

SELinux

Seguridad Informática

Pedro Allué Tamargo (758267)

Juan José Tambo Tambo (755742)

3 de noviembre de 2020

Índice

1. Parte 1: Información, estado y dominios en <i>SELinux</i>	1
2. Parte 2: Usuarios en <i>SELinux</i>	2
3. Parte 3: Listas de Control de Accesos (<i>ACLs</i>) basados en estándar <i>POSIX</i> de <i>Unix</i> del modelo <i>DAC</i>	3
4. Parte 4: Reglas de control de accesos y acceso a dominios	4

1. Parte 1: Información, estado y dominios en *SELinux*

Tras iniciar sesión con el usuario *u* se ejecuta el comando `ps aux` y se puede observar en la salida del comando que los usuarios efectivos que están ejecutando procesos en el sistema son: *root*, *dbus*, *rpc*, *rpcuser*, *68*, *postfix*, *gdm*, *rtkit*, *u*.

Si ejecutamos el comando `ps auxZ` se puede observar que una salida similar a la del comando anterior pero con la diferencia de que aparecen los usuarios, roles y dominios *SELinux* de los procesos en ejecución en el sistema.

Los usuarios *Linux* que están ejecutando procesos en dominios `unconfined_t` son: *u*.

Los usuarios *Linux* que están ejecutando procesos con el usuario y rol `system_u :system_r` son: *root*, *rpc*, *rpcuser*, *68*, *postfix*, *gdm*, *rtkit*.

La diferencia entre que unos procesos están en dominios `unconfined_t` y otros no radica en que los procesos con dominio `unconfined_t` se corresponde con los ejecutados por un usuario *logged-in* (usuario *u*), mientras que el resto se corresponden con procesos ejecutados por usuarios que no han iniciado sesión en el sistema.

Si se invoca el comando `passwd` en otra terminal se puede observar que el usuario *Linux* del proceso se corresponde con el usuario *root* ya que el binario `passwd` tiene el bit `setuid` activo. También se puede observar que el usuario y el rol del proceso sigue siendo `unconfined_u:unconfined_r` pero el dominio del proceso ha cambiado `passwd_t`. Esto se debe a que se ha producido una transición de dominio en el cual se ha sustituido `unconfined_t` por `passwd_t`.

La interacción con *SELinux* no es solo mediante línea de comandos, también se puede interactuar utilizando la herramienta gráfica *SELinux Manage* (Figura 1) disponible en el sistema. Esta herramienta utiliza la interfaz de línea de comandos y muestra la información de una forma más “amigable” para el usuario.

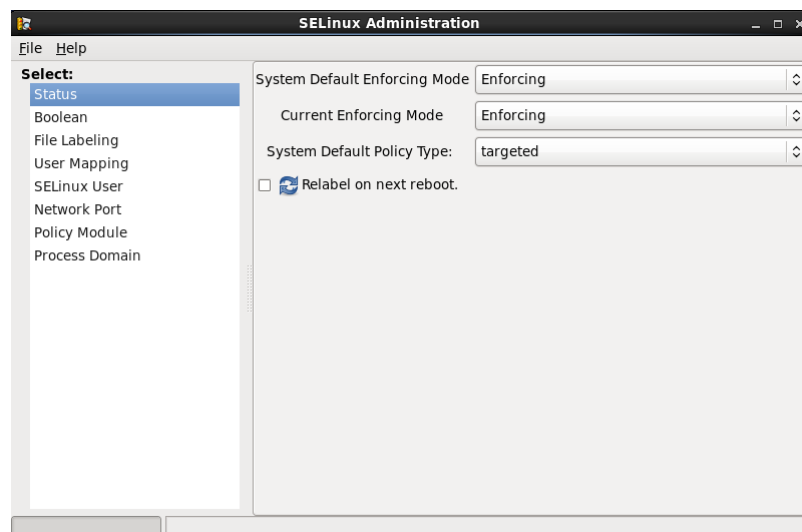


Figura 1: Captura de pantalla de la aplicación gráfica

Por ejemplo en el apartado “*User Mapping*” (Figura 2) se puede observar la relación entre los usuarios de *Linux* y los usuarios de *SELinux*. Esto también se puede observar utilizando el comando `semanage login -l` (Figura 3).

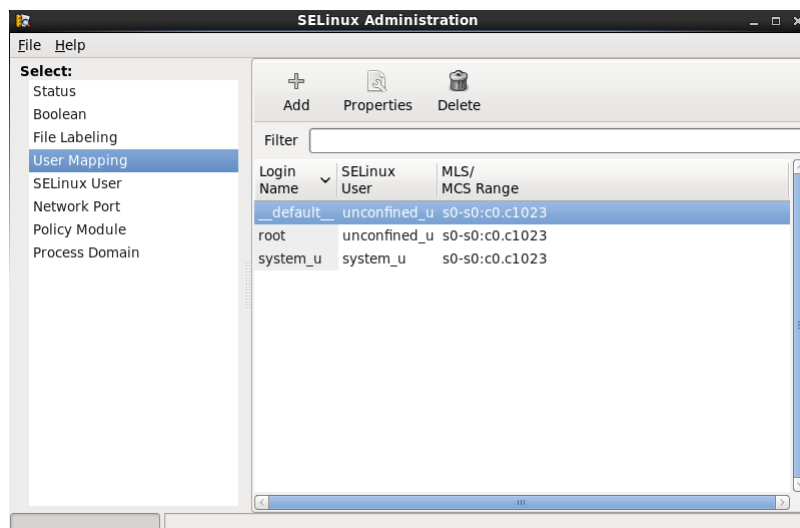


Figura 2: Captura de pantalla del apartado “User Mapping”

```
[root@localhost u]# semanage login -l
```

Login Name	SELinux User	MLS/MCS Range
default	unconfined_u	s0-s0:c0.c1023
root	unconfined_u	s0-s0:c0.c1023
system_u	system_u	s0-s0:c0.c1023

Figura 3: Captura de pantalla de la salida del comando `semanage login -l`

2. Parte 2: Usuarios en *SELinux*

Se va a proceder a crear un nuevo usuario *c*. Para ello se utilizará el comando `adduser -Z user_u c`. Este comando creará el usuario *Linux* *c* y el usuario *SELinux* *user_u*. Tras la creación del usuario se cambiará al usuario *c* (`su - c`) y se ejecutará el comando `id`. La salida de este comando (Figura 4) muestra en el apartado del contexto la cadena: `unconfined_u:unconfined_r:unconfined_t`. Esto muestra que el usuario *c* se corresponde con el usuario de *SELinux* *unconfined_u*. Esto no es completamente cierto. En la creación del usuario de *Linux* *c* se le ha asignado al usuario de *SELinux* *user_u*, pero al haber realizado el login con el usuario *u* se han mantenido su usuario, rol y tipo de *SELinux*.

```
[c@localhost ~]$ id
uid=501(c) gid=502(c) groups=502(c) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figura 4: Captura de pantalla de la salida del comando `id` con el usuario *c*

Si se ejecuta la orden `semanage login -l` (Figura 5) se puede observar que el usuario *u* no existe y aparece una entrada en la tabla para el usuario *c* que asocia ese usuario de *Linux* con el usuario *SELinux* *user_u*.

Login Name	SELinux User	MLS/MCS Range
default	unconfined_u	s0-s0:c0.c1023
c	user_u	s0
root	unconfined_u	s0-s0:c0.c1023
system_u	system_u	s0-s0:c0.c1023

Figura 5: Captura de pantalla de la salida del comando `semanage login -l`

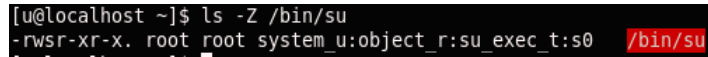
Si iniciamos sesión en la interfaz gráfica con el usuario *c* y volvemos a ejecutar el comando `id` en una terminal se obtendrá la salida mostrada en la Figura 6. Se puede observar que ha cambiado el contexto (usuario, rol y tipo de *SELinux*) y ahora muestra la siguiente información `user_u:user_r:user_t`.

```
[c@localhost ~]$ id
uid=501(c) gid=502(c) groups=502(c) context=user_u:user_r:user_t:s0
```

Figura 6: Captura de pantalla de la salida del comando `id` con el usuario *u*

Ahora utilizando al usuario *c* se va a intentar ejecutar el comando `su -` para acceder a la cuenta *root*. El resultado es que no se puede utilizar este binario (`su`) ya que en *SELinux* hay que proveer permisos explícitos para permitir ciertas acciones, en este caso ejecutar un fichero que no pertenece al dominio `bin_t`. Este binario (Figura 7) pertenece al dominio `su_exec_t`. Para poder utilizar este binario habrá que crear una regla de acceso que relacione el tipo `user_t` con el dominio `su_exec_t`. La regla de acceso tendría la forma:

```
allow user_t su_exec_t:file {read execute getaddr};
```



```
[u@localhost ~]$ ls -Z /bin/su
-rwsr-xr-x. root root system_u:object_r:su_exec_t:s0 /bin/su
```

Figura 7: Captura de pantalla de la salida del comando `ls -Z /bin/su`

3. Parte 3: Listas de Control de Accesos (*ACLs*) basados en estándar *POSIX* de *Unix* del modelo *DAC*

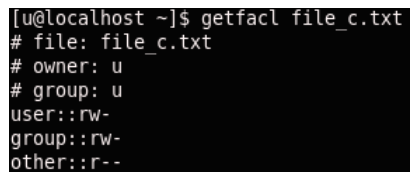
En este apartado se procede a crear un fichero “*home*” con el usuario *u* y añadir mediante el comando `setfacl` permisos de lectura y ejecución al usuario *c* y de lectura al grupo *c*.

Para ello, se debe ejecutar el siguiente comando:

```
setfacl -m u:c:rx -m g:c:r <fichero>
```

Con la opción “`-m`” se indican que se van a modificar los permisos del fichero indicado. Estos permisos se deben indicar de la siguiente forma: `[usuario(u)|grupo(g)]:[user|grupo destino]:[permisos]`.

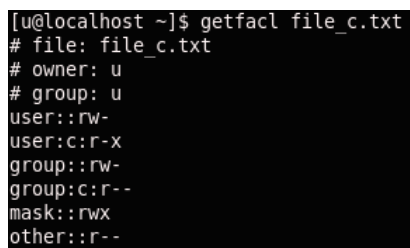
Con el comando `getfacl` se muestran los permisos del fichero indicado. Si se ejecuta el comando antes de modificar los permisos, se muestra lo siguiente:



```
[u@localhost ~]$ getfacl file_c.txt
# file: file_c.txt
# owner: u
# group: u
user::rw-
group::rw-
other::r--
```

Figura 8: Captura de pantalla de la salida del comando `getfacl file_c.txt` antes de modificar los permisos.

Como se puede observar, solo tiene permisos de lectura y escritura el usuario *u* y el grupo *u*. Sin embargo, tras modificar los permisos con `setfacl`, el comando `getfacl` muestra lo siguiente:



```
[u@localhost ~]$ getfacl file_c.txt
# file: file_c.txt
# owner: u
# group: u
user::rw-
user:c:r-x
group::rw-
group:c:r--
mask::rwx
other::r--
```

Figura 9: Captura de pantalla de la salida del comando `getfacl file_c.txt` tras modificar permisos.

Se indica que, además del usuario *u*, ahora tiene permisos de lectura y ejecución el usuario *c* y de lectura el grupo *c*.

Lo contenido en *mask* indica la unión de los permisos de todos los grupos y usuarios que tienen acceso.

4. Parte 4: Reglas de control de accesos y acceso a dominios

Para poder obtener los dominios a los que se les permite acceder a ficheros de tipo *shadow_t* y clase *file* con permisos de escritura, se debe ejecutar el siguiente comando:

```
sesearch -A -t shadow_t -c file -p write
```

La ejecución del comando anterior muestra lo siguiente:

```
[root@localhost ~]# sesearch -A -t shadow_t -c file -p write
Found 9 semantic av rules:
allow groupadd t shadow_t : file { ioctl read write create getattr setattr lock relabelfrom relabelto append unlink link rename open } ;
allow passwd t shadow_t : file { ioctl read write create getattr setattr lock relabelfrom relabelto append unlink link rename open } ;
allow useradd t shadow_t : file { ioctl read write create getattr setattr lock relabelfrom relabelto append unlink link rename open } ;
allow sysadm passwd t shadow_t : file { ioctl read write create getattr setattr lock relabelfrom relabelto append unlink link rename open } ;
allow sandbox domain shadow_t : file { ioctl read write getattr lock append } ;
allow updpwd t shadow_t : file { ioctl read write create getattr setattr lock append unlink link rename open } ;
allow files_unconfined_type file_type : file { ioctl read write create getattr setattr lock relabelfrom relabelto append unlink link rename execute swapon quotaon mounton execute no trans entrypoint open } ;
allow yppasswdd t shadow_t : file { ioctl read write create getattr setattr lock relabelfrom relabelto append unlink link rename open } ;
allow mount_t file_type : file { ioctl read write getattr lock append } ;
```

Figura 10: Captura de pantalla de la salida del comando `sesearch -A -t shadow_t -c file -p write`.

Esto nos indica que todos los procesos que pertenecen a los dominios que devuelve el comando (aparecen seguidos de *allow*) tienen permiso de escritura sobre archivos que pertenecen a la clase *file* del dominio o tipo *shadow_t*. La clase *file* indica que es un fichero regular.

Para obtener todos los permisos de rol al que pertenece el usuario *c*, es decir, *user_r*, se debe ejecutar el siguiente comando:

```
seinfo -ruser_r -x
```

Su ejecución devuelve lo siguiente (solo se muestra una parte de la devolución):

```
[root@localhost ~]# seinfo -ruser_r -x
user_r
Dominated Roles:
  user_r
Types:
  git session t
  sandbox_x_client_t
  virt content t
  policykit_grant_t
  httpd user htaccess t
  telepathy_mission_control_home_t
  gnome home t
  sandbox_net_client_t
  loadkeys t
  httpd user_script_exec t
  policykit_auth_t
  ssh keygen t
```

Figura 11: Captura de pantalla de la salida del comando `seinfo -ruser_r -x`

Con ello se observa que todo usuario cuyo rol sea *user_r* tiene acceso a todos esos tipos y dominios. Para que pueda tener acceso debe haberse establecido una regla de política de *SELinux* que lo permita.