

How to Install Redis® on Ubuntu 22.04

Learn how to install Redis on Ubuntu 22.04 with step-by-step instructions. This guide covers installation methods, configuration, and basic Redis commands.

Contents

| | | |
|----|--------------------------|---|
| 01 | Introduction | 3 |
| 02 | Prerequisites | 3 |
| 03 | Install Redis® | 3 |
| 04 | Configure Redis® | 4 |
| 05 | Secure Redis® | 6 |
| 06 | Access the Redis® Server | 7 |
| 07 | Conclusion | 9 |

Introduction

Redis® (Remote Dictionary Server) is an open-source, in-memory key-value store that functions as a database, cache, or message broker. It integrates with modern web applications to improve performance and reduce server load by storing frequent queries in memory.

This article explains how to install Redis® on Ubuntu 22.04 and access the database to integrate it with other applications on your server.s

Prerequisites

Before you begin:

- Have an [Ubuntu 22.04 server](#).
- Access the server using SSH as a non-root user with sudo privileges.s

Install Redis®

Redis® is available in the default package repositories on Ubuntu 22.04. Follow the steps below to install the latest Redis® package on your server.

1. Update the server's package index.

CONSOLE

```
$ sudo apt update
```

2. Install Redis®.

CONSOLE

```
$ sudo apt install -y redis-server
```

3. View the installed Redis® version on your server.

CONSOLE

```
$ redis-server --version
```

Output.

```
Redis server v=6.0.16 sha=00000000:0 malloc=jemalloc-5.2.1 bits=64  
build=a3fdef44459b3ad6
```

Configure Redis®

Redis® listens for connections on the default localhost port `6379`. In the following steps, configure Redis® to increase the database limit and restrict connections to the `127.0.0.1` localhost address.

1. Open the main Redis® configuration file using a text editor such as `nano`.

CONSOLE

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

- Find the following `bind` directive and verify that your localhost IPV4 `127.0.0.1` and IPV6 `::1` values are available.

INI

```
bind 127.0.0.1 -::1
```

- Find the `port` directive and verify the default Redis® port or modify it to a custom TCP port available on your server.

INI

```
port 6379
```

- Find the following `daemonize` directive and verify that it's set to `yes` to enable the Redis® service on your server.

```
INI
```

```
daemonize yes
```

Save and close the file.

2. Enable the Redis® server to automatically start at boot time.

```
CONSOLE
```

```
$ sudo systemctl enable redis-server.service
```

3. Start the Redis® service.

```
CONSOLE
```

```
$ sudo systemctl start redis
```

4. View the Redis® server status and verify it's running.

```
CONSOLE
```

```
$ sudo systemctl status redis
```

```
● redis-server.service - Advanced key-value store
   Loaded: loaded (/lib/systemd/system/redis-server.service; enabled; vendor
  preset: enabled)
   Active: active (running) since Sat 2025-04-05 17:25:36 UTC; 8min ago
     Docs: http://redis.io/documentation,
           man:redis-server(1)
  Main PID: 2169 (redis-server)
    Status: "Ready to accept connections"
     Tasks: 5 (limit: 9385)
    Memory: 2.6M
           CPU: 919ms
```

```
CGroup: /system.slice/redis-server.service
└─2169 /usr/bin/redis-server 127.0.0.1:6379"
```

Secure Redis®

By default, Redis® does not require authentication, allowing unrestricted access to its databases. Follow the steps below to enable authentication and secure your Redis® server for authorized user access only.

1. Open the main Redis® configuration file.

CONSOLE

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

2. Find the following `requirepass` directive, uncomment it, and replace `foobared` with a strong password of your choice.

INI

```
requirepass strong-password
```

Save and close the file.

The above configuration secures your Redis® server by enabling password authentication. To allow multiple users with unique passwords, uncomment the `aclfile` directive.

3. Restart the Redis® server to apply your configuration changes.

CONSOLE

```
$ sudo systemctl restart redis
```

Access the Redis® Server

The Redis® server accepts connections via the `redis-cli` utility or compatible application modules. Follow the steps below to test access by writing sample data to the default database and verify your server configurations.

1. Connect to the Redis® server.

CONSOLE

```
$ redis-cli
```

2. Test access to the server without authentication.

CONSOLE

```
127.0.0.1:6379> ping
```

Verify that your request fails with the following output:

```
(error) NOAUTH Authentication required.
```

3. Log in to the Redis® server using a valid password set in your configuration. Replace `strong-password` with the actual password you set earlier.

CONSOLE

```
127.0.0.1:6379> auth strong-password
```

Output:

```
OK
```

4. Test access to the database server again.

```
CONSOLE
```

```
127.0.0.1:6379> ping
```

Output:

```
PONG
```

5. Select a database to use on your server. For example `1`.

```
CONSOLE
```

```
127.0.0.1:6379> SELECT 1
```

6. Create a new sample key `testkey` with a value such as `Greetings from Vultr!`.

```
CONSOLE
```

```
127.0.0.1:6379[1]> set testkey "Greetings from Vultr!"
```

Output:

```
OK
```

7. Query the key value from the database.

```
CONSOLE
```

```
127.0.0.1:6379[1]> get testkey
```

Output.

```
"Greetings from Vultr!"
```

8. Exit the Redis® CLI.

```
CONSOLE
```

```
127.0.0.1:6379[1]> exit
```

Conclusion

In this article, you installed Redis® on your Ubuntu 22.04 server and configured it for secure connections. You can now integrate Redis® with your web applications to serve as a database or cache, enhancing both application and server performance.



VULTR

