Get started with Redis Enterprise Software

This guide helps you install Redis Enterprise Software on a Linux host to test its capabilities.

When finished, you'll have a simple clusert with a single node:

- Step 1: Install Redis Enterprise Software
- Step 2: Set up a Redis Enterprise Software cluster
- Step 3: Create a new Redis database
- Step 4: Connect to your Redis database

> **Note: This quick start is designed for local testing only.** For production environments, the install and setup guide walks through deployment options appropriate for a production environment.

Quick start guides are also available to help you:

- Run Redis Software using a Docker container, which lets you skip the installation process
- Set up a Redis on Flash cluster to optimize memory resources
- Set up an Active-Active cluster to enable high availability
- Benchmark Redis Enterprise Software performance.

Step 1: Install Redis Enterprise Software

To install Redis Enterprise Software:

1. Download the installation files from the Redis Enterprise Download Center and copy the download package to machine with a Linux-based OS. To untar the image:

    ```
    tar vxf <downloaded tar file name>
    ```

    > **Note:** You are required to create a free login to access the download center.

2. Once the tar command completes, run the `install.sh` script in the current directory.

    ```
    sudo ./install.sh -y
    ```

Port availability

If port 53 is in use, the installation fails. This is known to happen in default Ubuntu 18.04 installations where `systemd-resolved` (DNS server) is running. To workaround this issue, change the system configuration to make this port available before installing Redis Enterprise Software.

Here's one way to do so:

1. Run: `sudo vi /etc/systemd/resolved.conf`
2. Add `DNSStubListener=no` as the last line in the file and save the file.
3. Run: `sudo mv /etc/resolv.conf /etc/resolv.conf.orig`
4. Run: `sudo ln -s /run/systemd/resolve/resolv.conf /etc/resolv.conf`
5. Run: `sudo service systemd-resolved restart`

Step 2: Set up a cluster

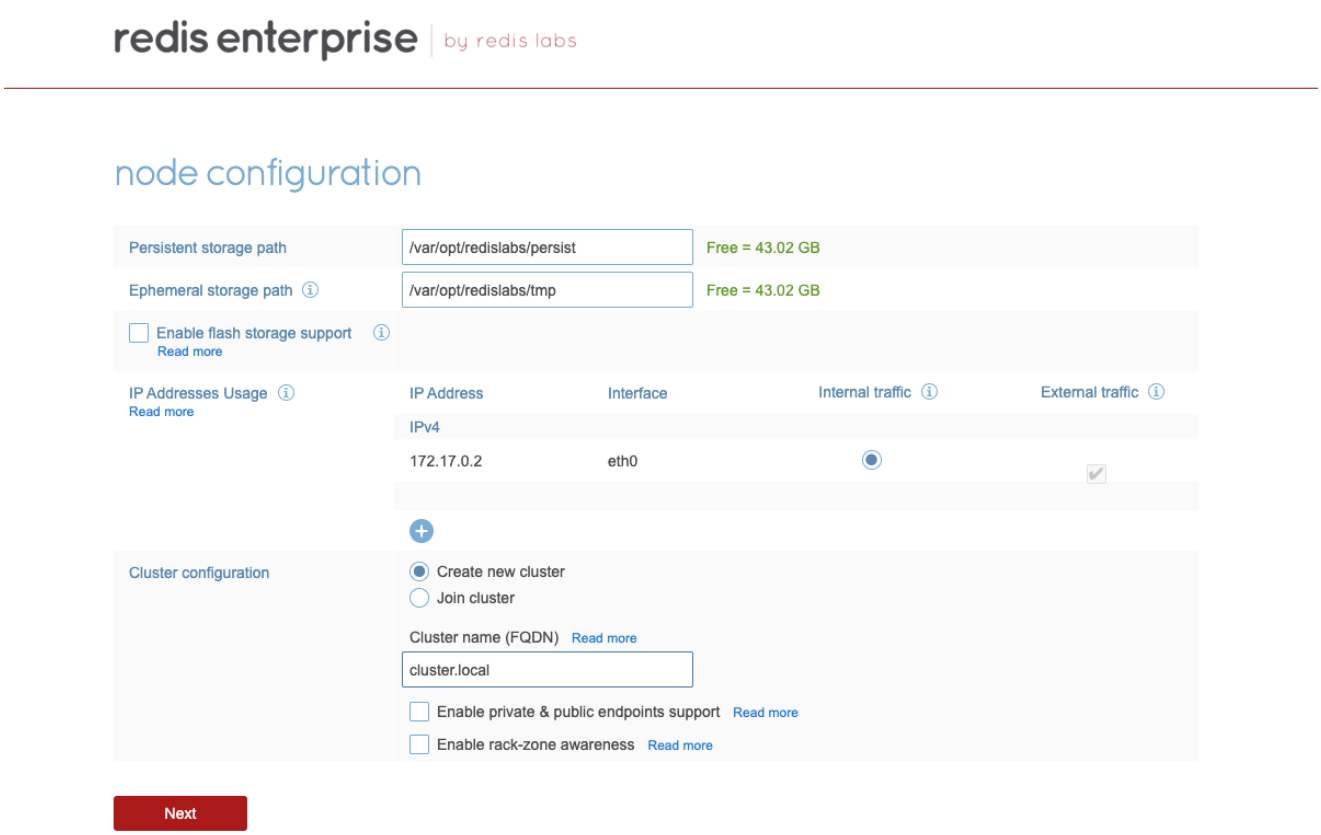
To set up your machine as a Redis Enterprise software cluster:

1. In the web browser on the host machine, go to https://localhost:8443 to see the Redis Enterprise Software web console.

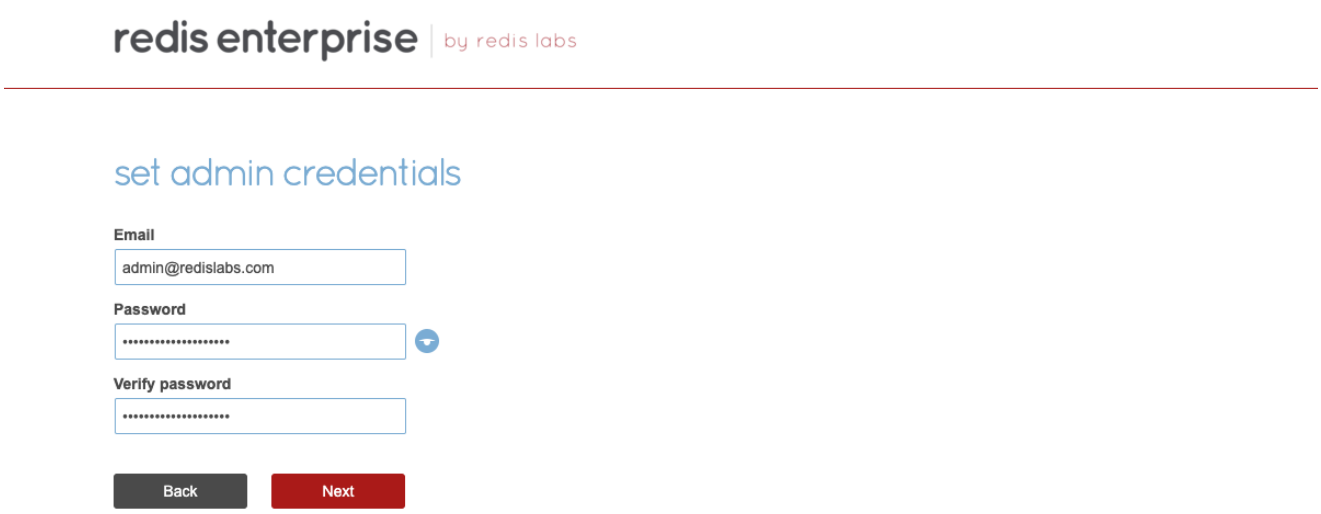
   > **Note:**
   > - Depending on your browser, you may see a certificate error. You can safely continue to the web console.
   > - If the server does not show the login screen, try again after a few minutes.

2. Click **Setup** to start the node configuration steps.

3. In the **Node Configuration** settings, enter a cluster FQDN such as `cluster.local`. Then click **Next** button.



4. Enter your license key, if you have one. If not, click the **Next** button to use the trial version.
5. Enter an email and password for the admin account for the web console.

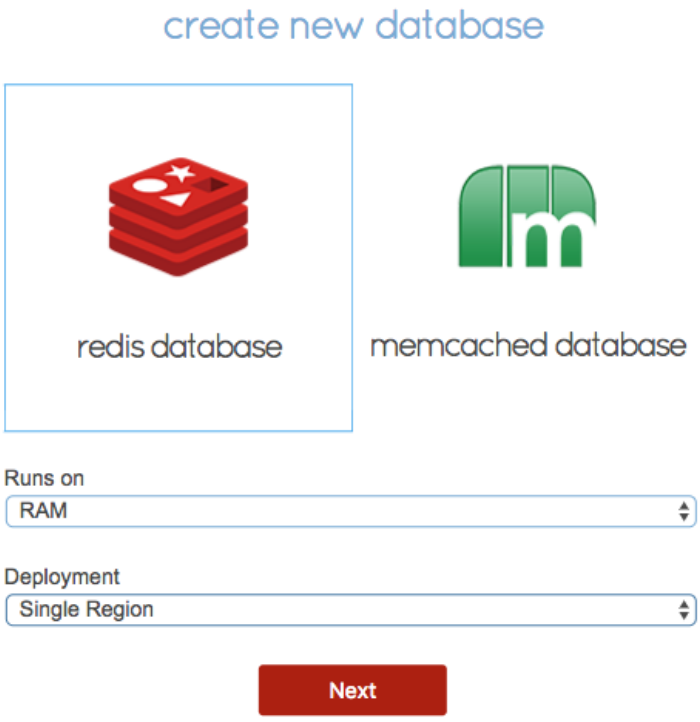


These credentials are also used for connections to the REST API.

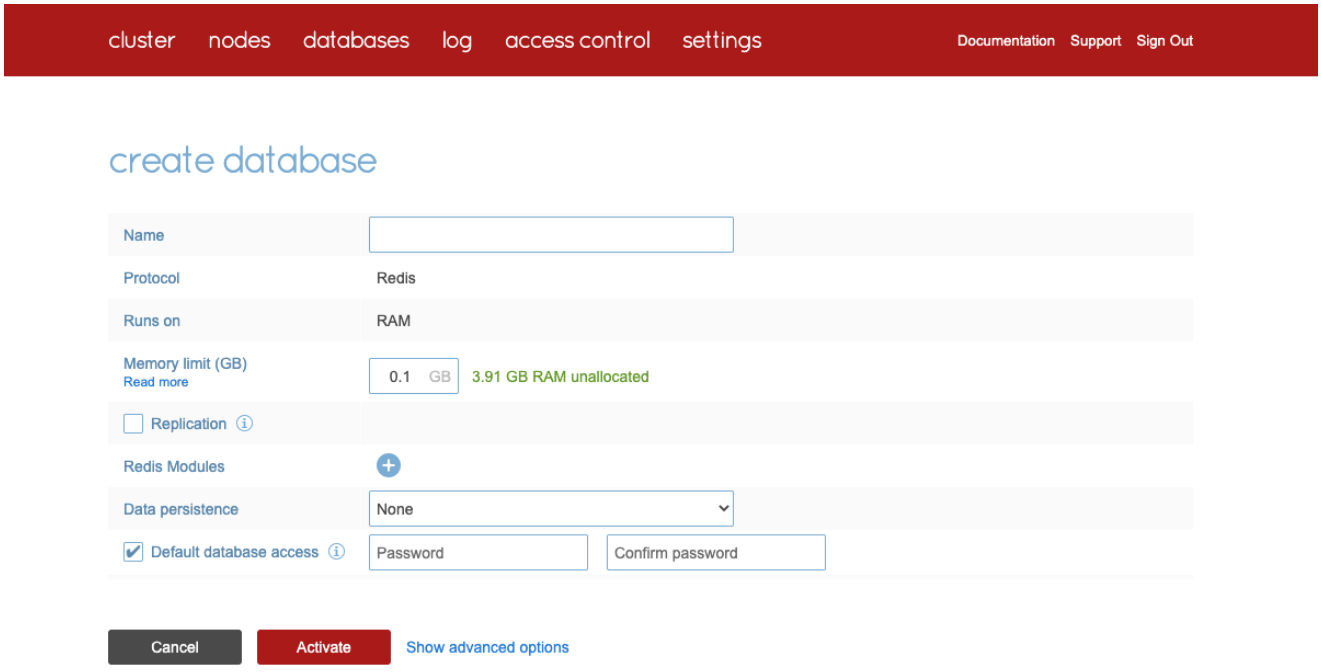
6. Click **OK** to confirm that you are aware of the replacement of the HTTPS SSL/TLS certificate on the node, and proceed through the browser warning.

Step 3: Create a database

1. Select “redis database” and the “single region” deployment, and click Next.



2. Enter a database name such as `database1` and then select **Show Advanced Options**.



3. In **Endpoint port number**, enter `12000`.
4. Select the **Activate** button to create your database.

You now have a Redis database!

Step 4: Connect to your database

After you create the Redis database, you are ready to store data in your database. You can test connectivity to your database with:

- redis-cli - the built-in command-line tool
- A *Hello World* application using Python

Connect using redis-cli

`redis-cli` is a simple command-line tool to interact with Redis database.

Here's how to run `redis-cli` to connect to port 12000 and perform basic database operations using Redis commands:

```
$ sudo /opt/redislabs/bin/redis-cli -p 12000
127.0.0.1:16653> set key1 123
OK
127.0.0.1:16653> get key1
"123"
```

Connecting using *Hello World* application in Python

A simple python application running on the **host machine**, not the container, can also connect to database1.

> **Note:** The following section assumes you already have Python and redis-py (python library for connecting to Redis) configured on the host machine running the container. You can find the instructions to configure redis-py on the github page for redis-py.

1. Create a new file called `redis_test.py` with this contents:

```
import redis

r = redis.StrictRedis(host='localhost', port=12000, db=0)
print ("set key1 123")
print (r.set('key1', '123'))
print ("get key1")
print(r.get('key1'))
```

2. Run the redis_test.py application to store and retrieve a key:

```
python.exe redis_test.py
```

If the connection is successful, the output of the application looks like this:

```
set key1 123
True
get key1
123
```

Supported web browsers

To use the Redis Software admin console, you need a modern browser with JavaScript enabled.

The following browsers have been tested with the current version of the admin console:

- Microsoft Windows, version 10 or later.
  - Google Chrome, version 48 and later
  - Microsoft Edge, version 20 and later
  - Mozilla Firefox, version 44 and and later

    - Opera, version 35 and later.

- Apple macOS:

    - Google Chrome, version 48 and later
    - Mozilla Firefox, version 44 and and later
    - Opera, version 35 and later.

- Linux:

    - Google Chrome, version 49 and later
    - Mozilla Firefox, version 44 and and later
    - Opera, version 35 and later.

Next steps

Now you have a Redis Enterprise cluster ready to go. You can connect to it with a redis client to start loading it with data or you can use the memtier_benchmark Quick Start to check the cluster performance.