Einführung in SELinux

Marcus Winkler

Chemnitz, den 19. November 2006

Inhaltsverzeichnis

1 Einleitung

Wenige Themen stehen für den Administrator eines komplexen Computersystems so sehr im Vordergrund wie die Sicherheit dieses Systems. Eine Schlüsselrolle nimmt dabei das Betriebssystem ein: es muss benötigte Ressourcen zur Verfügung stellen und gleichzeitig über deren Verwendung wachen.

Die schnelle Verbreitung des Internets hat zu einer fast vollständigen Vernetzung der Rechensysteme der Welt geführt. Es werden eine Vielzahl von Diensten angeboten und genutzt. Fast jede Software, die einen Dienst zur Verfügung stellt, muss früher oder später einem Angriff standhalten. Im schlimmsten Fall kann ein Angriff auf einen Dienst das ganze System gefährden, auf dem der Dienst angeboten wird.

Um das zu verhindern, haben bei der Entwicklung von Serversoftware einige Richtlinien breite Anerkennung gefunden: Es gilt das „principle of least privilege“, wonach ein Prozess immer nur diejenigen Zugriffsrechte haben sollte, die er für die Erfüllung seiner Aufgabe gerade benötigt. Eine weiterer Grundsatz ist die „privilege separation“, die Trennung einer Aufgabe und zugehöriger Zugriffsrechte in kleine Teilaufgaben.

Die Sicherheit von Software hängt damit ganz wesentlich vom Kenntnisstand und der Sorgfalt des Programmierers ab. Um dem Systemadministrator die Möglichkeit zu geben, potenziell unsichere Software stärker einzudämmen als vom Autor vorgesehen, reichen die herkömmlichen Hilfsmittel verbreiteter Betriebssysteme nicht aus.

Zudem stellen sie den Administrator vor Probleme, wenn einige Nutzer seiner Systeme mit zusätzlichen Privilegien ausgestattet werden sollen.

Mit SELinux steht eine Erweiterung von Linux zur Verfügung, die eben diese Defizite beheben soll. Aufbauend auf den Konzepten von Type Enforcement und Role Based Access Control ermöglicht SELinux eine fein abgestufte Vergabe von Rechten an Programme und Nutzer. Dieses Tutorial wird einen Einstieg in die Verwendung von SELinux geben und einige Einsatzszenarien am Beispiel vorführen.

2 Grundlagen

Am Beginn stehen die Definitionen einiger Konzepte, ohne deren Verständnis ein Einstieg in SELinux nicht möglich ist.

2.1 Mandatory Access Control

Erfüllt ein Betriebssystem folgende Kriterien, spricht man von Mandatory Access Control (MAC) [1]:

1. Die Vergabe von Rechten ist für praktisch alle Subjekte und Objekte eines Systems vorgesehen. Damit ist eine extrem genaue Rechteeinschränkung möglich.
2. Die vergebenen Rechte werden in einer Zentralen Richtlinie (*Policy*) festgelegt.
3. Die Policy ist nur durch den Systemadministrator konfigurierbar. Andere Nutzer des Systems haben keinen Einfluss auf Zugriffsentscheidungen.

SELinux erfüllt diese Kriterien mit Hilfe der Mechanismen, die im Folgenden vorgestellt werden. Im Gegensatz dazu bieten traditionelle Unix-Systeme diese Möglichkeiten nicht. Ihre Methoden der Zugriffskontrolle (z.B. Permission-Bits, Access Control Lists) zählen zur Discretionary Access Control (DAC).

Die Zugriffskontrolle durch SELinux greift erst nach den herkömmlichen Unix-Kontrollen. Sollte Unix einen Vorgang nicht gestatten, wird SELinux gar nicht erst konsultiert. SELinux kann demzufolge nichts erlauben, was Unix verbietet.

2.2 Type Enforcement

2.2.1 Domänen und Typen

Type Enforcement (TE) ist das grundlegende Modell, nach dem SELinux Zugriffsentscheidungen fällt. Es unterteilt das System in Subjekte und Objekte. Die Prozesse eines Systems sind Subjekte. Alle anderen Betriebsmittel, beispielsweise Dateien, sind Objekte.

Jedem Subjekt wird nun eine *Domain* (Domäne) zugeordnet, jedem Objekt ein *Type*. Wann immer ein Subjekt Zugriff auf ein Objekt haben will, muss seiner Domäne der Zugriff auf den Typ des Objektes gestattet sein.

Prozesse der gleichen Domäne werden gemäß Type Enforcement gleich behandelt. Das selbe gilt für Objekte von gleichem Typ. Domänen und Typen können demzufolge als Äquivalenzklassen betrachtet werden [4].

Der Administrator kann folgendes in der Policy festlegen:

Name	zugeordnet zu	Beschreibung
`init_t`	Prozess	Domäne des Init-Prozesses
`initrc_t`	Prozess	Domäne, in der Init-Skripte ablaufen
`ping_t`	Prozess	Domäne, in der Ping-Prozesse ablaufen
`ping_exec_t`	Dateien	Typ der ausführbaren Datei `/bin/ping`
`net_conf_t`	Dateien	Typ von u.a. `/etc/resolv.conf`
`smtp_port_t`	Netzwerk-Ports	Typ der TCP-Port 25, 465 und 587
`user_t`	Prozess	Standard-Domäne eines unprivilegierten Nutzers

Tabelle 1: Einige Domänen und Typen eines SELinux-Systems

1. Die Zugriffsrechte von Domänen auf andere Domänen
2. Die Zugriffsrechte von Domänen auf Typen
3. Den Wechsel von einer Domäne zu einer anderen (*Domain Transition*)
4. Den Wechsel von einem Type zu einem anderen (*Type Transition*)

Grundsätzlich gilt: Was nicht erlaubt ist, ist verboten. Damit muss jeder Zugriff einer Domäne auf eine Domäne/einen Typ explizit erlaubt werden.

Gemäß einer SELinux-Konvention werden Domänen und Typen mit `_t` am Ende des Typnamens gekennzeichnet. Einige Beispiele von Domänen eines typischen SELinux-Systems finden sich in Tabelle 1.

Die Implementierung von SELinux unterscheidet nicht zwischen Domäne und Typ. Beide werden in der Policy mit dem gleichen Schlüsselwort definiert. Jedoch ist zum Verständnis der Konfiguration und Dokumentation die Kenntnis des Unterschiedes notwendig.

2.2.2 Klassen

Um eine feinere Abstufung der Zugriffsregeln zu ermöglichen als dies nur mit Typen möglich wäre, führt SELinux zusätzlich Klassen (*security class*) ein. Die Klasse eines Objektes hängt nur von der Art des Objektes ab. Beispiele für Klassen sind `file`, `dir` und `socket`. Folgende Regel verdeutlicht den Zusammenhang von Typen und Klassen:

```
allow init_t etc_t:file { read write };
```

Als Teil der Policy erlaubt sie einem Prozess in der Domäne `init_t` auf ein Objekt der Klasse `file` mit dem Typ `etc_t` mit den Permissions `read` und `write` zuzugreifen.

2.2.3 Domain Transitions

Ein Prozess erhält seine Domäne durch eine sogenannte *Domain Transition*, in der Regel beim Erzeugen des Prozesses aus einer ausführbaren Datei. Alle auf einem System möglichen Domain Transitions müssen zentral festgelegt werden, um die Bedingungen von Mandatory Access Control zu erfüllen.

Schon beim Systemstart finden eine Vielzahl von Transitions statt. Die SELinux-Policy wird so früh wie möglich in den Betriebssystemkern geladen. Danach erhalten die Programme und Skripte bei ihrer Ausführung durch Init die in der Policy festgelegten Domains. Domain Transitions sind sowohl von privilegierten in weniger privilegierte Domains möglich, als auch umgekehrt.

Festgelegt werden Domain Transitions durch die Angabe einer Quelldomain, Zieldomain, sowie eines sogenannten Eintrittspunktes (*entrypoint*). Dabei handelt es sich um den Typ der ausführbaren Datei, aus der der Prozess gestartet wird. Folgendes Beispiel ist ein Auszug aus der Policy von SELinux:

```
type_transition user_t ping_exec_t:process ping_t;
```

Mit dieser Zeile wird eine Transition definiert, die es einem Prozess der Domäne `user_t` ermöglicht, das Programm Ping in der Domain `ping_t` auszuführen. Damit kann dem Programm Ping eine passend zugeschnittene Menge an Rechten zugewiesen werden.

Ist für ein Tripel aus Quelldomain, Zieldomain und Typ der Datei keine Transition definiert, wird bei der Ausführung der Datei die Domäne des Vaterprozesses vererbt.

Die Definition einer Domain Transition impliziert nicht deren Erlaubnis. Das muss extra geschehen.

2.2.4 Type Transitions

Dateien, die bei laufendem System erzeugt werden, erben den Typ des Verzeichnisses, in dem sie angelegt werden. Durch Festlegung einer *Type Transition* kann das übergangen werden. Die Definition einer Type Transition sieht beispielhaft so aus:

```
type_transition user_t user_home_dir_t:file user_home_t;
```

2.3 Role Based Access Control

Damit Type Enforcement sinnvoll genutzt werden kann, müssen die Nutzer eines Computersystems für verschiedene Domänen autorisiert werden. Da eine direkte Zuordnung von Nutzern zu Domains meist zu aufwändig wäre, nutzt SELinux den Mechanismus der *Role Based Access Control* (RBAC).

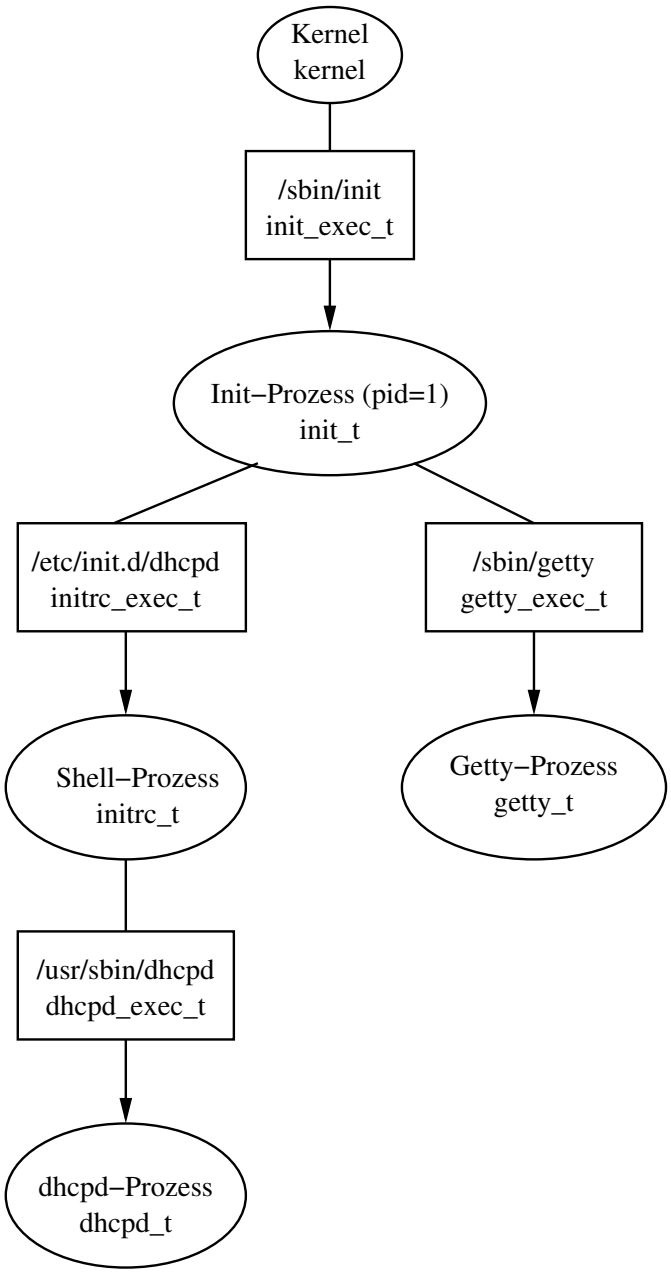

Abbildung 1: Einige Domain Transitions beim Systemstart

In RBAC-basierten Systemen ist jeder Nutzer typischerweise für eine oder mehrere Rollen autorisiert. Damit ist es möglich, Klassen von Nutzern zu definieren, die identische Privilegien auf dem System erhalten.

Role Based Access Control kann die Vergabe von erhöhten Privilegien so erheblich vereinfachen. Eine Rolle wird vom Administrator in der Policy definiert und mit den entsprechenden Rechten ausgestattet.

Das geschieht in SELinux, indem sie für eine Menge von Domains autorisiert wird. Es erfolgt so die Abbildung auf Type Enforcement als Mechanismus zur Rechtevergabe. Zusätzlich wird festgelegt, welche Nutzer in diese Rolle wechseln können.

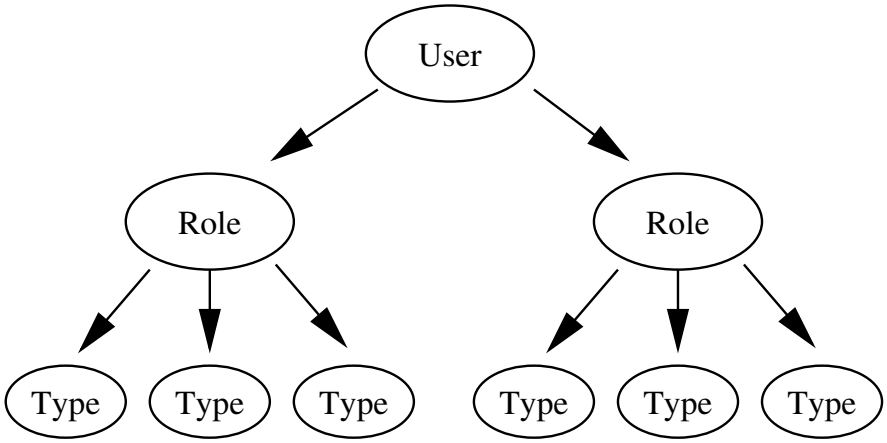

Abbildung 2: Zuordnung von Nutzern, Rollen und Typen/Domänen

2.4 Identität

Das Konzept der Nutzeridentität unterscheidet sich in SELinux stark von der traditionellen Verwendung unter Unix. Für das Verweigern oder Gewähren eines Zugriffs ist die Identität des Nutzers unter SELinux nicht relevant. Sie legt einzig und allein fest, für welche Rollen der Nutzer autorisiert ist und damit in welche Domänen die von ihm gestarteten Prozesse wechseln dürfen.

SELinux führt keine Authentifizierung von Nutzern durch. Es verlässt sich dabei vollständig auf das Linux-System. Hat sich ein Nutzer erfolgreich authentifiziert, wird ihn SELinux anhand dessen Nutzerkürzel für die gleich lautende SELinux-Identität autorisieren.

Zu beachten ist, dass die Definitionen der SELinux-Identitäten Teil der Policy sind, also nicht dynamisch, beispielsweise mittels LDAP, ermittelt werden können. Kennt SELinux einen durch Linux authentifizierten Nutzer nicht, wird ihm eine Standard-Identität zugewiesen.

2.5 Security Context

Type Enforcement und Role Based Access Control machen es nötig, dass jedem auf einem SELinux-System laufenden Prozess eine Identität, eine Rolle und eine Domain zugeordnet sind. Dieses Tripel wird als Security Context bezeichnet. Er wird üblicherweise in folgender Form angegeben:

```
user:role:domain
```

Andere Objekte auf dem System besitzen ebenfalls einen Security Context. Bei ihnen sind Nutzeridentität und Rolle zwar bedeutungslos, werden aber dennoch mit angegeben und gespeichert. Die Angabe erfolgt auch in dieser Form:

```
user:role:type
```

2.6 Labels

Mandatory Access Control verlangt, dass alle Aspekte der Policy zentral festgelegt sind. Dazu gehört auch, dass der Security Context der Dateien des Systems in der Policy definiert ist. Dennoch wird der Kontext jeder Datei zusätzlich im Dateisystem in sogenannten *Labels* abgelegt.

Für Zugriffsentscheidungen im laufenden System sind nur die Labels relevant. Die Festlegungen der Policy werden beispielsweise nach der Installation einer anderen Policy beim Systemstart in die Labels der Dateien übertragen. Dieser Vorgang nennt sich *Relabeling* (Neuetikettierung).

Labels werden in den *extended attributes* einer Datei gespeichert, was nicht von jedem Dateisystem unterstützt wird. Es existiert daher eine Möglichkeit, in der Policy einen Kontext für die Verzeichnisbäume nicht unterstützter Dateisysteme anzugeben. Eine fein granulierte Rechtevergabe ist auf diesen Dateisystemen nicht möglich.

2.7 Policy

Die Policy ist von zentraler Bedeutung in SELinux. Sie zu warten und zu konfigurieren ist die Aufgabe des Systemadministrators. Eine Policy von Grund auf neu zu erstellen wäre ein Unternehmen, das sehr viel Zeit verschlingen würde. Aus diesem Grund liegt der NSA-Distribution von SELinux eine Policy bei, die als Vorlage für eigene Anpassungen dienen kann.

Für Nutzer von Fedora Core Linux und dessen Derivate bietet es sich allerdings an, eine der beiden von Fedora bereitgestellten Policys zu verwenden. Diese sind an die Besonderheiten der Distribution weitgehend angepasst und werden von deren Autoren gepflegt. Somit wird die Last der Konfiguration zwischen Systemadministrator und Distributor aufgeteilt.

Die beiden Policys von Fedora Core werden auch die Grundlage des weiteren Tutorials sein.

2.8 targeted-Policy

Um den Einstieg in SELinux so leicht wie möglich zu machen und eine Integration in die Distribution voranzutreiben, hat das Fedora-Projekt die targeted-Policy entwickelt. Ist SELinux mit targeted-Policy aktiv, wird das von den Nutzern des Systems kaum bemerkt. Auch alle Systemdienste arbeiten weiter wie bisher. Sie ist daher ideal, um erste Experimente mit SELinux durchzuführen.

Die targeted-Policy ist sowohl in Zielsetzung als auch Umfang eng begrenzt worden. Sie beschränkt sich darauf, einige Systemdienste abzusichern. Sie legt für diese Dienste minimale Zugriffsrechte fest und setzt sie durch. Der Rest des Systems, insbesondere die Nutzer, unterliegt keinerlei zusätzlichen Einschränkungen durch SELinux. Zu diesem Zweck ist die Domain `unconfined_t` definiert, für die alle Prozesse (bis auf die oben genannten Systemdienste) autorisiert sind. Nutzerrollen haben kaum eine Bedeutung.

Gemessen an den Möglichkeiten, die SELinux bietet, ist der Zuwachs an Schutz durch die targeted-Policy relativ gering. Dennoch kann ihr Einsatz sehr sinnvoll sein. Immer dann, wenn keine komplexen Nutzerszenarien mit Hilfe von SELinux umzusetzen sind, sondern eine „privilege escalation“ von Serverdiensten zu verhindern ist, ist sie sehr hilfreich.

2.9 strict-Policy

Die strict-Policy kommt den Vorstellungen der SELinux-Entwickler am nächsten. Sie unterstützt mehrere Nutzerrollen und enthält Regeln für beinahe jedes Programm des Systems. Sie kann nicht nur Serverdienste, sondern auch Anwendungsprogramme an nicht autorisierten Zugriffen hindern. Damit wird der Schutz des Systems vor fehlerhafter Software maximiert.

Für die Umsetzung von fein abgestuften Nutzerrechten mittels Role Based Access Control kann die strict-Policy als Grundlage dienen.

Von Nachteil ist der höhere Aufwand, ein System zur strict-Policy kompatibel zu machen und über Updates hinweg kompatibel zu halten. Zudem wird diese Policy von Red Hat Enterprise Linux offiziell nicht unterstützt.

3 Absichern eines Serverdienstes mit der targeted-Policy

3.1 Zielstellung

Installation und Inbetriebnahme der targeted-Policy sind einfach, unkompliziert und sehr gut dokumentiert, etwa in [3]. Sie enthält allerdings nur für wenige Dienste des Systems Einschränkungen der Zugriffsrechte. Es soll deshalb gezeigt werden, wie man einen Regelsatz für einen beliebigen Serverdienst erstellen und in die Policy einfügen kann. Das Ziel besteht darin, den Dienst nur mit denjenigen Zugriffsrechten auf Objekte des Systems auszustatten, die er für die Erfüllung seiner Aufgabe benötigt.

Die einzelnen Schritte werden am Beispiel des Trivial FTP Daemon (tftpd) vorgeführt. Das TFTP-Protokoll wird genutzt, um Computersysteme über eine Netzwerkverbindung ohne Zugriffe auf die Festplatte des Systems zu starten.

Der tftpd-Serverdienst eignet sich gut als Anschauungsbeispiel, da er relativ wenige Rechte benötigt und nicht in komplizierter Art und Weise mit dem Rest des Serversystems interagiert. Die gezeigte Vorgehensweise ist prinzipiell auf alle Serverdienste anwendbar.

3.2 Installation von SELinux

Zur Erstellung dieses Tutorials wurde Scientific Linux 4.4 verwendet. Alle angegebenen Befehle und Paketnamen beziehen sich darauf. Da Scientific Linux auf der Fedora Core Distribution beruht, sollten aber alle Schritte auf anderen von Fedora Core abgeleiteten Distributionen nachvollziehbar sein.

Desweiteren bieten nur wenige Dateisysteme die nötigen Voraussetzungen für einen sinnvollen Betrieb von SELinux. Das verwendete System sollte entweder ext2 oder ext3 als Dateisystem für die Root-Partition und alle Systempartitionen benutzen. Wird für Home-Verzeichnisse der Nutzer beispielsweise ein Netzwerkdateisystem verwendet, werden die daraus resultierenden Einschränkungen an entsprechender Stelle erwähnt.

Voraussetzung für den Betrieb von SELinux ist ausserdem die Verwendung eines geeigneten Kernels. Es empfiehlt sich dringend Kernelversion 2.6, da die Unterstützung von SELinux darin bereits enthalten ist. Ausserdem muss SELinux mit einkompiliert sein, was beim standardmäßig von Scientific Linux installierten Kernel der Fall ist.

Bereits bei der Installation von Scientific Linux 4.4 können die SELinux-Pakete mit installiert werden. Ist das nicht geschehen, müssen folgende Pakete noch installiert werden:

```
libselinux-1.19.1-7.2.i386.rpm
libsepol-1.1.1-2.i386.rpm
selinux-doc-1.14.1-1.noarch.rpm
```

```
selinux-policy-targeted-1.17.30-2.140.noarch.rpm
selinux-policy-targeted-sources-1.17.30-2.140.noarch.rpm
setools-1.5.1-5.i386.rpm
setools-gui-1.5.1-5.i386.rpm
checkpolicy-1.17.5-1.i386.rpm
```

Um das Beispiel tftpd nachzuvollziehen, sollten folgende Pakete ebenfalls installiert werden:

```
tftp-0.39-1.i386.rpm
tftp-server-0.39-1.i386.rpm
```

Alle diese Pakete gehören zum Lieferumfang von Scientific Linux 4.4.

Jetzt sollte die targeted-Policy als aktive Policy eingestellt sein. Zur Überprüfung öffnen sie die Datei `/etc/selinux/config` und stellen sie sicher, dass sie folgende Zeile enthält:

```
SELINUXTYPE=targeted
```

Ändern sie ausserdem die Zeile

```
SELINUX=enforcing
```

in

```
SELINUX=permissive
```

SELinux unterstützt den sogenannten *Permissive Mode*. Dabei werden unerlaubte Zugriffe nicht blockiert, sondern stattdessen ein detaillierter Eintrag im Logfile des Systems erzeugt. Es bietet sich an, umfangreiche Änderungen an der Policy im Permissive Mode vorzunehmen und zu testen.

In der Policy ist für jede Datei des Systems ein Kontext vorgesehen. Diese Kontexte müssen jetzt in die Labels der Dateien übertragen werden. Dazu wird mit folgenden Befehlen ein sogenanntes Relabeling ausgeführt:

```
touch /.autorelabel
reboot
```

Der Neustart kann deutlich länger dauern als gewöhnlich.

Nach der Anmeldung als root überprüfen sie durch folgendes Kommando, ob SELinux aktiv ist und sich im Permissive Mode befindet:

```
sestatus
```

3.3 Erste Schritte unter SELinux

Die Installation und Aktivierung von SELinux hat einige Veränderungen am System bewirkt. Prozessen und Dateien sind jetzt jeweils Security-Kontexte zugeordnet. Um diese Kontexte sichtbar zu machen, wurden einige Hilfsprogramme angepasst.

Lassen sie sich die Kontexte von Prozessen und Dateien anzeigen:

```
ps aux -Z
ls -Z
ls --scontext
```

Es fällt auf, dass die meisten Prozesse vom Typ `unconfined_t` sind. Das heisst, sie werden nicht durch SELinux eingeschränkt.

Von besonderem Interesse ist der Kontext der eigenen Shell. Er spiegelt wieder, für welche Identität, Rolle und Typ der angemeldete Nutzer autorisiert ist. Vom Nutzer gestartete Programme werden ebenfalls mit diesem Kontext versehen, sofern keine Domain Transition für das Programm definiert ist. Der eigene Kontext kann auch folgendermaßen ermittelt werden:

```
id
```

Das bereits erwähnte Programm `sestatus` zeigt neben den Grundeigenschaften der aktiven Policy auch eine Menge von *Booleans* an. Dabei handelt es sich um binäre Parameter, die im laufenden Betrieb eine Änderung im Verhalten von SELinux bewirken können. Ihre Definition ist Teil der Policy.

3.4 Einfügen eines Serverdienstes in die Policy

Mit dem Paket `selinux-policy-targeted-sources-1.17.30-2.140.noarch.rpm` ist der Quelltext der targeted-Policy auf dem System installiert worden. Er findet sich vollständig unter `/etc/selinux/targeted/src/policy/` wieder.

Ein Großteil der Policy wird in der Makro-Sprache M4 konfiguriert. Man kann jedoch auch ohne Kenntnis dieser Sprache anhand der bereits vorhandenen Menge an Regeln leicht Änderungen und Ergänzungen vornehmen.

Das Erstellen eines Regelsatzes geschieht in folgenden Schritten:

1. Festlegung einer Domäne für den Serverprozess des Dienstes
2. Festlegung von Typen für die Dateien, die zum Serverdienst gehören
3. Festlegung der Domain Transitions, mit deren Hilfe der Prozess des Dienstes in seine zugwiesene Domäne wechselt
4. Zuordnung der oben definierten Typen zu Dateien/Verzeichnissen

5. Autorisation der Prozessdomäne für notwändige Zugriffe
6. ausgiebiges Testen des Dienstes
7. Auswertung der Log-Datei und gegebenenfalls Autorisation für weitere Zugriffe

3.4.1 Domäne

Im einfachsten Fall wird nur eine einzige Domäne für den Prozess des einzuschränkenden Dienstes benötigt. Deren Definition erfolgt in einer Datei im Unterverzeichnis `domains/program/` des Policy-Quelltextes.

Um immer wiederkehrende Aspekte der Konfiguration zu erleichtern, stellt die SELinux-Policy eine Reihe von Makros zur Verfügung. Man sollte diese Makros einem kompletten Neuschreiben der Typdefinition vorziehen, um die Policy übersichtlich zu halten.

Erstellen Sie die Datei `domains/program/tftpd.te` und fügen Sie folgende Zeile ein:

```
daemon_domain(tftpd)
```

Das Makro `daemon_domain` ist sehr umfangreich. Es erzeugt eine Domäne `tftpd_t` und stattet sie bereits mit grundlegenden Zugriffsrechten aus.

Der Quelltext von `daemon_domain` befindet sich in `macros/global_macros.te`.

3.4.2 Typen

Nachdem die Prozessdomäne definiert ist, müssen jetzt die Dateitypen festgelegt werden. Dabei ist zu überlegen, welche Dateien in welchen Typen zusammengefasst werden können. Dateien gleichen Typs werden gleich behandelt. Soll die Domäne des Prozesses auf unterschiedliche Dateien auch unterschiedliche Zugriffsrechte bekommen, müssen diesen Dateien entsprechend verschiedene Typen zugewiesen werden.

Der tftpd-Dienst arbeitet normalerweise im Verzeichnis `/tftpboot/`. Er erlaubt Klienten, die dort abgelegten Dateien über das TFTP-Protokoll zu empfangen, beziehungsweise Dateien dort zu überschreiben. Es ist daher ein Typ für dieses Verzeichnis und enthaltene Dateien anzulegen.

Hängen sie dazu folgende Zeile an die Datei domains/program/tftpd.te an:

```
type tftpdir_t, file_type, sysadmfile, root_dir_type;
```

Die Schlüsselwörter `file_type`, `sysadmfile` und `root_dir_type` sind sogenannte Attribute (*attribute*). Durch die Verwendung von Attributen lassen sich in SELinux

Regeln erstellen, die auf viele Typen gleichzeitig wirken. Die Definition der Attribute mit Kommentaren zu deren Bedeutung findet sich in der Datei `attrib.te` im Quelltext-Verzeichnis der Policy.

Bedeutsam sind die Attribute ausserdem für die Überprüfung der Konsistenz der Policy, sowie die Durchsetzung einiger Regeln gegenüber dem Policy-Autor. So wird beispielsweise sichergestellt, dass nur mit dem Attribut `domain` versehene Typen einem Prozess als Domäne zugewiesen werden können.

Diese betreffenden Regeln sind in in `assert.te` definiert. Sie werden beim Übersetzen der Policy ausgewertet.

3.4.3 Domain Transition

Die Domain Transition sorgt dafür, dass der Prozess in der vorgesehenen Domäne abläuft. Dazu muss zunächst ein Typ als Eintrittspunkt (*entrypoint*) definiert werden.

Die Definition des notwendigen Typs wird im Beispiel durch das Makro `daemon_domain` vorgenommen. Gemäß der SELinux-Konvention wurde der Typ des Eintrittspunktes `tftpd_exec_t` benannt.

Ausserdem wird `daemon_domain` folgende Makros aufrufen:

```
domain_auto_trans(initrc_t, tftpd_exec_t, tftpd_t)
domain_auto_trans(sysadm_t, tftpd_exec_t, tftpd_t)
```

Damit werden die eigentlichen Transitions definiert. Der Übergang zur Zieldomäne `tftpd_t` wird für die Quelldomänen `initrc_t` und `sysadm_t` vorgeschrieben. Das ist für das Starten des Dienstes durch ein Init-Skript ausreichend, da Init-Skripte in der Domäne `initrc_t` ablaufen.

Bemerkenswert ist, dass keine Domain Transition von `unconfined_t` nach `tftpd_t` definiert ist. Ruft der Administrator die ausführbare Datei des Dienstes direkt von der Konsole auf, wird kein Domänenwechsel nach `tftpd_t` stattfinden. Um trotzdem einen Test von der Kommandozeile aus zu ermöglichen, muss daher folgendes minimale Skript unter `/etc/init.d/tftpd` gespeichert werden:

```
#!/bin/sh
case "$1" in
start)
    /usr/sbin/in.tftpd -l -u nobody
    ;;
stop)
    killall in.tftpd
    ;;
*)
```

```
        echo "Es werden nur die Parameter start und stop unterstuetzt"
        exit 1
esac
```

Das Skript ist noch als ausführbar zu kennzeichnen:

```
chmod 755 /etc/init.d/tftpd
```

3.4.4 Dateikontexte

Die Kontexte fast aller Dateien des Systems werden im Verzeichnis `file_contexts` festgelegt. Beim Relabeling des Systems werden die Labels der Dateien aus den dortigen Angaben wiederhergestellt. Ausgenommen davon sind Dateisysteme, die keine Labels unterstützen.

Um den oben definierten Typen reale Dateien des Systems zuzuordnen, muss ein Eintrag in die Datei `file_contexts/program/tftpd.fc` erfolgen. Diese Datei ist in der targeted-Policy bereits vorhanden, allerdings sind die relevanten Einträge auskommentiert.

Entfernen sie die doppelten Bindestriche aus allen Zeilen der Datei.

Als Pfadangaben sind reguläre Ausdrücke zulässig, was die Behandlung einer großen Menge von Dateien stark vereinfacht. Jedem Eintrag muss statt eines Typen ein vollständiger Kontext zugeordnet werden. Dabei kann immer `system_u` als Nutzer und `object_r` als Rolle verwendet werden. Für die Bestimmung der Zugriffsrechte spielen diese beiden Angaben keine Rolle.

Zugriffsrechte Um die Konfiguration so weit wie möglich zu vereinfachen, finden auch hier vorgefertigte Makros Verwendung. Ergänzen Sie folgende Zeilen in `domains/program/tftpd.te`:

```
can_network(tftpd_t)
can_ypbind(tftpd_t)
rw_dir_file(tftpd_t, tftpdir_t)
type tftp_port_t, port_type, reserved_port_type;
```

Bei der letzten Zeile handelt es sich um die Definition eines Typs für den TCP-Port, auf dem der Dienst auf Verbindungen wartet.

3.4.5 Test

Beenden sie jetzt alle laufenden Instanzen von tftpd:

```
/etc/init.d/tftpd stop
```

Nun ist die geänderte Policy zu übersetzen und zu laden. Führen sie den folgenden Befehl im Verzeichnis `/etc/selinux/targeted/src/policy/` aus:

```
make load
```

Damit ist die geänderte Policy aktiv.

Um die geänderten Kontexte der Dateien in die Labels zu übertragen ist dieser Befehl notwendig:

```
make relabel
```

Ein Relabeling sollte immer beim Neustart stattfinden, da sonst Inkonsistenzen bezüglich laufender Programme auftreten könnten. In diesem Fall aber sind nur wenige, bekannte Labels geändert worden und das dazugehörige Programm wurde beendet.

Um festzustellen, welche Rechte dem Dienst nun genau erteilt wurden, bietet sich die Verwendung des Programms `sesearch` an. Es kann alle zu einem Typ gehörigen Regeln in der Policy finden und anzeigen. Sesearch wird auf die aktive Policy angewendet. Es bietet verschiedene Möglichkeiten:

Alle Regeln, die der Domäne `tftpd_t` Zugriffe auf andere Typen gestatten:

```
sesearch --allow --source tftpd_t
```

Alle Regeln, die Zugriff auf den Typ `tftpdir_t` erlauben:

```
sesearch --allow --target tftpdir_t
sesearch --allow --target tftpd_exec_t
```

Da in der aktiven Policy alle Makros aufgelöst sind, gibt `sesearch` eine teilweise große Mengen an Regeln aus.

Legen Sie eine Datei unter `/tftpboot` an und überzeugen sie sich, dass alle Dateien einen korrekten Kontext erhalten haben:

```
ls -Zd /tftpboot

echo "Hallo Welt" >/tftpboot/hallowelt
ls -Z /tftpboot

ls -Z /usr/sbin/in.tftpd
```

Der tftpd-Dienst kann jetzt gestartet werden.

```
/etc/init.d/tftpd start
```

Da sich das System im Permissive Mode befindet, sind dabei keine Probleme zu erwarten. Sehen sie nach, ob sich der tftpd-Prozess tatsächlich in der Domäne `tftpd_t` befindet:

```
ps aux -Z
```

Nun ist der Dienst ausgiebig zu testen. Alle Funktionen sollten ausprobiert werden. Es bietet sich die gleichzeitige Beobachtung von `/var/log/messages` an.

3.4.6 Auswertung der Log-Nachrichten

Gleich nach dem Start von tftpd finden sich eine Menge von Einträgen in der Log-Datei wieder, hier nur ein Beispiel:

```
avc:  denied  { read } for  pid=6187 comm="in.tftpd"
      name="nsswitch.conf" dev=hda2 ino=32752
      scontext=root:system_r:tftpd_t
      tcontext=system_u:object_r:etc_t tclass=file
```

Das Verständnis dieser Einträge ist dabei recht intuitiv möglich. Das Wort `denied` stellt unmissverständlich dar, dass ein Zugriff verweigert wurde. Bei dem mit `scontext` bezeichneten Kontext handelt es sich um den Kontext des zugreifenden Prozesses. Demzufolge ist `tcontext` der Kontext des zugegriffenen Objektes. Die weiteren Angaben lassen auf die genauen Umstände des verhinderten Zugriffs schließen.

Im Permissive Mode wird ine SELinux-Log-Nachricht nur einmal in die Log-Datei ausgegeben. Bei wiederholten, identischen Zugriffen wird kein neuer Log-Eintrag erzeugt.

Eine große Hilfestellung bietet das Skript `audit2allow`, welches aus Log-Einträgen automatisch Regeln generieren kann. Diese Regeln erlauben die entsprechenden Zugriffe, und müssen nur noch in die Policy einfügt werden.

Kopieren sie die relevanten Einträge aus `/var/log/messages` in eine Datei namens `~/avc-log` und geben sie folgendes Kommando ein:

```
audit2allow <~/avc-log
```

Die ausgegebenen Zeilen können sie nun an `domains/program/tftpd.te` anhängen. Nach erneuter Ausführung `make reload` ist der tftpd-Dienst fertig konfiguriert.

3.4.7 Abschluss der Konfiguration

Damit kann der Permissive Mode verlassen werden, die Erzwingung der SELinux-Policy wird mit folgendem Befehl sofort wirksam:

```
setenforce 1
```

Diese Einstellung bleibt bis zum nächsten Neustart erhalten. Soll der Enforcing Mode dauerhaft aktiviert werden, ist eine Anpassung von `/etc/selinux/config` notwendig.

Durch Ausführung von `sestatus` kann man sich davon überzeugen, dass ein zusätzlicher boolean-Parameter `tftpd_disable_trans` in der Policy angelegt wurde. Das ist automatisch durch das Makro `daemon_domain` geschehen. Mit Hilfe dieses Parameters lässt sich die Domain Transition zu `tftpd_t` einfach abschalten. Es ist damit möglich, alle vorgenommenen Einschränkungen ausser Kraft zu setzen. Sollten Probleme mit dem Regelsatz des tftpd-Dienstes auftreten, lässt sich damit ein Wechsel des gesamten System in den Permissive Mode vermeiden.

```
setsebool tftpd_disable_trans 1
/etc/init.d/tftpd stop
/etc/init.d/tftpd start
```

Ob spätere Nachbesserungen an der Policy nötig sind, hängt entscheidend vom Umfang der Tests und der Kenntnis des Policy-Autors ab. Idealerweise würden die SELinux-Regeln einer Anwendung vom Autor der Software selbst geschrieben, da er die Architektur und Anforderungen seines Programmes am besten kennt. Leider hat SELinux noch nicht die dafür nötige Verbreitung gefunden.

3.5 Verteilung der Policy auf andere Systeme

Damit die geänderte Policy auf anderen Systemen genutzt werden kann, muss nicht der gesamte Quellcode übertragen werden. Es genügt, folgende Verzeichnisse und Dateien zu kopieren:

```
/etc/selinux/targeted/policy
/etc/selinux/targeted/contexts
/etc/selinux/targeted/booleans
/etc/selinux/config
```

Ausserdem ist darauf zu achten, dass der Enforcing Mode in `/etc/selinux/config` eingestellt ist.

Selbstverständlich sind auch die SELinux-Pakete zu installieren. Auf den Quelltext der Policy kann dabei ebenfalls verzichtet werden.

Nach dem Übertragen der Policy auf das Zielsystem muss dort unbedingt ein Relabeling des Dateisystems erfolgen. Das geschieht am zuverlässigsten bei einem Neustart:

```
touch /.autorelabel
reboot
```

4 Privilegierte Nutzerrollen in der strict-Policy

4.1 Zielstellung

SELinux ermöglicht es, die simple Einteilung in unprivilegierte und voll privilegierte Nutzer zu überwinden. Dank Role Based Access Control können Rollen mit abgestuften Zugriffsrechten erstellt werden. Das Benutzerkonto `root` verliert damit an Bedeutung.

Es soll gezeigt werden, wie eine Rolle in der strict-Policy grundsätzlich eingerichtet werden kann und wie ihr Zugriffsrechte eingeräumt werden, die über die Rechte eines normalen Nutzers hinausgehen. Diese Rolle kann dazu dienen, einem Mitarbeiter die Kontrolle über ausgesuchte Systemdienste zu gewähren. Zu diesem Zweck muss er den Dienst konfigurieren können und in der Lage sein, ihn zu starten und zu beenden.

4.2 Installation strict-Policy

Auf Grund der allumfassenden Natur der strict-Policy ist die Installation alles andere als einfach. Grundsätzlich ist immer davon auszugehen, dass zur Inbetriebnahme umfangreiche Anpassungen an der Policy nötig sind. Die vollständige erfolgreiche Einrichtung und Konfiguration der Policy setzt ein komplettes Verständnis von SELinux, des Aufbaus der strict-Policy und von Linux im allgemeinen voraus. Sie ist zudem stark von den konkreten Einsatzbedingungen abhängig. Zudem finden noch immer zahlreiche Änderungen an SELinux durch die Entwickler statt, was gelegentlich eine Anpassung der eigenen Modifikationen nötig macht.

Die folgenden Anleitungsschritte setzen daher voraus, dass eine minimale Einrichtung an das Zielsystem erfolgt ist und grundlegende Betriebssystemfunktionen zur Verfügung stehen.

Die strict-Policy ist leider nicht Teil von Red Hed Enterprise Linux und damit auch nicht von Scientific Linux. Es kann jedoch auf die Pakete von Fedora Core 3 zurück gegriffen werden.

Zusätzlich zu den in Abschnitt 3.2 aufgeführten Paketen sind folgende Pakete zu installieren:

```
selinux-policy-strict-1.17.30-2
selinux-policy-strict-sources-1.17.30-2
```

In `/etc/selinux/config` ist folgendes einzustellen:

```
SELINUXTYPE=strict
SELINUX=Permissive
```

Es ist unbedint erforderlich, während der Konfiguration der Poliy im Permissive Mode zu arbeiten, da man sich sonst zu leicht den Zugang zum System selbst verbietet.

Die Installation einer neuen Policy macht immer ein Relabeling des Dateisystems und einen Neustart erforderlich:

```
touch /.autorelabel
reboot
```

4.3 Details der strict-Policy

Die targeted-Policy beschränkt sich darauf, die Rechte ausgesuchter Serverdienste zu begrenzen. Die strict-Policy hingegen findet auf das gesamte System Anwendung. Sie definiert angepasste Domänen auch für einige ausgesuchte Anwendungsprogramme. Deren Vertrauenswürigkeit hat damit wesentlich weniger Einfluss auf die Systemsicherheit. Der Regelsatzes für eine solche Anwendung wird ähnlich erstellt wie der Regelsatz für einen Dienst der targeted-Policy. Das Beispiel des TFTP-Dienstes kann dazu als grobe Anleitung dienen.

Ist für eine Anwendung keine spezielle Domäne angelegt, läuft sie in der Standard-Domäne der jeweiligen Nutzerrolle ab. Die strict-Policy wird mit folgenden vier Rollen ausgeliefert:

- `user_r` Eine unprivilegierte Rolle, die keinen Wechsel in andere Rollen erlaubt.
- `staff_r` Diese Rolle darf in andere Rollen wechseln, sonst praktisch keine zusätzlichen Rechte gegenüber `user_r`.
- `sysadm_r` Administratorolle, die Zugriff auf fast alle Systemfunktionen erlaubt und wenigen Einschränkungen unterliegt.
- `system_r` Die Rolle der Init- und Systemdienste.

Grundsätzlich werden Nutzern entweder `user_r` oder `staff_r` zugewiesen. Es ist nicht sinnvoll, sie für beide Rollen zu autorisieren. Ein Administratorkonto jedoch verlangt eine Kombination aus `staff_r` und `sysadm_r`.

Eine unveränderte strict-Policy kennt nur zwei Nutzer. Meldet sich `root` am System an, wird ihm auch die SELinux-Identität `root` zugewiesen. Alle anderen Nutzer werden auf `user_u` abgebildet, einen Nutzer der Rolle `user_r`.

4.4 Erstellung einer privilegierten Rolle

Zur Demonstration wird eine Rolle angelegt, die ihren Mitgliedern grundlegende Zugriffe auf den MySQL-Dienst erlaubt, ohne ihnen vollen Zugriff auf das restliche System zu gewähren. Der Vorgang kann in mehrere Schritte gegliedert werden:

1. Erstellen der Rolle und Autorisierung der Nutzer für diese Rolle
2. Ausführung des Init-Skriptes des Dienstes gestatten

3. Zugriff auf Konfigurationsdateien des Dienstes erlauben
4. Entzug der Administrator-Privilegien des root-Nutzers

4.4.1 Rolle erstellen

Melden sie sich als `root` am System an. Der Aufruf von `id -Z` verrät den aktiven Kontext:

```
uid=0(root) gid=0(root) {...} context=root:sysadm_r:sysadm_t
```

Alle folgenden Pfadangaben beziehen sich auf den Quelltext der Policy unter

`/etc/selinux/strict/src/policy/`.

Der Name der neuen Rolle soll zweckmäßigerweise `dbadmin_r` sein, der Beispielnutzer heisst `blau`. Die Datei `users` ist um

```
user blau roles { staff_r dbadmin_r }
```

zu ergänzen.

Da das Starten von Systemdiensten wie MySQL einen Rollenwechsel nach `system_r` erfordert, muss die notwendige Autorisation in die Datei `rbac` eingetragen werden:

```
allow dbadmin_r system_r;
```

Das Erstellen einer Rolle wird wieder durch einige Makros erleichtert. Hängen sie diese Zeilen an domains/user.te an:

```
full_user_role(dbadmin)
priv_user(dbadmin)

role_tty_type_change(staff, dbadmin)
role_tty_type_change(dbadmin, staff)
```

Nach diesen Schritten ist der Kontext `blau:dbadmin_r:dbadmin_t` gültig. Es ist eine Domäne `dbadmin_t` erzeugt worden, die die Standard-Domäne der Rolle darstellt. Jede Rolle besitzt eine solche Standard-Domäne.

SELinux wird den Nutzer `blau` nicht erzeugen, sofern er nicht auch Unix-Nutzer ist:

```
useradd blau
passwd blau
```

Die Policy kann nun zum ersten Mal neu übersetzt werden.

```
make reload
```

Ist der Vorgang erfolgreich, sollte ein Login von `blau` und ein anschließender Aufruf von `id` erfolgen.

```
uid=500(blau) gid=500(blau) { ... } context=blau:staff_r:staff_t
```

Ein Wechsel in die Rolle `dbadmin_r` ist bereits möglich. Dazu dient das Programm `newrole`, welches eine nochmalige Eingabe des Passwortes von `blau` erfordert:

```
newrole -r dbadmin_r
```

Die auftretenden Fehlermeldungen im der Log-Datei rühren daher, dass diese Rolle bisher keine Zugriffsrechte auf ein Homeverzeichnis von `staff_r` erhalten hat.

Sie können den Benutzer `blau` jetzt wieder vom System abmelden.

4.4.2 Ausführung von Init-Skripten gestatten

Die Init-Skripte unter `/etc/init.d` laufen im Kontext `system_r:initrc_t` ab. Nur in diesem Kontext besitzen sie die für ihre Aufgabe nötigen Zugriffsrechte. Werden sie aus einem Kontext `sysadm_r:sysadm_t` heraus aufgerufen, erfolgt automatisch ein Wechsel in die korrekte Rolle und Domäne. Dieser Wechsel ließe sich auch für den Kontext `dbadmin_r:dbadmin_t` ermöglichen. Allerdings könnte dbadmin dann alle Init-Skripte ausführen, denn die haben alle den gleichen Typ `initrc_exec_t`.

Es bestehen zwei Möglichkeiten:

1. Änderung des Typs des mysqld-Skriptes. Es sind über die Policy verstreut Anpassungen nötig, um die Funktion des Skriptes für alle bisherigen Fälle zu gewährleisten.
2. Der Rolle `dbadmin_r` wird die Ausführung einer Kopie des Skriptes erlaubt, die einen passenden Typ erhält.

Im folgenden wird Variante 2 beschrieben.

Rollenwechsel sind selten und werden nur wenigen Programmen erlaubt. Darunter sind z.B. `newrole` und `login`. Damit ein Wechsel nach `system_r` gelingt, kommt das Hilfsprogramm `run_init` zum Einsatz. Es wurde speziell zu diesem Zweck geschaffen und ist Teil von SELinux. Auch hier ist es wieder nötig, eine Kopie zu erstellen und diese mit dem richtigen Typ zu versehen. Dieser Typ darf Ausführung und Domain Transition nur durch `dbadmin_t` erlauben.

```
mkdir /usr/dbadmin
cp /usr/sbin/run_init /usr/dbadmin
cp /etc/init.d/mysqld /usr/dbadmin/run_dbadmin_init
```

Erstellen sie die Datei `dbadmin.te` unter `domains/misc/` mit folgendem Inhalt:

```
type dbadmin_initrc_exec_t, file_type, sysadmfile;
run_program(dbadmin_t, dbadmin_r, dbadmin_init,
            dbadmin_initrc_exec_t, initrc_t)
```

Domäne/Typ	Bedeutung
`dbadmin_t`	Nutzerdomäne der dbadmin-Rolle
`initrc_t`	Domäne des Init-Skript, besitzt weitreichende Privilegien
`run_dbadmin_init_exec_t`	Typ von `/usr/dbadmin/run_dbadmin_init`
`run_dbadmin_init_t`	Domäne, in der `/usr/dbadmin/run_dbadmin_init` abläuft
`dbadmin_initrc_exec_t`	Typ von `/usr/dbadmin/mysqld`

Tabelle 2: Domänen und Typen für die Rolle dbadmin

Wechsel von Domäne	Wechsel zu Domäne	Eintrittspunkt
`dbadmin_t`	`run_dbadmin_init_t`	Dateien vom Typ `run_dbadmin_init_exec_t`
`run_dbadmin_init_t`	`initrc_t`	Dateien vom Typ `dbadmin_initrc_exec_t`

Tabelle 3: Domain Transitions beim Aufruf des Skripts mysqld mittels run_dbadmin_init

Das Makro `run_program` ist dabei wieder umfangreich und legt die meisten benötigten Typen und Domänen, sowie dazugehörige Transitions an. Die Bedeutung der beteiligten Typen wird in Tabelle 2 dargelegt. Die Domänenwechsel sind in Tabelle 3 aufgeführt.

Bei der Ausführung von `/usr/dbadmin/run_dbadmin_init /usr/dbadmin/mysqld` wird es zu den in Bild 3 dargestellten Domain Transitions kommen.

Nun ist die Zuordnung der Typen zu Dateien in der Policy festzuschreiben. Erstellen sie die Datei `file_contexts/misc/dbadmin.fc` mit folgendem Inhalt:

```
/usr/dbadmin/run_dbadmin_init   system_u:object_r:run_dbadmin_init_exec_t
/usr/dbadmin/mysqld             system_u:object_r:dbadmin_initrc_exec_t
```

Die Domäne `run_dbadmin_init_t` benötigt noch einige weitere Rechte, um ihre Aufgabe zu erfüllen. Hängen sie diese Zeilen an `domains/misc/dbadmin.te` an:

```
# run_program startet chkpwd unter sysadm_chkpwd_t
allow run_dbadmin_init_t dbadmin_tty_device_t:chr_file
   { read write ioctl getattr };
role dbadmin_r types sysadm_chkpwd_t;
allow consoletype_t dbadmin_tty_device_t:chr_file
   { read write ioctl getattr };
allow sysadm_chkpwd_t dbadmin_tty_device_t:chr_file
   { read write ioctl getattr };

allow dbadmin_t self:capability { dac_override dac_read_search };
allow staff_t self:capability { dac_override dac_read_search };

# Init-Skripte verwenden restorecon
```

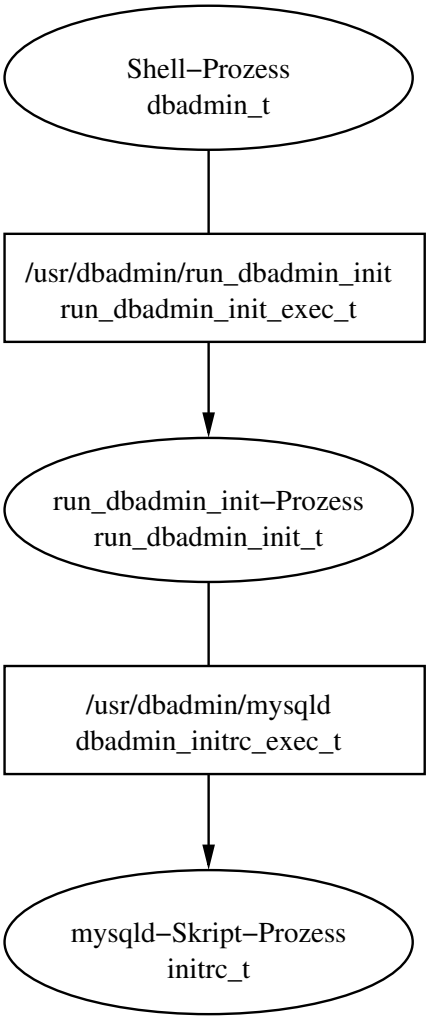

Abbildung 3: Die Domain Transitions bei Ausführung von run_dbadmin_init mysqld

```
allow restorecon_t dbadmin_tty_device_t:chr_file { read write };

# Zugriff auf Homeverzeichnisse von staff_r
file_type_auto_trans(dbadmin_t, staff_home_dir_t, staff_home_t)
rw_dir_file(dbadmin_t, staff_home_dir_t)
rw_dir_file(dbadmin_t, staff_home_t)

# anderes
allow loadkeys_t newrole_t:fd use;
allow system_chkpwd_t staff_tty_device_t:chr_file
   { read write };
```

Durch diese Regeln erfolgt gleichzeitig die Autorisation der Rolle `dbadmin_r` für die Homeverzeichnisse von `staff_r`.

Jetzt ist noch der Wechsel der Rollen von `dbadmin_r` zu `system_r` grundsätzlich zu erlauben. Hängen sie dazu an die Datei `rbac` folgende Zeile an:

```
allow dbadmin_r system_r;
```

4.4.3 Zugriff auf Konfigurationsdateien

Sind die Domäne der zugreifenden Nutzer und die Typen der Konfigurationsdateien bekannt, ist das Gestatten des Dateizugriffs sehr einfach. Folgende Regeln, an `domains/misc/dbadmin.te` angehängt, dienen als Beispiel:

```
allow dbadmin_t mysqld_etc_t:file { rw_file_perms };
allow dbadmin_t mysqld_log_t:file { r_file_perms };
```

4.4.4 Root konfigurieren

Die oben vorgenommenen Einstellungen an der Policy erlauben einem Nutzer der dbadmin_r-Rolle, das festgelegte Init-Skript auszuführen und auf bestimme Dateien des Systems zuzugreifen. Dazu wurde die Policy von SELinux entsprechend konfiguriert. Allerdings finden vor der Zugriffskontrolle durch SELinux die Kontrollen durch die herkömmlichen Unix-Mechanismen statt.

Soll ein Nutzer also tatsächlich den gewünschten Zugriff erhalten und in der Lage sein, Dinge zu tun, die root-Privilegien erfordern, so müssen ihm diese Privilegien gewährt werden.

Als einfachste Möglichkeit erhält der Nutzer das root-Passwort des Systems. Er kann dann mittels `su` die Unix-Root-Identität annehmen. Die Kontrollmechanismen von Unix erlegen ihm dann keine Schranken mehr auf. Eine Begrenzung seiner Zugriffsrechte erfolgt ausschließlich über die SELinux-Policy. Das ist eine bedeutsame

Abkehr von der Vorstellung von SELinux als „Firewall“, einer zusätzlichen Schutzmauer zwischen nicht vertrauenswürdigen Nutzern und Programmen und dem Rest des Systems.

Vor dem Einsatz der strict-Policy in dieser Art und Weise sind noch einige Anpassungen nötig.

Leider ist das Programm `su` für Fedora Core so verändert worden, dass bei einer erfolgreichen Authentifizierung für den Unix-Nutzer `root` auch der SELinux-Kontext auf `root:sysadm_r:sysadm_t` gewechselt wird. Dadurch wird der Einsatz der strict-Policy zunächst vereinfacht, da eine Anmeldung als `root` immer volle Privilegien beinhaltet. Ein eingeschränktes root-Benutzerkonto ist damit aber nicht möglich.

Um zu verhindern, dass `su` einen Kontext neu setzen kann, kommentieren sie in `macros/program/su_macros.te` Zeile 79 aus:

```
# can_setexec($1_su_t)
```

Damit `su` nicht mit einer Fehlermeldung abbricht, kommentieren sie in `/etc/pam.d/su` alle Zeilen aus, die `pam_selinux` enthalten.

Diese beiden Maßnahmen bewirken, dass der volle Nutzerkontext auch nach der Ausführung von `su` erhalten bleibt. Es hat sich nur die Unix-Identität auf `root` geändert.

Nun ist es aber immer noch möglich, sich mit dem root-Passwort auf anderem Wege am System anzumelden und somit Zugang zur sysadm_r-Rolle zu erlangen. Deshalb ist dem Nutzer `root` das Privileg dieser Rolle vollständig zu entziehen. Ändern sie den Eintrag für `root` in der Datei `users` auf:

```
user root roles { staff_r };
```

Der Superuser-Zugang ist damit nicht mehr zur Systemadministration nutzbar. Es sollte daher ein anderer Nutzer für `sysadm_r` autorisiert werden. Fügen sie einen solchen Nutzer in `users` ein:

```
user gelb roles { staff_r sysadm_r system_r };
```

Das Nutzerkonto `gelb` muss gültiges Unix-Nutzerkonto sein.

4.4.5 Abschluss und Test

Nach all diesen Änderungen kann die Policy nun neu übersetzt und geladen werden. Geben sie unter `/etc/selinux/strict/src/policy/` diese Anweisung ein:

```
make reload
```

Testweise können sie sich jetzt als `root` anmelden. Ausführung von `id` wird folgendes liefern:

```
uid=0(root) gid=0(root) { ... } context=root:staff_r:staff_t
```

Überzeugen sie sich davon, dass Zugriffe auf die meisten Systemressourcen zu `denied`-Einträgen im Log führen.

Melden sie sich nun als Nutzer `blau` am System an und wechseln sie in die Rolle `dbadmin_r`:

```
newrole -r dbadmin_r
```

Nach Ausführung von `su` und anschliessender Passworteingabe zeigt ein Aufruf von `id` folgendes:

```
uid=0(root) gid=0(root) { ... } context=blau:dbadmin_r:dbadmin_t
```

Starten und beenden sie den MySQL-Dienst mit diesen Aufrufen:

```
/usr/dbadmin/run_dbadmin_init /usr/dbadmin/mysqld start
/usr/dbadmin/run_dbadmin_init /usr/dbadmin/mysqld stop
```

Testen sie auch den Zugriff auf `/etc/my.cnf` und `/var/log/mysqld.log`.

Bleibt nur noch, auch den Zugang des Systemadministrators auszuprobieren. Melden sie sich dazu als Nutzer gelb an. Folgende Befehle erlauben den vollen Zugriff auf das System:

```
newrole -r sysadm_r
su -
```

Bestätigen sie den Kontext durch `id`:

```
uid=0(root) gid=0(root) { ... } context=rot:sysadm_r:sysadm_t
```

Das System sollte nun in den Enforcing Mode versetzt werden.

Literatur

[1] DTOS General System Security and Assurability Assessment Report, *http://www.cs.utah.edu/flux/fluke/html/dtos/HTML/final-docs/mer.pdf*

[2] The Inevitability of Failure: The Flawed Assumption of Security in Modern Computing Environments, *http://www.nsa.gov/selinux/papers/inevit-abs.cfm*

[3] Red Hat SELinux Guide, *http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/selinux-guide/*

[4] Configuring the SELinux Policy, *http://www.nsa.gov/selinux/papers/policy2-abs.cfm*