

Control de Acceso Obligatorio (MAC) con SELinux

¡PROHIBIDO EL PASO!

SELinux proporciona un sistema más seguro mediante el potente concepto de controles de acceso obligatorios. **POR THORSTEN SCHERF**

Linux es un sistema operativo extremadamente seguro, pero los permisos de acceso heredados no proporcionan protección frente a un software mal configurado o programado. Si un programa se descontrola porque el administrador ha olvidado instalar el último parche, o si un usuario realiza una escalada de privilegios debido a una configuración incorrecta, la seguridad original del sistema se queda sin protección. SELinux mitiga este peligro potencial al añadir un nivel extra de control de acceso denominado Control de Acceso Obligatorio (Mandatory Access Control, MAC).

Hace unos siete años, la National Security Agency (NSA) [1] lanzó la primera versión de SELinux. Propuesta como una extensión para el kernel 2.4 en aquel entonces, los parches del kernel fueron encontrando su lugar en el kernel 2.6 oficial. En muchas distribuciones, SELinux es parte de la configuración estándar. Los ejemplos comentados en este artículo se basan en la distribución comunitaria basada

en Red Hat, Fedora Core 8, aunque son válidas para cualquier otra plataforma que soporte SELinux. Lo importante es que el soporte necesario del kernel (`CONFIG_SECURITY_SELINUX`) y los paquetes `libselinux`, `policycoreutils` y `selinux-policy-targeted` estén instalados. SELinux requiere también algunos paquetes estándar (`SysV-Init`, `pam`, `util-linux`, `coreutils` y algunos otros).

El sistema de seguridad en Linux original estaba basado en Discretionary

Access Controls (DACs). Esto significa que el propietario de un archivo tiene el control absoluto sobre el objeto que ha creado. Si un usuario obtiene inadvertidamente acceso de escritura sobre ese archivo, no existe ningún proceso separado que valide este paso.

O, si un atacante consigue ejecutar código arbitrario en un servidor Web, aprovechando una vulnerabilidad en el software del servidor Web, el código del programa (por ejemplo, una shell) se ejecutará con los permisos de una cuenta de usuario del servidor en el que se ejecuta. Si la cuenta es la del usuario *apache*, el atacante tiene acceso a todos los archivos permitidos al usuario *apache*.

En la mayoría de los casos no existe un proceso que verifique si el servidor Web necesita acceder realmente a los archivos en cuestión para realizar sus tareas. Un atacante que consiga acceder al sistema puede ser capaz de realizar una escalada de privilegios en esa máquina. Hasta hace muy poco tiempo, muchos sistemas Linux

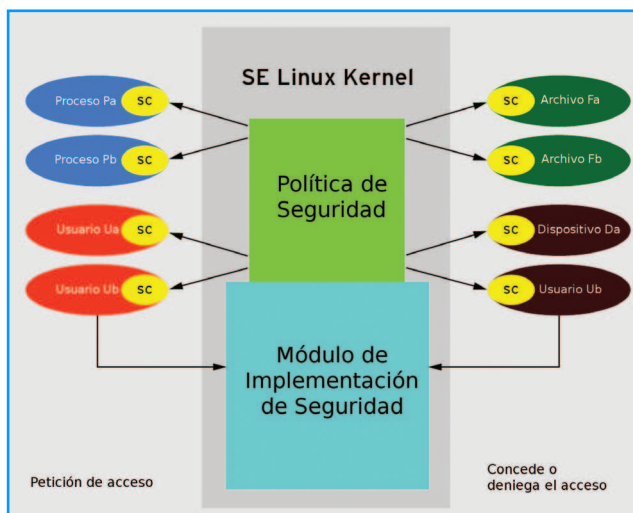


Figura 1: El servidor de seguridad basado en el kernel decide el nivel de acceso.

podían comprometerse debido a un bug en el kernel que permitía al atacante aprovechar la llamada al sistema *vmsplce()*. Como consecuencia, el atacante obtenía el control total del sistema.

En sistemas con SELinux, a cada archivo y a cada objeto se le otorga una etiqueta separada de seguridad mediante MAC para tener un nivel extra de control. En el caso de objetos archivo, Linux guarda esta etiqueta en los atributos extendidos. Al mismo tiempo, a cada proceso, o a cada sujeto también se le otorga su correspondiente etiqueta. Esta etiqueta, conocida como contexto de seguridad, comprende generalmente tres componentes: *User:Role:Type/Domain*. A continuación veamos dos ejemplos del archivo de procesos del servidor Apache */usr/bin/httpd*:

```
# ls -lZ /usr/sbin/httpd
-rwxr-xr-x root
root system_u:object_r:
httpd_exec_t /usr/sbin/httpd
```

y:

```
# ps -AZ|grep httpd
user_u:system_r:httpd_t 2571 ?
00:00:01 httpd
```

Al igual que las etiquetas de seguridad, las ACLs de Posix extienden los atributos de un archivo. En este ejemplo, tienen el siguiente aspecto:

```
# getfattr -d -m
security /usr/sbin/httpd
...
security.selinux=
"system_u:object_r:httpd_
exec_t\000"
```

Etiqueta de Seguridad del Sistema de Archivos

Aunque SELinux funciona en sistemas de archivos que no soportan atributos extendidos, como NFS o ISO9660, los administradores disponen de algunas soluciones para esto. El comando *mount* tiene una opción para controlarlo: *context=<security-label>*. Esto permite especificar una etiqueta de seguridad para el sistema de archivos completo.

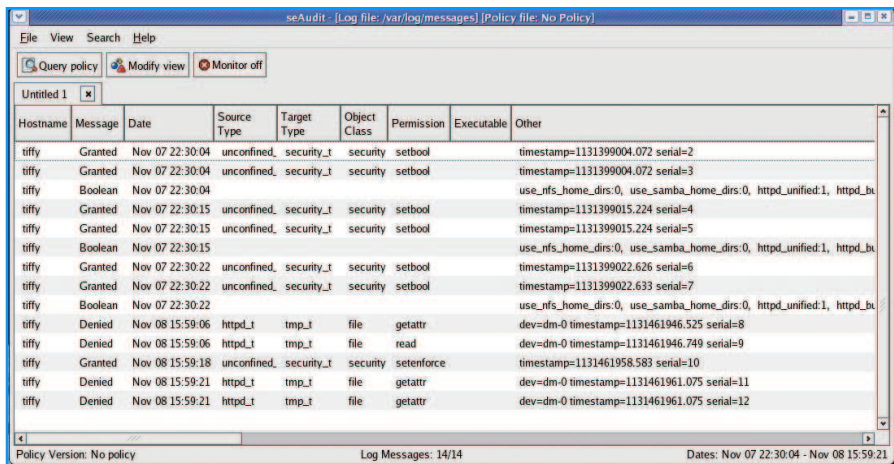


Figura 2: La herramienta seaudit muestra todas las entradas del log de SELinux en una lista ordenada.

Política SE

La política es otra componente importante. Una política SE define el acceso entre objetos y sujetos individuales. La

política específica a qué objetos se permite acceder (como en el caso de un proceso httpd con un rol específico). Si el acceso no está explícitamente permiti-

Funciones SELinux

SELinux distingue tres implementaciones diferentes:

- Implementación de Tipos (TE)
- Control de Acceso basado en Rol (RBAC)
- Seguridad Multi-Nivel (MLS)

TE especifica a qué sujeto se le permite acceder a qué objetos, por ejemplo, a qué procesos se les permite el acceso a determinados archivos. Sin embargo, existe una amplia variedad de objetos, como puertos de red o áreas de memoria. SELinux asigna un dominio para cada sujeto y un tipo para cada objeto. Para explicar esto genéricamente, la implementación del tipo controla a qué dominios se les permite acceder a qué tipos. Ambos, tipos y dominios se identifican de manera similar: siempre terminan con una *_t* (por ejemplo, *httpd_t*).

RBAC usa un modelo abstracto de usuario y asigna a cada usuario exactamente un rol. Los usuarios heredan los permisos asignados a este rol. Por tanto, es posible asignar un rol al usuario root que no posea ningún permiso de administración. Para cambiar a otro rol con permisos ampliados, el usuario previamente tendrá que autenticarse con una contraseña. Esta es una funcionalidad interesante si queremos configurar el equipo sin el omnipresente usuario root. El usuario sólo puede asumir un único rol en un momento determinado. Sin embargo, pueden hacer uso del comando *newrole* (que es similar a *su*) para cambiar roles, suponiendo que la política lo permita. Un nombre de rol típico es *user_r* (los roles siempre terminan con *_r*).

Russell Coker ofrece un par de máquinas de prueba SELinux en Internet [2] para permitir que la gente haga sus experimentos. Estas máquinas muestran la funcionalidad RBAC. Coker ha publicado la cuenta de root para éstas, permitiendo que cualquier persona que esté interesada se pueda loguear como administrador. Los que lo prueben notarán pronto que el conjunto de comandos disponible está extremadamente restringido, desde el momento en que el usuario *root* no es el usuario administrador de estas máquinas de prueba.

Finalmente, MLS define diferentes niveles de seguridad y se usa principalmente en entornos de alta seguridad, como aplicaciones militares. A los objetos se les asignan niveles de seguridad (confidencial, estrictamente confidencial, secreto y demás), y a los sujetos se les da permisos en función de estos niveles de confidencialidad. Allí donde se despliegue MLS, la etiqueta de seguridad se extiende añadiendo un cuarto y quinto componente hasta que tenga un aspecto como este:

User:Role:Type/
Domain:Sensitivity:Category

tido, se registra en primer lugar y finalmente se prohíbe, al menos en el modo implementación.

El servidor de seguridad del kernel se asegura de que no se produzcan infracciones de la política. Es una entidad a la que se referencia la política, mientras que la etiqueta de seguridad define si se permite el acceso. Para evitar mermas del rendimiento, el servidor basado en el kernel usa el Access Vector Cache (AVC)

Demostración

Como breve demostración de cómo funciona este tipo de implementación, consideraremos el siguiente ejemplo: el administrador crea un archivo llamado *index.html* en el directorio */tmp*. Tras hacerlo, mueve (no copia) el archivo al directorio raíz del servidor Web, que en Fedora es */var/www/html/*. Finalmente ejecuta el servidor Web (*/etc/init.d/httpd start*) y accede a la página creada mediante un navegador Web (*http://local-host/index.html*). Si el modo de implementación SELinux está habilitado (por defecto sí lo está), el navegador Web muestra un mensaje de error debido a que *index.html* fue creado en */tmp* y heredó su etiqueta de seguridad:

```
# ls -lZ /tmp/index.html
-rw-r--r-- root root 2
root:object_r:tmp_t 2
/tmp/index.html
```

Figura 3: seaudit-report puede generar estadísticas en HTML.

Por el contrario, el proceso del servidor Web se ejecuta en el dominio denominado *httpd_t*:

```
# ps -AZ|grep httpd
user_u:system_r:httpd_t 2
2571 ? 2
00:00:02 httpd
```

La política precisa una regla que proporcione al dominio *httpd_t* acceso a los archivos de tipo *tmp_t*, pero esta regla no existe. El servidor Web necesita acceder a sus propios archivos de configuración, scripts CGI y otro contenido de su directorio. Existen tipos

específicos para todos estos archivos, por ejemplo: *httpd_config_t*, *httpd_log_t*, *httpd_sys_script_exec_t* y *httpd_sys_content_t*.

Los archivos del directorio */tmp* no son generalmente del tipo de objetos al que accede un servidor Web. Por tanto, no permite instrucciones en su archivo de política para el tipo de *tmp_t*.

Si comprobamos el archivo */var/log/audit/audit.log*, encontraremos un mensaje que avisa de que se acaba de producir un intento no autorizado de abrir un archivo:

```
... audit(1202241301.521:12): 2
avc: denied
{getattr } for pid=6608 2
comm="httpd" name=2
"index.html" dev=dm-0 ino=179881
scontext=user_u:system_r:2
httpd_t tcontext=2
root:object_r:tmp_t tclass=file
```

Esta entrada de log revela que el proceso con PID 6608 y nombre *httpd* acaba de intentar efectuar la llamada al sistema *getattr* (es decir, ha intentado llamar a sus atributos) sobre un archivo con número de inodo 179881 y nombre *index.html*. *scontext* y *tcontext* refieren a la etiqueta de seguridad para el proceso origen (*apache*) y el archivo objetivo (*index.html*). *avc: denied* indica que el servidor de seguridad que ejecuta el kernel ha impedido esta acción.

Si preferimos un método más estructurado, podemos usar *seaudit* para ver

Listado 1: sealert -

Summary

SELinux is preventing the */usr/sbin/httpd* from using potentially mislabeled files (*/var/www/html/index.html*).

Detailed Description

SELinux has denied */usr/sbin/httpd* access to potentially mislabeled file(s) (*/var/www/html/index.html*). This means that SELinux will not allow */usr/sbin/httpd* to use these files. It is common for users to edit files in their home directory or *tmp* directories and then move (*mv*) them to system directories. The problem is that the files end up with the wrong file context which confined applications are not allowed to access.

Allowing Access

If you want */usr/sbin/httpd* to access this files, you need to relabel them using *restorecon -v /var/www/html/index.html*. You might want to relabel the entire directory using *restorecon -R -v /var/www/html*.

...

archivos de log (véase la Figura 2). La herramienta muestra entradas de log individuales de forma gráfica.

Mediante la llamada a *seaudit-report* —*html logfile* podemos incluso generar una página HTML que muestre tanto los logs como diferentes estadísticas del sistema SELinux (véase la Figura 3).

Problema a la Vista

Si se está ejecutando *setroubleshootd*, se muestra la siguiente entrada en el archivo de log */var/log/messages*:

```
setroubleshoot: #012
SELinux is preventing the /usr/sbin/httpd
from using potentially mislabeled files
(/var/www/html/index.html).
#012 For complete SELinux messages
run sealert -l 5e982b3b-3ae7-4848-b8bd-7d8553a07732
```

El demonio *setroubleshoot* se desarrolló hace algún tiempo para hacer que los mensajes, más bien crípticos, generados por el demonio de auditoría fuesen más legibles, y para ofrecer pistas al usuario a la hora de resolver problemas. Si el usuario ejecuta el comando especificado en la entrada de log, podrá ver la explicación mostrada en el Listado 1.

El escritorio Gnome presenta un pequeño icono (un escudo amarillo) en la barra de tareas cada vez que se cree una entrada en el log de SELinux. Los usuarios pueden pulsar sobre él para que aparezca el navegador gráfico de resolución de problemas SELinux, que nos permite explorar a través de los mensajes procesados (véase la Figura 4). Esta herramienta ofrece una manera de resolver problemas a los administradores novatos:

```
restorecon -v /var/www/html/index.html
```

El comando usa una política para fijar la etiqueta correcta para el archivo *index.html*. Alternativamente, el administrador puede especificar el tipo de archivo correcto de forma manual:

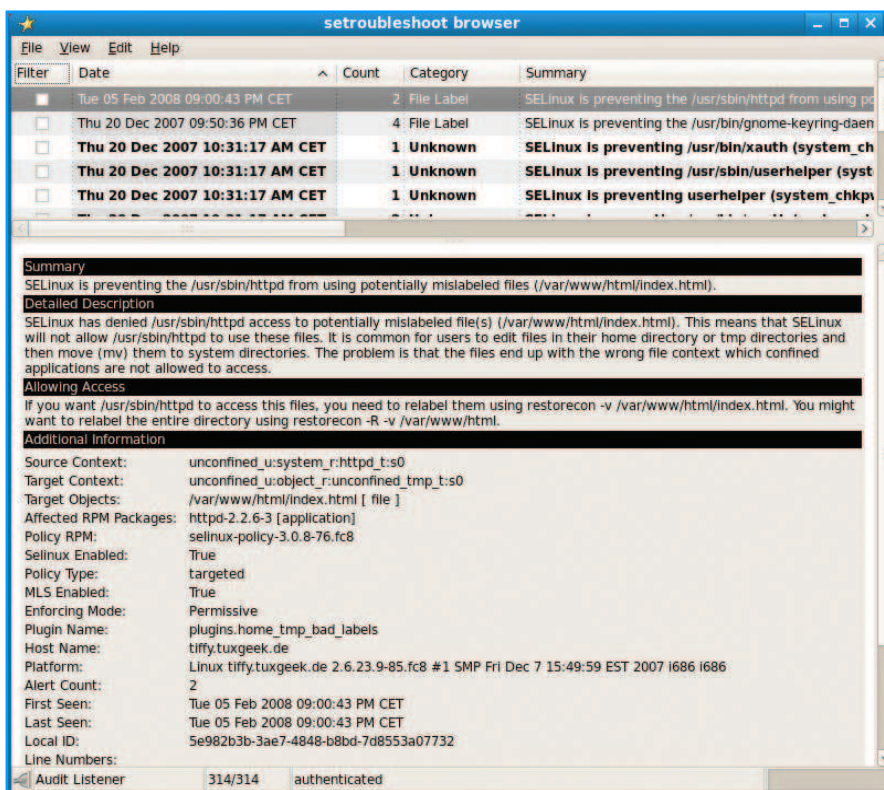


Figura 4: setroubleshootd ofrece un front end gráfico para los archivos de log.

```
chcon -t httpd_sys_content_t /var/www/html/index.html
```

De una manera u otra, los resultados deberían ser:

```
# ls -lZ /var/www/html/index.html
-rw-r--r-- root httpd_sys_content_t /var/www/html/index.html
```

La próxima vez que alguien trate de mostrar un archivo en un navegador Web no debería haber ningún obstáculo de seguridad.

El ejemplo muestra cómo funciona SELinux. Independientemente de per-

misos anteriores, Linux sólo permite el acceso si existe la correspondiente entrada en la política de SELinux (MAC). El administrador de seguridad sólo creará esta entrada si el acceso es realmente necesario.

Administración

Para la administración de un sistema SELinux disponemos de varias herramientas. Por ejemplo, una herramienta denominada *getenforce* muestra el modo actual de SELinux. *setenforce 0|1* permite al usuario cambiar el modo, donde *0* representa el modo permisivo, y *1* el modo restrictivo. El modo permisivo significa que una acción no autorizada se registra pero no se prohíbe, lo que es útil si estamos desarrollando un nuevo módulo de políticas. El servi-

```
Listado 2: /etc/selinux/config

# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#   enforcing - SELinux security policy is enforced.
#   permissive - SELinux prints warnings instead of enforcing.
#   disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#   targeted - Targeted processes are protected,
#   mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

dor de seguridad refiere a sus entradas de políticas para decidir qué está permitido. Para cambiar un modo de forma permanente necesitamos una entrada en el archivo `/etc/selinux/config` (véase el Listado 2).

La herramienta más interesante para SELinux es probablemente `system-config-selinux` (véase la Figura 5). Permite al administrador efectuar configuraciones básicas, como el modo de SELinux, y soporta tareas más complejas, como la creación de nuevos módulos de políticas. Podemos también configurar operaciones booleanas, que son instrucciones simples que habilitan reglas de políticas que hemos preparado, sin que para ello haga falta usar el lenguaje de macros `m4` (toda la política está basada en este lenguaje).

Existe toda una gama de operaciones booleanas predefinidas. Por ejemplo, podemos permitir que un servidor Web acceda a información ubicada en las carpetas de los usuarios (`UserDir`) o permitir que el servidor de nombres ejecute cambios en el archivo de zonas (`DDNS`). Los operadores booleanos generalmente pueden mostrarse en línea de comandos con `getsebool`. El comando `getsebool -a | grep httpd`, por ejemplo, lista todos los operadores booleanos del servidor Web Apache (véase el Listado 3).

Diferentes páginas man describen los operadores booleanos para los servicios de red más populares. La página `httpd_selinux` nos puede ayudar con el servidor Web.

Igualmente, podemos cambiar los operadores booleanos en línea de comandos con `setsebool`. El siguiente comando permite que ejecutar scripts CGI al servidor web:

```
setsebool -P httpd_enable_cgi 1
```

`sestatus` es una interesante herramienta en línea de comandos que resume la configuración actual de SELinux (véase el Listado 4).

RHEL 4 tenía una política monolítica pura, aunque hoy día se usan variantes modulares de la misma. Esto permite al administrador numerosas ventajas. Por ejemplo, un desarrollador de política ya no necesita preocuparse acerca de las fuentes completas de la política para el sistema SELinux. Es suficiente

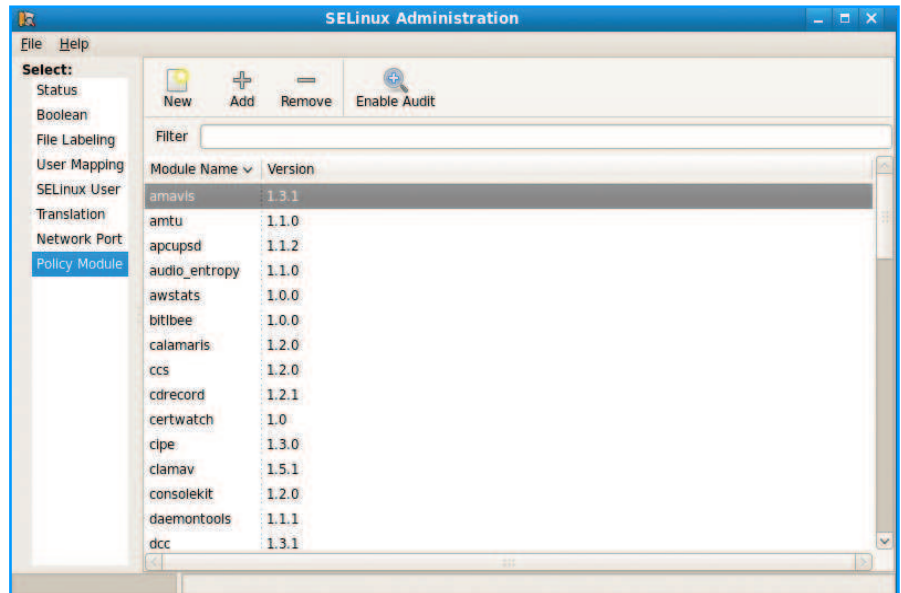


Figura 5: La herramienta de configuración gráfica `system-config-selinux` permite una administración adecuada del sistema SELinux.

con desarrollar un único módulo para la aplicación que queramos proteger y añadir este módulo al sistema.

La política por defecto, que es parte de la distribución Fedora (Targeted Policy), protege a muchas aplicaciones directamente. Los programas protegidos son referidos como targeted programs, lo que explica el nombre de la política. El comando `semodule` devuelve todos los módulos de políticas disponibles (véase el Listado 5).

Si el administrador quiere eliminar un módulo, y de esta manera la protección SELinux para este programa, simplemente puede pasarle el nombre del módulo con la opción `-r`:

```
semodule -r amavis
```

Al hacer esto se elimina la protección permanente para la aplicación especificada, aunque podemos reintroducir el módulo más tarde. El RPM de políticas guarda todos los módulos estándar en el directorio `/usr/share/selinux/targeted`. Un administrador puede recargar el módulo de Amavis mediante la siguiente llamada a `semodule`:

```
semodule -i   
 /usr/share/selinux/targeted/  
 amavis.pp
```

El comando recarga el módulo Amavis del servidor de seguridad que se ejecuta en el kernel. El conjunto de reglas

del módulo asume la responsabilidad de denegar o permitir cualquier acción que tenga que ver con el software Amavis. Si necesitamos un resumen preciso de qué reglas incluyen los módulos individuales, podemos instalar el SRPM (Source RPM) para la política que estamos usando, o simplemente instalar la herramienta gráfica `apol tool`, que puede mostrar los archivos binarios de políticas en formato de texto simple, permitiendo de esta manera investigar la política establecida.

Si esto le suena demasiado complicado, o si simplemente queremos tener un resumen genérico de la política que hemos implantado, `seinfo` es la herramienta indicadora (véase el Listado 6). Podemos ver con facilidad lo complejo que es en realidad el conjunto de reglas completo.

Una funcionalidad de la política de SELinux de Fedora 8 es que ahora contiene las propiedades de las anteriores políticas estrictas. Para ser más precisos, actualmente es posible usar la política de objetivos para restringir cuentas de usuario (es decir, para implementar RBAC). Por ejemplo, Dan Walsh ha publicado un módulo de políticas denominado `xguest` [3]. Éste permite al administrador convertir rápidamente cualquier escritorio Gnome en un sistema kiosk. Al usuario se le permite loguearse como `xguest`, y tiene permisos muy limitados sobre el

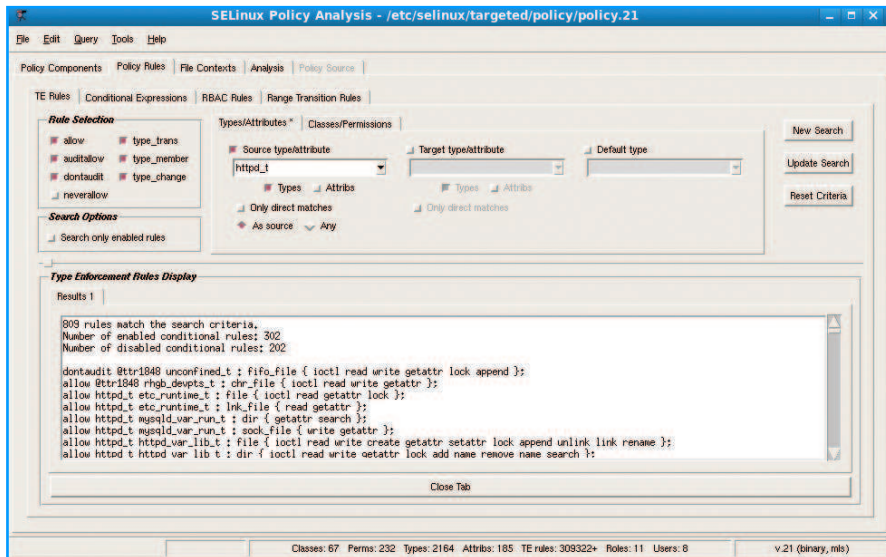


Figura 6: La herramienta gráfica apol muestra la política del binario en texto simple.

escritorio Gnome y una selección de programas muy restringida. Por ejemplo, el acceso a red se restringe al navegador Web Firefox. El resto de aplicaciones no tienen acceso a la red.

El módulo también prohíbe modificaciones en la configuración de Gnome (*gconf*). Esto es un entorno ideal para sistemas kiosk, como los que podemos encontrar en aeropuertos y vestíbulos de hoteles. El módulo de política *xguest* puede servir también como punto de partida para un desarrollo propio. Por ejemplo, podríamos añadir más instrucciones a las reglas existen-

tes para soportar el acceso basado en SSH.

Desarrollo

Si preferimos contribuir activamente con SELinux, en lugar de configurar simplemente sus políticas, Fedora 8 dispone de distintas herramientas que nos lo permiten. Por ejemplo, podemos modificar al vuelo, las políticas binarias con la herramienta *semanage* sin tener que acudir a las fuentes. Por supuesto, podemos cambiar cada una de las propiedades de esta manera, pero este método gene-

ralmente es mejor para los cambios más sencillos.

La política SELinux ofrece al servidor Web Apache la capacidad de fijarse a puertos de red específicos. Estos puertos se designan como tipos *http_port_t* en la política de SELinux. Al ejecutar *semanage* le indicamos qué puertos se han etiquetado de esta manera:

```
# semanage port -l |
grep http_port_t
tcp 80, 443, 488, 8008, 8009
```

Un administrador que quiera aplicar esta etiqueta a un nuevo puerto debe ejecutar *semanage* de la siguiente manera:

```
semanage port -a -t
http_port_t -p tcp 777
```

Y al ejecutar *apol* contra la política para encontrar una regla coincidente, éste revela lo siguiente:

```
allow httpd_t
http_port_t : tcp_socket
{ name_bind name_connect };
```

La regla permite a los procesos del dominio *httpd_t* acceder a cualquier puerto de red que tenga la etiqueta *httpd_port_t*. Estos puertos son ahora el 777, 80, 443, 488, 8008 y 8009.

Si quisiéramos llevar esta idea un paso más allá y crear módulos completamente nuevos por nuestra cuenta, existen dos herramientas a nuestra disposición: *system-config-selinux* y *policygentool*. *policygentool* forma parte del RPM *selinux-policy-devel* ubicado en el directorio */usr/share/seli-*

Listado 3: getsebool -a | grep httpd

```
allow_httpd_anon_write -> off
allow_httpd_dbus_avahi -> off
allow_httpd_mod_auth_pam -> off
allow_httpd_sys_script_anon_
write -> off
httpd_builtin_scripting -> on
httpd_can_network_connect ->
off
httpd_can_network_connect_db ->
off
httpd_can_network_relay -> off
httpd_can_sendmail -> off
httpd_enable_cgi -> on
httpd_enable_ftp_server -> off
httpd_enable_homedirs -> on
httpd_ssi_exec -> off
httpd_tty_comm -> on
httpd_unified -> on
httpd_use_cifs -> off
httpd_use_nfs -> off
```

Listado 4: sestatus -b

```
SELinux status:
enabled
SELinuxfs mount:
/selinux
Current mode:
permissive
Mode from config file:
permissive
Policy version:
21
Policy from config file:
targeted
Policy booleans:
allow_console_login
off
allow_cvs_read_shadow
off
allow_daemons_dump_core
on
allow_daemons_use_tty
on
allow_execheap
off
allow_execmem
on
...
```

Listado 5: semodule -l

```
amavis 1.3.1
amtu 1.1.0
apcupsd 1.1.2
audio_entropy 1.1.0
awstats 1.0.0
bitlbee 1.0.0
calamaris 1.2.0
ccs 1.2.0
cdrecord 1.2.1
certwatch 1.0
cipe 1.3.0
clamav 1.5.1
...
```

Listado 6: seinfo

```

Statistics for policy file: /etc/selinux/targeted/policy/policy.21
Policy Version & Type: v.21 (binary, mls)
Classes:          67  Permissions:      232
Sensitivities:   1   Categories:      1024
Types:           2166 Attributes:        185
Users:           8   Roles:           11
Booleans:        115 Cond. Expr.:     144
Allow:           17180 Neverallow:      0
Auditallow:      28  Dontaudit:       134379
Type_trans:      3286  Type_change:     88
Type_member:     14   Role allow:      16
Role_trans:      3   Range_trans:    158
Constraints:     59  Validatetrans:  0
Initial SIDs:   27   Fs_use:          17
Genfscon:       66   Portcon:         292
Netifcon:       0   Nodecon:         8

```

nux/devel. Nos ayuda a crear los archivos que vamos a necesitar para generar el binario de módulos de políticas. Una descripción de los pasos a seguir podría llenar un libro fácilmente, por lo que sólo vamos a ver un breve resumen de las instrucciones necesarias en la siguiente sección.

Crear una Política

En primer lugar, el administrador ejecuta una herramienta y le pasa el nombre de la aplicación que queremos proteger junto con el nombre del módulo de políticas a crear:

```
./policygentool foo ↵
/usr/bin/foo
```

La herramienta nos pregunta unos cuantos detalles acerca de la aplica-

ción, si usa un script init, dónde guarda los archivos de log, etc.

Una vez que hemos respondido a estas preguntas, *policygentool* crea tres archivos: *foo.fc*, *foo.if* y *foo.te*, que son el contexto de archivos, el tipo de implementación y la interfaz, respectivamente. El archivo de contexto de archivos permite al administrador enlazar los archivos de aplicación a una etiqueta SELinux. El tipo de implementación especifica las reglas coincidentes, es decir, qué le está permitido hacer a la aplicación. El archivo de interfaz pone las macros a disposición de otros módulos de políticas.

Una vez que hemos editado los archivos, podemos crear el módulo de políticas con la siguiente sentencia:

```
make -f /usr/share/selinux2
/dev/Makefile
```

Tras este paso deberíamos encontrar el nuevo archivo *foo.pp* bajo el directorio actual. Si ejecutamos el comando *semodule -i foo.pp*, se carga el módulo en el servidor de seguridad del kernel.

Si el proceso completo le parece demasiado complicado, siempre podremos usar el front end gráfico, *system-config-selinux*, para crear nuevos módulos de políticas. Para ello podemos acudir al extenso tutorial disponible en [4].

Conclusiones

SELinux es una extensión de seguridad muy útil. Una vez activado se ejecuta en segundo plano de manera más o menos transparente, monitorizando el sistema en ejecución (siempre que la distribución haya proporcionado una política que merezca recibir ese nombre). En el momento de escribir este artículo, Fedora es la distribución líder en este aspecto.

Los lanzamientos más recientes han mejorado la usabilidad de SELinux. Por ejemplo, los logs de SELinux son más fáciles de leer que antes gracias a la herramienta *setroubleshootd*. Incluso los usuarios sin experiencia pueden desarrollar sus propios módulos de políticas para ubicar nuevos programas bajo el escudo de protección de SELinux gracias a la ayuda del front end gráfico *system-config-selinux*.

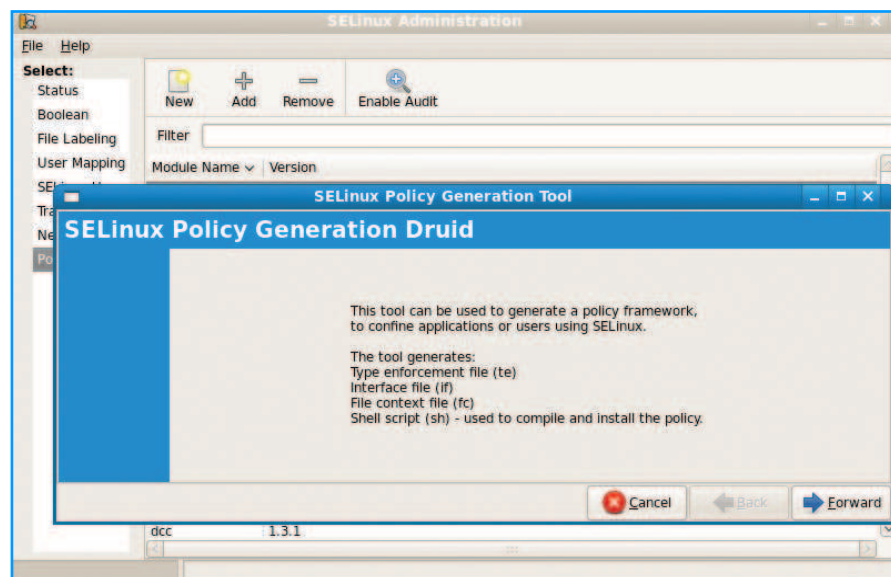


Figura 7: *system-config-selinux* facilita crear un nuevo módulo de políticas.

RECURSOS

- [1] Página de SELinux en la NSA: <http://www.nsa.gov/selinux>
- [2] Máquinas de prueba SELinux Debian de Russell Cocker: <http://www.coker.com.au/selinux/play.html>
- [3] Creación de una cuenta Kiosk, por Dan Walsh: <http://danwalsh.livejournal.com/13376>
- [4] "Guía paso a paso para crear un nuevo módulo de políticas", por Dan Walsh, Red Hat Magazine, Agosto de 2007: <http://www.redhatmagazine.com/2007/08/21/a-step-by-step-guide-to-building-a-new-selinux-policy-module.html>