Cloud Container Instance

Getting Started

Issue 01

Date 2024-07-25

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

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.

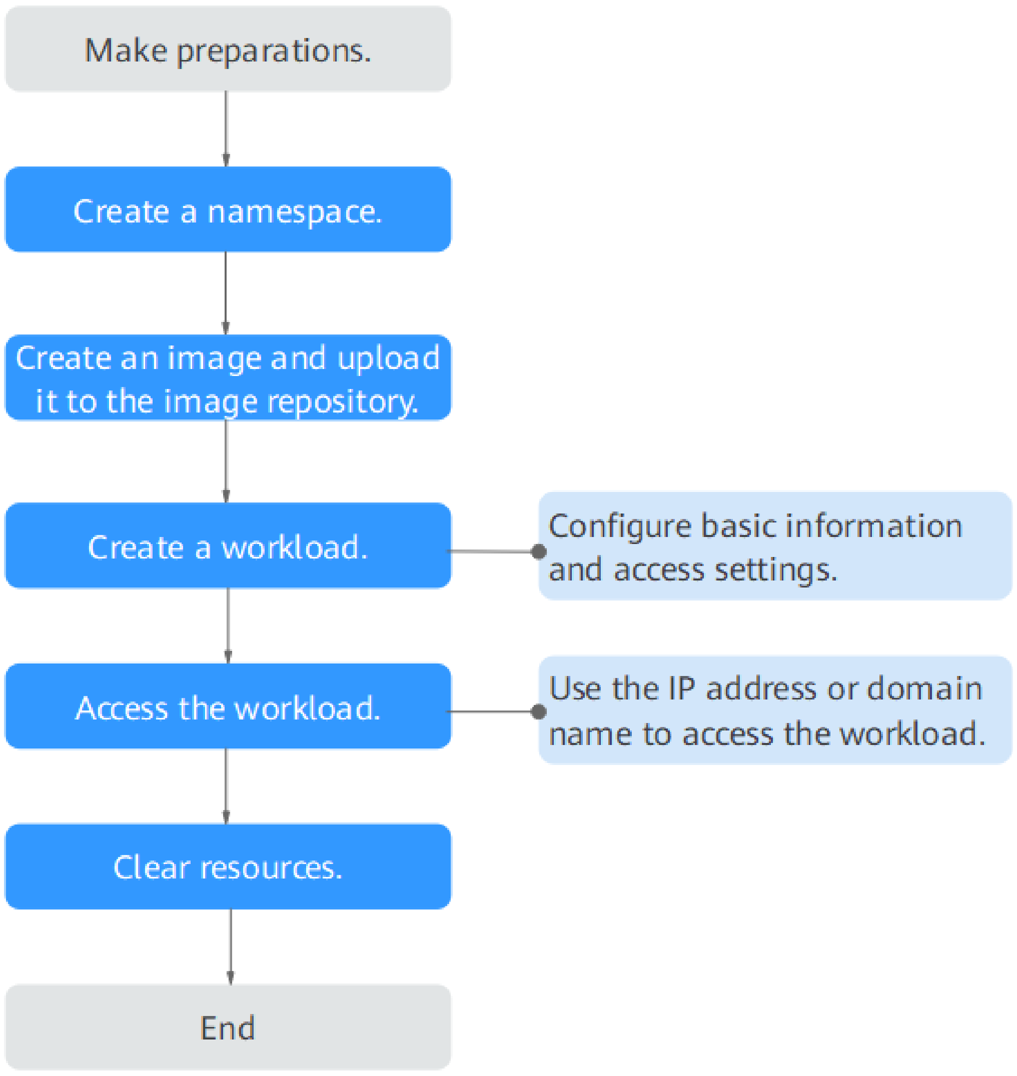
Contents

1 Introduction

This section uses a static web game application named **2048** as an example to illustrate how to use Cloud Container Instance (CCI).

Figure 1-1 shows the general process of using CCI.

Figure 1-1 Procedure for getting started with CCI


Make preparations.
Create a namespace.
Create an image and upload it to the image repository.
Create a workload.
Configure basic information and access settings.
Access the workload.
Use the IP address or domain name to access the workload.
Clear resources.
End

2 Preparations

Before using CCI, you need to make the following preparations:

- **Register with Huawei Cloud**
- **Topping Up Your Account**
- **Creating an IAM User**

Registering with Huawei Cloud

If you already have a Huawei Cloud account, skip this part. If you do not have a Huawei Cloud account, perform the following operations to create one:

1. Visit the **Huawei Cloud official website** and click **Register**.
2. On the **Register** page, create an account. For details, see **Registering with Huawei Cloud**.

   After the creation is complete, the system automatically redirects you to your personal information page.

Topping Up Your Account

Ensure that your account has sufficient balance.

Creating an IAM User

If you want to allow multiple users to manage your resources without sharing your password or keys, you can create users using IAM and grant permissions to the users. These users can use specified links and their own accounts to access Huawei Cloud and help you manage resources efficiently. You can also configure account security policies to ensure the security of these accounts.

Huawei Cloud accounts have the permissions to use CCI. However, IAM users created by Huawei Cloud accounts do not have the permissions. You need to manually assign the permissions to IAM users. For details, see **Permissions Management**.

3 Building an Image and Pushing It to the Image Repository

To deploy an existing application on CCI, build an image for the application and push it to the image repository. Then you can pull the image when creating a workload on CCI.

Installing the Container Engine

Before pushing an image, ensure that you have installed the container engine of version **1.11.2** or later.

Step 1 Create a Linux ECS with a public IP address. For details, see the **ECS documentation**.

For demonstration, you do not need to select high specifications for ECS and public IP address. For example, select **1 vCPUs | 2 GiB** for ECS specifications, **1 Mbit/s** for the IP bandwidth, and **CentOS 7.6** for the operating system.

NOTE

You can also install the container engine on other machines.

Step 2 Go to the ECS list and click **Remote Login** to log in to the ECS.

Step 3 Run the following command 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

Building an Image

The following describes how to use a Dockerfile and an Nginx image to build the **2048** image. Before building the image, you need to create a Dockerfile.

Step 1 Pull the Nginx image from the image repository as the base image.

```
docker pull nginx
```

Step 2 Download the **2048** static web application.

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

Step 3 Build a Dockerfile.

1. Run the following command:
   ```
   vi Dockerfile
   ```
2. Edit the Dockerfile.
   ```
   FROM nginx

   MAINTAINER Allen.Li@gmail.com
   COPY . /usr/share/nginx/html

   EXPOSE 80

   CMD ["nginx", "-g", "daemon off;"]
   ```
   – **nginx** indicates the base image. You can select the base image based on the type of the created application. For example, select a Java image as the base image to create a Java application.
   – **/usr/share/nginx/html** indicates the directory for the Nginx image to store the static web.
   – **80** indicates the container port.

   NOTE

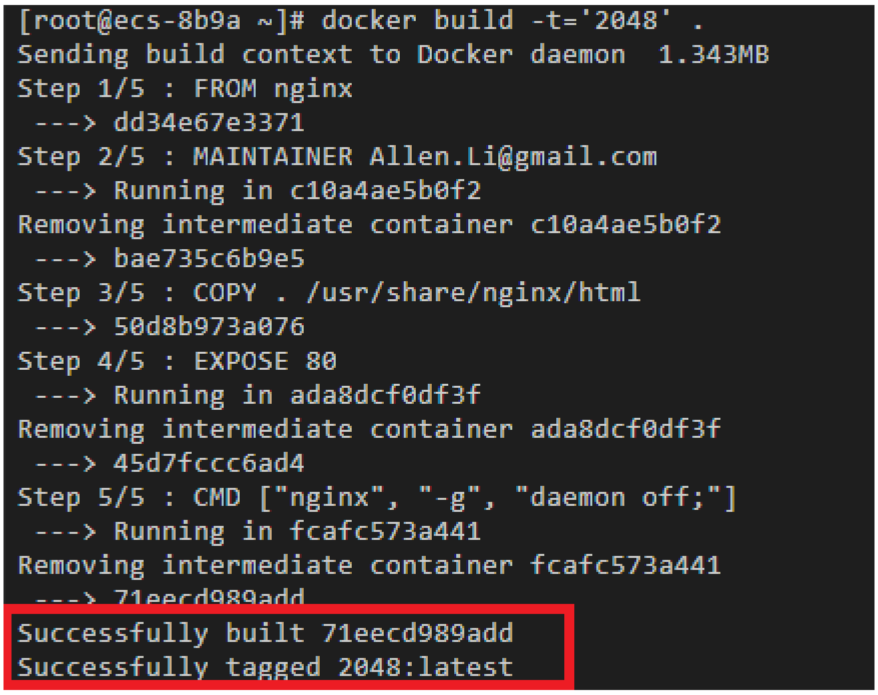
   For details about the Dockerfile content and format, see **Dockerfile reference**.

Step 4 Build the **2048** image.

1. Run the following command:
   ```
   docker build -t='2048' .
   ```

The following information will be displayed upon a successful image creation:

Figure 3-1 Successful image creation



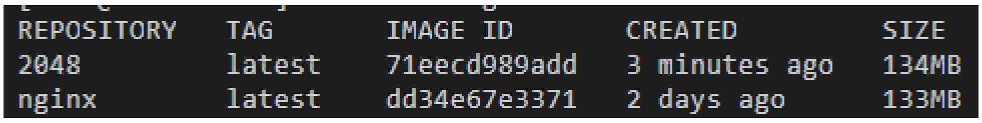
```
[root@ecs-8b9a ~]# docker build -t='2048' .
Sending build context to Docker daemon  1.343MB
Step 1/5 : FROM nginx
 ---> dd34e67e3371
Step 2/5 : MAINTAINER Allen.Li@gmail.com
 ---> Running in c10a4ae5b0f2
Removing intermediate container c10a4ae5b0f2
 ---> bae735c6b9e5
Step 3/5 : COPY . /usr/share/nginx/html
 ---> 50d8b973a076
Step 4/5 : EXPOSE 80
 ---> Running in ada8dcf0df3f
Removing intermediate container ada8dcf0df3f
 ---> 45d7fccc6ad4
Step 5/5 : CMD ["nginx", "-g", "daemon off;"]
 ---> Running in fcafc573a441
Removing intermediate container fcafc573a441
 ---> 71eecd989add
Successfully built 71eecd989add
Successfully tagged 2048:latest
```

2. Query the image.

```
docker images
```

If the following information is displayed, the image is successfully created.

Figure 3-2 Querying the image



```
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
2048          latest    71eecd989add   3 minutes ago    134MB
nginx         latest    dd34e67e3371   2 days ago       133MB
```

----End

Pushing the Image

Step 1 Access SWR.

1. Log in to the management console, choose **Service List** > **Application** > **SoftWare Repository for Container**.
2. In the navigation pane, choose **My Images** and then click **Upload Through Client**. On the page displayed, click **Generate a temporary login command** and click to copy the command.

   NOTE

   The validity period of the temporary login command is 24 hours. To obtain a long-term valid login command, see **Obtaining a Long-Term Valid Login Command**.
3. Run the login command on the machine where the container engine is installed.

   The message **login succeeded** will be displayed upon a successful login.

Step 2 Push the image.

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

   docker tag [*Image name:tag*] [*Image repository address*]/[*Organization name*]/[*Image name:tag*]

   Example:

   docker tag 2048:latest swr.cn-north-4.myhuaweicloud.com/cloud-develop/2048:latest

   In the preceding command:

   – **swr.cn-north-4.myhuaweicloud.com** indicates the address of the SWR image repository.
   – **cloud-develop** indicates the organization name of the image.
   – **2048:latest** indicates the image name and tag.
2. Run the following command to push the image to the image repository:

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

   Example:

   docker push swr.cn-north-4.myhuaweicloud.com/cloud-develop/2048:latest

   The following information will be returned upon a successful push:

```
6d6b9812c8ae: Pushed
695da0025de6: Pushed
fe4c16cbf7a4: Pushed
v1: digest: sha256:eb7e3bbd8e3040efa71d9c2cacfa12a8e39c6b2ccd15eac12bdc49e0b66cee63 size: 948
```

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

----End

4 Creating a Namespace

Step 1 Log in to the CCI console.

Step 2 Choose **Namespaces** in the navigation pane and then click **Create** in the **General-computing** namespace area.

Step 3 Enter a namespace name.

Select an existing VPC or create one. You must specify a CIDR block for the new VPC. The recommended CIDR blocks are 10.0.0.0/8-22, 172.16.0.0/12-22, and 192.168.0.0/16-22.

NOTICE

The VPC CIDR block and subnet CIDR block cannot be set to 10.247.0.0/16 because this CIDR block is reserved by CCI for workload access. If you use this CIDR block, IP address conflicts may occur, which may result in workload creation failures or service unavailability. If you do not need to access pods through workloads, you can allocate this CIDR block to a VPC.

Step 4 Configure a subnet CIDR block.

Ensure sufficient available IP addresses to create workloads.

Step 5 Click **Create**.

----End

5 Creating a Workload

Step 1 Log in to the CCI console.

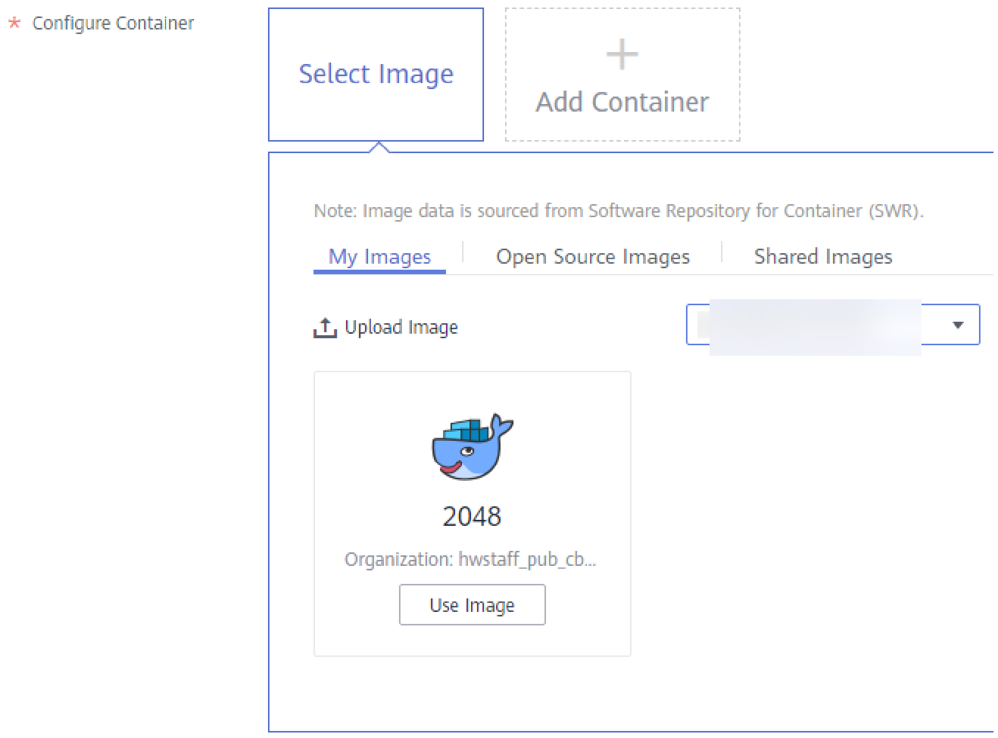
Step 2 In the navigation pane, choose **Workloads > Deployments**. On the page displayed on the right, click **Create from Image**.

Step 3 Specify basic information.

- **Workload Name**: Enter a workload name, for example, **deployment-2048**.
- **Namespace**: Select the namespace created in **Creating a Namespace**.
- **Pods**: Change the value to **1** in this example.
- **Pod Specifications**: Select the general-computing pod with 0.5-core CPU and 1GiB of memory.
- **Container Settings**

  On the **My Images** tab page, select the uploaded **2048** image.

Figure 5-1 Container settings



Step 4 Configure workload access settings.

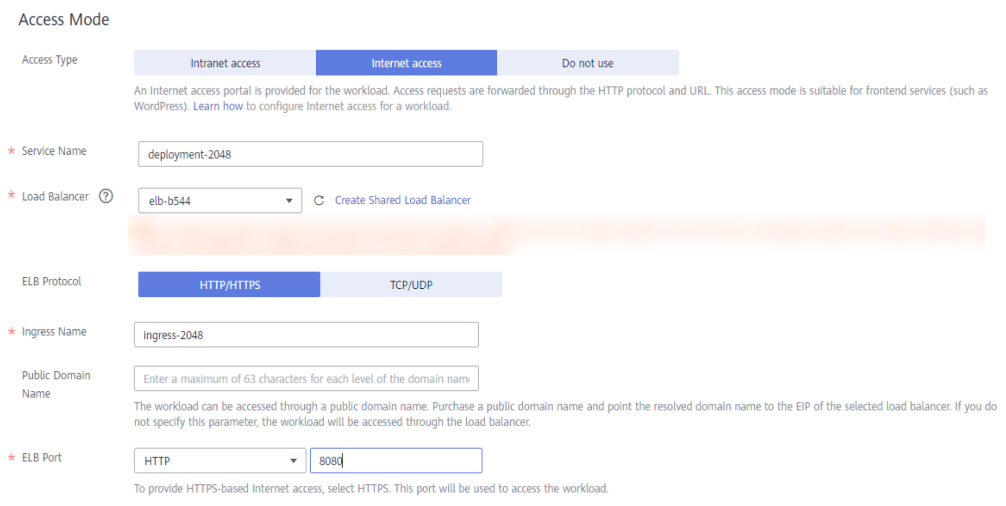
Three options are available:

- **Do not use**: No entry is provided to allow access from other workloads. This option is ideal for computing scenarios where communication with external systems is not required.
- **Intranet access**: There are two ways to allow the workload to be accessed by other workloads over the private network.
- **Internet access**: The workload is accessed from public networks through load balancers.

In this example, set the workload access option to **Internet access** to allow access to the **2048** workload using the EIP and port of the load balancer.

Set **Service Name** to **deployment-2048**, and select a load balancer. If no load balancer is available, click **Create Shared Load Balancer** to create one.

Set **Ingress Name** to **ingress-2048**, **ELB Protocol** to **HTTP/HTTPS**, and **ELB Port** to **HTTP 8080**.



Access Mode

Access Type: Intranet access | [✓ Internet access] | Do not use

An Internet access portal is provided for the workload. Access requests are forwarded through the HTTP protocol and URL. This access mode is suitable for frontend services (such as WordPress). Learn how to configure Internet access for a workload.

* Service Name: deployment-2048

* Load Balancer: elb-b544 — Create Shared Load Balancer

[redacted]

ELB Protocol: [✓ HTTP/HTTPS] | TCP/UDP

* Ingress Name: ingress-2048

Public Domain Name: Enter a maximum of 63 characters for each level of the domain name

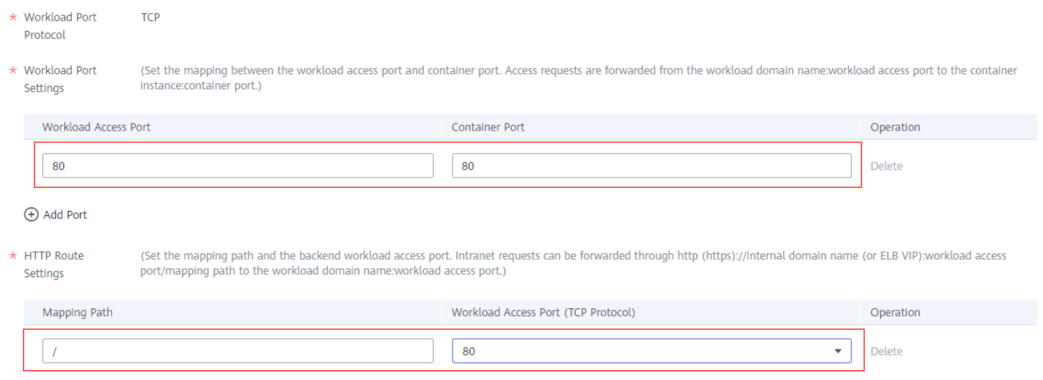
The workload can be accessed through a public domain name. Purchase a public domain name and point the resolved domain name to the EIP of the selected load balancer. If you do not specify this parameter, the workload will be accessed through the load balancer.

* ELB Port: HTTP 8080

To provide HTTPS-based Internet access, select HTTPS. This port will be used to access the workload.

Set **Workload Access Port** to **80** (or another port) and **Container Port** to **80**. The container port must be set to 80, which is the same port set for the **2048** image in the image repository.

Set **Mapping Path** to **/** and associate it with the workload access port so that you can access the **2048** workload using **Load balancer IP address:Port**.



* Workload Port Protocol: TCP

* Workload Port Settings: (Set the mapping between the workload access port and container port. Access requests are forwarded from the workload domain name:workload access port to the container instance:container port.)

Workload Access Port	Container Port	Operation
80	80	Delete

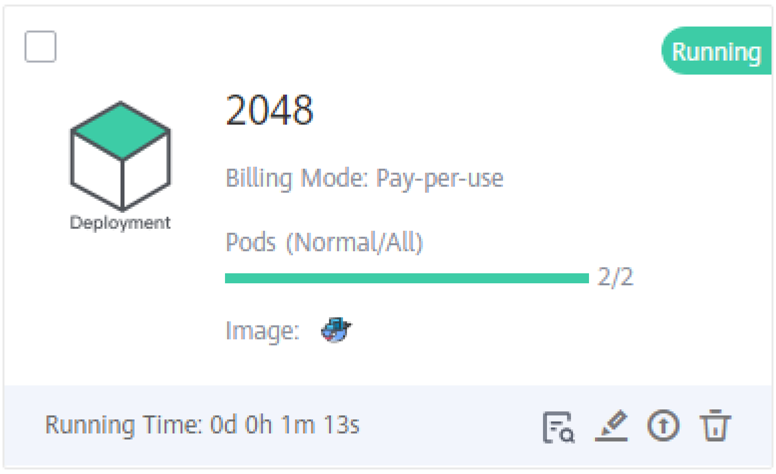
Add Port

* HTTP Route Settings: (Set the mapping path and the backend workload access port. Intranet requests can be forwarded through http (https)://internal domain name (or ELB VIP):workload access port/mapping path to the workload domain name:workload access port.)

Mapping Path	Workload Access Port (TCP Protocol)	Operation
/	80	Delete

Step 5 Click **Next: Configure Advanced Settings**. After you confirm the configuration, click **Submit**. Then click **Back to Deployment List**.

In the workload list, if the workload status is **Running**, the workload is created successfully.



2048

Billing Mode: Pay-per-use

Pods (Normal/All) 2/2

Image:

Running Time: 0d 0h 1m 13s

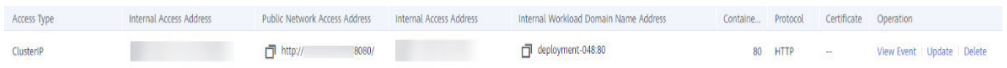
----End

6 Accessing the Workload

After the **2048** workload is created, you can access it using a browser.

Step 1 Click the workload name to enter its details page.

Step 2 Click ❐ in the **Public Network Access Address** column under **Access Settings** to copy the public network access address.



Access Type	Internal Access Address	Public Network Access Address	Internal Access Address	Internal Workload Domain Name Address	Containe...	Protocol	Certificate	Operation
ClusterIP	[redacted]	❐ http://[redacted]8080/	[redacted]	❐ deployment-048:80	80	HTTP	--	View Event | Update | Delete

Step 3 Paste the address in the browser address bar to access it.

----End

7 Clearing Resources

Step 1 In the navigation pane, choose **Workloads > Deployments**. In the **Deployments** list, click to delete the **2048** workload.

----End