paramiko

paramiko模块，基于SSH用于连接远程服务器并执行相关操作。

*更多案例参考：paramoko源码： https://github.com/paramiko/paramiko*

二、使用

**SSHClient**

用于连接远程服务器并执行基本命令

**基于用户名密码连接：**

1import paramiko



2



3# 创建SSH对象



4 ssh = paramiko.SSHClient()



5# 允许连接不在know\_hosts文件中的主机



* ssh.set\_missing\_host\_key\_policy(paramiko.AutoAddPolicy()) 7# 连接服务器



8 ssh.connect(hostname='c1.salt.com', port=22, username='wupeiqi', password='123')



9



10# 执行命令

11 stdin, stdout, stderr = ssh.exec\_command('ls')



12# 获取命令结果

13 result = stdout.read()



14

15# 关闭连接

1. ssh.close()



1import paramiko



2



3 transport = paramiko.Transport(('hostname', 22))



4 transport.connect(username='wupeiqi', password='123')



5



6 ssh = paramiko.SSHClient()



7 ssh.\_transport = transport



8



9 stdin, stdout, stderr = ssh.exec\_command('df')



10 print stdout.read()



11

1. transport.close()
2. 
3. SSHClient 封装 Transport

**基于公钥密钥连接：**

import paramiko

private\_key = paramiko.RSAKey.from\_private\_key\_file('/home/auto/.ssh/id\_rsa')

* 创建SSH对象

ssh = paramiko.SSHClient()

* 允许连接不在know\_hosts文件中的主机

ssh.set\_missing\_host\_key\_policy(paramiko.AutoAddPolicy())

* 连接服务器

ssh.connect(hostname='c1.salt.com', port=22, username='wupeiqi', key=private\_key)

* 执行命令

stdin, stdout, stderr = ssh.exec\_command('df')

* 获取命令结果

result = stdout.read()

* 关闭连接

ssh.close()

1import paramiko



2



3 private\_key = paramiko.RSAKey.from\_private\_key\_file('/home/auto/.ssh/id\_rsa')



4



5 transport = paramiko.Transport(('hostname', 22))



6 transport.connect(username='wupeiqi', pkey=private\_key)



7



8 ssh = paramiko.SSHClient()



9 ssh.\_transport = transport



10

11 stdin, stdout, stderr = ssh.exec\_command('df')



12

1. transport.close()
2. 
3. SSHClient 封装 Transport

1import paramiko



2from io import StringIO



3



4 key\_str = """-----BEGIN RSA PRIVATE KEY-----



5MIIEpQIBAAKCAQEAq7gLsqYArAFco02/55IgNg0r7NXOtEM3qXpb/dabJ5Uyky/8



6NEHhFiQ7deHIRIuTW5Zb0kD6h6EBbVlUMBmwJrC2oSzySLU1w+ZNfH0PE6W6fans



7H80whhuc/YgP+fjiO+VR/gFcqib8Rll5UfYzf5H8uuOnDeIXGCVgyHQSmt8if1+e



87hn1MVO1Lrm9Fco8ABI7dyv8/ZEwoSfh2C9rGYgA58LT1FkBRkOePbHD43xNfAYC



9tfLvz6LErMnwdOW4sNMEWWAWv1fsTB35PAm5CazfKzmam9n5IQXhmUNcNvmaZtvP



10c4f4g59mdsaWNtNaY96UjOfx83Om86gmdkKcnwIDAQABAoIBAQCnDBGFJuv8aA7A

11ZkBLe+GN815JtOyye7lIS1n2I7En3oImoUWNaJEYwwJ8+LmjxMwDCtAkR0XwbvY+

12c+nsKPEtkjb3sAu6I148RmwWsGncSRqUaJrljOypaW9dS+GO4Ujjz3/lw1lrxSUh

13IqVc0E7kyRW8kP3QCaNBwArYteHreZFFp6XmtKMtXaEA3saJYILxaaXlYkoRi4k8

14S2/K8aw3ZMR4tDCOfB4o47JaeiA/e185RK3A+mLn9xTDhTdZqTQpv17/YRPcgmwz

15zu30fhVXQT/SuI0sO+bzCO4YGoEwoBX718AWhdLJFoFq1B7k2ZEzXTAtjEXQEWm6

1601ndU/jhAasdfasdasdfasdfa3eraszxqwefasdfadasdffsFIfAsjQb4HdkmHuC

17OeJrJOd+CYvdEeqJJNnF6AbHyYHIECkj0Qq1kEfLOEsqzd5nDbtkKBte6M1trbjl

18HtJ2Yb8w6o/q/6Sbj7wf/cW3LIYEdeVCjScozVcQ9R83ea05J+QOAr4nAoGBAMaq

19UzLJfLNWZ5Qosmir2oHStFlZpxspax/ln7DlWLW4wPB4YJalSVovF2Buo8hr8X65

20lnPiE41M+G0Z7icEXiFyDBFDCtzx0x/RmaBokLathrFtI81UCx4gQPLaSVNMlvQA

21539GsubSrO4LpHRNGg/weZ6EqQOXvHvkUkm2bDDJAoGATytFNxen6GtC0ZT3SRQM

22WYfasdf3xbtuykmnluiofasd2sfmjnljkt7khghmghdasSDFGQfgaFoKfaawoYeH

23C2XasVUsVviBn8kPSLSVBPX4JUfQmA6h8HsajeVahxN1U9e0nYJ0sYDQFUMTS2t8

24RT57+WK/0ONwTWHdu+KnaJECgYEAid/ta8LQC3p82iNAZkpWlGDSD2yb/8rH8NQg

259tjEryFwrbMtfX9qn+8srx06B796U3OjifstjJQNmVI0qNlsJpQK8fPwVxRxbJS/

26pMbNICrf3sUa4sZgDOFfkeuSlgACh4cVIozDXlR59Z8Y3CoiW0uObEgvMDIfenAj

2798pl3ZkCgYEAj/UCSni0dwX4pnKNPm6LUgiS7QvIgM3H9piyt8aipQuzBi5LUKWw

28DlQC4Zb73nHgdREtQYYXTu7p27Bl0Gizz1sW2eSgxFU8eTh+ucfVwOXKAXKU5SeI

29+MbuBfUYQ4if2N/BXn47+/ecf3A4KgB37Le5SbLDddwCNxGlBzbpBa0=

30-----END RSA PRIVATE KEY-----"""

31

32 private\_key = paramiko.RSAKey(file\_obj=StringIO(key\_str))

33 transport = paramiko.Transport(('10.0.1.40', 22))

34 transport.connect(username='wupeiqi', pkey=private\_key)

35

36 ssh = paramiko.SSHClient()



37 ssh.\_transport = transport



38

39 stdin, stdout, stderr = ssh.exec\_command('df')



40 result = stdout.read()



41

1. transport.close()
2. 

def run(self): self.connect()

self.upload('/home/wupeiqi/tttttttttttt.py')

self.rename('/home/wupeiqi/tttttttttttt.py', '/home/wupeiqi/ooooooooo.py) self.close()

44print(result)

45

1. 基于私钥字符进行连接



**SFTPClient**

用于连接远程服务器并执行上传下载

**基于用户名密码上传下载：**

1import paramiko



2



3 transport = paramiko.Transport(('hostname',22))



* transport.connect(username='wupeiqi',password='123') 5

6 sftp = paramiko.SFTPClient.from\_transport(transport)

7# 将location.py 上传至服务器 /tmp/test.py

8 sftp.put('/tmp/location.py', '/tmp/test.py') 9# 将remove\_path 下载到本地 local\_path



10 sftp.get('remove\_path', 'local\_path')



11

1. transport.close()

**基于公钥密钥上传下载：**

1import paramiko



2



3 private\_key = paramiko.RSAKey.from\_private\_key\_file('/home/auto/.ssh/id\_rsa')



4



5 transport = paramiko.Transport(('hostname', 22))



6 transport.connect(username='wupeiqi', pkey=private\_key )



7



8 sftp = paramiko.SFTPClient.from\_transport(transport)



9# 将location.py 上传至服务器 /tmp/test.py



10 sftp.put('/tmp/location.py', '/tmp/test.py')



11# 将remove\_path 下载到本地 local\_path

12 sftp.get('remove\_path', 'local\_path')



13

1. transport.close()

1#!/usr/bin/env python



2 # -\*- coding:utf-8 -\*-



3 import paramiko



4import uuid



5



6class SSHConnection(object):



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 7 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8 | def | \_\_init\_\_ | | (self, host= | | | | | '172.16.103.191' | | | | | , port=22, username= | 'wupeiqi' | ,pwd= | '123' | ): |  |
|  |  |  |  |  | |  |  |  |  | |  |  |  |  | | | | | | |  |
| 9 |  | self.host = host | | | | |  |
|  |  |  |  |  | |  |  |  |  | |  |  |  |  | | | | | | |  |
| 10 |  | self.port = port | | | | |  |
|  |  |  |  |  | |  |  | |  | |  |  |  |  | | | | | | |  |
| 11 |  | self.username = username | | | | | | | | |  |
|  |  |  |  |  | |  |  | |  | |  | |  |  | | | | | | |  |
| 12 |  | self.pwd = pwd | | |  |
|  |  |  |  |  |  | |  | |  | |  | |  |  | | | | | | |  |
| 13 |  | self. | \_\_k | = None | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 14 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | | | | | |  | |  | |  |  | | | | | | |  |
| 15 | def | create\_file(self): | | | | | |  |
|  |  |  |  | | | | | | | |  | |  |  | | | | | | |  |
| 16 |  | file\_name = str(uuid.uuid4()) | | | | | | | | | |  |
|  |  |  |  | | | | | | |  |  | | |  | | | | | | |  |
| 17 |  | with open(file\_name, | | | | | | | 'w' | ) as f: | | |  |



1. f.write('sb')
2. return file\_name
3. 
4. 
5. 
6. 
7. 



25

26

27def connect(self):

1. transport = paramiko.Transport((self.host,self.port))
2. transport.connect(username=self.username,password=self.pwd)
3. self.\_\_transport = transport



31

32def close(self):

33

1. self.\_\_transport.close()
2. 
3. def upload(self,target\_path):
4. # 连接，上传



1. file\_name = self.create\_file()
2. 
3. sftp = paramiko.SFTPClient.from\_transport(self.\_\_transport)
4. # 将location.py 上传至服务器 /tmp/test.py
5. sftp.put(file\_name, target\_path)
6. 
7. def rename(self, old\_path, new\_path):
8. 
9. ssh = paramiko.SSHClient()
10. ssh.\_transport = self.\_\_transport
11. # 执行命令



1. cmd = "mv %s %s" % (old\_path, new\_path,)
2. stdin, stdout, stderr = ssh.exec\_command(cmd)
3. # 获取命令结果



1. result = stdout.read()
2. 
3. def cmd(self, command):
4. ssh = paramiko.SSHClient()
5. ssh.\_transport = self.\_\_transport
6. # 执行命令



1. stdin, stdout, stderr = ssh.exec\_command(command)
2. # 获取命令结果



1. result = stdout.read()
2. return result
3. 
4. 
5. 
6. ha = SSHConnection()
7. ha.run()
8. 
9. Demo



1 # 对于更多限制命令，需要在系统中设置



* /etc/sudoers 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 4 | Defaults | requiretty | |  |  |  |
|  |  |  | |  | |  |  |
| 5 | Defaults:cmdb | | !requiretty | |  |

1import paramiko



2import uuid



3



4class SSHConnection(object):



5



6def\_\_init\_\_(self, host='192.168.11.61', port=22, username='alex',pwd='alex3714'):



* self.host = host

8self.port = port

9self.username = username



1. self.pwd = pwd
2. self.\_\_k = None



12

13def run(self):

1. self.connect()
2. pass



1. self.close()
2. 
3. def connect(self):
4. transport = paramiko.Transport((self.host,self.port))



1. transport.connect(username=self.username,password=self.pwd)
2. self.\_\_transport = transport



22

23def close(self):

1. self.\_\_transport.close()
2. 
3. def cmd(self, command):
4. ssh = paramiko.SSHClient()
5. ssh.\_transport = self.\_\_transport
6. # 执行命令



1. stdin, stdout, stderr = ssh.exec\_command(command)
2. # 获取命令结果



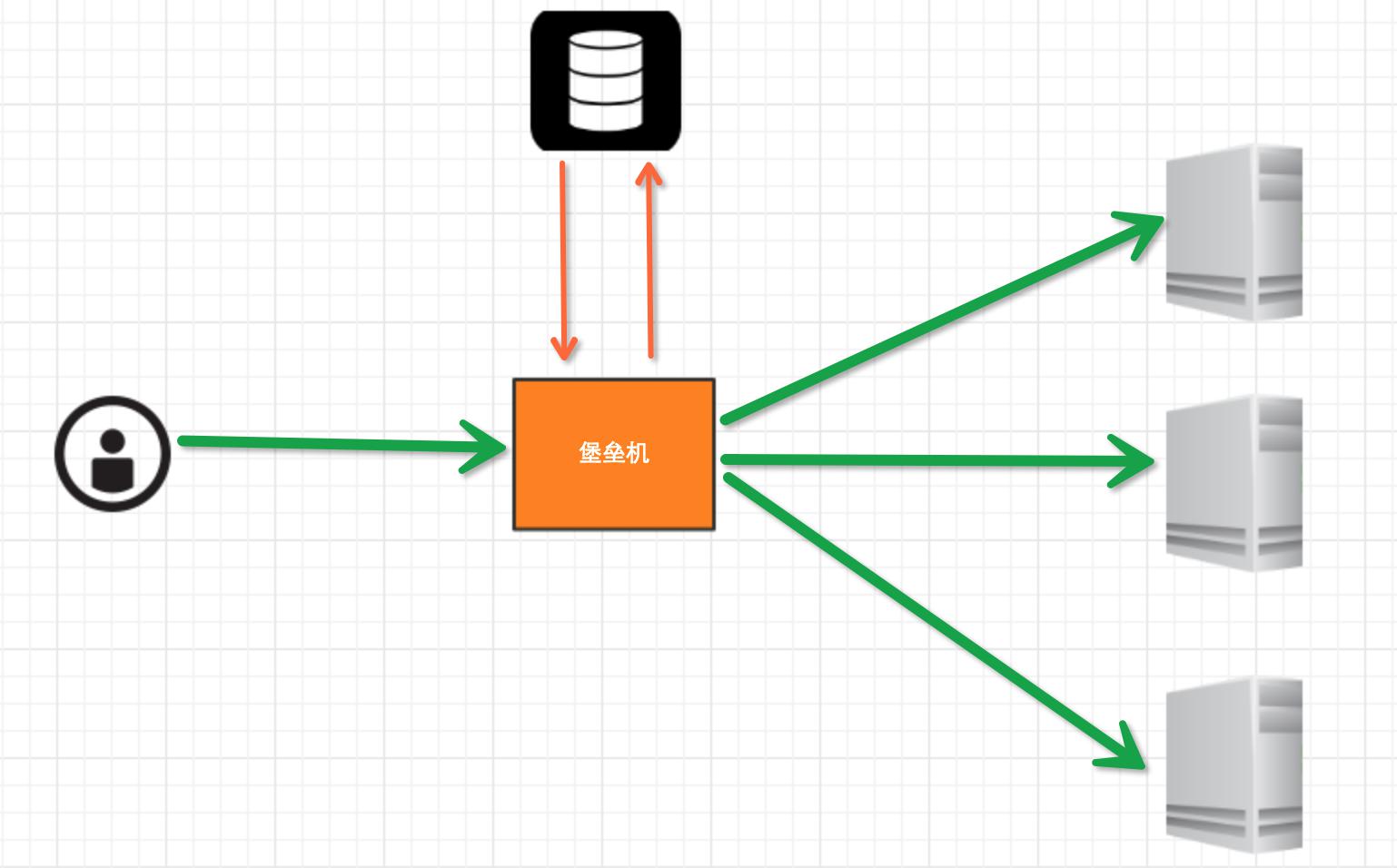
1. result = stdout.read()
2. return result
3. 
4. def upload(self,local\_path, target\_path):
5. # 连接，上传



1. sftp = paramiko.SFTPClient.from\_transport(self.\_\_transport)
2. # 将location.py 上传至服务器 /tmp/test.py
3. sftp.put(local\_path, target\_path)
4. 
5. ssh = SSHConnection()
6. ssh.connect()
7. r1 = ssh.cmd('df')
8. ssh.upload('s2.py', "/home/alex/s7.py")
9. ssh.close()
10. 
11. Demo



**堡垒机**



堡垒机执行流程：

1. 管理员为用户在服务器上创建账号（将公钥放置服务器，或者使用用户名密码）
2. 用户登陆堡垒机，输入堡垒机用户名密码，现实当前用户管理的服务器列表
3. 用户选择服务器，并自动登陆
4. 执行操作并同时将用户操作记录

*注：配置.brashrc实现ssh登陆后自动执行脚本，如：/usr/bin/python /home/wupeiqi/menu.py*

1#!/usr/bin/env python



2# -\*- coding:utf-8 -\*-



3



4from sqlalchemy import create\_engine, and\_, or\_, func, Table



5 from sqlalchemy.ext.declarative import declarative\_base



6from sqlalchemy import Column, Integer, String, ForeignKey, UniqueConstraint, DateTime



7 from sqlalchemy.orm import sessionmaker, relationship



8



9 Base = declarative\_base() # 生成一个SqlORM 基类



10

11

12class Host(Base):

13\_\_tablename\_\_ = 'host'



1. id = Column(Integer, primary\_key=True, autoincrement=True)
2. hostname = Column(String(64), unique=True, nullable=False)
3. ip\_addr = Column(String(128), unique=True, nullable=False)
4. port = Column(Integer, default=22)



18

19

20class HostUser(Base):

21\_\_tablename\_\_ = 'host\_user'



1. id = Column(Integer, primary\_key=True, autoincrement=True)
2. username = Column(String(64), unique=True, nullable=False)
3. AuthTypes = [
4. ('p', 'SSH/Password'),
5. ('r', 'SSH/KEY'),
6. ]
7. auth\_type = Column(String(16))
8. cert = Column(String(255))



30

1. host\_id = Column(Integer, ForeignKey('host.id'))
2. 
3. \_\_table\_args\_\_ = (



1. UniqueConstraint('host\_id', 'username', name='\_host\_username\_uc'),
2. )



36

37

38class Group(Base):

39\_\_tablename\_\_ = 'group'



1. id = Column(Integer, primary\_key=True, autoincrement=True)
2. name = Column(String(64), unique=True, nullable=False)



42

43

44class UserProfile(Base):

45\_\_tablename\_\_ = 'user\_profile'



1. id = Column(Integer, primary\_key=True, autoincrement=True)
2. username = Column(String(64), unique=True, nullable=False)
3. password = Column(String(255), nullable=False)



49

50

51class Group2UserProfile(Base):

52\_\_tablename\_\_ = 'group\_2\_user\_profile'



1. id = Column(Integer, primary\_key=True, autoincrement=True)
2. user\_profile\_id = Column(Integer, ForeignKey('user\_profile.id'))
3. group\_id = Column(Integer, ForeignKey('group.id'))



56 \_\_table\_args\_\_ = (



1. UniqueConstraint('user\_profile\_id', 'group\_id', name='ux\_user\_group'),
2. )



59

60

61class Group2HostUser(Base):

62\_\_tablename\_\_ = 'group\_2\_host\_user'



1. id = Column(Integer, primary\_key=True, autoincrement=True)
2. host\_user\_id = Column(Integer, ForeignKey('host\_user.id'))
3. group\_id = Column(Integer, ForeignKey('group.id'))



66\_\_table\_args\_\_ = (



1. UniqueConstraint('group\_id', 'host\_user\_id', name='ux\_group\_host\_user'),



1. )
2. 
3. 
4. class UserProfile2HostUser(Base):
5. \_\_tablename\_\_ = 'user\_profile\_2\_host\_user'
6. id = Column(Integer, primary\_key=True, autoincrement=True)
7. host\_user\_id = Column(Integer, ForeignKey('host\_user.id'))
8. user\_profile\_id = Column(Integer, ForeignKey('user\_profile.id'))
9. \_\_table\_args\_\_ = (



1. UniqueConstraint('user\_profile\_id', 'host\_user\_id', name='ux\_user\_host\_user'),
2. )



79

80

81class AuditLog(Base):

82\_\_tablename\_\_ = 'audit\_log'



1. id = Column(Integer, primary\_key=True, autoincrement=True)
2. 
3. action\_choices2 = [



1. (u'cmd', u'CMD'),
2. (u'login', u'Login'),
3. (u'logout', u'Logout'),
4. ]
5. action\_type = Column(String(16))
6. cmd = Column(String(255))
7. date = Column(DateTime)
8. user\_profile\_id = Column(Integer, ForeignKey('user\_profile.id'))
9. host\_user\_id = Column(Integer, ForeignKey('host\_user.id'))



95

1. 表结构示例



**实现过程**

1、前戏

1import paramiko



2import sys



3import os



4import socket



5import select



6import getpass



7



8 tran = paramiko.Transport(('10.211.55.4', 22,))



* tran.start\_client()



10 tran.auth\_password('wupeiqi', '123')



11

12# 打开一个通道

13 chan = tran.open\_session()



14# 获取一个终端

1. chan.get\_pty()
2. # 激活器



1. chan.invoke\_shell()
2. #########
3. # 利用sys.stdin,肆意妄为执行操作



21# 用户在终端输入内容，并将内容发送至远程服务器

22# 远程服务器执行命令，并将结果返回

23# 用户终端显示内容

24#########

25

1. chan.close()
2. tran.close()



2、肆意妄为（一）

1import paramiko



2import sys



3import os



4import socket



5import select



6import getpass



7from paramiko.py3compat import u



8



9 tran = paramiko.Transport(('10.211.55.4', 22,))



1. tran.start\_client()
2. tran.auth\_password('wupeiqi', '123')
3. # 打开一个通道



14 chan = tran.open\_session()



15# 获取一个终端

1. chan.get\_pty()
2. # 激活器



1. chan.invoke\_shell()
2. while True:
3. # 监视用户输入和服务器返回数据



22# sys.stdin 处理用户输入

23# chan 是之前创建的通道，用于接收服务器返回信息

1. readable, writeable, error = select.select([chan, sys.stdin, ],[],[],1)
2. if chan in readable:
3. try:



1. x = u(chan.recv(1024))
2. if len(x) == 0:
3. print('\r\n\*\*\* EOF\r\n')
4. break



1. sys.stdout.write(x)
2. sys.stdout.flush()
3. except socket.timeout:
4. pass
5. if sys.stdin in readable:
6. inp = sys.stdin.readline()



1. chan.sendall(inp)
2. chan.close()
3. tran.close()



1#!/usr/bin/env python



2# -\*- coding:utf-8 -\*-



3



4import paramiko



5import sys



6import os



7import socket



8import select



9import getpass



10from paramiko.py3compat import u



11

12

13 default\_username = getpass.getuser()

14 username = input('Username [%s]: ' % default\_username)



15if len(username) == 0:

1. username = default\_username
2. 
3. 
4. hostname = input('Hostname: ')
5. if len(hostname) == 0:
6. print('\*\*\* Hostname required.')
7. sys.exit(1)
8. 
9. tran = paramiko.Transport((hostname, 22,))



1. tran.start\_client()
2. 
3. default\_auth = "p"
4. auth = input('Auth by (p)assword or (r)sa key[%s] ' % default\_auth)
5. if len(auth) == 0:
6. auth = default\_auth
7. 
8. if auth == 'r':
9. default\_path = os.path.join(os.environ['HOME'], '.ssh', 'id\_rsa')
10. path = input('RSA key [%s]: ' % default\_path)
11. if len(path) == 0:
12. path = default\_path
13. try:
14. key = paramiko.RSAKey.from\_private\_key\_file(path)
15. except paramiko.PasswordRequiredException:
16. password = getpass.getpass('RSA key password: ')
17. key = paramiko.RSAKey.from\_private\_key\_file(path, password)
18. tran.auth\_publickey(username, key)
19. else:
20. pw = getpass.getpass('Password for %s@%s: ' % (username, hostname))
21. tran.auth\_password(username, pw)
22. 
23. 
24. 
25. # 打开一个通道



50 chan = tran.open\_session()



51# 获取一个终端

1. chan.get\_pty()
2. # 激活器



1. chan.invoke\_shell()
2. 
3. while True:
4. # 监视用户输入和服务器返回数据



58# sys.stdin 处理用户输入

59# chan 是之前创建的通道，用于接收服务器返回信息

1. readable, writeable, error = select.select([chan, sys.stdin, ],[],[],1)
2. if chan in readable:
3. try:



1. x = u(chan.recv(1024))
2. if len(x) == 0:
3. print('\r\n\*\*\* EOF\r\n')
4. break



1. sys.stdout.write(x)
2. sys.stdout.flush()
3. except socket.timeout:
4. pass
5. if sys.stdin in readable:
6. inp = sys.stdin.readline()



1. chan.sendall(inp)
2. 
3. chan.close()
4. tran.close()
5. 
6. 完整示例（一）



3、肆意妄为（二）

1import paramiko



2import sys



3import os



4import socket



5import select



6import getpass



7import termios



8import tty



9from paramiko.py3compat import u



10

11 tran = paramiko.Transport(('10.211.55.4', 22,))



1. tran.start\_client()
2. tran.auth\_password('wupeiqi', '123')
3. # 打开一个通道



16 chan = tran.open\_session()



17# 获取一个终端

1. chan.get\_pty()
2. # 激活器



1. chan.invoke\_shell()
2. # 获取原tty属性



24 oldtty = termios.tcgetattr(sys.stdin)



25try:

26# 为tty设置新属性

27# 默认当前tty设备属性：

28# 输入一行回车，执行

29 # CTRL+C 进程退出，遇到特殊字符，特殊处理。



30

31# 这是为原始模式，不认识所有特殊符号

32# 放置特殊字符应用在当前终端，如此设置，将所有的用户输入均发送到远程服务器

1. tty.setraw(sys.stdin.fileno())
2. chan.settimeout(0.0)



35

36while True:



37# 监视 用户输入 和 远程服务器返回数据（socket）

38 # 阻塞，直到句柄可读



1. r, w, e = select.select([chan, sys.stdin], [], [], 1)
2. if chan in r:
3. try:



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 42 |  | x = u(chan.recv(1024)) | | | | | | | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |
| 43 |  | if | len(x) == 0: | | | | | |  |
|  |  |  |  |  |  |  |  |  |  | |  |  | |  |  |  |
| 44 |  |  |  | print | | | ( | '\r\n\*\*\* EOF\r\n' | | | | | ) |  |
|  |  |  |  |  |  |  |  |  | | |  |  | | | |  |
| 45 |  |  |  | break | | |  |
|  |  |  |  | | |  |  | | | |  |  | | | |  |
| 46 |  | sys.stdout.write(x) | | | | | | | | |  |
|  |  |  |  | | |  |  | | | |  | | | | |  |
| 47 |  | sys.stdout.flush() | | | | | | | |  |
|  |  |  |  |  | |  |  | | | |  | | | | |  |
| 48 |  | except | socket.timeout: | | | | | | |  |
|  |  |  |  | |  |  |  | | | | | | | | |  |
| 49 |  | pass | |  |
|  |  |  |  | | |  |  | |  | | | | | | |  |
| 50 | if | sys.stdin | | | in | r: | |  |

1. x = sys.stdin.read(1)
2. if len(x) == 0:
3. break



1. chan.send(x)
2. finally:
3. # 重新设置终端属性



1. termios.tcsetattr(sys.stdin, termios.TCSADRAIN, oldtty)
2. chan.close()
3. tran.close()



1#!/usr/bin/env python



2 # -\*- coding:utf-8 -\*-



3 import paramiko



4import sys



5import os



6import socket



7import getpass



8import termios



9import tty



10import select



11from paramiko.py3compat import u



12

13

14def interactive\_shell(chan):

15# 获取原tty属性

1. oldtty = termios.tcgetattr(sys.stdin)
2. try:
3. # 为tty设置新属性



19# 默认当前tty设备属性：

20# 输入一行回车，执行

21# CTRL+C 进程退出，遇到特殊字符，特殊处理。

22

23 # 这是为原始模式，不认识所有特殊符号



24# 放置特殊字符应用在当前终端，如此设置，将所有的用户输入均发送到远程服务器

1. tty.setraw(sys.stdin.fileno())
2. tty.setcbreak(sys.stdin.fileno())
3. chan.settimeout(0.0)



28while True:



1. r, w, e = select.select([chan, sys.stdin], [], [])
2. if chan in r:
3. try:



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 32 |  |  |  |  | x = u(chan.recv(1024)) | | | | | | | | | | | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |
| 33 |  |  |  |  | if | len(x) == 0: | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  |  |  |
| 34 |  |  |  |  |  |  |  | sys.stdout.write( | | | | | | | | | | '\r\n\*\*\* EOF\r\n' | | ) |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  | | | | |  |
| 35 |  |  |  |  |  |  |  | break | | |  |
|  |  |  |  |  |  |  | |  | |  |  | |  |  | |  |  |  | | | | |  |
| 36 |  |  |  |  | sys.stdout.write(x) | | | | | | | | | | | |  |
|  |  |  |  |  |  |  | |  | |  |  | |  |  | |  |  | | | | | |  |
| 37 |  |  |  |  | sys.stdout.flush() | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  | |  |  | |  |  | |  |  | | | | | |  |
| 38 |  |  |  | except | | socket.timeout: | | | | | | | | | |  |
|  |  |  |  |  |  |  | |  |  |  |  | |  |  | |  | | | | | | |  |
| 39 |  |  |  |  | pass | | |  |
|  |  |  |  |  |  |  | |  | |  |  | |  |  | |  | | | | | | |  |
| 40 |  |  | if | sys.stdin | | | | | in | r: | |  |
|  |  |  |  |  |  |  | |  | | |  | | |  | |  | | | | | | |  |
| 41 |  |  |  | x = sys.stdin.read(1) | | | | | | | | | | |  |
|  |  |  |  |  |  |  | |  | | |  | | |  | | | | | | | | |  |
| 42 |  |  |  | if | len(x) == 0: | | | | | | | |  |
|  |  |  |  |  |  |  | |  | |  |  | | | | | | | | | | | |  |
| 43 |  |  |  |  | break | | | |  |
|  |  |  |  |  |  | | |  | | |  | | | | | | | | | | | |  |
| 44 |  |  |  | chan.send(x) | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 45 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | | | |  | | | | | | | | | | | | | | |  |
| 46 | finally | : |  |
|  |  |  | | | | | |  | | | | | | | | | | | | | | |  |
| 47 | # 重新设置终端属性 | | | | | |  |

1. termios.tcsetattr(sys.stdin, termios.TCSADRAIN, oldtty)
2. 
3. 
4. def run():
5. hostname = input('请输入主机名: ')
6. tran = paramiko.Transport((hostname, 22,))
7. tran.start\_client()
8. 
9. username = input('请输入用户名: ')
10. auth = input('请输入密码进行验证(p) 或 (r)sa Key进行验证?')
11. if auth == 'r':
12. path = input('请输入RSA key 路径: ')
13. try:



1. key = paramiko.RSAKey.from\_private\_key\_file(path)
2. except paramiko.PasswordRequiredException:



1. password = getpass.getpass('RSA key password: ')
2. key = paramiko.RSAKey.from\_private\_key\_file(path, password)
3. tran.auth\_publickey(username, key)



66else:

1. pw = getpass.getpass('请输入密码 %s@%s: ' % (username, hostname))
2. tran.auth\_password(username, pw)



69

70# 打开一个通道

1. chan = tran.open\_session()
2. # 获取一个终端



1. chan.get\_pty()
2. # 激活器



1. chan.invoke\_shell()
2. 
3. interactive\_shell(chan)
4. 
5. chan.close()
6. tran.close()
7. 
8. 
9. if\_\_name\_\_ == '\_\_main\_\_':
10. run()
11. 
12. 基于Passwd或者RSA进行登陆操作



1#!/usr/bin/env python



2 # -\*- coding:utf-8 -\*-



3 import paramiko



4import sys



5import os



6import socket



7import getpass



8import termios



9import tty



10import select



11from paramiko.py3compat import u



12

13

14def interactive\_shell(chan):

15# 获取原tty属性

1. oldtty = termios.tcgetattr(sys.stdin)
2. try:
3. # 为tty设置新属性



19# 默认当前tty设备属性：

20# 输入一行回车，执行

21# CTRL+C 进程退出，遇到特殊字符，特殊处理。

22

23 # 这是为原始模式，不认识所有特殊符号



24 # 放置特殊字符应用在当前终端，如此设置，将所有的用户输入均发送到远程服务器



1. tty.setraw(sys.stdin.fileno())
2. tty.setcbreak(sys.stdin.fileno())
3. chan.settimeout(0.0)



28while True:



1. r, w, e = select.select([chan, sys.stdin], [], [])
2. if chan in r:
3. try:



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 32 |  |  |  |  | x = u(chan.recv(1024)) | | | | | | | | | | | | | |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |
| 33 |  |  |  |  | if | len(x) == 0: | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  |  |  |
| 34 |  |  |  |  |  |  |  | sys.stdout.write( | | | | | | | | | | '\r\n\*\*\* EOF\r\n' | | ) |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  | | | | |  |
| 35 |  |  |  |  |  |  |  | break | | |  |
|  |  |  |  |  |  |  | |  | |  |  | |  |  | |  |  |  | | | | |  |
| 36 |  |  |  |  | sys.stdout.write(x) | | | | | | | | | | | |  |
|  |  |  |  |  |  |  | |  | |  |  | |  |  | |  |  | | | | | |  |
| 37 |  |  |  |  | sys.stdout.flush() | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  | |  |  | |  |  | |  |  | | | | | |  |
| 38 |  |  |  | except | | socket.timeout: | | | | | | | | | |  |
|  |  |  |  |  |  |  | |  |  |  |  | |  |  | |  | | | | | | |  |
| 39 |  |  |  |  | pass | | |  |
|  |  |  |  |  |  |  | |  | |  |  | |  |  | |  | | | | | | |  |
| 40 |  |  | if | sys.stdin | | | | | in | r: | |  |
|  |  |  |  |  |  |  | |  | | |  | | |  | |  | | | | | | |  |
| 41 |  |  |  | x = sys.stdin.read(1) | | | | | | | | | | |  |
|  |  |  |  |  |  |  | |  | | |  | | |  | | | | | | | | |  |
| 42 |  |  |  | if | len(x) == 0: | | | | | | | |  |
|  |  |  |  |  |  |  | |  | |  |  | | | | | | | | | | | |  |
| 43 |  |  |  |  | break | | | |  |
|  |  |  |  |  |  | | |  | | |  | | | | | | | | | | | |  |
| 44 |  |  |  | chan.send(x) | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 45 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | | | |  | | | | | | | | | | | | | | |  |
| 46 | finally | : |  |
|  |  |  | | | | | |  | | | | | | | | | | | | | | |  |
| 47 | # 重新设置终端属性 | | | | | |  |

1. termios.tcsetattr(sys.stdin, termios.TCSADRAIN, oldtty)



49

50

51 def run():



1. db\_dict = {



53'c1.salt.com': {

54'root': {'user': 'root', 'auth': 'r', "cert": 'key路径'},

55'alex': {'user': 'alex', 'auth': 'p', "cert": '密码'},

1. },



57'c2.salt.com': {

58'alex': {'user': 'alex', 'auth': 'p', "cert": '密码'},

1. },
2. 
3. }
4. 
5. for row in db\_dict.keys():
6. print(row)
7. 
8. hostname = input('请选择主机: ')
9. tran = paramiko.Transport((hostname, 22,))
10. tran.start\_client()
11. 
12. for item in db\_dict[hostname].keys():
13. print(item)
14. 
15. username = input('请输入用户: ')
16. 
17. user\_dict = db\_dict[hostname][username]
18. if username['auth'] == 'r':
19. key = paramiko.RSAKey.from\_private\_key\_file(user\_dict['cert'])
20. tran.auth\_publickey(username, key)
21. else:
22. pw = user\_dict['cert']
23. tran.auth\_password(username, pw)
24. 
25. # 打开一个通道



1. chan = tran.open\_session()
2. # 获取一个终端



1. chan.get\_pty()
2. # 激活器



1. chan.invoke\_shell()
2. 
3. interactive\_shell(chan)
4. 
5. chan.close()
6. tran.close()
7. 
8. 
9. if\_\_name\_\_ == '\_\_main\_\_':
10. run()
11. 
12. 提示用户选择主机和用户



1#!/usr/bin/env python

2# -\*- coding:utf-8 -\*-

3 import paramiko

4import sys



5import os



6import socket



7import getpass

8import termios

9import tty



10import select



11from paramiko.py3compat import u



12



13



14def interactive\_shell(chan):



15# 获取原tty属性



1. oldtty = termios.tcgetattr(sys.stdin)
2. try:
3. # 为tty设置新属性



19# 默认当前tty设备属性：



20# 输入一行回车，执行



21# CTRL+C 进程退出，遇到特殊字符，特殊处理。



22



23# 这是为原始模式，不认识所有特殊符号



24# 放置特殊字符应用在当前终端，如此设置，将所有的用户输入均发送到远程服务器



1. tty.setraw(sys.stdin.fileno())
2. tty.setcbreak(sys.stdin.fileno())
3. chan.settimeout(0.0)



28



1. log = open('handle.log', 'a+', encoding='utf-8')
2. flag = False
3. temp\_list = []



32



33while True:



1. r, w, e = select.select([chan, sys.stdin], [], [])
2. if chan in r:
3. try:



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 37 |  |  |  |  | x = u(chan.recv(1024)) | | | | | | | | | | | | | | | | | | | | |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |
| 38 |  |  |  |  |  | if | | len(x) == 0: | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |
| 39 |  |  |  |  |  |  |  |  | sys.stdout.write( | | | | | | | | | | | | | | | | '\r\n\*\*\* EOF\r\n' | | | | | | | ) |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  | | |  |  |  |  |  | | |  |  |
| 40 |  |  |  |  |  |  |  |  |  | break | | | |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  | |  |  |  |  | |  |  |  |  |  | | |  |  |  |  |  | | |  |  |
| 41 |  |  |  |  |  | # 如果用户上一次点击的是tab键，则获取返回的内容写入在记录中 | | | | | | | | | | | | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  | |  |  |  | |  |  |  |  | |  |  |  |  |  | | |  |  |  |  |  | | | |  |
| 42 |  |  |  |  |  | if | | flag: | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  | | |  |  |  |  | |  |  |  |  |  | | |  |  |  |  |  | | | |  |
| 43 |  |  |  |  |  |  |  |  |  | if | | x.startswith( | | | | | | | | | | | | '\r\n' | | | ): | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  |  |  |  | |  |  |  |  | | | | |  | |  |  | | | |  |
| 44 |  |  |  |  |  |  |  |  |  |  |  |  | pass | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | |  | |  |  |  |  | | | | |  | |  |  | | | |  |
| 45 |  |  |  |  |  |  |  |  |  | else | | | : |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  | | |  |  | |  | |  |  |  |  | | | | |  | |  |  | | | |  |
| 46 |  |  |  |  |  |  |  |  |  |  |  | temp\_list.append(x) | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  | |  |  | | |  |  | |  | |  |  |  |  | | | | | | |  |  | | | |  |
| 47 |  |  |  |  |  |  |  |  | flag = False | | | | | | | | | | |  |
|  |  |  |  |  |  | |  | |  | |  |  | | |  |  | |  | | |  |  |  | | | | | | |  |  | | | |  |
| 48 |  |  |  |  | sys.stdout.write(x) | | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | |  | |  | |  |  | | |  |  | |  | | |  |  | | | | | | | |  |  | | | |  |
| 49 |  |  |  |  | sys.stdout.flush() | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | |  |  |  | |  |  | | |  |  | |  | | |  |  | | | | | | | |  |  | | | |  |
| 50 |  |  |  | except | | | | socket.timeout: | | | | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  | |  |  |  |  | | |  |  | |  | | |  | | | | | | | | |  |  | | | |  |
| 51 |  |  |  |  |  | pass | | | |  |
|  |  |  |  |  |  | |  | |  | |  |  | | |  |  | |  | | |  | | | | | | | | |  |  | | | |  |
| 52 |  |  | if | sys.stdin | | | | | | | in | r: | | |  |
|  |  |  |  |  |  | |  | |  | | | | | | |  | |  | | |  | | | | |  | | | |  |  | | | |  |
| 53 |  |  |  | # 读取用户在终端数据每一个字符 | | | | | | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | |  | |  | | | | | | |  | |  | | |  | | | | | | | | |  |  | | | |  |
| 54 |  |  |  | x = sys.stdin.read(1) | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | |  | |  | | | | | | |  | |  | | | | | | | | | | | |  |  | | | |  |
| 55 |  |  |  | if | len(x) == 0: | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  | |  | |  | | | | | | |  | | | | | | | | | | | |  |  | | | |  |
| 56 |  |  |  |  |  | break | | | | |  |
|  |  |  |  |  | | |  | |  | | | | | | | | |  | | | | | | | | | | | |  |  | | | |  |
| 57 |  |  |  | # 如果用户点击TAB键 | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | |  | |  |  | | |  |  | | | | | | | | | | | | | | | |  |  | | | |  |
| 58 |  |  |  | if | x == | | | | | '\t' | | | : |  |
|  |  |  |  |  |  | |  | |  | | | | | | |  | | | | | | | | | | | | | |  |  | | | |  |
| 59 |  |  |  |  | flag = True | | | | | | | | | | |  |
|  |  |  |  |  | |  |  | |  | | | | | | | | | | | | | | | | | | | | |  |  | | | |  |
| 60 |  |  |  | else | | : |  |
|  |  |  |  |  |  |  | | |  | | | | | | | | | | | | | | | | | | | | |  |  | | | |  |
| 61 |  |  |  |  |  | # 未点击TAB键，则将每个操作字符记录添加到列表中，以便之后写入文件 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | | | |  | | | | | | | | | | | | | |  | | | | | | |  |  | | | |  |
| 62 |  |  |  |  | temp\_list.append(x) | | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 63 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  | | | | |  | | | | | | | | | | | | | | | | | | | | |  |  | | | |  |
| 64 |  |  |  | # 如果用户敲回车，则将操作记录写入文件 | | | | | | | | | | | | | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | | | |  |  | | |  |  | | | | | | | | | | | | | | | | |  | | | |  |
| 65 |  |  |  | if | x == | | | | | '\r' | | | : |  |
|  |  |  |  |  |  | | | |  | | | | | |  | |  | | | | | | | | | | | | | |  | | | |  |
| 66 |  |  |  |  | log.write( | | | | | | | | | | '' | | .join(temp\_list)) | | | | | | | | | | | | | |  |
|  |  |  |  |  |  | | | |  | | | | | | |  | | | | | | | | | | | | | | | | | | |  |
| 67 |  |  |  |  | log.flush() | | | | | | | | | | |  |
|  |  |  |  |  |  | | | |  | | | | | | | | | | | |  | | | | | | | | | | | | | |  |
| 68 |  |  |  |  | temp\_list.clear() | | | | | | | | | | | | | | | |  |
|  |  |  |  |  | | | | |  | | | |  | | | | | | | | | | | | | | | | | | | | | |  |
| 69 |  |  |  | chan.send(x) | | | | | | | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 70 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | | | | | |  | | | | | | | | | | | | | | | | | | | | | | | | | |  |
| 71 | finally | : |  |
|  |  | | | | | | | |  | | | | | | | | | | | | | | | | | | | | | | | | | |  |
| 72 | # 重新设置终端属性 | | | | | | | |  |

1. termios.tcsetattr(sys.stdin, termios.TCSADRAIN, oldtty)
2. 
3. 
4. def run():
5. db\_dict = {



78'c1.salt.com': {



79'root': {'user': 'root', 'auth': 'r', "cert": 'key路径'},



80'alex': {'user': 'alex', 'auth': 'p', "cert": '密码'},



1. },



82'c2.salt.com': {



83'root': {'user': 'root', 'auth': 'r', "cert": 'key路径'},



84'alex': {'user': 'alex', 'auth': 'p', "cert": '密码'},



1. },
2. 
3. }
4. 
5. for row in db\_dict.keys():
6. print(row)
7. 
8. hostname = input('请选择主机: ')
9. tran = paramiko.Transport((hostname, 22,))
10. tran.start\_client()
11. 
12. for item in db\_dict[hostname].keys():
13. print(item)
14. 
15. username = input('请输入用户: ')



100

1. user\_dict = db\_dict[hostname][username]
2. if username['auth'] == 'r':
3. key = paramiko.RSAKey.from\_private\_key\_file(user\_dict['cert'])
4. tran.auth\_publickey(username, key)
5. else:
6. pw = user\_dict['cert']
7. tran.auth\_password(username, pw)
8. 
9. # 打开一个通道



1. chan = tran.open\_session()
2. # 获取一个终端



1. chan.get\_pty()
2. # 激活器



1. chan.invoke\_shell()
2. 
3. interactive\_shell(chan)
4. 
5. chan.close()
6. tran.close()
7. 
8. 
9. if\_\_name\_\_ == '\_\_main\_\_':
10. run()
11. 
12. 提示用户选择主机和用户（记录操作日志）

