

# SELinux: Haciendo Nuestra Seguridad Impenetrable

Fundacion Codigo Libre Dominicano  
Junio 2008.-

**Departamento:** Investigacion Cientifica  
**División:** Academica

**Documentación realizada por:**

Jesus Rafael Sanchez Medrano  
**Director de Investigacion Cientifica**

# Índice de contenido

Que es SELinux?.....	1
Entendiendo como funciona SELinux:.....	2
Conceptos basicos de seguridad.....	4
Atributos de seguridad.....	4
Control de escalamiento de privilegios:.....	5
El sistema de archivos virtual /selinux/.....	6
El archivo de configuración /etc/sysconfig/selinux.....	7
Utilidades para SELinux.....	9
Creando nuevas politicas de SELinux, Paso a Paso.....	10
¿Como puedo conocer el estado actual de SELinux?.....	11
¿Que aplicaciones estan siendo denegadas por SELinux en mi sistema?.....	11
¿Como puedo permitir una aplicacion ya denegada por SELinux en el sistema?.....	11
¿Como permitir el acceso a las aplicaciones que han sido negadas por el sistema?...14	
¿Como puedo permitir que un servicio pueda usar cualquier puerto?.....	14
¿Cómo puedo cambiar la localización por defecto de algunos servicios?.....	15

## Que es SELinux?

Security-Enhanced Linux o SELinux, es una arquitectura de seguridad integrada en el kernel 2.6.x usando los módulos de seguridad linux (o *linux security modules*, **LSM**). Este es un proyecto de la Agencia de Seguridad Nacional (**NSA**) de los Estados Unidos y de la comunidad SELinux. La integración de SELinux en Red Hat Enterprise Linux fue un esfuerzo conjunto entre al NSA y Red Hat.

SELinux proporciona un sistema flexible de control de acceso obligatorio (*mandatory access control*, **MAC**) incorporado en el kernel. Bajo el Linux estándar se utiliza el control de acceso a discreción (*discretionary access control*, **DAC**), en el que un proceso o aplicación ejecutándose como un usuario (UID o SUID) tiene los permisos y de ese usuario en los objetos, archivos, sockets y otros procesos. Al ejecutar un kernel SELinux MAC se protege al sistema de aplicaciones maliciosas o dañadas que pueden perjudicar o destruir el sistema. SELinux define el acceso y los derechos de transición de cada usuario, aplicación, proceso y archivo en el sistema. SELinux gobierna la interacción de estos sujetos y objetos usando una política de seguridad que especifica cuán estricta o indulgente una instalación de GNU/Linux dada debería de ser.

En su mayor parte, SELinux es casi invisible para la mayoría de los usuarios. Solamente los administradores de sistemas se deben de preocupar sobre lo estricto que debe ser una política a implementar en sus entorno de servidores. La política puede ser tan estricta o tan indulgente como se requiera, y es bastante detallada. Este detalle le dá al kernel SELinux un control total y granular sobre el sistema completo.

Cuando un sujeto, tal como una aplicación, intenta acceder a un objeto tal como a un archivo, el servidor de aplicación de políticas verifica un *caché de vector de acceso (AVC)*, donde se registran los permisos de objeto y del sujeto. Si no se puede tomar una decisión basado en los datos en el AVC, la petición continua al servidor de seguridad, el cual busca el contexto de seguridad de la aplicación y del archivo en una matriz. Los permisos son entonces otorgados o negados, con un mensaje de `avc: denied` detallado en `/var/log/messages`. Los sujetos y objetos reciben su contexto de seguridad a partir de la política instalada, que también proporciona información para llenar la matriz de seguridad del servidor.

Además de ejecutarse en un modo impositivo u obligatorio, SELinux puede ejecutarse en un modo permisivo, donde el AVC es verificado y se registran los rechazos, pero SELinux no hace cumplir esta política.

## Entendiendo como funciona SELinux:

SELinux funciona como un modulo en el kernel, logrando un mayor nivel de abstraccion para los usuarios.

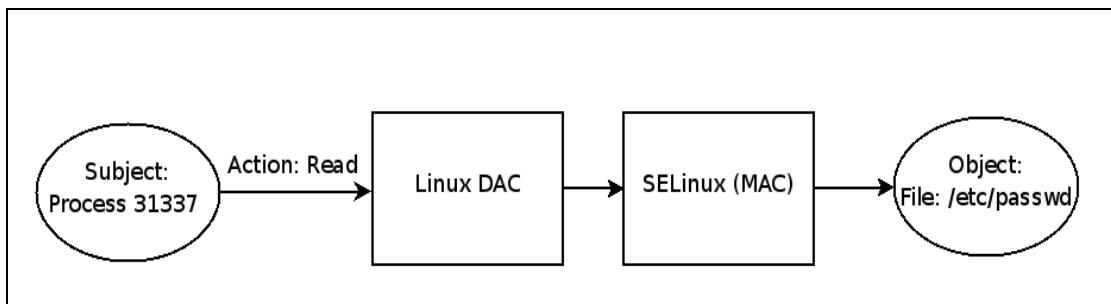
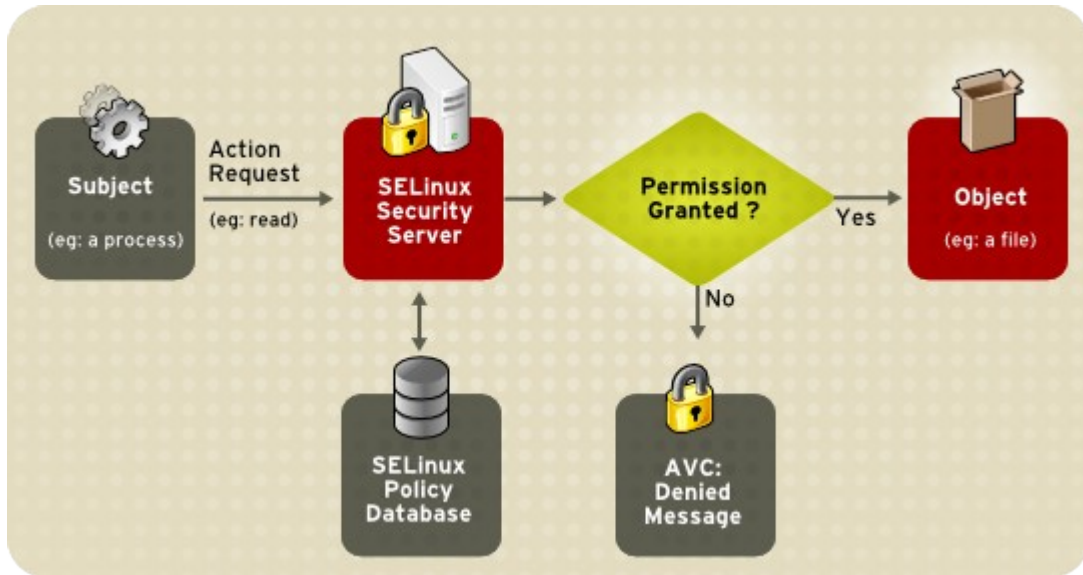
SELinux trabaja con lo que son contextos de seguridad, controles de acceso impositivos u obligatorios y control de acceso en base a roles; ofreciendo un control mas granular del acceso a los recursos del sistema por parte de los objetos (programas y aplicaciones) y los sujetos (roles, usuarios y grupos).

Este no reemplaza el modelo tradicional de seguridad de los sistemas tipo Unix, por el contrario, sirve de complemento de este en los puntos que la seguridad tradicional no es suficiente, en el cual la seguridad esta dividida por niveles de usuarios, grupos, derechos de accesos, listas de control de acceso y atributos extendidos de acceso; en donde un usuario puede ejecutar un conjunto de aplicaciones a las que tiene derecho y estas son ejecutadas con los niveles de acceso que posee el usuario. Este tipo de seguridad es llamada **DAC** (*Discretionary Access Control*), control de acceso discrecional.

La NSA introdujo un sistema de control tipo **MAC** (*Mandatory Access Control*), basado en contextos donde se indica cuando un objeto o sujeto puede acceder a otro objeto.

En este caso, el administrador debe definir los derechos de cada usuario que acceda a algunas aplicaciones que que accedan a cualquier objeto del sistema. Para evitar que esta operacion sea tediosa, se definen controles de acceso por roles (*Role-Based Access Control*, **RBAC**).

Por ejemplo, el usuario A le otorgamos acceso de lectura y escritura para escribir en los archivos del programa xx solamente a través de este, ahora el usuario B solamente necesita acceso de lectura y nada más sobre los mismos archivos a través del mismo programa.



Los métodos de seguridad de SELinux no reemplazan los tradicionales controles de acceso discrecionales, por el contrario, los complementa para brindar un control completo y granular de la seguridad. Todo inicialmente es consultado con la capa de control de acceso discrecional (DAC), si esta niega, no hay necesidad de consultar la capa de control de acceso obligatorio (MAC); en caso contrario que el DAC permite el acceso, entonces se hace la consulta hacia la capa MAC para terminar de verificar los controles de acceso.

## Conceptos basicos de seguridad

### *Atributos de seguridad*

SELinux usa una combinacion de un modelo de identidad, control de acceso en base a roles (RBAC), enforzamiento de tipos (TE) y categorias de acceso a la informacion. SELinux RBAC autoriza a cada usuario (de SELinux) para un conjunto de roles. Cada rol es autorizado para un conjunto de tipos.

Esto se logra utilizando cuatro atributos de seguridad:

1. **Identidad de usuario:** SELinux tiene su propia base de datos de usuarios que esta asociada a la base de datos normal de usuarios de Linux. Las identidades son usadas en ambos, sujetos y objetos. Solamente unos cuantos usuarios de SELinux son definidos: (pueden ser listados por el comando `semanage user -l`):

- **user\_u** – Usuarios normales.
- **system\_u** – Procesos iniciados (al arranque).
- **root** - administrador.

2. **Role:** Los usuarios pueden entrar en diferentes roles. Diferentes roles pueden entrar en diferentes dominios. Para objetos (archivos), esto es siempre **object\_r**.

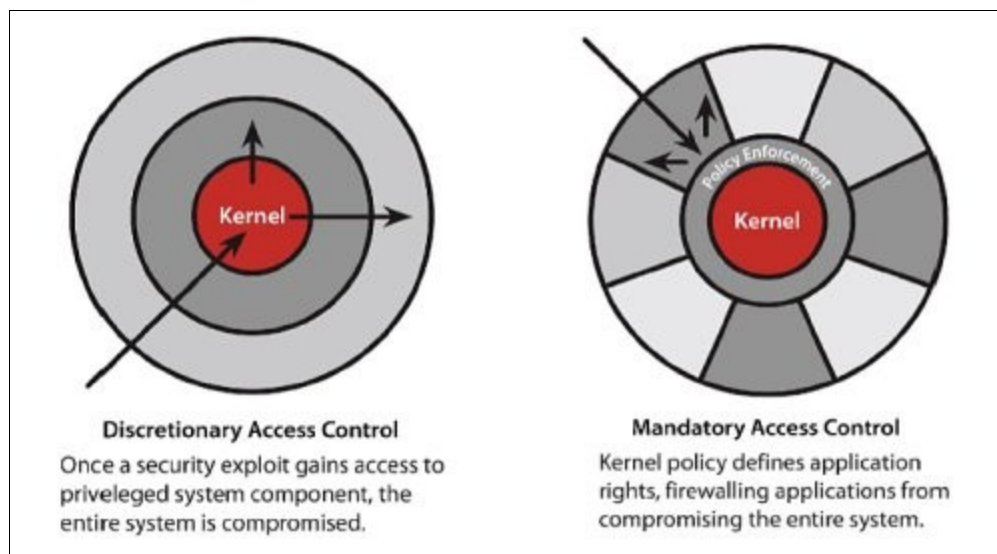
3. **Tipo / dominio:** Es el atributo “principal” en SELinux. Tambien llamado el “atributo primario”. Es, por lo general, solo unos pocos usuarios / roles, pero centenares de tipos. No existe diferencia entre “tipos” y “dominios”, pero los "dominios" son usados para cuando hablamos de procesos y los “tipos” para cuando hablamos de archivos. Cada proceso esta confinado en su propia caja de arena, con acceso restringido, tambien llamado “Enforzamiento de Tipos”.

4. **Categoria / Nivel:** Podra establecer la categoria / nivel. Introducido en RHEL 5, Habilita los controles de **seguridad Multi-nivel** (MLS) o **seguridad Multi-categoria** (MCS).

Estos cuatro atributos de seguridad, construyen lo que se denomina un “**contexto de seguridad**”:

**<usuario>: <rol>: <tipo>: <categoria/nivel>**

Atributo de seguridad	Convencion de nombre	Ejemplo
<b>usuario</b>	<b>_u</b>	<b>user_u</b>
<b>rol</b>	<b>_r</b>	<b>object_r</b>
<b>tipo</b>	<b>_t</b>	<b>unconfined_t</b>
<b>categoria/nivel</b>	<b>(ninguna)</b>	<b>s0:c0</b>



Cuando tenemos todos los servicios de red y sistema confinados, tenemos un mayor control de seguridad, porque no comprometemos al sistema, aislando cada servicio como una entidad independiente.

### ***Control de escalamiento de privilegios:***

La seguridad esta segmentada en roles principales

- **user\_r:** El mas minimo privilegio, asignado simplemente a los usuarios; alguien con este rol, no puede hacer escalamiento de privilegios ni alternar de rol.
- **staff\_r:** Similar al rol **user\_r**, con la diferencia que con este rol se puede alternar a los diferentes roles asignados con el perfil del usuario.
- **auditadm\_r:** Rol encargado de las herramientas de auditoria de manera unica y exclusiva, los usuarios de este rol solamente desempeñan esta funcion.
- **secadm\_r:** Rol encargado de manejar las herramientas de seguridad, declarar nuevos roles, crear nuevas politicas y de ver los log de auditoria, mas no puede modificarlos.
- **sysadm\_r:** Rol maximo en el sistema, pero no puede usar las herramientas de auditoria ni modificar estos reportes.

### *El sistema de archivos virtual /selinux/*

El pseudo-sistema de archivos `/selinux/` contiene los comandos que son utilizados más a menudo por el subsistema del kernel. Este tipo de sistema de archivos es similar al pseudo sistema `/proc/`.

En la mayoría de los casos, los administradores y usuarios no necesitan manipular este componente, en comparación con otros archivos y directorios SELinux.

El ejemplo siguiente presenta contenidos de muestra del directorio `/selinux/`:

```
-rw-rw-rw- 1 root root 0 Sep 22 13:14 access
dr-xr-xr-x 1 root root 0 Sep 22 13:14 booleans
--w----- 1 root root 0 Sep 22 13:14 commit_pending_bools
-rw-rw-rw- 1 root root 0 Sep 22 13:14 context
-rw-rw-rw- 1 root root 0 Sep 22 13:14 create
--w----- 1 root root 0 Sep 22 13:14 disable
-rw-r--r-- 1 root root 0 Sep 22 13:14 enforce
-rw----- 1 root root 0 Sep 22 13:14 load
-r--r--r-- 1 root root 0 Sep 22 13:14 mls
-r--r--r-- 1 root root 0 Sep 22 13:14 policyvers
-rw-rw-rw- 1 root root 0 Sep 22 13:14 relabel
-rw-rw-rw- 1 root root 0 Sep 22 13:14 user
```

Por ejemplo, al ejecutar el comando `cat` en el archivo `/selinux/enforce` revela un 1, para el modo impositivo u obligatorio, o un 0, para el modo permisivo.

## ***El archivo de configuración /etc/sysconfig/selinux***

Hay dos formas de configurar SELinux bajo Red Hat Enterprise Linux y CentOS: usando el **Herramienta de configuración de nivel de seguridad** (`system-config-securitylevel`), o manualmente editando el archivo de configuración (`/etc/sysconfig/selinux`).

El archivo `/etc/sysconfig/selinux` es el archivo de configuración principal para habilitar o inhabilitar SELinux, así como también para configurar cuál política de debe imponer en el sistema y cómo hacerlo.

*\*\*NOTA: El archivo /etc/sysconfig/selinux contiene un enlace simbólico al archivo de configuración real, /etc/selinux/config.*

A continuación se explica el subconjunto completo de opciones disponibles para la configuración:

- `SELINUX=<enforcing/permmissive/disabled>` — Define el estado superior para SELinux en un sistema.
  - `enforcing` o 'impositivo' — Se impone la política de seguridad SELinux.
  - `permmissive` o 'permisivo' — El sistema SELinux advierte pero no impone la política. Esto es útil para propósitos de depuración o de resolución de problemas. En modo permisivo, se registrarán más rechazos, pues los sujetos podrán continuar con acciones que de lo contrario serían rechazadas en el modo impositivo. Por ejemplo, navegar en un árbol de directorios producirá varios mensajes de `avc: denied` para cada nivel de directorio leído, pero un kernel en modo impositivo habría detenido la primera acción de este tipo, previniendo que se produjeran más mensajes de rechazo.
  - `disabled` o 'inhabilitado' — SELinux está completamente desactivado. Los ganchos de SELinux no están conectados al kernel y el pseudo sistema de archivos no está registrado.

- `SELINUXTYPE=<targeted/strict>`: Especifica cuál política está siendo implantada actualmente por SELinux.

- `targeted` (objetivo, dirigido): Solamente se protegen ciertos demonios particulares.

La imposición de políticas para estos demonios se puede activar y desactivar, utilizando valores booleanos controlados por **Herramienta de configuración de nivel de seguridad** (`system-config-securitylevel`). Al activar un valor booleano para un demonio objetivo se desactiva la transición de políticas para ese demonio, lo que previene, por ejemplo, que `init` pase a `dhcpcd` desde el dominio `unconfined_t` (ilimitado) al dominio especificado en `dhcpcd.te`. El dominio `unconfined_t` permite a los sujetos y objetos con ese contexto de seguridad a ejecutarse bajo la seguridad estándar de Linux.

- `MLS` (Seguridad Multi-nivel): Se definen capas de seguridad para cada objeto, en donde solo los usuarios que posean los niveles correspondientes o superiores pueden acceder a las secciones correspondientes; nunca en sentido inverso.
- `MCS` (Seguridad Multi-categoría): Muy similar a la seguridad Multi-nivel, pero con patrones mas flexibles y faciles de aplicar; debido a que uno segmenta la informacion como si fuese para ser manejada por departamentos.
- `strict` (estricta): Protección SELinux completa, para todos los demonios. Se definen los contextos de seguridad para todos los sujetos y objetos y cada simple acción es procesada por el servidor de aplicación de políticas.

## Utilidades para SELinux

Los siguientes son algunos de los programas de utilidades usados más a menudo por SELinux

- `/usr/bin/setenforce`: Modifica en tiempo real el modo en que se ejecuta SELinux. Al ejecutar `setenforce 1`, se coloca SELinux en modo impositivo. Al ejecutar `setenforce 0`, SELinux se coloca en modo permisivo. Para desactivar SELinux, se necesita que configure el parámetro en `/etc/sysconfig/selinux` o que pase el parámetro `selinux=0` al kernel, bien sea en `/etc/grub.conf` o al momento del arranque.

- `/usr/bin/sectatus -v`: obtiene el estado detallado de un sistema ejecutando SELinux. El ejemplo siguiente muestra un extracto de la salida `sectatus`:

```
SELinux status:      enabled
SELinuxfs mount:    /selinux
Current mode:       enforcing
Policy version:     18
```

- `/usr/bin/newrole`: Ejecuta un nuevo shell en un nuevo contexto o papel. La política debe permitir la transición al nuevo papel.

- `/sbin/restorecon`: Configura el contexto de seguridad de uno o más archivos, marcando los atributos extendidos con el archivo apropiado o contexto de seguridad.

- `/sbin/fixfiles`: Verifica o corrige la base de datos del contexto de seguridad en el sistema de archivos.

## Creando nuevas politicas de SELinux, Paso a Paso

Para crear un modulo de politicas de SELinux, se deben tomar en cuenta los siguientes puntos:

- Para saber si las politicas de seguridad de selinux estan enforzadas se ejecuta el comando `getenforce`.
- para obtener las metricas de efectividad de las politicas de seguridad implementadas se ejecuta `seinfo`.
- Para desactivar el enforzamiento de SELinux en el momento se ejecuta `setenforce 0` y para reactivarlo de nuevo se ejecuta la sentencia `setenforce 1`.
- Si se desea desactivar selinux de manera definitiva, se modifica el archivo `/etc/sysconfig/selinux` y se cambia la directiva `SELINUX=enforcing` por `SELINUX=disabled`.
- Para saber que aplicaciones se estan denegando por selinux en el sistema, se utiliza la herramienta grafica incluida llamada `setroubleshoot`, en caso de no tener soporte de Xwindow se puede ejecutar en modo consola:

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

- para permitir una aplicacion que habia sido denegada por selinux, se genera un modulo de politicas de seguridad con el siguiente ejemplo:

```
$ ausearch -c "telnetd" -m avc | audit2allow -M telnetd
```

Esto generara 3 archivos:

- `telnet.te` : que es el conjunto de reglas de seguridad que permiten o denegan permisos a tipos, roles, atributos y dominios de seguridad. Sobre los archivos de esta clase es que trabajaremos la mayoria del tiempo.
- `telnet.pp` : que es el paquete binario que contiene el modulo (o los modulos) a cargar hacia SELinux, para la ejecucion de las nuevas politicas.
- `telnet.mod` : que es el modulo de politicas compiladas, no empaquetadas aun.

Despues de generados todos los archivos, si no hay que hacer ninguna modificacion en las reglas, se procede a la instalacion con `semodule -i telnetd.pp`; ya esto nos permitira acceso a los recursos que nos fueron negados.

El tipo de politicas de enforzamiento que usaremos sera el modo `strict` (estricto), el cual nos provee un conjunto de politicas de proteccion para todos los daemons (servicios) del sistema.

## ***¿Como puedo conocer el estado actual de SELinux?***

Para conocer el estado actual de SELinux, debe ejecutar el siguiente comando:

```
$ getenforce
```

Un valor *Disable* le indica que SELinux esta desactivado, *Permissive* le indica que SELinux esta desactivado pero registrando cualquier evento que infringe sus políticas sin tomar ninguna acción a cambio y finalmente un valor de *Enforcing* le indica que SELinux esta actualmente activo y listo para actuar ante cualquier evento que infrinja cualquiera de sus políticas.

Incidentalmente también puede usar el comando `seinfo` para ampliar la información de estado mostrada por `getenforce`, para conocer por ejemplo, cantidad de reglas, atributos, roles, cadenas booleanas, tipo de política, versión, etc. También podría ser útil listar algunas sentencias booleanas con `seinfo -b`.

## ***¿Que aplicaciones estan siendo denegadas por SELinux en mi sistema?***

Fedora incluye potentes herramientas para auditar SELinux y con ello una completa descripción que nos ayudarán a solucionar cualquier problema relacionado a aquellos eventos que infringen las políticas de SELinux, puede utilizar la herramienta gráfica incluida llamada `setroubleshoot`, en caso de no disponer de soporte XWindow puede también ejecutar en modo consola:

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

## ***¿Como puedo permitir una aplicacion ya denegada por SELinux en el sistema?***

Es necesario crear políticas de acceso detalladas para cada aplicación que se desea tenga acceso a determinado objeto/dominio en su sistema, esta tarea puede ser terriblemente compleja, es por esto que resulta importante auditar las políticas mediante `sealert` o `setroubleshoot`, allí se dará cuenta que tipo de acceso fue denegado con mensajes del tipo: `avc: denied { read, write, unlink, etc. }`, basado en esto podemos crear y cargar políticas en tiempo de ejecución que permitan exactamente los tipos de acceso que SELinux esta prohibiendo y registrando.

Vamos a tomar como ejemplo el comando *su* y supongamos que SELinux esta denegando su ejecución con mensajes del siguiente tipo:

```
avc: denied { getattr } for pid=3414 comm="su" name="shadow" dev=dm-3
```

Aquí SELinux nos dice que denego el acceso al comando *su* cuando éste intentó leer los atributos del archivo */etc/shadow* sobre la partición *dm-3*, basado en esto crearemos una política que permita justamente el acceso al programa *su* para que pueda leer los atributos del archivo */etc/shadow* únicamente sobre la partición *dm-3* de la siguiente manera:

```
$ ausearch -c "su" -m avc | audit2allow -M su ; semodule -i su.pp
```

Analizando la linea anterior podemos ver que inicialmente el comando *ausearch* busca eventos que corresponden al comando *su*, luego en base a esto *audit2allow* genera un módulo de políticas en binario llamado *su.pp* y finalmente *semodule* carga este archivo de políticas en SELinux.

El ejemplo siguiente nos muestra cual es el código de políticas generado, del cual explicaremos como funciona y como deben realizarse las políticas para un programa determinado.

```
module su 1.0;

require {
    class dir search;
    class file { execute getattr };
    type bin_t;
    type default_t;
    type selinux_config_t;
    type staff_su_t;
    role staff_r;
};

allow staff_su_t bin_t:file execute;
allow staff_su_t default_t:dir search;
allow staff_su_t selinux_config_t:file getattr;
```

En un analisis rapido de este codigo, podemos apreciar lo siguiente:

- Se debe declarar el nombre del modulo en la primera linea, y con esta se debe declarar cual es el numero de version de la politica: *module su 1.0;*
- Se define cuales son las clases, macros, tipos, atributos, conjunto de subclases, y roles a permitir en la politica
  - Para definir los requerimientos es usa *require { ... };*
  - Dentro de *require*:
    - se definen las clases de acceso de seguridad a usar con *class* y son:
      - dir: para acceso de los directorios
      - file: para acceso de los archivos
      - process: para acceso de los procesos
    - se definen los tipos a usar con *type*
    - se definen los roles a usar con *role*
    - se definen los atributos de tipos a usar con *attribute*
  - Luego se comienza a permitir la transición de tipos y dominios con *allow*
  - Se debe tomar en cuenta que todo lo que no ha sido declarado como admitido, esta denegado por defecto; ya que, por su estructura de diseño, SELinux viene cerrado y enforzado por defecto; y es nuestro deber moldear las politicas que mejor ajusten con nuestro entorno operacional.

## ¿Como permitir el acceso a las aplicaciones que han sido negadas por el sistema?

Es una buena practica conocer que acceso vamos a otorgar, evaluando y auditando los registros *avc:denied* del archivo de registro de SELinux, para que finalmente crear las reglas en 2 simples pasos y terminar generando el nuevo modulo binario de politicas a aplicar:

```
$ audit2allow -M reglaslocales -l -i /var/log/audit/audit.log
```

Luego se carga la información de las nuevas politicas en SELinux:

```
$ semodule -i reglaslocales.pp
```

Alternativamente, si desea conocer las reglas que exactamente fueron creadas, puede crear el archivo *.te*, el cual contiene las reglas de ejecucion de cada tipo de dominio en SELinux basados en *audit.log*:

```
$ checkmodule -M -m -o reglaslocales reglaslocales.te
```

## ¿Como puedo permitir que un servicio pueda usar cualquier puerto?

Regularmente SELinux solo permite que determinadas aplicaciones utilicen *solo* aquellos puertos para los cuáles han sido configurados por defecto, por ejemplo, SSH regularmente escucha por el puerto 22 y de estar SELinux con la política de referencia *strict* resultaría casi imposible intentar cambiarlo a otro puerto, como por ejemplo al puerto 10022, para lograrlo necesitamos desplegar un conjunto de reglas confinadas en el dominio de SELinux llamado "*sshd\_port*" para ssh de la siguiente manera:

```
$ semanage port -a -t sshd_port_t -p tcp 10022
```

Otro ejemplo util es cambiar el puerto 80 por defecto por donde escucha el servidor web Apache:

```
$ semanage port -a -t http_port_t -p tcp 81
```

Sin embargo, en algunas aplicaciones es común el cambio del puerto de servicio, tal es el caso de **Squid**, regularmente squid escucha por el puerto 3128 pero algunos administradores prefieren usar el 8080 e incluso el 80, SELinux tiene una cadena booleana específica para squid llamada **squid\_connect\_any** la cuál permite cambiar el puerto de servicio con bastante flexibilidad:

```
$ setsebool -P squid_connect_any 1
```

## ¿Cómo puedo cambiar la localización por defecto de algunos servicios?

Casos comunes, supongamos que queremos cambiar la localización de nuestro servidor web, muchos administradores o webmasters cambian premeditadamente la ruta que trae Apache por defecto, por ejemplo un proveedor de hosting acostumbra a utilizar el directorio `/home/web/` para alojar las páginas webs de todo un universo de dominios, así pues el creera oportuno aplicar las políticas predefinidas para el servicio Apache, esto lo podemos lograr agregando el tipo de contexto a los archivos y directorios de la siguiente forma:

```
# semanage fcontext -a -t httpd_sys_content_t "/home/web(/.*)?"
```

Otro servicio es la base de datos MySQL, al cambiar la localización de los archivos de bases de datos que comúnmente se encuentran en `/var/lib/mysql/` a por ejemplo `/db/mysql/` debemos etiquetar sus archivos con:

```
$ semanage fcontext -a -t mysqld_db_t "/db/mysql(/.*)?"
```

Y así con todos los demás servicios de la misma forma, si no sabes cuál es el contexto de los archivos en SELinux para tu servicio, puedes simplemente buscarlo:

```
$ semanage fcontext -l | grep dhcpd
/var/lib/dhcp(3)?/dhcpd\.leases.* regular file system_u:object_r:dhcpd_state_t:s0
/var/lib/dhcpd(/.*)?          all files  system_u:object_r:dhcpd_state_t:s0
/usr/sbin/dhcpd.*             regular file system_u:object_r:dhcpd_exec_t:s0
/etc/dhcpd\.conf              regular file system_u:object_r:dhcp_etc_t:s0
/var/run/dhcpd\.pid           regular file system_u:object_r:dhcpd_var_run_t:s0
```

Entonces podemos deducir que, si deseo cambiar los datos almacenados en `/var/lib/dhcpd` para otro directorio llamado `/data/dhcpd` debo aplicarles el contexto de archivos `dhcpd_state_t`.

```
$ semanage fcontext -a -t dhcpd_state_t "/data/dhcpd(/.*)?"
```