Installing MariaDB on Ubuntu 10.10 (Maverick Meerkat)

Table of Contents

MariaDB is the version of the MySQL database server developed by the company founded by Michael Widenius, the originator of MySQL. You can do everything with MariaDB that you can do with MySQL. If the truth be told, you can do a whole lot more—because you won't be working in the shadow of Larry Ellison.

This article addresses a very specific issue—the installation of the binary version of the MariaDB database server on Ubuntu 10.10. There is no `.deb` package available so you must install the generic Linux binaries or compile from source code. If you can compile from source code, you don't need to read this article though you might find it useful. This article is for users who typically install applications using the package manager. (Don't we all if we get the chance?) Some familiarity with working from the command line is assumed.

Installation

To obtain the Linux binaries for use with Ubuntu 10.10, navigate to http://askmonty.org/wiki/MariaDB:Download and find the generic binary file suitable for your architecture. The file you want bears the name `mariadb-version-Linux-arch.tar.gz`.

Download this file and decompress it by entering the following commands:

```
shell> cd /path/to/downloads
shell> tar zxf mariadb-version-Linux-arch.tar.gz
```

This creates a directory with the name `mariadb-version-Linux-arch`. Navigate to this directory and you'll see the `INSTALL_BINARY` file. This file contains instructions for installing the binaries. We'll adapt these instructions for Ubuntu.

A `mysql` user and group is added automatically when you install software using the package manager but when you install from the binaries you must perform this task manually. These tasks require root privileges so you must use `sudo` on Ubuntu:

```
shell> sudo groupadd mysql
shell> sudo useradd -g mysql mysql
```

These commands create a group called `mysql` and a user of the same name who is a member of this group.

For easiest installation the MariaDB binaries need to be installed below the `/usr/local` directory. Since these files have already been decompressed, move them from the download directory to `/usr/local`.

```
shell> cd /path/to/downloads
shell> sudo mv mariadb-version-Linux-arch /usr/local
```

Because the directory name is so unwieldy, create a link to it so that we can reference `mysql` in its place.

```
shell> cd /usr/local
shell> sudo ln -s mariadb-version-Linux-arch mysql
```

Now `mysql` can stand in for `mariadb-version-Linux-arch`. You can confirm this by listing the contents of the current directory issuing the command **sudo ls -al**. This should display something similar to the following:

```
...
drwxr-xr-x 12 peter peter 4096 2010-11-11 09:26 mariadb-version-Linux-arch
lrwxrwxrwx  1 root  root    27 2010-11-11 09:29 mysql ->
mariadb-version-Linux-arch
...
```

The line indicating the linked directory begins with an 'l'. Also notice that this directory is owned by root. This needs to change in order for MariaDB to function properly. Execute the following commands to change the ownership of all files and directories in the `mysql` directory:

```
shell> cd mysql
shell> sudo chown mysql ./ -R
shell> sudo chgrp mysql ./ -R
```

With ownership changed, the installation script can be executed.

```
shell> sudo scripts/mysql_install_db --user=mysql
```

Executing this script creates a directory named `data` owned by the `mysql` user.

You should now check that the MariaDB server works by starting it up using the command **sudo ./bin/mysqld_safe &**.

Confirm that it has started by connecting to the MariaDB server using the **mysql** client command:

```
shell> ./bin/mysql
```

If you connect to the server, you should see output such as the following:

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 5.1.50-MariaDB (MariaDB - http://MariaDB.com/)

This software comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to modify and redistribute it under the GPL v2 license
```

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

MariaDB [none]>
```

Quit the client by entering \q.

Post-Installation

At this point you've installed the database server and confirmed that it runs. At the very least you will want the database server to start up automatically and to add the MariaDB binaries to your PATH variable. This section covers those tasks.

Adding the MariaDB Executables to Your Path

The MariaDB executables are found in the /usr/local/mysql/bin directory and since this directory has been newly created, it won't be in your PATH variable. Whenever you use any of the utilities in this directory you'll need to specify the full path.

To avoid doing this alter the PATH variable in the .bashrc file in your home directory. If the PATH variable already exists, append /usr/local/mysql/bin: to it. If not, add the following lines to .bashrc:

```
PATH=$PATH:/usr/local/mysql/bin:
export PATH
```

Note

Files that begin with a '.' are hidden files on Unix-like systems. If you list the contents of your home directory, you won't see the .bashrc file unless you use the -a option.

The next time you open a terminal window execute **echo $PATH** and confirm that /usr/local/mysql/bin is now included.

Note

The many MySQL command line programs such as **mysqldump** and **mysqladmin** are also found in /usr/local/mysql/bin and, as a result of adding /usr/local/mysql to the path, are accessible without specifying the full path.

Starting MariaDB Automatically

You probably want to start MariaDB automatically whenever your machine boots up. If you look at the INSTALL-BINARY file you'll be referred to Section 2.13.1.2, "Starting and Stopping MariaDB Automatically." You won't find this section in the INSTALL-BINARY file but no matter. The following instructions specify how to do this on Ubuntu 10.10.

To start MariaDB on reboot you need to create an initialization script in the /etc/init.d directory and make sure that it executes at the correct run levels. The script is ready made for you. It's called mysql.server and you'll find it in the /usr/local/mysql/support-files directory.

Here are the commands you need to copy mysql.server to /etc/init.d/mysql and make it executable:

```
shell> cd /usr/local/mysql/support-files
shell> sudo cp mysql.server /etc/init.d/mysql
shell> sudo chmod +x /etc/init.d/mysql
```

Next use the **update-rc.d** command to create the proper symlinks for the different run levels. Do this in the following way:

```
shell> sudo update-rc.d mysql defaults
```

If you are successful, you should see output like the following:

```
  Adding system startup for /etc/init.d/mysql ...
  /etc/rc0.d/K20mysql -> ../init.d/mysql
  /etc/rc1.d/K20mysql -> ../init.d/mysql
  /etc/rc6.d/K20mysql -> ../init.d/mysql
  /etc/rc2.d/S20mysql -> ../init.d/mysql
  /etc/rc3.d/S20mysql -> ../init.d/mysql
  /etc/rc4.d/S20mysql -> ../init.d/mysql
  /etc/rc5.d/S20mysql -> ../init.d/mysql
```

You can test that MariaDB starts automatically by rebooting and then running the client program, `mysql`—the server must be running in order for the client programme to connect to it. Since `/usr/local/mysql/bin` has been added to your path, you don't need to specify the full path to start up the client.

Adding the GRANT Tables and a New User

At this point the MariaDB server will start automatically on reboot but you still need to install the GRANT tables. To do this run the **/usr/local/mysql/scripts/mysql_secure_installation** script with root privileges. This script requires interaction from the user.

The script first asks for root's password. Note that this is the root password for the database server not the operating system password. Currently there is no password for root so just press the **enter** key. You will then be asked whether you wish to give the root user a password. Since this is not a production server security considerations aren't paramount but giving the root user a password is a good idea. Make sure that you make note of this password. Allow remote access as you see fit. However, do remove the anonymous user—in some circumstances the existence of this user is not only insecure but confusing.

After running this script you'll find that you can no longer connect to the server by simply entering the command **mysql**. Try it and you should see output similar to the following:

```
ERROR 1045 (28000): Access denied for user 'username'@'localhost' ...
```

You'll have to specify a username and password. Start up the database client by entering: **mysql --user root --password**. If you created a password for the root user, enter it when prompted.

Once you're logged in enter the following commands.

```
MariaDB [(none)]> CREATE USER 'your_username'@'%' IDENTIFIED BY 'password';
Query OK, 0 rows affected (0.00 sec)
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'your_username'@'%';
Query OK, 0 rows affected (0.00 sec)
```

Note that you must have quotation marks around the username, the '%' sign and the password. These commands create a new user with the right to log in from any host. This can be very useful for transferring data from other nodes on your network. However, always specifying a password at the command line can become tedious. Creating a configuration file solves this problem.

The `.my.cnf` File

You can avoid typing startup parameters by adding a configuration file called `.my.conf` to your home directory.

Using your favourite text editor, create a file named `.my.cnf` with the following contents:

```
#This section applies to all clients
[client]
user=your_name
password=your_password
```

You can now start the client programme by simply typing `mysql` at the command line. Depending upon your firewall settings, you should also be able to connect to the MariaDB server running on Ubuntu 10.10 from elsewhere on your local area network as long as the client programme, **mysql**, is installed on the remote system. To connect, go to the command line on your remote computer and enter **mysql --host *ubuntu1010_host* --user *username* --password**. You can use either the IP address or the host name for the `host` option. Enter your password when prompted.

About the Author

Peter Lavin is a technical writer who has been published in a number of print and online magazines. He is also the author of Object Oriented PHP, published by No Starch Press.

Please do not reproduce this article in whole or part, in any form, without obtaining written permission.