Setup SSH public/private keys and disabling password login, allowing only scp for specific users and using chroot

SSH (Secure Shell) can be set up with public/private key pairs so that you don’t have to type the password each time. Because SSH is the transport for other services such as SCP (secure copy), SFTP (secure file transfer), and other services (CVS – Concurrent Versions System, etc), this can be very convenient and save you a lot of typing. While SSH2 can use either DSA or RSA keys, SSH1 cannot. SSH only does the authentication using RSA or DSA algorithm, the “rest” is encoded using a cipher (like IDEA, DES, Blowfish, etc, etc). SSH2 will also not use patented cypers like IDEA .

Here is how to generate the SSH Version 2 keys:

- Type ssh-keygen -t dsa for DSA or ssh-keygen -t rsa -b 4096 for RSA into shell
- Just press enter to leave the default location and no passphrase.
- Now, copy your public key component located in ~/.ssh (id_dsa.pub or id_rsa.pub – whatever you created) to the remote machine in ~/.ssh/authorized_keys (authorized_keys is a file, not a folder)

Create DSA or RSA key pair.

Creating the DSA key

```
su@www:~$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/su/.ssh/id_dsa):
Created directory '/home/su/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/su/.ssh/id_dsa.
Your public key has been saved in /home/su/.ssh/id_dsa.pub.
The key fingerprint is:
1d:5d:00:55:73:39:0e:4b:75:3c:88:07:e5:98:67:6c su@www
The key's randomart image is:
+--[ DSA 1024]----+
|          .+*+*o+|
|           o=* *o|
|          .o+E+ o|
|         . .+. . |
|        S .      |
|                 |
|                 |
|                 |
|                 |
+-----------------+
```

Creating the RSA key

```
su@www:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/su/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/su/.ssh/id_rsa.
Your public key has been saved in /home/su/.ssh/id_rsa.pub.
The key fingerprint is:
80:f8:c6:c0:16:bb:52:f2:85:57:8a:97:9b:78:92:df su@www
The key's randomart image is:
+--[ RSA 4096]----+
|   .   .         |
|  . * =          |
|. X B .          |
|  = @ o .        |
|. * B   S        |
|  . = .          |
|     . E         |
|                 |
|                 |
+-----------------+
```

When the key is generated, copy it to the target machine using either of the two commands:

```
ssh-copy-id user@123.123.123.123
cat ~/.ssh/id_rsa.pub | ssh user@123.123.123.123 "mkdir -p ~/.ssh && cat >>
~/.ssh/authorized_keys"
```

Where ‘user’ is the remote systems existing user and ‘123.123.123.123’ is the remote systems IP. The first command will copy all generated keys, but the second command must be run separately for each generated key.

Deciding on which key pair to use

- DSA (Digital Signature Algorithm)
- RSA (Rivest-Shamir-Adleman)

DSA is faster in signing, but slower in verifying. A DSA key of the same strength as RSA (1024 bits) generates a smaller signature. A RSA 512 bit key has been cracked, but only a 280 DSA key. Also note that DSA can only be used for signing/verification, whereas RSA can be used for encryption/decrypt as well.

Using different keys on different hosts

Create multiple keys and create ~/.ssh/config file:

```
Host server1
IdentityFile ~/.ssh/key_file1

Host server2
IdentityFile ~/.ssh/key_file2
```

Prevent users from loging in using password, and disable root ssh login

To prevent login without password

```
sudo vim /etc/ssh/sshd_config
#Find ChallengeResponseAuthentication and set to no:
ChallengeResponseAuthentication no

#Find PasswordAuthentication set to no
PasswordAuthentication no

#Find UsePAM and set to no:
UsePAM no

#Find PermitRootLogin and set to no:
PermitRootLogin no
```

Prevent users from logging in, and allow only scp into chroot directory.

/etc/ssh/sshd_config

```
Subsystem sftp internal-sftp
Match Group sftponly
        ChrootDirectory /chroot/deploy/%u
        ForceCommand internal-sftp
```

Create /chroot/deploy/%u/%u directory (this is not a typo), and the limit access to it.

```
mkdir -p /chroot/deploy/user/user
chmod -R 0775 /chroot
chown -R root:root /chroot
chown -R user:user /chroot/deploy/user/user
```

Create a group 'sftponly', add users to it and restart ssh server

FAQ:

Q: I follow the exact steps, but ssh still ask me for my password!
A: Check your remote .ssh directory. It should have only your own read/write/access permission (octal 700) % chmod 700 ~/.ssh