VIM Connector Configurations

VIM Connector Configurations for OpenStack

You can configure the VIM connector for OpenStack specific operations.

Note To configure a VIM connector, seeConfiguring the VIM Connector .

Creating Non-admin Roles for ESC Users in OpenStack

By default, OpenStack assigns an admin role to the ESC user. Some policies may restrict using the default admin role for certain ESC operations. Starting from ESC Release 3.1, you can create non-admin roles with limited permissions for ESC users in OpenStack.

To create a non-admin role,

1. Create a non-admin role in OpenStack.

2. Assign the non-admin role to the ESC user.

   You must assign ESC user roles in OpenStack Horizon (Identity) or using the OpenStack command line interface. For more details see, OpenStack Documentation.

   The role name can be customized in OpenStack. By default, all non-admin roles in OpenStack have the same level of permissions.

3. Grant the required permissions to the non-admin role.

   You must modify the `policy.json` file to provide the necessary permissions.

Note You must grant permissions to the *create_port: fixed_ips* and *create_port: mac_address* parameters in the policy.json file for ESC user role to be operational.

The table below lists the ESC operations that can be performed by the non-admin role after receiving the necessary permissions.

Table 1: Non-admin role permissions for ESC operations

ESC VIM Operation	Description	Permission	Note
Create Project	To create an OpenStack project	`/etc/keystone/policy.json` `"identity:create_project"` `"identity:create_grant"`	For ESC managed OpenStack project, adding the user to the project with a role requires *identity:create_grant*.
Delete Project	To delete an OpenStack project	`/etc/keystone/policy.json` `"identity:delete_project"`	
Query Image	To get a list of all images	Not required	The owner (a user in the target project) can query. You can retrieve public or shared images.
Create Image	To create a public image	`/etc/glance/policy.json` `"publicize_image"`	By default an admin can create a public image. Publicizing an image is protected by the policy.

	To create a private image	Not required	You can use the following to create a private image
			`<image>   <name>mk-test-image</name>`
			`...    <disk_bus>virtio</disk_bus>`
			`<visibility>private</visibility>`
			`</image>`

Delete Image	To delete an image	Not required	The owner can delete the image.
Query Flavor	To query a pre-existing flavor	Not required	The owner can query a flavor. You can query public flavors as well.
Create Flavor	To create a new flavor	`/etc/nova/policy.json` `"os_compute_api:os-flavor-manage"`	Managing a flavor is typically only available to administrators of a cloud.
Delete Flavor	To delete a flavor	`/etc/nova/policy.json` `"os_compute_api:os-flavor-manage"`	
Query Network	To get a list of networks	`/etc/neutron/policy.json` `"get_network"`	Owner can get the list of networks including shared networks.

ESC VIM Operation	Description	Permission	Note
Create Network	To create a normal network	Not required	
	To create network with special cases	`/etc/neutron/policy.json` "create_network:provider:physical_network" "create_network:provider:network_type" "create_network:provider:segmentation_id" "create_network:shared"	You need these rules when you are creating network with physical_network (e.g., SR-IOV), or network_type (e.g., SR-IOV), or segmentation_id (e.g., 3008), or set the network for sharing. `<network> <name>provider-network</name> <!-- <shared>false</shared> //default is true --> <admin_state>true</admin_state> <provider_physical_network>VAR_PHYSICAL_NET </provider_physical_network> <provider_network_type>vlan </provider_network_type> <provider_segmentation_id>2330 </provider_segmentation_id> ... </network>`
Delete Network	To delete a network	Not required	The owner can delete the network.
Query Subnet	To get a list of subnets	`/etc/neutron/policy.json` "get_subnet"	The network owner can get a list of the subnets. You can get a list of subnets from a shared network as well. `<network> <name>esc-created-network</name> <!--network must be created by ESC--> <admin_state>false</admin_state> <subnet> <name>makulandyescextnet1-subnet1</name> <ipversion>ipv4</ipversion> <dhcp>true</dhcp> <address>10.6.0.0</address> <netmask>255.255.0.0</netmask> </subnet> </network>`
Create Subnet	To create a subnet	Not required	The network owner can create a subnet.
Delete Subnet	To delete a subnet	Not required	The network owner can delete a subnet.

ESC VIM Operation	Description	Permission	Note
Query Port	Get a pre-existing port	Not required	The owner can get a list of ports.
Create Port	To create a network interface with DHCP	Not required	
	Create a network interface with a mac address	`/etc/neutron/policy.json` `"create_port:mac_address"`	`<interfaces>` `<interface>` `<nicid>0</nicid>` `<mac_address>fa:16:3e:73:19:b5</mac_address>` `<network>esc-net</network>` `</interface> </interfaces>`VM recovery also requires this privilege.
	To create a network interface with a fixed IP or shared ips	`/etc/neutron/policy.json` `"create_port:fixed_ips"`	`<subnet> <name>IP-pool-subnet</name>` `<ipversion>ipv4</ipversion>` `<dhcp>false</dhcp>` `<address>172.16.0.0</address>` `<netmask>255.255.255.0</netmask>` `<gateway>172.16.0.1</gateway>` `</subnet><shared_ip> <nicid>0</nicid>` `<static>false</static> </shared_ip>` VM recovery also requires this privilege.
Update Port	Update port device owner	Not required	The owner can update the port.
	Update port to allow address pairs	`/etc/neutron/policy.json` `"update_port:allowed_address_pairs"`	`<interface>` `<nicid>0</nicid>` `<network>VAR_MANAGEMENT_NETWORK_ID</network>` `<allowed_address_pairs> <network>` `<name>VAR_MANAGEMENT_NETWORK_ID</name>` `</network> <address>` `<ip_address>172.16.0.0</ip_address>` `<netmask>255.255.0.0</netmask>` `</address> <address>` `<ip_address>172.16.6.1</ip_address>` `<ip_prefix>24</ip_prefix> </address>` `</allowed_address_pairs> </interface>`
Delete Port	To delete a port	Not required	The owner can delete the port.

ESC VIM Operation	Description	Permission	Note
Query Volume	To get a list of volumes	Not required	The owner can get the list of volumes.
Create Volume	To create a volume	Not required	
Delete Volume	To delete a volume	Not required	The owner can delete the volume.
Query VM	To get all the VMs in a project	Not required	The owner can get the list of all the VMs in a project.
Create VM	To create a VM	Not required	
	To create a VM in a host targeted deployment	/etc/nova/policy.json "os_compute_api:servers:create:forced_host"	`<placement>   <type>zone_host</type> <enforcement>strict</enforcement> <host>anyHOST</host> </placement>`
	To create VMs in a zone targeted deployment	Not required	
	To create VMs in the same Host Affinity/Anti-affinity	Not required	
	To create VMs in a servergroup Affinity/Anti-affinity	Not required	This support is for intragroup anti-affinity only.
Delete VM	To delete a VM	Not required	The owner can delete the VM.

For more details on managing resources on OpenStack, see Managing Resources on OpenStack.

Overwriting OpenStack Endpoints

By default, ESC uses endpoints catalog return option provided by OpenStack after a successful authentication. ESC uses these endpoints to communicate with different APIs in OpenStack. Sometimes the endpoints are not configured correctly, for example, the OpenStack instance is configured to use KeyStone V3 for authentication, but the endpoint returned from OpenStack is for KeyStone V2. You can overcome this by overwriting the OpenStack endpoints.

You can overwrite (configure) the OpenStack endpoints while configuring the VIM connector. This can be done at the time of installation using the bootvm.py parameters, and using the VIM connector APIs.

The following OpenStack endpoints can be configured using the VIM connector configuration:

- OS_IDENTITY_OVERWRITE_ENDPOINT

- OS_COMPUTE_OVERWRITE_ENDPOINT

- OS_NETWORK_OVERWRITE_ENDPOINT

- OS_IMAGE_OVERWRITE_ENDPOINT

- OS_VOLUME_OVERWRITE_ENDPOINT

To overwrite OpenStack endpoints at the time of installation, a user can create an esc configuration parameters file, and pass the file as an argument to bootvm.py while deploying an ESC VM.

Below is an example of the param.conf file:

```
openstack.os_identity_overwrite_endpoint=http://www.xxxxxxxxxx.com
```

For more information on configuring the VIM connector at the time of Installation, see Configuring the VIM Connector.

To overwrite (configure) the OpenStack endpoints for a non-default VIM connector using the VIM connector APIs (both REST and NETCONF), add the overwriting endpoints as the VIM connector properties either while creating a new VIM connector or updating an existing one.

Each VIM connector can have its own overwriting endpoints. There is no default overwriting endpoint.

In the example below, *os_identity_overwrite_endpoint* and *os_network_overwrite_endpoint* properties are added to overwrite the endpoints.

```xml
<esc_system_config xmlns="http://www.cisco.com/esc/esc">
  <vim_connectors>
    <!--represents a vim-->
    <vim_connector>
      <id>default_openstack_vim</id>
      <type>OPENSTACK</type>
      <properties>
        <property>
          <name>os_auth_url</name>
          <value>http://172.16.0.0:35357/v3</value>
        </property>
        <property>
          <name>os_project_domain_name</name>
          <value>default</value>
        </property>
        <property>
          <name>os_project_name</name>
          <value>admin</value>
        </property>
        <property>
```

```
            <name>os_identity_overwrite_endpoint</name>
            <value>http://some_server:some_port/</value>
        </property>
        <property>
            <name>os_network_overwrite_endpoint</name>
            <value>http://some_other_server:some_other_port/</value>
        </property>
    </properties>
  </vim_connector>
 </vim_connectors>
</esc_system_config>
```

VIM Connector Configurations for AWS

You can set the VIM credentials for an AWS deployment using the VIM connector and VIM User API.

Note AWS deployment does not support default VIM connector.

The VIM connector **aws_default_region** value provides authentication, and updates the VIM status. The default region cannot be changed after authentication.

Configuring the VIM Connector

To configure the VIM connector for AWS deployment, provide the AWS_ACCESS_ID, AWS_SECRET_KEY from your AWS credentials.

```
[admin@localhost ~]# esc_nc_cli edit-config
aws-vim-connector-example.xml
```

Note To edit the existing VIM connector configuration, use the same command after making the necessary changes.

The AWS VIM connector example is as follows:

```
<esc_system_config xmlns="http://www.cisco.com/esc/esc">
    <vim_connectors>
        <vim_connector>
            <id>AWS_EAST_2</id>
            <type>AWS_EC2</type>
            <properties>
                <property>
                    <name>aws_default_region</name>
                    <value>us-east-2</value>
                </property>
            </properties>
            <users>
                <user>
                    <id>AWS_ACCESS_ID</id>
                    <credentials>
                        <properties>
                            <property>
                                <name>aws_secret_key</name>
                                <encrypted_value>AWS_SERCRET_KEY</encrypted_value>
                            </property>
                        </properties>
```

```
                </credentials>
            </user>
        </users>
    </vim_connector>
  </vim_connectors>
</esc_system_config>
```

Deleting VIM Connector

To delete the existing VIM connector, you must first delete the deployment, the VIM user, and then the VIM connector.

```
[admin@localhost ~]# esc_nc_cli delete-vimuser
AWS_EAST_2 AWS_ACCESS_ID

[admin@localhost ~]# esc_nc_cli delete-vimconnector
AWS_EAST_2
```

Note   You can configure multiple VIM connectors, but for the same VIM type.

The VIM connectors for AWS deployment must be configured using the VIM connector API.

ESC supports one VIM user per VIM connector.

The VIM connector and its properties cannot be updated after deployment.

For information on deploying VNFs on AWS, seeDeploying VNFs on a Single or Multiple AWS Regions .

VIM Connector Configuration for VMware vCloud Director (vCD)

You must configure a VIM connector to connect to the vCD organization. The organization and the organization user must be preconfigured in the VMware vCD. For the deployment datamodel, see the Deploying Virtual Network Functions on VMware vCloud Director (vCD).

The VIM connector details are as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<esc_system_config xmlns="http://www.cisco.com/esc/esc">
   <vim_connectors>
      <vim_connector>
         <id>vcd_vim</id>
         <type>VMWARE_VCD</type>
         <properties>
            <property>
               <name>authUrl</name>
                <!-- vCD is the vCD server IP or host name -->
               <value>https://vCD</value>
            </property>
         </properties>
         <users>
            <user>
              <!-- the user id here represents {org username}@{org name} -->
              <id>user@organization</id>
              <credentials>
                  <properties>
```

```
<property>
    <name>password</name>
        <!—the organization user's password-->
    <value>put user's password here</value>
</property>
</properties>
</credentials>
</user>
</users>
</vim_connector>
</vim_connectors>
</esc_system_config>
```