> User Documentation Home

User Handbook

- User Handbook Overview
- Installing Jenkins
  - Docker
  - Kubernetes
  - Linux
  - macOS
  - Windows
  - Other Systems
  - WAR file
  - Other Servlet Containers
  - Offline Installations
  - Initial Settings
- Platform Information
- Using Jenkins
- Pipeline
- Blue Ocean
- Managing Jenkins
- Securing Jenkins
- System Administration
- Scaling Jenkins
- Troubleshooting Jenkins
- Glossary

Tutorials

- Guided Tour
- Jenkins Pipeline
- Using Build Tools

Resources

- Pipeline Syntax reference
- Pipeline Steps reference
- LTS Upgrade guides

⇑ Installing Jenkins Index
Kubernetes ⇒

Docker

Table of Contents

Docker is a platform for running applications in an isolated environment called a "container" (or Docker container). Applications like Jenkins can be downloaded as read-only "images" (or Docker images), each of which is run in Docker as a container. A Docker container is a "running instance" of a Docker image. A Docker image is stored permanently, based on when image updates are published, whereas containers are stored temporarily. Learn more about these concepts in Getting Started, Part 1: Orientation and setup in the Docker documentation.

Due to Docker's fundamental platform and container design, a Docker image for a given application, such as Jenkins, can be run on any supported operating system or cloud service also running Docker. Supported operating systems include macOS, Linux and Windows, and supported cloud services include AWS and Azure.

Installing Docker

To install Docker on your operating system, follow the instructions in the Guided Tour prerequisites.

Alternatively, visit Docker Hub, and select the **Docker Community Edition** suitable for your operating system or cloud service. Follow the installation instructions on their website.

If you are installing Docker on a Linux-based operating system, ensure you configure Docker so it can be managed as a non-root user. Read more about this in Docker's Post-installation steps for Linux page of their documentation. This page also contains information about how to configure Docker to start on boot.

Prerequisites

Minimum hardware requirements:

- 256 MB of RAM
- 1 GB of drive space (although 10 GB is a recommended minimum if running Jenkins as a Docker container)

Recommended hardware configuration for a small team:

- 4 GB+ of RAM
- 50 GB+ of drive space

Comprehensive hardware recommendations:

- Hardware: see the Hardware Recommendations page

Software requirements:

- Java: see the Java Requirements page
- Web browser: see the Web Browser Compatibility page
- For Windows operating system: Windows Support Policy
- For Linux operating system: Linux Support Policy
- For servlet containers: Servlet Container Support Policy

Downloading and running Jenkins in Docker

There are several Docker images of Jenkins available.

Use the recommended official `jenkins/jenkins` image from the Docker Hub repository. This image contains the current Long-Term Support (LTS) release of Jenkins, which is production-ready. However, this image doesn’t contain Docker CLI, and is not bundled with the frequently used Blue Ocean plugins and its features. To use the full power of Jenkins and Docker, you may want to go through the installation process described below.

A new `jenkins/jenkins` image is published each time a new release of Jenkins Docker is published. You can see a list of previously published versions of the `jenkins/jenkins` image on the tags page.

On macOS and Linux

1. Open up a terminal window.
2. Create a bridge network in Docker using the following `docker network create` command:

```
docker network create jenkins
```

bash Copied!

3. In order to execute Docker commands inside Jenkins nodes, download and run the `docker:dind` Docker image using the following `docker run` command:

```
docker run \
  --name jenkins-docker \(1)
  --rm \(2)
  --detach \(3)
  --privileged \(4)
  --network jenkins \(5)
  --network-alias docker \(6)
  --env DOCKER_TLS_CERTDIR=/certs \(7)
  --volume jenkins-docker-certs:/certs/client \(8)
  --volume jenkins-data:/var/jenkins_home \(9)
  --publish 2376:2376 \(10)
  docker:dind \(11)
  --storage-driver overlay2(12)
```

bash Copied!

1 (*Optional*) Specifies the Docker container name to use for running the image. By default, Docker generates a unique name for the container.
2 (*Optional*) Automatically removes the Docker container (the replica of the Docker image) when it is shut down.
3 (*Optional*) Runs the Docker container in the background. You can stop this process by running `docker stop jenkins-docker`.
4 Running Docker in Docker currently requires privileged access to function properly. This requirement may be relaxed with newer Linux kernel versions.
5 This corresponds with the network created in the earlier step.
6 Makes the Docker in Docker container available as the hostname `docker` within the `jenkins` network.
7 Enables the use of TLS in the Docker server. Due to the use of a privileged container, this is recommended, though it requires the use of the shared volume described below. This environment variable controls the root directory where Docker TLS certificates are managed.
8 Maps the `/certs/client` directory inside the container to a Docker volume named `jenkins-docker-certs` as created above.
9 Maps the `/var/jenkins_home` directory inside the container to the Docker volume named `jenkins-data`. This allows for other Docker containers controlled by this Docker container’s Docker daemon to mount data from Jenkins.
10 (*Optional*) Exposes the Docker daemon port on the host machine. This is useful for executing `docker` commands on the host machine to control this inner Docker daemon.
11 The `docker:dind` image itself. Download this image before running, by using the command: `docker image pull docker:dind`.
12 The storage driver for the Docker volume. Refer to the Docker storage drivers documentation for supported options.

If you have problems copying and pasting the above command snippet, use the annotation-free version below:

```
docker run --name jenkins-docker --rm --detach \
  --privileged --network jenkins --network-alias docker \
  --env DOCKER_TLS_CERTDIR=/certs \
  --volume jenkins-docker-certs:/certs/client \
  --volume jenkins-data:/var/jenkins_home \
  --publish 2376:2376 \
  docker:dind --storage-driver overlay2
```

bash Copied!

4. Customize the official Jenkins Docker image, by executing the following two steps:

   a. Create a Dockerfile with the following content:

   ```
   FROM jenkins/jenkins:2.479.2-jdk17
   USER root
   RUN apt-get update && apt-get install -y lsb-release
   RUN curl -fsSLo /usr/share/keyrings/docker-archive-keyring.asc \
     https://download.docker.com/linux/debian/gpg
   RUN echo "deb [arch=$(dpkg --print-architecture) \
     signed-by=/usr/share/keyrings/docker-archive-keyring.asc] \
     https://download.docker.com/linux/debian \
     $(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list
   RUN apt-get update && apt-get install -y docker-ce-cli
   USER jenkins
   RUN jenkins-plugin-cli --plugins "blueocean docker-workflow"
   ```

   Copied!

   b. Build a new docker image from this Dockerfile, and assign the image a meaningful name, such as "myjenkins-blueocean:2.479.2-1":

   ```
   docker build -t myjenkins-blueocean:2.479.2-1 .
   ```

   bash Copied!

   If you have not yet downloaded the official Jenkins Docker image, the above process automatically downloads it for you.

5. Run your own `myjenkins-blueocean:2.479.2-1` image as a container in Docker using the following `docker run` command:

```
docker run \
  --name jenkins-blueocean \(1)
  --restart=on-failure \(2)
  --detach \(3)
  --network jenkins \(4)
  --env DOCKER_HOST=tcp://docker:2376 \(5)
  --env DOCKER_CERT_PATH=/certs/client \
  --env DOCKER_TLS_VERIFY=1 \
  --publish 8080:8080 \(6)
  --publish 50000:50000 \(7)
  --volume jenkins-data:/var/jenkins_home \(8)
  --volume jenkins-docker-certs:/certs/client:ro \(9)
  myjenkins-blueocean:2.479.2-1 (10)
```

bash Copied!

1 (*Optional*) Specifies the Docker container name for this instance of the Docker image.

2 Always restart the container if it stops. If it is manually stopped, it is restarted only when Docker daemon restarts or the container itself is manually restarted.

3 (*Optional*) Runs the current container in the background, known as "detached" mode, and outputs the container ID. If you do not specify this option, then the running Docker log for this container is displayed in the terminal window.

4 Connects this container to the `jenkins` network previously defined. The Docker daemon is now available to this Jenkins container through the hostname `docker`.

5 Specifies the environment variables used by `docker`, `docker-compose`, and other Docker tools to connect to the Docker daemon from the previous step.

6 Maps, or publishes, port 8080 of the current container to port 8080 on the host machine. The first number represents the port on the host, while the last represents the container’s port. For example, to access Jenkins on your host machine through port 49000, enter `-p 49000:8080` for this option.

7 (*Optional*) Maps port 50000 of the current container to port 50000 on the host machine. This is only necessary if you have set up one or more inbound Jenkins agents on other machines, which in turn interact with your `jenkins-blueocean` container, known as the Jenkins "controller". Inbound Jenkins agents communicate with the Jenkins controller through TCP port 50000 by default. You can change this port number on your Jenkins controller through the Security page. For example, if you update the **TCP port for inbound Jenkins agents** of your Jenkins controller to 51000, you need to re-run Jenkins via the `docker run …` command. Specify the "publish" option as follows: the first value is the port number on the machine hosting the Jenkins controller, and the last value matches the changed value on the Jenkins controller, for example,`--publish 52000:51000`. Inbound Jenkins agents communicate with the Jenkins controller on that port (52000 in this example). Note that WebSocket agents do not need this configuration.

8 Maps the `/var/jenkins_home` directory in the container to the Docker volume with the name `jenkins-data`. Instead of mapping the `/var/jenkins_home` directory to a Docker volume, you can also map this directory to one on your machine’s local file system. For example, specify the option `--volume $HOME/jenkins:/var/jenkins_home` to map the container’s `/var/jenkins_home` directory to the `jenkins` subdirectory within the `$HOME` directory on your local machine — typically `/Users/<your-username>/jenkins` or `/home/<your-username>/jenkins`. NOTE: If you change the source volume or directory for this, the volume from the `docker:dind` container above needs to be updated to match this.

9 Maps the `/certs/client` directory to the previously created `jenkins-docker-certs` volume. The client TLS certificates required to connect to the Docker daemon are now available in the path specified by the `DOCKER_CERT_PATH` environment variable.

10 The name of the Docker image, which you built in the previous step.

If you have problems copying and pasting the command snippet, use the annotation-free version below:

```
docker run --name jenkins-blueocean --restart=on-failure --detach \
  --network jenkins --env DOCKER_HOST=tcp://docker:2376 \
  --env DOCKER_CERT_PATH=/certs/client --env DOCKER_TLS_VERIFY=1 \
  --publish 8080:8080 --publish 50000:50000 \
  --volume jenkins-data:/var/jenkins_home \
  --volume jenkins-docker-certs:/certs/client:ro \
  myjenkins-blueocean:2.479.2-1
```

bash Copied!

6. Proceed to the Post-installation setup wizard.

On Windows

The Jenkins project provides a Linux container image, not a Windows container image. Be sure that your Docker for Windows installation is configured to run `Linux Containers` rather than `Windows Containers`. Refer to the Docker documentation for instructions to switch to Linux containers. Once configured to run `Linux Containers`, the steps are:

1. Open up a command prompt window and similar to the macOS and Linux instructions above do the following:

2. Create a bridge network in Docker

```
docker network create jenkins
```

bash Copied!

3. Run a docker:dind Docker image

```
docker run --name jenkins-docker --rm --detach ^
  --privileged --network jenkins --network-alias docker ^
  --env DOCKER_TLS_CERTDIR=/certs ^
  --volume jenkins-docker-certs:/certs/client ^
  --volume jenkins-data:/var/jenkins_home ^
  --publish 2376:2376 ^
  docker:dind
```

bash Copied!

4. Customize the official Jenkins Docker image, by executing the following two steps:

   a. Create a Dockerfile with the following content:

   ```
   FROM jenkins/jenkins:2.479.2-jdk17
   USER root
   RUN apt-get update && apt-get install -y lsb-release
   RUN curl -fsSLo /usr/share/keyrings/docker-archive-keyring.asc \
     https://download.docker.com/linux/debian/gpg
   RUN echo "deb [arch=$(dpkg --print-architecture) \
     signed-by=/usr/share/keyrings/docker-archive-keyring.asc] \
     https://download.docker.com/linux/debian \
     $(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list
   RUN apt-get update && apt-get install -y docker-ce-cli
   USER jenkins
   RUN jenkins-plugin-cli --plugins "blueocean docker-workflow"
   ```

   dockerfile Copied!

   b. Build a new docker image from this Dockerfile and assign the image a meaningful name, e.g. "myjenkins-blueocean:2.479.2-1":

   ```
   docker build -t myjenkins-blueocean:2.479.2-1 .
   ```

   bash Copied!

   If you have not yet downloaded the official Jenkins Docker image, the above process automatically downloads it for you.

5. Run your own `myjenkins-blueocean:2.479.2-1` image as a container in Docker using the following `docker run` command:

```
docker run --name jenkins-blueocean --restart=on-failure --detach ^
  --network jenkins --env DOCKER_HOST=tcp://docker:2376 ^
  --env DOCKER_CERT_PATH=/certs/client --env DOCKER_TLS_VERIFY=1 ^
  --volume jenkins-data:/var/jenkins_home ^
  --volume jenkins-docker-certs:/certs/client:ro ^
  --publish 8080:8080 --publish 50000:50000 myjenkins-blueocean:2.479.2-1
```

bash Copied!

6. Proceed to the Setup wizard.

Accessing the Docker container

If you want to access your Docker container through a terminal/command prompt using the `docker exec` command, add an option like `--name jenkins-tutorial` to the `docker exec` command. That will access the Jenkins Docker container named "jenkins-tutorial".

You can access your docker container (through a separate terminal/command prompt window) with a `docker exec` command such as:

```
docker exec -it jenkins-blueocean bash
```

Accessing the Docker logs

You may want to access the Jenkins console log, for instance, when Unlocking Jenkins as part of the Post-installation setup wizard.

Access the Jenkins console log through the terminal/command prompt window from which you executed the `docker run …` command. Alternatively, you can also access the Jenkins console log through the Docker logs of your container using the following command:

```
docker logs <docker-container-name>
```

Your `<docker-container-name>` can be obtained using the `docker ps` command.

Accessing the Jenkins home directory

You can access the Jenkins home directory, to check the details of a Jenkins build in the `workspace` subdirectory, for example.

If you mapped the Jenkins home directory (`/var/jenkins_home`) to one on your machine's local file system, for example, in the `docker run …` command above, access the directory contents through your machine's usual terminal/command prompt.

If you specified the `--volume jenkins-data:/var/jenkins_home` option in the `docker run …` command, access the contents of the Jenkins home directory through your container’s terminal/command prompt using the `docker container exec` command:

```
docker container exec -it <docker-container-name> bash
```

As per the previous section, get your `<docker-container-name>` using the `docker container ls` command. If you specified the `--name jenkins-blueocean` option in the `docker container run …` command above (refer to Accessing the Jenkins/Blue Ocean Docker container if needed), use the `docker container exec` command:

```
docker container exec -it jenkins-blueocean bash
```

Post-installation setup wizard

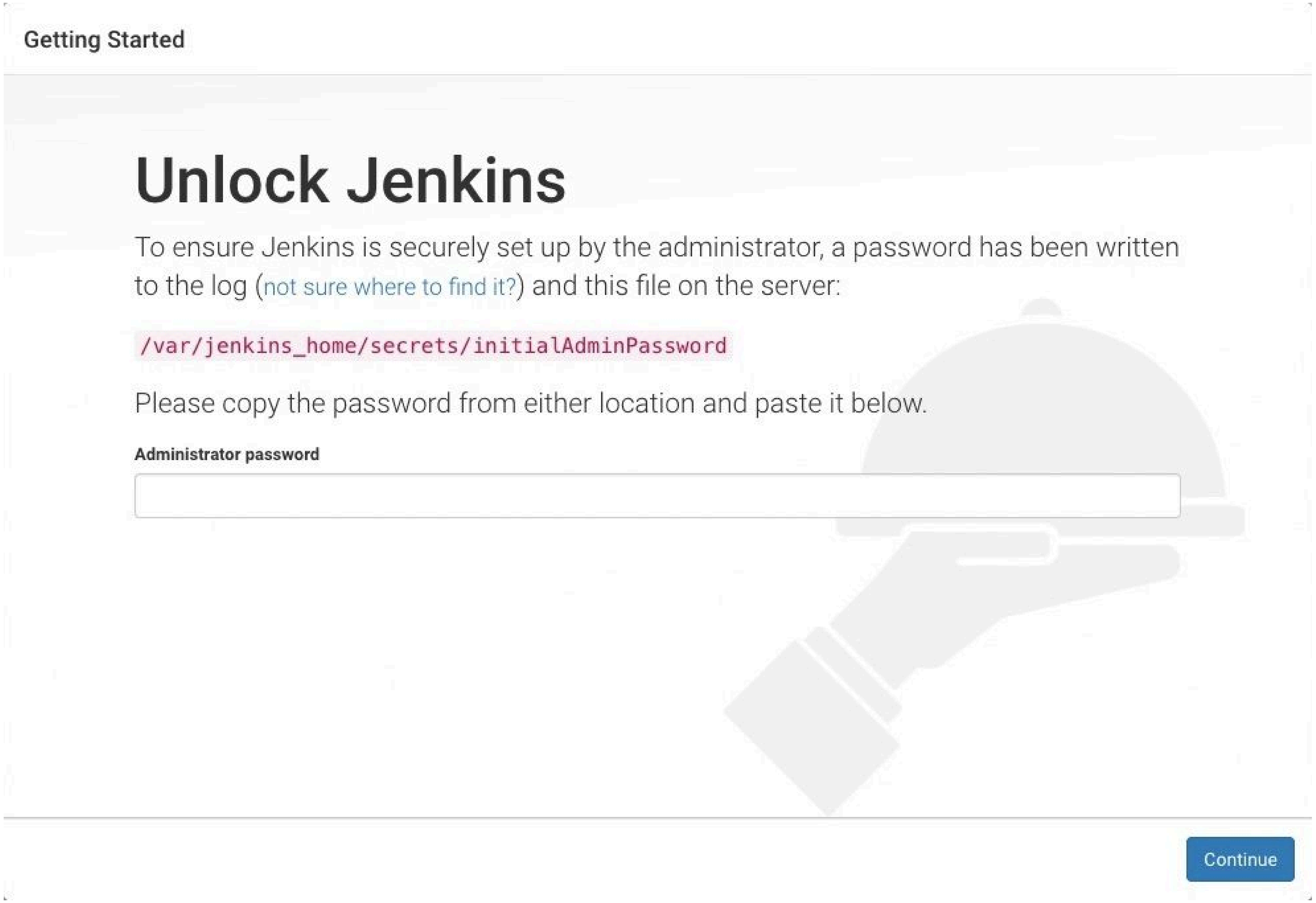
After downloading, installing and running Jenkins using one of the procedures above (except for installation with Jenkins Operator), the post-installation setup wizard begins.

This setup wizard takes you through a few quick "one-off" steps to unlock Jenkins, customize it with plugins and create the first administrator user through which you can continue accessing Jenkins.

Unlocking Jenkins

When you first access a new Jenkins controller, you are asked to unlock it using an automatically-generated password.

1. Browse to `http://localhost:8080` (or whichever port you configured for Jenkins when installing it) and wait until the **Unlock Jenkins** page appears.



Getting Started

Unlock Jenkins

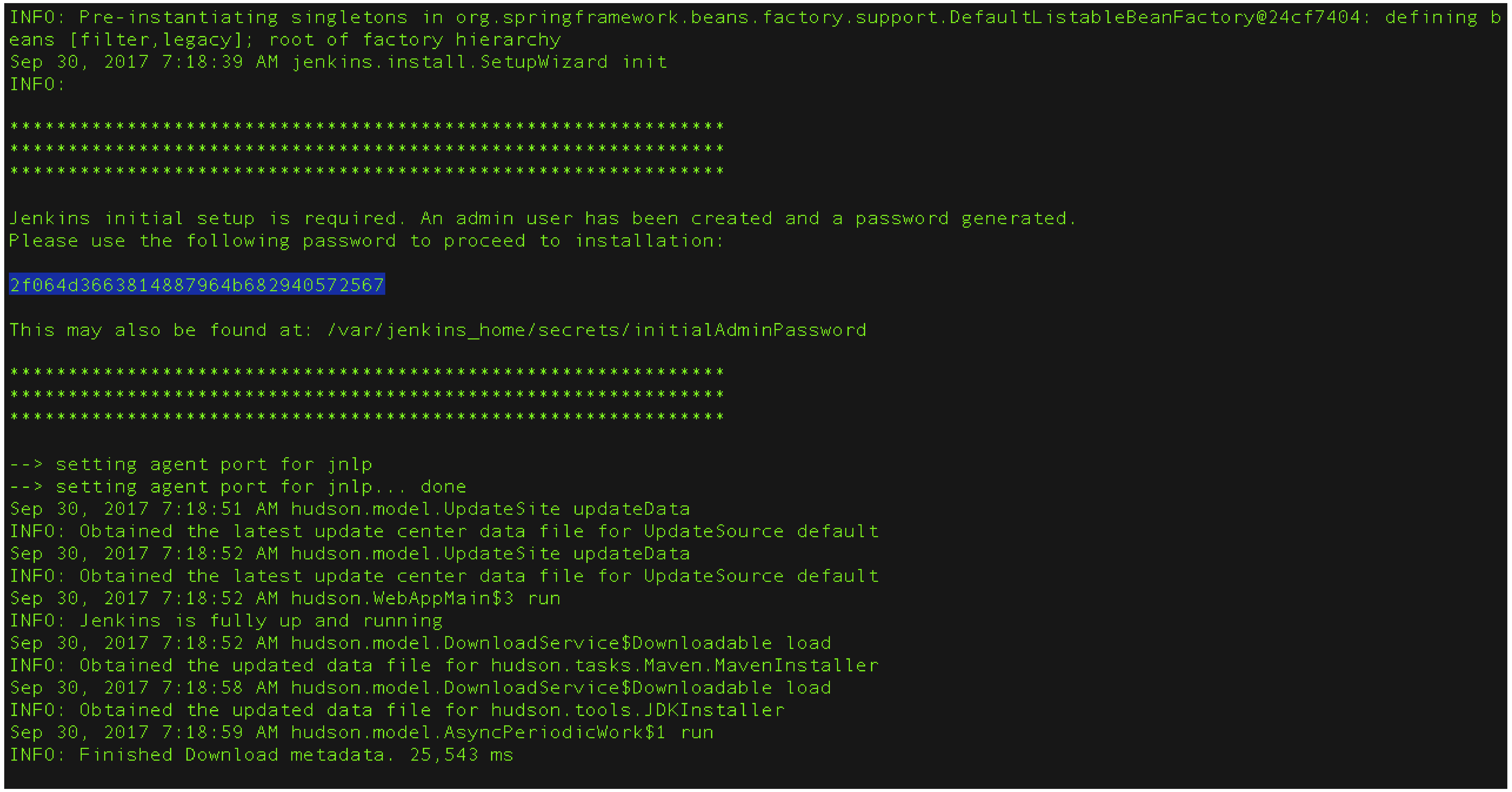
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

2. From the Jenkins console log output, copy the automatically-generated alphanumeric password (between the 2 sets of asterisks).



```
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@24cf7404: defining b
eans [filter,legacy]; root of factory hierarchy
Sep 30, 2017 7:18:39 AM jenkins.install.SetupWizard init
INFO:

*************************************************************
*************************************************************
*************************************************************

Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:

2f064d3663814887964b682940572567

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword

*************************************************************
*************************************************************
*************************************************************

--> setting agent port for jnlp
--> setting agent port for jnlp... done
Sep 30, 2017 7:18:51 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Sep 30, 2017 7:18:52 AM hudson.WebAppMain$3 run
INFO: Jenkins is fully up and running
Sep 30, 2017 7:18:52 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
Sep 30, 2017 7:18:58 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tools.JDKInstaller
Sep 30, 2017 7:18:59 AM hudson.model.AsyncPeriodicWork$1 run
INFO: Finished Download metadata. 25,543 ms
```

Note:

- The command: `sudo cat /var/lib/jenkins/secrets/initialAdminPassword` will print the password at console.
- If you are running Jenkins in Docker using the official `jenkins/jenkins` image you can use `sudo docker exec ${CONTAINER_ID or CONTAINER_NAME} cat /var/jenkins_home/secrets/initialAdminPassword` to print the password in the console without having to exec into the container.

3. On the **Unlock Jenkins** page, paste this password into the **Administrator password** field and click **Continue**.
   Note:
   - The Jenkins console log indicates the location (in the Jenkins home directory) where this password can also be obtained. This password must be entered in the setup wizard on new Jenkins installations before you can access Jenkins's main UI. This password also serves as the default administrator account's password (with username "admin") if you happen to skip the subsequent user-creation step in the setup wizard.

Customizing Jenkins with plugins

After unlocking Jenkins, the **Customize Jenkins** page appears. Here you can install any number of useful plugins as part of your initial setup.

Click one of the two options shown:

- **Install suggested plugins** - to install the recommended set of plugins, which are based on most common use cases.
- **Select plugins to install** - to choose which set of plugins to initially install. When you first access the plugin selection page, the suggested plugins are selected by default.

If you are not sure what plugins you need, choose **Install suggested plugins**. You can install (or remove) additional Jenkins plugins at a later point in time via the **Manage Jenkins** > **Plugins** page in Jenkins.

The setup wizard shows the progression of Jenkins being configured and your chosen set of Jenkins plugins being installed. This process may take a few minutes.

Creating the first administrator user

Finally, after customizing Jenkins with plugins, Jenkins asks you to create your first administrator user.

1. When the **Create First Admin User** page appears, specify the details for your administrator user in the respective fields and click **Save and Finish**.
2. When the **Jenkins is ready** page appears, click **Start using Jenkins**.
   Notes:

- This page may indicate **Jenkins is almost ready!** instead and if so, click **Restart**.
- If the page does not automatically refresh after a minute, use your web browser to refresh the page manually.

3. If required, log in to Jenkins with the credentials of the user you just created and you are ready to start using Jenkins!

⇑ Installing Jenkins Index
Kubernetes ⇒

Was this page helpful?

Please submit your feedback about this page through this quick form.

Alternatively, if you don't wish to complete the quick form, you can simply indicate if you found this page helpful?

Yes No

Type the answer to 2 plus 2 before clicking "Submit" below.

Submit

See existing feedback here.

Improve this page Report page issue

Resources
Downloads
Blog
Documentation
Plugins
Security
Contributing

Project
Structure and governance
Issue tracker
Roadmap
GitHub
Jenkins on Jenkins

Community
Forum
Events
Mailing lists
Chats
Special Interest Groups
𝕏 (formerly Twitter)
Reddit

Other
Code of Conduct
Press information
Merchandise
Artwork
Awards