RIFT.ware™ version 8.1.0.0.113704

Release Notes

April 2020

Notice

Table of Contents

RIFT.ware 8.1.0 Release Notes

This guide describes the RIFT.ware 8.1.0 release, including new features, fixed and known issues, with their workarounds.

New and Changed Features for 8.1.0

RIFT.ware version 8.1.0 introduces enhancements to improve management.

Feature	PFR and JIRAs
Import and Export support for ETSI GS NFV-SOL 001 and SOL 007 NSD and NS Packages	PFR-610
Launchpad UI Enhancements • Quota widget on Accounts Page and Instantiation Card • Redesign NS Viewport/History/Progress • RIFT Usability Improvement	PFR-622 RIFT-25956 RIFT-24790
Life Cycle Management Scaling Enhancements	RIFT-26110
Platform Improvements for Scalability	RIFT-26139 RIFT-28342
Retry and Rollback support for SOL 005	RIFT-28291
Support for Kubernetes as a VIM (Beta)	PFR-553

Configuration Manager Enhancements for CNFD

In this release, the rw-config-util python3 module is updated to handle CNF related data for configuration scripts. See Configuration Manager Enhancements for the entire rw-config-util python3 module. This module provides a standard interface for all configuration scripts.

Added the module rwconfigutil.cnf and updated the module rwconfigutil.main in this release.

module rwconfigutil.cnf

This API handles the CNFR section of the input file.

Classes

Class Cnf

class Cp(log, input_dict, cnfr_id=None)

Class to manage CNF related data.

Ancestors (in MRO)

- rwconfigutil.base.BaseSection

Instance variables

Variable *datacenter*

str: Name of the datacenter where this CNF is deployed.

Variable member_id

str: Member id of this CNF in the NSD.

Variable mgmt_interface

list: mgmt_interface list for all services belonging to this CNF.

Variable name

str: Name of this CNF.

Methods

Method get_mgmt_ip_list

```
def get_mgmt_ip_list(self, svc_name, exact_match=False)
```

list: management IPs for a given service.

Args

svc_name : str Service name to match.

exact_match (bool, optional) - Whether to do an exact match or check if name is in the service name. Default is false.

Returns

list of management IPs for the given service

Methods

Method get_mgmt_port

```
def get_mgmt_port(self, svc_name, exact_match=False)
```

int: management port for a given service.

Args

svc_name : str Service name to match.
exact_match (bool, optional) - Whether to do an exact match or check if name is in the service name. Default is false.

Returns

int: management port for the given service

module rwconfigutil.main

This is the main module for the RIFT configuration utility. The class is ConfigUtil. This module can be run from the command line by providing an input file.

Functions

Function main

```
def main(args=None)
```

Helper function to run this utility from the command line.

Classes

Class ConfigUtil

class ConfigUtil(input_dict=None, input_file=None, log=None, debug=False)

Configuration utility to parse the input for configuration script during NS LCM event.

Constructor for the ConfigUtil

Args

input_dict : dict, optional
Input for the config as a script.

input_file : dict, optional
Input file for the config.

log : logging.logger, optional
Pass the logger to use. If None, default logger used.

debug : str,optional
Enable debug logs in the stderr. Default is off.

Either input_dict or input_file should be provided.

Instance variables

Variable affected_cnfs

list of Cnf: Returns the list of affected CNFs.

The affected CNFs are the CNFS which got added or removed as part of the LCM event; for scale, it provides the list of CNFs being added or deleted.

Variable affected_vnfs

list of Vnf: Returns the list of affected VNFs.
The affected VNFs are the VNFS which got added or removed as part of the LCM event In case of VNF initial config, this provides the VNF that is being configured, for scale, it provides the list of VNFs being added or deleted.

Variable cnfs

list of Cnf: Returns the list of CNFs.

Variable config_agent

ConfigAgent: Returns the config agent section of the input.

Variable lcm

Lcm: Returns LCM section of the input.

Variable log

logging.Logger - Get the logger being used.

Variable log_file

str - Get the log filename, if available.

Variable ns_name

str: Returns the name for the NS.

Variable nsr_id

str: Returns the id for the NSR.

Variable other_cnfs

list of Cnf: Returns the list of the other CNFs which are not affected by the LCM.

Variable other_vnfs

list of Vnf: Returns the list of the other VNFs which are not affected by the LCM.

Variable parameters

list of Parameter: Returns the parameters section of the input.

Variable script_logger

logging.Logger - Get the logger to be used by invoking script.

Variable util_logger

generic logger for rw-config-util.

Variable vnfs

list of Vnf: Returns list of all VNFs.

Methods

Method get_asset_path

```
def get_asset_path(self, filepath)
```

Get the path to the asset.

Args

filepath : str
File path relative to the script.

Returns

str
Absolute path to the asset.

Raises

ValueError
If the absolute path is not under the script directory.

Method get_cnf_by_member_id

```
def get_cnf_by_member_id(self, member_id,
affected_only=False)
```

Get the first CNF matching the member id.

Args

member id : str
member id to match.

affected_only : bool, optional

Search in affected CNFs list or all CNFs. Default false, search in all CNFs. Name to match.

Returns

Cnf: CNF matching the member id.

Method get_vnf_by_name

```
def get_cnf_by_name(self, name, exact_match=True,
affected_only=False)
```

Get the first of CNF matching the name.

Args
name : str Name to match.

exact_match : bool, optional
Whether to do an exact match or check if name is in the CNF name. Default is
true, do exact match.

affected_only : bool, optional
Search in affected CNFs list or all CNFs. Default false, search in all CNFs.

Returns

Cnf: CNF matching the name.

Method get_cnfs_by_member_id

```
def get_cnfs_by_member_id(self, member_id,
affected_only=False)
```

Get the first CNFs matching the member id.

Args

member id : str
member id to match.

affected_only : bool, optional
Search in affected CNFs list or all CNFs. Default false, search in all CNFs.

Returns

list of rwconfigutil.cnf.Cnf: CNFs matching the member id.

Method get_cnfs_by_name

```
def get_cnfs_by_name(self, name, affected_only=False)
```

Get the list of CNFs containing the name.

Args

name : str Name to match.

affected_only : bool, optional
Search in affected CNFs list or all CNFs. Default is False, i.e., search in all CNFs.

Returns

List of Cnf: CNF matching the name.

Method get_vnf_by_member_id

```
def get_vnf_by_member_id(self, member_id,
affected_only=False)
```

Get the first VNF matching the member id.

Args

member id : int
member id to match.

affected_only : bool, optional
Search in affected VNFs list or all VNFs. Default false, search in all VNFs.

Returns

Vnf: VNF matching the member id.id

Method get_vnf_by_name

```
def get_vnf_by_name(self, name, exact_match=True,
affected_only=False)
```

Get the first VNF matching the name.

Args

name : str
Name to match.

exact_match : bool, optional
Whether to do an exact match or check if name is in the VNF name. Default is true, do exact match.

affected_only : bool, optional
Search in affected VNFs list or all VNFs. Default false, search in all VNFs.

Returns

Vnf: VNF matching the name.

Method get_vnfs_by_name
`def get_vnfs_by_name(self, name, affected_only=False)`
Get the list of VNFs containing the name.

Args
name : str
Name to match

affected_only : bool, optional
Search in affected VNFs list or all VNFs. Default is False, i.e., search in all VNFs.

Returns

list of Vnf: VNF matching the name

Import and Export support for ETSI GS NFV-SOL 001 and SOL 007 NSD and NS Packages

RIFT.ware now supports exporting and importing NSD ETSI SOL 001 and SOL 007 packages.

SOL 001 specifies the data model of NSD/VNFD using the TOSCA Simple Profile in YAML. SOL 007 specifies the structure of the Network Service Descriptor (NSD) file archive and the naming conventions for the different files it contains. The supported version is 2.6.1.

If you attempt to export a package from an OSM model, then features like scale, monitoring parameters, vnffgd and dependencies will not be transferred.

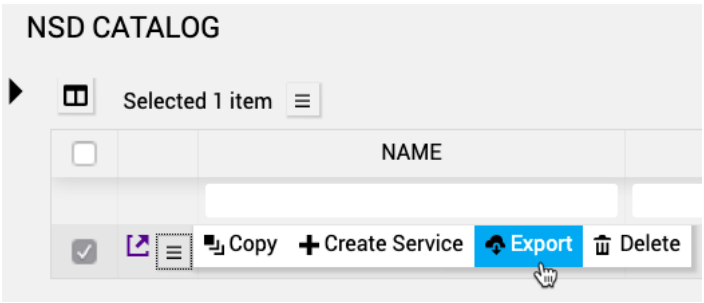
Import an NSD ETSI SOL 001 and SOL 007 package

On the **Launchpad** menu, go to the **CATALOG** page. Click the onboard a catalog package icon to onboard a descriptor package.

See the Onboarding Tutorial section.

Export an NSD ETSI SOL 001 and SOL 007 package

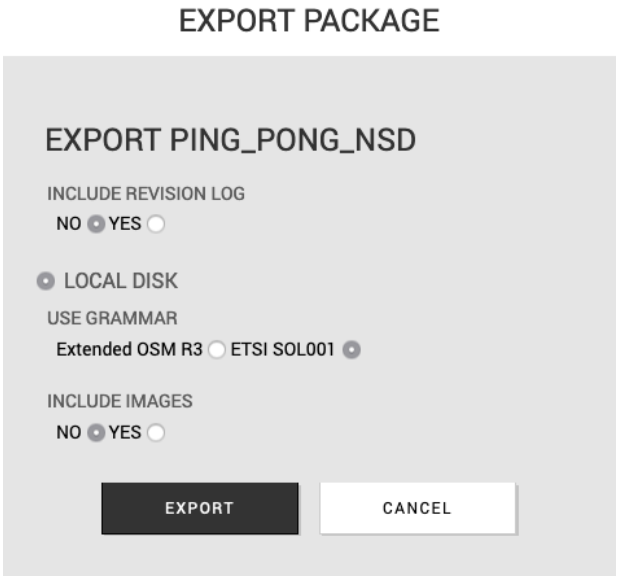
1. On the **Launchpad** menu, click **CATALOG** and select the appropriate network service descriptor from the catalog and click the ☰ icon.
2. Click Export.



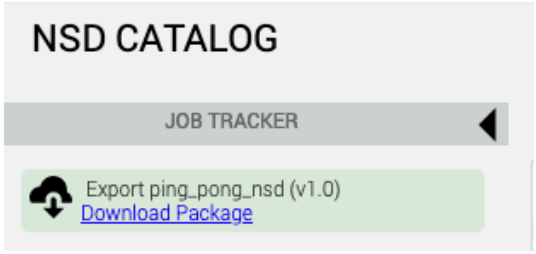
The **EXPORT SERVICE** screen appears.

3. Customize how you want to export the package. Choose **YES** in the **INCLUDE REIVSION LOG** option to export the package.
   - **INCLUDE REVISION LOG**: NO or YES
   - **LOCAL DISK USE GRAMMAR**: Extended OSM R3 or ETSI SOL001. Choose **ETSI SOL001** to export an NSD ETSI SOL 001 or SOL 007 package.
   - **INCLUDE IMAGES**: NO or YES

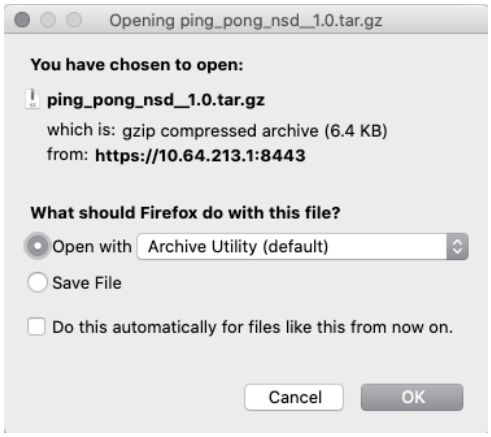
4. Click **EXPORT** to export the descriptor.



After clicking export, the Job Tracker displays the copied package.

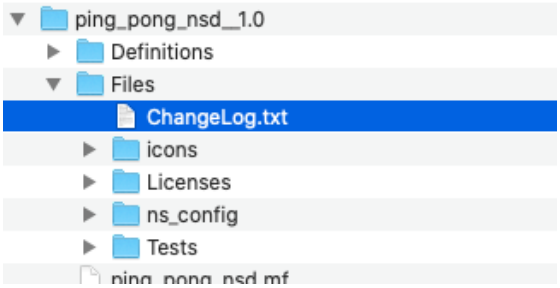


5. Click **Download Package** and choose the way that you'd like to open the exported package.



6. Click **OK** to open the zip file and the Downloads folder opens.

The exported catalog item is saved to your default download location with the following filename convention: ChangeLog.txt.



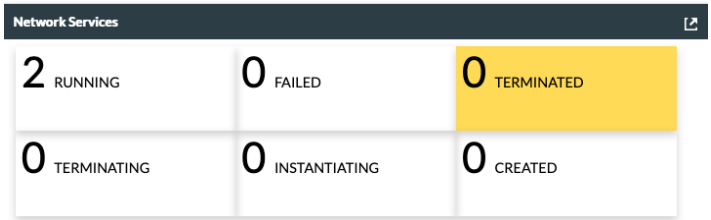
7. Open the ChangeLog.txt file. The file displays any modifications that you made to the file.

Launchpad Enhancements

In this release, the Launchpad UI has been restructured to be more efficient and useful to each individual operator. There are a variety of updates to many different sections of the UI including the following menus:

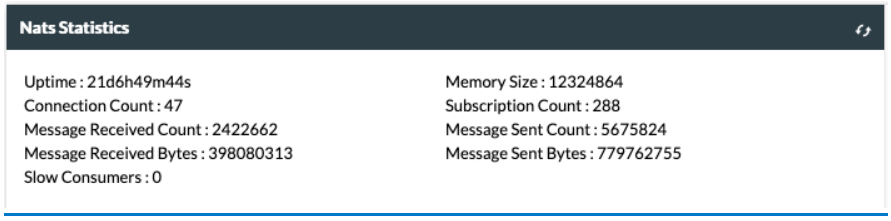
HOME Tab

This release includes enhancements to the Network Services widget on the **HOME** Dashboard. The Terminated phase is now captured in this section.



The Nats (a component of Launchpad) statistics now appear on the **HOME** dashboard. The statistics include:

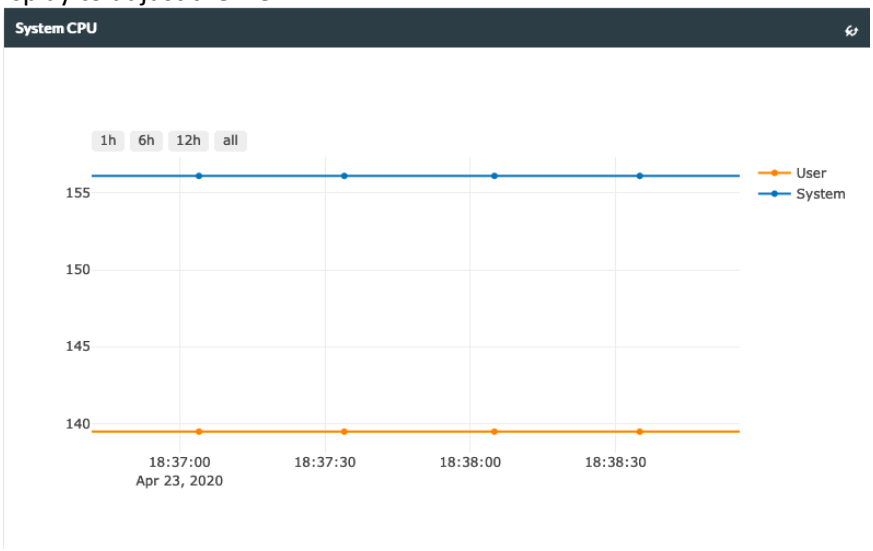
- Uptime
- Connection Count
- Message Received Count
- Message Received Bytes
- Slow Consumers
- Memory Size
- Subscription Count
- Message Sent Count
- Message Sent Bytes

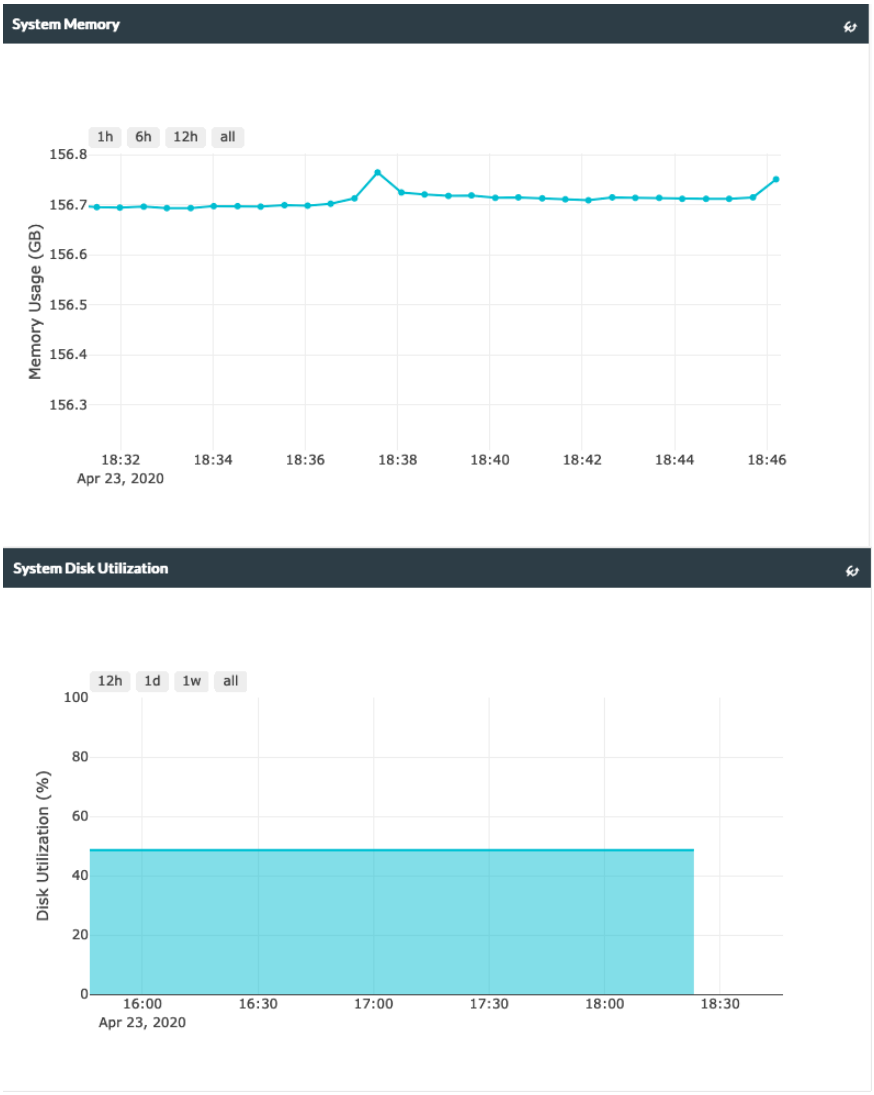


The Nginx statistics now appear on the **HOME** dashboard. The statistics include the:

- Active: Active connections.
- Read: Nginx reads request headers
- Write: Nginx reads request bodies, processes requests, or writes responses to a client.
- Waiting: Keep-alive connections. This number depends on the keepalive-timeout.
- Total Handled: Total number of handled connections.
- Accepted: All accepted connections.
- Handled: All handled connections.

System CPU, System Memory and System Disk Utility statistics now also appear on the **HOME** dashboard. This information is displayed over a period of time by moving your cursor over the display to adjust the view.

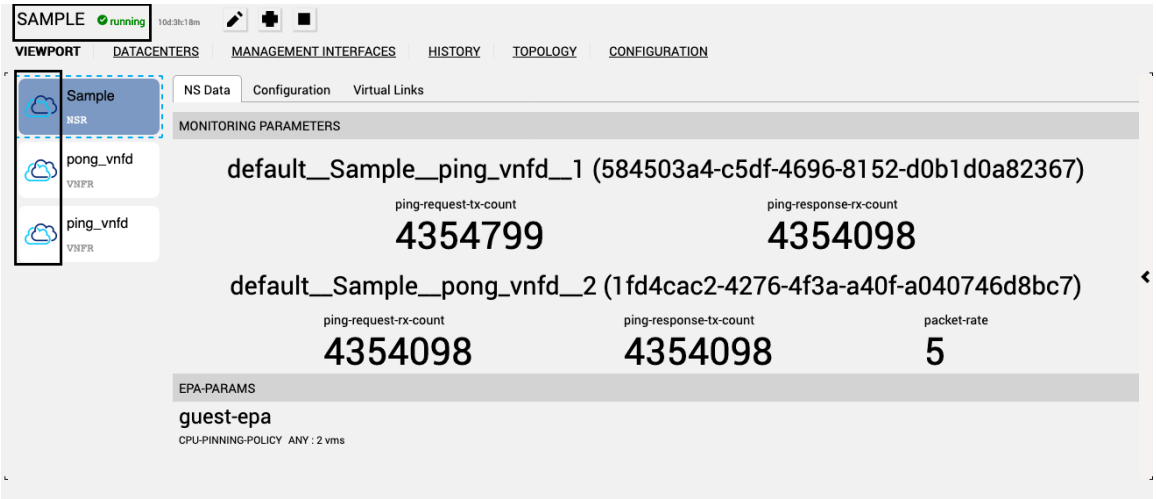




SERVICES Tab

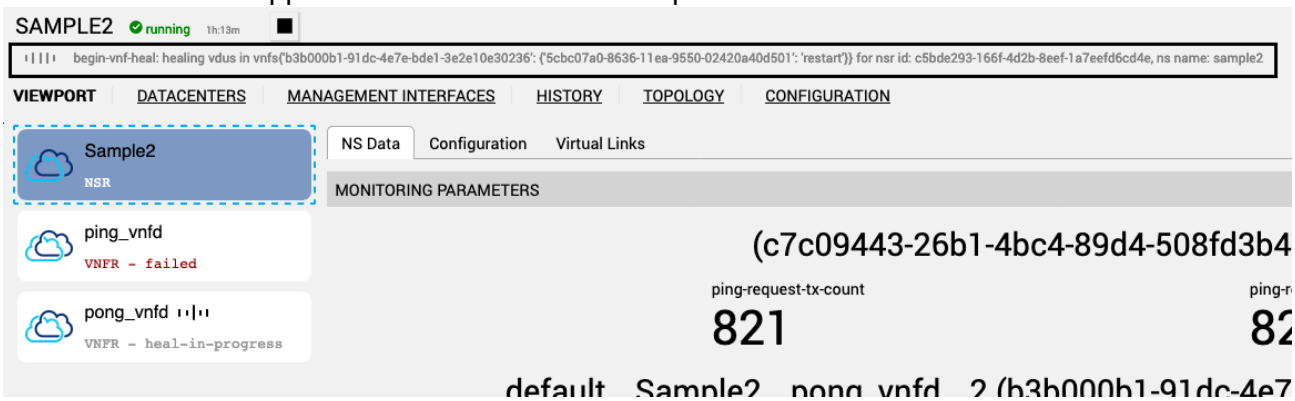
VIEWPORT Tab Changes

The **SERVICES → VIEWPORT** screen now shows the progress of LCM operations and the status appears next to the service name. The NSD or VNFD logo now appears on the left-hand side of the screen.

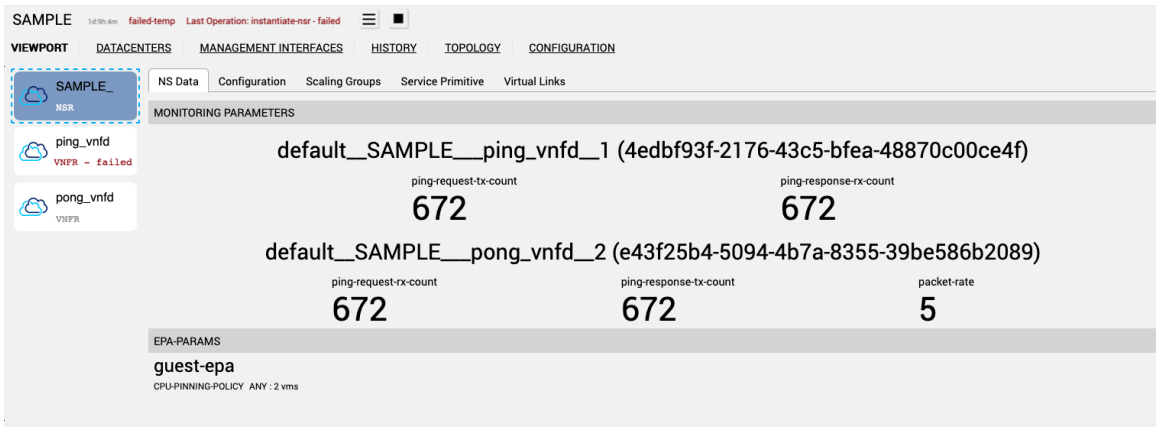


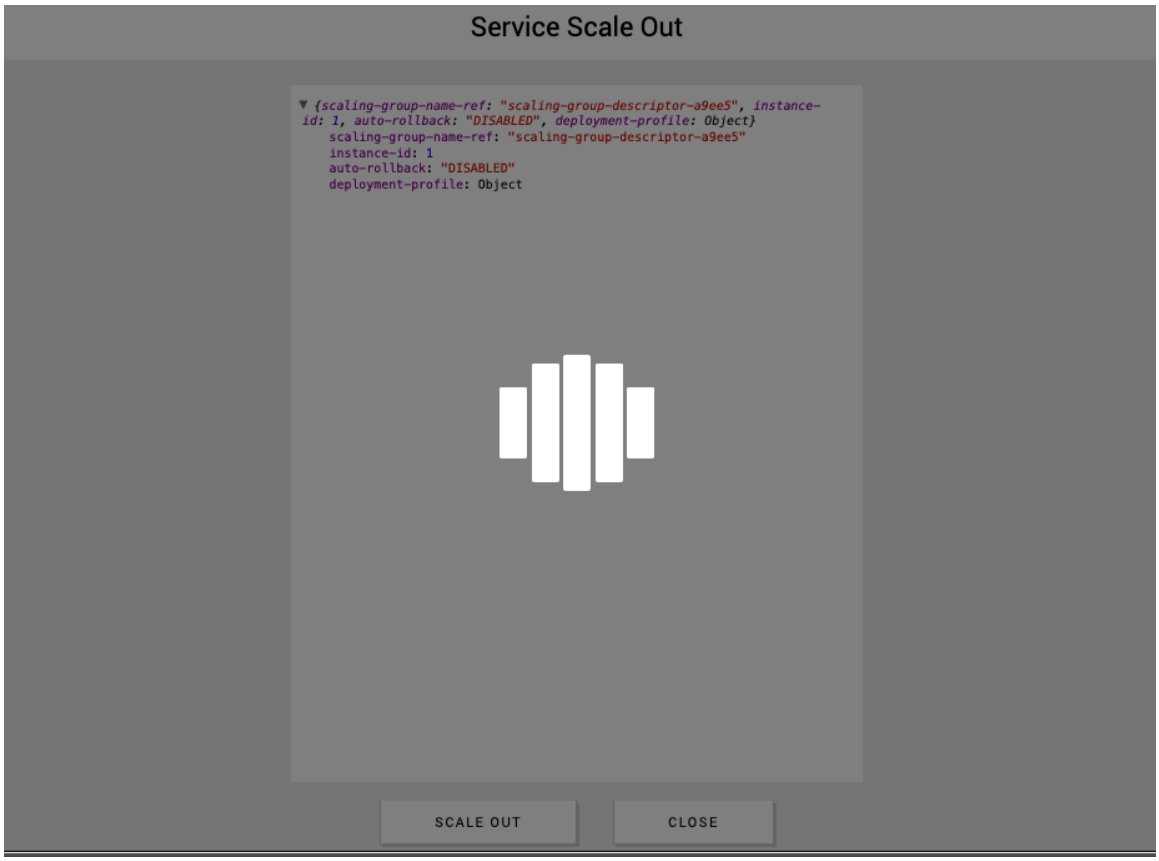
The **SERVICES** viewer screen also shows the most recent operational-event. The event and description appear in the header under the service name.

The heal state also appears on the Services → Viewport screen.

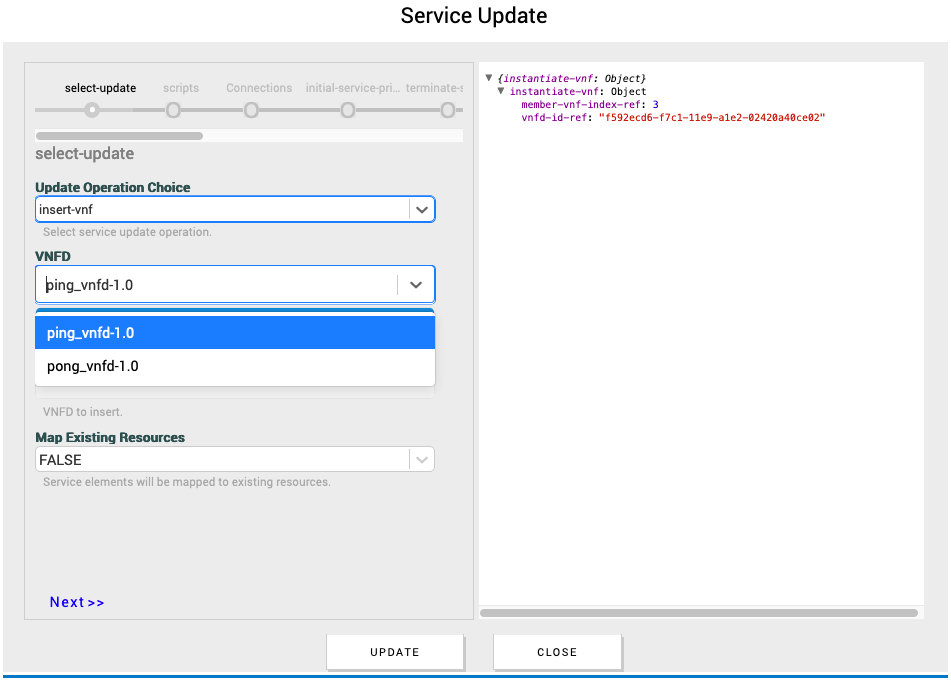


If a VNF heal attempt fails, it is now possible to recover using a scale out instance.

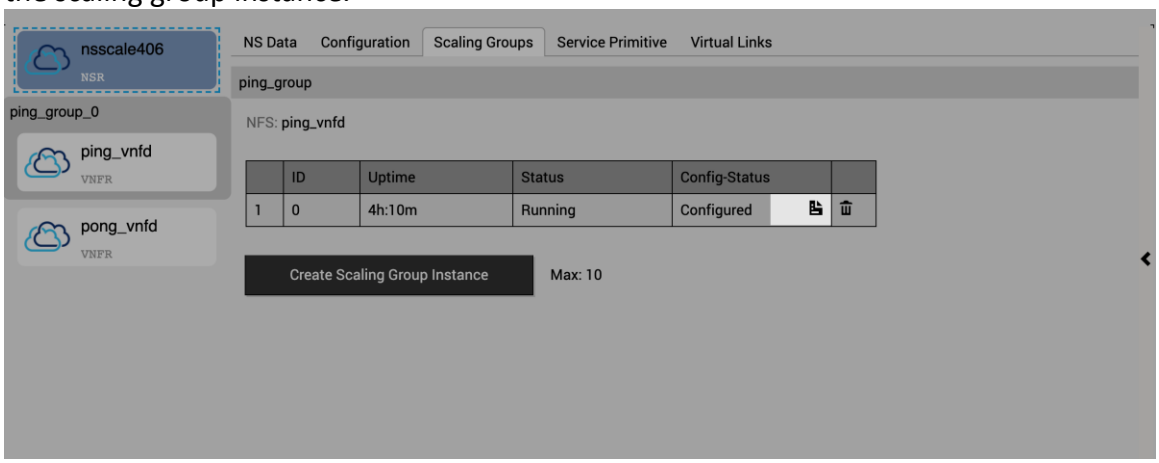




While adding a VNF to a Running Network Service, there is a new VNFD picker component to handle a large VNFD catalog. See the Add or Remove a VNF to a Running Network Service for more details on this process.

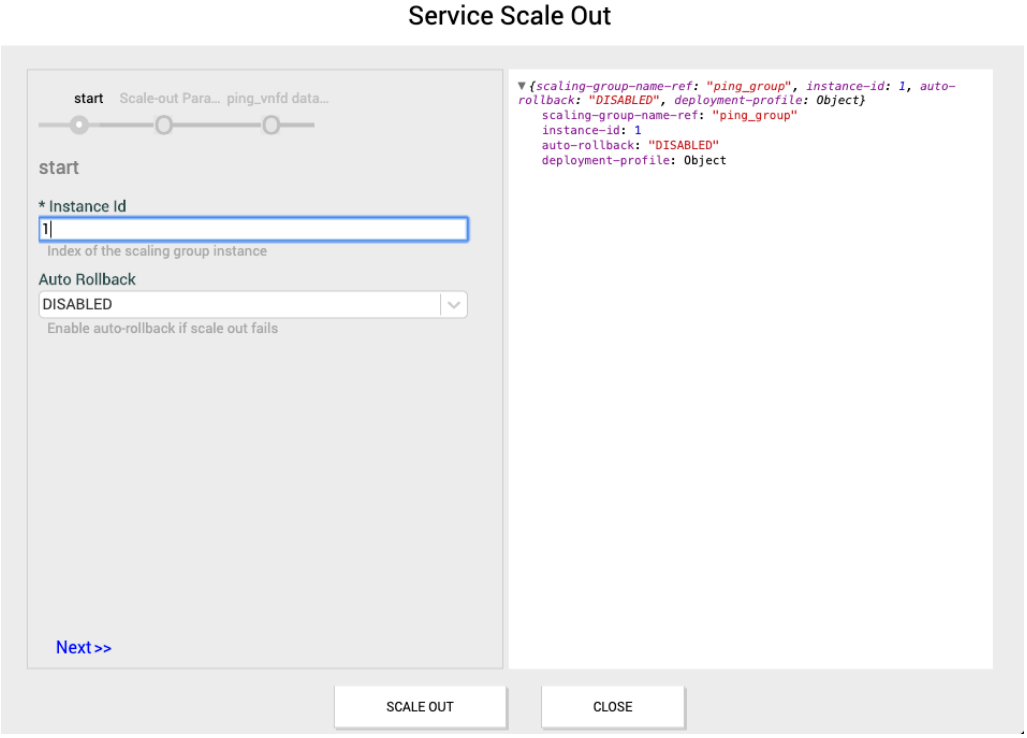


Scaling script output now appears on the Viewport screen. You can click an icon to open a new screen and display the results. The pre and post scale out config output is part of the scaling group instance.



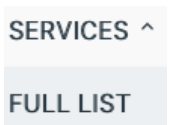
Post Scale Out :
Logs: 2020-02-13 05:42:05,541 INFO (ping_scale@30659:main.py:109) - Input data: {'cnfrs_in_group': {}, 'cnfrs_others': {}, 'config': {}, 'config_agent': {'name': 'RiftCA', 'type': 'riftca'}, 'instance_id': 0, 'nsr': {'id': 'b907bbb5-bdb0-455c-bd32-3e77c7e88d86', 'name': 'nsscale406'}, 'parameter': {}, 'rpc_ip': {'comment': 'Applying post_scale_out config for instance ScalingGroupInstance(ping_group #0)', 'lcm_event': 'scale', 'name': 'ping scale', 'nsr_id_ref': 'b907bbb5-bdb0-455c-bd32-3e77c7e88d86', 'project_name': 'default', 'scaling_group_name': 'ping_group', 'scaling_instance_id': 0, 'scaling_trigger': 'post_scale_out', 'scaling_vnfr': [{'id': '7e38d286-d5c8-4f2c-9264-bac7493eb2ab'}], 'user_defined_script': 'ping_scale.py'}, 'scale_instance': 'ScalingGroupInstance(ping_group #0)', 'trigger': 'post_scale_out', 'vnfrs_in_group': [{'connection_point': [{'connection_point_id': '320dca2d-e5e5-4007-b649-37b0bf95929f', 'ip_address': [{'port_ip_address': '10.5.0.155', 'public_ip': '10.68.100.157'}], 'name': 'ping_vnfd/cp0'}, {'connection_point_id': '3b93dbd7-516f-44e9-bcc2-8bd4fbadc6cf', 'ip_address': [{'port_ip_address': '31.31.31.11'}, {'port_ip_address': '31.31.31.3'}], 'name': 'ping_vnfd/cp1'}], 'connection_points': [{'connection_point_id': '320dca2d-e5e5-4007-b649-37b0bf95929f', 'ip_address': [{'port_ip_address': '10.5.0.155', 'public_ip': '10.68.100.157'}], 'name': 'ping_vnfd/cp0'}, {'connection_point_id': '3b93dbd7-516f-44e9-bcc2-8bd4fbadc6cf', 'ip_address': [{'port_ip_address': '31.31.31.11'}, {'port_ip_address': '31.31.31.3'}], 'name': 'ping_vnfd/cp1'}], 'dashboard_url': 'http://10.68.100.157:18888/api/v1/ping/stats', 'datacenter': 'acc1', 'id': '7e38d286-d5c8-4f2c-9264-bac7493eb2ab', 'member_vnf_index_ref': 1, 'mgmt_interface': {'ip_address': [{'ip_address': '10.68.100.157'}], 'port': 18888}, 'mgmt_ip_address': ['10.68.100.157'], 'mgmt_port': 18888, 'name': 'default__nsscale406__ping_group__0__ping_vnfd__1', 'rw_mgmt_ip': ['10.68.100.157'], 'rw_mgmt_port': 18888, 'vdur': [{'external_connection_point': [{'ip_address': [{'port_ip_address': '10.5.0.155', 'public_ip': '10.68.100.157'}], 'name': 'ping_vnfd/cp0'}, {'ip_address': [{'port_ip_address': '31.31.31.11'}, {'port_ip_address': '31.31.31.3'}], 'name': 'ping_vnfd/cp1'}], 'id': '608b5778-4e23-11ea-bd00-02420a40db01', 'management_ip': '10.68.100.157', 'name': 'default__nsscale406__ping_group__0__ping_vnfd__1__iovdu_0', 'unique_short_name': 'nsscale406-CR8RD6US9-iovdu0', 'vdu_id_ref': 'iovdu_0', 'vm_management_ip': '10.5.0.155', 'vm_mgmt_ip': '10.5.0.155', 'vm_name': 'default__nsscale406__ping_group__0__ping_vnfd__1__iovdu_0'}], 'vdur_data': [{'external_connection_point': [{'ip_address': [{'port_ip_address': '10.5.0.155', 'public_ip': '10.68.100.157'}], 'name': 'ping_vnfd/cp0'}, {'ip_address': [{'port_ip_address': '31.31.31.11'}, {'port_ip_address': '31.31.31.3'}], 'name': 'ping_vnfd/cp1'}], 'id': '608b5778-4e23-11ea-bd00-02420a40db01', 'management_ip': '10.68.100.157', 'name': 'default__nsscale406__ping_group__0__ping_vnfd__1__iovdu_0', 'unique_short_name': 'nsscale406-CR8RD6US9-iovdu0', 'vdu_id_ref': 'iovdu_0', 'vm_management_ip': '10.5.0.155', 'vm_mgmt_ip': '10.5.0.155', 'vm_name': 'default__nsscale406__ping_group__0__ping_vnfd__1__iovdu_0'}]}], 'vnfrs_others': [{'connection_point': [{'connection_point_id': '687a2c41-76f0-4cc9-87c8-763f51028211', 'ip_address': [{'port_ip_address': '10.5.0.120', 'public_ip': '10.68.100.104'}], 'name': 'pong_vnfd/cp0'}, {'connection_point_id': '24da3283-f2f7-4db0-a49a-04f6eb85d61a', 'ip_address': [{'port_ip_address': '31.31.31.6'}], 'name': 'pong_vnfd/cp1'}], 'connection_points': [{'connection_point_id': '687a2c41-76f0-4cc9-87c8-763f51028211', 'ip_address': [{'port_ip_address': '10.5.0.120', 'public_ip': '10.68.100.104'}], 'name': 'pong_vnfd/cp0'}, {'connection_point_id': '24da3283-f2f7-4db0-a49a-04f6eb85d61a', 'ip_address': [{'port_ip_address': '31.31.31.6'}], 'name': 'pong_vnfd/cp1'}], 'dashboard_url': 'http://10.68.100.104:18889/api/v1/pong/stats', 'datacenter': 'acc1', 'id': 'd0325f76-2807-4207-ab6d-db5740616783', 'member_vnf_index_ref': 2, 'mgmt_interface': {'ip_address': [{'ip_address': '10.68.100.104'}], 'port': 18889}, 'mgmt_ip_address': '10.68.100.104', 'mgmt_port': 18889, 'name': 'default__nsscale406__pong_vnfd__2', 'rw_mgmt_ip': '10.68.100.104', 'rw_mgmt_port': 18889, 'vdur': [{'external_connection_point': [{'ip_address': [{'port_ip_address': '10.5.0.120', 'public_ip': '10.68.100.104'}], 'name': 'pong_vnfd/cp0'}, {'ip_address': [{'port_ip_address': '31.31.31.6'}], 'name': 'pong_vnfd/cp1'}], 'id': '1a3e7426-4e23-11ea-bd00-02420a40db01', 'management_ip': '10.68.100.104', 'name': 'default__nsscale406__pong_vnfd__2__iovdu_0', 'unique_short_name': 'nsscale406-FmsJb2Lhp-iovdu0', 'vdu_id_ref': 'iovdu_0', 'vm_management_ip': '10.5.0.120', 'vm_mgmt_ip': '10.5.0.120', 'vm_name': 'default__nsscale406__pong_vnfd__2__iovdu_0'}], 'vdur_data': [{'external_connection_point': [{'ip_address': [{'port_ip_address': '10.5.0.120', 'public_ip': '10.68.100.104'}], 'name': 'pong_vnfd/cp0'}, {'ip_address': [{'port_ip_address': '31.31.31.6'}], 'name': 'pong_vnfd/cp1'}], 'id': '1a3e7426-4e23-11ea-bd00-02420a40db01', 'management_ip': '10.68.100.104', 'name': 'default__nsscale406__pong_vnfd__2__iovdu_0', 'unique_short_name': 'nsscale406-FmsJb2Lhp-iovdu0', 'vdu_id_ref': 'iovdu_0', 'vm_management_ip': '10.5.0.120', 'vm_mgmt_ip': '10.5.0.120', 'vm_name': 'default__nsscale406__pong_vnfd__2__iovdu_0'}]}]} 2020-02-13 05:42:08,129 INFO (ping_scale@30659:ping_scale.py:93) - Connect failure, retry after delay 2020-02-13 05:42:31,169 INFO (ping_scale@30659:ping_scale.py:93) - Connect failure, retry after delay 2020-02-13 05:42:54,204 INFO (ping_scale@30659:ping_scale.py:93) - Connect failure, retry after delay 2020-02-13 05:43:15,244 INFO (ping_scale@30659:ping_scale.py:93) - Connect failure, retry after delay 2020-02-13 05:43:35,284 INFO (ping_scale@30659:ping_scale.py:83) - Set server on service successfully 2020-02-13 05:43:35,304 INFO (ping_scale@30659:ping_scale.py:144) - Set ping rate successfully 2020-02-13 05:43:35,327 INFO (ping_scale@30659:ping_scale.py:184) - Enabled service successfully

In the Viewport → Scaling Groups section, there is now an option to add scale-out params when you click Create Scaling Group Instance.

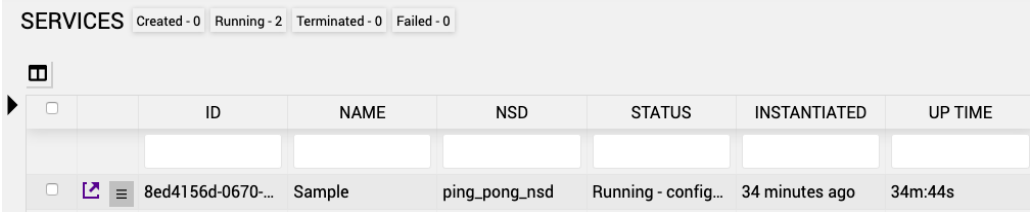


There is now an option to modify a running service by adding or removing a VL. While adding a VL, it is possible to edit the VL with specific customizations.

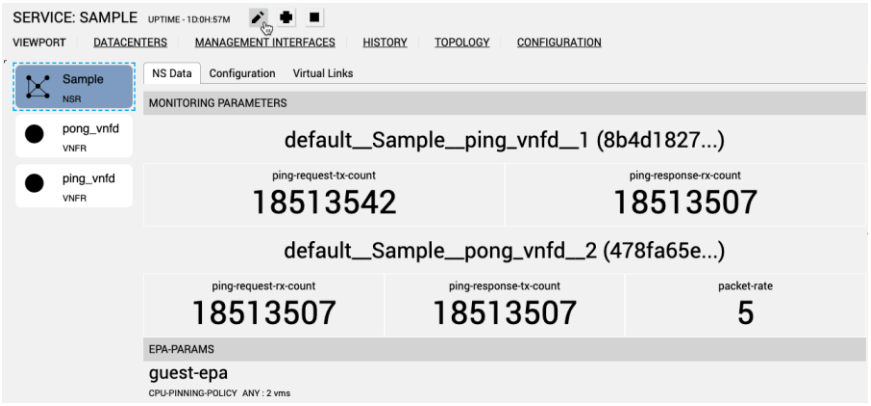
1. On the **Launchpad** menu, click **SERVICES** > **FULL LIST** to open the list of NS instances.



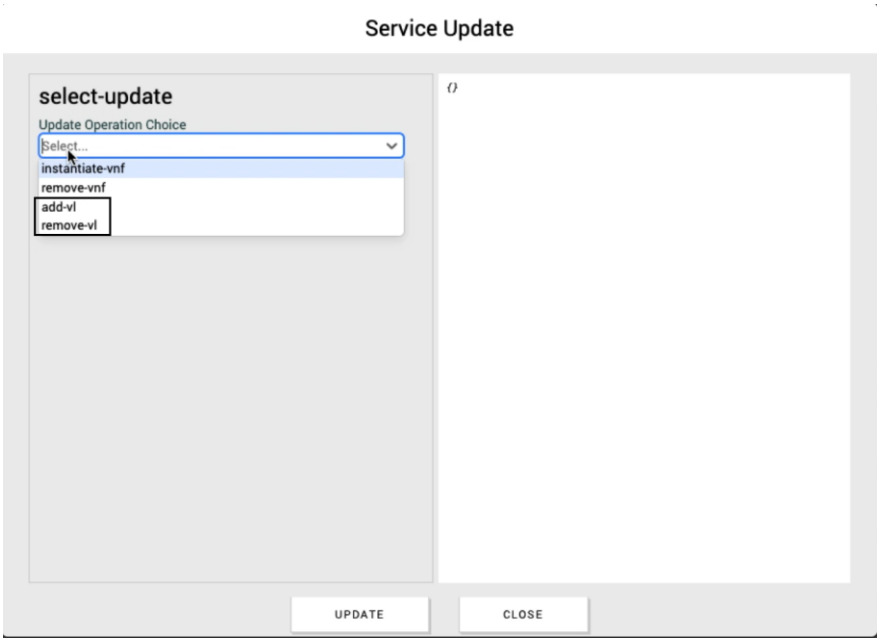
2. Click next to the NS that you want to see in more detail. The **VIEWPORT** screen opens.



3. Click to open the Service Update widget and provide the select-update details:



4. In the **Update Operation Choice** drop-down, either choose **add-vl** or **remove-vl**.



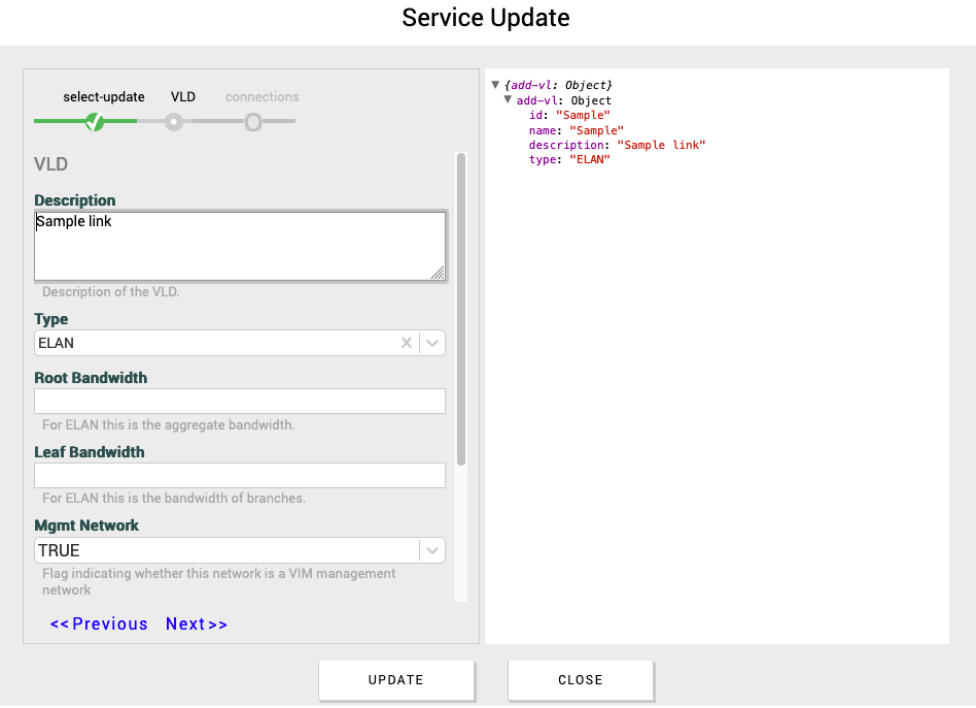
Insert a VL into a Running Service

This specific workflow is determined by the selected VNFD and NSD associated with the running service.

1. Provide the **select-update** details to insert a VL into a Running Service:

- **Update Operation Choice**: Choose **add-vl** to add a virtual link into a running service.
- **Name:** Name of the Virtual link.

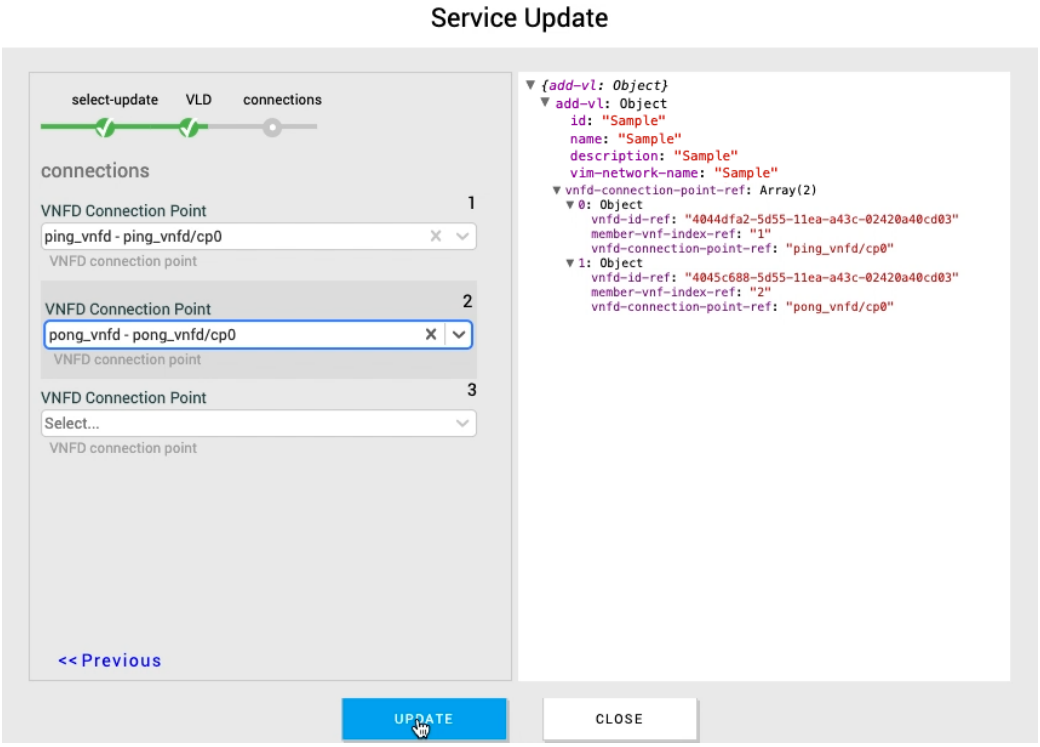
2. Click **Next >>** to provide the following details in the **VLD** step to identify the virtual links that you want to add to the service:



- **Description**: Description of the VLD.
- **Type**: Type of VLD.
- **Root Bandwidth**: This is the aggregate bandwidth for an ELAN link.
- **Leaf Bandwidth**: This is the bandwidth of branches for an ELAN link.
- **Mgmt Network**: This field indicates if the network is a VIM management network (True or False).
- **IPv4 Nat Pool Name**: IPv4 Nat pool name
- **NSD VLD Init Params**: The field determines the type of initialization parameters for VLD instantiation. The options include: vim-network, vim-network-profile or provider-network.
- **vim-network**: If you choose vim-network, then you must add the name of the network in the VIM account. This is used to indicate pre-provisioned network name in a cloud account.
- **vim-network-profile**: If you choose vim-network-profile, then you must add the IP Profile to use for VL instantiation.
- **provider-network**: If you choose provider-network, provide the appropriate details.
- **Physical Network:** Name of the physical network on which the provider network is built.

> **Note:** The requested details can change depending on what physical network you choose in this step.

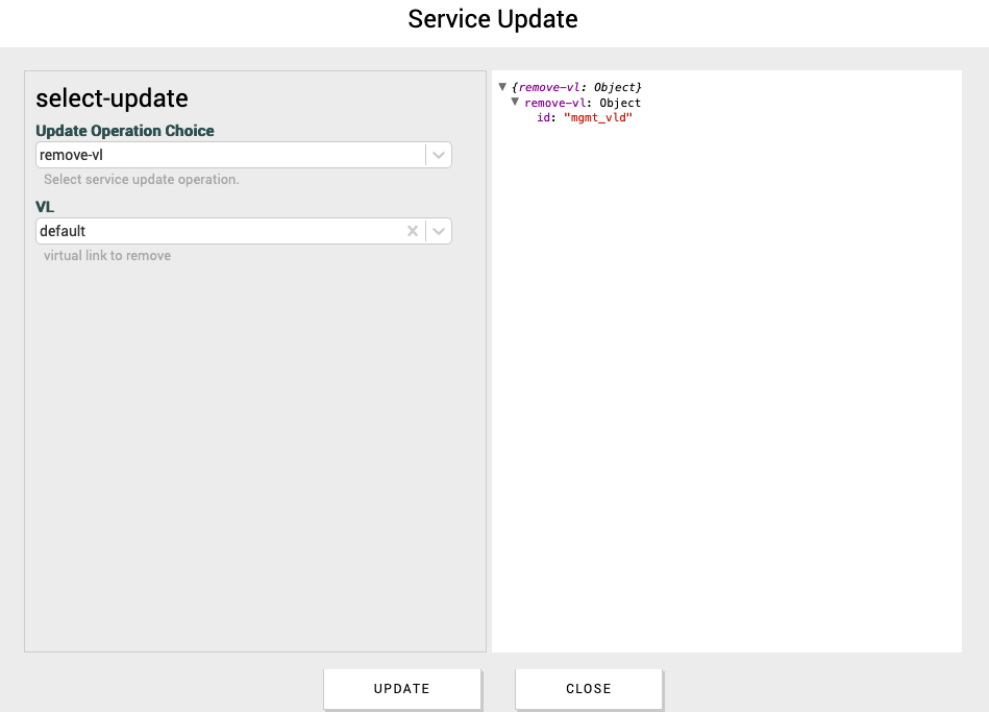
3. Click **Next >>** to provide details in the **Connections** step. The fields listed in the Connections section are the VLDs in the NSD. If you add connections, then the connections points are linked to a VL.



4. Click **UPDATE** and the insert progress bar appears.

Remove a VL from a Running Service

1. Provide the **select-update** details to remove a VL from a Running Service:
   - **Update Operation Choice**: Choose **remove-vl** to remove a virtual link from a running service.
   - **VL:** Choose the virtual link to remove from a running service.



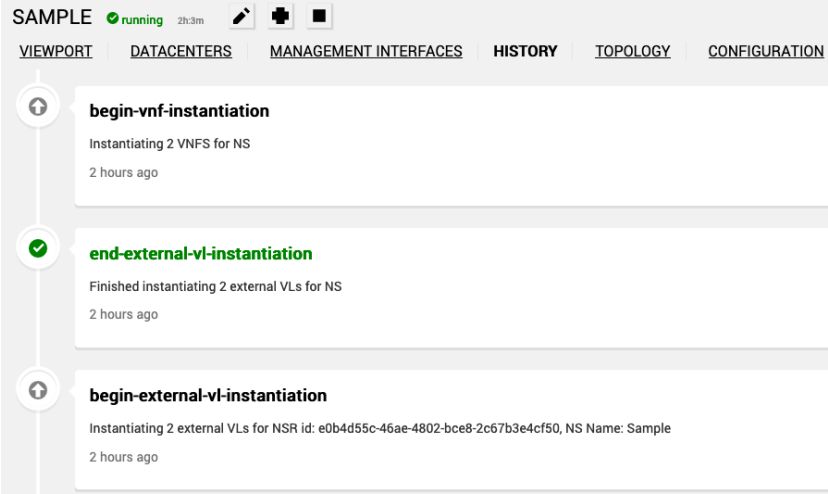
2. Click **UPDATE**.

Other VIEWPORT Tab Changes

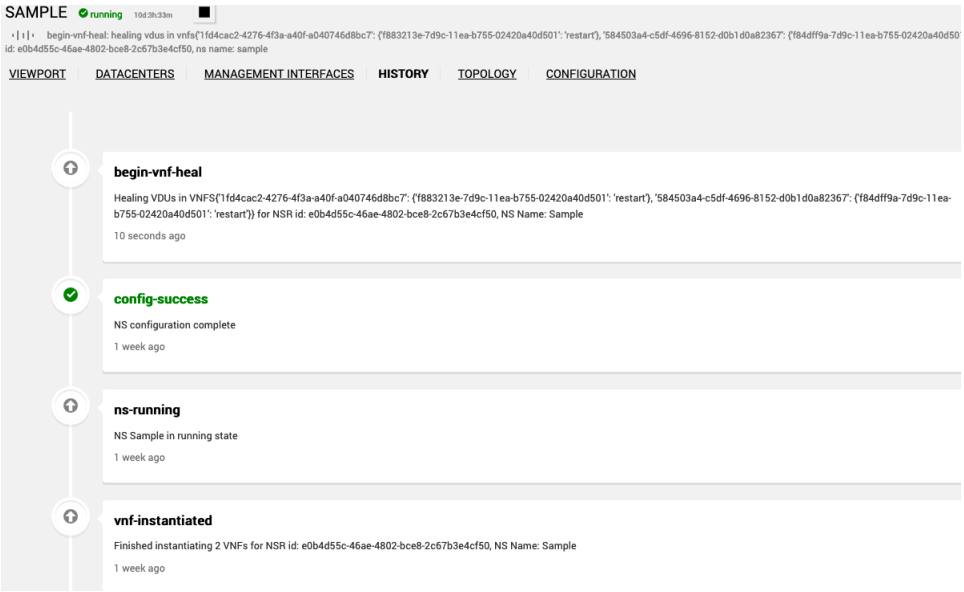
- If your service is in the **failed-temp – configured** or **heal-in-progress – configured** status, then the UI will not go back to progress view. It will stay on the Viewport screen.
- The **Take Action** button, the op-status of the service and the last operation status appears in the header.
- If an NS is not in the running state, then the scaling group (vnfr) status appears.

New HISTORY Tab

There is a new **HISTORY** tab in the **SERVICES** section (Services → History) to show the **nsr operational-events**. Scroll down to the bottom of the screen to see the 10 most recent operational events.



After performing a heal action, the UI now shows the status as Success or Failure in the **HISTORY** tab. See the Notifications and Event Viewer sections for more details about a specific event.

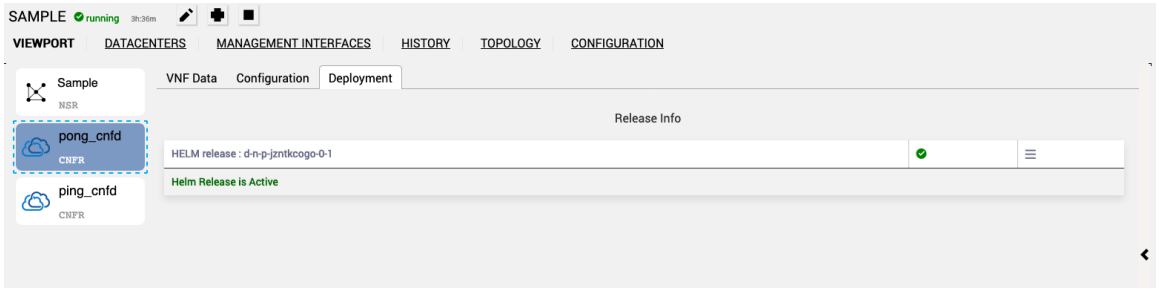


New DEPLOYMENT Section

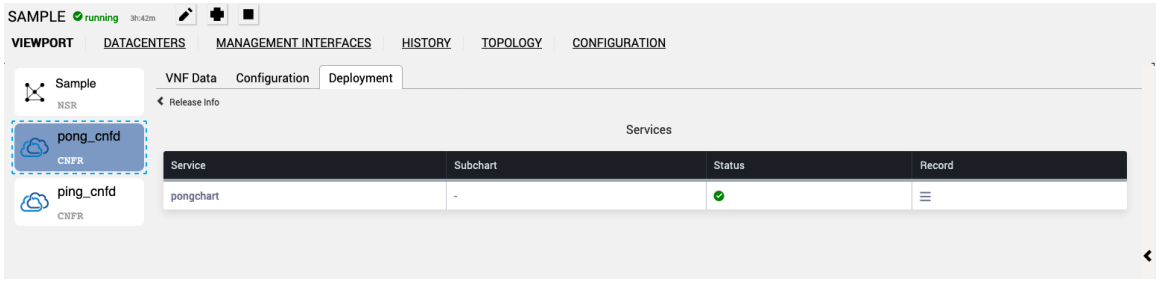
There is a new Deployment section in the Viewport screen. This tab is only for CNFR deployed services. This tab displays a summary of a specific deployed service. It shows the state of the service, deployments, pods, container and release info.

- Release Info: This section displays the release information of the CNFR.
- Services: This section displays the Service, Sub chart, Status and Record.
- Deployments: This section displays the Name, Namespace, Status and Record.
- Pods: This section displays the Name, Namespace, Status and Record.
- Containers: This section displays the ID, Name, Image, Status and Record.

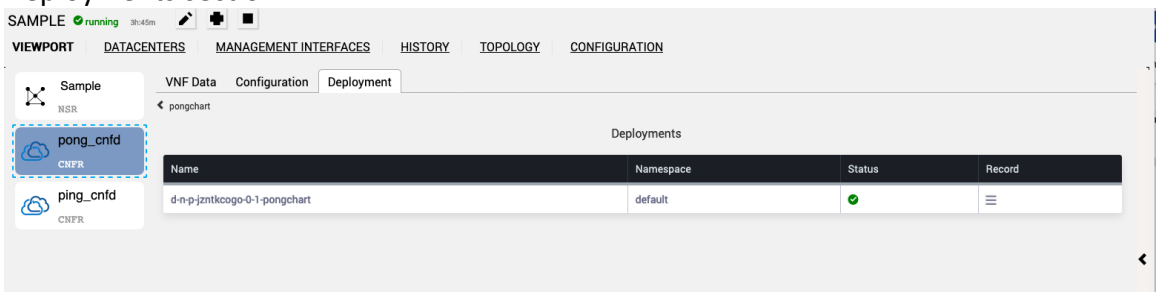
Release Info section



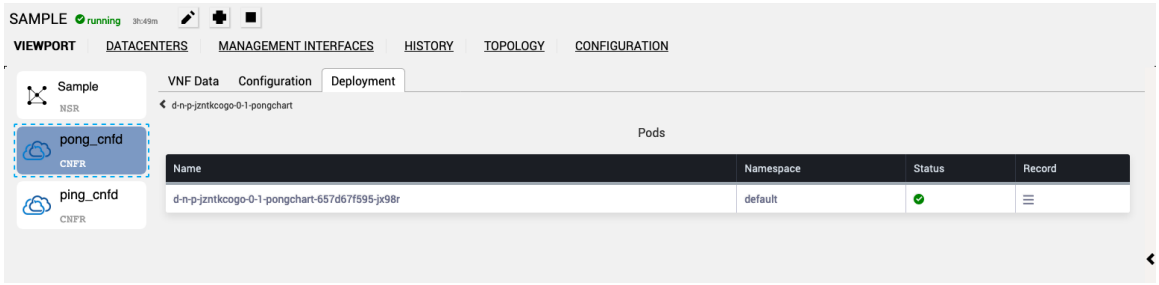
Services section



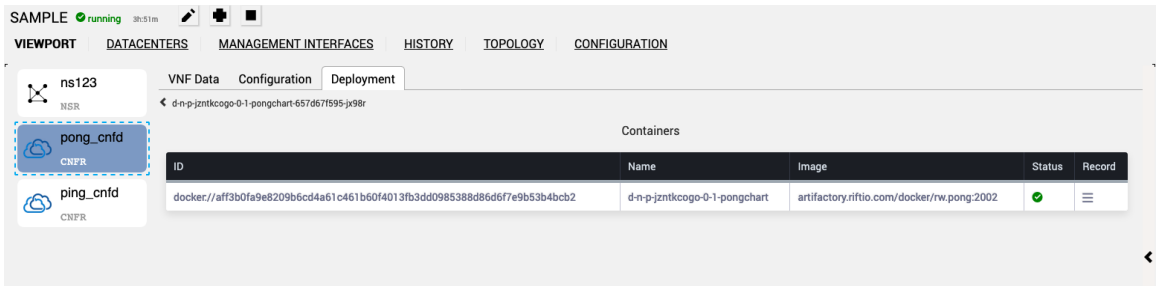
Deployments section



Pods section

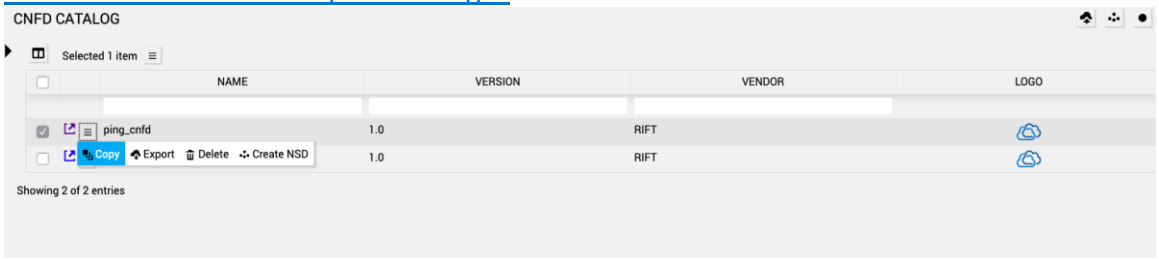


Containers section

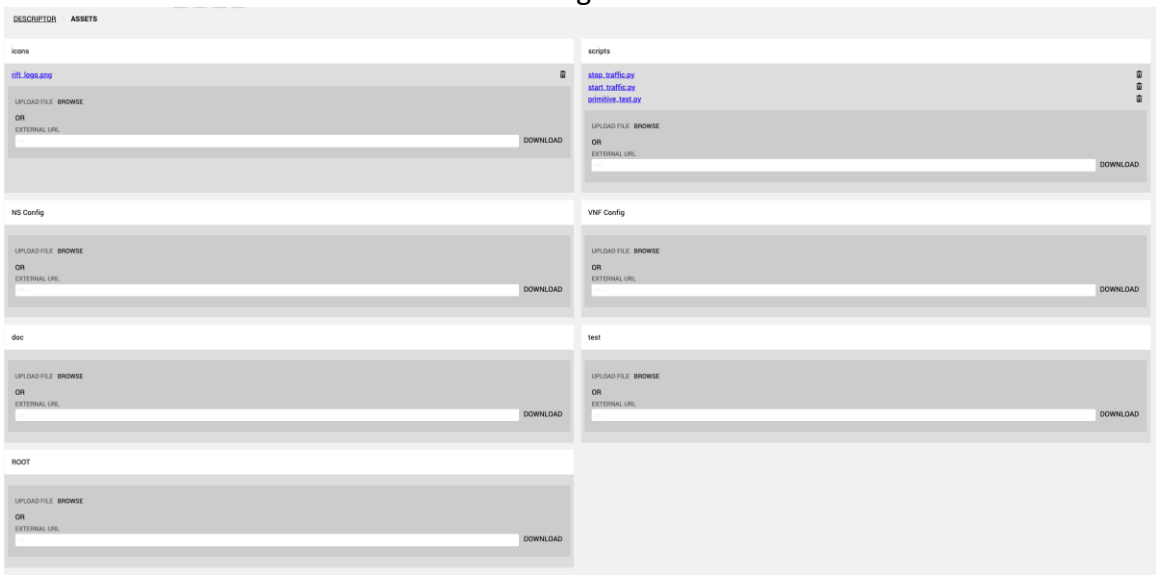


CATALOG Tab

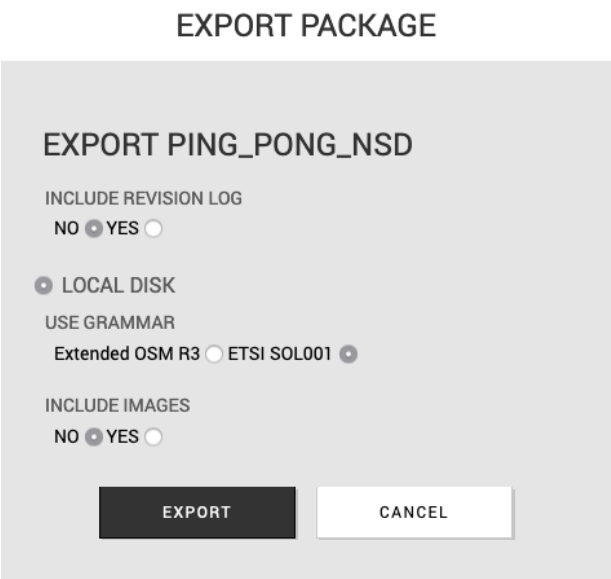
There is a now an option in the CATALOG tab to modify and copy a CNFD package. See Version Control of Descriptor Packages.



The **CATALOG → ASSETS** tab has a new design.

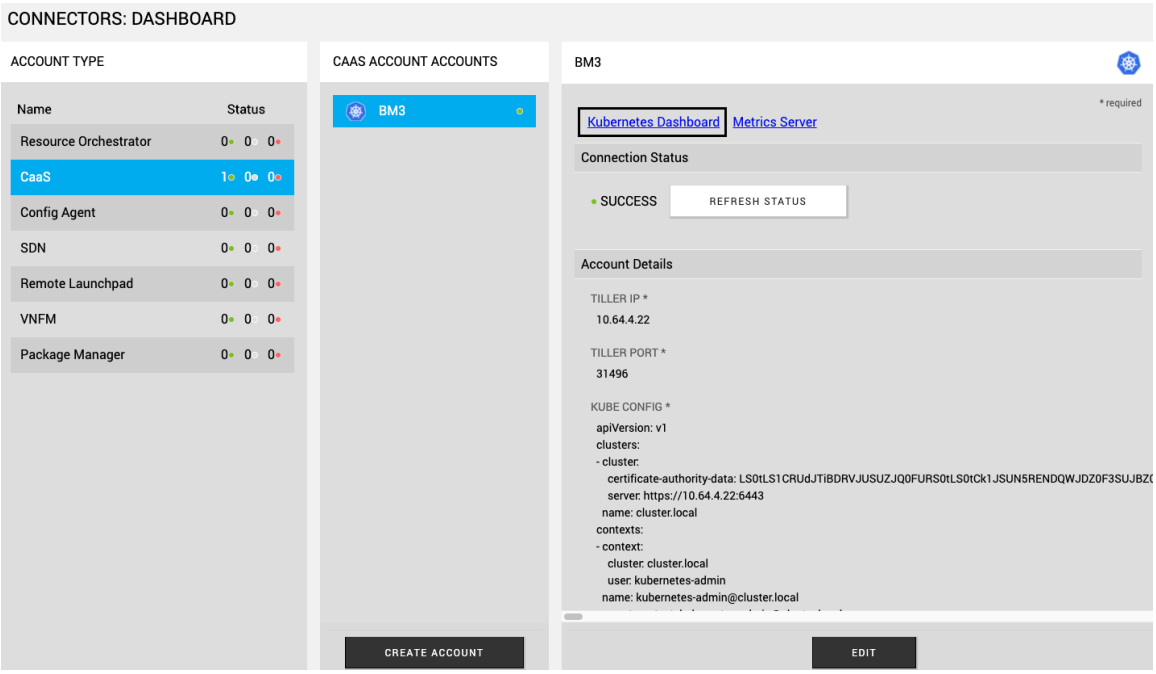


There is a new radio button to support exporting a package to ETSI SOL 001 for NSD. See Import and Export support for ETSI GS NFV-SOL 001 and SOL 007 NSD and NS Packages for details about exporting a package to ETSI SOL 001 for NSD.



CONNECTORS Tab

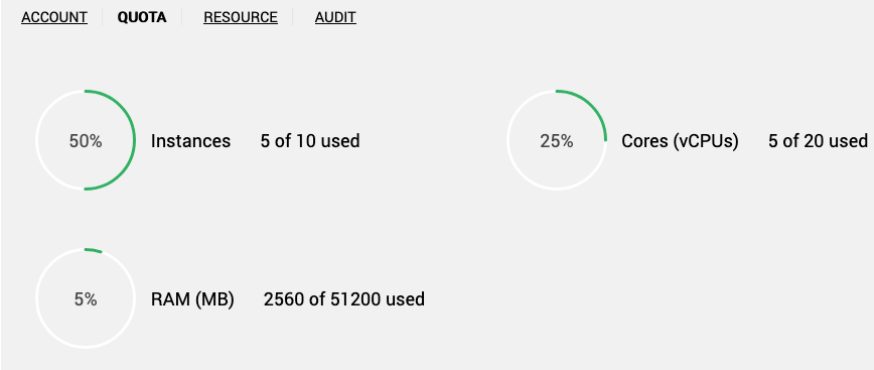
There is now a link to the K8s Dashboard on the CONNECTORS→ CaaS page for a created CaaS account.



DATACENTERS Tab

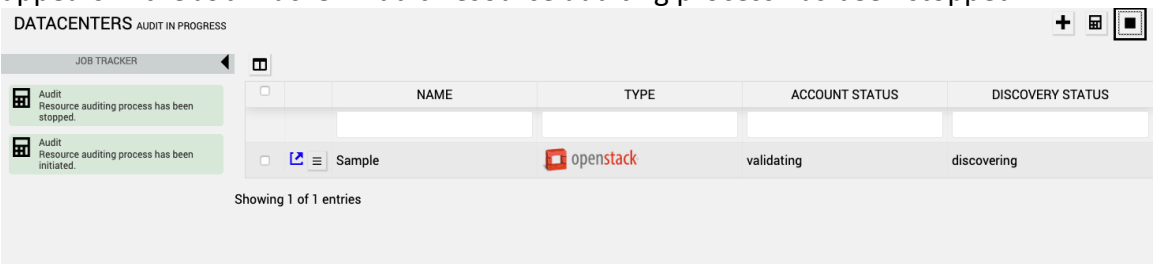
QUOTA Tab Changes

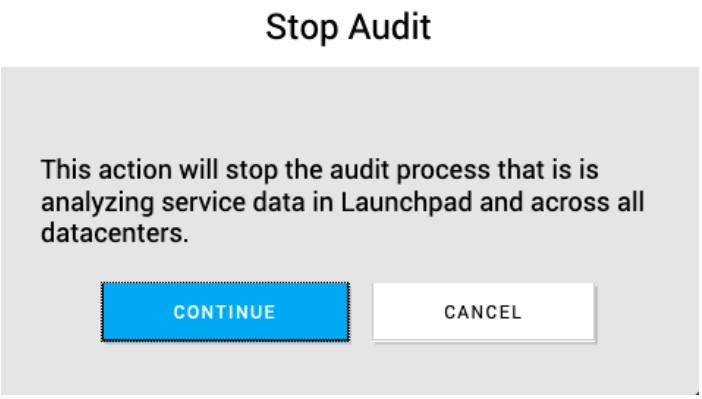
The **DATACENTER** tab now displays the show quotas graphic.



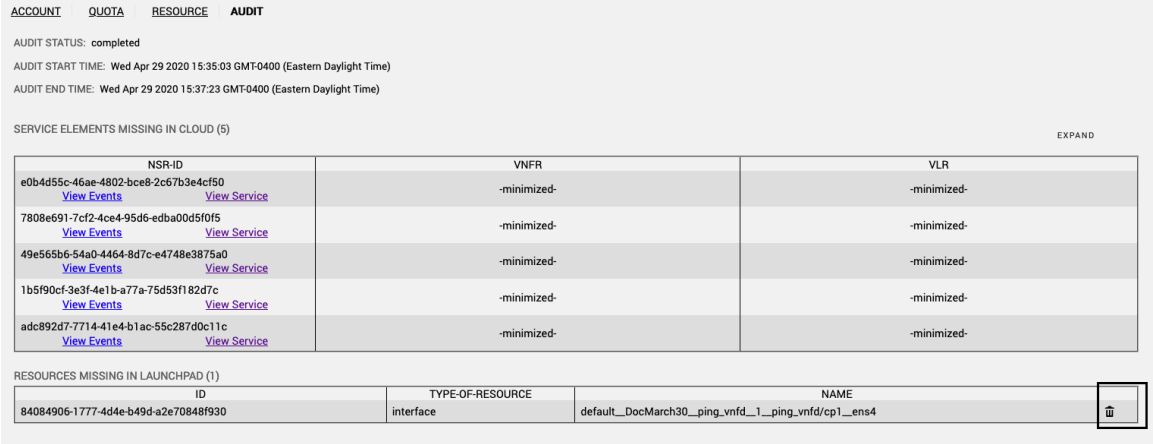
AUDIT Tab Changes

There is now an option in the **DATACENTER → AUDIT** tab to stop an audit in progress. **Click** ■ to stop an audit and then choose Continue to stop. The following message appears in the Job Tracker: Audit Resource auditing process has been stopped.





There is an option to delete VIM resources that are discovered as part of an Audit that are not associated to a Launchpad element.



Warning: While an audit is in progress, do not attempt to add existing resources into a running NS.

Life Cycle Management Scaling Enhancements

This feature enhances support for a large number of network services. Network Life Cycle Management v2 Operations are now queued and processed concurrently to increase performance.

API Enhancements

The RIFT.ware orchestration API has been enhanced to support large number of NSes. In addition, update network service operation now supports addition and remove of VLs with a running NS.

- Monitoring parameters are now retrieved using **nsr:fetch-mon-params**. Moved the `monitoring-param` field from **nsr:ns-instance-opdata** to **nsr:fetch-mon-params**.
- NS Instance Configuration details are now retrieved using **nsr:get-ns-running-config**.
- It is now possible to add and remove VLs using update-network-service-v2. Added `add-vl` and `remove-vl` fields in the **nsr:update-network-service-v2 Data Model.**

Platform Improvements for Scalability

This release includes improvements to the RIFT.ware orchestrator to meet the workflow needs of large deployments. The recommended configuration for RIFT.ware Launchpad is 16 VCPUs, 64 GB RAM and 500 GB disk and a minimum of 8 VCPUs, 32GB RAM and 80GB disk.

In release 8.1.0, the new method for v2 Life Cycle Management (LCM) for a Network Service (NS) is supported using only the below v2 APIs. Direct update of configuration as well as synchronous (v1) are no longer supported.

- Create a Network Service (NSR) – Use this API to create a network service.
  `https://<launchpad_ip/fqdn>:8008/api/operations/create-network-service-v2`

- Import Instantiation Variables – Use this API to import instantiation variables.
  `https://<launchpad_ip/fqdn>:8008/api/import-instantiation-variables-v2`

- Instantiate a Network Service – Use this API to instantiate a network service.
  `https://<launchpad_ip/fqdn>:8008/api/operations/insantiate-network-service-v2`

- Scale-Out Operation – Use this API to scale-out a service if an operation fails.
  `https://<launchpad_ip>:8008/api/operations/exec-scale-out`

- Scale-In Operation – Use this API to scale-in a service if an operation fails.
  `https://<launchpad_ip>:8008/api/operations/exec-scale-in`

- Heal Network Service – Use this API to heal a network service.
  `https://<launchpad_ip>:8008/api/operations/heal-network-service-v2`

- Update Network Service – Use this API to update a network service.
  `https://<launchpad_ip/fqdn>:8008/api/operations/update-network-service-v2`

- Terminate a Network Service – Use this API to terminate a network service.
  `https://<launchpad_ip/fqdn>:8008/api/operations/terminate-network-service-v2`

- Delete a Network Service – Use this API to delete a network service.
  `https://<launchpad_ip/fqdn>:8008/api/operations/delete-network-service-v2`

Retry and Rollback Support Improvements

This feature improves transaction-based support for retry and rollback at the NSR level for all NS operations. These enhancements include UI and API changes. The transaction now contains all the information needed for a retry operation. This ensures that information does not need to be retrieved from another artifact and the transaction also includes historical information.

Note: In release 8.1.0, retry for vl and vnf failures are supported. Configuration retry is targeted for a future release.

UI Enhancements

If an operation on a network service encounters an error, then the relevant operation may be retried or rolled back. Some operations can only be retried and cannot be rolled back. In the scale-out case, scaling-group-name can be used in place of transaction id.

Note: It is recommended to use the grouping scaling-group-instance-ref in the case of scaling operations.

Refer to the Create and Instantiate a Network Service section for more information on how to retry and rollback in the UI.

1. On the **Launchpad** menu, click **SERVICES** and select the appropriate network service from the catalog and click the Take Action icon.

   The **Take Action** screen appears.
2. Select the appropriate lcm operation.

3. Choose a retry or rollback attempt.

Note: The option to choose an LCM operation is only available when there are multiple instantiation failures.

4. Click **EXECUTE**.

Transaction API Enhancements

Enhanced the async-transaction-command in the nsr.yang model for retry and rollback operations. This release introduces new APIs to help with the management of NS transactions. A new get-ns-lcm-transactions API can fetch all transactions for a particular NSR. Each transaction record will have a transaction-status. If an operation fails, then a user can attempt a retry and/or rollback.

Prior to release 8.1.0, the transaction-id was the key for this command. Now, the async-transaction command uses a combination of the **transaction-id** and the **record-id** as the key identifier for the asynchronous transactions. While using Network Service LCM v2 APIs, a response is generated with transaction-id. An API client can use this information to attempt a retry or rollback operation

The following fields in the async-transaction command are new or updated in this release.

async-transaction Input Fields

Name	Type	Cardinality	Description
transaction-id			Identifier for the asynchronous transaction. The transaction-id field input replaces both heal-id (current heal details) and update-id (current update details). Previously, these were both IDs of async transactions. Now, there is no need to distinguish the type of transaction because this is part of the transaction record. **Note**: In release 8.1.0, the heal-id and update-id fields are removed from the ns op-data command.

record-id			Identifier for the record with which this transaction is associated.
is-retry-allowed	boolean	1	An indication that this transaction can be retried. This is a new fields in release 8.1.0.
is-rollback-allowed	boolean	1	An indication that this transaction can be rolled back. This is a new fields in release 8.1.0.
task	list	1	List of tasks run in the transaction. The initial-run, retry, rollback attempts on a transaction are now incorporated in this list under transaction. The first is the initial run and subsequent entries are for retry and rollback operations.
input-as-json	string	1	This is the deployment profile used to initiate the asynchronous LCM operation. It can be re-used if the action fails and is retried. It can also be used for historical purposes.

task

The task element updated descriptions for this release. This is the list of tasks that are run in the transaction.

Name	Type	Cardinality	Description
sequence-number	int16	1	The order of the tasks run in a transaction (initial-run, retries and rollbacks). The initial-run will have 1 and each retry/rollback will increment the sequence by one.
task-type	enum	1	The type of task carried out in each sequence
start-time	uint32	1	The start time of the attempt.
completed-time	uint32	1	The completed time of the attempt.
record-history	list	1	**Note**: This field was moved from the transaction level to the task level.
task-status	enum	1	Status of a particular run of the task (init, running, success or failed). The success/failure states of the last task will reflect in the overall transaction-status.

record-history

The record-history element has a new property timestamp.

Name	Type	Cardinality	Description
timestamp	uint32	1	The time of the status update.

As mentioned above in the Transaction API Enhancements section, the async-transaction command offers a choice of how an action is identified. You can either use the scaling group instance or transaction id. If you choose the scale-group instance option, then you must parse the corresponding scale-group name and instance-id.

Retry and Rollback APIs

If a user is attempting a retry or rollback, then the operator can complete these operations using scale-group-name and instance-id with the below APIs.

Retry

Input type 1

https://<ip-address>:8008/api/operations/lcm-retry

```
{
  "input": {
    "project-name": "default",
    "transaction": {
      "transaction-id": "83642d83-1bde-45f6-a181-1b2b9cc17269"
    }
  }
```

Input type 2 – Only for scale-out/scale-in

https://<ip-address>:8008/api/operations/lcm-retry

```
{
  "input": {
    "project-name": "default",
    "scale-group-instance-ref": {
      "nsr-id-ref": "366aa255-d7aa-4f6c-927b-9bab0167db8b",
      "scaling-group-name-ref": "ping_group",
      "instance-id": 1,
      "lcm-action": "scale-out"
    }
  }
}
```

Output

```
      {
          "output": {
              "status": "success",
              "error": ""
          }
      }
```

Rollback

Input type 1

https://<ip-address>:8008/api/operations/lcm-rollback

```
{
  "input": {
    "project-name": "default",
    "transaction": {
      "transaction-id": "83642d83-1bde-45f6-a181-1b2b9cc17269"
    }
  }
}
```

Input type 2 – Only for scale-out/scale-in

https://<ip-address>:8008/api/operations/lcm-rollback

```
{
  "input": {
    "project-name": "default",
    "scale-group-instance-ref": {
      "nsr-id-ref": "366aa255-d7aa-4f6c-927b-9bab0167db8b",
      "scaling-group-name-ref": "ping_grop",
      "instance-id": 1,
      "lcm-action": "scale-out"
    }
  }
}
```

Output

```
{
    "output": {
        "status": "success",
        "error": ""
    }
}
```

Support for Kubernetes as a VIM

RIFT.ware now includes the ability to use a service orchestration framework for cloud-native applications. The cloud-native application allows you to rapidly adapt to fast changing network conditions and utilize built-in microservice based scaling mechanisms as well as standards-based management and orchestration. Kubernetes is an open source project hosted by the Cloud Native Computing Foundation (CNCF). This technology allows a user to carefully examine provided functions, roles and behavior during normal run time and carefully break them down into a set of subfunctions or microservices that can be individually and automatically managed.

See Kubernetes Documentation for additional information.

RIFT.ware support for Container Orchestration (CNFD)

RIFT.ware uses a network service to instantiates a shared 5G slice subnet and the constituent 5G NFs. RIFT network services can be associated with slice meta-data and connect the new dedicated slice subnet to the shared slice subnet.

After instantiation, cloud-native NFs self-monitor to ensure that customer SLA's are being met. Degradation of a specific KPI will trigger the NF to automatically scale to meet demands. Once the demand has been addressed, then NF will return the added resource to the saved data center capacity.

If the network demand is too high and exceeds the capability of a single NF, then RIFT.ware orchestration is triggered to instantiate another NF on a different set of resources in the same or different data center. This new NF is automatically configured and inserted into the pre-existing subnet slice in a seamless manner.

UI Enhancements

The Datacenter section in the UI displays the Kubernetes cluster where a set of CNFs are instantiated. The service instantiation must be launched on the specific Kubernetes cluster that you created in the Connectors tab.

CaaS Connector Account

There is a new CaaS connector in the UI for containers.

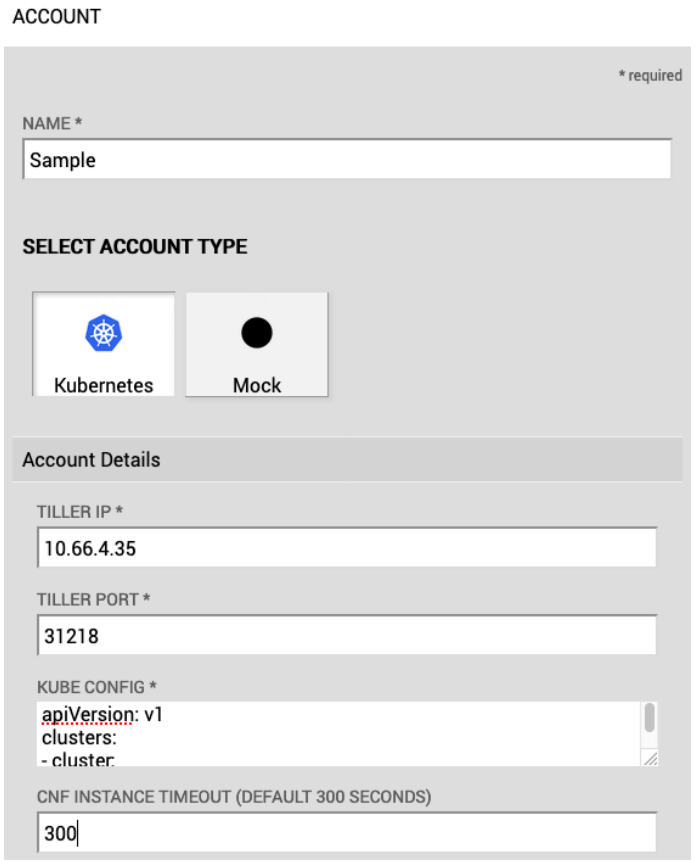
1. Sign-in to the Launchpad Dashboard menu and click **CONNECTORS**.
2. Under **ACCOUNT TYPE**, choose **CaaS** and click **CREATE ACCOUNT**.
3. Add the unique **CaaS** Account Name* name for the existing CaaS account you are linking to.
4. Under **SELECT ACCOUNT TYPE**, click Kubernetes.
5. Provide the following information (fields with * are required):
    - **TILLER IP***: Tiller service IP address in the Kubernetes environment
    - **TILLER PORT***: Tiller port in the Kubernetes environment
    - **KUBE CONFIG***: Kubernetes configuration file in YAML format. This configuration file must be provided by the operator.
      Sample KUBE CONFIG
      apiVersion: v1 clusters: - cluster: certificate-authority-data:
      LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUN5RENDQWJDZ0
      F3SUJBZ0lCQURBTkJna3Foa2lHOXcwQkFRc0ZBREFWTVJNd0VRWUR
      WUVFERXdwcmRXSmwKY201bGRHVnpNQjRYRFRJd01ESXhNREUyT
      VRRd05Wb1hEVE13TURJd056RTJNVFF3TlZvd0ZURVRNQkVHQTFVR
      QpBeE1LYTNWaVpYSnVaWFJsY3pDQ0FTSXdEUVlKS29aSWh2Y05BU
      UVCQlFBRGdnRVBBRENDQVFvQ2dnRUJBT3lxCjNyOHlZVm1RL01Ua
      043UEJ3Y3FBMmVFQjF6T2UveVpuSExaSjlsVWhtdmN3YWtrOHNmY05
      NelhqQVpZOGVsMGcKWUdRVThBRXRubHhEbTduOGZlQlRBTmVtbGs
      ybEplTW1HQVlGVGxrNkxsWldjZ0tQcS9sMWNweFNQaDkzc1VscApXNn
      BtTXgvQlM0aTFYOE1XV2NQY2h1c0MwSVE0MlpOYk9kMko4cDRsRk9
      QZnZpd1JlZG5OUjkzZGY1eXVJQjcwCnRhekdzYTRSOUg5djdRSzQ0Y2h
      vaFRUck4vQVg4WXIrWEtVRS81dFFPNk9OSk1TUmF3MWI1WERWK21
      5SXgrS0IKaEJzdk5uZ2dTL2UwbXd5Ny9ETzNVcFNkZllBTGRKYzRucml
      TcmtIM2kvU1hNVy9jTVdmV2VXNW1ZbFlhMGRsTQp6QzJvKzEwcTRiQ
      21PUy91ZEZzQ0F3RUFBYU1qTUNFd0RnWURWUjBQQVFIL0JBUURB
      Z0trTUE4R0ExVWRFd0VCCi93UUZNQU1CQWY4d0RRWUpLb1pJaHZj
      TkFRRUxCUUFEZ2dFQkFERVBBVc0tweWRlOHNjb0hyWjdLdi9SRmVnQ
      mwKZVl0NVprYkw2cDJsV2dWaUdCbUsxYmEycnhFbko3S1pxQU1kUE4
      ybjJ2Y1Q1dTBSRTRYSTNjUGNobE9SYVJrWQpPbVhYdTMzRTczYzlW
      VXdKcHZDTXQ0NzVPSVlSRHR0bXFpSVYzZzkvMUEzR2crMlZFdENZd
      Vp5VHFrNGtOa1NpCnJLcGJYWEYzcmE1cVExWlNuVEZCb2VrbXdmeG
      NvWGtXVmVzQjNTRjFoZmlFNGlCbGVsTU4rdFAwekVuYmxydkEKZX
      A0N3lPZWZkSnpNQnB4L2s3OGZiNHJVZGxXbktTWXdJdERTVVpxNF
      VxTEJtTmJmaHl3MXVhVXdJb0ZsL2YyZQpIa0lGM2FaR3p3YThqVTV2T
      FNMUTNlcytiUFJXSFJiNTB4bGg5MjFWSElNQUphEpZNmVwK1F0TEx
      iST0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtLS0tLQo= server:
      https://10.66.4.35:6443 name: cluster.local contexts: - context: cluster:
      cluster.local user: kubernetes-admin name: kubernetes-admin@cluster.local
      current-context: kubernetes-admin@cluster.local kind: Config preferences: {}
      users: - name: kubernetes-admin user: client-certificate-data:
      LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUM4akNDQWRxZ0
      F3SUJBZ0lJSUk2dEo1THk5YXN3RFFZSktvWklodmNOQVFFTEJRQXdG

VEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVGdzB5TU
RBeU1UQXhOakUwTURWYUZ3MHlNVEF5TURreE5qRTBNRFphTURR
eApGekFWQmdOVkJBb1REbk41YzNSbGJUcHRZWE4wWlhKek1Sa3dGd
1lEVlFRREV4QnJkV0psY201bGRHVnpMV0ZrCmJXbHVNSUlCSWpBTk
Jna3Foa2lHOXcwQkFRUZBQU9DQVE4QU1JSUJDZ0tDQVFFQXlCRUc
2Vk8wOWd6ZnVBL1UKeWpYUXU4aE5sRTdFL0cyK2F5TXg0WmtpYUJ
oU3NlRkVqWVMrU0ljZkFqaG1FbWVRTlFUYmI5emFDcmNXWTFTWgp
kWndkZGRtcWpSTnJpSzdmYlduWXJQRlZPRmc4aWlBN3ZzOWZEKzRL
UGNyMlNBUkRjUE9tN3NxSDBYUi90aGJ3ClU1T0xaVmx1bnNaamRtZ3Z
jdkxobGRWeUdIQjZObE54UWI0NTB2VTEyaU1vWi9UdURGRUs4UlA3Y
nVUakF1VUYKMkRGYzhVbUZRRWlWZzh4cVl6N2E4bWhtQzFZZ2JEa2
JoMmRjcnBIMzhTRk9obU5yRFQvbmVYNnFST2Z5VjdkQQp1NVZGdVk3
OVZuQW5vQTVrQXdwZjdnWGNJVXhEbHhhalY1MXNZMWUzT3dqWU
MvOU5kQzk1NWZ0QS9zL2JSZG1XCjd4cElxd0lEQVFBQm95Y3dKVEFP
QmdOVkhROEJBZjhFQkFNQ0JhQXdFd1lEVlIwbEJBd3dDZ1lJS3dZQkJR
VUgKQXdJd0RRWUpLb1pJaHZjTkFRUxCUUFEZ2dFQkFGWngvRWtu
Mm1kZERNM2xtR1BDL0tvcGp3UWN5RGRSK0JrQQpaZU9wamFWNkdz
VUVPUEVJcjZsU3d6U0JGK29BKy9TT0d5SFJGMGlyR3EzM1J5d1JCUDd
LNTZqd3pEdFhPQzRkCnQwZUtBM2lDSW4vT2FMSithNERnRGZ1RExD
QTZsNUl0MzFSOWFITk9QVi9LUmpYSGNLc2dXSjdoNXQ3YzdrUkgKS
EpRaytaWFl1NTZIcXYyU09WQ0I3NUFMV1VsOFlEcHAxWnFvOG9IZH
VLcnkxRmgvNlJOWjM2a1hFSGg1V1hoeQpod3VFaFZ1c1d5d2dTR25mMj
dmSVJKTm9Oc2lMdFJHQ1cwWDdLMk5sc0tlY1lld1JuLzl0VWU3Q1d3Zk
prNjR2CkFmWVpPbkFXcVlsRU52c2Q4WjRrbE1XUWptUXdJaHRQQTVC
UVFrY1E4TnlRQS8wTWVaTT0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtL
S0tLQo= client-key-data:
LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSUlFb3dJQkFB
S0NBUUVBeUJFRzZWTzA5Z3pmdUEvVXlqWFF1OGhObEU3RS9HMith
eU14NFpraWFCaFNzZUZFCmpZUytTSWNmQWpobUVtZVFOUVRiYjl6
YUNyY1dZMVNaZFp3ZGRkbXFqUk5yaUs3ZmJXbllyUEZWT0ZnOGlpQ
TcKdnM5ZkQrNEtQY3IyU0FSRGNQT203c3FIMFhSL3RoYndVNU9MWl
ZsdW5zWmpkbWd2Y3ZMaGxkVnlHSEI2TmxOeApRYjQ1MHZVMTJpT
W9aL1R1REZFSzhSUDdidVRqQXVVRjJERmM4VW1GUUVpVmc4eHFZ
ejdhOG1obUMxWWdiRGtiCmgyZGNycEgzOFNGT2htTnJEVC9uZVg2cVJ
PZnlWN2RBdTVWRnVZNzlWbkFub0E1a0F3cGY3Z1hjSVV4RGx4YWoK
VjUxc1kxZTNPd2pZQy85TmRDOTU1ZnRBL3MvYlJkbVc3eHBJcXdJREF
RQUJBb0lCQUQzbk50SS9Pa0RmbnVGZQoxS0owb0U5YUhOY0V3R0t5d
WphQVAxRGtod2JhYjh2bVBjWGtDdFI2S3BnUFIxWHVnV3BHUkhlS1N
GVUViWUx5CmpFRWR1dWZhSDdGakR1VE41d2Y5dGpOY3dOZWNSY
ThmWmgwV0lQM1cwNGhuSW1rS0dLdjhWK3lPOGhOWERxVVMKMSs
ya08zOXllbzNhS0xCY2NFSUs4TTNocTFudjlxN2VkN1p2aEFkc1lBS1FNe
UZyU2ZzSWZUM0liUzcrN200Vwo0WjF6d3diLzJmWjQ2WDJCQllNbmRx
UFpFeElhT0o1YkdvdWl3d1pyZWlYUlNCSlk1UVNLTGI4clNiRW9pOUdV
CjhLbzZmd3BMZVM1K2VCMTRKam5BS3MxbG1PdW1UUVoxZ05RSXd
qT3IrVGMyMnN1dWZDUGJMQmpsRnE3RHJHYnkKSi9WRmtRRUNnW
UVBKzA5Mk9udlY5TDloUjduallhNHpZRk1wMzU0S1MzWjBLcGsvQVJV

NHVCa3hZWUNUb3JGdgpGNHkyRjI2TXRQTzlPU3F1MDdlbHh4SGF1N
UFhU0NHVGlkdlJleFFsa2ZTQW9CUFNHTFhYZHhwc05ZakNYSWErCitJ
OG9aaTFiQ054SXhtY2dYdXNNemp0UEp6d2RjQ2g4UHB4dlVMTS9OT08
0UGZpeU1GYTg0V3NDZ1lFQXk4ekUKYi9OaisvZjV2ZEFPTjEzTE5ORT
hFbXpscFJIM05EaVpLNnFKdm56TzFGeWVXZUlRSDl1bHdKSFRFZnV1
UjdtdAp1QUpKeTIyZXRqMWlESldocUJ5MlIxQXpsdHRMcVV3S2Z4M3F
WSlZCeXZ3M25zTW9QTEZWZTZPZ04zZWZYamp4CnA5eFNod1hZRXJ
obDZmMkFsZmlvcDVnUnVQemowRjlUZjluM3hjRUNnWUFTYzhjdWpD
RXRrMC9GUUhUZG16alMKZmNpNVRwVmE3UnVpZFJZWE1rT3o0TVd
yYnhGb0JMY3ZkM2wzUnZxMTNwK3FMdVFmVDRDd21UTUNTcFpqYl
hPegp6NXdWK1dpNHlzY1crZDJYU3VMRE1BRjUvTXlvbG93M0crdlBkT
TBXWFhaS2V3LzVhRERNZzdaUUh5M2FLbDdWCkNnOXlVWGIybnpIT
Ct4SzZVZVFVZVFLQmdFR2lGbmsvVnlua2VJZUtvNmx0Q1c1ZkhBdUxD
b0lZd1JZT1REWGwKM0NwK28xVjg1bnBuOVcxdEhYcGcvQ3JFZFNJZ1
NEUUVlS09ORUJuOHRzRDZ4MjhPb0IxalZObjJrZTFaMGQrUwpVTThnc
W1tbXhIcmF1dkNnNDdqSHYyZGRuUE9KUHpvaUdHbm5sZWloZmlEWF
JUd0hNcm1XbmM2SGt3NmVSSlBlCkxJK0JBb0dCQVBCL05qaGVJK1RE
WVFLbmxpb1J1dmllQWNLTS9id1d5Y3NHQ3J2Nnl2N2p6Y2phNnZ5REJk
dFYKM0c0cGtpUFdDdGNxUm9xZ0tIY3FON1FoWkdaS1hrSjJldy9zYWVC
NzFuYzBFQnJod0kwb1NSOXdvVEphVXhkNgpzajR4MVdvVUpLVDlVTlJ
0NzREaXY4aEhtaGNCOFVKV3dGMU1RdHNlNUlKRlEzSGNKY3p5Ci0t
LS0tRU5EIFJTQSBQUklWQVRFIEtFWS0tLS0tCg==

- **CNF INSTANCE TIMEOUT (DEFAULT 300 SECONDS)**: Maximum time allocated for resource instantiation in the CaaS account. Default is 300.

6. Carefully check your entries and click **SAVE**.

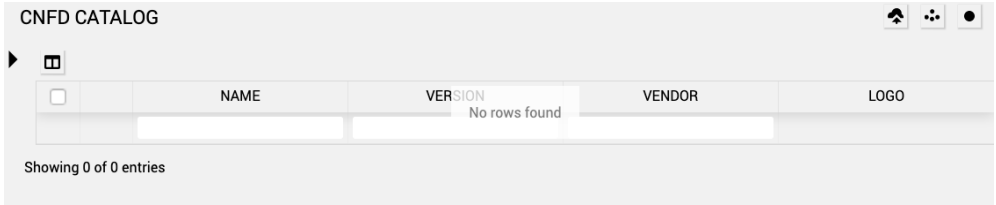


Note: In order to instantiate a sample NS using Kubernetes as a VIM, it is necessary to upload CNFD descriptors. See CNFD Descriptors section.

CNFD Descriptors

Note: You must add NSD and VNFD descriptors on the Catalog > NSD List page.

1. Click **Catalog** > **CNFD LIST**.
2. The CNFD CATALOG page opens



On this screen, you can filter for CNFDs by **NAME**, **VERSION** or **VENDOR**. If you want to open multiple or specific CNFDs, then click the white checkbox button to the right of the arrow.

3. Click  to import the CNFD descriptors.
An **IMPORT DESCRIPTOR** screen appears. An operator can type a **FILE URL** or browser to the CNFD descriptor files.
4. Click **IMPORT**.

API Changes

The following APIs are new or updated in this release.

- Get CNFD Catalog Items
- Add a New CaaS Account
- Query the CaaS Account Validation Status
- Refresh the CaaS Account
- Delete a CaaS Account
- Transaction Status

Model Changes

The nsd now has the ability to reference constituent-cnfds. The VLD section can now attach to cnfd-connection-points as well.

- CNFD Model
- CNFR Model
- MANO Enhancements

CNFD Model

cnfd data

- cnfd:cnfd-catalog
  - cnfd:cnfd
    - cnfd:config-data
    - cnfd:chart-info
      - cnfd:helm
        - cnfd:interface
    - cnfd:microservice
      - cnfd:interface
    - cnfd:connection-point
    - cnfd:mgmt-interface
    - cnfd:input-variable
    - cnfd:http-endpoint
      - cnfd:headers
    - cnfd:monitoring-param
      - cnfd:query-params
      - cnfd:json-query-params
      - cnfd:numeric-constraints
      - cnfd:text-constraints

CNFR Model

rw-project data

- cnfr:cnfr-catalog
  - cnfr:cnfr
    - cnfr:cnfd

- cnfr:config-data
- cnfr:chart-info
  - cnfr:helm
    - cnfr:interface
- cnfr:microservice
  - cnfr:interface
- cnfr:connection-point
- cnfr:mgmt-interface
- cnfr:input-variable
- cnfr:http-endpoint
  - cnfr:headers
- cnfr:monitoring-param
  - cnfr:query-params
  - cnfr:json-query-params
  - cnfr:numeric-constraints
  - cnfr:text-constraints

- cnfr:mgmt-interface
  - cnfr:ip-address
- cnfr:connection-point
  - cnfr:ip-address
- cnfr:caas-master-ip
- cnfr:release-info
  - cnfr:service-info
    - cnfr:deployments
      - cnfr:labels-for-pods
      - cnfr:pods
        - cnfr:interfaces
          - cnfr:ip-addresses
        - cnfr:containers
          - cnfr:state
        - cnfr:pod-events
    - cnfr:statefulsets
      - cnfr:labels-for-pods
      - cnfr:pods
        - cnfr:interfaces
          - cnfr:ip-addresses
        - cnfr:containers
          - cnfr:state
        - cnfr:pod-events
    - cnfr:other-objects
- cnfr:http-endpoint
  - cnfr:headers
- cnfr:monitoring-param
  - cnfr:query-params

        - cnfr:json-query-params
        - cnfr:numeric-constraints
        - cnfr:text-constraints
    - cnfr:operational-events

MANO Enhancements

- The pre-existing **provider-network** grouping in the mano-types.yang (being used in VNFD, VNFR, VLD, NSD, VLR) have new elements in this release.
- The pre-existing **ip-profile-info** grouping in the mano-types.yang (being used in NSR, VLR) has a new **"end-address"** leaf.
- NSR (Operational Data): The model has been enhanced to accommodate a list of constituent-cnfr-refs. The **vnf-init** operational state for the NSR has been changed to **nf-init**.

Fixed Issues in RIFT.ware 8.1.0

Support Tickets	Title	Description
RIFT-23681	VIM Account connection status shows failure which was a success before.	A VIM account was created successfully but then an error message appeared showing 'Failure connection status'. Refreshing the account did not fix the issue. This issue it closed. It is not observed in release 8.1.0.
RIFT-24509	Duplicate registrations created for RPCs for rw-image-mgmt.yang, rw-config-agent.yang, rw-cloud.yang	Most RPC registrations are done under project. This caused RPCs to be registered multiple times whenever an operator added projects. This issue is resolved.
RIFT-24990	Launchpad is crashing with an error- application communication failure.	Launchpad had 3 projects with brownfield discovered instances on all the projects. Launchpad stopped running when the user was checking the status of the network service. This issue it closed. It is not observed in release 8.1.0.
RIFT-25384	Kafka account in error state failover after an upgrade.	A user configured a new Kafka account in Launchpad. After a failover occurred, the Kafka account failed on the new Launchpad. This issue it closed. It is not observed in release 8.1.0.
RIFT-25619	NS is marked terminated successfully even when VM is still in deletion phase in VIM	This issue it closed. It is not observed in release 8.1.0.
RIFT-26478	Audit button should be disabled when no datacenter is configured.	The audit button was enabled even if no datacenter was configured. This issue is resolved.
RIFT-26494	Server Error 500 while deleting datacenter associated with SDN account.	A user created two SDN accounts with valid credentials and then navigated to the datacenters page. The operator tried to create one by selecting OpenStack → corresponding SDN account. After selecting the datacenter page, the user tried to delete the failed datacenter account and an error message appeared.

Support Tickets	Title	Description
		This issue is resolved.
RIFT-26571	Placement groups: UI needs to provide a way override discovered details.	The placement group configuration of the Instantiation form depends on information that the discovery process uncovers. This discovery process checks the selected datacenters. In addition, the credentials provided in the datacenter account configuration must have admin privileges in order for the information about these placement options to be gathered. If this information was not discovered, then the vnf deployment placement feature could not be used. This issue is resolved.
RIFT-25679	Exceptions in console during switching of project	Exceptions appeared while a user was switching from one project to another. This issue it closed. It is not observed in release 8.1.0.
RIFT-26071 RIFT-28432	MANO should not log event TAGGING FAILED for every resource	Logs were printed for every resource. This issue is resolved.
RIFT-26229	No Name for the instances in the AWS	The name of the instance was missing in the AWS web console. This issue is resolved.
RIFT-26779	Logs for vnf initial config scripts not appearing under vnfr-catalog	An operator created a service. Then the user added a new ping VNF in the running service using the update option. The NS came up with the new VNF. Next, the operator went to the VNF and clicked on the configuration tab > 'Ping_initial_config.py' script present in Day 1 - initial config primitive.' The script did not run for the original VNF present in the service. The output did not appear correctly in the UI. This issue is resolved.
RIFT-26823	Warning logs are not shown in UI or terminal without critinfo	Warning logs were not shown in UI or terminal without critinfo. This issue it closed. It is not observed in release 8.1.0.

Support Tickets	Title	Description
RIFT-27254	Incorrect error message for failed CNF instantiation	An operator used an incorrect value yaml file for the instantiation of a CNF NS. This resulted in the PODs failing to deploy with an error from K8S. Launchpad incorrectly showed the instantiation failure as a 'timeout' error. This issue is resolved.
RIFT-27270	NS termination workflow does not wait for all pods to be deleted	An operator launched ping pong in K8S and then terminated the NS. The NS termination workflow did not wait for all K8S pods to be deleted and completed before verifying all pods are deleted. This issue is resolved.
RIFT-27680	Sol002 Heal operation status was not completed even after the LP completes the Heal	An operator attempted a Sol002 heal operation process for 3 instantiated Network Services. The heal status completed in Launchpad but the status for some of the VDUs was 'in process'. This issue is resolved.

Known Issues in RIFT.ware 8.1.0

Support Tickets	Title	Description	Impact	Workaround
RIFT-13715	Confd configuration transaction abort results in inconsistent state.	Confd can abort configuration change transactions due to its own internal reasons. RIFT.ware cannot undo the changes (on an ABORT form Confd) because it is already internally committed.	This could result in a data mismatch between that is there in the configuration database and what is known to the RIFT.ware backend.	This issue mostly occurs when a user makes successive config changes without any idle time in between the modifications. Any test or automation script using RIFT REST APIs to complete configuration changes must have a delay between two successive config change operations.
RIFT-19249	RBAC API to Update user's project/MANO roles has changed.	RBAC API to Update user's project/MANO roles has changed. This information needs to be updated in the User Documentation.	An older REST API cannot be used to change the User's Project/MANO roles and might break existing scripts.	Use the Launchpad UI to change a user's Project/MANO role, or use the New API.
RIFT-21978	Password is displayed in plain text in event logs.	A password is displayed in plain text in the events log.	A password is displayed in plain text in the events log.	N/A
RIFT-21923	TC_CONCURRENT_NS_TERMINATE: PackageDeleteError	When multiple descriptor (NSD/VNFD) packages are deleted concurrently using an API, the config data in the ConfD might be missing for a brief period for a few packages.	The config data in the ConfD might be missing for a brief period for a few packages.	When using an API to delete multiple package entries, do not send multiple DELETE requests concurrently. Wait for a request to complete before firing another request.

Support Tickets	Title	Description	Impact	Workaround
RIFT-23621	Introduce a new field in VNFM account page for RIFT.ware<-->VNFM handshake URL.	This is a request to add a new field in the VNFM accounts page that the Operator so the operator can populate the URL for handshaking. If the URL field is empty, handshake is not required.	RIFT.ware uses a GET call to /vnf_instances to validate VNFM Account. If a SVNFM account does not support GET on /vnf_instances, then the VNFM Account validation may Fail. The UI shows the account status as failed. If the SVNFM account supports notification, then RIFT.ware sends a subscription request next as part of the validation. This is successfully and the VNFM account status is successful. Therefore, a SVNFM account which doesn't support GET call on /vnf_instances, then the VNFM Account status will appear to fail. This will not block any further operations such as instantiation and termination.	N/A
RIFT-23760	Rel-6.2: Terminating one service with the same name on any	This issue occurs because all services on all Launchpads have the same	If a user terminates any of the services on either of the network-services,	Use a unique NSR name if an organization is using multiple Launchpads.

Support Tickets	Title	Description	Impact	Workaround
	Launchpad interrupts existing services on all Launchpads.	UniqueID which is the InstanceID on ES2. When a user terminates a service, then terminates Instance on ES2 which causes issues for all existing services because they have the same UniqueID.	this will make any other services orphan.	Ensure that an OpenStack tenant is controlled by a single Launchpad. **Note**: There are no plans to change this behavior.
RIFT-24343	Cpu stays high for rwmain tasklet due to msgbroker.	In some instances when many packages are uploaded or instantiations are done, the main rwmain process starts using 100% CPU.	In this scenario, the CPU usage is high because the lib dispatch library starts subscribing for more threads to the DTS router queue and each thread keeps polling out of the event loop.	N/A
RIFT-24706	Adopted network-service fails after HA failover.	A user instantiated and NS with a vim-network-name provided for both the mgmt-network and ping-pong-vld. During instantiation, NS failed at the VL-init stage.	N/A	If user is going to configure the vim-network-name in VLDs using input variable xpaths, then the ip-profile-ref value in the NSD should not be set.
RIFT-24872	Updating ipv4 to ipv6 fails in the case of FIXED IP ADDRESS and vice versa.	A user modified the 'FIXED IP ADDRESS' field and then added an IPv4 address in the descriptor details pane for a VDU. Next the user updated the configuration to IPV6 and clicked the 'Update' button.	This issue only occurs when there are two IP addresses in interface configurations (IPv4 or IPv6) and the user tries to change one address to one with a different version.	Delete the existing address and add one new address with different a version.

Support Tickets	Title	Description	Impact	Workaround
		An error message appeared on the UI.		
RIFT-24992	Running AWS NS fails after failover to a new LP.	A user created an NS on Launchpad 1. The service is in the running state for tiny descriptors with an AWS VM account. The user set up an HA pairing with another Launchpad of a different version. Failover occurred to Launchpad 2; the NS was that running in the Launchpad 1 failed in instantiate in Launchpad 2.	N/A	Terminate the NS and re-instantiate in the second LP.
RIFT-25495	OpenId stats publisher overwrites old data on restart	After restating Launchpad, the stats corresponding to XPath rw-openidc-provider:openidc-provider-state/rw-openidc-provider:statistics are overwritten.	After restarting Launchpad, the openidc stats are reset rather than aggregated.	N/A
RIFT-25610	REL_7.1.1.0: AWS - NS stuck indefinitely in VNF-INIT phase when using a scaling-group and constituent VNF's start-by-default = False.	A user tried to instantiate a simple NS with one VNF, a single VDU (for an Ubuntu VM), and a single VLD. The NSD had a scaling group and the constituent VNFD's start-by-default value is False. When the NS is instantiated, it is stuck	No VNF is started by default. Since the min-instance-count for scaling group is 1, the VNF is supposed to be scaled out once NS instantiates.	N/A

Support Tickets	Title	Description	Impact	Workaround
		forever in the VNF-INIT phase. There are no errors seen in the Event Logs or in rift.log.		
RIFT-25902	Discovery gets triggered even on failed account due to non-reachable network status.	A user created a new VIM account to OpenStack. If the network is down and the cloud setup is not reachable, then when a user attempts discovery it fails but the process is still triggered.	Misleading status of VIM account on UI.	N/A
RIFT-26498	Creation of custom "User" in the VDUs via RIFT.ware isn't working.	In the NS instantiation page, Launchpad provides the option to create new users in the VDUs. This functionality isn’t working correctly.	A user is not able to add users to VMS from the Launchpad instantiation screen.	N/A
RIFT-26409	AWS alarm was failed to create after one of the instance (running) failure in AWS	RIFT supports AWS alarms (SNS notifications). These alarms help in monitoring the state of the VM (VDU). Upon failure of the instance in the AWS, an SNS notification appears.	After the instance failure from the AWS console, the NS should fail in Launchpad. This issue is not occurring correctly.	A user must depend on the monitoring params to check the state of VMs.
RIFT-26498	Creation of custom "User" in the VDUs via RIFT.ware isn't working.	In the NS instantiation page, Launchpad provides the option to create new users in the VDUs. This is not working correctly.	It is possible to add users to VMs from the Launchpad instantiation screen.	N/A
RIFT-26567	Placement groups: transaction failure	An operator loaded ping pong scaling	Duplicate traps are generated.	N/A

Support Tickets	Title	Description	Impact	Workaround
	prints 5 traps riftLPUserSessionEnd	descriptors and then deleted placement groups from the NSD an VNFDs. Next, the user created and instantiated the NS. An error message and 5 traps appear.		
RIFT-26409	AWS alarm was failed to create after one of the instance (running) failure in AWS	RIFT supports AWS alarms (SNS notifications). These alarms help in monitoring the state of the VM (VDU). Upon failure of the instance in the AWS, an SNS notification appears.	After the instance failure from the AWS console, the NS should fail in Launchpad. This issue is not occurring correctly.	A user must depend on the monitoring params to check the state of VMs.
RIFT-27186	Service instantiation fails with incorrect timeout error	RIFT.ware was not capturing the real reason why pods failed. Pods only retry in the case of failure, so it was not possible to say that pod creation was in a Failure state.	Instantiation fails in this scenario.	View the cluster event logs in Launchpad.
RIFT-27241	Instantiating ping-pong fails after ping VNF descriptor is modified and it is impossible to tell why	If a user choses the management interface type in a VNFD as VDU or Connection Point and the corresponding VDU/CP is not selected in the next input, then no error appears until the user attempts to instantiate. The configuration fails in	If a user chooses the mgmt-interface type but no ID is selected, then RIFT.ware considers both the type and the ID as empty.	If a user configures the NS properly, then an issue will not occur.

Support Tickets	Title	Description	Impact	Workaround
		the instantiation attempt.		
RIFT-27451	Instantiation variables not correctly applied to VNFD	An operator had an NSD with an instantiation variable associated with input-parameter-xpath entries that point to input-variables entries created on the ping vnfd. After instantiating the NS, the variables are not applied to the targets. If the operator changes the usage of the instantiation variables, then the results are different and unpredictable.	N/A	N/A
RIFT-27452	Datacenter instantiation variable not correctly applied	An operator had an NSD with an instantiation variable used to set the datacenter (xpath/datacenter). After instantiating the NS, the variable is not applied. The following message appears: "operational-status-details": "Datacenter None not present"	N/A	N/A
RIFT-28142	Create service progress did not complete but service is created. Cannot instantiate.	An operator attempted to instantiate a Network Service and the screen froze. However, the Services tab on the UI	This issue occurs when a Datacenter account is created before project creation is complete.	Create another project and wait for 30 seconds before creating a Datacenter account.

Support Tickets	Title	Description	Impact	Workaround
		lists the service as Created.	It might happen when using APIs, but it is not likely using the UI.	
RIFT-28449	Dependency issue in cloud accounts, config, operdata	If an operator has tens or hundreds of accounts/datacenters configured under a single project which are not being used by any NS, then it is necessary to delete all the cloud accounts prior to deleting that one project.	An error message appears if you do not delete all the cloud accounts prior to deleting the project	Delete all cloud accounts prior to deleting the project.
RIFT-28519	Config agent account broken	An operator went to the connector plugin in the UI and created a config agent account with random credentials. After 5 to 10 minutes, the account did not have a connection status or a message. Also, the field 'secret' from the UI is not present in the API response when a user queries config-agent. Lastly, the oper-state is missing.	RIFT is the only config agent that is working correctly. Other VNFDs and NSDs need to use RIFT as the configuration agent.	RIFT is the only config agent that is working correctly. Therefore, configuration scripts need to comply with RIFT.
RIFT-28683	LCM worker communication error with flavor 16/64/500	When lots of LCM transactions are performed concurrently on a high load, there is a chance that the Python	This will make the affected LCM workers unresponsive and no further LCM actions can be scheduled on those workers.	Restart the affected LCM worker after stopping the API/LCM transaction load. Wait for the LCM worker to recover.

Support Tickets	Title	Description	Impact	Workaround
		threads will get blocked. This is a libc (version 2.27) pthread library issue in the condition variable implementation. For more details, see https://sourceware.org/bugzilla/show_bug.cgi?id=25847.		
RIFT-28834	Overwrite existing package option is not available	An operator onboarded a package in the UI. An operator might want to onboard the same package again and overwrite the existing package. The option to overwrite the existing package is missing from the UI.	It is not possible to overwrite an existing package in the UI.	If a package is not in use, then it can be deleted from the catalog and then onboarded again. To onboard an updated version of a package, ensure that it has a new unique id. Note: When using Launchpad to copy a package, the new package is automatically given a new ID. The package with the new ID can be onboarded and it will co-exist with the current package.
RIFT-28883	NS Instantiation failed on using instantiation-variable.	While performing an audit discovery using instantiation variables, the NS adoption is failing and the following error message appears: Instantiating vnfr failed: VM Orig1-NxdnTWHwZ-iovdu0	In release 8.1.0.0, instantiation-variable-support is not working properly while trying to provide input variables in an attached csv file.	Use input-parameter-xpath to modify data during NS instantiation.

Support Tickets	Title	Description	Impact	Workaround
		is not found for discovery.		