We have used . to specify the path as the current working directory. It is equivalent to:

```
$ chmod 777 "$(pwd)" –R.
```

Applying ownership recursively

We can apply the ownership recursively by using the `-R` flag with the `chown` command as follows:

```
$ chown user.group . -R
```

Running an executable as a different user (setuid)

Some executables need to be executed as a different user (other than the current user that initiates the execution of the file), effectively, whenever they are executed, by using the file path, such as `./executable_name`. A special permission attribute for files called the `setuid` permission enables effective execution as the file owner when any other user runs the program.

First, change the ownership to the user that needs to execute it and then log in as the user. Then, run the following command:

```
$ chmod +s executable_file


# chown root.root executable_file
# chmod +s executable_file
$ ./executable_file
```

Now it executes effectively as the root user every time.

`setuid` is restricted such that `setuid` won't work for scripts, but only for Linux ELF binaries. This is a fix for ensuring security.

Making files immutable

Files on extended type filesystems, which are common in Linux (for example, **ext2**, **ext3**, **ext4**, and so on) can be made immutable using a certain type of file attributes. When a file is made immutable, any user or super user cannot remove the file until the immutable attribute is removed from the file. We can easily find out the filesystem type of any mounted partition by looking at the `/etc/mtab` file. The first column of the file specifies the partition device path (for example, `/dev/sda5`) and the third column specifies the file system type (for example, ext3).

Making a file immutable is one of the methods for securing files from modification. An example would be to use it for the `/etc/resolv.conf` file which stores a list of DNS servers. A DNS server is used to convert domain names (such as packtpub.com) to IP addresses. Usually, the DNS server will be set to your ISP's DNS server. However, some people prefer to use a third-party server and they can modify `/etc/resolv.conf` to point to that DNS. However, the next time you get connected to your ISP, `/etc/resolv.conf` will revert back to point to ISP's DNS. To prevent this, we can make it immutable.

In this recipe we will see how to make files immutable and make then mutable when required.

Getting ready

The `chattr` command can be used to make files immutable. In addition to this, `chattr` has other useful options as well.

How to do it...

1. A file can be made immutable using the following command:

   ```
   # chattr +i file
   ```

2. The file is, therefore, made immutable. Now try the following command:

   ```
   rm file
   rm: cannot remove `file': Operation not permitted
   ```

3. In order to make it writable, remove the immutable attribute as follows:

   ```
   chattr -i file
   ```

Generating blank files in bulk

Sometimes we may need to generate test cases to test programs that operate on thousands of files. Let's discuss how to generate such files in this recipe.

Getting ready

`touch` is a command that can create blank files or modify the timestamp of files if they already exist. Let's take a look at how to use them.

How to do it...

1. A blank file with the name `filename` will be created using the following command:

   ```
   $ touch filename
   ```