Docker quickstart for Redis Enterprise Software

> **Warning -** Docker containers are currently only supported for development and test environments, not for production.

For testing purposes, you can run Redis Enterprise Software on Docker containers on Linux, Windows, or MacOS. The Redis Enterprise Software container acts as a node in a cluster.

To get started with a single Redis Enterprise Software container:

1. Install Docker for your operating system
2. Run the Redis Enterprise Software Docker container
3. Set up a cluster
4. Create a new database
5. Connect to your database

Deployment topologies

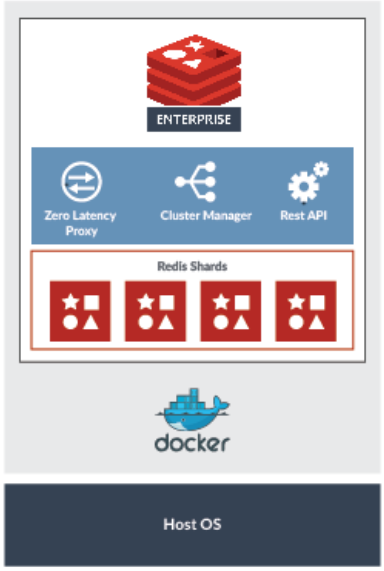
When deploying Redis Enterprise Software using Docker, several common topologies are available, according to your requirements:

- Single-node cluster – For local development or functional testing
- Multi-node cluster on a single host – For a small-scale deployment that is similar to production
- Multi-node cluster with multiple hosts – For more predictable performance or high availability compared to single-host deployments

Single node

The simplest topology is to run a single-node Redis Enterprise Software cluster with a single container on a single host machine. You can use this topology for local development or functional testing.

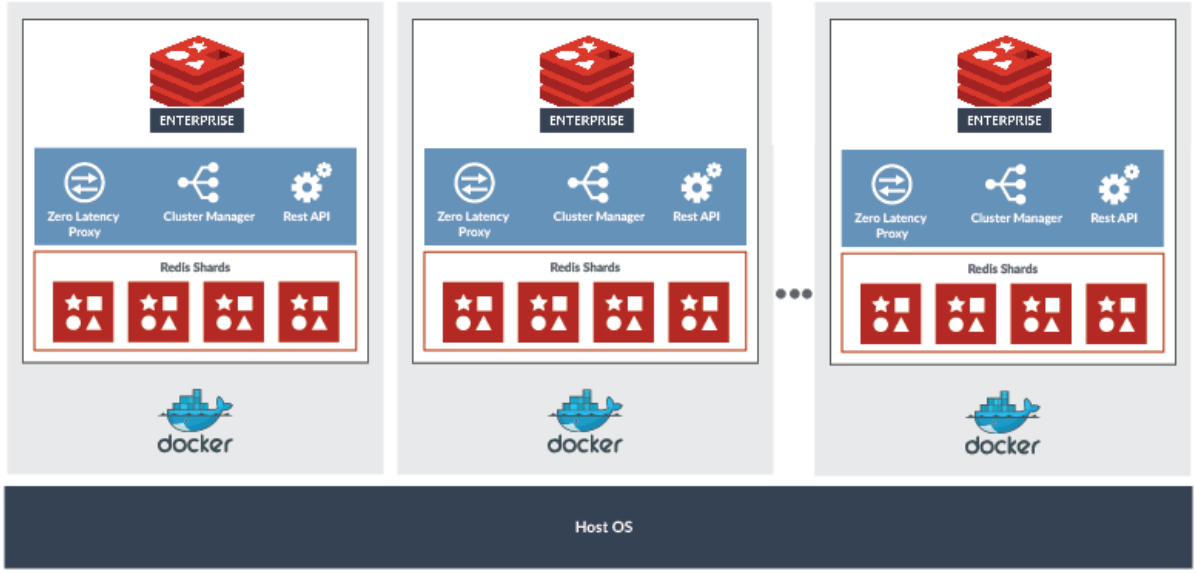
Single-node clusters have limited functionality. For example, Redis Enterprise Software can't use replication or protect against failures if the cluster has only one node.



Multiple nodes on one host

You can create a multi-node Redis Enterprise Software cluster by deploying multiple containers to a single host machine. The resulting cluster is scale minimized but similar to production deployments.

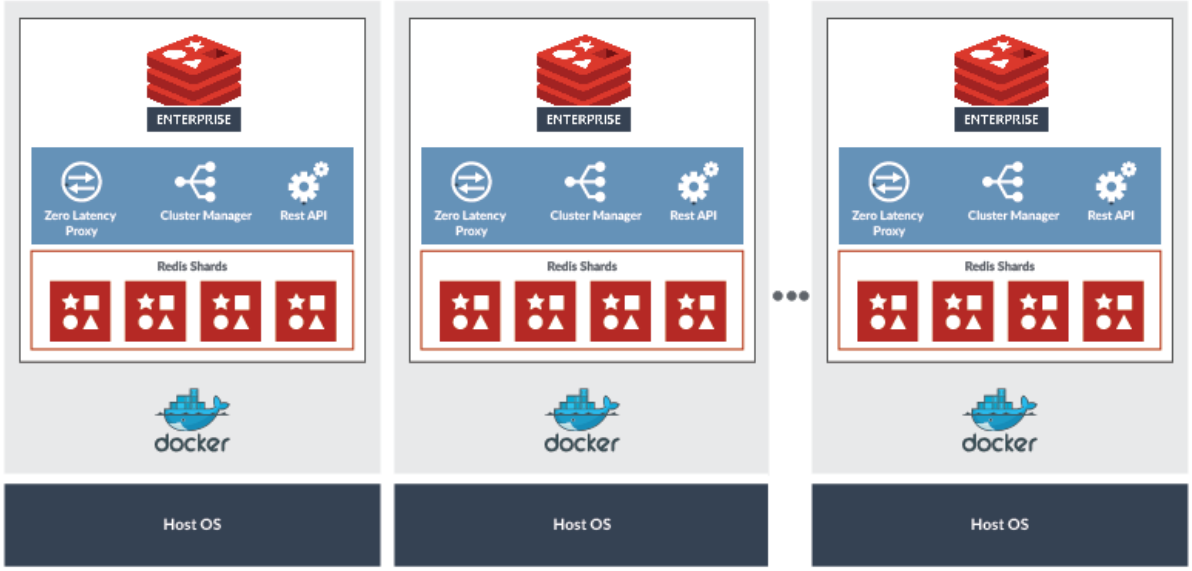
However, if you need predictable performance or high availability, don't host multiple nodes in containers on the same physical host.



Multiple nodes and hosts

You can also create a multi-node Redis Enterprise Software cluster with multiple containers by deploying each container to a different host machine.

This topology minimizes interference between containers, so Redis Enterprise Software performs more predictably than if you host multiple nodes on a single machine.



Install Docker

Follow the Docker installation instructions for your operating system:

- Linux
- MacOS
- Windows

Run the container

To download and start the Redis Enterprise Software Docker container, run the following `docker run` command in the terminal or command line for your operating system.

> **Note:** On Windows, make sure Docker is configured to run Linux-based containers.

```
docker run -d --cap-add sys_resource --name rp -p 8443:8443 -p 9443:9443 -p 12000:12000
redislabs/redis
```

The example command runs the Docker container with Redis Enterprise Software on `localhost` and opens the following ports:

- Port 8443 for HTTPS connections
- Port 9443 for REST API connections
- Port 12000 for Redis client connections

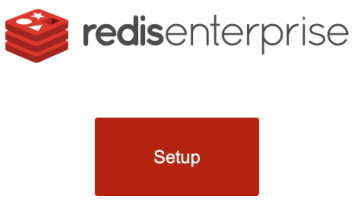
You can publish other ports with `-p <host_port>:<container_port>` or use the `--network host` option to open all ports to the host network.

Set up a cluster

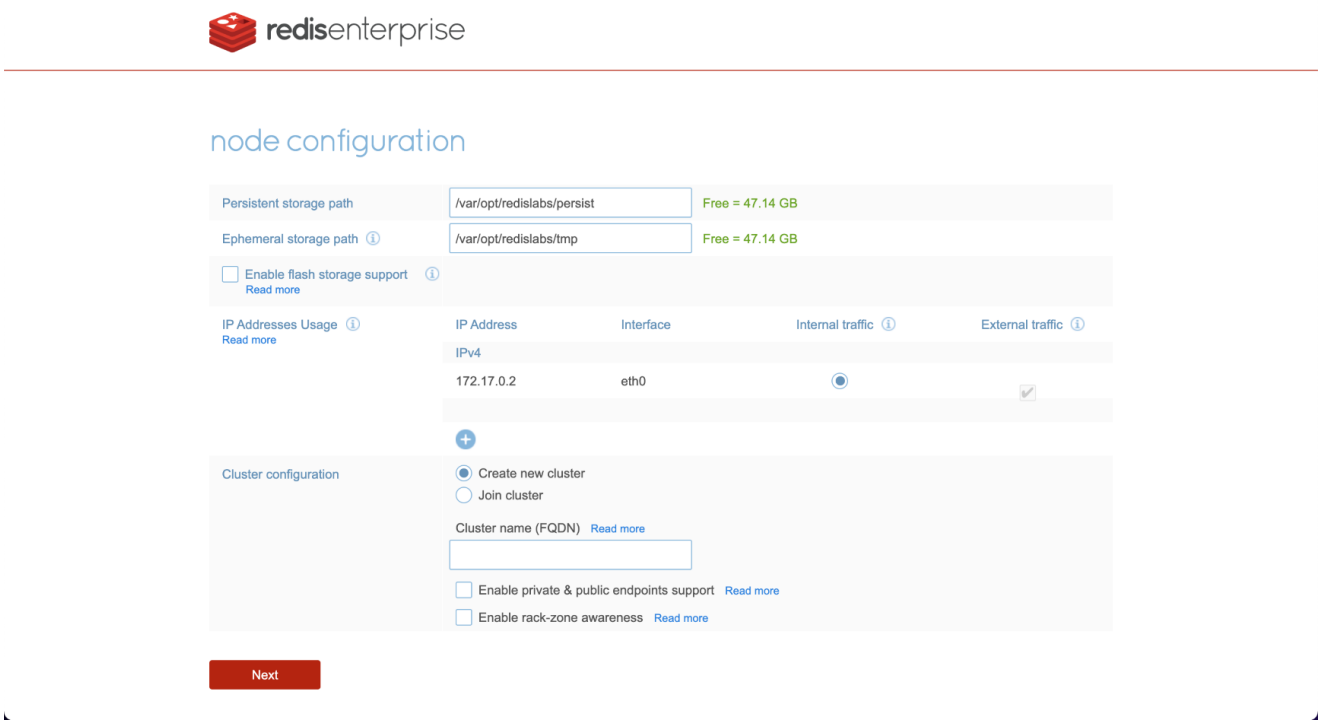
1. In the web browser on the host machine, go to https://localhost:8443 to see the Redis Enterprise Software admin console.

> **Note:**
>
> - If your browser displays a certificate error, you can safely proceed.
> - If the server does not show the login screen, try again after a few minutes.

2. Select **Setup** to start configuring the node.



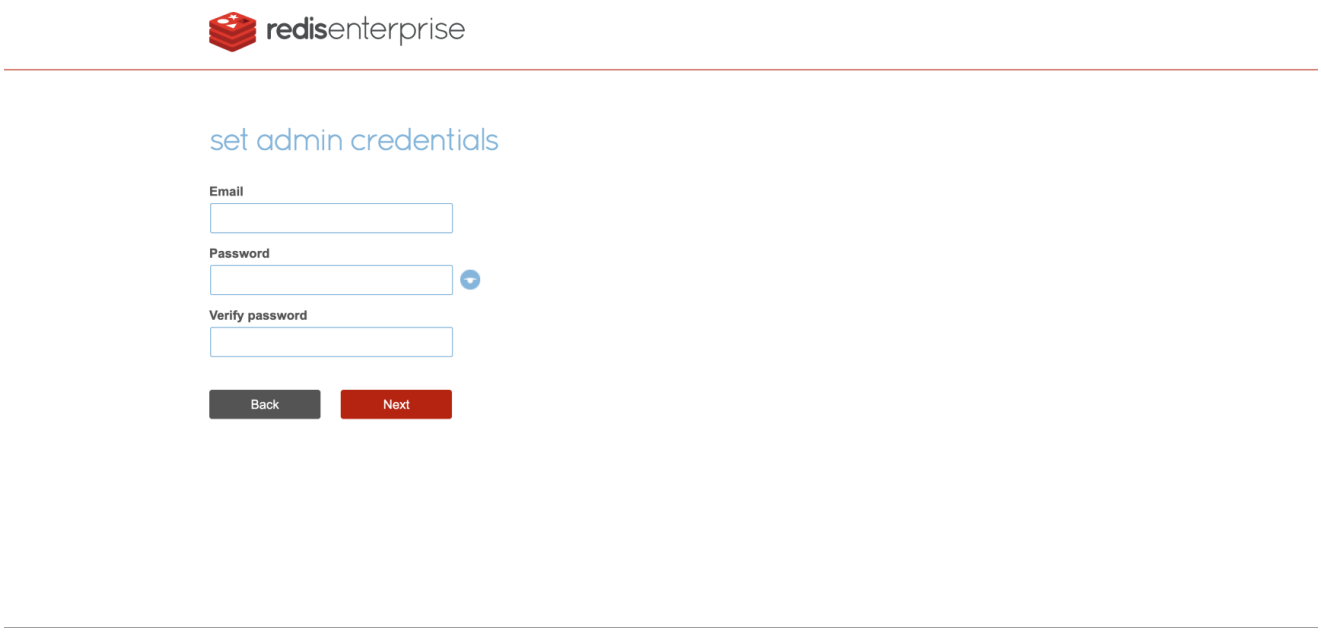
3. In the **node configuration** settings, enter a cluster FQDN such as `cluster.local` and select **Next**.



4. If you have a license key, enter it and then select **Next**.

   If you do not have a license key, a trial version is installed.

5. Enter an email and password for the administrator account.

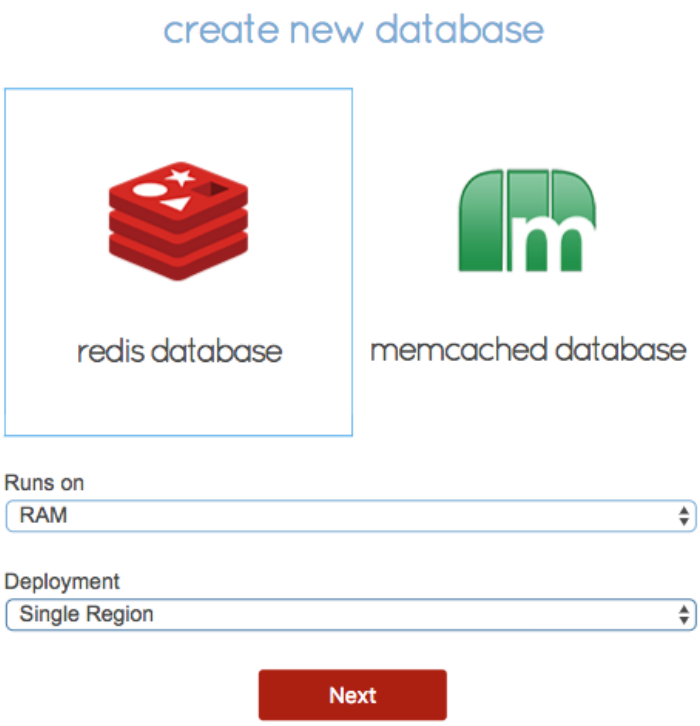


You can also use these credentials to connect to the REST API.

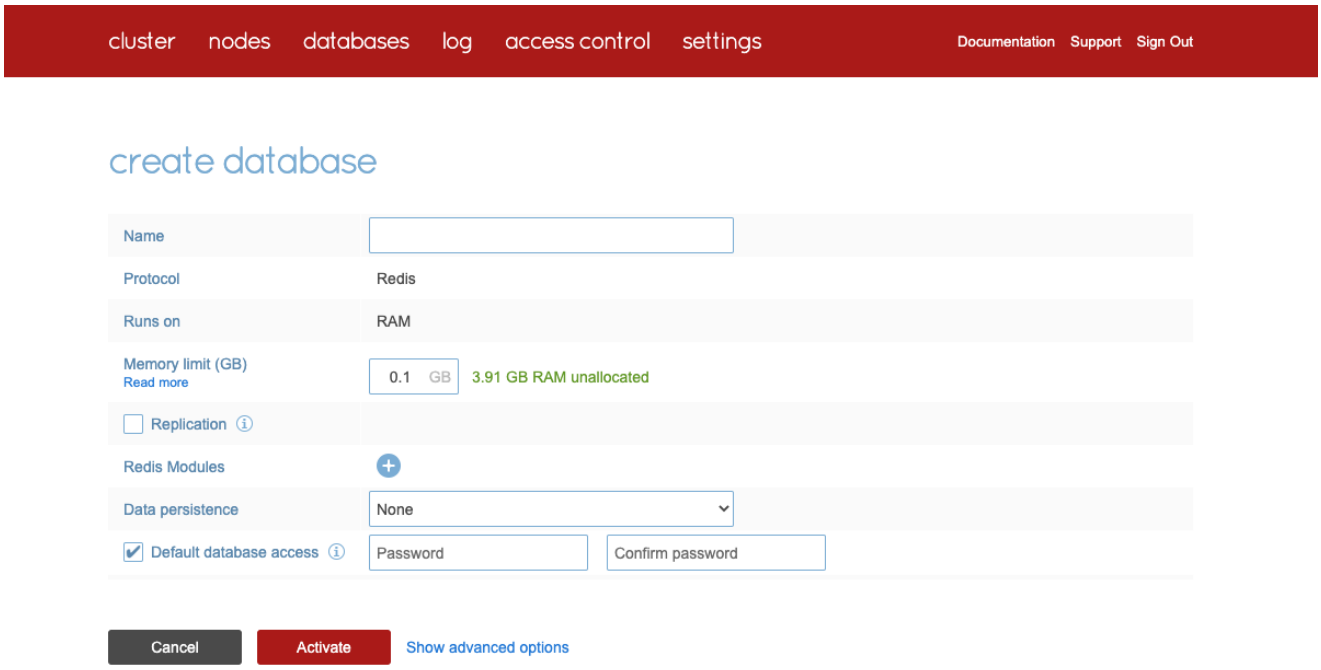
6. Select **OK** to acknowledge the replacement of the HTTPS TLS certificate on the node. If you receive a browser warning, you can proceed safely.

Create a database

1. Select **redis database** and the **Single Region** deployment, then select **Next**.



2. Enter a database name such as `database1`.



3. Select **Show advanced options** and enter `12000` for the **Endpoint port number**.

   If port 12000 is not available, enter any available port number between 10000 to 19999. You will use this port number to connect to the database.

4. Select **Activate** to create your database.

> **Note:** If you cannot activate the database because of a memory limitation, make sure that Docker has enough memory allocated in the Docker Settings.

When you see a green check mark appear on the database configuration screen, the database is activated and ready for you to use.

You now have a Redis database!

Connect to your database

After you create the Redis database, you can start storing data.

You can test connecting to your database with:

- `redis-cli`
- Python application

redis-cli

You can use the `redis-cli` command-line tool to interact with your Redis database.

1. Use `docker exec` to start an interactive shell session in the Redis Enterprise Software container:

   ```
   docker exec -it rp bash
   ```

2. Run `redis-cli` and provide the port number with `-p` to connect to the database. Then use `SET` to store a key and `GET` to retrieve it.

```
$ /opt/redislabs/bin/redis-cli -p 12000
127.0.0.1:12000> SET key1 123
OK
127.0.0.1:12000> GET key1
"123"
```

Python

You can also run a Python application on the host machine to connect to your database.

> **Note:** The following section assumes you already have Python and the Redis Python client `redis-py` set up on the host machine running the container. For `redis-py` installation instructions, see the Python client quickstart.

1. Create a new file called `redis_test.py` and add the following code:

```
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 `redis_test.py` to store a key in your database and then retrieve it:

```
$ python redis_test.py
set key1 123
True
get key1
123
```

Next steps

- Connect to your Redis database with a Redis client and start adding data.
- Use the `memtier_benchmark` quickstart to check the cluster performance.

Updated: June 13, 2023