CPS vDRA Operations Guide, Release 22.1.0

First Published: 2022-03-24

Last Modified: 2022-03-24

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Preface

About This Guide

Note The documentation set for this product strives to use bias-free language. For purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. While any existing biased terms are being substituted, exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

This document is a part of the Cisco Policy Suite documentation set.

For information about available documentation, see the *CPS Documentation Map* for this release at Cisco.com.

Note The PATS/ATS, ANDSF, and MOG products have reached end of life and are not supported in this release. Any references to these products (specific or implied), their components or functions in this document are coincidental and are not supported. Full details on the end of life for these products are available at: https://www.cisco.com/c/en/us/products/wireless/policy-suite-mobile/eos-eol-notice-listing.html.

Audience

This guide is best used by these readers:

- Network administrators

- Network engineers
- Network operators
- System administrators

This document assumes a general understanding of network architecture, configuration, and operations.

Additional Support

For further documentation and support:

- Contact your Cisco Systems, Inc. technical representative.
- Call the Cisco Systems, Inc. technical support number.
- Write to Cisco Systems, Inc. at support@cisco.com.
- Refer to support matrix at https://www.cisco.com/c/en/us/support/index.html and to other documents related to Cisco Policy Suite.

Conventions (all documentation)

This document uses the following conventions.

Conventions	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
courier font	Terminal sessions and information the system displays appear in courier font.
< >	Nonprinting characters such as passwords are in angle brackets.

Conventions	Indication
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note Means reader take note. Notes contain helpful suggestions or references to material not covered in the manual.

Caution Means reader be careful. In this situation, you might perform an action that could result in equipment damage or loss of data.

Warning IMPORTANT SAFETY INSTRUCTIONS.

Means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

Note Regulatory: Provided for additional information and to comply with regulatory and customer requirements.

Communications, Services, and Additional Information

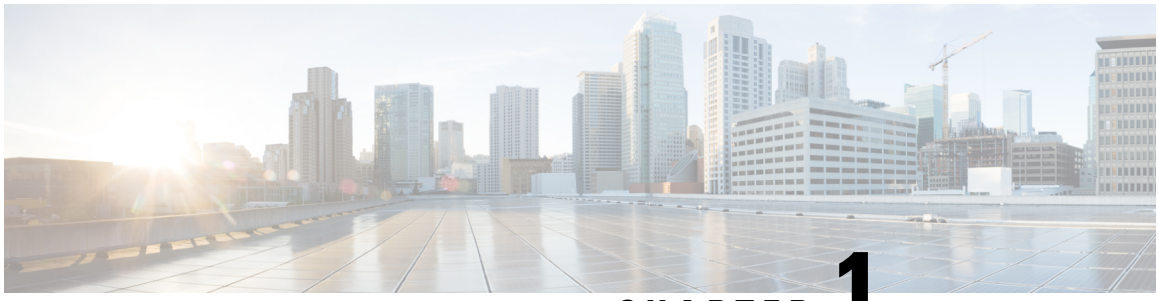
- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

Cisco Bug Search Tool

Cisco Bug Search Tool (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.

Important Notes

Important Any feature or GUI functionality that is not documented may not be supported in this release or may be customer specific, and must not be used without consulting your Cisco Account representative.



CHAPTER 1

Managing CPS vDRA Cluster

Accessing CPS vDRA Management CLI

There are two options for accessing the CPS vDRA Management CLI.

Access Via Web Browser

Perform the following steps to access the CPS vDRA Management CLI:

Step 1 Enter the following URL in Firefox or Chrome:

https://<masterip>/

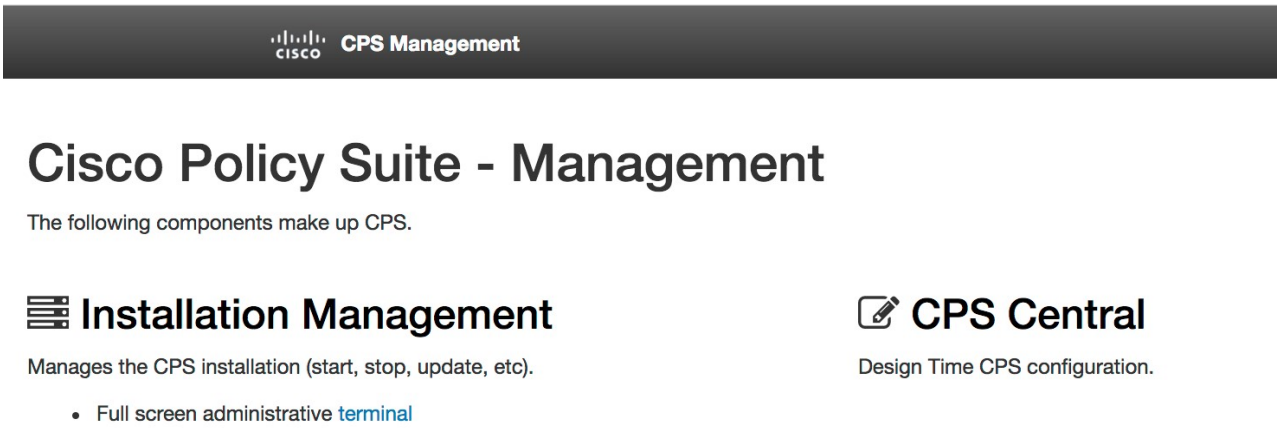
Step 2 Login to the application using your user ID and password.

Step 3 Follow the Installation Management hyperlink in the following screen:

Figure 1: CPS DRA Login

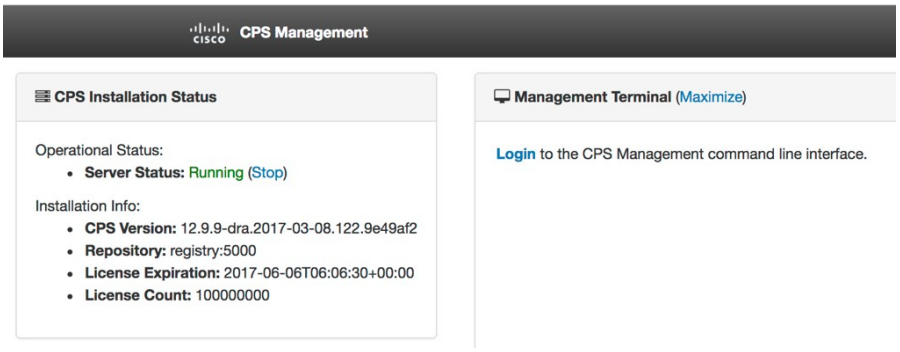
Step 4 In the Management screen, click the **Login** link to display the in-browser terminal window.

Figure 2: Installation Management



Step 5 Login with a valid user name and password.

Figure 3: Management Terminal Link



Access Via SSH

Access is available to the CPS vDRA via SSH listening on port 2024 of the master virtual machine. This port must be open in the OpenStack security rules in order to access the Management CLI via SSH.

Starting CPS vDRA Cluster

A CPS vDRA cluster is a self-organizing cluster that does not require operator actions to configure the system when you follow the instructions found in the installation guide. The system self-organizes by following the algorithm:

1. The cluster master node is started and bootstraps the Docker engine, an embedded Docker registry, the Weave overlay network, and the CPS vDRA scheduling application.

2. The worker nodes are started either after the master node is started or in parallel. The bootstrapping of the Docker engine and Weave overlay network point back to the master node.
3. The scheduling function on the master node begins an auto discovery function on engine startup of the Docker engines that have joined the Weave overlay network.
4. For each engine discovered, the system queries the Docker engine configuration to discover the node identifier and the role within the cluster that the engine will perform. The roles are used by the scheduling function to map application services to the appropriate virtual machines.
   a. The CPS vDRA application (for both Policy DRA and IMS DRA solutions) supports the following roles:
      1. master – This is always the master scheduling node.
      2. control-a[b] – This is a control node that works in concert with the other control node and the master node to provide OAM support for the application.
      3. diameter-endpoint – This is the node where all diameter traffic terminals.
      4. binding-worker – This is the node where binding/slf queries are executed.
   b. The vDRA Binding and SLF application supports the following roles:
      1. master – This is always the master scheduling node.
      2. control-a[b] – control node that works in concert with the other control nodes and the master node to provide OAM support for the application.
      3. persistence-router – node where binding/slf queries are routed.
      4. persistence-db – nodes where the binding database replica sets are located.
5. As the Docker engines are registered, the scheduling application begins executing a controlled startup by starting modules as the underlying engines become available.
   a. A module is a set of interrelated services that are started, stopped and scaled as a set of related processes. These processes are either collocated on the same virtual machine or across multiple virtual machines. There are three type of modules that exist:
      1. infrastructure – These are core modules that are not shutdown when the application shuts down.
      2. application – These are modules that are removed when the application is shutdown.
      3. External – These are external services that are installed on the system and whose images are built and loaded outside of the system. See the **scheduling external-service** command for more information on configuring external services.

Stopping Application Services In CPS vDRA Cluster

The modules of type “application” can be shut down in a controlled manner by running the **system stop** command. This command will unload all modules in reverse run-level order and stop the associated running Docker services.

Starting Services In CPS vDRA Cluster

The modules of type “application” can be started in a controlled manner by running the **system start** command. This command will start all modules in run-level order and schedule the underlying Docker services on the registered Docker engines.

Stopping External Services In CPS vDRA Cluster

The modules of type “external” can be shut down in a controlled manner by running the **system disable-external-services** command. This command will unload all modules in reverse run-level order and stop the associated running Docker services.

Starting External Services In CPS vDRA Cluster

The modules of type “external” can be shut down in a controlled manner by running the **system enable-external-services** command. This command will unload all modules in reverse run-level order and stop the associated running Docker services.

Restarting An Individual Docker Service

Perform the following steps to restart an individual docker service:

Step 1 Run the **show docker service** command to locate the container ID of the service to restart.

```
scheduler# show docker service

PENALTY
MODULE           INSTANCE  NAME               VERSION          ENGINE  CONTAINER ID       STATE
BOX      MESSAGE
-------------------------------------------------------------------------------------------------------------
admin-db         1         mongo-admin-a      3.6.9.0          aio     mongo-admin-a      HEALTHY
false    -
admin-db         1         mongo-admin-arb    3.6.9.0          aio     mongo-admin-arb    HEALTHY
false    -
admin-db         1         mongo-admin-b      3.6.9.0          aio     mongo-admin-b      HEALTHY
false    -
admin-db         1         mongo-admin-setup  12.9.9-SNAPSHOT  aio     mongo-admin-setup  HEALTHY
false    -
consul           1         consul-1           12.9.9-SNAPSHOT  aio     consul-1           HEALTHY
false    -
consul           1         consul-2           12.9.9-SNAPSHOT  aio     consul-2           HEALTHY
false    -
consul           1         consul-3           12.9.9-SNAPSHOT  aio     consul-3           HEALTHY
false    -
foobar           1         foobar             3.2.6.0          aio     foobar             HEALTHY
false    -
grafana          1         grafana            12.9.9-SNAPSHOT  aio     grafana            HEALTHY
false    -
haproxy-common   1         haproxy-common     12.9.9-SNAPSHOT  aio     haproxy-common-s1  HEALTHY
false    -
```

```
orchestrator-ui  1         orchestrator-ui    12.9.9-SNAPSHOT  aio     orchestrator-ui    HEALTHY
false    -
subversion       1         svn                12.9.9-SNAPSHOT  aio     svn                HEALTHY
false    -
```

Step 2 Using the provided container-id, run the **docker restart container-id** *container-id* command. This will issue a non-graceful stop on the Docker container and move the state of the container to ABORTED. The container will stay in this state for 10 seconds before restarting.

Step 3 Verify the health of the restarted docker service by running the **show docker service** command again and waiting for the service to progress into the HEALTHY state. Optionally the log of the individual container can be followed by running the **monitor log container** *container-id* using the same container ID from Step 2, on page 6.

CPS External Authentication and Authorization

CPS system supports LDAP external authentication and authorization.

Based on Conf-D group configurations, CPS roles are assigned to the applications running on CPS cluster.

The following command configures the gid mapping for various roles.

```
admin@orchestrator(config)# external-aaa pam gid-mapping
 1000 policy-admin
admin@orchestrator(config-gid-mapping-1000/policy-admin)# commit
Commit complete
```

You can also view the status of configuration with the following command:

```
admin@orchestrator# show running-config external-aaa | tab
```

Sample Output:

```
admin@orchestrator# show running-config external-aaa | tab
GID GROUP
--------------------
1000 policy-admin
```

Conf-D Group to CPS Roles Description

The following table describes the CPS roles and Conf-D groups of applications/services:

Table 1: Conf-D Group to CPS Roles Description

Application/Service	CPS Role	Conf-D Groups
Control center	SUMADMIN	crd-read-write
Control center	READONLY	crd-read-only
Policy Builder	READ&WRITE	policy-admin
Policy Builder	READ	*
SVN	READ&WRITE	policy-admin
SVN	READ	*

Application/Service	CPS Role	Conf-D Groups
Grafana	Admin	grafana-admin
Grafana	Editor	grafana-editor
Grafana	Viewer	*

***** Indicates all authenticated users

Bulkstats conf-D group: sftp daemon running on port 2026 retrieves all statistics within the /var/broadhop/stats directory. Users associated to the “bulkstats” or “admin” group are able to retrieve statistics.

Oper conf-D group is not used.

vDRA Containers

The following table describes the modules, containers, and the respective VM location in vDRA:

Module	Container	VM on which container runs	Description
admin-db	mongo-admin-a	master	Stores the collection of system and CRD related configurations
admin-db	mongo-admin-b	control-a	Stores the collection of system and CRD related configurations
admin-db	mongo-admin-c	control-b	Stores the collection of system and CRD related configurations
admin-db	mongo-admin-setup	master	Sets up the mongo database cluster across the master, control-a and control-b
binding	binding	dra-worker	Provides functionality for handling the requests from diameter-endpoint to binding database and vice versa
cc-monitor	cc-monitor	control-a, control-b	Manages haproxy instance for memcached servers and also for the collection of consolidated qns and engine logs.

Module	Container	VM on which container runs	Description
configuration-engine	configuration-engine	control-a	Maintains confd configuration engine details
consul	consul-1	master	Service discovery and configuration
consul	consul-2	control-a	Service discovery and configuration
consul	consul-3	control-b	Service discovery and configuration
control-plane	control-plane	master,control-a, control-b	Passes topology information via control messages from publishers to subscribers.
control-plane	control-plane-monitor	master,control-a, control-b	Monitors server running in control-plane container and restarts if the same is not responsive or down
diameter-endpoint	diameter-endpoint	dra-director	Maintains Diameter endpoint inbound and outbound connections,message handling and routing function.
diameter-endpoint	diameter-redis-q-a	dra-director	Facilitate inter process communication of application messages across nodes.
diameter-endpoint	diameter-redis-q-a-monitor	dra-director	Monitor IPC server process in "diameter-redis-q-a" and restarts if the same is not responsive or down
diameter-endpoint	global-control-plane	dra-director	Passes topology information via control messages from publishers to subscribers across DRA installations

Module	Container	VM on which container runs	Description
diameter-endpoint	interface-mover	dra-director	Provides functionality for moving of SCTP interface from host to inside container.
diameter-endpoint	socket-forwarder	dra-director	Forwards the socket bind connections from host to inside container
docker-registry	registry	master	Internal docker registry for storing and distributing of images running on the system
docker-registry	registry-extra	master	Utility container to support docker registry
grafana	grafana	control-a/control-b	Provides a graphical or text-based representation of statistics and counters collected in the Prometheus database
haproxy-common	haproxy-api	on all nodes except dra-worker	haproxy instance for the load balancing of API servers
haproxy-common	haproxy-common	on all nodes except dra-worker	Common haproxy instance for the load balancing of Policy Builder, Grafana, orchestrator CLI and UI, API, CC, etc.
haproxy-int-api	haproxy-int-api	control-a	haproxy instance for the load balancing of internal API servers.
haproxy-prometheus	haproxy-prometheus	control-a/control-b	haproxy instance for the load balancing of Prometheus services.

Module	Container	VM on which container runs	Description
memcached-vip	lbvip02		In-memory key-value store for small chunks of arbitrary data (strings, objects) from results of database calls, API calls, or page rendering. Intended for use in speeding up dynamic web applications by alleviating database load.
mongo-node	mongo	master, control-a, control-b	Maintains sharded clusters for managing of huge data.
mongo-node	mongo-monitor	master, control-a, control-b	Monitoring of Mongo shards that run on Mongo containers.
mongo-node	mongo-status	master	Monitoring of Mongo database configurations
monitoring	collectd-host	All	The collection utility collectd is used for collecting and storing statistics from each VM to the centralized collection nodes on the control-A and control-B virtual machines. The centralized collector writes the collected data to output CSV files.
monitoring	dnsmasq	All	Used for internal DNS forwarding and caching
monitoring	dnsmasq-monitor	All	Monitoring and managing dnsmasq container
monitoring	docker-host-info	All	Sytem utility container used for executing all system related commands
monitoring	keepalived	All	Manages the VIPs configured via VRRP protocol

Module	Container	VM on which container runs	Description
monitoring	keepalived-monitor	All	Monitors the keepalived process running on the system and starts the keepalived process with the given VIP name
monitoring	node-exporter	All	Exporter for the System metrics like CPU, RAM, DISK etc
monitoring	node-exporter-monitor	All	Monitoring of node exporter container
monitoring	ntpd	All	NTP service for time synchronization that runs either realtime or on client process based on the reachability of the NTP server .
orchestrator	orchestrator	master	1. Creates and maintains docker engines 2. Schedules and manages docker services 3. All system operations like upgrade, downgrades 4. CLI operations 5. Alert and SNMP functionalities et
orchestrator-backup-a	orchestrator-backup-a	control-a	Provides high availability support for the functionalities carried out by the orchestrator.
orchestrator-backup-b	orchestrator-backup-b	control-b	Provides high availability support for the functionalities carried out by the orchestrator.
orchestrator-ui	orchestrator-ui	master, control-a, control-b	To access the management console via HTTP

Module	Container	VM on which container runs	Description
policy-builder	policy-builder	control-a, control-b	Service configurations and policy rules
prometheus	blackbox-exporter	master, control-a, control-b	**Note**: Will be obselete in future releases, as ICMP statistics are now collected from orchestrator
prometheus	prometheus-hi-res	master, control-a, control-b	Monitors the system at 5-second intervals with 24-hour history
prometheus	prometheus-planning	master, control-a, control-b	Monitors the system at 120-second intervals with 365-day history
prometheus	prometheus-trending	master, control-a, control-b	Monitors the system at 20-second intervals with 30-day history
prometheus	statistics-gathering	master, control-a, control-b	Collection of statistics related to java applications as bulk stats
stats	collectd-jmx	control-a, control-b	Collection of statistics related to jmx using collectd
stats	stats-relay	control-a, control-b	Collection of statistics related to relay interfaces using collectd
stats	stats-sftp	control-a, control-b	Collection of statistics related to sftp
subversion	svn	control-a/control-b	Maintains all the CPS policy configurations and has repositories in which files can be created, updated and deleted
zvision	haproxy-zvision	master, control-a, control-b	haproxy instance for the load balancing of zvision servers
zvision	zvision	master, control-a, control-b	Provides functionality of Zing VM monitoring

Installing New Software Images

When a new ISO is provided with software, you need to perform the following steps to upgrade the current system software:

Step 1 Download the ISO image from CCO site.

Step 2 Copy the ISO to DRA VNF /data/iso/staged-isos.

Step 3 Run the following commands:

```
system software iso load category product file <ISO file name>
 activate true
```

```
show system software available-versions
```

Step 4 Repeat the steps for the DRA database ISO.

Upgrading to New Software Version

Perform the following steps to upgrade to a new software version:

Before you begin

Take a snapshot of the consul state to be used in case a rollback is required.

1. Login to CLI mode.

   ```
   docker connect consul-1
   ```
2. Take the backup and exit the CLI mode.

   Example:

   ```
   consul snapshot save <SITE-2-19.4-DBVNF-consul-backup.snap>
   ```
3. Copy the consul snapshot from orchestrator container to master VM.

   Example:

   ```
   docker cp consul-1:/ SITE-2-19.4-DBVNF-consul-backup.snap
   ```
4. Copy the backup to installer VM.

   Example:

   ```
   scp -i cps.pem <backupdirectorypath>/SITE-2-19.4-DBVNF-consul-backup.snap
   cps@<installerip>:/home/cps
   ```

Step 1 Run the following command:

```
system software iso load category product file cisco-policy-dra.iso activate true
```

Step 2 In the Management CLI, run **show system software available-versions** to determine if the correct version of has been uploaded:

```
scheduler# show system software available-versions
VERSION
---------------------------------
12.9.9-dra.2017-03-08.122.9e49af2
```

Step 3 In the Management CLI, run the **system upgrade version** command to upgrade to the version found in Step 2, on page 13:

```
scheduler# system upgrade version 12.9.9-dra.2017-03-08.122.9e49af2
```

At this point the application will begin downloading the new scheduling and application images from the on-board Docker Registry. The download will take several seconds and the scheduler application will disconnect and restart. You must re-login after the disconnect occurs.

Step 4 In the Management CLI, run the **show scheduling status** command to validate the progress of the upgrade.

Aborting an Upgrade

If an in-progress upgrade needs to be aborted, run the **system abort-upgrade** command. This will immediately stop all scheduling activities. Reverting to the previous versions is triggered by the downgrade to a previous software version procedure.

Downgrading to Previous Software Version

Perform the following steps to downgrade to a previous software version:

Before you begin

Make sure older version consul snapshot is listed by executing `consul list-snapshots` command.

If the snapshot is not available, copy the older version consul snapshot taken Upgrading to New Software Version, on page 13 to the directory `/data/orchestrator/config/snapshot-consul` in master VM.

Trigger the DRA App VNF downgrade to older version (for example, 19.4.0 release) with consul downgrade (entire ISO downgrade) using `system downgrade version <version-qualifier> consul-downgrade true snapshot-name <snapshot-name>` command.

Example: `system downgrade version 19.4.0-20200625_121852.7720 consul-downgrade true snapshot-name SITE-2-19.4-DRAVNF-consul-backup.snap`

Step 1 Select the qualifier for the version you want to downgrade and then activate the ISOs for downgrading as shown in the following example:

```
system abort-upgrade
show system software iso details
| tab
CATEGORY NAME           VERSION QUALIFIER                   CREATED                     ACTIVE SIZE
 MB
-------------------------------------------------------------------------------------------------------
product cisco-policy-dra 13.1.1 dra.2017-12-06.1366.b800a6d 2018-03-02T23:37:21.848+00:00 false
1339.99
product cisco-policy-dra 13.1.1 dra.2018-02-28.1793.f618c58 2018-03-12T22:42:19.225+00:00 false
```

```
1341.93
product cisco-policy-dra 13.1.1 dra.2018-03-28.1938.f618c58 2018-04-13T21:10:34.872+00:00 true
1342.13
admin@orchestrator[mpsl14fdrm01v]# system software iso activate category product
name cisco-policy-dra version 13.1.1 qualifier dra.2018-02-28.1793.f618c58
```

Step 2 In the Management CLI, run the **show system software available-versions** to determine if the correct version has been uploaded:

```
scheduler# show system software available-versions
VERSION
----------------------------------
12.9.9-dra.2017-03-08.122.9e49af2
```

Step 3 In the Management CLI, run the **system downgrade version** command to upgrade to the version found in Step 2, on page 15:

```
scheduler# system downgrade version 12.9.9-dra.2017-03-08.122.9e49af2
```

At this point the application begins downloading the new scheduling and application images from the on-board Docker Registry. The download takes several seconds and the scheduler application disconnects and restarts. You must re-login after the disconnect occurs.

Note During downgrade, make sure consul is using the proper snapshot file after downgrade. If a consul snapshot was taken before the upgrade to the running version, find the list of available consul snapshots using the following command:

```
scheduler# consul list-snapshots
```

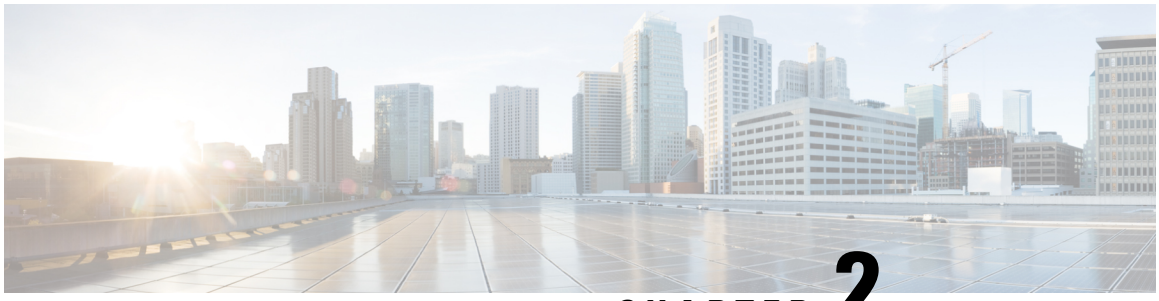
Select the correct consul snapshot for the version to be downgraded and downgrade DRA and consul using the following command:

```
scheduler# system downgrade version 12.9.9-dra.2017-03-08.122.9e49af2 consul-downgrade true
snapshot-name 12.9.9-dra.snap
```

Step 4 In the Management CLI, run the **show scheduling status** command to validate the progress of the upgrade.

Aborting a Downgrade

If an in-progress downgrade needs to be aborted, run the **system abort-downgrade** command. This will immediately stop all scheduling activities. Reverting to the previous versions is triggered by the upgrading to a new software version procedure.



CHAPTER 2

Prometheus and Grafana

Introduction

CPS system, application statistics and Key Performance Indicators (KPI) are collected by the system and are displayed using a browser-based graphical metrics tool. This chapter provides a high-level overview of the tools CPS uses to collect and display these statistics.

Prometheus

Prometheus is an application that is used to actively gather statistics and trigger alerts from the running virtual machines and application services. The CPS vDRA cluster deploys the following Prometheus services on each control node and on the master node:

- Prometheus Hi-Res – this instance of the Prometheus service is monitoring the system at 5 second intervals with 48-hour history
- Prometheus Trending – this instance of the Prometheus service is monitoring the system at 20 second intervals with 30-day history
- Prometheus Planning – this instance of the Prometheus service is monitoring the system at 120 second intervals with 365-day history

Internally, the Prometheus servers scrape statistics from target statistics sources on a regular basis. The following target data sources are included:

- Host Node Exporter for Host VM statistics.
- Mongo DB Exporter for Database statistics.
- Application Statistics.

In addition to scrapping, statistics in the Prometheus servers can be configured using the Management CLI alert rule command to trigger alerts on error conditions. In this scenario, a user defines the alert rule and the configuration for that rule is pushed into the Prometheus servers. It can generate SNMPv2 and SNMPv3 alarm based on the NMS destination configured in the system. You can configure multiple SNMP destination (SNMPv2, SNMPv3) to receive the alarms at multiple NMS.

Note Currently, SNMP get and walk facility is not supported.

For more information on Prometheus, refer https://prometheus.io/.

Prometheus Queries

The CPS vDRA supports exposing of Prometheus API queries on OAM network using HAProxy. vDRA allows operators to fetch necessary statistics from the system through the Prometheus API and further analyze in a single consolidated view. The following functions are supported:

- vDRA data gets pulled from Prometheus API and loaded directly into system for visualization. This includes data from the following three data stores:
  - Prometheus Hi-Res
  - Prometheus Trending
  - Prometheus Planning
- The Maximum TPS that is required for enabling these queries at required intervals are:
  - TPS: The TPS value is 160K.
  - Intervals: one minute and five minutes
- GET queries: GET /api/v1/query
- TLS or HTTPS-based authentication

Following statistics are collected from target sources and are available through Prometheus APIs:

- Host Node Exporter for Host VM statistics
- Mongo DB Exporter for Database statistics
- Application Statistics

Configuring HAProxy

To expose the Prometheus data to external users, you should modify HAProxy configurations on haproxy-common containers.

Set up HAProxy configurations to accept incoming requests on port 443. HAProxy then checks their URL paths or Prometheus and then forwards them to the correct backend.

The frontend and backend settings are segregated based on the URLs that are used for querying and the backend data stores respectively.

For example, different configurations are considered for frontend of Prometheus hi-res data where the backend is the Prometheus hi-res data store. Similarly, different configurations are used for Prometheus planning and trending data stores.

The following endpoint evaluates an instant query at a single point in time: `GET /api/v1/query`

Example 1:

```
curl -v -k -u
admin:admin https://172.18.63.223/trending_prometheus
/api/v1/query_range?query="sum((docker_service_up%7Bcontainer_name%
20%3D~%20%22diameter-endpoint-s.*%22%7D%3D%3D2)%2F2)
&start=1639029600&end=1639029915&step=15"
```

Example 2:

```
curl -v -k -u
admin:admin https://172.18.63.223/trending_prometheus
/api/v1/query_range?query="sum((docker_service_up
%7Bcontainer_name%20%3D~%20%22binding-s.*%22%7D%3D%3D2)%2F2)&
start=1639029675&end=1639029990&step=15"
```

Exposing Prometheus Hi-res, Trending, and Planning Data

Use the following table details to expose Prometheus Hi-res, trending, and planning data.

Note Installer IP refers to the virtual IP of OAM servers (master/control-0/control-1) that are exposed to external users. The **Query** field is provided with the required Prometheus queries.

Prometheus Service	Description	URL	Authentication
Prometheus - Hi-res data	This instance of the Prometheus service monitors the system at 5 second intervals with 48-hour history.	`"https://installer/hi_res_prometheus` `/api/v1/query_range?query="" "`	HTTPS or TLS-based authentication is supported.
Prometheus - Trending	This Prometheus service monitors the system at 20 second intervals with 30-day history.	`https://installer/trending_prometheus` `/api/v1/query_range?query=" "`	HTTPS or TLS-based authentication is supported.

Prometheus Service	Description	URL	Authentication
Prometheus - Planning	This Prometheus service monitors the system at 120 second intervals with 365-day history.	`https://installer/planning_prometheus /api/v1/query_range?query=" "`	HTTPS or TLS-based authentication is supported,

Grafana

Grafana is a third-party metrics dashboard and graph editor provided with CPS 7.0 and higher. Grafana provides a graphical or text-based representation of statistics and counters collected in the Prometheus database.

Note After the DRA Director (DD) failover/reboot, the TPS values in Grafana dashboards takes approx. 5 minutes to fetch and display the latest updated values. Until the values are updated, Grafana displays the old data.

Additional Grafana Documentation

This chapter provides information about the CPS implementation of Grafana. For more information about Grafana, or access the general Grafana documentation, refer to: http://docs.grafana.org.

Data Source Supported

The CPS implementation uses the Prometheus data source and does not use Graphite for queries. This requires the definition of queries to use the Prometheus query format as defined in https://prometheus.io/docs/querying/basics/.

Note After changing respective KPI panel's width to 24 (which is maximum), you can get all the spikes captured for 6 hours duration. So, if tou need to analyse longevity report for 12 hours or more, you can grep data by grouping in 6 hours interval.

Note If the control VM that hosts Grafana goes down, then the Prometheus data also not available during that downtime after the same control VM (hosting Grafana) is back. This results in some missing data. As a workaround, you can add the Prometheus datasource of other control VM in Grafana UI that was up during that downtime and view the missing statistics.

Note The `top` command output must not be compared with the Grafana CPU statistics panel display.

Manage Grafana Users

Note In Grafana, admin users can invite new users by email or a link. However, this is not supported in CPS vDRA.

Perform the following to add a new Grafana:

1. Enter config mode

```
scheduler# config
Entering configuration mode terminal
scheduler(config)#
```

2. Enter the **aaa authentication** command to create the user:

```
scheduler(config)# aaa authentication users user test2 gid 100 uid 9000 homedir / password
 testpassword ssh_keydir /
scheduler(config-user-test2)# commit
scheduler(config-user-test2)# exit
```

Note The **gid**, **uid**, **homedir** and **ssh_keydir** are required but not used by the application.

Add User To A Viewer Operational Group

In config mode, add the user to the “oper” group and commit as follows:

```
scheduler(config)# nacm groups group oper user-name test2
scheduler(config-group-oper)# commit
```

Add User To A Grafana Editor Group

In config mode, add the user to the “grafana-editor” group and commit as follows:

```
scheduler(config)# nacm groups group grafana-editor user-name test2
scheduler(config-group-grafana-editor)# commit
```

Add User To A Grafana Admin Group

In config mode, add the user to the “grafana-admin” group and commit as follows:

```
scheduler(config)# nacm groups group grafana-admin user-name test2
scheduler(config-group-grafana-admin)# commit
```

Change A Grafana Users Password

In the Management CLI, issue the **aaa authentication users** user **change-password** command as follows:

```
scheduler# aaa authentication users user test2 change-password
Value for 'old-password' (<string>): ************
Value for 'new-password' (<string>): ********
```

```
Value for 'confirm-password' (<string>): ********
scheduler#
System message at 2017-03-08 21:17:18...
Commit performed by system via system using system.
```

Specify Access Restrictions for a Group

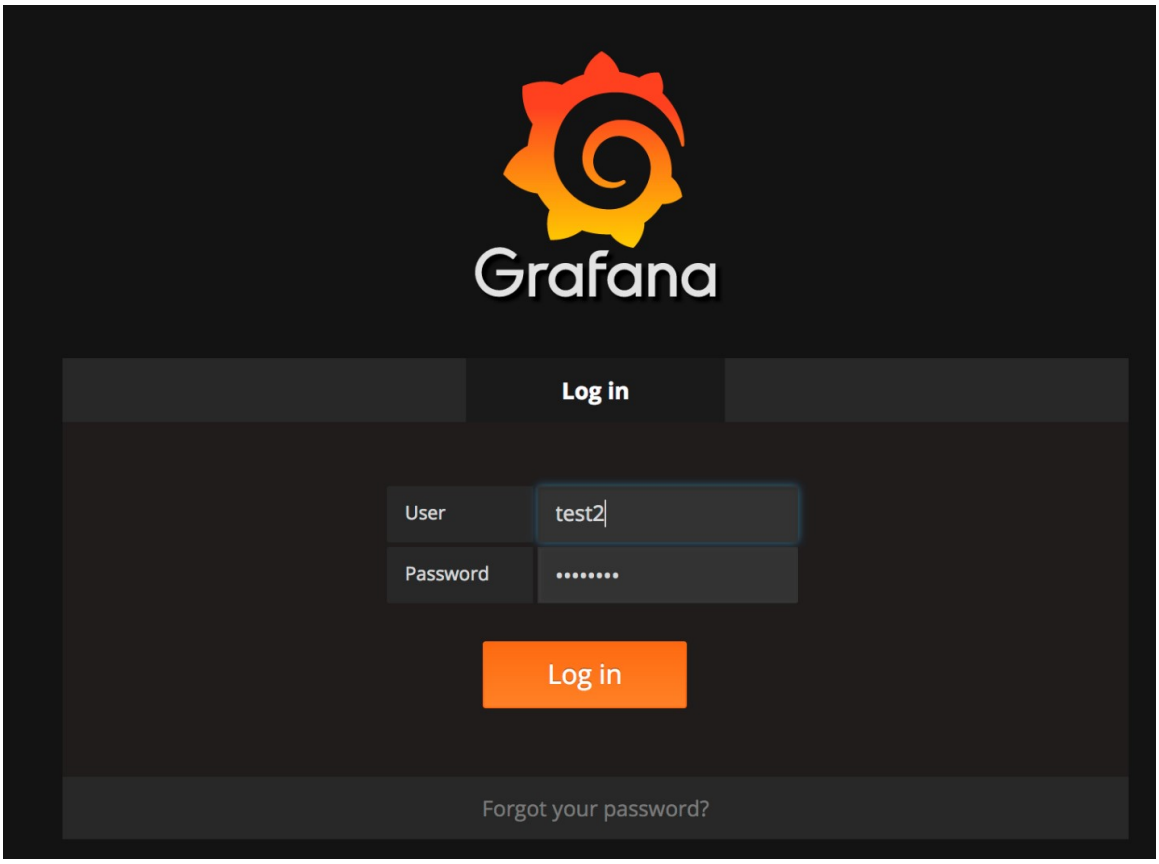
For more information, see the nacm rule-list command.

Connect to Grafana

Use the following URL to access Grafana and enter the user name and password:

https://<*masterip*>/grafana/

Figure 4: Grafana Login



Attention DRA is using the Grafana login page maintained as a part of Grafana code base. By default, when you open a web page in a new tab by clicking on a link with `target="_blank"`, you allow an attacker to redirect users clicking such a link to another web page. The issue is that the redirect concerns the initial tab (your web page), not the newly opened window. Also, the redirect is done without any warning. This can be used as a very effective phishing method. This kind of phishing method is called (reverse) tab nabbing. This issue of `target="_blank"` attribute is present in Grafana 5.2.3 used by DRA.

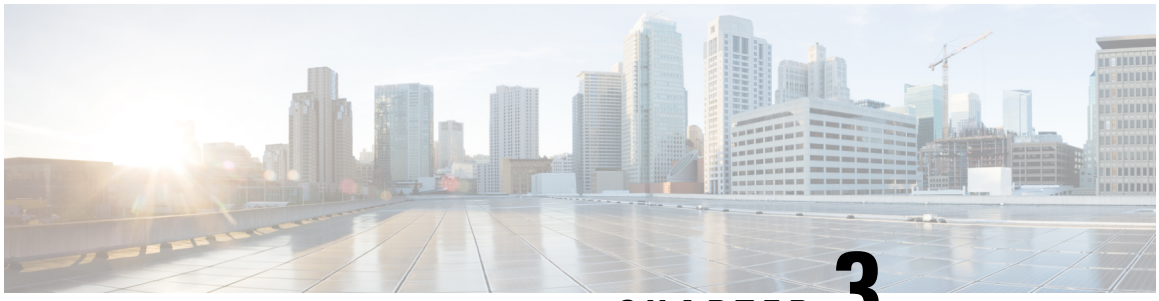
If you have to use `target="_blank"` attribute, you must also add `: rel="noopener"`. This attribute sets the **window.opener** value to null (forbids any URL change on the referring page). The `rel="noopenener"` attribute has been added in the latest version of Grafana for fixing this issue.

This is not a security vulnerability in CPS product. CPS uses Grafana in a controlled environment and no tab nabbing is possible.

Grafana Roles

The following types of user roles are supported:

- Admin: An admin user can view, update and create dashboards. Also, the admin can edit and add data sources and organization users.
- Viewer: A viewer can only view dashboards and cannot not save or create them.
- Editor: An editor can view, update and create dashboards.



CHAPTER 3

Managing CPS Interfaces and APIs

CPS vDRA Interfaces And APIs

CPS vDRA includes various application APIs to configure and manage the application.

CRD REST API

Purpose

The Custom Reference Data (CRD) REST API enables the query of creation, deletion, and update of CRD table data without the need to access the Control Center GUI. The CRD APIs are available using an HTTP REST interface.

URL and Port

https:// <*master ip or control node* >:443/custrefdata

Protocol

HTTPS

Accounts and Roles

Security for the CRD REST API is accomplished by using HTTP basic authentication to support read-only and read-write access to the CRD REST API.

Assigning a Read-Only User

Use the **nacm groups group** command to assign the user to the "crd-read-only" group.

For Example, nacm groups group crd-read-only user-name oper

Grafana

Purpose

Grafana is a metrics dashboard and graph editor used to display graphical representations of system, application KPIs, bulkstats of various CPS components.

Note After the DRA Director (DD) failover/reboot, the TPS values in Grafana dashboards takes approx. 5 minutes to fetch and display the latest updated values. Until the values are updated, Grafana displays the old data.

URL and Port

https:// <*master ip or control node* >:443/grafana

Protocol

HTTPS

Accounts and Roles

For more information on adding or deleting these user accounts, refer to the *Prometheus and Grafana* chapter in this guide.

JMX Interface

Purpose

Java Management Extension (JMX) interface can be used for managing and monitoring applications and system objects.

Resources to be managed or monitored are represented by objects called managed beans (mbeans). MBean represents a resource running in JVM and external applications can interact with mbeans through the use of JMX connectors and protocol adapters for collecting statistics (pull), for getting/setting application configurations (push/pull), and notifying events like faults or state changes(push).

CLI Access

Perform the following steps to access the jmxterm:

1. Run **docker connect** *container-id*.
2. Run the jmxterm command from the CLI prompt to bring up the jmx terminal

Port

All applications run on port 9045.

This port is not exposed externally.

Accounts and Roles

Not applicable.

OSGi Console

Purpose

CPS is based on Open Service Gateway initiative (OSGi) and OSGi console is a command-line shell which can be used for analyzing problems at OSGi layer of the application. It may become necessary to connect to the OSGi console to execute specific commands. These commands are not documented in this guide but the connection process is described below.

CLI Access

Use the following command to access the OSGi console:

1. Run the command **docker connect** *container-id*.
2. `telnet <ip> <port>`

Ports

All applications run on port 9091 within the executing container.

This port is not exposed externally.

Accounts and Roles

Not applicable.

Policy Builder GUI

Purpose

Policy Builder is the alternative web-based client interface for the configuration of the Cisco Policy Suite.

URL and Port

https://<*master or control ip*>/pb

Protocol

HTTPS

Accounts and Roles

Assigning a Read-Only User

It is not necessary to assign a read-only role. Any valid user that can login will have read-only access.

Assigning a Read-Write User

Use the **nacm groups group** command to assign the user to the "policy-admin" group.

For example, nacm groups group policy-admin user-name admin

DRA Central GUI

Purpose

DRA Central is the primary web-based client interface for the configuration and operational control of the CPS vDRA.

URL and Port

https://<*master or control ip*>/central/dra/

Protocol

HTTPS

Accounts and Roles

Assigning a Read-Only User

Use the **nacm groups group** command to assign the user to the "policy-ro" group.

Assigning a Read-Write User

Use the **nacm groups group** command to assign the user to the "policy-admin" group.

For example: `nacm groups group policy-admin user-name admin`

SVN Interface

Apache ™ Subversion (SVN) is the versioning and revision control system used within CPS. It maintains all the CPS policy configurations and has repositories in which files can be created, updated and deleted. SVN maintains the file difference each time any change is made to a file on the server and for each change it generates a revision number.

In general, most interactions with SVN are performed via Policy Builder.

CLI Access

From a remote machine with the SVN client installed, use the following command to access SVN:

Access all files from the server as follows:

```
svn checkout --username <username> --password <password> <SVN Repository URL> <Local Path>
```

Example:

```
svn checkout --username admin --password admin https://<master ip or control ip>/repos/
```

If <*Local Path*>is not provided, files are checked out to the current directory.

Check-in the changed files to the server as follows:

```
svn commit --username <username> --password <password> <Local Path> -m "modified config"
```

Example:

```
svn commit --username broadhop --password broadhop /root/configuration -m "modified config"
```

Update local copy to latest from SVN:

```
svn update <Local Path>
```

Example:

```
svn update /root/configuration/
```

Check current revision of files:

```
svn info <Local Path>
```

Example:

```
svn info /root/configuration/
```

Use **svn --help** for a list of other commands.

Protocol

HTTPS

URL and Port

https://<*master or control ip*>/repos/

Accounts and Roles

Assigning a Read-Only User

It is not necessary to assign a read-only role. Any valid user that can login will have read-only access.

Assigning a Read-Write User

Use the **nacm groups group** command to assign the user to the "policy-admin" group.

For example, nacm groups group policy-admin user-name admin

Multi-user Policy Builder

Multiple users can be logged into Policy Builder at the same time.

In the event that two users attempt to make changes on same screen and one user saves their changes to the client repository, the other user may receive errors. In such cases the user must return to the login page, revert the configuration, and repeat their changes.

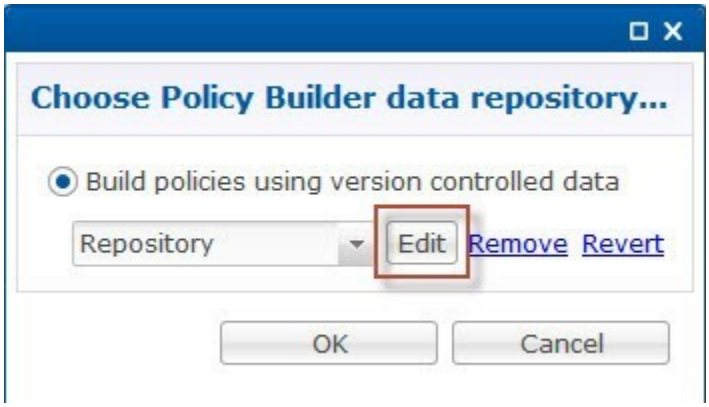
Revert Configuration

You can revert the configuration if changes since the last publish/save to client repository are not wanted.

This can also be necessary in the case of a ' syn conflict ' error where both pcrfclient01 and pcrfclient02 are in use at the same time by different users and publish/save to client repository changes to the same file. The effect of reverting changes is that all changes since the publish/save to client repository will be undone.

Step 1 On the Policy Builder login screen, verify the user for which changes need to be reverted is correct. This can be done by clicking **Edit** and verifying that the Username and Password fields are correct.

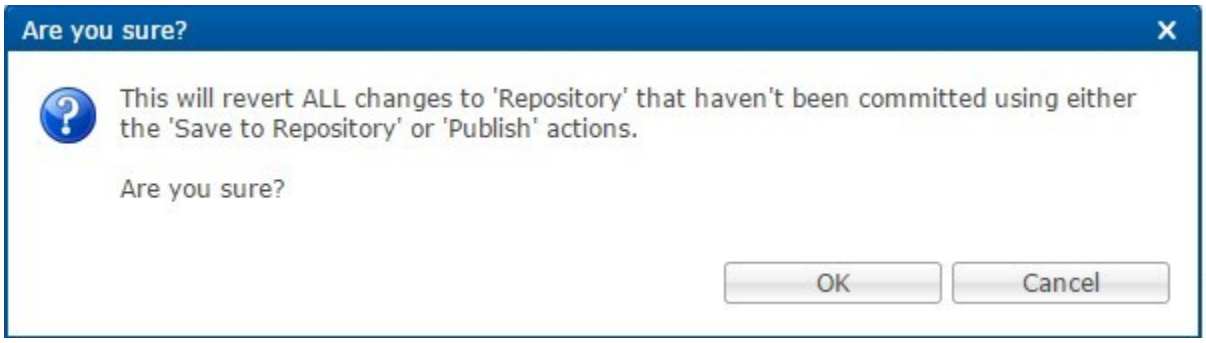
Figure 5: Verifying User



Step 2 Click **Revert**.

The following confirmation dialog opens.

Figure 6: Revert Confirmation Message



Step 3 Click **OK** to revert back to the earlier configuration. The following dialog confirms that the changes are reverted successfully.

Figure 7: Success Confirmation Message

Publishing Data

This section describes publishing Cisco Policy Builder data to the Cisco Policy Server. Publishing data occurs in the Cisco Policy Builder client interface, but affects the Cisco Policy Server.

Cisco Policy Builder manages data stored in two areas:

- The Client Repository stores data captured from the Policy Builder GUI in Subversion. This is a place where trial configurations can be developed and saved without affecting the operation of the Cisco Policy Builder server data.

  The default URL is http://svn/repos/configuration.

- The Server Repository is where a copy of the client repository is created/updated and where the CPS picks up changes. This is done on Publish from Policy Builder.

  The default URL is http://svn/repos/run.

CRD APIs

You can use Custom Reference Data (CRD) APIs to query, create, delete, and update CRD table data without the need to utilize the Control Center interface. The CRD APIs are available via a REST interface.

Limitations

These APIs allow maintenance of the actual data rows in the table. They do not allow the creation of new tables or the addition of new columns. Table creation and changes to the table structure must be completed via the Policy Builder application.

All table names should be in lowercase alphanumeric to utilize these APIs. Spaces and special characters are not allowed in the table name.

- Table names containing uppercase characters will return code 400 Bad Request.
- Spaces in the name are not allowed and will be flagged as an error in Policy Builder.
- Special characters even when escaped or encoded in ASCII throw errors with the APIs and should not be used.

Setup Requirements

Policy Builder

Step 1 Log in to the Policy Builder.

Step 2 Select **Reference Data** tab.

Step 3 Select **Systems** from the left pane.

Step 4 Select and expand your system name.

Step 5 Select **Plugin Configurations** (or a sub cluster or instance), a Custom Reference Data Configuration plugin configuration is defined.

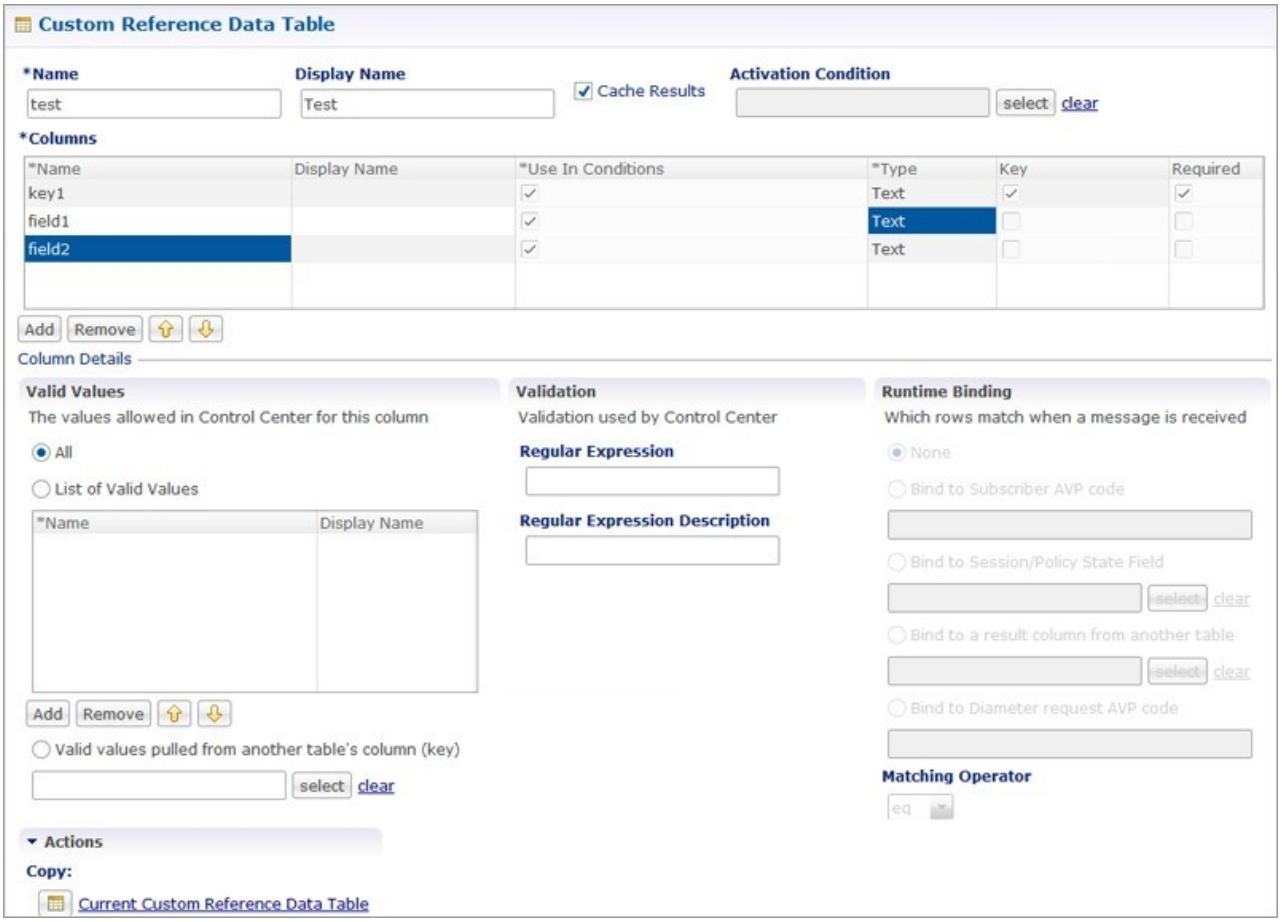
The following parameters can be configured under **Custom Reference Data Configuration**:

Table 2: Custom Reference Data Configuration Parameters

Parameter	Description
Primary Database IP Address	IP address of the primary sessionmgr database. This should remain the default of mongo-admin-a.
Secondary Database IP Address	Optional, this field is the IP address of a secondary, backup, or failover sessionmgr database. This should remain the default of mongo-admin-b.
Database Port	Port number of the sessionmgr. It should be the same for both the primary and secondary databases.
Db Read Preference	Read preference describes how sessionmgr clients route read operations to members of a replica set. You can select from the following drop-down list: • Primary: Default mode. All operations read from the current replica set primary. • PrimaryPreferred: In most situations, operations read from the primary but if it is unavailable, operations read from secondary members. • Secondary: All operations read from the secondary members of the replica set. • SecondaryPreferred: In most situations, operations read from secondary members but if no secondary members are available, operations read from the primary. For more information, refer to http://docs.mongodb.org/manual/core/read-preference/.
Connection Per Host	Number of connections that are allowed per database host. Default value is 100.

Step 6 In **Reference Data** tab > **Custom ReferenceData Tables**, at least one Custom Reference Data Table must be defined.

Figure 8: Custom Reference Data Configuration



The following parameters can be configured under Custom Reference Data Table:

Table 3: Custom Reference Data Table Parameters

Field	Description
Name	Name of the table that will be stored in the database. It should start with alphanumeric characters, should be lowercase or uppercase but not mixed case, and should not start with numbers, no special characters are allowed, use "_" to separate words. For example, logical_apn = GOOD, logicalAPN = BAD, no_spaces.
Display Name	Name of the table that will be displayed in Control Center.
Cache Results	Indicates if the tables should be cached in memory and should be checked for production.
Activation Condition	Custom Reference Data Trigger that needs to be true before evaluating this table. It can be used to create multiple tables with the same data depending on conditions or to improve performance if tables do not need to be evaluated based on initial conditions.
Best Match	When enabled, it allows '*' to be used in the values of the data and the best matching row is returned.

Field	Description
Evaluation Order	Indicates the order the tables within the search table group should be evaluated. Starting with 0 and increasing.
Columns	
Name	Name of the column in the database.
Display Name	More readable display name.
Use In Conditions	Represents the availability of the row for conditions in Policies or Use Case Templates. There is a performance cost to having these enabled, so it is recommended to disable unless they are required.
Type	Determines the values in the control centre as described below: • Text: Value can be any character. For example, example123!. • Number: Value should be a whole number. For example, 1234. • Decimal: Value can be any number. For example, 1.234. • True/False: Value can be true or false. For example, true. • Date: Value should be a date without time component. For example, May 17th 2020. • DateTime: Value should be a date and time. For example, May 17th, 2020 5:00pm.
Key	Indicates that this column is all or part of the key for the table that makes this row unique. By default, a key is required. Keys also are allowed to set the Runtime Binding fields to populate this data from the current message/session. Typically, keys are bound to data from the current session (APN, RAT Type) and other values are derived from them. Keys can also be set to a value derived from another custom reference data table.
Required	Indicates whether this field will be marked required in Control Center. A key is always required.
Column Details	
Valid Values	
All	All the values of the type selected by the user.
List of Valid	A list of name/display name pairs that will be used to create the list. Valid values can also contain a name which will be the actual value of the column and a display value which allows the Control Center to display use name.
Name	The name of the column in the database.
Display Name	Readable display name.
Validation	

Field	Description
Regular Expression	The Java regular expression that will be run on the proposed new cell value to validate it.
Regular Expression Description	A message to the user indicating what the regular expression is trying to check.
Runtime Binding	Runtime binding is how key column data gets filled out (bound) from data in the current session. There are multiple ways to bind this data and it is also possible to set an operator to define what should match (equals, less than, etc).
None	
Bind to Subscriber AVP	This pulls the value from an AVP on the subscriber. It will also pull values from a session AVP or a Policy Derived AVP.
Bind to Session/Policy State	This pulls the value from a Policy State Data Retriever which knows how to retrieve a single value for a session.
Bind to a result column from another table	This allows the key to be filled out from a columns value from another table. This allows 'normalizing' the table structure and not having on giant table with a lot of duplicated values.
Bind to Diameter request AVP code	This allows the key be filled out from an AVP on the diameter request.
Matching Operator	This allows the row to be 'matched' in other ways than having the value be 'equals'. Default value is equals. • eq: Equal • ne: Not Equal • gt: Greater than • gte: Greater than or equal • lt: Less than • lte: Less than or equal

Architecture

MongoDB Caching

The MongoDB database containing the CRD tables and the data is located in the MongoDB instance specified in the CRD plugin configuration.

The database is named cust_ref_data.

Two system collections exist in that database and do not actually contain CRD data:

- system.indexes - It is used by MongoDB. These are indices set on the database.
- crdversion - It contains a document indicating the version of all the CRD tables you have defined. The version field increments by one every time you make a change or add data to any of the CRD tables.

A collection is created for each CRD table defined in Policy Builder.

- This collection contains a document for each row you define in the CRD table.
- Each document contains a field for each column you define in the CRD table.
- The field contains the value specified for the column for that row in the table.
- Additionally, there is a _id field which contains the internal key used by MongoDB and _version which is used by CPS to provide optimistic locking protection, essentially to avoid two threads overwriting the other's update, on the document.

Setting the Cache Results to true (checked) is the default and recommended settings in most cases as it yields the best performance. Use of the cached copy also removes the dependency on the availability of the CRD database. So if there is an outage or performance issue the policy decisions utilizing the CRD data will not be impacted.

The cached copy of the table is refreshed on CPS restart and whenever the API writes a change to the CRD table, otherwise the cached copy is used and the database is not accessed.

API Endpoints And Examples

The URL used to access the CRD API is located at https://<masterip or control ip>/custrefdata/<tablename>/_<operation>

Query API

Purpose

Returns all rows currently defined in the specified table.

HTTP Operation Type

GET

Example URL

https://<master or control ip>:8443/custrefdata/test/_query

Example URL with Filtering

https://<master or control ip>:8443/custrefdata/test/_query?key1=Platinum

Payload

None, although parameters can be specified on the URL for filtering.

Response

Success returns code 200 Ok; XML indicating rows defined is returned. If there are no records in the table, 200 Ok is returned with empty rows in it.

If the table does not exist, code 400 Bad Request is returned.

Example Response without Filtering

```
<rows>
  <row>
    <field code=”field1” value=”1004”/>
    <field code=”field2” value=”testee”/>
    <field code=”key1” value=”Platinum”/>
  </row>
  <row>
    <field code=”field1” value=”1004”/>
    <field code=”field2” value=”testee”/>
    <field code=”key1” value=”Platinum99”/>
  </row>
  <row>
    <field code=”field1” value=”field1example1”/>
    <field code=”field2” value=”field2example1”/>
    <field code=”key1” value=”key1example1”/>
  </row>
  <row>
    <field code=”field1” value=”field1example2”/>
    <field code=”field2” value=”field2example2”/>
    <field code=”key1” value=”key1example2”/>
  </row>
</rows>
```

Example Response with Filtering

```
<rows>
<rows>
  <row>
    <field code=”field1” value=”1004”/>
    <field code=”field2” value=”testee”/>
    <field code=”key1” value=”Platinum”/>
  </row>
</rows>
```

The response returns keys with the tag “field code”. If you want to use the output of Query as input to one of the other APIs, the tag needs to be changed to “key code”. Currently using “field code” for a key returns code 404 Bad Request and a java.lang.NullPointerException.

Create API

Purpose

Create a new row in the specified table.

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/test/_create

Example Payload

```
<row>
  <key code=”key1” value=”Platinum”/>
  <field code=”field1” value=”1004”/>
  <field code=”field2” value=”testee”/>
</row>
```

Response

Success returns code 200 Ok; no data is returned. The key cannot already exist for another row; submission of a duplicate key returns code 400 Bad Request.

If creating a row fails, API returns 400 Bad Request.

Note Create API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Create operation is not allowed for subversion table

Update API

Purpose

Updates the row indicated by the key code in the table with the values specified for the field codes.

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/test/_update

Example Payload

```
<row>
  <key code="key1" value="Platinum"/>
  <field code="field1" value="1005"/>
  <field code="field2" value="tester"/>
</row>
```

Response

Success returns code 200 Ok; no data is returned. The key cannot be changed. Any attempt to change the key returns code 404 Not Found.

If updating a row fails, API returns 400 Bad Request.

Note Update API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Update operation is not allowed for subversion table

Delete API

Purpose

Removes the row indicated by the key code from the table.

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/test/_delete

Example Payload

```
<row>
<key code="key1" value="Platinum"/>"/>
</row>
```

Response

Success returns code 200 Ok; no data is returned. If the row to delete does not exist, code 404 Not Found is returned.

If deleting a row fails, API returns 400 Bad Request.

Note Delete API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Delete operation is not allowed for subversion table

Data Comparison API

Purpose

Determines whether the same CRD table data content is being used at different data centers.

The following three optional parameters can be provided to the API:

- tableName: Returns the checksum of a specified CRD table `tableName` indicating if there is any change in the specified table. If the value returned is same on different servers, it means there is no change in the configuration and content of that table.

- includeCrdversion: Total database checksum contains combination of checksum of all CRD tables configured in Policy Builder. If this parameter is passed as true in API, then total database checksum includes the checksum of "crdversion" table. Default value is false.
- orderSensitive: Calculates checksum of the table by utilizing the order of the CRD table content. By default, it does not sort the row checksums of the table and returns order sensitive checksum of every CRD table. Default value is true.

custrefdata/_checksum

Database level Checksum API returns checksum details for all the CRD tables and the database. If the value returned is same on different servers, there will be no change in the configuration and content of any CRD table configured in Policy Builder.

HTTP Operation Type

GET

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/_checksum

Response

```
<response>
    <checksum><all-tables-checksum></checksum>
    <tables>
        <table name="<table-1-name>" checksum="<checksum-of-table-1>"/>
        <table name="<table-2-name>" checksum="<checksum-of-table-2>"/>

        <table name="<table-n-name>" checksum="<checksum-of-table-n>"/>
   </tables>
</response>
```

/custrefdata/_checksum?tableName=<user-provided-table-name>

Table specific Checksum API returns the checksum details for the specific CRD table. If the value returned is same on different servers, there will be no change in the configuration and content of that table.

HTTP Operation Type

GET

Example Endpoint URL

https://<master or control ip>:8443 /custrefdata/_checksum?tableName=<user-provided-table-name>

Response

```
<response>
    <tables>
        <table name="<user-provided-table-name>" checksum="<checksum-of-specified-table"/>
    </tables>
</response>
```

Note Table specific Checksum API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Checksum operation is not allowed for subversion table

Table Drop API

Purpose

Drops custom reference table from MongoDB to avoid multiple stale tables in the system.

The Table Drop API is used in the following scenarios:

- If a CRD table does not exist in Policy Builder but exists in the database, the API can be used to delete the table from the database.
- If a CRD table exists in Policy Builder and database, the API cannot delete the table from the database. If this is attempted the API will return an error: “Not permitted to drop this table as it exists in Policy Builder”.
- If a CRD table does not exist in Policy Builder and database, the API will also return an error `No table found:<tablename>`.

/custrefdata/<table_name>/_drop

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/<table_name>/_drop

Note Drop API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Drop operation is not allowed for subversion table

Export API

Purpose

Exports single and multiple CRD table and its data.

/custrefdata/_export?tableName=<table_name>

Exports single CRD table and its data.

Returns an archived file containing csv file with information of specified CRD table `table_name`.

HTTP Operation Type

GET

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/_export?tableName=<table_name>

/custrefdata/_export

Exports all CRD tables and its data.

Returns an archived file containing csv file with information for each CRD Table.

HTTP Operation Type

GET

Example Endpoint URL

https://<master or control ip>:8443 /custrefdata/_export

Note Export API does not support Svn CRD tables and displays the following warning message in the Response Header "Export-Warning":

Datasource for tables [table1, table2,...] is subversion. Response will not contain data for these tables and skipped SVN CRD tables to be a part of archive.

Import API

Purpose

Imports CRD table and its data.

It takes an archived file as an input which contains one or more csv files containing CRD tables information.

Note If you try to import multiple CRD tables during traffic it may have call flow impact. It is recommended to import multiple CRD tables during Maintenance Window (MW).

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/_import

https://<lbvip01>:8443/custrefdata/_import?batchOperation=true

https://<lbvip01>:8443/custrefdata/_import?batchOperation=false&duplicateValidation=true

Note

1. The "batchOperation" flag is used to insert CRD data in the batch. The default value is true and if you do not provide it in the request parameter the default value is taken.
2. The "duplicateValidation" flag is used to validate or invalidate duplicate data in the archive. The default value is true and if you do not provide it in the request parameter the default value is taken which means it will always validate your data as duplicate.
3. If "batchOperation" is true, the API will validate your data as duplicate data regardless of the value provided for "duplicateValidation".

Note Import API supports SVN CRD table operations in the following scenarios:

- If the archive contains only mongodb tables, success message is displayed in the response.
- If the archive contains only SVN tables, success and warning messages are displayed in the response.
- If the archive contains both mongodb and SVN tables, success and warning messages are displayed in the response.

Snapshot POST API

Purpose

Creates a snapshot of the CRD tables on the system. The created snapshot will contain CRD table data, policy configuration and checksum information for all CRD tables.

/custrefdata/_snapshot?userId=<user_id>&userComments=<user_comments>

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/_snapshot?userId=<user_id>&userComments=<user_comments>

Optional Parameters

userComments

Note Snapshot POST API does not support export of the contents of Svn CRD tables. The API returns the following warning message if there are any Svn CRD tables present while creating snapshot:

Datasource for tables [table_1, table_2…] is subversion. Data for these tables will not come from database (mongodb)

Snapshot GET API

Purpose

Enables you to get the list of all valid snapshots in the system.

The following information is available in the list of snapshots:

- Snapshot name
- Snapshot path
- Date and time of snapshot creation
- User comments provided on creation of the snapshot
- Checksum information of CRD tables
- Policy configuration SVN version number

/custrefdata/_snapshot

HTTP Operation Type

GET

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/_snapshot

Example Response

```
<snapshots>
    <snapshot>
        <name><date-and-time>_<user-id></name>
        <snapshotPath>/var/broadhop/snapshot/20160620011825306_qns</snapshotPath>
        <creationDateAndTime>20/06/2016 01:18:25:306</creationDateAndTime>
        <comments>snapshot-1 june</comments>
        <policyVersion>903</policyVersion>
        <checksum checksum="60f51dfd4cd4554910da44a776c66db1">
            <table name=<table-name-1> checksum="<table-checksum-1>"/>
                …
            <table name=<table-name-n> checksum="<table-checksum-n>"/>
        </checksum>
    </snapshot>
    <snapshot>
                …
```

```
    </snapshot>
</snapshots>
```

Note Snapshot GET API does not return checksum information of Svn CRD tables as they are not part of created snapshots.

Revert API

Purpose

Enables you to revert the CRD data to a specific snapshot. If the specific snapshot name is not provided, the API will revert to the latest snapshot.

/custrefdata/_revert?snapshotName=<snapshot_name>

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/_revert?snapshotName=<snapshot_name>

Optional Parameter

snapshotName

Note Revert API does not support reverting of CRD data for Svn CRD tables. For Svn CRD table, it clears the mongodb table and displays the following warning message:

Datasource for tables [table_1, table_2…] is subversion. Data for these tables will be reverted using svn datasource not from database (mongodb)

Admin Disable API

Purpose

Create multiple rows in the Peer Admin Disabled List CRD table in a single operation.

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/peer_admin_disabled_list/_createRows

Note Once `https://<master or control ip>:8443/custrefdata/peer_admin_disabled_list/_createRows` API is complete, you need to run `/dra/api/localActivePeerEndpoints/disconnect` to disconnect the active peer endpoint.

Note In Active Peer Endpoints GUI, after admin disable of active peer, if peer's Admin State gets changed from Enabled to Disabled but still it is shown under Active Peer Endpoints, then peer has to be disconnected by using the disconnect action.

Example Payload

```
{
  "rows": [
    {
      "fields": [
        {
          "code": "origin_host",
          "value": "value_for_origin_host"
        },
        {
          "code": "origin_realm",
          "value": "value_for_origin_realm"
        },
        {
          "code": "admin_disable_time",
          "value": "time_in_this_format_only_1/9/2021 10:48:56"
        }
      ],
      "keys": [

      ]
    },
    {
      "fields": [
        {
          "code": "origin_host",
          "value": "value_for_origin_host"
        },
        {
          "code": "origin_realm",
          "value": "value_for_origin_realm"
        },
        {
          "code": "admin_disable_time",
          "value": "time_in_this_format_only_1/9/2021 10:48:56"
        }
      ],
      "keys": [

      ]
    }
  ]
}
```

Response

Success returns code 200 Ok; no data is returned. If creating a row fails, API returns 400 Bad Request.

Note Create rows API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Create operation is not allowed for subversion table

Admin Enable API

Purpose

Removes multiple rows indicated by the key code from the table in a single operation.

HTTP Operation Type

POST

Example Endpoint URL

https://<master or control ip>:8443/custrefdata/peer_admin_disabled_list/_deleteRows

Example Payload

```
{
  "rows": [
    {
      "fields": [
        {
          "code": "origin_host",
          "value": "value_for_origin_host"
        },
        {
          "code": "origin_realm",
          "value": "value_for_origin_realm"
        }
      ],
      "keys": [
        {
          "code": "origin_host",
          "value": "value_for_origin_host"
        },
        {
          "code": "origin_realm",
          "value": "value_for_origin_realm"
        }
      ]
    },
    {
      "fields": [
        {
          "code": "origin_host",
          "value": "value_for_origin_host"
        },
        {
```

```
            "code": "origin_realm",
            "value": "value_for_origin_realm"
          }
        ],
        "keys": [
          {
            "code": "origin_host",
            "value": "value_for_origin_host"
          },
          {
            "code": "origin_realm",
            "value": "value_for_origin_realm"
          }
        ]
      }
    ]
}
```

Response

Success returns code 200 Ok; no data is returned. If deleting a row fails, API returns 400 Bad Request.

Note Delete rows API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Delete operation is not allowed for subversion table

Tips for Usage

The Query API is a GET operation which is the default operation that occurs when entering a URL into a typical web browser.

The POST operations, Create, Update, and Delete, require the use of a REST client so that the payload and content type can be specified in addition to the URL. REST clients are available for most web browsers as plug-ins or as part of web service tools, such as SoapUI. The content type when using these clients should be specified as application/xml or the equivalent in the chosen tool.

View Logs

You can view the API logs with the following commands:

- monitor log application – tail the current application log
- monitor log engine – tail the current engine log
- monitor log container – tail a specific container log
- show log application - view the current application log
- show log engine – view the current engine log

Logging Support Using Journald

To monitor and view logs, journald system service has been added that collects and stores logging data. It creates and maintains structured, indexed journals based on logging information received from a variety of sources. The following is a sample of CLI commands:

- `monitor log application` - This command is used to tail the current Policy Server (qns) log.
- `monitor log engine` - This command is used to tail the current Policy Server (qns) engine log
- `monitor log container <container id>` - This command is used to tail the container logs.
- `show log application` - This command opens the consolidated logs.
- `show log engine` - This command is used to open the consolidate engine logs using Linux 'less' command.

For further log access, you need to connect to the OpenStack control node and from there to respective master or control node. For example, to connect to master/control nodes use the following command:

```
ssh -i cps.pem cps@IPAddress
```

where, *IPAddress* is the IP address of the master or control node.

To access the logs once you are connected to control node, use the following command:

```
docker logs container-id
```

For example, use `docker logs mongo-s1` to display all the logs of mongo-s1 container.

Retaining journalctl Logs in DRA

Note This feature has not been validated for all customer deployment scenarios. Please contact your Sales Account team for support.

In vDRA, Docker engine is configured with journald logging driver on every VM. The journald logging driver sends container's logs to journal deamon.

Use the **journalctl** command, through journal API, or use the **docker logs** command to retrieve the log entries.

As part of the logging enhancements, vDRA supports retaining of journalctl logs for longer duration around 10 days on all VMs. This helps in debugging any issues even though journal logs gets rolled over early.

All the logs are captured through automated cron job at daily basis on nonpeak time and cronjob timings are configurable through cron job file. The collected logs are stored under `/data/journal-logs` directory on each VM and also stored at remote server. You can configure the size of the logs folder and days of retention in the configuration file.

On every VM, log collection happens based on disk size of the `/data/journal-logs` folder, Default `/data/journal-logs` directory size is 10GB. If the `/data/journal-logs` directory size is less than 10GB it will collect the logs and it will copy to the Control VM and remote server, If the `/data/journal-logs` directory size exceeds to 10 GB , journal.sh script deletes files beyond 2 days to free up the disk space on the VM. This parameter is also configurable from cps-journal.conf file.

You can configure the retention days and size of log storage folder on `/etc/cps/cps-journal.conf file`. And copying journal logs to Control VM works with static and Virtual VIP IP.

While copying the journal logs to a control VM, journal.sh script checks the / disk usage on control VM. If the disk size is less than 60 % it copies files to the control VM, otherwise it won't copy and these log files are stored on same VM based on the retention period. This disk usage value for Control VM is configuration through cps-journal.conf file.

For the CPU usage optimization, this script is limited to execute with only 50 % of the system CPU.

Prerequisites

Before you begin:

1. Setup DRA/Binding VNF.
2. Ensure that *cps.pem* file is copied to all the VMs.
3. Configure the remote server as PEM key based authentication.
4. Control VM should be reachable to remote server.

Journal Configuration

Modify the custom general configuration file:

```
cat /etc/cps/cps-journal.conf
```

You can configure the following parameters.

Table 4: Journal Configuration File Parameters

Field	Description
retention_days	Specify the number of retention days to store log files. Example: retention_days=10
logfolder_size	Specify a size of the log storage directory. Memory value must be entered in KB format. Example: 10485760
clean_all	Specify the number of days for which the logs are saved after clean up.
DRA_USER	Displays the DRA user as CPS.
CONTROL_IP	Specify the Control IP. **Note** Control IP should be reachable to all internal VMs and remote server. Example: Control-1 IP
DESTINATION	Specify a centralized log storage path on the control-1 VM.

Field	Description
PEM_KEY	Specify an absolute path of the SSH key PEM file location. Example: PEM_KEY=/home/cps/cps.pem
DISK_SIZE	Specify the maximum disk usage percentage on Control VM for /dir (directory).
remote_server	Specify a remote server IP address.
remote_destination_path	Specify the Journal Logs storage path on remote server. **Note** Use different destination paths for multiple sites and setups. Example: DRA Site-1: remote_destination_path=/home/cps/dra-site-1 DRA Site-2: remote_destination_path=/home/cps/dra-site-2 Binding Site-1: remote_destination_path=/home/cps/binding-site-1 Binding Site-2: remote_destination_path=/home/cps/binding-site-2
remote_user	Specify a remoter server user to perform the operation. Example: remote_user=CPS
remote_pem_key	Remote a server user PEM key file absolute path.

Post Configuration and Validation Process

After all the configurations are set, perform the following steps:

1. Check the cron job scheduled for the root user. Sample configuration is shown.

```
#crontab -l  -u root
0 8 * * * cputool -c 50 bash /opt/custom-scripts/journal.sh
30 8 * * * cputool -c 50 bash /opt/custom-scripts/journal_scp.sh
Default cron job is scheduled at 8:00 AM UTC,
```

2. Verify the collected logs that are present under `/data/journal-logs` directory on each VM after the completion of cron job and check the remote server.

   Sample Log file format:.

```
journal-2021-06-06-09:00:01-dra1-sys04-master-0.log.gz // Log file created for VM with
hostname & timestamp.
journal-history.log // history of journals execution and file copying status
```

Bulk Provisioning of Records in SLF Database

CPS vDRA provides APIs for bulk provisioning of subscriber records in the SLF database.

You can use the CSV file to provision create and update of bulk subscriber records using SLF API. You can also check the status of the upload using the API.

Note SLF bulk provisioning generates high number of database write operations in a short duration of time. To spread out the operations over a period of time and mitigate the performance issue, configure the transactions per second (TPS) for SLF provisioning in Policy Builder.

For more information, see the *CPS vDRA Configuration Guide*.

CSV File

The CSV file format is used to bulk provision the subscriber records in SLF database. The Actions column in the CSV file determines whether the record is for creation, updation, or deletion.

You can use # in the beginning of the line to indicate comments in the CSV file. The line is ignored when the file is processed.

Table 5: CSV File Format

Column	Description
Action	The action to be performed on the subscriber record. • Create - creates subscriber record if it does not exist. • Put – creates the subscriber record, if it does not exist; if subscriber record already exists, updates the subscriber record. • Delete – deletes the subscriber record, if it exists.
Subscriber Id	The subscriber ID of the subscriber.
IMSI	The IMSI of the subscriber. If the same subscriber has multiple IMSI, then add multiple IMSI columns for the subscriber.
MSISDN	The MSISDN of the subscriber. If the same subscriber has multiple MSISDN, then add multiple MSISDN columns for the subscriber.
Destination:<Tag>	The destinations of the subscriber. To provision multiple destinations, add column name/header with prefix “Destination:” and suffix it with the tag, for example: Destination:HSS, Destination:MME, Destination:PCRF, etc

Sample CSV File

```
Action, Subscriber Id, IMSI, IMSI, MSISDN, MSISDN, Destination:MME, Destination:HSS
Put,    1001,         34101, 34102, 91001, 91002,  MME1,           HSS1
```

```
Put,    1001,         34101,      ,  91005,     ,  MME2,           HSS2
Delete, 1010, , , , , ,
```

Bulk Upload API

Schedules the SLF bulk subscribers provisioning task. Bulk Upload API takes the input as csv file and schedules the job to execute in the background.

Request

Method: POST

URI: /dra/slfapi/subscriber/bulkUpload

Header: Content-Type: multipart/form-data

Body: CSV File

Request Example

```
HTTP POST /dra/slfapi/subscriber/bulkUpload
```

Response Example

```
HTTP STATUS: 202 (Accepted)
{
  "success": {
    "code": 1,
    "message": "Request accepted, slf bulk upload task is scheduled for execution"
  }
}
```

Example of Curl Command

```
curl -X POST --progress-bar -H "Content-Type: multipart/form-data"
-H "Content-Type: application/json" \ -F "file=@create_subscribers.csv"
https://<MasterIP>/dra/slfapi/subscriber/bulkUpload --insecure
-u admin:admin
```

The file named create_subscribers.csv must be created before running this command.

Bulk Upload Status

Returns the list of bulk upload status of the bulk provisioning sorted by the latest first. Latest 10 statuses would be saved in the system for reference, old status will automatically get purged.

The following table describes the fields in the Bulk Upload Status:

Table 6: Bulk Upload Status

Field	Description
fileName	The name of csv file uploaded.
startTime	The time when task was scheduled.
endTime	The time when task was finished

Field	Description
approxEndTime	The future time when task is expected to be finished
status	The status of the task Status can be one of these statues (scheduled, in-progress, complete, failed)
statusMessage	The detailed status of the task
numberOfTotalSubscriber	Total number of subscriber in csv file
numberOfPending	The number of subscriber pending for execution
numberOfComplete	The number of subscriber, whose execution is finished
numberOfSuccess	The number of subscriber provisioned successfully.
numberOfFailure	The number of subscriber failed in provisioning.
failedSubscriber	This field contains the failure reason for each failed subscriber. This is a map, with key as error code and value as the list of failed subscribers.

Request

Method: GET

URI: /dra/slfapi/subscriber/bulkUploadStatus

Request Example

```
HTTP GET /dra/slfapi/subscriber/bulkUploadStatus
```

Response Example

```
HTTP STATUS: 200
 [{
 "approxEndTime": "08-17-2017 13:31:59",
        "failedSubscriber": {
            "1001": [
                "1000000000",
                "1000000001",
                "1000000002"
            ]
         },
         "fileName": "create_subscribers_1k.csv",
         "numberOfComplete": 700,
         "numberOfFailure": 3,
         "numberOfPending": 300,
         "numberOfSuccess": 697,
         "numberOfTotalSubscriber": 1000,
         "startTime": "08-17-2017 13:30:16",
         "status": "complete",
         "statusMessage": "Slf bulk upload task execution is in progress"
    },
   {
    "endTime": "08-18-2017 12:41:27",
```

```
        "failedSubscriber": {},
     "fileName": "create_subscribers_10.csv",
        "numberOfComplete": 10,
        "numberOfFailure": 0,
        "numberOfPending": 0,
     "numberOfSuccess": 10,
        "numberOfTotalSubscriber": 10,
        "startTime": "08-18-2017 12:41:27",
     "status": "complete",
        "statusMessage": "Slf bulk upload task is completed"
    }]
```


Example of Curl Command

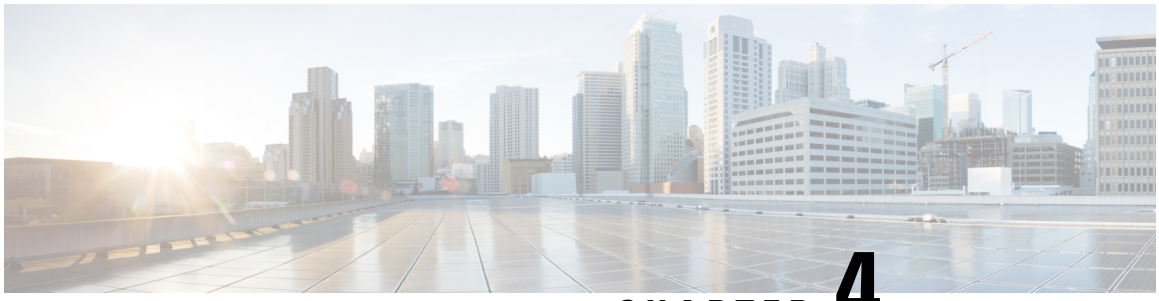
```
curl -X GET --progress-bar -H "Content-Type:
application/json" \https://<MasterIP>/dra/slfapi/subscriber/bulkUploadStatus
 --insecure -u admin:admin
```

vDRA Peer API

The vDRA Peer API provides a REST API interface for the following functions:

- view active and inactive peer endpoints - local and remote
- view peer details for each host and/or peer key
- peer status logs

For more information about the Peer API, see the API RAML at: https://<master ip>/central/dra/#!/dra/docs/api



CHAPTER 4

Method to Ship Docker, Journalctl, and QNS Logs to External EFK Stack

Feature Description

vDRA supports a unified method to forward all required logs such as journalctl,consolidated-qns logs, mongo logs to elasticsearch. You can have a consolidated view of all the logs with Elasticsearch Fluentbit Kibana (EFK) stack. In addition, using Kibana you can visualize and filter required logs for analysis.

Elasticsearch is an open source, full-text search and analytics engine, based on the Apache Lucene search engine. Elasticsearch indexes and stores the data.

Fluent Bit is an open source Log Processor and Forwarder which allows you to collect any data like metrics and logs from different sources, enrich them with filters and send them to multiple destinations. Fluent-Bit takes care of data collection and processing.

Kibana is a visualization layer that works on top of Elasticsearch, providing users with the ability to analyze and visualize the data. Kibana provides a user interface for querying the data and visualizing.

Logs are collected within each VM and same are forwarded to one of the OAM VMs. The logs are then forwarded from the corresponding OAM VM to external servers. Logs can be filtered based on keywords before it is sent to the elastic search server. For more information about CLI Command configurations, see the *CLI Commands* chapter in the *CPS vDRA Operations Guide*.

Configuration to Fetch Journalctl

Logs from journald are available to fluent-bit through the input plugin Systemd. With this plugin, Journalctl logs are available with required journald key value pairs. The configuration file is available at every VM at `/etc/td-agent-bit/td-agent-bit.conf`. For example:

```
[INPUT]
      Name systemd
      Tag host.*
      Systemd_Filter _SYSTEMD_UNIT=docker.service  <optional>
```

Limitations

Currently the journalctl logs are available as single line entries in fluent-bit. Multiline parsing is not available for trace errors in logs.

Configuration to fetch the consolidated-qns logs and mongo logs

consolidated-qns logs and mongo logs are part of the docker logs mounted to the host. These logs are available as part of the tail plugin, which is then forwarded to the required OAM vms.

consolidated-qns logs:

Configuration files for fetching the consolidated-qns logs are available at `/etc/td-agent-bit/td-agent-bit.conf` for each VMs control in DRA vnf. For example:

```
[INPUT]
           name tail
           path /data/cc-monitor-s102/var/log/broadhop/consolidated-qns.log
           tag consolidated_qns_logs
```

mongo logs:

Configuration files for fetching the mongo logs are available at `/etc/td-agent-bit/td-agent-bit.conf` for each VM in Database Base (DB) vnf. For example:

```
[INPUT]
           name tail
           path /data/mongod-node/db/mongo-*.log
           tag logs_on_db
```

Configuration for local Log forwarding

You can enable Log forwarding locally from all servers in DRA to forward the local logs to one of the control vm from where the logs can be extracted to external servers.

Enable Configuration for local forwarding:

```
Configuration file: /etc/td-agent-bit/td-agent-bit.conf
```

Sample Configuration:

```
[OUTPUT]
          name  forward
          match *
          host  <OAM-VIP>
          port  24224
```

Make sure to configure the OAM VIP using CLI commands.

Configuration for Controlling the Interval and Size Forwarding

Use the following configuration for controlling the interval when logs are forwarded and the size that can be forwarded with each batch. The following configuration is available on the OAM vm at `/etc/td-agent-bit/td-agent-bit.conf`

Sample Configuration:

```
[SERVICE]
    # Flush
    # set an interval of seconds before to flush records to a destination
    flush         400
    storage.path  /var/log/flb/
    storage.sync  normal
    storage.checksum   off
    storage.max_chunks_up 80
    storage.backlog.mem_limit 500M
[INPUT]
    name         forward
    listen       0.0.0.0
    port         24224
    storage.type filesystem
```

The “flush” interval decides the interval at which logs are flushed to output.

Configuration to Forward Remote Logs

Enable the Remote log forwarding ithrough elasticsearch plugin of Fluent-bit output configuration. Configurations for elasticsearch configuration with fluent-bit are available at `/etc/td-agent-bit/td-agent-bit.conf` on proxy vm (OAM).

This enables the proxy vm (control vm in this case) to forward all the logs collected to the external server.

Sample Configuration:

```
[OUTPUT]
    name  es
    match *
    host  <172.18.63.228>
    port  <9200>
         index fluent_bit
         HTTP_User      <username>
         HTTP_Passwd   <password>
         Logstash_Format    on
         Retry_Limit        5
```

The host IP (elasticsearch IP), port, username, password are configurable with CLI commands

Password authentication is enabled for external server with the elasticsearch plugin of fluent-bit.

Monitoring Healthcheck of Elasticsearch Server

The ElasticSearch Server (External Server) (elasticsearch server) is monitored to DRA if its reachable to the OAM vms. If there is an unreachability, the alert is trigerred. If there is no configuration provided for external

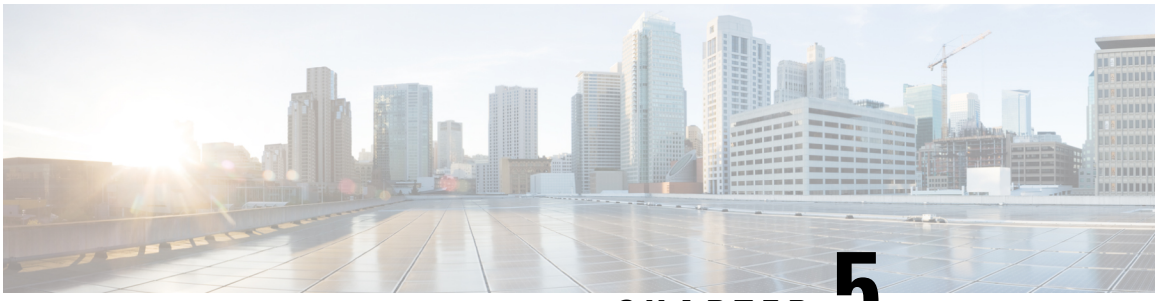
server IP, no alert is observed. Use the **ELASTICSEARCH_NOT_REACHABLE** alert to notify user if the External elasticsearch server is reachable to DRA. If this is not reachable to DRA, an alert is raised.

Configuration for Log Filteration

Use the following sample configuration for filtering the logs before it is forwarded to the external servers.The filter section configured at `/etc/td-agent-bit/td-agent-bit.conf for the OAM vm`:

```
[FILTER]
name grep
match *
regex log aa
```

Configure the "regex" through CLI to apply the pattern and filter logs before it is sent to the external server.



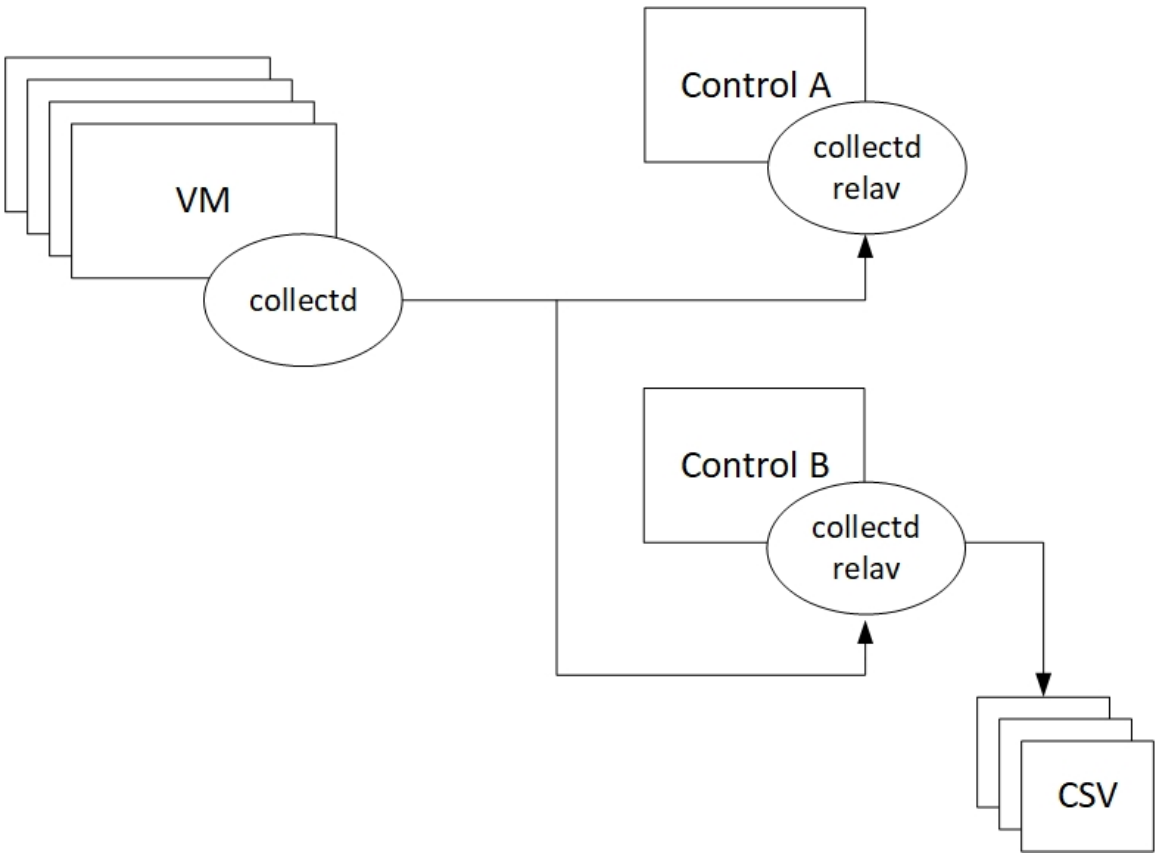
CHAPTER 5

CPS Statistics

Bulk Statistics Overview

Bulk Statistics are the statistics that are gathered over a given time period and written to a set of CSV files. These statistics can be used by external analytic processes and/or network management systems. The architecture of CPS bulk statistic collection is shown in the following illustration.

Figure 9: DRA Bulk Statistic Collection Architecture



The collection utility collectd is used for collecting and storing statistics from each VM. Detailed collectd documentation can be found on http://collectd.org/.

Collectd within CPS is deployed with nodes relaying data using the collectd network plug-in (https://collectd.org/wiki/index.php/Plugin:Network) to the centralized collection nodes on the control-A and control-B virtual machines. The centralized collector writes the collected data to output CSV files.

Note Control A and Control B collect bulk statistics independently. As a result, it is normal to have slight differences between the two files. For example, control-A will generate a file at time t and control-B will generate a file at time t +/- the clock drift between the two machines.

As a best practice, always use the bulk statistics collected from Control-A. Control-B can be used as a backup in the event of failure of control-A.

In the event that Control-A becomes unavailable, statistics will still be gathered on Control-B. Statistics data is not synchronized between Control-A and Control-B, so a gap would exist in the collected statistics while control-A is down.

Note The collectd collection mechanism are separate from the Prometheus / Grafana Monitoring.

CPS Statistics

The list of statistics available in CPS is consolidated in an Excel spreadsheet. After CPS is installed, this spreadsheet can be downloaded from the Bulk Stats link available on below URL:

https://<master ip>/central/dra/#!/dra/docs/stats

Bulk Statistics Collection

By default, CPS outputs a bulk statistics CSV file to the `/var/broadhop/stats/` directory on the control-A and control-B VMs in five-minute intervals.

An scp / sftp daemon running on port 2026 retrieves all statistics within the `/var/broadhop/stats` directory. Only locally defined users within the scheduling application associated to the "bulkstats" or "admin" group are able to retrieve statistics.

You can also retrieve statistics by logging into the virtual machine directly and retrieving the statistics from the `/data/stats` directory.

The default naming standard is `bulk-hostname-YYYY-MM-DD-HH-MI.csv`

These CSV files include all statistics collected from all VMs during the five-minute interval.

Note If a statistic is generated by the system multiple times within the five-minute interval, only the last measured statistics is collected in the CSV file.

The following list is a sample of the file names created in the `/var/broadhop/stats/` directory on the control-A VM:

```
[root@control-1 stats]# pwd
/data/stats-relay-s1/var/broadhop/stats [root@control-A stats]# ls
bulk-control-A-201510131350.csv
bulk-control-A-201510131355.csv
bulk-control-A-201510131400.csv
bulk-control-A-201510131405.csv
bulk-control-A-201510131410.csv
bulk-control-A-201510131415.csv
bulk-control-A-201510131420.csv
bulk-control-A-201510131425.csv
bulk-control-A-201510131430.csv
bulk-control-A-201510131435.csv
bulk-control-A-201510131440.csv
bulk-control-A-201510131445.csv
bulk-control-A-201510131450.csv
bulk-control-A-201510131455.csv
bulk-control-A-201510131500.csv
bulk-control-A-201510131505.csv
bulk-control-A-201510131510.csv
bulk-control-A-201510131515.csv
bulk-control-A-201510131520.csv
bulk-control-A-201510131525.csv
bulk-control-A-201510131530.csv
bulk-control-A-201510131535.csv
bulk-control-A-201510131540.csv
```

```
bulk-control-A-201510131545.csv
bulk-control-A-201510131550.csv
bulk-control-A-201510131555.csv
bulk-control-A-201510131600.csv
bulk-control-A-201510131605.csv
bulk-control-A-201510131610.csv
bulk-control-A-201510131615.csv
bulk-control-A-201510131620.csv
bulk-control-A-201510131625.csv
bulk-control-A-201510131630.csv
```

Retention of CSV Files

CPS retains each bulk statistics CSV file on the control-A/B VM for two days; after which the file is automatically removed.

If you need to preserve these CSV files, you must back up the files or move them to an alternate system.

Diameter Monitoring KPIs

The following table describes CPS KPIs that are useful for monitoring Diameter message traffic.

Note As each deployment is unique, no recommended ranges are provided. Cisco recommends monitoring these KPIs for a period of time (1-3 months) to establish a baseline. Deviations can then be monitored from the baseline values.

Table 7: Diameter Monitoring KPIs

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-I_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-I_2001. qns_stat.total _time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Gx/F	Diameter Round Trip	node[x].messages.e2e _<domain>_[realm_] Gx_CCR-I_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-I_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-I_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/A	Diameter Input Queue	node1.counters. [realm_] Gx_CCR-I.qns_count	Count of messages successfully sent to the policy engine	Policy Server (qns)
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Gx/A	Diameter Input Queue	node1.counters. [realm_] Gx_CCR-U. qns_count	Count of messages successfully sent to the policy engine	Policy Server (qns)
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/A	Diameter Input Queue	node1.counters. [realm_] Gx_CCR-U. qns_count	Count of messages successfully sent to the policy engine	Policy Server (qns)
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-T_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-T_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-T_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-T_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-T_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/A	Diameter Input Queue	node1.counters. [realm_] Gx_CCR-T.qns_count	Count of messages successfully sent to the policy engine	Policy Server (qns)
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_timeout. qns_stat.success	Success timeout count for RAR message	Policy Director
Gx/A	Diameter Input Queue	node1.counters. [realm_] Gx_RAA.qns_count	Count of all messages sent to the policy engine	Policy Server (qns)
Gx/A	Diameter Input Queue	node1.messages. in_q_Gx_RAA. qns_stat.error	Count of messages failed to be sent to the policy engine	Policy Server (qns)
Gx/A	Diameter Input Queue	node1.messages. in_q_Gx_RAA. qns_stat.success	Count of messages successfully sent to the policy engine	Policy Server (qns)
Gx/E	Diameter Output Queue	node1.counters. [realm_] Gx_RAR.qns_count	Count of messages successful sent to the Policy Director (LB)	Policy Server (qns)
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_timeout. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_RAA.qns_count	Count of messages successful sent to the Policy Director (LB)	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_AAR_drop. qns_count	Count of messages dropped due to exceedingSLA	Policy Server (qns)
Rx/E	Diameter Output Queue	node1.counters. [realm_] Rx_AAA_2001. qns_count	Count of AAA messages with result-code = 2001 sent successfully to the PolicyDirector (LB)	Policy Server (qns)
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_ASR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_ASR_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_ASR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_ASR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_ASR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_ASR_retry. qns_count	Retry count for ASR message	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_ASA_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_]Rx_ASA. qns_count	Count of messages successfully sent to the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_ASA_drop. qns_count	Count of messages dropped due to exceedingSLA	Policy Server (qns)
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Input Queue	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/A	Diameter Input Queue	node1.counters.[realm_] Rx_RAA_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)
Rx/A	Diameter Output Queue	node1.counters.[realm_] Rx_RAA.qns_count	Count of messages successfully sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Rx/A	Diameter Round Trip	node1.counters.[realm_] Rx_RAA_drop. qns_count	Count of messages dropped due to exceedingSLA	Policy Server (qns)
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_2001. qns_stat.total_time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/A	Diameter Input Queue	node1.counters.[realm_] Rx_STR.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_STR_drop. qns_count	Count of messages dropped due to exceedingSLA	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.messages. in_q_Rx_STR. qns_stat.success	Count of messages successful sent to the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.messages. in_q_Rx_STR. qns_stat. total_time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Rx/D	Engine Message	node1.messages. diameter_Rx_STR. qns_stat.success	Success message count	Policy Server (qns)
Rx/D	Engine Message	node1.messages. diameter_Rx_STR. qns_stat. total_time_in_ms	Total milliseconds of successful messages	Policy Server (qns)
Rx/E	Diameter Input Queue	node1.counters. [realm_] Rx_STA_2001. qns_count	Count of STA messages with result-code = 2001 sent successfully to the PolicyDirector (LB)	Policy Server (qns)
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SLR_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Sy/A	Diameter Input Queue	node1.counters.[realm_] Sy_SLR.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters.[realm_] Sy_SLR_drop.qns_count	Count of messages dropped due to exceedingSLA	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_SLA. qns_stat.success	Count of messages successfully sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages.in_q_Sy_SLA. qns_stat. total_time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_SLA. qns_stat.success	Success message count	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_SLA. qns_stat. total_time_in_ms	Total milliseconds of successful messages	Policy Server (qns)
Sy/B	Diameter Action	node1.actions. send.diameter_ Sy_SLR.qns_stat.success	Success actions count	Policy Server (qns)
Sy/B	Diameter Action	node1.actions. send.diameter_ Sy_SLR.qns_stat. total_time_in_ms	Total milliseconds of successful messages	Policy Server (qns)
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SNR.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SNR_drop. qns_count	Count of messages dropped due to exceedingSLA	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_ Sy_SNR. qns_stat.success	Count of messages successfully sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_SNR. qns_stat. total_time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching2001	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_STA_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_STA.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_STA_drop. qns_count	Count of messages dropped due to exceedingSLA	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_STA. qns_stat.success	Count of messages successfully sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_STA. qns_stat.total_ time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_STA. qns_stat.success	Success message count	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_STA. qns_stat.total_time_in_ms	Total milliseconds of successful messages	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/Node
Sy/B	Diameter Action	node1.actions.send. diameter_Sy_STR. qns_stat.success	Success actions count	Policy Server (qns)
Sy/B	Diameter Action	node1.actions.send. diameter_ Sy_STR.qns_stat. total_time_in_ms	Total milliseconds of successful actions	Policy Server (qns)
Sy/E	Diameter Output Queue	node1.counters. [realm_] Sy_STR.qns_count	Count of messages successfully sent to the Policy Director (LB)	Policy Server (qns)

Example Statistics

Sample CSV Files

The following list is a sample of the file names created in the /var/broadhop/stats directory on the control-A VM.

```
[root@control-A stats]# pwd
/var/broadhop/stats [root@control-A stats]# ls
bulk-control-A-201510131350.csv
bulk-control-A-201510131355.csv
bulk-control-A-201510131400.csv
bulk-control-A-201510131405.csv
bulk-control-A-201510131410.csv
bulk-control-A-201510131415.csv
bulk-control-A-201510131420.csv
bulk-control-A-201510131425.csv
bulk-control-A-201510131430.csv
bulk-control-A-201510131435.csv
bulk-control-A-201510131440.csv
bulk-control-A-201510131445.csv
bulk-control-A-201510131450.csv
bulk-control-A-201510131455.csv
bulk-control-A-201510131500.csv
bulk-control-A-201510131505.csv
bulk-control-A-201510131510.csv
bulk-control-A-201510131515.csv
bulk-control-A-201510131520.csv
bulk-control-A-201510131525.csv
bulk-control-A-201510131530.csv
bulk-control-A-201510131535.csv
bulk-control-A-201510131540.csv
bulk-control-A-201510131545.csv
bulk-control-A-201510131550.csv
bulk-control-A-201510131555.csv
bulk-control-A-201510131600.csv
bulk-control-A-201510131605.csv
bulk-control-A-201510131610.csv
bulk-control-A-201510131615.csv
```

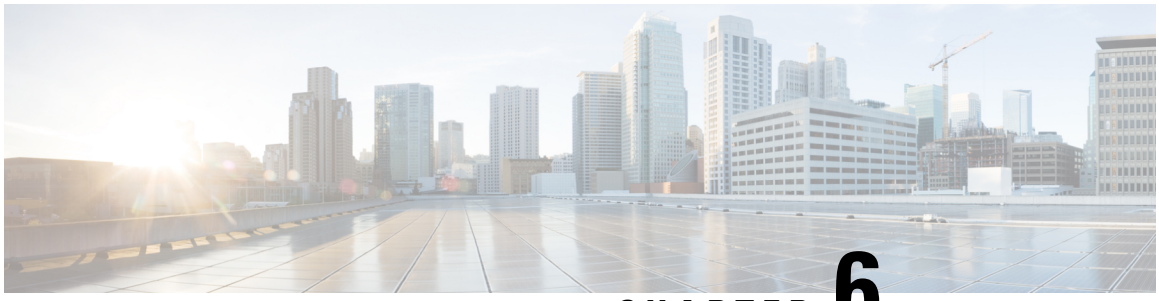
```
bulk-control-A-201510131620.csv
bulk-control-A-201510131625.csv
bulk-control-A-201510131630.csv
```

Sample Output

C,<VM_name>,node1.actions.send.diameter_Gx_CCA-I.qns_stat.success,19 where the <VM_Name> indicates the VM where statistics has been collected.

A sample bulk statistics.csv file is shown below:

```
C,qns01,node1.actions.SaveSubscriberActionImpl.qns_stat.error,0
C,qns01,node1.actions.SaveSubscriberActionImpl.qns_stat.success,6
C,qns01,node1.actions.send.diameter_Gx_CCA-I.qns_stat.error,0
C,qns01,node1.actions.send.diameter_Gx_CCA-I.qns_stat.success,19
C,qns01,node1.actions.send.diameter_Gx_CCA-T.qns_stat.error,0
C,qns01,node1.actions.send.diameter_Gx_CCA-T.qns_stat.success,9
D,qns01,node1.messages.in_q_Gx_CCR-I.qns_stat.total_time_in_ms,14
D,qns01,node1.messages.in_q_Gx_CCR-T.qns_stat.total_time_in_ms,2
D,qns01,node1.messages.in_q_Gx_CCR-U.qns_stat.total_time_in_ms,1
D,qns01,node1.messages.in_q_Gx_RAA.qns_stat.total_time_in_ms,0
D,qns01,node1.messages.in_q_Sh_SNA.qns_stat.total_time_in_ms,2
D,qns01,node1.messages.in_q_Sh_UDA.qns_stat.total_time_in_ms,0
D,qns01,node1.messages.TimerExpired.qns_stat.total_time_in_ms,7244
D,qns01,node1.spr.createSubscriber.qns_stat.total_time_in_ms,29
D,qns01,node1.spr.deleteSubscriber.qns_stat.total_time_in_ms,40
D,qns01,node1.spr.getSubscriber.qns_stat.total_time_in_ms,44
D,qns01,node1.spr.updateSubscriber.qns_stat.total_time_in_ms,21
G,lb02,node1.ldap.SITELDAP.qns_ldap_connection.MaximumAvailableConnections,10.0
G,lb02,node1.ldap.SITELDAP.qns_ldap_connection.NumAvailableConnections,0.0
G,lb02,node1.thread.gauge.daemon_thread_count,80.0
G,lb02,node1.thread.gauge.live_thread_count,184.0
```



CHAPTER 6

CLI Commands

CLI Command Overview

The command-line interface (CLI) is one of the available user interfaces to configure and monitor the launched application. This user interface provides direct access to execute commands via remote access methods over SSH.

In addition to the CLI, Cisco CPS provides a NETCONF and RESTCONF interface for API access to the application.

CLI Command Modes

The CLI provides two separate command modes – OPERATIONAL and CONFIG.

Each command mode has a separate set of commands available for configuration and monitoring of the application. Entering a "?" at the command problem will indicate the list of available commands for execution within a given mode.

When you start a session, the default mode is OPERATIONAL mode. From this mode, you can access monitoring “show” commands, debugging commands and system maintenance commands. You can enter CONFIG mode to change configuration by issuing the “config” command at the OPERATIONAL prompt.

OPERATIONAL Mode

Logging into the master VM on port 2024 via SSH will allow you to access OPERATIONAL mode. The login into the system will require the use of a username and password. You may attempt to enter a correct password up to three times before the connection attempt is refused.

The commands available at the OPERTIONAL level are separate from the ones available at the CONFIG level. In general, the OPERATIONAL commands encompass monitoring, debugging, and maintenance activity a user will perform.

To list the available OPERATIONAL commands, use the following command:

Table 8: List Commands of OPERATIONAL Mode

Command	Purpose
scheduler# ?	Lists the user OPERATIONAL commands

Example:

```
scheduler# ?
Possible completions:
  aaa                   AAA management
  apply
  autowizard            Automatically query for mandatory elements
  cd                    Change working directory
  clear                 Clear parameter
  commit                Confirm a pending commit
  compare               Compare running configuration to another configuration or a file
  complete-on-space     Enable/disable completion on space
  config                Manipulate software configuration information
  db                    DB connection and monitoring
  debug                 Debug commands
  describe              Display transparent command information
  devtools              Enable/disable development tools
  display-level         Configure show command display level
  docker                Docker Management
  exit                  Exit the management session
  file                  Perform file operations
  help                  Provide help information
  history               Configure history size
  id                    Show user id information
  idle-timeout          Configure idle timeout
  ignore-leading-space  Ignore leading whitespace (true/false)
  job                   Job operations
  logger                Log level management
  logout                Logout a user
  monitor               Application monitoring
  no                    Negate a command or set its defaults
  output-file           Copy output to file or terminal
  paginate              Paginate output from CLI commands
  prompt1               Set operational mode prompt
  prompt2               Set configure mode prompt
  pwd                   Display current mode path
  quit                  Exit the management session
  screen-length         Configure screen length
```

```
  screen-width           Configure screen width
  script                 Script actions
  send                   Send message to terminal of one or all users
  show                   Show information about the system
  show-defaults          Show default values when showing the configuration
  source                 File to source
  system                 System management
  terminal               Set terminal type
  timestamp              Enable/disable the display of timestamp
  who                    Display currently logged on users
  write                  Write configuration
scheduler#
```

The list of commands will vary based on the version of software installed.

CONFIG Mode

Within OPERATIONAL mode, you can enter CONFIG mode by issuing the “config” command. In general, the CONFIG commands modify the system configuration.

To enter CONFIG mode, use the following command:

Table 9: Enter CONFIG mode

Command	Purpose
scheduler# config	Enter CONFIG mode of the CLI

In CONFIG mode, the prompt changes to include a “(config)” at the end of the prompt.

Example:

```
scheduler# config
Entering configuration mode terminal
scheduler(config)#
```

To list the available CONFIG commands, use the following command:

Table 10: List commands in CONFIG mode

Command	Purpose
scheduler(config)# ?	List the user CONFIG commands

Example:

```
scheduler(config)# ?
Possible completions:
  aaa              AAA management
  alert            Alert status
  alias            Create command alias.
  binding          Binding DB connections
  control-plane    Cross data center control plane
  docker           Docker Management
  license          CPS License Management
  nacm             Access control
  ntp              NTP configuration
  scheduling       Service scheduling
  session          Global default CLI session parameters
  statistics       Application statistics
  system           System configuration
```

```
user            User specific command aliases and default CLI session parameters
webui           Web UI specific configuration
---
abort           Abort configuration session
annotate        Add a comment to a statement
clear           Remove all configuration changes
commit          Commit current set of changes
compare         Compare configuration
copy            Copy a list entry
describe        Display transparent command information
do              Run an operational-mode command
end             Terminate configuration session
exit            Exit from current mode
help            Provide help information
insert          Insert a parameter
load            Load configuration from an ASCII file
move            Move a parameter
no              Negate a command or set its defaults
pwd             Display current mode path
rename          Rename an identifier
resolved        Conflicts have been resolved
revert          Copy configuration from running
rollback        Roll back database to last committed version
save            Save configuration to an ASCII file
service         Modify use of network based services
show            Show a parameter
tag             Manipulate statement tags
top             Exit to top level and optionally run command
validate        Validate current configuration
```

abort

Used to terminate a configuration session and discard all uncommitted changes without system confirmations. You can use the abort command in any configuration mode.

Syntax

```
abort
```

Command Mode

CONFIG

VNFs

All

Command Usage

Use the abort command to terminate a configuration session and return to the operational mode from any configuration mode. This command causes all uncommitted configuration changes to be discarded. You are not prompted to commit the changes.

Examples

The following is an example:

```
aaa authentication users user test1 password test123 gid 100 homedir / ssh_keydir / uid
9340
admin@orchestrator[an-master](config-user-test1)# exit
admin@orchestrator[an-master](config)# abort
admin@orchestrator[an-master]#
```

alert rule

Creates a new alerting rule.

The alerting rule allows automatic creation of internal and SNMP traps based on system conditions. The Prometheus monitoring application must be running for alerts to trigger properly. If all Prometheus servers are down, then the system does not generate alerts.

Syntax

```
alert rule name duration duration event-host-label event-host-label expression expression
message message snmp-clear-message snmp-clear-message snmp-facility { application | hardware
 | networking | os | proc | virtualization } snmp-severity { alert | critical | debug |
emergency | error | info | none | notice | warning }
```

Command Parameters

Table 11: Parameter Description

Command Parameter	Description
name	The name of the alert rule.
duration	The duration measured the condition must exist before triggering an alarm. The format of the duration is <value><unit>. The value is any positive integer and the unit is one of the following: • s – second • m – minute • h – hour
event-host-label (optional)	The label received by the alerting engine from the Prometheus monitoring application. The application generates one alert per unique value of the given label. The valid labels are determined by the query executed and can be found by executing the query without the comparison operators in the Grafana application on a sample dashboard. If not defined, then the alert is considered global.

Command Parameter	Description
expression	The expression that makes up the alerting rule. The expression is built using a Prometheus expressions (https://prometheus.io/docs/querying/basics/) and must conform to the rules defined in the Prometheus alerting documentation: https://prometheus.io/docs/alerting/rules/
message	A configurable message to be sent with the alert. This message supports substitution of labels as defined in the templating section of the Prometheus documentation: https://prometheus.io/docs/alerting/rules/ . The resultant alert message is sent in any associated SNMP traps when the alert is triggered.
snmp-clear-message (optional)	A configurable message that is sent as the clear message when the alert condition is no longer valid.
snmp-facility (optional)	The target snmp-facility to use when generating SNMP trap: • application • hardware • networking • os • proc • virtualization Default is application.
snmp-severity	The target snmp-severity to use when generating an SNMP trap: • alert • critical • debug • emergency • error • info • none • notice • warning Default is alert.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the alert rule command to define monitoring rules for the system. When you create a new alert rule, the alert rule is exported to the Prometheus monitoring servers, which are monitoring the system on a 1-second interval. The Prometheus servers monitor the underlying expression defined in the alert rule and send alerts scheduling OAM node when they are triggered or when they are cleared. The OAM node tracks internally the status of all alerts and sends any SNMP traps if SNMP servers are defined.

Examples

The following example generates an alert when node_lode > 3:

```
alert rule test
 expression         "node_load5 > 3"
 event-host-label   instance
 message            "Node level exceeds 3"
 snmp-facility      application
 snmp-clear-message "Node level below 3"
!
```

alert snmp-v2-destination

Creates a new SNMPv2 destination.

Creation of a SNMPv2 destination causes the system to forward any triggered/cleared alerts to the SNMPv2 destination.

Syntax

```
alert snmp-v2-destination nms-address community community
```

Command Parameters

Table 12: Parameter Description

Command Parameter	Description
nms-address	The address to send SNMPv2 traps.
Community	The community to use for SNMPv2 traps

Command Mode

CONFIG

VNFs

All

Command Usage

Use the alert snmp-v2-destination to forward alerts from the system to an external SNMPv2 trap receiver. The traps are sent using the following algorithm:

- Sent once when the alert is cleared
- Sent once when the alert is firing
- Sent once if the OAM application is restarted and the alert is firing.

Examples

The following example sends all alerts to community “test” with address 10.10.10.10.

```
scheduler(config)# alert snmp-v2-destination 10.10.10.10 community test
```

alert snmp-v3-destination

Creates a new SNMPv3 destination.

Creation of a SNMPv3 destination causes the system to forward any triggered/cleared alerts to the SNMPv3 destination.

Syntax

```
alert snmp-v3-destination nms-address auth-password auth-password auth-proto auth-proto
engine-id engine-id privacy-password privacy-password user user
```

Command Parameters

Table 13: Parameter Description

Command Parameter	Description
nms-address	The address to send SNMPv3 traps.
auth-password	Authentication passphrase used for authenticated SNMPv3 messages.
auth-proto	Authentication protocol used for authenticated SNMPv3 messages. Valid values are MD5 and SHA
engine-id	Context engine id as a hexadecimal string.
privacy-password	Privacy passphrase used for encrypted SNMPv3 messages.
privacy-protocol	Privacy protocol used for encrypted SNMPv3 messages. Valid values are DES and AES.

Command Parameter	Description
user	Security name used for authenticated SNMPv3 messages.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the alert snmp-v3-destination to forward alerts from the system to an external SNMPv2 trap receiver. The traps are sent using the following algorithm:

- Sent once when the alert is cleared
- Sent once when the alert is firing
- Sent once if the OAM application is restarted and the alert is firing.

Examples

The following example sends all alerts to community “test” with address 10.10.10.10.

```
scheduler(config)# alert snmp-v3-destination 10.10.10.10 user test auth-proto SHA
auth-password test engine-id 0x01020304 privacy-protocol AES privacy-password  test
```

apply patches

Applies patches that are staged in the `/data/orchestrator/patches/` directory of the master VM.

This command should only be used by the Cisco TAC and Engineering team to address specific problems and debug the application.

Syntax

```
apply patches
```

Command Parameters

Table 14: Parameter Description

Command Parameter	Description
Service Name or Prefix	The exact name of the service to apply the patch or the prefix of the services to apply.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

This command should only be used at the recommendation of Cisco TAC and Engineering teams.

binding cluster-binding-dbs imsiapn-msisdnapn

Used to configure same connection pool on IMSIAPN-MSISDNAPN database transactions.

Note This command is applicable only for application client based sharding.

Syntax

```
binding cluster-binding-dbs imsiapn-msisdnapn
```

```
no binding cluster-binding-dbs
```

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use this CLI to indicate to the application that IMSI APN bindings DB and MSISDN APN Bindings DB will use the same connection pool for DB transactions.

IMSI-APN connection settings for both read and write will apply to this combined pool.

In this mode change in MSISDN APN connection settings for read or write connection pools will have no effect.

Note This is not recommended for small deployments. It is required for the deployments for which the database spans across 48 shards or more.

Examples

The following is an example:

```
admin@orchestrator(config)# binding cluster-binding-dbs imsiapn-msisdnapn
```

binding db-connection

Adds additional binding db connections from the DRA to a DRA binding database.

Note This command is applicable only for MongoDB based sharding.

Syntax

```
binding db-connection { ipv4 | ipv6 | imsiapn | msisdnapn| slf } address port
```

Command Parameters

Table 15: Parameter Description

Command Parameter	Description
ipv4	Connection definition for the IPv4 binding database.
ipv6	Connection definition for the IPv6 binding database.
imsiapn	Connection definition for the IMSI-APN binding database.
msisdnapn	Connection definition for the MSISDN-APN binding database.
slf	Connection definition for the SLF database.
address	Address of the binding DRA database. This is either an IP address or an FQDN.
port	Port of the binding DRA database.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the binding db-connection command to instruct the application on how to connect to the remote binding database. In general, there should be configuration lines entered per binding database type in order to support high availability.

Examples

The following configuration defines two redundant connections per database.

```
binding db-connection ipv6 172.16.82.195 27017
!
binding db-connection ipv6 172.16.82.196 27017
!
binding db-connection ipv4 172.16.82.195 27017
!
binding db-connection ipv4 172.16.82.196 27017
!
binding db-connection imsiapn 172.16.82.195 27017
!
binding db-connection imsiapn 172.16.82.196 27017
!
binding db-connection msisdnapn 172.16.82.195 27017
!
binding db-connection msisdnapn 172.16.82.196 27017
!
binding db-connection slf 172.16.82.195 27017
!
binding db-connection slf 172.16.82.196 27017
!
```

binding db-connection-settings

Used to configure the write mongo connection settings. The connections are used for database create/update and delete of session and bindings.

Note This command is applicable for MongoDB based and application client based sharding.

Syntax

```
binding db-connection-settings { drasession | imsiapn | ipv4 | ipv6 | msisdnapn | range |
slf } acceptable-latency-difference-for-read connect-timeout connections-per-host
max-wait-time socket-timeout

no binding db-connection-settings <database>
```

Note For Policy DRA, supported values are drasession/imsiapn/ipv4/ipv6/msisdnapn.

For recommended values, refer to *Database Connection Settings* section in the *CPS vDRA Advanced Tuning Guide*.

Command Parameters

Table 16: Parameter Description

Command Parameter	Description
drasession	Connection definition for the DRA session database.
imsiapn	Connection definition for the IMSI-APN binding database.
ipv4	Connection definition for the IPv4 binding database.

Command Parameter	Description
ipv6	Connection definition for the IPv6 binding database.
msisdnapn	Connection definition for the MSISDN-APN binding database.
range	Port range to be used.
slf	Connection definition for the SLF database.
acceptable-latency-difference-for-read	The maximum difference in ping-time latency between the fastest ping time and the slowest of the chosen servers. Default: 5
connect-timeout	Connection timeout in milliseconds. It is used only when establishing a new connection. Default: 500
connections-per-host	Maximum number of connections allowed per host for this MongoClient instance. Those connections are kept in a pool when idle. Once the pool is exhausted, any operation requiring a connection blocks waiting for an available connection. Default: 10
max-wait-time	Maximum wait time in milliseconds that a thread may wait for a connection to become available. Default: 500
socket-timeout	Socket timeout in milliseconds. It is used for I/O socket read and write operations. Default: 1000

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `binding db-connection-settings` command to configure the write mongo connection settings.

Examples

The following is an example:

```
admin@orchestrator(config)# binding db-connection-settings ?
Possible completions:
  drasession  imsiapn  ipv4  ipv6  msisdnapn  range  slf

admin@orchestrator(config)# binding db-connection-settings drasession ?
Possible completions:
  acceptable-latency-difference  connect-timeout  connections-per-host  max-wait-time
```

```
socket-timeout  <cr>

admin@orchestrator(config-db-connection-settings- drasession)# acceptable-latency-difference
 ?
Possible completions:
  <int>[5]
admin@orchestrator(config-db-connection-settings- drasession)# connect-timeout ?
Possible completions:
  <int>[500]

admin@orchestrator(config-db-connection-settings- drasession)# connections-per-host ?
Possible completions:
  <int>[10]

admin@orchestrator(config-db-connection-settings- drasession)# max-wait-time ?
Possible completions:
  <int>[500]

admin@orchestrator(config-db-connection-settings- drasession )# socket-timeout ?
Possible completions:
  <int>[1000]
```

binding db-max-record-limit

Used to configure maximum record limit on session and bindings.

Syntax

```
binding db-max-record-limit { all | drasession | imsiapn | ipv4 | ipv6 | msisdnapn | range
 | slf } <limit>
```

```
no binding db-max-record-limit drasession <limit>
```

Command Parameters

Table 17: Parameter Description

Command Parameter	Description
all	Maximum record limit on drasession, ipv6, ipv4, imsiapn and msisdnapn.
drasession	Maximum record limit on DRA session.
imsiapn	Maximum record limit on IMSI-APN.
ipv4	Maximum record limit on IPv4.
ipv6	Maximum record limit on IPv6.
msisdnapn	Maximum record limit on MSISDN-APN.
range	Not Applicable
slf	Not Applicable

Command Parameter	Description
limit	Maximum number of records to be stored in database. Default: Value of limit depends on deployment and number of shards. Hence, no default value for limit.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `db-max-record-limit` command to configure maximum record limit on session and bindings.

Examples

The following is an example:

```
admin@orchestrator[master-0m](config)# binding db-max-record-limit
Possible completions:
  all  drasession  imsiapn  ipv4  ipv6  msisdnapn  range  slf

admin@orchestrator[master-0m](config)# binding db-max-record-limit all  1000
admin@orchestrator[master-0m](config)# binding db-max-record-limit drasession 1000
admin@orchestrator[master-0m](config)# binding db-max-record-limit imsiapn 1000
admin@orchestrator[master-0m](config)# binding db-max-record-limit ipv4 1000
admin@orchestrator[master-0m](config)# binding db-max-record-limit ipv6 1000
admin@orchestrator[master-0m](config)# binding db-max-record-limit msisdnapn 1000
```

binding db-read-connection-settings

Used to configure the read mongo connection parameters.

Note This command is applicable only for application client based sharding.

Read connections are used for:

- Rx-AAR based binding look up
- Rest API binding query
- Reset of next evaluation time for both sessions and bindings
- Health checks

Syntax

```
binding db-read-connection-settings { drasession | imsiapn | ipv4 | ipv6 | msisdnapn | range
 | slf } acceptable-latency-difference-for-read connect-timeout-for-read
connections-per-host-for-read max-wait-time-for-read socket-timeout-for-read

no binding db-read-connection-settings <database>
```

Note For Policy DRA, supported values are drasession/imsiapn/ipv4/ipv6/msisdnapn.

For recommended values, refer to *Database Connection Settings* section in the *CPS vDRA Advanced Tuning Guide*.

Command Parameters

Table 18: Parameter Description

Command Parameter	Description
drasession	Connection definition for the DRA session database.
imsiapn	Connection definition for the IMSI-APN binding database.
ipv4	Connection definition for the IPv4 binding database.
ipv6	Connection definition for the IPv6 binding database.
msisdnapn	Connection definition for the MSISDN-APN binding database.
range	Port range to be used.
slf	Connection definition for the SLF database.
acceptable-latency-difference-for-read	The maximum difference in ping-time latency between the fastest ping time and the slowest of the chosen servers. Default: 5
connect-timeout-for-read	Connection timeout in milliseconds for read connection. It is used only when establishing a new connection. Default: 500
connections-per-host-for-read	Maximum number of connections allowed per host for this MongoClient instance of read connection. Those connections are kept in a pool when idle. Once the pool is exhausted, any operation requiring a connection blocks waiting for an available connection. Default: 5
max-wait-time-for-read	Maximum wait time in milliseconds that a thread may wait for a connection to become available. Default: 500

Command Parameter	Description
socket-timeout-for-read	Socket timeout in milliseconds. It is used for I/O socket read and write operations. Default: 1000

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `binding db-read-connection-setting` commands to configure the read mongo connection parameters. Applicable only for connection with client-sharded database cluster.

Examples

The following is an example for setting the connection-per-host for read connections with session-db to 5:

```
admin@orchestrator[master-0](config)# binding db-read-connection-settings drasession
connections-per-host-for-read 5
```

binding shard-metadata-db-connection

Used to configure binding shard metadata database connections from DRA to a DRA shard metadata binding database.

Note This command is applicable only for application client based sharding.

Syntax

```
binding shard-metadata-db-connection { all | drasession | imsiapn | ipv4 | ipv6 | loadmetrics
 | msisdnapn | range } <ip-address> <port>
```

```
no binding shard-metadata-db-connection { drasession | imsiapn | ipv6 | loadmetrics |
msisdnapn } <ip-address> <port>
```

Command Parameters

Table 19: Parameter Description

Command Parameter	Description
all	Connection definition for Session, IPv4, IPv6, IMSI-APN, MSISDN-APN shard metadata binding database.
drasession	Connection definition for the Session binding shard metadata database.

Command Parameter	Description
imsiapn	Connection definition for the IMSI-APN shard metadata binding database.
ipv4	Connection definition for the IPv4 binding shard metadata database.
ipv6	Connection definition for the IPv6 binding shard metadata database.
loadmetrics	Connection definition for the IMSI-APN or MSISDN-APN shard metadata binding database.
msisdnapn	Connection definition for the MSISDN-APN shard metadata binding database.
range	Not Applicable
ip-address	Address of the binding DRA database. This is either an IP address or an FQDN.
port	Port number of the binding DRA database.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `binding shard-metadata-db-connection` command to instruct the application on how to connect to the remote shard metadata binding database. In general, there should be configuration lines entered per binding database type in order to support high availability.

Examples

The following configuration defines two redundant connections per database:

```
binding shard-metadata-db-connection drasession 172.16.82.195 27017
!
binding shard-metadata-db-connection drasession 172.16.82.196 27017
!
binding shard-metadata-db-connection ipv6 172.16.82.195 27017
!
binding shard-metadata-db-connection ipv6 172.16.82.196 27017
!
binding shard-metadata-db-connection ipv4 172.16.82.195 27017
!
binding shard-metadata-db-connection ipv4 172.16.82.196 27017
!
binding shard-metadata-db-connection imsiapn 172.16.82.195 27017
!
binding shard-metadata-db-connection imsiapn 172.16.82.196 27017
!
binding shard-metadata-db-connection msisdnapn 172.16.82.195 27017
!
binding shard-metadata-db-connection msisdnapn 172.16.82.196 27017
!
binding shard-metadata-db-connection loadmetrics 172.16.82.195 27017
```

```
!
binding shard-metadata-db-connection loadmetrics 172.16.82.196 27017
!
```

binding throttle-db-operation

Used to configure CPU usage threshold value for read and write database operations.

Note This command is applicable only for application client based sharding.

Important The following commands need to be configured to monitor CPU usage of all the database VMs:

`binding shard-metadata-db-connection loadmetrics <ip-address> <port>`

For more information on `binding shard-metadata-db-connection`, refer to binding shard-metadata-db-connection, on page 99.

Syntax

`binding throttle-db-operation { range | read | write } max-cpu-usage <cpu_value>`

`no binding throttle-db-operation { range | read | write } max-cpu-usage`

Command Parameters

Table 20: Parameter Description

Command Parameter	Description
range	Not applicable.
read	CPU threshold for read database operations.
write	CPU threshold for write database operations.
cpu_value	CPU threshold value.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `binding throttle-db-operation` command to configure the read and write CPU threshold value to throttle the read and write database operations.

Examples

The following configuration defines CPU threshold value for read and write database operations:

```
binding throttle-db-operation read max-cpu-usage 70
!
binding throttle-db-operation write max-cpu-usage 70
!
```

clear

Used to clear uncommitted changes.

Syntax

```
clear
```

Command Mode

CONFIG

VNFs

All

Command Usage

Use the clear command to discard all the uncommited changes.

Examples

The following is an example:

```
clear
All configuration changes will be lost. Proceed? [yes, NO]
```

compare

Used to compare the similar configurations.

Syntax

```
compare cfg <configuration path> to <configuration path>
```

Command Mode

CONFIG

VNFs

All

Command Usage

- To compare the similar configurations in configuration mode.
- Need to represent exact ideal configuration paths.

Examples

The following is an example:

```
compare cfg aaa authentication users user admin to aaa authentication users user oper
- password $1$ftGF2fQE$4P145tnwbouLSr8pbm4EW1;
+ password $1$sFadxrqz$Tp88/Go3jTNUuloSdPB9K.;
- ssh_keydir /var/confd/homes/oper/.ssh;
+ ssh_keydir /var/confd/homes/admin/.ssh;
- homedir /var/confd/homes/oper;
+ homedir /var/confd/homes/admin;
```

consul

Used to list, save, delete, and restore the consul snapshot from the `/data/orchestrator/config/snapshot/` directory.

Syntax

```
consul [list-snapshots | save-snapshot snapshot-name nameofsnapshot | restore-snapshot
snapshot-name nameofsnapshot | delete-snapshot nameofsnapshot]
```

Command Parameters

Table 21: Parameter Description

Command Parameter	Description
list-snapshots	Lists all the snapshots present in `/data/orchestrator/config/snapshot/` directory.
save-snapshot	Saves the snapshot.
snapshot-name *nameofsnapshot*	Snapshot name.
restore-snapshot	Restore the snapshot.
delete-snapshot	Delete the snapshot.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `consul` command to list, save, delete, and restore the consul snapshot in the `/data/orchestrator/config/snapshot/` directory.

Examples

The following are the examples:

```
admin@orchestrator[ss-master-binding-0]# consul list-snapshots
Snapshot Name
*****************************************************

19.5.5-20200105_131756.6477
19.5.8-20200214_025459.6674

*****************************************************
admin@orchestrator[ss-master-binding-0]# consul save-snapshot snapshot-name snap1
result Snapshot is created in /data/orchestrator/config/snapshot-consul/snap1
admin@orchestrator[ss-master-binding-0]# consul list-snapshots
Snapshot Name
*****************************************************

19.5.5-20200105_131756.6477
19.5.8-20200214_025459.6674
snap1

*****************************************************
admin@orchestrator[ss-master-binding-0]# consul save-snapshot snapshot-name snap2
result Snapshot is created in /data/orchestrator/config/snapshot-consul/snap2
admin@orchestrator[ss-master-binding-0]# consul list-snapshots
Snapshot Name
*****************************************************

19.5.5-20200105_131756.6477
19.5.8-20200214_025459.6674
snap1
snap2

*****************************************************
admin@orchestrator[ss-master-binding-0]# consul delete-snapshot snap2
Snapshot is deleted
admin@orchestrator[ss-master-binding-0]# consul list-snapshots
Snapshot Name
*****************************************************

19.5.5-20200105_131756.6477
19.5.8-20200214_025459.6674
snap1

*****************************************************
admin@orchestrator[ss-master-binding-0]# consul restore-snapshot ?
Possible completions:
  snapshot-name
admin@orchestrator[ss-master-binding-0]# consul restore-snapshot snapshot-name snap1
result Snapshot restore success.
admin@orchestrator[ss-master-binding-0]# consul list-snapshots
Snapshot Name
*****************************************************
```

```
19.5.5-20200105_131756.6477
19.5.8-20200214_025459.6674
snap1

*******************************************************
admin@orchestrator[ss-master-binding-0]# consul delete-snapshot snap1
Snapshot is deleted
admin@orchestrator[ss-master-binding-0]#
```

control-plane relay

Adds additional control-plane entries between two disconnected CPS vDRA sites.

Syntax

```
control-plane relay name address address port port
```

Command Parameters

Table 22: Parameter Description

Command Parameter	Description
Name	A short name describing the connection.
address	An IP address or FQDN of the connection. IPv6 address must be enclosed in square brackets.
port (optional)	The destination port of the connection. Defaults to 6379 if not defined.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the control-plane relay command to instruct the application how which links it should use to relay CPS vDRA control traffic. CPS vDRA control traffic is the traffic that describes the current endpoints within a site and the relay IPs for site to site communication. For a 2 site model there should be at least 4 entries defined in this definition (two for each site). For a 3 site model there should be at least 6 entries in this definition.

Examples

The following configuration adds a relay connection to siteA over address 10.10.10.10 port 6379.

```
scheduler(config)# control-plane relay siteA-1 address 10.10.10.10 port 6379
```

control-plane ipc-endpoint update-interval

Used to configure IPC endpoint update interval.

Syntax

```
control-plane ipc-endpoint update-interval <time-in-milliseconds>
no control-plane ipc-endpoint update-interval
```

Command Parameters

Table 23: Parameter Description

Command Parameter	Description
time-in-milliseconds	IPC endpoint update interval in milliseconds. Default: 100 milliseconds

Command Mode

CONFIG

VNFs

DRA

Command Usage

This command is used to configure the frequency for updating the IPC endpoints.

Examples

The following configuration adds an 200 milliseconds interval for updating the IPC endpoints.

```
scheduler(config)# control-plane ipc-endpoint update-interval 200
```

Note For more information on the values to be configured, refer to *Control Plane Tuning Configuration* section in the *CPS vDRA Advanced Tuning Guide*.

control-plane remote-peer-policy global accept

Used to configure the control plane remote peer policy.

Syntax

```
control-plane remote-peer-policy global accept all
```

```
control-plane remote-peer-policy global accept diameter-applications [Gx | Gy | Rx | Sd |
Sy]
```

Command Parameters

Table 24: Parameter Description

Command Parameter	Description
[Gx | Gy | Rx | Sd | Sy]	Application type.

By default, DRA accepts all the applications from all the sites.

Command Mode

CONFIG

VNFs

DRA

Command Usage

This command is used to configure the control plane remote peer policy for the DRA system to accept peer connection information from other DRA systems. Policy can be configured to accept peer connection information for all Diameter application types or only specific Diameter application types. DRA system can route messages only to remote peers accepted by policy.

Examples

Example 1:

```
control-plane remote-peer-policy global accept diameter-applications Rx
```

Example 2:

```
control-plane remote-peer-policy global accept diameter-applications [ Gx Rx Gy ]
```

Example 3:

```
control-plane remote-peer-policy global accept all
```

Example 4:

```
no control-plane remote-peer-policy global accept diameter-applications [ Gx Rx Gy ]
```

control-plane remote-peer-policy mated-system id

Used to configure the mated system ID.

Syntax

```
control-plane remote-peer-policy mated-system id <system-id>
```

Command Parameters

Table 25: Parameter Description

Command Parameter	Description
system-id	System ID of the mated system.

Command Mode

CONFIG

VNFs

DRA

Command Usage

This command is used to configure the system ID of the mated DRA system. DRA system accepts peer information for all Diameter application types from the mated system.

Example

```
control-plane remote-peer-policy mated-system id system-02
```

control-plane timers peer-status-update-interval

Used to modify the value of peer status update interval and peer expiration duration.

Syntax

```
control-plane timers peer-status-update-interval <time-in-ms> peer-expiration-duration
<duration-in-ms>
```

Command Parameters

Table 26: Parameter Description

Command Parameter	Description
time-in-ms	Peer status update interval time in ms. Default: 2000 milliseconds
duration-in-ms	Peer expiration duration in ms. Default: 10000 milliseconds

Command Mode

CONFIG

VNFs

DRA

Command Usage

This command allows tuning the frequency at which director nodes send periodic status updates for peers connected to the nodes. The command also allows tuning the expiration time for peers maintained in topology when consecutive periodic status updates are not received for the peers.

Peer expiration duration should be equal to three times of peer status update interval.

For example, if peer-status-update-interval = 4000 ms then, peer-expiration-duration = 12000

To reflect the peer expiration duration change, application should be restarted in both director and worker nodes.

Example

```
control-plane timers peer-status-update-interval 4000 peer-expiration-duration 12000
```

database cluster

Create a MongoDB database sharded cluster.

Syntax

```
database cluster name sharded-cluster-master {true|false}
no database cluster name
```

Command Parameters

Table 27: Parameter Description

Command Parameter	Description
Name	A short name describing the DB cluster. Each application will use a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records.
sharded-cluster-master	This parameter indicates if the current VNF will execute provisioning operations on the given cluster. If multiple VNF (s) have the same database cluster configuration only one of them should have the “sharded-cluster-master” set to true.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster command and sub-commands to instruct the application to provision a database cluster for use in application database operations.

Examples

The following is an example of creating a “binding” sharded cluster that is being managed by the current VNF.

```
scheduler(config)# database cluster binding sharded-cluster-master true
```

database cluster *db-name* config-server *name*

Add a MongoDB configuration server process to the named database cluster.

Note This command is applicable only for MongoDB based sharding.

Syntax

```
database cluster db-name config-server name address address
no database cluster db-name config-server name
```

Command Parameters

Table 28: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application will use a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records
Name	A short description of the config server name.
address	The IPv4 or IPv6 address of the config server. This parameter does not accept FQDN address format.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster config-server to add a config-server to the system.

Examples

The following is an example of adding a new config server to the “binding” cluster.

```
scheduler(config)# database cluster binding config-server cfg-1 address 10.10.10.10
```

database cluster *db-name* config-server-seed *name*

Set the initial seed configuration server for boot-strapping the MongoDB replica set initialization process.

Note This command is applicable only for MongoDB based sharding.

Syntax

```
database cluster db-name config-server-seed name
```

Command Parameters

Table 29: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application will use a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records
Name	A reference to the configuration server name that will act as the seed for bootstrapping the initial replica set.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster config-server-seed command to set the initial seed configuration server for boot-strapping the MongoDB replica set initialization process. This is required if a config server is set.

Examples

The following is an example of setting cfg-1 as the initial seed for a new config server to the "binding" cluster.

```
scheduler(config)# database cluster binding config-server-seed cfg-1
```

database cluster *db-name* multi-db-collections *noOfShardsPerDB*

Used to add a MongoDB sharding configuration server process to the named database cluster.

Note This command is applicable only for application client based sharding.

Syntax

```
database cluster db-name mutli-db-collections noOfShardsPerDB
no database cluster db-name multi-db-collections
```

Command Parameters

Table 30: Parameter Description

Command Parameter	Description
DB Name	A short name describing the database cluster. Each application uses a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name "binding" for storing binding and session records.
noOfShardsPerDB	Number of shards created per database.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster multi-db-cluster to create those number of shards per database.

Examples

The following is an example of enabling multi-db-collections to the “binding” cluster.

```
admin@orchestrator[master-hostname](config)# database cluster binding multi-db-collections
 2
```

database cluster *db-name* router *name*

Add a new MongoDB router to the named DB cluster.

Note This command is applicable only for MongoDB based sharding.

Syntax

```
database cluster db-name router name
```

Command Parameters

Table 31: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application will use a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records
Name	A short description of the router name.
address	The IPv4 or IPv6 address of the config server. This parameter does not accept FQDN address format
port	The port to bind the router. Generally 27017

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster router command to add a router to named database cluster. Full initialization of database cluster requires at least one router to be defined and often for HA purposes multiple routers are required.

Examples

The following is an example of adding a router to the “binding” cluster.

```
scheduler(config)# database cluster binding router router-1 address 10.10.10.10 port 27017
```

database cluster *db-name* shard *name*

Add a new MongoDB shard to the named database cluster.

Syntax

```
database cluster db-name shard name
no database cluster db-name shard name
```

Command Parameters

Table 32: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application will use a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records
Name	A short description of the shard name.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster shard command to add a new shard to the named database cluster. Full initialization of database cluster requires at least the definition of one shard and often for scaling purposes multiple shards are required.

Examples

The following is an example of adding a shard to the “binding” cluster.

```
database cluster binding shard shard-1
```

database cluster *db-name* shard *shard-name* shard-server *name*

Add a new MongoDB shard to the named DB cluster.

Syntax

```
database cluster db-name shard shard-name shard-server name address address port port
[arbiter {true|false}] [memory_allocation_percent percent] [priority priority] [voter
{true|false}] [storage-engine {IN_MEMORY|MMAPv1|WT}]

no database cluster db-name shard shard-name server name
```

Note When creating replica set, ensure that all ports are the same, i.e, the replica set should have same port for ARBITER, PRIMARY, and SECONDARY.

Command Parameters

Table 33: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application will use a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records
Shard Name	A short description of the shard name.
Name	A short description of the server name.
address	The IPv4 or IPv6 address of the router server. This parameter does not accept FQDN address format.
port	The port to bind the router. Generally -27017
arbiter	Indicates if this node is only an arbiter node.
memory_allocation_percent	Percent (expresses as a positive integer) of the amount of memory to allocate to the DB process for the in-memory storage option.
priority	Relative priority of the node in the shard
voter	Whether this node is a voter.

Command Parameter	Description
storage-engine	The storage engine to provision for the process. Valid values are: • IN_MEMORY - pure in memory storage • MMAPv1 – Memory mapped files • WT –wired tirger

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster shard server command to add a new server to named database cluster. Full initialization of database cluster requires at least the definition of one shard server and for HA at least 3 nodes are required.

Examples

The following is an example of adding a new shard to the “binding” cluster.

```
scheduler(config)# database cluster binding shard shard-1 shard-server server-1 storage-engine
 WT address 10.10.10.10 port 27017
```

Note Ports to be used for all database operations must be in the range of 27017 to 27047. Ports outside the defined range are not supported since the application must limit the port mappings. The selected range is sufficient for 30 Mongo processes on a given node.

database cluster *db-name* shard *shard-name* shard-server-seed *name*

Set the initial seed shard server for boot-strapping the MongoDB replica set initialization process.

Syntax

```
database cluster db-name shard shard-name shard-server-seed name
```

Command Parameters

Table 34: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application will use a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name "binding" for storing binding and session records
Shard Name	A short description of the shard name.
Name	A reference to the shard server name that will act as the seed for bootstrapping the initial replica set.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster shard-server-seed command to set the initial seed shard server for boot-strapping the MongoDB replica set initialization process. This is required if a shard is defined.

Note To create or add a member to an existing replica set, you must also run the Mongo console-based commands as shown: `mongo> rs.add("name")`

To remove a replica set or a shard in a sharded cluster case, remove the member from the Mongo console as shown: `mongo> rs.remove("name")`

You must also navigate to the container and the VM on which the member resides and clear the data manually. The data path is the same as the one that is used when the replica-set member is created. Typically, the path is `//mmapv1-tmpfs-2xxxx` where `2xxxx` is the port where the replica set member is started.

Examples

The following is an example of setting server-1 as the initial seed for a new shard called "shard-1" to the "binding" cluster.

```
scheduler(config)# database cluster binding shard shard-1 shard-server-seed server-1
```

database cluster *db-name* sharding-db *name*

Adds a MongoDB sharding configuration server process to the named database cluster.

Note This command is applicable only for application client based sharding.

Syntax

```
database cluster db-name sharding-db name address address
no database cluster db-name sharding-db name
```

Command Parameters

Table 35: Parameter Description

Command Parameter	Description
DB Name	A short name describing the database cluster. Each application uses a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records.
Name	A short description of the sharding database name.
address	The IPv4 or IPv6 address of the configuration server. This parameter does not accept FQDN address format.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster sharding-db to add a sharding config-server to the system.

Examples

The following is an example of adding new sharding database to “binding” cluster.

```
admin@orchestrator[master-hostname](config)# database cluster binding sharding-db shdb-1
address 10.10.10.10
```

database cluster *db-name* sharding-db-seed *name*

Sets the initial seed configuration server for boot-strapping the MongoDB replica set initialization process.

Note This command is applicable only for application client based sharding.

Syntax

```
database cluster db-name sharding-db-seed name
```

Command Parameters

Table 36: Parameter Description

Command Parameter	Description
DB Name	A short name describing the database cluster. Each application uses a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name "binding" for storing binding and session records.
Name	A reference to the configuration server name that will act as the seed for bootstrapping the initial replica set.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the database cluster sharding-db-seed command to set the initial seed configuration server for boot-strapping the MongoDB replica set initialization process. This is required if a sharding database is set.

Examples

The following is an example of setting shdb-1 as the initial seed for a new sharding database to the "binding" cluster.

```
admin@orchestrator[master-hostname](config)# database cluster binding sharding-db-seed
shdb-1
```

database cluster *db-name* ipv6-zone-sharding

Enable the zone-based sharding for IPv6. When zone-based sharding is enabled on IPv6 database, hash-based sharding can still be configured on other databases.

Syntax

```
database cluster <db name> ipv6-zone-sharding true/false
```

Command Parameters

Table 37: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application uses a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name "binding" for storing binding and session records.
ipv6-zone-sharding	Enables (true) or disables (false) zone-based sharding for IPv6 database. Default: False

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use database cluster binding ipv6-zone-sharding to enable/disable zone sharding on IPv6 database.

Examples

The following is an example of enabling zone-based sharding for the IPv6 database in the cluster `binding`:

```
database cluster binding ipv6-zone-sharding true
```

database cluster *db-name* ipv6-zones-range *zone-name* zone-range *range-name* start *pool-starting-address* end *pool-ending-address*

Create zones for IPv6 shards based on IPv6 pools, so that the primary member of the replica set for an IPv6 address resides at the same physical location as the PGW assigning addresses from the IPv6 pool. This results in local writes (and reads) for the IPv6 binding database.

Note It is possible to create multiple ranges for each zone. Configure the IPv6 ranges in short format only.

Syntax

```
database cluster <db name> ipv6-zones-range <zone-name> zone-range <range-name> start <pool
 starting address> end <pool ending address>
```

Command Parameters

Table 38: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application uses a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records.
Zone name	A short name describing Zone name. Unique name to identify the zone that the shard configuration uses to map to zone.
Range name	A short name describing the range within the zone.
Pool Starting Address	The starting IPv6 Prefix address for the particular range that can be from same physical location as PGW.
Pool Ending Address	The ending IPv6 Prefix address for the particular range that can be from same physical location as PGW.

Command Mode

CONFIG

VNFs

DRA Binding

Command Usage

This command creates a zone and also creates ranges for the zone.

Examples

The following is an example of creating a IPv6 zone with name `pune` for the cluster `binding` and a range of 2003:3051:0000:0001 to 2003:3051:0000:0500 for the zone:

```
database cluster binding ipv6-zones-range pune zone-range range1 start 2003:3051:0000:0001
 end 2003:3051:0000:0500
```

database cluster *db-name* shard *shard-name* zone-name *zone-name*

Add shards to a zone.

Syntax

```
database cluster <db name> shard <shard name> zone-name <zone-name>
```

Command Parameters

Table 39: Parameter Description

Command Parameter	Description
DB Name	A short name describing the DB cluster. Each application uses a set of pre-defined names and this name should match one of the application names. For example, DRA uses the name “binding” for storing binding and session records.
Zone name	A short name describing Zone name.
Shard name	A short description of the shard name.

Command Mode

CONFIG

VNFs

DRA Binding

Command Usage

Use the command to add the shard to a zone.

Examples

The following is an example of mapping the IPv6 zone with name `pune` with the shard `shard-1` in the cluster `binding`:

```
database cluster binding shard shard-1 zone-name pune
```

database delete all-bindings-sessions

Deletes the data belonging to given range and zone for all the bindings and sessions databases.

Syntax

```
database delete all-bindings-sessions <bindings-cluster-name> <sessions-cluster-name>
<zone-name> <start-address> <end-address>
```

Command Parameters

Table 40: Parameter Description

Command Parameter	Description
bindings-cluster-name	Specifies the bindings cluster name on which deletion jobs has to be performed.
sessions-cluster-name	Specifies the sessions cluster name on which deletion job has to be performed.
zone-name	Specifies the zone from which bindings have to be deleted. **Note** • If zone name is default, bindings records (all types and sessions) for the specified range data are deleted from all shards in database clusters. • If zone name is not default, bindings records (all types include sessions) for the specified range data are deleted from shards assigned to the specified zone.
start-address	Start address of IPv6 address range.
end-address	End address of IPv6 address range.

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use the `database delete all-bindings-sessions zone` command to delete IPv6 bindings and all the associated bindings for the specified address range from the specified zone.

Examples

The following example deletes IPv6 bindings and all the associated bindings from a specific zone:

```
database delete all-bindings-sessions imsi-msisdn session-ipv6-AB pune 7507:9903:1808:8000
 7507:9903:1808:8fff
```

The following example deletes IPv6 bindings and all the associated bindings from the default zone:

```
database delete all-bindings-sessions imsi-msisdn session-ipv6-AB default 7507:9903:1808:8000
 7507:9903:1808:8fff
```

database delete ipv6bindings

Deletes IPv6 bindings for the specified address range from the specified zone.

Syntax

```
database delete ipv6bindings <sessions-cluster-name> <zone-name> <start-address> <end-address>
```

Command Parameters

Table 41: Parameter Description

Command Parameter	Description
sessions-cluster-name	Specifies the sessions cluster name on which deletion job has to be performed.
zone-name	Specifies the zone from which bindings have to be deleted. **Note** • If zone name is default, IPv6 bindings for the range are deleted from all shards of database cluster. • If zone name is not default, IPv6 bindings for the range are deleted only from shards assigned to the specified zone.
start-address	Start address of IPv6 address range.
end-address	End address of IPv6 address range.

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use the `database delete ipv6bindings` command to delete IPv6 bindings for the specified address range from the specified zone.

Examples

The following example deletes IPv6 bindings from a specific zone:

```
database delete ipv6bindings session-ipv6-AB pune 7507:9903:1808:8000 7507:9903:1808:8fff
```

The following example deletes IPv6 bindings from the default zone:

```
database delete ipv6bindings session-ipv6-AB default 7507:9903:1808:8000 7507:9903:1808:8fff
```

database fcvcheck

Used to check and set fcv on databases.

Syntax

```
database fcvcheck
```

Command Parameters

Table 42: Parameter Description

Command Parameter	Description
check	Check fcv on all members (primary/secondary).
set	Check and set fcv on primary member.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the database fcvcheck command to check and set fcv on databases.

Examples

Binding VNF (set): The following is an example to set fcv on primary member.

```
database fcvcheck

Press "set" to check and set fcv on primary members or
Press "check" to only check fcv on all members.
(set/check) << set
This is going to check and set required FCV on orchestrator,mongo-admin-db,shards and
shardingDb databases.
Press yes to continue
(yes/no) << yes
Please do not kill the terminal untill FCV check completes.Kindly check
logs(Path:/var/log/broadhop/fcv.log) for more info
FCV check for shard databases is in Progress......
FCV check for shard databases is Completed
FCV check for shardingDb databases is in Progress......
FCV check for shardingDb databases is Completed
FCV check for orchestrator and mongo-admin-db databases is in Progress......
FCV check for orchestrator and mongo-admin-db databases is Completed
```

Binding VNF (check): The following is an example to check fcv on all members.

```
database fcvcheck
```

```
Press "set" to check and set fcv on primary members or
Press "check" to only check fcv on all members.
(set/check) << check
FCV check for shard databases for all members is in Progress......
FCV check for shard databases for all members is Completed
FCV check for shardingDb databases for all members is in Progress......
FCV check for shardingDb databases for all members is Completed
FCV check for orchestrator and mongo-admin-db databases for all members is in Progress......
FCV check for orchestrator and mongo-admin-db databases for all members is Completed
```

DRA VNF (set): The following is an example to set fcv on primary member.

```
database fcvcheck

Press "set" to check and set fcv on primary members or
Press "check" to only check fcv on all members.
(set/check) << set
This is going to check and set required FCV on orchestrator and mongo-admin-db databases.
Press yes to continue
(yes/no) << yes
Please do not kill the terminal untill FCV check completes.Kindly check
logs(Path:/var/log/broadhop/fcv.log) for more info
FCV check for orchestrator and mongo-admin-db databases is in Progress......
FCV check for orchestrator and mongo-admin-db databases is Completed
```

DRA VNF (check): The following is an example to check fcv on all members.

```
database fcvcheck

Press "set" to check and set fcv on primary members or
Press "check" to only check fcv on all members.
(set/check) << check
FCV check for orchestrator and mongo-admin-db databases for all members is in Progress......
FCV check for orchestrator and mongo-admin-db databases for all members is Completed
```

database query

Fetches records for database in a specified cluster.

Note This feature has not been validated for all customer deployment scenarios. Please contact your Sales Account team for support.

Syntax

```
database query clustername cluster-name dbname db-name query query-to-run
method operation max max-value start start-value
```

Command Parameters

Table 43: Parameter Description

Command Parameter	Description
cluster-name	Cluster name to obtain records from.
db-name	DB to obtain records from.

Command Parameter	Description
query-to-run	Input query inside double quotes.
operation	Operation to perform on database.
max-value	Number of records to be limited. Default:10.
start-value	Range of records to start from. Default: 0.

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use the database query command to fetch records for db in a specified cluster.

Examples

The following is an example of fetching 30 records starting from 10 index from IPv6 database in binding cluster for query "{'uuid': {'$regex': 'vpas-system-1200'}}".

Default <= 5 records are displayed, all 30 records are saved in /data/config/Query.log.
admin@orchestrator[Binding-master]# database query clustername binding dbname IPV6 query "{'uuid': {'$regex': 'vpas-system-1200'}}" method find max 30 start 10

```
Press yes to continue
(yes/no) << yes
{'_id': '2507:9903:1808:0583',
 'fqdn': 'site-d-client-calipers21-gx.pcef.gx',
 'sessionid': 'kmanchan21411kunalmanchanda2',
 'srk': 'server.sitedstandalone',
 'staleBindingExpiryTime': datetime.datetime(2021, 5, 10, 13, 27, 24, 762000),
 'systemId': 'vpas-system-1',
 'ts': 1620048444762,
 'uuid': 'vpas-system-12002439730'}
{'_id': '2507:9903:1808:061a',
 'fqdn': 'site-d-client-calipers21-gx.pcef.gx',
 'sessionid': 'kmanchan21562kunalmanchanda2',
 'srk': 'server.sitedstandalone',
 'staleBindingExpiryTime': datetime.datetime(2021, 5, 10, 13, 27, 24, 763000),
 'systemId': 'vpas-system-1',
 'ts': 1620048444763,
 'uuid': 'vpas-system-12002439881'}
{'_id': '2507:9903:1808:065d',
 'fqdn': 'site-d-client-calipers21-gx.pcef.gx',
 'sessionid': 'kmanchan21629kunalmanchanda2',
 'srk': 'server.sitedstandalone',
 'staleBindingExpiryTime': datetime.datetime(2021, 5, 10, 13, 27, 24, 763000),
 'systemId': 'vpas-system-1',
 'ts': 1620048444763,
 'uuid': 'vpas-system-12002439948'}
{'_id': '2507:9903:1808:0694',
 'fqdn': 'site-d-client-calipers21-gx.pcef.gx',
```

```
 'sessionid': 'kmanchan21684kunalmanchanda2',
 'srk': 'server.sitedstandalone',
 'staleBindingExpiryTime': datetime.datetime(2021, 5, 10, 13, 27, 24, 764000),
 'systemId': 'vpas-system-1',
 'ts': 1620048444764,
 'uuid': 'vpas-system-12002440003'}
{'_id': '2507:9903:1808:06b7',
 'fqdn': 'site-d-client-calipers21-gx.pcef.gx',
 'sessionid': 'kmanchan21719kunalmanchanda2',
 'srk': 'server.sitedstandalone',
 'staleBindingExpiryTime': datetime.datetime(2021, 5, 10, 13, 27, 24, 766000),
 'systemId': 'vpas-system-1',
 'ts': 1620048444766,
 'uuid': 'vpas-system-12002440038'}
Default <= 5 records are displayed, Kindly check records in /data/config/Query.log based
on max value provided.
```

Query Example:

format: Query should be in standard mongo db query format

query "{'key1': 'value1'}"

query "{'key1': 'value1', 'key2': 'value2'}"

Query Restrictions

Queries are allowed only for exact key matches that are specified in the Parameters Choices table. Any new key is allowed in query-based on requirement as a part of a patch. This ensures unnecessary processing on the shards if invalid/service impacting fields are mentioned in queries.

Adding New Query

For new query key support, update the `queries.jar` as part of a patch.

- **method**: [count, find]
- **method restrictions**: Other methods [update, insert, delete] are not allowed.
- **max**: [5,25000], default =5
- **start**: default = 0

database repair

Used to recover single/multiple/all shards and sharding database.

Attention

In HA deployment, CLI needs to be run on single site.

Logs (`/var/log/broadhop/shardrecovery.log`) should be checked after executing CLI.

Syntax

```
database repair <clustername> <shardname>
```

```
database repair <clustername> <shardname1> <shardname2> <shardname3>
database repair <clustername> All
database repair <clustername> sharddb
```

Command Parameters

Table 44: Parameter Description

Command Parameter	Description
clustername	Name of the cluster to which shard belongs.
shardname	Name of the shard to be recovered.
All	All shards in the cluster.
sharddb	Sharding database recovery.

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use the database repair commands to recover single/multiple/all shards and sharding database.

Examples

The following is an example to recover shard1 in binding cluster.

```
database repair binding shard1
```

The following is an example to recover shard1, shard2, shard3, and shard4 in binding cluster

```
database repair binding shard1 shard2 shard3 shard4
```

The following is an example to recover all shards in the binding cluster.

```
database repair All.
```

The following is an example to recover sharding database in the binding cluster

```
databse repair shard-db
```

db-authentication set-password database redis password

To set the Redis authentication password.

Syntax

```
db-authentication set-password database redis password <clear text password>
```

Command Parameters

Table 45: Parameter Description

Command Parameter	Description
<clear text password>	A clear text password used for Redis authentication. The password is stored in consul datastore in encrypted format. The Redis password is stored in consul datastore in encrypted format and synchronized to `draTopology.ini` which is used by dra-endpoint application. The service reads the password from consul datastore and password is updated in the console data store with encrypted password. Data store and `draTopology.ini` format: `redis/config/password:<encrypted password>`

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use the database authentication command to set the Redis password which is used to access Redis data store.

Examples

The following is an example to set the Redis authentication password:

```
admin@orchestrator[master-0m]# db-authentication set-password database redis password
Value for 'password' (<string>): ***********
result SUCCESS
```

db-authentication show-password database redis

To display the encrypted redis password.

Syntax

```
db-authentication show-password database redis
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use the database authentication command to display the Redis password.

Examples

The following is an example to display the Redis authentication password:

```
admin@orchestrator[master-0m]# db-authentication show-password database redis
result
result PASSWORD : 72261348A44594381D2E84ADDD1E6D9A
```

db-authentication remove-password database redis

To remove Redis authentication password.

Syntax

```
db-authentication remove-password database redis current-password password
```

Command Parameters

Table 46: Parameter Description

Command Parameter	Description
password	Clear text password to be removed on redis need to be provided.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use the `db-authentication` command to remove Redis authentication password.

Examples

The following is an example to remove Redis authentication password:

```
admin@orchestrator[master-0m]# db-authentication remove-password database redis
Value for 'current-password' (<string>): *******
result SUCCESS
```

db-authentication show-password database mongo

To display the encrypted MongoDB password.

Syntax

```
db-authentication show-password database mongo
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the database authentication command to display the MongoDB password.

Examples

The following is an example:

```
scheduler# db-authentication show-password database mongo
result
adminuser: 3300901EA069E81CE29D4F77DE3C85FA
```

db-authentication set-password database mongo password

Used to create users (adminuser and backupuser) with credentials in the MongoDB.

Syntax

```
db-authentication set-password database mongo password <password>
```

Command Parameters

Table 47: Parameter Description

Command Parameter	Description
password	Clear text password to be set on Mongo DB need to be provided.

Command Mode

OPERATIONAL

VNFs

DRA and Binding

Command Usage

This command is used to create users (adminuser and backupuser) with credentials in the MongoDB.

Examples

The following is an example to create users with credentials:

```
admin@orchestrator[binding-master]# db-authentication set-password database mongo password
Value for 'password' (<string>): ******
result SUCCESS
```

db-authentication remove-password database mongo

Used to remove the users (admin user and backup user) and password from all the databases.

Syntax

```
db-authentication remove-password database mongo current-password <password>
```

Command Parameters

Table 48: Parameter Description

Command Parameter	Description
password	Clear text password to be removed on MongoDB need to be provided.

Command Mode

OPERATIONAL

VNFs

DRA and Binding

Command Usage

Use to remove users and password from the mongo databases. Before using this command the database should be in transition authentication state and after this command rolling restart is mandatory.

Examples

The following is an example to remove-password in mongo database:

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

db-authentication change-password database mongo

Used to change the admin user password in all the databases.

Syntax

```
db-authentication change-password database mongo current-password <current pasword>
new-password <New password> user adminuser
```

Command Parameters

Table 49: Parameter Description

Command Parameter	Description
Current Password	Current password set in MongoDB.
New Password	New password to be set in MongoDB.

Command Mode

OPERATIONAL

VNFs

DRA and Binding

Command Usage

This command change password of adminuser in all the MongoDB.

Examples

The following is an example to change-password in MongoDB:

```
admin@orchestrator[binding-master]# db-authentication change-password database mongo user
adminuser
Value for 'current-password' (<string>): ******
Value for 'new-password' (<string>): ********
result SUCCESS
```

db-authentication sync-password database mongo

Used to synchronize the backup user password same as admin user password..

Syntax

```
db-authentication sync-password database mongo
```

Command Mode

OPERATIONAL

VNFs

DRA and Binding

Command Usage

This command is used to sync password in all the MongoDB.

Examples

The following is an example to synchronize the passwords:

```
admin@orchestrator[binding-master]# db-authentication sync-password database mongo
result
SUCCESS : Mongo password sync successful
```

db-authentication enable-transition-auth database mongo

Used to configure the transition authentication parameter. Rolling restart should be executed after this command.

Syntax

```
db-authentication enable-transition-auth database mongo
```

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use this command to configure the transition authentication parameter.

Examples

The following is an example to configure the transition authentication parameter:

```
admin@orchestrator[binding-master]# db-authentication enable-transition-auth database mongo
```

db-authentication disable-transition-auth database mongo

Used to remove the transition authentication parameter. Rolling restart should be done after this command.

Syntax

```
db-authentication disable-transition-auth database mongo
```

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use this command to remove the transition authentication parameter.

Examples

The following is an example to disable transition authorization in MongoDB:

```
admin@orchestrator[binding-master]# db-authentication disable-transition-auth database mongo
```

db-authentication rolling-restart database mongo

Used to restart all the database instances where primary members is followed by secondary members.

Syntax

```
db-authentication rolling-restart database mongo
```

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use this command to restart all the database instances where primary members will be followed by secondary members.

Examples

The following is an example to restart all the database instances:

```
admin@orchestrator[binding-master]# db-authentication rolling-restart database mongo
```

db-authentication rolling-restart-parallel database mongo

Used to restart multiple database instances in parallel without affecting the availability of DB cluster.

Syntax

```
db-authentication rolling-restart-parallel database mongo
```

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use `db-authentication rolling-restart-parallel database mongo` command to restart multiple database instances in parallel without affecting the availability of DB cluster.

Note `db-authentication rolling-restart-parallel database mongo` command is dependent on `show database parallel-upgrade-plan`.

If `show database parallel-upgrade-plan` does not provide any output, then do not use `db-authentication rolling-restart-parallel database mongo`. Instead use `db-authentication rolling-restart database mongo` command.

Example: If `show database parallel-upgrade-plan` displays the following output:

```
admin@orchestrator[an-dbmaster]# show database parallel-upgrade-plan
BATCH  MODULE          HOST            ADDRESS
-----------------------------------------------------
1      mongo-node-101  an-dbmaster     192.168.11.40
1      mongo-node-102  an-dbcontrol-0  192.168.11.41
1      mongo-node-103  an-dbcontrol-1  192.168.11.42
2      mongo-node-104  an-pers-db-0    192.168.11.43
3      mongo-node-105  an-pers-db-1    192.168.11.44
```

then, `db-authentication rolling-restart-parallel database mongo` combines 101, 102 and 103 in batch 1 and restarts all of them at the same time.

After batch 1, node 104 from batch 2 is restarted followed by node 105 (from batch 3). So, all the nodes from same batch are restarted in parallel. However nodes from different batch are restarted in sequential manner.

A batch_interval parameter can be added as follows:

```
admin@orchestrator[an-master]# db-authentication rolling-restart-parallel batch_interval 8

database mongo
```

where, batch_interval is an integer and accepts range between 8 to 60. By default, the value is 10. It represents the delay duration in seconds between processing of 2 subsequent batches. After executing this command, batch-wise status can be tracked using `db-authentication rolling-restart-parallel-status database mongo` command.

Examples

The following is an example to trigger a parallel restart for mongo-nodes.

```
db-authentication rolling-restart-parallel database mongo
```

db-authentication rolling-restart-parallel-status database mongo

Used to track the status of rolling-restart-parallel command.

Syntax

```
db-authentication rolling-restart-parallel-status database mongo
```

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use `db-authentication rolling-restart-parallel-status database mongo` command to track the status of rolling-restart-parallel command.

Examples

The following example shows which batch is completed out of total batches.

```
admin@orchestrator[an-dbmaster]# db-authentication rolling-restart-parallel-status database
 mongo
result Still in progress...Batch 1 out of total 3 is completed at 2019-12-10T23:16:25.799
admin@orchestrator[an-dbmaster]# db-authentication rolling-restart-parallel-status database
 mongo
result Still in progress...Batch 2 out of total 3 is completed at 2019-12-10T23:16:37.656
admin@orchestrator[an-dbmaster]# db-authentication rolling-restart-parallel-status database
 mongo
result Parallel rolling restart completed!! Batch 3 out of total 3 got completed at
2019-12-10T23:16:49.844
admin@orchestrator[an-dbmaster]# db-authentication rolling-restart-parallel-status database
 mongo
result
Parallel Rolling Restart: Not Scheduled/Completed/Just triggered
admin@orchestrator[an-dbmaster]#
```

db-authentication rolling-restart-status database mongo

Used to display the status of rolling restart as in-progress or completed.

Syntax

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

Command Mode

OPERATIONAL

VNFs

Binding

Command Usage

Use this command to display the status of rolling restart as in-progress or completed.

Examples

The following is an example to display the status of rolling restart:

```
admin@orchestrator[binding-master]# db-authentication rolling-restart-status database mongo
result
Rolling Restart: In Progress ...
```

db connect admin

Connects to an underlying admin database.

Syntax

No additional arguments.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the db connect admin command to connect to the underlying admin database. Once within this database, the user will have read / write access to the admin database via a mongodb CLI. The capabilities of the mongodb CLI are not described in this document.

db connect binding

Connects to an underlying binding database.

Syntax

```
db connect binding { ipv4 | ipv6 | imsi-apn | msisdn-apn | slf }
```

Command Parameters

Table 50: Parameter Description

Command Parameter	Description
ipv4	Connect to the IPv4 binding database.
ipv6	Connect to the IPv6 binding database.
imsi-apn	Connect to the IMSI-APN binding database.
msisdn-apn	Connect to the MSISDN-APN binding database.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use the db connect binding command to connect to the underlying binding database. Once within this database, the user will have read / write access to the binding database via the mongodb CLI. The capabilities of the mongodb CLI are not described in this document.

db connect session

Connects to an underlying admin database.

Syntax

No additional arguments.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use the db connect session command to connect to the underlying session database. Once within this database, the user will have read / write access to the session database via a mongodb CLI. The capabilities of the mongodb CLI are not described in this document.

debug collect-db-logs-advanced collect

Used to collect mongod logs from specified VMs based on the start and end timestamps.

You can also add the maximum storage capacity of logs to be allowed as input. Once the maximum capacity is reached, the log collection stops.

Note The log collection is limited to 15 days. If you need logs beyond 15, you must login to VM directly to pull the logs.

Syntax

```
debug collect-db-logs-advanced collect <max-allowed-log-size-in-gb> <start-time> <end-time>
 [VM-names]
```

Command Parameters

Table 51: Parameter Description

Command Parameter	Description
max-allowed-log-size-in-gb	Maximum size of the logs to be collected.
start-time	Specify the start time to start collecting the logs.
end-time	Specify the end time to end collecting the logs.
VM-names (Optional)	Docker engine VM names to be mentioned with space in between. If the VM names are not specified, the logs are collected for all the VMs in VNF.

Command Mode

Operational

VNFs

DRA Binding

Command Usage

Use this command to collect mongod logs from specified binding VNF VMs based on the start and end timestamps.

Output files from this command can be accessed using the following link:

https://<*MasterVM*>/orchestrator/downloads/debug/consolidated/consolidated-db-logs/

Examples

The following is an example:

```
debug collect-db-logs-advanced collect 4 2020-07-14T23:30:09 2020-07-15T04:15:20 VM-1 VM-2
```

Output file: `consolidated-db-logs_<StartDate>_<EndDate>.tar.gz`

debug collect-db-logs-advanced scan

Used to create a single consolidated log file of all MongoDB logs collected from different VMs based on start and end timestamps.

Before executing `debug collect-db-logs-advanced scan` command, you need to execute `collect` command which pulls all the logs from different VMs into `tar.gz`.

Note This command allows you to input timestamps in maximum of 6 hours time interval.

Currently, this command expects `tar.gz` file to be present in the respective storage location and creates `consolidated-log-output` in same place.

Syntax

```
debug collect-db-logs-advanced scan <start-time> <end-time> [VM-names]
```

Command Parameters

Table 52: Parameter Description

Command Parameter	Description
start-time	Specify the start time to start scanning the logs.
end-time	Specify the end time to end scanning the logs.
VM-names (Optional)	Docker engine VM names to be mentioned with space in between. If the VM names are not specified, the logs are collected for all the VMs in VNF.

Command Mode

Operational

VNFs

DRA Binding

Command Usage

Use this command to scan the MongoDB logs collected from different binding VMs based on start and end timestamps.

Output files from this command can be accessed using the following link:

https://<*MasterVM*>/orchestrator/downloads/debug/consolidated/consolidated-db-logs/

Examples

The following is an example:

```
debug collect-db-logs-advanced scan 2020-07-14T23:30:09 2020-07-15T04:15:20 VM-1 VM-2
```

Output file: `consolidated-logs-output`

debug log collect

Used to gather various logs to support troubleshooting.

Syntax

```
debug log collect [ docker { all all | vmname name  } ]
[ journalctl { all all | vmname name  } ]
[ pb { all all | vmname name  } ]
[ qns { all all | vmname name  } ]
[ tech  ]
[ top { all all | vmname name  } ]
[ mongo { all all | vmname name  } ]
```

Command Parameters

Table 53: Parameter Description

Command Parameter	Description
VM-names (Optional)	Docker engine, journalctl, qns, mongo DB VM names to be mentioned with space in between. If the VM names are not specified, the logs are collected for all the VMs in VNF.

Command Mode

Operational

VNFs

DRA

Command Usage

Use this command to gather various logs to support troubleshooting.

debug log collect heapdump containername <name>

Considers diameter-endpoint container name as input and collects heapdump outputs.

Examples

The following is an example:

```
debug log collect heapdump containername diameter-endpoint-s104
Collection of heapdump completed
```

debug log collect threaddump containername <name>

Considers diameter-endpoint container name as input and collects a set of threadump outputs.

Examples

The following is an example:

```
debug log collect threaddump containername diameter-endpoint-s104
Collection of threaddump completed
```

debug packet-capture gather

Gathers all running packet captures.

Syntax

```
debug packet-capture gather directory directory
```

Command Parameters

Table 54: Parameter Description

Command Parameter	Description
directory	The directory to store the resultant pcap files. This directory is available for downloading via the web file download interface at https://<master ip>/orchestrator/downloads/debug/<directory>.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `debug packet-capture gather` to gather all completed or currently running pcaps. This command is sent to all machines with active tcpdump commands and stops the given commands. After all commands are stopped, the command will gather the resultant pcap files and make them available at https://<master ip>/orchestrator/downloads/debug/<directory>.

debug packet-capture purge

Purges all existing pcap files.

Syntax

```
debug packet-capture purge
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `debug packet-capture purge` after all relevant packet captures have been downloaded from the application. The system does not automatically purge packet captures. You need to manage the amount of space used by the packet captures using this command.

debug packet-capture start

Starts a packet capture on a given IP address and port.

Syntax

`debug packet-capture start ip-address` *`ip-address`* `port` *`port`* `timer-seconds` *`timer-seconds`*

Command Parameters

Table 55: Parameter Description

Command Parameter	Description
ip-address	The IP address to start the packet capture. This address can either be IPv4 or IPv6..
port	The port to start the packet capture.
timer-seconds	Duration to run the packet capture - measured in seconds

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the debug packet-capture start command to start a tcp-dump on the given IP address and port within the CPS cluster. The packet capture will run for the given timer period and then shutdown automatically. The packet captures can be gathered using the debug packet-capture gather command.

debug tech

Gather logs and debug information to support troubleshooting.

Syntax

`debug tech`

Command Parameters

None

Command Mode

OPERATIONAL – Not available via NETCONF/RESTCONF

VNFs

All

Command Usage

Use this command to gather logs and debug information to support troubleshooting.

The results of the command are available at https://<master ip>/orchestrator/downloads/debug/tech.

Examples

```
scheduler# debug tech
```

docker connect

Connects to a docker service and launches a bash shell running on the system.

Syntax

```
docker connect container-id
```

Command Parameters

Table 56: Parameter Description

Command Parameter	Description
container-id	The docker container to open a bash shell. Use the **show docker service** command to find the list of valid container-ids.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `docker connect` to open a bash shell within a container. This command is primarily used for advanced debugging of the system. Once within a container, you can execute Linux commands and interact with the running container processes.

docker exec

Used to support executing specific command on specific or all the containers.

Syntax

```
docker exec <container-name> <command>
```

Command Parameters

Table 57: Parameter Description

Command Parameter	Description
container-name	Specifies the container-name (prefix or full-name). Enter the complete name for running the command in all the containers.
command	The command that needs to be executed on the containers.

Command Mode

Operational

VNFs

All

Command Usage

Use `docker exec <container-name> <command>` to take container-name and command as an argument. Container-name can be prefix or full name. If the command is having space then it should be provided between double quotes.

Examples

The following example shows sample commands:

Example 1: Stop the db-monitor process in mongo-monitor containers.

```
docker exec mongo-mon "supervisorctl stop db-monitor"
```

Example 2: Get the supervisorctl status from all the containers.

```
docker exec all "supervisorctl status"
```

docker repair

Used to remove mongo-s running containers from VMs.

Note This command must be executed in Maintenance Window (MW).

Syntax

```
docker repair <prefix> <VM-1 VM-2 ... VM-n>
```

Command Parameters

Table 58: Parameter Description

Command Parameter	Description
prefix	Container name to be removed. **Note** Currently, only mongo-s prefix is supported.
--no-prompt	Used to force repair.
VMs	Specify engine node name. You can get the VM names using `show docker engine` command.
all <module-name>	Used to remove all the containers which contain module-name as mongo-node and prefix as mongo-s. It won't remove mongo-monitor containers.

Command Mode

Operational

VNFs

All

Command Usage

Use this command to remove the mongo-s containers from VMs to clear the high usage of tmpfs file system memory. In case if any mongo-s container fails to come up or mongod inside it doesn’t come up with healthy state then the entire repair operation is aborted.

Examples

The following example shows sample commands:

Example 1: Remove mongo-s container from a single VM with user prompt.

```
docker repair mongo-s control-binding-0
Are you sure to repair this mongo-s102 (y/n)? y
mongo-s102
Checking health status for mongo-s102.
Healthy Check Status for mongo-s102 = true
```

Example 2: Remove mongo-s container from multiple VMs with user prompt.

```
docker repair mongo-s control-binding-0 control-binding-1
Are you sure to repair this mongo-s102 (y/n)? y
mongo-s102
Checking health status for mongo-s102.
Healthy Check Status for mongo-s102 = true
Are you sure to repair this mongo-s103 (y/n)? y
mongo-s103
```

```
Checking health status for mongo-s103.
Healthy Check Status for mongo-s103 = true
```

Example3: Remove mongo-s container from multiple VMs without user prompt.

```
docker repair mongo-s --no-prompt control-binding-0 control-binding-1
mongo-s102
Checking health status for mongo-s102.
Healthy Check Status for mongo-s102 = true
mongo-s103
Checking health status for mongo-s103.
Healthy Check Status for mongo-s103 = true
```

Example 4: Remove all the mongo-s containers from the module-name with prefix mongo-s with user prompt.

```
docker repair mongo-s all mongo-node
Are you sure to repair this mongo-s101 (y/n)? y
mongo-s101
Checking health status for mongo-s101.
Healthy Check Status for mongo-s101 = true
Are you sure to repair this mongo-s102 (y/n)? y
mongo-s102
Checking health status for mongo-s102.
Healthy Check Status for mongo-s102 = true
Are you sure to repair this mongo-s103 (y/n)? y
mongo-s103
Checking health status for mongo-s103.
Healthy Check Status for mongo-s103 = true
Are you sure to repair this mongo-s104 (y/n)? y
mongo-s104
Checking health status for mongo-s104.
Healthy Check Status for mongo-s104 = true
Are you sure to repair this mongo-s105 (y/n)? y
mongo-s105
Checking health status for mongo-s105.
Healthy Check Status for mongo-s105 = true
Are you sure to repair this mongo-s106 (y/n)? y
mongo-s106
Checking health status for mongo-s106.
Healthy Check Status for mongo-s106 = true
Are you sure to repair this mongo-s107 (y/n)? y
mongo-s107
Checking health status for mongo-s107.
Healthy Check Status for mongo-s107 = true
```

Example 5: Remove all the mongo-s containers from the module-name with prefix mongo-s without user prompt.

```
docker repair mongo-s --no-prompt  all mongo-node
mongo-s101
Checking health status for mongo-s101.
Healthy Check Status for mongo-s101 = true
mongo-s102
Checking health status for mongo-s102.
Healthy Check Status for mongo-s102 = true
mongo-s103
Checking health status for mongo-s103.
Healthy Check Status for mongo-s103 = true
mongo-s104
Checking health status for mongo-s104.
Healthy Check Status for mongo-s104 = true
mongo-s105
Checking health status for mongo-s105.
Healthy Check Status for mongo-s105 = true
mongo-s106
Checking health status for mongo-s106.
```

```
Healthy Check Status for mongo-s106 = true
mongo-s107
Checking health status for mongo-s107.
Healthy Check Status for mongo-s107 = true
```

docker restart

Restarts a docker service that is currently running.

Syntax

`docker restart container-id container-id`

Command Parameters

Table 59: Parameter Description

Command Parameter	Description
container-id	The docker container to restart. Use the **show docker service** command to find the list of valid container-ids.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `docker restart` to restart a running docker service. This command is primarily useful to restore a non-responsive service at the request of Cisco TAC or Cisco Engineering.

docker start

Starts Diameter application container.

Syntax

`docker start <container-id>`

Command Parameters

Table 60: Parameter Description

Command Parameter	Description
container-id	Diameter application container name

Command Mode

OPERATIONAL

VNFs

DRA

docker stop

Stops Diameter application container.

Syntax

```
docker stop <container-id>
```

Command Parameters

Table 61: Parameter Description

Command Parameter	Description
container-id	Diameter application container name

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

This command ensures the following tasks are completed before the container is stopped:

- the required DPR messages are sent out to all connected peers
- VIP moves to another director

dra-distributor balance connection

Used to audit peer connections with the provided service name.

Syntax

```
dra-distributor balance connection <cluster-name> <service-name> audit
```

Command Parameters

Table 62: Parameter Description

Command Parameter	Description
cluster-name	Cluster name of the distributor service.
service-name	Service name of the floating IP address.
audit	Displays the number of connections per director based on the service name. Used to verify the total active peer connections on each diameter-endpoint containers and determines whether the connections are balanced or unbalanced between the containers.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

This command is used to audit the peer connections.

Syntax

```
dra-distributor balance connection <cluster-name> <service-name>
```

Command Parameters

Table 63: Parameter Description

Command Parameter	Description
cluster-name	Cluster name of the distributor service.
service-name	Service name of the floating IP address.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

This command checks the balancing and determines if connections need to be balanced. If the connections are unbalanced, it allows user to balance the connections.

Example

```
admin@orchestrator[vpas-A-dra-master-0]# dra-distributor balance connection client Gx-PCRFA
 audit
================================================================================
Total Directors                   8
Total Weight                      8
Total Active Connections          184
Connection Per Weight             23.0
================================================================================

Real-Server                                              Weight  Active  Expected
                                                                 Conn    Conn
172.16.XX.YY:3868 (diameter-endpoint-s104)               1       23      23.0
172.16.XX.YY:3868 (diameter-endpoint-s105)               1       23      23.0
172.16.XX.YY:3868 (diameter-endpoint-s106)               1       23      23.0
172.16.XX.YY:3868 (diameter-endpoint-s107)               1       23      23.0
172.16.XX.YY:3868 (diameter-endpoint-s108)               1       23      23.0
172.16.XX.YY:3868 (diameter-endpoint-s109)               1       23      23.0
172.16.XX.YY:3868 (diameter-endpoint-s110)               1       23      23.0
172.16.XX.YY:3868 (diameter-endpoint-s111)               1       23      23.0
================================================================================

Connections are properly distributed
```

dra-distributor balance traffic

Used to audit per director's TPS with the provided service name.

Syntax

```
dra-distributor balance traffic <cluster-name> <service-name> <threshold> <margin> audit
```

Command Parameters

Table 64: Parameter Description

Command Parameter	Description
cluster-name	Cluster name of the distributor service.
service-name	Service name of the floating IP address.
threshold	Threshold value. Balance traffic when any director traffic exceeds this threshold. **Note** Currently, threshold attribute is not considered for this command. It's for experimental purpose only.

Command Parameter	Description
margin	Traffic margin value. Balance traffic when any director traffic is not in the range from (Threshold-Margin) to (Threshold+Margin). **Note** Currently, margin attribute is not considered for this command. It's for experimental purpose only.
audit	Displays the per director TPS based on service name. Audits whether the existing traffic is distributed or not-distributed equally among directors.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

This command is used to view per director's traffic to VIPs.

Example

```
admin@orchestrator[vpas-A-dra-master-0]# dra-distributor balance traffic client Sy-OCSA 100
 122 audit
Peer disconnect is sensitive operation, so please re-authentication
Enter The Admin Role User Name [default:admin]:
Enter Password:
=========================================================

Real-Server Active Traffic
Conn
diameter-endpoint-s104(172.16.XX.YY) 1 1224 *
diameter-endpoint-s105(172.16.XX.YY) 1 1211 *
diameter-endpoint-s106(172.16.XX.YY) 1 1196 *
diameter-endpoint-s107(172.16.XX.YY) 1 1193 *
=========================================================================
Total Directors 4
Total Traffic 4824
Traffic Per Director 1206
=========================================================================
Traffic of all directors between 1084 and 1328
Traffic are properly distributed
```

dra migration

enable-migration

Enable migration handling for moving from mongo-sharded database to application-sharded database.

Syntax

```
dra migration enable-migration true

no dra migration enable-migration
```

Command Mode

CONFIG

VNFs

DRA VNF

Command Usage

Enable handling of database migration. If binding record is not found in primary database (default, application-sharded database cluster) then the binding lookup is done in secondary database (default, mongo-sharded database cluster).

Examples

The following is an example:

```
admin@orchestrator[master-0](config)# dra migration enable-migration true
```

enable-mongo-sharded-db-as-primary-db

Mongo-sharded database is considered as primary database during migration handling.

Syntax

```
dra migration enable-mongo-sharded-db-as-primary-db [true|false]
no dra migration enable-mongo-sharded-db-as-primary-db
```

Command Mode

CONFIG

VNFs

DRA VNF

Command Usage

Make mongo-sharded database as the primary database for binding lookup (lookup bindings in mongo-sharded database first and if the binding record is not found then the binding is lookup in application-sharded database).

Note By default, application-sharded database is considered as primary database.

Examples

The following is an example:

```
admin@orchestrator[master-0](config)# dra migration enable-mongo-sharded-db-as-primary-db
true
```

enable-skipping-probe-message-binding-lookup

Skip binding lookup in secondary database for probe/dummy AAR messages.

Syntax

```
dra migration enable-skipping-probe-message-binding-lookup [true|false]
no dra migration enable-skipping-probe-message-binding-lookup
```

Command Mode

CONFIG

VNFs

DRA VNF

Command Usage

Enable skipping binding lookup in secondary database for probe/dummy Rx AAR messages (sent by PCRF as part of binding database health check).

Examples

The following is an example:

```
admin@orchestrator[master-0](config)# dra migration
enable-skipping-probe-message-binding-lookup true
```

dra subscriber-trace db-connection

To configure mongo db uri.

Syntax

```
dra subscriber-trace db-connection <ip> <port>
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use this CLI command to set new mongo db uri. By default, DRA uses mongo-admin-a:27017, mongo-admin-b:27017,mongo-admin-c:27017 to store all pcap, version, and trace keys.

Examples

The following is an example:

```
show running-config dra subscriber-trace
dra subscriber-trace db-connection 182.22.31.60.27017
!
admin@orchestrator[master-00]
```

dra subscriber-trace db-pcap-collection-max-size

To change pcap collection size in Megabytes.

Syntax

```
dra subscriber-trace db-pcap-collectection-max-size <size in MB>
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use this command to change the size of pcap_files collection. By default, the collection “pcap_files” is created with size 1024 MB. Since the collection “pcap_files” is created as capped collection, DRA automatically deletes oldest pcap entries from the collection and stores new pcap entries.

Examples

The following is an example:

```
show running-config dra subcriber-trace
dra subscriber-trace db-pcap-collection-max-size 1024
!
admin@orchestrator[master-00]
```

drasubscriber-monitor-activitydb-activity-collection-max-size

To change activity collection size in megabytes.

Syntax

```
dra subscriber-monitor-activity db-activity-collection-max-size <size in MB>
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use this command to change the activity collection size in megabytes. By default, the collection to store subscriber activities is created with size 1024 MB.

Examples

The following is an example:

```
show running-config dra subcriber-monitor-activity
dra subscriber-monitor-activity db-activity-collection-max-size 1024
!
admin@orchestrator[master-00]
```

dra subscriber-monitor-activity db-connection

To change mongo db uri:

Note The monitor subscriber-activity CLI is used only to view live logs and is not used to store/stop monitor logging activity.

Syntax

```
dra subscriber-monitor-activity db-connection <ip> <port>
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use this CLI command to change mongo db uri. By default, DRA stores monitor activity keys and activity logs in mongo-admin-a:27017, mongo-admin-b:27017, mongo-admin-c:27017.

Examples

The following is an example:

```
show running-config subscriber-monitor-activity
dra subscriber-monitor-activity db-connection 182.22.31.60.27017
!
admin@orchestrator[master-00]
```

dra set-ratelimit binding-api

To configure common rate limit for all binding APIs.

Syntax

dra set-ratelimit binding-api *rate limit value*

Command Parameters

Table 65: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which all binding API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set the rate limit for all binding API queries, such as imsi imsi-apn, msisdn,msisdn-apn, or Ipv6.

Examples

The following is an example:

dra set-ratelimit binding-api 100.

dra set-ratelimit binding-api-imsi

To configure rate limit for imsi binding API queries.

Syntax

dra set-ratelimit binding-api-imsi *rate limit value*

Command Parameters

Table 66: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which imsi binding API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit only for imsi binding API queries.

Examples

The following is an example:

dra set-ratelimit binding-api-imsi 200.

dra set-ratelimit binding-api-imsi-apn

To configure binding API with IMSI APN rate limit.

Syntax

dra set-ratelimit binding-api-imsi-apn *rate limit value*

Command Parameters

Table 67: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which imsi-apn binding API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit only for imsi-apn binding API queries.

Examples

The following is an example:

dra set-ratelimit binding-api-imsi-apn 200.

dra set-ratelimit topology-api

To configure rate limit for topology related API queries.

Syntax

dra set-ratelimit topology-api *rate limit value*

Command Parameters

Table 68: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which topology related API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit for topology related API queries.

Examples

The following is an example:

dra set-ratelimit topology-api 150.

dra set-ratelimit binding-api-ipv6

To configure rate limit for ipv6 binding API queries.

Syntax

dra set-ratelimit binding-api-ipv6 *rate limit value*

Command Parameters

Table 69: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which ipv6 binding API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit only for ipv6 binding API queries.

Examples

The following is an example:

dra set-ratelimit binding-api-ipv6 100.

dra set-ratelimit oam-api

To configure rate limit for OAM-related API queries.

Syntax

dra set-ratelimit oam-api *rate limit value*

Command Parameters

Table 70: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which OAM (svn/grafana/prometheus/pb/central/custrefdata) related API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit for OAM (svn/grafana/prometheus/pb/centralcustrefdata) related API queries.

Examples

The following is an example:

dra set-ratelimit oam-api 50

dra set-ratelimit slf-api

To configure rate limit for SLF-related API queries.

Syntax

dra set-ratelimit slf-api *rate limit value*

Command Parameters

Table 71: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which slf related API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit for SLF-related API queries..

Examples

The following is an example:

dra set-ratelimit slf-api 300

dra set-ratelimit session-api

To configure rate limit for session related API queries.

Syntax

dra set-ratelimit session-api *rate limit value*

Command Parameters

Table 72: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which session related queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit for session related API queries.

Examples

The following is an example:

dra set-ratelimit session-api 50

dra set-ratelimit binding-api-msisdn

To configure rate limit only for msisdn binding API queries.

Syntax

dra set-ratelimit binding-api-msisdn *rate limit value*

Command Parameters

Table 73: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which msisdn binding API queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit only for msisdn binding API queries.

Examples

The following is an example:

dra set-ratelimit binding-api-msisdn 200

dra set-ratelimit binding-api-msisdn-apn

To configure rate limit only for msisdn-apn binding API queries.

Syntax

dra set-ratelimit binding-api-msisdn-apn *rate limit value*

Command Parameters

Table 74: Parameter Description

Command Parameter	Description
rate limit value	Specifies the value at which msisdn-apn binding queries will be rate limited.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to set rate limit only for msisdn-apn binding API queries.

Examples

The following is an example:

dra set-ratelimit binding-api-msisdn-apn 200.

dra remove-ratelimit binding-api-imsi

Removes rate limit for all binding API queries.

Syntax

```
dra remove-ratelimit binding-api-imsi
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit only for imsi binding API queries.

dra remove-ratelimit binding-api-imsi-apn

Removes rate limit only for imsi-apn binding API queries.

Syntax

dra remove-ratelimit binding-api-imsi-apn

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this CLI command to remove rate limit only for imsi-apn binding API queries.

dra remove-ratelimit binding-api-ipv6

Removes rate limit for ipv6 binding API queries.

Syntax

```
dra remove-ratelimit binding-api-ipv6
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit only for ipv6 binding API queries.

dra remove-ratelimit binding-api-msisdn-apn

Removes rate limit for msisdn-apn binding API queries.

Syntax

```
dra remove-ratelimit binding-api-msisdn-apn
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit only for msisdn-apn binding API queries.

dra remove-ratelimit binding-api-msisdn

Removes rate limit for msisdn binding API queries.

Syntax

```
dra remove-ratelimit binding-api-msisdn
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit only for msisdn binding API queries.

dra remove-ratelimit binding-api

To remove rate limit for all binding API queries.

Syntax

```
dra remove-ratelimit binding-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit for all binding API queries (imsi/imsi-apn/msisdn/msisdn-apn/Ipv6), which was configured using option *binding-api*.

Examples

The following is an example to remove rate limit binding-api:

```
dra remove-ratelimit binding-api
```

dra remove-ratelimit oam-api

Removes rate limit for rate limit for OAM related API queries.

Syntax

```
dra remove-ratelimit oam-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit for OAM (svn/grafana/prometheus/pb/centralcustrefdata) related API queries.

dra remove-ratelimit session-api

Removes rate limit for session related API queries.

Syntax

```
dra remove-ratelimit session-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit for session related API queries.

dra remove-ratelimit slf-api

Removes rate limit for slf related API queries.

Syntax

```
dra remove-ratelimit slf-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to remove rate limit for slf related API queries.

dra show-ratelimit topology-api

Used to display the configured rate limit for topology related API queries.

Syntax

```
dra show-ratelimit topology-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for topology related API queries.

Examples

The following is an example:

```
dra show-ratelimit topology-api
dra/ratelimit/topology-api:150
```

dra show-ratelimit binding-api-imsi-apn

Used to display the configured rate limit for imsi-apn binding API queries.

Syntax

```
dra show-ratelimit binding-api-imsi-apn
```

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for imsi-apn binding API queries.

Examples

The following is an example:

```
dra show-ratelimit binding-api-imsi-apn
dra/ratelimit/binding-api-imsi-apn:200
```

dra show-ratelimit binding-api-imsi

Used to display the configured rate limit for imsi or imsi-apn binding API queries.

Syntax

```
dra show-ratelimit binding-api-imsi
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for imsi or imsi-apn binding API queries.

Examples

The following is an example:

```
dra show-ratelimit binding-api-imsi
dra/ratelimit/binding-api-imsi:200
dra/ratelimit/binding-api-imsi-apn:200
```

dra show-ratelimit binding-api-msisdn-apn

Used to display the configured rate limit for msisdn-apn binding API queries.

Syntax

```
dra show-ratelimit binding-api-msisdn-apn
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for msisdn-apn binding API queries.

Examples

The following is an example:

```
dra show-ratelimit binding-api-msisdn-apn
dra/ratelimit/binding-api-msisdn-apn:200
```

dra show-ratelimit binding-api-ipv6

Used to display the configured rate limit for ipv6 binding API queries.

Syntax

```
dra show-ratelimit binding-api-ipv6
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for ipv6 binding API queries.

Examples

The following is an example:

```
dra show-ratelimit binding-api-ipv6
dra/ratelimit/binding-api-ipv6:100
```

dra show-ratelimit binding-api-msisdn

Used to display the configured rate limit for msisdn or msisdn-apn binding API queries.

Syntax

```
dra show-ratelimit binding-api-msisdn
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for msisdn or msisdn-apn binding API queries.

Examples

The following is an example:

```
dra show-ratelimit binding-api-msisdn
dra/ratelimit/binding-api-msisdn:200
dra/ratelimit/binding-api-msisdn-apn:200
```

dra show-ratelimit binding-api

Used to display the configured rate limit of all binding API queries.

Syntax

```
dra show-ratelimit binding-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit of all binding API queries.

Examples

The following is an example:

```
dra show-ratelimit binding-api
dra/ratelimit/binding-api:100
dra/ratelimit/binding-api-imsi:200
dra/ratelimit/binding-api-imsi-apn:200
dra/ratelimit/binding-api-ipv6:100
dra/ratelimit/binding-api-msisdn:200
dra/ratelimit/binding-api-msisdn-apn:200
```

dra show-ratelimit oam-api

Used to display the configured rate limit for OAM-related API queries.

Syntax

```
dra show-ratelimit oam-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for OAM (svn/grafana/prometheus/pb/centralcustrefdata) related API queries.

Examples

The following is an example:

```
dra show-ratelimit oam-api
dra/ratelimit/oam-api:50
```

dra show-ratelimit session-api

Used to display the configured rate limit for session related API queries.

Syntax

```
dra show-ratelimit session-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for session related API queries.

Examples

The following is an example:

```
dra show-ratelimit session-api
dra/ratelimit/session-api:50
```

dra show-ratelimit slf-api

Used to display the configured rate limit for SLF related API queries.

Syntax

```
dra show-ratelimit slf-api
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for SLF related API queries.

Examples

The following is an example:

```
dra show-ratelimit slf-api
dra/ratelimit/slf-api:300
```

dra show-ratelimit

Used to display the configured rate limit for all ingress APIs.

Syntax

```
dra show-ratelimit
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to view the configured rate limit for all ingress APIs.

Examples

The following is an example:

```
dra show-ratelimit
dra/ratelimit/binding-api:100
dra/ratelimit/binding-api-imsi:200
dra/ratelimit/binding-api-imsi-apn:200
dra/ratelimit/binding-api-ipv6:100
dra/ratelimit/binding-api-msisdn:200
dra/ratelimit/binding-api-msisdn-apn:200
dra/ratelimit/oam-api:50
dra/ratelimit/session-api:50
dra/ratelimit/slf-api:300
dra/ratelimit/topology-api:150
```

dra ipc-send-thread

Used to configure the IPC send thread parameters.

Syntax

dra ipc-send-thread limit *thread-limit* **lock-sla-timeout** *time-in-ms* **message-throttle-duration** *duration-in- ms* **timeout-sample-to-throttle** *max-samples*

Command Parameters

Table 75: Parameter Description

Command Parameter	Description
thread-limit	The maximum number of IPC send threads that waits to acquire the lock per peer connection.
time-in-ms	The maximum time the IPC send thread that waits to acquire the lock in Milli-Seconds. .
duration-in-ms	The maximum duration the IPC send threads throttle the messages in MilliSeconds.
max-samples	The maximum number of SLA timeout samples to throttle the peer.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use this command to tune the IPC send thread parameters to handle the slow network peers.

Examples

The following is an example of tuning IPC send thread parameters.

```
(config)# dra ipc-send-thread limit 50 lock-sla-timeout 250 message-throttle-duration 30000
 timeout-sample-to-throttle 150
```

Note For more information, see the *IPC Queue Send Thread Tuning Configuration* section in the *CPS vDRA Advance Tuning Guide*.

end

Used to terminate a configuration session.

Syntax

```
end
```

Command Mode

CONFIG

VNFs

All

Command Usage

Use the end command to exit any configuration mode and return directly to operational mode. If you enter this command without committing the changes to the target configuration, you are prompted to do so:

```
Uncommitted changes found, commit them before exiting(yes/no/cancel)?[cancel]:
```

- Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns to the operational mode.
- If errors are found in the running configuration, the configuration session does not end. To view the errors, enter the `show configuration (config)` command with the failed keyword.
- Entering **no** exits the configuration session and returns to the operational mode without committing the configuration changes.
- Entering **cancel** leaves the CLI prompt in the current configuration session without exiting or committing the configuration changes.

Examples

The following is an example:

```
network dns host reladsdsydra1.client.3gppnetwork.org local address X:X::X:X
admin@orchestrator[scheduler](config-host-reladsdsydra1.client.3gppnetwork.org/local)# end
Uncommitted changes found, commit them? [yes/no/CANCEL]
```

external-aaa pam gid-mapping

Configures the gid mapping for various group roles.

Syntax

```
external-aaa pam gid-mapping <gid:int> <group name>
```

Command Parameters

Table 76: Parameter Description

Command Parameter	Description
gid:int	GID mapping value.
group name	Group name for which gid mapping is required.

Command Mode

CONFIG

VNFs

All

Command Usage

Use `external-aaa pam gid-mapping` to configure LDAP user gid mapping for various group roles such as, grafana-admin, policy-admin, policy-ro, and so on.

Based on the roles configured for the LDAP user gid, access permissions can be set accordingly.

Example

```
admin@orchestrator(config)# external-aaa pam gid-mapping 1000 policy-admin
admin@orchestrator(config-gid-mapping-1000/policy-admin)# commit
Commit complete.
```

You can display the status of configuration by running the following command:

```
admin@orchestrator# show running-config external-aaa | tab
```

Sample Output:

```
admin@orchestrator# show running-config external-aaa | tab
GID   GROUP
--------------------
1000  policy-admin
```

license feature

Registers a system license.

Syntax

```
license feature id encrypted-license encrypted-license
no license feature id
```

Command Parameters

Table 77: Parameter Description

Command Parameter	Description
id	ID of the license as provided by Cisco.
encrypted-license	The encrypted license as provided by Cisco.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the `license feature` to add and remove licenses from the running system.

load

Used to load configuration from file or terminal.

Syntax

```
load { merge | replace | override } { <file> | terminal }
```

Command Parameters

Table 78: Parameter Description

Command Parameter	Description
merge	Merge content of file/terminal with current configuration..
replace	Replace the content of file/terminal for the corresponding parts of the current configuration. In case of replace, the parts that are common in the file/terminal are replaced and rest of the configuration is not modified.

Command Parameter	Description
override	In case of override, the entire configuration is deleted (with the exception of hidden data) before loading the new configuration from the file/terminal.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the load command to merge/replace/override from file or terminal.

Examples

The configuration file can contain replace: and delete: directives. The following is an example:

```
system {
parent-mo {
child-mo 1 {
attr 10;
}
child-mo 2 {
attr 5;
}
}
}
```

If you want to delete child-mo 2, you can create a configuration file containing either:

- **replace:**

```
system {
replace:
parent-mo {
child-mo 1 {
attr 2;
}
}
}
```

- **delete:**

```
system {
parent-mo {
delete:
child-mo 2 {
attr 5;
}
}
}
```

logger set

Sets the various log levels for application logging.

Syntax

```
logger set logger-name { trace | debug | info | warn | error | off }
```

Command Parameters

Table 79: Parameter Description

Command Parameter	Description
logger-name	Name of the logger to enable at the given log level.
trace	Enables trace logging and higher.
debug	Enables debug logging and higher.
info	Enables info logging and higher.
warn	Enables warn logging and higher.
error	Enables error logging.
off	Turns off all logging for the logger.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `logger set` to enable various levels of application logging. The logger names are provided by Cisco per application and are not defined here.

Examples

The following is an example:

```
logger set com.broadhop debug
```

logger clear

Clears a log level defined using the `logger set` command.

Syntax

`logger clear logger-name`

Command Parameters

Table 80: Parameter Description

Command Parameter	Description
logger-name	Name of the logger to enable at the given log level.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `logger clear` to reset the logging level for an application logger to the default level. The current set of logger levels can be found using the `show logger level` command.

log collect config

Configures the destination server details for log collection.

Syntax

Log collect config ip *ip* **port** *port* **user** *user*

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the **log collect config** command to configure the destination server details. You can specify a password in the interactive mode.

If user is "cps", PEM file will be used.

If user is gtac or any other user, a correct password must be specified in the interactive mode.

The password is stored in an encrypted format and displayed in an encrypted format only.

Examples

The following sample output is an example for log collect feature configs.

```
log collect show
Log collect configurations    Current Value
---------------------------  --------------------------------
ip                           173.39.57.214
port                         22
user                         msivapra
password                     42A61FBF99537F7C972E7ADDC2BA453F
```

log collect all

Collects all required logs.

Syntax

log collect all
[duration *timeperiod in hours* **]**

Command Mode

DEBUG

VNFs

All

Command Usage

Use the **log collect all** to collect all required logs.

Internally output files are stored in the below location inside orchestrator container.

`/var/broadhop/fileserver/logs/`

The log files are copied to the DIM server and removed from the Master VM after copying is complete.

If the SCP to External server fails, the files remains in the `/var/broadhop/fileserver/logs/` in the orchestrator. You can copy or delete these log files manually.

If such files are not manually copied, they can be copied in the next run of the command and gets removed from the orchestrator.

log-forward fluentbit local-forward

Used to configure the OAM IP address (master, control-a or control-b) to forward all logs from other non-proxy VMs to the OAM VM/proxy VM before it is forwarded to external server.

Syntax

`log-forward fluentbit local-forward OAM-ip <OAM-ip> OAM-Port <OAM-port>`

Command Parameters

Table 81: Parameter Description

Command Parameter	Description
OAM-ip	The VIP to be set to enable local forwarding.
OAM-port	The port to be set for local forwarding.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `log-forward fluentbit local-forward` command to forward all logs locally before forwarding to external server.

Examples

The following is an example:

```
admin@orchestrator[site](config)# log-forward fluentbit local-forward
Value for 'OAM-ip' (<IP address>): x.x.x.x
Value for 'OAM-port' (<int>): 9200
status {
 data Starting Local Forwarding logs to : host x.x.x.x, port 9200
}
```

log-forward fluentbit elasticsearch

Used to configure the Elasticsearch details to forward all DRA logs. Make sure to use this command after **log-forward fluentbit local-forward OAM-ip***OAM-ip* **OAM-port** *OAM-port*

Syntax

```
log-forward fluentbit elasticsearch elastic-ip <elastic-ip> elastic-port
<elastic-port> elastic-user <elastic-user> elastic-password <elastic-password>
```

Command Parameters

Table 82: Parameter Description

Command Parameter	Description
elasticsearch IP	IP to be set to enable remote forwarding for elasticsearch server.
elasticsearch port	Port to be set to enable remote forwarding.

Command Parameter	Description
elastic user	User to be specified to enable remote forwarding.
elastic password	Password to be set to authenticate remote forwarding.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `log-forward fluentbit elasticsearch` command to configure the Elasticsearch details to forward all DRA logs.

Examples

The following is an example:

```
admin@orchestrator[site](config)# log-forward fluentbit elasticsearch
Value for 'elastic-ip' (<IP address>): x.x.x.x
Value for 'elastic-port' (<int>): 9400
Value for 'elastic-user' (<string>): elastic
Value for 'elastic-password' (<string>): ********
status {
 data Starting forwarding logs to elasticsearch: host x.x.x.x, port 9400 for elastic user
}
```

log-forward fluentbit filter

Used to set the filters to suppress any logs that should not be forwarded to elasticsearch.

Syntax

```
log-forward fluentbit filter key <Key-name> pattern <pattern-to-be-suppressed>
```

Command Parameters

Table 83: Parameter Description

Command Parameter	Description
key	Key name to be set for journalctl keys or log keys.
pattern	Pattern to be set for regex patterns of logs to be suppressed. **Note** Filter-regex: Regex expression to be parsed to apply filter configuration at OAM VM.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `log-forward fluentbit filter` command to set the filters to suppress any logs that should not be forwarded to elasticsearch.

Examples

The following is an example:

```
admin@orchestrator[site](config)# log-forward fluentbit filter
Value for 'key' (<string>): CONTAINER_NAME
Value for 'pattern' (<string>): orchestrator
status {
 data Filters Applied for Forwarding logs : Key CONTAINER_NAME, Pattern orchestrator
}
```

log-forward fluentbit filter-clear

Used to clear all filters.

Syntax

```
log-forward fluentbit filter-clear
```

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `log-forward fluentbit filter-clear` command to clear all filters.

Examples

The following is an example:

```
admin@orchestrator[site2-dra-master0](config)# log-forward fluentbit filter-clear
status {
 data Filters are cleared successfully
}
```

log-forward fluentbit tune

Used to configure tuning parameters.

Syntax

```
log-forward fluentbit tune flush_interval <flush_interval>
max_chunks_up <max_chunks_up> max_memory_backlog <max_memory_backlog>
```

Command Parameters

Table 84: Parameter Description

Command Parameter	Description
flush interval	Interval to be set for flushing logs to elastisearch.
max chunks up	The maximum chunks at the OAM, which should be active for getting flushed at one time.
max memory backlog	The maximum backlog size OAM server to keep the logs.

Command Mode

CONFIG

VNFs

DRA

Command Usage

Use the `log-forward fluentbit tune` command to tune paramters for flushing.

Examples

The following is an example:

```
admin@orchestrator[site](config)# log-forward fluentbit tune flush_interval 900 max_chunks_up
 3500 max_memory_backlog 2048M
status {
    data Tunings modified for remote forwarding
}
```

monitor log application

Tails the cluster wide application log.

Syntax

```
monitor log application
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use the `monitor log application` to tail the `consolidated-qns.log` running on the cc-monitor docker services. If the cc-monitor docker services are not running, this command will fail.

Examples

The following is an example:

```
scheduler# monitor log application
binding-s3.weave.local  2017-03-06 00:07:07,256 [LicenseManagerProxy] INFO
consolidated.sessions - TPS_COUNT:                              SESSION_COUNT:
              LICENSE_COUNT: 100000000
binding-s4.weave.local  2017-03-06 00:07:15,577 [LicenseManagerProxy] INFO
consolidated.sessions - TPS_COUNT:                              SESSION_COUNT:
              LICENSE_COUNT: 100000000
diameter-endpoint-s1.weave.local  2017-03-06 00:07:21,041 [LicenseManagerProxy] INFO
consolidated.sessions - TPS_COUNT:                              SESSION_COUNT:
```

monitor log container

Tails a specific docker container using the `monitor log container` command.

Syntax

`monitor log container` *`container-id`*

Command Parameters

Table 85: Parameter Description

Command Parameter	Description
container-id	The container's log file to monitor. Use the `show docker service` command to list the valid container-ids.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the `monitor log container` command to tail the docker container log. This will provide the output for all non-application messages for the given container.

Examples

The following is an example:

```
scheduler# monitor log container svn
<<< Started new transaction, based on original revision 94
     * editing path : __tmp_run_stage ... done.

------- Committed revision 94 >>>

<<< Started new transaction, based on original revision 95
     * editing path : __tmp_run_backup ... done.
```

monitor log engine

Tails the cluster wide engine log using the `monitor log engine` command.

Syntax

```
monitor log engine
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use the `monitor log engine` to tail the `consolidated-engine.log` running on the cc-monitor docker services. If the cc-monitor docker services are not running this command will fail.

monitor subscriber-activity

To view live monitor subscriber activity logs in the vDRA

Note The monitor subscriber-activity CLI is used only to view live logs and is not used to store/stop monitor logging activity.

Syntax

```
monitor subscriber-activity imsi <IMSI value> user <admin>
monitor subscriber-activity msisdn <MSISDN value> user <admin>
monitor subscriber-activity ipv6 <IPv6 value> user <admin>
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use this CLI command to view only monitor subscriber activity logs. Specify the Subscriber identity (IMSI/MSISDN/IPV6), DRA central username, and password to fetch live monitor logs from "monitor_activity_db" in admin-db for the subscriber.

Examples

The following is an example:

```
admin@orchestrator[master-00]# monitor subscriber-activity imsi 450005978851103 user admin
Enter host password for user 'admin':
```

nacm rule-list

Specifies access restrictions for a user group.

Verify the users in the group before applying restrictions. To specify restrictions for any group, ensure that the admin user is not part of that group. By default, admin user is configured in a each group.

Syntax

```
nacm rule-list <rule-name> group <group-name> cmdrule <cmdrule-name> command <command to restrict> access-operations exec action deny
```

Command Parameters

Table 86: Parameter Description

Command Parameter	Description
rule-list	Name of rule list.
group	Name of the group or list of groups to which the rules apply.
command	Command that is restricted for the user group.
access-operations	Used to match the operation that ConfD tries to perform. It must be one or more of the values from the accessoperations-type: create, read, update, delete, exec

Command Parameter	Description
action	If all of the previous fields match, the rule as a whole matches and the value of action (permit or deny) is taken. If a match is found, a decision is made whether to permit or deny the request in its entirety. If action is permit, the request is permitted; if action is deny, the request is denied.

Command Mode

CONFIG

VNFs

All

Command Usage

To delete the admin user from the read-only group, use the following command:

```
scheduler(config)#no nacm groups group crd-read-only user-name admin
```

For the configuration to take effect, log out of the CLI session and log in again after configuring any nacm rule-list.

Examples

Restrict crd-read-only group from config command:

```
scheduler(config)#nacm rule-list crdreadgrp group crd-read-only cmdrule denyconfig command
 config access-operations exec action deny
scheduler(config-cmdrule-denyconfig)# commit
```

Restrict crd-read-only and policy-ro group from config command:

```
scheduler(config)#nacm rule-list readonly-restrict group [ crd-read-only policy-ro ] cmdrule
 cfg-restrict command config access-operations exec action deny
scheduler(config-cmdrule-cfg-restrict)#commit
```

Restrict crd-read-only and policy-ro group from docker command:

```
scheduler(config)#nacm rule-list readonly-restrict group [ crd-read-only policy-ro ] cmdrule
 docker-restrict command docker access-operations exec action deny
scheduler(config-cmdrule-docker-restrict)# commit
```

Restrict crd-read-only and policy-ro group from system stop command:

```
scheduler(config)#nacm rule-list readonly-restrict group [ crd-read-only policy-ro ] cmdrule
 sys-stop command "system stop" access-operations exec action deny
scheduler(config-cmdrule-sys-stop)# commit
```

Restrict crd-read-only and policy-ro group from system start command:

```
scheduler(config)#nacm rule-list readonly-restrict group [ crd-read-only policy-ro ] cmdrule
 sys-start command "system start" access-operations exec action deny
scheduler(config-cmdrule-sys-start)# commit
```

Restrict load override command for all the users including admin:

```
scheduler(config)#nacm rule-list readonly-restrict group [ * ] cmdrule load-override command
 "load override" access-operations exec action deny
scheduler(config-cmdrule-load-override)# commit
```

network dns server

Adds a network DNS server for the cluster to use.

Syntax

```
network dns server address
```

```
no network dns server address
```

Command Parameters

Table 87: Parameter Description

Command Parameter	Description
address	The IP address of the DNS server that the cluster can use. **Note** This address must be available to all servers within the cluster and is generally on an OAM network or the internal network.

Command Mode

CONFIG

VNFs

All

Command Usage

The network DNS server command triggers the addition of a DNS server to the DNS resolution that the application utilizes. These servers are added in the order they appear in the configuration to the DNS resolution.

Examples

The following example adds a DNS server:

```
scheduler(config)# network dns server 10.10.10.10
```

network dns host

Adds a network host to IP address mapping for the cluster to use.

Syntax

```
network dns host host domain address address
no network dns host host domain
```

Command Parameters

Table 88: Parameter Description

Command Parameter	Description
host	The host name of the host mapping to store.
domain	The domain name of the host mapping to store. Use local for hosts that do not have a domain name.
address	The IP address of the host / domain name mapping. **Note** Local address must not be used from the pool 172.17.0.0/16. This IP address set is dedicated to docker containers.

Command Mode

CONFIG

VNFs

All

Command Usage

The network DNS host command triggers the addition of a host / domain mapping to a specific IP address. This is useful when the upstream DNS services do not have a host / domain name mapping or upstream DNS server is not available to the cluster.

Examples

The following example adds a DNS server:

```
scheduler(config)# network dns host test local address 10.10.10.10
```

network virtual-service

Used to configure virtual floating IP address on various interfaces.

Syntax

```
network virtual-service name of floating ip floating-ip floating ip address mask net mask
digits broadcast broadcast address interface interface-id virtual-router-id virtual router
 id tracking-service prefix of service to monitor for IP address diameter-endpoint host ip
 address of host to put the floating ip priority priority of host
exit
```

```
host ip address of host to put the floating ip priority priority of host
commit
end
```

Command Parameters

Table 89: Parameter Description

Command Parameter	Description
name of floating ip	Name of the floating IP address. to be configured Virtual Network Service Name must contain a minimum of 1 character and a maximum length of 8 characters.
floating ip address	The floating IP address to manage with the virtual service.
net mask digits	The network mask (digits) for the floating IP address. Default: 24
broadcast address	The broadcast address of the floating IP.
interface-id	Interface ID.
virtual router id	virtual-router-id is the identity for a virtual router for hosts that are managed for VIP. Value range is from 0 to 255. For more details, refer to VRRP (Virtual Router Redundancy Protocol) RFC 3768 and keepalive documentation.
prefix of service to monitor for IP address	This parameter is a string used to define the service to be monitored.
ip address of host to put the floating ip	IP address of the host where floating IP is hosted.
priority of host	Priority of the host on which the service must run. Priority range is from 1 to 255. Higher the value, higher is the priority.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the `network virtual-service` command to configure virtual floating IP address on various interfaces that is managed using keepalive and the VRRP protocol. This command should be used in conjunction with the network virtual-service host command to assign floating IPs to given hosts.

Note To use within OpenStack, you must enable Protocol 112 on the security group – this is the VRRP protocol used by Keepalive. VRRP is configured as protocol number and not name. Hence, while configuring from dashboard, select protocol as 'Other' and in the text box below, enter 112 as protocol.

Examples

The following example creates a floating IP on two hosts:

Note Enter the command manually.

IPv4 VIP config:

```
scheduler(config)# network virtual-service GxVip12 floating-ip 172.22.33.51 mask 24 broadcast
 172.22.33.255 interface ens161 virtual-router-id 1 tracking-service diameter-endpoint host
 172.22.33.43 priority 2
exit
host 172.22.33.44 priority 1
commit
end
```

IPv6 VIP config:

```
scheduler(config)# network virtual-service RxVip12 floating-ip 2003:2235::51 mask 64 interface
 ens192 virtual-router-id 2 tracking-service diameter-endpoint host 2003:2235::44 priority
 2
exit
host 2003:2235::43 priority 1
commit
end
```

You can check the status of configuration on the scheduler by running the following command:

```
show running-config network
```

Sample Output:

```
network virtual-service GxVip12
 virtual-router-id 1
 floating-ip       172.22.33.51
 mask              24
 broadcast         172.22.33.255
 host 172.22.33.43
  priority 2
 !
 host 172.22.33.44
  priority 1
 !
!
```

Requirement

As a part of OpenStack configuration to have allowed-address-pairs configured on the VMs that are going to host the VIP.

Here is an example for ESC:

Under **vm_group** > **interfaces** > **interface**, you need to add the following configuration:

```
<allowed_address_pairs>
  <address>
    <ip_address>10.81.70.44</ip_address>
    <netmask>255.255.255.0</netmask>
  </address>
</allowed_address_pairs>
```

Note The above mentioned configuration needs to be done on all the interfaces of all the VMs where you want a virtual IP.

network virtual-service name host

Adds a new virtual-service floating IP address to the system.

Syntax

```
network virtual-service name host address priority priority
no network virtual-service name host address
```

Command Parameters

Table 90: Parameter Description

Command Parameter	Description
name	The logical name of the virtual service floating IP. Virtual Network Service Name must contain a minimum of 1 character and a maximum length of 8 characters.
address	The IP of the host that should manage this floating IP.
priority	The priority of the host relative other hosts within the group. Default: 100

Command Mode

CONFIG

VNFs

All

Command Usage

Use this command to add new hosts to a virtual service. The hosts added will be start a Keepalive process to manage the floating IP via the VRRP process.

Examples

The following example adds a floating IP on a host:

```
scheduler(config)# network virtual-service test host 10.84.100.136 priority 100
```

ntp server

Creates an NTP server for the system to synchronize system clocks.

Syntax

```
ntp server name address address
```

Command Parameters

Table 91: Parameter Description

Command Parameter	Description
name	Name of the server.
address	IP address or FQDN of the NTP server.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the `ntp server` command to synchronize the clocks of each virtual machine within the cluster. When this command is used, each node will run an NTP service. The NTP service is either a client or relay as described below:

- A relay node is a node that can reach at least one of the NTP servers defined in the configuration. The relay nodes are configured to point to the ntp servers defined in the server.
- A client node is an internal node that cannot reach an NTP server. The client nodes are configured to point to the relay nodes.

Examples

The following is an example:

```
scheduler(config)# ntp server server1 address 10.10.10.10
```

revert

Used to copy running configuration into current configuration.

Syntax

```
revert
```

Command Mode

CONFIG

VNFs

All

Command Usage

Use the revert command to copy running configuration into the current configuration.

Examples

The following is an example:

```
admin@orchestrator[an-master](config)#revert
```

rollback configuration

Used to rollback the running configuration to a previous configuration.

Syntax

```
rollback configuration <commit-id>
```

Command Mode

CONFIG

VNFs

All

Command Usage

- Each time the commit command is entered, a commit ID is assigned to the new configuration. You can revert the system to the configuration of a previous commit ID with the rollback configuration command.

- The system stores a limited number of old configurations. The number of old configurations to store is configured in the `confd.conf` file. If more configurations are stored than the configured number, then the oldest configuration is removed before creating a new one.
- The most recently committed configuration (the running configuration) is number 0, the next most recent 1, and so on.
- The files are called rollback0 - rollbackX, where X is the maximum number of saved committed configurations.
- Use `show configuration commit list` to display a list of the commit IDs available for rollback operations.

```
show configuration commit list
2018-10-15 09:58:21
SNo. ID       User       Client      Time Stamp          Label       Comment
~~~~ ~~       ~~~~       ~~~~~~      ~~~~~~~~~~          ~~~~~       ~~~~~~~
0    10012    admin      cli         2018-10-15 09:57:59
```

Examples

The following is an example:

```
rollback configuration 0
```

scheduling external-service

Creates a docker service that is external to the installed application.

Syntax

```
scheduling external-service name image image cap-add cap-add environment environment
host-network { true | false } port-mapping port-mapping run-level run-level scalable { true
 | false } scheduling-slot scheduling-slot volume volume
```

Command Parameters

Table 92: Parameter Description

Command Parameter	Description
name	Name of the service
image	Fully qualified image name.
scalable (optional)	Scale multiple instances across hosts. Default is false.
run-level (optional)	Relative run level between external services. Default is 0.

Command Parameter	Description
host-network (optional)	Bind to the host network. Default is to the overlay network.
volume (optional)	Volume mounts in the format is as follows: <host path>:<docker path>. Additional mounts are separated by ",".
port-mapping (optional)	Port mapping of the format is as follows: <external>:<internal>. Additional mounts are separated by ",".
cap-add (optional)	Linux capabilities to add to the container. Additional mounts are separated by ",".
scheduling-slot (optional)	Scheduling slot to start the container (for all containers). Use the **show running-config docker engine** command to view list of scheduling slots.
environment (optional)	Environment variables to export into the container in the format given below: <KEY>=<VALUE> Additional mounts are separated by ",".

Command Mode

CONFIG

VNFs

All

Command Usage

The `scheduling external-service` instructs the scheduling application to run the defined docker image on the given scheduling slots based on the configuration defined. Once scheduled the external-service appears in the `show scheduling status` and the `show docker service` commands.

scheduling vm-target

Calculates a vm-target for an external scaling system.

Syntax

```
scheduling vm-target name group-size group-size k k max max min min override override query
 query scale-up-threshold scale-up-threshold
```

```
no scheduling vm-target name
```

Command Parameters

Table 93: Parameter Description

Command Parameter	Description
name	Name or identifier for the vm-target rule.
group-size (optional)	Size of the scaling group. Default is one
k (optional)	K value in an n + k redundancy model. Default is one.
max (optional)	Maximum value to calculate for the vm-target.
min (optional)	Minimum value to calculate for the vm-target.
override (optional)	Override value for the vm-target. This overrides anything the equation would calculate.
query	Query to calculate a raw scaling value.
scale-up-threshold	Divisor when calculating the scaling number. The query's raw value is divided by the scale-up-threshold to get a the value of n in an n+k redundancy model.

Command Mode

CONFIG

VNFs

All

Command Usage

The `scheduling vm-target` instructs the system to calculate VM scaling targets which can be used by the system to add and remove scaling VMs as required. The following algorithm is used to calculate the VM target for a given “name”:

vm-target(name) = roundup ((query value) / (scale-up-threshold))*group-size+K

show alert status

Displays the status of all alerts in the system. It displays either all alert statuses or alerts for a specific named alert.

Syntax

`show alert status` *`rule-name`*

Command Parameters

Table 94: Parameter Description

Command Parameter	Description
rule-name (optional)	Displays alert statuses for a given rule-name.

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show scheduling status
                              OUT
                 RUN          OF
MODULE INSTANCE LEVEL STATE DATE
-------------------------------------------------------
consul 1         50   RUNNING false
admin-db 1       75 RUNNING false
memcached-vip 1 100 RUNNING false
prometheus 1     100 RUNNING false
prometheus 2     100 RUNNING false
prometheus 3     100 RUNNING false
```

Table 95: Parameter Description

Parameter	Description
Name	Rule-name of the alert.
Event Host	Host where the alert was generated.
Status	Status of the alert. Valid values are: • firing • resolved
Message	Current alert message.
Update Time	Timestamp of the first alert message that transitioned to the given status.

show configuration

Used to display information about the current configuration session changes.

Syntax

```
show configuration
```

Command Mode

CONFIG

VNFs

All

Command Usage

- To display the configuration changes compared to the running configuration if any.
- Possible to display the configuration changes based on configuration component.

Examples

The following is an example:

```
admin@orchestrator[an-master](config)# aaa authentication users user test1 password ****
gid 100 homedir / ssh_keydir / uid 9340
admin@orchestrator[an-master](config-user-test1)#
admin@orchestrator[an-master](config)# show configuration
aaa authentication users user test1
 uid        9340
 gid        100
 password   $1$AWYdJW5S$g2wXilsJSumbCXPYgGzQW0
 ssh_keydir /
 homedir    /
!
```

show configuration commit

Used to display the changes made to the running configuration by previous configuration commits, a configuration commit, or for a range of configuration commits.

Use the `show configuration commit changes` command to display the information about the current configuration session changes.

Syntax

```
show configuration commit changes
```

```
show configuration commit list
```

Command Mode

CONFIG

VNFs

All

Command Usage

- Each time a configuration is committed with the commit command, the configuration commit operation is assigned a commit ID. The `show configuration commit changes` command displays the configuration changes made since the specified commit.
- To display a list of the available commit IDs, enter the `show configuration commit list` command.

Examples

The following is an example:

```
show configuration commit changes
!
! Created by: admin
! Date: 2018-10-15 09:57:59
! Client: cli
!
aaa authentication users user anil
 uid        9340
 gid        100
 password   $1$7aB1WW0D$3ln7YEGkLeTjWHoK2cVOE/
 ssh_keydir /
 homedir    /
!
```

```
show configuration commit list
2018-10-15 11:20:39
SNo. ID       User       Client      Time Stamp          Label       Comment
~~~~ ~~       ~~~~       ~~~~~~      ~~~~~~~~~~          ~~~~~       ~~~~~~~
0    10012    admin      cli         2018-10-15 09:57:59
```

show configuration rollback

Used to display changes that are made by the rollback configuration command. To display the list of rollback commit IDs, use the `show configuration rollback changes` command.

Syntax

```
show configuration rollback changes
```

Command Mode

ALL

VNFs

All

Command Usage

Use `show configuration rollback changes` command to display changes that are made by the rollback configuration command.

Note The most recent commits are retained by the system. As new commit IDs are added, the oldest commit IDs are discarded and are no longer available for rollback operations.

Examples

The following is an example:

```
show configuration rollback changes 0
no aaa authentication users user test1
```

show control-plane remote-peer-policy

Used to display the configured control plane remote peer policy.

Syntax

```
show control-plane remote-peer-policy
```

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

This command is used to display the current configured control plane remote peer policy in DRA.

Example

```
admin@orchestrator[vpas-A-dra-master-0]# show control-plane remote-peer-policy
Mated System: system-02
Accept remote peers for diameter applications : All
All Systems:
Accept remote peers for diameter applications :  Rx
```

show database

`show database status` displays the currently configured database clusters members.

`show database parallel-upgrade-plan` is used to print the parallel upgrade plan appropriate for database shard layout across nodes. If parallel upgrade option is selected, all the nodes in a batch are upgraded in parallel.

`show database parallel-upgrade-plan-details` is used to print the parallel upgrade plan with details of shards and servers selected in each batch. Orchestrator ensures that the members of the same shard are scheduled

in different batches to minimize the impact on the shards during parallel upgrade. You can use this command to review the plan and assess the impact of performing a parallel upgrade of DB cluster.

Syntax

```
show database status
show database parallel-upgrade-plan
show database parallel-upgrade-plan-details
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show database status
                                                    CLUSTER
ADDRESS       PORT   NAME    STATUS     TYPE         NAME  SHARD   REPLICA SET
-------------------------------------------------------------------------------------
192.168.65.2  27018  shardA  PRIMARY    replica_set   test  shardA  rs-shardA
192.168.65.2  27019  -       PRIMARY    config_server test  cfg     test-configsrv
192.168.65.2  27017  -       CONNECTED  mongos        test  router-1 test-configsrv
```

Table 96: Output Description

Command Parameter	Description
Address	The address of the database process.
Port	The port the database service is running.
Name	Name of the database process.

Command Parameter	Description
Status	The current status of the mongo process. Valid states are: • CONNECTED – The mongo router is connected to the config servers • NOT_CONNECTED – The mongo router is not connected to the config servers • NO_CONNECTION – The process is not up or is not monitored • STARTUP – The DB node is in the STARTUP mode • PRIMARY – The DB node is the current PRIMARY • SECONDARY – The DB node is a SECONDARY node • RECOVERING – The DB node is currently RECOVERING from a restart or other failure • STARTUP2 – The DB node is in STARTUP2 mode • UNKNOWN – The DB node is in an UNKNOWN state • ARBITER – The DB node is currently an active ARBITER • NOT_INITIALIZED – The DB node is not initialized and pending initialization
Type	The type of the mongo process. Valid values are: • replica_set – a member of the replica set • config_server – a member of the config server replica set • mongos – a mongo router process
Cluster Name	The name of the cluster that owns the process.
Shard	The name of the associated shard.
Replica Set	The name of the replica set associated to the process.

To print the parallel upgrade plan appropriate for database shard layout across nodes.

```
admin@orchestrator[master-6]# show database parallel-upgrade-plan
BATCH  MODULE          HOST              ADDRESS
--------------------------------------------------------------------------------
```

```
1      mongo-node-101  master-6          172.20.27.36
1      mongo-node-102  control-7         172.20.27.40
1      mongo-node-103  control-8         172.20.27.39
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54
```

To print the parallel upgrade plan with details of shards and servers selected in each batch.

```
admin@orchestrator[master-6]# show database parallel-upgrade-plan-details
BATCH  MODULE          HOST              ADDRESS                              PORT   CLUSTER
 NAME      SHARD    SERVER           STATUS
-------------------------------------------------------------------------------------------
1      mongo-node-101  master-6          172.20.27.36                         27019
imsi-msisdn      shdb-3   imsi-msisdn      SECONDARY
1      mongo-node-102  control-7         172.20.27.40                         27019
session-ipv6-AB  shdb-3   session-ipv6-AB  SECONDARY
1      mongo-node-103  control-8         172.20.27.39                         27019
imsi-msisdn      shdb-2   imsi-msisdn      SECONDARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27017
imsi-msisdn      shard-1  server-c         SECONDARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27018
imsi-msisdn      shard-2  server-c         SECONDARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27021
imsi-msisdn      shard-3  server-c         PRIMARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27020
imsi-msisdn      shard-4  server-c         SECONDARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27022
session-ipv6-AB  shard-5  server-c         SECONDARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27023
session-ipv6-AB  shard-6  server-c         SECONDARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27024
session-ipv6-AB  shard-7  server-c         PRIMARY
1      mongo-node-104  persistence-db-3  2003:3030:27c1:913:250:56ff:fea6:53  27025
session-ipv6-AB  shard-8  server-c         SECONDARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27017
imsi-msisdn      shard-1  server-d         SECONDARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27018
imsi-msisdn      shard-2  server-d         SECONDARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27021
imsi-msisdn      shard-3  server-d         SECONDARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27020
imsi-msisdn      shard-4  server-d         PRIMARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27022
session-ipv6-AB  shard-5  server-d         SECONDARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27023
session-ipv6-AB  shard-6  server-d         SECONDARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27024
session-ipv6-AB  shard-7  server-d         SECONDARY
2      mongo-node-105  persistence-db-4  2003:3030:27c1:913:250:56ff:fea6:54  27025
session-ipv6-AB  shard-8  server-d         PRIMARY
```

show docker engine

Displays the status of the clusters docker engines.

Syntax

```
show docker engine
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show docker engine
                                      MISSED
ID                        STATUS      PINGS
------------------------------------------
binding-73d3dc            CONNECTED   0
binding-8a8d17            CONNECTED   0
binding-c74547            CONNECTED   0
binding-dabba5            CONNECTED   0
control-0                 CONNECTED   0
control-1                 CONNECTED   0
control-2                 CONNECTED   0
diameter-endpoint-0       CONNECTED   0
diameter-endpoint-1       CONNECTED   0
diameter-endpoint-2       CONNECTED   0
diameter-endpoint-3       CONNECTED   0
master-0                  CONNECTED   0
session-shard-1-e079cf    CONNECTED   0
session-shard-2-80941f    CONNECTED   0
```

Table 97: Parameter Description

Parameter	Description
ID	The identifier within the cluster of the docker engine. Generally, this maps to the hostname where the engine resides.
Status	Indicates if the scheduling application is connected to the docker engine running on a host.
Missed Pings	The number of consecutive missed pings for a given host.

show docker service

Displays the currently running docker services.

Syntax

```
show docker service
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show docker service
MODULE   INSTANCE  NAME           VERSION              ENGINE         CONTAINER ID
STATE   MESSAGE PENALTY BOX
-------------------------------------------------------------------------------------------------------
admin-db    1     mongo-admin-a      3.4.0.0          control-0       mongo-admin-a
HEALTHY  false     -
admin-db    1     mongo-admin-arb    3.4.0.0          master-0        mongo-admin-arb
HEALTHY  false     -
admin-db    1     mongo-admin-b      3.4.0.0          control-1       mongo-admin-b
HEALTHY  false     -
admin-db    1     mongo-admin-setup  12.9.9-2017      master-0        mongo-admin-setup
HEALTHY  false     -
                                -03-03.123.797af71
binding     1     binding            12.9.9-dra.2017  binding-73d3dc  binding-s1
HEALTHY  false     -
                                -03-03.115.0f485ef
binding     1     session-router     3.4.0.0          binding-73d3dc  session-router-s1
HEALTHY  false     -
binding     2     binding            12.9.9-dra.2017  binding-8a8d17  binding-s2
HEALTHY  false
                                -03-03.115.0f485ef
```

Table 98: Parameter Description

Parameter	Description
Module	Scheduling module that is executing the docker service.
Instance	For scalable modules, the instance number that the service relates.
Name	Logical name of the service.
Version	Version of the image executing.
Engine	Engine identifier that is executing the docker service.
Container ID	Container id of the docker service.
State	Current state of the docker service.
Penalty Box	Indicates if the service is waiting to be rescheduled if an error occurred.
Message	Message related to the penalty box designation.

show dra-distributor

Displays the output of ipvsadm (Virtual Server administration) from all distributor VMs.

Syntax

```
show dra-distributor [ daemon | list | rate | stats ]
```

Command Parameters

Table 99: Parameter Description

Command Parameter	Description
daemon	Displays the sync daemon status and multicast interface.
list	Lists the Distributor Service table.
rate	Displays rate information for connection, bytes, and packets per second of Distributor services.
stats	Displays statistic information of Distributor Services.

Command Mode

OPERATIONAL

VNFs

All

Examples

The following are examples:

```
show dra-distributor list
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-a
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.16.241.10:3868 wlc
  -> 172.16.241.3:3868            Route   1      6          0
  -> 172.16.241.4:3868            Route   1      7          0
  -> 172.16.241.5:3868            Route   1      6          0
  -> 172.16.241.6:3868            Route   1      6          0
TCP  172.16.241.74:3868 wlc
  -> 172.16.241.67:3868           Route   1      1          0
  -> 172.16.241.68:3868           Route   1      1          0
  -> 172.16.241.69:3868           Route   1      1          0
  -> 172.16.241.70:3868           Route   1      1          0
TCP  [2606:ae00:3001:8311:172:16:241:109]:3868 wlc
  -> [2606:ae00:3001:8311:172:16:241:102]:3868 Route   1      5          0
  -> [2606:ae00:3001:8311:172:16:241:103]:3868 Route   1      5          0
  -> [2606:ae00:3001:8311:172:16:241:104]:3868 Route   1      5          0
  -> [2606:ae00:3001:8311:172:16:241:105]:3868 Route   1      9          0
```

```
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-b
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.16.241.10:3868 wlc
  -> 172.16.241.3:3868            Route   1      6          0
  -> 172.16.241.4:3868            Route   1      7          0
  -> 172.16.241.5:3868            Route   1      6          0
  -> 172.16.241.6:3868            Route   1      6          0
TCP  172.16.241.74:3868 wlc
  -> 172.16.241.67:3868           Route   1      1          0
  -> 172.16.241.68:3868           Route   1      1          0
  -> 172.16.241.69:3868           Route   1      1          0
  -> 172.16.241.70:3868           Route   1      1          0
TCP  [2606:ae00:3001:8311:172:16:241:109]:3868 wlc
  -> [2606:ae00:3001:8311:172:16:241:102]:3868 Route   1      5          0
  -> [2606:ae00:3001:8311:172:16:241:103]:3868 Route   1      5          0
  -> [2606:ae00:3001:8311:172:16:241:104]:3868 Route   1      5          0
  -> [2606:ae00:3001:8311:172:16:241:105]:3868 Route   1      9          0
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-a
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.16.242.10:3868 wlc
  -> 172.16.242.3:3868            Route   1      3          0
  -> 172.16.242.4:3868            Route   1      3          0
  -> 172.16.242.5:3868            Route   1      3          0
  -> 172.16.242.6:3868            Route   1      3          0
TCP  172.16.242.138:3868 wlc
  -> 172.16.242.131:3868          Route   1      5          0
  -> 172.16.242.132:3868          Route   1      4          0
  -> 172.16.242.133:3868          Route   1      4          0
  -> 172.16.242.134:3868          Route   1      4          0
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-b
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port           Forward Weight ActiveConn InActConn
TCP  172.16.242.10:3868 wlc
  -> 172.16.242.3:3868            Route   1      3          0
  -> 172.16.242.4:3868            Route   1      3          0
  -> 172.16.242.5:3868            Route   1      3          0
  -> 172.16.242.6:3868            Route   1      3          0
TCP  172.16.242.138:3868 wlc
  -> 172.16.242.131:3868          Route   1      5          0
  -> 172.16.242.132:3868          Route   1      4          0
  -> 172.16.242.133:3868          Route   1      4          0
  -> 172.16.242.134:3868          Route   1      4          0
```

```
show dra-distributor daemon
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-a
backup sync daemon (mcast=ens160, syncid=201)
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-b
backup sync daemon (mcast=ens160, syncid=201)
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-a
backup sync daemon (mcast=ens160, syncid=202)
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-b
backup sync daemon (mcast=ens160, syncid=202)
```

```
show dra-distributor rate
==============================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-a
```

```
Prot LocalAddress:Port                  CPS    InPPS   OutPPS    InBPS   OutBPS
  -> RemoteAddress:Port
TCP  172.16.241.10:3868                   0    35080        0 17784626        0
  -> 172.16.241.3:3868                    0     7753        0  3856300        0
  -> 172.16.241.4:3868                    0     9521        0  4867718        0
  -> 172.16.241.5:3868                    0     7249        0  3607296        0
  -> 172.16.241.6:3868                    0    10557        0  5453342        0
TCP  172.16.241.74:3868                   0     2896        0  1269265        0
  -> 172.16.241.67:3868                   0      740        0   317735        0
  -> 172.16.241.68:3868                   0      824        0   321847        0
  -> 172.16.241.69:3868                   0      550        0   309638        0
  -> 172.16.241.70:3868                   0      782        0   320007        0
TCP  [2606:ae00:3001:8311:172:16:241:109]:3868        0    18551        0 17887169
0
  -> [2606:ae00:3001:8311:172:16:241:102]:3868        0     3714        0  3581895
0
  -> [2606:ae00:3001:8311:172:16:241:103]:3868        0     4037        0  3878454
0
  -> [2606:ae00:3001:8311:172:16:241:104]:3868        0     4012        0  3877344
0
  -> [2606:ae00:3001:8311:172:16:241:105]:3868        0     6789        0  6549476
0
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-b
Prot LocalAddress:Port                  CPS    InPPS   OutPPS    InBPS   OutBPS
  -> RemoteAddress:Port
TCP  172.16.241.10:3868                   0        0        0        0        0
  -> 172.16.241.3:3868                    0        0        0        0        0
  -> 172.16.241.4:3868                    0        0        0        0        0
  -> 172.16.241.5:3868                    0        0        0        0        0
  -> 172.16.241.6:3868                    0        0        0        0        0
TCP  172.16.241.74:3868                   0        0        0        0        0
  -> 172.16.241.67:3868                   0        0        0        0        0
  -> 172.16.241.68:3868                   0        0        0        0        0
  -> 172.16.241.69:3868                   0        0        0        0        0
  -> 172.16.241.70:3868                   0        0        0        0        0
TCP  [2606:ae00:3001:8311:172:16:241:109]:3868        0        0        0        0
0
  -> [2606:ae00:3001:8311:172:16:241:102]:3868        0        0        0        0
0
  -> [2606:ae00:3001:8311:172:16:241:103]:3868        0        0        0        0
0
  -> [2606:ae00:3001:8311:172:16:241:104]:3868        0        0        0        0
0
  -> [2606:ae00:3001:8311:172:16:241:105]:3868        0        0        0        0
0
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-a
Prot LocalAddress:Port                  CPS    InPPS   OutPPS    InBPS   OutBPS
  -> RemoteAddress:Port
TCP  172.16.242.10:3868                   0    29969        0 19567201        0
  -> 172.16.242.3:3868                    0     7363        0  4884850        0
  -> 172.16.242.4:3868                    0     7435        0  4885241        0
  -> 172.16.242.5:3868                    0     7636        0  4911014        0
  -> 172.16.242.6:3868                    0     7534        0  4886099        0
TCP  172.16.242.138:3868                  0    24373        0  8103149        0
  -> 172.16.242.131:3868                  0     5940        0  1677292        0
  -> 172.16.242.132:3868                  0     8316        0  3543717        0
  -> 172.16.242.133:3868                  0     4823        0  1429692        0
  -> 172.16.242.134:3868                  0     5293        0  1452448        0
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-b
Prot LocalAddress:Port                  CPS    InPPS   OutPPS    InBPS   OutBPS
  -> RemoteAddress:Port
```

```
TCP  172.16.242.10:3868                  0        0        0        0        0
  -> 172.16.242.3:3868                   0        0        0        0        0
  -> 172.16.242.4:3868                   0        0        0        0        0
  -> 172.16.242.5:3868                   0        0        0        0        0
  -> 172.16.242.6:3868                   0        0        0        0        0
TCP  172.16.242.138:3868                 0        0        0        0        0
  -> 172.16.242.131:3868                 0        0        0        0        0
  -> 172.16.242.132:3868                 0        0        0        0        0
  -> 172.16.242.133:3868                 0        0        0        0        0
  -> 172.16.242.134:3868                 0        0        0        0        0
```

```
show dra-distributor stats
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-a
Prot LocalAddress:Port               Conns   InPkts  OutPkts  InBytes OutBytes
  -> RemoteAddress:Port
TCP  172.16.241.10:3868                  5  130888K        0   67428M        0
  -> 172.16.241.3:3868                   1 28763786        0   14532M        0
  -> 172.16.241.4:3868                   2 34872671        0   17887M        0
  -> 172.16.241.5:3868                   1 26758954        0   13554M        0
  -> 172.16.241.6:3868                   1 37715757        0   19818M        0
TCP  172.16.241.74:3868                  1  9892533        0    4791M        0
  -> 172.16.241.67:3868                  0  2535586        0    1206M        0
  -> 172.16.241.68:3868                  0  2627786        0    1208M        0
  -> 172.16.241.69:3868                  1  1940733        0    1058M        0
  -> 172.16.241.70:3868                  0  2578653        0    1208M        0
TCP  [2606:ae00:3001:8311:172:16:241:109]:3868        5 70270305        0   68098M
0
  -> [2606:ae00:3001:8311:172:16:241:102]:3868        0 14039247        0   13718M
0
  -> [2606:ae00:3001:8311:172:16:241:103]:3868        0 15233935        0   14707M
0
  -> [2606:ae00:3001:8311:172:16:241:104]:3868        0 15271681        0   14903M
0
  -> [2606:ae00:3001:8311:172:16:241:105]:3868        5 24425635        0   23490M
0
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-client-b
Prot LocalAddress:Port               Conns   InPkts  OutPkts  InBytes OutBytes
  -> RemoteAddress:Port
TCP  172.16.241.10:3868                 25    3046M        0    1577G        0
  -> 172.16.241.3:3868                   5  575759K        0     295G        0
  -> 172.16.241.4:3868                   5  568825K        0     288G        0
  -> 172.16.241.5:3868                   5  564563K        0     282G        0
  -> 172.16.241.6:3868                   5  534960K        0     273G        0
TCP  172.16.241.74:3868                  4  172396K        0   83111M        0
  -> 172.16.241.67:3868                  1 41803986        0   20709M        0
  -> 172.16.241.68:3868                  1 45090996        0   20883M        0
  -> 172.16.241.69:3868                  0       11        0    10472        0
  -> 172.16.241.70:3868                  1 41847316        0   20711M        0
TCP  [2606:ae00:3001:8311:172:16:241:109]:3868       24    1635M        0    1581G
0
  -> [2606:ae00:3001:8311:172:16:241:102]:3868        5  306946K        0     298G
0
  -> [2606:ae00:3001:8311:172:16:241:103]:3868        5  357153K        0     342G
0
  -> [2606:ae00:3001:8311:172:16:241:104]:3868        5  339724K        0     332G
0
  -> [2606:ae00:3001:8311:172:16:241:105]:3868        4  300748K        0     290G
0
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-a
Prot LocalAddress:Port               Conns   InPkts  OutPkts  InBytes OutBytes
  -> RemoteAddress:Port
TCP  172.16.242.10:3868                 38  401155K        0    3050G        0
```

```
  -> 172.16.242.3:3868                   7  782570K        0     508G        0
  -> 172.16.242.4:3868                   7  789557K        0     513G        0
  -> 172.16.242.5:3868                   7  979702K        0     623G        0
  -> 172.16.242.6:3868                   8    1166M        0     759G        0
TCP  172.16.242.138:3868                55    3724M        0    1230G        0
  -> 172.16.242.131:3868                11  579165K        0     159G        0
  -> 172.16.242.132:3868                12    1079M        0     418G        0
  -> 172.16.242.133:3868                11  843389K        0     306G        0
  -> 172.16.242.134:3868                10  557000K        0     156G        0
================================================================================
    dra-distributor stats for  vpas-A-dra-distributor-server-b
Prot LocalAddress:Port               Conns   InPkts  OutPkts  InBytes OutBytes
  -> RemoteAddress:Port
TCP  172.16.242.10:3868                  0        0        0        0        0
  -> 172.16.242.3:3868                   0      785        0  1149048        0
  -> 172.16.242.4:3868                   0      817        0    43388        0
  -> 172.16.242.5:3868                   0     1178        0  2029844        0
  -> 172.16.242.6:3868                   0     2069        0  2386744        0
TCP  172.16.242.138:3868                 0    10926        0  3854392        0
  -> 172.16.242.131:3868                 0     2648        0   994176        0
  -> 172.16.242.132:3868                 0     1358        0   537100        0
  -> 172.16.242.133:3868                 0     2271        0   995296        0
  -> 172.16.242.134:3868                 0     1440        0   618544        0
```

show history

Displays the history of commands executed on the system.

Syntax

```
show history
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show history
03-04 16:56:03 -- show docker service | include diameter
03-04 16:56:22 -- show docker service | include diameter | include diameter-endpoint-0
03-04 16:57:31 -- docker connect docker-host-info-s8
03-04 16:59:19 -- docker connect socket-forwarder-s1
03-04 17:01:02 -- ifconfig
03-04 17:01:22 -- docker connect socket-forwarder-s1
03-04 17:01:54 -- docker connect diameter-endpoint-s2
03-04 17:03:32 -- docker connect diameter-endpoint-s2
03-04 17:05:25 -- docker connect diameter-endpoint-s1
```

show license details

Displays the current license details installed on the system.

Syntax

```
show license details
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show license details
ID       DEFAULT  COUNT      EXPIRATION
-----------------------------------------------------------
SP_CORE  true     100000000  2017-06-02T02:04:07+00:00
```

Table 100: Parameter Description

Parameter	Description
ID	ID of the license entry.
Default	Indicates if this is the default 90 day license installed on system install.
Count	Count for the given license.
Expiration	Expiration timestamp for the license.

show log application

Displays the application log in a viewer that enables you to scroll and search.

Syntax

```
show log application
```

Command Mode

OPERATIONAL

VNFs

DRA

show log engine

Displays the engine log in a viewer that enables you to scroll and search.

Syntax

```
show log engine
```

Command Mode

OPERATIONAL

VNFs

DRA

show logger level

Displays the current logger levels in the system that overrides the default logging.

Syntax

```
show logger level
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show logger level
Logger    Current Level
--------  ---------------
dra       warn
```

Table 101: Parameter Description

Parameter	Description
Logger	The logger that is overridden.
Current Level	The current level of logging.

show orchestrator-database-status

Displays the MongoDB members database status running on orchestrator, orchestrator-backup-a, and orchestrator-backup-b containers.

Syntax

```
show orchestrator-database-status
```

Command Parameters

None

Command Mode

Operational

VNFs

All

Examples

The following example also shows a sample output:

```
admin@orchestrator[an-dbmaster]# show orchestrator-database-status
ADDRESS                  PORT        STATUS
-------------------------------------------------
orchestrator             27017     PRIMARY
orchestrator-backup-a    27017     SECONDARY
orchestrator-backup-b    27017     SECONDARY
```

Note In case any member is down or not able to retrieve its status, it is shown as NO_CONNECTION. For all other members, respective mongo status is displayed.

For example, if orchestrator-backup-a mongo member is down.

```
admin@orchestrator[an-dbmaster]# show orchestrator-database-status
ADDRESS                  PORT        STATUS
-------------------------------------------------
orchestrator             27017     PRIMARY
orchestrator-backup-a    27017     NO_CONNECTION
orchestrator-backup-b    27017     SECONDARY
```

show patches

Lists the patches that are in `/data/orchestrator/patches` directory.

Syntax

```
show patches
```

Command Mode

OPERATIONAL

VNFs

All

Command Usage

The `show patches` indicates the patch that is loaded in the given patch directory and not a patch that is applied to the system .

show running-config binding db-connection-settings

Displays the binding DB write connection settings.

Syntax

```
show running-config binding db-connection-settings
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show running-config binding db-connection-settings | tab
                          MAX
BINDING  CONNECT  SOCKET   WAIT  CONNECTIONS
TYPE     TIMEOUT  TIMEOUT  TIME  PER HOST
-----------------------------------------------
drasession       500      1000     500   10
```

show running-config binding db-read-connection-settings

Displays the binding DB read connection settings.

Syntax

```
show running-config binding db-read-connection-settings
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show running-config binding db-connection-settings | tab
                                MAX
            CONNECT  SOCKET     WAIT                ACCEPTABLE
            TIMEOUT  TIMEOUT    TIME    CONNECTIONS LATENCY
BINDING     FOR      FOR        FOR     PER HOST    DIFFERENCE
TYPE        READ     READ       READ    FOR READ    FOR READ
-------------------------------------------------------------
ipv6        500      1000       500     10          5
imsiapn     500      1000       500     10          5
msisdnapn   500      1000       500     10          5
drasession  500      1000       500     10          5
```

show running-config binding shard-metadata-db-connection

Displays the binding shard metadata database connection.

Syntax

```
show running-config binding shard-metadata-db-connection
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show running-config binding shard-metadata-db-connection | tab
SHARD
METADATA
BINDING
TYPE        HOST           PORT
---------------------------------
ipv6        193.1.163.114  27019
ipv6        193.1.163.115  27019
ipv6        193.1.163.164  27019
ipv4        193.1.163.114  27019
ipv4        193.1.163.115  27019
ipv4        193.1.163.164  27019
imsiapn     193.1.163.116  27019
imsiapn     193.1.163.25   27019
imsiapn     193.1.163.63   27019
imsiapn     193.1.163.65   27019
imsiapn     93.1.163.165   27019
msisdnapn   193.1.163.116  27019
msisdnapn   193.1.163.25   27019
```

```
msisdnapn   193.1.163.63   27019
msisdnapn   193.1.163.65   27019
msisdnapn   93.1.163.165   27019
drasession  193.1.163.114  27019
drasession  193.1.163.115  27019
drasession  193.1.163.164  27019
```

show scheduling effective-scheduler

Displays the effective scheduler running in the system.

Valid results are HA and AIO.

Syntax

```
show scheduling effective-scheduler
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show scheduling effective-scheduler
scheduling effective-scheduler HA
```

show scheduling status

Displays the currently loaded modules.

Syntax

```
show scheduling status
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
scheduler# show scheduling status
                                                OUT
                                    RUN         OF
MODULE                  INSTANCE    LEVEL  STATE    DATE
-------------------------------------------------------
consul                  1           50     RUNNING  false
admin-db                1           75     RUNNING  false
memcached-vip           1           100    RUNNING  false
prometheus              1           100    RUNNING  false
prometheus              2           100    RUNNING  false
prometheus              3           100    RUNNING  false
```

Table 102: Parameter Description

Parameter	Description
Module	Module name that is running.
Instance	The instance number scheduled for scalable modules.
Run Level	The relative run level of the module compared to other modules. In an upgrade, the system reschedules from highest run level to lowest run level and in a downgrade the system schedules from low to high.
State	The current state of the module. Valid states are: • RUNNING • SCHEDULING • STOPPING
Out of Date	Indicates whether the software is out of date with the running system.

show scheduling vm-target

Displays the results of the scheduling vm-target calculation.

Syntax

```
show scheduling vm-target
```

Command Mode

OPERATIONAL

VNFs

All

Parameter Description

Parameter	Description
group	The vm-target group name that the count applies.
Count	The calculated count of VMs for scaling.

show system diagnostics

Shows the current diagnostics.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 103: Parameter Description

Command Parameter	Description
Node ID	ID of the node where the diagnostics was run.
Check	The ID of the check that was run.
IDX	For Checks that return multiple results the corresponding index number
Status	Indicates if the check is passing or not.
Message	The corresponding message for the diagnostic.

Examples

```
scheduler# show system diagnostics | tab
NODE        CHECK ID                  IDX  STATUS   MESSAGE

-------------------------------------------------------------------------------------------------------------------------

binding-s1  serfHealth                1    passing  Agent alive and reachable

binding-s1  service:cisco-policy-api  1    passing  TCP connect localhost:8080: Success
```

```
binding-s1  service:cisco-policy-app  1    passing  CLEARED: Session creation is allowed

binding-s1  service:cisco-policy-app  2    passing  CLEARED: -Dcom.broadhop.developer.mode
 is disabled
```

show system history

Shows the history of system events.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 104: Parameter Description

Command Parameter	Description
IDX	The index of the event in the system history log.
Event Time	Timestamp of the event in the system history log.
Module	The internal module that generated the history log entry.
Message	The message associated with the log entry.

Examples

```
scheduler# show system history
IDX  EVENT TIME                    MODULE         MESSAGE

---------------------------------------------------------------------------------------------------------
1    2017-02-04T02:04:02.469+00:00  system         System started

2    2017-02-04T02:04:29.021+00:00  docker-engine  Adding docker engine session-shard-2-80941f

3    2017-02-04T02:04:29.096+00:00  docker-engine  Adding docker engine diameter-endpoint-3

4    2017-02-04T02:04:29.187+00:00  docker-engine  Adding docker engine diameter-endpoint-2

5    2017-02-04T02:04:29.303+00:00  docker-engine  Adding docker engine binding-c74547

6    2017-02-04T02:04:29.375+00:00  docker-engine  Adding docker engine control-2

7    2017-02-04T02:04:29.503+00:00  docker-engine  Adding docker engine session-shard-1-e079cf
```

```
8    2017-02-04T02:04:29.583+00:00  docker-engine  Adding docker engine control-1

9    2017-02-04T02:04:29.671+00:00  docker-engine  Adding docker engine control-0

10   2017-02-04T02:04:29.751+00:00  docker-engine  Adding docker engine binding-dabba5

11   2017-02-04T02:04:29.843+00:00  docker-engine  Adding docker engine binding-73d3dc

12   2017-02-04T02:04:29.981+00:00  docker-engine  Adding docker engine binding-8a8d17
```

show system secrets open

Shows if the system secrets are unsealed.

This command returns true if the secrets are unsealed and false if they are still sealed. To open the system secrets, see system secrets unseal , on page 245.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Examples

```
scheduler# show system secrets open
system secrets open true
```

show system secrets paths

Shows the current set secrets.

This command does not show the value of the secrets only the path and if the value is readable by the system.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 105: Parameter Description

Command Parameter	Description
Path	The identifying path of the secret.
Status	Indicates if the path can be read by the system.

Examples

```
scheduler# show system secrets paths
PATH  STATUS
--------------
test  valid
```

show system software available-versions

Shows the list of available software versions to upgrade or downgrade a system.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Examples

```
scheduler# show system software available-versions
VERSION
-----------------------------------
12.9.9-dra.2017-03-03.115.0f485ef
```

show system software docker-repository

Shows the currently configured docker-repository.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Examples

```
scheduler# show system software docker-repository
system software docker-repository registry:5000
```

show system software version

Shows the currently installed software version.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Examples

```
scheduler# show system software version
system software version 12.9.9-dra.2017-03-03.115.0f485ef
```

show system software iso stage file

Displays the currently staged files in the /data/isos/staged-isos folder.

Syntax

```
show system software iso stage file
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Examples

The following example also shows a sample output:

```
scheduler# show system software iso stage file
NAME                  CREATED                    SIZE MB  MD5 SUM

-------------------------------------------------------------------------------------------
cisco-policy-dra.iso  2017-05-17T12:35:58+00:00  1100.04  c636794475b76e84041901b0ca3dcac4
```

Where:

- Name: The filename of the iso.
- Created: The date the file was created on the file system.
- Size MB: The size of the file in megabytes.
- MD5 Sum: The MD5 sum of the file.

show system software iso details

Displays the currently active ISOs that are loaded on the system.

Syntax

```
show system software iso details
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Examples

The following example also shows a sample output:

```
CATEGORY  NAME     VERSION  QUALIFIER    CREATED  ACTIVE  MB
-------------------------------------------------------------------------
product cisco-policy-dra 12.9.9   dra.2017-05-   2017-05  true   1102.9
                                    17.441.69    -17T13:
                                    68d89        4:15.708
                                                 +00:00
```

Where:

- Category: The type of ISO. Either product or extras. Extras can be used to load external docker images for use by external services.
- Name: The product name of the ISO

- Version: The version of the ISO
- Qualifier: The qualifier of the ISO
- Created Date: The creation date of the ISO on the file system
- Active: Indicates if the registry is currently pointing to the ISO to download images.
- Size: The size of the ISO on the file system.

show system status

Shows 100% if the minimum set of containers are running for the system to operate.

A system status of 100% does not guarantee the following:

- The system is fully configured through the CLI or Policy Builder.
- All redundant VMs are UP. For example, a worker VM, and a distributor VM.
- Distributor VMs are UP.

To verify a system is healthy and all desired VMs are active, execute the following commands:

- `show docker engine`
- `show system status`
- `show system diagnostics`
- `show docker service`
- `show alerts`

Syntax

```
show system status
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Examples

The following example also shows a sample output:

```
scheduler# show system status
```

show system status debug

Shows if the system is currently configured with debug tools.

Syntax

```
show system status debug
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Examples

The following example also shows a sample output:

```
scheduler# show system status debug
system status debug false
```

Where:

- Debug: Indicates if the system is configured to deploy containers with debug tools

show system status downgrade

Shows if the system is currently downgrading the installed software.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Examples

```
scheduler# show system status downgrade
system status downgrade false
```

show system status running

Shows if the system is currently running.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Examples

```
scheduler# show system status running
system status running true
```

show system status upgrade

Shows if the system is currently upgrading an installed software.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Examples

```
scheduler# show system status upgrade
system status upgrade false
```

statistics bulk file

Defines a new bulk statistics file that the system generates on a regular basis.

Syntax

```
statistics bulk file name header
 header query query format
format no bulk file name
```

Command Parameters

Table 106: Parameter Description

Command Parameter	Description
name	The base name of the bulk statistics file to create. The final file name generated has the following format:`<name>-<timestamp in seconds>.csv`
header	The exact text of the header to put at the start of all new files.
query	The Prometheus query to execute to build the bulk statistics. The query format is described in the Prometheus documentation:https://prometheus.io/docs/querying/basics/
format	The format of the output line. Each time series returned from the query that is executed will pass through the formatting string. Substitution variables appear as ${variable}. The following pre-defined variables exist in addition to the ones returned from Prometheus: • current-value – last value returned • max-value – max value over last 5 minutes • avg-value – average value over last 5 minutes • min-value – minimum value over last 5 minutes • timestamp – timestamp of when the sample was taken in the following format: yyyy-MM-dd'T'HH:mm:ss'Z'

Command Mode

CONFIG

VNFs

All

Command Usage

Use the bulk file command to define a bulk statistics file that supplements the default bulk statistics files created by the system. The format and queries are user defined.

Examples

The following example creates a bulk file on peer message rates:

```
statistics bulk file peer_tps
 query  "peer_message_total{remote_peer!=\"\"}"
 format ${app_id},${direction},${instance},${local_peer},
${remote_peer},${type},${current-value}
!
```

statistics bulk interval

Modifies the timer that the system uses to generate the bulk statistics that are defined via the bulk file command.

Syntax

```
statistics bulk interval interval no bulk interval
```

Command Parameters

Table 107: Parameter Description

Command Parameter	Description
interval	Timer length (in seconds) used to trigger a new bulk statistics file.

Command Mode

CONFIG

VNFs

All

Command Usage

Use the bulk interval command to control the timer length in triggering a new bulk statistics file.

Notes:

1. The generation of bulk statistics runs +/- 10 seconds of the interval.
2. The generation of bulk statistics is not synchronized to the minute.
3. The default interval, if not defined, is 300 seconds.

Examples

The following example creates a bulk file every 10 minutes:

```
scheduler(config)# bulk interval 600
```

statistics detail

Adds a statistics detail for the system to capture.

Syntax

```
statistics detail query category name query query format format scale scale
```

Command Parameters

Table 108: Parameter Description

Command Parameter	Description
category	Category of the statistic.
name	Name of the statistic.
query	Prometheus query to execute in order to retrieve the statistics.
format (optional)	Formatting rule for the statistic. The labels from the Prometheus query are substituted using the ${label} format.
scale (optional)	Scaling factor to take the raw value and scale to by the scale factor. A negative value divides by the scale factor and a positive value multiples by the scale factor.

Command Mode

CONFIG

VNFs

All

Command Usage

The statistics detail command triggers the application to monitor a given statistic and record it in memory and for reporting using the show statistics detail command. The values are refreshed every 10 seconds.

Examples

```
statistics detail query diameter success-message-tps
 query  "sum(rate(diameter_endpoint_request_total{result_code=\"2001\"}[10s])) by
(app_id,message_type)"
```

```
 format "${app_id} ${message_type}"
!
```

statistics icmp-ping

Creates a probe that tests whether a host is up using ICMP ping.

Syntax

`statistics icmp-ping address no statistics icmp-ping address`

Command Parameters

Table 109: Parameter Description

Command Parameter	Description
address	The address to ping via ICMP. The resultant statistics are stored in the following metric: • probe_success • probe_duration_seconds • probe_ip_protocol

Command Mode

CONFIG

VNFs

All

Command Usage

Use the statistic icmp-ping command to instruct the monitoring system to ping the given address using the ICMP protocol. The IP address must be reachable via the master, control-a, and control-b hosts.

Examples

The following example creates an ICMP ping test:

```
scheduler(config)# statistics icmp-ping 10.10.10.10
```

statistics summary

Adds a statistics summary for the system to capture.

Syntax

`statistics summary query category name query query scale scale`

Command Parameters

Table 110: Parameter Description

Command Parameter	Description
category	Category of the statistic.
name	Name of the statistic.
query	Prometheus query to execute in order to retrieve the statistics.
scale (optional)	Scaling factor to take the raw value and scale to by the scale factor. A negative value divides by the scale factor and a positive value multiples by the scale factor.

Command Mode

CONFIG

VNFs

All

Command Usage

The statistics summary command triggers the application to monitor a given statistic and record it in memory and for reporting using the show statistics summary command. The values are refreshed every 10 seconds.

The summary command does not support "group by" operations to show multiple lines from a single query.

Examples

```
statistics summary query diameter tps
 query "sum(rate(diameter_endpoint_request_total{result_code=\"2001\"}[10s]))"
!
```

Storage Health Check Service Commands

show storage-health-check service

Displays the health check settings. The following are default values:

Interval = 3 seconds

Failover Hold Time = 30 seconds

Syntax

```
admin@orchestrator[vPAS-A-master]# show storage-health-check service
Parameter           Value
------------------  -------
```

```
enable              true
failover-hold-time  10
interval            2
```

storage-health-check service <enable | disable | restart>

enable – Enables storage health check on diameter nodes

disable – Disables storage health check on diameter nodes

restart – Restarts storage health check on diameter nodes. Restart needs to be performed if health check settings are modified after enabling the service.

Configuring Storage Health Check Settings

The following commands can be used to configure storage health check settings.

```
storage-health-check set interval <value in seconds>
```

```
storage-health-check set failover-hold-time <value in seconds>
```

```
storage-health-check clear interval
```

Reset to default.

```
storage-health-check clear failover-hold-time
```

Reset to default.

Applying Configuration Changes

If the interval or failover-hold-time is updated after enabling health check service, then the changes does not automatically take effect. The service needs to be restarted for the changes to take effect by using the following command:

```
storage-health-check service restart
```

If the configuration is updated prior to enabling the service, enabling the service applies the latest settings.

```
storage-health-check service enable
```

system abort-downgrade

Stops a downgrade that is in progress.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

The system abort-downgrade command stops the current rolling downgrade of the system. This command is only available when the system is in the process of downgrading and is not available after the downgrade is complete. Once this command is issued, system upgrade , on page 250 command should be issued to revert this software to the previous version.

system abort-upgrade

Stops an upgrade that is in progress.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Usage Guidelines

The system abort-upgrade command stops the current rolling upgrade of the system. This command is only available when the system is in the process of upgrading is not available after the upgrade is complete. Once the command is issued, system downgrade, on page 238 command should be issued to revert this software to the previous version.

system downgrade

Downgrades the system to a older software version.

Syntax

```
system downgrade version version [consul-downgrade [true/false] [snapshot_name]]
```

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 111: Parameter Description

Command Parameter	Description
version	The new software version to install into the system.

Command Parameter	Description
consul-downgrade	consul-downgrade is an optional parameter. For more information , see consul-downgrade, on page 239.

consul-downgrade

During upgrade, it takes a snapshot of existing consul data which are yet to be upgraded and saves as <*version-name*> (to which you are upgrading) and upgrade proceeds normally. Post upgrade all consul servers/agents will be upgraded to newer version.

For example, if you are upgrading from 20.2.0.release to 21.x.0.release, snapshot name is *21.x.0.release*.

If the value is set as true, following operations are carried out:

- Check if you have provided snapshot-name. If you have not provided the snapshot name, by default, it takes current version as snapshot. You can also provide the snapshot name. To list all the available snapshots, use `consul list-snapshots` command.
- If snapshot is present, then consul is restored with the snapshot and further downgrade proceeds normally.
- If snapshot is not present, then downgrade does not get started and an error is displayed.
- If you have provided the snapshot-name, then snapshot (if exists) in `/data/orchestrator/config/snapshot/` is verified and consul is restored with the given snapshot and downgrade continues.
- In case of no snapshot, an error is displayed.

Note Post rollback, consul data is of state before upgrade if consul downgrade is selected during ISO rollback. Hence, if changes are made to the consul config post upgrade, they are lost and you need to reapply the config changes.

Caution You cannot restore newer version snapshot on an old consul server.

Example:

1. When upgrading to any new version (for example, from v1 to v2), it takes consul snapshot as `/data/orchestrator/config/snapshot-consul/v2`.
2. During downgrade (for example, from v2 to v1), provide snapshot name in system-downgrade command.
3. When upgrading to v3 from v2 (for example, consul version for v1 is 1.0.0, consul version for v2 is 1.5.3 and consul version for v3 is 1.5.3). Upgrade from v1 to v2, snapshot is store as v2; from v2 to v3, snapshot is stored as v3.
4. If you want to downgrade directly from v3 to v1 and you do not provide the snapshot name, by default, it takes the snapshot of v3 and consul version is 1.5.3. The downgrade fails. You must provide the snapshot name in system-downgrade command as v2.

Command Usage

The system downgrade command installs new software on the system using a rolling downgrade approach to minimize service interruption. Care must be taken to ensure that the system downgrade command is used when moving from a higher software version to a lower version of the software. The rolling downgrade upgrades the software modules in startup order. After the command is issued, the CLI disconnects while the CLI software is restarted. The CLI generally becomes available within 30 seconds. Once the CLI becomes available, the status of the upgrade can be monitored using the show scheduling status, on page 221 command.

Examples

```
system downgrade version 12.9.9-dra.2017-03-03.115.0f485ef
```

system disable-debug

Disables debug tools in deployed containers.

Syntax

```
system disable-debug
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the system disable-debug command to turn off debugging tools on newly launched containers.

Examples

The following example disables debug tools:

```
scheduler# system disable-debug
```

system disable-external-services

Disables external services that are currently running in the system.

Syntax

```
system disable-external-services
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the system disable-external-services to stop all services registered with the scheduling external-service command.

Examples

The following example disables external services:

```
scheduler# system disable-external-services
```

system enable-debug

Enables debug tools in deployed containers.

Syntax

```
system enable-debug
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the system enable-debug command to turn on debugging tools on newly launched containers.

Examples

The following example enables debug tools:

```
scheduler# system enable-debug
```

system enable-external-services

Enable external registered services.

Syntax

```
system enable-external-services
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Command Usage

Use the system enable-external-services command to enable external services that are currently registered with the scheduling external-service command.

Examples

The following example enables external services:

```
scheduler# system enable-external-services
```

show fluent-bit configurations

Displays the status of fluent-bit configurations.

Syntax

```
show fluent-bit configurations
```

Command Mode

OPERATIONAL

VNFs

All

Examples

The following is an example:

```
show fluent-bit configurations
Fluent-Bit configurations    Current Value
```

```
--------------------------  --------------------------------
OAM-ip                       x.x.x.x
OAM-port                    9200
backlog-mem-limit           2048M
elasticsearch-ip            x.x.x.x
elasticsearch-password      3300901EA069E81CE29D4F77DE3C85FA
elasticsearch-port          9400
elasticsearch-user          elastic
flush-interval              900
max-chunks-up               3500
```

system secrets add-secret

Adds a secret to the system.

Syntax

`system add-secret path` *`path`* `secret` *`secret`*

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 112: Parameter Description

Command Parameter	Description
Path	The identifying path of the secret to add.
Secret	The clear text value of the secret to add.

Command Usage

The system add-secret command adds a secret to the system. This command is available only if the secrets are open. See show system secrets open , on page 225.

system secrets remove-secret

Removes a secret from the system.

Syntax

`system remove-secret path` *`path`*

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 113: Parameter Description

Command Parameter	Description
Path	The identifying path of the secret to remove.

Command Usage

The system remove-secret command removes a secret from the system. This command is available only if the secrets are open. See show system secrets open , on page 225.

system secrets set-passcode

Overwrites the current passcode that is used to encrypt or decrypt the master key for the secrets.

Syntax

```
system secrets set-passcode passcode
```

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 114: Parameter Description

Command Parameter	Description
Passcode	The new passcode to seal the secrets.

Command Usage

The system secrets command is used to change the passcode to unlock the secrets stored within the operational database. All secrets are encrypted using a randomly generated master-key that is encrypted/decrypted by the end-user provided passcode. If the passcode is lost, then the secrets currently stored are not recoverable. This command is available only if the secrets are open. See show system secrets open , on page 225.

system secrets unseal

Unseals the secrets if a non-default passcode is used to seal the secrets.

Syntax

```
system secrets unseal passcode passcode
```

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 115: Parameter Description

Command Parameter	Description
Passcode	The passcode to unseal the secrets.

Command Usage

The system secrets unseal command is used to unlock any stored secrets so that they can be shared with services that require a clear text secret or password. An example of this is a database connection password.

system software iso stage clean

Remove all downloaded ISOs from the stage directory.

Syntax

```
system software iso stage clean
```

Command Parameters

None

Command Mode

OPERATIONAL

VNFs

All

Command Usage

The system software iso stage clean command removes all files that have been staged in the hosts /data/isos/staged-isos/ directory. This command should be run after an ISO file has been uploaded via the system software iso load command.

Examples

```
scheduler# system software iso stage clean
```

system software iso stage pull

Downloads a software ISO to the stage directory on the host.

Syntax

```
system software iso stage pull URL
```

Command Parameters

Table 116: Parameter Description

Command Parameter	Description
URL	The URL to download into the hosts /data/isos/staged-isos/ directory. If the URL ends with the zsync suffix, then the zsync command is invoked to retrieve the file.

Command Mode

OPERATIONAL - Not available via NETCONF/RESTCONF

VNFs

All

Command Usage

Invocation of the command downloads the given URL to the /data/isos/staged-isos/ directory. After invocation of this command, invocation of the show system software iso stage file command shows details of the downloaded file and the system software iso load command loads the file into the system.

Examples

The following example also shows a sample output:

```
scheduler# system software iso stage pull
http://171.70.34.121/microservices/latest/cisco-policy-dra.iso
--2017-05-17 15:08:39--  http://171.70.34.121/microservices
/latest/cisco-policy-dra.iso
Connecting to 171.70.34.121:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1153468416 (1.1G) [application/octet-stream]
```

```
Saving to: 'cisco-policy-dra.iso'

cisco-policy-dra.iso                                4%[=====>

                                ]  45.85M  4.07MB/s    eta 4m 27s
```

system software iso activate

Activate an existing ISO.

Syntax

```
system software iso activate category [product|extras] name name version version qualifier
 qualifier
```

Command Parameters

Table 117: Parameter Description

Command Parameter	Description
Category	The category to load the ISO. Either product or extras can be selected. The extras category represents a docker registry that contains external (non-product) docker images.
Name	The product name of the ISO to activate.
Version	The version of the ISO to activate
Qualifier	The qualifier of the ISO to activate

Command Mode

OPERATIONAL

VNFs

All

Command Usage

The system software iso activate command triggers the system to restart the local docker registry to point to the given ISO. This command should be run before upgrading or downgrading the software.

Examples

The following example loads and activates a product ISO:

```
scheduler# system software iso activate category product name cisco-policy-dra version
12.9.9 qualifier dra.2017-05-17.441.6968d89
```

system software iso delete

Deletes an existing ISO.

Syntax

```
system software iso delete category [product|extras] name name version version qualifier
qualifier
```

Command Parameters

Table 118: Parameter Description

Command Parameter	Description
Category	The category to load the ISO. Either product or extras can be selected. The extras category represents a docker registry that contains external (non-product) docker images.
Name	The product name of the ISO to delete.
Version	The version of the ISO to delete
Qualifier	The qualifier of the ISO to delete

Command Mode

OPERATIONAL

VNFs

All

Command Usage

The system software iso delete command triggers the system to remove the ISO. This command can only be run on non-active ISOs.

Examples

The following example deletes an ISO:

```
scheduler# system software iso delete category product name cisco-policy-dra version 12.9.9
 qualifier dra.2017-05-17.441.6968d89
```

system software iso load

Load a new ISO into the system.

Syntax

```
system software iso load category [product|extras] file filename activate [true|false]
```

Command Parameters

Table 119: Parameter Description

Command Parameter	Description
Category	The category to load the ISO. Either product or extras can be selected. The extras category represents a docker registry that contains external (non-product) docker images.
Filename	The filename of the ISO to load.
Activate	Indicates whether the system should switch the internal docker registry to point to the new ISO.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

The system software iso load command triggers unpacking of the staged ISO into a permanent location on the host. This command is executed before a system upgrade command can be executed.

Examples

The following example loads and activates an ISO:

```
scheduler# system software iso load category product file cisco-policy-dra.iso activate
true
```

system start

Starts all the services on a system that has been currently stopped.

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Usage Guidelines The system start command performs a controlled startup of the system by starting all the services in a rolling fashion taking into account various service dependencies.

system stop

Stops all the services on the system (excluding the CLI, NETCONF, and RESTCONF service).

Syntax

There are no arguments for this command.

Command Mode

OPERATIONAL

VNFs

All

Command Usage

The system stop commands performs a controlled shutdown of the system by stopping all the services in the reverse order of start-up.

Note For ephemeral databases (such as session), all data is lost on a system stop command.

system upgrade

Upgrades the system to a new software version.

The option `database-upgrade-parallel` enables parallel upgrade of database nodes. This option is provided to reduce the upgrade time without impacting the availability of DB cluster.

Syntax

```
system upgrade version <version>
system upgrade version <version> database-upgrade-parallel <true/false>
```

Command Mode

OPERATIONAL

VNFs

All

Command Parameters

Table 120: Parameter Description

Command Parameter	Description
version	The new software version to install into the system.

Command Usage

The system upgrade command installs new software on the system using a rolling upgrade approach to minimize service interruption. Care must be taken to ensure that upgrade command is used when moving from a lower software version to a higher version of the software. The rolling upgrade upgrades the software modules in reverse start-up order. After the command is issued, the CLI disconnects while the CLI software is restarted. The CLI generally become available within 30 seconds. Once the CLI becomes available, the status of the upgrade can be monitored using the `show scheduling status` command.

Examples

To trigger an upgrade as usual. Mongo nodes goes sequential upgrade.

```
system upgrade version 12.9.9-dra.2017-03-03.115.0f485ef
```

To trigger a parallel upgrade for mongo-nodes.

```
system upgrade version 19.5.0 database-upgrade-parallel true
```

vip-failover

Used to move the VIP between directors/distributors.

Syntax

```
vip-failover <vip-name> <current-vip-host-vm> <vip-ip> <vip-tracking-service> [ timeout ]
```

Command Parameters

Table 121: Parameter Description

Command Parameter	Description
vip-name	The VIP name.
current-vip-host-vm	The hostname where the VIP is present.
vip-ip	The floating IP of the VIP.
vip-tracking-service	The tracking service of the VIP in the format "Service-ip:Service-port".
timeout	The optional timeout value in seconds.

Command Mode

OPERATIONAL

VNFs

DRA

Command Usage

Use this command to move the VIP between director or distributor.

Examples

The following example moves the VIP between director or distributor.

```
network vip-failover testvip an-dra-director-0 10.225.115.253 10.225.115.253-3868
VIP failover completed successfully
```