链滴

3-ssh 服务

作者：Carey

原文链接：https://ld246.com/article/1610627255617

来源网站：链滴

3 ssh服务

3.1 ssh服务介绍

ssh: secure shell protocol, 22/tcp, 安全的远程登录，实现加密通信，代替传统的 telnet 协议

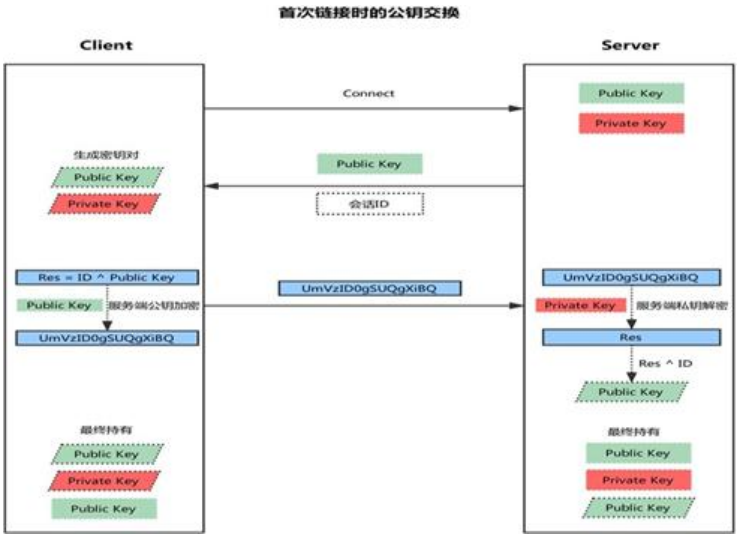
具体的软件实现：

- OpenSSH：ssh协议的开源实现，CentOS 默认安装
- dropbear：另一个ssh协议的开源项目的实现

SSH 协议版本

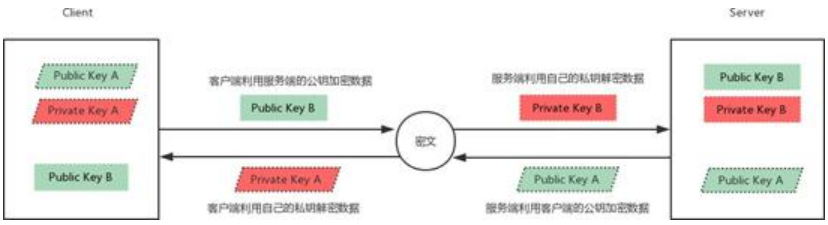
- v1：基于CRC-32做MAC，不安全；man-in-middle
- v2：双方主机协议选择安全的MAC方式，基于DH算法做密钥交换，基于RSA或DSA实现身份认证

3.1.1 公钥交换原理



- 客户端发起链接请求
- 服务端返回自己的公钥，以及一个会话ID（这一步客户端得到服务端公钥）
- 客户端生成密钥对
- 客户端用自己的公钥异或会话ID，计算出一个值Res，并用服务端的公钥加密
- 客户端发送加密后的值到服务端，服务端用私钥解密，得到Res
- 服务端用解密后的值Res异或会话ID，计算出客户端的公钥（这一步服务端得到客户端公钥）
- 最终：双方各自持有三个秘钥，分别为自己的一对公、私钥，以及对方的公钥，之后的所有通讯都被加密

3.1.2 ssh加密通讯原理



3.2 openssh服务

OpenSSH是SSH （Secure SHell） 协议的免费开源实现，一般在各种Linux版本中会默认安装，基于/S结构

Openssh软件相关包：

- openssh
- openssh-clients
- openssh-server

范例：openssh 相关包

```
[10:59:08 root@centos8 ~]#rpm -qa openssh*
openssh-clients-8.0p1-4.el8_1.x86_64
openssh-server-8.0p1-4.el8_1.x86_64
```

openssh-8.0p1-4.el8_1.x86_64

服务器端程序：/usr/sbin/sshd

Unit 文件：/usr/lib/systemd/system/sshd.service

客户端：

- Linux Client: ssh, scp, sftp，slogin
- Windows Client：xshell, MobaXterm,putty, securecrt, sshsecureshellclient

3.2.1 客户端ssh命令

ssh命令是ssh客户端，允许实现对远程系统经验证地加密安全访问

当用户远程连接ssh服务器时，会复制ssh服务器/etc/ssh/ssh_host*key.pub文件中的公钥到客户机的

~/.ssh/know_hosts中。下次连接时，会自动匹配相对应的私钥，不能匹配，将拒绝连接

ssh客户端配置文件： /etc/ssh/ssh_config

主要配置

```
#StrictHostKeyChecking ask #首次登录不显示检查提示StrictHostKeyChecking no
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa
# IdentityFile ~/.ssh/id_ecdsa
# IdentityFile ~/.ssh/id_ed25519
# Port 22
```

范例：禁止首次连接的询问过程

```
[11:08:32 root@centos8 ~]#sed -i.bak '/StrictHostKeyChecking/s@.*@StrictHostKeyChecking
o@' /etc/ssh/ssh_config
```

格式：

```
ssh [user@]host [COMMAND]
ssh [-l user] host [COMMAND]
```

常见选项：

```
-p port：远程服务器监听的端口
-b 指定连接的源IP
-v 调试模式
-C 压缩方式
-X 支持x11转发
-t 强制伪tty分配，如：ssh -t remoteserver1 ssh -t  remoteserver2  ssh remoteserver3
-o option  如：-o StrictHostKeyChecking=no
-i  指定私钥文件路径，实现基于key验证，默认使用文件： ~/.ssh/id_dsa,
~/.ssh/id_ecdsa, ~/.ssh/id_ed25519，~/.ssh/id_rsa等
```

范例：

```
[10:09:59 root@centos7 certs]#ssh -t 192.168.10.81 ssh -t 192.168.10.71
root@192.168.10.81's password:
root@192.168.10.71's password:
Last login: Tue Jan 12 09:07:30 2021 from 192.168.10.1
```

范例：远程执行命令

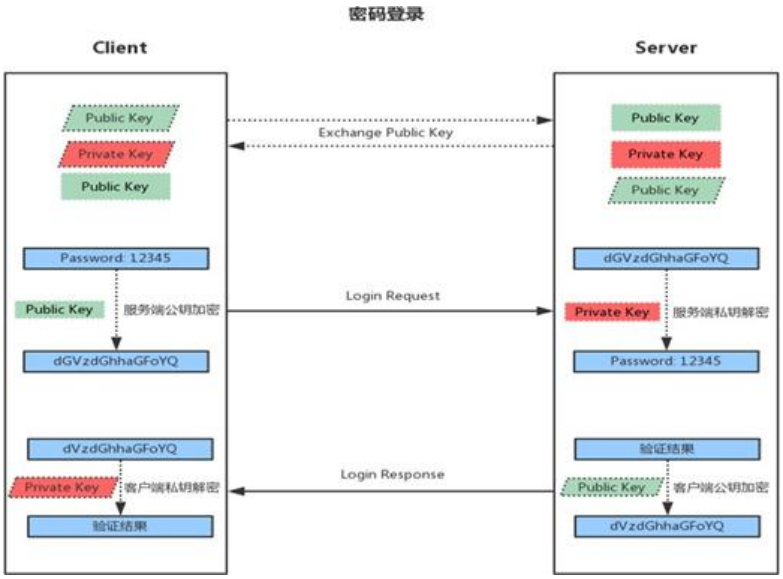
```
[11:12:05 root@centos8 ~]#ssh 192.168.10.71 "sed -i.bak '/StrictHostKeyChecking/s@.*@Stric
HostKeyChecking no@' /etc/ssh/ssh_config"
root@192.168.10.71's password:
```

范例：在远程主机运行本地shell脚本

```
[11:15:22 root@centos8 ~]#hostname -I
192.168.10.81
[11:15:25 root@centos8 ~]#cat test.sh
#!/bin/bash
hostname -I
[11:15:27 root@centos8 ~]#ssh 192.168.10.71 /bin/bash < test.sh
root@192.168.10.71's password:
192.168.10.71
```

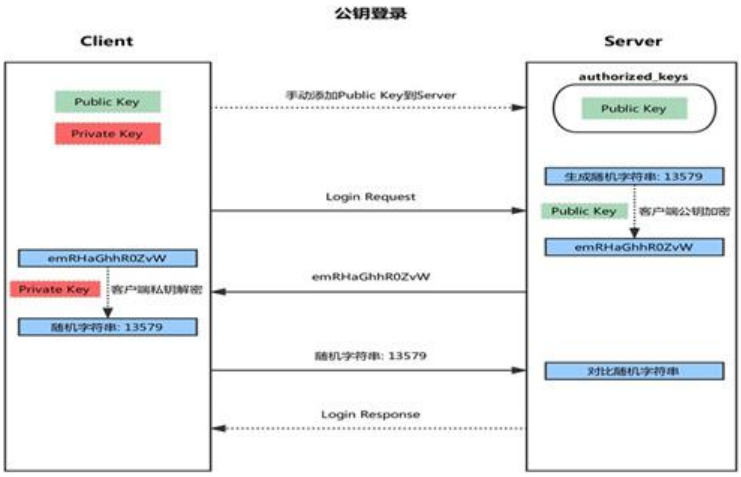
3.2.2 ssh登录验证方式

ssh服务登录的常用验证方式



1. 客户端发起ssh请求，服务器会把自己的公钥发送给用户
2. 用户会根据服务器发来的公钥对密码进行加密
3. 加密后的信息回传给服务器，服务器用自己的私钥解密，如果密码正确，则用户登录成功

基于密钥的登录方式



1. 首先在客户端生成一对密钥（ssh-keygen）
2. 并将客户端的公钥ssh-copy-id 拷贝到服务端
3. 当客户端再次发送一个连接请求，包括ip、用户名
4. 服务端得到客户端的请求后，会到authorized_keys中查找，如果有响应的IP和用户，就会随机生一个字符串，例如：magedu
5. 服务端将使用客户端拷贝过来的公钥进行加密，然后发送给客户端
6. 得到服务端发来的消息后，客户端会使用私钥进行解密，然后将解密后的字符串发送给服务端
7. 服务端接受到客户端发来的字符串后，跟之前的字符串进行对比，如果一致，就允许免密码登录

3.2.3 实现基于密钥的登录方式

在客户端生成密钥对

```
ssh-keygen -t rsa [-P 'password'] [-f “~/.ssh/id_rsa"]
```

把公钥文件传输至远程服务器对应用户的家目录

```
ssh-copy-id [-i [identity_file]] [user@]host
```

重设私钥口令

```
ssh-keygen -p
```

验证代理（authentication agent）保密解密后的密钥，口令就只需要输入一次，在GNOME中，代被自动提供给root用户

```
#启用代理
ssh-agent bash
#钥匙通过命令添加给代理
ssh-add
```

在SecureCRT或Xshell实现基于key验证

在SecureCRT工具—>创建公钥—>生成Identity.pub文件

转化为openssh兼容格式（适合SecureCRT，Xshell不需要转化格式），并复制到需登录主机上相应件authorized_keys中,注意权限必须为600，在需登录的ssh主机上执行：

ssh-keygen -i -f Identity.pub >> .ssh/authorized_keys

范例：实现基于key验证

```
[11:20:21 root@centos7 ~]#ssh-keygen
#下面的id_rsa是私钥id_rsa.pub是公钥
[11:23:48 root@centos7 ~]#ll .ssh/
total 12
-rw------- 1 root root 1675 Jan 12 11:23 id_rsa
-rw-r--r-- 1 root root  394 Jan 12 11:23 id_rsa.pub
-rw-r--r-- 1 root root  175 Jan 11 19:28 known_hosts

[11:24:19 root@centos7 ~]#ssh-copy-id root@192.168.10.81
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are
lready installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to
nstall the new keys
root@192.168.10.81's password: #输入远程用户密码

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'root@192.168.10.81'"
and check to make sure that only the key(s) you wanted were added.

[11:29:11 root@centos8 ~]#ll .ssh
total 16
-rw------- 1 root root  394 Jan 12 11:24 authorized_keys
-rw------- 1 root root 2602 Jan 11 14:06 id_rsa
-rw-r--r-- 1 root root  566 Jan 11 14:06 id_rsa.pub
-rw-r--r-- 1 root root  175 Jan  9 09:11 known_hosts

[11:31:21 root@centos7 ~]#ssh 192.168.10.81

[11:32:14 root@centos7 ~]#scp /etc/fstab 192.168.10.81:/data
fstab                 100%  150   123.2KB/s   00:00

#对私钥加密
[11:35:47 root@centos7 ~]#ssh-keygen -p
Enter file in which the key is (/root/.ssh/id_rsa):  #回车，选择默认文件
Enter new passphrase (empty for no passphrase):      #输入密码
Enter same passphrase again:                         #确认密码
Your identification has been saved with the new passphrase.
[11:36:05 root@centos7 ~]#ssh 192.168.10.81
Enter passphrase for key '/root/.ssh/id_rsa': 输入加密的密码
Activate the web console with: systemctl enable --now cockpit.socket
Last login: Tue Jan 12 11:35:06 2021 from 192.168.10.71

#启用ssh代理，之后第一次输入私钥密码后以后不用输入
[11:37:24 root@centos7 ~]#ssh-agent bash
[11:37:31 root@centos7 ~]#ps aux | grep agent
root       1686  0.0  0.0  72472   776 ?        Ss   11:37   0:00 ssh-agent bash
root       1700  0.0  0.0 112812   972 pts/0    R+   11:37   0:00 grep --color=auto agent
[11:37:41 root@centos7 ~]#ssh-add
```

```
Enter passphrase for /root/.ssh/id_rsa:
Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
[11:37:58 root@centos7 ~]#ssh 192.168.10.81
Activate the web console with: systemctl enable --now cockpit.socket
Last login: Tue Jan 12 11:36:19 2021 from 192.168.10.71
```

范例：基于key验证实现批量主机管理

```
[11:42:10 root@centos8 ~]#cat hosts.txt
192.168.10.81
192.168.10.71
[11:42:21 root@centos8 ~]#for i in `cat hosts.txt`;do ssh $i hostname -I;done
192.168.10.81
192.168.10.71
```

范例：实现xshell的基于key验证

先生成用户密钥，在导出公钥，上传至服务器

```
[11:54:49 root@centos8 ~]#cat id_rsa_2048.pub >.ssh/authorized_keys
```

3.2.4 其他ssh客户端工具

3.2.4.1 scp命令

```
scp [options] SRC... DEST/
```

方式：

```
scp [options] [user@]host:/sourcefile /destpath
scp [options] /sourcefile [user@]host:/destpath
scp [options] [user@]host1:/sourcetpath  [user@]host2:/destpath
```

常用选项：

```
-C  压缩数据流
-r  递归复制
-p  保持原文件的属性信息
-q  静默模式
-P PORT 指明remote host的监听的端口
```

3.2.4.2 rsync命令

rsync工具可以基于ssh和rsync协议实现高效率的远程系统之间复制文件，使用安全的shell连接做为输方式，比scp更快，基于增量数据同步，即只复制两方不同的文件，此工具来自于rsync包

注意：通信两端主机都需要安装 rsync软件

```
rsync  -av /etc server1:/tmp      #复制目录和目录下文件
rsync  -av  /etc/  server1:/tmp   #只复制目录下文件
```

常用选项：

-n 模拟复制过程
-v 显示详细过程
-r 递归复制目录树
-p 保留权限
-t 保留修改时间戳
-g 保留组信息
-o 保留所有者信息
-l 将软链接文件本身进行复制（默认）
-L 将软链接文件指向的文件复制
-u 如果接收者的文件比发送者的文件较新，将忽略同步
-z 压缩，节约网络带宽
-a 存档，相当于-rlptgoD，但不保留ACL（-A）和SELinux属性（-X）
--delete 源数据删除，目标数据也自动同步删除

范例：

[14:18:25 root@centos8 ~]#rsync -auv --delete /data/fstab 192.168.10.71:/root

3.2.4.3 sftp命令

交互式文件传输工具，用法和传统的ftp工具相似，利用ssh服务实现安全的文件上传和下载使用ls cd kdir rmdir pwd get put等指令，可用？或help获取帮助信息

sftp [user@]host
sftp> help

3.2.5 ssh高级应用

SSH 会自动加密和解密所有 SSH 客户端与服务端之间的网络数据。此外，SSH 还能够将其他 TCP 口的网络数据通过 SSH 连接来转发，并且自动提供了相应的加密及解密服务。这一过程也被叫做“道”（tunneling），这是因为 SSH 为其他 TCP 链接提供了一个安全的通道来进行传输而得名。例，Telnet，SMTP，LDAP 这些 TCP 应用均能够从中得益，避免了用户名，密码以及隐私信息的明文输。而与此同时，如果工作环境中的防火墙限制了一些网络端口的使用，但是允许 SSH 的连接，也够通过将 TCP 端口转发来使用 SSH 进行通讯

SSH 端口转发能够提供两大功能：

- 加密 SSH Client 端至 SSH Server 端之间的通讯数据
- 突破防火墙的限制完成一些之前无法建立的 TCP 连接

3.2.5.1 SSH 本地端口转发

SSH本地端口转发

ssh -L localport:remotehost:remotehostport sshserver

选项：

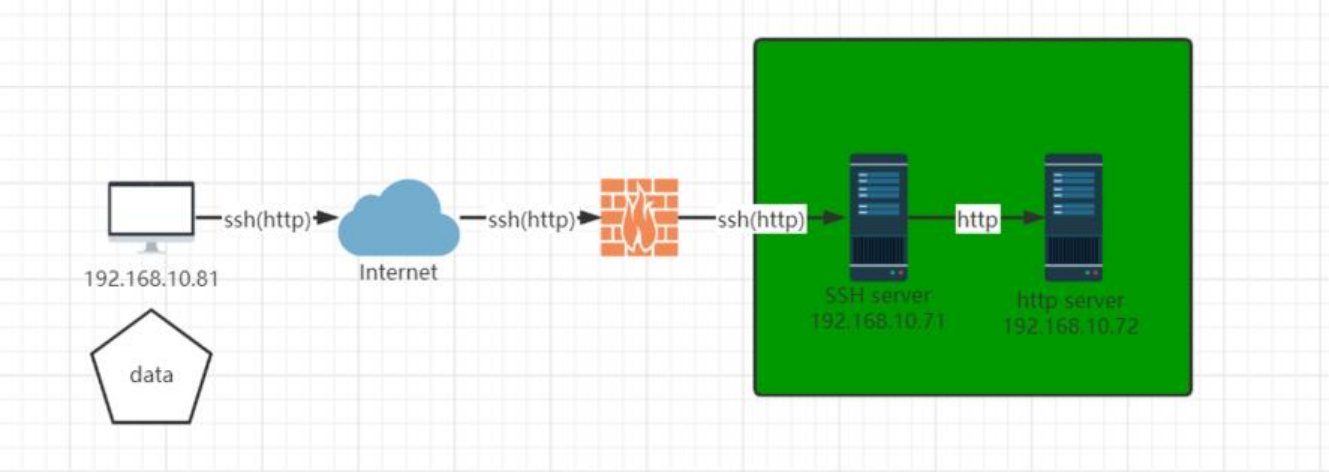
-f 后台启用
-N 不打开远程shell，处于等待状态
-g 启用网关功能

范例：

#当访问本机的9527的端口时，被加密后转发到sshsrv的ssh服务，再解密被转发到telnetsrv:23
#data<-->localhost:9527 <-->localhost:XXXXX<-->sshsrv:22<-->sshsrv:YYYYY<-->telnetsrv:3

ssh -L 9527:telnetsrv:23 -Nfg sshsrv
telnet 127.0.0.1 9527

范例：本地端口转发



#在192.168.10.81操作
[19:23:28 root@centos8 ~]#hostname -I
192.168.10.81
[19:23:30 root@centos8 ~]#ssh -fNL 9527:192.168.10.72:80 192.168.10.71
[19:25:08 root@centos8 ~]#curl 127.0.0.1:9527
zhangzhuo

3.3.5.2 SSH远程端口转发

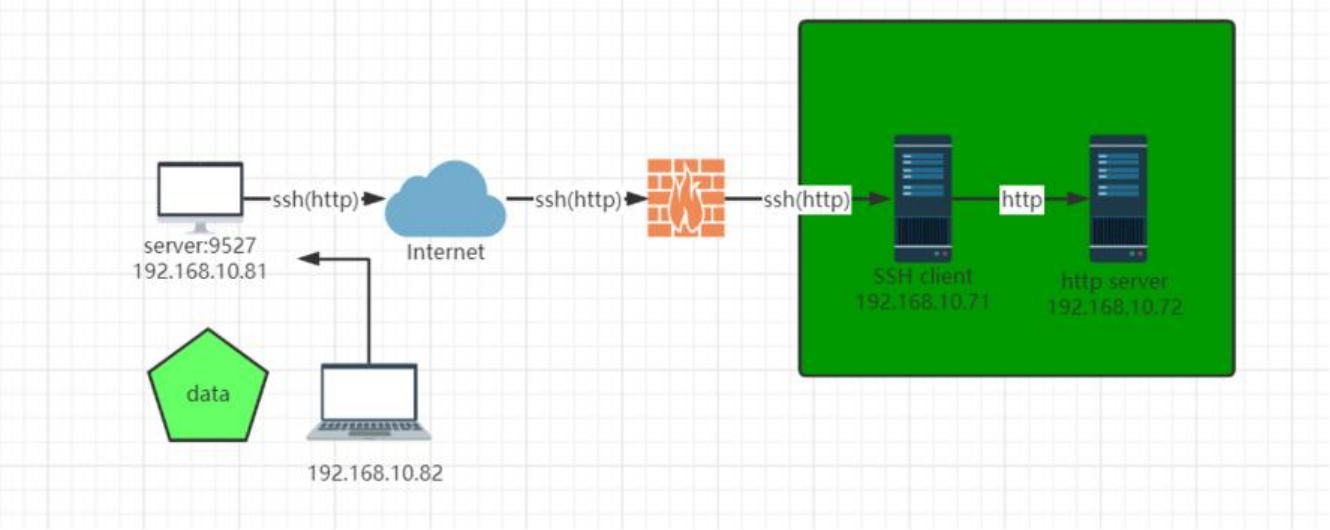
ssh -R sshserverport:remotehost:remotehostport sshserver

示例：

#让sshsrv侦听9527端口的访问，如有访问，就加密后通过ssh服务转发请求到本机ssh客户端，再由机解 密后转发到telnetsrv:23
#Data<-->sshsrv:9527<-->sshsrv:22<-->localhost:XXXXX<-->localhost:YYYYY<-->telnetsrv:2

ssh -R 9527:telnetsrv:23 -Nf sshsrv

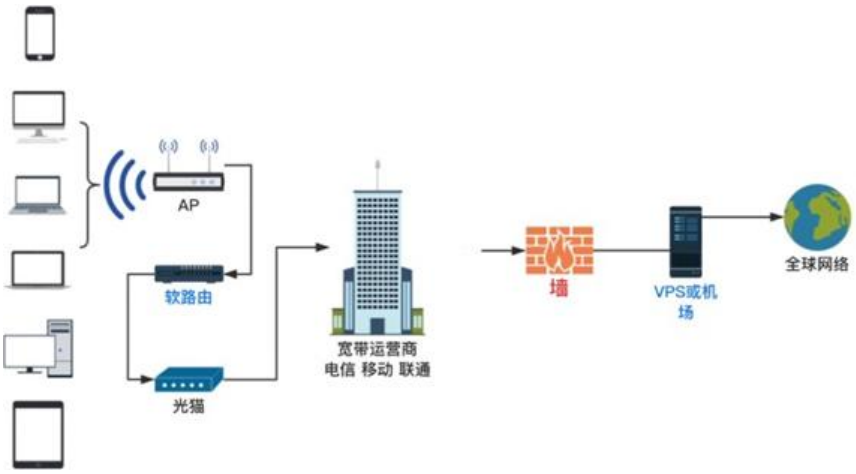
范例：远程端口转发并实现网关功能



#在192.168.10.81执行
[19:32:27 root@centos8 ~]#hostname -I
192.168.10.81
#必须先打开gateway功能，否则无法打开所有IP对应端口
[19:39:20 root@centos8 ~]#vim /etc/ssh/sshd_config
GatewayPorts yes
[19:40:47 root@centos8 ~]#systemctl restart sshd

#在192.168.10.71执行
[19:28:04 root@centos7 ~]#ssh -fNgR 9527:192.168.10.72:80 192.168.10.81
#在192.168.10.81执行
[root@localhost ~]# curl 192.168.10.81:9527
zhangzhuo

3.3.5.3 SSH动态端口转发

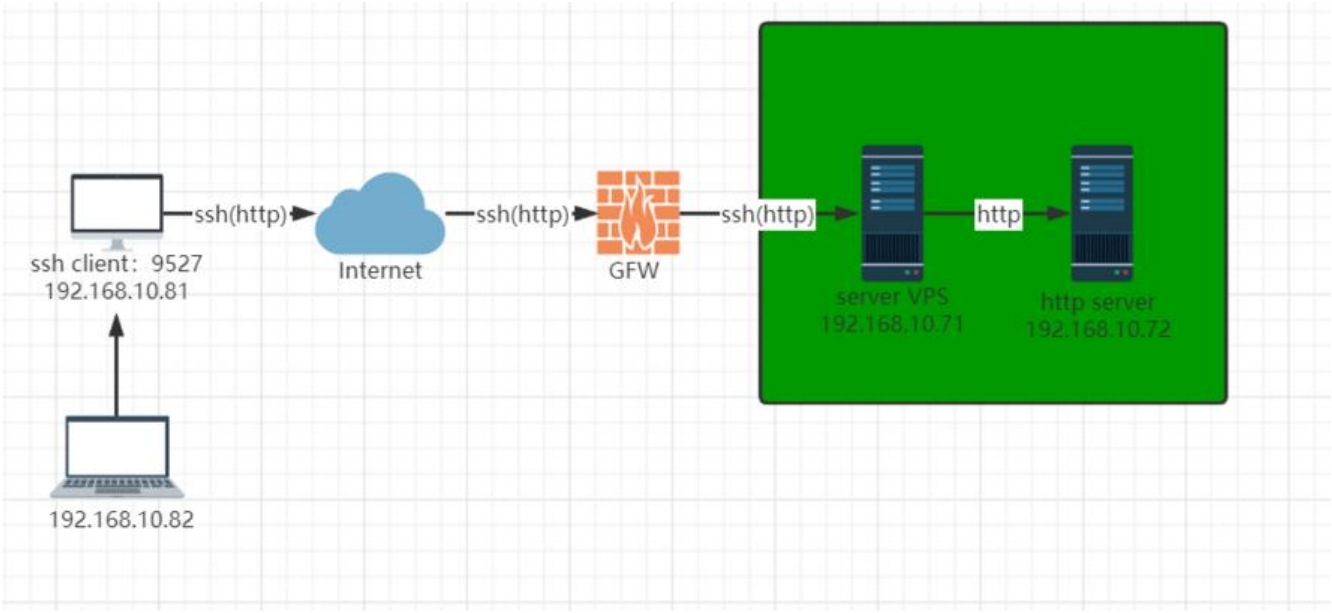


#当用firefox访问internet时，本机的1080端口做为代理服务器，firefox的访问请求被转发到sshserve上，由sshserver替之访问internet

ssh -D 1080 root@sshserver -fNg

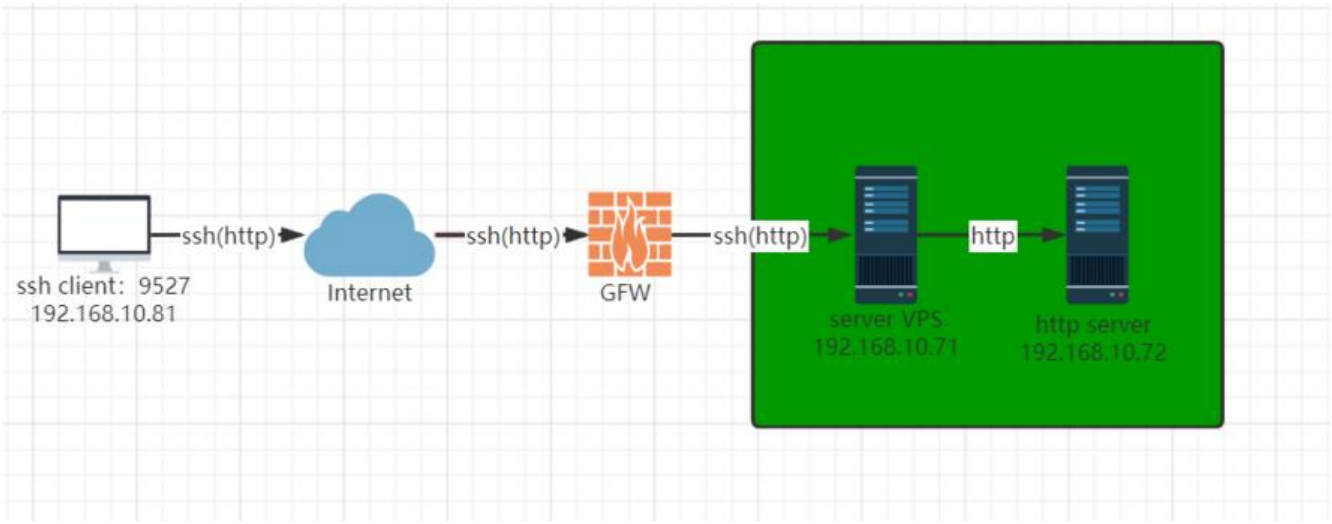
#在本机firefox设置代理socket proxy:127.0.0.1:1080
curl --socks5 127.0.0.1:1080 http://www.google.com

范例：动态端口转发实现科学上网方式1




```
#在192.168.10.8执行
[19:49:06 root@centos8 ~]#ssh -fND 9527 192.168.10.71
[19:56:04 root@centos8 ~]#curl --socks5 127.0.0.1:9527 http://192.168.10.72
zhangzhuo
#这种方式只能实现本机上网
```

范例：动态端口转发实现科学上网方式2




```
#在192.168.10.71执行
[19:48:52 root@centos7 ~]#ssh -gfND 9527 192.168.10.71
#在192.168.10.81执行
[20:02:19 root@centos8 ~]#curl --socks5 192.168.10.71:9527 http://192.168.10.72
zhangzhuo
#这种可以实现其他主机也借助这个上网
```

火狐浏览器可以设置代理进行上网

3.2.6 ssh服务器配置

服务器端：sshd

服务器端的配置文件: /etc/ssh/sshd_config

服务器端的配置文件帮助：man 5 sshd_config

常用参数:

```
Port ListenAddress ip
LoginGraceTime 2m
PermitRootLogin yes       #默认ubuntu不允许root远程ssh登录
StrictModes yes           #检查.ssh/文件的所有者，权限等
MaxAuthTries   6          #pecifies the maximum number of authenticationattempts permitted
er connection.  Once the number of failures reaches halfthis value, additional failures are logg
d.  The default is 6.
MaxSessions  10           #同一个连接最大会话
PubkeyAuthentication yes   #基于key验证
PermitEmptyPasswords no    #空密码连接
PasswordAuthentication yes #基于用户名和密码连接
GatewayPorts no
ClientAliveInterval 10    #单位:秒
ClientAliveCountMax 3     #默认3
```

```
UseDNS yes                #提高速度可改为no
GSSAPIAuthentication yes  #提高速度可改为no
MaxStartups               #未认证连接最大值，默认值10
Banner /path/file

#以下可以限制可登录用户的办法：
AllowUsers user1 user2 user3
DenyUsers user1 user2 user3
AllowGroups g1 g2
DenyGroups  g1 g2
```

范例：设置ssh空闲60s自动注销

```
Vim  /etc/ssh/sshd_config
ClientAliveInterval   60
ClientAliveCountMax   0

Service sshd restart #注意：新开一个连接才有效
```

范例：解决ssh登录缓慢的问题

```
vim /etc/ssh/sshd_config
UseDNS no
GSSAPIAuthentication no

systemctl restart sshd
```

范例：在 ubuntu 上启用root 远程ssh登录

```
#修改sshd服务配置文件
vim  /etc/ssh/sshd_config
#PermitRootLogin prohibit-password  注释掉此行
PermitRootLogin yes  修改为下面形式

systemctl restart sshd
```

ssh服务的最佳实践

- 建议使用非默认端口
- 禁止使用protocol version 1
- 限制可登录用户
- 设定空闲会话超时时长
- 利用防火墙设置ssh访问策略
- 仅监听特定的IP地址
- 基于口令认证时，使用强密码策略，比如：tr -dc A-Za-z0-9_ < /dev/urandom | head -c 12|xargs
- 使用基于密钥的认证
- 禁止使用空密码
- 禁止root用户直接登录
- 限制ssh的访问频度和并发在线数
- 经常分析日志

3.4 ssh其他相关工具

3.4.1 挂载远程ssh目录sshfs

由EPEL源提供，目前CentOS8 还没有提供，可以利用ssh协议挂载远程目录

```
[15:14:09 root@centos7 ~]#yum install fuse-sshfs
[15:16:22 root@centos7 ~]#sshfs 192.168.10.81:/data /mnt/ -o nonempty
[15:16:41 root@centos7 ~]#df
Filesystem              Size  Used Avail Use% Mounted on
devtmpfs                475M     0  475M   0% /dev
tmpfs                   487M     0  487M   0% /dev/shm
tmpfs                   487M  7.7M  479M   2% /run
tmpfs                   487M     0  487M   0% /sys/fs/cgroup
/dev/mapper/centos-root   17G  2.8G   15G  17% /
/dev/sda1               1014M  178M  837M  18% /boot
tmpfs                    98M     0   98M   0% /run/user/0
192.168.10.81:/data       17G  5.8G   12G  34% /mnt
```

注意：如果被挂载的目录非空需要加 -o nonempty参数

3.4.2 自动登录ssh工具sshpass

由EPEL源提供，ssh登陆不能在命令行中指定密码。sshpass的出现，解决了这一问题。sshpass用于非 交互SSH的密码验证，一般用在sh脚本中，无须再次输入密码（本机known_hosts文件中有的主机 能生效）。它允许你用 -p 参数指定明文密码，然后直接登录远程服务器，它支持密码从命令行、文 、环境变量中读取。

格式：

```
sshpass [option] command parameters
```

常见选项：

```
-p password #后跟密码它允许你用 -p 参数指定明文密码，然后直接登录远程服务器
-f filename #后跟保存密码的文件名，密码是文件内容的第一行。
-e #将环境变量SSHPASS作为密码
```

范例：

```
[15:15:05 root@centos8 ~]#yum -y install sshpass
[15:19:52 root@centos8 ~]#rpm -ql sshpass
/usr/bin/sshpass
/usr/lib/.build-id
/usr/lib/.build-id/1f
/usr/lib/.build-id/1f/c5d6cf03500df846a1a801aab749f478845a4d
/usr/share/doc/sshpass
/usr/share/doc/sshpass/AUTHORS
/usr/share/doc/sshpass/COPYING
/usr/share/doc/sshpass/ChangeLog
/usr/share/doc/sshpass/NEWS
/usr/share/man/man1/sshpass.1.gz
```

```
[15:20:08 root@centos8 ~]#sshpass -p 123456 ssh 192.168.10.71

[15:21:07 root@centos8 ~]#sshpass -p 123456 ssh 192.168.10.71 hostname -I
192.168.10.71

[15:22:14 root@centos8 ~]#cat pass.txt
123456
[15:22:19 root@centos8 ~]#sshpass -f pass.txt ssh 192.168.10.71

[15:24:18 root@centos8 ~]#export SSHPASS=123456;sshpass -e ssh 192.168.10.71
```

范例：批量修改多台主机的root密码为随机密码

```
[16:00:10 root@centos8 ~]#cat chanage_root_password.sh
#!/bin/bash
#
#****************************************************************
#Author:zhangzhuo
#QQ: 1191400158
#Date: 2021-01-12
#FileName：  chanage_root_password.sh
#URL: https://www.zhangzhuo.ltd
#Description：  The test script
#Copyright (C): 2021 All rights reserved
#****************************************************************
rpm -q sshpass &>/dev/null || yum -y install sshpass
export SSHPASS=123456
NET=192.168.10
for i in {1..254};do
{
PASS=`openssl rand -base64 9`
sshpass -e ssh -o StrictHostKeyChecking=no $NET.$i "echo $PASS | passwd --stdin root &>/d
v/null" &>/dev/null
echo $NET.$i:$PASS >> host.txt
}&
done
wait
```

范例：批量部署多台主机key验证脚本1

```
[16:31:17 root@centos8 ~]#cat sshpass1.sh
#!/bin/bash
NET=192.168.10
PASS=123456
ssh-keygen -P "" -f /root/.ssh/id_rsa &>/dev/null
rpm -q sshpass &>/dev/null || yum -y install sshpass &>/dev/null
for i in {1..100};do
{
sshpass -p $PASS ssh-copy-id -o StrictHostKeyChecking=no -i /root/.ssh/id_rsa.pub $NET.$i
>/dev/null
}&
done
wait
```

范例：批量部署多台主机key验证脚本2

```
[16:37:27 root@centos8 ~]#cat sshpass2.sh
#!/bin/bash
HOSTS="
192.168.10.71
192.168.10.72
"
PASS=123456
ssh-keygen -P "" -f /root/.ssh/id_rsa &>/dev/null
rpm -q sshpass &>/dev/null || yum install -y sshpass &>/dev/null
for i in $HOSTS;do
{
sshpass -p $PASS ssh-copy-id -o StrictHostKeyChecking=no -i /root/.ssh/id_rsa.pub $i &>/de
/null
}&
done
wait
```

3.4.3 轻量级自动化运维工具pssh

EPEL源中提供了多个自动化运维工具

- pssh：基于python编写，可在多台服务器上执行命令的工具，也可实现文件复制，提供了基于ssh scp的多个并行工具，项目：http://code.google.com/p/parallel-ssh/
- pdsh：Parallel remote shell program，是一个多线程远程shell客户端，可以并行执行多个远程机上的命令。 可使用几种不同的远程shell服务，包括rsh，Kerberos IV和ssh，项目：https://pdsh.ooglecode.com/
- mussh：Multihost SSH wrapper，是一个shell脚本，允许使用命令在多个主机上通过ssh执行命。 可使用ssh-agent和RSA/DSA密钥，以减少输入密码，项目：http://www.sourceforge.net/projects/mussh

pssh 命令选项如下：

```
-H：主机字符串，内容格式”[user@]host[:port]”
-h file：主机列表文件，内容格式”[user@]host[:port]”
-A：手动输入密码模式
-i：每个服务器内部处理信息输出
-l：登录使用的用户名
-p：并发的线程数【可选】
-o：输出的文件目录【可选】
-e：错误输出文件【可选】
-t：TIMEOUT 超时时间设置，0无限制【可选】
-O：SSH的选项
-P：打印出服务器返回信息
-v：详细模式
--version：查看版本
```

范例：

```
#默认使用ssh的key认证，通过 -A选项，使用密码认证批量执行指令
[16:41:11 root@centos8 ~]#pssh -H "192.168.10.71" -A hostname
#输出信息
[16:41:35 root@centos8 ~]#pssh -H zhang@192.168.10.71 -A -i hostname
#通过pssh批量关闭seLinux
```

```
[16:53:23 root@centos8 ~]#pssh -H root@192.168.10.71  -A -i 'sed -i "s/^SELINUX=.*/SELIN
X=disabled/" /etc/selinux/config'
#多台主机
[16:53:49 root@centos8 ~]#pssh -H "192.168.10.71 192.168.10.72" -i -A hostname
#多台主机
[16:55:23 root@centos8 ~]#cat host.txt
192.168.10.71
192.168.10.72
[16:55:27 root@centos8 ~]#pssh -h host.txt -i -A echo $HOSTNAME
#将标准错误和标准正确重定向分别保存至本地主机的/data/stdout和/data/stderr目录下
[16:58:18 root@centos8 ~]#pssh -H 192.168.10.71 -o /data/stdout -e /data/stderr -i -A "host
ame"
#变量需要加单引号引起来
[16:59:54 root@centos8 ~]#pssh -h host.txt -A -i  'echo $HOSTNAME'
#*需要用双或单引号引起来
[17:01:10 root@centos8 ~]#pssh -h host.txt -A -i 'ls /data/*'
```

pscp.pssh命令

pscp.pssh功能是将本地文件批量复制到远程主机

pscp [-vAr] [-h hosts_file] [-H [user@]host[:port]] [-l user] [-p par] [-o outdir] [-e errdir] [-t tim
out] [-O options] [-x args] [-X arg] local remote

pscp-pssh选项

-v 显示复制过程
-r 递归复制目录

范例：

```
#将本地curl.sh 复制到/data/目录
[17:05:46 root@centos8 ~]#pscp.pssh -H 192.168.10.71 -A /root/curl.sh /data/

#将本地多个文件批量复制到/app/目录
[17:06:32 root@centos8 ~]#pscp.pssh -H 192.168.10.71 -A /root/chanage_root_password.sh /
oot/push_ssh_key.sh /data/

#将本地目录批量复制到/app/目录
[17:09:16 root@centos8 ~]#pscp.pssh -H 192.168.10.71 -A -r /data/ /data/
```

pslurp命令

pslurp功能是将远程主机的文件批量复制到本地

pslurp [-vAr] [-h hosts_file] [-H [user@]host[:port]] [-l user] [-p par][-o outdir] [-e errdir] [-t ti
eout] [-O options] [-x args] [-X arg] [-L localdir] remote local（本地名）

pslurp选项

-L 指定从远程主机下载到本机的存储的目录，local是下载到本地后的名称
-r 递归复制目录

范例：

```
#批量下载目标服务器的passwd文件至/app下，并更名为user
[17:12:24 root@centos8 ~]#pslurp -H 192.168.10.71 -A -L /app /etc/passwd user

[17:14:26 root@centos8 ~]#pslurp -h host.txt -A -L  /data/ /etc/redhat-release version
Warning: do not enter your password if anyone else has superuser
privileges or access to your account.
Password:
[1] 17:14:33 [SUCCESS] 192.168.10.71
[2] 17:14:34 [SUCCESS] 192.168.10.72
[17:14:34 root@centos8 ~]#tree /data/
/data/
├── 192.168.10.71
│   └── version
└── 192.168.10.72
└── version

2 directories, 2 files
```

3.5 dropbear

由Matt Johnston所开发的Secure Shell软件。Dropbear是一个相对较小的SSH服务器和客户端。它行在一个基于POSIX的各种平台。 Dropbear是开源软件，在麻省理工学院式的许可证。 Dropbear特别有用的“嵌入”式的Linux（或其他Unix）系统，如无线路由器，期望在存储器与运算能力有限情况 下取代OpenSSH，尤其是嵌入式系统

官网：http://matt.ucc.asn.au/dropbear/dropbear.html

范例：编译安装dropbear

```
#安装相关包：
[17:16:21 root@centos8 ~]#yum install gcc zlib-devel
#下载
[17:19:00 root@centos8 ~]#wget https://matt.ucc.asn.au/dropbear/dropbear-2020.80.tar.bz2
[17:19:31 root@centos8 ~]#tar xvf dropbear-2020.80.tar.bz2
#编译安装
[17:19:50 root@centos8 dropbear-2020.80]#less INSTALL README
[17:21:06 root@centos8 dropbear-2020.80]#./configure --prefix=/apps/dropber --sysconfdir=
etc/dropber
[17:23:48 root@centos8 dropbear-2020.80]#make PROGRAMS="dropbear dbclient dropbear
ey dropbearconvert scp"
[17:23:48 root@centos8 dropbear-2020.80]#make PROGRAMS="dropbear dbclient dropbear
ey dropbearconvert scp" install
[17:23:17 root@centos8 dropbear-2020.80]#tree /apps/
/apps/
└── dropber
├── bin
│   ├── dbclient
│   ├── dropbearconvert
│   ├── dropbearkey
│   └── scp
├── sbin
│   └── dropbear
└── share
└── man
```

```
├── man1
│   ├── dbclient.1
│   ├── dropbearconvert.1
│   └── dropbearkey.1
└── man8
    └── dropbear.8
#配置PATH变量
[17:24:55 root@centos8 ~]#echo 'PATH=/apps/dropbear/sbin:/apps/dropbear/bin:$PATH' >/
tc/profile.d/dropber.sh
#生成私钥
[17:31:21 root@centos8 ~]#mkdir /etc/dropbear
[17:31:46 root@centos8 ~]#dropbearkey -t rsa -f /etc/dropbear/dropbear_rsa_host_key -s 20
8
[17:32:37 root@centos8 ~]#dropbearkey -t dss -f /etc/dropbear/dropbear_dsa_host_key
#启动ssh服务
[17:32:38 root@centos8 ~]#dropbear -p :2222 -FE  #前台运行，相当于sshd
[17:34:56 root@centos8 ~]#dropbear -p :2222      #后台运行
客户端访问
[17:09:40 root@centos7 ~]#ssh -p 2222 192.168.10.81
[17:35:51 root@centos8 ~]#dropbear -p 2222 192.168.10.81  #相当于ssh
```