What is SELinux Trying to Tell Me?

The four key causes of SELinux errors

Dan Walsh

Security Enhanced Linux (SELinux) has been available for a few years now, but its error messages still have the ability to confuse administrators and users. This paper will explain the common causes of SELinux errors, and help administrators remedy SELinux problems.

SELINUX ERROR MESSAGES

SELinux error messages are called AVCs, standing for Access Vector Cache. These messages are usually stored by the audit subsystem, in the **/var/log/audit/audit.log** file. If you do not have the audit subsystem enabled, the AVC messages will be sent to **/var/log/messages**. SELinux messages that occur before the audit daemon starts can show up in **dmesg** output. The **setroubleshoot** tool will attempt to translate AVC messages into something more easily understandable. It writes data into **/var/log/messages** and can communicate with the desktop via a GUI.

1: IS THERE A LABELING PROBLEM?

SELinux is all about labels. Every process, file, directory, and device on an SELinux system has a label. If these labels are wrong, SELinux will not function properly. If a file is mislabeled, a confined application might not be allowed access to the mislabeled file. If an executable is mislabeled, it may not transition to the correct label when executing, causing access violations and potentially causing it to mislabel files it creates.

Sometimes an admin or software developer decides to change the the location of files used by a confined domain. For example, if you want to store web pages in a unusual location, such as **/srv/myweb**, you need to tell the SELinux system that these files should be accessible to the web server process. You do this by setting the labeling correctly in the system. In this example, the **httpd** process is allowed to access files labeled with the **httpd_sys_content_t** type. Therefore, you need to set the label for the chosen director. One way to do this is the **chcon** command:

```
# chcon -R -t httpd_sys_content_t /srv/myweb
```

This will set the labels correctly. However, you have not yet told the SELinux system to permanently label these files and directories with this label. In some circumstances, a system relabel could change these labels back to the default. You can use the **semanage** command to make permanent changes to the SELinux system, as shown here:

```
# semanage fcontext -a -t httpd_sys_content_t '/srv/myweb(/.*)?
```

This command tells the SELinux data store that the **/src/myweb** directory and all files under it should be labeled **httpd_sys_content_t**. Tools such as **restorecon** and **rpm** read this data store when they are labeling or relabeling files. Note, however, that the **semanage** command will not change the actual labels on files on your machine. You still need to execute **restorecon** to fix the labels:

```
# restorecon -R /srv/myweb
```

The **restorecon** command reads the SELinux data store to determine how files under **/srv/myweb** should be labeled, and then fixes them.

You can query the system for the default label associated with a particular path using the **matchpathcon** command:.

```
# matchpathcon /srv/myweb
```

The **matchpathcon** command reads the SELinux file context files and prints the default label for the specified path. You can also use **system-config-selinux** to set up your SELinux labeling.

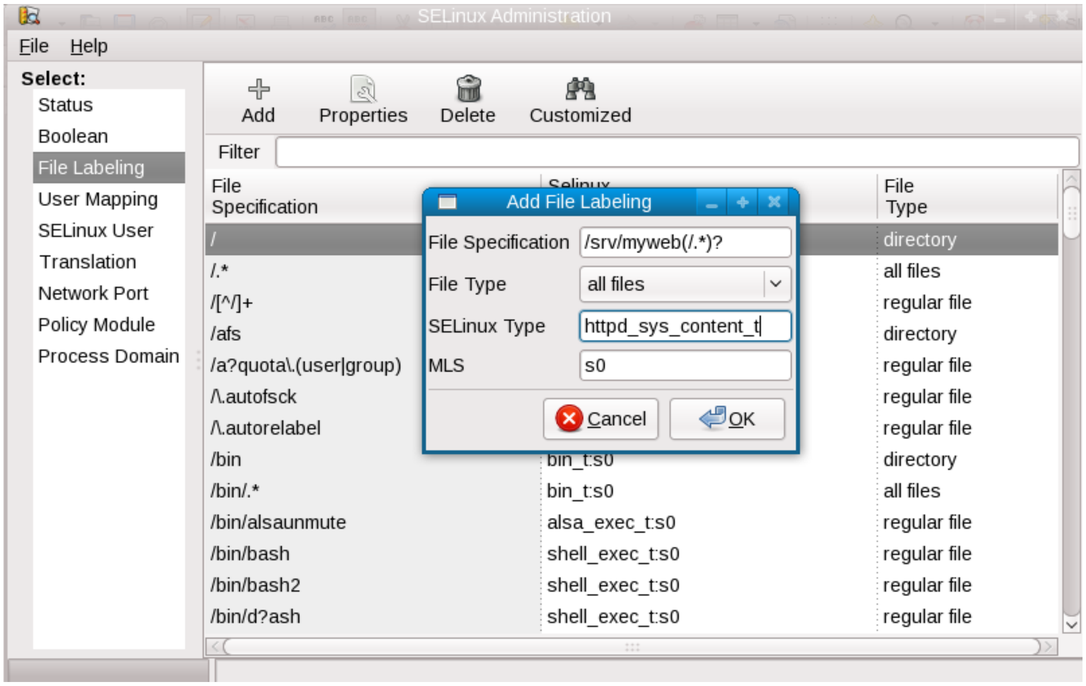
Now, if you create a file under **/srv/myweb**, or copy a file there, the file will be labeled **httpd_sys_content_t** automatically.

NOTE: If you use the **mv** command to move files into this directory, the files could be mislabeled. The **mv** command maintains the file context of the src file, so if you **mv** a file from your home directories, the file would end up being labeled **user_home_t**, and **httpd** would not be able to read it. You would use **restorecon** to fix the label.

In the past, some applications have modified files in their post-install and left them mislabeled. For example, the vmware package copies the **/etc/services** file to the **/tmp** directory. It then adds a port line to it, and moves it back to **/etc/**. Since **rpm** labels files created in **/tmp** during post-install, the file gets labeled **rpm_script_tmp_t**, and the **mv** command maintains the label. Since no confined domains are allowed to read files labeled **rpm_script_tmp_t**, the confined domains start complaining about this label.

You can use the command **restorecon /etc/services** to fix the label. The **restorecond** service is a daemon service that has the ability to watch file creations and maintain their file context. The **restorecond** service reads the **/etc/selinux/restorecond.conf** file for a list of files to watch, and then if those files are created with the wrong context, **restorecond** resets them to the correct context. Fortunately, **restorecond** becomes less important as applications and admins become more SELinux-aware. However, as we move to confined user applications, maintaining the labels in a user's home directory will become more important.

In the SELinux Administrator Tool (**system-config-selinux)**, shown below, you can view and change the SELinux file labeling:



File Specification: /srv/myweb(/.*)?

File Type: all files

SELinux Type: httpd_sys_content_t

MLS: s0

File Specification	Selinux	File Type
/		directory
/.*		all files
/[^/]+		regular file
/afs		directory
/a?quota\.(user|group)		regular file
/\.autofsck		regular file
/\.autorelabel		regular file
/bin	bin_t:s0	directory
/bin/.*	bin_t:s0	all files
/bin/alsaunmute	alsa_exec_t:s0	regular file
/bin/bash	shell_exec_t:s0	regular file
/bin/bash2	shell_exec_t:s0	regular file
/bin/d?ash	shell_exec_t:s0	regular file

2: ARE YOUR CONFINED APPLICATIONS SET UP DIFFERENTLY FROM THE DEFAULT?

A confined process or application can be run in many different ways. You need to tell SELinux how you have configured the application to run, and then SELinux will allow it the proper access. SELinux does not do this automatically.

SELinux policy has built-in if/then/else rules, called booleans, that allow you to tweak the predefined rules to allow different access.

Out of the box, SELinux policy for **httpd** does not allow it to send mail. This is to prevent a compromised website from becoming a spam box.

However, you might want your **httpd** to send mail legitimately. You can set the **httpd_can_sendmail** boolean to tell SELinux that it is okay to connect to the mail port or to execute one of the commands used to send mail:

```
# setsebool -P httpd_can_sendmail 1
```

This permanently changes SELinux policy to allow **httpd** to send mail.

To view all booleans for http you can execute (output truncated):

```
# semanage boolean -l | grep http
httpd_can_network_relay         -> off   Allow httpd to act as a relay
httpd_can_network_connect_db    -> off   Allow HTTPD scripts and modules to 
connect to databases over the network.
httpd_enable_cgi                -> on    Allow httpd cgi support
httpd_use_cifs                  -> off   Allow httpd to access cifs file systems
allow_httpd_mod_auth_pam        -> off   Allow Apache to use mod_auth_pam
allow_httpd_anon_write          -> off   Allow Apache to modify public files 
used for
```

Tools like **system-config-selinux** or **getsebool -a** will list all of the possible booleans.

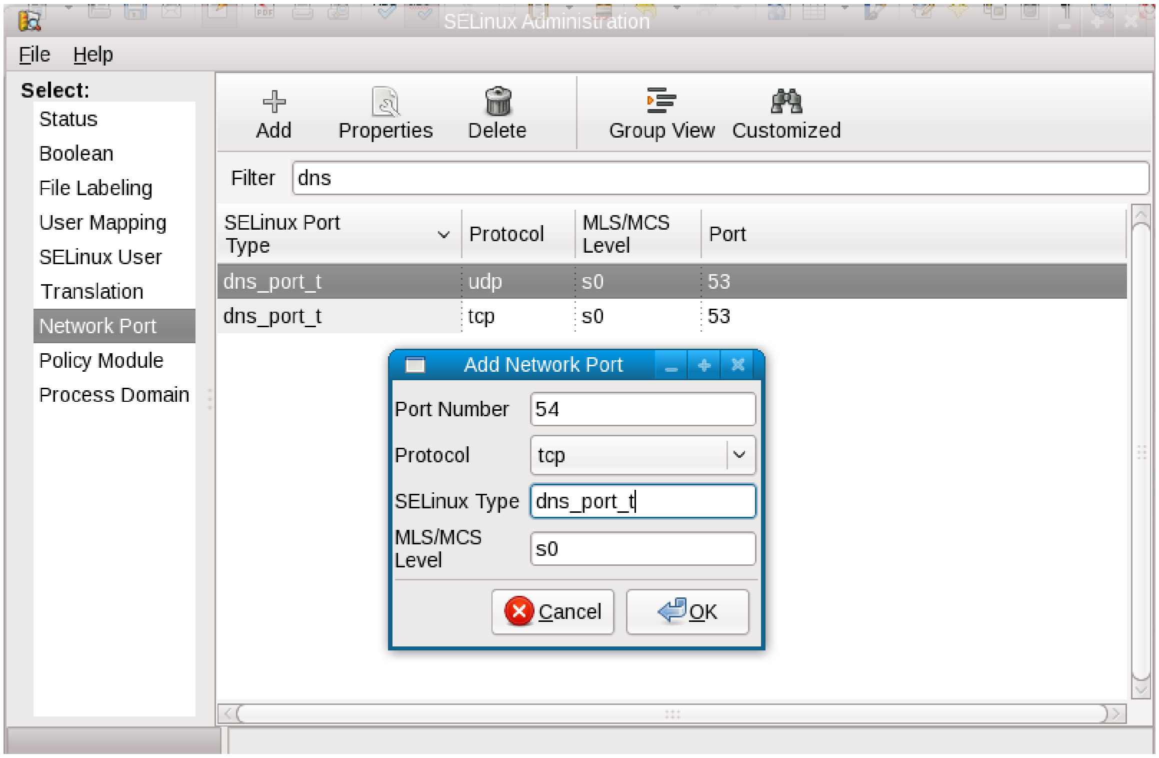
On Red Hat Enterprise Linux 6 and all current Fedora Systems you can run SELinux error messages (AVC) through **audit2allow -w**. This command will check to see if any boolean could be toggled to allow the access. The **setroubleshoot** command can also be useful in diagnosing these problems.

You might want to change the network ports that a confined application is allowed to listen on or to connect to. In certain cases there is a boolean to allow the connection. For example, the **httpd_can_sendmail** boolean allows the **httpd** daemon to connect to the mail port.

Generally, you will need to tell SELinux if you want to use non-default ports with a confined application. Use the **semanage** command to tell SELinux which ports you want to use. For example, if you want to allow the bind daemon to listen on tcp port 54, you would need to execute the following command:

```
# semanage port -a -t dns_port_t -p tcp 54
# semanage port -l | grep dns
dns_port_t      tcp  54, 53
dns_port_t      udp  53
```

You can also use **system-config-selinux** to manage SELinux network configuration, as shown below.



Filter: dns

SELinux Port Type	Protocol	MLS/MCS Level	Port
dns_port_t	udp	s0	53
dns_port_t	tcp	s0	53

Port Number: 54

Protocol: tcp

SELinux Type: dns_port_t

MLS/MCS Level: s0

3: IS THERE A BUG IN SELINUX POLICY OR IN A CONFINED APPLICATION?

SELinux policy is written for a confined domain by looking at what an application does, putting the application or system into permissive mode, and collecting the AVC messages. The policy is then updated using these messages.

Sometimes a confined application is run with a code path that the policy writer did not know about, so the policy denies the access even though it should be allowed. These kinds of problems can be reported to Red Hat Support for assistance. To overcome these problems, you can add custom policy to your system simply by piping the SELinux error messages through **audit2allow**.

For example, suppose a new version of **postgresql** is released, and SELinux is mistakenly denying access to a resource that **postgresql** should be allowed to access. You can use **audit2allow** to build a custom policy module that can be installed on your system to allow the access using the following command:

```
# grep postgresql /var/log/audit/audit.log | audit2allow -R -M mypostgresql
```

This command will generate a local policy package (**mypostgresql.pp**) that will allow all **postgresql** accesses that are currently being denied. You can examine the source type enforcement file, **mypostgresql.te**, for the generated policy package. This file contains all of the new allow rules. You should examine these before installing them to make sure that it is safe to install the rules. Red Hat Support can help to determine if it is safe to use the rules. If you decide to modify the source file you can recompile the policy package file using the following command:

```
# make -f /usr/share/selinux/devel/Makefile
```

Once you have verified the policy is safe, you can install and load it on the system using the semodule command:

```
# semodule -i mypostgresql.pp
```

This command installs the local policy modifications permanently to your system. If you report the SELinux errors to Red Hat Support, your local modifications can be added to the distribution's policy or upstream.

4: HAS YOUR MACHINE BEEN COMPROMISED?

SELinux is not an intrusion detection system, and it is not always possible to distinguish between an intrusion and a general configuration, labeling, or SELinux policy error. Several tools are available to detect intrusions, and some of these use the SELinux logs to watch for intrusions.

SELinux will trigger lots of AVCs if an application is actually compromised and tries to do something it is not designed to do. In RHEL6 and the latest Fedora systems, **setroubleshoot** looks for compromised applications signatures.

If an application requires major security privileges, SELinux policy probably already allows it. If you see AVCs that do not make sense or seem to indicate an application trying to change security settings, your application might be compromised.

Some potential signatures of a compromised confined application include:

- A confined application should never try to change SELinux enforcement. This includes changing the enforcement mode or trying to write to **/etc/selinux**. Setting booleans would also be a very unusual thing for a confined application to do.
- A confined application should not try to modify the kernel. This includes loading kernel modules, writing to kernel directories, and writing to boot loader or image directories.
- A confined application should not attempt to write to files labeled **etc_t** (**/etc**), because a confined domain that can write to **etc_t** would be able to overwrite **passwd_t**.
- Confined applications should not try to write to security configuration files. This includes certificates, kerberos files, and most configuration data.
- Most confined applications should not try to write to **shadow_t** or, in most cases, read from **shadow_t**.
- Confined applications should not try to overwrite log files, particularly if the log file is not related to the application.
- Most confined applications should not try to read files in the user's home directory (**user_home_t**).
- Confined application should not try to suddenly connect to random network ports. For example, spambots will try to connect to the mail port.
- A confined application should not try to execute mail programs or connect to mail ports, if they were not set up to send mail.

Any of these could be a bug, but the potential damage is too significant to ignore. If you encounter anything like the above, you should seek help in diagnosing the AVC messages. Contact Red Hat Support for help diagnosing these issues.