SSH服务详解

惨绿少年 Linux运维, 玩转Linux 0评论 来源：本站原创 29℃ 字体：小 中 大

第1章 SSH服务

1.1 SSH服务协议说明

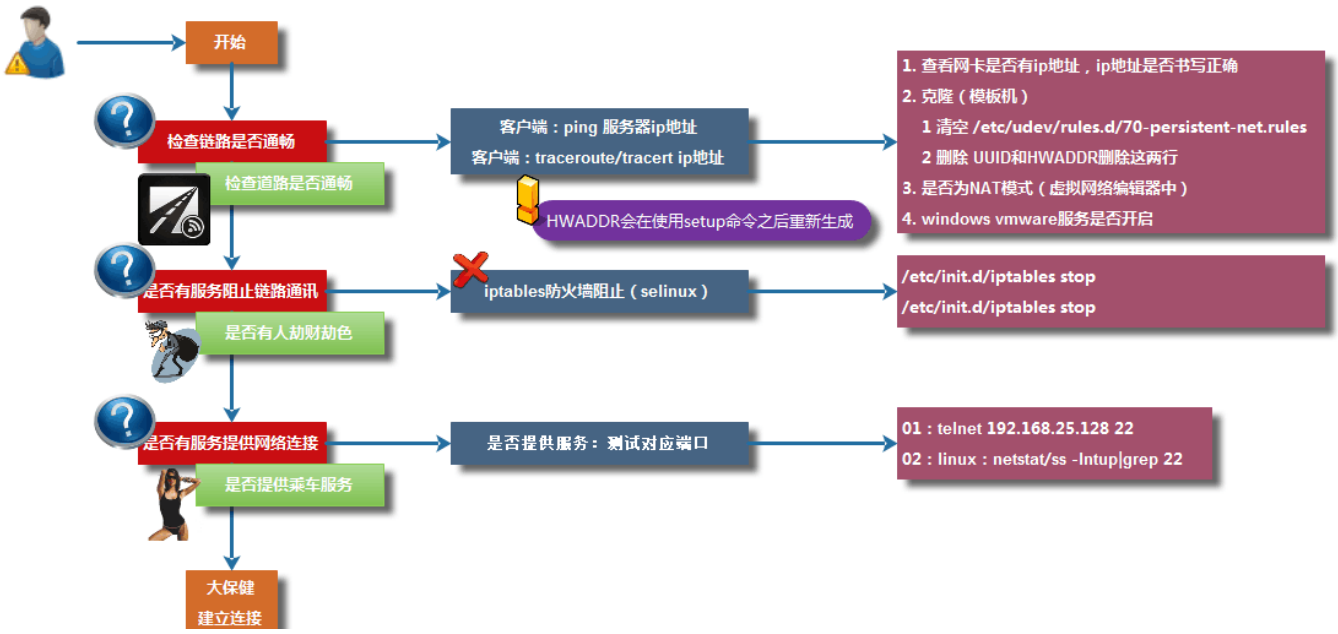
SSH 是 **Secure Shell Protocol** 的简写，由 IETF 网络工作小组（Network Working Group)制定；在进行数据传输之前，SSH先对联机数据包通过加密技术进行加密处理，加密后在进行数据传输。确保了传递的数据安全。

SSH是专为远程登录会话和其他网络服务提供的安全性协议。利用 SSH 协议可以有效的防止远程管理过程中的信息泄露问题，在当前的生产环境运维工作中，绝大多数企业普遍采用SSH协议服务来代替传统的不安全的远程联机服务软件，如telnet(23端口，非加密的)等。

在默认状态下，SSH服务主要提供两个服务功能：

- 一是提供类似telnet远程联机服务器的服务，即上面提到的SSH服务。
- 另一个是类似FTP服务的sftp-server,借助SSH协议来传输数据的.提供更安全的SFTP服务（vsftp，proftp)。

1.1.1 ssh远程连接排错过程



1.2 SSH加密技术说明

简单的说，SSH加密技术就是将人类可以看得懂的数据，通过一些特殊的程序算法，把这些数据变成杂乱的无意义的信息，然后，通过网络进行传输，而当到了目的地后，在通过对应的解密算法，把传过来的加密的数据信息解密成加密前的可读的正常数据。因此，当数据在互联网上传输时即使被有心的黑客监听窃取了，也很难获取到真正黑要的数据。

当前，网络上的数据包加密技术一般是通过所谓的一对公钥与私钥（PublickeyandPivatekey)组合成的密钥对进行加密与解密操作。如下图，A-Server要给B_Client传数据，首先会通过本地的公钥加密后再到发到网络上传输。而加密的数据到达B_Client端后，再经由B_Client本地的私钥将加密的数据解密出来。由于在intemet上传输过程中的数据是加密过的，所以，传输的数据内容一般来说是比较安全的。

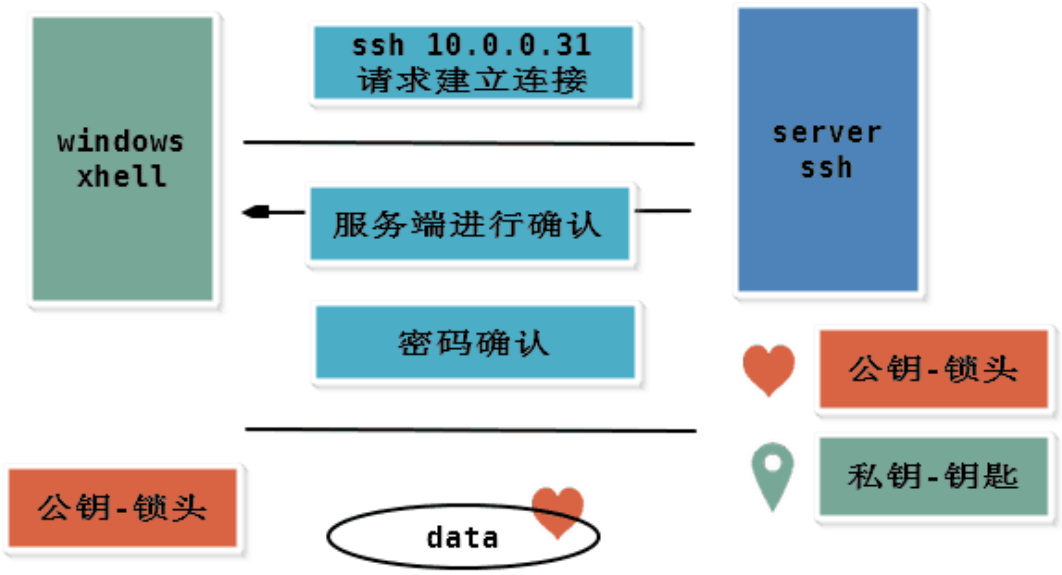


图1-1 ssh认证连接的过程

1.2.2 ssh实现安全链接建立，利用要是和锁头

1. 钥匙=私钥 锁头=公钥，私钥可以解密公钥

2. 公钥可以再网络中传输，私钥再本地主机保存

1.2.3 ssh加密算法

v1漏洞： 密钥不更换

v2 定期更换密钥

利用Diffie-Hellman机制定期更新密钥

1.3 ssh知识要点:

> ssh是安全的加密协议，用于远程链接linux服务器
>
> ssh 默认端口是22，安全协议版本sshv2，出来2之外还有1（有漏洞）
>
> ssh服务端主要包括两个服务功能 ssh远程链接和sftp服务
>
> linux ssh 客户端包括ssh 远程链接命令，以及远程拷贝scp命令等

1.4 SSH服务软件详细说明

1.4.1 什么是ssh服务

SSH服务端是一个守护讲程 (daemon).他在后台运行并响应来自客户端的连接请求。 SSH服务端的讲程名为sshd，负责实时监听远程SSH客户端的远程连接请求，并进行处理，一般包括公共密钥认证、密钥交换、对称密钥加密和非安全连接等。这个SSH服务就是我们前面基础系统优化中保留开机自启动的服务之。

ssh客户端包含ssh以及像scp(远程拷贝） slogin(远程登陆) sftp(安全FTP文件传输）等应用程序。

ssh的工作机制大致是本地的ssh客户端先发送一个连接请求到远程的ssh服务端，服务端检查连接的客户端发送的数据包和IP地址，如果确认合法，就会发送密钥给 SSH的客户端，此时，客户端本地再将密钥发回给服务端，自此连接建立。

1.4.2 ssh软件安装

客户端

```
[root@nfs01 ~]# rpm -qf `which ssh`
openssh-clients-5.3p1-122.el6.x86_64
```

服务端软件

```
[root@nfs01 ~]# rpm -qf `which sshd`
openssh-server-5.3p1-122.el6.x86_64
```

1.4.3 openssh-clinets 软件的主要内容:

```
[root@nfs01 ~]# rpm -ql openssh-clients
/etc/ssh/ssh_config         #ssh客户端配置文件
/usr/bin/.ssh.hmac
/usr/bin/scp                  #远程复制命令
/usr/bin/sftp                 #远程文件传输服务
/usr/bin/slogin              #远程登陆命令
/usr/bin/ssh                  #ssh远程登陆管理主机
/usr/bin/ssh-add
/usr/bin/ssh-agent
/usr/bin/ssh-copy-id        #ssh服务分发公钥命令
/usr/bin/ssh-keyscan
```

1.4.4 openssh-server 软件的主要内容

```
[root@nfs01 ~]# rpm -ql openssh-server
/etc/rc.d/init.d/sshd      #ssh服务启动脚本
/etc/ssh/sshd_config       #ssh服务配置文件
/etc/sysconfig/sshd        #ssh创建密钥有关
/usr/sbin/.sshd.hmac       #ssh加密算法有关文件
/usr/sbin/sshd             #ssh服务进程启动命令
```

注意：使用sshd采用绝对路径进行启动

```
[root@test ~]# sshd
sshd re-exec requires execution with an absolute path
```

1.5 ssh服务配置文件说明：

01. 配置文件中所有注释信息，表示默认参数配置

02. 配置文件中#空格 后面内容表示说明信息

    #参数 表示配置参数信息

03. 配置文件参数信息修改后，一旦变为注释，即还原为默认配置

1.5.1 ssh服务的配置文件路径

```
vim  /etc/ssh/sshd_config
```

修改SSH服务的运行参数，是通过修改配置文件**/etc/ssh/sshd_config**实现的。

一般来说SSH服务使用默认的配置已经能够很好的工作了，如果对安全要求不高，仅仅提供SSH服务的情况，可以不需要修改任何配置。

1.5.2 配置文件中常用配置说明

```
[root@test ~]# vim /etc/ssh/sshd_config
#       $OpenBSD: sshd_config,v 1.80 2008/07/02 02:24:18 djm Exp $

# This is the sshd server system-wide configuration file.  See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options change a
# default value.

# OpenSSH is to specify options with their default value where
# possible, but leave them commented.  Uncommented options change a
# default value.

Port 25113                    #端口
ListenAddress 10.0.0.41   #监听地址（本地网卡地址），指定本地网卡那个网卡提供服务
PermitRootLogin no          #是否允许root用户登陆
#PermitEmptyPasswords no  #禁止空密码登陆
#UseDNS no                    #不使用DNS
GSSAPIAuthentication no    #API认证
# 连接慢的解决
#AddressFamily any          #指定监听ipv4地址，或是ipv6地址，或者所有都监听
```

配置文件内容说明：

> 井号(#)注释的参数信息为默认配置
>
> 井号(#)后面有空格的为描述信息

井号(#)后面没有空格的为参数信息

另外：配置文件参数信息修改后，一旦变为注释，即还原为默认配置

1.5.3 配置文件语法检查方法

使用sshd -t 命令 对配置文件的语法进行检查

```
正确↓
[root@backup ~]# sshd -t /etc/ssh/sshd
Extra argument /etc/ssh/sshd.
语法格式有错误  ↓
[root@test ~]# sshd -t /etc/ssh/sshd_config
/etc/ssh/sshd_config: line 50: Bad configuration option: uthorizedKeysFile
/etc/ssh/sshd_config: terminating, 1 bad configuration options
```

1.5.4 ListenAddress 监听地址的说明

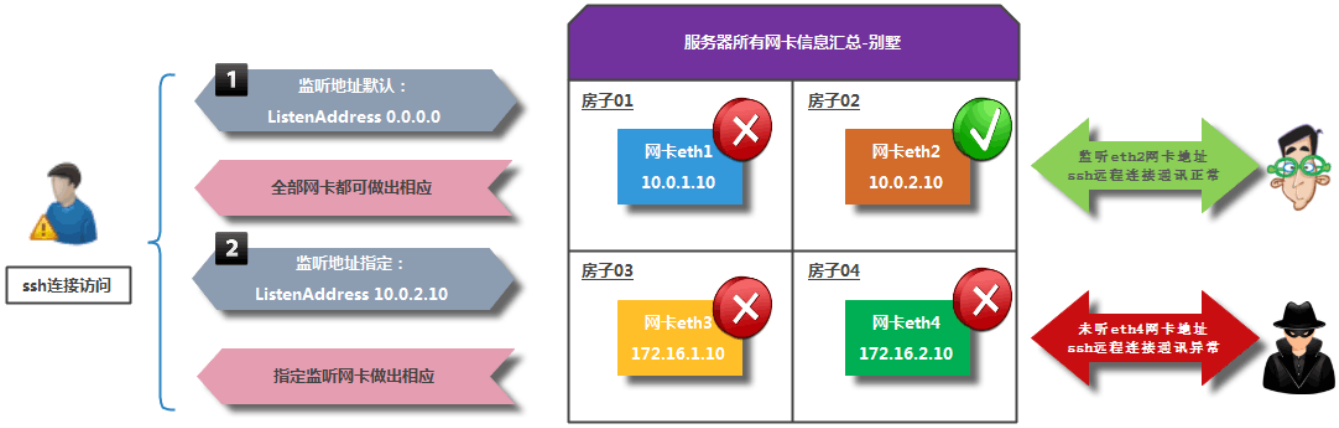


图1-2 ssh服务监听参数说明

如图所示，sshd_config配置文件中实际监听本地的网卡，并非网络地址

监听地址只能监听本地网卡上配置的地址，**监听的网卡可以对请求做出相应**，为未监听的网卡不响应请求。

1.5.5 SSH配置文件相关参数详细说明

命令参数	**参数说明**
Port	指定sshd进程监听的端口号，默认为22.可以使用多条指令监听多个端口. 默认将在本机的所有网络接口上监听，但是可以通过ListenAddress指走只在某个特定的接口上监听.
PermitEmptyPasswords	是否允许密码为空的用户远程登录.默认为“no”
	是否允许root登录.可用值如下：“yes”(默认）表示允许.”no”表示禁止.

PermitRootLogin	“without-password” 表示禁止使用密码认证登录.”forced-commands-only”表示只有在指走了command选项的情况下才允许使用公钥认证登录.同时其它认证方法全部被禁止.这个值常用于做远程备份之类的事情. 1. 多开一个窗口 2. 临时多部署一条连接方式 3. 给普通用户sudo权限
UseDNS	指定定sshd是否应该对远程主机名进行反向解折，以检查此主机名是否与其IP地址真实对应.默认值为“yes”.
ListenAddress	指定监听并提供服务相应的网卡地址信息

1.5.6 快速修改配置参数

```
sed -i '13 iPort 52113\nPermitRootLogin no\nPermitEmptyPasswords no\nUseDNS no\nGSSAPIAuthenticat
```

1.6 ssh服务认证类型

01. 基于口令认证方式
02. 基于密钥认证方式

1.6.1 基于密码的认证类型

基于口令的安全验证的方式就是大家现在一直在用的，只要知道服务器的SSH连接帐号和口令(当然也要知道对应服务器的 IP及开放的 SSH端口，默认为22),就可以通过 ssh客户端登录到这台远程主机。此时，联机过程中所有传输的数据都是加密的。

演示了 SecureCR及ssh客户端连接，口令验证的测试。

```
[root@test ~]# ssh 10.0.0.250
The authenticity of host '10.0.0.250 (10.0.0.250)' can't be established.
RSA key fingerprint is d3:41:bb:0d:43:88:da:a3:2c:e8:36:91:11:c9:e4:9c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.0.250' (RSA) to the list of known hosts.
root@10.0.0.250's password:    #需要输入密码
Last login: Mon Oct 16 21:13:58 2017 from 10.0.0.1
[root@test ~]#
```

1.6.2 基于密钥的安全认证方法

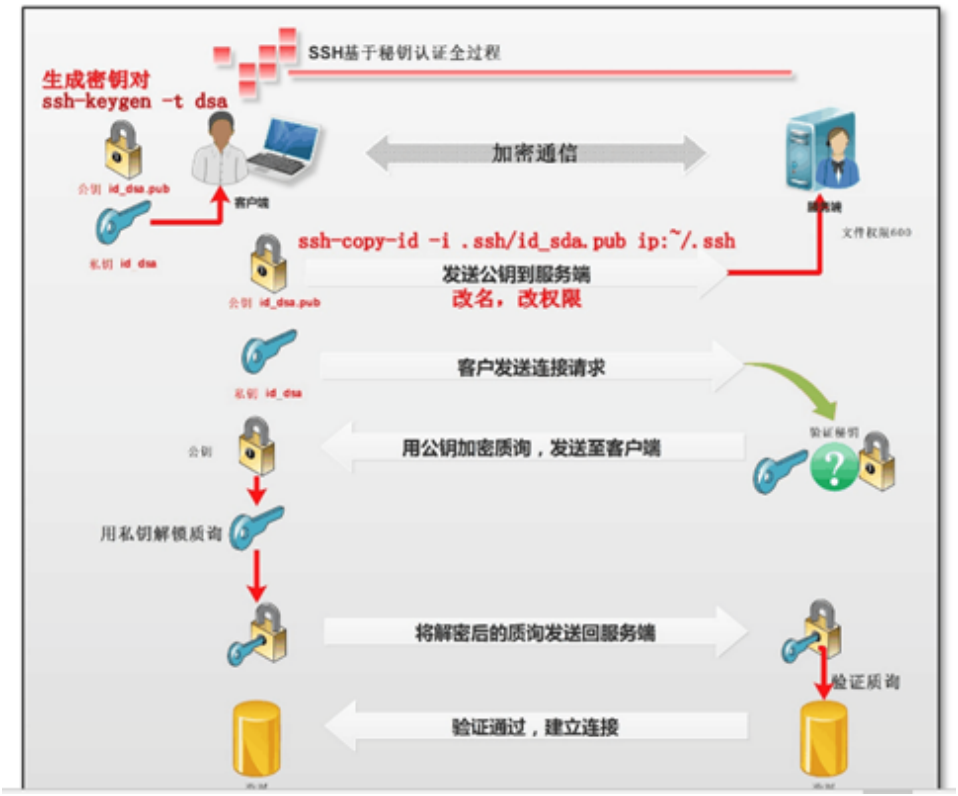
基于密钥的安全验证方式是指，需要依靠密钥，也就是必须事先建立一对密钥对，然后把公用密钥(锁头)（Public key)放在需要访问的目标服务器上，另外，还需要把私有密钥（钥匙）（Private key）放到SSH

的客户端或对应的客户端服务器上。

> 私钥不能在网络中传输—私钥可以解密公钥
>
> 公钥可以再网路中传输—公钥不能解密私钥

此时，如果要想连接到这个带有公用密钥的SSH服务器，客户端SSH软件或者客户端服务器就会向SSH服务器发出请求，请求用联机的用户密钥进行安全验证。SSH服务器收到请求之后，会先在该SSH服务器上连接的用户的家目录下寻找事先放上去的对应用户的公用密钥，然后把它和连接的SSH客户端发送过来的公用密钥进行比较。如果两个密钥一致，SSH服务器就用公用密钥加密“质询“（challenge)并把它发送给SSH客户端。

1.7 基于秘钥登录配置



1.7.1 环境准备

作用	主机名	ip
管理服务器	m01	10.0.0.61
备份服务器	backup	10.0.0.41
存储服务器	nfs01	10.0.0.31

1.7.2 第一个里程碑： 在备份服务器上创建密钥对

```
[root@backup ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):   #指定密钥对的保存路径
```

```
Enter passphrase (empty for no passphrase):        #为密钥对创建密码
Enter same passphrase again:                         #确认为密钥对创建的密码
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:
72:48:65:1d:25:69:e1:4c:ae:2b:6f:a5:aa:70:96:1e root@backup
The key's randomart image is:
+--[ RSA 2048]----+     #2048表示加密的位数为2048位
|        o.==.      |
|       o =+.       |
|      .  .+        |
|     . . .         |
|      o S          |
|     . o ..        |
|  . E . .o         |
|   = . oo          |
|    o..o.          |
+-----------------+
```

参数说明：

-t 指定创建密钥对的类型，可以创建的类型如下↓

> Specifies the type of key to create. The possible values are “**rsa1**” for protocol version 1 and “**dsa**”, “**ecdsa**” or “**rsa**” for protocol version 2.

查看创建出来的密钥对：

```
[root@backup ssh]# ll ~/.ssh/
total 12
-rw------- 1 root root 1675 Oct 18 11:07 id_rsa          #私钥文件
-rw-r--r-- 1 root root  393 Oct 18 11:07 id_rsa.pub     #公钥文件
-rw-r--r-- 1 root root  784 Oct 11 16:27 known_hosts
```

1.7.3 第二个里程：将公钥分发给存储服务器

```
[root@backup ssh]# ssh-copy-id -i ~/.ssh/id_rsa.pub 172.16.1.31 #用户使用当前用户
root@172.16.1.31's password:   #注意：首次分发密钥需要输入对端服务器的用户密码
Now try logging into the machine, with "ssh '172.16.1.31'", and check in:
  .ssh/authorized_keys    #公钥分发到对端后进行改名

to make sure we haven't added extra keys that you weren't expecting.
```

> Formatting page, please wait…
>
> SSH-COPY-ID(1) SSH-COPY-ID(1)
>
> NAME
>
> ssh-copy-id – install your public key in a remote machine’s autho-
>
> rized_keys
>
> SYNOPSIS
>
> ssh-copy-id [-i [identity_file]] [user@]machine

1.7.4 第三个里程碑：进行登录测试

```
[root@backup ~]# ssh nfs01
Last login: Wed Oct 18 10:13:17 2017 from 10.0.0.1
[root@nfs01 ~]#
```

1.8 telnet服务简介

1.8.1 部署telnet服务

第一个里程碑：安装telnet服务软件

```
[root@test ~]# yum install telnet telnet-server -y
```

第二个里程碑：设置开机自启动

```
[root@test ~]# vim /etc/xinetd.d/telnet
```

修改xinetd下的配置文件，从而管理telnet服务

```
[root@test ~]# vim /etc/xinetd.d/telnet
# default: on
# description: The telnet server serves telnet sessions; it uses \
#       unencrypted username/password pairs for authentication.
service telnet
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
        disable         = no
}
```

第三个里程碑： 启动xinetd 服务，让telnet能够开机自启动

```
[root@test ~]# /etc/init.d/xinetd start
Starting xinetd:                                           [  OK  ]
```

1.8.2 客户端测试

说明:

telnet服务默认不支持root用户直接登陆。

```
[C:\]$ telnet 10.0.0.250
Connecting to 10.0.0.250:23...
Connection established.
To escape to local shell, press 'Ctrl+Alt+]'.
CentOS release 6.9 (Final)
Kernel 2.6.32-696.el6.x86_64 on an x86_64
test login: oldboy
Password:
```

```
Last login: Wed Oct 18 09:41:19 from 10.0.0.1
[oldboy@test ~]$
```

1.8.3 ssh与telnet区别（对比示意图）

01．ssh是加密的服务协议，telnet服务是非加密的

02.ssh服务默认支持root用户登陆，telnet用户默认不支持root用户登陆

1.9 进行免密码scp传输测试

```
[root@backup ~]# scp -rp  /etc/hosts nfs01:/tmp/
hosts                                         100%  357     0.4KB/s   00:00
```

1.9.1 scp的基本语法使用：

```
scp - secure copy (remote file copy program)
# 每次都是全量拷贝，增量拷贝rsync
```

推：*PUSH*

```
scp -P22 -rp /tmp/clsn clsn@10.0.0.143:/tmp
```

参数说明：

<- -P（大写，注意和ssh命令的不同）接端口，默认22端口时可以省略-P22;

<- -r递归，表示拷贝目录;

<- -p表示在拷贝前后保持文件或目录属性;

<- -l limit 限制速度。

<- /tmp/clsn为本地的目录。“@”前为用户名，“@”后为要连接的服务器的IP。IP后的:/tmp目录，为远端的目标目录。

说明：

以上命令作用是把本地/tmp/clsn拷贝到远端服务器10.0.0.143的/tmp目录;

拉：*PULL*

```
scp -P22 -rp root@10.0.0.7:/tmp/clsn /opt/
```

说明:

还可以把远端目录抓到本地

结论:

```
scp为远程拷贝文件或目录的命令，更多用法，请man scp；
拷贝权限为连接的用户对应的权限。
```

1.10 使用sftp进行基于密钥的文件传输

1.10.1 sftp简介

> sftp是Secure File Transfer Protocol的缩写，安全文件传送协议。可以为传输文件提供一种安全的网络的加密方法。sftp 与 ftp 有着几乎一样的语法和功能。**SFTP 为 SSH的其中一部分**.

1.10.2 sftp命令说明

```
[root@m01 ~]# sftp -oPort=22  172.16.1.31
Connecting to 172.16.1.31...
sftp>
```

说明:

-o 连接的时候指定选项

Port=22 端口指定为22、

1.10.3 sftp使用参数说明

操作远程服务器

```
ls    显示远端主机的列表
cd    切换远程的工作目录
pwd   显示远程的工作目录
```

操作本地服务器

```
lls   显示本地主机的列表
lcd   切换本地的工作目录
lpwd  查看本地目录信息
```

上传下载文件参数

```
get   --- 表示从远程服务器下载数据（单个文件）
mget  --- 表示下载多个文件
put   --- 表示从本地服务器上传数据（单个文件）
mput  --- 表示上传多个文件
```

查看帮助的方式:

```
sftp> help          显示帮助信息
```

1.11 ssh相关重点知识总结

> * ssh协议：sshd—远程连接（sshd),sftp
> * 为加密的远程连接协议，相关软件有openssh.openssh—https。
> * 默认端口22
> * 协议版本1X和2.x,2.x更安全。了解SSH协议原理（ssh连接过程X
> * 服务端ssh远程连接服务，sftp服务。sshd守护进程，开机要自启动。
> * ssh客户端包含ssh,scp,sftp命令。
> * ssh安全验证方式：口令和密钥，这两种都是基于口令的，SSH密钥登录的原理。
> * ssh服务安全优化，修改默认端口22,禁止root远程连接，禁止dns,SSH只监听内网IP
> * ssh密钥对，公钥(publickey)在服务器端，比喻就是锁头，私钥（privatekey)在客户端，比喻就是钥匙。

第2章 重点知识补充

2.1 一个服务始终无法启动

01.服务的查日志/系统日志

02.检查服务端口有没有冲突

2.2 给你一个端口如何命令行查出对应的服务是什么?

测试服务端口有没有开启

```
ss -lntup|grep 22
netstat -lntup|grep -w "22"
lsof -i:22
grep  "\b22/\b" /etc/services
nmap -p 22 172.16.1.41
nc 172.16.1.41 22
telnet 172.16.1.41 22
```

2.2.1 根据进程名查看对应的端口是什么?

```
netstat/ss -lntup|grep 进程或服务名字
```

2.3 ssh入侵案例说明

被入侵实例http://phenixikki.blog.51cto.com/7572938/1546669

IP何防止SSH登录入侵小结:

> 1、用密钥登录，不用密码登陆
>
> 2、牝牛阵法：解决SSH安全问题
>
> a.防火墙封闭SSH,指定源IP限制(局域网、信任公网)
>
> b.开启 SSH 只监听本地内网 IP （ ListenAddress 172.16.1.61 ）
>
> 3、尽量不给服务器外网ip
>
> 4、最小化(软件安装-授权)
>
> 5、给系统的重要文件或命令做一个指纹
>
> 6、给他锁上 chattr +i +a

第3章 扩展问题

3.1 服务端端口号变化了，如何基于秘钥连接

3.1.1 环境准备

实验环境:

```
[root@test ~]# cat /etc/redhat-release
CentOS release 6.9 (Final)
```

将一台服务器的ssh服务端口修改为63389

```
[root@test ~]# netstat -lntup|grep sshd
tcp        0      0 0.0.0.0:63389  0.0.0.0:*         LISTEN      5083/sshd
tcp        0      0 :::63389         :::*              LISTEN      5083/sshd
```

3.1.2 通过另外一台服务器创建并分发密钥

第一个里程碑： 现创建密钥使用 ssh-keygen

```
[root@backup ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):   #指定密钥对的保存路径
Enter passphrase (empty for no passphrase):        #为密钥对创建密码
Enter same passphrase again:                          #确认为密钥对创建的密码
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:
```

```
72:48:65:1d:25:69:e1:4c:ae:2b:6f:a5:aa:70:96:1e root@backup
The key's randomart image is:
+--[ RSA 2048]----+     #2048表示加密的位数为2048位
|        o.==.      |
|       o =+.       |
|      .  .+        |
|     . . .         |
|      o S          |
|     . o ..        |
|  . E . .o         |
|   = . oo          |
|    o..o.          |
+-----------------+
```

第二个里程碑：分发密钥，注意ssh的端口

```
[root@backup ~]# ssh-copy-id  -i ~/.ssh/id_rsa.pub "-p63389 172.16.1.250"
The authenticity of host '[172.16.1.250]:63389 ([172.16.1.250]:63389)' can't be established.
RSA key fingerprint is d3:41:bb:0d:43:88:da:a3:2c:e8:36:91:11:c9:e4:9c.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[172.16.1.250]:63389' (RSA) to the list of known hosts.
root@172.16.1.250's password:
Now try logging into the machine, with "ssh '-p63389 172.16.1.250'", and check in:
  .ssh/authorized_keys   #分发到对端服务器后进行改名

to make sure we haven't added extra keys that you weren't expecting.
```

说明：

通过 man 手册找到密钥分发的命令格式。

-i 参数指定 公钥文件的存放位置

[use@]表示使用的用户，默认使用当前登陆的用户

-p 指定端口，主要要在双引号之间（通过 cat `which ssh-copy-id` 命令脚本内容得知）

```
[root@backup ~]# man ssh-copy-id
Formatting page, please wait...
SSH-COPY-ID(1)                                                  SSH-COPY-ID(1)
NAME
       ssh-copy-id  -  install  your  public  key in a remote machine's autho-
       rized_keys
SYNOPSIS
```

第三个里程碑： 测试密钥登陆

```
[root@backup ~]# ssh 172.16.1.250 -p 63389
Last login: Wed Oct 18 15:42:05 2017 from 10.0.0.41
[root@test ~]#
```

3.2 如何实现自动创建秘钥对，同时分发公钥（编写脚本实现）

脚本内容：

```
[root@m01 ~]# vim /server/scripts/piliang_fenfa.sh
#!/bin/bash
#make key
\rm -f /root/.ssh/id_dsa
ssh-keygen -t dsa -f /root/.ssh/id_dsa -P "" -q
#fengfagongyao
for ip in 8 31 41
do
echo ====fenfa key to host 172.16.1.$ip====
sshpass -p123456 ssh-copy-id -i /root/.ssh/id_dsa.pub "-o StrictHostKeyChecking=no root@172.16.1.
echo ===============fenfa end==============
echo ""
done
```

脚本说明：

```
   ssh-keygen -t dsa -f /root/.ssh/id_dsa -P "" -q
```

创建密钥，-f指定存放位置，-P 密钥加密的密码 -q 减少信息输出

```
sshpass -p123456 ssh-copy-id -i /root/.ssh/id_dsa.pub "-o StrictHostKeyChecking=no root@172.16.1.
```

这里需要安装一个软件 yum install sshpass -y 用来提供中户密码

ssh-copy-id 命令来分发密钥 -i 指定密钥本地存放的路径

-o StrictHostKeyChecking=no 在登陆其他服务器是不选择yes/no

```
for ip in 8 31 41
```

这里使用for循环来对ip地址进行变化。

系统负载过高，无法连接ssh怎么办?

可以通过修改 进程优先级的方法来是sshd服务能够更多的使用资源，保证sshd的连接

修改进程优先级的方法：

优先系数由系统内核决定，不可更改

nice值可以手动更改，范围是 -20~19

优先级的值越低，优先级越高；优先级的值越高，优先级越低。

所以想调整成最高优先级的话，就将nice值设为-20；想调整成最低优先级的话，将nice值设为19。

1、任务未运行前进行调整

```
shell> nice -n-20 sh clsn.sh   #以最高优先级运行clsn.sh这个脚本
shell> nice -n19 sh clsn.sh    #以最低优先级运行clsn.sh这个脚本
```

2、任务已经开始运行的情况下调整

①方法一

```
# top                           #查看系统当前进程运行情况
> r                             #键入小r
> PID to renice:                #提示输入运行的进程的pid
> Renice PID 23302 to value: #把这个进程的nice值设置为多少,根据需要进行调整
```

② 方法二

```
shell> renice -20 PID    #将进程的nice值改为-20
shell> renice 19 PID     #将进程的nice值改为19
```

查看进程优先级

```
[root@clsn ~]# ps -lef |grep sshd
5 S root      1050     1  0  61 -19 - 16560 poll_s Feb04 ?        00:00:00 /usr/sbin/sshd
4 S root     21018  1050  0  80   0 - 25003 poll_s 17:21 ?        00:00:00 sshd: root@pts/0
0 S root     22852 21022  0  80   0 - 25829 pipe_w 20:47 pts/0    00:00:00 grep sshd
[root@clsn ~]#
```

赞0

如无特殊说明，文章均为本站原创，转载请注明出处

- 转载请注明来源：SSH服务详解
- 本文永久链接地址：https://www.nmtui.com/clsn/lx900.html

该文章由 惨绿少年 发布

惨绿少年Linux www.nmtui.com