How To Install the Latest MySQL on Ubuntu 18.04

Posted July 12, 2018 24.7k MYSQL DATABASES UBUNTU UBUNTU 18.04

By: Brian Boucheron

Not using **Ubuntu 18.04**? Choose a different version:

Introduction

MySQL is a prominent open source database management system used to store and retrieve data for a wide variety of popular applications. MySQL is the **M** in the *LAMP* stack, a commonly used set of open source software that also includes Linux, the Apache web server, and the PHP programming language.

In order to use newly released features, it's sometimes necessary to install a more up-to-date version of MySQL than that provided by your Linux distribution. Conveniently, the MySQL developers maintain their own software repository we can use to easily install the latest version and keep it up to date.

To install the latest version of MySQL, we'll add this repository, install the MySQL software itself, secure the install, and finally we'll test that MySQL is running and responding to commands.

Prerequisites

Before starting this tutorial, you will need:

- An Ubuntu 18.04 server with a non-root, sudo-enabled user, as described in this Ubuntu 18.04 server setup tutorial.

Step 1 — Adding the MySQL Software Repository

The MySQL developers provide a `.deb` package that handles configuring and installing the official MySQL software repositories. Once the repositories are set up, we'll be able to use Ubuntu's standard `apt` command to install the software. We'll download this `.deb` file with `curl` and then install it with the `dpkg` command.

First, load the MySQL download page in your web browser. Find the **Download** button in the lower-right corner and click through to the next page. This page will prompt you to log in or sign up for an Oracle web account. We can skip that and instead look for the link that says **No thanks, just start my download**. Right-click the link and select **Copy Link Address** (this option may be worded differently, depending on your browser).

Now we're going to download the file. On your server, move to a directory you can write to:

```
$ cd /tmp
```

Download the file using `curl`, remembering to paste the address you just copied in place of the highlighted portion below:

```
$ curl -OL https://dev.mysql.com/get/mysql-apt-config_0.8.10-1_all.deb
```

We need to pass two command line flags to `curl`. `-O` instructs `curl` to output to a file instead of standard output. The `L` flag makes `curl` follow HTTP redirects, necessary in this case because the address we copied actually redirects us to another location before the file downloads.

The file should now be downloaded in our current directory. List the files to make sure:

```
$ ls
```

You should see the filename listed:

```
Output
mysql-apt-config_0.8.10-1_all.deb
. . .
```

Now we're ready to install:

```
$ sudo dpkg -i mysql-apt-config*
```

`dpkg` is used to install, remove, and inspect `.deb` software packages. The `-i` flag indicates that we'd like to install from the specified file.

During the installation, you'll be presented with a configuration screen where you can specify which version of MySQL you'd prefer, along with an option to install repositories for other MySQL-related tools. The defaults will add the repository information for the latest stable version of MySQL and nothing else. This is what we want, so use the down arrow to navigate to the `Ok` menu option and hit `ENTER`.

The package will now finish adding the repository. Refresh your `apt` package cache to make the new software packages available:

```
$ sudo apt update
```

Let's also clean up after ourselves and delete the file we downloaded:

```
$ rm mysql-apt-config*
```

Now that we've added the MySQL repositories, we're ready to install the actual MySQL server software. If you ever need to update the configuration of these repositories, just run `sudo dpkg-reconfigure mysql-apt-config`, select new options, and then `sudo apt update` to refresh your package cache.

Step 2 — Installing MySQL

Having added the repository and with our package cache freshly updated, we can now use `apt` to install the latest MySQL server package:

```
$ sudo apt install mysql-server
```

`apt` will look at all available `mysql-server` packages and determine that the MySQL provided package is the newest and best candidate. It will then calculate package dependencies and ask you to approve the installation. Type `y` then `ENTER`. The software will install.

You will be asked to set a **root** password during the configuration phase of the installation. Be sure to choose a secure password. After you enter it twice and hit `ENTER`, you will be prompted to configure an authentication plugin. The default of **Use Strong Password Encryption** is recommended, so hit `ENTER` to choose it. The installation process will continue until completion.

MySQL should now be installed and running. Let's check using `systemctl`:

```
$ systemctl status mysql
```

```
Output
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2018-07-12 17:46:42 UTC; 17s ago
     Docs: man:mysqld(8)
           http://dev.mysql.com/doc/refman/en/using-systemd.html
 Main PID: 7918 (mysqld)
   Status: "SERVER_OPERATING"
    Tasks: 37 (limit: 1152)
   CGroup: /system.slice/mysql.service
           └─7918 /usr/sbin/mysqld
```

The `Active: active (running)` line means MySQL is installed and running. Now we'll make the installation a little more secure.

Step 3 — Securing MySQL

MySQL comes with a command we can use to perform a few security-related updates on our new install. Let's run it now:

```
$ mysql_secure_installation
```

This will ask you for the MySQL **root** password that you set during installation. Type it in and press `ENTER`. Now we'll answer a series of yes or no prompts. Let's go through them:

First, we are asked about the **validate password plugin**, a plugin that can automatically enforce certain password strength rules for your MySQL users. Enabling this is a decision you'll need to make based on your individual security needs. Type `y` and `ENTER` to enable it, or just hit `ENTER` to skip it. If enabled, you will also be prompted to choose a level from 0–2 for how strict the password validation will be. Choose a number and hit `ENTER` to continue.

Next you'll be asked if you want to change the **root** password. Since we just created the password when we installed MySQL, we can safely skip this. Hit `ENTER` to continue without updating the password.

The rest of the prompts can be answered **yes**. You will be asked about removing the **anonymous** MySQL user, disallowing remote **root** login, removing the **test** database, and reloading privilege tables to ensure the previous changes take effect properly. These are all a good idea. Type `y` and hit `ENTER` for each.

The script will exit after all the prompts are answered. Now our MySQL installation is reasonably secured. Let's test it again by running a client that connects to the server and returns some information.

Step 4 – Testing MySQL

`mysqladmin` is a command line administrative client for MySQL. We'll use it to connect to the server and output some version and status information:

```
$ mysqladmin -u root -p version
```

The `-u root` portion tells `mysqladmin` to log in as the MySQL **root** user, `-p` instructs the client to ask for a password, and `version` is the actual command we want to run.

The output will let us know what version of the MySQL server is running, its uptime, and some other status information:

```
Output
```

```
mysqladmin  Ver 8.0.11 for Linux on x86_64 (MySQL Community Server - GPL)
Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Server version      8.0.11
Protocol version    10
Connection      Localhost via UNIX socket
UNIX socket     /var/run/mysqld/mysqld.sock
Uptime:         2 min 21 sec

Threads: 2  Questions: 10  Slow queries: 0  Opens: 136  Flush tables: 2  Open tables: 112  Queries pe
```

If you received similar output, congrats! You've successfully installed the latest MySQL server and secured it.

Conclusion

You've now completed a basic install of the latest version of MySQL, which should work for many popular applications. If you have more advanced needs you might continue with some other configuration tasks:

- If you'd like a graphical interface for administering your MySQL server, phpMyAdmin is a popular web-based solution. Our tutorial How To Install and Secure phpMyAdmin can get you started.
- Currently, your database is only accessible to applications running on the same server. Sometimes you'll want separate database and application servers, for performance and storage reasons. Take a look at How To Configure SSL/TLS for MySQL to learn how to set up MySQL for secure access from other servers.
- Another common configuration is to change the directory where MySQL stores its data. You'll need to do this if you want your data stored on a different storage device than the default directory. This is covered in How To Move a MySQL Data Directory to a New Location.

By: Brian Boucheron

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