How To Troubleshoot SELinux with Audit Logs


Audit Logs with SELinux Messags

I'm post configuring a new RHEL 8 setup on my old PC and want to share some useful **SELinux** troubleshooting techniques.

How To Check Audit Logs for SELinux

I had a problem with **SSH** not accepting keys for login. Specifically, I wanted the keys to be in a non-standard /var/ssh/greys/authorized_keys location (instead of my homedir), but SSH daemon would just ignore this file.

I double checked permissions, restarted **SSHd** and eventuall realised that the issue must have been due to SELinux. So I went to inspect the audit logs.

Red Hat Enterprise Linux puts audit logs into /var/log/audit directory. If you're looking for SELinux issues, just grep for **denied** – it will show you everything that has recently been blocked:

```
root@rhel8:~ # grep denied /var/log/audit/*
type=AVC msg=audit(1567799177.932:3031): avc:  denied  { read } for  pid=24527 comm="sshd" name="authorized_keys"
```

dev="dm-11" ino=26047253 scontext=system_u:system_r:sshd_t:s0-s0:c0.c1023 tcontext=system_u:object_r:var_t:s0 tclass=file permissive=0
 type=AVC msg=audit(1567799177.943:3033): avc: denied { read } for pid=24527 comm="sshd" name="authorized_keys" dev="dm-11" ino=26047253 scontext=system_u:system_r:sshd_t:s0-s0:c0.c1023 tcontext=system_u:object_r:var_t:s0 tclass=file permissive=0
 type=AVC msg=audit(1567799177.956:3035): avc: denied { read } for pid=24527 comm="sshd" name="authorized_keys" dev="dm-11" ino=26047253 scontext=system_u:system_r:**sshd_t**:s0-s0:c0.c1023 tcontext=system_u:object_r:**var_t**:s0 tclass=file permissive=0

I also highlighted the likely problem: SSH daemon is running under **sshd_t** context, but files in **/var/ssh/** directories inherited standard **var_t** context.

Just to be sure, I checked the context on the default **/home/greys/.ssh/authorized_keys** file:

root@rhel8:~ # **ls -alZ /home/greys/.ssh/authorized_keys**
 -rw-------. 1 greys greys unconfined_u:object_r:**ssh_home_t**:s0 95 Sep 6 20:28 /home/greys/.ssh/authorized_keys

That's the answer! We need to change **/var/ssh/greys/authorized_keys** file to the **ssh_home_t** context.

Updating SELinux context for a file

First, let's change the SELinux context:

```
root@rhel8:~ # semanage fcontext -a -t ssh_home_t /var/ssh/greys/authorized_keys
```

… and now we relabel the actual file:

```
root@rhel8:~ # restorecon -Rv /var/ssh/greys/authorized_keys
Relabeled /var/ssh/greys/authorized_keys from system_u:object_r:var_t:s0 to system_u:object_r:ssh_home_t:s0
```

That's it – after that my logins using SSH keys started working just fine. Hope you find this example useful!

See Also

- Confirm Current SELinux Mode
- How To Disable SELinux
- How To: List SELinux Contexts for Files
- Where To Learn More About SELinux

Show List of Available SELinux Users

```
[greys@rhel8 ~]$ seinfo -u

Users: 8
   guest_u
   root
   staff_u
   sysadm_u
   system_u
   unconfined_u
   user_u
   xguest_u
```

I'm slowly improving my understanding of the **SELinux setup**, currently looking into controlling user access. As you know, there may be lots of different users created in your Linux system. For them to be controlled by the SELinux framework, we need to map all users to one of the users in SELinux policy.

Install SELinux Tools

The command we need is called **seinfo**, and it's not installed by default. We have to install the **setools-console** package first:

```
[greys@rhel8 ~]$ sudo yum install setools-console
[sudo] password for greys:
Updating Subscription Management repositories.
Updating Subscription Management repositories.
Red Hat Enterprise Linux 8 for x86_64 - AppStream Beta (RPMs)
3.0 kB/s | 4.1 kB 00:01
Red Hat Enterprise Linux 8 for x86_64 - BaseOS Beta (RPMs) 3.0
kB/s | 4.1 kB 00:01
Dependencies resolved.
================================================================
===========
Package Arch Version Repository Size
================================================================
===========
```

```
Installing:
setools-console x86_64 4.1.1-11.el8 rhel-8-for-x86_64-baseos-
beta-rpms 28 k

Transaction Summary
================================================================
===========

Install 1 Package

Total download size: 28 k
Installed size: 109 k
Is this ok [y/N]: y
Downloading Packages:
setools-console-4.1.1-11.el8.x86_64.rpm 15 kB/s | 28 kB 00:01
----------------------------------------------------------------
----------------------------------------------------------------
---
Total 15 kB/s | 28 kB 00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing : 1/1
Installed: setools-console-4.1.1-11.el8.x86_64
Installing : setools-console-4.1.1-11.el8.x86_64 1/1
Installed: setools-console-4.1.1-11.el8.x86_64
Running scriptlet: setools-console-4.1.1-11.el8.x86_64 1/1
Verifying : setools-console-4.1.1-11.el8.x86_64 1/1

Installed:
setools-console-4.1.1-11.el8.x86_64

Complete!
```

List Available SELinux Users

Now that the package is installed, run the seinfo -u command to show list of SELinux users:

```
[greys@rhel8 ~]$ seinfo -u

Users: 8
guest_u
root
staff_u
sysadm_u
system_u
unconfined_u
user_u
xguest_u
```

While we're at it, let's check the current user's SELinux context: usually you're mapped to the unconfined_u user:

```
[greys@rhel8 ~]$ id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

See Also

- List files with SELinux contexts
- SELinux status
- Advanced sestatus
- Enable SELinux
- Disable SELinux
- Linux Commands

How To: List Files with SELinux Contexts

```
[greys@rhel8 ~]$ ls -alZ /home/greys
total 64
drwx------. 17 greys greys unconfined_u:object_r:user_home_dir_t:s0   4096 Feb 19 12:14 .
drwxr-xr-x.  3 root  root  system_u:object_r:home_root_t:s0             19 Jan 15 17:34 ..
-rw-------.  1 greys greys unconfined_u:object_r:user_home_t:s0       2035 Feb 19 12:14 .bash_history
-rw-r--r--.  1 greys greys unconfined_u:object_r:user_home_t:s0         18 Oct 12 17:56 .bash_logout
-rw-r--r--.  1 greys greys unconfined_u:object_r:user_home_t:s0        218 Jan 28 17:42 .bash_profile
-rw-r--r--.  1 greys greys unconfined_u:object_r:user_home_t:s0        312 Oct 12 17:56 .bashrc
drwx------. 12 greys greys unconfined_u:object_r:cache_home_t:s0      4096 Jan 21 06:41 .cache
drwx------. 14 greys greys unconfined_u:object_r:config_home_t:s0      278 Jan 21 06:41 .config
drwx------.  3 greys greys unconfined_u:object_r:dbus_home_t:s0         25 Jan 20 18:28 .dbus
drwxr-xr-x.  2 greys greys unconfined_u:object_r:user_home_t:s0          6 Jan 20 18:28 Desktop
drwxr-xr-x.  2 greys greys unconfined_u:object_r:user_home_t:s0          6 Jan 20 18:28 Documents
drwxr-xr-x.  2 greys greys unconfined_u:object_r:user_home_t:s0          6 Jan 20 18:28 Downloads
-rw-------.  1 greys greys unconfined_u:object_r:pulseaudio_home_t:s0   16 Jan 15 19:15 .esd_auth
-rw-------.  1 greys greys unconfined_u:object_r:iceauth_home_t:s0    1244 Jan 20 18:46 .ICEauthority
```

When [running a **SELinux** based setup](), it might be useful to know how to quickly inspect files and directories to confirm their current SELinux context.

What is SELinux Context?

Every process and file in [SELinux based environment]() can be labeled with additional information that helps fulfill RBAC (Role-Based Access Control), TE (Type Enforcement) and MLS (Multi-Level Security).

SELinux context is the combination of such additional information:

- user
- role
- type
- level

In the following example we can see that **unconfined_u** is the SELinux user, **object_r** is the role, **user_home_dir_t** is the object type (home user directory) and the SELinux sensitivity (MCS terminology) level is **s0**:

```
drwx------.  17  greys  greys  unconfined_u:object_r:user_home_dir_t:s0  4096  Feb  19  12:14  .
```

Use ls -Z to show SELinux Context

Using ls command with -Z option will show the SELinux contexts. This command line option is totally made to be combined with other **ls command** options:

```
[greys@rhel8 ~]$ ls -alZ .
total 64
drwx------.           17           greys           greys
unconfined_u:object_r:user_home_dir_t:s0 4096 Feb 19 12:14 .
drwxr-xr-x. 3 root root system_u:object_r:home_root_t:s0 19
Jan 15 17:34 ..
-rw-------. 1 greys greys unconfined_u:object_r:user_home_t:s0
2035 Feb 19 12:14 .bash_history
-rw-r--r--. 1 greys greys unconfined_u:object_r:user_home_t:s0
18 Oct 12 17:56 .bash_logout
-rw-r--r--. 1 greys greys unconfined_u:object_r:user_home_t:s0
218 Jan 28 17:42 .bash_profile
-rw-r--r--. 1 greys greys unconfined_u:object_r:user_home_t:s0
312 Oct 12 17:56 .bashrc
drwx------.           12           greys           greys
unconfined_u:object_r:cache_home_t:s0 4096 Jan 21 06:41 .cache
drwx------.           14           greys           greys
unconfined_u:object_r:config_home_t:s0  278  Jan  21  06:41
.config
drwx------. 3 greys greys unconfined_u:object_r:dbus_home_t:s0
25 Jan 20 18:28 .dbus
drwxr-xr-x. 2 greys greys unconfined_u:object_r:user_home_t:s0
6 Jan 20 18:28 Desktop
drwxr-xr-x. 2 greys greys unconfined_u:object_r:user_home_t:s0
6 Jan 20 18:28 Documents
drwxr-xr-x. 2 greys greys unconfined_u:object_r:user_home_t:s0
6 Jan 20 18:28 Downloads
-rw-------.           1           greys           greys
unconfined_u:object_r:pulseaudio_home_t:s0  16  Jan  15  19:15
.esd_auth
-rw-------.           1           greys           greys
unconfined_u:object_r:iceauth_home_t:s0  1244  Jan  20  18:46
.ICEauthority
-rw-------. 1 greys greys unconfined_u:object_r:user_home_t:s0
3434 Jan 22 18:06 id_rsa_4k
```

```
-rw-r--r--. 1 greys greys unconfined_u:object_r:user_home_t:s0
737 Jan 22 18:06 id_rsa_4k.pub
-rw-rw-r--. 1 greys greys unconfined_u:object_r:user_home_t:s0
21 Jan 28 17:53 infile2.txt
-rw-------. 1 greys greys unconfined_u:object_r:user_home_t:s0
38 Jan 22 18:05 .lesshst
drwxr-xr-x.           3           greys           greys
unconfined_u:object_r:gconf_home_t:s0 19 Jan 20 18:28 .local
drwxr-xr-x.           2           greys           greys
unconfined_u:object_r:audio_home_t:s0 6 Jan 20 18:28 Music
-rw-rw-r--. 1 greys greys unconfined_u:object_r:user_home_t:s0
0 Jan 22 18:01 newkey
drwxr-xr-x. 2 greys greys unconfined_u:object_r:user_home_t:s0
6 Jan 20 18:28 Pictures
drwxrw----. 3 greys greys unconfined_u:object_r:home_cert_t:s0
19 Jan 20 18:28 .pki
drwxr-xr-x. 2 greys greys unconfined_u:object_r:user_home_t:s0
6 Jan 20 18:28 Public
drwxrwxr-x. 4 greys greys unconfined_u:object_r:user_home_t:s0
165 Jan 16 11:00 screenFetch
-rw-------.           1           greys           greys
unconfined_u:object_r:xauth_home_t:s0  150  Jan  20  18:44
.serverauth.1859
-rw-------.           1           greys           greys
unconfined_u:object_r:xauth_home_t:s0  50  Jan  20  18:39
.serverauth.1893
drwx------. 2 greys greys unconfined_u:object_r:ssh_home_t:s0
70 Jan 22 18:07 .ssh
-rw-rw-r--. 1 greys greys unconfined_u:object_r:user_home_t:s0
0 Jan 21 07:49 system_u:object_r:shell_exec_t:s0
drwxr-xr-x. 2 greys greys unconfined_u:object_r:user_home_t:s0
6 Jan 20 18:28 Templates
drwxr-xr-x. 2 greys greys unconfined_u:object_r:user_home_t:s0
6 Jan 20 18:28 Videos
-rw-------. 1 greys greys unconfined_u:object_r:user_home_t:s0
2874 Jan 29 04:40 .viminfo
-rw-------.           1           greys           greys
unconfined_u:object_r:xauth_home_t:s0  260  Feb  19  12:14
.Xauthority
```

See Also

- Enable SELinux
- Disable SELinux
- Confirm SELinux Status
- Get SELinux mode with getenforce
- Advanced sestatus Usage
- Linux Commands
- **SELinux Reference**

Unix Tutorial Digest – February 4th, 2019

Unix Tutorial Digest: monthly digest of Unix/Linux topics

Thanks to the super busy January, this is actually the first Unix Tutorial digest of 2019! Lots of news and so many things

to follow up and test now!
As always, Please get in touch if you want to suggest a useful link for the next digest.

Unix and Linux News

- Linux kernel 4.20
- Ubuntu Core 18.10 got released – sounds like a pretty cool idea for IoT things. I must try it on one of my Raspberry Pi systems, but it seems it won't work on Raspberry Pi Model 1.
- I seem to have forgotten Red Hat Enterprise Linux 8 beta in my previous digest
- Bash 5.0 got released! Not just any software, but one of the most popular Unix shells!

Software News

- LetsEncrypt (my SSL provider of choice for all the self-hosted elements) reported 2018 progress and LetsEncrypt 2019 plans – 150M websites are using them now, impressive!
- in mid January, 4 issues with SCP in OpenSSH, Putty and WinSCP were discovered. Hope you were not affected!
- Were you affected by the DNS Flag Day on February 1st? Great initiative to highlight dependency on DNS and to steer users towards better implementations of it.
- Wine 4.0 got released, quickly followed up by Wine Staging 4.0.
- Kodi 18 platform got released, followed by LibreELEC 9.0.0 based on it

Interesting and Useful

- Scraping TripAdvisor: text mining and sentiment analysis – can't believe how simple and readable the Python code is for a task of such complexity

Unix Tutorial articles

I made some New Years' resolutions, one of them is a dramatically improved commitment to updating Unix Tutorial. As the result, I published more content in January 2019, than in the previous 9 years!

- Happy New Year 2019!
- ISO to USB in MacOS
- Test TCP connectivity with curl
- VirtualBox 6.0
- How To: Test Disk I/O with dd
- How To Use 7zip
- SSH port
- GitHub: Private Repositories are Free Now
- Docker – List Containers
- HW Virtualization
- Bash 5.0Bash 5.0
- Unix Epoch
- Linux: List All Users
- How To: Mount ISO image in Linux
- pcs STONITH
- How To: Restart MySQL
- screenFetch in RHEL 8
- How To: Disable SElinux
- Docker: Stop All Containers
- How To: Confirm Current Kernel Boot Commands in Linux
- SElinux Status
- How To: Generate SSH key
- Unix Diff
- SElinux: Advanced sestatus usage

- List RHEL subscriptions
- YUM: list and install software groups
- tmux: basic configuration
- Show network errors with netstat

That’s it for today!

See Also

- Unix Tutorial Digest
- Unix commands
- Basic Unix commands
- Advanced Unix commands

How To Enable SElinux

SElinux – Security Enhanced Linux

If you're using RedHat or CentOS Linux distros (or sporting a Fedora Linux desktop), you probably have **SELinux** enabled by default. But if it's been disabled for some reason and you want it back – here's how you can enable **SELinux** in your Linux system.

Confirm current SELinux mode

Run the getenforce command to confirm that SELinux is actually disabled:

```
[root@rhel8 ~]# getenforce
Disabled
```

Check SElinux status with sestatus

sestatus normally shows verbose SElinux status information, but if SELinux is disabled, you'll only get one line of output, like this:

```
root@rhel8 ~]# sestatus
SELinux status: disabled
[root@rhel8 ~]#
```

If **sestatus** shows that SELinux is disabled, you'll need to enable it via /etc/selinux.png/config file and reboot the server as shown below.

Permanently Enable SELinux

Do the following two steps to enable SELinux:

1. Update **/etc/selinux.png/config** file (change **SELINUX=disabled** to **SELINUX=enforcing**)
2. Reboot your Linux system (**shutdown -r now**)

Once your server comes back online, run **sestatus** again to make sure **SElinux is enabled** now:

```
[root@rhel8 ~]# sestatus
SELinux status: enabled
SELinuxfs mount: /sys/fs/selinux.png
SELinux root directory: /etc/selinux.png
Loaded policy name: targeted
Current mode: enforcing
Mode from config file: enforcing
Policy MLS status: enabled
Policy deny_unknown status: allowed
Memory protection checking: actual (secure)
Max kernel policy version: 31
```

See Also

- SELinux Status
- How To Disable SELinux
- Advanced Unix Commands
- Linux Commands
- SELinux Reference

SElinux: Advanced sestatus usage

I learned something new today! Apparently, **sestatus command** can report security contexts of the key system files – really neat for quickly recognising possible security compromise.

Files and processes in /etc/sestatus.conf

The way this works is you must use the **/etc/sestatus.conf** file which contains list of files and list of processes that are checked for **SElinux** contexts. These are the most common security attack vectors, so SElinux notes them and helps you to quickly confirm their contexts using **sestatus -v** command.

VERY IMPORTANT: at this stage **sestatus command** does NOT highlight or warn you about any non-standard contextual changes. So the only thing is does is show you all the important files you selected and report their current contexts – if some of these have been changed, the task of recognising or fixing this is still on you.

You can add any files and process you like here, but here's

the default list in RHEL8:

```
[greys@rhel8 ~]$ cat /etc/sestatus.conf
[files]
/etc/passwd
/etc/shadow
/bin/bash
/bin/login
/bin/sh
/sbin/agetty
/sbin/init
/sbin/mingetty
/usr/sbin/sshd
/lib/libc.so.6
/lib/ld-linux.so.2
/lib/ld.so.1

[process]
/sbin/mingetty
/sbin/agetty
/usr/sbin/sshd
```

Files and processes contexts with sestatus

```
[greys@rhel8 ~]$ sestatus -v
SELinux status: enabled
SELinuxfs mount: /sys/fs/selinux.png
SELinux root directory: /etc/selinux.png
Loaded policy name: targeted
Current mode: enforcing
Mode from config file: enforcing
Policy MLS status: enabled
Policy deny_unknown status: allowed
Memory protection checking: actual (secure)
Max kernel policy version: 31

Process contexts:
Current  context:  unconfined_u:unconfined_r:unconfined_t:s0-
s0:c0.c1023
```

```
Init context: system_u:system_r:init_t:s0

File contexts:
Controlling terminal: unconfined_u:object_r:user_devpts_t:s0
/etc/passwd system_u:object_r:passwd_file_t:s0
/etc/shadow system_u:object_r:shadow_t:s0
/bin/bash system_u:object_r:shell_exec_t:s0
/bin/login system_u:object_r:login_exec_t:s0
/bin/sh         system_u:object_r:bin_t:s0         ->
system_u:object_r:shell_exec_t:s0
/sbin/agetty system_u:object_r:getty_exec_t:s0
/sbin/init       system_u:object_r:bin_t:s0       ->
system_u:object_r:init_exec_t:s0
/usr/sbin/sshd system_u:object_r:sshd_exec_t:s0
```

See Also

- Disable SElinux
- Check SElinux Status
- Security commands in Unix

SElinux Status

SELinux

This post shows you how to confirm current **SElinux status**

before you decide to disable SELinux.

SElinux Enforcing vs Permissive

The most burning question usually is: does my RedHat/CentOS Linux enforce SELinux (and prevent some of my applications from running out of the box) or is it in the permissive state (which means it logs security concerns but doesn't block anything from running).

Answering this is very easy with the help of the **getenforce command**:

```
[greys@rhel8 ~]$ getenforce
Enforcing
```

SElinux status with sestatus

If you're more curious about the way SELinux is configured, then **sestatus command** will be much more useful:

```
[greys@rhel8 ~]$ sestatus
SELinux status: enabled
SELinuxfs mount: /sys/fs/selinux.png
SELinux root directory: /etc/selinux.png
Loaded policy name: targeted
Current mode: enforcing
Mode from config file: enforcing
Policy MLS status: enabled
Policy deny_unknown status: allowed
Memory protection checking: actual (secure)
Max kernel policy version: 31
```

How to read the sestatus output

Although the output of **sestatus** is fairly standard, you'll appreciate how useful it is once you start making changes to your SELinux policies.

- **Loaded policy name** is useful because you can make SELinux load a strict policy as well, and it's important to understand which one is currently in use.
- **Current mode**: will confirm if SELinux is running in enforcing or permissive mode.
- **Policy MLS status**: must research more! I know MLS is Multi Level Security, but need to understand why it's separate option here.
- **Memory protection checking** – must come back to this as I'm not finding enough information. This is a flag confirming that SELinux still protects certain memory access syscalls in your Linux.

See Also

- How To: Disable SELinux
- How To: Enable SELinux
- **SELinux Reference**

How To: Disable SELinux

SElinux – Security Enhanced Linux

If you're using RedHat or CentOS Linux distros (or sporting a Fedora Linux desktop), you probably have SELinux enabled by default. SELinux is a Security-Enhanced Linux – a framework for securely managing processes, users and files on your RedHat OS.

Confirm current SElinux mode

Just run the getenforce command to see what the story is. Most likely it will say "Enforcing" which is really good – means your OS is under solid protection:

```
[root@rhel8 ~]# getenforce
Enforcing
```

Temporarily Disable SELinux

If you need to disable SELinux just for a few minutes to debug some issue (mind you, there are better ways to debug than disabling SELinux!), you should use the setenforce command:

```
[root@rhel8 ~]# setenforce 0
```

As you can see, getenfore will now report that your system is running in a Permissive mode – not very safe:

```
[root@rhel8 ~]# getenforce
Permissive
```

IMPORTANT: This change won't survive a reboot, so next time you restart your system it will come back with SELinux enabled and enforcing again.

Permanently Disable SELinux

If you're serious about disabling SELinux altogether, you'll have to do two things:

1. Update **/etc/selinux.png/config** file (change **SELINUX=enforcing** to **SELINUX=disabled**)
2. Reboot your Linux system

See Also

- make iptables survice a reboot
- **SELinux Reference**
- Advanced Unix Commands
- Linux Commands