

Jenkins and Docker go well together. Containers are a powerful tool for building software in different environments, and Jenkins itself is easy to run in a container. But how do you manage a Jenkins controller and agents together in Docker?

In this tutorial, you'll see how easy it is to install and run Jenkins with Docker Compose. Let's get started.

What Is Docker Compose?

Sometimes you need to run more than one container, but you don't want or need the complexity of Kubernetes. You also don't want to manage a bunch of shell scripts, especially when the containers need to talk to each other.

You need Docker Compose. It's a tool for defining how to run multiple containers in a single configuration file and start, stop, and restart them with a single command.

Since Docker Compose lets you configure related containers in a single YAML file, you get the same Infrastructure-as-Code abilities as Kubernetes. But they come in a simpler system that's more suited to smaller applications that don't need Kubernetes' resiliency and scaling.

What Is Jenkins?

Jenkins is an automation server. While you can use it to automate just about any task, it's most often associated with building source code and deploying the results. For many, Jenkins is synonymous with continuous integration and continuous delivery (CI/CD).

One of Jenkins's most powerful features is its ability to distribute jobs across multiple nodes. A Jenkins controller sends jobs to the appropriate agent based on the job requirements and the

```
1 # docker-compose.yml
2 version: '3.8'
3 services:
4   jenkins:
5     image: jenkins/jenkins:lts
6     privileged: true
7     user: root
8     ports:
9       - 8080:8080
10      - 50000:50000
11     container_name: jenkins
12     volumes:
13       -
14         /home/${myname}/jenkins_compose/jenkins_configuration:/var/jenkins_home
15       - /var/run/docker.sock:/var/run/docker.sock
```

Line #1 is a comment.

Line #2 tells Docker Compose which version of the Compose specification we're using. As of the time of this writing, 3.8 is the latest.

Line #3 starts defining services. For now, we just have one.

The rest of the file defines the Jenkins container.

It will run the latest Jenkins image with root privileges. We're running the container with host networking, so lines #9 and #10 tell Docker to redirect ports 8080 and 50000 to the host's

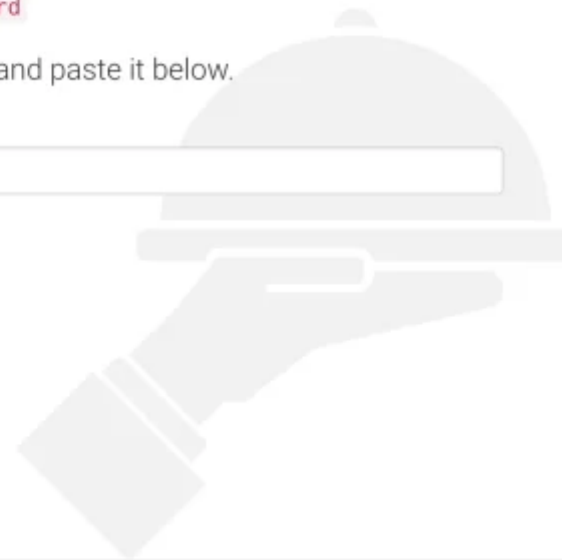
Unlock Jenkins

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



Continue

The page tells you to find the initial password in a log file, but Jenkins prints the initial password to standard output too. So, you can retrieve it from the Docker log.

Go back to your shell and view the logs with **docker logs**.

```
$ docker logs jenkins | less
```

Look for a block enclosed with six lines of asterisks like this:

Create First Admin User

Username:	<input type="text" value="admin"/>
Password:	<input type="password" value="*****"/>
Confirm password:	<input type="password" value="*****"/>
Full name:	<input type="text" value="Administrator"/>
E-mail address:	<input type="text" value="eric@ericgoebelbecke@"/>

Instance Configuration

Jenkins URL:

<http://192.168.7.63:8080/>

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

```

egoebelbecker@zaku:~/jenkins_compose$ ssh-keygen -t rsa -f jenkins_agent
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in jenkins_agent
Your public key has been saved in jenkins_agent.pub
The key fingerprint is:
SHA256:qxw6RHZ8x302wIH4qChkvGmsl4DY6C1FHdBsPhFQjtE egoebelbecker@zaku
The key's randomart image is:
+---[RSA 3072]-----+
|  +Bo. . o.. |
|  +E . . o |
|  .+oo + . . |
|  +.o+o o + . + |
|o*.= o.oS. o . |
| = B.o . . |
| .+o+ . . |
| .oo..o o |
| .. ..o |
+---[SHA256]-----+

```

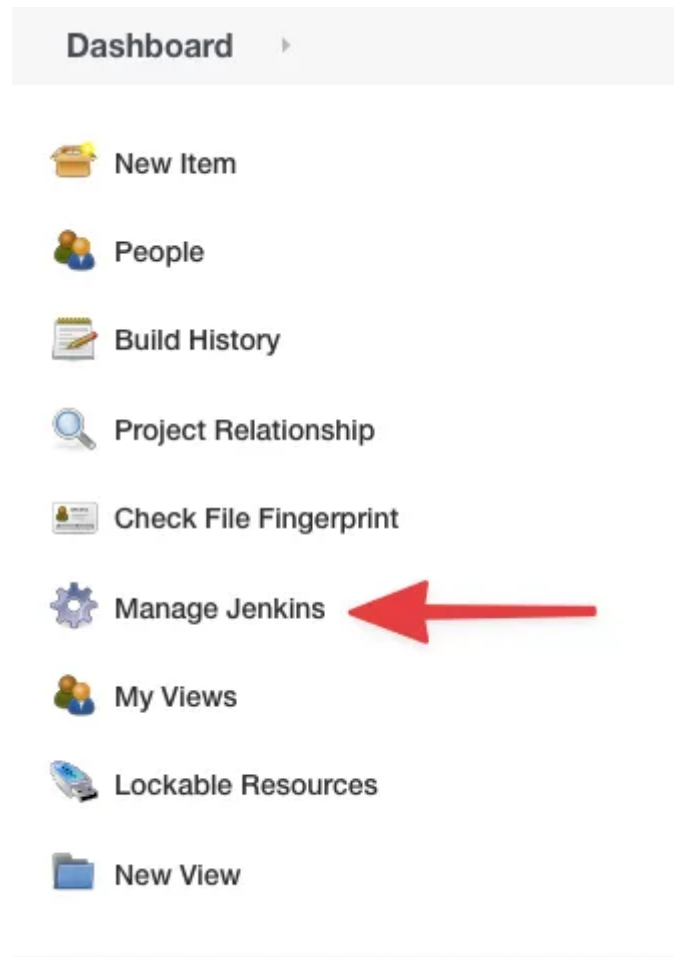
This command creates two files: **jenkins_agent**, which holds the private key, and **jenkins-agent.pub**, the public key.

Here are the contents of **jenkins_agent** (provided as an example for the purposes of this tutorial).

```


egoebelbecker@zaku:~/jenkins_compose$ cat jenkins_agent
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAABAG5vbmUAAAABbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAEYA60hdlyArnMV//4c9D1YA1b9xIrG0gvfbvbANUKGUnR7l1vx+Yv9Q
U9wrVMwQLAjicXUrmnSnyWZzQxC/fJJZmoFO/bQIDY1YAp1MqChJGhDs+CeNenFLBf4Uf6y
lHkYpnKMegZ3MIsWLH5pnGTKNSJilqnWCA7ukyVdEY4XaWcn59UU8VdFVz9XJ+cizuGhs/
GQjy5r8VTjcpHNBSwVc914G6RVvcWNzuImkq6aGJ/QntJvk8cdJT2JZuW7egtheoadC2gQ
ArHpAGc9KAFwigHpFYI9u5fL1zY9Hjdi fH8UHbEw7Wej553e9wE3ytdeHHiwxy9lclfJtS
sQ+QpvsxYAewH2dMQbDfihgbT6oWu/e9ZWUll4yr0TV5z5N8a88Ezxii/L5yTlInbMfuis
0kzwf7JPC/WLBMvYtNtymU0DF079CKy6fAW5t1eMFhrHX5r4izKt8dkDEjF5aRNNGpQmq9
nkmod0NuTzkqchLSxRA49rVEVYz/i5iq6HPGziC3AAAFiI/CzQyPws0MAAAAB3NzaC1yc2
EAAAGBAOtIXZcaK5zFf/+HP09WANW/cSKxioL3272wDVChlJ0e5db8fml/UFPcK1TMECwI


```



Then go to **Manage Credentials**.

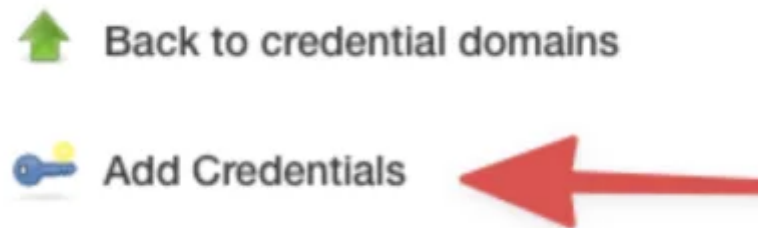


 **System**

Domain	Description
 Global credentials (unrestricted)	Credentials that should be available irrespective of domain specification to requirements matching.

Icon: [S](#) [M](#) [L](#)

Finally, click **Add Credentials** in the menu on the left.




Set these options on this screen.

1. Select **SSH Username with private key**.
2. Limit the scope to System. This means the key can't be used for jobs.
3. Give the credential an ID.
4. Provide a description.
5. Enter **jenkins** for a username. Don't use the username used to create the key.
6. Under **Private Key**, check **Enter directly**.

Now, paste the contents of **jenkins_agent** in the text box.


```
1 # docker-compose.yml
2 version: '3.8'
3 services:
4   jenkins:
5     image: jenkins/jenkins:lts
6     privileged: true
7     user: root
8     ports:
9       - 8080:8080
10      - 50000:50000
11     container_name: jenkins
12     volumes:
13       -
14         /home/${myname}/jenkins_compose/jenkins_configuration:/var/jenkins_home
15         - /var/run/docker.sock:/var/run/docker.sock
16   agent:
17     image: jenkins/ssh-agent:jdk11
18     privileged: true
19     user: root
20     container_name: agent
21     expose:
22       - 22
23     environment:
24       - JENKINS_AGENT_SSH_PUBKEY=ssh-rsa
25       AAAAB3NzaC1yc2EAAAADAQABAAQGBgQDrSF2XICucxX//hz0PVgDVv3EisY6C99u9sA1QoZSdHuXW/
26       egoebelbecker@zaku
```

 [Back to Dashboard](#)

 [Manage Jenkins](#)

 [New Node](#)



 [Configure Clouds](#)

 [Node Monitoring](#)

Now define your Jenkins agent.

Name

jenkins_agent

Description

Number of executors

1

Remote root directory

/home/jenkins/agent

Labels

Usage

Use this node as much as possible

Launch method

Launch agents via SSH

Host

agent

Credentials

✓ jenkins (Jenkins Agent Key)

Add ▾

Host Key Verification Strategy

Non verifying Verification Strategy

Availability

Keep this agent online as much as possible

Click **Advanced** on the right.

Advanced...

Host Key Verification Strategy

Non verifying Verification Strategy

Port

22

JavaPath

/usr/local/openjdk-11/bin/java

JVM Options

Prefix Start Agent Command

Suffix Start Agent Command

Connection Timeout in Seconds

60

Maximum Number of Retries

10

Seconds To Wait Between Retries

15

```

JENKINS_AGENT_SSH_PUBKEY='ssh-rsa
AAAAB3NzaClyc2EAAAADAQABAAQGDrsF2XICucxX//h201...vv3E1f6C99d9akig0z0d0uM/H5i/1BT3CtUzBAaCOJxdSuadKfJZnNDEL98lmagU79tAgNjVgCnUyoKEkaEOz4J416cUaF/h
04aGz8ZCPLwvxVORykc0FLBVz3XgbpFW9xY3O4iaSrpoYn9Ce0m+Txx01PYlm5bt6C2F6hp0LaBACsekAZz0oAXCKAel9gJ2718vXNj0eN2J8fxQdeTDtZ6Pnd73ATfK114eeL9HL2VYV8m1KxD5
TPB/sk8L9YsEy9i023KZTQMXTv0Irlp8Bbm3V4wGadfmviLMq3x2QMSMXlpE00alCar2eSah0425POSpyEtLFEDj2tURVjP+LmKroc8boILc= egoebelbecker@zaku'
LOGNAME=jenkins
MACHTYPE=x86_64-pc-linux-gnu
MAIL=/var/mail/jenkins
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/local/bin:/usr/bin:/bin:/usr/games
PIPESTATUS=([0]="0")
PFID=46
PS4='+ '
PWD=/home/jenkins
SHELL=/bin/bash
SHELLOPTS=braceexpand:hashall:interactive-comments
SHLVL=1
SSH_CLIENT='172.22.0.2 59058 22'
SSH_CONNECTION='172.22.0.2 59058 172.22.0.3 22'
TERM=dumb
UID=1000
USER=jenkins
_=']'
[11/16/21 19:29:28] [SSH] Starting sftp client.
[11/16/21 19:29:28] [SSH] Remote file system root /home/jenkins/agent does not exist. Will try to create it...
[11/16/21 19:29:28] [SSH] Copying latest remoting.jar...
[11/16/21 19:29:29] [SSH] Copied 1,507,326 bytes.
Expanded the channel window size to 4MB
[11/16/21 19:29:29] [SSH] Starting agent process: cd "/home/jenkins/agent" && /usr/local/openjdk-11/bin/java -jar remoting.jar -workDir /home/jenkin
Nov 16, 2021 7:29:29 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using /home/jenkins/agent/remoting as a remoting work directory
Nov 16, 2021 7:29:29 PM org.jenkinsci.remoting.engine.WorkDirManager setupLogging
INFO: Both error and output logs will be printed to /home/jenkins/agent/remoting
<===[JENKINS REMOTING CAPACITY]==>channel started
Remoting version: 4.10
This is a Unix agent
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by jenkins.slaves.StandardOutputSwapper$ChannelSwapper to constructor java.io.FileDescriptor(int)
WARNING: Please consider reporting this to the maintainers of jenkins.slaves.StandardOutputSwapper$ChannelSwapper
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
Evacuated stdout
Agent successfully connected and online

```

The most important entry you'll see is **Agent successfully connected and online**.

But if you look at the other entries, you'll see plenty of information about your agent, including the SSH key.