Setup Environment

# Install mongodb on Ubuntu 16.04

## **Prerequisites**

To follow this tutorial, you will need:

* One Ubuntu 16.04 server set up by following this [initial server setup tutorial](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-16-04), including a sudo non-root user

## **Step 1 — Adding the MongoDB Repository**

MongoDB is already included in Ubuntu package repositories, but the official MongoDB repository provides most up-to-date version and is the recommended way of installing the software. In this step, we will add this official repository to our server.

Ubuntu ensures the authenticity of software packages by verifying that they are signed with GPG keys, so we first have to import they key for the official MongoDB repository.

* sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv EA312927

After successfully importing the key, you will see:

Output

gpg: Total number processed: 1

gpg: imported: 1 (RSA: 1)

Next, we have to add the MongoDB repository details so apt will know where to download the packages from.

Issue the following command to create a list file for MongoDB.

* echo "deb http://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/3.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.2.list

After adding the repository details, we need to update the packages list.

* sudo apt-get update

## **Step 2 — Installing and Verifying MongoDB**

Now we can install the MongoDB package itself.

* sudo apt-get install -y mongodb-org

This command will install several packages containing latest stable version of MongoDB along with helpful management tools for the MongoDB server.

In order to properly launch MongoDB as a service on Ubuntu 16.04, we additionally need to create a unit file describing the service. A unit file tells systemd how to manage a resource. The most common unit type is a service, which determines how to start or stop the service, when should it be automatically started at boot, and whether it is dependent on other software to run.

We'll create a unit file to manage the MongoDB service. Create a configuration file namedmongodb.service in the /etc/systemd/system directory using nano or your favorite text editor.

* sudo nano /etc/systemd/system/mongodb.service

Paste in the following contents, then save and close the file.

/etc/systemd/system/mongodb.service

[Unit]

Description=High-performance, schema-free document-oriented database

After=network.target

[Service]

User=mongodb

ExecStart=/usr/bin/mongod --quiet --config /etc/mongod.conf

[Install]

WantedBy=multi-user.target

This file has a simple structure:

* The **Unit** section contains the overview (e.g. a human-readable description for MongoDB service) as well as dependencies that must be satisfied before the service is started. In our case, MongoDB depends on networking already being available, hence network.target here.
* The **Service** section how the service should be started. The User directive specifies that the server will be run under the mongodb user, and the ExecStart directive defines the startup command for MongoDB server.
* The last section, **Install**, tells systemd when the service should be automatically started. The multi-user.target is a standard system startup sequence, which means the server will be automatically started during boot.

Next, start the newly created service with systemctl.

* sudo systemctl start mongodb

While there is no output to this command, you can also use systemctl to check that the service has started properly.

* sudo systemctl status mongodb

Output

● mongodb.service - High-performance, schema-free document-oriented database

Loaded: loaded (/etc/systemd/system/mongodb.service; enabled; vendor preset: enabled)

Active: active (running) since Mon 2016-04-25 14:57:20 EDT; 1min 30s ago

Main PID: 4093 (mongod)

Tasks: 16 (limit: 512)

Memory: 47.1M

CPU: 1.224s

CGroup: /system.slice/mongodb.service

└─4093 /usr/bin/mongod --quiet --config /etc/mongod.conf

The last step is to enable automatically starting MongoDB when the system starts.

* sudo systemctl enable mongodb

The MongoDB server now configured and running, and you can manage the MongoDB service using thesystemctl command (e.g. sudo systemctl mongodb stop, sudo systemctl mongodb start).

## **Step 3 — Adjusting the Firewall (Optional)**

Assuming you have followed the [initial server setup tutorial](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-16-04) instructions to enable the firewall on your server, MongoDB server will be inaccessible from the internet.

If you intend to use the MongoDB server only locally with applications running on the same server, it is a recommended and secure setting. However, if you would like to be able to connect to your MongoDB server from the internet, we have to allow the incoming connections in ufw.

To allow access to MongoDB on its default port 27017 from everywhere, you could use sudo ufw allow27017. However, enabling internet access to MongoDB server on a default installation gives unrestricted access to the whole database server.

in most cases, MongoDB should be accessed only from certain trusted locations, such as another server hosting an application. To accomplish this task, you can allow access on MongoDB's default port while specifying the IP address of another server that will be explicitly allowed to connect.

* sudo ufw allow from your\_other\_server\_ip/32 to any port 27017

You can verify the change in firewall settings with ufw.

* sudo ufw status

You should see traffic to 27017 port allowed in the output.If you have decided to allow only a certain IP address to connect to MongoDB server, the IP address of the allowed location will be listed instead ofAnywhere in the output.

Output

Status: active

To Action From

-- ------ ----

27017 ALLOW Anywhere

OpenSSH ALLOW Anywhere

27017 (v6) ALLOW Anywhere (v6)

OpenSSH (v6) ALLOW Anywhere (v6)

More advanced firewall settings for restricting access to services are described in [UFW Essentials: Common Firewall Rules and Commands](https://www.digitalocean.com/community/tutorials/ufw-essentials-common-firewall-rules-and-commands).

# Install Nodejs on Ubuntu 16.04

## Step 1: Add NodeJs PPA

First you need to node.js ppa in our system provide by nodejs official website. We also need to install python-software-properties package if not installed already.

$ sudo apt-get install python-software-properties

$ curl -sL https://deb.nodesource.com/setup\_6.x | sudo -E bash -

## Step 2: Install Node.js and NPM

After adding required PPA file, lets install Nodejs package. NPM will also be installed with node.js. This command will also install many other dependent packages on your system.

$ **sudo apt-get install nodejs**

## Step 3: Check Node.js and NPM Version

After installing node.js verify and check the installed version. You can find more details about current version on node.js [official website](http://nodejs.org/download/).

$ node -v

v6.5.0

Also check the version of npm.

$ npm -v

3.10.3

## Step 4: Create Demo Web Server (Optional)

This is an optional step. If you want to test your node.js install. Lets create an web server with “Hello World!” text. Create a file **http\_server.js**

$ **vim http\_server.js**

and add following content

var http = require('http');

http.createServer(function (req, res) {

res.writeHead(200, {'Content-Type': 'text/plain'});

res.end('Hello Worldn');

}).listen(3001, "127.0.0.1");

console.log('Server running at http://127.0.0.1:3001/');

Now start the web server using below command.

$ node --debug http\_server.js

debugger listening on port 5858

Server running at http://127.0.0.1:3001/

Web server has been started on port 3001. Now access **http://127.0.0.1:3001/** url in browser.