Red Hat Enterprise Linux 6 Virtualization Security Guide

Securing your virtual environment

Scott Radvan

Red Hat Enterprise Linux 6 Virtualization Security Guide

Securing your virtual environment

Scott Radvan
Red Hat, Inc. Engineering Content Services
sradvan@redhat.com

Keywords

Abstract
This guide provides an overview of virtualization security technologies provided by Red Hat. It also provides recommendations for securing hosts, guests, and shared infrastructure and resources in virtualized environments.

Table of Contents

Preface

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the Liberation Fonts set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keys and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a key, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from an individual key by the plus sign that connects each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F2`** to switch to a virtual terminal.

The first example highlights a particular key to press. The second example highlights a key combination: a set of three keys pressed simultaneously.

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`mono-spaced bold`**. For example:

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **`Buttons`** tab, click the **`Left-handed mouse`** check box and click **`Close`** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications** → **Accessories** →

Character Map from the main menu bar. Next, choose **Search** → **Find...** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

To connect to a remote machine using ssh, type **ssh *username*@*domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.

The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.

To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

Publican is a *DocBook* publishing system.

1.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }
}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' will not cause data loss but may cause irritation and frustration.

Warning

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

2. Getting Help and Giving Feedback

2.1. Do You Need Help?

If you experience difficulty with a procedure described in this documentation, visit the Red Hat Customer Portal at http://access.redhat.com. Through the customer portal, you can:

- search or browse through a knowledgebase of technical support articles about Red Hat products.
- submit a support case to Red Hat Global Support Services (GSS).
- access other product documentation.

Red Hat also hosts a large number of electronic mailing lists for discussion of Red Hat software and technology. You can find a list of publicly available mailing lists at https://www.redhat.com/mailman/listinfo. Click on the name of any mailing list to subscribe to that list or to access the list archives.

2.2. We Need Feedback!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla: http://bugzilla.redhat.com/ against the product **Red Hat Enterprise Linux 6.**

When submitting a bug report, be sure to mention the manual's identifier: *doc-Virtualization_Security_Guide*

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

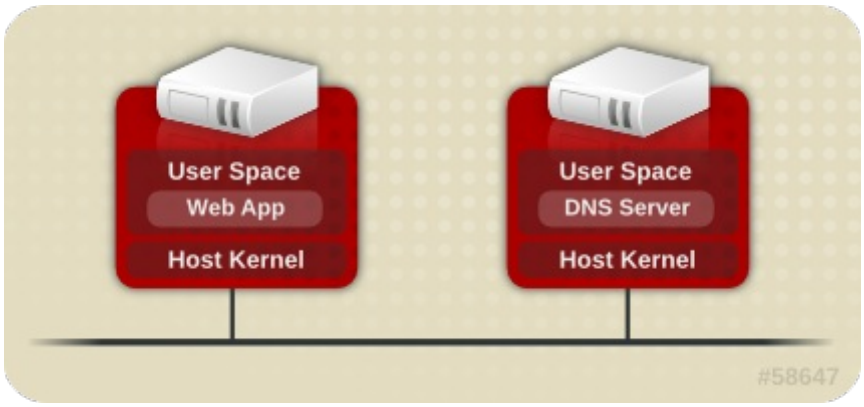
Chapter 1. Introduction

1.1. Virtualized and Non-Virtualized Environments

A virtualized environment presents opportunities for both the discovery of new attack vectors and the refinement of existing exploits that may not previously have presented value to an attacker. It is therefore important to take steps to ensure the security of both the physical hosts and the guests running on them when creating and maintaining virtual machines.

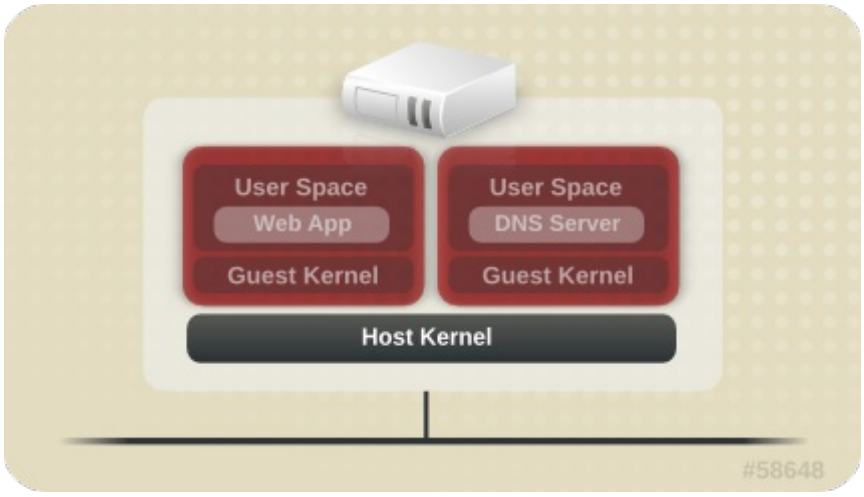
Non-Virtualized Environment

In a non-virtualized environment, hosts are separated from each other physically and each host has a self-contained environment, consisting of services such as a web server, or a DNS server. These services communicate directly to their own user space, host kernel and physical host, offering their services directly to the network. The following image represents a non-virtualized environment:



Virtualized Environment

In a virtualized environment, several operating systems can be housed (as "guests") within a single host kernel and physical host. The following image represents a virtualized environment:



When services are not virtualized, machines are physically separated. Any exploit is therefore usually contained to the affected machine, with the obvious exception of network attacks. When services are grouped together in a virtualized environment, extra vulnerabilities emerge in the system. If there is a security flaw in the hypervisor that can be exploited by a guest instance, this guest may be able to not only attack the host, but also other guests running on that host. This is not theoretical; attacks already exist on hypervisors. These attacks can extend beyond the guest instance and could expose other guests to attack.

1.2. Why Virtualization Security Matters

Deploying virtualization in your infrastructure provides many benefits but can also introduce new risks. Virtualized resources and services should be deployed with the following security considerations:

- The host/hypervisor become prime targets; in effect, they are often a single point of failure for guests and data.
- Virtual machines can interfere with each other in undesirable ways.
- Resources and services can become difficult to track and maintain; with rapid deployment of virtualized systems comes an increased need for management of resources, including sufficient patching, monitoring and maintenance.
- There may be a lack of knowledge, gaps in skill sets, and minimal experience among technical staff. This is often a gateway to vulnerabilities.
- Resources such as storage can be spread across, and dependent upon, several machines. This can lead to overly complex environments, and poorly-managed and maintained systems.
- Virtualization does not remove any of the traditional security risks present in your environment; the entire solution stack, not just the virtualization layer, must be secured.

This guide aims to assist you in mitigating your security risks by offering a number of virtualization recommended practices for Red Hat Enterprise Linux and Red Hat Enterprise Virtualization that will help you secure your virtualized infrastructure.

1.3. Leveraging SELinux with sVirt

sVirt integrates virtualization into the existing security framework provided by SELinux (Security-Enhanced Linux), applying *Mandatory Access Control* (MAC) to virtual machines. The main objective of sVirt is to protect hosts and guests from attacks via security vulnerabilities in the hypervisor. SELinux secures a system by applying access policy across different processes. sVirt extends this capability to hosts and guests by treating each guest as a process, allowing administrators to apply similar policies designed to prevent malicious guests from accessing restricted resources. For more information on sVirt, refer to Chapter 4, *sVirt*.

1.4. Further Resources

While the guide you are reading is focused on securing your virtual environment, several other virtualization guides are available in the Red Hat documentation suite.

Please refer to https://access.redhat.com/knowledge/docs/Red_Hat_Enterprise_Linux/ for the following:

- **Virtualization Getting Started Guide**
  - An introduction to virtualization concepts, advantages, and tools, and an overview of Red Hat virtualization documentation and products.
- **Virtualization Host Configuration and Guest Installation Guide**
  - A guide to installing virtualization software and configuring guest machines on a virtualization host.
- **Virtualization Administration Guide**
  - A guide covering administration of hosts, networking, storage, device and guest management, using either virt-manager or virsh, including a libvirt and qemu reference and troubleshooting information.
- **Virtualization Tuning and Optimization Guide**

- A guide that covers KVM and virtualization performance, with tips and suggestions for making full use of KVM performance features and options for your host systems and virtualized guests.

- **V2V Guide**
  - A guide to importing virtual machines from foreign hypervisors to Red Hat Enterprise Virtualization.
- **Hypervisor Deployment Guide**
  - A guide to obtaining, deploying, and configuring the Red Hat Enterprise Virtualization Hypervisor.

Chapter 2. Host Security

2.1. Why Host Security Matters

When deploying virtualization technologies, the security of the host should be paramount. The Red Hat Enterprise Linux host system is responsible for managing and controlling access to the physical devices, storage and network as well as all virtualized guests themselves. If the host system were to be compromised, not only would the host system be vulnerable, but so would the guests and their data.

Virtualized guests are only as secure as their host system; securing the Red Hat Enterprise Linux host system is the first step towards ensuring a secure virtualization platform.

2.2. Host Security Recommended Practices for Red Hat Enterprise Linux

With host security being such a critical part of a secure virtualization infrastructure, the following recommended practices should serve as a starting point for securing a Red Hat Enterprise Linux host system:

- Run only the services necessary to support the use and management of your guest systems. If you need to provide additional services, such as file or print services, you should consider running those services in a Red Hat Enterprise Linux guest.
- Limit direct access to the system to only those users who have a need to manage the system. Consider disallowing shared root access and instead use tools such as **sudo** to grant privileged access to administrators based on their administrative roles.
- Ensure that SELinux is configured properly for your installation and is operating in enforcing mode. Besides being a good security practice, the advanced virtualization security functionality provided by sVirt relies on SELinux. Refer to Chapter 4, *sVirt* for more information on SELinux and sVirt.
- Ensure that auditing is enabled on the host system and that libvirt is configured to emit audit records. When auditing is enabled, libvirt will generate audit records for changes to guest configuration as well start/stop events which help you track the guest's state. In addition to the standard audit log inspection tools, the libvirt audit events can also be viewed using the specialized auvirt tool.
- Ensure that any remote management of the system takes place only over secured network channels. Tools such as SSH and network protocols such as TLS or SSL provide both authentication and data encryption to help ensure that only approved administrators can manage the system remotely.
- Ensure that the firewall is configured properly for your installation and is activated at boot. Only those network ports needed for the use and management of the system should be allowed.
- Refrain from granting guests direct access to entire disks or block devices (for example, **`/dev/sdb`**); instead, you should use partitions (for example, **`/dev/sdb1`**) or LVM volumes for guest storage.

Note

The objective of this guide is to explain the unique security related challenges, vulnerabilities, and solutions that are present in most virtualized environments, and the recommended method of addressing them. However, there are a number of recommended practices to follow when securing a Red Hat Enterprise Linux system that apply regardless of whether the system is a standalone, virtualization host, or guest instance. These recommended practices include procedures such as system updates, password security, encryption, and firewall configuration. This information is discussed in more detail in the *Red Hat Enterprise Linux Security Guide* which can be found at https://access.redhat.com/knowledge/docs/.

2.2.1. Special Considerations for Public Cloud Operators

Public cloud service providers are exposed to a number of security risks beyond that of the traditional virtualization user. Virtual guest isolation, both between the host and guest as well as between guests, is critical due to the threat of malicious guests and the requirements on customer data confidentiality and integrity across the virtualization infrastructure.

In addition to the Red Hat Enterprise Linux virtualization recommended practices previously listed, public cloud operators should also consider the following items:

- Disallow any direct hardware access from the guest. PCI, USB, FireWire, Thunderbolt, eSATA and other device passthrough mechanisms not only make management difficult, but often rely on the underlying hardware to enforce separation between the guests.
- Network traffic should be separated such that the cloud operator's private management network is isolated from the customer guest network, helping to ensure that the guests can not access the host systems over the network. The network should be further isolated such that customer guest systems run in a private, virtualized network so that one customer can not access another customer's guest systems directly via the cloud provider's internal network.

2.3. Host Security Recommended Practices for Red Hat Enterprise Virtualization

2.3.1. Red Hat Enterprise Virtualization Network Ports

Red Hat Enterprise Virtualization uses various network ports for management and other virtualization features. These ports must be open for Red Hat Enterprise Linux to function as a host with Red Hat Enterprise Virtualization. The list below covers ports and their usage by Red Hat Enterprise Virtualization:

- Incoming ICMP echo requests and outgoing ICMP echo replies must be allowed.
- Port 22 (TCP) should be open for SSH access and the initial installation.
- Port 161 (UDP) is required for SNMP (Simple Network Management Protocol).
- Ports 5634 to 6166 (TCP) are used for guest console access.
- Ports 8080 or 8443 (TCP), depending on the security settings on the Manager, are used by the *vdsm-reg* service to communicate information about the host.
- Port 16514 (TCP) is used to support migration communication generated by libvirt.
- Ports 49152 to 49216 (TCP) are used for migrations. Migration may use any port in this range depending on the number of concurrent migrations occurring.
- Port 54321 (TCP) is used by default, by *VDSM* for management, storage and inter-host

communication. This port can be modified.

Warning

Take special care to filter SNMP on port 161 (UDP) at the border of your network unless it is absolutely necessary to externally manage devices.

Chapter 3. Guest Security

3.1. Why Guest Security Matters

While the security of the host system is critical in ensuring the security of the guests running on the host, it does not remove the need for properly securing the individual guest machines. All of the security risks associated with a conventional, non-virtualized system still exist when the system is run as a virtualized guest. Any resources accessible to the guest system, such as critical business data or sensitive customer information, could be made vulnerable if the guest system were to be compromised.

3.2. Guest Security Recommended Practices

All of the recommended practices for securing a Red Hat Enterprise Linux system documented in the *Red Hat Enterprise Linux Security Guide* apply to the conventional, non-virtualized systems as well as those systems installed as a virtualized guest. However, there are a few security practices which are of critical importance when running in a virtualized environment:

- With all management of the guest likely taking place remotely, ensure that the management of the system takes place only over secured network channels. Tools such as SSH and network protocols such as TLS or SSL provide both authentication and data encryption to ensure that only approved administrators can manage the system remotely.
- Some virtualization technologies use special guest agents or drivers to enable some virtualization specific features. Ensure that these agents and applications are secured using the standard Red Hat Enterprise Linux security features, e.g. SELinux.
- In virtualized environments there is a greater risk of sensitive data being accessed outside the protection boundaries of the guest system. Protect stored sensitive data using encryption tools such as dm-crypt and GnuPG; although special care needs to be taken to ensure the confidentiality of the encryption keys.

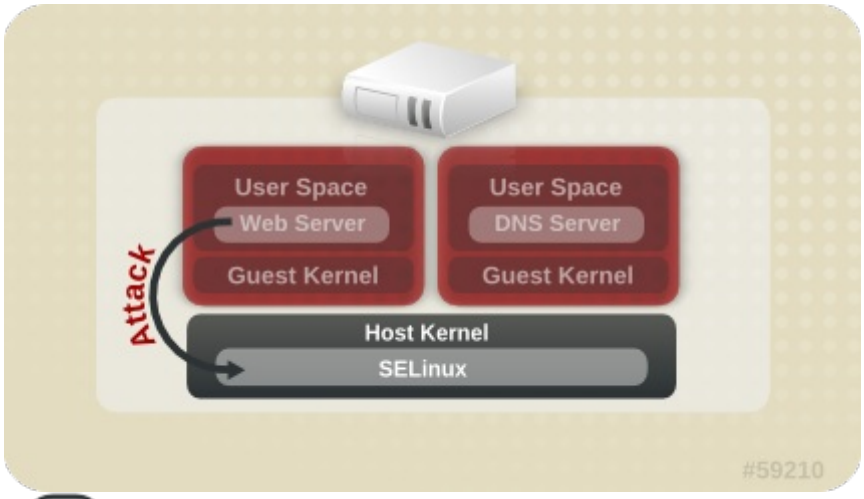
Chapter 4. sVirt

4.1. Introduction

Since virtual machines under KVM are implemented as Linux processes, KVM leverages the standard Linux security model to provide isolation and resource controls. The Linux kernel includes SELinux (Security-Enhanced Linux), a project developed by the US National Security Agency to add mandatory access control (MAC), multi-level security (MLS) and multi-category security (MCS) through a flexible and customizable security policy. SELinux provides strict resource isolation and confinement for processes running on top of the Linux kernel, including virtual machine processes. The sVirt project builds upon SELinux to further facilitate virtual machine isolation and controlled sharing. For example, fine-grained permissions could be applied to group virtual machines together to share resources.

From a security point of view, the hypervisor is a tempting target for attackers, as a compromised hypervisor could lead to the compromise of all virtual machines running on the host system. Integrating SELinux into the virtualization technologies helps improve the security of the hypervisor against malicious virtual machines trying to gain access to the host system or other virtual machines.

Refer to the following image which represents isolated guests, limiting the ability for a compromised hypervisor (or guest) to launch further attacks, or to extend to another instance:



Note

For more information on SELinux, refer to *Red Hat Enterprise Linux Security-Enhanced Linux* at https://access.redhat.com/knowledge/docs/.

4.2. SELinux and Mandatory Access Control (MAC)

Security-Enhanced Linux (SELinux) is an implementation of MAC in the Linux kernel, checking for allowed operations after standard discretionary access controls (DAC) are checked. SELinux can enforce a user customizable security policy on running processes and their actions, including their attempts to access file system objects. Enabled by default in Red Hat Enterprise Linux, SELinux limits the scope of potential damage that can result from the exploitation of vulnerabilities in applications and system services, such as the hypervisor.

sVirt integrates with libvirt, a virtualization management abstraction layer, to provide a MAC framework for virtual machines. This architecture allows all virtualization platforms supported by libvirt and all MAC implementations supported by sVirt to interoperate.

4.3. sVirt Configuration

SELinux Booleans are variables that can be toggled on or off, quickly enabling or disabling features or other special conditions. The following table shows the SELinux Boolean values that affect KVM when launched by libvirt.

KVM SELinux Booleans

SELinux Boolean	Description
virt_use_comm	Allow virt to use serial/parallel communication ports.
virt_use_fusefs	Allow virt to read FUSE mounted files.
virt_use_nfs	Allow virt to manage NFS mounted files.
virt_use_samba	Allow virt to manage CIFS mounted files.
virt_use_sanlock	Allow confined virtual guests to interact with the sanlock.
virt_use_sysfs	Allow virt to manage device configuration (PCI).
virt_use_usb	Allow virt to use USB devices.
virt_use_xserver	Allow virtual machine to interact with the X Window System.

Note

For more information on SELinux Booleans, refer to *Red Hat Enterprise Linux Security-Enhanced Linux* and *Red Hat Enterprise Linux Managing Confined Services* at https://access.redhat.com/knowledge/docs/.

4.4. sVirt Labeling

Like other services under the protection of SELinux, sVirt uses process based mechanisms, labels and restrictions to provide extra security and control over guest instances. Labels are applied automatically to resources on the system based on the currently running virtual machines (dynamic), but can also be manually specified by the administrator (static), to meet any specific requirements that may exist.

4.4.1. Types of sVirt Labels

The following table outlines the different sVirt labels that can be assigned to resources such as virtual machine processes, image files and shared content:

Table 4.1. sVirt Labels

Type	SELinux Context	Description/Effect
Virtual Machine Processes	system_u:system_r:svirt_t:***MCS1***	***MCS1*** is a randomly selected field. Currently approximately 500,000 labels are supported.
Virtual Machine Image	system_u:object_r:svirt_image_t: ***MCS1***	Only *svirt_t* processes with the same ***MCS1*** fields are able to read/write these image files and devices.
Virtual Machine Shared Read/Write Content	system_u:object_r:svirt_image_t: s0	All *svirt_t* processes are allowed to write to the *svirt_image_t:s0* files and devices.
Virtual Machine Shared Shared Read Only content	system_u:object_r:svirt_content _t:s0	All *svirt_t* processes are able to read files/devices with this label.
Virtual Machine Image	system_u:object_r:virt_content_t :s0	System default label used when an image exits. No *svirt_t* virtual processes are allowed to read files/devices with this label.

4.4.2. Dynamic Configuration

Dynamic label configuration is the default labeling option when using sVirt with SELinux. Refer to the following example which demonstrates dynamic labeling:

```
# ps -eZ | grep qemu-kvm

system_u:system_r:svirt_t:s0:c87,c520 27950 ?  00:00:17 qemu-kvm
```

In this example, the **qemu-kvm** process has a base label of **system_u:system_r:svirt_t:s0**. The libvirt system has generated a unique MCS label of **c87,c520** for this process. The base label and the MCS label are combined to form the complete security label for the process. Likewise, libvirt takes the same MCS label and base label to form the image label. This image label is then automatically applied to all host files that the VM is required to access, such as disk images, disk devices, PCI devices, USB devices, and kernel/initrd files. Each process is isolated from other virtual machines with different labels.

The following example shows the virtual machine's unique security label (with a corresponding MCS label of **c87,c520** in this case) as applied to the guest disk image file in **/var/lib/libvirt/images**:

```
# ls -lZ /var/lib/libvirt/images/*

  system_u:object_r:svirt_image_t:s0:c87,c520   image1
```

The following example shows dynamic labeling in the XML configuration for the guest:

```
<seclabel type='dynamic' model='selinux' relabel='yes'>
  <label>system_u:system_r:svirt_t:s0:c87,c520</label>
  <imagelabel>system_u:object_r:svirt_image_t:s0:c87,c520</imagelabel>
</seclabel>
```

4.4.3. Dynamic Configuration with Base Labeling

To override the default base security label in dynamic mode, the <baselabel> option can be configured manually in the XML guest configuration, as shown in this example:

```
<seclabel type='dynamic' model='selinux' relabel='yes'>
  <baselabel>system_u:system_r:svirt_custom_t:s0</baselabel>
  <label>system_u:system_r:svirt_custom_t:s0:c87,c520</label>
  <imagelabel>system_u:object_r:svirt_image_t:s0:c87,c520</imagelabel>
</seclabel>
```

4.4.4. Static Configuration with Dynamic Resource Labeling

Some applications require full control over the generation of security labels but still require libvirt to take care of resource labeling. The following guest XML configuration demonstrates an example of static configuration with dynamic resource labeling:

```
<seclabel type='static' model='selinux' relabel='yes'>
  <label>system_u:system_r:svirt_custom_t:s0:c87,c520</label>
</seclabel>
```

4.4.5. Static Configuration without Resource Labeling

Primarily used in MLS (multi-level security) or otherwise strictly controlled environments, static configuration without resource relabeling is possible. Static labels allow the administrator to select a specific label, including the MCS/MLS field, for a virtual machine. Administrators who run statically-labeled virtual machines are responsible for setting the correct label on the image files. The virtual machine will always be started with that label, and the sVirt system will never modify the label of a statically-labelled virtual machine's content. The following guest XML configuration demonstrates an example of this scenario:

```
<seclabel type='static' model='selinux' relabel='no'>
  <label>system_u:system_r:svirt_custom_t:s0:c87,c520</label>
</seclabel>
```

Chapter 5. Network Security in a Virtualized Environment

5.1. Network Security Overview

In almost all situations, the network is the only way to access systems, applications and management interfaces. As networking plays such a critical role in the management of virtualized systems and the availability of their hosted applications, it is very important to ensure that the network channels both to and from the virtualized systems are secure.

Securing the network allows administrators to control access and protect sensitive data from information leaks and tampering.

5.2. Network Security Recommended Practices

Network security is a critical part of a secure virtualization infrastructure. Refer to the following recommended practices for securing the network:

- Ensure that remote management of the system takes place only over secured network channels. Tools such as SSH and network protocols such as TLS or SSL provide both authentication and data encryption to assist with secure and controlled access to systems.
- Ensure that guest applications transferring sensitive data do so over secured network channels. If protocols such as TLS or SSL are not available, consider using IPsec.
- Configure firewalls and ensure they are activated at boot. Only those network ports needed for the use and management of the system should be allowed. Test and review firewall rules regularly.

5.2.1. Securing Connectivity to Spice

The Spice remote desktop protocol supports SSL/TLS which should be enabled for all of the Spice communication channels (main, display, inputs, cursor, playback, record).

5.2.2. Securing Connectivity to Storage

You can connect virtualized systems to networked storage in many different ways. Each approach presents different security benefits and concerns, however the same security principles apply to each: authenticate the remote store pool before use, and protect the confidentiality and integrity of the data while it is being transferred.

The data must also remain secure while it is stored. It is considered a recommended practice to encrypt and/or digitally sign the data before storing it.

Chapter 6. Further Information

6.1. Contributors

Thanks go to the following people for enabling the creation of this guide:

- Paul Moore - Red Hat Engineering
- Kurt Seifried - Red Hat Engineering
- David Jorm - Red Hat Engineering

6.2. Other Resources

SELinux and sVirt

Further information on SELinux and sVirt:

- Main SELinux website: http://www.nsa.gov/research/selinux/index.shtml.
- SELinux documentation: http://www.nsa.gov/research/selinux/docs.shtml.
- Main sVirt website: http://selinuxproject.org/page/SVirt.
- Dan Walsh's blog: http://danwalsh.livejournal.com/.
- The unofficial SELinux FAQ: http://www.crypt.gen.nz/selinux/faq.html.

Virtualization Security

Further information on virtualization security:

- NIST (National Institute of Standards and Technology) full virtualization security guidelines: http://www.nist.gov/itl/csd/virtual-020111.cfm.

Revision History

Revision 0.4-17 **Mon Feb 18 2013** **Scott Radvan**
Version for 6.4 GA release.

Revision 0.4-16 **Sun Feb 17 2013** **Scott Radvan**
Minor wording updates.

Revision 0.4-15 **Wed Feb 13 2013** **Scott Radvan**
Minor wording updates.

Revision 0.4-14 **Tue Jan 22 2013** **Scott Radvan**
Add `Further Resources` to Introduction.

Revision 0.4-13 **Tue Jan 22 2013** **Scott Radvan**
Review for 6.4 release.

Revision 0.4-12 **Sun Nov 25 2012** **Scott Radvan**
Boolean descriptions now match those from the 'semanage boolean -l' command.

Revision 0.4-11 **Sun Oct 28 2012** **Scott Radvan**
fix subtitle

Revision 0.4-10 **Tue Oct 16 2012** **Scott Radvan**
Review for 6.4 accuracy.

Revision 0.4-9 **Thu Sep 27 2012** **Scott Radvan**
Correct links to new docs site.

Revision 0.4-08 **Fri Jul 20 2012** **Laura Bailey**
Ensuring that the document complies with word usage policy.

Revision 0.4-07 **Sun Jun 17 2012** **Scott Radvan**
Publish for 6.3 GA release.

Revision 0.4-06 **Fri Jun 15 2012** **Scott Radvan**
Remove draft watermark, prepare for 6.3 branch.

Revision 0.4-05 **Fri Jun 08 2012** **Scott Radvan**
Static labeling example output re-ordered.

Revision 0.4-04 **Fri Jun 08 2012** **Scott Radvan**
Minor fixes from SME review.

Revision 0.4-03 **Tue Jun 05 2012** **Scott Radvan**
Add fixes from QE review (BZ#828032).

Revision 0.4-02 **Mon Jun 04 2012** **Scott Radvan**
Include Further Information chapter, add contributors and links.

Revision 0.4-01 **Mon Jun 04 2012** **Scott Radvan**
Bump major, performed proof and minor edits.

Revision 0.2-06 **Wed May 30 2012** **Scott Radvan**
Initial work on 'Chapter 5 - Network Security in a Virtualized Environment'.

Revision 0.2-05 **Tue May 15 2012** **Scott Radvan**
Add note about root access in guests not being desirable, to Guest_Security.xml

Revision 0.2-04 **Mon May 14 2012** **Scott Radvan**
Add warning about SNMP at network border.

Revision 0.2-03 **Mon May 14 2012** **Scott Radvan**
Add `Guest Security' sections. Expand Network Ports to specify Echo Request (ping) and TCP/UDP. Include additional passthrough mechanisms for public cloud recommendations. Expand Booleans to specify mounted file systems.

Revision 0.2-02 **Tue May 08 2012** **Scott Radvan**
s/Overview/Introduction

Revision 0.2-01 **Tue May 08 2012** **Scott Radvan**
Bump major. Added SME content for `Host Security' chapter and performed edits/proof.

Revision 0.1-15 **Thu Mar 29 2012** **Scott Radvan**
Import `Red Hat Enterprise Virtualization Network Ports' section to Host_Security.xml.

Revision 0.1-14 **Thu Mar 29 2012** **Scott Radvan**
Rename titles for Best Practices sections. Re-wording of Best Practices.

Revision 0.1-13 **Wed Mar 28 2012** **Scott Radvan**
Build with new version of publishing toolchain.

Revision 0.1-12 **Mon Mar 20 2012** **Scott Radvan**
Initial publish of new section: `1.2. Why Virtualization Security Matters' for SME review.

Revision 0.1-11 **Mon Mar 20 2012** **Scott Radvan**
Move `Best Host Security Practices for Red Hat Enterprise Linux' to `Host Security' chapter.

Revision 0.1-10 **Mon Mar 20 2012** **Scott Radvan**
Major SME-assisted restructure of TOC.

Revision 0.1-09 **Mon Feb 14 2012** **Scott Radvan**
Introduction wording.

Revision 0.1-08 **Mon Feb 13 2012** **Scott Radvan**
New version of publishing tool.

Revision 0.1-07 **Tue Feb 01 2012** **Scott Radvan**

Publish for SME review of sVirt chapter.

Revision 0.1-06 **Tue Jan 31 2012** **Scott Radvan**

Add draft watermark to denote development/internal status.

Revision 0.1-05 **Mon Jan 30 2012** **Scott Radvan**

Additions to Introduction. Arrange sVirt labeling sections. Add admonition regarding system hardening topics that this guide does not cover. Introduce dynamic labeling.

Revision 0.1-04 **Mon Jan 30 2012** **Scott Radvan**

Minor wording updates to sVirt chapter. Restructure and improve flow of Chapter 1: Introduction.

Revision 0.1-03 **Fri Jan 20 2012** **Scott Radvan**

Further restructure sVirt chapter. Incorporate initial developer feedback/fixes.

Revision 0.1-02 **Tue Jan 17 2012** **Scott Radvan**

Major restructure of document flow. Make file names consistent.

Revision 0.1-01 **Tue Jan 17 2012** **Scott Radvan**

Initial creation of book. Import existing text/remarks.