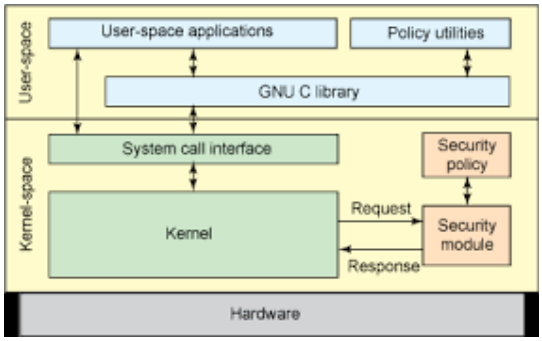
12. SELINUX

12.1.- Introducción

SELinux (**S**ecurity-**E**nhanced **Linux**) es un mecanismo de control de acceso mandatorio (MAC) implementado en el kernel de Linux. Desarrollado por la NSA en el 2000, se ha implementado y habilitado en la mayoría de las distribuciones Linux: Fedora, Debian, Ubuntu, CentOS, RHEL, Scientific Linux,...

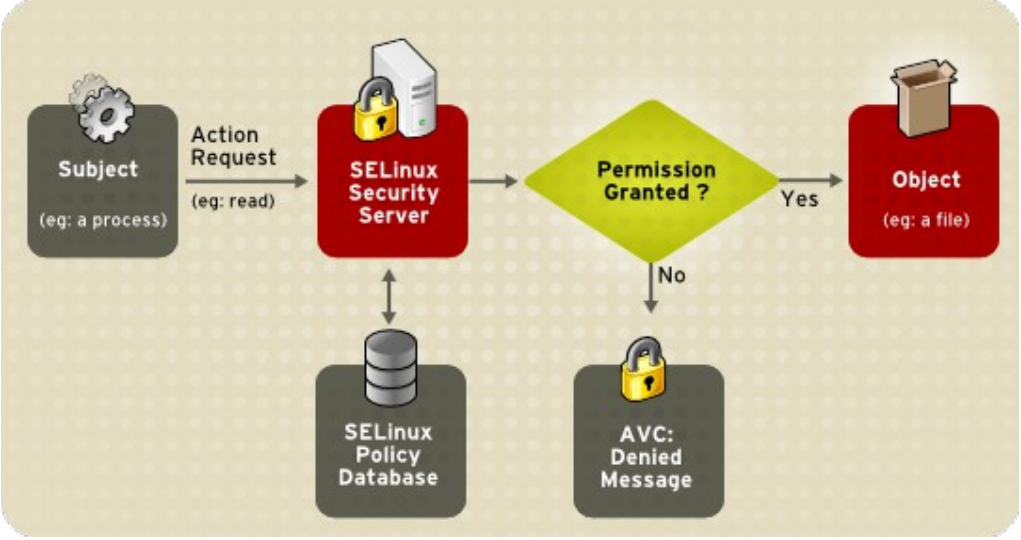
Un control de acceso mandatorio, es un modelo de privilegios mínimos, como el modo de funcionamiento *enforcing* de SELinux: todo es denegado y se definen políticas para dar a cada proceso/servicio del sistema sólo el acceso a los objetos (archivos, directorios, puertos, dispositivos) que necesite para funcionar.

Desde su origen ha tenido mala fama y se dice que reporta más problemas que beneficios pero un sistema con SELinux funcionando y bien configurado, es más seguro y fiable.



12.2.- Funcionamiento

SELinux añade una capa extra de seguridad haciendo que cada proceso del sistema se etiquete con un tipo, y se definen unas políticas o reglas de a qué objetos (archivos, directorios, puertos) puede acceder siendo de ese tipo, de forma que, si no hay regla, no hay acceso.



Funcionamiento de SELinux

A su vez, cada objeto tiene una "etiqueta SELinux" o contexto, formado por 4 campos: *user, role, type* y *level*, separados por el carácter dos puntos.

La política por defecto en un sistema CentOS es *targeted*, que afecta al campo *type*. Cuando hablamos de contexto SELinux, nos referiremos a este tercer campo de tipo. En esta política, una serie de procesos del sistema están bajo el control de SELinux: apache, dns, nfs, smb,…. y el resto de procesos aplica la seguridad estándar de Linux. Con esta politica se alcanza un control bastante eficaz y es sencillo de administrar.

La otra política posible es la de multinivel/multicategoría (MLS/MCS), bastante compleja de administrar y sólo adecuada para entornos donde la seguridad es algo muy crítico.

Con la instalación del sistema existen una serie de políticas por defecto ya definidas para los procesos que se van ampliando con la instalación de cualquier paquete.

Casi todos los comandos básicos como *ps*, *ls*, *cp*, … tienen la opción *-Z* que tiene que ver con SELinux.

12.2.1.- Modos de funcionamiento SELinux:

- *Enforcing*: SELinux funcionando "a pleno rendimiento", bloquea accesos y registra las violaciones a las políticas en el archivo */var/log/audit/audit.log*.
- *Permissive*: permite los accesos pero registra las violaciones.
- *Disabled*: totalmente desactivado, no funciona.

Para ver el modo de funcionamiento de SELinux en un momento dado en el sistema, usaremos el comando *getenforce*. Nos devolverá uno de los tres modos de funcionamiento.

En un momento dado, podemos cambiar el modo de funcionamiento de SELinux, usando:

setenforce <valor>

donde <*valor*> será:

- *0* o *Permissive*: para pasar de modo *Enforcing* a modo *Permissive*.
- *1* o *Enforcing*: para pasar de modo *Permissive* a modo *Enforcing*.

pero no podremos pasar a modo *Disabled* con este comando, ni estando en modo *Disabled*, pasar a uno de los otros dos modos.

El comando *setenforce* no es persistente, el modo de funcionamiento configurado en el sistema cuando arranca está en el archivo */etc/selinux/config*, en la línea *SELINUX=*<*modo*> donde <*modo*> es *enforcing*, *permissive* o *disabled*.

El archivo */etc/sysconfig/selinux*, es un enlace simbólico a */etc/selinux/config*.

Para que los cambios en este archivo entren en funcionamiento, hay que reiniciar el sistema. Si cambiamos el modo de funcionamiento a *Disabled* en este archivo, cuando el sistema se inicia, le quita todas las etiquetas SELinux a los archivos, directorios, …. al igual que si del modo *Disabled* pasamos a *Enforcing* o *Permissive*, cuando el sistema arranca, tiene que etiquetar todos los objetos y empleará un tiempo extra en dicho arranque.

SELinux debería estar siempre en modo de funcionamiento *Enforcing*. Si en un momento dado, tenemos problemas con algún servicio/proceso del sistema y no sabemos si es SELinux el causante, podemos pasar del modo *Enforcing* a *Permissive*. Si se soluciona, tenemos un problema con las políticas SELinux, falta añadir reglas o modificar booleanos para que el servicio/proceso tenga acceso a lo que debe o realmente no tiene que tener acceso a algo y el servicio/proceso no funciona adecuadamente. Si una vez puesto SELinux en modo *Permissive*, el problema no ha variado, hay que buscar la causa en otro sitio: firewall, configuración del servicio/proceso, … y debemos volver a poner SELinux en modo *Enforcing*.

12.3.- Contextos SELinux en archivos y directorios

El contexto de un directorio, se hereda en los archivos y sudirectorios que contiene, salvo que haya alguna política definida que lo modifique o hayamos copiado o movido el archivo o subdirectorio usando la opción *-a* para preservar los permisos.

Usando el comando *ls* con la opción *-Z* veremos la etiqueta SELinux de archivos y directorios.

12.3.1.- Gestionar reglas de contexto para archivos o directorios al sistema

Para añadir o modificar reglas de archivos o directorios a la política SELinux del sistema, utilizaremos el comando *semanage fcontext*.

Se utilizan expresiones regulares extendidas para especificar el path y los nombres de archivo. La que se usa de forma más habitual es ***(/.*)?***, que indica: opcionalmente un carácter / seguido de cualquier número de caracteres. Se pone detrás del nombre del directorio y hará que encaje en la expresión regular el propio directorio y todo su contenido de forma recursiva.

Opciones de *semanage fcontext* usadas:

- *-l*: lista todas las reglas existentes en la política del sistema.
- *-a -t <contexto_tipo> '<expresion_regular>'*: añade la regla que aplica a los archivos y directorios que encajen en la expresión regular dada.
- *-d -t <contexto_tipo> '<expresión_regular>'*: borra la regla.
- *-m -t <contexto_tipo> '<expresión_regular>'*: modifica en la regla existente el contexto tipo dado.

Una vez añadida una regla a la política SELinux del sistema, para que entre en funcionamiento hay que aplicarla utilizando el comando *restorecon* (ver más abajo).

NOTA: El comando *semanage* está dentro del paquete *policycoreutils-python*, que en ocasiones no está instalado por defecto en el sistema. Nos puede parecer raro que con SELinux funcionando, al utilizar el comando *semanage*, el sistema nos diga *"command not found"*, simplemente instalar el paquete para poder utilizar el comando.

12.3.2.- Cambiar el contexto SELinux de un archivo o directorio

Existen dos comandos para modificar el contexto: *chcon* y *restorecon*. Ambos comando utilizan la opción *-t <contexto_tipo>* para modificar el tercer campo de la etiqueta SELinux.

- ***chcon**:* modifica el contexto de un archivo al proporcionado con el comando. NO se debe usar este comando ya que al especificar el contexto de forma explícita podemos equivocarnos y además, lo modificado volverá a tener el valor anterior cuando se reinicie el sistema. Usar sólo para pruebas. Sintaxis:

  chcon -t <contexto_tipo> <archivo> o chcon -tR <contexto_tipo> <directorio>

- ***restorecon**:* aplica las reglas definidas en la política SELinux del sistema sin necesidad de proporcionar el tipo de contexto. Este comando está dentro del paquete *policycoreutils*, que normalmente está ya instalado en el sistema.

Sintaxis:

restorecon -vvF <archivo> o restorecon -vvFR <directorio>

con la opción *-vv* vemos los cambios realizados por el comando, con la opción *-F* forzamos el reseteo del contexto y en el caso de directorios, con *-R*, lo hacemos recursivo, que aplique también a todo el contenido del directorio.

12.3.3.- Contextos SELinux en puertos del sistema

Los puertos por defecto de los procesos/servicios también están etiquetados de tal forma que sólo el proceso/servicio adecuado los puede usar. El problema es cuando queremos ejecutar un proceso/servicio en un puerto no estándar, para que SELinux lo permita, tenemos que etiquetar el nuevo puerto con la etiqueta de puerto correspondiente al proceso/servicio.

Para ver las etiquetas de los puertos, se usa:

semanage port -l.

Si queremos etiquetar un puerto no estándar, con la etiqueta de puerto de un proceso o servicio, deberemos mirar, primero cuál es la etiqueta que tiene el puerto estándar.

Por ejemplo, en el caso del proceso del apache *httpd*, vemos con *semanage port -l | grep http* que la etiqueta del puerto *80* es *http_port_t*.

Una vez que sabemos cuál es la etiqueta a poner en el nuevo puerto, ejecutamos:

semanage port -a -t <etiqueta_puerto> -p <protocolo> <número_puerto>

En el caso de querer cambiar que el proceso *httpd* escuche también por el puerto 90, ejecutaremos *semanage port -a -t http_port_t -p tcp 90*.

Para eliminar una etiqueta de un puerto, haremos:

semanage port -d -t <etiqueta_puerto> -p <tcp|udp> <número_puerto>

Y para modificar la etiqueta que tiene un puerto por otra:

semanage port -m -t <etiqueta_puerto> -p <tcp|udp> <número_puerto>

12.4.- Booleanos

Los booleanos de SELinux son parámetros que pueden ser activados o desactivados (valores *1* o *0*) y que modifican el comportamiento de la política SELinux.

Podemos listar todos los booleanos existentes en el sistema usando:

getsebool -a ó semanage boolean -l

Para ver el valor de un booleano en concreto usamos:

getsebool <booleano> ó semanage boolean -l | grep <booleano>

Para modificar su valor, usaremos:

```
setsebool -P <booleano> <0|1|on|off>
```

Es muy importante la opción *-P* para que la modificación sea persistente, de omitirla, perderíamos la modificación del booleano en el siguiente reinicio.

Una opción interesante es poder ver el estado de los booleanos cuya configuración difiere de la definida en la configuración por defecto en la política:

```
semanage boolean -l -C
```

12.5.- Ayuda SELinux del sistema

Las páginas del man del comando *semanage*, nos pueden ser muy útiles para buscar ejemplos del uso del comando: *man semanage-fcontext*, *man semanage-port*, *man semanage-boolean*.

Además, existen unas páginas del man que explican a fondo las etiquetas a utilizar en los objetos pero sobre todo, los booleanos de cada proceso/servicio. Están dentro del paquete *selinux-policy-doc* (en RHEL el paquete es *selinux-policy-devel*) que es muy recomendable tener instalado en el sistema. El nombre de estas páginas del man termina en "*_selinux*", con lo que con una búsqueda con *man search _selinux* nos saldrían todas las disponibles en el sistema.

Hay que recordar que la base de datos del man del sistema, se actualiza una vez al día con una tarea del Cron, con lo que si acabamos de instalar el paquete *selinux-policy-devel*, *man -k* no encontrará las páginas recien instaladas ya que su base de datos interna no ha sido actualizada. Simplemente ejecutando *mandb* (como usuario privilegiado), actualizamos dicha base de datos y a partir de ahora las búsquedas ya funcionarán.

12.6.- Monitorizar las violaciones SELinux

El paquete *setroubleshoot-server* que debe ser instalado para que se envíen los mensajes SELinux al */var/log/message*s. Se monitoriza lo que se va escribiendo en el archivo de logs de SELinux */var/log/audit/audit.log* y se envía una pequeña descripción al archivo de *messages* con un UUID. Con este UUID podemos ejecutar *sealert -l <uuid>* para ver un reporte sobre ese incidente en particular.

Obtendremos un reporte de todos los incidentes registrados en ese archivo con:

```
sealert -a /var/log/audit/audit.log
```

12.7.- Caso práctico

Se va a comprobar que en ambos sistemas **central** y **server1** está SELinux en modo Enforcing.

Se va a poner en marcha un servidor web de pruebas en **server1** en http://localhost. Para ello, modificaremos el *DocumentRoot* por defecto de Apache por **/prueba** y además estará escuchando en el puerto 90.

Además queremos que los usuarios del sistema en **server1** puedan tener en sus home, una página web personal en el directorio *web_publica*, p.e. el usuario *admin*, dentro de su home, tendrá un directorio *web_publica* con su web personal.

RESOLUCIÓN

- Instalamos el paquete *setroubleshoot-server* y *policycoreutils-python* en **server1** para poder usar los comandos *sealert* y *semanage*:

```
[root@server1 ~]#  yum install -y setroubleshoot-server policycoreutils-python
[root@server1 ~]# sealert -a /var/log/audit/audit.log
100% done
found 0 alerts in /var/log/audit/audit.log
```

- Instalamos las páginas del man extras de SELinux que están en el paquete *selinux-policy-devel y selinux-policy-doc* :

```
[root@server1 ~]#  yum install -y selinux-policy-devel selinux-policy-doc
[root@server1 ~]# man -k _selinux
pam_selinux (8)      - PAM module to set the default security context
```

Vemos que como no hemos actualizado la base de datos interna del man, sólo hay una página disponible. Actualizamos la base de datos interna y nos encontraremos con 825 páginas:

```
[root@server1 ~]# mandb
[root@server1 ~]# man -k _selinux | wc -l
825
```

- Instalamos y configuramos Apache:

```
[root@server1 ~]#  yum install -y httpd
[root@server1 ~]#  mkdir /prueba
[root@server1 ~]# echo "Prueba" > /prueba/index.html
[root@server1 ~]#  sed -i 's/\/var\/www\/html/\/prueba/g' /etc/httpd/conf/httpd.conf
[root@server1 ~]#  systemctl start httpd
```

- Accedemos a la web:

```
[root@server1 ~]#  curl http://localhost/index.html
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access /index.html
on this server.</p>
</body></html>
```

Vemos si poniendo SELinux en permisive, dejamos de tener el error:

```
[root@server1 ~]#  setenforce 0
[root@server1 ~]#  curl http://localhost/index.html
        Prueba
```

Luego el problema está con SELinux, ejecutamos *sealert* para obtener más datos del problema, pero antes volvemos a poner SELinux en modo *Enforcing*:

```
[root@server1 ~]# setenforce 1
[root@server1 ~]# sealert -a /var/log/audit/audit.log
100% done
found 1 alerts in /var/log/audit/audit.log
--------------------------------------------------------------------------------
SELinux is preventing /usr/sbin/httpd from getattr access on the file /prueba/index.html.
*****  Plugin catchall_labels (83.8 confidence) suggests   ******************
If you want to allow httpd to have getattr access on the index.html file
Then you need to change the label on /prueba/index.html
Do
# semanage fcontext -a -t FILE_TYPE '/prueba/index.html'
where FILE_TYPE is one of the following: NetworkManager_exec_t, … ← listado etiquetas
...
Then execute:
restorecon -v '/prueba/index.html'
```

Esto nos está diciendo que el archivo */prueba/index.html* al cuál estamos intentando acceder mediante *httpd*, no tiene el contexto adecuado, buscaríamos en las páginas del man:

```
[root@server1 ~]# man -k _selinux | grep http
apache_selinux (8)   - Security Enhanced Linux Policy for the httpd processes
httpd_helper_selinux (8) - Security Enhanced Linux Policy for the httpd_helper processes
httpd_passwd_selinux (8) - Security Enhanced Linux Policy for the httpd_passwd processes
httpd_php_selinux (8) - Security Enhanced Linux Policy for the httpd_php processes
httpd_rotatelogs_selinux (8) - Security Enhanced Linux Policy for the httpd_rotatelogs processes
httpd_selinux (8)    - Security Enhanced Linux Policy for the httpd processes
httpd_suexec_selinux (8) - Security Enhanced Linux Policy for the httpd_suexec processes
httpd_sys_script_selinux (8) - Security Enhanced Linux Policy for the httpd_sys_script processes
httpd_unconfined_script_selinux (8) - Security Enhanced Linux Policy for the
httpd_unconfined_script pro...
httpd_user_script_selinux (8) - Security Enhanced Linux Policy for the httpd_user_script processes
```

consultando *httpd_selinux*:

```
[root@server1 ~]# man httpd_selinux
```

vemos que el contexto a poner es *httpd_sys_content_t* y vemos cual tiene ahora */prueba*:

```
[root@server1 ~]# ls -ldZ /prueba
-rw-r--r--. root root unconfined_u:object_r:default_t:s0 index.html
[root@server1 ~]# semanage fcontext -a -t httpd_sys_content_t '/prueba(/.*)?'
```

[root@server1 ~]# restorecon -vvFR /prueba

restorecon reset /prueba context unconfined_u:object_r:default_t:s0
->system_u:object_r:httpd_sys_content_t:s0

restorecon reset /prueba/index.html context unconfined_u:object_r:default_t:s0
->system_u:object_r:httpd_sys_content_t:s0

[root@server1 ~]# ls -ldZ /prueba

-rw-r--r--. root root unconfined_u:object_r:**httpd_sys_content_t**:s0 index.html

Intentamos de nuevo acceder a la página web:

[root@server1 ~]# curl http://localhost/index.html

Prueba

Ahora si podemos ver el contenido de la página web.

- Cambiamos el puerto por defecto de Apache del 80 al 90 y reiniciamos el servicio *httpd*:

[root@server1 ~]# sed -i 's/Listen 80/Listen 90/g' /etc/httpd/conf/httpd.conf

[root@server1 ~]# systemctl restart httpd

Job for httpd.service failed because the control process exited with error code. See "systemctl status httpd.service" and "journalctl -xe" for details.

[root@server1 ~]# journalctl -xe -l

Nos dice que hay problemas con el puerto 90 (*could not bind to address 0.0.0.0:90*), también podemos volver a ejecutar *sealert* y nos dice:

SELinux is preventing /usr/sbin/httpd from name_bind access on the tcp_socket port 90.

Then you need to modify the port type.

Do

semanage port -a -t PORT_TYPE -p tcp 90

where PORT_TYPE is one of the following: http_cache_port_t, **http_port_t**, jboss_management_port_t, jboss_messaging_port_t, ntop_port_t, puppet_port_t.

- Ponemos el contexto *http_port_t* al puerto 90, tal y como nos indica la salida anterior y reiniciamos el servicio:

[root@server1 ~]# semanage port -a -t http_port_t -p tcp 90

[root@server1 ~]# systemctl restart httpd

[root@server1 ~]# curl http://localhost:90/index.html

Prueba

Funciona!

- Modificamos la configuración de Apache para permitir que los usuarios puedan tener sus páginas personales en el directorio *web_publica*, para ello modificamos el archivo de configuración */etc/httpd/conf.d/userdir.conf*, poniendo la directiva *UserDir* con valor *web_publica* y cambiado *public_html* por *web_publica*:

```
[root@server1 ~]# sed -i 's/UserDir disabled/Userdir web_publica/g'
                                                    /etc/httpd/conf.d/userdir.conf
[root@server1 ~]# sed -i 's/public_html/web_publica/g' /etc/httpd/conf.d/userdir.conf
```

- El usuario *admin* crea su web personal que se servirá en http://localhost:90/~admin/index.html para esto, debe permitir que Apache pueda acceder a su subdirectorio *web_publica*.

```
[root@server1 ~]# su – admin
[admin@server1 ~]# mkdir ~/web_publica
[admin@server1 ~]# echo 'Pagina personal de admin' >   ~/web_publica/index.html
[admin@server1 ~]# chmod 711 ~
[admin@server1 ~]# exit
[root@server1 ~]# curl http://localhost:90/~admin/index.html
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access /~admin/index.html
on this server.</p>
```

Ejecutando de nuevo *sealert*, nos da la pista de lo que debemos modificar, el *booleano httpd_enable_homedirs*:

```
[root@server1 ~]# sealert -a /var/log/audit/audit.log
SELinux is preventing /usr/sbin/httpd from getattr access on the file
/home/admin/web_publica/index.html.
*****  Plugin catchall_boolean (24.7 confidence) suggests   ******************
If you want to allow httpd to read home directories
Then you must tell SELinux about this by enabling the 'httpd_enable_homedirs' boolean.
You can read 'httpd_selinux' man page for more details.
Do
setsebool -P httpd_enable_homedirs 1
[root@server1 ~]# getsebool -a | grep http | grep home
httpd_enable_homedirs --> off
[root@server1 ~]# setsebool -P httpd_enable_homedirs on
[root@server1 ~]# curl http://localhost:90/~admin/index.html
Pagina personal de admin
```

Funciona!