SELinux Policy Editor(seedit) Administration Guide

Yuichi Nakamura [*]

July 3, 2006

Contents

[*] himainu-ynakam@miomio.jp

This document is manual for SELinux Policy Editor. You can learn by example what is SELinux Policy Editor, and How to use.
About how to install see Install Guide.

1 What is SELinux Policy Editor

SELinux is included in many distros, but it has been disabled by many users because of they feel SELinux is too difficult. SELinux Policy Editor(seedit) is a tool that make SELinux easy. seedit is composed of Simplified Policy and utilities that handle Simplified Policy.The main component is Simplified Policy. Simplified Policy is a SELinux policy that is described by Simplified Policy Description Language(SPDL).SPDL resolves difficulty of SELinux. SPDL simplifies SELinux by reducing number of permissions and hiding labels. Following is a example of policy described by SPDL.

```
{
domain httpd_t;
program /usr/sbin/httpd;
...
allow /var/www/** r,s;
allownet -protocol tcp -port 80 server;
...
}
```

You can easily understand what the policy says. Customize is also easy, because of helper tools. It's original version was developed by Hitachi Software(http://www.selinux.hitachi-sk.co.jp/). It has been re-designed and almost re-written by Yuichi Nakamura(ynakam@gwu.edu) since version 1.0.

2 Background of SELinux

You have to be familiar with some SELinux background, especially following.

(1) TE(Type-Enforcement)
Access control model of SELinux is called TE. In TE, process is given *domain.* SELinux decides access control based on configuration file called *policy.* In policy , *What kind of resource a domain is allowed to access ?* is described. To identify resources, SELinux uses label called *type*, but you do not have to be worry about *type*, because it is hidden in seedit world. By giving proper domain to application and configuring domain properly, the application have least privilege.

(2) Enforcing/permissive mode
SELinux have two mode, enforcing and permissive mode. Enforcing mode is normal mode. Access control is effective.
Permissive mode is a test mode. Even if there is a access that is denied by

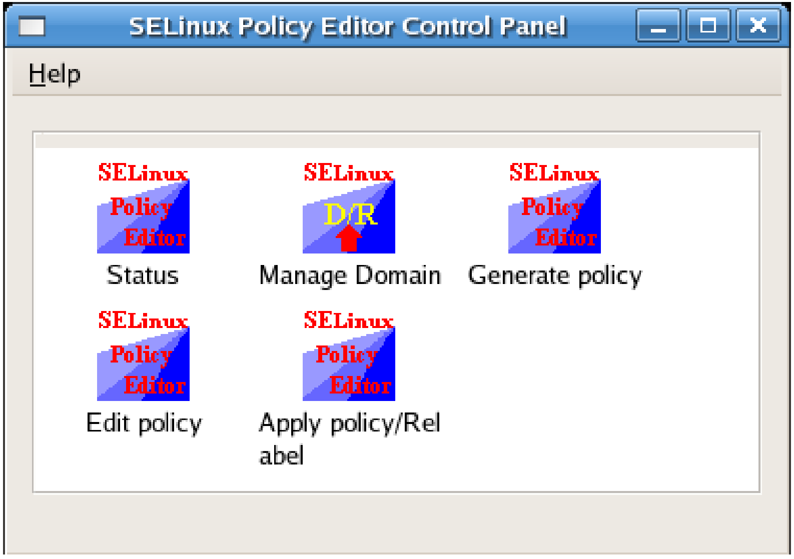
SELinux, it is not actually denied, but only written to log. In permissive mode, SELinux is effectively disabled, but useful to test the behavior of access control. To see current mode, you can use *getenforce* command. To switch between enforcing/permissive mode, you can use *setenforce* command. The usage will appear later in the document.

(3) SELinux access denial log
Access denial is outputted in /var/log/messages in Fedora Core5. In Fedora Core4 or using auditd service, it is outputted to /var/log/audit/audit.log.

3 Overview of GUI

You can manage SELinux from SELinux Policy Editor's GUI. GUI is X Window based. In this tutorial, how to use GUI is shown first. You can also manage from command line, it is shown later.
You can launch GUI from Gnome menu. Choose Desktop → Manage → SELinux Policy Editor. You can also launch GUI, by typing seedit-gui.
You will see window *SELinux Policy Editor Control Panel*. In the control panel, you will see icons(you will see buttons instead of icons in Cent OS 4) like figure 1.

Figure 1: SELinux Policy Editor Control Panel



By double-clicking icon, management windows open. What you can do by double-clicking icons is summarized in following.

- Status
The usage is shown in section 4.3.

  - See status of SELinux, such as SELinux mode, domains for running process, domains for network process.
  - Change SELinux mode

- Manage Domain
  - Create new domain(section 9)
  - Remove domain, disable domain(section 6.1)

- Generate Policy(section 8.3)
  - Generate policy from access log

- Edit policy(section 7.3)
  - Edit policy by text editor. The text editor has some useful features.

- Apply policy,Relabel
  - Load policy manually
    Policy is load automatically in GUI, but you can do it manually.
  - Initialize all file labels

  - Run restorecon command

4 See what's happening on your system

After install, you have to see what's happening(status of SELinux) on your system.

4.1 Where is policy?

Simplified policy(policy described by SPDL) is located at /etc/seedit/policy, it is explained in later section. Simplified Policy is converted into SELinux policy by seedit-load command(inside the command, seedit-converter runs, it does main task), and SELinux Policy(binary SELinux policy,file_contexts is generated). Generated SELinux policy is located at /etc/selinux/seedit/policy, generated file_contexts is at /etc/selinux/seedit/contexts/files. Usually, you do not have to care about generated policy.

4.2 Default policy

Installed simplified policy is a targeted one. Not strict policy. It does not include RBAC support. Only selected daemons are protected. Simplified policy can support RBAC and more strict policy, but such policies are under construction :-) .About RBAC, it is ready to use, if you are interested in it, see RBAC guide.

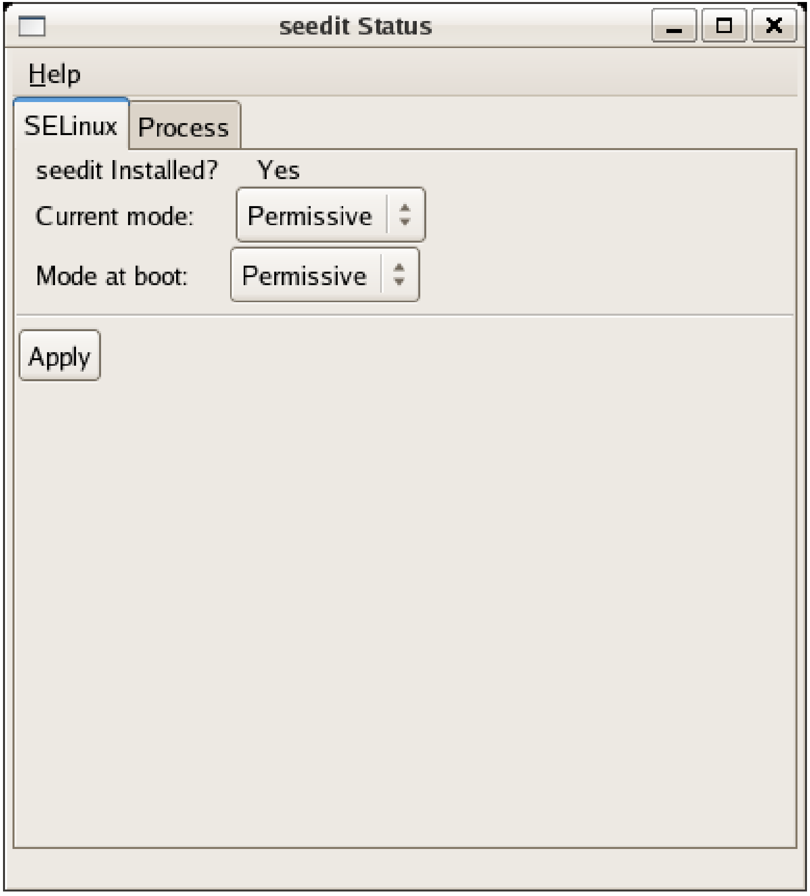
4.3 Check status of SELinux(GUI)

To check status of SELinux, select *Status* from control panel, then window named*seedit Status* opens.

4.3.1 Check/Switch enforcing/permissive mode

From *SELinux* tab, you can check/change mode of SELinux. Figure 2 is screenshot.

Figure 2: Check/change SELinux mode



seedit Installed? Yes
Current mode: Permissive
Mode at boot: Permissive

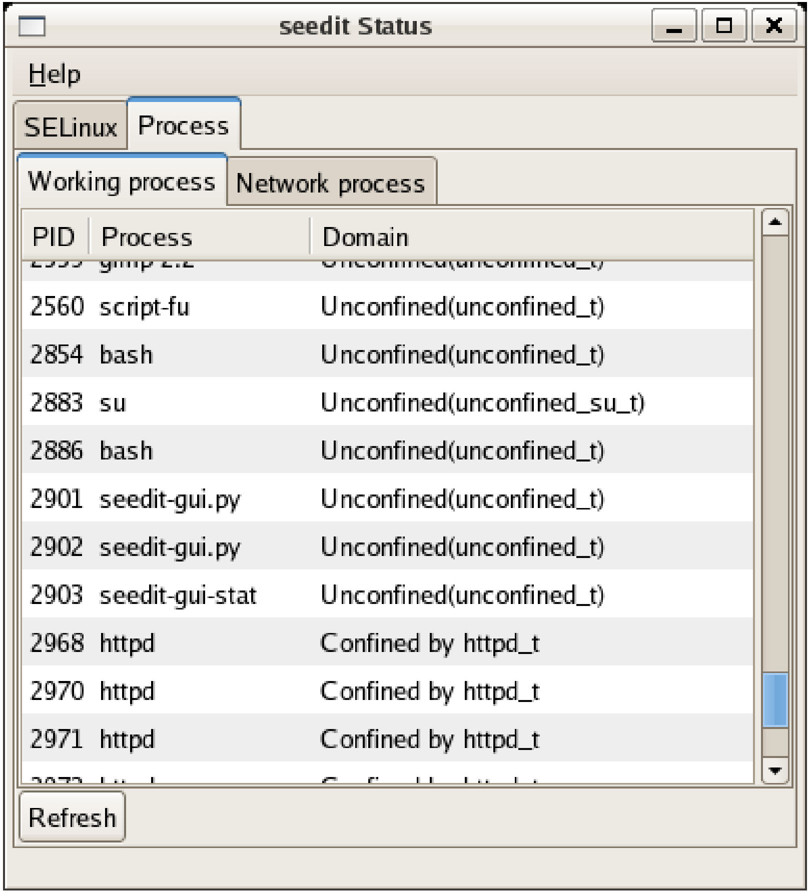
By *seedit Installed? Yes*, you can know seedit is successfully installed. From *Current mode*, you can see current mode is permissive mode. You can change current mode from this box, select *Enforcing* and press *Apply* button.
Mode at boot is mode at system boot, if it is *Permissive*, after reboot, system will run as permissive mode. It is strongly recommended to configure as Enforcing when you use real system.

4.3.2 See running process

Some processes are given domain and behavior is confined.

From process→Working Process, you can see domain of running process. Fig 3 is example output.

Figure 3: Check domain of running processes



PID	Process	Domain
2560	script-fu	Unconfined(unconfined_t)
2854	bash	Unconfined(unconfined_t)
2883	su	Unconfined(unconfined_su_t)
2886	bash	Unconfined(unconfined_t)
2901	seedit-gui.py	Unconfined(unconfined_t)
2902	seedit-gui.py	Unconfined(unconfined_t)
2903	seedit-gui-stat	Unconfined(unconfined_t)
2968	httpd	Confined by httpd_t
2970	httpd	Confined by httpd_t
2971	httpd	Confined by httpd_t

You can see such process as bash is unconfined, and httpd is confined by httpd_t domain.

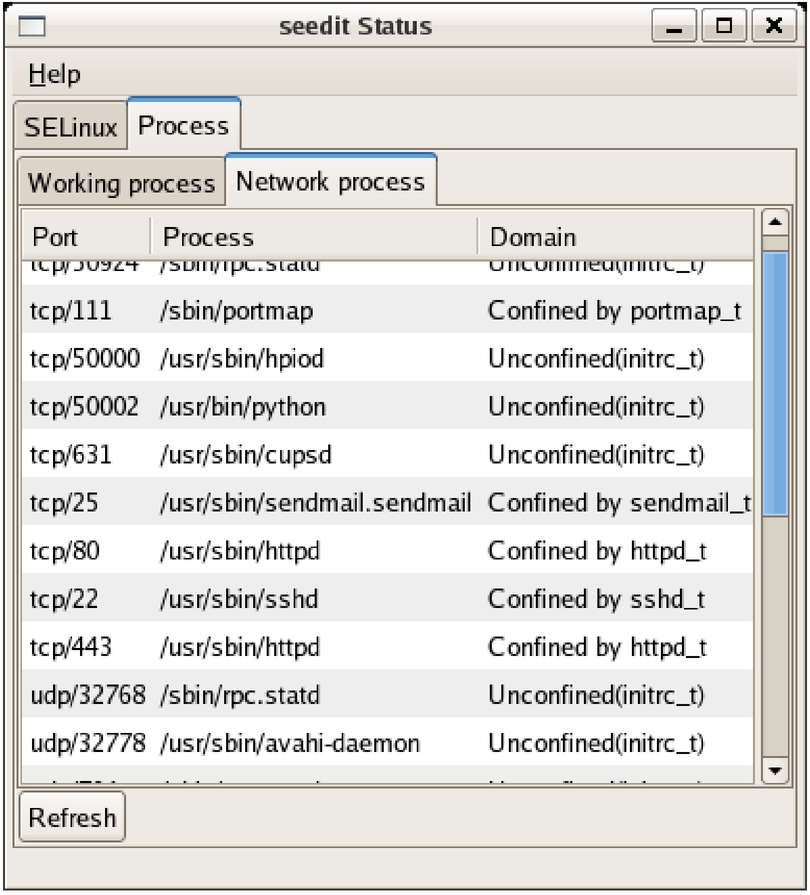
You can also sort result by selecting PID, Process, Domain. And by *Refresh* button, result is updated.

4.3.3 See network process

You can see status of network process(that is waiting network connection) from process→Network Process It is important to know status of network process. Because attackers use network process to intrude.

Fig 4 is example output. From above, You have to be especially careful to service whose domain is unconfined. If you create domain, you can confine it, or you may decide not to run the service.

Figure 4: Check domain of network processes



Port	Process	Domain
tcp/30924	/sbin/rpc.statd	Unconfined(initrc_t)
tcp/111	/sbin/portmap	Confined by portmap_t
tcp/50000	/usr/sbin/hpiod	Unconfined(initrc_t)
tcp/50002	/usr/bin/python	Unconfined(initrc_t)
tcp/631	/usr/sbin/cupsd	Unconfined(initrc_t)
tcp/25	/usr/sbin/sendmail.sendmail	Confined by sendmail_t
tcp/80	/usr/sbin/httpd	Confined by httpd_t
tcp/22	/usr/sbin/sshd	Confined by sshd_t
tcp/443	/usr/sbin/httpd	Confined by httpd_t
udp/32768	/sbin/rpc.statd	Unconfined(initrc_t)
udp/32778	/usr/sbin/avahi-daemon	Unconfined(initrc_t)

4.4 seedit-unconfined(Command)

You can also do the same task from seedit-unconfined command.

4.4.1 See running process

You can see status of processes by seedit-unconfined -e. You have to be root to obtain correct result. Following is example output.

```
$ su -
# seedit-unconfined -e
```

```
Current SELinux mode: permissive ----(1)
PID     Comm    Domain
1       init    Unconfined(init_t) ---(2)
...
1853    sshd    Confined by sshd_t ---(3)
```

(1) is current SELinux mode this says permissive mode. Note again that system is not actually protected by SELinux in permissive mode.(2) says process init is not unconfined, and given unconfined domain init_t. *unconfined* means: the given domain is allowed everything, and effectively not protected by SELinux. (3) says sshd is given sshd_t domain and sshd_t domain is configured to confine behavior of sshd.
You can also see domains by ps -eZ command, but it does not tell us what kind of domains are unconfined. /etc/selinux/seedit/policy/unconfined_domains will tell you what kind domains are unconfined.

4.4.2 See network process

You can see status of network process(that is waiting network connection), by seedit-unconfined -n. It is important to know status of network process. Because attackers use network process to intrude. Sample output is following.

```
#seedit-unconfined -n
Current SELinux mode: permissive ----(1)
/usr/sbin/smbd  Unconfined(initrc_t) -- (2)
/usr/sbin/sendmail.sendmail     Confined by sendmail_t --(3)
...
```

It is like AppArmor's unconfined command. (1) is current SELinux mode. After network programs are shown. (2) says smbd is not confined. (3) says sendmail is confined by domain sendmail_t.

4.4.3 Switching enforcing/permissive mode

After install system is permissive mode. you can switch to enforcing mode by following commands.

```
# setenforce 1
# getenforce
enforcing
```

If you want to be enforcing mode in boot time, you have to modify /etc/selinux/config like following. It is strongly recommended when you use real system.

```
SELinux=permissive
-->
SELINUX=enforcing
```

5 Then, what should we do?

Now you know SELinux status of your system(what's confined,unconfined). What you have to do after that can be summarized like below.

(1) Unconfine applications
If you find application does not run due to SELinux access denial, you can choose unconfine such application.It's easy. You can see how to in section 6.

(2) Modify policy
If you find application does not run due to SELinux access denial, but still want to confine application, you can modify policy. See section 8.3.

(3) Confine more applications by creating domains
If you want to confine unconfined applications, you can prepare domain for them. See section 9.

6 Unconfine applications

You can unconfine application by 2 ways. Use boolean or remove config file.

6.1 GUI

You can do it from GUI. Select *Manage Domain*, then *seedit Domain/Role Manager* window opens. Select *Delete Domain* tab.

6.1.1 Temporally disable

The easiest way to unconfine application is to disable domain temporally.

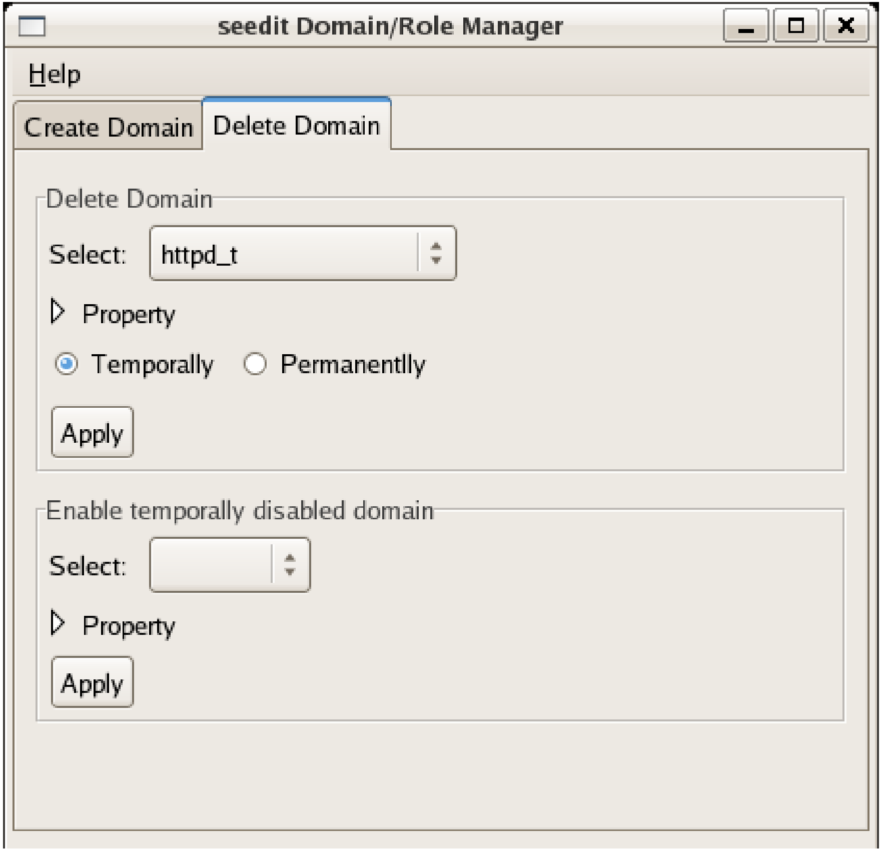
(1) Select domain which you want to disable

(2) Select radio button *Temporally*

(3) Press Apply button

For example, Apache is confined by httpd_t domain and you want to unconfine Apache. Select httpd_t, and press apply button. Restart Apache and check domain by *Status* GUI, you will see *Unconfined(initrc_t)* is shown in domain.
To confine it again, select domain from *Enable temporally disabled domain*, and press *Apply* button.
This behavior is using boolean of SELinux, you can see detail by command line.

Figure 5: Unconfine application



seedit Domain/Role Manager

Help

Create Domain | Delete Domain

Delete Domain

Select: httpd_t

▷ Property

● Temporally ○ Permanentlly

Apply

Enable temporally disabled domain

Select:

▷ Property

Apply

6.1.2 Remove domain

Another way to unconfine application is to remove configuration file for domain. The procedure is following.

(1) Select domain which you want to disable

(2) Select radio button *Permanently*

(3) Press Apply button

To confine application again, you have to do it by hand.

(1) cd /etc/seedit/policy

(2) mv /etc/seedit/policy/extras/*name of domain*.sp /etc/seedit/policy/*name of domain*.sp

(3) seedit-load

File size of generated policy will be smaller than previous method.

6.2 Command line

You can also do it from command line.

6.2.1 Use boolean

If you know SELinux boolean, it's very easy. For example, confined domain name is httpd_t, then turn on httpd_disable_trans boolean and restart daemon.

```
Example:
# setsebool -P httpd_disable_trans 1
# /etc/init.d/httpd restart
# seedit-unconfined -e
Current SELinux mode: enforcing
PID     Comm    Domain
1111     httpd    Unconfined(initrc_t)
```

You can confine again by turning off boolean

```
Example:
# setsebool -P httpd_disable_trans 0
# /etc/init.d/httpd restart
# seedit-unconfined -e
Current SELinux mode: enforcing
PID     Comm    Domain
1111     httpd   Confined by httpd_t domain
```

6.2.2 Remove config file

Config file of domain is located /etc/seedit/policy/*domain name*.sp. If you remove it and restart application, you can unconfine application. See example below.

```
Example:
# cd /etc/seedit/policy
# mkdir unused
# mv httpd_t.sp unused
# seedit-load
# /etc/init.d/httpd restart
# seedit-unconfined -e
Current SELinux mode: enforcing
PID     Comm    Domain
1111     httpd    Unconfined(initrc_t)
```

If you want confine again, place config file to /etc/seedit/policy dir.

```
Example:
# cd /etc/seedit/policy
# mv unused/httpd_t.sp .
```

Figure 6: Typical example of policy: Policy for Apache Web server

```
 1  {
 2  domain httpd_t;
 3  program /usr/sbin/httpd;
 4  include common-relaxed.sp;
 5  include daemon.sp;
 6  include nameservice.sp;
 7  allow /var/www/** r,s;
 8  allow /var/log/httpd/** r,a,s;
 9  allow /etc s;
...<snip>..
10  allownet -protocol tcp -port 80,443 server;
11  allowpriv netlink;
12  }
```

```
# seedit-load
# /etc/init.d/httpd restart
# seedit-unconfined -e
...
```

7 Simplified Policy basics

7.1 Where is simplified policy?

Our simplified policy is located at /etc/seedit/policy directory. In the directory, files named *domain name*.sp are located.

7.2 Policy syntax overview

Simplified policy is described in syntax called Simplified Policy Description Language(SPDL). For detail, see other document(SPDL specification). You do not have to understand its full specification, because there is a helper tool when you describe policy. But it is better for you to be able to understand policy.
Let's see overview of SPDL by looking at example. Figure 6 shows policy for Apache web server.

7.2.1 Give domain to application

Line 2 and 3 is configuration that gives domain to application . Line 2 names domain as httpd_t. Access rights for the domain is configured. By default domain has *no* access rights, by configuring to allow access to resources, domains can access resources.

line 3 means we will give httpd_t domain to /usr/sbin/httpd. By them, when /usr/sbin/httpd is executed, it is confined by httpd_t domain.

- Note to experts
  To give domain we are using SELinux's domain transition mechanism. By line 2 and 3, domain transition from unconfined domain(such as initrc_t) to httpd_t, and entry point is /usr/sbin/httpd(which is labeled automatically by SPDL compiler). So you have to notice that domain transition does not happen from confined domain.

7.2.2 Import typical configuration

By line 4,5,6 configurations common to applications are imported. To see what is imported, see files in include directory. For example, *include include nameservice.sp;*, configuration that is described in include/nameservice.sp is imported. Read access to files such as /etc/hosts are allowed.

7.2.3 Allow access to file

Line 7-10 allows access to files. File name and permissions are described. For filename, you can use grab like below.

```
directory/* : means files under directory, not include subdirectory.
directory/**: means files under directory, including files under
subdirectory.
```

File name that starts with ~ represents home directory(Not including /root).

```
~/public_html/**
```

means public_html directories under each user's home directories(except /root).

You can specify following permissions.

- Basic permissions
  - s
    Search.Permission to search file tree.i.e. Read file name list in directory. For file, it means nothing.
  - r
    Read files.
  - x
    Execute files.
  - w
    Write. This includes write,append,create,delete files.
- Detailed permissions
  w permission is allowing too much, if you want more security, w permission can be splitted into 5.

  - a
    Append.
  - o
    Overwrite. This means, write open file.
  - c
    Create. Create files.
  - e
    Erase. Delete files.
  - t
    Setattr.Modify file attribute(not including file security attribute).

Now you can understand line 7-9.

- Line7: http_t can see file lists and read all files under /var/www, including sub-directories.
- Line 8: http_t can see file lists, read/append all files under /var/log/httpd, including sub-directories.
- Line 9: httpd_t can see file list in /etc. Can not do anything files in /etc directory. Because /etc/* is not described.

7.2.4 Allow access to network

Access to network can be described by simplified policy. By line 10, httpd_t is allowed to behave as a server using tcp 80,443 port.
If you want to allow httpd_t to connect MySQL(TCP 3306), you have to specify following.
allownet -protocol tcp -port 3306 client;
You can use -1023, 1024-, and * for port number. -1023 means all wellknown ports(excepts ports used by other domains). 1024- means all ports(except ports used by other domains) over 1024, and * means all port number.

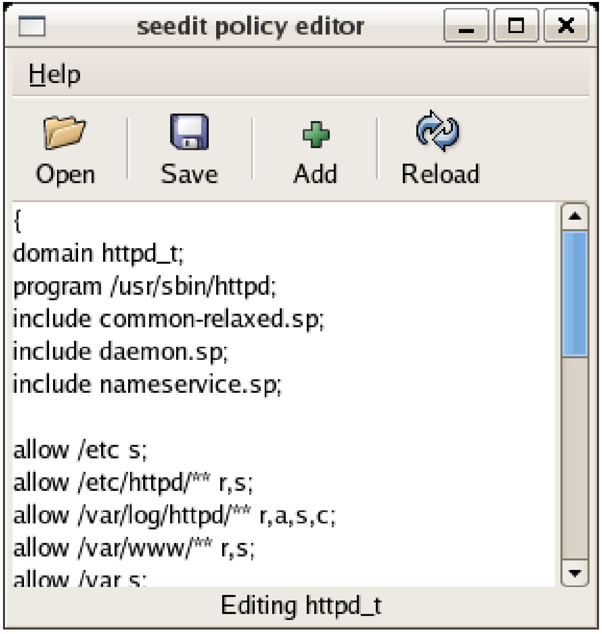
7.2.5 Allow other privilege

Other important operations not related to files and networks are restricted by SELinux. You can use *allowpriv name_of_privilege;*. For example, usage of netlink socket(it is used to communicate with kernel) is allowed in line 11.

7.3 GUI Editor

By selecting *Edit Policy* from control panel, you can launch policy Editor. From *Open* icon, you can select domain. For example, when you open httpd_t domain, you can see like figure 7. You can edit policy like text editor.
By *Save* button, you can save it and apply edited policy, policy load is automatically done.
Reload button read configuration file for domain again, it is useful when you

Figure 7: GUI Editor



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

allow /etc s;
allow /etc/httpd/** r,s;
allow /var/log/httpd/** r,a,s,c;
allow /var/www/** r,s;
allow /var s;
```

Editing httpd_t

edit policy from another tool(such as generator).
By *Add* button, you can insert policy at the end of file from GUI selection window. You can see window like Figure 8 and 9. From file tab, you can insert configuration related to file, in the example of e Figure 8, after pressing *Add*,

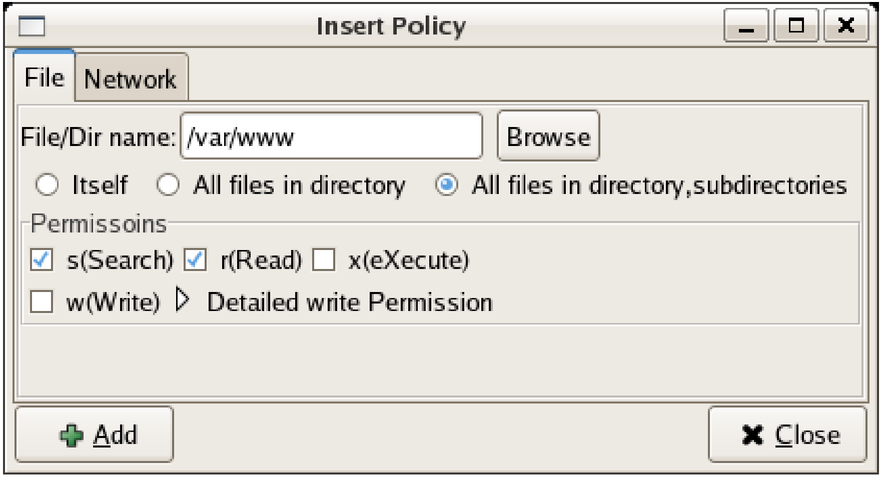
```
allow /var/www/** r,s;
```

will be inserted. From network tab, you can insert configuration related to file, in the example of e Figure 9, after pressing *Add*,

```
allownet -protocol tcp -port 80 server;
```

will be inserted.

Figure 8:



File/Dir name: /var/www

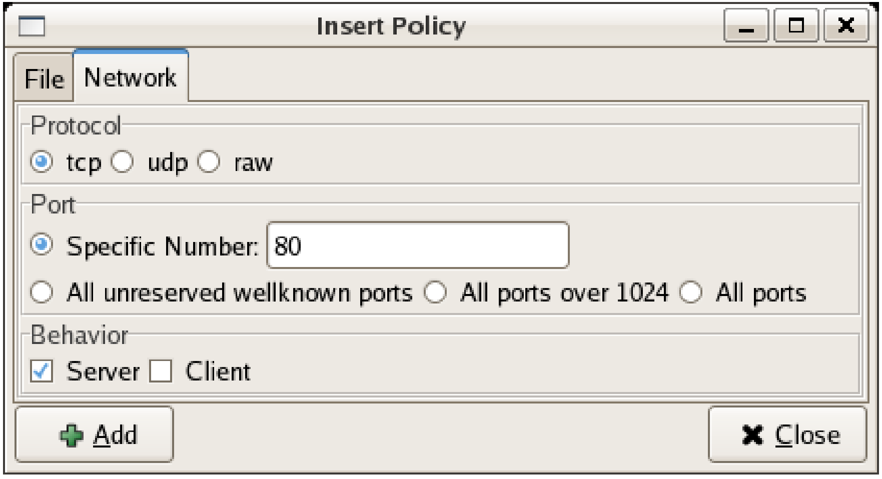
○ Itself ○ All files in directory ● All files in directory,subdirectories

Permissoins

☑ s(Search) ☑ r(Read) □ x(eXecute)

□ w(Write) ▷ Detailed write Permission

Figure 9:



Protocol

● tcp ○ udp ○ raw

Port

● Specific Number: 80

○ All unreserved wellknown ports ○ All ports over 1024 ○ All ports

Behavior

☑ Server □ Client

8 Add policy by policy generation tool

8.1 Test in permissive mode

If you find confined application run due to SELinux denial, you have to add policy.
Before trying to add policy, test in permissive mode. If application run in permissive mode, it is highly possible that SELinux is denying some access. You have to add policy. We have GUI and command line utility.

8.2 How policy generator works?

Policy is generated from SELinux access log. In test in permissive mode, access log is obtained. In permissive mode, if access is denied by policy, it only takes log. To make application work, you have to allow denied access.

Following is example of access denial log in audit.log. By this example, how policy generator generates policy is shown.

```
----
time->Wed Apr 26 18:34:32 2006
1: type=PATH msg=audit(1146090872.442:29): item=0
name="/etc/vsftpd/vsftpd.conf" flags=101  inode=584775 dev=03:03
mode=0100600 ouid=0 ogid=0 rdev=00:00
type=CWD msg=audit(1146090872.442:29):
cwd="/etc/selinux/seedit/src/policy/simplified_policy"
2: type=SYSCALL msg=audit(1146090872.442:29): arch=40000003
syscall=5 success=yes exit=3 a0=bfb04c52 a1=8800 a2=0 a3=8800
items=1 pid=13151 auid=4294967295 uid=0 gid=0 euid=0 suid=0
fsuid=0 egid=0 sgid=0 fsgid=0 comm="vsftpd" exe="/usr/sbin/vsftpd"
3: type=AVC msg=audit(1146090872.442:29): avc:  denied  { read }
for  pid=13151 comm="vsftpd" name="vsftpd.conf" dev=hda3
ino=584775 scontext=user_u:system_r:ftpd_t
tcontext=system_u:object_r:default_t tclass=file
----
```

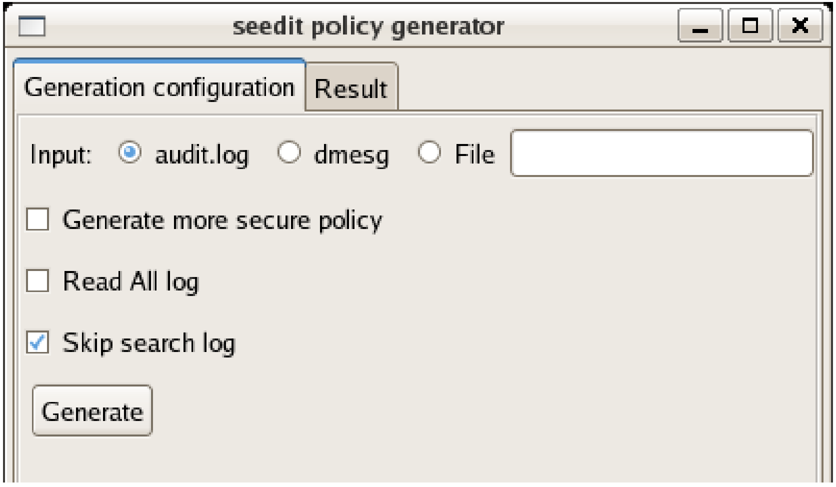
Line 3 means, *read access by ftpd_t domain to filename vsftpd.conf is denied.* From line 3, we can say that following should be added.

```
allow vsftpd.conf r;
```

However, full path for vsftpd.conf is not known. To obtain full path we use log in line 1. It says full path for vsftpd.conf is /etc/vsftpd/vsftpd.conf. By combining line 1 and 3 we can conclude that following should be added.

```
allow /etc/vsftpd/vsftpd.conf r;
```

Figure 10: Policy Generate tool



Input: ● audit.log ○ dmesg ○ File

□ Generate more secure policy

□ Read All log

☑ Skip search log

If you are not running auditd service log in line 1 is not obtained. In other words, full path information is not included in log. That is why we recommend to run auditd in using SELinux Policy Editor.
When auditd is not running policy generator tries to obtain full path by locate command, but it often fails.

8.3 GUI(policy generator)

8.3.1 Launch tool

Select *Generate Policy* from control panel, window like figure 10 will open.

Usually, you do not have to configure nothing. Press *Generate policy* button. What can be configured is shown below for reference.

- Input
  This is input source of SELinux access log. If auditd service is running, choose audit.log. auditd service is not running choose dmesg.

- Generate more secure policy
  If checked, generator trys to generate more secure policy. In current version, it trys to use detailed file write permission(a,o,c,e,t). If not checked, those permissions are not used, instead w permission is used in generated policy.

- Read All log
  When checked, read all message from log input. If not input is read only from last policy load(the place where load_policy permission is granted).

- Skip search log
  When checked, access denial for dir:search permission is skipped. dir:search often lead to unwanted output.

8.3.2 Examine result and add policy

After pressing *Generate policy* button, policy is generated. It takes some time. When finished, result is outputted in Result tab. Figure 11 is example output.

First row is asking, do you want to add *allownet -protocol tcp -port 1024-server;* to vsftpd_t domain? And Log is access denial of SELinux, from that log policy is generated. If you want to add policy, check check box.
Glob button is very useful.You can allow access to all files in directories. For example, select row *allow /etc/vsftpd/vsftpd.conf r,s;* After clicking *Glob* button, the filename changes like below.

```
/etc/vsftpd/vsftpd.conf ->
/etc/vsftpd/* ->
/etc/vsftpd/** ->
/etc/* ->
/etc/** ->
/* ->
/** ->
```

By *Undo Glob*, filename get back to previous one.
Policy that is going to be added is displayed in *Following will be saved*, like figure 12. If it looks good, press *Save and Apply* button. The policy is added to domain, and policy is loaded to kernel.

8.4 Command line(audit2spdl)

You can add policy by audit2spdl command. The usage is easy. When auditd is running

```
# audit2spdl -al
```

When auditd is not running,

```
# audit2spdl -dl
```

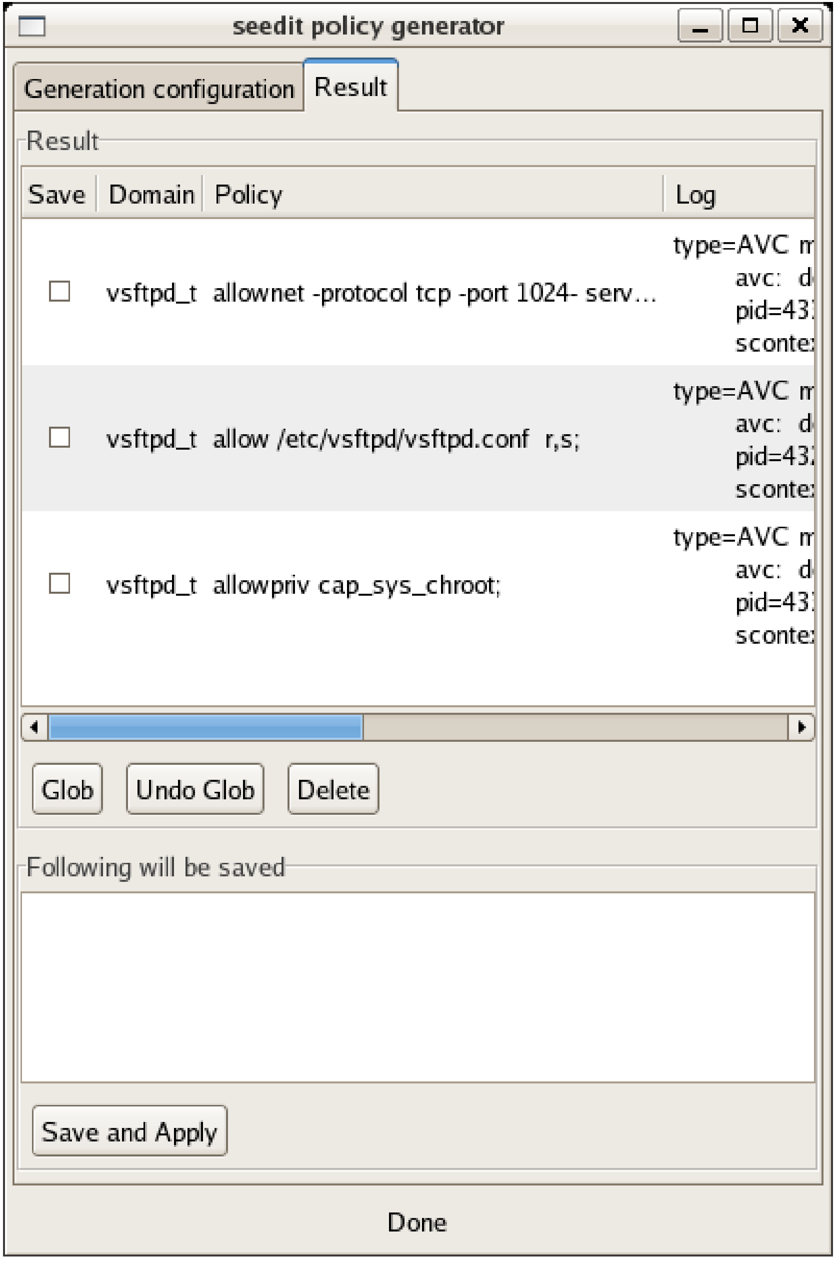
You can read log by specifying filename,

```
# audit2spdl -l -i /var/log/messages
```

This command translate SELinux log denial to simplified policy. It is recommended to use auditd service if it is prepared for your distro. For Fedora Core 5, you can install auditd by following commands.

```
#yum install audit
#chkconfig auditd on
#/etc/init.d/auditd start
```

Figure 11: Policy Generate result



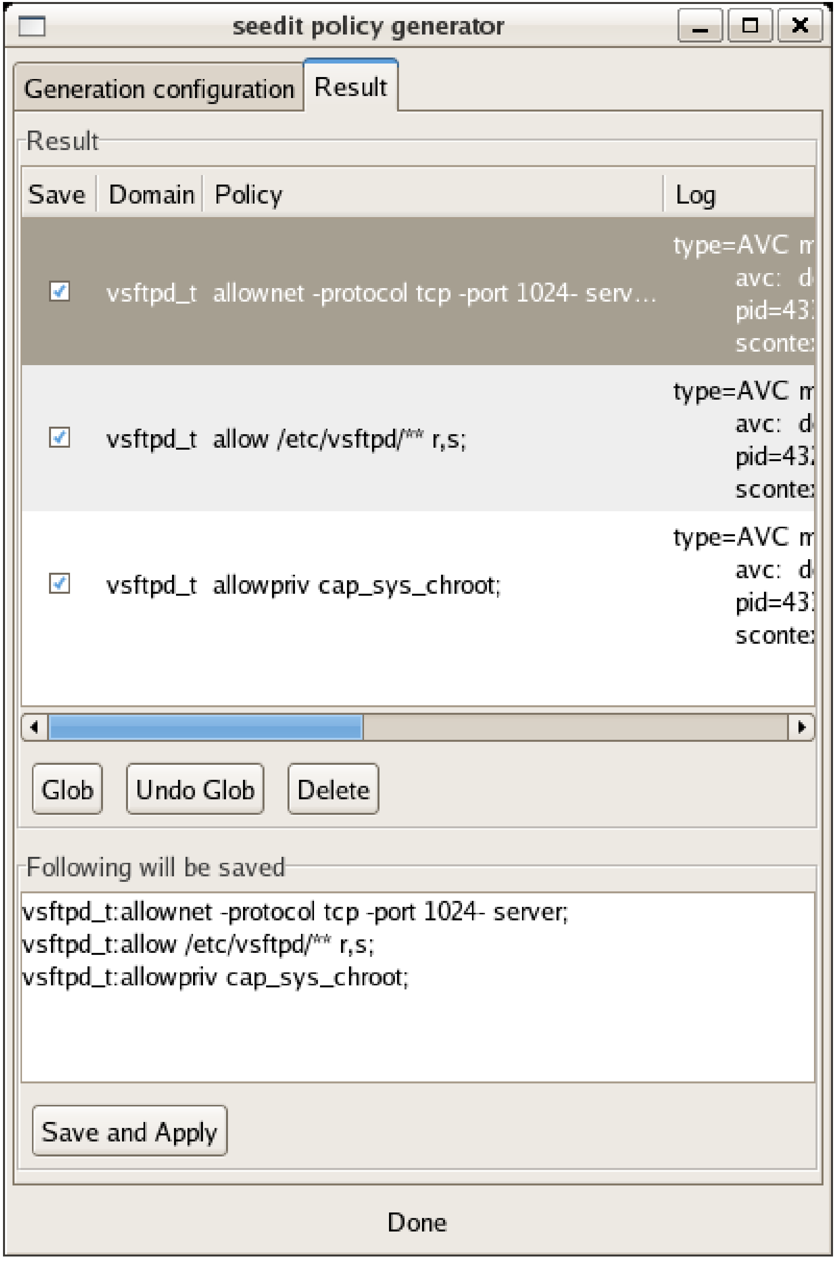
Save	Domain	Policy	Log

□	vsftpd_t	allownet -protocol tcp -port 1024- serv...	type=AVC m
			avc: d
			pid=43
			sconte

□	vsftpd_t	allow /etc/vsftpd/vsftpd.conf r,s;	type=AVC m
			avc: d
			pid=43
			sconte

□	vsftpd_t	allowpriv cap_sys_chroot;	type=AVC m
			avc: d
			pid=43
			sconte

Figure 12: Before save



Save	Domain	Policy	Log

☑	vsftpd_t	allownet -protocol tcp -port 1024- serv…	type=AVC m
			avc: d
			pid=43
			sconte

☑	vsftpd_t	allow /etc/vsftpd/** r,s;	type=AVC m
			avc: d
			pid=43
			sconte

☑	vsftpd_t	allowpriv cap_sys_chroot;	type=AVC m
			avc: d
			pid=43
			sconte

```
vsftpd_t:allownet -protocol tcp -port 1024- server;
vsftpd_t:allow /etc/vsftpd/** r,s;
vsftpd_t:allowpriv cap_sys_chroot;
```

Following is sample output.

```
#audit2spdl -dl
.... It takes some time...
-------------------------
#SELinux deny log:
audit(1146162965.963:16): avc:  denied  { read } for  pid=6653
comm="vsftpd" name="vsftpd.conf" dev=hda3 ino=584775
scontext=user_u:system_r:ftpd_t
tcontext=system_u:object_r:default_t tclass=file
#Suggested configuration
File ftpd_t.sp:
allow /etc/vsftpd/vsftpd.conf  r;
-------------------------
...
```

Above says you have to add

```
allow /etc/vsftpd/vsftpd.conf  r;
```

to ftpd_t.sp to resolve SELinux access denial.

To add generated policy, in above example, open /etc/seedit/policy/vsftpd.sp. and add allow /etc/vsftpd/vsftpd.conf r; between , like below

```
{
domain vsftpd\_t
program /usr/sbin/program;
allow ....
<add here!!>

}
```

After adding file, you have to notice it to SELinux kernel. Type seedit-load.

```
#seedit-load
seedit-load: Success
```

You can see progress of seedit-load by -v option like below

```
# seedit-load -v
mkdir -p ./sepolicy;
m4 -s ./simplified_policy/*.sp >./simplified_policy/all.sp;
/usr/bin/seedit-converter -i ./simplified_policy/all.sp -o
./sepolicy -b ./base_policy -I ./simplified_policy/include ;
................

cp /etc/selinux/seedit/contexts/files/file_contexts.all
/etc/selinux/seedit/contexts/files/file_contexts.all.old
seedit-load: Success
```

In above case, you can add

```
allow /etc/vsftpd/* r;.
```

It is a little different from that suggested by audit2spdl. Because system administrate knows /etc/vsftpd is vsftp's configuration directory, so it is more effective to allow access whole /etc/vsftpd directory.

8.4.1 Advanced topic:Notice about audit2spdl

(1) Not best security
audit2spdl not always suggest best solution. You have to review suggested configuration carefully. For example, audit2spdl does not generate detailed file permission(o,a,c,e,t), but generate w.
To generate permission o,a,c,e,t, use s option, like audit2spdl -dls .

(2) Fail to suggest
In special case, suggesting configuration fails, following message appear.

```
#Failed to generate, because failed to obtain fullpath.
```

In SELinux log, full-path is not contained. To obtain full path audit2spdl is doing some works, but it sometimes fail. To obtain full-path in all cases, you have to use auditd. auditd is not installed in Fedora Core 5. You can use auditd like below.

```
#yum install audit
#chkconfig auditd on
#/etc/init.d/auditd start
```

The use audit2spdl -al, and try again. It is also faster than audit2spdl -dl.

9 Creating domain

Let's see by example to create domain ftpd_t for vsftpd, and confine behavior of vsftpd. By following this example you'll be able to prepare domain for other programs. Generally, process of creating domain is following.

(1) Create template

(2) Check domain

(3) Test run and add policy

9.1 Create domain from GUI

We will create vsftp_t domain, and configure the domain to work as Anonymous FTP server.

9.1.1 Create template

By Domain manager GUI you can create template configuration. From control panel, select *Manage Domain*. Let's assign vsftpd vsftpd_t domain.

You can do it like fig 13. First, specify name of executable file. Enter */usr/sbin/vsftpd*.
Next, specify domain name. Enter *vsftpd_t*.
There are some questions you have to answer.
Then press *Create Template* button. You will see created configuration is shown in *Created template*.

If you know about application, you already know what kind of access rights are necessary. You can add it now. If you do not know, do not worry. You can generate policy later.
You will prepare anonymous FTP server,so read access to /var/ftp is necessary.
And you need access right to TCP 21 port to behave as FTP server.
You can add configuration from GUI. Press *Add policy* button. Figure 14, you are configuring to allow read access under /var/ftp. Figure 15, you are configuring to allow usage of tcp port 21. By pressing *Add* button, configuration

```
allow  /var/ftp/** r,s;
allownet -protocol tcp -port 21 server;
```

is added. After adding configuration, press *Save and Apply* button.

9.1.2 Check domain

Switch to permissive mode(you can do by setenforce 0 or status GUI). Start vsftpd and make sure its domain is vsftpd_t(You can do it by status GUI)

9.1.3 Test run and add policy

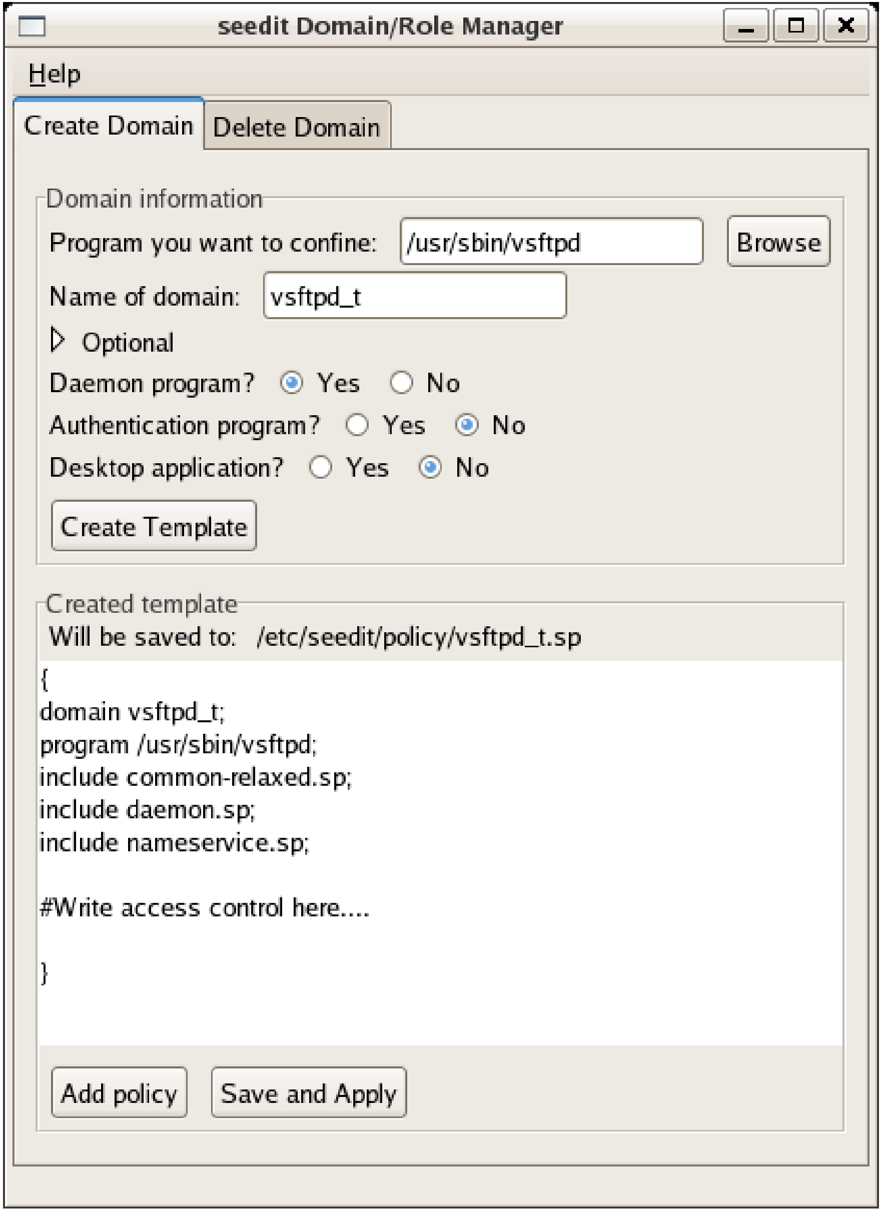
Test run vsftpd in permissive mode.
Let's test ftp like following.

```
$ ftp localhost
Name (localhost:ynakam): Anonymous
Password: <anything is OK>
...
and do something..
```

And see log.

```
#auserch -m AVC
```

Figure 13: Create new domain



Program you want to confine: /usr/sbin/vsftpd

Name of domain: vsftpd_t

Daemon program? ● Yes ○ No

Authentication program? ○ Yes ● No

Desktop application? ○ Yes ● No

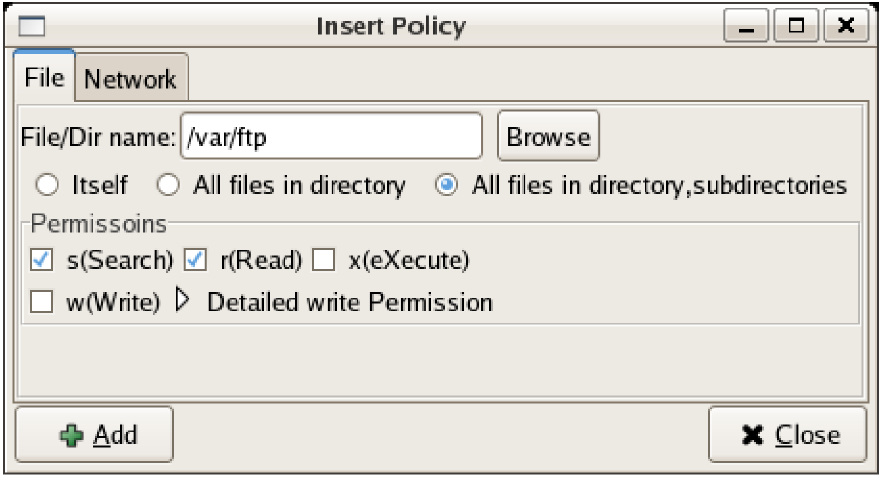
Will be saved to: /etc/seedit/policy/vsftpd_t.sp

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

#Write access control here....

}
```

Figure 14: Insert file access control policy



File/Dir name: /var/ftp

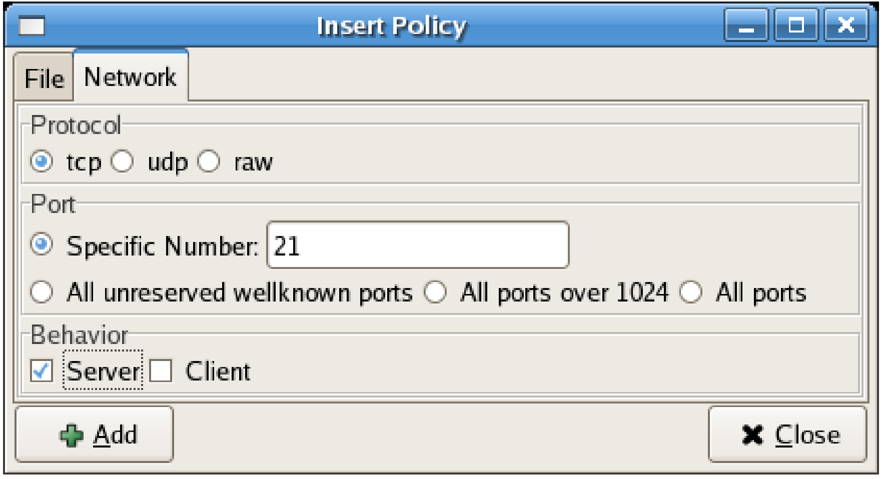
○ Itself ○ All files in directory ● All files in directory,subdirectories

Permissoins

☑ s(Search) ☑ r(Read) □ x(eXecute)

□ w(Write) ▷ Detailed write Permission

Figure 15: Insert network access control policy



Protocol

● tcp ○ udp ○ raw

Port

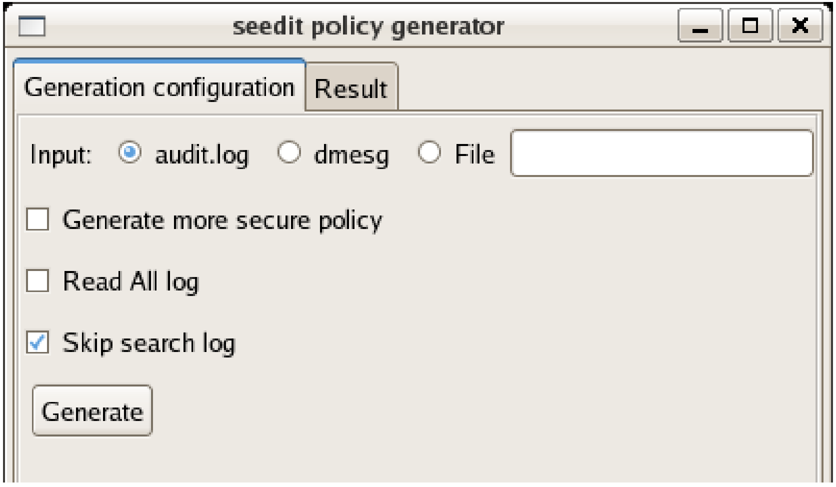
● Specific Number: 21

○ All unreserved wellknown ports ○ All ports over 1024 ○ All ports

Behavior

☑ Server □ Client

Figure 16: Policy Generate tool



Input: ● audit.log ○ dmesg ○ File

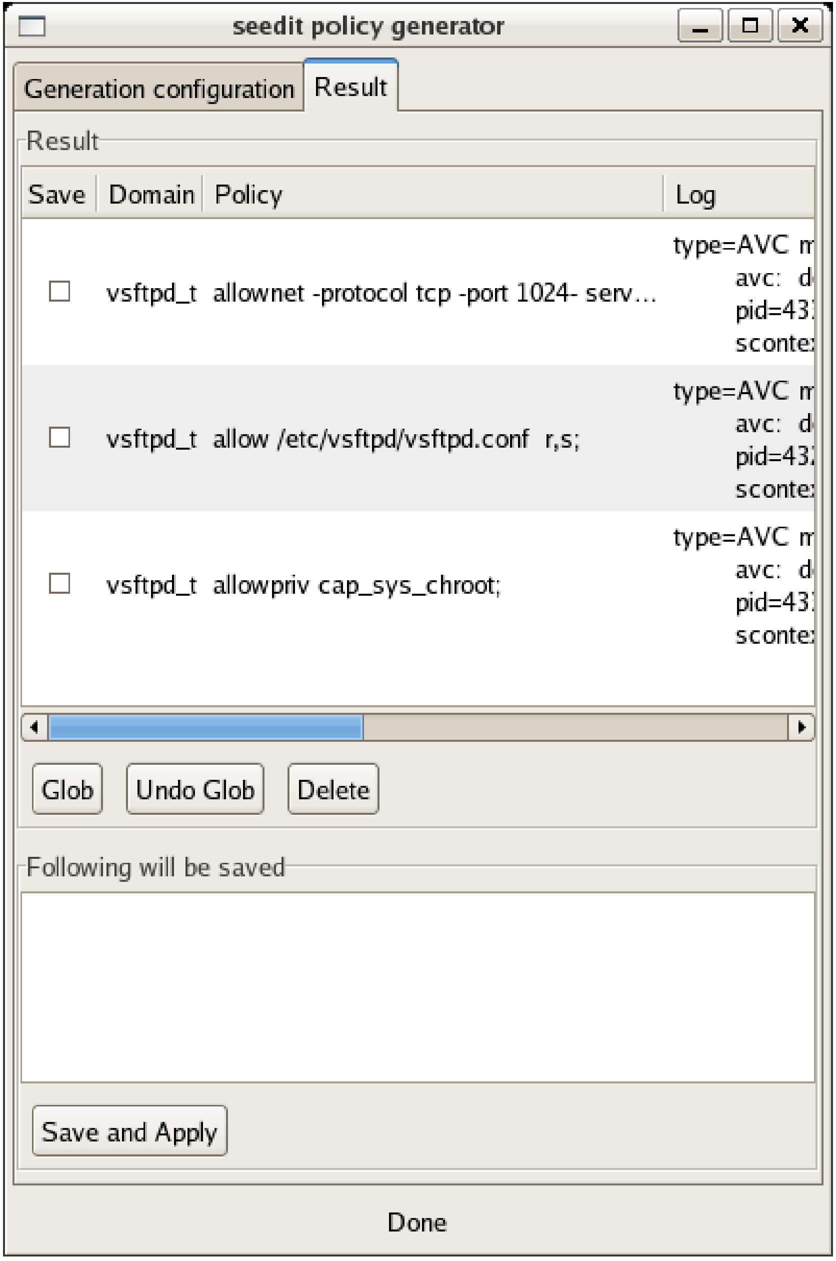
□ Generate more secure policy

□ Read All log

☑ Skip search log

Various accesses are denied. So you have to add policy that allows denied access. You can do it easily by policy generation tool. Select *Generate Policy* from control panel, window like figure 16 will open. Usually, you do not have to change configuration, if you want more secure configuration, check *Generate more secure policy*, it will generate policy that uses detailed write permission, but you will need more time to complete configuration. To generate policy press *generate* button. Then result is shown like figure 17. If you want to add generated policy, check check boxes, and if you want to allow to directory, use Glob button. In this case, it is more efficient to allow access to /etc/vsftpd directory, so click Glob button twice.
Filename changes /etc/vsftpd/vsftpd.conf → /etc/vsftpd/* → /etc/vsftpd/**.
In this case, configuration like 18 will be saved. Click *Save and Apply* button.
Test vsftpd again and use Policy Generation tool again, if you see nothing, then test in Enforcing mode. If it works, all done!. If not, add configuration by Policy Generation tool until it works.

Figure 17: Policy Generate result



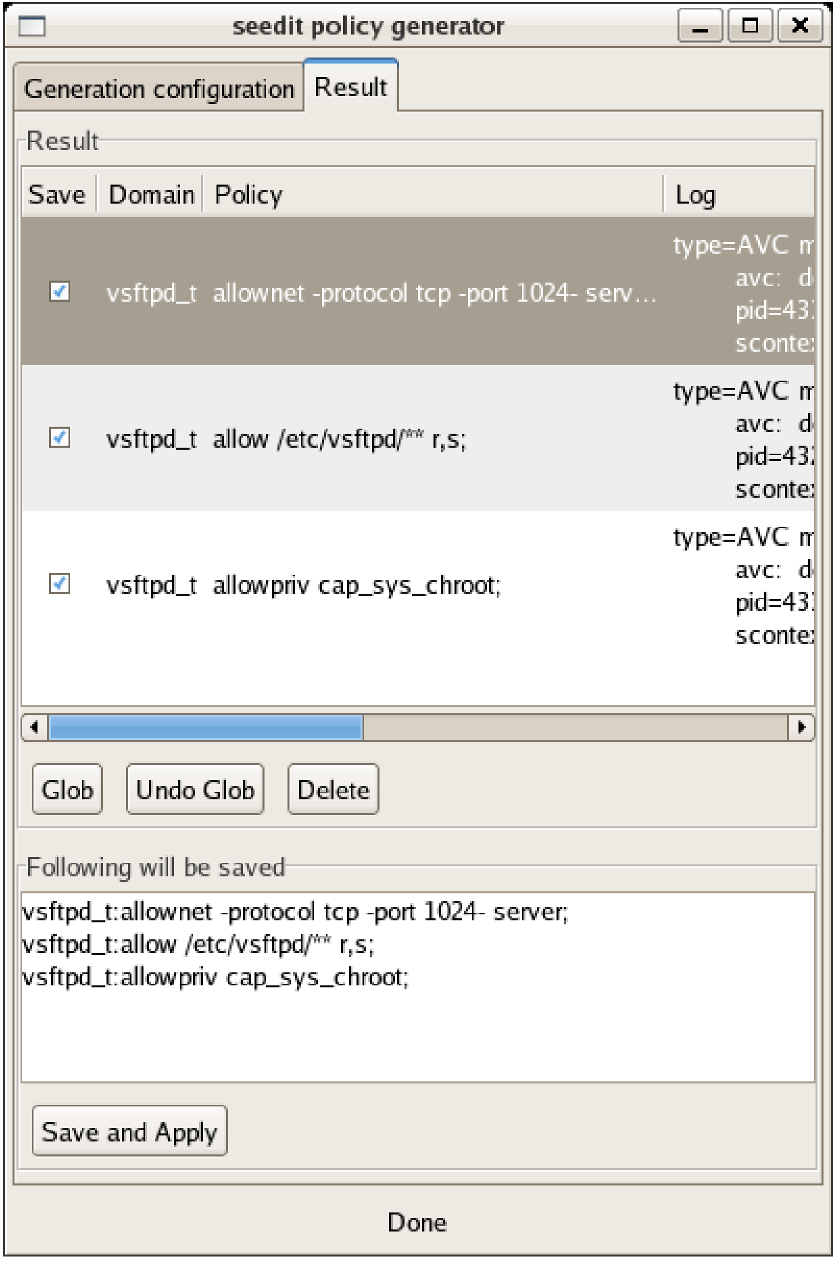
Save	Domain	Policy	Log

□	vsftpd_t	allownet -protocol tcp -port 1024- serv…	type=AVC m
			avc: d
			pid=43
			sconte

□	vsftpd_t	allow /etc/vsftpd/vsftpd.conf r,s;	type=AVC m
			avc: d
			pid=43
			sconte

□	vsftpd_t	allowpriv cap_sys_chroot;	type=AVC m
			avc: d
			pid=43
			sconte

Figure 18: Before save



Save	Domain	Policy	Log

☑	vsftpd_t	allownet -protocol tcp -port 1024- serv…	type=AVC m
			avc: d
			pid=43
			sconte

☑	vsftpd_t	allow /etc/vsftpd/** r,s;	type=AVC m
			avc: d
			pid=43
			sconte

☑	vsftpd_t	allowpriv cap_sys_chroot;	type=AVC m
			avc: d
			pid=43
			sconte

```
vsftpd_t:allownet -protocol tcp -port 1024- server;
vsftpd_t:allow /etc/vsftpd/** r,s;
vsftpd_t:allowpriv cap_sys_chroot;
```

9.2 Create domain from command line

9.2.1 Create template

You can create template configuration by seedit-template command. Usage is following.

```
seedit-template -d <domain> -e <path to application> -o <output>
```

Following is output for us.

```
# seedit-template -d vsftpd_t -e /usr/sbin/vsftpd
{
domain ftpd_t;
program /usr/sbin/vsftpd;
include common-relaxed.sp;
include daemon.sp;
include nameservice.sp;
}
```

Template configuration is generated. In this, domain is named, and given to /usr/sbin/vsftpd. By *include* , access rights commonly used for daemons are imported. You have to save this configuration under /etc/seedit/policy/ftpd_t.sp. Note that file name must be *domain-name*.sp, otherwise seedit-load fails.
Save above configuration to /etc/seedit/policy/ftpd_t.sp.

9.2.2 Check domain

Load simplified policy by following.

```
#seedit-load
```

And switch to permissive mode, because vsftpd will fail to work in enforcing mode due to SELinux access denial.

```
#setenforce 0
#getenforce
Permissive
```

Start vsftpd and check domain of vsftpd by seedit-unconfined.

```
# /etc/init.d/vsftpd restart
# seedit-unconfined -e
10530   vsftpd  Confined by ftpd_t
```

You will see vsftpd is running as ftpd_t as above.

9.2.3 Test run and add policy

Let's run application in permissive mode, and find out what access is denied, and allow such access by audit2spdl. In this article, we aim to develop policy for vsftpd to work as Anonymous FTP server. Let's test ftp like following.

```
$ ftp localhost
Name (localhost:ynakam): Anonymous
Password: <anything is OK>
...
and do something..
```

And see log.

```
#dmesg
```

Various accesses are denied. Let's audit2spdl.

```
#audit2spdl -dl
-------------------------
#SELinux deny log:
audit(1146179470.043:86): avc:  denied  { search } for
pid=10904 comm="vsftpd" name="vsftpd" dev=hda3 ino=584772
scontext=user_u:system_r:ftpd_t
tcontext=system_u:object_r:etc_t tclass=dir
#Suggested configuration
File ftpd_t.sp:
allow /etc/vsftpd  s;
-------------------------

-------------------------
#SELinux deny log:
audit(1146179477.891:108): avc:  denied  { search } for
pid=10911 comm="vsftpd" name="ftp" dev=hda3 ino=163477
 scontext=user_u:system_r:ftpd_t
tcontext=system_u:object_r:var_t tclass=dir
#Suggested configuration
File ftpd_t.sp:
allow /var/ftp  s;
-------------------------
.....
```

audit2spdl tells us various configuration should be added. You might find following.

```
-------------------------
#SELinux deny log:
type=AVC msg=audit(1148486754.718:36): avc:  denied  { lock } for
pid=11763 comm="vsftpd" name="test.txt" dev=hda3 ino=163311
```

```
scontext=user_u:system_r:ftpd_t tcontext=system_u:object_r:default_t
 tclass=file
#Suggested configuration
File ftpd_t.sp:
#Failed to generate, because failed to obtain fullpath.
#allow test.txt  r,s;
-------------------------
```

It means failed to suggest configuration because audit2spdl failed to guess fullpath for test.txt. But you know fullpath of test.txt is /var/ftp/pub/test.txt. So, you will add allow /var/ftp/pub/test.txt r,s;

In my case, following are suggested at first test.

```
allow /etc/vsftpd  s;
allow /var/ftp  s;
allow /root  s;
allownet -protocol tcp -port 21 server;
allowpriv cap_sys_chroot;
allow /var/log/xferlog  r,w;
allow /etc/vsftpd/vsftpd.conf  r;
allow /etc/vsftpd/vsftpd.conf  s;
```

You can add them, but you will notice it is more effective To allow r,s to /var/ftp and /etc/vsftpd. So, your ftpd_t.sp will be following.

```
{
domain ftpd_t;
program /usr/sbin/vsftpd;
include common-relaxed.sp;
include daemon.sp;
include nameservice.sp;
# added by audit2spdl suggestion
allow /etc/vsftpd/**  r,s;
allow /var/ftp/**  r,s;
allow /var/log/xferlog  r,w;
allow /root  s;
allownet -protocol tcp -port 21 server;
allowpriv cap_sys_chroot;
}
```

After seedit-load, test vsftpd again. You will find access denial again. By audit2spdl you will find like following.

```
allownet -protocol tcp -port 6353 server;
```

This says ftpd is trying to behave server using 6553 tcp. But the port number varies time to time.So it seems ftpd requires to use all port over 1024. You can add following.

```
allownet -protocol tcp -port 1024- server;
```

Then, test again and add policy until no access denial is outputted. At last, switch to enforcing mode.

```
#setenforce 1
```

Test vsftpd again.

SPDL has advanced feature to configure more secure policy. For example, login user is not confined by default, to enhance login user security, you can use RBAC feature. About RBAC see RBAC guide.

10 Other notices

- Notice about file move
  When you move file to another directory by mv command, you have to use restorecon command. Like below.

  ```
  * Example: Upload homepage
  # pwd
  /root/homepage/index.html
  # mv index.html /var/www/html
  # restorecon -R /var/www/html
  ```

  If you forget last restorecon command, Apache will not be able to access /var/www/html/index.html, even if allow /var/www/** is described for httpd_t.sp. This is because, access right is inherited from source file in mv command. In this example, httpd_t domain can not access under /root, but can read under /var/www. After mv command, access rights to /var/www/html/index.html is the same as /root/homepage/index.html.
  To fix this situation you have to use restorecon command.
  GUI policy generator sometimes suggests restorecon.

- File creation
  Newly created file inherits access rights from directory. See example below. Assume following configuration exists.

  ```
  domain foo_t;
  allow /foo/bar/** r,s;
  allow /foo/bar/test.txt r,w,s;
  ```

  foo_t can read files under /foo/bar, and write /foo/bar/test.txt. If /foo/bar/test.txt does not exist at the time of configuration, and after configuration /foo/bar/test.txt is created, foo_t can *read* test.txt, because newly created test.txt inherits allow /foo/bar/**. This is confusing, but limitation of our implementation.
  To fix this, you have to do

```
restorecon -R /foo/bar
```

Then foo_t can read/write test.txt.

- Why restorecon is necessary?
  SELinux internally identifies resources by label called *type*. When mv command, label is preserved.When file is newly created, type is inherited from belonging directory. We have to fix relationship between file and type by restorecon command.

- cron jobs
  Cron jobs are not confined. If you want to protect cron job, edit system_crond_t.sp, delete *allowpriv all;*. However, configuring cron jobs correctly will be very difficult.

- Dynamically created/deleted files
  For files that are dynamically created/deleted, access control sometimes do not work well. See example below.

  ```
  domain foo_t;
  allow /foo/bar/** r,s;
  allow /foo/bar/test.txt r,w,s;
  ```

  In this, if test.txt is deleted, and created, test.txt inherits access right of directory. So, foo_t can read test.txt but can not write test.txt. You can fix by restorecon, but when it is re-created, you have to do restorecon again..
  Therefore, for files whose access right is different from directory, and they are deleted/created, access control does not work well. To resolve this, you have following solutions.

  (1) Give up controlling access for individual files
  you can write
  allow /foo/var/** r,s,*w*. By configuring above, if test.txt is deleted and created, foo_t can write test.txt, however foo_t can write other files under /foo/bar directory.

  (2) Use allowtmp
  To protect such temporally files, SPDL supports allowtmp statement. You can configure like following.

  ```
          allowtmp /foo/bar -name auto  r,w,s;
  ```

  By allowtmp statement, file can be identified by label in SPDL. This allowtmp statement means, files created under /foo/bar is labeled as foo_foo_bar_t(-name auto generates label name based on domain and directory name, foo_t + /foo/bar = foo_foo_bar_t). And foo_t can read, write files that have foo_foo_bar_t label. By above, when test.txt

is deleted and created again, test.txt is given label(foo_foo_bar_t) and identified by label. In allowtmp, file type transition in SELinux is internally used.
If you want to access test.txt from other domain, you have to specify label not filename, like below.

```
allow foo_foo_bar_t r;
```

In default policy,allowtmp is used to control access to /etc/mtab and temporally files under /tmp, /var/tmp.

- Device files other than /dev
Devices have critical impact to security, so it is treated specially. In default policy, device is assumed to exist under /dev. If you write allow statement for devices other than /dev directory, you can not access it. If you want to access devices other than /dev, you have to write allowdev statement. If you want to read access devices in /var/chroot/dev/null, you have to write following, before describing allow /var/chroot/dev/null.

```
allowdev -root /var/chroot/dev;
```

- Symbolic link
Configuration to file that contains symbolic link is ignored. For example,
allow /etc/init.d/httpd r;
is ignored(init.d is symbolic link to rc.d/init.d).

- Hardlink
In Linux system, contents of file can be refereed by multiple name using hard link. Hardlink is rarely used recent distro, but you have to note about this if you want to preserve security.
In SPDL, following rule exists about hard link.
If file has multiple hardlink, to access the file, you must specify originally existing file name. Other file names are ignored
For example, /etc/shadow and /var/chroot/etc/shadow is hardlinked, and /etc/shadow exists originally, to access contents of /etc/shadow, you have to use file name /etc/shadow. Configuration using /var/chroot/etc/shadow will be ignored. If some domain(assume foo_t) want to read /var/chroot/etc/shadow, you have to configure *allow /etc/shadow r;*
Next, there is a question, what is criteria of file name *originally* exist? Following is answer.
In following, /etc/shadow and /var/shadow is assumed as hardlinked files.

  (1) If rule is described to one file name, the file name is treated as original.
  Ex: allow /etc/shadow r; is described in some domain, but rules using filename /var/shadow is not described, /etc/shadow is treated as original.

(2) If rules are described to multiple hardlinked file name, the filename that name is the youngest is treated as original
Ex: allow /etc/shadow r, and allow /var/shadow r; are described in some domains, /etc/shadow is treated as original, because /var/shadow > /etc/shadow.

(3) If rules are not described for hardlinked files, the directory names that hardlinks exist are compared. The file whose directory name is oldest is original.
Ex: /etc/shadow, /var/shadow do not appear in any domain. Then /var/shadow is treated as original. Because /var > /etc.

If you are not sure which hardlink is *original*, you can use all names. It means, you can describe

```
allow /etc/shadow r;
allow /var/shadow r;
```

1 of 2 will be ignored, and do no harm.

Above treatment of hardlink is necessary to avoid a kind of *back door* of path name based configuration. Assume hard link to /etc/shadow is created by some trick under /var/www/html, without this behavior, Apache web server can access contents of /etc/shadow via /var/www/html/shadow. To protect this, we must limit way to access hard link to 1.
http://securityblog.org/brindle/2006/04/19 is good reference.

11 Tips

(1) Confining Web applications

By default, CGI will run as httpd_t domain. httpd_t domain is a domain for Apache Web Server, but domain is inherited to child programs by default. If you want to change domain for CGI, you have to use domain_trans element. You can give CGI programs individual domain. If you place your CGI in /var/www/cgi-bin, and give it cgi_t domain, create cgi_t.sp like below.

```
{
domain cgi_t;
domain_trans  httpd_t /var/www/cgi-bin/**;
include common-relaxed.sp;
##### allowxxx will be here...
}
```

For PHP, you *can not* change domain from httpd_t. It is because PHP is internally executed not using exec system call. SELinux can not give domain for such case, unless PHP is extended to use SELinux system call.

(2) deny
You can register important files in black list by *deny* element. Following is example.

```
{
domain foo_t;
deny /etc/shadow;
allow /etc/** r,s;
}
```

In above, foo_t is allowed to all files by allow /etc/**, but can not access /etc/shadow. To access /etc/shadow, you have to write *allow /etc/shadow* explicitly. Some deny elements are written by default in include/common-relaxed.xp