



30 SELinux-Policy-Editoren und -IDEs

Bei der Entwicklung von SELinux-Policies wünscht man sich als Administrator schnell eine Oberfläche, die einem bei der Erzeugung und Fehlersuche behilflich ist. Es gibt eine ganze Reihe von Programmen, die versuchen, dies zu leisten. Im Folgenden will ich eine Auswahl kurz vorstellen.

Es handelt sich im Einzelnen um:

- Vim
- SELinux Policy IDE (SLIDE)
- SELinux Policy Editor
- Cross Domain Solutions Framework (CDS Framework)

30.1 Vim als SELinux-Editor

Der Editor Vi oder Vim ist sicherlich für viele Administratoren das Werkzeug der Wahl zum Editieren von Textdateien. Zumindest auf mich trifft das zu. Vim kennt Syntax-Highlighting, das bei der Erstellung von Type-Enforcement-Dateien helfen kann. Sie erkennen dann auf den ersten Blick, ob die Syntax Ihrer Datei stimmt oder nicht. Hierzu benötigen Sie jedoch eine Syntax-Beschreibungsdatei. Eine derartige Datei wurde von Thomas Bleher erzeugt und unter der folgenden URL veröffentlicht: <http://www.cip.informatik.uni-muenchen.de/~bleher/selinux/te.vim>.

Diese Datei müssen Sie in Ihrem Verzeichnis `~/ .vim/syntax` ablegen. Dann müssen Sie noch die folgende Zeile zu Ihrer Datei `~/ .vimrc` hinzufügen:

```
autocmd BufRead,BufNewFile          *.te set filetype=te
```

Anschließend wird der Vim Type-Enforcement-Dateien mit Syntax-Highlighting in verschiedenen Farben darstellen. Einen Screenshot spare ich mir hier, da dieser in Schwarz-Weiß keinen Sinn machen würde.

30.2 SELinux Policy IDE (SLIDE)

Die SELinux Policy IDE ist ein Plugin für die Eclipse Integrated Development Environment (IDE). Eclipse ist ein Open-Source-Framework. Dabei unterstützt die gra-

fische Oberfläche Eclipse die Entwicklung von Software jeder Art. Die häufigste Anwendung ist sicherlich der Einsatz als Java-IDE. Dies ist auch der ursprüngliche Zweck der Software.

Die Firma Tresys hat für Eclipse ein SELinux-Plugin geschaffen, das Sie bei der Entwicklung von SELinux-Modulen unterstützt.

30.2.1 SLIDE-Installation

Bevor Sie SLIDE installieren, müssen Sie einige Voraussetzungen schaffen. Sie benötigen:

- Eclipse Version ≥ 3.1
- Setools
- Reference Policy
- Checkpolicy Version ≥ 1.28

Diese Pakete sind jedoch in den meisten modernen SELinux-Distributionen enthalten.

Auf der SLIDE-Homepage¹ stehen fertige Pakete für die Fedora Core-Distributionen zur Verfügung. Falls Sie diese Distribution nutzen, ist die Verwendung des RPM-Pakets sicherlich die einfachste Variante.

Falls Sie die Debian-Distribution einsetzen, können Sie SLIDE entweder über die Eclipse-Update-Site oder aus den Quellen installieren. Wenn Sie den Eclipse-Update wählen, nutzen Sie im Menü HELP den Punkt SOFTWARE UPDATES und dort FIND AND INSTALL. Nach Auswahl von SEARCH FOR NEW FEATURES TO INSTALL legen Sie eine neue Site mit den folgenden Informationen an:

- Name: Tresys Technology
- URL: <http://oss.tresys.com/eclipse-update/>

Nun können Sie direkt SLIDE aus dem Internet installieren. Ansonsten können Sie aber auch den Subversion-Zugang unter *svn co* <http://oss.tresys.com/repos/slide/trunk/slide-plugin> benutzen. Dieser letzte Weg wird jedoch nicht empfohlen.

30.2.2 SLIDE-Funktionen

SLIDE unterstützt Sie nun bei der Anlage und Verwaltung Ihrer Module. Es bietet Syntax-Highlighting und automatische Vervollständigung.

Um ein neues Projekt anzulegen, wählen Sie FILE NEW und dann PROJECT. In dem nun erschienenen Dialog (Abbildung 30.1) wählen Sie das SLIDE Project aus.

In dem nächsten Dialog wählen Sie, ob Sie nur ein Modul oder eine komplette neue Policy erzeugen möchten (Abbildung 30.2). Im zweiten Fall wird die vorhandene Reference-Policy für die Anpassung kopiert.

¹ <http://oss.tresys.com/projects/slide>

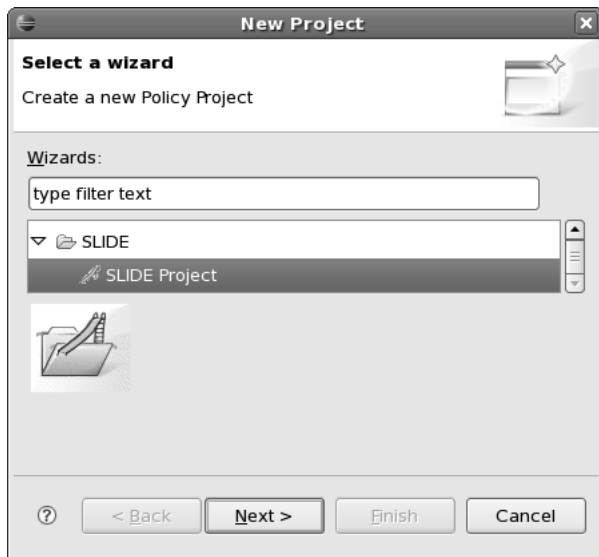


Abbildung 30.1: Eclipse bietet die Erzeugung von SLIDE-Projekten.

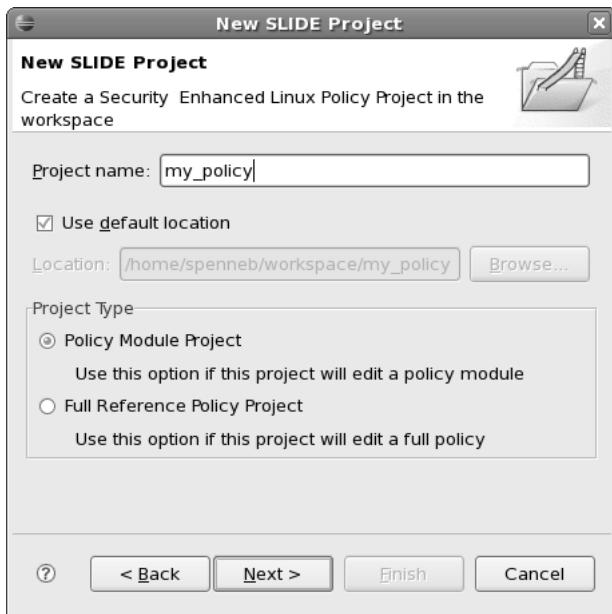


Abbildung 30.2: SLIDE kann Sie sowohl beim Erstellen von Policy-Modulen als auch beim Erstellen von ganz neuen Reference-Policies unterstützen.

Auf der folgenden Seite müssen Sie den Ort der Reference-Policy angeben (Abbildung 30.3). Möchten Sie nur ein Modul erzeugen, müssen Sie hier mindestens den

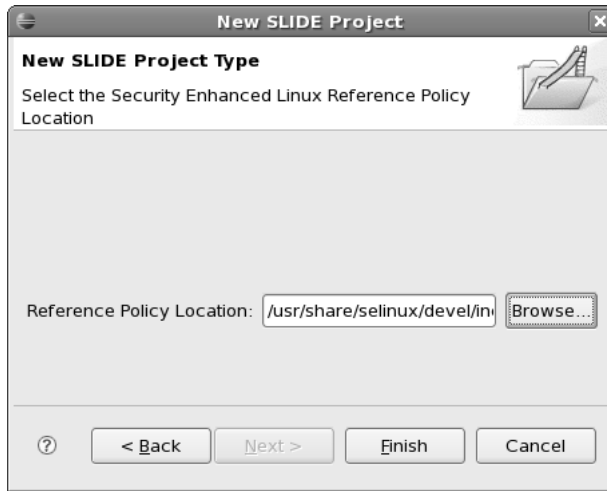


Abbildung 30.3: SLIDE benötigt den Pfad zur Reference-Policy.

Pfad zu den Include-Dateien der aktuellen Policy angeben. Möchten Sie eine neue Policy bauen, müssen sich hier auch alle Quellen befinden, denn SLIDE wird diese zunächst für Ihr Projekt kopieren, damit Sie diese anpassen können.

Nun können Sie ein neues Modul erzeugen. Hierzu wählen Sie FILE/NEW und dann SLIDE MODULE. In dem folgenden Fenster (Abbildung 30.4) geben Sie einen Namen für das Modul an.

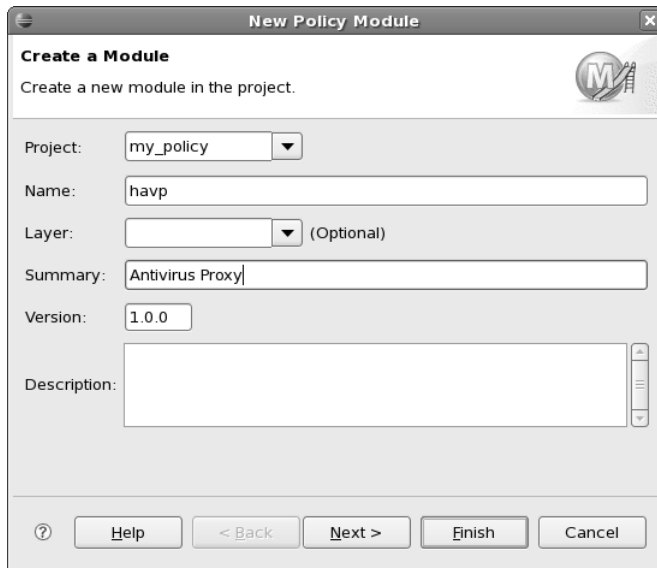


Abbildung 30.4: SLIDE fragt Sie nach den wesentlichen Informationen des Moduls.

Auf der nächsten Seite (Abbildung 30.5) stellt SLIDE Ihnen Fragen, die denen des Befehls `policygentool` ähneln. Auch aus diesen Fragen erzeugt SLIDE ein Grundgerüst für das Modul.

Anschließend können Sie in der IDE auf Ihre *Type-Enforcement*-Datei zugreifen. Bei dem Editieren unterstützt SLIDE Sie mit der Vervollständigung Ihrer Eingabe bei Bedarf. Hierzu drücken Sie einfach `STRG`+`Leertaste` (Abbildung 30.6). Die Auswahl erfolgt dann mit der Maus.

Über die rechte Maustaste steht auch noch ein Kontext-Menü mit Autotexten zur Verfügung, die größere Block-Templates automatisch hinzufügen können. Hier stehen zur Verfügung:

- ADD TUNABLE POLICY BLOCK
- ADD OPTIONAL POLICY BLOCK
- ADD IFDEF BLOCK

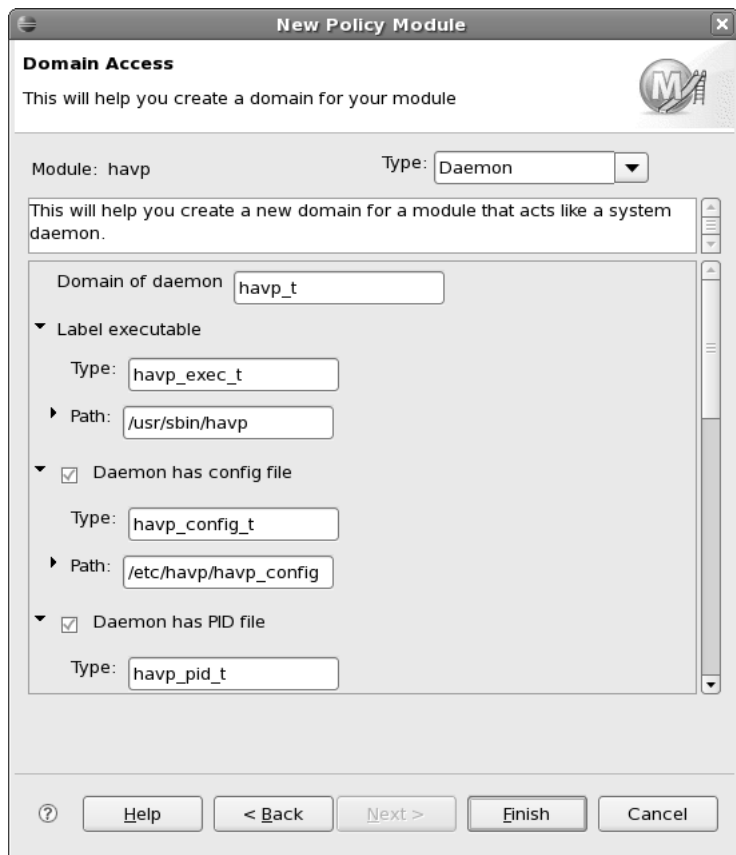


Abbildung 30.5: SLIDE erstellt aus den Antworten ein Grundgerüst für die Policy.

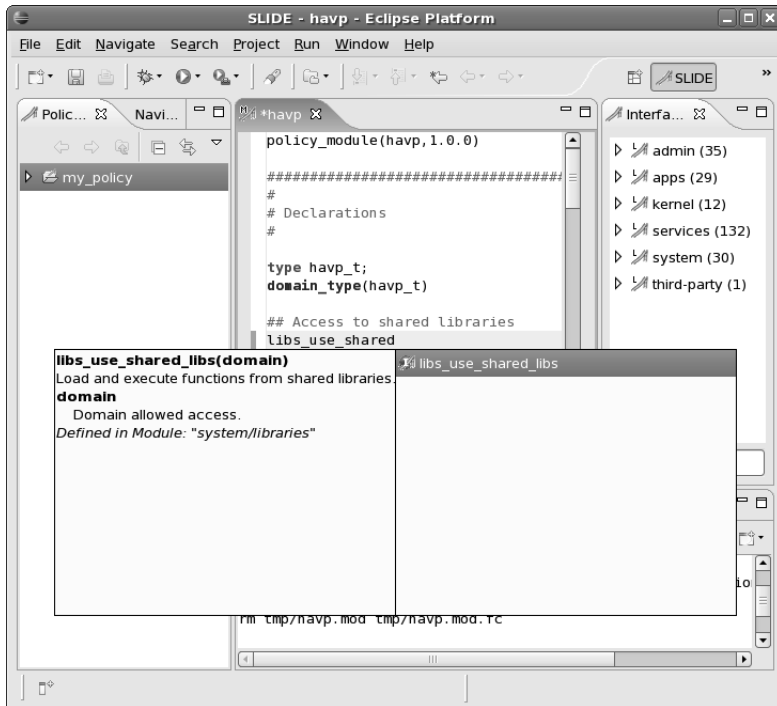


Abbildung 30.6: SLIDE kennt alle Interfaces und kann mit **STRG+Leertaste** eine Autovervollständigung anbieten.

- MOVE OUT OF BLOCK
- REMOVE BLOCK
- TOGGLE COMMENT

Ein Test des Moduls kann über die rechte Maustaste und das Menü RUN AS und RUN ausgelöst werden. Zunächst müssen Sie hier aber noch die Umgebung konfigurieren. Hierzu existiert nach der Auswahl POLICY TEST ein Button NEW². Wählen Sie dann in der Registerkarte MAIN zunächst Ihr Projekt und dann in der Registerkarte POLICY Ihr Modul (Abbildung 30.7). Geben Sie der Konfiguration einen sinnvollen Namen, und Sie sind mit der Konfiguration fast fertig. Optional können Sie auch noch ein Test-Script angeben. Dies wird gestartet, sobald die Policy geladen wurde.

Damit der Test aber auch tatsächlich durchgeführt werden kann, benötigen Sie noch ein Testsystem, und SLIDE benötigt eine Netzwerkverbindung mit dem Testsystem. Hierbei kann es sich auch um das lokale System handeln. Auf dem Testsystem muss aber zuvor das Paket *SLIDE!-Remote* installiert sein. Dieser Dienst wird von SLIDE für die Kommunikation mit dem Testsystem genutzt. SLIDE erhält über diesen Dienst

² Dieser ist bei Fedora als Icon mit einem kleinen Plus oben versteckt.

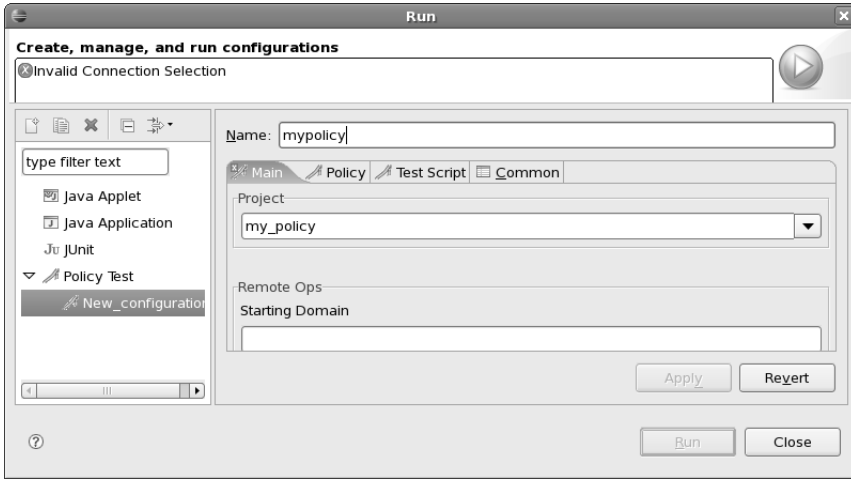


Abbildung 30.7: SLIDE erlaubt auch den Test der Module.

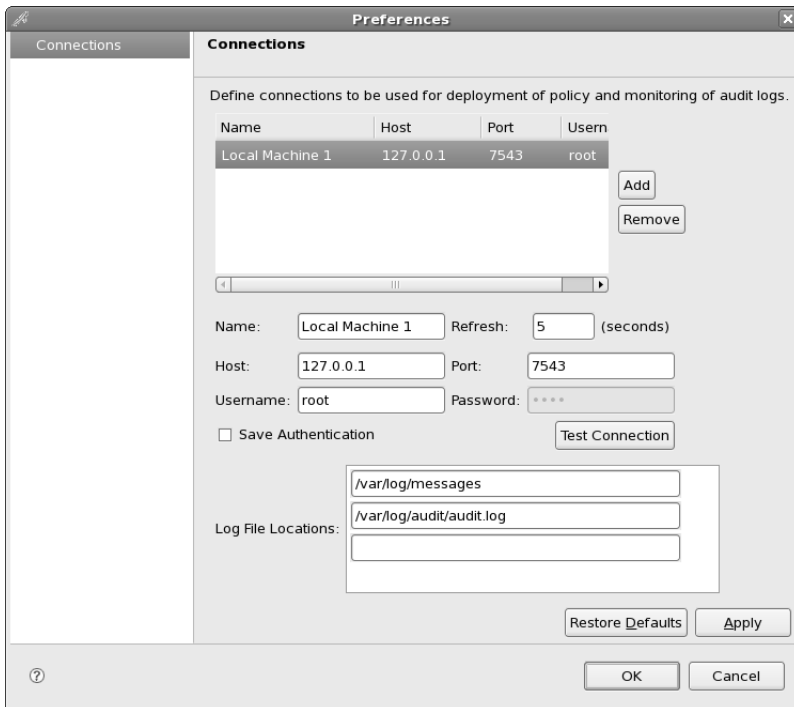


Abbildung 30.8: Für den Test benötigt SLIDE eine Verbindung zu SLIDE-Remote auf dem Testsystem.

die Audit-Meldungen des Testsystems und überträgt das Modul auf das Testsystem (Abbildung 30.8).

SLIDE-Remote ist auch über die SLIDE-Homepage verfügbar. Es sollte im Moment lediglich für die Entwicklung von SELinux-Richtlinien eingesetzt und nicht auf Produktionssystemen verwendet werden.

30.3 SELinux Policy Editor (SEedit)

Das Programm `seedit` wurde zuerst von Hitachi Software entwickelt³. Seitdem hat Yuichi Nakamura es überarbeitet und verbessert. Die aktuelle Version steht unter <http://seedit.sourceforge.net> zur Verfügung.

Bei SEedit handelt es sich im Wesentlichen um drei Komponenten:

- Eine neue SELinux-Policy (*Simplified-Policy*)
- Einen SELinux-Policy-Compiler für die Simplified Policy Description Language
- Einen grafischen Editor für die Verwaltung

Die neue Simplified Policy wurde in der *Simplified Policy Description Language* (SPDL)⁴ geschrieben. Für die Übersetzung in die binäre SELinux-Policy enthält das Projekt den SPDL-Compiler `seedit-converter`. Dieser erzeugt die binäre Policy. Die Administration erfolgt sehr bequem über eine grafische Oberfläche.

30.3.1 Simplified Policy Description Language (SPDL)

Die Simplified Policy Description Language verwendet eine Syntax, die der Syntax ähnelt, die von AppArmor verwendet wird. Dadurch kapselt sie die komplizierten internen Funktionen wirkungsvoll, sodass der Anwender sich keine Gedanken um Typen, abstrakte Berechtigungen etc. machen muss. Am deutlichsten wird die Funktion der SPDL an einer Beispiel-Policy für den Dienst `ntpd`:

```
{
domain ntpd_t;
program /usr/sbin/ntpd;
include common-relaxed.sp;
include daemon.sp;
include nameservice.sp;

allowpriv cap_ipc_lock;
allowpriv cap_sys_time;
allowpriv cap_sys_resource;
allowpriv netlink;

allow /etc/ntp.conf r,s;
```

³ <http://hitachisoft.jp/Products/secure-linux/>

⁴ http://seedit.sourceforge.net/doc/2.1/spdl_spec.pdf

```
allow /etc/ntp/** r,s;
allow /var/lib/ntp/** r,w,s;

allownet -protocol udp,tcp -port 123 client,server;
allownet -protocol  udp -port 53 client;

allow etc_runtime_t r,s;
}
```

Eine File-Context-Datei ist nicht erforderlich. Diese wird aus der SPDL-Datei automatisch erzeugt. Schlüsselwörter definieren nun das Verhalten:

- `domain`: Dies definiert den Namen der Domäne.
- `program`: Dies definiert das Binary, das überwacht werden soll.
- `include`: Hiermit werden die Regeln aus weiteren Dateien eingelesen.
- `allowpriv`: Hiermit erlauben Sie den Zugriff auf eine Capability.
- `allow`: Hiermit erlauben Sie den Zugriff auf Dateien. Wildcards (*) sind erlaubt. Die Angabe von zwei Sternen erlaubt den Zugriff auf den gesamten Verzeichnisbaum.
- `allownet`: Hiermit erlauben Sie den Zugriff auf bestimmte Protokolle und Ports.

Die Policy in der SPDL-Sprache ähnelt einem AppArmor-Profil. Sie ist sofort von jedem UNIX-Administrator nachzuvollziehen und zu verstehen.

Für die automatische Erzeugung der benötigten Richtlinien können Sie das Werkzeug `audit2spdl` verwenden. Dieses arbeitet analog dem Befehl `audit2allow`.

Während die aktuellen Richtlinien noch keine Role-Based-Access-Control nutzen, kann die aktuelle Version 2.1 des Kommandos `seedit` bereits damit umgehen. Nach dem Aktivieren der RBAC-Unterstützung mit `seedit-rbac on` stehen drei Rollen zur Verfügung:

- `sysadm_r`
- `staff_r`
- `user_r`

Diese Rollen können auch einfach um weitere Rollen erweitert werden. Um zum Beispiel eine Rolle `webmaster_r` in der SPDL anzulegen, verwenden Sie den folgenden Befehl, der Ihnen zunächst die Grundregeln liefert:

```
[root@supergrobi ~]# seedit-template -r webmaster_r -u webmaster

role webmaster_r;
user webmaster;
include user_common.sp;
include common-relaxed.sp;
allow ~/** r,w,s;
```

```
allowpriv part_relabel;  
allowpriv dac_override;  
allowpriv dac_read_search;
```

Diese können Sie nun anpassen und zum Beispiel die folgenden Zeilen hinzufügen:

```
allow /etc/httpd/** r,w,s;  
allow /var/www/* r,w,s;
```

30.3.2 SEedit-Installation

Die Installation von `seedit` ist unkritisch. Für Fedora existieren ab der Version 6 fertige Pakete. Auch für CentOS 5 sind Pakete über die Homepage verfügbar.



Achtung

Leider lassen sich die mit `seedit` erzeugten Richtlinien nicht mit den auf klassischem Wege erzeugten Richtlinien mischen. Sie müssen sich dann schon entscheiden, ob Sie die klassische Methode oder die SEedit-Methode wählen möchten.

30.4 Cross Domain Solutions Framework (CDS Framework)

Das *CDS Framework* Toolkit ist ein neues Werkzeug, das die konzeptionelle Erzeugung von Richtlinien erlaubt. Es stellt ähnlich SEedit eine Hochsprache zur Verfügung, in der die Eigenschaften einer Applikation oder Gruppe von Applikationen beschrieben werden. Diese Hochsprache kann anschließend in eine SELinux-Richtlinie übertragen werden. Somit erleichtert dieses Werkzeug die Erzeugung von SELinux-Richtlinien.

Das CDS Framework Toolkit besteht aus den folgenden Komponenten:

- Konzeptionelles Modell: Dieses Modell bietet dem Richtlinienautor nur wenige wichtige Sicherheitseigenschaften. Hierbei handelt es sich in erster Linie um den Informationsfluss.
- Hochsprache
- Integrated Design Environment(IDE): Eine Oberfläche, in der die Richtlinie entsprechend dem Modell beschrieben werden kann.
- Compiler: Dieser Compiler erzeugt aus der Hochsprache direkt die SELinux-Richtlinie.

Das CDS Framework benötigt das Eclipse Graphical Editing Framework. Dieses lässt sich bei den meisten Distributionen über ein Paket nachinstallieren.

30.4.1 CDS-Konzept

Das CDS verfolgt mit seinem Konzept zwei Ziele: Überwachung des Informationsflusses und Trennung der Informationsdomänen. Hierzu definiert es die folgenden Begriffe:

- `domain`
Eine CDS-Domäne ist eine Informationsdomäne. Eine Domäne kann verschiedene Objekte, wie Prozesse, Dateien und Sockets, enthalten. Alle Prozesse innerhalb einer Domäne besitzen dieselben Privilegien, und alle Ressourcen genießen denselben Schutz. Prozesse innerhalb einer Domäne haben normalerweise unbeschränkten Zugriff auf die Ressourcen in dieser Domäne⁵. Im grafischen Editor werden Domänen als Kästen dargestellt.
- `shared resource`
Eine gemeinsam genutzte Ressource ist ein Objekt, über das zwei Domänen Informationen austauschen können. Typisch ist dies für Dateien, Sockets und Named Pipes. Im grafischen Editor werden diese als Kreise dargestellt.
- `access`
Der Zugriff definiert, wie eine Domäne auf eine Ressource zugreifen darf. Normalerweise hat jeder Prozess vollen Zugriff auf die Ressourcen in der eigenen Domäne. Das bedeutet, dass nur noch die Zugriffe auf gemeinsam mit anderen Domänen genutzte Ressourcen (`shared resource`) definiert werden müssen. CDS kennt drei Zugriffe, die mit Pfeilen symbolisiert werden: lesen (`read`), schreiben (`write`) und lesen-und-schreiben (`readwrite`).
- `decomposition`
Dieses Modell würde nicht den Aufbau von komplexen Rechtesystemen erlauben. Daher ermöglicht es das CDS, eine Domäne in weitere Domänen (`children`) zu unterteilen und die Zugriffe innerhalb der Domäne zu verwalten. Auch die Subdomänen dürfen nur über Shared Resources kommunizieren. Auch Subdomänen können weiter unterteilt werden.

Das CDS Toolkit enthält einige grundsätzlich definierte Ressourcen und Domänen. Diese werden im grafischen Editor als Kreise und Kästen mit gestrichelter Linie dargestellt. Diese Domänen können nicht unterteilt werden.

Auch der Domänenwechsel kann grafisch dargestellt werden. Das CDS verwendet hier sowohl den `entrypoint` als auch `enter`. Ein `Entrypoint` ist eine Datei, deren Aufruf den Domänenwechsel auslöst. Dieser `Entrypoint` wird grafisch als Fünfeck dargestellt. Zusätzlich zeigt ein Pfeil von der ausgehenden Domäne zur Zieldomäne (siehe Abbildung 30.9).

Schließlich erlaubt das CDS noch die Definition von Steuerungseinrichtungen wie `Inter-Process-Calls` und `Message-Queues`. Diese werden als kleine Kreise dargestellt.

⁵ Dies ist bei SELinux normalerweise nicht so!

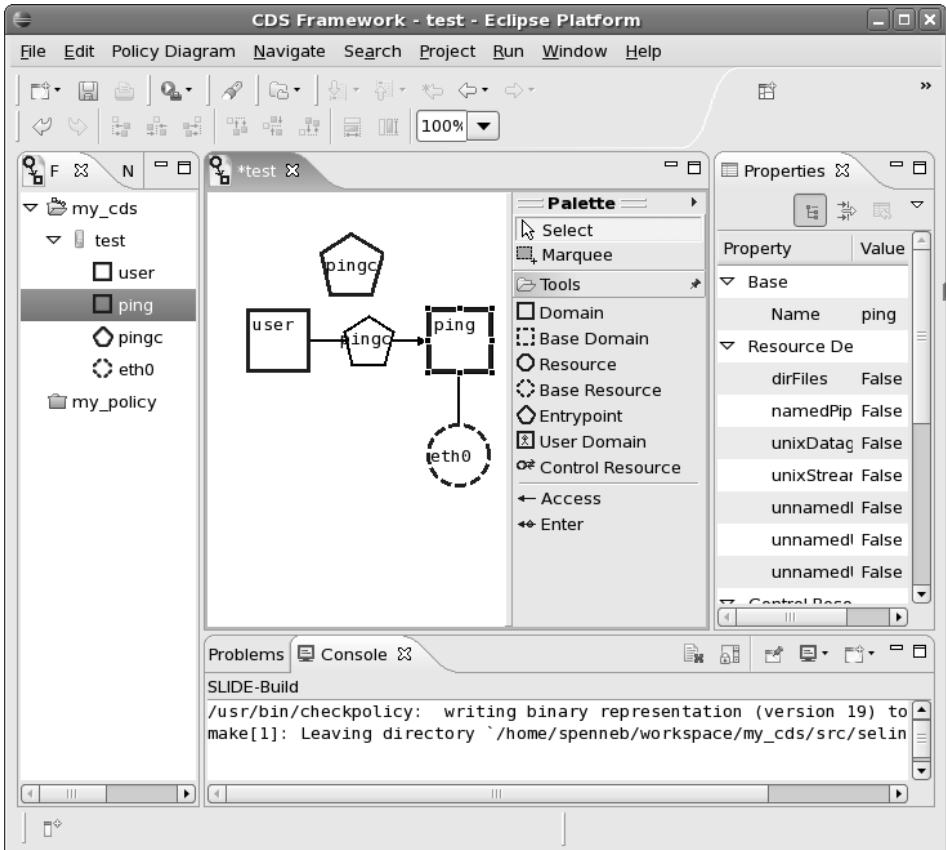


Abbildung 30.9: Eclipse erlaubt mit dem CDS Toolkit die Erzeugung abstrakter Richtlinien.