Cisco Elastic Services Controller Troubleshooting Guide

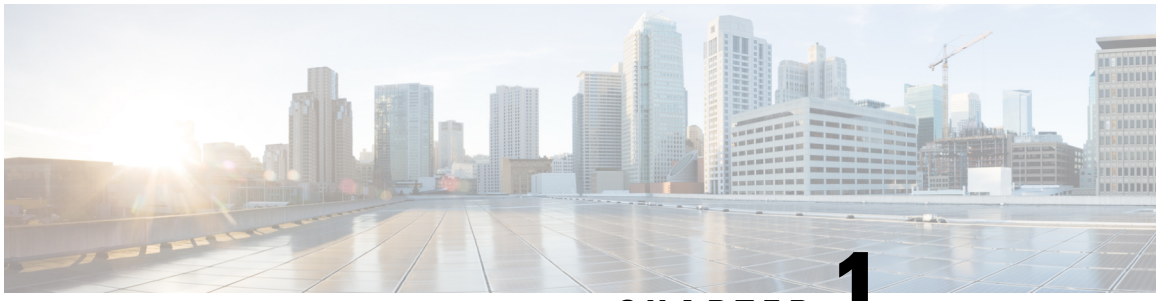
First Published: 2022-07-22

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

PART I

Troubleshooting Cisco Elastic Services Controller



CHAPTER 1

Troubleshooting Cisco Elastic Services Controller Installation

Cisco Elastic Services Controller Installation Troubleshooting Overview

ESC uses a python-based script called bootvm.py for its installation in OpenStack and Libvirt (KVM) environments. Refer to the Cisco Elastic Services Controller Install and Upgrade Guide to understand all the arguments of the boovtm.py script. It is important to use a specific bootvm.py that was released along with the ESC image.

Bootvm.py has the dependency on Python and OpenStack client for ESC OpenStack installation. Ensure that the environment where you plan to execute bootvm.py has the python and OpenStack client installed.

OpenStack Credentials from Cisco Elastic Services Controller Installation Do Not Work

Problem Statement:

You might encounter some errors while executing bootvm.py for ESC installation; one such common error is:

- OpenStack Credentials Errors

Description:

You may get a long python stack trace information after running bootvm.py and you must refer couple of lines at the bottom of the error messages. For example:

```
Unauthorized: The request you have made requires authentication. (HTTP 401) (Request-ID:
req-e93d90b0-aced-4b88-b4ca-bcc3d88e8bc0)
```

```
The request you have made requires authentication. (HTTP 401) (Request-ID:
req-e93d90b0-aced-4b88-b4ca-bcc3d88e8bc0) -- Booting up ESC VM has failed.
```

Solution:

In such scenarios, you must check your OpenStack credentials information either in your bootvm.py arguments, or in your global environment (if you have not specified those in the bootvm.py arguments).

Following is an example to check OpenStack credential parameters through global environment:

```
$ env | grep OS_
OS_USER_DOMAIN_NAME=default
OS_IMAGE_API_VERSION=2
OS_PROJECT_NAME=admin
OS_IDENTITY_API_VERSION=3
OS_PASSWORD=cisco123
OS_AUTH_TYPE=password
OS_AUTH_URL=http://10.85.103.145:35357/v3
OS_USERNAME=admin
OS_TENANT_NAME=admin
OS_PROJECT_DOMAIN_NAME=default
```

Similar to other OpenStack clients (OpenStack, Nova, Neutron, etc.), bootvm.py is used to install ESC on OpenStack. You can pass OpenStack credentials to ESC installer through the following arguments of bootvm.py:

```
--os_auth_url
--os_username
--os_password
--os_tenant_name
--os_project_name
--os_user_domain_name
--os_project_domain_name
--os_identity_api_version

--bs_os_auth_url
--bs_os_username
--bs_os_password
--bs_os_tenant_name
--bs_os_project_name
--bs_os_user_domain_name
--bs_os_project_domain_name
--bs_os_identity_api_version
```

The bootstrap arguments starting with `bs_` is only used for ESC installation on OpenStack, and the arguments starting with `os_` is used for ESC to perform the VNF lifecycle management (as default VIM connector in ESC 3.x).

If you do not specify those arguments, ESC uses the same OpenStack credentials from global environment variables of Linux for both ESC installation and VNF lifecycle management. Similar to the OpenStack client, you can create an openrc file and source the file to add global environment variables.

For OpenStack V2 API, you need the following items exported to your global environment variables:

```
OS_PASSWORD
OS_AUTH_URL
OS_USERNAME
OS_TENANT_NAME
```

For OpenStack V3 API, you must set OS_IDENTITY_API_VERSION=3 to use OpenStack V3 API. You need the following items exported to your global environment variables:

```
OS_USER_DOMAIN_NAME
OS_PROJECT_DOMAIN_NAME
OS_PROJECT_NAME
OS_TENANT_NAME
```

```
OS_PASSWORD
OS_AUTH_URL
OS_USERNAME
OS_IDENTITY_API_VERSION
```

Certificate Verification Failed During Cisco Elastic Services Controller Installation

Problem Statement:

If your OpenStack is configured with self-signed certificate but you don't provide the ca_cert file for ESC installation, you may hit the following errors:

```
SSLError: SSL exception connecting to https://10.85.103.49:35357/v3: [SSL:
CERTIFICATE_VERIFY_FAILED] certificate verify failed (_ssl.c:590)
SSL exception connecting to https://10.85.103.49:35357/v3: [SSL: CERTIFICATE_VERIFY_FAILED]
 certificate verify failed (_ssl.c:590) -- Booting up ESC VM has failed.
```

Solution:

Bootvm.py does not provide the arguments to be passed in a command line for ESC Installation for a specific CA certification. If your OpenStack endpoint is configured with https (check OS_AUTH_URL) and self-signed certificate, you must set the CA certificate file through the global environment by exporting the following two environment variables:

```
export OS_CACERT=<path_to_ca_cert_file>
export REQUESTS_CA_BUNDLE=<path_to_ca_cert_file>
```

Note The previous approach specifies the CA certificate for ESC installation and not for VNF lifecycle management.

If you want to pass the CA certificate for VNF lifecycle management, specify the following arguments in ESC's bootvm.py commands:

```
--cert_file <path_to_ca_cert_file>
```

Bootvm.py Script Failing with Error

Problem Statement:

While executing the bootvm.py script to create ESC VM, you might encounter the following error, and the bootvm.py script exits abnormally:

```
bootvm script fails with error "object of type 'NoneType' has no len()"
```

Description:

The bootvm.py script fails to authenticate the Open Stack credentials and the connectivity details, because the details are not specified or partially specified.

Solution:

Ensure that you have an openrc file to source with up-to-date values.

For example:

```
export OS_USERNAME=admin
export OS_PASSWORD=<HIDDEN>
export OS_REGION_NAME=RegionOne
export OS_AUTH_URL=http://172.29.91.77:5000/v3
export OS_PROJECT_NAME=admin
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_DOMAIN_NAME=Default
export OS_IDENTITY_API_VERSION=3
```

Source the openrc file before running bootvm.py once again.

You can also pass the values directly on the bootvm.py using `os_<variable_name>`, that is `--os_auth_url=http://172.29.91.77:5000/v3`.

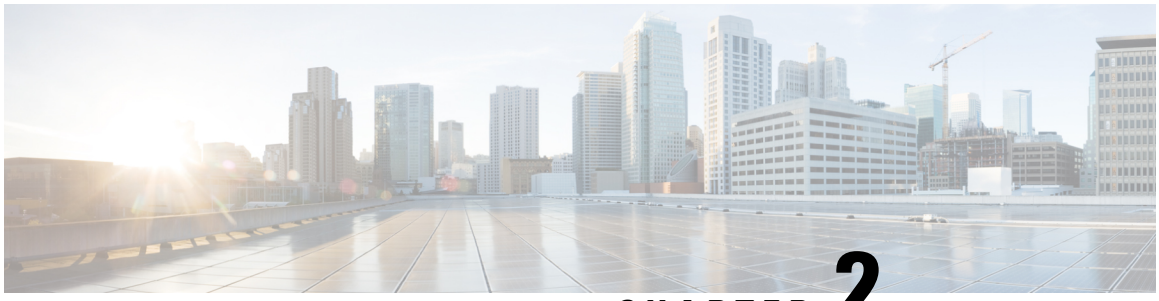
If you still encounter the same error, run the bootvm.py script with the debug option, and redirect the output to a file. To perform this action, add the following command-line argument.

```
--loglevel DEBUG --log /tmp/esc-install.log
```

For any technical support attach the resulting esc-install.log file.

PART II

Troubleshooting Cisco Elastic Services Controller High Availability



CHAPTER 2

Troubleshooting Cisco Elastic Services Controller High Availability

Troubleshooting Cisco Elastic Services Controller High Availability

ESC HA consists of many components/services and keeps monitoring the self health checking. Any failure of ESC micro services causes HA synchronization and other related issues.

Cisco Elastic Services Controller High Availability Troubleshooting Overview

Following are some generic troubleshooting items for ESC HA:

Problem : Network Problem

Solution: If you have a networking problem, check for the following items:

- The static IP addresses for both ESC nodes are correct based on the OpenStack configuration and each node is able to access the other node.
- The gateway for each network interface is accessible from each instance.
- Virtual ipaddress (kad_vip) is pingable from master node. (to find kad_vip, run: "sed -n '/virtual_ipaddress/{n;p;}' /etc/keepalived/keepalived.conf").

Checking the logs:

Following are some logs and their locations to check ESC HA troubleshooting:

- The ESC manager log, located at `/var/log/esc/escmanager.log`.
- The ESC HA log about esc service startup/stop, located at /var/log/esc/esc_haagent.log (ESC 2.X) and /var/log/esc/escadm.log (ESC 3.X).
- The exabgp log, located at `/var/log/exabgp.log`.

Configuration and log check for Keepalived:

Verify the keepalived configuration in the following path:

- You can check the configuration file at `/etc/keepalived/keepalived.conf` to verify the keepalived configuration .
- The keepalived log is located at `/var/log/messages` by grep keepalived or vrrp.

Configuration and log check for DRBD:

Verify the DRBD configuration in the following path:

- To verify the DRBD configuration, check the file at `/etc/drbd.d/esc.res` .
- The DRBD log is located at `/var/log/messages` by grep drbd.

Configuration check for BGP:

Verify the BGP configuration:

- The BGP configuration must be the same as installation arguments and ASR configuration.
- The BGP configuration can be verified by checking the file at /opt/cisco/esc/esc-scripts/bgp-sa/exabgp/neighbor_init.conf.

High Availability Active Node Stays in the Switching-to-Active State

The ESC High Availability (HA) cluster might have some issues at the startup. The following are the possible issues listed:

Problem:

- ESC HA node cannot reach its peer during the initial installation. Verify that ESC HA is able to reach its peer when switching to Active for the first time.
- ESC service (tomcat/escmanager) cannot start properly due to database problems (etc. database migration, database file corruption).
- Confd cannot start due to the CDB file corruption.
- Postgresql cannot start or init due to issues in the file system (disk space is 100% full).
- The connection between ESC nodes is too slow (MTU issue).

Verification:

Verify the following are the items to troublehsoot the previous problems:

- The connectivity between ESC Active node and standby node. For initial installation, ESC active (escadm) service will not be up if it cannot reach the standby node. Ensure that you have both ESC nodes successfully deployed and they can reach each other.
- Check ESC logs at /var/log/esc/esc_haagent.log (ESC 2.X), or /var/log/esc/escadm.log (ESC 3.X and up). In most of the cases, it displays why ESC service gets blocked and which step/service startup did not work well.
- If esc_service/escadm and postgresql have started, check the log at /var/log/esc/escmanager.log for more information about the error messages.

Keepalived Service State on Both HA VM Instances Stay in Backup State

Problem :

ESC HA has four different states: Active, Backup, Fault, and Stop. The Backup state is a transit state between Stop to Active, or Fault to Active. It is probable that both ESC VMs stick to the Backup state but usually do not last for a longer period. If you observe that the keepalived state on both ESC HA VMs is in a Backup state for more than two minutes, it could be a problem. However, there is a possibility of VRRP broadcast interference in your network.

Solution :

Run the following commands in any of your ESC VM to diagnose this problem:

```
$ sudo tcpdump -vvv -n -i ethX host ff02::12 (for IPv6 network)
$ sudo tcpdump -vvv -n -i ethX host 224.0.0.18 (for IPv4 Network)
```

The previous tcpdump commands listens to the VRRP broadcast packets in your ESC's heartbeat network. Use your heartbeat network interface to replace the ethX in the previous commands. For example, eth0. It provides you the information that whether your ESC VM is able to listen to the VRRP broadcast generated by any node in the subnet and you will find out who is doing the VRRP broadcasting in your network. For example:

```
# sudo tcpdump -vvv -n -i eth0 host 224.0.0.18
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
21:40:37.269728 IP (tos 0xc0, ttl 255, id 16606, offset 0, flags [none], proto VRRP (112),
 length 40)
    152.16.3.76 > 224.0.0.18: vrrp 152.16.3.76 > 224.0.0.18: VRRPv2, Advertisement, vrid
78, prio 101, authtype none, intvl 5s, length 20, addrs: 152.16.3.78
21:40:37.271332 IP (tos 0xc0, ttl 255, id 63866, offset 0, flags [none], proto VRRP (112),
 length 40)
    152.16.7.228 > 224.0.0.18: vrrp 152.16.7.228 > 224.0.0.18: VRRPv2, Advertisement, vrid
 230, prio 101, authtype none, intvl 5s, length 20, addrs: 152.16.7.230
21:40:38.269976 IP (tos 0xc0, ttl 255, id 49799, offset 0, flags [none], proto VRRP (112),
 length 40)
    152.16.3.61 > 224.0.0.18: vrrp 152.16.3.61 > 224.0.0.18: VRRPv2, Advertisement, vrid
74, prio 101, authtype none, intvl 5s, length 20, addrs: 152.16.3.74
21:40:39.271020 IP (tos 0xc0, ttl 255, id 20946, offset 0, flags [none], proto VRRP (112),
 length 40)
    152.16.1.195 > 224.0.0.18: vrrp 152.16.1.195 > 224.0.0.18: VRRPv2, Advertisement, vrid
 193, prio 101, authtype none, intvl 5s, length 20, addrs: 152.16.1.193
21:40:42.270541 IP (tos 0xc0, ttl 255, id 16607, offset 0, flags [none], proto VRRP (112),
 length 40)
```

Solution :

Ensure that no other VM or machine is doing the broadcasting with the same VRID as your ESC HA configuration. Otherwise, it will cause the interferences to your ESC HA heartbeat thereby causing both the ESC HA VMs to stay in Backup state. Run the following command to find the VRID value of your ESC HA:

```
$ cat /etc/keepalived/keepalived.conf | grep virtual_router_id
```

If you find that the VRID of your ESC HA is used by other systems in your subnet, specify a value of `--kad_vri` in your bootvm.py argument.

Cisco Elastic Services Controller HA is Running Slow

Problem :

In some OpenStack environment, the neutron configuration is different, the network throughput is extremely slow. In such cases, ESC VM needs to reduce the MTU for network interfaces from 1500 to 1450.

Note The MTU value for ESC's network interface should match the MTU for other network interfaces for VMs which manage components ESC directly communicates with, such as a VIM, an NFVO or an administrative jump box.

Solution :

Use the following steps to reduce the MTU value:

- Identify the interface interface you want to change and then go to the `/etc/sysconfig/network-scripts/ifcfg-ethX`. X represents the interface number you want to change.
- Use a text editor like VIM to add or edit the MTU items.

  ```
  mtu=1450
  ```

- Use the following command to restart the network interface:

  ```
  # network service restart
  i.e: sudo ifdown eth0 && sudo ifup eth0
  ```

Unable to Access Cisco Elastic Services Controller HA with the VIP

Ensure that your VIP is in allowed_address_pairs of the ports of ESC instances.

Before you begin

Problem 1:

Unable to access ESC HA with the VIP

ESC VIP floats across ESC HA instances and it redirects the connection to ESC Master.

Verification and Troubleshooting:

Check the following two items if your VIP does not work in the OpenStack environment:

- You must assign the VIP as allowed address pair to the original interface of ESC instances.
- Check the port of your ESC's interfaces and ensure that the allowed address pair configurations are correct.

Procedure

Step 1 Find the port UUID of your ESC interface for VIP Failover. In the following example, `152.16.3.76` is the IP:

```
$ neutron port-list | grep 152.16.3.76
| 80d7e031-04cd-4fb7-8f48-dcbcdbbd8685 |     | fa:16:3e:87:c9:e5 | {"subnet_id":
"7b2ce63b-eb20-4ff8-8d49-e46ee8dde0f5", "ip_address": "152.16.3.76"}
```

Step 2 Check the allowed address pairs for the port and add the VIP to the allowed address pair of the port.

For example:

```
$ neutron port-show 80d7e031-04cd-4fb7-8f48-dcbcdbbd8685
+----------------------+-------------------------------------------------------------------------------------+
| Field                | Value
                    |
+----------------------+-------------------------------------------------------------------------------------+
| admin_state_up       | True
                    |
| allowed_address_pairs |
                    |
| binding:host_id      | my-ucs-64
                    |
| binding:profile      | {}
                    |
| binding:vif_details  | {"port_filter": true, "ovs_hybrid_plug": false}
                    |
| binding:vif_type     | ovs
                    |
| binding:vnic_type    | normal
                    |
| created_at           | 2017-12-13T21:16:56
                    |
| description          |
                    |
| device_id            | b895cd19-2491-4ac0-b4b5-087a4f76b701
                    |
| device_owner         | compute:None
                    |
| extra_dhcp_opts      |
                    |
| fixed_ips            | {"subnet_id": "7b2ce63b-eb20-4ff8-8d49-e46ee8dde0f5", "ip_address":
 "152.16.3.76"} |
| id                   | 80d7e031-04cd-4fb7-8f48-dcbcdbbd8685
                    |
| mac_address          | fa:16:3e:87:c9:e5
                    |
| name                 |
                    |
| network_id           | c7fafeca-aa53-4349-9b60-1f4b92605420
                    |
| port_security_enabled | True
                    |
```

```
| security_groups       | e8e9e10c-0e73-4e01-b364-115f785f787d
                  |
| status                | ACTIVE
                  |
| tenant_id             | d972982b511d4caa973f2ab71b58c2fe
                  |
| updated_at            | 2017-12-13T21:17:20
                  |
+---------------------+-------------------------------------------------------------------------------+


$ neutron port-update <your_esc_port_id> --allowed-address-pairs type=dict list=true
ip_address=<your_vip_address>
For Example:
$ neutron port-update 80d7e031-04cd-4fb7-8f48-dcbcdbbd8685 --allowed-address-pairs type=dict
 list=true ip_address=152.16.3.78
Updated port: 80d7e031-04cd-4fb7-8f48-dcbcdbbd8685

$ neutron port-show 80d7e031-04cd-4fb7-8f48-dcbcdbbd8685
+---------------------+-------------------------------------------------------------------------------+
| Field                 | Value
                  |
+---------------------+-------------------------------------------------------------------------------+
| admin_state_up        | True
                  |
| allowed_address_pairs | {"ip_address": "152.16.3.78", "mac_address": "fa:16:3e:87:c9:e5"}
                |
| binding:host_id       | my-ucs-64
                  |
| binding:profile       | {}
                  |
| binding:vif_details   | {"port_filter": true, "ovs_hybrid_plug": false}
                  |
| binding:vif_type      | ovs
                  |
| binding:vnic_type     | normal
                  |
| created_at            | 2017-12-13T21:16:56
                  |
| description           |
                  |
| device_id             | b895cd19-2491-4ac0-b4b5-087a4f76b701
                  |
| device_owner          | compute:None
                  |
| extra_dhcp_opts       |
                  |
| fixed_ips           | {"subnet_id": "7b2ce63b-eb20-4ff8-8d49-e46ee8dde0f5", "ip_address":
 "152.16.3.76"} |
| id                    | 80d7e031-04cd-4fb7-8f48-dcbcdbbd8685
                  |
| mac_address           | fa:16:3e:87:c9:e5
                  |
| name                  |
                  |
| network_id            | c7fafeca-aa53-4349-9b60-1f4b92605420
                  |
| port_security_enabled | True
                  |
| security_groups       | e8e9e10c-0e73-4e01-b364-115f785f787d
                  |
| status                | ACTIVE
                  |
| tenant_id             | d972982b511d4caa973f2ab71b58c2fe
```

```
                    |
| updated_at             | 2018-01-29T21:35:17
                    |
+---------------------+-------------------------------------------------------------------------------------+
```

What to do next

Other VM takes over the VIP IP address:

In such scenarios, you must find out who took the VIP IP address. Once you know that, you release the IP address or select another IP address for your HA VIP. To ensure that the VIP you are using is safe and no one takes over it, you can create a port to occupy the VIP. To reserve the VIP address, run the following command:

```
$ neutron port-create <network_name> --fixed-ip ip_address=<your_vip_address> --name kad-vip


For example:

$ neutron port-create esc-net --fixed-ip ip_address=152.16.3.78 --name kad-vip
Created a new port:
+---------------------+-------------------------------------------------------------------------------------+
| Field                  | Value
                    |
+---------------------+-------------------------------------------------------------------------------------+
| admin_state_up         | True
                    |
| allowed_address_pairs |
                    |
| binding:host_id        |
                    |
| binding:profile        | {}
                    |
| binding:vif_details    | {}
                    |
| binding:vif_type       | unbound
                    |
| binding:vnic_type      | normal
                    |
| created_at             | 2018-01-29T21:53:33
                    |
| description            |
                    |
| device_id              |
                    |
| device_owner           |
                    |
| extra_dhcp_opts        |
                    |
| fixed_ips              | {"subnet_id": "7b2ce63b-eb20-4ff8-8d49-e46ee8dde0f5", "ip_address":
 "152.16.3.78"} |
| id                     | 3c037a4b-4245-4554-adf5-56ca6bbffa98
                    |
| mac_address            | fa:16:3e:4e:f2:96
                    |
| name                   | kad-vip
                      |
| network_id             | c7fafeca-aa53-4349-9b60-1f4b92605420
                    |
| port_security_enabled | True
                    |
| security_groups        | e8e9e10c-0e73-4e01-b364-115f785f787d
```

```
                     |
| status               | DOWN
                     |
| tenant_id            | d972982b511d4caa973f2ab71b58c2fe
                     |
| updated_at           | 2018-01-29T21:53:33
                     |
+--------------------+-------------------------------------------------------------------------------+
```

VIP is in a different network than the management network:

ESC HA configuration provides the following three configuration parameters (bootvm.py arguments):

- **`--ha_node_list`**: The list of IP addresses for HA nodes in the Active/Standby cluster. For ESC nodes with multiple network interfaces, these IPs should be the addresses in the network used for data synchronization. This argument is utilized for replication-based HA solution only. For example:

  ```
  --ha_node_list 192.168.0.12 192.168.0.22
  ```

- **`--kad_vip`** : The IP address for keepalived VIP (virtual IP) and the interface for keepalived VIP (ESC 2.2). For example:

  ```
  -kad_vip 10.20.0.194
  ```

  From ESC 2.2, the interface of VIP is specified in the following format:

  ```
  --kad_vip 10.20.0.194:eth2 or --kad_vip [2001:cc0:2020::fc]:eth2;
  ```

- **`--kad_vif`**:
  - The interface for keepalived VRRP and VIP (ESC 1.0 ~ ESC 2.1).
  - The interface for keepalived VRRP only if the VIP interface is specified in kad_vip (ESC 2.2). For example:

    ```
    --kad_vif eth0
    ```

    Use the VIP in a different interface than where network/interface of synchronization interface (kad_vif), --ha_node_list, and --kad_vif should be configured in one network/interface (eth1) and the --kad_vip in another network/interface (eth0).

    For example, for following bootvm.py commands, ESC HA uses eth1 (192.168.0.0/24) for data synchronization and heartbeat and uses eth0 (192.168.5.0/24) for VIP access. The VIP 192.168.5.200 floats between ESC nodes in the network (192.168.5.0/24).

    ```
    ./bootvm.py esc-ha-1 --image ESC-2_2_8_106 --net lab-net-0 esc-net --gateway_ip
    192.168.0.1 --ipaddr 192.168.5.239 192.168.0.239 --ha_node_list 192.168.0.239
    192.168.0.243 --kad_vip 192.168.5.200/24:eth0 --kad_vif eth1 --ha_mode drbd --route
     10.85.103.0/24:192.168.0.1:eth1 --avail_zone nova:my-ucs-26
    ./bootvm.py esc-ha-0 --image ESC-2_2_8_106 --net lab-net-0 esc-net --gateway_ip
    192.168.0.1 --ipaddr 192.168.5.243 192.168.0.243 --ha_node_list 192.168.0.239
    192.168.0.243 --kad_vip 192.168.5.200/24:eth0 --kad_vif eth1 --ha_mode drbd --route
     10.85.103.0/24:192.168.0.1:eth1 --avail_zone nova:my-ucs-27
    ```

Status Check in Active VM Not Displaying the Status of BACKUP VM

The heartbeat of ESC HA is based on VRRP protocol. Based on VRRP protocol, ESC Active VM does not know the status of Backup VM instance. Hence, the status check also does not include the Backup VM status because the ESC service works fine as long as Active VM is working.

If you want to check the status of Backup VM, in ESC Active VM, run the following command:

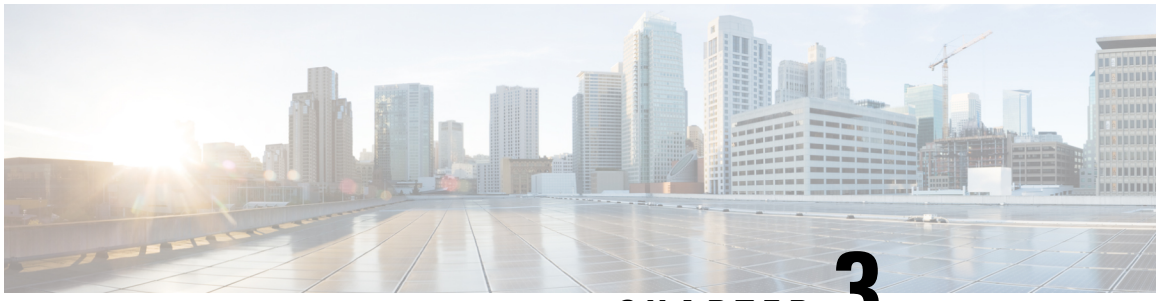
```
$ sudo cat /proc/drbd
version: 8.4.10-1 (api:1/proto:86-101)
GIT-hash: a4d5de01fffd7e4cde48a080e2c686f9e8cebf4c build by abcbuild@, 2017-09-15 14:23:22
 1: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r-----
    ns:5883476 nr:3012 dw:5886500 dr:378689 al:26 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:f oos:0
```

Ensure that the `ro` shows `Primary/Secondary` and the `ds` shows `UpToDate/UpToDate`. It means that your Backup is connected to the Active VM and synchronization between Active and Backup is good. The following example shows when your Backup VM gets disconnected:

```
$ sudo cat /proc/drbd
version: 8.4.10-1 (api:1/proto:86-101)
GIT-hash: a4d5de01fffd7e4cde48a080e2c686f9e8cebf4c build by abcbuild@, 2017-09-15 14:23:22
 1: cs:WFConnection ro:Primary/Unknown ds:UpToDate/DUnknown C r-----
    ns:5888880 nr:3012 dw:5891912 dr:378689 al:26 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:f oos:84
```

PART III

Troubleshooting Cisco Elastic Services Controller Micro-Services



CHAPTER 3

Troubleshooting Cisco Elastic Services Controller Micro-Services

ESC is designed based on Micro-Service software architecture and there are many components or applications integrated together in a micro-service software architecture to provide vendor-agnostic, customizable and programmable platform.

Overview of Cisco Elastic Services Controller Micro-Services

ESC is based on Micro-Service software architecture which provide various services to its vendors. To ensure that all micro-services are running in healthy state. Following are the methods to check the health condition of ESC's micro-services and the overall service status:

To check ESC's overall status, run the following command in ESC VM or the Active VM of ESC HA.

The first line of the output shows the overall status of ESC service and the following lines indicate the status of each micro-service or component.

```
# escadm status --v
0 ESC status=0 ESC HA Master Healthy
vimmanager (pgid 6432) is running
monitor (pgid 6450) is running
mona (pgid 6529) is running
drbd (pgid 0) is master
confd (pgid 6656) is running
keepalived (pgid 1374) is running
pgsql (pgid 6760) is running
filesystem (pgid 0) is running
snmp (pgid 6598) is running
escmanager (pgid 6924) is running
```

Cisco Elastic Services Controller status is Not Healthy

Problem :

In some scenarios, when you check your ESC status, the output of ESC health status displays that ESC is in critical error status and probably one or some of the ESC's component or micro-service are in stopped/dead or not running state. For example:

```
# escadm status --v
2 ESC status=0 ESC HA Master Critical
vimmanager (pgid 6432) is running
monitor (pgid 6450) is running
mona is stopped
drbd (pgid 0) is master
confd (pgid 6656) is running
keepalived (pgid 1374) is running
pgsql (pgid 6760) is running
filesystem (pgid 0) is running
snmp (pgid 6598) is running
escmanager (pgid 6924) is running
```

Solution :

Do the following actions to get the ESC service back.

- Restart ESC service

  Following commands help you to restart ESC services in a Standalone ESC:

  For ESC 2.X:

  ```
  $ sudo service esc_service stop
  $ sudo service esc_service status (make sure ESC service is stopped)
  $ sudo service esc_service start
  ```

  For ESC 3.X and later releases:

  ```
  $ sudo escadm stop
  $ sudo escadm status (make sure ESC service is stopped)
  $ sudo escadm start
  ```

  For ESC HA, run the following commands to restart ESC service.

  For ESC 2.X:

  ```
  $ sudo service keepalived stop
  $ sudo service keepalived status (make sure ESC service is stopped)
  $ sudo service keepalived start
  ```

  For ESC 3.X and later releases:

  ```
  $ sudo escadm stop
  $ sudo escadm status (make sure ESC service is stopped)
  $ sudo escadm start
  ```

  Note that ESC HA failover will be triggered when you restart the ESC service. The BACKUP VM will switch running as the HA MASTER node after execution of the above mentioned commands. If you do not want to trigger the switchover, the two extra steps mentioned below should be taken care.

Note ESC HA failover triggers when you restart the ESC service. The BACKUP VM switches and starts running as the HA Active VM when you execute the previous commands.

Use the following steps if you do not want to trigger the switchover:

Before you execute the service restart commands in Active VM, login to the Backup VM first and run the following commands:

For ESC 2.X:

```
$ sudo service keepalived stop
$ sudo service keepalived status (make sure ESC service is stopped)
```

For ESC 3.X and later releases:

```
$ sudo escadm stop
$ sudo escadm status (make sure ESC service is stopped)
```

Once you restart the ESC service in the Active VM, log into the Backup VM again and run the following commands:

For ESC 2.X:

```
$ sudo service keepalived start
```

For ESC 3.X and later releases:

```
$ sudo escadm start
```

- Reboot ESC VM

  If restart ESC service still doesn't help, run the following command to reboot the ESC VM:

  ```
  $ sudo reboot
  ```

Note ESC HA switchover triggers when you reboot the ESC Active VM. The Backup VM becomes the new Active and start all the services.

- Check ESC's start up logs.

If ESC service still gets stuck at the startup, check the ESC logs to find out the details. You must verify the following logs files:

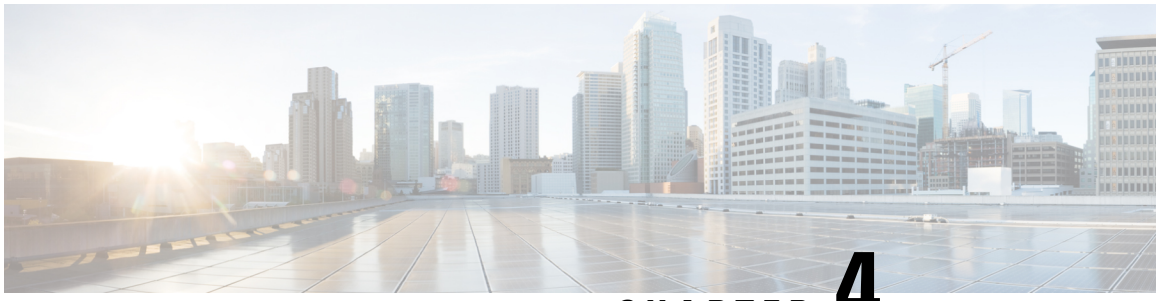
- `/var/log/esc/escadm.log` -The log of ESC service start/stop to check which micro-service causes the problem.
- `/var/log/esc/escmanager.log` -The log of ESCManager about ESCManager service start/stop.
- `/var/log/messages` -The OS message logging file also contains fatal startup errors at the system level.

If you cannot find the problem, collect the ESC logs and send the log files from ESC VMs (two VMs in HA) to the technical support team. To collect the logs, use the following command:

```
$ sudo escadm log collect
```

PART IV

Troubleshooting Cisco Elastic Services Controller Upgrades



CHAPTER 4

Troubleshooting Cisco Elastic Services Controller Upgrades

Troubleshooting Cisco Elastic Services Controller Upgrades

Problem Statement:

How can I choose ESC upgrade approach?

Description:

There are three upgrade approaches for ESC Upgrade and following are the criteria for your reference to choose the best approach for upgrading your ESC setup.

Solution:

- If your upgrade path is between ESC patch builds, for example, from ESC 3.1.0.116 to ESC 3.1.0.145, consider RPM upgrade as your first choice.
- If your upgrade path is between ESC official FCS release, for example, from ESC 3.0.0 to ESC 3.1.0, RPM upgrade will not work in your situation and you can only do the image upgrade.
    - For a healthy ESC HA, consider in-service upgrade.
    - For Standalone ESC or ESC HA with only Master VM, take the database backup/restore approach.

Rolling Back Cisco Elastic Services Controller Upgrade

Problem Statement:

Upgrade does not go well, and you want to roll back.

Description:

There are three upgrade approaches for ESC Upgrade and following are the criteria for your reference to select the best approach for upgrading your ESC setup.

Solution:

The following are the two important points for ESC installation and upgrade:

- Once you have successfully installed ESC or ESC HA, keep the complete bootvm.py command line in your record. You can use the bootvm.py command line next time when you take an image upgrade approach to perform the ESC upgrade.
- Ensure you take the backup of ESC database while doing ESC upgrade. Before backing up the database, stop ESC services and wait until the services are in stop state.

  Example:

  ```
  #ESC backup DB ESC 2.X:
  $ sudo /opt/cisco/esc/esc-scripts/esc_dbtool.py backup --file
  scp://<dest_user>:<dest_password>@<dest_host>:/tmp/db.tar.bz2

  #ESC backup DB ESC 3.X and up:
  sudo escadm backup --file /tmp/db.tar.bz
  scp /tmp/db.tar.bz <dest_user>@<dest_host>:/tmp/db.tar.bz
  ```

- Once you successfully take the backup of the database, it is easy to roll back the upgrade of ESC, if any problem arises during or after the ESC upgrade procedure.

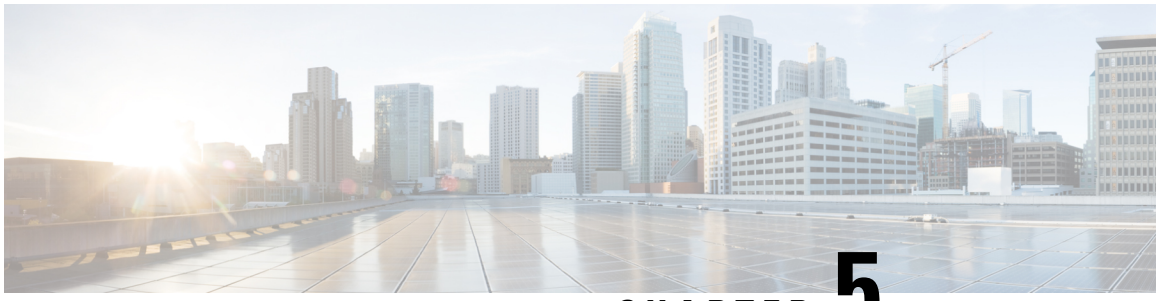
If you have the bootvm.py commands in record with the database backup file, you can take the following approaches to roll back your ESC upgrade:

- If you are performing the in-service upgrade and have not deleted the old Primary ESC VM (or just renamed the old Primary ESC VM), you can delete the upgraded VM instances first and then turn on the old (renamed) Primary ESC. In this case, the database is retained with no changes in the old Primary VM. Re-install the Backup ESC VM using the complete bootvm.py command line from your records with the old released image. The redeployed Backup VM automatically syncs its database from the Primary VM. No database restore is required.
- If you have deleted both the older version of Primary and Backup ESC VMs, or you are performing a DB backup/restore upgrade for a standalone ESC, redeploy the ESC VM using the complete bootvm.py command line from your records with the old release image. Then follow the database restore guide to restore the database for the rollback. Before restoring the DB, stop ESC services and wait till the services are in a stop state.

  Example:

  ```
  #ESC Restore DB ESC 2.X:
  $ sudo /opt/cisco/esc/esc-scripts/esc_dbtool.py restore --file
  scp://<dest_user>:<dest_password>@<dest_host>:/tmp/db.tar.bz2

  #ESC Restore DB ESC 3.X and up:
  scp <src_user>@<src_host>:/tmp/db.tar.bz /tmp/db.tar.bz
  sudo escadm restore --file /tmp/db.tar.bz
  ```



CHAPTER 5

Troubleshooting Cisco Elastic Services Controller Backup and Restore

Corrupt Database Backup File

Problem Statement:

I have a corrupted database backup file.

Solution:

Once you run the database backup command, you get a tar file which includes all the database backup files. To ensure that the DB backup file is not corrupted, run the following command to do the verification and make sure the echo command output is 0.

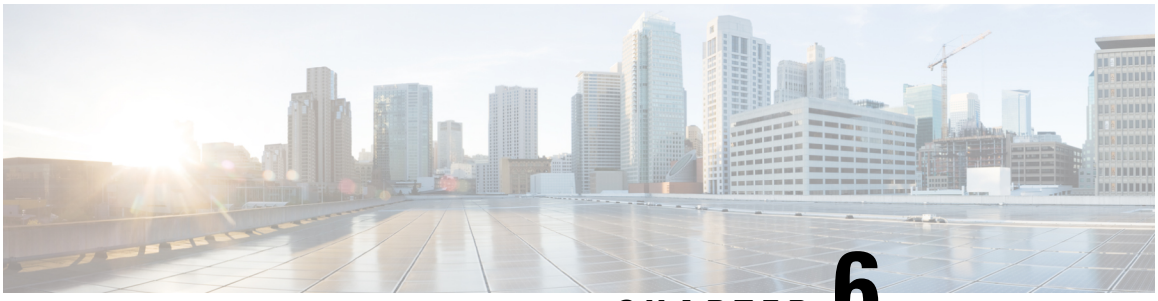
```
$tar -tvf <your_backup_tar_file>
$echo $?
0
```

To get a to copy the backup tar file to other VMs, use checksum approach to make sure the data is integrated. For example:

```
$ sha256sum /tmp/db.tar.bz
d4a831983d0cafddf1c734eed5ad7f39904f948f86ffd64c675107d94ca15a4f  /tmp/db.tar.bz
```

PART V

Troubleshooting ConfD and NETCONF API



CHAPTER 6

Troubleshooting ConfD and NETCONF API

Troubleshooting ConfD and NETCONF API

For security reasons, by default, ESC disables ConfD's developer log and netconf trace log.

Use the following two logs for debugging.

To manually enable these two logs, on an ESC VM, open the following ConfD's configuration file:

```
$ sudo vim /opt/cisco/esc/esc_database/esc_production_confd.conf
```

Change the following manually in the following log:

enabled → true.

```
<netconfTraceLog>
    <enabled>true</enabled>
    <filename>/var/log/esc/confd/netconf.trace</filename>
    <format>pretty</format>
</netconfTraceLog>
```

Change this section to enable ***developers log***:

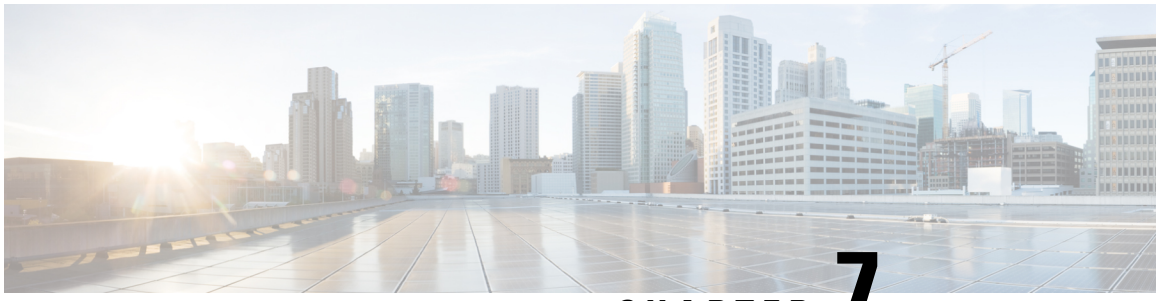
Change both enabled and file.enabled → true.

```
<developerLog>
    <enabled>true</enabled>
    <file>
        <enabled>true</enabled>
        <name>/var/log/esc/confd/devel.log</name>
    </file>
    <syslog>
        <enabled>false</enabled>
    </syslog>
</developerLog>
```

Troubleshooting ConfD and NETCONF API

PART VI

Troubleshooting VNF Deployment



CHAPTER 7

Troubleshooting VNF Deployment

Overview

This guide provides step by step instructions on how to troubleshoot issues among the VNFs deployed and managed by ESC.

Logs for Troubleshooting

Before performing ESC troubleshooting for VNF deployment, the first step is to collect and check logs. To collect ESC logs:

For ESC Release 2.3.2:

```
sudo /opt/cisco/esc/esc-scripts/collect_esc_log.sh
```

For ESC Release 3.0 and later:

```
sudo escadm log collect
```

There are several important logs containing useful error messages.

- YangESC log contains the incoming request and notification:

```
/var/log/esc/yangesc.log
```

- ESCManager log contains the ESC processing details:

  ```
  /var/log/esc/escmanager.log
  ```

- VimManager log contains the VimManager processing details:

  ```
  /var/log/esc/vimmanager/vimmanager.log
  ```

- Vim_VimManager log contains raw request/response between VimManager and VIM

  ```
  /var/log/esc/vimmanager/vim_vimmanager.log
  ```

- Mona log conatins monitoring procesing defails plus script executions

  ```
  /var/log/esc/mona/mona.log
  ```

VNF VM Deployed Failed with VIM Related Error

Problem

When you received the VM_DEPLOYED notification (thru Netconf, REST, Portal) with a non-200 status code means the deployment failed due to an error.

Notification found in /var/log/esc/yangesc.log

```
02:19:25,758 23-Jan-2018 WARN  ===== SEND NOTIFICATION STARTS =====
02:19:25,758 23-Jan-2018 WARN  Type: VM_DEPLOYED
02:19:25,758 23-Jan-2018 WARN  Status: FAILURE
02:19:25,758 23-Jan-2018 WARN  Status Code: 500
02:19:25,759 23-Jan-2018 WARN  Status Msg: VIM Driver: Exception while creating VM: Error
creating VM from template, the host [10.67.103.255] does not exist
02:19:25,759 23-Jan-2018 WARN  Tenant: admin
02:19:25,759 23-Jan-2018 WARN  Deployment ID: 169384d7-c67b-40a4-bcaa-dd3294305ba3
02:19:25,759 23-Jan-2018 WARN  Deployment name:
Vmware-NetConf-Intra-Affinity-Anti-Affinity-With-InvalidCluster-InvalidHost
02:19:25,759 23-Jan-2018 WARN  VM group name:
Group2-uLinux-Intra-Anti-Affinity-With-InvalidHost
02:19:25,759 23-Jan-2018 WARN  User configs: 1
02:19:25,759 23-Jan-2018 WARN      VM Name:
Vmware-NetConf-I_Group2_0_65aa6ca8-3b53-4eb3-a39f-a3f12394a190
02:19:25,759 23-Jan-2018 WARN      Host ID:
02:19:25,759 23-Jan-2018 WARN      Host Name:
02:19:25,760 23-Jan-2018 WARN  =====  SEND NOTIFICATION ENDS  =====
```

Solution

The reason of the failure is within the status message itself. The previous example shows that the targeted host for deployment does not exist. Following is another example of a failed VM deployment:

Notification found in /var/log/esc/yangesc.log

```
07:20:56,164 25-Jan-2018 WARN  Status Msg: Failed to create VM ports for instance :
jenkins-ErrHandl_ErrorG_2_cc0f8c28-8900-4977-90d9-b9f996c8ca71. Create port operation failed:
 Exception during processing: com.cisco.esc.vimmanager.exceptions.CreatePortException:
Create port failed: ClientResponseException{message=No more IP addresses available on network
 0b7965b4-c604-444c-8cbb-7c2399e912d4., status=409, status-code=CONFLICT}.
```

The previous example shows a deployment failure message containing direct response from VIM that has an error message and status code. In case of deployment failure due to role access issues such as these VIM

related issues, perform the required action on the VIM instance, or adjust ESC deployment datamodel with proper configurations. Here are some common VIM related issues:

1. Out of quota errors
   a. Either delete some resource in question via ESC under that tenant/project/user or,
   b. Configure your VIMs resource limit per tenant/project/user.
2. Already in use errors
   a. Change resource name/configuration or sometimes there are restrictions on the VIM disallow and is sometimes configurable.
   b. Delete the resource in question.

VNF VM Deployed Failed with LCM

Problem

If the VNF deployment datamodel involves LCM action (for example, a staging script), the deployment may have failed because the action failed to complete. In this case, you get the the following error message in */var/log/esc/escmanager.log*:

```
22:12:11,912 25-Jan-2018
VM_STATE_MACHINE-ab-auto-test-vnf_ab-aut_0_31ebad33-e12f-4772-a89c-3bdc239acf69 ERROR
[StateMachineCloudUtils.java:setupPersonalities():1081]
[tid=fffae7af-a321-4ea5-a1bc-3b30c903f3a5]
com.cisco.esc.datamodel.exceptions.ESCException: Action [GEN_VPC_ISO] failed
    at
com.cisco.esc.statemachines.utils.StateMachineCloudUtils.setupPersonalities(StateMachineCloudUtils.java:1069)
```

Description

To find out the details of a staging script failure, look for some entries like the following in the */var/log/esc/mona/mona.log.* log:

/var/log/esc/mona/mona.log

```
2018-01-25 19:34:45.751 [http-nio-127.0.0.1-8090-exec-5] Script:
[/opt/cisco/esc/esc-scripts/esc_volume_em_staging.sh] execution in progress
2018-01-25 19:34:45.751 [http-nio-127.0.0.1-8090-exec-5] Use the original script path and
skip downloading: no protocol: /opt/cisco/esc/esc-scripts/esc_volume_em_staging.sh
2018-01-25 19:49:45.772 [http-nio-127.0.0.1-8090-exec-5] Script execution failed, timer
expired for script: /opt/cisco/esc/esc-scripts/esc_volume_em_staging.sh
2018-01-25 19:49:45.805 [http-nio-127.0.0.1-8090-exec-5] Script execution failed
com.cisco.esc.mona.exceptions.ActionExecutionException: Script execution failed, timer
expired for script:/opt/cisco/esc/esc-scripts/esc_volume_em_staging.sh
```

Solution

Some common errors can be a permission issue, or script execution timed out. Do a dry run for the script on the ESC VM to ensure it works.

VNF VM Deployed Failed Due to Role Access Issue (ESC Release 3.1 and Later)

Problem

When deploying VNF on OpenStack as a non-admin user, deployment may encounter role access errors such as:

```
02:19:25,758 23-Jan-2018 WARN  ===== SEND NOTIFICATION STARTS =====
02:19:25,758 23-Jan-2018 WARN  Type: VM_DEPLOYED
02:19:25,758 23-Jan-2018 WARN  Status: FAILURE
02:19:25,758 23-Jan-2018 WARN  Status Code: 500
02:19:25,759 23-Jan-2018 WARN  Status Msg: VIM Driver: Exception while creating VM:
{"message": "You are not authorized to perform the requested action: identity:create_project",
 "code": 403, "title": "Forbidden"}}
02:19:25,759 23-Jan-2018 WARN  Tenant: admin
02:19:25,759 23-Jan-2018 WARN  Deployment ID: 169384d7-c67b-40a4-bcaa-dd3294305ba3
02:19:25,759 23-Jan-2018 WARN  Deployment name:
Vmware-NetConf-Intra-Affinity-Anti-Affinity-With-InvalidCluster-InvalidHost
02:19:25,759 23-Jan-2018 WARN  VM group name:
Group2-uLinux-Intra-Anti-Affinity-With-InvalidHost
02:19:25,759 23-Jan-2018 WARN  User configs: 1
02:19:25,759 23-Jan-2018 WARN      VM Name:
Vmware-NetConf-I_Group2_0_65aa6ca8-3b53-4eb3-a39f-a3f12394a190
02:19:25,759 23-Jan-2018 WARN      Host ID:
02:19:25,759 23-Jan-2018 WARN      Host Name:
02:19:25,760 23-Jan-2018 WARN  =====  SEND NOTIFICATION ENDS  =====
```

Solution

ESC Release 3.1 and later requires two permissions granted in Neutron

```
create_port:fixed_ips
create_port:mac_address
```

1. Create a new role on OpenStack for ESC. Go to the OpenStack Horizon (Identity -> Roles), and create a new role with the name **vnfm** or any other name you want.
2. Assign the user with the vnfm role to a project managed by ESC on OpenStack Horizon (Identity → Projects). Click **Manage members** and make sure the user for ESC has the vnfm role.
3. Modify the following items in OpenStack controller by adding 'or role:vnfm' in the default values. Changes to the `policy.json` file becomes effective immediately and does not require service restart.

/etc/neutron/policy.json	"create_port:fixed_ips": "rule:context_is_advsvc or rule:admin_or_network_owner",	"create_port:fixed_ips": "rule:context_is_advsvc or rule:admin_or_network_owner or role:vnfm",
/etc/neutron/policy.json	"create_port:mac_address": "rule:context_is_advsvc or rule:admin_or_network_owner"	"create_port:mac_address": "rule:context_is_advsvc or rule:admin_or_network_owner or role:vnfm"

VNF VM Deployed But Goes Into a Boot Loop

Problem

Consider the VNF is deployed, the VM_DEPLOYED notification with a 200 status code is received, but VM_ALIVE is not yet received. While checking the VNF's console through VIM UI (for example, OpenStack Horizon, VMware vCenter), the VNF goes into a reboot cycle/loop. Most of these cases are a symptom of a failure day-0 data passed. To verify the day 0 data passed in, for OpenStack, check the */var/log/esc/vimmanger/vim_vimmanager.log*, looking for the POST request sent to OpenStack for creating a server.

```
2018-01-26 16:02:55.648 INFO  os -   1 * Sending client request on thread
http-nio-127.0.0.1-8095-exec-4
1 > POST http://ocata1-external-controller:8774/v2/d6aee06abdbe42edaade348280199a64/servers
1 > Accept: application/json
1 > Content-Type: application/json
1 > User-Agent: OpenStack4j / OpenStack Client
1 > X-Auth-Token: ***masked***
{
  "server" : {
    "name" : "jenkins-jenkinsy_MAKULA_0_bbc61ba6-6c63-4fb9-b9cd-5ae92a898943",
    "imageRef" : "67fc9890-230e-406c-bd01-f2e1ffa2437f",
    "flavorRef" : "cc12dec2-411a-46bd-b8c2-4ff8738ddb02",
    "personality" : [ {
      "path" : "iosxe_config.txt",
      "contents" :
[redacted]

    } ],
    "config_drive" : true,
    "networks" : [ {
      "port" : "01b3b168-8fab-4da4-b195-f9652d36674e"
    }, {
      "port" : "dfe709c9-4ca4-400f-a765-2dbc88828585"
    } ],
    "block_device_mapping_v2" : [ {
      "source_type" : "image",
      "destination_type" : "local",
      "uuid" : "67fc9890-230e-406c-bd01-f2e1ffa2437f",
      "boot_index" : 0,
      "delete_on_termination" : true
    } ]
  }
}
```

Solution

Decode the base64 encoded string value of the personality content.

```
hostname csr
!
platform console serial
!
ip subnet-zero
no ip domain-lookup
ip domain name cisco.com
!
enable password cisco123
username admin password cisco123
username admin privilege 15
!
```

```
interface GigabitEthernet1
 description management network
 ip address dhcp
 no shut
!
interface GigabitEthernet2
 description service network
 ip address dhcp
 no shut
!
interface GigabitEthernet3
 description service network
 ip address dhcp
 no shut
!
crypto key generate rsa modulus 1024
ip ssh version 2
ip ssh authentication-retries 5
ip scp server enable
file prompt quiet
!
line con 0
 stopbits 1
line vty 0 4
 login local
 privilege level 15
 transport input ssh telnet
 transport output ssh telnet
!
snmp-server community public RO
!
end
```

Verify if the content contains correct day 0 configuration or not. If the day 0 config is passed through a volume, detach the volume from the VNF, and attach it to another VM to check its content.

For VMware, if the day 0 config is passed through the ovf settings, verify it from the vCenter:

1. Open VM settings.
2. Under Options, select 'OVF Settings'.
3. Click 'View' under OVF Enviroment.

If the day 0 config is passed through CDROM, you can verify through the ISO file that was mounted to the CDROM.

Download the ISO file from the specific datastore, then mount the ISO file locally to verify its content.

VNF VM Deployed But Never Goes to ALIVE State

Problem

If a VNF is deployed successfully, but never received a VM_ALIVE notification, then it is inferred that the ESC was unable to reach the newly deployed VNF. This is mostly because of an issue in the network. Check the KPI section of the VNF deployment datamodel:

```
<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_occurrences_true>1</metric_occurrences_true>
    <metric_occurrences_false>30</metric_occurrences_false>
    <metric_collector>
      <type>ICMPPing</type>
      <nicid>2</nicid>
      <poll_frequency>10</poll_frequency>
      <polling_unit>seconds</polling_unit>
      <continuous_alarm>false</continuous_alarm>
    </metric_collector>
  </kpi>
</kpi_data>
```

Solution

To verify if the VNF is alive, do an ICMP ping from ESC VM to the VNF using a particular IP indicated by,

```
<nicid>2</nicid>
```

where, nicid 2 refers to the IP of the inteface with nicid of 2 which ESC is about to ping, pointing to:

```
<interface>
  <nicid>2</nicid>
  <network>NVPGW100-UAS-uas-orchestration</network>
  <allowed_address_pairs>
    <address>
      <ip_address>172.168.11.0</ip_address>
      <netmask>255.255.255.0</netmask>
    </address>
  </allowed_address_pairs>
</interface>
```

Here 172.168.11.0 is the IP. Ensure that the interface shares the same network with ESC. In the previous example, the network is NVPGW100-UAS-uas-orchestration. If the ping fails, you can ping the the gateway or another IP available on the subnet to find out if the issue is with the network.

VNF Recovery Failed

The following are some common recovery issues:

1. Recovery behaviour not behaving as expected. ESC is not attempting a redeploy after a failed reboot.
   a. Ensure the recovery policy in the XML file is set to REBOOT_THEN_REDEPLOY, and not set to reboot only. Read the recovery documentation to understand the recovery options and expectations.
2. ESC only attempts the recovery once, or too many times
   a. Double check the config parameter "VM_RECOVERY_RETRIES_MAX", the default value is 3 times. To check this value, execute the REST call within the ESC VM.

      ```
      curl -H "accept: Application/json"
      http://127.0.0.1:8080/ESCManager/v0/config/default/VM_RECOVERY_RETRIES_MAX | python
       -mjson.tool
      ```

   b. If it is set correctly, ensure ESC was healthy at the time of recovery and that a switch over did not occur; tt may have continued the recovery attempt in the second ESC VM.

VNF VM Recovery Failing Due to Failure Reboot

Problem

VNF VM recovery has failed because of VM reboot failed event. This depends on the VM recovery policy definition:

```
<recovery_policy>
  <recovery_type>AUTO</recovery_type>
  <action_on_recovery>REBOOT_ONLY</action_on_recovery>
  <max_retries>3</max_retries>
</recovery_policy>
```

Description

ESC tries to reboot the VNF VM three times with non-success states before it has received the RECOVERY_COMPLETED event with an error state. The reboot operation also depends on two other system-wide configurations. parameters

```
VM_STATUS_POLLING_VM_OPERATION_RETRIES
VM_STATUS_POLLING_WAIT_TIME_SEC
```

After ESC asks VIM to reboot VM, it will keep polling VM status. The VM_STATUS_POLLING_VM_OPERATION_RETRIES defines how many times ESC tries to poll, and VM_STATUS_POLLING_WAIT_TIME_SEC defines how long does ESC wait between polls. The following are their default values:

```
VM_STATUS_POLLING_VM_OPERATION_RETRIES=10
VM_STATUS_POLLING_WAIT_TIME_SEC=5
```

Solution

If VNF VM takes more than 50 seconds to transition from REBOOT to ACTIVE state in OpenStack, then change the VM_STATUS_POLLING_WAIT_TIME_SEC to a higher number through the ESC REST API:

```
curl -X PUT -H "accept:application/json"
http://localhost:8080/ESCManager/v0/config/openstack/vm_status_polling_wait_time_sec/20 -k
 | python -mjson.tool
```

After receiving a success response, do a VM manual recovery again.

Get VNF VM Out of Error State

If a VNF VM is in an ERROR state in ESC, there are two options to transition it back to an ALIVE state considering that the external issue(s) that caused the ERROR state is resloved (for example, an issue on VIM). Before doing any of the following two options, ensure there is no ongoing operation performed on the same deployment. Check this in the */var/log/esc/yangesc.log.* Look for any previously initiated action without a completed notification (either success or failure). If any ongoing operation is found, wait for the operation to complete before performing the following actions:

Manual Recovery the VNF VM

Execute the following command:

```
/opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli recovery-vm-action DO {ESC generated VM Name}
```

Manual Unset/Set VM Monitoring (ESC Release 3.1 and above)

Execute the following command to unset the monitoring:

```
/opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli vm-action DISABLE_MONITOR {ESC generated VM
Name}
```

Then enable it again:

```
/opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli vm-action ENABLE_MONITOR {ESC generated VM Name}
```

Remove/Add VNF thru Service Update

Using Script to Prepare Datamodels

Please run the script on the ESC VM. The script generates two datamodel xml files, one for deleting the VM group(s), and the other one for adding the VM group(s) back.

```
[admin@abc-test-232 ~]$ ./genVMGroupDeletionDM.py -h
usage: genVMGroupDeletionDM.py [-h] vm_group_name [vm_group_name ...]

*************************************************************
Utility tool for generating VM group removing datamodel for ESC
Check the following wiki for details
https://confluence-eng-sjc1.cisco.com/conf/display/ESCWIKI/How+to+Use+Service+Update+to+Remove+a+VM+Group


positional arguments:
  vm_group_name  <Required> VM group name(s) separate by space

optional arguments:
  -h, --help     show this help message and exit
```

Example

```
[admin@abc-test-232 ~]$ ./genVMGroupDeletionDM.py g1 g2

Datamodel is generated:
[/home/admin/delete_g1_g2.xml]
[/home/admin/add_g1_g2.xml]
** Use on your own risk! **

[admin@abc-test-232 ~]$
```

Manual Prepare Datamodels

1. Get the current ESC datamodel

   ```
   /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli get esc_datamodel/tenants > {file name}
   ```

2. Remove the extra wrappers `<data>` and `<rpc-reply>` from the original file (also any cli outputs before the `<xml>` tag) from step 1. The end result will look like:

   Example Datamodel After Step #2

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
     <data>
       <esc_datamodel xmlns="http://www.cisco.com/esc/esc">
   ```

Note If the datamodel contains multiple deployments, ensure to keep the other deployments as is when the datamodel is being edited. Do not change any text formatting. Remove other deployment section from the datamodel completely. This ensures that when the service update happens, these deployment(s) will remain untouched. For example, if the VM group to be deleted is c3, when editing the datamodel, you can remove the deployment portion for EM from the datamodel.

3. The following example shows how to delete a VM group *c3*. First check if there is any `<placement_group>` or `<placement> policy(ies)` defined under `<policies>` which has VM group c3 involved. Mark the policies and delete them. There are 8 placement policies in the following example:

```
...
                <placement>
                  <target_vm_group_ref>c3</target_vm_group_ref>
                  <type>anti_affinity</type>
                  <enforcement>strict</enforcement>
                  <vm_group_ref>c1</vm_group_ref>
                  <vm_group_ref>s2</vm_group_ref>
                </placement>
                <placement>
                  <target_vm_group_ref>s10</target_vm_group_ref>
                  <type>anti_affinity</type>
                  <enforcement>strict</enforcement>
                  <vm_group_ref>c1</vm_group_ref>
                  <vm_group_ref>c3</vm_group_ref>
                  <vm_group_ref>s2</vm_group_ref>
                  <vm_group_ref>s4</vm_group_ref>
                  <vm_group_ref>s5</vm_group_ref>
                  <vm_group_ref>s6</vm_group_ref>
                  <vm_group_ref>s7</vm_group_ref>
                  <vm_group_ref>s8</vm_group_ref>
                  <vm_group_ref>s9</vm_group_ref>
                </placement>
...
                <placement>
                  <target_vm_group_ref>s4</target_vm_group_ref>
                  <type>anti_affinity</type>
                  <enforcement>strict</enforcement>
                  <vm_group_ref>c1</vm_group_ref>
                  <vm_group_ref>c3</vm_group_ref>
                  <vm_group_ref>s2</vm_group_ref>
                </placement>
                <placement>
                  <target_vm_group_ref>s5</target_vm_group_ref>
                  <type>anti_affinity</type>
                  <enforcement>strict</enforcement>
                  <vm_group_ref>c1</vm_group_ref>
                  <vm_group_ref nc:operation='delete'>c3</vm_group_ref>
                  <vm_group_ref>s2</vm_group_ref>
                  <vm_group_ref>s4</vm_group_ref>
                </placement>
                <placement>
                  <target_vm_group_ref>s6</target_vm_group_ref>
                  <type>anti_affinity</type>
                  <enforcement>strict</enforcement>
                  <vm_group_ref>c1</vm_group_ref>
                  <vm_group_ref>c3</vm_group_ref>
                  <vm_group_ref>s2</vm_group_ref>
                  <vm_group_ref>s4</vm_group_ref>
                  <vm_group_ref>s5</vm_group_ref>
```

```
</placement>
<placement>
  <target_vm_group_ref>s7</target_vm_group_ref>
  <type>anti_affinity</type>
  <enforcement>strict</enforcement>
  <vm_group_ref>c1</vm_group_ref>
  <vm_group_ref>c3</vm_group_ref>
  <vm_group_ref>s2</vm_group_ref>
  <vm_group_ref>s4</vm_group_ref>
  <vm_group_ref>s5</vm_group_ref>
  <vm_group_ref>s6</vm_group_ref>
</placement>
<placement>
  <target_vm_group_ref>s8</target_vm_group_ref>
  <type>anti_affinity</type>
  <enforcement>strict</enforcement>
  <vm_group_ref>c1</vm_group_ref>
  <vm_group_ref>c3</vm_group_ref>
  <vm_group_ref>s2</vm_group_ref>
  <vm_group_ref>s4</vm_group_ref>
  <vm_group_ref>s5</vm_group_ref>
  <vm_group_ref>s6</vm_group_ref>
  <vm_group_ref>s7</vm_group_ref>
</placement>
<placement>
  <target_vm_group_ref>s9</target_vm_group_ref>
  <type>anti_affinity</type>
  <enforcement>strict</enforcement>
  <vm_group_ref>c1</vm_group_ref>
  <vm_group_ref>c3</vm_group_ref>
  <vm_group_ref>s2</vm_group_ref>
  <vm_group_ref>s4</vm_group_ref>
  <vm_group_ref>s5</vm_group_ref>
  <vm_group_ref>s6</vm_group_ref>
  <vm_group_ref>s7</vm_group_ref>
  <vm_group_ref>s8</vm_group_ref>
</placement>
```

For c3 as the target_vm_group, delete the whole placement policy by adding attribute `nc:operation='delete'` to the XML element.

.\

```
<placement nc:operation='delete'>
  <target_vm_group_ref>c3</target_vm_group_ref>
  <type>anti_affinity</type>
  <enforcement>strict</enforcement>
  <vm_group_ref>c1</vm_group_ref>
  <vm_group_ref>s2</vm_group_ref>
</placement>
```

For c3 as the vm_group_ref, remove the vm_group_ref entry itself, and keep other relationships as is.

```
<placement>
  <target_vm_group_ref>s10</target_vm_group_ref>
  <type>anti_affinity</type>
  <enforcement>strict</enforcement>
  <vm_group_ref>c1</vm_group_ref>
  <vm_group_ref nc:operation='delete'>c3</vm_group_ref>
  <vm_group_ref>s2</vm_group_ref>
  <vm_group_ref>s4</vm_group_ref>
  <vm_group_ref>s5</vm_group_ref>
  <vm_group_ref>s6</vm_group_ref>
  <vm_group_ref>s7</vm_group_ref>
  <vm_group_ref>s8</vm_group_ref>
```

```
  <vm_group_ref>s9</vm_group_ref>
</placement>
```

For a placement policy that has only one vm_group_ref element, either the the vm_group_ref is c3 or the target_vm_group is c3, remove the whole policy. This is becuase, when c3 is removed this policy does not have any meaning:

```
<placement nc:operation='delete'>
  <target_vm_group_ref>c11</target_vm_group_ref>
  <type>anti_affinity</type>
  <enforcement>strict</enforcement>
  <vm_group_ref>c3</vm_group_ref>
</placement>
```

4. The last step is to mark the VM group itself for deletion by adding attribute *nc:operation='delete'* to the XML element

```
<vm_group nc:operation='delete'>
                <name>c3</name>
                <flavor>SFPCF101-DEPLOYMENT-control-function</flavor>
                <bootup_time>1800</bootup_time>
                <recovery_wait_time>1</recovery_wait_time>
...
```

5. To prepare the datamodel for adding the same VM group back, take the deletion datamodel, remove all the nc:operation='delete' everywhere.

Once the two datamodel files are ready, use the following command for service update:

```
/opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli edit-config {deleting datamodel file}
```

Wait until the service update is complete. Then add the VNF back:

Warning Service state must be ALIVE before you can perform adding the VNF back. If not, please recovery any VMs that is not in ALIVE state.

```
/opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli edit-config {adding datamodel file}
```

Get VNF Service (Deployment) Out of INERT State

For ESC Release 3.1 and later releases, the service may get stuck in the inert state when stop VM operation fails, and no recovery is triggered. One VM is an error, but the service is inert. You can use ENABLE MONITOR to get the VM and service back to alive or error.

```
/opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli vm-action ENABLE_MONITOR {VM Name}
```

This operation can enable the monitoring of the VM. If the VM is running in the VIM, the VM ALIVE event should be back to the VM state machine. The VM finally transits to an alive state. If the VM is not running in the VIM, the timer expires, the recovery procedure can bring the VM back. Meanwhile, the service transits to an active or error state.

VNF Recovery Rejected Because of Wrong Service Status

Problem

When doing VNF VM manual recovery, the request gets rejected due to worng service status (ESC 3.1 and later releases) as:

```
$ /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli recovery-vm-action DO vm-name

Recovery VM Action
/opt/cisco/esc/confd/bin/netconf-console --port=830 --host=127.0.0.1 --user=admin
--privKeyFile=/home/admin/.ssh/confd_id_dsa --privKeyType=dsa --rpc=/tmp/esc_nc_cli.L1WdqyIE7r
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <rpc-error>
    <error-type>application</error-type>
    <error-tag>operation-failed</error-tag>
    <error-severity>error</error-severity>
    <error-path xmlns:esc="http://www.cisco.com/esc/esc"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
    /nc:rpc/esc:recoveryVmAction
  </error-path>
    <error-message xml:lang="en">Exception from action callback: Recovery VM Operation:
recovery_do is not applicable since the service is in [SERVICE_INERT_STATE]
state.</error-message>
    <error-info>
      <bad-element>recoveryVmAction</bad-element>
    </error-info>
  </rpc-error>
</rpc-reply>
```

Solution

At this point, check the opdata to find out the service state and VM state.

```
$ /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli get esc_datamodel/opdata

  <state_machine>
    <state>SERVICE_INERT_STATE</state>
    <vm_state_machines>
        <vm_state_machine>
            <vm_name>depz_g1_0_b6d19896-bc3b-400a-ad50-6d84c522067d</vm_name>
            <state>VM_MONITOR_UNSET_STATE</state>
        </vm_state_machine>
        <vm_state_machine>
            <vm_name>depz_g1_1_f8445a8a-29ba-457d-9224-c46eeaa97f72</vm_name>
            <state>VM_ALIVE_STATE</state>
        </vm_state_machine>
     </vm_state_machines>
  </state_machine>
```

Enable monitor for the VM in the unset monitor state:

```
$ /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli vm-action ENABLE_MONITOR vm-name
```

After sometime, check the opdata again. The service should transit to active state or error state:

```
$ /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli get esc_datamodel/opdata

        <state_machine>
            <state>SERVICE_ACTIVE_STATE</state>
            <vm_state_machines>
                <vm_state_machine>
                    <vm_name>depz_g1_0_b6d19896-bc3b-400a-ad50-6d84c522067d</vm_name>
                    <state>VM_ALIVE_STATE</state>
                </vm_state_machine>
                <vm_state_machine>
                    <vm_name>depz_g1_1_f8445a8a-29ba-457d-9224-c46eeaa97f72</vm_name>
                    <state>VM_ALIVE_STATE</state>
                </vm_state_machine>
```

```
            </vm_state_machines>
        </state_machine>
```

Now do a manual recovery of the VM in error state:

```
$ /opt/cisco/esc/esc-confd/esc-cli/esc_nc_cli recovery-vm-action DO vm-name
```

VNF Operation Rejected Because of VIM Connector Issue

Problem

When VNF operation (for example, deploy, recovery) gets denied with the following reason:

```
Default VIM Connector is not set up, or is unreachable. Please check your VIM Connector
credentials and VIM status.
```

Description

If ESC was setup for mulit-vim, ensure that at least one VIM connector is marked as "default". Otherwise, check the ESC VIM connector status.

```
[admin@leishi-test ~]$ escadm vim show
[
    {
        "status": "CONNECTION_SUCCESSFUL",
        "status_message": "Successfully connected to VIM",
        "type": "OPENSTACK",
        "id": "default_openstack_vim",
        "properties": {
            "property": [
                {
                    "name": "os_project_domain_name",
                    "value": "default"
                },
                {
                    "name": "os_auth_url",
                    "value": "http://10.85.103.143:35357/v3"
                },
                {
                    "name": "os_project_name",
                    "value": "admin"
                }
            ]
        }
    }
]
{
    "user": [
        {
            "credentials": {
                "properties": {
                    "property": [
                        {
                            "name": "os_password",
                            "value": "cisco123"
                        },
                        {
                            "name": "os_user_domain_name",
                            "value": "default"
                        }
                    ]
                }
            },
```

```
            "vim_id": "default_openstack_vim",
            "id": "admin"
        }
    ]
}
```

Solution

If there is no VIM connector returned, add one. If one VIM connector is returned but the status is not "CONNECTION_SUCCESSFUL", check the ***/var/log/esc/vimmanager/vimmanager.log*** for the following entry:

```
2017-12-07 23:11:49.760 [http-nio-127.0.0.1-8095-exec-5] INFO
c.c.e.v.c.VimConnectionManagerService - Registering an user.
```

If there is an exception or error after the entry, it indicates the root cause. For example, if there is an error related to SSL, it means the certificate is missing or wrong.

```
2017-12-07 23:11:49.818 [http-nio-127.0.0.1-8095-exec-5] ERROR c.c.e.v.p.i.o.OpenStackProvider
 - Failed to register a user
org.openstack4j.api.exceptions.ConnectionException: javax.net.ssl.SSLHandshakeException:
sun.security.validator.ValidatorException: PKIX path building failed:
sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification
 path to requested target
    at
org.openstack4j.connectors.jersey2.HttpExecutorServiceImpl.invoke(HttpExecutorServiceImpl.java:58)
```

If there is a connection timeout or host name cannot reach exception, try to make a CURL get call to the AuthUrl provided, ensure that the OpenStack is reachable from the ESC VM.

```
curl -k https://www.xxxxx.com:5000/
```

If there is no error nor exception after the "Register an user" entry, it means that the authentication information provided is not correct. In this case, check the ***/var/log/esc/vimmanager/vim_vimmanager.log***. Look at the beginning of the log file where the inital authentication happens:

```
2017-12-07 23:11:49.748 INFO  os -   1 * Sending client request on thread
http-nio-127.0.0.1-8095-exec-4
1 > POST https://10.85.103.49:35357/v3/auth/tokens
1 > Accept: application/json
1 > Content-Type: application/json
1 > OS4J-Auth-Command: Tokens
{
  "auth" : {
    "identity" : {
      "password" : {
        "user" : {
          "name" : "admin",
          "domain" : {
            "name" : "default"
          },
          "password" : "****"
        }
      },
      "methods" : [ "password" ]
    },
    "scope" : {
      "project" : {
        "name" : "admin",
        "domain" : {
          "name" : "default"
        }
      }
    }
```

```
   }
}
```

Double check the authUrl, user, project/tenant. For V3 authentication ensure that the authUrl is the actual V3 endpoint, otherwise a *404* is returned. Also for V3 authentication, ensure that the user domain and project domain are provided. If you use an openrc file from Horizon to boot ESC VM, and the openrc does not contain the project domain or user domain, declare explicitly:

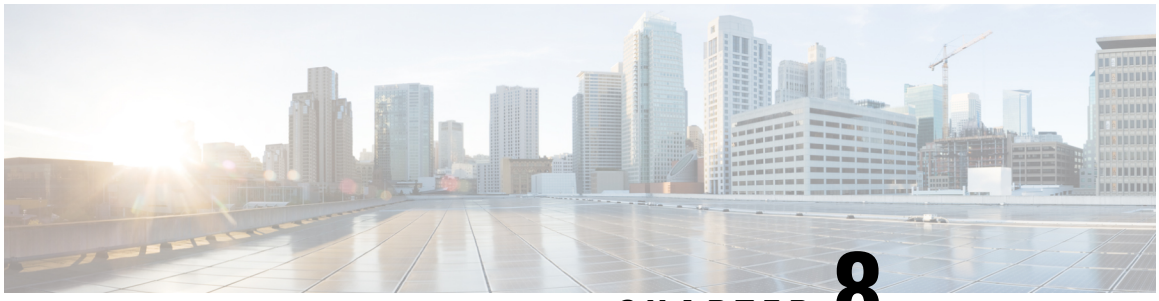
```
OS_PROJECT_DOMAIN_NAME=default
OS_USER_DOMAIN_NAME=default
```

To verify if ESC gets the correct password for the default vim connector with bootvm, do the following:

```
admin@leishi-test ~]$ sudo escadm reload
[sudo] password for admin:
[admin@leishi-test ~]$ cat /opt/cisco/esc/esc-config/esc_params.conf
openstack.os_auth_url= http://10.85.103.153:35357/v3
openstack.os_project_name= admin
openstack.os_tenant_name= admin
openstack.os_user_domain_name= default
openstack.os_project_domain_name= default
openstack.os_identity_api_version= 3
openstack.os_username = admin
openstack.os_password = cisco123
```

PART VII

Troubleshooting Cisco Elastic Services Controller Post Install



CHAPTER 8

Troubleshooting Cisco Elastic Services Controller Post Install

Troubleshooting Cisco Elastic Services Controller Post Install

Following are a few examples of the issues you may face:

Before you begin

Problem:

If there are communication issues between ESC and other components within the management and orchestration stack, there may be several reasons for this.

Procedure

Step 1 Client makes the request, and the request hangs after it makes a call to the server.

Step 2 Notifications are not getting transmitted successfully between the components. Following is an example which shows an extract from the mona.log, where a D-MONA agent is unable to send notifications to the central ESC:

```
2022-03-01 16:44:31.355 [QuartzScheduler_Worker-2] [ruleName] :
rule-VM_ALIVE-1-esc-etsi-sol3-vnf-info-001__319bb3d8-71c4-4957-8fe3-34240296d8e9__0:PostVmEventAction
 execution, Post to url:https://x.x.x.x:8443/ESCManager/dmona/api/events/notif,
payload:rule-VM_ALIVE-1-esc-etsi-sol3-vnf-info-001__319bb3d8-71c4-4957-8fe3-34240296d8e9__0
2022-03-01 16:44:41.488 [QuartzScheduler_Worker-2] Error during esc post execution:I/O error
 on POST request for "https://x.x.x.x:8443/ESCManager/dmona/api/events/notif": Read timed
out; nested exception is java.net.SocketTimeoutException: Read timed out
```

What to do next

Solution:

Issue a curl command from the client to the server to check routing issues. Send a small or empty payload and ensure that the server receives it. Once you know that there are no issues with the routing, try the actual call that the client is attempting and expect the call to get failed.

When a system tries to transmit data over an interface, the data gets fragmented into packets based on the Maximum Transmission Unit (MTU) size configured on the interface.

For example, D-MONA communicates to the VNF on the orchestration network, but ESC through the management network. If the management network does not receive the notifications successfully, check the MTU size.

Often, a larger MTU fails to work in the event. Hence, there are intermediate systems that transmit the packets towards their destination. The intermediate systems are configured with smaller MTUs (as the packets are transferred through their ingress and egress interfaces).

To check the MTU size on an interface:

```
$ ifconfig eth0
ether: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9000
        inet 172.29.0.33  netmask 255.255.240.0  broadcast 172.29.15.255
```

And to update it to a safe value of 1500:

```
$ sudo ifconfig eth0 mtu 1500
```

Now, retry the actual call that the client was failing with and check that it is received successfully by the server.