# 天翼云•容器镜像服务

## 用户使用指南

天翼云科技有限公司

# 目 录

# 1 产品概述

## 1.1 产品定义

容器镜像服务（Software Repository for Container，简称 SWR）是一种支持镜像全生命周期管理的服务， 提供简单易用、安全可靠的镜像管理功能，帮助您快速部署容器化服务。您可以通过界面、社区 CLI 和原生 API 上传、下载和管理容器镜像。

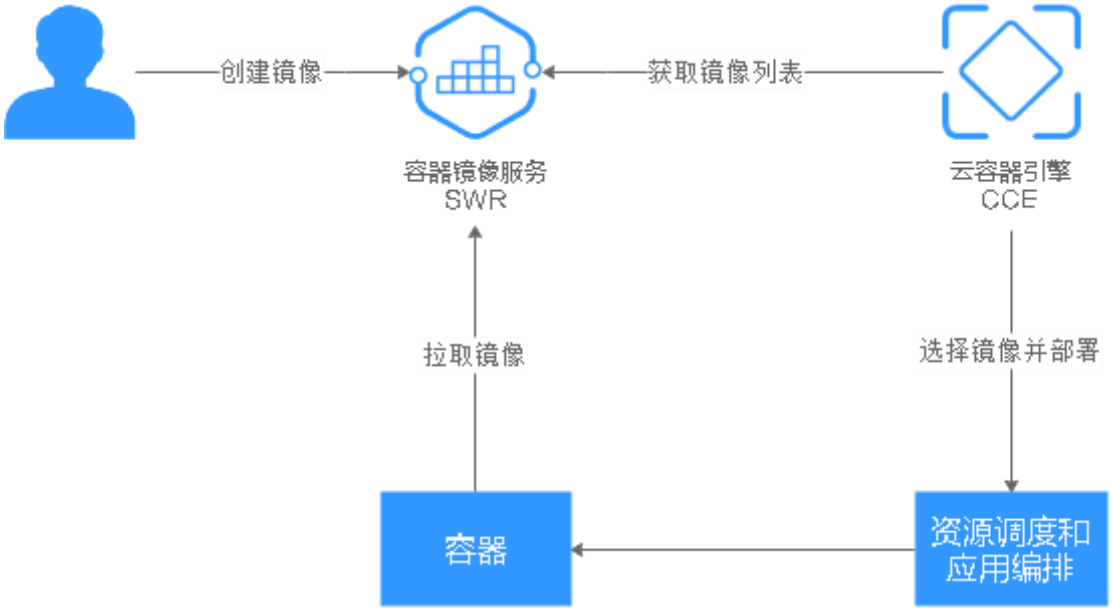
## 1.2 应用场景

**镜像生命周期管理**

提供镜像构建、镜像上传、下载、同步、删除等完整的生命周期管理能力。

优势

- 高可靠的存储：依托专业存储服务，确保镜像存储的超高可靠性。
- 更安全的存储：细粒度的授权管理，让用户更精准的控制镜像访问权限。




建议搭配使用

云容器引擎 CCE

## 1.3 产品优势

**简单易用**

容器镜像服务的管理控制台简单易用，支持镜像的全生命周期管理。

**安全可靠**

容器镜像服务遵循 HTTPS 协议保障镜像安全传输，提供账号间、账号内多种安全隔离机制，确保用户数据访问的安全。

容器镜像服务依托专业存储服务，确保镜像存储更可靠。

**无缝对接 CCE**

容器镜像服务提供镜像部署入口，与云容器引擎 CCE 无缝对接，一键式部署容器应用。

**开放兼容**

全面支持社区 Registry V2 协议，支持使用 CLI 以及原生 API 方式管理镜像

## 1.4 常用概念

**镜像（Image）**

容器镜像是一个模板，是容器应用打包的标准格式，在部署容器化应用时可以指定镜像，镜像可以来自于镜像中心或者用户的私有 Registry。例如一个容器镜像可以包含一个完整的 Ubuntu 操作系统环境，里面仅安装了用户需要的应用程序及其依赖文件。容器镜像用于创建容器。容器引擎本身提供了一个简单的机制来创建新的镜像或者更新已有镜像，您也可以下载其他人已经创建好的镜像来使用。

**容器（Container）**

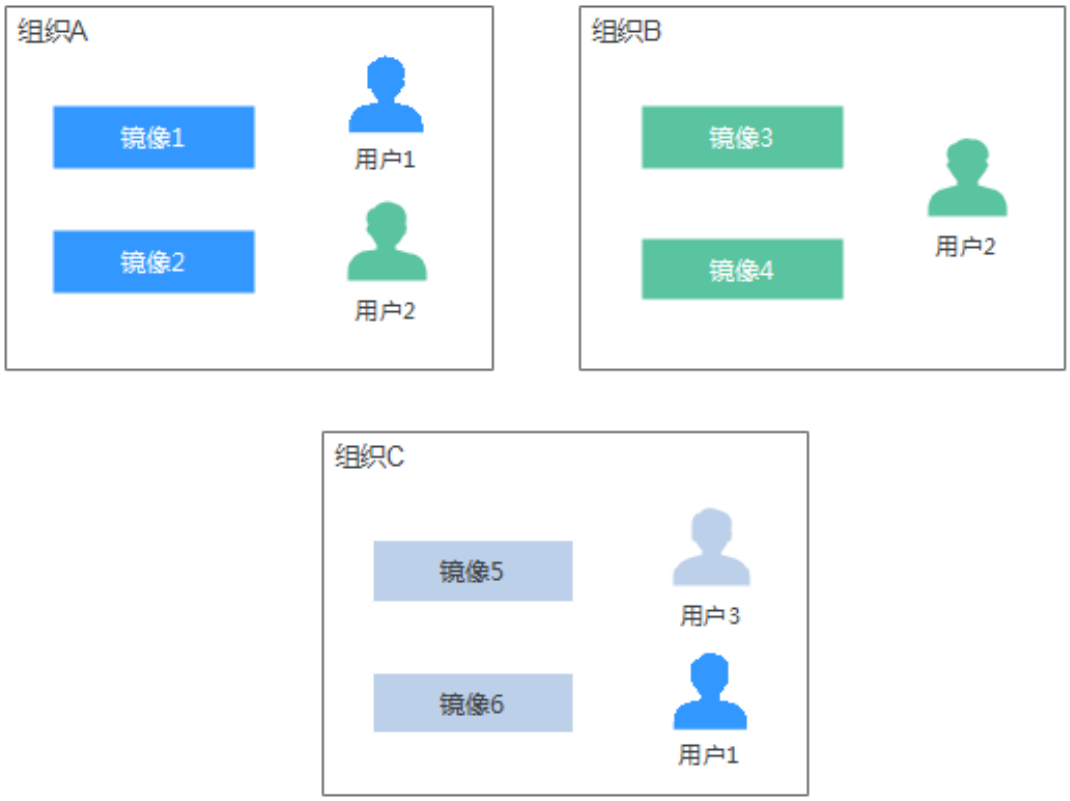
一个通过容器镜像创建的运行实例，一个节点可运行多个容器。容器的实质是进程，但与直接在宿主执行的进程不同，容器进程运行于属于自己的独立的命名空间。

镜像（Image）和容器（Container）的关系，就像是面向对象程序设计中的类和实例一样，镜像是静态的定义，容器是镜像运行时的实体。容器可以被创建、启动、停止、删除、暂停等。

**组织**

组织用于隔离镜像仓库，每个组织可对应一个公司或部门，将其拥有的镜像集中在该组织下。同一用

户可属于不同的组织。支持为账号下不同用户分配相应的访问权限（读取、编辑、管理）。




## 1.5 与其他服务关系

容器镜像服务需要与其他云服务协同工作，容器镜像服务和其他云服务的关系。


- 云容器引擎

云容器引擎（Cloud Container Engine，简称 CCE）提供高可靠高性能的企业级容器应用管理服务，支持 Kubernetes 社区原生应用和工具，简化云上自动化容器运行环境搭建。

容器镜像服务能无缝对接 CCE，您可以将容器镜像服务中的镜像部署到 CCE 中。

- 云审计服务

云审计服务（Cloud Trace Service，简称 CTS）为您提供云服务资源的操作记录，记录内容包括您从公有云管理控制台或者开放 API 发起的的云服务资源操作请求以及每次请求的结果，可用于支撑安全分析、合规审计、资源跟踪和问题定位等常见应用场景。

通过 CTS，您可以记录与容器镜像服务相关的操作事件，便于日后的查询、审计和回溯。

## 1.6 约束与限制

### 配额

容器镜像服务对单个用户的组织数量限定了配额，配额的详细信息请参见关于配额。

当前容器镜像服务的配额如下表所示。如果您需要添加更多的组织，请提交工单申请。

| 资源类型 | 配额 |
| --- | --- |
| 组织数量 | 5 |

# 2 购买指南

## 2.1 价格

目前容器镜像服务 **免费。**

用户可直接进入控制台使用该服务。

# 3 快速入门

## 3.1 容器客户端上传镜像

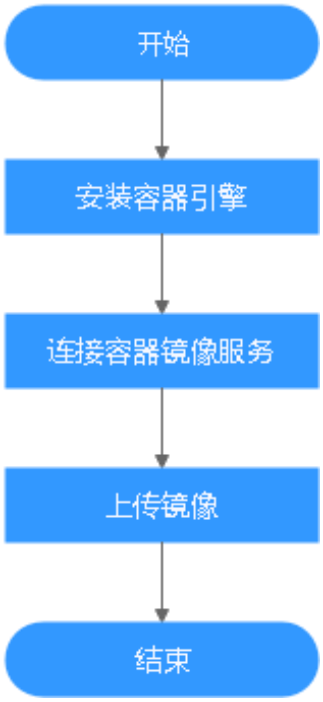
### 入门指引

容器镜像服务是一种支持容器镜像全生命周期管理的服务， 提供简单易用、安全可靠的镜像管理功能，帮助用户快速部署容器化服务。本文档将帮助您学习如何安装容器引擎以及如何使用容器引擎客户端上传镜像到容器镜像仓库。

### 说明

上传镜像仅适用于管理控制台操作。

快速入门流程图



**准备工作**

在使用容器镜像服务前，您需要完成天翼云注册并实名认证的准备工作。

本文以一个 2048 应用为例，讲述根据该应用编写 Dockerfile 文件构建镜像并上传至容器镜像服务的操作。您可以编写一个 Dockerfile 文件，以 alpine:3.7 为基础镜像，来构建一个 2048 容器镜像。

· **前提条件**

- 已安装 Docker
- 已获取 2048 应用，并将该镜像下载至本地。

· **构建镜像**

**步骤 1**：使用 root 用户登录虚拟机。

**步骤 2**：创建 **2048-demo** 目录。

```
mkdir 2048-demo
```

**步骤 3**：下载 2048 源码至 2048-demo 目录中。

该应用源码地址为：https://github.com/gabrielecirulli/2048.git。

**步骤 4**：创建名为 Dockerfile 的文件。

```
cd 2048-demo

touch Dockerfile
```

**步骤 5**：编辑 Dockerfile。

```
vi Dockerfile
```

```
FROM alpine:3.7
RUN apk --update add nginx
COPY 2048 /usr/share/nginx/html
```

```
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

- FROM：指定基础镜像 alpine:3.7。
- RUN：安装 nginx。
- COPY：将 2048 源码拷贝到容器内的“/usr/share/nginx/html”目录。
- EXPOSE：暴露容器的 80 端口。
- CMD：指定容器运行时的默认命令。

保存并退出。

**步骤 6：**执行以下命令构建镜像。

```
docker build -t 2048-demo:v1 .
```

镜像构建的格式为：docker build [选项] <上下文路径>，详细命令可参考

https://docs.docker.com/engine/reference/commandline/build/。

- -t 2048-demo:v1：指定镜像的名称和版本。
- .：指定 Dockerfile 所在目录，镜像构建命令会按照 Dockerfile 的内容构建镜像。

**步骤 7：**执行以下命令，可查看到已成功构建的 **2048-demo** 镜像，版本为 v1。

```
docker images | grep 2048-demo
```

查看已构建的 2048-demo 镜像


**创建组织**

组织用于隔离镜像，并为租户下用户指定不同的权限（读取、编辑、管理）。

- 读取：只能下载镜像，不能上传。
- 编辑：下载镜像、上传镜像、编辑镜像属性以及创建触发器。
- 管理：下载镜像、上传镜像、删除镜像或版本、编辑镜像属性、添加授权、以及共享镜像。

**步骤 1：**登录容器镜像服务控制台。

**步骤 2：**在左侧菜单栏选择“组织管理”，单击右侧“创建组织”，在弹出的页面中填写“组织名称”，然后单击“确定”。

**客户端上传镜像**

docker 客户端上传镜像，是指使用 docker 命令将镜像上传到容器镜像服务的镜像仓库。

本章节以 **2048-demo:v1** 镜像为例，介绍如何上传镜像。上传成功后，在“我的镜像”中显示已上传成功的镜像。

注意

- 使用客户端上传镜像，镜像的每个 layer 大小不能超过 10G。
- 上传镜像的 Docker 客户端版本必须为 1.11.2 及以上。

**步骤 1：**连接容器镜像服务。

1. 登录容器镜像服务控制台。
2. 在左侧菜单栏选择“我的镜像”，单击右侧“客户端上传”，在弹出的页面中单击“生成临时 docker login 指令”，单击 复制 docker login 指令。docker login 指令末尾的域名即为当前镜像仓库地址，记录该地址。

   说明

   此处获取的 docker login 指令有效期为 24 小时，若需要长期有效的 docker login 指令，请参见获取长期有效 docker login 指令。

3. 在安装 Docker 的机器中执行上一步复制的 docker login 指令。

登录成功会显示“login succeeded”。

**步骤 2：**在安装 docker 的机器给 **2048-demo:v1** 镜像打标签。

docker tag [镜像名称:版本名称] [镜像仓库地址]/[组织名称]/[镜像名称:版本名称]

说明

镜像名称不支持多级目录格式，例如：镜像名称可命名为“2048-demo”，不可命名为“2048-demo/test”。

样例如下：

docker tag 2048-demo:v1 {Public image address}/group/2048-demo:v1

其中：

- {Public image address}为容器镜像服务的镜像仓库地址。获取该地址的方式：单击“我的镜像”，单击镜像列表中的镜像名称，在“Pull/Push 指南”页签中的**“1. 本镜像地址”**下可以看到镜像仓库地址。
- group 为组织名称，如果该组织还没有创建，容器镜像服务会根据组织名称自动创建一个组织。
- 2048-demo:v1 为镜像名称和版本号。

**步骤 3：**上传镜像至镜像仓库。

docker push [镜像仓库地址]/[组织名称]/[镜像名称:版本名称]

样例如下：

docker push {Public image address}/group/2048-demo:v1

终端显示如下信息，表明 push 镜像成功。

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

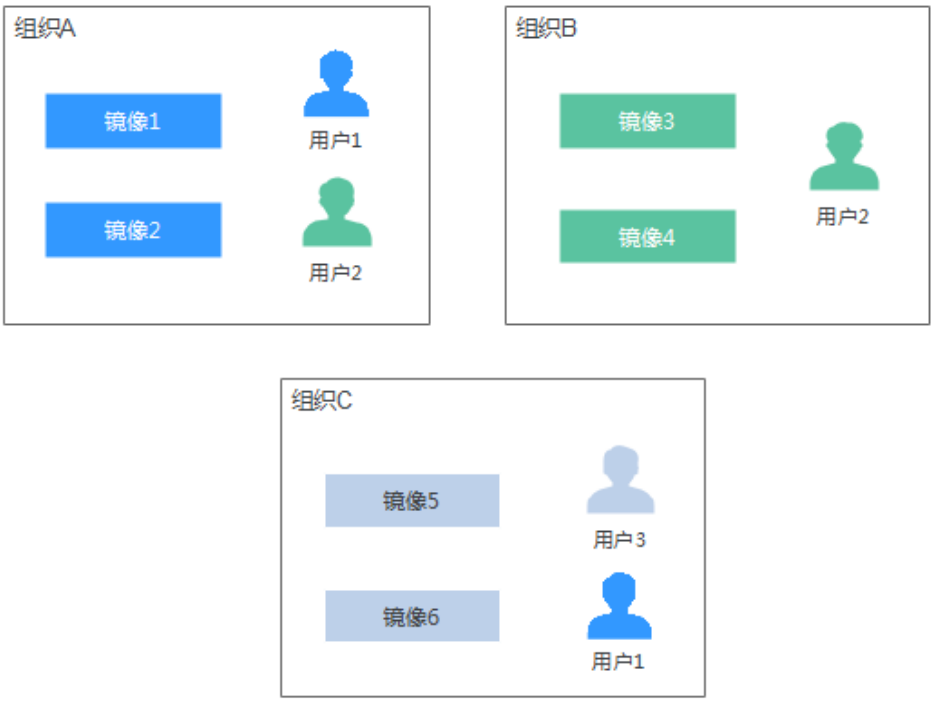
返回系统，在“我的镜像”页面，执行刷新操作后可查看到对应的镜像信息。

# 4 操作指导

## 4.1 组织管理

组织用于隔离镜像，并为租户下用户指定不同的权限（读取、编辑、管理）。

组织用于隔离镜像仓库，每个组织可对应一个公司或部门，将其拥有的镜像集中在该组织下。在不同的组织下，可以有同名的镜像。同一 IAM 用户可属于不同的组织。SWR 支持为账户下 IAM 用户分配相应的访问权限（读取、编辑、管理）。




**创建组织**

容器镜像服务为您提供组织管理功能，方便您根据自身组织架构来构建镜像的资源管理。上传镜像前，请先创建组织。

1. 登录容器镜像服务控制台。

2. 在左侧菜单栏选择“组织管理”，单击右侧“创建组织”，在弹出的页面中填写“组织名称”，然后单击“确定”。

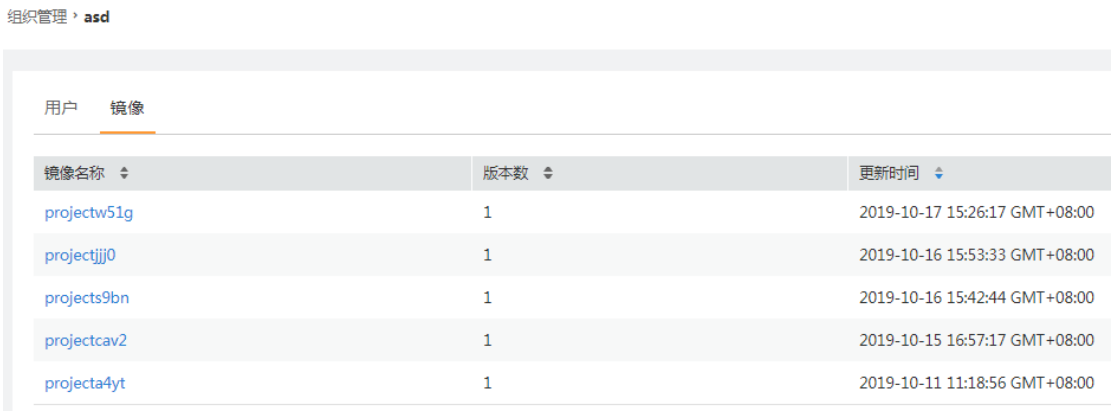

**查看组织中的镜像**

创建组织后，您可以查看当前组织中的镜像。

1. 登录容器镜像服务控制台。

2. 在左侧菜单栏选择“组织管理”，单击右侧组织名称。

3. 单击“镜像”页签，查看当前组织中的镜像。

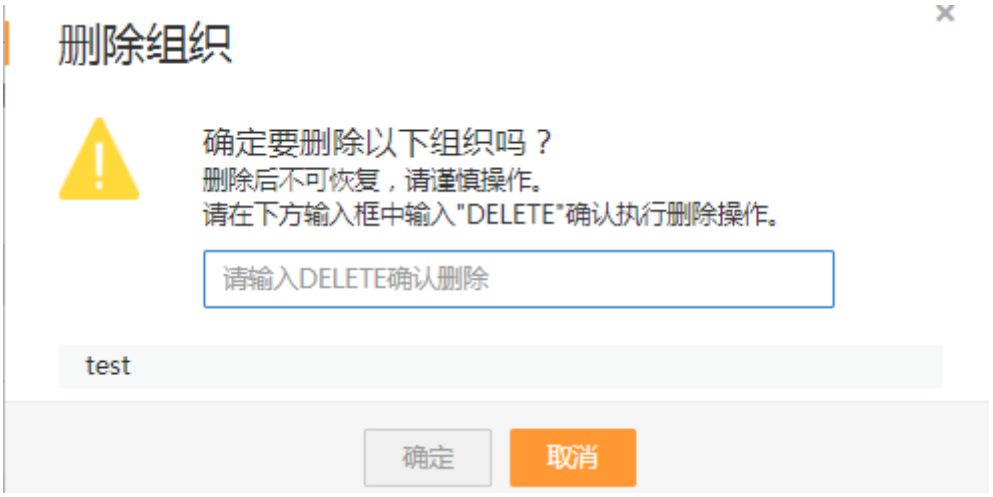



**删除组织**

删除组织前，请先删除组织下的所有镜像和软件仓库。

1. 登录容器镜像服务控制台。
2. 在左侧菜单栏选择“组织管理”，单击具体组织名称进入详情页面。
3. 单击右上角“删除”按钮，在弹出的对话框中根据提示输入“DELETE”，然后单击“确定”。

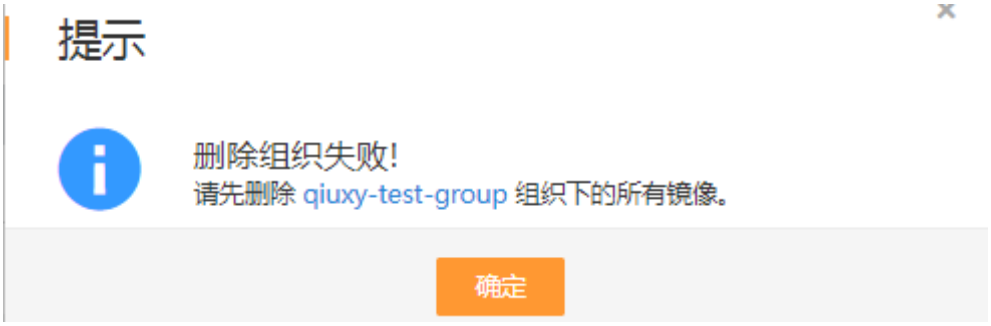
图 删除组织




**说明：**

如该组织中还有镜像，则无法删除组织，需要按提示先将组织中的镜像删除后，确认组织中

无镜像文件后，再进行组织删除操作。




# 4.2 镜像管理

## 4.2.1 客户端上传镜像

### 操作场景

本章节以 **2048-demo:v1** 镜像为例，介绍如何使用客户端上传镜像。客户端上传镜像，是指使用命令将镜像上传到容器镜像服务的镜像仓库。

本章节以 **2048-demo:v1** 镜像为例，介绍如何上传镜像。

### 前提条件

已创建组织。

### 操作步骤

docker 客户端上传镜像，是指使用 docker 命令将镜像上传到容器镜像服务的镜像仓库。

本章节以 **2048-demo:v1** 镜像为例，介绍如何上传镜像。上传成功后，在“我的镜像”中显示已上传成功的镜像。

注意

- 使用客户端上传镜像，镜像的每个 layer 大小不能超过 10G。
- 上传镜像的 Docker 客户端版本必须为 1.11.2 及以上。

**步骤 1：连接容器镜像服务。**

1. 登录容器镜像服务控制台。
2. 在左侧菜单栏选择“我的镜像”，单击右侧“客户端上传”，在弹出的页面中单击“生成临时 docker login 指令”，单击 复制 docker login 指令。docker login 指令末尾的域名即为当前镜像仓库地址，记录该地址。

   说明

   此处获取的 docker login 指令有效期为 24 小时，若需要长期有效的 docker login 指令。

3. 在安装 Docker 的机器中执行上一步复制的 docker login 指令。

登录成功会显示“login succeeded”。

**步骤 2：在安装 docker 的机器给 2048-demo:v1 镜像打标签。**

docker tag [镜像名称:版本名称] [镜像仓库地址]/[组织名称]/[镜像名称:版本名称]

说明

镜像名称不支持多级目录格式，例如：镜像名称可命名为“2048-demo”，不可命名为“2048-demo/test”。

样例如下：

docker tag 2048-demo:v1 {Public image address}/group/2048-demo:v1

其中：

- {Public image address}为容器镜像服务的镜像仓库地址。获取该地址的方式 :单击“我的镜像”，单击镜像列表中的镜像名称 ,在“Pull/Push 指南”页签中的**“1. 本镜像地址”**下可以看到镜像仓库地址。
- group 为组织名称，如果该组织还没有创建，容器镜像服务会根据组织名称自动创建一个组织。
- 2048-demo:v1 为镜像名称和版本号。

**步骤 3：上传镜像至镜像仓库。**

docker push [镜像仓库地址]/[组织名称]/[镜像名称:版本名称]

样例如下：

docker push {Public image address}/group/2048-demo:v1

终端显示如下信息，表明 push 镜像成功。

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

返回系统，在“我的镜像”页面，执行刷新操作后可查看到对应的镜像信息

### 4. 2. 2 获取长期有效 docker login 指令

#### 操作场景

本章节介绍如何获取长期有效的登录指令，长期有效登录指令的有效期为永久。

说明：为保证安全，获取登录指令过程建议在开发环境执行。

#### 操作步骤

**步骤 1：获取镜像仓库访问地址、区域项目名称。**

1、访问我的凭证：登录控制台，鼠标移动到右上角您的用户名处，单击后下拉选项中选择进入“我的凭证”。


2、在“项目列表”页签中查找当前区域对应的项目。

在容器镜像服务控制台中获取镜像仓库地址。

获取该地址的方式：

单击“我的镜像”，单击镜像列表中的镜像名称，在“Pull/Push 指南”页签中的**“step3 下载镜像或上传镜像”**下可以看到镜像仓库地址。


**步骤 2：获取 AK/SK 访问密钥。**

说明

如果已有 AK/SK，可以直接使用，无需再次获取。

1、访问我的凭证：登录控制台，鼠标移动到右上角您的用户名处，单击“我的凭证”。

2、在“管理访问密钥”页签，单击列表下侧的“新增访问密钥”，创建新的访问密钥。

3、输入当前用户的登录密码，并通过邮箱或者手机进行验证。

说明

在统一身份认证服务中创建的用户，如果创建时未填写邮箱或者手机号，则只需校验登录密码。

4、单击“确定”，下载访问密钥。

请妥善保存已下载的密钥，后续创建集群时需要上传该密钥，否则会无法创建集群。

说明

为防止访问密钥泄露，建议您将其保存到安全的位置。

**步骤 3：登录一台 linux 系统的计算机，执行如下命令获取登录密钥。**

```
printf "$AK" | openssl dgst -binary -sha256 -hmac "$SK" | od -An -vtx1 | sed 's/[ \n]//g' | sed 'N;s/\n//'
```

其中$AK 和$SK 为步骤 2 获取的 AK/SK。

示例：

```
[root@SZV1000258977 ~]# printf "DKAKX9J6OBEVMARHLBQM" | openssl d
gst -binary -sha256 -hmac "0uDrd9HcRhmngEhAXo6SQiflN1UqufLF531jiF
kX" | od -An -vtx1 | sed 's/[ \n]//g' | sed 'N;s/\n//'
7ca3582173f52caa98fcf87389e9cc26d007a2e4b2f6231006a301568f2e1ef8
```

**步骤 4：使用如下的格式拼接 docker login 指令。**

docker login -u [区域项目名]@[AK] -p [登录密钥] [镜像仓库地址]

其中，区域项目名和镜像仓库地址在步骤 1 中获取，AK 在步骤 2 中获取，登录密钥为步骤 3 的执行结果。

### 4. 2. 3 页面上传镜像

**操作场景**

从页面上传镜像，是指直接通过页面将镜像上传到容器镜像服务。

注意

每次最多上传 10 个文件，单个文件大小（含解压后）不得超过 2G，页面上传支持 tar、tar.gz 格式镜像文件。

**前提条件**

- 已创建组织，
- 镜像已存为 tar 或 tar.gz 文件
- 仅支持上传 1.11.2 及以上 Docker 客户端版本制作的镜像压缩包。

操作步骤

**步骤 1：**登录容器镜像服务控制台。

**步骤 2：**在左侧菜单栏选择“我的镜像”，单击右侧“页面上传”。

**步骤 3：**在弹出的窗口中选择镜像要上传的组织，单击“选择镜像文件”，选择要上传的镜像文件。


说明：多个镜像同时上传时，镜像文件会按照顺序逐个上传，不支持并发上传。

**步骤 4：**在弹出的窗口中单击“开始上传”。

待任务进度显示“上传完成”，表示镜像上传成功。

### 4. 2. 4 下载镜像

#### 操作场景

镜像上传后，您可以获取镜像下载地址，使用 docker pull 命令下载镜像。

#### 操作步骤

1. 以 root 用户登录容器引擎所在的虚拟机。
2. 参考 1 获取登录访问权限，连接容器镜像服务。
3. 登录容器镜像服务控制台。

4. 在左侧菜单栏选择“我的镜像”，单击右侧镜像名称。

5. 在镜像详情页面中，单击对应镜像版本“下载指令”列的复制图标，复制镜像下载指令。

   图获取镜像下载指令


6. 在虚拟机执行 5 复制的镜像下载指令。

7. 执行如下命令将镜像保存为归档文件。

   **docker save** [*镜像名称:版本名称*] **>** [*归档文件名称*]

### 4.2.5 编辑镜像属性

#### 操作场景

镜像上传后默认为私有镜像，您可以设置镜像的属性，包括镜像的类型（“公开”或“私有”）、分类及描述。

公开镜像所有用户都能下载，私有镜像则受具体权限管理控制。您可以为用户添加授权，授权完成后，用户享有读取、编辑或管理私有镜像的权限。

#### 操作步骤

1. 登录容器镜像服务控制台。

2. 在左侧菜单栏选择“我的镜像”，单击右侧要编辑镜像的名称。

3. 在镜像详情页面，单击右上角“编辑”，根据需要在弹出的窗口中编辑类型(“公开”或“私有”)、分类及描述，然后单击“确定”。

图编辑镜像属性


| 参数 | 说明 |
| --- | --- |
| 组织 | 镜像所属组织。 |
| 名称 | 镜像名称 |
| 类型 | 镜像类型，可选择：公开、私有<br>**说明：**<br>公开镜像所有用户都可以下载使用。<br>· 如果您的节点与镜像仓库在同一区域，访问仓库是通过内网访问。<br>· 如果您的节点与镜像仓库在不同区域，通过公网才能访问仓库，下载跨区域仓库的镜像需要节点可以访问公网 |
| 分类 | 镜像分类，可选择：<br>· 应用服务器<br>· Linux<br>· Windows<br>· Arm<br>· 框架与应用 |

| | |
|---|---|
| | · 数据库<br>· 语言<br>· 其他 |
| 描述 | 输入镜像仓库描述，0-30000 个字符 |

### 4.2.6 共享私有镜像

#### 操作场景

镜像上传后，您可以共享**私有镜像**给其他账号，并授予下载该镜像的权限。

被共享的用户需要登录容器镜像服务控制台，在“我的镜像 > 他人共享”页面查看共享的镜像。

被共享的用户单击镜像名称，可进入镜像详情页面查看镜像版本、下载指令等。

#### 使用约束

- 镜像共享功能只支持私有镜像进行共享，不支持公有镜像共享。
- 仅账号和具备该私有镜像管理权限 IAM 用户才能共享镜像，被共享者只有只读权限，只能下载镜像。
- 镜像共享功能只能在同一区域内使用，不支持在不同区域间镜像共享。

#### 操作步骤

1. 登录容器镜像服务控制台。
2. 在左侧菜单栏选择“我的镜像”，单击右侧镜像的名称。
3. 在镜像详情页面选择“共享”页签。
4. 单击“共享镜像”，根据表格提示填写相关参数，然后单击“确定”。

图 共享镜像


表 共享镜像

| 参数 | 说明 |
| --- | --- |
| 共享给 | 输入账号名称。 |
| 截止日期 | 选择共享截止日期。如勾选“永久有效”，则共享永久有效。 |
| 描述 | 输入描述，0-1000 个字符。 |
| 权限 | 当前仅支持“下载”权限 |

5. 共享完成后，您可以在“我的镜像 > 自有镜像”中，勾选“我共享的镜像”，查看所有共享的镜像。

# 5 常见问题

## 5.1 通用类

### 5.1.1 什么是镜像仓库服务？

容器镜像服务（SoftWare Repository for Container）是一种支持容器镜像全生命周期管理的服务， 提供简单易用、安全可靠的镜像管理功能，帮助用户快速部署容器化服务。

### 5.1.2 什么是组织？

组织用于隔离镜像仓库，每个组织可对应一个公司或部门，将其拥有的镜像集中在该组织下。同一用户可属于不同的组织。容器镜像服务支持为组织中不同用户分配相应的访问权限（读取、编辑、管理）。

### 5.1.3 容器镜像仓库服务是否收费？

容器镜像服务暂不收费，可以免费使用。

### 5.1.4 SWR 最多能存储多少个镜像？

SWR 对存储的镜像数量没有限制，您可以根据需要上传镜像。

### 5.1.5 SWR 的配额是多少？

容器镜像服务对镜像仓库数量没有配额限制，您可以根据需要创建镜像仓库。

容器镜像服务对单个用户的组织数量限定了配额，5 个。

如果您需要添加更多的组织，请提交工单申请。

### 5.1.6 为什么创建组织提示组织已存在？

SWR 中组织名称是所有账号共用的，比如 A 用户创建了“组织 B”，B 用户就没法使用“组织 B”。因此创建组织时如果提示组织已存在，可能该组织名称已被其他用户使用，请重新设置一个组织名称。

### 5.1.7 容器镜像服务的带宽多大？

容器镜像服务的带宽会根据用户使用情况动态变化。

### 5.1.8 如何安装容器引擎？

容器引擎几乎支持在所有操作系统上安装，用户可以根据需要选择要安装的容器引擎版本。

说明：

- 容器镜像服务支持使用容器引擎 1.11.2 及以上版本上传镜像。
- 安装容器引擎需要连接互联网，内网服务器需要绑定弹性 IP 后才能访问。

另外，在 Linux 操作系统下，可以使用如下命令快速安装容器引擎。

```
curl -fsSL get.docker.com -o get-docker.sh

sh get-docker.sh
```

### 5.1.9 如何制作容器镜像？

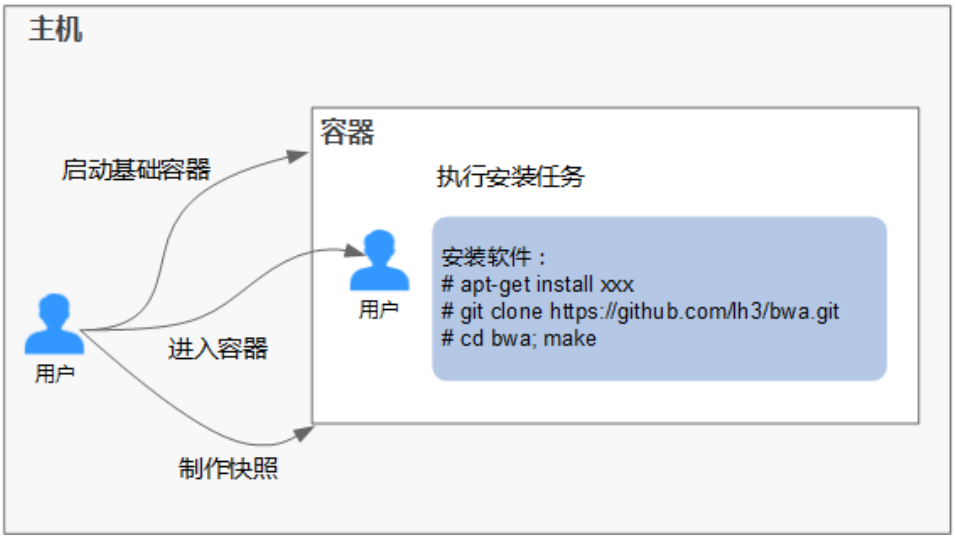
自己制作容器镜像，主要有两种方法：

- 制作快照方式获得镜像（偶尔制作的镜像）：在基础镜像上，比如 Ubuntu，先登录镜像系统并安装容器引擎软件，然后整体制作快照。

- Dockerfile 方式构建镜像( 经常更新的镜像 ):将软件安装的流程写成 DockerFile ,使用 Docker build 构建成容器镜像。

- **方法一：制作快照方式获得镜像**

如果后续镜像没有变化，可采用方法一制作镜像。




具体操作如下：

1. 找一台主机，安装容器引擎软件。

2. 启动一个空白的基础容器，并进入容器。

例如：启动一个 CentOS 的容器。

**docker run -it centos**

3. 执行安装任务。

**yum install XXX**
**git clone https://github.com/lh3/bwa.git**
**cd bwa;make**

> 说明：
>
> 请预先安装好 Git，并检查本机是否有 ssh key 设置。

4. 输入 **exit** 退出容器。

5. 制作快照。

   **docker commit -m "xx" -a "tsj" container-id tsj/image:tag**

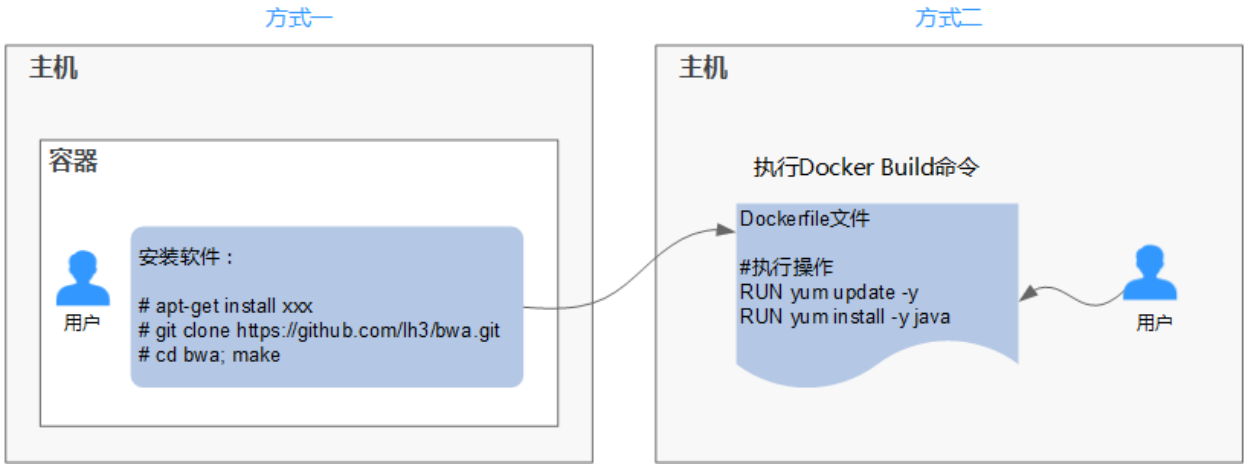
- -a：提交的镜像作者。
- container-id：操作步骤 2 中的容器 id。可以使用 **docker ps -a** 查询得到容器 id。
- -m：提交时的说明文字。
- tsj/image:tag：仓库名/镜像名:TAG 名。

6. 执行 **docker images** 可以查看到制作完成的容器镜像。

- **方法二：使用 Dockerfile 方式构建**

如果后续镜像经常变更（例如某个软件更新版本），则需要采用方法二制作镜像。若仍采用方法一制作镜像，则每次变更都需要重新执行方法一的命令，操作过程比较繁琐，所以建议使用自动化制作镜像的方法。

其实就是将方法一制作镜像的方法，用文件方式写出来（文件名为 DockerFile）。然后执行：**docker build -t tsj/image:tag.**命令（命令中“.”表示 DockerFile 文件的路径），自动完成镜像制作。



简单的 DockerFile 示例：

> 说明：如果依赖外部网络，请搭建网络环境，并保证网络可用。

```
#Version 1.0.1
FROM centos:latest
MAINTAINER ***u "***u@cloud.com"

#设置 root 用户为后续命令的执行者

USER root

#执行操作

RUN yum update -y
RUN yum install -y java

#使用&&拼接命令

RUN touch test.txt && echo "abc" >>abc.txt

#对外暴露端口

EXPOSE 80 8080 1038

#添加网络文件

ADD https://www.baidu.com/img/bd_logo1.png /opt/

#设置环境变量

ENV WEBAPP_PORT=9090

#设置工作目录

WORKDIR /opt/

#设置启动命令

ENTRYPOINT ["ls"]
```

```
#设置启动参数
CMD ["-a", "-l"]

#设置卷
VOLUME ["/data", "/var/www"]

#设置子镜像的触发操作
ONBUILD ADD . /app/src
ONBUILD RUN echo "on build excuted" >> onbuild.txt
```

详细的操作步骤可以参考：《容器镜像服务 最佳实践》。

## DockerFile 基本语法

- FROM：

指定待扩展的父级镜像（基础镜像）。除注释之外，文件开头必须是一个 FROM 指令，后面的指令便在这个父级镜像的环境中运行，直到遇到下一个 FROM 指令。通过添加多个 FROM 命令，可以在同一个 Dockerfile 文件中创建多个镜像。

- MAINTAINER：

声明创建镜像的作者信息：用户名、邮箱，非必须参数。

- RUN：

修改镜像的命令，常用来安装库、安装程序以及配置程序。一条 RUN 指令执行完毕后，会在当前镜像上创建一个新的镜像层，接下来对的指令会在新的镜像上继续执行。RUN 语句有两种形式：

  - **RUN yum update**：在**/bin/sh** 路径中执行的指令命令。
  - **RUN ["yum", "update"]**：直接使用系统调用 **exec** 来执行。

- **RUN yum update && yum install nginx** :使用&&符号将多条命令连接在同一条 RUN 语句中。

- EXPOSE：

指明容器内进程对外开放的端口，多个端口之间使用空格隔开。

运行容器时，通过设置参数**-P**（大写）即可将 EXPOSE 里所指定的端口映射到主机上其他的随机端口，其他容器或主机可以通过映射后的端口与此容器通信。

您也可以通过设置参数**-p**（小写）将 Dockerfile 中 EXPOSE 中没有列出的端口设置成公开。

- ADD：

向新镜像中添加文件，这个文件可以是一个主机文件，也可以是一个网络文件，也可以使一个文件夹。

- 第一个参数：源文件（夹）。
  - 如果是相对路径，必须是相对于 Dockerfile 所在目录的相对路径。
  - 如果是 URL，会将文件先下载下来，然后再添加到镜像里。
- 第二个参数：目标路径。
  - 如果源文件是主机上的 zip 或者 tar 形式的压缩文件，容器引擎会先解压缩，然后将文件添加到镜像的指定位置。
  - 如果源文件是一个通过 URL 指定的网络压缩文件，则不会解压。

- VOLUME：

在镜像里创建一个指定路径(文件或文件夹)的挂载点，这个容器可以来自主机或者其它容器。多个容器可以通过同一个挂载点共享数据，即便其中一个容器已经停止，挂载点也仍可以访问。

- WORKDIR：

为接下来执行的指令指定一个新的工作目录，这个目录可以是绝对目录，也可以是相对目录。根据需要，WORKDIR 可以被多次指定。当启动一个容器时，最后一条 WORKDIR 指令所指的目录将作为容器运行的当前工作目录。

- ENV：

设置容器运行的环境变量。在运行容器的时候，通过设置**-e** 参数可以修改这个环境变量值，也可以添加新的环境变量。

例如：

**docker run -e WEBAPP_PORT=8000 -e WEBAPP_HOST=www.example.com ...**

- CMD：

用来设置启动容器时默认运行的命令。

- ENTRYPOINT：

用来指定容器启动时的默认运行的命令。区别在于：运行容器时添加在镜像之后的参数，对 ENTRYPOINT 是拼接，CMD 是覆盖。

  - 若在 DockerFile 中指定了容器启动时的默认运行命令为 **ls -l** 则运行容器时默认启动命令为 **ls -l**，例如：

    - **ENTRYPOINT [ "ls", "-l"]**：指定容器启动时的程序及参数为 **ls -l** 。
    - **docker run centos**：当运行 centos 容器时，默认执行的命令是 **docker run centos ls -l**
    - **docker run centos -a**：当运行 centos 容器时拼接了-a 参数，则默认运行的命令是 **docker run centos ls -l -a**

- 若在 DockerFile 中指定了指定了容器启动时的默认运行命令为**--entrypoint**，则在运行容器时若需要替换默认运行命令，可以通过添加**--entrypoint** 参数来替换 Dockerfile 中的指定。例如：**docker run gutianlangyu/test --entrypoint echo "hello world"**

- USER：

为容器的运行及 RUN、CMD、ENTRYPOINT 等指令的运行指定用户或 UID。

- ONBUILD：

触发器指令。构建镜像时，容器引擎的镜像构建器会将所有的 ONBUILD 指令指定的命令保存到镜像的元数据中，这些命令在当前镜像的构建过程中并不会执行。只有新的镜像使用 FROM 指令指定父镜像为当前镜像时，才会触发执行。

使用 FROM 以这个 Dockerfile 构建出的镜像为父镜像，构建子镜像时：

**ONBUILD ADD . /app/src**：自动执行 **ADD . /app/src**

### 5.1.10 如何制作镜像压缩包？

使用 docker save 命令可将容器镜像制作成 tar 或 tar.gz 文件压缩包，具体命令格式如下：

**docker save [OPTIONS] IMAGE [IMAGE...]**

OPTIONS 说明：--output , -o，表示导出到文件。

示例：

```
$ docker save nginx:latest > nginx.tar
$ ls -sh nginx.tar
108M nginx.tar

$ docker save php:5-apache > php.tar.gz
$ ls -sh php.tar.gz
```

```
372M php.tar.gz

$ docker save --output nginx.tar nginx
$ ls -sh nginx.tar
108M nginx.tar

$ docker save -o nginx-all.tar nginx
$ docker save -o nginx-latest.tar nginx:latest
```

# 5.2 镜像管理类

## 5.2.1 为什么登录指令执行失败？

登录指令执行失败有以下几种情况：

1. 容器引擎未安装正确：

**解决方法：**重新安装容器引擎。

- 因容器镜像服务支持容器引擎 1.11.2 及以上版本上传镜像，建议下载对应版本。
- 安装容器引擎需要连接互联网，内网服务器需要绑定弹性 IP 后才能访问。

2. 临时登录指令已过期或登录指令中区域项目名称、AK、登录密钥错误：

**解决方法：**登录容器镜像服务控制台，在左侧菜单栏选择“我的镜像”，单击右侧“客户端上传”获取登录指令。

a. 获取临时的登录指令：单击“生成临时登录指令”，在弹出的页面中单击 复制登录指令。

b. 获取长期有效的登录指令：单击“如何获取长期有效登录指令”。

3. 登录指令中镜像仓库地址错误，报如下图所示错误：

**解决方法：**

a. 修改登录指令中的镜像仓库地址：访问“我的凭证”，在“项目列表”页签中查找当前区域对应的项目，镜像仓库地址为 ：swr.区域项目名称.mycloud.com，如北京一对应的镜像仓库地址为 swr.cn-north-1.mycloud.com。

获取临时的登录指令：

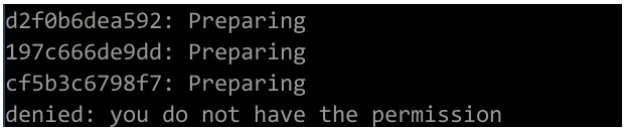
### 5.2.2 长期有效的登录指令与临时登录指令的区别是什么？

- 临时的登录指令代指 24 个小时后会过期失效，不能再被使用的登录指令。
- 长期有效的登录指令有效期为永久。

获取了长期有效的登录指令后，在有效期内的临时登录指令仍然可以使用。

### 5.2.3 为什么使用客户端上传镜像失败？

1. **问题现象：**使用客户端上传镜像，报如下图所示错误：

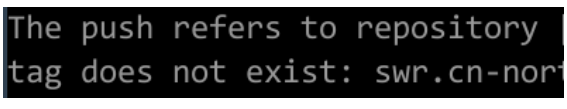



**问题原因：**使用未创建的组织名上传镜像，该组织名已被其他用户注册或当前 SWR 组织数量已超过配额。

**解决方法：**

- 该组织名已被其他用户注册时：建议您先创建组织然后再上传镜像。
- SWR 组织数量超过配额时：单个用户的组织数量限制为 5 个，您可以将镜像上传到已存在的组织下，也可以提交工单申请增加配额。

2. **问题现象：**使用客户端上传镜像，报如下图所示错误：



```
The push refers to repository [swr.cn-north-1.
An image does not exist locally with the tag:
```

**问题原因：**上传的镜像或镜像版本不存在。

**解决方法：**通过 docker images 查看本地镜像，确认要上传的镜像名称及版本后，重新上传镜像。

3. **问题现象：**使用客户端上传镜像，报如下图所示错误：

```
The push refers to repository [swr.cn-
d2f0b6dea592: Retrying in 1 second
197c666de9dd: Retrying in 1 second
cf5b3c6798f7: Retrying in 1 second
name invalid: 'repository' is invalid
```

**问题原因：**组织命名或镜像命名不规范。

**解决方法：**以下分别是组织名（namespace）和仓库名（repository）的命名正则表达式：

namespace：^([a-z]+(?:(?:(?:_|__|[-]*)[a-z0-9]+)+)?)$ ，长度范围为：1-64；

repository：^([a-z0-9]+(?:(?:(?:_|__|[-]*)[a-z0-9]+)+)?)$ ，长度范围为：1-128。

您可以按照上述命名规范，重新指定上传的组织和镜像名称。

### 5.2.4 为什么使用客户端上传镜像失败？

SWR 对镜像的命名和地址有严格的规范。如果镜像的命名不规范或镜像地址不规范都会导致镜像上传失败。

1. **问题现象：**通过页面上传镜像，出现“镜像格式不合法”的报错。

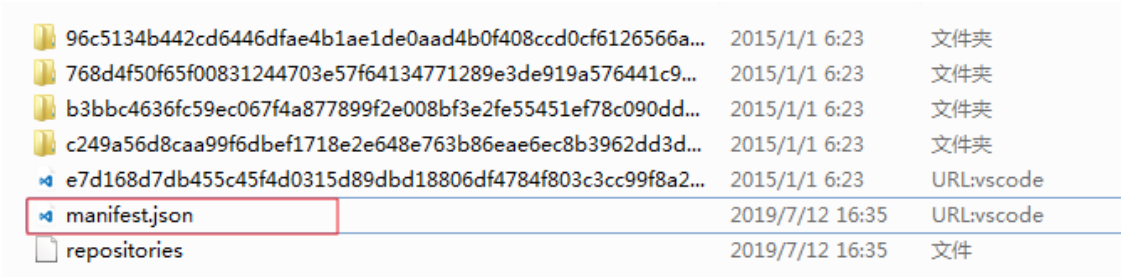
   **问题原因：**镜像地址不规范，导致上传失败。

   **解决方法：**镜像地址各个部分的含义如下，最后的 tag( 版本号 )可省略，如果省略则表示 latest 版本，其余部分均不可省略，且不可多余。

   样例：swr.cn-north-1.mycloud.com/repo_namespace/repo_name:tag

- swr.cn-north-1.mycloud.com 为容器镜像服务的镜像仓库地址。
- repo_namespace 为组织名称，命名正则表达式为^([a-z]+(?:(?:(?:_|__|[-]*)[a-z0-9]+)+)?)$ ，长度范围为：1-64。
- repo_name:tag 为镜像名称和版本号，镜像命名正则表达式为^([a-z0-9]+(?:(?:(?:_|__|[-]*)[a-z0-9]+)+)?)$ ，长度范围为：1-128。

您可以将镜像解压，打开文件 manifest.json 文件查看 RepoTags 字段的值是否符合上述规范。




2. **问题现象：**通过页面上传镜像，一直卡在上传界面直到超时。

   **问题原因：**镜像命名不规范，导致上传失败。

   **解决方法：**您可以按照镜像命名规范修改镜像名称后，重新上传镜像。

   **须知：**

   SWR 判定镜像名是否合法不是以用户在界面上传镜像时的文件名为依据，而是依据镜像包中的 repositories 和 manifest.json 文件。

### 5.2.5 为什么通过客户端和页面上传的镜像大小不一样？

使用客户端上传的镜像每一层 layer 进行了 tgz 压缩，而页面上传的镜像每一层 layer 只进行了打包，没有压缩，所以两种方式上传的镜像大小显示会不一致。

### 5.2.6 为什么通过客户端和 docker images 看到的镜像大小不一样？

使用客户端上传的镜像每一层 layer 进行了 tgz 压缩，而本地查看的镜像大小是没有经过压缩的，所以大小显示会不一致。

### 5.2.7 如何通过 API 上传镜像到 SWR？

SWR 暂时没有开放镜像上传的 API。您可以使用 docker push 上传镜像。即通过客户端的方式上传。

### 5.2.8 docker push 将镜像推送到 SWR 使用的什么协议？

docker push 将镜像推送到 SWR 使用的是 HTTPS 协议。

### 5.2.9 如何通过页面下载容器镜像？

目前 SWR 界面不支持直接下载镜像压缩包。建议您使用如下命令下载镜像：

**docker pull** [*镜像仓库地址*]/[*组织名称*]/[*镜像名称*:*版本名称*]

### 5.2.10 docker pull 下载的镜像存放在什么地方？如何拷贝？

docker pull 将镜像下载到节点本地上，您可以通过 docker save 命令将镜像保存成 tar 归档文件。