

Fuel Synchronization

Cloud Execution Environment

OPERATING INSTRUCTION

Copyright

© Ericsson AB 2016-2018. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



Contents

1	Introduction	1
1.1	Purpose	1
1.2	Target Groups	1
1.3	Prerequisites	2
1.3.1	Documents	2
1.3.2	Tools and Equipment	2
1.3.3	User Access	2
1.3.4	Configuration Data	2
1.3.5	Conditions	2
2	Tasks	4
2.1	Synchronize the Active Fuel VM to the Cold Standby (Backup and Restore)	4
2.2	Failover to the Cold Standby Fuel VM (Recover from Failure)	5
2.3	Recreate Fuel VM on Repaired Host	5
3	Operations	6
3.1	Log on to one of the vCICs	6
3.2	Log on to Fuel	6
3.3	Check Fuel Application Sanity	6
3.4	Check Fuel Tasks in Progress	7
3.5	Check Nodes Under Provision	8
3.6	Find Active and Standby Fuel VMs	8
3.7	Find Active and Standby Fuel Hosts	9
3.8	Store IP addresses on the vCIC as Environment Variables	10
3.9	Shut Down the Fuel VM	11
3.10	Copy the Fuel VM Image	12
3.11	Start the Copied Fuel VM	13
3.12	Check Fuel States	13
3.13	Start up the Fuel VM	14
3.14	Create virsh Domain for Fuel VM on the Repaired Host	15





1 Introduction

This document describes the manual Fuel synchronization procedure and the change to cold standby Fuel VM procedure. Automatic Fuel backup and restore are not possible in CEE.

A Fuel VM is active on one compute host and another Fuel VM is passive (shut off, “cold standby”) on another compute host. The synchronization is performed by copying the active Fuel VM over the cold standby Fuel VM.

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.

Refer to the document [Backup and Restore Overview](#) for more information on Fuel synchronization, including the following:

- Procedure durations
- Recommended backup strategy

For an overview of all backup and restore options available in CEE, refer to the [Backup and Restore Overview](#).

1.1 Purpose

The purpose of this document is to provide manual instruction on usage and maintenance of the Fuel VM cold standby 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.

The following are mandatory for the operations described in the document:

- The CEE is installed.
- Fuel VM is migrated in the CEE Region.
- The compute host hosting vFuel is operational.

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 6.



- There are no Fuel tasks in progress. For detailed instructions, see Section 3.4 on page 7.
- There are no nodes under provision. For detailed instructions, see Section 3.5 on page 8.



2 Tasks

Caution!

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

2.1 Synchronize the Active Fuel VM to the Cold Standby (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 8.
2. Shut down Fuel, see Section 3.9 on page 11.
3. Copy the Fuel VM image, see Section 3.10 on page 12.
4. In case of failover, start the copied Fuel VM, see Section 3.11 on page 13.

Else, start the active Fuel VM, see Section 3.13 on page 14.



2.2 Failover to the Cold Standby Fuel VM (Recover from Failure)

Reason

Failover to the passive standby 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 11.
2. Start up the passive Fuel VM, see Section 3.13 on page 14.

2.3 Recreate Fuel VM on Repaired Host

Reason

If the compute host hosting the cold standby Fuel VM has failed, and has been repaired or replaced, the virsh domain must be defined again for the cold standby Fuel VM, and the Fuel VM image copied to the repaired or replaced host.

Process

1. Collect information on the Fuel VMs and the compute hosts hosting the Fuel VMs, see Section 3.7 on page 9.
2. Define the IP addresses of the hosts as environment variables on vCICs, see Section 3.8 on page 10
3. Create configuration dump XML file for the Fuel VM, copy it to the replaced host and define the virsh domain on the replaced host. See Section 3.14 on page 15
4. Shut down the Active Fuel VM, see Section 3.9 on page 11.
5. Copy the Fuel VM image to the replaced host, see Section 3.10 on page 12.
6. In case of failover, start the copied Fuel VM, see Section 3.11 on page 13.
Else, start the active Fuel VM, see Section 3.13 on page 14.



3 Operations

This section describes the operations for the tasks detailed in Section 2 on page 4.

Caution!

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

3.1 Log on to one of the vCICs

Do the following:

1. Log on to vCIC:

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

3.2 Log on to Fuel

Prerequisites:

- Logged on to vCIC, see Section 3.1 on page 6.

Do the following:

1. Log on to Fuel:

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

Note: Fuel commands require root access. If running the command as a personal user (for example, ceeadm), use `sudo`.

3.3 Check Fuel Application Sanity

Prerequisites:

- Logged on to Fuel, see Section 3.2 on page 6.

Do the following:

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

```
sudo 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 "yBvRPabbY7VRovPUpxmtCpsg" 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 on to Fuel, see Section 3.2 on page 6.

Do the following:

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



sudo 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 on to Fuel, see Section 3.2 on page 6.

Do the following:

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

sudo fuel node

Example output:

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

pending_roles	online	group_id
	True	1
	True	1
	True	1
	True	1
	True	1
	True	1
	True	1
	True	1

3.6 Find Active and Standby Fuel VMs

Do the following:

1. Find active and standby Fuel hosts, see Section 3.7 on page 9.



2. Store the IP addresses on the vCIC as environment variables, see Section 3.8 on page 10.

3.7 Find Active and Standby Fuel Hosts

Prerequisites:

— Logged on to Fuel, see Section 3.2 on page 6.

Note: In case vFuel is not reachable, run the following script on any vCIC to get the name of the compute where vFuel is hosted:

```
sudo python <<EOF
import yaml
with open('/etc/astute.yaml', 'r') as f:
    file = yaml.load(f)
    for node in file["ericsson"]["nodes"]:
        reserved_cpus = file["ericsson"]⇒
["nodes"][node]["reserved_cpus"]
        if reserved_cpus.has_key('vfuel'):
            print node
EOF
```

An example of the printout is the following:

```
compute-0-1
compute-0-2
```

To check if vFuel is active on a compute host, execute the command **virsh list --all** on the compute host. If it is active, it is listed in the printout with State: `running`.

Do the following:

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

Note: Record the printout as it can be required in a later step or procedure.

```
for node in primary secondary ;
do
    ip=$(sudo get_vfuel_info --ip --$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 " " $4}'); stat2=$⇒
    (if [ "$stat" == "running " ]⇒
    ; then echo "ACTIVE_vFUEL_COMPUTE=$ip"; ⇒
    else echo "PASSIVE_vFUEL_COMPUTE=$ip";fi);
    printf "%-10s | %s | %s | %s | %s\n" "$name" "$ip" "$node"⇒
    "$stat" "$stat2";
done
```

An example of the printout is the following:



```
compute-0-1 | 192.168.0.23 | primary | running | ACTIVE_vFUEL_COMPUTE=192.168.0.23
compute-0-3 | 192.168.0.21 | secondary | shut off | PASSIVE_vFUEL_COMPUTE=192.168.0.21
```

The printout has the following layout:

Variable	Description	Values
<compute_id>	ID of the compute host hosting the Fuel VM	
<compute_ip>	IP address of the compute host hosting the Fuel VM	
<node_type>	Node type of the Fuel VM	primary secondary
<vm_state>	State of the Fuel VM	running - the VM is the active Fuel VM shut off - the VM is the cold standby Fuel VM no state - the host was repaired or replaced, the Fuel VM must be recreated. See Section 3.14 on page 15.
<environment_variable>=<ip_address>	This value pair can be used for defining the environment variables in Section 3.8 on page 10.	ACTIVE_vFUEL_COMPUTE=<compute_ip> PASSIVE_vFUEL_COMPUTE=<compute_ip>

2. Optionally, log out from Fuel to get back to the vCIC:

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 vCIC as Environment Variables

Prerequisites:

- Logged on to vCIC, see Section 3.1 on page 6.
- The addresses of the Fuel hosts are available, see Section 3.7 on page 9.

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 or not defined:

```
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 on to vCIC, see Section 3.1 on page 6.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 10.

Do the following:

1. Log on to the compute that hosts the running Fuel VM.
2. Check that this compute hosts the running Fuel VM.

Note: If the command is issued for the first time for any compute host hosting a Fuel VM, the following printout can appear:

```
The authenticity of host '<host_ip> (<host_ip>)' can't be
established.
ECDSA key fingerprint is <fingerprint>.
Are you sure you want to continue connecting (yes/no)?
Continue by typing yes to establish connection with the host.
```

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 on to vCIC, see Section 3.1 on page 6.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 10.
- Fuel VM is shut off on both hosts.

Do the following:

1. Check that the Fuel VM is shut off on both hosts.

Note: If the command is issued for the first time for any compute host hosting a Fuel VM, the following printout can appear:
The authenticity of host '<host_ip> (<host_ip>)' can't be established.
ECDSA key fingerprint is <fingerprint>.
Are you sure you want to continue connecting (yes/no)?
Continue by typing **yes** to establish connection with the host.

2. Check that the Fuel VM can be located on both hosts.

Note: If this procedure is performed as part of recreating the Fuel VM on a replaced host, as described in Section 2.3 on page 5, the Fuel VM image is not present on the replaced host.

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.

Note: The procedure takes approximately 20 minutes.



Example:

```

ceeadm@cic-1:~$ for i in $ACTIVE_vFUEL_COMPUTE $PASSIVE_vFUEL_COMP
UTE; 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 $
/var/lib/nova -name fuel\*.qcow2' 2>/dev/null; done
192.168.0.23:/var/lib/nova/fuel_master.qcow2
192.168.0.21:/var/lib/nova/fuel_master.qcow2

ceeadm@cic-1:~$ for i in $ACTIVE_vFUEL_COMPUTE $PASSIVE_vFUEL_COMPUTE; do echo -n "$i: ";ssh $
-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/zJRw1KpnGgGXnlmCyLL7tyHFj0coSXZ8D2cXip=>
AadV2lc5qaALPE+qghymwRlUk6UelxtkjRbrC57j0xgkkl/vWiNp7LwXkLk8qA24gTFYX4tTPltQnG3p4Gfu83cXJU4sPr0Z/7RVUHK4XZJ=>
+L03vne6FLG0WECKlrHC6ayp0ekTSjRyyy5omH0vZopxKTdFBkIwtko02b/1TxA4K/X13SpxQ9Phy8hq0U8UZiPvJad4z1j6h4KbVs6mbcn=>
mAkxXX1YA3QL5SuCSzZ6129QN3MWG5LluIDhi2WLVnrPwlCLPxHCgNCqBw== ansible-generated on fuel.domain.tld
192.168.0.21:
Attention! Unauthorized remote access is strictly prohibited!

ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEAqTrf1g7rpxY2CHLZUxm3VitLD2UrP/n/zJRw1KpnGgGXnlmCyLL7tyHFj0coSXZ8D2cXip=>
AadV2lc5qaALPE+qghymwRlUk6UelxtkjRbrC57j0xgkkl/vWiNp7LwXkLk8qA24gTFYX4tTPltQnG3p4Gfu83cXJU4sPr0Z/7RVUHK4XZJ=>
+L03vne6FLG0WECKlrHC6ayp0ekTSjRyyy5omH0vZopxKTdFBkIwtko02b/1TxA4K/X13SpxQ9Phy8hq0U8UZiPvJad4z1j6h4KbVs6mbcn=>
mAkxXX1YA3QL5SuCSzZ6129QN3MWG5LluIDhi2WLVnrPwlCLPxHCgNCqBw== ansible-generated on fuel.domain.tld

ceeadm@cic-1:~$ scp -BC3 root@$ACTIVE_vFUEL_COMPUTE:/var/lib/nova/fuel_master.qcow2 =>
root@$PASSIVE_vFUEL_COMPUTE:/var/lib/nova/fuel_master.qcow2

Attention! Unauthorized remote access is strictly prohibited!

Attention! Unauthorized remote access is strictly prohibited!

```

3.11 Start the Copied Fuel VM

Prerequisites:

- Logged on to vCIC, see Section 3.1 on page 6.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 10.
- Fuel VMs are shut off on both hosts, see Section 3.12 on page 13.

Do the following:

1. Start up the passive Fuel VM, see Section 3.13 on page 14.

3.12 Check Fuel States

Prerequisites:

- Logged on to vCIC, see Section 3.1 on page 6.



- Fuel host addresses are defined as environment variables, see Section 3.8 on page 10.

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
```

Note: If the command is issued for the first time for any compute host hosting a Fuel VM, the following printout can appear:

```
The authenticity of host '<host_ip> (<host_ip>)' can't be
established.
ECDSA key fingerprint is <fingerprint>.
Are you sure you want to continue connecting (yes/no)?
Continue by typing yes to establish connection with the host.
```

3.13 Start up the Fuel VM

Prerequisites:

- Logged on to vCIC, see Section 3.1 on page 6.
- Fuel host addresses are defined as environment variables, see Section 3.8 on page 10.

Do the following:

1. Log on to the Fuel compute:

```
ssh $ACTIVE_vFUEL_COMPUTE
```

Note: If the cold standby vFuel VM must be started, for example, due to failover, use the environment variable `$PASSIVE_vFUEL_COMPUTE` instead of `$ACTIVE_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 5 minutes, the Fuel VM should be running fully functional. Perform the sanity checks as required in Section 1.3.5 on page 2.

To find the hosts of the active and the standby Fuel VM, refer to Section 3.7 on page 9.

5. Log on to the active Fuel compute:

```
ssh $ACTIVE_vFUEL_COMPUTE
```

6. Set Fuel VM to autostart:

```
sudo virsh autostart fuel_master
```

7. Log on to the passive Fuel compute:

```
ssh $PASSIVE_vFUEL_COMPUTE
```

8. Disable autostarting the old Fuel VM:

Note: If the compute host hosting the old Fuel VM is not operational, perform this step after the recovery of the compute host.

```
sudo virsh autostart --disable fuel_master
```

3.14 Create virsh Domain for Fuel VM on the Repaired Host

Note: In this procedure, variables refer to the information collected in Section 3.7 on page 9.

Prerequisites:

- Logged on to Fuel, see Section 3.2 on page 6.
- Printouts are available from the procedure described in Section 3.7 on page 9.
- IP addresses of the compute hosts hosting the Fuel VMs are defined as environment variables, see Section 3.8 on page 10.

Do the following:

1. In Fuel, create a configuration dump for the active Fuel VM, and pipe it to an XML file in the /tmp folder by executing the following command:

```
ssh ceeadm@<compute_ip_address> 'sudo virsh dumpxml
fuel_master' =>
> /tmp/fuel_master.xml
```



,where `<compute_ip_address>` corresponds to the IP address of the host running the active Fuel VM, as listed in the printout in Section 3.7 on page 9.

2. Modify the configuration dump XML file by executing the following script:

```
modify_vfuel_xml --nodetype <replaced_node_node_type>
```

where `<replaced_node_node_type>` is the node type listed for the replaced VM in the printout of Section 3.7 on page 9, and has one of the following values:

- `primary`, if the node type of the replaced host is primary,
- `secondary`, if the node type of the replaced node is secondary.

3. Copy the modified XML file to the replaced host by executing the following command:

```
scp /tmp/fuel_master.xml ceeadm@<replaced_host_ip_address>:/tmp/fuel_master.xml
```

,where `<replaced_host_ip_address>` is the IP address of the replaced host in the printout of Section 3.7 on page 9

An example of the printout is the following:

```
[ceeadm@fuel ~]$ scp /tmp/fuel_master.xml ceeadm@192.168.0.21:/tmp/fuel_master.xml
Attention! Unauthorized remote access is strictly prohibited!
fuel_master.xml                                100% 3101      3.0KB/s   00:00
```

4. From vFuel, log on to the replaced compute host using SSH by executing the following command:

```
ssh ceeadm@<replaced_compute_ip_address>
```

,where `<replaced_compute_ip_address>` corresponds to the IP address of the replaced host in the printout in Section 3.7 on page 9.

5. Define the virsh domain for the vFuel VM on the replaced host by executing the following command:

```
sudo virsh define /tmp/fuel_master.xml
```

An example of the printout is the following:

```
Domain fuel_master defined from /tmp/fuel_master.xml
```

6. Remove the ssh key for the removed host by executing the following command on all vCICs:

```
ssh-keygen -f "/home/ceeadm/.ssh/known_hosts" -R <replaced_compute_ip_address>
```

where `<replaced_compute_ip_address>` corresponds to the IP address of the replaced host in the printout in Section 3.7 on page 9.

An example of the printout is the following:

```
ceeadm@cic-1:~$ ssh-keygen -f "/home/ceeadm/.ssh/known_hosts" -R 192.168.0.21
```



```
# Host 192.168.0.21 found: line 2 type ECDSA
/home/ceedm/.ssh/known_hosts updated.
Original contents retained as /home/ceedm/.ssh/known_hosts.old
```