June 26, 2007

UniForum
Chicago

SSH
The Secure Shell

Hemant Shah
shahhe@gmail.com

Platform: Linux and Unix
What is SSH?
What is SSH?

• The Secure Shell
• It is a protocol not a product
• Software based approach to network security
• Encrypts the data sent between the computers
• Client/Server architecture
• Comes with all Linux distribution, Mac OS X, AIX, Sun Solaris, OpenBSD and other Unix variants
• Ported to other operating systems, such as Windows, Palm OS, Amiga, etc.
• Other clients, such as, scp, sftp, etc. are also available
• Replacement for telnet, rlogin, rsh, rcp, ftp, etc.
What is SSH Not
What SSH is NOT

- It is not a true shell like csh, ksh, sh, etc.
- It is not a command interpreter
- It creates secure channel for running commands on remote computer
- It is not a complete security solution
- It will not protect against trojans, viruses, etc.
History
History

- In 1995, Tatu Ylönen, a researcher at Helsinki University designed the first version of the protocol (now called SSH-1)
- In July of 1995, he released SSH1 as free software
- In December of 1995 he formed SSH Communication Security to market and develop SSH
- In 1996 SSH-2 was developed, it was incompatible with SSH-1
- SCS released SSH-2 in 1998 and had more restrictive license
- IETF formed group called SECSH to standardize the protocol
- OpenSSH, free implementation of SSH-2 protocol was released from OpenBSD project.
- In 2006 IETF SECSH group released SSH-2 as internet standard (RFC 4251)
Terminology
SSH - Generic term used for SSH protocols
ssh - Client command for running remote command
sshd - Server program
SSH-1 - Version 1 of the protocol
SSH-2 - Version 2 of the protocol
OpenSSH - Product from open BSD project
SSH Architecture
SSH Architecture

The brown fox jumped over the cow

Anw@dc%9r&6cbditop*dekisn@h

ssh client

Authentication

Network

ssh server
## SSH Layers

<table>
<thead>
<tr>
<th>Layer</th>
<th>Functions</th>
</tr>
</thead>
</table>
| **Application Layer**        | **ssh-connection**  
Session multiplexing, X11 and port forwarding, remote command execution, SOCKS proxy, etc. |
|                              | **ssh-userauth**  
User authentication using public key, password, host based, etc. |
|                              | **ssh-transport**  
Initial key exchange and server authentication, setup encryption |
| **Transport Layer**          | TCP                                                                      |
| **Internet Layer**           | IP                                                                       |
| **Network Access Layer**     | Ethernet                                                                 |
SSH Connection Sequence

- A cryptographic handshake is made with the server
- The connection between client and remote server is encrypted using symmetric cipher
- Client authenticates itself
- Client can now interact safely with remote server over encrypted connection
SSH Features

• Strong encryption
• Strong authentication
• Authorization
• Integrity of communication
• Forwarding or tunneling
SSH will protect against

- Eavesdropping of data transmitted over the network
- Manipulation of data at intermediate elements in the network (e.g. routers)
- IP address spoofing where an attack host pretends to be a trusted host by sending packets with the source address of the trusted host
- DNS spoofing of trusted host names/IP addresses
- IP source routing
SSH will not protect against

• Incorrect configuration or usage
• A compromised root account
  ▪ If you login from a host to a server and an attacker has control of root on either side, he/she can listen to your session by reading from the pseudo-terminal device, even though SSH is encrypted on the network, SSH must communicate in clear text with the terminal device
• Insecure home directories: if an attacker can modify files in your home directory (e.g. via NFS) he may be able to fool SSH
Installing SSH
You may download the source from

http://www.openssh.com/

Read installation instructions to check if you have pre-requisite packages and libraries.
Building and installing OpenSSH

```
gtar -xzf openssh-4.5p1.tar.gz
cd openssh-4.5p1
./configure
make
make install
```
Configuration files
SSH Configuration Files

• SSH has two different sets of configuration files
  ▪ System wide configuration files
  ▪ User specific configuration files
The system wide configuration are stored in `/etc/ssh` directory

- `ssh_config` - Client configuration file. It is overridden by configuration file in user's home directory
- `sshd_config` - Configuration file for `sshd` server daemon
- `ssh_host_dsa_key` - The DSA private key used by the `sshd` daemon
- `ssh_host_dsa_key.pub` - The DSA public key used by the `sshd` daemon
- `ssh_host_rsa_key` - The RSA private key used by the `sshd` daemon for version 2 of the SSH protocol
- `ssh_host_rsa_key.pub` - The RSA public key used by the `sshd` for version 2 of the SSH protocol
System wide configuration files

sshd.pid - Server's PID is stored in this file
User specific configuration files

• The user specific configuration files are stored in ~UserName/.ssh directory

authorized_keys2 - This file holds a list of authorized public keys for users. When the client connects to a server, the server authenticates the client by checking its signed public key stored within this file

id_dsa - Contains the DSA private key of the user
id_dsa.pub - The DSA public key of the user
id_rsa - The RSA private key of the user
id_rsa.pub - The RSA public key of the user

known_hosts - This file contains DSA host keys of SSH servers accessed by the user. This file is very important for ensuring that the SSH client is connecting the correct SSH server
User specific configuration files

config - Client configuration file
Configuration files

- Specify authentication methods supported
- Specify SSH protocols supported
- Need to make trade-offs between security and easy-of-use
- Behavior of the server can be controlled in following order:
  - Compiling time configuration
  - Configuration file
  - Command line options
• Server configuration is stored in `/etc/ssh/sshd_config` file.
• Client configuration is stored in `/etc/ssh/ssh_config` and `~/.ssh/config` files.
• The file contains two types of entries:
  ▪ Comment or blank line
  ▪ Key/Value pair

**Example:**
```plaintext
# This is a comment line
Port 22
```
Server Recommendations
Server recommendations

Protocol
Possible values are 1 or 2

Protocol 2

Protocol 1 has been deprecated because of vulnerabilities, it is recommended that you do not support protocol 1.
Server recommendations

Port
Possible values are any integer less than 65535

Port 22
Server recommendations

ListenAddress
IP address of the system and optionally port number

ListenAddress 10.90.10.101
ListenAddress 10.90.10.102:12345

By default sshd will listen to all network interfaces, if you want to limit sshd to service only certain interface then use this option.
Server recommendations

TCPKeepAlive
Send TCP keepalive messages

TCPKeepAlive yes

If keepalive messages are not sent then server may not realize that the client has crashed. It will keep running and use resources. However, this means that connections will die if the route is down temporarily.
Server recommendations

Compression
CompressionLevel

Compression yes
CompressionLevel 6

Not needed on intranet.
Server recommendations

IgnoreRhosts

IgnoreRhosts yes
Server recommendations

UsePrivilegeSeparation

UsePrivilegeSeparation yes

Separates privileges by creating an unprivileged child process to deal with incoming network traffic.
Server recommendations

PermitRootLogin

PermitRootLogin no

Specifies whether root can login using ssh. The argument must be `yes', `without-password', `forced-commands-only' or `no'. The default is `yes'.
Server recommendations

Subsystem

Subsystem sftp /usr/local/libexec/sftp-server

Configures external subsystem (e.g. sftp server).
Client Recommendations
Client recommendations

Host

Host hostname or wildcard pattern.

Restricts the following configuration, up to the next Host keyword, to the matching host(s).
Client recommendations

BatchMode

BatchMode no

If set to yes, passphrase/password querying is disabled. This option is useful in scripts and other batch jobs.
Client recommendations

ForwardX11

ForwardX11 yes
Client recommendations

ForwardX11Trusted

ForwardX11Trusted yes

If the this option is set to ``yes'' then remote X11 clients will have full access to the original X11 display. If this option is set to ``no'' then remote X11 clients will be considered untrusted and prevented from stealing or tampering with data belonging to trusted X11 clients.
Client recommendations

IdentityFile

IdentityFile /path/to/private/key/file
**Client recommendations**

**HostName**

`HostName real hostname or IP address`

Specifies the real host name to log into. This can be used to specify nicknames or abbreviations for hosts.
Client recommendations

LocalForward

LocalForward port host:port

Specifies that a TCP/IP port on the local machine be forwarded over the secure channel to the specified host and port from the remote machine.
Client recommendations

Port

Port *port_number*

Specifies the port number to connect on the remote host.
Checking configuration

- To check the configuration run following command:

  sshd -t
Key Management
Key pairs

- ssh authenticates users using key pairs
  - private key
  - public key
Key management commands

- `ssh-keygen` - Create key pairs
- `ssh-agent` - Holds private key in memory
- `ssh-add` - Adds key to the key agent
Generating key pairs

shahhe@kubuntu1:~$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/shahhe/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/shahhe/.ssh/id_dsa.
Your public key has been saved in /home/shahhe/.ssh/id_dsa.pub.
The key fingerprint is:
Generating key pairs

shahhe@kubuntu1:~$ cat ~/.ssh/id_dsa
-----BEGIN DSA PRIVATE KEY-----
MIIBuwIBAAKBgQDPmC7jSBnJMoQ8o6/cF4GUDP/gsCqonA0UGs2g/92N8qVTxxZg
U3MgZAQ96FAsaGKFdfsxoqbpleXX7IXUS+erPOMQnDtbooLgZN3VpvStvV/hulnn
HoFJoDmoE5MnrY0Su93jZe2mPp4hOrYYQu0/8r3YRFtAazz6TCauHFxO4DQIVAJYo
apGVvbg8J1rAefSBReOef/iX AoGBAJUXbyDtR0wpyz5UKT11FmVS/a34ST9lfz1d
OjR38c9sRCf8k1RZ6IuqoLUZZ3jSo56+SRsraFQReCB5GLWPx5qKzHz9xi9XFseTa
Cb3Qh70EbiP3uAFqnTv k2K8voKC4dNIEXZ7SZXZUsWBImLaY Xf/scvL7fM1My9df
fCMf8By2AoGAGEEdK17lr1D7zfWUVyJm+26Z aQ/QU4Yhff6Cfoellnq/1UmT6SEVF
SZWs j9n8fj7Ez8103gU/g+otZXdcsS6OmNMooWkADImkHfQ6o eOklh/3z0hV8TY4
HnOtMZuHJMf1LPFNoINbenLS+qldGvi19aTxZUKcQJiHpdr6GR3jn9cCF E9xHd8q
Y8k1JEyIPYK+KQ4UrbhZ
-----END DSA PRIVATE KEY-----
Generating key pairs

shahhe@kubuntu1:~$ cat ~/.ssh/id_dsa.pub
ssh-dss AAAAB3NzaC1kc3MAAACCBAM+YLuNIGckyhDyjr9wXgZQM/+CwKqicDRQazaD/3Y3ypVPHFmBTcyBkBD3oUCxoYoUN+zGipunV5dfshdRL56s84xCcOluiguBk3dWm9K29X+G6WecegUmgoagTkyetjRK73eN17aY+niE6thhC7T/yvdhEW0DPPpMJq4cXE7gNAAAAFQCWKGqRlb24PCdawHn0gUXjnn/4lwAAAIEA1RdvIO1HTCnlPlQpPXUWZVL9rfhJP0t/OV06NHfxz2xEJ/ySVFnoi6qgtRlnrNJKjnr5JGytoVBF4IHzYtY/HmorfP3GL1cWx5NoJvCHvQRuI/e4AWqdO+TYry+goLh00gRdntJld1SxYEiYtphd/+xy8vt8yUzLl18Ix/wHLYAAACAGEkK171r1D7zfWUVyJm+262aQ/QU4Yhff6Cfoe1lnq/1UmT6SEVfSZWsj9n8fj7Ez8103gU/g+otZXDcsS6OmNoWkADIkHfQ6oeoK1h/3z0hV8TY4HnOtMZuHJMf1LPFNvINbenLS+qldGvi19aTxZUkcQJiHdpr6GR3jn9c= shahhe@kubuntu1
Executing commands
Logging into remote system

shahhe@kubuntu1:~$ ssh shah@xnet.com

* Problems? Questions? Email: help@xnet.com
* Type "whatsup" to see information posted to our "What's Up?" page.

You have mail.
You have 17 read messages.
You have no new mail.

/home/customer/shah
{shah@typhoon} 1>
shahhe@kubuntu1:~$ scp .profile shah@xnet.com:tmp/profile.kubuntu
.profile  100%  566  0.6KB/s  00:00

shahhe@kubuntu1:~$ scp shah@xnet.com:tmp/profile.kubuntu tmp/.
profile.kubuntu  100%  566  0.6KB/s  00:00
Executing commands on remote system

shahhe@kubuntu1:~$ ssh shah@xnet.com ls
Mail
News
bin
mail
public_html
tmp
Executing commands on remote system

shahhe@kubuntu1:~$ ssh -Y shah@xnet.com /opt/sfw/bin/xterm
Force execution of command
To force an execution of a command use **command** keyword in authorized_key2 file.

```plaintext
command="~/bin/DumpEmpNames",no-port-forwarding ssh-dss
AAAAB3NzaC1kc3MAAACBAIB8B1MvY1WnVeyPE6bMwrTr1OM8O2HXiQQKq9801q
fmOf9x3QYZzXVFegdNYDtN4o1sr6T7bmCNvOTC7szOg1aFIbfQoHfmIexabVyz
xin/2d2Juof7YU53Zrx1BjHKzqQpCj6jx7FxjPqlLD0BvL9R3qoPIpJ6Jt0YvY
Ae4Zj9AAAAAFQDoejxCMgfZ0O/Zxwxn3mFidTpogQAAI1BDQvrhRsDFhA1UUkBO
203pVujfnNYF7X58mD/WPGZ+Z4aR8dGuD21X7hC6M8ko9a9wLLYigELSkuWiWps
VZ/NJyBxhrCCD3YCNXeltJ7L0KaWGP96H2KkDtYsP7RMhAmztVpm1OrPzXbIpU
3jpq8dRJqUksG8mq2dbXPBWgh9xHyQAAAIBG9iwGfjPLDTH1niXk5tbZQUuEGk
GZzCaBw8jJlKPXMWeE7rVmBXV5sC/zhcX30AXUNj8OUpaFzZbxtnnzIgnehW
duWTWmiQPOi2f8oV9fCulpFnYWGNn4VhmqD1ScWNoIe3ObV05WTerdYJAY8bv
2Zfh9EJJEJvFFerdur/g== Key for Dumping active user names.
```
Force execution of command

• Execute command as follows:

    ssh -i ~/keys/DumpEmpNames.dsa user@remotehost
Options for authorized_keys2 file

<table>
<thead>
<tr>
<th>Option</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>command=&quot;command name&quot;</td>
<td>Specify a force command</td>
</tr>
<tr>
<td>environment=&quot;variable=value&quot;</td>
<td>Set environment variable</td>
</tr>
<tr>
<td>from=&quot;host or ip address&quot;</td>
<td>Limit incoming hosts</td>
</tr>
<tr>
<td>no-agent-forwarding</td>
<td>Disable forwarding agent</td>
</tr>
<tr>
<td>no-port-forwarding</td>
<td>Disable port forwarding</td>
</tr>
<tr>
<td>no-pty</td>
<td>Do not allocate TTY</td>
</tr>
<tr>
<td>no-x11-forwarding</td>
<td>Disable X11 forwarding</td>
</tr>
</tbody>
</table>
Port forwarding
Port forwarding

Client App

ssh client

Secure

Server App

ssh server
Port forwarding

• Create SSH tunnel

```bash
ssh -f -N -L10112:localhost:80 www.example.com
```

• Add to ~/.ssh/config file and run ssh command

```bash
Host webtunnel
  User shahhe
  Hostname www.example.com
  LocalForward 10112 www.example.com:80
```

```bash
ssh -f -N webtunnel
```
Port forwarding

- Start application using port on localhost

  firefox http://localhost:10112
Agent forwarding
Agent forwarding

You want to login to the computer at work from your home computer or from hotel while traveling. The computer at work is behind the firewall so you cannot connect to it directly.

You are allowed to connect to a bastion host, but are not allowed to store private keys on it.

What can you do?
Agent Forwarding

Home

Bastion

sshd
(proxy agent)

ssh

Work

Login

ssh

sshd
Port 46464
Protocol 2
PasswordAuthentication no
X11Forwarding yes
Compression no
Subsystem sftp /usr/libexec/sftp-server
The configuration is stored in `~/.ssh/tunnel.cfg` file.

Host *
   ForwardX11 yes
   ForwardAgent yes
   NoHostAuthenticationForLocalhost yes

Host bastionhost
   User RemoteUser
   IdentityFile /home/LocalUser/.ssh/work_dsa
   HostName 69.2.50.60
   Port 46464
ssh client configuration on home system

Host *
  ForwardX11 yes
  ForwardAgent yes
  NoHostAuthenticationForLocalhost yes

Host portmap
  HostName localhost
  LocalForward 10001 10.60.80.101:22
  LocalForward 10002 10.60.80.102:22

Host host1
  User RemoteUser
  IdentityFile /home/LocalUser/.ssh/work_dsa
  HostName localhost
  Port 10001

Host host2
  User RemoteUser
  IdentityFile /home/LocalUser/.ssh/work_dsa
  HostName localhost
  Port 10002
Setting Key Pair

- Generate key with password
- Store private key on **Home** system
- Store public key on **Bastion** host
- Store public key on **Work** system
Login into work systems

Do the following on the HOME system:

• Start ssh-agent and add the key
  ssh-agent $SHELL
  ssh-add ~/.ssh/work_dsa

• Create tunnel to bastion host
  ssh -f -N -F ~/.ssh/tunnel.cfg bastionhost
  ssh -f -N UserName@portmap

• Login in to work systems
  ssh host1
  ssh host2
# Environment variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSH_CONNECTION</td>
<td>Client and server socket information</td>
<td>10.90.10.107 45756 10.90.10.182 22</td>
</tr>
<tr>
<td>SSH_AUTH_SOCK</td>
<td>Path to socket</td>
<td>/tmp/ssh-FcRCl22249/agent.22249</td>
</tr>
<tr>
<td>SSH_CLIENT</td>
<td>Client socket information</td>
<td>10.90.10.107 45756 22</td>
</tr>
<tr>
<td>SSH_TTY</td>
<td>Name of TTY</td>
<td>/dev/pts/48</td>
</tr>
</tbody>
</table>
Other ssh based applications
Other ssh based applications

• sshfs - ssh based file system client
  
  http://fuse.sourceforge.net/sshfs.html

• sftp - secure file transfer. Part of OpenSSH
  
  http://www.openssh.com/
OpenSSH alternatives for windows

PuTTY
TTSSH
Cygwin
MSSH
WinSCP
FileZilla
Advantages of using ssh
Advantages

- Proven technology
- Strong encryption
- Both free and commercial versions exist
- Runs on many platforms
- Tunneling of ports works well and can be used for simple VPNs
- Many authentication methods supported
- Can be SOCKS5 proxy aware
- Use it instead of VPN
Disadvantages of using ssh
Disadvantages

• Port ranges & dynamic ports can't be forwarded
• SSH server daemon:
  ▪ Cannot restrict what ports may or may not be forwarded, per user
  ▪ When a user is authenticated by password, the client's RSA identity is not verified (against ssh_known_hosts). The verification only takes place when .[sr]hosts trust is used
• Port forwarding can also introduce security problems. The SSH server doesn't allow detailed configuration of what forwarding is allowed from what client to what server etc.
• A client on the Internet that uses SSH to access the Intranet, can expose the Intranet by port forwarding
Resources

http://www.openssh.com/

http://fuse.sourceforge.net/sshfs.html


SSH FAQ
http://www.employees.org/~satch/ssh/faq/ssh-faq.html