An Introduction to OpenSSH on a Local/Home Network.

What is SSH(Secure SHell)? (Reference `man ssh`)

ssh is a program which provides secure client/server communication on Port 22 (default)
ssh client can log into a remote machine running **sshd** and execute commands on that machine.
sshd is the **ssh d**aemon program for the **ssh** server. **sshd** listens for connections from **ssh** clients. It forks a new daemon for each incoming **ssh** connection. The forked daemons handle key exchange, encryption, authentication, command execution, and data exchange.
ssh has two versions, protocol 2 is current, protocol 1 is less secure and obsolete. Always use 2.

Related programs.

scp (**s**ecure **c**o**p**y) is an interactive remote file copy program which uses **ssh** transport.
sftp (**s**ecure **f**ile **t**ransfer **p**rogram) provides interactive file transfer using **ssh** transport.
rsync (**r**emote **sync**)is a fast and extraordinarily versatile file copying & remote sync tool. It can utilise **ssh** transport to securely copy/sync to/from a remote host. **rsync** behaves in a similar fashion to **rcp**, but has many more options and uses the **rsync** remote-update protocol to greatly speed up file transfers when the destination file is being updated.
sshfs (**ssh f**ile **s**ystem) is a FUSE (**F**ilesystem in **U**ser**s**pac**e**) client utilising ssh transport. The user can seamlessly interact with remote files being securely served over **ssh** just as if they were local files on the users computer.

/etc/hosts file.

The /etc/hosts is a plain text data base that maps hostnames to IP addresses and is usually referred to prior to calling any other DNS.
For each local **ssh** client & server a single line should be added to **/etc/hosts**.

```
IP_address      canonical_hostname     [aliases...]
```

example

```
192.168.0.2     tre-01      bedroom
```

Note the **canonical_hostname** can be any name you choose to assign to the **IP_address**. For a local network it is usually the host name in **/etc/hostname** of the machine to be addressed. For a server on the Internet the FQDN (fully qualified domain name) of the host server can be used as the canonical hostname or as an alias. Each name (including any FQDN) on the right of the IP address will be treated as a hostname for that IP address
An example of a FQDN for an Internet device which has local hostname *myhost* and a parent domain name *example.com*, has a FQDN *myhost.example.com.*

Secure Communication has two principle elements.

Encryption - to keep the communication secret.
Encryption should be done before Authorisation to ensure passwords are hidden. ssh uses the asymetric Diffie–Hellman key agreement which enables two parties to establish a shared cryptographic key This key can then be used to encrypt subsequent communications using a symmetric key cipher (eg RSA).
Authorisation to ensure both parties communicating with a known and authorised host.
This is done by using a password known to both parties or RSA authentication key

Installation Preparation.

First gather the client & server IPs, hostnames and other useful information you may need.
The IP address of a machine on the local network can be found with the command **ifconfig**
The host name of a machine is stored in the file **/etc/hostname**
The Internet facing IP of your local network can be found at *www.**whatismyip.com***
This article is written for the Linux Bash version of **OpenSSH**. **/etc/passwd** shows users login shell.

File Permissions.

SSH files must be installed with the correct **U**ser-**G**roup-**O**thers permissions to ensure security, and correct operation.

		Binary	Octal
Read Write Execute	**rwx**	**111**	**7**
Read Write	**rw-**	**110**	**6**
Read Execute	**r-x**	**101**	**5**
Read	**r--**	**100**	**4**

example usage of change mode **`sudo chmod 600 ~/.ssh/id_rsa`**

Client User `~/.ssh/`		**bm**	**Client Global** `/etc/`		**bm**
`~/.ssh/`	700U		**`/etc/hosts`** □	644R	cf02
`~/.ssh/config`	600U	cf03	**`/etc/ssh/ssh_config`** □	644R	cf03
`~/.ssh/authorized_keys`	600U	cf06	Note: □ indicates file usually present on a new linux install.		
`~/.ssh/id_rsa.pub`	644U	cf05			
`~/.ssh/id_rsa`	600U	cf05			
`~/.ssh/known_hosts`			**`/etc/ssh/ssh_known_hosts`**		
Server User			**Server Global** `/etc/`		
			`/etc/ssh/sshd_config`	644R	cf04

Install the Client and the Server software

An SSH device can be an SSH client an SSH server, or both Client and Server
The following command will install both **`sshd`** server and **`ssh`** client and on to a computer
`sudo apt-get install openssh-server openssh-client`

Test your setup with the following commands:
`ssh localhost` this tests ssh against your local sshd daemon. **`Ctrl-D`** exits.
`ssh username@server-hostname` will connect your client to a remote sshd host

The **`/etc/hosts.allow`** and **`/etc/hosts.deny`** files Reference **`man 5 hosts_access`**
Once SSH tests OK you may need to tune your **`/etc/hosts.allow`** and **`/etc/hosts.deny`** files if you plan to expose **`sshd`** to the Internet. There are various "How to" guides available on the internet describing how to security harden **`sshd`** servers using **`/etc/hosts.allow`** and **`/etc/hosts.deny`** files.

`~/.ssh/authorized_keys` (Ref. **`man 5 authorized_keys (same as 'man sshd'))`**
Protocol 2 supports both RSA and DSA keys each host has a host-specific key, (by default 2048) bits, used to identify the host.
/etc/ssh/ssh_known_hosts and **~/.ssh/known_hosts** files contain host public keys for all known hosts.

Start, Stop & Restart `ssh/sshd`

After changes to config files, you must restart SSH for the changes to take effect.

sudo service ssh start or
sudo /etc/init.d/ssh start starts your local sshd daemon.

sudo service ssh stop or
sudo /etc/init.d/ssh stop stops your local sshd daemon.

sudo service ssh restart or
sudo /etc/init.d/ssh restart restarts your local ssh server

Tricks with Tilde ~ the (default) escape character.

The Tilde escape sequence must be immediately at the start of a new line.
Because Tilde is used as an escape character it does not print. If you see a tilde at the beginning of the new line it means you have pressed tilde twice and the sequence will not work.
To close a link use **Ctrl-D** or **Tilde dot** (**~.**) Ifyou are having trouble closing a failing/failed link with **Ctrl-D** try **Tilde dot** (**~.**). **Tilde dot** (**~.**) often works when **Ctrl-D** fails
Tilde - Ctrl-Z suspends the connection
Tilde - question mark (~?) displays a list of all the supported escape sequences.

The default escape character can be temporarily changed at the beginning of session using the **-e** option at the start **ssh** command line. The argument should be a single character, (eg **^**) or the word "**none**" which will disable the escape character making the connection transparent for binary data.
Examples: **ssh -e ^ username@server-hostname** or
ssh -e none username@server-hostname

ssh configuration files (Reference **man 5 ssh_config**).

ssh takes its configuration data from the following sources in the priority order shown,

1. command-line options
2. **~/.ssh/config** user config, must have strict read/write permissions:for the user.
3. **/etc/ssh/ssh_config** system-wide config, permissions must be world-readable.

sshd configuration files (Reference **man 5 sshd_config**).

sshd takes its configuration data from
/etc/ssh/sshd_config.

Ports

A port number is an extension to an IP address that enables TCP to uniquely identify a particular process running on a computer. The IP address + port number combined together are globally unique.

example.com:5022 specifies the use of port 5022 on the address **example.com**

Using a non-standard Port

By default the **ssh** server runs on TCP port 22. This can make you an easy target on the Internet For security purposes you may choose any port in the range 49152–65535

To change ports log on to the server open the **sshd_config** file and look for the line *Port 22* and change to *Port 50022*. Restart the sshd daemon with **sudo service ssh restart**.

sshd is now running on a non-standard port, so your command to the **ssh** client must specify the port.
$ ssh -p 50022 user@server> or it can be specified on a per-host basis in the **ssh_config** file.
scp supports same option but uses an upper-case **P.**

SSH Public-Key authentication (No password required)**.**

ssh-keygen generates, manages and converts authentication keys for **ssh**. The type of key is specified with the **-t** option. The default is an RSA key for SSH protocol 2. Because SSH is the transport for other

services such as SCP (secure copy), SFTP (secure file transfer), and other services (CVS, etc), this can be very convenient and save a lot of typing. Public-Key authentication is convenient but anyone who has read access to your Private Key can use it to access your remote servers.

Generate Private and Public RSA Keys.

When the client Private and Public RSA Keys are generated they will be stored in two files
~/.ssh/id_rsa (rsa Private Key)
~/.ssh/id_rsa.pub (rsa Public Key)
Check on the Client **ls -al ~/.ssh** to see if there is an existing Public and Private Key key setup.
If you see **id_rsa** and **id_rsa.pub**, your computer already has a RSA Public and Private Key setup, otherwise do.**ssh-keygen -t rsa -b 4096 -C "your-email"** -**b** specifies the number of bits in the key (default 2048) -**C** is a comment option which is appended to the end of the public key.
To read public key and comments **less ~/.ssh/id_rsa.pub**

Now just press enter and follow the prompts.
(References **man 1 ssh-keygen**
https://help.github.com/articles/generating-ssh-keys)

The output from the **ssh-keygen** command will be similar to that shown below.

Generating public/private rsa key pair.
Enter file in which to save the key (/home/hayden/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/hayden/.ssh/id_rsa.
Your public key has been saved in /home/hayden/.ssh/id_rsa.pub.
The key fingerprint is:(Note the fingerprint is a hash of the public key)
9b:24:2a:c0:40:f1:cc:da:ae:b1:0e:8a:ba:7f:b1:b2 hayden@tre-01
The key's randomart image is:

```
 +--[ RSA 2048]-----+
 | o.                |
 |. +                |
 |.  +               |
 |o o                |
 |.o .  . S          |
 | .. .. o o         |
 |.....o  o          |
 |+ =.o              |
 |B*E+               |
 +-------------------=--+
```

The fingerprint is a hash used to authenticate or look up a public key.
The randomart is intended to be an faster visual recognition method than comparing fingerprints.
If a fingerprint or randomart unexpectedly changes you may be the victim of a man-in-the-middle attack, so do not enter your password.

Now enter **ls -al ~/.ssh** it should show your **id_rsa** and **id_rsa.pub** Key pairs. These keys are each one long line, your text viewer will probably automatically word wrap the line. To view the tesxt contained in the Keys enter
less ~/.ssh/id_rsa
less ~/.ssh/id_rsa.pub
To view your fingerprint and randomart use
ssh-keygen -lv -f ~/.ssh/id_rsa (Private Key)
ssh-keygen -lv -f ~/.ssh/id_rsa.pub (Public Key)

If you add this **VisualHostKey=yes** to your **~/.ssh/config**:
Or add this option to your **ssh** command
ssh -o VisualHostKey=yes servername you will see your servers randomart each time you login (handy to ensure you are deploying to the right server) .

Setup the Server.

Log onto the server
To ensure everything is setup on correctly on the server. SSH into the server and run
ls -al ~/.ssh

If there is no **~/.ssh** on the server create one. **mkdir -m 644 ~/.ssh**
Now create an **authorized_keys** file. with the command **touch ~/.ssh/authorized_keys**

Run **ls -al ~/.ssh** and check for the **authorized_keys** file.
Next make sure that permissions are OK
chmod 700 ~/.ssh
chmod 600 ~/.ssh/authorized_keys

To append the content of the client users Public Key file to the server's user file **authorized_keys**, file first create the file by logging into the server and enter **touch ~/.ssh/authorized_keys**
then enter from the client user the following command.
cat ~/.ssh/id_rsa.pub | ssh user@host 'cat >> .ssh/authorized_keys'
cat ~/.ssh/id_rsa.pub | ssh hayden@lounge 'cat >> .ssh/authorized_keys'
Protecting your Private Key. Use the passphrase option when setting up your private key. You can use **ssh-agent(1)** and **ssh-add(1)** to type your passphrase only once for all uses of a specific key in a session. You can automatically load all your keys in the agent by adding the following lines to your **~/.xsession** fileRef **man 5 Xsession.options**:

```
# if use-ssh-agent is specified in /etc/X11/Xsession.options (default)
# then you need only the second line
# eval ssh-agent
ssh-add
```

The package **ssh-askpass** must be installed if you intend to run **ssh-add** without a terminal