How To Reset Your MySQL or MariaDB Root Password on Ubuntu 18.04

Posted September 4, 2018 13.4k MYSQL MARIADB DATABASES SECURITY UBUNTU 18.04

By: Mateusz Papiernik

Introduction

Forgetting passwords happens to the best of us. If you forget or lose the **root** password to your MySQL or MariaDB database, you can still gain access and reset the password if you have access to the server and a user account with `sudo` privileges.

> **Note:** On fresh Ubuntu 18.04 installations, the default MySQL or MariaDB configuration usually allows you to access the database (with full administrative privileges) without providing a password as long as you make the connection from the system's **root** account. In this scenario it may not be necessary to reset the password. Before you proceed with resetting your database **root** password, try to access the database with the `sudo mysql` command. If this results in an *access denied* error, follow the steps in this tutorial.

This tutorial demonstrates how to reset the **root** password for MySQL and MariaDB databases installed with the `apt` package manager on Ubuntu 18.04. The procedure for changing the root password differs depending on whether you have MySQL or MariaDB installed, as well as the default systemd configuration that ships with the distribution or packages from other vendors. While the instructions in this tutorial may work with other system or database server versions, they have been tested specifically with Ubuntu 18.04 and distribution-supplied packages.

Prerequisites

To recover your MySQL or MariaDB **root** password, you will need:

- Access to the Ubuntu 18.04 server running MySQL or MariaDB with a sudo user or other way of accessing the server with root privileges. In order to try out the recovery methods in this tutorial without affecting your production server, use the initial server setup tutorial to create a test server with a regular, non-root user with sudo privileges. Then install MySQL following How to install MySQL on Ubuntu 18.04.

Step 1 — Identifying the Database Version and Stopping the Server

Ubuntu 18.04 runs either MySQL or MariaDB, a popular drop-in replacement which is fully compatible with MySQL. You'll need to use different commands to recover the `root` password depending on which of these you have installed, so follow the steps in this section to determine which database server you're running.

Check your version with the following command:

```
$ mysql --version
```

If you're running MariaDB, you'll see "MariaDB" preceded by the version number in the output:

```
MariaDB output
mysql  Ver 15.1 Distrib 10.1.29-MariaDB, for debian-linux-gnu (x86_64) using readline 5.2
```

You'll see output like this if you're running MySQL:

```
MySQL output
mysql  Ver 14.14 Distrib 5.7.22, for Linux (x86_64) using  EditLine wrapper
```

Make note of which database, as this determines the appropriate commands to follow in the rest of this tutorial.

In order to change the **root** password, you'll need to shut down the database server. If you're running MariaDB, you can do so with the following command:

```
$ sudo systemctl stop mariadb
```

For MySQL, shut down the database server by running:

```
$ sudo systemctl stop mysql
```

With the database stopped, you can restart it in safe mode to reset the root password.

Step 2 — Restarting the Database Server Without Permission Checks

Running MySQL and MariaDB without permission checking allows accessing the database command line with root privileges without providing a valid password. To do this, you need to stop the

loading the *grant tables*, which store user privilege information. Since this is a bit of a security risk, you may also want to disable networking to prevent other clients from connecting to the temporarily vulnerable server.

Depending on which database server you've installed, the way of starting the server without loading the *grant tables* differs.

Configuring MariaDB to Start Without Grant Tables

In order to start the MariaDB server without the grant tables, we'll use the `systemd` unit file to set additional parameters for the MariaDB server daemon.

Execute the following command which sets the MYSQLD_OPTS environment variable used by MariaDB upon startup. The `--skip-grant-tables` and `--skip-networking` options tell MariaDB to start up without loading the grant tables or networking features:

```
$ sudo systemctl set-environment MYSQLD_OPTS="--skip-grant-tables --skip-networking"
```

Then start the MariaDB server:

```
$ sudo systemctl start mariadb
```

This command won't produce any output, but it will restart the database server, taking into account the new environment variable settings.

You can ensure it started with `sudo systemctl status mariadb`.

Now you should be able to connect to the database as the MariaDB **root** user without supplying a password:

```
$ sudo mysql -u root
```

You'll immediately see a database shell prompt:

```
MariaDB prompt
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Now that you have access to the database server, you can change the **root** password as shown in Step 3.

Configuring MySQL to Start Without Grant Tables

In order to start the MySQL server without its grant tables, you'll alter the systemd configuration for MySQL to pass additional command-line parameters to the server upon startup.

To do this, execute the following command:

```
$ sudo systemctl edit mysql
```

This command will open a new file in the `nano` editor, which you'll use to edit MySQL's *service overrides*. These change the default service parameters for MySQL. This file will be empty, so add the following content:

MySQL service overrides

```
[Service]
ExecStart=
ExecStart=/usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid --skip-grant-tables --skip-n
```

The first `ExecStart` statement clears the default value, while the second one provides `systemd` with the new startup command including parameters to disable loading the grant tables and networking capabilities.

Press `CTRL-x` to exit the file, then `Y` to save the changes that you made, then `ENTER` to confirm the file name.

Reload the `systemd` configuration to apply these changes:

```
$ sudo systemctl daemon-reload
```

Now start the MySQL server:

```
$ sudo systemctl start mysql
```

The command will show no output, but the database server will start. The grant tables and networking will not be enabled.

Connect to the database as the root user:

```
$ sudo mysql -u root
```

You'll immediately see a database shell prompt:

MySQL prompt

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql>
```

Now that you have access to the server, you can change the **root** password.

Step 3 — Changing the Root Password

The database server is now running in a limited mode; the grant tables are not loaded and there's no networking support enabled. This lets you access the server without providing a password, but it prohibits you from executing commands that alter data. To reset the **root** password, you must load the grant tables now that you've gained access to the server.

Tell the database server to reload the grant tables by issuing the `FLUSH PRIVILEGES` command.

```
mysql> FLUSH PRIVILEGES;
```

You can now change the **root** password. The method you use depends on whether you are using MariaDB or MySQL.

Changing the MariaDB Password

If you are using MariaDB, execute the following statement to set the password for the **root** account, making sure to replace `new_password` with a strong new password that you'll remember.

```
MariaDB [(none)]> UPDATE mysql.user SET password = PASSWORD('new_password') WHERE user = 'root';
```

You'll see this output indicating that the password changed:

```
Output
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

MariaDB allows using custom authentication mechanisms, so execute the following two statements to make sure MariaDB will use its default authentication mechanism for the new password you assigned to the **root** account:

```
MariaDB [(none)]> UPDATE mysql.user SET authentication_string = '' WHERE user = 'root';
MariaDB [(none)]> UPDATE mysql.user SET plugin = '' WHERE user = 'root';
```

You'll see the following output for each statement:

```
Output
Query OK, 0 rows affected (0.01 sec)
Rows matched: 1  Changed: 0  Warnings: 0
```

The password is now changed. Type `exit` to exit the MariaDB console and proceed to Step 4 to restart the database server in normal mode.

Changing the MySQL Password

For MySQL, execute the following statement to change the **root** user's password, replacing `new_password` with a strong password you'll remember:

```
mysql> UPDATE mysql.user SET authentication_string = PASSWORD('new_password') WHERE user = 'root';
```

You'll see this output indicating the password was changed successfully:

```
Output
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

MySQL allows using custom authentication mechanisms, so execute the following statement to tell MySQL touse its default authentication mechanism to authenticate the **root** user using the new password:

```
mysql> UPDATE mysql.user SET plugin = 'mysql_native_password' WHERE user = 'root';
```

You'll see output similar to the previous command:

```
Output
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

The password is now changed. Exit the MySQL console by typing `exit`.

Let's restart the database in normal operational mode.

Step 4 — Reverting Your Database Server to Normal Settings

In order to restart the database server in its normal mode, you have to revert the changes you made so that networking is enabled and the grant tables are loaded. Again, the method you use depends on whether you used MariaDB or MySQL.

For MariaDB, unset the `MYSQLD_OPTS` environment variable you set previously:

```
$ sudo systemctl unset-environment MYSQLD_OPTS
```

Then, restart the service using `systemctl`:

```
$ sudo systemctl restart mariadb
```

For MySQL, remove the modified systemd configuration:

```
$ sudo systemctl revert mysql
```

You'll see output similar to the following:

```
Output
Removed /etc/systemd/system/mysql.service.d/override.conf.
Removed /etc/systemd/system/mysql.service.d.
```

Then, reload the systemd configuration to apply the changes:

```
$ sudo systemctl daemon-reload
```

Finally, restart the service:

```
$ sudo systemctl restart mysql
```

The database is now restarted and is back to its normal state. Confirm that the new password works by logging in as the **root** user with a password:

```
$ mysql -u root -p
```

You'll be prompted for a password. Enter your new password and you'll gain access to the database prompt as expected.

Conclusion

You have restored administrative access to the MySQL or MariaDB server. Make sure the new password you chose is strong and secure and keep it in a safe place.

For more information on user management, authentication mechanisms, or ways of resetting database password for other version of MySQL or MariaDB, please refer to the official MySQL documentation or MariaDB documentation.

Related Tutorials

How To Ensure Code Quality with SonarQube on Ubuntu 18.04

How To Sync and Share Your Files with Seafile on Ubuntu 18.04

How To Troubleshoot Issues in MySQL

How To Set Up a Remote Database to Optimize Site Performance with MySQL on Ubuntu 18.04

How To Set Up Laravel, Nginx, and MySQL with Docker Compose

0 Comments

Leave a comment...

Log In to Comment