HP Helion OpenStack 1.1.1: Update Procedure

Contents

HP Helion OpenStack® 1.1.1: Update Procedure

Welcome to the HP Helion OpenStack v1.1.1 update instructions. These instructions apply to existing HP Helion OpenStack v1.1 installations and describe how you can update your HP Helion OpenStack cloud environment from v1.1 to v1.1.1. (The process of updating a Helion OpenStack v1.0 or v1.01 release to Helion OpenStack v1.1.1 is not supported.)

Note to Helion Development Platform users: The HP Helion Development Platform requires HP Helion OpenStack v1.1.1. However, HP Helion OpenStack v1.1.1 does not currently support updating the HP Helion Development Platform from v1.1 to v1.2.

Note: You can also deploy the HP Helion OpenStack v1.1.1 release from scratch. For those instructions, please see *Installing Helion OpenStack*.

The process of updating HP Helion OpenStack v1.1 to HP Helion OpenStack v1.1.1 consists of three procedures:

1. *Update the seed host*
2. *Update the undercloud*
3. *Update the overcloud*

These procedures are described in the following sections. You must perform each of these three procedures in the order presented: first the seed host, then the undercloud, and finally, the overcloud.

Allow a full day for the HP Helion OpenStack v1.1.1 update to complete. During the update process, affected systems will be offline.

Note: Nodes to be updated must be in a "good" state before running the update. Specifically, being in a "good" state means:

- Nodes must be up
- Nodes must be running
- Heat state of each node to be updated must not include any FAIL status

Note: You must take a backup of the seed, undercloud, and overcloud using the standard procedures before commencing the update process.

Deployment considerations

Check that the deployed nodes (all seed VM, undercloud, and overcloud nodes) are accessible (SSH) and pingable. Note the network and bridge configurations on all nodes using the `ifconfig` and `ovs-vsctl show` commands.

Check the version of the Glance images that were used to deploy each of the overcloud nodes before the update. To check the Glance image versions:

1. SSH to the undercloud node.
2. Enter:

```
source stackrc
glance image-list
nova list
nova show <instance id>
```

3. Compare the Image ID returned from the `glance image-list` command to that returned from the `nova show` command.
4. Make sure the following commands are working before you start the update:
   - keystone user-list

- nova list
- neutron net-list
- glance image-list
- os-refresh-config
- os-collect-config

5. If you are unsure of the version of Helion you are running, from any Helion node (seed VM, undercloud, or overcloud node), enter:

```
# cat /etc/HP_Helion_version
```

A sample of the output is:

```
HP Helion Openstack 1.1 Build 81
```

Helion v1.1 to v1.1.1 seed host update process

The procedure will guide you through updating to Helion v1.1.1 from the seed host using `update_sd.sh`.

Note: You must have the `jq` package for your version of Ubuntu installed on the seed host. If you need to install this package, enter:

```
apt-get install jq
```

To prepare the seed VM:

1. Log into the seed VM by entering:

```
ssh root@<seed-ip>
```

2. Remove any update folders and content from previous update attempts in the `/root` folder using the `rm` command.
3. From `/root`, create the `helion-update-1.1-to-<version>` directory then change to this directory by entering:

```
cd /root
mkdir helion-update-1.1-to-<version>
cd helion-update-1.1-to-<version>
```

4. Download the update kit `HP_Helion_Openstack` to the `helion-update` directory on the seed VM.
5. Extract the update kit by entering:

```
tar -xvzf `HP_Helion_Openstack
```

6. Copy the `kvm-custom-ips.json` file to the new directory by entering:

```
cp ~/tripleo/configs/kvm-custom-ips.json ~/helion-update-1.1-to-<version>/
tripleo/configs/kvm-custom-ips.json
```

7. Verify that there is enough free space on the system to carry out the update. The space you need must support a copy of the following folders:
   - /root/tripleo
   - /mnt/state
   - /root/helion-update/helion-update-1.1-to-1.1.1
   - /tftpboot
8. To see the amount of space these folders require, enter:

```
du -hcs /root/tripleo /mnt/state /root/helion-update* /tftpboot
```

9. Check the free space on the system by entering:

```
df -h /
```

   The space required cannot exceed the space available (that is, the output from the `du -hcs` command).

   If you need to free up space, you can safely remove the `HP_Helion_Openstack` update kit that you just extracted
10. To prepare the seed host for starting the update, verify that the following files exist in the seed VM.
    - /root/eca.key
    - /root/eca.crt

    If you specified the ephemeral CA key and certificate from a different directory using environment variables `EPHEMERAL_CA_KEY_FILE` and `EPHEMERAL_CA_CERT_FILE`, you must copy them to the seed VM `/root` directory. Then rename them to `eca.crt` and `eca.key`.

    If you do not have these files, you may be able to regenerate them from the installer data using the following commands:

    echo -e $(jq '.parameters.EphemeralCaKey' ../tripleo/overcloud-env.json | sed 's/\"//g') > /root/eca.key echo -e $(jq '.parameters.EphemeralCaCert' ../tripleo/overcloud-env.json | sed 's/\"//g') > /root/eca.crt
11. If you generated these files from the installer data, you need to verify that the files are well formed by entering:

```
openssl rsa -noout -in /root/eca.key
openssl x509 -noout -in /root/eca.crt
```

    If nothing displays, it means the file is readable and at least reasonably well formatted. Otherwise, a message similar to this one displays:

```
unable to load certificate
140170817259168:error:0906D06C:PEM routines:PEM_read_bio:no start
 line:pem_lib.c:703:Expecting: TRUSTED CERTIFICATE".
```

12. From `/root` on the seed host, create the `helion-update-1.1-to-<version>` directory then change to this directory by entering:

```
cd /root
mkdir helion-update-1.1-to-<version>
cd helion-update-1.1-to-<version>
```

13. Copy the seed update script from the downloaded update kit on the seed VM to the seed host by entering:

```
scp root@<seed-vm-ip>:/root/helion-update-1.1-to-<version>/tripleo/helion-update/seed_update/update_sd.sh .
```

14. Run the seed update script `update_sd.sh` from the seed host. Make sure you run `update_sd.sh` from the update directory you created: `/root/helion-update-1.1-to-<version>)` using the seed host's `tripleo` root directory (`/root/tripleo`) and remote update root (extracted updated kit directory on the seed VM) as arguments.

    For example:

```
host_tripleo_root_directory = /root/tripleo
remote_update_tripleo_root_directory = /root/helion-update-1.1-to-<version>
```

15. To source the `kvm-custom-ips.json` file and run the update script enter:

```
source ~/tripleo/tripleo-incubator/scripts/hp_ced_load_config.sh ~/tripleo/configs/kvm-custom-ips.json
./update_sd.sh <host_tripleo_root_directory>
 <remote_update_tripleo_root_directory> | tee seed_update.log
```

    NOTE: If you used any custom variables for the initial installation of the seed, you will need to export them again prior to executing the `update_sd.sh` script.

    Once the update is completed, the `ssh host_key` of the seed will have changed.

16. To purge the known_hosts entries on the seed host machine, enter:

```
ssh-keygen -R <seed_ip_address>
```

    You will be prompted to re-add the seed host to your known_hosts file when you SSH into the seed to update the undercloud and overcloud.

HP Helion OpenStack 1.1 to 1.1.X undercloud update process

The procedure will guide you through updating the undercloud by running the `update_uc.sh` shell script.

Important: If you are updating to a release that includes new Orchestration (Heat) templates, you must apply these new templates to your system before you attempt to update it.

The update process does not automatically overwrite your Heat template, which would cause you to lose your modifications. If you have modified your Heat template, then you will need to update the new templates with these modifications.

To apply updated Heat templates, as root, run:

```
  ./tripleo/tripleo-incubator/scripts/hp_ced_installer.sh --update-undercloud
```

To update your HP Helion OpenStack undercloud from v1.1 to v1.1.1:

1. Log on to the seed VM.

```
ssh root@<seed-ip>
```

2. Download the update kit onto the seed VM.

```
cd /root
mkdir helion-update-1.1-to-<version>
cd helion-update-1.1-to-<version>
wget http://<url>/helion_ee_<version>.tgz
```

3. Extract the update kit, copy the `kvm-custom-ips.json`, and run the update script.

```
tar -xvzf helion_ee_<version>.tgz
cp ~/tripleo/configs/kvm-custom-ips.json ~/helion-update-1.1-to-<version>/
tripleo/configs/kvm-custom-ips.json
source tripleo/tripleo-incubator/scripts/hp_ced_load_config.sh tripleo/
configs/kvm-custom-ips.json
cd tripleo/helion-update/undercloud_update
./update_uc.sh ~/helion-update-1.1-to-<version>/tripleo
```

4. Your HP Helion OpenStack undercloud is now updated. Confirm that update was successful by examining the `/var/log/ansible/ansible.log`. This report should not list any unreachable nodes:

```
PLAY RECAP
 ********************************************************************
10.23.67.141   : ok=122  changed=74   unreachable=0failed=0
10.23.67.143   : ok=37   changed=22   unreachable=0failed=0
10.23.67.144   : ok=122  changed=74   unreachable=0failed=0
10.23.67.145   : ok=103  changed=62   unreachable=0failed=0
10.23.67.146   : ok=37   changed=22   unreachable=0failed=0
10.23.67.147   : ok=60   changed=30   unreachable=0failed=0
10.23.67.148   : ok=60   changed=30   unreachable=0failed=0
```

5. Change file permissions by entering:

```
chmod 775 /mnt/state/var/eon
chmod 775 /mnt/state/var/eon/data
```

Helion v1.1 to v1.1.1 overcloud update process

This procedure explains how to update the overcloud using `update_oc.sh`.

The v1.1.1 release includes new Orchestration (Heat) templates and new passthrough files. To apply these new templates to your system before updating the overcloud:

1. Log on to the seed VM.

```
ssh root@<seed-ip>
```

2. Download the update kit on the seed.

```
cd /root
mkdir helion-update-1.1-to-<version>
cd helion-update-1.1-to-<version>
wget http://<url>/helion_ee_<version>.tgz
```

3. Extract the update kit:

```
tar -xvzf helion_ee_<version>.tgz
```

4. Copy the new templates from the location where you unpacked the new kit to the original `/root/tripleo/tripleo-heat-templates` directory

```
cp -r /root/helion-update-1.1-to-<version>/tripleo/tripleo-heat-
templates /root/tripleo/
```

Note: if you have made modifications to your original heat templates, you will need to re-apply them to the new templates. These instructions are included in this procedure.

5. Copy the new passthrough files from the location where you unpacked the new kit to the original `/root/tripleo/hp_passthrough` directory.

```
cp -r /root/helion-update-1.1-to-<version>/tripleo/hp_passthrough /root/
tripleo/
```

Note: if you have made modifications to your original passthrough files, you will need to re-apply them to the new passthrough files. These instructions are included in this procedure.

6. Copy the new installation scripts from the location where you unpacked the new kit to the original `/root/tripleo/tripleo-incubator/scripts` directory

```
cp -r /root/helion-update-1.1-to-<version>/tripleo/tripleo-incubator/
scripts /root/tripleo/tripleo-incubator/
```

7. Copy the new `my.cnf` image element from the undercloud to each of the overcloud controller nodes. Log on to the seed and source the seed credentials. To do this, enter:

```
root@hLinux:~# . stackrc
root@hLinux:~# undercloudip=`nova list | grep ctlplane | awk '{print $12}'
 | sed -e "s/.*=\\([0-9.]*\\).*/\1/"`
root@hLinux:~# scp heat-admin@${undercloudip}:/usr/libexec/os-apply-
config/templates/mnt/state/etc/mysql/my.cnf /tmp/my.cnf
```

Source the undercloud creds and copy the file to each of the overcloud controllers by entering:

```
TE_DATAFILE=/root/tripleo/ce_env.json . /root/tripleo/tripleo-incubator/
undercloudrc
for ip in `nova list | grep controller[0-2] | awk '{print $12}' | sed -e
 "s/.*=\\([0-9.]*\\).*/\1/"`; do
```

```
echo $ip;
scp /usr/libexec/os-apply-config/templates/mnt/state/etc/mysql/my.cnf
 heat-admin@$ip:~
ssh heat-admin@$ip "sudo cp ~heat-admin/my.cnf /usr/libexec/os-apply-
config/templates/mnt/state/etc/mysql/"
done
```

8. Create the new `from-heat.conf` file on the seed using the following command:

```
root@hLinux:~# cat > from-heat.conf
{{#stunnel}}
{{#connect_host}}
pid = /var/run/stunnel4/from-heat.pid
cert = {{stunnel.cert_location}}
key = {{stunnel.key_location}}
options = NO_SSLv2
options = NO_SSLv3
debug = 4
output = /var/log/stunnel4/helion_stunnel.log
syslog = no

{{#verify}}
verify = {{{verify}}}
{{/verify}}
{{#ports}}

[{{{name}}}]
accept = {{#accept_host}}{{{.}}}:{{/accept_host}}{{{accept}}}
connect = {{connect_host}}:{{connect}}
{{#client}}
client = {{.}}
{{/client}}
{{#timeout}}
TIMEOUTclose = {{.}}
{{/timeout}}
{{#session_cache}}
sessionCacheSize = {{.}}
{{/session_cache}}
{{#ciphers}}
ciphers = {{{.}}}
{{/ciphers}}
{{/ports}}
{{/connect_host}}
{{/stunnel}}
```

9. Copy the new stunnel image element to `/usr/libexec/os-apply-config/templates/etc/stunnel/from-heat.conf` on each of the overcloud controllers. Log on to the seed and source the undercloud credentials and copy the file to each of the overcloud controllers by entering:

```
TE_DATAFILE=/root/tripleo/ce_env.json . /root/tripleo/tripleo-incubator/
undercloudrc
for ip in `nova list | grep controller[0-2] | awk '{print $12}' | sed -e
 "s/.*=\\([0-9.]*\\).*/\1/"`; do
echo $ip;
scp from-heat.conf heat-admin@$ip:~
ssh heat-admin@$ip "sudo cp ~heat-admin/from-heat.conf /usr/libexec/os-
apply-config/templates/etc/stunnel/"
ssh heat-admin@$ip "sudo /usr/local/bin/os-apply-config  --validate"
done
```

10. Run `update-overcloud` to apply the new templates and passthrough files by logging on to the seed as root and changing to the original /root directory and running:

```
cd /root
source tripleo/tripleo-incubator/scripts/hp_ced_load_config.sh tripleo/
configs/kvm-custom-ips.json
./tripleo/tripleo-incubator/scripts/hp_ced_installer.sh --update-overcloud
```

Note: If you defined any custom variables for the initial installation of the overcloud, you need to export them again before executing the hp_ced_installer.sh script. For example:

```
root@hLinux:~# OVERCLOUD_COMPUTESCALE=5
OVERCLOUD_NEUTRON_DVR=True
OVERCLOUD_NTP_SERVER=10.22.33.44
./tripleo/tripleo-incubator/scripts/hp_ced_installer.sh --update-overcloud
 --skip-demo |& tee /roo/tq_up_oc_1.log
```

The above command should apply the new templates and passthrough files. However, because the images have not yet been updated, the process should end in failure with the message:

```
++ wait_for 30 10 nova service-list --binary nova-compute '2>/dev/null'
 '|' grep 'enabled.*\ up\ '
Timing out after 300 seconds:
COMMAND=nova service-list --binary nova-compute 2>/dev/null | grep
 enabled.*\ up\
OUTPUT=
root@hLinux:~#
```

This is expected behavior for KVM-based systems.

11. Log onto each overcloud controller in turn and restart MySQL and stunnel. This ensures that they are listening on the correct ports. To restart MySQL, enter:

```
root@overcloud-ce-controller-controller0-xoni6co2hxes:~# service mysql
 stop
[ ok ] Stopping MySQL (Percona XtraDB Cluster): mysqld.
root@overcloud-ce-controller-controller0-xoni6co2hxes:~# service mysql
 start
[ ok ] Starting MySQL (Percona XtraDB Cluster) database server:
 mysqld . . . .[....] SST in progress, setting sleep higher: mysqld ..
root@overcloud-ce-controller-controller0-xoni6co2hxes:~# service mysql
 status
[info] Percona XtraDB Cluster up and running.
```

To restart stunnel, enter:

```
root@overcloud-ce-controller-controller0-xoni6co2hxes:~# service stunnel4
 stop
Stopping SSL tunnels: [stopped: /etc/stunnel/from-heat.conf] stunnel.
root@overcloud-ce-controller-controller0-xoni6co2hxes:~# service stunnel4
 start
```

```
Starting SSL tunnels: [Started: /etc/stunnel/from-heat.conf] stunnel.
root@overcloud-ce-controller-controller0-xoni6co2hxes:~# service stunnel4
 status
SSL tunnels status: /etc/stunnel/from-heat.conf: running
root@overcloud-ce-controller-controller0-xoni6co2hxes:~#
```

Verify that MySQL is now listening on port 3307 by entering:

```
root@overcloud-ce-controller-controller0-xoni6co2hxes:~# netstat -napd |
 grep 3307 | grep LIST
tcp        0      0 127.0.0.1:3307          0.0.0.0:*               LISTEN
      32049/mysqld
root@overcloud-ce-controller-controller0-xoni6co2hxes:~#
```

Warning: Do not proceed with the following steps if MySQL is not listening on port 3307.

Note: The Nova service-list will report that overcloud services are 'down'. At this stage, this is expected behavior. (You may need to wait for approximately 10 minutes to allow `os-apply-config` to complete running on each controller node before `nova service-list` returns this information.)

12. Apply the following edit to the pre-flight_check.yml file:

```
root@hLinux:~# sed -i 's/status=started/state=started/g' ~/helion-
update-1.1-to-<version>/tripleo/helion-update/tripleo-ansible/playbooks/
pre-flight_check.yml
```

13. Add the 61-sleep playbook by creating the 61-sleep file as follows:

```
root@hLinux:~# cat > ~/helion-update-1.1-to-<version>/tripleo/helion-
update/tripleo-ansible/playbooks/files/61-sleep
#!/bin/bash
set -x
echo 'Short sleep'
sleep 20
date
```

14. Update the playbook by entering:

```
    root@hLinux:~# cp -p ~/helion-update-1.1-to/tripleo/helion-update/
tripleo-ansible/playbooks/step_run_occ.yml ~/helion-update-1.1-to-/
tripleo/helion-update/tripleo-ansible/playbooks/step_run_occ.yml.orig
    root@hLinux:~# cat &lt; ~/helion-update-1.1-to-
/tripleo/helion-update/tripleo-ansible/playbooks/step_run_occ.yml
    # Copyright (C) 2014 Hewlett-Packard Development Company, L.P.
    #
    # Licensed under the Apache License, Version 2.0 (the "License");
    # you may not use this file except in compliance with the License.
    # You may obtain a copy of the License at
    #
    #    http://www.apache.org/licenses/LICENSE-2.0
    #
    # Unless required by applicable law or agreed to in writing, software
    # distributed under the License is distributed on an "AS IS" BASIS,
    # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
    # express or implied.
    # See the License for the specific language governing permissions and
```

```
    # limitations under the License.
    ---
    - name: Remove os-collect-config disable sentinel file
      file: path=/mnt/state/disable-os-collect-config state=absent
      sudo: yes
    - name: "Setting default fact to run os-collect-config"
      set_fact: test_bypass_os_collect_config="False"
    - name: "Evaluate if os-collect-config needs to be run"
      command: grep -q -i "Completed phase migration" /var/log/upstart/os-
collect-config.log
      register: test_did_os_collect_config_complete
      ignore_errors: yes
      when: online_upgrade is not defined
    - name: "Setting fact to bypass os-collect-config if applicable"
      set_fact: test_bypass_os_collect_config="True"
      when: online_upgrade is not defined and
 test_did_os_collect_config_complete.rc == 0
    - name: "SLEEP20"
      copy:
        dest: /opt/stack/os-config-refresh/configure.d/61-sleep
        src: files/61-sleep
        owner: root
        group: root
        mode: 0755
    - name: "Execute os-collect-config"
      command: os-collect-config --force --one
      when: test_bypass_os_collect_config != true
```

15. To verify that the new file is correct, please run:

```
root@hLinux:~# python -c 'import yaml; f=open("/root/helion-update-1.1-
to-<version>/tripleo/helion-update/tripleo-ansible/playbooks/
step_run_occ.yml", "r"); yaml.safe_load(f);'
root@hLinux:~# echo $?
0
```

16. Run the update script:

```
source tripleo/tripleo-incubator/scripts/hp_ced_load_config.sh tripleo/
configs/kvm-custom-ips.json
cd helion-update-1.1-to-<version>/tripleo/helion-update/overcloud_update
./update_oc.sh ~/helion-update-1.1-to-<version>/tripleo |& tee
 update_oc.log
```

Your HP Helion OpenStack overcloud is now updated.

17. Confirm that update was successful by examining the `/var/log/ansible/ansible.log`. This report should not list any unreachable nodes:

```
PLAY RECAP
 ********************************************************************
10.23.67.141   : ok=122  changed=74   unreachable=0failed=0
```

```
10.23.67.143   : ok=37   changed=22   unreachable=0failed=0
10.23.67.144   : ok=122  changed=74   unreachable=0failed=0
10.23.67.145   : ok=103  changed=62   unreachable=0failed=0
10.23.67.146   : ok=37   changed=22   unreachable=0failed=0
10.23.67.147   : ok=60   changed=30   unreachable=0failed=0
10.23.67.148   : ok=60   changed=30   unreachable=0failed=0
```

Post update cleanup

After you have successfully updated the overcloud, there are some files that you should delete. Perform the following steps on all nodes in the overcloud (including controller nodes, Swift nodes, Nova compute nodes, etc.) To find and delete these files:

1. Stop the `os-collect-config` service by entering:

```
service os-collect-config stop
```

2. Find the files named `host_metadata.js*` by entering:

```
find / -name host_metadata.json* -ls
```

3. Delete the files by entering:

```
find / -name host_metadata.json* -delete
```

4. Verify that the files are all gone by entering:

```
find / -name host_metadata.json* -ls
```

   There should be no files found.

5. Restart the service by entering:

```
service os-collect-config start
```

Updating the HP-version type

It is a good practice to copy the `HP_Helion_Version` type to the Tripleo directory so that it reflects the version that was updated. This lets other administrators know that the environment has been updated.

To replace the version file with the latest, on the seed VM, enter:

```
cp ~/helion-update-1.1-to-<version>/tripleo/HP_Helion_Version ~/tripleo/
```

Validating the update

This section explains how you should validate your update.

Validating the overcloud controller update

After the update, the overcloud controller should be back online without any errors. You should be able to SSH to the Nova compute node from the seed host. You should be able to open the Horizon dashboard as well as execute lifecycle operations with any instance already deployed in the overcloud.

The node should re-join the MySQL Cluster, HAProxy should show the servers as online, the instances deployed before the update should still be up and accessible. It should be possible to deploy new instances and ping them and SSH to them.

Icinga should show that all the monitors are green for the update node. To check Icinga, go to the **Icinga Dashboard** (http://<Undercloud_IP>/icinga/). Log in with the credential: icingaadmin/icingaadmin

Do the following:

1. Enable the HAProxy status page and check if the all the services from the node are green (meaning they are up/running).
2. SSH to the overcloud Management controller.
3. Edit the file `/etc/haproxy/haproxy.cfg` to add at the end of the file `mode http`. (Make this edit in the listen `haproxy.stats`:1993 section.)
4. Restart HAProxy (`service haproxy restart`).
5. Open the status page and check all the services.
6. Make sure that the following commands are working after the update:
   - keystone user-list
   - nova list
   - neutron net-list
   - glance image-list
   - os-refresh-config
   - os-collect-config
7. Use `ifconfig` to check that all your networks still exist after the update.
8. Use `ovs-vsctl show` to check that all bridges and ports still exist after the update.

Validating the compute node

After an update, verify that instances deployed before the update are still up and accessible. You should be able to SSH to the Nova Compute node from the seed host. You should be able to ping and SSH to new instances.

The `os-refresh-config` command needs to be working after an update.

Use `ifconfig` to check that all your networks still exist after the update. Use `ovs-vsctl show` to check that all bridges and ports still exist after the update.

Validating Swift

After the update:

- Check that the images are still there.
- Confirm that you can SSH to the Nova compute node from seed host.
- Verify that you can load new images.
- Check that you can deploy new instances with a new image.

The `os-refresh-config` command needs to be working after an update.

Validating Virtual Storage Appliances

After an update:

- Check if the node can join the cluster (CMC).
- Verify that you can SSH to the Nova compute node from the seed host.

- Check that the volumes are still present.
- Verify that you can create new volumes.
- Verify that you can delete old volumes.
- Verify that you can attach/detach volumes.
- Check that the
- version of the Glance image used to update is the same by SSHing to the undercloud node and run:
  - `glance image-list`
  - `nova list`
  - `nova show <instance id>`

Compare the Image ID from `glance image-list` with `nova show`.

Troubleshooting an update

This section explains how to fix common problems that might arise when performing an update to Helion.

Backup fails

If you see the error message:

```
  + echo 'ERROR: Backup of seed failed!!!!'
ERROR: Backup of seed failed!!!!
+ exit 1
```

Your backup attempt has failed. To recover from this state, you need to get `rabbitmq`, `mysql` and `Openstack` services running again by running the seed recovery script. to do this, enter:

```
  ssh root@<seed-ip>
cd helion-update-1.1-to-<version>/tripleo/helion-update/seed_update
./seed_recover.sh
```

Backup failed due to seed host running out of disk space

If your backup fails because the disk lacks sufficient space you will note that the backup sequence froze or failed. You will also see that the disk space utilization check of the seed host returns 100%. You can check disk utilization by entering:

```
  df -h /
```

To fix this problem:

1. Remove the failed backup from the `/tmp` folder by entering:

   ```
   rm -r -f /tmp/backp_root
   ```

2. Restart MySQL and RabbitMQ by entering:

   ```
   service mysql restart
   service rabbitmq-server restart
   ```

3. Run `os-collect-config` by entering:

```
servie os-collect-config stop
os-collect-config --force
service os-collect-config start
```

If this command returns an error, RabbitMQ may be in a bad state as a result of running out of disk space. If executing `os-collect-config` errors with RabbitMQ, enter:

```
rabbitmqctl stop_app
rabbitmqctl reset
service rabbitmq-server stop
```

4. Once RabbitMQ has been reset, re-attempt step 3 which will re-configure RabbitMQ and restart services. If the `rabbitmqctl reset` command does not work, use the `rabbitmqctl force_reset` command.

Seed host update fails noting unable to ping 192.0.2.1

If your seed host update fails, your deployment will NOT utilize the 192.0.2.0/24 demo IP network range. You will see in your log:

```
  "Waiting for seed host to configure"
```

Pings to host 192.0.2.1 will time out with the following message:

```
  Timing out after 10 seconds:
COMMAND=ping -c 1 192.0.2.1
OUTPUT=PING 192.0.2.1 (192.0.2.1) 56(84) bytes of data.
```

The likely reason for this failure is the installer update of the seed host image has failed as the configuration that was used during the install was not available or passed on to the installer to perform the seed host update operation. As a result, the installer has indicated that it has failed, although the seed host has likely rebooted without issue.

To fix this problem:

1. Identify the following:
   - Expected IP address of the seed host.
   - Location of the backup folder. It should be at `/root/helion-update-1.1-to-<version>/backup*`. For example:`/root/helion-update-1.1-to-1.1.74/backup-0/`
2. Verify that you can ping the IP address of the seed host by entering:

```
ping -c 1 <seed IP address>
```

3. Run the restore script directly by entering:

```
/root/helion-update-1.1-to-<version>/helion-update/seed_update/
seed_update.sh --restore-seed <backup directory> --ip-address <seed IP
 address>
```

```
```

For example:

```
/root/helion-update-1.1-to-<version>/helion-update/seed_update/
seed_update.sh --restore-seed /root/helion-update-1.1-to-<version>/
backup-0/ --ip-address 192.2.0.1
```

4. The restoration command will take some time, but once completed you should be able to log in to the seed host. To verify that the node is in working status, enter:

```
source /root/stackrc &&
nova list &&
glance image-list &&
heat stack-list &&
neutron net-list
```

Retrying failed actions

In some cases, steps may fail because some components may still be initializing and not yet be ready for use. In this event, you have two options: to re-attempt or resume playbook executions.

1. Use the Ansible `ansible-playbook` command with the `--start-at-task="TASK NAME"` option. This command allows resumption of a playbook, when used with the `-l limit` option.
2. Use the Ansible `ansible-playbook` command with the `--step` option. This command allows you to confirm each task before it is executed by Ansible.

A node goes to ERROR state during rebuild

A node can go into an error state due to network errors or a temporary overload of the undercloud. This can happen from time to time due to network errors or a temporary overload of the undercloud. In this case, the `nova list` command returns node in ERROR. To fix this:

- Make sure your hardware is in working order.
- Verify that approximately 20% of the disk space on the Ironic server node is free.
- Get the image ID of the machine in question using `nova show`

```
nova show $node_id
```

- Manually rebuild by running:

```
nova rebuild --preserve-ephemeral $node_id $image_id
```

A node times out after rebuild

While rare, there is the possibility that something unexpected happened during a rebuild and the host has failed to reboot. When this happens, you will get this error Message:

```
msg: Timeout waiting for the server to come up.. Please check manually
```

To fix this problem, follow the steps detailed in: "A node goes to ERROR state during rebuild".

MySQL CLI configuration file is missing

Should the post-rebuild restart fail, the cause might be that the MySQL CLI configuration file is missing. If you have this issue, attempts to access the MySQL CLI command return:

```
 ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using
password: NO)
```

To fix this problem:

- Verify that the MySQL CLI config file stored on the state drive is present and has content. To display the contents, run:

  ```
  sudo cat /mnt/state/root/metadata.my.cnf
  ```

- If the file is empty, retrieve current metadata and update the config files on disk by running:

  ```
  sudo os-collect-config --force --one --command=os-apply-config
  ```

- Verify that the MySQL CLI config file is present in the root user directory by running:

  ```
  sudo cat /root/.my.cnf
  ```

- If that file does not exist, or is empty, you have two options.

  1. Add the following to your MySQL CLI command line:

     ```
     --defaults-extra-file=/mnt/state/root/metadata.my.cnf
     ```

  2. Or copy the configuration from the state drive by entering:

     ```
     sudo cp -f /mnt/state/root/metadata.my.cnf /root/.my.cnf
     ```

MySQL fails to start after retrying the update

If the update was aborted or failed during the Update sequence before a single MySQL controller was operational, MySQL will fail to start upon retrying. In this case, you will see the following error messages:

```
  * `msg: Starting MySQL (Percona XtraDB Cluster) database server:
 mysqld . . . . The server quit without updating PID file (/var/run/mysqld/
mysqld.pid)`

* `stderr: ERROR 2002 (HY000): Can't connect to local MySQL server through
 socket '/var/run/mysqld/mysqld.sock' (111)`

* `FATAL: all hosts have already failed  - aborting`
```

```
* Update automatically aborts.
```

WARNING:

The command `/etc/init.d/mysql bootstrap-pxc`, mentioned below, should only ever be executed when an entire MySQL cluster is down, and then only on the last node to have been shut down. Running this command on multiple nodes will cause the MySQL cluster to enter a split brain scenario effectively breaking the cluster which will result in unpredictable behavior.

To fix this problem:

- Verify that NTP is working.
- Use the `nova list` command to determine the IP address of the controller0 node, then SSH into it by entering:

```
ssh heat-admin@$IP
```

- Verify that MySQL is down by running (as root) the `mysql` client. It should fail:

```
sudo mysql -e "SELECT 1"
```

- Attempt to restart MySQL in case another cluster node is online. This should fail in this error state. However, if it succeeds your cluster should again be operational and you can skip the next step:

```
sudo /etc/init.d/mysql start
```

- Start MySQL in single node bootstrap mode by entering:

```
sudo /etc/init.d/mysql bootstrap-pxc
```

MySQL/Percona/Galera is out of sync

OpenStack is configured to store all of its states in a multi-node, synchronous replication Percona XtraDB Cluster database, which uses Galera for replication. This database must be in sync and have the full complement of servers before updates can be performed safely.

The problem is update fails with errors about Galera and/or MySQL being `Out of Sync`.

To fix this issue:

- Use the `nova list` command to determine the IP address of controller0 node, then SSH to it by entering:

```
ssh heat-admin@$IP
```

- Verify that replication is out of sync by entering:

```
sudo mysql -e "SHOW STATUS like 'wsrep_%'"
```

- Stop MySQL by entering:

```
```

```
sudo /etc/init.d/mysql stop
```

- Verify that MySQL is down by running (as root) the `mysql` client. It should fail:

```
sudo mysql -e "SELECT 1"
```

- Start controller0 MySQL in single-node bootstrap mode by entering:

```
sudo /etc/init.d/mysql bootstrap-pxc
```

- On the remaining controller nodes observed to be having issues, get their IP addresses using the `nova list` command and log in to them by entering:

```
ssh heat-admin@$IP
```

- Verify that replication is out of sync by entering:

```
sudo mysql -e "SHOW STATUS like 'wsrep_%'"
```

- Stop MySQL by entering:

```
sudo /etc/init.d/mysql stop
```

- Verify that MySQL is down by running (as root) the `mysql` client. It should fail:

```
sudo mysql -e "SELECT 1"
```

- Start MySQL so it attempts to connect to controller0 by entering:

```
sudo /etc/init.d/mysql start
```

If restarting MySQL fails, then the database is most certainly out of sync and the MySQL error logs, located at `/var/log/mysql/error.log`, will need to be checked. In this case, never attempt to restart MySQL with `sudo /etc/init.d/mysql bootstrap-pxc` as it will bootstrap the host as a single-node cluster thus worsening what already appears to be a split-brain scenario. If you need help with this matter, contact HP support.

MySQL "Node appears to be the last node in a cluster" error

This error occurs when one of the controller nodes does not have MySQL running. The playbook has detected that the current node is the last running node, although based on its sequence, it should not be the last node. As a result the error is thrown and update is aborted.

The full message is:

```
 Galera Replication. Node appears to be the last node in a cluster; cannot
safely proceed unless overridden via `single_controller`setting. See
README.rst
```

To fix this problem:

- Run the `pre-flight_check.yml` playbook. It will attempt to restart MySQL on each node in the `Ensuring MySQL is running` step. If that step succeeds, you should be able to re-run the playbook and not encounter this `Node appears to be last node in a cluster` error.
- If pre-flight_check fails to restart MySQL, review the MySQL logs (`/var/log/mysql/error.log`) to determine why the other nodes are not restarting.

SSH connectivity is lost

Ansible uses SSH to communicate with remote nodes. In heavily loaded, single host virtualized environments, SSH can lose connectivity. It should be noted that similar issues in a physical environment may indicate issues in the underlying network infrastructure.

When Ansible loses SSH connectivity causing an update attempt to fail, you will see the following output:

```
  fatal: [192.0.2.25] => SSH encountered an unknown error. The
output was: OpenSSH_6.6.1, OpenSSL 1.0.1i-dev xx XXX xxxx
debug1: Reading configuration data /etc/ssh/ssh_config debug1:
/etc/ssh/ssh_config line 19: Applying options for * debug1:
auto-mux: Trying existing master debug2: fd 3 setting
O_NONBLOCK mux_client_hello_exchange: write packet: Broken
pipe FATAL: all hosts have already failed  - aborting
```

To fix this problem, you can generally re-run the playbook to complete the upgrade, unless SSH connectivity is lost while all MySQL nodes are down. (See 'MySQL fails to start upon retrying update' to correct this issue.)

Early Ubuntu Trusty kernel versions have known issues with KVM which severely impact SSH connectivity to instances. To avoid this issue, Test hosts should have a minimum kernel version of 3.13.0-36-generic.

The update steps, running as root, are:

```
  apt-get update
apt-get dist-upgrade
reboot
```

If you continue to encounter this issue in a physical environment, check the network infrastructure for errors.

Similar error messages may occur with long running processes, such as database creation/upgrade steps. These cases will generally have partial program execution log output immediately before the broken pipe message visible. Should this be the case, Ansible and OpenSSH may need to have their configuration files tuned to meet the needs of the environment.

Consult the Ansible configuration file to see available connection settings ssh_args, timeout, and possibly pipelining. To see this file, enter:

```
      https://github.com/ansible/ansible/blob/release1.7.0/examples/
ansible.cfg
```

As Ansible uses OpenSSH, consult the `ssh_config` manual, in paricular the `ServerAliveInterval` and `ServerAliveCountMax` options.

Postfix fails to reload

Occasionally the postfix mail transfer agent will fail to reload because it is not running when the system expects it to be running.

Confirm that this is the case by examining the `/var/log/upstart/os-collect-config.log` for an indication that `service postfix reload` failed.

To fix this issue, start postfix by entering:

```
sudo service postfix start
```

Ephemeral certificates location

The default ephemeral certificate location is `/root`. This works for normal Helion installations. If you have specified another location, when you update Helion, you must specify the certificate location as `/root`. For example, the correct environment variable location and file name is:

```
  EPHEMERAL_CA_KEY_FILE=/root/eca.key
EPHEMERAL_CA_CERT_FILE=/root/eca.crt
```

Apache2 fails to start

Apache2 requires several self-signed SSL certificates to be properly configured but because of earlier failures in the setup process these certificaties may not have been configured correctly. If this is the case, you will see the following error message:

```
  * failed: [192.0.2.25] => (item=apache2) => {"failed": true, "item":
 "apache2"}
* msg: start: Job failed to start
```

You will also note the following symptoms:

- the Apache2 service fails to start
- the `/etc/ssl/certs/ssl-cert-snakeoil.pem` file is missing or empty.

To fix this problem, re-run `os-collect-config` to reassert the SSL certificates by entering:

```
sudo os-collect-config --force --one
```

RabbitMQ still running when restart is attempted

There are certain system states that cause RabbitMQ to ignore normal kill signals. In these cases, RabbitMQ continues to run. You will notice that you have this issue when your attempts to start `rabbitmq` fail because it is already running.

To fix this problem, find any processes running as `rabbitmq` on the server, and kill them, forcibly if need be.

Instance reported with status as "SHUTOFF" and task_state as "powering on"

When Nova Compute attempts to restart an instance when the Compute node is not ready, it is possible that Nova can enter a confused state where it thinks that an instance is starting when in fact the Compute node is doing nothing. You have reason to suspect that this is the case when:

- The command `nova list --all-tenants` reports instance(s) with STATUS of "SHUTOFF" and `task_state` is "powering on".
- The instance does not respond to pings.
- No instance appears to be running on the Compute node.
- Nova hangs when retrieving logs or returns old logs from the previous boot.
- A console session cannot be established.

To fix this problem:

- Log in to a controller as root and enter:

  `source stackrc`
- Execute `nova list --all-tenants` to obtain instance ID(s)
- Execute `nova show <instance-id>` on each suspected ID to identify suspected Compute nodes.
- Log into the suspected Compute node(s) and execute: `os-collect-config --force --one`
- Return to the controller node that you were logged into previously, and using the instancce IDs obtained previously, take the following steps:

  1. Execute `nova reset-state --active <instance-id>`
  2. Execute `nova stop <instance-id>`
  3. Execute `nova start <instance-id>`

1. Once the above steps have been taken in order, you should see the instance status return to ACTIVE and the instance become accessible via the network.

State drive `/mnt` is not mounted

In the rare event that an error occurred between the state drive being unmounted and the rebuild command being triggered, the `/mnt` volume on the instance upon which the rebuild command was executed will be in an unmounted state.

In this state, you will not be able to start MySQL and RabbitMQ. You will likely see these (pre-flight check) error messages:

```
  failed: [192.0.2.24] => {"changed": true, "cmd":
"rabbitmqctl -n rabbit@$(hostname) status" stderr: Error:
unable to connect to node
'rabbit@overcloud-controller0-vahypr34iy2x': nodedown
```

Attempts to start MySQL or RabbitMQ manually will fail and you will see:

```
  start: Job failed to start
```

Upgrade attempts return with an error indicating:

```
  TASK: [fail msg="Galera Replication, Node appears to be the last node in a
 cluster;  cannot safely proceed unless overridden via `single_controller`
 setting. See README.rst"]
```

If you run the `df` command, the return does not show a volume mounted as `/mnt`.

To fix this problem:

- Execute the `os-collect` config which will re-mount the state drive. This command may fail without additional intervention. However it should mount the state drive which is all that is needed to proceed to the next step. To run `os-collect-config`, enter:

```
sudo os-collect-config --force --one
```

- At this point, the `/mnt` volume should be visible in the output of the `df` command.
- Start MySQL by entering:

```
sudo /etc/init.d/mysqld start
```

- If MySQL fails to start, and it has been verified that MySQL is not running on any controller nodes, then you will need to identify the Last node that MySQL was stopped on and consult the section "MySQL fails to start upon retrying update" for guidance on restarting the cluster.
- Start RabbitMQ by entering:

```
service rabbitmq-server start
```

- If `rabbitmq-server` fails to start, then the cluster may be down. If this is the case, then the last node to be stopped will need to be identified and started before attempting to restart RabbitMQ on this node.
- At this point, re-execute the pre-flight check, and proceed.

VMs do not shut down properly during upgrade

During the upgrade process, VMs on Compute nodes are shut down gracefully. If the VMs do not shut down, this can cause the upgrade to stop. If this is the case, you will see a playbook run which ends with a message similar to:

```
  failed: [10.23.210.31] => {"failed": true}} msg: The ephemeral
storage of this system failed to be cleaned up properly and
processes or files are still in use. The previous ansible play
should have information to help troubleshoot this issue.
```

The output of the playbook run prior to this message contains a process listing and a listing of open files.

The state drive on the Compute node, `/mnt`, is still in use and cannot be unmounted. You can confirm this by entering:

```
lsof -n | grep /mnt
```

To see which VMs are running, enter:

```
virsh list
```

If `virsh list` fails, you may need to restart `libvirt-bin` or `libvirtd` depending on which process you are running. To restart, enter:

```
service libvirt-bin restart
```

or

```
service libvirtd restart
```

To fix this problem, you will have to intervene manually. You will need to determine why the VMs did not shut down properly, and resolve the issue.

You can forcely shutdown non-responsive VMs by entering:

```
virsh destroy <id>
```

Note that this can corrupt filesystems on the VM.

Resume the playbook run once the VMs have been shut down.

Instances are inaccessible via network

Upon restarting, it is possible that the virtual machine is unreachable due to Open vSwitch not being ready for the virtual machine networking. If this is the case, you will not be able to ping instances after a restart.

To fix this problem:

- Log into a controller node and execute:

  ```
  source /root/stackrc
  ```

- Stop all virtual machines on a Compute node by entering:

  ```
  nova hypervisor-servers <hostname>
  ```

  and

  ```
  nova stop <id>
  ```

- Log into the undercloud node and enter:

  ```
  source /root/stackrc
  ```

- Obtain a list of nodes by entering:

  ```
  nova list
  ```

- Execute `nova stop <id>` for the affected Compute node.
- Once the compute node has stopped, execute `nova start <id>` to reboot the Compute node.

Online upgrade fails with message saying glanceclient is not found

This problem occurs when you attempt to perform an online upgrade, However the playbook execution failed when you attempted to download the new image from Glance. You get a message that `glanceclient` was not found.

If you are attempting to execute the Ansible playbook on the seed host or undercloud node, source the Ansible virtual environment by entering:

```
source /opt/stack/venvs/ansible/bin/activate
```

Once the Ansible virtual environment has been sourced, on the node from which you are attempting to execute Ansible, enter:

```
sudo pip install python-glanceclient
```

Online upgrade of Compute node failed

In the event that an online upgrade of a Compute node fails, you can recover the node utilizing a traditional rebuild.

The problem occurs when you perform an online upgrade. The result of which is that a Compute node cannot be logged into, or is otherwise in a non-working state.

To fix this issue, from the undercloud enter:

```
source /root/stackrc
```

Identify the instance ID of the broken Compute node using the `nova list` command. Then stop the instance by entering:

```
nova stop <instance-id>
```

Return to the host from which you ran the upgrade and re-run the playbook without the `-e online_upgrade=True` option.

You can also utilize the `-e force_rebuild=True` option to force the instance to rebuild.

Ironic fails because nodes are in maintenance mode

During an update, nodes must NOT be in maintenance mode; otherwise Ironic returns an error message such as the following:

```
  During sync_power_state, max retries exceeded for node 0677d7e8-2e2b-4b12-
b426-4b2950d7a5f2, node state None does not match expected state 'power on'.
 Updating DB state to 'None' Switching node to maintenance mode.
```

If the command `ironic node-set-maintenance` fails to properly change nodes from maintenance mode, you must either update the database directly using:

```
mysql> update nodes set maintenance=0;
```

or enter:

```
ironic node-update <ironic node-id> replace maintenance=False.
```

Environment variables being ignored

When running the `update_sd.sh` script to upgrade from HP Helion OpenStack v1.1 to v1.1.1, the default 192.0.6.0/24 network is used instead of the networks that are defined by the `kvm-custom-ips.json` file. The reason this happens is the person doing the update neglected to set the ENV variables before running the update command. These ENV variables were set when the system was installed and the same ENV variable set (with the same values) must be used for the update.

The following is an example of ENV variables:

```
SEED_NTP_SERVER=10.34.56.78 BRIDGE_INTERFACE=eth2
BM_NETWORK_SEED_IP=10.34.55.66
BM_NETWORK_CIDR=10.34.55.0/24
BM_NETWORK_GATEWAY=10.34.55.1
SEED_NAMESERVER=172.16.222.5
./update_sd.sh /root/tripleo /root/helion-update-1.1-to-<version> | tee
 seed_update.log
```

Overcloud error message: ANSIBLE_HOST_KEY_CHECKING=False

If, during the update-overcloud operation, you see the following failure:

```
      Attempting to connect to each host
    + ANSIBLE_HOST_KEY_CHECKING=False
    + ansible -o -i /opt/stack/tripleo-ansible/plugins/inventory/heat.py -u
 heat-admin -m ping all
    Traceback (most recent call last):
    File "/opt/stack/venvs/ansible/bin/ansible", line 194, in <module>
        (runner, results) = cli.run(options, args)
    File "/opt/stack/venvs/ansible/bin/ansible", line 112, in run
        inventory_manager = inventory.Inventory(options.inventory,
 vault_password=vault_pass)
    File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/inventory/__init__.py", line 118, in __init__
        self.parser = InventoryScript(filename=host_list)
    File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/inventory/script.py", line 49, in __init__
        self.groups = self._parse(stderr)
    File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/inventory/script.py", line 57, in _parse
        self.raw  = utils.parse_json(self.data)
     File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/utils/__init__.py", line 552, in parse_json
        results = json.loads(data)
    File "/usr/lib/python2.7/json/__init__.py", line 338, in loads
```

```
        return _default_decoder.decode(s)
    File "/usr/lib/python2.7/json/decoder.py", line 366, in decode
        obj, end = self.raw_decode(s, idx=_w(s, 0).end())
    File "/usr/lib/python2.7/json/decoder.py", line 384, in raw_decode
        raise ValueError("No JSON object could be decoded")
    ValueError: No JSON object could be decoded
```

then, you can check the source of the issue by:

```
      source /opt/stack/venvs/ansible/bin/activate

    (ansible)root@hLinux:~/helion-update-1.1-to-<version>/tripleo/helion-
update/overcloud_update# ansible -vvv -o -i /opt/stack/tripleo-ansible/
plugins/inventory/heat.py -u heat-admin -m ping all
    Traceback (most recent call last):
    File "/opt/stack/venvs/ansible/bin/ansible", line 194, in <module>
        (runner, results) = cli.run(options, args)
    File "/opt/stack/venvs/ansible/bin/ansible", line 112, in run
        inventory_manager = inventory.Inventory(options.inventory,
 vault_password=vault_pass)
    File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/inventory/__init__.py", line 118, in __init__
        self.parser = InventoryScript(filename=host_list)
    File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/inventory/script.py", line 49, in __init__
        self.groups = self._parse(stderr)
    File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/inventory/script.py", line 57, in _parse
        self.raw  = utils.parse_json(self.data)
    File "/opt/stack/venvs/ansible/local/lib/python2.7/site-packages/
ansible/utils/__init__.py", line 552, in parse_json
        results = json.loads(data)
    File "/usr/lib/python2.7/json/__init__.py", line 338, in loads
        return _default_decoder.decode(s)
    File "/usr/lib/python2.7/json/decoder.py", line 366, in decode
        obj, end = self.raw_decode(s, idx=_w(s, 0).end())
    File "/usr/lib/python2.7/json/decoder.py", line 384, in raw_decode
        raise ValueError("No JSON object could be decoded")
    ValueError: No JSON object could be decoded
    (ansible)root@hLinux:~/helion-update-1.1-to-<version>/tripleo/helion-
update/overcloud_update# /opt/stack/tripleo-ansible/plugins/inventory/
heat.py --list
    overcloud-ce-vsastorage0/6dd10abb-482e-4eaf-b061-158efe5f8a15 stack is
 incomplete, in state FAILED
    (ansible)root@hLinux:~/helion-update-1.1-to-<version>/tripleo/helion-
update/overcloud_update#
```

In this case, the overcloud-ce-vsastorage0 node was in the FAILED state for Heat.

Return to Top