

CodeArts Deploy

Best Practices

Issue 01
Date 2023-12-08



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2023. 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 Internal Network Using a Proxy Host.....	1
2 Using Nginx for Gray Release.....	6
3 Implementing Grays Release Based on Kubernetes Nginx-Ingress.....	29

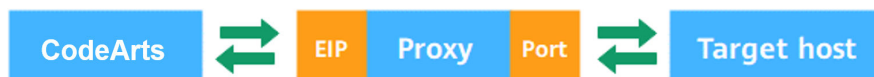
1 Deploying an Application on Internal Network Using a Proxy Host

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

Process

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.

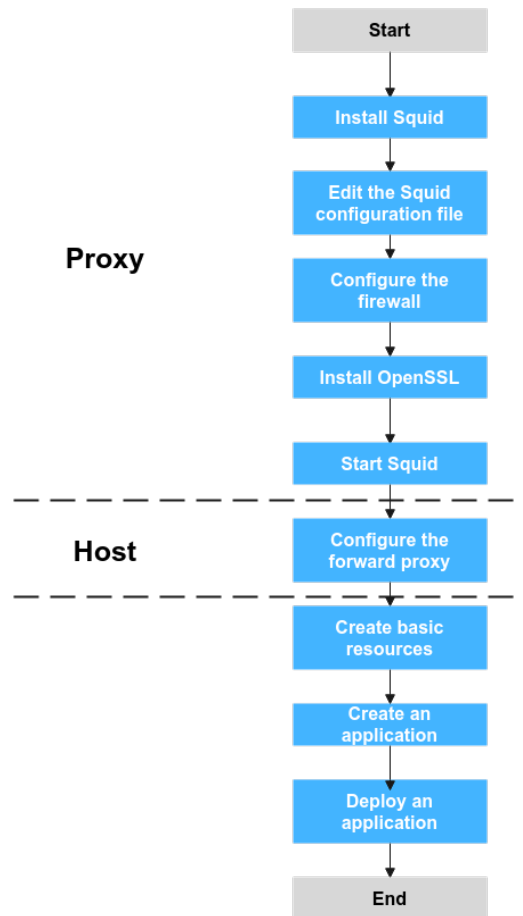
For more information about Squid, go to [Squid official website](#). The following procedure uses a Linux host as an example.



Prerequisites

- A host (**Proxy-B**) bound to a public IP address is available. If no proxy host is available, see [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.

Procedure



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 10.0.0.0/24          # RFC1918 possible internal network
acl localnet src 172.16.0.0/24       # RFC1918 possible internal network
acl localnet src 192.168.0.0/24      # 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
```

NOTE

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:

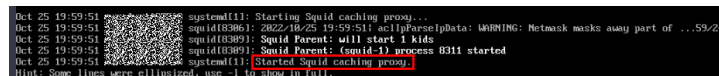
```
yum install openssl
```

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.
```



```
Oct 25 19:59:51 localhost systemd[1]: Starting Squid caching proxy...
Oct 25 19:59:51 localhost squid[8386]: 2022/10/25 19:59:51: aclParseIpData: WARNING: Netmask masks away part of ...59/24
Oct 25 19:59:51 localhost squid[8389]: Squid Parent: will start 1 kids
Oct 25 19:59:51 localhost squid[8391]: Squid Parent: squid[8391] process 8311 started
Oct 25 19:59:51 localhost systemd[1]: Started Squid caching proxy.
Hint: Some lines were ellipsized, use -l to show in full.
```

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. In the target project, choose **Settings > General > Basic Resources**. The **Host Clusters** page is displayed.
2. Click **Create Host Cluster**, enter the following information, and click **Save**.

Parameter	Mandatory	Description
Cluster Name	Yes	Enter a custom name.
OS	Yes	Select Linux based on the OS of the host to be added.

Parameter	Mandatory	Description
Use Proxy	Yes	Enable the option.
Execution Host	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 the description of the host cluster.

3. Click **Add Proxy Host**, enter the following information, and click **OK**.

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.

4. Click **Add Target Host**, enter the following information, and click **OK**.

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.
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.

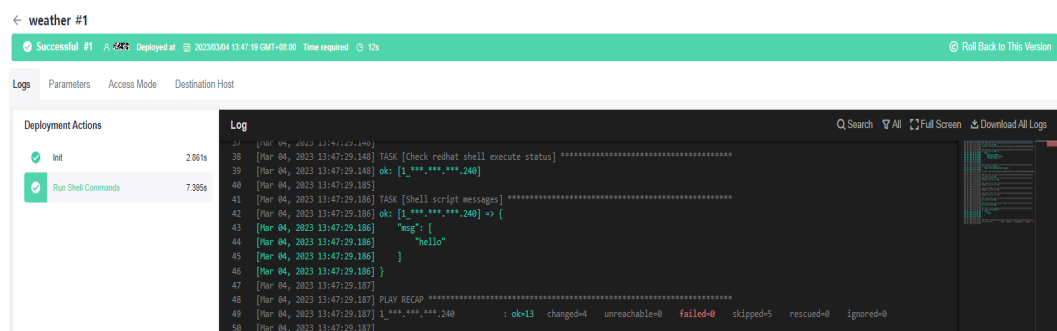
Parameter	Mandatory	Description
Authorization	Yes	In this scenario, the Password is used for authentication. Enter the username and password of Host-A .
SSH Port	Yes	Port 22 is recommended.

- Click  in the **Operation** column of a host to start the host for connectivity verification.

Step 8 Create an application.

- Log in to the CodeArts homepage and click the target project name to access the project.
- Choose **CICD > Deploy**.
- Click **Create Application**. On the **Set Basic Information** page, modify the basic information such as **App Name**, **Description**, and **Execution Resource Pool** as required.
- After editing the basic application information, click **Next**. On the deployment template selection 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 dialog box that is displayed, select the target host cluster and import **Proxy-B** and **Host-A** to the environment.

Step 9 Deploy the application. For details, see [Managing Applications](#).



```
37 [Mar 04, 2023 13:47:29.149]
38 [Mar 04, 2023 13:47:29.148] TASK [Check redhat shell execute status] *****
39 [Mar 04, 2023 13:47:29.148] ok: [1***.***.***.240]
40 [Mar 04, 2023 13:47:29.185]
41 [Mar 04, 2023 13:47:29.186] TASK [Shell script messages] *****
42 [Mar 04, 2023 13:47:29.186] ok: [1***.***.***.240] => {
43 [Mar 04, 2023 13:47:29.186]   "msg": [
44 [Mar 04, 2023 13:47:29.186]     "hello"
45 [Mar 04, 2023 13:47:29.186]   ]
46 [Mar 04, 2023 13:47:29.186] }
47 [Mar 04, 2023 13:47:29.187]
48 [Mar 04, 2023 13:47:29.187] PLAY RECAP *****
49 [Mar 04, 2023 13:47:29.187] 1***.***.***.240
50 [Mar 04, 2023 13:47:29.187]      : ok=13  changed=4  unreachable=0  failed=0  skipped=5  rescued=0  ignored=0
```

----End

2 Using Nginx for Gray Release

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](#).

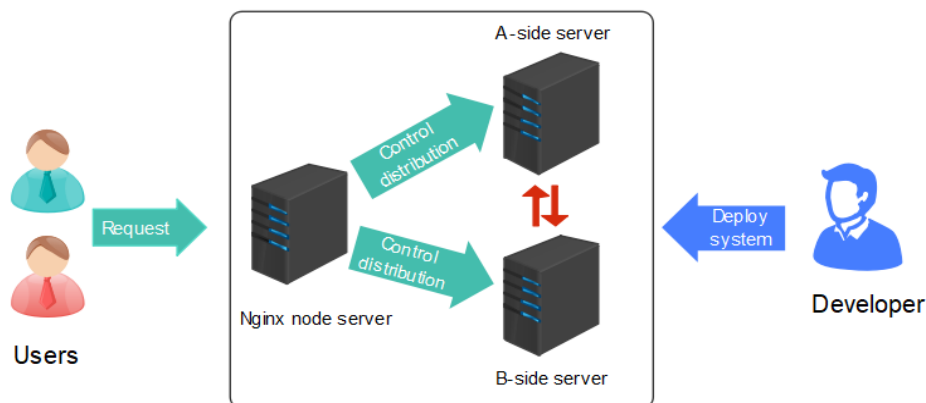
Application Scenario

When you upgrade 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.

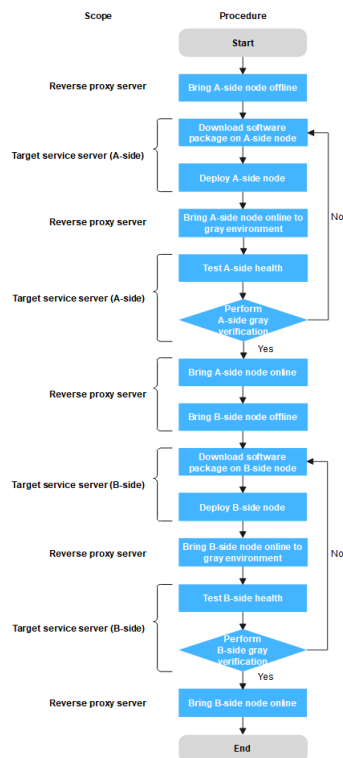
Solution Architecture

During system upgrade, if the blue-green deployment mode is used, developers bring the server on side A (original blue environment) offline and distribute all access traffic to the server on side B. In this case, the server on side A is upgraded. After the A-side server is upgraded, set the server as the gray test environment. A tester performs gray verification on the A-side server. After the gray verification is complete and the functions are normal, the A-side server (green environment) is officially released, and all traffic is distributed to the A-side server. In this case, the blue-green deployment is complete. If an emergency occurs on the A-side server during service running, perform a blue-green switchover to quickly restore services.

Figure 2-1 Gray release scheme



If you use canary release, repeat the preceding operations to upgrade the B-side server, complete the gray test, and release the system officially. In this case, the gray release of the new system is complete.



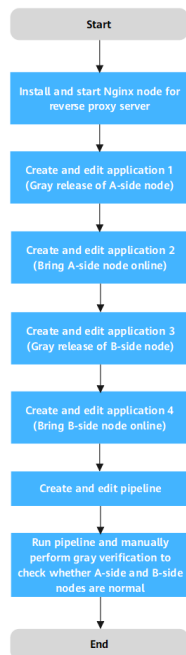
Prerequisites

- A project is available. If no project is available, [create one](#) first.
- You have the permission 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.

NOTE

Ensure that servers can communicate with each other. For example, add all servers to the same Virtual Private Cloud (VPC).

Procedure



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

NOTE

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

1. Create basic resources.
 - a. In the target project, choose **Settings > General > Basic Resources**. The **Host Clusters** page is displayed.
 - b. Click **Create Host Cluster**, enter basic information such as the **Cluster Name**, **OS**, **Use Proxy**, **Execution Host**, and **Description**, and click **Save**.
 - c. Click **Create Target Host**, enter the **Host Name** (for example, **A_test**, **B_test**, or **Gray_release**), **IP**, **Username**, **Password** or **Key**, and **SSH port**, and click **OK**. Repeat the preceding steps to create the three target hosts and verify the connectivity.
2. Create an application.
 - a. Choose **CICD > Deploy**.
 - b. Click **Create Application**. On the **Set Basic Information** page, modify the basic information such as **App 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 and add and edit an environment.


- Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to the server, and enter the description.
 - 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 dialog box that is displayed, 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 and edit the following steps:
- 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	Description
Environment	Select Reverse_proxy_server_group .
Nginx Version	Select a target version. Example: nginx-1.14.2 .
Installation Path	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	Description
Environment	Select Reverse_proxy_server_group .
Operation	Select Start Nginx .
Nginx Installation Path	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.
- After the deployment is complete, the application status bar changes to green and the message **Successful** is displayed, indicating that the application is successfully deployed.
- If the application status bar turns red and displays **Failed**, the application fails to be deployed. In this case, click **View Solution**.

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


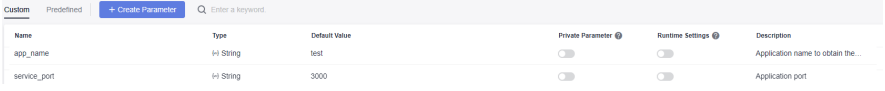
1. Create an application.
 - a. Choose **CICD > Deploy**.
 - b. Click **Create Application**. On the **Set Basic Information** page, modify the basic information such as **App Name**, **Description**, and **Execution Resource Pool** as required.
 - c. After editing the basic application information, click **Next**. The deployment template selection page is displayed.
 - d. Select the **Deploy a General Application** template and click **OK**.
2. Edit the application.
 - a. Switch to the **Environment Management** tab page and add and edit an environment.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to the server, and enter the description.
 - 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 dialog box that is displayed, 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 **Target service server group_A-side node** and add the **A_test** server.
 - b. Switch to the **Parameters** tab page and add the following parameters:
 
 - c. Switch to the **Deployment Actions** tab page. Add and edit the following steps:
 - 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	Description
Action Name	Enter a name such as Bring_A-side_node_offline .
Environment	Select the target environment. Example: Reverse_proxy_server_group .
Operation	Specify the operation type Example: Reload configuration file .

Parameter	Description
Nginx Installation Path	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	Select this parameter.
Nginx Configuration File Path	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	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	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 change the parameter values to those listed in the following table (Linux is used as an example).

Table 2-4 Parameter description

Parameter	Description
Action Name	Enter a name Download_software_package_on_A-side_node .
Source	Select a source Example: Artifact .
Environment	Select the target environment. Example: Target service server group_A-side node .
Software package	Select a software package to be deployed in CodeArts Artifact.
Download Path	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 as follows (Linux is used as an example):

Table 2-5 Parameter description

Parameter	Description
Action Name	Enter the action name Deploy A-side node .
Environment	Select the target environment. Example: Target service server group_A-side node .
Shell Commands	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

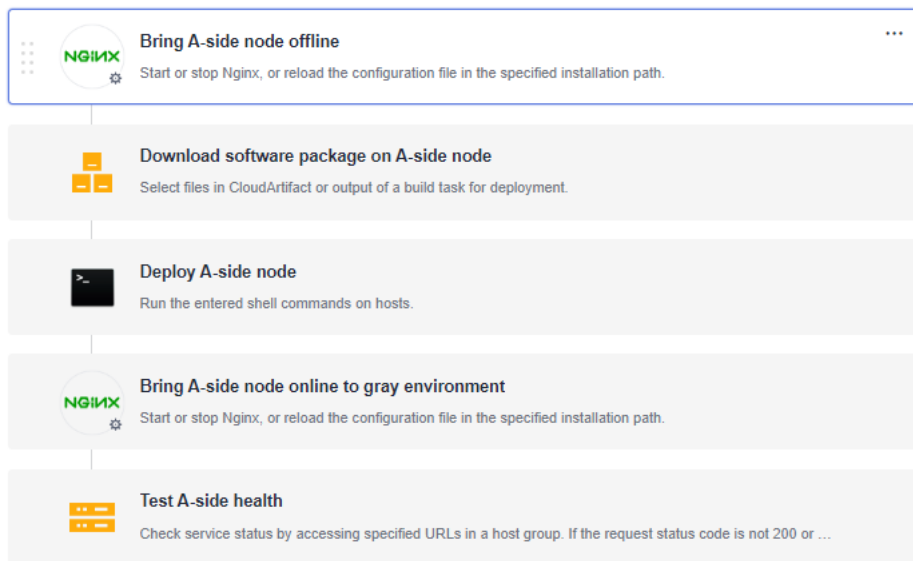
Parameter	Description
Action Name	Enter a name such as Bring A-side node online to gray environment .
Environment	Select the target environment. Example: Reverse_proxy_server_group .
Operation	Specify the operation type Example: Reload configuration file .
Nginx Installation Path	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	Select this parameter.
Nginx Configuration File Path	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	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	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 as follows (Linux is used as an example):

Table 2-7 Parameter description

Parameter	Description
Action Name	Enter a name such as Test_A-side_health .
Environment	Select the target environment. Example: Target service server group_A-side node .
Retries	If a service does not start up when the health test reaches the maximum retries, the service fails this test. Example: 1
Interval (s)	Interval between two retries, in seconds. Example: 60
Test Path	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 **Set Basic Information** page, modify the basic information such as **App 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 and add and edit an environment.


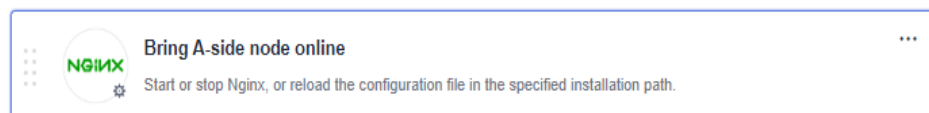
- Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to the server, and enter the description.
 - Click **Save**. The environment is created.
 - Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the dialog box that is displayed, 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 and edit the following steps:
- 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	Description
Action Name	Enter a name such as Bring_A-side_node_online .
Environment	Select the target environment. Example: Reverse_proxy_server_group .
Operation	Specify the operation type Example: Reload configuration file .
Nginx Installation Path	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	Select this parameter.
Nginx Configuration File Path	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	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	Enter content of the new configuration file. See Example code to bring a node online in the appendix.

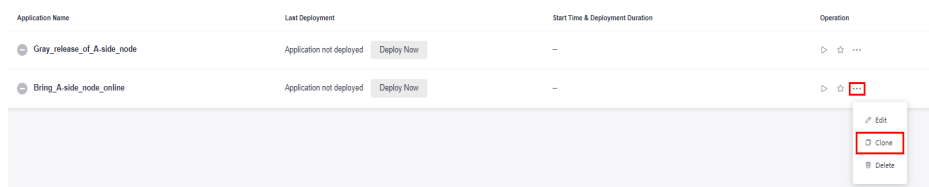
3. Click **Save**. The application is created.



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 and add and edit an environment.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to the server, and enter the description.
 - 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 dialog box that is displayed, 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 **Target service server group_B-side node** and add the **B_test** server.
- b. Switch to the **Deployment Actions** tab page. Add and edit the following steps:
 - 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	Description
Action Name	Enter a name such as Bring_B-side_node_offline .
Environment	Select the target environment. Example: Reverse_proxy_server_group .
Operation	Specify the operation type Example: Reload configuration file .
Nginx Installation Path	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .

Parameter	Description
Modify configuration file before execution	Select this parameter.
Nginx Configuration File Path	Enter the path of the Nginx configuration file on the target host. Example: <code>/usr/local/nginx/conf/nginx.conf</code> .
Configuration File Backup Path	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	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	Description
Action Name	Enter a name such as <code>Download_software_package_on_B-side_node</code> .
Source	Select a source Example: Artifact .
Environment	Select the target environment. Example: B_group .
Software package	Select a software package to be deployed in CodeArts Artifact.
Download Path	Enter the path for downloading the software package to the target host. Example: <code>/usr/local/</code> .

- 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	Description
Action Name	Enter an action name such as <code>Deploy_B-side_node</code> .

Parameter	Description
Environment	Select the target environment. Example: B_group .
Shell Commands	Enter the commands to be executed. Example: See Example code to deploy a 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

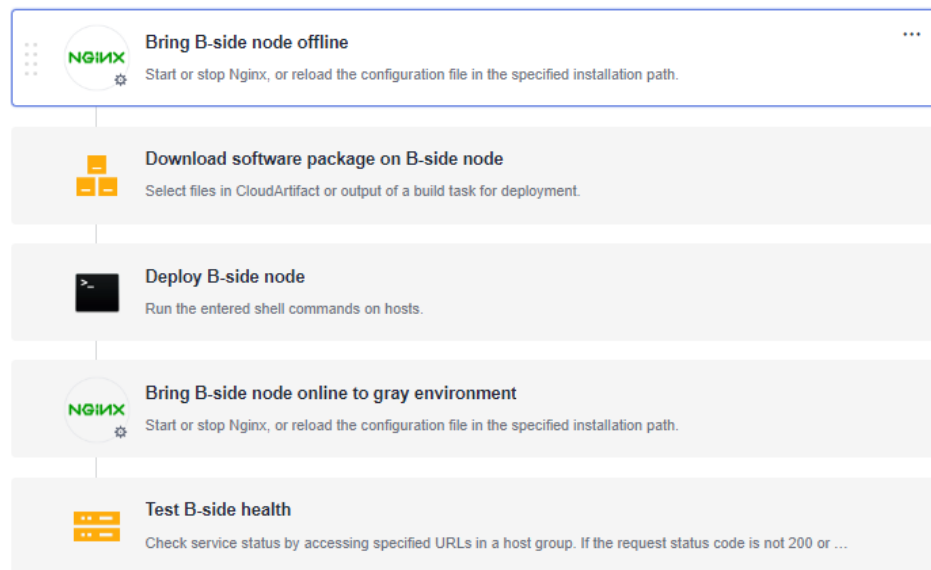
Parameter	Description
Action Name	Enter a name such as Bring_B-side_node_online_to_gray_environment .
Environment	Select the target environment. Example: Reverse_proxy_server_group .
Operation	Specify the operation type Example: Reload configuration file .
Nginx Installation Path	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	Select this parameter.
Nginx Configuration File Path	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	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	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	Description
Action Name	Enter a name such as Test_B-side_health .
Environment	Select the target environment. Example: B_group .
Retries	If a service does not start up when the health test reaches the maximum retries, the service fails this test. Example: 1
Interval (s)	Interval between two retries, in seconds. Example: 60
Test Path	Used for the health test via URLs. You can add multiple URLs. Example: <code>http://127.0.0.1:3000</code> (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 and add and edit an environment.
 - Click **Create Environment**, enter the environment name, for example, **Reverse_proxy_server_group**, select the OS corresponding to the server, and enter the description.
 - Click **Save**. The environment is created.

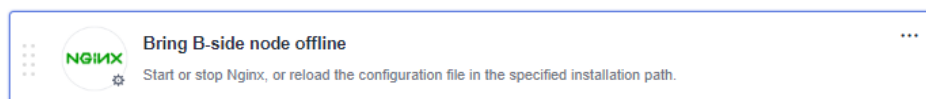
- Click **Import Host**. The system automatically filters all clusters that meet the requirements of the current environment. In the dialog box that is displayed, select the target host cluster and click  in the **Operation** column of the target host to import the host to the environment.
 - b. Switch to the **Deployment Actions** tab page. Add and edit the following steps:

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	Description
Action Name	Enter a name such as Bring_B-side_node_online .
Environment	Select the target environment. Example: Reverse_proxy_server_group .
Operation	Specify the operation type Example: Reload configuration file .
Nginx Installation Path	Enter the installation path of the Nginx service in the target environment. Example: /usr/local/nginx .
Modify configuration file before execution	Select this parameter.
Nginx Configuration File Path	Enter the path of the Nginx configuration file on the target host. Example: /usr/local/nginx/conf/nginx.conf .
Configuration File Backup Path	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	Enter content of the new configuration file. See Example code to bring a node online in the appendix.

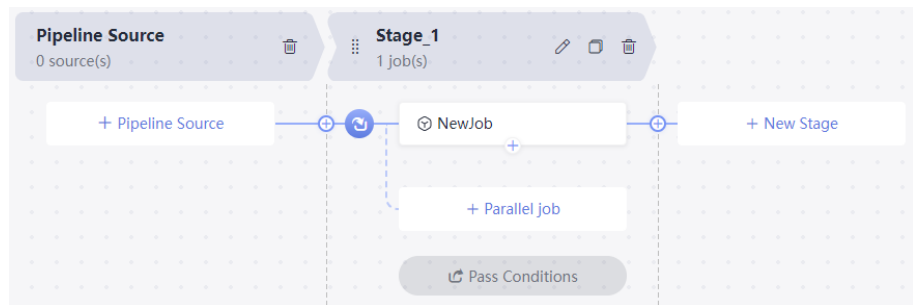
3. Click **Save**. The application is created.



Step 6 Create and edit a pipeline.

1. 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**.




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

Table 2-15 Parameter description

Parameter	Description
Stage Name	Enter a name such as Gray_release_of A-side_node .
Always Run	Select No .


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

Table 2-16 Parameter description

Parameter	Description
Name	Enter a name such as Gray_release_of A-side_node .
Select Task	Select Gray_release_of A-side_node .
Build Task	Leave it not configured.



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

Table 2-17 Parameter description

Parameter	Description
Name	Enter a name such as Bring_A-side_node_online .
Always Run	Select No .



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

Table 2-18 Parameter description

Parameter	Description
Name	Enter a name such as Gray_release_of A-side_node .
Reviewer	Select the service verification personnel.
Review Mode	Select Review by all .
Timeout Processing	Select Review failed and pipeline terminated .
Review Duration	Example: 4 hours.
Description	This parameter is optional.

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

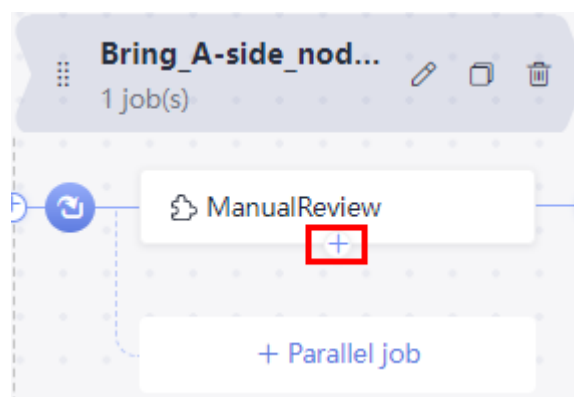


Table 2-19 Parameter description

Parameter	Description
Name	Enter a name such as Bring_A-side_node_online .
Select Task	Select Bring_A-side_node_online .
Build Task	Leave it not configured.

4. Edit job 3 (**Gray_release_of_B-side_node**) in the pipeline stage.



- Click  and . In the dialog box that is displayed, set the parameters as follows and click **Confirm**.

Table 2-20 Parameter description

Parameter	Description
Name	Enter a name such as Gray_release_of_B-side_node .
Always Run	Select No .


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

Table 2-21 Parameter description

Parameter	Description
Name	Enter a name such as Gray_release_of_B-side_node .
Select Task	Select Gray_release_of_B-side_node .
Build Task	Leave it not configured.

5. Create and edit job 4 (**Bring_B-side_node_online**) in the pipeline stage.



- Click  and . In the dialog box that is displayed, set the parameters as follows and click **Confirm**.

Table 2-22 Parameter description

Parameter	Description
Name	Enter a name such as Bring_B-side_node_online .
Always Run	Select No .



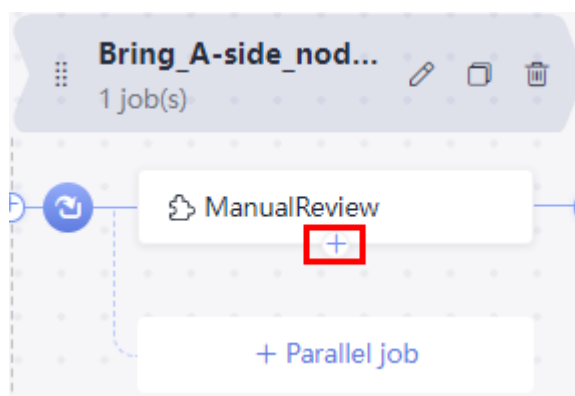
- Click . In the displayed dialog box, set **Entry Type** to **Automatic** and click **OK**.
- Click **NewJob**. In the window that is displayed, click the **Normal** tab, select **ManualReview** and click **Add**, set the parameters as follows, and click **OK**.

Table 2-23 Parameter description

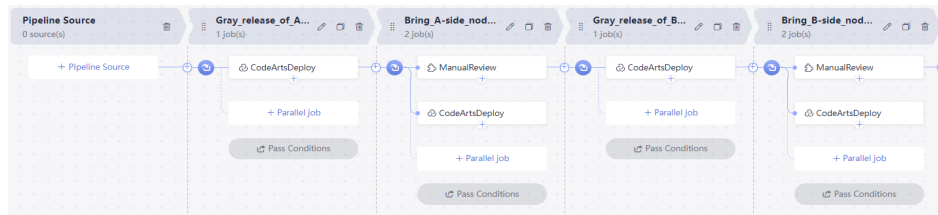
Parameter	Description
Name	Enter a name such as Gray_verification_of_B-side_node .
Reviewer	Select the service verification personnel.
Review Mode	Select Review by all .
Timeout Processing	Select Review failed and pipeline terminated .
Review Duration	Example: 4 hours.
Description	This parameter is optional.

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

**Table 2-24** Parameter description

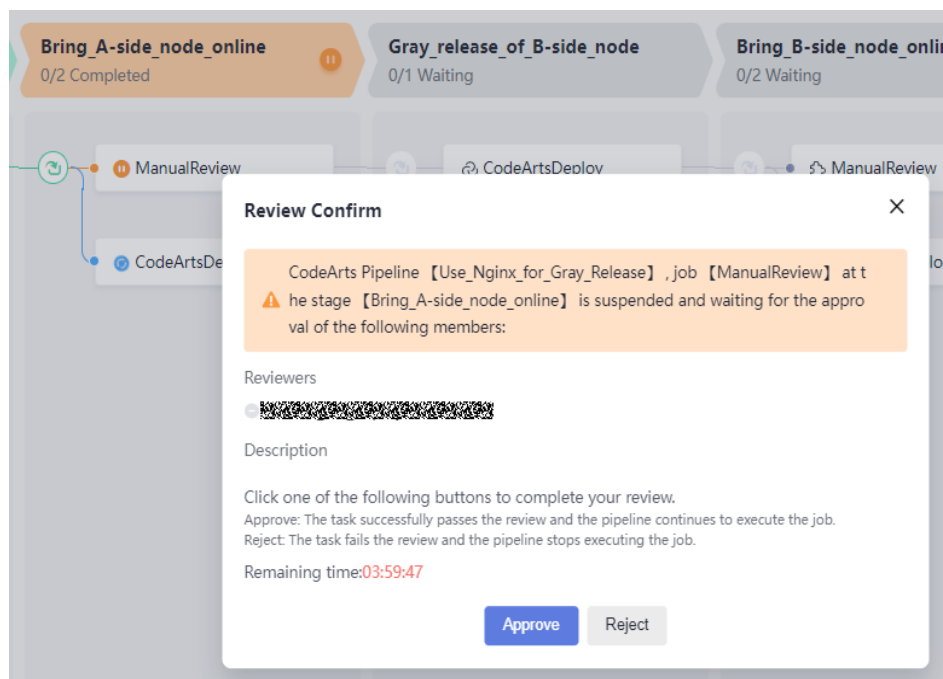
Parameter	Description
Name	Enter a name such as Bring_B-side_node_online .
Select Task	Select Bring_B-side_node_online .
Build Task	Leave it not configured.

- 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.

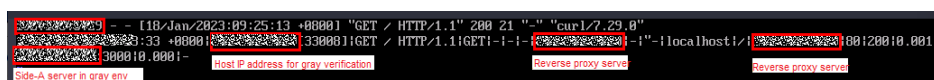


Gray users can run the **curl** command to check whether the gray environment is normal.

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

NOTE

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 view 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|
$content_type|$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; #Bring node B offline.
}
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;
}
}
}
```

- **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;
#Enter the IP address and application service port number of host B.
#server 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;
#Enter the IP address and application service port number of host B.
server 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|
$content_type|$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 Implementing Grays Release Based on Kubernetes Nginx-Ingress

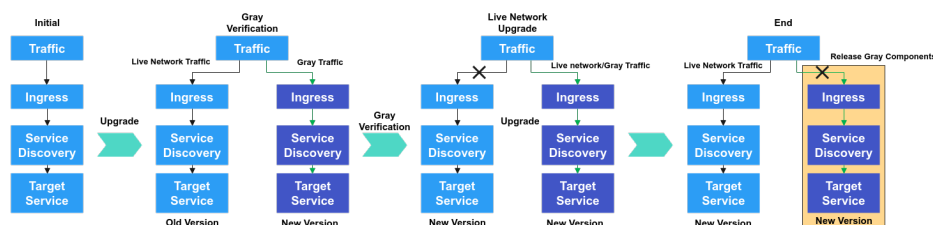
Application Scenario

This practice implements gray release based on native Kubernetes features. When you upgrade 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



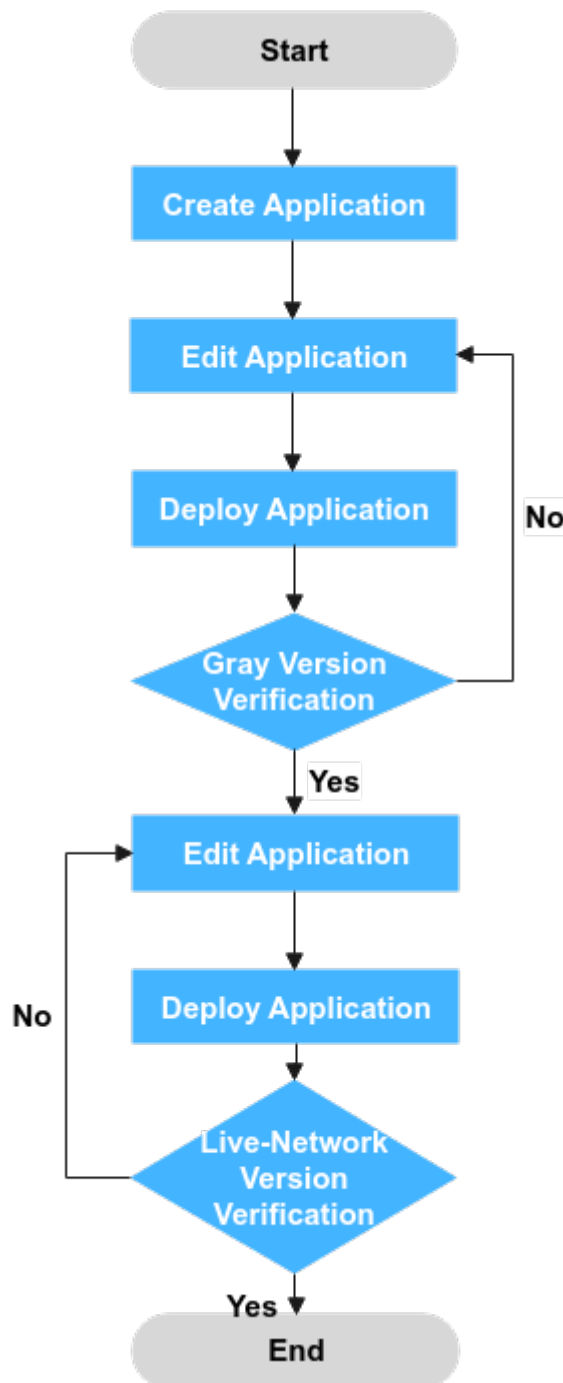
Prerequisites

- A project is available. If no project is available, [create one](#) first.
- You have the permission to create applications. For details, see [Editing Permissions](#).
- Service version 1 is available and contains the following resources:

- A CCE cluster, for example, **cce-ldf**, 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.

Procedures

Figure 3-2 Process flow



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	Description
App Name	Mandatory. Name of an application. Example: Kubernetes_Nginx-Ingress_Gray_Deployment
Project	Retain the default value. Project to which an application belongs.
Description	Optional. Description of an application. Example: None
Execution Host	Optional. 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	Optional. After this function is enabled, the app can be executed only by the pipeline driver and cannot be executed independently. Example: disabled.

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.	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

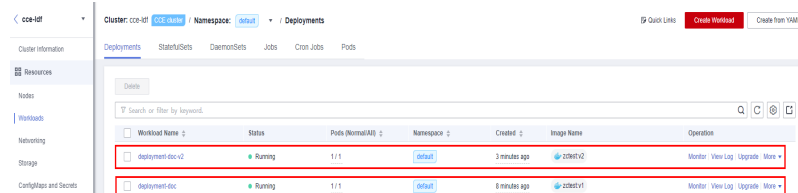
Parameter	Description	Example
Enabling grayscale configuration	<p>Gray Strategy:</p> <ul style="list-style-type: none"> Header Header-Key: You can enter the key of a custom header. Header-Value: You can enter a custom header value. The value can be a character string or a regular expression. The regular 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. <p>NOTE The values of Header and Cookie can contain a maximum of 500 characters.</p>	<p>Selected</p> <p>Gray Strategy: Header Header-Key: foo Header-Value: bar Gray Traffic Weight(%): 30</p>

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**

NOTE

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

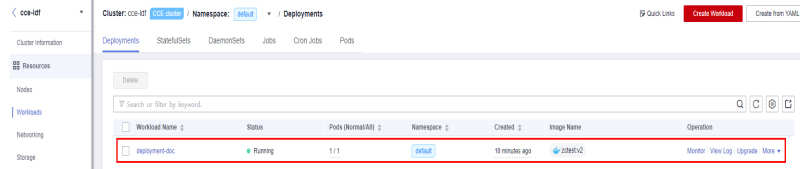
Parameter	Example
Enabling grayscale configuration	Deselect it.

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**
- **Ingress: ingress-doc-v2**

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



NOTE

You can check whether the system is the latest version on the live network.

----End