

《信息安全综合实践》实验报告

实验名称: SSH 实验

姓名: 黎锦灏 学号: 518021910771 邮箱: 1197993966@qq.com 实验时长: 75 分钟

一、实验目的

1. 加深对密码算法使用的理解
2. 学习 OpenSSH 的相关命令及应用 (<http://www.openssh.com/>)
3. 了解和体验 SSH 的应用

二、实验内容 (10 分)

| 序 | 内容 | 项目 |
|----|--------|---|
| 1. | SSH 登录 | 帐号新增: user1, user2, user3 |
| 2. | | Linux 下口令登录 & 密钥登录 服务器 IP: 192.168.1.230 客户端 IP: 192.168.0.95 |
| 3. | | 比较分析: 1、SSH 登录的数据包被加密了, 而 telnet 的数据包是明文传输的, 可以直接从数据包中读取到明文信息。 2、Telnet 的端口为 22, SSH 的端口为 23 |
| 4. | SSH 应用 | 端口转发 本地服务器 A IP: 192.168.0.95 远端服务器 B IP: 192.168.1.230 |
| 5. | | 比较分析: 1、设置 Socket 代理后: 端口转发前, 数据传输报文为明文传输; 端口转发后, 数据传输报文是经过加密的, 查看数据包只能看到密文。 2、设置端口映射后: 直接访问时, 数据传输采取明文的形式; 端口映射后, 访问端口有差异, 而且 wireshark 抓取到的 SSH 数据包为密文。 |

1. 服务端实验准备

1.1 环境确认

网络环境与 IP 地址, 客户端 IP 为 192.168.0.95。

```
test@ubuntu1804: ~$ ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.95 netmask 255.255.254.0 broadcast 192.168.1.255
    inet6 fe80::d0a3:2b5f:a4eb:9041 prefixlen 64 scopeid 0x20<link>
    ether fc:b3:c6:2d:12:01 txqueuelen 1000 (Ethernet)
    RX packets 3071 bytes 1055174 (1.0 MB)
    RX errors 298 dropped 0 overruns 0 frame 298
    TX packets 454 bytes 49741 (49.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 237 bytes 21861 (21.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 237 bytes 21861 (21.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

查看 SSH 服务状态: `ps -ef|grep ssh`

```
test@ubuntu1804:~$ ps -ef|grep ssh
root      775      1   0 15:12 ?        00:00:00 /usr/sbin/sshd -D
test      1321  1226   0 15:12 ?        00:00:00 /usr/bin/ssh-agent /usr/bin/ln-
aunch env GNOME_SHELL_SESSION_MODE=ubuntu gnome-session --session=ubuntu
test      1850  1828   0 15:16 pts/0    00:00:00 grep --color=auto ssh
test@ubuntu1804:~$
```

1.2 服务端账号设置

在服务端添加三个新帐号 `user1,user2,user3` 供远程登录。

```
adduser: Only root may add a user or group to the system.
test@ubuntu1804:~$ sudo adduser user1
[sudo] password for test:
Adding user 'user1' ...
Adding new group 'user1' (1002) ...
Adding new user 'user1' (1002) with group 'user1' ...
Creating home directory '/home/user1' ...
Copying files from '/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for user1
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n]
```

2. Linux 客户端登录 SSH 服务器

2.1 口令登录

从 Linux 客户端以口令用户(`user1`)身份登录 SSH 服务器

```
user1@ubuntu1804: ~$
File Edit View Search Terminal Help
user1@192.168.1.230's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

446 packages can be updated.
346 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

user1@ubuntu1804:~$
```

2.2 密钥登录 1 (客户端生成密钥)

在本地客户端生成公私钥对: `ssh-keygen -t rsa`

```
test@ubuntu1804:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/test/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/test/.ssh/id_rsa.
Your public key has been saved in /home/test/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:LQno7P8AVTqsBEPeVky8Tro4ahSuGHUIGD2V2g8lChs test@ubuntu1804
The key's randomart image is:
+--[RSA 2048]-----+
|E=+o ....          |
|+=+o+o            |
|+o+o+o+o          |
|o+o+o+o+o+o       |
|+..+..S..         |
|.B o . . .         |
|O.. . . .          |
|*.. . . .          |
|B o . . .          |
+---[SHA256]-----+
test@ubuntu1804:~$
```

使用 `ssh-copy-id` 命令把本地的 `ssh` 公钥文件安装到远程主机的账户(user2)下：
`ssh-copy-id -i /home/test/.ssh/id_rsa.pub user2@192.168.1.230`

```
test@ubuntu1804:~$ ssh-copy-id -i /home/test/.ssh/id_rsa.pub user2@192.168.1.230
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/test/.ssh/id_rsa.pub"
The authenticity of host '192.168.1.230 (192.168.1.230)' can't be established.
ECDSA key fingerprint is SHA256:dt6pD+S+e0kX1lSzonUetjZ3ISTp4phZZIrcKvKfJw.
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 prompt ed now it is to install the new keys
user2@192.168.1.230's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'user2@192.168.1.230'"
and check to make sure that only the key(s) you wanted were added.

test@ubuntu1804:~$
```

在服务端查看 `user2` 下 `authorized_keys` 文件内容

```
test@ubuntu1804:~$ sudo cat /home/user2/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGDwJF7ehMvYA/aeBIRdWBLt8Z/s2LP4S+MChgxEag
s4L2XoeS3ancZ+IouKXc49T8Jfh2deRLvY1gzhQr8gSyFIAaBCC0CrfXGEPCE7+VH5Sfwk3Q05vu
Z/azYf9Q1Y6t4wXcxsrBocJrvwJnnBwSU01VMD0YwQ51K105XBdJ8qLrY2n5duG7Rok/skZYP55
dkP3kra9Ask+XIj/pnShd76BYld0eP6sVeSpD/cSR4KXt2F4vz2KJRKu4e/R/MDCXrNKQ1R254j9f
xLVehL78Be8x5Y1EWQr6gJpNgA+Ich1Zes/v5LaeQzc0qC3ZYBL0FE9ot1S test@ubuntu1804
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDnW6Hdehuc3XYShjhvkW4MawMZlmbhJRnuC9khB3s+
8upTdk0lkyPg5T/HTnT/gnMuxentBD2EF1yaybwQ4LTSScPACXvXpD7A29LRsqdOk6rr8sUW+dvz1OZ
xT22PHTH3VSM3gEUGqgErQBrGgoJUn+fKtL5p6L9/8Hx2CpK0JlkqcrvmwF19pz3L+zSpKzyGRZ6IP
aGFsqdRquJxZS+d6Q8lCeQ39kwcPh+end9JbbUjvkwkHN07C1tBDKfSKLH4dZxRbNsx2FTAm9qIXhDSK
BB1P9yHmqzeyY7X4Vls16ZnSnHULmttuhVZJ5KT7wdTR4JpVhP6tSwT2nv5/ test@ubuntu1804
test@ubuntu1804:~$
```

以密钥登录用户身份登录 **SSH 服务器**：

```
test@ubuntu1804:~$ ssh user2@192.168.1.230
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

446 packages can be updated.
346 updates are security updates.

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

user2@ubuntu1804:~$
```

2.3 密钥登录 2（服务端生成密钥）

注意先将客户端/home/test/.ssh 中的文件删除。

```
rm: cannot remove 'id_rsa': No such file or directory
test@ubuntu1804:~$ sudo rm /home/test/.ssh/*
test@ubuntu1804:~$ sudo ls /home/test/.ssh
test@ubuntu1804:~$
```

`user3` 登录服务器，在服务器生成公私钥对，并重启 `ssh`：

`ssh-keygen -t rsa`

service sshd restart

```

user3@ubuntu1804:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user3/.ssh/id_rsa):
Created directory '/home/user3/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user3/.ssh/id_rsa.
Your public key has been saved in /home/user3/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:RN0o2XMukuU8tUhdyc6t5N39zty/K20DQ56fn+Enhcs8 user3@ubuntu1804
The key's randomart image is:
+----[RSA 2048]-----+
|
|..+..+
|..+..+
|..+..+
|..+..+
|..+..+
|..+..+
|..+..+
|..+..+
|..+..+
+----[SHA256]-----+

user3@ubuntu1804:~$ service sshd restart
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to restart 'ssh.service'.
Authenticating as: test,,, (test)
Password:
==== AUTHENTICATION COMPLETE ====
user3@ubuntu1804:~$

```

将私钥传递给远端客户端，保存在客户端相应目录下：

```

sftp test@192.168.0.95
cd /home/test/.ssh
put /home/user3/.ssh/id_rsa

```

```

==== AUTHENTICATION COMPLETE ====
user3@ubuntu1804:~$ sftp test@192.168.0.95
The authenticity of host '192.168.0.95 (192.168.0.95)' can't be established.
ECDSA key fingerprint is SHA256:dt6pD+5+e0kX1l5ZonUetjZ3ISTp4phZZIrcKvKfjw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.95' (ECDSA) to the list of known hosts.
test@192.168.0.95's password:
Connected to 192.168.0.95.
sftp> cd /home/test/.ssh
sftp> put /home/user3/.ssh/id_rsa
Uploading /home/user3/.ssh/id_rsa to /home/test/.ssh/id_rsa
/home/user3/.ssh/id_rsa 100% 1679 955.0KB/s 00:00
sftp>

```

将服务器中的公钥 id_rsa.pub 内容追加到 authorized_keys 中：

```
cp ./id_rsa.pub ./authorized_keys
```

```

user3@ubuntu1804:~$ cd /home/user3/.ssh
user3@ubuntu1804:~/.ssh$ cp ./id_rsa.pub ./authorized_keys
user3@ubuntu1804:~/.ssh$

```

从 linux 客户端以密钥登录用户身份登录 SSH 服务器：

```

test@ubuntu1804:~$ ssh user3@192.168.0.129
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

446 packages can be updated.
346 updates are security updates.

New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri May 21 15:45:43 2021 from 192.168.0.95
user3@ubuntu1804:~$

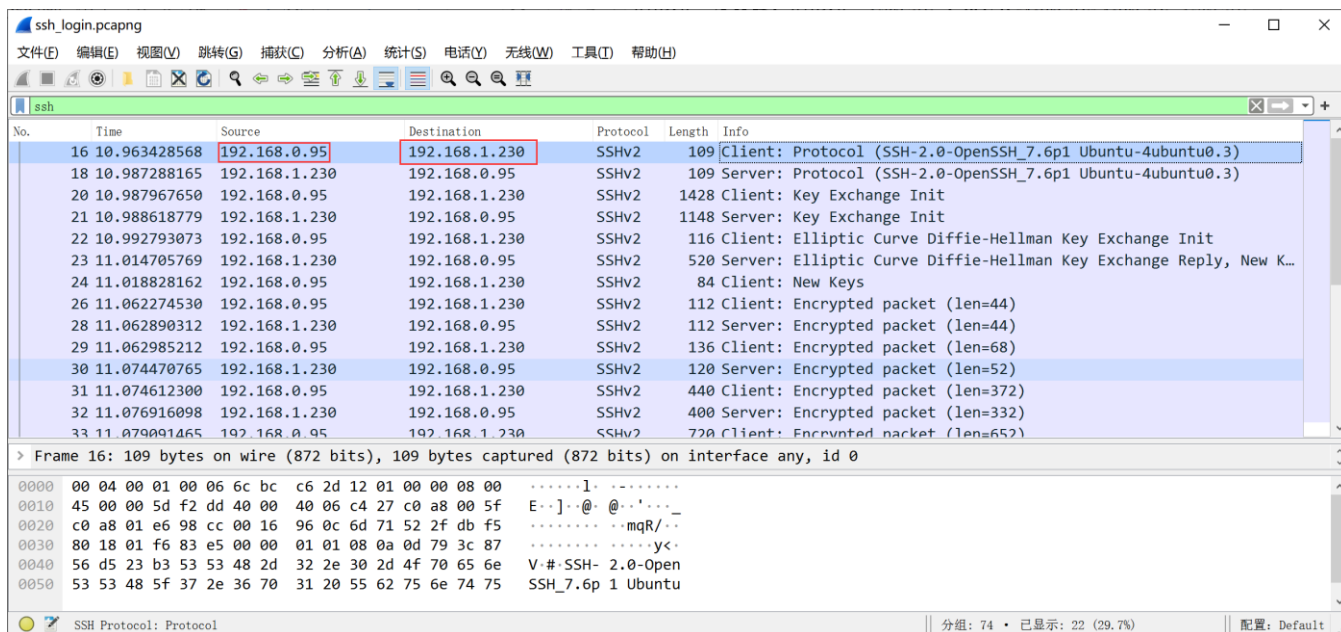
```

2.4 比较分析

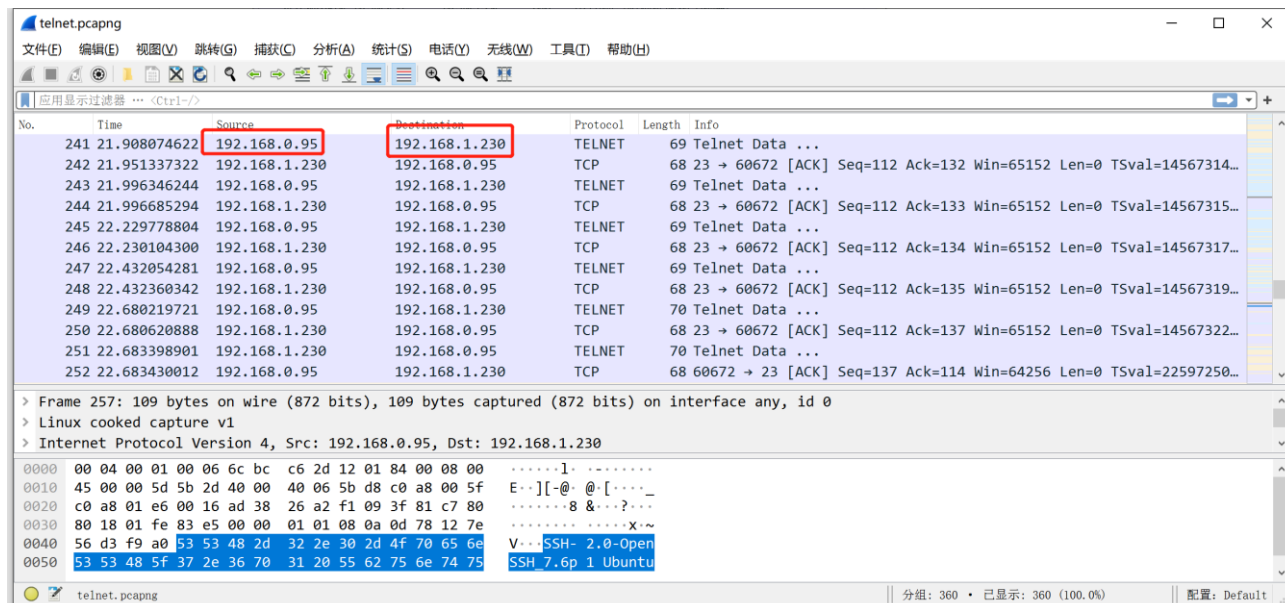
关于/etc/ssh/sshd_config 文件内容等分析见思考题 2。

用 sudo wireshark 指令运行 wireshark 观察 telnet 和 SSH 登录的区别。

首先通过 SSH 登录远端服务器：



再通过 telnet 登录远端服务器：



对比 wireshark 抓包信息，可以发现：

- 1、SSH 登录的数据包被加密了，而 telnet 的数据包是明文传输的，可以直接从数据包中读取到明文信息。
- 2、Telnet 的端口为 22，SSH 的端口为 23

三、分析和思考（80 分）

1. 查看服务端的公钥证书文件、authorized_keys 文件等的内容和权限，试图修改权限后再次进行实验，试分析权限设置的原因。（20 分）

User3 的公钥证书文件内容：

```
test@ubuntu1804:~$ sudo cat /home/user3/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCTg3UGqDlkm/vOF5VDr7+/r/RhLLQkIXb/mMULv9iQXAbmfiLvbAGkrI3r
q2MdC0leq7Qhqw3H4Pb223ITpHEWYwZyUudWnsg2TFR41n/NvtU3JzhpeoCVtJUjkh4EvCn46A+ovJlGQ7m9TZpUHyO8LTey
MvQBywxHrZiVaFuGoPAWNR/MtUYg60HQdLGe3eddkqvzPkHICdF5k/t03MIZ3BAjIeRx0oESnw+rIPfp+KNehV5d6LQn08C
/dJ0mc2X5NMVTtyQsvG1tLToGsDLNxAzo8ZUAZYxP39PELGR/BDH9+IKvBfQFUpAA13BvRw1fKpBtXoVewv5YyMnUX7 use
r3@ubuntu1804
test@ubuntu1804:~$
```

User3 的私钥证书文件内容：

```
test@ubuntu1804:~$ sudo cat /home/user3/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAu4N1Bgg5ZJv7zheVQ6+/v6/0YzS0JCF2/5jFC7/YkFwG5n4i
72WBpKyN66tjHXDonqu0IasNx+D29ttyE6RxFmFs2FLNvp7Ink30eNZ/zb7VNYc4
aXqAlbSVI5B+BLwp+OgPqLyZRk05VU2aVB8qPJU3sJl0AcsMR62SL2qn7hqDWMdu
fzLVGI0jH0ZRNt3nXapL8z5ByAnReZP7TtzImdwQIyHkcDK8EPq4j36fijXoV
eXeI0JzAv3SdJnNL+TTFU7ckLLxtbZU76BrAyzcQGaPGVAGWMTyFTxCxkfwQx/f
iCwR0BVQAndwVa1tXyqQbV6FXsL+WmjJlF+wIDAQABoIABcUDNku5zFsyUCk
PALCpLTW8QD5jlpMdbtVMBtMF1ldGjRQHfidsG4WfRmEtIQxG1A8DcV/CgkptPuY
8qdc09bnUsALI/ftPwz/9zDQJZarAA7zssu4mg751PA1nmzyw6+w0AySecLEHwQf
gvkLH0VD8EDSah+3/+Dqle1wE08K2fyYKwngzepe0+GdL97xC08N/yKfaW4PIiB
q0oqK0NBvVrYeWrbUNxJEAm3njPU71szNvwUtTGBvSLDrbpXW3thjyF4L82VFXf
gvsS5b83JhuerzVXURegZbZHT7dieqecyReB0930GanWfy+TeJ38rhnm5h6ssq
FzGfRhEcgyEA+ZM5SAX1BEpkKVAcjogRL03gW0ML0tcfQ8RhhPL1gq1Jc5yFLpS
wn2G4hje5K+cfcIN4wAFWbNkbvhJiLYIMu6k8CepD2M+gaZmyJ07UxL8MWZlub3i
U02YFW+Pdr98JW6PK5Ms76h61CApmprSKuCng90/OjuHXb7feub2y0cCgyEAWFc6
gAFjU8Cuor7AGh90Qp0k0Uv0LZuJQa1sQkPdXXWmZ7ryQA0iCyA0JXrbs/3/uks
x0kGFN3kfKZKRKES1M0wXhv++8xNQIXq7MH9gfgL/cYqHPEWkYry1bXtnx+yL75G
A4Vocq3SwdXbv6YK6AbXJRpsn+2X3bBokqpUYa0CgyEALRECApxNlYbVHRFcgJz
R3Su+62IGL6zQ0wnI1YfobCzkelE6mx/kfs33UPNfnmFpcZHEQ27eTQyIC0JJvV
OdagXXDDWgWFCBtdUPnEIVZPF5IRoGdyDx808X7YyLQNIq4GhZhadqrC8oBjWryv
+xD6bWUhlj+fJPuSzoX3ECgyEAnbfpdIsDUe4c88b055zbtLRN0G179z0Qq+nQ
mzb+w3vYhB/44H13bRpxFpmY00kgRpsV60RsZDf/4GbLiQhbVLSrYPP9th0lvA5
T6NTXkBuGg6qY4M8YuqzKu+CjmHd7bi2ojsKf13T73aUDDryI+zY+1jHVFksrKX
TXL2dpECgyBbk66o/Fe6SLzkbZe1HG6Jry7+EZ01JJp4FdKUB/NIEKLS+33C2zm
/F/41Z5Pfc6ZfyjokQcIFLPhd40PRNEjuPA/eLKxUZAoe+oALSLsZjq1vM+VGHC
8wLEe80PYliUQ3G/plEPnyNf6z90NzwIUEGAPzvs3083vseH9/Lew==
-----END RSA PRIVATE KEY-----
test@ubuntu1804:~$
```

User2 的 authorized_keys 文件内容：

```
test@ubuntu1804:~$ sudo cat /home/user2/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDdUwJF7ehMVyA/aeb1Rdw8Lt8Z/s2LP4S+MGhgxEag
s4L2X6eS3eNcZ+lo+LKGX49T8Jfh2deRLcyIlgzHQrBgZ5yFIAa8CoCHRGfxGEPCE7+vHSSfwk30Q5vu
Z/azYf69Q1Y6t4wXGxsRbocJrvwjnmBw5U01VMdDyWQS13Kl0Sxx8dJ68qLrY2m5duOG7R0k/skZYP55
dkP3kRa9Ask+XIj/pm5hdT6BYyIdD0eP6sVe5p0/cSR4KXt2F4vz2KJRKul4e/R/MDCXrNKQirZ54j9f
xlVehLg7Be8x5Y1EWQR6LgJpNqA+Ich1Zes/v5laeQzc0qC3zYBL0fE9oti5 test@ubuntu1804
```

User3 的公钥文件及 authorized_keys 文件权限如下：

```
test@ubuntu1804:/home/user3$ sudo ls -la .ssh
total 20
drwx----- 2 user3 user3 4096 5月 27 11:36 .
drwxr-xr-x 5 user3 user3 4096 5月 27 11:20 ..
-rw----- 1 root root 398 5月 27 11:36 authorized_keys
-rw----- 1 user3 user3 1679 5月 27 11:20 id_rsa
-rw-r--r-- 1 user3 user3 398 5月 27 11:20 id_rsa.pub
test@ubuntu1804:/home/user3$
```

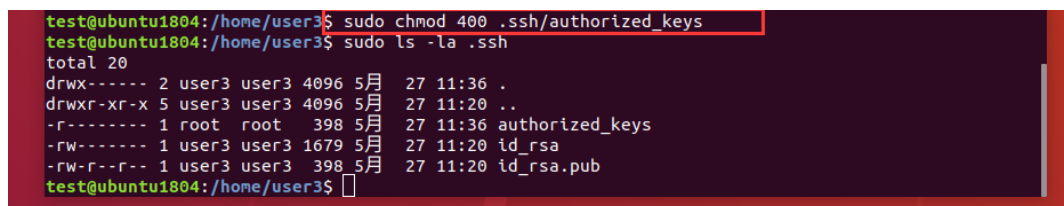
由图中可见：

公钥文件具有 644 权限

私钥文件具有 600 权限

authorized_keys 文件具有 600 权限

修改 authorized_keys 权限为 400：



```
test@ubuntu1804:/home/user3$ sudo chmod 400 .ssh/authorized_keys
test@ubuntu1804:/home/user3$ sudo ls -la .ssh
total 20
drwx----- 2 user3 user3 4096 5月 27 11:36 .
drwxr-xr-x 5 user3 user3 4096 5月 27 11:20 ..
-r----- 1 root root 398 5月 27 11:36 authorized_keys
-rw----- 1 user3 user3 1679 5月 27 11:20 id_rsa
-rw-r--r-- 1 user3 user3 398 5月 27 11:20 id_rsa.pub
test@ubuntu1804:/home/user3$
```

在客户端生成密钥对时，客户端需要读写 authorized_keys 文件的权限，则此时不能再使用 ssh-copy-id 将公钥保存到服务器，不能通过密钥登录。

在服务器生成密钥对时，服务器不需要写文件权限，所以可以将私钥 id_rsa 传给客户端，仍能实现密钥登录。

2. /etc/ssh/目录下的 sshd_config 和 ssh_config 配置文件两者有什么区别？如要禁止某用户的远程 SSH 登录，或禁止密码认证方式的 SSH 登录应做怎样的设置？（给出相应的实验截图至少 2 个）（20 分）

一、区别：

ssh_config 和 sshd_config 都是 ssh 服务器的配置文件，都允许通过设置不同的选项来改变客户端程序的运行方式。

ssh_config 是针对客户端的配置文件

sshd_config 是针对服务端的配置文件。

二、实验配置

禁止用户的远程 SSH 登录：

DenyUsers [用户名]

禁止密码认证方式：

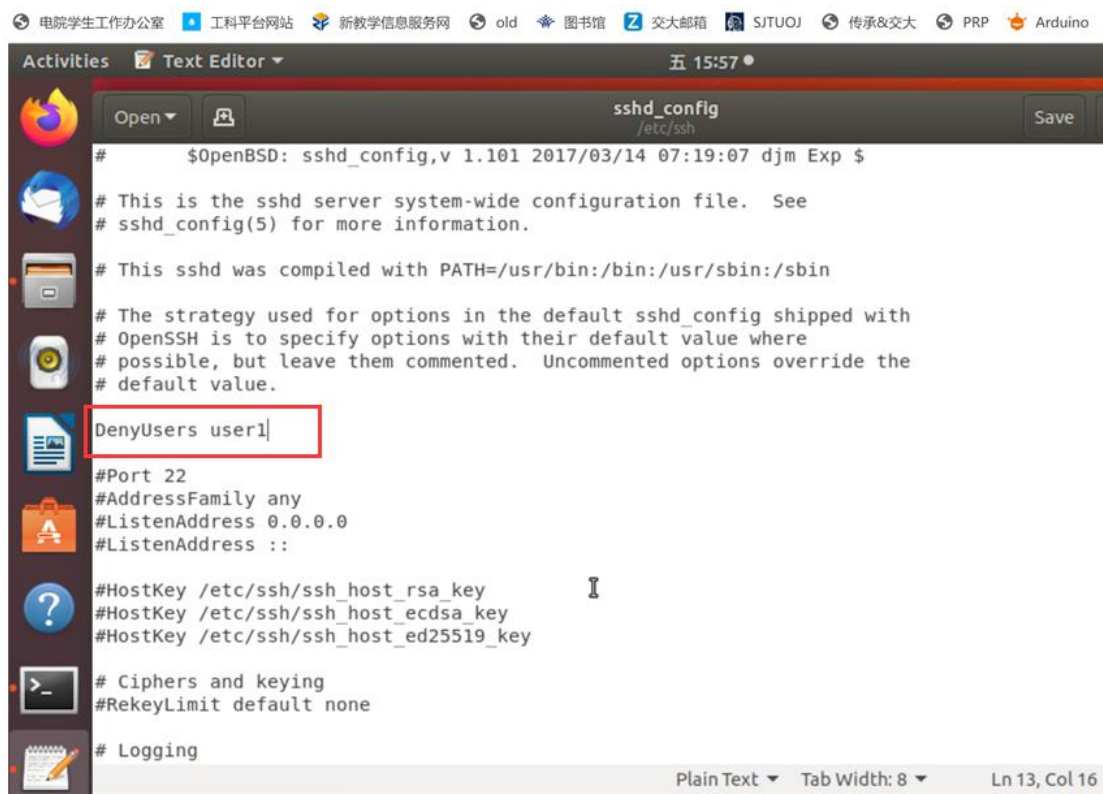
PasswordAuthentication no

下面具体介绍设置方式：

修改 sshd_config 配置文件，为禁止用户 User1 的远程 SSH 登录，添加如下配置：

DenyUsers user1

具体配置如下图所示：



修改配置后重启 ssh 服务，输入 **ssh user1@192.168.1.230** 后，登录连接被阻止了。

测试效果如下图所示：



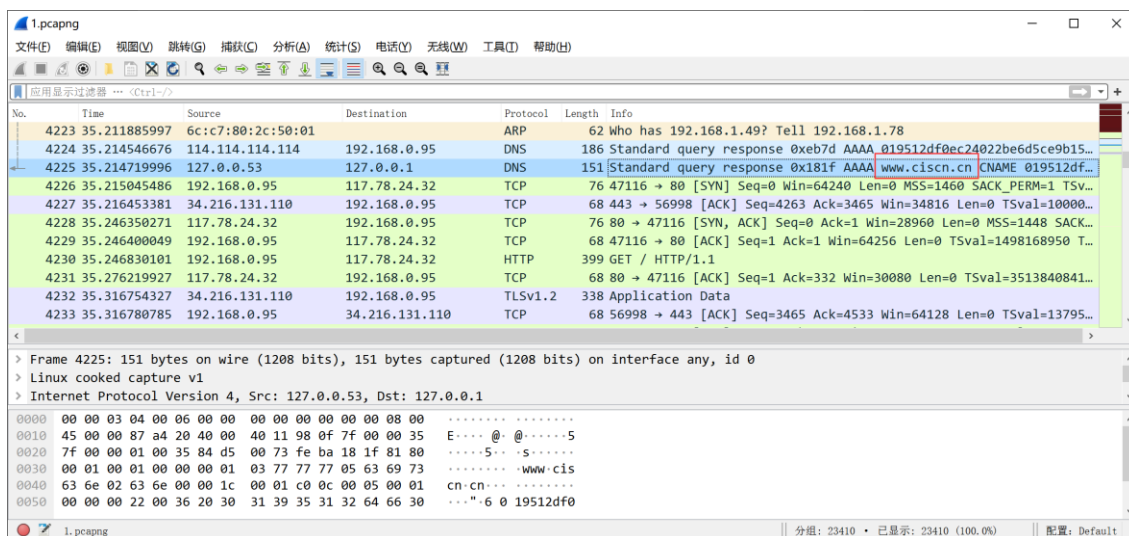
3. 给出三个 SSH 端口转发实验的相应设置命令、实验截图并简单说明(每个至少 2 个截图)。并画出端口转发实验中任一功能实现的网络拓扑及数据传输示意图,并指出该功能的可能(具体)应用场景。(20 分)

SSH 应用共有三个端口转发实验:

实验一:设置服务器 A 的浏览器 socks 代理[IP 设置为 127.0.0.1,端口设置为 1080]

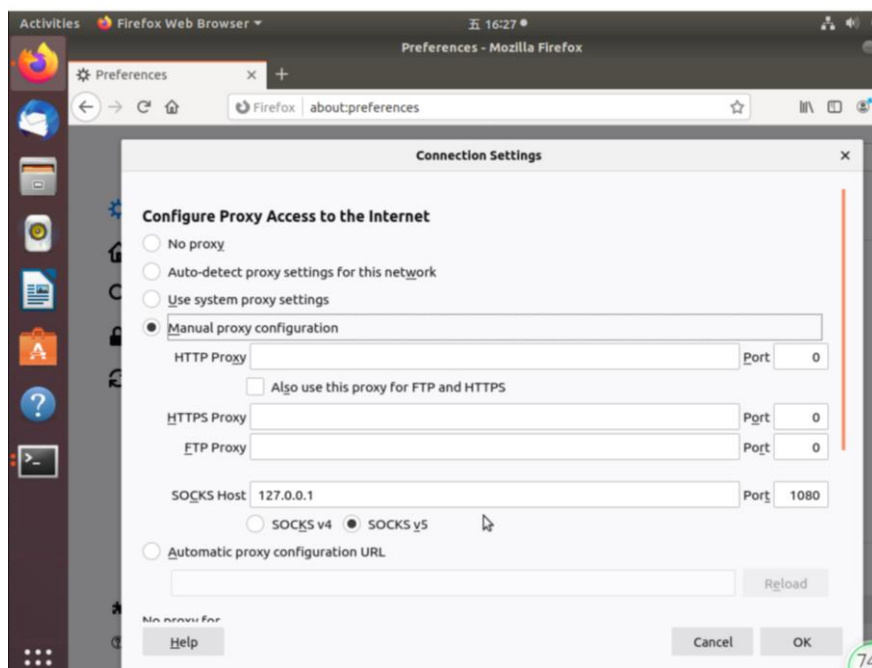
1、端口转发前:

直接访问网页 www.ciscn.cn, 用 wireshark 抓包如下:



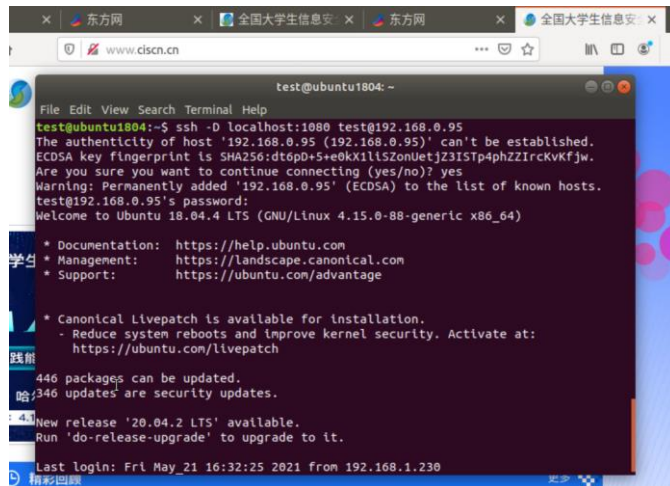
2、端口转发后:

开启端口转发, 开启代理, 配置方式如下图所示:

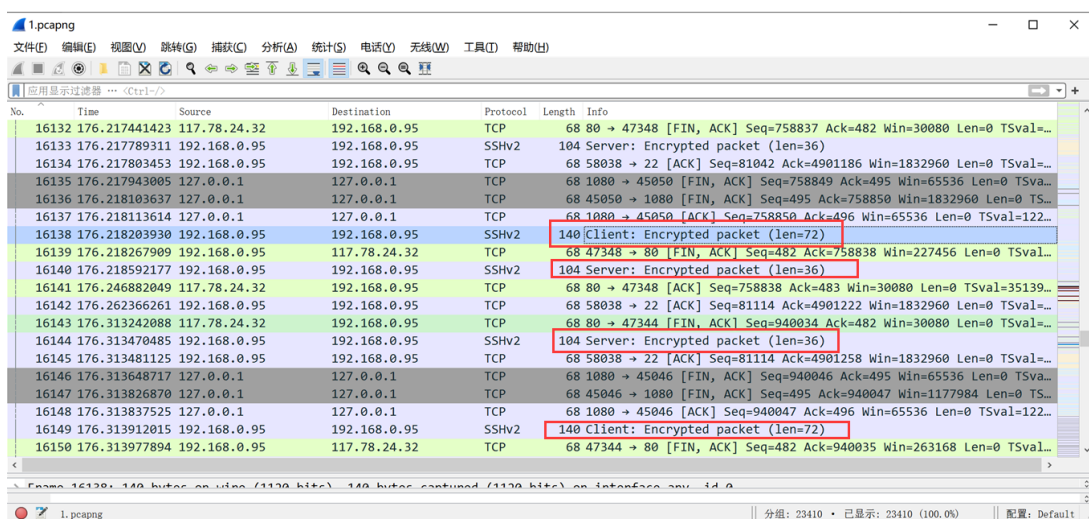


输入端口转发指令：

```
ssh -D localhost:1080 test@192.168.0.95
```



用 Wireshark 抓包查看 SSH 加密报文如下：



对比上述转发前后数据包信息，可见端口转发后发生了改变，SSH 将数据报文信息映射到 Sockets 指定端口，浏览器代理通过这个端口获取数据。而传输过程中 SSH 充当映射的作用。

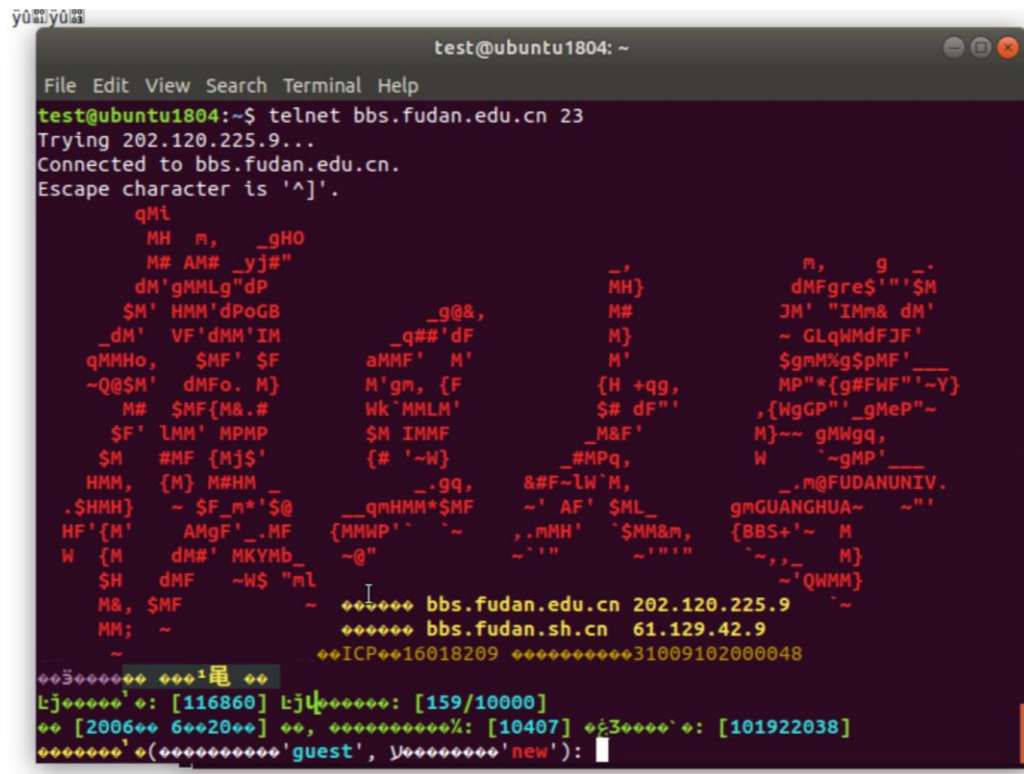
端口转发前，数据传输报文为明文传输；

端口转发后，数据传输报文是经过加密的，查看数据包只能看到密文。

实验二: 服务器通过 telnet 访问 A 的 3456 端口来访问 bbs.fudan.edu.cn 的 bbs 服务;

1、端口转发前:

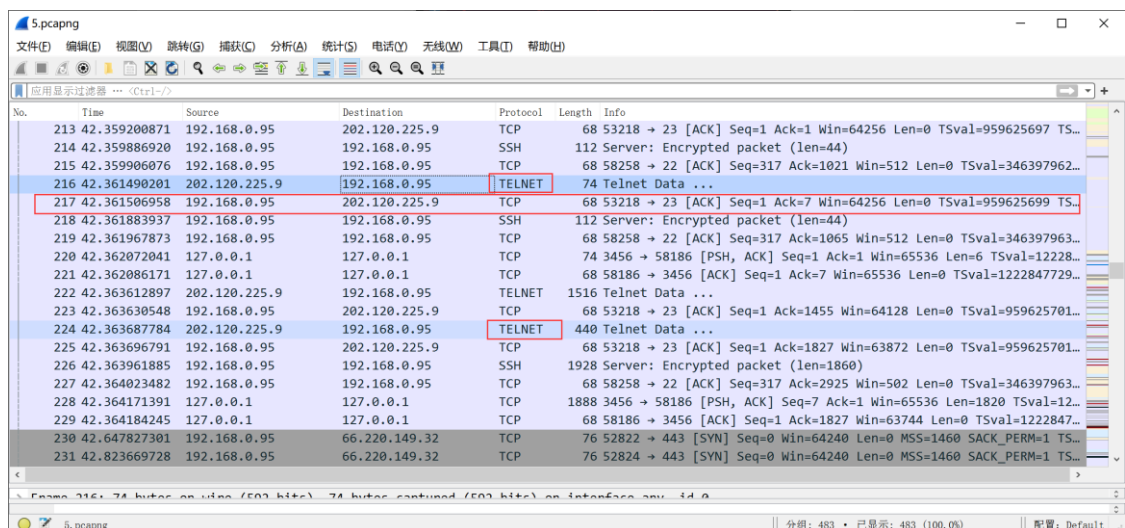
直接访问 bbs.fudan.edu.cn, 测试如下:



```

test@ubuntu1804: ~
File Edit View Search Terminal Help
test@ubuntu1804:~$ telnet bbs.fudan.edu.cn 23
Trying 202.120.225.9...
Connected to bbs.fudan.edu.cn.
Escape character is '^]'.
qM!
MH m, _gho
M# AM# _y#
dM'gMMLg"dP
$M' HMM'dPoGB
_dM' VF'dMM'IM
qMHo, $MF' $F
-Q@SM' dMfo. M)
M# $MF{M&.#
$F' LMM' MPMP
$M #MF {Mj$'
HMM, {M} M#HM
.SHHH) ~ $F_m*$@
HF'[M' AMgF'_.MF
W {M dM# MKYmb
$H dMF ~W$ "mL
M&, $MF
MM; ~
bbs.fudan.edu.cn 202.120.225.9
bbs.fudan.sh.cn 61.129.42.9
ICP16018209 31009102000048
[116860] [159/10000]
[2006] [6020] [10407] [101922038]
(*****'guest', *****'new'):
  
```

使用 wireshark 抓包分析如下:

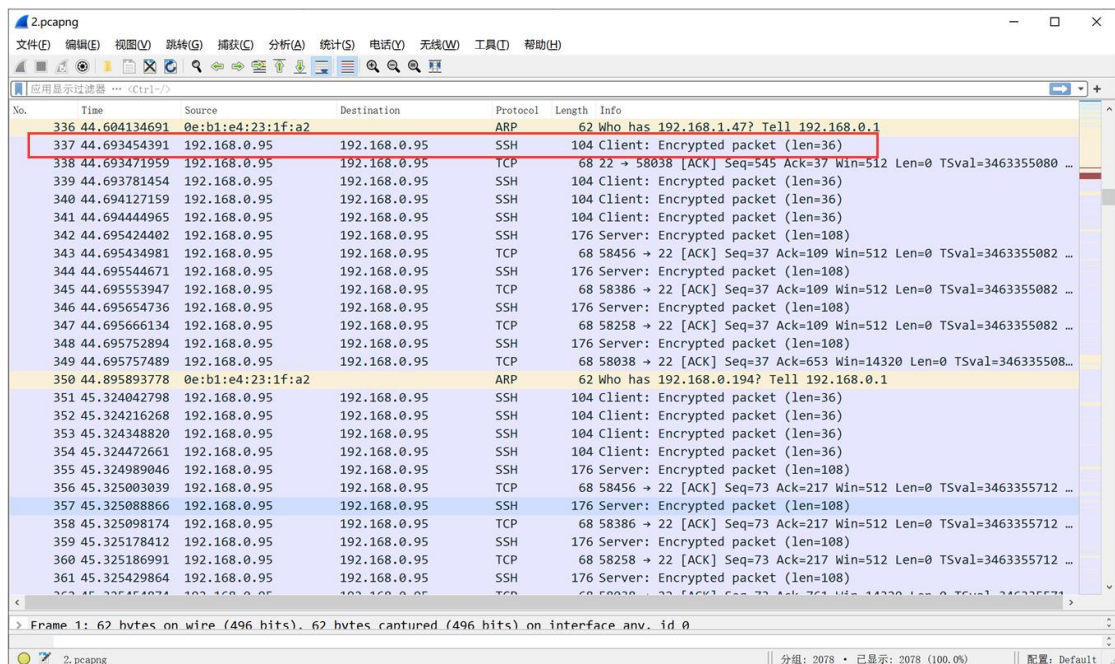


| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|--------------|---------------|---------------|----------|--------|--|
| 213 | 42.359200871 | 192.168.0.95 | 202.120.225.9 | TCP | 68 | 53218 → 23 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=959625697 TS... |
| 214 | 42.359886920 | 192.168.0.95 | 192.168.0.95 | SSH | 112 | Server: Encrypted packet (len=44) |
| 215 | 42.359906076 | 192.168.0.95 | 192.168.0.95 | TCP | 68 | 58258 → 22 [ACK] Seq=317 Ack=1021 Win=512 Len=0 TSval=346397962... |
| 216 | 42.361490201 | 202.120.225.9 | 192.168.0.95 | TELNET | 74 | Telnet Data ... |
| 217 | 42.361506958 | 192.168.0.95 | 202.120.225.9 | TCP | 68 | 53218 → 23 [ACK] Seq=1 Ack=7 Win=64256 Len=0 TSval=959625699 TS... |
| 218 | 42.361883937 | 192.168.0.95 | 192.168.0.95 | SSH | 112 | Server: Encrypted packet (len=44) |
| 219 | 42.361967873 | 192.168.0.95 | 192.168.0.95 | TCP | 68 | 58258 → 22 [ACK] Seq=317 Ack=1065 Win=512 Len=0 TSval=346397963... |
| 220 | 42.362072041 | 127.0.0.1 | 127.0.0.1 | TCP | 74 | 3456 → 58186 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=6 TSval=12228... |
| 221 | 42.362086171 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 58186 → 3456 [ACK] Seq=1 Ack=7 Win=65536 Len=0 TSval=1222847729... |
| 222 | 42.363612897 | 202.120.225.9 | 192.168.0.95 | TELNET | 1516 | Telnet Data ... |
| 223 | 42.363630548 | 192.168.0.95 | 202.120.225.9 | TCP | 68 | 53218 → 23 [ACK] Seq=1 Ack=1455 Win=64128 Len=0 TSval=959625701... |
| 224 | 42.363687784 | 202.120.225.9 | 192.168.0.95 | TELNET | 440 | Telnet Data ... |
| 225 | 42.363696791 | 192.168.0.95 | 202.120.225.9 | TCP | 68 | 53218 → 23 [ACK] Seq=1 Ack=1827 Win=63872 Len=0 TSval=959625701... |
| 226 | 42.363961885 | 192.168.0.95 | 192.168.0.95 | SSH | 1928 | Server: Encrypted packet (len=1860) |
| 227 | 42.364023482 | 192.168.0.95 | 192.168.0.95 | TCP | 68 | 58258 → 22 [ACK] Seq=317 Ack=2925 Win=502 Len=0 TSval=346397963... |
| 228 | 42.364171391 | 127.0.0.1 | 127.0.0.1 | TCP | 1888 | 3456 → 58186 [PSH, ACK] Seq=7 Ack=1 Win=65536 Len=1820 TSval=12... |
| 229 | 42.364184245 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 58186 → 3456 [ACK] Seq=1 Ack=1827 Win=63744 Len=0 TSval=1222847... |
| 230 | 42.647827301 | 192.168.0.95 | 66.220.149.32 | TCP | 76 | 52822 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TS... |
| 231 | 42.823669728 | 192.168.0.95 | 66.220.149.32 | TCP | 76 | 52824 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TS... |

2、端口转发后:

在服务器 A 通过 telnet 命令访问 A 的 3456 端口, 将会访问 bbs.fudan.edu.cn

使用 wireshark 抓包分析如下：



端口转发前后差别：

端口转发前：直接访问 bbs 时，通过 wireshark 抓取到的数据报文是 telnet 的报文。

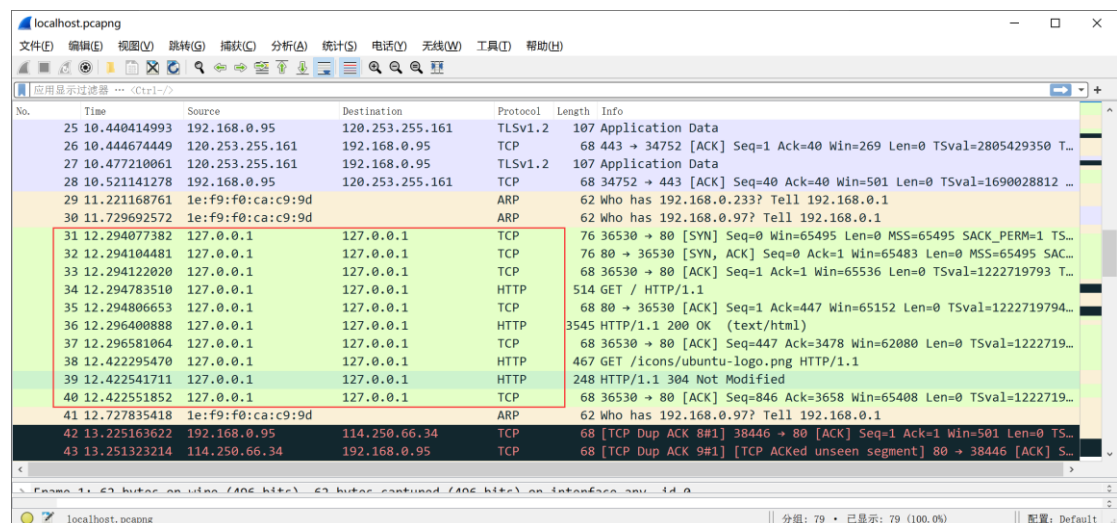
端口转发后：端口映射后，wireshark 抓取到的是通过 SSH 传递的加密报文。

实验三：服务器 B 通过浏览器访问 B 的 3456 端口将会访问服务器 A 的 web 服务

1、端口转发前：

首先直接访问 localhost，可以正常访问 web 服务。

使用 wireshark 抓取数据包分析如下：



2、端口转发后：

在服务器 B 通过浏览器访问 B 的 3456 端口，将会访问服务器 A 的 web 服务。

```
ssh -R 3456:localhost:80 test@192.168.0.95
```

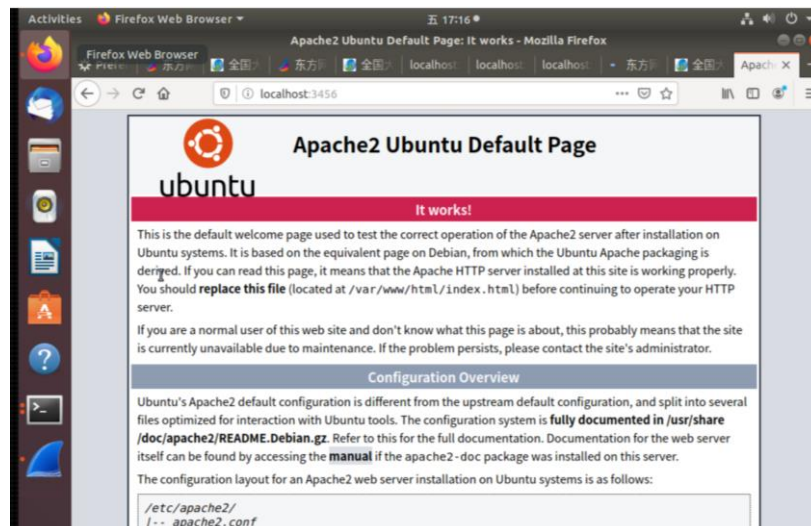
配置方式如下图所示：

```
test@ubuntu1804:~$ ssh -R 3456:localhost:80 test@192.168.0.95
test@192.168.0.95's password:
Warning: remote port forwarding failed for listen port 3456
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
446 packages can be updated.
346 updates are security updates.
New release '20.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
Last login: Fri May 21 16:43:36 2021 from 192.168.0.95
test@ubuntu1804:~$
```

在浏览器中直接访问 localhost:3456 可以访问 web 服务，如下图所示：



使用 wireshark 抓取数据包分析如下：

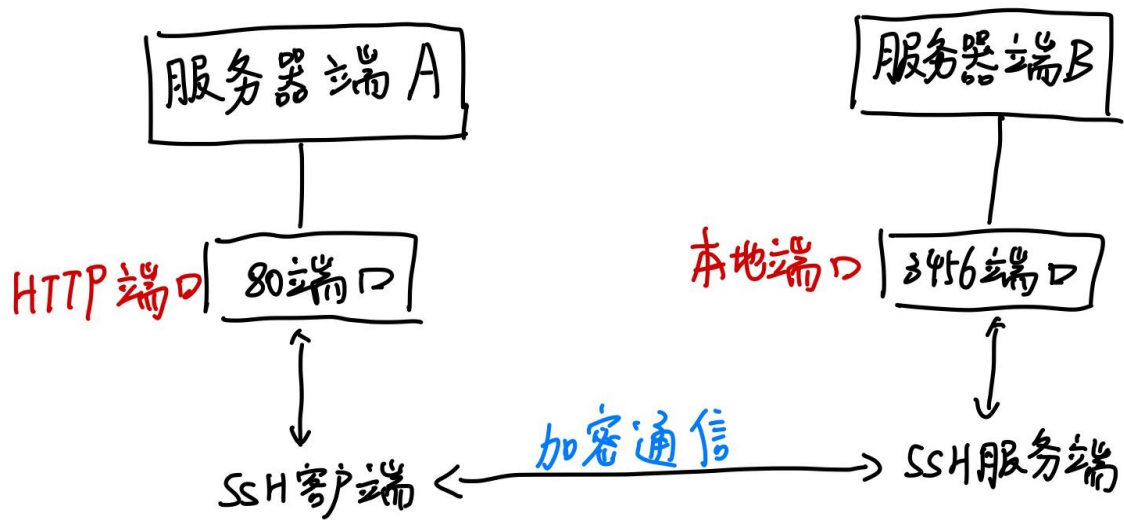
| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|--------------|--------------|--------------|----------|--------|--|
| 126 | 17.516268252 | 127.0.0.1 | 127.0.0.1 | HTTP | 401 | GET / HTTP/1.1 |
| 127 | 17.516296701 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 3456 → 58226 [ACK] Seq=1 Ack=334 Win=65152 Len=0 TSval=12229455... |
| 128 | 17.516576149 | 192.168.0.95 | 192.168.0.95 | SSHV2 | 440 | Server: Encrypted packet (len=372) |
| 129 | 17.516898602 | 127.0.0.1 | 127.0.0.1 | HTTP | 401 | GET / HTTP/1.1 |
| 130 | 17.516956845 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 80 → 36600 [ACK] Seq=1 Ack=334 Win=65152 Len=0 TSval=1222945600... |
| 131 | 17.519016494 | 127.0.0.1 | 127.0.0.1 | HTTP | 3545 | HTTP/1.1 200 OK (text/html) |
| 132 | 17.519032216 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 36600 → 80 [ACK] Seq=334 Ack=3478 Win=63360 Len=0 TSval=1222945... |
| 133 | 17.519136559 | 192.168.0.95 | 192.168.0.95 | SSHV2 | 3584 | Client: Encrypted packet (len=3516) |
| 134 | 17.519152987 | 192.168.0.95 | 192.168.0.95 | TCP | 68 | 22 → 58910 [ACK] Seq=3682 Ack=7514 Win=63360 Len=0 TSval=346407... |
| 135 | 17.519259784 | 127.0.0.1 | 127.0.0.1 | HTTP | 3545 | HTTP/1.1 200 OK (text/html) |
| 136 | 17.519539445 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 58226 → 3456 [ACK] Seq=334 Ack=3478 Win=62080 Len=0 TSval=12229... |
| 137 | 17.721116654 | 127.0.0.1 | 127.0.0.1 | HTTP | 365 | GET /icons/ubuntu-logo.png HTTP/1.1 |
| 138 | 17.721206298 | 192.168.0.95 | 192.168.0.95 | SSHV2 | 400 | Server: Encrypted packet (len=332) |
| 139 | 17.721301303 | 127.0.0.1 | 127.0.0.1 | HTTP | 365 | GET /icons/ubuntu-logo.png HTTP/1.1 |
| 140 | 17.721614520 | 127.0.0.1 | 127.0.0.1 | HTTP | 3691 | HTTP/1.1 200 OK (PNG) |
| 141 | 17.721625993 | 127.0.0.1 | 127.0.0.1 | TCP | 68 | 36600 → 80 [ACK] Seq=631 Ack=3478 Win=63360 Len=0 TSval=1222945... |
| 142 | 17.721797873 | 192.168.0.95 | 192.168.0.95 | SSHV2 | 3728 | Client: Encrypted packet (len=3660) |
| 143 | 17.721805932 | 192.168.0.95 | 192.168.0.95 | TCP | 68 | 22 → 58910 [ACK] Seq=4014 Ack=11174 Win=63360 Len=0 TSval=34640... |
| 144 | 17.721905432 | 127.0.0.1 | 127.0.0.1 | HTTP | 3691 | HTTP/1.1 200 OK (PNG) |

端口转发前后差别:

直接正常访问时, 数据传输采取明文的形式;

端口映射后, 访问端口有差异, 而且 SSH 数据包为密文, 经过了加密。

网络拓扑及数据传输示意图见下:



应用场景:

1、许多大型互联网企业都有居家办公模式, 而企业往往部署了防火墙拒绝外部流量。当员工在家需要访问公司服务器时, 可以在公司提前设置端口转发, 将服务器上开放的端口转发到本地端口, 以实现内网穿透。

```
ssh -L 3456:网站 A:80 [UserName]@[企业服务器 IP]
```

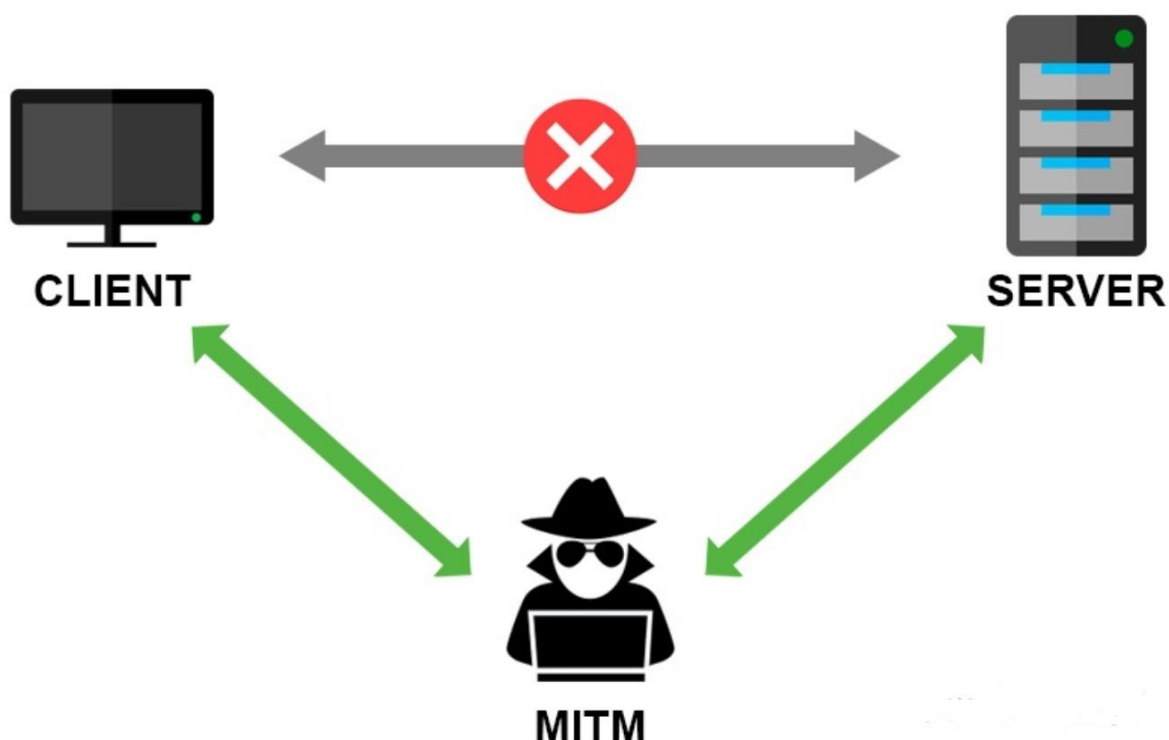
2、对端监听: 公网环境与内网交互, A 传文件给 C 机器上, B 在过程中监听在 B 机 (192.168.1.2) 上: `ssh -R 2345:192.168.1.3:22 10.12.12.14`
其中, A 机 (10.12.12.14), C 机 (192.168.1.3)

此时启动了 2345 监听端口, 而监听端口是在 A 机上

4. 中间人攻击的产生原因是什么？试以 SSH 应用（登录或端口转发）实验内容之一的过程及 Wireshark 所抓取内容进行分析，说明 SSH 为何可以抵抗中间人攻击。（20 分）

一、中间人攻击产生原因：

信息在网络上传输时会被中间节点保存并转发，攻击者可以在传输过程中进行窃听，还能以中间人的伪造身份对窃听的数据进行篡改（例如：替换密钥）。当通信双方的身份认证机制不完善时，攻击者容易伪装成合法通信方与双方进行信息交换。

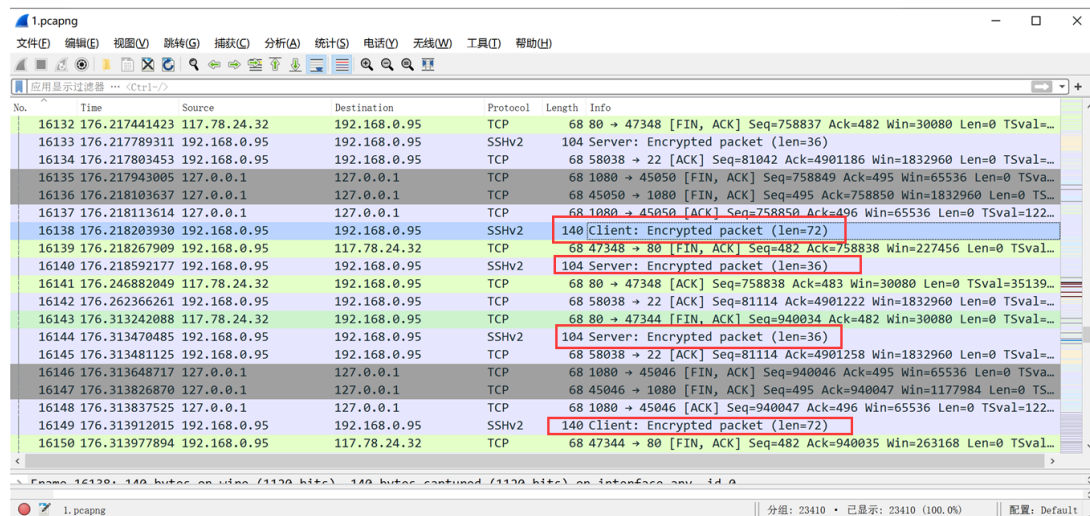


二、SSH 为何可以抵抗中间人攻击：

对于 SSH 的密钥登陆中，双方需要先进行加密机制的协商，协商结束时服务器保存客户端公钥。所有用户都可以连接服务器请求认证，但只有**拥有私钥的客户端**才能通过认证并连接服务器。此时，没有客户端私钥的中间人无法伪装成已认证用户与服务器通信。后续通信的密钥由客户端和服务**器共同生成**，所以中间人只能中转客户端加密后的数据。

由此可见，知道 SSH 客户端公钥的中间人可以伪装成**目标服务器**与客户端通信，但不能与真实服务器通信（不具有客户端私钥）。

三、wireshark 抓包分析：端口转发



由上图可见，SSH 在端口转发时对数据包进行了加密，抓取的数据报文只能查到报文长度，而看不到明文信息。

四、实验总结（收获和心得）（5 分）

本次实验主要涉及 SSH，在实验过程中我加深了对密码算法的使用与理解，尤其是公钥算法的应用。实验前在官网学习了 OpenSSH 的相关命令及应用，在实验中加深了对 SSH 的理解。

在修改 SSH 服务器配置文件的过程中，我熟悉了登录权限管理的内容，大概了解了访问控制的原理。在口令登录和密钥登录实验时，其中用到了公私钥生成、保存和分发，我对 SSH 中公钥密码的应用更加熟悉。传输公钥、私钥时为了服务器和客户端之间的安全传输，采用了 scp 和 ssh-copy-id 命令，用以实现密文传输。实验三涉及端口转发和网络服务，让我学到了远程命令执行的相关功能。我深入理解端口转发的几类模型和命令后，对于 HTTP、Socks、Telnet 等各类网络服务更加熟悉。

整个实验涵盖了新建用户、密钥配置、用户登录到管理的各项功能，我对于 SSH 协议和服务器的的工作原理有了更深的认识，收获颇丰。

五、尚存问题或疑问、建议 (5 分)

问题:

实验三中对比分析了 Sockets 代理前后的结果,那么对于 Sockets 的具体原理和实现方式是什么样的呢?

建议:

实验指导书或许可以更细致一些,比如这次的“比较分析”,在实验过程中大家可能在配置好相关命令之后,才意识到需要在配置前截图、抓包并对比分析。如果指示可以再细致一点,同学们或许能更好完成。