

**SoftWare Repository for Container**

# Getting Started

**Issue**            02  
**Date**             2023-08-03



**Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2024. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

## **Trademarks and Permissions**



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

## **Notice**

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

---

# Contents

---

<b>1 Uploading an Image.....</b>	<b>1</b>
<b>2 Pulling an Image to Deploy an Application in a CCE Cluster.....</b>	<b>8</b>

# 1 Uploading an Image

---

## Introduction

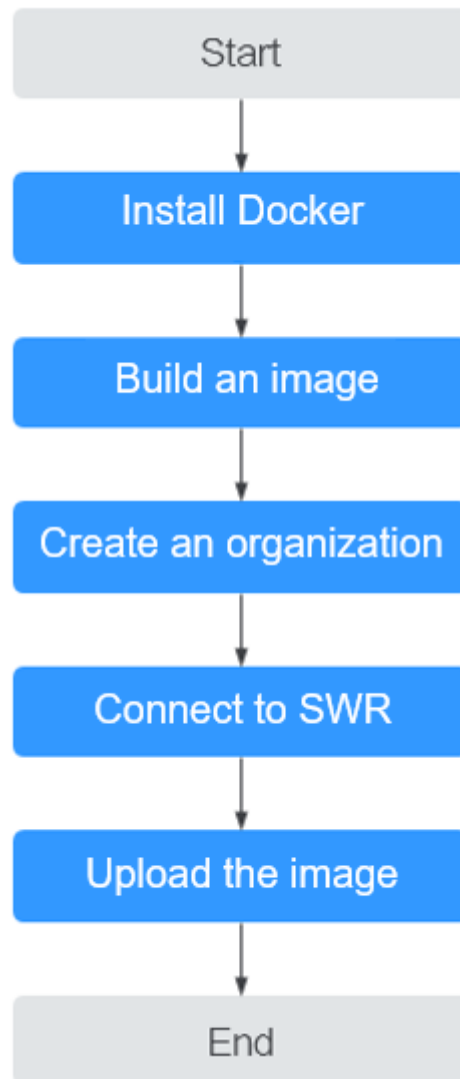
SWR provides easy, secure, and reliable management for container images throughout their lifecycles, facilitating the deployment of containerized applications. This section uses the 2048 application as an example to show how to install the container engine, build an image, and use the container engine to upload the image to SWR.

### NOTE

- There are two ways to upload an image to SWR: using a container engine client or the SWR console. This section describes how to use a container engine client to upload an image. For details about how to upload an image on the console, see [Uploading an Image Through SWR Console](#).
- Currently, there are no SWR APIs available for uploading images.

The following diagram shows the process of uploading an image to SWR.

**Figure 1-1** Process



## Prerequisites

You already have a Huawei Cloud account.

If you do not have a Huawei Cloud account, perform the following steps to create one:

1. Visit [Huawei Cloud](#) and click **Sign Up**.
2. On the page displayed, sign up for an account. After you have successfully signed up, the system automatically redirects you to your personal information page.

## Step 1: Install the Container Engine

Prepare an ECS, on which Docker **1.11.2** or later is installed.

- Step 1** Create a Linux ECS with an EIP. For details, see [Purchasing and Logging In to a Linux ECS](#).

For demonstration, you do not need to select high-specification ECS and public IP address. For example, select the ECS with **1 vCPUs | 2 GB**, the public IP bandwidth of **1 Mbit/s**, and the operating system with **CentOS 7.5**.

 **NOTE**

- You can also install the container engine on other machines.
- If you use a **CentOS**, you are advised to use CentOS 7, CentOS 7.2, CentOS 7.3, CentOS 7.4, CentOS 7.5 or CentOS 7.6 to avoid exceptions during the installation.

**Step 2** After the ECS is created, return to the ECS list and click **Remote Login** to log in to the ECS as user **root**.

**Step 3** Run the following commands to quickly install the container engine:

```
curl -fsSL get.docker.com -o get-docker.sh
sh get-docker.sh
sudo systemctl daemon-reload
sudo systemctl restart docker
```

----End

## Step 2: Build an Image

**Step 1** Run the following command on the ECS where Docker is installed to download the source code of the 2048 application:

```
git clone https://gitee.com/jorgensen/2048.git
```

 **NOTE**

If the message "git: command not found" is displayed, the Git tool is not installed. In this case, run the **yum install git** command to install it first.

**Step 2** After the download is successful, access the **2048** directory.

```
cd 2048
```

**Step 3** Modify the Dockerfile file.

**vim Dockerfile**

```
FROM nginx
COPY ./usr/share/nginx/html

EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

- **FROM:** specifies **nginx** as the base image.
- **COPY:** copies the source code of the 2048 application to the directory **/usr/share/nginx/html** in the container.
- **EXPOSE:** exposes port 80 of the container.
- **CMD:** specifies the default command to run the container.

Press **Esc** and enter **:wq** to save the settings and exit.

**Step 4** Run the **docker build** command to build an image.

```
docker build -t 2048 .
```

In the preceding command:

- **-t** indicates to label the image, that is, to name the image. In this example, the image name is **2048**.
- **.** indicates the context path. All contents in this path are packed and sent to the container engine to build an image.

**Step 5** Run the following command to view the 2048 image that has been successfully built. The image tag is **latest** by default.

#### docker images

```
# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
2048 latest 8d421c503ed0 About a minute ago 134MB
nginx latest dd34e67e3371 6 days ago 133MB
```

You can also see a nginx image, which is pulled from the image repository and used as the base image of the 2048 image.

**Step 6** (Optional) Run the container image.

After the image is successfully built, you can run the **docker run** command to run the container image.

**docker run -p 8080:80 2048**

The **docker run** command starts a container. In the preceding command, **-p** indicates that port 8080 of the VM is mapped to port 80 of the container. When you access **https://EIP of the ECS:8080**, the container is accessed. The 2048 application page is displayed.

----End

## Step 3: Create an Organization

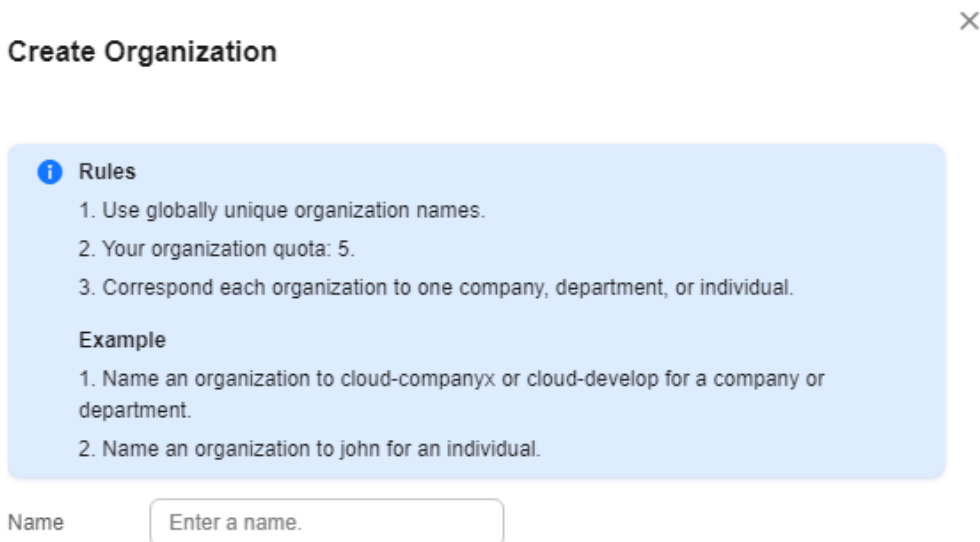
An organization is used to isolate images and grant access permissions, namely, read, edit, and manage, to different users under an account.

**Step 1** Log in to the SWR console.

**Step 2** In the navigation pane on the left, choose **Organization Management**. On the displayed page, click **Create Organization** in the upper right corner.

**Step 3** Enter the organization name and click **OK**.

Figure 1-2 Creating an organization



----End

## Step 4: Connect to SWR

**Step 1** Log in to the SWR console.


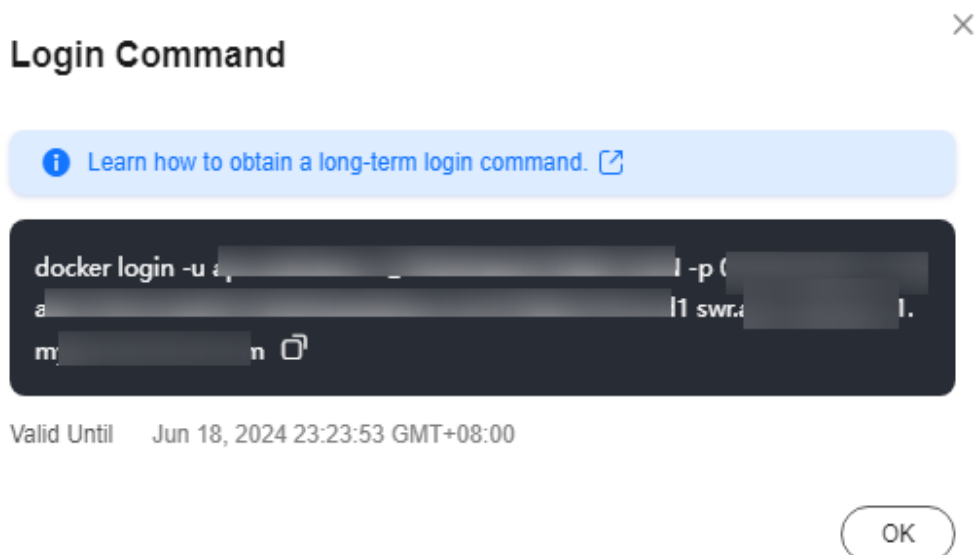
**Step 2** In the navigation pane on the left, choose **Dashboard** and click **Generate Login Command** in the upper right corner. On the displayed page, click  to copy the login command.

Figure 1-3 Generating a login command





 NOTE

The validity period of the generated login command is 6 hours. To obtain a long-term valid login command, see [Obtaining a Long-Term Valid Login Command](#).

**Step 3** Run the login command on the VM where the container engine is installed.

The message **Login Succeeded** will be displayed upon a successful login.

----End

## Step 5: Upload the Image

**Step 1** Run the following command to label the 2048 image on the VM where the container engine is installed:

```
docker tag[Image name 1:tag 1] [Image repository address]/[Organization name]/[Image name 2:tag 2]
```

In the preceding command:

- **[Image name 1:tag 1]**: name and tag of the image to be uploaded.
- **[Image repository address]**: The domain name at the end of the login command in [Step 4: Connect to SWR](#) is the image repository address, which can be obtained on the SWR console.
- **[Organization name]**: name of the organization created in [Step 3: Create an Organization](#).
- **[Image name 2:tag 2]**: desired image name and tag.

Example:

```
docker tag 2048:latest swr.ap-southeast-1.myhuaweicloud.com/cloud-develop/2048:v1
```

**Step 2** Push the image to the image repository.

```
docker push [Image repository address]/[Organization name]/[Image name 2:tag 2]
```

Example:

```
docker push swr.ap-southeast-1.myhuaweicloud.com/cloud-develop/2048:v1
```

The following information will be returned upon a successful push:

```
The push refers to repository [swr.ap-southeast-1.myhuaweicloud.com/cloud-develop/2048]
fbce26647e70: Pushed
fb04ab8effa8: Pushed
8f736d52032f: Pushed
009f1d338b57: Pushed
678bbd796838: Pushed
d1279c519351: Pushed
f68ef921efae: Pushed
v1: digest: sha256:0cdfc7910db531bfa7726de4c19ec556bc9190aad9bd3de93787e8bce3385f8d size: 1780
```

To view the pushed image, go to the SWR console and refresh the **My Images** page.

**Step 3** Use the pushed image to **deploy a workload on CCE**.  
----End

# 2 Pulling an Image to Deploy an Application in a CCE Cluster

---

You can use an image to quickly deploy a single-pod workload that can be accessed from the public network. This section describes how to use an image to deploy Nginx in a CCE cluster.

## Prerequisites

You have a CCE cluster that contains at least one node with 4 vCPUs and 8 GiB memory. The node is bound with an EIP.

## Deploying an Application

- Step 1** Log in to the [CCE console](#).
- Step 2** Click the name of the target cluster to access the cluster console.
- Step 3** In the navigation pane, choose **Workloads**. Then, click **Create Workload**.
- Step 4** Configure the following parameters and retain the default value for other parameters:

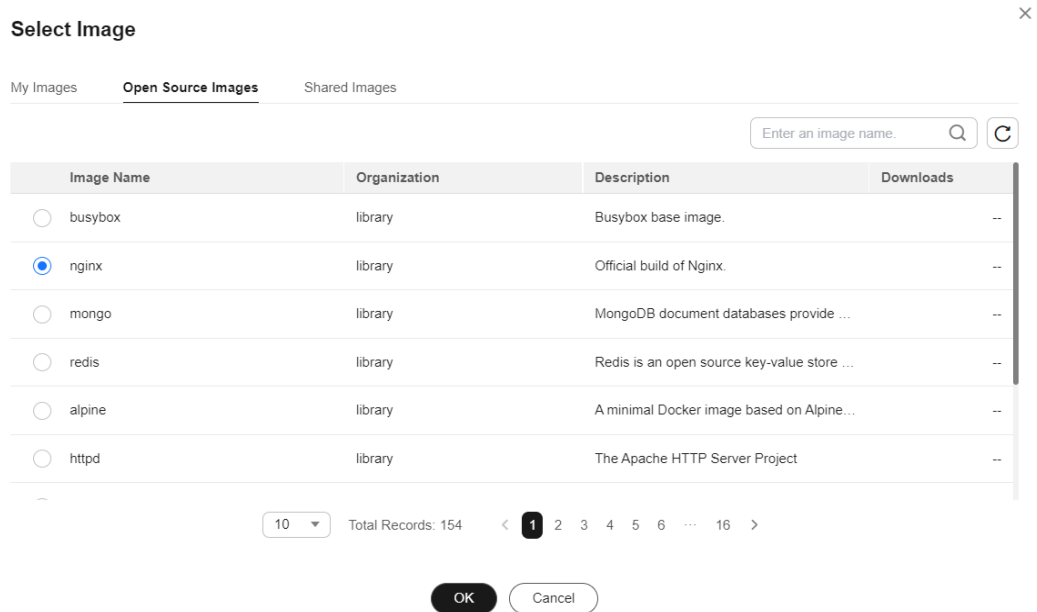
### Basic Info

- **Workload Type:** Select **Deployment**.
- **Workload Name:** Set it to **nginx**.
- **Namespace:** Select **default**.
- **Pods:** Set the quantity of pods to **1**.

### Container Settings

In the **Container Information** area, click **Basic Info** and click **Select Image**. In the dialog box displayed, click the **Open Source Images** tab, search for **nginx**, and select the **nginx** image.

**Figure 2-1** Selecting the nginx image



### Service Settings

Click the plus sign (+) to create a Service for accessing the workload from an external network. This example shows how to create a LoadBalancer. Configure the following parameters in the window that slides out from the right:

- **Service Name:** Enter **nginx**. The name of the Service is exposed to external networks.
- **Service Type:** Select **LoadBalancer**.
- **Service Affinity:** Retain the default value.
- **Load Balancer:** If a load balancer is available, select an existing load balancer. If not, select **Auto create** to create one.
- **Ports:**
  - **Protocol:** Select **TCP**.
  - **Service Port:** Set this parameter to **8080**, which is mapped to the container port.
  - **Container Port:** port on which the application listens. For containers created using the nginx image, set this parameter to **80**. For other applications, set this parameter to the port of the application.

**Figure 2-2** Creating a Service

**Create Service**

Service Name:

Service Type:

- ClusterIP**: Expose services through the internal IP of the cluster, which can only be accessed within the cluster
- NodePort**: Expose services via IP and static port (NodePort) on each node
- LoadBalancer** (Selected): Provide external services through ELB load balancing, high availability, ultra-high performance, stability and security
- DNAT**: Expose cluster node access type services through NAT gateway, support multiple nodes to share and use elastic IP

It is recommended to select the load balancing access type for out-of-cluster access

Service Affinity: **Cluster-level** | Node-level

Load Balancer: **Shared** | Use existing... | [Create Load Balancer](#)

Only shared load balancers in VPC vpc-cce where the cluster resides are supported.  
Set ELB: Load balancing algorithm: Weighted round robin; Sticky session: Disable; [I have read Notes on Using Load Balancers.](#)

Health Check: **Global health check** | Disable | Custom health check  
protocol: TCP | delay(s): 5 | timeout(s): 10 | maxRetries: 3

Protocol	Container Port	Service Port	Operation
TCP	8080	80	Delete

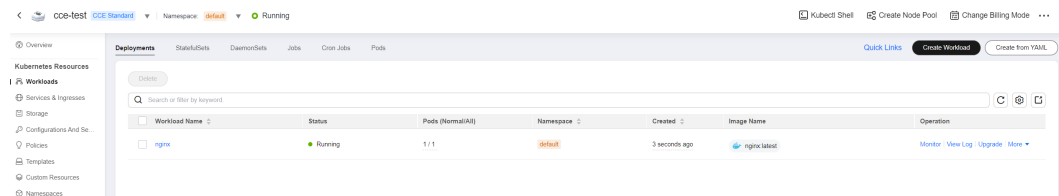
Annotation: Key = Value | [Confirm](#) | [Quick Links](#)

**Step 5** Click **Create Workload**.

Wait until the workload is created.

The created Deployment will be displayed on the **Deployments** tab.

**Figure 2-3** Workload created successfully



----End