SSH 实现原理及实践

2015/2/11
renyl

1 概述

SSH 全称 Secure shell，顾名思义其意思就是非常安全的 shell。SSH 的主要目的是用来取代传统的 telnet 和 R 系列命令（rlogin,rsh,rexec 等），通过加密的方式来远程登录和远程执行命令，从而防止由于网络监听而出现的密码泄漏等情况。

SSH 是一种加密协议，不仅在登陆过程中对密码进行加密传送，而且对登陆后执行的命令的数据也进行加密，这样即使别人在网络上监听并截获了你的数据包，也看不到其中的内容。

2 SSH 工作原理

SSH 在整个通讯过程中为了实现安全连接，服务器端与客户端需要经历如下五个阶段：

2.1 版本协商阶段

具体步骤如下：

1) 服务器打开端口 22，等待客户端连接。

2) 客户端向服务器端发起 TCP 初始连接请求。

3) TCP 连接建立后，服务器向客户端发送第一个报文，包括版本标志字符串，格式为“SSH-<主协议版本号>.<次协议版本号>-<软件版本号>”，协议版本号由主版本号和次版本号组成，软件版本号主要是为调试使用。

4) 客户端收到报文后，首先解析该数据包，通过与服务器端的协议版本号进行对比，决定要使用的协议版本号。如果服务器端的协议版本号比自己的低，且客户端能支持服务器端的低版本，就使用服务器端的低版本协议号，否则使用自己的协议版本号。

5) 然后，客户端回应服务器一个报文，包含了客户端决定使用的协议版本号。

6) 服务器比较客户端发来的版本号，如果服务器支持该版本，则使用该版本，并进入密钥和算法协商阶段，否则，版本协商失败，服务器端断开 TCP 连接。

2.2 密钥和算法协商阶段

具体步骤如下：

1) 服务器端和客户端分别发送算法协商报文给对端，报文中包含自己支持的公钥算法列表、加密算法列表、MAC（Message Authentication Code，消息验证码）算法列表、压缩算法列表等。

2) 服务器端和客户端根据对端和本端支持的算法列表得出最终使用的算法。任何一种算法协商失败，都会导致服务器端和客户端的算法协商过程失败，服务器将断开与客户端的连接。

3) 服务器端和客户端利用 DH 交换(Diffie-Hellman Exchange)算法、主机密钥对等参数，生成会话密钥和会话 ID，并完成客户端对服务器身份的验证。

通过以上步骤，服务器端和客户端就取得了相同的会话密钥和会话 ID。对于后续传输的数据，两端都会使用会话密钥进行加密和解密，保证了数据传送的安全。会话 ID 用来标识一个 SSH 连接，在认证阶段，会话 ID 还会用于两端的认证。

2.3 认证阶段

SSH 提供两种认证方法：

1) password 认证：
   a) 利用 AAA（Authentication、Authorization、Accounting，认证、授权和计费）对客户端身份进行认证。

   b) 客户端向服务器发出 password 认证请求，将用户名和密码加密后发送给服务器；

   c) 服务器将该信息解密后得到用户名和密码的明文，通过本地认证或远程认证验证用户名和密码的合法性，并返回认证成功或失败的消息。

   d) 如果远程认证服务器要求用户进行二次密码认证，则会在发送给服务器端的认证回应消息中携带一个提示信息，该提示信息被服务器端透传给客户端，由客户端输出并要求用户再次输入一个指定类型的密码，当用户提交正确的密码并成功通过认证服务器的验证后，服务器端才会返回认证成功的消息。

2) publickey 认证：
   a) 采用数字签名的方法来认证客户端。目前，可以利用 RSA 和 DSA 两种公钥算法实现数字签名。

   b) 客户端发送包含用户名、公钥和公钥算法的 publickey 认证请求给服务器端。

   c) 服务器对公钥进行合法性检查，如果不合法，则直接发送失败消息；否则，服务器利用数字签名对客户端进行认证，并返回认证成功或失败的消息。

认证阶段的具体步骤如下：

1) 客户端向服务器端发送认证请求，认证请求中包含用户名、认证方法（password 认证或 publickey 认证）、与该认证方法相关的内容（如：password 认证时，内容为密码）。

2) 服务器端对客户端进行认证，如果认证失败，则向客户端发送认证失败消息，其中包含可以再次认证的方法列表。

3) 客户端从认证方法列表中选取一种认证方法再次进行认证。该过程反复进行，直到认证成功或者认证次数达到上限，服务器关闭连接为止。

2.4 会话请求阶段

1) 认证通过后，客户端向服务器发送会话请求。

2) 服务器等待并处理客户端的请求。

3) 在这个阶段，请求被成功处理后，服务器会向客户端回应 SSH_SMSG_SUCCESS 包，SSH 进入交互会话阶段；否则回应 SSH_SMSG_FAILURE 包，表示服务器处理请求失败或者不能识别请求。

2.5 交互会话阶段

1) 会话请求成功后，连接进入交互会话阶段。

2) 在这个阶段，数据被双向传送。客户端将要执行的命令加密后传给服务器，服务器接收到报文，解密后执行该命令，将执行的结果加密发还给客户端，客户端将接收到的结果解密后显示到终端上。

3 SSH 实践

SSH 协议目前有 SSH1 和 SSH2，SSH2 协议兼容 SSH1。Linux 下的 OpenSSH 是 OpenBSD 组织开发的一款免费的 SSH 软件。接下来介绍 OpenSSH 软件所提供的功能。

安装 OpenSSH 包后，Linux 系统会提供如下几个命令：

命令	说明
ssh	ssh 客户端程序
ssh-add	ssh 代理相关程序，用来向 ssh 代理添加 ssh-key
ssh-agent	ssh 代理程序
ssh-copy-id	使用本地的 ssh-key 授权给远程机器登录
sshd	ssh 服务器端程序
ssh-keygen	ssh-key 生成器

安装 OpenSSH 包后，在/etc/ssh/目录有如下几个文件：

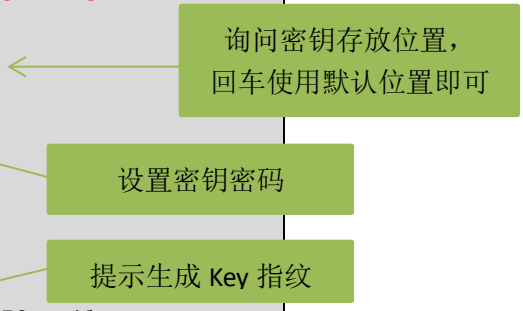
文件	说明
ssh_config	针对客户端的配置文件
sshd_config	针对服务端的配置文件
ssh_host_ecdsa_key	dsa 加密方式的私钥文件
ssh_host_ecdsa_key.pub	dsa 加密方式的共钥文件
ssh_host_rsa_key	rsa 加密方式的私钥文件
ssh_host_rsa_key.pub	rsa 加密方式的共钥文件

3.1 无密码登陆

SSH 提供了 password 和 publickey 两种认证登陆方式。为了实现无密码登录，需要使用 publickey 认证以及 OpenSSH 提供的一些命令来实现，具体步骤如下：

1) 使用 ssh-keygen 命令产生公钥和私钥

```
[root@localhost .ssh]# ssh-keygen -b 1024 -t rsa -C kulong0105@gmail.com
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
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:
45:4f:85:cc:66:28:15:cb:e7:cf:08:0f:55:98:3a:e7 kulong0105@gmail.com
The key's randomart image is:
+--[ RSA 1024]----+
|         .+=.=o  |
```



```
|         .o.+O.     |
|          .+++     |
|          .o+.     |
|         S o+.     |
|            +E+    |
|             o o   |
|                   |
|                   |
+-----------------+
[root@localhost .ssh]#ls
id_rsa  id_rsa.pub
[root@localhost .ssh]#
```

注：

1) -b 1024：采用长度为 1024 字节的公钥/私钥对，最长 4096 字节，一般 1024 和 2048。

2) -t rsa：采用 rsa 加密方式的公钥和私钥对，除了 rsa 还有 dsa 加密方式。

3) -C kulong0105@gmail.com：起到对这个公钥/私钥对的注释和说明作用，一般用邮箱。

说明：

生成的公钥和私钥在用户 home 目录的.ssh 目录下，其中 id_rsa 是私钥，id_rsa.public 是公钥。

2）把公钥上传到需要登录的服务器的对应用户 home 目录的.ssh 目录下

```
[root@localhost .ssh]# scp id_rsa.pub  193.168.220.17:/root/.ssh/authorized_keys
The authenticity of host '193.168.220.17 (193.168.220.17)' can't be established.
ECDSA key fingerprint is dc:c7:b6:db:31:76:0e:e1:e2:e6:36:21:d0:2b:c3:3a.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '193.168.220.17' (ECDSA) to the list of known hosts.
root@193.168.220.17's password:
id_rsa.pub
100%  234     0.2KB/s   00:00
[root@localhost .ssh]# ls
id_rsa  id_rsa.pub  known_hosts
[root@localhost .ssh]# cat known_hosts
193.168.220.17                                                ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBMIKsBVCewhycxt8GpV8vXRUFbo
X/h1bOjkuf22d1ccvqB8AGfYpGdcL0I03lDU5z8acUP2UymKuosgpvpP++dc=
[root@localhost .ssh]#
```

输入服务器的登录密码

该文件表示可进行 ssh 连接的服务器列表，每个列表内容为服务器端/et/ssh/ssh_host_ecdsa_key.pub 的内容。

3）登录到服务器

```
[root@localhost .ssh]# ssh 193.168.220.17
Enter passphrase for key '/root/.ssh/id_rsa':
Last login: Thu Jul 31 17:29:27 2014 from 193.168.249.155
[root@localhost ~]#
```

需要输入创建密钥时输入的密码

说明：

a) 客户端发起 ssh 连接，服务器端在接收到连接请求时，默认会先检查用户 home 目录下的.ssh/authorized_keys 文件，如果该文件存在并且存储着客户端的公钥，那么就使用 publickey 方式认证登录，否则就使用 password 方式认证登录

b) 由于客户端在创建密钥时，使用了密码，所以使用密钥登录到服务器时需要输入密码。

当然，在创建密钥的时候，可以不输入密码，这样在使用密钥登录服务器的时候就不需要输入密码，不过这种方法安全性低。接下来介绍 OpenSSH 提供的一种方法来解决这个问题。

4）管理密钥（在客户端执行）

```
[root@localhost .ssh]# ssh-agent
SSH_AUTH_SOCK=/tmp/ssh-aNYr1ZfiCHk6/agent.31944; export SSH_AUTH_SOCK;
SSH_AGENT_PID=31945; export SSH_AGENT_PID;
echo Agent pid 31945;
[root@localhost .ssh]# eval `ssh-agent`
Agent pid 31947
[root@localhost .ssh]# ssh-add  /root/.ssh/id_rsa
Enter passphrase for /root/.ssh/id_rsa:
Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
[root@localhost .ssh]# ssh 193.168.220.17
Last login: Thu Jul 31 17:38:28 2014 from 193.168.249.155
[root@localhost ~]# exit
logout
Connection to 193.168.220.17 closed.
[root@localhost .ssh]# eval `ssh-agent -k`
Agent pid 31947 killed
[root@localhost .ssh]# ssh 193.168.220.17
Enter passphrase for key '/root/.ssh/id_rsa':
Last login: Fri Aug  1 10:15:06 2014 from 193.168.249.155
[root@localhost ~]#
```

ssh-agent 会打印出两个环境变量，于是使用 eval `ssh-agent`可以设置这两个环境变量。

ssh-add 把私钥托管给 ssh-agent。

登录服务器不需要输入密码了。

使用 ssh-agent -k 可以关闭当前的 ssh-agent 进程。

这样再次登录，又需要输入密码了。

说明：

a) ssh-agent 作用在于管理密钥。

b) ssh-add 用于将密钥加入到 ssh-agent 中，SSH 服务端可以和 ssh-agent 通信获取密钥，这样就不需要用户手工输入密码了。

为了在每次登录之后不需要每次执行 ssh-agent 和 ssh-add 命令，在/root/.bashrc 中添加自动化脚本来实现 ssh-agent 的 daemonize，如下所示：

```
[root@localhost ~]# cat ~/.bashrc
# .bashrc
# User specific aliases and functions
alias rm='rm -i'
alias cp='cp -i'
```

```
alias mv='mv -i'
# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi

if [ -f ~/.agent.env ]; then
        . ~/.agent.env >/dev/null
         var=` ps -ef | grep -v  grep | grep  "${SSH_AGENT_PID}" `
         if [ "${var}" == "" ];then
                eval `ssh-agent |tee ~/.agent.env`
                ssh-add ~/.ssh/id_rsa
         fi
else
        #echo "Starting ssh-agent..."
        eval `ssh-agent |tee ~/.agent.env`
        ssh-add ~/.ssh/id_rsa
fi
```

说明：

a) ~/.agent.env 存放这次会话时可用的 agent 环境变量，供下个会话使用。

b) if ["${var}" == ""];判断语句是检测当前 ssh-agent 进程是否存在，如果不存在就重新打开一个，然后向当前 agent 添加 key。

c) 通过以上设置，只有在第一次登录时的时候输入密码，以后都不需要。

需要注意的是，OpenSSH 还提供了一个命令“ssh-copy-id”，使用该命令将非常方便的实现无密码登录，过程如下：

```
[root@localhost .ssh]# ls
[root@localhost .ssh]# ssh-keygen -b 1024 -t rsa -C kulong0105@gmail.com
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
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:
36:90:08:ba:8d:a9:9a:9d:d4:ec:23:0d:28:86:df:20 kulong0105@gmail.com
The key's randomart image is:
+--[ RSA 1024]----+
|  .              |
| . . . .         |
|.   . o          |
| =     .         |
|=..     S        |
```

重新生成一对密钥

直接按 Enter 键即可

设置密钥密码

```
|E.oo    . .        |
|+o.=o              |
|.+oo+              |
|o o...             |
+-----------------+
[root@localhost .ssh]# ssh-copy-id  193.168.220.17
The authenticity of host '193.168.220.17 (193.168.220.17)' can't be established.
ECDSA key fingerprint is 5d:14:09:b9:f7:cc:4b:72:87:b5:d6:5a:93:d0:b0:6d.
Are you sure you want to continue connecting (yes/no)? yes
/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@193.168.220.17's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh '193.168.220.17'"
and check to make sure that only the key(s) you wanted were added.

[root@localhost .ssh]# ssh 193.168.220.17
Last login: Fri Aug  1 11:31:07 2014 from 10.167.226.165
[root@localhost ~]# file /usr/bin/ssh-copy-id
/usr/bin/ssh-copy-id: POSIX shell script, ASCII text executable
[root@localhost ~]#
```

服务器 IP 地址

输入服务器登录密码

登录不需要输入密码了

说明：

a) 通过上面几个步骤，就可以非常方便的无需密码登录服务器了。

b) 命令 ssh-copy-id 其实为一个 shell 脚本，通过执行一系列操作完成无密码登陆（如：把本地的公钥文件 id_rsa.pub 上传到服务器端的/root/.ssh/目录下，并改名为 authorized_keys 文件。）

另外需要注意的是：

a) 用户 home 目录下的.ssh 目录的权限必须是 700,并且用户的 home 目录也不能给其他用户写权限，否则 ssh 服务器会拒绝登陆。

b) 私钥必须是 600 权限，否则 ssh 服务器也会拒绝用户登录。

c) 如果发生不能登陆的问题，查看服务器上的日志文件/var/log/secure。通常能很快找到不能登陆的原因。

3.2 配置详解

/etc/ssh/ssh_config 配置：

参数	说明
Host	只对能够匹配后面字串的计算机有效。“*”表示所有的计算机。
ForwardAgent	设置连接是否经过验证代理（如果存在）转发给远程计算机。
PasswordAuthentication	设置是否使用口令验证
BatchMode no	如果设为“yes”，passphrase/password（交互式输入口令）的提示将被禁止。当不能交互式输入口令的时候，这个选项对脚本文件和批处理任务十分有用。
IdentityFile	设置从哪个文件读取用户的 RSA 安全验证标识。
Port	设置连接到远程主机的端口。

/etc/ssh/sshd_config 配置：

参数	说明
Port	设置 sshd 监听的端口号
ListenAddress	设置 sshd 服务器绑定的 IP 地址
HostKey	设置包含计算机私人密匙的文件
ServerKeyBits	定义服务器密匙的位数
LoginGraceTime(s)	如果用户不能成功登录,在切断连接之前服务器需要等待的时间
PermitRootLogin	设置 root 能不能用 ssh 登录
PasswordAuthentication	设置是否允许口令验证。
PermitEmptyPasswords	设置是否允许用口令为空的帐号登录

4 参考文献

1) http://www.ibm.com/developerworks/cn/linux/security/openssh/part1/index.html#10
2) http://www.ibm.com/developerworks/cn/linux/security/openssh/part2/
3) http://smilejay.com/2011/10/sshd_config/