Deploying SAP HANA on Red Hat Virtualization

A guide to deploying SAP HANA on Red Hat Virtualization 4.2 and 4.3 with Intel Xeon Scalable Platform 1. and 2. Generation CPUs (Skylake and Cascade Lake).

Abstract: This guide contains information about SAP HANA hardware requirements and best practices. It includes examples of SAP HANA and RHV-specific configuration settings and deployment options to consider when using the two products together.

Date: July 2021
Version: 3.0

Contents

Red Hat

1. Introduction

Use this guide to deploy SAP HANA as a supported workload on Red Hat Virtualization (RHV) with up to 6 TB single and multiple virtual machines (VMs) on Intel Xeon Scalable Platform 1st and 2nd generation CPUs (1st generation "Skylake": up to 3TB guests supported, 2nd generation "Cascade Lake": up to 6TB guests). [SAP Note 2599726](#) provides information about the support statement and certified hardware and software. RHV 4.2 and RHV 4.3 are supported by SAP HANA as they both use hypervisor version `qemu-kvm-rhev-2.12.0-18.el7_6.5` or newer.

Note that as of now there are two storage variants explicitly tested and validated for the usage with SAP HANA: Fibre Channel with HBA Passthrough or NFS storage, which also supports live migration.

Note: You must have an SAP account to access SAP Notes.

2. Hardware requirements

Red Hat tested SAP HANA on RHV for deployment on SAP Certified and Supported SAP HANA Hardware platforms. Before deploying SAP HANA for use with RHV, check the current list of certified platforms to verify that the planned deployment is supported according to SAP Notes. For more information, refer to SAP Note 2399995.

3. Configuring the BIOS settings of the RHV hosts

These are the required BIOS settings of the RHV hosts:

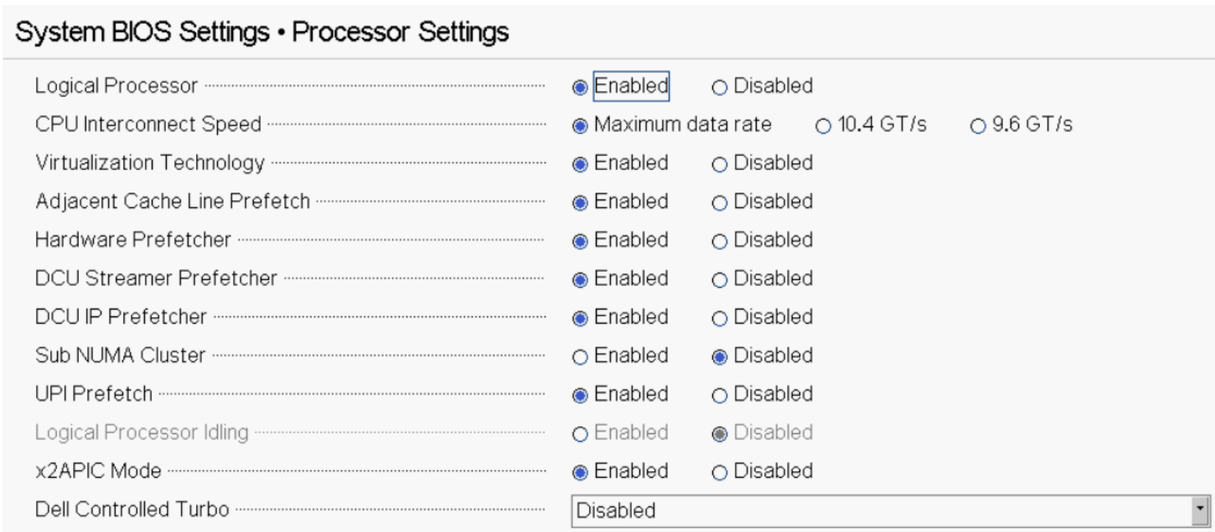
1. Enable **x2APIC**
2. If available, select the **Maximum Performance** profile
3. Enable **Turbo**
4. Enable **Hyperthreading**
5. Set **C-States** to OS control
6. Enable Virtualization Technologies such as VT-d and SR-IOV for NIC

Contact your system vendor for detailed instructions on how to configure your system for SAP

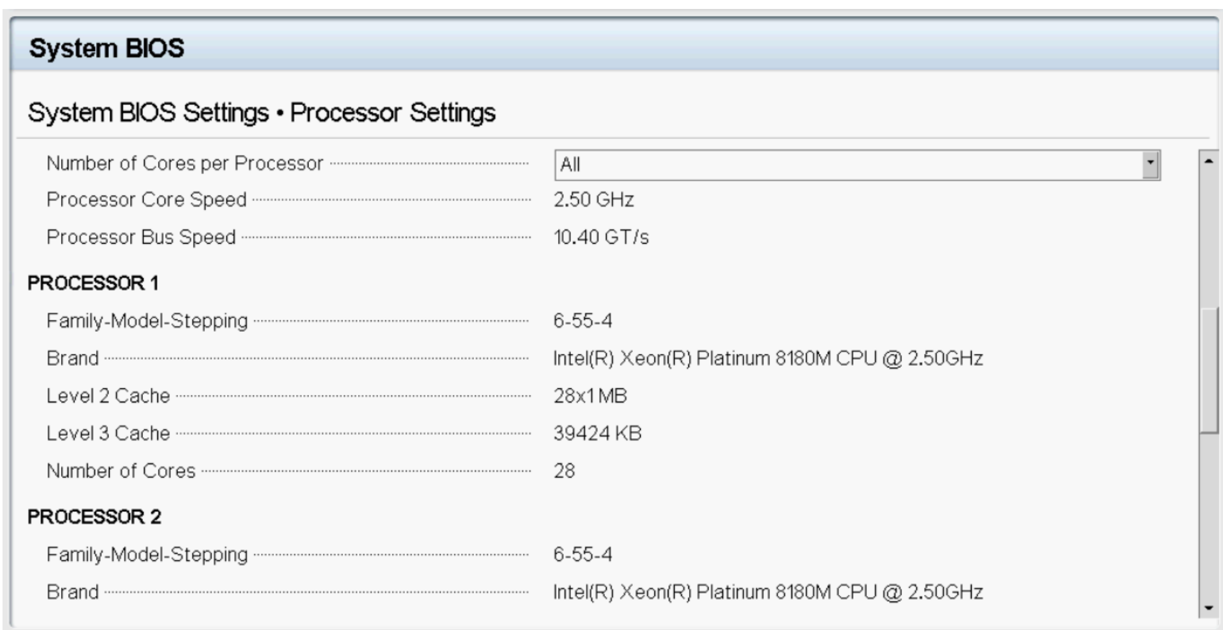
HANA. The following procedure uses a Dell R940 server as an example:

Procedure

1. In the host BIOS settings, navigate to **System BIOS Settings** -> **Processor Settings**.
2. Set **x2APIC Mode** to **Enabled** as shown in the following figure.

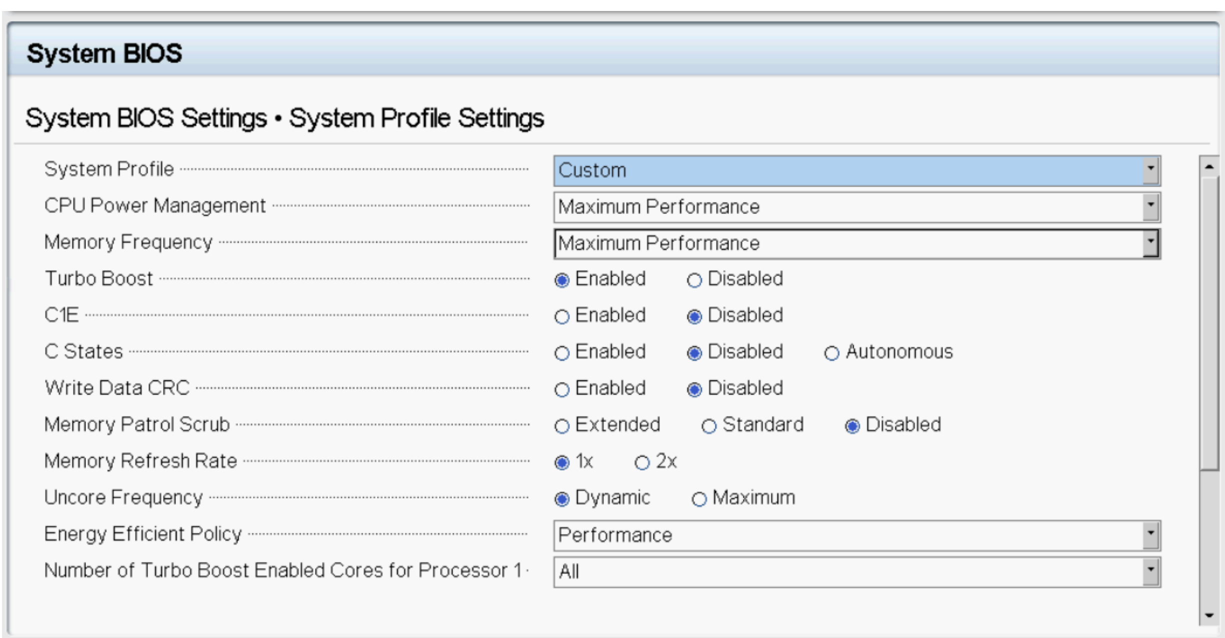


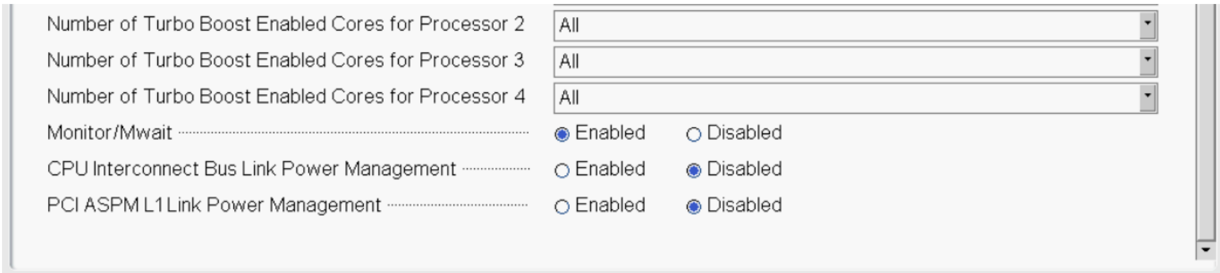
3. Set the **Number of Cores per Processor** to **All** as shown in the following figure.



3. Navigate to **System BIOS Settings** -> **System Profile Settings** and configure the settings as shown below and in the following figure:

- **System Profile:** Custom
- **CPU Power Management:** Maximum Performance
- **Memory Frequency:** Maximum Performance
- **Turbo Boost:** Enabled
- **C1E:** Disabled
- **C States:** Disabled
- **Write Data CRC:** Disabled
- **Uncore Frequency:** Dynamic
- **Energy Efficient Policy:** Performance
- **Number of Turbo Boost Enabled Cores for Processor 1:** All
- **Number of Turbo Boost Enabled Cores for Processor 2:** All
- **Number of Turbo Boost Enabled Cores for Processor 3:** All
- **Number of Turbo Boost Enabled Cores for Processor 4:** All
- **Monitor/Mwait:** Enabled
- **CPU Interconnect Bus Link Power Management:** Disabled
- **PCI ASPM L1 Link Power Management:** Disabled





4. Installing and configuring RHV hosts

This section describes how to install and configure RHV hosts.

Verifying system requirements for the RHV host

Refer to the "Host System Requirements" section of the Virtualization and Deployment Administration Guide for the dedicated disk space and RAM specifications needed to run the RHV host. The RHV 4.2 host includes the `qemu-kvm-rhev` package version 2.12. See the "Installing Hosts" section of the Red Hat Virtualization Installation Guide for assistance about installing an RHV host.

Installing the RHV host

To install the RHV environment, refer to the following RHV product documentation:

- Planning and Prerequisites Guide
- Self-Hosted Engine Guide
- Installation Guide
- Administration Guide

Updating the kernel

The minimal kernel version supported is `kernel-3.10.0-957.36.1.el7`. It is recommended that you use the latest kernel that is available through Red Hat update channels.

Setting the kernel boot options

For optimal network performance, Red Hat recommends using SR-IOV, which requires specific IOMMU settings for the kernel. Ensure that IOMMU functionality is enabled in the server BIOS. If you are unsure about how to enable IOMMU in the BIOS, contact your hardware vendor for support.

Red Hat also recommends using static 1 GB hugepages for the SAP-HANA VM as static hugepages reduce TLB misses and speed up virtual machine (VM) memory management, which is essential for SAP HANA performance. In addition, it is recommended that CPU power management states are disabled to improve overall CPU performance.

Procedure

To ensure the correct amount of static hugepages are reserved during startup, complete the steps:

1. Calculate the number of hugepages required based on the amount of memory required for the SAP HANA VM. You must be able to evenly divide the number of 1 GB hugepages by the number of sockets or NUMA nodes on the system you are using for the guest. For example, to run a 128 GB SAP HANA VM, you must configure at least 128 1 GB static hugepages. If you have four sockets or NUMA nodes on your RHV host, each virtual NUMA node for the guest would have 32 GB of memory.

2. Add the following parameters to the RHV host kernel command line, adjusting the values as required for your configuration:

   ```
   default_hugepagesz=1GB
   hugepagesz=1GB
   hugepages=[# hugepages]
   ```

4. To enable IOMMU, add the following parameter to the RHV host kernel command line:

   ```
   intel_iommu=on iommu=pt
   ```

5. Add the parameters to the kernel tab during host deploy.

   Warning: If added or changed later, the RHV host needs to be redeployed. See "Adding a Host to the Red Hat Virtualization Manager" of the Red Hat Virtualization Installation Guide.

Red Hat

Deploying a new RHV host

To deploy a new RHV host, complete the following steps in RHV Manager.

Procedure

1. Click **Hosts.**
2. Click **New.**
3. Navigate to the **Kernel** tab.
4. Add the parameters, separated by spaces, to the **kernel-command-line**. For example:

   ```
   iommu=pt intel_iommu=on default_hugepagesz=1GB hugepagesz=1GB
   hugepages=128
   ```

   If using a Skylake CPU, also add: `spectre_v2=retpoline`
5. When the deployment is completed, select **Management** > **Maintenance** to put the host into maintenance.
6. Once in maintenance, reboot the RHV host by selecting **Management > SSH Management > Restart.**
7. When the reboot is completed, activate the RHV host by selecting **Management > Activate.**

Changing an existing RHV host

To change an existing RHV host, complete the following steps in RHV Manager.

Procedure

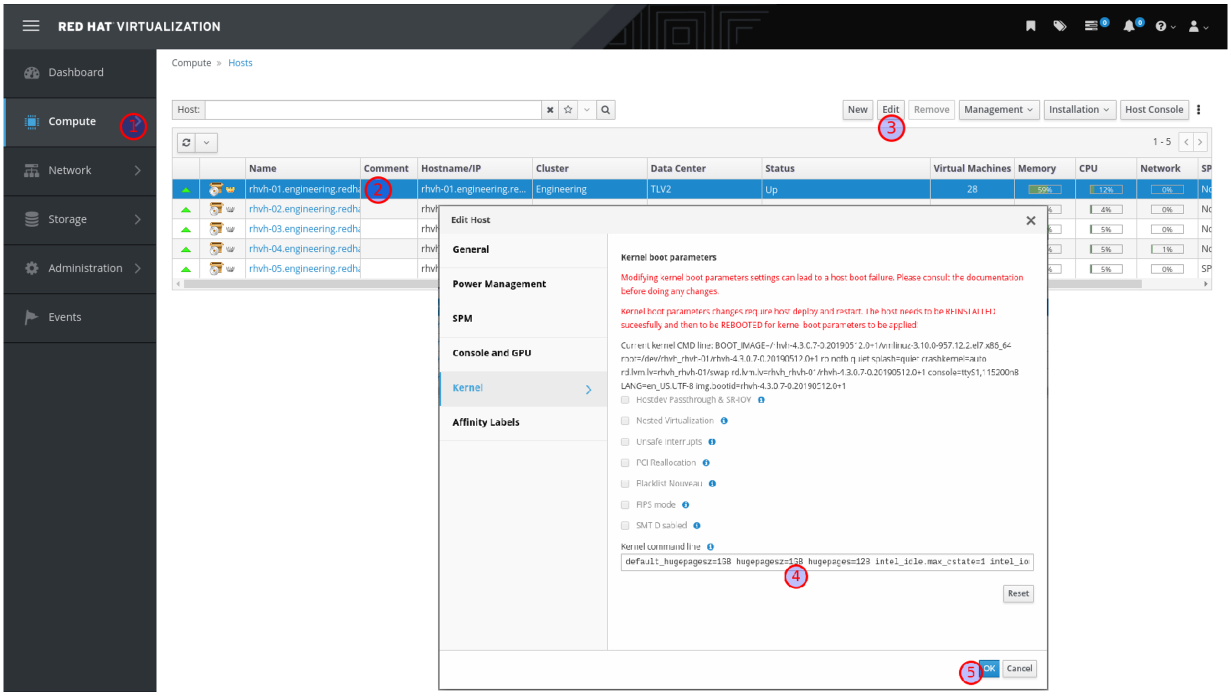
1. Click **Compute** > **Hosts**.
2. Select the relevant RHV host.
3. Click **Edit**.
4. In the **Edit Host** window, click **Kernel**.
5. Add the parameters, separated by spaces, to the kernel command line. For example:

   ```
   iommu=pt intel_iommu=on default_hugepagesz=1GB hugepagesz=1GB
   hugepages=128
   ```

If using a Skylake CPU, also add: `spectre_v2=retpoline`

6. Click **OK.**

   See the example in the following figure:



7. Select **Management > Maintenance** to put the RHV host into maintenance.

8. Once in maintenance, select **Installation > Reinstall** to apply the new kernel parameters to the RHV host.

Note: As the RHV 4.2 and 4.3 hosts are based on RHEL 7, refer to the [RHEL 7 Kernel Administration Guide] for more information on kernel command-line parameters. Verify that the parameters have been correctly applied by checking the current kernel command line on the RHV host:

```
# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-3.10.0-693.11.1.el7.x86_64
root=/dev/mapper/rhel_rhvh01-root ro crashkernel=auto
rd.lvm.lv=rhel_rhvh01/root rd.lvm.lv=rhel_rhvh01/swap rhgb quiet
LANG=en_US.UTF-8 iommu=pt intel_iommu=on default_hugepagesz=1GB hugepagesz=1GB
hugepages=128
```

Setting the tuned profile for RHV host

Complete the following steps in this procedure to set the tuned profile for a RHV host. The commands and files described in this section are available in a zip file at https://access.redhat.com/articles/4448131.

Procedure

1. Create `/usr/lib/tuned/sap-hana-kvm-host/tuned.conf:`

```
# tuned configuration

[main]
summary=Optimize for running as KVM host for SAP HANA as virtual guest
include=virtual-host

[cpu]
force_latency=cstate.id:1|2
```

Note the setting force_latency=cstate.id:1|2 above. Syntax is as follows:

```
cstate.id:<C-STATE>|<Latency value>
```

See below for more information on setting and verifying the C-States.

2. Activate the new tuned profile by entering the command:

```
# tuned-adm profile sap-hana-kvm-host
```

3. Check that the following command displays sap-hana-kvm-host as a profile:

```
# tuned-adm active
```

Information on C-States

The steps listed above will set the maximal C-State the CPUs can enter to C1, fallback latency value to 2 in case the cstate.id can not be found. See below how to determine the latency value. According to your workload, deeper sleep state might perform better since they allow the CPU to cool down better during phases of low load and then leverage a higher thermal budget when entering a high load phase. Refer to the following table for other configuration options:

Allowed States	force_latency Setting	Remark
C1	`cstate.id:1|2`	Keeps all CPU cores busy in C1, latency to wake up is lower, power consumption is higher and thermal budget (time CPU is able to run in Turbo mode) is lower
C1, C1E	`cstate.id:2|10`	Allows the CPU to go to C1 and C1E
C1, C1E, C6	`cstate.id:3|133`	Allows the CPUs to go to deeper sleep states down to C6. Latency to wake up is higher, power consumption lower and thermal budget is bigger

To check the states available on your system use:

```
# grep . /sys/devices/system/cpu/cpu0/cpuidle/state*/name
/sys/devices/system/cpu/cpu0/cpuidle/state0/name:POLL
/sys/devices/system/cpu/cpu0/cpuidle/state1/name:C1-SKX
/sys/devices/system/cpu/cpu0/cpuidle/state2/name:C1E-SKX
/sys/devices/system/cpu/cpu0/cpuidle/state3/name:C6-SKX
```

To get the latency value which can be used to set

```
# grep . /sys/devices/system/cpu/cpu0/cpuidle/state*/latency
/sys/devices/system/cpu/cpu0/cpuidle/state0/latency:0
/sys/devices/system/cpu/cpu0/cpuidle/state1/latency:2
/sys/devices/system/cpu/cpu0/cpuidle/state2/latency:10
/sys/devices/system/cpu/cpu0/cpuidle/state3/latency:133
```

In order to check which C-States your CPU is in you can use turbostat:

```
# turbostat -qS -i1  -n 10
```

And look for the C1, C1E, C6 columns and the amount of cycles / percentage spent in the various C-States:

```
C1     C1E    C6    POLL%   C1%      C1E%    C6%    CPU%c1    CPU%c6
3383     0     0    0.00    99.99    0.00    0.00    99.97    0.00
2914     0     0    0.00    99.97    0.00    0.00    99.97    0.00
3163     0     0    0.00    99.97    0.00    0.00    99.97    0.00
```

Configuring Skylake-specific settings

On Skylake CPUs, the following setting must be configured to achieve SAP KPIs.

Procedure

1. Set the PLE GAP to zero: For the Intel Skylake processor architecture, add or edit the following parameter in `/etc/modprobe.d/kvm.conf`:

   ```
   options kvm_intel ple_gap=0
   ```

2. Ensure that the new setting is loaded by either rebooting the server or reloading the `kvm_intel` module.

Also make sure that `spectre_v2=retpoline` is set on the kernel command line for both the guest and the hypervisor.

Configuring an RHV cluster running SAP HANA

You can use RHV Manager to administer and manage all RHV hosts and guests deployed on a virtualized infrastructure from a centralized, web browser-based console. All hosts belonging to an RHV cluster share settings that are set across the cluster.

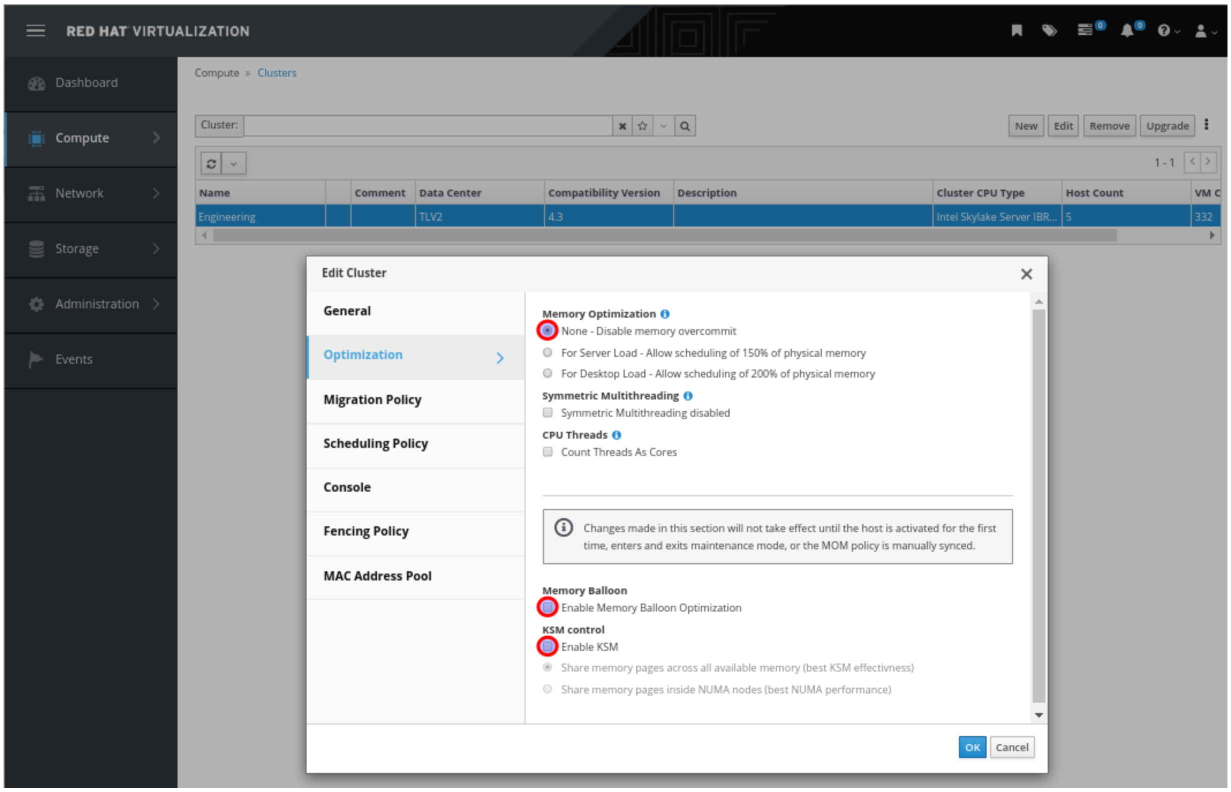
To avoid overcommitting memory resources, complete the following procedure to disable Memory Overcommit, Memory Balloon Optimization, and Kernel Samepage Merging (KSM).

Procedure

1. Click **Compute** > **Clusters** as shown in the following figure.
2. Select the cluster where the RHV host running the SAP HANA VM is located.
3. Click **Edit.**
4. Select the **Optimization** tab.
5. Click **None** for Memory Optimization.
6. Clear the **Enable Memory Balloon Optimization** check box for the entire cluster or on a per VM basis when creating the VM. Disabling ballooning at the cluster level prevents individual VMs from using ballooning.

7. Clear the **Enable KSM** check box.

8. Click **OK.**



Disabling KSM on the host manually

Kernel Samepage Merging (KSM) is used by the KVM hypervisor, allowing KVM guests to share identical memory pages. It is recommended that KSM is deactivated in the RHV-M clustering setup, and also manually deactivated by stopping the `ksmtuned` and the `ksm` service on the hypervisor. The reason is that if Memory Overcommit Manager (MOM) detects memory utilization at 95% or more, it will activate KSM even if it was deactivated in the RHV-M clustering setup. When KSM is disabled, any memory pages shared prior to deactivating KSM are still shared.

Procedure

1. To deactivate KSM, run the following in a terminal as root:

```
# systemctl stop ksmtuned
Stopping ksmtuned:                                          [  OK  ]
# systemctl stop ksm
```

```
Stopping ksm:                                              [  OK  ]

# systemctl disable ksm
# systemctl disable ksmtuned
```

2. To delete all of the Page KSM in the system, run the following command in a terminal as root:

```
# echo 2 >/sys/kernel/mm/ksm/run
```

Installing required hooks for the virtual guests

Configurations that require a high performance VM in RHV environments need a hook. Complete the following procedure to install the required hook.

1. Log in to your hypervisor and create a file, `/usr/libexec/vdsm/hooks/before_vm_start/50_hana` from: https://github.com/mtessun/ovirt-api-scriptlets/blob/master/hooks/before_vm_start/50_hana

   Note: You have to uncomment and modify the following line for live migration. See Set minimal TSC clock frequency in the Configuring Live Migration section.

```
timer.setAttribute('frequency','2494140000')
```

   The following is an example:

```
#!/usr/bin/python2

import os
import sys
import traceback

import hooking

'''
Syntax:
hana=1 (value doesn't matter)

The VM must be configured as High Performance with 1 GB hugepages.
For that the following kernel boot line is required for the hypervisor:

"default_hugepagesz=1GB hugepagesz=1GB hugepages=[# hugepages needed]"

In addition, the "hugepages" custom property needs to be set to 1048576.
'''
```

```
if 'hana' in os.environ:
    try:
        domxml = hooking.read_domxml()
        domain = domxml.getElementsByTagName('domain')[0]
        if not len(domain.getElementsByTagName('memoryBacking')):
            sys.stderr.write('hugepages: VM is no High Performance VM\n')
            sys.exit(0)

        if len(domain.getElementsByTagName('cpu')):
            cpu = domain.getElementsByTagName('cpu')[0]
            feature_tsc = domxml.createElement('feature')
            feature_tsc.setAttribute('policy', 'require')
            feature_tsc.setAttribute('name', 'invtsc')
            feature_rdt = domxml.createElement('feature')
            feature_rdt.setAttribute('policy', 'require')
            feature_rdt.setAttribute('name', 'rdtscp')
            feature_x2apic = domxml.createElement('feature')
            feature_x2apic.setAttribute('policy', 'require')
            feature_x2apic.setAttribute('name', 'x2apic')
            feature_lvl3 = domxml.createElement('cache')
            feature_lvl3.setAttribute('level','3')
            feature_lvl3.setAttribute('mode','emulate')
            cpu.appendChild(feature_tsc)
            cpu.appendChild(feature_rdt)
            cpu.appendChild(feature_lvl3)
            cpu.appendChild(feature_x2apic)

        if len(domain.getElementsByTagName('clock')):
            clock = domain.getElementsByTagName('clock')[0]
            tscClock = domxml.createElement('clock')
            tscClock.setAttribute('offset', 'utc')
            timer = domxml.createElement('timer')
            timer.setAttribute('name','tsc')
            # Uncomment and adjust for live migration (adjust frequency
to match the lowest value in your cluster)
            #timer.setAttribute('frequency','2494140000')
            tscClock.appendChild(timer)
            domain.removeChild(clock)
            domain.appendChild(tscClock)

        hooking.write_domxml(domxml)
    except Exception:
        sys.stderr.write('highperf hook: [unexpected error]: %s\n' %
                         traceback.format_exc())
        sys.exit(2)
```

2. Make the above script executable by running the command:

```
# chmod +x /usr/libexec/vdsm/hooks/before_vm_start/50_hana
```

3. Get the current properties:

```
[root@rhv-m ~]# engine-config -g UserDefinedVMProperties
UserDefinedVMProperties:  version: 3.6
UserDefinedVMProperties:  version: 4.0
UserDefinedVMProperties:  version: 4.1
UserDefinedVMProperties:  version: 4.2
UserDefinedVMProperties:  version: 4.3
```

4. Set the HANA property to use the RHV version 4.2 or 4.3, in this example it is 4.2:

```
[root@rhv-m ~]# engine-config -m UserDefinedVMProperties='hana=^[0-9]+$'
--cver=4.2
```

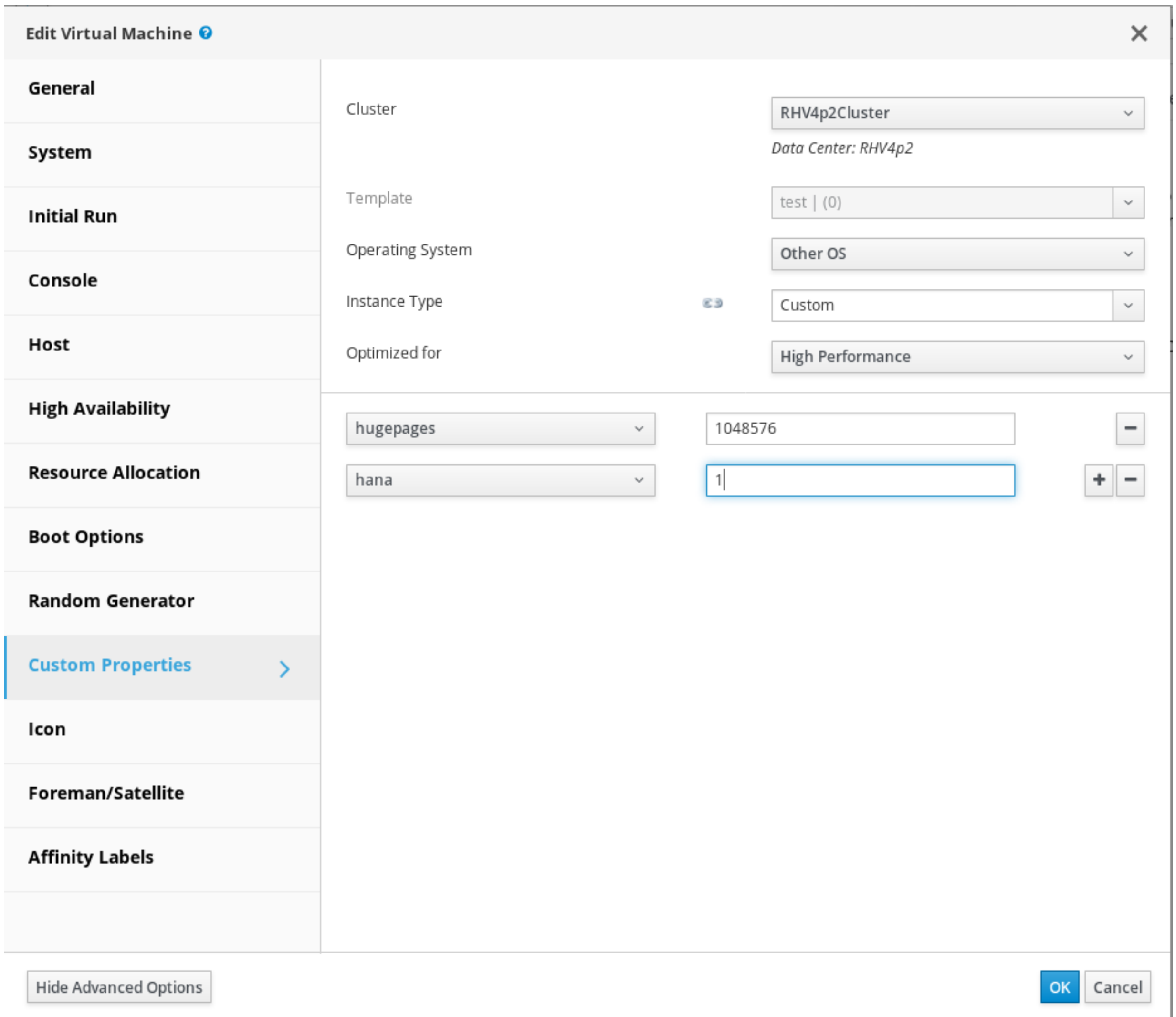
5. Check that the property is set:

```
[root@rhv-m ~]# engine-config -g UserDefinedVMProperties
UserDefinedVMProperties:  version: 3.6
UserDefinedVMProperties:  version: 4.0
UserDefinedVMProperties:  version: 4.1
UserDefinedVMProperties:  hana=^[0-9]+$ version: 4.2
UserDefinedVMProperties:  hana=^[0-9]+$ version: 4.3
```

6. Restart ovirt engine for the changes to take effect:

```
[root@rhv-m ~] /bin/systemctl restart ovirt-engine.service
```

7. To activate the hook, edit the VM in RHV Manager, go to **Custom Properties** and add the variable `hana=1`, as displayed in the following image.


Edit Virtual Machine
General
System
Initial Run
Console
Host
High Availability
Resource Allocation
Boot Options
Random Generator
Custom Properties
Icon
Foreman/Satellite
Affinity Labels
Cluster
RHV4p2Cluster
Data Center: RHV4p2
Template
test | (0)
Operating System
Other OS
Instance Type
Custom
Optimized for
High Performance
hugepages
1048576
hana
1
Hide Advanced Options
OK
Cancel

5. Sizing Guidelines for VMs

There is a performance degradation when running SAP HANA in virtualized environments with Red Hat Virtualization. During validation, the penalty for a single VM was measured to be within the 15% range, depending on the type of the workload. Multi VM environments can cause up to 5% degradation on top. The different workload categories, such as OLTP and OLAP, have shown different degradations.

Influence of Hyperthreading

Using hyperthreading (HT) gives about 30% more performance compared to the same system running without HT, for both virtual and bare metal systems. Before enabling hyperthreading, consider security risks and how they apply to your setup. For more information, see https://access.redhat.com/security/vulnerabilities/mds. When sizing a virtual machine, you must take the performance penalty into account and create a single VM with 10 to 15% more resources than a bare metal system. The data presented here was measured with an Intel Skylake system.

Reviewing the performance degradation between virtual and bare metal systems

Hyperthreading turned on: When a single VM is compared to an equally equipped bare metal (BM) system, the performance loss is up to 15% between BM and VM.

Hyperthreading turned off: The performance delta between the virtual and the equivalent bare metal system, also with HT=off, was in the 10% range. The following table displays the degradation when hyperthreading is turned on and off.

Hyperthreading bare metal system	Hyperthreading virtual system	Degradation to equal bare metal	Vulnerable to MDS
On	On	Up to 15%	Yes
Off	Off	Up to 10%	No

In cases where hyperthreading is set to on, a VM must be 15% larger than an equally equipped bare metal server. When setting hyperthreading to off, the sizing must be 10% larger than an equally configured bare metal system, also with hyperthreading turned off. When comparing a virtual system with hyperthreading turned off to a bare metal system with hyperthreading turned on, the sizing recommendation is 40% larger. Refer to the following table for sizing implications depending on your setting for hyperthreading.

When comparing a bare metal system to a virtual system, consider the following sizing implications regarding the setting of hyperthreading.		
Bare metal Hyperthreading	**Virtual Machine Hyperthreading**	**Sizing implication**
On	On	15% more than an equally equipped bare metal system
Off	Off	10% more than an equally equipped bare metal system
On	Off	40% more than an equally equipped system with HT=on

Sizing for multi VM environments

In order to size a system for a multi VM environment, an additional overhead of 5% has to be taken into account compared to a single VM scenario.

Observations with SAP's BWH performance test

The performance implications depending on which storage variant used are shown in this table.

Storage Variant	**Difference**
Fibre Channel PT	below 5%
NFS	7.76%

This performance drawback of about 3% when using NFS has to be taken into account for data load intensive workloads, such as BWH systems.

Impact of huge page shattering (kvm.nx_huge_pages)

There is a bug in Intel Xeon Scalable Platform CPUs (Skylake and Cascade Lake) that can cause the hypervisor host to crash if an attacker issues certain commands from within the guest. This vulnerability is called X86_BUG_ITLB_MULTIHIT and is automatically mitigated with a technique called huge page shattering. See the section below for more details on how the bug and the mitigation work.

In regards to the SAP HANA validation on RHV performance impact on whether huge page shattering was turned on or not was measured with the following results: In Performance Barrier Tests (PBO) the impact was about 23% worse when huge page shattering was turned on, while other workloads such as ML4 and BWH have shown a significant smaller footprint of maximal 5% for ML4, whereas BWH did seem to perform faster by -2%. These performance numbers were measured on a Cascade Lake system.

The default behavior is that huge page shattering is turned on and should not be changed unless you are really sure what you are doing. The behaviour is controlled via the kernel command line parameter:

kvm.nx_huge_pages=<option>

with the following options:

- force - Always deploy workaround.
- off - Never deploy workaround.
- auto (default) - Deploy workaround based on the presence of X86_BUG_ITLB_MULTIHIT.

If the software workaround is enabled for the host, guests do not need to enable it for nested guests. See also [Important changes to external kernel parameters](#).

Insights on X86_BUG_ITLB_MULTIHIT and huge page shattering

With some Intel processors, putting the same virtual address in the TLB as both a 4 KiB and 2 MiB page can confuse the instruction fetch unit and cause the processor to issue a machine check resulting in a CPU lockup.

Unfortunately when EPT page tables use huge pages, it is possible for a malicious guest to trigger this situation.

Add a knob to mark huge pages as non-executable. When the nx_huge_pages parameter is enabled (and we are using EPT), all huge pages are marked as NX. If the guest attempts to execute in one of those pages, the page is broken down into 4K pages, which are then marked as executable.

This is not an issue for shadow paging (except nested EPT), because the host is then in control of TLB flushes , which prevents this situation from happening.

6. Storage Setup

There are two options presented in this document which have both been certified for usage with SAP HANA. Variant A uses storage controller passthrough but not allowing the VM to be live migratable. Variant B uses an NFS storage server and allows live migration.

To use the Tailored Data Center Integration (TDI) approach, SAP HANA requires an SAP HANA TDI certified storage subsystem. See SAP Certified and Supported SAP HANA Hardware for the list of certified storage systems.

It is necessary to apply the file system layout or partitioning outlined in the SAP HANA Server Installation and Update Guide and the SAP HANA – Storage Requirements Guide. Review the TDI documentation for hardware vendors for vendor-specific storage system setup requirements.

Information on how to set up scale-out scenarios can be found in the SAP HANA Server Installation and Update Guide and the storage hardware vendor's documentation.

For more information about file system layout, partitioning, and sizing, see the *Recommended File System Layout* section in the SAP HANA Server Installation and Update Guide and the SAP HANA – Storage Requirements Guide.

The recommended tool for checking the system compliance are the SAP Hardware and Cloud Measurement Tools (HCMT) available here SAP Note 2493172. It provides tests and reports for additional scenarios to help you determine whether the hardware you plan to use meets the minimum performance criteria required to run SAP HANA in production deployments. SAP requires that you use HCMT to check the hardware infrastructure setup according to the SAP HANA TDI approach

For reference, there is the deprecated Hardware Configuration Check Tool (HWCCT) available in SAP Note 1943937 but should not be used any more.

Storage variant A: Controller passthrough

Using PCI passthrough for the storage controller, e.g. a Fibre Channel Host-Bus-Adapter (HBA) is a tested and certified configuration for SAP HANA. In order to configure PCI storage passthrough, the HBA driver must be blacklisted and the device ID added to `vfio-pci` on the hypervisor.

Make sure that "iommu=pt intel_iommu=on" are present on the boot kernel command line of the hypervisor:

```
# cat /proc/cmdline
```

```
... iommu=pt intel_iommu=on ...
```

Note: This procedure is hardware-dependent and does not allow Live Migration or adapter sharing between multiple VMs. The steps in this procedure include example values for QLogic adapters (qla2xxx driver module). Adjust the values to match your particular hardware.

Note: Passing a HBA to a guest means that the HBA is no longer accessible to the hypervisor. Therefore, the user must either have dedicated HBAs for passthrough, independent of those in use by the hypervisors to access FibreChannel storage domains, or use other forms of storage for the storage domains. The HBAs for passthrough are required to use a different driver than the ones used for the hypervisor. For example, use QLogic adapters for passthrough and Emulex for the hypervisor.

Procedure

1. Blacklist the `qla2xxx` kernel module by adding the following line to `/etc/modprobe.d/blacklist.conf`:

   ```
   blacklist qla2xxx
   ```

2. Determine the device ID. In this example, a QLogic HBA is used. Therefore, enter `lspci -nn` for QLogic and search the output for the ID **1077:2261**:

   ```
   # lspci -nn | grep QLogic
   6d:00.0 Fibre Channel [0c04]: QLogic Corp. ISP2722-based 16/32Gb Fibre
   Channel to PCIe Adapter [1077:2261] (rev 01)
   6d:00.1 Fibre Channel [0c04]: QLogic Corp. ISP2722-based 16/32Gb Fibre
   Channel to PCIe Adapter [1077:2261] (rev 01)
   ```

3. Add the device ID to `vfio-pci` by adding the following line, which contains the ID from the previous step, to `/etc/modprobe.d/vfio.conf`:

   ```
   options vfio-pci ids=1077:2261
   ```

4. Update `initrd`. Ensure the following line is in the ramdisk for the kernel:

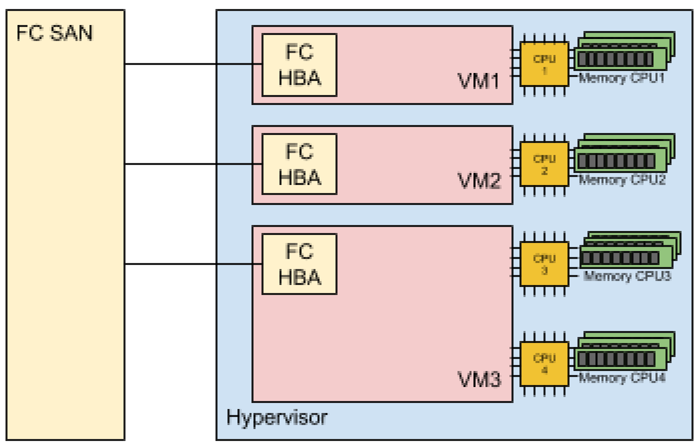
   ```
   # dracut --force
   ```

5. Reboot the host through RHV-M GUI:

    a. Set the RHV host into maintenance by selecting **Management > Maintenance**.
    b. Reboot the RHV host by selecting **Management > SSH Management > Restart.**
    c. After the reboot is completed, activate the RHV host by selecting **Management > Activate.**

Setting up Fibre Channel HBA passthrough for virtual machines

Running multiple VMs on the same physical hypervisor requires one storage device per VM, for example, a Fibre Channel HBA. The HBA is connected to the guest via IOMMU using the virtio-pci module. The supported VM sizes are one, two, and four NUMA nodes. The amount of memory available in a VM is limited to the memory that is physically attached to the CPUs that are used by the VM.



Note: PCIe slots are physically wired to a specific CPU. To achieve the best performance, the adapter must be located in the slot that is attached to the CPU where the VM is running. Consult your hardware vendor for more information. For a two-NUMA node VM, the HBA must be attached to either one of the nodes, and the IO thread must be pinned to that node. Complete the following procedure to verify which slot is attached to a specific NUMA node.

Procedure

1. To verify which slot is attached to a specific NUMA node, run the command (on hypervisor):

```
# cat /sys/bus/pci/devices/<PCI device>/numa_node
```

The following example is for QLogic FC HBAs:

```
# lspci | grep QLogic
25:00.0 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
25:00.1 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
33:00.0 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
33:00.1 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
6d:00.0 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
6d:00.1 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
85:00.0 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
85:00.1 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
c5:00.0 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
c5:00.1 Fibre Channel: QLogic Corp. ISP2722-based 16/32Gb Fibre Channel to PCIe Adapter
(rev 01)
```

The PCI device IDs are: `25:00.x, 33:00.x, 6d:00.x, 85:00.x, and c5:00.x.`

2. To check the IDs, run the command:

```
# for i in 25 33 6d 85 c5; do echo $i\:00.x; cat
/sys/bus/pci/devices/0000\:$i\:00.?/numa_node; done
25:00.x
0
0
33:00.x
0
0
6d:00.x
1
1
85:00.x
2
2
c5:00.x
3
```

3

The command output displays that 25:00.x and 33:00.x are connected to NUMA node 0, 6d:00.x is connected to NUMA node 1, 85:00.x is connected to node 2, and c5:00.x is connected to node 3. Consider this information when selecting the HBA for a specific VM.

Example configuration for a two-socket VM

The following example configuration shows the relevant snippets of the XML config file for a NUMA node VM running on nodes two and three. One core per node has been reserved for the hypervisor and I/O threads. From a total of 28 physical cores per CPU, 27 are passed through to the guest. The Fibre Channel HBA connected to NUMA node two is also passed through to the guest. Refer to Chapter 15. PCI passthrough for more information.

```
<domain>
  ...
  <memoryBacking>
        <hugepages>
        <page size='1048576' unit='KiB'/>
        </hugepages>
  </memoryBacking>
  <vcpu placement='static'>108</vcpu>    <!-- TOTAL NUMBER OF CORES/THREADS -->
  <iothreads>1</iothreads>
  <iothreadids>
        <iothread id='1'/>
  </iothreadids>
  <cputune>
        <vcpupin vcpu='0' cpuset='6,118'/>    <!-- CORES OF PHYSICAL NUMA NODE 2 -->
        <vcpupin vcpu='1' cpuset='6,118'/>    <!-- NOT USING THE FIRST CORE+HT -->
        <vcpupin vcpu='2' cpuset='10,122'/>
        <vcpupin vcpu='3' cpuset='10,122'/>
       ...
        <vcpupin vcpu='54' cpuset='7,119'/>    <!-- CORES OF NUMA NODE 3 -->
        <vcpupin vcpu='55' cpuset='7,119'/>
        <vcpupin vcpu='56' cpuset='11,123'/>
        <vcpupin vcpu='57' cpuset='11,123'/>
       ...
       <vcpupin vcpu='106' cpuset='111,223'/>
        <vcpupin vcpu='107' cpuset='111,223'/>
        <emulatorpin cpuset='2,114'/>               <!--CORES FOR IOTHREADS ON NUMA NODE 2 -->
        <iothreadpin iothread='1' cpuset='2,114'/>  <!--WHICH WERE RESERVED FOR HYPERVISOR AND IOTHREAD-->
  </cputune>
  <numatune>
        <memory mode='strict' nodeset='2-3'/>    <!-- BIND THE MEMORY WHICH IS ON NUMA NODE 2 AND 3 -->
        <memnode cellid='0' mode='strict' nodeset='2'/>
        <memnode cellid='1' mode='strict' nodeset='3'/>
  </numatune>
  ...
  <cpu mode='host-passthrough' check='none'>
        <topology sockets='2' cores='27' threads='2'/><!--CPU TOPOLOGY TO REFLECT 1:1 WHATS THERE IN HW-->
        <cache level='3' mode='emulate'/>
        <feature policy='require' name='rdtscp'/>
```

```
        <feature policy='require' name='invtsc'/>
        <feature policy='require' name='x2apic'/>
        <numa>
        <cell id='0' cpus='0-53' memory='536870912' unit='KiB'/>      <!-- AMOUNT OF MEMORY USED -->
        <cell id='1' cpus='54-107' memory='536870912' unit='KiB'/>    <!-- PER VIRTUAL NUMA NODE -->
        </numa>
           ...
        <hostdev mode='subsystem' type='pci' managed='yes'>    <!-- PASSTHROUGH OF THE ADAPTER WITH -->
        <driver name='vfio'/>                                  <!-- PCI ID 85:00.x FOR BOTH PORTS -->
        <source>
        <address domain='0x0000' bus='0x85' slot='0x00' function='0x0'/>
        </source>
        <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0' multifunction='on'/>
        </hostdev>
        <hostdev mode='subsystem' type='pci' managed='yes'>
        <driver name='vfio'/>
        <source>
        <address domain='0x0000' bus='0x85' slot='0x00' function='0x1'/>
        </source>
        <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x1' multifunction='on'/>
        </hostdev>
        <memballoon model='none'/>
  </devices>
  ...
</domain>
```

Storage variant B: NFS Storage

For the correct setup of your NFS server for SAP HANA, consult your storage vendor. For the validation we used a Netapp AFF 300 Filer and applied the settings described in Netapp's Guide: SAP HANA on NetApp FAS Systems with NFS
https://www.netapp.com/us/media/tr-4290.pdf

The key components of this setup are outlined here as follows:

For a four socket system there are at least two dedicated physical 10G NIC ports required: one for /hana/data, and one for /hana/log, respectively. Use SR-IOV to bring the virtual functions (virtual NIC) into the VM. See this section: Create an SR-IOV Network on how to setup SR-IOV.

Use Jumbo Frames (9000) as MTU for your network.

Distribute the /hana/data and /hana/log volumes equally on the NFS server, e.g. for a Netapp Filer there are two aggregates, /hana/data should go to aggregate01, /hana/log to aggregate02. For multiple VMs reverse the mapping, e.g. VM1:/hana/data uses NIC1 and aggregate01, VM1:/hana/log uses NIC2 and aggregate02. VM2:/hana/data uses NIC2 and aggregate02, VM2:/hana/log uses NIC1 and aggregate01 and so on.

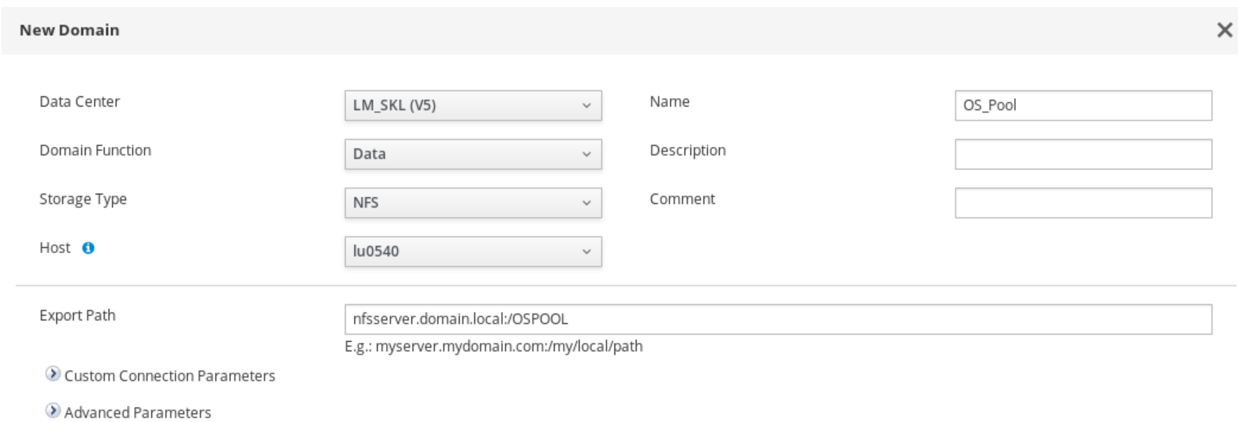
The volume for /hana/share can be either on NIC1 or on NIC2 (and on aggregate 01 or aggregate 02), but when deploying multiple VMs they should be distributed equally, e.g.

VM1:/hana/shared is on NIC1 and aggregate 01, VM2:/hana/shared is on NIC2 and aggregate02, VM3:/hana/shared on NIC1 and aggregate 01 and so on.

Size the NFS storage server side accordingly and also according to your HA requirements.

Create OS Storage Domain

Create a volume on your NFS storage server which will contain the Operating System (OS) for the guests. Add this volume as NFS storage domain to RHVM, see screenshot below for an example. Make sure that this volume is available on all hypervisors in your cluster. Use this later for the OS disk when installing the guests.



MOUNT NFS Volumes in GUEST and Mount Options

The volumes for SAP HANA are mounted directly in the guest. The mount options used here for Netapp Filer were:

```
rw,vers=4.1,hard,timeo=600,rsize=1048576,wsize=1048576,bg,noatime,lock
```

Add entries in /etc/fstab so the volumes get mounted automatically at boot time, e.g.:

```
192.168.2.100:/lu0563_hana_data /hana/data                nfs↲
rw,vers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,bg,noatime,lock 0 0

192.168.2.101:/lu0563_hana_log  /hana/log                 nfs↲
rw,vers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,bg,noatime,lock 0 0

192.168.2.101:/lu0563_hana_shared  /hana/shared           nfs↲
rw,vers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,bg,noatime,lock 0 0
```

7. Configuring Virtual Machines

For optimal performance, you must configure the VM settings to reflect the CPU and memory resources of the underlying physical hardware.

About virtual NUMA and virtual CPU Pinning

Virtual CPU pinning is required to ensure that the NUMA configuration in the VM matches the one on the RHV host. This pinning helps avoid performance degradation caused by cross-NUMA node calls. CPU pinning ensures a virtual CPU thread is assigned to a specific physical CPU. To align the SAP HANA VM with the hardware it is running on, complete the following procedures to configure the SAP HANA VM to use virtual CPU pinning and virtual NUMA.

Note: Minimize the memory overhead by disabling virtual devices that are not required as described in the following procedure. If you require a graphical console, use VNC rather than SPICE.

Configuring an SAP HANA VM to use CPU pinning and Virtual NUMA

Complete this procedure to configure an SAP HANA VM to use the CPU pinning and virtual NUMA settings.

Procedure

1. To get the host hardware topology, log in to the RHV host and enter the following command:

   ```
   # lscpu
   ```

   Review the parameters in the `lscpu` command output:

   - CPU(s)
   - Thread(s) per core
   - Core(s) per socket
   - Socket(s)
   - Numa node(s)
   - CPUs per NUMA node

   For example, the following output is from a 4-socket Intel[R] Xeon[R] Scalable Platform Platinum 8280M CPU server:

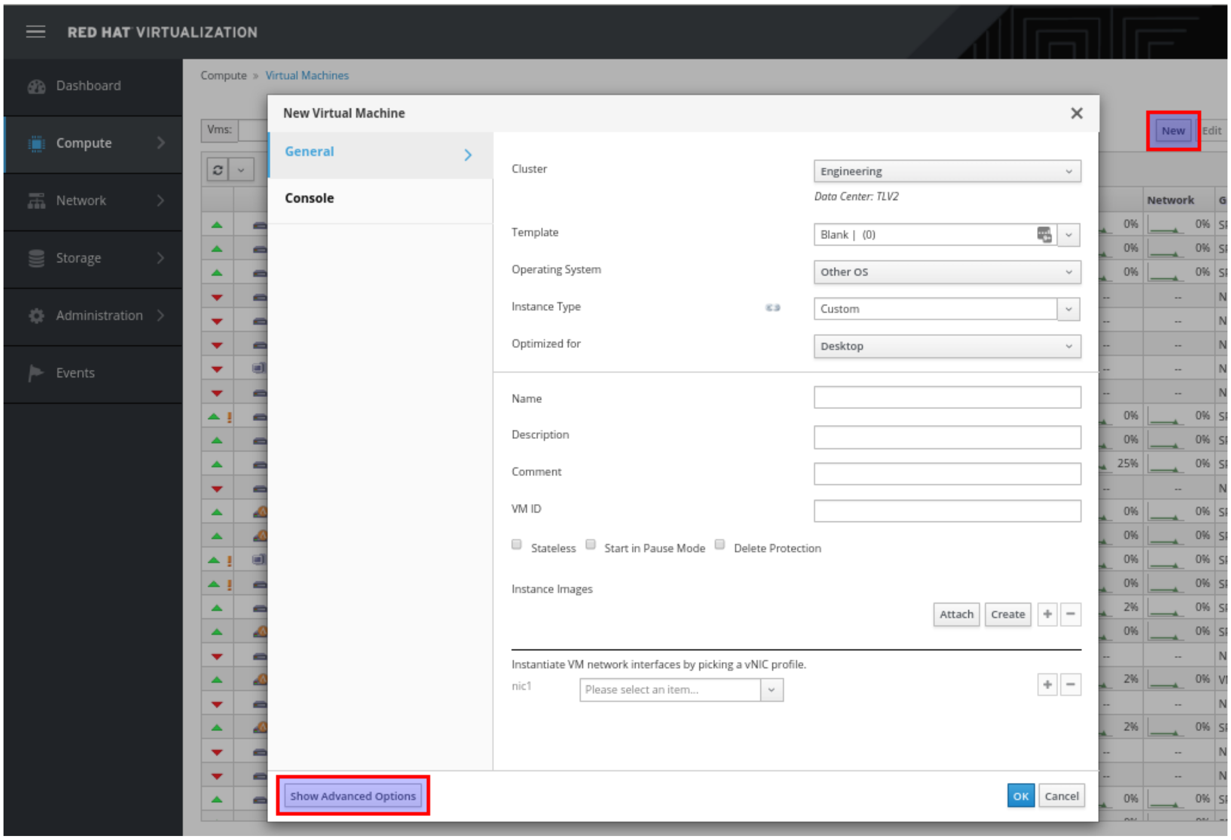
   ```
   # lscpu
   Architecture:            x86_64
   CPU op-mode(s):          32-bit, 64-bit
   Byte Order:              Little Endian
   CPU(s):                  224
   On-line CPU(s) list:   0-223
   Thread(s) per core:      2
   Core(s) per socket:      28
   Socket(s):               4
   NUMA node(s):            4
   Vendor ID:               GenuineIntel
   CPU family:              6
   Model:                   85
   Model name:              Intel(R) Xeon(R) Platinum 8280M CPU @ 2.70GHz
   Stepping:                7
   CPU MHz:                 2700.000
   CPU max MHz:             2700.0000
   CPU min MHz:             1000.0000
   BogoMIPS:                5400.00
   Virtualization:          VT-x
   L1d cache:               32K
   L1i cache:               32K
   L2 cache:                1024K
   ```

```
L3 cache:                39424K
NUMA node0 CPU(s):
0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60,64,68,72,76,80,84,88,92,96,1
00,104,108,112,116,120,124,128,132,136,140,144,148,152,156,160,164,168,17
2,176,180,184,188,192,196,200,204,208,212,216,220
NUMA node1 CPU(s):
1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61,65,69,73,77,81,85,89,93,97,1
01,105,109,113,117,121,125,129,133,137,141,145,149,153,157,161,165,169,17
3,177,181,185,189,193,197,201,205,209,213,217,221
NUMA node2 CPU(s):
2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62,66,70,74,78,82,86,90,94,98,
102,106,110,114,118,122,126,130,134,138,142,146,150,154,158,162,166,170,1
74,178,182,186,190,194,198,202,206,210,214,218,222
NUMA node3 CPU(s):
3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63,67,71,75,79,83,87,91,95,99,
103,107,111,115,119,123,127,131,135,139,143,147,151,155,159,163,167,171,1
75,179,183,187,191,195,199,203,207,211,215,219,223
Flags:                   fpu vme de pse tsc msr pae mce cx8 apic sep mtrr
pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe
syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts
rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq
dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid
dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx
f16c rdrand lahf_lm abm 3dnowprefetch epb cat_l3 cdp_l3 invpcid_single
intel_ppin intel_pt ssbd mba ibrs ibpb stibp ibrs_enhanced tpr_shadow
vnmi flexpriority ept vpid fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2
erms invpcid rtm cqm mpx rdt_a avx512f avx512dq rdseed adx smap
clflushopt clwb avx512cd avx512bw avx512vl xsaveopt xsavec xgetbv1
cqm_llc cqm_occup_llc cqm_mbm_total cqm_mbm_local dtherm arat pln pts pku
ospke avx512_vnni md_clear spec_ctrl intel_stibp flush_l1d
arch_capabilities
```

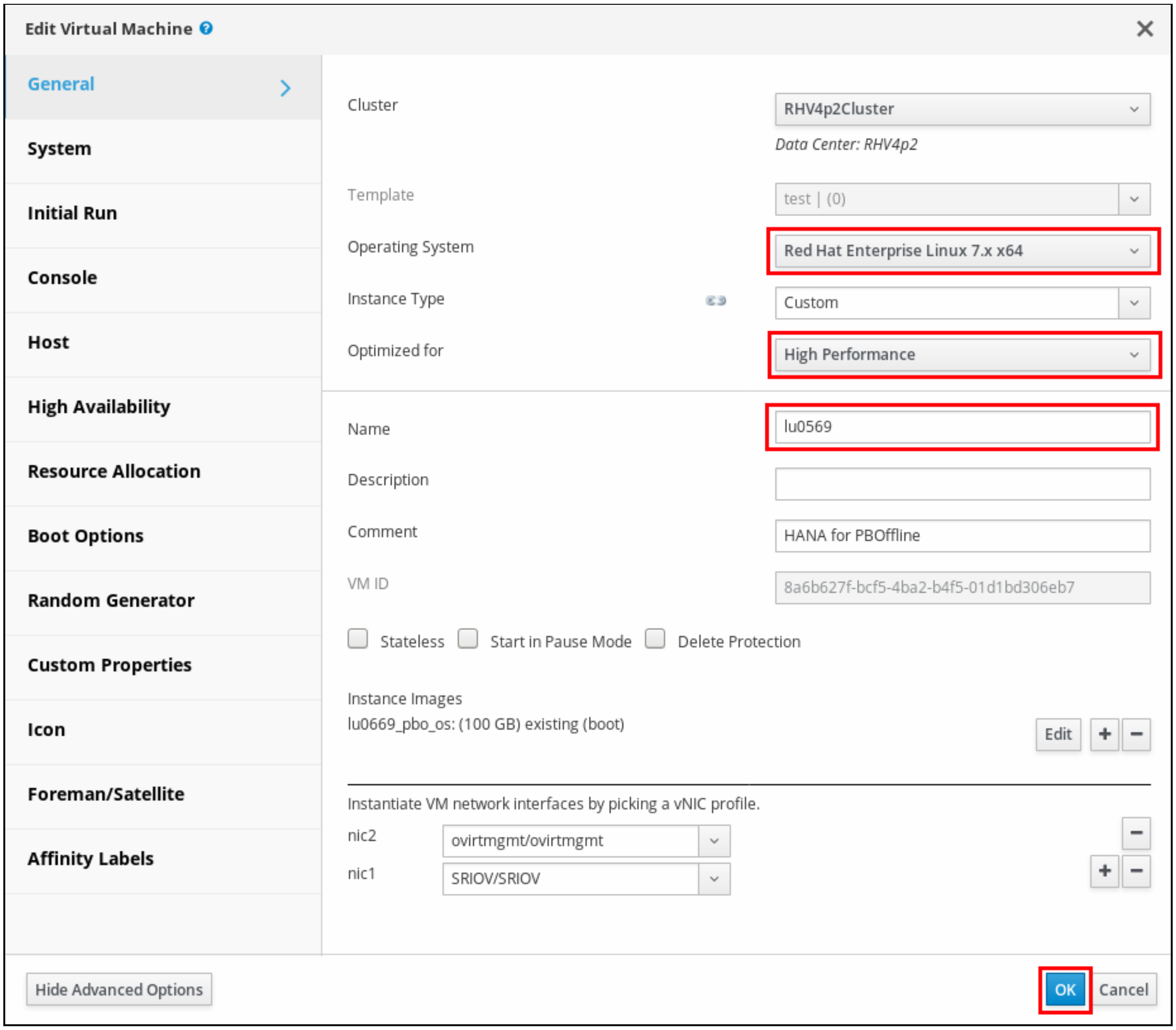
NOTE: The output of the `lscpu` command varies depending on the hardware vendor. You must understand the format used by the hardware vendor to interpret the output of this query.

To calculate the pinning for your guest, use the script provided in Appendix: Calculate CPU Pinning.

2. In RHV Manager, click **Compute** > **Virtual Machines**.
3. Click **New.**
4. In the lower-left corner of the **New Virtual Machine** window, click **Show Advanced Options** as shown in the following figure.



5. At a minimum, complete the parameters in the **General** tab as shown below and in the following figure:
    a. Operating System: **Red Hat Enterprise Linux <your major OS Version>**
    b. Optimized for: **High Performance VM.**
    c. Name: **<Name of your SAP HANA VM>.**



6. Click the **System** tab and set the following parameters:

    a. Enter the **Memory Size** for your SAP HANA VM.

       NOTE: The memory allocated for the SAP HANA VM i262144 MB262144 MB

    b. Enter the **Maximum Memory** equal to **Memory Size**.

    c. Enter the number of **Total Virtual CPUs,** which needs to be lower than or equal to the **CPU(s)** from the `lscpu` command output obtained earlier from the RHV host.

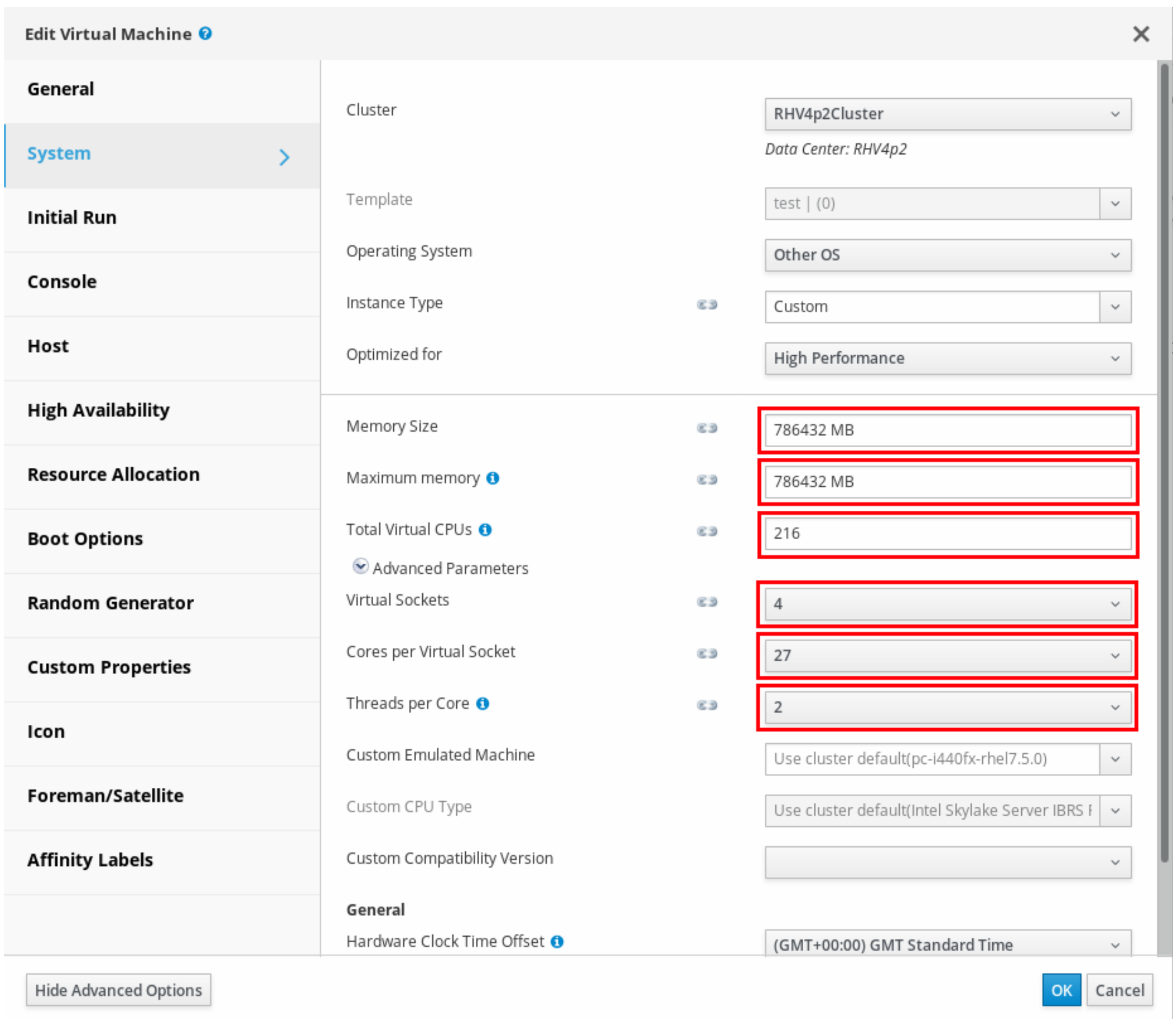
    d. Click **Advanced Parameters**.

    e. Adjust the CPU topology according to the `lscpu` command output:

i. **Virtual Sockets** equal to **Socket(s)** from the `lscpu` command output.

ii. **Cores per Virtual Socket** equal to **Core(s) per socket** minus 1 (-1) from the `lscpu` command output for reserving some resources for the RHV host.

iii. **Threads per Core** equal to **Thread(s) per core** from the `lscpu` command output.



7. For a graphical console, click the **Console** tab and set the following:

   a. Clear **Headless Mode**.

   b. Select **VNC** for **Graphics protocol.**

c. Clear **Soundcard enabled.**

8. Click the **Host** tab and set the following parameters:

   a. Select **Specific host** in the **Start Running On** section.

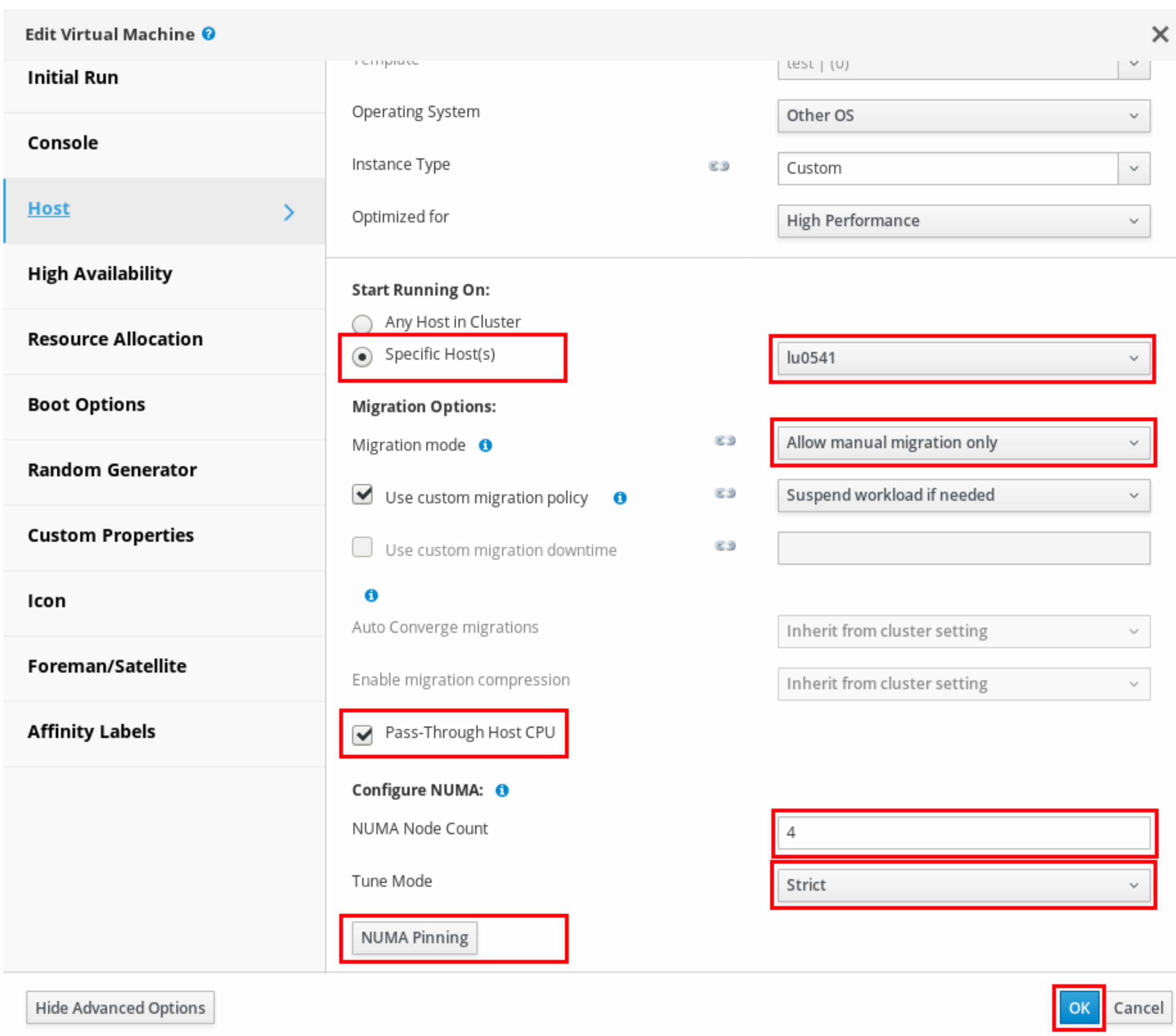
   b. Select the RHV hosts you want to run your SAP HANA VM on from the drop-down list that appears.

   c. Select **Allow manual migration only** from the drop-down list for **Migration Mode.**

   d. Verify that **Pass-Through Host CPU** is selected.

   e. Set the **NUMA Node Count** equal to the **Numa node(s)** from the `lscpu` output.

   f. Select **Strict** from the drop-down list for **Tune Mode.**

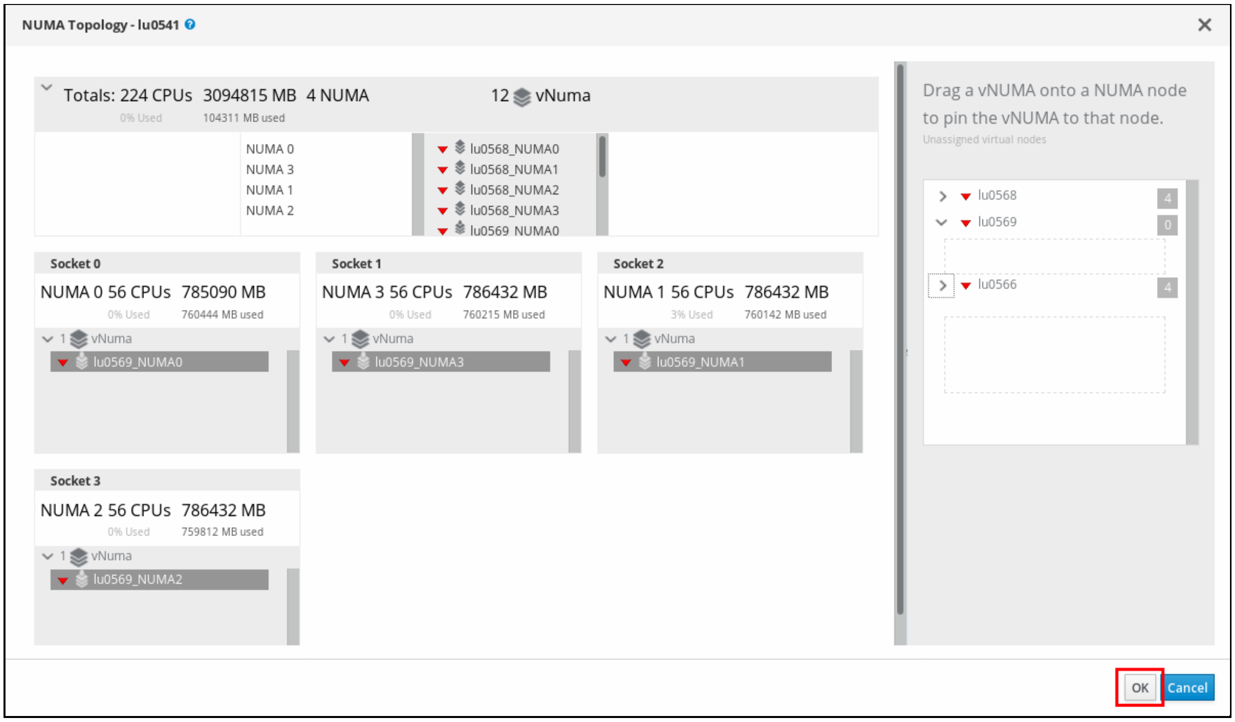


g. Click **NUMA Pinning.**

h. Drag and drop the virtual NUMA nodes according to the physical NUMA nodes. As shown in the following figure, the NUMA-node numbers for physical and virtual NUMA nodes must match.

   Note: that there might be a bug which does not align Socket number and NUMA node number, as shown in screenshot. In that case, make sure to match the NUMA numbers, e.g. referring to the screenshot below, NUMA3 has to go to Socket 1.

i. Click **OK.**



9. Click the **Resource Allocation** tab and set the following:

    a. Ensure **Disabled** is selected from the drop-down list for **CPU shares.**

    b. Enter the virtual CPU pinning to the **CPU pinning topology** as follows:

    `vcpu0#pcpu1_vcpu1#pcpu2_vcpu2#pcpu3…` and so on.

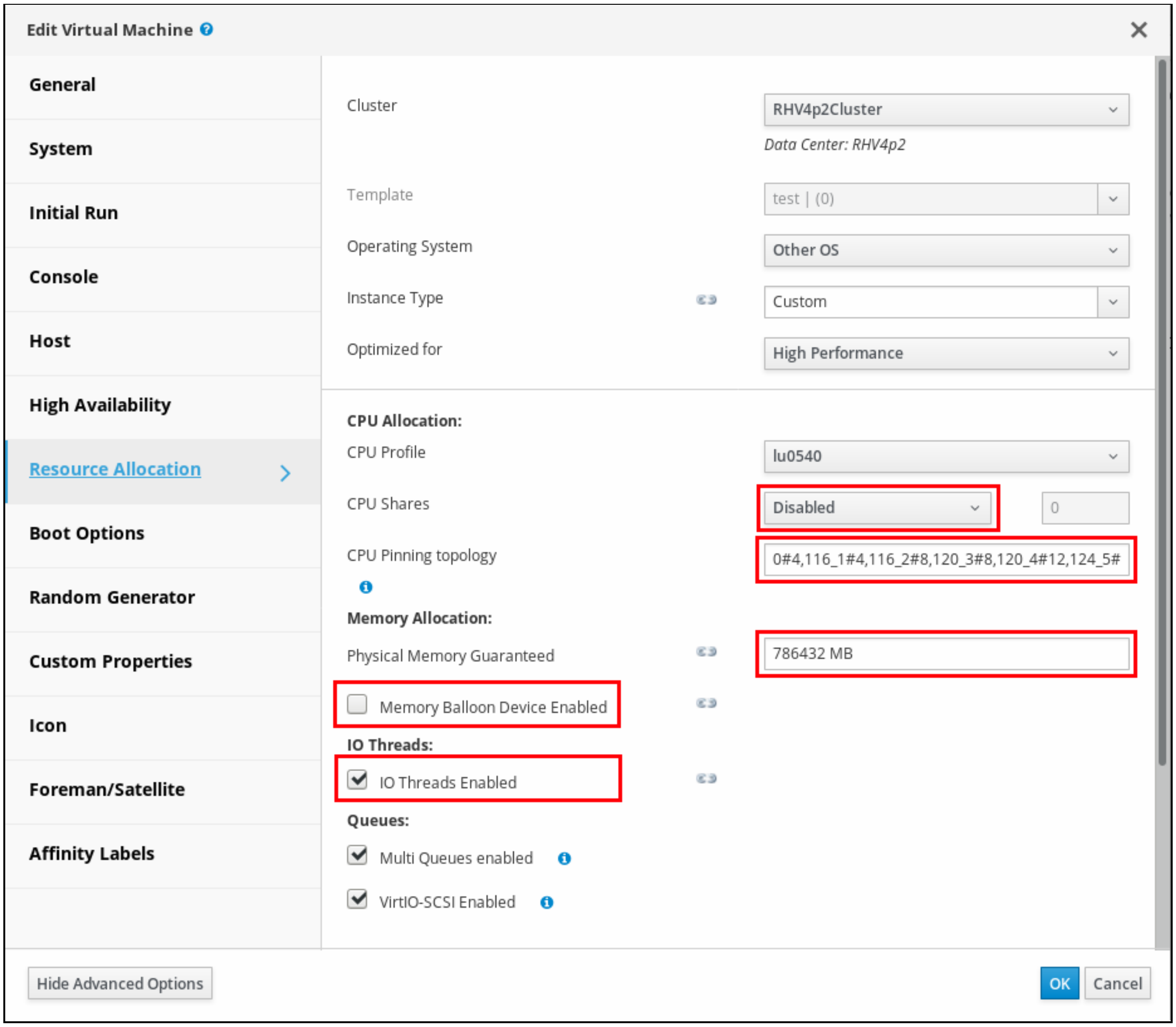
    Refer to Appendix section Calculate CPU Pinning and the script `calculate_cpu_pinning.sh` provided there in order to create the pinning as shown here:

```
# calculate_cpu_pinning.sh 216

limiting to 27 cpus/numa node.

0#4,116_1#4,116_2#8,120_3#8,120_4#12,124_5#12,124_6#16,128_7#16,128_8#20,132
_9#20,132_10#24,136_11#24,136_12#28,140_13#28,140_14#32,144_15#32,144_16#36,
148_17#36,148_18#40,152_19#40,152_20#44,156_21#44,156_22#48,160_23#48,160_24
#52,164_25#52,164_26#56,168_27#56,168_28#60,172_29#60,172_30#64,176_31#64,17
6_32#68,180_33#68,180_34#72,184_35#72,184_36#76,188_37#76,188_38#80,192_39#8
0,192_40#84,196_41#84,196_42#88,200_43#88,200_44#92,204_45#92,204_46#96,208_
47#96,208_48#100,212_49#100,212_50#104,216_51#104,216_52#108,220_53#108,220_
54#5,117_55#5,117_56#9,121_57#9,121_58#13,125_59#13,125_60#17,129_61#17,129_
62#21,133_63#21,133_64#25,137_65#25,137_66#29,141_67#29,141_68#33,145_69#33,
145_70#37,149_71#37,149_72#41,153_73#41,153_74#45,157_75#45,157_76#49,161_77
#49,161_78#53,165_79#53,165_80#57,169_81#57,169_82#61,173_83#61,173_84#65,17
```

```
7_85#65,177_86#69,181_87#69,181_88#73,185_89#73,185_90#77,189_91#77,189_92#8
1,193_93#81,193_94#85,197_95#85,197_96#89,201_97#89,201_98#93,205_99#93,205_
100#97,209_101#97,209_102#101,213_103#101,213_104#105,217_105#105,217_106#10
9,221_107#109,221_108#6,118_109#6,118_110#10,122_111#10,122_112#14,126_113#1
4,126_114#18,130_115#18,130_116#22,134_117#22,134_118#26,138_119#26,138_120#
30,142_121#30,142_122#34,146_123#34,146_124#38,150_125#38,150_126#42,154_127
#42,154_128#46,158_129#46,158_130#50,162_131#50,162_132#54,166_133#54,166_13
4#58,170_135#58,170_136#62,174_137#62,174_138#66,178_139#66,178_140#70,182_1
41#70,182_142#74,186_143#74,186_144#78,190_145#78,190_146#82,194_147#82,194_
148#86,198_149#86,198_150#90,202_151#90,202_152#94,206_153#94,206_154#98,210
_155#98,210_156#102,214_157#102,214_158#106,218_159#106,218_160#110,222_161#
110,222_162#7,119_163#7,119_164#11,123_165#11,123_166#15,127_167#15,127_168#
19,131_169#19,131_170#23,135_171#23,135_172#27,139_173#27,139_174#31,143_175
#31,143_176#35,147_177#35,147_178#39,151_179#39,151_180#43,155_181#43,155_18
2#47,159_183#47,159_184#51,163_185#51,163_186#55,167_187#55,167_188#59,171_1
89#59,171_190#63,175_191#63,175_192#67,179_193#67,179_194#71,183_195#71,183_
196#75,187_197#75,187_198#79,191_199#79,191_200#83,195_201#83,195_202#87,199
_203#87,199_204#91,203_205#91,203_206#95,207_207#95,207_208#99,211_209#99,21
1_210#103,215_211#103,215_212#107,219_213#107,219_214#111,223_215#111,223
```

The `lscpu` output should look similar to the following example:

```
[...]
NUMA node0 CPU(s):
0,4,8,12,16,20,24,28,32,36,40,44,48,52,56,60,64,68,72,76,80,84,88,92,96,100,
104,108,112,116,120,124,128,132,136,140,144,148,152,156,160,164,168,172,176,
180,184,188,192,196,200,204,208,212,216,220
NUMA node1 CPU(s):
1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61,65,69,73,77,81,85,89,93,97,101,
105,109,113,117,121,125,129,133,137,141,145,149,153,157,161,165,169,173,177,
181,185,189,193,197,201,205,209,213,217,221
NUMA node2 CPU(s):
2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62,66,70,74,78,82,86,90,94,98,102
,106,110,114,118,122,126,130,134,138,142,146,150,154,158,162,166,170,174,178
,182,186,190,194,198,202,206,210,214,218,222
NUMA node3 CPU(s):
3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63,67,71,75,79,83,87,91,95,99,103
,107,111,115,119,123,127,131,135,139,143,147,151,155,159,163,167,171,175,179
,183,187,191,195,199,203,207,211,215,219,223
```



NOTE:

- Reserve the first core/thread pair of each physical socket for the RHV host as shown in the previous example.

c. If selected, clear the **Memory Balloon Device Enabled** check box.

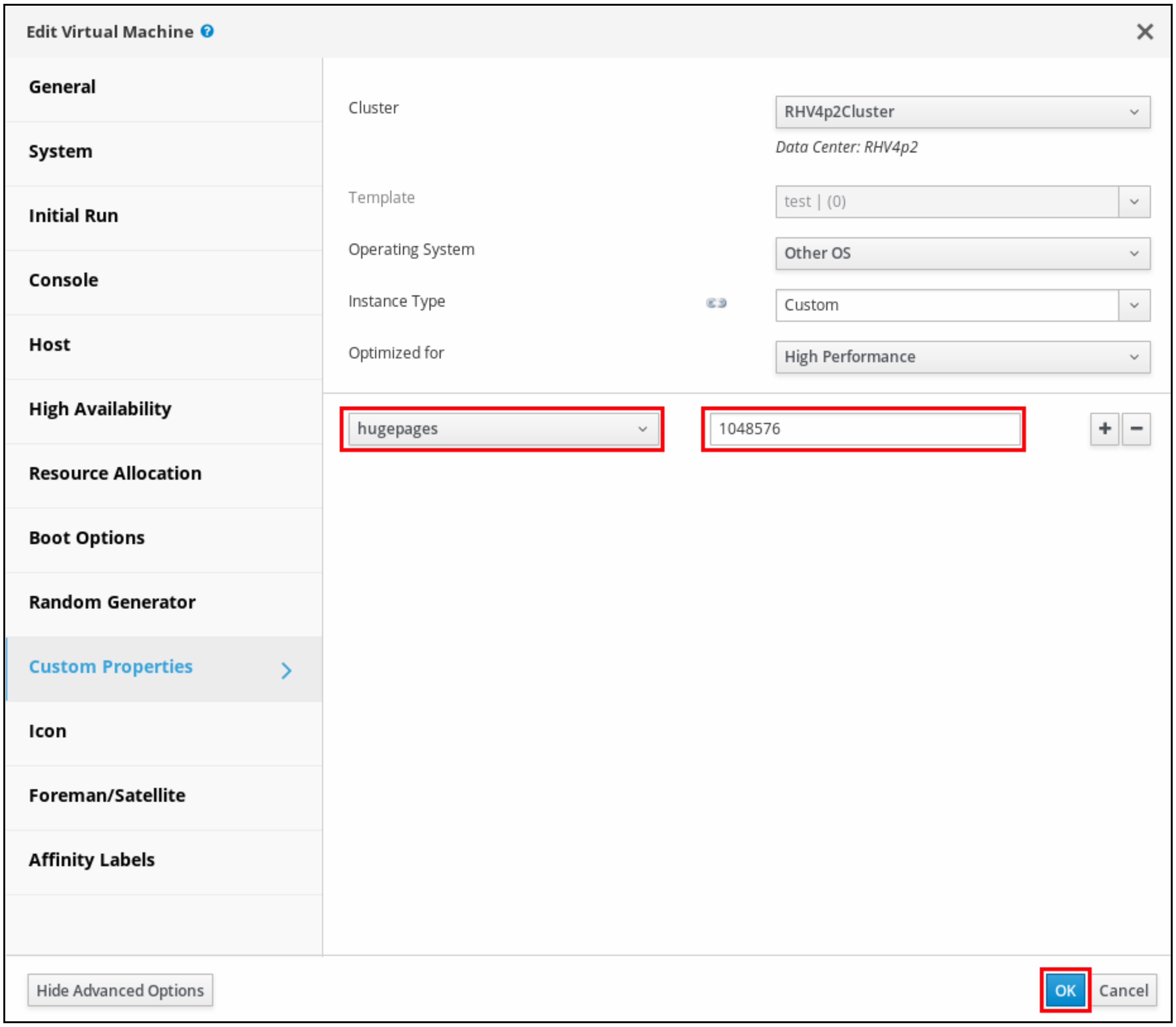
d. Select the **IO Threads Enabled** check box.

10. Click the **Custom Properties** tab and set the following options as shown below and in the following figure:

    a. Select **hugepages** from the drop-down list.

    b. Enter **1048576** (equals 1 GiB in KiB) in the field that is displayed to the right of the

hugepages drop-down list.



11. Click **OK** to create the SAP HANA VM.

12. Attach required storage as described in the *Creating a Linux Virtual Machine* section in the *Virtual Machine Management Guide*.

Setting up networking

This chapter describes how to setup networking for the various use cases in context of SAP HANA on RHV. Note that in this document there are two general types of networks: SR-IOV and bridged. While SR-IOV provides the best performance, it has the downside of not being live migratable. Therefore a documented workaround with a bridge network exists, refer to section Configuring Live Migration for more details. The following table gives an overview of the networks needed per use case, each of which require a dedicated network port.

<table>
<tr><th>Network Purpose</th><th>Type</th><th>Minimal Speed</th><th>MTU</th><th>Remark</th></tr>
<tr><td>OS Access</td><td>Bridge or SR-IOV</td><td>-</td><td>Depending on customer environment</td><td>Can be joined with HANA Access</td></tr>
<tr><td>HANA Access</td><td>SR-IOV</td><td colspan="2" rowspan="2">Depending on your workload, usually 10GbE with 9000 MTU</td><td>Client access to DB</td></tr>
<tr><td>HANA Access Live Migration Bridge Fallback</td><td>Bridge</td><td>Only needed when using Live Migration</td></tr>
<tr><td>HANA Cluster</td><td>SR-IOV</td><td>10GbE</td><td>9000</td><td>Scale-out only</td></tr>
<tr><td>HANA NFS Data</td><td>SR-IOV</td><td>10GbE</td><td>9000</td><td>NFS only</td></tr>
<tr><td>HANA NFS Log</td><td>SR-IOV</td><td>10GbE</td><td>9000</td><td>NFS only</td></tr>
<tr><td>Live Migration</td><td>Bridge or SR-IOV</td><td>10GbE</td><td>9000</td><td>Live Migration network, configure IP and subnet on the RHV hypervisor</td></tr>
</table>

Determine which networks are needed for the use cases you want to implement. Eachnetwork then has to be defined in RHV-M. Only the Live Migration network needs also IP addresses set for each hypervisor.

Depending on the type of network (bridged or SR-IOV), there are slightly different steps required in order to create them. Refer to the following sections how to create them.

Create an SR-IOV Network

For an optimum of network performance use SR-IOV-enabled network interfaces. With SR-IOV devices you can achieve performance levels that are close to bare metal because SR-IOV does not require any virtual bridges to communicate with the network.

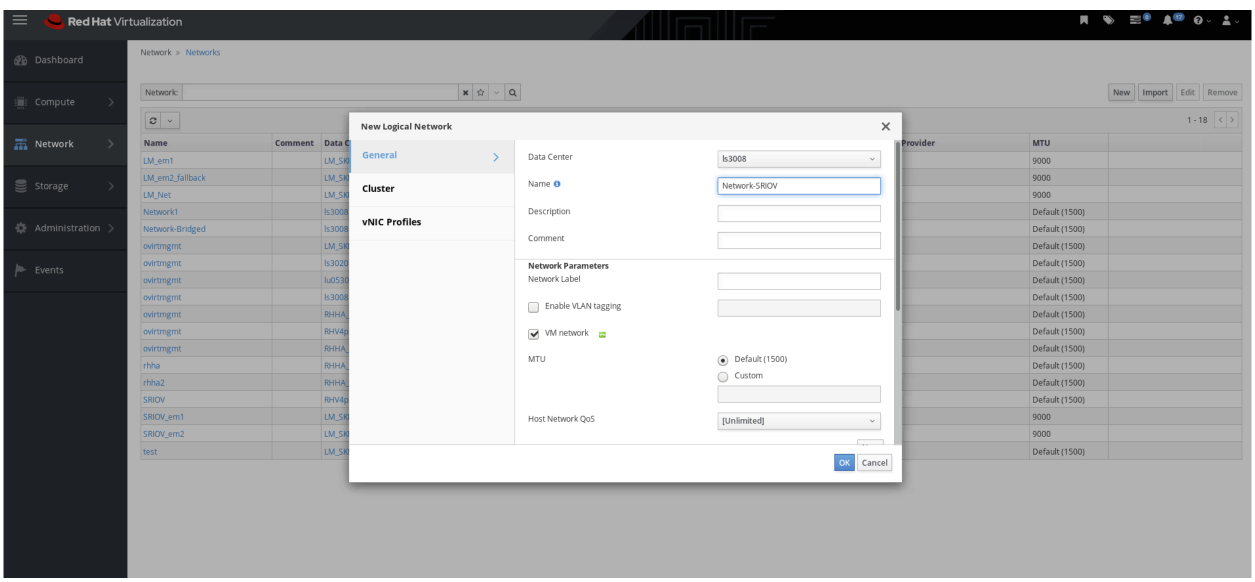
Prerequisites

- SR-IOV capable network cards. Ensure that SR-IOV is enabled for the NIC in BIOS.
- IOMMU enabled in the BIOS of the Server (VT-d enable).

Complete this procedure to create an SR-IOV-enabled virtual network. For more information, see the Setting up and configuring SR-IOV and Enabling Passthrough on a vNIC Profile section of the *Red Hat Virtualization Administration Guide*.

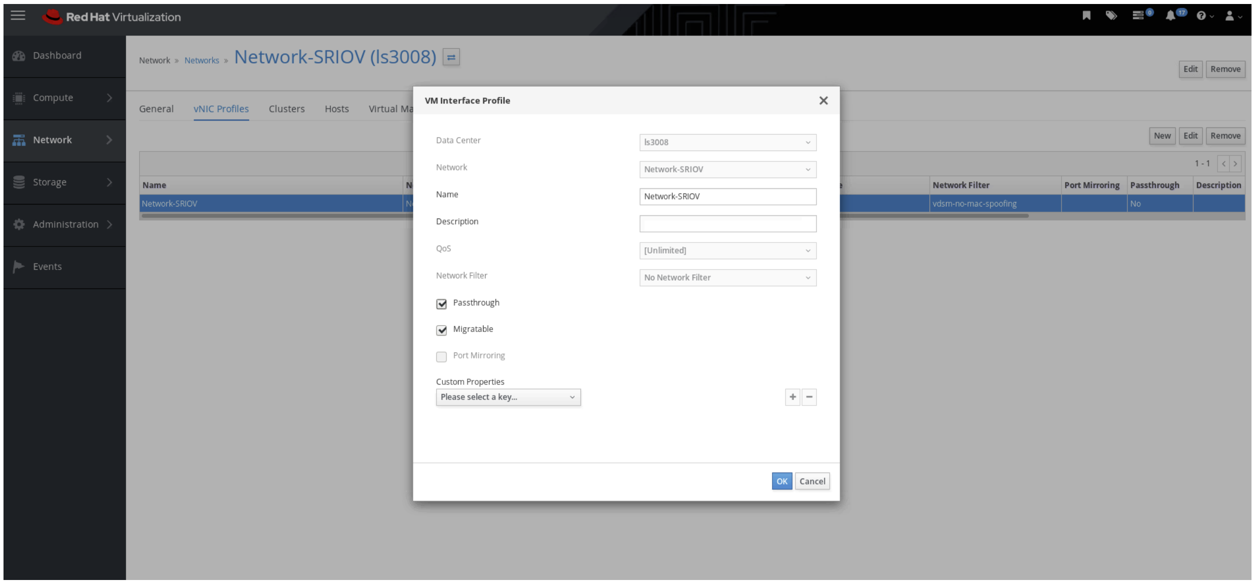
Procedure

1. In RHV Manager, click the **Networks** tab.
2. Click **New** to create a new virtual network.
3. Enter the network name and select the relevant **Data Center**.
4. Modify MTU according to requirements.
5. Select **VM-Network**.
6. Click **OK**.



7. From the Network table, select the virtual network that you created.
8. On the Network overview page, select the **vNIC Profiles** tab.

9. Highlight the **vNIC Profile** that was created with your **Network** and select Edit.



9. Set **No Network Filter**.
10. Select **Passthrough**.
11. Select **Migratable**.
12. Click **OK**.

Attaching the created network to the network interface card

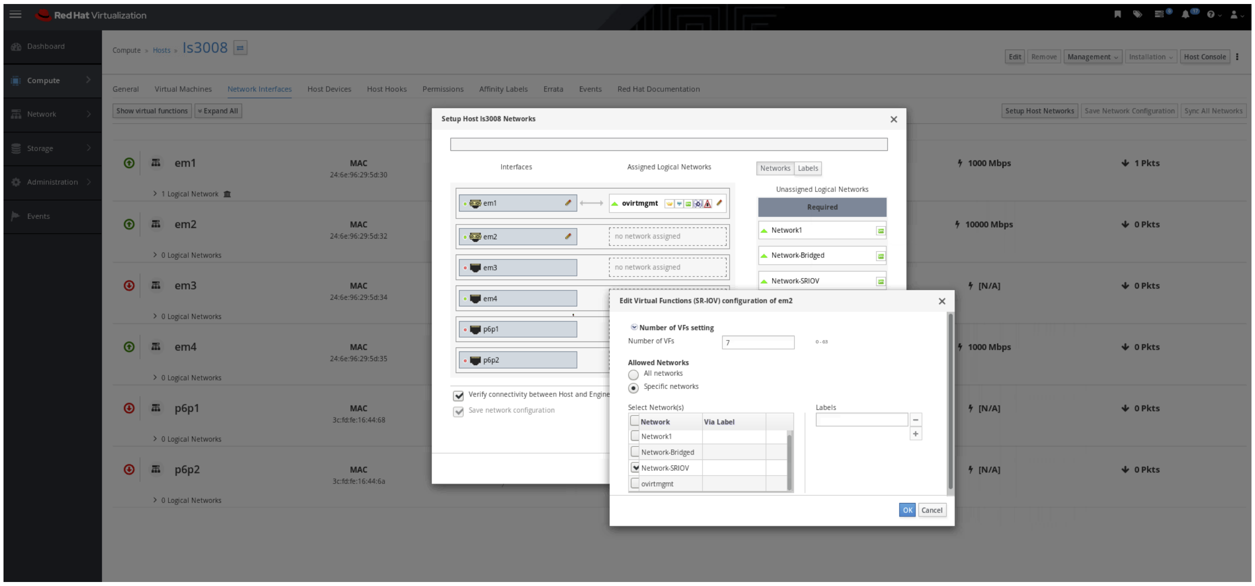
After the virtual network has been created, it needs to be added to a device. Complete the following procedure to create virtual functions on the network card where the new virtual network is attached.

Procedure

1. In RHV Manager, click **Compute** > **Hosts**.
2. Select the **SAP HANA** host.
3. On the host overview page, select the **Network Interfaces** tab.
4. Click **Setup Hosts Networks**.
5. Click the pencil icon for the required SR-IOV network card under the **Interfaces** section.

NOTE: SR-IOV-capable network-cards have an **SR-IOV** icon next to them.

6. For **Number of VFs**, enter the number of required virtual functions. Choose the network for this Interface in the **Select Network(s)** list.



7. Click **OK**.

8. Click **OK** when you are finished setting up the network for the host.

Adding the virtual function to the VM

After the virtual network for SR-IOV has been assigned to the host, you must complete the following procedure to add the virtual function to the VM.

Procedure

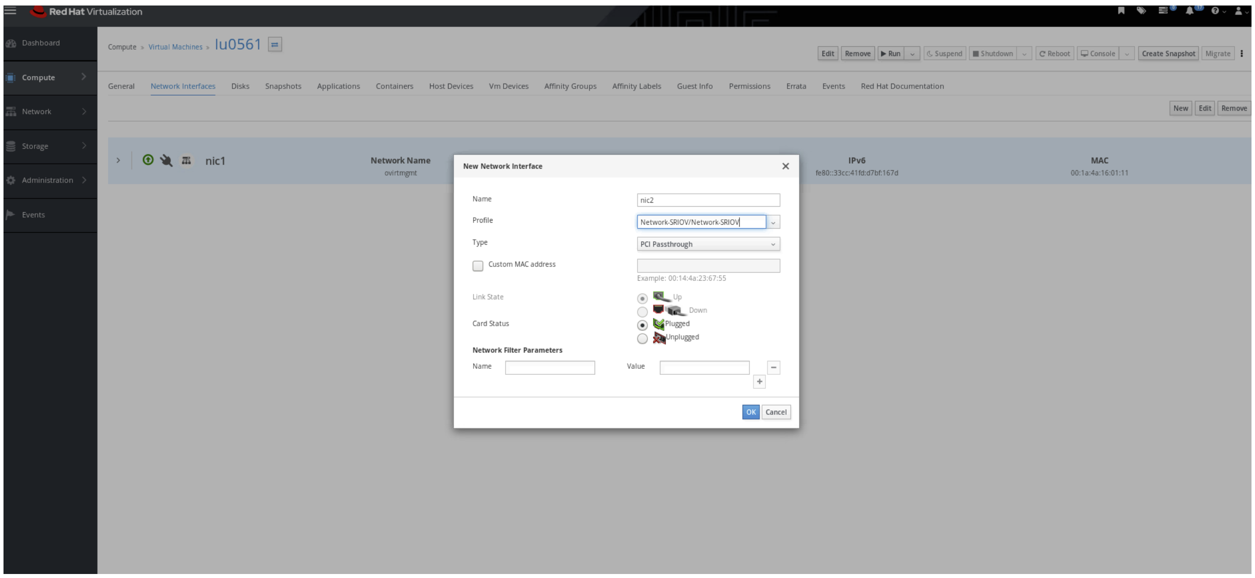
1. In RHV Manager, click **Compute** > **Virtual Machines**.
2. Select the previously created SAP HANA virtual machine.
3. Click the **Network Interfaces** tab.
4. Click **New** to add a new interface.

5. In the new window, specify the following settings:

    a. **Profile**: The Virtual NIC Profile that you created.
    NOTE: The default profile for this network is also listed. Ensure that you select the profile you added earlier.

    b. **Type**: PCI-Passthrough

6. Click **OK.**



7. Repeat the procedure to add more virtual network interfaces to your environment.

After attaching the network, start the VM and install the Red Hat Enterprise Linux operating system.

Create a bridged network

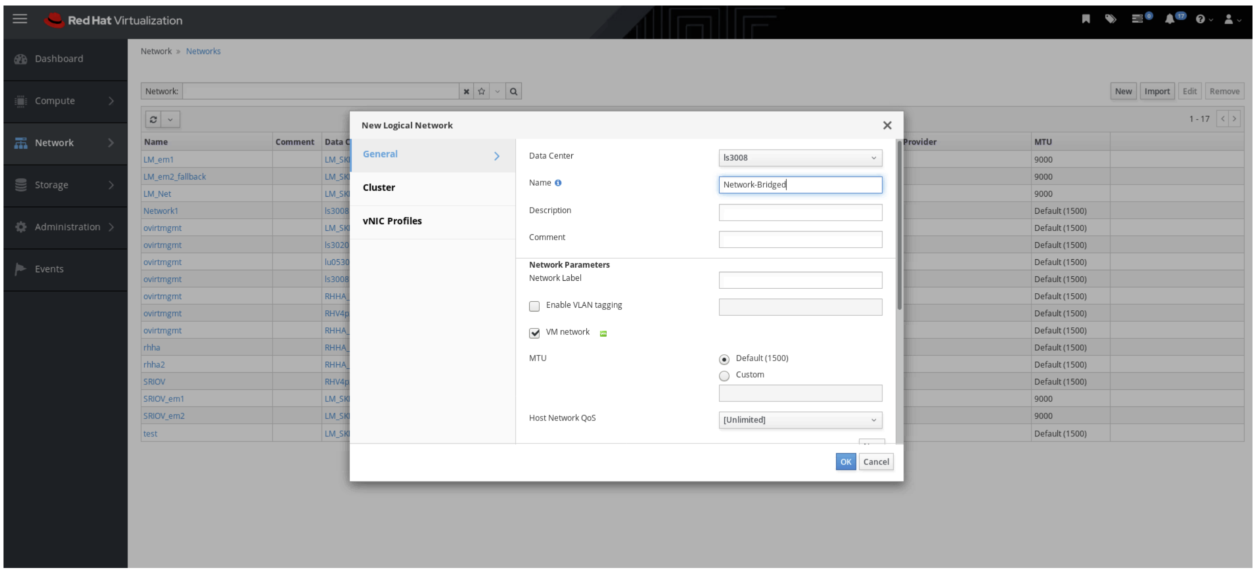
Bridged networking (also known as network bridging or virtual network switching) is used to place virtual machine network interfaces on the same network as the physical interface. Bridges require minimal configuration and make a virtual machine appear on an existing network, which reduces management overhead and network complexity.

As bridges contain few components and configuration variables, they provide a transparent setup which is straightforward to understand and troubleshoot, if required.

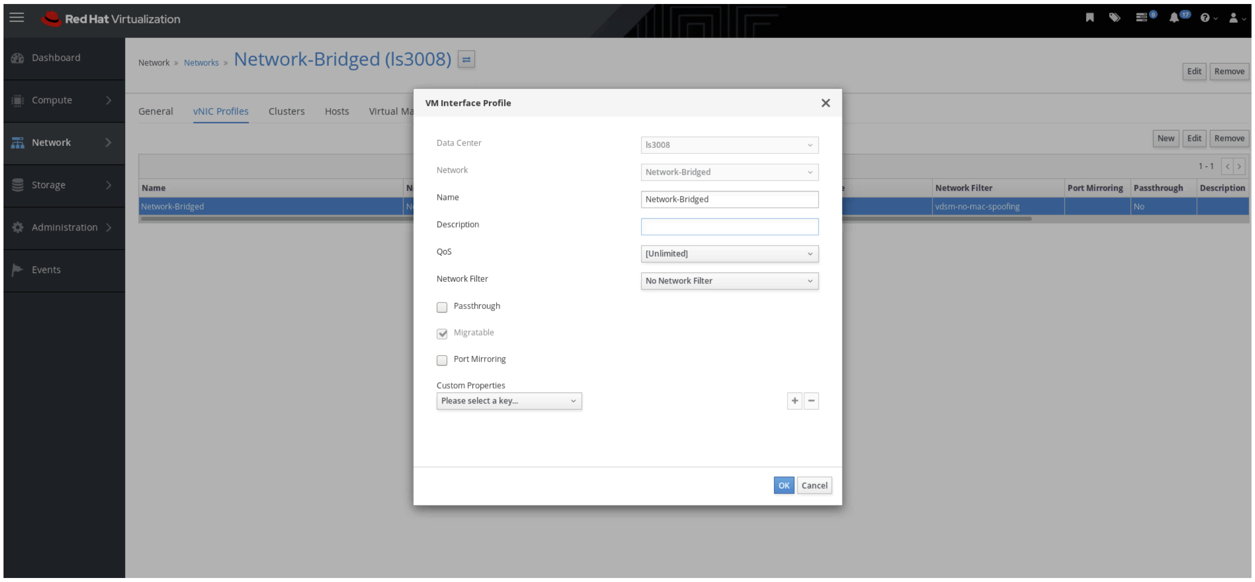
Note: Also for live migration, configuring a bridged network can be used as a fallback network for the SR-IOV client network.

Procedure

1. In RHV Manager, click the **Networks** tab.
2. Click **New** to create a new virtual network.
3. Enter the network name and select the relevant **Data Center**.
4. Modify MTU according to requirements.
5. Select **VM-Network**.
6. Click **OK**.



7. From the **Network** table, select the virtual network that you created.
8. In the left navigation bar, click **Network** > On the Network overview page, select the **vNIC Profiles** tab.
9. Highlight the **vNIC Profile** that was created with your **Network** and select **Edit**.
10. Set **No Network Filter**.
11. Select **Migratable**.
12. Click **OK**.

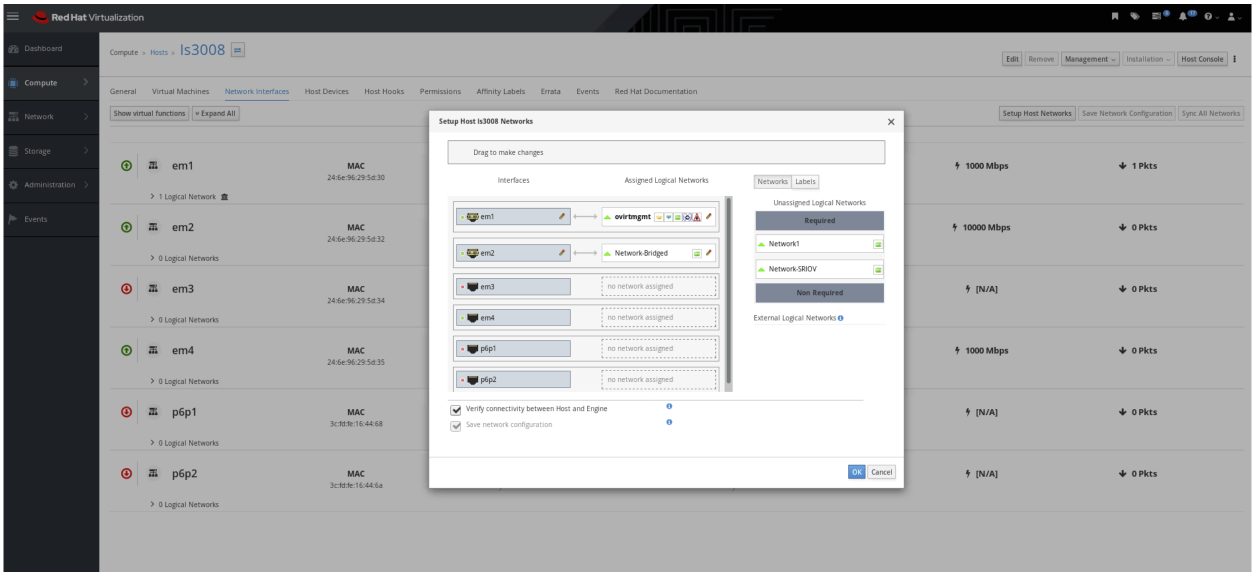


Attaching the created virtual network to the network interface card

After the virtual network has been created, it needs to be added to a device. Complete the following procedure to link the new virtual network to the required network card.

Procedure

1. In RHV Manager, click **Compute** > **Hosts**.
2. Select the **SAP HANA** host.
3. On the host overview page, select the **Network Interfaces** tab.
4. Click **Setup Hosts Networks**.
5. Drag that bridged network to the **Assigned Logical Networks**.



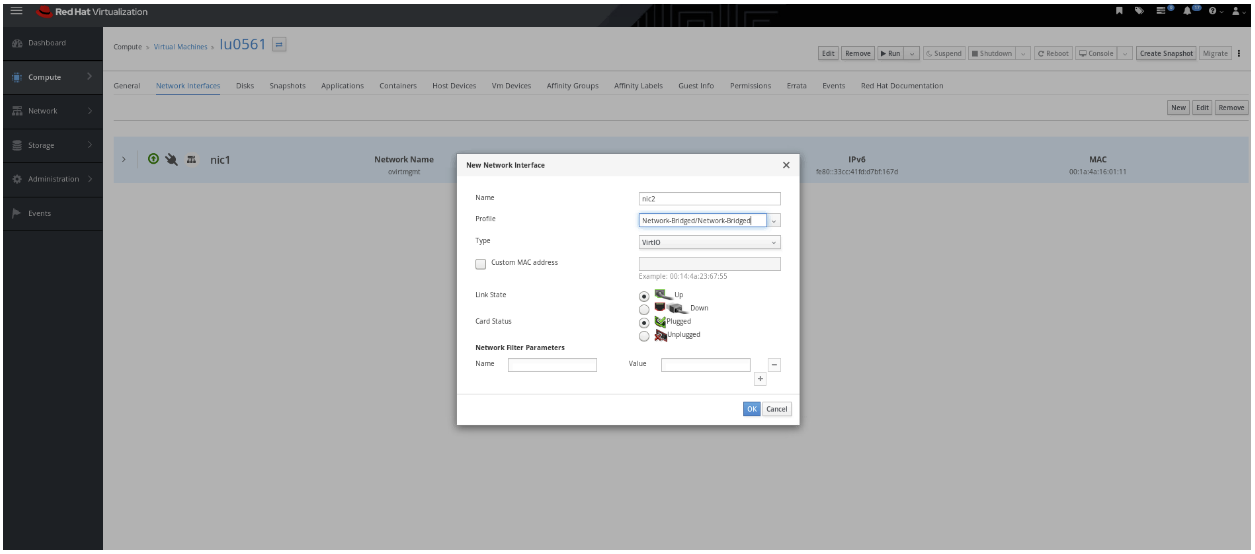
6. Click **OK** when you are finished setting up the network for the host.

Adding the virtual network to the VM

After the virtual network has been linked to the host, you must complete the following procedure to add the virtual network to the VM.

Procedure

8. In RHV Manager, click **Compute** > **Virtual Machines**.
9. Select the previously created SAP HANA virtual machine.
10. Click the **Network Interfaces** tab.
11. Click **New** to add a new interface.
12. In the window that opens, specify the following settings:
    a. **Profile**: The Virtual NIC Profile that you created.
    NOTE: The default profile for this network is also listed. Ensure that you select the profile you added earlier.
    b. **Type**: virtIO
13. Click **OK.**



14. Repeat the procedure to add more virtual network interfaces to your environment.

After attaching the network, start the VM and install the Red Hat Enterprise Linux operating system.

8. Configuring Live Migration

Live migrations allow a running virtual machine to move from one physical host to another. In the context of SAP HANA, an NFS Storage is required for live migration.

See [Migrating Virtual Machines Between Hosts] on the Red Hat Customer Portal for more information.

Preparations

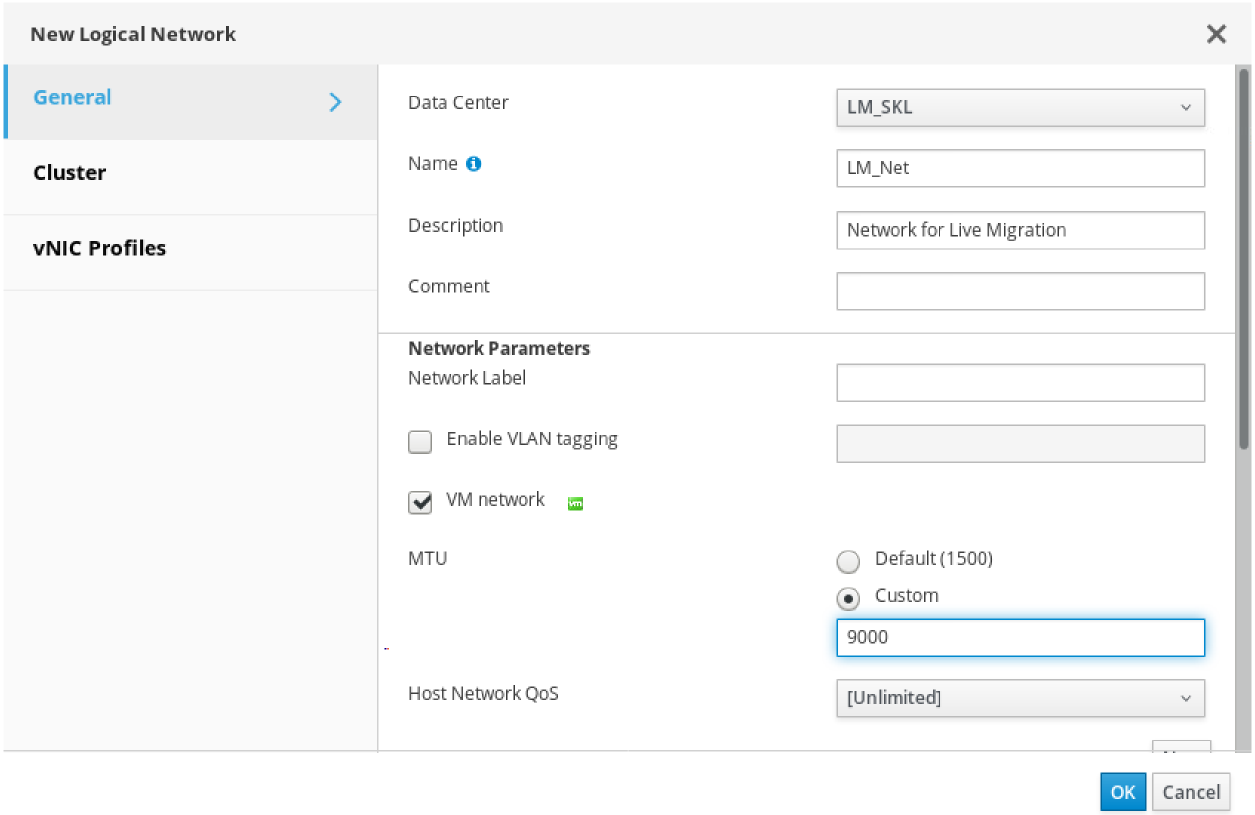
Before you begin the live migration on your landscape, make sure you follow these steps. Please note that the process described here is the only officially supported method endorsed by Red Hat. All other methods are currently untested and unsupported.

Network requirements

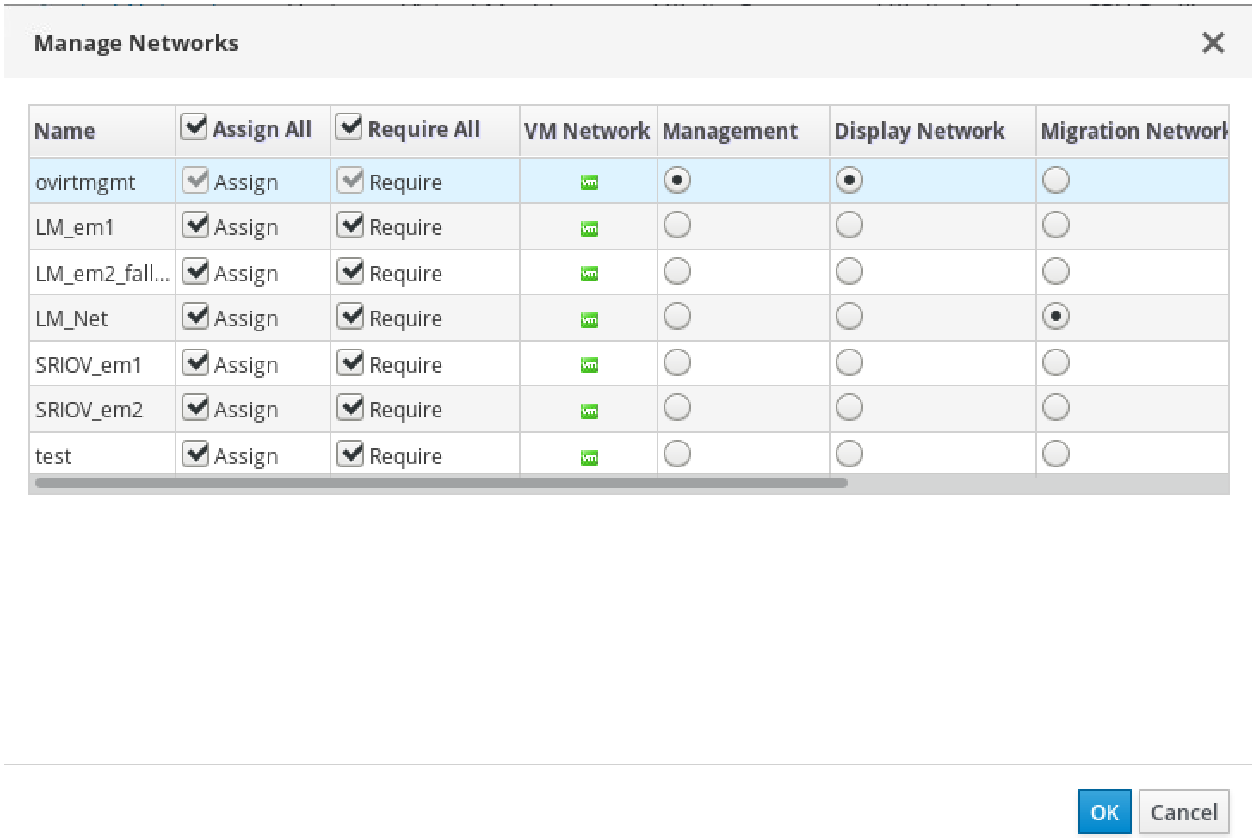
There is at least one dedicated 10G link required for the live migration network. When SR-OIV is used for the guest network interfaces, this workaround is needed as of now in order to ensure connectivity during the migration. See the section "Configuring Virtual Machines with SR-IOV-Enabled vNICs to Reduce Network Outage during Migration" here: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.3/html/virtual_machine_management_guide/sect-migrating_virtual_machines_between_hosts

Create Migration Network

In RHV-M go to **Network** → **Networks** and click on **New** to add a network. Set MTU to 9000, see screenshot.



At **Compute → Clusters** click on the name of your cluster so that it opens. Then navigate to the tab **Logical Networks** and click **Manage Networks**. There select the above created network (here: “LM_Net”) as **Migration Network**, see screenshot.



Migration settings

The settings used for live migration are pre-copy with auto converge turned off. In order to set pre-copy cluster wide, go to **Compute** → **Clusters** → Select your cluster and click edit.

1. Then in the tab **Migration Policy** select **Minimal Downtime** as **Migration Policy**.
2. Set a custom **Migration bandwidth limit** which suits your migration scenario. Note that the unit is in Megabits per second (Mbps) and the value entered specifies in- and out-going traffic accumulated. See formula below.

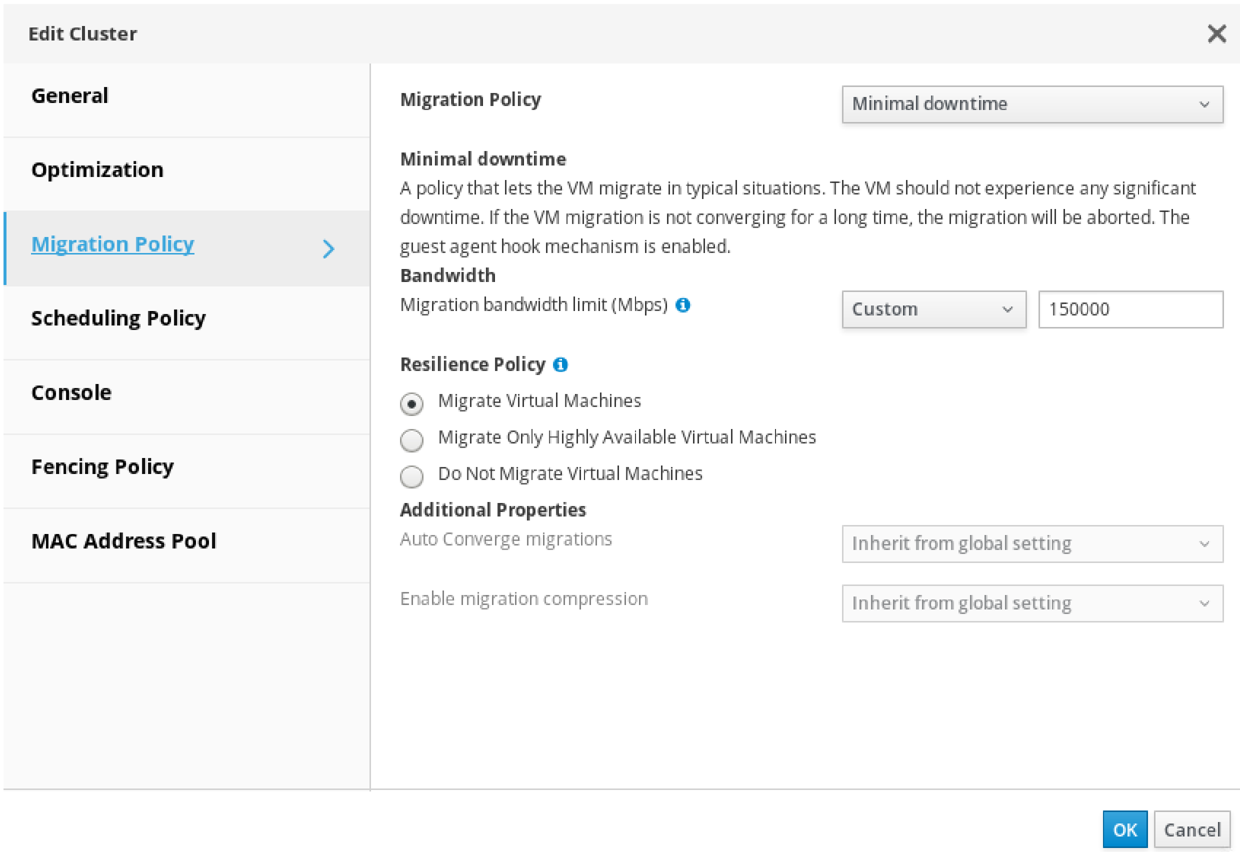
Screenshot below shows details for reference.

Custom migration bandwidth [Mbps] = Migration bandwidth in [Mbps] + Migration bandwidth out [Mbps]

Example: For a dedicated 10 GbE link the theoretical maximum bandwidth would be:

Custom migration bandwidth [Mbps] = 10.000 Mbps + 10.000 Mbps = 20.000 Mbps

Note that this bandwidth is per VM, so in case you want to migrate multiple VMs at the same time, choose a lower number according to your number of VMs. E.g. if you want to be able to migrate 5 VMs at the same time, the custom migration bandwidth should be set to 4.000 Mbps.



Next is setting auto convergence to off, see also this documentation section: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.3/html/virtual_machine_management_guide/sect-migrating_virtual_machines_between_hosts#Optimizing_Live_Migration

This has to be done on the RHV-M machine:

```
# engine-config -s DefaultAutoConvergence=False
```

And check with:

```
# engine-config -g DefaultAutoConvergence
```

There have been cases reported where the above was not sufficient to disable auto convergence. To disable auto convergence for the migration policy, follow this procedure:

Dump migration policies to a file (on RHV-M):

```
#  engine-config -g MigrationPolicies > migration_policy.json
```

Set in the migration policy file autoConvergence to false, below there is an example only shown for the RHV 4.3 migration policy to keep it compact here:


```
[{"id":{"uuid":"80554327-0569-496b-bdeb-fcbbf52b827b"},"maxMigrations
":2,"autoConvergence":false,"migrationCompression":false,"enableGuest
Events":true,"name":"Minimal downtime","description":"A policy that
lets the VM migrate in typical situations. The VM should not
experience any significant downtime. If the VM migration is not
converging for a long time, the migration will be aborted. The guest
agent hook mechanism is

enabled.","config":{"convergenceItems":[{"stallingLimit":1,"convergen
ceItem":{"action":"setDowntime","params":["150"]}},{"stallingLimit":2
,"convergenceItem":{"action":"setDowntime","params":["200"]}},{"stall
ingLimit":3,"convergenceItem":{"action":"setDowntime","params":["300"
]}},{"stallingLimit":4,"convergenceItem":{"action":"setDowntime","par
ams":["400"]}},{"stallingLimit":6,"convergenceItem":{"action":"setDow
ntime","params":["500"]}}],"initialItems":[{"action":"setDowntime","p
arams":["100"]}],"lastItems":[{"action":"abort","params":[]}]}},{"id"
:{"uuid":"80554327-0569-496b-bdeb-fcbbf52b827c"},"maxMigrations":1,"a
utoConvergence":false,"migrationCompression":true,"enableGuestEvents"
:true,"name":"Suspend workload if needed","description":"A policy
that lets the VM migrate in most situations, including VMs running
heavy workloads. On the other hand, the VM may experience a more
significant downtime. The migration may still be aborted for extreme
workloads. The guest agent hook mechanism is
enabled.","config":{"convergenceItems":[{"stallingLimit":1,"convergen
ceItem":{"action":"setDowntime","params":["150"]}},{"stallingLimit":2
,"convergenceItem":{"action":"setDowntime","params":["200"]}},{"stall
ingLimit":3,"convergenceItem":{"action":"setDowntime","params":["300"
]}},{"stallingLimit":4,"convergenceItem":{"action":"setDowntime","par
ams":["400"]}},{"stallingLimit":6,"convergenceItem":{"action":"setDow
ntime","params":["500"]}}],"initialItems":[{"action":"setDowntime","p
arams":["100"]}],"lastItems":[{"action":"setDowntime","params":["5000
"]},{"action":"abort","params":[]}]}},{"id":{"uuid":"a7aeedb2-8d66-4e
51-bb22-32595027ce71"},"maxMigrations":2,"autoConvergence":false,"mig
```

rationCompression":false,"enableGuestEvents":true,"name":"Post-copy migration","description":"The VM should not experience any significant downtime. If the VM migration is not converging for a long time, the migration will be switched to post-copy. The guest agent hook mechanism is enabled.","config":{"convergenceItems":[{"stallingLimit":1,"convergenceItem":{"action":"setDowntime","params":["150"]}},{"stallingLimit":2,"convergenceItem":{"action":"setDowntime","params":["200"]}}],"initialItems":[{"action":"setDowntime","params":["100"]}],"lastItems":[{"action":"postcopy","params":[]},{"action":"abort","params":[]}]}}]

Set the modified migration policy:

```
# MIGRATION=$(cat migration_policy.json)
# engine-config -s MigrationPolicies="$MIGRATION"
```

Activate:

```
# service ovirt-engine restart
```

Set minimal TSC clock frequency

Determine the smallest TSC clocksource frequency in your cluster. Execute the following command on all RHV hypervisors, e.g.:

```
# dmesg | grep "tsc: Refined TSC clocksource calibration"
```

```
[6.037158] tsc: Refined TSC clocksource calibration: 2494.140 MHz
```

Choose the smallest value among all your hosts. For the following setting, this frequency is needed in the unit Hertz [HZ] so make sure to convert it accordingly. See the numbers printed here as an example.

Modify the SAP HANA hook `/usr/libexec/vdsm/hooks/before_vm_start/50_hana` on all RHV hypervisors and make sure to uncomment / add the following lines printed in **bold red** letters in the clock section:

```
    if len(domain.getElementsByTagName('clock')):
        clock = domain.getElementsByTagName('clock')[0]
        tscClock = domxml.createElement('clock')
        tscClock.setAttribute('offset', 'utc')
        timer = domxml.createElement('timer')
        timer.setAttribute('name','tsc')
```

```
          # Uncomment and adjust for live migration (adjust
frequency to match the lowest value in your cluster)↲
          timer.setAttribute('frequency','2494140000')
          tscClock.appendChild(timer)
          domain.removeChild(clock)
          domain.appendChild(tscClock)
```

The actual migration

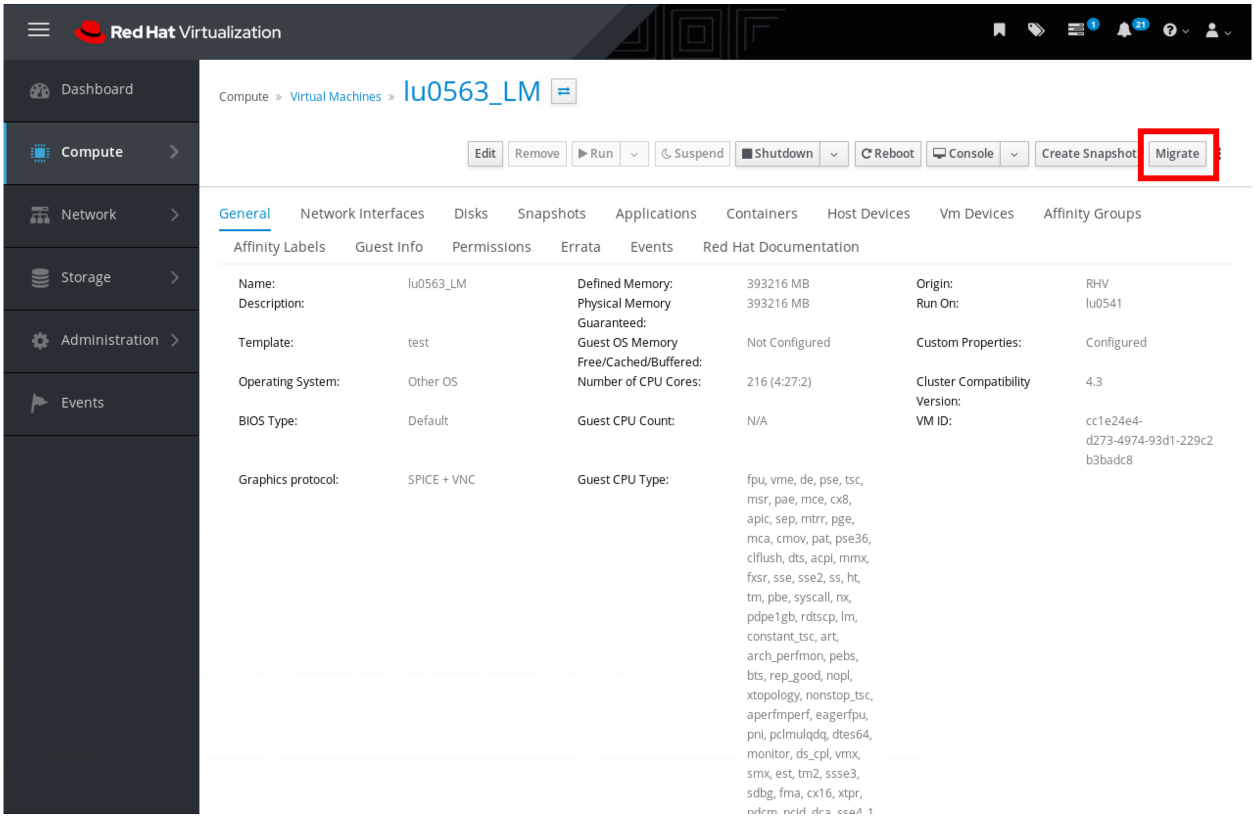
This section describes the steps for the actual migration. Note the VM specific migration options in the VM settings on the **Host** tab.

Disabling the NUMA pinning during migration

During a migration the strict NUMA pinning has to be removed. Navigate to the VM and edit its settings. Go to the **Host** tab scroll down and click on **NUMA Pinning**. Remove the mapping between the virtual NUMA nodes and the physical NUMA nodes. You might want to note the actual mapping down or take a screenshot since you want to restore that after the migration has been done.

Triggering the actual live migration

Note the **Migrate** button on the top right corner of the screenshot shown below. There is also the possibility to use the context menu showing up when right clicking the VM in the Virtual Machines Overview. There is also an API available that allows to trigger migration, e.g. check ovirt-engine-sdk-python.



9. Installing and configuring Red Hat Enterprise Linux Guest OS

This chapter describes how to configure and optimize a Red Hat Enterprise Linux guest on RHV for SAP HANA.

Installing SAP HANA on Red Hat Enterprise Linux

Review the required documentation before starting an SAP HANA deployment. The documentation contains information about supportability, configuration, recommended operating system settings, and guidelines for running SAP HANA on Red Hat Enterprise Linux.

Related SAP Notes

- SAP Note 2009879 – SAP HANA guidelines for Red Hat Enterprise Linux Operating System. A link to the Installation Guide is included.

- SAP Note 2292690 - SAP HANA DB: Recommended OS settings for Red Hat Enterprise Linux 7.x
- SAP Note 1943937 - Hardware Configuration Check Tool - Central Note and it contains the user guide for HWCCT.
- SAP Note 1788665 - SAP HANA Support for virtualized / partitioned (multi-tenant) environments

Related Red Hat KnowledgeBase articles

- Overview of the Red Hat Enterprise Linux for SAP Solutions subscription
- Red Hat Enterprise Linux for SAP HANA: system updates and supportability
- How to subscribe to SAP HANA systems to the Update Services for SAP Solutions
- RHEL 7.2 and higher: How to subscribe a RHEL 7 system to RHEL for SAP HANA child channel?
- How to subscribe RHEL 7 SAP HANA system to Extended Update Support (EUS) channel?

Optimizing performance for SAP HANA running on a guest

To ensure optimal integration to RHV of the SAP HANA VM, install and enable the `qemu-guest-agent` from the `rhel-7-server-rpms` repository, which should be enabled by default.

Procedure

1. Run the command:

```
# yum install -y qemu-guest-agent
# systemctl start qemu-guest-agent
```

Activating the tuned profile on an SAP HANA VM

When you install Red Hat Enterprise Linux 7 in an RHV guest, the virtual-guest profile is automatically selected as it is recommended for virtual machines. For SAP HANA on KVM, use the `sap-hana-kvm-guest` profile. The files in this procedure are in the zip file that is available with this document on the Red Hat Customer Portal.

1. Create `/usr/lib/tuned/sap-hana-kvm-guest/tuned.conf:`

```
#
# tuned configuration
#

[main]
summary=Optimize for running SAP HANA on KVM inside a virtual guest
include=sap-hana

[haltpoll]
type=script
script=${i:PROFILE_DIR}/haltpoll.sh

[sysfs]
/sys/devices/system/clocksource/clocksource0/current_clocksource=ts
c
```

2. Create `/usr/lib/tuned/sap-hana-kvm-guest/haltpoll.sh:`

```
#!/bin/bash
```

```
guest_halt_poll_ns=800000
guest_halt_poll_grow_start=200000

if [ "$1" == "start" ]; then
      modprobe cpuidle-haltpoll
      if [ -e /sys/module/cpuidle_haltpoll/parameters/ ]; then
      echo $guest_halt_poll_ns >
/sys/module/cpuidle_haltpoll/parameters/guest_halt_poll_ns
      echo $guest_halt_poll_grow_start >
/sys/module/cpuidle_haltpoll/parameters/guest_halt_poll_grow_start
      elif [ -e /sys/module/haltpoll/parameters/ ]; then
      echo $guest_halt_poll_ns >
/sys/module/haltpoll/parameters/guest_halt_poll_ns
      echo $guest_halt_poll_grow_start >
/sys/module/haltpoll/parameters/guest_halt_poll_grow_start
      fi
fi
```

3. Make the script executable by entering the command:

```
# chmod +x /usr/lib/tuned/sap-hana-kvm-guest/haltpoll.sh
```

4. To enable this profile, enter the command:

```
tuned-adm profile sap-hana-kvm-guest
```

For more information about tuned profiles, see the tuned and tuned-adm section of the Virtualization Tuning and Administration Guide.

Check that haltpoll driver is enabled

Complete this procedure to ensure that the CPU haltpoll driver is available in the guest.

Procedure

1. Enter the command:

```
# lsmod | grep cpuidle_haltpoll
```

   The command output indicates that the module is loaded, for example:

```
cpuidle_haltpoll        12511   54
```

2. Verify the interface is in place by entering:

```
# ls /sys/module/haltpoll/parameters/
guest_halt_poll_allow_shrink  guest_halt_poll_grow
```

```
guest_halt_poll_grow_start  guest_halt_poll_ns  guest_halt_poll_shrink
```

3. Optional: For earlier kernel versions, the parameters can be found here:
```
# ls /sys/module/cpuidle_haltpoll/parameters
```

4. Verify the values are set by entering:
```
# cat /sys/module/haltpoll/parameters/guest_halt_poll_ns
800000
# cat /sys/module/haltpoll/parameters/guest_halt_poll_grow_start
200000
```

Disable seccomp sandbox

Edit `/etc/libvirt/qemu.conf` and make sure that

```
seccomp_sandbox = 0
```

is set by searching for that line and modifying it. Restart libvirtd and any qemu processes.

Updating and configuring the kernel

The minimal kernel version supported is `kernel-3.10.0-957.36.1.el7`. It is recommended that you use the latest kernel available at Red Hat update channels.

Add the following parameters, separated by spaces, to the kernel command line:

```
selinux=0
```

In case of using a Skylake CPU, add `spectre_v2=retpoline` above.

In order to make this permanent, append the above to the `GRUB_CMDLINE_LINUX` variable in `/etc/default/grub`. Then recreate the initrd with:

```
# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```

After a reboot, the settings have to appear on the kernel command line:

```
# cat /proc/cmdline
```

Verifying RHV Host/KVM guest timing management

SAP HANA performance benefits from the use of the RDTSC hardware timer. Complete this procedure to verify that RDTSC is used.

Procedure

1. Switch to the following directory:

   ```
   /hana/shared/<SID>/HDB<Instance Nr.>/<systemname>/trace/DB_<TenantDBSID>/
   ```

2. Check the indexserver trace file for the timer:

   ```
   # grep Timer.cpp indexserver_<system name>.<30003>.<001>.trc
   ```

3. If the hardware timer is in use, the grep output displays a message that is similar to the following example:

   ```
   [4548]{-1}[-1/-1] 2017-03-16 08:43:36.137096 i Basis Timer.cpp(00642) :
   Using RDTSC for HR timer
   ```

4. If the hardware timer is not in use, the log indicates that a software fallback is being used, resulting in a significant performance impact:

   ```
   [4330]{-1}[-1/-1] 2017-04-18 10:50:59.068073 w Basis Timer.cpp(00718) :
   Fallback to system call for HR timer
   ```

For more information about guest timing management, see the KVM Guest Timing Management section of the Virtualization Deployment and Administration Guide.

Verifying the CPU/NUMA settings

Complete this procedure to verify the vCPU/vNUMA topology.

Procedure

1. Run the following command on the RHV host and the SAP HANA guest:

   ```
   # lscpu
   ```

   On the host, the `lscpu` output is similar to this example:

```
# lscpu
Architecture:            x86_64
CPU op-mode(s):          32-bit, 64-bit
Byte Order:        Little Endian
CPU(s):                  144
On-line CPU(s) list:     0-143
Thread(s) per core:      2
Core(s) per socket:      18
Socket(s):         4
NUMA node(s):            4
[...]

L1d cache:         32K
L1i cache:         32K
L2 cache:                256K
L3 cache:                46080K
NUMA node0 CPU(s):       0-17,72-89
NUMA node1 CPU(s):       18-35,90-107
NUMA node2 CPU(s):       36-53,108-125
NUMA node3 CPU(s):       54-71,126-143
[...]
```

On the guest, the `lscpu` output is similar to this example:

```
# lscpu
Architecture:                  x86_64
CPU op-mode(s):                32-bit, 64-bit
Byte Order:              Little Endian
CPU(s):                        136
On-line CPU(s) list:           0-135
Thread(s) per core:            2
Core(s) per socket:            17 (NOTE: 1 core per socket is used
                                for IO, admin)
Socket(s):               4
NUMA node(s):                  4
[...]

L1d cache:               32K
L1i cache:               32K
L2 cache:                      4096K
L3 cache:                      16384K
NUMA node0 CPU(s):             0-33
NUMA node1 CPU(s):             34-67
NUMA node2 CPU(s):             68-101
NUMA node3 CPU(s):             102-135
```

Check the following values from the output:

1. **CPU(s)** in the VM must be lower or equal to the host. In this example 136<144.
2. **Thread(s) per Core** must be identical. In this case 2=2.
3. **Core(s) per socket** in the VM must be lower or equal to the host. In this example 17<18.
4. **Socket(s)** must be identical. In this case 4=4.
5. **NUMA node(s)** must be identical. In this case 4=4.
6. **L3 cache** must be present in the VM. In this example 16384K.
7. **NUMA node# CPU(s)** need to match the CPU Pinning completed earlier. Each vCPU pinned to a physical CPU must reside in the same NUMA node. In this example, vCPUs #0-33 are pinned to physical CPUs #1-17 and #73-89.

   If the CPUs are not pinned, recheck the NUMA and CPU pinning outlined in the [About Virtual NUMA and Virtual CPU Pinning] section in this document.

10. Appendices

This section contains appendices for the following topics:

- Calculate CPU Pinning
- Network Configuration and Monitoring
- Example libvirt XML file for an SAP HANA VM running on an RHV host
- Failing HCMT Subtest: System Process Tree Microcode Validation
- Virtualization limits for RHV

Calculate CPU Pinning

This script generates the appropriate vCPU pinning. Copy and paste the output of this `calculate_cpu_pinning.sh` script into the CPU pinning field in the RHV Manager **Resource Allocation** tab.

```
#!/bin/bash

usage() {
cat << EOR
Usage: $0 [num_vcpu]

This script creates CPU pinning according to the host CPU topology up to
the max number of CPUs and always leaving cpu core0 and according threads
to the host for each numa node

EOR
exit 1
}

num_sockets=$(lscpu | awk -F: '/^Socket\(s\)/ { print $2}' | tr -d " ")
num_threads=$(lscpu | awk -F: '/^Thread\(s\) per core/ { print $2}'| tr
-d " ")
if [ -n "$1" ]; then
  max_cnt=$(( $1 / $num_sockets / $num_threads ));
  PARM="-v CNT=$max_cnt"
fi

numactl --hardware | awk -F: -v THREADS=$num_threads $PARM '
      BEGIN { num_threads=THREADS;
            vcpu=0
            if ( CNT ) { cnt = CNT;
                  print ("limiting to "cnt++" cpus/numa node. " );
```

```
                          }
                else {
                      cnt = 0;
                      print ("Using all cpus/numa node. " );
                }
                }
          /node [0-9]+ cpus/ {
           # split $2 by " " in array cores
           num_vcpu=split($2,threadnum," ");
           num_core=num_vcpu / num_threads;
           if ( cnt == 0 ) { cnt=num_core }
           for (i=2;i<=cnt ;i++) {
                 for (t=0;t<num_threads ;t++) {
                       printf
(vcpu++"#"threadnum[i]","threadnum[i+num_core]"_");
                 }
           }
                }
      END { printf("\n"); }' |  sed 's/.$//'
```

Example: All CPUs in use for SAP HANA VM:

```
# ./print_cpu_pinning
Using all cpus/numa node.
0#4,116_1#4,116_2#8,120_3#8,120_4#12,124_5#12,124_6#16,128_7#16,128_8#20,132_
9#20,132_10#24,136_11#24,136_12#28,140_13#28,140_14#32,144_15#32,144_16#36,14
8_17#36,148_18#40,152_19#40,152_20#44,156_21#44,156_22#48,160_23#48,160_24#52
,164_25#52,164_26#56,168_27#56,168_28#60,172_29#60,172_30#64,176_31#64,176_32
#68,180_33#68,180_34#72,184_35#72,184_36#76,188_37#76,188_38#80,192_39#80,192
_40#84,196_41#84,196_42#88,200_43#88,200_44#92,204_45#92,204_46#96,208_47#96,
208_48#100,212_49#100,212_50#104,216_51#104,216_52#108,220_53#108,220_54#5,11
7_55#5,117_56#9,121_57#9,121_58#13,125_59#13,125_60#17,129_61#17,129_62#21,13
3_63#21,133_64#25,137_65#25,137_66#29,141_67#29,141_68#33,145_69#33,145_70#37
,149_71#37,149_72#41,153_73#41,153_74#45,157_75#45,157_76#49,161_77#49,161_78
#53,165_79#53,165_80#57,169_81#57,169_82#61,173_83#61,173_84#65,177_85#65,177
_86#69,181_87#69,181_88#73,185_89#73,185_90#77,189_91#77,189_92#81,193_93#81,
193_94#85,197_95#85,197_96#89,201_97#89,201_98#93,205_99#93,205_100#97,209_10
1#97,209_102#101,213_103#101,213_104#105,217_105#105,217_106#109,221_107#109,
221_108#6,118_109#6,118_110#10,122_111#10,122_112#14,126_113#14,126_114#18,13
0_115#18,130_116#22,134_117#22,134_118#26,138_119#26,138_120#30,142_121#30,14
2_122#34,146_123#34,146_124#38,150_125#38,150_126#42,154_127#42,154_128#46,15
8_129#46,158_130#50,162_131#50,162_132#54,166_133#54,166_134#58,170_135#58,17
0_136#62,174_137#62,174_138#66,178_139#66,178_140#70,182_141#70,182_142#74,18
6_143#74,186_144#78,190_145#78,190_146#82,194_147#82,194_148#86,198_149#86,19
8_150#90,202_151#90,202_152#94,206_153#94,206_154#98,210_155#98,210_156#102,2
```

```
14_157#102,214_158#106,218_159#106,218_160#110,222_161#110,222_162#7,119_163#
7,119_164#11,123_165#11,123_166#15,127_167#15,127_168#19,131_169#19,131_170#2
3,135_171#23,135_172#27,139_173#27,139_174#31,143_175#31,143_176#35,147_177#3
5,147_178#39,151_179#39,151_180#43,155_181#43,155_182#47,159_183#47,159_184#5
1,163_185#51,163_186#55,167_187#55,167_188#59,171_189#59,171_190#63,175_191#6
3,175_192#67,179_193#67,179_194#71,183_195#71,183_196#75,187_197#75,187_198#7
9,191_199#79,191_200#83,195_201#83,195_202#87,199_203#87,199_204#91,203_205#9
1,203_206#95,207_207#95,207_208#99,211_209#99,211_210#103,215_211#103,215_212
#107,219_213#107,219_214#111,223_215#111,223
```

This example uses 32 vCPUs. The RHV host needs to have at least 32 physical cores:

```
./print_cpu_pinning 32
limiting to 4 cpus/numa node.
0#4,116_1#4,116_2#8,120_3#8,120_4#12,124_5#12,124_6#16,128_7#16,128_8#5,1
17_9#5,117_10#9,121_11#9,121_12#13,125_13#13,125_14#17,129_15#17,129_16#6
,118_17#6,118_18#10,122_19#10,122_20#14,126_21#14,126_22#18,130_23#18,130
_24#7,119_25#7,119_26#11,123_27#11,123_28#15,127_29#15,127_30#19,131_31#1
9,131
```

NOTE: The first physical CPU core of each physical socket is not pinned, and as a result, it is available for the host and the iothreads.

Network Configuration and Monitoring

Network configuration can impact overall system performance. Poor network performance causes increased latency between the application servers and the HANA database server. As a result, it can result in poor response time and a longer time to complete a given transaction.

For example, overall system performance can be affected if the physical distance between the database server and application servers is too far or if there are too many hops between these systems. Poor performance can also result from other network traffic consuming bandwidth needed by SAP HANA.

To achieve consistently good performance, the network segment between the application servers and the database server should not exceed 50% of theoretical capacity. For example, a 1 Gbit NIC and related network segment should not exceed 500 Megabits per second. If the amount of network traffic exceeds that value, a more capable network, for example, 10 Gbit or more, is required for the best performance. Monitoring the total network traffic on a given network segment is typically done with the tools provided by the network switch vendor or a network sniffer.

One tool that can be used to monitor network traffic for your system is `sar.` Running the `sar -n DEV` command provides utilization data that helps you to determine whether the network traffic to your server is causing a performance bottleneck. Refer to the `sar man` page for a description of the fields in the `sar` command output.

Example libvirt XML file for SAP HANA VM running on an RHV host

This example illustrates a libvirt XML file generated by RHV for a 32 vCPU-based SAP HANA VM on a 64 CPU RHV host. This can be obtained by running `virsh -r dumpxml <VM-Name>` on the RHV host that is running the SAP HANA VM:

```
<domain type='kvm' id='1' xmlns:qemu='http://libvirt.org/schemas/domain/qemu/1.0'>
  <name>hana_i440fx_27c_ht</name>
  <uuid>05d342dc-9b48-4242-b8de-3b40a2ed4a76</uuid>
  <memory unit='KiB'>2977955840</memory>
  <currentMemory unit='KiB'>2977955840</currentMemory>
  <memoryBacking>
      <hugepages>
      <page size='1048576' unit='KiB'/>
      </hugepages>
  </memoryBacking>
  <vcpu placement='static'>216</vcpu>
  <iothreads>1</iothreads>
  <iothreadids>
      <iothread id='1'/>
  </iothreadids>
  <cputune>
      <vcpupin vcpu='0' cpuset='4,116'/>
      <vcpupin vcpu='1' cpuset='4,116'/>
      <vcpupin vcpu='2' cpuset='8,120'/>
      <vcpupin vcpu='3' cpuset='8,120'/>
      <vcpupin vcpu='4' cpuset='12,124'/>
      <vcpupin vcpu='5' cpuset='12,124'/>
      <vcpupin vcpu='6' cpuset='16,128'/>
      <vcpupin vcpu='7' cpuset='16,128'/>
      <vcpupin vcpu='8' cpuset='20,132'/>
      <vcpupin vcpu='9' cpuset='20,132'/>
      <vcpupin vcpu='10' cpuset='24,136'/>
      <vcpupin vcpu='11' cpuset='24,136'/>
      <vcpupin vcpu='12' cpuset='28,140'/>
      <vcpupin vcpu='13' cpuset='28,140'/>
      <vcpupin vcpu='14' cpuset='32,144'/>
      <vcpupin vcpu='15' cpuset='32,144'/>
      <vcpupin vcpu='16' cpuset='36,148'/>
      <vcpupin vcpu='17' cpuset='36,148'/>
      <vcpupin vcpu='18' cpuset='40,152'/>
      <vcpupin vcpu='19' cpuset='40,152'/>
      <vcpupin vcpu='20' cpuset='44,156'/>
```

```
<vcpupin vcpu='21' cpuset='44,156'/>
<vcpupin vcpu='22' cpuset='48,160'/>
<vcpupin vcpu='23' cpuset='48,160'/>
<vcpupin vcpu='24' cpuset='52,164'/>
<vcpupin vcpu='25' cpuset='52,164'/>
<vcpupin vcpu='26' cpuset='56,168'/>
<vcpupin vcpu='27' cpuset='56,168'/>
<vcpupin vcpu='28' cpuset='60,172'/>
<vcpupin vcpu='29' cpuset='60,172'/>
<vcpupin vcpu='30' cpuset='64,176'/>
<vcpupin vcpu='31' cpuset='64,176'/>
<vcpupin vcpu='32' cpuset='68,180'/>
<vcpupin vcpu='33' cpuset='68,180'/>
<vcpupin vcpu='34' cpuset='72,184'/>
<vcpupin vcpu='35' cpuset='72,184'/>
<vcpupin vcpu='36' cpuset='76,188'/>
<vcpupin vcpu='37' cpuset='76,188'/>
<vcpupin vcpu='38' cpuset='80,192'/>
<vcpupin vcpu='39' cpuset='80,192'/>
<vcpupin vcpu='40' cpuset='84,196'/>
<vcpupin vcpu='41' cpuset='84,196'/>
<vcpupin vcpu='42' cpuset='88,200'/>
<vcpupin vcpu='43' cpuset='88,200'/>
<vcpupin vcpu='44' cpuset='92,204'/>
<vcpupin vcpu='45' cpuset='92,204'/>
<vcpupin vcpu='46' cpuset='96,208'/>
<vcpupin vcpu='47' cpuset='96,208'/>
<vcpupin vcpu='48' cpuset='100,212'/>
<vcpupin vcpu='49' cpuset='100,212'/>
<vcpupin vcpu='50' cpuset='104,216'/>
<vcpupin vcpu='51' cpuset='104,216'/>
<vcpupin vcpu='52' cpuset='108,220'/>
<vcpupin vcpu='53' cpuset='108,220'/>
<vcpupin vcpu='54' cpuset='5,117'/>
<vcpupin vcpu='55' cpuset='5,117'/>
<vcpupin vcpu='56' cpuset='9,121'/>
<vcpupin vcpu='57' cpuset='9,121'/>
<vcpupin vcpu='58' cpuset='13,125'/>
<vcpupin vcpu='59' cpuset='13,125'/>
<vcpupin vcpu='60' cpuset='17,129'/>
<vcpupin vcpu='61' cpuset='17,129'/>
<vcpupin vcpu='62' cpuset='21,133'/>
<vcpupin vcpu='63' cpuset='21,133'/>
<vcpupin vcpu='64' cpuset='25,137'/>
<vcpupin vcpu='65' cpuset='25,137'/>
<vcpupin vcpu='66' cpuset='29,141'/>
```

```
<vcpupin vcpu='67' cpuset='29,141'/>
<vcpupin vcpu='68' cpuset='33,145'/>
<vcpupin vcpu='69' cpuset='33,145'/>
<vcpupin vcpu='70' cpuset='37,149'/>
<vcpupin vcpu='71' cpuset='37,149'/>
<vcpupin vcpu='72' cpuset='41,153'/>
<vcpupin vcpu='73' cpuset='41,153'/>
<vcpupin vcpu='74' cpuset='45,157'/>
<vcpupin vcpu='75' cpuset='45,157'/>
<vcpupin vcpu='76' cpuset='49,161'/>
<vcpupin vcpu='77' cpuset='49,161'/>
<vcpupin vcpu='78' cpuset='53,165'/>
<vcpupin vcpu='79' cpuset='53,165'/>
<vcpupin vcpu='80' cpuset='57,169'/>
<vcpupin vcpu='81' cpuset='57,169'/>
<vcpupin vcpu='82' cpuset='61,173'/>
<vcpupin vcpu='83' cpuset='61,173'/>
<vcpupin vcpu='84' cpuset='65,177'/>
<vcpupin vcpu='85' cpuset='65,177'/>
<vcpupin vcpu='86' cpuset='69,181'/>
<vcpupin vcpu='87' cpuset='69,181'/>
<vcpupin vcpu='88' cpuset='73,185'/>
<vcpupin vcpu='89' cpuset='73,185'/>
<vcpupin vcpu='90' cpuset='77,189'/>
<vcpupin vcpu='91' cpuset='77,189'/>
<vcpupin vcpu='92' cpuset='81,193'/>
<vcpupin vcpu='93' cpuset='81,193'/>
<vcpupin vcpu='94' cpuset='85,197'/>
<vcpupin vcpu='95' cpuset='85,197'/>
<vcpupin vcpu='96' cpuset='89,201'/>
<vcpupin vcpu='97' cpuset='89,201'/>
<vcpupin vcpu='98' cpuset='93,205'/>
<vcpupin vcpu='99' cpuset='93,205'/>
<vcpupin vcpu='100' cpuset='97,209'/>
<vcpupin vcpu='101' cpuset='97,209'/>
<vcpupin vcpu='102' cpuset='101,213'/>
<vcpupin vcpu='103' cpuset='101,213'/>
<vcpupin vcpu='104' cpuset='105,217'/>
<vcpupin vcpu='105' cpuset='105,217'/>
<vcpupin vcpu='106' cpuset='109,221'/>
<vcpupin vcpu='107' cpuset='109,221'/>
<vcpupin vcpu='108' cpuset='6,118'/>
<vcpupin vcpu='109' cpuset='6,118'/>
<vcpupin vcpu='110' cpuset='10,122'/>
<vcpupin vcpu='111' cpuset='10,122'/>
<vcpupin vcpu='112' cpuset='14,126'/>
```

```
<vcpupin vcpu='113' cpuset='14,126'/>
<vcpupin vcpu='114' cpuset='18,130'/>
<vcpupin vcpu='115' cpuset='18,130'/>
<vcpupin vcpu='116' cpuset='22,134'/>
<vcpupin vcpu='117' cpuset='22,134'/>
<vcpupin vcpu='118' cpuset='26,138'/>
<vcpupin vcpu='119' cpuset='26,138'/>
<vcpupin vcpu='120' cpuset='30,142'/>
<vcpupin vcpu='121' cpuset='30,142'/>
<vcpupin vcpu='122' cpuset='34,146'/>
<vcpupin vcpu='123' cpuset='34,146'/>
<vcpupin vcpu='124' cpuset='38,150'/>
<vcpupin vcpu='125' cpuset='38,150'/>
<vcpupin vcpu='126' cpuset='42,154'/>
<vcpupin vcpu='127' cpuset='42,154'/>
<vcpupin vcpu='128' cpuset='46,158'/>
<vcpupin vcpu='129' cpuset='46,158'/>
<vcpupin vcpu='130' cpuset='50,162'/>
<vcpupin vcpu='131' cpuset='50,162'/>
<vcpupin vcpu='132' cpuset='54,166'/>
<vcpupin vcpu='133' cpuset='54,166'/>
<vcpupin vcpu='134' cpuset='58,170'/>
<vcpupin vcpu='135' cpuset='58,170'/>
<vcpupin vcpu='136' cpuset='62,174'/>
<vcpupin vcpu='137' cpuset='62,174'/>
<vcpupin vcpu='138' cpuset='66,178'/>
<vcpupin vcpu='139' cpuset='66,178'/>
<vcpupin vcpu='140' cpuset='70,182'/>
<vcpupin vcpu='141' cpuset='70,182'/>
<vcpupin vcpu='142' cpuset='74,186'/>
<vcpupin vcpu='143' cpuset='74,186'/>
<vcpupin vcpu='144' cpuset='78,190'/>
<vcpupin vcpu='145' cpuset='78,190'/>
<vcpupin vcpu='146' cpuset='82,194'/>
<vcpupin vcpu='147' cpuset='82,194'/>
<vcpupin vcpu='148' cpuset='86,198'/>
<vcpupin vcpu='149' cpuset='86,198'/>
<vcpupin vcpu='150' cpuset='90,202'/>
<vcpupin vcpu='151' cpuset='90,202'/>
<vcpupin vcpu='152' cpuset='94,206'/>
<vcpupin vcpu='153' cpuset='94,206'/>
<vcpupin vcpu='154' cpuset='98,210'/>
<vcpupin vcpu='155' cpuset='98,210'/>
<vcpupin vcpu='156' cpuset='102,214'/>
<vcpupin vcpu='157' cpuset='102,214'/>
<vcpupin vcpu='158' cpuset='106,218'/>
```

```
<vcpupin vcpu='159' cpuset='106,218'/>
<vcpupin vcpu='160' cpuset='110,222'/>
<vcpupin vcpu='161' cpuset='110,222'/>
<vcpupin vcpu='162' cpuset='7,119'/>
<vcpupin vcpu='163' cpuset='7,119'/>
<vcpupin vcpu='164' cpuset='11,123'/>
<vcpupin vcpu='165' cpuset='11,123'/>
<vcpupin vcpu='166' cpuset='15,127'/>
<vcpupin vcpu='167' cpuset='15,127'/>
<vcpupin vcpu='168' cpuset='19,131'/>
<vcpupin vcpu='169' cpuset='19,131'/>
<vcpupin vcpu='170' cpuset='23,135'/>
<vcpupin vcpu='171' cpuset='23,135'/>
<vcpupin vcpu='172' cpuset='27,139'/>
<vcpupin vcpu='173' cpuset='27,139'/>
<vcpupin vcpu='174' cpuset='31,143'/>
<vcpupin vcpu='175' cpuset='31,143'/>
<vcpupin vcpu='176' cpuset='35,147'/>
<vcpupin vcpu='177' cpuset='35,147'/>
<vcpupin vcpu='178' cpuset='39,151'/>
<vcpupin vcpu='179' cpuset='39,151'/>
<vcpupin vcpu='180' cpuset='43,155'/>
<vcpupin vcpu='181' cpuset='43,155'/>
<vcpupin vcpu='182' cpuset='47,159'/>
<vcpupin vcpu='183' cpuset='47,159'/>
<vcpupin vcpu='184' cpuset='51,163'/>
<vcpupin vcpu='185' cpuset='51,163'/>
<vcpupin vcpu='186' cpuset='55,167'/>
<vcpupin vcpu='187' cpuset='55,167'/>
<vcpupin vcpu='188' cpuset='59,171'/>
<vcpupin vcpu='189' cpuset='59,171'/>
<vcpupin vcpu='190' cpuset='63,175'/>
<vcpupin vcpu='191' cpuset='63,175'/>
<vcpupin vcpu='192' cpuset='67,179'/>
<vcpupin vcpu='193' cpuset='67,179'/>
<vcpupin vcpu='194' cpuset='71,183'/>
<vcpupin vcpu='195' cpuset='71,183'/>
<vcpupin vcpu='196' cpuset='75,187'/>
<vcpupin vcpu='197' cpuset='75,187'/>
<vcpupin vcpu='198' cpuset='79,191'/>
<vcpupin vcpu='199' cpuset='79,191'/>
<vcpupin vcpu='200' cpuset='83,195'/>
<vcpupin vcpu='201' cpuset='83,195'/>
<vcpupin vcpu='202' cpuset='87,199'/>
<vcpupin vcpu='203' cpuset='87,199'/>
<vcpupin vcpu='204' cpuset='91,203'/>
```

```
        <vcpupin vcpu='205' cpuset='91,203'/>
        <vcpupin vcpu='206' cpuset='95,207'/>
        <vcpupin vcpu='207' cpuset='95,207'/>
        <vcpupin vcpu='208' cpuset='99,211'/>
        <vcpupin vcpu='209' cpuset='99,211'/>
        <vcpupin vcpu='210' cpuset='103,215'/>
        <vcpupin vcpu='211' cpuset='103,215'/>
        <vcpupin vcpu='212' cpuset='107,219'/>
        <vcpupin vcpu='213' cpuset='107,219'/>
        <vcpupin vcpu='214' cpuset='111,223'/>
        <vcpupin vcpu='215' cpuset='111,223'/>
        <emulatorpin cpuset='0,112'/>
        <iothreadpin iothread='1' cpuset='0,112'/>
    </cputune>
    <numatune>
        <memory mode='strict' nodeset='0-3'/>
        <memnode cellid='0' mode='strict' nodeset='0'/>
        <memnode cellid='1' mode='strict' nodeset='1'/>
        <memnode cellid='2' mode='strict' nodeset='2'/>
        <memnode cellid='3' mode='strict' nodeset='3'/>
    </numatune>
    <resource>
        <partition>/machine</partition>
    </resource>
    <os>
        <type arch='x86_64' machine='pc-i440fx-rhel7.6.0'>hvm</type>
        <bootmenu enable='yes'/>
    </os>
    <features>
        <acpi/>
        <apic/>
        <pae/>
    </features>
    <cpu mode='host-passthrough' check='none'>
        <topology sockets='4' cores='27' threads='2'/>
        <cache level='3' mode='emulate'/>
        <feature policy='require' name='rdtscp'/>
        <feature policy='require' name='invtsc'/>
        <feature policy='require' name='x2apic'/>
        <numa>
        <cell id='0' cpus='0-53' memory='744488960' unit='KiB'/>
        <cell id='1' cpus='54-107' memory='744488960' unit='KiB'/>
        <cell id='2' cpus='108-161' memory='744488960' unit='KiB'/>
        <cell id='3' cpus='162-215' memory='744488960' unit='KiB'/>
        </numa>
    </cpu>
```

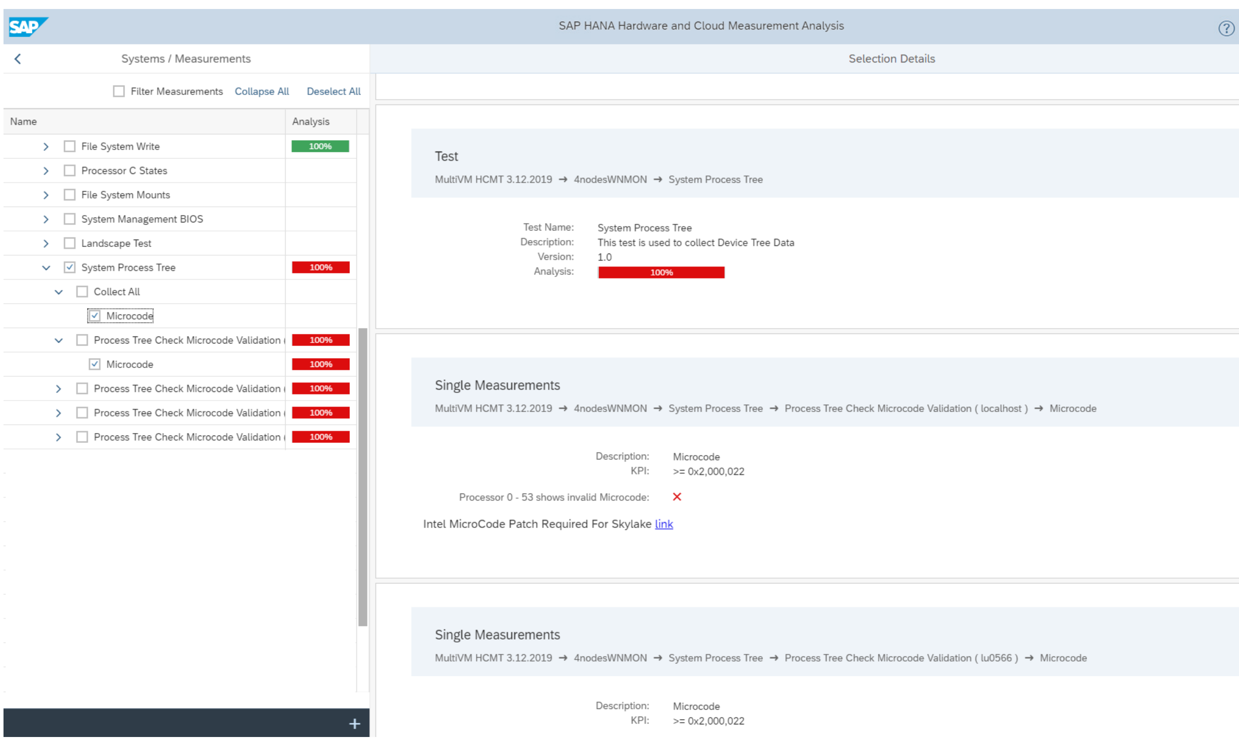
```
<clock offset='utc'>
    <timer name='rtc' tickpolicy='catchup'/>
    <timer name='pit' tickpolicy='delay'/>
    <timer name='hpet' present='no'/>
</clock>
<on_poweroff>destroy</on_poweroff>
<on_reboot>restart</on_reboot>
<on_crash>restart</on_crash>
<pm>
    <suspend-to-mem enabled='no'/>
    <suspend-to-disk enabled='no'/>
</pm>
<devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type='file' device='disk'>
    <driver name='qemu' type='raw' cache='none' io='native'/>
    <source file='/home/hanavm/hana_os.raw'/>
    <backingStore/>
    <target dev='vda' bus='virtio'/>
    <boot order='1'/>
    <alias name='virtio-disk0'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
    </disk>
    <controller type='pci' index='0' model='pci-root'>
    <alias name='pci.0'/>
    </controller>
    <controller type='sata' index='0'>
    <alias name='sata0'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x1f' function='0x2'/>
    </controller>
    <controller type='usb' index='0' model='piix3-uhci'>
    <alias name='usb'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x2'/>
    </controller>
    <interface type='bridge'>
    <mac address='52:63:00:53:6f:3f'/>
    <source bridge='br0'/>
    <target dev='vnet0'/>
    <model type='virtio'/>
    <alias name='net0'/>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
    </interface>
    <input type='tablet' bus='usb'>
    <alias name='input0'/>
    <address type='usb' bus='0' port='1'/>
    </input>
```

```
<input type='mouse' bus='ps2'>
<alias name='input1'/>
</input>
<input type='keyboard' bus='ps2'>
<alias name='input2'/>
</input>
<graphics type='vnc' port='5900' autoport='yes' listen='0.0.0.0'>
<listen type='address' address='0.0.0.0'/>
</graphics>
<video>
<model type='qxl' ram='65536' vram='65536' vgamem='16384' heads='1'
primary='yes'/>
<alias name='video0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>
</video>
<hostdev mode='subsystem' type='pci' managed='yes'>
<driver name='vfio'/>
<source>
<address domain='0x0000' bus='0x6d' slot='0x00' function='0x0'/>
</source>
<alias name='hostdev0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
</hostdev>
<hostdev mode='subsystem' type='pci' managed='yes'>
<driver name='vfio'/>
<source>
<address domain='0x0000' bus='0x6d' slot='0x00' function='0x1'/>
</source>
<alias name='hostdev1'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x07' function='0x0'/>
</hostdev>
<hostdev mode='subsystem' type='pci' managed='yes'>
<driver name='vfio'/>
<source>
<address domain='0x0000' bus='0x6e' slot='0x00' function='0x0'/>
</source>
<alias name='hostdev2'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>
</hostdev>
<hostdev mode='subsystem' type='pci' managed='yes'>
<driver name='vfio'/>
<source>
<address domain='0x0000' bus='0x6e' slot='0x00' function='0x1'/>
</source>
<alias name='hostdev3'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x09' function='0x0'/>
```

```
        </hostdev>
        <hostdev mode='subsystem' type='pci' managed='yes'>
        <driver name='vfio'/>
        <source>
        <address domain='0x0000' bus='0x17' slot='0x00' function='0x1'/>
        </source>
        <alias name='hostdev4'/>
        <address type='pci' domain='0x0000' bus='0x00' slot='0x0a' function='0x0'/>
        </hostdev>
        <memballoon model='none'>
        <alias name='balloon0'/>
        </memballoon>
    </devices>
    <seclabel type='none' model='none'/>
    <seclabel type='dynamic' model='dac' relabel='yes'>
        <label>+107:+107</label>
        <imagelabel>+107:+107</imagelabel>
    </seclabel>
  </domain>
```

Failing HCMT Subtest: System Process Tree Microcode Validation

Hardware and Cloud Measurement Tool (HCMT) is an SAP tool used by partners to validate that a system is compliant with SAP KPIs. At the time of publishing, the test **System Process Tree** → **Process Tree Check Microcode Validation** is failing when HCMT is run in the guest VM. This issue was identified during certification and a fix is planned for a future release of RHV. It is not a serious issue as the **System Process Tree** → **Process Tree Check Microcode Validation** test is for a specific microcode version, which is not reported in the guest VM even though the hypervisor CPU is running the correct version. To verify the microcode version, run HCMT on the bare metal system. Note that with qemu-kvm version later than `qemu-kvm-rhev-2.12.0-44.el7_8.2` this issue is fixed.



Virtualization limits for RHV

Refer to the guidelines in this document and in Supported limits for Red Hat Virtualization resource planning, sizing, and dedicated memory for the RHV host and guest VMs.

Red Hat

Revision History

Revision	Date	Description	Author
2.0	26.09.2019	First release	Nils Koenig
2.1	25.02.2020	Updated for RHV 4.3 and Multi VM	Nils Koenig
2.1.1	11.03.2020	Fixed references	Nils Koenig
2.1.2	13.4.2020	Updated for style and consistency	CCS
2.1.3	06.8.2020	Added NFS Storage and Live Migration; Minor reworks	SAP Alliance Team
2.1.4	8.2020	Fixed C-States Description	SAP Alliance Team
2.1.5	09.12.2020	Cascade Lake 6TB Support	SAP Alliance Team
3.0	16.07.2021	Added note for HBA passthrough configuration	SAP Alliance Team