# 基础软件的安装

# 服务器准备

开发时最好有一台自己的阿里云服务器，如果没有，也可以使用本地的VMware去安装一台虚拟机。在这里推荐直接使用云服务器，这样每次开发之前就不用启动一些必备的软件了。

## 云服务器配置要求

### 开发阶段

最低配置

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | CPU | 内存 | 宽带 | 数量 |
| 1 | 2核 | 4G | 2M | 1台 |

注意：

* 以上为服务器的最低配置
* 服务器必须有公网的IP地址

### 部署阶段

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | CPU | 内存 | 宽带 | 数量 |
| 1 | 2核 | 4G | 4M | 10台 |
| 2 | 4核 | 16G | 8M | 3台 |
| 3 | 2核 | 8G | 4M | 6台 |

注意：

* 开发阶段只需要购买1台服务器即可
* 部署阶段将采用Kubenetes来部署

## 云服务器的购买

在这里推送使用**阿里云**服务器！

浏览器输入：

|  |
| --- |
| https://www.aliyun.com/ |

登录自己的账号：



选择你的登录方式：



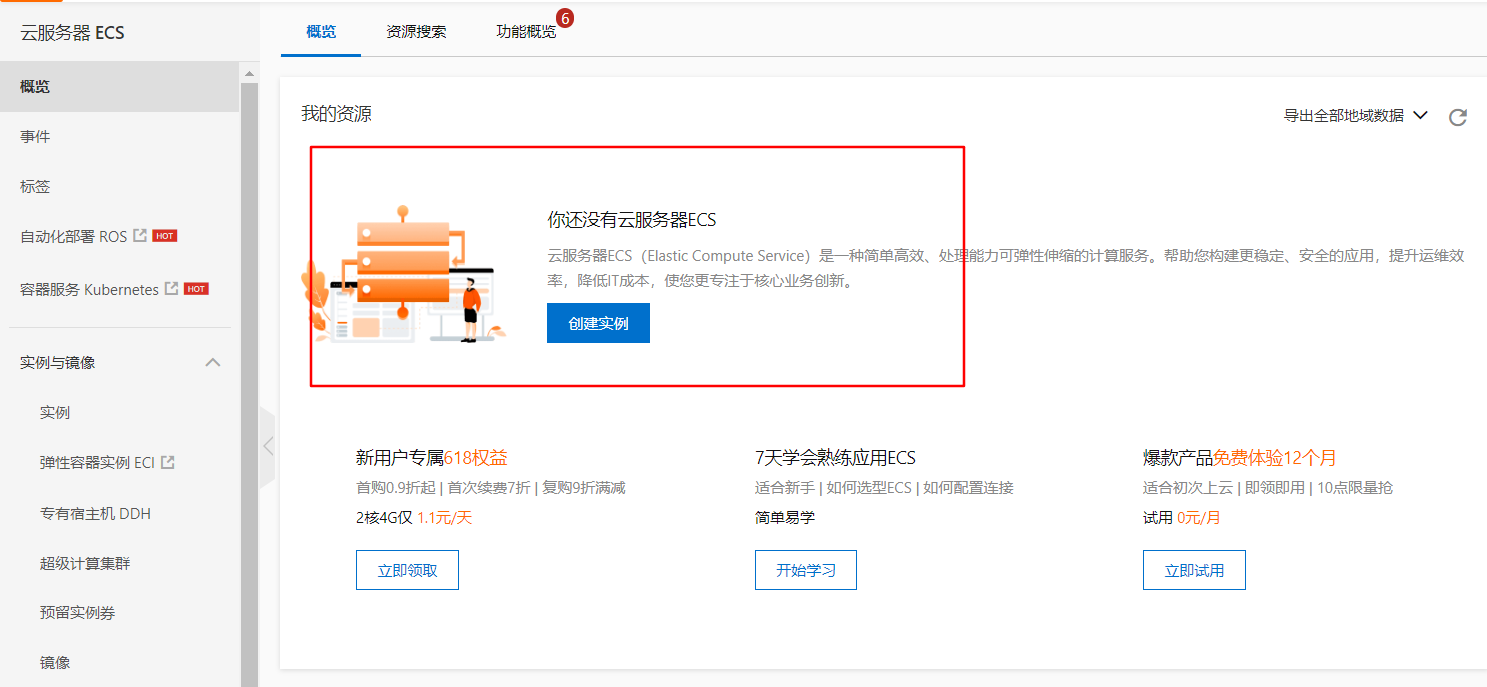
打开你的控制台：



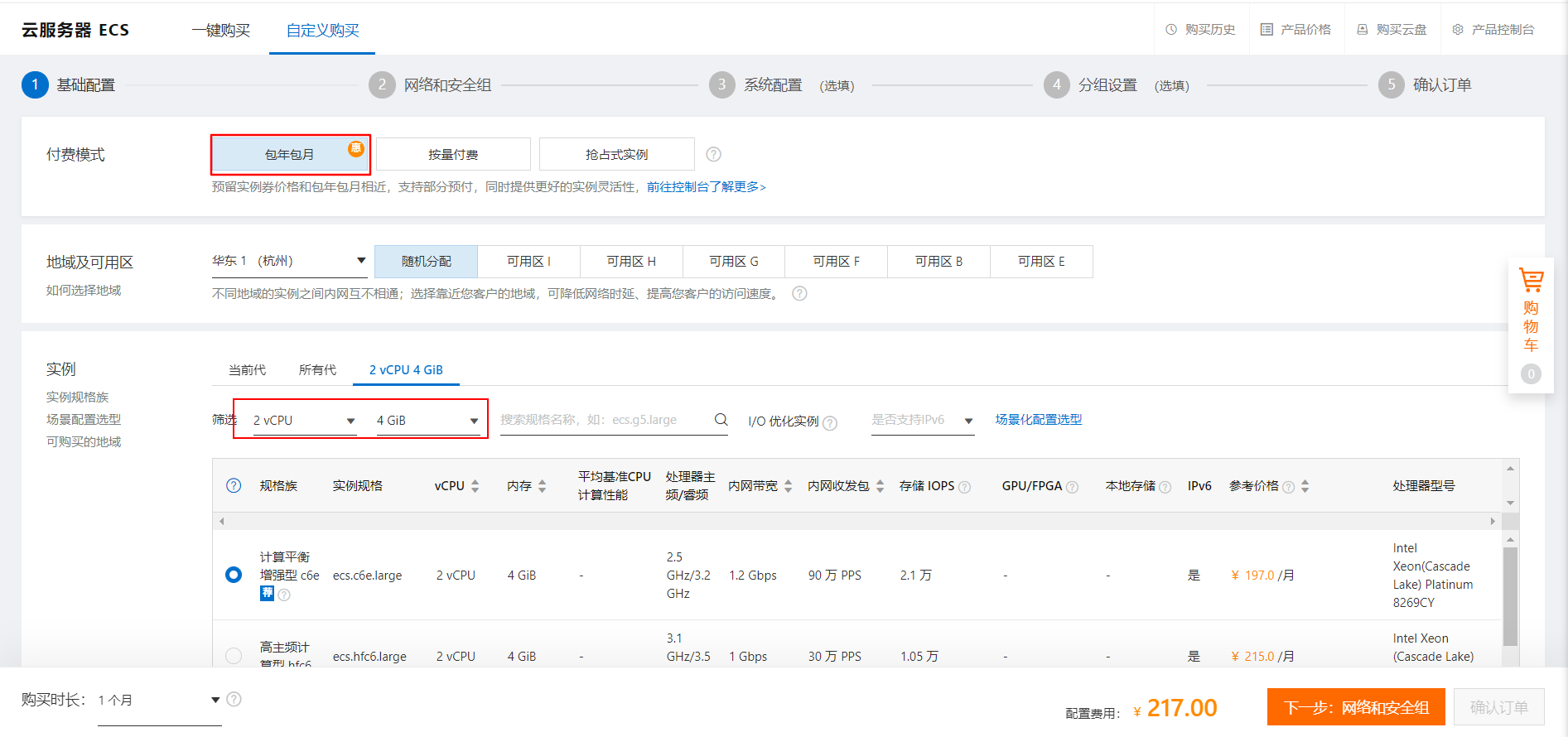
找到云服务器：



点击创建实例：



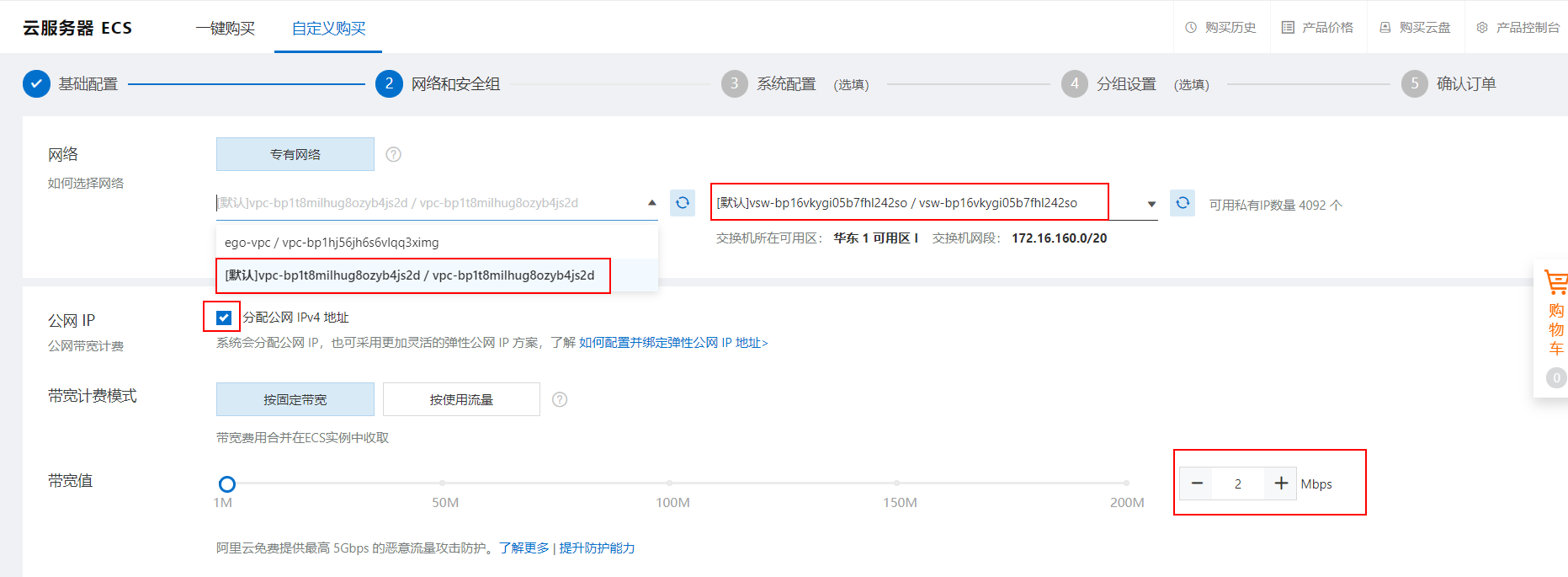
选择你的规格：



选择你的操作系统：



配置网络和安全组：



其他保持不变，点击下一步，进入系统配置：

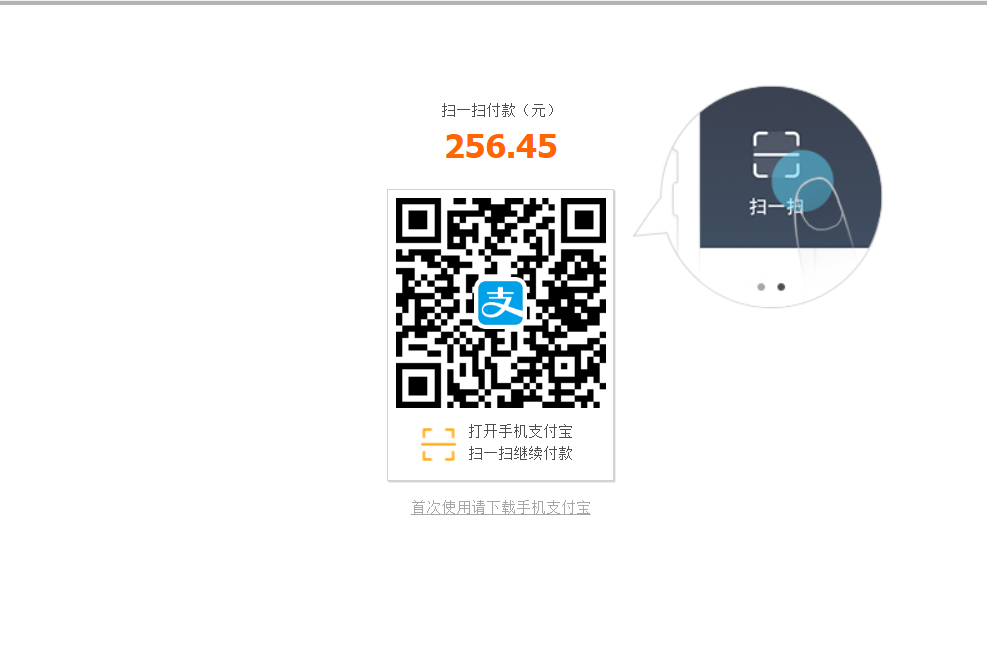


密码建议使用数字、字母、受支持的符号组成。

完成后点击确认下单即可。



支付订单即可：



查询我的实例列表：



## 云服务钱的安全组设置

找到实例列表：



配置规则：

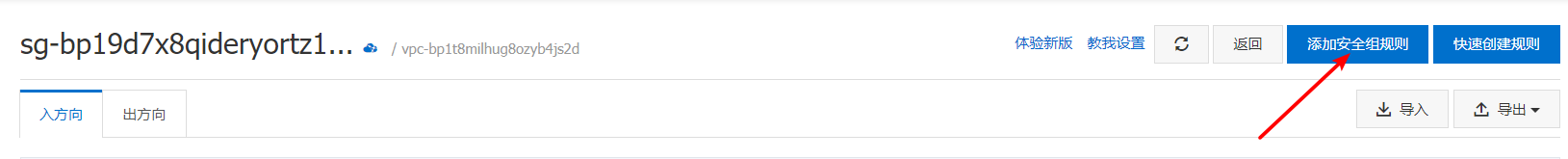


## 使用SSH工具连接云服务

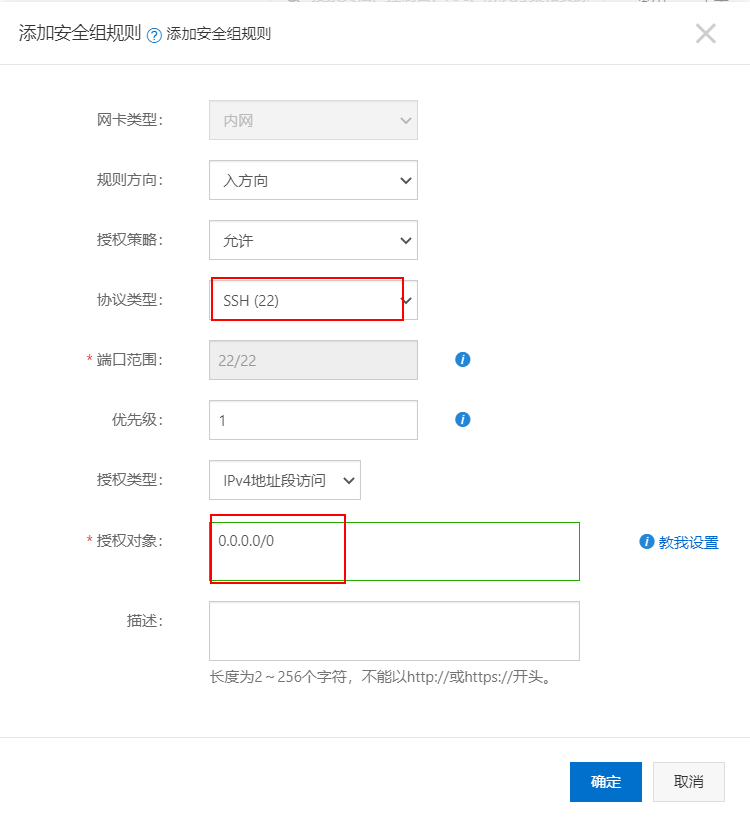
发现它只放行了22 端口（远程SSH端口）



如果你的没有放行，请看下面的操作：

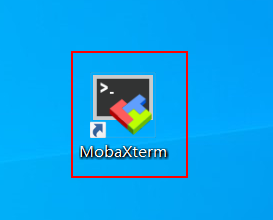


新增一个：



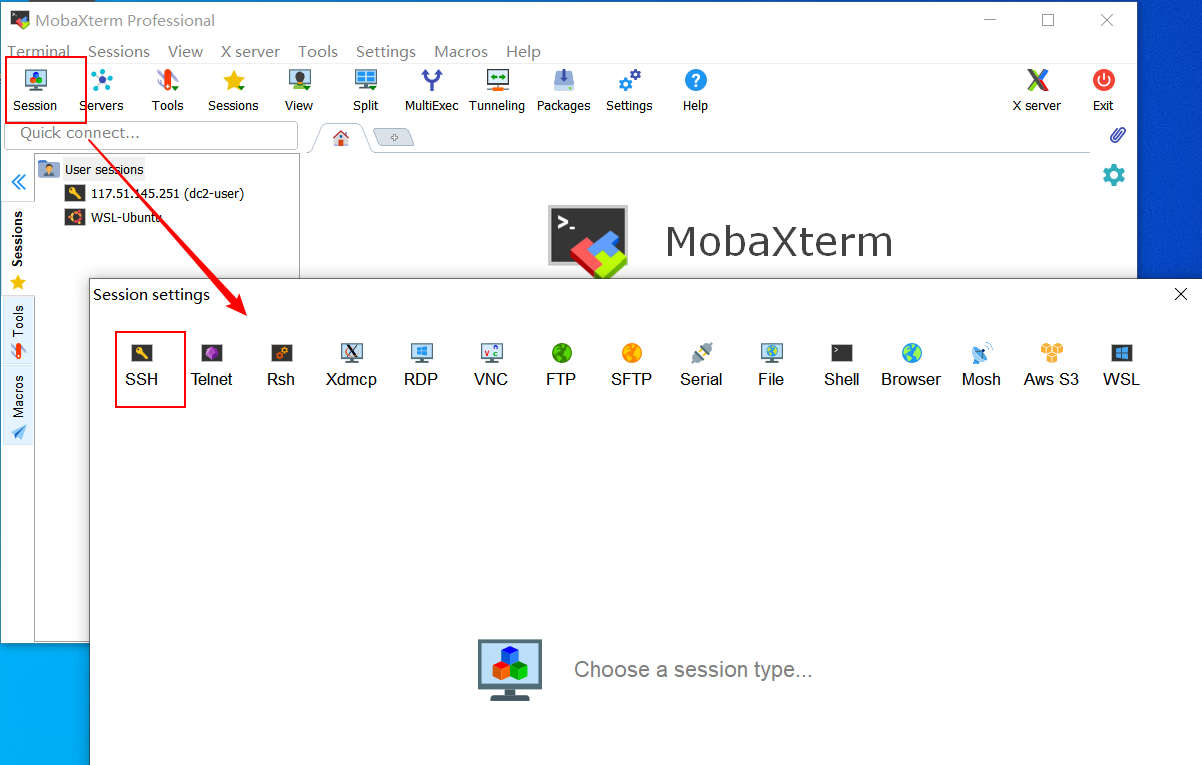
点击确定就完成了。

打开你的Shell工具：

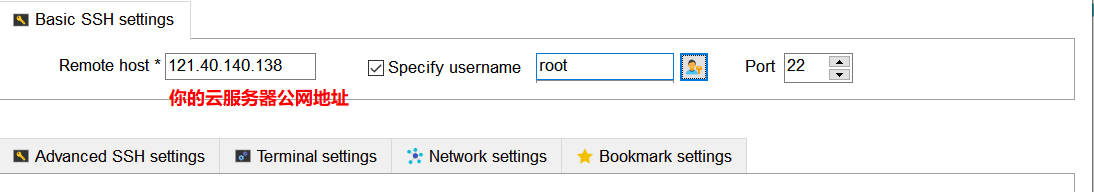


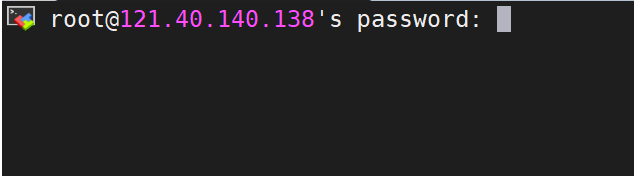
我使用的时MobaXterm ，你也可以使用你喜欢的xshell，finalshell等等...

现在我们开始连接：

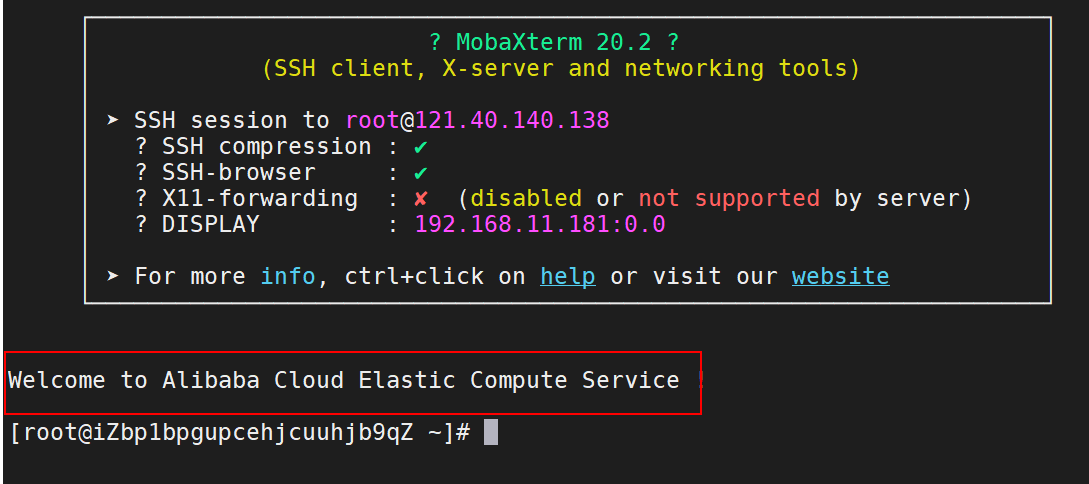


填写地址：

  
点击OK即可连接：



输入你的密码即可。



# Docker安装

Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的镜像中，然后发布到任何流行的 Linux或Windows 机器上，也可以实现虚拟化。容器是完全使用沙箱机制，相互之间不会有任何接口。

我们在本次开发中将使用Docker部署我们所有基础软件！

## 2.1 安装Docker

请执行以下命令：

* 安装Docker：

|  |
| --- |
| yum -y install docker |

* 开机自启：

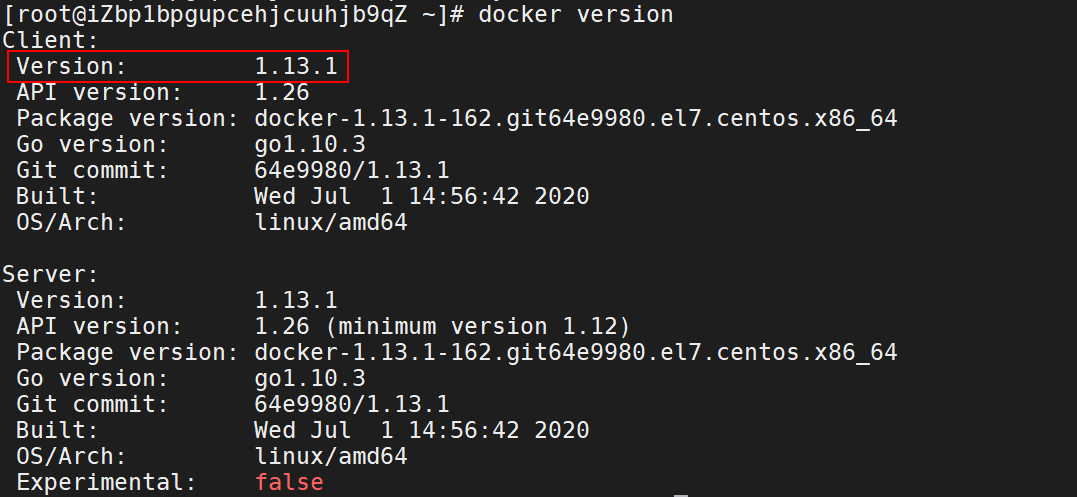
|  |
| --- |
| systemctl enable docker |

* 启动Docker：

|  |
| --- |
| systemctl start docker |

* 查看Docker当前的版本

|  |
| --- |
| docker version |



至此，Docker已经安装完成。

## 2.2 Docker加速仓库配置

Docker的镜像仓库默认在国外的服务器上，导致我们拉取镜像的速度很慢，因此，我们需要配置它的加速地址来加快镜像的拉取速度。

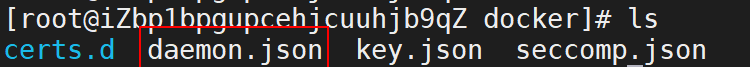
执行以下命令：

* 切换到Docker的配置目录里面：

|  |
| --- |
| cd /etc/docker/ |

* 查询当前目录所有的文件：

|  |
| --- |
| ls |



daemon.json就是我们要修改的配置文件。

* 修改daemon.json：

|  |
| --- |
| vi daemon.json |

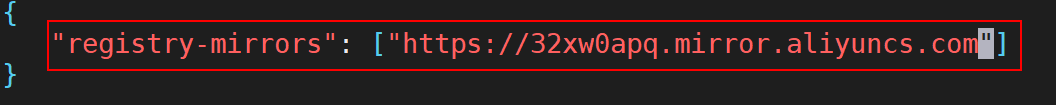
按a 键进入编辑模式：



* 填写以下内容：

|  |
| --- |
| "registry-mirrors": ["https://32xw0apq.mirror.aliyuncs.com"] |

完毕后如下：



按wq 保存退出。

* 配置文件修改后，重启Docker：

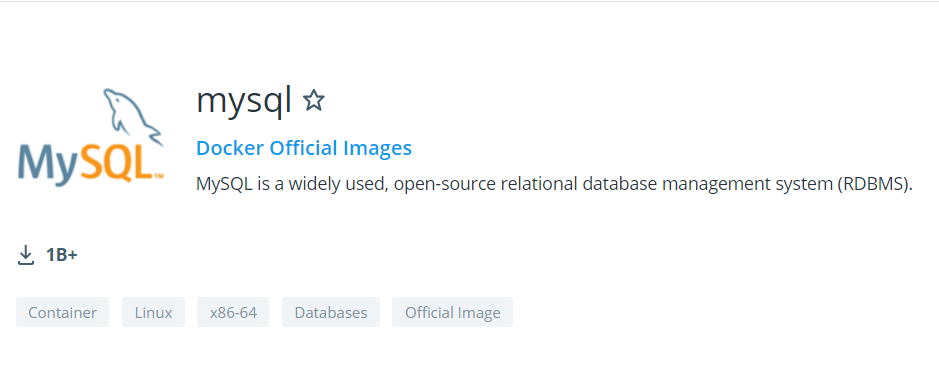
|  |
| --- |
| systemctl restart docker |

至此，Docker已经安装配置完毕了。

# Mysql安装

## 3.1 Mysql镜像地址

|  |
| --- |
| https://hub.docker.com/\_/mysql |



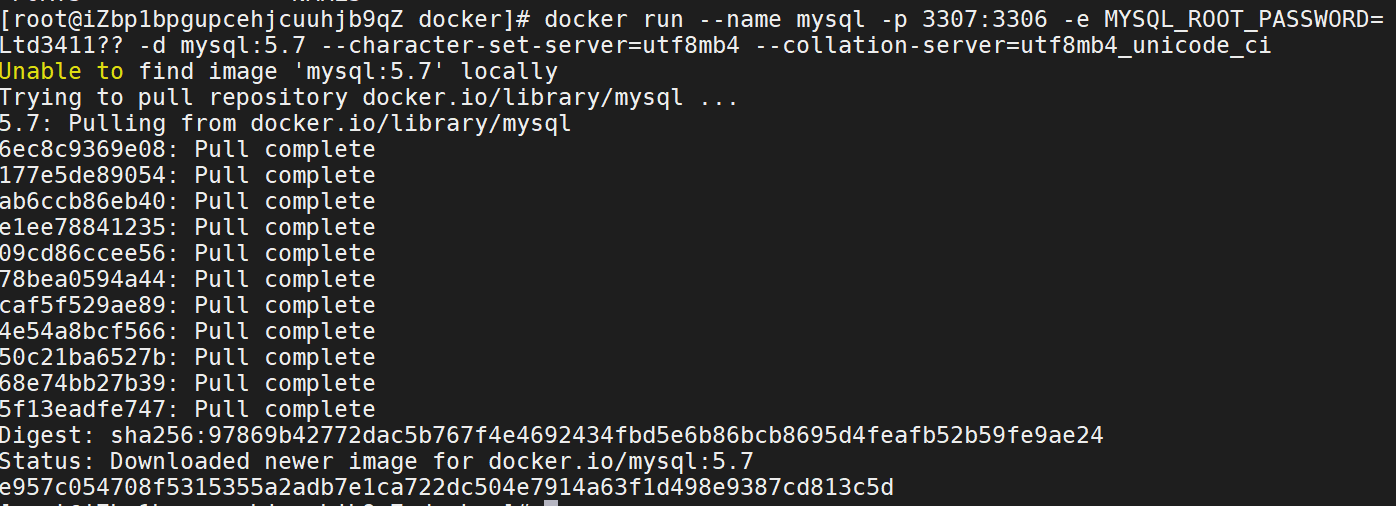
## 3.2 安装Mysql

执行如下命令：

|  |
| --- |
| docker run --name mysql -p 3307:3306 -e MYSQL\_ROOT\_PASSWORD=Ltd3411?? -d mysql:5.7 --character-set-server=utf8mb4 --collation-server=utf8mb4\_unicode\_ci |

说明：

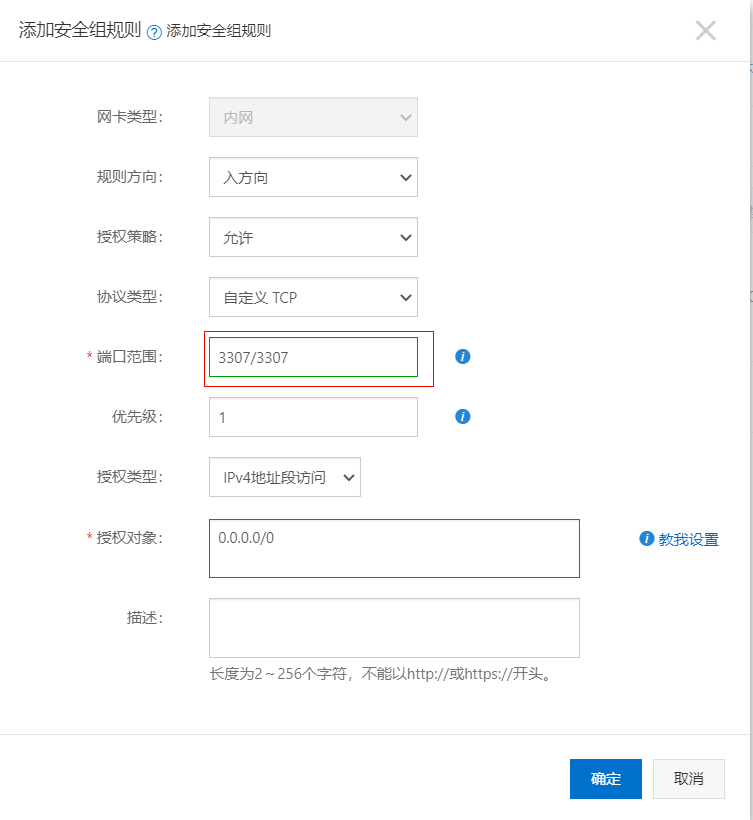
* 3307 我们以后将使用3307 该端口来访问Mysql，主要是为了防止黑客扫描攻击
* MYSQL\_ROOT\_PASSWORD是指定Mysql的root密码，这里不用使用123456简单的数字
* character-set-server、collation-server设置默认的数据库编码为UTF-8



## 3.3 安全组配置

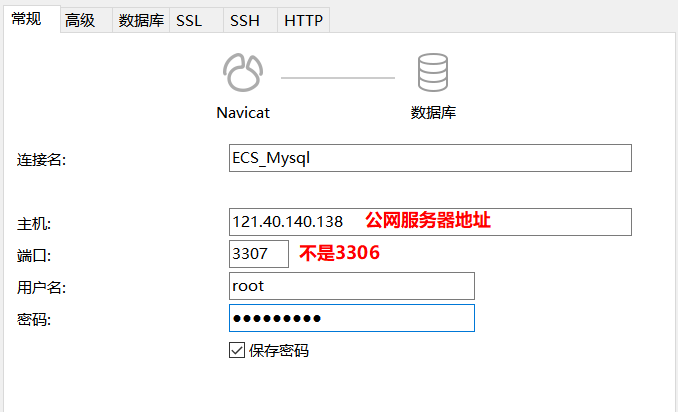
我们上面使用-p 3307:3306 将Mysql 服务器监听端口绑定在了3307上面，因此需要放行3307端口，供外界访问到3307该端口。

打开安全组配置：

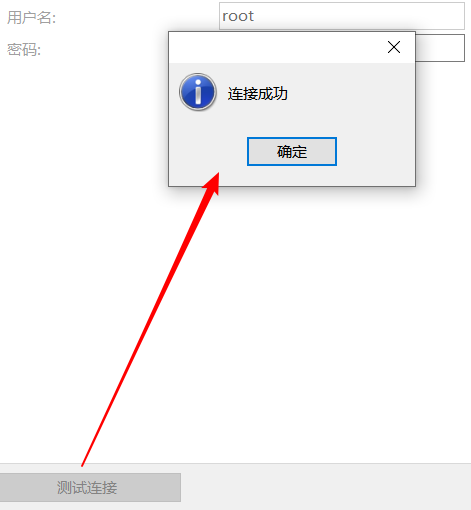




## 3.4 连接Mysql



点击测试连接：

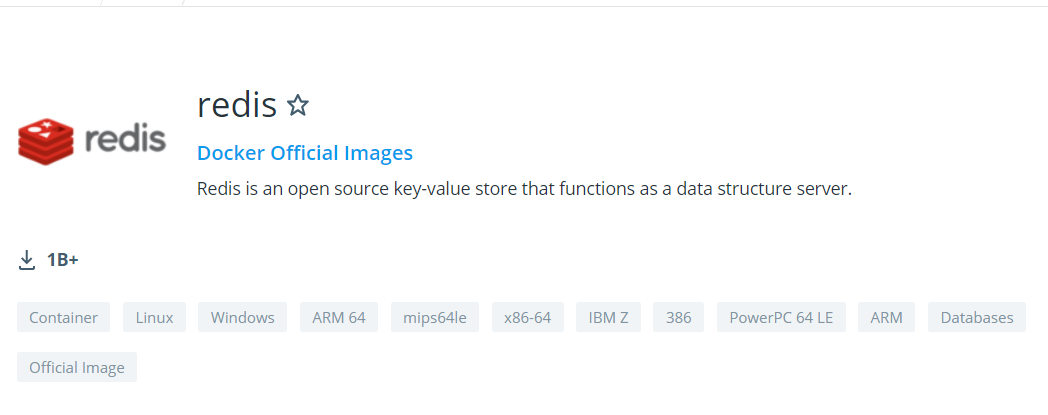


代表安装已经完成了。

# Redis安装

## 4.1 Redis镜像地址

|  |
| --- |
| https://hub.docker.com/\_/redis |



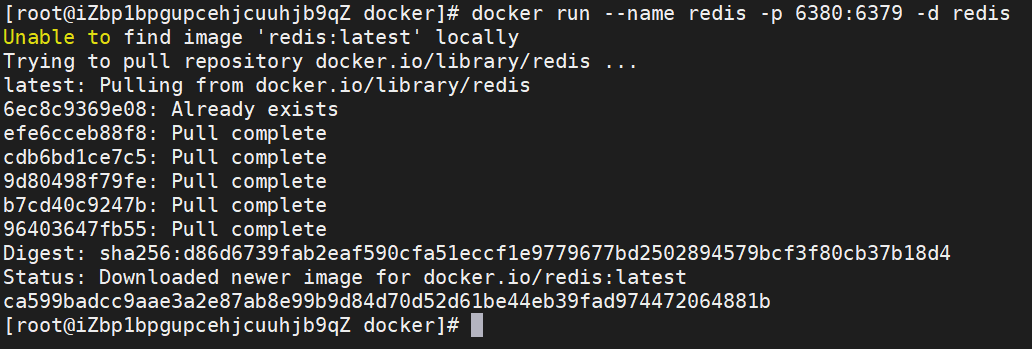
## 4.2 安装Redis

执行如下命令：

|  |
| --- |
| docker run --name redis -p 6380:6379 -d redis |

说明：

* -p 6380 是将redis服务器监听的端口绑定在本地6380的端口上



## 4.3 Redis密码配置

Redis必须配置数据库密码。

执行以下的命令：

* 进入redis容器里面

|  |
| --- |
| docker exec -it redis bash |

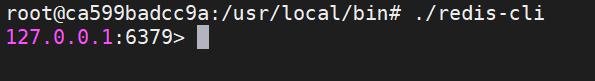
* 切换目录

|  |
| --- |
| cd /usr/local/bin |

* 配置密码

|  |
| --- |
| ./redis-cli |

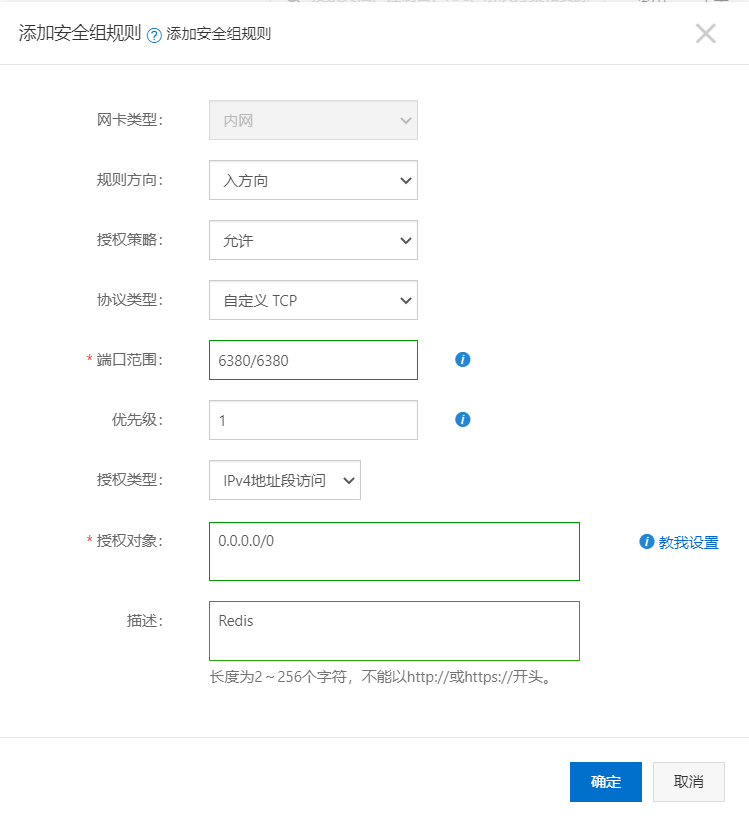
|  |
| --- |
| CONFIG SET requirepass Ltd3411?? |



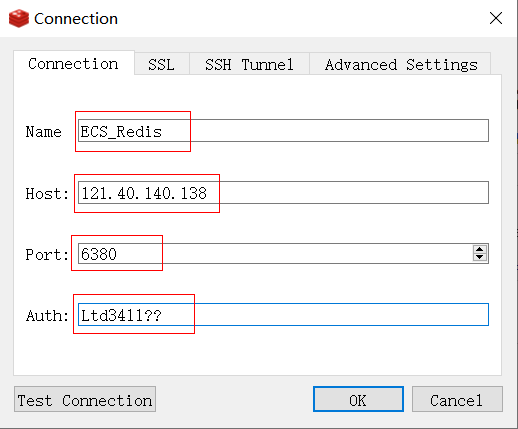


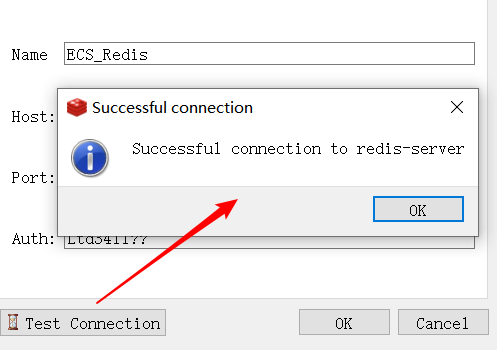
## 4.4 安全组配置

放行6380 端口：



## 4.5连接Redis





连接成功即可。

# RocketMQ安装

## 5.1镜像地址

rocketmq-namesrv地址

|  |
| --- |
| https://hub.docker.com/r/rocketmqinc/rocketmq-namesrv |

rocketmq-broker地址

|  |
| --- |
| https://hub.docker.com/r/rocketmqinc/rocketmq-broker |

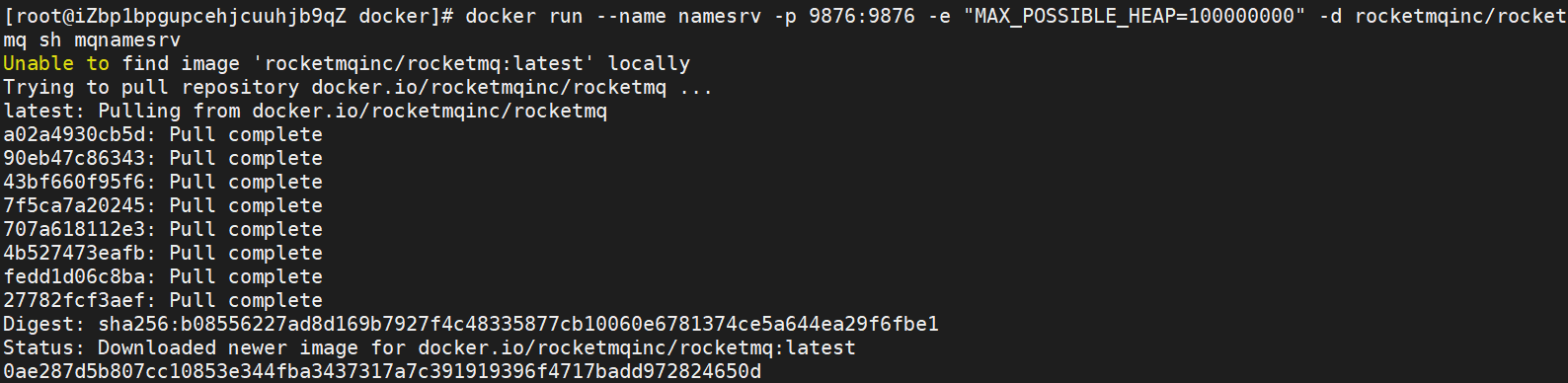
rocketmq-console-ng地址

|  |
| --- |
| https://hub.docker.com/r/styletang/rocketmq-console-ng |

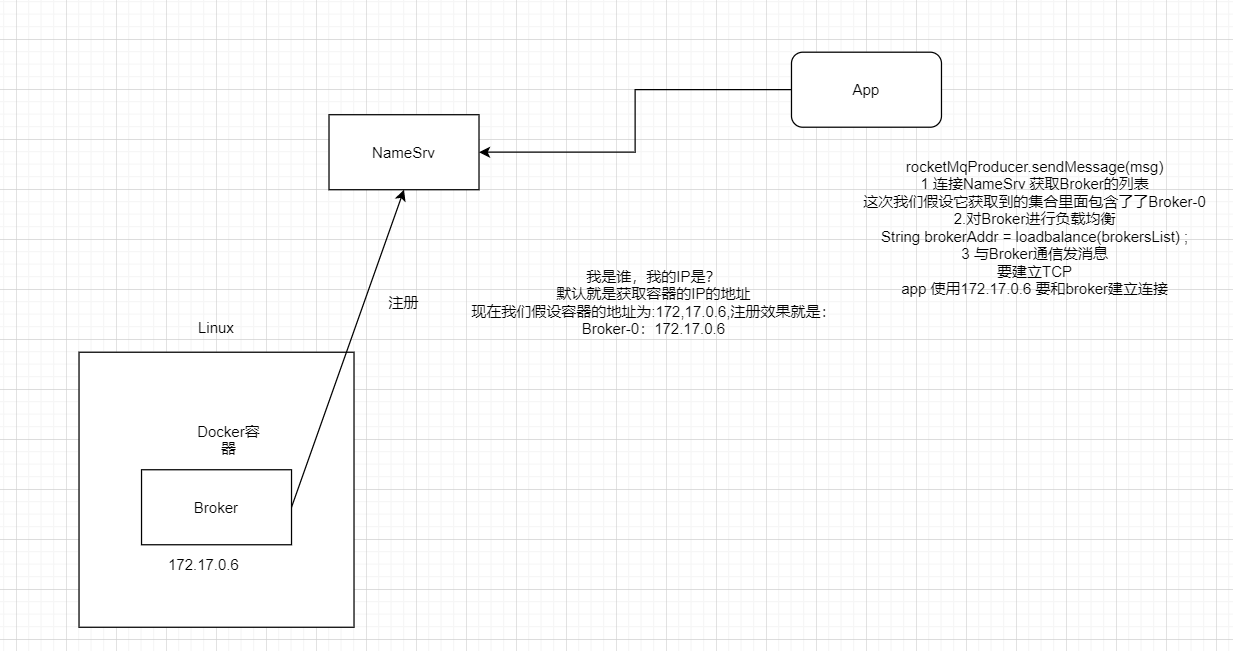
## **5.2 Namesrv的安装**

执行以下命令：

|  |
| --- |
| docker run --name namesrv -p 9876:9876 -e "MAX\_POSSIBLE\_HEAP=100000000" -d rocketmqinc/rocketmq sh mqnamesrv |



## **5.3 broker的安装**



### 5.3.1 创建一个配置文件

执行以下命令：

* 创建文件夹：

|  |
| --- |
| mkdir -p /usr/local/rocketmq |

* 切换目录：

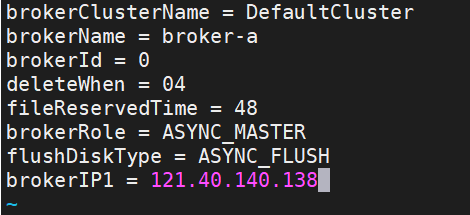
|  |
| --- |
| cd /usr/local/rocketmq |

* 新建broker.conf 文件

|  |
| --- |
| vim broker.conf |

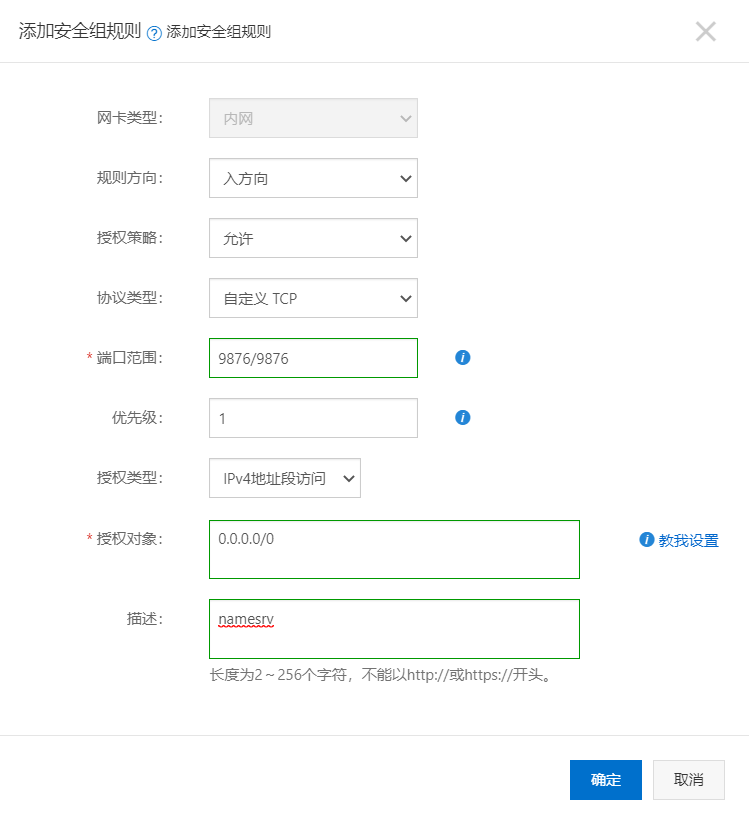
* 添加以下的配置：

|  |
| --- |
| brokerClusterName = DefaultCluster  brokerName = broker-a  brokerId = 0  deleteWhen = 04  fileReservedTime = 48  brokerRole = ASYNC\_MASTER  flushDiskType = ASYNC\_FLUSH  brokerIP1 = 121.40.140.138 |



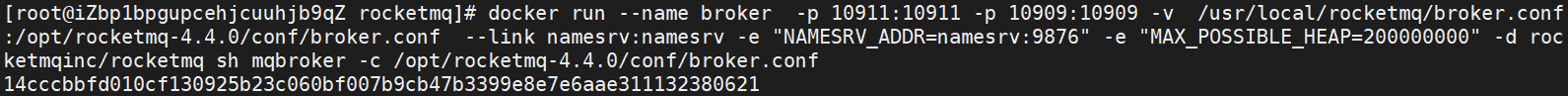
### 5.3.2 放行9876端口

因为broker要注册到namesrv上面，因为我们需要让broker通过外网访问到9876该端口。



### 5.3.3 安装broker

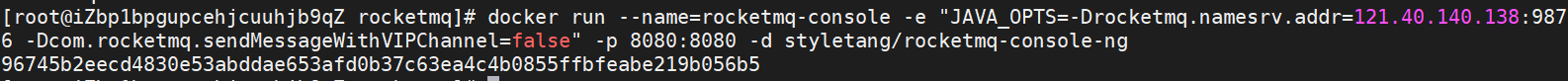
|  |
| --- |
| docker run --name broker -p 10911:10911 -p 10909:10909 -v /usr/local/rocketmq/broker.conf:/opt/rocketmq-4.4.0/conf/broker.conf --link namesrv:namesrv -e "NAMESRV\_ADDR=namesrv:9876" -e "MAX\_POSSIBLE\_HEAP=200000000" -d rocketmqinc/rocketmq:4.4.0 sh mqbroker -c /opt/rocketmq-4.4.0/conf/broker.conf |



## **5.4 console的安装**

执行以下命令：

|  |
| --- |
| docker run --name=rocketmq-console -e "JAVA\_OPTS=-Drocketmq.namesrv.addr=121.40.140.138:9876 -Dcom.rocketmq.sendMessageWithVIPChannel=false" -p 8080:8080 -d styletang/rocketmq-console-ng |



## 5.5 放行端口

放行8080，10911，10909

浏览器访问：

|  |
| --- |
| http://121.40.140.138:8080/#/ |

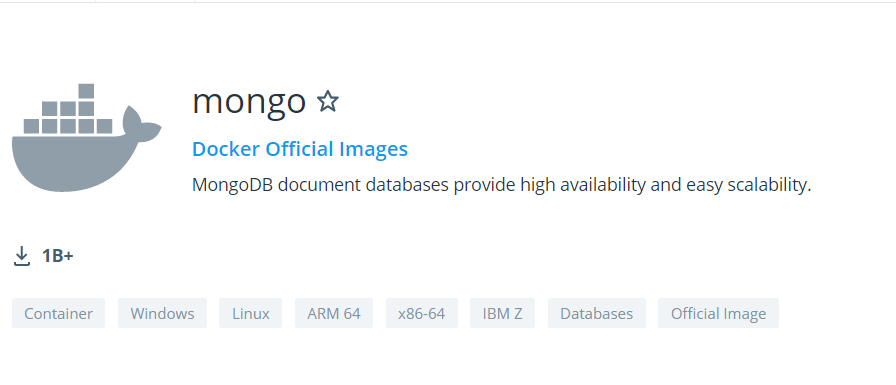


没有任何问题

# MongoDB安装

## 6.1 镜像地址

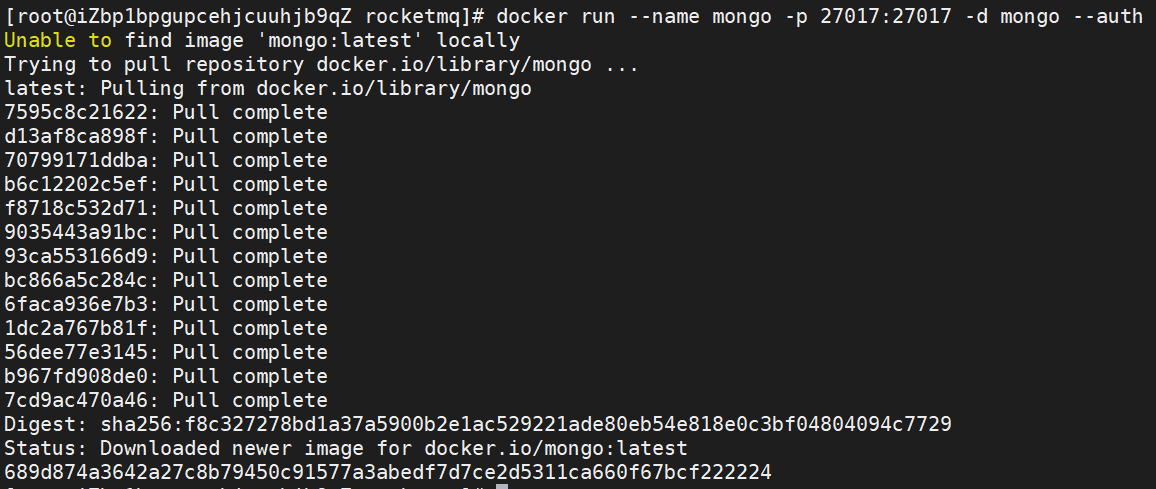
|  |
| --- |
| https://hub.docker.com/\_/mongo |



## 6.3 安装MongoDB

执行以下命令：

|  |
| --- |
| docker run --name mongo -p 27017:27017 -d mongo --auth |



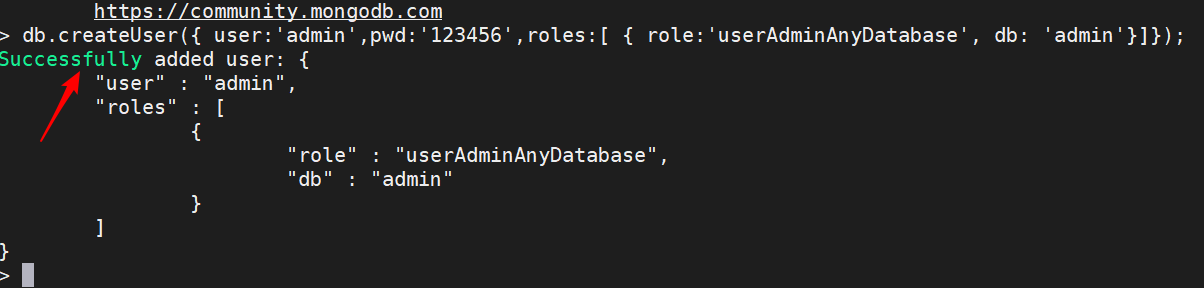
## 6.4 MongDB密码配置

* 执行以下命令：

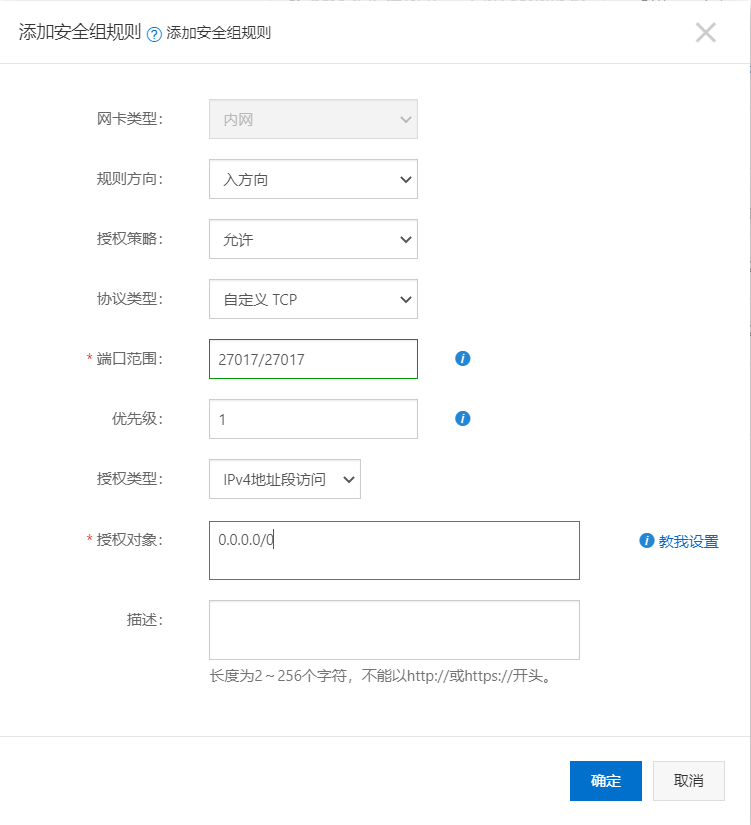
|  |
| --- |
| docker -exec -it mongo mongo admin |

* 设置密码：

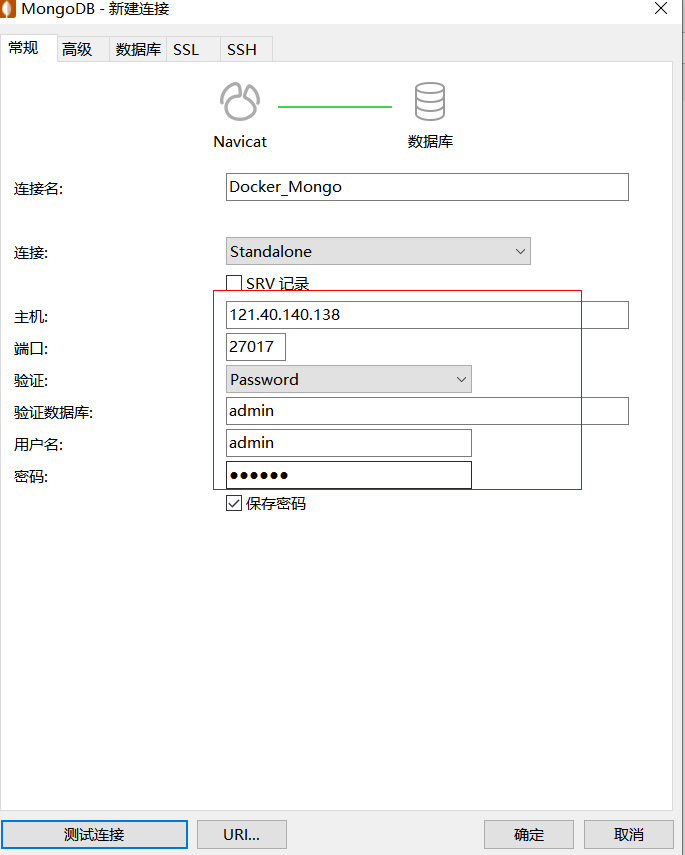
|  |
| --- |
| db.createUser({ user:'admin',pwd:'123456',roles:[ { role:'userAdminAnyDatabase', db: 'admin'}]}); |



## 6.5 安全组配置



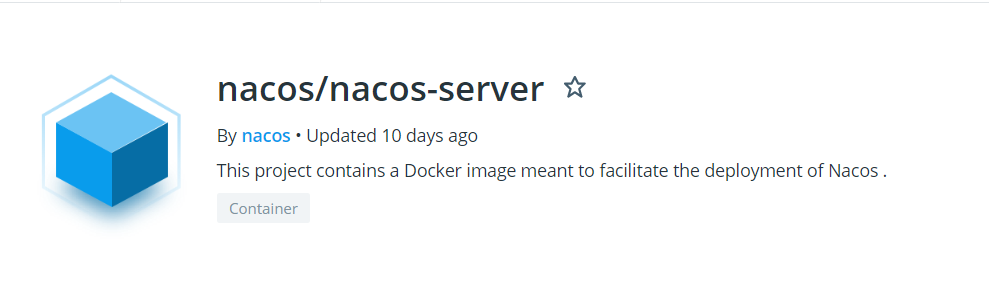
## 6.6 连接MongoDB



# Nacos-Server安装

## 7.1 镜像地址

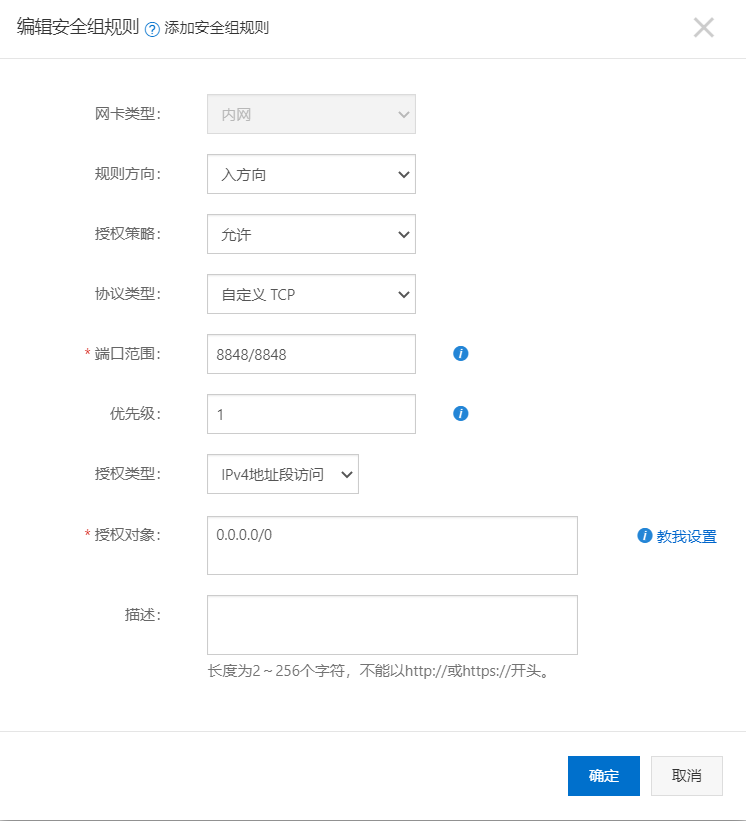
|  |
| --- |
| https://hub.docker.com/r/nacos/nacos-server |



## 7.2 安装Nacos-Server

|  |
| --- |
| docker run --name nacos -e MODE=standalone -p 8848:8848 -d nacos/nacos-server:1.1.4 |

## 7.3 安全组配置

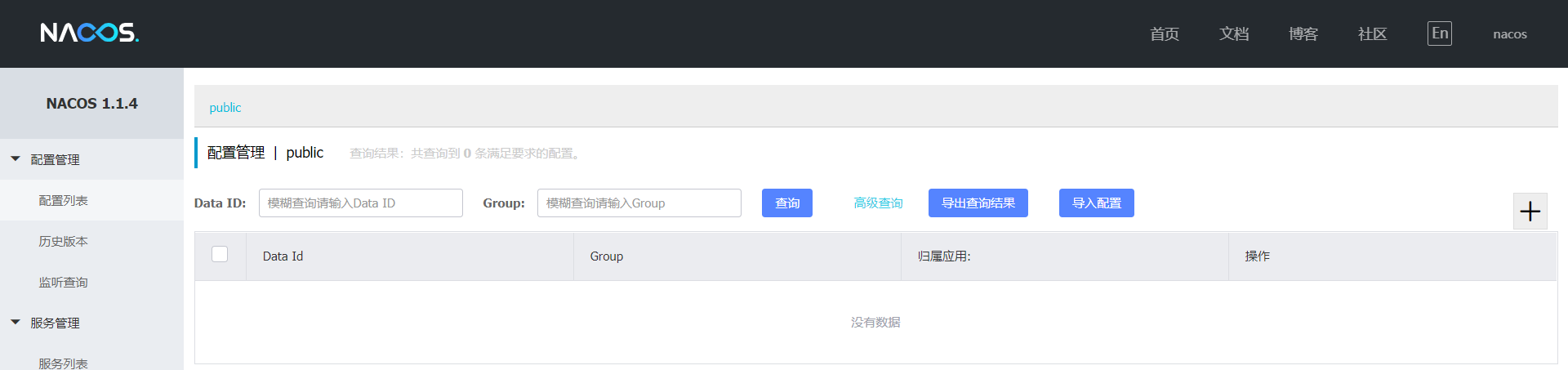


## 7.4 访问Nacos-Server

|  |
| --- |
| http://121.40.140.138:8848/nacos/ |

注意：

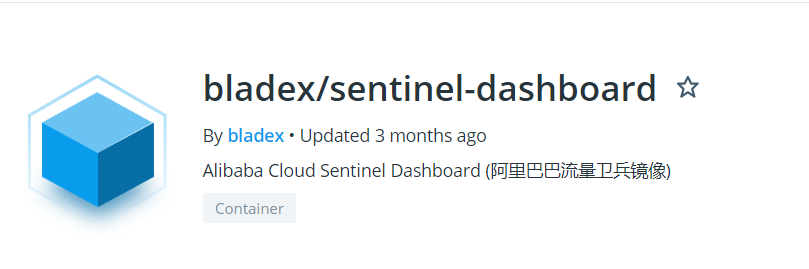
* 用户名：nacos
* 密码：nacos



# Sentinel-Dashboard安装

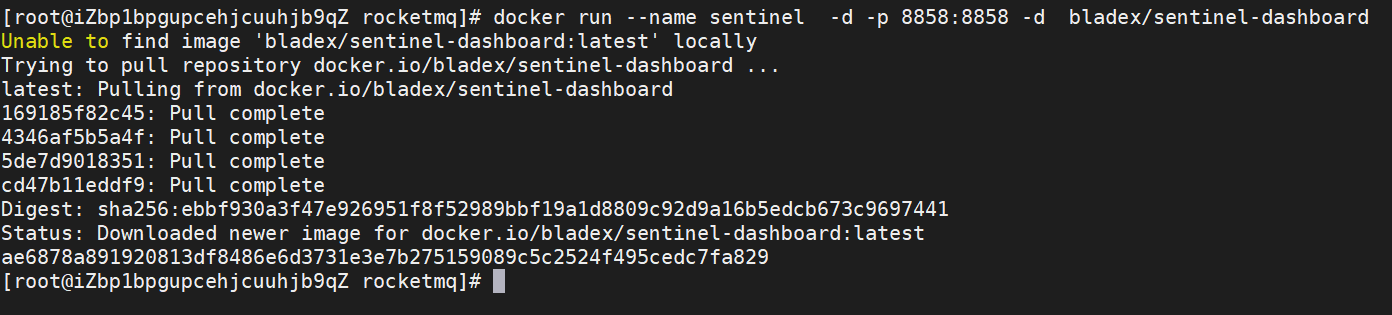
## 8.1 镜像地址

|  |
| --- |
| https://hub.docker.com/r/bladex/sentinel-dashboard |



## 8.2 安装Sentinel-Dashboard

|  |
| --- |
| docker run --name sentinel -p 8858:8858 -d bladex/sentinel-dashboard |



## 8.3 安全组配置



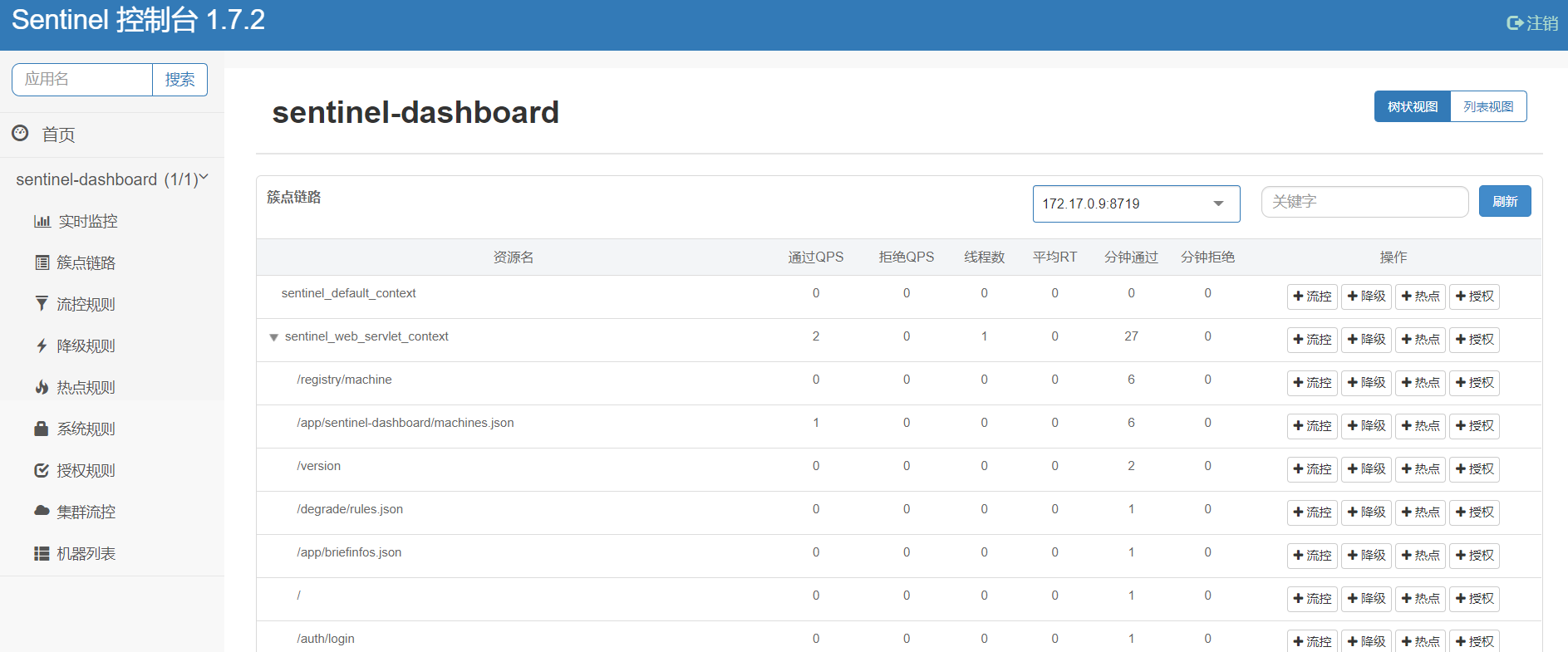
## 8.4 访问测试

|  |
| --- |
| http://121.40.140.138:8858/ |



用户名：sentinel

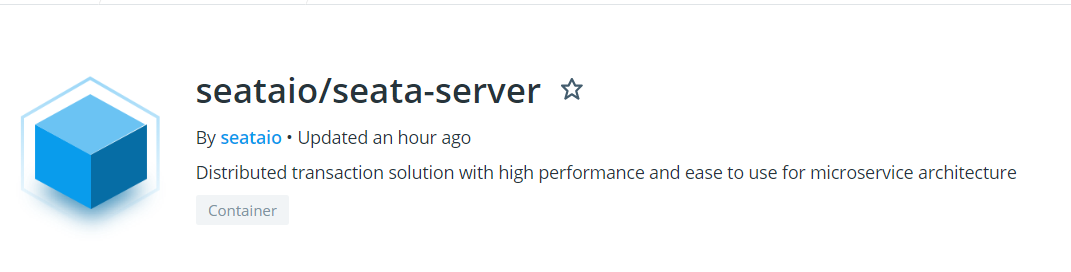
密码：sentinel



# Seata-Server安装

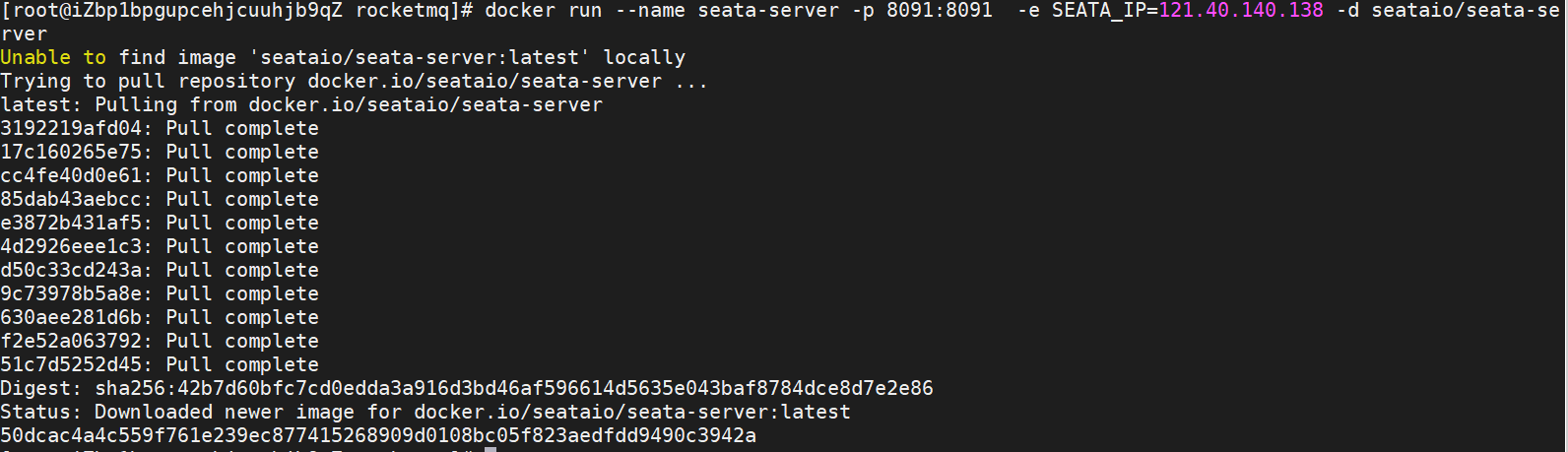
## 9.1 镜像地址

|  |
| --- |
| https://hub.docker.com/r/seataio/seata-server |

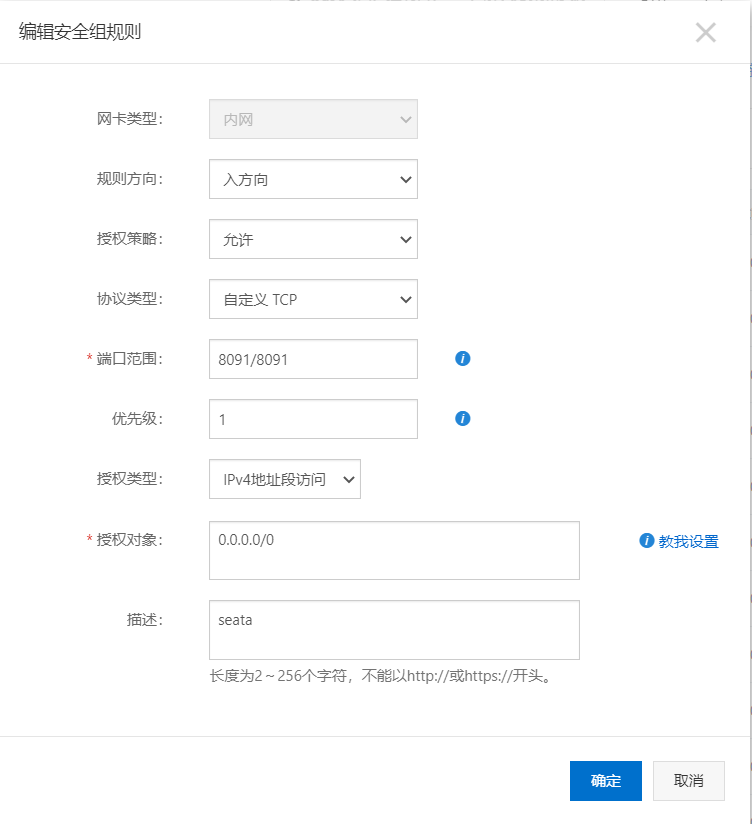


## 9.2 安装Seata-Server

|  |
| --- |
| docker run --name seata-server -p 8091:8091 -e SEATA\_IP=121.40.140.138 -d seataio/seata-server |



## 9.3 安全组配置

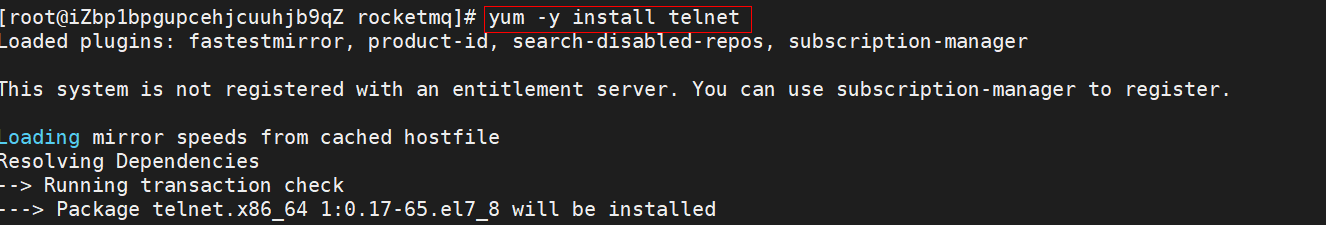


## 9.4 测试该端口

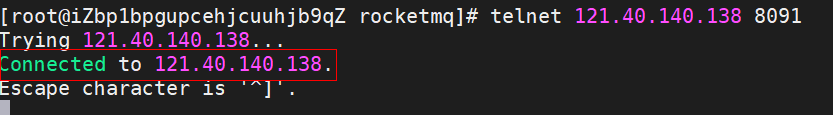
执行以下命令：

* 安装telnet：

|  |
| --- |
| yum -y install telnet |



* 尝试与8091 端口通信：



通信是成功了。