



## Post Deployment Operations

---

- [Deactivating the USP Deployment, on page 1](#)
- [Terminating the AutoDeploy VM, on page 2](#)
- [Terminating the AutoIT VM, on page 2](#)
- [Deploy and Undeploy the Card with the NCS CLI, on page 3](#)
- [Monitoring and Troubleshooting the Deployment, on page 6](#)
- [Monitoring AutoDeploy Operations, on page 33](#)
- [Monitoring AutoIT Operations, on page 38](#)
- [Monitoring AutoVNF Operations, on page 43](#)
- [UAS Log Collection, on page 59](#)
- [Secure File Transfer, on page 66](#)
- [Monitoring VNFM Operations, on page 71](#)
- [Monitoring VNF Operations, on page 74](#)
- [Monitoring and Recovering AutoVNF Through AutoIT, on page 77](#)
- [Monitoring and Recovering VNFC Through AutoVNF, on page 79](#)
- [Troubleshooting Deactivation Process and Issues, on page 81](#)
- [Troubleshooting UEM Issues, on page 85](#)

## Deactivating the USP Deployment



### Caution

It is recommended that you perform the checks identified in [Pre-Deactivation/Post-Activation Health Check Summary, on page 6](#) before performing any deactivations. It is also recommended that you back up relevant data before proceeding. Refer to [Backing Up Deployment Information](#) for more information.

Execute the following command to deactivate the entire USP deployment:

```
deactivate nsd <nsd_name>
```

The output of this command is a transaction-id which can be used to monitor the deactivation progress using the following command

```
show log <transaction_id> | display xml
```

Example output for a successful USP deactivation:

## Terminating the AutoDeploy VM

Terminating the AutoDeploy VM leverages the same *boot\_uas.py* script used to instantiate the AutoDeploy VM.



### Important

- Ensure that no changes have been made to this file since it was used to deploy AutoDeploy.
- Be sure to take a backup of the VM content if you are terminating the VM in order to upgrade with a new ISO.
- If AutoDeploy was deployed with HA support, this process terminates both VMs.

To terminate the AutoDeploy VM:

1. Log on to the Ultra M Manager Node.
2. Terminate the AutoDeploy VM.

```
./boot_uas.py --kvm --autodeploy --ha --delete-uas
```

Example command output:

```
2018-01-24 16:30:23,821 - Removing old deployment 'AutoDeploy_instance_0', if it exists
2018-01-24 16:30:24,176 - Removing old deployment 'AutoDeploy_instance_1', if it exists
```

3. View the status.

```
show uas
```

Example command output:

```
Id      Name                               State
-----
```

## Terminating the AutoIT VM

Terminating the AutoIT VM leverages the same *boot\_uas.py* script used to instantiate the AutoIT-VNF VM.



### Important

- Ensure that no changes have been made to this file since it was used to deploy AutoIT.
- Be sure to take a backup of the VM content if you are terminating the VM in order to upgrade with a new ISO.
- If AutoIT was deployed with HA support, this process terminates both VMs.

To terminate the AutoIT VM:

1. Log on to the Ultra M Manager Node.
2. Terminate the AutoIT VM.

```
./boot_uas.py --kvm --autoit --ha --delete-uas
```

Example command output:

```
2018-01-24 16:25:23,734 - Removing old deployment 'AutoIT_instance_0', if it exists
2018-01-24 16:25:24,056 - Removing old deployment 'AutoIT_instance_1', if it exists
```

3. View the status.

```
show uas
```

Example command output:

```
Id      Name                               State
-----
```

## Deploy and Undeploy the Card with the NCS CLI

To undeploy and redeploy the card (service or session function) using the NCS CLI:

1. Log on to the master UEM VM.
2. Access the NCS CLI.

```
sudo -i
ncs_cli -u admin -C
```

3. Undeploy or suspend the card.

```
suspend-vnfci vnfid <name> vdu <VDU> vnfci <VNFCI Instance>
```

For example:

```
suspend-vnfci vnfid abc vdu sf vnfci sf1 success true
```

4. Verify the operational status of VNF, card, VDUs. Suspending card removes the card, e.g. from CF.

In 6.2 and earlier releases:

```
show vnf-state
```

```
vnf-state running
```

In 6.3 and later releases:

```
show vnfproxy:vnfd <vnfd_name>vnf-state
```

```
vnf-state running
```

```
show card table
```

Slot	Card Type	Oper State	SPOF	Attach
1: CFC	Control Function Virtual Card	Active	No	
2: CFC	Control Function Virtual Card	Standby	-	
4: FC	1-Port Service Function Virtual Card	Standby	-	

In 6.2 and earlier releases:

```
show vdus
```

ID	CARD	TYPE	ID	DEVICE	MEMORY		CONSTITUENT		IS	INITIALIZED	VIM
					CPU	UTILS	STORAGE	ELEMENT			
ID				NAME	GROUP	BYTES	GROUP	GROUP	INFRA		
cf	control-function		cf1	scm-cf-nc	scm-cf-nc	ugp			true	true	
76e2f28a-4427-4b1d-9c44-72ff51e0d124						-	-	-			
			cf2	scm-cf-nc	scm-cf-nc	ugp			true	true	
b1155c6e-26f1-44c1-8832-0e9a02f7acd3						-	-	-			
sf	session-function		sf1	-	-	ugp			true	false	
7822cea9-1707-4790-abb3-33bb4d26b567						-	-	-			
			sf2	-	-	ugp			true	false	
7fd8f37f-59cf-4c9a-811f-faa0abd30b58						-	-	-			

In 6.3 and later releases:

```

show vnfmpoxy:vnfd <vnfd_name> vdus
vdus vdu cf1
card-type control-function
vnfci cf1
  device-name          vnfdeployment1
  device-group         cf-nc
  constituent-element-group ugp-standalone
  is-infra             true
  initialized          true
  vim-id              1ca3fca7-8929-4830-9e35-4cac294b62bf
vnfci cf2
  device-name          vnfdeployment2
  device-group         cf-nc
  constituent-element-group ugp-standalone
  is-infra             true
  initialized          true
  vim-id              6ce2a22c-1c03-4f5e-95a4-c791d09a1024
vdus vdu sf1
card-type session-function
vnfci sf1
  constituent-element-group ugp-standalone
  is-infra             true
  initialized          false
  vim-id              de5bec01-c3e4-4bbf-8f45-ac2354fc9fbc
vnfci sf2
  constituent-element-group ugp-standalone
  is-infra             true
  initialized          false
  vim-id              8fe9eebc-614a-464a-8e87-5288be0528c9

```

UEM changes the status of suspended card to *undeployed*. For example, UEM Zookeeper:

In 6.2 and earlier releases:

```

[zk: localhost:2181 (CONNECTED) 0] get /config/vnf
{"state": "run", "name": "abcabc-autovnf-vpc-abcabc"}

[zk: localhost:2181 (CONNECTED) 1] get /config/vnfd
{"name": "abcabc-autovnf-vpc-abcabc", "version": "6.0", "deployment-flavor-id": ["generic"],
"anti-affinity-cards": ["control-function", "session-function"], "card-type-to-vd": {"control-function": ["cf"], "session-function": ["sf"]}}

[zk: localhost:2181 (CONNECTED) 2] get /config/vdus/sf/sf1
{"cpts": [{"vnfc": "sf-vnfc-ugp", "cpid": null, "vl": null}], "affinity": null,
"initvars": [{"best_path": "staros_param.cfg", "path vars": [{"name": "CARD_TYPE_NM", "val": "0x42020100"}, {"name": "SLOT_CARD_NUMBER", "val": "3"}, {"name": "MEM_PROXY_ADDRS", "val": "101.101.14.9,101.101.14.16,101.101.14.13"}]}, {"operation": "create", "ceg-id": "ugp", "vnfci-id": "sf1",
"context-vars": null, "nat-pool": null, "vim-id": "abcabc-autovnf-vpc-abcabc-sf-1", "volume": null}

```

```
[zk: localhost:2181(CONNECTED) 3] get /oper/vdus/sf/sf1
{"id":"sf1","state":"undeployed","vnfcId":"sf-vnfc-ugp",
"uuid":"sf1","host":"tblano-osd-compute-2.localdomain","vimId":"7822cea9-1707-4790-abb3-33bb4d26b567",
"opts":[{"cpid":"eth0","state":"undeployed","subnet":"94c4ea79-eb81-4a7d-b726-4f780a05436f","netmask":"255.255.255.0",
"dhcp":true,"vl":"vl-di-internall","vnfc":"sf-vnfc-ugp","port_id":"0b5db23-5c43-463c-ae14-f1fb94f90dc","ip address":"192.168.1.124",
"mac address":"fa:16:3e:b6:26:da","network":"55d41c29-f8ed-4006-b29c-5ad3bf73bf42"},{"cpid":"eth1","state":"undeployed","nicid":1,
"subnet":"472e0423-a9c8-4eb6-9782-a741afe3b93a","netmask":"255.255.255.0","dhcp":true,"vl":"vl-autoit-abcabc_ord","vnfc":"sf-vnfc-ugp",
"port_id":"55667788-99aa-bb-cc-dd-ee-f1f2f3f4f5f6f7f8f9fa","ip address":"192.168.1.125",
"mac address":"fa:16:3e:b6:26:da","network":"55d41c29-f8ed-4006-b29c-5ad3bf73bf42"}]}

[zk: localhost:2181(CONNECTED) 4] get /oper/vnf
{"state":"running","name":"abcabc-autovnf-vpc-abcabc"}

[zk: localhost:2181(CONNECTED) 5] get /oper/vdrs/sf1
{"state":"undeployed","vnfcId":"sf-vnfc-ugp-standalone","vmId":"7822cea9-1707-4790-abb3-33bb4d26b567",
"opts":[{"cpid":"eth0","state":"undeployed","subnet":"94c4ea79-eb81-4a7d-b726-4f780a05436f","netmask":"255.255.255.0",
"dhcp":true,"vl":"vl-di-internall","vnfc":"sf-vnfc-ugp-standalone","port_id":"0b5db23-5c43-463c-ae14-f1fb94f90dc",
"ip address":"192.168.1.124","mac address":"fa:16:3e:b6:26:da","network":"55d41c29-f8ed-4006-b29c-5ad3bf73bf42"},
{"cpid":"eth1","state":"undeployed","nicid":1,"subnet":"472e0423-a9c8-4eb6-9782-a741afe3b93a",
"netmask":"255.255.255.0","dhcp":true,"vl":"vl-autoit-abcabc_ord","vnfc":"sf-vnfc-ugp-standalone",
"port_id":"55667788-99aa-bb-cc-dd-ee-f1f2f3f4f5f6f7f8f9fa","ip address":"192.168.1.125",
"mac address":"fa:16:3e:b6:26:da","network":"55d41c29-f8ed-4006-b29c-5ad3bf73bf42"}]}

```

### In 6.3 and later releases:

```
[zk: 30.31.14.18(CONNECTED) 0] get /config/vnfs/vnfd-deployment/vnf
{"state":"run","name":"vnfd-deployment"}

[zk: 30.31.14.18(CONNECTED) 1] get /config/vnfs/vnfd-deployment/vnfd
{"name":"vnfd-deployment","asio":"63","ipnet-fac-id":{"cpid":"","affinity-ack":{"total-functio":"","session-functio":"","ack-to-vid":{"total-functio":"","session-functio":"","sf1"}}}}

[zk: 30.31.14.18(CONNECTED) 2] get /config/vnfs/vnfd-deployment/vnfvdus/sf1/sf2
{"opts":[{"vnfc":"sf-vnfc-ugp-standalone","cid":"null","vl":"null","affinity":"null","vnfcId":"sf-vnfc-ugp-standalone","vmId":"0640000","name":"SOLO-CARD-NUMBER","val":"4"},
{"name":"VMPOXY-ADDR","val":"30.31.14.12,30.31.14.18"}]},{"action":"create","virtualPortId":[],"cpid":"","ipnet-standalone","nic-id":"sf2","totet-vas":"null","tot-ool":"null","vnfd":
"vnfd-deployment-sf1-1","volume":null,"virtual-port-id":[]}]

[zk: 30.31.14.18(CONNECTED) 3] get /oper/vnfs/vnfd-deployment/vdus/sf1/sf2
{"id":"sf2","state":"undeployed","vnfcId":"vnfd-deployment","vnfcId":"sf-vnfc-ugp-standalone",
"uuid":"sf2","host":"tb2-compute-2.localdomain","vimId":"8fe9eebc-614a-464a-8e87-5288be0528c9","opts":[{"cpid":"eth0",
"state":"undeployed","subnet":"f3b6a640-bcec-4be4-ab2c-e058042e0f83","netmask":"255.255.255.0","dhcp":true,
"vl":"vl-di-internall","vnfc":"sf-vnfc-ugp-standalone","port_id":"c69a5d11-b803-4189-bc9e-7bd6ea56a820","ip address":"192.168.10.85",
"mac address":"fa:16:3e:4f:a4:67","network":"e45b4f35-7438-4250-b8c6-ccc44197776e"},{"cpid":"eth1","state":"undeployed","nicid":1,
"subnet":"786a04c-d87-482d-ad40-c7e5920f00c6","netmask":"255.255.255.0","dhcp":true,"vl":"vl-autoit-vim_ord","vnfc":"sf-vnfc-ugp-standalone",
"port_id":"4325c98d-57a6-4a1a-ba4e-a3b77a092d0d",
"ip address":"30.31.14.15","mac address":"fa:16:3e:8a:ce:0e","network":"0317d09-5e70-4851-9d15-389f31773d1"},{"cpid":"eth2","state":"undeployed","nicid":2,
"subnet":"5a604f74-73e0-6a46609","netmask":"255.255.255.0","dhcp":true,"vl":"vl-secure","vnfc":"sf-vnfc-ugp-standalone","port_id":"a3ff646c-47c6-a23a1e20","ip address":"192.168.12.67",
"mac address":"fa:16:3e:7e:7e:7e","network":"9667e54-42c5e-3d83635d"},{"cpid":"eth3","state":"undeployed","nicid":3,"subnet":"5a604f74-73e0-6a46609","netmask":"255.255.255.0","dhcp":true,"vl":"vl-secure",
"vnfc":"sf-vnfc-ugp-standalone","port_id":"5a66933-43b2b-46ba220","ip address":"192.168.12.3","mac address":"fa:16:3e:7e:7e:7e","network":"9667e54-42c5e-3d83635d"},{"total":true,"vid":"sf1"}]}

[zk: 30.31.14.18(CONNECTED) 4] get /oper/vnfs/vnfd-deployment/vdus/vnf
{"state":"running","name":"vnfd-deployment"}

[zk: 30.31.14.18(CONNECTED) 6] get /oper/vnfs/vnfd-deployment/vdrs/sf2
{"state":"undeployed","vnfcId":"sf-vnfc-ugp-standalone","vmId":"7822cea9-1707-4790-abb3-33bb4d26b567",
"opts":[{"cpid":"eth0","state":"undeployed","subnet":"94c4ea79-eb81-4a7d-b726-4f780a05436f","netmask":"255.255.255.0",
"dhcp":true,"vl":"vl-di-internall","vnfc":"sf-vnfc-ugp-standalone","port_id":"0b5db23-5c43-463c-ae14-f1fb94f90dc",
"ip address":"192.168.1.124","mac address":"fa:16:3e:b6:26:da","network":"55d41c29-f8ed-4006-b29c-5ad3bf73bf42"},
{"cpid":"eth1","state":"undeployed","nicid":1,"subnet":"472e0423-a9c8-4eb6-9782-a741afe3b93a",
"netmask":"255.255.255.0","dhcp":true,"vl":"vl-autoit-abcabc_ord","vnfc":"sf-vnfc-ugp-standalone",
"port_id":"55667788-99aa-bb-cc-dd-ee-f1f2f3f4f5f6f7f8f9fa","ip address":"192.168.1.125",
"mac address":"fa:16:3e:b6:26:da","network":"55d41c29-f8ed-4006-b29c-5ad3bf73bf42"}]}

```

## 5. Redeploy or resume the card by executing the following command:

```
resume-vnfc vnfid <name> vdu <VDU> vnfc <VNFC Instance>
```

# Monitoring and Troubleshooting the Deployment

## Pre-Deactivation/Post-Activation Health Check Summary

Table 1: Pre-deactivation/Post-activation Health Checks, on page 6 contains a summary of items to check/verify before performing a deactivation and/or after an activation.

**Table 1: Pre-deactivation/Post-activation Health Checks**

Item to Check	Notes
Checking OSP-D Server Health	Perform all identified checks.
Checking Controller Server Health	Perform all identified checks.
Checking OSD Compute Server Health	Perform all identified checks.
Viewing AutoVNF Operational Data	In particular, check the outputs of the following commands: <ul style="list-style-type: none"> <li>• <b>show uas</b></li> <li>• In releases prior to 6.0: <b>show autovnf-oper:vip-port</b> In 6.0 and later releases: <b>show vnfr</b></li> <li>• In releases prior to 6.0: <b>show autovnf-oper:vnf-em</b> In 6.0 and later releases: <b>show vnfr</b></li> <li>• In releases prior to 6.0: <b>show autovnf-oper:vnfm</b> In 6.0 and later releases: <b>show vnfr</b></li> </ul>
Viewing ESC Status	Perform all identified checks.
Viewing ESC Health	Perform all identified checks.
Viewing UEM Service Status	Perform all identified checks.
Viewing VNF Information through the Control Function	Perform all identified checks.

## Checking OSP-D Server Health

### Viewing Stack Status

Log on to the server on which OSP-D is running to view the stack status by executing the following command:

```
openstack stack list
```

Example output:

ID	Stack Name	Stack Status	Creation Time
db229d67-212d-4086-a266-e635b2902708	tb3-ultram	CREATE_COMPLETE	2017-06-20T02:31:31Z



**Note** Prior to an update, the stack status may be “CREATE\_COMPLETE” at the beginning of the update procedure. The stack status should read “UPDATE\_COMPLETE” and list and update time at the successful completion of the update procedure.

## Viewing the Bare Metal Node List

Log on to the server on which OSP-D is running to view the node list by executing the following command:

```
openstack baremetal node list
```

**Example command output:**

UUID	Name	Instance UUID	Power State	Provisioning State	Maintenance
6725bb18-2895-4a8a-86ad-96b00cc9df4d	None	bc903f51-8483-4522-bcd7-ac396ac626b1	power on	active	False
flaa6356-40a0-41de-belb-fa6033c9affb	None	05fbfb44-ccd9-475d-b263-58b2deaf8554	power on	active	False
f02357a3-6f9b-46ae-b31f-1a21f6d33543	None	dd0596b1-bd35-451a-85bc-c635e7fa6d14	power on	active	False
ca1153d6-ffaf-481a-ac9b-bc2afc450152	None	96d2725c-9c70-4a66-9d3c-4a0356faf1c0	power on	active	False
8f338102-c114-4a7a-94f0-9e1a54494519	None	85a9a708-5eae-4ea2-8b29-dc2acd6e515d	power on	active	False
5d3d3525-2528-4801-b885-6c4b340393a6	None	315c7aea-acef-4341-aa9e-bcd594cae592	power on	active	False
ac21208b-36fd-4404-8e68-53a90df3a29f	None	9f0b2ff3-5234-42e9-81dd-c0ef5e454137	power on	active	False
a6d92bfc-0136-4c22-9988-0108df775a03	None	2a3e2086-3516-40ac-a584-3714e91858f5	power on	active	False
5f0593b7-31de-4291-b43f-a549699cd470	None	f4cc50d4-441e-4728-9984-53df29f0b7f7	power on	active	False
99225e1b-085e-4ef7-8173-4687900b741a	None	200a918e-abb3-4539-a1c4-7e30f2d8ebc2	power on	active	False
c6ec143b-a522-4d69-ab31-5b4934ad3c42	None	7c675ed5-17d9-47ad-a2ef-592353e27713	power on	active	False
e1026c43-f2a3-44ad-a385-4d4462552977	None	45b45041-656f-4ee1-8be2-976c71a35b1f	power on	active	False
122188ea-09ae-486c-b225-17cf0defe025	None	bd38818e-36ca-4fd9-a65f-c4b0e5b34977	power on	active	False
f6ecf896-6e5e-4735-8727-942478dee58a	None	82a79351-5520-4e89-ae19-48c7b6f6b39f	power on	active	False
e6db159e-008e-4186-8967-92a9faeee368	None	986affe6-23ba-48ba-ae4e-0d2226aabf55	power on	active	False
44f3a434-eaf8-4b1a-97e5-6340d277fa4e	None	1f385454-3ddb-40bd-bc6e-a55ad69fff47	power on	active	False
7ab70571-64ea-439b-a0f4-34147d01dfbf	None	6f9f76ac-3cf7-4002-94ba-39bc6f0b4c40	power on	active	False
6d478a22-874c-4611-834d-21f4809f90ce	None	8e37407f-c784-4f5f-942f-2e2c36aa3fa4	power on	active	False

## Viewing the OpenStack Server List

```

| 0a57a5ad-d160-477e-807f-11997307bc9c | None | 25b53356-9f02-4810-b722-efb6fd887879 | power
on | active | False |
| 6fff3d83-ed37-4934-89e0-d632aeb37b15 | None | 0ea048c0-6f4b-460d-99b2-796dd694c226 | power
on | active | False |
| 5496919c-c269-4860-b49a-e0d103a6a460 | None | 6a8e05aa-26fe-43bb-b464-ede86b9f4639 | power
on | active | False |
| 513b936d-1c52-4b0a-9ac4-4101fe812f07 | None | b92c5720-7db9-417b-b3d5-023046788c8e | power
on | active | False |

```

## Viewing the OpenStack Server List

Log on to the server on which OSP-D is running to ensure that stack components are active and running the same image by executing the following command:

```
openstack server list
```

**Example command output:**

```

+-----+-----+-----+-----+-----+
| ID | Image Name | Name | Status | Networks |
+-----+-----+-----+-----+-----+
| 9f0b2ff3-5234-42e9-81dd-c0ef5e454137 | tb3-ultram-compute-3 | ACTIVE |
ctlplane=192.200.0.133 | overcloud-full_20170620T011048 |
| 25b53356-9f02-4810-b722-efb6fd887879 | tb3-ultram-compute-15 | ACTIVE |
ctlplane=192.200.0.131 | overcloud-full_20170620T011048 |
| 986affe6-23ba-48ba-ae4e-0d2226aabf55 | tb3-ultram-compute-11 | ACTIVE |
ctlplane=192.200.0.128 | overcloud-full_20170620T011048 |
| 45b45041-656f-4ee1-8be2-976c71a35b1f | tb3-ultram-compute-8 | ACTIVE |
ctlplane=192.200.0.130 | overcloud-full_20170620T011048 |
| bd38818e-36ca-4fd9-a65f-c4b0e5b34977 | tb3-ultram-compute-9 | ACTIVE |
ctlplane=192.200.0.127 | overcloud-full_20170620T011048 |
| 82a79351-5520-4e89-ae19-48c7b6f6b39f | tb3-ultram-compute-10 | ACTIVE |
ctlplane=192.200.0.126 | overcloud-full_20170620T011048 |
| 1f385454-3ddb-40bd-bc6e-a55ad69fff47 | tb3-ultram-compute-12 | ACTIVE |
ctlplane=192.200.0.118 | overcloud-full_20170620T011048 |
| 8e37407f-c784-4f5f-942f-2e2c36aa3fa4 | tb3-ultram-compute-14 | ACTIVE |
ctlplane=192.200.0.117 | overcloud-full_20170620T011048 |
| 315c7aea-acef-4341-aa9e-bcd594cae592 | tb3-ultram-compute-2 | ACTIVE |
ctlplane=192.200.0.114 | overcloud-full_20170620T011048 |
| 2a3e2086-3516-40ac-a584-3714e91858f5 | tb3-ultram-compute-4 | ACTIVE |
ctlplane=192.200.0.120 | overcloud-full_20170620T011048 |
| b92c5720-7db9-417b-b3d5-023046788c8e | tb3-ultram-osd-compute-2 | ACTIVE |
ctlplane=192.200.0.110 | overcloud-full_20170620T011048 |
| 7c675ed5-17d9-47ad-a2ef-592353e27713 | tb3-ultram-compute-7 | ACTIVE |
ctlplane=192.200.0.111 | overcloud-full_20170620T011048 |
| 0ea048c0-6f4b-460d-99b2-796dd694c226 | tb3-ultram-osd-compute-0 | ACTIVE |
ctlplane=192.200.0.112 | overcloud-full_20170620T011048 |
| f4cc50d4-441e-4728-9984-53df29f0b7f7 | tb3-ultram-compute-5 | ACTIVE |
ctlplane=192.200.0.108 | overcloud-full_20170620T011048 |
| dd0596b1-bd35-451a-85bc-c635e7fa6d14 | tb3-ultram-controller-2 | ACTIVE |
ctlplane=192.200.0.115 | overcloud-full_20170620T011048 |
| 85a9a708-5eae-4ea2-8b29-dc2acd6e515d | tb3-ultram-compute-1 | ACTIVE |
ctlplane=192.200.0.102 | overcloud-full_20170620T011048 |
| bc903f51-8483-4522-bcd7-ac396ac626b1 | tb3-ultram-controller-0 | ACTIVE |
ctlplane=192.200.0.105 | overcloud-full_20170620T011048 |
| 6a8e05aa-26fe-43bb-b464-ede86b9f4639 | tb3-ultram-osd-compute-1 | ACTIVE |
ctlplane=192.200.0.106 | overcloud-full_20170620T011048 |
| 200a918e-abb3-4539-a1c4-7e30f2d8ebc2 | tb3-ultram-compute-6 | ACTIVE |
ctlplane=192.200.0.109 | overcloud-full_20170620T011048 |
| 05fbfb44-ccd9-475d-b263-58b2deaf8554 | tb3-ultram-controller-1 | ACTIVE |
ctlplane=192.200.0.113 | overcloud-full_20170620T011048 |

```



```
| 96d2725c-9c70-4a66-9d3c-4a0356faf1c0 | tb3-ultram-compute-0 | ACTIVE |
ctlplane=192.200.0.107 | overcloud-full_20170620T011048 |
| 6f9f76ac-3cf7-4002-94ba-39bc6f0b4c40 | tb3-ultram-compute-13 | ACTIVE |
ctlplane=192.200.0.103 | overcloud-full_20170620T011048 |
```

## Viewing the OpenStack Stack Resource List

Log on to the server on which OSP-D is running to view the stack resources and their status by executing the following command:

```
openstack stack resource list name
```

**Example command output:**

```
|-----|
| resource_name | physical_resource_id |
resource_type | resource_status | updated_time |
|-----|
| UpdateWorkflow | 94270702-cd8b-4441-a09e-5c9da0c2d02b |
OS::TripleO::Tasks::UpdateWorkflow | CREATE_COMPLETE | 2017-06-27T22:04:00Z |
| CephStorageHostsDeployment | 196dbba7-5d66-4a9c-9308-f47ff4d8be2d |
OS::Heat::StructuredDeployments | CREATE_COMPLETE | 2017-06-27T22:04:00Z |
| OsdComputeAllNodesDeployment | 6a5775c0-03d8-453f-92d8-be6ea5aed853 |
OS::Heat::StructuredDeployments | CREATE_COMPLETE | 2017-06-27T22:04:00Z |
| BlockStorageHostsDeployment | 97b2f70a-c295-4437-9222-8248ec30badf |
OS::Heat::StructuredDeployments | CREATE_COMPLETE | 2017-06-27T22:04:00Z |
| CephStorage | 1bc20bb0-516a-4eb5-85e2-be9d30e2f6e8 |
OS::Heat::ResourceGroup | CREATE_COMPLETE | 2017-06-27T22:04:00Z |
| AllNodesDeploySteps | da9ead69-b83e-4cc9-86e8-8d823c02843b |
OS::TripleO::PostDeploySteps | CREATE_COMPLETE | 2017-06-27T22:04:00Z |
| CephStorageAllNodesDeployment | e5ee9df8-fae1-4641-9cfb-038c8f4eca85 |
OS::Heat::StructuredDeployments | CREATE_COMPLETE | 2017-06-27T22:04:00Z |
```

## Verifying Node Reachability

Log on to the server on which OSP-D is running to ensure the node reachability and availability by executing the following command:

```
for i in $(nova list| grep ACTIVE| awk '{print $12}' | sed 's\ctlplane=\g'
) ; do ssh heat-admin@${i} uptime ; done
```

This command establishes an SSH session with each node and report the system uptime. Investigate any node that does not reply or has an unexpected uptime.

**Example command output:**

```
14:47:10 up 18:15, 0 users, load average: 0.01, 0.02, 0.05
14:47:11 up 18:14, 0 users, load average: 9.50, 9.15, 12.32
14:47:11 up 18:14, 0 users, load average: 9.41, 9.09, 12.26
14:47:11 up 18:14, 0 users, load average: 10.41, 10.28, 10.49
14:47:12 up 18:15, 0 users, load average: 0.00, 0.02, 0.05
14:47:12 up 18:14, 0 users, load average: 0.18, 0.06, 0.06
14:47:12 up 18:15, 0 users, load average: 0.00, 0.03, 0.05
14:47:12 up 18:15, 0 users, load average: 0.00, 0.01, 0.05
14:47:13 up 18:14, 0 users, load average: 0.02, 0.02, 0.05
14:47:13 up 18:14, 0 users, load average: 8.23, 8.66, 12.29
14:47:13 up 18:14, 0 users, load average: 8.76, 8.87, 12.14
14:47:14 up 18:15, 0 users, load average: 0.01, 0.04, 0.05
14:47:14 up 18:15, 0 users, load average: 9.30, 9.08, 10.12
14:47:14 up 18:15, 0 users, load average: 0.01, 0.06, 0.05
14:47:14 up 18:14, 0 users, load average: 8.31, 8.61, 11.96
```

```

14:47:15 up 18:14,  0 users,  load average: 17.08, 12.09, 11.06
14:47:15 up 17:09,  0 users,  load average:  1.64,  1.33,  1.10
14:47:15 up 17:04,  0 users,  load average:  1.02,  0.77,  0.79
14:47:16 up 16:58,  0 users,  load average:  0.55,  0.63,  0.72
14:47:16 up 23:46,  0 users,  load average:  2.68,  3.46,  3.89
14:47:16 up 1 day, 5 min,  0 users,  load average:  4.10,  4.27,  4.44
14:47:17 up 23:53,  0 users,  load average:  1.90,  2.32,  2.24

```

## Verify NTP is running

To verify the operational status of NTP server:

1. Log on to the server on which OSP-D is running to ensure that NTP is running on all nodes in the cluster by executing the following command:

```

for i in $(nova list | grep ACTIVE | awk '{print $12}' | sed
's/ctlplane=\\g' ) ; do ssh heat-admin@${i} systemctl status ntpd |grep
Active; done

```

This command establishes an SSH session with each node and lists the ntpd status.

### Example command output:

```

Active: active (running) since Tue 2017-07-11 20:32:25 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:28 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:50 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:28 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:14 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:30 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:22 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:16 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:35 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:31 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:30 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:25 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:19 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:14 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:41 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 20:32:30 UTC; 18h ago
Active: active (running) since Tue 2017-07-11 21:37:32 UTC; 17h ago
Active: active (running) since Tue 2017-07-11 21:43:16 UTC; 17h ago
Active: active (running) since Tue 2017-07-11 21:48:57 UTC; 17h ago
Active: active (running) since Tue 2017-07-11 15:01:30 UTC; 23h ago
Active: active (running) since Tue 2017-07-11 14:42:10 UTC; 24h ago
Active: active (running) since Tue 2017-07-11 14:54:06 UTC; 23h ago

```

2. Verify that all the OpenStack nodes are synced to NTP server.

```

for i in $(nova list | grep -i overc- | awk '{print $12}' | sed
's/ctlplane=//g') ; do (ssh -o StrictHostKeyChecking=no heat-admin@${i}
sudo ntpstat | grep NTP) ; done

```

```

[stack@j19bxb-ospd ~]$ for i in $(nova list | grep -i overc- | awk '{print $12}' | sed
's/ctlplane=//g') ; do (ssh -o StrictHostKeyChecking=no heat-admin@${i} sudo ntpstat |
grep NTP) ; done
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3

```

```
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
synchronised to NTP server (10.84.96.130) at stratum 3
```

3. Check the NTP status on the server on which OSP-D is running by executing the following command:

```
systemctl status ntpd |grep Active
```

Investigate any node that is not actively running NTP.

## Checking OSP-D Server Health

### Verifying VM and Other Service Status and Quotas

Log on to the server on which OSP-D is running to verify that Overcloud VMs are active and running by executing the following commands:

```
cd /home/stack
source ~/<stack_name>rc-core
nova list
```



**Note** Overcloud VM status can also be checked through the Horizon GUI.

#### Example command output:

```
+-----+-----+-----+-----+-----+
| ID                                     | Name                                     |
+-----+-----+-----+-----+-----+
| 407891a2-85bb-4b84-a023-bca4ff304fc5 | auto-deploy-vm-uas-0                   |
| ACTIVE | - | Running | mgmt=172.16.181.21, 10.84.123.13 |
+-----+-----+-----+-----+-----+
| bb4c06c5-b328-47bd-ac57-a72a9b4bb496 | auto-it-vm-uas-0                       |
| ACTIVE | - | Running | mgmt=172.16.181.19, 10.84.123.12 |
+-----+-----+-----+-----+-----+
| fc0e47d3-e59e-41a3-9d8d-99371de1c4c5 | tb3-bxb-autovnf1-uas-0                 |
| ACTIVE | - | Running |                                     |
tb3-bxb-autovnf1-uas-orchestration=172.17.180.10; tb3-bxb-autovnf1-uas-management=172.17.181.8 |
+-----+-----+-----+-----+-----+
```

```

| 8056eff1-913e-479a-ac44-22eba42ceee1 | tb3-bxb-autovnf1-uas-1
| ACTIVE | - | Running |
tb3-bxb-autovnf1-uas-orchestration=172.17.180.6; tb3-bxb-autovnf1-uas-management=172.17.181.12

|
| 4e9fab14-dad0-4789-bc52-1fac3e40b7cc | tb3-bxb-autovnf1-uas-2
| ACTIVE | - | Running |
tb3-bxb-autovnf1-uas-orchestration=172.17.180.13; tb3-bxb-autovnf1-uas-management=172.17.181.3

|
| 1a4e65e3-9f9d-429f-a604-6dfb45ef2a45 | tb3-bxb-vnfm1-ESC-0
| ACTIVE | - | Running |
tb3-bxb-autovnf1-uas-orchestration=172.17.180.3; tb3-bxb-autovnf1-uas-management=172.17.181.4

|
| 7f4ec2dc-e8a8-4f6c-bfce-8f29735e9fca | tb3-bxb-vnfm1-ESC-1
| ACTIVE | - | Running |
tb3-bxb-autovnf1-uas-orchestration=172.17.180.14; tb3-bxb-autovnf1-uas-management=172.17.181.5

|
| 1c9fc0bd-dc16-426f-b387-c2b75b3a1c16 |
tb3-bxb-vnfm1-em_tb3-bx_0_190729a1-c703-4e15-b0b3-795e2e876f55 | ACTIVE | - |
Running | tb3-bxb-autovnf1-uas-orchestration=172.17.180.4;
tb3-bxb-autovnf1-uas-management=172.17.181.9

|
| 9a407a06-929a-49ce-8bae-4df35b5f8b40 |
tb3-bxb-vnfm1-em_tb3-bx_0_92c5224b-1f1f-4f3f-8ac8-137be69ce473 | ACTIVE | - |
Running | tb3-bxb-autovnf1-uas-orchestration=172.17.180.5;
tb3-bxb-autovnf1-uas-management=172.17.181.10

|
| e4528022-6e7b-43f9-94f6-a6ab6289478d |
tb3-bxb-vnfm1-em_tb3-bx_0_d9f7ecb2-a7dc-439b-b492-5ce0402264ea | ACTIVE | - |
Running | tb3-bxb-autovnf1-uas-orchestration=172.17.180.2;
tb3-bxb-autovnf1-uas-management=172.17.181.7

|
| 2calle5b-8eec-456d-9001-1f2600605ad4 |
vnfd1-deployment_c1_0_5b287829-6a9d-4c0a-97d0-a5e0f645b767 | ACTIVE | - |
Running | tb3-bxb-autovnf1-uas-orchestration=172.17.180.16;
tb3-bxb-vnfm1-di-internal1=192.168.1.4; tb3-bxb-autovnf1-uas-management=172.17.181.15;
tb3-bxb-vnfm1-di-internal2=192.168.2.5

|
| 0bdbd9e3-926a-4abe-81b3-95dc42ea0676 |
vnfd1-deployment_c2_0_7074a450-5268-4c94-965b-8fb809410d14 | ACTIVE | - |
Running | tb3-bxb-autovnf1-uas-orchestration=172.17.180.15;
tb3-bxb-vnfm1-di-internal1=192.168.1.2; tb3-bxb-autovnf1-uas-management=172.17.181.18;
tb3-bxb-vnfm1-di-internal2=192.168.2.6

|
| 8b07a9b1-139f-4a12-b16e-d35cb17f6668 |
vnfd1-deployment_s10_0_f6d110f9-9e49-43fe-be14-4ab87ca3334c | ACTIVE | - |
Running | tb3-bxb-autovnf1-uas-orchestration=172.17.180.7;
tb3-bxb-vnfm1-di-internal1=192.168.1.8; tb3-bxb-vnfm1-service-network1=10.10.10.3,
10.10.10.10; tb3-bxb-vnfm1-service-network2=20.20.20.5, 20.20.20.4;
tb3-bxb-vnfm1-di-internal2=192.168.2.12

|
| 4ff0ce2e-1d97-4056-a7aa-018412c0385d |
vnfd1-deployment_s3_0_5380ef6c-6fe3-4e92-aa44-d94ef6e94235 | ACTIVE | - |
Running | tb3-bxb-autovnf1-uas-orchestration=172.17.180.19;
tb3-bxb-vnfm1-di-internal1=192.168.1.5; tb3-bxb-vnfm1-service-network1=10.10.10.7, 10.10.10.2;
tb3-bxb-vnfm1-service-network2=20.20.20.9, 20.20.20.6; tb3-bxb-vnfm1-di-internal2=192.168.2.8

|
| 3954cd6e-0f12-4d4b-8558-2e035c126d9a |
vnfd1-deployment_s4_0_e5ae4aa9-a90e-4bfe-aaff-82ffd8f7fe34 | ACTIVE | - |

```

```

Running      | tb3-bxb-autovnf1-uas-orchestration=172.17.180.8;
tb3-bxb-vnfm1-di-internal1=192.168.1.9; tb3-bxb-vnfm1-service-network1=10.10.10.13,
10.10.10.8; tb3-bxb-vnfm1-service-network2=20.20.20.12, 20.20.20.10;
tb3-bxb-vnfm1-di-internal2=192.168.2.3      |
| 2cc6728c-2982-42bf-bb8b-198a14fdbcb31 |
vnfd1-deployment_s5_0_1d57c15d-alde-40d4-aac2-1715f01ac50a      | ACTIVE | -
Running      | tb3-bxb-autovnf1-uas-orchestration=172.17.180.17;
tb3-bxb-vnfm1-di-internal1=192.168.1.7; tb3-bxb-vnfm1-service-network1=10.10.10.5,
10.10.10.18; tb3-bxb-vnfm1-service-network2=20.20.20.11, 20.20.20.2;
tb3-bxb-vnfm1-di-internal2=192.168.2.4      |
| 876cc650-ae8b-497b-805a-24a305be6c13 |
vnfd1-deployment_s6_0_05e13a62-623c-4749-ae2a-15c70dd12e16      | ACTIVE | -
Running      | tb3-bxb-autovnf1-uas-orchestration=172.17.180.11;
tb3-bxb-vnfm1-di-internal1=192.168.1.6; tb3-bxb-vnfm1-service-network1=10.10.10.12,
10.10.10.9; tb3-bxb-vnfm1-service-network2=20.20.20.13, 20.20.20.18;
tb3-bxb-vnfm1-di-internal2=192.168.2.16     |
| 89f7245e-c2f7-4041-b5e6-1eee48641cfd |
vnfd1-deployment_s7_0_3a4d7273-e808-4b5f-8877-7aa182483d93      | ACTIVE | -
Running      | tb3-bxb-autovnf1-uas-orchestration=172.17.180.24;
tb3-bxb-vnfm1-di-internal1=192.168.1.12; tb3-bxb-vnfm1-service-network1=10.10.10.14,
10.10.10.6; tb3-bxb-vnfm1-service-network2=20.20.20.20, 20.20.20.8;
tb3-bxb-vnfm1-di-internal2=192.168.2.7     |
| 535b0bca-d3c5-4d99-ba41-9953da6339f4 |
vnfd1-deployment_s8_0_1e0f3ebf-b6e0-4bfe-9b1c-985dc32e1519      | ACTIVE | -
Running      | tb3-bxb-autovnf1-uas-orchestration=172.17.180.18;
tb3-bxb-vnfm1-di-internal1=192.168.1.14; tb3-bxb-vnfm1-service-network1=10.10.10.17,
10.10.10.11; tb3-bxb-vnfm1-service-network2=20.20.20.17, 20.20.20.15;
tb3-bxb-vnfm1-di-internal2=192.168.2.9     |
| dfdffafb-a624-4063-bae6-63c4a757473f |
vnfd1-deployment_s9_0_26db8332-8dac-43fc-84c5-71a8b975fd17      | ACTIVE | -
Running      | tb3-bxb-autovnf1-uas-orchestration=172.17.180.22;
tb3-bxb-vnfm1-di-internal1=192.168.1.10; tb3-bxb-vnfm1-service-network1=10.10.10.21,
10.10.10.24; tb3-bxb-vnfm1-service-network2=20.20.20.23, 20.20.20.22;
tb3-bxb-vnfm1-di-internal2=192.168.2.19   |
+-----+-----+-----+-----+

```

## Checking Cinder Type

Log on to the server on which OSP-D is running to check the Cinder volume type by executing the following commands:

```

cd /home/stack
source ~/<stack_name>rc-core
cinder type-list

```

**Example command output:**

```

+-----+-----+-----+-----+
| ID                | Name | Description | Is_Public |
+-----+-----+-----+-----+
| 208ef179-dfe4-4735-8a96-e7beee472944 | LUKS | -           | True      |
+-----+-----+-----+-----+

```

```

cinder type-show LUKS

```

**Example command output:**

```

+-----+-----+-----+-----+
| Property          | Value
+-----+-----+-----+-----+
| description       | None
| extra_specs      | {}
| id                | bf855b0f-8b3f-42bf-9497-05013b4ddad9 |
+-----+-----+-----+-----+

```

```

| is_public          | True          |
| name              | LUKS         |
| os-volume-type-access:is_public | True         |
| qos_specs_id     | None         |
+-----+-----+

```

## Checking Core Project (Tenant) and User Core

Log on to the server on which OSP-D is running to check the core projects and users by executing the following commands:

```

cd /home/stack
source ~/<stack_name> rc-core
openstack project list

```

**Example command output:**

```

+-----+-----+
| ID              | Name        |
+-----+-----+
| 271ab207a197465f9d166c2dc7304b18 | core       |
| 52547e0fca994cd682aa733b941d0f68 | service   |
| 9543ad9db4dd422ea5aedf04756d3682 | admin     |
+-----+-----+

```

```

openstack project show core

```

**Example command output:**

```

+-----+-----+
| Field          | Value      |
+-----+-----+
| description    | core tenant |
| enabled       | True       |
| id            | 271ab207a197465f9d166c2dc7304b18 |
| name          | core       |
| properties    |            |
+-----+-----+

```

```

openstack project show service

```

**Example command output:**

```

+-----+-----+
| Field          | Value      |
+-----+-----+
| description    | Tenant for the openstack services |
| enabled       | True       |
| id            | 52547e0fca994cd682aa733b941d0f68 |
| name          | service    |
| properties    |            |
+-----+-----+

```

```

openstack project show admin

```

**Example command output:**

```

+-----+-----+
| Field          | Value      |
+-----+-----+
| description    | admin tenant |
| enabled       | True       |
| id            | 9543ad9db4dd422ea5aedf04756d3682 |
| name          | admin      |
+-----+-----+

```

```
| properties |
+-----+
```

### openstack user list

#### Example command output:

```
+-----+
| ID | Name |
+-----+
| 1ac7208b033a41ccba805d86bf60dbb7 | admin |
| a6adac4ee79c4206a29de5165d7c7a6a | neutron |
| 79da40fe88c64de7a93bc691a42926ea | heat |
| 525048a99816474d91d692d9516e951c | nova |
| 8d6688db8d19411080eeb4c84c1d586b | glance |
| 9aadd12171474d1e8bcacf890e070ab | cinder |
| d2ee641a72c4493995de70a1a9671f2b | heat-cfn |
| 7fbb088c15e1428ab6ce677aad5415f4 | swift |
| 828cbf69cf564747a81bb313208a1c21 | core |
| 40563efc469d4c1295de0d6d4cf545c2 | tom |
+-----+
```

### openstack user show core

#### Example command output:

```
+-----+
| Field | Value |
+-----+
| email | None |
| enabled | True |
| id | 828cbf69cf564747a81bb313208a1c21 |
| name | core |
| project_id | 271ab207a197465f9d166c2dc7304b18 |
| username | core |
+-----+
```

### openstack role list

#### Example command output:

```
+-----+
| ID | Name |
+-----+
| 315d3058519a4b1a9385e11aa5ffe25b | admin |
| 585de968688e4257bc76f6dec13752cb | ResellerAdmin |
| 9717fe8079ba49e9ba9eadd5a37689e7 | swiftoperator |
| 9fe2ff9ee4384b1894a90878d3e92bab | _member_ |
| d75dcf507bfa4a6abee3aee3bb0323c6 | heat_stack_user |
+-----+
```

### openstack role show admin

#### Example command output:

```
+-----+
| Field | Value |
+-----+
| domain_id | None |
| id | 315d3058519a4b1a9385e11aa5ffe25b |
| name | admin |
+-----+
```

## Checking Nova/Neutron Security Groups

Log on to the server on which OSP-D is running to check Nova and Neutron security groups by executing the following commands:

**nova secgroup-list**

### Example command output:

WARNING: Command secgroup-list is deprecated and will be removed after Nova 15.0.0 is released. Use python-neutronclient or python-openstackclient instead.

```

+-----+-----+-----+
| Id                | Name    | Description                |
+-----+-----+-----+
| ce308d67-7645-43c1-a83e-89d3871141a2 | default | Default security group |
+-----+-----+-----+

```

**neutron security-group-list**

### Example command output:

```

+-----+-----+-----+
| id                | name    | security_group_rules      |
+-----+-----+-----+
| 4007a7a4-e7fa-4ad6-bc74-fc0b20f0b60c | default | egress, IPv4              |
|                                     |         | egress, IPv6              |
|                                     |         | ingress, IPv4, remote_group_id: |
4007a7a4-e7fa-4ad6-bc74-fc0b20f0b60c |         | ingress, IPv6, remote_group_id: |
4007a7a4-e7fa-4ad6-bc74-fc0b20f0b60c |         |                               |
| 8bee29ae-88c0-4d5d-b27a-a123f20b6858 | default | egress, IPv4              |
|                                     |         | egress, IPv6              |
|                                     |         | ingress, IPv4, 1-65535/tcp,    |
remote_ip_prefix: 0.0.0.0/0 |         | ingress, IPv4, 1-65535/udp,    |
|                                     |         |                               |
remote_ip_prefix: 0.0.0.0/0 |         | ingress, IPv4, icmp, remote_ip_prefix: |
0.0.0.0/0 |         | ingress, IPv4, remote_group_id: |
8bee29ae-88c0-4d5d-b27a-a123f20b6858 |         | ingress, IPv6, remote_group_id: |
| 8bee29ae-88c0-4d5d-b27a-a123f20b6858 |         |                               |
| b6b27428-35a3-4be4-af9b-38559132d28e | default | egress, IPv4              |
|                                     |         | egress, IPv6              |
|                                     |         | ingress, IPv4, remote_group_id: |
b6b27428-35a3-4be4-af9b-38559132d28e |         | ingress, IPv6, remote_group_id: |
| b6b27428-35a3-4be4-af9b-38559132d28e |         |                               |
| ce308d67-7645-43c1-a83e-89d3871141a2 | default | egress, IPv4              |
|                                     |         | egress, IPv6              |
|                                     |         | ingress, IPv4, 1-65535/tcp,    |
remote_ip_prefix: 0.0.0.0/0 |         | ingress, IPv4, 1-65535/udp,    |
|                                     |         |                               |
remote_ip_prefix: 0.0.0.0/0 |         | ingress, IPv4, icmp, remote_ip_prefix: |
0.0.0.0/0 |         |

```



```

|                                     | ingress, IPv4, remote_group_id:
ce308d67-7645-43c1-a83e-89d3871141a2 |
|                                     | ingress, IPv6, remote_group_id:
ce308d67-7645-43c1-a83e-89d3871141a2 |
+-----+-----+-----+-----+

```

**neutron security-group-show ce308d67-7645-43c1-a83e-89d3871141a2**

**Example command output:**

```

+-----+-----+-----+-----+
| Field          | Value
+-----+-----+-----+-----+
| created_at     | 2017-06-03T04:57:01Z
| description    | Default security group
| id            | ce308d67-7645-43c1-a83e-89d3871141a2
| name          | default
| project_id    | 271ab207a197465f9d166c2dc7304b18
| revision_number | 4
| security_group_rules | {
|               |   {
|               |     "remote_group_id": null,
|               |     "direction": "egress",
|               |     "protocol": null,
|               |     "description": null,
|               |     "ethertype": "IPv4",
|               |     "remote_ip_prefix": null,
|               |     "port_range_max": null,
|               |     "updated_at": "2017-06-03T04:57:01Z",
|               |     "security_group_id": "ce308d67-7645-43c1-a83e-89d3871141a2",
|               |     "port_range_min": null,
|               |     "revision_number": 1,
|               |     "tenant_id": "271ab207a197465f9d166c2dc7304b18",
|               |     "created_at": "2017-06-03T04:57:01Z",
|               |     "project_id": "271ab207a197465f9d166c2dc7304b18",
|               |     "id": "337838dd-0612-47f8-99e8-7d4f58dc09d6"
|               |   }
|               |   {
|               |     "remote_group_id": null,

```

```
|         |     "direction": "ingress",
|         |     "protocol": "udp",
|         |     "description": "",
|         |     "ethertype": "IPv4",
|         |     "remote_ip_prefix": "0.0.0.0/0",
|         |     "port_range_max": 65535,
|         |     "updated_at": "2017-06-03T04:57:20Z",
|         |     "security_group_id": "ce308d67-7645-43c1-a83e-89d3871141a2",
|         |     "port_range_min": 1,
|         |     "revision_number": 1,
|         |     "tenant_id": "271ab207a197465f9d166c2dc7304b18",
|         |     "created_at": "2017-06-03T04:57:20Z",
|         |     "project_id": "271ab207a197465f9d166c2dc7304b18",
|         |     "id": "48b04902-d617-4e25-ad0d-4d087128f3b9"
|         | }
|         | {
|         |     "remote_group_id": null,
|         |     "direction": "ingress",
|         |     "protocol": "icmp",
|         |     "description": "",
|         |     "ethertype": "IPv4",
|         |     "remote_ip_prefix": "0.0.0.0/0",
|         |     "port_range_max": null,
|         |     "updated_at": "2017-06-03T04:57:33Z",
|         |     "security_group_id": "ce308d67-7645-43c1-a83e-89d3871141a2",
|         |     "port_range_min": null,
|         |     "revision_number": 1,
|         |     "tenant_id": "271ab207a197465f9d166c2dc7304b18",
|         |     "created_at": "2017-06-03T04:57:33Z",
|         |     "project_id": "271ab207a197465f9d166c2dc7304b18",
|         |     "id": "68913f31-6788-4473-8b3b-90a264e9ef62"
|         | }
```

```
|
|                                     | {
|                                     |
|                                     |     "remote_group_id": null,
|                                     |     "direction": "ingress",
|                                     |     "protocol": "tcp",
|                                     |     "description": "",
|                                     |     "ethertype": "IPv4",
|                                     |     "remote_ip_prefix": "0.0.0.0/0",
|                                     |     "port_range_max": 65535,
|                                     |     "updated_at": "2017-06-03T04:57:02Z",
|                                     |     "security_group_id": "ce308d67-7645-43c1-a83e-89d3871141a2",
|                                     |     "port_range_min": 1,
|                                     |     "revision_number": 1,
|                                     |     "tenant_id": "271ab207a197465f9d166c2dc7304b18",
|                                     |     "created_at": "2017-06-03T04:57:02Z",
|                                     |     "project_id": "271ab207a197465f9d166c2dc7304b18",
|                                     |     "id": "85ece95b-d361-4986-8db0-78d1a404dd3c"
|                                     | }
|                                     |
|                                     | {
|                                     |
|                                     |     "remote_group_id": null,
|                                     |     "direction": "egress",
|                                     |     "protocol": null,
|                                     |     "description": null,
|                                     |     "ethertype": "IPv6",
|                                     |     "remote_ip_prefix": null,
|                                     |     "port_range_max": null,
|                                     |     "updated_at": "2017-06-03T04:57:01Z",
|                                     |     "security_group_id": "ce308d67-7645-43c1-a83e-89d3871141a2",
|                                     |     "port_range_min": null,
|                                     |     "revision_number": 1,
|                                     |     "tenant_id": "271ab207a197465f9d166c2dc7304b18",
|                                     |     "created_at": "2017-06-03T04:57:01Z",
|                                     |     "project_id": "271ab207a197465f9d166c2dc7304b18",
```



```

|                                     |      "created_at": "2017-06-03T04:57:01Z",
|                                     |
|                                     |      "project_id": "271ab207a197465f9d166c2dc7304b18",
|                                     |
|                                     |      "id": "deb7752c-e642-462e-92f0-5dff983f0739"
|                                     | }
| tenant_id                          | 271ab207a197465f9d166c2dc7304b18
| updated_at                          | 2017-06-03T04:57:33Z
+-----+-----+

```

## Checking Tenant Project Default Quotas

Log on to the server on which OSP-D is running to check default project quotas by executing the following commands:

**nova quota-show**

**Example command output:**

```

+-----+-----+
| Quota                | Limit |
+-----+-----+
| instances            | 1000  |
| cores                | 1000  |
| ram                  | 51200000 |
| metadata_items      | 128   |
| injected_files       | 100   |
| injected_file_content_bytes | 1024000 |
| injected_file_path_bytes | 255   |
| key_pairs            | 100   |
| server_groups        | 10    |
| server_group_members | 10    |
+-----+-----+

```

**openstack project list | grep core**

**Example command output:**

```
| 271ab207a197465f9d166c2dc7304b18 | core |
```

**nova quota-class-show 271ab207a197465f9d166c2dc7304b18**

**Example command output:**

```

+-----+-----+
| Quota                | Limit |
+-----+-----+
| instances            | 10    |
| cores                | 20    |
| ram                  | 51200 |
| floating_ips         | 10    |
| fixed_ips            | -1    |
| metadata_items      | 128   |
| injected_files       | 5     |
| injected_file_content_bytes | 10240 |
| injected_file_path_bytes | 255   |
| key_pairs            | 100   |
| security_groups      | 10    |
| security_group_rules | 20    |
+-----+-----+

```

**neutron quota-show****Example command output:**

```
+-----+-----+
| Field          | Value |
+-----+-----+
| floatingip     | 100   |
| network        | 1000  |
| port           | 4092  |
| rbac_policy    | 10    |
| router         | 100   |
| security_group | 100   |
| security_group_rule | 300  |
| subnet         | 1000  |
| subnetpool     | -1    |
| trunk          | -1    |
+-----+-----+
```

**openstack project list | grep core****Example command output:**

```
| 271ab207a197465f9d166c2dc7304b18 | core |
```

**cinder quota-show 271ab207a197465f9d166c2dc7304b18****Example command output:**

```
+-----+-----+
| Property       | Value |
+-----+-----+
| backup_gigabytes | 1000  |
| backups         | 10    |
| gigabytes       | 8092  |
| gigabytes_LUKS  | -1    |
| per_volume_gigabytes | -1   |
| snapshots       | 300   |
| snapshots_LUKS  | -1    |
| volumes         | 500   |
| volumes_LUKS    | -1    |
+-----+-----+
```

## Checking the Nova Hypervisor List

Log on to the server on which OSP-D is running to check the status of nova api on all compute nodes by executing the following command:

**nova hypervisor-list****Example command output:**

```
+-----+-----+-----+-----+
| ID | Hypervisor hostname | State | Status |
+-----+-----+-----+-----+
| 3  | tb3-ultram-compute-7.localdomain | up   | enabled |
| 6  | tb3-ultram-compute-6.localdomain | up   | enabled |
| 9  | tb3-ultram-osd-compute-0.localdomain | up   | enabled |
| 12 | tb3-ultram-compute-9.localdomain | up   | enabled |
| 15 | tb3-ultram-compute-0.localdomain | up   | enabled |
| 18 | tb3-ultram-compute-14.localdomain | up   | enabled |
| 21 | tb3-ultram-compute-2.localdomain | up   | enabled |
| 24 | tb3-ultram-compute-8.localdomain | up   | enabled |
| 27 | tb3-ultram-compute-13.localdomain | up   | enabled |
| 30 | tb3-ultram-compute-15.localdomain | up   | enabled |
+-----+-----+-----+-----+
```

```

| 33 | tb3-ultram-compute-12.localdomain | up | enabled |
| 36 | tb3-ultram-compute-5.localdomain | up | enabled |
| 39 | tb3-ultram-osd-compute-1.localdomain | up | enabled |
| 42 | tb3-ultram-compute-10.localdomain | up | enabled |
| 45 | tb3-ultram-compute-11.localdomain | up | enabled |
| 48 | tb3-ultram-compute-3.localdomain | up | enabled |
| 51 | tb3-ultram-osd-compute-2.localdomain | up | enabled |
| 54 | tb3-ultram-compute-4.localdomain | up | enabled |
| 57 | tb3-ultram-compute-1.localdomain | up | enabled |
+-----+-----+-----+-----+

```

## Checking the Router Main Configuration

Log on to the server on which OSP-D is running to check the Neutron router by entering the following commands:

**neutron router-list**

**Example command output:**

```

+-----+-----+-----+-----+-----+-----+
| id | distributed | ha | name | external_gateway_info |
+-----+-----+-----+-----+-----+
| 2d0cdee4-bb5e-415b-921c-97caf0aa0cd1 | main | {"network_id": "1c46790f-cab5-4b1d-afc7-a637fe2dbe08", "enable_snat": true, "external_fixed_ips": [{"subnet_id": "a23a740e-3ad0-4fb1-8526-3353dfd0010f", "ip_address": "10.169.127.176"}]} |
+-----+-----+-----+-----+-----+

```

```
[stack@lbucs001-ospd ~]$ neutron router-show 2d0cdee4-bb5e-415b-921c-97caf0aa0cd1
```

**Example command output:**

```

+-----+-----+-----+-----+
| Field | Value |
+-----+-----+-----+-----+
| admin_state_up | True |
| availability_zone_hints | |
| availability_zones | nova |
| created_at | 2017-06-03T05:05:08Z |
| description | |
| distributed | False |
| external_gateway_info | {"network_id": "1c46790f-cab5-4b1d-afc7-a637fe2dbe08", "enable_snat": true, "external_fixed_ips": [{"subnet_id": "a23a740e-3ad0-4fb1-8526-3353dfd0010f", "ip_address": "10.169.127.176"}]} |
| flavor_id | |
| ha | True |
+-----+-----+-----+-----+

```

```

| id | 2d0cdee4-bb5e-415b-921c-97caf0aa0cd1 |
| name | main |
| project_id | 271ab207a197465f9d166c2dc7304b18 |
| revision_number | 94 |
| routes | |
| status | ACTIVE |
| tenant_id | 271ab207a197465f9d166c2dc7304b18 |
| updated_at | 2017-07-28T00:44:27Z |

```

## Checking the External Network Using the core-project-id

Log on to the server on which OSP-D is running to check the external network configuration by entering the following commands:

```
neutron net-list
```

**Example command output:**

```

+-----+-----+-----+
| id | name |
| subnets |
+-----+-----+-----+
| 1236bd98-5389-42f9-bac8-433997525549 | LBUCS001-AUTOIT-MGMT |
| c63451f2-7e44-432e-94fc-167f6a31e4aa | 172.16.182.0/24 |
| 1c46790f-cab5-4b1d-afc7-a637fe2dbe08 | LBUCS001-EXTERNAL-MGMT |
| a23a740e-3ad0-4fb1-8526-3353dfd0010f | 10.169.127.160/27 |
| 1c70a9ab-212e-4884-b7d5-4749c44a87b6 | LBPGW101-DI-INTERNAL1 |
| e619b02e-84e0-48d9-9096-f16adc84f1cc | HA network tenant 271ab207a197465f9d166c2dc7304b18 |
| cefd5f5f-0c97-4027-b385-ca1a57f2cfac | 169.254.192.0/18 |
+-----+-----+-----+

```

```
neutron net-show 1c46790f-cab5-4b1d-afc7-a637fe2dbe08
```

**Example command output:**

```

+-----+-----+
| Field | Value |
+-----+-----+
| admin_state_up | True |
| availability_zone_hints | |
| availability_zones | |
| created_at | 2017-06-05T07:18:59Z |
| description | |
| id | 1c46790f-cab5-4b1d-afc7-a637fe2dbe08 |
| ipv4_address_scope | |
| ipv6_address_scope | |
| is_default | False |
| mtu | 1500 |
| name | LBUCS001-EXTERNAL-MGMT |
| port_security_enabled | True |
| project_id | 271ab207a197465f9d166c2dc7304b18 |
| provider:network_type | vlan |

```



```

| provider:physical_network | datacentre |
| provider:segmentation_id | 101 |
| qos_policy_id | |
| revision_number | 6 |
| router:external | True |
| shared | False |
| status | ACTIVE |
| subnets | a23a740e-3ad0-4fb1-8526-3353dfd0010f |
| tags | |
| tenant_id | 271ab207a197465f9d166c2dc7304b18 |
| updated_at | 2017-06-05T07:22:51Z |
+-----+-----+

```

Note down the **provider:segmentation\_id**. In this example, 101 is the vlan for the external interface.

### neutron subnet-list

#### Example command output:

```

+-----+-----+-----+-----+
| id | allocation_pools | name | cidr |
+-----+-----+-----+-----+
| a23a740e-3ad0-4fb1-8526-3353dfd0010f | LBUCS001-EXTERNAL-MGMT | 10.169.127.160/27 | {"start": "10.169.127.168", "end": "10.169.127.190"} |
| c63451f2-7e44-432e-94fc-167f6a31e4aa | LBUCS001-AUTOIT-MGMT | 172.16.182.0/24 | {"start": "172.16.182.2", "end": "172.16.182.254"} |
| cefd5f5f-0c97-4027-b385-cala57f2cfac | HA subnet tenant | 169.254.192.0/18 | {"start": "169.254.192.1", "end": "169.254.255.254"} |
| | | 271ab207a197465f9d166c2dc7304b18 | |
+-----+-----+-----+-----+

```

### neutron subnet-show a23a740e-3ad0-4fb1-8526-3353dfd0010f

#### Example command output:

```

+-----+-----+
| Field | Value |
+-----+-----+
| allocation_pools | {"start": "10.169.127.168", "end": "10.169.127.190"} |
| cidr | 10.169.127.160/27 |
| created_at | 2017-06-05T07:22:51Z |
| description | |
| dns_nameservers | |
| enable_dhcp | False |
| gateway_ip | 10.169.127.163 |
| host_routes | |
| id | a23a740e-3ad0-4fb1-8526-3353dfd0010f |
| ip_version | 4 |
| ipv6_address_mode | |
| ipv6_ra_mode | |
| name | LBUCS001-EXTERNAL-MGMT |
| network_id | 1c46790f-cab5-4b1d-afc7-a637fe2dbe08 |
| project_id | 271ab207a197465f9d166c2dc7304b18 |
| revision_number | 2 |
| service_types | |
| subnetpool_id | |
| tenant_id | 271ab207a197465f9d166c2dc7304b18 |
| updated_at | 2017-06-05T07:22:51Z |
+-----+-----+

```

## Checking the Staging Network Configuration

Log on to the server on which OSP-D is running to check the staging network configuration by entering the following commands:

```
neutron subnet-show <ext-mgmt-id>
```

<ext-mgmt-id> is the ID for the external management interface as obtained through the **neutron subnet-list** command output.

### Example output:

```
+-----+-----+
| Field          | Value                                                                 |
+-----+-----+
| allocation_pools | {"start": "10.169.127.168", "end": "10.169.127.190"} |
| cidr            | 10.169.127.160/27 |
| created_at      | 2017-06-05T07:22:51Z |
| description     | |
| dns_nameservers | |
| enable_dhcp     | False |
| gateway_ip      | 10.169.127.163 |
| host_routes     | |
| id              | a23a740e-3ad0-4fb1-8526-3353dfd0010f |
| ip_version      | 4 |
| ipv6_address_mode | |
| ipv6_ra_mode    | |
| name            | LBUCS001-EXTERNAL-MGMT |
| network_id      | 1c46790f-cab5-4b1d-afc7-a637fe2dbe08 |
| project_id      | 271ab207a197465f9d166c2dc7304b18 |
| revision_number | 2 |
| service_types   | |
| subnetpool_id   | |
| tenant_id       | 271ab207a197465f9d166c2dc7304b18 |
| updated_at      | 2017-06-05T07:22:51Z |
+-----+-----+
```

```
neutron subnet-show <autoit-mgmt-id>
```

<autoit-mgmt-id> is the ID for the AutoIT management interface as obtained through the **neutron subnet-list** command output.

### Example output:

```
+-----+-----+
| Field          | Value                                                                 |
+-----+-----+
| allocation_pools | {"start": "172.16.182.2", "end": "172.16.182.254"} |
| cidr            | 172.16.182.0/24 |
| created_at      | 2017-06-05T07:41:45Z |
| description     | |
| dns_nameservers | |
| enable_dhcp     | True |
| gateway_ip      | 172.16.182.1 |
| host_routes     | |
| id              | c63451f2-7e44-432e-94fc-167f6a31e4aa |
| ip_version      | 4 |
| ipv6_address_mode | |
| ipv6_ra_mode    | |
| name            | LBUCS001-AUTOIT-MGMT |
| network_id      | 1236bd98-5389-42f9-bac8-433997525549 |
| project_id      | 271ab207a197465f9d166c2dc7304b18 |
| revision_number | 2 |
| service_types   | |
+-----+-----+
```

```

| subnetpool_id      |
| tenant_id         | 271ab207a197465f9d166c2dc7304b18 |
| updated_at        | 2017-06-05T07:41:45Z             |
+-----+-----+

```

## Checking the DI-Internal and Service Network Configurations

Log on to the server on which OSP-D is running to check the DI-internal and service network configuration by entering the following commands:

```
neutron net-list
```

**Example command output:**

```

+-----+-----+-----+-----+
| id                | name                |
| subnets          |                     |
+-----+-----+-----+-----+
| 1236bd98-5389-42f9-bac8-433997525549 | LBUCS001-AUTOIT-MGMT |
| c63451f2-7e44-432e-94fc-167f6a31e4aa | 172.16.182.0/24      |
| 1c46790f-cab5-4b1d-afc7-a637fe2dbe08 | LBUCS001-EXTERNAL-MGMT |
| a23a740e-3ad0-4fb1-8526-3353dfd0010f | 10.169.127.160/27   |
| 1c70a9ab-212e-4884-b7d5-4749c44a87b6 | LBPGW101-DI-INTERNAL1 |
| e619b02e-84e0-48d9-9096-f16adc84f1cc | HA network tenant 271ab207a197465f9d166c2dc7304b18 |
| cefd5f5f-0c97-4027-b385-ca1a57f2cfac | 169.254.192.0/18   |
+-----+-----+-----+-----+

```

```
neutron net-show LBPGW101-DI-INTERNAL1
```

**Example command output:**

```

+-----+-----+
| Field                | Value                |
+-----+-----+
| admin_state_up      | True                 |
| availability_zone_hints |                     |
| availability_zones   |                     |
| created_at          | 2017-07-28T22:25:53Z |
| description         |                     |
| id                   | 1c70a9ab-212e-4884-b7d5-4749c44a87b6 |
| ipv4_address_scope  |                     |
| ipv6_address_scope  |                     |
| mtu                  | 1500                 |
| name                 | LBPGW101-DI-INTERNAL1 |
| port_security_enabled | True                 |
| project_id           | 271ab207a197465f9d166c2dc7304b18 |
| provider:network_type | flat                 |
| provider:physical_network | phys_pcie1_0        |
| provider:segmentation_id |                     |
| qos_policy_id        |                     |
| revision_number      | 3                    |
| router:external      | False                |
| shared               | True                 |
| status               | ACTIVE               |
| subnets             |                     |
| tags                 |                     |
| tenant_id            | 271ab207a197465f9d166c2dc7304b18 |
| updated_at           | 2017-07-28T22:25:53Z |
+-----+-----+

```

```
neutron subnet-list
```

**Example command output:**

id	allocation_pools	name	cidr
96ae7e6e-f2e9-4fa5-a816-769c5a79f8f4	{ "start": "192.168.1.2", "end": "192.168.1.254" }	LBPGW101-DI-INTERNAL1-SUBNET	
a23a740e-3ad0-4fb1-8526-3353dfd0010f	{ "start": "10.169.127.168", "end": "10.169.127.190" }	LBUCS001-EXTERNAL-MGMT	
c63451f2-7e44-432e-94fc-167f6a31e4aa	{ "start": "172.16.182.2", "end": "172.16.182.254" }	LBUCS001-AUTOIT-MGMT	
cefd5f5f-0c97-4027-b385-ca1a57f2cfac	{ "start": "169.254.192.1", "end": "169.254.255.254" }	HA subnet tenant	271ab207a197465f9d166c2dc7304b18

## Checking the Flavor List

Log on to the server on which OSP-D is running to check the flavor list and to by entering the following command:

```
nova flavor-list
```

**Example command output:**

ID	Name	Memory_MB	Disk	Ephemeral
eff0335b-3374-46c3-a3de-9f4b1ccea04	DNUCS002-AUTOIT-FLAVOR	8192	80	0

## Checking Host Aggregate and Availability Zone Configuration

Log on to the server on which OSP-D is running to check the host aggregate and availability zone configurations for the OSD Compute and for the AutoDeploy and AutoIT VMs.



**Note** It is assumed that the AutoDeploy and AutoIT VMs reside on the same OSD Compute node.

This is done by executing the following commands:

```
cd /home/stack  
source ~/<stack_name>rc-core  
nova aggregate-list
```

**Example command output:**

Id	Name	Availability Zone
108	LBUCS001-AUTOIT	mgmt

```
| 147 | LBPGW101-EM-MGMT1 | - |
| 150 | LBPGW101-SERVICE1 | - |
| 153 | LBPGW101-CF-MGMT1 | - |
+-----+-----+-----+
```

#### **nova aggregate-show LBUCS001-AUTOIT**

```
+-----+-----+-----+-----+-----+
| Id | Name | Availability Zone | Hosts | Metadata |
+-----+-----+-----+-----+-----+
| 108 | LBUCS001-AUTOIT | mgmt | 'newtonoc-osd-compute-0.localdomain' | 'availability_zone=mgmt', 'mgmt=true' |
+-----+-----+-----+-----+-----+
```



**Note** This information can also be verified through the Horizon GUI. Login to Horizon as the user core and navigate to **Project > Compute > Instances**. Check each instance to verify that the status is Active and the power state is Running.

Correct any instance that does not meet these criteria before continuing.

## Checking Controller Server Health



**Note** The commands in this section should be executed on any one of the Controller nodes and do not need to be repeated on the other Controller nodes unless an issue is observed.

## Checking the Pacemaker Cluster Stack (PCS) Status

Log on to one of the Controller nodes and verify that the group of resources in the PCS cluster are active and in the expected state by executing the following command:

```
sudo pcs status
```

#### **Example command output:**

```
Cluster name: tripleo_cluster
Stack: corosync
Current DC: tb3-ultram-controller-0 (version 1.1.15-11.e17_3.4-e174ec8) - partition with quorum
Last updated: Wed Jul 12 13:28:56 2017 Last change: Tue Jul 11 21:45:09 2017 by root via crm_attribute on tb3-ultram-controller-0
```

```
3 nodes and 22 resources configured
```

```
Online: [ tb3-ultram-controller-0 tb3-ultram-controller-1 tb3-ultram-controller-2 ]
```

```
Full list of resources:
```

```
ip-192.200.0.104 (ocf::heartbeat:IPaddr2): Started tb3-ultram-controller-1
ip-10.84.123.6 (ocf::heartbeat:IPaddr2): Started tb3-ultram-controller-0
ip-11.119.0.42 (ocf::heartbeat:IPaddr2): Started tb3-ultram-controller-0
Clone Set: haproxy-clone [haproxy]
Started: [ tb3-ultram-controller-0 tb3-ultram-controller-1 tb3-ultram-controller-2 ]
Master/Slave Set: galera-master [galera]
```

```

Masters: [ tb3-ultram-controller-0 tb3-ultram-controller-1 tb3-ultram-controller-2 ]
ip-11.120.0.47 (ocf::heartbeat:IPAddr2):      Started tb3-ultram-controller-1
ip-11.118.0.49 (ocf::heartbeat:IPAddr2):      Started tb3-ultram-controller-0
Clone Set: rabbitmq-clone [rabbitmq]
  Started: [ tb3-ultram-controller-0 tb3-ultram-controller-1 tb3-ultram-controller-2 ]
ip-11.120.0.48 (ocf::heartbeat:IPAddr2):      Started tb3-ultram-controller-1
Master/Slave Set: redis-master [redis]
  Masters: [ tb3-ultram-controller-0 ]
  Slaves: [ tb3-ultram-controller-1 tb3-ultram-controller-2 ]
openstack-cinder-volume      (systemd:openstack-cinder-volume):      Started
tb3-ultram-controller-0
my-ipmilan-for-controller-0  (stonith:fence_ipmilan):      Started
tb3-ultram-controller-0
my-ipmilan-for-controller-1  (stonith:fence_ipmilan):      Started
tb3-ultram-controller-1
my-ipmilan-for-controller-2  (stonith:fence_ipmilan):      Started
tb3-ultram-controller-0

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled

```

From the output of this command, ensure that:

- All 3 controllers are listed as Online
- haproxy-clone is started on all 3 controllers
- galera-master lists all 3 controllers as Masters
- rabbitmq-clone is started on all 3 controllers
- redis-master lists one controller as master and the other 2 controllers as slaves
- openstack-cinder-volume is started on one node
- my-ipmilan/stonith is started on all 3 controllers
- Daemons corosync, pacemaker and pcsd are active and enabled




---

**Note** If the output displays any “Failed Actions”, execute the **sudo pcs resource cleanup** command and then re-execute the **sudo pcs status** command.

---

## Checking Ceph Storage Status

Log on to the Controller node and verify the health of the Ceph storage from the Controller node by executing the following command:

```
sudo ceph status
```

### Example command output:

```

cluster eb2bb192-b1c9-11e6-9205-525400330666
  health HEALTH_OK
  monmap e1: 3 mons at
{tb3-ultram-controller-0=11.118.0.10:6789/0,tb3-ultram-controller-1=11.118.0.11:6789/0,
tb3-ultram-controller-2=11.118.0.12:6789/0}
  election epoch 152, quorum 0,1,2
tb3-ultram-controller-0,tb3-ultram-controller-1,tb3-ultram-controller-2

```

```

osdmap e158: 12 osds: 12 up, 12 in
      flags sortbitwise,require_jewel_osds
pgmap v1417251: 704 pgs, 6 pools, 321 GB data, 110 kobjects
      961 GB used, 12431 GB / 13393 GB avail
      704 active+clean
client io 53755 B/s wr, 0 op/s rd, 7 op/s wr

```

From the output of this command, ensure that:

- health is listed as HEALTH\_OK
- The correct number of monitors are listed in the monmap
- The correct number of OSDs are listed in the osdmap

## Checking Controller Node Services

Log on to the Controller node and check the status of all services by executing the following command:

```
sudo systemctl list-units "openstack*" "neutron*" "openvswitch"
```

**Example command output:**

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
neutron-dhcp-agent.service Agent	loaded	active	running	OpenStack Neutron DHCP
neutron-l3-agent.service 3 Agent	loaded	active	running	OpenStack Neutron Layer
neutron-metadata-agent.service Agent	loaded	active	running	OpenStack Neutron Metadata
neutron-openvswitch-agent.service vSwitch Agent	loaded	active	running	OpenStack Neutron Open
neutron-ovs-cleanup.service vSwitch Cleanup Utility	loaded	active	exited	OpenStack Neutron Open
neutron-server.service	loaded	active	running	OpenStack Neutron Server
openstack-cinder-api.service Server	loaded	active	running	OpenStack Cinder API
openstack-cinder-scheduler.service Server	loaded	active	running	OpenStack Cinder Scheduler
openstack-cinder-volume.service openstack-cinder-volume	loaded	active	running	Cluster Controlled
openstack-glance-api.service (code-named Glance) API server	loaded	active	running	OpenStack Image Service
openstack-glance-registry.service (code-named Glance) Registry server	loaded	active	running	OpenStack Image Service
openstack-heat-api-cfn.service CFN-compatible API Service	loaded	active	running	Openstack Heat
openstack-heat-api-cloudwatch.service API Service	loaded	active	running	OpenStack Heat CloudWatch
openstack-heat-api.service	loaded	active	running	OpenStack Heat API Service
openstack-heat-engine.service Service	loaded	active	running	Openstack Heat Engine
openstack-nova-api.service	loaded	active	running	OpenStack Nova API Server
openstack-nova-conductor.service Server	loaded	active	running	OpenStack Nova Conductor
openstack-nova-consoleauth.service auth Server	loaded	active	running	OpenStack Nova VNC console
openstack-nova-novncproxy.service Proxy Server	loaded	active	running	OpenStack Nova NoVNC
openstack-nova-scheduler.service Server	loaded	active	running	OpenStack Nova Scheduler
openstack-swift-account-auditor.service (swift) - Account Auditor	loaded	active	running	OpenStack Object Storage
openstack-swift-account-reaper.service	loaded	active	running	OpenStack Object Storage

```

(swift) - Account Reaper
openstack-swift-account-replicator.service loaded active running OpenStack Object Storage
(swift) - Account Replicator
openstack-swift-account.service loaded active running OpenStack Object Storage
(swift) - Account Server
openstack-swift-container-auditor.service loaded active running OpenStack Object Storage
(swift) - Container Auditor
openstack-swift-container-replicator.service loaded active running OpenStack Object Storage
(swift) - Container Replicator
openstack-swift-container-updater.service loaded active running OpenStack Object Storage
(swift) - Container Updater
openstack-swift-container.service loaded active running OpenStack Object Storage
(swift) - Container Server
openstack-swift-object-auditor.service loaded active running OpenStack Object Storage
(swift) - Object Auditor
openstack-swift-object-expirer.service loaded active running OpenStack Object Storage
(swift) - Object Expirer
openstack-swift-object-replicator.service loaded active running OpenStack Object Storage
(swift) - Object Replicator
openstack-swift-object-updater.service loaded active running OpenStack Object Storage
(swift) - Object Updater
openstack-swift-object.service loaded active running OpenStack Object Storage
(swift) - Object Server
openstack-swift-proxy.service loaded active running OpenStack Object Storage
(swift) - Proxy Server
openvswitch.service loaded active exited Open vSwitch

```

LOAD = Reflects whether the unit definition was properly loaded.

ACTIVE = The high-level unit activation state, i.e. generalization of SUB.

SUB = The low-level unit activation state, values depend on unit type.

43 loaded units listed. Pass --all to see loaded but inactive units, too.  
To show all installed unit files use 'systemctl list-unit-files'.

## Check the RabbitMQ Database Status

From each of the controller nodes, determine if the rabbitmq database is in a good state by executing the following command:

```
sudo rabbitmqctl eval 'rabbit_diagnostics:maybe_stuck().'
```

### Example command output:

```

2017-07-20 01:58:02 There are 11020 processes.
2017-07-20 01:58:02 Investigated 0 processes this round, 5000ms to go.
2017-07-20 01:58:03 Investigated 0 processes this round, 4500ms to go.
2017-07-20 01:58:03 Investigated 0 processes this round, 4000ms to go.
2017-07-20 01:58:04 Investigated 0 processes this round, 3500ms to go.
2017-07-20 01:58:04 Investigated 0 processes this round, 3000ms to go.
2017-07-20 01:58:05 Investigated 0 processes this round, 2500ms to go.
2017-07-20 01:58:05 Investigated 0 processes this round, 2000ms to go.
2017-07-20 01:58:06 Investigated 0 processes this round, 1500ms to go.
2017-07-20 01:58:06 Investigated 0 processes this round, 1000ms to go.
2017-07-20 01:58:07 Investigated 0 processes this round, 500ms to go.
2017-07-20 01:58:07 Found 0 suspicious processes.
ok

```

If the database is healthy, the command returns “Found 0 suspicious processes.” If the database is not healthy, the command returns 1 or more suspicious processes. Contact your local support representative if suspicious processes are found.



## Checking OSD Compute Server Health

### Checking Ceph Status

Log on to the OSD Compute and check the Ceph storage status by executing the following command:

```
sudo ceph status
```

#### Example command output:

```
sudo ceph status
  cluster eb2bb192-b1c9-11e6-9205-525400330666
  health HEALTH_OK
  monmap e1: 3 mons at
{tb3-ultram-controller-0=11.118.0.10:6789/0,tb3-ultram-controller-1=11.118.0.11:6789/0,
tb3-ultram-controller-2=11.118.0.12:6789/0}
  election epoch 152, quorum 0,1,2
tb3-ultram-controller-0,tb3-ultram-controller-1,tb3-ultram-controller-2
  osdmap e158: 12 osds: 12 up, 12 in
  flags sortbitwise,require_jewel_osds
  pgmap v1417867: 704 pgs, 6 pools, 321 GB data, 110 kobjects
  961 GB used, 12431 GB / 13393 GB avail
  704 active+clean
  client io 170 kB/s wr, 0 op/s rd, 24 op/s wr
```

### Checking OSD Compute Node Services

Log on to each OSD Compute node and check the status of all services by executing the following command:

```
sudo systemctl list-units "openstack*" "neutron*" "openvswitch"
```

#### Example command output:

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
neutron-openvswitch-agent.service	loaded	active	running	OpenStack Neutron Open vSwitch Agent
neutron-ovs-cleanup.service	loaded	active	exited	OpenStack Neutron Open vSwitch Cleanup Utility
neutron-sriov-nic-agent.service	loaded	active	running	OpenStack Neutron SR-IOV NIC Agent
openstack-nova-compute.service	loaded	active	running	OpenStack Nova Compute Server
openvswitch.service	loaded	active	exited	Open vSwitch

LOAD = Reflects whether the unit definition was properly loaded.  
 ACTIVE = The high-level unit activation state, i.e. generalization of SUB.  
 SUB = The low-level unit activation state, values depend on unit type.

6 loaded units listed. Pass --all to see loaded but inactive units, too.  
 To show all installed unit files use 'systemctl list-unit-files'.

## Monitoring AutoDeploy Operations

This section identifies various commands that can be used to determine the status and health of AutoDeploy.

To use them, you must:

1. Log on to the AutoDeploy VM as *ubuntu*. Use the password that was created earlier for this user.
2. Become the root user.

```
sudo -i
```

## Viewing AutoDeploy Logs

AutoDeploy logs are available on the AutoDeploy VM in the following directory:

```
/var/log/upstart/autodeploy.log
```



**Important** To access the command used to view logs, you must be logged in to the Confd CLI as the *admin* user on the AutoDeploy VM:

```
confd_cli -u admin -C
```

When prompted, you must enter the *admin* user password.

## AutoDeploy Transaction Logs

Execute the following command to display AutoDeploy transaction logs:

```
show log $TX-ID | display xml
```

### Example VIM-ORCH and VIM Activation Log:

```
2018-01-23 22:01:56,266 - Send Deployment notification for: autoit-instance
2018-01-23 22:08:36,876 - Deployment activate-ns-deployment: autoit initiated
2018-01-23 22:08:36,919 - Send Deployment notification for: autoit-instance
2018-01-23 22:08:36,951 - Deployment activate-ns-deployment: autoit initiated
2018-01-23 22:08:37,004 - Send Deployment notification for: autoit-deploy
2018-01-23 22:08:37,029 - Image '/var/cisco/isos/rhel-server-7.3-x86_64-dvd.iso' exists
2018-01-23 22:08:37,134 - Send Deployment notification for: autoit-instance
2018-01-23 22:08:37,165 - Deployment activate-ns-deployment: autoit started
2018-01-23 22:08:37,181 - Adding NSR: autoit-instance
2018-01-23 22:08:37,215 - Start pipeline of 1 tasks
2018-01-23 22:08:37,257 - Scheduling Task: autoit
2018-01-23 22:08:37,269 - Waiting for all workers to finish the transactions
2018-01-23 22:08:37,364 - Send Deployment notification for: autoit-deploy
2018-01-23 22:08:37,387 - Deployment activate-ns-deployment: autoit started
2018-01-23 22:08:37,395 - Skipping VNF pre-deployment , since VNF is not defined
2018-01-23 22:08:37,424 - Skipping VNF-Package pre-deployment, since is not defined
2018-01-23 22:08:37,440 - Skipping VIM-Artifact pre-deployment, since VIM-Artifact is not
defined
2018-01-23 22:08:37,463 - VIM-Orchestrator deployment pre-check success, entry already
exists. Continuing...
2018-01-23 22:08:37,470 - VIM deployment pre-check success, entry already exists.
Continuing...
2018-01-23 22:08:37,501 - NS pre-check success
2018-01-23 22:08:37,513 - Copying '/var/cisco/isos/rhel-server-7.3-x86_64-dvd.iso' to
'/var/cisco/isos/underc_rhel-server-7.3-x86_64-dvd.iso'
/tmp/MEIulQrBS/Crypto/Cipher/blockalgo.py:141: FutureWarning: CTR mode needs counter
parameter, not IV
2018-01-23 22:09:00,685 - Connected to AutoIT[172.21.203.121]
2018-01-23 22:09:02,281 - Skipping VNFDs
2018-01-23 22:09:02,298 - Skipping VNF-PACKAGE
2018-01-23 22:09:02,314 - Skipping VIM-Artifact
2018-01-23 22:09:02,332 - XML:[<config>
  <nsd xmlns="http://www.cisco.com/usp/nfv/usp-nsds">
    <nsd-id>autoit</nsd-id>
    <vim-orch>underc</vim-orch>
```

```

    <vim>overc</vim>
  </nsd>
  <vim-orchd xmlns="http://www.cisco.com/usp/nfv/usp-vim-orch">
    <vim-orch-id>underc</vim-orch-id>
    <hostname>tb3-undercloud</hostname>
    <domain-name>cisco.com</domain-name>
  .
  .
  .
2018-01-23 22:38:53,531 - VIM-ORCH: in-progress:84/84
2018-01-23 22:38:53,781 - Received vim-orchestrator-deployment-event for
underc:1516745343-313472/1516745343-460684 with status:success
2018-01-23 22:38:53,811 - VIM-ORCH: success:None/None
2018-01-23 22:38:53,983 - Received vim-deployment-event for
overc:1516745343-313472/1516745343-581981 with status:in-progress
2018-01-23 22:38:54,426 - Received vim-deployment-event for
overc:1516745343-313472/1516745343-581981 with status:in-progress
2018-01-23 23:39:15,038 - Received vim-deployment-event for
overc:1516745343-313472/1516745343-581981 with status:success
2018-01-23 23:39:15,113 - Received ns-deployment-event for autoit:1516745343-313472 with
status:success
2018-01-23 23:39:15,167 - RPC NS[autoit:autoit-instance] success
2018-01-23 23:39:15,271 - Deployment activate-ns-deployment: autoit succeeded
2018-01-23 23:39:15,344 - Send Deployment notification for: autoit-deploy
No handlers could be found for logger "AutoVNF-Traces"
2018-01-23 23:39:15,518 - All workers finished the job
2018-01-23 23:39:15,532 - Deployment activate-ns-deployment: autoit succeeded
2018-01-23 23:39:15,571 - Send Deployment notification for: autoit-instance

```

#### Example Tenant Creation Log:

```

2018-01-23 23:48:54,420 - Deployment activate-ns-deployment: autoit initiated
2018-01-23 23:48:54,449 - Send Deployment notification for: autoit-instance
2018-01-23 23:48:54,465 - Parsing role for tenant 'sjccore'
2018-01-23 23:48:54,473 - Parsing credentials for tenant 'sjccore'
2018-01-23 23:48:54,484 - Parsing attributes for tenant 'sjccore'
2018-01-23 23:48:54,540 - Deployment activate-ns-deployment: autoit initiated
2018-01-23 23:48:54,574 - Send Deployment notification for: autoit-deploy
2018-01-23 23:48:54,599 - Image '/var/cisco/isos/rhel-server-7.3-x86_64-dvd.iso' exists
2018-01-23 23:48:54,666 - Send Deployment notification for: autoit-instance
2018-01-23 23:48:54,689 - Deployment activate-ns-deployment: autoit started
2018-01-23 23:48:54,691 - Adding NSR: autoit-instance
2018-01-23 23:48:54,712 - Start pipeline of 1 tasks
2018-01-23 23:48:54,723 - Scheduling Task: autoit
2018-01-23 23:48:54,749 - Waiting for all workers to finish the transactions
2018-01-23 23:48:54,804 - Send Deployment notification for: autoit-deploy
2018-01-23 23:48:54,806 - Deployment activate-ns-deployment: autoit started
2018-01-23 23:48:54,822 - Skipping VNF pre-deployment , since VNFD is not defined
2018-01-23 23:48:54,829 - Skipping VNF-Package pre-deployment, since is not defined
2018-01-23 23:48:54,862 - VIM-Artifact deployment pre-check success
2018-01-23 23:48:54,866 - VIM-Orchestrator deployment pre-check success, entry already
exists. Continuing...
2018-01-23 23:48:54,879 - VIM deployment pre-check success, entry already exists.
Continuing...
2018-01-23 23:48:54,885 - NS pre-check success
2018-01-23 23:48:54,895 - Skipping copy, file
'/var/cisco/isos/underc_rhel-cisver-7.3-x86_64-dvd.iso' already exists
/tmp/_MEIulQrBS/Crypto/Cipher/blockalgo.py:141: FutureWarning: CTR mode needs counter
parameter, not IV
2018-01-23 23:48:55,244 - Connected to AutoIT[172.21.203.121]
2018-01-23 23:48:55,259 - Skipping VNFDs
2018-01-23 23:48:55,274 - Skipping VNF-PACKAGE
2018-01-23 23:48:55,279 - XML:[<config>
  <nsd xmlns="http://www.cisco.com/usp/nfv/usp-nsds">
    <nsd-id>autoit</nsd-id>

```

```

<vim-identity>vim1</vim-identity>
.
.
.
2018-01-23 23:48:56,419 - Received vim-orchestrator-deployment-event for
underc:1516751336-209342/1516751336-428695 with status:success
2018-01-23 23:48:56,441 - VIM-ORCH: success:None/None
2018-01-23 23:48:56,540 - Received vim-deployment-event for
overc:1516751336-209342/1516751336-532373 with status:in-progress
2018-01-23 23:48:56,671 - Received vim-deployment-event for
overc:1516751336-209342/1516751336-532373 with status:success
2018-01-23 23:48:56,802 - Received vim-deployment-event for
sjccore:1516751336-209342/1516751336-654858 with status:in-progress
2018-01-23 23:49:13,305 - Received vim-deployment-event for
sjccore:1516751336-209342/1516751336-654858 with status:success
2018-01-23 23:49:13,387 - Received ns-deployment-event for autoit:1516751336-209342 with
status:success
2018-01-23 23:49:13,414 - RPC NS[autoit:autoit-instance] success
2018-01-23 23:49:13,496 - Deployment activate-ns-deployment: autoit succeeded
2018-01-23 23:49:13,540 - Send Deployment notification for: autoit-deploy
No handlers could be found for logger "AutoVNF-Traces"
2018-01-23 23:49:13,670 - All workers finished the job
2018-01-23 23:49:13,689 - Deployment activate-ns-deployment: autoit succeeded
2018-01-23 23:49:13,723 - Send Deployment notification for: autoit-instance

```

### Example AutoVNF Creation Log:

```

<config xmlns="http://tail-f.com/ns/config/1.0">
  <log xmlns="http://www.cisco.com/usp/nfv/usp-transaction">
    <tx-id>1516900912-955117</tx-id>
    <log>
      2018-01-25 17:21:54,162 - Send Deployment notification for: autoit-instance
      2018-01-25 17:21:54,195 - Deployment activate-ns-deployment: autoit started
      2018-01-25 17:21:54,225 - Adding NSR: autoit-instance
      2018-01-25 17:21:54,288 - Start pipeline of 1 tasks
      2018-01-25 17:21:54,312 - Scheduling Task: autoit
      2018-01-25 17:21:54,342 - Waiting for all workers to finish the transactions
      2018-01-25 17:23:19,325 - All workers finished the job
      2018-01-25 17:23:19,365 - Deployment activate-ns-deployment: autoit succeeded
      2018-01-25 17:23:19,517 - Send Deployment notification for: autoit-instance
      2018-01-25 17:24:28,117 - Deployment activate-ns-deployment: tb3-autovnf_vpc initiated
      2018-01-25 17:24:28,209 - Send Deployment notification for: tb3-autovnf_vpc-instance
      2018-01-25 17:21:54,505 - Send Deployment notification for: autoit-deploy
      2018-01-25 17:21:54,550 - Deployment activate-vnf-deployment: autoit started
      2018-01-25 17:21:54,588 - Adding NSR: autoit-instance, VNFR: autoit-tb3-autovnf1, vlrs:
      None
      2018-01-25 17:21:54,661 - VNF deployment pre-check success(all-not-present)
      2018-01-25 17:21:55,001 - Connected to AutoIT[10.84.123.51]
      2018-01-25 17:21:55,039 - XML:[&lt;config>
        &lt;nsd xmlns="http://www.cisco.com/usp/nfv/usp-nsds">
          &lt;nsd-id>autoit&lt;/nsd-id>
          &lt;vim-identity>vim2&lt;/vim-identity>
        .
        .
        .
        2018-01-25 17:25:04,646 - &lt;?xml version="1.0" encoding="UTF-8"?>
        &lt;rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
        message-id="urn:uuid:1d0dd00b-a3a9-4e10-9a71-376680d05dca"
        xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">&lt;transaction-id
        xmlns='http://www.cisco.com/usp/nfv/usp-nsds'>1516901142-922838&lt;/transaction-id>
        &lt;/rpc-reply>
        2018-01-25 17:25:04,736 - Waiting for deployment notifications for tx-id '1516901142-922838'
        2018-01-25 17:25:04,816 - Received ns-deployment-event for tb3-autovnf_vpc:1516901142-922838

```

```

with status:requested
2018-01-25 17:25:04,851 - Received vim-deployment-event for
tb3-vnfl-rack:1516901142-922838/1516901143-301032 with status:requested
2018-01-25 17:25:04,908 - VIM: requested:None/None
2018-01-25 17:25:04,977 - Received vnf-package-deployment-event for
usp_6_0:1516901142-922838/1516901143-337769 with status:requested
2018-01-25 17:25:05,034 - VNF-PKG[usp_6_0]: requested, activate-vnf-package
2018-01-25 17:25:05,118 - Received vnf-deployment-event for
esc:1516901142-922838/1516901143-372586 with status:requested
2018-01-25 17:25:05,166 - Received vnf-deployment-event for
vpc:1516901142-922838/1516901143-418832 with status:requested
2018-01-25 17:25:05,201 - Received ns-deployment-event for tb3-autovnf_vpc:1516901142-922838
with status:in-progress
2018-01-25 17:25:05,235 - Received vim-deployment-event for
tb3-vnfl-rack:1516901142-922838/1516901143-301032 with status:in-progress
2018-01-25 17:25:05,269 - VIM: in-progress:None/None
2018-01-25 17:25:15,753 - Received vim-deployment-event for
tb3-vnfl-rack:1516901142-922838/1516901143-301032 with status:success
2018-01-25 17:25:15,786 - VIM: success:None/None
2018-01-25 17:25:15,889 - Received vnf-package-deployment-event for
usp_6_0:1516901142-922838/1516901143-337769 with status:in-progress
2018-01-25 17:25:15,927 - VNF-PKG[usp_6_0]: in-progress, activate-vnf-package
2018-01-25 17:27:44,479 - Received vnf-package-deployment-event for
usp_6_0:1516901142-922838/1516901143-337769 with status:success
2018-01-25 17:27:44,566 - VNF-PKG[usp_6_0]: success, activate-vnf-package
2018-01-25 17:27:44,624 - Received vnf-deployment-event for
esc:1516901142-922838/1516901143-372586 with status:in-progress
2018-01-25 17:31:13,916 - Received vnf-deployment-event for
esc:1516901142-922838/1516901143-372586 with status:success
2018-01-25 17:31:13,972 - Received vnf-deployment-event for
vpc:1516901142-922838/1516901143-418832 with status:in-progress
2018-01-25 17:45:29,291 - Received vnf-deployment-event for
vpc:1516901142-922838/1516901143-418832 with status:success
2018-01-25 17:45:29,318 - Received ns-deployment-event for tb3-autovnf_vpc:1516901142-922838
with status:success
2018-01-25 17:45:29,382 - RPC NS[tb3-autovnf_vpc:tb3-autovnf_vpc-instance] success
2018-01-25 17:45:30,000 - Deployment activate-ns-deployment: tb3-autovnf_vpc succeeded
2018-01-25 17:45:30,141 - Send Deployment notification for: tb3-autovnf_vpc-deploy</log>
</log>
</config>

```

## Checking AutoDeploy Processes

Check the status of AutoDeploy VM by entering the following commands:

```
service autodeploy status
```

```
service uas-confd status
```

## Determining the Running AutoDeploy Version

To display the version of the AutoDeploy software role that is currently operational:

```
show uas
```

**Example output:**

```

uas version                6.0.0
uas state                  active
uas external-connection-point 172.28.185.132
INSTANCE IP      STATE  ROLE
-----

```

```
172.28.185.133 alive CONF-D-MASTER
172.28.185.134 alive CONF-D-SLAVE
```

NAME	LAST HEARTBEAT
AutoDeploy-MASTER	2018-01-24 21:29:54
USPCFMWorker	2018-01-24 21:29:45
USPCHBWorker	2018-01-24 21:29:45
USPCWorker	2018-01-24 21:29:45

## Monitoring AutoIT Operations

This section identifies various commands that can be used to determine the status and health of AutoIT.

To use them, you must:

1. Log on to the AutoIT VM as *ubuntu*. Use the password that was created earlier for this user.
2. Become the *root* user.

```
sudo -i
```

## Viewing AutoIT Logs

AutoIT maintains logs containing information pertaining to UAS deployment and termination transactions. The *autoit.log* file is located in the following directory on the Ultra M Manager Node:

```
/var/log/cisco/usp/auto-it/autoit.log
```

### Example Deployment Log:

```
tail -100f /var/log/cisco/usp/auto-it/autoit.log &^C
```

```
2017-05-25 22:04:57,527 - INFO: Received a request to list config folder names.
2017-05-25 22:04:57,527 - INFO: config contents are:
2017-05-25 22:04:57,536 - INFO: Received a request to list config folder names.
2017-05-25 22:04:57,536 - INFO: config contents are:
2017-05-25 22:04:57,545 - INFO: Received a request to create a configuration folder.
2017-05-25 22:04:57,551 - INFO: Received a request to create a configuration folder.
2017-05-25 22:04:57,553 - INFO: Received request to download package: system.cfg from ISO
2017-05-25 22:04:57,563 - INFO: Received request to download package: system.cfg from ISO
2017-05-25 22:04:57,565 - INFO: Received request to download package: system.cfg from ISO
2017-05-25 22:04:57,566 - INFO: Received request to upload config file system.cfg to config
named vnf-pkg1
2017-05-25 22:04:57,567 - INFO: Uploaded file system.cfg to config named vnf-pkg1

2017-05-25 22:05:54,268 - INFO: Received request to upload ISO usp-5_1_0.iso
2017-05-25 22:05:54,268 - INFO: Saving ISO to /tmp/tmpxu7Mu0/usp-5_1_0.iso
2017-05-25 22:06:30,678 - INFO: Mounting ISO to /tmp/tmpxu7Mu0/iso_mount
2017-05-25 22:06:30,736 - INFO: ISO version already installed, (5.1.0-662)
2017-05-25 22:06:31,355 - INFO: Received a request to list file names in config named
vnf-pkg1.
2017-05-25 22:06:31,355 - INFO: config contents are: system.cfg
2017-05-25 22:06:31,362 - INFO: Received a request to list file names in config named
vnf-pkg1-images.
2017-05-25 22:06:31,362 - INFO: config contents are:
2017-05-25 22:06:31,370 - INFO: Received request to get ISO details 5.1.0-662
2017-05-25 22:06:31,391 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:06:31,857 - INFO: Getting Host Aggregate failed: Aggregate
'auto-test-sjc-service1' not found on OpenStack setup
```

```

2017-05-25 22:06:31,872 - INFO: Received a request to deploy an Host Aggregate
2017-05-25 22:06:32,415 - INFO: Deploying Host Aggregate 'auto-test-sjc-service1' completed
2017-05-25 22:06:32,427 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:06:32,975 - INFO: Getting Host Aggregate failed: Aggregate
'auto-test-sjc-cf-esc-mgmt1' not found on OpenStack setup
2017-05-25 22:06:32,986 - INFO: Received a request to deploy an Host Aggregate
2017-05-25 22:06:33,513 - INFO: Deploying Host Aggregate 'auto-test-sjc-cf-esc-mgmt1'
completed
2017-05-25 22:06:33,524 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:06:33,881 - INFO: Getting Host Aggregate failed: Aggregate
'auto-test-sjc-em-autovnf-mgmt1' not found on OpenStack setup
2017-05-25 22:06:33,891 - INFO: Received a request to deploy an Host Aggregate
2017-05-25 22:06:34,535 - INFO: Deploying Host Aggregate 'auto-test-sjc-em-autovnf-mgmt1'
completed
2017-05-25 22:06:34,580 - INFO: Received a request to deploy AutoVnf
2017-05-25 22:06:40,340 - INFO: Creating AutoVnf deployment (3 instance(s)) on
'http://172.21.201.217:5000/v2.0' tenant 'core' user 'core', ISO '5.1.0-662'
2017-05-25 22:06:40,340 - INFO: Creating network 'auto-testautovnf1-uas-management'
2017-05-25 22:06:42,241 - INFO: Created network 'auto-testautovnf1-uas-management'
2017-05-25 22:06:42,241 - INFO: Creating network 'auto-testautovnf1-uas-orchestration'
2017-05-25 22:06:42,821 - INFO: Created network 'auto-testautovnf1-uas-orchestration'
2017-05-25 22:06:42,888 - INFO: Created flavor 'auto-testautovnf1-uas'
2017-05-25 22:06:42,888 - INFO: Loading image 'auto-testautovnf1-usp-uas-1.0.0-601.qcow2'
from '/opt/cisco/usp/bundles/5.1.0-662/uas-bundle/usp-uas-1.0.0-601.qcow2'
2017-05-25 22:06:53,927 - INFO: Loaded image 'auto-testautovnf1-usp-uas-1.0.0-601.qcow2'
2017-05-25 22:06:53,928 - INFO: Creating volume 'auto-testautovnf1-uas-vol-0' with command

[/opt/cisco/usp/apps/auto-it/vnf/./common/autoit/./autoit_os_utils/scripts/autoit_volume_staging.sh
OS_USERNAME core OS_TENANT_NAME core OS_PASSWORD **** OS_AUTH_URL
http://172.21.201.217:5000/v2.0 ARG_TENANT core ARG_DEPLOYMENT test-uas ARG_VM_NAME
auto-testautovnf1-uas-vol-0 ARG_VOLUME_TYPE LUKS FILE_1 /tmp/tmpHsTAj6/encrypted.cfg]
2017-05-25 22:07:06,104 - INFO: Created volume 'auto-testautovnf1-uas-vol-0'
2017-05-25 22:07:06,104 - INFO: Creating volume 'auto-testautovnf1-uas-vol-1' with command

[/opt/cisco/usp/apps/auto-it/vnf/./common/autoit/./autoit_os_utils/scripts/autoit_volume_staging.sh
OS_USERNAME core OS_TENANT_NAME core OS_PASSWORD **** OS_AUTH_URL
http://172.21.201.217:5000/v2.0 ARG_TENANT core ARG_DEPLOYMENT test-uas ARG_VM_NAME
auto-testautovnf1-uas-vol-1 ARG_VOLUME_TYPE LUKS FILE_1 /tmp/tmpHsTAj6/encrypted.cfg]
2017-05-25 22:07:17,598 - INFO: Created volume 'auto-testautovnf1-uas-vol-1'
2017-05-25 22:07:17,598 - INFO: Creating volume 'auto-testautovnf1-uas-vol-2' with command

[/opt/cisco/usp/apps/auto-it/vnf/./common/autoit/./autoit_os_utils/scripts/autoit_volume_staging.sh
OS_USERNAME core OS_TENANT_NAME core OS_PASSWORD **** OS_AUTH_URL
http://172.21.201.217:5000/v2.0 ARG_TENANT core ARG_DEPLOYMENT test-uas ARG_VM_NAME
auto-testautovnf1-uas-vol-2 ARG_VOLUME_TYPE LUKS FILE_1 /tmp/tmpHsTAj6/encrypted.cfg]
2017-05-25 22:07:29,242 - INFO: Created volume 'auto-testautovnf1-uas-vol-2'
2017-05-25 22:07:30,477 - INFO: Assigned floating IP '172.21.201.59' to IP '172.57.11.101'
2017-05-25 22:07:33,843 - INFO: Creating instance 'auto-testautovnf1-uas-0' and attaching
volume 'auto-testautovnf1-uas-vol-0'
2017-05-25 22:08:00,717 - INFO: Created instance 'auto-testautovnf1-uas-0'
2017-05-25 22:08:00,717 - INFO: Creating instance 'auto-testautovnf1-uas-1' and attaching
volume 'auto-testautovnf1-uas-vol-1'
2017-05-25 22:08:27,577 - INFO: Created instance 'auto-testautovnf1-uas-1'
2017-05-25 22:08:27,578 - INFO: Creating instance 'auto-testautovnf1-uas-2' and attaching
volume 'auto-testautovnf1-uas-vol-2'
2017-05-25 22:08:58,345 - INFO: Created instance 'auto-testautovnf1-uas-2'
2017-05-25 22:08:58,345 - INFO: Deploy request completed
2017-05-25 22:14:07,201 - INFO: Received request to download file system.cfg from config
named vnf-pkg1
2017-05-25 22:19:05,050 - INFO: Received a request to list config folder names.
2017-05-25 22:19:05,051 - INFO: config contents are: vnf-pkg1-images, vnf-pkg1
2017-05-25 22:19:05,059 - INFO: Received a request to list config folder names.
2017-05-25 22:19:05,059 - INFO: config contents are: vnf-pkg1-images, vnf-pkg1
2017-05-25 22:19:05,066 - INFO: Received a request to create a configuration folder.

```

```

2017-05-25 22:19:05,073 - INFO: Received a request to create a configuration folder.
2017-05-25 22:19:05,076 - INFO: Received request to download package: system.cfg from ISO
2017-05-25 22:19:05,083 - INFO: Received request to download package: system.cfg from ISO
2017-05-25 22:19:05,085 - INFO: Received request to download package: system.cfg from ISO
2017-05-25 22:19:05,086 - INFO: Received request to upload config file system.cfg to config
named vnf-pkg2
2017-05-25 22:19:05,087 - INFO: Uploaded file system.cfg to config named vnf-pkg2
2017-05-25 22:19:59,895 - INFO: Received request to upload ISO usp-5_1_0.iso
2017-05-25 22:19:59,895 - INFO: Saving ISO to /tmp/tmpWbdnxm/usp-5_1_0.iso
2017-05-25 22:20:21,395 - INFO: Mounting ISO to /tmp/tmpWbdnxm/iso_mount
2017-05-25 22:20:22,288 - INFO: ISO version already installed, (5.1.0-662)
2017-05-25 22:20:23,203 - INFO: Received a request to list file names in config named
vnf-pkg2.
2017-05-25 22:20:23,203 - INFO: config contents are: system.cfg
2017-05-25 22:20:23,211 - INFO: Received a request to list file names in config named
vnf-pkg2-images.
2017-05-25 22:20:23,211 - INFO: config contents are:
2017-05-25 22:20:23,220 - INFO: Received request to get ISO details 5.1.0-662
2017-05-25 22:20:23,251 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:20:23,621 - INFO: Getting Host Aggregate failed: Aggregate
'auto-test-sjc-em-autovnf-mgmt2' not found on OpenStack setup
2017-05-25 22:20:23,633 - INFO: Received a request to deploy an Host Aggregate
2017-05-25 22:20:24,301 - INFO: Deploying Host Aggregate 'auto-test-sjc-em-autovnf-mgmt2'
completed
2017-05-25 22:20:24,313 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:20:24,843 - INFO: Getting Host Aggregate failed: Aggregate
'auto-test-sjc-service2' not found on OpenStack setup
2017-05-25 22:20:24,853 - INFO: Received a request to deploy an Host Aggregate
2017-05-25 22:20:25,524 - INFO: Deploying Host Aggregate 'auto-test-sjc-service2' completed
2017-05-25 22:20:25,537 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:20:25,898 - INFO: Getting Host Aggregate failed: Aggregate
'auto-test-sjc-cf-esc-mgmt2' not found on OpenStack setup
2017-05-25 22:20:25,909 - INFO: Received a request to deploy an Host Aggregate
2017-05-25 22:20:26,540 - INFO: Deploying Host Aggregate 'auto-test-sjc-cf-esc-mgmt2'
completed
2017-05-25 22:20:26,584 - INFO: Received a request to deploy AutoVnf
2017-05-25 22:20:31,604 - INFO: Creating AutoVnf deployment (3 instance(s)) on
'http://172.21.201.217:5000/v2.0' tenant 'core' user 'core', ISO '5.1.0-662'
2017-05-25 22:20:31,605 - INFO: Creating network 'auto-testautovnf2-uas-management'
2017-05-25 22:20:33,720 - INFO: Created network 'auto-testautovnf2-uas-management'
2017-05-25 22:20:33,720 - INFO: Creating network 'auto-testautovnf2-uas-orchestration'
2017-05-25 22:20:34,324 - INFO: Created network 'auto-testautovnf2-uas-orchestration'
2017-05-25 22:20:34,402 - INFO: Created flavor 'auto-testautovnf2-uas'
2017-05-25 22:20:34,402 - INFO: Loading image 'auto-testautovnf2-usp-uas-1.0.0-601.qcow2'
from '/opt/cisco/usp/bundles/5.1.0-662/uas-bundle/usp-uas-1.0.0-601.qcow2'
2017-05-25 22:20:43,169 - INFO: Loaded image 'auto-testautovnf2-usp-uas-1.0.0-601.qcow2'
2017-05-25 22:20:43,169 - INFO: Creating volume 'auto-testautovnf2-uas-vol-0' with command

[/opt/cisco/usp/apps/auto-it/vnf/./common/autoit/./autoit_os_utils/scripts/autoit_volume_staging.sh
OS_USERNAME core OS_TENANT_NAME core OS_PASSWORD **** OS_AUTH_URL
http://172.21.201.217:5000/v2.0 ARG_TENANT core ARG_DEPLOYMENT test-uas ARG_VM_NAME
auto-testautovnf2-uas-vol-0 ARG_VOLUME_TYPE LUKS FILE_1 /tmp/tmpelmMIL/encrypted.cfg]
2017-05-25 22:20:54,713 - INFO: Created volume 'auto-testautovnf2-uas-vol-0'
2017-05-25 22:20:54,714 - INFO: Creating volume 'auto-testautovnf2-uas-vol-1' with command

[/opt/cisco/usp/apps/auto-it/vnf/./common/autoit/./autoit_os_utils/scripts/autoit_volume_staging.sh
OS_USERNAME core OS_TENANT_NAME core OS_PASSWORD **** OS_AUTH_URL
http://172.21.201.217:5000/v2.0 ARG_TENANT core ARG_DEPLOYMENT test-uas ARG_VM_NAME
auto-testautovnf2-uas-vol-1 ARG_VOLUME_TYPE LUKS FILE_1 /tmp/tmpelmMIL/encrypted.cfg]
2017-05-25 22:21:06,203 - INFO: Created volume 'auto-testautovnf2-uas-vol-1'
2017-05-25 22:21:06,204 - INFO: Creating volume 'auto-testautovnf2-uas-vol-2' with command

[/opt/cisco/usp/apps/auto-it/vnf/./common/autoit/./autoit_os_utils/scripts/autoit_volume_staging.sh
OS_USERNAME core OS_TENANT_NAME core OS_PASSWORD **** OS_AUTH_URL

```



```

http://172.21.201.217:5000/v2.0 ARG_TENANT core ARG_DEPLOYMENT test-uas ARG_VM_NAME
auto-testautovnf2-uas-vol-2 ARG_VOLUME_TYPE LUKS FILE_1 /tmp/tmpelmMIL/encrypted.cfg]
2017-05-25 22:21:18,184 - INFO: Created volume 'auto-testautovnf2-uas-vol-2'
2017-05-25 22:21:19,626 - INFO: Assigned floating IP '172.21.201.64' to IP '172.67.11.101'
2017-05-25 22:21:22,762 - INFO: Creating instance 'auto-testautovnf2-uas-0' and attaching
volume 'auto-testautovnf2-uas-vol-0'
2017-05-25 22:21:49,741 - INFO: Created instance 'auto-testautovnf2-uas-0'
2017-05-25 22:21:49,742 - INFO: Creating instance 'auto-testautovnf2-uas-1' and attaching
volume 'auto-testautovnf2-uas-vol-1'
2017-05-25 22:22:16,881 - INFO: Created instance 'auto-testautovnf2-uas-1'
2017-05-25 22:22:16,881 - INFO: Creating instance 'auto-testautovnf2-uas-2' and attaching
volume 'auto-testautovnf2-uas-vol-2'
2017-05-25 22:22:43,304 - INFO: Created instance 'auto-testautovnf2-uas-2'
2017-05-25 22:22:43,304 - INFO: Deploy request completed
2017-05-25 22:28:08,865 - INFO: Received request to download file system.cfg from config
named vnf-pkg2
2017-05-25 22:40:03,550 - INFO: Received request to download file system.cfg from config
named vnf-pkg1

```

### Example Termination Log:

```

2017-05-25 22:53:30,970 - INFO: Received a request to destroy AutoVnf
2017-05-25 22:53:31,310 - INFO: Destroying AutoVnf deployment on
'http://172.21.201.217:5000/v2.0' tenant 'core' user 'core', ISO '5.1.0-662'
2017-05-25 22:53:32,698 - INFO: Removed floating IP '172.21.201.64'
2017-05-25 22:53:34,114 - INFO: 3 instance(s) found with name matching 'auto-testautovnf2'
2017-05-25 22:53:34,448 - INFO: Removing volume 'auto-testautovnf2-uas-vol-2'
2017-05-25 22:53:43,481 - INFO: Removed volume 'auto-testautovnf2-uas-vol-2'
2017-05-25 22:53:43,481 - INFO: Removing instance 'auto-testautovnf2-uas-2'
2017-05-25 22:53:47,080 - INFO: Removed instance 'auto-testautovnf2-uas-2'
2017-05-25 22:53:47,283 - INFO: Removing volume 'auto-testautovnf2-uas-vol-1'
2017-05-25 22:53:56,508 - INFO: Removed volume 'auto-testautovnf2-uas-vol-1'
2017-05-25 22:53:56,508 - INFO: Removing instance 'auto-testautovnf2-uas-1'
2017-05-25 22:54:00,290 - INFO: Removed instance 'auto-testautovnf2-uas-1'
2017-05-25 22:54:00,494 - INFO: Removing volume 'auto-testautovnf2-uas-vol-0'
2017-05-25 22:54:04,714 - INFO: Removed volume 'auto-testautovnf2-uas-vol-0'
2017-05-25 22:54:04,714 - INFO: Removing instance 'auto-testautovnf2-uas-0'
2017-05-25 22:54:11,647 - INFO: Removed instance 'auto-testautovnf2-uas-0'
2017-05-25 22:54:15,107 - INFO: 1 image(s) 'auto-testautovnf2-usp-uas-1.0.0-601.qcow2'
found, removing
2017-05-25 22:54:19,289 - INFO: Removed network 'auto-testautovnf2-uas-management'
2017-05-25 22:54:20,463 - INFO: Removed network 'auto-testautovnf2-uas-orchestration'
2017-05-25 22:54:20,541 - INFO: Removed flavor 'auto-testautovnf2-uas'
2017-05-25 22:54:20,541 - INFO: Destroy request completed
2017-05-25 22:54:20,562 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:54:20,925 - INFO: Getting Host Aggregate 'auto-test-sjc-em-autovnf-mgmt2'
completed
2017-05-25 22:54:20,940 - INFO: Received a request to destroy an Host Aggregate
2017-05-25 22:54:21,564 - INFO: Destroying Host Aggregate 'auto-test-sjc-em-autovnf-mgmt2'
completed
2017-05-25 22:54:21,575 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:54:21,930 - INFO: Getting Host Aggregate 'auto-test-sjc-service2' completed
2017-05-25 22:54:21,947 - INFO: Received a request to destroy an Host Aggregate
2017-05-25 22:54:22,456 - INFO: Destroying Host Aggregate 'auto-test-sjc-service2' completed
2017-05-25 22:54:22,468 - INFO: Received a request to get an Host Aggregate details
2017-05-25 22:54:22,826 - INFO: Getting Host Aggregate 'auto-test-sjc-cf-esc-mgmt2' completed
2017-05-25 22:54:22,840 - INFO: Received a request to destroy an Host Aggregate
2017-05-25 22:54:23,394 - INFO: Destroying Host Aggregate 'auto-test-sjc-cf-esc-mgmt2'
completed
2017-05-25 22:56:55,925 - INFO: Received a request to destroy AutoVnf
2017-05-25 22:56:56,391 - INFO: Destroying AutoVnf deployment on
'http://172.21.201.217:5000/v2.0' tenant 'core' user 'core', ISO '5.1.0-662'
2017-05-25 22:56:57,507 - INFO: Removed floating IP '172.21.201.59'
2017-05-25 22:56:58,614 - INFO: 3 instance(s) found with name matching 'auto-testautovnf1'
2017-05-25 22:56:58,949 - INFO: Removing volume 'auto-testautovnf1-uas-vol-2'

```

```

2017-05-25 22:57:08,166 - INFO: Removed volume 'auto-testautovnf1-uas-vol-2'
2017-05-25 22:57:08,166 - INFO: Removing instance 'auto-testautovnf1-uas-2'
2017-05-25 22:57:15,117 - INFO: Removed instance 'auto-testautovnf1-uas-2'
2017-05-25 22:57:15,323 - INFO: Removing volume 'auto-testautovnf1-uas-vol-1'
2017-05-25 22:57:24,501 - INFO: Removed volume 'auto-testautovnf1-uas-vol-1'
2017-05-25 22:57:24,502 - INFO: Removing instance 'auto-testautovnf1-uas-1'
2017-05-25 22:57:28,275 - INFO: Removed instance 'auto-testautovnf1-uas-1'
2017-05-25 22:57:28,722 - INFO: Removing volume 'auto-testautovnf1-uas-vol-0'
2017-05-25 22:57:37,702 - INFO: Removed volume 'auto-testautovnf1-uas-vol-0'
2017-05-25 22:57:37,703 - INFO: Removing instance 'auto-testautovnf1-uas-0'
2017-05-25 22:57:44,622 - INFO: Removed instance 'auto-testautovnf1-uas-0'
2017-05-25 22:57:47,921 - INFO: 1 image(s) 'auto-testautovnf1-usp-uas-1.0.0-601.qcow2'
found, removing
2017-05-25 22:57:52,453 - INFO: Removed network 'auto-testautovnf1-uas-management'
2017-05-25 22:57:53,677 - INFO: Removed network 'auto-testautovnf1-uas-orchestration'
2017-05-25 22:57:53,760 - INFO: Removed flavor 'auto-testautovnf1-uas'
2017-05-25 22:57:53,760 - INFO: Destroy request completed

```

## Viewing AutoIT Operational Data

View the AutoIT operational data by executing the following command:

```
show uas
```

### Example show uas Command Output

```

uas version          6.0.0
uas state            active
uas external-connection-point 172.28.185.132
INSTANCE IP        STATE   ROLE
-----
172.28.185.133    alive   CONFD-MASTER
172.28.185.134    alive   CONFD-SLAVE

NAME                LAST HEARTBEAT
-----
AutoIT-MASTER      2018-01-24 21:24:30
USPCFMWorker        2018-01-24 21:24:30
USPCHBWorker        2018-01-24 21:24:30
USPCWorker          2018-01-24 21:24:30

```



### Important

In case of standalone mode (non-HA) deployments, the *uas external-connection-point* information and *Instance IP* table are not applicable and are not displayed.

## Checking AutoIT Processes

Verify that key processes are running on the AutoIT VM:

With Ubuntu 14.04:

```
service autoit status
```

### Example output:

```
AutoIT is running.
```

Check ConfD.

```
service uas-confd status
```

With Ubuntu 16.04:

```
* autoit.service - Job that runs the autoit daemon
Loaded: loaded (/etc/systemd/system/autoit.service; static; vendor preset: enabled)
Active: active (running) since Fri 2018-09-21 22:11:54 UTC; 1 weeks 0 days ago
Main PID: 1320 (autoit.sh)
CGroup: /system.slice/autoit.service
|-1320 /bin/sh /etc/cisco/autoit.sh start
|-1337 /bin/sh /etc/cisco/autoit.sh start
|-1338 /opt/cisco/usp/uas/autoit/autoit
|-1339 tee -a /var/log/upstart/autoit.log
|-1341 /opt/cisco/usp/uas/autoit/autoit
|-1346 /opt/cisco/usp/uas/autoit/autoit
|-1347 /opt/cisco/usp/uas/autoit/autoit
|-1348 /opt/cisco/usp/uas/autoit/autoit
|-1349 /opt/cisco/usp/uas/autoit/autoit
|-1350 /opt/cisco/usp/uas/autoit/autoit
|-1352 /opt/cisco/usp/uas/autoit/autoit
|-1353 /opt/cisco/usp/uas/autoit/autoit
```

## Monitoring AutoVNF Operations

This section identifies various commands that can be used to determine the status and health of AutoVNF.

To use them, you must:

1. Log on to the AutoVNF VM as *ubuntu*. Use the password that was created earlier for this user.
2. Become the root user.

```
sudo -i
```

## Viewing AutoVNF Logs

### General AutoVNF Logs

AutoVNF logs are available on the AutoVNF VM in the following file:

```
/var/log/upstart/autovnf.log
```

To collect AutoVNF logs:

1. Navigate to the *scripts* directory.

```
cd /opt/cisco/usp/uas/scripts
```

2. Launch the *collect-uas-logs.sh* script to collect the logs.

```
sudo ./collect-uas-logs.sh
```

#### Example log output:

```
Creating log tarball uas-logs-2017-05-26_00.24.55_UTC.tar.bz2 ...
uas-logs/
uas-logs/autovnf/
uas-logs/autovnf/autovnf_server.log
uas-logs/autovnf/a15bf26c-41a1-11e7-b3ab-fa163eccaffc/
uas-logs/autovnf/a15bf26c-41a1-11e7-b3ab-fa163eccaffc/netconf_traces
uas-logs/autovnf/a15bf26c-41a1-11e7-b3ab-fa163eccaffc/vnfd
```

```

uas-logs/autovnf/audit.log
uas-logs/autovnf/579b4546-41a2-11e7-b3ab-fa163eccaffc/
uas-logs/autovnf/579b4546-41a2-11e7-b3ab-fa163eccaffc/netconf_traces
uas-logs/autovnf/579b4546-41a2-11e7-b3ab-fa163eccaffc/vnfd
uas-logs/ha/
uas-logs/ha/info.log
uas-logs/uas_manager/
uas-logs/uas_manager/info.log
uas-logs/zk/
uas-logs/zk/zookeeper.out
uas-logs/zk/zookeeper.log
uas-logs/upstart/
uas-logs/upstart/uas-confd.log
uas-logs/upstart/zk.log
uas-logs/upstart/autovnf.log
uas-logs/upstart/uws-ae.log
uas-logs/upstart/ensemble.log

===== Tarball available at: /tmp/uas-logs-2017-05-26_00.24.55.UTC.tar.bz2
=====

To extract the tarball, run: "tar jxf /tmp/uas-logs-2017-05-26_00.24.55.UTC.tar.bz2"

```

## AutoVNF Transaction Logs

AutoVNF server and transaction logs are available on the Ultra M Manager Node in the following directory on the UAS VM:

```
/var/log/cisco-uas/autovnf
```

Inside this directory are transaction sub-directories, VNFD information and NETCONF traces are provided for the given transaction.

### Example:

```

total 3568
drwxr-xr-x 4 root root    4096 May 25 23:31 ./
drwxr-xr-x 7 root root    4096 May 25 19:39 ../
drwxr-xr-x 2 root root    4096 May 25 23:31 579b4546-41a2-11e7-b3ab-fa163eccaffc/
drwxr-xr-x 2 root root    4096 May 25 23:29 a15bf26c-41a1-11e7-b3ab-fa163eccaffc/
-rw-r--r-- 1 root root 3632813 May 26 18:33 audit.log
-rw-r--r-- 1 root root      0 May 25 23:26 autovnf_server.log

cd a15bf26c-41a1-11e7-b3ab-fa163eccaffc
total 2568
drwxr-xr-x 2 root root    4096 May 25 23:29 ./
drwxr-xr-x 4 root root    4096 May 25 23:31 ../
-rw-r--r-- 1 root root 2614547 May 25 23:37 netconf_traces
-rw-r--r-- 1 root root      0 May 25 23:29 vnfd

```

## AutoVNF Event Logs

Event logs provide useful information on UAS task progress. These logs are located in the *autovnf.log* file within the following directory on the UAS VM:

```
/var/log/upstart
```

Event logs are filed by transaction ID. To view transaction IDs:

1. Log in to the ConfD CLI as the *admin* user.

```
confd_cli -u admin -C
```

2. Enter the *admin* user password when prompted.
3. List the transactions.

### show transactions

#### Example output:

TX ID	STATUS	TX TYPE	DEPLOYMENT ID	TIMESTAMP
562c18b0-4199-11e7-ad05-fa163ec6a7e4		vnf-deployment	vnfd2-deployment	
2017-05-25T22:27:28.962293-00:00		deployment-success		
abf51428-4198-11e7-ad05-fa163ec6a7e4		vnfm-deployment	ab-auto-test-vnfm2	
2017-05-25T22:22:43.389059-00:00		deployment-success		

To view the logs associated with a specific transaction:

### show log <transaction\_id> | display xml

#### Example log pertaining to VNFM deployment:

```
<config xmlns="http://tail-f.com/ns/config/1.0">
  <logs xmlns="http://www.cisco.com/usp/nfv/usp-autovnf-oper">
    <tx-id>abf51428-4198-11e7-ad05-fa163ec6a7e4</tx-id>
    <log>2017-05-25 22:22:43,402 - VNFM Deployment RPC triggered for deployment:
ab-auto-test-vnfm2, deactivate: 0
2017-05-25 22:22:43,446 - Notify deployment
2017-05-25 22:22:43,472 - VNFM Transaction: abf51428-4198-11e7-ad05-fa163ec6a7e4 for
deployment: ab-auto-test-vnfm2 started
2017-05-25 22:22:43,497 - Downloading Image:
http://172.21.201.63:80/bundles/5.1.0-662/vnfm-bundle/ESC-2_3_2_143.qcow2
2017-05-25 22:22:49,146 - Image: //opt/cisco/vnf-staging/vnfm_image downloaded
successfully
2017-05-25 22:22:49,714 - Checking network 'public' existence
2017-05-25 22:22:49,879 - Checking flavor 'ab-auto-test-vnfm2-ESC-flavor' non existence
2017-05-25 22:22:50,124 - Checking image 'ab-auto-test-vnfm2-ESC-image' non existence
2017-05-25 22:22:50,598 - Checking network 'auto-testautovnf2-uas-management' existence
2017-05-25 22:22:50,752 - Checking network 'auto-testautovnf2-uas-orchestration' existence
2017-05-25 22:22:50,916 - Checking instance 'ab-auto-test-vnfm2-ESC-0' non existence
2017-05-25 22:22:51,357 - Checking instance 'ab-auto-test-vnfm2-ESC-1' non existence
2017-05-25 22:22:52,084 - Creating flavor 'ab-auto-test-vnfm2-ESC-flavor'
2017-05-25 22:22:52,184 - Loading image 'ab-auto-test-vnfm2-ESC-image' from
'//opt/cisco/vnf-staging/vnfm_image'...
2017-05-25 22:23:06,444 - ESC HA mode is ON
2017-05-25 22:23:07,118 - Allocated these IPs for ESC HA: ['172.67.11.3', '172.67.11.4',
'172.67.11.5']
2017-05-25 22:23:08,228 - Creating VNFM 'ab-auto-test-vnfm2-ESC-0' with [python
//opt/cisco/vnf-staging/bootvm.py ab-auto-test-vnfm2-ESC-0 --flavor
ab-auto-test-vnfm2-ESC-flavor --image b29e7a72-9ad0-4178-aa35-35df0a2b23b7 --net
auto-testautovnf2-uas-management --gateway_ip 172.67.11.1 --net
auto-testautovnf2-uas-orchestration
--os_auth_url http://172.21.201.217:5000/v2.0 --os_tenant_name core --os_username *****
--os_password ***** --bs_os_auth_url http://172.21.201.217:5000/v2.0 --bs_os_tenant_name
core --bs_os_username ***** --bs_os_password ***** --esc_ui_startup false
--esc_params_file /tmp/esc_params.cfg --encrypt_key ***** --user_pass *****
--user_confid_pass ***** --kad_vif eth0 --kad_vip 172.67.11.5 --ipaddr 172.67.11.3 dhcp
--ha_node_list 172.67.11.3 172.67.11.4 --file
root:0755:/opt/cisco/esc/esc-scripts/esc_volume_em_staging.sh:
/opt/cisco/usp/uas/autovnf/vnfms/esc-scripts/esc_volume_em_staging.sh
--file
root:0755:/opt/cisco/esc/esc-scripts/esc_vpc_chassis_id.py:/opt/cisco/usp/uas/autovnf/vnfms/esc-scripts/esc_vpc_chassis_id.py
--file
root:0755:/opt/cisco/esc/esc-scripts/esc_vpc-di-internal-keys.sh:/opt/cisco/usp/uas/autovnf/vnfms/esc-scripts/esc_vpc-di-internal-keys.sh)...
2017-05-25 22:24:13,329 - ESC started!
```

```

2017-05-25 22:24:13,803 - Creating VNF 'ab-auto-test-vnfm2-ESC-1' with [python
//opt/cisco/vnf-staging/bootvm.py ab-auto-test-vnfm2-ESC-1 --flavor
ab-auto-test-vnfm2-ESC-flavor --image b29e7a72-9ad0-4178-aa35-35df0a2b23b7 --net
auto-testautovnf2-uas-management --gateway_ip 172.67.11.1 --net
auto-testautovnf2-uas-orchestration
--os_auth_url http://172.21.201.217:5000/v2.0 --os_tenant_name core --os_username *****
--os_password ***** --bs_os_auth_url http://172.21.201.217:5000/v2.0 --bs_os_tenant_name
core --bs_os_username ***** --bs_os_password ***** --esc_ui_startup false
--esc_params_file /tmp/esc_params.cfg --encrypt_key ***** --user_pass *****
--user_confd_pass ***** --kad_vif eth0 --kad_vip 172.67.11.5 --ipaddr 172.67.11.4 dhcp
--ha_node_list 172.67.11.3 172.67.11.4
--file root:0755:/opt/cisco/esc/esc-scripts/esc_volume_em_staging.sh:
/opt/cisco/usp/uas/autovnf/vnfms/esc-scripts/esc_volume_em_staging.sh --file
root:0755:/opt/cisco/esc/esc-scripts/esc_vpc_chassis_id.py:/opt/cisco/usp/uas/autovnf/vnfms/esc-scripts/esc_vpc_chassis_id.py
--file
root:0755:/opt/cisco/esc/esc-scripts/esc_vpc-di-internal-keys.sh:/opt/cisco/usp/uas/autovnf/vnfms/esc-scripts/esc_vpc-di-internal-keys.sh]...
2017-05-25 22:25:12,660 - ESC started!
2017-05-25 22:25:12,677 - Waiting for VIM to declare 2 instance(s) active
2017-05-25 22:25:18,254 - Instance(s) are active
2017-05-25 22:25:18,271 - Waiting for VNF to be ready...
2017-05-25 22:25:18,292 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:25:21,313 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:25:31,341 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:25:31,362 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:25:41,379 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:25:41,397 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:25:51,424 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:25:51,495 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:26:01,521 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:26:01,539 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:26:11,563 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:26:11,591 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:26:21,617 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:26:21,635 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:26:31,662 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:26:31,680 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:26:41,706 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:26:41,726 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:26:51,748 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:26:51,765 - Could not establish NETCONF session to 172.67.11.5
2017-05-25 22:27:01,791 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:27:02,204 - NETCONF Sessions (Transaction/Notifications) established
2017-05-25 22:27:02,507 - Notify VNF Up
2017-05-25 22:27:02,525 - VNF Transaction: abf51428-4198-11e7-ad05-fa163ec6a7e4 for
deployment: ab-auto-test-vnfm2 completed successfully.
2017-05-25 22:27:02,545 - Notify deployment</log>
</logs>
</config>

```

### Example log pertaining to VNF deployment:

```

<config xmlns="http://tail-f.com/ns/config/1.0">
  <logs xmlns="http://www.cisco.com/usp/nfv/usp-autovnf-oper">
    <tx-id>562c18b0-4199-11e7-ad05-fa163ec6a7e4</tx-id>
    <log>2017-05-25 22:27:29,039 - Notify deployment
2017-05-25 22:27:29,062 - Connection to VNF (esc) at 172.67.11.5
2017-05-25 22:27:29,404 - NETCONF Sessions (Transaction/Notifications) established
2017-05-25 22:27:29,420 - Get Images
2017-05-25 22:27:29,435 - NETCONF get-config Request sent, waiting for reply
2017-05-25 22:27:29,560 - NETCONF Transaction success!
2017-05-25 22:27:29,570 - Get Flavors List
2017-05-25 22:27:29,582 - Adding images ...
2017-05-25 22:27:29,592 - Creating Images
2017-05-25 22:27:29,603 - image: ab-auto-test-vnfm2-element-manager
2017-05-25 22:27:29,620 - src:

```

```
http://172.21.201.63:80/bundles/5.1.0-662/em-bundle/em-1_0_0_532.qcow2
2017-05-25 22:27:29,630 - disk_format: qcow2
2017-05-25 22:27:29,641 - container_format: bare
2017-05-25 22:27:29,655 - serial_console: True
2017-05-25 22:27:29,665 - disk_bus: virtio
2017-05-25 22:27:29,674 - NETCONF edit-config Request sent, waiting for reply
2017-05-25 22:27:29,901 - NETCONF Transaction success!
2017-05-25 22:27:29,911 - Waiting for VNFM to process CREATE_IMAGE transaction
2017-05-25 22:27:46,987 - | CREATE_IMAGE | ab-auto-test-vnfm2-element-manager | SUCCESS
| (1/1)
2017-05-25 22:27:47,004 - NETCONF transaction completed successfully!
2017-05-25 22:27:47,749 - Creating Images
2017-05-25 22:27:47,764 - image: ab-auto-test-vnfm2-control-function
2017-05-25 22:27:47,776 - src:
http://172.21.201.63:80/bundles/5.1.0-662/ugp-bundle/qvpc-di-cf.qcow2
2017-05-25 22:27:47,793 - disk_format: qcow2
2017-05-25 22:27:47,805 - container_format: bare
2017-05-25 22:27:47,819 - serial_console: True
2017-05-25 22:27:47,831 - disk_bus: virtio
2017-05-25 22:27:47,841 - NETCONF edit-config Request sent, waiting for reply
2017-05-25 22:27:48,317 - NETCONF Transaction success!
2017-05-25 22:27:48,331 - Waiting for VNFM to process CREATE_IMAGE transaction
2017-05-25 22:27:56,403 - | CREATE_IMAGE | ab-auto-test-vnfm2-control-function | SUCCESS
| (1/1)
2017-05-25 22:27:56,434 - NETCONF transaction completed successfully!
2017-05-25 22:27:56,822 - Creating Images
2017-05-25 22:27:56,838 - image: ab-auto-test-vnfm2-session-function
2017-05-25 22:27:57,267 - src:
http://172.21.201.63:80/bundles/5.1.0-662/ugp-bundle/qvpc-di-sf.qcow2
2017-05-25 22:27:57,412 - disk_format: qcow2
2017-05-25 22:27:57,423 - container_format: bare
2017-05-25 22:27:57,523 - serial_console: True
2017-05-25 22:27:57,535 - disk_bus: virtio
2017-05-25 22:27:57,550 - NETCONF edit-config Request sent, waiting for reply
2017-05-25 22:27:58,378 - NETCONF Transaction success!
2017-05-25 22:27:58,391 - Waiting for VNFM to process CREATE_IMAGE transaction
2017-05-25 22:28:06,339 - | CREATE_IMAGE | ab-auto-test-vnfm2-session-function | SUCCESS
| (1/1)
2017-05-25 22:28:06,355 - NETCONF transaction completed successfully!
2017-05-25 22:28:06,367 - Images added successfully
2017-05-25 22:28:06,378 - Creating flavors ...
2017-05-25 22:28:06,388 - Creating flavors
2017-05-25 22:28:06,432 - flavor: ab-auto-test-vnfm2-element-manager
2017-05-25 22:28:06,444 - vcpus: 2
2017-05-25 22:28:06,457 - memory_mb: 4096
2017-05-25 22:28:06,469 - root_disk_mb: 40960
2017-05-25 22:28:06,481 - ephemeral_disk_mb: 0
2017-05-25 22:28:06,491 - swap_disk_mb: 0
2017-05-25 22:28:06,505 - NETCONF edit-config Request sent, waiting for reply
2017-05-25 22:28:06,781 - NETCONF Transaction success!
2017-05-25 22:28:06,793 - Waiting for VNFM to process CREATE_FLAVOR transaction
2017-05-25 22:28:07,286 - | CREATE_FLAVOR | ab-auto-test-vnfm2-element-manager | SUCCESS
| (1/1)
2017-05-25 22:28:07,298 - NETCONF transaction completed successfully!
2017-05-25 22:28:07,310 - Creating flavors
2017-05-25 22:28:07,328 - flavor: ab-auto-test-vnfm2-control-function
2017-05-25 22:28:07,341 - vcpus: 8
2017-05-25 22:28:07,358 - memory_mb: 16384
2017-05-25 22:28:07,374 - root_disk_mb: 6144
2017-05-25 22:28:07,386 - ephemeral_disk_mb: 0
2017-05-25 22:28:07,398 - swap_disk_mb: 0
2017-05-25 22:28:07,410 - NETCONF edit-config Request sent, waiting for reply
2017-05-25 22:28:07,586 - NETCONF Transaction success!
2017-05-25 22:28:07,603 - Waiting for VNFM to process CREATE_FLAVOR transaction
```

```

2017-05-25 22:28:07,818 - | CREATE_FLAVOR | ab-auto-test-vnfm2-control-function | SUCCESS
| (1/1)
2017-05-25 22:28:07,830 - NETCONF transaction completed successfully!
2017-05-25 22:28:07,842 - Creating flavors
2017-05-25 22:28:07,853 -   flavor: ab-auto-test-vnfm2-session-function
2017-05-25 22:28:07,865 -   vcpus: 8
2017-05-25 22:28:07,877 -   memory_mb: 16384
2017-05-25 22:28:07,889 -   root_disk_mb: 6144
2017-05-25 22:28:07,901 -   ephemeral_disk_mb: 0
2017-05-25 22:28:07,917 -   swap_disk_mb: 0
2017-05-25 22:28:07,928 - NETCONF edit-config Request sent, waiting for reply
2017-05-25 22:28:08,204 - NETCONF Transaction success!
2017-05-25 22:28:08,216 - Waiting for VNFM to process CREATE_FLAVOR transaction
2017-05-25 22:28:08,455 - | CREATE_FLAVOR | ab-auto-test-vnfm2-session-function | SUCCESS
| (1/1)
2017-05-25 22:28:08,473 - NETCONF transaction completed successfully!
2017-05-25 22:28:08,489 - Flavors created successfully
2017-05-25 22:28:08,501 - Onboarding configuration file: ('control-function',
'staros_config.txt', 'http://172.21.201.63:5001/configs/vnf-pkg2/files/system.cfg')
2017-05-25 22:28:08,547 - NETCONF get-operational Request sent, waiting for reply
2017-05-25 22:28:08,724 - NETCONF Transaction success!
2017-05-25 22:28:08,855 - Notify VDU Create Catalog for : element-manager, status:
SUCCESS, txid: 562c18b0-4199-11e7-ad05-fa163ec6a7e4
2017-05-25 22:28:08,892 - Notify VDU Create Catalog for : control-function, status:
SUCCESS, txid: 562c18b0-4199-11e7-ad05-fa163ec6a7e4
2017-05-25 22:28:09,008 - Notify VDU Create Catalog for : session-function, status:
SUCCESS, txid: 562c18b0-4199-11e7-ad05-fa163ec6a7e4
2017-05-25 22:28:09,024 - NETCONF get-config Request sent, waiting for reply
2017-05-25 22:28:09,151 - NETCONF Transaction success!
2017-05-25 22:28:14,837 - Deployment: vnfd2-deployment started ...
2017-05-25 22:28:14,858 - Generating VNFD
2017-05-25 22:28:14,930 - VNFD generated successfully.
2017-05-25 22:28:14,966 - Generating configuration files for EM
2017-05-25 22:28:14,979 - Creating VIP Ports
2017-05-25 22:28:16,970 - VIP ports created successfully
2017-05-25 22:28:16,987 - Deploying EM
2017-05-25 22:28:17,000 - Added anti-affinity placement policy for ab-auto-test-vnfm2-em-1
2017-05-25 22:28:17,012 - Added anti-affinity placement policy for ab-auto-test-vnfm2-em-2
2017-05-25 22:28:17,025 - Added anti-affinity placement policy for ab-auto-test-vnfm2-em-3
2017-05-25 22:28:17,041 - Starting Service Deployment: ab-auto-test-vnfm2-em
2017-05-25 22:28:17,054 - Start VM: ab-auto-test-vnfm2-em-1
2017-05-25 22:28:17,066 - Start VM: ab-auto-test-vnfm2-em-2
2017-05-25 22:28:17,077 - Start VM: ab-auto-test-vnfm2-em-3
2017-05-25 22:28:17,089 - NETCONF edit-config Request sent, waiting for reply
2017-05-25 22:28:17,721 - NETCONF Transaction success!
2017-05-25 22:28:17,733 - Waiting for VNFM to process SERVICE_ALIVE transaction
2017-05-25 22:29:37,185 - | VM_DEPLOYED | ab-auto-test-vnfm2-em-1 | SUCCESS | Waiting
for: SERVICE_ALIVE|
2017-05-25 22:29:59,679 - | VM_ALIVE | ab-auto-test-vnfm2-em-1 | SUCCESS | Waiting for:
SERVICE_ALIVE|
2017-05-25 22:30:42,170 - | VM_DEPLOYED | ab-auto-test-vnfm2-em-2 | SUCCESS | Waiting
for: SERVICE_ALIVE|
2017-05-25 22:30:59,620 - | VM_ALIVE | ab-auto-test-vnfm2-em-2 | SUCCESS | Waiting for:
SERVICE_ALIVE|
2017-05-25 22:31:51,510 - | VM_DEPLOYED | ab-auto-test-vnfm2-em-3 | SUCCESS | Waiting
for: SERVICE_ALIVE|
2017-05-25 22:32:13,584 - | VM_DEPLOYED | c2 | SUCCESS | Waiting for: SERVICE_ALIVE|
2017-05-25 22:32:29,639 - | VM_ALIVE | ab-auto-test-vnfm2-em-3 | SUCCESS | Waiting for:
SERVICE_ALIVE|
2017-05-25 22:32:29,661 - | SERVICE_ALIVE | ab-auto-test-vnfm2-em | SUCCESS | (1/1)
2017-05-25 22:32:29,674 - NETCONF transaction completed successfully!
2017-05-25 22:32:29,687 - EM Online !
2017-05-25 22:32:29,699 - HA-VIP[element-manager] : 172.67.11.12
2017-05-25 22:32:29,716 - HA-VIP[control-function] : 172.67.11.13

```



```

2017-05-25 22:32:29,729 - Deployment: vnf2-deployment completed successfully.
2017-05-25 22:32:29,742 - NETCONF get-operational Request sent, waiting for reply
2017-05-25 22:32:30,221 - NETCONF Transaction success!
2017-05-25 22:32:30,261 - Notify EM Up
2017-05-25 22:32:30,274 - VNF Transaction completed successfully!
2017-05-25 22:32:30,292 - Notify deployment</log>
</logs>
</config>

```

## Viewing AutoVNF Operational Data

AutoVNF maintains history information for all transactions, associated events, and related error/information logs in persistent storage. These logs are useful for monitoring deployment progress and for troubleshooting issues.

These logs can be retrieved at time using the “task-id” returned as well as by running ConfD “show” commands.

To access these commands, you must be logged in to the ConfD CLI as the *admin* user on the AutoVNF VM:

```
confd_cli -u admin -C
```

When prompted, enter the *admin* user password.

[Table 2: ConfD Log Descriptions, on page 49](#) provides a list of the available commands and describes the information in the output.

**Table 2: ConfD Log Descriptions**

ConfD Command	Purpose
In releases prior to 6.0: <b>show autovnf-oper:errors</b> In 6.0 and later releases: <b>show uas</b>	Displays a list of any deployment errors that may have occurred.
In releases prior to 6.0: <b>show autovnf-oper:logs   display xml</b> In 6.0 and later releases: <b>show log   display xml</b>	Displays log messages for AutoVNF transactions.
In releases prior to 6.0: <b>show autovnf-oper:network-catalog</b> In 6.0 and later releases: <b>show vnf-packager</b>	Displays information for the networks deployed with USP.
In releases prior to 6.0: <b>show autovnf-oper:transactions</b> In 6.0 and later releases: <b>show transaction</b>	Displays a list of transaction IDs that correspond to the USP deployment along with their execution date, time, and status.

ConfD Command	Purpose
In releases prior to 6.0: <b>show autovnf-oper:vdu-catalog</b>  In 6.0 and later releases: <b>show vnfr</b>	Displays information pertaining to the virtual descriptor units (VDUs) used to deploy USP.
In releases prior to 6.0: <b>show autovnf-oper:vip-port</b>  In 6.0 and later releases: <b>show vnfr</b>	Displays information port, network, and virtual IP addresses information.
In releases prior to 6.0: <b>show autovnf-oper:vnfm</b>  In 6.0 and later releases: <b>show vnfr</b>	Displays information pertaining to the VNFM deployment and UEM VM deployment.
<b>show confd-state</b>	Displays information pertaining to confd-state on AutoVNF.
<b>show confd-state ha</b>	Displays information pertaining to HA specific confd-state on AutoVNF.
<b>show log &lt;transaction_id&gt;</b>	Displays detailed log information for a specific transaction ID.
<b>show running-config</b>	Displays the configuration running on the AutoVNF.
<b>show uas</b>	Displays information pertaining to the AutoVNF VM deployment.
In releases prior to 6.0: <b>show usp</b>  In 6.0 and later releases: <b>show vnfr</b>	Displays information pertaining to the overall USP VM deployment.

**NOTES:**

- Log information can be saved out of ConfD to a file for later retrieval using one of the following commands:

```
show log transaction_id | save url
```

OR

```
show autovnf-oper: command | save url
```

Where *transaction\_id* is a specific ID, *url* is a valid directory path, and *command* is one of the command operators identified in [Table 2: ConfD Log Descriptions, on page 49](#).

## Example show confd-state Command Output

### show confd-state

```
confd-state version 6.3.1
confd-state epoll false
confd-state daemon-status started
confd-state ha mode master
confd-state ha node-id confd-master
confd-state ha connected-slave [ a2dd5178-afae-4b3a-8b2b-910216583501 ]
```

NAME	PREFIX	EXPORTED		NAMESPACE
		TO ALL	EXPORTED TO	
iana-crypt-hash			2014-08-06	urn:ietf:params:xml:ns:yang:iana-crypt-hash
ianach		X	-	
ietf-inet-types			2013-07-15	urn:ietf:params:xml:ns:yang:ietf-inet-types
inet		X	-	
ietf-netconf-acm			2012-02-22	urn:ietf:params:xml:ns:yang:ietf-netconf-acm
nacm		X	-	
ietf-netconf-monitoring			2010-10-04	urn:ietf:params:xml:ns:yang:ietf-netconf-monitoring
ncm		X	-	

<-- SNIP -->

## Example show confd-state ha Command Output

### show confd-state ha

```
confd-state ha mode master
confd-state ha node-id confd-master
confd-state ha connected-slave [ a2dd5178-afae-4b3a-8b2b-910216583501 ]
```

## Example show log Command Output

```
show log <transaction_id> | display xml
```

## Example show running-config Command Output

### show running-config

```
<-- SNIP -->
autovnf:secure-token autovnf-admin
  user      $8$YQiswhu0QLpA4N2kBo7t5eZN2uUW0L19m8WaaBzkVoc=
  password  $8$mSaszfxjZ8My8Y/FqLL3Sasn1b/DmRh3pdblatq49cM=
  !
autovnf:secure-token autovnf-oper
  user      $8$kTEQZ4YNdV6BcnH3ggRHJpMhk61sh5KQFqhsQnh/KV8=
  password  $8$KdTbd7ZeYuHrpdKlK5m888ckE3ZGIM7RbEMJwMwCjfo=
  !
autovnf:secure-token em-login
  user      $8$jVDkSMi/wlXzkZj/qx07kEfHB9PlpPlnzCKUSjWiPXA=
  password  $8$52ELrKMilGT/nad5WcPgUh7cijHiizAt8A8Tly79Q/I=
  !
autovnf:secure-token confd-auth
  user      $8$bHYvP179/hlGW08qoTnJFmm8A1Hqq1RESasX+G1SAPw=
  password  $8$S52APq1vb9WhLjbSPNSWiBmAmaG1tzTTmSkkKs8reo=
  !
volume-catalog em-volume
  volume type LUKS
  volume size 1024
  volume bus ide
```

## Example show uas Command Output

```

    volume bootable false
    !
volume-catalog cf-boot
  volume type LUKS
  volume size 16
  volume bus ide
  volume bootable true
  !
volume-catalog cf-cdr
  volume type LUKS
  volume size 200
  volume bus ide
  volume bootable false
  !
autovnf:network-catalog di-internall
  pre-created di-internall
  type          sriov-flat
  physnet       phys_pcie1_0
  ip-prefix     192.168.1.0/24
  dhcp          true
  vlan-tag      true
  vlan          2110
<-- SNIP -->

<-- SNIP -->
autovnf:vdu-catalog control-function
  ha-type          one-to-one
  health-check-frequency 10
  health-probe-max-miss 6
  recovery-type    recovery-restart
  image location  http://172.21.201.63:80/bundles/5.1.0-662/ugp-bundle/qvpc-di-cf.qcow2
  neds netconf
    ned-id          cisco-staros-nc
    port-number     830
    authentication  confd-auth
  !
volumes cf-cdr
!
volumes cf-boot
!
flavor host-aggregate  auto-test-sjc-cf-esc-mgmt1
flavor vcpus           8
flavor ram              16384
flavor root-disk        6
flavor ephemeral-disk  0
flavor swap-disk        0
flavor anti-affinity-placement true
configuration staros_config.txt
  apply-at  day-zero
  source-url http://172.21.201.63:5001/configs/vnf-pkg1/files/system.cfg
<-- SNIP -->

```

## Example show uas Command Output

```

show uas
uas version          6.0.0
uas state            active
uas external-connection-point 50.50.50.67 INSTANCE
IP      STATE  ROLE
-----
10.2.3.6  alive  CONFD-MASTER
10.2.3.11 alive  CONFD-SLAVE

```

NAME	LAST HEARTBEAT
AutoVNF-MASTER	2018-01-20 02:35:03
ESCHearBeatMonitor-fremont-autovnf-vpc	2018-01-20 02:35:00
USPCFMWorker	2018-01-20 02:34:51
USPCHBWorker	2018-01-20 02:35:00
USPCWorker	2018-01-20 02:35:00

**Important**

In this example, 10.2.3.6 is the confd-master and the active UAS VM.

**Important**

In case of standalone mode (non-HA) deployments, the *uas external-connection-point* information and *Instance IP* table are not applicable and are not displayed.

Example output that shows the floating IP for AutoVNF:

```
-SNIP-
nsd autoit
vim-identity vim1
vim-artifact vim_artifact_one
vnf-package [ usp_5_7 ]
vld mgmt
vl-type          management
network-instance bmarconi-management
!
vld orch
vl-type orchestration
network sjc-orch
!
vnfd f-autovnf
vnf-type          usp-uas
version           6.0
high-availability true
nsd               fremont-autovnf
configuration boot-time 1800
configuration set-vim-instance-name true
external-connection-point avf
connection-point eth0
floating-ip enabled
floating-ip external-network public
!
vnfc avf
health-check disabled
health-check boot-time 300
vdu vdu-id autovnf
connection-point eth0
virtual-link service-vl mgmt
!
connection-point eth1
virtual-link service-vl orch
!
!
!
-SNIP-
```

The current version of AutoVNF software can also be seen through the USP UWS – AutoVNF User Interface under –

- the Site Overview screen (Service Deployment > Site) only if the AutoVNF configuration type is a record.
- the Auto-Vnf Configuration Overview screen only if the AutoVNF configuration type is a record.
- the UWS – AutoVNF dashboard.

## Example show vnfr Command Output

### show vnfr

```

vnfr sj-autovnf-esc
vnfd     esc
vnf-type esc
state    deployed
external-connection-point esc
connection-point-instance-id sj-autovnf-esc-ha-vip
virtual-link-ref             uas-management
ip-address                   12.12.12.40
mac-address                   fa:16:3e:6a:db:9b
connection-point-type        virtual-port
port-id                       37a14e07-52f7-48c0-9dbb-471146a709a5
vdu esc
vnfc-instance sj-autovnf-esc-esc-1
state          deployed
vnfc           esc
flavor-key     sj-autovnf-esc
uuid           83f44e0f-380e-4320-a35a-34de82cf84dd
image name     /vnfm-bundle/ESC-4_2_0_74.qcow2
image version  "Version: 4.2.0.74, SHA1: de45b53, Date: Sat Sep 01 08:51:12 EDT 2018"
image package  usp_6_0
image uuid     c35c2a86-6d60-4259-85cc-d023803c7245
host           tb2-compute-15.localdomain
vdu-type       cisco-esc
connection-point-instance eth0
virtual-link-ref             uas-management
ip-address                   12.12.12.22
mac-address                   fa:16:3e:e8:d6:b1
connection-point-type        virtual-port
port-id                       f0f6b82f-336f-4f9f-aae5-d581be8cfa63
connection-point-instance eth1
virtual-link-ref             uas-orchestration
ip-address                   22.22.22.27
mac-address                   fa:16:3e:16:32:4c
connection-point-type        virtual-port
port-id                       f2b7aeae-83f1-4f83-b45e-f92b3a1f6600
vnfc-instance sj-autovnf-esc-esc-2
state          deployed
vnfc           esc
flavor-key     sj-autovnf-esc-esc
uuid           087a5b48-db45-4002-a157-51fa37236545
image name     /vnfm-bundle/ESC-4_2_0_74.qcow2
image version  "Version: 4.2.0.74, SHA1: de45b53, Date: Sat Sep 01 08:51:12 EDT 2018"
image package  usp_6_0
image uuid     c35c2a86-6d60-4259-85cc-d023803c7245
host           tb2-compute-12.localdomain
vdu-type       cisco-esc
connection-point-instance eth0
virtual-link-ref             uas-management
ip-address                   12.12.12.37
mac-address                   fa:16:3e:48:c4:6c
connection-point-type        virtual-port
port-id                       8cb138ab-c575-4eb2-a622-d2648042f48f
connection-point-instance eth1

```

```

    virtual-link-ref    uas-orchestration
    ip-address          22.22.22.28
    mac-address         fa:16:3e:98:78:07
    connection-point-type virtual-port
    port-id             7d73aeae-81e1-410b-ac3a-e34c1bd23c16
vnfr sj-autovnf-vpc
  vnfd      vpc
  vnf-type  ugp
  state     ha-error
external-connection-point cf
  connection-point-instance-id CF-sj-autovnf-vpc-vip
  virtual-link-ref            uas-management
  ip-address                  12.12.12.43
  mac-address                 fa:16:3e:04:80:b7
  connection-point-type      virtual-port
  port-id                     984a6e8b-107a-48f7-b0b4-398a308aff9a
external-connection-point em
  connection-point-instance-id em-sj-autovnf-vpc-vip
  virtual-link-ref            uas-management
  ip-address                  12.12.12.35
  mac-address                 fa:16:3e:b4:7e:b8
  connection-point-type      virtual-port
  port-id                     f47c2150-932c-455f-99c1-7b77fe47a9d7
vdu cf
vnfc-instance sj-autovnf-vpc-cf-0
  state      alive
  vnfc      cf
  flavor-key sj-autovnf-vpc-cf
  uuid      a46de643-b76d-4307-91e8-996b79da4c1e
  image name /ugp-bundle/qvpc-di-cf.qcow2
  image version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  image package usp_6_0
  image uuid 6d63f613-9b46-4bd9-853d-024dcf27f1a7
  host      tb2-compute-9.localdomain
  vdu-type  control-function
connection-point-instance eth0
  virtual-link-ref    di-internall
  ip-address          192.168.10.105
  mac-address         fa:16:3e:46:f8:79
  connection-point-type pnic-sriov
  port-id             b408eedd-8650-44e2-930c-95ee2c9ae380
connection-point-instance eth1
  virtual-link-ref    uas-management
  ip-address          12.12.12.44
  mac-address         fa:16:3e:5e:e0:bc
  connection-point-type virtual-port
  port-id             3e94bcd8-0e58-44e1-99a5-366f7453df02
connection-point-instance eth2
  virtual-link-ref    uas-orchestration
  ip-address          22.22.22.33
  mac-address         fa:16:3e:c5:58:c6
  connection-point-type virtual-port
  port-id             e0a51253-5740-4e34-b4a2-ba6cdaa504cf
vnfc-instance sj-autovnf-vpc-cf-1
  state      alive
  vnfc      cf
  flavor-key sj-autovnf-vpc-cf
  uuid      10ble4c2-d3e5-494c-bec9-26bd38e4c705
  image name /ugp-bundle/qvpc-di-cf.qcow2
  image version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  image package usp_6_0
  image uuid 6d63f613-9b46-4bd9-853d-024dcf27f1a7
  host      tb2-compute-12.localdomain
  vdu-type  control-function

```

## Example show vnfr Command Output

```

connection-point-instance eth0
  virtual-link-ref      di-internall1
  ip-address            192.168.10.99
  mac-address           fa:16:3e:94:3d:38
  connection-point-type pnic-sriov
  port-id               c1df9769-fcdc-4cb1-b7ea-f791ef80ff65
connection-point-instance eth1
  virtual-link-ref      uas-management
  ip-address            12.12.12.47
  mac-address           fa:16:3e:66:27:71
  connection-point-type virtual-port
  port-id               7d77aac2-6409-499a-a4b0-afc4c70e6904
connection-point-instance eth2
  virtual-link-ref      uas-orchestration
  ip-address            22.22.22.45
  mac-address           fa:16:3e:c3:c1:a4
  connection-point-type virtual-port
  port-id               75d7b8c7-1801-4cce-b665-64a060414abd
vdu em
vnfc-instance sj-autovnf-vpc-em-1
  state                ha-error
  vnfc                 em
  flavor-key           sj-autovnf-vpc-em
  uuid                 119edc4c-9ba0-48f8-a928-63e0c3c88f22
  image name           /em-bundle/em-6_3_0_4148.qcow2
  image version        "Version: 6.3.0, SHA1: 40d8f29, Date: Thu Aug 30 22:15:22 EDT 2018"
  image package        usp_6_0
  image uuid           d21b6d92-9964-4db8-8376-4a645fecfbf2
  host                 tb2-compute-14.localdomain
  vdu-type             element-manager
connection-point-instance eth0
  virtual-link-ref      uas-orchestration
  ip-address            22.22.22.40
  mac-address           fa:16:3e:33:57:a6
  connection-point-type virtual-port
  port-id               050d8843-f309-45b3-889a-a1516a338c9f
connection-point-instance eth1
  virtual-link-ref      uas-management
  ip-address            12.12.12.26
  mac-address           fa:16:3e:02:b8:4a
  connection-point-type virtual-port
  port-id               ae8036c5-1a91-488d-98f2-65a8fe57a033
vnfc-instance sj-autovnf-vpc-em-2
  state                ha-error
  vnfc                 em
  flavor-key           sj-autovnf-vpc-em
  uuid                 dd2c9327-c954-49bf-803c-ca38d718da2c
  image name           /em-bundle/em-6_3_0_4148.qcow2
  image version        "Version: 6.3.0, SHA1: 40d8f29, Date: Thu Aug 30 22:15:22 EDT 2018"
  image package        usp_6_0
  image uuid           d21b6d92-9964-4db8-8376-4a645fecfbf2
  host                 tb2-compute-15.localdomain
  vdu-type             element-manager
connection-point-instance eth0
  virtual-link-ref      uas-orchestration
  ip-address            22.22.22.46
  mac-address           fa:16:3e:e5:f7:18
  connection-point-type virtual-port
  port-id               30816589-9a12-4c1d-840c-c84100f714f4
connection-point-instance eth1
  virtual-link-ref      uas-management
  ip-address            12.12.12.45
  mac-address           fa:16:3e:f3:ff:4e
  connection-point-type virtual-port

```



```

    port-id                cf9d991f-e45b-41ed-9ac1-7e6f0bee620b
vdu sf
vnfc-instance sj-autovnf-vpc-sf-0
state         alive
vnfc          sf
flavor-key    sj-autovnf-vpc-sf
uuid          d9b13253-a67e-4078-a75c-04d834577cc2
image name    /ugp-bundle/qvpc-di-xf.qcow2
image version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
image package usp_6_0
image uuid    c65df544-0230-4e86-88bf-4aa93e0e268d
host          tb2-compute-14.localdomain
vdu-type      session-function
connection-point-instance eth0
virtual-link-ref          di-internall
ip-address                192.168.10.95
mac-address               fa:16:3e:87:49:22
connection-point-type     pnic-sriov
port-id                   5d9a9a89-5857-48cb-8081-7273c4b9354c
connection-point-instance eth1
virtual-link-ref          uas-orchestration
ip-address                22.22.22.18
mac-address               fa:16:3e:8f:47:ce
connection-point-type     virtual-port
port-id                   d7dd7006-0134-4767-af02-1922d351d1d5
connection-point-instance eth2
virtual-link-ref          vpc-svc
ip-address                22.11.11.8
mac-address               fa:16:3e:a6:fa:9e
connection-point-type     virtual-port
port-id                   1c5dda23-65f0-4541-ace5-0d6e5e1564ea
vnfc-instance sj-autovnf-vpc-sf-1
state         alive
vnfc          sf
flavor-key    sj-autovnf-vpc-sf
uuid          868158de-e202-4af4-9f3e-c5c7722c5a7f
image name    /ugp-bundle/qvpc-di-xf.qcow2
image version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
image package usp_6_0
image uuid    c65df544-0230-4e86-88bf-4aa93e0e268d
host          tb2-compute-15.localdomain
vdu-type      session-function
connection-point-instance eth0
virtual-link-ref          di-internall
ip-address                192.168.10.97
mac-address               fa:16:3e:bb:ee:38
connection-point-type     pnic-sriov
port-id                   c166c76d-3ef9-4f52-a243-25b49ae0886f
connection-point-instance eth1
virtual-link-ref          uas-orchestration
ip-address                22.22.22.47
mac-address               fa:16:3e:b0:8e:75
connection-point-type     virtual-port
port-id                   9dd61ba8-9455-4f0a-a6ce-13ef28ce6c39
connection-point-instance eth2
virtual-link-ref          vpc-svc
ip-address                22.11.11.13
mac-address               fa:16:3e:25:5a:56
connection-point-type     virtual-port
port-id                   3f8b60aa-4155-4192-b537-afb812d784da

```

## Example show vnf-packager Command Output

### show vnf-packager

```

version      "Version: 6.4.M0, SHA1: cdd46bcm, Build-Number: 0"
image application-function
  image-uri   /ugp-bundle/qvpc-di-xf.qcow2
  vim-id      c65df544-0230-4e86-88bf-4aa93e0e268d
  version     "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  disk-format qcow2
image automation-service
  image-uri   /uas-bundle/usp-uas-6.3.0-0.qcow2
  vim-id      b32d2aeb-9dbe-42f0-99bf-982db8ae7ae8
  version     "Version: 6.3.0, SHA1: 175ea8em, Date: Thu Sep 06 16:17:26 PDT 2018"
  disk-format qcow2
image cisco-esc
  image-uri   /vnfm-bundle/ESC-4_2_0_74.qcow2
  vim-id      c35c2a86-6d60-4259-85cc-d023803c7245
  version     "Version: 4.2.0.74, SHA1: de45b53, Date: Sat Sep 01 08:51:12 EDT 2018"
  disk-format qcow2
image control-function
  image-uri   /ugp-bundle/qvpc-di-cf.qcow2
  vim-id      6d63f613-9b46-4bd9-853d-024dcf27f1a7
  version     "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  disk-format qcow2
image element-manager
  image-uri   /em-bundle/em-6_3_0_4148.qcow2
  vim-id      d21b6d92-9964-4db8-8376-4a645fecfbf2
  version     "Version: 6.3.0, SHA1: 40d8f29, Date: Thu Aug 30 22:15:22 EDT 2018"
  disk-format qcow2
image network-function
  image-uri   /ugp-bundle/qvpc-di-xf.qcow2
  vim-id      c65df544-0230-4e86-88bf-4aa93e0e268d
  version     "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  disk-format qcow2
image session-function
  image-uri   /ugp-bundle/qvpc-di-xf.qcow2
  vim-id      c65df544-0230-4e86-88bf-4aa93e0e268d
  version     "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  disk-format qcow2
image user-plane-function
  image-uri   /ugp-bundle/qvpc-si-21.10.M0.70226.qcow2
  vim-id      078bc882-d29c-4974-a21d-dbf2bc59149b
  version     "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  disk-format qcow2
configuration bootvm
  data-id 1538437650-071830
configuration staros
  data-id 1538437650-060109
vnf-packager 6.4.M0-6133
vnf-package usp_6_t
version      "Version: 6.4.M0, SHA1: cdd46bcm, Build-Number: 6133"
image application-function
  image-uri   /ugp-bundle/qvpc-di-xf.qcow2
  vim-id      d3b3dd85-464d-4b49-90f1-5dc59c9a111b
  version     "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
  disk-format qcow2
image automation-service
  image-uri   /uas-bundle/usp-uas-6.3.0-4206.qcow2
  vim-id      294e5f52-453a-4bd8-8192-b8144607759f
  version     "Version: 6.3.0, SHA1: 175ea8e, Date: Wed Sep 05 06:15:40 EDT 2018"
  disk-format qcow2
image cisco-esc

```

```
image-uri /vnfm-bundle/ESC-4_2_0_74.qcow2
vim-id 87a322cc-3736-407d-855f-f2a566fadd22
version "Version: 4.2.0.74, SHA1: de45b53, Date: Sat Sep 01 08:51:12 EDT 2018"
disk-format qcow2
image control-function
image-uri /ugp-bundle/qvpc-di-cf.qcow2
vim-id 22b34ebf-060c-4e99-8083-e702cef96aca
version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
disk-format qcow2
image element-manager
image-uri /em-bundle/em-6_3_0_4148.qcow2
vim-id c4424476-a9b6-4308-98b3-4aa0f441d5c1
version "Version: 6.3.0, SHA1: 40d8f29, Date: Thu Aug 30 22:15:22 EDT 2018"
disk-format qcow2
image network-function
image-uri /ugp-bundle/qvpc-di-xf.qcow2
vim-id d3b3dd85-464d-4b49-90f1-5dc59c9a111b
version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
disk-format qcow2
image session-function
image-uri /ugp-bundle/qvpc-di-xf.qcow2
vim-id d3b3dd85-464d-4b49-90f1-5dc59c9a111b
version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
disk-format qcow2
image user-plane-function
image-uri /ugp-bundle/qvpc-si-21.10.M0.70226.qcow2
vim-id 8c78ef58-4556-4e8c-bef6-8f98a33bf6c1
version "Version: 21.10.M0.70226, SHA1: NA, Date: Thu Sep 06 10:07:27 EDT 2018"
disk-format qcow2
configuration bootvm
data-id 1538437651-235460
configuration staros
data-id 1538437651-221341
```

## UAS Log Collection

The UAS generates and consolidates a comprehensive set of UAS logs and VNF diagnostic information from StarOS device for troubleshooting purposes. The log collection includes logs from all components in a deployed UAS cluster, i.e. from AutoIT, AutoDeploy, AutoVNF, UEM, ESC and StarOS.

This section describes the following topics:

- [Feature Description, on page 60](#)
- [Limitations, on page 60](#)
- [Collecting the UAS Logs, on page 60](#)
  - [Via the ConfD CLI Command, on page 60](#)
  - [Via the Standalone Script, on page 61](#)
- [Collecting VNF Diagnostic Information, on page 64](#)
- [Sample Logs, on page 64](#)

## Feature Description

Automation of UAS log collection is facilitated through the use of ConfD CLI command **collect-logs**. When this command is executed, the logs from all or required components can be collected and copied to a common location.

## Limitations

The following limitations exist with the UAS log collection feature.

- With the use of ConfD CLI approach, it is not possible to collect logs for AutoDeploy and AutoIT. To aggregate logs for these two components, use the standalone script.
- Direct collection of logs from CF, SF, UP instances is not supported. If VNFDs corresponding to these instances are invoked directly, appropriate error message will be recorded in the log file of AutoDeploy and AutoVNF.

## Collecting the UAS Logs

The UAS logs can be collected using one of the following approaches:

- [Via the ConfD CLI Command, on page 60](#)
- [Via the Standalone Script, on page 61](#)

### Via the ConfD CLI Command

UAS logs collection is automated through a remote procedure call (RPC) executed from the ConfD command line interface (CLI). The RPC “collect-logs” has been introduced to collect logs from the AutoVNF, ESC and UEM.

This command, on execution, fetches the logs from components under given NSD and VNFD levels in a deployed setup and creates a final consolidated tar ball comprising all logs.



#### Important

When the ConfD CLI command is used, the log collection for AutoDeploy and AutoIT components is not supported.

This command can be invoked from ConfD CLI of AutoDeploy and AutoVNF.



#### Important

The IP and credentials of instances are fetched from oper data of AutoDeploy/AutoVNF. So, the logs can be collected for instances whose VNFR is present in oper db.

To collect the UAS logs via ConfD CLI:

1. Login to the ConfD CLI as the *admin* user on the AutoDeploy VM or AutoVNF VM.

```
confd_cli -u admin -C
```

2. Enter the *admin* user password when prompted.

- Execute the following command:

```
collect-logs nsd-id <nsd id>
```

This command collects logs from all components present under given NSD and also from deploy-nsd if present.

To collect logs by VNFD ID, use the following command:

```
collect-logs nsd-id <nsd id> vnfd vnf id
```

This command collects logs from components specific to the given VNFD. Additionally, under given vnf, corresponding AutoVNF logs of the given NSD will also be collected in case of RPC invoked from AutoDeploy (outside AutoVNF).

For example, if VNFD of ESC, or UEM is invoked from AutoDeploy, the corresponding AutoVNF logs of given VNFD will also be collected, since log collection happens through respective AutoVNF in case of multiple AutoVNFs deployed through a single AutoDeploy.

The vnf-diags from StarOS instances will be collected along with the UEM logs.

- View the status of log collection using the AutoDeploy or AutoVNF logs under `/var/log/upstart/` based on where it is invoked.

If invoked from AutoDeploy, the RPC internally connects with AutoVNF and collects logs from UAS instances. The respective progress can be viewed from AutoVNF log.

- Untar the `autocollect_logs.tgz` file to extract the collected logs.

```
tar -zxvf autocollect_logs.tgz
```

Note that the output consolidated tar ball `autocollect_logs.tgz` is created under `/var/log/autocollect/` directory of the instance (AutoDeploy/AutoVNF) from where the log collection RPC was invoked.



### Important

Every time the log collection is triggered, the `/var/log/autocollect` directory will be automatically cleaned and then new logs will be copied.

Example output:

```
vnf-logs/
vnf-logs/em/
vnf-logs/em/abc-vnf-vnf1-em-rmuruga-em1-1-em-logs-2018-08-09_06.41.25.UTC.tar.bz2
vnf-logs/em/abc-vnf-vnf1-em-rmuruga-em1-2-em-logs-2018-08-09_06.40.35.UTC.tar.bz2
vnf-logs/autovnf/
vnf-logs/autovnf/autoit-f-autovnf1-rmuruga-avf-2-uas-logs-2018-08-09_06.40.06.UTC.tar.bz2
vnf-logs/autovnf/autoit-f-autovnf1-rmuruga-avf-1-uas-logs-2018-08-09_06.40.49.UTC.tar.bz2
vnf-logs/esc/
vnf-logs/esc/esc_log_abc-vnf-vnf1-esc-rmuruga-esc-2_2018-08-09_06.40.15.UTC.tar.bz2
vnf-logs/esc/esc_log_abc-vnf-vnf1-esc-rmuruga-esc-1_2018-08-09_06.39.42.UTC.tar.bz2
```

## Via the Standalone Script

The UAS uses a standalone script "`collect_all_uas_logs.py`" to generate and consolidate a comprehensive set of UAS logs and VNF diagnostic information from StarOS device for troubleshooting purposes.

Before using this script, you should be aware of the following:

- AutoVNF IP and SSH credentials (username and password)

- AutoVNF NETCONF login credentials
- Login credentials for UEM and ESC
- Login credentials for AutoIT and AutoDeploy

This script is available in the AutoIT, AutoDeploy and AutoVNF VMs in the `/opt/cisco/usp/uas/scripts/` directory.

A sample yaml file (`sample_config.yaml`) is present in the same directory along with the script. The yaml file should be updated with proper IP and credential details.

Example configuration of `sample_config.yaml` file:

```
uas-cluster:
  autovnf:
    172.21.201.237:
      autovnf:
        login:
          user: ubuntu
          password: Cisco@123
        netconf:
          user: admin
          password: Cisco@123
  .
  .
  .
```




---

#### Important

You can exclude AutoDeploy or AutoIT for log collection by commenting out the phrase ‘autodeploy’ or ‘autoit’ using ‘#’ in the yaml file.

For example:

```
#autodeploy:
```

---

To collect the UAS logs via the standalone script:

1. Log on to the AutoDeploy, AutoIT or AutoVNF VM using the user credentials specified during deployment.
2. Navigate to the `scripts` directory.

```
cd /opt/cisco/usp/uas/scripts
```

3. Edit the `sample_config.yaml` file using a standard text editor to update the appropriate credential details and save it.
4. Launch the `collect_all_uas_logs.py` script to collect the logs.

```
sudo ./collect_all_uas_logs.py - -cfgfile sample_config.yaml
```

The script starts collecting the logs and the progress is displayed on the console. Upon completion, a final tar file `autocollect_logs.tgz` will be copied to `/var/log/autocollect/` directory.

Detailed logs are stored in a log file named `autocollect.log` under the `/var/log/autocollect/` directory.




---

#### Note

The logs directory will be cleared and re-created for every execution of the script.

---

Example output:

```
sudo ./collect_all_uas_logs.py --cfgfile sample_config.yaml
2018-07-26 08:52:57,273 - Uas-cluster present in config
2018-07-26 08:52:57,273 - .....Executing step 1 of 4.....
2018-07-26 08:52:57,273 - Collecting logs from uas instances
2018-07-26 08:53:06,530 - Logs collected successfully from esc 48.48.48.24
2018-07-26 08:53:09,677 - Logs collected successfully from autovnf 38.38.38.23
2018-07-26 08:53:09,784 - Logs collected successfully from esc 48.48.48.16
2018-07-26 08:53:15,404 - Logs collected successfully from autovnf 38.38.38.26
2018-07-26 08:54:21,160 - Logs collected successfully from vnf-em 38.38.38.11
2018-07-26 08:54:24,829 - Logs collected successfully from vnf-em 38.38.38.27
2018-07-26 08:54:24,831 - Generating tar file for uas instance 10.225.202.93
2018-07-26 08:54:25,260 - Log collection completed for uas instance 10.225.202.93.
2018-07-26 08:54:25,261 - Auto-it present in config
2018-07-26 08:54:25,261 - .....Executing step 2 of 4.....
2018-07-26 08:54:25,261 - Collecting logs from Autoit 10.225.202.77
2018-07-26 08:54:25,847 - Logs collected successfully for instance 10.225.202.77
2018-07-26 08:54:25,847 - Auto-deploy present in config
2018-07-26 08:54:25,848 - .....Executing step 3 of 4.....
2018-07-26 08:54:25,848 - Collecting logs from Autodeploy 10.225.202.64
2018-07-26 08:54:26,454 - Logs collected successfully for instance 10.225.202.64
2018-07-26 08:54:26,454 - .....Executing step 4 of 4.....
2018-07-26 08:54:26,454 - Generating consolidated final tar file
2018-07-26 08:54:26,499 - Log collection script completed. Please collect the tar file
'autocollect_logs.tgz' under /var/log/autocollect/
```

5. Extract the contents of *autocollect\_logs.tgz* file using the following command:

```
tar -zxvf autocollect_logs.tgz
```

The following are the contents of the *autocollect\_logs.tgz* tar file:

- .bz2 files for each AutoDeploy and AutoIT logs
- Two log files — *autocollect.log* for AutoDeploy logs and *autocollect-vnf\_<autovnf ip>.log* for logs collected from AutoVNF and its instances.




---

**Important** In case of failures of any UAS cluster nodes, check the *autocollect-vnf\_<autovnf ip>.log* for the detailed error message.

---

- *uas\_logs\_<autovnf ip>.tgz* — This file comprises tar bundles for the UAS cluster.

When the *uas\_logs\_<autovnf ip>.tgz* tar file is extracted, a *vnf-logs* directory is created with individual sub-directories for *em*, *esc*, and *autovnf* for storing the corresponding collected tar file contents.

If the VNF diagnostic log files are unavailable in the *diags* directory under *em*, you can use **vnf-collect-diags** RPC command from UEM VM to collect the logs. For information on collecting logs through the RPC, see the [Collecting VNF Diagnostic Information, on page 64](#) section.

The logs collected through the RPC are also stored in the *diags* directory under *em* directory.

6. Verify that files have been extracted.

```
ls -lt
```

Example output:

```
total 2488
```

```
-rw-r--r-- 1 root root 1174824 Jul 26 08:54 uas_logs_10.225.202.93.tgz
-rw-r--r-- 1 root root 43113 Jul 26 08:54
auto-it-5571-cups-1-uas-logs-2018-07-26_08.54.25.UTC.tar.bz2
-rw-r--r-- 1 root root 3197 Jul 26 08:54 autocollect-vnf_10.225.202.93.log
-rw-r--r-- 1 root root 47104 Jul 26 08:54
auto-deploy-5571-cups-1-uas-logs-2018-07-26_08.54.26.UTC.tar.bz2
-rw-r--r-- 1 root root 3488 Jul 26 08:54 autocollect.log
-rw-r--r-- 1 root root 1266857 Jul 26 08:54 autocollect_logs.tgz
```

## Collecting VNF Diagnostic Information

The VNF diagnostic information can also be collected through the use of **vnf-collect-diags** RPC command in UEM VM.

To collect the diagnostic information from StarOS device:

1. Log on to the master UEM VM.
2. Access the NCS CLI.

```
sudo -i
ncs_cli -u admin -C
```

3. Execute the following command to collect VNF diagnostic logs from StarOS device.

```
vnf-collect-diags [ correlator <correlator id> vnfd <vnfd-id> ]
```

This command collects the VNF diagnostic log files and makes it available in the */var/log/em/diags-<correlator id>* directory.

4. Check the per-VNF diagnostic collection status using the following command:

```
show vnf-collect-diags-status
```

Example output:

```
NAME                                STATUS
-----
xyz_autovnf-cups-c-abc  in-progress
```

## Sample Logs

This section provides a few sample log files.

### AutoDeploy Log:

```
2018-08-09 06:40:10,385 - Deployment uas-log-collection: autoit initiated
2018-08-09 06:40:10,385 - Send Deployment notification for: autoit-instance
2018-08-09 06:40:10,385 - Deploy nsd present under given nsd: abc-vnf
2018-08-09 06:40:10,415 - avf nsd, vnfd is autoit, f-autovnf1
2018-08-09 06:40:10,430 - Direct log collection not supported for vdu-type: control-function
2018-08-09 06:40:10,442 - Uas-cluster present in config
2018-08-09 06:40:10,448 - Collecting logs from uas instances
2018-08-09 06:40:10,453 - Fetching Autovnf, Esc and EM details for 10.225.202.64
2018-08-09 06:40:10,459 - Initiating nc session to 10.225.202.64
/usr/lib/python2.7/dist-packages/Crypto/Cipher/blockalgo.py:141: FutureWarning: CTR mode
needs counter parameter, not IV
  self._cipher = factory.new(key, *args, **kwargs)
2018-08-09 06:40:11,091 - VNFR list fetched successfully
```



```

2018-08-09 06:40:11,097 - Instances details retrieved from Autovnf
2018-08-09 06:40:11,333 - Uas ip details retrieved for 10.225.202.64 : {None: ['45.45.45.53',
'45.45.45.30'], 'esc': ['44.44.44.14', '44.44.44.23'], 'autovnf': ['45.45.45.6',
'45.45.45.28'], 'vnf-em': ['45.45.45.11', '45.45.45.13']}
2018-08-09 06:40:11,509 - Removing staged files from Autovnf
2018-08-09 06:40:11,586 - Files removed successfully
2018-08-09 06:40:11,870 - Connected to Autovnf[10.225.202.64]
2018-08-09 06:40:12,117 - Collecting logs from esc 44.44.44.14
2018-08-09 06:40:17,122 - Collecting logs from esc 44.44.44.23
2018-08-09 06:40:22,125 - Collecting logs from autovnf 45.45.45.6
2018-08-09 06:40:23,364 - Logs collected successfully from autovnf 45.45.45.6
2018-08-09 06:40:23,625 - Logs collected successfully from esc 44.44.44.23
2018-08-09 06:40:27,131 - Collecting logs from autovnf 45.45.45.28
2018-08-09 06:40:32,139 - Collecting logs from vnf-em 45.45.45.11
2018-08-09 06:40:32,230 - Logs collected successfully from autovnf 45.45.45.28
2018-08-09 06:40:32,248 - Logs collected successfully from esc 44.44.44.14
2018-08-09 06:40:37,144 - Collecting logs from vnf-em 45.45.45.13
2018-08-09 06:40:38,332 - Logs collected successfully from vnf-em 45.45.45.13
2018-08-09 06:41:10,963 - Logs collected successfully from vnf-em 45.45.45.11
2018-08-09 06:41:11,084 - All threads finished working for uas instance 10.225.202.64
2018-08-09 06:41:11,091 - Generating tar file for uas instance 10.225.202.64
2018-08-09 06:41:11,315 - Copying the tar file to Autodeploy...
2018-08-09 06:41:11,469 - Tar file successfully copied to Autodeploy under
/var/log/autocollect/ directory
2018-08-09 06:41:11,477 - Log collection completed for uas instance 10.225.202.64.
2018-08-09 06:41:11,491 - Log collection script completed. Please collect the tar file
'autocollect_logs.tgz' under /var/log/autocollect/
2018-08-09 06:41:11,498 - Deployment uas-log-collection: autoit succeeded
2018-08-09 06:41:11,521 - Send Deployment notification for: autoit-instance

```

### AutoVNF Log:

```

2018-08-09 06:40:34,474 - Directory /var/log/autocollect/vnf-logs/esc created successfully
to stage files
2018-08-09 06:40:34,474 - Attempting ssh to 44.44.44.14
2018-08-09 06:40:34,641 - Executing script collect_esc_log.sh in 44.44.44.14
2018-08-09 06:40:39,475 - Attempting ssh to 44.44.44.23
2018-08-09 06:40:39,641 - Executing script collect_esc_log.sh in 44.44.44.23
2018-08-09 06:40:44,477 - Directory /var/log/autocollect/vnf-logs/autovnf created successfully
to stage files
2018-08-09 06:40:44,477 - Attempting ssh to 45.45.45.6
2018-08-09 06:40:44,642 - Executing script collect-uas-logs.sh in 45.45.45.6
2018-08-09 06:40:45,199 - Output file to collect in 45.45.45.6 is
autoit-f-autovnf1-rmuruga2-avf-2-uas-logs-2018-08-09_06.40.06.UTC.tar.bz2
2018-08-09 06:40:45,340 - Collecting bz2 file from instance 45.45.45.6
2018-08-09 06:40:45,346 - Logs collected successfully for instance 45.45.45.6
2018-08-09 06:40:45,415 - Output file to collect in 44.44.44.23 is
esc_log_abc-vnf-vnf1-esc-rmuruga2-esc-2_2018-08-09_06.40.15.UTC.tar.bz2
2018-08-09 06:40:45,623 - Collecting bz2 file from instance 44.44.44.23
2018-08-09 06:40:45,629 - Logs collected successfully for instance 44.44.44.23
2018-08-09 06:40:49,483 - Attempting ssh to 45.45.45.28
2018-08-09 06:40:49,652 - Executing script collect-uas-logs.sh in 45.45.45.28
2018-08-09 06:40:53,979 - Output file to collect in 45.45.45.28 is
autoit-f-autovnf1-rmuruga2-avf-1-uas-logs-2018-08-09_06.40.49.UTC.tar.bz2
2018-08-09 06:40:54,090 - Output file to collect in 44.44.44.14 is
esc_log_abc-vnf-vnf1-esc-rmuruga2-esc-1_2018-08-09_06.39.42.UTC.tar.bz2
2018-08-09 06:40:54,118 - Collecting bz2 file from instance 45.45.45.28
2018-08-09 06:40:54,216 - Logs collected successfully for instance 45.45.45.28
2018-08-09 06:40:54,296 - Collecting bz2 file from instance 44.44.44.14
2018-08-09 06:40:54,357 - Logs collected successfully for instance 44.44.44.14
2018-08-09 06:40:54,491 - Attempting to collect vnf diags through Em 45.45.45.11
2018-08-09 06:40:59,490 - Attempting to collect vnf diags through Em 45.45.45.13
2018-08-09 06:40:59,741 - Vnf diags not collected through 45.45.45.13 as it can be collected
only from Master Em instance
2018-08-09 06:40:59,741 - Directory /var/log/autocollect/vnf-logs/em created successfully

```

```

to stage files
2018-08-09 06:40:59,741 - Attempting ssh to 45.45.45.13
2018-08-09 06:40:59,913 - Executing script collect-em-logs.sh in 45.45.45.13
2018-08-09 06:41:00,178 - Output file to collect in 45.45.45.13 is
abc-vnf-vnf1-em-rmuruga2-em1-2-em-logs-2018-08-09_06.40.35.UTC.tar.bz2
2018-08-09 06:41:00,317 - Collecting bz2 file from instance 45.45.45.13
2018-08-09 06:41:00,322 - Logs collected successfully for instance 45.45.45.13
2018-08-09 06:41:25,592 - Vnf-diags logs collected through EM: 45.45.45.11
2018-08-09 06:41:25,592 - Attempting ssh to 45.45.45.11
2018-08-09 06:41:25,724 - Executing script collect-em-logs.sh in 45.45.45.11
2018-08-09 06:41:32,846 - Output file to collect in 45.45.45.11 is
abc-vnf-vnf1-em-rmuruga2-em1-1-em-logs-2018-08-09_06.41.25.UTC.tar.bz2
2018-08-09 06:41:32,993 - Collecting bz2 file from instance 45.45.45.11
2018-08-09 06:41:33,044 - Logs collected successfully for instance 45.45.45.11

```

### Logs for unsupported vnfd

```

2018-08-09 07:00:11,466 - Deployment uas-log-collection: abc-vnf initiated
2018-08-09 07:00:11,481 - Send Deployment notification for: abc-vnf-instance
2018-08-09 07:00:11,496 - Unsupported Log collection. No oper data retrieved for given vnfd:
vpc-up
2018-08-09 07:00:11,502 - Deployment uas-log-collection: abc-vnf failed
2018-08-09 07:00:11,520 - Send Deployment notification for: abc-vnf-instance

```

# Secure File Transfer

## Feature Description

UAS provides **upload-file** RPC in ConfD to transfer a file or an image to the VNFC components under given NSD and VNFD levels in a deployed setup.

This command can be invoked from ConfD CLI of AutoDeploy and AutoVNF.



### Important

Though the **upload-file** command can be invoked from AutoDeploy and AutoVNF, it is highly important that the **nsd-id** must be specified as AutoVNF name only.

## Limitations

The file transfer cannot be initiated to the components with following vnf-types — UEM, USP-UAS, ESC. That is, if **esc** is specified as **vnfd** in the **upload-file** command, then the file cannot be transferred to ESC.

## How it Works

Perform the following procedure to transfer a file or an image to the VNFs.

1. Log on to AutoDeploy VM or AutoVNF VM as the default user, *ubuntu*.
2. Switch to the *root* user.

```
sudo su
```
3. Enter the ConfD CLI.

```
confd_cli -C -u admin
```

4. Enter the *admin* user password when prompted.
5. Initiate the file transfer to the VNFs using the following command:

For AutoDeploy:

```
nsd:upload-file nsd-id <nsd id> vnfd <vnfd name> source <path of the file>
destination < path >
```

For AutoVNF:

```
upload-file nsd-id <nsd id> vnfd <vnfd name> source <path of the file>
destination < path >
```

Notes:

- The **nsd-id** must always be specified as AutoVNF name.
- **vnfd** is an optional parameter in this configuration. This parameter must be alpha and/or numeric characters, and it accepts more than one value as an input. For example: [ vpc1 ], [ vpc1 vpc2 vpc3 ].
- If **vnfd** is specified in the **upload-file** command and it is a valid VNFD, the file or image is transferred successfully. For the list of invalid or unsupported VNFDs, see the [Limitations, on page 66](#).
- If **vnfd** is not specified in the **upload-file** command, then the file or image is transferred only to the valid VNFDs in the given NSD deployment.
- If the command includes a single invalid VNFD, the file transfer will not be executed and an error indicating invalid argument in AutoDeploy is displayed. For the list of invalid or unsupported VNFDs, see the [Limitations, on page 66](#).

Command example:

```
nsd:upload-file nsd-id abc-autovnf vnfd [ vpc ] source
/home/ubuntu/x.cfg destination /sftp
```

6. Monitor the progress of the file transfer operation.

```
show transaction <transaction-id>
```

*transaction\_id* is the ID displayed as a result of the **upload-file** command executed in the previous step.

Example command output:

```
show transaction 15407
TX ID      TX TYPE  DEPLOYMENT ID      TIMESTAMP                      STATUS  STATUS  DETAIL
15407 upload-file  vnf-autovnf  2018-10-29T05:43:47.666386-00:00  error  -
```

Also, view the logs associated with a specific transaction.

```
show log <transaction-id>
```

## Monitoring File Transfer Operations

AutoDeploy and AutoVNF maintain logs for all transactions in persistent storage. The status/progress of file transfer can be viewed in AutoDeploy/AutoVNF logs archived under */var/log/upstart/* based on where it is invoked.

If invoked from AutoDeploy, then RPC internally connects with AutoVNF and performs the file transfer. The respective progress can be viewed through the AutoVNF logs.

To view the logs associated with a specific transaction:

```
show log <transaction-id>
```

### Sample AutoDeploy Logs:

```
2018-10-26 16:12:30,156 - allowed-address-pair: 90.90.90.0/24 on eth0
2018-10-26 16:12:30,163 - Adding pre-created network: suneduvv-orch into catalog
2018-10-26 16:12:30,169 - Adding uplink action check-liveness-using-ping to eth1
2018-10-26 16:12:30,178 - Found VNF 'suneduvv-autovnf' of type UAS
/usr/lib/python2.7/dist-packages/Crypto/Cipher/blockalgo.py:141: FutureWarning: CTR mode
needs counter parameter, not IV
  self._cipher = factory.new(key, *args, **kwargs)
2018-10-26 16:12:30,634 - Connected to AutoVNF[10.225.202.246]
2018-10-26 16:12:30,641 - dst file name x.cfg
2018-10-26 16:12:30,645 - abs_dest_file /var/cisco/isos/x.cfg
2018-10-26 16:12:30,650 - Skipping copy, file '/var/cisco/isos/x.cfg' already exists
2018-10-26 16:12:30,676 - Updated path to URL in handle_file_transfer
'http://90.90.90.23:5000/isos/x.cfg'
2018-10-26 16:12:31,145 - <?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
message-id="urn:uuid:d9ad94ed-8c42-4059-829c-96182b384b27"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><transaction-id
xmlns='http://www.cisco.com/usp/nfv/usp-nsds'>1540570351-451419</transaction-id>
</rpc-reply>
2018-10-26 16:12:31,150 - Waiting for deployment notifications for tx-id '1540570351-451419'
2018-10-26 16:12:31,155 - [({'urn:ietf:params:xml:ns:netconf:notification:1.0'}notification',
None), ({'urn:ietf:params:xml:ns:netconf:notification:1.0'}eventTime',
'2018-10-26T16:12:31.472658+00:00'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}upload-file-event', '\n '),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}instance-id',
'suneduvv-autovnf-instance'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}descriptor-id', 'suneduvv-autovnf'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}transaction-id', '1540570351-451419'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}operation-type', 'upload-file'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}status', 'requested')]
2018-10-26 16:12:31,160 - Received upload-file-event for suneduvv-autovnf:1540570351-451419
with status:requested
2018-10-26 16:12:31,164 - [({'urn:ietf:params:xml:ns:netconf:notification:1.0'}notification',
None), ({'urn:ietf:params:xml:ns:netconf:notification:1.0'}eventTime',
'2018-10-26T16:12:31.764652+00:00'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}upload-file-event', '\n '),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}instance-id',
'suneduvv-autovnf-instance'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}descriptor-id', 'suneduvv-autovnf'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}transaction-id', '1540570351-451419'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}operation-type', 'upload-file'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}status', 'instantiated')]
2018-10-26 16:12:31,169 - Received upload-file-event for suneduvv-autovnf:1540570351-451419
with status:instantiated
2018-10-26 16:12:31,173 - [({'urn:ietf:params:xml:ns:netconf:notification:1.0'}notification',
None), ({'urn:ietf:params:xml:ns:netconf:notification:1.0'}eventTime',
'2018-10-26T16:12:31.790449+00:00'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}upload-file-event', '\n '),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}instance-id',
'suneduvv-autovnf-instance'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}descriptor-id', 'suneduvv-autovnf'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}transaction-id', '1540570351-451419'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}operation-type', 'upload-file'),
({'http://www.cisco.com/usp/nfv/usp-uas-common-oper'}status', 'in-progress')]
2018-10-26 16:12:31,178 - Received upload-file-event for suneduvv-autovnf:1540570351-451419
with status:in-progress
```

```

2018-10-26 16:12:31,183 - [{"urn:ietf:params:xml:ns:netconf:notification:1.0"}notification',
None), ('{urn:ietf:params:xml:ns:netconf:notification:1.0"}eventTime',
'2018-10-26T16:12:31.842616+00:00'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}upload-file-event', '\n '),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}instance-id',
'suneduvv-autovnf-instance'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}descriptor-id', 'suneduvv-autovnf'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}transaction-id', '1540570351-451419'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}operation-type', 'upload-file'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}status', 'in-progress')]
2018-10-26 16:12:31,188 - Received upload-file-event for suneduvv-autovnf:1540570351-451419
with status:in-progress
2018-10-26 16:12:31,257 - [{"urn:ietf:params:xml:ns:netconf:notification:1.0"}notification',
None), ('{urn:ietf:params:xml:ns:netconf:notification:1.0"}eventTime',
'2018-10-26T16:12:31.925373+00:00'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}upload-file-event', '\n '),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}instance-id',
'suneduvv-autovnf-instance'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}descriptor-id', 'suneduvv-autovnf'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}transaction-id', '1540570351-451419'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}operation-type', 'upload-file'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}status', 'in-progress')]
90.90.90.25 - - [26/Oct/2018 16:12:31] "GET /isos/x.cfg HTTP/1.0" 200 -
2018-10-26 16:12:31,262 - Received upload-file-event for suneduvv-autovnf:1540570351-451419
with status:in-progress
2018-10-26 16:12:32,833 - [{"urn:ietf:params:xml:ns:netconf:notification:1.0"}notification',
None), ('{urn:ietf:params:xml:ns:netconf:notification:1.0"}eventTime',
'2018-10-26T16:12:33.493671+00:00'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}upload-file-event', '\n '),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}instance-id',
'suneduvv-autovnf-instance'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}descriptor-id', 'suneduvv-autovnf'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}transaction-id', '1540570351-451419'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}operation-type', 'upload-file'),
('{http://www.cisco.com/usp/nfv/usp-uas-common-oper}status', 'error')]
2018-10-26 16:12:32,838 - Received upload-file-event for suneduvv-autovnf:1540570351-451419
with status:error
2018-10-26 16:12:32,843 - RPC NS[suneduvv-autovnf:suneduvv-autovnf-instance] failed
2018-10-26 16:12:32,849 - Failed to transfer a file
2018-10-26 16:12:32,854 - Deployment upload-file: suneduvv-autovnf failed
2018-10-26 16:12:32,871 - Send Deployment notification for: suneduvv-autovnf-instance
No handlers could be found for logger "AutoVNF-Traces"
2018-10-26 16:12:32,954 - One or more tasks failed, break the pipeline
2018-10-26 16:12:32,961 - Deployment upload-file: suneduvv-autovnf failed
2018-10-26 16:12:32,982 - Send Deployment notification for: suneduvv-autovnf-instance

```

### Sample AutoVNF Logs:

```

2018-10-26 16:14:10,009 - Waiting for all workers to finish the transactions
2018-10-26 16:14:10,037 - Send Deployment notification for: suneduvv-autovnf-instance
2018-10-26 16:14:10,044 - Deployment upload-file: suneduvv-autovnf started
2018-10-26 16:14:10,050 - DOWNLOADING FILE TO STAGING FOLDER FROM
/home/ubuntu/em-6_3_0_4765.qcow2 ===== /var/cisco/isos/em-6_3_0_4765.qcow2
2018-10-26 16:14:10,057 - URL IS NONE []
2018-10-26 16:14:10,063 - Skipping copy, file '/var/cisco/isos/em-6_3_0_4765.qcow2' already
exists
2018-10-26 16:14:10,070 - I AM HERE56565656 ['vpc']
2018-10-26 16:14:10,087 - vnfrs for the given nsd is suneduvv-autovnf-esc suneduvv-autovnf-vpc
2018-10-26 16:14:10,100 - vnfr_vnfc is [{'vnfr': 'suneduvv-autovnf-esc', 'vnfc': 'esc',
'ip-addr': '90.90.90.32', 'floating-ip': None}, {'vnfr': 'suneduvv-autovnf-vpc', 'vnfc':
'cf', 'ip-addr': '90.90.90.47', 'floating-ip': None}, {'vnfr': 'suneduvv-autovnf-vpc',
'vnfc': 'em', 'ip-addr': '90.90.90.38', 'floating-ip': None}]
2018-10-26 16:14:10,106 - vnfr_vnfd is [{'vnfr': 'suneduvv-autovnf-esc', 'vnfd': 'esc'},
{'vnfr': 'suneduvv-autovnf-vpc', 'vnfd': 'vpc'}]
2018-10-26 16:14:10,112 - vnfdid_list is [{'vnfcid': 'cf', 'fl-ip': None, 'vnfdid': 'vpc',

```

```

'vnfr': 'suneduvv-autovnf-vpc', 'ips': [], 'ha-vip': '90.90.90.47'}, {'vnfcid': 'em',
'fl-ip': None, 'vnfdid': 'vpc', 'vnfr': 'suneduvv-autovnf-vpc', 'ips': [], 'ha-vip':
'90.90.90.38'}}
/usr/lib/python2.7/dist-packages/Crypto/Cipher/blockalgo.py:141: FutureWarning: CTR mode
needs counter parameter, not IV
  self._cipher = factory.new(key, *args, **kwargs)
2018-10-26 16:14:10,429 - Removing staged files from Autovnf
2018-10-26 16:14:10,508 - Files removed successfully
2018-10-26 16:14:10,967 - Copying the file to EM staging...
2018-10-26 16:14:47,430 - XML REQUEST COMMAND <ns0:vnf-put-file
xmlns:ns0="http://www.cisco.com/usp/scm/vnf-utils">
  <file xmlns="http://www.cisco.com/usp/scm/vnf-utils">/tmp/staging/em-6_3_0_4765.qcow2</file>

  <vnfs xmlns="http://www.cisco.com/usp/scm/vnf-utils">
    <vnfd xmlns="http://www.cisco.com/usp/scm/vnf-utils">suneduvv-autovnf-vpc-suneduvv</vnfd>

  </vnfs>
  <destination-path xmlns="http://www.cisco.com/usp/scm/vnf-utils">/fash</destination-path>
</ns0:vnf-put-file>

2018-10-26 16:14:47,602 - rpc executed <?xml version="1.0" encoding="UTF-8"??>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
message-id="urn:uuid:95ebbb6d-aal6-48ba-855b-b73cf14ac5a2"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><status
xmlns='http://www.cisco.com/usp/scm/vnf-utils'>Success</status>
</rpc-reply>
2018-10-26 16:14:47,607 - XML REQUEST FOR STATUS COMMAND <show>
  <vnf-put-files-status xmlns="http://www.cisco.com/usp/scm/vnf-utils"/>
</show>

2018-10-26 16:14:47,676 - res is <?xml version="1.0" encoding="UTF-8"??><data
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><vnf-put-files-status
xmlns="http://www.cisco.com/usp/scm/vnf-utils">Completed</vnf-put-files-status></data>
2018-10-26 16:14:47,681 - status is In progress
2018-10-26 16:14:47,799 - res is <?xml version="1.0" encoding="UTF-8"??><data
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><vnf-put-files-status
xmlns="http://www.cisco.com/usp/scm/vnf-utils">In progress</vnf-put-files-status></data>
2018-10-26 16:14:47,805 - status is In progress
2018-10-26 16:14:47,874 - res is <?xml version="1.0" encoding="UTF-8"??><data
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><vnf-put-files-status
xmlns="http://www.cisco.com/usp/scm/vnf-utils">Completed</vnf-put-files-status></data>
2018-10-26 16:14:47,879 - status is In progress
2018-10-26 16:14:47,997 - res is <?xml version="1.0" encoding="UTF-8"??><data
xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"><vnf-put-files-status
xmlns="http://www.cisco.com/usp/scm/vnf-utils">Completed</vnf-put-files-status></data>
2018-10-26 16:14:48,009 - Deployment upload-file: suneduvv-autovnf failed
2018-10-26 16:14:48,027 - Send Deployment notification for: suneduvv-autovnf-instance
2018-10-26 16:14:48,040 - One or more tasks failed, break the pipeline
2018-10-26 16:14:48,046 - Deployment upload-file: suneduvv-autovnf failed
2018-10-26 16:14:48,103 - Send Deployment notification for: suneduvv-autovnf-instance

```

# Monitoring VNFM Operations



**Note** The Cisco Elastic Services Controller (ESC) is the only VNFM supported in this release.

## Viewing ESC Status

ESC status can be viewed from the ESC command line or by executing a REST API from AutoVNF.

### Monitoring Status Through the ESC Command Line

Log on to the primary ESC VM and execute the following command from the command line:

```
escadm status
```

**Example command output:**

```
0 ESC status=0 ESC Master Healthy
```

### Monitoring Status Through an AutoVNF API

Log on to the master AutoVNF VM and execute the following command:

```
curl -u admin:<password> -k https://<master_vnfm_address>:60000/esc/health
```

**Example command output:**

```
{"message": "ESC services are running.", "status_code": "2000"}
```

Status code and message display information about ESC health conditions as identified in [Table 3: ESC Status Code Messages, on page 71](#). Status codes in the 2000s imply ESC is operational, 5000 status codes imply at least one of the ESC components is not in service.

**Table 3: ESC Status Code Messages**

Code	Message
2000	ESC services are running
2010	ESC services are running. ESC High-Availability node not reachable.
2020	ESC services are running. One or more VIM services (keystone, nova) not reachable.*
2030	ESC services are running. VIM credentials not provided.
2040	ESC services running. VIM is configured, ESC initializing connection to VIM.
2100	ESC services are running. ESC High-Availability node not reachable. One or more VIM services ( nova ) not reachable
5010	ESC service ESC_MANAGER not running.

Code	Message
5020	ESC service CONFD not running.
5030	ESC service MONA not running.
5040	ESC service VIM_MANAGER not running.
5090	More than one ESC service (confd, mona) not running.**

## Viewing ESC Health

ESC health can be viewed by logging on to the primary ESC VM and executing the following command from the command line:

**health.sh**

### Example command output:

```

esc ui is disabled -- skipping status check
esc_monitor start/running, process 840
esc_mona is up and running ...
vimmanager start/running, process 2807

vimmanager start/running, process 2807
esc_confid is started
tomcat6 (pid 2973) is running...           [ OK ]
postgresql-9.4 (pid 2726) is running...
ESC service is running...
Active VIM = OPENSTACK
ESC Operation Mode=OPERATION

/opt/cisco/esc/esc_database is a mountpoint
===== ESC_HA (MASTER) with DRBD =====
DRBD_ROLE_CHECK=0
MNT_ESC_DATABASE_CHECK=0
VIMMANAGER_RET=0
ESC_CHECK=0
STORAGE_CHECK=0
ESC_SERVICE_RET=0
MONA_RET=0
ESC_MONITOR_RET=0
=====
ESC HEALTH PASSED

```

## Viewing ESC Logs

ESC logs are available on the VNF VM in the following directory:

*/var/log/esc/*

Two levels of logs are available for ESC:

- [ESC Logs, on page 73](#)
- [ESC YANG Logs, on page 74](#)



Refer also to the ESC user documentation for additional information on monitoring and maintaining the software.

## ESC Logs

To collect ESC logs:

1. Log on to the primary VNF VM.
2. Navigate to the scripts directory.  
**cd /opt/cisco/esc/esc-scripts**
3. Launch the *collect-esc-logs.sh* script to collect the logs.  
**sudo ./collect-esc-logs.sh**

### Example log output:

We trust you have received the usual lecture from the local System Administrator. It usually boils down to these three things:

```
#1) Respect the privacy of others.
#2) Think before you type.
#3) With great power comes great responsibility.
```

```
[sudo] password for admin: Creating log tarball:
/var/tmp/esc_log-2017-05-25_18.09.31.UTC.tar.bz2
Creating temporary working directory: /var/tmp/esc_log-2017-05-25_18.09.31.UTC
```

```
Dumping thread status of ESCManager from tomcat pid 2973 to catalina.out
escadm-output.txt
vm_info.txt
esc_version.txt
esc/
esc/vimmanager/
esc/vimmanager/operations_vimmanager.log
esc/vimmanager/vimmanager.log
esc/esc_gc.log.2.current
esc/esc_gc.log.0
esc/escmanager.log
esc/event_escmanager.log
esc/escmanager_tagged.log
esc/esc_gc.log.1
esc/custom_script/
esc/pgstartup.log
esc/mona/
esc/mona/actions_mona.log
esc/mona/mona_gc.log.0.current
esc/mona/rules_mona.log
esc/mona/mona.log
tar: esc/mona/mona.log: file changed as we read it
esc/confd/
esc/confd/global.data
esc/confd/devel.log
esc/confd/confd.log
esc/confd/browser.log
esc/confd/audit.log
esc/confd/netconf.trace
esc/confd/netconf.log
esc/spy.log
esc/error_escmanager.log
esc/esc_monitor.log
```

```

esc/esc_haagent.log
esc/yangesc.log
esc/debug_yangesc.log
esc/esc_confd.log
boot.log
secure
messages
dmesg
tomcat6/
tomcat6/localhost.2017-05-24.log
tomcat6/host-manager.2017-05-24.log
tomcat6/manager.2017-05-24.log
tomcat6/catalina.out
tomcat6/catalina.2017-05-24.log
audit/
audit/audit.log
postgresql/data/pg_log/
postgresql/data/pg_log/postgresql-Thu.log
postgresql/data/pg_log/postgresql-Wed.log
esc-config/esc-config.xml
Warning: tar completed with status: 1

Tarball file: /var/tmp/esc_log-2017-05-25_18.09.31.UTC.tar.bz2
Symbolic link: /tmp/esc_log-2017-05-25_18.09.31.UTC.tar.bz2

Suggestions:
1. Transfer the tarball file from the esc vm
2. Remove the tarball and symbolic link (to save ESC disk space):
   sudo rm /var/tmp/esc_log-2017-05-25_18.09.31.UTC.tar.bz2
   sudo rm /tmp/esc_log-2017-05-25_18.09.31.UTC.tar.bz2
3. Command to list contents of tarball:
   tar jtvf esc_log-2017-05-25_18.09.31.UTC.tar.bz2
4. Command to extract from the tarball:
   tar jxf esc_log-2017-05-25_18.09.31.UTC.tar.bz2

```

## ESC YANG Logs

ESC YANG logs are stored in the following file:

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

# Monitoring VNF Operations

## Viewing UEM Service Status

1. Log on to the master UEM VM as the user *ubuntu*.
2. Access the NCS CLI.  
*/opt/cisco/usp/packages/nso/ncs-4.1.1/bin/ncs\_cli -C -u admin*
3. Check the NCS state.

```
show ncs-state ha
```

**Example command output:**

```
ncs-state ha mode master
ncs-state ha node-id 3-1501714180
ncs-state ha connected-slave [ 4-1501714262 ]
```

4. Display the health of cluster.

```
show ems
```

**Example command output:**

EM ID	SLA	SCM	VNFM PROXY	VERSION
3	UP	UP	UP	5.7.0
6	UP	UP	UP	5.7.0



**Important**

The UEM services will no longer run on the slave UEM to simplify troubleshooting, maintenance, and synchronization related issues.

## Viewing UEM Logs

To collect UEM logs:

1. Navigate to the *scripts* directory.

```
cd /opt/cisco/em-scripts
```

2. Launch the *collect-em-logs.sh* script to collect the logs.

```
sudo ./collect-em-logs.sh
```

**Example log output:**

```
Collecting Zookeeper nodes...
Traceback (most recent call last):
  File "/opt/cisco/em-scripts/zk_dump.py", line 2, in <module>
    from kazoo.client import KazooClient
ImportError: No module named kazoo.client

Creating log tarball em-logs-2017-05-26_00.37.28.UTC.tar.bz2 ...
em-logs/
em-logs/upstart/
em-logs/upstart/proxy.log
em-logs/upstart/zk.log
em-logs/upstart/ncs.log
em-logs/scm/
em-logs/scm/audit.log.1.gz
em-logs/scm/ncserr.log.1
em-logs/scm/ncs-java-vm.log.2.gz
em-logs/scm/xpath.trace.1.gz
em-logs/scm/ncs-java-vm.log.1.gz
em-logs/scm/xpath.trace.2.gz
em-logs/scm/ncs-java-vm.log
em-logs/scm/ncserr.log.siz
em-logs/scm/xpath.trace
em-logs/scm/audit.log
em-logs/scm/audit.log.2.gz
em-logs/scm/ncserr.log.idx
em-logs/sla/
em-logs/sla/sla-mgr.log
em-logs/sla/sla-system.log
em-logs/zookeeper/
em-logs/zookeeper/zookeeper.out
```

```

em-logs/zookeeper/zookeeper.log
em-logs/vnfm-proxy/
em-logs/vnfm-proxy/vnfm-proxy.log

===== Tarball available at: /tmp/em-logs-2017-05-26_00.37.28.UTC.tar.bz2
=====

To extract the tarball, run: "tar jxf /tmp/em-logs-2017-05-26_00.37.28.UTC.tar.bz2"

```

## Viewing UEM Zookeeper Logs

The UEM maintains logs on the Zookeeper process. The logs are located in the following directories:

```

/var/log/em/zookeeper/zookeeper.log
/var/log/em/zookeeper/zookeeper.out

```

For logs from 3rd Zookeeper instance, check the following directories:

```

/var/log/em/zookeeper/arbiter/zookeeper.log
/var/log/em/zookeeper/arbiter/zookeeper.out

```

To collect the contents of UEM Zookeeper database, use the following command:

```
sudo ./collect-em-logs.sh -add-zookeeper
```

If "new -add-zookeeper" flag is specified, then by default, zookeeper data is collected in a single file named *zk\_data*.

For a larger deployment, Zookeeper content can be collected in sub-folders. To collect the Zookeeper content in sub-folders, specify "-tree-output" flag as shown in the following command:

```
sudo ./collect-em-logs.sh -add-zookeeper -tree-output
```

When this command is executed, Zookeeper contents are collected under *zk\_data* directory.

## Viewing VNF Information through the Control Function

Information on the VNF deployment can be obtained by executing commands on the Control Function (CF) VNFC. To access the CF CLI:

1. Open an SSH connection to the IP address of the management interface associated with CF1.
2. Press **Enter** to bring up the log in prompt.
3. Enter the username and password.
4. At the Exec mode prompt, enter each of the following commands and observe the results to ensure that the VNF components have been properly deployed according to the desired configuration:

Command	Purpose
<b>show card table</b>	Displays all VM types (e.g. CF, SF, NF, and AF) that have been deployed.
<b>show crash list</b>	Displays software crash events records and associated dump files (minicore, NPU or kernel) for all crashes or a specified crash event. Verify that there are no new or unexpected crashes listed.

Command	Purpose
<b>show emetrl vdu list</b>	Displays card to VM mappings for the VNF. Each card should have a valid universally unique identifier (UUID).
<b>show rct stats</b>	Displays statistics associated with Recovery Control Task (RCT) events, including migrations, switchovers and shutdowns. RCT statistics are associated with card-to-card session recovery activities.
<b>show session progress</b>	Displays session progress information for the current context filtered by the options specified. Check for any active or new calls before proceeding with a deactivation.
<b>show version verbose</b>	Displays the software version that has been deployed.
<b>show vdu summary</b>	Displays general information pertaining to the virtual descriptor units (VDUs) that have been deployed.
In releases prior to 6.0: <b>show usf vdu all</b> In 6.0 and later releases: <b>show vnfr vdu all</b>	Displays detailed information for the VDUs that have been deployed for the USF VDU.
In releases prior to 6.0: <b>show usf vdu-group all</b> In 6.0 and later releases: <b>show vnfr vdu-group all</b>	Displays information for VDU groups pertaining to the USF VNF use case (if deployed).
In releases prior to 6.0: <b>show usf network-path all</b> In 6.0 and later releases: <b>show vnfr network-path all</b>	Displays network path information for USF VNF components (if deployed).
In releases prior to 6.0: <b>show usf service-function-chain all</b> In 6.0 and later releases: <b>show vnfr service-function-chain all</b>	Displays SFC information for the USF VNF (if deployed).

## Monitoring and Recovering AutoVNF Through AutoIT

AutoIT provides the ability to monitor and auto-recover AutoVNF instances.

This functionality is enabled through configuration of the AutoVNF VNFC(s) at the time of deployment. Once enabled, AutoIT automatically monitors for faults/failures of the AutoVNF VNFC(s) for which the functionality is enabled. If a fault/failure is detected, AutoIT automatically attempts to auto-heal/recover (redeploy) the VNFC(s).



**Important** The Provisioning Network (floating) IP address is required to leverage the health monitoring functionality.

The following parameters must be configured at the VNFC-level:

**Table 4: Health Check Descriptor Parameters**

Parameter	Required	Type	Description
enabled	O	bool	Enable/Disable health monitoring.
probe-frequency	O	uint16	Health Check Frequency in seconds. UAS uses this as health probe time, meaning every polling interval UAS will invoke health check. Default value is 10 seconds.
probe-max-miss	O	uint16	Maximum number of health probe misses before VNFC instance is declared dead. Default value is 6.
recovery-type	O	choice string	Recovery type. It can be one of the following: <ul style="list-style-type: none"> <li>• restart: Recovery only by restarting, move the VNFC instance to error after max retries</li> <li>• external: Recovery performed by external entity. No auto-recovery</li> <li>• restart-then-redeploy: Restart the VM on failure. After maximum retries, redeploy the failed VNFC instances.</li> </ul> Default value is restart-then-redeploy.
retry-count	O	uint16	Number of retries to recover the VNFC Instance. Default value is set to restart-then-redeploy.
boot-time	O	uint16	Initial Bootup time for the VNFC. Default value is 300 seconds.
script	O	string	Script to check VNFC health, by default UAS ICMP script will be used.

The above parameters are configured at the VNFC-level within the VNF descriptor information that is part of the deployment network service descriptor (NSD) as shown in the following example configuration:

```
nsd <nsd_name>
...
  vnfd <autovnf_vnfd_name>
...
```

```
vnfc <autovnf_vnfc_name>
health-check enabled
health-check probe-frequency 10
health-check probe-max-miss 6
health-check retry-count 6
health-check recovery-type restart-then-redeploy
health-check boot-time 300
...
```

Refer to the *Cisco Ultra Services Platform NETCONF API Guide* for more information on the use of these and other parameters related to VNF configuration and deployment.

In the event that automatic recovery is not possible, an API is available to manually recover the VNFC(s).

VNFC status can be viewed by executing the **show vnfr** command from AutoIT. Additional details can be found in the transaction logs for the deployment.

To manually recover a failed AutoVNF VNFC, execute the following command:

```
recover nsd-id <nsd_name> vnfd <vnfd_name>
```

## Monitoring and Recovering VNFC Through AutoVNF

The UEM, CF, and SF VNFCs were autorecovered through the VNFM (ESC). In these situations, AutoVNF was not informed of these events. With this release, the AutoVNF monitors these VNFC VMs and can auto-recover them if required. Additionally, the AutoVNF can also monitor the VNFM (ESC) VMs and provide auto-recovery as needed.

This functionality is enabled through configuration of the VNFC(s) at the time of deployment. Once enabled, AutoVNF automatically monitors for faults/failures of the VNFCs for which the functionality is enabled. If a fault/failure is detected, AutoVNF automatically attempts to auto-heal/recover (redeploy) the VNFC(s).



### Important

The Provisioning Network (floating) IP address is required to leverage the health monitoring functionality.

This functionality is currently only supported for the following VNFCs:

- VNFM (ESC)
- UEM
- CF
- SF



### Important

Ultra M Manager sends fault notification when VMs are down and/or recovered.

The following parameters must be configured at the VNFC-level:

**Table 5: Health Check Descriptor Parameters**

Parameter	Required	Type	Description
enabled	O	bool	Enable/disable health monitoring.

Parameter	Required	Type	Description
probe-frequency	O	uint16	Health Check Frequency in seconds. UAS uses this as health probe time, meaning every polling interval UAS will invoke health check. Default value is 10 seconds.
probe-max-miss	O	uint16	Maximum number of health probe misses before VNFC instance is declared dead. Default value is 6.
recovery-type	O	choice string	Recovery type. It can be one of the following: <ul style="list-style-type: none"> <li>• restart: Recovery only by restarting, move the VNFC instance to error after max retries</li> <li>• external: Recovery performed by external entity. No auto-recovery</li> <li>• restart-then-redeploy: Restart the VM on failure. After maximum retries, redeploy the failed VNFC instances.</li> </ul> Default value is restart-then-redeploy.
retry-count	O	uint16	Number of retries to recover the VNFC Instance. Default value is set to restart-then-redeploy.
boot-time	O	uint16	Initial bootup time for the VNFC. Default value is 300 seconds.
script	O	string	Script to check VNFC health, by default UAS ICMP script will be used.

The above parameters are configured at the VNFC-level within the VNF descriptor information that is part of the deployment network service descriptor (NSD) as shown in the following example configuration:

```

nsd <nsd_name>
...
  vnfd <autovnf_vnfd_name>
...
    vnfc <vnfm_vnfc_name>
      health-check enabled
      health-check probe-frequency 10
      health-check probe-max-miss 6
      health-check retry-count 6
      health-check recovery-type restart-then-redeploy
      health-check boot-time 300
...
    vnfc <uem_vnfc_name>
      health-check enabled
      health-check probe-frequency 10
      health-check probe-max-miss 6
      health-check retry-count 6
      health-check recovery-type restart-then-redeploy
      health-check boot-time 300
...
    vnfc <cf_vnfc_name>

```



```

health-check enabled
health-check probe-frequency 10
health-check probe-max-miss 6
health-check retry-count 6
health-check recovery-type restart-then-redeploy
health-check boot-time 300
...
vnfc <sf_vnfc_name>
health-check enabled
health-check probe-frequency 10
health-check probe-max-miss 6
health-check retry-count 6
health-check recovery-type restart-then-redeploy
health-check boot-time 300
...

```

Refer to the *Cisco Ultra Services Platform NETCONF API Guide* for more information on the use of these and other parameters related to VNF configuration and deployment.

In the event that automatic recovery is not possible, an API is available to manually recover the VNFC(s).

VNFC status can be viewed by executing the **show vnfr** command from AutoIT. Additional details can be found in the transaction logs for the deployment.

To manually recover a failed VNFC, execute the following command:

```
recover nsd-id <nsd_name> vnfd <vnfd_name>
```

## Troubleshooting Deactivation Process and Issues

### NOTES:

- The deactivate process is idempotent and can be multiple times and without error. The system will retry to remove any resources that remain.
- If a deactivation fails (a transaction failure occurs), look at the logs on various UAS software components (AutoDeploy, AutoIT, and AutoVNF), VNFM (ESC), and UEM.
- If deactivation has failed, you must ensure that a clean up is performed either using automation tools or manually if necessary.
- Activation must not be reattempted until all of the previous artifacts have been removed.

## Deactivation Fails Due to Communication Errors with AutoVNF

### Problem Description

During the AutoVNF deactivation process, AutoDeploy indicates that it is unable to deactivate the AutoVNF. This is observed through:

- AutoDeploy transaction log
- AutoDeploy upstart log

### Possible Cause(s)

- AutoDeploy is not able to communicate with AutoVNF.

**Action(s) to Take**

- Check network connectivity between the AutoDeploy VM and the AutoVNF VIP.
- Check the management and orchestration network.
- Address any connectivity issues.

**Next Steps**

- Once connectivity issues are addressed, perform the deactivate procedure again.

## Deactivation Fails Because AutoDeploy Generates an Exception

**Problem Description**

AutoDeploy generates an exception error during the deactivation process.

**Possible Cause(s)**

- Connectivity issues
- Configuration issues
- OpenStack/VIM specific issues
- Hardware issues

**Action(s) to Take**

1. Capture logs from `/var/log/upstart/autodeploy.log` along with exception error message.
2. Log on to AutoIT and collect the logs from `/var/log/upstart/autoit.log` along with the exception message, if any.
3. Log on to VIP of the active (master) AutoVNF VM and perform a cleanup by running the **deactivate** command from there.
  - a. Log on to the AutoVNF VM as the default user, *ubuntu*.
  - b. Switch to the root user.
 

```
sudo su
```
  - c. Enter the ConfD CLI.
 

```
confd_cli -C -u admin
```
  - d. Enter the *admin* user password when prompted.
  - e. Deactivate the deployment.
 

```
deactivate nsd-id <nsd_name>
```
4. Check the last transaction log to verify that the deactivation was successful. (Transactions are auto-sorted by timestamp, so it should be the last one in the list.)

**Example commands and outputs:**

```
show transactions
```

TX ID	TX TYPE	ID	TIMESTAMP	STATUS
DETAIL				
1500605583-055162	vnf-deployment	dep-5-5	2017-07-21T02:53:03.055205-00:00	deployment-failed -
1500606090-581863	vnf-deployment	dep-5-5	2017-07-21T03:01:30.581892-00:00	deployment-success -
1500606127-221084	vnf-deployment	dep-5-5	2017-07-21T03:02:07.221114-00:00	deployment-success -

```

show log 1500606127-221084 | display xml
<config xmlns="http://tail-f.com/ns/config/1.0">
  <log xmlns="http://www.cisco.com/usp/nfv/usp-autovnf-oper">
    <tx-id>1500606127-221084</tx-id>
    <log>2017-07-21 03:02:07,276 - Notify deployment
2017-07-21 03:02:07,297 - Connection to VNFM (esc) at 172.16.181.107
2017-07-21 03:02:07,418 - NETConf Sessions (Transaction/Notifications) established
...

```

- Manually delete the AutoDeploy VM using the information in [Terminating the AutoDeploy VM, on page 2](#).

#### Next Steps

- Open a support case providing all of the log information that was collected.

## Deactivation Fails Because of AutoVNF-VNFM Communication Issues

### Problem Description

During the AutoVNF deactivation process, AutoVNF indicates that it is unable to deactivate the VNFM. This is observed through:

- AutoVNF transaction log
- AutoVNF upstart log

### Possible Cause(s)

- AutoVNF is not able to communicate with the VNFM.

### Action(s) to Take

- Check network connectivity between the master AutoVNF VM and the VNFM VIP.
- Check the management and orchestration network.
- Address any connectivity issues.

### Next Steps

- Once connectivity issues are addressed, perform the deactivate procedure again.

## Deactivation Fails Because of Issue at VNFM

### Problem Description

During the AutoVNF deactivation process, the VNFM returns an error. This is observed through:

- AutoVNF transaction log
- AutoVNF upstart log
- ESC logs

#### Possible Cause(s)

- ESC health is not good due to an issue or network connectivity.
- ESC is not able to communicate with the VIM.
- ESC has an internal error.
- AutoVNF is unable to create/delete OpenStack artifacts.

#### Action(s) to Take

1. Check `/var/log/esc/yangesc.log` for any issues or error messages.
2. Run `health.sh` to determine the health of ESC.
3. Check network connectivity and address an issues. Retry the deactivation.
4. Check network connectivity with the VIM and address any issues. Retry the deactivation.
5. Determine if ESC has a deployment configuration. From the active ESC VM:

```
/opt/cisco/esc/confd/bin/confd_cli -C
show running-config
```

If a configuration is present, most likely ESC is still retrying the deactivation, allow more time for the process to continue.

If no configuration exists, check if there are deployment artifacts still on the VIM. Retry the deactivation.

6. Collect logs by running `collect_esc_log.sh` from both the active and standby ESC VMs.
7. Perform a manual cleanup.




---

**Note** Only artifacts which UAS created need to be removed. Any pre-created artifacts must remain in place.

---

- a. Login on to the VIM as tenant.
- b. Remove all VMs.
- c. Remove all VIP Ports.
- d. Remove all networks.
- e. Remove all flavors.
- f. Remove all volumes.
- g. Remove all images.
- h. Remove host-aggregate created as part of automation.

### Next Steps

- Open a support case providing all of the log information that was collected.

## Deactivation Fails Because AutoVNF Generates an Exception

### Problem Description

AutoVNF generates an exception error during the deactivation process.

### Possible Cause(s)

- Connectivity issues
- Configuration issues
- OpenStack/VIM specific issues
- Hardware issues

### Action(s) to Take

1. Collect all logs from `/var/log/cisco-uas`.
2. Perform a manual cleanup.



---

**Note** Only artifacts which UAS created need to be removed. Any pre-created artifacts can remain in place.

---

- a. Login on to the VIM as tenant.
- b. Remove all VMs.
- c. Remove all VIP Ports.
- d. Remove all networks.
- e. Remove all flavors.
- f. Remove all volumes.
- g. Remove all images.
- h. Remove host-aggregate created as part of automation.

### Next Steps

- Open a support case providing all of the log information that was collected.

## Troubleshooting UEM Issues

This section contains information on troubleshooting UEM issues.

## UEM VM Stuck in a Boot Loop

### Problem Description

Processes that normally run on the UEM VM are unable to start and the VM is stuck in a boot-loop.

### Possible Cause(s)

There is an error with the Zookeeper database keeping the Zookeeper process and other UEM processes from starting. (No other UEM process can be started unless the Zookeeper process has started.)



#### Important

Release 6.3 onwards, no UEM services will run on the slave UEM.

### Action(s) to Take

1. Check the UEM Zookeeper logs. Refer to [Viewing UEM Zookeeper Logs, on page 76](#).
2. Look for error messages similar to the following:

```
[myid:4] - INFO [main:FileSnap@83] - Reading snapshot
/var/lib/zookeeper/data/version-2/snapshot.5000004ba
[myid:4] - ERROR [main:QuorumPeer@557] - Unable to load database on disk
java.io.EOFException
```

If the above errors exist, proceed to the next step. If not, further debugging is required. Please contact your local support representative.

3. Rebuild the Zookeeper database.
  - a. Check the health of Master and Slave EM instances. Execute the following commands on each instance.

#### Master UEM VM:

```
sudo -i
ncs_cli -u admin -C
admin connected from 127.0.0.1 using console on deploymentem-1
```

#### show ems

EM ID	SLA	SCM	VNFM PROXY	VERSION
3	UP	UP	UP	5.7.0
6	UP	UP	UP	5.7.0

```
exit
```



#### Important

Only the master UEM status may be displayed in the above command because the slave UEM is in the boot loop.

#### show ncs-state ha

```
ncs-state ha mode master
ncs-state ha node-id 6-1506059686
ncs-state ha connected-slave [ 3-1506059622 ]
```

#### Slave UEM VM:



**Important** The slave UEM may not be accessible if it is experiencing the boot loop issue.

```

sudo -i
ncs_cli -u admin -C
admin connected from 127.0.0.1 using console on deploymentem-1

show ems
EM          VNFM
ID  SLA  SCM  PROXY  VERSION
-----
3   UP   UP   UP     5.7.0
6   UP   UP   UP     5.7.0

exit

show ncs-state ha
ncs-state ha mode slave
ncs-state ha node-id 3-1506059622
ncs-state ha master-node-id 6-1506059686

```

- b. Log on to the node on which Zookeeper data is corrupted.
- c. Enable the debug mode.

```

/opt/cisco/em-scripts/enable_debug_mode.sh
Disable EM reboot. Enable debug mode

```

- d. Reboot the VM in order to enter the debug mode.
- e. Remove the corrupted data.

```

cd /var/lib/zookeeper/data/
ls
myid  version-2  zookeeper_server.pid

```

```

mv version-2 version-2_old

```

Check for corrupted data in the arbiter directory as well.

```

cd /var/lib/zookeeper/arbiter/data
ls
myid  version-2  zookeeper_server.pid

```

```

mv version-2 version-2_old

```



**Important** This process removes the Zookeeper database by renaming it for additional debugging/recovery.

- f. Reboot the node instance for it to reconcile and rebuild the Zookeeper database from a healthy UEM instance.

```

reboot

```

- g. Log on to the UEM VM upon reboot.
- h. Validate that the database has been successfully rebuilt on the previously failing UEM node.

```
sudo -i
ncs_cli -u admin -C
admin connected from 127.0.0.1 using console on vnfddeploymentem-0
```

**show ems**

```
EM          VNFM
ID  SLA  SCM  PROXY  VERSION
-----
3   UP   UP   UP     5.7.0
6   UP   UP   UP     5.7.0
```

**show ncs-state ha**

```
ncs-state ha mode slave
ncs-state ha node-id 3-1506093933
ncs-state ha master-node-id 6-1506093930
```

**exit**

```
cd /var/lib/zookeeper/data/
```

**ls**

```
myid version-2 version-2_old zookeeper_server.pid
```

**cat /var/log/em/zookeeper/zookeeper.log**

```
<---SNIP--->
2017-09-22 15:25:35,192 [myid:3] - INFO
[QuorumPeer[myid=3]/0:0:0:0:0:0:0:2181:Follower@61] - FOLLOWING - LEADER ELECTION
TOOK - 236
2017-09-22 15:25:35,194 [myid:3] - INFO
[QuorumPeer[myid=3]/0:0:0:0:0:0:0:2181:QuorumPeer$QuorumServer@149] - Resolved
hostname: 30.30.62.6 to address: /30.30.62.6
2017-09-22 15:25:35,211 [myid:3] - INFO
[QuorumPeer[myid=3]/0:0:0:0:0:0:0:2181:Learner@329] - Getting a snapshot from leader
2017-09-22 15:25:35,224 [myid:3] - INFO
[QuorumPeer[myid=3]/0:0:0:0:0:0:0:2181:FileTxnSnapLog@240] - Snapshotting:
0x200000050 to /var/lib/zookeeper/data/version-2/snapshot.200000050
2017-09-22 15:25:37,561 [myid:3] - INFO
[NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:NIOServerCxnFactory@192] - Accepted socket
connection from /30.30.62.15:58011
2017-09-22 15:25:37,650 [myid:3] - WARN
[NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:ZooKeeperServer@882] - Connection request
from old client /30.30.62.15:58011; will be dropped if server is in r-o mode
2017-09-22 15:25:37,652 [myid:3] - INFO
[NIOServerCxn.Factory:0.0.0.0/0.0.0.0:2181:ZooKeeperServer@928] - Client attempting
to establish new session at /30.30.62.15:58011
<---SNIP--->
```

Also, check the logs in the `/var/log/em/zookeeper/arbitrator` directory.

- i. Disable the UEM debug mode on the VM on which the Zookeeper database was rebuilt.

```
/opt/cisco/em-scripts/disable_debug_mode.sh
```

```
Disable debug mode
```

**Next Steps**

Open a support case providing all the log information that was collected.