SELinux

Secure MariaDB Server with SELinux. This section guides you through configuring SELinux policies to enhance the security posture of your MariaDB deployments on Linux systems.

Security-Enhanced Linux (SELinux) is a Linux kernel module that provides a framework for configuring mandatory access control (MAC) system for many resources on the system. It is enabled by default on some Linux distributions, including RHEL, CentOS, Fedora, and other similar Linux distribution. SELinux prevents programs from accessing files, directories or ports unless it is configured to access those resources.

Verifying Whether SELinux Is Enabled

To verify whether SELinux is enabled, execute the getenforce command. For example:

```
getenforce
```

Temporarily Putting mysqld Into Permissive Mode

When you are troubleshooting issues that you think SELinux might be causing, it can help to temporarily put `mysqld_t` into permissive mode. This can be done by executing the semanage command. For example:

```
sudo semanage permissive -a mysqld_t
```

If that solved the problem, then it means that the current SELinux policy is the culprit. You need to adjust the SELinux policy or labels for MariaDB.

Configuring a MariaDB Server SELinux Policy

MariaDB Server should work with your default distribution policy (which is usually part of the `selinux-policy` or `selinux-policy-targeted` system package). If you use `mysqld_safe`, you will need an additional policy file, `mariadb.pp`, which is installed together with the MariaDB Server. It will be loaded automatically if you have `/usr/sbin/semodule` installed, but you can load it manually anytime with

```
/usr/sbin/semodule -i /usr/share/mariadb/policy/selinux/mariadb.pp
```

Note that this policy file extends, but not replaces the system policy.

Setting File Contexts

SELinux uses file contexts as a way to determine who should be able to access that file.

File contexts are managed with the semanage fcontext and restorecon commands.

On many systems, the semanage utility is installed by the `policycoreutils-python` package, and the restorecon utility is installed by the `policycoreutils` package. You can install these with the following command:

```
sudo yum install policycoreutils policycoreutils-python
```

A file or directory's current context can be checked by executing `ls` with the `--context` or `--scontext` options.

Setting the File Context for the Data Directory

If you use a custom directory for datadir, then you may need to set the file context for that directory. The SELinux file context for MariaDB data files is `mysqld_db_t`. You can determine if this file context is present on your system and which files or directories it is associated with by executing the following command:

```
sudo semanage fcontext --list | grep mysqld_db_t
```

If you would like to set the file context for your custom directory for your datadir, then that can be done by executing the semanage fcontext ↗ and restorecon ↗ commands. For example:

```
sudo semanage fcontext -a -t mysqld_db_t "/mariadb/data(/.*)?"
sudo restorecon -Rv /mariadb/data
```

If you would like to check the current file context, you can do so by executing `ls` with the `--context` or `--scontext` options. For example:

```
ls --directory --scontext /mariadb/data
```

Setting the File Context for Log Files

If you use a custom directory for log files, then you may need to set the file context for that directory. The SELinux file context for MariaDB log files is `mysqld_log_t`. You can determine if this file context is present on your system and which files or directories it is associated with by executing the following command:

```
sudo semanage fcontext --list | grep mysqld_log_t
```

If you would like to set the file context for your custom directory for log files, then that can be done by executing the semanage fcontext and restorecon commands. For example:

```
sudo semanage fcontext -a -t mysqld_log_t "/var/log/mysql(/.*)?"
sudo restorecon -Rv /var/log/mysql
```

If you would like to check the current file context, you can do so by executing `ls` with the `--context` or `--scontext` options. For example:

```
ls --directory --scontext /var/log/mysql
```

Setting the File Context for Option Files

If you use a custom directory for option files, then you may need to set the file context for that directory. The SELinux file context for MariaDB option files is `mysqld_etc_t`. You can determine if this file context is present on your system and which files or directories it is associated with by executing the following command:

```
sudo semanage fcontext --list | grep mysqld_etc_t
```

If you would like to set the file context for your custom directory for option files, then that can be done by executing the semanage fcontext and restorecon commands. For example:

```
sudo semanage fcontext -a -t mysqld_etc_t "/etc/mariadb(/.*)?"
sudo restorecon -Rv /etc/mariadb
```

If you would like to check the current file context, you can do so by executing `ls` with the `--context` or `--scontext` options. For example:

```
ls --directory --scontext /etc/mariadb
```

Setting the Unix Socket

A custom location for the socket means it needs to have the right file context of `mysqld_var_run_t` for permitted application to connect to the socket.

```
sudo semanage fcontext -a -t mysqld_var_run_t "/mariadb/run/
mariadb.sock"
```

A newly created socket with get the right context.

Allowing Access to the Tmpfs File Context

If you wanted to mount your tmpdir on a `tmpfs` file system or wanted to use a `tmpfs` file system on `/run/shm`, then you might need to allow `mysqld_t` to have access to a couple tmpfs-related file contexts. For example:

```
cd /usr/share/mysql/policy/selinux/
tee ./mysqld_tmpfs.te <<EOF
module mysqld_tmpfs 1.0;

require {
type tmpfs_t;
type mysqld_t;
class dir { write search read remove_name open getattr add_name };
class file { write getattr read lock create unlink open };
}

allow mysqld_t tmpfs_t:dir { write search read remove_name open
getattr add_name };

allow mysqld_t tmpfs_t:file { write getattr read lock create
unlink open }
EOF
sudo checkmodule -M -m mysqld_tmpfs.te -o mysqld_tmpfs.mod
sudo semodule_package -m mysqld_tmpfs.mod -o mysqld_tmpfs.pp
sudo semodule -i mysqld_tmpfs.pp
```

Troubleshooting SELinux Issues

You might need to troubleshoot SELinux-related issues in cases, such as:

- MariaDB is using a non-default port.
- MariaDB is reading from or writing to some files (datadir, log files, option files, etc.) located at non-default paths.
- MariaDB is using a plugin that requires access to resources that default installations do not use.

File System Permission Errors

If the file system permissions for some MariaDB directory look fine, but the MariaDB error log still has errors that look similar to the following:

```
130321 11:50:51 mysqld_safe Starting mysqld daemon with databases
from /datadir
...
2013-03-21 11:50:52 2119 [Warning] Can't create test file /
datadir/
2013-03-21 11:50:52 2119 [Warning] Can't create test file /
datadir/
...
2013-03-21 11:50:52 2119 [ERROR] /usr/sbin/mysqld: Can't create/
write to file
    '/datadir/boxy.pid' (Errcode: 13 - Permission denied)
2013-03-21 11:50:52 2119 [ERROR] Can't start server: can't create
PID file:
    Permission denied
130321 11:50:52 mysqld_safe mysqld from pid file /datadir/boxy.pid
ended
```

Then check SELinux's `/var/log/audit/audit.log` for log entries that look similar to the following:

```
type=AVC msg=audit(1363866652.030:24): avc:  denied  { write } for
pid=2119
    comm="mysqld" name="datadir" dev=dm-0 ino=394
    scontext=unconfined_u:system_r:mysqld_t:s0
    tcontext=unconfined_u:object_r:default_t:s0 tclass=dir
```

If you see any entries that look similar to this, then you most likely need to adjust the file contexts for some files or directories. See Setting File Contexts for more information on how to do that.

SELinux and MariaDB On a Different Port

TCP and UDP ports are enabled for permission to bind too. If you are using a different port, or some Galera ports, configure SELinux to be able to use those ports:

```
sudo semanage port -a -t mysqld_port_t -p tcp 3307
```

Generating SELinux Policies with audit2allow

In some cases, a MariaDB system might need non-standard policies. It is possible to create these policies from the SELinux audit log using the audit2allow utility. The semanage and semodule utilities will also be needed.

On many systems, the audit2allow and semanage utilities are installed by the `policycoreutils-python` package, and the semodule utility is installed by the `policycoreutils` package. You can install these with the following command:

```
sudo yum install policycoreutils policycoreutils-python
```

The following process can be used to generate a policy from the audit log:

- Remove dontaudits from the policy:

```
sudo semodule -DB
```

- Temporarily put `mysqld_t` into permissive mode. For example:

```
sudo semanage permissive -a mysqld_t
```

- Start MariaDB.
- Do whatever was causing SELinux errors.
- Use the generated audit log to create a policy:

```
sudo grep mysqld /var/log/audit/audit.log | audit2allow -M
mariadb_local
sudo semodule -i mariadb_local.pp
```

- Pull `mysqld_t` out of permissive mode. For example:

```
sudo semanage permissive -d mysqld_t
```

- Restore dontaudits for the policy:

```
sudo setmodule -B
```

The same procedure can be used if MariaDB starts but SELinux prevents it from functioning correctly. For example, SELinux may prevent PAM plugin from authenticating users. The solution is the same — enable auditing, switch to permissive, do, whatever SELinux didn't allow you to, create a policy from the audit log.

When you discover any needed SELinux permissions, please report the needed permissions to your operating system bug tracking so all users can benefit from your work (e.g. Red Hat Bugzilla).

Previous
Running mariadbd as root

Next
Encryption

Last updated 16 days ago

Was this helpful?

Products

Enterprise Platform

Community Server

Download MariaDB

Pricing

Support

Customer Login

Technical Support

Remote DBA

Professional Services

Resources

MariaDB Blog

Webinars

Customer Stories

MariaDB Events

Documentation

Developer Hub

Company

About MariaDB

Newsroom

Leadership

MariaDB Careers

Legal

Privacy Policy