How To Install and Secure Memcached on Ubuntu 16.04

Posted March 6, 2018 46k SECURITY FIREWALL CACHING SERVER OPTIMIZATION

7

By: Kathleen Juell

Not using **Ubuntu 16.04**? Choose a different version:

CentOS 7

Automated: Docker request

Automated: Bash request

Introduction

Memory object caching systems like Memcached can optimize backend database performance by temporarily storing information in memory, retaining frequently or recently requested records. In this way, they reduce the number of direct requests to your databases.

Because systems like Memcached can contribute to denial of service attacks if improperly configured, it is important to secure your Memcached servers. In this guide, we will cover how to protect your Memcached server by binding your installation to a local or private network interface and creating an authorized user for your Memcached instance.

Prerequisites

This tutorial assumes that you have a server set up with a non-root sudo user and a basic firewall. If that is not the case, set up the following:

- One Ubuntu 16.04 server, set up following our Initial Server Setup with Ubuntu 16.04 tutorial.

With these prerequisites in place, you will be ready to install and secure your Memcached server.

Step 1 — Installing Memcached from the Official Repositories

If you don't already have Memcached installed on your server, you can install it from the official Ubuntu repositories. First, make sure that your local package index is updated:

```
$ sudo apt-get update
```

Next, install the official package as follows:

```
$ sudo apt-get install memcached
```

We can also install `libmemcached-tools`, a library that provides several tools to work with your Memcached server:

```
$ sudo apt-get install libmemcached-tools
```

Memcached should now be installed as a service on your server, along with tools that will allow you to test its connectivity. We can now move on to securing its configuration settings.

Step 2 — Securing Memcached Configuration Settings

To ensure that our Memcached instance is listening on the local interface `127.0.0.1`, we will check the default setting in the configuration file located at `/etc/memcached.conf`. The current version of Memcached that ships with Ubuntu and Debian has the `-l` parameter set to the local interface, which prevents denial of service attacks from the network. We can inspect this setting to ensure that it is set correctly.

You can open `/etc/memcached.conf` with `nano`:

```
$ sudo nano /etc/memcached.conf
```

To inspect the interface setting, find the following line in the file:

/etc/memcached.conf

```
. . .
-l 127.0.0.1
```

If you see the default setting of `-l 127.0.0.1` then there is no need to modify this line. If you do modify this setting to be more open, then it is also a good idea to disable UDP, as it is more likely to be exploited in denial of service attacks. To disable UDP (while leaving TCP unaffected), add the following option to the bottom of this file:

/etc/memcached.conf

```
. . .
-U 0
```

Save and close the file when you are done.

Restart your Memcached service to apply your changes:

```
$ sudo systemctl restart memcached
```

Verify that Memcached is currently bound to the local interface and listening only for TCP connections by typing:

```
$ sudo netstat -plunt
```

You should see the following output:

```
Output
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
. . .
tcp        0      0 127.0.0.1:11211         0.0.0.0:*               LISTEN      2383/memcached
. . .
```

This confirms that `memcached` is bound to the `127.0.0.1` address using only TCP.

Step 3 — Adding Authorized Users

To add authenticated users to your Memcached service, it is possible to use Simple Authentication and Security Layer (SASL), a framework that de-couples authentication procedures from application protocols. We will enable SASL within our Memcached configuration file and then move on to adding a user with authentication credentials.

Configuring SASL Support

We can first test the connectivity of our Memcached instance with the `memcstat` command. This will help us establish that SASL and user authentication are enabled after we make changes to our configuration files.

To check that Memcached is up and running, type the following:

```
$ memcstat --servers="127.0.0.1"
```

You should see output like the following:

```
Output
Server: 127.0.0.1 (11211)
     pid: 3831
     uptime: 9
     time: 1520028517
     version: 1.4.25
     . . .
```

Now we can move on to enabling SASL. First, we will add the `-S` parameter to `/etc/memcached.conf`. Open the file again:

```
$ sudo nano /etc/memcached.conf
```

At the bottom of the file, add the following:

/etc/memcached.conf

```
. . .
-S
```

Next, find and uncomment the `-vv` option, which will provide verbose output to `/var/log/memcached`. The uncommented line should look like this:

/etc/memcached.conf

```
. . .
-vv
```

Save and close the file.

Restart the Memcached service:

```
$ sudo systemctl restart memcached
```

Next, we can take a look at the logs to be sure that SASL support has been enabled:

```
$ sudo journalctl -u memcached
```

You should see the following line, indicating that SASL support has been initialized:

```
Output
. . .
Mar 02 22:03:58 memcached systemd-memcached-wrapper[2760]: Initialized SASL.
. . .
```

We can check the connectivity again, but because SASL has been initialized, this command should fail without authentication:

```
$ memcstat --servers="127.0.0.1"
```

This command should not produce output. We can type the following to check its status:

```
$ echo $?
```

`$?` will always return the exit code of the last command that exited. Typically, anything besides `0` indicates process failure. In this case, we should see an exit status of `1`, which tells us that the `memcstat` command failed.

Adding an Authenticated User

Now we can download `sasl2-bin`, a package that contains administrative programs for the SASL user database. This will allow us to create our authenticated user:

```
$ sudo apt-get install sasl2-bin
```

Next, we will create the directory and file that Memcached will check for its SASL configuration settings:

```
$ sudo mkdir -p /etc/sasl2
$ sudo nano /etc/sasl2/memcached.conf
```

Add the following to the SASL configuration file:

/etc/sasl2/memcached.conf

```
mech_list: plain
log_level: 5
sasldb_path: /etc/sasl2/memcached-sasldb2
```

In addition to specifying our logging level, we will set `mech_list` to `plain`, which tells Memcached that it should use its own password file and verify a plaintext password. We will also specify the path to the user database file that we will create next. Save and close the file when you are finished.

Now we will create a SASL database with our user credentials. We will use the `saslpasswd2` command to make a new entry for our user in our database using the `-c` option. Our user will be **sammy** here, but you can replace this name with your own user. Using the `-f` option, we will specify the path to our database, which will be the path we set in `/etc/sasl2/memcached.conf`:

```
$ sudo saslpasswd2 -a memcached -c -f /etc/sasl2/memcached-sasldb2 sammy
```

Finally, we will give the `memcache` user ownership over the SASL database:

```
$ sudo chown memcache:memcache /etc/sasl2/memcached-sasldb2
```

Restart the Memcached service:

```
$ sudo systemctl restart memcached
```

Running `memcstat` again will confirm whether or not our authentication process worked. This time we will run it with our authentication credentials:

```
$ memcstat --servers="127.0.0.1" --username=sammy --password=your_password
```

You should see output like the following:

```
Output
Server: 127.0.0.1 (11211)
     pid: 3831
     uptime: 9
     time: 1520028517
     version: 1.4.25
     . . .
```

Our Memcached service is now successfully running with SASL support and user authentication.

Step 4 — Allowing Access Over the Private Network (Optional)

We have covered how to configure Memcached to listen on the local interface, which can prevent denial of service attacks by protecting the Memcached interface from exposure to outside parties. There may be instances where you will need to allow access from other servers, however. In this case, you can adjust your configuration settings to bind Memcached to the private network interface.

> **Note:** We will cover how to configure firewall settings using **UFW** in this section, but it is also possible to use DigitalOcean Cloud Firewalls to create these settings. For more information on setting up DigitalOcean Cloud Firewalls, see our Introduction to DigitalOcean Cloud Firewalls. To learn more about how to limit incoming traffic to particular machines, check out the section of this tutorial on applying firewall rules using tags and server names and our discussion of firewall tags.

Limiting IP Access With Firewalls

Before you adjust your configuration settings, it is a good idea to set up firewall rules to limit the machines that can connect to your Memcached server. You will need to know the **client server's private IP address** to configure your firewall rules.

If you are using the **UFW** firewall, you can limit access to your Memcached instance by typing the following:

```
$ sudo ufw allow from client_servers_private_IP/32 to any port 11211
```

You can find out more about UFW firewalls by reading our ufw essentials guide.

With these changes in place, you can adjust the Memcached service to bind to your server's private networking interface.

Binding Memcached to the Private Network Interface

Now that your firewall is in place, you can adjust the Memcached configuration to bind to your server's private networking interface instead of `127.0.0.1`.

We can open the `/etc/memcached.conf` file again by typing:

```
$ sudo nano /etc/memcached.conf
```

Inside, find the `-l 127.0.0.1` line that you checked or modified earlier, and change the address to match your server's private networking interface:

/etc/memcached.conf

```
. . .
-l memcached_servers_private_IP
. . .
```

Save and close the file when you are finished.

Next, restart the Memcached service:

```
$ sudo systemctl restart memcached
```

Check your new settings with `netstat` to confirm the change:

```
$ sudo netstat -plunt
```

```
Output
```

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address                        Foreign Address         State       PI
. . .
tcp        0      0 memcached_servers_private_IP:11211   0.0.0.0:*               LISTEN      23
. . .
```

Test connectivity from your external client to ensure that you can still reach the service. It is a good idea to also check access from a non-authorized client to ensure that your firewall rules are effective.

Conclusion

In this tutorial we have covered how to secure your Memcached server by configuring it to bind to your local or private network interface, and by enabling SASL authentication.

To learn more about Memcached, check out the project documentation. For more information about how to work with Memcached, see our tutorial on How To Install and Use Memcache on Ubuntu 14.04.

By: Kathleen Juell

Upvote (7) Subscribe Share

Related Tutorials

How To Install and Secure Memcached on Ubuntu 18.04

How To Secure a Containerized Node.js Application with Nginx, Let's Encrypt, and Docker Compose

How to Set Up an Nginx Ingress with Cert-Manager on DigitalOcean Kubernetes

How To Secure Nginx with NAXSI on Ubuntu 16.04

How To Set Up an OpenVPN Server on Debian 9

3 Comments

Leave a comment...

Log In to Comment

angelpokeronline *March 11, 2018*

0 cant restart memcached service, but thanks for project documentation!

agen bola
jasa seo

StanislavKo *August 18, 2018*

0 Thanks
But it's impossible to connect to memcache outside DigitalOcean.
I commented -l with #-l. Then restart.
I can telnet xxx.xxx.xxx.xxx 11211 from localhost and my other droplet.
But i can't telnet from my home computer.

---

tazotodua *November 27, 2018*

0 while installing, in journalctl -xe, i see FAILED SASL INITIALIZATION

with php, this tutorial helped me: http://www.servermom.org/install-use-memcached-nginx-php-7-ubuntu-16-04/3670/