SSH Passwordless Login Using SSH Keygen in 5 Easy Steps

SSH (**Secure SHELL**) is an open-source and most trusted network protocol that is used to login to remote servers for the execution of commands and programs. It is also used to transfer files from one computer to another computer over the network using a secure copy (**SCP**) Protocol.
In this article, we will show you how to setup password-less login on **RHEL/CentOS** and **Fedora** using **ssh keys** to connect to remote **Linux** servers without entering a password. Using Password-less login with **SSH keys** will increase the trust between two **Linux** servers for easy file synchronization or transfer.

My Setup Environment

```
SSH Client : 192.168.0.12 ( Fedora 21 )

SSH Remote Host : 192.168.0.11 ( CentOS 7 )
```

If you are dealing with a number of **Linux** remote servers, then **SSH Password-less** login is one of the best ways to automate tasks such as automatic backups with scripts, synchronization files using SCP, and remote command execution.
In this example, we will set up **SSH password-less** automatic login from server **192.168.0.12** as user **tecmint** to **192.168.0.11** with user **sheena**.

Step 1: Create Authentication SSH-Keygen Keys on – (192.168.0.12)

First login into server **192.168.0.12** with user **tecmint** and generate a pair of public keys using the following command.

```
[tecmint@tecmint.com ~]$ ssh-keygen -t rsa

Generating public/private rsa key pair.
Enter file in which to save the key (/home/tecmint/.ssh/id_rsa): [Press enter key]
Created directory '/home/tecmint/.ssh'.
Enter passphrase (empty for no passphrase): [Press enter key]
Enter same passphrase again: [Press enter key]
Your identification has been saved in /home/tecmint/.ssh/id_rsa.
Your public key has been saved in /home/tecmint/.ssh/id_rsa.pub.
The key fingerprint is:
5f:ad:40:00:8a:d1:9b:99:b3:b0:f8:08:99:c3:ed:d3 tecmint@tecmint.com
The key's randomart image is:
+--[ RSA 2048]----+
|        ..oooE.++|
|         o. o.o  |
|          ..   . |
|         o  . . o|
|        S .  . + |
|       . .    . o|
|      . o o    ..|
|       + +       |
|        +.       |
+-----------------+
```

```
[tecmint@tecmint ~]$
```

Step 2: Create .ssh Directory on – 192.168.0.11

Use SSH from server **192.168.0.12** to connect server **192.168.0.11** using **sheena** as a user and create **.ssh** directory under it, using the following command.

```
[tecmint@tecmint ~]$ ssh sheena@192.168.0.11 mkdir -p .ssh

The authenticity of host '192.168.0.11 (192.168.0.11)' can't be established.
RSA key fingerprint is 45:0e:28:11:d6:81:62:16:04:3f:db:38:02:la:22:4e.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.11' (ECDSA) to the list of known hosts.
sheena@192.168.0.11's password: [Enter Your Password Here]
```

```
[tecmint@tecmint ~]$ 
```

Step 3: Upload Generated Public Keys to – 192.168.0.11

Use SSH from server **192.168.0.12** and upload a new generated public key (**id_rsa.pub**) on server **192.168.0.11** under **sheena**'s **.ssh** directory as a file name **authorized_keys**.

```
[tecmint@tecmint ~]$ cat .ssh/id_rsa.pub | ssh sheena@192.168.0.11 'cat >>
.ssh/authorized_keys'

sheena@192.168.1.2's password: [Enter Your Password Here]
```

```
[tecmint@tecmint ~]$ 
```

Step 4: Set Permissions on – 192.168.0.11

Due to different SSH versions on servers, we need to set permissions on .ssh directory and authorized_keys file.

```
[tecmint@tecmint ~]$ ssh sheena@192.168.0.11 "chmod 700 .ssh; chmod 640
.ssh/authorized_keys"

sheena@192.168.0.11's password: [Enter Your Password Here]
```

```
[tecmint@tecmint ~]$ 
```

Step 5: Login from 192.168.0.12 to 192.168.0.11 Server without Password

From now onwards you can log into **192.168.0.11** as **sheena** user from server **192.168.0.12** as **tecmint** user without a password.

```
[tecmint@tecmint ~]$ ssh sheena@192.168.0.11
```

```
[tecmint@tecmint ~]$
```