```
# Turn the interface down before setting new config
/sbin/ifconfig $IFACE down

if [ $UID -ne 0 ];
then
  echo "Run as root"
  exit 1;
fi

if [[ -n $HW_ADDR  ]];
then
  /sbin/ifconfig $IFACE hw ether $HW_ADDR
  echo Spoofed MAC ADDRESS to $HW_ADDR
fi

/sbin/iwconfig $IFACE essid $ESSID $KEY_PART freq $FREQ

/sbin/ifconfig $IFACE $IP_ADDR netmask $SUBNET_MASK

route add default gw $GW $IFACE

echo Successfully configured $IFACE
```

How it works...

The commands `ifconfig`, `iwconfig`, and `route` are to be run as root. Hence, a check for the root user is performed before performing any actions in the scripts.

Wireless LAN requires some parameters such as **essid**, **key**, and **frequency** to connect to the network. essid is the name of the wireless network to which we need to connect. Some networks use a WEP key for authentication, which is usually a 5 or 10 letter hex passphrase. Another parameter is the frequency assigned to the network which the `iwconfig` command uses to attach the wireless card with the proper wireless network.

We can scan and list the available wireless network using the utility `iwlist`. To scan, use the following command:

```
# iwlist scan
wlan0     Scan completed :
          Cell 01 - Address: 00:12:17:7B:1C:65
                    Channel:11
                    Frequency:2.462 GHz (Channel 11)
                    Quality=33/70  Signal level=-77 dBm
                    Encryption key:on
                    ESSID:"model-2"
```

The `Frequency` parameter can be extracted from the scan result, from the line `Frequency:2.462 GHz (Channel 11)`.

WEP is used in this example for simplicity, but its worthy to note that currently it is considered insecure. If you are administering the wireless network, make sure that you use a variant of **Wi-Fi Protected Access2** (**WPA2**) to be secure.

See also

- The *Comparisons and tests* recipe of *Chapter 1*, *Shell Something Out*, explains string comparisons

Password-less auto-login with SSH

SSH is widely used with automation scripting, as it makes it possible to remotely execute commands at remote hosts and read their outputs. Usually, SSH is authenticated by using a username and password, which are prompted during the execution of SSH commands. However, providing passwords in automated scripts is impractical, so we need to automate logins. SSH has an in-built feature by which SSH can auto-login using SSH keys. This recipe describes how to create SSH keys and facilitate auto-login.

Getting ready

SSH uses an encryption technique called **asymmetric keys** consisting of two keys: a public key and a private key for automatic authentication. We can create an authentication key pair using the `ssh-keygen` command. For automating the authentication, the public key must be placed at the server (by appending the public key to the `~/.ssh/authorized_keys` file) and its private key file of the pair should be present at the `~/.ssh` directory of the user at the client machine, which is the computer you are logging in from. Several configurations (for example, path and name of the `authorized_keys` file) regarding the SSH can be configured by altering the configuration file `/etc/ssh/sshd_config`.

How to do it...

There are two steps towards setup of automatic authentication with SSH. They are:

- Creating the SSH key on the machine, which requires a login to a remote machine
- Transferring the public key generated to the remote host and appending it to `~/.ssh/authorized_keys`