Install Docker Engine on Ubuntu

To get started with Docker Engine on Ubuntu, make sure you meet the prerequisites, and then follow the installation steps.

Prerequisites

Note

If you use ufw or firewalld to manage firewall settings, be aware that when you expose container ports using Docker, these ports bypass your firewall rules. For more information, refer to Docker and ufw.

OS requirements

To install Docker Engine, you need the 64-bit version of one of these Ubuntu versions:

- Ubuntu Lunar 23.04
- Ubuntu Kinetic 22.10
- Ubuntu Jammy 22.04 (LTS)
- Ubuntu Focal 20.04 (LTS)

Docker Engine for Ubuntu is compatible with x86_64 (or amd64), armhf, arm64, s390x, and ppc64le (ppc64el) architectures.

Uninstall old versions

Before you can install Docker Engine, you need to uninstall any conflicting packages.

Distro maintainers provide unofficial distributions of Docker packages in APT. You must uninstall these packages before you can install the official version of Docker Engine.

The unofficial packages to uninstall are:

- `docker.io`
- `docker-compose`
- `docker-compose-v2`
- `docker-doc`
- `podman-docker`

Moreover, Docker Engine depends on `containerd` and `runc`. Docker Engine bundles these dependencies as one bundle: `containerd.io`. If you have installed the `containerd` or `runc` previously, uninstall them to avoid conflicts with the versions bundled with Docker Engine.

Run the following command to uninstall all conflicting packages:

```
$ for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker containerd runc; do sudo apt-get remove $pkg; done
```

`apt-get` might report that you have none of these packages installed.
Images, containers, volumes, and networks stored in `/var/lib/docker/` aren't automatically removed when you uninstall Docker. If you want to start with a clean installation, and prefer to clean up any existing data, read the [uninstall Docker Engine]() section.

Installation methods

You can install Docker Engine in different ways, depending on your needs:

- Docker Engine comes bundled with Docker Desktop for Linux. This is the easiest and quickest way to get started.
- Set up and install Docker Engine from Docker's `apt` repository.
- Install it manually and manage upgrades manually.
- Use a convenience script. Only recommended for testing and development environments.

Install using the apt repository

Before you install Docker Engine for the first time on a new host machine, you need to set up the Docker repository. Afterward, you can install and update Docker from the repository.

1. Set up Docker's `apt` repository.

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Add the repository to Apt sources:
echo \
  "deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
12.    "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
13.    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

Note

If you use an Ubuntu derivative distro, such as Linux Mint, you may need to use `UBUNTU_CODENAME` instead of `VERSION_CODENAME`.

14. Install the Docker packages.

Latest Specific version

To install the latest version, run:

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

15. Verify that the Docker Engine installation is successful by running the `hello-world` image.

```
16.  $ sudo docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

You have now successfully installed and started Docker Engine.

Tip

Receiving errors when trying to run without root?

The `docker` user group exists but contains no users, which is why you're required to use `sudo` to run Docker commands. Continue to Linux postinstall to allow non-privileged users to run Docker commands and for other optional configuration steps.

Upgrade Docker Engine

To upgrade Docker Engine, follow step 2 of the installation instructions, choosing the new version you want to install.

Install from a package

If you can't use Docker's `apt` repository to install Docker Engine, you can download the `deb` file for your release and install it manually. You need to download a new file each time you want to upgrade Docker Engine.

1. Go to https://download.docker.com/linux/ubuntu/dists/open in new.
2. Select your Ubuntu version in the list.
3. Go to `pool/stable/` and select the applicable architecture (`amd64`, `armhf`, `arm64`, or `s390x`).
4. Download the following `deb` files for the Docker Engine, CLI, containerd, and Docker Compose packages:
   - `containerd.io_<version>_<arch>.deb`
   - `docker-ce_<version>_<arch>.deb`
   - `docker-ce-cli_<version>_<arch>.deb`
   - `docker-buildx-plugin_<version>_<arch>.deb`
   - `docker-compose-plugin_<version>_<arch>.deb`
5. Install the `.deb` packages. Update the paths in the following example to where you downloaded the Docker packages.

```
6. $ sudo dpkg -i ./containerd.io_<version>_<arch>.deb \
7.   ./docker-ce_<version>_<arch>.deb \
8.   ./docker-ce-cli_<version>_<arch>.deb \
9.   ./docker-buildx-plugin_<version>_<arch>.deb \
10.    ./docker-compose-plugin_<version>_<arch>.deb
```

The Docker daemon starts automatically.

11. Verify that the Docker Engine installation is successful by running the `hello-world` image.

```
12.  $ sudo service docker start
13.  $ sudo docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints a confirmation message and exits.

You have now successfully installed and started Docker Engine.

Tip

Receiving errors when trying to run without root?

The `docker` user group exists but contains no users, which is why you're required to use `sudo` to run Docker commands. Continue to Linux postinstall to allow non-privileged users to run Docker commands and for other optional configuration steps.

Upgrade Docker Engine

To upgrade Docker Engine, download the newer package files and repeat the installation procedure, pointing to the new files.

Install using the convenience script

Docker provides a convenience script at https://get.docker.com/open in new to install Docker into development environments non-interactively. The convenience script isn't recommended for production environments, but it's useful for creating a provisioning script tailored to your needs. Also refer to the install using the repository steps to learn about installation steps to install using the package repository. The source code for the script is open source, and you can find it in the `docker-install` repository on GitHubopen in new.

Always examine scripts downloaded from the internet before running them locally. Before installing, make yourself familiar with potential risks and limitations of the convenience script:

- The script requires `root` or `sudo` privileges to run.
- The script attempts to detect your Linux distribution and version and configure your package management system for you.
- The script doesn't allow you to customize most installation parameters.
- The script installs dependencies and recommendations without asking for confirmation. This may install a large number of packages, depending on the current configuration of your host machine.
- By default, the script installs the latest stable release of Docker, containerd, and runc. When using this script to provision a machine, this may result in unexpected major version upgrades of Docker. Always test upgrades in a test environment before deploying to your production systems.
- The script isn't designed to upgrade an existing Docker installation. When using the script to update an existing installation, dependencies may not be updated to the expected version, resulting in outdated versions.

Tip: preview script steps before running

You can run the script with the `--dry-run` option to learn what steps the script will run when invoked:

```
$ curl -fsSL https://get.docker.com -o get-docker.sh
$ sudo sh ./get-docker.sh --dry-run
```

This example downloads the script from https://get.docker.com/open in new and runs it to install the latest stable release of Docker on Linux:

```
$ curl -fsSL https://get.docker.com -o get-docker.sh
$ sudo sh get-docker.sh
Executing docker install script, commit:
7cae5f8b0decc17d6571f9f52eb840fbc13b2737
<...>
```

You have now successfully installed and started Docker Engine. The `docker` service starts automatically on Debian based distributions. On `RPM` based distributions, such as CentOS, Fedora, RHEL or SLES, you need to start it manually using the appropriate `systemctl` or `service` command. As the message indicates, non-root users can't run Docker commands by default.

Use Docker as a non-privileged user, or install in rootless mode?

The installation script requires `root` or `sudo` privileges to install and use Docker. If you want to grant non-root users access to Docker, refer to the post-installation steps for Linux. You can also install Docker without `root` privileges, or configured to run in rootless mode. For instructions on running Docker in rootless mode, refer to run the Docker daemon as a non-root user (rootless mode).

Install pre-releases

Docker also provides a convenience script at https://test.docker.com/open_in_new to install pre-releases of Docker on Linux. This script is equal to the script at `get.docker.com`, but configures your package manager to use the test channel of the Docker package repository. The test channel includes both stable and pre-releases (beta versions, release-candidates) of Docker. Use this script to get early access to new releases, and to evaluate them in a testing environment before they're released as stable.

To install the latest version of Docker on Linux from the test channel, run:

```
$ curl -fsSL https://test.docker.com -o test-docker.sh
$ sudo sh test-docker.sh
```

Upgrade Docker after using the convenience script

If you installed Docker using the convenience script, you should upgrade Docker using your package manager directly. There's no advantage to re-running the convenience script. Re-running it can cause issues if it attempts to re-install repositories which already exist on the host machine.

Uninstall Docker Engine

1. Uninstall the Docker Engine, CLI, containerd, and Docker Compose packages:
2.
```
$ sudo apt-get purge docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin docker-ce-rootless-extras
```
3. Images, containers, volumes, or custom configuration files on your host aren't automatically removed. To delete all images, containers, and volumes:
4.
```
$ sudo rm -rf /var/lib/docker
```

```
5. $ sudo rm -rf /var/lib/containerd
```

You have to delete any edited configuration files manually.

Next steps

- Continue to Post-installation steps for Linux.
- Review the topics in Develop with Docker to learn how to build new applications using Docker.

Linux post-installation steps for Docker Engine

These optional post-installation procedures describe how to configure your Linux host machine to work better with Docker.

Manage Docker as a non-root user

The Docker daemon binds to a Unix socket, not a TCP port. By default it's the `root` user that owns the Unix socket, and other users can only access it using `sudo`. The Docker daemon always runs as the `root` user.
If you don't want to preface the `docker` command with `sudo`, create a Unix group called `docker` and add users to it. When the Docker daemon starts, it creates a Unix socket accessible by members of the `docker` group. On some Linux distributions, the system automatically creates this group when installing Docker Engine using a package manager. In that case, there is no need for you to manually create the group.

Warning

The `docker` group grants root-level privileges to the user. For details on how this impacts security in your system, see Docker Daemon Attack Surface.

Note

To run Docker without root privileges, see Run the Docker daemon as a non-root user (Rootless mode).
To create the `docker` group and add your user:

1. Create the `docker` group.
2.
```
$ sudo groupadd docker
```
3. Add your user to the `docker` group.
4.
```
$ sudo usermod -aG docker $USER
```
5. Log out and log back in so that your group membership is re-evaluated.

   If you're running Linux in a virtual machine, it may be necessary to restart the virtual machine for changes to take effect.

   You can also run the following command to activate the changes to groups:

```
$ newgrp docker
```
6. Verify that you can run `docker` commands without `sudo`.
7.
```
$ docker run hello-world
```

This command downloads a test image and runs it in a container. When the container runs, it prints a message and exits.

If you initially ran Docker CLI commands using `sudo` before adding your user to the `docker` group, you may see the following error:

```
WARNING: Error loading config file:
/home/user/.docker/config.json -
stat /home/user/.docker/config.json: permission denied
```

This error indicates that the permission settings for the `~/.docker/` directory are incorrect, due to having used the `sudo` command earlier.
To fix this problem, either remove the `~/.docker/` directory (it's recreated automatically, but any custom settings are lost), or change its ownership and permissions using the following commands:

```
$ sudo chown "$USER":"$USER" /home/"$USER"/.docker -R
$ sudo chmod g+rwx "$HOME/.docker" -R
```

Configure Docker to start on boot with systemd

Many modern Linux distributions use systemd to manage which services start when the system boots. On Debian and Ubuntu, the Docker service starts on boot by default. To automatically start Docker and containerd on boot for other Linux distributions using systemd, run the following commands:

```
$ sudo systemctl enable docker.service
$ sudo systemctl enable containerd.service
```

To stop this behavior, use `disable` instead.

```
$ sudo systemctl disable docker.service
$ sudo systemctl disable containerd.service
```

If you need to add an HTTP proxy, set a different directory or partition for the Docker runtime files, or make other customizations, see customize your systemd Docker daemon options.

Configure default logging driver

Docker provides logging drivers for collecting and viewing log data from all containers running on a host. The default logging driver, `json-file`, writes log data to JSON-formatted files on the host filesystem. Over time, these log files expand in size, leading to potential exhaustion of disk resources.

To avoid issues with overusing disk for log data, consider one of the following options:

- Configure the `json-file` logging driver to turn on log rotation.

- Use an alternative logging driver such as the "local" logging driver that performs log rotation by default.
- Use a logging driver that sends logs to a remote logging aggregator.

Next steps

- Read the Get started training modules to learn how to build an image and run it as a containerized application.
- Review the topics in Develop with Docker to learn how to build new applications using Docker.