```
then

  printf "%-8s %-14s %-9s %-8s %-6s %-6s %-6s %s\n" "Date" "IP
address" "Device" "Capacity" "Used" "Free" "Percent" "Status" >
$logfile
fi

IP_LIST="127.0.0.1 0.0.0.0"
#provide the list of remote machine IP addresses

(
for ip in $IP_LIST;
do
  #slynux is the username, change as necessary
  ssh slynux@$ip 'df -H' | grep ^/dev/ > /tmp/$$.df

  while read line;
  do
    cur_date=$(date +%D)
    printf "%-8s %-14s " $cur_date $ip
    echo $line | awk '{ printf("%-9s %-8s %-6s %-6s
%-8s",$1,$2,$3,$4,$5); }'

  pusg=$(echo $line | egrep -o "[0-9]+%")
  pusg=${pusg/\%/};
  if [ $pusg -lt 80 ];
  then
    echo SAFE
  else
    echo ALERT
  fi

  done< /tmp/$$.df
done

) >> $logfile
```

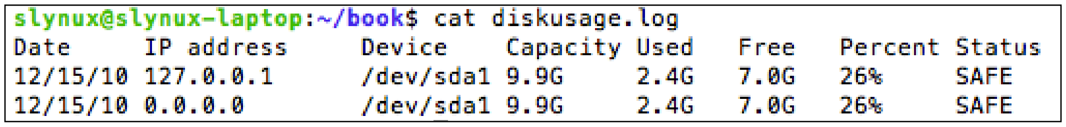
We can use the `cron` utility to run the script at regular intervals. For example, to run the script every day at 10 a.m., write the following entry in the `crontab`:

```
00 10 * * * /home/path/disklog.sh /home/user/diskusg.log
```

Run the command `crontab -e` and add the preceding line. You can run the script manually as follows:

```
$ ./disklog.sh
```

A sample output log for the previous script is as follows:



```
slynux@slynux-laptop:~/book$ cat diskusage.log
Date     IP address     Device    Capacity Used   Free   Percent Status
12/15/10 127.0.0.1      /dev/sda1 9.9G     2.4G   7.0G   26%     SAFE
12/15/10 0.0.0.0        /dev/sda1 9.9G     2.4G   7.0G   26%     SAFE
```

How it works...

In the `disklog.sh` script, we can provide the logfile path as a command-line argument or else it will use the default logfile. If the logfile does not exist, it will write the logfile header text into the new file. `-e $logfile` is used to check whether the file exists or not. The list of IP addresses of remote machines are stored in the variable `IP_LIST` delimited with spaces. It should be made sure that all the remote systems listed in the `IP_LIST` have a common user `test` with auto-login with SSH configured. A `for` loop is used to iterate through each of the IP addresses. A remote command `df -H` is executed to get the disk free usage data using the `ssh` command. It is stored in a temporary file. A `while` loop is used to read the file line by line. Data is extracted using `awk` and is printed. The date is also printed. The percentage usage is extracted using the `egrep` command and `%` is replaced with nothing to get the numeric value of percent. It is checked whether the percentage value exceeds 80. If it is less than 80, the status is set as `SAFE` and if greater than, or equal to 80, the status is set as `ALERT`. The entire printed data should be redirected to the logfile. Hence, the portion of code is enclosed in a subshell `()` and the standard output is redirected to the logfile.

See also

- The *Scheduling with cron* recipe in *Chapter 9*, *Administration Calls*, explains the `crontab` command

Finding out active user hours on a system

Consider a web server with shared hosting. Many users log in and log out to the server every day and the user activity gets logged in the server's system log. This recipe is a practice task to make use of the system logs and to find out how many hours each of the users have spent on the server and rank them according to the total usage hours. A report should be generated with the details, such as rank, user, first logged in date, last logged in date, number of times logged in, and total usage hours. Let's see how we can approach this problem.

Getting ready

The `last` command is used to list the details about the login sessions of the users in a system. The log data is stored in the `/var/log/wtmp` file. By individually adding the session hours for each user, we can find out the total usage hours.