

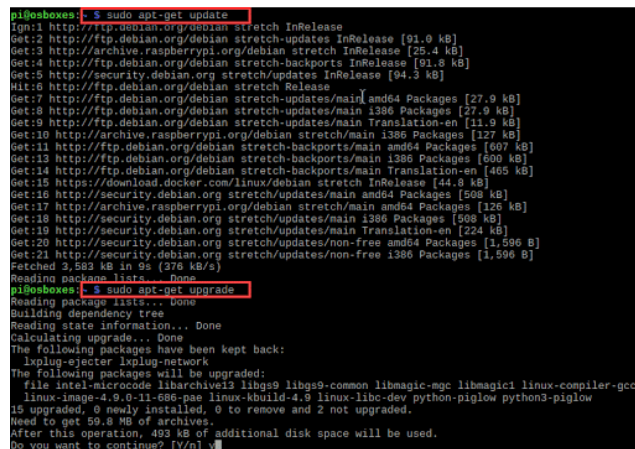
# Steps to install ThingsBoard on Raspberry PI 4

## Step 1:

Upgrade and Update

In this step, keep upgrading and updating your system and install the most advanced software version.

Run this command "**sudo apt-get update && sudo apt-get upgrade**" as shown in the below screenshot.



```
pi@osboxes:~$ sudo apt-get update
Ign:1 http://ftp.debian.org/debian stretch InRelease
Get:2 http://ftp.debian.org/debian stretch-updates InRelease [91.0 kB]
Get:3 http://archive.raspberrypi.org/debian stretch InRelease [25.4 kB]
Get:4 http://ftp.debian.org/debian stretch-backports InRelease [91.8 kB]
Get:5 http://security.debian.org stretch/updates InRelease [94.3 kB]
Hit:6 http://ftp.debian.org/debian stretch Release
Get:7 http://ftp.debian.org/debian stretch-updates/main amd64 Packages [27.9 kB]
Get:8 http://ftp.debian.org/debian stretch-updates/main i386 Packages [27.9 kB]
Get:9 http://ftp.debian.org/debian stretch-updates/main Translation-en [11.9 kB]
Get:10 http://archive.raspberrypi.org/debian stretch/main i386 Packages [127 kB]
Get:11 http://ftp.debian.org/debian stretch-backports/main amd64 Packages [607 kB]
Get:12 http://ftp.debian.org/debian stretch-backports/main i386 Packages [600 kB]
Get:13 http://ftp.debian.org/debian stretch-backports/main Translation-en [465 kB]
Get:14 http://download.docker.com/linux/debian stretch InRelease [44.8 kB]
Get:15 http://security.debian.org stretch/updates/main amd64 Packages [598 kB]
Get:16 http://archive.raspberrypi.org/debian stretch/main amd64 Packages [126 kB]
Get:17 http://security.debian.org stretch/updates/main i386 Packages [598 kB]
Get:18 http://security.debian.org stretch/updates/main Translation-en [224 kB]
Get:19 http://security.debian.org stretch/updates/non-free amd64 Packages [1,596 B]
Get:20 http://security.debian.org stretch/updates/non-free i386 Packages [1,596 B]
Get:21 http://security.debian.org stretch/updates/non-free Translation-en [1,596 B]
Fetched 3,583 kB in 9s (376 kB/s)
Reading package lists... Done
pi@osboxes:~$ sudo apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  lxplug-ejector lxplug-network
The following packages will be upgraded:
  file intel-microcode libarchive12 libgs9 libgs9-common libmagic-mgc libmagic1 linux-comiler-gcc-
  linux-image-4.9.0-11-686-pae linux-kbuild-4.9 linux-libc-dev python-piglow python3-piglow
  15 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
Need to get 59.8 MB of archives.
After this operation, 492 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

## Step 2:

Download the right script and install Docker on the Raspberry Pi environment.

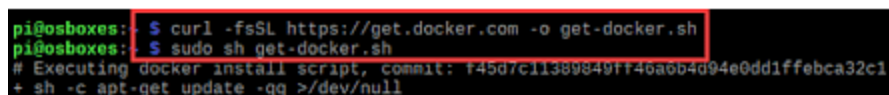
Push this below installation script.

**curl -fsSL https://get.docker.com -o get-docker.sh**

Run the script with the help of the below command:

**sudo sh get-docker.sh**

This establishes the expected packages on the Raspbian Linux administration environment.



```
pi@osboxes:~$ curl -fsSL https://get.docker.com -o get-docker.sh
pi@osboxes:~$ sudo sh get-docker.sh
# Executing docker install script, commit: f45d7c11389849ff46a6b4d94e0dd1ffebca32c1
+ sh -c apt-get update -qq >/dev/null
```

Here is the running output of the Docker version as per the above command.

```
Server: Docker Engine - Community
Engine:
 Version:      19.03.5
 API version:  1.40 (minimum version 1.12)
 Go version:   go1.12.12
 Git commit:   633a0ea838
 Built:        Wed Nov 13 07:28:22 2019
 OS/Arch:      linux/amd64
 Experimental: false
containerd:
 Version:      1.2.10
 GitCommit:    b34a5c8af56e510852c35414db4c1f4fa6172339
runc:
 Version:      1.0.0-rc8+dev
 GitCommit:    3e425f80a8c931f88e6d94a8c831b9d5aa481657
docker-init:
 Version:      0.18.0
 GitCommit:    fec3683
If you would like to use Docker as a non-root user, you should now consider
adding your user to the "docker" group with something like:

    sudo usermod -aG docker your-user

Remember that you will have to log out and back in for this to take effect!

WARNING: Adding a user to the "docker" group will grant the ability to run
containers which can be used to obtain root privileges on the
docker host.
Refer to https://docs.docker.com/engine/security/security/#docker-daemon-attac
k-surface
for more information.
```

### Step 3:

Append a non-root user on the Docker group

As per the Raspberry Pi process, whichever user has the administrative rights whom we can consider as root user can execute containers. For example, if a case user is not logged in to the admin root, then they should use the sudo prefix.

We can also add the non-root users to the Docker group, which will enable running the executed docker commands.

Here is the syntax for adding users to the Docker group:

```
sudo usermod -aG docker [user_name]
```

We can consider it the default user for adding the Pi user in Raspbian.

Refer to the below command:

```
sudo usermod -aG docker Pi
```

We need to log out of the system to check the output process.

### Step 4:

For checking the Updated Docker Version and info details on your Raspberry Pi, refer to the below command.

```
docker version
```

The below output screenshot will show the docker version with all the relevant details.

For getting the system-wide details, which involve the list of containers, kernel version, and docker images, run this below command.

```
docker info
```

## Step 5:

Execute/run the sample Hello World Container. Here we need to configure and set up this below command.

### **docker run hello-world**

After running the above command, it will contact the Docker daemon and pull the “hello-world” image.

The output will show the “installation appears to be working correctly” message.

```
pi@osboxes:~$ sudo docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

## Docker Raspberry Pi 4

Raspberry Pi 4 can work as a low-cost Docker resolution for application development and various responsibilities. It has the latest 8GB version.

A Raspberry Pi 4 is a single-board processor. To connect Docker on your Raspberry Pi 4, we need the following:

A Raspberry Pi 4 Type-C power accumulation.

A microSD card has a minimum of 32GB storage space, including the Raspberry Pi OS image on it.

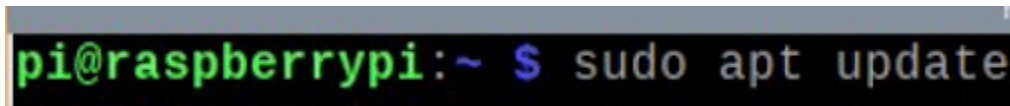
Internet connectivity using the Raspberry Pi 4.

A laptop or computer for VNC remote desktop accessibility using SSH way to the Raspberry Pi 4.

## Installing Docker Raspberry Pi 4

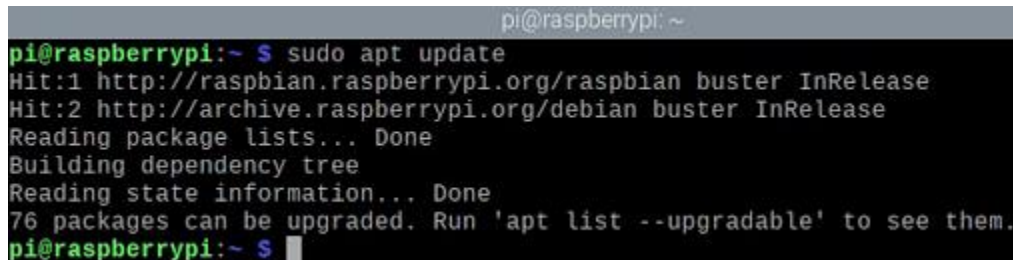
Below is the command for updating Raspberry Pi OS.

**\$ sudo apt update**



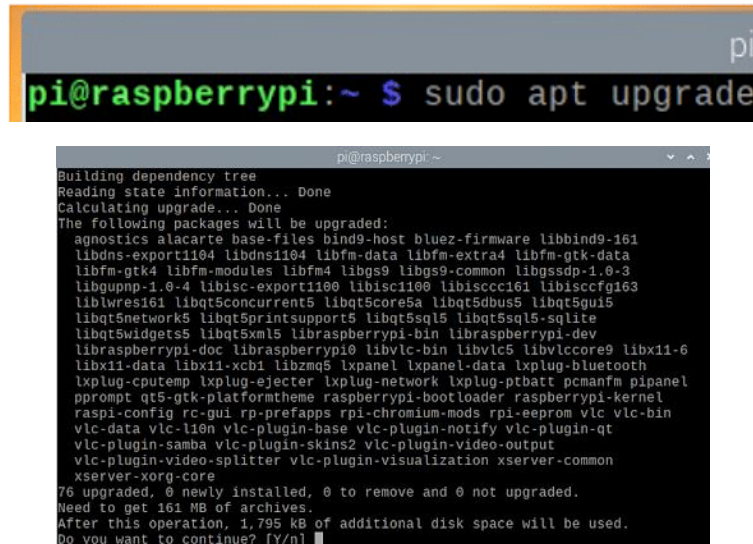
```
pi@raspberrypi:~ $ sudo apt update
```

After running the command, it will show the below output.



```
pi@raspberrypi:~  
pi@raspberrypi:~ $ sudo apt update  
Hit:1 http://raspbian.raspberrypi.org/raspbian buster InRelease  
Hit:2 http://archive.raspberrypi.org/debian buster InRelease  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
76 packages can be upgraded. Run 'apt list --upgradable' to see them.  
pi@raspberrypi:~ $
```

```
$ sudo apt upgrade
```



# Raspberry Pi 4 Docker

Raspberry Pi 4 can be applied to build a private cloud resolution with Ansible and Docker, which are great tools used by various large-scale cloud systems that automate the tasks, configure them, and enable containerization for the applications.

After booting your system, the following commands can implement the most advanced Rasbian updates.

```
$ sudo apt updat
```

```
$ sudo apt dist-upgrade
```

To define the hostname, we need to add two files as per the below command.

```
$ sudo nano /etc/hostname
```

```
$ sudo nano /etc/hosts
```