Quick Start Guide - Data Collection for Various SD-WAN Issues

Contents

Introduction

This document describes several SD-WAN issues along relevant data that must be collected in advance before you open a TAC case to improve the speed of troubleshooting and/or problem resolution. This document is broken up into two main technical sections: vManage and Edge routers. Relevant outputs and command syntax are provided dependent upon the device in question.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Cisco's SDWAN architecture
- General understanding of solution, including vManage controller as well as cEdge (IOS-XE SD-WAN routers) and vEdge devices (ViptelaOS routers)

Components Used

This document is not restricted to specific software and hardware versions.

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.

Base Information Requested

- Describe the problem and its impact on your network and users: Describe an expected behavior.Describe in details observed behavior.Prepare a topology diagram with addressing if possible, even if this is hand-drawn.
- When did the problem start? Note the day and time the problem was first observed/noticed.
- What could be a potential trigger of the problem? Document any recent changes made prior to when the problem started.Note any specific actions or events that occurred that can have triggered the problem to start.Does this problem correspond to any other network events or actions?
- What is the frequency of the problem? Was this a one-time occurrence?If not, how often does the problem happen?
- Provide information about the device(s) in question: If specific devices are affected (not random), what do they have in common?System-IP and Site-ID for each device.If the issue is on a vManage cluster, provide the node details (if not the same across all nodes in the cluster).For general issues inside of the vManage GUI, capture all screenshots to a file that show error messages or other anomalies/disparties that need to be investigated.
- Provide information on desired outcome from the TAC and your priorities: Do you want to recover from the failure as soon as possible or find out the root cause of the failure?

vManage

The issues here are common problem conditions reported for vManage along with useful outputs for each problem that must be collected in addition to an **admin-tech** file(s). For cloud-hosted controllers, Technical Assistance Center (TAC) engineer can have access to collect the required **admin-tech** outputs for the devices based on the feedback in the Base Information Requested section if you provide explicit consent for this. However, we recommend to capture **admin-tech** outputs if the steps described here to ensure the data contained within is relevant to the time of the problem. This is specifically true if the problem isn't persistent, meaning that the problem can disappear by the time TAC is engaged. For on-prem controllers**,** an **admin-tech** must also be included with each set of data here**.** For a vManage cluster, ensure you capture an **admin-tech** for each node in the cluster or only the affected node(s).

Slowness/Sluggishness

Problem Report: Slowness in accessing the vManage GUI, latency when performing operations inside of the GUI, general slowness or sluggishness seen within vManage

Step 1. Capture 2-3 instances of a thread print, rename each **thread-print** file with a numerical designation after each (note the use of the username that you log into vManage with in the file path), example:

```
vManage# request nms application-server jcmd thread-print | save /home/<username>/thread-print.1
```

Step 2. Log in to **vshell** and run **vmstat** as below:

```
vManage# vshell
vManage:~$ vmstat 1 10
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
r b swpd free buff cache si so bi bo in cs us sy id wa st
1 0 0 316172 1242608 5867144 0 0 1 22 3 5 6 1 93 0 0
0 0 0 316692 1242608 5867336 0 0 0 8 2365 4136 6 1 93 0 0
0 0 0 316204 1242608 5867344 0 0 0 396 2273 4009 6 1 93 0 0
0 0 0 316780 1242608 5867344 0 0 0 0 2322 4108 5 2 93 0 0
0 0 0 318136 1242608 5867344 0 0 0 0 2209 3957 9 1 90 0 0
0 0 0 318300 1242608 5867344 0 0 0 0 2523 4649 5 1 94 0 0
1 0 0 318632 1242608 5867344 0 0 0 44 2174 3983 5 2 93 0 0
0 0 0 318144 1242608 5867344 0 0 0 64 2182 3951 5 2 94 0 0
0 0 0 317812 1242608 5867344 0 0 0 0 2516 4289 6 1 93 0 0
0 0 0 318036 1242608 5867344 0 0 0 0 2600 4421 8 1 91 0 0
vManage:~$
```

Step 3. Collect additional details from the **vshell**:

```
vManage:~$ top (press '1' to get CPU counts)
vManage:~$ free -h
vManage:~$ df -kh
```

Step 4. Capture all NMS services diagnostics:

```
vManage# request nms application-server diagnostics
vManage# request nms configuration-db diagnostics
vManage# request nms messaging-server diagnostics
vManage# request nms coordination-server diagnostics
vManage# request nms statistics-db diagnostics
```

API Failures/Issues

Problem Report: API calls fail to return any data or the correct data, general problems executing queries

Step 1. Check the memory available:

```
vManage:~$ free -h
total used free shared buff/cache available
Mem: 31Gi 24Gi 280Mi 60Mi 6.8Gi 6.9Gi
Swap: 0B 0B 0B
vManage:~$
```

Step 2. Capture 2-3 instances of a thread print with a 5-second gap in between, rename each **thread-print** file with a numerical designation after each run of the command (note the use of the username that you log into vManage with in the file path):

```
vManage# request nms application-server jcmd thread-print | save /home/<username>/thread-print.1
<WAIT 5 SECONDS>
vManage# request nms application-server jcmd thread-print | save /home/<username>/thread-print.2
```

Step 3. Collect details for any active HTTP sessions:

```
vManage# request nms application-server jcmd gc-class-histo | i
io.undertow.server.protocol.http.HttpServerConnection
```

Step 4. Provide this details:

1. API calls executed

2. Invocation frequency

3. Log in method (i.e., usage of a single token to execute subsequent API calls or usage of basic authentication to execute the call and then logout)

4. Is the JSESSIONID being re-used?

> **Note** Starting from 19.2 vManage software, only token-based authentication is supported for API calls. For more details on token generation, timeout, and expiration, see this link.

Deep Packet Inspection (DPI) Stats/Slowness

Problem Report: With DPI enabled, statistics processing can be slow or introduce slowness inside of the vManage GUI.

Step 1. Check the disk size allocated for DPI inside of vManage by navigating to **Administration > Settings > Statistics Database > Configuration**.

Step 2. Check the index health by running the following CLI command from vManage:

```
vManage# request nms statistics-db diagnostics
```

Step 3. Confirm if any API calls related to DPI stats are executed externally.

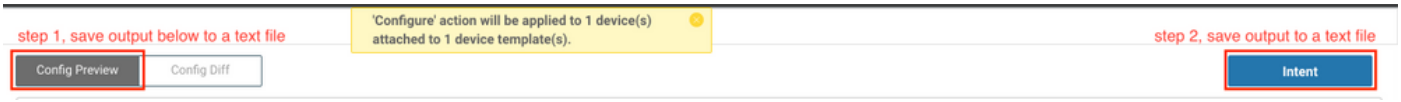
Step 4. Check disk I/O stats with help of this CLI command from vManage:

```
vManage# request nms application-server diagnostics
```

Template Push Failures

Problem Report: Template push or device template update fails or times out.

Step 1. Capture the **Config Preview** and **Intent** config from vManage before you click the **Configure Devices** button (navigation example provided here):



Step 2. Enable **viptela.enable.rest.log** from the **logsettings** page (this must be disabled after capturing the required information):

```
https://<vManage IP>:8443/logsettings.html
```

Step 3. If the template push failure involves a NETCONF issue or error, enable **viptela.enable.device.netconf.log** in addition to the REST log in Step 1. Note that this log must

also be disabled after the outputs from Step 3 and Step 4 are captured.

Step 4. Attempt to attach the failed template again from vManage and capture an **admin-tech** using this CLI (capture this for each node of for a cluster):

```
vManage# request admin-tech
```

Step 5. Provide screenshots from the task in vManage and the Config Diff to confirm the failure details along with any CSV files used for the template.

Step 6. Include details about the failure and task, including the time of the failed push, **system-ip** of the device that failed, and error message you see in the vManage GUI.

Step 7. If a template push failure happens with an error message reported for the configuration by the device itself, collect an **admin-tech** from the device as well.

Cluster-Related Issues

Problem Report: Cluster instability leading to GUI timeouts, sluggishness, or other anomalies.

Step 1. Capture the output from **server_configs.json** from each vManage node in the cluster. For example:

```
vmanage# vshell
vmanage:~$ cd /opt/web-app/etc/
vmanage:/opt/web-app/etc$ more server_configs.json | python -m json.tool
{
"clusterid": "",
"domain": "",
"hostsEntryVersion": 12,
"mode": "SingleTenant",
"services": {
"cloudAgent": {
"clients": {
"0": "localhost:8553"
},
"deviceIP": "localhost:8553",
"hosts": {
"0": "localhost:8553"
},
"server": true,
"standalone": false
},
"container-manager": {
"clients": {
"0": "169.254.100.227:10502"
},
"deviceIP": "169.254.100.227:10502",
"hosts": {
"0": "169.254.100.227:10502"
},
"server": true,
"standalone": false
},
"elasticsearch": {
"clients": {
"0": "169.254.100.227:9300",
```

```
"1": "169.254.100.254:9300",
"2": "169.254.100.253:9300"
},
"deviceIP": "169.254.100.227:9300",
"hosts": {
"0": "169.254.100.227:9300",
"1": "169.254.100.254:9300",
"2": "169.254.100.253:9300"
},
"server": true,
"standalone": false
},
"kafka": {
"clients": {
"0": "169.254.100.227:9092",
"1": "169.254.100.254:9092",
"2": "169.254.100.253:9092"
},
"deviceIP": "169.254.100.227:9092",
"hosts": {
"0": "169.254.100.227:9092",
"1": "169.254.100.254:9092",
"2": "169.254.100.253:9092"
},
"server": true,
"standalone": false
},
"neo4j": {
"clients": {
"0": "169.254.100.227:7687",
"1": "169.254.100.254:7687",
"2": "169.254.100.253:7687"
},
"deviceIP": "169.254.100.227:7687",
"hosts": {
"0": "169.254.100.227:5000",
"1": "169.254.100.254:5000",
"2": "169.254.100.253:5000"
},
"server": true,
"standalone": false
},
"orientdb": {
"clients": {},
"deviceIP": "localhost:2424",
"hosts": {},
"server": false,
"standalone": false
},
"wildfly": {
"clients": {
"0": "169.254.100.227:8443",
"1": "169.254.100.254:8443",
"2": "169.254.100.253:8443"
},
"deviceIP": "169.254.100.227:8443",
"hosts": {
"0": "169.254.100.227:7600",
"1": "169.254.100.254:7600",
"2": "169.254.100.253:7600"
},
"server": true,
"standalone": false
},
```

```
"zookeeper": {
"clients": {
"0": "169.254.100.227:2181",
"1": "169.254.100.254:2181",
"2": "169.254.100.253:2181"
},
"deviceIP": "169.254.100.227:2181",
"hosts": {
"0": "169.254.100.227:2888:3888",
"1": "169.254.100.254:2888:3888",
"2": "169.254.100.253:2888:3888"
},
"server": true,
"standalone": false
}
},
"vmanageID": "0"
}
```


Step 2. Capture details on which services are enabled or disabled for each node. For this, navigate to **Administration > Cluster Management** in the vManage GUI.

Step 3. Confirm underlay reachability on the cluster interface. For this, run **ping <ip-address>** from each vManage node in VPN 0 to the cluster interface IP of the other nodes.

Step 4. Collect diagnostics from all NMS services for each vManage node in the cluster:

```
vManage# request nms application-server diagnostics
vManage# request nms configuration-db diagnostics
vManage# request nms messaging-server diagnostics
vManage# request nms coordination-server diagnostics
vManage# request nms statistics-db diagnostics
```

Edge (vEdge/cEdge)

The issues here are common problem conditions reported for Edge devices along with useful outputs for each that must be collected. Ensure that for each problem, an **admin-tech** is collected for all necessary and relevant Edge devices. For cloud-hosted controllers, TAC can have access to collect the required admin-tech outputs for the devices based on the feedback in the **Base Information Requested** section. However, as with vManage, it can be necessary to capture these before you open a TAC case to ensure the data contained within is relevant to the time of the problem. This is specifically true if the problem isn't persistent, meaning that the problem can disappear by the time TAC is engaged.

Control Connections Not Forming Between Device and Controller

Problem Report: Control connection not forming from a vEdge/cEdge to one or more of the controllers

Step 1. Identify the local/remote error of the control connection failure:

- For vEdge: output of **show control connections-history** command.
- For cEdge: output of **show sdwan control connection-history** command.

Step 2. Confirm the state of the TLOC(s) and that any and all show 'up':

- For vEdge: output of **show control local-properties** command.

- For cEdge: output of **show sdwan control local-properties** command.

Step 3. For errors around timeouts or connection failures (i.e., DCONFAIL or VM_TMO), take control-plane captures on both the edge device as well as the controller in question:

- For controllers:

```
vManage# tcpdump vpn 0 interface eth1 options "-vvvvvv host 192.168.44.6"
tcpdump -p -i eth1 -s 128 -vvvvvv host 192.168.44.6 in VPN 0
tcpdump: listening on eth1, link-type EN10MB (Ethernet), capture size 128 bytes
20:02:07.427064 IP (tos 0xc0, ttl 61, id 50139, offset 0, flags [DF], proto UDP (17), length
168)
192.168.44.6.12346 > 192.168.40.1.12346: UDP, length 140
20:02:07.427401 IP (tos 0xc0, ttl 64, id 37220, offset 0, flags [DF], proto UDP (17), length
210)
192.168.40.1.12346 > 192.168.44.6.12346: UDP, length 182
```

- For vEdge:

```
vEdge-INET-Branch2# tcpdump vpn 0 interface ge0/2 options "-vvvvvv host 192.168.40.1"
tcpdump -p -i ge0_2 -vvvvvv host 192.168.40.1 in VPN 0
tcpdump: listening on ge0_2, link-type EN10MB (Ethernet), capture size 262144 bytes
20:14:16.136276 IP (tos 0xc0, ttl 64, id 55858, offset 0, flags [DF], proto UDP (17), length
277)
10.10.10.1 > 192.168.40.1.12446: [udp sum ok] UDP, length 249
20:14:16.136735 IP (tos 0xc0, ttl 63, id 2907, offset 0, flags [DF], proto UDP (17), length 129)
192.168.40.1.12446 > 10.10.10.1.12346: [udp sum ok] UDP, length 101
```

- For cEdge (capture below assumes the device was moved to CLI mode and an Access Control List (ACL) called **CTRL-CAP** was created to filter - see more details in the EPC capture example in the **Application/Network Performance** scenario):

```
cEdge-Branch1#config-transaction
cEdge-Branch1(config)# ip access-list extended CTRL-CAP
cEdge-Branch1(config-ext-nacl)# 10 permit ip host 10.10.10.1 host 192.168.40.1
cEdge-Branch1(config-ext-nacl)# 20 permit ip host 192.168.40.1 host 10.10.10.1
cEdge-Branch1(config-ext-nacl)# commit
cEdge-Branch1(config-ext-nacl)# end

cEdge-Branch1#monitor capture CAP control-plane both access-list CTRL-CAP buffer size 10
cEdge-Branch1#monitor capture CAP start

cEdge-Branch1#show monitor capture CAP buffer brief
------------------------------------------------------------------------
# size timestamp source destination dscp protocol
------------------------------------------------------------------------
0 202 0.000000 192.168.20.1 -> 50.50.50.3 48 CS6 UDP
1 202 0.000000 192.168.20.1 -> 50.50.50.4 48 CS6 UDP
2 220 0.000000 50.50.50.3 -> 192.168.20.1 48 CS6 UDP
3 66 0.000992 192.168.20.1 -> 50.50.50.3 48 CS6 UDP
4 220 0.000992 50.50.50.4 -> 192.168.20.1 48 CS6 UDP
5 66 0.000992 192.168.20.1 -> 50.50.50.4 48 CS6 UDP
6 207 0.015991 50.50.50.1 -> 12.12.12.1 48 CS6 UDP
```

Step 4. For other errors observed in the control connection history outputs and for more details on the issues described, please refer to the following guide .

Control Connections Flapping Between Edge Device and Controller

Problem Report: One or more control connections flap between a vEdge/cEdge and one or more controllers. This can be frequent, intermittent, or random in nature.

- Control connection flaps are generally the result of packet loss or forwarding issues between a device and a controller. Often times, this will be tied to **TMO** errors, depending on the directionality of the failure. To check this further, first verify the reason for the flap: For vEdge/controllers: output of **show control connections-history** command.For cEdge: output of **show sdwan control connection-history** command.
- Confirm the state of the TLOC(s) and that any and all show 'up' when the flapping is occurring: For vEdge: output of **show control local-properties** command.For cEdge: output of **show sdwan control local-properties** command.
- Collect packet captures on both the controller(s) and the edge device. Please refer to the **Control Connections Not Forming Between Device and Controller** section for details on capture parameters for each side.

Bidirectional Forwarding Detection (BFD) Sessions Not Forming or Flapping Between Edge Devices

Problem Report: BFD session is down or is flapping up and down between two edge devices.

Step 1. Collect the state of the BFD session on each device:

- For vEdge: output of **show bfd sessions** command.
- For cEdge: output of **show sdwan bfd sessions** command.

Step 2. Collect Rx and Tx packet counts on each edge router:

- For vEdge: output of **show tunnel statistics bfd** command.
- For cEdge: output of **show platform hardware qfp active feature bfd datapath sdwan summary** command.

Step 3. If counters do not increase for BFD session on one end of the tunnel in the outputs above, captures can be taken using ACLs to confirm if packets are being received locally. More details on this along with other validations that can be done can be found here .

Device Crashes

Problem Report: Device unexpectedly reloaded and problems with power are ruled out. Indications from the device are that it crashed potentially.

Step 1. Check the device to confirm if a crash or unexpected reload was observed:

- For vEdge: output of **show reboot history** command.
- For cEdge: output of **show sdwan reboot history** command.
- Alternatively, navigate to **Monitor > Network**, select the device, and then navigate to **System Status > Reboot** to confirm if any unexpected reloads were seen.

Step 2. If confirmed, capture an admin-tech from the device through vManage by navigating to **Tools > Operational Commands**. Once there, select the **Options** button for the device and select **Admin Tech**. Ensure all check boxes are checked, which will include all logs and core files on the device.

Application/Network Performance Degraded or Failing Between Sites

Problem Report: Application does not work/HTTP pages not load, slowness/latency in performance, failures after making policy or configuration changes

Step 1. Identify the source/destination IP pair for an application or flow exhibiting the problem.

Step 2. Determine all Edge devices in the path and collect an **admin-tech** from each through vManage.

Step 3. Take a packet capture on the edge devices at each site for this flow when the problem is seen:

- For vEdge: Enable Data Stream under **Administration > Settings** For **Hostname** field, enter the system IP of vManage. For **VPN**, enter **0**Ensure HTTPS is enabled under the **allow-service** configuration of the vManage VPN 0 interface.Follow the steps here to capture traffic on the service-side VPN interface.
- For cEdge: Move the cEdge(s) to CLI mode via **Configuration > Devices > Change Mode > CLI mode**On the cEdge(s), configure an extended ACL to match traffic bidirectionally. Make this as specific as possible to include protocol and port to limit the size and data in the capture.
- Configure Embedded Packet Capture (EPC) for the service-side interface in both directions, using the ACL created in (b) to filter the traffic. The capture can be exported to PCAP format and copied off the box. A sample configuration is provided here for GigabitEthernet0/0/0 on a router using an ACL named **BROKEN-FLOW**:

```
monitor capture CAP interface GigabitEthernet0/0/0 both access-list BROKEN-FLOW buffer size 10
monitor capture CAP start

show monitor capture CAP parameter
show monitor capture CAP buffer [brief]

monitor capture CAP export bootflash:cEdge1-Broken-Flow.pcap
```

- Configure Packet Trace for the traffic in both directions, using the ACL created in (b) to filter the traffic. A sample configuration is provided below:

```
debug platform packet-trace packet 2048 fia-trace
debug platform packet-trace copy packet input l3 size 2048
debug platform condition ipv4 access-list BROKEN-FLOW both
debug platform condition start

show platform packet-trace summary
show platform packet-trace packet all | redirect bootflash:cEdge1-PT-OUTPUT.txt
```

Step 4. If possible, repeat Step 3 in a working scenario for comparison.

> **Tip**: If there are no other ways to copy the corresponding files off of the cEdge directly, the files can be copied to vManage first using method described here. Run the command on vManage:
> **request execute scp -P 830 <username>@<cEdge system-IP>:/bootflash/<filename> .**
> This file will then be stored in **/home/<username>/** directory for the username you used to login to vManage. From there, you can use Secure Copy Protocol (SCP) of Secure File

Transfer Protocol (SFTP) to copy file off a vManage using a third-party SCP/SFTP client or a Linux/Unix machine CLI with OpenSSH utilities.