ESC REST API

ESC REST Northbound API Documentation

Table of Contents

Resources managed by ESC REST

- Tenants
- Networks
- Subnets
- Flavors
- Images
- Volumes
- Deployments

Behavior of ESC REST operations

GET operations are synchronous the request/response parameters are described with an example of a request/response

POST, DELETE, and PUT operations are asynchronous. The asynchronous design use the “web hooks” approach using two independent "one-way" invocations - one to start a long-running operation (Client to ESC) and the other one to notify a requester that it is completed (ESC to client)

ESC REST API Headers/Path/Body Parameters

Callbacks

One of the header parameters of the operation request will contain a callback field, whose value is a URI where the client expects a callback. If this field is not present, no callback will be performed.

When the operation has completed (either successfully or unsuccessfully), ESC will perform a POST request to the callback URI, with an entity body containing the resource for this operation. The status will be returned in the HTTP header called esc-status-code and the status message will be returned in the HTTP header called esc-status-message.

ESC Clients can match a completion report back to the original request by comparing the value of the HTTP header called esc-transaction-id to the one returned in the initial status response.

ESC Clients are expected to provide a REST implementation for all the callbacks.

Media Type

All POST, DELETE, and PUT operations that require request and/or response body must provide a header parameter for media type. Note, that ESC is currently tested to accept and return XML.

Internal IDs

In some of the ESC REST API, there is a path parameter called "*_internal_id" where the asterisk is replaced by the resource type. ESC Clients must provide an internal id for its resource it wants to create. The internal ids are enforced so the ESC Clients are able to refer to them in future operations such as GET, PUT, or DELETE requests.

It is recommended that the ESC Client generated internal ids within the scope of the ESC.

NOTE: Reusing (recycling) of internal ids is not recommended. This practice can lead to confusion in trouble shooting.

Body

In some of the POST, PUT, and DELETE requests you are required to provide a JSON/XML body. Depending on the resource, you will need to provide certain fields in the body to successfully execute a request. Check the body section for each API (bottom of page) to see what fields are supported.

String Restrictions in Body/Internal IDs

Body

Some string parameters/fields, such as id, name, are restricted by the underlying cloud provider, such as Openstack. The allowable range for these parameters is included in the Create section for each resource, in this notation: String minimum-length . . maximum length, for example, for Tenant name: String 1 . . 64

Generally, names are either 1..64 (for Tenant/User) or 1..255 (for Network/Subnet/Flavor/Image/VM).

Internal ID

Ids are restricted to 72 characters such as internal_id 1..72 and external_id 1..72.

Internal id naming must conform the standards stated in RFC 3986 Uniform Resource Identifier (URI): Generic Syntax. See sections related to URI encoding, sections 2.2 Reserved Characters and 2.3 Unreserved Characters. ESC will accept an internal id specified in the request URL form the unreserved character set: unreserved = ALPHA / DIGIT / "-" / "." / "_"

NOTE: the tilde “~” is not supported in the internal id character set.

The internal id received in a request must be unique within the scope of all the domain resources (tenant, network, subnet, service catalog, deployment) For example, you cannot create 2 networks with the same internal_network_id, even if the networks are created under different tenants.

REST API Security

REST Authentication

Overview

ESC REST API uses http basic access authentication where the ESC client will have to provide a username and password when making ESC REST requests. The user name and password will be encoded with Base64 in transit, but not encrypted or hashed. HTTPS will be used in conjunction with Basic Authentication to provide the encryption.

Setup

By default, the REST authentication is disabled. To enable it the pass the argument --enable-auth to ESC bootvm.py, ESC installation script.

Username and Password

The REST interface has only one default username/password (admin credentials).

The REST password can be updated using escadm tool from the ESC VM CLI:

```
$ escadm rest set --username admin --password test123
```

The REST password can be reset using escadm tool from the ESC VM CLI:

```
$ escadm rest set
```

You can also update the password through the REST API:

http://[ESCVM_IP]:8080/ESCManager/v0/authentication/setpassword?userName=yourUsername&password=yourPassword

Sending an Authorized Request

To send an authorized request an ESC client should send the request with the following header:

```
Authorization: Basic <hashed-pass>
```

where is the Base64 enconded string of the default username/password

Most libraries and web clients have an interface for simply providing the username/password and the app will encode the username/password and add the HTTP Basic Auth header.

Example using the default admin credentials:

```
http://[ESCVM_IP]:8080/ESCManager/v0/tenants/

Headers:
Content-Type    application/xml
Authorization    Basic <hashed-pass>
```

REST Https Support

Overview

ESC supports https communication over port 8443. ESC will generate a self-signed certificate that the client will need to trust to get the https communication going.

Setup

By default, the REST https is disabled and restricted to localhost. To enable it pass the argument --enable-https-rest to ESC bootvm.py, ESC installation script.

HA Considerations

In an HA setup every ESC HA VM will have its own self-signed generated certificate. Upon an HA switchover, ESC north bound client will be required to re-accept the new certificate it will be served.

REST API List

01-Tenants : Manage ESC tenants

GET /v0/tenants — Get all tenants

Implementation Notes

All tenants in ESC will be returned as a List

Response Class (Status 200)

Model | Model Schema


```
[
  {
    "internal_tenant_id": "string",
    "external_tenant_id": "string",
    "name": "string",
    "enabled": true,
    "vim_mapping": true,
    "extensions": {
      "extension": [
        {
          "name": "string",
          "properties": {
            "property": [
              {
                "name": "string",
                "value": "string"
              }
            ]
          },
          "containers": {
            "container": [
              {
                "name": "string",
                "properties": {
                  "property": [
                    {
                      "name": "string",
                      "value": "string"
                    }
                  ]
                }
              }
            ]
          }
        }
      ]
```

```
    },
    "event_type": "CREATE_TENANT",
    "managed_resource": true
  }
]
```


Response Content Type application/xml

Try it out!

DELETE /v0/tenants/{internal_tenant_id} Delete a tenant from ESC

Implementation Notes

Delete a tenant from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internal_tenant_id	**(required)**	**Internal tenant ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Tenant deletion request accepted		
400	Invalid Request		
404	Invalid Tenant Id		

Try it out!

GET /v0/tenants/{internal_tenant_id} Get a tenant by its name

Implementation Notes

Only tenants created using ESC will be checked

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internal_tenant_id	**(required)**	**Internal tenant ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers

HTTP Status Code	Reason	Response Model	Headers
200	Get request accepted		
404	Invalid Request - Invalid Tenant ID		

Try it out!

POST /v0/tenants/{internal_tenant_id} Create a tenant in ESC

Implementation Notes

Creates a tenant which will be managed by ESC and can be used for creating other resources under it.Example request:

```
POST /v0/tenants/tentest HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Callback: http://127.0.0.1:9010/
Body:
<tenant xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <name>example-tenant</name>
  <managed_resource>true</managed_resource>
</tenant>
```

After the tenant creation operation is done, ECS will send a callback to the client with the result. Example callback:

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: Tenant successfully created
<tenant xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <enabled>true</enabled>
  <event_type>CREATE_TENANT</event_type>
  <external_tenant_id>484c620285ea4f588227ff3396215980</external_tenant_id>
  <internal_tenant_id>tentest</internal_tenant_id>
  <name>example-tenant</name>
  <managed_resource>true</managed_resource>
</tenant>
```

Parameters

Parameter	Value	Description	Parameter Type	Data Type

Parameter	Value	Description	Parameter Type	Data Type
body	**(required)** Parameter content type: application/xml	**Tenant configuration payload**	body	Model | Model Schema


```
{
  "internal_tenant_id": "string",
  "external_tenant_id": "string",
  "name": "string",
  "enabled": true,
  "vim_mapping": true,
  "extensions": {
    "extension": [
      {
        "name": "string",
        "properties": {
          "property": [
            {
              "name": "string",
              "value": "string"
            }
          ]
        },
        "containers": {
          "container": [
            {
              "name": "string",
              "properties": {
                "property": [
                  {
                    "name": "string",
                    "value": "string"
                  }
                ]
              }
            }
          ]
        }
      }
    ]
  },
  "event_type": "CREATE_TENANT",
  "managed_resource": true
}
```


Click to set as parameter value

Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internal_tenant_id	**(required)**	**Internal tenant ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Tenant creation request accepted		
400	Invalid Request		

HTTP Status Code	Reason	Response Model	Headers
409	Conflicting Request		

Try it out!

PUT /v0/tenants/{internal_tenant_id} Update a tenant from ESC

Implementation Notes

Update a tenant from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
body	(required) Parameter content type: application/xml	**Tenant payload**	body	Model | Model Schema


```
{
  "internal_tenant_id": "string",
  "external_tenant_id": "string",
  "name": "string",
  "enabled": true,
  "vim_mapping": true,
  "extensions": {
    "extension": [
      {
        "name": "string",
        "properties": {
          "property": [
            {
              "name": "string",
              "value": "string"
            }
          ]
        },
        "containers": {
          "container": [
            {
              "name": "string",
              "properties": {
                "property": [
                  {
                    "name": "string",
                    "value": "string"
                  }
                ]
              }
            }
          ]
        }
      }
    ]
  },
  "event_type": "CREATE_TENANT",
  "managed_resource": true
}
```


Click to set as parameter value

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internal_tenant_id	**(required)**	**Internal tenant ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Tenant update request accepted		
400	Invalid Request		
404	Invalid Tenant Id		
500	Internal Server Error		

Try it out!

02-Networks : Manage ESC networks

DELETE /v0/{internalTenantId}/networks/{internalNetworkId} Delete a network from an existing tenant in ESC

Implementation Notes

Delete a network from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internalNetworkId	**(required)**	**Internal network ID (max length 72 characters)**	path	string
internalTenantId	**(required)**	**Internal tenant ID of the network (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid request		
500	Internal server error		

Try it out!

GET /v0/{internalTenantId}/networks/{internalNetworkId} Get a network by its tenant and internal network ID

Implementation Notes

Only networks created using ESC will be checked

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalTenantId	**(required)**	**Internal tenant ID of the network (max length 72 characters)**	path	string
internalNetworkId	**(required)**	**Internal network ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Network Id not found		

Try it out!

POST /v0/{internalTenantId}/networks/{internalNetworkId} Create a network under an existing tenant in ESC

Implementation Notes

Create a network in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internalTenantId	**(required)**	**Internal tenant ID of the network (max length 72 characters)**	path	string
internalNetworkId	**(required)**	**Internal network ID (max length 72 characters)**	path	string

Parameter	Value	Description	Parameter Type	Data Type
body	(required) Parameter content type: application/xml	**Network configuration payload**	body	Model | Model Schema


```
{
  "locator": {
    "vim_id": "string",
    "vim_type": "OPENSTACK",
    "vim_project": "string",
    "vim_project_id": "string",
    "vim_region": "string",
    "vim_vdc": "string"
  },
  "vimNetName": "string",
  "internal_tenant_id": "string",
  "internal_network_id": "string",
  "internal_deployment_id": "string",
  "name": "string",
  "shared": true,
  "external_tenant_id": "string",
  "tenant_name": "string",
  "external_network_id": "string",
  "subnet": [
    "string"
  ],
  "admin_state_up": true,
  "locators": {
    "datacenter": "string",
    "switch_name": "string"
  },
  "event_type": "CREATE_NETWORK",
  "provider_segmentation_id": "string",
  "router_external": true,
  "provider_physical_network": "string",
  "provider_network_type": "string",
  "switch_name": "string",
  "vlan_id": 0,
  "number_of_ports": 0
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid request		
404	Network Id not found		
409	Conflicting request		
503	Unable to query VIM		

Try it out!

GET /v0/{internalTenantId}/networks — Get all networks under a tenant

Implementation Notes

All networks under that tenant in ESC will be returned as a List

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalTenantId	**(required)**	**Internal tenant ID of the network (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		

Try it out!

03-Subnets : Manage ESC subnets

GET /v0/{internalTenantId}/subnets — Get all subnets under a network

Implementation Notes

All subnet under that network in ESC will be returned as a List

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalNetworkId		Internal network ID of the subnet (max length 72 characters)	header	string
internalTenantId	**(required)**	**Internal tenant ID of the subnet (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Not found		

Try it out!

DELETE /v0/{internalTenantId}/subnets/{internalSubnetId} — Delete a subnet from an existing tenant in ESC

Implementation Notes

Delete a subnet from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internalTenantId	**(required)**	**Internal tenant ID of the subnet (max length 72 characters)**	path	string
internalSubnetId	**(required)**	**Internal subnet ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	Not found		
500	Internal server error		

Try it out!

GET /v0/{internalTenantId}/subnets/{internalSubnetId} Get a subnet by its tenant and internal subnet ID

Implementation Notes

Only subnets created using ESC will be checked

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalTenantId	**(required)**	**Internal tenant ID of the subnet (max length 72 characters)**	path	string
internalSubnetId	**(required)**	**Internal subnet ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Not found		

Try it out!

POST /v0/{internalTenantId}/subnets/{internalSubnetId} Create a subnet under an existing tenant in ESC

Implementation Notes

Create a subnet in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
internalTenantId	(required)	**Internal tenant ID of the subnet (max length 72 characters)**	path	string
internalSubnetId	(required)	**Internal subnet ID (max length 72 characters)**	path	string
body	(required) Parameter content type: application/xml	**Subnet configuration payload**	body	Model | Model Schema


```
{
  "networkFromCdg": {
    "locator": {
      "vim_id": "string",
      "vim_type": "OPENSTACK",
      "vim_project": "string",
      "vim_project_id": "string",
      "vim_region": "string",
      "vim_vdc": "string"
    },
    "vimNetName": "string",
    "internal_tenant_id": "string",
    "internal_network_id": "string",
    "internal_deployment_id": "string",
    "name": "string",
    "shared": true,
    "external_tenant_id": "string",
    "tenant_name": "string",
    "external_network_id": "string",
    "subnet": [
      "string"
    ],
    "admin_state_up": true,
    "locators": {
      "datacenter": "string",
      "switch_name": "string"
    },
    "event_type": "CREATE_NETWORK",
    "provider_segmentation_id": "string",
    "router_external": true,
    "provider_physical_network": "string",
    "provider_network_type": "string",
    "switch_name": "string",
    "vlan_id": 0,
    "number_of_ports": 0
  },
```

Parameter	Value	Description	Parameter Type	Data Type


```
  "internal_tenant_id": "string",
  "external_tenant_id": "string",
  "internal_network_id": "string",
  "external_network_id": "string",
  "internal_subnet_id": "string",
  "external_subnet_id": "string",
  "name": "string",
  "allocation_pools": [
    {
      "start": "string",
      "end": "string"
    }
  ],
  "gateway_ip": "string",
  "ip_version": 0,
  "cidr": "string",
  "enable_dhcp": true,
  "event_type": "CREATE_SUBNET"
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	Not found		
409	Conflicting Request		
500	Internal server error		

Try it out!

04-Images : Manage ESC images

DELETE /v0/images/{internalImageId} — Delete an image from ESC

Implementation Notes

Delete an image from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internalImageId	**(required)**	**Internal image ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
500	Internal server error		

Try it out!

GET /v0/images/{internalImageId} Get the image by the internal image ID

Implementation Notes

Images not created by ESC will not be checked

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalImageId	**(required)**	**Internal image ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	Image Id not found		

Try it out!

POST /v0/images/{internalImageId} Create an image in ESC

Implementation Notes

Creates an Image(Template) which will be managed by ESC and can be used for creating instance VMs. An optional properties list can be included in the image creation payload. This is only supported for creation of images on openstack. Image properties will be be stored in openstack as image metadata. Property name and values are not validated by ESC.

Example 1 create image request (without properties):

```
POST /v0/images/my_image_id HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Callback: http://127.0.0.1:9010/
Body:

<?xml version="1.0"?>
<image xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <name>uLinuxImg</name>
    <src>http://VAR_FILE_SERVER_IP/share/images/myULinux.qcow2</src>
```

```
    <disk_bus>virtio</disk_bus>
</image>
```

Example 2 create image request with properties:

```
POST /v0/images/my_image_id HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Callback: http://127.0.0.1:9010/
Body:

<?xml version="1.0"?>
<image xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <name>uLinuxImg</name>
    <src>http://VAR_FILE_SERVER_IP/share/images/myULinux.qcow2</src>
    <disk_bus>virtio</disk_bus>
    <properties>
        <property>
           <name>os_secure_boot</name>
           <value>enabled</value>
        </property>
        <property>
           <name>hw_rng_model</name>
           <value>virtio</value>
        </property>
        <property>
           <name>hw_vif_multiqueue_enabled</name>
           <value>true</value>
        </property>
    </properties>
</image>
```

Example 3 create image with properties Use of image properties allow flexible assigment of image metadata, such as hw_disk_bus and hw_vif_model. This extends current limitation of ESC schema with disk_bus := ide | scsi | virtio and e1000_net := true | false and virtio_net : = true | false.

```
POST /v0/images/my_image_id HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Callback: http://127.0.0.1:9010/
Body:

<?xml version="1.0"?>
<image xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <name>uLinuxImg</name>
    <src>http://VAR_FILE_SERVER_IP/share/images/myULinux.qcow2</src>
    <disk_bus>virtio</disk_bus>
    <properties>
        <property>
           <name>hw_disk_bus</name>
           <value>pcnet</value>
        </property>
        <property>
```

```
            <name>hw_vif_model</name>
            <value>pcnet</value>
        </property>
    </properties>
</image>
```

After the Image creation operation is done, ESC will send a callback to the client with the result.

Example callback:

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: Image successfully created


<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<image xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <disk_bus>virtio</disk_bus>
    <e1000_net>false</e1000_net>
    <event_type>CREATE_IMAGE</event_type>
    <external_image_id>5021843b-fcc5-2c1b-8fc4-935147f95872</external_image_id>
    <external_tenant_id>SystemAdminTenantId</external_tenant_id>
    <imageisenabled>true</imageisenabled>
    <name>uLinuxImg</name>
    <internal_image_id>my_image_id</internal_image_id>
    <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
    <visibility>public</visibility>
    <src>http://VAR_FILE_SERVER_IP/share/images/myULinux.qcow2</src>
</image>
```

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
internalImageId	(required)	**Internal image ID (max length 72 characters)**	path	string
body	(required) Parameter content type: application/xml	**ESC datamodel with the image to create**	body	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
409	Conflicting Request		
500	Internal server error		

Try it out!

GET /v0/images — Get all images

Implementation Notes

All images created in ESC will be returned as a List

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		

Try it out!

05-Flavors : Manage ESC flavors

GET /v0/flavors — Get all flavors

Implementation Notes

All flavors created in ESC will be returned as a List

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		

Try it out!

DELETE /v0/flavors/{internalFlavorId} — Delete a flavor from ESC

Implementation Notes

Delete a flavor from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string

Parameter	Value	Description	Parameter Type	Data Type
internalFlavorId	**(required)**	**Internal flavor ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Flavor Id not found		
500	Internal server error		

Try it out!

GET /v0/flavors/{internalFlavorId} Get the flavor by the internal flavor ID

Implementation Notes

Flavors not created by ESC will not be checked

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalFlavorId	**(required)**	**Internal flavor ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Flavor Id not found		
406	Operation not supported on VIM/Flavor part of service registration		

Try it out!

POST /v0/flavors/{internalFlavorId} Create a flavor in ESC

Implementation Notes

Creates a Flavor which could be used for creating instance VMs.

Example request:

```
POST /v0/flavors/my_flavor_id HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Callback: http://127.0.0.1:9010/
Body:
```

```
<?xml version="1.0"?>
<flavor xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
        <name>lightVMFlavor</name>
        <vcpus>2</vcpus>
        <memory_mb>2048</memory_mb>
</flavor>
```

After the Flavor creation operation is done, ESC will send a callback to the client with the result.

Example callback:

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: Flavor creation completed successfully created
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<flavor xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <ephemeral_disk_mb>0</ephemeral_disk_mb>
    <event_type>CREATE_FLAVOR</event_type>
    <external_flavor_id>fb63c114-de1a-40d8-abae-fa6011abd150</external_flavor_id>
    <name>lightVMFlavor</name>
    <internal_flavor_id>my_flavor_id</internal_flavor_id>
    <memory_mb>2048</memory_mb>
    <vcpus>2</vcpus>
</flavor>
```

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internalFlavorId	**(required)**	**Internal flavor ID (max length 72 characters)**	path	string
body	**(required)** Parameter content type: application/xml	**ESC datamodel with the flavor to create**	body	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	Flavor Id not found		

HTTP Status Code	Reason	Response Model	Headers
406	Operation not supported on VIM/Flavor part of service registration		
409	Conflicting Request		
500	Internal server error		

Try it out!

06-Deployments : Manage ESC deployments

GET /v0/deployments — Find all deployments

Implementation Notes

All deployments created in ESC will be returned as a List

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		

Try it out!

GET /v0/deployments/config/{internal_deployment_id} — Find deployment configuration data by its internal ID

Implementation Notes

Deployment configuration data will be returned if found in CDB

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internal_deployment_id	**(required)**	**Internal ID of the deployment (max length 1024 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Deployment Id not found		

Try it out!

DELETE /v0/deployments/{internal_deployment_id} — Delete a deployment from ESC

Implementation Notes

Delete a deployment from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internal_deployment_id	**(required)**	**Internal ID of the deployment (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	Deployment Id not found		

Try it out!

GET /v0/deployments/{internal_deployment_id} Find a deployment by its internal ID

Implementation Notes

A deployment will be returned if found in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internal_deployment_id	**(required)**	**Internal ID of the deployment (max length 1024 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Deployment Id not found		

Try it out!

POST /v0/deployments/{internal_deployment_id} Creates a deployment

Implementation Notes

Creates a Simple deployment of Instance VMs.

NOTE: As deployment requirements vary a lot, please see ESC User guide for datamodel excerpts of specific functionality you may desire that's not displayed in the simple example below.

Example request:

```
POST /v0/deployments/my_dep_id HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Callback: http://127.0.0.1:9010/
Callback-ESC-Events: http://127.0.0.1:9010/

Body:

<esc_datamodel xmlns="http://www.cisco.com/esc/esc" xmlns:ns2="urn:ietf:params:xml:ns:netconf:notification:1.
  <tenants>
    <tenant>
      <name>admin</name>
      <deployments>
        <deployment>
          <name>MyDemoDepName</name>
          <vm_group>
            <name>ASA-1</name>
            <bootup_time>300</bootup_time>
            <reboot_time>100</reboot_time>
            <recovery_wait_time>10</recovery_wait_time>
            <image>ASA</image>
            <interfaces>
              <interface>
                <nicid>1</nicid>
                <network>MgtNetwork</network>
                <ip_address>18.0.0.16</ip_address>
              </interface>
            </interfaces>
            <scaling>
              <min_active>1</min_active>
              <max_active>1</max_active>
              <elastic>true</elastic>
              <static_ip_address_pool>
                <network>MgtNetwork</network>
                <ip_address>18.0.0.16</ip_address>
              </static_ip_address_pool>
            </scaling>
            <kpi_data>
              <kpi>
                <event_name>VM_ALIVE</event_name>
                <metric_value>1</metric_value>
                <metric_cond>GT</metric_cond>
                <metric_type>UINT32</metric_type>
                <metric_collector>
                  <type>ICMPPing</type>
                  <nicid>1</nicid>
                  <poll_frequency>3</poll_frequency>
                  <polling_unit>seconds</polling_unit>
```

```
                <continuous_alarm>false</continuous_alarm>
              </metric_collector></kpi>
            </kpi_data>
            <rules>
              <admin_rules>
                <rule>
                  <event_name>VM_ALIVE</event_name>
                  <action>"ALWAYS log"</action>
                  <action>"TRUE esc_vm_alive_notification"</action>
                </rule>
              </admin_rules>
            </rules>
            <config_data>
              <configuration>
                <dst>day0-config</dst>
                <file>http://MY_FILE_SERVER_IP:/share/images/day0-configASAv</file>
              </configuration>
              <configuration>
                <dst>idtoken</dst>
                <file>http://MY_FILE_SERVER_IP:/share/images/asa-smart.config</file>
              </configuration>
            </config_data>
            <placement>
              <type>zone_host</type>
              <zone>MY_CLUSTER1</zone>
            </placement>
            <volumes>
              <volume>
                <name>MY_DATASTORE1</name>
                <volid>1</volid>
              </volume>
            </volumes>
          </vm_group>
        </deployment>
      </deployments>
    </tenant>
  </tenants>
</esc_datamodel>
```

While the Deployment operation is being processed successfully, ESC will send a VM_DEPLOYED callback for each VM created on the VIM as part of this deployment

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: VM Deployed

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <datacenter>
    <default>false</default>
  </datacenter>
  <deployment_details>
```

```
    <host_uuid>host-14882</host_uuid>
    <host_name>10.85.103.13</host_name>
    <vm_uuid>50217d2d-8afb-41c2-cffd-d76b1239f51d</vm_uuid>
    <interfaces>
      <interface>
        <network_uuid>MgtNetwork</network_uuid>
        <ip_address>18.0.0.16</ip_address>
        <mac_address>00:50:56:a1:6a:12</mac_address>
        <nic_id>1</nic_id>
        <port_forwarding></port_forwarding>
        <port_uuid></port_uuid>
        <security_groups></security_groups>
        <subnet_uuid></subnet_uuid>
        <type>virtual</type>
      </interface>
    </interfaces>
    <vm_group_name>ASA-1</vm_group_name>
    <vm_name>Sanity-vmware-de_ASA-1_0_b323effd-7f70-4055-b11e-0ef66f0a758d</vm_name>
    <vm_state_machine_state>VM_INERT_STATE</vm_state_machine_state>
  </deployment_details>
  <deployment_name>MyDemoDepName</deployment_name>
  <deployment_stage>VM_DEPLOYED</deployment_stage>
  <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ac</external_deployment_id>
  <external_tenant_id>SystemAdminTenantId</external_tenant_id>
  <internal_deployment_id>my_dep_id</internal_deployment_id>
  <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
  <service_state_machine_state>SERVICE_DEPLOYING_STATE</service_state_machine_state>
</deployment>
```

While the Deployment operation is being processed successfully, ESC will send a VM_ALIVE callback for each monitorable/pingable VM created on the VIM as part of this deployment

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: VM_Alive event received, VM ID: [Sanity-vmware-de_ASA-1_0_b323effd-7f70-4055-b11e-0ef66f0

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <datacenter>
    <default>false</default>
  </datacenter>
  <deployment_details>
    <host_uuid>host-14882</host_uuid>
    <host_name>10.85.103.13</host_name>
    <vm_uuid>50217d2d-8afb-41c2-cffd-d76b1239f51d</vm_uuid>
    <interfaces>
      <interface>
        <network_uuid>MgtNetwork</network_uuid>
        <ip_address>18.0.0.16</ip_address>
        <mac_address>00:50:56:a1:6a:12</mac_address>
        <nic_id>1</nic_id>
        <port_forwarding></port_forwarding>
```

```
        <port_uuid></port_uuid>
        <security_groups></security_groups>
        <subnet_uuid></subnet_uuid>
        <type>virtual</type>
      </interface>
    </interfaces>
    <vm_group_name>ASA-1</vm_group_name>
    <vm_name>Sanity-vmware-de_ASA-1_0_b323effd-7f70-4055-b11e-0ef66f0a758d</vm_name>
    <vm_state_machine_state>VM_ALIVE_STATE</vm_state_machine_state>
  </deployment_details>
  <deployment_name>MyDemoDepName</deployment_name>
  <deployment_stage>VM_ALIVE</deployment_stage>
  <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ac</external_deployment_id>
  <external_tenant_id>SystemAdminTenantId</external_tenant_id>
  <internal_deployment_id>my_dep_id</internal_deployment_id>
  <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
  <service_state_machine_state>SERVICE_INERT_STATE</service_state_machine_state>
</deployment>
```

After the Deployment operation is done, ESC will send a SERVICE_ALIVE callback to the client with the result.

Example callback:

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: Service group deployment completed successfully!

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <datacenter>
    <default>false</default>
  </datacenter>
  <deployment_name>MyDemoDepName</deployment_name>
  <deployment_stage>SERVICE_ALIVE</deployment_stage>
  <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ac</external_deployment_id>
  <external_tenant_id>SystemAdminTenantId</external_tenant_id>
  <internal_deployment_id>my_dep_id</internal_deployment_id>
  <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
  <service_state_machine_state>SERVICE_ACTIVE_STATE</service_state_machine_state>
</deployment>
```

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
Callback-ESC-Events	**(required)**	**Callback URL for ESC event notifications**	header	string

Parameter	Value	Description	Parameter Type	Data Type
internal_deployment_id	**(required)**	**Internal ID of the deployment (max length 72 characters)**	path	string
body	**(required)** Parameter content type: application/xml	**ESC datamodel to deploy**	body	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	An associated resource not found		
406	Not Acceptable		
409	Conflicting Request		
500	Internal server error		
503	Failed to query VIM		

Try it out!

PUT /v0/deployments/{internal_deployment_id} Update an existing deployment in ESC

Implementation Notes

Updates an existing deployment. Updates could be addition or removal of VM groups, interfaces, networks, KPIs and Rules, Day 0 config.

After the service update operation is done, ESC will send a callback to the client with the result.

Example request of adding VM group:

```
PUT /v0/deployments/my_dep_id HTTP/1.1
Content-Type: application/xml
Accept: application/xml
Callback: http://127.0.0.1:9010/
Callback-ESC-Events: http://127.0.0.1:9010/

Body:

    <esc_datamodel xmlns:ns2="urn:ietf:params:xml:ns:netconf:notification:1.0" xmlns="http://www.cisco.com/es
    <tenants>
    <tenant>
    <name>admin</name>
```

```
<deployments>
<deployment>
<name>MyDemoDepName</name>
<vm_group>
<name>ASA-1</name>
<bootup_time>300</bootup_time>
<recovery_wait_time>0</recovery_wait_time>
<image>ASA</image>
<interfaces>
<interface>
<nicid>1</nicid>
<network>MgtNetwork</network>
<ip_address>18.0.0.16</ip_address>
</interface>
</interfaces>
<scaling>
 <min_active>1</min_active>
 <max_active>1</max_active>
 <elastic>true</elastic>
 <static_ip_address_pool>
  <network>MgtNetwork</network>
  <ip_address>18.0.0.16</ip_address>
 </static_ip_address_pool>
</scaling>
<kpi_data>
 <kpi>
  <event_name>VM_ALIVE</event_name>
  <metric_value>1</metric_value>
  <metric_cond>GT</metric_cond>
  <metric_type>UINT32</metric_type>
  <metric_collector>
  <type>ICMPPing</type>
  <nicid>1</nicid>
  <poll_frequency>3</poll_frequency>
  <polling_unit>seconds</polling_unit>
  <continuous_alarm>false</continuous_alarm>
 </metric_collector></kpi>
</kpi_data>
<rules>
 <admin_rules>
  <rule>
   <event_name>VM_ALIVE</event_name>
   <action>"ALWAYS log"</action>
   <action>"TRUE esc_vm_alive_notification"</action>
  </rule>
 </admin_rules>
</rules>
<config_data>
 <configuration>
  <dst>day0-config</dst>
  <file>http://MY_FILE_SERVER_IP:/share/images/day0-configASAv</file>
 </configuration>
 <configuration>
  <dst>idtoken</dst>
  <file>http://MY_FILE_SERVER_IP:/share/images/asa-smart.config</file>
```

```
 </configuration>
</config_data>
<placement>
 <type>zone_host</type>
 <zone>MY_CLUSTER1</zone>
</placement>
<volumes>
<volume>
 <name>MY_DATASTORE1</name>
 <volid>1</volid>
</volume>
</volumes>
</vm_group>
<name>CSR-1</name>
<bootup_time>300</bootup_time>
<recovery_wait_time>0</recovery_wait_time>
<image>ASA</image>
<interfaces>
<interface>
<nicid>1</nicid>
<network>MgtNetwork</network>
<ip_address>18.0.0.16</ip_address>
</interface>
</interfaces>
<scaling>
 <min_active>1</min_active>
 <max_active>1</max_active>
 <elastic>true</elastic>
 <static_ip_address_pool>
  <network>MgtNetwork</network>
  <ip_address>18.0.0.16</ip_address>
 </static_ip_address_pool>
</scaling>
<kpi_data>
 <kpi>
  <event_name>VM_ALIVE</event_name>
  <metric_value>1</metric_value>
  <metric_cond>GT</metric_cond>
  <metric_type>UINT32</metric_type>
  <metric_collector>
  <type>ICMPPing</type>
  <nicid>1</nicid>
  <poll_frequency>3</poll_frequency>
  <polling_unit>seconds</polling_unit>
  <continuous_alarm>false</continuous_alarm>
 </metric_collector></kpi>
</kpi_data>
<rules>
 <admin_rules>
  <rule>
   <event_name>VM_ALIVE</event_name>
   <action>"ALWAYS log"</action>
   <action>"TRUE esc_vm_alive_notification"</action>
  </rule>
 </admin_rules>
```

```
    </rules>
    <config_data>
     <configuration>
      <dst>day0-config</dst>
      <file>http://MY_FILE_SERVER_IP:/share/images/day0-configASAv</file>
     </configuration>
     <configuration>
      <dst>idtoken</dst>
      <file>http://MY_FILE_SERVER_IP:/share/images/asa-smart.config</file>
     </configuration>
    </config_data>
    <placement>
     <type>zone_host</type>
     <zone>MY_CLUSTER1</zone>
    </placement>
    <volumes>
    <volume>
     <name>MY_DATASTORE1</name>
     <volid>1</volid>
    </volume>
    </volumes>

    </deployment>
    </deployments>
    </tenant>
    </tenants>
    </esc_datamodel>
```

While the Deployment Update operation is being processed successfully, ESC will send a VM_DEPLOYED callback for each VM created on the VIM as part of this update request.

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: VM Deployed

    <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <datacenter><default>false</default></datacenter>
    <deployment_details>
    <host_uuid>host-14882</host_uuid>
    <host_name>10.85.103.13</host_name>
    <vm_uuid>50217d2d-8afb-41c2-cffd-d76b1239f51e</vm_uuid>
    <interfaces>
    <interface>
    <network_uuid>MgtNetwork</network_uuid>
    <ip_address>18.0.0.17</ip_address>
    <mac_address>00:50:56:a1:6a:13</mac_address>
    <nic_id>1</nic_id>
    <port_forwarding></port_forwarding>
    <port_uuid></port_uuid>
    <security_groups></security_groups>
    <subnet_uuid></subnet_uuid>
```

```
    <type>virtual</type>
    </interface>
    </interfaces>
    <vm_group_name>CSR-1</vm_group_name>
    <vm_name>Sanity-vmware-de_CSR-1_0_b323effd-7f70-4055-b11e-0ef66f0a758d</vm_name>
    <vm_state_machine_state>VM_INERT_STATE</vm_state_machine_state>
    </deployment_details>
    <deployment_name>MyDemoDepName</deployment_name>
    <deployment_stage>VM_DEPLOYED</deployment_stage>
    <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ad</external_deployment_id>
    <external_tenant_id>SystemAdminTenantId</external_tenant_id>
    <internal_deployment_id>my_dep_id</internal_deployment_id>
    <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
    <service_state_machine_state>SERVICE_DEPLOYING_STATE</service_state_machine_state>
    </deployment>
```

While the Deployment Update operation is being processed successfully, ESC will send a VM_ALIVE callback for each monitorable/pingable VM on the VIM created as part of this update request.

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: VM_Alive event received, VM ID: [Sanity-vmware-de_CSR-1_0_b323effd-7f70-4055-b11e-0ef66f0

    <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <datacenter><default>false</default></datacenter>
    <deployment_details>
    <host_uuid>host-14882</host_uuid>
    <host_name>10.85.103.13</host_name>
    <vm_uuid>50217d2d-8afb-41c2-cffd-d76b1239f51e</vm_uuid>
    <interfaces>
    <interface>
    <network_uuid>MgtNetwork</network_uuid>
    <ip_address>18.0.0.17</ip_address>
    <mac_address>00:50:56:a1:6a:13</mac_address>
    <nic_id>1</nic_id>
    <port_forwarding></port_forwarding>
    <port_uuid></port_uuid>
    <security_groups></security_groups>
    <subnet_uuid></subnet_uuid>
    <type>virtual</type>
    </interface>
    </interfaces>
    <vm_group_name>CSR-1</vm_group_name>
    <vm_name>Sanity-vmware-de_CSR-1_0_b323effd-7f70-4055-b11e-0ef66f0a758d</vm_name>
    <vm_state_machine_state>VM_ALIVE_STATE</vm_state_machine_state>
    </deployment_details>
    <deployment_name>MyDemoDepName</deployment_name>
    <deployment_stage>VM_ALIVE</deployment_stage>
    <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ac</external_deployment_id>
    <external_tenant_id>SystemAdminTenantId</external_tenant_id>
    <internal_deployment_id>my_dep_id</internal_deployment_id>
```

```
    <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
    <service_state_machine_state>SERVICE_INERT_STATE</service_state_machine_state>
    </deployment>
```

After the Deployment Update operation is done, ESC will send a SERVICE_UPDATED callback to the client with the result.

Example callback:

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: Service group update completed successfully

    <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <datacenter><default>false</default></datacenter>
    <deployment_name>MyDemoDepName</deployment_name>
    <deployment_stage>SERVICE_UPDATED</deployment_stage>
    <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ac</external_deployment_id>
    <external_tenant_id>SystemAdminTenantId</external_tenant_id>
    <internal_deployment_id>my_dep_id</internal_deployment_id>
    <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
    <service_state_machine_state>SERVICE_ACTIVE_STATE</service_state_machine_state>
    </deployment>
```

If a VM specific resource like interface, is updated, then a VM_UPDATED followed by a SERVICE_UPDATED notification is received.

Example callback:

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: VM has been updated successfully. vm: VM ID: [Sanity-vmware-de_CSR-1_0_b323effd-7f70-4055

    <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <datacenter><default>false</default></datacenter>
    <deployment_details>
    <host_uuid>host-14882</host_uuid>
    <host_name>10.85.103.13</host_name>
    <vm_uuid>50217d2d-8afb-41c2-cffd-d76b1239f51e</vm_uuid>
    <interfaces>
    <interface>
    <network_uuid>MgtNetwork</network_uuid>
    <ip_address>18.0.0.17</ip_address>
    <mac_address>00:50:56:a1:6a:13</mac_address>
    <nic_id>1</nic_id>
    <port_forwarding></port_forwarding>
    <port_uuid></port_uuid>
    <security_groups></security_groups>
    <subnet_uuid></subnet_uuid>
```

```
    <type>virtual</type>
    </interface>
    <interface>
    <network_uuid>MgtNetwork2</network_uuid>
    <ip_address>18.0.1.18</ip_address>
    <mac_address>00:50:56:a1:6b:14</mac_address>
    <nic_id>2</nic_id>
    <port_forwarding></port_forwarding>
    <port_uuid></port_uuid>
    <security_groups></security_groups>
    <subnet_uuid></subnet_uuid>
    <type>virtual</type>
    </interface>
    </interfaces>
    <vm_group_name>CSR-1</vm_group_name>
    <vm_name>Sanity-vmware-de_CSR-1_0_b323effd-7f70-4055-b11e-0ef66f0a758d</vm_name>
    <vm_state_machine_state>VM_ALIVE_STATE</vm_state_machine_state>
    </deployment_details>
    <deployment_name>MyDemoDepName</deployment_name>
    <deployment_stage>VM_ALIVE</deployment_stage>
    <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ac</external_deployment_id>
    <external_tenant_id>SystemAdminTenantId</external_tenant_id>
    <internal_deployment_id>my_dep_id</internal_deployment_id>
    <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
    <service_state_machine_state>SERVICE_INERT_STATE</service_state_machine_state>
    </deployment>
```

Followed by:

```
HTTP/1.1 201 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: d53f0047-2315-466c-b7e3-aa78e0a567e8
ESC-Status-Code: 200
ESC-Status-Message: Service group update completed successfully

    <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
    <deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <datacenter><default>false</default></datacenter>
    <deployment_name>MyDemoDepName</deployment_name>
    <deployment_stage>SERVICE_UPDATED</deployment_stage>
    <external_deployment_id>8d084aab-6a66-4a70-8f47-42b91f7477ac</external_deployment_id>
    <external_tenant_id>SystemAdminTenantId</external_tenant_id>
    <internal_deployment_id>my_dep_id</internal_deployment_id>
    <internal_tenant_id>SystemAdminTenantId</internal_tenant_id>
    <service_state_machine_state>SERVICE_ACTIVE_STATE</service_state_machine_state>
    </deployment>
```

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string

Parameter	Value	Description	Parameter Type	Data Type
Callback-ESC-Events	**(required)**	**Callback URL for ESC event notifications**	header	string
internal_deployment_id	**(required)**	**Internal ID of the deployment (max length 72 characters)**	path	string
body	**(required)** Parameter content type: application/xml	**ESC datamodel to update**	body	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	An associated resource not found		
406	Not Acceptable		
409	Conflicting Request		
500	Internal server error		

Try it out!

07-Deployment Operations : Operations to a deployment

POST /v0/{internal_tenant_id}/deployments/service/{internal_deployment_id}

Handle deployment operation request with the operation type as payload

Implementation Notes

Performs actions like stop, start, reboot, disable_monitoring or enable_monitoring on a deployment. Example request:

```
POST /v0/tenants/test-tenant1/deployments/vm/test-dep1-cirros_overid_0_cfeefcee-a1fe-4e94-9ffc-81792b265788 H
Content-Type: application/xml
Accept: application/json
Callback: http://127.0.0.1:9010/

Body:
<service_operation xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <operation>stop</operation>
</service_operation>
```

While the stop deployment operation is being performed, ESC will send a VM_STOPPED callback to the client for each VM stopped as part of this request. Example callback:

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: 5e47b5ff-bc59-4ece-b15b-8941910def74
ESC-Status-Code: 200
ESC-Status-Message: VM successfully stopped

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <datacenter>
    <default>false</default>
  </datacenter>
  <deployment_details>
    <host_uuid>17be6eae40795ec068ce77a05fcb1c47f7ac42dfba160261719d4087</host_uuid>
    <host_name>my-ucs-3</host_name>
    <vm_uuid>cb86a952-c030-46f6-96f6-e81069bcea3a</vm_uuid>
    <interfaces>
      <interface>
        <network_uuid>943fda9e-79f8-400c-b442-3506f102721a</network_uuid>
        <gateway>192.168.0.1</gateway>
        <ip_address>192.168.0.135</ip_address>
        <mac_address>fa:16:3e:3c:17:b0</mac_address>
        <netmask>255.255.255.0</netmask>
        <nic_id>0</nic_id>
        <port_forwarding></port_forwarding>
        <port_uuid>263eaf20-2662-4a6b-a1f1-ed76001a1b5c</port_uuid>
        <security_groups></security_groups>
        <subnet_uuid>e313b95c-ca1f-4c81-8d60-c9e721a85d0b</subnet_uuid>
        <type>virtual</type>
      </interface>
    </interfaces>
    <vm_group_name>overidegrp2</vm_group_name>
    <vm_name>test-dep1-cirros_overid_0_cfeefcee-a1fe-4e94-9ffc-81792b265788</vm_name>
    <vm_state_machine_state>VM_SHUTOFF_STATE</vm_state_machine_state>
  </deployment_details>
  <deployment_name>test-dep1-cirros</deployment_name>
  <deployment_stage>VM_STOPPED</deployment_stage>
  <external_deployment_id>5c278010-ad0f-4e29-bdce-f4b6c6946c38</external_deployment_id>
  <external_tenant_id>29f2cd7934dc4311ad3a34d72d1ab7b8</external_tenant_id>
  <internal_deployment_id>test-tenant1test-dep1-cirros</internal_deployment_id>
  <internal_tenant_id>test-tenant1</internal_tenant_id>
  <service_state_machine_state>SERVICE_STOPPED_STATE</service_state_machine_state>
</deployment>
```

After the stop deployment operation is done successfully, ESC will send a SERVICE_STOPPED callback to the client with the result. Example callback:

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: 5e47b5ff-bc59-4ece-b15b-8941910def74
ESC-Status-Code: 200
ESC-Status-Message: Service group suspension completed successfully
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<deployment xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <datacenter>
    <default>false</default>
  </datacenter>
  <deployment_details>
    <host_uuid>17be6eae40795ec068ce77a05fcb1c47f7ac42dfba160261719d4087</host_uuid>
    <host_name>my-ucs-3</host_name>
    <vm_uuid>cb86a952-c030-46f6-96f6-e81069bcea3a</vm_uuid>
    <interfaces>
      <interface>
        <network_uuid>943fda9e-79f8-400c-b442-3506f102721a</network_uuid>
        <gateway>192.168.0.1</gateway>
        <ip_address>192.168.0.135</ip_address>
        <mac_address>fa:16:3e:3c:17:b0</mac_address>
        <netmask>255.255.255.0</netmask>
        <nic_id>0</nic_id>
        <port_forwarding></port_forwarding>
        <port_uuid>263eaf20-2662-4a6b-a1f1-ed76001a1b5c</port_uuid>
        <security_groups></security_groups>
        <subnet_uuid>e313b95c-ca1f-4c81-8d60-c9e721a85d0b</subnet_uuid>
        <type>virtual</type>
      </interface>
    </interfaces>
    <vm_group_name>overidegrp2</vm_group_name>
    <vm_name>test-dep1-cirros_overid_0_cfeefcee-a1fe-4e94-9ffc-81792b265788</vm_name>
    <vm_state_machine_state>VM_SHUTOFF_STATE</vm_state_machine_state>
  </deployment_details>
  <deployment_name>test-dep1-cirros</deployment_name>
  <deployment_stage>SERVICE_STOPPED</deployment_stage>
  <external_deployment_id>5c278010-ad0f-4e29-bdce-f4b6c6946c38</external_deployment_id>
  <external_tenant_id>29f2cd7934dc4311ad3a34d72d1ab7b8</external_tenant_id>
  <internal_deployment_id>test-tenant1test-dep1-cirros</internal_deployment_id>
  <internal_tenant_id>test-tenant1</internal_tenant_id>
  <service_state_machine_state>SERVICE_STOPPED_STATE</service_state_machine_state>
</deployment>
```

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
Callback-ESC-Events	**(required)**	**Callback URL for ESC event notifications**	header	string
internal_tenant_id	**(required)**	**Internal tenant ID (max length 72 characters)**	path	string

Parameter	Value	Description	Parameter Type	Data Type
internal_deployment_id	**(required)**	**Internal deployment ID (max length 72 characters)**	path	string

body	**(required)**
	Parameter content type: application/xml
		Deployment operation payload specifying the type of operation	body	Model | Model Schema
				{
				"operation": "stop",
				"properties": {}
				}
				Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Deployment operation request accepted		
400	Missing callback or, Deployment operation is not applicable		
404	Deployment or tenant not found		

Try it out!

08-VM Operations : Operations to a VM

POST /v0/{internal_tenant_id}/deployments/recovery-vm/{vm_name}

Handle recovery VM operation request with the operation type as payload

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback		Callback URL for notifications	header	string
Callback-ESC-Events		Callback URL for ESC event notifications	header	string
internal_tenant_id	**(required)**	**Internal tenant ID (max length 72 characters)**	path	string
vm_name	**(required)**	**VM name**	path	string

Parameter	Value	Description	Parameter Type	Data Type

body	(required)
	Parameter content type: application/xml
		Recovery VM operation payload specifying the type of operation: recovery_do	body	Model | Model Schema
				{
				"operation": "string",
				"checkVmBeforeRecovery": true,
				"properties": {}
				}
				Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Recovery VM operation request accepted		
400	Recovery VM operation is not applicable		
404	VM or tenant not found		

Try it out!

POST /v0/{internal_tenant_id}/deployments/snapshot-vm/{vm_name}

Handle VM snapshot operation request with the operation type as payload

Implementation Notes

vm_operation_notes

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for notifications**	header	string
Callback-ESC-Events	(required)	**Callback URL for ESC event notifications**	header	string
internal_tenant_id	(required)	**Internal tenant ID (max length 72 characters)**	path	string
vm_name	(required)	**VM name**	path	string

Parameter	Value	Description	Parameter Type	Data Type
body	(required)	**VM operation payload specifying the type of operation**	body	Model | Model Schema

Parameter content type: application/xml


```
{
  "operation": "snapshot",
  "name": "string",
  "metadata": {}
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	VM snapshot operation request accepted		
400	Missing callback or, VM operation is not applicable		
404	VM or tenant not found		

Try it out!

POST /v0/{internal_tenant_id}/deployments/vm/{vm_name}

Handle VM operation request with the operation type as payload

Implementation Notes

Performs actions like stop, start, reboot, disable_monitoring or enable_monitoring on an existing VM managed by ESC. Example request:

```
POST /v0/tenants/test-tenant1/deployments/vm/test-dep1-cirros_overid_0_cfeefcee-a1fe-4e94-9ffc-81792b265788 H
Content-Type: application/xml
Accept: application/json
Callback: http://127.0.0.1:9010/
Callback-ESC-Events: http://127.0.0.1:9010/

Body:
<vm_operation xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <operation>stop</operation>
</vm_operation>
```

After the VM stop operation is done, ESC will send a VM_STOP_COMPLETE callback to the client with the result. Example callback:

```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=UTF-8
ESC_TRANSACTION_ID: 4546c40a-d6e3-4abe-a7e7-51afd4fc5362
ESC-Status-Code: 200
ESC-Status-Message: Successfully stopped VM [test-dep1-cirros_overid_0_cfeefcee-a1fe-4e94-9ffc-81792b265788].

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<esc_event xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <deployment_name>test-dep1-cirros</deployment_name>
  <event_details></event_details>
  <event_type>VM_STOP_COMPLETE</event_type>
  <external_deployment_id>5c278010-ad0f-4e29-bdce-f4b6c6946c38</external_deployment_id>
  <external_tenant_id>29f2cd7934dc4311ad3a34d72d1ab7b8</external_tenant_id>
  <internal_deployment_id>test-tenant1test-dep1-cirros</internal_deployment_id>
  <internal_tenant_id>test-tenant1</internal_tenant_id>
  <vm_source>
    <interfaces>
      <network_uuid>943fda9e-79f8-400c-b442-3506f102721a</network_uuid>
      <gateway>192.168.0.1</gateway>
      <ip_address>192.168.0.135</ip_address>
      <mac_address>fa:16:3e:3c:17:b0</mac_address>
      <netmask>255.255.255.0</netmask>
      <nic_id>0</nic_id>
      <port_forwarding></port_forwarding>
      <port_uuid>263eaf20-2662-4a6b-a1f1-ed76001a1b5c</port_uuid>
      <security_groups></security_groups>
      <subnet_uuid>e313b95c-ca1f-4c81-8d60-c9e721a85d0b</subnet_uuid>
      <type>virtual</type>
    </interfaces>
    <host_uuid>17be6eae40795ec068ce77a05fcb1c47f7ac42dfba160261719d4087</host_uuid>
    <host_name>my-ucs-3</host_name>
    <vm_uuid>cb86a952-c030-46f6-96f6-e81069bcea3a</vm_uuid>
    <vm_group_name>overidegrp2</vm_group_name>
    <vm_name>test-dep1-cirros_overid_0_cfeefcee-a1fe-4e94-9ffc-81792b265788</vm_name>
  </vm_source>
</esc_event>
```

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for notifications**	header	string
Callback-ESC-Events	**(required)**	**Callback URL for ESC event notifications**	header	string
internal_tenant_id	**(required)**	**Internal tenant ID (max length 72 characters)**	path	string
vm_name	**(required)**	**VM name**	path	string

Parameter	Value	Description	Parameter Type	Data Type
body	(required)	**VM operation payload specifying the type of operation**	body	Model | Model Schema

Parameter content type: application/xml

```
{
  "operation": "stop"
}
```

Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	VM operation request accepted		
400	Missing callback or, VM operation is not applicable		
404	VM or tenant not found		

Try it out!

09-Configuration Parameters

GET /v0/config/{category}/{key} — Retrieve a ESC config parameters under a category, with a given key

Implementation Notes

Retrieve a ESC config parameters under a category, with a given key

Parameters

Parameter	Value	Description	Parameter Type	Data Type
category	(required)	**ESC config parameter category**	path	string
key	(required)	**ESC config parameter key**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Get key under category request accepted.		
400	Invalid key or category name provided.		
404	Key or category not found.		

Try it out!

POST /v0/config/{category}/{key} — Create a new key under a category.

Implementation Notes

Create a new key under a catgeory.

Response Class (Status 200)

Model | Model Schema


```
{
  "category": "DEFAULT",
  "key": "MAX_LOGS",
  "value": {},
  "type": "STRING"
}
```


Response Content Type application/xml

Parameters

Parameter	Value	Description	Parameter Type	Data Type
category	**(required)**	**ESC config parameter category**	path	string
key	**(required)**	**ESC config parameter key**	path	string
body	**(required)** Parameter content type: application/json	**ESC config parameter value**	body	string

Try it out!

PUT /v0/config/{category}/{key}/{value} Change an ESC config parameter under a category, with a given key

Implementation Notes

Change an ESC config parameter under a category, with a given key.

Format: PUT /v0/config/{category}/{key}/{value}

Example request:

curl -X PUT -H "accept: Application/xml" http://127.0.0.1:8080/ESCManager/v0/config/log/level/debug

Example response:


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<escconfigparameter>
  <category>LOG</category>
```

```
    <key>LEVEL</key>
    <type>STRING</type>
  <value xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs="http://www.w3.org/2001/XMLSchema"
  </escconfigparameter>
```

Details of Config Parameters: Please refer to installation and user guide for more details.

Category: DEFAULT Key: VM_RECOVERY_RETRIES_MAX Default Value: 3 Type: Int Description: Number of maximum recovery attempts allowed per VM.

Category: OPENSTACK Key: ENDPOINT Default Value: adminURL Type: String Description: The parameter to set up the keystone endpoint value of ESC. Options : adminURL, publicURL

Category: LOG Key: LEVEL Default Value: INFO Type: String Description: Level of logging of ESCManager. Options: INFO, TRACE, DEBUG.

Category: AFFINITY Key: FILTER Default Value: SameHostFilter Type: String Description: A constant string used to build PolicyEngine and initializing VM policy table. Options: SameHostFilter, ServerGroupAffinity

Category: ANTI-AFFINITY Key: FILTER Default Value: DifferentHostFilter Type: String Description: A constant string used to build PolicyEngine and initializing VM policy table. Options: DifferentHostFilter, ServerGroupAffinity

Parameters

Parameter	Value	Description	Parameter Type	Data Type
category	**(required)**	**ESC config parameter category**	path	string
key	**(required)**	**ESC config parameter key**	path	string
value	**(required)**	**ESC config parameter value**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Update value request accepted.		
400	Invalid request.		
404	Associated resource not found.		
406	Not acceptable.		
409	Conflicting resource.		
500	Internal server error.		

Try it out!

GET /v0/config — Retrieve all ESC config parameters

Implementation Notes

Retrieve all ESC config parameters

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Get all parameters request accepted		

Try it out!

GET /v0/config/{category} Retrieve all ESC config parameters under a category

Implementation Notes

Retrieve all ESC config parameters under a category

Parameters

Parameter	Value	Description	Parameter Type	Data Type
category	**(required)**	**ESC config parameter category**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Get category request accepted.		
400	Invalid category name provided.		
404	Category not found.		

Try it out!

10-Volumes

GET /v0/{internalTenantId}/volumes Get all volumes under a tenant

Implementation Notes

All volumes under that tenant in ESC will be returned as a List

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalTenantId	**(required)**	**Internal tenant ID of the volume (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		

Try it out!

DELETE /v0/{internalTenantId}/volumes/{internalVolumeId} Delete a volume from an existing tenant in ESC

Implementation Notes

Delete a volume from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internalVolumeId	**(required)**	**Internal volume ID (max length 72 characters)**	path	string
internalTenantId	**(required)**	**Internal tenant ID of the volume (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid request		
500	Internal server error		

Try it out!

GET /v0/{internalTenantId}/volumes/{internalVolumeId} Get a volume by its tenant and internal volume ID

Implementation Notes

Only volumes created using ESC will be checked

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalTenantId	**(required)**	**Internal tenant ID of the volume (max length 72 characters)**	path	string
internalVolumeId	**(required)**	**Internal volume ID (max length 72 characters)**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Volume Id not found		

Try it out!

POST /v0/{internalTenantId}/volumes/{internalVolumeId} Create a volume under an existing tenant in ESC

Implementation Notes

Create a volume in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
internalTenantId	(required)	**Internal tenant ID of the volume (max length 72 characters)**	path	string
internalVolumeId	(required)	**Internal volume ID (max length 72 characters)**	path	string
body	(required) Parameter content type: application/xml	**Volume configuration payload**	body	Model | Model Schema


```
{
  "external_volume_id": "string",
  "size": "string",
  "sizeunit": "string",
  "bus": "string",
  "type": "string",
  "outOfBand": true,
  "bootIndex": "string",
  "name": "string",
  "format": "string",
  "storageLocation": "string",
  "external_tenant_id": "string",
  "internal_tenant_id": "string",
  "internal_volume_id": "string",
  "volid": 0,
  "event_type": "CREATE_VOLUME",
  "image": "string"
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid request		
404	Volume Id not found		
409	Conflicting request		

HTTP Status Code	Reason	Response Model	Headers
503	Unable to query VIM		

Try it out!

11-ESC Operation Mode

POST /v0/operationmode/{escOpMode} Set operation mode in ESC

Implementation Notes

Set operation mode in ESC

Response Class (Status 200)

Model | Model Schema

```
{
  "message": "string",
  "mode": "string"
}
```

Response Content Type application/xml

Parameters

Parameter	Value	Description	Parameter Type	Data Type
escOpMode	**MAINTENANCE**	**ESC maintenance mode.**	path	string

Try it out!

GET /v0/operationmode Get operation mode in ESC

Implementation Notes

Set operation mode in ESC

Response Class (Status 200)

Model | Model Schema

```
{
  "message": "string",
  "mode": "string"
}
```

Response Content Type application/xml

Try it out!

12-Host Actions

GET /v0/hosts/{hostName}/status — Get the current status of the host

Implementation Notes

This is to see if the host can be schedule to deploy instances

Response Class (Status 200)

Response Content Type application/xml

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Client-Transaction-Id			header	string
hostName	**(required)**		path	string

Try it out!

POST /v0/hosts/{hostName}/disable — Disable a host

Implementation Notes

To disable a host

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback-ESC-Events	**(required)**	**Callback URL for host action notifications**	header	string
hostName	**(required)**	**Target host name**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
default	successful operation		

Try it out!

POST /v0/hosts/{hostName}/enable — Enable a host

Implementation Notes

To enable a host

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback-ESC-Events	**(required)**	**Callback URL for host action notifications**	header	string
hostName	**(required)**	**Target host name**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
default	successful operation		

Try it out!

13-ESC VIM Connectors : Manage ESC VIM Connectors

DELETE /v0/vims/{vim_id} Delete a VIM Connector from ESC

Implementation Notes

Delete a VIM Connector from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
vim_id	**(required)**	**VIM ID**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	VIM Connector deletion request accepted		
400	Invalid Request		
404	VIM Id not found		
500	Internal Server Error		

Try it out!

GET /v0/vims/{vim_id} Get a VIM Connector by its id

Parameters

Parameter	Value	Description	Parameter Type	Data Type
vim_id	**(required)**	**VIM ID**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Get request accepted		
404	VIM Id not found		

Try it out!

PUT /v0/vims/{vim_id} Update an existing VIM connector in ESC

Implementation Notes

Update an existing VIM connector in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
vim_id	(required)	**VIM ID**	path	string
body	(required) Parameter content type: application/xml	**Update VIM Connector configuration payload**	body	Model | Model Schema


```
{
  "id": "string",
  "type": "OPENSTACK",
  "properties": {
    "property": [
      {
        "name": "string",
        "value": "string",
        "encrypted_value": "string"
      }
    ]
  },
  "event_type": "string"
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
500	Internal Server Error		

Try it out!

GET /v0/vims Get all VIM Connectors

Implementation Notes

All VIM Connectors in ESC will be returned as a List

Response Class (Status 200)

Model | Model Schema


```
[
  {
    "id": "string",
    "type": "OPENSTACK",
    "properties": {
      "property": [
        {
          "name": "string",
          "value": "string",
          "encrypted_value": "string"
        }
      ]
    },
    "event_type": "string"
  }
]
```


Response Content Type application/xml

Try it out!

POST /v0/vims — Create a VIM connector in ESC

Implementation Notes

Create a VIM connector in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string

Parameter	Value	Description	Parameter Type	Data Type
body	(required)	**VIM Connector configuration payload**	body	Model | Model Schema

Parameter content type: application/xml


```
{
  "id": "string",
  "type": "OPENSTACK",
  "properties": {
    "property": [
      {
        "name": "string",
        "value": "string",
        "encrypted_value": "string"
      }
    ]
  },
  "event_type": "string"
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
500	Internal Server Error		

Try it out!

GET /v0/vims/default_vim_connector — Get default VIM Connector name

Response Class (Status 200)

Model | Model Schema


```
{
  "defaultVimConnectorId": "string",
  "event_type": "string"
}
```


Response Content Type application/xml

Try it out!

POST /v0/vims/default_vim_connector — Create a Default VIM connector in ESC

Implementation Notes

Create a Defaukt VIM connector in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string

body	(required)
	Parameter content type: application/xml
		Default VIM Connector configuration payload	body	Model | Model Schema
				{
				"defaultVimConnectorId": "string",
				"event_type": "string"
				}
				Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
500	Internal Server Error		

Try it out!

PUT /v0/vims/{vim_id}/check_status — Trigger VIM connector status checking

Implementation Notes

Trigger VIM connector status checking

Parameters

Parameter	Value	Description	Parameter Type	Data Type
vim_id	(required)	**VIM ID**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	Not Found		
500	Internal Server Error		

Try it out!

14-ESC VIM Users : Manage ESC VIM Users

DELETE /v0/vims/{vim_id}/vim_users/{vim_user_id} — Delete a VIM User from ESC

Implementation Notes

Delete a VIM User from ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
vim_id	**(required)**	**Vim id**	path	string
vim_user_id	**(required)**	**Vim user id**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
404	VIM User Id not found		
500	Internal Server Error		

Try it out!

GET /v0/vims/{vim_id}/vim_users/{vim_user_id} Get a VIM User by its id

Parameters

Parameter	Value	Description	Parameter Type	Data Type
vim_id	**(required)**	**Vim id**	path	string
vim_user_id	**(required)**	**Vim user id**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Get request accepted		
404	VIM User Id not found		

Try it out!

PUT /v0/vims/{vim_id}/vim_users/{vim_user_id} Update a VIM user under an existing VIM in ESC

Implementation Notes

Update a VIM user in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
vim_id	(required)	**Vim id**	path	string
vim_user_id	(required)	**Vim user id**	path	string
body	(required) Parameter content type: application/xml	**Vim user configuration payload**	body	Model | Model Schema


```
{
  "credentials": {
    "properties": {
      "property": [
        {
          "name": "string",
          "value": "string",
          "encrypted_value": "string"
        }
      ]
    }
  },
  "id": "string",
  "vim_id": "string",
  "event_type": "string"
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
500	Internal Server Error		

Try it out!

GET /v0/vims/{vim_id}/vim_users — Get all VIM Users

Implementation Notes

All VIM Users in ESC will be returned as a List

Parameters

Parameter	Value	Description	Parameter Type	Data Type
vim_id	(required)		path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Get request accepted		

HTTP Status Code	Reason	Response Model	Headers
404	Unknown VIM Id provided		

Try it out!

POST /v0/vims/{vim_id}/vim_users — Create a VIM user under an existing VIM in ESC

Implementation Notes

Create a VIM user in ESC

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
vim_id	(required)	**Vim id**	path	string
body	(required) Parameter content type: application/xml	**Vim user configuration payload**	body	Model | Model Schema


```
{
  "credentials": {
    "properties": {
      "property": [
        {
          "name": "string",
          "value": "string",
          "encrypted_value": "string"
        }
      ]
    }
  },
  "id": "string",
  "vim_id": "string",
  "event_type": "string"
}
```


Click to set as parameter value

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
400	Invalid Request		
500	Internal Server Error		

Try it out!

15-System States

GET /v0/systemstate/{name} — Retrieve system state of a given name

Implementation Notes

Retrieve system state of a given name

Parameters

Parameter	Value	Description	Parameter Type	Data Type
name	**(required)**	**System state name**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Get system state of specified name request accepted.		
404	Name not found.		

Try it out!

POST /v0/systemstate/{name}/reset — Reset system state of a given name

Implementation Notes

Reset system state of a given name

Parameters

Parameter	Value	Description	Parameter Type	Data Type
name	**(required)**	**System state name**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Reset system state of specified name request accepted.		
404	Name not found.		

Try it out!

16-File Servers

GET /v0/fileserver — Get all File Servers

Implementation Notes

All file servers created will be returned as a List

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		

Try it out!

DELETE /v0/fileserver/{internalServerId} Delete a File Server

Implementation Notes

Delete a File Server

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	**(required)**	**Callback URL for ESC notifications**	header	string
internalServerId	**(required)**	**File Server ID**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Not found		
500	Internal server error		

Try it out!

GET /v0/fileserver/{internalServerId} Get specific File Server

Implementation Notes

Matching File server will be returned if exists

Parameters

Parameter	Value	Description	Parameter Type	Data Type
internalServerId	**(required)**	**File Server Id**	path	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
404	Not found		

Try it out!

POST /v0/fileserver/{internalServerId} Create a File Server

Implementation Notes

Create a File Server

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
internalServerId	(required)	**File Server ID**	path	string
body	(required)	**ESC datamodel with the file server payload**	body	string

Parameter content type: application/xml

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
204	No content		
400	Invalid Request		
409	Conflicting Request		

Try it out!

PUT /v0/fileserver/{internalServerId} Update a File Server

Implementation Notes

Update a File Server

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Callback	(required)	**Callback URL for ESC notifications**	header	string
internalServerId	(required)	**File Server ID**	path	string
body	(required)	**ESC datamodel with the file server payload**	body	string

Parameter content type: application/xml

Response Messages

HTTP Status Code	Reason	Response Model	Headers
200	Request accepted		
204	No content		
400	Invalid Request		
404	Not Found		

Try it out!

SSL Management

PUT /internal/ssl/reload — Reload ESC Truststore from file system

Implementation Notes

Reload ESC Truststore from file system

Response Messages

HTTP Status Code	Reason	Response Model	Headers
204	ESC Truststore reloaded successfully		
500	Failed to reload ESC truststore		

Try it out!

[BASE URL: /ESCManager , API VERSION: 2.2.0]

INVALID {···}