第1章 为什么需要SSH
1.远程连接Linux的姿势
2.为什么使用SSH连接
3.抓包演示
第2章 SSH远程连接流程
1.SSH加密原理
2.SSH交换公钥流程
3.SSH身份认证方式
3.流程总结
第3章 SSH密钥对方式连接
1.SSH密钥对创建流程
2.创建密钥对
3.查看生成的密钥对
4.分发公钥到远程机器
5.验证是否可以免密码登陆
6.客户端检查服务端公钥信息
7.SSH密钥对关键文件解释
第4章 SSH远程执行命令
1.SSH远程执行命令说明
2.SSH远程执行命令实战
第5章 SSH安全防范
1.目前SSH存在的安全隐患
2.SSH安全优化步骤
3.测试方法
第6章 SSH免交互分发密钥
1.目前存在的问题
2.我们想要的效果
3.收集正常创建密钥对的交互步骤
4.解决第一次交互
5.解决第二次交互
6.编写面交互分发密钥脚本
第7章 拓展练习

第1章 为什么需要SSH

1.远程连接Linux的姿势

```
1.SSH连接
2.Telnet连接
3.本地显示器键盘支直连
```

2.为什么使用SSH连接

```
SSH:
linux默认 安全性好 传输加密 默认端口号22

TELNET:
网络设备上默认的是telnet 安全性较差 传输明文 默认不允许用root登录 默认端口号23
```

3.抓包演示

安装telnet服务

```
yum provides telnet*
yum install -y telnet-server
systemctl start telnet.socket
systemctl status telnet.socket
netstat -lntup|grep 23
```

使用telnet连接虚拟机:

```
telnet 10.0.0.61
```

telnet抓包:

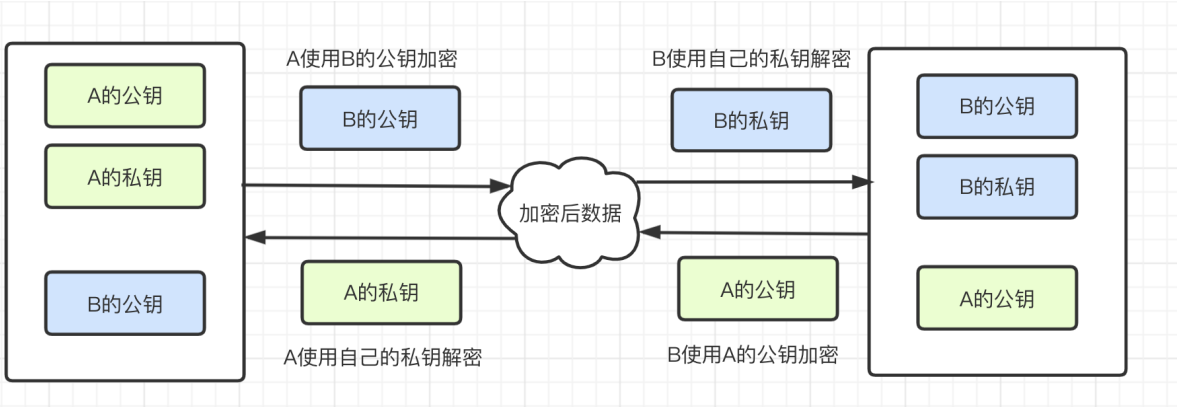
SSH抓包:

第2章 SSH远程连接流程

1.SSH加密原理

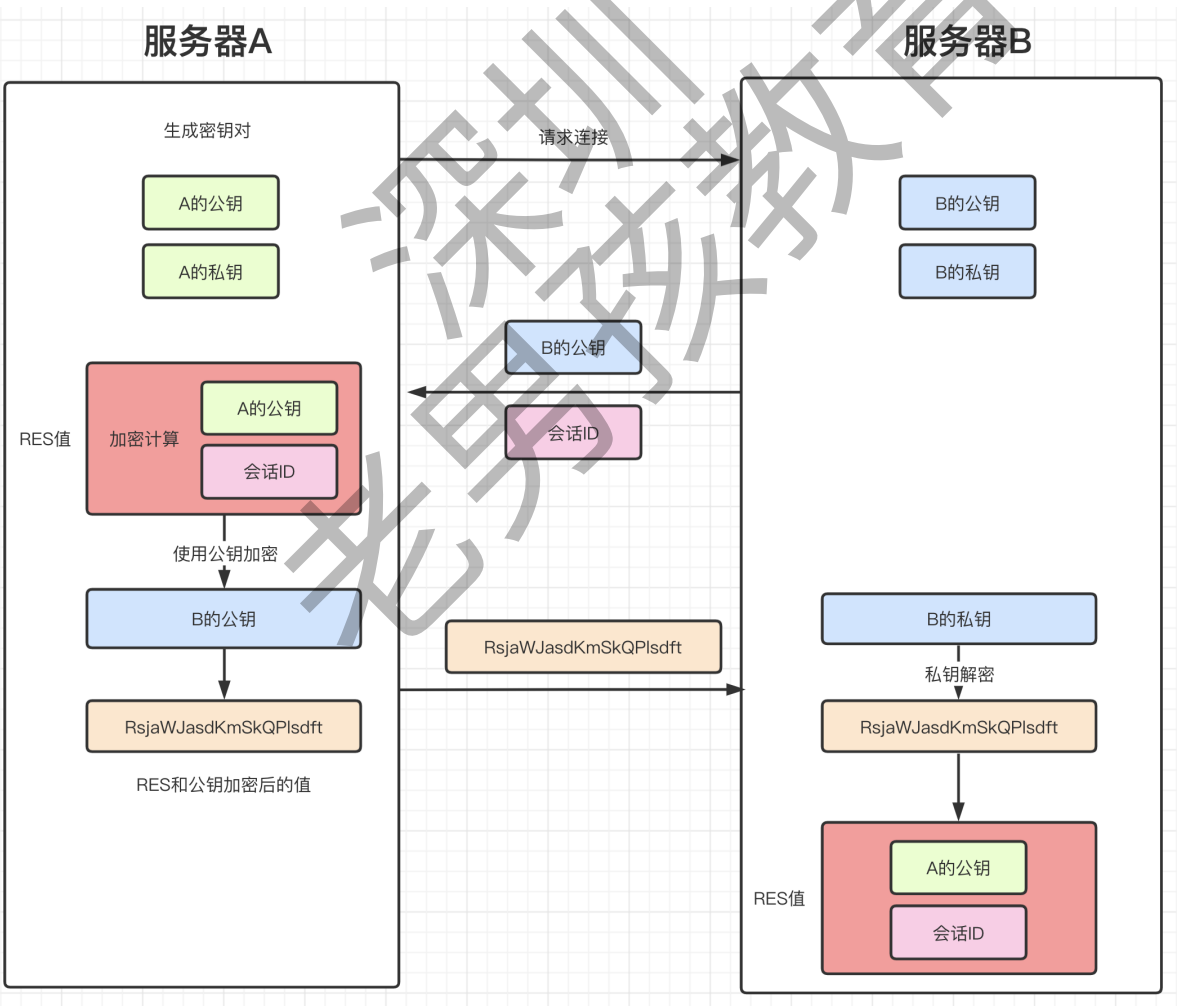
对称加密：
就是指双方共用一套公钥和私钥，加密解密使用的都是同一套公钥私钥，如果被泄露，双方的数据都有风险。

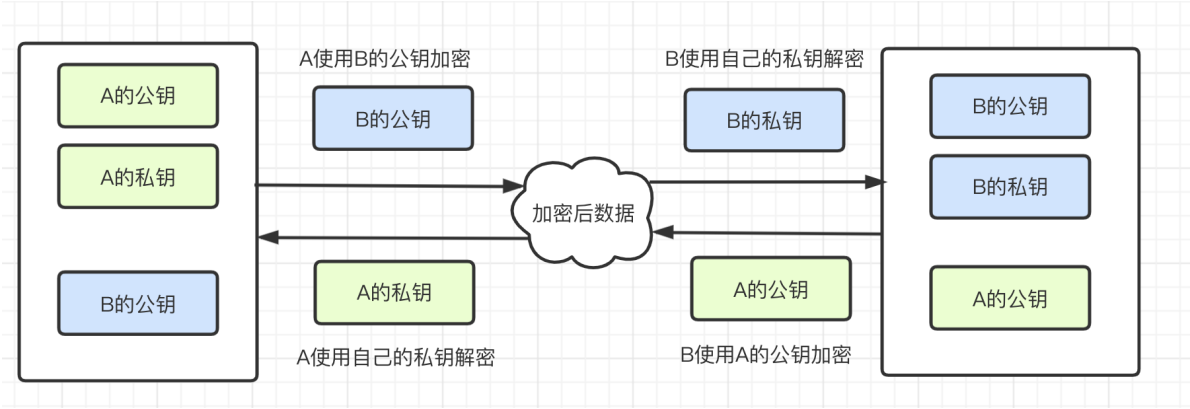
非对称加密：
非对称加密则包含了两套秘钥，公钥 以及 私钥
其中公钥用来加密，私钥用来解密。双方都是用对方的公钥加密数据，使用自己的私钥来解密数据，而公钥可以随便传递，即使泄露也无风险。



2.SSH交换公钥流程

```
1.A服务器发起连接B服务器的请求
2.B服务端返回自己的公钥以及一个会话ID,此时A服务器得到了B服务器的公钥
3.A服务器生成密钥对
4.A服务器用自己的公钥异或会话ID，计算出一个值，并用B服务器的公钥加密
5.A服务器发送加密后的值到B服务器，B服务器用私钥解密
6.B服务端用解密后的值异或会话ID，计算出客户端的公钥，此时B服务器得到了A服务器的公钥
7.最后双方服务器都持有了3个密钥，分别为自己的一对公钥私钥，以及对方的公钥
```





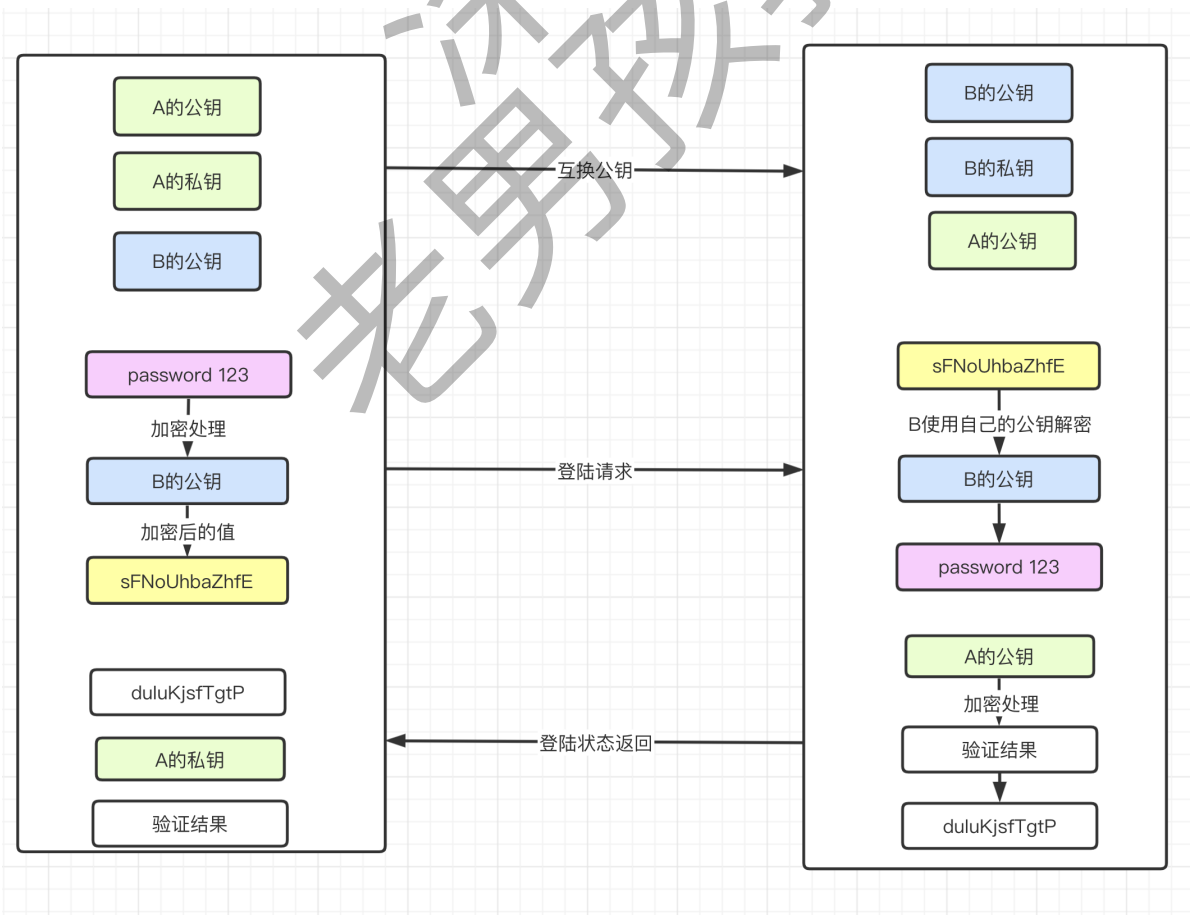
3.SSH身份认证方式

SSH的身份认证分为两个流程：

```
1.基于密码方式认证
2.基于密钥方式认证
```

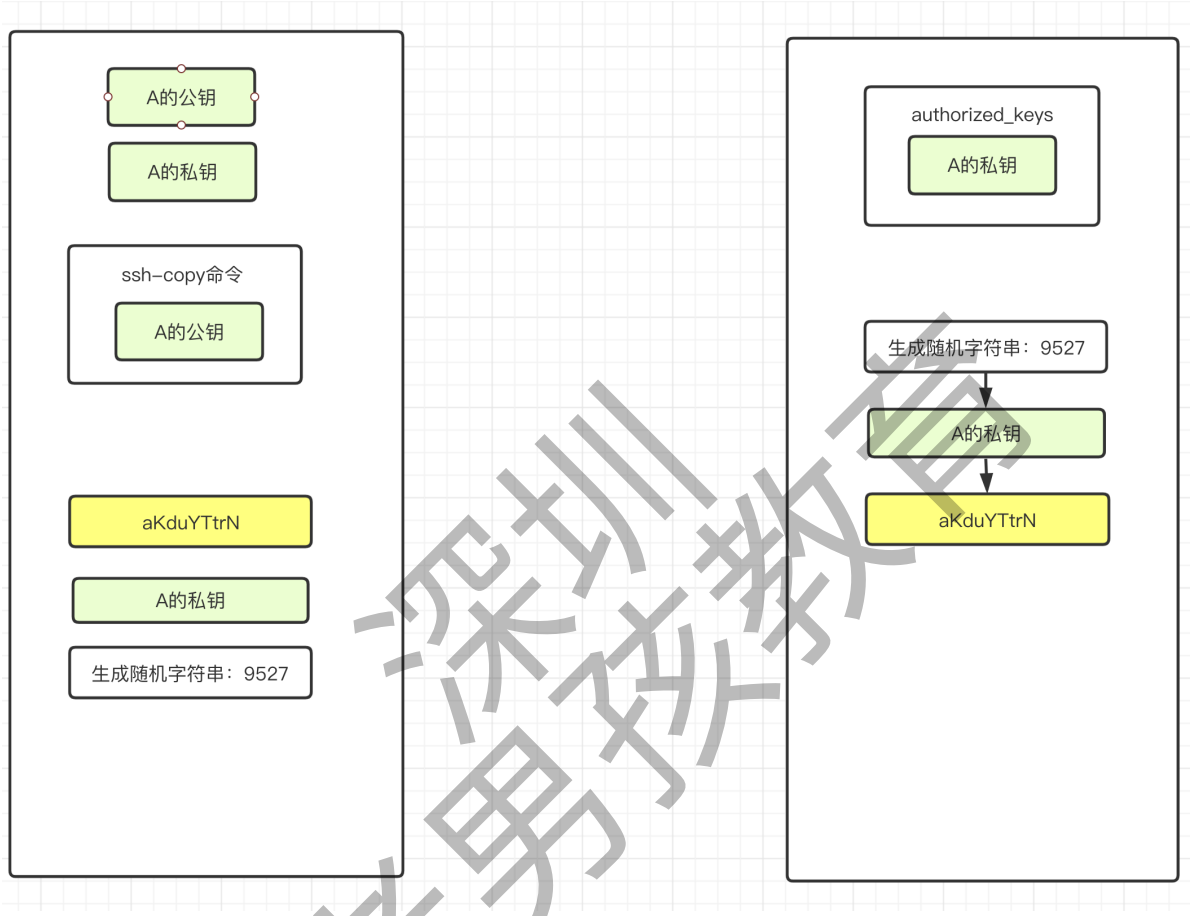
基于密码方式认证：

```
A为客户端，B为服务端
1.服务端收到登录请求后，首先互换公钥
2.客户端用服务端的公钥加密用户密码并发送给服务端
3.服务端用自己的秘钥解密后得到账号密码，然后进行验证
4.服务端用客户端的公钥加密验证结果并返回
5.服务端用自己的秘钥解密后得到验证结果
```



基于密钥方式认证：

```
A为客户端，B为服务端
1.客户端在本地生成一对密钥
2.客户端将自己的公钥使用ssh-copy-id命令发送到服务端
3.此时会提示客户端输入账号密码。客户端再次发送连接请求，包括ip、用户名
4.服务端得到客户端的请求后，会在家目录下.ssh目录里的authorized_keys中查找，如果有响应的IP和用户，就会随机生成一个字符串
5.服务端将生成的字符串使用客户端发送过来的公钥进行加密，然后再发送给客户端
6.客户端收到加密后的内容后，会使用自己的私钥进行解密，然后将解密后的随机字符串发送给服务端
7.服务端接受到客户端发来的随机字符串后，跟之前的随机字符串进行比对，如果一致，就允许免密码登录
```



3.流程总结

```
1. 客户端 - 服务端   请求建立ssh远程连接
2. 服务端 - 客户端   请求确认公钥信息
3. 客户端 - 服务端   确认接收公钥信息，保存到~/.ssh/authorized_keys文件中
4. 服务端 - 客户端   询问用户密码信息
5. 客户端 - 服务端   用户密码信息
6. 服务端 - 客户端   确认密码信息正确  远程连接建立

1之后：不用反复确认公钥信息
6之后：所有传输的数据信息会进行加密处理
```

第3章 SSH密钥对方式连接

1.SSH密钥对创建流程

```
1.执行密钥对创建命令生成公钥和私钥
2.执行命令分发公钥到远程服务器
3.根据提示输入远程服务器的密码
4.公钥分发成功后即可使用私钥免密码登陆
```

2.创建密钥对

```
[root@m-61 ~]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):    #私钥存储位置，直接会车
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):             #输入密码，直接回车
Enter same passphrase again:                                    #确认密码，直接回车
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:n2WWZ2Xy7gA4oiPg3nTKRcQXeUt9MCv2nOgbVYxLuGY root@m-61
The key's randomart image is:
+---[RSA 2048]----+
|        .. .o.   |
|     .  ..o..=.  |
|      o .o+.+.+ o|
|     . . ..O = = |
|  .   . S E % o .|
| . . . . * B + . |
|  . o =   =   . .|
| . + = .   o   o |
|  . +     .     .|
+----[SHA256]-----+
[root@m-61 ~]#
```

3.查看生成的密钥对

```
[root@m-61 ~]# ls -l ~/.ssh/
总用量 8
-rw------- 1 root root 1675 4月  22 20:21 id_rsa
-rw-r--r-- 1 root root  391 4月  22 20:21 id_rsa.pub
```

4.分发公钥到远程机器

```
[root@m-61 ~]# ssh-copy-id 10.0.0.31        #分发命令
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
The authenticity of host '10.0.0.31 (10.0.0.31)' can't be established.
ECDSA key fingerprint is SHA256:iX3CX/sXoReWtwFzV0Xdiyhmim53Q5kPomfgPNTAMIA.
ECDSA key fingerprint is MD5:06:ee:3f:fa:0f:02:6f:0e:4f:37:50:fd:cd:60:3f:43.
Are you sure you want to continue connecting (yes/no)? yes      #确认指纹信息
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@10.0.0.31's password:                              #输入密码
```

```
Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '10.0.0.31'"
and check to make sure that only the key(s) you wanted were added.
```

5.验证是否可以免密码登陆

```
[root@m-61 ~]# ssh 10.0.0.31
Last failed login: Thu Apr 22 20:23:23 CST 2021 from 10.0.0.61 on ssh:notty
There was 1 failed login attempt since the last successful login.
Last login: Thu Apr 22 15:10:16 2021 from 10.0.0.1

[root@nfs-31 ~]# hostname
nfs-31
[root@nfs-31 ~]# whoami
root
[root@nfs-31 ~]# 登出
Connection to 10.0.0.31 closed.
```

6.客户端检查服务端公钥信息

```
[root@nfs-31 ~]# cat ~/.ssh/authorized_keys
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCnFtguJhyxJnNs7rMiE/VSNUm6SHne6YpbSgcRBOoJpcTaIO
qvxo7lInzYwJZGxzTzbIIcQdSD8+Px63+utbKxZn1r2go4v23VHE4CgYMR+Nc5tnHS9G76giCJdZsP
6Glx8QuqvOlT6/9wA+mnYIKZLThPxFWuz2dUEN6dQtQM25HCBxqibzlm2v6vrXVM7kwpGCYA9vdqF0
ZjS+CF6P6hy9kG7Df/LTMbGiZ1L9wE/AP4nhe61EzVjRjMMLSwOds2UXp7hhZGd9Oj+wJNUcupvW5f
Old7bd3td+X3nwLP2JKUVknAw8C6dCQydDVPcK4x6rdVyhOVqeHYaXOCfHq5 root@m-61
```

7.SSH密钥对关键文件解释

SSH服务端:

```
[root@m-61 ~]# ll ~/.ssh/
总用量 12
-rw------- 1 root root 1675 4月  22 20:21 id_rsa                  #私钥，自己保留，注意，权限是600
-rw-r--r-- 1 root root  391 4月  22 20:21 id_rsa.pub          #公钥，发给需要被连接的服务器
-rw-r--r-- 1 root root  171 4月  22 20:23 known_hosts         #保存已经连接过的主机的指纹信息
```

SSH客户端:

```
[root@nfs-31 ~]# ll ~/.ssh/
总用量 4
-rw------- 1 root root 391 4月  22 20:23 authorized_keys  #用来保存接收到的公钥,注意,权限是600
```

第4章 SSH远程执行命令

1.SSH远程执行命令说明

```
SSH不仅仅可以用来连接服务器，也可以远程执行命令。
SSH远程执行命令并不需要登陆到远程服务器，只需要配置好密钥对即可，执行完成后进程就结束了。
```

2.SSH远程执行命令实战

```
#1.远程连接命令后直接加上需要执行的命令即可
[root@m-61 ~]# ssh 10.0.0.31 hostname
nfs-31

#2.远程执行命令不支持别名
[root@m-61 ~]# ssh 10.0.0.31 'll /root'
bash: ll: 未找到命令
[root@m-61 ~]# ssh 10.0.0.31 'ls -l /root'
总用量 12
-rw-------. 1 root root 1270 12月 13 11:04 anaconda-ks.cfg
-rw-r--r--  1 root root 1920 12月 13 11:30 init.sh
-rwxr-xr-x  1 root root  698 12月 13 11:50 set-ip.sh
```

第5章 SSH安全防范

1.目前SSH存在的安全隐患

```
问题1: 虽然使用了秘钥连接,但是原来的密码连接还是可以使用的

问题2: 默认端口号所有人都知道,需要修改

问题3: 如果被别人偷走了私钥,在不受信任的机器可以不需要root账号密码登录
```

2.SSH安全优化步骤

优化1: 禁止使用账号密码登陆

```
PubkeyAuthentication yes
PasswordAuthentication no
```

优化2:修改默认端口

```
Port 9527
```

优化3:限制主机登陆条件

```
#1.更改SSH监听地址为内网地址
ListenAddress 172.16.1.41

#2.配置防火墙规则，只允许通过跳板机登陆，其他地址的连接拒绝掉
yum install iptables-services -y
modprobe ip_tables
modprobe iptable_filter
modprobe iptable_nat
modprobe ip_conntrack
```

```
modprobe ip_conntrack_ftp
modprobe ip_nat_ftp
modprobe ipt_state
systemctl stop firewalld
systemctl disable firewalld
systemctl start iptables.service
systemctl enable iptables.service
iptables -F
iptables -X
iptables -Z
iptables -A INPUT ! -s 172.16.1.61 -p tcp --dport 9527 -j DROP
```

3.测试方法

目前被管理的机器只能通过管理机的内网网段并且只能使用秘钥方式连接

```
ssh 172.16.1.41 -p 9527
```

第6章 SSH免交互分发密钥

1.目前存在的问题

正常操作的步骤:

```
1.生成秘钥对
2.服务端分发秘钥需要手动确认
  - yes
  - 密码
3.修改客户端SSH配置文件,端口号,监听IP,秘钥方式,拒绝密码
4.重启服务
5.测试是否可以免密登录
```

存在的问题:

```
1.操作步骤繁琐，如果服务器数量很多，工作量非常大，效率非常低
2.手动操作容易出现失误，出现失误不好检查，总之，人不可靠
```

2.我们想要的效果

```
新服务器安装好系统后，管理机只需要运行一次脚本,上述所有操作全部自动完成,不需要人工介入，不需要交互输入确认
```

3.收集正常创建密钥对的交互步骤

第一次交互：解决yes

```
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_dsa.pub"
The authenticity of host '10.0.0.41 (10.0.0.41)' can't be established.
ECDSA key fingerprint is SHA256:S+jjKXhBPHotvqYLV5PISxrgnOcPljD9njCavfx+OFo.
ECDSA key fingerprint is MD5:e3:59:ff:b9:3c:75:78:5f:6a:29:53:e5:22:09:0a:51.
Are you sure you want to continue connecting (yes/no)? yes
```

第二次交互：输入密码

```
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@10.0.0.41's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '10.0.0.41'"
and check to make sure that only the key(s) you wanted were added.
```

4.解决第一次交互

```
ssh-copy-id 172.16.1.31 -o StrictHostKeyChecking=no
```

5.解决第二次交互

```
yum install sshpass -y
sshpass -p '123456' ssh-copy-id 172.16.1.31 -o StrictHostKeyChecking=no
```

6.编写面交互分发密钥脚本

```
[root@m-61 /server/scripts]# cat fenfa.sh
#!/bin/bash

#1.生成秘钥
if [ -f /root/.ssh/id_rsa ]
then
    echo "秘钥已经创建"
else
    ssh-keygen -t dsa -f /root/.ssh/id_rsa -N '' > /dev/null 2>&1
    echo "秘钥已经创建"
fi

for ip in {31,41}
do
    echo "========================================================================"
    #2.分发秘钥
    sshpass -p '123456' ssh-copy-id 172.16.1.${ip} -o StrictHostKeyChecking=no > /dev/null 2>&1
    if [ $? == 0 ]
    then
        echo "秘钥分发完毕: 172.16.1.${ip}"
  else
      echo "秘钥分发失败"
    fi

    #3.测试
    ssh 172.16.1.${ip} hostname
    if [ $? == 0 ]
    then
```

```
        echo "OK boy!"
    else
        echo "NO boy!"
    fi
done
```

第7章 拓展练习

正常人类作业：

```
windows:
1.xshell生成密钥对
2.将xshell公钥上传到管理机m-61

管理机-可手动操作:
1.更改ssh默认端口号为9999
2.关闭用户名密码登录方式
3.开启通过密钥对方式连接

被管理机-可手动操作:
1.更改ssh默认端口号为9999
2.关闭用户名密码登录方式
3.开启通过密钥对方式连接
4.指定监听内网地址 172.16.1.x

实现效果:
1.管理机m01只能通过xshell的秘钥方式连接,IP地址用户名密码方式不允许
2.所有主机ssh端口修改为9999
3.被管理机只能通过内网地址使用秘钥认证连接,10网段不允许连接.不允许使用账号密码方连接
```

非正常人类作业：提前做好快照

```
一个脚本完成以下所有任务：
1.管理机自动创建密钥对
2.管理机自动将公钥发送到被管理机
3.修改被管理机的SSH连接端口为9999
4.修改被管理机的SSH监听地址为172.16.1.X
5.修改被管理机不允许使用账号密码方式登陆，只允许使用密钥方式连接
6.修改完之后验证是否生效，验证方法为使用9999端口连接被管理机并打印出主机信息
```