

RIFT.ware™ version 7.2.2.111877

Release Notes
March 2020

Notice

The information and descriptions contained herein embody confidential and proprietary information that is the property of RIFT, Inc. Such information and descriptions may not be copied, reproduced, disclosed to others, published or used, in whole or in part, for any purpose other than that for which it is being made available without the express prior written permission of RIFT, Inc.

Nothing contained herein shall be considered a commitment by RIFT to develop or deliver such functionality at any time. RIFT reserves the right to change, modify, or delete any item(s) at any time for any reason.

Table of Contents

Notice.....2

RIFT.ware 7.2.2 Release Notes4

New and Changed Features for 7.2.25

ETSI GS NFV-SOL 002 Support 6

 ETSI GS NFV-SOL 002 APIs 6

 VNF Lifecycle Management Interface APIs..... 6

 ETSI GS NFV-SOL 002 API Examples 7

Support for Kubernetes as a VIM..... 18

 RIFT.ware support for Container Orchestration (CNFD) 18

 UI Enhancements 18

 API Changes..... 23

 Model Changes..... 23

Fixed Issues in RIFT.ware 7.2.226

Known Issues in RIFT.ware 7.2.2.....35

RIFT.ware 7.2.2 Release Notes

This guide describes the RIFT.ware 7.2.2 release, including new features, fixed and known issues, with their workarounds.

New and Changed Features for 7.2.2

RIFT.ware version 7.2.2 introduces enhancements to improve management.

Feature	PFR and JIRAs
ETSI GS NFV-SOL 002 Support	PFR-636 RIFT-26732
Support for Kubernetes as a VIM	PFR-553 RIFT-25035

ETSI GS NFV-SOL 002 Support

RIFT.ware now supports VNF Lifecycle Management Interface of SOL002 APIs defined by the ETSI specs SOL002.

- [ETSI GS NFV-SOL 002 APIs](#)
- [ETSI GS NFV-SOL 002 API Examples](#)

RIFT.ware supports the following ETSI GS NFV-SOL 002 APIs. Refer to the [ETSI GS NFV-SOL 002 V2.6.1 \(2019-04\)](#) documentation for additional information on these APIs.

Note: There are no deviations in any of these APIs for RIFT.ware besides the notes indicated below.

ETSI GS NFV-SOL 002 APIs

VNF Lifecycle Management Interface APIs

/sol002/default/vnflcm/v1/api_versions

GET: Gets version information of the sol002 implementation.

/sol002/default/vnflcm/v1/vnf_instances

GET: Get VNFs available in the system. Filter support available ([See examples](#)).

POST: Currently, creating a VNF via sol002 is not available.

/sol002/default/vnflcm/v1/vnf_instances/<vnfInstanceId>/heal

POST: Heal the VNF. This can be used to restart the VM or to run a VNF config-script ([See examples](#)).

Note: vnfInstanceId is the same as vnfr-id.

/sol002/default/vnflcm/v1/vnf_lcm_op_occs

GET: Get Information about all executed LCM operations.

/sol002/default/vnflcm/v1/vnf_lcm_op_occs/<vnfLcmOpOccsId>

GET: Get Information about a specific LCM operation executed before.

Note: vnfLcmOpOccsId can be obtained from the response header of LCM request or by running /vnflcm/v1/vnf_lcm_op_occs to get all lcm occurrence information.

/sol002/default/vnflcm/v1/vnf_lcm_op_occs/<vnfLcmOpOccsId>/retry

POST: Retry a failed LCM. Currently, only HEAL LCM is supported.

VNF Lifecycle Management Interface APIs

Note: vnfLcmOpOccsId can be obtained from the response header of LCM request or by running /vnflcm/v1/vnf_lcm_op_occs to get all lcm occurrence information.

/sol002/default/vnflcm/v1/subscriptions

GET: Get Information about all active subscriptions.

POST: Creates a new subscription.

/sol002/default/vnflcm/v1/subscriptions/<subscriptionId>

GET: Get Information for a specific subscription.

DEL: Delete a subscription.

Note: See the [ETSI GS NFV-SOL 002 V2.6.1 \(2019-04\)](#) for more information about these APIs.

ETSI GS NFV-SOL 002 API Examples

The following API examples are used for configuring operations using SOL 002. It is assumed that RIFT.ware Launchpad is running with a default project.

Note: The headers follow the following format unless called out below.

```
headers = {
  'Version': 'v1',
  'Authorization': 'Basic YWRtaW46YWRtaW4='
}
```

Body is *None* unless specified.

GET

https://10.64.219.4:8008/sol002/default/vnflcm/v1/vnf_instances

```
[
  {
    "id": "a2d1668b-b6d0-441d-804d-bb332ee5f6fa",
    "instantiationState": "INSTANTIATED",
    "vnfInstanceDescription": "Toy Rift.ware VNF ",
    "vnfInstanceName": "default__pp_service__ping_vnfd__1",
    "vnfProductName": "ping_vnfd",
    "vnfProvider": "RIFT",
    "vnfSoftwareVersion": "1.0",
```

```

    "vnfdId": "02f25998-5164-11e9-b230-02420a6e0500",
    "vnfdVersion": "1.0"
  },
  {
    "id": "7abde384-acaf-4d5e-8405-081d0d408f4b",
    "instantiationState": "INSTANTIATED",
    "vnfInstanceDescription": "Toy Rift.ware VNF ",
    "vnfInstanceName": "default__pp_service__pong_vnfd__2",
    "vnfProductName": "pong_vnfd",
    "vnfProvider": "RIFT",
    "vnfSoftwareVersion": "1.0",
    "vnfdId": "02f301fe-5164-11e9-b230-02420a6e0500",
    "vnfdVersion": "1.0"
  }
]

```

GET

https://10.64.219.4:8008/sol002/default/vnflcm/v1/vnf_instances?all_fields

```

[
  {
    "id": "a2d1668b-b6d0-441d-804d-bb332ee5f6fa",
    "instantiatedVnfInfo": {
      "extCpInfo": [
        {
          "associatedVnfcCpId": "54fef25c-32b7-11ea-88bc-02420a40db04",
          "cpdId": "ping_vnfd/cp0",
          "extLinkPortId": "c3afec5c-2c51-45ec-af21-cfa6018862a5",
          "id": "4799d919-4a16-4e38-8829-6183cb4989ee"
        },
        {
          "associatedVnfcCpId": "54fef25c-32b7-11ea-88bc-02420a40db04",
          "cpdId": "ping_vnfd/cp1",
          "extLinkPortId": "20dfd9c8-fbf3-401d-b866-d28c6ec14b98",
          "id": "f5651791-a229-458a-9e51-965dd388021b"
        }
      ],
      "extVirtualLinkInfo": [
        {
          "id": "c3afec5c-2c51-45ec-af21-cfa6018862a5",
          "linkPorts": [
            {
              "id":
"default__pp_service__ping_vnfd__1__ping_vnfd/cp0__ens3",
              "resourceHandle": {
                "resourceId": "vimAcc",
                "vimConnectionId": "vimAcc"
              }
            }
          ],
          "resourceHandle": {
            "resourceId": "c3afec5c-2c51-45ec-af21-cfa6018862a5",
            "vimConnectionId": "vimAcc"
          }
        }
      ]
    }
  }
]

```

```

    },
    {
      "id": "20dfd9c8-fbf3-401d-b866-d28c6ec14b98",
      "linkPorts": [
        {
          "id":
"default__pp_service_ping_vnfd__1_ping_vnfd/cpl__ens4",
          "resourceHandle": {
            "resourceId": "vimAcc",
            "vimConnectionId": "vimAcc"
          }
        }
      ],
      "resourceHandle": {
        "resourceId": "20dfd9c8-fbf3-401d-b866-d28c6ec14b98",
        "vimConnectionId": "vimAcc"
      }
    }
  ],
  "flavourId": "default",
  "localizationLanguage": "US",
  "monitoringParameters": [
    {
      "id": "1",
      "name": "ping-request-tx-count",
      "performanceMetric": "PacketIncomingVnfExtCP"
    },
    {
      "id": "2",
      "name": "ping-response-rx-count",
      "performanceMetric": "PacketOutgoingVnfExtCP"
    }
  ],
  "vnfState": "STARTED",
  "vnfcResourceInfo": [
    {
      "computeResource": {
        "resourceId": "b40294fd-39a0-48ca-b60b-8186ad740d21",
        "vimConnectionId": "vimAcc"
      },
      "id": "54fef25c-32b7-11ea-88bc-02420a40db04",
      "vduId": "iovdu_0",
      "vnfcCpInfo": [
        {
          "cpProtocolInfo": [
            {
              "ipOverEthernet": {
                "ipAddresses": [
                  {
                    "addresses": [
                      "10.5.2.141"
                    ],
                    "subnetId": "3e793e27-2114-4cf4-9327-6b9393f2b970",
                    "type": "IPv4"
                  }
                ]
              }
            ]
          ]
        }
      ]
    }
  ],

```

```

        "macAddress": "fa:16:3e:e4:d8:af"
      },
      "layerProtocol": "IP_OVER_ETHERNET"
    }
  ],
  "cpdId": "ping_vnfd/cp0",
  "id": "4799d919-4a16-4e38-8829-6183cb4989ee",
  "vnfExtCpId": "ping_vnfd/cp0"
},
{
  "cpProtocolInfo": [
    {
      "ipOverEthernet": {
        "ipAddresses": [
          {
            "addresses": [
              "31.31.31.10"
            ],
            "subnetId": "69d007c2-7775-47eb-8dea-
749c288c0ea6",
            "type": "IPv4"
          }
        ],
        "macAddress": "fa:16:3e:6a:0b:0a"
      },
      "layerProtocol": "IP_OVER_ETHERNET"
    }
  ],
  "cpdId": "ping_vnfd/cp1",
  "id": "f5651791-a229-458a-9e51-965dd388021b",
  "vnfExtCpId": "ping_vnfd/cp1"
}
]
},
{
  "computeResource": {
    "resourceId": "b40294fd-39a0-48ca-b60b-8186ad740d21",
    "vimConnectionId": "vimAcc"
  },
  "id": "54fef25c-32b7-11ea-88bc-02420a40db04",
  "vduId": "iovd0",
  "vnfcCpInfo": [
    {
      "cpProtocolInfo": [
        {
          "ipOverEthernet": {
            "ipAddresses": [
              {
                "addresses": [
                  "10.5.2.141"
                ],
                "subnetId": "3e793e27-2114-4cf4-9327-
6b9393f2b970",
                "type": "IPv4"
              }
            ],
            "macAddress": "fa:16:3e:e4:d8:af"
          }
        ]
      ]
    }
  ]
}
]
}

```

```

        },
        "layerProtocol": "IP_OVER_ETHERNET"
    }
],
"cpdId": "ping_vnfd/cp0",
"id": "4799d919-4a16-4e38-8829-6183cb4989ee",
"vnfExtCpId": "ping_vnfd/cp0"
},
{
    "cpProtocolInfo": [
        {
            "ipOverEthernet": {
                "ipAddresses": [
                    {
                        "addresses": [
                            "31.31.31.10"
                        ],
                        "subnetId": "69d007c2-7775-47eb-8dea-
749c288c0ea6",
                        "type": "IPv4"
                    }
                ],
                "macAddress": "fa:16:3e:6a:0b:0a"
            },
            "layerProtocol": "IP_OVER_ETHERNET"
        }
    ],
    "cpdId": "ping_vnfd/cp1",
    "id": "f5651791-a229-458a-9e51-965dd388021b",
    "vnfExtCpId": "ping_vnfd/cp1"
}
]
}
],
},
"instantiationState": "INSTANTIATED",
"vnfInstanceDescription": "Toy Rift.ware VNF ",
"vnfInstanceName": "default__pp_service__ping_vnfd__1",
"vnfProductName": "ping_vnfd",
"vnfProvider": "RIFT",
"vnfSoftwareVersion": "1.0",
"vnfdId": "02f25998-5164-11e9-b230-02420a6e0500",
"vnfdVersion": "1.0"
},
{
    "id": "7abde384-acaf-4d5e-8405-081d0d408f4b",
    "instantiatedVnfInfo": {
        "extCpInfo": [
            {
                "associatedVnfcCpId": "5529547a-32b7-11ea-88bc-02420a40db04",
                "cpdId": "pong_vnfd/cp0",
                "extLinkPortId": "c3afec5c-2c51-45ec-af21-cfa6018862a5",
                "id": "fd380e02-6161-4e66-9857-5d8150ee9c27"
            },
            {
                "associatedVnfcCpId": "5529547a-32b7-11ea-88bc-02420a40db04",
                "cpdId": "pong_vnfd/cp1",
            }
        ]
    }
}

```

```

        "extLinkPortId": "20dfd9c8-fbf3-401d-b866-d28c6ec14b98",
        "id": "14947fd2-cfa7-40be-a270-e7cb956f8129"
    }
],
"extVirtualLinkInfo": [
    {
        "id": "c3afec5c-2c51-45ec-af21-cfa6018862a5",
        "linkPorts": [
            {
                "id":
"default__pp_service__pong_vnfd__2__pong_vnfd/cp0__ens3",
                "resourceHandle": {
                    "resourceId": "vimAcc",
                    "vimConnectionId": "vimAcc"
                }
            }
        ],
        "resourceHandle": {
            "resourceId": "c3afec5c-2c51-45ec-af21-cfa6018862a5",
            "vimConnectionId": "vimAcc"
        }
    },
    {
        "id": "20dfd9c8-fbf3-401d-b866-d28c6ec14b98",
        "linkPorts": [
            {
                "id":
"default__pp_service__pong_vnfd__2__pong_vnfd/cp1__ens4",
                "resourceHandle": {
                    "resourceId": "vimAcc",
                    "vimConnectionId": "vimAcc"
                }
            }
        ],
        "resourceHandle": {
            "resourceId": "20dfd9c8-fbf3-401d-b866-d28c6ec14b98",
            "vimConnectionId": "vimAcc"
        }
    }
],
"flavourId": "default",
"localizationLanguage": "US",
"monitoringParameters": [
    {
        "id": "1",
        "name": "ping-request-rx-count",
        "performanceMetric": "PacketIncomingVnfExtCP"
    },
    {
        "id": "2",
        "name": "ping-response-tx-count",
        "performanceMetric": "PacketOutgoingVnfExtCP"
    },
    {
        "id": "3",
        "name": "packet-rate",
        "performanceMetric": "PacketOutgoingVnfExtCP"
    }
]

```

```

    }
  ],
  "vnfState": "STARTED",
  "vnfcResourceInfo": [
    {
      "computeResource": {
        "resourceId": "c2c93fa0-a3fb-4494-863c-d01bf5bf7cad",
        "vimConnectionId": "vimAcc"
      },
      "id": "5529547a-32b7-11ea-88bc-02420a40db04",
      "vduId": "iovd_u_0",
      "vnfcCpInfo": [
        {
          "cpProtocolInfo": [
            {
              "ipOverEthernet": {
                "ipAddresses": [
                  {
                    "addresses": [
                      "10.5.2.125"
                    ],
                    "subnetId": "3e793e27-2114-4cf4-9327-6b9393f2b970",
                    "type": "IPV4"
                  }
                ],
                "macAddress": "fa:16:3e:4e:f0:a1"
              },
              "layerProtocol": "IP_OVER_ETHERNET"
            }
          ],
          "cpdId": "pong_vnfd/cp0",
          "id": "fd380e02-6161-4e66-9857-5d8150ee9c27",
          "vnfExtCpId": "pong_vnfd/cp0"
        },
        {
          "cpProtocolInfo": [
            {
              "ipOverEthernet": {
                "ipAddresses": [
                  {
                    "addresses": [
                      "31.31.31.1"
                    ],
                    "subnetId": "69d007c2-7775-47eb-8dea-749c288c0ea6",
                    "type": "IPV4"
                  }
                ],
                "macAddress": "fa:16:3e:0d:e1:bc"
              },
              "layerProtocol": "IP_OVER_ETHERNET"
            }
          ],
          "cpdId": "pong_vnfd/cp1",
          "id": "14947fd2-cfa7-40be-a270-e7cb956f8129",
          "vnfExtCpId": "pong_vnfd/cp1"
        }
      ]
    }
  ]
}

```

```

    }
  ]
},
{
  "computeResource": {
    "resourceId": "c2c93fa0-a3fb-4494-863c-d01bf5bf7cad",
    "vimConnectionId": "vimAcc"
  },
  "id": "5529547a-32b7-11ea-88bc-02420a40db04",
  "vduId": "iovdu_0",
  "vnfcCpInfo": [
    {
      "cpProtocolInfo": [
        {
          "ipOverEthernet": {
            "ipAddresses": [
              {
                "addresses": [
                  "10.5.2.125"
                ],
                "subnetId": "3e793e27-2114-4cf4-9327-6b9393f2b970",
                "type": "IPV4"
              }
            ],
            "macAddress": "fa:16:3e:4e:f0:a1"
          },
          "layerProtocol": "IP_OVER_ETHERNET"
        }
      ],
      "cpdId": "pong_vnfd/cp0",
      "id": "fd380e02-6161-4e66-9857-5d8150ee9c27",
      "vnfExtCpId": "pong_vnfd/cp0"
    },
    {
      "cpProtocolInfo": [
        {
          "ipOverEthernet": {
            "ipAddresses": [
              {
                "addresses": [
                  "31.31.31.1"
                ],
                "subnetId": "69d007c2-7775-47eb-8dea-749c288c0ea6",
                "type": "IPV4"
              }
            ],
            "macAddress": "fa:16:3e:0d:e1:bc"
          },
          "layerProtocol": "IP_OVER_ETHERNET"
        }
      ],
      "cpdId": "pong_vnfd/cp1",
      "id": "14947fd2-cfa7-40be-a270-e7cb956f8129",
      "vnfExtCpId": "pong_vnfd/cp1"
    }
  ]
}

```

```

    ]
  }
]
},
"instantiationState": "INSTANTIATED",
"vnfInstanceDescription": "Toy Rift.ware VNF ",
"vnfInstanceName": "default__pp_service__pong_vnfd__2",
"vnfProductName": "pong_vnfd",
"vnfProvider": "RIFT",
"vnfSoftwareVersion": "1.0",
"vnfdId": "02f301fe-5164-11e9-b230-02420a6e0500",
"vnfdVersion": "1.0"
}
]

```

GET

[https://10.64.219.4:8008/sol002/default/vnflcm/v1/vnf_instances?filter=\(eq,vnfdId,02f25998-5164-11e9-b230-02420a6e0500\);\(eq,vnfProvider,RIFT\)](https://10.64.219.4:8008/sol002/default/vnflcm/v1/vnf_instances?filter=(eq,vnfdId,02f25998-5164-11e9-b230-02420a6e0500);(eq,vnfProvider,RIFT))

```

[
  {
    "id": "a2d1668b-b6d0-441d-804d-bb332ee5f6fa",
    "instantiationState": "INSTANTIATED",
    "vnfInstanceDescription": "Toy Rift.ware VNF ",
    "vnfInstanceName": "default__pp_service__ping_vnfd__1",
    "vnfProductName": "ping_vnfd",
    "vnfProvider": "RIFT",
    "vnfSoftwareVersion": "1.0",
    "vnfdId": "02f25998-5164-11e9-b230-02420a6e0500",
    "vnfdVersion": "1.0"
  }
]

```

POST

https://10.64.219.4:8008/sol002/default/vnflcm/v1/vnf_instances/a2d1668b-b6d0-441d-804d-bb332ee5f6fa/heal

Header

```

headers = {
  'Version': 'v1',
  'Content-Type': 'application/json',
  'Authorization': 'Basic YWRtaW46YWRtaW4='
}

```

Body

```

{
  "vnfcInstanceId": [
    "3d82a5ac-305e-11ea-a3c9-02420a40db04"
  ]
}

```

```
    ]
  }
```

Response

None, 202 ACCEPTED

GET

<https://10.64.219.4:8008/sol002/default/vnflcm/v1/vnf lcm op occs>

```
[
  {
    "id": "e3a309b7-f1d7-496c-a46e-174fbf4464df",
    "isAutomaticInvocation": "false",
    "isCancelPending": "false",
    "operation": "HEAL",
    "operationParams": {
      "healScript": "random_cprimitive.py",
      "vnfcInstanceId": [
        "3d82a5ac-305e-11ea-a3c9-02420a40db04"
      ]
    },
    "operationState": "FAILED_TEMP",
    "startTime": "08:24:23.755241",
    "stateEnteredTime": "08:26:03.767354",
    "vnfInstanceId": "a2d1668b-b6d0-441d-804d-bb332ee5f6fa"
  },
  {
    "id": "9ea484b2-b9aa-4b07-8a97-f858bcdbfd85",
    "isAutomaticInvocation": "false",
    "isCancelPending": "false",
    "operation": "HEAL",
    "operationParams": {
      "vnfcInstanceId": [
        "3d82a5ac-305e-11ea-a3c9-02420a40db04"
      ]
    },
    "operationState": "COMPLETED",
    "startTime": "08:24:38.182950",
    "stateEnteredTime": "08:26:03.767411",
    "vnfInstanceId": "a2d1668b-b6d0-441d-804d-bb332ee5f6fa"
  }
]
```

POST

<https://10.64.219.4:8008/sol002/default/vnflcm/v1/vnf lcm op occs/e3a309b7-f1d7-496c-a46e-174fbf4464df/retry>

None, 202 ACCEPTED

GET

<https://10.64.219.4:8008/sol002/default/vnflcm/v1/subscriptions>

```
[
  {
    "_links": {
      "self": {
        "href":
"https://10.64.219.4:8008/sol002/default/vnflcm/v1/subscriptions/f29297
31-6246-422a-bb5f-5559bdadb8e1"
      }
    },
    "callbackUri": "http://10.64.219.4",
    "id": "f2929731-6246-422a-bb5f-5559bdadb8e1"
  }
]
```

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 \(CNFD\)](#)
- [UI Changes](#)
 - [CaaS Connector Account](#)
 - [CNFD Descriptors](#)
- [API Changes](#)
- [Model Changes](#)
 - [CNFD Model](#)
 - [CNFR Model](#)
 - [MANO Enhancements](#)

RIFT.ware support for Container Orchestration (CNFD)

RIFT.ware uses a network service to instantiate 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
F3SUJBZ0lCQURBTklna3Foa2lHOXcwQkFRc0ZBREFTVJNd0VRWUR
WUVFERXdwcmlRXXSmwKY201bGRHVnpNQjRYFRjJd01ESXhNREUyT
VRRd05Wb1hEVE13TURJd056RTJNVFF3TlZvd0ZURVRNqkVHQTFR
QpBeE1LYTNWVpYSnVaWFJsY3pDQ0FTSXdeUVlKS29aSWh2Y05BU
UVCQlFBRGdnRVBIBRENDQVFvQ2dnRUJBT3lxClNjOHlZVm1RL01Ua
043UEJ3Y3FBMmVFQjF6T2UveVpuSExaSjlsVWhtdmN3YWtrOHNmY05
NelhqQVpZOGVsMGcKWUdRVThBRXRubHhEbTduOGZlQlRBTmVtbGs
ybEplTW1HQVlGVGxrNkxsWldjZ0tQcS9sMWNweFNQaDkzc1VscApXNn
BtTXgvQlM0aTFYOE1XV2NQY2h1c0MwSVE0MlpOYk9kMko4cDRsRk9
QZnZpd1JlZG5OUjZkZGY1eXVJQjcwCnRhekdyYTRSOUG5djdrSzQ0Y2h
vaFRUck4vQVg4WXIrWEtVRS81dFFPNk9OSk1TUmf3MWI1WERWK21
5SXgrS0IKaEJzdk5uZ2dTL2UwbXd5Ny9ETzNVcFNkZlBTGRKYzRucml
TcmtlM2kvU1hNVy9jTVdmV2VXNW1ZbFlhMGRsTQp6QzJvKzEwcTRiQ
21PUy91ZEZzQ0F3RUFBYU1qTUNFd0RnWURWUjBQVVFIL0JBUURB
Z0trTUE4R0ExVWRFd0VCCi93UUZNQU1CQWY4d0RRWUpLb1pJaHZj
TkFRRUxCUUFEZ2dFQkFERVBVc0tweWRlOHNjb0hyWjdldi9SRmVnQ
mwKZVl0NVprYkV2cDJsV2dWaUdCbUsxYmEycnhFbko3S1pxQU1kUE4
ybJ2Y1Q1dTBSRTRYSTNjUGNobE9SYVJrWQpPbVhYdTmZRTczYzlw
VXdkKzZDTXQ0NzVPSVlSRHR0bXFPbSVYzZzZkVMEZr2crMlZFdENZd
Vp5VHFrNGtOa1NpCnJLcGJYWEYzcmE1cVExWlNuVEZCb2VrbXdmeG
NvWGtXVmVzQjNTRjFoZmlFNGLCbGVsTU4rdFAwekVuYmxydkEKZX
A0N3IPZWZkSnNpNQnB4L2s3OGZiNHJVZGxXbktTWXdJdERTVWpxNF
VxTEJtTmJmaHl3MXVhVXdJb0ZsL2YyZQpIa0lGM2FaR3p3YThqVTV2T
FNMUTNlcytiUFJXSFIjNTB4bGg5MjFWSElNQUpaEpZNmVwK1F0TEx
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
F3SUJBZ0lJSUk2dEo1THk5YXN3RFFZSkvWklodmNOQVFFTEJRQXdG
```

VEVUTUJFR0ExVUUKQXhNS2EzVmlaWEp1WlhSbGN6QWVGdzB5TU
RBeU1UQXhOakUwTURWYUZ3MHINVEF5TURreE5qRTBNRFphTURR
eApGekFWQmdOVkJBb1REbk41YzNSbGJUcHRZWE4wWlhKek1Sa3dGd
1lEVlFRREV4QnJkV0psY201bGRHVnpMV0ZrCmJXbHVNSUICSWpBTk
Jna3Foa2lHOXcwQkFRRUZBQU9DQVE4QU1JSUJDZ0tDQVFFQXlCRUc
2Vk8wOWd6ZnVBL1UKeWpYUXU4aE5sRTdFL0cyK2F5TXg0WmtpYUJ
oU3NlRkVqWVMrU0ljZkFqaG1FbWVVRTlFUyml5emFDcmNXWTFWgp
kWndkZGRtcWpSTnJpSzdmYlduWXJQRlZPRmc4aWlBN3ZzOWZEKzRL
UGNyMlNBURkUE9tN3NxSDBYUi90aGJ3ClU1T0xaVmx1bnNaamRtZ3Z
jdkxobGRWeUdlQjZObE54UWI0NTB2VTEyaU1vWi9UdURGRUs4U1A3Y
nVUakF1VUYKMkRGYzhVbUZRRWlWZzh4cVl6N2E4bWhtQzFZZ2JEa2
JoMmRjcnBIMzhTRk9obU5yRFQvbmVYNnFST2Z5VjdkQQp1NVZGdVb3
OVZuQW5vQTVrQXdwZjdnWGNJVXhEbHhhalY1MXNZMWUzT3dqWU
MvOU5kQzk1NWZ0QS9zL2JSZG1XCjd4cElxd0lEQVFBQm95Y3dKVEFP
QmdOVkhROEJBZjhFQkFNQ0JhQXdFd1lEVlIwbEJBd3dDZ1lJS3dZQkJR
VUgKQXdJd0RRWUpLb1pJaHZjTkFRRUxCUUFEZ2dFQkFGWngvRWtu
Mm1kZERNM2xtR1BDL0tvcGp3UWN5RGRSK0JrQQpaZU9wamFWNkdz
VUVPUEVJcjZsU3d6U0JGK29BK9y9TT0d5SFJGMGlyR3EzM1J5d1JCUDd
LNTZqd3pEdFhPQzRkCnQwZUtBM2lDSW4vT2FMSithNERnRGZ1RExD
QTZsNUl0MzFSOWFITk9QVi9LumpYSGNLc2dXSjdoNXQ3YzdrUkgKS
EpRaytaWFl1NTZlclXYyU09WQ0I3NUFMV1VsOFIEcHAxWnFvOG9lZH
VLcnkxRmgvNlJOWjM2a1hFSGg1V1hoeQpod3VFaFZ1c1d5d2dTR25mMj
dmSVJKTM9Oc2lMdFJHQ1cwWDdLMk5sc0tlY1lld1JuLzl0VWU3Q1d3Zk
prNjR2CkFmWVpPbkFXcVlsRU52c2Q4WjRrbE1XUWptUXdJaHRQQTVc
UVFrY1E4TnlRQS8wTWVaTT0KLS0tLS1FTkQgQ0VSVElGSUNBVEUtl
S0tLQo= client-key-data:

LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0tLQpNSU1Fb3dJQkFB
S0NBUEVBeUJFRzZWTzA5Z3pmdUEvVXlqWFF1OGhObEU3RS9HMith
eU14NFpraWFCaFNzZUZFCmpZUytTSWNmQWpobUVtZVFOUVRiYjl6
YUNyY1dZMVNaZFp3ZGRkbXFqUk5yaUs3ZmJXbllYUEZWT0ZnOGlpQ
TcKdnM5ZkQrNEtQY3IyU0FSRGNQT203c3FIMFhSL3RoYndVNU9MWl
ZsdW5zWmpkbWd2Y3ZMaGxkVnlHSEI2TmxOeApRYjQ1MHZVMTJpT
W9aL1R1REZFSzhSUDdidVRqQXVVRjJERmM4VW1GUUVpVmc4eHFZ
ejdhOG1obUMxWWdiRGtiCmgyZGNycEgzOFNGT2htTnJEVC9uZVg2cVJ
PZnlWN2RBdTVWRnVZNzlwbkFub0E1a0F3cGY3Z1hjSVV4RGx4YWoK
VjUxc1kxZTNpd2pZQy85TmRDOTU1ZnRBL3MvYlJkbVc3eHBjCxdJREF
RQUJBb0lCQUQzbnk5OjSS9Pa0RmbnVGZQoxS0owb0U5YUhoY0V3R0t5d
WphQVAXRGtod2JhYjh2bVBjWGTdDFl2S3BnUFIxWHVnV3BHUKhlS1N
GVUViWUx5CmpFRWR1dWZhSDdGakR1VE41d2Y5dGpOY3dOZWNSY
ThmWmgwV0lQM1cwNGhuSW1rS0dLdjhWK3lPOGHOWERxVVMKMSs
ya08zOXllbzNhS0xCY2NFSUs4TTNocTFudjlxN2VkN1p2aEFkc1lBS1FNe
UZyU2ZzSWZUM0liUzcrN200Vwo0WjF6d3diLzJmWjQ2WDJCQllNbmRx
UFpFeElhT0o1YkdvdWl3d1pyZWlYUINCslk1UVNLtGI4clNiRW9pOUdV
CjhLbzZmd3BMZVM1K2VCMTRKam5BS3MxbG1PdW1UUVoxZ05RSXd
qT3lrVGMyMnN1dWZDUGJMqmpsRnE3RHJHYnkKS9WRmtRRUNnW
UVBKzA5Mk9udlY5TDloUjdualhNHpZrk1wMzU0S1MzWjBLCGsvQVJV

NHVCa3hZWUNUb3JGdgpGNHkyRjI2TXRQTzIPIU3F1MDdlbHh4SGF1N
 UFhU0NHVGldlJleFFsa2ZTQW9CUFNHTFhYZHhwc05ZakNYSWErCitJ
 OG9aaTFiQ054SXhtY2dYdXNNemp0UEp6d2RjQ2g4UHB4dlVMTS9OT08
 0UGZpeU1GYTg0V3NDZ1lFQXk4ekUKYi9OaisvZjV2ZEFPTjEzTE5ORT
 hFbXpscFJIM05EaVpLNnFKdm56TzFGewVWXZUIRSDl1bHdKSFRFZnV1
 UjdttdAp1QUpKeTIyZXRqMWIESldocUJ5MllxQXpsdHRMcVV3S2Z4M3F
 WSlZCeXZ3M25zTW9QTEZWTZPZ04zZWZYamp4CnA5eFNod1hZRXJ
 obDZmMkFsZmlvcDVnUnVQemowRjlUZjluM3hjRUNnWUFTYzhjdWpD
 RRRrMC9GUUhUZG16alMKZmNpNVRwVmE3UnVpZFJZWE1rT3o0TVd
 yYnhGb0JMY3ZkM2wzUnZxMTNwK3FMdVFmVDRDd21UTUNTcFpqYl
 hPegp6NXdWK1dpNHlzY1crZDJYU3VMRE1BRjUvTXlrbG93M0crdlBkT
 TBXWFhaS2V3LzVhRERNZzdaUUh5M2FLbDdWCKnN0XIVWGIybnpIT
 Ct4SszZVZVFVZVFLQmdFR2lGbmsvVnlua2VJZUtvNmx0Q1c1ZkhBdUxD
 b0lZd1JZT1REWGwKM0NwK28xVjg1bnBuOVcxdehYcGcvQ3JFZFNJZ1
 NEUUVIS09ORUJuOHRzRDZ4MjhPb0lXalZObjJrZTFaMGQRUwpVTThnc
 W1tbXhIcmF1dkNnNDdqSHYyZGRuUE9KUHpvaUdHbm5sZWloZmlEWF
 JUd0hNcm1XbmM2SGt3NmVSSlBlCkxJK0JBb0dCQVBCL05qaGVJK1RE
 WVFLbmxbp1J1dmlQWNLTs9id1d5Y3NHQ3J2Nnl2N2p6Y2phNnZ5REJk
 dFYKM0c0cGtpUFdDdGNxUm9xZ0tIY3FON1FoWkdaS1hrSjJldy9zYWVC
 NzFuYzBFQnJod0kwb1NSOXdvVEphVXhkNgpzajR4MVdvVUPLVDIVTlJ
 0NzREaXY4aEhtaGNCOFVKV3dGMU1RdHNlNUIKRlEzSGNKY3p5Ci0t
 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**.

ACCOUNT

* required

NAME *

Sample

SELECT ACCOUNT TYPE



Kubernetes



Mock

Account Details

TILLER IP *

10.66.4.35

TILLER PORT *

31218

KUBE CONFIG *

apiVersion: v1
clusters:
- cluster:

CNF INSTANCE TIMEOUT (DEFAULT 300 SECONDS)

300

Note: In order to instantiate a sample NS using Kubernetes as a VIM, it is necessary to upload CNFD ping/pong 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

CNFD CATALOG

☐

NAME	VERSION	VENDOR	LOGO
No rows found			

Showing 0 of 0 entries

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-cnfd. 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: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 (Internal Model): 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 7.2.2

Support Tickets	Title	Description
RIFT-26002	Sol005 ns-lcm-op-occs details not recovered after LP reboot.	A user instantiated an NS using SOL005 APIs. The user verified that ns_lcm_op_occs API had a proper response. The user performed a restart/reboot of Launchpad and called the same API again. The reboot of Launchpad did not handle the op occurrence details and the response is empty. This scenario also occurred during HA failover. This issue is resolved.
RIFT-26481	Launchpad blanked upon the presence of a wrong RO config.	A user configured the wrong resource orchestrator, and this caused the Launchpad to be unusable. The HOME/Dashboard page of the Launchpad crashed if a RO with missing mandatory configuration was present. It was only possible to create a RO with the missing mandatory configuration from the CLI. It was not possible from the UI. This issue is resolved.
RIFT-26496	Injection of SSH public key via Rift.ware isn't working.	RIFT.ware has the capability of injecting public SSH keys into the VDUs. There are two different mechanisms to add SSH keys. An operator can upload an SSH key to a Project (Administration -> SSH Keys). On the NS Instantiation page, you can refer to this SSH key. This SSH key should be injected into the VDU. A user can also define an SSH key in an NSD and this key-pair should be injected into the VDU accordingly. The second mechanism was not working correctly. This issue is resolved.
RIFT-26499	Disable take action button for terminate failed NS restart scenario.	A user created a VIM account, onboarded scaling descriptors and instantiated NS. Next, the operator edited /etc/resolv.conf with the wrong data. In this scenario, the

Support Tickets	Title	Description
		user tried to terminate the NS, but it failed. This issue is closed.
RIFT-26503	External cp creation disabled option makes NS fail.	A user created a VNFM account where ext cp creation is set to No and instantiation failed. It was not possible to successfully instantiate for a VNFM account with external cp creation disabled (set to no). This issue is resolved.
RIFT-26506	Sol003: For failed-init NS terminate has no effect; calls rollback; delete is not clean.	When a user deleted a failed SOL003 NS, it was successful. When the user deleted the corresponding datacenter account, the account deletion failed. This issue only occurred for services which use a VNFM account with an external cp disabled (set to no). This issue is resolved.
RIFT-26519	Placement groups implementation do not include NS level.	The support of placement groups is at the VNFD level. It was previously possible to configure it but there was no way to add it at the NSD level during instantiation. This issue is resolved.
RIFT-26532	Network Services are running failed after the Stand-alone upgrade of the LP (7.1 to 7.2).	An NS failed after upgrading the Launchpad. This issue is resolved.
RIFT-26537	Datacenter edit option always populates field values from SDN account.	A user created an SDN OpenStack account and a datacenter account associated with an SDN account. The fields were auto-populated from the SDN account. The user edited the datacenter account details with different credentials. The user clicked on the edit option. The details populated in the text fields were not the same as the ones that the user added. The text fields were being populated from the SDN account. This issue is resolved.
RIFT-26559	Inside datacenter discovery host-aggregate and	After discovery was done on datacenters, the UI showed the host-aggregates and

Support Tickets	Title	Description
	availability-zone do not list any info.	availability-zones connected to the datacenters in the RESOURCE tab in the UI along with other resources like vim-quotas, interfaces, etc. The host-aggregates and availability-zones were only populated for datacenters with admin access. The data for host-aggregates and availability-zones was not populated in the UI. This issue is resolved.
RIFT-26593	In FAILED-INIT case TAKE-ACTION option is not shown; instead its asks for instantiation	During exception handling, mano was trying to call format() from rwpyplog instead of event(). This is invalid and it caused the problem. This issue is resolved.
RIFT-26594	NS in FAILED-INIT; terminate does not happen	While instantiating an NS during configuration update, the NS failed because there was one extra field passed in the rpc (placement_group_ref). The NS config admin status stayed in the Disabled state. When the user tried to terminate the request, the admin-status did not change because it was already in the Disabled state. This is why the request to terminate was not successful. This issue is resolved.
RIFT-26634	Elements which are 'Not' missing are populated under 'SERVICE ELEMENTS MISSING' section in Audit.	While running an Audit, missing data is populated in the "Service Elements Missing" section. In this scenario, elements were not missing from the NS, but NSR-ID was still appearing in the Missing Element section. Updated the UI to avoid confusion when showing missing data in the cloud. This issue is resolved.
RIFT-26643	Failure in Audit operation causes Failure in Account validation.	A failure in an audit operation caused a failure in account validation. Also, the audit feature was only workable in OpenStack Pike and above. If an audit attempt fails after the fix, then the account state is not affected. The audit

Support Tickets	Title	Description
		operation works from OpenStack version pike and above. This issue is resolved.
RIFT-26740	'File path missing error' appears when trying to add a new VNF using the update option	An error appeared when trying to add a VNF without providing the file path value. This issue is resolved.
RIFT-26565	Placement groups: error message http 405 doesn't indicate the reason for failure	After loading ping pong scaling descriptors with placement groups, a user deleted the placement groups from the NSD and VNFDs . Next, the user created and instantiated the NS. An error message appears but it was not clear in the event log. This issue is resolved.
RIFT-27243	Retry option is marking the Ping pong CNF service as success even though helms are in 'Failed' state in cloud	In a Kubernetes setup, a user uploaded attached descriptors and then created and instantiated a service. If there are no other deployments in the VIM, then it was successful. However, when the user tried to create another service with the same NSD descriptor and then instantiate, it failed. The user attempted to 'Retry', and it appeared like the retry option was successful in the Viewport screen, but the helm list still showed that it failed. This issue is resolved.
RIFT-27256	CNFR/release-info/service-info displays extra subcharts	The CNFR records displayed extra sub charts that were not present in the source helm charts. This issue is resolved.
RIFT-27258	CNFD xpath support: invalid event log entries "Member VNF Index is Invalid"	Launchpad logged a warning event log when the constituent-cnfd/member-id (NSD) was a string. Removed the check that only allowed integers. This issue is resolved.
RIFT-27259	CNFR release-info/service-info/deployments/pods/containers shows oam-sidecar twice	CNFR data was not being replaced while updating DTS. The containers were already present with the ID as 'NONE' so when the

Support Tickets	Title	Description
		same information with the same ID was received, DTS was keeping the older data. This issue is resolved.
RIFT-27264	Scale out creates 2 VNFS when an earlier scale-out had failed with auto-rollback enabled	When a VNF scale-out operation failed with rollback enabled, the scaled-out vnf was removed. The associated ns-instance-config record for scaling-group instance was not cleared. This caused a later scale out to add multiple VNFS. This issue is resolved.
RIFT-27267	Viewport details for service running on K8's as VIM are blank	An operator was running a service on Kubernetes as a VIM. The Viewport details were blank in the Datacenters, Management Interfaces and Virtual Links sections. This issue is resolved.
RIFT-27278	Dashboard Caas Widget to accommodate for duplicate CaaS Accounts	A user can configure multiple CaaS Accounts with the same credentials. In this case, the actual quote (capacity) of the account did not increase. This fix ensures that duplicate CaaS accounts are captured and quota is not added for duplicate accounts. This issue is resolved.
RIFT-27313	RBAC roles are not working for Sol02 interface	User permissions were not working in Sol002 interface for any users except for super admin. Any user with any role in project-admin, lcm-admin, vnfm-admin, vnfm-oper can now operate Sol002 interface. This issue is resolved.
RIFT-27317	Heal is not working through sol002 interface.	User permissions were not working to complete a heal operation in Sol002 interface for any users except for super admin. Any user with any role in project-admin, lcm-admin, vnfm-admin, vnfm-oper can now attempt a heal operation in Sol002 interface. This issue is resolved.

Support Tickets	Title	Description
RIFT-27319	LCM operational details are not working for a separate project (other than default)	The vnf_lcm_op_occs API was not working for projects other than the default project. This issue is resolved.
RIFT-27360	Errors in events list while ping pong scale out in K8 as VIM	A user uploaded ping pong descriptor for Kubernetes as a VIM and instantiated the NS. Next, the operator does a scale-out from the service Viewport option screen. After creating 2 or more ping pods, errors appear in the Events screen. This issue is resolved.
RIFT-27405	The customer values files also need to be fixed - modules/ext/vnfs/'customer'/cnfd	Modified customer values files for customer descriptors. (See RIFT-27426 for more information about this feature.) This issue is resolved.
RIFT-27426	Accommodate for rift-variables to be used as a substring in HELM values	RIFT.ware allows an update of the HELM values file with rift specific parameters. Inside the helm values file, the vendor can use "{{ input-variable.macvlan-sbi }}" syntax to refer to a rift specific variable. These rift specific variable are present as cnfd:input-variable. This issue is resolved.
RIFT-27432	Nsr topology for cnfr with microservices not appearing nicely	Information was not properly displayed on the Topology page. The details are now represented correctly on the Topology page. This issue is resolved.
RIFT-27434	NS terminate/delete not cleaning up K8s when Helm deployment fails	If a user attempts to instantiate a second NS on the same Kubernetes as a VIM as an existing NS, then instantiation should fail because there is already a helm release present. When the failed NS is deleted or terminated, the failed deployments in Kubernetes as VIM are not deleted. This issue is resolved.
RIFT-27435	No event log filter in UI for CNFD/CNF items	In the Events page in the UI, there was no filter option for any CNF related items. cnfr-id and cnfd-id were already present. Added cnfd-name.

Support Tickets	Title	Description
		This issue is resolved.
RIFT-27444	CNF monitoring params query should not use hardcoded Kubernetes master	Previously, cluster IP was incorrectly hard coded into Mano code. RIFT.ware now dynamically takes the cluster IP. This issue is resolved.
RIFT-27449	ns-instance-opdata monitoring-param entries for ping missing vnfr-mon-param-ref	The label in the monitoring params section of the UI was not correct. This issue is resolved.
RIFT-27453	Input variable substitution fails for more than 1 in a single line	A HELM values file requires multiple variables to be used in a single line. Fixed this limitation. This issue is resolved.
RIFT-27455	Monitoring param for K8 seems broken	The monitoring param for Kubernetes was not working correctly. This issue is resolved.
RIFT-27474	Event log errors when successfully instantiating a NS	Some error events (expected logs) were logged for successful instantiations. Converted these logs to info and warnings logs. This issue is resolved.
RIFT-27507	AMF Scaling primitive not working in multi-NF NS	An operator tried to submit an AMF scale primitive in the SERVICE-PRIMITIVE section in the UI and it did not complete. The service primitives are launched using the python3 asyncio subprocess library. It is not recommended to use proc.wait() when waiting for the script to exit. In this scenario, script output was high, and it led to a deadlock. This issue is resolved.
RIFT-27517	Scale-out input variable substitution broken	RIFT.ware supports input parameter in exec-sale-out RPC. Previously, MANO was not considering the input parameter that is passed through scale-out RPC. This issue is resolved.
RIFT-27552	Helm deployment not happening on restart	Kubernetes clusters delete resources for the future. This is the default behavior. If an operator wants to use a helm client directly to delete a release (not using rift-

Support Tickets	Title	Description
		mano), then it is necessary to use – purge for deletion and the cluster completely removes the resources. The user deleted a helm release without using purge and then restarted Launchpad. While restarting, RIFT.ware checks if a helm release is present in the cluster or not. In this scenario, the cluster returns the release info as a deleted state. RIFT.ware was not checking the state, so Launchpad assumed that the release was already present, and instantiation started. Added a helm release state check to address this issue. This issue is resolved.
RIFT-27555	NS in progress state even if release is not deployed	Mano was not checking the status of a helm release. The cluster was actually in the deleted state (see RIFT-27552). This caused the NS to stay in an In-Progress state. Added a state check for helm release. This issue is resolved.
RIFT-27556	UI not showing retry option while NS fail for CNFR	If a Network Service failed with an error stating that the CNFR release already exists, then there is no retry option in the UI. This issue is resolved.
RIFT-27643	Topology: both topology views sometime do not appear to be "up-to-date"	A user launched a Kubernetes service and watched the progress view. When the Viewport screen appears, the user clicked on topology and the connection lines were missing. After refreshing the page, the information appeared correctly on the UI. The user refreshed the page and the information appeared correctly on the UI. Users no longer need to refresh the browser page to view new content. This issue is resolved.
RIFT-27654	No microservices shown in topology view	The topology view did not reflect the microservices in an intermittent manner. This issue is resolved.

Support Tickets	Title	Description
RIFT-27824	Update running VDU's metadata to reflect the BrownField initiated LP details.	For the corresponding VDUs (VM) of a running NS in OpenStack, the metadata field had the details about the RIFT.ware Launchpad that instantiated it. A new Launchpad performs Brownfield Discover and takes over the LCM of the NS. Once Brownfield is successful, RIFT.ware should update the VDUs metadata with the details of the new Launchpad. This issue is resolved.

Known Issues in RIFT.ware 7.2.2

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-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. Reconfigure kaka account post failover.	N/A	Reconfigure kaka account post failover.
RIFT-23621	Introduce a new field in VNFM account page for	This is a request to add a new field in the VNFM accounts page	RIFT.ware uses a GET call to /vnf_instances to	N/A

Support Tickets	Title	Description	Impact	Workaround
	RIFT.ware<-->VNFM handshake URL.	that the Operator so the operator can populate the URL for handshaking. If the URL field is empty, handshake is not required.	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.	
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	Sometimes, the UI shows an error message when deleting an account. This error message appears in the UI	N/A

Support Tickets	Title	Description	Impact	Workaround
		selecting OpenStack → corresponding SDN account. After selecting the datacenter page, the user tried to delete the failed datacenter account and an error message appeared. Sometimes, the UI shows an error message when deleting an account. This error message appears in the UI when certain criteria is met.	when certain criteria is met.	
RIFT-23681	VIM Account connection status shows failure which was a success before.	A VIM account is created successfully but then an error message appears showing 'Failure connection status'. Refreshing the account does not fix the issue. Create a new account with the same details and the VIM account is created successfully. Create a new VIM account with the same details.	Create a new account with the same details and the VIM account is created successfully.	Create a new VIM account with the same details.
RIFT-23760	Rel-6.2: Terminating one service with the same name on any Launchpad interrupts existing	This issue occurs because all services on all Launchpads have the same UniqueID which is the InstanceID on ES2. When a user	If a user terminates any of the services on either of the network-services, this will make any other services orphan.	Use a unique NSR name if an organization is using multiple Launchpads. Ensure that an OpenStack tenant is

Support Tickets	Title	Description	Impact	Workaround
	services on all Launchpads.	terminates a service, then terminates Instance on ES2 which causes issues for all existing services because they have the same UniqueID.		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. An error message appeared on the UI.	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
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 is not observed in release 7.2.1.0.	N/A
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-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-25610	REL_7.1.1.0: AWS - NS stuck indefinitely in VNF-INIT phase when using a	A user tried to instantiate a simple NS with one VNF, a single VDU (for an Ubuntu VM), and a	No VNF is started by default. Since the min-instance-count for scaling group is 1, the VNF is	N/A

Support Tickets	Title	Description	Impact	Workaround
	scaling-group and constituent VNF's start-by-default = False.	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 forever in the VNF-INIT phase. There are no errors seen in the Event Logs or in rift.log.	supposed to be scaled out once NS instantiates.	
RIFT-26478	Audit button should be disabled when no datacenter is configured.	This is a request to disable the Audit button when no datacenter is configured in the UI.	The audit button is enabled even if no datacenter is configured.	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-26517	VNFR should show cloud-init with substituted variables.	A VDU's cloud-init can have Custom Meta Data variables. These variables can be substituted with actual values when a NS is instantiated and finally the cloud-init (with substituted values) is pushed into the VM. This is the actual cloud-init inside a VM. The issue occurs after NS instantiation. In the VNFR, a user	In Launchpad, an operator has no way of knowing whether the cloud-init (with substituted values) were actually injected into the VDUs.	Once the NS is instantiated, an operator can log into the VDUs and cross-check whether cloud-init was actually applied.

Support Tickets	Title	Description	Impact	Workaround
		should see the cloud-init with the substituted variables. However, in the VNFR the cloud-init includes the previous variables and not the substituted values.		
RIFT-26571	Placement groups: UI needs to provide a way override discovered details.	The placement group configuration of the Instantiation form depends on information the discovery process uncovers. This discovery process is checking 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 is not discovered, then the vnf deployment placement feature cannot be used.	Provide the appropriate credentials in the datacenter account configuration and run the discovery operation on that datacenter.
RIFT-25679	Exceptions in console during switching of project	While a user is switching from one project to another, then exceptions appear. These exceptions do not always occur.	N/A	N/A
RIFT-26229	No Name for the instances in the AWS	The name of the instance is missing in the AWS web console.	Users cannot match the name present in the RIFT Launchpad to the AWS web console.	The instance ID can be used to identification purposes.

Support Tickets	Title	Description	Impact	Workaround
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 prints 5 traps riftLPUserSessionEnd	An operator loaded ping pong scaling descriptors and then deleted placement groups from the NSD and VNFDs. Next, the user created and instantiated the NS. An error message and 5 traps appear.	N/A	N/A
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	The output does not appear correctly in the UI.	The output of the initial script can be found in the location - RVR/log/cfgmgr.

Support Tickets	Title	Description	Impact	Workaround
		- initial config primitive.' The script did not run for the original VNF present in the service.		
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 the instantiation attempt.	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.
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	N/A	N/A

Support Tickets	Title	Description	Impact	Workaround
		not applied to the targets. If the operator changes the usage of the instantiation variables, then the results are different and unpredictable.		
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-27680 From Venkat's query	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'.	N/A	N/A