Oracle Linux

Administering SELinux

F22957-18
August 2024

Oracle Linux Administering SELinux,

F22957-18

Contents

Preface

Oracle Linux: Administering SELinux provides an overview of the SELinux feature and includes tasks for administering SELinux on Oracle Linux systems.

Documentation License

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at https://www.oracle.com/corporate/accessibility/.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit https://www.oracle.com/corporate/accessibility/learning-support.html#support-tab.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and

the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

1

About Administering SELinux in Oracle Linux

This chapter describes the SELinux feature and provides tasks for administering SELinux on Oracle Linux systems.

> **Note:**
>
> The content in this document was tested against Oracle Linux 8 and Oracle Linux 9, but also applies to most Oracle Linux releases, and might also apply to other distributions.

Traditional Linux security is based on a Discretionary Access Control (DAC) policy, which provides minimal protection from broken software or from malware that's running as a normal user or as `root`. Access to files and devices is based solely on user identity and ownership. Malware or broken software can do anything with files and resources that the user that started the process can do. If the user is `root` or the application is `setuid` or `setgid` to `root`, the process can have `root`-access control over the entire file system.

The National Security Agency created Security Enhanced Linux (SELinux) to provide a finer-grained level of control over files, processes, users, and applications in the Linux OS. The SELinux enhancement to the Linux kernel implements the Mandatory Access Control (MAC) policy, which enables you to define a security policy that provides granular permissions for all users, programs, processes, files, and devices. The kernel's access control decisions are based on all the security relevant information available, and not solely on the authenticated user identity.

When security-relevant access occurs, such as when a process opens a file, SELinux intercepts the operation in the kernel. If a MAC policy rule allows the operation, it continues. Otherwise, SELinux blocks the operation and returns an error to the process. The kernel checks and enforces DAC policy rules before MAC rules, so it doesn't check SELinux policy rules if DAC rules have already denied access to a resource.

SELinux Package Descriptions

SELinux contains several packages, each of which contain specific utilities that you can use to administer SELinux on Oracle Linux systems. Some packages are installed by default, while other packages are optional.

The following table describes the SELinux packages that are installed by default with Oracle Linux.

Package	Description
`policycoreutils`	Provides utilities such as `load_policy`, `restorecon`, `secon`, `setfiles`, `semodule`, `sestatus`, and `setsebool` for operating and managing SELinux.

Package	Description
libselinux	Provides the API that SELinux applications use to get and set process and file security contexts, and to obtain security policy decisions.
python3-libselinux	Contains Python bindings for developing SELinux applications.
selinux-policy	Provides the SELinux Reference Policy, which is used as the basis for other policies, such as the SELinux targeted policy.
selinux-policy-targeted	Provides the SELinux targeted policy, where objects outside the targeted domains run under DAC.
libselinux-utils	Provides the avcstat, getenforce, getsebool, matchpathcon, selinuxconlist, selinuxdefcon, selinuxenabled, setenforce, and togglesebool utilities.

The following table describes useful SELinux packages that aren't installed by default.

Package	Description
mcstrans	Translates SELinux levels, such as s0-s0:c0.c1023, to an easier-to-read form, such as SystemLow-SystemHigh.
policycoreutils-python-utils	Provides Python utilities for operating SELinux, such as audit2allow, audit2why, chcat, and semanage.
selinux-policy-mls	Provides a strict Multi-Level Security (MLS) policy as an alternative to the SELinux targeted policy.
selinux-policy-doc	Provides manual pages for many SELinux policy elements.
setroubleshoot	Enables you to view setroubleshoot-server messages by using the sealert command.
setroubleshoot-server	Translates access-denial messages from SELinux into detailed descriptions that you can view on the command line using the sealert command.
setools-console	Provides the Tresys Technology SETools distribution of tools and libraries, which you can use to analyze and query policies, monitor and report audit logs, and to manage file context.

Use the dnf command or another suitable package manager to install SELinux packages that you require for the system.

For more information, see the SELinux Project Wiki, the selinux(8) manual page, and other manual pages for the SELinux commands.

Using SELinux Utilities

The following table describes the utilities that you can use to administer SELinux and information about the packages that contain each utility.

Utility	Package	Description
audit2allow	policycoreutils-python-utils	Generates SELinux policy allow_audit rules from logs of denied operations.
audit2why	policycoreutils-python-utils	Generates SELinux policy don't_audit rules from logs of denied operations.
avcstat	libselinux-utils	Displays statistics for the SELinux Access Vector Cache (AVC).
chcat	policycoreutils-python-utils	Changes or removes the security category for a file or user.
findcon	setools-console	Searches for file context.
fixfiles	policycoreutils	Fixes the security context for file systems.
getenforce	libselinux-utils	Reports the current SELinux mode.
getsebool	libselinux-utils	Reports SELinux Boolean values.
indexcon	setools-console	Indexes file context.
load_policy	policycoreutils	Loads a new SELinux policy into the kernel.
matchpathcon	libselinux-utils	Queries the system policy and displays the default security context that's associated with the file path.
replcon	setools-console	Replaces file context.
restorecon	policycoreutils	Resets the security context on one or more files.
restorecond	policycoreutils	Daemon that watches for file creation and sets the default file context.
sandbox	policycoreutils-python-utils	Runs a command in an SELinux sandbox.
sealert	setroubleshoot-server, setroubleshoot	Acts as the user interface to the setroubleshoot system for diagnosing and explaining SELinux AVC denials and providing recommendations on how to prevent such denials.
sechecker	setools-console	Checks SELinux policies.

Utility	Package	Description
secon	policycoreutils	Displays the SELinux context from a file, program, or user input.
sediff	setools-console	Compares SELinux polices.
seinfo	setools-console	Queries SELinux policies.
selinuxconlist	libselinux-utils	Displays all SELinux contexts that are reachable by a user.
selinuxdefcon	libselinux-utils	Displays the default SELinux context for a user.
selinuxenabled	libselinux-utils	Indicates whether SELinux is enabled.
semanage	policycoreutils-python-utils	Manages SELinux policies.
semodule	policycoreutils	Manages SELinux policy modules.
semodule_deps	policycoreutils	Displays the dependencies between SELinux policy packages.
semodule_expand	policycoreutils	Expands a SELinux policy module package.
semodule_link	policycoreutils	Links SELinux policy module packages together.
semodule_package	policycoreutils	Creates a SELinux policy module package.
sesearch	setools-console	Queries SELinux policies.
sestatus	policycoreutils	Displays the SELinux mode and the SELinux policy that are in use.
setenforce	libselinux-utils	Changes the SELinux mode.
setsebool	policycoreutils	Sets SELinux Boolean values.
setfiles	policycoreutils	Sets the security context for one or more files.
togglesebool	libselinux-utils	Flips the current value of an SELinux Boolean.

Setting SELinux Modes

SELinux runs in one of three modes:

Disabled
The kernel uses only DAC rules for access control. SELinux doesn't enforce any security policy because no policy is loaded into the kernel.

Enforcing
The kernel denies access to users and programs if they aren't granted permissions by SELinux security policy rules. All denial messages are logged as AVC (Access Vector Cache) denials. This is the default mode that enforces SELinux security policy.

Permissive
The kernel doesn't enforce security policy rules but SELinux sends denial messages to a log file. In this manner, you can see what actions would have been denied if SELinux were running in enforcing mode. This mode is intended to be used for diagnosing the behavior of SELinux.

To display current SELinux mode:

```
getenforce
```

To set the current mode to `Enforcing`:

```
sudo setenforce enforcing
```

To set the current mode to `Permissive`:

```
sudo setenforce permissive
```

The current value that you set for a mode using `setenforce` doesn't persist across reboots. To configure the default SELinux mode, edit the configuration file for SELinux, `/etc/selinux/config`, and set the value of the `SELINUX` directive to `disabled`, `enforcing`, or `permissive`.

Getting More Information

SELinux is complex. You can obtain information about different policies more easily by installing the `selinux-policy-doc` package and then navigating the associated manual pages.

1. Install the package:

   ```
   sudo dnf install -y selinux-policy-doc
   ```

2. Update the manual page database:

   ```
   sudo mandb
   ```

3. Start searching through the new SELinux policy manual pages. To get a complete listing of all the SELinux manual documentation, run:

   ```
   man -k _selinux
   ```

The policy documentation contains information about users and roles. For example, you can read more about the SELinux unprivileged `user_u` user and the `user_r` role in the `user_selinux(8)` manual page. The policy documentation outlines the restrictions applied for different security contexts and what Boolean options are available to you to customize the policy for an environment.

2

Administering SELinux Policies

An SELinux policy describes the access permissions for all users, programs, processes, and files, and for the devices upon which they act. You can configure SELinux to implement either Targeted Policy or Multi-Level Security (MLS) Policy. This chapter describes SELinux policies and how to administer them.

Targeted Policy

A targeted policy applies access controls to a limited number of processes that are believed to be most likely to be the targets of an attack on the system. Targeted processes run in their own SELinux domain, known as a *confined domain*, which restricts access to files that an attacker could exploit. If SELinux detects that a targeted process is trying to access resources outside the confined domain, it denies access to those resources and logs the denial. Only specific services run in confined domains. Examples are services that listen on a network for client requests, such as `httpd`, `named`, and `sshd`, and processes that run as `root` to perform tasks on behalf of users, such as `passwd`. Other processes, including most user processes, run in an unconfined domain where only DAC rules apply. If an attack compromises an unconfined process, SELinux doesn't prevent access to system resources and data.

The following table shows examples of SELinux domains.

Domain	Description
`init_t`	`systemd`
`httpd_t`	HTTP daemon threads
`kernel_t`	Kernel threads
`syslogd_t`	`journald` and `rsyslogd` logging daemons
`unconfined_t`	Processes that are started by Oracle Linux users run in the unconfined domain

Multi-Level Security Policy

A Multi-Level Security (MLS) policy applies access controls to multiple levels of processes with each level having different rules for user access. Users can't obtain access to information if they don't have the correct authorization to run a process at a specific level. In SELinux, MLS implements the Bell-LaPadula (BLP) model for system security, which applies labels to files, processes, and other system objects to control the flow of information between security levels. In a typical implementation, the labels for security levels might range from the most secure, `top secret`, through `secret`, and `classified`, to the least secure, `unclassified`. For example, under MLS, you might configure a program labeled `secret` that can write to a file that's labeled `top secret`, but can't read from it. Similarly, you would configure the same program to read from and write to a file labeled `secret`, but only to read `classified` or `unclassified` files. So, information that passes through the program can flow upwards through the hierarchy of security levels, but not downwards.

> **Note:**
>
> You must install the `selinux-policy-mls` package to apply the MLS policy.

> **Note:**
>
> Oracle does not recommend using the MLS policy on a system that is running the X Window System.

> **Note:**
>
> SELinux denials are more common with MLS for the following main reasons:
>
> - MLS disables the unconfined policy module.
> - MLS makes use of sensitivity levels.

Setting or Switching SELinux Policies

> **Note:**
>
> You can't change the policy type of a running system.

You can configure the default policy type by editing the `/etc/selinux/config` file and setting the value of the the `SELINUXTYPE` directive to `targeted` or `mls`.

Before switching from one policy to another, change the SELinux mode to permissive. Relabeling while in enforcing mode may prevent confined domains from accessing files, which would also prevent your system from starting correctly.

Customizing SELinux Policies

You can customize an SELinux policy by enabling or disabling the members of a set of Boolean values. Any changes that you make take effect immediately and do not require a reboot.

To display the Boolean values and their descriptions, use the following command:

```
semanage boolean -l


SELinux boolean                State  Default Description

ftp_home_dir                   (off  ,  off)
Determine whether ftpd can read and write files in user home directories.
```

```
smartmon_3ware                 (off  ,  off)
Determine whether smartmon can support devices on 3ware controllers.
mpd_enable_homedirs            (off  ,  off)
Determine whether mpd can traverse user home directories.
...
```

You can use the `getsebool` and `setsebool` commands to display and set the value of a specific Boolean.

```
getsebool boolean
sudo setsebool boolean on|off
```

The following example shows how you to display and set the value of the `ftp_home_dir` Boolean:

```
getsebool ftp_home_dir
ftp_home_dir --> off
sudo setsebool ftp_home_dir on
getsebool ftp_home_dir
ftp_home_dir --> on
```

To switch the value of a Boolean, use the `togglesebool` command, as shown in the following example:

```
sudo togglesebool ftp_home_dir
ftp_home_dir: inactive
```

To make the value of a Boolean persist across reboots, specify the `-P` option to `setsebool`, for example:

```
sudo setsebool -P ftp_home_dir on
getsebool ftp_home_dir
ftp_home_dir --> on
```

3

Administering SELinux Security Context

Under SELinux, all file systems, files, directories, devices, and processes have an associated security context. For files, SELinux stores a context label in the extended attributes of the file system. The context contains more information about a system object: the SELinux user, their role, their type, and the security level. SELinux uses this context information to control access by processes, Linux users, and files. This chapter provides information about how to administer SELinux Security Context

You can specify the `-Z` option with certain commands (`ls`, `ps`, and `id`) to display the SELinux context by using the following syntax:

`SELinux user``:`*`Role`*`:`*`Type`*`:`*`Level`*

SELinux user
An SELinux user account compliments a regular Linux user account. SELinux maps every Linux user to an SELinux user identity that is used in the SELinux context for the processes in a user session.

Role
In the Role-Based Access Control (RBAC) security model, a role acts as an intermediary abstraction layer between SELinux process domains or file types and an SELinux user. Processes run in specific SELinux domains, and file system objects are assigned SELinux file types. SELinux users are authorized to perform specified roles, and roles are authorized for specified SELinux domains and file types. A user's role defines which process domains and file types the user can access, and hence which processes and files the user can access.

Type
A type defines an SELinux file type or an SELinux process domain. Processes are separated from each other by running in their own domains. This separation prevents processes from accessing files that other processes use, and prevents processes from accessing other processes. The SELinux policy rules define the access that process domains have to file types and to other process domains.

Level
A level is an attribute of Multi-Level Security (MLS) and Multi-Category Security (MCS). An MLS range is a pair of sensitivity levels, written as *`low_level-high_level`*. The range can be abbreviated as *`low_level`* if the levels are identical. For example, `s0` is the same as `s0-s0`. Each level has an optional set of security categories to which it applies. If the set is contiguous, it can be abbreviated. For example, `s0:c0.c3` is the same as `s0:c0,c1,c2,c3`.

Displaying SELinux User Mapping

Display the mapping between SELinux and Linux user accounts by using the `semanage` command.

```
semanage login -l
```

```
Login Name           SELinux User         MLS/MCS Range        Service

__default__          unconfined_u         s0-s0:c0.c1023       *
root                 unconfined_u         s0-s0:c0.c1023       *
system_u             system_u             s0-s0:c0.c1023       *
```

By default, SELinux maps Linux users other than `root` and the default system-level user, `system_u`, to the Linux `__default__` user, and in turn to the SELinux `unconfined_u` user. The MLS/MCS Range is the security level used by Multi-Level Security (MLS) and Multi-Category Security (MCS).

Displaying SELinux Context Information

To display the context information that's associated with files, use the `ls -Z` command:

```
ls -Z
```

```
-rw-------. root root system_u:object_r:admin_home_t:s0 anaconda-ks.cfg
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 config
-rw-r--r--. root root system_u:object_r:admin_home_t:s0 initial-setup-ks.cfg
drwxr-xr-x. root root unconfined_u:object_r:admin_home_t:s0 jail
-rw-r--r--. root root unconfined_u:object_r:admin_home_t:s0 team0.cfg
```

To display the context information that's associated with a specified file or directory, type:

```
ls -Z /etc/selinux/config
```

```
-rw-r--r--. root root system_u:object_r:selinux_config_t:s0 /etc/selinux/
config
```

To display the context information that's associated with processes, use the `ps -Z` command:

```
ps -Z
```

```
LABEL                                          PID  TTY   TIME     CMD
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3038 pts/0 00:00:00 su
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3044 pts/0 00:00:00 bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 3322 pts/0 00:00:00 ps
```

To display the context information that's associated with the current user, use the `id -Z` command:

```
id -Z


unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Changing the Default File Type

Under some circumstances, you might need to change the default file type for a file system hierarchy. For example, you might want to use a `DocumentRoot` directory other than `/var/www/html` with `httpd`.

To change the default file type of the directory hierarchy `/var/webcontent` to `httpd_sys_content_t`:

1. Use the `semanage` command to define the file type `httpd_sys_content_t` for the directory hierarchy:

   ```
   sudo /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "/var/
   webcontent(/.*)?"
   ```

   This command adds the following entry to the file `/etc/selinux/targeted/contexts/files/file_contexts.local`:

   ```
   /var/webcontent(/.*)?     system_u:object_r:httpd_sys_content_t:s0
   ```

2. Use the `restorecon` command to apply the new file type to the entire directory hierarchy.

   ```
   sudo /sbin/restorecon -R -v /var/webcontent
   ```

Restoring the Default File Type

The following steps restore the default file type of the directory hierarchy `/var/webcontent` that has the `httpd_sys_content_t` setting:

1. Use the `semanage` command to delete the file type definition for the directory hierarchy from the file `/etc/selinux/targeted/contexts/files/file_contexts.local`:

   ```
   sudo /usr/sbin/semanage fcontext -d "/var/webcontent(/.*)?"
   ```

2. Use the `restorecon` command to apply the default file type to the entire directory hierarchy.

   ```
   sudo /sbin/restorecon -R -v /var/webcontent
   ```

Relabeling a File System

If you see an error message that contains the string `file_t`, the problem typically lies with a file system having an incorrect context label.

To relabel a file system by using the command line:

1. Create the file `/.autorelabel` and reboot the system.
2. Run the `fixfiles onboot` command, then reboot the system.

4

Administering SELinux Users

As described in Administering SELinux Security Context, each SELinux user account compliments a regular Oracle Linux user account. SELinux maps every Oracle Linux user to an SELinux user identity that's used in the SELinux context for the processes in a user session.

SELinux users form part of a SELinux policy that's authorized for a specific set of roles and for a specific MLS (Multi-Level Security) range, and each Oracle Linux user is mapped to an SELinux user as part of the policy. As a result, Linux users inherit the restrictions and security rules and mechanisms placed on SELinux users. To define the roles and levels of users, the mapped SELinux user identity is used in the SELinux context for processes in a session.

By default, users are mapped to the `unconfined_u` SELinux user when they're created, unless otherwise specified. With that setting, SELinux functions in a nonrestrictive capacity. To improve system security, you can change the default user mapping and start applying different user mappings for different user requirements on the system.

Understanding Confined SELinux Users

SELinux includes several confined users that are restricted to different security domains and that have predefined security rules and mechanisms to control what a user is allowed to do. SELinux policies include rules that apply to the different roles that a user can belong to, and these are used to enforce what operations are allowed to for each SELinux user.

By convention, SELinux users have the suffix `_u`, such as `user_u`.

Oracle Linux includes several SELinux users that are already set up through which you can restrict system access immediately:

`unconfined_u`
A largely unrestricted SELinux user often set as the default SELinux user mapping for system user accounts on new systems in a less restrictive environment. In a hardened environment, no system user accounts must map to this user.

`root`
The SELinux user meant for the root account.

`sysadm_u`
The SELinux user with direct system administrative role assigned. This user isn't intended to run nonadministrative commands.

`staff_u`
The SELinux user for users that need to run both nonadministrative commands (through the `staff_r` role) and administrative commands (through the `sysadm_r` role).

`user_u`
The SELinux user for nonprivileged accounts that don't need to run any administrative commands.

system_u
The SELinux user for system services.

xguest_u
The SELinux user for guest access to a system and provisioned with limited access.

Users are confined to their SELinux domains, and policies control the types of things that they can do on the system. The following table illustrates how certain predefined security rules work for different users.

SELinux User	SELinux Domain	Permit Running su and sudo?	Permit Network Access?	Permit Logging in Using X Window System?	Permit Executing Applications in $HOME and /tmp?
guest_u	guest_t	No	Yes	No	No
staff_u	staff_t	sudo	Yes	Yes	Yes
system_u	ssystem_t	Yes	Yes	Yes	Yes
user_u	user_t	No	Yes	Yes	Yes
xguest_x	xguest_t	No	Firefox only	Yes	No

SELinux users are distinct and managed separately from standard Oracle Linux system users within SELinux. You can map Oracle Linux system user accounts to different SELinux users to apply a more restrictive security policy framework to any of the system user accounts.

Mapping Oracle Linux Users to SELinux Confined Users

By default, users are mapped to the `unconfined_u` SELinux user when they're created, unless otherwise specified. Users can check their security context by running:

```
id -Z
```

Output might be similar to the following example:

```
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

A system administrator can map an Oracle Linux user to an SELinux confined user to apply different levels of access. For example, to map the *oracle* user to the SELinux `user_u` user, use the `semanage` command:

```
sudo semanage login -a -s user_u oracle
```

When you create a user, you can specify the SELinux user mapping when you run the `useradd` command. For example, to add a privileged *oracleadmin* user that maps onto the SELinux `staff_u` user, run:

```
sudo useradd -Z staff_u oracleadmin
```

Setting the Default User Mapping

On most newly installed systems, the default user mapping is set to the `unconfined_u` SELinux user to provide a less restrictive environment for general use. In some environments where strict policy enforcement is required, such as when conforming to a Security Technical Implementation Guide (STIG), you might need to map all Oracle Linux user accounts to appropriate confined SELinux users so that a system is better protected by the SELinux policy rules that you're enforcing.

1. To change the default user mapping so that any user accounts that don't have explicit SELinux user mappings are confined to the SELinux `user_u` user, run:

   ```
   sudo semanage login -m -s user_u -r s0 __default__
   ```

2. Verify that the `__default__` user mapping is no longer set to the `unconfined_u` SELinux user by running:

   ```
   semanage login -l
   ```

Note that the unconfined security context continues to apply to users after this change until the user session or the process is restarted under the new context. To enforce this change at a system-wide level, reboot the system.

Configuring the Behavior of Application Execution for Users

To help prevent flawed or malicious applications from revising a user's files, you can use Boolean values to specify whether users are permitted to run applications in directories for which they have write access, such as the user's home directory hierarchy and `/tmp`.

To enable Oracle Linux users in the `guest_t` and `xguest_t` domains to run applications in directories to which they have write access, thpe:

```
sudo setsebool -P allow_guest_exec_content on
sudo setsebool -P allow_xguest_exec_content on
```

The following example shows how to prevent users in the `staff_t` and `user_t` domains from running applications in directories to which they have write access:

```
sudo setsebool -P allow_staff_exec_content off
sudo setsebool -P allow_user_exec_content off
```

For more information, see Customizing SELinux Policies.

5

Extending SELinux Policies with Multi-Category Security

Multi-Category Security (MCS) extends the SELinux targeted and Multi-Level Security (MLS) policies so you can assign category labels to processes and files. With MCS, files can be accessed only by processes or users that are assigned to the same categories that apply to the file. MCS is applied after all other security checks have been performed. Thus MCS is typically used to further restrict access. Category tags range from `c0` to `c1023`, but you can define text labels for these category values to make them easier to work with. The `mcstrans` service can be used to translate between the category values and text labels when handling system inputs and outputs.

While MLS can be used to define different security levels or sensitivity for data, MCS can be used to group data for different purposes. For example, you might run the same service for several different projects on a system and data within each project that might have different levels of sensitivity. Users must only be granted access to data that meets their sensitivity clearance for a particular project. MCS enforces this restriction by associating a category tag with each project. The resulting security context of a file or process is a combination of SELinux user, SELinux role, SELinux type, MLS sensitivity level, and MCS category.

Table 5-1 Matrix to illustrate data sensitivity and category application

Sensitivity	Category			
	Not specified	Accountancy	Marketing	Development
Unclassified	`s0`	`s0:c0`	`s0:c1`	`s0:c2`
Internal	`s1`	`s1:c0`	`s1:c1`	`s1:c2`
Restricted	`s2`	`s2:c0`	`s2:c1`	`s2:c2`
Highly Restricted	`s3`	`s3:c0`	`s3:c1`	`s3:c2`

In the example table, a highly privileged user in the accountancy department (`c0`) with a requirement to access highly restricted data (`s3`) might have the following security context defined:

```
user_u:user_r:user_t:s3:c0
```

MCS Requirements

Before configuring a system for MCS, fulfill the following requirements:

- SELinux must be configured in `enforcing` mode.
- SELinux must be configured to use either the `targeted` or `mls` policies.
- The `policycoreutils-python-utils` package must be installed so you can use the `chcat` and `semanage` commands.
- The `setools-console` package can be installed to use the `seinfo` command for verification.

- SELinux confined user mappings are typical when using MCS. For example, nonprivileged users are assigned to `user_u`, while privileged users are assigned to `staff_u`. It is helpful to have any user mappings defined before configuring MCS. See Administering SELinux Users.

Enabling MCS for Users

MCS is active by default in SELinux, but isn't configured for users. To configure MCS for users, you must create a policy module that adds a rule to assign the `mcs_constrained_type` attribute to the user domain.

1. Create a file that contains the rule, for example:

   ```
   echo '(typeattributeset mcs_constrained_type (user_t))' >
   local_mcs_user.cil
   ```

2. Load the new policy module.

   ```
   sudo semodule -i local_mcs_user.cil
   ```

3. Verify that the `mcs_constrained_type` is now applied to `user_t` domain.

   ```
   seinfo -xt user_t|grep mcs_constrained_type
   ```

You can add the `mcs_constrained_type` attribute to any other SELinux domain in the same way.

Applying MCS Categories to a User

You can control a user's access to resources by applying MCS categories to the user. You can define category ranges that are available to each SELinux user and you can specify subranges for each Oracle Linux user account that's mapped to an SELinux user. See Administering SELinux Users for more information on the different SELinux users and how to manage mappings between these users and standard Oracle Linux users.

1. To define the category ranges that are available to the SELinux `user_u` user, run:

   ```
   sudo semanage user -m -rs0:c0,c1-s0:c0.c9 user_u
   ```

   Use category numbers `c0` to `c1023`, or category aliases if you are using the `mcstrans` service. In the example, the category range of `c0` to `c9` is assigned to the *user_u* user.

2. For each Oracle Linux user that's mapped to an SELinux user, for which you have defined a category range, you can specify the individual categories that apply. For example, to apply the `c1` category to the *oracle* user you can run:

   ```
   sudo semanage login -m -rs0:c1 oracle
   ```

   The categories that you assign to users must be within the range that you defined for the mapped SELinux user.

You can equally use the `chcat -l` command to modify which categories apply to a user. For example, you can add the `c2` category to *oracle* and remove the `c1` category:

```
sudo chcat -l -- +c2,-c1 oracle
```

The command uses `--` to indicate that the `-` character isn't to be interpreted as an option switch.

See the `chcat(8)` and `semanage-user(8)` manual pages for more information.

Applying MCS Categories to Files

Any user that has access rights to a file can apply an MCS category to the file if the category is assigned to that user. By applying a category to a file, a user can block access to that file for other users on the system that don't have the same category assigned to them. Note that as with all SELinux policies, standard Linux discretionary access controls are also in effect, so even if a user has category access to a file, the user may still be unable to access the file if the file permissions and mode prevent access for that user.

A user can set the categories that apply to a file if the categories that the user sets are also assigned to the user. File categories are set using the `chcat` command. For example, to add the `c1` and `c2` categories to a file, the user can run:

```
chcat -- +c1,+c2 /path/to/file
```

To remove the `c1` category, the user can run:

```
chcat -- -c1 /path/to/file
```

The command uses `--` to indicate that the `-` character isn't to be interpreted as an option switch. See the `chcat(8)` manual page for more information.

You can check which categories are assigned to a file by listing the file's security context:

```
ls -lZ /path/to/file
```

New files and directories, by default, inherit the SELinux type of their parent directories. You can check which categories are assigned to the parent directory of a file by running:

```
ls -dZ /path/to/file
```

Enabling the mcstrans Service

The `mcstrans` service automatically translates MCS category and MLS sensitivity values against a map of human-readable text labels that are defined as editable configuration entries. If you're using a `targeted` policy, the configuration file is in `/etc/selinux/targeted/setrans.conf`. If you're using an `mls` policy, the configuration file is in `/etc/selinux/mls/setrans.conf` or as individual configuration files within `/etc/selinux/mls/setrans.d`.

The `mcstrans` service can make it easier for users to make sense of category and sensitivity values returned by the system for different SELinux outputs and can make it easier to set

appropriate values when defining security contexts. See the `setrans.conf(8)` and `mcstransd(8)` manual pages for more information.

To install and enable the `mcstrans` service, run:

```
sudo dnf install -y mcstrans
sudo enable --now mcstrans
```

If you update any of the `setrans.conf` files to create custom mappings, you must restart the `mcstrans` service:

```
sudo systemctl restart mcstrans
```

You can verify that translations are applied by running:

```
chcat -L
```

The command returns a list of the current mappings applied by the `mcstrans` service.

6
Troubleshooting Access-Denial Messages

This chapter provides information about how to troubleshoot access-denial messages.

The decisions that SELinux makes about allowing and denying access are stored in the Access Vector Cache (AVC). If the auditing service (`auditd`) isn't running, SELinux logs AVC denial messages to `/var/log/messages`. Otherwise, the messages are logged to the `/var/log/audit/audit.log` file. If the `setroubleshootd` daemon is running, easier-to-read versions of the denial messages are also written to `/var/log/messages`.

If you have installed the `setroubleshoot` and `setroubleshoot-server` packages, the `auditd` and `setroubleshoot` services are running. If you're using the X Window System, you can also use the `sealert -b` command to run the SELinux Alert Browser, which displays information about SELinux AVC denials. To view the details of the alert, click **Show**. To view a recommended solution, click **Troubleshoot**.

The following example shows how you would search the `/var/log/audit/audit.log` file for messages containing the string `denied`:

```
grep denied /var/log/audit/audit.log
type=AVC msg=audit(1364486257.632:26178): avc:  denied  { read } for
pid=5177 comm="httpd" name="index.html" dev=dm-0 ino=396075
scontext=unconfined_u:system_r:httpd_t:s0
tcontext=unconfined_u:object_r:acct_data_t:s0 tclass=file
```

The main causes of access-denial problems include the following:

- Context labels for an application or file are incorrect.

  A solution might be to change the default file type of the directory hierarchy. For example, change the default file type from `/var/webcontent` to `httpd_sys_content_t`:

  ```
  sudo /usr/sbin/semanage fcontext -a -t httpd_sys_content_t "/var/
  webcontent(/.*)?"
  sudo /sbin/restorecon -R -v /var/webcontent
  ```

- A Boolean that configures a security policy for a service is set incorrectly.

  A solution might be to change the value of a Boolean. For example, you can open users' home directories to be browsable by turning on `httpd_enable_homedirs`:

  ```
  sudo setsebool -P httpd_enable_homedirs on
  ```

- A service is accessing a port to which a security policy prohibits access.

  If the service's use of the port is valid, a solution is to use `semanage` to add the port to the policy configuration. For example, to set the Apache HTTP server to listen on port 8000:

  ```
  sudo semanage port -a -t http_port_t -p tcp 8000
  ```

- An update to a package causes an application to behave in a way that breaks an existing security policy.

  You can use the `audit2allow -w -a` command to view the reason why an access denial occurred.

  If you then run the `audit2allow -a -M module` command, it creates a type enforcement (`.te`) file and a policy package (`.pp`) file. You can use the policy package file with the `semodule -i module.pp` command to stop the error from reoccurring. This procedure is typically intended to make package updates function until an updated policy is available. If used incorrectly, you can create potential security holes on the system.