In order to create an SSH key, enter the ssh-keygen command with the encryption algorithm type specified as RSA as follows:

```
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/slynux/.ssh/id_rsa):
Created directory '/home/slynux/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/slynux/.ssh/id_rsa.
Your public key has been saved in /home/slynux/.ssh/id_rsa.pub.
The key fingerprint is:
f7:17:c6:4d:c9:ee:17:00:af:0f:b3:27:a6:9c:0a:05 slynux@slynux-laptop
The key's randomart image is:
+--[ RSA 2048]----+
|           .     |
|            o . .|
|     E       o o.|
|      ...oo |
|       .S .+  +o.|
|      .  . .=....|
|     .+.o...|
|      . . + o.  .|
|       ..+       |
+-----------------+
```

You need to enter a passphrase for generating the public-private key pair. It is also possible to generate the key pair without entering a passphrase, but it is insecure. We can write monitoring scripts that use automated login from the script to several machines. In such cases, you should leave the passphrase empty while running the `ssh-keygen` command to prevent the script from asking for a passphrase while running.

Now `~/.ssh/id_rsa.pub` and `~/.ssh/id_rsa` have been generated. `id_rsa.pub` is the generated public key and `id_rsa` is the private key. The public key has to be appended to the `~/.ssh/authorized_keys` file on remote servers where we need to auto-login from the current host.

In order to append a key file, use:

```
$ ssh USER@REMOTE_HOST "cat >> ~/.ssh/authorized_keys" < ~/.ssh/id_rsa.
pub
Password:
```

Provide the login password in the previous command.

The auto-login has been set up and from now onwards, SSH will not prompt for passwords during execution. Test this with the following command:

```
$ ssh USER@REMOTE_HOST uname
Linux
```

You will not be prompted for a password.

Most Linux distros ship with a tool called ssh-copy-id which will automatically add your private key to the authorized_keys file on the remote server. Use it like this:

```
ssh-copy-id USER@REMOTE_HOST
```

Port forwarding using SSH

Port forwarding is a technique by which you can enable other computers to connect to a particular service on a remote server using your machine. To understand this with an example, let's say your machine is assigned the IP 192.168.1.2 on a network and it has an Internet connection as well. Now, if you forward your machine's port 8000 to port 80 of `www.kernel.org`, it will be possible for some other computer to access the Linux Kernel website by going to `http://192.168.1.2:8000` using a browser. Let's see how to do this.

How to do it...

You can either forward a port on your local machine to another machine and it's also possible to forward a port on a remote machine to another machine. In the following methods, you will eventually get a shell prompt once the forwarding is complete. Keep this shell open to use the port forward and exit it whenever you want to stop the port forward.

1. Use this command to forward a port 8000 on your local machine to port 80 of `www.kernel.org`:

   ```
   ssh -L 8000:www.kernel.org:80 user@localhost
   ```

   Here, replace `user` with the username on your local machine.

2. Use this command to forward a port 8000 on a remote machine to port 80 of `www.kernel.org`:

   ```
   ssh -L 8000:www.kernel.org:80 user@REMOTE_MACHINE
   ```

   Here, replace `REMOTE_MACHINE` with the hostname or IP address of the remote machine and `user` with the username that you have SSH access to.