OpenSSH

By
Subrahmanya Bhat, Dept M.C.A
Srinivas University, Mangaluru.

SSH

- SSH (Secure Shell) is a protocol which facilitates secure communications between two systems using a client-server architecture
- allows users to log in to server host systems remotely.

SSH..

- Unlike other remote communication protocols, such as **FTP** or **Telnet**, **SSH** encrypts the login session, rendering the connection difficult for intruders to collect unencrypted passwords.

- The **ssh** program is designed to replace older, less secure terminal applications **telnet** or **rsh**.

- A related program called **scp** replaces older program **rcp** designed to copy files between hosts

- Since these older applications do not encrypt passwords transmitted between the client and the server, avoid them whenever possible.

- Using secure methods to log in to remote systems decreases the risks for both the client system and the remote host

- Red Hat Enterprise Linux includes the general OpenSSH package, ***openssh***, as well as the OpenSSH server, ***openssh-server***, and client, ***openssh-clients***, packages

- The OpenSSH packages require the OpenSSL package **openssl-libs**, which installs several important cryptographic libraries

Why Use SSH?

- Potential intruders have a variety of tools to disrupt, intercept, and re-route network traffic

Threats

- **Interception** of communication between two systems

- **Impersonation** of a particular host

Interception

- The attacker can be somewhere on the network between the communicating parties, copying any information passed between them.
- He may intercept and keep the information, or alter the information and send it on to the intended recipient.

- This attack is usually performed using a **packet sniffer**, a network utility that captures each packet flowing through the network, and analyzes its content.

Impersonation

- Attacker's system is configured to pose as the intended recipient of a transmission.

- This attack can be performed using a technique known as DNS poisoning, and IP spoofing

DNS poisoning

- In this, the intruder uses a cracked DNS server to point client systems to a maliciously duplicated host.

IP spoofing

- In this case, the intruder sends falsified network packets that appear to be from a trusted hos

- If SSH is used for remote shell login and file copying, these security threats can be greatly diminished.
- This is because the SSH client and server use digital signatures to verify their identity.
- Additionally, all communication between the client and server systems is encrypted.

Main Features of SSH

- No one can pose as the intended server
- No one can capture the authentication information
- No one can intercept the communication

Main Features of SSH..

- It provides secure means to use graphical applications over a network
- It can be used to create a secure channel
- It supports the Kerberos authentication

version

- Two varieties of SSH currently exist: version 1, and newer version 2

- The following series of events help protect the integrity of SSH communication between two hosts

Step 1

- A cryptographic handshake is made so that the client can verify that it is communicating with the correct server

Step 2

- The transport layer of the connection between the client and remote host is encrypted using a symmetric cipher

Step 3

- The client authenticates itself to the server

Step 4

- The client interacts with the remote host over the encrypted connection

Configuration Files in SSH

- There are two different sets of configuration files: those for client programs (that is, **ssh**, **scp**, and **sftp**), and those for the server (the **sshd** daemon).

- System-wide SSH configuration information is stored in the **/etc/ssh/**

Starting an OpenSSH Server

- In order to run an **OpenSSH** server, you must have the **openssh-server** package installed

systemctl start sshd.service

- To stop the running **sshd** daemon in the current session

```
systemctl  stop  sshd.service
```

- If you want the daemon to start automatically at the boot time, type as root

  systemctl enable sshd.service

- For SSH to be truly effective, using insecure connection protocols should be prohibited

- Some services to be disabled are **telnet, rsh, rlogin,** and **vsftpd**

OpenSSH Clients

- To connect to an OpenSSH server from a client machine, you must have the **openssh-clients** package installed

- The **ssh** utility allows you to log in to a remote machine

- **Ssh** It is a secure replacement for the **rlogin, rsh,** and **telnet** programs.

- log in to a remote machine by using the following command:

  ssh   hostname

- This will log you in with the same user name you are using on the local machine

- If you want to specify a different user name, use a command in the following form

ssh username@hostname

- After entering the password, you will be provided with a shell prompt for the remote machine

- ssh program can be used to execute a command on the remote machine without logging in to a shell prompt

ssh [username@]hostname command

- To view the contents of this file on host penguin.example.com,

```
ssh  USER@penguin.example.com  cat /etc/redhat-release
```

ex

- The first time you initiate a connection, you will be presented with a message similar to this:

```
]$ ssh USER@penguin.example.com
The authenticity of host 'penguin.example.com' can't be established. ECDSA key fingerprint is 256
da:24:43:0b:2e:c1:3f:a1:84:13:92:01:52:b4:84:ff.
Are you sure you want to continue connecting (yes/no)?
```

- Type yes to accept the key and confirm the connection.

- You will see a notice that the server has been added to the list of known hosts, and a prompt asking for your password:

```
Warning: Permanently added
'penguin.example.com' (ECDSA) to the list of
known hosts.

USER@ penguin.example.com's password:
```

- After entering the password, you will be provided with a shell prompt for the remote machine.

- Users should always check if the fingerprint is correct before answering YES to the question in this dialog.

- The fingerprint can be checked by using the **ssh-keygen** command as follows:

  ]# **ssh-keygen -l -f /etc/ssh/ssh_host_ecdsa_key. pub**

  256 da:24:43:0b:2e:c1:3f:a1:84:13:92:01:52:b4:84:ff

  (ECDSA)

- **ssh** program can be used to execute a command on the remote machine

  ```
  ssh [username@]hostname command
  ```

Ex

```
]$ ssh USER@penguin.example.com
cat   /etc/redhat-release


USER@ penguin.example.com's password:
Red Hat Enterprise Linux Server release 7.0
(Maipo)
```

Using the **scp** Utility

- **scp** can be used to transfer files between machines over a secure, encrypted connection
- To transfer a local file to a remote system, use a command in the following form

```
scp  localfile  username@hostname:remotefile
```

Ex

```
]$ scp taglist.vim
USER@penguin.example.com:
.vim/plugin/taglist.vim

USER@ penguin.example.com's password:
taglist.vim
100% 144KB 144.5KB/s
00:00
```

- Multiple files can be specified at once.
- To transfer the contents of **.vim/plugin/** to the same directory

  scp .vim/plugin/* USER@penguin.example.com:.vim/plugin/

- To transfer a remote file to the local system, use the following syntax

```
scp  username@hostname:remotefile  localfile
```

- For instance, to download the **.vimrc** configuration file from the remote machine, type:

scp USER@penguin.example.com: .vimrc   .vimrc

Using the **sftp** Utility

- The **sftp** utility can be used to open a secure, interactive FTP session
- To connect to a remote system, use a command in the following form

sftp username@hostname

Ex

```
$ sftp  USER@penguin.example.com

USER@ penguin.example.com's password:
Connected to penguin.example.com.
sftp>
```

- After you enter the correct password, you will be presented with a prompt.
- The **sftp** utility accepts a set of commands similar to those used by **ftp**

set of commands in sftp

- **ls** [directory]
- **cd** directory
- **mkdir** directory
- **rmdir** path
- **put** localfile [remotefile]
- **get** remotefile [localfile]

X11 Forwarding by ssh

- To open an X11 session over an SSH connection, use a command

```
ssh  -Y  username@hostname
```

Ex

```
$ ssh  -Y  USER@penguin.example.com

USER@ penguin.example.com's password:
```

- Note that the X Window system must be installed on the remote system before X11 forwarding can take place

Port Forwarding

- SSH can secure otherwise insecure TCP/IP protocols via port forwarding
- Port forwarding works by mapping a local port on the client to a remote port on the server.

- SSH can map any port from the server to any port on the client.
- Port numbers do not need to match for this technique to work

- To create a TCP/IP port forwarding channel which listens for connections on the local host, use a command

ssh -L local-port:remote-hostname: remote-port username@hostname