Alibaba Cloud Container Registry

User Guide

Issue: 20200702

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 contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, 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 this document 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.

Document conventions

Style	Description	Example
	A danger notice 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.
	A warning notice 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 restart an instance.
	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	 **Notice:** If the weight is set to 0, the server no longer receives new requests.
	A note indicates supplemental instructions, best practices, tips, and other content.	 **Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings** > **Network** > **Set network type**.
Bold	Bold formatting is used for buttons, menus, page names, and other UI elements.	Click **OK**.
Courier font	Courier font is used for commands.	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`

Style	Description	Example
{} or {a|b}	This format is used for a required value, where only one item can be selected.	switch {active|stand}

Contents

1 Container Registry Default Instance Edition

1.1 Build a repository and images

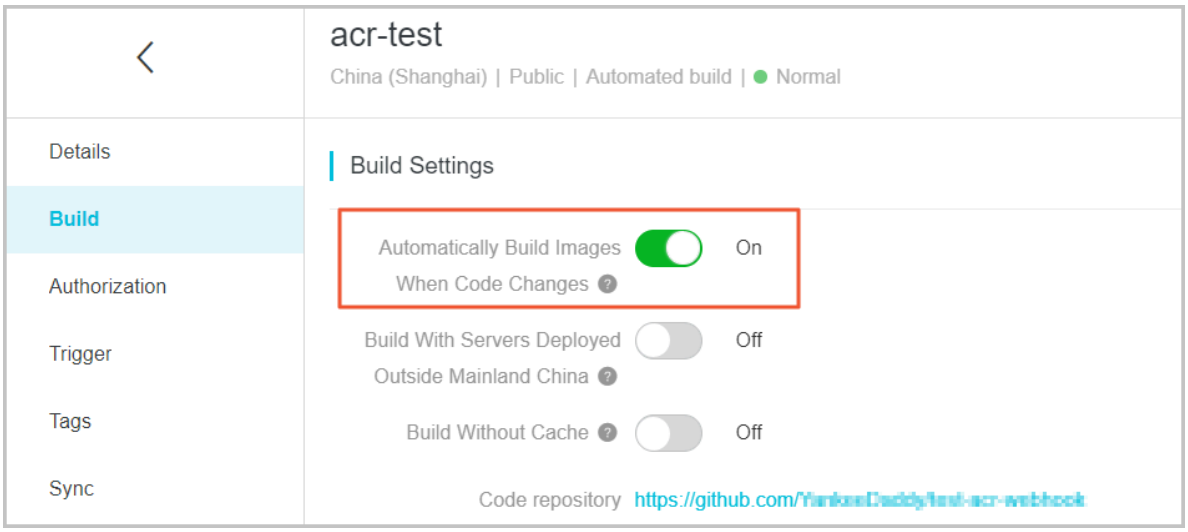
If you do not have high requirements on the number of namespaces, private repositories, and build rules, we recommend that you use Container Registry Default Instance that supports base images. This topic describes how to use Container Registry Default Instance to create a repository, set build rules, and build images.

Features

- Container Registry supports automatically triggering an image build when the code changes.

  After you switch on **Automatically Build Images When Code Changes**, an automatic image build is triggered each time you commit the code. This lowers the manual workload.

  1. Log on to the Container Registry console. In the top navigation bar, select the target region.
  2. In the left-side navigation pane, choose **Default Instance** > **Repositories**.
  3. On the **Repositories** page, click **Manage** in the Actions column of the target repository.
  4. In the left-side navigation pane, click **Build** and then switch on **Automatically Build Images When Code Changes**.



- Container Registry supports building images on an overseas machine.

  A source code repository outside China may be involved in building images. However, you may have no access to such a repository over the network. To resolve this issue, we provide the **Build With Servers Deployed Outside Mainland China** feature. After you build an image on an overseas machine, you can push the image to a repository in the specified region.

  > **Note:**
  > Sometimes, image pushes may time out if the network over which images are transmitted from outside China to Mainland China is unstable.

- Container Registry supports building images without cache.

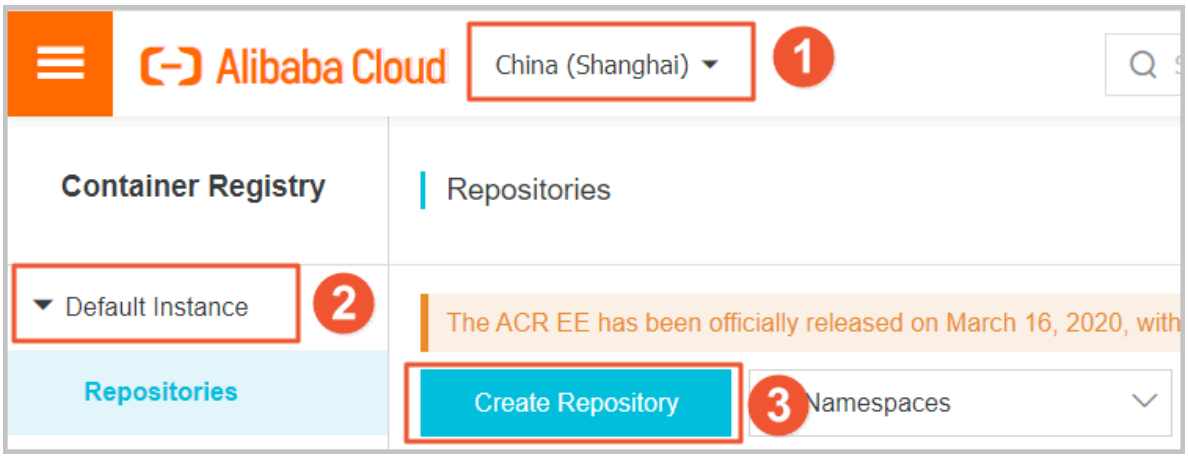
  After you switch on **Build Without Cache**, the dependent base image is pulled each time you build an image. This may slow down the build time. We recommend that you switch off this option.

- Container Registry supports multi-stage build.

Create a repository

Before creating a repository, make sure that a namespace is created in the target region. For more information, see Basic operations on a namespace.

1. Log on to the Container Registry console.
2. In the top navigation bar, select a region. In the left-side navigation pane, choose Default Instance > **Repositories**. On the Repositories page that appears, click **Create Repository**.

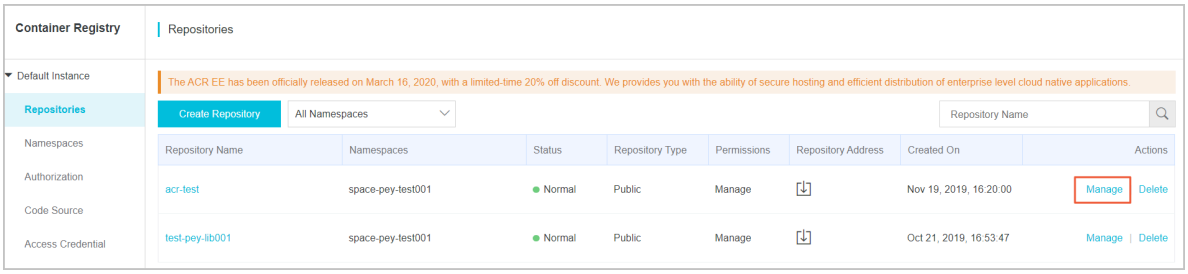


3. In the **Create Repository** dialog box, set the namespace, repository name, repository type, and digest. In this example, set the repository type to **Private**. Then, click **Next**.

4. In the **Code Source** step, set **Code Source** to **Code** and click **Create Repository**.
   - **Automatically Build Images When Code Changes**: If you select this check box, an automatic image build is triggered when code is committed from a branch.
   - **Build With Servers Deployed Outside Mainland China**: If you select this check box, images are built in a data center outside China and then pushed to repositories in the specified region.
   - **Build Without Cache**: If you select this check box, the system forcibly pulls the dependent base image each time you build an image. This may slow down the build time.

Set build rules

1. Log on to the Container Registry console.
2. Click **Manage** in the **Actions** column of the target repository to go to the repository details page.



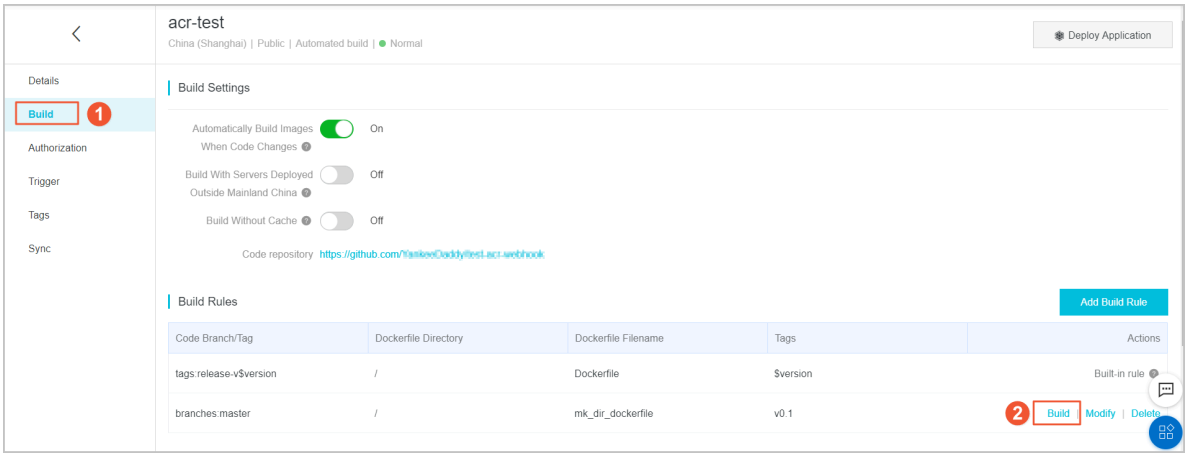
3. In the left-side navigation pane, click **Build**. On the page that appears, click **Add Rule** on the right side of the **Build Rules** section.

> **Note:**
> To modify a build rule, click **Modify** in the Actions column of the target rule.

4. Set a build rule and click **OK**.
   - **Type**: Select a type for the source code repository. Valid values: Branch and Tag.
   - **Code Branch/Tag**: Set the code branch for building images.
   - **Dockerfile Directory**: Set the directory for storing the Dockerfile. The specified directory is a relative directory, with the root directory of the code branch as its parent directory.
   - **Dockerfile Filename**: Set the Dockerfile filename. Default value: Dockerfile.
   - **Tag**: Set a tag for the image, for example, latest.

Build images

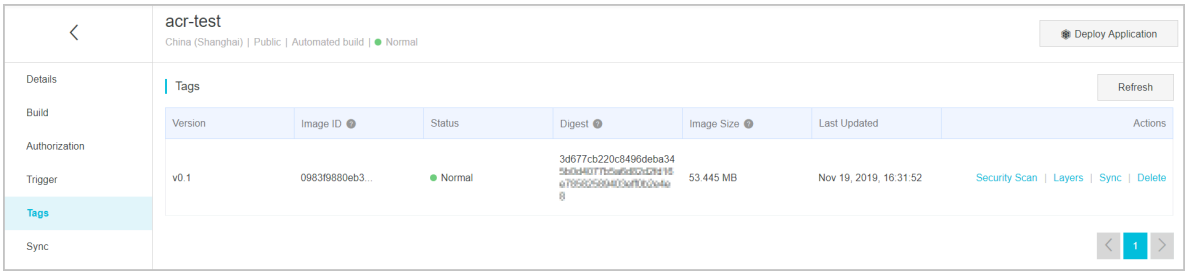
1. Go to the repository details page. In the left-side navigation pane, click **Build**.
2. In the **Build Rules** section, click **Build** in the Actions column of the target rule.



After the image is built, a build record is generated.

3. Click **Log** in the Actions column of the target record to view the log details.
4. After the image is built, click **Tags** in the left-side navigation pane to view the list of created images.

5. To view all the images, click **Build** in the left-side navigation pane. On the page that appears, switch on **Build overseas** and **Build without cache** and retain the default setting for **Automatically build image** in the **Build Settings** section. You can then view all the tags.



What to do next

- #unique_6
- #unique_7
- #unique_8

1.2 Basic operations on a namespace

A namespace allows you to effectively manage a collection of repositories, including repository permissions and repository attributes. This topic describes the basic operations on a namespace.

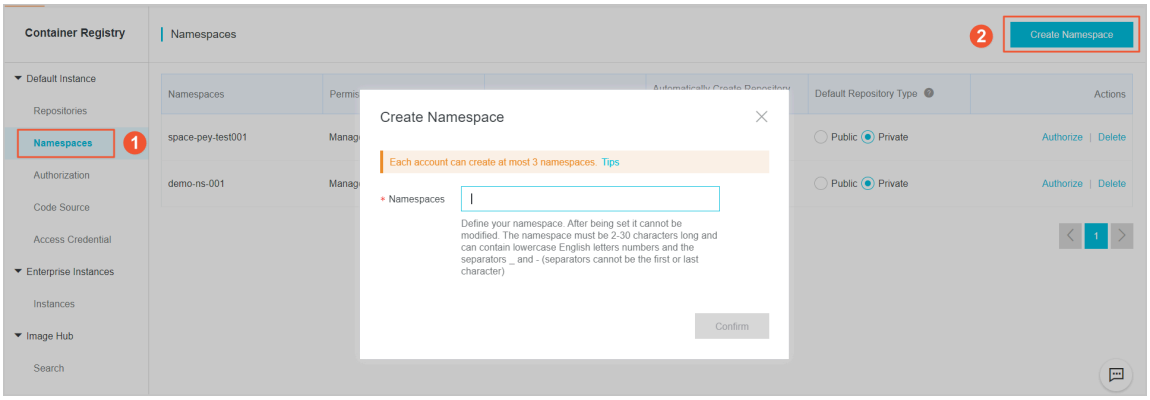
Best practices

A namespace is a collection of repositories. We recommend that you place the repositories of a company or an organization in the same namespace.

- Example of using the company name as the name of a namespace: aliyun or alibaba
- Example of using the team or organization name as the name of a namespace: misaka-team

Basic operations on a namespace

- Create a namespace
  - Click **Create Namespace**. In the dialog box that appears, enter the name of the namespace.

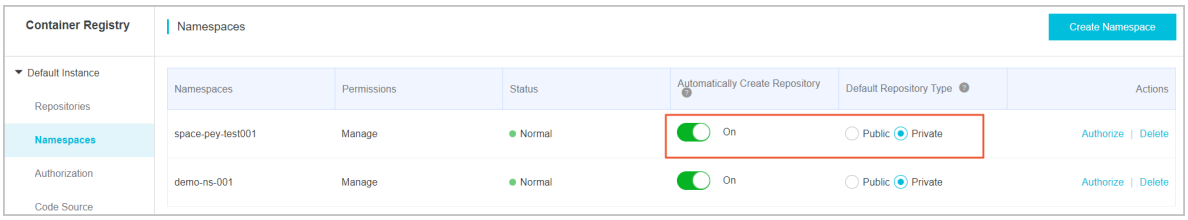


  - Currently, each Alibaba Cloud account can create a maximum of five namespaces.

- Configure a namespace
  - By default, if you push an image to a repository that does not exist, Container Registry can automatically create the repository based on the repository name that you specify.

    To disable this function, set **Automatically Create Repository** to Off.
  - By default, the repository automatically created based on a pushed image is private.

    You can set **Default Repository Type** to Public to change the repository attribute.



1.3 Main features of a repository

A repository is a collection of images. We recommend that you place all image tags of an application or a feature in the same repository. This topic describes the main features of a repository.

Repository naming

We recommend that you use the name of a software package or an application as the name of a repository.

- Example of using the software package name as the name of a repository: centos or jetty
- Example of using the application name as the name of a repository: console-web or console-service

Main features of a repository

- Configure repository visibility
  - If you configure a public repository, all users can pull images from it anonymously.
  - If you configure a private repository, only authorized users can pull images from it after logon.

- Deploy images

  On the details page of a repository, click Deploy Application in the upper-right corner. In the dialog box that appears, click Deploy. The specified images in the repository are deployed in the Container Service console.

- Query images
  - Query images in a repository and obtain the Digest and ImageId values of each image.
  - Check the image layer information, including the size and metadata for each layer of the images.
  - Scan the images to identify vulnerabilities in them and provide solutions for some vulnerabilities.
- Set webhooks
  - Enable the system to send webhook notifications to the specified webhook URL after images are pushed to a repository.
  - Integrate with the downstream processes of Container Registry.
- Authorize a repository

  Grant the access permissions on a repository to Resource Access Management (RAM) users.

- Build images
  - Manage your source code repositories. After you commit the code, images are built based on the build rules you specify and then pushed to the specified repository.
  - Integrate with the upstream processes of Container Registry.

1.4 Repository access control

Alibaba Cloud allows you to use Resource Access Management (RAM) and Security Token Service (STS) to manage access permissions on repositories in a flexible and secure way. This topic describes how to configure access control for repositories in different scenarios.

Background

By default, you have full operation permissions on the resources under your Alibaba Cloud account. With RAM and STS, you can grant different permissions on image resources to different RAM users and provide users with temporary access permissions. Before you configure authorization policies, read RAM documentation.

System policy configuration

- AliyunContainerRegistryFullAccess

  This policy grants a RAM user the same permissions on image resources as those of an Alibaba Cloud account. The RAM user can perform any operations.

```
{
  "Statement": [
    {
      "Action": "cr:*",
      "Effect": "Allow",
      "Resource": "*"
    }
  ],
  "Version": "1"
}
```

- AliyunContainerRegistryReadOnlyAccess

  This policy grants a RAM user the read-only permission on all image resources. For example, the RAM user can view the repository list and pull images.

```
{
  "Statement": [
    {
      "Action": [
        "cr:Get*",
        "cr:List*",
        "cr:PullRepository"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ],
  "Version": "1"
}
```

Policy configuration for typical scenarios

- Scenario 1

  Scenario: Grant a RAM user the read-only permission on a namespace, such as juzhong . After logging on to Container Registry, the RAM user can pull all the images in the namespace juzhong. The RAM user can view information about the namespace and all repositories in the namespace through the API.

```
{
  "Statement": [
    {
      "Action": [
        "cr:Get*",
        "cr:List*",
        "cr:PullRepository"
```

```
    ],
    "Effect": "Allow",
    "Resource": [
     "acs:cr:*:*:repository/juzhong/*"
    ]
   }
  ],
  "Version": "1"
}
```

Notice:

To allow the RAM user to view the namespaces in the console, add the following authorization configuration. Then, the RAM user can view all the namespaces and the repository list. However, the RAM user can only pull images from the repositories in the namespace juzhong.

```
{
  "Statement": [
    {
      "Action": [
        "cr:Get*",
        "cr:List*",
        "cr:PullRepository"
      ],
      "Effect": "Allow",
      "Resource": [
        "acs:cr:*:*:repository/juzhong/*"
      ]
    },
    {
      "Action": [
        "cr:ListNamespace",
        "cr:ListRepository"
      ],
      "Effect": "Allow",
      "Resource": [
        "*"
      ]
    }
  ],
  "Version": "1"
}
```

- Scenario 2

  Scenario: Grant a RAM user all permissions on a repository, such as the repository nginx in the namespace juzhong in the China (Hangzhou) region.

  Notice:

To allow the RAM user to manage repositories in the console, add the relevant configuration by referring to scenario 1.

```
{
  "Statement": [
    {
      "Action": [
        "cr:*"
      ],
      "Effect": "Allow",
      "Resource": [
        "acs:cr:cn-hangzhou:*:repository/juzhong/nginx"
      ]
    },
    {
      "Action": [
        "cr:Get*",
        "cr:List*"
      ],
      "Effect": "Allow",
      "Resource": [
        "acs:cr:*:*:repository/juzhong"
      ]
    }
  ],
  "Version": "1"
}
```

Notes on RAM authorization

When you authorize a RAM user, pay attention to the following instructions to avoid granting excessive permissions to the RAM user.

If you grant a RAM user the AdministratorAccess permission, that is, management permissions on all Alibaba Cloud resources, the RAM user possesses all permissions on Container Registry, regardless of whether the RAM user is granted the permissions before.

Authentication rules for Container Registry

- Resource description

  The following table lists the resource description in an authorization policy when you use RAM to authorize access to resources.

Resource	Resource description in an authorization policy
Repository	acs:cr:$regionid:$accountid:repository/$namespacename/$repositorynameacs:cr:$regionid:$accountid:repository/$namespacename/*acs:cr:$regionid:$accountid:repository/$namespacename

  The following table describes the parameters in the resource description.

Parameter	Description
$regionid	The ID of the region, which can be replaced by an asterisk (*).
$accountid	The ID of the Alibaba Cloud account, which can be replaced by an asterisk (*).
$namespacename	The name of the namespace.
$repositoryname	The name of the repository.

- Authentication rules

  When you access the Container Registry API as a RAM user or using STS, Container Registry informs RAM to perform a permission check to make sure that the caller has the required permissions. The permissions to be checked are determined by the resources used by an API operation and the API syntax. The following table describes the API authentication rules.

API operation	Authenticated action	Authenticated resource
Create a namespace	cr:CreateNamespace	*
Delete a namespace	cr:DeleteNamespace	acs:cr:$regionid:$accountid:repository/$namespacename
Update a namespace	cr:UpdateNamespace	acs:cr:$regionid:$accountid:repository/$namespacename

API operation	Authenticated action	Authenticated resource
Obtain the information about a specified namespace	cr:GetNamespace	acs:cr:$regionid:$accountid:repository/$namespacename
Obtain namespaces	cr:ListNamespace	*
Create a repository	cr:CreateRepository	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
Delete a repository	cr:DeleteRepository	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
Update repository information	cr:UpdateRepository	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
View information of a repository	cr:GetRepository	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
View information of repositories	cr:ListRepository	*
View information of repositories based on the namespace	cr:ListRepository	*
View the tag information of a repository	cr:ListRepositoryTag	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
Delete an image tag	cr:DeleteRepositoryTag	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
View the Manifest information of an image	cr:GetRepositoryManifest	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname

API operation	Authenticated action	Authenticated resource
View image layer information	cr:GetRepositoryLayers	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
Obtain a temporary authorization token	cr:GetAuthorizationToken	*
Pull images	cr:PullRepository	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname
Push images	"cr:PushRepository",	acs:cr:$regionid:$accountid:repository/$namespacename/$repositoryname

1.5 Webhook management

Container Registry provides a webhook for each repository to push messages about successful image building. This facilitates redeployment of images for applications. If you set a webhook for Container Service, the applications on Container Service will be triggered to automatically pull and redeploy the new images that are built. This topic describes how to use a webhook.

Container Registry provides two triggering methods for a webhook: expression-based triggering and tag-based triggering. If no triggering condition is specified, an image pull is triggered each time a new image tag is generated or the image tag is updated.

- Expression-based triggering: Tags are filtered based on a regular expression. An image pull is triggered only when a tag matches the regular expression.
- Tag-based triggering: An image pull is triggered based on the specified tags.

Examples

- Expression-based triggering

  You can enter a simple regular expression, such as `release-v.*`. An image pull is triggered only after an image with a tag starting with `release-v` is built. Otherwise, no image pull is triggered and the access status code in the access log is "untriggered."

  Click **Access Log** to view the access history of the webhook.

- Tag-based triggering

  You can specify a maximum of 10 tags for which an image pull needs to be triggered . Then, an image pull is triggered only when images containing the specified tags are built. Otherwise, no image pull is triggered and the access status code in the access log is "untriggered."

  Click **Access Log** to view the access history of the webhook.

Notification content

The notification content of a webhook includes the repository information and image tag information. The repository information includes the namespace, name, and region of the repository.

```
POST /payload HTTP/1.1

Content-Type: application/json
Request URL: https://cs.console.aliyun.com/hook/trigger?triggerUrl=YzRmMWE5Yz
M2ZjMzYzQ0NmFiMGYzNWJlMmM2MjM2NzIyfGV4cHJlc3N8cmVkZXBsb3l8MThlMmllY2dr
dXYyZXw=&secret=365a4a664b45615438716a487a75695a7ac48329224b35b073
c2197374e7d62a
Request method: POST

{
   "push_data": {
      "digest": "sha256:457f4aa83fc9a6663ab9d1b0a6e2dce25a12a943ed5bf2c174
7c58d48bbb4917",
      "pushed_at": "2016-11-29 12:25:46",
      "tag": "latest"
   },
   "repository": {
      "date_created": "2016-10-28 21:31:42",
      "name": "repoTest",
      "namespace": "namespace",
      "region": "cn-hangzhou",
      "repo_authentication_type": "NO_CERTIFIED",
      "repo_full_name": "namespace/repoTest",
      "repo_origin_type": "NO_CERTIFIED",
      "repo_type": "PUBLIC"
   }
}
```

1.6 Image Scanning

Continer Registry allows you to perform security scan on all Linux-based images. Discover any known vulnerabilities in packages or other dependencies defined in the container

image file. You can receive vulnerability assessments and recommendations, including specific remediation guidance.

Continer Registry provides three methods for scanning. You can **manually** scan container images by one click, or you can configure cloud-native delivery chain to **automatically** scan images when you push them to a repository. Besides, by leveraging ACR's image scan OpenAPI with OOS's scheduled tasks or FC's Time Trigger, you can set up automate periodic scan of your container images with ease.

With the cloud-native delivery chain, Container Registry can **automatically scan** the new container images after pushing. If the image meets conditions defined in the chain blocking policy, the system will **automatically block** the risky image to deploy. Otherwise, the system proceeds with follow-up steps. The chain with image security policy guarantees that images are safe enough to distribute.

Manually scan

1. Log on to the Container Registry console.
2. Select the region.
3. Click **Manage** at the right of an image repository to enter the repository details page.
4. Click **Image Versions** in the left-side navigation pane. Click **Security Scan** at the right of the image version.
5. Click **Trigger Scan** to **manually** scan container images by one click.

Automatically scan

1. Configure the basic information of chain.
2. Configure image security scanning and node blocking rule.
3. ACR will **automatically** scan new images when they're uploaded.
4. ACR will **automatically lock down risky images** follow the related block strategy of the cloud-native delivery chain.

Periodically scan

1. Configure the OOS's scheduled tasks and ACR's image scan OpenAPI.
2. ACR will **periodically scan** of your container images.

Result

After an image security scan is completed, a vulnerability report is generated as follows. Vulnerability information is categorized into four levels: **High**, **Medium**, **Low**, and **Unknown**. Additionally, it gives vulnerability details and the corresponding guidance for how to remediate the specific vulnerabilities found on each image pushed to registry.

Currently, All Linux-based images are supported. The images wiith following base OS are tested.

- Ubuntu Linux: 12.04 or later
- RedHat Enterprise Linux: 5, 6, and 7
- CentOS Linux: 5, 6, and 7
- Oracle Linux: 5, 6, and 7
- Debian Linux: 7, 8, 9, and 10
- Alpine Linux: 3.3 or later