Troubleshooting CPS vDRA

Overview

CPS vDRA is a functional element that ensures that all Diameter sessions established over Gx, Rx interfaces and for unsolicited application reporting, the Sd interface for a certain IP-CAN session reach the same PCRF or destined PCRF when multiple and separately addressable PCRFs have been deployed in a Diameter realm.

General Troubleshooting

- Run the following command in CLI to view the diagnostics status. Verify that the status of all the nodes is in passing state.

  ```
  admin@orchestrator[master-0]# show system diagnostics status
  ```

- Run the following command in CLI to view the docker engines status. Verify that all docker engines are in CONNECTED state.

  ```
  admin@orchestrator[master-0]# show docker engine
  ```

System Maintenance Procedures

Backup Procedures

Back up CLI Configuration

Back up the CLI configuration of APP VNF and DB VNF. Then copy the backups to an external server. The following sections describe the commands for APP VNF and DB VNF.

DRA VNF

The following commands saves each configuration as a separate file in the system.

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# config
admin@orchestrator# show running-config binding | save
/data/config/binding_cli_backup
admin@orchestrator# show running-config license | save
/data/config/license_cli_backup
admin@orchestrator# show running-config network virtual-service
| save /data/config/vip_cli_backup
admin@orchestrator# show running-config alert snmp-v2-destination
| save /data/config/alert_snmp-v2_cli_backup
admin@orchestrator# show running-config alert rule | save
/data/config/alert_rule_cli_backup
admin@orchestrator# show running-config external-aaa | save
/data/config/external-aaa_cli_backup
admin@orchestrator# show running-config ntp | save
/data/config/ntp_backup
admin@orchestrator# show running-config aaa authentication users
user | save /data/config/aaa_users_backup
admin@orchestrator# show running-config nacm groups group |
save /data/config/nacm_groups_backup
```

Copy the backup of the CLI configs to an external server.

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cd /data/orchestrator/config
cps@${DRM-hostname}:~$ scp -i /home/cps/cps.pem *_backup
<user>@<external-server>:<external-folder>
```

DB VNF

The following commands saves each configuration as a separate file in the system.

```
# node: DRA DB Master
# user: cps
cps@${DBM-hostname}:~$ cli
admin@orchestrator# config
admin@orchestrator# show running-config binding |
save /data/config/database_cli_backup
admin@orchestrator# show running-config license |
save /data/config/license_cli_backup
admin@orchestrator# show running-config network
virtual-service | save /data/config/vip_cli_backup
```

```
admin@orchestrator# show running-config alert snmp-v2-destination
 | save /data/config/alert_snmp-v2_cli_backup
admin@orchestrator# show running-config alert rule |
save /data/config/alert_rule_cli_backup
admin@orchestrator# show running-config external-aaa
| save /data/config/external-aaa_cli_backup
admin@orchestrator# show running-config ntp | save
/data/config/ntp_backup
```

Copy the backup of the CLI configs to an external server.

```
# node: DRA DB Master
# user: cps
cps@${DBM-hostname}:~$ cd /data/orchestrator/config
cps@${DBM-hostname}:~$ scp -i /home/cps/cps.pem *_backup
 <user>@<external-server>:<external-folder>
```

Back up Policy Builder

Export the CPS service configuration to a single file.

1. Open DRA Central GUI : https://<master ip>/central/dra
2. Click **Import/Export** under Policy Builder.
3. Select/enter the following details:
   - Export Type
   - Export URL
   - Export File Prefix
   - Use zip file extension
4. Click **Export**.
5. Save the ZIP file.

Back up CRD

Back up the CRD data to a single file.

For more information, see .

1. Open DRA Central GUI : https://<master ip>/central/dra
2. Click **Custom Reference Data** under Custom Reference Data.
3. Select/enter the following details under **Export**:
   - Use zip file extension
4. Click **Export**.
5. Save the ZIP file.

Shutting Down CPS

Shut down DRA VNF

1. Use the following command to shut down the application processes in DRA VNF:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# system stop
```

2. Run the following command to verify that the system status running is "false".

```
admin@orchestrator# show system status
```

3. Use the following command to verify that only the infrastructure items are running:

```
admin@orchestrator# show scheduling status
```

Shut down DB VNF

1. Use the following command to shut down the application processes in DRA DB VNF:

```
# node: DRA DB Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# system stop
```

2. Run the following command to verify that the system status running is "false".

```
admin@orchestrator# show system status
```

3. Use the following command to verify that only the infrastructure items are running:

```
admin@orchestrator# show scheduling status
```

Starting up CPS

Use the following commands to start up the system after a maintaince window is completed and the VMs are powered on.

Start up DRA VNF

Use the following command to start the application processes in DRA VNF:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# system start
```

Start DB VNF

Use the following command to start the application processes in DRA DB VNF:

```
# node: DRA DB Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# system start
```

Post Power up VM Health Check

Perform a health check on both VNFs after the maintenance window is complete and the VMs are powered on. For more information, see System Health Checks, on page 17.

Diameter Troubleshooting and Connections

For messages belonging to particular interface, CPS vDRA should be ready to make diameter connection on the configured application port. As CPS vDRA acts as a server, it should be listening on ports for different applications to accept any incoming diameter requests for the application.

If you are facing problems making diameter connections, check for the following configuration:

DRA Plug-in Configuration in DRA Policy Builder (PB)

Step 1 Login to director VM and check the following files to find the ports that re open.

- For IPv4 endpoints, /var/broadhop/iptables/dra.iptables
- For IPv6 endpoints, /var/broadhop/iptables/dra.iptables6

Examples:

```
cps@dra1-sys04-director-1:~$ cat /var/broadhop/iptables/dra.iptables
dra,a72f412ed2d9d48386b543123f817a6bea4cc12c21b4ffaf5575681c9be5309f,-d 172.18.63.234 -p tcp
-m tcp --dport 4567 -m addrtype --dst-type LOCAL -j DNAT --to-destination 172.17.0.14:13868
```

```
cps@dra1-sys04-director-1:~$ cat /var/broadhop/iptables/dra.iptables6
dra,b76ddc032f1012c486547d5c2666fa6a3ec0082d6a502ffb2ae0d8f995434883,-d
2606:ae00:3001:8311:172:16:241:109 -p tcp -m tcp --dport 3868 -m addrtype
--dst-type LOCAL -j DNAT --to-destination [fd00:dead:beef:0:0:242:ac11:e]:13869
```

This indicates that the stack is up and running at IP and port 172.17.0.14:13868.

Step 2 Login to the Diameter endpoint container and check the port where Diameter stack is running.

If the port 3868 is configured in Policy Builder, internally in container Diameter stack runs on port 13868 (appends 1 infront of port number, this is internal port mapping). Similary for 3869, it shows diameter stack is running on 13869.

Example:

```
root@diameter-endpoint-s106:/# netstat -na | grep 3868
tcp6       0      0 :::13868                :::*                    LISTEN
```

Step 3 Listen for Diameter traffic by logging into Director VMs diameter endpoint container and execute the following command:

```
tcpdump -i any port 13868 -s 0 -vv
```

Troubleshooting Basics

Troubleshooting CPS vDRA consists of these types of basic tasks:

- Gathering Information
- Collecting Logs
- Running Traces

Diameter Error Codes and Scenarios

Table 1: Diameter Error Codes and Scenarios

Result-Code	Result-Code Value	Description
Informational		
DIAMETER_MULTI_ ROUND_AUTH	1001	Subsequent messages triggered by client shall also used in Authentication and to get access of required resources. Generally used in Diameter NAS.
Success		
DIAMETER_SUCCESS	2001	Request processed Successfully.
DIAMETER_LIMITED_SUCCESS	2002	Request is processed but some more processing is required by Server to provide access to user.
Protocol Errors [E-bit set]		
DIAMETER_COMMAND _UNSUPPORTED	3001	Server returns it if Diameter Command-Code is un-recognized by server.
DIAMETER_UNABLE _TO_DELIVER	3002	Message cannot be delivered because there is no Host with Diameter URI present in Destination-Host AVP in associated Realm.
DIAMETER_REALM_NOT _SERVED	3003	Intended Realm is not recognized.
DIAMETER_TOO_BUSY	3004	Shall return by server only when server unable to provide requested service, where all the pre-requisites are also met. Client should also send the request to alternate peer.
DIAMETER_LOOP_DETECTED	3005	-
DIAMETER_REDIRECT _INDICATION	3006	In Response from Redirect Agent.
DIAMETER_APPLICATION _UNSUPPORTED	3007	-

Result-Code	Result-Code Value	Description
DIAMETER_INVALID_HDR_BITS	3008	It is sent when a request is received with invalid bits combination for considered command-code in DIAMETER Header structure. For example, Marking Proxy-Bit in CER message.
DIAMETER_INVALID_AVP_BITS	3009	It is sent when a request is received with invalid flag bits in an AVP.
DIAMETER_UNKNOWN_PEER	3010	A DIAMETER server can be configured whether it shall accept DIAMETER connection from all nodes or only from specific nodes. If it is configured to accept connection from specific nodes and receives CER from message from any node other than specified.
Transient Failures [Could not satisfy request at this moment]		
DIAMETER_AUTHENTICATION_REJECTED	4001	Returned by Server, most likely because of invalid password.
DIAMETER_OUT_OF_SPACE	4002	Returned by node, when it receives accounting information but unable to store it because of lack of memory.
ELECTION_LOST	4003	Peer determines that it has lost election by comparing Origin-Host value received in CER with its own DIAMETER IDENTITY and found that received DIAMETER IDENTITY is higher.
Permanent Failures [To inform peer, request is failed, should not be attempted again]		
DIAMETER_AVP_UNSUPPORTED	5001	AVP marked with Mandatory Bit, but peer does not support it.
DIAMETER_UNKNOWN_SESSION_ID	5002	-
DIAMETER_AUTHORIZATION_REJECTED	5003	User can not be authorized. For example, Comes in AIA on s6a interface.
DIAMETER_INVALID_AVP_VALUE	5004	-
DIAMETER_MISSING_AVP	5005	Mandatory AVP in request message is missing.

Result-Code	Result-Code Value	Description
DIAMETER_RESOURCES _EXCEEDED	5006	A request was received that cannot be authorized because the user has already expended allowed resources. An example of this error condition is a user that is restricted to one dial-up PPP port, attempts to establish a second PPP connection.
DIAMETER_CONTRADICTING _AVPS	5007	Server has identified that AVPs are present that are contradictory to each other.
DIAMETER_AVP_NOT_ALLOWED	5008	Message is received by node (Server) that contain AVP must not be present.
DIAMETER_AVP_OCCURS _TOO_MANY_TIMES	5009	If message contains the a AVP number of times that exceeds permitted occurrence of AVP in message definition.
DIAMETER_NO_COMMON _APPLICATION	5010	In response of CER if no common application supported between the peers.
DIAMETER_UNSUPPORTED _VERSION	5011	Self explanatory.
DIAMETER_UNABLE _TO_COMPLY	5012	Message rejected because of unspecified reasons.
DIAMETER_INVALID_BIT _IN_HEADER	5013	When an unrecognized bit in the Diameter header is set to one.
DIAMETER_INVALID _AVP_LENGTH	5014	Self explanatory.
DIAMETER_INVALID _MESSAGE_LENGTH	5015	Self explanatory.
DIAMETER_INVALID_AVP _BIT_COMBO	5016	For example, marking AVP to Mandatory while message definition doesn't say so.
DIAMETER_NO_COMMON _SECURITY	5017	In response of CER if no common security mechanism supported between the peers.

Policy DRA Error Codes

Non-compliant Diameter requests are checked for errors in routing AVP and P-bits. The following table describes the error codes and the reasons for errors in Diameter requests:

Table 2: Policy DRA Error Codes

Policy DRA Error String	Error Code	Sub-code	Description
No application route found	3002	001	Route List Availability Status is “Unavailable”
Timeout triggered	3002	002	Timeout triggered
No peer group	3002	003	No peer group
Session DB Error	3002	004	Session DB Error
Binding DB Error	3002	005	Binding DB Error
No key for binding lookup	3002	006	No key for binding lookup
Binding not found	3002	007	Binding not found
Message loop detected	3005	008	Message loop detected
Parsing exception with message	3009	009	Parsing exception with message
CRD DB Error	3002	010	CRD DB Error
Retries exceeded	3002	011	Retries exceeded
No peer route	3002	012	No peer routing rule found for a Realm-only or non-peer Destination-Host
P-bit not set	3002	013	P-bit in the Request message is set to “0”
Missing Origin-Host AVP	5005	014	Mandatory Origin-Host AVP missing
Missing Origin-Realm AVP	5005	015	Mandatory Origin-Realm AVP missing
Missing Destination-Realm AVP	5005	016	Mandatory Destination-Realm AVP missing
No avp found in request for SLF lookup type	3002	101	No avp found in request for SLF lookup type
SLF DB Error	3002	102	SLF DB Error
SLF credential not found in DB	3002	103	SLF credential not found in DB
SLF Destination type not found in DB	3002	104	SLF Destination type not found in DB

Policy DRA Error String	Error Code	Sub-code	Description
Destination not found in SLF Mapping Table	3002	105	Destination not found in SLF Mapping Table
Binding DB Overload	3002	022	Binding record limit exceeded

Default HTTP Error Codes

You can configure the HTTP response error code (such as 4xx, 5xx) corresponding to each vDRA Rest API JSON error response code for the GET binding (for example imsi, imsiApn, msisdn, msisdnApn, ipv4, ipv6). For more information about the CRD, see the *CPS vDRA Configuration Guide*.

If you do not configure the Rest API HTTP Error Code in the CRD, vDRA uses the default HTTP error codes for GET binding Rest API.

The following table lists the default HTTP error codes:

Table 3: Default HTTP Error Codes

vDRA Rest API Error Code	HTTP Error Code	HTTP Reason-Phrase
1001 (INTERNAL_ERROR)	500	Internal Server Error
2014 (DATA_NOT_FOUND)	404	Not Found
2019 (INVALID_API_FORMAT)	400	Bad Request

Debug ping / ping6

Run the following commands to check ping connectivity from the VM to other nodes using IPv4 and IPv6:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# debug ping <wtc2b1fdrd02v> -n <IPv4 address>
admin@orchestrator# debug ping6 <wtc2b1fdrd02v> -n <IPv6 address>
```

Where:

- -n:

Debug traceroute

Run the following commands to check traceroute connectivity from the VM to other nodes:

IPv4:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# debug traceroute <VMHOST> <IPv4address>
```

IPv6:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# debug traceroute <VMHOST> -6 <IPv6address>
```

Debug tcpdump

Use the following command to get packet capture from the VM. Specify interface and port details to avoid big packet capture files.

If you use the `-i any` option, you may see the same packet twice: once as it traverses the VMs interface, and again when it traverses the Docker container's virtual interface.

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# debug tcpdump wtc2b1fdrd01v test.pcap
60s -s 0 -i ens162 port 3868
admin@orchestrator# debug packet-capture gather directory test_debug
admin@orchestrator# debug packet-capture purge
```

You can download the packer capture file from : https://<master ip>/orchestrator/downloads/ after logging in to https://<master ip>/

After you download the file, delete the packet capture files to clean up the disk space.

Monitoring Application Logs

Use the following commands to monitor application logs :

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# monitor log application
```

Debug Tech to Capture Logs

Run the following command to capture SVN, CRD, logs, and save it at http://<master ip>/orchestrator/downloads/debug/tech/:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# debug tech
```

Monitoring Container Logs

Use the following command to monitor specific container logs:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# monitor log container <container-name>
```

Monitoring Orchestrator Logs

Use the following command to monitor orchestrator logs during an upgrade/downgrade:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# monitor log container orchestrator
| include AUDIT
```

If the CLI is not accessible or is giving errors when executing commands, use the following command from the master VM for more information:

```
cps@${DRM-hostname}:~$ docker logs orchestrator
```

Change CLI User Password

If you know the existing password, use the following steps to change the user password in CLI:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# aaa authentication users user fpasapi change-password
Value for 'old-password' (<string>): *****
Value for 'new-password' (<string>): ********
Value for 'confirm-password' (<string>): ********
```

If you do not know the password, use the following commands to reset the password:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# config
admin@orchestrator(config)# aaa authentication users user fpasapi gid 100
 uid 9000 homedir "" ssh_keydir "" password <password>
admin@orchestrator(config-user-apiuser)# commit
Commit complete.
admin@orchestrator(config-user-apiuser)# end
```

Restart Docker Container

If the commands `show docker service` or `system diagnostics` show errors, check the docker service for any unhealthy processes. If there are unhealthy processes, use the command `monitor container logs` to view logs and then restart the docker container.

```
Action
# node: DRA Master / DB Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show docker service | tab | exclude HEALTHY
admin@orchestrator# show system diagnostics | tab | exclude passing
# container-name is unhealthy process container id.
admin@orchestrator# docker restart container-id <container-name>
```

Check DNS Config

Check the VMs dnsmasq file to verify whether the DNS entries are present; if not, perform the following steps:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cat /data/dnsmasq/etc/dnsmasq.conf
# If DNS entries are missing, perform the following steps:
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show running-config network dns |
save /data/config/dns_cli_backup
admin@orchestrator# config
admin@orchestrator(config)# no network dns
admin@orchestrator(config)# commit
admin@orchestrator(config)# end
admin@orchestrator# config
admin@orchestrator(config)# load merge /data/config/dns_cli_backup
admin@orchestrator(config)# commit
admin@orchestrator(config)# end
admin@orchestrator# exit
cps@${DRM-hostname}:~$ cat /data/dnsmasq/etc/dnsmasq.conf
```

Redeploy Master VM

When the master VM is deleted or redeployed for some reason, you must make it part of the existing cluster. workaround to make it part of the cluster as described in the following steps:

```
# node: DRA Master
# user: cps
# After the master VM is redeployed, log into the master VM, and wait til
# cpsinstall is complete
cps@${DRM-hostname}:~$ journalctl -u cpsinstall.service –f
# Verify that the following log apperas: log <date time stamp> master-0
bootstrap.sh[1521]: Install script completed.
# Once cpsinstall is finished; execute the following
# commands on the master VM in the order specified.
cps@${DRM-hostname}:~$ docker stop $(docker ps -a -q)
cps@${DRM-hostname}:~$ docker rm $(docker ps -a -q)
cps@${DRM-hostname}:~$ weave launch-router --ipalloc-init consensus=3
cps@${DRM-hostname}:~$ sudo rm -rf /data/orchestrator
cps@${DRM-hostname}:~$ sudo rm /var/cps/bootstrap-status
cps@${DRM-hostname}:~$ sudo /root/bootstrap.sh
cps@${DRM-hostname}:~$ ssh-keygen -f "/home/cps/.ssh/known_hosts" -R
[localhost]:2024
```

Remove MongoDB Replica Set Member

Perform the following steps to remove a replica set member from MongoDB.

Caution The command `no database cluster` deletes the configuration completely, so ensure the information is correct.

```
# node: DRA Master
# user: cps
cps@${DBM-hostname}:~$ cli
cps@${DBM-hostname}:~$ config
admin@orchestrator(config)# no database cluster binding shard
binding-shard-1 shard-server fn6-1a1bs1k
admin@orchestrator(config)# commit
admin@orchestrator(config)# end
#connect to the replica set primary member container to remove the node, take a note
```

```
#of port of the replica set
cps@${DBM-hostname}:~$ docker connect mongo-s104
root@mongo-s104:/# mongo --port 27033
rs-binding-shard-1:PRIMARY> rs.status()
#Note the name of the member from rs status output and then input it to
#rs.remove to remove the member
rs-binding-shard-1:PRIMARY> rs.remove
("[2606:ae00:2001:2420:8000::9]:27034")
```

Monitoring MongoDB Logs

The MongoDB logs are stored under `/data/mongod-node/db` on every VM that has mongod instance running.

Clean the Database

Perform the following steps if you want to clean the database and recreate a fresh database.

Warning All the data will be lost.

```
# node: DRA Master
# user: cps
cps@${DBM-hostname}:~$ cli
# Stop all the application process:
cps@${DBM-hostname}:~$ system stop
# Wait for some time till all the application proceses stop.
# You can check the process using the commands:
# show scheduling status and show system status
# Repeat the following steps in all the database nodes
cps@${DBM-hostname}:~$ rm -rf /data/configdb/*
cps@${DBM-hostname}:~$ rm -rf /data/db/*
cps@${DBM-hostname}:~$ rm -rf /mmapv1-tmpfs-<port>/*
cps@${DBM-hostname}:~$ cli
# Restart the system:
cps@${DBM-hostname}:~$ system start
```

Reset the CLI Configuration

Perform the following steps to reset the CLI configuration:

Caution The complete configuration will be reset.

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ docker exec -it orchestrator bash
cps@${DRM-hostname}:~$ /var/confd/bin/confd_load -D -m -l
/data/cdb//*.xml
```

Policy DRA Logger Levels

Policy DRA Application logs are available for debugging purposes.

Note that turning on logs in a production system can have a substantial impact on the system performance and is not recommended.

Enabling and Disabling Logs

Use the orchestrator CLI to enable and disable application logs.

```
admin@orchestrator# logger set ?
Possible completions:
  <logger name>

admin@orchestrator# logger set com.broadhop.dra.service ?
Possible completions:
  debug  error  info  off  trace  warn

admin@orchestrator# logger clear com.broadhop.dra.service ?
Possible completions:
  |  <cr>
```

View Log Levels

The different log levels in the order of increasing details in the log are:

- Error (error logs)
- Warn (error and warning logs)
- Info
- Debug
- Trace (all logs)

The default log level is warn.

Use the following orchestrator CLI command to view the current log levels set on a per application module basis.

```
admin@orchestrator# show logger level
Logger                          Current Level
-------------------------------------------------
com.broadhop.dra.service               warn
dra.trace                   warn
org.jdiameter                    warn
```

View Logs

To view application logs continuously similar to the `tail -f` command, use the following command:

```
"monitor log application"
```

To view application logs that were previously collected in a consolidated log file (similar to the `more` command), use the following command:

```
show log application
```

Common Loggers

The following table describes the different loggers and their default log level:

Table 4: Common Loggers

Logger Name	Description	Default Log Level
com.broadhop.dra.service	Policy DRA application logs. This displays logs from various modules of the Policy DRA system.	warn
dra.trace	Policy DRA audit logs. This displays a summary of the Diameter message request and response processing.	warn
org.jdiameter	jDiameter module logs. This displays logs from various modules of the jDiameter module.	warn
com.broadhop.dra.session. expiration.service	Checks and deletes stale sessions.	warn
com.broadhop.dra.service .stack	vDRA stack-related logs to enable debugging at stack level. To be used with org.jdiameter log level.	warn
com.broadhop.dra.service. mongo.sharding.impl	This logger provides logs about the binding and API handling operations managed by the Worker.	warn
com.mongodb	Logging related to MongoDB library as the Worker invokes MongoDB API for database operations.	warn
com.broadhop.dra.service .routing	vDRA routing-related messages to debug issues in routing.	warn
com.broadhop.dra.service .control	vDRA logs related to control messaging.	warn

Common Troubleshooting Steps

CPS vDRA Logs

Step 1 Use the following command in CLI to view the consolidated application logs.

```
admin@orchestrator[master-0]# show log application
```

Step 2 Use the following command in CLI to view the consolidated engine logs.

```
admin@orchestrator[master-0]# show log engine
```

Counters and Statistics

Check for statistics generated at pcrfclient01/02 in `/var/broadhop/stats` and counters in beans at jmx terminal.

System Health Checks

View System Status

Use the following command to view the system status and verify whether the system is running, or if any upgrade or downgrade is in progress, and whether it is 100% deployed.

APP VNF

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show system status
```

DB VNF

```
# node: DRA DB Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show system status
```

If system is not 100% deployed, use the following command to view the current scheduling status: `system scheduling status`

View System Diagnostics

Use the following command to view the system disgnostics and debug failed processes.

APP VNF

```
# node: DRA Master
# user: cps
```

```
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show system diagnostics | tab | exclude passing
```

DB VNF

```
# node: DRA DB Master
# user: cps
cps@ ${DBM-hostname}:~$ cli
admin@orchestrator# show system software | tab
admin@orchestrator# show system diagnostics | tab | exclude passing
```

You can monitor the log of the container using the command: `monitor container logs`

Check System Scheduling Status

Use the following command to verify the installer scheduler status. The scheduler must reach `haproxy-int-api 1 500` and all states indicate running.

APP VNF

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show scheduling status
```

DB VNF

```
# node: DRA DB Master
# user: cps
cps@${DBM-hostname}:~$ cli
admin@orchestrator# show scheduling status
```

Check Docker Engine

Use the following commands to check the docker engine:

- `show docker engine | tab`: Check docker engine connected status to verify whether all VM engines are connected.
- `show running-config docker | tab`: Check the running configuration of the docker to verify whether all VMs are registered to the Master VM correctly and whether all VMs are shown with internal IP and scheduling slots.

APP VNF

Command:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show docker engine | tab
```

Command:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show running-config docker | tab
```

DB VNF

Command:

```
# node: DRA DB Master
# user: cps
cps@${DBM-hostname}:~$ cli
admin@orchestrator# show docker engine | tab
```

Command:

```
# node: DRA DB Master
# user: cps
cps@${DBM-hostname}:~$ cli
admin@orchestrator# show running-config docker | tab
```

Check Docker Service

Use the following commands to check the docker service:

- `show docker service | tab`: to verify whether all the docker services are running.
- `show docker service | tab | exclude HEALTHY` : to view unhealthy docker services.

APP VNF

Command:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show docker service | tabb
```

Command:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show docker service | tab | exclude HEALTHY
```

DB VNF

Command:

```
# node: DRA DB Master
# user: cps
cps@${DBM-hostname}:~$ cli
admin@orchestrator# show docker service | tab
```

Command:

```
# node: DRA DB Master
# user: cps
cps@${DBM-hostname}:~$ cli
admin@orchestrator# show docker service | tab | exclude HEALTHY
```

View Alert Status

Check the alert status in both VNFs and verify that there are no issues.

APP VNF

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show alert status | tab | include firing
```

DB VNF

```
# node: DB Master
# user: cps
cps@${DBM-hostname}:~$ cli
admin@orchestrator# show alert status | tab | include firing
```

Troubleshooting Application

Call Failures

In case of call failures, check the Peer Connection, Binding Monitoring, Peer Errors, Error Result Code in Central GUI as described:

1. Log into the Central GUI as admin.

   In the Peer Monitoring, filter by the host where call failures are observed.

   If there is any problem with connection; that peer is not listed in Active Peer Endpoints screen and is listed in Inactive peers.

   Figure 1: Peer Monitoring - Active Peer Endpoints

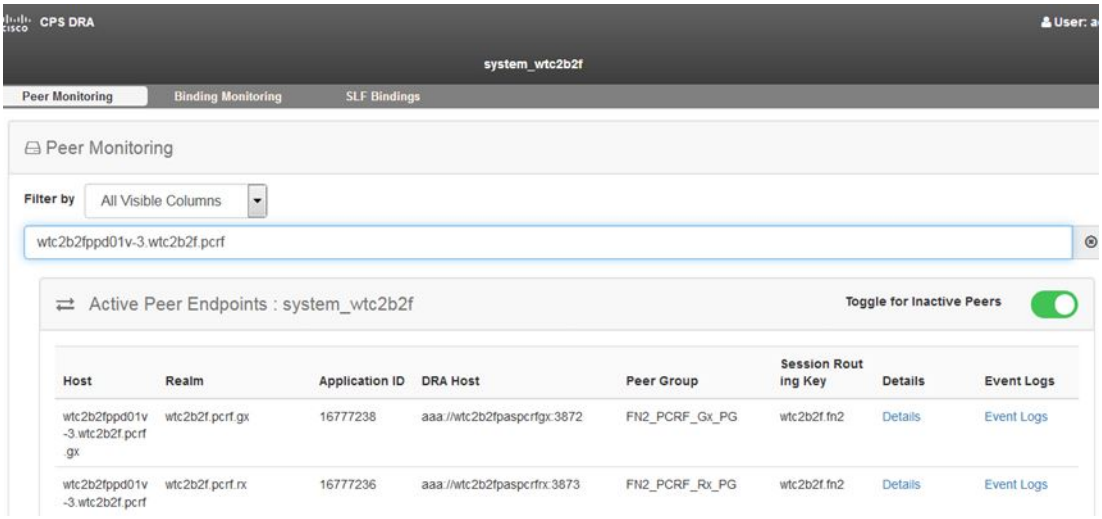
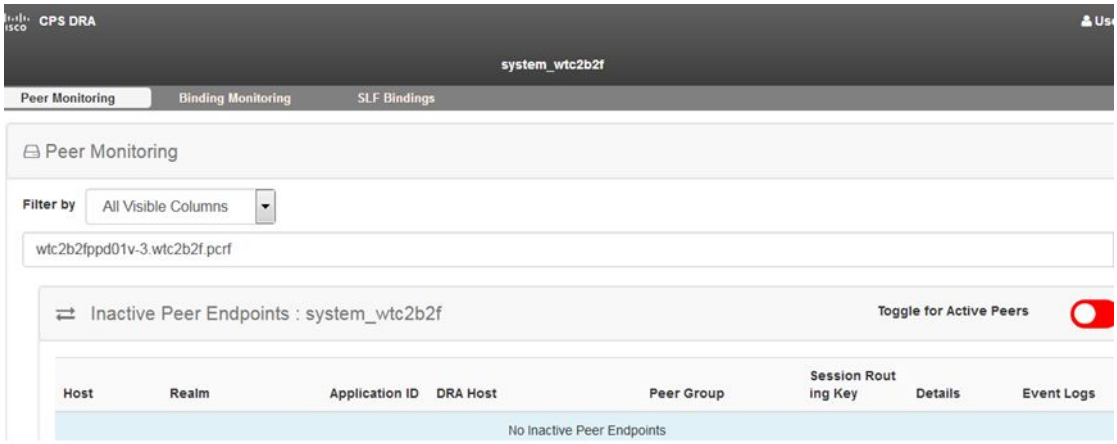
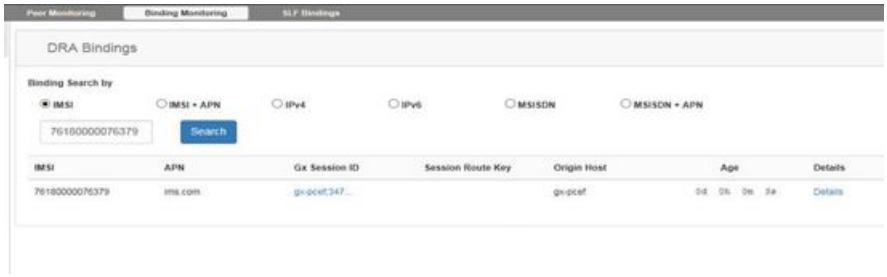


Figure 2: Peer Monitoring - Inactive Peer Endpoints



2. Check if the bindings are getting created. Filter the results for the imsiApn/ msisdnApn / ipv4/ ipv6 binding for which binding has to be retrieved.

Figure 3: DRA Bindings



3. Log into Central GUI/Grafana as admin and go to the **Home** > **Application Summary**.

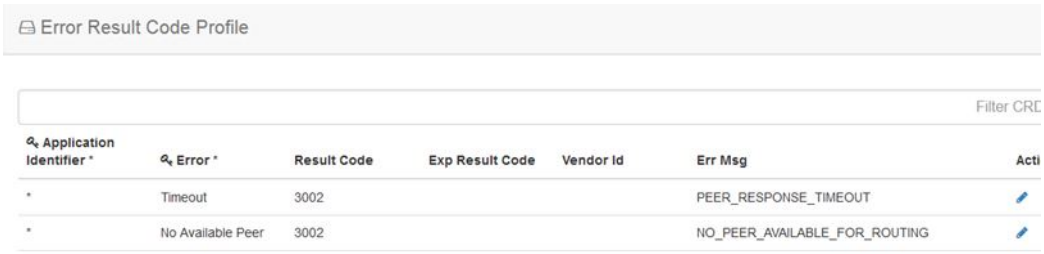
   Check for specific errors in Grafana. The errors indicate the exact result code received from peer.

Figure 4: Application Summary

4. Log into Central GUI/Customer Reference Data as admin.

   Check for the descriptions of specific errors from customer reference data so that necessary action can be taken.

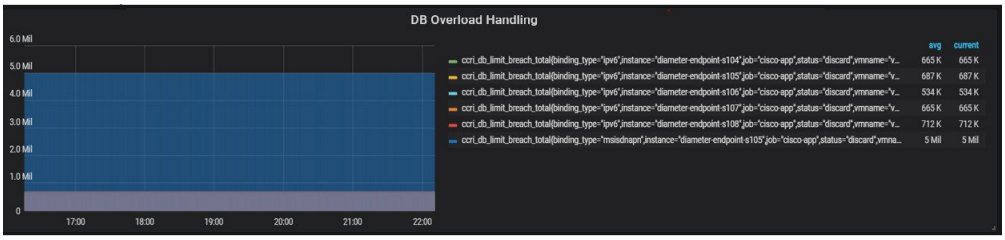
Figure 5: Error Result Code Profile



5. Log into Central GUI/Grafana as admin and go to the **Home** > **Application Summary**.

   Check for “discard” status in Grafana in DB Overload Handling graph. If entries are found in the graph, then check if maximum record limit has been set on database.

Figure 6: DB OVerload Handling



Relay Failure Between Two vDRA Instances

Use the following command to check traceroute connectivity from the VM to other nodes:

```
# node: DRA Director VM
# user: cps
cps@${drd-hostname}:~$ ping6 <Relay hostname configured in Policy Builder>
```

If there is any issue with the other vDRA, ping6 results in “timeouts.”

Monitoring Exceptions

Use the following command to monitor exceptions in Redis or database:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# monitor log application | include Exception
```

Monitoring Performance

To check if there are any performance issues with vDRA, log into Central GUI as admin and check the System Health.

Monitor for any timeouts, spikes or decrease in TPS for database response times, peer response timeouts, average response timeouts.

Figure 7: System Health

Figure 8: Timeouts

Figure 9: Database Queries

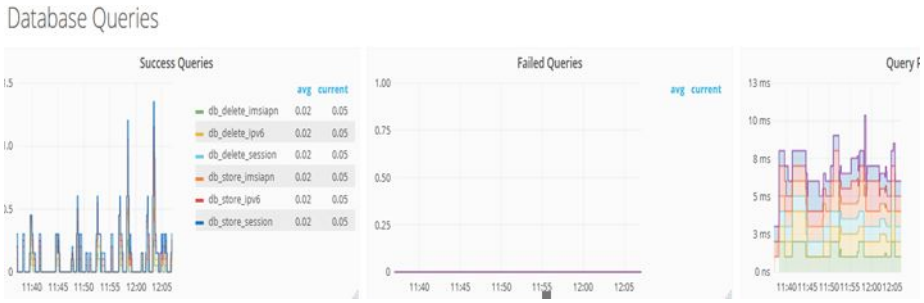
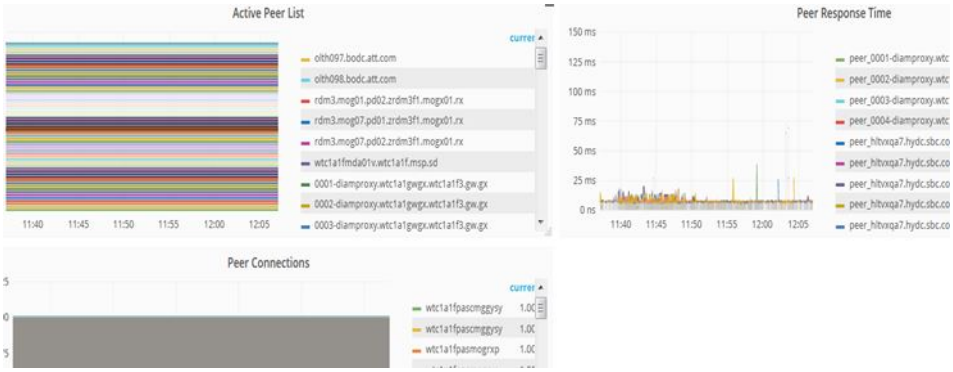
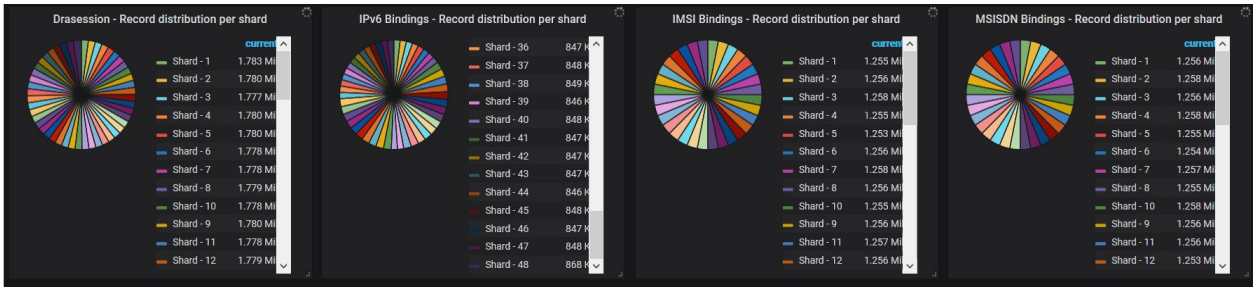


Figure 10: Peer Details



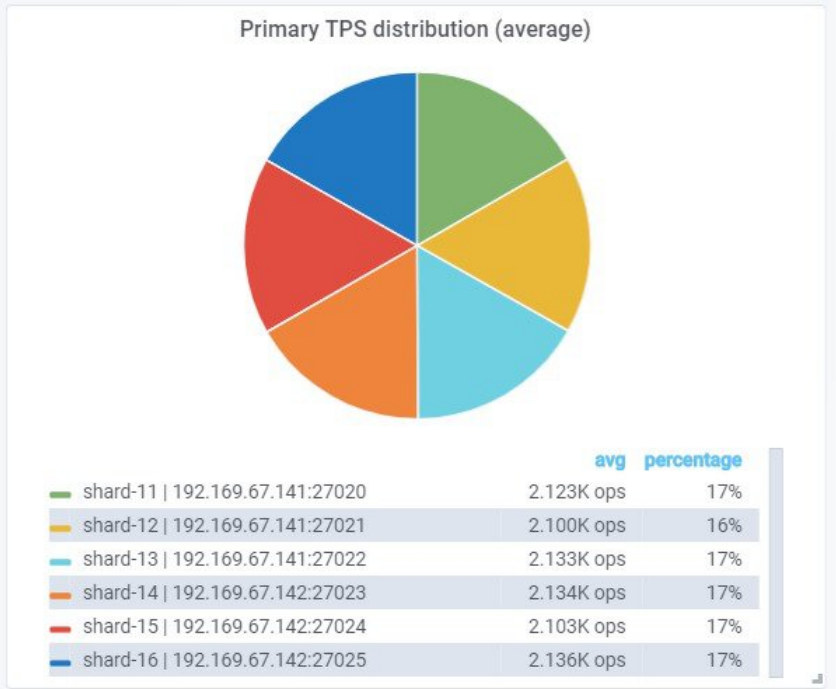
Monitor session and binding records are uniformly distributed across all the shards from ‘Application Summary’ dashboard of DRA VNF.

Figure 11: Record Distribution per Shard



Monitor Primary TPS is uniformly distributed across all the shards from 'Database Monitoring' dashboard of Binding VNF.

Figure 12: Primary TPS Distribution



Check Alerts

Use the following command to check for alerts and any issues with peer connections, low memory, low disk, or link failures.

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show alert status | tab | include firing
```

Frequently Encountered Troubles in CPS vDRA

Redis Not Working

Step 1 Check which containers are available using the following commands:

```
admin@orchestrator[dra1-sys04-master-0]# show docker service | include control-plane | tab | exclude
 monitor
control-plane        101    control-plane        3.2.6.0   dra1-sys04-master-0    control-plane-s101
        HEALTHY  false -
control-plane        102    control-plane        3.2.6.0   dra1-sys04-control-0   control-plane-s102
        HEALTHY  false -
control-plane        103    control-plane        3.2.6.0   dra1-sys04-control-1   control-plane-s103
        HEALTHY  false -
diameter-endpoint    104    global-control-plane 3.2.6.0   dra1-sys04-director-1
global-control-plane-s104 HEALTHY  false -
```

```
diameter-endpoint    105    global-control-plane 3.2.6.0   dra1-sys04-director-2
global-control-plane-s105 HEALTHY  false -

amin@orchestrator[dra1-sys04-master-0]# show docker service | include redis | tab | exclude monitor
diameter-endpoint    104   diameter-redis-q-a    3.2.6.0   dra1-sys04-director-1
diameter-redis-q-a-s104   HEALTHY  false -
diameter-endpoint    105   diameter-redis-q-a    3.2.6.0   dra1-sys04-director-2
diameter-redis-q-a-s105   HEALTHY  false -
```

Step 2 Login in into each of the above containers.

The following example shows that the redis server is working.

```
admin@orchestrator[dra1-sys04-master-0]# docker connect control-plane-s101
/data # ps -ef
PID   USER     TIME   COMMAND
    1 redis    332:42 redis-server
```

Step 3 Check the following entries in `/etc/broadhop/draTopology.ini` file at DRA directors diameter-endpoint container.

```
root@diameter-endpoint-s104:/# cd /etc/broadhop
root@diameter-endpoint-s104:/etc/broadhop# cat draTopology.ini

dra.local-control-plane.redis.0=control-plane-s101:6379
dra.local-control-plane.redis.1=control-plane-s102:6379
dra.local-control-plane.redis.2=control-plane-s103:6379

dra.global-control-plane.redis.0=192.169.67.178:6379

root@diameter-endpoint-s104:/etc/broadhop# cat redisTopology.ini
#Generate file from consul configuration

dra.redis.qserver.0=diameter-redis-q-a-s104:6379
dra.redis.qserver.1=diameter-redis-q-a-s105:6379
```

Step 4 Verify that the global control plane is configured correctly from the CLI. For more on commands, see the *CPS vDRA Operations Guide*.

Gx Bindings not happening on Mongo

Step 1 Check if the binding's exceptions are coming in `consolidated-qns.log` file.

Step 2 Check for the entries in `/etc/broadhop/draTopology.ini` file.

```
dra.redis.qserver.1=lb02:6379
dra.redis.qserver.2=lb02:6380
dra.redis.qserver.3=lb02:6381
dra.redis.qserver.4=lb02:6382
dra.redis.qserver.4=lb02:6383
dra.local-control-plane.redis.1=lb02:6379
db.shards.metadata.ipv6.uri=mongodb://[2606:ae00:3001:8311:172:16:244:3]:27019,[2606:ae00:3001:8311:172:16:244:2a]:27019
db.shards.metadata.ipv4.uri=mongodb://[2606:ae00:3001:8311:172:16:244:4]:27019,[2606:ae00:3001:8311:172:16:244:2b]:27019
db.shards.metadata.imsiapn.uri=mongodb://[2606:ae00:3001:8311:172:16:244:4]:27019,[2606:ae00:3001:8311:172:16:244:2b]:27019
db.shards.metadata.msisdnapn.uri=mongodb://[2606:ae00:3001:8311:172:16:244:4]:27019,[2606:ae00:3001:8311:172:16:244:2b]:27019
db.shards.metadata.session.uri=mongodb://[2606:ae00:3001:8311:172:16:244:3]:27019,[2606:ae00:3001:8311:172:16:244:2a]:27019
```

For example, make sure if the primary binding server is 27019 only as per above example.

Step 3 Check for the Binding Keys entries in binding key type profile and the application attached to the profile.

Rx Call Failing at CPS vDRA

Step 1 Check for the Binding key Retriever for Rx Profile.

Step 2 Check if the Gx Binding is available for that Binding key.

Step 3 Check the `consolidated-qns.log` file if CPS vDRA is able to retrieve SRK from the bindings.

Step 4 Check for any exception in `consolidated-qns.log` file during binding retrieval.

Step 5 If Rx peer is available for the same SRK at CPS vDRA, CPS vDRA should forward the Rx message to that peer.

Step 6 Check the connection for that peer and proper entries in Peer Group, Peer Routing, Peer Group Peer and Rx_Routing for Rx New session rules.

Call Failing at CPS vDRA due to Binding

Step 1 Check the `consolidated-qns.log` file to see if there are any warn logs on MongoShardPinger class related to unreachable mongo.

Step 2 If MongoShardPinger logs are present with text containing unreachable mongos it indicates the shard is not reachable.

Step 3 Check the connection for that shard.

CPS vDRA Forwarding Message to Wrong Peer

Step 1 Check the Control Center configuration in Gx_Routing for new session rules. Gx routing should have the AVP defined on the basis of which, one wants to route the traffic.

Step 2 Check whether the Control Center configuration for the Peer is bonded to correct Peer Group.

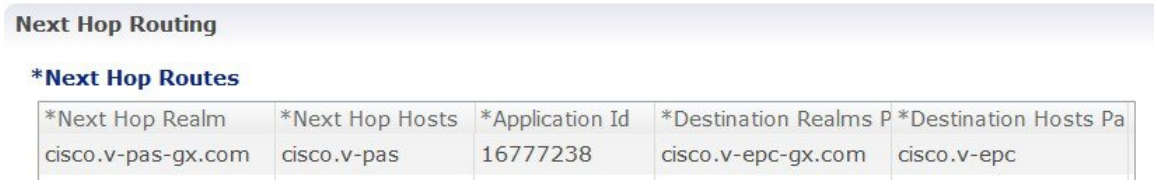
Step 3 Check whether the Peer Group is assigned to correct Peer Route and Dynamic AVPs are properly aligned with Peer Route in Gx New Session Rules.

Step 4 Diameter Connection with the desired Destination Peer should be established with CPS vDRA.

PCRF Generated Messages not Reaching CPS vDRA

Step 1 Make sure PCRF has the correct entry of CPS vDRA as next hop.

Figure 13: Next Hop Routes



Next Hop definition is mandatory in PCRF to forward the messages to CPS vDRA generated by PCRF itself.

For example, Gx-RAR, Sd-TSR

Step 2 Wild Card Entry not supported in Next Hop Routing configuration.

Issues in Reaching Ports and Setup IPs

Step 1 Check firewall is running or not.

Step 2 Make sure the firewall configuration is OK.

a) To check if this is the problem, then stop the firewall.

```
/etc/init.d/iptables stop
```

PB and CRD Inaccessible

Policy Builder and CRD are inaccessible when there are multiple route entries on the master node.

This issue occurs only on OpenStack setups.

OpenStack Neutron configures multiple default routes, if the gateway is also present in the interfaces static configuration.

For example, when configuring multiple interfaces on any VM, set "gateway" for only one interface, preferably public interface.

```
# public network
auto ens160
iface ens160 inet static
address x.x.x.60
netmask 255.255.255.0
gateway x.x.x.1

# private network
auto ens192
iface ens192 inet static
address y.y.y.155
netmask 255.255.255.0
```

Workaround

Run the following command to delete the default route to the internal network.

```
sudo route del default gw <internal network gateway IP>
```

For example: `sudo route del default gw y.y.y.1`

If the default route is not present for public network, run the following command:

```
ip route add default via <public network gateway IP>
```

For example: `ip route add default via x.x.x.1`

Central GUI Returns 503 Service Unavailable Error

After rebooting the master and control VMs, if the Central GUI returns 503 service unavailable error, perform the following steps:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# docker restart container-id haproxy-common-s101
```

Clear the browser cache and check the UI again.

Mog API Call Failure

If the MOG API calls fails internittently with an unauthroized message in a DRA director, then run the following commands to restart the container:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show network ips | include mogAPI
admin@orchestrator# show docker service | tab | include drd02v | include haproxy-common-s
admin@orchestrator# docker restart container-id haproxy-common-s10x
```

DRA Configuration API Call Returns NULL

If the DRA configuration API call returns null, restart the Policy Builder container as shown:

```
# node: DRA Master
# user: cps
cps@${DRM-hostname}:~$ cli
admin@orchestrator# show docker service | tab |
include drc01v | include policy-builder-s
admin@orchestrator# docker restart container-id
policy-builder-s10x
```

Diameter Errors and Call Model Issues when Running Heap Dump

Case: Messages timed out when running Heap Dump of DRA process.

Condition: Taking heap dump of director and worker process. Heap dumps taken results in full GC. This in turn causes major application pause and results in message time out.

Solution: It is recommended to take the heap dump only during Maintenance Window (MW)

Recovery Steps when Master VM is Down

Case: When Master VM is powered ON after 12 hrs it is stuck on orchestrator container throwing following logs:

```
2019-07-01T03:40:12.858+0000 I NETWORK [conn78] end connection 127.0.0.1:33586 (4 connections
 now open)
Waiting for 5s as mongod database is not yet up. Mon Jul 1 03:40:12 UTC 2019
2019-07-01T03:40:13.469+0000 I REPL [replexec-2] Canceling priority takeover callback
2019-07-01T03:40:13.469+0000 I REPL [replexec-2] Not starting an election for a priority
takeover, since we are not electable due to: Not standing for election because member is
not currently a secondary; member is not caught up enough to the most up-to-date member to
 call for priority takeover - must be within 2 seconds (mask 0x408)
2019-07-01T03:40:15.246+0000 I REPL [replexec-2] Scheduling priority takeover at
2019-07-01T03:40:26.616+0000
2019-07-01T03:40:17.919+0000 I NETWORK [listener] connection accepted from 127.0.0.1:33596
 #79 (5 connections now open)
```

Solution: To recover the master VM, you need to execute the following commands on Master VM:

```
docker exec -it orchestrator bash
supervisorctl stop mongo
rm -rf /data/db/*
supervisorctl start mongo
```

Call Failure Observed when Database VNF VMs are Recovering

Issue: Calls failure observed when database VNF VMs are in recovery mode.

Expected Behavior: Few call failures are expected when a shard-member recovers after restart and gets elected as new primary. The following is the expected behavior when a Binding VNF recovers after failover:

- All the shards members of the database VNF do not come up at the same time. They resynchronize with the existing shard-members and transition from STARTUP2 to Secondary to Primary state is not same for all the shards.
- Two elections for each shard are possible based on the database VM recovery time. The following is the sequence:
  1. **First Election:** Database VM having shard member with second highest priority completes the resynchronization first and becomes primary.
  2. **Second Re-election:** The shard member with highest priority completes the resynchronization and becomes primary (This behavior is as per the MongoDB replica-set protocol version 1 - pv1).

Timeout Observed after Policy Builder Publish on DRA

Issue: Timeout is observed when publishing is done during load.

Solution: Policy Builder publishing during load have impact on running calls.

Note It is recommended to perform Policy Builder publishing during Maintenance Window (MW).

No User Authentication Observed after MongoDB Auth Set on DRA VNF

Issue: No users are authenticated after MongoDB auth set on DRA VNF.

Solution: After new password is set on binding VNF and DRA VNF if `there are no users authenticated` exception is observed, restart the binding container.

Container not Recovering during Automation Run

Issue: `show system diagnostics` displays errors though the container is in HEALTHY state.

Note This issue can be observed on both DRA or Binding VNFs in any of the following scenarios:

- After VM restart
- ISO upgrade
- ISO downgrade
- Restarting of containers multiple times in short duration (for any testing)

Workaround: The following steps can be executed to confirm and resolve the issue:

1. Execute `show system diagnostics | tab | exclude passing` command to check the diagnostics status.

   Here is a sample output that displays errors in the container.

   ```
   admin@orchestrator[site2-master-0]# show system diagnostics | tab | exclude passing

   NODE              CHECK ID    IDX   STATUS       MESSAGE
   ------------------------------------------------------------------------------
   stats-relay-s102  serfHealth   1   critical  Agent not live or unreachable
   ```

2. Execute `show docker service | tab | include <containerName>` command to verify the health of the container.

   Here is a sample:

   ```
   admin@orchestrator[an-master]# show docker service | tab | include stats-relay-s102
   stats  102  stats-relay  19.4.0-xxx  an-control-0  stats-relay-s102  HEALTHY  false -
   ```

3. If Step 2 displays the state as **HEALTHY**, use the following workaround. Otherwise, diagnostics error is valid and check the container logs to find the root cause.

   a. From CLI, execute `docker connect <container-name>` to connect to a docker service and launch a bash shell running on the system.

      For example, `docker connect consul-1`

      1. Execute `consul members | grep -v alive` command.

         Here is the sample output.

         ```
         root@consul-1:/# consul members | grep -v alive
         Node                          Address        Status  Type    Build  Protocol
         DC   Segment
         ```

```
stats-relay-s102.weave.local 10.45.0.32:8301  failed  client  1.0.0   2       dc1
  <default>
```

2. Execute `consul force-leave stats-relay-s102.weave.local` command for all the containers which are in failed state.

b. Execute `docker restart container-id stats-relay-s102` to restart the container.

4. Execute `show system diagnostics | tab | exclude passing` command to verify that the issue has been fixed.

vDRA Database Troubleshooting

This section provides the information about vDRA database troubleshooting in Binding VNFs:

Note All commands under this section needs to be executed from Binding VNF CLI.

Database Operational Status

The following command provides database operational status of all database clusters configured. Execute the command in operational mode.

```
show database status | tab
```

Example:

```
admin@orchestrator[an-dbmaster]# show database status | tab
                                                          CLUSTER
ADDRESS         PORT   NAME         STATUS     TYPE         NAME     SHARD     REPLICA SET
-------------------------------------------------------------------------------------------
192.168.11.42  27026  arbiter-21  ARBITER    replica_set  session  shard-21  rs-shard-21
192.168.11.43  27026  server-x    PRIMARY    replica_set  session  shard-21  rs-shard-21
192.168.11.44  27026  server-y    SECONDARY  replica_set  session  shard-21  rs-shard-21
192.168.11.42  27027  arbiter-22  ARBITER    replica_set  session  shard-22  rs-shard-22
192.168.11.43  27027  server-x    SECONDARY  replica_set  session  shard-22  rs-shard-22
192.168.11.44  27027  server-y    PRIMARY    replica_set  session  shard-22  rs-shard-22
192.168.11.43  27019  session     PRIMARY    shard_db     session  shdb-4    session-sharddb
192.168.11.44  27019  session     SECONDARY  shard_db     session  shdb-5    session-sharddb

admin@orchestrator[an-dbmaster]#
```

Validate Sharding Database Metadata

1. Execute the following command to find sharding database PRIMARY of particular cluster.

   ```
   show database status cluster-name session | tab | include PRIMARY | include shard_db
   ```

2. Connecting to sharding database primary member.

   Non-Mongo Auth:

   ```
   mongo --ipv6 mongodb://[2606:ae00:2001:230b::2b]:27019
   ```

Mongo Auth:

```
mongo --ipv6 mongodb://adminuser:<password>@[2606:ae00:2001:230b::2b]:27019/admin
```

3. After successfully connecting to sharding database primary member, execute the following step:

   For example, to validate DRA sessions sharding database metadata information:

```
session-sharddb:PRIMARY> use drasessionsShardDB
switched to db drasessionsShardDB
session-sharddb:PRIMARY> db.shards.count()
2
session-sharddb:PRIMARY> db.buckets.count()
8192
session-sharddb:PRIMARY>
```

Validate Zone Aware Sharding Database Metadata

1. Execute the following command to find sharding database PRIMARY of particular cluster:

```
show database status cluster-name session | tab | include PRIMARY | include shard_db
```

2. Connecting to sharding database primary member.

   Non-Mongo Auth:

```
mongo --ipv6 mongodb://[2606:ae00:2001:230b::2b]:27019
```

   Mongo Auth:

```
mongo --ipv6 mongodb://adminuser:<password>@[2606:ae00:2001:230b::2b]:27019/admin
```

3. Validate configured shard and zone mappings.

   Example:

```
use ipv6ShardDB
switched to db ipv6ShardDB
binding-sharddb:PRIMARY>
binding-sharddb:PRIMARY> db.shards.find()
```

```
binding-sharddb:PRIMARY> db.shards.find()
{ "_id" : 1, "name" : "shard-1", "hosts" : "182.22.31.13:27017,182.22.31.14:27017", "zone" : "mumbai" }
{ "_id" : 2, "name" : "shard-2", "hosts" : "182.22.31.13:27018,182.22.31.14:27018", "zone" : "pune" }
{ "_id" : 3, "name" : "shard-3", "hosts" : "182.22.31.13:27020,182.22.31.14:27020", "zone" : "hyd" }
{ "_id" : 4, "name" : "shard-4", "hosts" : "182.22.31.13:27021,182.22.31.14:27021", "zone" : "bglr" }
{ "_id" : 5, "name" : "shard-5", "hosts" : "182.22.31.13:27022,182.22.31.14:27022", "zone" : "chennai" }
{ "_id" : 6, "name" : "shard-6", "hosts" : "182.22.31.13:27023,182.22.31.14:27023", "zone" : "hyd" }
{ "_id" : 7, "name" : "shard-7", "hosts" : "182.22.31.13:27024,182.22.31.14:27024", "zone" : "bglr" }
{ "_id" : 8, "name" : "shard-8", "hosts" : "182.22.31.13:27025,182.22.31.14:27025", "zone" : "pune" }
binding-sharddb:PRIMARY>
```

4. Validate configured zones and ranges.

   Example:

```
binding-sharddb:PRIMARY> db.zoneinfo.find()
{ "_id" : 1, "name" : "r1", "start" : "2017:6000:0000:0001", "end" : "2017:6000:0000:0500", "zone" : "bglr" }
{ "_id" : 2, "name" : "r2", "start" : "2018:6000:0000:0001", "end" : "2018:6000:0000:0500", "zone" : "bglr" }
{ "_id" : 3, "name" : "r1", "start" : "2013:6000:0000:0001", "end" : "2013:6000:0000:0500", "zone" : "chennai" }
{ "_id" : 4, "name" : "r2", "start" : "2014:6000:0000:0001", "end" : "2014:6000:0000:0500", "zone" : "chennai" }
{ "_id" : 5, "name" : "r1", "start" : "2015:6000:0000:0001", "end" : "2015:6000:0000:0500", "zone" : "hyd" }
{ "_id" : 6, "name" : "r2", "start" : "2016:6000:0000:0001", "end" : "2016:6000:0000:0500", "zone" : "hyd" }
{ "_id" : 7, "name" : "r1", "start" : "2008:5000:0000:0100", "end" : "2008:5000:0000:0500", "zone" : "mumbai" }
{ "_id" : 8, "name" : "r2", "start" : "2009:5000:0000:0100", "end" : "2009:5000:0000:0500", "zone" : "mumbai" }
{ "_id" : 9, "name" : "r1", "start" : "2011:6000:0000:0001", "end" : "2011:6000:0000:0500", "zone" : "pune" }
{ "_id" : 10, "name" : "r2", "start" : "2012:6000:0000:0001", "end" : "2012:6000:0000:0500", "zone" : "pune" }
```

5. Validate buckets/shard/range mapping.

   Example:

```
binding-sharddb:PRIMARY> db.buckets.find()
{ "_id" : ObjectId("5cb9845c63ebec62ea803d42"), "bucket-id" : 1, "shard" : 4, "migration" : false, "zone" : "bglr" }
{ "_id" : ObjectId("5cb9845c63ebec62ea803d43"), "bucket-id" : 2, "shard" : 4, "migration" : false, "zone" : "bglr" }
{ "_id" : ObjectId("5cb9845c63ebec62ea803d44"), "bucket-id" : 3, "shard" : 4, "migration" : false, "zone" : "bglr" }
{ "_id" : ObjectId("5cb9845c63ebec62ea803d45"), "bucket-id" : 4, "shard" : 4, "migration" : false, "zone" : "bglr" }
{ "_id" : ObjectId("5cb9845c63ebec62ea803d46"), "bucket-id" : 5, "shard" : 4, "migration" : false, "zone" : "bglr" }
{ "_id" : ObjectId("5cb9845c63ebec62ea803d47"), "bucket-id" : 6, "shard" : 4, "migration" : false, "zone" : "bglr" }
{ "_id" : ObjectId("5cb9845c63ebec62ea803d48"), "bucket-id" : 7, "shard" : 4, "migration" : false, "zone" : "bglr" }
```

MongoDB Authentication Validation

1. Make sure the database status of all the shards and sharding database members comes up with either PRIMARY, SECONDARY or ARBITER.

   ```
   show database status | tab
   ```

2. If the database status is not PRIMARY, nor SECONDARY or ARBITER, login to specific VM and check whether mongod instance is running appropriate options to enable mongo authentication or not.

   Example for working mongod instance with authentication enabled on it:

   ```
   root     15379     1  4 03:54 ?        00:16:49 mongod --keyFile=/mongodb.key
   --storageEngine mmapv1 --nojournal --noprealloc --smallfiles --ipv6 --bind_ip_all
   --port 27023 --dbpath=/mmapv1-tmpfs-27023 --replSet rs-shard-13 --quiet --slowms 500
   --logpath /data/db/mongo-27023.log --oplogSize 3221 --logappend --logRotate reopen
   ```

3. If the key file is present, but the database status is not good, check whether user exists or not for that mongod instance.

   Example:

   a. Login to VM and its mongo container (container name : mongo-s<instance>).

   b. Connect to mongod with its port.

   c. Use admin user and execute the following command:

Note The `db.getUsers()` command should display adminuser and backupuser for all the non-aribiter members.

For arbiter member users, connect to PRIMARY of that shard and execute `db.getUsers()` command.

```
session-sharddb:PRIMARY> db.getUsers()
[
        {
                "_id" : "admin.adminuser",
                "user" : "adminuser",
                "db" : "admin",
                "roles" : [
                        {
                                "role" : "root",
                                "db" : "admin"
                        }
                ]
        },
        {
                "_id" : "admin.backupuser",
                "user" : "backupuser",
                "db" : "admin",
                "roles" : [
                        {
                                "role" : "root",
                                "db" : "admin"
```

```
                                }
                            ]
                        }
                ]
```

4. If users are present, check whether mongo connection gets established manually or not by executing the following command:

   Example for Mongo Auth:

   ```
   mongo --ipv6 mongodb://adminuser:<password>@[2606:ae00:2001:230b::2b]:27019/admin
   ```

5. To validate the configurations and operational status of mongod instance, execute the following commands:

   ```
   db-authentication show-password database mongo password
   db-authentication rolling-restart-status database mongo password
   ```

Reconfigure the Databases

Here are few conditions when database reconfiguration is needed:

- All database members must be in STARTUP2 state for one or more replica sets.

  In shard replica-set, if all the data bearing members are down at the same time and arbiter is still running and once they are UP, they fail to elect new Primary and gets stuck in STARTUP2 state.

- Change in database configuration.

To reconfigure the databases, perform the following steps in the following sections:

Note The following steps are required for single cluster as well as multiple clusters. In case of mated pair deployments, the steps must be performed on all the sites.

Steps to be executed on DRA VNF

1. Login to vDRA VNF CLI and run `no binding shard-metadata-db-connection` command to remove shard metadata-db-connection configurations.

   Example:

   ```
   admin@orchestrator[an-master](config)# no binding shard-metadata-db-connection
      admin@orchestrator[an-master](config)# commit
      Commit complete.
      admin@orchestrator[an-master](config)# end
      admin@orchestrator[an-master]# show running-config binding
      % No entries found.
      admin@orchestrator[an-master]#
   ```

2. Login to vDRA VNF CLI and run `db-authentication remove-password database mongo` command to remove MongoDB password.

   Example:

   ```
   admin@orchestrator[an-master]# db-authentication remove-password database mongo
      Value for 'current-password' (<string>): ********
      admin@orchestrator[an-master]#
      admin@orchestrator[an-master]# db-authentication show-password database mongo
   ```

```
result Mongo password is not configured.
admin@orchestrator[an-master]#
```

Steps to be executed on DRA Binding VNF

1. Edit the nacm rule (config mode) to allow deleting the database configurations.

   ```
   nacm rule-list any-group rule data-base access-operations delete action permit
   ```

2. Delete database configurations (config mode).

   ```
   no database cluster <cluster name>
   ```

   where, <*cluster name*> is the name of the database cluster which has issues or need to be reconfigured.

Note As there are multiple database's clusters, you need to remove all the clusters from configuration.

3. Stop the system and wait for all the application containers to be stopped.

   ```
   system stop
   ```

4. Verify that the application containers are stopped by running `show scheduling status | include application` command. The command should not show any containers with issues.

5. (Only for MongoDB Authentication enabled database) Disable MongoDB authentication.

   ```
   db-authentication remove-password database mongo current-password XXX
   ```

6. To delete the persistent storage and old log information, run the following command on all VMs:

   ```
   rm -rf /data/mongod-node/db/*
   rm -rf /data/mongod-node/supervisord/supervisord-mongo.conf
   ```

   Example:

   ```
   $ for dbHost in `weave status connections | grep ss-persistence-db | awk '{print $2}'
    | cut -d':' -f1`;
   do ssh -o StrictHostKeyChecking=no -i cps.pem $dbHost 'sudo /bin/rm -fr
   /data/mongod-node/supervisord/supervisord-mongo.conf'; done
   ```

   ```
   $ for dbHost in `weave status connections | grep ss-persistence-db | awk '{print $2}'
    | cut -d':' -f1`;
   do ssh -o StrictHostKeyChecking=no -i cps.pem $dbHost 'sudo /bin/rm -fr
   /data/mongod-node/db/*'; done
   ```

Note Before proceeding to next step, make sure Step 1, on page 36 to Step 6, on page 36 has been executed from all the affected/reconfigured binding VNFs sites.

7. (Only when arbiter is on other site) Clear data directory for all STARTUP2 replica sets on arbiter VM.

   a. Find container where particular arbiter member is running. On arbiter site, run the following command to find arbiter hostname.

      ```
      show network ips | tab | include <arbiter-ipaddress>
      ```

   b. Find MongoDB container name for this host.

      ```
      show docker service | tab | include <host-name> | include mongo-s
      ```

c. Connect to MongoDB container by running `docker connect mongo-s<id>` command.

d. Clean the data directory. Stop all supervisor processes on arbiter container.

```
supervisorctl stop all
```

e. Clean database directoy on arbiter container.

```
rm -fr /mmapv1-tmpfs-*/*
```

f. Start all supervisor processes on arbiter container.

```
supervisorctl start all
```

8. Start the system and wait for system percentage to turn 100%.

```
system start
```

`show system status` should show system percentage as 100.

9. Apply the database configurations again (config mode).

Before proceeding to next step, make sure Step 8, on page 37 to Step 9, on page 37 has been executed from all the affected/reconfigured binding VNFs sites.

10. Verify all the database sets are UP by running `show database status | exclude "PRIMARY|SECONDARY|ARBITER"` command.

Note `show database status | exclude "PRIMARY|SECONDARY|ARBITER"` command should not show any unhealthy database member.

Before proceeding to next step, make sure Step 9, on page 37 to Step 10, on page 37 has been executed from all the affected/reconfigured binding VNFs sites.

11. (Only for MongoDB Authentication enabled database) Enable MongoDB authentication.

For more information on enabling MongoDB authentication, see *Configuring MongoDB Authentication* section in the *CPS vDRA Configuration Guide*.

Note Before proceeding to next step, make sure this step has been executed from all the affected/reconfigured binding VNFs sites.

Example 1: Enable MongoDB authentication with transition.

```
db-authentication set-password database mongo password XXX confirm-password XXX
db-authentication enable-transition-auth database mongo
db-authentication rolling-restart database mongo
db-authentication rolling-restart-status database mongo
Verify database member in healthy state
```

Example 2: Enable MongoDB authentication without transition.

```
db-authentication disable-transition-auth database mongo
db-authentication rolling-restart database mongo
db-authentication rolling-restart-status database mongo
```

12. Wait for all the database sets to come UP. Verify the status by running `show database status | exclude "PRIMARY|SECONDARY|ARBITER"` command.

Note `show database status | exclude "PRIMARY|SECONDARY|ARBITER"` command should not show any unhealthy database member.

13. Login to vDRA VNF CLI and restart the binding containers to read the latest metadata from new configurations.

Note Before proceeding to next step, make sure this step has been executed from all the affected/reconfigured DRA VNFs.

```
docker restart container-id <container-id>
```

Example:

```
docker restart container-id binding-s<number>
```

`binding-s<number>` is an example for binding container. binding-s<*number*> is the container-id. <*number*> varies according to which worker VM binding container is running.

14. Disable deleting the database configurations by editing the nacm rule (config mode).

Note Make sure this step has been executed from all the affected/reconfigured binding VNFs sites.

```
nacm rule-list any-group rule data-base access-operations delete action deny
```