Troubleshoot SD-WAN Hub Tunnels with Duplicate NAT Endpoints

Contents

Introduction

Prerequisites

- Requirements
- Components Used

Background Information

- Control-Plane NAT Learning
- Data-Plane NAT Learning and VCONTROL
- Port Preservation, Port Overlap, and Port Randomization

Sample Topology

Why the Collision Happens

- Different vBonds Can Observe the Same Public Endpoint
- The Same Advertisements Reach One Hub
- Private-as-Public Collisions with Repeated Branch Addresses
- NAT Endpoint Reassignment

Symptoms and Impact

Diagnose the Problem

- Step 1. Record the Software and Spoke Transport State
- Step 2. Find Duplicate Endpoints in OMP
- Step 3. Compare Logical TLOCs with Hub Session Ownership
- Step 4. Verify What the NAT Gateway Actually Does
- Step 5. Inspect VCONTROL in Wireshark
- Step 6. Confirm the Affected Path and Preserve Evidence

Workarounds and Limitations

- Use Stable Endpoint-Independent Mappings with Unique Allocations
- Disable Port Overlap Across Different Spokes
- Use Port Offset for a Small, Controlled Group
- Disable Source-Port Preservation or Enable Random Allocation
- Separate Public Addresses or Reserve Port Blocks
- Retain an Alternate Transport and Coordinate Recovery

Related Software Defects

Appendix: Decode the IPv4 VCONTROL Trailer

Related Information

Introduction

This document describes how to identify and mitigate hub tunnel failures when spokes share one public Network Address Translation (NAT) endpoint.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Cisco Catalyst software-defined wide area network (SD-WAN) architecture and control connections
- Overlay Management Protocol (OMP), Transport Locators (TLOCs), Bidirectional Forwarding Detection (BFD), and IP Security (IPsec)
- Multiprotocol Label Switching (MPLS) transport networks
- NAT mapping and filter behavior
- Basic packet analysis with Wireshark

Components Used

The information in this document is based on these software and hardware versions:

- Cisco IOS® XE Catalyst SD-WAN routers
- Cisco Catalyst SD-WAN Controller (vSmart)
- Cisco Catalyst SD-WAN Validator (vBond)

The hub session-ownership failure was reproduced on Cisco IOS XE 17.18.2 with pairwise keys operational. Command availability and output fields vary by platform and release.

Examples use private addresses or documentation addresses reserved by RFC 5737, with generic device names. CLI excerpts are shortened and use normalized identifiers. Interface names and some output fields vary by platform and release.

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.

⚠ **Caution**: Configuration changes and connection resets can interrupt traffic. Use an approved maintenance plan. The initial diagnostic commands are read-only.

Background Information

Start with a scenario in which symmetric NAT is correctly identified. Spoke A registers with vBond 1, and

Spoke B registers with vBond 2. The NAT gateway can assign the same translated source IP address and port to these connections when their destinations differ, as RFC 7857 permits. Both spokes can consequently advertise the same public endpoint for their public-internet TLOCs, even when both report NAT type A.

The sample topology uses the public-internet color. The same scenario can involve other public colors, such as biz-internet. Apply the diagnostic filters to the actual affected transport color; the literal public-internet color is not a required condition.

Note: Cisco Catalyst SD-WAN does not support TLOCs from different sites advertising the same public IP address and UDP port toward a common receiving Edge. Receiver behavior under this collision condition is unpredictable. This can occur even when symmetric NAT is correctly identified. Correct NAT classification alone does not guarantee that advertised endpoints are unique.

Control-Plane NAT Learning

An Edge sends its private transport address and port during registration with vBond. vBond observes the source address and port on that connection and returns the observed endpoint to the Edge. The Edge can advertise this information through OMP as the public address and port attributes of a TLOC, including a TLOC with a public color such as public-internet or biz-internet.

The observation describes the connection to that vBond. With destination-dependent NAT, the mapping toward a hub can be different. The router compares NAT observations in order to classify the mapping behavior. In **show sdwan control local-properties**, A means address-and-port-dependent mapping, and E means endpoint-independent mapping. The displayed classification describes the decision of the router; a NAT translation table or packet capture verifies the actual behavior of the NAT gateway.

For example, run **show sdwan control local-properties** and locate the **NAT TYPE** column for the transport. This shortened output retains the NAT-type legend and chosen interface columns, with normalized addresses:

<#root>

```
Spoke-A# show sdwan control local-properties
```

```
...
 NAT TYPE: E -- indicates End-point independent mapping
           A -- indicates Address-port dependent mapping
           N -- indicates Not learned
           Note: Requires minimum two vbonds to learn the NAT type

                         PUBLIC        PUBLIC PRIVATE        PRIVATE                         NAT
INTERFACE                IPv4          PORT   IPv4           PORT    COLOR         STATE     TYPE
GigabitEthernet2         192.168.9.11   12346  192.168.9.11   12346   mpls           up        E
GigabitEthernet3         203.0.113.10   23012  192.168.11.11  12346   biz-internet   up        A
```

The biz-internet transport reports A and advertises **203.0.113.10:23012** as its public endpoint. The MPLS transport reports E and has equal private and public endpoints. Read the classification from **NAT TYPE**;

address or port equality alone does not identify the NAT type. N means the router has not learned a classification.

Data-Plane NAT Learning and VCONTROL

Before a hub can process an encrypted BFD packet, it must find the inbound IPsec security association (SA) for the corresponding peer and tunnel. The SA identifies the cryptographic key and other state required to authenticate and decrypt the packet. The receiver uses endpoint information together with the Encapsulating Security Payload (ESP) Security Parameters Index (SPI) in order to locate that state. The SPI alone is not a globally unique spoke identifier.

A spoke behind symmetric NAT normally initiates traffic toward a reachable hub. This creates a mapping for that particular destination. The source address or port seen by the hub can differ from the public endpoint previously learned through vBond and advertised in OMP. As a result, the received outer headers do not always match the endpoint in the installed tunnel state. The hub needs a way to associate this new on-wire endpoint with the known remote TLOC and its IPsec state before it can decrypt BFD.

For the IPv4 BFD-over-IPsec format discussed here, SD-WAN places endpoint information in a visible VCONTROL trailer at the end of the UDP payload, outside the encrypted BFD payload. The hub can read this information before decryption. VCONTROL carries endpoint information, not encryption keys, and does not establish peer identity by itself.

For example, during initial tunnel establishment, before data-plane learning completes:

```
Spoke B advertised through OMP:       203.0.113.10:12346
Spoke B observed at the hub:          203.0.113.10:45678
VCONTROL carried source endpoint:    203.0.113.10:12346
```

The receiver first attempts its normal endpoint/SPI lookup. If that lookup does not find an SA, the VCONTROL source endpoint provides a fallback. In this example, the hub can use the carried **203.0.113.10:12346**, together with its local destination context and the SPI, in order to locate the inbound SA associated with the known remote TLOC, even though the packet arrives from **203.0.113.10:45678**.

The chosen SA provides the receive-key context. The SPI chooses the applicable key generation within that context. The hub then performs IPsec authentication and decryption; only a packet that passes the required checks is available for BFD processing. A readable trailer or a successful lookup alone does not prove successful authentication, decryption, or BFD reception.

The difference between the observed and carried source ports can therefore be normal with symmetric NAT. For the peer relationship to work, the hub must associate the packet with the correct tunnel and key state and learn the actual return endpoint. In this example, successful learning lets the hub send return traffic to **203.0.113.10:45678**. This is separate from control-plane learning through vBond.

VCONTROL does not force the receiver to override an existing endpoint/SPI match. If duplicate endpoint state associates a packet with the wrong peer or key generation, the trailer does not guarantee recovery. It

supplies a lookup hint; it cannot distinguish two peers that carry the same endpoint without the correct surrounding tunnel and cryptographic state.

VCONTROL is also present on applicable BFD packets when NAT type E is reported. Its presence is not a symmetric-NAT flag, and it is not included in every ordinary IPsec data packet. For the IPv4 BFD format described here, the public-TLOC information is omitted between two private colors.

The peer-specific endpoint learned through the data plane does not mean the spoke must replace its OMP advertisement with a new endpoint for every destination. Different learned mappings are an expected part of symmetric-NAT traversal.

Once a tunnel is Up, a sender that has learned its own translated endpoint for that tunnel can carry that endpoint as its source information. This is the per-tunnel public endpoint of the sender, not the remote endpoint or a new global OMP advertisement. Always compare the packet, OMP advertisement, and session state from the same observation period.

Port Preservation, Port Overlap, and Port Randomization

These are different behaviors:

Behavior	Meaning
Source-port preservation	The NAT gateway tries to retain the original UDP source port of the spoke, for example, 12346, as the translated source port.
Port overlap across destinations	The NAT gateway reuses one translated IP: port for different internal endpoints that communicate with different external destinations.
Port randomization	The NAT gateway chooses translated ports using a randomized allocation method.
Endpoint-independent mapping	One internal endpoint retains the same translated endpoint across external destinations while that mapping exists.

RFC 7857, Section 3 permits overlapping external ports for different internal endpoints when the destinations differ. The NAT distinguishes the flows using their five-tuples. This does not permit two simultaneous indistinguishable UDP flows to the same destination. Random allocation and preservation are separate allocation choices; neither establishes endpoint independence.

Sample Topology

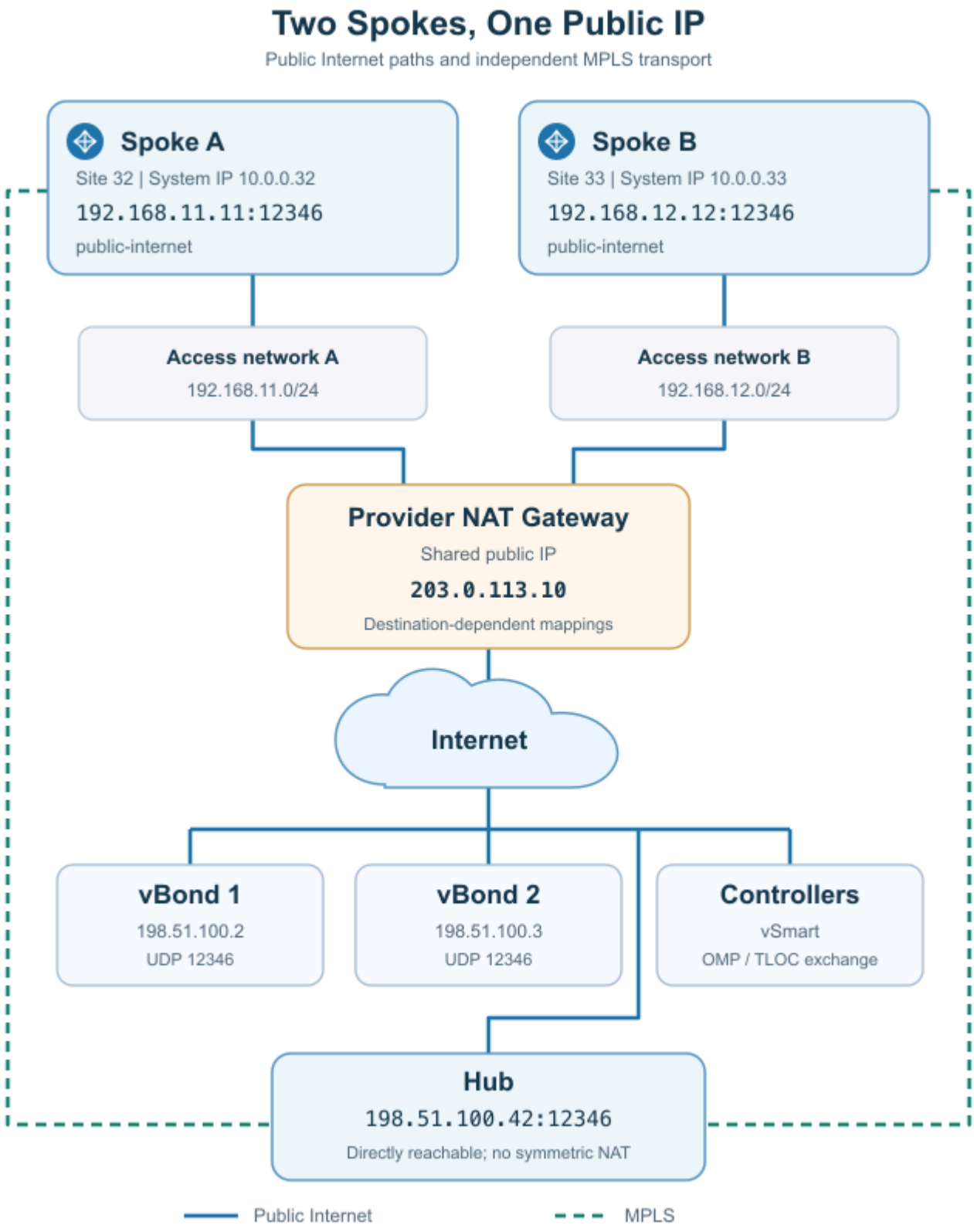


Figure 1. Shared NAT gateway, separate spoke access networks, and independent MPLS transport.

Spoke A uses **192.168.11.11:12346**, and Spoke B uses **192.168.12.12:12346**. Both share the public address **203.0.113.10** through the NAT gateway. The hub has a directly reachable address and is not behind symmetric NAT. The two spokes can advertise the same public IP:port even though their private endpoints differ.

Why the Collision Happens

Different vBonds Can Observe the Same Public Endpoint

Consider these simultaneous UDP mappings:

Peer	Private Source	Translated Source	Destination
A	192.168.11.11:12346	203.0.113.10:12346	vBond 1, 198.51.100.2:12346
B	192.168.12.12:12346	203.0.113.10:12346	vBond 2, 198.51.100.3:12346

Replies return correctly because the NAT also checks the remote endpoint. A reply from vBond 1 belongs to the mapping for A; a reply from vBond 2 belongs to the mapping for B.

Both routers can also contact the other vBond and correctly detect destination-dependent mapping. For example, keep the private source tuple for each spoke unchanged and consider this observation order:

Peer	First Accepted vBond Observation	Comparison from the Other vBond	Result
A	vBond 1 returns 203.0.113.10:12346	vBond 2 returns 203.0.113.10:40001	NAT A; retained public tuple 203.0.113.10:12346
B	vBond 2 returns 203.0.113.10:12346	vBond 1 returns 203.0.113.10:40002	NAT A; retained public tuple 203.0.113.10:12346

The vBond observations differ for each router, so detecting NAT A is correct. The retained first observations nevertheless match across the two routers. Classification A does not allocate a unique public port across all spokes, and visiting both vBonds does not guarantee that uniqueness.

Each spoke receives the public address and port observed by vBond in the registration reply and uses the retained observation to populate its local transport properties. The spoke includes that public endpoint, together with its private endpoint, in the TLOC attributes it advertises to the SD-WAN Controller through OMP. The Controller then distributes the TLOC to other Edges according to the applicable control policy; vBond itself does not originate the OMP advertisement.

If both spokes retain the same observed public endpoint, OMP can therefore contain:

```
TLOC system IP   Color            Public endpoint       Private endpoint
10.0.0.32        public-internet   203.0.113.10:12346    192.168.11.11:12346
10.0.0.33        public-internet   203.0.113.10:12346    192.168.12.12:12346
```

These remain different logical TLOCs because their system IPs differ.

The Same Advertisements Reach One Hub

The NAT gateway must distinguish both spokes when they contact the same hub. One possible allocation is:

Peer	Translated Source toward Hub	Destination
A	203.0.113.10:12346	198.51.100.42:12346
B	203.0.113.10:45678	198.51.100.42:12346

The hub has two logical peers advertising the same destination endpoint, while packets can arrive from different actual endpoints. Under normal symmetric-NAT operation, data-plane learning helps establish the return mappings.

The hub session-ownership limitation arises when different peers share session state chosen by this transport key:

```
Hub-local IP + hub-local port + remote IP + remote port + protocol
```

If both advertisements choose the same key, an update from a later peer can replace ownership information in shared state. A subsequent withdrawal, deletion, or recreation of the session for one peer can disturb the IPsec/BFD state for the other peer. Data-plane learning does not guarantee recovery when that state is already associated with the wrong peer or flow generation.

The visible trailer is not an independent peer identifier: both spokes can carry the same source endpoint. If the hub chooses an existing IPsec state that belongs to the wrong peer or key generation, authentication and learning can fail even though packets reach the hub. In order to establish that sequence for a particular packet, the Cisco Technical Assistance Center (TAC) must correlate the chosen SA with its peer and generation. The packet trailer alone cannot prove it.

The same public IP alone does not establish a collision. Different remote ports normally produce different transport keys. Also check the chosen hub-local address, local port, and protocol. If a third peer advertises the same endpoint, it can also compete for that key; the failure pattern depends on event order.

Private-as-Public Collisions with Repeated Branch Addresses

Cisco bug ID CSCwx12986 can cause an Edge to advertise a private RFC 1918 tuple in its public TLOC attributes after an automatic source-port hop during NAT comparison. If different branches reuse the same private address and port, those incorrect public attributes can match across peers and expose the hub session-ownership problem.

For example, both spokes can use **192.168.11.11** behind separate branch firewalls. Each firewall keeps its branch network separate and translates the source address. If an upstream provider NAT gateway translates it again, the path has two NAT stages, or double NAT. Separate virtual routing and forwarding (VRF) contexts are another way to keep overlapping branch addresses isolated.

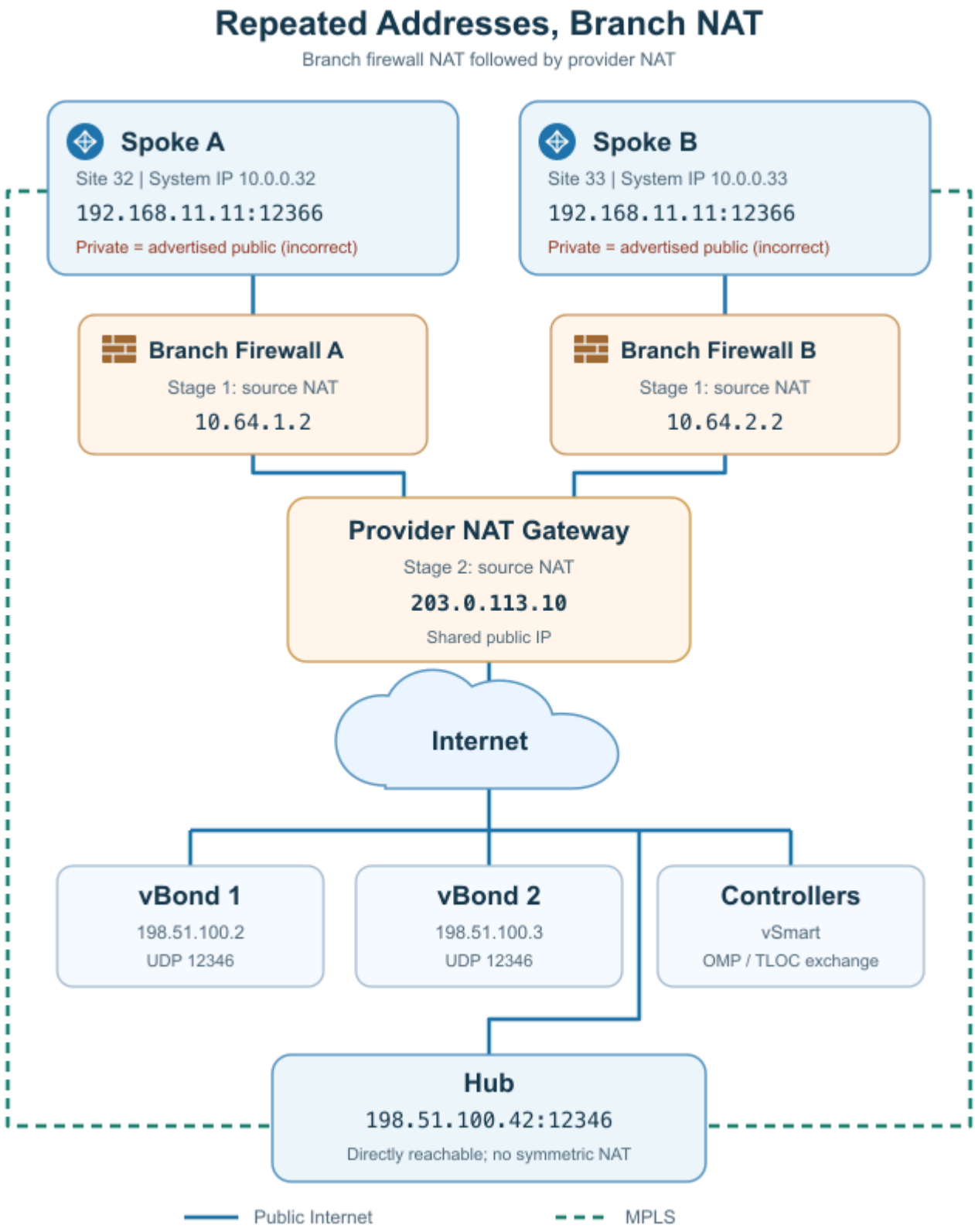


Figure 2. Branch firewall and provider NAT translations. The incorrect public TLOC attributes retain the original private endpoint, while packets use the translated source addresses.

If both spokes encounter the defect on their public-internet TLOCs and use the same post-hop source port, the hub can receive these advertisements:

Peer	Private Endpoint After Port Hop	Incorrect Public TLOC Endpoint
A	192.168.11.11:12366	192.168.11.11:12366
B	192.168.11.11:12366	192.168.11.11:12366

Note: Repeated private addresses and double NAT are not prerequisites for Cisco bug ID CSCwx12986 itself. The additional cross-peer collision requires matching advertised IP:port values and the same hub-local address, port, and protocol. The same private IP with different advertised ports does not establish this collision.

Public and private address equality is normal on a transport without NAT. An RFC 1918 address in a public field is suspicious only when it contradicts the intended path and the observed translation. Unique branch private addresses avoid the repeated-private-tuple overlap in this example; they do not prevent the separate translated-public port-reuse scenario described earlier.

NAT Endpoint Reassignment

An endpoint can also be reassigned to another spoke after a NAT mapping expires while the hub still retains old peer state. Compare the mapping history with the hub session lifecycle in order to identify overlap between the old and new owners.

Symptoms and Impact

Typical indicators include:

- Control connections remain Up while the affected BFD paths are Down or fail to recover.
- Two different system IPs advertise the same public IP:port.
- The hub retains both logical TLOCs, but a filtered platform session view is missing one peer or shows inconsistent system IP, remote color, discriminator, tunnel, or SA information.
- A transport withdrawal or recreation for one peer disturbs another peer with the same endpoint.
- IPsec input drops can increase on the affected path.

The failure can make the affected transport path unusable. If this is the only usable transport path for a spoke, the spoke is isolated from the overlay. At a large hub, many spokes sharing one NAT address increase the population requiring correlation, but the number of affected paths must be measured; it cannot be inferred from the number of sites behind that address.

Diagnose the Problem

Step 1. Record the Software and Spoke Transport State

On each affected Cisco IOS XE Edge, replace **<affected-color>** with the actual transport color, for example **biz-internet** or **public-internet**:

<#root>

```
show clock detail
```

```
show version | include Cisco IOS XE Software
```

```
show sdwan control local-properties
```

```
show sdwan control connections | include <affected-color>
```

```
show sdwan omp peers
```

Record the private/public address and port, interface, color, NAT type, site ID, and control state. After the first complete local-properties sample, use this compact filter to retain the interface headings and the chosen transport row. Replace **TenGigabitEthernet0/1/0.10** with the affected interface:

<#root>

```
show sdwan control local-properties "wan-interface-list" | include INTERFACE|PUBLIC|TenGigabitEthernet0/
```

For the primary scenario, the relevant fields can read on both spokes. This is a field summary, not the native column layout:

Field	Spoke A	Spoke B
System IP	10.0.0.32	10.0.0.33
Color	public-internet	public-internet
Private endpoint	192.168.11.11:12346	192.168.12.12:12346
Public endpoint	203.0.113.10:12346	203.0.113.10:12346
NAT type	A	A
vSmart controls on transport	Up	Up

An Up control connection verifies that controller relationship. It does not verify a spoke-to-hub BFD path or uniqueness of the public endpoint.

Step 2. Find Duplicate Endpoints in OMP

On the Cisco IOS XE hub:

<#root>

show clock detail

show sdwan omp tlocs table | include 203.0.113.10

On the SD-WAN Controller (vSmart):

<#root>

show clock

show omp tlocs | tab | include 203.0.113.10

This Controller output uses normalized identifiers. IPv6 and region columns are omitted:

<#root>

vSmart# show omp tlocs | tab | include 203.0.113.10

```
TLOC IP     COLOR            ENCAP FROM PEER  STATUS KEY PUBLIC IP     PORT  PRIVATE IP     PORT
10.0.0.32   public-internet ipsec 10.0.0.5   C,R    1   203.0.113.10 12346 192.168.11.11 12346
                              10.0.0.32  C,I,R  1   203.0.113.10 12346 192.168.11.11 12346
10.0.0.33   public-internet ipsec 10.0.0.5   C,R    1   203.0.113.10 12346 192.168.12.12 12346
                              10.0.0.33  C,I,R  1   203.0.113.10 12346 192.168.12.12 12346
```

Compare the TLOC system IP, color, encapsulation, public IP, and public port. Multiple paths to the same logical TLOC from different controllers are normal. The collision candidate is two different peer identities with the same effective endpoint.

Repeat the filtered command after 30-60 seconds and retain both timestamps. Persistent overlap is useful evidence; perfectly simultaneous commands are not required. For a short transient, collect OMP events and corresponding session lifecycle records with TAC.

For two suspect peers, preserve their individual TLOC table entries:

<#root>

```
show sdwan omp tlocs table | include 10.0.0.32
```

```
show sdwan omp tlocs table | include 10.0.0.33
```

For a compact view of the IPv4 TLOC details on Cisco IOS XE, use this filter. Replace the system IP with the peer under investigation and retain the quotation marks around the **ip <system-IP>** selector:

<#root>

```
show sdwan omp tlocs "ip 10.0.0.32" | exclude not set| 0|None|efault|::
```

```
show sdwan omp tlocs "ip 10.0.0.33" | exclude not set| 0|None|efault|::
```

Compare the public and private address/port attributes for both peers. The filter omits matching unset, zero-valued, default, and IPv6 lines from this view.

Step 3. Compare Logical TLOCs with Hub Session Ownership

Inspect the TLOC state delivered to FTM. Compare the system IPs, colors, public IP addresses, and public UDP ports of both the peers.

<#root>

```
show platform software sdwan ttmd tlocs
```

Run these commands in order to compare Forwarding Table Manager (FTM), BFD, and platform session records:

<#root>

```
show sdwan ftm bfd tloc | include 10.0.0.32
```

```
show sdwan ftm bfd tloc | include 10.0.0.33
```

```
show sdwan bfd sessions | include 10.0.0.32
```

```
show sdwan bfd sessions | include 10.0.0.33
```

```
show platform software sdwan session dstip 203.0.113.10

show platform software sdwan session remote-sysip 10.0.0.32 detail

show platform software sdwan session remote-sysip 10.0.0.33 detail
```

This example shows the pattern with normalized identifiers. Column labels are added to the filtered FTM rows for clarity; they are not part of the filtered command output:

<#root>

```
Hub# show sdwan ftm bfd tloc | include 10.0.0.32

System IP  Remote color     Encap  TLOC index  State
10.0.0.32  public-internet  IPSEC  32827       DOWN

Hub# show sdwan ftm bfd tloc | include 10.0.0.33

System IP  Remote color     Encap  TLOC index  State
10.0.0.33  public-internet  IPSEC  32828       DOWN

Hub# show platform software sdwan session dstip 203.0.113.10

RemoteSysIP Color        Proto SrcIp         SPort DstIp        DPort DPubIp       PPort BFD-LD TUN-ID
10.0.0.32   mpls         IPSEC 192.168.9.42  12346 203.0.113.10 12346 203.0.113.10 12346 20184  798
10.0.0.32   biz-internet IPSEC 192.168.10.42 12346 203.0.113.10 12346 203.0.113.10 12346 20183  799
```

OMP and FTM retain both logical peer identities, while this session lookup shows only Spoke A for the shared destination. Spoke A also has a Down FTM state; presence in the platform session table does not indicate a healthy path. The FTM TLOC index is distinct from the BFD local discriminator (**BFD-LD**) in the session table. This pattern is a reason to investigate ownership across layers. A missing CLI row alone does not establish packet loss.

In this table, **Color** represents the local transport; use **Remote color** in the detailed view in order to identify the remote TLOC. Compare **DstIp** and **DPubIp** as well: a learned actual destination can differ from the public endpoint associated with the session. Use the peer-specific view if a destination-only filter no longer matches the learned address.

An **Invalid** next hop, SA identifier of 0, or **In HW: FALSE** provides additional evidence rather than a unique signature. These fields can also occur during ordinary setup or underlay failure.

Step 4. Verify What the NAT Gateway Actually Does

Ask the NAT or service-provider operator for concurrent UDP translation records with these fields:

```
Spoke           Inside source       Translated source    External destination
A               192.168.11.11:12346 203.0.113.10:12346   198.51.100.2:12346
B               192.168.12.12:12346 203.0.113.10:12346   198.51.100.3:12346
A               192.168.11.11:12346 203.0.113.10:12346   198.51.100.42:12346
B               192.168.12.12:12346 203.0.113.10:45678   198.51.100.42:12346
```

This is an illustrative translation table. Use the equivalent diagnostic command for your NAT platform. On a Cisco IOS XE NAT gateway, start with:

<#root>

```
show ip nat translations | include 203.0.113.10
```

Retain both the inside source and destination columns. A list containing only the translated source cannot show why reuse is valid toward different vBonds or which spoke owns the hub mapping.

Capture the affected UDP flows before and after NAT for both vBonds and the hub. Preserve full packet lengths so the VCONTROL trailer is not truncated. Use a bounded Switched Port Analyzer (SPAN) or Embedded Packet Capture (EPC) collection agreed with TAC. Compare the same flow across capture points; do not infer latency from unsynchronized capture clocks.

Step 5. Inspect VCONTROL in Wireshark

Use the Lua decoder in the appendix for the IPv4 BFD/VCONTROL format described here. Save the code as **sdwan_vctrl.lua** in the same folder as your capture. This diagnostic decoder displays visible fields; it does not decrypt ESP or verify authentication. Its scope is complete, non-nested IPv4/UDP packets with the expected BFD/VCONTROL flag and trailer structure.

Run the **Wireshark** command from the folder that contains the decoder and your capture:

<#root>

```
wireshark -n \

  -X lua_script:sdwan_vctrl.lua \

  -d udp.port==12346,sdwan_vctrl \

  -d udp.port==45678,sdwan_vctrl \
```

```
  -r hub.pcap
```

Replace the example ports with those in the capture. On Windows or macOS, use the full path to the Wireshark executable if it is not in PATH.

Alternatively, load the **Lua script** at Wireshark startup, choose a relevant **UDP packet**, and navigate to **Analyze > Decode As**. Assign the applicable source and destination UDP port entries to **SDWAN_VCTRL / SD-WAN VCTRL (IPv4)**. An explicit **UDPENCAP** entry on the other port can take precedence over this decoder. Apply the change to this analysis session.

Apply this display filter:

```
sdwan_vctrl && ip.addr == 203.0.113.10
```

Expand **SD-WAN VCTRL > Visible VCONTROL trailer** and compare:

```
Outer IPv4 source:       203.0.113.10
Outer UDP source port:   45678
Carried source IPv4:     203.0.113.10
Carried source UDP port: 12346
Carried destination:     198.51.100.42:12346
```

These illustrative values describe initial tunnel establishment before data-plane learning completes. They show the actual hub mapping for B alongside its carried endpoint. NAT modifies the outer headers; it does not rewrite the endpoint information inside this trailer. This mismatch alone does not demonstrate a fault.

Run the **TShark** command for a compact export:

<#root>

```
tshark -n -X lua_script:sdwan_vctrl.lua \

  -d udp.port==12346,sdwan_vctrl \

  -d udp.port==45678,sdwan_vctrl \

  -r hub.pcap -Y sdwan_vctrl -T fields \

  -e frame.number -e ip.src -e udp.srcport -e ip.dst -e udp.dstport \
```

```
  -e sdwan_vctrl.spi \

  -e sdwan_vctrl.src_ip -e sdwan_vctrl.src_port \

  -e sdwan_vctrl.dst_ip -e sdwan_vctrl.dst_port
```

Without a custom decoder, the IPv4 format described here has this 17-byte type-length-value (TLV) trailer at the end of the UDP payload. Link-layer padding is outside the UDP payload:

```
81 00 0f cb 00 71 0a 30 3a c6 33 64 2a 30 3a 00 11
|  |     |           |     |           |     |
|  |     |           |     |           |     +-- Complete trailer length: 17
|  |     |           |     |           +-------- Destination port: 12346
|  |     |           |     +-------------------- Destination: 198.51.100.42
|  |     |           +-------------------------- Source port: 12346
|  |     +-------------------------------------- Source: 203.0.113.10
|  +-------------------------------------------- TLV length: 15
+----------------------------------------------- Type: 0x81
```

Do not apply this fixed layout to IPv6, other control formats, truncated packets, or arbitrary encrypted data. An empty decoder filter can mean the packet is outside its supported format. A readable source endpoint is also not an authenticated system-IP label.

Note: Generic ESP sequence warnings in Wireshark are not router anti-replay diagnostics. Correlate the packet with router counters and logs before you draw an authentication or ownership conclusion.

Step 6. Confirm the Affected Path and Preserve Evidence

On the hub, take two QuantumFlow Processor (QFP) counter samples around the observed failure:

<#root>

```
show clock detail

show platform hardware qfp active statistics drop | include Ipsec

show sdwan bfd sessions | include 10.0.0.32

show sdwan bfd sessions | include 10.0.0.33
```

Use counter differences rather than absolute totals. Aggregate IPsec drops can have other causes. Confirm packet arrival on the affected transport and correlate it with the peer, endpoint, BFD discriminator, tunnel, SA, and event time.

Provide TAC with the two OMP snapshots, spoke local properties and controls, hub session details for both peers, BFD state, translation records, bounded captures, and software versions. If already available, include the relevant admin-techs and connection/withdrawal history. Avoid repeated collection of entire tables at a large hub.

Test service reachability separately and record the actual chosen transport. A zero-second BFD Up indication is not sustained recovery.

Workarounds and Limitations

Evaluate these mitigation options with Cisco TAC and the NAT operator. The limitation concerns duplicate advertised endpoints reaching a common hub; it does not mean that symmetric NAT, shared public addresses, or repeated private addressing are generally unsupported. Recovery procedures and mitigation effectiveness depend on the deployment.

Use Stable Endpoint-Independent Mappings with Unique Allocations

One NAT-side prevention option for the illustrated sequence is to give each distinct spoke transport endpoint a stable, unique translated UDP endpoint across vBond, controller, and hub destinations. Request both endpoint independence and uniqueness across spokes. Verify the result in translation records and captures, also after NAT gateway failover or state recreation.

For example, A can consistently use **203.0.113.10:40001**, while B consistently uses **203.0.113.10:40002**. Static translations or a suitable allocation policy can provide this behavior, subject to the NAT platform and access design.

Endpoint-independent mapping does not require unrestricted inbound access. Preserve the required firewall controls. A NAT policy correction does not automatically remove previously advertised or programmed session state; coordinate a new discovery cycle and recovery with TAC.

Disable Port Overlap Across Different Spokes

Where supported, reserve translated UDP endpoints uniquely across the relevant SD-WAN spokes. Do not reuse them solely because their destinations differ. This directly removes the simultaneous duplicate-vBond mapping shown earlier, even if the NAT gateway retains some destination-dependent behavior.

The setting can consume more ports or require additional public addresses. It does not by itself make the mapping for each spoke endpoint-independent, and it does not resolve stale receiver state after later endpoint reassignment.

With multiple NAT stages or redundant NAT gateways, check the allocation visible at the final Internet boundary. Uniqueness inside one NAT instance does not establish uniqueness across other instances that share the same public address.

Use Port Offset for a Small, Controlled Group

The **port-offset** command provides values 0 through 19. For example, Spoke A can retain offset 0, and Spoke B can use offset 1. Run these configuration commands on Spoke B:

<#root>

```
Spoke-B# config-transaction

Spoke-B(config)# system

Spoke-B(config-system)# port-offset 1

Spoke-B(config-system)# commit

Spoke-B(config-system)# end

Spoke-B# show sdwan control local-properties
```

Verify the active source and translated ports after the change. A change to this system setting can rebuild control and data connections across transports. It is not a nondisruptive diagnostic step.

The 20 offsets provide 20 distinct base-port groups with offset 0 as one choice. They are not a general limit of 20 SD-WAN routers behind one public IP. This particular workaround provides only 20 unique offset choices and requires the NAT gateway to preserve them and distinguish between them. It fails as a uniqueness scheme when offsets are reused. Port hopping does not add more offset identities.

For example, 124 routers cannot all receive unique offsets behind one shared address. An offset-only scheme with at most 20 routers per public-address group requires at least seven groups. This arithmetic is not a capacity recommendation or a substitute for verification of NAT behavior. A NAT gateway that rewrites or overlaps translated ports can defeat the scheme even with fewer routers.

Disable Source-Port Preservation or Enable Random Allocation

A policy without a preservation preference can reduce the tendency for many spokes to receive **:12346** toward different vBonds. Randomized allocation can distribute those mappings over a larger port pool.

However, neither change alone guarantees collision avoidance. A randomized allocator can still reuse a port across destinations, change a mapping toward the hub, or reassign an expired endpoint while old hub state remains. Verify whether allocation is unique across subscribers and destinations. Random ports alone do not establish uniqueness.

Review source-port preservation and allocation settings on the NAT gateway with its operator. The applicable configuration depends on the NAT platform; verify the resulting mappings toward both vBonds and the hub with translation records and packet captures.

Translated ports can fall outside the original SD-WAN base-port range. Review firewall rules so they do not reject legitimate translated traffic solely because the source port changed.

Separate Public Addresses or Reserve Port Blocks

Dedicated translated addresses, or nonoverlapping port blocks reserved per spoke, can provide distinct advertised endpoints for different active spokes. The policy must apply consistently across the discovery and hub paths and across redundant NAT nodes. Additional address space and provider support can be required. Dynamic reuse after mapping retirement still requires correct hub session lifecycle management.

Retain an Alternate Transport and Coordinate Recovery

A verified MPLS or other eligible path can preserve service while the affected Internet path is investigated. An underlay access control list (ACL) that blocks the shared NAT address provides containment only: it can remove Internet redundancy for every spoke behind that address and does not withdraw their OMP advertisements automatically.

A control clear, interface reset, NAT-state clear, or hub reload can temporarily recreate state. None guarantees prevention of recurrence. Do not perform broad clears as a first diagnostic step. Preserve the evidence, correct the allocation conditions where possible, and agree on the smallest recovery action with TAC. Validate sustained BFD Up and bidirectional traffic on the recovered path.

Related Software Defects

Cisco Bug ID	Scope Relevant to this Article
CSCww45867	Documentation of the hub session-ownership limitation when different peers advertise the same public TLOC endpoint. This record tracks the documentation update, rather than a software correction.
CSCwx12986	An automatic port-hop/NAT-learning sequence can leave a private tuple in the public TLOC fields. If branches reuse the same private IP:port, the incorrect public endpoints can match across peers. Correcting this producer defect does not prevent valid translated-public port reuse.

Cisco Bug ID	Scope Relevant to this Article
CSCwx48929	NAT classification can incorrectly become E when discovery returns to the same vBond. Correct classification does not prevent the valid different-vBond port-reuse scenario.

Corrections for Cisco bug ID CSCwx12986 and Cisco bug ID CSCwx48929 do not guarantee unique advertised public endpoints and do not, by themselves, eliminate this collision scenario. Closure of the Cisco bug ID CSCww45867 documentation record does not establish a change in software behavior. Confirm applicable limitations and mitigations with Cisco TAC.

Appendix: Decode the IPv4 VCONTROL Trailer

Save this code as **sdwan_vctrl.lua**. The diagnostic decoder does not authenticate packets or decrypt their payloads. It uses the **sdwan_vctrl** display filter and registers an explicit **Decode As** choice. The packet capture remains unchanged.

```
-- Cisco SD-WAN IPv4 VCONTROL trailer decoder.
-- Explicit UDP "Decode As" only; does not decrypt ESP.
-- Scope: complete ordinary IPv4/UDP packets, one IP/UDP layer.

local p = Proto("sdwan_vctrl", "SD-WAN VCTRL (IPv4)")

local f = {
    spi       = ProtoField.uint32("sdwan_vctrl.spi",
                    "Wire SPI including control flags", base.HEX),
    sequence  = ProtoField.uint32("sdwan_vctrl.sequence",
                    "ESP sequence number", base.DEC),
    type      = ProtoField.uint8("sdwan_vctrl.type",
                    "Trailer type", base.HEX),
    tlv_len   = ProtoField.uint16("sdwan_vctrl.tlv_len",
                    "TLV length", base.DEC),
    src_ip    = ProtoField.ipv4("sdwan_vctrl.src_ip",
                    "Carried source IPv4"),
    src_port  = ProtoField.uint16("sdwan_vctrl.src_port",
                    "Carried source UDP port", base.DEC),
    dst_ip    = ProtoField.ipv4("sdwan_vctrl.dst_ip",
                    "Carried destination IPv4"),
    dst_port  = ProtoField.uint16("sdwan_vctrl.dst_port",
                    "Carried destination UDP port", base.DEC),
    total_len = ProtoField.uint16("sdwan_vctrl.total_len",
                    "Complete trailer length", base.DEC)
}
p.fields = f

-- Field extractors must be created outside the dissector.
local get_udp_length = Field.new("udp.length")
local get_ip_version = Field.new("ip.version")
local get_ip_proto   = Field.new("ip.proto")

function p.dissector(tvb, pinfo, tree)
    local lengths = { get_udp_length() }
    local versions = { get_ip_version() }
    local protocols = { get_ip_proto() }

    -- Deliberately skip nested/tunnelled protocol stacks.
```

```
    if #lengths ~= 1 or #versions ~= 1 or #protocols ~= 1 then
        return 0
    end
    if tonumber(versions[1].value) ~= 4 or
       tonumber(protocols[1].value) ~= 17 then
        return 0
    end

    -- UDP length includes its 8-byte header.
    local n = tonumber(lengths[1].value) - 8

    -- Require the full declared payload, not a truncated fragment.
    if n < 25 or tvb:reported_len() ~= n or tvb:len() ~= n then
        return 0
    end

    -- Exact BFD + VCTRL control-bit combination.
    local spi = tvb(0, 4):uint()
    if math.floor(spi / 0x10000000) ~= 0xA then
        return 0
    end

    -- Trailer is at the end of UDP payload, not Ethernet frame.
    local o = n - 17

    if tvb(o, 1):uint() ~= 0x81 or
       tvb(o + 1, 2):uint() ~= 15 or
       tvb(o + 15, 2):uint() ~= 17 then
        return 0
    end

    local root = tree:add(p, tvb())
    root:add(f.spi,      tvb(0, 4))
    root:add(f.sequence, tvb(4, 4))

    local trailer = root:add(tvb(o, 17),
                             "Visible VCONTROL trailer")
    trailer:add(f.type,      tvb(o,      1))
    trailer:add(f.tlv_len,   tvb(o + 1,  2))
    trailer:add(f.src_ip,    tvb(o + 3,  4))
    trailer:add(f.src_port,  tvb(o + 7,  2))
    trailer:add(f.dst_ip,    tvb(o + 9,  4))
    trailer:add(f.dst_port,  tvb(o + 13, 2))
    trailer:add(f.total_len, tvb(o + 15, 2))

    pinfo.cols.protocol = "SDWAN-VCTRL"
    return n
end

DissectorTable.get("udp.port"):add_for_decode_as(p)
```

Related Information

- RFC 7857, Section 3: Port Overlapping Behavior
- RFC 4787: NAT Behavioral Requirements for Unicast UDP
- RFC 5737: IPv4 Address Blocks Reserved for Documentation
- Cisco Catalyst SD-WAN Design Guide

- Cisco IOS XE Catalyst SD-WAN System Commands: port-offset
- Cisco SD-WAN Port Offset Configuration
- Wireshark Command-Line Reference
- TShark Command-Line Reference
- Technical Support & Documentation - Cisco Systems