Managing VNF Lifecycle Using ETSI API

The NFVO communicates with ESC using the ETSI MANO API for lifecycle management of a VNF. A configuration template, the Virtual Network Function Descriptor (VNFD) file describes the deployment parameters and operational behaviors of a VNF type. The VNFD is used in the process of deploying a VNF and managing the lifecycle of a VNF instance.

The lifecycle operations of a VNF instance is as follows:

1. **Create a VNF Identifier**—ESC generates a new VNF Instance Id (a universally unique identifier) that is subsequently used as a handle to reference the instance upon which to execute further operations.

2. **Instantiate / Deploy VNF**—As part of VNF instantiation, ESC instantiates a new VNF instance in the VIM. ESC receives a request to instantiate a VNF instance from NFVO. The instantiate request contains resource requirements, networking and other service operational behaviors. All these requirements along with the VNFD and the grant information provides all the necessary information to instantiate the VNF.

3. **Operate VNF**—ESC allows you to start and stop a VNF instance. The resources are not released or changed, but the VNF instance in the VIM is toggled between these two states.

4. **Query VNF**—To query one or more VNF instances known to ESC. To query one or more VNF instances known to ESC. This is a specific REST end point that can be filtered to find specific instances. In ESC Release 4.0, the instances can be filtered using the VNF Instance Id.

   Also, a separate REST end point allows the NFVO to query the status of one or more lifecycle operation occurrences associated with a VNF. In ESC Release 4.0, the lifecycle operations can be filtered using a specific occurrence identifier.

5. **Terminate / Undeploy VNF**—To terminate the VNF instance in the VIM. The resources themselves remain reserved for the VNF instance, however the VNF itself is undeployed.

6. **Delete VNF Identifier**—The resources are fully released in the VIM and in ESC and the associated VNF instance identifier is also released.

For VNF lifecycle operations using REST and NETCONF APIs, see Deploying and Configuring Virtual Network Functions.

- VNF Lifecycle Operations Using ETSI API, page 2
VNF Lifecycle Operations Using ETSI API

Prerequisites

The following prerequisites must be met for VNF lifecycle operations:

- The resource definitions must be created out of band and must be available before VNF instantiation.
- The VIM connector specifies the VIM used. The VIM connector information must be available before instantiating the VNF. For ESC Release 4.0, ESC must contain a valid, and authenticated default OpenStack VIM connector to deploy the VNFs.
- The VNF to be instantiated has to be onboarded to the NFVO within an ETSI compliant VNF Package.
  - The NFVO must provide ETSI compliant VNF Packages to ESC.
  - The VNF package must contain a VNF Descriptor (VNFD) file.

The ETSI MANO NFV specifications supports /vnf_packages API to allow access to the package artifacts. See chapter 10 in the ETSI GS NFV-SOL 003 V2.3.1 specification on the ETSI website.

The properties file provides details about the NFVO to the ETSI VNFM service.
The properties file, etsi-vnfm.properties exists under: /opt/cisco/esc/esc_database

For ESC Release 4.0, the single property nfvo.apiRoot allows specification of the NFVO host and port.
For example, nfvo.apiRoot=localhost:8280

Note

The initial implementation of the ETSI MANO API supports only a single VIM and a single (default) tenant.

For notes on ESC in HA mode, with ETSI service enabled, see the Cisco Elastic Services Controller Install and Upgrade Guide.

Creating the VNF Identifier

Creating the VNF Identifier is the first request for any VNF instance. This identifier is used for all further LCM operations executed by the ETSI API. Resources are neither created nor reserved at this stage.

ESC sends a POST request to create VNF instances:

Method Type:
POST

VNFM Endpoint:
/vnf_instances/

HTTP Request Headers:
Content-Type:application/json

Request Payload(ETSI data structure: CreateVnfRequest):
{
  "vnfInstanceName": "Test-VNF-Instance",
  "vnfdId": "vnfd-88c6a03e-019f-4525-ae63-de58ee89db74"
}
Response Headers:

HTTP/1.1 201
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Application-Context: application:8250
Accept-Ranges: none
Content-Type: application/json;charset=UTF-8
Transfer-Encoding: chunked
Date: Thu, 04 Jan 2018 12:18:13 GMT

Response Body (ETSI Data structure:VnfInstance)

```json
{
    "_links": {
        "instantiate": {
        },
        "self": {
        }
    },
    "id": "14924fca-fb10-45da-bcf5-59c581d675d8",
    "instantiationState": "NOT_INSTANTIATED",
    "onboardedVnfPkgInfoId": "vnfpkg-bb5601ef-cae8-4141-ba4f-e96b6cad0f74",
    "vnfInstanceName": "Test-VNF-Instance",
    "vnfProductName": "vnfd-1VDU",
    "vnfProvider": "",
    "vnfSoftwareVersion": "not-implemented",
    "vnfdId": "vnfd-88c6a03e-019f-4525-ae63-de58ee89db74",
    "vnfdVersion": ""
}
```

Instantiating Virtual Network Functions

The instantiation request triggers a number of message exchanges, which allows the call flow to be completed in order to instantiate a VNF instance. The resources are allocated when the VNF instance is instantiated. It requires the VNF instance identifier, returned by the create VNF request, encoded into the URL to which the request is posted.

The instantiation request sub-tasks within the flow include:

1. Requesting permission from the NFVO (bi-directional Grant flow). For more information see, Requesting Grant Permission.
2. Retrieving the VNF Descriptor template from the NFVO.
3. Requesting the VNF resources to be allocated to the instance from the VIM.

Typically, the request permission, and retrieve messages are customized for the NFVO.

Method type: POST

VNFM Endpoint:
```
/vnf_instances/{vnfInstanceId}/instantiate
```
HTTP Request Header:
Content-Type: application/json

Request Payload (ETSI data structure: InstantiateVNFRRequest)
{
    "flavourId": "vdu_node1",
    "extManagedVirtualLinks": [
        {
            "id": "esc-net",
            "resourceId": "RES1",
            "virtualLinkDescId": "VLD1"
        }
    ],
    "vimConnectionInfo": [
        {
            "accessInfo": {
                "resourceGroupId": "tenantName"
            },
            "id": "vimConnId",
            "vimType": "Openstack"
        }
    ]
}

Response Headers:
HTTP/1.1 202
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Application-Context: application:8250
Accept-Ranges: none
Location: http://localhost:8250/vnflcm/v1/vnf_lcm_op_occs/457736f0-c877-4e07-8055-39dd406c616b
Content-Length: 0
Date: Thu, 04 Jan 2018 12:20:40 GMT

Response Body:
not applicable.

Requesting Grant Permission

The ETSI API requests for permission from the NFVO for lifecycle management operations to complete the operations in progress for the VNF instance resources.

{  
    "vnfInstanceId": "b9909dde-e21e-45ec-9cc0-9e9ae413ee0",
    "vnfdId": "d1409dde-f23e-61fb-9dc4-9e98e66e9gb7",
    "flavourId": "aaaa-bbbb-1111-2222-cccc-1a3c5bb6543",
    "flavourId": "vanilla", /* not required? */
    "operation": "instantiate",
    "isAutomaticInvocation": "false", /* not required? */
    "instantiationLevelId": "admin", /* not required? */
    "addResources": { /* not required? */
        "": ""
    }, /* not required? */
    "tempResources": "", /* not required? */
    "placementConstraints": {
        "affinityOrAntiAffinity": "ANTI_AFFINITY",
        "scope": "ZONE",
        "resource": { /* not required? */
            "idType": "GRANT",
            "resourceId": "myresourceid123"
        }, /* not required? */
        "additionalParams": { /* not required? */
            "project": "tenant1"
Querying Virtual Network Functions

Querying VNFs does not affect the state of any VNF instance. This operation simply queries ESC for all the VNF instances it knows about, or a specific VNF instance.

Method Type:
GET

VNFM Endpoint:
/vnflcm/v1/vnf_instances
/vnflcm/v1/vnf_instances/{vnfInstanceId}

HTTP Request Headers:
Content-Type: application/json

Request Payload:
not applicable.

Response Headers:
HTTP/1.1 200
X-Content-Type-Options: nosniff
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Pragma: no-cache
Expires: 0
Expires: 0
X-Frame-Options: DENY
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000; includeSubDomains
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Application-Context: application:8250
X-Application-Context: application:8250
Accept-Ranges: none
Accept-Ranges: none
Content-Type: application/json; charset=UTF-8
Content-Type: application/json; charset=UTF-8
Transfer-Encoding: chunked
Transfer-Encoding: chunked
Date: Thu, 04 Jan 2018 12:25:32 GMT
Date: Thu, 04 Jan 2018 12:25:32 GMT

Response Body for a single VNF Instance (ETSI Data structure: VnfInstance)

```
{
  "links": {
    "instantiate": {
      "href":
    },
    "self": {
      "href":
    }
  }
```
Response Body for all VNF Instances (ETSI Data structure: VnfInstance[])

```
{
  "_embedded": {
    "vnfInstances": [
      {
        "_links": {
          "instantiate": {
          },
          "self": {
          }
        },
        "id": "14924fca-fb10-45da-bcf5-59c581d675d8",
        "instantiationState": "NOT_INSTANTIATED",
        "onboardedVnfPkgInfoId": "vnfpkg-bb5601ef-cae8-4141-ba4f-e96b6cad0f74",
        "vnfInstanceId": "Test-VNF-Instance",
        "vnfProductName": "vnfd-1VDU",
        "vnfProvider": "",
        "vnfSoftwareVersion": "not-implemented",
        "vnfdId": "vnfd-88c6a03e-019f-4525-ae63-de58ee89db74",
        "vnfdVersion": ""
      },
      {
        "_links": {
          "instantiate": {
            "href": "http://localhost:8250/vnflcm/v1/vnf_instances/9b82e224-2be5-44d8-821b-64ad7f4997fb/instantiate"
          },
          "self": {
            "href": "http://localhost:8250/vnflcm/v1/vnf_instances/9b82e224-2be5-44d8-821b-64ad7f4997fb"
          }
        },
        "id": "9b82e224-2be5-44d8-821b-64ad7f4997fb",
        "instantiationState": "NOT_INSTANTIATED",
        "onboardedVnfPkgInfoId": "vnfpkg-bb5601ef-cae8-4141-ba4f-e96b6cad0f74",
        "vnfInstanceId": "Test1-VNF-Instance",
        "vnfProductName": "vnfd-1VDU",
        "vnfProvider": "",
        "vnfSoftwareVersion": "not-implemented",
        "vnfdId": "vnfd-88c6a03e-019f-4525-ae63-de58ee89db74",
        "vnfdVersion": ""
      }
    ]
  },
  "_links": {
    "self": {
      "href": "http://localhost:8250/vnflcm/v1/vnf_instances"
    }
  }
}
```
Operating Virtual Network Functions

You can start or stop a VNF instance using the operate lifecycle management operation. The VNF instance can be stopped gracefully or forcefully.

Permission is also required from the NFVO (bi-directional Grant flow) for this operation. See Requesting Grant Permission for more information.

Method Type:
POST

VNFM Endpoint:
/vnf_instances/{vnfInstanceId}/operate

HTTP Request Headers:
Content-Type: application/json

Request Payload:
{
  "additionalParams": {
    "username": "user1"
  },
  "changeStateTo": "STOPPED",
  "gracefulStopTimeout": "5",
  "stopType": "GRACEFUL"
}

Response Headers:
HTTP/1.1 202
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
Accept-Ranges: none
Location: http://localhost:8250/vnflcm/v1/vnf_lcm_op_occs/e775aad5-8683-4450-b260-43656b6b13e9
Content-Length: 0
Date: Thu, 04 Jan 2018 12:40:27 GMT

Response Body:
not applicable.

Terminating Virtual Network Functions

The terminating VNF request terminates a VNF instance. The resources are deallocated but remain reserved for this instance until it is deleted. Permission is required from the NFVO (bi-directional Grant flow) for this operation. The VNF instance can be decommissioned gracefully or forcefully.

As per the Instantiate VNF Request, the terminate VNF request requires the VNF instance identifier encoded into the URL to which the request is posted.

Method Type:
POST

VNFM Endpoint:
/vnf_instances/{vnfInstanceId}/terminate

HTTP Request Headers:
Content-Type: application/json
Deleting Virtual Network Functions

Deleting VNF operation releases the VIM resources reserved for the VNF instance as well as deletes the VNF instance identifier. Upon deletion, the VNF instance identifier is no longer available. So, no further lifecycle management operations are possible using this identifier.

Method Type:
DELETE

VNFM Endpoint:
/vnf_instances/{vnfInstanceId}

HTTP Request Headers:
Content-Type:application/json

Request Payload:
not applicable.

Response Headers:
HTTP/1.1 204
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Application-Context: application:8250
Accept-Ranges: none
Location: http://localhost:8250/vnf lcm/v1/vnf lcm_op_occs/dae25dbc-fcde-4ff9-8fd6-31797d19dbc1
Content-Length: 0
Date: Thu, 04 Jan 2018 12:45:59 GMT

Response Body:
not applicable.

Request Payload (ETSI data structure: TerminateVnfRequest)

{  "terminationType":"GRACEFUL”,  "gracefulTerminationTimeout": "10",  "additionalParams": {    "password": "pass1234",    "username": "admin"  }}

Response Headers:
HTTP/1.1 202
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
X-Frame-Options: DENY
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Application-Context: application:8250
Accept-Ranges: none
Location: http://localhost:8250/vnf lcm/v1/vnf lcm_op_occs/dae25dbc-fcde-4ff9-8fd6-31797d19dbc1
Content-Length: 0
Date: Thu, 04 Jan 2018 12:45:59 GMT

Response Body:
not applicable.
Accept-Ranges: none
< Date: Thu, 04 Jan 2018 12:48:59 GMT
Date: Thu, 04 Jan 2018 12:48:59 GMT

Response Body:
not applicable.