

How to Install Memcached on Debian 12

Learn how to install Memcached on Debian 12 with our step-by-step guide. Improve your application performance with this powerful caching system.

Contents

| | | |
|----|----------------------|----|
| 01 | Introduction | 3 |
| 02 | Prerequisites | 3 |
| 03 | Install Memcached | 3 |
| 04 | Configure Memcached | 5 |
| 05 | Secure Memcached | 6 |
| 06 | Connect to Memcached | 9 |
| 07 | Conclusion | 11 |

Introduction

Memcached is a high-performance, distributed memory caching system designed to speed up dynamic web application requests by caching frequent operations such as database calls. It stores data in memory for quick retrieval, making it an essential tool for enhancing the performance and responsiveness of applications that continuously run database queries.

This article explains how to install Memcached on a Debian 12 server.

Prerequisites

Before you begin:

- Deploy [Debian 12 server](#) instance on Vultr.
- Access the server using SSH.
- Create a non-root account with sudo privileges and switch to the account.

Install Memcached

The latest Memcached version is available in the Debian 12 default repositories. Follow the steps below to install Memcached using the default APT package manager on your server.

1. Install Memcached and the required `libmemcached-tools` add-ons package.

CONSOLE

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

2. View the installed Memcached version on your server.

CONSOLE

```
$ memcached --version
```

Output.

```
memcached 1.6.18
```

3. Enable the Memcached service to automatically start at system boot.

CONSOLE

```
$ sudo systemctl enable memcached
```

4. Start the Memcached service.

CONSOLE

```
$ sudo systemctl start memcached
```

5. View the Memcached status and verify that it's running.

CONSOLE

```
$ sudo systemctl status memcached
```

Output:

```
● memcached.service - memcached daemon
   Loaded: loaded (/lib/systemd/system/memcached.service; enabled; preset:
   enabled)
   Active: active (running) since Sun 2024-07-21 21:03:31 UTC; 41s ago
     Docs: man:memcached(1)
  Main PID: 1537 (memcached)
    Tasks: 10 (limit: 2299)
   Memory: 3.8M
      CPU: 37ms
   CGroup: /system.slice/memcached.service
           └─1537 /usr/bin/memcached -m 64 -p 11211 -u memcache -l 127.0.0.1 -
           P /var/run/memcached/memcached.pid
```

Configure Memcached

The main Memcached configuration file `/etc/memcached.conf` contains all the necessary rules that manage the runtime and performance of the application on your server. Modify the file to set up the Memcached listening address, port, connection limits, and memory limits. Follow the steps below to modify the file and configure Memcached.

1. Open the Memcached configuration file.

CONSOLE

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

2. Modify the following Memcached options to match your needs.

INI

```
-v  
-l 127.0.0.1  
-m 128  
-p 11211  
-c 1024
```

Save and close the file.

Within the above configuration:

- `-v`: Sets the Memcached log verbosity level. When set to `-v`, detailed information about all process activities is added to the Memcached logs.
- `-l 127.0.0.1`: Sets the server IP address Memcached should use to listen for incoming connections. The loopback address `127.0.0.1` enables Memcached to only accept localhost connections.
- `-m 128`: Sets the maximum amount of memory to dedicate to the Memcached processes.

- `-p 11211`: Sets the TCP port Memcached should use to run and listen for incoming connections on the server.
- `-c 1024`: Sets the maximum number of simultaneous connections Memcached should process on the server.

3. Restart Memcached to apply the configuration changes.

CONSOLE

```
$ sudo systemctl restart memcached
```

4. Use the `ss` utility to verify that Memcached is actively running and listening for connection requests on your server.

CONSOLE

```
$ sudo ss -plunt | grep memcached
```

Output:

```
tcp    LISTEN 0      1024          127.0.0.1:11211
0.0.0.0:*    users: (("memcached",pid=1635,fd=22))
```

Secure Memcached

Memcached uses insecure security configurations by default. Any user with access to the Memcached server can use the application without any authentication. Follow the steps below to enable the Memcached SASL authentication package to require user passwords and secure the server.

1. Install the SASL package on your server.

CONSOLE

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

2. Create a new directory to store your SASL authentication files.

```
CONSOLE
```

```
$ sudo mkdir /etc/sasl2
```

3. Open the Memcached configuration file.

```
CONSOLE
```

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

4. Add the following directive at the end of the file to enable SASL authentication.

```
INI
```

```
-S
```

Save and close the file.

5. Create a new SASL configuration file to use with Memcached.

```
CONSOLE
```

```
$ sudo nano /etc/sasl2/memcached.conf
```

6. Add the following contents to the file.

```
INI
```

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

Save and close the file.

The above configuration enables Memcached authentication using the SASL database file. Within the configuration:

- `log_level`: Sets the SASL logging verbosity level for debugging purposes.
- `mech_list`: Sets the Memcached authentication mechanism. The value `plain` mechanism sends the username and password data in a base64-encoded format.
- `sasldb_path`: Specifies the `sasldb` database path that contains the user authentication credentials.

7. Create a new SASL user using the `saslpasswd2` utility. Replace `example-user` with your desired username.

CONSOLE

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

Enter a new strong user password when prompted.

8. Grant the Memcached user `memcache` ownership privileges to the SASL database file `/etc/sasl2/sasldb2`.

CONSOLE

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

9. Restart Memcached to apply the configuration changes.

CONSOLE

```
$ sudo systemctl restart memcached
```

10. Connect to the Memcached server using your username and password to test SASL authentication.

CONSOLE

```
$ memcstat --servers="127.0.0.1" --username=example-user --password=your-secure-password
```

Your output should be similar to the one below when successful.

```
Server: 127.0.0.1 (11211)
  pid: 1590
  uptime: 32
  time: 1722261233
  version: 1.6.18
  libevent: 2.1.12-stable
  pointer_size: 64
  rusage_user: 0.011623
  rusage_system: 0.011623
  max_connections: 1024
  curr_connections: 1
  total_connections: 2
```

Connect to Memcached

Memcached is compatible with multiple client libraries for programming languages such as PHP, Perl, and Python to cache frequently requested application data. Follow the steps below to connect to Memcached on your server using Python.

1. Install Python and the Memcached module.

CONSOLE

```
$ sudo apt install python3 python3-pylibmc -y
```

2. Create a new sample Python application `test_memcached.py` to connect to Memcached.

CONSOLE

```
$ nano test_memcached.py
```

3. Add the following contents to the file.

```
PYTHON

import pylibmc

def test_memcached_connectivity(server_address, server_port,
username, password):
    # Connecting to the Memcached server with SASL
    authentication
    client = pylibmc.Client(
        [f"{server_address}:{server_port}"],
        binary=True,
        username=username,
        password=password,
        behaviors={"tcp_nodelay": True, "ketama": True}
    )

    # Message to store in Memcached
    message_key = "greeting"
    message_value = "Connected to Memcached successfully!
Greetings from Vultr!!!"

    try:
        # Set the message in Memcached
        client.set(message_key, message_value)

        # Retrieve the message from Memcached
        retrieved_value = client.get(message_key)

        if retrieved_value:

print(f"Connection successful! Retrieved message:
{retrieved_value}")
        else:

print("Failed to retrieve the message. Connection might not
be successful.")
        except pylibmc.Error as e:
            print(f"An error occurred: {e}")
        finally:
            # Close the connection
            client.disconnect_all()

if __name__ == "__main__":
    # Replace with your server address, port, username, and
```

```
password
server_address = "127.0.0.1"
server_port = 11211
username = "example-user"
password = "your-secure-password"

test_memcached_connectivity(server_address, server_port,
username, password)
```

Save and close the file.

The above application code imports the Python Memcached module `pylibmc` library and sets a new key `greeting` with the value `Connected to Memcached successfully! Greetings from Vultr!!!` in the Memcached database. The application outputs the key value when the connection is successful.

4. Run the Python application to test the Memcached connection.

CONSOLE

```
$ python3 test_memcached.py
```

Output:

```
Connection successful! Retrieved message: Connected to Memcached successfully!
Greetings from Vultr!!!
```

Conclusion

You have installed Memcached on a Debian 12 server and secured the application with SASL authentication to validate all users. You can integrate Memcached with your existing web applications to enable caching, improve the application's performance, and enable fast data retrieval on your server. For more information, visit the [Memcached documentation](#).



VULTR

