Deploying Virtual Network Functions

This chapter describes several deployment scenarios for Elastic Services Controller (ESC), the procedures to deploy VNFs (OpenStack and VMware environments), and the operations that you can perform during deployment.

Important

Starting from ESC Release 2.0, you can assign a static IP address to connect the network to the VNF. The deployment datamodel introduces a new `ip_address` attribute to specify the static IP address. See the Cisco Services Controller Deployment Attributes for more details.

The following table lists the VNF features that are supported on OpenStack or VMware:

<table>
<thead>
<tr>
<th>Feature</th>
<th>OpenStack</th>
<th>VMware</th>
</tr>
</thead>
</table>
| Affinity and Anti-affinity Rule | Creating and deleting affinity and anti-affinity rule definitions is done in one of the following ways:  
• NETCONF API  
• REST API | Creating and deleting affinity rule definition in one of the following ways:  
• NETCONF API  
• REST API |
| Note                        | ESC 2.1 supports only Affinity rules in VMware environment. |                                             |
| Single Root I/O Virtualization | Configuration of Single Root I/O Virtualization is done in one of the following ways:  
• NETCONF API  
• REST API | —                                           |
| Individual and Composite VNFs | Configuration of Individual and Composite VNFs is done in one of the following ways:  
• NETCONF API  
• REST API | Configuration of Individual and Composite VNFs is done in one of the ways:  
• NETCONF API  
• REST API |
<table>
<thead>
<tr>
<th>Feature</th>
<th>OpenStack</th>
<th>VMware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undeploy Virtual Network Functions</td>
<td>Undeploying is done in one of the following ways:</td>
<td>Undeploying VNFs is done in one of the following ways:</td>
</tr>
<tr>
<td></td>
<td>• NETCONF API</td>
<td>• NETCONF API</td>
</tr>
<tr>
<td></td>
<td>• REST API</td>
<td>• REST API</td>
</tr>
<tr>
<td></td>
<td>• ESC Portal</td>
<td>• ESC Portal</td>
</tr>
<tr>
<td>Day Zero Configuration</td>
<td>Day Zero configuration is done in one of the following ways:</td>
<td>Day Zero configuration is done in one of the following ways:</td>
</tr>
<tr>
<td></td>
<td>• NETCONF API</td>
<td>• NETCONF API</td>
</tr>
<tr>
<td></td>
<td>• REST API</td>
<td>• REST API</td>
</tr>
<tr>
<td></td>
<td>• ESC Portal</td>
<td>• ESC Portal</td>
</tr>
<tr>
<td>VNF Operations</td>
<td>VNF Operations is done in one of the following ways:</td>
<td>VNF Operations is done in one of the following ways:</td>
</tr>
<tr>
<td></td>
<td>• REST API</td>
<td>• REST API</td>
</tr>
<tr>
<td></td>
<td>For more information, see the Virtual Network Function Operations.</td>
<td>For more information, see the Cisco Elastic Services Controller Portal.</td>
</tr>
<tr>
<td>Multi Cluster</td>
<td>Not applicable</td>
<td>Multi Cluster configuration is done in one of the following ways:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• REST API</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ESC Portal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For more information, see the Deploying Virtual Network Functions in ESC Portal (VMware Only), on page 7.</td>
</tr>
<tr>
<td>Multiple Virtual Datacenter</td>
<td>Not applicable</td>
<td>Multiple Virtual Datacenter selection is done in one of the following ways:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• REST API</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• ESC Portal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For more information, see the Multi-Virtual Data Center Support (VMware Only), on page 6.</td>
</tr>
</tbody>
</table>
Deploying Virtual Network Functions in the OpenStack Environment

This section describes several deployment scenarios for Elastic Services Controller (ESC) and the procedure to deploy VNFs. The following table lists the different deployment scenarios:

<table>
<thead>
<tr>
<th>Feature</th>
<th>OpenStack</th>
<th>VMware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Acceleration</td>
<td>Hardware Acceleration is supported in one of the following ways:</td>
<td>Not applicable</td>
</tr>
<tr>
<td></td>
<td>• NETCONF API</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• REST API</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For more information, see the Hardware Acceleration Support (OpenStack Only), on page 19.</td>
<td></td>
</tr>
</tbody>
</table>

After you have decided on the deployment scenario and the environment you want to implement, see the following sections:

- Deploying Virtual Network Functions in the OpenStack Environment, page 3
- Deploying Virtual Network Functions in the VMware Environment, page 5
- Deploying Virtual Network Functions in ESC Portal (VMware Only), page 7
- Day-0 Configuration, page 9
- Unified Deployment Request, page 11
- Applying Affinity and Anti-affinity Rules, page 12
- Advanced Interface Configurations, page 16
- Managing Individual and Composite VNFs, page 18
- Hardware Acceleration Support (OpenStack Only), page 19
- Undeploying Virtual Network Functions, page 19
- Updating Deployments, page 20
Deploying Virtual Network Functions

In ESC, VNF deployment is initiated as a service request either originating from the ESC portal or the northbound interfaces. The service request comprises of XML payloads. These resources must either be available in OpenStack using the northbound interfaces. ESC supports the following deployment scenarios:

- Deploying VNFs (Creating images, and flavors through ESC)
- Deploying VNFs (Using out-of-band images, flavors, and volumes)

Before you deploy VNFs, you must ensure that images, flavors, and volumes are available in OpenStack or you must create these resources. For more details on creating images, flavors, and volumes see Managing VM Resources in ESC.

During deployment, ESC looks for the deployment details in the deployment datamodel. For more information on the deployment datamodel, see Cisco Elastic Services Controller Deployment Attributes. If ESC is unable to find the deployment details for a particular service, it uses the existing flavor and image under the vm_group to continue the deployment. If ESC is unable to find the image, and flavor details, the deployment fails.

<table>
<thead>
<tr>
<th>Scenarios</th>
<th>Description</th>
<th>Datamodel</th>
<th>Images, Flavors, and Volumes</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deploying VNFs (Creating Images and Flavors through ESC)</td>
<td>The process of VNF deployment is as follows:</td>
<td>• deployment datamodel</td>
<td>Images and Flavors are created through ESC using NETCONF/REST APIs.</td>
<td>• The images and flavors can be used in multiple VNF deployments.</td>
</tr>
<tr>
<td></td>
<td>1 VNF Deployment - The deployment datamodel refers to the images and flavors created and then deploys VNFs.</td>
<td>• image datamodel</td>
<td></td>
<td>• You can add or delete image definitions through ESC.</td>
</tr>
<tr>
<td>Deploying VNFs (Using out-of-band images, flavors, and volumes)</td>
<td>The process of VNF deployment is as follows:</td>
<td>• flavor datamodel</td>
<td>Images, Flavors and Volumes are not created through ESC.</td>
<td>• The images, flavors, and volumes can be used in multiple VNF deployments.</td>
</tr>
<tr>
<td></td>
<td>1 VNF Deployment - The deployment datamodel refers to the out-of-band images, flavors, and volumes in OpenStack and then deploys VNFs.</td>
<td>• Images, Flavors, and Volumes in OpenStack</td>
<td></td>
<td>• You cannot add or delete images through ESC.</td>
</tr>
</tbody>
</table>
From ESC Release 2.0 and later, you can also specify the subnet that is used for a network. The deployment datamodel introduces a new `subnet` attribute to specify the subnet. See the Cisco Elastic Services Controller Deployment Attributes for more details.

For deployments in OpenStack, the UUID or name can be used to refer to the image and flavor. The name has to be unique on VIM. If there are multiple images with the same name, the deployment cannot identify the right image and the deployment fails.

## Deploying Virtual Network Functions in the VMware Environment

This section describes the deployment scenario for Elastic Services Controller (ESC) and the procedure to deploy VNFs in the VMware environment. In ESC Release 2.1, you can deploy VNFs using out-of-band image definitions. The following table lists the deployment scenario:

<table>
<thead>
<tr>
<th>Scenarios</th>
<th>Description</th>
<th>datamodel templates</th>
<th>Images</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deploying VNFs (Creating Images through ESC)</td>
<td>The process of VNF deployment is as follows:</td>
<td></td>
<td>Images are created through ESC using REST APIs.</td>
<td>• The images can be used in multiple VNF deployments.</td>
</tr>
<tr>
<td><strong>Important</strong></td>
<td>Images are also referred to as Templates in the VMware environment</td>
<td></td>
<td></td>
<td>• You can add or delete image definitions through ESC.</td>
</tr>
<tr>
<td></td>
<td>VNF Deployment- The <code>deployment datamodel</code> refers to the images created and then deploys VNFs.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deploying VNFs (Using out-of-band images)</td>
<td>VNF Deployment- The <code>deployment datamodel</code> refers to the out-of-band images in VMware and then deploys VNFs.</td>
<td></td>
<td>Images cannot be created or deleted through ESC.</td>
<td>• The images can be used in multiple VNF deployments.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• You can view images through ESC portal.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• During out-of-band deployment, you can choose images.</td>
</tr>
</tbody>
</table>
Multi-Virtual Data Center Support (VMware Only)

A Virtual Data Center (VDC) is an environment that combines virtual resources, operational details, rules, and policies to manage specific group requirements. A group can manage multiple VDCs, images, templates, and policies. This group can allocate quotas and assign resource limits for individual groups at the VDC level.

When you deploy a VNF, you need to specify the virtual datacenter name on which the VNF needs to be provisioned. You need to specify the datacenter name as below:

```xml
<deployments>
  <deployment>
    <datacenter>OTT-03</datacenter>
  </deployment>
  ...
</deployments>
```

If the data center name is not specified during deployment, ESC deploys the VM in a default data center that is provided in the configuration parameters.

Cisco Elastic Services Controller Portal allows you to choose the VDC on which the VM is provisioned. When you are creating a service request, you can choose the VDC on which this VM is provisioned. For more information on deploying VNF on a VDC, see Deploying Virtual Network Functions in ESC Portal (VMware Only), on page 7.

To view the list of VDCs that are available and on the ESC portal, choose Resources > Datacenter.

**Before you Begin**

Before you deploy VNFs on multiple VDCs, ensure that the following conditions are met:

- Verify that a standard external network spanning both VDCs is available for the ESC to ping the deployed VMs.
- Verify that at least one management interface on the VMs is connected to the external network.
- Verify that the VDC is present in the vCenter.

**Limitations**

The following is the list of VDC limitations:

- Supported only through REST interface.
- Supports only deployment and not resource management. ESC assumes all required resources to be created in VDC are out of band and present in the VDC.
- ESC does not support admin level control on VDC scoping. Any present VDC on a vCenter can be accessed by ESC.
Passing OVF Properties to a VM

As a part of deploying a VNF in VMware environment, you can pass the name value pair as OVF property to the VM. To pass these configurations while deploying a VNF, you must include additional arguments in the deployment datamodel template.

A sample configuration is as follows:

```xml
<esc_datamodel ...>
  ...
  <config_data>
    <configuration>
      <dst>ovfProperty:mgmt-ipv4-addr</dst>
      <data>$NICID_1_IP_ADDRESS/24</data>
    </configuration>
    <configuration>
      <dst>ovfProperty:com.cisco.csr1000v:hostname</dst>
      <data>$HOSTNAME</data>
      <variable>
        <name>HOSTNAME</name>
        <val>csrhost1</val>
        <val>csrhost2</val>
      </variable>
    </configuration>
  </config_data>
  ...
</esc_datamodel>
```

Deploying Virtual Network Functions in ESC Portal (VMware Only)

The ESC portal allows you deploy a single VNF or multiple VNFs together. An existing deployment datamodel is either uploaded through the portal, or a new deployment datamodel created. A new deployment datamodel is created by filling all the appropriate fields in the ESC portal. ESC also allows you to export a deployment datamodel from the portal. The following section explains multiple ways to deploy VNFs using the ESC portal.

Deploy Using a File (deployment datamodel)

An existing deployment datamodel is used to deploy VNFs. The deployment datamodel is pre configured with the number of VNFs and other specifications. It is either uploaded by locating the deployment datamodel or you can drag and drop the existing deployment datamodel. The drag and drop feature allows you to grab an existing deployment datamodel and to re-use it by dragging the file to the drop off area.

---

Note

Only XML files are accepted.
### Procedure

**Step 1** Choose Deployments.

**Step 2** Click New Deployment.

**Step 3** Select Deploy from a File.

**Step 4** Do one of the following:

a) Click the Drop File Here and locate the file.

b) Drag and drop your file to the Drop File Here area.

**Note** The drag and drop feature executes a REST call as of now and does not execute NETCONF calls.

---

### Deploying Using a Form

To create a new deployment template, do the following:

**Note** Click Export Template to export a deployment datamodel.

---

### Procedure

**Step 1** Choose Deployments.

**Step 2** Click New Deployment.

**Step 3** Select the Deploy from a form.

**Step 4** Enter a Deployment name.

**Step 5** From the Datacenter drop-down list, choose a datacenter on which you want to deploy the VNF. For more information on virtual datacenter, see Multi-Virtual Data Center Support (VMware Only), on page 6.

**Step 6** In the General tab, enter the appropriate values for the fields.

a) In the Placement field, select the Cluster or Host radio button.

   - **Cluster**—Choose the name of a cluster to deploy a VNF in the same cluster.
   - **Host**—Choose a host to deploy a VNF in the same host.
   - **Datastore**—Choose a datastore for the selected cluster.
   - **Image** Choose an image.

**Step 7** Click Enable Smart Licensing to enable smart licensing.

**Step 8** Click Enable Intragroup Rules to enable intragroup rules.

a) From the Type drop-down list, choose Affinity to enable affinity rules.

For more information on intragroup affinity rules, Applying Affinity and Anti-affinity Rules, on page 12.
Step 9  (Optional): Click the Add VNF Intergroup Rule tab to select VNFs for which you the affinity rules to be applicable.
For more information on intergroup affinity rules, see Applying Affinity and Anti-affinity Rules, on page 12.

Step 10  To specify the parameters that ESC will utilize to heal the VNFs when there is a failure, click the Recovery tab.
For more information on recovery or healing, see Healing Virtual Network Functions.

Step 11  To specify the number of interfaces and properties for each interface, click the Interfaces tab. The order of the interfaces specified here does not correspond to the order of the interfaces in the VM.
   a) Click Add Interface to add interfaces.

Step 12  To specify the number of instances of a particular type of VM that needs to be instantiated and to elastic scale in and scale out, click the Scaling tab.
   a) Click Add Static IP Pool to add a static IP pool.

Step 13  To specify the monitoring rules that will be used to configure the monitor module within ESC, click the Monitoring tab.
For more information on monitoring, see Monitoring Virtual Network Functions.

Step 14  In the Config Data tab, enter the appropriate values for the fields.
Step 15  In the OVF Settings tab, enter the appropriate values for the fields.
   a) Click Add OVF Property to add a list of OVF properties.

Day-0 Configuration

The initial or day-0 configuration of a VNF is based on the VM type. A VNF administrator configures the initial template for each VM type at the time of VNF deployment. The same configuration template is applied to all deployed and new VMs of that VM type. The template is processed at the time of individual VM deployment. The day-0 configuration continues to persists, so that all initial deployment, healing and scaling of VMs have the same day-0 template.

Some of the day-0 configuration tasks include bringing up the interface, managing the network, support for static or dynamic IP (DHCP, IPAM), SSH keys, and NetConf enabled configuration support on VNF.

Day-0 in the Configuration datamodel

Day 0 configuration is defined in the datamodel under the config_data tag. Each user data and the configuration drive file is defined under the configuration tag. The contents are in the form of a template. ESC processes the template through the Apache Velocity Template Engine before passing to the VM.

The config_data tag is defined for each vm_group. The same configuration template is applied to all VMs in the vm_group. The template file is retrieved and stored at deployment initialization. Template processing is applied at time of VM deployment. The content of the config file can be retrieved from the file or data

<file> url </file>
<data> inline config content </data>

The url specifies a file on the ESC VM file system or file hosted on report http server.

A destination name is assigned to the config by <dst>. User Data is a treated as a special case with
<dst>--user-data</dst>.
A sample config datamodel,

```xml
<config_data>
<configuration>
<file>file://cisco/userdata_file.txt</file>
<dst>--user-data</dst>
<variable>
<name>CUSTOM_VARIABLE_FOR_USERDATA</name>
<val>SOME_VALUE_XXX</val>
</variable>
</configuration>
<configuration>
<file>file://cisco/config.sh</file>
<dst>config.sh</dst>
<variable>
<name>CUSTOM_VARIABLE_FOR_CONFIG</name>
<val>SOME_VALUE_XXX</val>
</variable>
</configuration>
</config_data>
```

Custom variable can be specified in the variables tag within the configuration. Zero or more variables can be included in each configuration. Each variable can have multiple values. Multiple values are only useful when creating more than one VM per vm_group. Also, when performing scale-in and scale-out, additional VMs can be added and removed from the VM group.

The contents of `<file>` are a template that is processed by the Velocity Template Engine. ESC populates a set of variables for each interface before processing the configuration template:

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NICID_n_IP_ALLOCATION_TYPE</td>
<td>string containing FIXED</td>
</tr>
<tr>
<td>NICID_n_NETWORK_ID</td>
<td>string containing neutron network uuid</td>
</tr>
<tr>
<td>NICID_n_IP_ADDRESS</td>
<td>ipv4 or ipv6 address</td>
</tr>
<tr>
<td>NICID_n_MAC_ADDRESS</td>
<td>string</td>
</tr>
<tr>
<td>NICID_n_GATEWAY</td>
<td>ipv4 or ipv6 gateway address</td>
</tr>
<tr>
<td>NICID_n_CIDR_ADDRESS</td>
<td>ipv4 or ipv6 cidr prefix address</td>
</tr>
<tr>
<td>NICID_n_CIDR_PREFIX</td>
<td>integer with prefix-length</td>
</tr>
<tr>
<td>NICID_n_NETMASK</td>
<td>If an ipv4 CIDR address and prefix are present, ESC will automatically calculate and populate the netmask variable. This is not substituted in the case of an IPv6 address and should not be used.</td>
</tr>
<tr>
<td>NICID_n_ANYCAST_ADDRESS</td>
<td>string with ipv4 or ipv6</td>
</tr>
<tr>
<td>NICID_n_IPV4_OCTETS</td>
<td>string with last 2 octets of ip address, such as 16.66, specific to CloudVPN</td>
</tr>
</tbody>
</table>

Where n is the interface number from the datamodel, for example, 0, 1, 2, 3

Example

```
NICID_0_IP_ALLOCATION_TYPE: FIXED
NICID_0_NETWORK_ID: 9f8d9a97-d873-4a1c-8e95-1a123686f038
```
NICID_0_IP_ADDRESS: 2a00:c31:7fe2:1d:0:0:1:1000
NICID_0_MAC_ADDRESS: null
NICID_0_GATEWAY: 2a00:c31:7fe2:1d:1
NICID_0_CIDR_ADDRESS: 2a00:c31:7fe2:1d:
NICID_0_CIDR_PREFIX: 64
NICID_0_ANYPACT_ADDRESS: null
NICID_0_IPV4_OCTETS: 16.0
NICID_1_IP_ALLOCATION_TYPE: DHCP
NICID_1_NETWORK_ID: 0c468d8e-2385-4641-b1db-9080c170cb1a
NICID_1_IP_ADDRESS: 6.0.0.2
NICID_1_MAC_ADDRESS: null
NICID_1_GATEWAY: 6.0.0.1
NICID_1_CIDR_ADDRESS: 6.0.0.0
NICID_1_CIDR_PREFIX: 24
NICID_1_ANYPACT_ADDRESS: null
NICID_1_NETMASK: 255.255.255.0

You must follow these tips while processing the template through the velocity template engine.

- To escape dollar sign in the template insert,
  ```
  #set ($DS = "$")
  then replace the variable with
  passwd: ${DS}1${DS}h1VxC40U${DS}uf2qLUwGTjHgZplkP78xA
  ```

- To escape a block in the template, insert `#[[ and #]`. For example,
  `#[[ passwd: $1$h1VxC40U$uf2qLUwGTjHgZplkP78xA ]]#`

## Unified Deployment Request

ESC creates OpenStack resources such as tenants, networks, and sub-networks before deploying a VNF. During a unified deployment request, you send a unified request to create or delete the OpenStack resources, and deploy a VNF. You can create multiple networks and sub-networks, but can create only a single VNF and a single tenant using unified deployment request.

Update the deployment datamodel and the files with the necessary information such as the service and deployment ID, tenant, network and sub-network ids and so on. You can either use NETCONF or REST APIs. For example, send POST REST and DELETE REST calls.

See the Elastic Services Controller Registration and Deployment Attributes for a list of registration and deployment attributes.

- To create and update deployment datamodel with a single deployment request, send POST REST call to:
  ```
  http://[ESC_IP]:8080/v0/deployments/[internal_dep_id]
  ```

- To delete a single deployment request, send DELETE REST call to:
  ```
  http://[ESC_IP]:8080/v0/deployments/[internal_dep_id]
  ```

The VNF will be undeployed, and the subnet, network and tenant will be deleted in the specified order.

**Note**

- If a tenant is not created as part of unified deployment request, there is nothing to roll back. An error occurs and a manual un-deploy must be triggered to cleanup.

- If both create and delete unified deployment requests fail causing two failed events, there is an attempt to roll back twice. But, there is nothing to rollback.
Limitations:

- A unified deployment request can reuse an existing tenant only. It cannot reuse any existing networks, sub-networks or deployments. If there is a duplicate network, sub-network, or deployment, then the unified deployment request is rejected.
- During an undeploy request, any network and subnetwork created as part of the unified deployment request will be deleted along with the VNF. However, tenant created through unified deployment request will not be deleted.

**Applying Affinity and Anti-affinity Rules**

Affinity and anti-affinity rules create relationship between virtual machines (VMs) and hosts. The rule can be applied to VMs, or a VM and a host. The rule either keeps the VMs and hosts together (affinity) or separated (anti-affinity).

ESC 2.1 supports only affinity rules in the VMware environment.

Policies are applied during individual VM deployment. Policies such as affinity and anti-affinity streamline the deployment process. During a composite VNF deployment, rules defined in these policies enable easy deployment of individual VMs reducing the deployment time and cost.

Affinity and anti-affinity rules are created and applied on VMs at the time of deployment. VM receives the placement policies when the deploy workflow is initialized.

During a composite VNF deployment, if a couple of VMs need to communicate with each other constantly, they can be grouped together (affinity rule) and placed on the same host.

If two VMs are over-loading a network, they can be separated (anti-affinity rule) and placed on different hosts to balance the network.

Grouping or separating VMs and hosts at the time of deployment helps ESC to manage load across the VMs and hosts in the network. Recovery and scale-up of these VMs do not impact the affinity and anti-affinity rules.

The anti-affinity rule can also be applied between VMs within the same group and on a different host. These VMs perform similar functions and support each other. When one host is down, VM on the other host continues to run preventing any loss of service.

**Affinity and Anti-Affinity Rules in the OpenStack Environment**

The following sections describe affinity and anti-affinity policies with examples.

**Intra Group Affinity Policy**

The VNFs within the same VM group can either be deployed on the same host, or into the same availability zone.

Example for Intra Group Affinity Policy:

```xml
<vm_group>
  <name>affinity-test-gp</name>
  <placement>
```

Cisco Elastic Services Controller 2.1 User Guide
Starting from ESC Release 2.0, the type zone-host is used to deploy VNFs in the same host, or into the same availability zone.

**Zone or Host Based Placement**

Starting from ESC Release 2.0, The VNFs are within the same VM group and deployed on the same host or the same available zone. The host tag is used to deploy VMs on the same host and the zone tag is used to deploy VMs in the same available zone. Before deploying, you need to make sure that the host exists in OpenStack. ESC validates the specified host in OpenStack. The zone-host tag specifies the type of placement. Hence, if a host or zone not specified during deployment, the deployment fails.

### Important
You cannot specify both the host and zone tag to deploy VM on the same host or the same available zone.

Example for host placement:

```xml
<vm_group>
  <name>zone-host-test-gp1</name>
  <placement>
    <type>zone_host</type>
    <enforcement>strict</enforcement>
    <host>my-ucs-4</host>
  </placement>
</vm_group>
```

Example for zone placement:

```xml
<vm_group>
  <name>zone-host-test-gp2</name>
  <placement>
    <type>zone_host</type>
    <enforcement>strict</enforcement>
    <zone>dt-zone</zone>
  </placement>
</vm_group>
```

### Intra Group Anti-Affinity Policy

The VNFs within the same VM group are explicitly deployed on different hosts. For example, back-up VNFs.

Example for Intra Group Anti-Affinity Policy:

```xml
<vm_group>
  <name>anti-affinity-test-gp</name>
  <placement>
    <type>anti_affinity</type>
    <enforcement>strict</enforcement>
  </placement>
</vm_group>
```

### Inter Group Affinity Policy

The VNFs in the same deployment but different VM groups can be explicitly deployed in the same host. For example VNF bundles. Multiple VM groups can follow this policy by adding the vm_group_ref tag and providing the VM group name as the value.
You can only use one `vm_group_ref` tag, type tag and enforcement tag under the placement tag. The host or zone cannot be specified.

Example for Inter Group Affinity Policy:
```xml
<services>
  <service_definition>
    <name>test-strict-affinity-2groups</name>
    <version>1.4</version>
    <policies>
      <placement>
        <target_vm_group_ref>affinity-test-gp1</target_vm_group_ref>
        <type>affinity</type>
        <vm_group_ref>affinity-test-gp2</vm_group_ref>
        <enforcement>strict</enforcement>
      </placement>
    </policies>
  </service_definition>
</services>
```

**Inter Group Anti-Affinity Policy**

The VNFs in the same deployment but different VM Groups can be explicitly deployed in different hosts. For example back-up VNFs or High-availability VNFs. Multiple VM groups can follow this policy by adding the `vm_group_ref` tag and providing the VM group name as the value.

You can only use one `target_vm_group_ref` tag, type tag and enforcement tag under the placement tag. The host or zone cannot be specified.

You can use multiple `vm_group_ref` tags, however the anti-affinity policy only applies between each `vm_group_ref` and their `target_vm_group_ref`, which means that 2 or more `vm_group_ref` can be deployed on the same host, as long as each of them are deployed on a different host from their `target_vm_group_ref` that is acceptable.

Example for Inter Group Anti-Affinity Policy:
```xml
<service_definition>
  <name>test-strict-affinity-2groups</name>
  <version>1.4</version>
  <policies>
    <placement>
      <target_vm_group_ref>affinity-test-gp1</target_vm_group_ref>
      <type>anti_affinity</type>
      <vm_group_ref>affinity-test-gp2</vm_group_ref>
      <enforcement>strict</enforcement>
    </placement>
  </policies>
</service_definition>
```

**Affinity Rules in the VMWare Environment**

Affinity rules are created as host-affinity and during deployment ESC deploys the first VM as an anchor VM for the affinity. All other VMs that follow the same affinity rule will be deployed to the same host as the anchor VM. The anchor VM deployment helps to optimize the resource usage.
Intra Group Affinity Policy

The VNFs within the same VM group can either be deployed on the same host, or into the same availability zone.

Example for Intra Group Affinity Policy:

```
<vm_group>
  <name>affinity-test-gp</name>
  <placement>
    <type>affinity</type>
    <enforcement>strict</enforcement>
  </placement>
...
```

Starting from ESC Release 2.0, the type `zone-host` is used to deploy VNFs in the same host, or into the same availability zone.

Cluster Placement

All VMs in a VM group can be deployed to a cluster. For example, all VMs in the vm group CSR-gp1 can be deployed to cluster ott-cluster2.

```
<name>CSR-gp1</name>
  <placement>
    <type>zone_host</type>
    <enforcement>strict</enforcement>
    <zone>ott-cluster2</zone>
  </placement>
```

Host Placement

All VMS in a VM group can be deployed to a host. For example, all VMs in the vm group CSR-gp1 will be deployed to host 10.2.0.2.

```
<name>CSR-gp1</name>
  <placement>
    <type>zone_host</type>
    <enforcement>strict</enforcement>
    <host>10.2.0.2</host>
  </placement>
```

Inter Group Affinity Policy

The VMs in different VM groups can be deployed to the same host. For example, all VMs in the VM group ASA-gp1 can be deployed to the same host as the VMs in the VM group CSR-gp1.

```
<deployment>
  <deployment>
```

Note

The Vmware cluster must be created by the administrator.

```
<name>CSR-gp1</name>
  <placement>
    <type>zone_host</type>
    <enforcement>strict</enforcement>
    <zone>ott-cluster2</zone>
  </placement>
```

To ensure that the inter-group affinity rules are applied within a single cluster, verify that all VM groups in a deployment is specified to the same cluster (`<zone>` in esc data_model).

```
<deployment>
  <deployment>
```
Limitations

Following are the limitations when affinity rules are applied in VMware environment:

- All Affinity rules defined in the Vmware will be implemented in a cluster.
- DPM and HA must be turned off.
- VM deployment and recovery are managed by ESC.
- DRS must be set to manual mode if it is turned on.
- DPM and Vmotion should be unchecked.
- Supported value for <enforcement> tag should be 'strict'.
- <zone_host> must be used for any vm group being deployed as part of an Intragroup affinity or Intergroup affinity placement.

Advanced Interface Configurations

This section describes several interface configurations for Elastic Services Controller (ESC) and the procedure to configure the hardware interfaces.

Configuring Single Root I/O Virtualization in ESC

Prerequisites

Before you configure Single Root I/O Virtualization (SR-IOV) in ESC, we highly recommend that you configure the hardware and OpenStack with the correct parameters.

Configuring SR-IOV in ESC

SR-IOV allows multiple VMs running a variety of guest operating systems to share a single PCIe network adapter within a host server. It also allows a VM to move data directly to and from the network adapter, bypassing the hypervisor for increased network throughput and lower server CPU burden.

The ESC deployment datamodel allows you to configure SR-IOV by using the interface type as direct. The following example explains how to configure SR-IOV:

```
<interfaces>
  <interface>
    <nicid>0</nicid>
    <network>esc-net</network>
    <type>direct</type>
  </interface>
</interfaces>
```
Configuring Allowed Address Pair

Cisco Elastic Services Controller allows you to specify the address pairs in the deployment datamodel to pass through a specified port regardless of the subnet associated with the network.

The address pair is configured in the following ways:

- List of Network—When a list of network is provided on a particular interface, ESC will get the subnet details from the OpenStack for these networks and add them to the corresponding port or interface. The following example explains how to configure address pairs as a list of network:

```xml
<interface>
  <nicid>1</nicid>
  <network>network1</network>
  <allowed_address_pairs>
    <network>
      <name>bb8c5cfb-921c-46ea-a95d-59feda61cac1</name>
    </network>
    <network>
      <name>6ae017d0-50c3-4225-be10-30e4e5c5e8e3</name>
    </network>
  </allowed_address_pairs>
</interface>
```

- List of Address—When a list of address is provided, ESC will add these address to the corresponding interface. The following example explains how to configure address pairs as a list of address:

```xml
<interface>
  <nicid>0</nicid>
  <network>esc-net</network>
  <allowed_address_pairs>
    <address>
      <ip_address>10.10.10.10</ip_address>
      <netmask>255.255.255.0</netmask>
    </address>
    <address>
      <ip_address>10.10.20.10</ip_address>
      <netmask>255.255.255.0</netmask>
    </address>
  </allowed_address_pairs>
</interface>
```

Configuring Security Group Rules

Cisco Elastic Services Controller (ESC) allows you to associate security group rules to the deployed instances in OpenStack. These security group rules are configured by specifying the necessary parameters in the deployment datamodel. In addition to configuring security group rules, if any VNF instance fails, ESC recovers the instance and applies the security group rules for the redeployed VNF.

To configure security group rules, do the following:

**Before You Begin**

- Make sure you have created a tenant through ESC.
- Make sure you have security groups created.
- Make sure you have the list of UUIDs for the security groups.
Procedure

Step 1  Log in to the ESC VM as a root user.
Step 2  Run the following command to check the UUIDs of a given security group:
        nova --os-tenant-name <NameOfTheTenant> secgroup-list
Step 3  Pass the following arguments in the deployment datamodel:
        <interfaces>
          <interface>
            <nicid>0</nicid>
            <network>esc-net</network><!-- depends on network name -->
            <security_groups>
              <security_group>0c703474-2692-4e84-94b9-c29e439848b8</security_group>
              <security_group>bbcdbc62-a0de-4475-b258-740bdf33861b</security_group>
            </security_groups>
          </interface>
          <interface>
            <nicid>1</nicid>
            <network>Leke_VmGrpNet</network><!-- depends on network name -->
            <security_groups>
              <security_group>b8b9c4ac-48cb-4eff-adaf-c82c14c5feaa</security_group>
            </security_groups>
          </interface>
        </interfaces>

Step 4  Run the following command to verify whether the security groups are associated with the VM instance:
        nova --os-tenant-name <NameOfTenant> show <NameOfVMinstance>

Managing Individual and Composite VNFs

An individual service consists of a single VNF. A coupled service or a composite VNF consists of several VMs of different types. The ESC interface receives VM interdependency information from the northbound system, and uses this information during VM and VNF creation, and life cycle management. Interdependency could include bootup sequence, VM specific workflow in the group of VMs in a single VNF, VNF monitoring and scalability and so on.

Create, read, update and delete operations are allowed on the VMs. To add more VM instances to a deployed VNF using static IP, you must provide additional IP address into the static IP pool. If you are using an existing static IP deployment, the minimum number of VMs is altered.

If the new minimum value, which is the number of VMs is greater than the active VMs, a new VM is added to the service. If the value is greater than the max value, the update is rejected.
Hardware Acceleration Support (OpenStack Only)

Starting from ESC Release 2.1 and later, you can configure hardware acceleration features in OpenStack using the `flavor datamodel`. The following hardware acceleration features can be configured:

- **vCPU Pinning**—enables the binding and unbinding of a process to a vCPU (Virtual Central Processing Unit) or a range of CPUs, so that the process executes only on the designated CPU or CPUs rather than any CPU.

- **Openstack Kilo performance optimization for Large Pages and NUMA**—enables improvement of system performance for large pages and NUMA i.e., system's ability to accept higher load and modifying the system to handle a higher load.

- **Openstack Kilo support for PCIe Passthrough interface**—enables assigning a PCI device to an instance in OpenStack.

The following example explains how to configure hardware acceleration features using `flavor datamodel`:

```xml
$ cat fl.xml
<?xml version='1.0' encoding='ASCII'?>
<esc_datamodel xmlns="http://www.cisco.com/esc/esc">
  <flavors>
    <flavor>
      <name>testfl6</name>
      <vcpus>1</vcpus>
      <memory_mb>2048</memory_mb>
      <root_disk_mb>10240</root_disk_mb>
      <ephemeral_disk_mb>0</ephemeral_disk_mb>
      <swap_disk_mb>0</swap_disk_mb>
      <properties>
        <property>
          <name>pci_passthrough:alias</name>
          <value>nic1g:1</value>
        </property>
      </properties>
    </flavor>
  </flavors>
</esc_datamodel>
$ sudo /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli edit-config ./fl.xml
```

Undeploying Virtual Network Functions

You can undeploy an already deployed VNF. Use the REST or NetConf / Yang APIs to un-deploy the VNF.

**Important**

You can also undeploy VNFs using the ESC portal. For more information see Cisco Elastic Services Controller Portal.

Sample undeploy request

```
DELETE /v0/deployments/567 HTTP/1.1
Host: client.host.com
Content-Type: application/xml
Accept: application/xml
Client-Transaction-Id: 123456
Callback:/undeployservicecallback
```

For more details, see Cisco Elastic Services Controller API Guide.
Updating Deployments

Starting from Cisco ESC Release 1.1, you can add or delete a vm_group, add or delete a ephemeral network in a vm_group, and add or delete an interface in a vm_group during deployment.

On OpenStack environment you can perform all the updates such as add or delete a vm_group, add or delete a ephemeral network in a vm_group, and add or delete an interface in a vm_group) in a single deployment.

Adding a VM Group

You can add or delete a vm_group from a running deployment using existing images and flavors.

NETCONF request to add a vm_group:

```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc"> 
<tenants>
<tenant>
    <name>Admin</name>
    <deployments>
        <deployment>
            <deployment_name>NwDepModel_nosvc</deployment_name>
            <vm_group>
                <image></image>
                <Flavor></Flavor>
            </vm_group>
            <vm_group>
                <image></image>
                <Flavor></Flavor>
            </vm_group>
            <vm_group>
                <image></image>
                <Flavor></Flavor>
            </vm_group>
        </deployment>
    </deployments>
</tenant>
</tenants>
</esc_datamodel>
```

NETCONF notification upon successful addition of VM Group:

```
UPDATE SERVICE REQUEST RECEIVED (UNDER TENANT)
VM_DEPLOYED
VM_ALIVE
SERVICE_UPDATED
UPDATE SERVICE REQUEST RECEIVED (UNDER TENANT)
```

Deleting a VM Group

NETCONF request to delete a vm_group:

```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc"> 
<tenants>
<tenant>
    <name>Admin</name>
    <deployments>
        <deployment>
            <deployment_name>NwDepModel_NoSvc</deployment_name>
            <vm_group>
                <image></image>
                <Flavor></Flavor>
            </vm_group>
        </deployment>
    </deployments>
</tenant>
</tenants>
</esc_datamodel>
```
Adding an Ephemeral Network in a VM Group

You can add an ephemeral network in a vm_group using existing images and flavors.

NETCONF request to add an ephemeral in a vm_group:
```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc"> <tenants><tenant>
  <name>Admin</name>
  <deployments>
    <deployment>
      <deployment_name>NwDepModel_nosvc</deployment_name>
      <networks>
        <network>
          ........
        </network>
        <network>
          ........
        </network>
        <network>
          ........
        </network>
      </networks>
      <vm_group>
        <image></image>
        <Flavor></Flavor>
        ........
      </vm_group>
    </deployment>
  </deployments>
  </tenant></tenants>
</esc_datamodel>
```

NETCONF notification upon successful addition of an ephemeral network in a vm_group:
UPDATE SERVICE REQUEST RECEIVED (UNDER TENANT)
  CREATE_NETWORK
  CREATE_SUBNET
  SERVICE_UPDATED
UPDATE SERVICE REQUEST RECEIVED (UNDER TENANT)

Deleting an Ephemeral Network in a VM Group

NETCONF request to delete an ephemeral network in a vm_group
```
<esc_datamodel xmlns="http://www.cisco.com/esc/esc"> <tenants><tenant>
  <name>Admin</name>
  <deployments>
    <deployment>
      <deployment_name>NwDepModel</deployment_name>
      <networks>
        <network nc:operation="delete">
          ........
        </network>
        <network>
          ......
        </network>
        <network>
          ......
        </network>
      </networks>
      <vm_group>
        <image></image>
        <Flavor></Flavor>
        ......
      </vm_group>
    </deployment>
  </deployments>
  </tenant></tenants>
</esc_datamodel>
```

NETCONF notification upon successful deletion of vm_group:
UPDATE SERVICE REQUEST RECEIVED (UNDER TENANT)
  VM UNDEPLOYED
  SERVICE_UPDATED
UPDATE SERVICE REQUEST RECEIVED (UNDER TENANT)
Deploying Virtual Network Functions

Adding an Interface in a VM Group

You can add an interface in a vm_group from a running deployment using existing images and flavors.

NETCONF request to add an interface in a vm_group:

```xml
<interfaces>
  <interface>
    <nicid>0</nicid>
    <network>esc-net</network>
  </interface>
  <interface>
    <nicid>1</nicid>
    <network>utroycho-net</network>
  </interface>
  <interface>
    <nicid>2</nicid>
    <network>utroycho-net-1</network>
  </interface>
</interfaces>
```

Deleting an Interface in a VM Group

NETCONF request to delete an interface in a vm_group:

```xml
<interfaces>
  <interface>
    <nicid>0</nicid>
    <network>esc-net</network>
  </interface>
  <interface>
    <nicid>1</nicid>
    <network>utroycho-net</network>
  </interface>
  <interface nc:operation="delete">
    <nicid>2</nicid>
    <network>utroycho-net-1</network>
  </interface>
</interfaces>
```