# Table Of Contents

nixCraft: Linux Tips, Hacks, Tutorials, And Ideas In Blog Format
http://www.cyberciti.biz/



# keychain: Set Up Secure Passwordless SSH Access For Backup Scripts

Posted by Vivek Gite <vivek@nixcraft.com>

[1]

We establish connections to remote systems without supplying a password, however I do not want to store my password less keys ( passphrase-free keys) on my servers. ssh-agent, takes care of keys with passphase, which allowing me to easily have ssh-agent process per system per login session. How do I dramatically reduces the number of times I've to punch my passphrase from once per new login session to once every time my local server is rebooted? How do I use keychain utility for all my backup scripts for secure passwordless login?


[2]

OpenSSH offers RSA and DSA authentication to remote systems without supplying a password. keychain is a special bash script designed to make key-based authentication incredibly convenient and flexible. It offers various security benefits over passphrase-free keys.

## How Does Keychain Make It Better Than a Key Less Passphrase?

If attacker broken into server with passphrase-free keys, all other your servers / workstation on which keys are used are also security risk (they can be easily breached). With keychain or ssh-agent attacker won't able to touch your remote systems without breaking your passphrase. Another example, if your laptop or harddisk stolen, an attacker can simply copy your key and use it anywhere as it is not protected by a passphrase.

keychain is a manager for ssh-agent, typically run from ~/.bash_profile. It allows your shells and cron jobs to share a single ssh-agent process. By default, the ssh-agent started by keychain is long-running and will continue to run, even after you have logged out from the system. If you want to change this behavior, take a look at the --clear and --timeout options, described below. Our sample setup is as follows:

```
peerbox.nixcraft.net.in => Remote Backup Server. Works in pull only mode. It will backup se
vivek-desktop.nixcraft.net.in => My desktop computer.
server1.nixcraft.net.in => General purpose remote server.
server2.nixcraft.net.in => General purpose remote web / mail / proxy server.
```

Install keychain software on peerbox.nixcraft.net.in so that it can login securely to other two servers for backup.

## Install keychain on CentOS / RHEL / Fedora Linux

You need RPMForge repo [3] enabled to install keychain package.

```
# yum install keychain
```

## Install keychain on Debian / Ubuntu Linux

```
# apt-get update && apt-get install keychain
```

## Install keychain on FreeBSD

```
# portsnap fetch update
# cd /usr/ports/security/keychain
# make install clean
```

## How Do I Setup SSH Keys With passphrase?

Simply type the following commands:

```
$ ssh-keygen -t rsa
```

OR

```
$ ssh-keygen -t dsa
```

Assign the pass phrase when prompted. See the following step-by-step guide for detailed information:

1. Howto Linux / UNIX setup SSH with DSA [4] public key authentication (password less login)
2. Howto use multiple SSH keys for password [5] less login

## How Do I Use Keychain?

Once OpenSSH keys are configured with a pass phrase, update your $HOME/.bash_profile file which is your personal initialization file, executed for login BASH shells:

```
$ vi $HOME/.bash_profile
```

Append the following code:

```
### START-Keychain ###
# Let  re-use ssh-agent and/or gpg-agent between logins
/usr/bin/keychain $HOME/.ssh/id_dsa
source $HOME/.keychain/$HOSTNAME-sh
### End-Keychain ###
```

Now you've keychanin configured to call keychain tool every login. Just log out and log back in to server from your desktop to test your setup:

```
$ ssh root@www03.nixcraft.net.in
```

Sample Output:


[6]

Fig.01 - Keychain in Action

keyhcain is up and running. Now, all you have to do is append your servers key file $HOME/.ssh/id_dsa.pub to other UNIX / Linux / BSD boxes:

```
# scp $HOME/.ssh/id_dsa.pub server1.nixcraft.net.in:~/pubkey
# scp $HOME/.ssh/id_dsa.pub server2.nixcraft.net.in:~/pubkey
# ssh server1.nixcraft.net.in cat ~/pubkey >> ~/.ssh/authorized_keys2; rm ~/pubkey
# ssh server2.nixcraft.net.in cat ~/pubkey >> ~/.ssh/authorized_keys2; rm ~/pubkey
# ssh root@server1.nixcraft.net.in
# ssh user@server2.nixcraft.net.in
```

### Task: Clear / Delete All Of Ssh-agent's Key

```
# keychain --clear
```

## Security Task: Make Sure Intruder Cannot Use Your Existing SSH-Agent's Keys (only allow cron jobs to use password less login)

The idea is pretty simply only allow backup shell scripts and other cron job to do password less login but all users including an intruder must provide a passphrase-key for interactive login. This is done by deleting all of ssh-agent's keys. This option will increases security, it still allows your cron jobs to use your ssh keys when you're logged out. Update your ~/.bash_profile as follows:

```
/usr/bin/keychain --clear $HOME/.ssh/id_dsa
```

If you are using RSA, use:

```
/usr/bin/keychain --clear $HOME/.ssh/id_rsa
```

Now, just log in to remote server box once :

```
$ ssh root@peerbox.nixcraft.net.in
```

Log out (only grant access to cron jobs such as backup)

```
# logout
```

### Task: Use Keychain With Backup Scripts for Passwordless login via cron

Add the following before your rsync, tar over ssh or any other network backup command:

```
source $HOME/.keychain/$HOSTNAME-sh
```

Here is a sample rsync script:

```
#!/bin/bash
# Remote Server Rsync backup Replication Shell Script
# Local dir location
LOCALBAKPOINT=/iscsi
LOCALBAKDIR=/backups/server1.nixcraft.net.in/wwwroot
# Remote ssh server setup
SSHUER=root
SSHSERVER=server1.nixcraft.net.in
SSHBACKUPROOT=/wwwroot

# Make sure you can log in to remote server without a password
source $HOME/.keychain/$HOSTNAME-sh

# Make sure local backup dir exists
[ ! -d ${LOCALBAKPOINT}${LOCALBAKDIR} ] && mkdir -p ${LOCALBAKPOINT}${LOCALBAKDIR}

# Start backup
/usr/bin/rsync --exclude '*access.log*' --exclude '*error.log*' -avz -e 'ssh ' ${SSHUER}@$

# See if backup failed or not to /var/log/messages file
[ $? -eq 0 ] && logger 'RSYNC BACKUP : Done' || logger 'RSYNC BACKUP : FAILED!'
```

If you are using rsnaphot backup server (see how to setup RHEL / CentOS [7] / Debian rsnapshot [8] backup server) add the following to your /etc/rsnapshot.conf file

```
# Get ssh login info via keychain
cmd_preexec     source /root/.keychain/hostname.example.com-sh
```

## Final Note About Keychain and Security

- Cracker with an advanced attacking with deadly coding skills can still get key from memory. However, keychain makes it pretty difficult for normal users and attackers to steal your keys and use it.
- OpenSSH sshd server offers two additional options to protect abuse of keys. First, make sure root login disabled (**PermitRootLogin yes**). Second, specify which user accounts on the server are allowed to be used for authentication by adding **AuthorizedKeysFile %h/.ssh/authorized_keys_FileName**. See sshd_config man page for further details.

**Suggested Readings:**

- man pages sshd, sshd_config, keychain
- rsnapshot MySQL backup [9] script.
- The OpenSSH project [10] official website.
- The Keychain project [11] official website.
- Introducing ssh-agent and keychain [12] by Daniel Robbins





URLs in this post:

[1] Image: **http://www.cyberciti.biz/faq/category/bash-shell/**
[2] Image: **http://www.cyberciti.biz/faq/category/networking/**
[3] RPMForge repo: **http://www.cyberciti.biz/faq/rhel-centos-fedora-media-mp3-players-installtion/**
[4] Linux / UNIX setup SSH with DSA: **http://www.cyberciti.biz/faq/ssh-password-less-login-with-dsa-publickey-authentication/**
[5] multiple SSH keys for password: **http://www.cyberciti.biz/tips/linux-multiple-ssh-key-based-authentication.html**
[6] Image: **http://www.cyberciti.biz/faq/ssh-passwordless-login-with-keychain-for-scripts/keychain/**
[7] RHEL / CentOS: **http://www.cyberciti.biz/faq/redhat-cetos-linux-remote-backup-snapshot-server/**
[8] Debian rsnapshot: **http://www.cyberciti.biz/faq/linux-rsnapshot-backup-howto/**
[9] rsnapshot MySQL backup: **http://bash.cyberciti.biz/backup/rsnapshot-remote-mysql-backup-shell-script/**
[10] OpenSSH project: **http://www.openssh.com/**
[11] Keychain project: **http://www.gentoo.org/proj/en/keychain/**
[12] ssh-agent and keychain: **http://www.ibm.com/developerworks/linux/library/l-keyc2/**