云容器

实验手册



华为技术有限公司

目录

[1 实验一 Docker容器的基本操作 3](#_Toc57618259)

[1.1 实验介绍 3](#_Toc57618260)

[1.1.1 关于本实验 3](#_Toc57618261)

[1.1.2 实验目的 3](#_Toc57618262)

[1.1.3 实验组网介绍 3](#_Toc57618263)

[1.2 购买实验资源 3](#_Toc57618264)

[1.2.1 登录华为公有云 3](#_Toc57618265)

[1.2.2 创建虚拟私有云 4](#_Toc57618266)

[1.2.3 创建并配置安全组 6](#_Toc57618267)

[1.2.4 购买弹性云服务器 7](#_Toc57618268)

[1.3 Docker的安装和配置 12](#_Toc57618269)

[1.3.1 Docker的安装和配置 12](#_Toc57618270)

[1.4 镜像的基本操作 13](#_Toc57618271)

[1.4.1 获取镜像 13](#_Toc57618272)

[1.4.2 查询及删除镜像 13](#_Toc57618273)

[1.5 容器的基本操作 15](#_Toc57618274)

[1.5.1 容器的创建与启停 15](#_Toc57618275)

[1.5.2 容器的运行 17](#_Toc57618276)

[1.5.3 进入容器 18](#_Toc57618277)

[1.5.4 删除容器 20](#_Toc57618278)

[1.6 私有镜像仓库搭建 21](#_Toc57618279)

[1.6.1 安装运行docker-registry 21](#_Toc57618280)

[2 实验二 Dockerfile的基本操作 24](#_Toc57618281)

[2.1 实验介绍 24](#_Toc57618282)

[2.1.1 关于本实验 24](#_Toc57618283)

[2.1.2 实验目的 24](#_Toc57618284)

[2.2 购买实验资源 24](#_Toc57618285)

[2.3 Dockerfile文件构建 24](#_Toc57618286)

[2.4 Nginx镜像验证 27](#_Toc57618287)

[2.5 Dockerfile指令的添加 28](#_Toc57618288)

[2.6 删除弹性云服务器及相关资源 29](#_Toc57618289)

[3 实验三 平台OA系统容器化部署 31](#_Toc57618290)

[3.1 实验介绍 31](#_Toc57618291)

[3.1.1 关于本实验 31](#_Toc57618292)

[3.1.2 实验目的 31](#_Toc57618293)

[3.1.3 实验资源 31](#_Toc57618294)

[3.2 通过Dockerfile构建OA系统镜像 32](#_Toc57618295)

[3.2.1 Docker安装 32](#_Toc57618296)

[3.2.2 Dockerfile构建OA系统镜像 33](#_Toc57618297)

[3.2.3 将镜像上传至华为镜像服务SWR 35](#_Toc57618298)

[3.3 CCE集群创建 38](#_Toc57618299)

[3.3.1 创建CCE集群 38](#_Toc57618300)

[3.4 CCE部署OA系统 41](#_Toc57618301)

[3.4.1 CCE部署OA系统 41](#_Toc57618302)

[4 释放实验资源 45](#_Toc57618303)

[4.1 删除云数据库GaussDB 45](#_Toc57618304)

[4.2 删除CCE集群 45](#_Toc57618305)

[4.3 删除ECS弹性云服务器 45](#_Toc57618306)

[4.4 删除IMS镜像 45](#_Toc57618307)

[4.5 资源检查 45](#_Toc57618308)

# 实验一 Docker容器的基本操作

### 实验介绍

### 关于本实验

通过本次实验，学员将完成Docker主机的安装和配置、镜像的搜索和下载、容器生命周期的基本管理、容器网络的管理。

### 实验目的

* 完成Docker的安装和配置。
* 完成Docker镜像的基本操作。
* 完成Docker Hub的基本操作。
* 完成Docker容器的基本操作。

### 实验组网介绍



说明：通过华为云弹性云服务器ECS实例构建Docker主机，完成Docker镜像、容器等基本操作。

## 购买实验资源

### 登录华为公有云

登录华为云[www.huaweicloud.com](http://www.huaweicloud.com)，并登录华为云账号。



输入对应的用户名和密码后，单击登录按钮。



### 创建虚拟私有云

选择产品基础服务网络虚拟私有云 VPC。



点击访问“控制台”



在网络控制台界面中，点击“创建虚拟私有云”



填写如下配置信息，然后点击“立即创建”。

* 区域：华北-北京四
* 名称：vpc-docker
* 网段：默认



子网配置：

* 可用区：可用区1
* 名称：subnet-docker
* 子网网段：默认



返回到网络控制台，可看见VPC与子网已创建。

### 创建并配置安全组

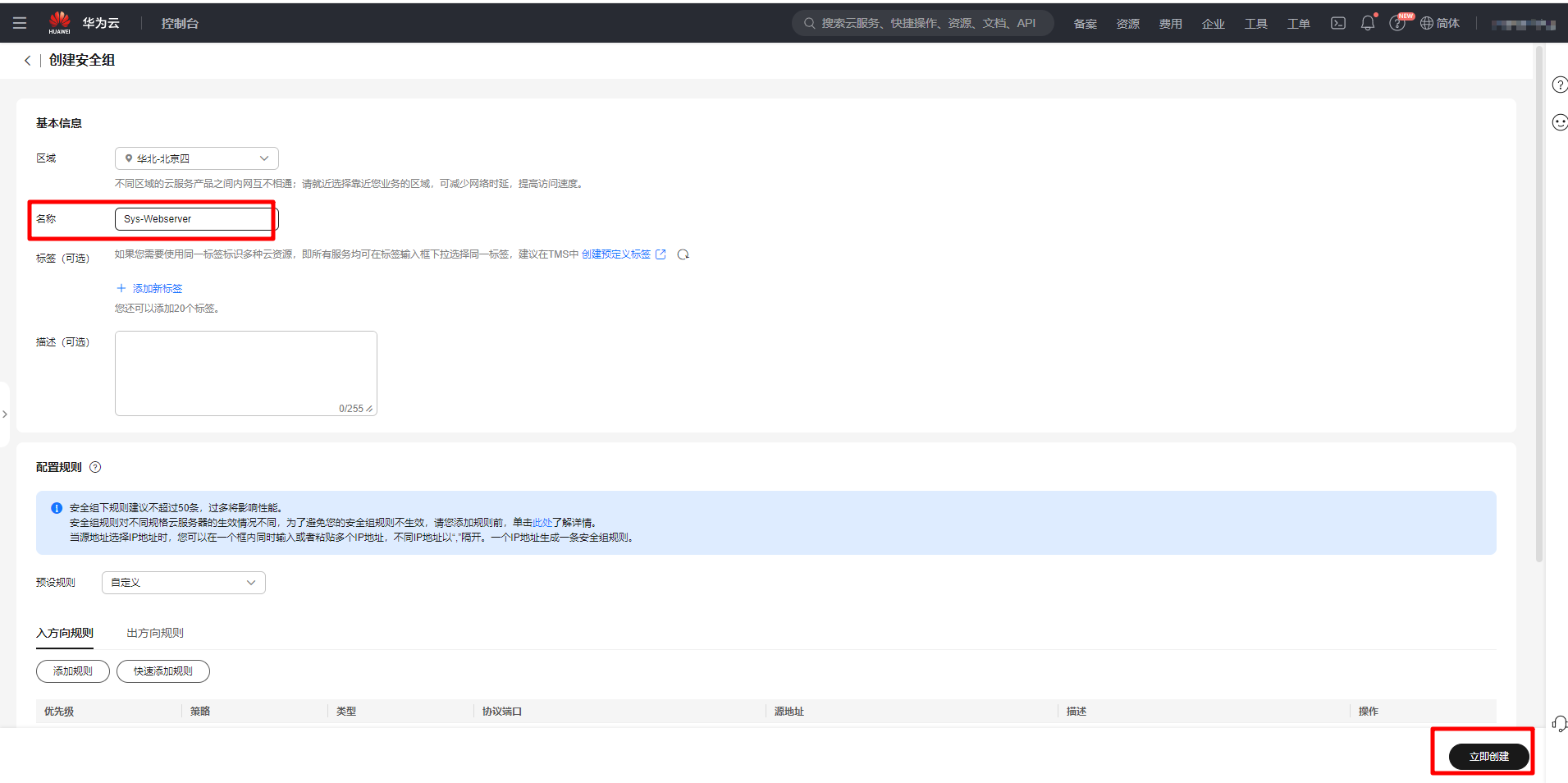
展开网络控制台左侧列表的访问控制，选择“安全组”，进入安全组页签。点击右上角“创建安全组”。



在“创建安全组”窗口中配置模板选择为“通用Web服务器”，名称设置为“Sys-Webserver”，然后点击“确定”。默认放通22、3389、80、443端口和ICMP协议。



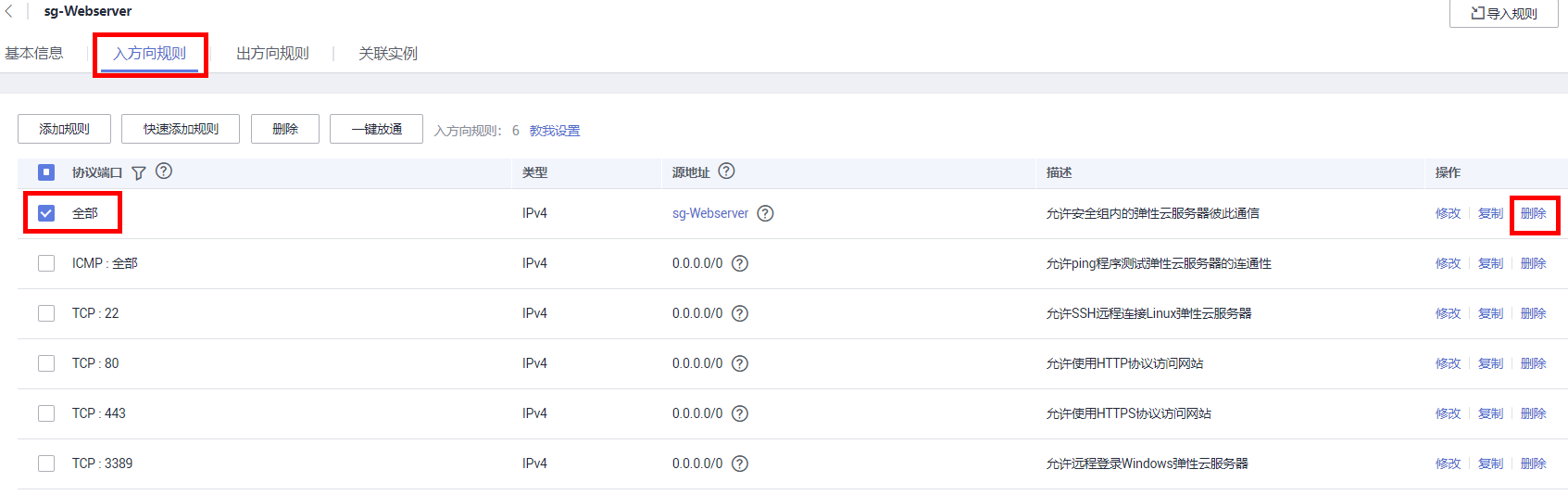
更正：目前创建安全组时已经没有“通用Web服务器”这个模板可以供选择。



单击安全组名称“Sys-Webserver”，进入安全组规则配置界面。

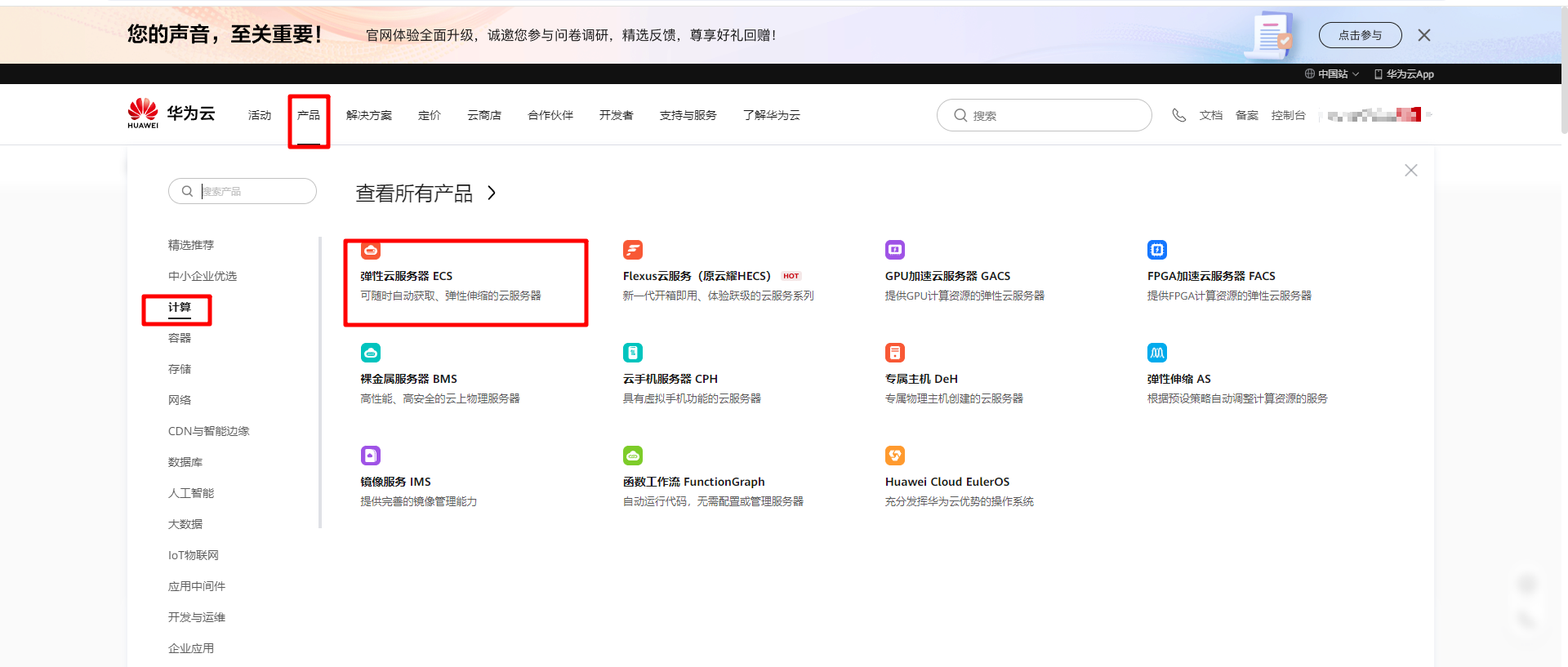


点击“入方向规则”，为安全起见，将第一行“全部”放通删除，完成安全组配置。



### 购买弹性云服务器

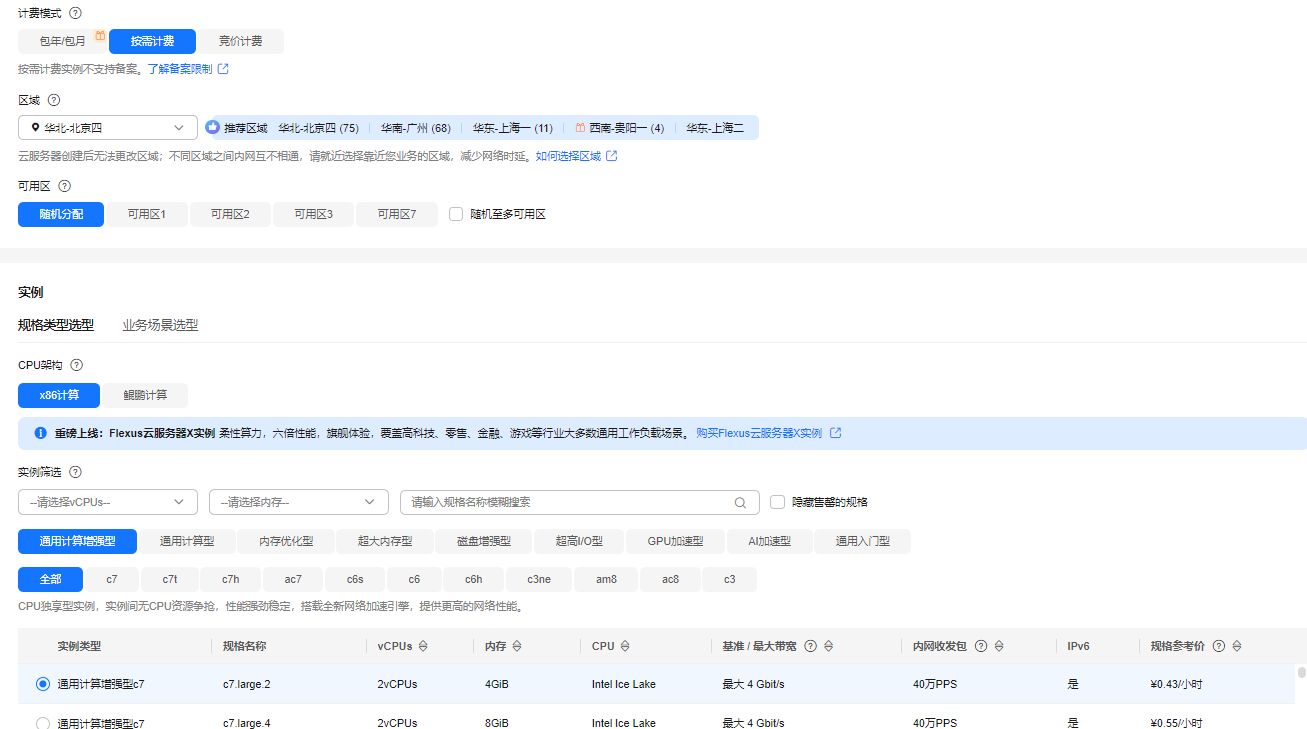
选择“产品“>“计算”>“弹性云服务器ECS”，然后点击“立即购买”。





填写如下基础配置信息，然后点击“下一步”。

* 计费模式：按需计费
* 区域：华北-北京四
* 可用区：随机分配
* CPU架构：x86计算
* 规格：通用计算增强型 | c7.large.2 | 2vCPUs | 4GiB
* 镜像：公共镜像CentOS 7.9 64bit (40GB)
* 系统盘：通用型SSD | 40G





填写如下网络配置信息，然后点击“下一步”。

* 网络：选择已创建的网络和子网，如vpc-docker和subnet-docker
* 安全组：Sys-Webserver
* 弹性公网IP：现在购买
* 规格：全动态BGP
* 计费方式：按带宽计费
* 带宽：5 Mbit/s



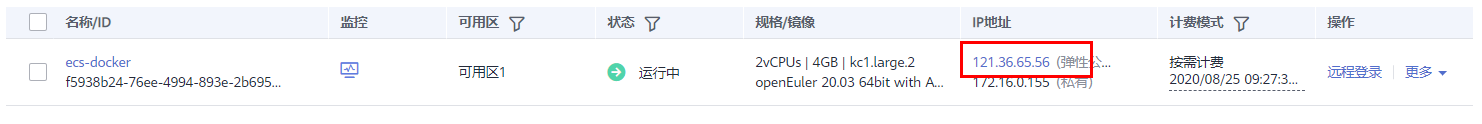
填写如下高级配置信息，然后点击“下一步”。

* 云服务器名称：ecs-docker
* 登录凭证：密码
* 密码/确认密码：自行设置密码，要求8位以上且包含大小写字母、数字、特殊字符中三种以上字符
* 企业项目：default

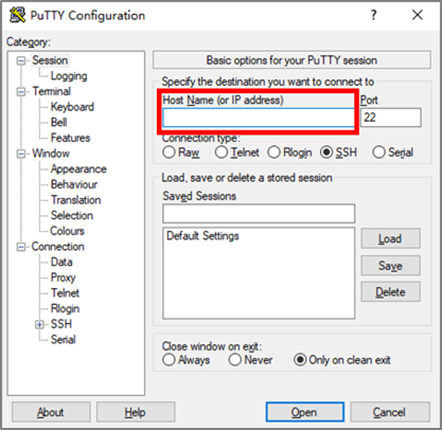


在确认配置界面，勾选“我已经阅读…….”后，点击“立即购买”。

购买完成后，返回控制台，稍等片刻，可以看到ECS状态为运行中，复制弹性IP地址。



打开putty，输入弹性公网IP地址后，点击open。



用root用户名，和之前设置的密码登录ECS。

出现“welcome to Huawei Cloud Service”说明登录成功。

检查内核版本。

[root@ecs-docker ~]# uname -r

3.10.0-1160.119.1.el7.x86\_64

若您安装过docker，需要先删掉，之后再安装依赖。

[root@ecs-docker ~]# sudo yum remove docker docker-common docker-selinux docker-engine

[root@ecs-docker ~]# sudo yum install -y yum-utils device-mapper-persistent-data lvm2

下载repo文件，替换仓库地址。

[root@ecs-docker ~]# wget -O /etc/yum.repos.d/docker-ce.repo https://mirrors.huaweicloud.com/docker-ce/linux/centos/docker-ce.repo

[root@ecs-docker ~]# sudo sed -i 's+download.docker.com+mirrors.huaweicloud.com/docker-ce+' /etc/yum.repos.d/docker-ce.repo

## Docker的安装和配置

### Docker的安装和配置

输入以下命令，更新索引文件并安装docker。

[root@ecs-docker ~]# sudo yum makecache fast

[root@ecs-docker ~]# sudo yum install docker-ce

…

Installed:

docker-ce.x86\_64 3:26.1.4-1.el7

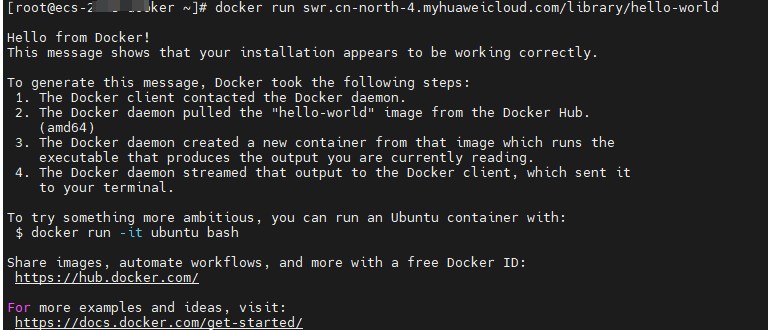
Complete!

出现“Complete”说明下载成功。

启动Docker 后台服务。

测试运行hello-world。（docker pull swr.cn-north-4.myhuaweicloud.com/library/hello-world:latest）

[root@ecs-docker ~]# docker run swr.cn-north-4.myhuaweicloud.com/library/hello-world



若能正常输出信息，则说明安装成功。

查看下载的hello-world镜像。

[root@ecs-docker ~]# docker images



## 镜像的基本操作

### 获取镜像

从Docker镜像仓库获取镜像的命令是docker pull，其命令格式为：

docker pull [选项] [Docker Registry 地址[:端口号]/]仓库名[:标签]

下载nginx镜像。

由于直接从国外下载镜像有网络限制，建议先按照文档配置华为SWR镜像加速器：[https://console.huaweicloud.com/swr/?region=cn-north-4#/swr/mirror](https://console.huaweicloud.com/swr/?region=cn-north-4" \l "/swr/mirror)

vi /etc/docker/daemon.json

{

"registry-mirrors": [ "https://7046a839d8b94ca190169bc6f8b55644.mirror.swr.myhuaweicloud.com" ]

}

**systemctl restart docker**

[root@ecs-docker ~]# docker pull nginx

Using default tag: latest

latest: Pulling from library/nginx

3742235f1882: Pull complete

c937a3b553c0: Pull complete

029f3c29ff1e: Pull complete

25ba16bee64a: Pull complete

e1f88c638905: Pull complete

Digest: sha256:b0ad43f7ee5edbc0effbc14645ae7055e21bc1973aee5150745632a24a752661

Status: Downloaded newer image for nginx:latest

### 查询及删除镜像

想要查询已经下载的镜像，可以用docker images 或docker images ls命令。

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

nginx latest 783da3d04b36 11 days ago 126MB

hello-world latest a29f45ccde2a 7 months ago 9.14kB

列表包含了仓库名、标签、镜像ID、创建时间以及所占用的空间。

查询部分镜像。

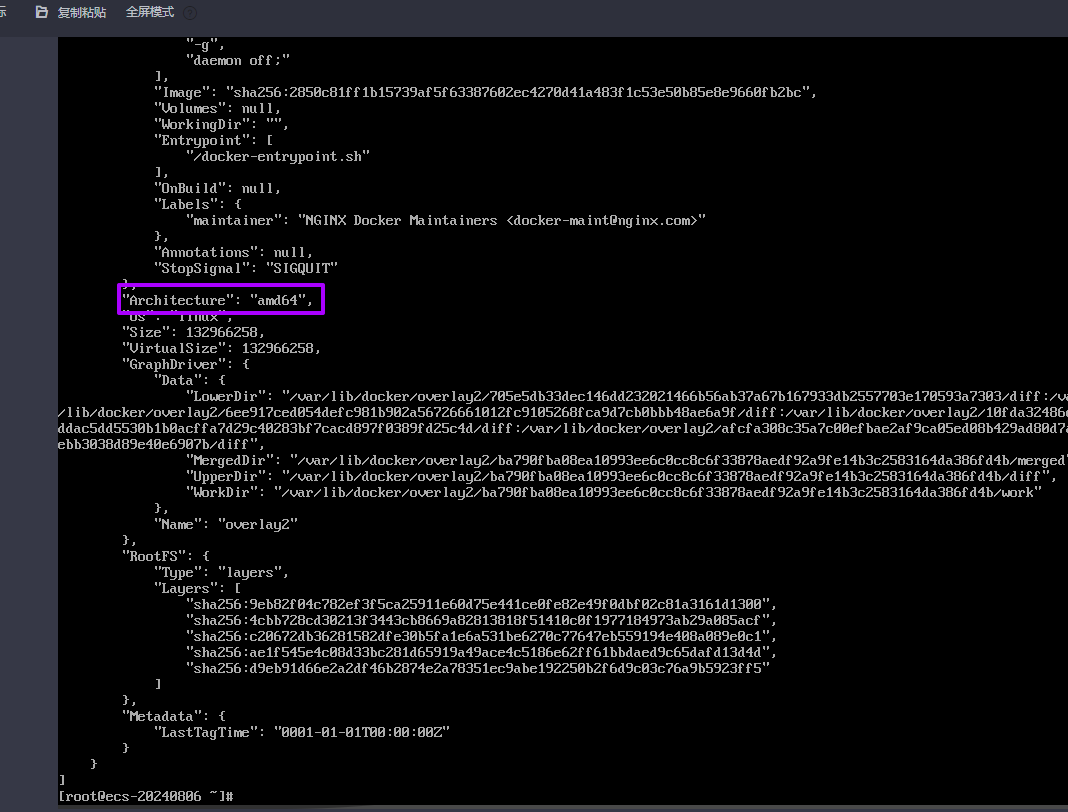
[root@ecs-docker ~]# docker image ls nginx

REPOSITORY TAG IMAGE ID CREATED SIZE

nginx latest 783da3d04b36 11 days ago 126MB

查看镜像的大小。

注：docker inspect查看镜像支持的CPU架构



[root@ecs-docker ~]# docker system df

TYPE TOTAL ACTIVE SIZE RECLAIMABLE

Images 2 1 125.9MB 125.9MB (99%)

Containers 1 0 0B 0B

Local Volumes 0 0 0B 0B

Build Cache 0 0 0B 0B

* 删除镜像命令格式说明：

如果要删除本地一个或多个镜像，可以使用 docker rmi或docker image rm 命令，其格式为：

docker rmi [选项] <镜像1> [<镜像2> ...]

通过短ID或镜像完整ID删除镜像。

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

nginx latest 783da3d04b36 11 days ago 126MB

hello-world latest a29f45ccde2a 7 months ago 9.14kB

[root@ecs-docker ~]# docker rmi 783

Untagged: nginx:latest

Untagged: nginx@sha256:b0ad43f7ee5edbc0effbc14645ae7055e21bc1973aee5150745632a24a752661

Deleted: sha256:783da3d04b368e8d6d4c74b82133467f2d2bc237dfc151371fb9818c98e66623

Deleted: sha256:d75ccb2e66eda5be7c48e39db34a79022b146010d4632ccfdd08b413b8234340

Deleted: sha256:567247fef93be15eb6f672d0c646f802f42bc377b63199fd84bb2588ea022ca2

Deleted: sha256:428117fcab0344e41d0f51e5aede787b2d1579ca75be7de741d3a76527bdf385

Deleted: sha256:d314a6060a72af8ae952926664cf22a54ccf8abb927ac62acdc81f9dbd15e3c1

Deleted: sha256:5a227e6cfbaa39a8568ca64d6e99623270b6d7c3a6f15d02ff030fb3a0f4a353

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

hello-world latest a29f45ccde2a 7 months ago 9.14kB

docker images查看删除后镜像

通过仓库名+标签删除镜像，如果删除的镜像已经产生了容器实例，不管容器实例是否启动都会提示无法删除，因为镜像被占用。这时需要先删除容器实例或添加删除参数

-f :强制删除，一般用于被占用的删除。

[root@ecs-docker ~]# docker image rm hello-world

Error response from daemon: conflict: unable to remove repository reference "hello-world" (must force) - container 0ee63008bf98 is using its referenced image a29f45ccde2a

[root@ecs-docker ~]# docker image rm hello-world -f

Untagged: hello-world:latest

Untagged: hello-world@sha256:7f0a9f93b4aa3022c3a4c147a449bf11e0941a1fd0bf4a8e6c9408b2600777c5

Deleted: sha256:a29f45ccde2ac0bde957b1277b1501f471960c8ca49f1588c6c885941640ae60

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

## 容器的基本操作

### 容器的创建与启停

创建一个基于httpd镜像的新容器。若主机中没有对应镜像，将会从docker Hub中拉取最新镜像。

[root@ecs-docker ~]# docker create httpd

Unable to find image 'httpd:latest' locally

latest: Pulling from library/httpd

3742235f1882: Pull complete

cc46a80438d5: Pull complete

ec0db235a655: Pull complete

7de1ef210508: Pull complete

0e4a7a5e30f2: Pull complete

Digest: sha256:3cbdff4bc16681541885ccf1524a532afa28d2a6578ab7c2d5154a7abc182379

Status: Downloaded newer image for httpd:latest

cb98db916abf990edf86382fb58351221bfb479b202b3c736c52808805ee0be9

查看容器信息，docker ps命令表示列出所有在运行的容器信息；docker ps -a 命令显示所有的容器，包括未运行的。

[root@ecs-docker ~]# docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" 24 seconds ago Created inspiring\_haslett

0ee63008bf98 a29f45ccde2a "/hello" 31 minutes ago Exited (0) 31 minutes ago elated\_kapitsa

可以查询到容器ID为cb98db916abf，容器名称为inspiring\_haslett，容器状态为已创建Created。

根据显示的容器ID或容器名称启动容器。

[root@ecs-docker ~]# docker start cb98db916abf

cb98db916abf

[root@ecs-docker ~]# docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" 2 minutes ago Up 5 seconds 80/tcp inspiring\_haslett

0ee63008bf98 a29f45ccde2a "/hello" 33 minutes ago Exited (0) 33 minutes ago elated\_kapitsa

停止容器运行。

[root@ecs-docker ~]# docker stop cb98db916abf

cb98db916abf

[root@ecs-docker ~]# docker ps -a

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" 3 minutes ago Exited (0) 3 seconds ago inspiring\_haslett

0ee63008bf98 a29f45ccde2a "/hello" 34 minutes ago Exited (0) 34 minutes ago elated\_kapitsa

重启容器。

[root@ecs-docker ~]# docker restart cb98db916abf

cb98db916abf

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" 5 minutes ago Up 3 seconds 80/tcp inspiring\_haslett

暂停容器。

[root@ecs-docker ~]# docker pause cb98db916abf

cb98db916abf

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" 6 minutes ago Up About a minute (Paused) 80/tcp inspiring\_haslett

恢复暂停的容器。

[root@ecs-docker ~]# docker unpause cb98db916abf

cb98db916abf

杀掉容器进程，强制停止容器。

[root@ecs-docker ~]# docker kill cb98db916abf

cb98db916abf

启动容器，给容器重新命名。

[root@ecs-docker ~]# docker start cb98db916abf

cb98db916abf

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" 7 minutes ago Up 5 seconds 80/tcp inspiring\_haslett

docker rename <container ID> new-name

[root@ecs-docker ~]# docker rename cb98db916abf myhttpd

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" 8 minutes ago Up About a minute 80/tcp myhttpd

### 容器的运行

运行一个新容器，该容器基于ubuntu:14.04。若主机中没有对应镜像，将会从docker Hub中拉取最新镜像。

[root@ecs-docker ~]# docker run ubuntu:14.04 /bin/echo 'Hello world'

Unable to find image 'ubuntu:14.04' locally

14.04: Pulling from library/ubuntu

d1a5a1e51f25: Pull complete

9592077f8570: Pull complete

e7b48639f381: Pull complete

Digest: sha256:ffc76f71dd8be8c9e222d420dc96901a07b61616689a44c7b3ef6a10b7213de4

Status: Downloaded newer image for ubuntu:14.04

Hello world

下面的命令则启动一个 bash 终端，允许用户进行交互。

[root@ecs-docker ~]# docker run -it ubuntu:14.04 /bin/bash

root@b4774b865e86:/# pwd

/

root@b4774b865e86:/# ls

bin boot dev etc home lib media mnt opt proc root run sbin srv sys tmp usr var

启动容器后，可以输入pwd命令查看当前目录，输入ls查看当前目录下有哪些文件夹。

-it ：这是两个参数，一个是 -i ：交互式操作，一个是 -t终端。我们这里打算进入bash执行一些命令并查看返回结果，因此我们需要交互式终端。

bash ：这里我们希望有个交互式Shell，因此用的是bash。

执行exit命令可以退出容器。

root@b4774b865e86:/# exit

exit

[root@ecs-docker ~]#

更多的时候，需要让 Docker 在后台运行而不是直接把执行命令的结果输出在当前宿主机下。此时，可以通过添加-d参数来实现。下面看下两者的区别：

* 如果不使用-d参数运行容器：

[root@ecs-docker ~]# docker run ubuntu:14.04 /bin/sh -c "while true; do echo hello world; sleep 1; done"

hello world

hello world

hello world

hello world

hello world

hello world

^C[root@ecs-docker ~]#

* 如果使用了-d参数运行容器：

[root@ecs-docker ~]# docker run -d ubuntu:14.04 /bin/sh -c "while true; do echo hello world; sleep 1; done"

fd9662ff976aa0027777b9da798dfaa6d07de28a34fee58ba11d66a0e9c74ba0

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

fd9662ff976a ubuntu:14.04 "/bin/sh -c 'while t…" 9 seconds ago Up 7 seconds focused\_mcnulty

cb98db916abf httpd "httpd-foreground" 35 minutes ago Up 28 minutes 80/tcp myhttpd

通过docker logs [container ID or NAMES]获取容器的输出信息。

[root@ecs-docker ~]# docker logs fd9662ff976a

hello world

hello world

hello world

hello world

hello world

hello world

hello world

hello world

hello world

hello world

hello world

hello world

…

### 进入容器

某些时候需要进入容器进行操作，可以使用docker attach命令或docker exec命令。

* 使用docker attach命令进入容器。

启动一个容器。

[root@ecs-docker ~]# docker run -dit ubuntu:14.04

478f2e622a633e988ee9068f4bba8c7ea3b197957694b75ff9ea56020dfc8328

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

478f2e622a63 ubuntu:14.04 "/bin/bash" 8 seconds ago Up 7 seconds unruffled\_khayyam

fd9662ff976a ubuntu:14.04 "/bin/sh -c 'while t…" 13 minutes ago Up 13 minutes focused\_mcnulty

cb98db916abf httpd "httpd-foreground" About an hour ago Up 41 minutes 80/tcp myhttpd

使用attach命令，直接进入容器启动命令的终端。

docker attach <容器ID> or <容器名称>

[root@ecs-docker ~]# docker attach 478f2e622a63

root@478f2e622a63:/#

root@478f2e622a63:/# ps

PID TTY TIME CMD

1 pts/0 00:00:00 bash

17 pts/0 00:00:00 ps

执行exit命令可以退出容器。

root@478f2e622a63:/# exit

exit

[root@ecs-docker ~]#

* 使用docker exec命令进入容器。

启动一个容器。

[root@ecs-docker ~]# docker run -dit ubuntu:14.04

15e1c17ee21c124df4905b5ca355edfb5e5ece8c3df481a4df426ae9712ce524

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

15e1c17ee21c ubuntu:14.04 "/bin/bash" 4 seconds ago Up 4 seconds trusting\_euler

fd9662ff976a ubuntu:14.04 "/bin/sh -c 'while t…" 16 minutes ago Up 16 minutes focused\_mcnulty

cb98db916abf httpd "httpd-foreground" About an hour ago Up 44 minutes 80/tcp myhttpd

通过docker exec进入容器

docker exec -it <container ID> bash

[root@ecs-docker ~]# docker exec -it 15e1c17ee21c bash

root@15e1c17ee21c:/# ps

PID TTY TIME CMD

17 pts/1 00:00:00 bash

31 pts/1 00:00:00 ps

执行exit命令可以退出容器。

root@15e1c17ee21c:/# exit

exit

[root@ecs-docker ~]#

### 删除容器

使用docker rm 来删除一个处于终止状态的容器。若容器没有退出则无法删除，需要先停止容器。

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

15e1c17ee21c ubuntu:14.04 "/bin/bash" 3 minutes ago Up 3 minutes trusting\_euler

fd9662ff976a ubuntu:14.04 "/bin/sh -c 'while t…" 20 minutes ago Up 20 minutes focused\_mcnulty

cb98db916abf httpd "httpd-foreground" About an hour ago Up About an hour 80/tcp myhttpd

[root@ecs-docker ~]# docker rm 15e1c17ee21c

Error response from daemon: You cannot remove a running container 15e1c17ee21c124df4905b5ca355edfb5e5ece8c3df481a4df426ae9712ce524. Stop the container before attempting removal or force remove

[root@ecs-docker ~]# docker stop 15e1c17ee21c

15e1c17ee21c

[root@ecs-docker ~]# docker rm 15e1c17ee21c

15e1c17ee21c

使用docker rm -f来删除一个处于运行状态的容器。若容器暂停则无法删除，需要先恢复容器。

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

fd9662ff976a ubuntu:14.04 "/bin/sh -c 'while t…" 22 minutes ago Up 22 minutes focused\_mcnulty

cb98db916abf httpd "httpd-foreground" About an hour ago Up About an hour 80/tcp myhttpd

[root@ecs-docker ~]# docker rm -f fd9662ff976a

fd9662ff976a

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

cb98db916abf httpd "httpd-foreground" About an hour ago Up About an hour 80/tcp myhttpd

删除所有已终止的容器。

[root@ecs-docker ~]# docker rm -v $(docker ps -aq -f status=exited)

478f2e622a63

7866eee50250

b4774b865e86

3a965fbfef80

0ee63008bf98

## 私有镜像仓库搭建

docker-registry是官方提供的工具，可以用于构建私有的镜像仓库。

### 安装运行docker-registry

获取官方registry 镜像并运行容器。

[root@ecs-docker ~]# docker run -d -p 5000:5000 --restart=always --name registry registry

Unable to find image 'registry:latest' locally

latest: Pulling from library/registry

29e5d40040c1: Pull complete

3e44ffe5a703: Pull complete

bc788ee7896b: Pull complete

f89902f2529a: Pull complete

215110f71eb4: Pull complete

Digest: sha256:8be26f81ffea54106bae012c6f349df70f4d5e7e2ec01b143c46e2c03b9e551d

Status: Downloaded newer image for registry:latest

77eb25c57c782470b4c19b3bf0a3b2c189f3df73a1f083d745ad097cf6bc0792

[root@ecs-docker ~]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

77eb25c57c78 registry "/entrypoint.sh /etc…" 7 seconds ago Up 6 seconds 0.0.0.0:5000->5000/tcp registry

cb98db916abf httpd "httpd-foreground" About an hour ago Up About an hour 80/tcp myhttpd

docker ps查看容器是否已运行。

先在本机查看已有的镜像。此实验以ubuntu:14.04为例。

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

httpd latest dfd370c3ce20 2 weeks ago 155MB

registry latest 1525b096095b 2 months ago 24.5MB

ubuntu 14.04 104c127e7519 8 months ago 187MB

以ubuntu:14.04基础镜像为例，通过docker tag命令将基础镜像ubuntu:14.04镜像进行标记。

docker tag <基础镜像名称：标签> 127.0.0.1:5000/<镜像名称>:<镜像标签>

[root@ecs-docker ~]# docker tag ubuntu:14.04 127.0.0.1:5000/myubuntu:14.04

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

httpd latest dfd370c3ce20 2 weeks ago 155MB

registry latest 1525b096095b 2 months ago 24.5MB

127.0.0.1:5000/myubuntu 14.04 104c127e7519 8 months ago 187MB

ubuntu 14.04 104c127e7519 8 months ago 187MB

使用docker push 上传标记的镜像。

docker push 127.0.0.1:5000/<镜像名称>:<镜像标签>

[root@ecs-docker ~]# docker push 127.0.0.1:5000/myubuntu:14.04

The push refers to repository [127.0.0.1:5000/myubuntu]

400588327aed: Pushed

004444c9ba25: Pushed

59199d90878e: Pushed

14.04: digest: sha256:79b0f81ad6fc8f0bced3919beee0a79d60d1c8358911188d0787b1f155656d84 size: 945

用curl查看仓库中的镜像。

[root@ecs-docker ~]# curl 127.0.0.1:5000/v2/\_catalog

{"repositories":["myubuntu"]}

删除已有镜像，再尝试从私有仓库中下载这个镜像。

docker image rm 127.0.0.1:5000/<镜像名称>:<镜像标签>

[root@ecs-docker ~]# docker image rm 127.0.0.1:5000/myubuntu:14.04

Untagged: 127.0.0.1:5000/myubuntu:14.04

Untagged: 127.0.0.1:5000/myubuntu@sha256:79b0f81ad6fc8f0bced3919beee0a79d60d1c8358911188d0787b1f155656d84

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

httpd latest dfd370c3ce20 2 weeks ago 155MB

registry latest 1525b096095b 2 months ago 24.5MB

ubuntu 14.04 104c127e7519 8 months ago 187MB

[root@ecs-docker ~]# docker pull 127.0.0.1:5000/myubuntu:14.04

14.04: Pulling from myubuntu

Digest: sha256:79b0f81ad6fc8f0bced3919beee0a79d60d1c8358911188d0787b1f155656d84

Status: Downloaded newer image for 127.0.0.1:5000/myubuntu:14.04

[root@ecs-docker ~]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

httpd latest dfd370c3ce20 2 weeks ago 155MB

registry latest 1525b096095b 2 months ago 24.5MB

127.0.0.1:5000/myubuntu 14.04 104c127e7519 8 months ago 187MB

ubuntu 14.04 104c127e7519 8 months ago 187MB

# 实验二 Dockerfile的基本操作

## 实验介绍

### 关于本实验

本实验通过Dockerfile来构建镜像，了解Dockerfile镜像构建过程。

### 实验目的

* 理解Dockerfile命令含义
* 了解Dockerfile镜像构建过程
* 掌握如何通过Dockerfile构建镜像

## 购买实验资源

备注: 本实验购买的ECS实例与实验一规格相同，如实验一资源未删除，则无需重复购买。但在安全组配建上，需要添加端口“TCP:81”，即可直接开始实验。

## Dockerfile文件构建

下载基础镜像nginx。

[root@ecs-docker ~]# docker pull nginx:alpine-perl

alpine-perl: Pulling from docker.io/library/nginx

Digest: sha256:93e17ed1880aadc5e3d94e537570101847804add7293897a7a6b226a10dd7586

Status: Downloaded newer image for docker.io/nginx:alpine-perl

创建nginx\_demo文件夹。

[root@ecs-docker ~]# pwd

/root

[root@ecs-docker ~]# mkdir nginx\_demo

[root@ecs-docker ~]# ls

nginx\_demo

创建Dockerfile文件

[root@ecs-docker nginx\_demo]# vim Dockerfile

输入i，编辑如下内容到Dockerfile中。

# base image

FROM nginx:alpine-perl

# MAINTAINER

MAINTAINER test @huawei.com

RUN chmod -R 755 /usr/lib/nginx/

EXPOSE 80

完成后点击“Ecs”，然后输入:wq保存退出文档。

通过Dockerfile创建nginx镜像。

[root@ecs-docker nginx\_demo]# docker build -t my\_nginx:v1 .

Sending build context to Docker daemon 984.6 kB

Step 1/4 : FROM nginx:alpine-perl

---> 31b981f47c18

Step 2/4 : MAINTAINER test @huawei.com

---> Running in bb8c75c9d1c7

---> c5ae59a413fe

Removing intermediate container bb8c75c9d1c7

Step 3/4 : RUN chmod -R 755 /usr/lib/nginx/

---> Running in e74b62500b5a

---> eb1318096ee1

Removing intermediate container e74b62500b5a

Step 4/4 : EXPOSE 80

---> Running in 3df008057bf4

---> d9e6238a664b

Removing intermediate container 3df008057bf4

Successfully built d9e6238a664b

查看构建的镜像

[root@ecs-docker nginx\_demo]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

my\_nginx v1 a84254a38800 3 minutes ago 590MB

centos 7 190630274c4b 2 weeks ago 289MB

httpd latest dfd370c3ce20 2 weeks ago 155MB

registry latest 1525b096095b 2 months ago 24.5MB

127.0.0.1:5000/myubuntu 14.04 104c127e7519 8 months ago 187MB

ubuntu 14.04 104c127e7519 8 months ago 187MB

说明nginx镜像创建成功，镜像名称：my\_nginx，标签v1，镜像ID a84254a38800。

## Nginx镜像验证

通过构建的镜像，运行一个容器，将端口进行映射。

[root@ecs-docker nginx\_demo]# docker run -d -p 80:80 my\_nginx:v1 /usr/sbin/nginx -g "daemon off;"

610c3a2bf220ea9a29fa8ac851bd0d968719951f9be891e401c36687ab4b3b30

命令说明：

* -p 80:80 : 将容器的80端口映射到主机的80端口
* /usr/sbin/nginx -g "daemon off;" : 设置nginx非daemon守护进程,否则容器会自动退出

查看容器状态

[root@ecs-docker nginx\_demo]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

610c3a2bf220 my\_nginx:v1 "/usr/sbin/ngin…" 13 seconds ago Up 12 seconds 0.0.0.0:80->80/tcp sharp\_noyce

77eb25c57c78 registry "/entrypoint.sh /etc…" 28 minutes ago Up 28 minutes 0.0.0.0:5000->5000/tcp registry

cb98db916abf httpd "httpd-foreground" 2 hours ago Up 2 hours 80/tcp myhttpd

打开浏览器，输入ecs-docker弹性IP地址，默认端口为80，进行验证，显示“Welcome to nginx!”，说明容器运行正常。



## Dockerfile指令的添加

我们也可以基于以上Dockerfile文件依次添加其他的指令进行构建，比如我们可以添加CMD命令，设置nginx非daemon守护进程，这样容器启动时不会自动退出。

创建Dockerfile文件

[root@ecs-docker nginx\_demo]# vim Dockerfile

输入i进入编辑模式，在原有Dockerfile基础上，增加如下内容到Dockerfile最后一行。

CMD /usr/sbin/nginx -g "daemon off;"

通过Dockerfile创建新的nginx镜像。

[root@ecs-docker nginx\_demo]# docker build -t my\_nginx:v2 .

Sending build context to Docker daemon 984.6 kB

Step 1/5 : FROM nginx:alpine-perl

---> 31b981f47c18

Step 2/5 : MAINTAINER test @huawei.com

---> Using cache

---> c5ae59a413fe

Step 3/5 : RUN chmod -R 755 /usr/lib/nginx/

---> Using cache

---> eb1318096ee1

Step 4/5 : EXPOSE 80

---> Using cache

---> d9e6238a664b

Step 5/5 : CMD /usr/sbin/nginx -g "daemon off;"

---> Running in 13649568eaf8

---> 66251f415268

Removing intermediate container 13649568eaf8

Successfully built 66251f415268

查看构建的镜像。

[root@ecs-docker nginx\_demo]# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

my\_nginx v2 85473c1bda55 About a minute ago 597MB

my\_nginx v1 f02bbcca678a 3 minutes ago 597MB

centos 7 190630274c4b 2 weeks ago 289MB

通过构建的镜像，运行一个容器，将端口进行映射，将容器的80端口映射到主机的81端口。

[root@ecs-docker nginx\_demo]# docker run -d -p 81:80 my\_nginx:v2

67a376ebb5c6b43adb92b6b9ff01c8e3faf85f8a54516df0ebed88cfa932f032

查看容器状态。

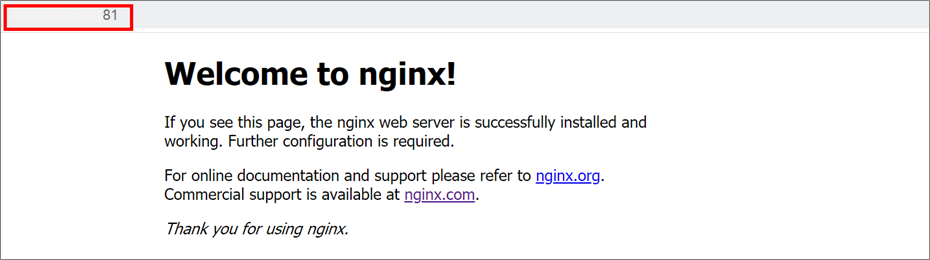
[root@ecs-docker nginx\_demo]# docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

67a376ebb5c6 my\_nginx:v2 "/bin/sh -c '/usr/lo…" 20 seconds ago Up 19 seconds 0.0.0.0:81->80/tcp optimistic\_bhabha

906f15e3942a my\_nginx:v1 "/usr/local/nginx/sb…" 7 minutes ago Up 7 minutes 0.0.0.0:80->80/tcp peaceful\_lumiere

打开浏览器进行验证，打开浏览器，输入弹性IP地址，端口为81，进行验证，显示“Welcome to nginx!”，说明容器运行正常。



## 删除弹性云服务器及相关资源

完成实验后请务必删除华为云上的收费资源，以免造成不必要的收费。找到创建的弹性云服务器ECS，按照如下步骤进行删除。

打开云服务器控制台，在需要删除的云服务器后面选择“更多>删除”。



在弹出对话框中勾选“释放云服务器绑定的弹性公网IP地址”和“删除云服务器挂载的数据盘”，然后点击“是”。



查看到列表中已没有资源时，表示弹性云服务器已删除。



# 实验三 平台OA系统容器化部署

## 实验介绍

### 关于本实验

本实验在购买了服务器产品的前提下，本实验介绍如何用容器构建OA系统镜像，并通过华为云容器引擎CCE部署OA系统，使得应用快速部署至华为云平台。

### 实验目的

* 理解容器移植的方法及过程
* 掌握Docker根据基础镜像构建OA系统
* 了解华为镜像服务SWR的使用
* 掌握验证构建的OA系统镜像

### 实验资源

设备名称、型号与版本的对应关系如下：

实验设备详细信息

|  |  |  |
| --- | --- | --- |
| **设备名称** | **设备型号** | **软件版本** |
| OA系统服务器  ecs-oa | X86计算，通用计算增强型，kc1.large.2 | 2vCPUs | 4GB | 公共镜像CentOS 7.9 64bit (40GB) |
| 云数据库 RDS for MySQL  rds-oa | RDS for MySQL，通用增强型，2vCPUs | 4 GB | 5.7 |

注意，若《云上应用高可用部署》实验中的弹性云服务器ecs-oa和云数据库rds-oa已删除，在本实验中则需要重新购买资源并配置OA系统和数据库连接。

## 通过Dockerfile构建OA系统镜像

### Docker安装

使用实验一所创建的ecs服务器，如果已经删除，按照实验一的步骤重新创建一台ecs服务器，远程登录ECS，检查内核版本。

uname -r



若您安装过docker，需要先删掉，之后再安装依赖:。

yum install -y docker

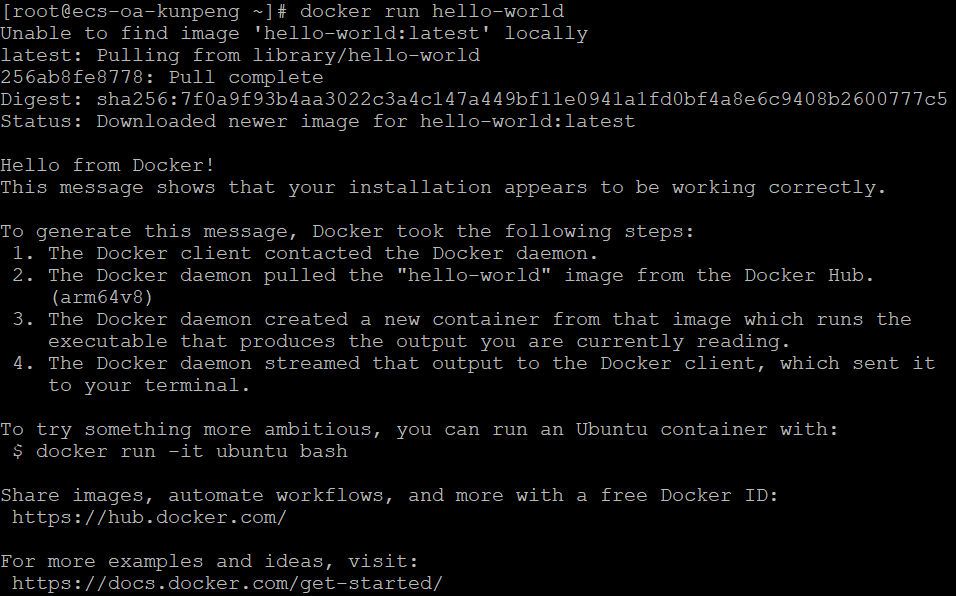
启动Docker 后台服务。

systemctl start docker

测试运行hello-world。

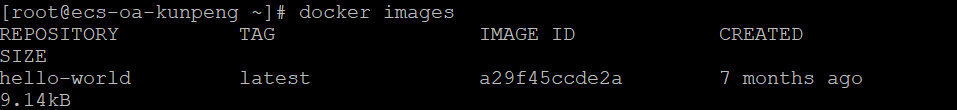
docker run hello-world

由于本地没有hello-world这个镜像，所以会下载一个hello-world的镜像，并在容器内运行。



查看下载的hello-world镜像。

docker images



到此步骤，说明docker下载并运行成功。

### Dockerfile构建OA系统镜像

按照《云上应用高可用部署》实验中的步骤2.4.2安装maven，然后根据步骤2.4.4步骤克隆代码到服务器上。

进入目录，创建Dockerfile。

cd /home/oasys\_mysql/target/  
vim Dockerfile

按"i"进入编辑模式，输入如下内容到Dockerfile中。完成后点击“Esc”键，然后输入“:wq”后按回车键，保存退出文档

FROM java:openjdk-8-alpine  
MAINTAINER kunpeng  
ADD oasys.jar oasys.jar  
EXPOSE 8088  
ENTRYPOINT ["java","-jar","oasys.jar"]

配置镜像加速器，通过以下操作修改daemon配置文件“/etc/docker/daemon.json”来使用加速器。

mkdir -p /etc/docker  
tee /etc/docker/daemon.json <<-'EOF'  
{  
"registry-mirrors": ["https://7046a839d8b94ca190169bc6f8b55644.mirror.swr.myhuaweicloud.com"]  
}  
EOF  
​  
systemctl daemon-reload  
systemctl restart docker

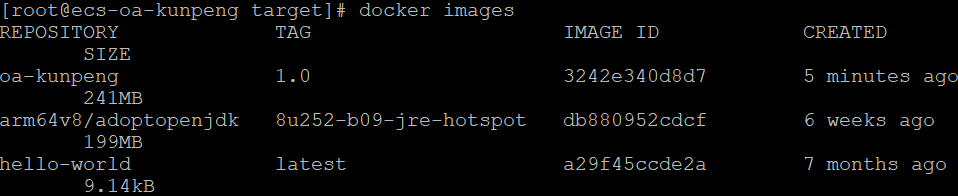
通过Dockerfile创建OA系统镜像，镜像名为“oa-kunpeng”，标签为1.0 。

docker build -t oa-kunpeng:1.0 .



查看创建的OA系统镜像。

docker images



### 将镜像上传至华为镜像服务SWR

登录华为云官网，选择“产品 -> 容器 -> 容器镜像服务 SWR”，然后点击立即使用。



点击右上方“创建组织”。



输入自定义的组织名称，点击“确定”并确认。



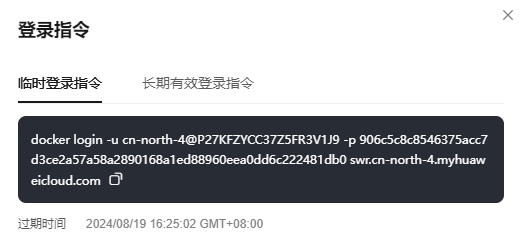
进入容器镜像服务页面，点击“我的镜像 -> 客户端上传”。

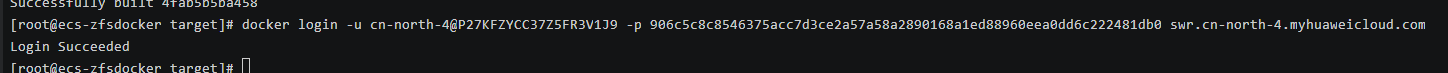


页面会提出如下提示，点击“生成临时docker login指令”。



将docker login指令复制到ECS下的docker环境中。显示login Succeeded，说明登录成功。





对要上传的镜像进行命名打标签，命令如下：

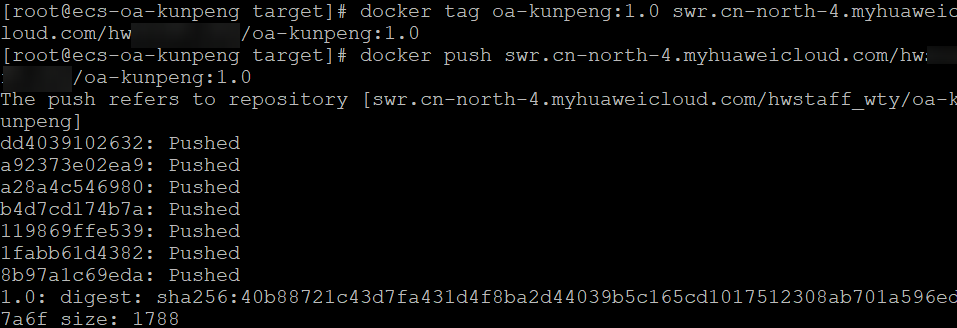
docker tag [{镜像名称}:{版本名称}] swr.cn-north-4.myhuaweicloud.com/{组织名称}/{镜像名称}:{版本名称}

其中镜像名称和版本名称为之前设置的oa-kunpeng:1.0，组织名称为第3步所创建的。

docker tag oa-kunpeng:1.0 swr.cn-north-4.myhuaweicloud.com/{组织名称}/oa-kunpeng:1.0

将打完标签的镜像上传。

docker push swr.cn-north-4.myhuaweicloud.com/{组织名称}/oa-kunpeng:1.0



返回容器镜像服务页面，点击“我的镜像”，可以看到镜像已经上传成功。



## CCE集群创建

### 创建CCE集群

分别点击“产品 -> 基础服务 -> 云容器引擎CCE”。



在弹出的CCE界面中，单击“购买”并同意授权。



请按照以下参数设置CCE集群服务选项信息。

* 集群类型：CCE Standard集群
* 计费模式:按需计费
* 集群名称：oa-cluster
* 企业项目：default
* 版本：v1.29
* 集群规模：50节点
* 集群master实例数：3
* 虚拟私有云：vpc-default
* 所在子网：subnet-default
* 网络类型：容器隧道网络
* 容器网段：自动选择
* 服务网段：使用默认网段
* 服务转发模式：iptables





单击“下一步：插件选择”，默认的就可以不需要更改

单击“下一步：插件配置”，按照默认配置即可

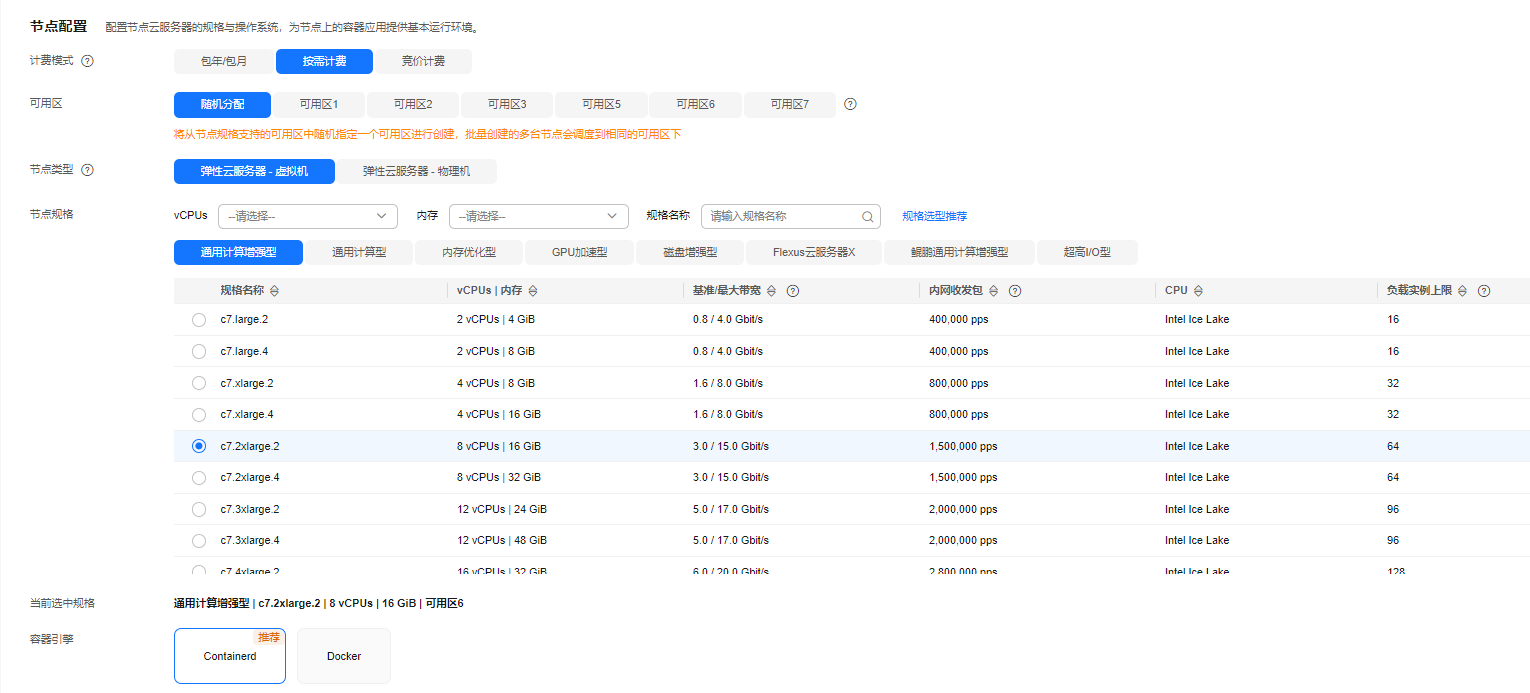
单机“下一步：确认配置”，勾选“我已阅读并知晓上述使用说明”确认购买即可

创建好集群，找到右侧“…”图标，点击“创建节点”



请按照以下参数创建节点。

* 计费模式：按需计费
* 可用区：随机分配
* 节点类型：弹性云服务器-虚拟机
* 节点规格：通用计算增强型 | c7.2xlarge.2 | 8 vCPUs | 16 GiB | 可用区5
* 容器类型：Containerd
* 操作系统：EulerOS 2.9
* 节点名称：自定义
* 企业项目：default
* 登录方式：选择密码，并自定义符合规则的密码
* 系统盘：通用SSD 50G
* 数据盘：通用SSD 100G
* 虚拟私有云/节点子网：会自动选择集群的虚拟私有云和子网
* 节点IP：自动分配
* 弹性公网IP：自动创建
* 线路：全动态BGP
* 计费模式：按流量计费
* 带宽类型：独享
* 带宽大小：5 Mbit/s







单击"下一步：规格确认"，勾选“我已阅读并知晓上述使用说明和[《云容器引擎服务声明》](http://www.huaweicloud.com/declaration/tsa_cce.html" \t "_blank)” ，点击“提交”等待节点创建成功

集群创建需要6-10分钟。

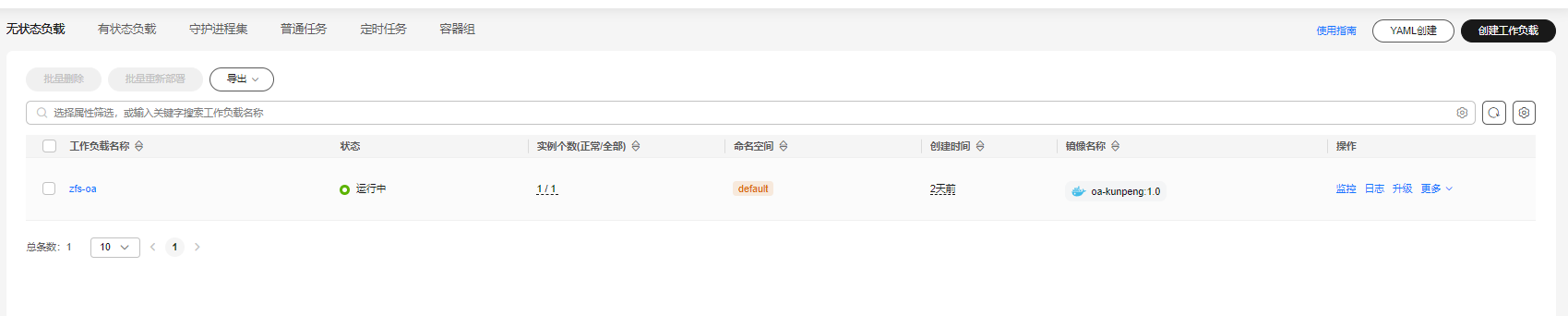
## CCE部署OA系统

### CCE部署OA系统

返回云容器引擎页面，点击新创建的集群，然后单击左侧菜单的“工作负载”进入工作负载页面



点击右上角“创建负载”按钮进入创建负载页面



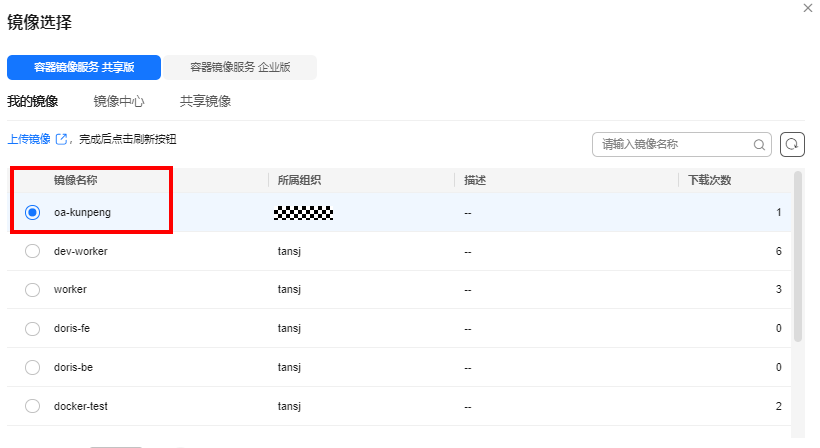
请按照以下参数配置设置工作负载基本信息，

* 工作负载名称：oa-kunpeng
* 实例数量：1

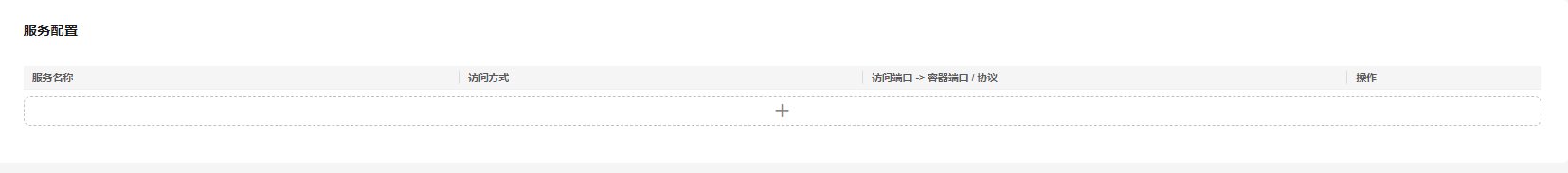


* 容器配置：点击“我的镜像”，然后选择之前上传至的“oa-kunpeng”镜像，点击“确定”。





* 服务配置：点击“+”图标创建服务，
* 访问类型：“负载均衡”，
* 服务亲和：“集群级别”，
* 负载均衡器：企业项目选择default，弹性公网ip自动创建，其余选择默认即可
* 端口配置：容器端口和服务端口都配置8088







单击“创建负载均衡”。



待“oa-kunpeng”实例执行状态为安装成功后，单击“访问方式”，复制访问地址中的弹性公网IP。

通过弹性公网IP加8088端口访问OA系统登录页，使用用户名“soli”和密码“123456”登录进入OA系统主界面，如<http://119.3.234.64:8088>。



# 释放实验资源

## 删除云数据库RDS

在云数据库RDS页面，选择rds-oa，点击“更多 -> 删除实例”。

在对话框中点击“是”，删除gauss-oa实例。

## 删除CCE集群

在CCE集群管理页面，点击“更多 -> 删除集群”。

在对话框中点击“全选”，输入“DELETE”，点击“是”。

## 删除ECS弹性云服务器

进入ECS列表，点击全选按钮，点击“更多 -> 删除”。

在对话框中选择“释放云服务器绑定的公网IP地址”和“删除云服务器挂载的数据盘”，点击“是”。

## 删除IMS镜像

在容器镜像服务“我的镜像”列表中，勾选镜像，点击“删除”。

在对话框中输入“DELETE”，点击“确定”。

## 资源检查

在控制台点击“更多 | 资源 | 我的资源”菜单项，检查资源是否全部删除。

注意：(1) 虚拟私有云VPC和安全组可以不删除，以留下次使用。(2) 若在其他区域购买了ECS和EIP等资源，请切换到那个区域查看。