

高性能计算

用户指南

用户指南

购买须知

ECS设置

安全组是一个逻辑上的分组，这个分组是由同一个地域内具有相同安全保护需求并相互信任的实例组成。

安全组，类似防火墙，用于设置单台或多台云服务器的网络访问控制，它是重要的安全隔离手段。

每个实例至少属于一个安全组，在创建的时候就需要指定。

不同安全组的实例之间默认内网不通，可以授权两个安全组之间互访。[更多详情](#)

ECS安全组配置

目前，ECS安全组有三种方式

- 配置方式1：阿里云管理控制台配置 目前，非VPC用户无法在阿里云控制台看到安全组配置，可提交工单申请开通，注明“开通控制台显示‘本实例安全组’”。

在开通安全组之后，用户可以在ECS实例控制台看到安全组，如图:



点击右上角 “创建安全组”：

刷新

创建安全组

操作

修改 | 管理实例 | 配置规则

每页显示：10条

«

<

1

>

»

填写安全组名称和备注，选择网络类型:

创建安全组

* 安全组名称：

gpu_test

长度为2-128个字符，以大小写字母或中文开头，可包含数字，".", "_" 或 "-"。

描述：

gpu_test_note

长度为2-256个字符，不能以http://或https://开头。

网络类型：

经典网络

确定

取消

安全组创建成功后，需要配置规则，点击安全组的“配置规则”链接:



添加安全组配置规则时，规则方向需要选“入方向”，网卡类型选“内网”：

添加安全组规则 ×

授权策略：

允许

规则方向：

入方向

协议类型：

TCP

* 端口范围：

1/65535

取值范围为1~65535；例如“1/200”、“80/80”。

优先级：

1

优先级可选范围为1-100，默认值为1，即最高优先级。

网卡类型：

内网

授权类型：

地址段访问

授权对象：

0.0.0.0/0

确定

取消

规则创建成功后，可以看到此时的安全组还没有应用到任何ECS实例，如图显示，相关实例是0:

☐ 安全组ID/名称	所属专有网络	相关实例	网络类型
☐ sg-280j7ro29 gpu_test		0	经典网络

下面，还需要将安全组绑定到ecs实例，才能使ecs按此安全组规则生效。

选择ecs实例的“安全组配置”选项

付费方式(全部) ▾		操作	
18	包年包月 15-11-25 00:00到期	管理 续费	变配 更多 ▾

每页显示：20条

启动

停止

重启

重置密码

修改信息

连接管理终端...

连接帮助

重新初始化磁盘

更换系统盘

购买相同配置

编辑标签

安全组配置

修改私网IP

点击右上角“加入安全组”：



在弹出对话框中，选择刚刚创建的安全组:



此时返回安全组列表，可以看到相关实例的值已经变为1:

☐	安全组ID/名称	所属专有网络	相关实例	网络类型
☐	sg-280j7ro29 gpu_test		1	经典网络

此时测试从配置的网段访问此ecs的特定端口，即可访问。

- 配置方式2：命令行jar包配置 首先下载jar包到一个有java环境的linux上。执行如下命令，进入交互模式：

```
$ java -jar ~/aliyuncs.jar -id <您的Access Key ID> -secret <您的Access KeySecret> -service ECS:2014-05-26
```

其中，aliyuncs.jar为上面下载的文件解压得到，2014-05-26为ECS api版本，此处可以指定为2014-05-26

然后创建安全组实例，如下命令，RegionId为ecs所在城市，Description为安全组名称

```
@ECS:/> CreateSecurityGroup RegionId=cn-qingdao,Description=gpu_test
```

创建成功后，会返回安全组实例名，当前操作是sg-28fcdqtel，接着可以给安全组赋予更多的属性。下例是实现10.239.23.101/28网段的所有机器可以访问当前ecs的80端口。

```
@ECS:/> AuthorizeSecurityGroup SecurityGroupId=sg-28fcdqtel,IpProtocol=TCP,PortRange=80/80,SourceCidrIp=10.239.23.101/28,NicType=intranet,RegionId=cn-qingdao
```

完善安全组实例属性后，就可以将安全组实例和ecs实例进行绑定。

```
@ECS:/> JoinSecurityGroup SecurityGroupId=sg-28fcdqtel,InstanceId=i-28weu9ksr
```

以上步骤操作后，即可实现10.239.23.101/28网段的所有主机访问当前ecs的80端口。

除以上操作外，还有以下一些相关操作。

列出所有安全组

```
@ECS:/> DescribeSecurityGroups RegionId=cn-qingdao
```

列出当前安全组的所有属性

```
@ECS:/> DescribeSecurityGroupAttribute SecurityGroupId=sg-28fcdqtel,NicType=intranet,RegionId=cn-qingdao
```

解除当前安全组和当前ecs实例的绑定

```
@ECS:/> LeaveSecurityGroup SecurityGroupId=sg-28fcdqtel,InstanceId=i-28weu9ksr
```

删除安全组

```
@ECS:/> DeleteSecurityGroup SecurityGroupId=sg-28fcdqtel,RegionId=cn-qingdao
```

授权从当前ecs向外访问的安全组规则

```
@ECS:/> AuthorizeSecurityGroupEgress SecurityGroupId=sg-28fcdqtel,IpProtocol=TCP,PortRange=1/65535,DestCidrIp=0.0.0.0/0,NicType=intranet,RegionId=cn-qingdao
```

更多信息可以参考Jar包实践

- 配置方式3：API方式配置 详见ECS API 使用文档，使用方式较复杂，不推荐使用。

注：

当前的RegionId，可参考：

可选区域	RegionId
------	----------

华东1	cn-hangzhou
华东2	cn-shanghai
华北1	cn-qingdao
华北2	cn-beijing
香港	cn-hongkong
华南1	cn-shenzhen
美国硅谷	us-west-1

阿里云HPC物理机本身不能访问外网，只能通过ECS正向代理访问。

本文档将指导用户如何设置代理服务器。

1. 确定IP地址

用户应首先确认这几个IP地址：

ECS外网IP（不便于透露，本文用XXX.XXX.XXX.XXX表示）和内网IP（实验用10.10.10.10）

HPC物理机内网IP（实验用10.239.23.4）

2. 登录ECS跳板机

用户可以用PUTTY工具（Windows环境）或SSH命令（Linux环境）登录ECS，注意应使用ECS外网IP登入。

```
ssh -l login_name XXX.XXX.XXX.XXX (ECS外网IP)
```

登录成功后，可以在ECS跳板机上用SSH命令登录HPC物理机：

```
ssh -l root 10.239.23.4 (HPC物理机内网IP)
```

3. ECS跳板机上部署代理服务器squid

这里选择squid，因为它不仅支持访问HTTP的服务还支持访问HTTPS的服务。

3.1 安装squid

重新开一个终端，登录到ECS跳板机。

直接用yum安装：

```
yum install squid
```

默认情况下安装位置在 /usr/sbin/squid

3.2 编辑squid配置文件

用root权限打开 /etc/squid/squid.conf 文件，首先把不需要的内网地址全部注释上，在上面增加一行，添加自己的HPC物理机IP地址，然后把不需要开放的端口注释上，只留下80和443端口，然后增加一行：
: access_log /var/log/squid/access.log用来记录访问情况。

修改后如下：

```
.....

acl localnet src 10.239.23.4    # 这里增加一行，添加自己的HPC物理机IP地址
#acl localnet src 10.0.0.0/8    # RFC1918 possible internal network，注释上
#acl localnet src 172.16.0.0/12 # RFC1918 possible internal network，注释上
#acl localnet src 192.168.0.0/16 # RFC1918 possible internal network，注释上
#acl localnet src fc00::/7      # RFC 4193 local private network range，注释上
#acl localnet src fe80::/10     # RFC 4291 link-local (directly plugged) machines，注释上

acl SSL_ports port 443
acl Safe_ports port 80          # http服务端口打开
#acl Safe_ports port 21         # ftp可以根据情况是否打开
acl Safe_ports port 443        # https服务端口打开
#acl Safe_ports port 70         # gopher，注释上
#acl Safe_ports port 210        # wais，注释上
#acl Safe_ports port 1025-65535 # unregistered ports，注释上
#acl Safe_ports port 280        # http-mgmt，注释上
#acl Safe_ports port 488        # gss-http，注释上
#acl Safe_ports port 591        # filemaker，注释上
#acl Safe_ports port 777        # multiling http，注释上
acl CONNECT method CONNECT

.....

# And finally deny all other access to this proxy
http_access deny all           # 除上面允许的外，其他一律禁止访问

# Squid normally listens to port 3128
http_port 3128                 # squid默认监听端口号，可以修改成别的端口号

# Added
access_log /var/log/squid/access.log    # 这里增加一行，用于监控访问记录
```

保存该文件。

3.3 启动squid

用root权限运行：sudo service squid start

查看squid运行状态：

```
sudo service squid status
squid (pid 31659) is running...
```

3.4 设置ECS防火墙

出于节省流量和安全考虑，需要在ECS上设置防火墙规则，将除了HPC物理机之外的所有访问3128端口的请求都挡在防火墙外。

步骤如下：

3.4.1 开启防火墙

```
CentOS 6: service iptables start
CentOS 7: systemctl start firewalld
```

3.4.2 添加防火墙规则

首先允许HPC物理机IP地址（本文用10.239.23.4，请根据实际情况修改）访问3128端口：

```
iptables -I INPUT -s 10.239.23.4 -p TCP --dport 3128 -j ACCEPT
```

端口3128要和3.2节squid配置文件中的端口设置相同。

然后禁止所有访问3128端口的tcp连接：

```
iptables -A INPUT -p TCP --dport 3128 -j DROP
```

然后保存iptables设置：

```
service iptables save
```

查看规则是否生效：

```
iptables -L -n
```

可以看到新增了两条规则：

target	prot	opt	source	destination	
ACCEPT	tcp	--	10.239.23.4	0.0.0.0/0	tcp dpt:3128
DROP	tcp	--	0.0.0.0/0	0.0.0.0/0	tcp dpt:3128

4. 在HPC物理机上设置代理

回到HPC物理机终端，进行代理设置。

最简单的方式是使用环境变量，假设ECS内网IP为10.10.10.10（用户需要自行替换为真实ECS内网IP），则可以执行：

```
export http_proxy=http://10.10.10.10:3128
```

```
export https_proxy=http://10.10.10.10:3128
```

注意: 这里的代理服务器端口设置应该和ECS跳板机上squid.conf中监听端口一致，另外必须是ECS内网地址。

也可以将上述语句放入/etc/profile或 ~/.bashrc实现登录HPC物理机时自动配置代理服务器。

5. 测试

在HPC物理机上使用wget、git clone和yum install测试结果如下：

5.1 测试http的访问

```
wget http://www.cmake.org/files/v3.3/cmake-3.3.1.tar.gz
--2015-09-29 18:12:52-- http://www.cmake.org/files/v3.3/cmake-3.3.1.tar.gz
Connecting to 120.26.218.226:3128... connected.
Proxy request sent, awaiting response... 200 OK
Length: 6577869 (6.3M) [application/x-gzip]
Saving to: 'cmake-3.3.1.tar.gz'

100%[=====
=====>] 6,577,869
28.7KB/s in 3m 55s

2015-09-29 18:16:47 (27.4 KB/s) - 'cmake-3.3.1.tar.gz' saved [6577869/6577869]
```

可以查看/var/log/squid/access.log文件，找到上述下载的log:

```
1443530801.311 7144 10.239.23.4 TCP_MISS/200 132972 GET http://www.cmake.org/files/v3.3/cmake-3.3.1.tar.gz -
DIRECT/66.194.253.19 application/x-gzip
```

5.2 测试https的访问

```
wget https://codeload.github.com/gflags/gflags.tar.gz/v2.1.2
--2015-09-29 18:18:13-- https://codeload.github.com/gflags/gflags.tar.gz/v2.1.2
Connecting to 120.26.218.226:3128... connected.
Proxy request sent, awaiting response... 200 OK
Length: 95716 (93K) [application/x-gzip]
Saving to: 'v2.1.2'
```

```
100%[=====
=====>] 95,716
78.7KB/s  in 1.2s

2015-09-29 18:18:17 (78.7 KB/s) - 'v2.1.2' saved [95716/95716]
```

可以查看/var/log/squid/access.log文件，找到上述下载的log:

```
1443531073.042 11238 10.239.23.4 TCP_MISS/200 101394 CONNECT codeload.github.com:443 -
DIRECT/192.30.252.145 -
```

5.3 测试git clone

```
git clone https://github.com/gflags/gflags.git
Cloning into 'gflags'...
remote: Counting objects: 1772, done.
remote: Total 1772 (delta 0), reused 0 (delta 0), pack-reused 1772
Receiving objects: 100% (1772/1772), 1.27 MiB | 493.00 KiB/s, done.
Resolving deltas: 100% (1013/1013), done.
```

可以查看/var/log/squid/access.log文件，找到上述下载的log:

```
1443531119.620 29512 10.239.23.4 TCP_MISS/200 1356342 CONNECT github.com:443 - DIRECT/192.30.252.129 -
```

5.4 测试yum

```
# yum install openssl
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirrors.aliyuncs.com
* extras: mirrors.aliyuncs.com
* updates: mirrors.aliyuncs.com
Resolving Dependencies
--> Running transaction check
---> Package openssl.x86_64 1:1.0.1e-42.el7.9 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
=====
=====
Package                Arch              Version           Repository
Size
=====
=====
=====
```

```
Installing:
  openssl                x86_64                1:1.0.1e-42.el7.9                updates
711 k

Transaction Summary
=====
=====
=====
Install 1 Package

Total download size: 711 k
Installed size: 1.5 M
Is this ok [y/d/N]: y
Downloading packages:
openssl-1.0.1e-42.el7.9.x86_64.rpm | 711
kB 00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
Warning: RPMDB altered outside of yum.
** Found 1 pre-existing rpmdb problem(s), 'yum check' output follows:
authconfig-6.2.8-9.el7.x86_64 has missing requires of /usr/bin/openssl
Installing : 1:openssl-1.0.1e-42.el7.9.x86_64
1/1
Verifying : 1:openssl-1.0.1e-42.el7.9.x86_64
1/1

Installed:
  openssl.x86_64 1:1.0.1e-42.el7.9

Complete!
```

可以查看/var/log/squid/access.log文件，找到上述yum安装的log：

```
1443522032.861 129 10.239.23.4 TCP_MISS/200 728129 GET
http://mirrors.aliyuncs.com/centos/7/updates/x86_64/Packages/openssl-1.0.1e-42.el7.9.x86_64.rpm -
DIRECT/10.143.34.200 application/x-redhat-package-manager
```

通过以上测试，验证了HPC物理机已经可以通过ECS正向代理访问外网。

阿里云GPU物理机本身不能访问外网，只能通过ECS正向代理访问。本文档将指导用户如何设置代理服务器。

1. 确定IP地址

用户应首先确认这几个IP地址：ECS外网IP（不便于透露，本文用XXX.XXX.XXX.XXX表示）和内网IP（实验用10.10.10.10）；GPU物理机内网IP（实验用10.239.23.4）；

2. 登录ECS跳板机

用户可以用PUTTY工具（Windows环境）或SSH命令（Linux环境）登录ECS，注意应使用ECS外网IP登入。

```
ssh -l login_name XXX.XXX.XXX.XXX (ECS外网IP)
```

登录成功后，可以在ECS跳板机上用SSH命令登录GPU物理机：

```
ssh -l root 10.239.23.4 (GPU物理机内网IP)
```

3. ECS跳板机上部署代理服务器

这里选择Tengine，它是在NGINX的基础上由淘宝网发起的开源Web服务器项目。用户应注意，NGINX做正向代理服务器是不支持HTTPS连接的，所以客户端只能访问HTTP服务。如果用户需要在物理机上访问HTTPS服务可以选择其他Web服务器做代理。

3.1 安装Tengine

重新开一个终端，登录到ECS跳板机。获取Tengine源码：

```
wget http://tengine.taobao.org/download/tengine-2.1.1.tar.gz
```

解压：

```
tar zxvf tengine-2.1.1.tar.gz
cd tengine-2.1.1/
```

配置和编译：

```
./configure
make
sudo make install
```

默认情况下安装位置在 /usr/local/nginx/

3.2 编辑Tengine配置文件

用root权限打开 /usr/local/nginx/conf/nginx.conf 文件，在http{}语句块内增加如下内容：（“//”后面为注释，真正的conf文件中应删除）

```
server {
    resolver 8.8.8.8;
    // 设置DNS的IP，可以根据实际情况修改
    resolver_timeout 5s;
    // DNS连接超时设置
```

```
listen 0.0.0.0:8080;
// 用于连接客户端的监听端口，也可改为其他端口
access_log /root/logs/proxy.access.log;
// 连接日志，用于记录所有连接建立的情况
error_log /root/logs/proxy.error.log;
// 错误日志，用于记录所有错误情况

location / {
    allow 10.239.23.4;
// 允许接入的物理机内网IP，根据需要设置
    deny all;
// 拒绝为除了上一句指定物理机内网IP之外的所有主机服务
    // 以下不需要用户修改，保持默认即可
    proxy_pass $scheme://$host$request_uri;
    proxy_set_header Host $http_host;

    proxy_buffers 256 4k;
    proxy_max_temp_file_size 0;

    proxy_connect_timeout 30;

    proxy_cache_valid 200 302 10m;
    proxy_cache_valid 301 1h;
    proxy_cache_valid any 1m;
}
}
```

保存该文件。

3.3 启动Tengine

用root权限运行：sudo /usr/local/nginx/sbin/nginx 如果报错，请根据报错信息对3.2节中的nginx.conf配置文件做必要的修改。

3.4 设置ECS防火墙

出于节省流量和安全考虑，需要在ECS上设置防火墙规则，将除了GPU物理机之外的所有访问8080端口的请求都挡在防火墙外。步骤如下：

3.4.1 开启防火墙

```
CentOS6: service iptables start
CentOS7: systemctl start firewalld
```

3.4.2 添加防火墙规则

首先允许GPU物理机IP地址（本文用10.239.23.4，请根据实际情况修改）访问8080端口：

```
iptables -I INPUT -s 10.239.23.4 -p TCP --dport 8080 -j ACCEPT
```

端口8080要和3.2节Tengine配置文件中的端口设置相同。然后禁止所有访问8080端口的tcp连接：

```
iptables -A INPUT -p TCP --dport 8080 -j DROP
```

查看规则是否生效：

```
iptables -L -n
```

可以看到新增了两条规则：

```
target  prot opt source          destination
ACCEPT  tcp  --  10.239.23.4      0.0.0.0/0      tcp dpt:8080
DROP    tcp  --  0.0.0.0/0        0.0.0.0/0      tcp dpt:8080
```

4. 在GPU物理机上设置代理

回到GPU物理机终端，进行代理设置。最简单的方式是使用环境变量，假设ECS内网IP为10.10.10.10（用户需要自行替换为真实ECS内网IP），则可以执行：

```
export http_proxy=http://10.10.10.10:8080
```

注意这里的代理服务器端口设置应该和ECS跳板机上nginx.conf中监听端口（listen 0.0.0.0:8080）相互对应。也可以将上述语句放入/etc/profile或 ~/.bashrc实现登录GPU物理机时自动配置代理服务器。

5. 测试

在GPU物理机上使用不同的工具测试外网连接情况，结果如下：

5.1 wget和curl测试

```
$ wget http://tengine.taobao.org/download/tengine-2.1.1.tar.gz
--2015-09-21 16:15:50-- http://tengine.taobao.org/download/tengine-2.1.1.tar.gz
Connecting to 10.10.10.10:8080... connected.
Proxy request sent, awaiting response... 200 OK
Length: 2062650 (2.0M) [application/octet-stream]
Saving to: 'tengine-2.1.1.tar.gz'

100%[=====>] 2,062,650  1014KB/s  in 2.0s

2015-09-21 16:15:52 (1014 KB/s) - 'tengine-2.1.1.tar.gz' saved [2062650/2062650]
```

通过以上测试，验证了GPU物理机已经可以通过ECS正向代理访问外网。

5.2 测试vum

```
$ sudo yum install openssl
Loaded plugins: fastestmirror, langpacks
base                                     | 3.6 kB   00:00
Loading mirror speeds from cached hostfile
* base: mirrors.aliyuncs.com
* extras: mirrors.aliyuncs.com
* updates: mirrors.aliyuncs.com
Resolving Dependencies
--> Running transaction check
---> Package openssl.x86_64 1:1.0.1e-42.el7 will be updated
---> Package openssl.x86_64 1:1.0.1e-42.el7.9 will be an update
--> Processing Dependency: openssl-libs(x86-64) = 1:1.0.1e-42.el7.9 for package: 1:openssl-1.0.1e-42.el7.9.x86_64
--> Running transaction check
---> Package openssl-libs.x86_64 1:1.0.1e-42.el7 will be updated
---> Package openssl-libs.x86_64 1:1.0.1e-42.el7.9 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

=====
=====
Package           Arch      Version              Repository    Size
=====
=====
Updating:
openssl           x86_64    1:1.0.1e-42.el7.9    updates      711 k
Updating for dependencies:
openssl-libs      x86_64    1:1.0.1e-42.el7.9    updates      949 k

Transaction Summary
=====
=====
Upgrade 1 Package (+1 Dependent package)

Total download size: 1.6 M
Is this ok [y/d/N]:y
Downloading packages:
Delta RPMs disabled because /usr/bin/applydeltarpm not installed.
(1/2): openssl-1.0.1e-42.el7.9.x86_64.rpm | 711 kB   00:00
(2/2): openssl-libs-1.0.1e-42.el7.9.x86_64.rpm | 949 kB   00:00
-----
Total                               5.7 MB/s | 1.6 MB   00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Updating : 1:openssl-libs-1.0.1e-42.el7.9.x86_64                1/4
  Updating : 1:openssl-1.0.1e-42.el7.9.x86_64                    2/4
  Cleanup  : 1:openssl-1.0.1e-42.el7.x86_64                       3/4
  Cleanup  : 1:openssl-libs-1.0.1e-42.el7.x86_64                 4/4
  Verifying : 1:openssl-libs-1.0.1e-42.el7.9.x86_64              1/4
  Verifying : 1:openssl-1.0.1e-42.el7.9.x86_64                   2/4
  Verifying : 1:openssl-1.0.1e-42.el7.x86_64                     3/4
  Verifying : 1:openssl-libs-1.0.1e-42.el7.x86_64                4/4
```

```
Updated:
  openssl.x86_64 1:1.0.1e-42.el7.9

Dependency Updated:
  openssl-libs.x86_64 1:1.0.1e-42.el7.9

Complete!
```

在ECS跳板机上查看Tengine access log文件（/root/logs/proxy.access.log），找到上述与yum安装相关的log如图所示。

```
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/os/x86_64/Packages/libcom_err-devel-1.42.9-7.el7.x86_64.rpm HTTP/1.1" 502
690 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/updates/x86_64/Packages/krb5-libs-1.12.2-15.el7_1.x86_64.rpm HTTP/1.1" 502
691 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/updates/x86_64/Packages/krb5-devel-1.12.2-15.el7_1.x86_64.rpm HTTP/1.1"
502 692 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/os/x86_64/Packages/keyutils-libs-devel-1.5.8-3.el7.x86_64.rpm HTTP/1.1" 502
692 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/os/x86_64/Packages/libselinux-devel-2.2.2-6.el7.x86_64.rpm HTTP/1.1" 200
178532 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/os/x86_64/Packages/libsepol-devel-2.1.9-3.el7.x86_64.rpm HTTP/1.1" 200
72300 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/os/x86_64/Packages/libverto-devel-0.2.5-4.el7.x86_64.rpm HTTP/1.1" 200
11776 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET http://mirrors.aliyuncs.com/centos/7/os/x86_64/Packages/pcre-
devel-8.32-14.el7.x86_64.rpm HTTP/1.1" 200 488780 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET http://mirrors.aliyuncs.com/centos/7/os/x86_64/Packages/zlib-
devel-1.2.7-13.el7.x86_64.rpm HTTP/1.1" 200 50592 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyun.com/centos/7/os/x86_64/Packages/keyutils-libs-devel-1.5.8-3.el7.x86_64.rpm HTTP/1.1" 200
38232 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:11 +0800] "GET
http://mirrors.aliyun.com/centos/7/os/x86_64/Packages/libcom_err-devel-1.42.9-7.el7.x86_64.rpm HTTP/1.1" 200
30804 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:12 +0800] "GET
http://mirrors.aliyuncs.com/centos/7/updates/x86_64/Packages/openssl-devel-1.0.1e-42.el7.9.x86_64.rpm
HTTP/1.1" 200 1235792 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:12 +0800] "GET
http://mirrors.aliyun.com/centos/7/updates/x86_64/Packages/krb5-devel-1.12.2-15.el7_1.x86_64.rpm HTTP/1.1" 200
655972 "-" "urlgrabber/3.10 yum/3.4.3"
10.239.23.4 - - [21/Sep/2015:17:26:12 +0800] "GET
http://mirrors.aliyun.com/centos/7/updates/x86_64/Packages/krb5-libs-1.12.2-15.el7_1.x86_64.rpm HTTP/1.1" 200
845708 "-" "urlgrabber/3.10 yum/3.4.3"
```

上述安装包所在的yum源为阿里云内网，故ECS代理会通过内网流量获取，这样节省了用户流量费用。

阿里云GPU物理机本身不能被外网直接访问，需要通过ECS反向代理。本文档将指导用户如何设置代理服务器。

1. 确定IP地址

用户应首先确认这几个IP地址：ECS外网IP（不便于透露，本文用XXX.XXX.XXX.XXX表示）和内网IP（实验用10.10.10.10）；GPU物理机内网IP（实验用10.239.23.4）；

2. 登录ECS跳板机

用户可以用PUTTY工具（Windows环境）或SSH命令（Linux环境）登录ECS，注意应使用ECS外网IP登入。

```
ssh -l login_name XXX.XXX.XXX.XXX (ECS外网IP)
```

登录成功后，可以在ECS跳板机上用SSH命令登录GPU物理机：

```
ssh -l root 10.239.23.4 (GPU物理机内网IP)
```

3. ECS跳板机上部署代理服务器

这里选择Tengine，它是在NGINX的基础上由淘宝网发起的开源Web服务器项目。用户应注意，NGINX做前向代理服务器是不支持HTTPS连接的，所以客户端只能访问HTTP服务。

3.1 安装Tengine

重新开一个终端，登录到ECS跳板机。获取Tengine源码：

```
wget http://tengine.taobao.org/download/tengine-2.1.1.tar.gz
```

解压：

```
tar zxvf tengine-2.1.1.tar.gz
cd tengine-2.1.1/
```

配置和编译：

```
./configure
make
sudo make install
```

默认情况下安装位置在 /usr/local/nginx/ 出于测试目的，我们需要在GPU物理机上也安装Tengine，步骤与ECS上安装基本一致，安装路径也在/usr/local/nginx/。

3.2 编辑ECS Tengine配置文件

登录ECS跳板机，用root权限打开 /usr/local/nginx/conf/nginx.conf 文件，增加一个server块，作用为监听本机的10000端口，将所有请求转发给GPU物理机（10.239.23.4:10000），配置内容如下：（“//” 后面为注释，真正的conf文件中应删除）

```
server {
    listen    10000;    // 监听本机的10000端口
    server_name localhost;

    #charset koi8-r;

    #access_log logs/host.access.log main;

    location / {
        proxy_pass http://10.239.23.4:10000;
        // 10.239.23.4为本实验物理机内网IP，请根据需要修改
        // 这里实现了将来自外网的请求转发至物理机
    }

    #error_page 404          /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page 500 502 503 504 /50x.html;
    location = /50x.html {
        root html;
    }
}
```

保存该文件。

3.3 编辑GPU物理机Tengine配置文件和主页

登录GPU物理机，用root权限编辑 /usr/local/nginx/conf/nginx.conf：

```
server {
    listen    10000;
        // 本文实验将默认的80改为10000，用户可根据需要修改
    server_name localhost;

    #charset koi8-r;

    #access_log logs/host.access.log main;

    location / {
        root html;
        index index.html index.htm;
    }
}
```

```
    }  
    .....略.....  
}
```

为了区分GPU物理机和ECS跳板机的主页内容，我们修改物理机的主页（`/usr/local/nginx/html/index.html`）代码如下：

```
<!DOCTYPE html>  
<html>  
<head>  
<title>Welcome to GPU tengine!</title>  
<style>  
  body {  
    width: 35em;  
    margin: 0 auto;  
    font-family: Tahoma, Verdana, Arial, sans-serif;  
  }  
</style>  
</head>  
<body>  
<h1>Welcome to GPU tengine!</h1>  
<p>If you see this page, the tengine web server is successfully installed and  
working. Further configuration is required.</p>  
  
<p>For online documentation and support please refer to  
<a href="http://tengine.taobao.org/">tengine.taobao.org</a>.</p>  
  
<p><em>Thank you for using tengine.</em></p>  
</body>  
</html>
```

ECS上主页保持不变即可。

3.4 启动GPU物理机上的Tengine

登录到GPU物理机，用root权限运行：`sudo /usr/local/nginx/sbin/nginx` 如果报错，请根据报错信息对3.2节中的`nginx.conf`配置文件做必要的修改。查看NGINX进程：

```
# ps -ef | grep nginx  
root  17205  1 0 10:58 ?    00:00:00 nginx: master process /usr/local/nginx/sbin/nginx  
nobody 17206 17205 0 10:58 ?    00:00:00 nginx: worker process  
root   20108 17042 0 11:54 pts/1  00:00:00 grep --color=auto nginx
```

可以看到NGINX主进程的pid为17205，worker进程的pid为17206，在需要关闭web server时可以直接以root权限运行：`kill master_pid`

3.5 设置GPU物理机防火墙

3.5.1 开启GPU物理机防火墙

```
CentOS6: service iptables start
CentOS7: systemctl start firewalld
```

3.5.2 添加防火墙规则

GPU物理机服务器需要使用10000端口，故添加防火墙例外：

```
iptables -I INPUT -p TCP --dport 10000 -j ACCEPT
```

4. 启动ECS反向代理服务

回到ECS跳板机终端，启动代理服务。在启动ECS跳板机上的Tengine之前，我们需要确定ECS跳板机能访问物理机上的web server。在ECS跳板机上运行：`curl 物理机内网IP:端口号` 这里物理机内网IP为10.239.23.4，端口号在3.2节配置为10000，因此命令如下：

```
# curl 10.239.23.4:10000
<!DOCTYPE html>
<html>
<head>
<title>Welcome to GPU tengine!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
</head>
<body>
<h1>Welcome to GPU tengine!</h1>
<p>If you see this page, the tengine web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://tengine.taobao.org/">tengine.taobao.org</a>.</p>

<p><em>Thank you for using tengine.</em></p>
</body>
</html>
```

根据返回内容，对照3.3节的主页修改情况可以确认返回内容来自GPU物理机。确认ECS跳板机可以访问HPC物理机上的web server之后，开启ECS跳板机上的Tengine：（root权限执行）`/usr/local/nginx/sbin/nginx` 如果启动时报错，请根据报错信息修改。查看ECS跳板机上的Tengine进程：

```
# ps -ef | grep nginx
root   16487   1 0 11:33 ?        00:00:00 nginx: master process /usr/local/nginx/sbin/nginx
nobody 16488 16487 0 11:33 ?        00:00:12 nginx: worker process
root   16565 16419 0 12:06 pts/8    00:00:00 grep  nginx
```

可以看到相应主进程和worker进程的pid。注意不要与物理机上的pid混淆。

5. 测试

5.1 ECS上本地回环测试

可以在ECS跳板机上直接用curl进行本地回环测试：

```
# curl http://localhost:10000
<!DOCTYPE html>
<html>
<head>
<title>Welcome to GPU tengine!</title>
<style>
  body {
    width: 35em;
    margin: 0 auto;
    font-family: Tahoma, Verdana, Arial, sans-serif;
  }
</style>
</head>
<body>
<h1>Welcome to GPU tengine!</h1>
<p>If you see this page, the tengine web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://tengine.taobao.org/">tengine.taobao.org</a>.</p>

<p><em>Thank you for using tengine.</em></p>
</body>
</html>
```

对比3.3节的主页内容，可见返回的确实是GPU物理机上web server主页。

5.2 远程web页面测试

在浏览器的地址栏输入：<http://XXX.XXX.XXX.XXX:10000/>（这里XXX.XXX.XXX.XXX是ECS跳板机的公网IP），显示结果如下图所示：

可见，实现了远程访问HPC物理机上的web server，也验证了通过ECS进行反向代理的有效性。

使用流程

您在购买了高性能计算产品后，可以通过 **管理控制台** 找到 **高性能计算** 进行实例的管理。快速入口 <https://hpc.console.aliyun.com/>。

购买HPC实例后

用户购买高性能计算实例后，对于每个实例，用户将获得以下登录信息：

ECS公网IP。

ECS内网IP。

GPU物理机内网IP。

ECS登录账号（默认为root）和密码。

GPU物理机登录账号（默认为root）和密码。

首次登录之前

对于新购高性能计算实例，请修改初始化登录密码。

登录密码将作为您登录高性能计算实例的唯一凭证。

阿里云将不会以任何形式储存，因而无法提供密码找回功能。

跳转机密码需要进行重置才能登录。

初始物理机登录密码将通过站内消息发送给您，可在 **消息中心** 中进行查看。

重置跳板机密码

跳板机是您访问物理机前，需要首先登陆的虚拟机，随物理机所赠送。

在 **高性能计算** 控制台中，点击 **实例** 标签。

选择对应的高性能计算实例的 **操作** 选择 **跳板机**，点击 **重置跳板机密码**。

在弹出的对话框中输入新密码，需要符合规定的密码强度。

需要重启跳板机后登录密码才能生效。

重置物理机密码

物理机仅能通过对应的跳板机进行访问，与其他用户之间网络完全隔离。

由于您拥有完全的访问权限，您可以在登录后输入 `passwd` 根据命令行提示进行修改。

物理机登录密码修改后将无法由控制台进行重置。

登录GPU物理机

登录方式如下：

在控制台中查看跳板机的公网 IP、内网 IP 以及物理机的内网 IP。

登录跳板机，打开终端输入 `ssh root@跳板机公网IP` 输入跳板机密码后登录到跳板机。

登录到跳板机后，在终端输入 `ssh root@物理机内网IP` 输入物理机密码后登录到物理机。

登录到物理机后，您将可以操作具有极致性能的 HPC 产品。

您在购买了高性能计算实例后，可以通过**管理控制台**找到**高性能计算**进行实例的管理。或者选择快速入口进入 HPC控制台

购买HPC实例后

用户购买高性能计算实例后，对于每个实例，用户将获得以下登录信息

1. GPU物理机内网IP
2. GPU物理机登录账号（默认为root）和密码（通过控制台站内信告知）

华北2 区域HPC不再提供ECS跳板机，需要使用用户自己的任意一台华北2 区域经典网络ECS作为跳板机

首次登录之前

对于新购高性能计算实例，请修改HPC实例初始化登录密码。

1. 登录密码将作为您登录高性能计算实例的唯一凭证。
2. 初始登录密码将通过站内消息发送给您，可在 **消息中心** 中进行查看。
3. 若忘记密码，可以通过控制台重置HPC实例密码。

重置物理机密码

加入安全组的ECS实例以及同一账户下的HPC实例之间可以相互访问，但与其他用户之间网络完全隔离。

1. 由于您拥有完全的访问权限，您可以在登录后输入 passwd 根据命令行提示进行修改。
2. 可以通过控制台重置HPC实例密码。

登录GPU物理机

登录方式如下：

1. 在控制台中查看加入 HPC 安全组规则的 ECS 实例信息。公共云HPC如何使用安全组
2. 为保障 HPC 能够内网访问 ECS，请设置您接入 HPC 的 ECS 所在的安全组规则 内网入方向已经打开。
ECS安全组默认规则
3. 登录跳板机，打开终端输入 ssh root@跳板机公网IP 输入跳板机密码后登录到跳板机
4. 登录到跳板机后，在终端输入ssh root@物理机内网IP输入物理机密码后登录到物理机
5. 登录到物理机后，您将可以操作具有极致性能的 HPC 产品

控制台使用

进入控制台后，单击左侧的**实例**。

管理控制台 产品与服务									
高性能计算 HPC									
实例管理									
实例名称 请输入实例名进行模糊查找 搜索									
名称	状态	配置	物理机 IP	跳板机 IP	外网带宽	生命周期	操作		
215268770	物理机: 运行中 跳板机: 运行中	CPU: Intel Xeon 8-core x2 GPU: NVIDIA Tesla K40 x2	10.172.66.XX	121.40.202.XX (公) 10.168.61.XX(内)	10 Mbps	创建: 北京时间 2016-01-09 11:00:00 到期: 北京时间 2016-01-16 11:00:00	跳板机 物理机		

查看实例

购买HPC后，您可以进入控制台查看购买的实例，包括物理机和ECS的状态、物理机IP、ECS的公网和内网IP以及实例生命周期等等。

实例名称 ▾

请输入实例名进行模糊查找

搜索

名称	状态	配置	物理机 IP	跳板机 IP	外网带宽	生命周期	操作
215268770	物理机: 运行中 跳板机: 运行中	CPU: Intel Xeon 8-core x2 GPU: Nvidia Tesla K40 x2	10.172.66.XX	121.40.202.XX (公) 10.168.61.XX(内)	10 Mbps	创建: 北京时间 2016-01-09 11:00:00 到期: 北京时间 2016-01-16 11:00:00	跳板机 ▾ 物理机 ▾

重启物理机

您可以在控制台对物理机进行重启。首先找到想要重启的物理机，单击该实例右侧的**物理机**选项，然后在下拉菜单中选择**重启物理机**。

实例名称 ▾

请输入实例名进行模糊查找

搜索

名称	状态	配置	物理机 IP	跳板机 IP	外网带宽	生命周期	操作
215268770	物理机: 运行中 跳板机: 运行中	CPU: Intel Xeon 8-core x2 GPU: Nvidia Tesla K40 x2	10.172.66.XX	121.40.202.XX (公) 10.168.61.XX(内)	10 Mbps	创建: 北京时间 2016-01-09 11:00:00 到期: 北京时间 2016-01-16 11:00:00	跳板机 ▾ 物理机 ▾
							重启物理机

操作ECS跳板机

您可以在控制台对ECS跳板机进行启动、关闭、重启和重置密码等操作。首先找到想要重启ECS的实例，点击该实例右侧的**跳板机**选项，然后在下拉菜单中选择需要的操作。

实例名称 ▾

请输入实例名进行模糊查找

搜索

名称	状态	配置	物理机 IP	跳板机 IP	外网带宽	生命周期	操作
215268770	物理机: 运行中 跳板机: 运行中	CPU: Intel Xeon 8-core x2 GPU: Nvidia Tesla K40 x2	10.172.66.XX	121.40.202.XX (公) 10.168.61.XX(内)	10 Mbps	创建: 北京时间 2016-01-09 11:00:00 到期: 北京时间 2016-01-16 11:00:00	跳板机 ▾
							启动跳板机 关闭跳板机 重启跳板机 重置跳板机密码

默认安全组

- 1. 目前仅支持经典网络
- 2. 目前仅支持**华北2**

目前暂不支持用户增加新安全组，当用户开通HPC服务后，系统会自动为客户创建一个名为default的默认安全组



目前最多允许用户增加**32条安全组规则**

默认安全组的默认网络访问控制规则为:

- i. 内网入方向 deny all, 内网出方向 accept all;
- ii. HPC暂不支持直接访问公网
- iii. 同一账号的多个HPC实例可以相互访问

查询安全组内HPC实例

1. 登录HPC管理控制台
2. 单击左侧导航中的**安全组**
3. 选择地域为:**华北2**

选择default安全组，单击操作中的**管理实例**



在新窗口中可以看到您账号下所有的HPC实例，这些实例之间可以内网相互访问



查询安全组内安全组规则

1. 登录HPC管理控制台
2. 单击左侧导航中的**安全组**
3. 选择地域为:**华北2**

选择default安全组，单击操作中的**配置规则**



新页面中将显示您目前所有的安全组规则



授权安全组规则

授权安全组规则可以允许或者禁止与安全组相关的HPC实例的内网入出方向的访问，您可以随时授权和取消安全组规则。您的变更安全组规则会自动应用于与安全组相关联的HPC实例上。用户自己的ECS要访问HPC时，必须先给HPC增加一条安全组规则，允许该ECS访问HPC。**安全组规则的优先级不能重复**

操作步骤

1. 登录HPC管理控制台
2. 单击左侧导航中的**安全组**
3. 选择地域为**华北2**

选择default安全组，单击操作中的**配置规则**



在新窗口中单击右上角的**添加规则**



在弹出的对话框中，设置下面参数：

- i. 需要允许或拒绝访问的ECS IP地址

- ii. 授权策略：目前仅支持允许操作，因此下拉菜单中选择 **允许**
- iii. 网络类型：目前仅支持内网，因此下拉菜单中选择 **私有**

优先级：填写该规则的优先级，优先级为1-2999内整数，数字越大，优先级约低

新建安全组规则

* 输入IP:

127.0.0.1

① 请输入正确IP地址

* 授权策略:

允许

① 请选择

* 网络类型:

私有

① 请选择

* 优先级:

1000

① 优先级在1-5000内, 且不能重复

确定

取消

1. 点击 **确定**，成功为该安全组授权一条安全组规则

管理控制台 产品与服务

安全组规则

安全组实例列表

安全组规则

IP地址

请输入IP地址进行精确查找

搜索

源IP地址	授权策略	网络类型	优先级	创建时间	操作
	accept	Intranet	999	2016-12-02 22:35:56	删除规则
	accept	Intranet	997	2016-12-14 17:15:19	删除规则
127.0.0.1	accept	Intranet	1000	2016-12-15 15:29:52	删除规则

批量删除

共有3条。每页显示: 5条

删除安全组规则

操作步骤

1. 登录HPC管理控制台
2. 单击左侧导航中的**安全组**
3. 选择地域为: **华北2**

选择default安全组，单击操作中的**配置规则**



选择您要删除的规则，点击删除规则



或者您也可以批量删除安全组规则，选择需要删除的安全组规则前面选择框，单击批量删除



桌面环境设置方法

在HPC物理机上，如果需要使用桌面环境，可以参考如下步骤：

安装X Window运行 `yum groups install "X Window System"`

安装桌面这里以KDE桌面为例。运行 `yum groups install "KDE Plasma Workspaces"`

安装VNCServer运行 `yum install tigervnc-server`

启动VNCServer运行 `vncserver -geometry 1920x1080`后面窗口尺寸可以根据你的本地显示器分辨率进行修改。此时会提示需要创建VNC登录密码，根据需要设置。

VNC端口代理设置在ECS跳板机上，运行命令：`ssh -4 -f -g -L 5901:GPU私网IP:5901 -N localhost` 开启端口代理。这样可以通过ECS公网IP访问物理机上的VNCServer。

使用VNCClient登录Windows下可以使用Linux下可以使用Mac下可以使用Chicken of the VNC工

具

如果用户需要将本地文件传输到HPC上，除了使用OSS，还可以将HPC磁盘通过NFS方式挂载到ECS跳板机，利用ECS跳板机的公网IP可以直接从用户本地scp拷贝到挂载的物理机磁盘上。注意，这里ECS跳板机为NFS Client，而HPC物理机为NFS Server。假设物理机和ECS跳板机的操作系统均为Cent OS 7，其他系统可根据实际情况做调整。

1. ECS跳板机上安装nfs

```
yum upgrade lvm2
yum install nfs-utils
```

注意：如果没有执行yum upgrade lvm2，在安装nfs-utils会报错：

Transaction check error:

file /usr/lib/systemd/system/blk-availability.service from install of device-mapper-7:1.02.107-5.el7_2.1.x86_64 conflicts with file from package lvm2-7:2.02.105-14.el7.x86_64

file /usr/sbin/blkdeactivate from install of device-mapper-7:1.02.107-5.el7_2.1.x86_64 conflicts with file from package lvm2-7:2.02.105-14.el7.x86_64

file /usr/share/man/man8/blkdeactivate.8.gz from install of device-mapper-7:1.02.107-5.el7_2.1.x86_64 conflicts with file from package lvm2-7:2.02.105-14.el7.x86_64

HPC物理机上安装nfs

```
yum install nfs-utils
```

```
vi /etc/exports
```

```
/disk1 ECS跳板机内网ip(rw,secure,no_root_squash,sync)
```

例如：

```
/disk1 10.117.12.27 (rw,secure,no_root_squash,sync)
```

```
vi /etc/hosts
```

GPU内网ip GPU的hostname

例如：

```
10.172.66.99 AliHPC-GPU017
```

HPC物理机上启动NFS服务

```
service rpcbind start
service nfs start
```

查看是否生效：

```
showmount -e
```

显示：

```
Export list for AliHPC-GPU017:  
/disk1 10.117.12.27
```

1. 在ECS跳板机上挂载物理机磁盘

查看GPU上export的地址：

```
showmount -e GPU内网ip
```

例如：

```
showmount -e 10.172.66.99
```

显示：

```
Export list for 10.172.66.99:  
/disk1 10.117.12.27
```

mount GPU上的nfs目录：

```
mount -t nfs GPU内网ip:/disk1 /disk1
```

例如：

```
mkdir /disk1  
mount -t nfs 10.172.66.99:/disk1 /disk1
```

运行后，物理机的/disk1将被挂载至ECS跳板机的/disk1下。此时用户可以在本地执行scp命令，将文件拷贝至ECS跳板机的/disk1/下，即可自动同步到物理机。

```
scp my_local_file username@ECS公网IP:/disk1/
```

输入登录信息即可启动文件传输。

访问其他云产品

应用场景

当用户需要将大量数据（几十GB到上TB）传输到HPC机器上时，使用scp拷贝有两个缺陷：（1）需要通过ECS中转数据，scp须执行两次；（2）数据太大时ECS自带磁盘容量很可能不够用；此时，使用OSS服务是比较理想的方式，无需ECS中转，直接实现端对端的传输，而且在HPC上通过内网访问OSS，速度更快、更稳定。本节介绍如何在HPC机器上使用OSS服务。

物理机访问OSS方法

首先登录阿里云官网，进入OSS控制台<https://oss.console.aliyun.com/index#/>

点击Access Keys，“显示” Access Key，用手机短信验证。

拿到Access Key ID和Access Key Secret，用于鉴别身份。

创建一个Bucket，位于华东1区。其host为oss-cn-**-internal.aliyuncs.com，具体可以在控制台查到。

注意：目前GPU物理机通过内网只能访问华东1区的OSS服务

在HPC机器上运行：

```
osscommand config --host=oss-cn-*****-internal.aliyuncs.com \
--id=用户的OSS Access Key ID \
--key=用户的OSS Access Key Secret
```

之后运行

osscommand ls

可以看到你创建的Bucket，说明配置成功，可以通过osscommand进行文件上传/下载操作。

先浏览有哪些bucket：

```
# osscommand ls
CreateTime      BucketLocation  BucketName
2015-09-17 17:17:08 oss-cn-*****  hpc-data-release
2015-09-15 11:29:58 oss-cn-*****  hpc100users

Bucket Number is: 2
0.047(s) elapsed
```

可见有两个bucket，都位于华东1区。

进一步浏览其中一个bucket详细内容：

```
# osscommand ls oss://hpc100users
prefix list is:
object list is:
2015-09-18 10:38:49 1066.97MB Standard oss://hpc100users/cuda_7.0.28_linux.run
(oss://hpc100users/cuda_7.0.28_linux.run)

prefix list number is: 0
object list number is: 1
0.022(s) elapsed
```

注意OSS路径都以"oss://"开头，后面紧跟bucket名称，再向后为文件名或目录名。

下载命令：

```
# osscommand get oss://Bucket名称/文件名称 本地文件名称
```

测试下载：

```
# osscmd get oss://hpc-data-release/cuda_7.0.28_linux.run cuda_7.0.28_linux.run
100% The object cuda_7.0.28_linux.run is downloaded to cuda_7.0.28_linux.run, please check.
61.593(s) elapsed
```

下载cuda_7.0.28_linux.run到本地用时61.593s，该文件大小为1066.97MB，下载速度约为17.3 MB/s
上传命令：

```
# osscmd put 本地文件名称 oss://Bucket名称/文件名称
```

测试上传：

```
# osscmd put cuda_7.0.28_linux.run oss://hpc-data-release/cuda_7.0.28_linux.run
100%
Object URL is: http://hpc-data-release.oss-cn-*****-internal.aliyuncs.com/cuda_7.0.28_linux.run
Object abstract path is: oss://hpc-data-release/cuda_7.0.28_linux.run
ETag is "312AEDE1C3D1D3425C8CAA67BBB7A55E"
61.287(s) elapsed
```

常见问题

运行osscmd ls后报错如下：

Error Headers:

```
[('content-length', '467'), ('server', 'AliyunOSS'), ('connection', 'keep-alive'), ('x-oss-request-id',
'56921E152530B7D05CA805DD'), ('date', 'Sun, 10 Jan 2016 09:02:13 GMT'), ('content-type', 'application/xml')]
```

Error Body:

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
  <Code>RequestTimeTooSkewed</Code>
  <Message>The difference between the request time and the current time is too large.</Message>
  <RequestId>56921E152530B7D05CA805DD</RequestId>
  <HostId>oss-cn-*****-internal.aliyuncs.com</HostId>
  <MaxAllowedSkewMilliseconds>900000</MaxAllowedSkewMilliseconds>
  <RequestTime>2015-10-09T16:00:04.000Z</RequestTime>
  <ServerTime>2016-01-10T09:02:13.000Z</ServerTime>
</Error>
```

Error Status:

```
403
ls Failed!
```

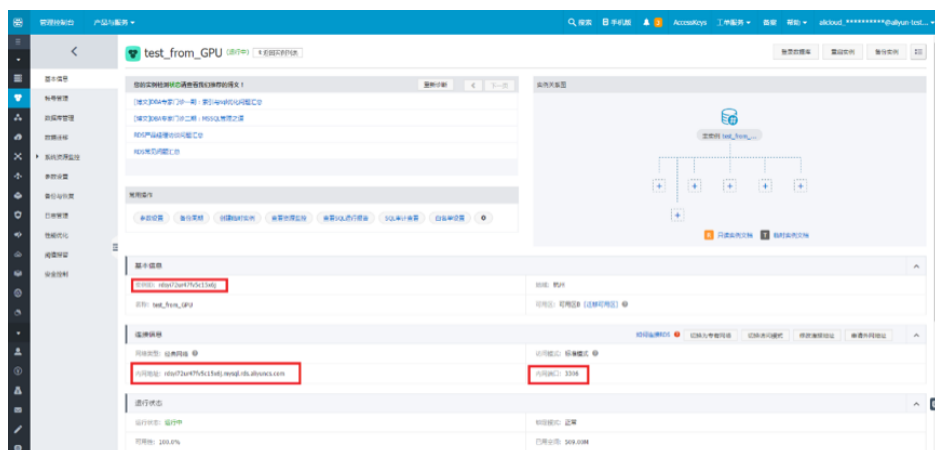
这是由于HPC机器日期与OSS服务器日期差别太大造成的。通过date命令修改日期时间为正确的时间即可。

HPC访问云数据库RDS

购买和配置RDS

用户可以登录阿里云官网购买RDS实例。

购买成功后，在RDS管理控制台可以看到当前实例：



注意将内网地址、端口号和访问账号记录下来，访问密码在创建实例时已经设置，如果忘记，则需要通过“账号管理”——“修改密码”进行修改。

重要：在“常用操作”中点击“白名单设置”，将GPU物理机的内网IP加入白名单，这里实验用10.239.23.4。设置完成后如下图所示：



GPU物理机访问RDS

目前GPU物理无法直接访问RDS，需要通过ECS中转，方法如下。

如果物理机上没有安装mysql，则运行以下命令安装：

```
sudo yum install mysql
```

建立ssh tunnel，在GPU物理上运行命令

```
ssh -fN -v -N root@ECS内网IP -L 中转端口号/RDS内网地址/RDS内网端口
```

其中中转端口号由用户指定，RDS内网端口默认为3306，可以在控制台查到，RDS内网地址用户可以在控制台查到。

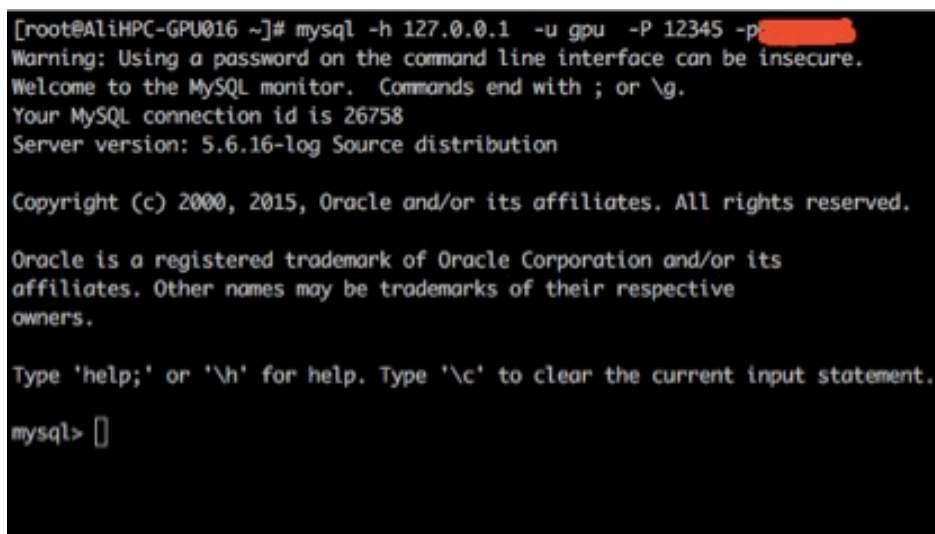
运行上述命令后，输入ECS跳板机的密码。

之后，执行下面命令来登录RDS：

```
mysql -h 127.0.0.1 -u dbuser -P 12345 -ppassword
```

其中-h 指定本地IP-u 是RDS的访问账号，在控制台创建，本例中用户名为dbuser-P 是上一步中指定的中转端口号-p 后面是dbuser对应的密码，-p与密码之间无空格

连接成功后出现如下信息

A terminal window screenshot showing a successful MySQL connection. The prompt is [root@AliHPC-GPU016 ~]#. The command executed is mysql -h 127.0.0.1 -u gpu -P 12345 -p[redacted]. The output includes a warning about using a password on the command line, a welcome message to the MySQL monitor, the connection ID 26758, and the server version 5.6.16-log Source distribution. It also displays copyright information for Oracle and instructions for help. The prompt is now mysql>.

```
[root@AliHPC-GPU016 ~]# mysql -h 127.0.0.1 -u gpu -P 12345 -p[redacted]
Warning: Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 26758
Server version: 5.6.16-log Source distribution

Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

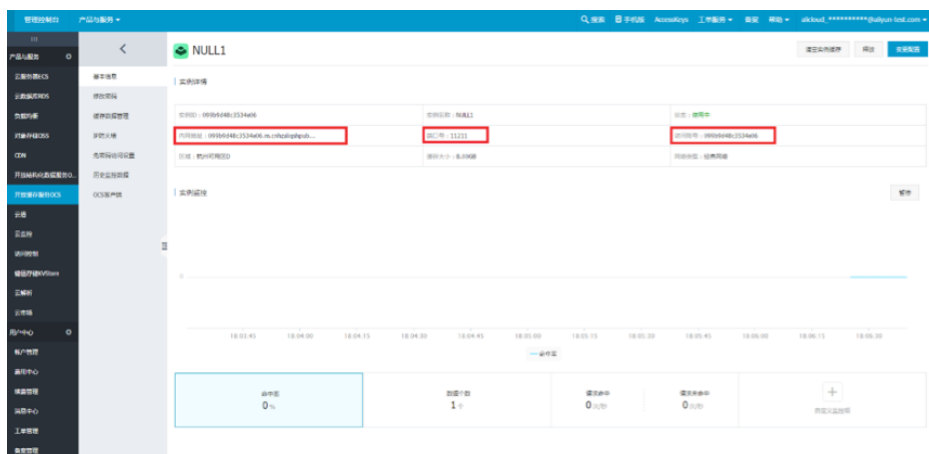
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

上图中用户名为gpu，使用的中转端口为12345

用户可以在[在阿里云官网购买]云数据库memcache版实例(<http://www.aliyun.com/product/ocs>)

购买成功后，在云数据库memcache版管理控制台可以看到当前实例。



注意将内网地址、端口号和访问账号记录下来，访问密码在创建实例时已经设置，如果忘记，则需要点击左侧面板“修改密码”进行修改。

通过ECS跳板机登录GPU物理机，从github上下载文件python-binary-memcached-0.24.2.tar.gz并拷贝到物理机上。

解压：

```
tar zxvf python-binary-memcached-0.24.2.tar.gz
```

安装：

```
cd python-binary-memcached-0.24.2/
sudo python setup.py install
```

目前物理机无法直接访问云数据库memcache版，需要通过ECS中转，方法如下。

建立ssh tunnel，在GPU物理上运行命令

```
ssh -fN -v -N root@ECS内网IP -L 中转端口号/云数据库memcache版内网地址/云数据库memcache版内网端口
```

其中中转端口号由用户指定，云数据库memcache版端口号、云数据库memcache版内网地址用户可以在控制台查到。

运行上述命令后，输入ECS跳板机的密码。

之后，物理机可以访问云数据库memcache版，但是需要用localhost来替代云数据库memcache版内网地址，用中转端口号来替换云数据库memcache版端口号。

测试：在物理机上编写test_ocs.py代码如下：

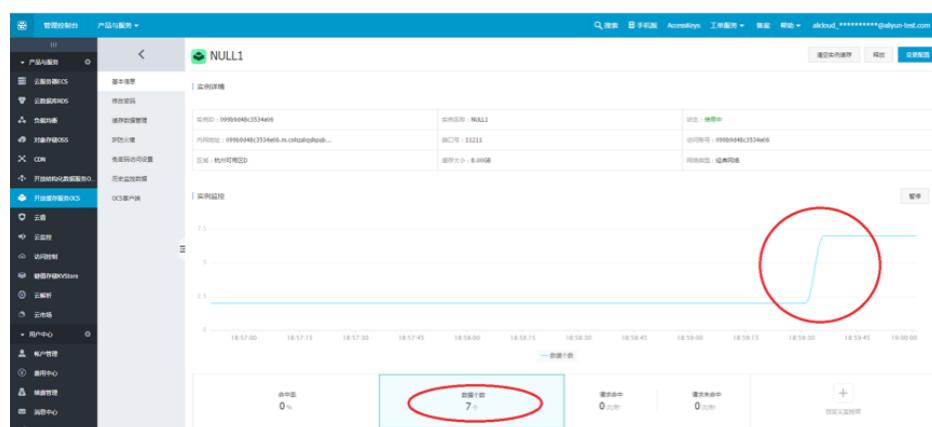
```
#!/usr/bin/env python
import bmemcached
client = bmemcached.Client(('localhost:中转端口号'), 'OCS访问账号', 'OCS密码')
print client.set('city1', 'Beijing')
print client.set('city2', 'Shanghai')
print client.set('city3', 'Guangzhou')
```

```
print client.set('city4', 'Shenzhen')
print client.set('city5', 'Hangzhou')
print client.get('city5')
print client.get('city4')
print client.get('city3')
print client.get('city2')
print client.get('city1')
```

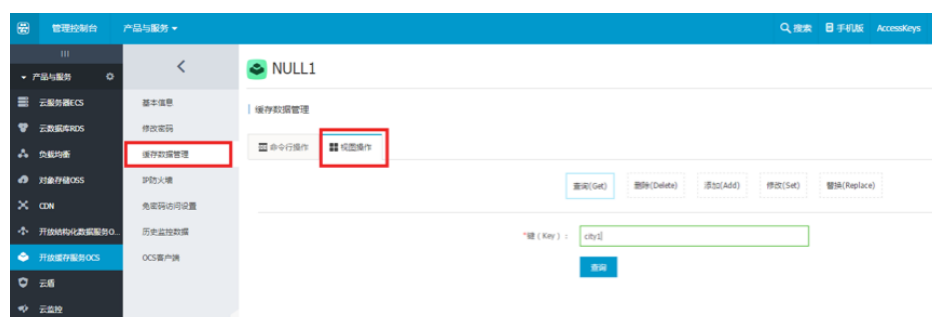
运行：

```
python test_ocs.py
True
True
True
True
True
Hangzhou
Shenzhen
Guangzhou
Shanghai
Beijing
```

回到OCS管理控制台查看，如下图所示



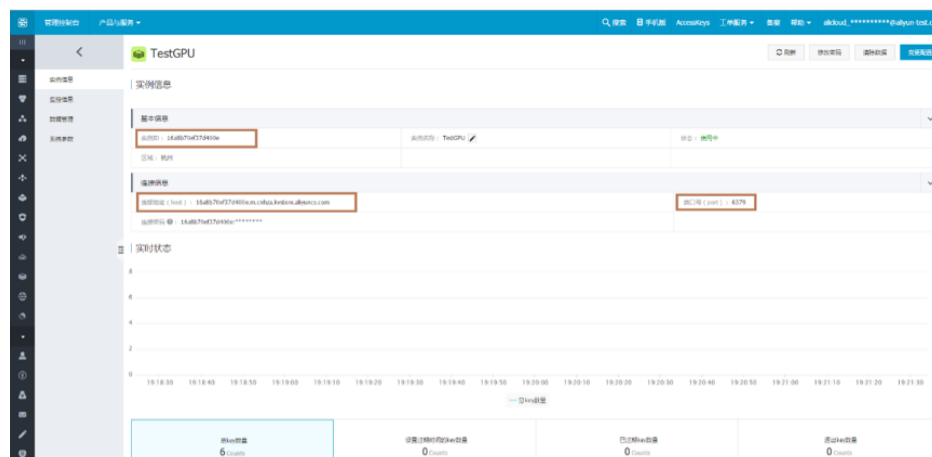
可以看到OCS的数据个数增加了5个，这是由GPU物理机写入的。查看数据可以通过控制台左侧面板中的“缓存数据管理”实现。



依次输入“city1”~“city5”，结果应该与test_ocs.py中预设值一致。

关于OCS更多文档参考OCS文档中心。

购买成功后，在云数据库Redis版可以看到当前实例：



通过ECS跳板机登录GPU物理机。 获取redis-py：

```
git clone https://github.com/andymccurdy/redis-py.git
```

（如果不能访问外网，需要参考如何通过ECS设置代理）

安装：

```
cd redis-py/  
sudo python setup.py install
```

建立ssh tunnel, 在GPU物理上运行命令

```
ssh -fN -v -N root@ECS内网IP -L 中转端口号/云数据库Redis版内网地址/云数据库Redis版内网端口
```

运行上述命令后，输入ECS跳板机的密码。

测试：在物理机上编写test_kvstore.py代码如下：

```
#!/usr/bin/env python
#-*- coding: utf-8 -*-
import redis

#这里替换为localhost的内网IP和中转端口号
host = 'localhost'
port = 中转端口号

#这里替换为实例id和实例password
```

```
user = '16a8b70ef37d400e'
pwd = '*****'

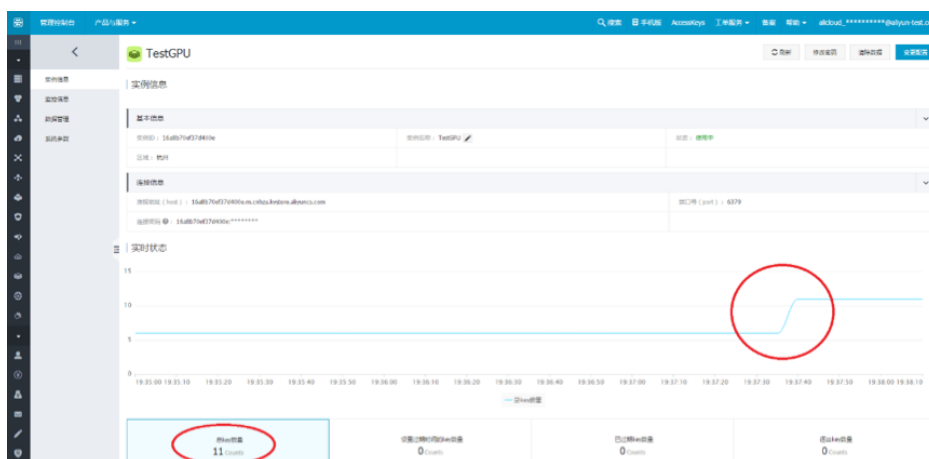
#连接时通过password参数指定AUTH信息，由user,pwd通过":"拼接而成
r = redis.StrictRedis(host=host, port=port, password=user+'.'+pwd)

#连接建立后就可以进行数据库操作，详情文档参考https://github.com/andymccurdy/redis-py
r.set('book1', 'Algorithms');
r.set('book2', 'Assembly');
r.set('book3', 'Mathmatics');
r.set('book4', 'Artists');
r.set('book5', 'Calculors');
print r.get('book1')
print r.get('book2')
print r.get('book3')
print r.get('book4')
print r.get('book5')
```

运行：

```
python test_kvstore.py
Algorithms
Assembly
Mathmatics
Artists
Calculors
```

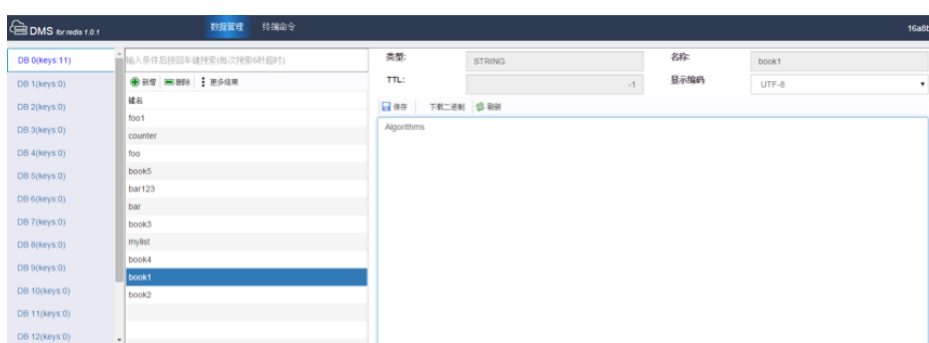
回到KVStore管理控制台查看，如下图所示



可以看到Key个数增加了5个，这是由GPU物理机写入的。查看数据可以通过控制台左侧面板中的“数据管理”实现



点击“立即体验”进入阿里云数据管理系统（Aliyun DMS），界面如下：



依次点击“book1” ~ “book5”，结果应该与test_kvstore.py中预设值一致。

关于KVStore更多文档参考KVStore文档中心。

软件镜像

在集群环境中，物理机上的数据卷有很大的局限性：

- 容器在机器间迁移时，数据无法迁移
- 不同机器之间不能共享本地数据卷

另一方面，在高性能计算和深度学习的场景里，需要使用大量的数据。上传、共享和下载数据也需要借助OSSFS和NAS这类数据服务。

为了解决这一矛盾，阿里云容器服务提供第三方数据卷。将各种云存储包装成数据卷，可以直接挂载在容器上，并在容器重启、迁移时自动重新挂载。目前支持 OSSFS 数据卷，您可以通过容器服务快速挂载数据卷。

注意：目前北京HPC仅支持OSS数据卷，后续会提供NAS的支持。

创建OSS bucket

首先您需要创建一个和容器服务处于同一区域中的bucket，并且将权限设置为私有。这样容器应用可以通过内网地址来访问bucket中存储的文件数据，提升访问速度并节省公网带宽：

新建Bucket

BucketName:

neural-art

Bucket命名规范:
» 只能包含小写字母、数字和短横线
» 必须以小写字母和数字开头和结尾
» bucketName的长度限制在3-63之间

所属地域:

华东 2

相同地域内的产品内网可以互通；订购后不支持更换地域，请谨慎选择

读写权限:

私有

» 私有：对object的所有访问操作需要进行身份验证
» 公共读：对object写操作需要进行身份验证；可以对object进行匿名读
» 公共读写：所有人都可以对object进行读写操作

提交

取消

注：更多的细节可以查看oss新手入门。

利用OSS客户端上传和下载数据

建议您在本地电脑上下载云市场中的OSS图形客户端，您在本地电脑上只需要通过图形用户界面上简单的拖拽实现文件的上传和下载。



在容器服务中创建OSS数据卷

数据卷是Docker提供的容器储存模型，可以实现容器和数据生命周期的解耦，当容器被删除或重建之后数据依然存在；提供了可扩展的插件机制，支持不同的存储实现。

阿里云容器服务内置了针对阿里云的数据卷驱动，支持不同类型的云存储服务：包括NAS(文件存储服务 NFS)，OSS(对象存储服务，OSSFS)和云盘（即将推出）。关于数据卷的详细信息可以参阅帮助文档。目前阿里云北京HPC容器服务支持OSS数据卷。

点击容器服务控制台左侧**数据卷**，展开数据卷功能。目前仅有OSS数据卷是开放的。

OSS数据卷将oss的bucket包装成数据卷，创建界面如下：

创建数据卷

数据卷类型: ☒ OSS

数据卷名:

AccessKey ID:

AccessKey Secret:

其他参数:

其他参数的填写格式可以 参考该文档，例如 -o allow_other -o default_permission=666 -onoxattr

注意：只有volume driver在0.7版本及以上的集群才支持该参数。您可以到“应用”列表中找到 acsvolumedriver 应用，然后在其详情页的服务列表中查看volumedriver服务的镜像版本，如果是0.7以下的话，请联系客服升级volumedriver

Bucket ID: [选择Bucket](#)

访问域名: ☒ 内网域名

文件缓存: ☐ 打开 ☒ 关闭

[创建](#) [取消](#)

- **数据卷名**：数据卷的id，在集群内唯一。
- **AccessKeyId、AccessKeySecret**：访问OSS所需的AK，可以从**AK控制台**获取。
- **访问域名**：如果bucket跟ECS在同一个区（Region），选内网域名；否则选外网域名。
- **文件缓存**：如果需要在不同机器间同步同一个文件的修改（比如在A机器中修改文件，在B机器中读取修改后的内容），请关闭文件缓存。但请注意，关闭文件缓存将导致ls文件夹变得很慢，尤其是同一个文件夹下文件比较多时。没有上述需求时，请打开文件缓存，提高ls的速度。

3. 点击**选择Bucket**按钮，选择您之前创建neural-art数据卷，再点击**创建**：

[redacted]	oss-cn-shanghai	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-shanghai	选择
billing-measure-data-packages	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
[redacted]	oss-cn-hangzhou	选择
neural-art	oss-cn-beijing	选择
[redacted]	oss-cn-hangzhou	选择

取消

4. 这样就能看到您创建的OSS数据卷已经成功的出现在了数据卷列表中：

小助手: 数据卷指南

集群: HPC

☐ 节点	卷名	驱动	挂载点	引用容器	卷参数	操作
☐ [redacted]	87eeaecc6bc1ed49137ce7b9...	本地磁盘	/disk2/docker/volumes/87...	dashboard_grafana_1		删除所有同名卷
☐ [redacted]	ae554d1d05330a3f292570b2...	本地磁盘	/disk2/docker/volumes/ae...	dashboard_grafana_1		删除所有同名卷
☐ [redacted]	neural-art	oss文件系统	/mnt/acs_mnt/ossfs/neura...		查看	删除所有同名卷
☐ [redacted]	d2276eff7493321e1bda501...	本地磁盘	/disk2/docker/volumes/d2...	dashboard_grafana_1		删除所有同名卷
☐ [redacted]	nvidia_driver_367.48		/var/lib/nvidia-docker/v...		查看	删除所有同名卷
☐ [redacted]	5bd50d6d67de9f395ede8986...	本地磁盘	/disk2/docker/volumes/5b...	influxdb		删除所有同名卷

5. 此时您集群中的每台HPC机器（包括未来新加入集群的HPC机器）就能在/mnt目录下看到您挂载的数据卷了。您也可以在使用容器服务的时候指定Volumes了。

```
version: '2'
labels:
  aliyun.project_type: "batch"
services:
  neural:
    image: registry-internal.cn-beijing.aliyuncs.com/cheyang/neural-style:latest
    volumes:
      - neural-art:/neural
```