

04_Deploy_Best Practice_Public Cloud-Copy

04_Deploy_Best Practice_Public Cloud-Copy

Issue 01
Date 2025-07-03



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2025. 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.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: <https://www.huaweicloud.com/intl/en-us/>

Contents

1 Deploying an Application on an Intranet Host Using a Proxy Host..... 1

2 Using Nginx for Gray Release..... 7

3 Using Kubernetes Nginx-Ingress for Gray Release.....36

4 HE2E DevOps Practice - Deploying an Application.....42

4.1 Overview..... 42

4.2 Deploying an Application on CCE..... 42

4.3 Deploying an Application on ECS..... 44

4.4 Releasing Resources..... 48

1 Deploying an Application on an Intranet Host Using a Proxy Host

Application Scenario

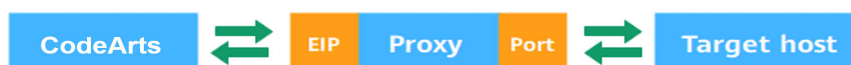
Deploy applications on the intranet through proxy hosts to effectively control intranet and extranet communication, enhance data security and network performance, and meet compliance requirements. This policy is widely used in various key scenarios, such as resource access control, secure communication between data centers, content cache acceleration, environment isolation, security audit, and sensitive data processing.

Solution Architecture

The Internet forward proxy function of Squid is used to specify the IP address and port of the target host on the proxy, enabling the target host to access the public network.

For more information about Squid, go to [Squid official website](#). The following procedure uses a Linux host as an example.

Figure 1-1 Principles of specifying a target host on a proxy



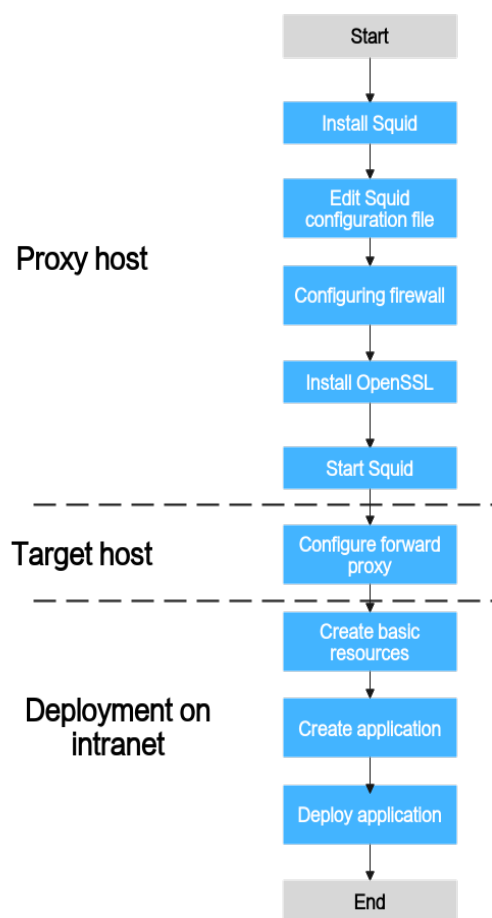
Prerequisites

- A host (**Proxy-B**) bound to a public IP address is available. If no proxy host is available, see [\(Optional\) Applying for an ECS](#).
- A host (**Host-A**) not bound to a public IP address is available.
- **Proxy-B** and **Host-A** can access each other through the intranet.

Process

This section describes how to deploy an application on an intranet host or server using a proxy host.

Figure 1-2 Process flowchart

**Step 1** Install Squid.

Access the command line tool of **Proxy-B** and run the following command:

```
yum install squid -y
```

If **Complete** is displayed, run the following command:

```
yum install iptables-services
```

Enter **Y**. If **Complete** is displayed, the installation is complete.

Step 2 Edit the Squid configuration file.

1. Access the command line tool of **Proxy-B** and run the following command:
`vim /etc/squid/squid.conf`

```
# Example rule allowing access from your local networks.
# Adapt to list your (internal) IP networks from where browsing
# should be allowed
acl localnet src 10.0.0.0/8          # RFC1918 possible internal network
acl localnet src 172.16.0.0/12       # RFC1918 possible internal network
acl localnet src 192.168.0.0/16      # RFC1918 possible internal network
acl localnet src 10.0.0.0/8          # RFC 4193 local private network range
acl localnet src ::0/0               # RFC 4291 link-local (directly plugged) machines
acl SSL_ports port 443
acl Safe_ports port 80               # http
```

2. Add the following command to the position marked in the red box in the preceding figure:

```
acl local src Internal IP address of the host/24
```

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

Step 3 Configure the firewall of **Proxy-B**.

Access the command line tool of **Proxy-B** and run the following commands in sequence:

```
systemctl stop firewalld.service
systemctl disable firewalld.service
yum install iptables-services iptables-devel -y
systemctl enable iptables.service
systemctl start iptables.service
iptables -I INPUT 1 -s Internal IP address of the host/24 -p tcp --dport 3128 -j ACCEPT
iptables -I INPUT 2 -p tcp --dport 3128 -j DROP
```

The IP address in the last but one line must be set to the internal IP address segment or IP address of **Host-A**. **3128** is the proxy port of Squid.

Step 4 Install OpenSSL.

Access the command line tool of **Proxy-B** and run the following command:

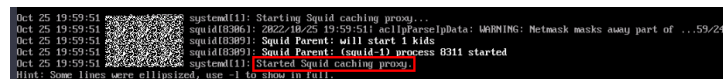
```
yum install openssl
```

If **Complete** is displayed, the installation is complete.

Step 5 Start Squid.

Access the command line tool of **Proxy-B** and run the following command:

```
systemctl start squid //Start Squid.
systemctl status squid //Check the status of Squid.
```



Step 6 Configure the forward proxy.

Access the command line tool of **Host-A** and run the following command:

```
echo "export http_proxy=http://Internal IP address of the proxy host:3128" >>/etc/profile
echo "export https_proxy=http://Internal IP address of the proxy host:3128" >>/etc/profile
echo "export http_proxy=http://Internal IP address of the proxy host:3128" >>~/.bashrc
echo "export https_proxy=http://Internal IP address of the proxy host:3128" >>~/.bashrc
echo "export http_proxy=http://Internal IP address of the proxy host:3128" >>~/.bash_profile
echo "export https_proxy=http://Internal IP address of the proxy host:3128" >>~/.bash_profile
source /etc/profile
source ~/.bashrc
source ~/.bash_profile
```

Step 7 Create basic resources.

1. Click **Homepage** to view all created projects, and then go to the target project.
2. Choose **Settings > General > Basic Resources** to access the **Host Clusters** page.
3. Click **Create Host Cluster**, enter the following information, and click **Save**.

Parameter	Mandatory	Description
Cluster Name	Yes	Enter a custom name.

Parameter	Mandatory	Description
OS	Yes	Select Linux based on the OS of the host to be added.
Host Connection Mode	Yes	Select Proxy .
Execution Resource Pool	Yes	A resource pool is a collection of physical environments where commands are executed during software package deployment. In this scenario, select official .
Description	No	Enter a description.

- Click **Add Host**, select **Adding IP** for **Add hosts by**, enter the following information, and click **OK**. The proxy host is created.

Table 1-1 Parameters of a Linux proxy host


Parameter	Mandatory	Description
Host Name	Yes	Enter a custom name, for example, Proxy-B .
IP	Yes	Enter the public IP address bound to Proxy-B .
OS	Yes	Keep the default value because it is the OS of your host cluster.
Authorization	Yes	In this scenario, the Password is used for authentication. Enter the username and password of Proxy-B .
SSH Port	Yes	Port 22 is recommended.

- Click **Add Host**, select **Adding IP** for **Add hosts by**, enter the following information, and click **OK**. The target host is created.

Table 1-2 Parameters of a Linux target host

Parameter	Mandatory	Description
Host Name	Yes	Enter a custom name, for example, Host-A .
Proxy Host	Yes	Select Proxy-B as the network proxy for the target host that cannot connect to the public network.


Parameter	Mandatory	Description
IP	Yes	Enter the private IP address of Host-A .
OS	Yes	Keep the default value because it is the OS of your host cluster.
Authorization	Yes	In this scenario, the Password is used for authentication. Enter the username and password of Host-A .
SSH Port	Yes	Port 22 is recommended.

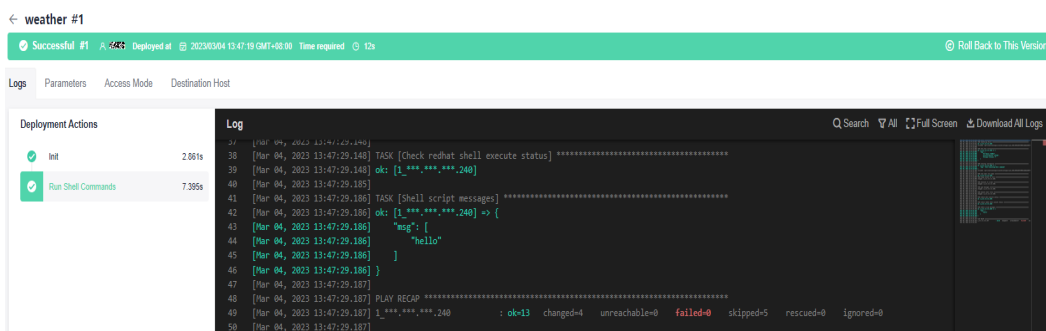
- Click  in the **Operation** column of a host to start the connectivity verification for the host. For details about connectivity verification, see [Host Management](#).

Step 8

- Log in to the CodeArts homepage and click the target project name to access the project.
- Choose **CICD > Deploy**.
- Click **Create Application**. On the **Basic Information** page, specify the **Name**, **Description**, and **Execution Resource Pool** as required.
- After editing the basic application information, click **Next**. On the **Select Template** page, select **Blank Template** and click **OK**.
- On the **Deployment Actions** tab page, find the action list on the right, click **Add** to add an action to the orchestration area on the left.
- On the **Environment Management** page, click **Create Environment**, enter the basic information, and click **Save**.
- Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the displayed dialog box, select the target host cluster and import **Proxy-B** and **Host-A** to the environment.

Step 9 Deploy the application. For details, see [Deploying an Application](#).

- Select the target application in the application list and click .
- After the deployment is complete, click the application name and click the target deployment record. The application status bar changes to green and the message **Successful** is displayed, indicating that the application is successfully deployed.



For more deployment problems, see [Creating an Application FAQs](#).

----End

2 Using Nginx for Gray Release

Application Scenario

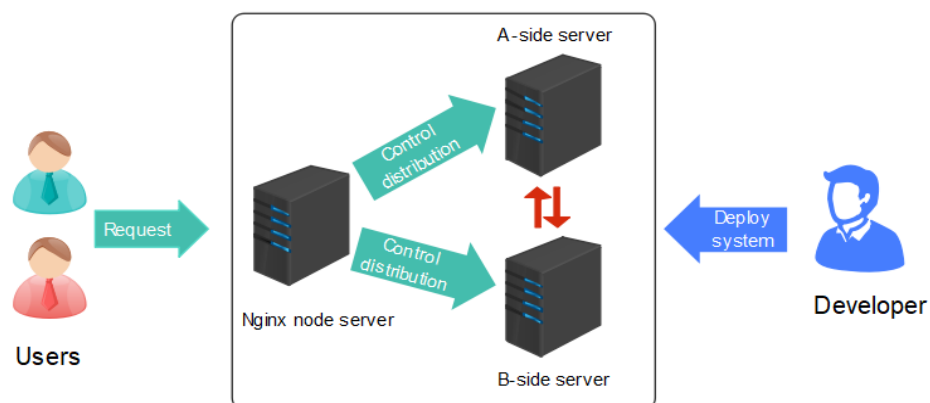
When you upgrade to a new system, services may be stopped or gray verification may fail. In this practice, you can use the Nginx load balancing mechanism for smooth system upgrade without affecting service running.

Solution Architecture

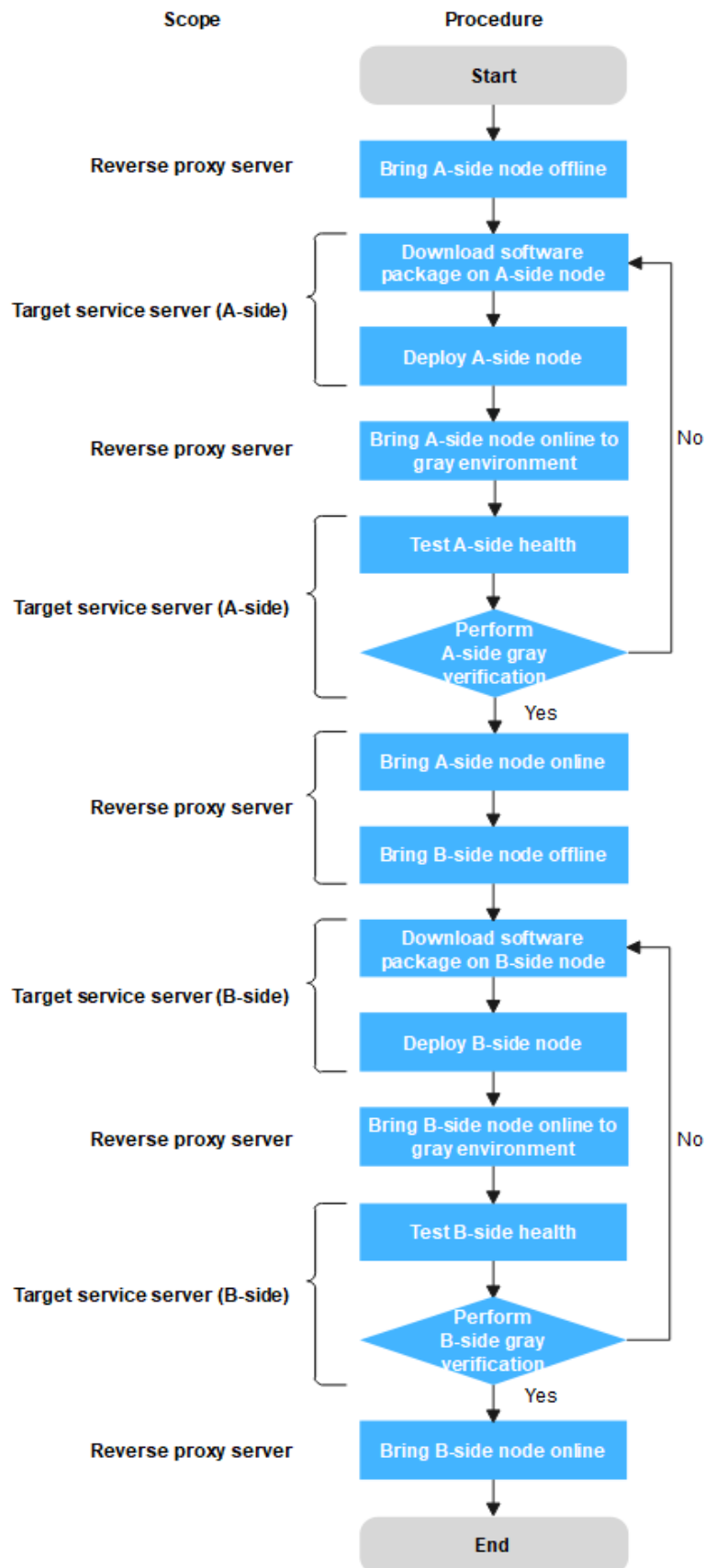
When upgrading the system using blue-green deployment, perform the following operations:

1. Bring server A (blue environment, carrying the current workloads) offline and distribute all access traffic to server B. Upgrade server A.
2. After the server A is upgraded, set it as the gray test environment and let testers perform gray verification on it.
3. After the gray verification is complete and the functions become normal, release server A (green environment) and move all traffic from B to it. Now, the blue-green deployment is complete.
4. If an emergency occurs on the server A during service running, perform a blue-green switchover to quickly restore services.

Figure 2-1 Gray release scheme



If you use canary release, upgrade server B by referring to operations described in blue-green deployment. Complete the gray test on it and release it to complete the gray release.



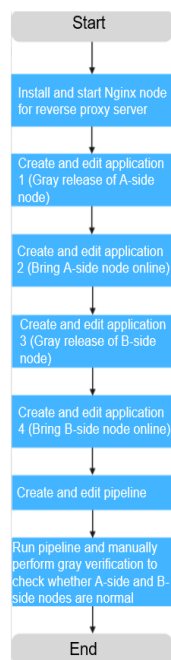
Prerequisites

- A project is available. If there is no project, [Project Managers: Creating and Configuring Projects](#) first.
- You have the permissions to create applications. For details, see [Editing Permissions](#).
- Target service servers **A_test** and **B_test** are available and application services are running on the servers.
- A reverse proxy server **Gray_release** is available.
- A gray verification host is available. This host represents a gray tester.
- Ensure that servers can communicate with each other. To ensure it, you can add all servers to the same virtual private cloud (VPC).

Process

Based on the Nginx load balancing mechanism, this practice implements blue-green release and gray release in host deployment scenarios. For more information about Nginx, see [Nginx official website](#).

Figure 2-2 Process flowchart



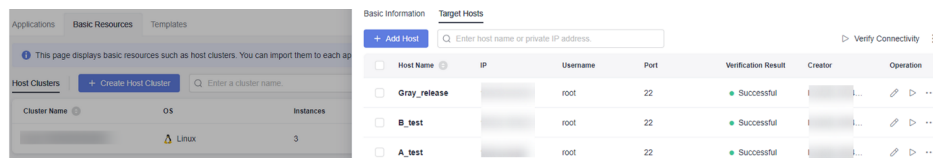
Step 1 (Optional) Install and start an Nginx node for the reverse proxy server.

If the Nginx node has been installed and started on your reverse proxy server, skip this step.

1. Create basic resources.
 - a. Go to the CodeArts homepage and click the target project name to access the project.
 - b. Choose **Settings > General > Basic Resources** to access the **Host Clusters** page.

Alternatively, choose **CICD > Deploy > Basic Resources** to access the **Host Clusters** page.

- c. Click **Create Host Cluster**, enter basic information such as the **Cluster Name**, **OS**, **Proxy**, **Execution Resource Pool**, and **Description**, and click **Save**.
- d. Create three hosts (**A_test**, **B_test**, and **Gray_release**) and verify their connectivity. Specifically, click **Add Host**, select **Adding IP**, enter the host name, **IP**, **Username**, **Password** or **Key**, and **SSH Port**, and click **OK**. For details about connectivity verification, see [Hosts](#).




2. Create an application.
 - a. Choose **CICD > Deploy**.
 - b. Click **Create Application**. On the **Basic Information** page, specify the **Name**, **Description**, and **Execution Resource Pool** as required.
 - c. After editing the basic application information, click **Next**. The deployment template selection page is displayed.
 - d. Select **Blank Template** and click **OK**. The **Deployment Actions** tab page is displayed.
3. Edit the application.
 - a. Switch to the **Environment Management** tab page. Create an environment and edit it.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to each server, and enter the description information.
 - Click **Save**. The environment is created.
 - Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the displayed dialog box, select the target host cluster and click  in the **Operation** column of the target host to import the host to the environment.
 - b. Switch to the **Deployment Actions** tab page. Add the following actions and edit them.
 - Add the **Install Nginx** action and modify the parameters in the following table (Linux is used as an example).

Table 2-1 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_.,:./()
Environment	Yes	Select Reverse_proxy_server_group .
Nginx Version	Yes	Select a target version. Example: nginx-1.14.2 .
Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .

- Add the **Start/Stop Nginx** action and modify the parameters in the following table (Linux is used as an example).

Table 2-2 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_.,:./()
Environment	Yes	Select Reverse_proxy_server_group .
Operation	Yes	Select Start Nginx .
Nginx Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .

- c. Click **Save & Deploy** to deploy the application.

4. Deploy the application.

If the application status bar turns green and displays **Successful**, the application is deployed successfully.

If the application status bar turns red and displays **Failed**, the application fails to be deployed. In this case, click **View Solution**.


For more deployment problems, see [Creating an Application](#).



Step 2 Create and edit application 1 (Gray release of A-side node).

1. Create an application.

- Choose **CICD > Deploy**.
- Click **Create Application**. On the **Basic Information** page, specify the **Name**, **Description**, and **Execution Resource Pool** as required.
- After editing the basic application information, click **Next**. The deployment template selection page is displayed.
- Select the **Deploy a General Application** template and click **OK**.

2. Edit the application.

- Switch to the **Environment Management** tab page. Create an environment and edit it.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to each server, and enter the description information.
 - Click **Save**. The environment is created.
 - Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the displayed dialog box, select the target host cluster and click  in the **Operation** column of the target host to import the host to the environment.
 - Repeat the preceding steps to create **Target service server group_A-side node** and add the **A_test** server.
- Switch to the **Parameters** tab page. Add the following parameters:

Custom		Preddefined		Q. Enter a name or default value.					
	Name	Type	Default Value	Private Parameter	Runtime Settings	Description	Operation		
≡	app_name	String	test	<input type="checkbox"/>	<input type="checkbox"/>	Application name	+ 		
≡	service_port	String	3000	<input type="checkbox"/>	<input type="checkbox"/>	Application port	+ 		
+ Create Parameter									

- Switch to the **Deployment Actions** tab page. Add the following actions and edit them.
 - Add the **Start/Stop Nginx** action and modify the parameters in the following table (Linux is used as an example).

Table 2-3 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;;./() Enter a name such as Bring_A-side_node_offline .
Environment	Yes	Select a target environment. Example: Reverse_proxy_server_group .
Operation	Yes	Specify an operation type. Example: Reload configuration file .
Nginx Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	No	Select this option for this example.
Nginx Configuration File Path	Yes	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	No	Enter the target path for backing up the original Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx_bak.conf .
Configuration File Content	Yes	Enter content of the new configuration file. See Example code to bring A-side node offline in the appendix.

- Edit the **Download Software Package** action and modify the parameters in the following table (Linux is used as an example).

Table 2-4 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;;./() Example: Download_software_package_on_A-side_node.
Source	Yes	Select a source. Example: Artifact.
Environment	Yes	Select a target environment. Example: Target service server group_A-side node.
Software Package	Yes	Select a software package to be deployed in CodeArts Artifact.
Download Path	Yes	Enter the path for downloading the software package to the target host. Example: /usr/local/.

- Edit the **Run Shell Commands** action and modify the parameters in the following table (Linux is used as an example).

Table 2-5 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;;./() Example: Deploy A-side node.
Environment	Yes	Select a target environment. Example: Target service server group_A-side node.

Parameter	Mandatory	Description
Shell Commands	Yes	Enter the commands to be executed. Example: See Deployment node in the appendix.

- Add the **Start/Stop Nginx** action and modify the parameters in the following table (Linux is used as an example).

Table 2-6 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_.,:./() Example: Bring A-side node online to gray environment.
Environment	Yes	Select a target environment. Example: Reverse_proxy_server_group.
Operation	Yes	Specify an operation type. Example: Reload configuration file.
Nginx Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx.
Modify configuration file before execution	No	Select this option for this example.
Nginx Configuration File Path	Yes	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf.
Configuration File Backup Path	No	Enter the target path for backing up the original Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx_bak.conf.

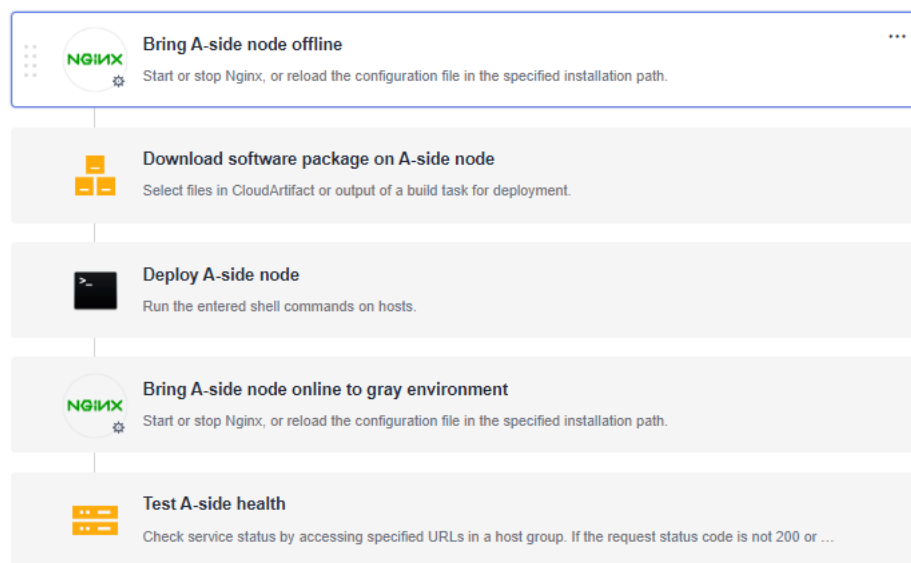
Parameter	Mandatory	Description
Configuration File Content	Yes	Enter content of the new configuration file. See Example code to bring A-side node online to the gray environment in the appendix.

- Edit the **Health Test via URLs** action and modify the parameters in the following figure (Linux is used as an example):

Table 2-7 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;;./() Enter a name such as Test_A-side_health .
Environment	Yes	Select a target environment. Example: Target service server group_A-side node .
Retries	Yes	If a service does not start up when the health test reaches the maximum retries, the service fails this test. Example: 1
Interval (s)	Yes	Interval between two retries, in seconds. Example: 60
Test Path	Yes	Used for the health test via URLs. You can add multiple URLs. Example: http://127.0.0.1:3000 (IP address and port number of the application service)

3. Click **Save**. The application is created.



Step 3 Create and edit application 2 (Bring A-side node online).


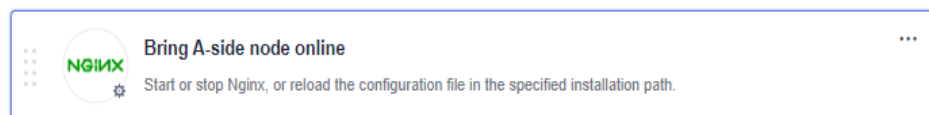
1. Create an application.
 - a. Click **Create Application**. On the **Basic Information** page, specify the **Name**, **Description**, and **Execution Resource Pool** as required.
 - b. After editing the basic application information, click **Next**. The deployment template selection page is displayed.
 - c. Select **Blank Template** and click **OK**.
2. Edit the application.
 - a. Switch to the **Environment Management** tab page. Create an environment and edit it.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to each server, and enter the description information.
 - Click **Save**. The environment is created.
 - Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the displayed dialog box, select the target host cluster and click  in the **Operation** column of the target host to import the host to the environment.
 - b. Switch to the **Deployment Actions** tab page. Add the following actions and edit them.
Add the **Start/Stop Nginx** action and modify the parameters in the following table (Linux is used as an example).

Table 2-8 Parameter description


Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;/() Example: Bring_A-side_node_online .
Environment	Yes	Select a target environment. Example: Reverse_proxy_server_group .
Operation	Yes	Specify an operation type. Example: Reload configuration file .
Nginx Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	No	Select this option for this example.
Nginx Configuration File Path	Yes	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	No	Enter the target path for backing up the original Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx_bak.conf .
Configuration File Content	Yes	Enter content of the new configuration file. See Example code to bring a node online in the appendix.

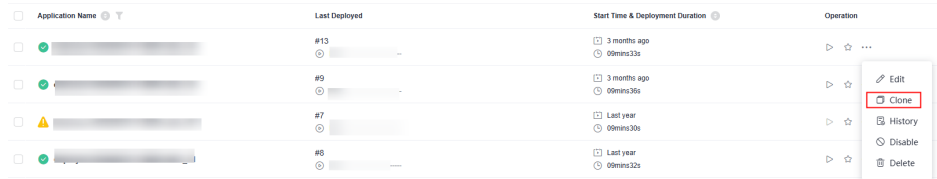
3. Click **Save**. The application is created.



Step 4 Clone and edit application 1. Create application 3 (gray release of B-side node).

1. Clone an application.

Click  and choose **Clone**.



2. Edit the application.


- a. Switch to the **Environment Management** tab page. Create an environment and edit it.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to each server, and enter the description information.
 - Click **Save**. The environment is created.
 - Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the displayed dialog box, select the target host cluster and click  in the **Operation** column of the target host to import the host to the environment.
 - Repeat the preceding steps to create **Target service server group_B-side node** and add the **B_test** server.
- b. Switch to the **Deployment Actions** tab page. Add the following actions and edit them.
 - Edit the **Bring A-side node offline** action and modify the parameters as follows (Linux is used as an example):

Table 2-9 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_.,:;/() Example: Bring_B-side_node_offline .
Environment	Yes	Select a target environment. Example: Reverse_proxy_server_group .
Operation	Yes	Specify an operation type. Example: Reload configuration file .
Nginx Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .

Parameter	Mandatory	Description
Modify configuration file before execution	No	Select this option for this example.
Nginx Configuration File Path	Yes	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	No	Enter the target path for backing up the original Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx_bak.conf .
Configuration File Content	Yes	Enter content of the new configuration file. See Example code to bring B-side node offline in the appendix.

- Edit the **Download software package on A-side node** action and change the parameter values to those listed in the following table (Linux is used as an example).

Table 2-10 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_,:./() Example: Download_software_package_on_B-side_node .
Source	Yes	Select a source. Example: Artifact .
Environment	Yes	Select a target environment. Example: B_group .
Software Package	Yes	Select a software package to be deployed in CodeArts Artifact.

Parameter	Mandatory	Description
Download Path	Yes	Enter the path for downloading the software package to the target host. Example: /usr/local/ .

- Edit the **Deploy A-side node** action and modify the parameters as follows (Linux is used as an example):

Table 2-11 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: <code>-_./();</code> Example: Deploy_B-side_node .
Environment	Yes	Select a target environment. Example: B_group .
Shell Commands	Yes	Enter the commands to be executed. Example: See Deployment node in the appendix.

- Edit the **Bring A-side node online to gray environment** action and modify the parameters as follows (Linux is used as an example):

Table 2-12 Parameter description

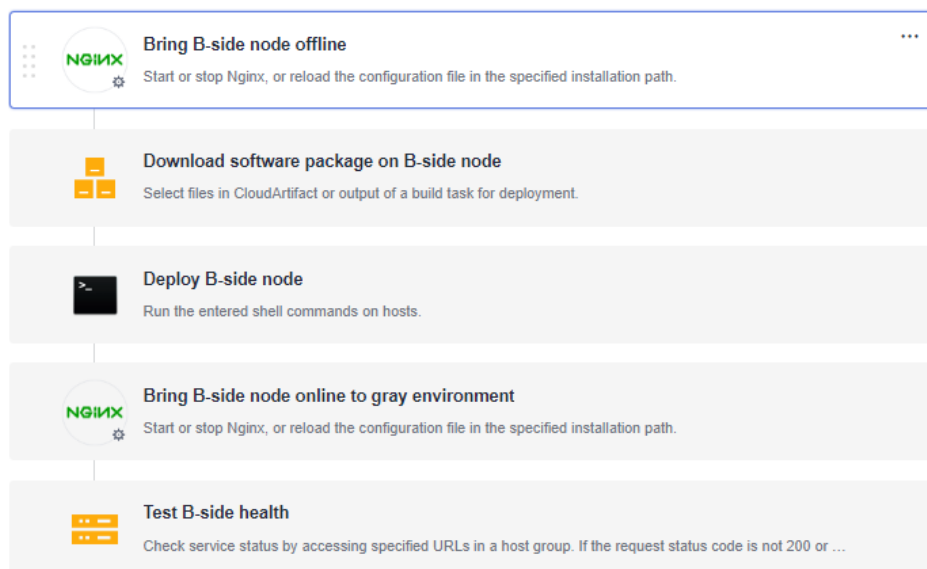
Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;;./() Example: Bring_B-side_node_online_to_gray_environment .
Environment	Yes	Select a target environment. Example: Reverse_proxy_server_group .
Operation	Yes	Specify an operation type. Example: Reload configuration file .
Nginx Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	No	Select this option for this example.
Nginx Configuration File Path	Yes	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	No	Enter the target path for backing up the original Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx_bak.conf .
Configuration File Content	Yes	Enter content of the new configuration file. See Example code to bring B-side node online to the gray environment in the appendix.

- Edit the **Test A-side health** action and modify the parameters as follows (Linux is used as an example):

Table 2-13 Parameter description


Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;;./() Enter a name such as Test_B-side_health .
Environment	Yes	Select a target environment. Example: B_group .
Retries	Yes	If a service does not start up when the health test reaches the maximum retries, the service fails this test. Example: 1
Interval (s)	Yes	Interval between two retries, in seconds. Example: 60
Test Path	Yes	Used for the health test via URLs. You can add multiple URLs. Example: http://127.0.0.1:3000 (IP address and port number of the application service)

3. Click **Save**. The application is created.




Step 5 Clone and edit application 2. Create application 4 (Bring B-side node online).

1. Clone an application.

Click  and choose **Clone**.

2. Edit the application.

- a. Switch to the **Environment Management** tab page. Create an environment and edit it.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to each server, and enter the description information.
 - Click **Save**. The environment is created.
 - Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the displayed dialog box, select the target host cluster and click  in the **Operation** column of the target host to import the host to the environment.
- b. Switch to the **Deployment Actions** tab page. Add the following actions and edit them.

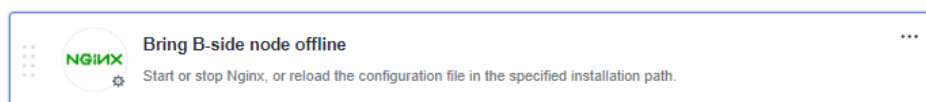
Edit the **Bring A-side node online** action and modify the parameters as follows (Linux is used as an example):

Table 2-14 Parameter description

Parameter	Mandatory	Description
Action Name	Yes	Enter a customized action name displayed in the deployment actions. Enter 1 to 128 characters. Do not start or end with a space. Use letters, digits, spaces, and these special characters: -_;;./() Example: Bring_B-side_node_online .
Environment	Yes	Select a target environment. Example: Reverse_proxy_server_group .
Operation	Yes	Specify an operation type. Example: Reload configuration file .
Nginx Installation Path	Yes	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	No	Select this option for this example.
Nginx Configuration File Path	Yes	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .

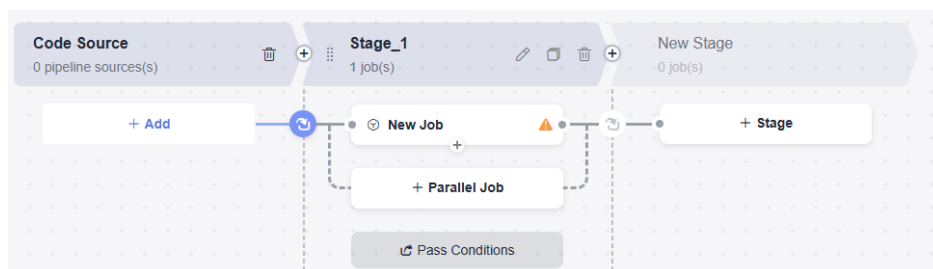
Parameter	Mandatory	Description
Configuration File Backup Path	No	Enter the target path for backing up the original Nginx configuration file on the target host. Example: <code>/usr/local/nginx/conf/nginx_bak.conf</code> .
Configuration File Content	Yes	Enter content of the new configuration file. See Example code to bring a node online in the appendix.

- Click **Save**. The application is created.



Step 6 Create and edit a pipeline.

- Create a pipeline.
 - Choose **CICD > Pipeline**.
 - Click **Create Pipeline**, select a **Project**, enter a **Name**, set **Pipeline Source** to **None**, and click **Next**.
 - Select **Blank Template** and click **OK**.




- Edit job 1 (**Gray release of A-side node**) in the pipeline stage.
 - Click . In the displayed dialog box, set the parameters as follows and click **Confirm**.

Table 2-15 Parameter description

Parameter	Mandatory	Description
Stage Name	Yes	Example: Gray_release_of A-side_node .

Parameter	M a n d a t o r y	Description
Always Run	Yes	Select No .




- Click . In the displayed dialog box, set **Entry Type** to **Automatic** and click **OK**.
- Click **New Job**, click the **Deploy** tab, select **Deploy**, and click **Add**. In the displayed dialog box, set the parameters as follows and click **OK**.

Table 2-16 Parameter description

Parameter	M a n d a t o r y	Description
Name	Yes	Example: Gray_release_of A-side_node .
Select Task	Yes	Select Gray_release_of A-side_node .
Build Task	No	Leave it blank for this example.




3. Create and edit job 2 (**Bring_A-side_node_online**) in the pipeline stage.
 - Click  and . In the displayed dialog box, set the parameters as follows and click **Confirm**.

Table 2-17 Parameter description

Parameter	M a n d a t o r y	Description
Name	Yes	Example: Bring_A-side_node_online .
Always Run	Yes	Select No .




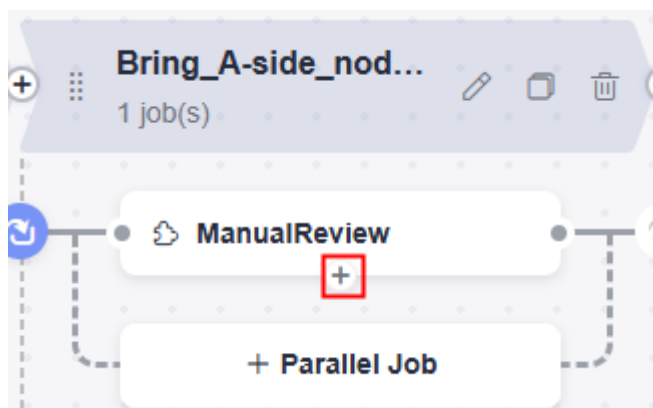
- Click . In the displayed dialog box, set **Entry Type** to **Automatic** and click **OK**.

- Click **New Job**. In the window that is displayed, click the **Normal** tab, select **Manual Review** and click **Add**, set the parameters as follows, and click **OK**.

Table 2-18 Parameter description

Parameter	Mandatory	Description
Name	Yes	Example: Gray_release_of A-side_node .
Reviewer	Yes	Select the service verification personnel.
Review Mode	Yes	Select Review by all .
Timeout Processing	Yes	Select Review failed and pipeline terminated .
Review Duration	Yes	Example: 4 hours.
Description	No	Enter the review description.

- Click , click the **Deploy** tab, select **Deploy**, and click **Add**. In the displayed dialog box, set the parameters as follows and click **OK**.

**Table 2-19** Parameter description

Parameter	Mandatory	Description
Name	Yes	Example: Bring_A-side_node_online .
Select Task	Yes	Select Bring_A-side_node_online .

Parameter	M a n d a t o r y	Description
Build Task	No	Leave it blank for this example.



4. Edit job 3 (**Gray_release_of_B-side_node**) in the pipeline stage.
- Click  and . In the displayed dialog box, set the parameters as follows and click **Confirm**.

Table 2-20 Parameter description

Parameter	M a n d a t o r y	Description
Name	Yes	Example: Gray_release_of_B-side_node .
Always Run	Yes	Select No .


- Click . In the displayed dialog box, set **Entry Type** to **Automatic** and click **OK**.
- Click **New Job**, click the **Deploy** tab, select **Deploy**, and click **Add**. In the displayed dialog box, set the parameters as follows and click **OK**.

Table 2-21 Parameter description

Parameter	M a n d a t o r y	Description
Name	Yes	Example: Gray_release_of_B-side_node .
Select Task	Yes	Select Gray_release_of_B-side_node .
Build Task	No	Leave it blank for this example.



5. Create and edit job 4 (**Bring_B-side_node_online**) in the pipeline stage.
- Click  and . In the displayed dialog box, set the parameters as follows and click **Confirm**.

Table 2-22 Parameter description

Parameter	Mandatory	Description
Name	Yes	Example: Bring_B-side_node_online .
Always Run	Yes	Select No .





- Click . In the displayed dialog box, set **Entry Type** to **Automatic** and click **OK**.
- Click **New Job**. In the window that is displayed, click the **Normal** tab, select **Manual Review** and click **Add**, set the parameters as follows, and click **OK**.

Table 2-23 Parameter description

Parameter	Mandatory	Description
Name	Yes	Example: Gray_verification_of_B-side_node .
Reviewer	Yes	Select the service verification personnel.
Review Mode	Yes	Select Review by all .
Timeout Processing	Yes	Select Review failed and pipeline terminated .
Review Duration	Yes	Example: 4 hours.
Description	No	Enter the review description.

- Click , click the **Deploy** tab, select **Deploy**, and click **Add**. In the displayed dialog box, set the parameters as follows and click **OK**.

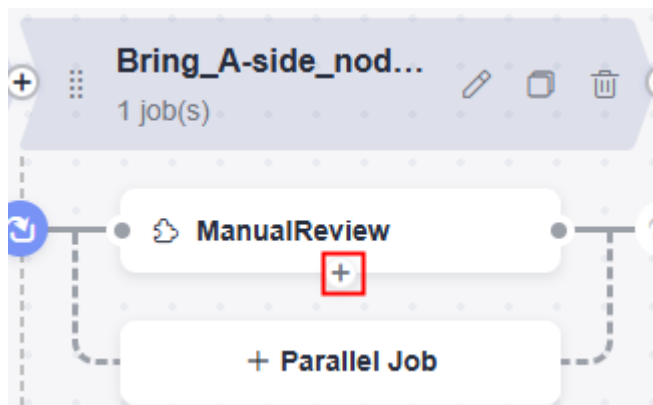
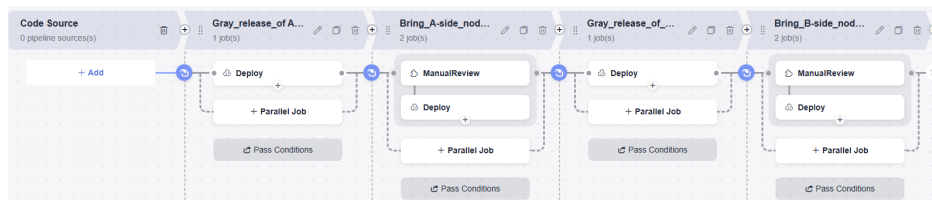


Table 2-24 Parameter description

Parameter	Mandatory	Description
Name	Yes	Example: Bring_B-side_node_online .
Select Task	Yes	Select Bring_B-side_node_online .
Build Task	No	Leave it blank for this example.

- After the preceding operations are complete, click **Save and Run** to run pipeline jobs.



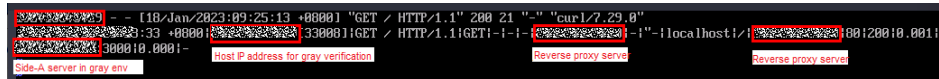
Step 7 Run the pipeline and manually perform gray verification to check whether A-side and B-side nodes are normal.

When CodeArts Pipeline is executed to bring node A or B online, pipeline execution is suspended. Gray users need to manually verify whether the servers on node A or B in the gray environment are working. Continue to run the pipeline if the servers are working.

Gray users can run the **curl** command to check whether the gray environment is normal.

```
curl http://IP address of the reverse proxy server:Nginx port
```

To check whether the gray user has accessed the target gray environment server, log in to the **reverse proxy server** and go to the path **logs/access.log** to check logs.



----End

Appendixes

- **Example code to bring A-side node offline**

```
worker_processes 1;
events {
    worker_connections 1024;
}
http {
    include mime.types;
    default_type application/octet-stream;
    log_format main '$time_local $remote_addr [$remote_port] $request $request_method $content_length';
    log_format '$content_type $http_referer $host $http_x_forwarded_for'
        '$http_true_client_ip $server_name $request_uri $server_addr $server_port'
        '$status $request_time $upstream_addr $upstream_response_time $cookie_domain_tag';
    access_log logs/access.log main; #Access log: storage path and log level
    error_log logs/error.log; #Error log: storage path
    sendfile on;
    keepalive_timeout 65;
    upstream portal {
        #Enter the IP address and application service port number of host A.
        #server X.X.X.X; #Bring node A offline.
        #Enter the IP address and application service port number of host B.
        server X.X.X.X;
    }
    upstream portal_test {
        #Enter the IP address and application service port number of host A.
        server X.X.X.X;
        #Enter the IP address and application service port number of host B.
        server X.X.X.X;
    }

    server {
        listen XXX; #Enter the Nginx port number.
        server_name localhost;

        location / {
            set $backend portal;
            set $test portal_test;
            #Enter the IP address of the gray verification host.
            #if ( $remote_addr ~* "X.X.X.X" ) {
            # set $backend $test;
            #}
            proxy_pass https://$backend;
        }
        error_page 500 502 503 504 /50x.html;
        location = /50x.html {
            root html;
        }
    }
}
```

- **Deployment node**

```
#Obtain the application process ID.
pid=`ps -ef | grep app_name | grep -v grep | awk '{print $2}'`
if [ -z "$pid" ];
then
    echo "[app_name pid is not exist.]"
else
    echo "app_name pid: $pid "
    #End the process.
    kill -15 $pid
fi
```

```
#Restart the application. You can run the deployment script or command to start the application.  
#Method 1: Run the deployment script to start the application.  
# sh startup.sh  
#Method 2: Run the command to start the application. nohup is recommended for backend startup.  
# nohup java -jar /usr/local/app/SpringbootDemo.jar &
```

- **Example code to bring A-side node online to the gray environment**

```
worker_processes 1;  
events {  
    worker_connections 1024;  
}  
http {  
    include mime.types;  
    default_type application/octet-stream;  
    log_format main '$time_local$remote_addr[$remote_port]$request$request_method|  
$content_length|  
    '$content_type|$http_referer|$host|$http_x_forwarded_for|  
    '$http_true_client_ip|$server_name|$request_uri|$server_addr|$server_port|  
    '$status|$request_time|$upstream_addr|$upstream_response_time|$cookie_domain_tag';  
    access_log logs/access.log main; #Access log: storage path and log level  
    error_log logs/error.log; #Error log: storage path  
    sendfile on;  
    keepalive_timeout 65;  
    upstream portal {  
        #Enter the IP address and application service port number of host A.  
        #server X.X.X.X:X; #Bring node A offline.  
        #Enter the IP address and application service port number of host B.  
        server X.X.X.X:X;  
    }  
    upstream portal_test {  
        #Enter the IP address and application service port number of host A.  
        server X.X.X.X:X; #Gray release of node A  
        #Enter the IP address and application service port number of host B.  
        #server X.X.X.X:X;  
    }  
}  
  
server {  
    listen XXX;#Enter the Nginx port number.  
    server_name localhost;  
  
    location / {  
        set $backend portal;  
        set $test portal_test;  
        #Enter the IP address of the gray verification host.  
        if ( $remote_addr ~* "X.X.X.X" ) {  
            set $backend $test;  
        }  
        proxy_pass https://$backend;  
    }  
    error_page 500 502 503 504 /50x.html;  
    location = /50x.html {  
        root html;  
    }  
}
```

- **Example code to bring B-side node offline**

```
worker_processes 1;  
events {  
    worker_connections 1024;  
}  
http {  
    include mime.types;  
    default_type application/octet-stream;  
    log_format main '$time_local$remote_addr[$remote_port]$request$request_method|  
$content_length|  
    '$content_type|$http_referer|$host|$http_x_forwarded_for|  
    '$http_true_client_ip|$server_name|$request_uri|$server_addr|$server_port|  
    '$status|$request_time|$upstream_addr|$upstream_response_time|$cookie_domain_tag';  
    access_log logs/access.log main; #Access log: storage path and log level
```

```
error_log logs/error.log; #Error log: storage path
sendfile on;
keepalive_timeout 65;
upstream portal {
    #Enter the IP address and application service port number of host A.
    server X.X.X.X:X;
    #Enter the IP address and application service port number of host B.
    #server X.X.X.X:X; #Bring node B offline.
}
upstream portal_test {
    #Enter the IP address and application service port number of host A.
    server X.X.X.X:X;
    #Enter the IP address and application service port number of host B.
    server X.X.X.X:X;
}

server {
    listen XXX;#Enter the Nginx port number.
    server_name localhost;

    location / {
        set $backend portal;
        set $test portal_test;
        #Enter the IP address of the gray verification host.
        #if ( $remote_addr ~* "X.X.X.X") {
        # set $backend $test;
        #}
        proxy_pass https://$backend;
    }
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root html;
    }
}
```

- **Example code to bring B-side node online to the gray environment**

```
worker_processes 1;
events {
    worker_connections 1024;
}
http {
    include mime.types;
    default_type application/octet-stream;
    log_format main '$time_local$remote_addr[$remote_port]$request$request_method|
$content_length|
$content_type|$http_referer|$host|$http_x_forwarded_for|
$http_true_client_ip|$server_name|$request_uri|$server_addr|$server_port|
$status|$request_time|$upstream_addr|$upstream_response_time|$cookie_domain_tag';
    access_log logs/access.log main; #Access log: storage path and log level
    error_log logs/error.log; #Error log: storage path
    sendfile on;
    keepalive_timeout 65;
    upstream portal {
        #Enter the IP address and application service port number of host A.
        server X.X.X.X:X;
        #Enter the IP address and application service port number of host B.
        #server X.X.X.X:X; #Bring node B offline.
    }
    upstream portal_test {
        #Enter the IP address and application service port number of host A.
        #server X.X.X.X:X;
        #Enter the IP address and application service port number of host B.
        server X.X.X.X:X; #Gray release of node B
    }

    server {
        listen XXX;#Enter the Nginx port number.
        server_name localhost;
```

```
location / {
    set $backend portal;
    set $test portal_test;
    #Enter the IP address of the gray verification host.
    if ( $remote_addr ~* "X.X.X.X" ) {
        set $backend $test;
    }
    proxy_pass https://$backend;
}
error_page 500 502 503 504 /50x.html;
location = /50x.html {
    root html;
}
}
```

- **Example code to bring a node online**

```
worker_processes 1;
events {
    worker_connections 1024;
}
http {
    include mime.types;
    default_type application/octet-stream;
    log_format main '$time_local $remote_addr [$remote_port] $request $request_method $content_length';
    log_format test '$content_type $http_referer $host $http_x_forwarded_for'
    '$http_true_client_ip $server_name $request_uri $server_addr $server_port'
    '$status $request_time $upstream_addr $upstream_response_time $cookie_domain_tag';
    access_log logs/access.log main; #Access log: storage path and log level
    error_log logs/error.log; #Error log: storage path
    sendfile on;
    keepalive_timeout 65;
    upstream portal {
        #Enter the IP address and application service port number of host A.
        server X.X.X.X:X;
        #Enter the IP address and application service port number of host B.
        server X.X.X.X:X;
    }
    upstream portal_test {
        #Enter the IP address and application service port number of host A.
        server X.X.X.X:X;
        #Enter the IP address and application service port number of host B.
        server X.X.X.X:X;
    }

    server {
        listen XXX; #Enter the Nginx port number.
        server_name localhost;

        location / {
            set $backend portal;
            set $test portal_test;
            #Enter the IP address of the gray verification host.
            #if ( $remote_addr ~* "X.X.X.X" ) {
            #    set $backend $test;
            #}
            proxy_pass https://$backend;
        }
        error_page 500 502 503 504 /50x.html;
        location = /50x.html {
            root html;
        }
    }
}
```

3 Using Kubernetes Nginx-Ingress for Gray Release

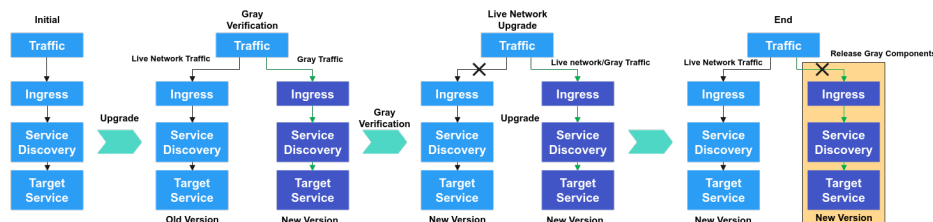
Application Scenario

This practice implements gray release based on native Kubernetes features. When you upgrade to a new system, services may be stopped or gray verification may fail. The native Kubernetes service features help you upgrade system smoothly without affecting services.

Solution Architecture

During system upgrade, a group of gray loads are created when developers deploy applications for the first time. In this case, the system version in the gray loads is the new version. The Service forwards some traffic to the gray loads, and the testers verify the version in the gray loads. After the version verification is complete, the developers start to deploy the application for the second time to upgrade the services on the live network. In this case, the Service forwards all traffic to the gray loads and upgrades the services to the latest version on the live network. After the upgrade is complete, the Service forwards all traffic back to the live network load and releases the gray loads. Now, the new system is released.

Figure 3-1 Gray release scheme



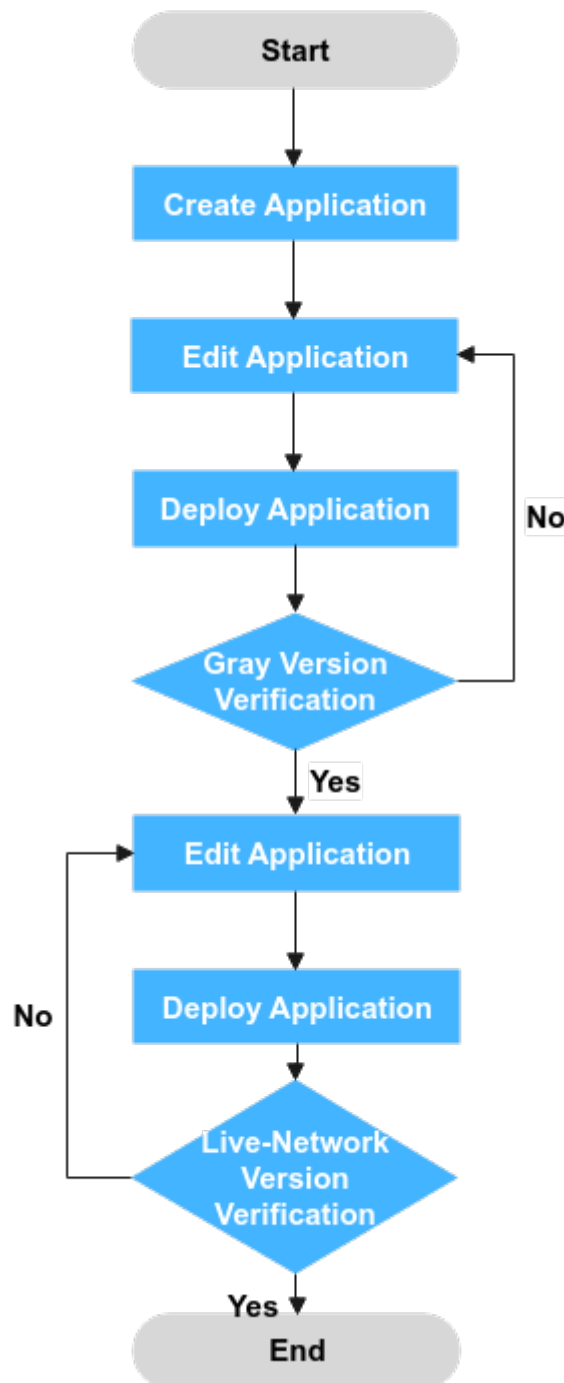
Prerequisites

- A project is available. If there is no project, [create one](#) first.
- You have the permissions to create applications. For details, see [Editing Permissions](#).
- The service contains the following resources and they are defined as version 1:

- A CCE cluster, for example, **cce-demo**, is available.
- A deployment, for example, **deployment-doc**, has been created in the CCE cluster.
- A Service, for example, **service-doc**, has been created in the CCE cluster.
- A route, for example, **ingress-doc**, has been created in the CCE cluster.
- The nginx-ingress plug-in has been installed in the CCE cluster.

Process

Figure 3-2 Process flowchart



Step 1 Create an application.

1. Go to the CodeArts homepage and click the target project name to access the project.
2. Choose **CICD > Deploy** and click **Create Application**. The **Set Basic Information** page is displayed.
3. You can modify the following basic information as required:

Parameter	Mandatory	Description
Name	Yes	Name of an application. Example: Kubernetes_Nginx-Ingress_Gray_Deployment
Project	Yes	Retain the default value. Project to which an application belongs.
Description	No	Description of an application. Example: None
Execution Resource Pool	No	A resource pool is a collection of physical environments where commands are executed during software package deployment. You can use the official resource pool hosted by Huawei Cloud or host your own servers as a self-hosted resource pool on Huawei Cloud. For details about how to host your own servers, see Self-hosted Resource Pool . Example: Official
Deploy from pipeline	No	After this function is enabled, the app can be executed only by the pipeline driver and cannot be executed independently.

4. After editing the basic application information, click **Next**. On the **Select Template** page, select **Blank Template** and click **OK**.

Step 2 Edit the application.

On the **Deployment Actions** tab page, add **Kubernetes Nginx-Ingress Gray Deployment (CCE cluster)** and modify the parameters described in the following table.

Table 3-1 Parameter description

Parameter	Description	Example
Action Name	Name of an action displayed in Deployment Actions area.	Retain the default value.

Parameter	Description	Example
Tenant	<ul style="list-style-type: none">• Current tenant: The software package is deployed in your CCE cluster for release. Select Current tenant. You must have the CCE cluster operation permission. If you do not have it, select Authorized User for deployment.• Other tenant: The software package is deployed in the CCE cluster of another tenant for release in IAM authorization mode. If you select Other tenant, you must select an authorized tenant to deploy the CCE cluster.	Select Current tenant .
Authorized User	If you do not have the permission to execute an API, this parameter enables you to obtain the temporary AK/SK of the parent user to execute the CCE API through IAM.	Deselect it.
Region	Select the region for deployment.	Retain the default value.
Cluster Name	Select the Kubernetes cluster applied on CCE.	cce-ldf
Namespace	Select the namespace of the Kubernetes cluster on CCE.	Retain the default value.
Workload	Select the target deployment.	deployment-doc
Service	Name of the service bound to the target workload.	service-doc
Ingress	Select the name of the route bound to the target service.	ingress-doc
Container	Select the name of the CCE container to be deployed.	container-1
Image	Select the image to be deployed.	Retain the default value.
Image Tag	Select the tag of the image to be deployed.	v2

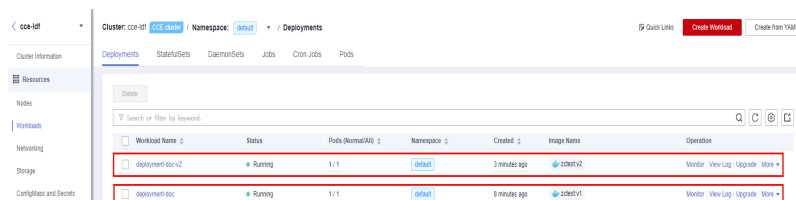
Parameter	Description	Example
Enabling grayscale configuration	Gray Strategy: <ul style="list-style-type: none">• Header Header-Key: You can enter the key of a custom header. Header-Value: You can enter a custom header value. The value can be a character string or a regular expression. The regular expression format is <code>^....\$</code>. Gray Traffic Weight (%): Traffic can be customized.• Cookie Cookie: Custom cookie content can be entered. Gray Traffic Weight (%): Traffic can be customized. Length limit for Header and Cookie : 500 characters each.	Selected Gray Strategy: Header Header-Key: foo Header-Value: bar Gray Traffic Weight(%): 30

Step 3 Deploy an application (create a gray version).

Click **Save & Deploy** to deploy the application. CodeArts Deploy has created the following gray version resources in the CCE cluster and diverts 30% of the live network traffic to the gray load.

- **Workload: deployment-doc-v2.** The image version is V2.

Figure 3-3 Adding a workload whose image version is V2



- **Service: service-doc-v2**
- **Route: ingress-doc-v2**

In this case, you can add a data record (the value of **Key** is **foo** and the value of **Value** is **bar**) to the header to verify the latest version in the gray load.

Step 4 Edit the application (deploy the latest version).

Go to the application created in **Step 1** and modify the following parameters.

Table 3-2 Parameter description

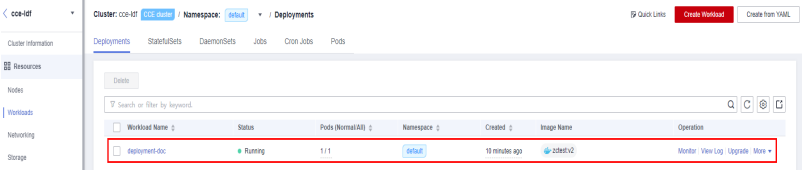
Parameter	Ma nda tor y	Example
Enabling grayscale configuration	No	Deselect it.

Step 5 Deploy the application (deploy the latest version).

Click **Save & Deploy** to deploy the application. CodeArts Deploy has deleted the following gray environment resources from the CCE cluster and replaced the V1 image with the V2 image:

- **Workload:** deployment-doc-v2
- **Service:** service-doc-v2
- **Route:** ingress-doc-v2

Figure 3-4 The image version is upgraded to V2.



You can check whether the system is the latest version on the live network. For more deployment problems, see [Creating an Application FAQs](#).

----End

4 HE2E DevOps Practice - Deploying an Application

4.1 Overview

This section uses the DevOps process example project to describe how to deploy applications on CCE and ECS.

There are three deployment applications preset in the sample project.

The first application is used for CCE deployment, and the second and third applications are used for ECS deployment.

Table 4-1 Preset applications

Preset Application	Description
phoenix-cd-cce	For deployment to CCE
phoenix-sample-standalone	For deployment to ECS
phoenix-sample-predeploy	For installing dependency tools on ECS

4.2 Deploying an Application on CCE

Buying and Configuring CCE

This section uses Cloud Container Engine (CCE) for deployment.

[Buy a CCE cluster](#) on the console.

For details about the mandatory configurations of clusters and nodes, see [Table 4-2](#) and [Table 4-3](#). You can select the configurations that are not listed in the table based on the site requirements.

Table 4-2 Buying a CCE cluster

Category	Parameter	Suggestion
Basic Settings	Type	Select CCE Standard Cluster .
	Billing Mode	Select Pay-per-use .
	Cluster Name	Enter a name.
	Cluster Version	Select a version as required. You are advised to select the latest version.
Network Settings	Network Model	Select Tunnel network .
	VPC	Select an existing VPC. If no proper VPC is available in the list, click Create VPC .
	Default Node Subnet	Select an existing subnet. If no proper subnet is available in the list, click Create Subnet .
	Container CIDR Block	Click Auto select .

Table 4-3 Configuring a node

Category	Parameter	Suggestion
Node Configuration	Billing Mode	Select Pay-per-use .
	Node Type	Select Elastic Cloud Server (VM) .
	Specifications	Select 2 vCPUs and 8 GiB memory or higher.
	OS	Click Public image and select an Euler image.
	Node Name	Enter a custom name.
	Login Mode	Select Password .

Category	Parameter	Suggestion
	Password	Enter a password.
Network Settings	Node IP	Select Automatic .
	EIP	Select Auto create .

4.3 Deploying an Application on ECS

Purchasing and Configuring an ECS

This section uses Elastic Cloud Server (ECS). You can also use your own Linux host that runs Ubuntu 16.04 OS.

Step 1 [Buy an ECS](#).

The following table lists mandatory configurations. You can also select other configurations as necessary.

Table 4-4 Configuring an ECS purchase

Category	Parameter	Suggestion
Configure Basic Settings	Billing Mode	Select Pay-per-use .
Instance	CPU Architecture	Select x86 .
	Specifications	Select 2 vCPUs and 8 GiB memory or higher.
OS	Image	Choose Public image > Ubuntu > Ubuntu 16.04 server 64bit .
Public Network Access	EIP	Select Auto assign .
	Billed By	Select Bandwidth .
Instance Management	Login Mode	Select Password .
	Password	Enter a password.

Step 2 Configure security group rules.

Use ports 5000 and 5001 to verify the sample project. Therefore, add an inbound rule that allows access over ports 5000 and 5001.

The procedure is as follows:

1. Log in to the ECS list page, locate the ECS purchased in step [Step 1](#), and click the ECS name.
2. Click the **Security Groups** tab, and add an inbound rule with **Protocol** set to **TCP** and **Port** set to **5000-5001** by referring to [Configuring Security Group Rules](#).

----End

Adding a Target Host to the Project

Before deploying applications to ECSs, add the target hosts to the basic resources of the project.

Step 1 Go to the **Phoenix Mall** project and choose **Settings > General > Basic Resources** from the navigation pane.

Step 2 Click **Create Host Cluster**, configure the following information, and click **Save**.

Table 4-5 Creating a host cluster

Parameter	Suggestion
Cluster Name	Enter host-group .
OS	Select Linux .
Host Connection Mode	Select Direct Connection .
Execution Resource Pool	Select Official .

Step 3 Click **Add Host**, configure the following information, and click **OK**.

Table 4-6 Adding a host

Parameter	Suggestion
Add Hosts by	Select Adding IP .
Host Name	Keep this name same as the name of the purchased ECS.
IP	Enter the EIP generated when buying the ECS.
Authorization	Select Password .
Username	Enter root .
Password	Enter the password set when buying the ECS.
SSH Port	Enter 22 .

Step 4 A host record is displayed on the page. If **Succeed** is displayed in the **Verification Result** column, the host is added successfully.

If the host fails to be added, check the host configuration based on the failure details.

----End

Installing Dependency Tools on ECS

The sample program depends on Docker and Docker-Compose, which must be installed on the target ECS.

Step 1 Go to the **Phoenix Mall** project, choose **CICD > Deploy**, and find the **phoenix-sample-predeploy** application in the list.

Step 2 Click *** and choose **Edit** from the drop-down list.

Step 3 Click the **Environment Management** tab and configure the host environment.

1. Click **Create Environment**, configure the following information, and click **Save**.

Table 4-7 Creating an environment

Parameter	Value
Environment	Enter phoenix-env .
Resource Type	Select Host .
OS	Select Linux .

NOTE

If you do not have permission to create environments, contact the administrator to grant you permissions on the permission management page of the application.

2. Click **Import Host**. In the displayed dialog box, select the configured host cluster and host and click **Import**.
3. When a message indicating that the import is successful is displayed, close the window.

Step 4 On the **Deployment Actions** tab page, edit the actions of the application.

In action **Install Docker**, select **phoenix-env** from the **Environment** drop-down list. If a dialog box is displayed, asking you to confirm whether you want to change the environment to **phoenix-env** for the subsequent actions, click **OK**.

Step 5 Click **Save & Deploy** to start the deployment task.

If a message is displayed indicating successful deployment, the task is successfully executed.

Step 6 Log in to the ECS and check whether the dependency tools are successfully installed:

- Run the **docker -v** command to check the Docker image version.
- Run the **docker-compose -v** command to check the Docker-Compose version.

If the command output similar to [Figure 4-1](#) is displayed, the installation is successful.

Figure 4-1 Checking the Docker and Docker-Compose versions

```
root@ecs-he2e:~# docker -v
Docker version 18.09.0, build 4d60db4
root@ecs-he2e:~# docker-compose -v
docker-compose version 1.17.1, build 6d101fb
root@ecs-he2e:~#
```

----End

Configuring and Executing an Application

During deployment, configure the ECS in the environment list of the application and set the build task **phoenix-sample-ci** as the deployment source.

- Step 1** Go to the **Phoenix Mall** project, choose **CICD > Deploy**, and find the **phoenix-sample-standalone** application in the list.
- Step 2** Click ******* and choose **Edit** from the drop-down list.
- Step 3** Click the **Environment Management** tab and configure the host environment.
1. Click **Create Environment**, configure the following information, and click **Save**.

Table 4-8 Creating an environment

Parameter	Value
Environment	Enter phoenix-env .
Resource Type	Select Host .
OS	Select Linux .

2. Click **Import Host**. In the displayed dialog box, select the configured host cluster and host and click **Import**.
 3. When a message indicating that the import is successful is displayed, close the window.
- Step 4** On the **Deployment Actions** tab page, edit the actions of the application.
1. Click **Select a Deployment Source**. Set the deployment source by referring to [Table 4-9](#).

Table 4-9 Configuring the deployment source

Parameter	Suggestion
Source	Select Build task .
Build Task	Select phoenix-sample-ci .
Environment	Select phoenix-env . If a dialog box is displayed, asking you to confirm whether you want to change the environment to phoenix-env for the subsequent actions, click OK .

2. Retain the default settings in actions **Decompress Files** and **Run Shell Commands**.

Step 5 Click the **Parameters** tab and set parameters based on the SWR login command.
Obtain the login command from the console.

Step 6 Click **Save & Deploy** to start deployment.

If a message is displayed indicating that the deployment is successful, continue with the next step. If the deployment fails, rectify the fault based on the failed action and the error information in logs.

Step 7 Verify the deployment result.

1. Open a browser, enter **http://IP:5000** in the address box, and press **Enter**. *IP* indicates the elastic IP address of the ECS. The Phoenix Mall homepage is displayed.
2. Enter **http://IP:5001** and press **Enter**. *IP* indicates the elastic IP address of the ECS. The Phoenix Mall dashboard is displayed.

----End

Releasing Resources

To avoid unnecessary fees, after the practice.


If you need to create a pipeline after the deployment, you can delete the ECS after completing the pipeline practice.

4.4 Releasing Resources

To avoid unnecessary expenses, Sarah can release unused resources after trying the sample project.

The following resources can be released.

Table 4-10 Releasing resources

Resource	Procedure
CodeArts project	Choose Settings > General > Basic Information , click Delete Project , and follow the prompts to delete the project.
SWR organization and image	<ol style="list-style-type: none">1. Log in to the SWR console.2. On the My Images page, select the image created in this example, click Delete, and follow the prompts to delete the image.3. On the Organizations page, click the name of the organization to be deleted. Click Delete and follow the prompts to delete the organization.
CCE cluster	Log in to the CCE console. Locate the cluster to be deleted in the list, click  , select Delete Cluster , and follow the prompts to delete it.
ECS	Log in to the ECS console. Locate the ECS to be deleted in the list, choose More > Delete , and follow the prompts to delete it.

NOTICE

Released resources cannot be recovered. Exercise caution when performing these operations.