



HPE SERVICEGUARD FOR LINUX WITH RED HAT, SUSE LINUX ENTERPRISE SERVER KVM AND RHEV GUESTS

CONTENTS

About this paper.....	3
Introduction.....	3
Terms and symbols.....	4
Supported KVM host and guest OS.....	5
Supported cluster deployment models with KVM guests.....	5
Cluster in a box.....	5
Cluster across box (host shared).....	5
Cluster across box (host exclusive).....	6
Hybrid cluster (host shared).....	6
Hybrid cluster (host exclusive).....	7
Cluster with VM as a package.....	8
Summary of supported cluster deployment models with various storage configurations.....	8
Supported KVM guest network interface device models.....	9
Configuring a KVM guest.....	9
Configuring HPE Serviceguard cluster on KVM guests.....	10
Configuration requirements for a HPE Serviceguard cluster with KVM guests.....	10
Configured resilient network.....	10
Configure shared storage.....	14
Restrictions and exclusion.....	19
Summary.....	20
References.....	20

LIST OF FIGURES

FIGURE 1. Cluster with multiple KVM guests all from same host.....	5
FIGURE 2. Cluster with KVM guest nodes across multiple hosts.....	5
FIGURE 3. Two clusters with multiple guests where none of them shares a host.....	6
FIGURE 4. Cluster with KVM guests and physical machine.....	7
FIGURE 5. Two clusters each with one KVM guest and physical machine.....	7
FIGURE 6. Cluster with two hosts each as a HPE Serviceguard node each having one KVM guest as a HPE Serviceguard package.....	8
FIGURE 7. Typical network configurations for RHEL (KVM) host and guest.....	10
FIGURE 8. virt-manager window of all network interfaces on guest.....	13
FIGURE 9. virt-manager window of a network interface on guest.....	13
FIGURE 10. virt-manager window to add new network interface on guest.....	14
FIGURE 11. High-level architecture diagram for using iSCSI shared storage in HPE Serviceguard cluster environments.....	14
FIGURE 12. High-level architecture diagram for using NPIV enabled FC shared storage as pass-through in HPE Serviceguard cluster environments.....	15
FIGURE 13. High-level architecture diagram for using FC shared storage as pass-through in Serviceguard cluster environments.....	15

LIST OF TABLES

TABLE 1. KVM terminology used in this document.....	4
TABLE 2. Symbols used in this document.....	4
TABLE 3. Snapshot of supported HPE Serviceguard cluster deployment models with various storage configurations with RHEL, SLES and RHEV-H guests.....	9
TABLE 4. Network configuration details.....	11
TABLE 5. Snapshot of supported HPE Serviceguard deployment models with different type of Adapter (HBA) using which FC device can be exposed to KVM guests.....	16
TABLE 6. Attribute name and required values can be used in XML.....	16
TABLE 7. Attribute name and required values can be used in XML.....	16



ABOUT THIS PAPER

Virtual machine technology is a powerful capability that can reduce costs while improving utilization of resources. HPE is also applying virtualization to other aspects of the data center and uniting virtual and physical resources to create an environment suitable for deploying mission-critical applications.

HPE Serviceguard for Linux® is certified for deployment on Red Hat® and SUSE Linux Enterprise Server® (SLES) kernel-based virtual machine (KVM) guests. This white paper discusses the different ways in which a KVM guest can be integrated in a Serviceguard for Linux cluster. It describes how a HPE Serviceguard for Linux cluster can be configured using KVM guests from a single host and multiple hosts, as well as a combination of KVM guests and physical machines, to provide high availability for applications.

This white paper provides details on recommended network and storage configurations for VMs used as HPE Serviceguard cluster nodes. In addition, this paper recommends how to eliminate single point of failures and provides pointers to other useful, relevant documents as appropriate.

For the complete list of supported operating systems, certified network and storage configurations, and Red Hat hypervisor versions with the listed version of HPE Serviceguard for Linux release, please refer to the “HPE Serviceguard for Linux Certification Matrix” document at hpe.com/info/linux-serviceguard-docs.

NOTE

Except as noted in this technical white paper, all HPE Serviceguard configuration options documented in the “Managing HPE Serviceguard for Linux Manual” available at hpe.com/info/linux-serviceguard-docs are supported for Red Hat and SLES KVM guests, and all the documented requirements apply.

INTRODUCTION

KVM-based VMs and RHEV-H Virtual machines are increasingly being deployed for server consolidation and flexibility. Virtual machine technology allows one physical server to simulate multiple servers, each concurrently running its own operating system (OS). In virtual machine technology, the virtualization layer also known as hypervisor¹ abstracts the physical resources so that each instance of an OS appears to have its own processor, memory, NIC, etc., when in fact they are virtual instances. This abstraction allows you to replace numerous existing physical servers with just one or few, but at the cost of greater exposure to single points of failure.

Linux KVM provided by Red Hat Enterprise Linux (RHEL) and SLES as a full virtualization solution or by RHEV-H as thin hypervisors. KVM differs from other popular alternatives like Xen and VMware® in terms of operation, performance, and flexibility. KVM comes as a kernel module, with a set of user-space utilities to create and manage the virtual machines.

HPE Serviceguard for Linux software is designed to protect applications and services from planned and unplanned downtime. By packaging an application or service with its associated resources, and moving that package to other servers as needed, HPE Serviceguard for Linux ensures 24x7 application availability. Packages can be moved automatically when HPE Serviceguard detects a failure in a resource, or manually to perform system maintenance or upgrades. By monitoring the health of each server (node) within a cluster, HPE Serviceguard for Linux can quickly respond to failures such as those that affect processes, memory, LAN media and adapters, disk, operating environments, and more.

HPE Serviceguard for Linux provides a significant level of protection. Specifically, it fails over an application when any of a large number of failures occurs, including:

- Application failure
- Failure of any of the components in the underlying network infrastructure that can cause failure of the application network
- Failure of storage
- An “OS stops responding” or failure of the virtual machine itself
- Failure of the physical machine

In addition, HPE Serviceguard for Linux provides a framework for integrating custom user-defined monitors, using the generic resource monitoring service.

¹ Hypervisor often refers to a layer that resides directly on server hardware, but terms are not used consistently across the industry.





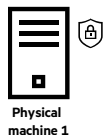



HPE Serviceguard for Linux, combined with KVM (RHEL, SLES, and RHEV-H) software solutions, can protect your applications, while also optimizing the cost, with no compromises on application availability and reliability.

TERMS AND SYMBOLS

TABLE 1. KVM terminology used in this document

Term	Definition
KVM	Kernel-based virtual machine
RHEL	Red Hat Enterprise Linux
SLES	SUSE Linux Enterprise Server
RHEV-H	Red Hat Enterprise Virtualization-hypervisor
RHEV-M	Red Hat Enterprise Virtualization-manager
KVM Host, host, hypervisor	Physical server on which KVM Hypervisor (RHEL, SLES or RHEV-H) is installed
KVM guest, guest, VM	KVM virtual machine carved out of the hypervisor
Physical machine	Physical server configured as a HPE Serviceguard cluster node
Bridge	A device bound to a physical network interface on the host which enables any number of guests to connect to the local network on the host. It is mapped to a physical NIC which acts as a switch to KVM guests
Cluster, HPE Serviceguard cluster	HPE Serviceguard for Linux cluster
FC	Fibre Channel storage
iSCSI	Internet Small Computer System Interface storage (IP based SCSI storage)
NPIV	N-Port ID Virtualization
vHBA	Virtual host bus adapter
NPIV enabled FC	Fibre Channel storage exposed using NPIV enabled vHBA

TABLE 2. Symbols used in this document

Symbol	Definition
	Virtual machine guest which is a HPE Serviceguard cluster node
KVM host	KVM host SLES or RHEL
	KVM host which is a HPE Serviceguard cluster node
	Physical machine which is a HPE Serviceguard cluster node
	HPE Serviceguard
	HPE Serviceguard package
	Shared storage



SUPPORTED KVM HOST AND GUEST OS

For the complete list of hypervisor and guest operating systems supported with HPE Serviceguard for Linux, refer to the “[HPE Serviceguard for Linux Certification Matrix](#).”

SUPPORTED CLUSTER DEPLOYMENT MODELS WITH KVM GUESTS

The supported HPE Serviceguard for Linux cluster deployment models when using KVM guests as cluster nodes are as follows:

Cluster in a box

In the “Cluster in a box” model, a cluster is formed with KVM guests, all of which are carved out of a single host. Though this configuration provides consolidation of resources, it is not an ideal configuration, as failure of the host will bring down all the nodes of the cluster. Hence, this configuration is not recommended.

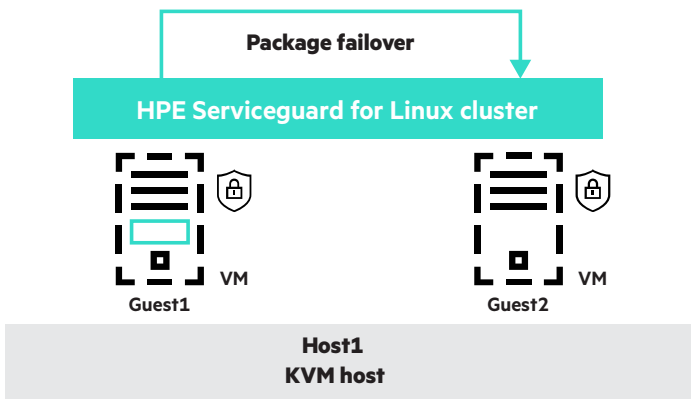


FIGURE 1. Cluster with multiple KVM guests all from same host

Cluster across box (host shared)

In the “Cluster across box (host shared)” model, a cluster is formed with multiple guests that are hosted on two or more hosts. The term “host shared” denotes that multiple VMs sharing a host can be configured in the same cluster. When designing such a cluster, one must ensure that the cluster nodes are distributed across the hosts in such a manner, that the failure of any one of the hosts will not result in more than half the cluster nodes going down, thereby creating a single point of failure (SPOF).

HPE Serviceguard is installed on the guests and a cluster is formed with these VMs. HPE Serviceguard provides high availability to the applications deployed as packages in the VMs, and fails over the applications to adoptive nodes in case of failures.

Let us consider the example (Figure 2) of a four-node cluster. In this case, the correct distribution would be to have two VMs each from Host1 and Host2 configured as HPE Serviceguard cluster nodes, rather than having three VMs from Host1 and one from Host2. In the second case, the failure of Host1 with three VMs will bring down the entire cluster.

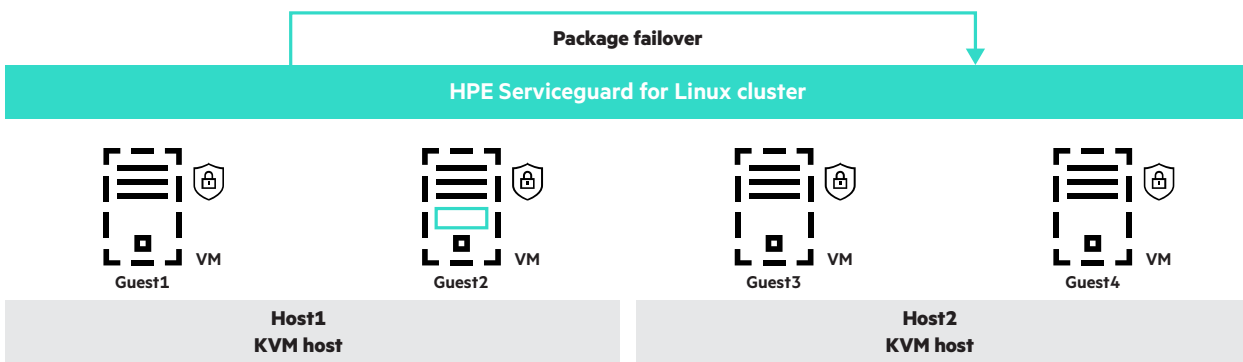


FIGURE 2. Cluster with KVM guest nodes across multiple hosts



Cluster across box (host exclusive)

In the “Cluster across box (host exclusive)” model, a cluster is formed with multiple guests, each hosted on a different host. The term “host exclusive” denotes that none of the cluster nodes shares a host as shown in Figure 3. In other words, one host can have multiple guests, all of which can be part of different clusters, but no two guests from the same host can belong to the same cluster.

This model does not mandate the use of NPIV-enabled storage infrastructure when using FC devices as shared storage.

HPE Serviceguard is installed on the guests and a cluster is formed with these VMs. HPE Serviceguard provides high availability to the applications deployed as packages in the VMs and fails over the applications to adoptive nodes in case of failures.

The example in Figure 3 shows two hosts—Host1 and Host2. Host1 has Guest1 and Guest3, and Host2 has Guest2 and Guest4 hosted on them. Here, two clusters are configured where Cluster1 is configured with Guest1 and Guest2 and Cluster2 is configured with Guest3 and Guest4. In both the clusters, none of the guests shares a host.

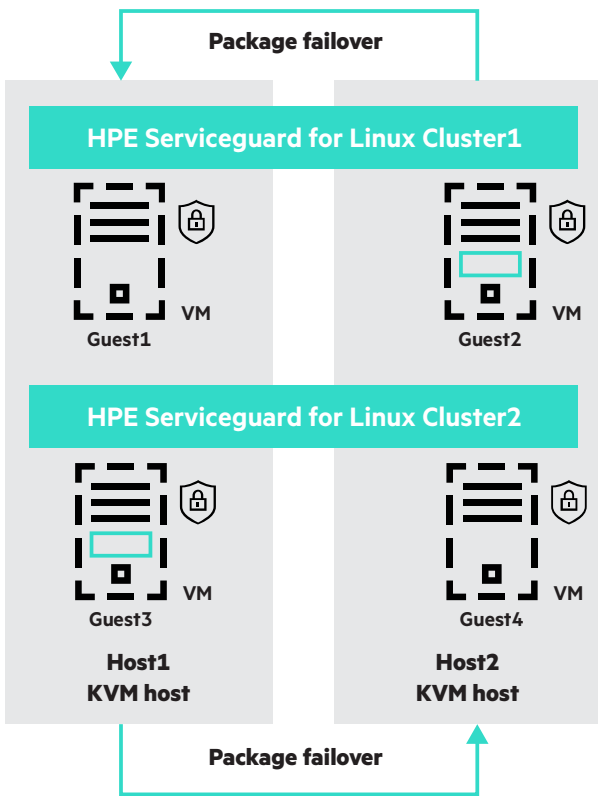


FIGURE 3. Two clusters with multiple guests where none of them shares a host

Hybrid cluster (host shared)

In the “Hybrid cluster (host shared)” model, a combination of one or more physical machines and VMs are used as nodes in a HPE Serviceguard cluster. The VMs may be hosted on multiple hosts. The term “host shared” denotes that multiple VMs sharing a host can be configured in the same cluster. When designing such a cluster one must ensure that the VMs are distributed across the hosts in such a manner that the failure of any one of the hosts will not result in more than half the cluster nodes going down, thereby creating a single point of failure (SPOF).

HPE Serviceguard is installed on the guests and physical machines, and a cluster is formed among them. HPE Serviceguard provides high availability to the applications deployed as packages in the VMs and physical machines. In case of failures, HPE Serviceguard fails over the application to other adoptive cluster nodes. The application can be failed over from a VM to a physical machine and vice versa.

This is a very powerful model where the application can primarily run on the physical machine and, in case of failures, can fail over to an adoptive VM. This enables users to take advantage of the performance of a physical machine, and at the same time allows for consolidation of standby resources.



The example in Figure 4 of a four-node cluster, a cluster is configured using two physical machines and two VMs. The package can be failed over between any of the cluster nodes. This example shows a correct distribution of VMs where failure of Host1 will not result in more than half of the nodes going down.

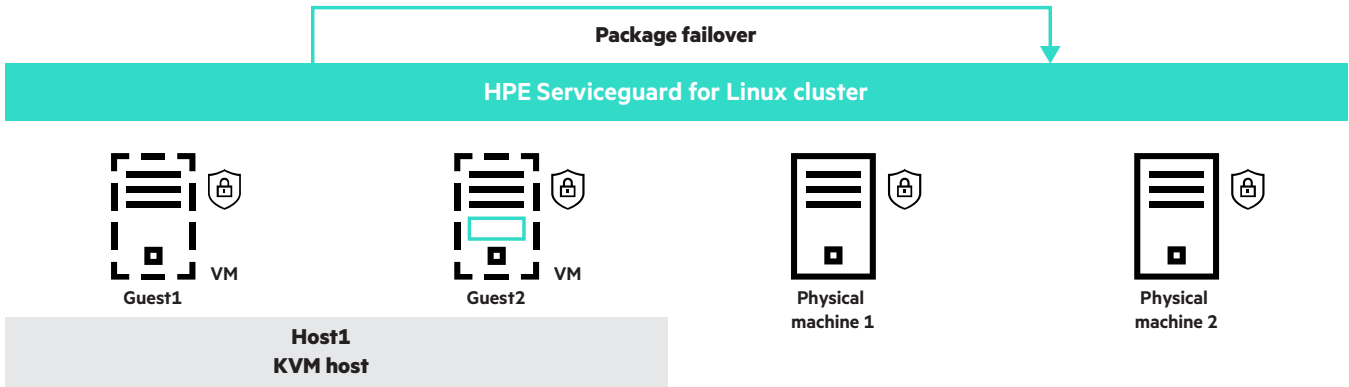


FIGURE 4. Cluster with KVM guests and physical machine

Hybrid cluster (host exclusive)

In the “Hybrid cluster (host exclusive)” model a combination of one or more physical machines and one or more VMs (all hosted on different hosts) are used as nodes in a HPE Serviceguard cluster. The term “host exclusive” denotes that none of the cluster nodes shares a host as shown in Figure 5.

This model does not mandate the use of NPV-enabled storage infrastructure when using FC devices as shared storage. HPE Serviceguard is installed on the guests and physical machines, and a cluster is formed among them. HPE Serviceguard provides high availability to the applications deployed as packages in the VMs and physical machines. In case of failures, HPE Serviceguard fails over the application to other adoptive cluster nodes. The application can be failed over from a VM to a physical machine and vice versa.

This is a very powerful model where the application can primarily run on the physical machine and, in case of failures, can fail over to an adoptive VM. This enables users to take advantage of the performance of a physical machine, and at the same time allows for consolidation of standby resources.

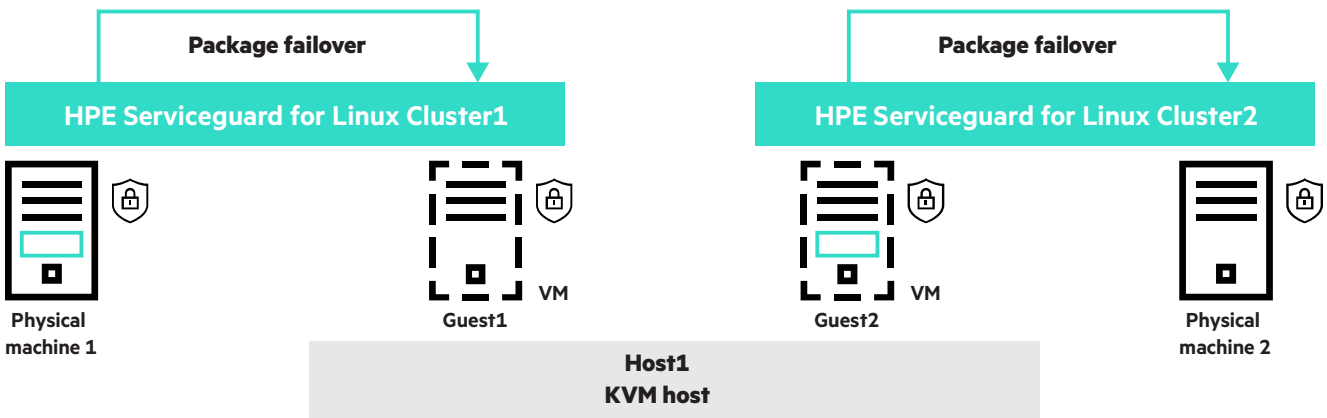


FIGURE 5. Two clusters each with one KVM guest and physical machine



Cluster with VM as a package

In the “Cluster with VM as a package” model, HPE Serviceguard is deployed on the KVM hosts and the KVM guests are deployed as HPE Serviceguard packages, and HPE Serviceguard provides high availability for the VMs. HPE Serviceguard starts, stops, and monitor the VM that are deployed as packages. In the event of a failure (VM, Host failure, etc.) HPE Serviceguard and will restart the VM on the adoptive Host cluster node. Compared to the VM as a node deployment in this model, HPE Serviceguard monitors the VM and the Host for failures.

This model provides multiple benefits:

1. This model reduces the resources required for standby as there is no need for a dedicated standby VM that needs to be up and running all the time.
2. This model translates to lesser resources like IP addresses, etc. This also makes management easier as one needs to install and maintain only one OS image.
3. This model provides root disk monitoring.

In this deployment model, VM must have its VM image/root disk on a logical volume of a volume group carved out of a shared disk which is available on all the adoptive nodes (Host). Ensure that each VM image is placed on a single volume group, and must be exclusively used only for this purposes. This VM image/root disk must be configured in the package.

This model does not mandate the use of NPIV-enabled storage infrastructure when using FC devices as shared storage.

For more details about this deployment model, please refer to the document “HPE Serviceguard Toolkit for KVM on Linux User Guide” at hpe.com/info/linux-serviceguard-docs.

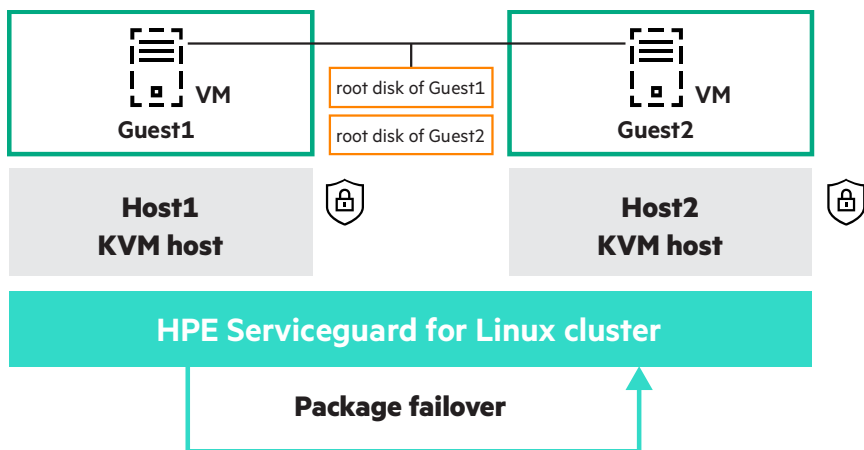


FIGURE 6. Cluster with two hosts each as a HPE Serviceguard node each having one KVM guest as a HPE Serviceguard package

Summary of supported cluster deployment models with various storage configurations

An HPE Serviceguard for Linux cluster that includes virtual machines as cluster nodes have multiple deployment model. Table 3 provides a summary of the supported models with different storage configurations in a HPE Serviceguard cluster. Please refer to the above sections in this document to find out more about each supported model.



TABLE 3. Snapshot of supported HPE Serviceguard cluster deployment models with various storage configurations with RHEL, SLES, and RHEV-H guests

Supported cluster models	RHEL			SLES			RHEV-H		
	FC	NPIV enabled FC	iSCSI	FC	NPIV enabled FC	iSCSI	FC	NPIV enabled FC	iSCSI
Cluster in a box	✗	✓	✓	✗	✗	✓	✗	✗	✓
Cluster across box (host shared)	✗	✓	✓	✗	✗	✓	✗	✗	✓
Cluster across box (host exclusive)	✓	✓	✓	✗	✗	✓	✗	✗	✓
Hybrid cluster (host shared)	✗	✓	✓	✗	✗	✓	✗	✗	✓
Hybrid cluster (host exclusive)	✓	✓	✓	✗	✗	✓	✗	✗	✓
Cluster with VM as a package	✓	✓	✓	✗	✗	✓	✗	✗	✗
Extended Distance Cluster (XDC)	✓	✓	✗	✗	✗	✗	✗	✗	✗
Metrocluster	✓	✓	✓	✗	✗	✓	✗	✗	✓
Continentalclusters	✓	✓	✓	✗	✗	✓	✗	✗	✓

SUPPORTED KVM GUEST NETWORK INTERFACE DEVICE MODELS

When configuring the guest network interfaces the “device model” used must be one of the HPE Serviceguard supported device models as listed in “[HPE Serviceguard for Linux Certification Matrix](#).” Ensure that all interfaces across the KVM guest cluster nodes have the same “device model.”

Configuring a KVM guest

Linux KVM available with RHEL, SLES as a full virtualization solution and with RHEV-H as a thin hypervisors.²

SLES and RHEL provides a tool, “virt-manager,” which is a very simple, easy-to-use, and intuitive GUI interface to create, manage, and administer virtual machines. A command line alternative, “virsh,” also gives a shell that can be used to create, manage, and administer virtual machines using a rich set of commands.

Similarly, RHEV-H provides a tool—Red Hat Enterprise Virtualization Management (RHEV-M) to create, manage, and administer virtual machine created on RHEV-H.

Here is a high-level overview of the steps required to set up a HPE Serviceguard for Linux cluster using KVM guests.

Before creating KVM guests, ensure that the CPU, memory resources required by the KVM guests are available and if the virtualization support is enabled at the BIOS on all KVM hosts to be used in the cluster.

1. Make sure that the required KVM packages are installed on the hosts.
2. Make sure that the service “libvirtd” is running on the hosts where KVM guests are to be created.
3. Create the KVM guests and install the required operating system in the KVM guests. HPE Serviceguard for Linux does not have any restriction on where the boot image will be stored. You can use “virt-manager” for creating and configuring the KVM guests. For the list of supported guest OSs, refer to [Supported KVM host and guest OS](#).
4. Complete the network configuration as mentioned in the section [Configured resilient network](#) for HPE Serviceguard cluster using KVM guests on all guests.
5. Complete the storage configuration as mentioned in the section [Configure shared storage for HPE Serviceguard cluster on KVM guests on all guests](#).

² Hypervisor often refers to a layer that resides directly on server hardware, but terms are not used consistently across the industry.



CONFIGURING HPE SERVICEGUARD CLUSTER ON KVM GUESTS

Install HPE Serviceguard for Linux and its prerequisites on all the KVM guests that need to be in the cluster. For information about installing and updating HPE Serviceguard, see the latest HPE Serviceguard for Linux Release Notes at hpe.com/info/linux-serviceguard-docs.

Configure and creating HPE Serviceguard for Linux cluster and other required resources like packages, service, etc. For more information on how to configure, refer to Managing HPE Serviceguard for Linux at hpe.com/info/linux-serviceguard-docs.

CONFIGURATION REQUIREMENTS FOR A HPE SERVICEGUARD CLUSTER WITH KVM GUESTS

Configured resilient network

HPE Serviceguard for Linux recommends having a highly resilient network configuration with redundant heartbeats and redundant data networks to avoid single point of failures. The following section describes how to achieve network redundancy using bridged network configuration (also known as physical device sharing).

There are multiple ways of configuring a resilient network using networking technologies like bridging and bonding or NIC teaming, at different layers. Figure 7 shows a simple example of a resilient network configuration using bridging at host and bonding or teaming at KVM guests.

NOTE:

In below figure the bonding configuration at KVM guest level can be replaced with the NIC teaming configuration. The configuration can either have bonding or teaming, or both, depending on the requirement. The teaming uses a kernel driver to implement fast handling of packet flows, as well as user-space libraries and services for other tasks. The network teaming is an easily extensible and scalable solution for load-balancing and redundancy requirements. For configuration details, refer to the network administration guides of the respective supported OS distributions. For more information about the teaming and the supported modes, refer to the section “Interface Teaming” in the latest edition of “Managing HPE Serviceguard for Linux” manual available at: hpe.com/info/linux-serviceguard-docs.

Figure 7 shows a typical network configuration where four KVM guest across two physical hosts are bridged to form an HPE Serviceguard for Linux cluster with redundant heartbeat networks.

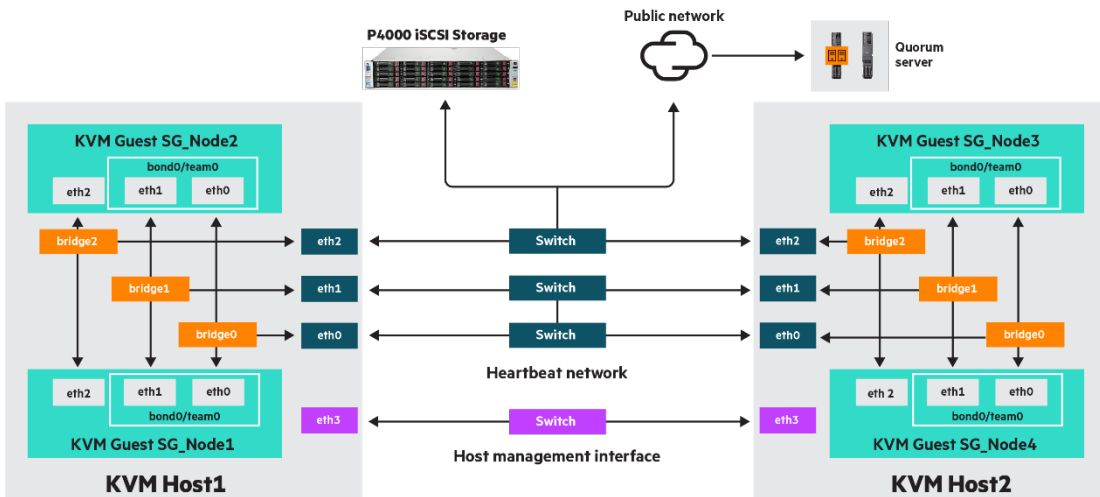


FIGURE 7. Typical network configurations for RHEL (KVM) host and guest



TABLE 4. Network configuration details

Node name	Interface name	Remarks
KVM Guest SG_NODE1	bond0/team0 (eth0 and eth1)	Assign heartbeat IP
KVM Guest SG_NODE2	bond0/team0 (eth0 and eth1)	Assign heartbeat IP
KVM Guest SG_NODE3	bond0/team0 (eth0 and eth1)	Assign heartbeat IP
KVM Guest SG_NODE4	bond0/team0 (eth0 and eth1)	Assign heartbeat IP
KVM Guest SG_NODE1	eth2	Assign public IP
KVM Guest SG_NODE2	eth2	Assign public IP
KVM Guest SG_NODE3	eth2	Assign public IP
KVM Guest SG_NODE4	eth2	Assign public IP
KVM Host1 and KVM Host2	bridge0 (uses eth0 as slave)	Used as "Source device" for eth0 in all KVM guests
KVM Host1 and KVM Host2	bridge1 (uses eth1 as slave)	Used as "Source device" for eth1 in all KVM guests
KVM Host1 and KVM Host	bridge2 (uses eth2 as slave)	Used as "Source device" for eth2 in all KVM guests
KVM Host1 and KVM Host	eth0	Connected to heartbeat network switch
VM Host1 and KVM Host2	eth1	Connected to heartbeat network switch
KVM Host1 and KVM Host2	eth2	Connected to public network
KVM Host1 and KVM Host2	eth3	Can be used for host management (not mandatory)

The bridge network configuration must be performed in two parts:

- Network configuration on host
- Network configuration on guest

NOTE

The steps listed after this section are based on the respective OS documentation; for any updates or alternatives of these steps refer to the latest versions of the below Red Hat Enterprise Linux 7/8 guides

- Virtualization Deployment and Administration Guide
- Virtualization Host Configuration and Guest Installation Guide available at access.redhat.com/documentation/en-us/red_hat_enterprise_linux/

For SUSE Linux Enterprise Server 12/15 refer to latest version of the "Storage Administration Guide" which is available at: documentation.suse.com/

Similarly, to configure a network for RHEV-H and their guests, use RHEV-M. For detailed steps to configure network, Refer to "Red Hat Virtualization Administration Guide" available at access.redhat.com/documentation/en-us/red_hat_virtualization/

Network configuration on host

This bridge can be created using the following steps:

1. Create a new network interface script file in "/etc/sysconfig/network-scripts/" directory with the name "ifcfg-<device_name>" where the <device_name> must match the value of the DEVICE parameter inside the file. Also disable NetworkManager for the interface to be bridged by add "NM_CONTROLLED=no" to the ifcfg-* network script being used for the bridge. Its contents are as follows:

```
DEVICE=bridge0
TYPE=Bridge
ONBOOT=yes
DELAY=0
NM_CONTROLLED=no
```



NOTE

The line TYPE=Bridge is case sensitive. It must have uppercase "B" and lowercase "ridge."

2. Add the physical interface to the bridge by modifying the network interface script of the given physical interface. Edit the file and add a line BRIDGE=bridge0, so that the contents of the configuration file look like the following example:

```
DEVICE=eth0
BRIDGE=bridge0
BOOTPROTO=none
HWADDR=00:19:b9:7e:c8:63
ONBOOT=yes
TYPE=Ethernet
IPV6INIT=no
NM_CONTROLLED=no
```

NOTE

Repeat steps 1 and 2 for every interface that requires to be bridged.

3. Configure IP tables to allow all traffic to be forwarded across the bridge

```
# iptables -I FORWARD -m physdev --physdev-is-bridged -j ACCEPT
# service iptables save
# service iptables restart
```

Alternatively, prevent bridged traffic from being processed by iptables rules by appending these lines in the file

/etc/sysctl.conf:

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```

Reload the kernel parameter configured with sysctl by executing this command:

```
# sysctl -p /etc/sysctl.conf
```

4. Restart the network services to bring all the network configuration changes into effect.

```
# service network restart
```

5. Verify that your eth0 was added to the bridge0 using the brctl show command.

```
# brctl show
```

The output must look similar to the following:

bridge name	bridge Id	STP enabled interfaces
virbr0	8000.00000000000000	yes
bridge0	8000.0019b97ec863	yes eth0



“bridge0” is now available through virt-manager and libvirt. Guests can now connect to this device for full network access. This can be also viewed through virt-manager as shown here:

Click on Edit → Connection Details → Network Interfaces. You can see the list of bridge interface created in Figure 8.

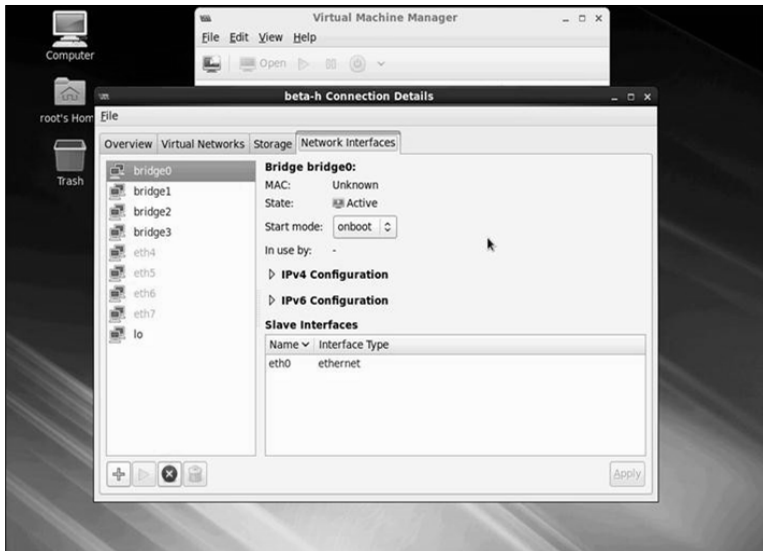


FIGURE 8. virt-manager window of all network interfaces on guest

Network configuration on guest

Network access is provided to the guest VMs using the underlying bridged network interfaces configured in the host as shown in the previous steps. To achieve this, add the required virtual interfaces to the guest and configure the appropriate “Source device” and “Device model” for each of them.

- Step 1:** Open virt-manager by executing the command “virt-manager.”
- Step 2:** From the list of VM select the VM to which the interfaces are to be added and click to open its window.
- Step 3:** Go to “Details” view, and click “Add Hardware.” As shown in Figure 9.

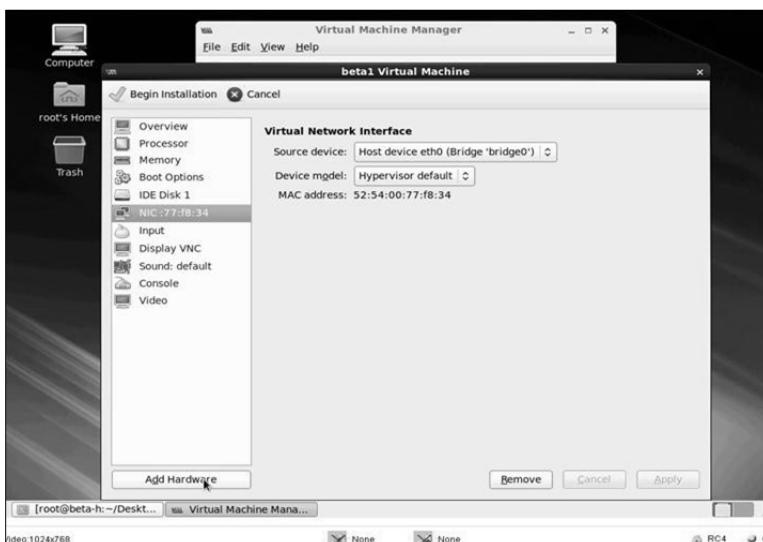


FIGURE 9. virt-manager window of a network interface on guest



Step 4: Click the “Network” tab, choose the “Host device” and “Device model” and click “Finish”. “Host device” should be the appropriate bridge interface, which has been created on the host as explained under [Network configuration on host](#) section.

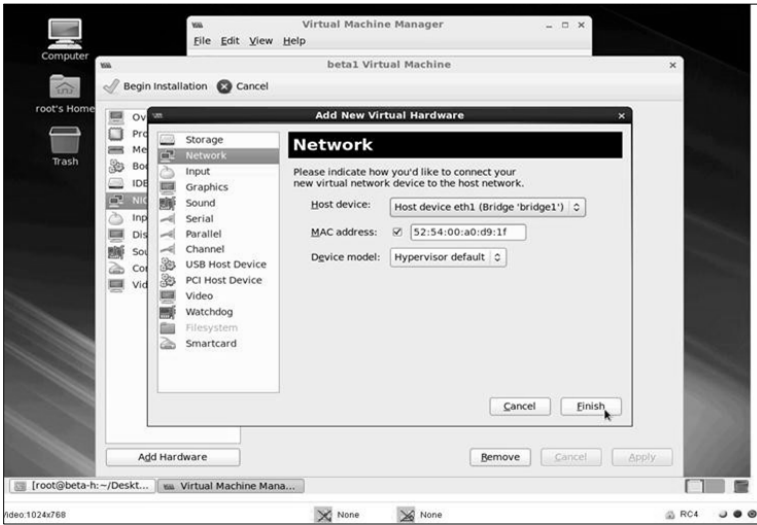


FIGURE 10. virt-manager window to add new network interface on guest

Step 5: Repeat the steps 1–4 for all the required network interfaces on the guest.

Configure shared storage

HPE Serviceguard for Linux is a shared storage HA cluster that requires all applications deployed as HPE Serviceguard packages to have their data on shared storage so as to enable access from all cluster nodes. In shared storage clustering environments, data integrity is of utmost importance and to ensure that, during all failover scenarios HPE Serviceguard uses SCSI-3 Persistent Reservation. For more information, see the section “About Persistent Reservations” in the latest edition of Managing HPE Serviceguard For Linux available at: hpe.com/info/linux-serviceguard-docs.

HPE Serviceguard for Linux supports various cluster layouts with FC and iSCSI shared storage, refer [Summary of supported cluster deployment models with various storage configurations](#). Further in this section, we will discuss how to configure FC and iSCSI shared storage in KVM guest.

Configure iSCSI devices as shared storage for KVM guest nodes

HPE Serviceguard for Linux supports iSCSI devices exposed using software initiator only. The iSCSI devices must be directly exposed to the software initiator configured in the guest as shown in the Figure 11.

HPE Serviceguard for Linux supports use of iSCSI shared storage with KVM guests carved out from SLES (KVM), RHEL (KVM) and RHEV-H.

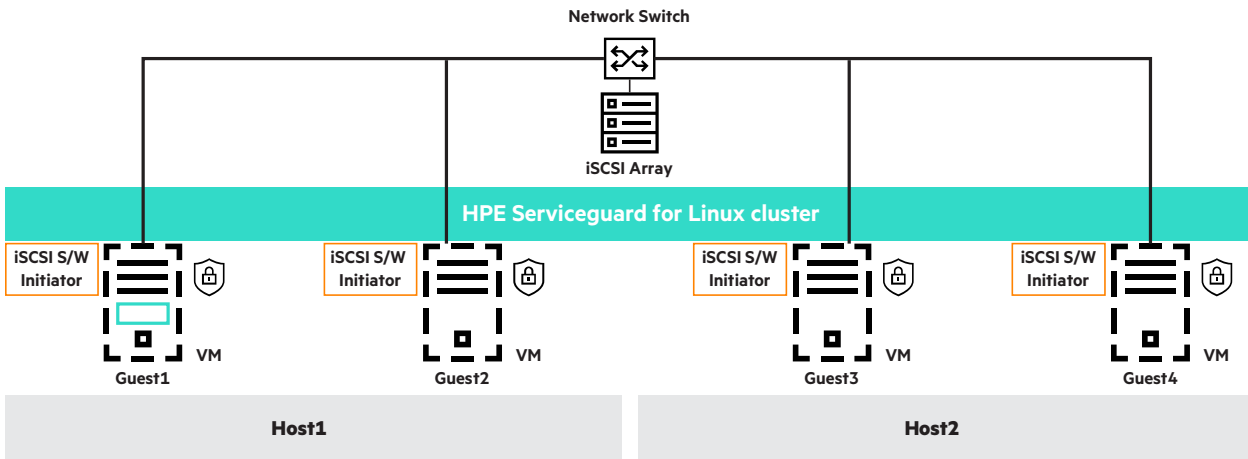


FIGURE 11. High-level architecture diagram for using iSCSI shared storage in HPE Serviceguard cluster environments



The steps to discover and log in to iSCSI targets from the guests can be found in the appropriate operating system manuals as listed below.

Refer to latest version of the below Red Hat Enterprise Linux 7/8 guides

- Virtualization Deployment and Administration Guide
- Virtualization Host Configuration and Guest Installation Guide available at access.redhat.com/documentation/en-us/red_hat_enterprise_linux/
- Refer to “Red Hat Virtualization Administration Guide” available at access.redhat.com/documentation/en-us/red_hat_virtualization/

For SUSE Linux Enterprise Server 12/15 refer to

“Storage Administration Guide” which is available at: documentation.suse.com/

It is also recommended to have multipath configured for iSCSI devices.

Configure Fibre Channel devices as shared storage for KVM guest nodes

When using FC devices as shared storage, it must be configured to allow SCSI-3 PR operations from the guest. To allow SCSI-3 PR operations, the FC devices must be configured as pass-through devices as shown in Figure 12. The OS image’s disk need not be configured as pass-through.

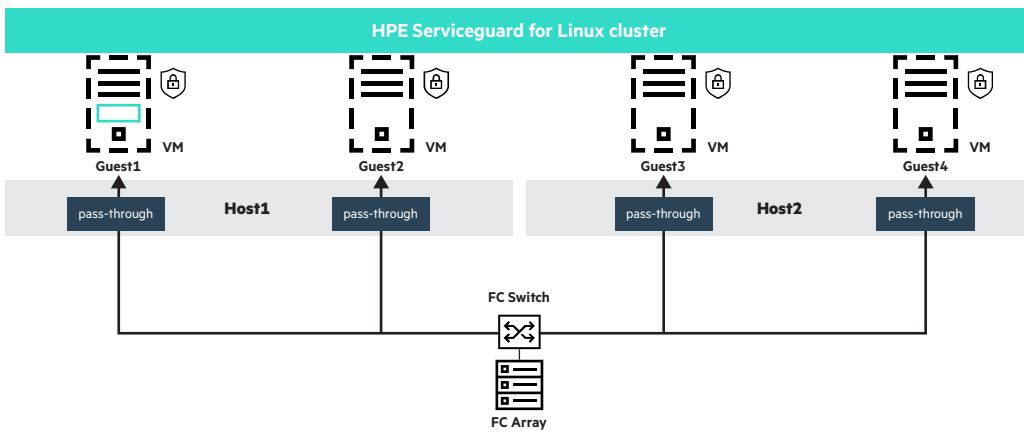


FIGURE 12. High-level architecture diagram for using NPIV enabled FC shared storage as pass-through in HPE Serviceguard cluster environments

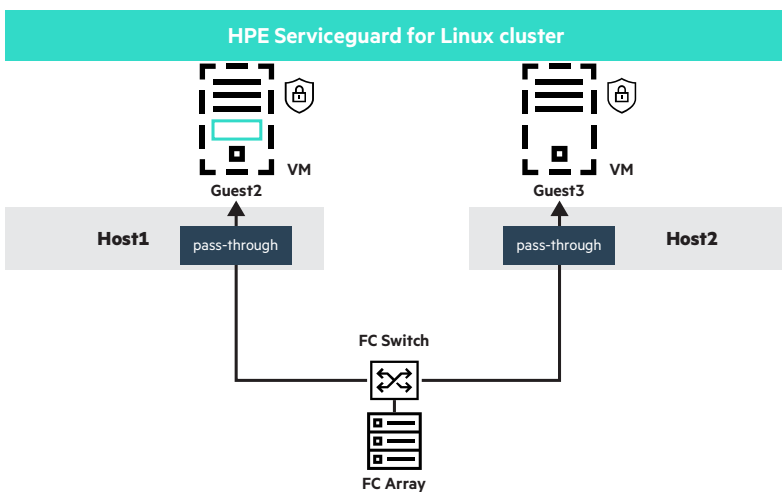


FIGURE 13. High-level architecture diagram for using FC shared storage as pass-through in HPE Serviceguard cluster environments

NOTE

VM created using RHEV-H do not support FC devices as shared storage.



TABLE 5. Snapshot of supported HPE Serviceguard deployment models with different type of Adapter (HBA) using which FC device can be exposed to KVM guests

Supported cluster models	Adapter (HBA) type		
	Physical	Single NPIV enabled vHBA on a physical HBA	Virtual Multiple NPIV enabled vHBA on same physical HBA
Cluster in a box	✗	✗	✓
Cluster across box (host shared)	✗	✗	✓
Cluster across box (host exclusive)	✓	✓	✗
Hybrid cluster (host shared)	✗	✗	✓
Hybrid cluster (host exclusive)	✓	✓	✗
Cluster with VM as a package	✓	✓	✓
Extended Distance Cluster	✓	✓	✓
Metrocluster	✓	✓	✓
Continentalclusters	✓	✓	✓

For exposing a block device using a physical HBA to KVM guest as a SCSI LUN

TABLE 6. Attribute name and required values can be used in XML

Attribute	Required value	Description
type	block	To define the type of source device
device	lun	To define type of target device. The name "Target dev" must be unique and a device of the same name should not be exist in KVM guests
sgio	unfiltered	To allow guest-issued SGIO ioctl commands to pass-through, for persistence reservation implementation
bus	scsi	

"virsh attach-device" command uses XML configuration file to create SCSI LUN on guest. HPE Serviceguard requires certain attributes to set as mentioned in Table 6 and Table 7 for a XML configuration:

1. The <shareable> element must be set in XML configuration to share a LUN between guests.
2. Required value for each attribute in XML configuration must be set.

For exposing a storage pool volume using a NPIV enabled HBA to KVM guest as a SCSI LUN

TABLE 7. Attribute name and required values can be used in XML

Attribute	Required value	Description
type	block	To define the type of source device
device	lun	To define type of target device. The name "Target dev" must be unique and a device of the same name should not be exist in KVM guests
sgio	unfiltered	To allow guest-issued SGIO ioctl commands to pass-through, for persistence reservation implementation
bus	scsi	

The steps to expose the RHEL host's shared FC disks to guests as pass-through and configuring persistent NPIV/vHBA, can be found in the "Virtualization Deployment and Administration Guide" available at access.redhat.com/documentation/en-us/red_hat_enterprise_linux/.



Below example to illustrate how to expose an FC device having two paths to the host (in this case light4), as SCSI pass-through device to KVM guest (in this case nickel3), which can be used for [Cluster across box \(host exclusive\)](#) deployment model. This is just a reference for any change in command and XML attribute, please refer latest version of “Virtualization Deployment and Administration Guide” available at access.redhat.com/documentation/en-us/red_hat_enterprise_linux/.

1. Check for a pre-existing SCSI controller on the KVM guest (nickel3), run the following command on KVM hypervisor (light4).

```
# virsh dumpxml nickel3 |grep controller.*scsi
```

If a device controller is present, the command will list one or more lines similar to the following:

```
<controller type='scsi' model='virtio-scsi' index='0' />
```

2. If previous step did not show a device controller on guest (nickel3), create the description for a new controller in a new file and add it to the virtual machine, using the following steps:
 - a. Create the device controller by writing a <controller> element in a new file and save this file with an XML extension. virtio-scsi-controller.xml on host (light3), for example.

```
<controller type='scsi' model='virtio-scsi' />
```

- b. Attach the device controller you just created in virtio-scsi-controller.xml with your guest virtual machine (nickel3), run the following command on the host (light3).

```
# virsh attach-device --persistent nickel3 controller_scsi.xml
```

NOTE

The “virsh attach-device” command supports multiple options out of which only the “--persistent” option must be used. This is to ensure that added controllers will continue to exist on KVM guests even after a guest or host reboot.

3. Expose all required paths for the FC device individually to the KVM guests. To expose an FC device path from host to KVM guest, find the corresponding stable path under “/dev/disk/by-path” directory. For example, if “/dev/sdb” is one of the FC device paths on the host, then find the stable path using the following command:

```
# ls -l /dev/disk/by-path/ | grep sdb
lrwxrwxrwx. 1 root root    5 Aug 10 21:28 pci-0000:08:00.0-fc-
0x5001438011362ad8-lun-1-> ../../sdb
```



4. Create an empty XML file <sdb>.xml and populate with information as shown below.

```
<disk type='block' device='lun' sgio='unfiltered'>
  <driver name='qemu' type='raw' />
  <source dev='/dev/disk/by-path/pci-0000\:08\:00.0-fc-
0x5001438011362ad8-lun-1' />
  <target dev='sda' bus='scsi' />
  <shareable />
</disk>
```

The “source dev” used here is the stable path found in step 3 and “target dev” is the device name as it appears in the guest after attaching the device. In some cases, if the name is already used by the guest, then a different name will be automatically selected. That may appear on KVM guest after the attach.

Make sure specify all required values for corresponding attribute in XML as explained in [Table 6](#).

5. Attach a device path to KVM guest, run “virsh attach-device” command on host and make it persistent using the “- -persistent” option:

```
# virsh attach-device --persistent nickel3 sdb.xml
Device attached successfully
```

NOTE

“virsh attach-device” command supports multiple options out of which only the “- -persistent” option must be used. This is to ensure that all exposed FC device paths will continue to exist on KVM guests even after a guest reboot.

6. After successful execution of above command on host as explained in step 5, the device path must be visible to the KVM guest; verify it by running “fdisk” command on KVM guests (below snapshot a new device name “sda” has appeared).

```
# fdisk -l |grep -i disk
Disk /dev/vda: 8589 MB, 8589934592 bytes
Disk identifier: 0x00058bac
Disk /dev/mapper/vg_nickel2-lv_root: 7205 MB, 7205814272 bytes Disk
identifier: 0x00000000
Disk /dev/mapper/vg_nickel2-lv_swap: 855 MB, 855638016 bytes Disk
identifier: 0x00000000
Disk /dev/sda: 2147 MB, 214703648 bytes
Disk identifier: 0x00000000
```

NOTE

Device path name appearing on the guest may not be the same as specified in the XML.

7. To expose another path of FC disk from host to KVM guest, repeat steps 3 to 5.

In this example, the FC disk used has two paths. After successful execution of steps 3 to 5, another device path must be made visible in the KVM guest. This can again be verified by running the “fdisk” command on the KVM guests (the following snapshot shows the new device name “sdb”).

```
# fdisk -l |grep -i disk
Disk /dev/vda: 8589 MB, 8589934592 bytes Disk
identifier: 0x00058bac
Disk /dev/mapper/vg_nickel2-lv_root: 7205 MB, 7205814272 bytes Disk
identifier: 0x00000000
Disk /dev/mapper/vg_nickel2-lv_swap: 855 MB, 855638016 bytes Disk
identifier: 0x00000000
Disk /dev/sda: 2147 MB, 2147483648 bytes Disk
identifier: 0x00000000
Disk /dev/sdb: 2147 MB, 2147483648 bytes Disk
identifier: 0x00000000
```

Repeat Steps 3 to 5, to expose all required paths to a KVM guest. It is recommended to have multipath configured for FC devices using Device Mapper Multipath Software only at guest level.

It is quite possible that user might configure multipath at both KVM host and guest level, then user must follow below recommended steps:

- Blacklist the LUN paths of the devices at host when the same LUNs are configured for multipathing at guest level. Devices can be blacklisted by editing host's file “/etc/multipath.conf” and followed by restarting multipath daemon at host to see immediate reflect of the changes.
- Use LVM filtering at host level to avoid managing disks or volume groups belonging to guests at host level.
- Stop and disable LVM metadata daemon (lvm2md) at both host and guest.

Please refer to Virtualization Deployment and Administration Guide or Logical Volume Manager Administration Guide of operating system for more details on how to blacklist or filter the LVM devices at host level.

8. Repeat all the steps from 1–8, to expose all required paths to all KVM guests from where the shared device will be accessed.

RESTRICTIONS AND EXCLUSION

1. Lock LUN as arbitration is not supported with iSCSI disk; use Quorum server for arbitration.
2. Live migration of KVM guests that are configured as HPE Serviceguard cluster nodes is not supported.
3. FC devices cannot be used as shared storage, when guests created using RHEV-H and SLES are used as HPE Serviceguard cluster nodes.
4. NPIV enabled FC cannot be used as shared storage when guests created using RHEV-H and SLES are used as cluster nodes.
5. HPE Serviceguard supports NPIV enabled FC devices exposed to guest via persistent vHBA only.
6. HPE Serviceguard supports exposing FC devices to multiple KVM guest via multiple NPIV enabled vHBAs created on a physical HBA, here for each guest one NPIV enabled vHBA must be created on a physical HBA.



SUMMARY

This white paper describes best practices for deploying HPE Serviceguard for Linux cluster in a KVM environment. It is not the intent of this document to duplicate the strategies and best practices of other HPE or Red Hat or SLES technical white paper. The strategies and best practices offered here are presented at a very high level to provide general knowledge. Where appropriate, you are referred to specific documentation that provides more detailed information.

REFERENCES

Refer to latest versions of the below Red Hat Enterprise Linux 7/8 guides at access.redhat.com/documentation/en-us/red_hat_enterprise_linux/.

- Virtualization Deployment and Administration Guide
- Virtualization Getting Started Guide
- Virtualization Host Configuration and Guest Installation Guide
- Virtualization Administration Guide
- SUSE Linux Enterprise Server 12/15 Storage Administration Guide available at documentation.suse.com/
- SUSE Linux Enterprise Server 12/15 Virtualization Guide available at documentation.suse.com/

LEARN MORE AT

hpe.com/info/linux-serviceguard-docs

Make the right purchase decision.
Contact our presales specialists.



Chat



Email



Call



Get updates

© Copyright 2013, 2015–2017, 2020–2021 Hewlett Packard Enterprise Development LP. The information contained herein is subject to change without notice. The only warranties for Hewlett Packard Enterprise products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Hewlett Packard Enterprise shall not be liable for technical or editorial errors or omissions contained herein.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries. Red Hat is a registered trademark of Red Hat, Inc. in the United States and other countries. VMware is a registered trademark or trademark of VMware, Inc. and its subsidiaries in the United States and other jurisdictions. All third-party marks are property of their respective owners.