Getting ready

Logfiles are very good for helping you deduce what is going wrong with a system. Hence, while writing critical applications, it is always a good practice to log the progress of an application with messages into a logfile. We will learn the command `logger` to log into log files with `syslogd`. Before getting to know how to write into logfiles, let's go through a list of important logfiles used in Linux:

Logfile	Description
`/var/log/boot.log`	Boot log information.
`/var/log/httpd`	Apache web server log.
`/var/log/messages`	Post boot kernel information.
`/var/log/auth.log`	User authentication log.
`/var/log/dmesg`	System boot up messages.
`/var/log/mail.log`	Mail server log.
`/var/log/Xorg.0.log`	X Server log.

How to do it...

Let's see how to use `logger` to create and manage log messages:

1. In order to log to the syslog file `/var/log/messages`, use:

   ```
   $ logger LOG_MESSAGE
   ```

   For example:

   ```
   $ logger This is a test log line


   $ tail -n 1 /var/log/messages
   Sep 29 07:47:44 slynux-laptop slynux: This is a test log line
   ```

   The logfile `/var/log/messages` is a general purpose logfile. When the `logger` command is used, it logs to `/var/log/messages` by default.

2. In order to log to the syslog with a specified tag, use:

   ```
   $ logger -t TAG This is a message


   $ tail -n 1 /var/log/messages
   Sep 29 07:48:42 slynux-laptop TAG: This is a message
   ```

syslog handles a number of logfiles in `/var/log`. However, while logger sends a message, it uses the tag string to determine in which logfile it needs to be logged. `syslogd` decides to which file the log should be made by using the `TAG` associated with the log. You can see the tag strings and associated logfiles from the configuration files located in the `/etc/rsyslog.d/` directory.

3. In order to log in to the system log with the last line from another logfile, use:

```
$ logger -f /var/log/source.log
```

See also

- The *Using head and tail for printing the last or first 10 lines* recipe of *Chapter 3*, *File In, File Out*, explains the head and tail commands

Monitoring user logins to find intruders

Logfiles can be used to gather details about the state of the system. Here is an interesting scripting problem statement:

We have a system connected to the Internet with SSH enabled. Many attackers are trying to log in to the system, and we need to design an intrusion detection system by writing a shell script. Intruders are defined as users who are trying to log in with multiple attempts for more than two minutes and whose attempts are all failing. Such users are to be detected, and a report should be generated with the following details:

- User account to which a login is attempted
- Number of attempts
- IP address of the attacker
- Host mapping for the IP address
- Time for which login attempts were performed

Getting ready

We can write a shell script that scans through the logfiles and gather the required information from them. For dealing with SSH login failures, it is useful to know that the user authentication session log is written to the logfile `/var/log/auth.log`. The script should scan the logfile to detect the failure login attempts and perform different checks on the log to infer the data. We can use the `host` command to find out the host mapping from the IP address.