

25 监视系统活动和查看进程。滴水不漏

更新时间：2019-07-17 16:37:36



学习从来无捷径，循序渐进登高峰。

更多一手资源+V：Andyqc1——高永祚
aa:3118617541

内容简介

1. 前言
2. w 命令：都有谁，在做什么？
3. ps 命令和 top 命令：列出运行的进程
4. 总结

1. 前言

经过上两课流和输出重定向，心之所向和输入重定向和管道，随意流转的锤炼，现在大家对 Linux 的命令行应该有了新的认识，而且水准大概已经提高到了一个不错的档次。

如果你还没有，快，快去给我练习去~

上两课算是比较难的，大家都辛苦了。所以这课让大家轻松一下，可以愉快地学完。

放眼现在的操作系统，基本都是多任务操作系统了，Linux 当然也不例外。因此，Linux 可以管理多个同时运行的程序。

通过之前的课程，我们知道，Linux 也是一个多用户的系统。多个用户可以同时在不同地方通过网络连接到同一个 Linux 系统上进行操作。

多用户多任务的特点有好处，但也有隐患。可能某个用户或者某个任务（其实就是运行着的程序）在某时让 Linux 系统过载了，就是有点太累了，任务太繁重了。

这时，我们可能想知道：

- 到底是哪个“小子”干了这等好事？
- 是哪个程序胆敢宕我的系统？
- 如何才能停止一个不再响应的程序？

在 Windows 系统下，我们可能常听说用超级组合键“Ctrl + Shift + Delete”来调出任务管理器，结束未响应的程序。

在 Linux 下，我们会用其他工具和别样的技术来处理僵局。这一章我们会学到不少新的 Linux 命令。

闲话不说，我们开始学习吧。

2. w 命令：都有谁，在做什么？

第一个出场的命令是迄今为止我们遇到过的最短的命令，这个命令只有一个字母：w。

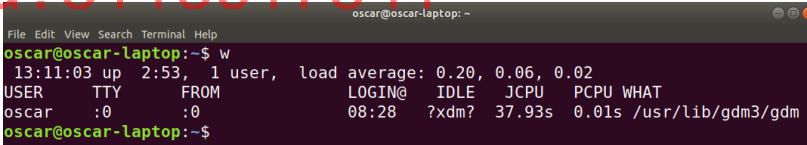
就是这么一个简单的命令，却挺实用，它可以帮助我们快速了解系统中目前有哪些用户登录着，以及他们在干什么。

如果你负责一台 Linux 服务器，有时候它会过载，变得很慢，这时你可以登录此服务器，然后运行 w 命令，快速了解到到底发生了什么事。

但对于大部分读者，因为是在自己的个人电脑上学习本课程的，Linux 系统一般只登录了一个用户，就是读者自己，所以用 w 命令之后，只显示个人用户的信息：

更多一手资源+V：Andyqcl
qq:3118617541

```
w
```



```
oscar@oscar-laptop: ~$ w
13:11:03 up 2:53, 1 user, load average: 0.20, 0.06, 0.02
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
oscar     :0        :0            08:28    ?xdm?  37.93s  0.01s  /usr/lib/gdm3/gdm
oscar@oscar-laptop: ~$
```

如上图所见，目前 Linux 系统中只有一个用户登录，就是我自己的用户，名叫 oscar。

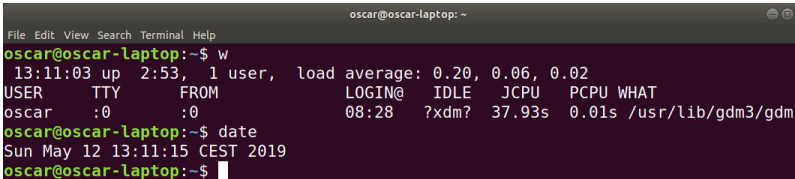
w 命令输出的信息虽然短，但是比较密集，看第一眼并不容易了解到到底是什么。但其实 w 命令给出的信息非常有用。

我们将其分解成不同的部分来解释，按照从上到下，从左到右的顺序。

时间（用 date 命令也可以做到）

我们看到信息的第一行中有 13:11:03，这就是当前时间：13点11分03秒。

我们之前学过 date 这个命令，它可以显示当前日期、时间和时区：



```
oscar@oscar-laptop: ~$ w
13:11:03 up 2:53, 1 user, load average: 0.20, 0.06, 0.02
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
oscar     :0        :0            08:28    ?xdm?  37.93s  0.01s  /usr/lib/gdm3/gdm
oscar@oscar-laptop: ~$ date
Sun May 12 13:11:15 CEST 2019
oscar@oscar-laptop: ~$
```

可以看到 date 命令显示当前时间：Sun May 12 13:11:15 CEST 2019

- Sun 是 Sunday 的缩写，表示“星期日”。
- May 表示“五月”。
- 12 是日期，表示 12 日。所以是 5 月 12 日。
- 13:11:15 是当前时间。13 点 11 分 15 秒。
- CEST 是欧洲中部夏令时间（Central European Summer Time，简称 CEST），比世界标准时间（UTC）早两个小时的时区，因为我写这篇文章时在法国巴黎。
- 2019 是年份，就是 2019 年。

date 命令的输出中的 13:11:15 就是 w 命令的输出中的时间那部分。

运行时间（用 uptime 命令也可以做到）

w 命令的输出中，紧跟在当前时间后面的是 up 2:53。

up 是英语“运行正常的”的意思。

所以 up 2:53 表示系统已经运行了 2 小时 53 分钟了，就是从开机登录到现在经过的时间。

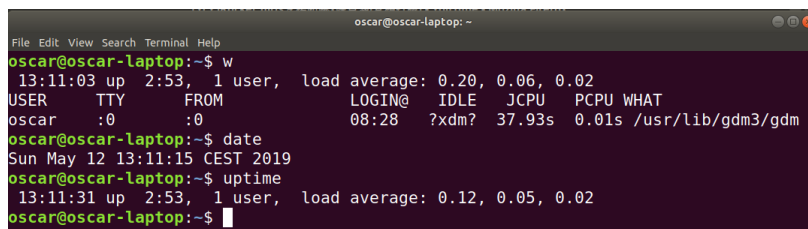
这个信息还是挺有用的，因为一旦系统重启或关机，那么运行时间会重归 0。

在 Windows 中，很多程序一旦安装完成，经常会要求重启系统来完成安装。

但 Linux 中不需要如此，一般只有系统的内核升级才需要重启系统。这也是 Linux 被大量用作服务器的原因，因为一般网站的服务器最好能够 7 天 24 小时不重启，一直运行。

系统正常运行时间一般就称为 uptime，Linux 中有一个 uptime 命令可以显示：

uptime



```
File Edit View Search Terminal Help
oscar@oscar-laptop: ~
oscar@oscar-laptop:~$ w
13:11:03 up 2:53, 1 user, load average: 0.20, 0.06, 0.02
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT
oscar :0 :0 08:28 ?xdm? 37.93s 0.01s /usr/lib/gdm3/gdm
oscar@oscar-laptop:~$ date
Sun May 12 13:11:15 CEST 2019
oscar@oscar-laptop:~$ uptime
13:11:31 up 2:53, 1 user, load average: 0.12, 0.05, 0.02
oscar@oscar-laptop:~$
```

可以看到，uptime 的输出其实就是 w 命令的第一行的内容，一模一样。

很多大型网站的服务器，如果你用 uptime 命令来获知其正常运行时间，可能会出现令你惊讶的好几百天的结果。

Linux 的稳定性和抗病毒性包您满意。

负载（同样可以用 uptime 命令获知）

在 w 命令的输出的第一行的右边，有三个数值，表示负载：

load average: 0.20, 0.06, 0.02

load 是“负载，负荷”的意思，average 是“平均值”的意思。负载是系统活动的一个指标：

这三个数值从左到右分别表示：

- 1 分钟以内的平均负载（0.20）
- 5 分钟之内的平均负载（0.06）
- 15 分钟之内的平均负载（0.02）

这些数值具体表示什么呢？

说起来稍微有点复杂。阅读手册，我们可以知道：这些数值表示一段时间内的平均活跃进程数（也就是使用 CPU 处理器的进程数，进程简单地说就是运行起来的程序）。

由此可知，近 1 分钟内平均有 0.20 个进程使用了处理器，也就是说处理器有 20% 的时间是活跃的。

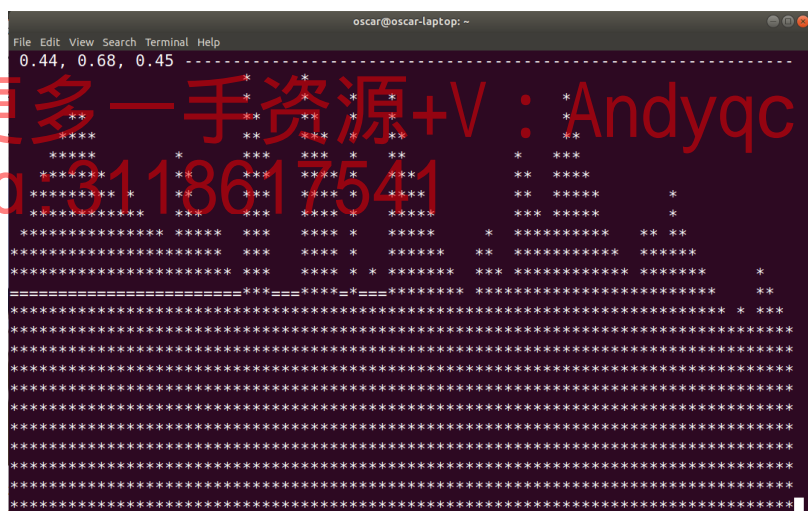
负载的数值也取决于电脑的处理器的核心数目。一个单核的处理器如果负载超过了 1，那就是过载了；双核的处理器如果负载超过了 2，那就是过载了；四核的处理器如果负载超过了 4，那就是过载了，依次类推。

有的服务器过载的时候负载的数值甚至能达到 50 那么多。

如果负载值在较长的时间中都维持比较大的话，就说明系统出问题了。我们可以用 `tload` 命令来绘制随时间变化的负载“曲线”图。

```
tload
```

`tload` 命令的输出（随时间变化）：



图中横坐标是时间，纵坐标的高低表示负载大小。

你可以用 `Ctrl + C` 来结束 `tload` 的运行。

登录的用户列表（用 `who` 命令也可以做到）

`w` 命令的输出中，除去第一行，下面的行可以归为一个部分，就是登录的用户列表。

这几行信息显示当下系统中连接的用户是哪几位，他们在做什么任务，任务进行多长时间了。

USER	TTY	FROM	LOGIN@	IDLE	JCPU	PCPU	WHAT
oscar	:0	:0	08:28	?xdm?	37.93s	0.01s	/usr/lib/gdm3/gdm


可以看到，有两行信息。大写英文的第一行是标题，第二行开始才是实际的信息。我目前是在自己的个人电脑上，所以 Linux 系统中只有我一个用户登录。

我们先来了解下各列的意义：

- **USER:** 用户名（登录名），user 是英语“用户”的意思。
- **TTY:** 此处的信息是“:0”，和旧版的 Ubuntu 上信息不一样，意思应该是指本地。旧版的 Ubuntu（17.10 之前的 Ubuntu 版本）中会显示终端信息，类似 ttyX 之类。Linux 中默认提供六个命令行终端和一个图形终端：tty1 ~ tty7。新版的 Ubuntu 中有所改变，我们之前的课程也演示过了。目前在 Ubuntu 中，tty2 ~ tty6 这 5 个是命令行终端（就是全屏、黑底白字的控制台），tty1 是图形终端（就是平时我们启动 Ubuntu 桌面版时默认登录的图形用户界面，也是全屏的）。Ubuntu 中可以通过 Ctrl + Alt + F1~F6 切换这 6 个终端。除了这 6 个基本的“大环境”终端，我们还可以在 tty1 中开很多不是全屏的终端，也就是我们平时用来输入命令行的图形终端（在 Ubuntu 中默认可以用 Ctrl + Alt + T 快捷键来启动）。这些终端的名字是以 pts 开头的，pts 是 pseudo terminal slave 的缩写，表示“伪终端从属”。如果我新开一个图形终端，那么显示名称为 pts/0。如果我再开一个图形终端，那么它的名字就是 pts/1。依次类推。
- **FROM:** 用户连接到的服务器的 IP 地址（或者主机名）。因为我们并没有登录远程服务器，只是在本地自己的电脑上测试，所以 FROM 那列显示的并不是实际的 IP 地址，而只是显示“:0”。from 是英语“从...”的意思。
- **LOGIN@:** 用户连接系统的时间，login 是英语“登录”的意思。
- **IDLE:** 用户有多久没活跃了（没运行任何命令）。idle 是英语“不活跃的，空闲的”的意思。
- **WHAT:** 当下用户正运行的程序，what 是英语“什么”的意思。此处是 `/usr/lib/gdm3/gdm-x-session`。gdm 是 GNOME Display Manager 的缩写，就是“GNOME 显示管理器”的意思。因为我们目前的 Ubuntu 18.04 默认使用 GNOME 桌面系统。session 是英语“会话”的意思。

我们如果单独运行 who（who 是英语“谁”的意思）命令，会输出当前哪些用户正登录着：

```
who
```



```
oscar@oscar-laptop:~$ who
oscar      pts/0        2019-05-12 18:32  (pts/0)
oscar@oscar-laptop:~$
```

w 命令虽然很有用，但是给出的信息还是不够详尽。我们下面来看看如何获得详细的系统进程的信息。

3. ps 命令和 top 命令：列出运行的进程

简单说来，进程就是加载到内存中运行的程序。

大多数程序运行时都只在内存中启动一个进程，例如 Linux 中的 OpenOffice 这个软件。有的程序运行时却会创建不少进程，例如 Google 的 Chrome 浏览器，每开一个标签栏都会创建一个新的进程。

在网络服务器上，一般我们都是用 Apache 这个软件来发送网页给网民。Apache 在运行时就会创建很多进程，分别负责不同的任务。一般的数据库软件，例如 MySQL，PostgreSQL 也是如此。

因此，假如你发现一个程序创建了好多个进程的话，并没有什么好吃惊的。

在 Windows 中，我们要查看系统中运行的进程，会使用 Ctrl + Alt + Delete 快捷键，调出任务管理器，然后点击“进程”这个菜单。

在 Linux 中，我们有两个命令可以帮助我们查看系统中运行的进程。

ps: 进程的静态列表

ps 是 Process Status 的缩写，process 是英语“进程”的意思，status 是“状态”的意思，所以 ps 命令用于显示当前系统中的进程。

ps 命令显示的进程列表不会随时间而更新，是静态的，只是运行 ps 命令当时的那个状态，或者说是一个进程的快照，英语称为 snapshot，就好像照了一张照片一样。

我们试着不加任何参数直接运行 ps 命令：

```
File Edit View Search Terminal Help
oscar@oscar-laptop:~$ ps
  PID TTY          TIME CMD
 1812 pts/0        00:00:00 bash
 1963 pts/0        00:00:00 ps
oscar@oscar-laptop:~$
```

可以看到显示结果有四列：

- PID：进程号，pid 是 process identifier 的缩写，每个进程有唯一的进程号。之后我们学习如何结束进程时需要用到进程号。
- TTY：进程运行所在的终端。pts 上面我们已经讲过，是 pseudo terminal slave 的缩写，表示“伪终端从属”。如果我新开一个图形终端，那么显示名称为 pts/0；如果我再开一个图形终端，那么它的名字就是 pts/1。依次类推。
- TIME：进程运行了多久。
- CMD：产生这个进程的程序名。如果你在进程列表中看到有好几行都是同样的程序名，那么就是同样的程序产生了不少一个进程（例如 MySQL 程序）。

在我的情况，我们运行 ps 命令时显示了两个进程正在运行，一个是 bash，也就是操控当前终端的 Shell 程序；另一个是 ps 命令自己。

两个进程，这就是全部了？

当然不是。ps 命令不带参数使用时，只会列出当前运行 ps 命令的用户在当前这个终端中所运行的进程。有好多进程是 root 用户运行的，就没列出来。还有的进程，虽然也是当前用户运行的，但不是在当前的终端里，因此也没列出来。

ps 命令有很多不同的参数。你可以用 man ps 来查看，可不要被众多的选项参数吓到。

我们不可能介绍每个参数，下面就介绍一些常用的参数吧。

ps -ef：列出所有进程

-ef 参数可以使 ps 命令列出所有用户在所有终端的所有进程。

```
ps -ef
```



```
File Edit View Search Terminal Help
root      1989    2 0 21:52 ?      00:00:00 [kworker/u2:0]
root        1    0 0 21:31 ?      00:00:01 /sbin/init splash
root      216    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-journald
root      248    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-udev
systemd+  303    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-resolved
syslog    454    1 0 21:31 ?      00:00:00 /usr/sbin/rsyslogd -n
root      456    1 0 21:31 ?      00:00:00 /usr/sbin/ModemManager
root      458    1 0 21:31 ?      00:00:01 /usr/lib/udev/snapd/snapd
root      460    1 0 21:31 ?      00:00:00 /usr/lib/udev/udisks2/udisksd
root      465    1 0 21:31 ?      00:00:00 /usr/sbin/cron -f
root      471    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-logind
root      479    1 0 21:31 ?      00:00:00 /usr/sbin/acpid
avahi     480    1 0 21:31 ?      00:00:00 avahi-daemon: running [oscar-lap
avahi     498    480 0 21:31 ?      00:00:00 avahi-daemon: chroot helper
root      485    1 0 21:31 ?      00:00:00 /usr/bin/python3 /usr/bin/networ
message+  493    1 0 21:31 ?      00:00:00 /usr/bin/dbus-daemon --system --
root      537    1 0 21:31 ?      00:00:00 /usr/sbin/NetworkManager --no-da
root      630    537 0 21:31 ?      00:00:00 /sbin/dhclient -d -q -sf /usr/
root      538    1 0 21:31 ?      00:00:00 /sbin/wpa_supplicant -u -s -0 /r
root      539    1 0 21:31 ?      00:00:00 /usr/lib/AccountsService/account
root      573    1 0 21:31 ?      00:00:00 /usr/lib/policykit-1/polkitd --n
root      612    1 0 21:31 ?      00:00:00 /usr/bin/python3 /usr/share/unat
whoopsie  652    1 0 21:31 ?      00:00:00 /usr/bin/whoopsie -f
kernoops  656    1 0 21:31 ?      00:00:00 /usr/sbin/kerneloops --test
```

可以看到，用了 `-ef` 参数，因为列出所有用户的所有进程，所以比刚才不加参数的 `ps` 命令多了一些列。

例如第一列 UID，是 `user identifier` 的缩写，`user` 是“用户”的意思，`identifier` 是“标识符”的意思。所以 UID 是表示“用户名”，也就是运行进程的用户。

ps -efH: 以乔木状列出所有进程

```
ps -efH
```

比上面的 `-ef` 多加了一个 `H` 参数，可以使 `ps` 命令按照乔木状列出进程。有的进程是某些进程的子进程。

```
File Edit View Search Terminal Help
root      1989    2 0 21:52 ?      00:00:00 [kworker/u2:0]
root        1    0 0 21:31 ?      00:00:01 /sbin/init splash
root      216    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-journald
root      248    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-udev
systemd+  303    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-resolved
syslog    454    1 0 21:31 ?      00:00:00 /usr/sbin/rsyslogd -n
root      456    1 0 21:31 ?      00:00:00 /usr/sbin/ModemManager
root      458    1 0 21:31 ?      00:00:01 /usr/lib/udev/snapd/snapd
root      460    1 0 21:31 ?      00:00:00 /usr/lib/udev/udisks2/udisksd
root      465    1 0 21:31 ?      00:00:00 /usr/sbin/cron -f
root      471    1 0 21:31 ?      00:00:00 /lib/systemd/systemd-logind
root      479    1 0 21:31 ?      00:00:00 /usr/sbin/acpid
avahi     480    1 0 21:31 ?      00:00:00 avahi-daemon: running [oscar-lap
avahi     498    480 0 21:31 ?      00:00:00 avahi-daemon: chroot helper
root      485    1 0 21:31 ?      00:00:00 /usr/bin/python3 /usr/bin/networ
message+  493    1 0 21:31 ?      00:00:00 /usr/bin/dbus-daemon --system --
root      537    1 0 21:31 ?      00:00:00 /usr/sbin/NetworkManager --no-da
root      630    537 0 21:31 ?      00:00:00 /sbin/dhclient -d -q -sf /usr/
root      538    1 0 21:31 ?      00:00:00 /sbin/wpa_supplicant -u -s -0 /r
root      539    1 0 21:31 ?      00:00:00 /usr/lib/AccountsService/account
root      573    1 0 21:31 ?      00:00:00 /usr/lib/policykit-1/polkitd --n
root      612    1 0 21:31 ?      00:00:00 /usr/bin/python3 /usr/share/unat
whoopsie  652    1 0 21:31 ?      00:00:00 /usr/bin/whoopsie -f
kernoops  656    1 0 21:31 ?      00:00:00 /usr/sbin/kerneloops --test
```

比如上图中，我们可以看到 `/sbin/init` 这个程序（进程）就启动了其他的进程，比如 `rsyslogd`，`cron`，`snapped`等等。

ps -u 用户名: 列出此用户运行的进程

```
ps -u
```

我们就用当前用户来测试看看：

```
oscar@oscar-laptop:~$ ps -u oscar
PID TTY          TIME CMD
 939 ?            00:00:00 systemd
 940 ?            00:00:00 (sd-pam)
 953 ?            00:00:00 gnome-keyring-d
 957 tty1         00:00:00 gdm-x-session
 959 tty1         00:00:02 Xorg
 963 ?            00:00:00 dbus-daemon
 967 tty1         00:00:00 gnome-session-b
1067 ?            00:00:00 VBoxClient
1068 ?            00:00:00 VBoxClient
1077 ?            00:00:00 VBoxClient
1078 ?            00:00:00 VBoxClient
1084 ?            00:00:00 VBoxClient
1085 ?            00:00:00 VBoxClient
1090 ?            00:00:00 VBoxClient
1091 ?            00:00:02 VBoxClient
1104 ?            00:00:00 ssh-agent
1106 ?            00:00:00 at-spi-bus-laun
1111 ?            00:00:00 dbus-daemon
1114 ?            00:00:00 at-spi2-registr
1132 tty1         00:00:09 gnome-shell
1146 ?            00:00:00 gvfsd
1151 ?            00:00:00 gvfsd-fuse
```

可以看到有不少进程，单独用 `ps` 的时候只有两个进程，那是因为有好些进程不是在 `pts/0` 那个终端里运行的。

关于 `ps` 命令，我们暂时介绍到这里。下一课，我们接着来讲另一个命令：`top`，它可以显示进程的动态列表。

小结

1. Linux 是多任务多用户的操作系统。
2. `w` 命令会显示当下有哪些用户在系统上登录着，在做什么，还可以获知系统时间，持续运行时间，负载等信息。
3. `ps` 命令显示进程的快照（snapshot），是静态的。

今天的课就到这里，一起加油吧！