Alibaba Cloud Container Service

User Guide

Issue: 20181116

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.
5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade

secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).

6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 **Danger:** Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 **Warning:** Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 **Note:** Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	 **Note:** You can use **Ctrl** + **A** to select all files.
>	Multi-level menu cascade.	**Settings** > **Network** > **Set network type**
Bold	It is used for buttons, menus, page names, and other UI elements.	Click **OK**.
`Courier font`	It is used for commands.	Run the `cd /d C:/windows` command to enter the Windows system folder.
Italics	It is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *[-all|-t]*
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich` *{stand | slave}*

Contents

1 Authorizations

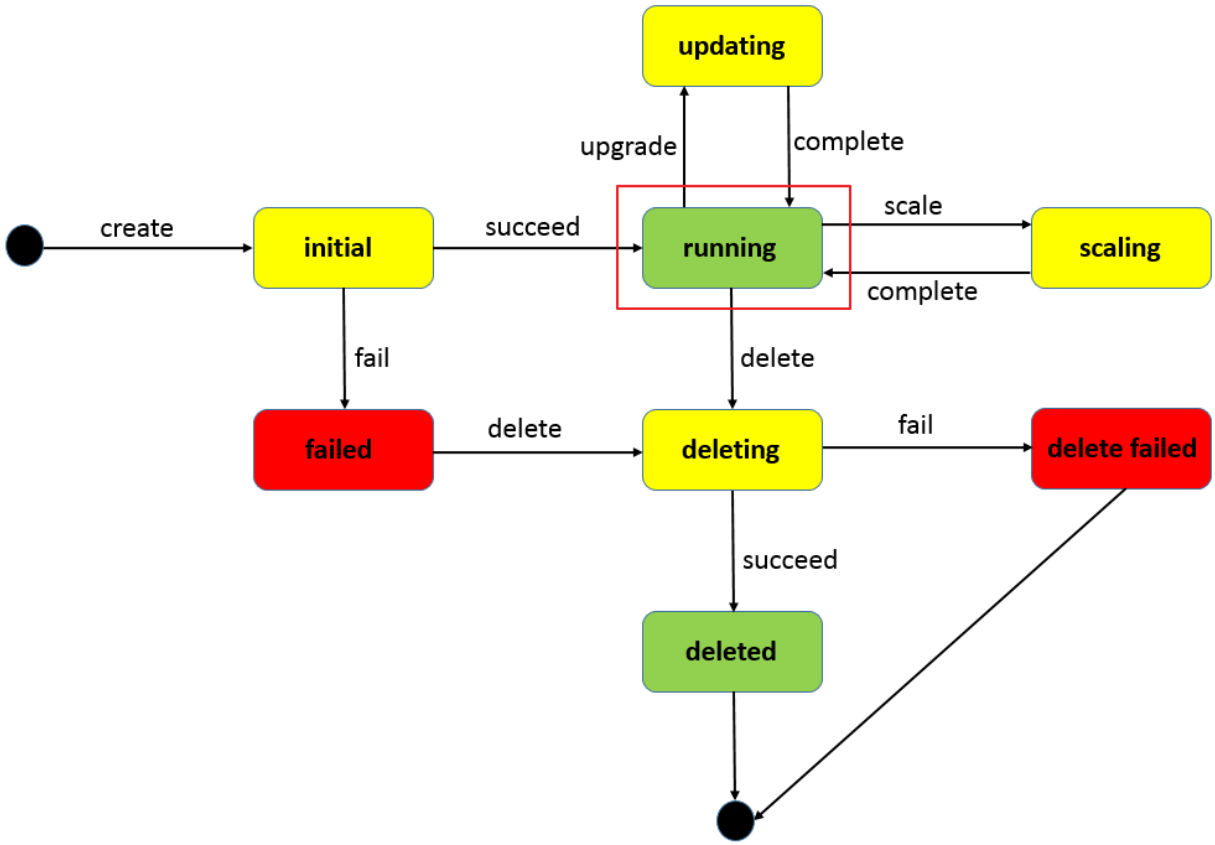
2 Clusters

2.1 Cluster lifecycle

Table 2-1: A complete cluster lifecycle includes the following statuses.

Status	Description
inactive	The successfully created cluster does not contain any node.
initial	The cluster is applying for corresponding cloud resources.
running	The cluster successfully applied for the cloud resources.
updating	The cluster is upgrading the Agent.
scaling	Change the number of cluster nodes.
failed	The cluster application for cloud resources failed.
deleting	The cluster is being deleted.
delete_failed	The cluster failed to be deleted.
deleted (invisible to users)	The cluster is successfully deleted.

Figure 2-1: Cluster status flow



2.2 Add an existing ECS instance

You can add a purchased Elastic Compute Service (ECS) instance to a specified cluster.

Note:
At most 20 ECS instances can be added to a cluster by default. To add more ECS instances, *open a ticket*.

You can add an existing ECS instance in the following ways:

- **Add ECS instances automatically:** The image and system disk of the ECS instance are reset by using this method. You can add one or more ECS instances to the cluster at a time.
- **Add the ECS instance manually:** Manually add the ECS instance by running scripts on the ECS instance. You can only add one ECS instance to the cluster at a time.

Prerequisites

If you have not created a cluster before, create a cluster first. For information about how to create a cluster, see *Create a cluster*.

Instructions

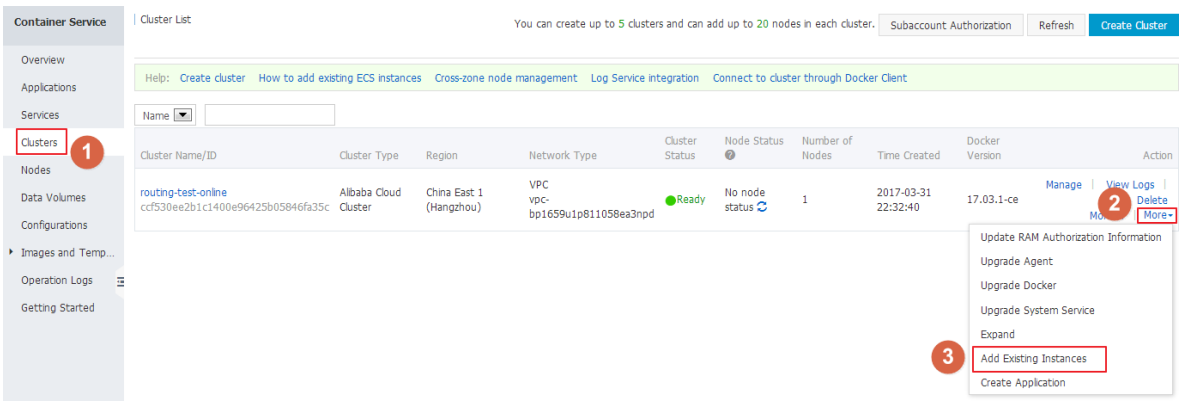
- The ECS instance to be added must be in the same region and use the same network type (Virtual Private Cloud (VPC)) as the cluster.
- When adding an existing ECS instance, make sure that your ECS instance has an Elastic IP (EIP) for the network type VPC, or the corresponding VPC has configured the NAT gateway. In short, make sure the corresponding node can access public network normally. Otherwise, the ECS instance fails to be added.
- The ECS instance to be added must be under the same account as the cluster.
- If you select to **manually add** the ECS instance, note that:
  - If you have already installed Docker on your ECS instance, the ECS instance may fail to be added. We recommend that you uninstall Docker and remove the Docker folders before adding the ECS instance by running the following command:

    Ubuntu: `apt-get remove -y docker-engine`, `rm -fr /etc/docker/ /var/lib/docker /etc/default/docker`

    CentOS: `yum remove -y docker-engine`, `rm -fr /etc/docker /var/lib/docker`
  - Container Service nodes have special requirements for the operating system of the ECS instance. We recommend that you use Ubuntu 14.04/16.04 or CentOS 7 as the operating system. We have strictly tested the stability and compatibility of these operating systems.

Procedure

1. Log on to the *Container Service console*.
2. Click Swarm > **Clusters** in the left-side navigation pane.
3. Click **More** at the right of the cluster that you want to add ECS instances and then select **Add Existing Instances** from the drop-down list.



4. Add ECS instances.

   The ECS instances displayed are filtered and synchronized from your ECS instance list according to the region and network type defined by the cluster.

   Add the ECS instances in the following ways:

   - Add ECS instances automatically.

     Note:
     As this method will reset the image and system disk of the ECS instance, proceed with caution. Create a snapshot to back up your data before adding the ECS instance. For information about how to create a snapshot, see *Create a snapshot*.

     1. Select the ECS instances you want to add to the cluster and click **Next Step**.

        You can add one or more ECS instances at a time.

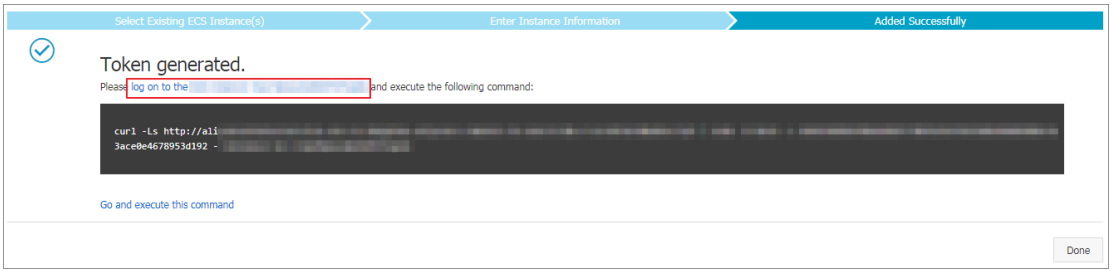
     2. Configure the instance information. Click **Next Step** and then click **Confirm** in the confirmation dialog box.
     3. Click **Finish**.

   - Manually add the ECS instance by running scripts on the ECS instance.

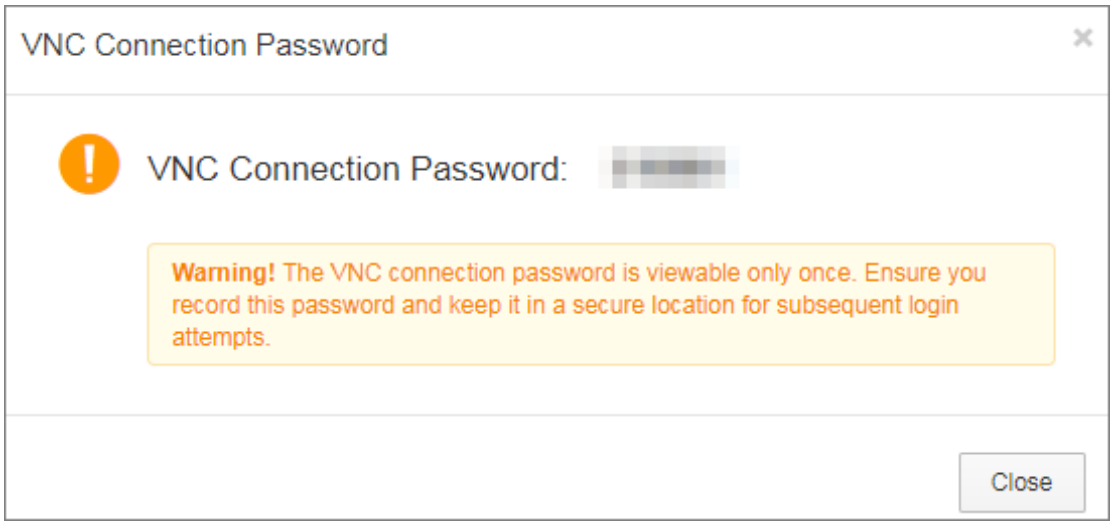
     1. Select **Manually Add**. Select an ECS instance, and then click **Next Step**.

        You can only add one ECS instance at a time.

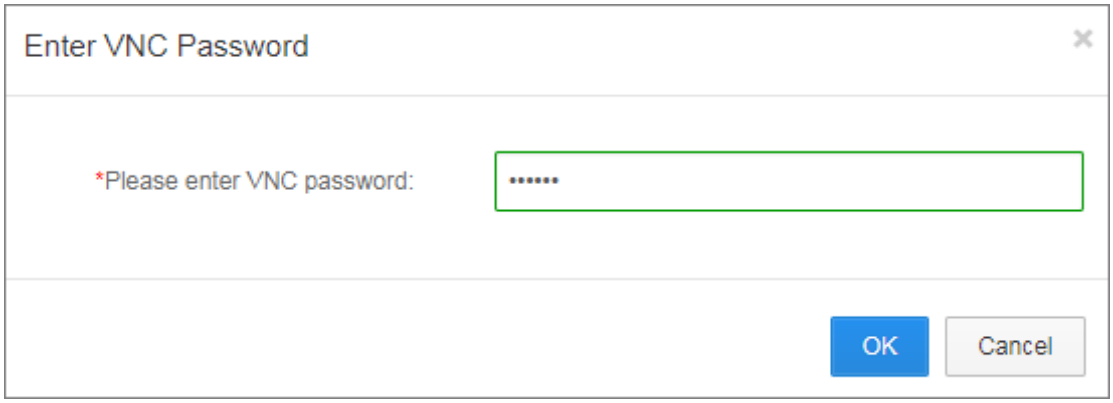
     2. Confirm the instance information and click **Next Step**.
     3. The scripts unique to this ECS instance are displayed. Click **log on to the ECS instance xxxxxx**.

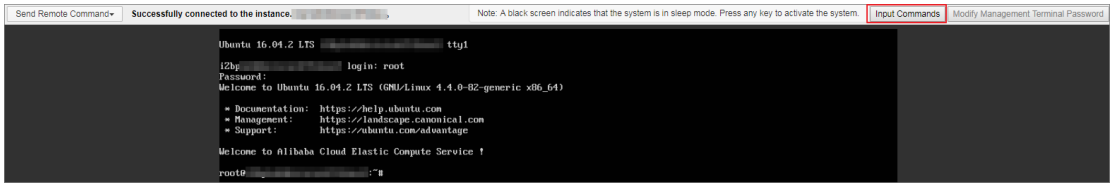
     4. The VNC connection password is displayed in the dialog box. Copy the password and click **Close**.



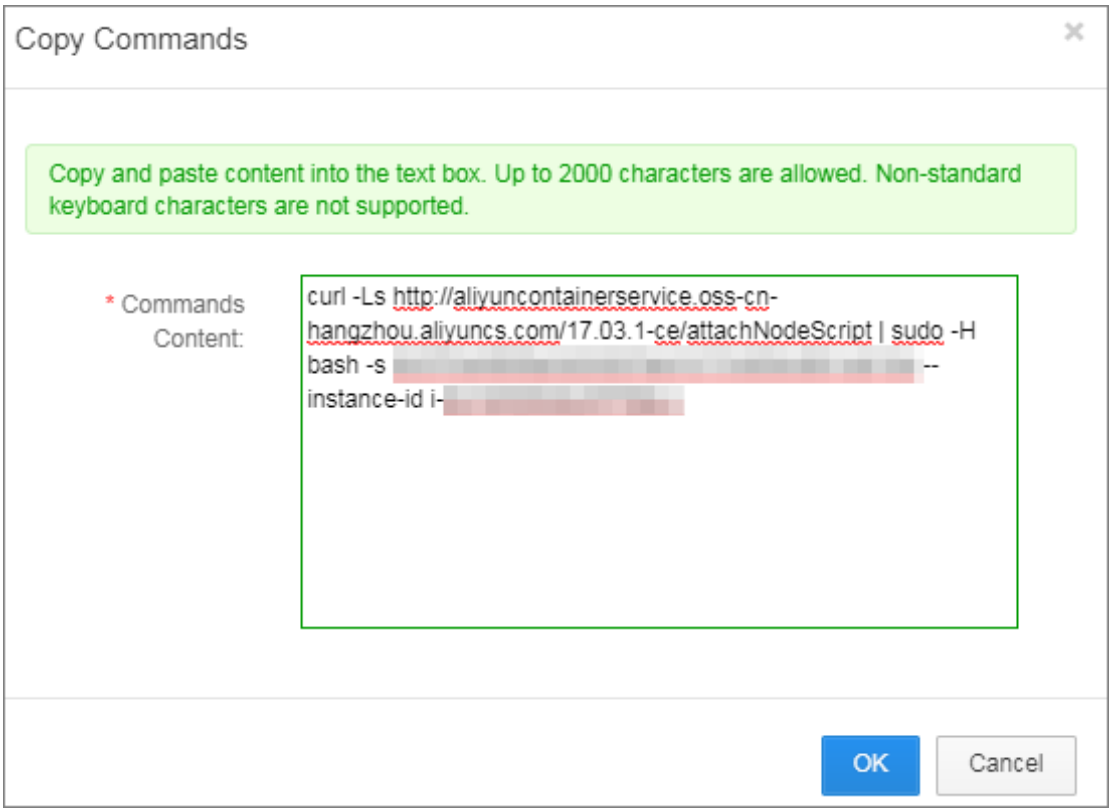
5. In the dialog box, enter the VNC connection password and click **OK**.



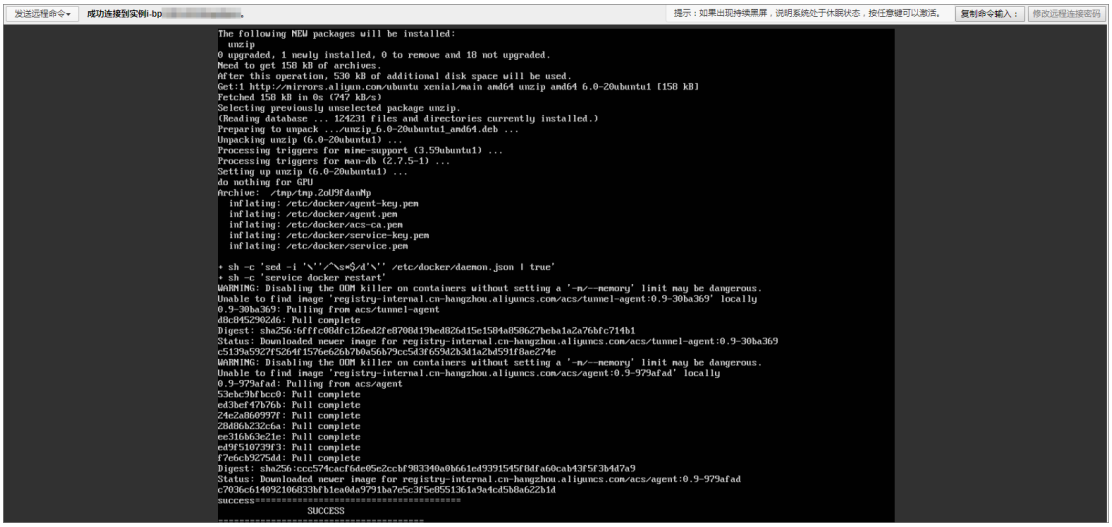
6. Enter the logon account (root) and password of the ECS instance, and press Enter to log on to the ECS instance.



7. Click **Input Commands**. Paste the preceding scripts into the dialog box, click **OK** and press Enter.

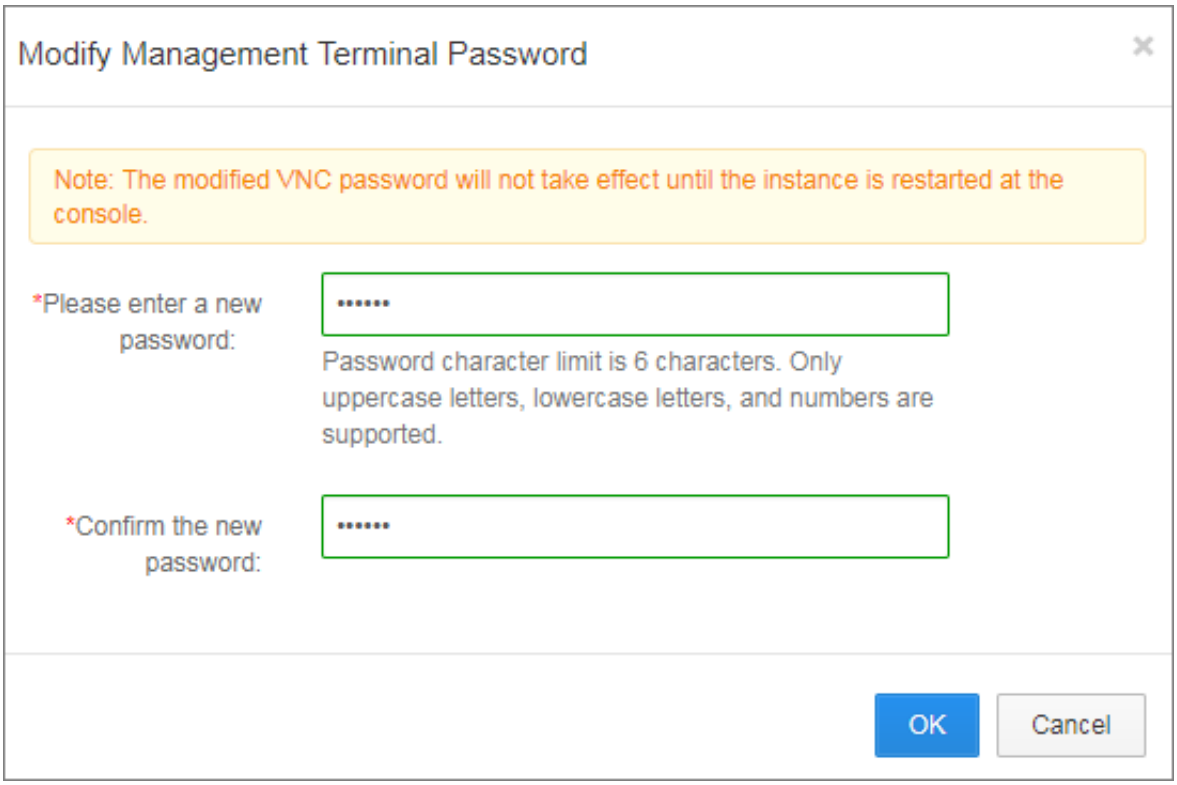


The system runs the scripts. Wait until the scripts are successfully run. A success message is displayed. The ECS instance is successfully added.



Related operation

You can modify the VNC connection password of the ECS instance in the remote terminal connection page. Click **Modify Management Terminal Password**, enter the new password and click **OK** in the dialog box.



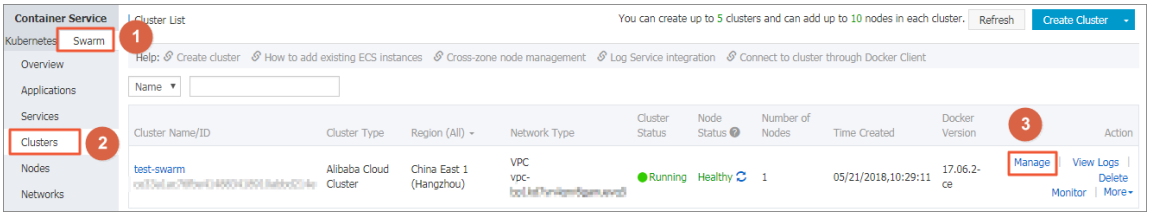
2.3 Download cluster certificate

Context

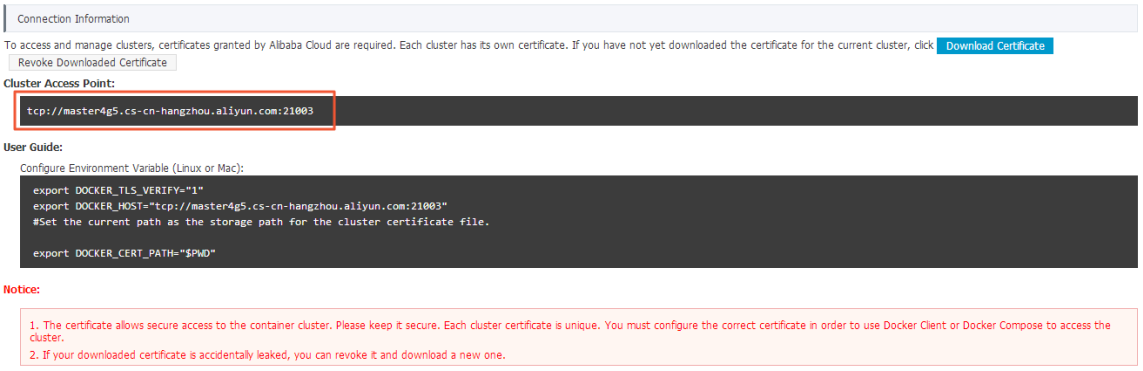
With the downloaded certificate, you can connect to the endpoint exposed from the cluster by using Docker Swarm API or Docker client. For more information, see *Connect to a cluster by using Docker tools*.

Procedure

1. Obtain the access address.
   a) Log on to the *Container Service console*.
   b) Log on to the *Container Service console*.
   c) Click **Clusters** in the left-side navigation pane. On the Cluster List page, click **Manage** at the right of a cluster.



   d) The cluster details page is displayed, showing the cluster connection information.



2. Download and save the TLS certificate.

   Configure a TLS certificate before you use the preceding access address to access the Docker cluster.

   Click **Download Certificate** in the cluster details page to download the TLS certificate. The certFiles.zip file is downloaded. certificate. The `certFiles.zip` file is downloaded. In the following example, the downloaded certificate is saved to the *`~/.acs/certs/ClusterName/`* directory. `ClusterName` indicates the name of your cluster. You can save the certificate to a different directory, but we recommend using the *`~/.acs/certs/ClusterName/`* directory for easy management.

   ```
   mkdir ~/.acs/certs/ClusterName/ #Replace ClusterName with your
   cluster name
    cd ~/.acs/certs/ClusterName/
    cp /path/to/certFiles.zip .
    unzip certFiles.zip
   ```

   The *`certFiles.zip`* file contains *`ca.pem`*, *`cert.pem`*, and *`key.pem`*.

2.4 Migrate a cluster

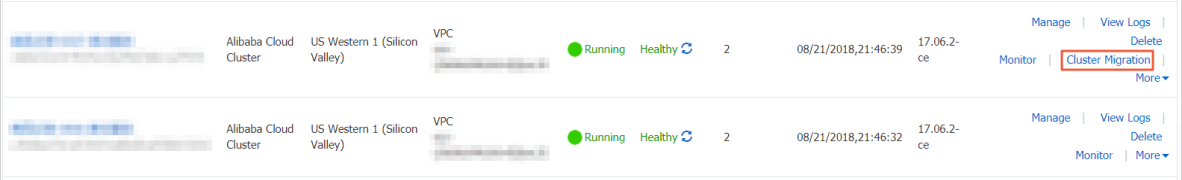
For a Swarm cluster created earlier, you can guarantee the performance and stability of the cluster by migrating the cluster.

Context

- The latest time for migrating a cluster is displayed through SMS, station message, or email . Complete the Swarm cluster migration before the latest time. The system automatically migrates the cluster if you do not migrate the cluster before the latest time.
- Cluster migration rebuilds connections from cluster nodes to the container server without affecting applications deployed in the cluster, nor adding or modifying any data. Make sure that you perform this operation during the low peak period of your business because unpredictable risks might still exist throughout the migration process.

Procedure

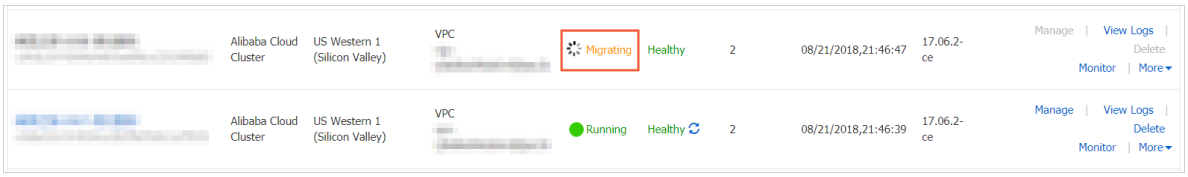
1. Log on to the *Container Service console*.
2. Under the Swarm menu, click **Clusters**.
3. Click **Cluster Migration** in the action column at the right of the cluster to be migrated.



4. Click **OK** in the **Prompt** dialog box.

> **Note:**
>
> During cluster migration:
>
> - Information query, deployment, upgrade, and other operations cannot be performed in the console.
> - The cluster cannot be connected to through the cluster access point API.
> - The data and application status in the cluster remain unchanged. Applications deployed on the cluster are still accessible.
> - The migration process takes about three minutes.

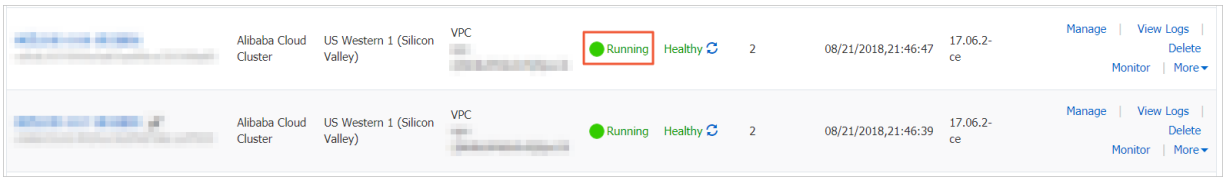
On the **Cluster List** page, **Migrating** is displayed in the **Cluster Status** column.



Result

After cluster migration is completed, on the **Cluster List** page, **Running** is displayed in the **Cluster Status** column.

> **Note:**
>
> - The cluster ID, access point address, and other attributes remain unchanged.
> - Please be sure to confirm that your business is running properly.
> - During the migration process, if you have any questions, please open a ticket in which you include the cluster ID and state whether your deployed applications are normal.


Alibaba Cloud Cluster
US Western 1 (Silicon Valley)
VPC
Running
Healthy
2
08/21/2018,21:46:47
17.06.2-ce
Manage | View Logs | Delete
Monitor | More
Alibaba Cloud Cluster
US Western 1 (Silicon Valley)
VPC
Running
Healthy
2
08/21/2018,21:46:39
17.06.2-ce
Manage | View Logs | Delete
Monitor | More

3 Nodes

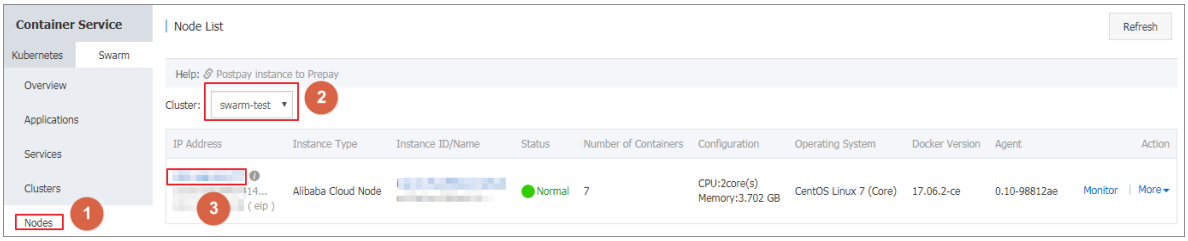
3.1 View containers running on a node

Context

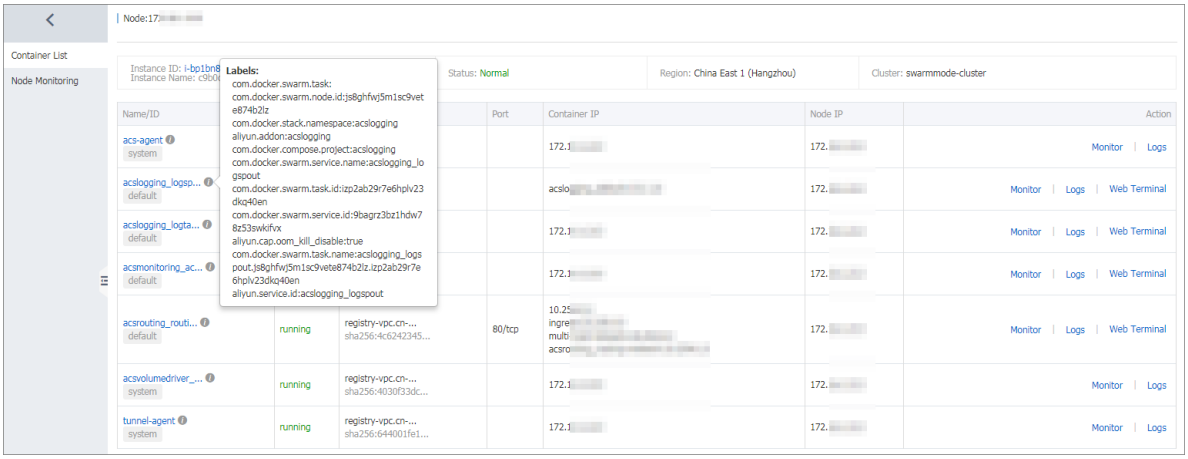
You can view containers running on a node on the Node List page.

Procedure

1. Log on to the *Container Service console*.
2. Click Swarm > **Nodes** in the left-side navigation pane.
3. On the Node List page, select a cluster from the Cluster drop-down list.
4. Click the node ID.



You can see the list of containers running on the node.



What's next

In the list, you can view the labels, images, the image SHA256 values, logs, and monitoring information of containers and perform operations on containers, including starting and stopping containers, deleting containers, and operating on containers on a remote terminal.

3.2 Update a node certificate

You can update a node certificate of a Swarm cluster to avoid node certificate expiration.

Prerequisites

1. You have created a swarm cluster, see *Create a cluster*.
2. Updating a node certificate reboots the node Docker Daemon. Make sure that containers on the node are all configured to restart automatically.

   Note:
   You can configure a container restart policy when creating an application. When you create an application by using an image, select the **Always** check box for **Restart**. When you create an application by using a template, configure a container restart policy in the template `restart : always`.
3. If a node certificate expires within 60 days, a prompt is displayed. You must timely update the node certificate.

Context

Each cluster node has a certificate used to access system control services. Each issued certificat e has a valid period. When the valid period of a certificate is about to expire, you must manually renew the certificate. Otherwise, the service of the node is affected.

Procedure

1. Log on to the *Container Service console*.
2. Under the Swarm menu, click **Nodes** in the left-side navigation pane. The certificate expiration information of each cluster node is displayed.

   Note:
   The certificate expiration time is displayed in the status column only if the node certificate expires within 60 days.
3. Select a node in the node list, and click **More** > **Update Certificate** on the right to reissue the node certificate.

   Note:
   We recommend that you upgrade the cluster agent to the latest version before updating the node certificate.

4. Optional: If the system prompts you to upgrade the cluster agent after you click **Update Certificate**, the current cluster agent does not support this feature. You need to upgrade the cluster agent to the new version first, see *Upgrade Agent*. If no prompt is displayed, go to the next step.
5. If no prompt is displayed or the cluster agent is updated, click **Update Certificate**. Confirm updating information and then update the node cluster certificate.

   Note:

   - When the node certificate update is completed, the Docker Daemon node is automatically restarted about 1 minute later.
   - To guarantee that containers on the node can automatically restart, make sure that an automatic restart policy is configured.
6. After the cluster node certificate is updated, the node certificate information is no longer displayed.

4 Service orchestrations

4.1 routing

The routing label configures the access domain name of a service.

Format:

```
aliyun.routing.port_$container_port: [http://]$domain|$domain_prefix[:
$context_path]
```

Field description:

- `$container_port`: container port. **Note:** This is not the host port.
- `$domain`: domain name. Enter a domain name.
- `$domain_prefix`: domain name prefix. If you enter a domain name prefix, Container Service provides you with a test domain name and the domain name suffix is `.<cluster_id>.<region_id>.alicontainer.com`.
- `$context_path`: requested service path. You can select services according to the requested path.

Domain name selection:

- If the HTTP protocol is used to expose the service, you can use the internal domain name (the top-level domain is `alicontainer.com`) provided by Container Service for testing, or use your own domain name.
- If the HTTPS protocol is used, you can use only your own domain name. For example, `www.example.com`. You must modify the DNS settings to assign the domain name to the Server Load Balancer service provided by the container cluster.

Format requirements of the label statement:

- Container Service allocates a subdomain name to each cluster, and you only need to provide the domain name prefix to bind the internal domain name. The domain name prefix only indicates a domain name level and cannot be separated with periods (.).
- If you do not specify `scheme`, the HTTP protocol is used by default.
- The length of the domain name cannot exceed 128 characters. The length of the context root cannot exceed 128 characters.
- When you bind multiple domain names to the service, use semicolons (;) to separate them.

- A backend service can have multiple ports. These ports are exposed by the container. A port can only be assigned one label. Therefore, a service with multiple ports must be assigned multiple labels.

Example:

Use the routing label.

Bind the internal domain name`wordpress.<cluster_id>.<region_id>.alicontainer.com` provided by Container Service and your own domain name `http://wp.sample.com/context` to port 80 of the Web service.

```
web:
  image: wordpress:4.2
  links:
    - db:mysql
  labels:
    aliyun.routing.port_80: wordpress;http://wp.sample.com/context
db:
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
```

The internal domain name that you finally get is `wordpress.cd3dfe269056e4543acbec5e19b01c074.cn-beijing.alicontainer.com`.

After starting the Web service, you can access the corresponding Web services by using the URL: `http://wordpress.cd3dfe269056e4543acbec5e19b01c074.cn-beijing.alicontainer.com` or`http://wp.sample.com/context`.

To support the HTTPS service, upload the HTTPS certificate by using the Server Load Balancer console on the Alibaba Cloud website, and then bind the corresponding cluster to access the Server Load Balancer terminal.

routing.session_sticky

By using this feature, you can determine whether to maintain session sticky (session persistence) when you set the routing for a routing request. With session persistence, during the session, each request is routed to the same backend container instead of being randomly routed to different containers.

Note:

- The setting takes effect only when you have configured `aliyun.routing.port_$containr_port`.

- Simple routing session persistence is based on the Cookie mechanism. By default, the maximum expiration time of Cookie is 8 hours and the idle expiration time is 30 minutes.
- Simple routing session persistence is enabled by default.

The setting methods are as follows:

- Enable session persistence

  ```
  aliyun.routing.session_sticky: true
  ```

- Disable session persistence

  ```
  aliyun.routing.session_sticky: false
  ```

Example of a template orchestration file:

```
web:
  image: wordpress:4.2
  links:
    - db:mysql
  labels:
    aliyun.routing.port_80: wordpress;http://wp.sample.com/context
    aliyun.routing.session_sticky: true
db:
  image: mysql
  environment:
    - MYSQL_ROOT_PASSWORD=password
```

5 Data volumes

6 DevOps

6.1 Jenkins-based continuous delivery

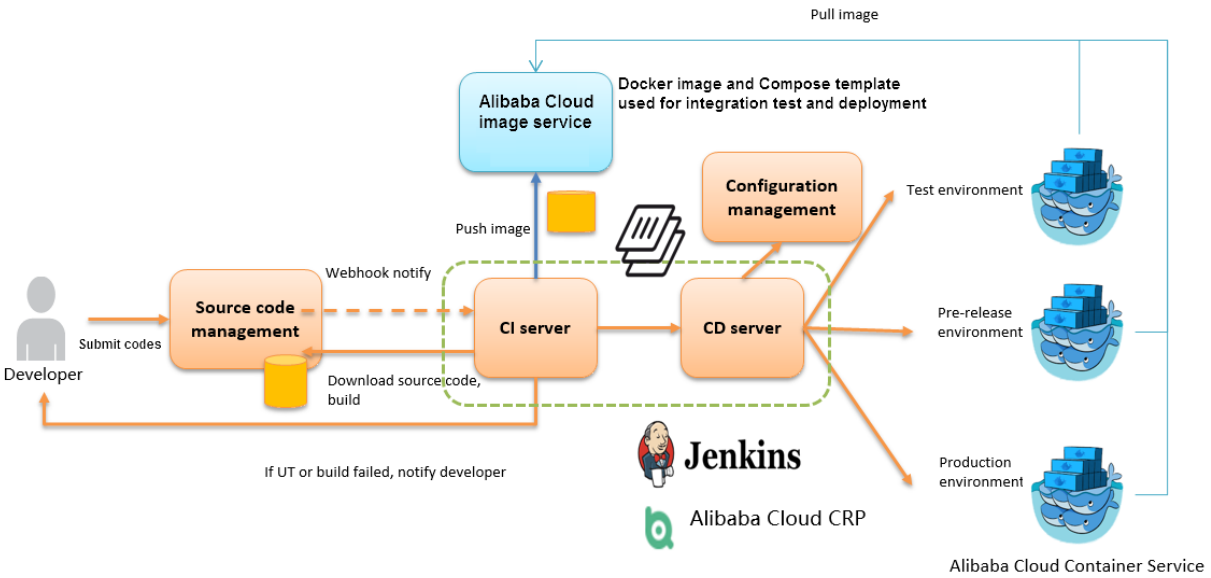
As an important step in agile development, continuous integration aims to maintain high quality while accelerating product iteration. Every time codes are updated, an automated test is performed to test the codes and function validity. The codes can only be delivered and deployed after they pass the automated test. This document mainly introduces how to integrate Jenkins, one of the most popular continuous integration tools, with Alibaba Cloud Container Service to realize automated test and image building push.

The following example demonstrates how to perform automated test and build a Docker image by using Alibaba Cloud Container Service Jenkins, which realizes high-quality continuous integration.

Background information

Every time codes are submitted to nodejs project in GitHub, Alibaba Cloud Container Service Jenkins will automatically trigger a unit test. If the test is successful, Jenkins continues to build images and then pushes them to a target image repository. Finally, Jenkins notifies you of the results by email.

A general process is as follows.



Slave-nodejs is a slave node used for unit test and building and pushing the image.

Jenkins introduction

Jenkins is an open-sourced continuous integration tool developed on Java. It monitors and triggers continuously repeated work and supports expansion of multiple platforms and plug-ins. Jenkins is an open-sourced tool featuring easy installation and interface-based management. It uses job to describe every work step, and node is a project execution environment. The master node is a default execution environment of a Jenkins job and also the installation environment for Jenkins applications.

Master/slave

Master/slave is equivalent to the server/agent concept. A master provides Web interface with which you manage the job and slave. The job can run on the master or be assigned to the slave . One master can be associated with several slaves to serve different jobs or different configurat ions of the same job.

Several slaves can be configured to prepare a separate test and building environment for different projects.

Note:

The Jenkins job and project mentioned in this document all refer to a build unit of Jenkins, namely, an execution unit.

Step 1 Deploy Jenkins applications and slave nodes

The building and testing of different applications need different dependencies. The best practice is to use different slave containers with corresponding runtime dependencies and tools to perform the test and building. By using the slave images and sample templates provided by Alibaba Cloud Container Service for different environments such as Python, Node.js, and Go, you can quickly and easily generate Jenkins applications and various slave nodes, configure node information in Jenkins applications, and specify the execution nodes in the build projects so as to implement the entire continuous integration process.

Note:

For images provided by Alibaba Cloud Container Service for developing slave nodes, see *https://github.com/AliyunContainerService/jenkins-slaves*.

1.1 Create a Jenkins orchestration template

Create a template and create the orchestration based on the following contents.

The labels supported by Alibaba Cloud Container Service Jenkins master are: 1.651.3, 2.19.2, and 2.32.2.

Note:

For how to create an orchestration template, see *#unique_21*.

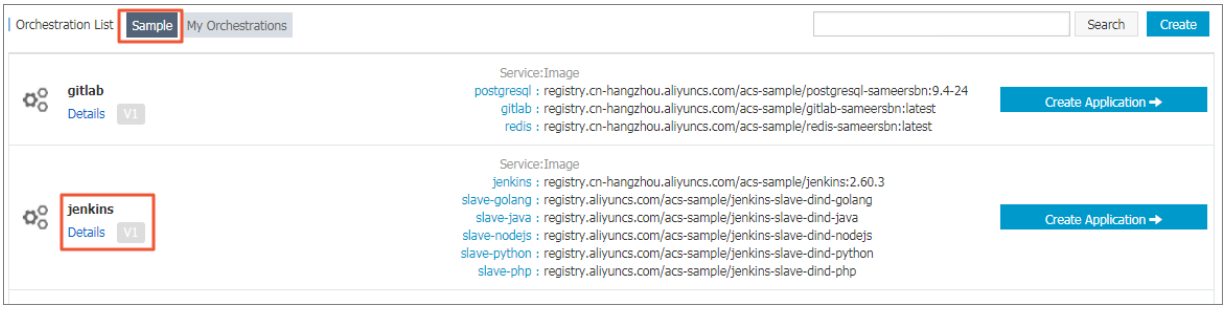
```
jenkins:
    image: 'registry.aliyuncs.com/acs-sample/jenkins:1.651.3'
    volumes:
        - /var/lib/docker/jenkins:/var/jenkins_home
    restart: always
    labels:
        aliyun.scale: '1'
        aliyun.probe.url: 'tcp://container:8080'
        aliyun.probe.initial_delay_seconds: '10'
        aliyun.routing.port_8080: jenkins
    links:
        - slave-nodejs
slave-nodejs:
    image: 'registry.aliyuncs.com/acs-sample/jenkins-slave-dind-nodejs
'
    volumes:
        - /var/run/docker.sock:/var/run/docker.sock
    restart: always
    labels:
        aliyun.scale: '1'
```

1.2 Use the template to create Jenkins application and slave node

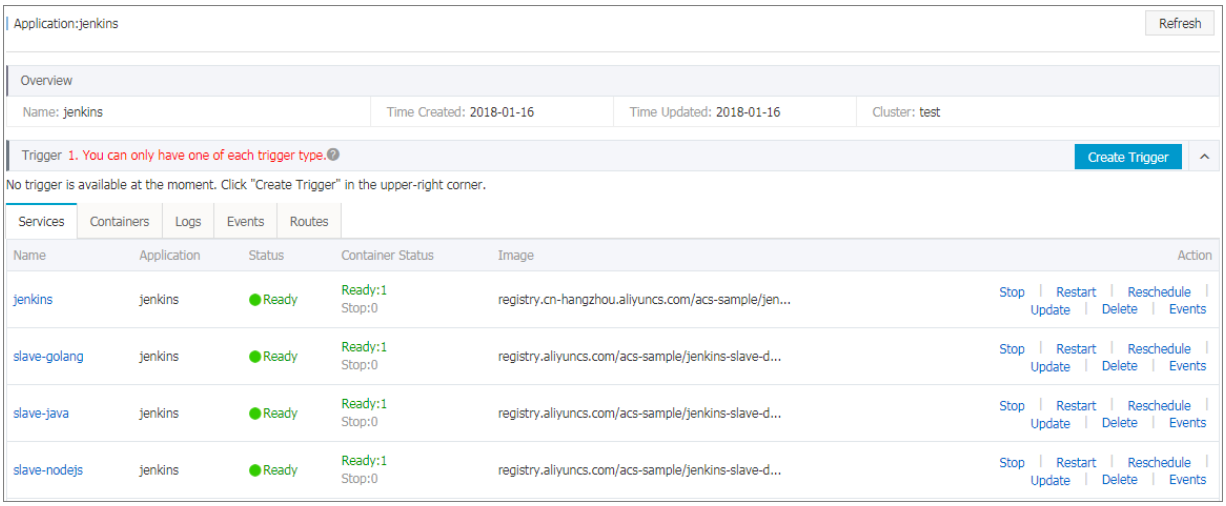
Use the orchestration template created in the preceding section or the Jenkins sample template provided by Alibaba Cloud Container Service to create the Jenkins application and slave node.

Note:

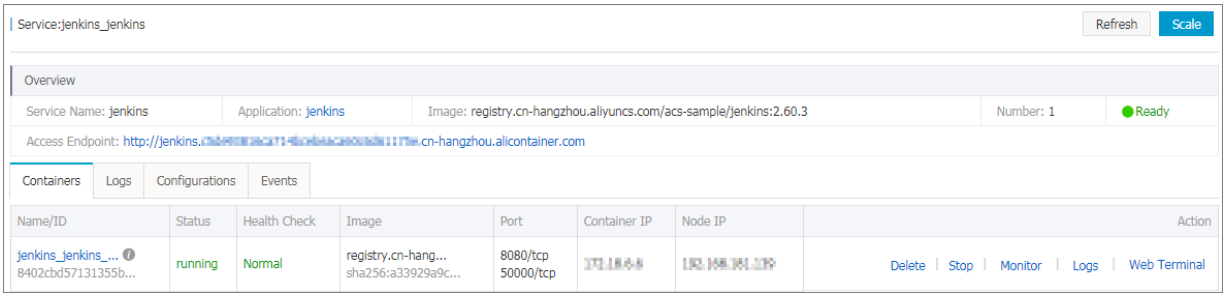
For how to create an application by using an orchestration template, see *Create an application* .



After a successful creation, the Jenkins application and slave node are displayed in the service list.



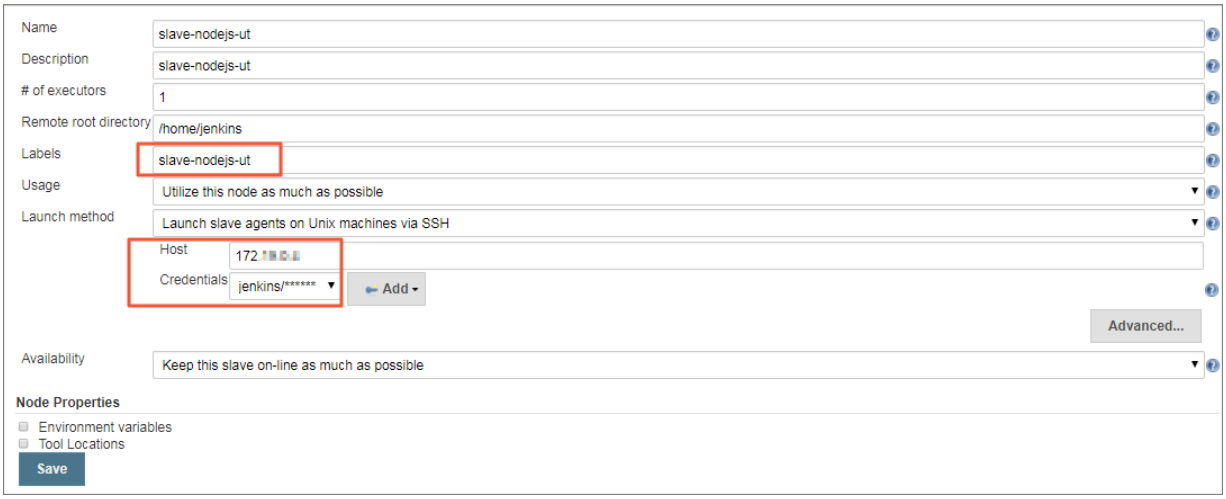
Open the access endpoint provided by Container Service to use the deployed Jenkins application.



Step 2 Realize automated test and automated build and push of image

2.1 Configure the slave container as the slave node of the Jenkins application

Open the Jenkins application. Click Manage Jenkins in the left-side navigation pane. Click Manage Nodes on the right pane. Click New Node in the left-side navigation pane. Enter the node name and then click OK. Then, complete the parameters as follows.

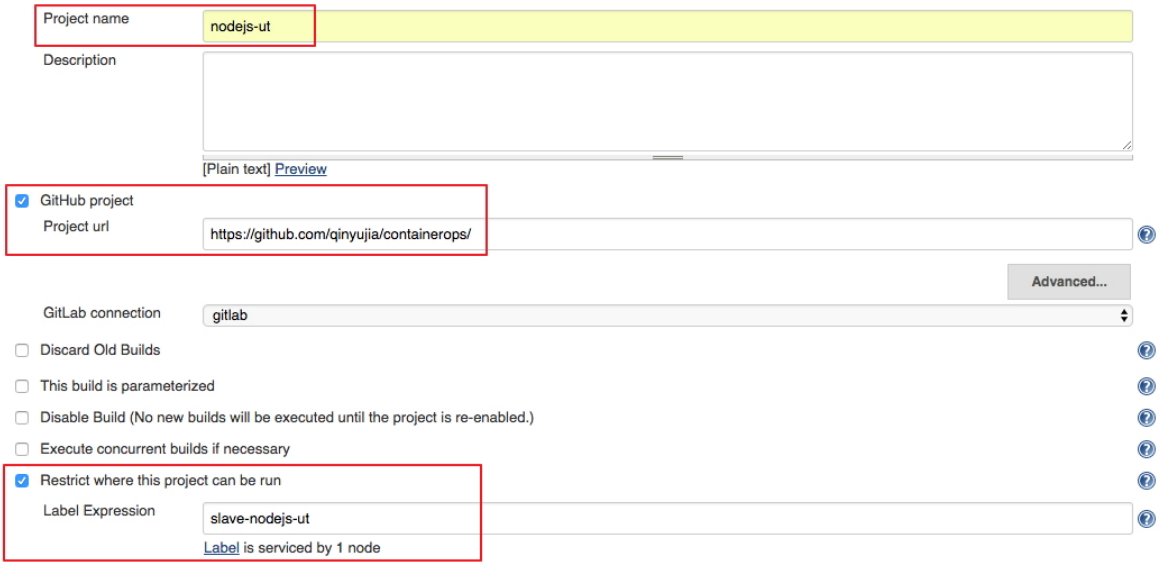


Note:

- Label is the unique identifier of the slave.
- The slave container and Jenkins container run on the Alibaba Cloud platform at the same time . Therefore, enter a container node IP address that is inaccessible to the Internet to isolate the test environment.
- When adding the credentials, use the jenkins account and password (the initial password is jenkins) in Dockerfile for the creation of the slave-nodejs image. The image Dockerfile address is *jenkins-slave-dind-nodejs*.

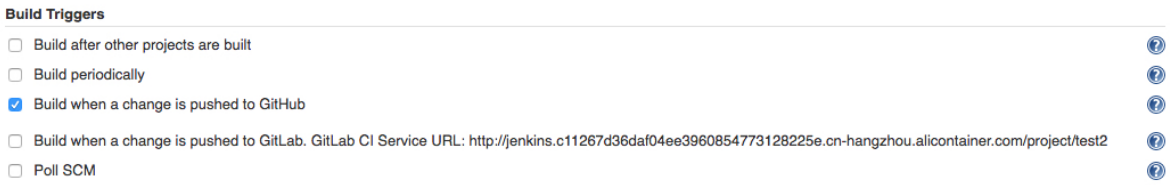
2.2 Create a project to implement automated test

1. Go back to the Jenkins home page. Click New Item in the left-side navigation pane. Enter the item name, select Freestyle project, and then click OK.
2. Enter the project name and select a node for running the project. In this example, enter the slave-nodejs-ut node prepared in the preceding section.



3. Configure the source code management and code branch. In this example, use GitHub to manage source codes.

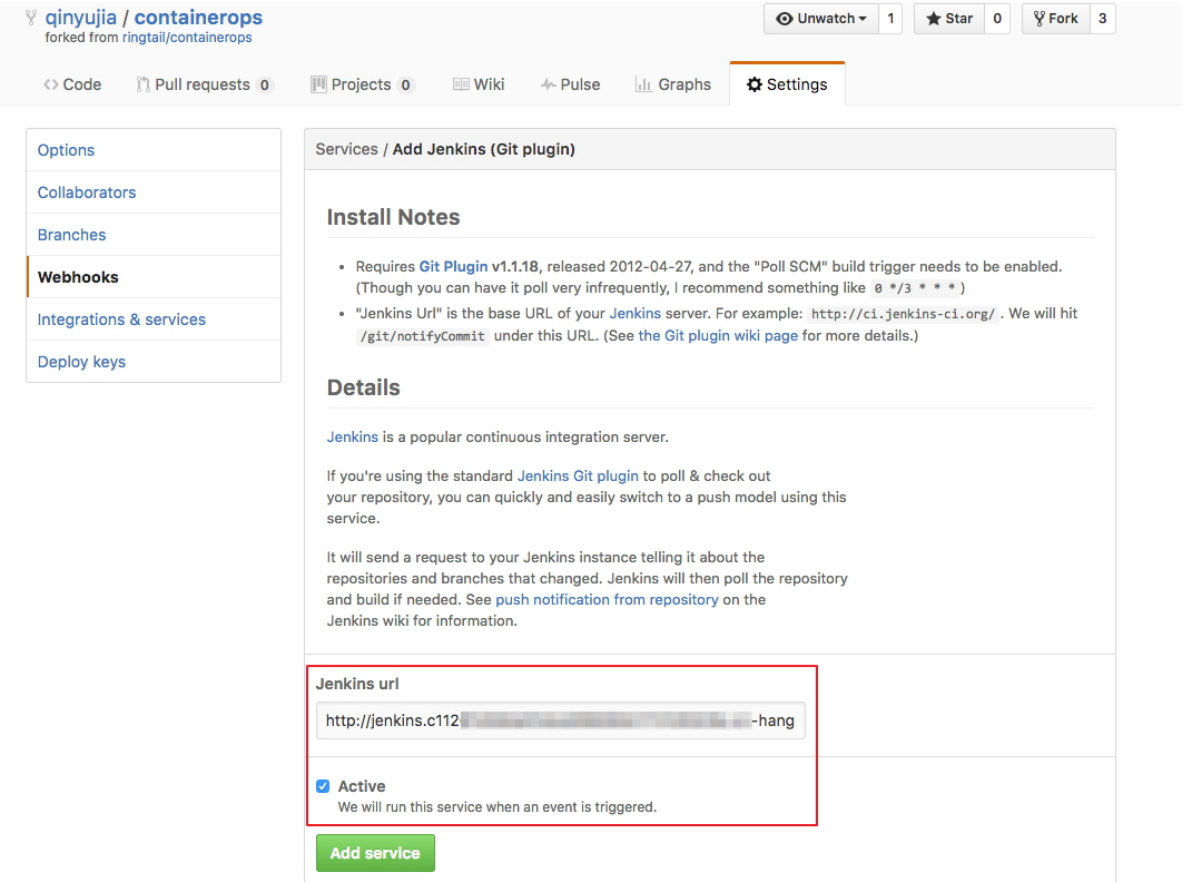
4. Configure the build trigger. In this example, automatically trigger project execution by combining GitHub Webhooks & services.



5. Add the Jenkins service hook to GitHub to implement automatic triggering.

On the GitHub project home page, click the **Settings**. Click **Webhooks & services**, click **Add Service**, and then select **Jenkins(Git plugin)** from the drop list. In the dialog box of **`Jenkins hook url`**, enter `${Jenkins IP}/github-webhook/`. For example:

```
http://jenkins.cd****************.cn-beijing.alicontainer.com/github-webhook/
```



6. Add a build step of Execute shell type and write shell scripts to perform the test.

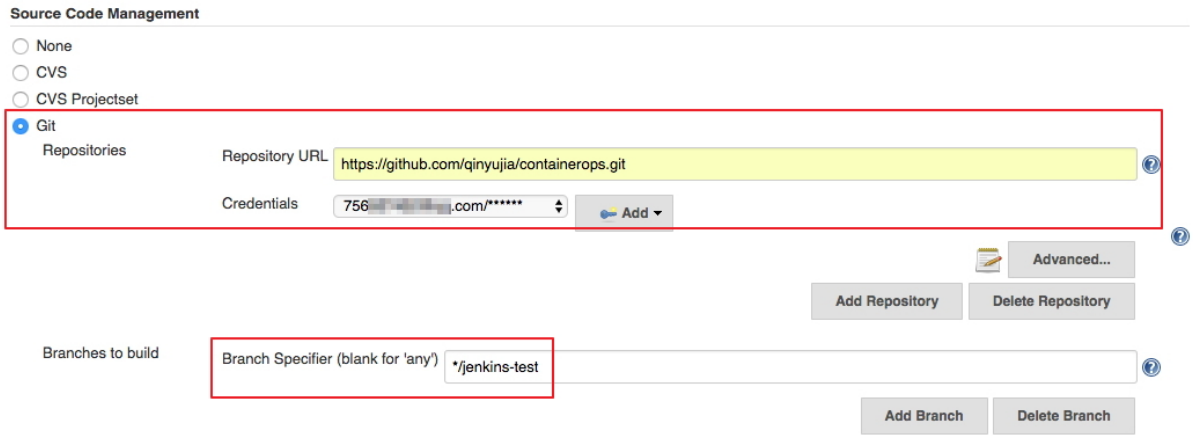
The commands in this example are as follows:

```
pwd
 ls
 cd chapter2
```

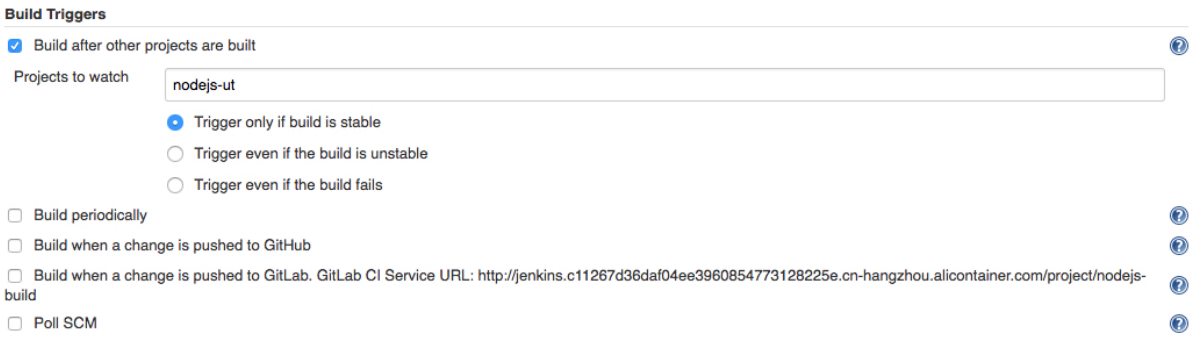
```
npm test
```

SVN source code example:

Select **Subversion** in Source Code Management and enter the SVN repository address in the **`Repository URL`** field (if the Jenkins master and SVN server are in different time zones, add @HEAD at the end of the repository address). Add the username and password of the SVN server in **`Credentials`** .



Configure the build trigger. In this example, Post-commit hook is used to automatically trigger the project execution. Enter your configured token in **`Token Name`** .



Log on to the SVN server. Create a *`post-commit`* file in the *`hooks`* directory of the code repository (svn-java-demo).

```
cd /home/svn/svn-java-demo/hooks
cp post-commit.tmpl post-commit
chmod 755 post-commit
```

Add the curl -u ${Jenkins_account}:${password}

```
${Jenkins_url}/job/svn/build?
```

```
        token=${token}  command
```

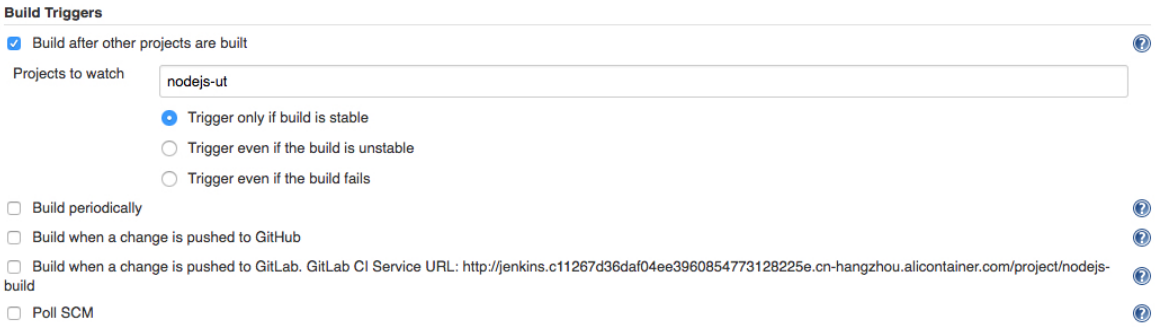
in the <g id="1">post-commit</g> file. For example:

```
curl -u test:test
     http://127.0.0.1:8080/jenkins/job/svn/build?token=qinyujia
```

.

2.3 Create a project to automatically build and push images

1. Go back to the Jenkins home page. Click New Item in the left-side navigation pane. Enter the item name, select Freestyle project, and then click OK.
2. Enter the project name and select a node for running the project. In this example, enter the slave-nodejs-ut node prepared in the preceding section.
3. Configure the source code management and code branch. In this example, use GitHub to manage source codes.
4. Add the following trigger and set to automatically build the image only after the unit test is successful.



5. Write the shell script for building and pushing images.

The commands in this example are as follows:

```
cd chapter2
 sudo docker build -t registry.aliyuncs.com/qinyujia-test/nodejs-
demo .
 sudo docker login -u ${yourAccount} -p ${yourPassword} registry.
aliyuncs.com
```

```
sudo docker push registry.aliyuncs.com/qinyujia-test/nodejs-demo
```

Step 3 Automatically redeploy the application

3.1 Deploy the application for the first time

Use the orchestration template to deploy the image created in step 2.3 to Container Service and create the nodejs-demo application.

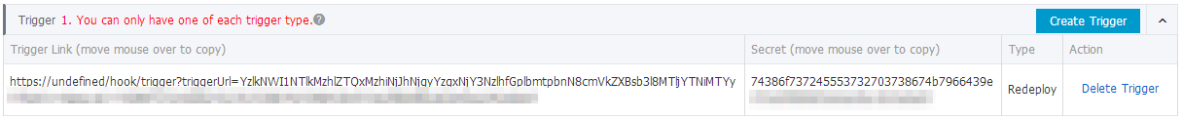
Example:

```
express:
image: 'registry.aliyuncs.com/qinyujia-test/nodejs-demo'
expose:
    - '22'
    - '3000'
restart: always
labels:
    aliyun.routing.port_3000: express
```

3.2 Automatic redeployment

1. Select the created application **nodejs-demo** and create the trigger.

> **Note:**
> For how to create a trigger, see *Triggers*.



2. Add a line to the shell script in 2.3. The address is the trigger link of the created trigger.

```
curl ‘https://cs.console.aliyun.com/hook/trigger?triggerUrl=***==&
secret=***’
```

3. Change the command in the example of 2.3 as follows:

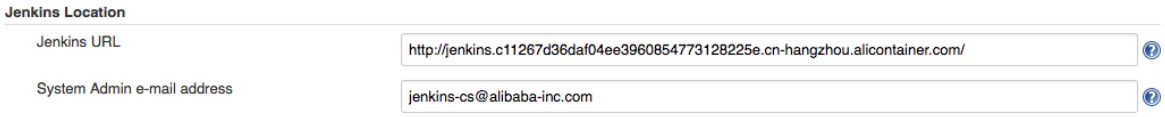
```
 cd chapter2
 sudo docker build -t registry.aliyuncs.com/qinyujia-test/nodejs-
demo .
 sudo docker login -u ${yourAccount} -p ${yourPassword} registry.
aliyuncs.com
 sudo docker push registry.aliyuncs.com/qinyujia-test/nodejs-demo
 curl ‘https://cs.console.aliyun.com/hook/trigger?triggerUrl=***==&
secret=***’
```

After pushing the image, Jenkins automatically triggers the redeployment of the nodejs-demo application.

Step 4 Configure email notification of the results

To send the unit test or image building results to relevant developers or project execution initiators by email, perform the following configurations:

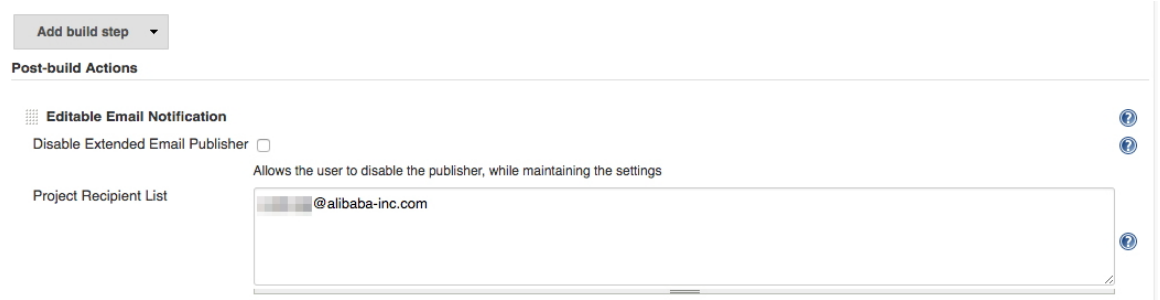
1. On the Jenkins homepage, click Manage Jenkins > Configure System, and configure the Jenkins system administrator email.



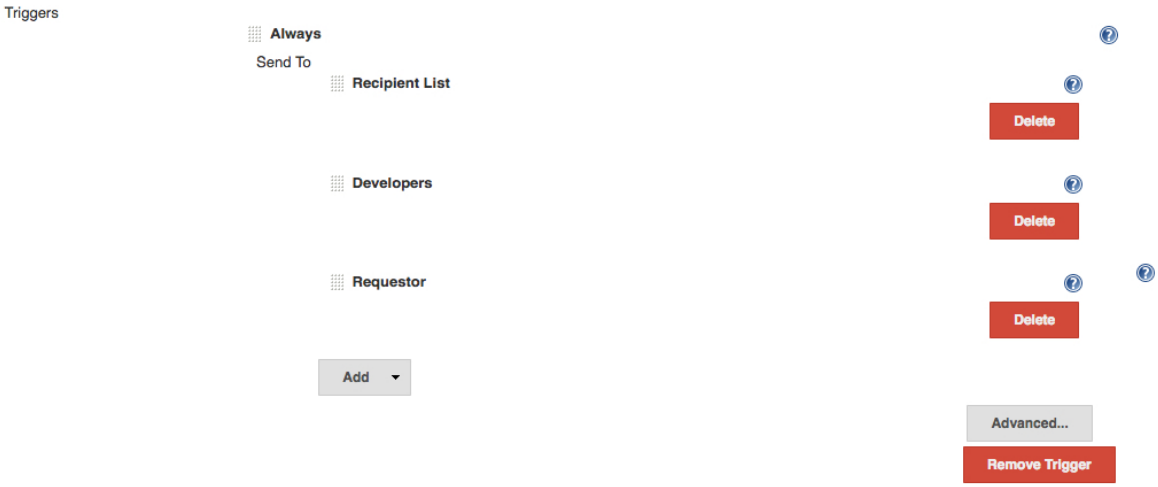
2. Install the Extended Email Notification plug-in, configure the SMTP server and other relevant information, and then set the default email recipient list, as shown in the following figure:

The preceding example shows the parameter settings of the Jenkins application system. The following example shows the relevant configurations for Jenkins projects whose results are to be pushed by email.

3. Add post-building steps in the Jenkins project, select Editable Email Notification and enter the email recipient list.



4. Add a trigger to send emails.


Triggers
Always
Send To
Recipient List
Delete
Developers
Delete
Requestor
Delete
Add
Advanced...
Remove Trigger

7 Service discovery and load balancing

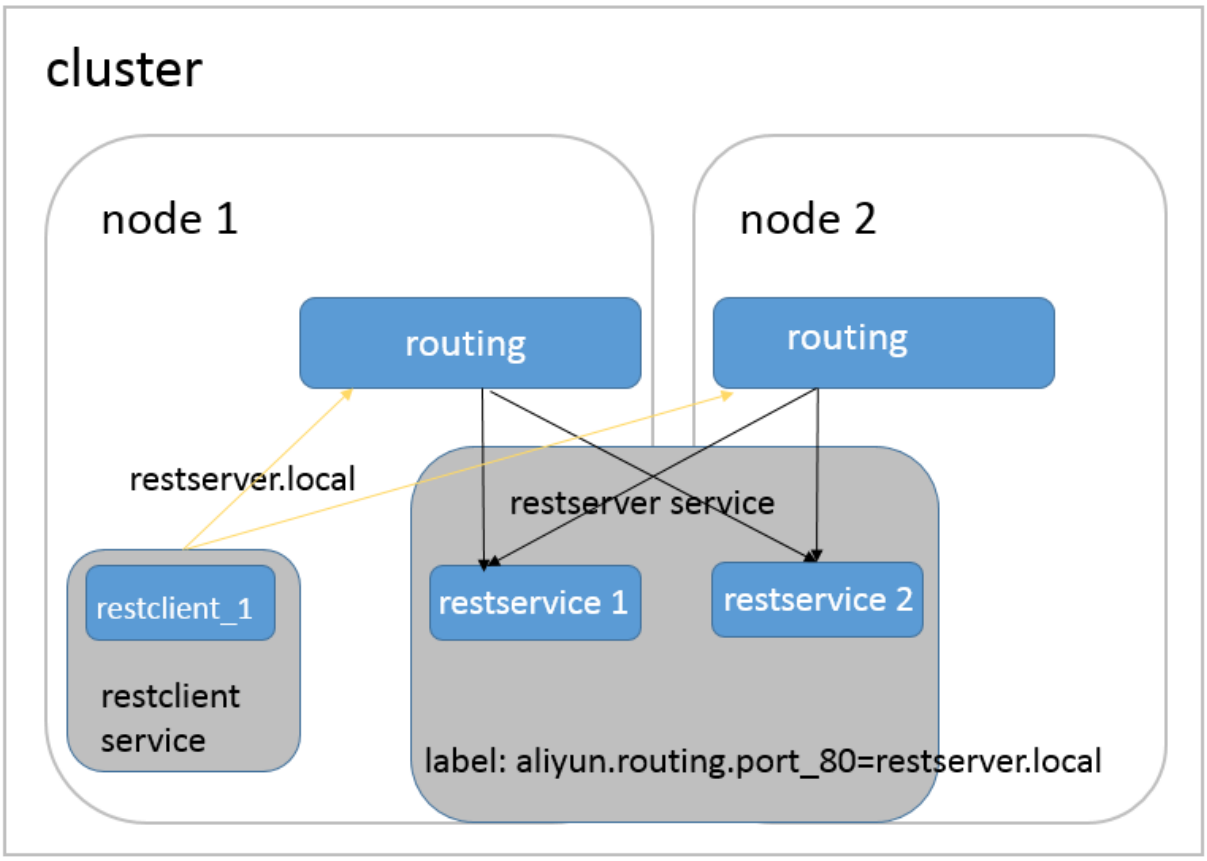
7.1 Routing and Server Load Balancer between services in a cluster

Container Service can expose the HTTP service based on domain names by using acsrouting, and work with health check to enable the automatic Server Load Balancer and service discovery. When one container malfunctions, routing will automatically remove the container that failed the health check from the backend, which achieves the automatic service discovery. However, in this way, the service is exposed to the Internet.

Then, how can automatic service discovery and Server Load Balancer be achieved between services in a cluster by using this method? The routing container of Alibaba Cloud Container Service has the function of Server Load Balancer. Use the domain name ending with `.local` to make the container can only be accessed by the other containers in the cluster, and then work with the external_links label to implement the inter-service discovery and Server Load Balancer in the cluster.

Implementation principle

1. Docker version later than 1.10 supports alias resolution in the container. In the restservice container that depends and loads on the restserver.local, the `restserver.local` domain name resolves the address of the routing container. When the restclient service initiates a request, the HTTP request is forwarded to the routing container, with `HOST` as the request header of restserver.local.
2. Routing container monitors the health status of the containers configured with `aliyun.routing.port_xxx: restserver.local` label and mounts the status to the backend of HAProxy. When HAProxy receives the HTTP request with the `restserver.local` HOST header, the request can be forwarded to the corresponding container.



Advantages

- Compared with the DNS-based method using link or hostname, the inconsistent handling of DNS cache by different clients will delay service discovery, and the DNS solution which only includes round robin cannot meet the requirements of microservice scenarios.
- Compared with other microservice discovery solutions, this solution provides a mechanism to achieve unrelated service discovery and Server Load Balancer, which can be used without any modification on the server side or client application.
- In decoupling service lifecycle, every microservice can adopt a Docker Compose template for independent deployment and update. Only a virtual domain name is required to achieve dynamic mutual binding.

Orchestration example

In the following orchestration example, add the `aliyun.routing.port_80:restserver.local` label to the restserver service to make sure only the containers in the cluster can access this domain name. Then, configure `external_links` for the restclient service, pointing to the

restserver.local domain name. The restclient service can use this domain name to access the restserver service, and work with health check to implement automatic service discovery.

```
restserver: # Simulate the rest service.
  image: nginx
  labels:
    aliyun.routing.port_80: restserver.local # Use the local domain
 name and only the containers in the cluster can access this domain
name.
    aliyun.scale: "2" # Expand two instances to simulate the Server
Load Balancer.
     aliyun.probe.url: "http://container:80" # Define the container
health check policy as http and the port as 80.
    aliyun.probe.initial_delay_seconds: "2" # The health check starts
two seconds after the container is started.
     aliyun.probe.timeout_seconds: "2" # The timeout for health check
. A container is considered as unhealthy if no result is returned in
two seconds.
restclient: # Simulate the rest service consumer.
  image: registry.aliyuncs.com/acs-sample/alpine:3.3
  command: "sh -c 'apk update; apk add curl; while true; do curl --
head restserver.local; sleep 1; done'" # Access the rest service and
test the Server Load Balancer.

  tty: true
  external_links:
      - "restserver.local" # Specify the link service domain name.
Make sure that you set external_links. Otherwise, the access fails.
```

The following restclient service logs show that the HTTP request of restclient curl is routed to the containers of different rest services. The container ID is `053cb232fdfbcb5405ff791650a0746ab77f26cce74fea2320075c2af55c975f` and `b8c36abca525ac7fb02d2a9fcaba8d36641447a774ea956cd93068419f17ee3f`.

```
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066803626Z
Server: nginx/1.11.1
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066814507Z
Date: Fri, 01 Jul 2016 06:43:49 GMT
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066821392Z
Content-Type: text/html
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066829291Z
Content-Length: 612
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066835259Z
Last-Modified: Tue, 31 May 2016 14:40:22 GMT
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066841201Z
ETag: "574da256-264"
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066847245Z
Accept-Ranges: bytes
internal-loadbalance_restclient_1 | 2016-07-01T06:43:49.066853137Z
 Set-Cookie: CONTAINERID=053cb232fdfbcb5405ff791650a0746ab77f26cc
e74fea2320075c2af55c975f; path=/
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.080502413Z
HTTP/1.1 200 OK
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082548154Z
Server: nginx/1.11.1
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082559109Z
Date: Fri, 01 Jul 2016 06:43:50 GMT
```

```
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082589299Z
Content-Type: text/html
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082596541Z
Content-Length: 612
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082602580Z
Last-Modified: Tue, 31 May 2016 14:40:22 GMT
internal-loadbalance_restclient_1  2016-07-01T06:43:50.082608807Z ETag
: "574da256-264"
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082614780Z
Accept-Ranges: bytes
internal-loadbalance_restclient_1 | 2016-07-01T06:43:50.082621152Z
 Set-Cookie: CONTAINERID=b8c36abca525ac7fb02d2a9fcaba8d36641447a7
74ea956cd93068419f17ee3f; path=/
```