Upload image

May 28, 2020

Use SSH Public Key Authentication

Natapol Pornputtapong[1]

[1]Faculty of Pharmaceutical Sciences, Chulalongkorn University

1 *Works for me* This protocol may be deleted by the owner

S3Bio

Natapol Pornputtapong
Faculty of Pharmaceutical Sciences, Chulalongkorn University

1

ssh-keygen

$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/yourname/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/yourname/.ssh/id_rsa.
Your public key has been saved in /Users/yourname/.ssh/id_rsa.pub.
The key fingerprint is:
d7:21:c7:d6:b8:3a:29:29:11:ae:6f:79:bc:67:63:53 yourname@machine
The key's randomart image is:

```
+--[ RSA 2048]----+
|            |
|        . o  |
|     .  .*. |
|    . .  = o  |
|     o S . o  |
|    . . o oE  |
|   . .oo +.   |
|    .o.o.*.   |
|    ....= o   |
+-----------------+
```

ssh-keygen generates, manages and converts authentication keys for ssh(1). ssh-keygen can create keys for use by SSH protocol version 2.

The type of key to be generated is specified with the -t option. If invoked without any arguments, ssh-keygen will generate an RSA key.

ssh-keygen is also used to generate groups for use in Diffie-Hellman group exchange (DH-GEX). See the MODULI GENERATION section for details.

Finally, ssh-keygen can be used to generate and update Key Revocation Lists, and to test whether given keys have been revoked by one. See the KEY REVOCATION LISTS section for details.

Normally each user wishing to use SSH with public key authentication runs this once to create the authentication key in ~/.ssh/id_dsa, ~/.ssh/id_ecdsa, ~/.ssh/id_ed25519 or ~/.ssh/id_rsa. Additionally, the system administrator may use this to generate host keys, as seen in /etc/rc.

Normally this program generates the key and asks for a file in which to store the private key. The public key is stored

in a file with the same name but ".pub" appended. The program also asks for a passphrase. The passphrase may be empty
to indicate no passphrase (host keys must have an empty passphrase), or it may be a string of arbitrary length. A passphrase is similar to a password, except it can be a phrase with a series of words, punctuation, numbers, whitespace,
or any string of characters you want. Good passphrases are 10-30 characters long, are not simple sentences or otherwise
easily guessable (English prose has only 1-2 bits of entropy per character, and provides very bad passphrases), and contain a mix of upper and lowercase letters, numbers, and non-alphanumeric characters. The passphrase can be changed later
by using the -p option.

There is no way to recover a lost passphrase. If the passphrase is lost or forgotten, a new key must be generated and the corresponding public key copied to other machines.

ssh-keygen will by default write keys in an OpenSSH-specific format. This format is preferred as it offers better protection for keys at rest as well as allowing storage of key comments within the private key file itself. The key comment may be useful to help identify the key. The comment is initialized to ``user@host'' when the key is created, but can be changed using the -c option.

It is still possible for ssh-keygen to write the previously-used PEM format private keys using the -m flag. This may be used when generating new keys, and existing new-format keys may be converted using this option in conjunction with the -p
(change passphrase) flag.

After a key is generated, instructions below detail where the keys should be placed to be activated.
linux

2

ssh-copy-id

$ ssh-copy-id user@adress

ssh-copy-id is a script that uses ssh(1) to log into a remote machine (presumably using a login password, so password authentication should be enabled, unless you've done some clever use of multiple identities). It assembles a list of one
or more fingerprints (as described below) and tries to log in with each key, to see if any of them are already installed (of course, if you are not using ssh-agent(1) this may result in you being repeatedly prompted for pass-phrases). It then assembles a list of those that failed to log in, and using ssh, enables logins with those keys on the remote server. By default it adds the keys by appending them to the remote user's ~/.ssh/authorized_keys (creating the file, and directory, if necessary). It is also capable of detecting if the remote system is a NetScreen, and using its `set ssh pka-dsa key ...' command instead.
linux