

# OpenStack Compute API in CEE

Cloud Execution Environment

INTERWORK DESCRIPTION

**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

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	API Version	1
1.2	Document References	1
1.2.1	API Design Base Reference	1
<b>2</b>	<b>Supported Operations</b>	<b>2</b>
2.1	Basic OpenStack Operations	2
2.2	OpenStack Extensions	2
<b>3</b>	<b>Ericsson Extended Functions</b>	<b>3</b>
3.1	forcemove	3
3.1.1	API Operation	3
3.1.2	VM Migration and Evacuation	4
3.1.3	Scheduling	5
3.1.4	Hints for VM Affinity and Antiaffinity	5
3.2	redefine	7
3.2.1	API Operation	8
3.3	PCI Passthrough and SR-IOV Physical Function Passthrough Use in OpenStack	8
3.3.1	Nova Flavor Creation for PCI Passthrough Ports and SR-IOV Physical Function Passthrough Ports	9
3.4	Bandwidth Based Scheduling	9
3.4.1	API Operation	10
3.4.2	VMs with and without Bandwidth Requirements	11
<b>4</b>	<b>Limitations</b>	<b>13</b>
4.1	Limitations for Multi-Server Deployment	16
4.2	Limitations for Single Server Deployment	18
<b>5</b>	<b>Additional Information</b>	<b>20</b>
5.1	Host Aggregates in CEE	20
5.1.1	Removal of Host from Host Aggregate	20
5.2	Libvirt Real Time Instances	20





# 1 Introduction

This document serves as an introduction to the use of the Application Programming Interface (API) of the OpenStack component “compute” in the Cloud Execution Environment (CEE).

While the main aim of the document is to present the compute API in CEE, it also contains descriptive information about the features of CEE compute.

**Note:** Unless otherwise indicated, values in this document are given in KB, MB, and GB, according to JESD100B.01. For more information, refer to Units of measurement in the document [Glossary of Terms and Acronyms](#).

## 1.1 API Version

By default, the external Virtualized Network Function Manager (VNFM) and Network Function Virtualization Orchestrator (NFVO) use the CEE compute API microversion based on the OpenStack compute API v2.1.

**Note:** OpenStack compute API v2.0 is supported but deprecated in this release and is to be removed in the next major release.

## 1.2 Document References

This section contains the official OpenStack API reference.

### 1.2.1 API Design Base Reference

For the description of the API operations, refer to the [OpenStack Compute API](#) and section “Compute” in the [OpenStack Administrator Guide](#).

These are stored copies of the OpenStack document versions that were the base for the development of this version of CEE.



## 2 Supported Operations

The following sections contain the API operations and API extensions that are supported in CEE.

### 2.1 Basic OpenStack Operations

For the detailed description of basic compute API operations, refer to the [OpenStack Compute API](#).

CEE compute supports basic compute API operations, with the limitations listed in Section 4 on page 13.

### 2.2 OpenStack Extensions

Not applicable for OpenStack compute API v2.1.



## 3 Ericsson Extended Functions

This section presents the extended API functions that are specific to the CEE.

### 3.1 forcemove

`forcemove` is an API function that supports the migration and evacuation of Virtual Machines (VMs) from a compute host.

`forcemove` uses Nova rescheduling, honoring `same_host`, `different_host`, `server_groups` affinity filters, and High Availability (HA) policy.

**Note:** The HA policy are configured in the `metadata` field of the VM.

#### 3.1.1 API Operation

In CEE, the standard OpenStack API is extended with a call for `forcemove`:

**POST** `v2/<tenant_id>/servers/<server_id>/action`

Request body:

```
{
  "forcemove": {
    "ignore_hints": false,
    "ignore_broken_dependencies": false,
    "block_migrate": false,
    "disk_over_commit": false
  }
}
```

**Note:** It is required to pass all four `forcemove` parameters, even though, in most cases, these are false (OpenStack default).

If the request succeeds, the response body is the following:

```
{
  "needs_start": false
}
```

In case of an error, the response body is the following:

```
{
  "badRequest": {
    "message": "No valid host was found.",
    "code": 507
  }
}
```



```
}
```

**Note:** The message and the code can vary.

### 3.1.2 VM Migration and Evacuation

CEE allows the configuration of HA policy for each VM. This policy is honored by the `forcemove` function.

CEE 15B policies VM Migration and VM Evacuation are removed from CEE 6 and replaced by HA-policies. For more information, see Section 3.1.2.1 on page 4.

#### 3.1.2.1 HA Policy

The user can define the level of HA needed on a specific VM. Three levels of HA can be achieved by defining a HA Policy on the VM.

The possible configuration values for `ha-policy` are the following:

<b>unmanaged</b>	“Unmanaged” means that CEE does not try to manage the VM after it was started. No action is performed by CEE on this VM. (This is the default, if no HA-policy is provided.)
<b>managed-on-host</b>	“Managed on host” means that the VM starts up with the host and shuts down with it. In case of failure, the VM is not moved to another host, but it is restarted when the node is restarted. On <code>forcemove</code> , the VM is shut down.
<b>ha-offline</b>	“High Availability with offline migration” means that the VM is evacuated in case of failure, moved to another host on <code>forcemove</code> .

**Note:** `ha-policy` is case-sensitive.

#### 3.1.2.2 Policy Configuration

During the creation of a VM, configure policies in the metadata field in the following way:





```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

Here, the migration policy is set to managed-on-host.

### 3.1.3 Scheduling

The default OpenStack migration call contains a parameter for the destination compute host to where the VM is moved.

For evacuation, <host> is optional:

```
root@cic-1:~# nova evacuate
```

Syntax:

```
nova evacuate [--password <password>] =>
[--on-shared-storage] <server> [<host>]
```

In CEE, a VM can be moved simply from one compute host to another compute host in the cluster, using compute rescheduling to choose the destination compute host.

This process is supported by the forcemove operation.

### 3.1.4 Hints for VM Affinity and Antiaffinity

In order to have high availability and redundancy, the application layer needs to guarantee that some VMs are on different compute hosts.

To achieve this, OpenStack provides a feature called “scheduler hints”. For example, if vm1 is booted, and, after that, vm2 is booted using **--hint different\_host=<vm1\_uuid>**, then OpenStack guarantees that vm2 will be on a different node than vm1.

The forcemove functionality of CEE adds a patch to OpenStack to save hints used during boot.



#### 3.1.4.1 Supported Hints

In CEE, `server_group` hint is supported and recommended, refer to the section “Server groups (os-server-groups)” in the [OpenStack Compute API](#).

The hints `same_host` and `different_host` are also supported, but deprecated.

#### 3.1.4.2 Prerequisites

In order to use scheduler hints, `nova.conf` must add them through `scheduler_default_filters`.

**Note:** This prerequisite is automatically met during installation and does not require further action.

#### 3.1.4.3 Recommended Setup

The following scheduling filters are configured with CEE:

- In multi-server deployments: `RetryFilter`, `AvailabilityZoneFilter`, `RamFilter`, `CoreFilter`, `DiskFilter`, `ComputeFilter`, `ComputeCapabilitiesFilter`, `ImagePropertiesFilter`, `AggregateInstanceExtraSpecsFilter`, `SameHostFilter`, `DifferentHostFilter`, `ServerGroupAntiAffinityFilter`, `ServerGroupAffinityFilter`, `PciPassthroughFilter`, `NUMATopologyFilter`, `AggregateMultiTenancyIsolation`.
- In single server deployments: `RamFilter`, `CoreFilter`, `DiskFilter`, `ComputeFilter`, `ImagePropertiesFilter`.

#### 3.1.4.4 Hint Configuration

During the boot of a VM, configure scheduler hints in the following way:

```
1 #
2 #
3 #
4 #
5 #
6 #
7 #
8 #
9 #
10 #
11 #
12 #
13 #
14 #
15 #
16 #
17 #
18 #
19 #
20 #
21 #
22 #
23 #
24 #
25 #
26 #
27 #
28 #
29 #
30 #
31 #
32 #
33 #
34 #
35 #
36 #
37 #
38 #
39 #
40 #
41 #
42 #
43 #
44 #
45 #
46 #
47 #
48 #
49 #
50 #
51 #
52 #
53 #
54 #
55 #
56 #
57 #
58 #
59 #
60 #
61 #
62 #
63 #
64 #
65 #
66 #
67 #
68 #
69 #
70 #
71 #
72 #
73 #
74 #
75 #
76 #
77 #
78 #
79 #
80 #
81 #
82 #
83 #
84 #
85 #
86 #
87 #
88 #
89 #
90 #
91 #
92 #
93 #
94 #
95 #
96 #
97 #
98 #
99 #
100 #
```



### 3.1.4.5 Broken Dependencies

`forcemove` always tries to consider the saved scheduler hints, but, in certain cases, it cannot follow the saved hints.

```
$ nova boot vm1 --image ... --flavor ... # this vm got id 8088e8b6-fd1a-4bf2-bff5-e2debb668a3a
$ nova boot vm2 --image ... --flavor ... --hint same_host=8088e8b6-fd1a-4bf2-bff5-e2debb668a3a
$ nova delete vm1
$ nova forcemove vm2
```

Server UUID	Move accepted	Error Message	Needs Start
59961962-fc06-4d90-82e4-366aaa3a9b38	False	No valid host was found.	False

#### Example 1 Broken Dependencies 1

Here, `forcemove` tries to use the saved hint, but cannot succeed, since `vm1` has been deleted.

However, deleted VMs can be ignored during the handling of hints:

```
$ nova forcemove vm2 --ignore-broken-dependencies
```

Server UUID	Move accepted	Error Message	Needs Start
59961962-fc06-4d90-82e4-366aaa3a9b38	True		False

#### Example 2 Broken Dependencies 2

Here, the hint `same_host=8088e8b6-fd1a-4bf2-bff5-e2debb668a3a` is ignored.

**Note:** If the `server_groups` affinity filter is used, hints can be ignored, but `forcemove` does not check the deleted VMs, so broken dependencies have no effect on `forcemove`.

### 3.1.4.6 Ignore Hints

`forcemove` can be configured to ignore all hints:

```
$ nova forcemove vm1 --ignore-hints
```

Server UUID	Move accepted	Error Message	Needs Start
59961962-fc06-4d90-82e4-366aaa3a9b38	True		False

#### Example 3 Ignore Hints

## 3.2 redefine

`redefine` is an API function that defines the properties of VMs on a compute host from the Nova database. It is used when the libvirt XML or the file system image is missing.



`nova reconfigure` creates a new libvirt XML based on the previously stored content in the Nova database and defines the instance for libvirt.

If the file system image is missing (for example, in case of board replacement), `nova reconfigure` downloads the base image from Glance.

The functionality checks if the following criteria apply:

- The `nova compute` service is in state up.
- The status of the VM is ACTIVE or SHUTOFF.
- The VM is not defined on the virtual layer.

If the above criteria are not met, `reconfigure` raises an exception.

**Note:** If the VM was in ACTIVE state before the `reconfigure` operation, it starts automatically once the operation is completed. If the VM was in SHUTOFF state, it has to be started manually.

### 3.2.1 API Operation

In CEE, the standard OpenStack API is extended with a call for `reconfigure`:

**POST** `v2.1/<tenant_id>/servers/<server_id>/action`

Request body:

```
{
  "reconfigure": {}
}
```

An example response body for a successful request is:

```
{"adminPass": "x8C3iYRUUnJ86"}
```

**Note:** The message and the code can vary.

## 3.3 PCI Passthrough and SR-IOV Physical Function Passthrough Use in OpenStack

This section describes Nova flavor creation for PCI passthrough and SR-IOV Physical Function passthrough ports and VM boot using PCI passthrough or SR-IOV Physical Function passthrough devices.

**Note:** Interface configuration and driver installation on the tenant VM is not performed by CEE and is the responsibility of the operator.



### 3.3.1 Nova Flavor Creation for PCI Passthrough Ports and SR-IOV Physical Function Passthrough Ports

VMs can be provisioned with PCI passthrough ports using the Nova extra spec `pci_passthrough:alias` with the aliases defined before deployment in the configuration file, or using the default aliases. The default aliases are the following:

- `PCI_devs` devices can be used for VMs for the PCI passthrough feature (SR-IOV virtualization mode is set to off on the NIC) in which the `device_type` is type-PCI.
- `PF_devs` devices can be used for VMs for the SR-IOV Physical Function passthrough feature (SR-IOV virtualization mode is set to on on the NIC), in which the `device_type` is type-PF.

For more information about PCI passthrough alias configuration, refer to the [Configuration File Guide](#).

An example of a flavor configuration to pass two predefined PCI passthrough devices called `my_pci_dev` through the configuration file to the VM:

```
openstack flavor create --ram 4096 --disk 100 =>
--vcpus 2 m1.medium.pci_passthrough
openstack flavor set --property "pci_passthrough:alias"="my_pci_dev:2" m1.medium.pci_passthrough
nova flavor-key m1.medium.pci_passthrough set =>
hw:mem_page_size=1048576
nova flavor-key m1.medium.pci_passthrough set =>
hw:cpu_policy=dedicated
```

An example of a flavor configuration to pass two default SR-IOV Physical Function passthrough devices called `PF_devs` file to the VM:

```
openstack flavor create --ram 4096 --disk 100 =>
--vcpus 2 m1.medium.sr-iov_pf_pt
openstack flavor set --property "pci_passthrough:alias"="PF_devs:2" m1.sr-iov_pf_pt
nova flavor-key m1.sr-iov_pf_pt set =>
hw:mem_page_size=1048576
nova flavor-key m1.sr-iov_pf_pt set =>
hw:cpu_policy=dedicated
```

## 3.4 Bandwidth Based Scheduling

Bandwidth based scheduling is an extension that enables VM scheduling based on free network bandwidth. The user requests the required bandwidth using attributes in flavor. Nova scheduler uses these attributes when scheduling VMs with that specific flavor. Nova keeps track of the reserved bandwidth on each network interface controller (NIC) on every host. Nova ensures that no



NIC is overprovisioned. Both bit rate and packet rate capacity are taken into consideration.

### 3.4.1 API Operation

This extension extends the `extra_specs` attribute in flavor. Two new attributes are added to `extra_specs`:

- `bandwidth:vif_inbound_average`
- `bandwidth:vif_outbound_average`

**Note:** The unit of measurement for the requested bandwidth is kbps, kilobytes per second,  $10^3$  bytes per second.

The values for the new attributes are in JSON format and contain requested bandwidth and average packet size. Rate is requested bandwidth. Size is average packet size in bytes of the VM frames on this interface. The size is used together with the byte rate to calculate the requested frames per second.

#### Format:

```
'{ "<device name of sriov hwnic>" : [<minimum-bandwidth-vf1>, =>
<minimum-bandwidth-vf2>, ...],
"<device name of neutron network hwnic>" : { "rate": =>
[<rate-vnic1>, <rate-vnic2>, ...], "size": =>
[<avg-packet-size-vnic1> , <avg-packet-size-vnic2>, ...]} ... }'
```

In `config.yaml`, the SR-IOV section contains the `physical_network` parameter, which is used in the bandwidth based scheduling extension as an SR-IOV interface name. For `physical_network` configuration, refer to the “SR-IOV” section of the [Configuration File Guide](#).

An example of the configuration:

```
---
# SR-IOV configuration
sriov:
  physical_network: br-nic
  # ... other configuration ...
```

#### Example 4 SR-IOV Configuration



```
nova flavor-key bw-sriov-flavor set
bandwidth:vif_inbound_average='{ "physnet0":[20000, 10000], "physnet1":[10000], "default":{"rate":⇒
[ 10000, 20000, 30000 ], "size": [512, 1024, 1024]}}' bandwidth:vif_outbound_average='{ "physnet0":⇒
[20000, 10000], "physnet1":[10000], "default":{"rate":[ 10000, 20000, 30000], "size": [512, 1024, 1024] }}'
```

#### Example 5 bandwidth:vif\_\*bound\_average Attribute

The `pci_passthrough:alias` is not necessary.

The vNICs specified in the flavor have to match the vNICs specified on “nova boot”. To use this feature, the user has to specify bandwidth on all NICs on the VM, including SR-IOV interfaces. For example, it is not allowed to specify one bandwidth in flavor and then boot the VM with two vNICs. It is accepted to not use the extension by not specifying any “bandwidth:\*” attributes in flavor.

### 3.4.2 VMs with and without Bandwidth Requirements

This section provides information about handling VMs with and without bandwidth requirements in the same CEE region.

VMs with unspecified bandwidth cannot be scheduled on the same hosts as VMs with specified bandwidth. If VMs with unspecified bandwidth consume all bandwidth, fragmentation issues can occur. In CEE, the default `ram_weight_multiplier` is set to 1, meaning the scheduler spreads VMs on all available hosts. This can result in all hosts having VMs without specified bandwidth, which makes it impossible to schedule VMs with specified bandwidth. This worst-case scenario is shown in Figure 1.

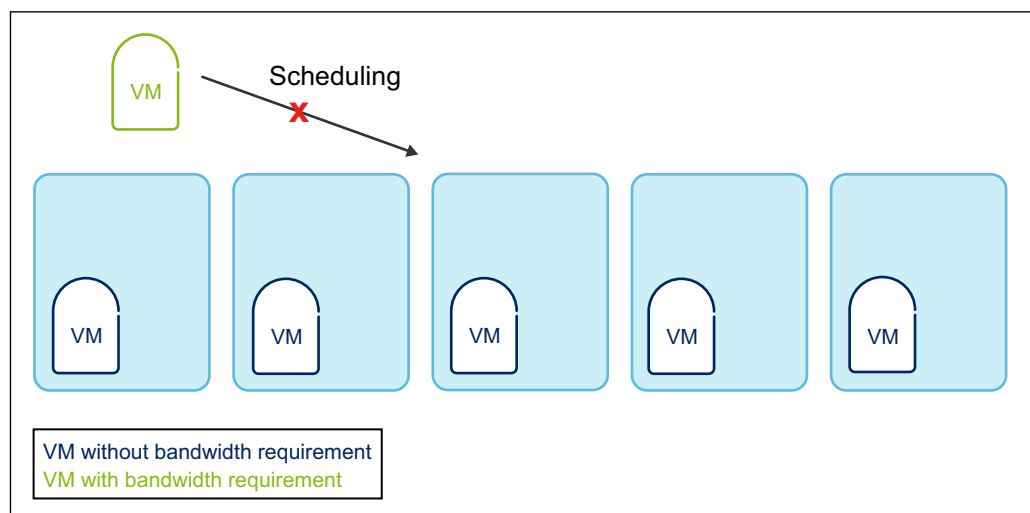


Figure 1 Bandwidth Based Scheduling Fails

To avoid this issue, use **host aggregates** to divide the compute hosts into two groups, as shown in Figure 2: VMs with bandwidth requirements and VMs without bandwidth requirements. When a flavor is created, the host aggregate must be set accordingly.

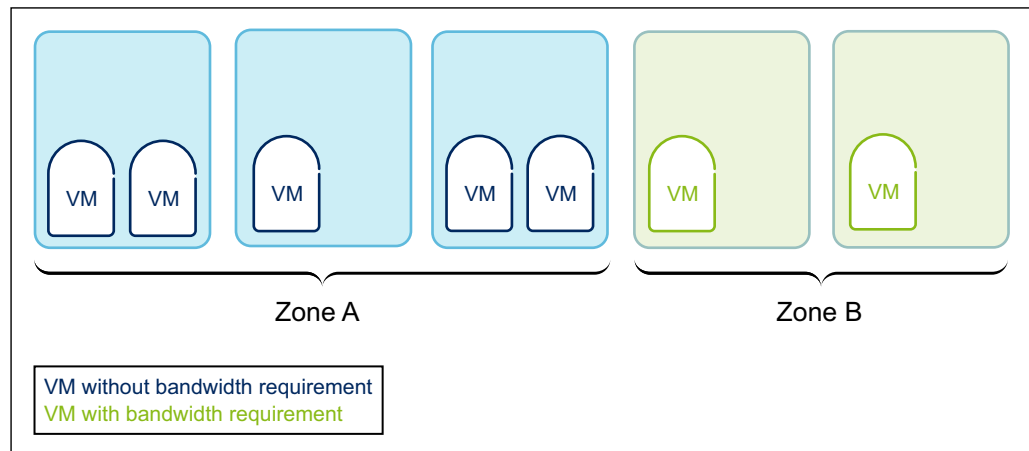


Figure 2 Using Host Aggregates to Manage Bandwidth Requirements

For the use of host aggregates in CEE, see Section 5.1 on page 20.





## 4 Limitations

**Note:** In addition to the limitations listed in this section, also refer to Section 4.1 on page 15 for limitations specific to CEE in multi-server deployment and Section 4.2 on page 18 for limitations specific to CEE in single server deployment.

The following limitations exist in CEE compute:

- The following operations are not supported by 3PP:
  - `nova pause`
  - `nova unpause`
  - `nova shelve`
  - `nova unshelve`
- `nova suspend` and `nova resume` are only supported in systems using Cloud SDN Switch (CSS) version 4.
- `nova rebuild` is not supported by 3PP for volume booted VMs.
- `metadata service` is not possible to use.
- As the maximum body size for HTTP requests is set to 112 KB in the default settings of the Nova API, larger environment files cannot be injected, even if the `injected_file_content_bytes` quota value is changed for the specific tenant.
- To refer to another VM within a hint, only the ID of the VM can be used instead of the name of the VM.
 

```
nova boot ... --hint same_host=cbd9fa13-f51e-4ab0-b2⇒
06-31526a236bc9
```
- Only VM offline migration is supported. Live migration is not possible in this release.
- Connecting VMs to additional networks on the fly with `nova interface-attach` is not supported by 3PP.
- The following default flavors are configured during installation:

ID	Name	Memory_MB	Disk	Ephemeral Swap	VCPUs	RXTX_Factor	Is_Publics
2	m1.small	2048	20	0	1	1.0	True
3	m1.medium	4096	40	0	2	1.0	True



ID	Name	Memory_MB	Disk	Ephemeral Swap	VCPUs	RXTX_Factor	Is_Publics
4	m1.large	8192	80	0	4	1.0	True
5	m1.xlarge	16384	160	0	8	1.0	True

- Only flavors with the below constraints are supported:

**Note:** Nova API does not reject any unused or unimplemented keys. It can result in a command getting executed but with no visible impact or failed instantiation of the VM.

- RAM allocated in steps of 1024 MB, matching the size of the hugepages used by the instances (1024, 2048, 3072 MB and so on)

The following `extra_specs` need to be defined when creating the flavor:

- `hw:mem_page_size=1048576`

This enables the hugepage backing of the instances by 1 GB hugepages.

- Creation of flavors with a disk value of 0 is supported, but for VMs booted with these flavors, the scheduler cannot select the compute host based on the virtual image. Therefore, flavors with a disk value of 0 are recommended only to be created and used for volume booted instances and testing purposes.
- A VM booted with a flavor created with NUMA topology cannot be resized with another flavor with different NUMA topology. The new NUMA topology is not applied to the resized VM. In multi-server deployment, this is for example when `hw:numa_nodes` is used in the extra specs of the flavor.
- The flavor key `hw:watchdog_action` is supported with the following values:
  - `disabled`
  - `poweroff`
  - `reset`
  - `none`

**Note:** This limitation also applies to setting the `hw_watchdog_action` key as a Glance image property.

Example:

`hw:watchdog_action=poweroff`

- Nova supports two ways to pass user data to the guest VM during boot: `file injection` and `user-data`. The files injected with the `--file` flag of the `nova boot` command are not kept injected after the VM is evacuated. However, the data passed with the `--user-data` flag of the `nova boot` command are kept during the evacuation. Therefore using the `--file` flag is discouraged. Use the `--user-data` flag instead to preserve user data.



- Live snapshot is supported. If the VM does not support live snapshots (for example, qemu-agent is not installed in the VM), the VM needs to be stopped before creating a snapshot, by following these steps:
  - 1 Stop the VM you want to create a snapshot from:
 

```
nova stop <vm_id>
```
  - 2 Create the snapshot:
 

```
nova image-create <vm_id> <snapshot_name>
```
  - 3 Restart the VM:
 

```
nova start <vm_id>
```
- The host-describe and hypervisor-show commands of Nova are not reliable regarding memory usage. The used memory does not consider different page sizes and does not consider the reservations for vCIC and vFuel. For proper values check the /proc/meminfo file on the compute host. This limitation is only about the shown values, the actual reservation is taken into account during scheduling.
- For the admin only API call --availability-zone no scheduling and resource checking happens, if the host name is used after the availability zone.
 

Using the call in the format --availability-zone nova:<host\_name>.domain.tld forces Nova to start the VM on the specified compute, regardless of the available resources. If you remove the compute name from the command, Nova selects a proper host with sufficient resources. If no such host exists, your request fails.
- If a VM is removed while nova-compute is not running, the attached volumes of the VM are not detached and removed. There are no errors indicating that the volumes are not removed.
 

To remove the attached volumes, execute the following commands:

```
cinder reset-state --state available $<volume_uuid>
cinder reset-state --attach-status detached $<volume_uuid>
cinder delete $<volume_uuid>
```
- Nova in CEE supports about 500 parallel stop/detach root volume operations.
- From the complete list of extra\_specs described in the [OpenStack Administrator Guide](#), only the extra\_specs described in this document are supported (with respect to the deployment type). In case an extra\_specs parameter is needed that is not described in this document, contact the next level of support.
- If after a failed VM evacuation the VM is manually recovered on a new host and the VM also recovers on the original host, the VM instance on the original host must be removed manually. In this case, contact the next level of support.



## 4.1 Limitations for Multi-Server Deployment

In addition to the limitations described in Section 4 on page 13, the following limitations apply to CEE in multi-server deployment:

— Only flavors with the below constraints are supported:

- An even number of CPUs
- The following `extra_specs` must be defined when creating the flavor:

`hw:cpu_policy=dedicated`

This setting ensures that the instances get dedicated CPU cores from the host.

— The `hw:numa_nodes` flavor key can be used in the following supported ways:

- Not defining the `hw:numa_nodes` key at all. In this case, the value 1 is used by default.  
**Note:** There is no visible impact between `hw:numa_nodes=1` and not defining the key at all.
- Setting `hw:numa_nodes=<n>`, where `<n>` equals an integer value of two or greater, with the following limitations:

**Note:** This setting forces symmetric splitting of vCPU and RAM resource allocations across all NUMA nodes. Therefore, the restrictions listed below are in place to prevent incorrect VM placement or failure to instantiate a VM.

- Supported only on flavors where the requested number of vCPU resources is a multiple of 4 (4, 8, 12, 16, 20, and so on).
- Supported only on flavors where the requested amount of RAM resources is a multiple of 2048 MB (2048, 4096, 6144, 8192, 10240, and so on).
- Supported only with hardware architecture where multiple NUMA nodes exist (for example: HDS, Dell physical servers).
- Nova can only provide vCPUs from a certain predefined number of NUMA nodes. The exact number must be given. Requesting vCPUs from any number of NUMA nodes is not supported by 3PP.

For example, `hw:numa_nodes=<1>` is used on physical servers with a single NUMA node, such as on BSP/GEP).

— The below flavor keys are not available or supported:

- `hw_video` and `hw_rng` is not available in the current release of CEE.
- Instance resource quota is not supported by 3PP.



- VMs booted with `same_host` hint can only have managed-on-host or unmanaged HA-policy. For such VMs, the ha-offline policy is not possible.

The limitation also applies to chains of VMs with `same_host` hints, including the VM at the beginning of the chain booted without the `same_host` hint.

For VMs booted with unsupported configuration, `forcemove` can fail.

Nova and the CM-HA automatic recovery function cannot check if a VM with ha-offline policy is using `same_host` hint, therefore the CM-HA automatic recovery function can fail.

- OpenStack services use RabbitMQ in a cluster configuration. RabbitMQ clustering guarantees consistency and partition tolerance, but does not guarantee availability. For more information, refer to the RabbitMQ documentation.

When the vCIC deployed with master RabbitMQ server is rebooted, the recovery of RabbitMQ can take up to 1.5 minutes. During the recovery of RabbitMQ, the Nova API and Neutron agents are unavailable on all vCICs, and the nova compute service and Neutron agents are unavailable on all compute hosts.

When a vCIC deployed with slave RabbitMQ server is rebooted, during the recovery of RabbitMQ, the Nova API and Neutron agents are unavailable on the respective vCIC, and the nova compute service is unavailable on some of the compute hosts.

The recovery time depends on the message queue size.

**Note:** The Nova API outage does not affect tenant VMs.

- CM-HA operates in active-passive mode. If a compute host containing a vCIC that runs the active CM-HA restarts, the VM evacuation does not start within one minute. The evacuation only starts when the CM-HA process is moved to another vCIC by Corosync, and the compute host unavailability is detected by the CM-HA.
- If multiple VMs are booted from volume in parallel, some of the VMs might be provisioned unsuccessfully because the block device mapping is invalid. This issue can occur if the storage back end does not respond within the configured timeframe when initializing the connection with the compute host, resulting in a timeout. Increasing the following timeout parameter gives the Cinder back end more time to respond:
  - `cinder_backend_response_timeout`
- VMs assigned with PCI devices from different NUMA nodes do not boot up because of a Nova limitation (I/O based NUMA scheduling spec) present since the OpenStack Juno release. Nova only boots instances with PCI devices on the NUMA nodes that the PCI devices are associated with.



For example, in an environment with two NUMA nodes and one PCI device (for example, an SR-IOV card associated with the first NUMA node), it is only possible to boot an instance with one NUMA node and SR-IOV ports on the first NUMA node. As a result, the CPUs and RAM on the second NUMA node cannot be used.

- During the parallel reboot of three vCICs, RabbitMQ Mnesia core dumps might be generated, because it is the expected behavior after an abrupt shutdown. It is possible that the data is corrupted on all nodes, and the startup order cannot be determined.

As Pacemaker can restore the RabbitMQ cluster independently of the shutdown and startup sequence, the core dumps do not have a system impact, and can be ignored.

## 4.2 Limitations for Single Server Deployment

In addition to the limitations described in Section 4 on page 13, the following limitations apply to CEE in single server deployments:

Single server installation requires special flavor metadata (`extra_specs`) settings.

- `hw:cpu_list` specifies the number of physical CPUs regarding the physical NUMA node, on which the virtual CPUs are pinned.

For example, the string `node1:{9,15}/node1:{11}/node0:{10,16}/node1:{3,9}` means that there are 4 vCPUs in the flavor, which are selected by the next sequence:

- The first vCPU is selected from NUMA node 1, physical CPUs 9 or 15.
- The second vCPU is selected from NUMA node 1, physical CPU 11.
- The third vCPU is selected from NUMA node 0, physical CPUs 10 or 16.
- The fourth vCPU is selected from NUMA node 1, physical CPUs 3 or 9.

**Note:** Assigning syntax does not support negation (^). For example, `{2-9,^8}` is not a valid definition.

Use `lscpu` on the designated compute node to get the NUMA topology available for pinning.

**Note:** Make sure to allocate the physical CPUs enlisted with owner `vm` in the `config.yaml` file as pinned vCPUs for the VM. Physical CPUs enlisted with owner `host` can be used to allocate floating vCPUs for the VM.

- `hw:numa_memory` allows defining and assigning memory of host NUMA nodes to VMs.



For example, the string `node0:1048576/node1:1048576` means that 1 GB memory from host NUMA node 1 and 1 GB memory from second host NUMA node is used. (In this, case altogether 2 GB is needed for the flavor.)



## 5 Additional Information

### 5.1 Host Aggregates in CEE

Follow the below procedure to use OpenStack host aggregates:

1. To configure the scheduler to support host aggregates, add filter `AggregateInstanceExtraSpecsFilter` to the `scheduler_default_filters` configuration options in `/etc/nova/nova.conf`.
2. Create the host aggregate in the nova availability zone and add the required compute nodes:

```
openstack aggregate create --zone nova =>  
<host_aggregate_name>
```

```
openstack aggregate add host <host_aggregate_name> =>  
<host_name>
```

```
openstack aggregate set --property <key=value> =>  
<aggregate>
```

3. Map the host aggregate metadata to the flavor:

```
openstack flavor set =>  
--property aggregate_instance_extra_specs:=  
<key=value> <flavor>
```

4. Start an instance using the flavor:

```
nova boot --flavor <flavor> (--image <image> =>  
| --volume <volume>) <server_name>
```

#### 5.1.1 Removal of Host from Host Aggregate

**Note:** Do not remove a host from an aggregate if there are any running instances on the host, as it causes a mismatch between the Nova database record and the `nova show` output.

To remove a host from the aggregate, use the following command:

```
openstack aggregate remove host <host_aggregate_name> =>  
<host_name>
```

### 5.2 Libvirt Real Time Instances

The CPU pinning feature adds the ability to assign guest virtual CPUs to dedicated host CPUs, providing guarantees for CPU time, and improving the worst-case





latency for CPU scheduling. The addition of the real time scheduling policy further improves the worst-case scheduler latency for vCPUs.

**Note:** Enabling the real time feature compromises the overall throughput of the system. As such, it is only to be enabled in case the guest workload demands it.

To enable the real time feature for an instance:

1. Map a flavor with `cpu_realtime` specifications:

```
openstack flavor set --property ⇒  
hw:cpu_policy=dedicated <flavor_name>
```

```
openstack flavor set --property ⇒  
hw:mem_page_size=1048576 <flavor_name>
```

```
openstack flavor set --property ⇒  
hw:cpu_realtime=yes <flavor_name>
```

```
openstack flavor set --property ⇒  
hw:cpu_realtime_mask=^1 <flavor_name>
```

**Note:** `hw:cpu_realtime_mask=^1` indicates that vCPU1 remains non real time, while all other vCPUs have a real time policy. The vCPU(s) with non real time policy will run the emulator thread(s).

2. Boot an instance using the flavor:

```
nova boot --flavor <flavor_name> --image <image_id> ⇒  
--nic net-id=<network_id> <server_name>
```