The `show tunnel statistics` can display information about the data plane tunnels. You can determine if you send or receive packets for a particular IPSEC tunnel between the vEdges.

This can help you understand if packets arrive on each end, and isolate connectivity issues between the nodes.

In the example, when you run the command multiple times, you can notice an increment or no increment in the `tx-pkts` or `rx-pkts`.

---

 **Tip:** If your counter for tx-pkts increment, you transmit data out to the peer. If your rx-pkts does not increment, it means that data is not received from your peer. In this event, check the other end and confirm if the tx-pkts increments.

---

```
<#root>
```

```
vEdge2# show tunnel statistics
```

| TUNNEL PROTOCOL | SOURCE IP     | DEST IP        | SOURCE PORT | DEST PORT | SYSTEM IP | LOCAL COLOR     | REMOTE COLOR    |     |
|-----------------|---------------|----------------|-------------|-----------|-----------|-----------------|-----------------|-----|
| ipsec           | 172.16.16.147 | 10.88.244.181  | 12386       | 12406     | 10.1.0.5  | public-internet | default         |     |
| ipsec           | 172.16.16.147 | 10.152.201.104 | 12386       | 63364     | 10.1.0.0  | public-internet | default         | 144 |
| ipsec           | 172.16.16.147 | 10.152.204.31  | 12386       | 58851     | 10.1.0.7  | public-internet | public-internet |     |

|          |               |                |             |           |           |                 |                 |     |
|----------|---------------|----------------|-------------|-----------|-----------|-----------------|-----------------|-----|
| ipsec    | 172.24.90.129 | 10.88.244.181  | 12426       | 12406     | 10.1.0.5  | biz-internet    | default         |     |
| ipsec    | 172.24.90.129 | 10.152.201.104 | 12426       | 63364     | 10.1.0.0  | biz-internet    | default         | 144 |
| ipsec    | 172.24.90.129 | 10.152.204.31  | 12426       | 58851     | 10.1.0.7  | biz-internet    | public-internet |     |
| TUNNEL   |               |                |             |           |           |                 |                 |     |
| PROTOCOL | SOURCE IP     | DEST IP        | SOURCE PORT | DEST PORT | SYSTEM IP | LOCAL COLOR     | REMOTE COLOR    |     |
| -----    |               |                |             |           |           |                 |                 |     |
| ipsec    | 172.16.16.147 | 10.88.244.181  | 12386       | 12406     | 10.1.0.5  | public-internet | default         |     |
| ipsec    | 172.16.16.147 | 10.152.201.104 | 12386       | 63364     | 10.1.0.0  | public-internet | default         | 144 |
| ipsec    | 172.16.16.147 | 10.152.204.31  | 12386       | 58851     | 10.1.0.7  | public-internet | public-internet |     |
| ipsec    | 172.24.90.129 | 10.88.244.181  | 12426       | 12406     | 10.1.0.5  | biz-internet    | default         |     |
| ipsec    | 172.24.90.129 | 10.152.201.104 | 12426       | 63364     | 10.1.0.0  | biz-internet    | default         | 144 |
| ipsec    | 172.24.90.129 | 10.152.204.31  | 12426       | 58851     | 10.1.0.7  | biz-internet    | public-internet |     |

Another useful command is `show tunnel statistics bfd` that can be used to check the number of BFD packets sent and received within particular data plane tunnel:

```
vEdge1# show tunnel statistics bfd
```

| TUNNEL | SOURCE IP     | DEST IP       | SOURCE PORT | DEST PORT | BFD ECHO TX PKTS | BFD ECHO RX PKTS | BFD ECHO TX OCTETS | BFD ECHO RX OCTETS | BFD PMTU TX PKTS | BFD PMTU RX PKTS |
|--------|---------------|---------------|-------------|-----------|------------------|------------------|--------------------|--------------------|------------------|------------------|
| ipsec  | 192.168.109.4 | 192.168.109.5 | 4500        | 4500      | 0                | 0                | 0                  | 0                  | 0                | 0                |
| ipsec  | 192.168.109.4 | 192.168.109.5 | 12346       | 12366     | 1112255          | 1112253          | 186302716          | 186302381          | 487              | 487              |
| ipsec  | 192.168.109.4 | 192.168.109.7 | 12346       | 12346     | 1112254          | 1112252          | 186302552          | 186302210          | 487              | 487              |
| ipsec  | 192.168.109.4 | 192.168.110.5 | 12346       | 12366     | 1112255          | 1112253          | 186302716          | 186302381          | 487              | 487              |

## Access List

An access list is a useful and a necessary step after you look at the `show bfd sessions` output.

Now that the private, and public IPs and Ports are known, you can create an **Access Control List (ACL)** to match against the `SRC_PORT`, `DST_PORT`, `SRC_IP`, `DST_IP`.

This can help to verify sent and received BFD messages.

Here, you can find an example of an ACL configuration:

```
policy
access-list checkbfd-out
sequence 10
match
source-ip 192.168.0.92/32
destination-ip 198.51.100.187/32
source-port 12426
destination-port 12426
!
```

```

    action accept
      count bfd-out-to-dc1-from-br1
    !
  !
  default-action accept
!
access-list checkbfd-in
sequence 20
  match
    source-ip      198.51.100.187/32
    destination-ip 192.168.0.92/32
    source-port    12426
    destination-port 12426
  !
  action accept
    count bfd-in-from-dc1-to-br1
  !
!
  default-action accept
!
vpn 0
interface ge0/0
  access-list checkbfd-in in
  access-list checkbfd-out out
!
!
!
```

In the example, this ACL uses two sequences. The sequence 10 matches the BFD messages that are sent from this vEdge to the peer. Sequence 20 does the opposite.

It matches against the source (**Private**) port and destination (**Public**) ports. If the vEdge uses NAT, ensure to check the right source and destination ports.

To check the hits on each sequence counter issue the `show policy access-list counters <access-list name>`

```
vEdge1# show policy access-list-counters
```

| NAME     | COUNTER NAME            | PACKETS | BYTES |
|----------|-------------------------|---------|-------|
| checkbfd | bfd-out-to-dc1-from-br1 | 10      | 2048  |
|          | bfd-in-from-dc1-to-br1  | 0       | 0     |

## Network Address Translation

### How to Use tools stun-client to Detect NAT Maps and Filters.

If you have done all the steps and you are behind NAT, the next step is to identify the **UDP NAT Traversal (RFC 4787) Map and Filter** behavior.

This tool is used to discover the local vEdge external IP address when that vEdge is located behind a NAT device.

This command obtains a port mapping for the device and optionally discovers properties about the NAT between the local device and a server (public server: example google stun server).

---

 **Note:** For more detailed information visit: [Docs Viptela - STUN Client](#)

---

<#root>

```
vEdge1# tools stun-client vpn 0 options "--mode full --localaddr 192.168.12.100 12386 --verbosity 2 stunclient --mode full --localaddr 192.168.12.100 stun.l.google.com in VPN 0
Binding test: success
Local address: 192.168.12.100:12386
Mapped address: 203.0.113.225:4501
Behavior test: success

Nat behavior: Address Dependent Mapping

Filtering test: success

Nat filtering: Address and Port Dependent Filtering
```

On newer versions of software, syntax can be bit different:


<#root>

```
vEdge1# tools stun-client vpn 0 options "--mode full --localaddr 192.168.12.100 --localport 12386 --verb
```

In this example, you perform a full NAT detection test with the use of UDP source port 12386 to the Google STUN server.

The output of this command gives you NAT behavior and the NAT filter type based on RFC 4787.

---

 **Note:** When you use `tools stun`, remember to allow the STUN service in the tunnel interface, otherwise it does not work. Use `allow-service stun` in order to let the stun data pass.

---

<#root>

```
vEdge1# show running-config vpn 0 interface ge0/0
vpn 0
interface ge0/0
 ip address 10.19.145.2/30
 !
 tunnel-interface
 encapsulation ipsec
 color gold
 max-control-connections 1
 no allow-service bgp
 allow-service dhcp
 allow-service dns
 no allow-service icmp
 no allow-service sshd
```

```

no allow-service netconf
no allow-service ntp
no allow-service ospf

allow-service stun

!
no shutdown
!
!
```

This shows the mapping between STUN terminology (Full-Cone NAT) and RFC 4787 (NAT Behavioral for UDP).

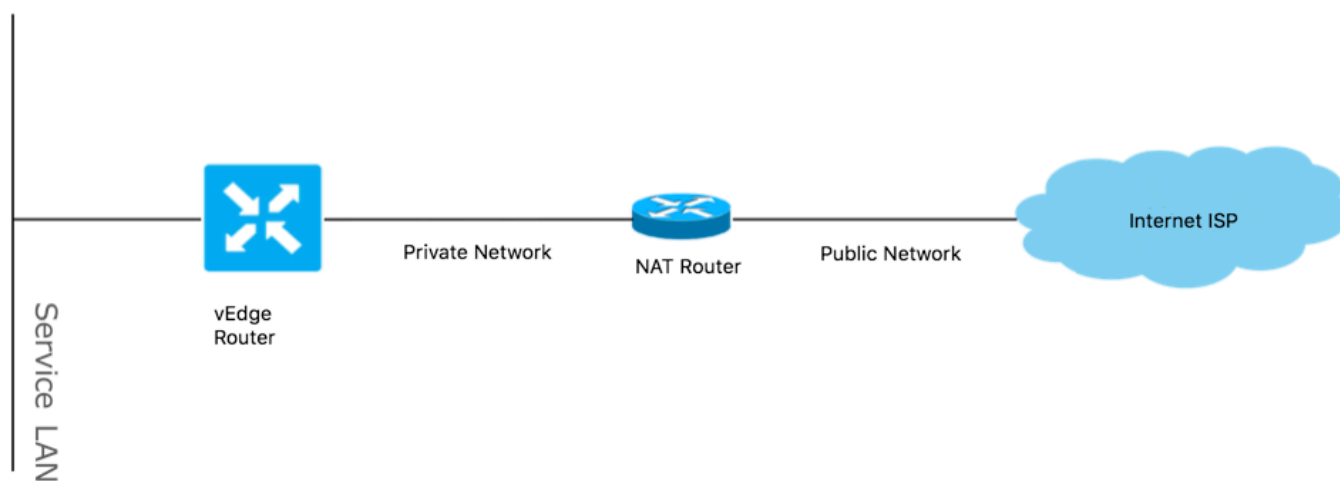
| NAT Traversal Mapping Between used Viptela Terminologies |  |                                      |
|--|--|--------------------------------------|
| STUN RFC 3489 Terminology                                | RFC 4787 Terminology                   |                                      |
|  | Mapping Behavior                       | Filtering Behavior                   |
| Full-cone NAT  | Endpoint-Independent Mapping           | Endpoint-Independent Filtering       |
| Restricted Cone NAT                                      | Endpoint-Independent Mapping           | Address-Dependent Filtering          |
| Port-Restricted Cone NAT                                 | Endpoint-Independent Mapping           | Address and Port-Dependent Filtering |
| Symmetric NAT  | Address-and(or) Port-Dependent Mapping | Address-Dependent Filtering          |
|  |  | Address and Port-Dependent Filtering |

## Supported NAT Types for Data Plane Tunnels“Sending” used in CLI

In most of the cases, your public colors like biz-internet or public-internet can be directly attached to the internet.

In other cases, there is a NAT device behind the vEdge WAN interface and the actual Internet Service Provider.

In this manner, the vEdge can have a private IP and the other device (Router, Firewall, etc) can be the device with the public facing IP addresses.



If you have an incorrect NAT type, then it could potentially be one of the most common reasons that do not allow the formation of Data Plane tunnels. These are the supported NAT types.

| NAT Traversal Support    |                          |                    |
|--------------------------|--------------------------|--------------------|
| Source                   | Destination              | Supported (YES/NO) |
| Full-Cone NAT            | Full-cone NAT            | Yes                |
| Full-Cone NAT            | Restricted Cone NAT      | Yes                |
| Full-Cone NAT            | Port-Restricted Cone NAT | Yes                |
| Full-Cone NAT            | Symmetric NAT            | Yes                |
| Restricted Cone NAT      | Full-cone NAT            | Yes                |
| Restricted Cone NAT      | Restricted Cone NAT      | Yes                |
| Restricted Cone NAT      | Port-Restricted Cone NAT | Yes                |
| Restricted Cone NAT      | Symmetric NAT            | Yes                |
| Port-Restricted Cone NAT | Full-cone NAT            | Yes                |
| Port-Restricted Cone NAT | Restricted Cone NAT      | Yes                |
| Port-Restricted Cone NAT | Port-Restricted Cone NAT | Yes                |
| Port-Restricted Cone NAT | Symmetric NAT            | No                 |
| Symmetric NAT            | Full-cone NAT            | Yes                |
| Symmetric NAT            | Restricted Cone NAT      | yes                |
| Symmetric NAT            | Port-Restricted Cone NAT | No                 |
| Symmetric NAT            | Symmetric NAT            | No                 |

## Firewalls

If you already checked the NAT and its not in the unsupported **Source** and **Destination** types, it is possible that a Firewall blocks the ports used to form the **Data Plane** tunnels.

Ensure that these ports are open in the Firewall for Data Plane connections: **vEdge to vEdge Data Plane**:

UDP 12346 to 13156

For control connections from vEdge to controllers:

UDP 12346 to 13156

TCP 23456 to 24156

Ensure that you open these ports in order to achieve successful connection of the Data Plane tunnels.

When you check the source and destination ports used for Data Plane tunnels, you can use `show tunnel statistics` or `show bfd sessions | tab` but NOT `show bfd sessions`.

It does not show any source ports, only destination ports as you can see:

```
vEdge1# show bfd sessions
```

| SYSTEM IP      | SITE ID | STATE | SOURCE TLOC COLOR | REMOTE TLOC COLOR | SOURCE IP       |
|----------------|---------|-------|-------------------|-------------------|-----------------|
| 192.168.30.105 | 50      | up    | biz-internet      | biz-internet      | 192.168.109.181 |
| 192.168.30.105 | 50      | up    | private1          | private1          | 192.168.110.181 |

```
vEdge1# show bfd sessions | tab
```

| SRC IP | DST IP | PROTO | SRC PORT | DST PORT | SYSTEM IP | SITE ID | LOCAL COLOR | COLOR |
|--------|--------|-------|----------|----------|-----------|---------|-------------|-------|
|--------|--------|-------|----------|----------|-----------|---------|-------------|-------|

```
192.168.109.181 192.168.109.182 ipsec 12346 12346 192.168.30.105 50 biz-internet biz-internet
192.168.110.181 192.168.110.182 ipsec 12346 12346 192.168.30.105 50 private1 private1
```

---

 **Note:** More information about SD-WAN firewall ports used can be found [here](#).

---

## Security

If you observe that the ACL counter increases inbound and outbound, check several iterations `show system statistics diff` and ensure there are no drops.

```
<#root>
```

```
vEdge1# show policy access-list-counters
```

| NAME     | COUNTER NAME            | PACKETS | BYTES |
|----------|-------------------------|---------|-------|
| checkbfd | bfd-out-to-dc1-from-br1 | 55      | 9405  |
|          | bfd-in-from-dc1-to-br1  | 54      | 8478  |

In this output, `rx_replay_integrity_drops` increase with every iteration of the `show system statistics diff` command.

```
<#root>
```

```
vEdge1#show system statistics diff
```

```
rx_pkts : 5741427
ip_fwd : 5952166
ip_fwd_arp : 3
ip_fwd_to_egress : 2965437
ip_fwd_null_mcast_group : 26
ip_fwd_null_nhop : 86846
ip_fwd_to_cpu : 1413393
ip_fwd_from_cpu_non_local : 15
ip_fwd_rx_ipsec : 1586149
ip_fwd_mcast_pkts : 26
rx_bcast : 23957
rx_mcast : 304
rx_mcast_link_local : 240
rx_implicit_acl_drops : 12832
rx_ipsec_decap : 21
rx_spi_ipsec_drops : 16
```

```
rx_replay_integrity_drops : 1586035
```

```
port_disabled_rx : 2
rx_invalid_qtags : 212700
rx_non_ip_drops : 1038073
pko_wred_drops : 3
```

```
bfd_tx_record_changed : 23
rx_arp_non_local_drops : 19893
rx_arp_reqs : 294
rx_arp_replies : 34330
arp_add_fail : 263
tx_pkts : 4565384
tx_mcast : 34406
port_disabled_tx : 3
tx_ipsec_pkts : 1553753
tx_ipsec_encap : 1553753
tx_pre_ipsec_pkts : 1553753
tx_pre_ipsec_encap : 1553753
tx_arp_replies : 377
tx_arp_reqs : 34337
tx_arp_req_fail : 2
bfd_tx_pkts : 1553675
bfd_rx_pkts : 21
bfd_tx_octets : 264373160
bfd_rx_octets : 3600
bfd_pmtu_tx_pkts : 78
bfd_pmtu_tx_octets : 53052
rx_icmp_echo_requests : 48
rx_icmp_network_unreach : 75465
rx_icmp_other_types : 47
tx_icmp_echo_requests : 49655
tx_icmp_echo_replies : 48
tx_icmp_network_unreach : 86849
tx_icmp_other_types : 7
vEdge1# show system statistics diff
```

```
rx_pkts : 151
ip_fwd : 157
ip_fwd_to_egress : 75
ip_fwd_null_nhop : 3
ip_fwd_to_cpu : 43
ip_fwd_rx_ipsec : 41
rx_bcast : 1
```

```
rx_replay_integrity_drops : 41
```

```
rx_invalid_qtags : 7
rx_non_ip_drops : 21
rx_arp_non_local_drops : 2
tx_pkts : 114
tx_ipsec_pkts : 40
tx_ipsec_encap : 40
tx_pre_ipsec_pkts : 40
tx_pre_ipsec_encap : 40
tx_arp_reqs : 1
bfd_tx_pkts : 40
bfd_tx_octets : 6800
tx_icmp_echo_requests : 1
```

```
vEdge1# show system statistics diff
```

```
rx_pkts : 126
ip_fwd : 125
ip_fwd_to_egress : 58
ip_fwd_null_nhop : 3
ip_fwd_to_cpu : 33
ip_fwd_rx_ipsec : 36
rx_bcast : 1
```



rx\_implicit\_acl\_drops : 1

rx\_replay\_integrity\_drops : 35

rx\_invalid\_qtags : 6  
rx\_non\_ip\_drops : 22  
rx\_arp\_replies : 1  
tx\_pkts : 97  
tx\_mcast : 1  
tx\_ipsec\_pkts : 31  
tx\_ipsec\_encap : 31  
tx\_pre\_ipsec\_pkts : 31  
tx\_pre\_ipsec\_encap : 31  
bfd\_tx\_pkts : 32  
bfd\_tx\_octets : 5442  
rx\_icmp\_network\_unreach : 3  
tx\_icmp\_echo\_requests : 1  
tx\_icmp\_network\_unreach : 3

vEdge1# show system statistics diff

rx\_pkts : 82  
ip\_fwd : 89  
ip\_fwd\_to\_egress : 45  
ip\_fwd\_null\_nhop : 3  
ip\_fwd\_to\_cpu : 24  
ip\_fwd\_rx\_ipsec : 22  
rx\_bcast : 1  
rx\_implicit\_acl\_drops : 1

rx\_replay\_integrity\_drops : 24

rx\_invalid\_qtags : 2  
rx\_non\_ip\_drops : 14  
rx\_arp\_replies : 1  
tx\_pkts : 62  
tx\_mcast : 1  
tx\_ipsec\_pkts : 24  
tx\_ipsec\_encap : 24  
tx\_pre\_ipsec\_pkts : 24  
tx\_pre\_ipsec\_encap : 24  
tx\_arp\_reqs : 1  
bfd\_tx\_pkts : 23  
bfd\_tx\_octets : 3908  
rx\_icmp\_network\_unreach : 3  
tx\_icmp\_echo\_requests : 1  
tx\_icmp\_network\_unreach : 3

vEdge1# show system statistics diff

rx\_pkts : 80  
ip\_fwd : 84  
ip\_fwd\_to\_egress : 39  
ip\_fwd\_to\_cpu : 20  
ip\_fwd\_rx\_ipsec : 24

rx\_replay\_integrity\_drops : 22

rx\_invalid\_qtags : 3  
rx\_non\_ip\_drops : 12

```
tx_pkts : 66
tx_ipsec_pkts : 21
tx_ipsec_encap : 21
tx_pre_ipsec_pkts : 21
tx_pre_ipsec_encap : 21
bfd_tx_pkts : 21
bfd_tx_octets : 3571
```

First, perform a `request security ipsec-rekey` on the vEdge. Then, go through several iterations of `show system statistics diff` and see if you still see `rx_replay_integrity_drops`.

If you do, check your security configuration.

```
vEdge1# show running-config security
security
 ipsec
 authentication-type sha1-hmac ah-sha1-hmac
!
```

## ISP Problems with DSCP Marked Traffic

By default, all control and management traffic from the vEdge router to the controllers travels over DTLS or TLS connections and marked with a DSCP value of CS6 (48 decimal).

For data plane tunnels traffic, vEdge routers use IPsec or GRE encapsulation to send data traffic to each other.

For data plane failure detection and performance measurement, routers periodically send each other BFD packets.

These BFD packets are also marked with a DSCP value of CS6 (48 decimal).

From the perspective of ISP, this type of traffic is seen as UDP traffic with DSCP value CS6 as well because vEdge routers and SD-WAN controllers copy DSCP that marks to the outer IP header by default.

Here is how it can look like if tcpdump runs on transit ISP router:

```
14:27:15.993766 IP (tos 0xc0, ttl 64, id 44063, offset 0, flags [DF], proto UDP (17), length 168)
 192.168.109.5.12366 > 192.168.20.2.12346: [udp sum ok] UDP, length 140
14:27:16.014900 IP (tos 0xc0, ttl 63, id 587, offset 0, flags [DF], proto UDP (17), length 139)
 192.168.20.2.12346 > 192.168.109.5.12366: [udp sum ok] UDP, length 111
14:27:16.534117 IP (tos 0xc0, ttl 63, id 0, offset 0, flags [DF], proto UDP (17), length 157)
 192.168.109.5.12366 > 192.168.110.6.12346: [no cksum] UDP, length 129
14:27:16.534289 IP (tos 0xc0, ttl 62, id 0, offset 0, flags [DF], proto UDP (17), length 150)
 192.168.110.6.12346 > 192.168.109.5.12366: [no cksum] UDP, length 122
```

As can be seen here, all packets are marked with TOS byte 0xc0 also known as DS field (that is equal to decimal 192, or 110 000 00 in binary).

First 6 high order bits correspond to DSCP bits value 48 in decimal or CS6).

First 2 packets in the output correspond to a control plane tunnel and the 2 that remain, to a data plane tunnel traffic.

Based on the packet length and the TOS mark, it can conclude with high confidence that it was BFD packets (RX and TX directions). These packets are marked with CS6 as well.

Sometimes some service providers (especially MPLS L3 VPN/MPLS L2 VPN service providers) maintain different SLA and can handle a different class of traffic based on DSCP marks differently.

For example, if you have premium service to prioritize DSCP EF and CS6 voice and signaling traffic.

Since priority traffic is almost always policed, even if the total bandwidth of an uplink is not exceeded, for this type of traffic packet loss can be seen and hence BFD sessions can be flapping as well.

It was seen in some cases that if dedicated priority queue on service provider router is starved, you do not see any drops for normal traffic (for example, when you run simple **ping** from vEdge router).

This is because such traffic is marked with default DSCP value 0 as can be seen here (TOS byte):

```
15:49:22.268044 IP (tos 0x0, ttl 62, id 0, offset 0, flags [DF], proto UDP (17), length 142)
  192.168.110.5.12366 > 192.168.109.7.12346: [no cksum] UDP, length 114
15:49:22.272919 IP (tos 0x0, ttl 62, id 0, offset 0, flags [DF], proto UDP (17), length 142)
  192.168.110.5.12366 > 192.168.109.7.12346: [no cksum] UDP, length 114
15:49:22.277660 IP (tos 0x0, ttl 62, id 0, offset 0, flags [DF], proto UDP (17), length 142)
  192.168.110.5.12366 > 192.168.109.7.12346: [no cksum] UDP, length 114
15:49:22.314821 IP (tos 0x0, ttl 62, id 0, offset 0, flags [DF], proto UDP (17), length 142)
  192.168.110.5.12366 > 192.168.109.7.12346: [no cksum] UDP, length 114
```

But at the same time, your BFD sessions flap:

```
show bfd history
```

| SYSTEM IP    | SITE ID | COLOR           | STATE | DST PUBLIC IP | DST PUBLIC PORT | ENCAP | TIME         |
|--------------|---------|-----------------|-------|---------------|-----------------|-------|--------------|
| 192.168.30.4 | 13      | public-internet | up    | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | up    | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | down  | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | down  | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | up    | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | up    | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | down  | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | down  | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.4 | 13      | public-internet | up    | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.6 | 13      | public-internet | up    | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |
| 192.168.30.6 | 13      | public-internet | down  | 192.168.109.4 | 12346           | ipsec | 2019-05-01T0 |

And here **nping** comes handy in order to troubleshoot:

```
vedge2# tools nping vpn 0 options "--tos 0x0c --icmp --icmp-type echo --delay 200ms -c 100 -q" 192.168.1.10
Nping in VPN 0
```

```
Starting Nping 0.6.47 ( http://nmap.org/nping ) at 2019-05-07 15:58 CEST
Max rtt: 200.305ms | Min rtt: 0.024ms | Avg rtt: 151.524ms
Raw packets sent: 100 (2.800KB) | Rcvd: 99 (4.554KB) | Lost: 1 (1.00%)
Nping done: 1 IP address pinged in 19.83 seconds
```

## Debug BFD

If deeper investigation is required, run debugging of BFD on the vEdge router.

Forwarding Traffic Manager (FTM) is responsible for BFD operations on vEdge routers and hence you need **debug ftm bfd**.

All debugging output is stored in `/var/log/tmplog/vdebug` file and if you want to have those messages on the console (similar to Cisco IOS **terminal monitor** behavior), you can use **monitor start /var/log/tmplog/vdebug**.

In order to stop logging, you can use **monitor stop /var/log/tmplog/vdebug**

Here is how the output looks for BFD session that goes down because of the timeout (remote TLOC with IP address 192.168.110.6 is not reachable anymore):

```
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_state[1008]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_proc_tunnel_public_tloc_msg[252]: tun_rec_index 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_increment_wanif_bfd_flap[2427]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_state[1119]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1140]: Attempting to add TLOC : from_tloc 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_set_del_marker_internal[852]: (32771:32772) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_set_del_marker_internal[852]: (32770:32771) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_create[238]: Attempting BFD session 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_clear_delete_marker[828]: (32771:32772) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_create[238]: Attempting BFD session 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_clear_delete_marker[828]: (32770:32771) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_sa[1207]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1653]: BFD (32771:32772) src 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_sa[1207]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1653]: BFD (32770:32772) src 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_state[1008]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_proc_tunnel_public_tloc_msg[252]: tun_rec_index 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_increment_wanif_bfd_flap[2427]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_state[1119]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1140]: Attempting to add TLOC : from_tloc 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_set_del_marker_internal[852]: (32771:32772) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_set_del_marker_internal[852]: (32770:32771) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_create[238]: Attempting BFD session 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_clear_delete_marker[828]: (32771:32772) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_create[238]: Attempting BFD session 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_clear_delete_marker[828]: (32770:32771) 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_sa[1207]: BFD-session TNL 192.168.110.6
log:local17.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1653]: BFD (32771:32772) src 192.168.110.6
```

```

log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_sa[1207]: BFD-session TNL 192
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1653]: BFD (32770:32772) src 192.168.1
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_send_bfd_msg[499]: Sending BFD notification Dow
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1140]: Attempting to add TLOC : from_t
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_set_del_marker_internal[852]: (32771
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_set_del_marker_internal[852]: (32770
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1285]: UPDATE local tloc
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_create[238]: Attempting BFD session
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_clear_delete_marker[828]: (32771:327
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_create[238]: Attempting BFD session
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_clear_delete_marker[828]: (32770:327
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_sa[1207]: BFD-session TNL 192
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1653]: BFD (32771:32772) src 192.168.1
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: bfdmgr_session_update_sa[1207]: BFD-session TNL 192
log:local7.debug: May 7 16:23:09 vedge2 FTMD[674]: ftm_tloc_add[1653]: BFD (32770:32772) src 192.168.1
log:local7.info: May 7 16:23:09 vedge2 FTMD[674]: %Viptela-vedge2-ftmd-6-INFO-1400002: Notification: 5
log:local7.info: May 7 16:23:09 vedge2 FTMD[674]: %Viptela-vedge2-ftmd-6-INFO-1400002: Notification: 5

```

Another valuable debug in order to enable is **Tunnel Traffic Manager (TTM)** events debug is **debug ttm events**.

Here is how **BFD DOWN** event looks like from the perspective of TTM:

```

log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[194]: Received TTM Msg LINK_
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[413]: Remote-TLOC: 192.1
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[194]: Received TTM Msg LINK_
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[413]: Remote-TLOC: 192.1
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[194]: Received TTM Msg BFD,
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[402]: TLOC: 192.168.30.6
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_af_tloc_db_bfd_status[234]: BFD message: I SA
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[194]: Sent TTM Msg TLOC_ADD,
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[213]: TLOC: 192.168.30.6
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[217]: Attributes: GROU
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[220]: Preference: 0
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[223]: Weight: 1
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[226]: Gen-ID: 214748
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[229]: Version: 2
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[232]: Site-ID: 13
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[235]: Carrier: 4
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[241]: Restrict: 0
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[249]: Group: Count:
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[262]: Groups: 0
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[269]: TLOCv4-Public:
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[273]: TLOCv4-Private
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[277]: TLOCv6-Public:
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[281]: TLOCv6-Private
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[285]: TLOC-Encap: ip
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[295]: Authenticati
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[312]: SPI 334, Fla
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[317]: Number of pr
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[328]: Number of en
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[333]: Encrypt type
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[333]: Encrypt type
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[339]: Number of in
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[344]: integrity ty
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[349]: #Paths: 0
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[194]: Sent TTM Msg TLOC_ADD,
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[213]: TLOC: 192.168.30.6
log:local7.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[217]: Attributes: GROU

```

```

log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[220]: Preference: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[223]: Weight: 1
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[226]: Gen-ID: 214748
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[229]: Version: 2
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[232]: Site-ID: 13
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[235]: Carrier: 4
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[241]: Restrict: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[249]: Group: Count:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[262]: Groups: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[269]: TLOCv4-Public:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[273]: TLOCv4-Private:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[277]: TLOCv6-Public:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[281]: TLOCv6-Private:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[285]: TLOC-Encap: ip
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[295]: Authentication:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[312]: SPI 334, Fla
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[317]: Number of pr
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[328]: Number of en
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[333]: Encrypt type
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[333]: Encrypt type
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[339]: Number of in
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[344]: integrity ty
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[349]: #Paths: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[194]: Sent TTM Msg TLOC_ADD,
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[213]: TLOC: 192.168.30.6
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[217]: Attributes: GROU
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[220]: Preference: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[223]: Weight: 1
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[226]: Gen-ID: 214748
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[229]: Version: 2
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[232]: Site-ID: 13
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[235]: Carrier: 4
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[241]: Restrict: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[249]: Group: Count:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[262]: Groups: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[269]: TLOCv4-Public:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[273]: TLOCv4-Private:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[277]: TLOCv6-Public:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[281]: TLOCv6-Private:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[285]: TLOC-Encap: ip
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[295]: Authentication:
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[312]: SPI 334, Fla
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[317]: Number of pr
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[328]: Number of en
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[333]: Encrypt type
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[333]: Encrypt type
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[339]: Number of in
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[344]: integrity ty
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[349]: #Paths: 0
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[194]: Sent TTM Msg DATA_DEVI
log:local17.debug: May 7 16:58:19 vedge2 TTMD[683]: ttm_debug_announcement[431]: Device: 192.168.30
log:local17.info: May 7 16:58:19 vedge2 FTMD[674]: %Viptela-vedge2-ftmd-6-INFO-1400002: Notification: 5
log:local17.info: May 7 16:58:20 vedge2 FTMD[674]: %Viptela-vedge2-ftmd-6-INFO-1400002: Notification: 5

```

## Related Information

- [Technical Support & Documentation - Cisco Systems](#)