Mount remote directory by SSHFS in CentOS 7

SSHFS is a filesystem client to mount and interact with directories and files located on a remote server or workstation over a normal ssh connection.

Install SSHFS

To mount directory in remote server, you will need to install SSHFS

```
sudo yum -y install sshfs
```

Once installed, you can simply try to mount your remote server. Note that the remote server must support SSH.

```
sshfs ck@192.168.10.7:/data/disk1 /mnt/disk1
```

Auto-mount remote directory while server boot

If you want to setup auto-mounting remote directory, you need to create your own signature like below

```
ssh-keygen -t rsa -b 4096 -C "ck@ckii.com"
```

While you create your own key, you will need to put your location where to save your key. By default, it will be saved to **/YOUR_HOME_DIR/.ssh/id_rsa**

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ck/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ck/.ssh/id_rsa.
Your public key has been saved in /home/ck/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:EociK2DFhCX85+RuLAoj2pGdr/QXqbNzSiAzUwxOdV4 ck@ckii.com
The key's randomart image is:
+---[RSA 4096]----+
| .+*= o E        |
| oo=.o + .       |
|  +.o + . .      |
| . o. o+ .       |
|  = o=. S.       |
|   B +o o        |
|  o +..o .       |
|...o +B +        |
|o.ooo+=O         |
+----[SHA256]-----+
```

You may want pem file in some cases, and below commands can do that:

```
ssh-keygen -f id_rsa -m 'PEM' -e > id_rsa.pem
```

Follow-up action you need to do is copying ID by **ssh-copy-id <id@remote_server>**

```
ssh-copy-id ck@192.168.10.7
```

If everything went well, you will be able to connect to your server without typing pass-phrase like before.

In order to auto-mount remote directory, you will need to add following lines at /etc/fstab

```
user_id@server_address:<where_to_mount> <target_directory> fuse.sshfs defaults,idmap=user,allow_other,
delay_connect,reconnect,_netdev,users,IdentityFile=<your_key_location> 0 0
```

Below is an working example

```
kurapa@192.168.10.7:/data/disk1 /mnt/disk1 fuse.sshfs defaults,idmap=user,allow_other,delay_connect,reconnect,
_netdev,users,IdentityFile=/home/ck/.ssh/id_rsa 0 0
```

Note that I had a problem that the CentOS server can't auto-mount by sshfs in some cases, delay_connect solved my paint points, and it doesn't make any performance issues but stable auto-mounting in the latest multi-core servers.

In AWS EC2, you will need to use keypair to access your server, and following is an example to auto-mount your AWS EC2 instances:

```
sshfs#centos@10.0.1.6:/pub/ /mnt/live1 fuse    user,_netdev,reconnect,uid=1000,gid=1000,idmap=user,allow_other,
IdentityFile=your-keypair_root.pem  0   0
sshfs#centos@10.0.1.7:/pub/ /mnt/live2 fuse    user,_netdev,reconnect,uid=1000,gid=1000,idmap=user,allow_other,
IdentityFile=your-keypair_root.pem  0   0
```

Note that your-keypair_root.pem should be owned by **root** with **400** in permission, **sshfs#** need to be added to the account information, and fuse.sshfs needs to be changed to **fuse**.

Enhance access speed with no encryption

If you want to enhance its access speed even though its security is poor than before, you can stop encryption by adding **Ciphers=aes128-ctr, Compression=no** like below:

For on-prem servers:

```
kurapa@192.168.10.91:/ /data/synology fuse.sshfs   defaults,idmap=user,Ciphers=aes128-ctr,Compression=no,
allow_other,delay_connect,reconnect,_netdev,users,umask=0002,IdentityFile=/root/.ssh/id_rsa 0 0
```

For AWS servers

```
sshfs#centos@10.0.1.6:/pub/ /mnt/live1 fuse    user,_netdev,reconnect,uid=1000,gid=1000,idmap=user,
Ciphers=aes128-ctr,Compression=no,allow_other,IdentityFile=your-keypair_root.pem  0   0
```

If you want to have more performance, please try NFS.