阿里云同步资产信息到 Jumpserver

作者：cuijianzhe
原文链接：https://ld246.com/article/1595154506096
来源网站：链滴

同步资产信息到Jumpserver

Jumpserver开发文档:

- **创建API Key**

阿里云API文档:

具体代码:

- 添加本地IDC机房服务器 **白名单**，避免同步阿里云删除本地机房服务器资产
- 同步比较Jumpserver创建的重复资产
- 实时同步线上资产到Jumpserver
- 同步操作发送结果到 **钉钉**消息
- 根据阿里云标签键进行同步到Jump对应的组里（prod:ops--->prod-ops）

```
#!/usr/bin/env python3
#coding=utf-8
import json
import requests
import time
from httpsig.requests_auth import HTTPSignatureAuth

from aliyunsdkcore.client import AcsClient
from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRe
uest
from collections import Counter
#新建jumpserver
KEY_ID = 'KEY_ID'
SECRET = 'SECRET'
Jumpserver_url = 'https://opt-jumpserver.limikeji.com'
#aliyun参数
aliyun_ip_list = []
aliyun_name_list = []
aliassets_node = {}  #阿里云标签字典
#jumpserver
JumpIP_list = []
```

```
JumpID_list = []

class aliyun_ecs():
    def __init__(self):
        self._client =  AcsClient('<accessKeyId>', '<accessSecret>', 'cn-beijing')

    def page_num(self):
        request = DescribeInstancesRequest()
        request.set_accept_format('json')
        response = json.loads(self._client.do_action_with_exception(request))
        _ecs_num = response.get('TotalCount')//100 + 2
        return _ecs_num

    def assets_list(self):
        request = DescribeInstancesRequest()
        request.set_accept_format('json')
        request.set_PageSize(100)
        for num in range(1,self.page_num()):
            request.set_PageNumber(num)
            response = json.loads(self._client.do_action_with_exception(request))
            instances_list = response.get('Instances').get('Instance')
            for info in instances_list:
                assetsName = info.get('InstanceName')
                aliyun_name_list.append(assetsName)
                assetsIp = ''.join(info.get('VpcAttributes').get('PrivateIpAddress').get('IpAddress'))
                aliyun_ip_list.append(assetsIp)
                try:
                    TagKey = info.get('Tags').get('Tag')
                    aliassets_node[assetsIp] = TagKey[0]['TagKey'] + '-' + TagKey[0]['TagValue']
                except AttributeError:
                    aliassets_node[assetsIp] = 'Jumpserver资产同步'

class new_Jumpserver():
    def __init__(self,host=Jumpserver_url,keyid=KEY_ID,secret=SECRET):
        self.host = host
        self.keyid = keyid
        self.secret = secret

    def _auth(self):
        signature_headers = ['(request-target)', 'accept', 'date', 'host']
        auth = HTTPSignatureAuth(key_id=self.keyid, secret=self.secret,
                                 algorithm='hmac-sha256',
                                 headers=signature_headers)
        return auth
    def _headers(self):
        headers = {
            'Accept': 'application/json',
            'Date': str(time.strftime("%a %b %d %H:%M:%S %Y", time.localtime()))
        }
        return headers

    def get_assets(self):
        url = self.host + '/api/v1/assets/assets/'
        req = requests.get(url, auth=self._auth(), headers=self._headers())
```

```
        return json.loads(req.content)
    def get_nodes(self):
        url = self.host + '/api/v1/assets/nodes/'
        req = requests.get(url,auth=self._auth(),headers=self._headers())
        return json.loads(req.content)
    def create_assets(self,ip,hostname,node):
        url = self.host + '/api/v1/assets/assets/'
        data = {
            'hostname': hostname,
            'ip': ip,
            'platform':'Linux',
            'nodes': node,
            "admin_user_display": "limi_admin",
            "protocols": ["ssh/5203"],
            "created_by": "Administrator",
            "admin_user": "ddd041bc-91b2-4fc6-9887-a38c5d445080",
            "is_active": 'true',
        }
        req = requests.post(url,auth=self._auth(),headers=self._headers(),data=data)
        return json.loads(req.content)

    def delete_assets(self,id):
        # /assets/nodes/{id}/
        url = Jumpserver_url + '/api/v1/assets/assets/{}/'.format(id)
        req = requests.delete(url, auth=self._auth(), headers=self._headers())
        return req.content.decode('utf-8')

def send_msg(text):
    headers = {'Content-Type': 'application/json;charset=utf-8'}
    api_url = "https://oapi.dingtalk.com/robot/send?access_token=access_token"
    json_text= {
        "actionCard": {
            "title": "Jumpserver同步资产通知",
            "text":
             text,
            "hideAvatar": "0",
            "btnOrientation": "0",
            "btns": [
                {
                    "title": "Jumpserver链接",
                    "actionURL": "https://opt-jumpserver.jumpserver.com"
                },
            ]
        },
        "msgtype": "actionCard"
    }
    Text = requests.post(api_url,data=json.dumps(json_text),headers=headers).json()
    return Text
if __name__ == '__main__':
    #本地机房服务器白名单
    white_list_ip = ['192.168.51.200','192.168.51.201','192.168.51.202','192.168.51.203','192.168.
1.204','192.168.51.205','192.168.51.206',
                     '192.168.51.207','192.168.51.208','192.168.51.209','192.168.51.210','192.168.51.21
','192.168.51.212','192.168.51.213',
```

```
                '192.168.51.214',]
    #ali全部资产写入列表
    aliyun_all = aliyun_ecs().assets_list()
    aliassets_dict = dict(list(zip(aliyun_ip_list,aliyun_name_list))) #将aliyun资产ip和命名合并成字典

    # #jumpserver全部资产
    Jump_assetsInfo = new_Jumpserver()
    node_dict = {}  #jumpserver node节点
    for node_info in Jump_assetsInfo.get_nodes():
        node_dict[node_info.get('value')] = node_info.get('id')

    for var in Jump_assetsInfo.get_assets():
        JumpIP_list.append(var.get('ip'))
        JumpID_list.append(var.get('id'))
    jumpserver_dict = dict(list(zip(JumpIP_list,JumpID_list))) #将Jumpserver资产合并成子字典
    # 检查Jumpserver是否存在重复资产
    dup = dict(Counter(JumpIP_list))
    dup_set = [key for key,value in dup.items()if value > 1]
    if dup_set:
        message = '**Jumpserver中存在重复资产信息列表**:' + '\n\n' + str(dup_set)
        send_msg(message)
        with open('/alidata/jumpserver/rsync_assets/log', 'a', encoding='utf-8') as dup_assets:
            dup_assets.write(message +'\n')
    if len(aliyun_ip_list) > 400:  #检查是否有阿里云获取到的值，有则执行下一步
        #ali云和Jumpserver资产对比,如果jumpserver有而阿里云以及本地机房没有则删除Jumpserve
资产
        delete_ip_list = [ip for ip in JumpIP_list if ip not in aliyun_ip_list+white_list_ip]
        if len(delete_ip_list) > 0:
            delete_msg = '**Jumpserver删除资产通知**:' + '\n\n' + \
                    '**当前时间**:' + time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()) + '\n
n' + \
                    '**删除ip列表**:' + str(delete_ip_list)
            for ip in delete_ip_list:
                del_assets = new_Jumpserver().delete_assets(id=jumpserver_dict[ip])

            with open('/alidata/jumpserver/rsync_assets/log','a',encoding='utf-8') as f:
                f.write(delete_msg + '\n')
            send_msg(delete_msg)

        #列表推导式--> 同步阿里云资产到Jumpserver
        add_ip_list = [ip for ip in aliyun_ip_list if ip not in JumpIP_list]
        if len(add_ip_list) > 0:
            add_msg = '**Jumpserver添加资产通知**:' + '\n\n' + \
                    '**当前时间**:' + time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()) + '\n\n
 + \
                    '**添加ip列表**:' + str(add_ip_list)

            for ip in add_ip_list:
                try:
                    nodeId = node_dict['{}'.format(aliassets_node[ip])]
                    add_assets = new_Jumpserver().create_assets(ip=ip, hostname=aliassets_dict[ip]
 '-' + ip,
                                                node=nodeId)
```

原文链接：阿里云同步资产信息到 Jumpserver

```
            except KeyError:
                node_id = node_dict['Jumpserver资产同步']
                add_assets = new_Jumpserver().create_assets(ip=ip, hostname=aliassets_dict[ip]
'-' + ip,
                                                            node=node_id)
        send_msg(add_msg)
```

人机交互同步添加的资产

```
#!/usr/bin/env python3
#coding=utf-8
import json
import requests
import time
from httpsig.requests_auth import HTTPSignatureAuth

from aliyunsdkcore.client import AcsClient
from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRe
uest

#新建jumpserver
KEY_ID = 'KEY_ID'
SECRET = 'SECRET'
Jumpserver_url = 'https://opt-jumpserver.limikeji.com'
#aliyun参数
aliyun_ip_list = []
aliyun_name_list = []
JumpIP_list = []
JumpID_list = []
class aliyun_ecs():
    def __init__(self):
        self._client =  AcsClient('<accessKeyId>', '<accessSecret>', 'cn-beijing')

    def page_num(self):
        request = DescribeInstancesRequest()
        request.set_accept_format('json')
        response = json.loads(self._client.do_action_with_exception(request))
        _ecs_num = response.get('TotalCount')//100 + 2
        return _ecs_num

    def assets_list(self):
        request = DescribeInstancesRequest()
        request.set_accept_format('json')
        request.set_PageSize(100)
        for num in range(1,self.page_num()):
            request.set_PageNumber(num)
            response = json.loads(self._client.do_action_with_exception(request))
            instances_list = response.get('Instances').get('Instance')
            for info in instances_list:
                assetsName = info.get('InstanceName')
                aliyun_name_list.append(assetsName)
                assetsIp = ''.join(info.get('VpcAttributes').get('PrivateIpAddress').get('IpAddress'))
                aliyun_ip_list.append(assetsIp)
        return aliyun_ip_list, aliyun_name_list
```

```
class new_Jumpserver():
    def __init__(self,host=Jumpserver_url,keyid=KEY_ID,secret=SECRET):
        self.host = host
        self.keyid = keyid
        self.secret = secret

    def _auth(self):
        signature_headers = ['(request-target)', 'accept', 'date', 'host']
        auth = HTTPSignatureAuth(key_id=self.keyid, secret=self.secret,
                                 algorithm='hmac-sha256',
                                 headers=signature_headers)
        return auth
    def _headers(self):
        headers = {
            'Accept': 'application/json',
            'Date': str(time.strftime("%a %b %d %H:%M:%S %Y", time.localtime()))
        }
        return headers

    def get_assets(self):
        url = self.host + '/api/v1/assets/assets/'
        req = requests.get(url, auth=self._auth(), headers=self._headers())
        return json.loads(req.content)
    def get_nodes(self):
        url = self.host + '/api/v1/assets/nodes/'
        req = requests.get(url,auth=self._auth(),headers=self._headers())
        return json.loads(req.content)
    def create_assets(self,ip,hostname,node):
        url = self.host + '/api/v1/assets/assets/'
        data = {
            'hostname': hostname,
            'ip': ip,
            'platform':'Linux',
            'nodes': node,
            "admin_user_display": "limi_admin",
            "protocols": ["ssh/5203"],
            "created_by": "Administrator",
            "admin_user": self.get_assets()[0].get('admin_user'),
            "is_active": 'true',
        }
        req = requests.post(url,auth=self._auth(),headers=self._headers(),data=data)
        return json.loads(req.content)


def send_msg(text):
    headers = {'Content-Type': 'application/json;charset=utf-8'}
    api_url = "https://oapi.dingtalk.com/robot/send?access_token=access_token"
    json_text= {
        "actionCard": {
            "title": "Jumpserver同步资产通知",
            "text":
             text,
            "hideAvatar": "0",
```

```
            "btnOrientation": "0",
            "btns": [
                {
                    "title": "Jumpserver链接",
                    "actionURL": "http://opt-jumpserver.limikeji.com"
                },
            ]
        },
        "msgtype": "actionCard"
    }
    Text = requests.post(api_url,data=json.dumps(json_text),headers=headers).json()
    return Text

if __name__ == '__main__':

    #ali全部资产写入列表
    aliyun_ecs().assets_list()
    aliassets_dict = dict(list(zip(aliyun_ip_list,aliyun_name_list))) #将aliyun资产ip和命名合并成字典
    #jumpserver全部资产
    Jump_assetsInfo = new_Jumpserver()
    node_list = []
    node_dict = {}
    for node_info in Jump_assetsInfo.get_nodes():
        node_list.append(node_info.get('value'))
        node_dict[node_info.get('value')] = node_info.get('id')
    if 'Jumpserver资产同步' not  in node_list:
        node_id = node_dict['Default']
    else:
        node_id = node_dict['Jumpserver资产同步']
    for num,char in enumerate(node_list):
        print(num,char)
    node_num = int(input('请按照以上node节点名称输入相应的序号:'))
    node_id = node_dict['%s' % node_list[node_num]]
    for var in Jump_assetsInfo.get_assets():
        JumpIP_list.append(var.get('ip'))
        JumpID_list.append(var.get('id'))
    jumpserver_dict = dict(list(zip(JumpIP_list,JumpID_list))) #将Jumpserver资产合并成子字典
    if len(aliyun_ip_list) > 400:  #检查是否有阿里云获取到的值，有则执行下一步
        # 列表推导式--> 同步阿里云资产到Jumpserver
        add_ip_list = [ip for ip in aliyun_ip_list if ip not in JumpIP_list]
        if len(add_ip_list) > 0:
            add_msg = '**Jumpserver添加资产通知**:' + '\n\n' + \
                      '**当前时间**:' + time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()) + '\n\n
 + \
                      '**添加ip列表**:' + str(add_ip_list)
            for ip in add_ip_list:
                add_assets = new_Jumpserver().create_assets(ip=ip, hostname=aliassets_dict[ip] + '-
 + ip,node=node_id)

            with open('./Jumpserver同步阿里云资产_log','a',encoding='utf-8') as f:
                f.write(add_msg + '\n')
            send_msg(add_msg)
```

效果图：

原文链接：阿里云同步资产信息到 Jumpserver