How to do it...

In order to broadcast a message to all users and all logged in terminals, use:

```
$ cat message | wall
```

Or:

```
$ wall< message
Broadcast Message from slynux@slynux-laptop
        (/dev/pts/1) at 12:54 ...


This is a message
```

The message outline will show who sent the message (which user and which host). It is noteworthy that the message gets displayed to the current terminal only if the **write message** option is enabled. In most distros, **write message** is enabled by default. If the sender of the message is root, then the message gets displayed on the screen irrespective of whether the **write message** option is enabled or disabled by the user.

In order to enable write messages, use:

```
$ mesg y
```

In order to disable write messages, use:

```
$ mesg n
```

Let's write a script for sending messages specifically to a given user's terminal:

```
#/bin/bash
#Filename: message_user.sh
#Description: Script to send message to specified user logged
terminals.
USERNAME=$1

devices=`ls /dev/pts/* -l | awk '{ print $3,$10 }' | grep $USERNAME |
awk '{ print $2 }'`
for dev in $devices;
do
  cat /dev/stdin > $dev
done
```

Run the script as:

```
./message_user.sh USERNAME < message.txt
# Pass message through stdin and username as argument
```

The output will be as follows:

```
$ cat message.txt
A message to slynux. Happy Hacking!
# ./message_user.sh slynux  < message.txt
# Run message_user.sh as root since the message is to be send to specific
user.
```

Now, the `slynux` command's terminal will receive the message text.

How it works...

The `/dev/pts` directory will contain character devices corresponding to each of the logged in terminals on the system. We can find out who logged into which terminal by looking at the owner of the device files from the `ls -l` output. This information is extracted by using `awk`, and then `grep` is used to extract the lines corresponding to the specified user only. The username is accepted as the first argument for the script and stored as variable `USERNAME`. `$devices` contains a list of terminals for a given user, which is iterated using a `for` loop. `/dev/stdin` will contain standard input data passed to the current process, which is redirected to the corresponding terminal (TTY) devices.

Gathering system information

Collecting information about the current system from the command line is very important in logging system data. The different system information data includes hostname, kernel version, Linux distro name, CPU information, memory information, disk partition information, and so on. This recipe will show you different sources in a Linux system to gather information about the system.

How to do it...

1. In order to print the hostname of the current system, use:

   ```
   $ hostname
   ```

   Or:

   ```
   $ uname -n
   ```

2. Print long details about the Linux kernel version, hardware architecture, and more by using:

   ```
   $ uname -a
   ```

3. In order to print the kernel release, use:

   ```
   $ uname -r
   ```