Getting Connected

When your account is created you will be provided with details on how to connect to the Linux server. You may be provided with some or all of the following information:

- Username. This is also known as an account, login, or ID.
- Password
- SSH key
- Server name or IP address
- Port number
- Connection protocol

The connection protocol will either be SSH (Secure Shell) or telnet. SSH and telnet provide ways to connect to a server over the Internet or a local area network. In the vast majority of cases the connection protocol will be SSH. Telnet is practically obsolete at this point, however you may run into it if you need to access a legacy system.

Choosing an SSH Client

If you were given a specific SSH client to use, use that program and follow the documentation for that product. If you are free to choose your own client or were not provided one, I suggest using PuTTY for Windows or Terminal for Mac.

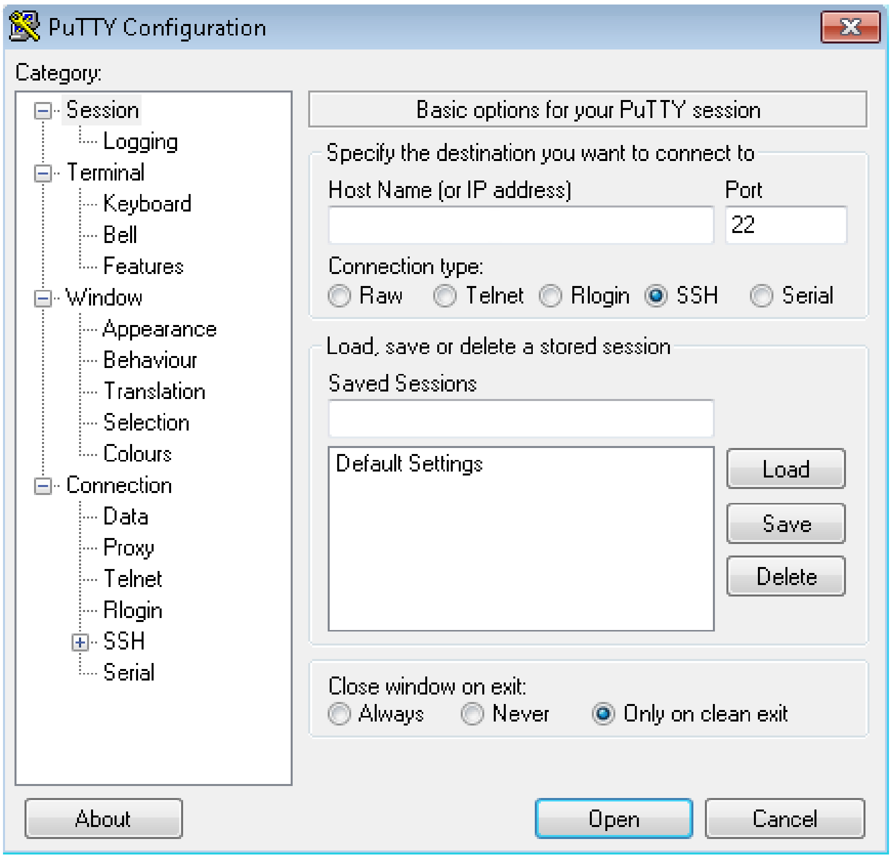
PuTTY can be downloaded from this website: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html. You only need putty.exe to get started. Here's the direct link to putty.exe: http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe.

The Terminal application comes pre-installed on Macs and is located in the `/Applications/Utilities` folder.

A list of other SSH clients is provided in the Deep Dive section at the end of this chapter.

Connecting via SSH with a Password from Windows

To connect to the Linux server using the SSH connection protocol, launch PuTTY.



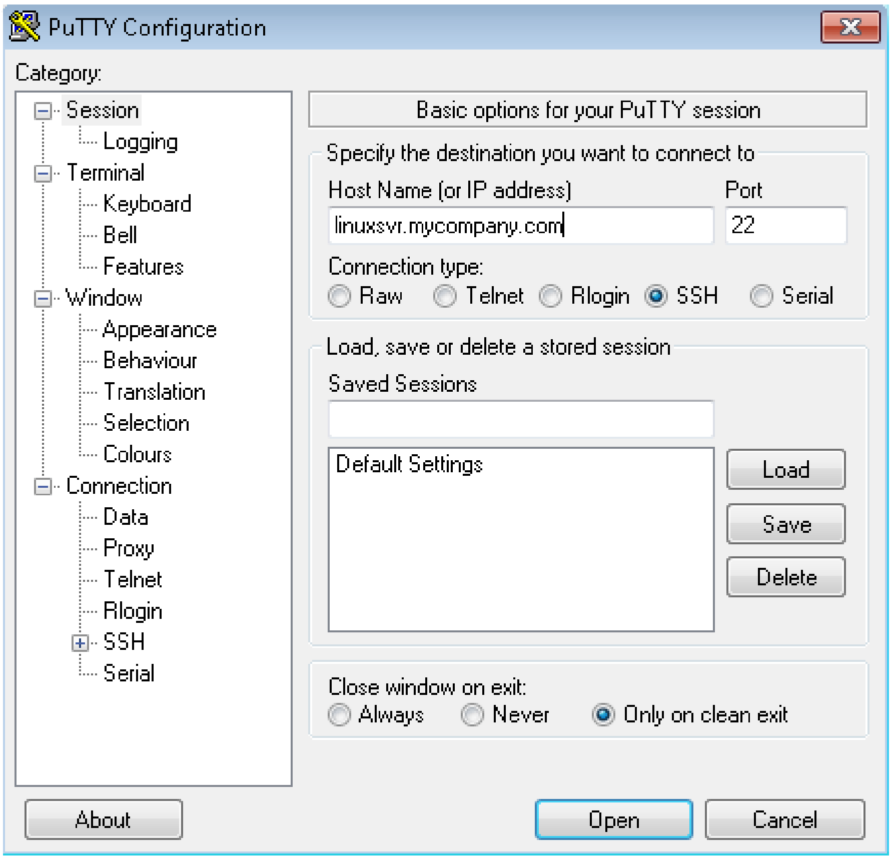
Host Name (or IP address):

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Close window on exit: ○ Always ○ Never ● Only on clean exit

Type the host name or IP address you were given into the `Host Name (or IP address)` box. If no port was given to you, leave it at the default value of 22.



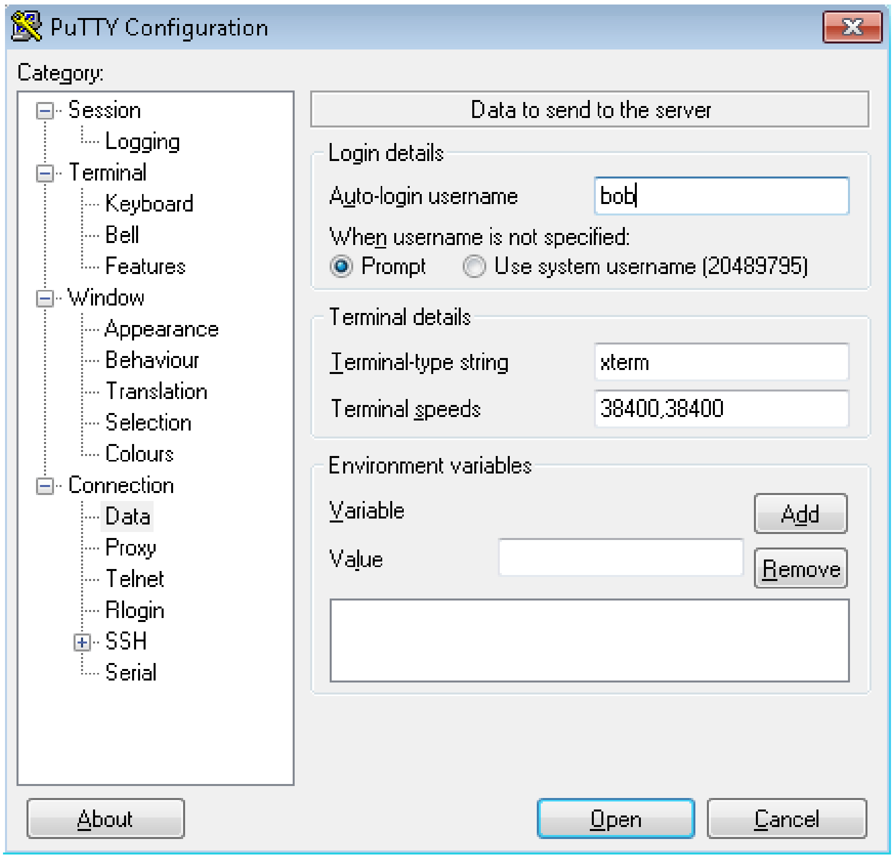
Host Name (or IP address): linuxsvr.mycompany.com

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Close window on exit: ○ Always ○ Never ● Only on clean exit

Enter your username by clicking on `Data` in the left pane. It is located directly below `Connection`. Type your username into the `Auto-login username` field. If you skip this step you will be prompted for your username when you connect to the server.



Data to send to the server

Login details

Auto-login username: bob

When username is not specified:
● Prompt ○ Use system username (20489795)

Terminal details

Terminal-type string: xterm

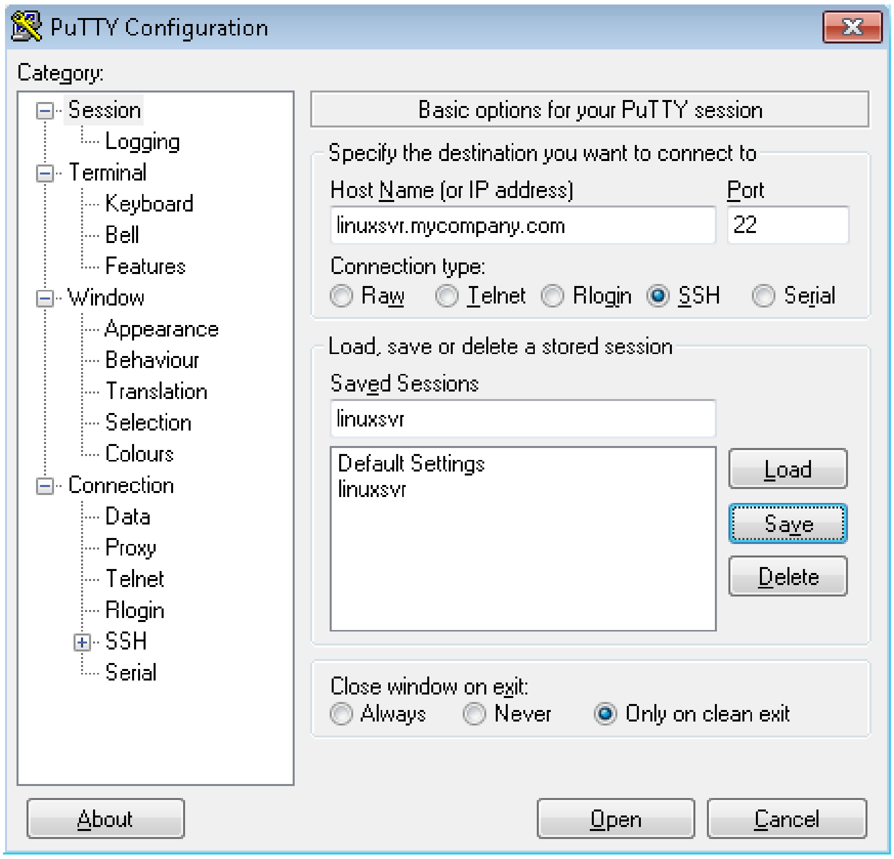
Terminal speeds: 38400,38400

Environment variables

Variable

Value

Save your session by typing in a name in the `Saved Sessions` box and clicking `Save`. This allows you to speed up this process by simply double clicking on your saved session to connect to the Linux server.



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): linuxsvr.mycompany.com

Port: 22

Connection type:
○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

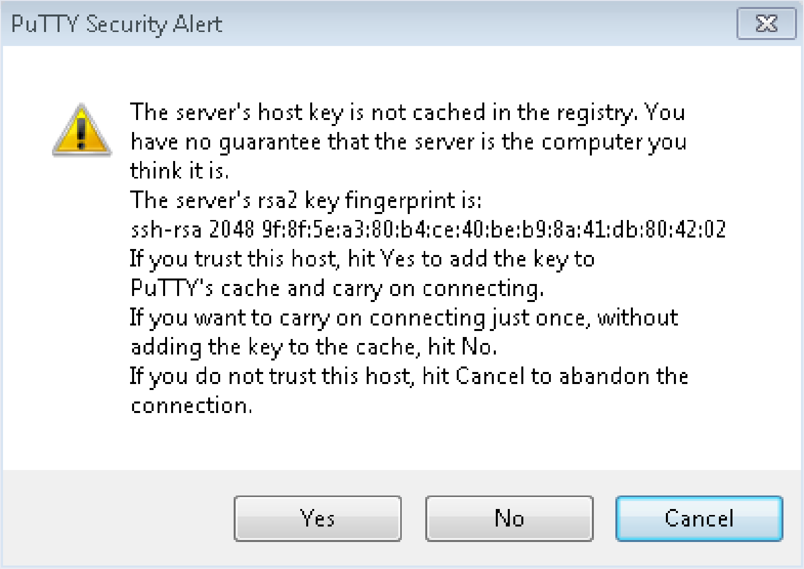
Load, save or delete a stored session

Saved Sessions: linuxsvr

Default Settings
linuxsvr

Close window on exit:
○ Always ○ Never ● Only on clean exit

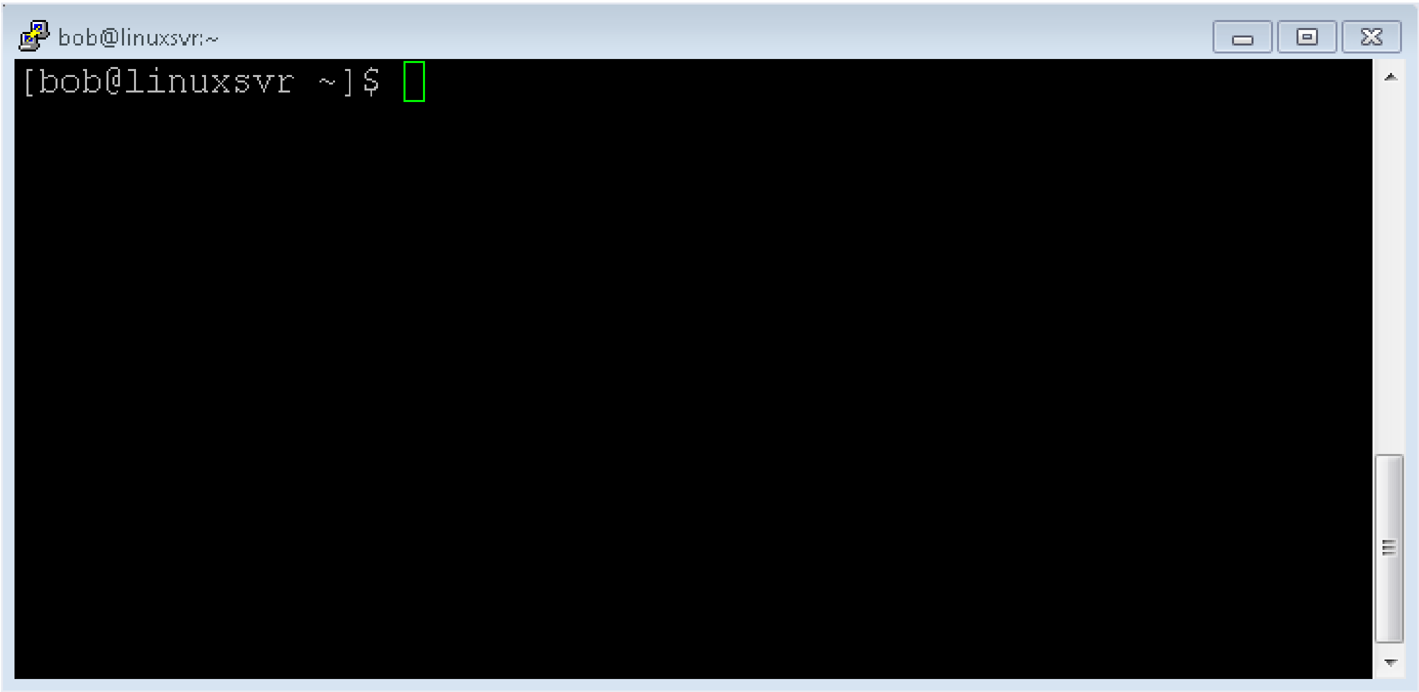
When you click `Open` a connection attempt will be made. The first time you connect to a particular server, PuTTY will ask to cache that server's host key. You will not be prompted again on subsequent connections. To add the server's SSH host key to PuTTY's cache, simply click `Yes` when prompted.



PuTTY Security Alert

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.
The server's rsa2 key fingerprint is:
ssh-rsa 2048 9f:8f:5e:a3:80:b4:ce:40:be:b9:8a:41:db:80:42:02
If you trust this host, hit Yes to add the key to PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without adding the key to the cache, hit No.
If you do not trust this host, hit Cancel to abandon the connection.

Once you are successfully logged in, you will see something similar to this:



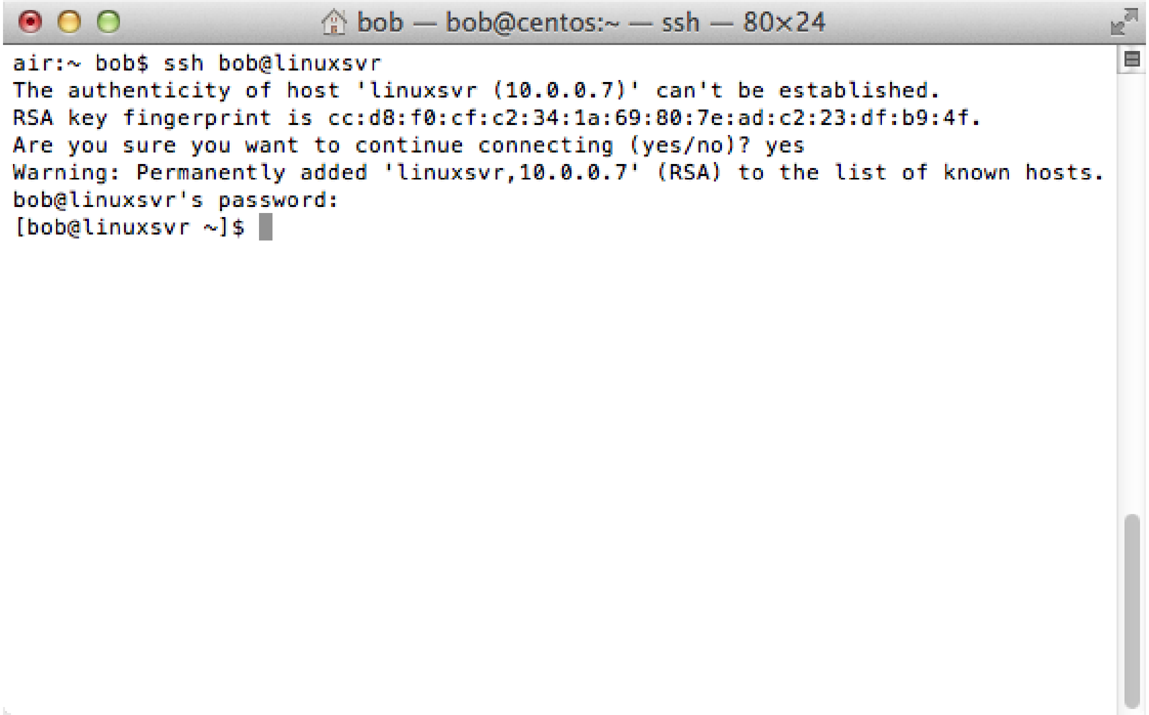
```
[bob@linuxsvr ~]$
```

Connecting via SSH with a Password from Mac

The built-in SSH client on Mac is a command line program. Command line programs can be run with the `Terminal` application that comes with the Mac operating system. It is located in the `/Applications/Utilities` folder. The format of the ssh command is `ssh -p port_number username@servername`. If you were not provided a port number, then the default port of 22 is assumed and you can omit `-p 22` from the ssh command. Similarly, the username only needs to be specified if it is different on the server than it is on your Mac workstation. For example, if your username on your Mac is `bob` and your username on `linuxsvr` is also `bob`, you can omit `bob@` and simply type `ssh linuxsvr`. Once Terminal is running, type in the ssh command. Commands are case-sensitive and the `ssh` command is lowercase. It should look like one of these three options:

```
ssh linuxsvr
ssh bob@linuxsvr
ssh -p 2222 bob@linuxsvr
```

The first time you connect to a particular server you will be asked to verify that server's host key. You will not be prompted again on subsequent connections. When you are asked `Are you sure` `you want to continue connecting (yes/no)?` type `yes` and press `Enter`. Once you have established a connection, you will be prompted for your password.



```
air:~ bob$ ssh bob@linuxsvr
The authenticity of host 'linuxsvr (10.0.0.7)' can't be established.
RSA key fingerprint is cc:d8:f0:cf:c2:34:1a:69:80:7e:ad:c2:23:df:b9:4f.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'linuxsvr,10.0.0.7' (RSA) to the list of known hosts.
bob@linuxsvr's password:
[bob@linuxsvr ~]$
```

Like Mac, Linux also comes with a terminal program and an SSH client. Once you are connected to one Linux server you can use the `ssh` command to connect to another Linux server. You can nest multiple connections and navigate through your network of Linux servers in this fashion.

General Information on Connecting via SSH with Keys

You may have not be given a password, but rather given an SSH key or even asked to generate one. In the physical world a key unlocks a door. Similarly, an SSH key is used to unlock the access to your account on a server. If you do not have a key, you cannot unlock the door.

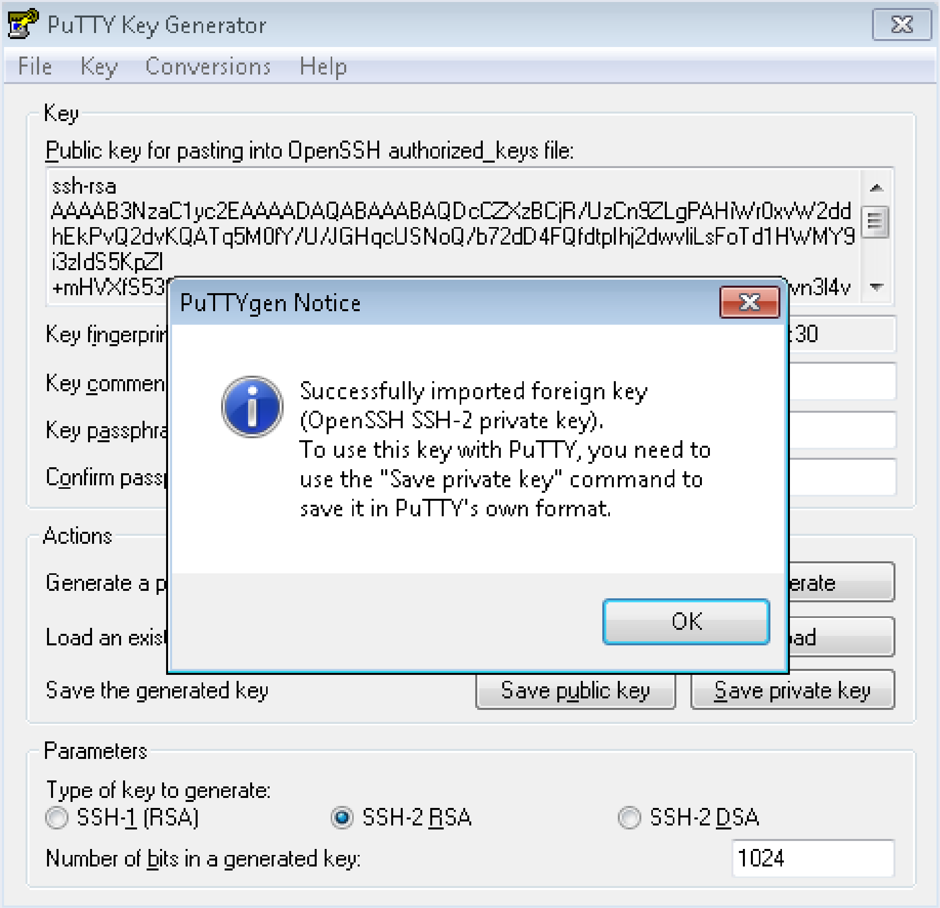
Using account passwords or a combination of account passwords and SSH keys is a common practice. With the growth of cloud computing in recent years, it is becoming more and more popular to use SSH keys exclusively. Since cloud servers are often connected to the public internet, they are prone to brute force attacks. A mischievous person could write a program that repeatedly connects to your server trying a new username and password combination each time. They can increase their odds of gaining entry by using a list of common usernames and passwords. Configuring your cloud server to not accept account passwords and to only accept SSH keys eliminates this threat.

You can further increase the security of your SSH key by giving it a passphrase. In this case it takes something you have -- the key -- and something you know -- the passphrase -- to gain access to your account. If you feel confident that your key will only be under your control, you can forgo providing a passphrase for your key. This will allow you to log into servers without typing a password at all. Having an SSH key without a passphrase can allow you to automate and schedule tasks that require logging in to remote systems.

Importing SSH Keys on Windows

If you were given an SSH key that is not already in the PuTTY format, you will need to convert it. PuTTYgen is required in order to convert an SSH key on a Windows system. Here is a direct download link to puttygen.exe: http://the.earth.li/~sgtatham/putty/latest/x86/puttygen.exe.

Run PuTTYgen, click `Load` and navigate to the private SSH key you were given. The names of the files are typically id_rsa or id_dsa for private keys, and id_rsa.pub or id_dsa.pub for public keys.



Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDcCZXzBCjR/UzCn9ZLgPAHiWr0xvW2dd
hEkPvQ2dvKQATq5M0fY/U/JGHqcUSNoQ/b72dD4FQfdtplhj2dwvliLsFoTd1HWMY9
i3zldS5KpZI
+mHVXfS53                                                  vn3l4v
```

Successfully imported foreign key
(OpenSSH SSH-2 private key).
To use this key with PuTTY, you need to
use the "Save private key" command to
save it in PuTTY's own format.

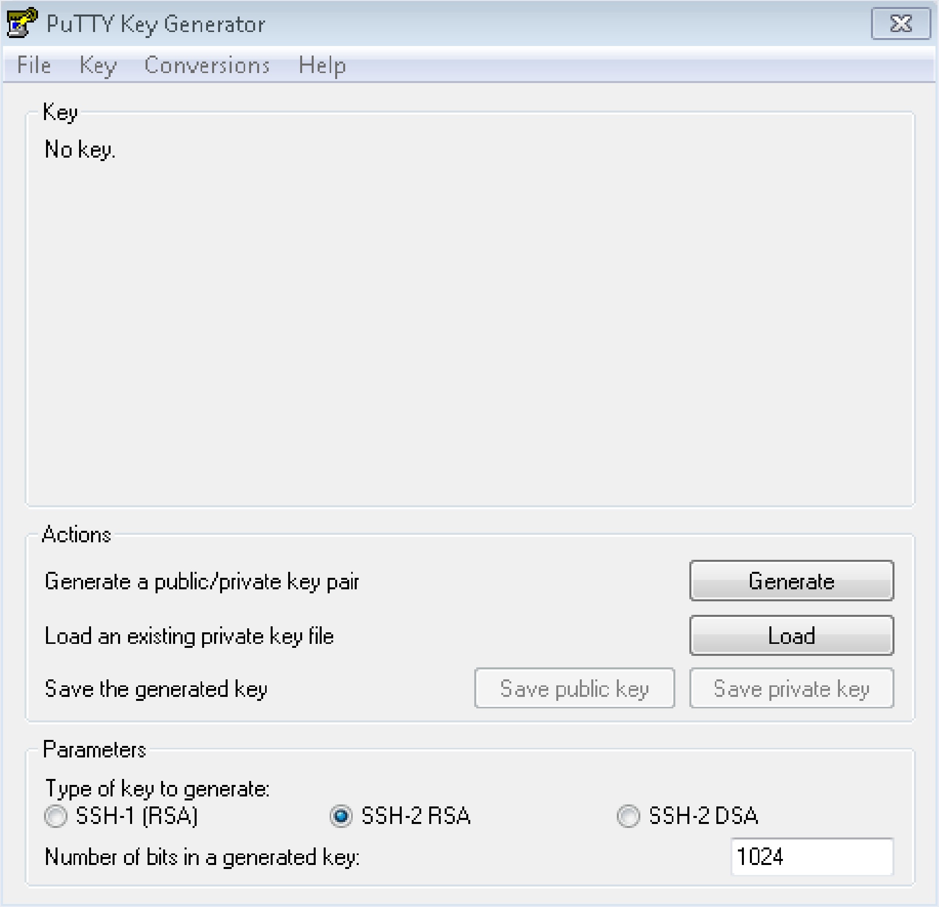
Type of key to generate:
○ SSH-1 (RSA) ● SSH-2 RSA ○ SSH-2 DSA

Number of bits in a generated key: 1024

Now you can save the public and private keys for later use with PuTTY.

Generating SSH Keys on Windows

In order to create an SSH key on a Windows system, you will need PuTTYgen. Here is a direct download link to puttygen.exe: http://the.earth.li/~sgtatham/putty/latest/x86/puttygen.exe.



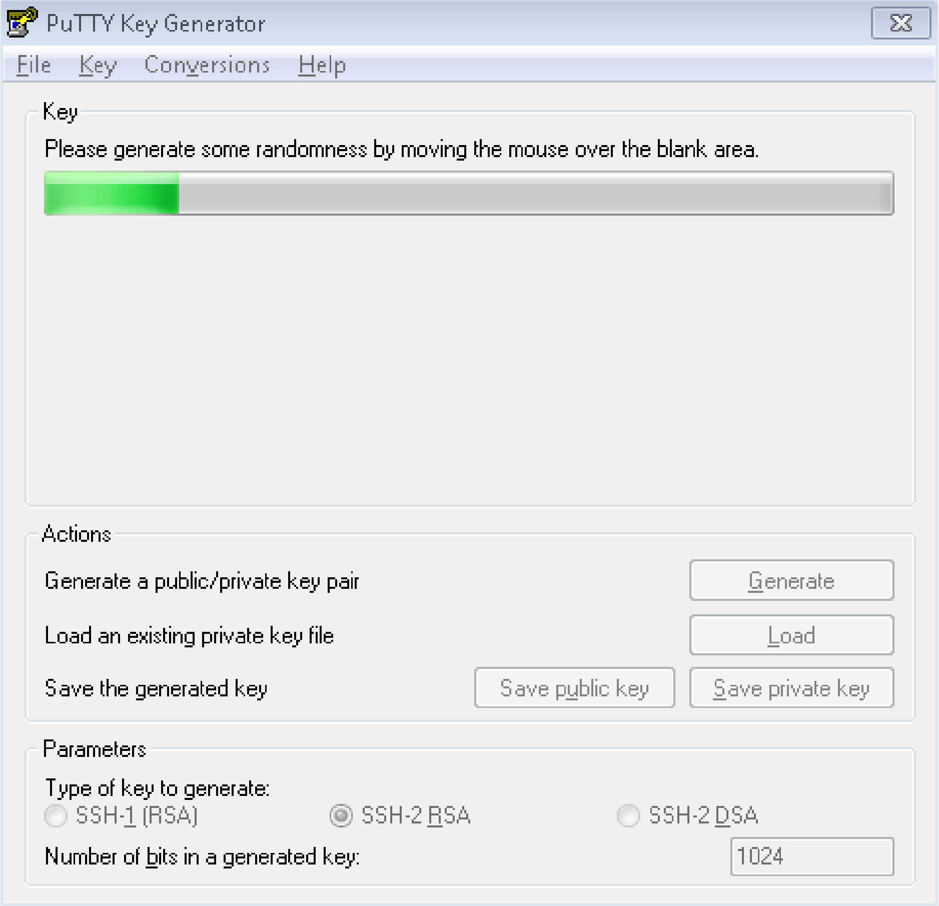
Key

No key.

Type of key to generate:
○ SSH-1 (RSA) ● SSH-2 RSA ○ SSH-2 DSA

Number of bits in a generated key: 1024

When you run PuTTYgen you will be asked to move the mouse around to create some random data that will be used in the generation of the key.



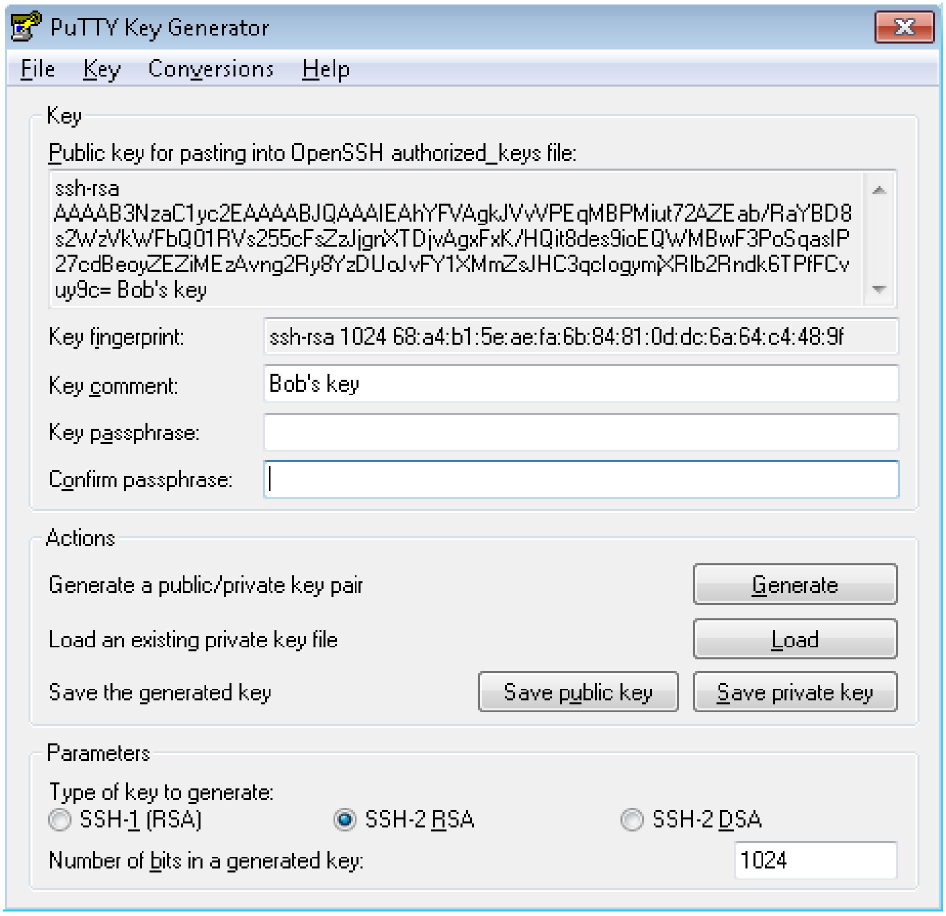
Key

Please generate some randomness by moving the mouse over the blank area.

Type of key to generate:
○ SSH-1 (RSA) ● SSH-2 RSA ○ SSH-2 DSA

Number of bits in a generated key: 1024

You have the option to use a passphrase for your key. You can also change the comment to something more meaningful like `Bob's key`.


PuTTY Key Generator
File Key Conversions Help
Key
Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABJQAAAIEAhYFVAgkJVvWPEqMBPMiut72AZEab/RaYBD8
s2WzVkWFbQ01RVs255cFsZzJjgnXTDjvAgxFxK/HQit8des9ioEQWMBwF3PoSqasIP
27cdBeoyZEZiMEzAvng2Ry8YzDUoJvFY1XMmZsJHC3qclogymjXRIb2Rndk6TPfFCv
uy9c= Bob's key
```

Key fingerprint: ssh-rsa 1024 68:a4:b1:5e:ae:fa:6b:84:81:0d:dc:6a:64:c4:48:9f

Key comment: Bob's key

Key passphrase:

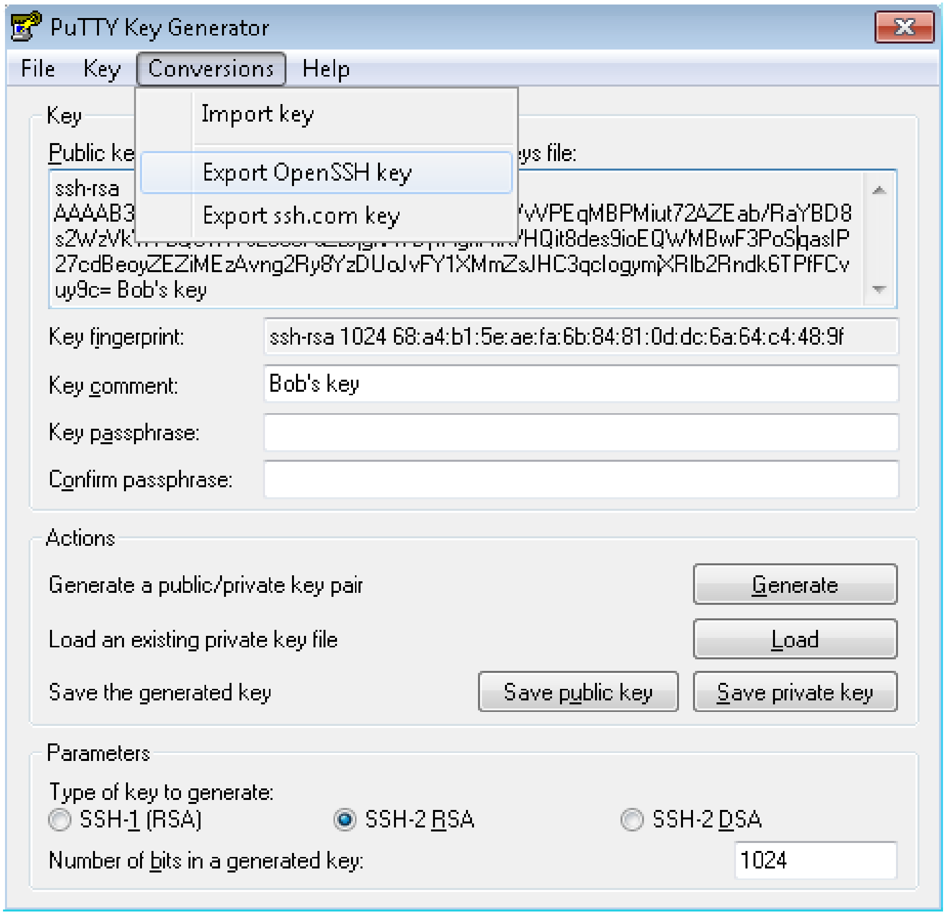
Confirm passphrase:

Type of key to generate:
○ SSH-1 (RSA) ● SSH-2 RSA ○ SSH-2 DSA

Number of bits in a generated key: 1024

Now, save the public and private keys buy pressing `Save public key` and then `Save private key`. Give the public key to the system administrator so they can associate it with your account. The private key is for your eyes only. Do not share your private key!

Next, export the key as an OpenSSH key by clicking on `Conversions` and then `Export OpenSSH Key`. This OpenSSH key can later be used on Unix or Linux systems.



```
ssh-rsa
AAAAB3                    VvWPEqMBPMiut72AZEab/RaYBD8
s2WzVk                    /HQit8des9ioEQWMBwF3PoSqasIP
27cdBeoyZEZiMEzAvng2Ry8YzDUoJvFY1XMmZsJHC3qclogymjXRIb2Rndk6TPfFCv
uy9c= Bob's key
```

Key fingerprint: ssh-rsa 1024 68:a4:b1:5e:ae:fa:6b:84:81:0d:dc:6a:64:c4:48:9f

Key comment: Bob's key

Key passphrase:

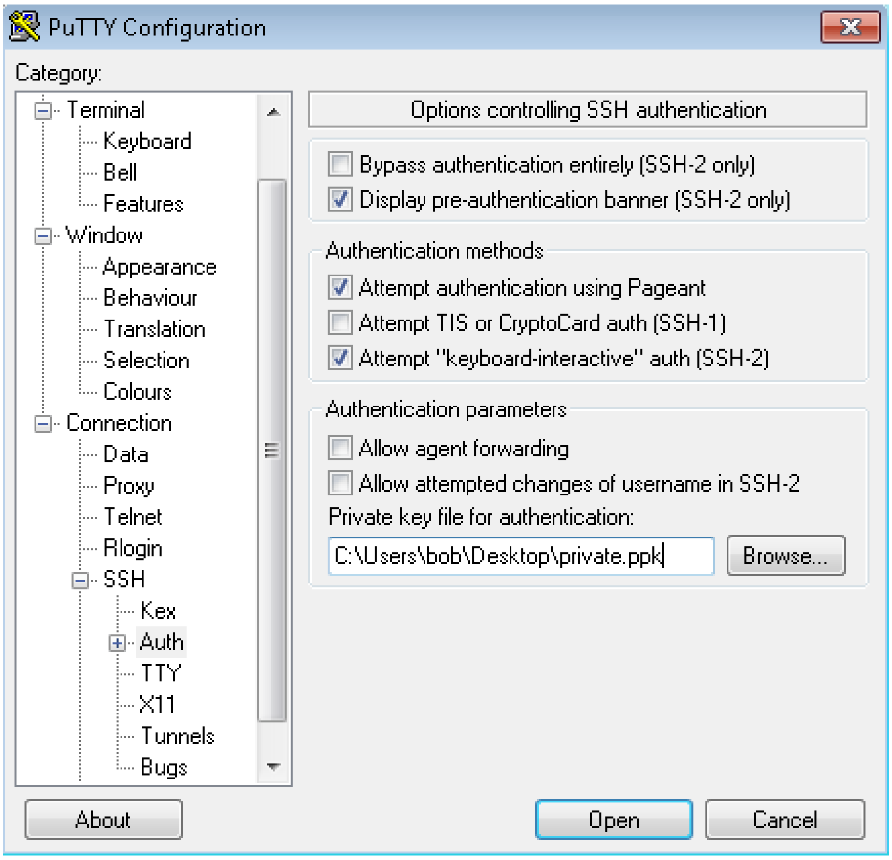
Confirm passphrase:

Type of key to generate:
○ SSH-1 (RSA) ● SSH-2 RSA ○ SSH-2 DSA

Number of bits in a generated key: 1024

Connecting via SSH from Windows

Follow the "Connecting via SSH with a Password from Windows" instructions, but this time add one additional step to specify your SSH private key file. You can do this by by clicking on the plug sign (`+`) next to `SSH` in the left pane to reveal more options. Next click on `Auth`. In the right pane select `Browse` under the `Private key file for authentication` field and locate your private SSH key.



Options controlling SSH authentication

□ Bypass authentication entirely (SSH-2 only)
☑ Display pre-authentication banner (SSH-2 only)

Authentication methods

☑ Attempt authentication using Pageant
□ Attempt TIS or CryptoCard auth (SSH-1)
☑ Attempt "keyboard-interactive" auth (SSH-2)

Authentication parameters

□ Allow agent forwarding
□ Allow attempted changes of username in SSH-2

Private key file for authentication:
C:\Users\bob\Desktop\private.ppk Browse...

Generating SSH Keys on Mac

If you are asked to generate an SSH key, launch the Terminal application and use the command line utility named `ssh-keygen`. You will be asked a series of questions. Accept all the defaults by pressing `Enter`. Optionally enter a passphrase for your SSH key.

```
mac:~ bob$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/bob/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/bob/.ssh/id_rsa.
Your public key has been saved in /Users/bob/.ssh/id_rsa.pub.
The key fingerprint is:
0b:14:c5:85:5f:55:77:35:5f:9e:15:a9:b4:b0:54:05 bob@mac
The key's randomart image is:
+--[ RSA 2048]----+
|      .o.o. .E+=@|
|       .o  o.. oO|
|      .  ...+ o.o|
|     .    .. o   |
|      . S        |
|       . .       |
|        .        |
|                 |
|                 |
+-----------------+
```

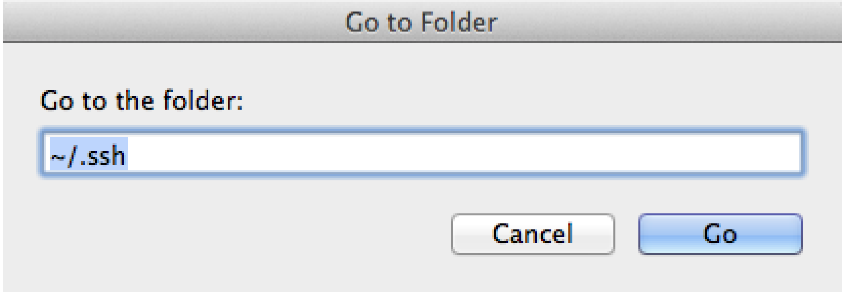
Connecting via SSH with Keys from Mac

If you generated your keys, this part is already done for you. If you were given an SSH key, you need to place it in a directory named `.ssh` underneath your home directory. Open the Terminal application and type in the following commands. Press the `Enter` key at the end of each line.

```
mkdir ~/.ssh
chmod 700 ~/.ssh
```

You will gain a full understanding of what these commands do as you progress through this book. In order to expedite the process of getting connected, the details will be saved for later.

Switch to the Finder to copy your keys into the `.ssh` folder. In the Finder menu click `Go` and then `Go to Folder...` and type `~/.ssh` when prompted. When you click `go`, the `.ssh` folder will be displayed. Now you can drag your keys into place.



Go to the folder: ~/.ssh

Back in the Terminal window, set the proper permissions on your key files. (Again, these commands will be covered later.)

```
cd ~/.ssh
chmod 600 *
```

I highly recommend naming the keys in the following format: id_rsa and id_rsa.pub or id_dsa and id_dsa.pub Otherwise, you will have to specify the location of your key when you use the ssh command or perform some additional configuration to tell SSH that your keys are not named in the standard way.

As a general rule it makes your life much easier if you follow the standard conventions and common practices. I will point them out along the way. One of the things I love most about Linux is the freedom and power it gives you to do things in a myriad of ways. There are cases were not following the standard conventions will be the right thing to do.

If you still wish to name your key something else, you can tell SSH where to find it by adding `-i key_location` to the `ssh` command. Remember, the format of the ssh command we used above is `ssh -p port_number username@servername`. It can be expanded to `ssh -i key_location -p port_number username@servername`. Here's an example:

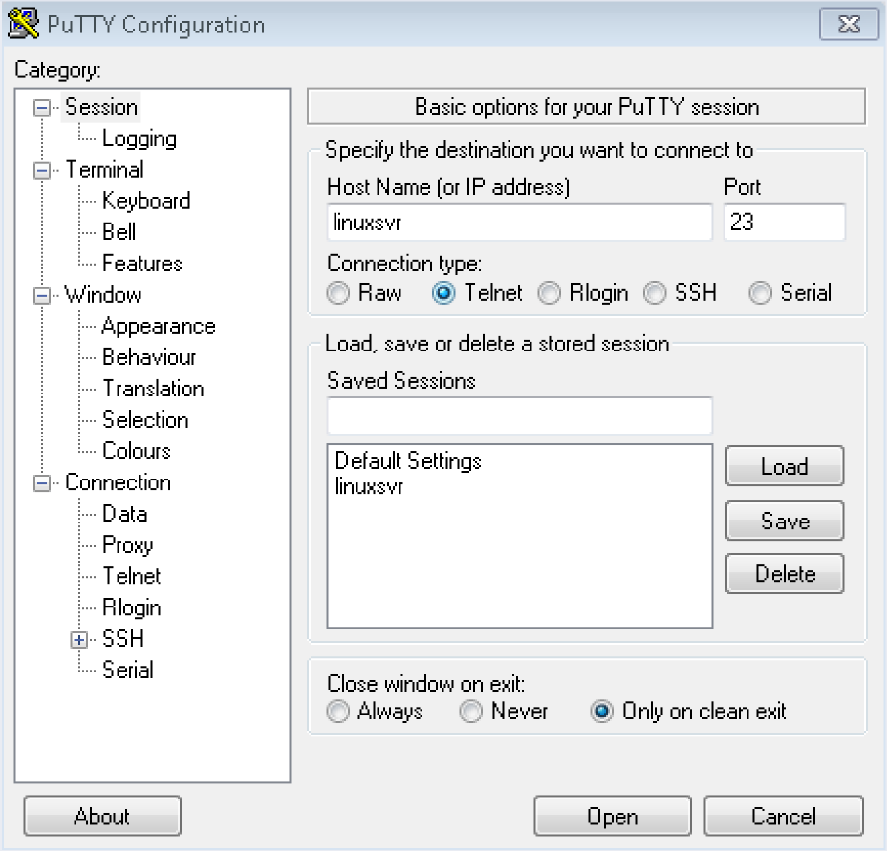
`ssh -i /Users/bob/.ssh/bobs_key bob@linuxsvr`

Connecting via Telnet

Telnet used to be the de facto way to connect to a Unix or Linux server. Over the years telnet has been replaced with Secure Shell, abbreviated SSH. SSH is, as its name implies, more secure than telnet. Telnet sends your login credentials over the network in plain text. SSH encrypts the communications between the client and the server, thus greatly improving security. If someone were to be packet snooping or eavesdropping on your connection, they would see garbled text and random characters. If you do have a need to telnet to a system you can use the SSH instructions from above, but with a couple of minor changes.

Connecting via Telnet from Windows

Run PuTTY and select the `Telnet` radio button. If no port was given to you, leave it at the default value of 23. You will be prompted for your username and password when you connect to the server.



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): linuxsvr

Port: 23

Connection type: ○ Raw ● Telnet ○ Rlogin ○ SSH ○ Serial

Load, save or delete a stored session

Saved Sessions

Default Settings
linuxsvr

Close window on exit: ○ Always ○ Never ● Only on clean exit

Connecting via Telnet from Mac

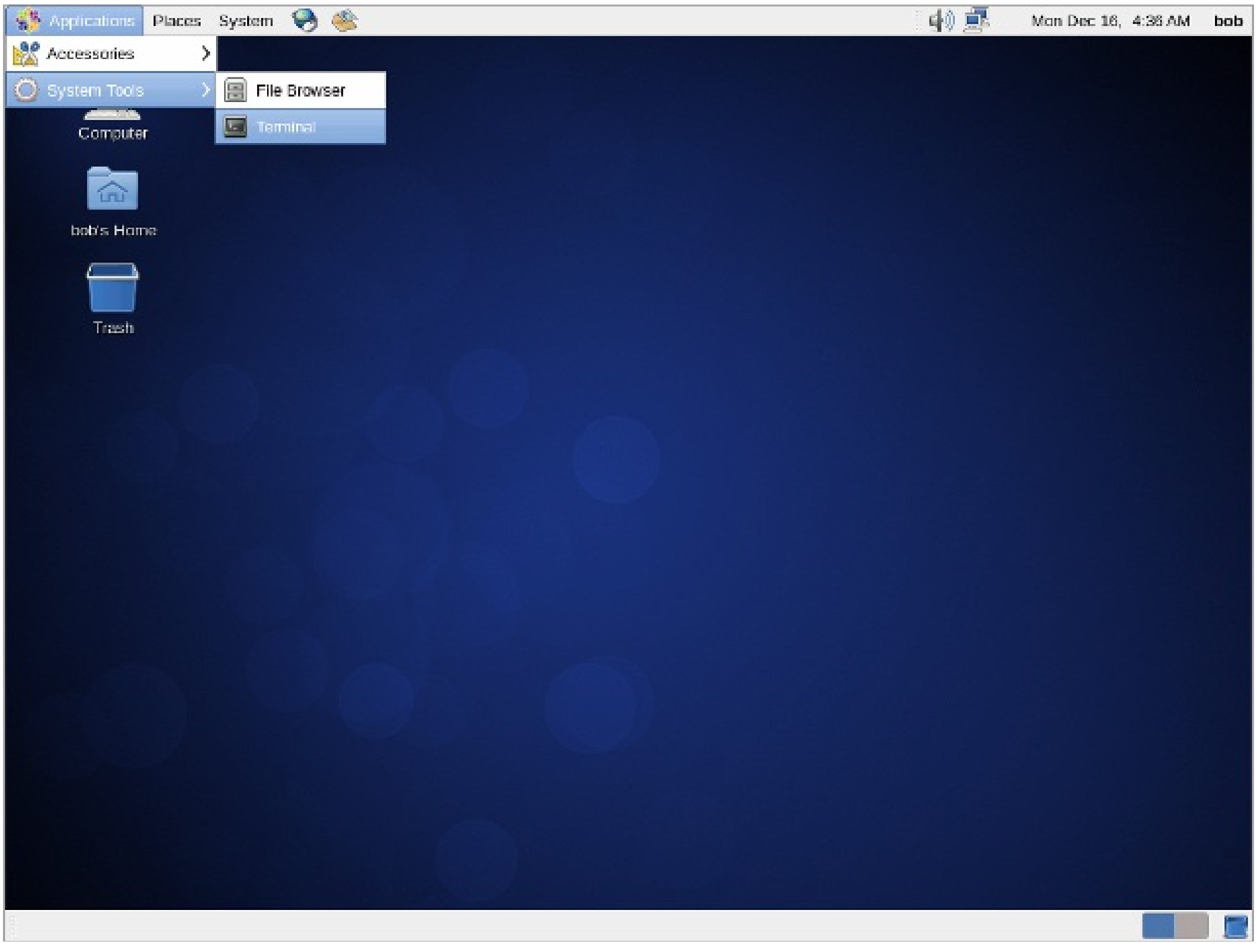
The built-in telnet client on Mac is a command line program. Command line programs can be run with the `Terminal` application that comes with the Mac operating system. It is located in the `/Applications/Utilities` folder. The format of the telnet command is `telnet servername port_number`. You only need to include a port number if it is different than the default value of 23. You will be prompted for your username and password when you connect to the server.

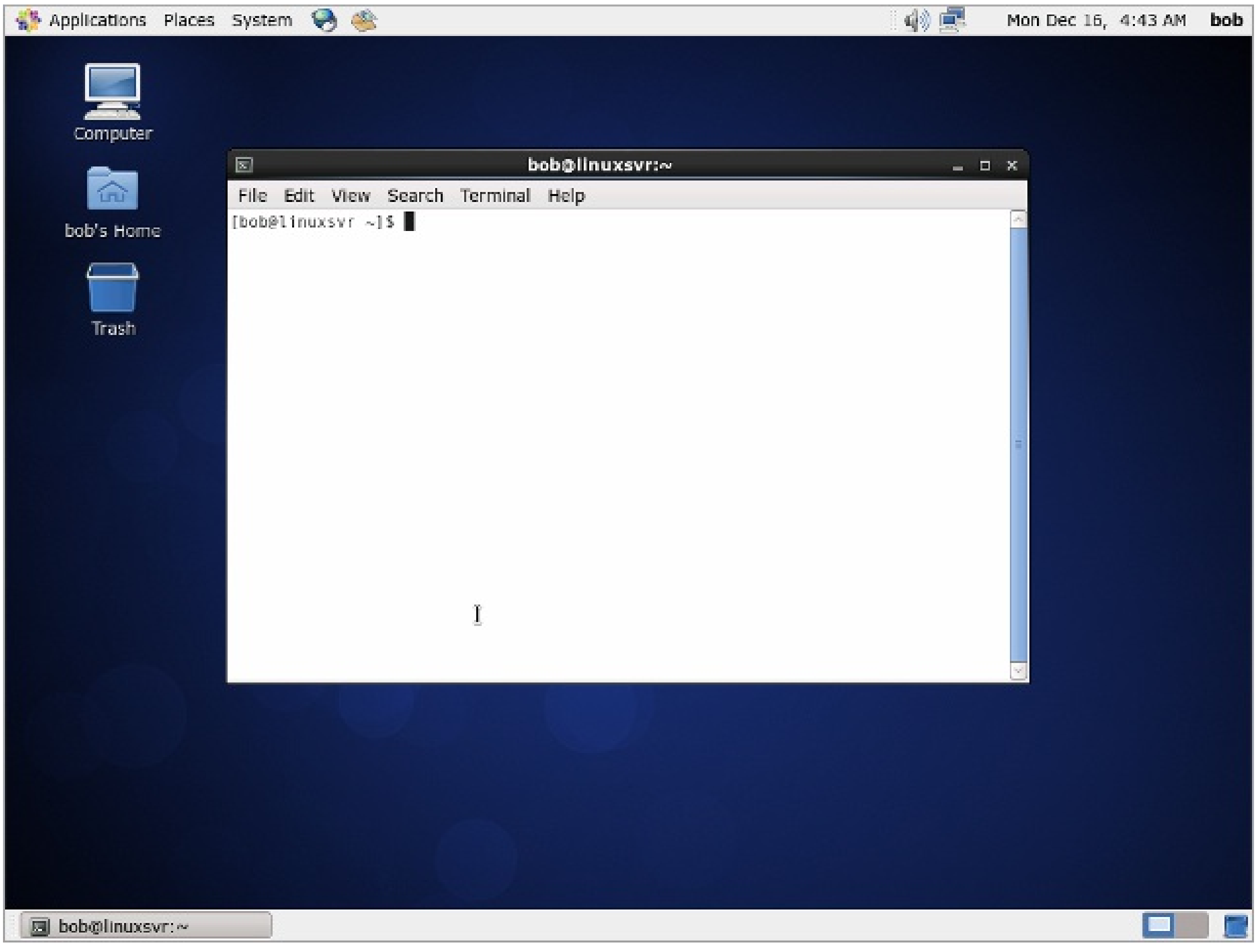
```
mac:~ bob$ telnet linuxsvr
Trying 10.0.0.7...
Connected to 10.0.0.7.
Escape character is '^]'.
Ubuntu 12.04.3 LTS
linuxsvr login: bob
Password:
Last login: Thu Nov  7 01:26:37 UTC 2013
Welcome to Ubuntu 12.04.3 LTS

 * Documentation:  https://help.ubuntu.com/

  System info as of Nov 7 01:26:52 UTC 2013

  System load:  0.42
  Usage of /:   3.1% of 40GB
  Memory usage: 32%
  Swap usage:   0%
  Processes:           89
  Users logged in:     0
  IP address for eth0: 10.0.0.7

bob@linuxsvr:~$
```

Connecting Directly

If you are running Linux in VirtualBox as described in the previous chapter or you have dedicated hardware with Linux installed on it, you can simply log in directly to the server. You will be presented with a prompt requesting your username and password. If it is a graphical environment, you will need to find a terminal application to use after you have logged in. In most cases it will literally be "terminal", but you might see some slight variations like "gnome terminal", "konsole", or "xterm."

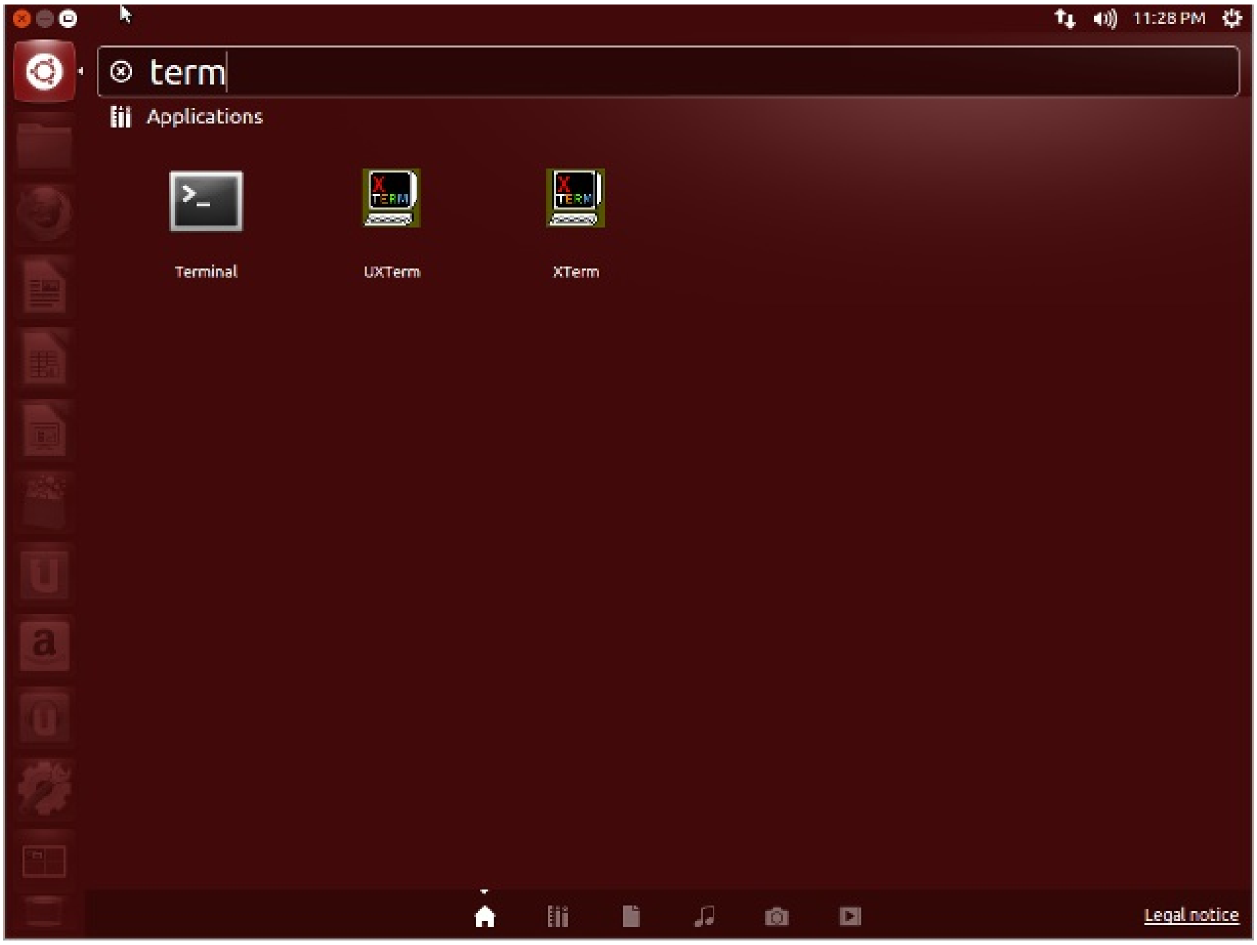
Here is what opening the terminal application looks like in CentOS. You will find it in one of the menus.

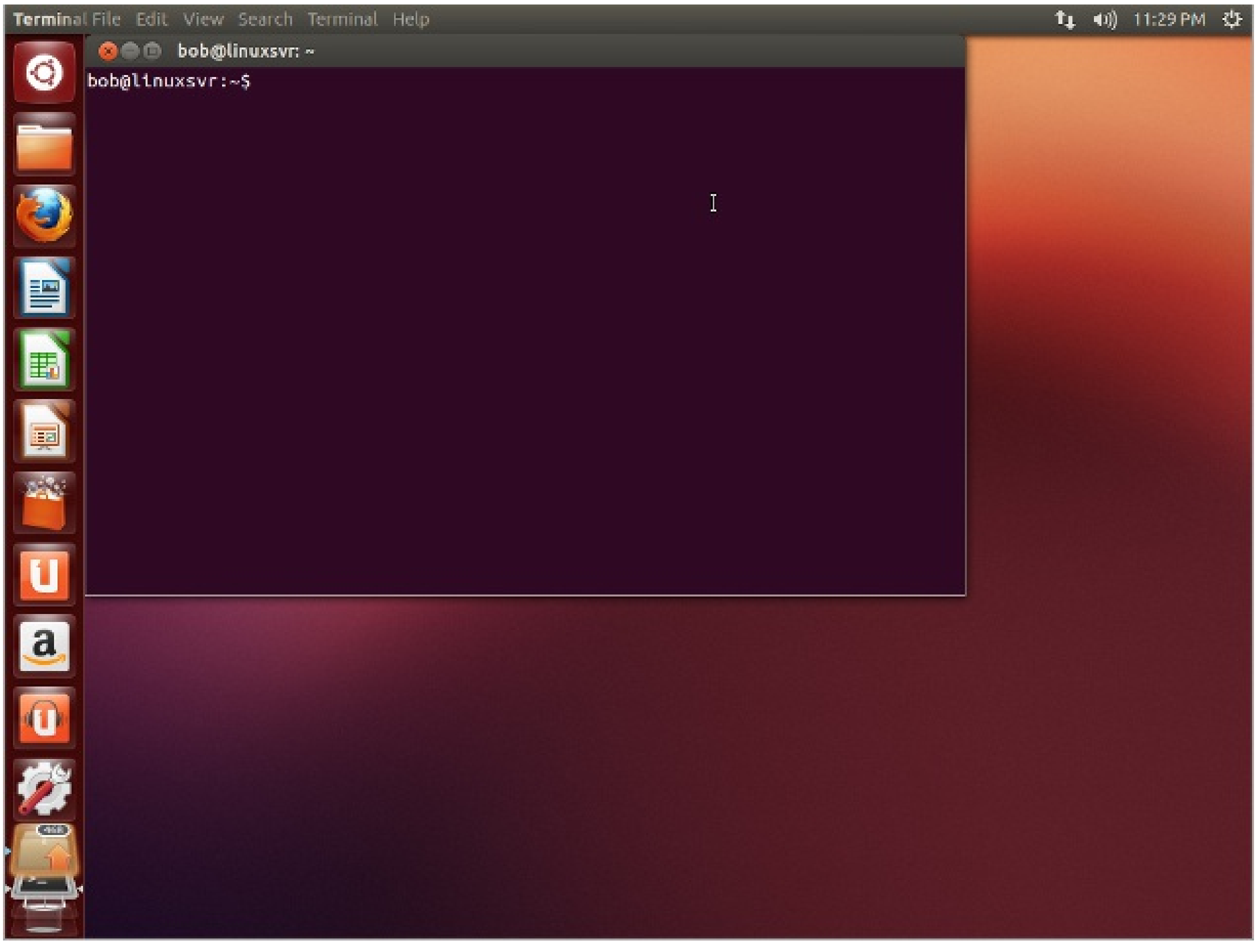




```
[bob@linuxsvr ~]$
```

In some Linux graphical environments there may not be a traditional menuing system. In these cases you will want to search for the terminal application. In this Ubuntu example, click the button in the top left of the screen to bring up the dashboard. You can now start typing to find applications that are installed on the system.





```
bob@linuxsvr:~$
```

Deep Dive

- 20 Windows SSH Clients You Can Use to Connect to Your Linux Server - An article that lists 20 of the most popular Windows SSH clients.
- List of Mac SSH clients
- List of SSH clients, all platforms
- List of Terminal Emulators - Includes terminals for Windows, Mac, and Linux.
- List of Telnet Clients
- List of Windows SSH clients
- OpenSSH.org - The official website for OpenSSH.
- PuTTY
    - PuTTY Documentation
    - putty.exe
    - puttygen.exe
    - putty.zip - A zip file containing all of the PuTTY program files.
- Watch Star Wars over a telnet connection.
    - `telnet towel.blinkenlights.nl`
    - To disconnect, hold down the `Ctrl` key and press the right bracket (`]`). At the `telnet >` prompt type `quit` and press `Enter`.
- Using SSH Public Key Authentication

Thanks for Reading

https://github.com/RatneshMallah/Linux_w0r1d