ERICSSON

Fuel Synchronization

Cloud Execution Environment

OPERATING INSTRUCTION

18/1543-CRA 119 1959/5 Uen B

Contents

1 Introduction

This document describes the manual Fuel synchronization procedure and the change to cold stand-by Fuel VM procedure. Automatic Fuel backup and restore are not supported.

A Fuel VM is active on one compute host and another Fuel VM is passive (shut off, ‘‘cold stand-by’’) on another compute host. The synchronization is performed by copying the active Fuel VM over the cold stand-by Fuel VM.

Fuel is an open source deployment and management tool for OpenStack. For more information refer to the *CEE Architecture Description*, Reference [1].

In this document the definition of terms are as follows:

Fuel	Open source component for OpenStack SW life cycle management, adding installation, upgrade, and equipment management support for a Cloud Execution Environment (CEE) instance.
Fuel VM	Ericsson component based on Fuel, running as a Virtual Machine (VM), providing the runtime environment for Fuel services.

1.1 Purpose

The purpose of this document is to provide manual instruction on usage and maintenance of the Fuel VM cold stand-by copy.

1.2 Target Groups

This document is aimed at skilled professionals from the following groups:

- Support organization personnel
- Customer O&M personnel

1.3 Prerequisites

This section states the prerequisites that have to be fulfilled.

It is mandatory to:

- Have an installed CEE system.
- Have Fuel VM migrated in the CEE Region.

1.3.1 Documents

No documents are required to perform the actions in this OPI.

1.3.2 Tools and Equipment

Ensure that the following tools are available:

- Computer with SSH connection to the virtual Cloud Infrastructure Controller (vCIC)

1.3.3 User Access

The Operator must have access to the deployment-specific credentials:

- A personal user with `sudo` access.

1.3.4 Configuration Data

Ensure that the following configuration data is available:

- IP address of CIC VM.
- IP address of Fuel VM.

  <fuel_address> is the Fuel static address in the `fuel_ctrl_sp` VLAN. The factory default value is `192.168.0.11`. Refer to the local version of the IP and VLAN plan, updated with site-specific IP addresses.

1.3.5 Conditions

Ensure that the following conditions are met:

- There are no errors in the Fuel application state. For detailed instructions, see Section 3.3 on page 5.
- There are no Fuel tasks in progress. For detailed instructions, see Section 3.4 on page 6.
- There are no nodes under provision. For detailed instructions, see Section 3.5 on page 7.

2 Tasks

Caution!

Changes in Fuel VM settings (that is, RAM, CPU, DISK, NETWORK) are not supported in these processes, and will be disregarded.

2.1 Synchronize the Active Fuel VM to the Cold Stand-by (Backup and Restore)

Reason

Synchronization is necessary in the following cases:

- Environment changes, for example:
  - Successful update
  - Expansion
  - Configuration changes
  - Repairing CEE
- A fault is observed on the compute node running Fuel VM actively.
- Fuel VM Operating System changes occurred that require back up.
- Synchronization of Fuel VMs is required for any other reason.

Process

1. Find the active and passive Fuel VMs, see Section 3.6 on page 7.
2. Shut down Fuel, see Section 3.9 on page 9.
3. Copy the Fuel VM image, see Section 3.10 on page 9.
4. Start the copied Fuel VM, see Section 3.11 on page 10.

2.2 Fail Over to the Cold Stand-by Fuel VM (Recover from Failure)

Reason

Fail over to the passive stand-by Fuel VM must be considered in the following cases:

- The Fuel VM is not operational.
- The compute that hosts the active Fuel VM is not operational or is under maintenance.

Process

1. Shut down the active Fuel VM, see Section 3.9 on page 9.
2. Start up the passive Fuel VM, see Section 3.13 on page 11.

3 Operations

This section describes the operations for the tasks detailed in Section 2 on page 3.

Caution!

Changes in Fuel VM settings (that is, RAM, CPU, DISK, NETWORK) are not supported in these operations, and will be disregarded.

3.1 Log in to one of the CICs

Do the following:

1. Log in to CIC:

   ```
   ssh <personal_user>@<cic_address>
   ```

3.2 Log in to Fuel

Prerequisites:

- Logged in to CIC, see Section 3.1 on page 5.

Do the following:

1. Log in to Fuel:

   ```
   ssh <personal_user>@<fuel_address>
   ```

3.3 Check Fuel Application Sanity

Prerequisites:

- Logged in to Fuel, see Section 3.2 on page 5.

Do the following:

1. Make sure that **`fuel-utils check_all`** command lists all services having `active` or `ready` state:

   ```
   fuel-utils check_all
   ```

Note: In case the compute hosting the Fuel VM is in an error state, the corresponding services show error. This case is an exception to the condition requirements.

Example output:

```
checking with command "systemctl is-active nailgun"
active
checking with command "! pgrep puppet"
nailgun is ready.
checking with command "egrep -q ^[2-4][0-9]? < <(curl --connect-timeout 1 -s -w '%{http_code}'
http://192.168.0.11:8777/ostf/not_found -o /dev/null)"
checking with command "! pgrep puppet"
ostf is ready.
checking with command "ps waux | grep -q 'cobblerd -F' && pgrep dnsmasq"
1764
checking with command "cobbler profile find --name=ubuntu* | grep -q ubuntu && cobbler profile
find --name=*bootstrap* | grep -q bootstrap"
checking with command "! pgrep puppet"
cobbler is ready.
checking with command "curl -f -L -i  -u "naily:RdHj4WxW6uJ5m6DdqrWzMu4Y" http://127.0.0.1:15672/api/nodes
1>/dev/null 2>&1"
checking with command "curl -f -L -u "mcollective:yli8lamU4IyMTpKZRYo19ViC" -s
http://127.0.0.1:15672/api/exchanges | grep -qw 'mcollective_broadcast'"
checking with command "curl -f -L -u "mcollective:yli8lamU4IyMTpKZRYo19ViC" -s
http://127.0.0.1:15672/api/exchanges | grep -qw 'mcollective_directed'"
checking with command "! pgrep puppet"
rabbitmq is ready.
checking with command "PGPASSWORD=jzKpSchKkgDcFeyT95UNFgN7 /usr/bin/psql -h 192.168.0.11 -U "nailgun"
"nailgun" -c '\copyright' 2>&1 1>/dev/null"
checking with command "! pgrep puppet"vFuel VM sync and changing to cold stand by vFuel
postgres is ready.
checking with command "ps waux | grep -q 'astuted'"
checking with command "curl -f -L -u "naily:RdHj4WxW6uJ5m6DdqrWzMu4Y" -s http://127.0.0.1:15672/api/exchanges
 | grep -qw 'nailgun'"
checking with command "curl -f -L -u "naily:RdHj4WxW6uJ5m6DdqrWzMu4Y" -s http://127.0.0.1:15672/api/exchanges
| grep -qw 'naily_service'"
checking with command "! pgrep puppet"
astute is ready.
checking with command "ps waux | grep -q mcollectived"
checking with command "! pgrep puppet"
mcollective is ready.
checking with command "ps waux | grep -q nginx"
checking with command "! pgrep puppet"
nginx is ready.
checking with command "keystone  --os-auth-url "http://192.168.0.11:35357/v2.0" --os-username "nailgun"
--os-password "yBvRPqbbY7VRovPUpXmtCpsg" token-get &>/dev/null"
checking with command "! pgrep puppet"
keystone is ready.
checking with command "netstat -nl | grep -q 514"
checking with command "! pgrep puppet"
rsyslog is ready.
checking with command "netstat -ntl | grep -q 873"
checking with command "! pgrep puppet"
rsync is ready.
```

3.4 Check Fuel Tasks in Progress

Prerequisites:

- Logged in to Fuel, see Section 3.2 on page 5.

Do the following:

1. Make sure that **`fuel task`** command lists all the tasks with status `ready`:

fuel task

If any of the tasks are still in progress, then wait until they become ready.

Example output:

```
id | status | name           | cluster | progress | uuid
---|--------|----------------|---------|----------|-------------------------------------
1  | ready  | check_networks | 1       | 100      | d6c0b037-e69b-485b-91e1-13cdc8dd1901
7  | ready  | deployment     | 1       | 100      | f240a2e0-86c7-4079-9aad-1c6d167431ec
2  | ready  | provision      | 1       | 100      | 7286c719-c578-4004-9c23-15f07151f266
6  | ready  | deploy         | 1       | 100      | 0797cd53-0a31-4d59-8b1c-ec32e9528fa5
4  | ready  | deployment     | 1       | 100      | 240f1d3c-77f5-437f-bb10-b0ca6d58b37c
3  | ready  | spawn_vms      | 1       | 100      | c9138575-ebf2-4c71-92b4-20268c150fdb
5  | ready  | provision      | 1       | 100      | dfb860a2-6b05-40de-a123-f4c6ba767e1f
```

3.5 Check Nodes Under Provision

Prerequisites:

- Logged in to Fuel, see Section 3.2 on page 5.

Do the following:

1. Make sure **fuel node** command lists all the nodes with status `ready`:

   fuel node

Example output:

```
id | status | name        | cluster | ip           | mac               | roles              | pending_roles | c
---|--------|-------------|---------|--------------|-------------------|--------------------|---------------|--
7  | ready  | cic-1       | 1       | 192.168.0.32 | 6a:df:69:05:25:4d | controller, mongo  |               | T
8  | ready  | cic-3       | 1       | 192.168.0.31 | 8e:f0:49:45:6a:43 | controller, mongo  |               | T
1  | ready  | cinder-0-5  | 1       | 192.168.0.24 | 90:55:ae:3a:05:f6 | cinder             |               | T
2  | ready  | compute-0-8 | 1       | 192.168.0.22 | 90:55:ae:3a:e5:76 | compute            |               | T
5  | ready  | compute-0-1 | 1       | 192.168.0.23 | 90:55:ae:39:f7:26 | compute, virt      |               | T
4  | ready  | compute-0-7 | 1       | 192.168.0.21 | 90:55:ae:3a:e3:ae | compute, virt      |               | T
6  | ready  | cic-2       | 1       | 192.168.0.30 | 92:f9:49:4c:d4:4f | controller, mongo  |               | T
3  | ready  | compute-0-6 | 1       | 192.168.0.20 | 90:55:ae:3a:e3:96 | compute, virt      |               | T
```

3.6 Find Active and Stand-by Fuel VMs

Do the following:

1. Find active and stand-by Fuel hosts, see Section 3.7 on page 7.
2. Store the IP addresses on the CIC as environment variables, see Section 3.8 on page 8.

3.7 Find Active and Stand-by Fuel Hosts

Prerequisites:

- Logged in to Fuel, see Section 3.2 on page 5.

Do the following:

1. Run the following script to get information about the computes:

```
for node in primary secondary ;
do
    ip=$(get_vfuel_info --ip --$node);
    name=$(ssh $ip hostname -s 2>&1 | grep compute);
    stat=$(ssh $ip sudo virsh list --all 2>&1 | grep fuel) stat=$(echo $stat | awk '{print $3 " " $4}');
    printf "%-10s | %s  |   %s\n" "$name" "$ip"  "$stat";
done
```

2. Optionally logout from Fuel to get back to CIC:

   logout

Example:

```
[ceeadm@fuel ~]$ for node in primary secondary ; do ⇒
ip=$(get_vfuel_info --ip --$node); name=$(ssh $ip ⇒
hostname -s 2>&1 | grep compute); stat=$(ssh $ip ⇒
sudo virsh list --all 2>&1 | grep fuel) stat=$(echo ⇒
$stat | awk '{print $3 " " $4}'); printf "%-10s | %s  ⇒
|   %s\n" "$name" "$ip"  "$stat"; done
compute-0-1 | 192.168.0.23  |   running
compute-0-3 | 192.168.0.21  |   shut off
[ceeadm@fuel ~]$ logout
```

Save this information to be able to find these compute hosts in case Fuel cannot be accessed.

3.8 Store IP addresses on the CIC as Environment Variables

Prerequisites:

- Logged in to CIC, see Section 3.1 on page 5.
- The addresses of the Fuel hosts are available, see Section 3.7 on page 7.

Do the following:

1. Export compute IP where Fuel VM is running:

   export ACTIVE_vFUEL_COMPUTE = *<compute_ip>*

2. Export compute IP where Fuel VM is shut down:

   export PASSIVE_vFUEL_COMPUTE = *<compute_ip>*

Example:

```
ceeadm@cic-1:~$ export ACTIVE_vFUEL_COMPUTE=192.168.0.23
ceeadm@cic-1:~$ export PASSIVE_vFUEL_COMPUTE=192.168.0.20
```

Note: The scope of these variables is limited by the current session.

3.9 Shut Down the Fuel VM

Prerequisites:

- Logged in to CIC, see Section 3.1 on page 5.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 8.

Do the following:

1. Log in to the compute that hosts the running Fuel VM.
2. Check that this compute hosts the running Fuel VM.
3. Shut down the running Fuel VM.
4. Check that the VM is shut off.

Example:

```
ceeadm@cic-1:~$ ssh $ACTIVE_vFUEL_COMPUTE

Attention! Unauthorized remote access is strictly prohibited!

Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.4.0-22-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
Last login: Mon Sep 12 19:55:42 2016 from 192.168.0.31
ceeadm@compute-0-1:~$ sudo virsh list --all
 Id    Name                           State
----------------------------------------------------
 2     cic-1_vm                       running
 5     fuel_master                    running

ceeadm@compute-0-1:~$ sudo virsh destroy fuel_master
Domain fuel_master destroyed

ceeadm@compute-0-1:~$ sudo virsh list --all
 Id    Name                           State
----------------------------------------------------
 2     cic-1_vm                       running
 -     fuel_master                    shut off

ceeadm@compute-0-1:~$
```

3.10 Copy the Fuel VM Image

Prerequisites:

- Logged in to CIC, see Section 3.1 on page 5.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 8.
- Fuel VM is shut off on both hosts.

Do the following:

1. Check that the Fuel VM is shut off on both hosts.
2. Check that the Fuel VM can be located on both hosts.
3. Set the authorized_keys file on the compute (this is needed only once).
4. Copy the Fuel VM image from the active compute host over the Fuel VM image in the passive compute host.

Example:

```
ceeadm@cic-1:~$ for i in $ACTIVE_vFUEL_COMPUTE $PASSIVE_vFUEL_COMPUTE; do echo -n $i; ssh $i ⇒
'sudo virsh list --all|grep fuel_master' 2>/dev/null; done
192.168.0.23 -     fuel_master                    shut off
192.168.0.21 -     fuel_master                    shut off

ceeadm@cic-1:~$ for i in $ACTIVE_vFUEL_COMPUTE $PASSIVE_vFUEL_COMPUTE; do echo -n "$i:"; ssh $i 'sudo find ⇒
/var/lib/nova -name fuel\*.qcow2' 2>/dev/null; done
192.168.0.23:/var/lib/nova/fuel_br3005.qcow2
192.168.0.21:/var/lib/nova/fuel_br3005.qcow2

ceeadm@cic-1:~$ for i in $ACTIVE_vFUEL_COMPUTE $PASSIVE_vFUEL_COMPUTE; do echo -n "$i: ";ssh $i "sudo su ⇒
-c 'cat /home/ceeadm/.ssh/authorized_keys >> /root/.ssh/authorized_keys';sudo tail -n 1 ⇒
/root/.ssh/authorized_keys"; done
192.168.0.23:
Attention! Unauthorized remote access is strictly prohibited!

ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAqTrf1g7rpxY2CHLZUxm3VitLD2UrP/n/zJRw1KpnGgGXnlmCyLL7tyHFjOcoSXZ8D2cXipAadV2lc
192.168.0.21:
Attention! Unauthorized remote access is strictly prohibited!

ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAqTrf1g7rpxY2CHLZUxm3VitLD2UrP/n/zJRw1KpnGgGXnlmCyLL7tyHFjOcoSXZ8D2cXipAadV2lc


ceeadm@cic-1:~$ scp -BC3 root@$ACTIVE_vFUEL_COMPUTE:/var/lib/nova/fuel_br3005.qcow2 ⇒
root@$PASSIVE_vFUEL_COMPUTE:/var/lib/nova/fuel_br3005.qcow2

Attention! Unauthorized remote access is strictly prohibited!


Attention! Unauthorized remote access is strictly prohibited!
```

3.11 Start the Copied Fuel VM

Prerequisites:

- Logged in to CIC, see Section 3.1 on page 5.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 8.
- Fuel VMs are shut off on both hosts, see Section 3.12 on page 11.

Do the following:

1. Start up the passive Fuel VM, see Section 3.13 on page 11.

3.12 Check Fuel States

Prerequisites:

- Logged in to CIC, see Section 3.1 on page 5.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 8.

Do the following:

1. Run the following script:

```
for i in $ACTIVE_vFUEL_COMPUTE $PASSIVE_vFUEL_COMPUTE;
do
    echo -n $i;
    ssh $i 'sudo virsh list --all|grep fuel_master' 2>/dev/null;
done
```

Example:

```
ceeadm@cic-1:~$ for i in $ACTIVE_vFUEL_COMPUTE ⇒
$PASSIVE_vFUEL_COMPUTE; do echo -n $i; ssh $i 'sudo virsh ⇒
list --all|grep fuel_master' 2>/dev/null; done
192.168.0.23 -     fuel_master                    shut off
192.168.0.21 -     fuel_master                    shut off
```

3.13 Start up the Fuel VM

Prerequisites:

- Logged in to CIC, see Section 3.1 on page 5.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 8.

Do the following:

1. Log in to the active or passive Fuel compute, as required:

   ssh ACTIVE_vFUEL_COMPUTE

   or

   ssh PASSIVE_vFUEL_COMPUTE

2. Start the Fuel VM:

```
sudo virsh start fuel_master
```

3. Check that the Fuel VM started up:

   ```
   sudo virsh list --all
   ```

   The output must be (within a reasonable time):

   ```
   fuel_master                running
   ```

4. After about 20 minutes Fuel VM should be running fully functional. Perform the sanity checks as required in Section 1.3.5 on page 2.

Reference List

[1] *CEE Architecture Description*, 4/155 53-AZE 102 01 Uen