SELinux

Was Sie schon immer über SELinux
wissen wollen aber nie gefragt haben!

SELinux Einführung

Inhalt

- Was ist SELinux - Geschichte
- Was ist SELinux - Features
- Was ist SELinux - Vokabeln
- Vorteile / Nachteile
- Funktionsweise
- Betriebsmodus
- Policy
- Booleans
- Userland Tools
- Audit (LAF)
- Beispiel
- Fazit

Was ist SELinux

Geschichte

- 1976 Bell & La Padula veröffentlichen ein Grundlagenpapier
- 1992/1993 Erster Forschungsarbeiten – DTMach
- 1999 Ursprung im Projekt FLASK („ Flux Advanced Security Kernel“)
- 1999 Erster Patch von der NSA, Kernel 2.2
- 2001 SELinux Patch für Kernel 2.4
- 2002 Implementation LSM („Linux Security Modules“)

Was ist SELinux

Geschichte

- Seit 2003 im Mainline („Vanilla“) Kernel
- 2004 Freigabe Fedora Core 3 mit Targeted Policy
- Seit 2005 SELinux in Redhat Enterprise Linux verfügbar, RHEL4
- Seit 2007 SELinux in der Standardinstallation aktiv, RHEL5
- 2007 EAL4+/LSPP Zertifizierung nach common criteria, RHEL5

Was ist SELinux

Features – Bell LaPadula

- Mandatory Access Control (MAC)
- Type Enforcement (TE)
- Rule Based Access Control (RBAC)
- „no read up, no write down“

Was ist SELinux

Features – SELinux Implementation

- Ergänzt die bestehende DAC
- Modus zur Laufzeit wählbar (enforcing vs permissive).
- Flexible Policy, zur Laufzeit änderbar.
- AVC („Access Vector Cache“).
- Zusätzlich zur DAC („Discretionary Access Control“).
- „root“ ist nicht mehr Allmächtig.
- Prozesse laufen in eigenen Sicherheitsdomänen.

Was ist SELinux - Vokabeln

DAC - Discretionary Access Control

- Klassisches UNIX Sicherheitskonzept.
- Objektbezogene Zugriffskontrolle.
- Ohne ACL unflexibel.
- Rechte werden pro Benutzer festgelegt.
- Benutzer können eigenmächtig Rechte vergeben.
- SUID verändert die Identität des Benutzers

Was ist SELinux - Vokabeln

MAC - Mandatory Access Control

- Zugriffsberechtigung durch zentrale Policy.
- Policy wird durch zentralen „security policy administrator“ bestimmt.
- Feingranulare und flexible Rechtevergabe.
- Prozesse und Threads sind Subjekte.
- Alles andere sind Objekte (Files, TCP Ports, …).
- Policy bezieht sich auf Subjekte und Objekte statt Benutzer.

Was ist SELinux - Vokabeln

Subjekt – Handelnde Elemente des Systems

- Prozess
- Thread

Object - Resourcen

- Datei
- Verzeichnis
- TCP/UDP Port
- Shared Memory Segment
- Device

Was ist SELinux - Vokabeln

TE - Type Enforcement

- Eine Implementation von MAC (SELinux)
- Alle Systemresourcen haben einen Typ/eine Domäne.
- Typ regelt den Zugriff von Objekten auf Subjekte.
- Subjekte des selben Typs haben die selben Rechte.
- Objekte des selben Typs werden immer gleich behandelt.
- Whitelisting – Was nicht geregelt ist, ist verboten.
- Subjekte erben ihren Typ vom erzeugenden Subjekt.

Was ist SELinux - Vokabeln

Security Context

- Sichereitsbereich.
- Objekt: Benutzer:Rolle:Domäne:Level.
- Subjekt: Benutzer:Rolle:Typ:Level
- Benutzer + Rolle != User+Group
- Domäne/Typ relevant

Was ist SELinux - Vokabeln

Policy

- Herzstück der SELinux Implementation.
- Verschiedene Policys werden unterstützt (Targeted, Strict, MLS).
- Definiert Regeln für den Zugriff von Subjekten auf Objekte
- Definiert Regeln für Domain Transition eines Subjekts.

Vorteile vs Nachteile

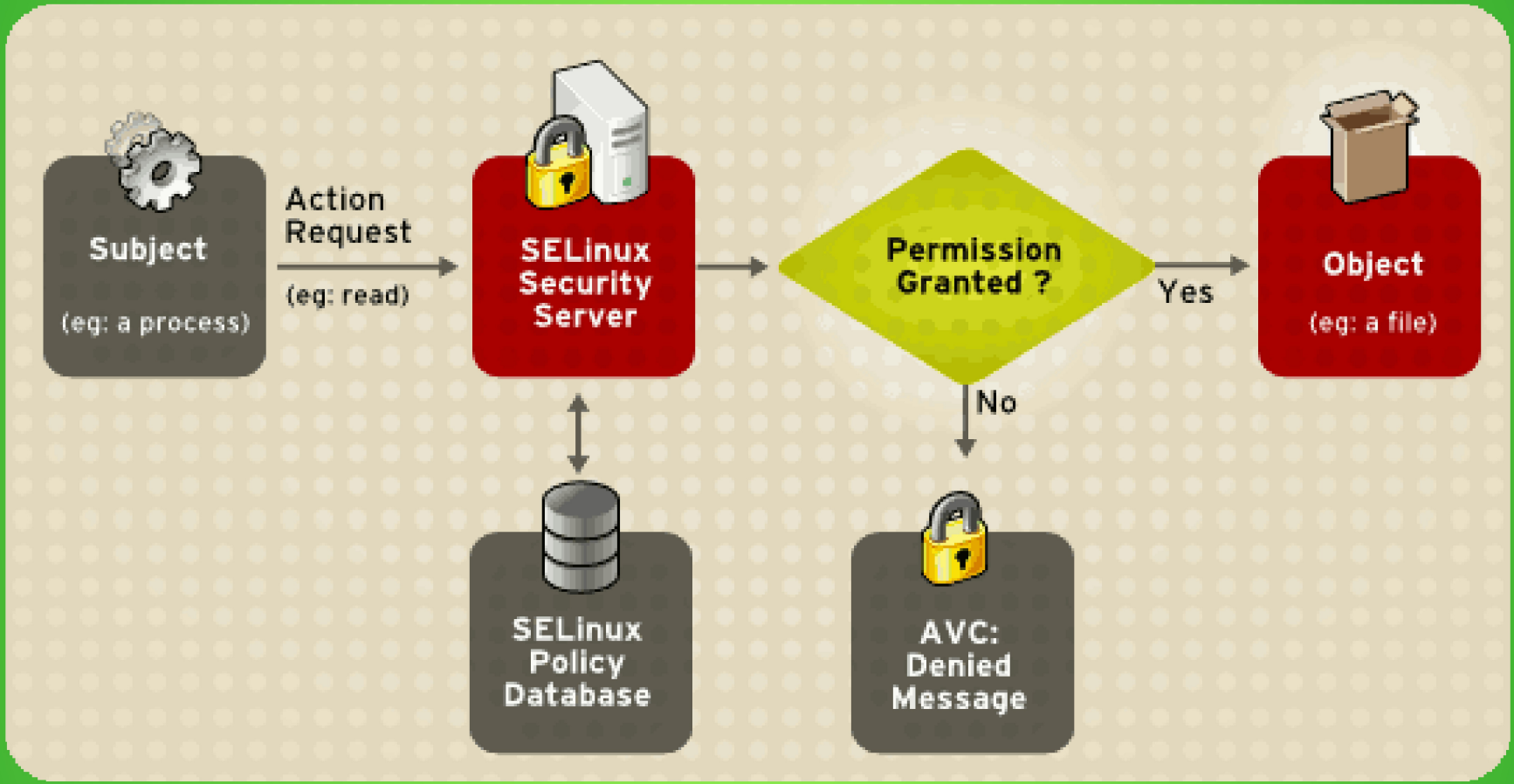
Vorteile

- Erweiterter Systemschutz. Mandatory Access Control.
- Exaktere Zugriffssteuerung.
- SELinux Policy wird Systemweit durchgesetzt.
- „Privilege Escalation“ wird massiv erschwert.
- Policy kann flexibel zur Laufzeit angepasst werden.
- Transparent für Applikationen.
- „root“ ist nicht allmächtig

Nachteile

- Geringfügiger Perfomanceverlust (ca. 7%).
- Erhöhte Komplexität.
- Mögliche Inkompatibilitäten.

Funktionsweise



Betriebsmodus

SELinux kann in verschiedenen Modi betrieben werden. Wobei „enforcing“ der Normalzustand sein sollte. Zur Fehlerbehebung kann es aber manchmal notwendig sein in den „permissive“ Modus zu wechseln.

Alle relevanten Informationen über den aktuellen Betriebsmodus und die geladenen Policy erhält man mit dem Kommando „sestatus“

```
# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /selinux
Current mode:                   enforcing
Mode from config file:          enforcing
Policy version:                 24
Policy from config file:        targeted
```

Betriebsmodus

enforcing

- SELinux ist vollständig aktiv und erzwingt („enforced“) die eingestellte Policy.
- Nicht durch die aktive Policy gedeckte Zugriffe werden durch den Security Server unterbunden.
- Ein Wechsel in den „permissive“ Betriebsmodus ist ohne reboot jederzeit möglich.
- Neu angelegte Dateien werden automatisch mit dem korrekten Filesystemkontext versehen („gelabelt“).

Betriebsmodus

permissive

- SELinux ist aktiv, die Policy wird nicht durchgesetzt.
- Verstöße gegen die Policy werden Protokolliert.
- Nützlich zum finden von Problemen die durch SELinux verursacht werden.
- Wechsel in den „enforcing“ Modus jederzeit ohne reboot möglich
- Filesystemkontexte für neue Dateien werden automatisch korrekt erstellt.

Betriebsmodus

disabled

- SELinux ist vollständig deaktiviert, keine Policy geladen.
- Systemneustart notwendig um SELinux zu deaktivieren.
- Reaktivierung erfordert Systemneustart und ein komplettes relabeling aller Dateien.
- Aus Anwendungssicht kein Unterschied zum „permissive“ Modus.

Policy

targeted

- Standardpolicy für RHEL Systeme seit RHEL4.
- Vergleichsweise niedrige Komplexität.
- Nur ausgesuchte Prozesse laufen in einer eigenen Domäne.
- Finetuning mit „booleans“.
- In der Praxis ausreichend.

Policy

mls (multi level security)

- Komplexeste, von Redhat, supportete Policy.
- Alle Prozesse laufen unter SELinux Kontrolle.
- Finetuning mit „booleans“
- RHEL5: Zertifizierung EAL4+/LSPP.
- Nur bei stark erhöhtem Schutzbedürfnis.

Policy - Booleans

Booleans

- Feintunen der aktiven Policy ohne neuladen.
- Keine Kenntnisse über Policyentwicklung notwendig.
- Booleans könen permanent gesetzt werden (rebootfest)
- „semanage boolean -l“. Zeigt booleans der aktiven Policy.
- Relevante „man pages“ -> apropos _selinux

Userland Tools

Angepasste Kommandos

- '-Z' ist der SELinux Switch.
- Gegen 'libselinux.so.1' gelinkt.

ls, ps, netstat, id	Zeigt Sicherheitskontext von Subjekten und Objekten.
cp,mkdir	Erzeugt Objekte im richtigen Kontext

Userland Tools

SELinux utilities

Low-Level Tools zum Administrieren von Systemen unter Selinux Kontrolle.

getenforce/setenforce	SELinux Betriebsmodus ermitteln/umstellen
restorecon/fixfiles/chcon	Sicherheitskontext von Subjekten verändern.
sestatus	Statusüberblick
avcstat	Statistiken des „access vector cache“ anzeigen.

Userland Tools

SELinux services

Bibliotheken für Programme, die unter Selinux Kontrolle stehen sollen.

libselinux.so.1	Die SElinux Bibliothek
libsepol.so.1	SELinux binary policy manipulation library

Userland Tools

Policy management

Kommandos zum Managen/Installieren/Aktualisieren/Entfernen der Policy

setsebool/getsebool	Setzen bzw. anzeigen von SElinux booleans.
semanage	Selinux policy management. Konfigurieren der Policy. Die Änderungen sind Permanent.
semodule	Installieren,aktualisieren und löschen von SElinux Policymodulen. Liste der vorhandenen Module. Neuladen der Policy.

Userland Tools

Policy development

Kommandos zum Entwickeln/Erweitern der Policy

audit2allow	Erzeugt SElinux Regeln, die einen protokollierten Verstoß gegen die Policy erlaube würden.
checkmodule	Übersetzen der Regeln in die binäre Form.
semodule_package	Erzeugt ein SELinux Policy Modul aus der binären Form und optionler weiterer Informationen, z.B. Filekontexte..

Userland Tools

Troubleshooting

Kommandos zum Troubleshooting

audit2why	Analysiert das Selinux Protokoll und „erklärt“ Policyverletzungen.

Audit (LAF)

Audit (LAF)

- Beim Linux Audit Framework handelt um ein System zum Sammeln von Ereignissen, die das laufende Linux System betreffen.
- Das Audit Framework erzeugt, unter Verwendung von Informationen die mit dem Event gesammelt werden, Logeinträge die den Event dokumentieren.
- Die erzeugten Audit Trails können mit den im LAF enthaltenen Werkzeugen bequem analysiert werden.
- Im SELinux Umfeld wird das LAF verwendet um Verstöße gegen die aktive Policy zu protokollieren.
- Die protokollierten Verstöße können mit „audit2allow“ leicht zu Policyerweiterungen verarbeitet werden.

Beispiel

Beispiel

Webserver Content wird in /tmp erzeugt und danach an die endgültige Position im Filesystem verschoben.

```
[root@box ~]# echo 'Hallo Welt!' >/tmp/index.html
[root@box ~]# ls -lZ /tmp/index.html
-rw-r--r--. root root unconfined_u:object_r:user_tmp_t:s0 /tmp/index.html
[root@box ~]# mv /tmp/index.html /var/www/html/
[root@box ~]# cd /var/www/html
[root@box ~]# ls -lZ index.html
-rw-r--r--. root root unconfined_u:object_r:user_tmp_t:s0 index.html
[root@box ~]# wget http://localhost/index.html
--2013-06-05 10:48:11--  http://localhost/index.html
Resolving localhost... 127.0.0.1, ::1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 403 Forbidden
```

Warum gibt das jetzt einen 403er ?

Beispiel

Beispiel

Prüfen wir mal das error_log des Webserver

```
[root@box ~]# cd /var/log/httpd
[root@box ~]# tail error_log
[10:46:46][notice] SELinux policy enabled;
                    httpd running as context
unconfined_u:system_r:httpd_t:s0
[10:46:46] [notice] suEXEC mechanism enabled (wrapper: /usr/sbin/suexec)
[10:46:46] [notice] Digest: generating secret for digest authentication ...
[10:46:46] [notice] Digest: done
[10:46:46] [notice] Apache/2.2.15 configured -- resuming normal operations
[10:47:40] [error] [client 127.0.0.1] (13)Permission denied: file
                    permissions deny server access:
/var/www/html/index.html
```

Hhhmmm, die File Permissions sind doch in Ordnung, warum tut das nicht ?

Beispiel

Beispiel

Prüfen wir mal das audit log

```
[root@box ~]# cd /var/log/audit
[root@box ~]# tail audit.log
type=AVC msg=audit(1370438179.254:4922): avc:  denied  { read } for  pid=4448
comm="httpd" name="index.html" dev=dm-2 ino=563
scontext=unconfined_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:user_tmp_t:s0
tclass=file

type=AVC msg=audit(1370438179.254:4922): avc:  denied  { open } for  pid=4448
comm="httpd" name="index.html" dev=dm-2 ino=563
scontext=unconfined_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:user_tmp_t:s0
tclass=file

type=SYSCALL msg=audit(1370438179.254:4922): arch=c000003e syscall=2 success=yes
exit=10 a0=7f306c0c5ff0 a1=80000 a2=0 a3=2 items=1 ppid=4443 pid=4448 auid=0 uid=48
gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(non
e) ses=136 comm="httpd" exe="/usr/sbin/httpd" subj=unconfined_u:system_r:httpd_t:s0
key=(null)
```

Beispiel

Beispiel

Was sagt audit2why

```
[root@box ~]# audit2why -i /var/log/audit/audit.log
type=AVC msg=audit(1370438179.254:4922): avc:  denied  { open }
for pid=4448 comm="httpd" name="index.html" dev=dm-2 ino=563
scontext=unconfined_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:user_tmp_t:s0
tclass=file

Was caused by:
Missing type enforcement (TE) allow rule.
You can use audit2allow to generate a loadable module to allow this access.
```

Die Meldung ist ein bisschen Irreführend, eigentlich ist nur der Filekontext falsch. Das kann man natürlich auch durch die vorgeschlagenen Policyänderung lösen ....

Beispiel

Beispiel

Mögliche Lösungen – Filekontext korrigieren

```
[root@box ~]# cd /var/www/html
[root@box ~]# ls -lZ index.html
-rw-r--r--. root root unconfined_u:object_r:user_tmp_t:s0 index.html
[root@box ~]# restorecon index.html
[root@box ~]# ls -lZ index.html
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 index.html
[root@box ~]# wget -O- http://localhost/index.html
--2013-06-05 17:36:50--  http://localhost/index.html
Resolving localhost... 127.0.0.1, ::1
Connecting to localhost|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12 [text/html]
Saving to: âSTDOUTâ


100%[=====================================================================>] 12
     --.-K/s   in 0s

2013-06-05 17:36:50 (1.69 MB/s) - written to stdout [12/12]
```

Beispiel

Beispiel

Ich will aber meine Webseiten unter /www ablegen! - Eigentlich kein Problem

```
[root@box ~]# cd /var/www/html
[root@box ~]# semanage fcontext  -a -t httpd_sys_content_t "/www(/.*)?"
[root@box ~]# restorecon -R -v /www/
restorecon reset /www context
unconfined_u:object_r:default_t:s0->unconfined_u:object_r:httpd_sys_content_t:s0
restorecon reset /www/index.html
context unconfined_u:object_r:default_t:s0-unconfined_u:object_r:httpd_sys_content_t:s0
```

Andere Ports als 80 & 443 sind auch kein Problem

```
[root@box ~]# semanage port -a -t http_port_t -p tcp 81
```

Bekannte Herausforderungen

Backup

- SELinux verwendet erweiterte Dateiattribute („xattr“)
- Filekontext wird automatisch gesetzt, wenn SELinux aktiv ist.
- Keine besonderen Anforderungen an die Backupsoftware. Es schadet aber auch nicht wenn die erweiterten Attribute mit gesichert werden.

Worauf man achten sollte

Nicht von der Policy gedeckte Standardsoftware

- Bei der Verwendung der „targeted“ Policy stellt nicht unterstützte Software kein Problem dar.
- Sind keine spezifischen Regeln vorhanden, wird die Anwendung im Kontext ‚unconfined‘ ausgeführt und SELinux damit transparent ist.
- Soll die entsprechende Anwendung unter Kontrolle von SELinux laufen, kann jederzeit ein entsprechendes Modul zur Policy hinzugefügt werden.

Fazit

Tldl - Zusammenfassung

- Seit RHEL4 Standard.
- Erhöhte Sicherheit durch MAC.
- „targeted“ Policy – isoliert ausgewählte Prozesse in eigenen Domänen.
- „unconfined“ Prozesse werden ohne SELinux Kontrolle ausgeführt.
- Abweichungen vom FHS erfordert Anpassungen der Policy oder der Filekontexte

- AKTIVIEREN!