

Principios básicos de seguridad en sistemas anfitriones KVM Red Hat

Bibliografía:

Red Hat Enterprise Linux 7. Virtualization Deployment and Administration Guide.

https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/7/html/virtualization_deployment_and_administration_guide/index

SELinux User's Administrator's Guide - Red Hat Enterprise Linux 7.

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/selinux_users_and_administrators_guide/index

Documentación oficial proyecto Selinux: http://selinuxproject.org/page/Main_Page

Contenidos

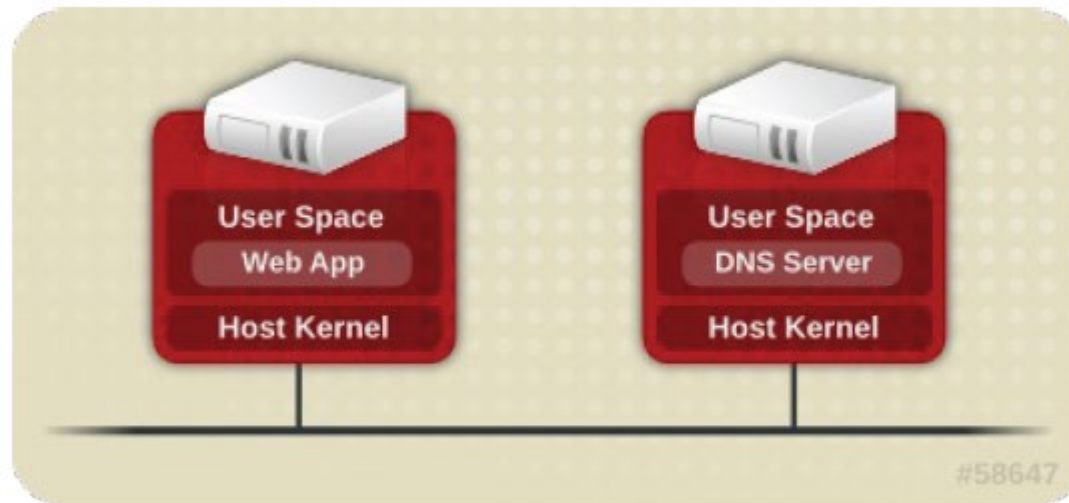
1. Buenas prácticas en el sistema anfitrión
2. Virtualización y el acceso controlado a los recursos.
 - 2.1. Fundamentos sobre *SELinux*.
 - 2.2. Virtualización y *SELinux*.
3. Virtualización y el control de tráfico de la red: cortafuego.

1. Buenas prácticas en el sistema anfitrión

- Ejecutar en él sólo los servicios necesarios.
- Protegerlo con *SELinux*.
 - # yum install policycoreutils-python
 - # yum install policycoreutils-gui
 - *SELinux* debe ejecutarse en modo “*enforcing*”
 - # getenforce
- No permitir que sea accedido por usuarios normales.

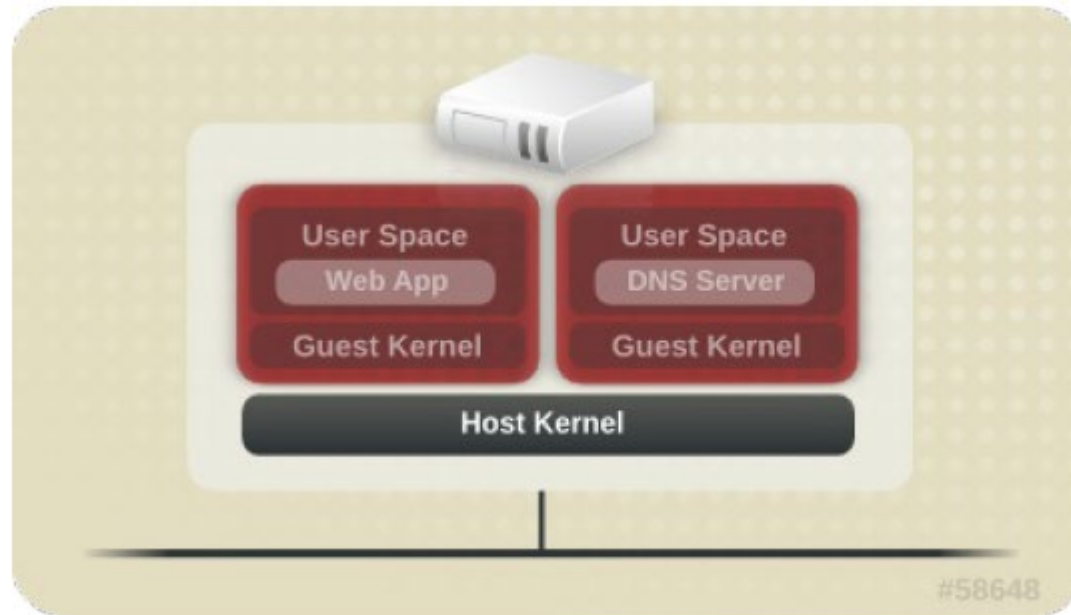
2. Virtualización y el acceso controlado a los recursos (I)

- **Entorno no virtualizado.** Los sistemas están físicamente separados. Cada sistema posee sus servicios cuyas configuraciones están en el mismo sistema.



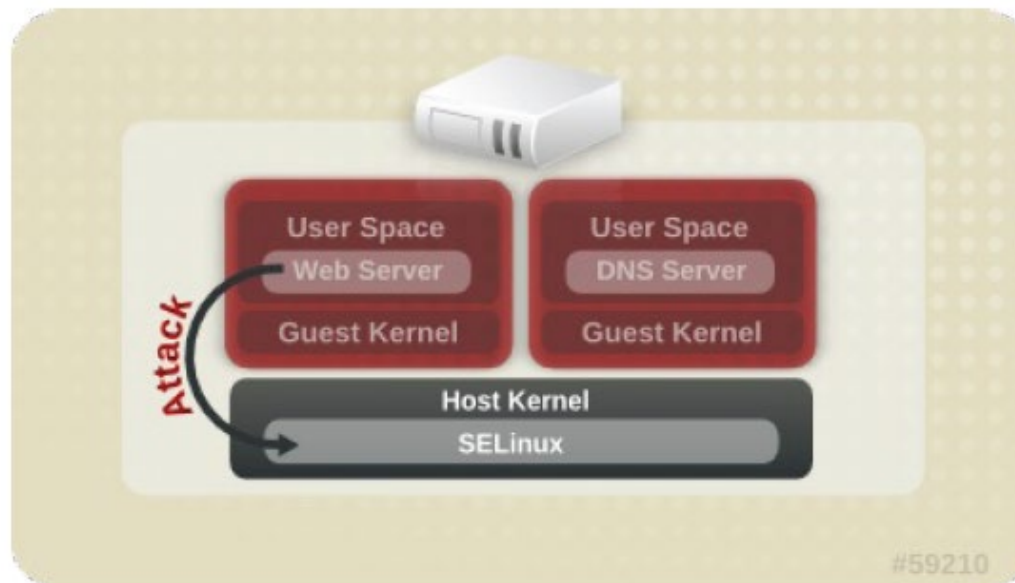
2. Virtualización y el acceso controlado a los recursos (II)

- **Entorno virtualizado KVM.** Varios sistemas operativos pueden estar instalados en un único sistema anfitrión haciendo uso del núcleo del sistema operativo anfitrión.



2. Virtualización y el acceso controlado a los recursos (III)

- **Entorno virtualizado KVM.** Surgen nuevas amenazas potenciales con respecto a un sistema no virtualizado.
 - Si hay un fallo en la seguridad del hipervisor ésta puede afectar, a los sistemas invitados.
 - Un fallo en la seguridad puede ser aprovechado para que un sistema invitado ataque a otro sistema invitado

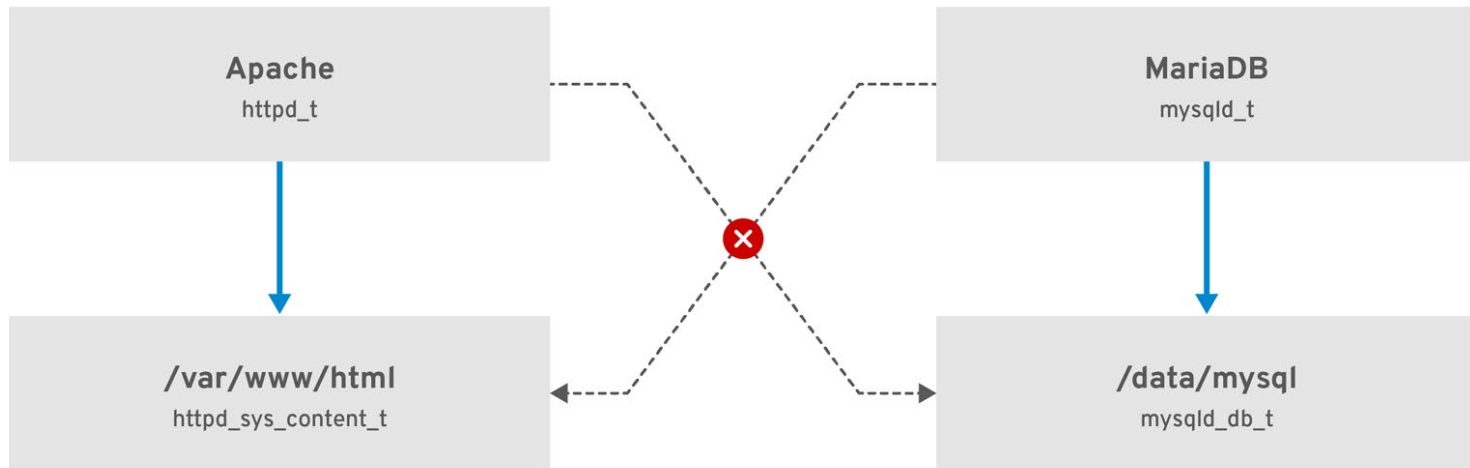


2.1. Fundamentos sobre *SElinux* (I)

- **Discretionary Access Control (DAC).** *Modelo básico de seguridad utilizado sistemas Linux, basados en categorías de usuarios (u,g,o) y tipos de acceso (r,w,x), es insuficiente para proporcionar la seguridad adecuada.*
- *SElinux* es un caso de implementación del modelo **Mandatory Access Control**. Su principio de funcionamiento:
“Puede <proceso> Hacer <acción> En <objeto>”
- *En sistemas Linux primero se aplica el control de acceso básico (DAC) y si el acceso es permitido, entonces se aplica el control de acceso SELinux.*

2.1. Fundamentos sobre *SELinux* (I)

- Un ejemplo que ilustra el funcionamiento de SELinux*



RHEL_467048_0218

2.1. Fundamentos sobre *SElinux* (I)

- En *SElinux* el control de acceso se realiza en función de políticas que definen quién tiene acceso (sujetos) y a qué recursos (objetos).
- En *SElinux*:
 - Sujetos: son los procesos.
 - Objetos: son los discos, la memoria, los canales de comunicación, los archivos, etc.
 - Cada sujeto u objeto posee un conjunto de atributos de seguridad (contexto).
 - Servidor de seguridad lleva a cabo el control de seguridad mediante reglas (política de control de acceso).

2.1. Fundamentos sobre *SElinux* (II)

- *SElinux* soporta varios modelos MAC:
 - Para sistemas de propósito general existen tres tipos:
 - Strict
 - Targeted
 - Multi-Level Security (MLS)
 - Multi-Category Security (MCS)
- *SElinux* ya forma parte de las distribuciones *Red Hat Enterprise Linux* y sus derivadas (*Fedora* y *CentOS*).
- *SElinux*
 - Clasifica los procesos y recursos de un sistema mediante el concepto de Tipo, por ejemplo *home_t*.
 - Los procesos y recursos poseen un contexto (etiqueta). Un contexto se define mediante 4 atributos (3 obligatorios y 1 opcional). Un atributo (el tercero) es el Tipo (obligatorio).
 - El control de acceso se realiza mediante reglas (políticas de acceso).
`allow firefox_t user_home_t : file { read write };`

2.1. Fundamentos sobre *SElinux* (III)

- Las políticas implementadas:
 - Las escriben las distribuciones mediante un proceso de retroalimentación entre expertos en seguridad y usuarios de aplicaciones.
 - *Reference Policy* marco en el que están definidas.
 - *Tresys Technology* se encarga del mantenimiento.
- *SElinux* no es un cortafuego.
- *SElinux* debe utilizarse en todo tipo de sistemas Linux. Modos de ejecución:
 - *Enforcing*.
 - *Permissive*.
 - *Disabled*.
- ¿Cómo saber en qué modo se está ejecutando?
 - # sestatus
 - # setenforce 0

2.1. Fundamentos sobre *SElinux* (IV)

- Componentes básicos:
 - Usuarios (`_u`). Los usuarios *SElinux* no coinciden con los usuarios del sistema anfitrión (`user_u`, `system_u`, ...)
 - Roles (`_r`). El papel que juega un usuario *SElinux* en el sistema (`sysadm_r`, `user_r`, ...)
 - Tipos (`_t`). Todas las entidades controladas por *SElinux* se clasifican en categorías o tipos (`file_t`, `user_home_t`, ...)
 - Contextos (etiquetas): `user:role:type:range`
 - `system_u:system_r:xserver_t`
 - `system_u:system_r:xserver_t:s0-s0:c0.c1023`
 - `system_u:system_r:xserver_t:SystemLow-SystemHigh`
 - Reglas:

```
allow user_t user_home_t:file { create read write unlink };
```

2.1. Fundamentos sobre *SElinux* (V)

- Configurando el modo de funcionamiento
 - Archivo `/etc/selinux/config`
 - En vivo

```
# getenforce
Permissive

# setenforce 1
# getenforce
Enforcing

# setenforce 0
# getenforce
Permissive
```

2.1. Fundamentos sobre *SELinux* (VI)

- Manejando contextos:
 - Visualizando:
 - # `id -Z`
 - # `ls -Z /bin/bash`
 - # `ps -Z`
 - Modificando:
 - # `chcon -t user_home_t /tmp/myfile`

 - # `semanage fcontext -a -t user_home_t /var/cache/myfile`
 - # `restorecon /var/cache/myfile`

 - # `newrole -r system_r -t unconfined_t`
 - # `runcon -u system_u /bin/bash`

 - # `restorecon /tmp/myfile`

2.1. Fundamentos sobre *SElinux* (VII)

- Arrancado servicios bajo el control de *SElinux*

- Arranque sin el control *SElinux*

```
# /etc/init.d/ssh start
```

```
* Starting OpenBSD Secure Shell server sshd [ OK ]
```

```
# ps auxZ | grep sshd
```

```
unconfined_u:system_r:sshd_t:s0-s0:c0.c255 root 1781 0.0 0.0 48940 1176 ? Ss 22:40 0:00 /usr/sbin/sshd
```

- Arranque con el control *SElinux*

```
# run_init /etc/init.d/ssh
```

```
start Authenticating root.
```

```
Password:
```

```
Starting OpenBSD Secure Shell server sshd [ OK ]
```

```
# ps auxZ | grep sshd
```

```
system_u:system_r:sshd_t:s0-s0:c0.c255 root 2017 0.0 0.0 48940 1176 ? Ss 22:46 0:00 /usr/sbin/sshd
```

2.1. Fundamentos sobre *SElinux* (VIII)

- Ficheros de configuración globales `/etc/selinux`
 - `/etc/selinux/config`
 - `/etc/selinux/semanage.conf`
 - `/etc/selinux/restorecond.conf`
 - `/etc/sestatus.conf`
 - `/etc/security/sepermit.conf`
- Ficheros asociados a la política empleada
 - Ficheros de configuración de la política básicos
 - `/etc/selinux/<policy_name>/module`
 - Ficheros “run-time” asociados a la política empleada
- Ficheros de configuración del núcleo `/selinux`

2.1. Fundamentos sobre *SElinux* (IX)

- Ficheros asociados a la política empleada
 - Ficheros de configuración de la política básicos: `/etc/selinux/<policy_name>/module`

```
base.pp
base.linked
commit_num
file_contexts.template
file_contexts
homedir_template
file_contexts.homedirs
netfilter_contexts & netfilter.local
policy.kern
seusers.final and seusers
users_extra, users_extra.local and users.local
booleans.local
file_contexts.local
interfaces.local
nodes.local
ports.local
```

2.2. Virtualización y *SElinux*

- Entorno **sVirt**. Entorno de seguridad que permite que la seguridad del sistema anfitrión y los sistemas invitados estén bajo el control de *SElinux*.
 - Definición uniforme de contextos para el sistema anfitrión y sus sistemas invitados.
 - Aplicación uniforme de la reglas al sistema anfitrión y sus sistemas invitados.
- Se requiere el paquete `policycoreutils-python`
`yum install policycoreutils-python`

2.2. Virtualización y *SELinux*

- Etiquetas *sVirt*:

| Type/Description | SELinux Context |
|--|--------------------------------------|
| Virtualized guest processes. MCS1 is a random MCS field. Approximately 500,000 labels are supported. | system_u:system_r:svirt_t:MCS1 |
| Virtualized guest images. Only <i>svirt_t</i> processes with the same MCS fields can read/write these images. | system_u:object_r:svirt_image_t:MCS1 |
| Virtualized guest shared read/write content. All <i>svirt_t</i> processes can write to the <i>svirt_image_t:s0</i> files. | system_u:object_r:svirt_image_t:s0 |
| Virtualized guest shared read only content. All <i>svirt_t</i> processes can read these files/devices. | system_u:object_r:svirt_content_t:s0 |
| Virtualized guest images. Default label for when an image exits. No <i>svirt_t</i> virtual processes can read files/devices with this label. | system_u:object_r:virt_content_t:s0 |

2.2. Virtualización y *SELinux*

- Etiquetas *sVirt*
 - Cada proceso en un sistema invitado se aísla mediante un contexto (etiqueta) que posee un nivel (cuarto atributo de un contexto) que se genera de forma dinámica. Siendo diferente para cada sistema invitado.

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

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

2.2. Virtualización y *SELinux*

- Variables booleanas que controlan al KVM lanzado por *libvirt*:

| Entidad | Significado |
|-------------------------|--|
| virt_use_comm | Permite a virt el uso de comunicaciones series y paralelas |
| virt_use_fusefs | Permite a virt leer archivos fuse |
| virt_use_nfs | Permite a virt manejar sistemas de archivos NFS |
| virt_use_samba | Permite a virt manejar archivos CIFS |
| virt_use_sanlock | Permite a sanlock manejar archivos virt lib |
| virt_use_sysfs | Permite a virt manejar la configuración de dispositivos PCI |
| virt_use_xserver | Permite a las máquinas virtuales interactuar con el servidor X |
| virt_use_usb | Permite a virt utilizar dispositivos USB |

2.2. Virtualización y *SELinux*

- Manejando variables booleanas SELinux. Ejemplos de órdenes:

```
# semanage boolean -l
```

```
# getsebool -a
```

```
# getsebool virt_use_nfs
```

```
# setsebool virt_use_nfs on
```

```
# setsebool -P virt_use_nfs on
```

3. Virtualización y el control de tráfico de la red: cortafuego

- Paquetes ICMP, utilizados para la verificación de la red, deben ser aceptados. Si los paquetes ICMP no está permitidos, entonces, por ejemplo, no podrá realizar un ping a una máquina virtual.
- El puerto 22 debe estar abierto. Este puerto se utiliza para los acceso SSH y en la instalación.
- El puerto 80 o 443 (dependiendo de la configuración de seguridad en el *Manager RHEV*) se utiliza por el servicio *vdsm-reg* para intercambiar información sobre el sistema anfitrión.
- El rango de puertos 5634 - 6166 se utilizan para el acceso a las consolas de las máquinas virtuales, utilizando el protocolo SPICE.
- El rango de puertos 49152 – 49216 se utilizan en KVM para las operaciones de migración de máquinas virtuales. Cualquier puerto de este rango puede ser utilizado en este tipo de operaciones dependiendo del grado de concurrencia de estas.
- Se debe habilitar *IP forwarding* (`net.ipv4.ip_forward = 1`) para el uso de *bridge* por defecto y para *bridges* compartidos. En la instalación de los paquetes de instalación esta opción de configuración se habilita.
- No se requiere la habilitación de *IP forwarding* cuando se utilizan dispositivos hardware *bridges*. Cuando una máquina virtual se conecta a través de un *bridge* hardware, el tráfico se realiza en un nivel que no requiere configuración IP.