



链滴

# 必须要知道的 Linux 技巧

作者: [An](#)

原文链接: <https://ld246.com/article/1358385729723>

来源网站: 链滴

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

今天早上在GR上面看到的一篇很好的文章，转过来收藏下。原文链接请看文章末尾

这篇文章来源于Quroa的一个问答《What are some time-saving tips that every Linux user should know?》—— Linux用户有哪些应该知道的提高效率的技巧。我觉得挺好的，总结得比较好，其转过来，并加了一些自己的理解。首先，我想告诉大家，在Unix/Linux下，最有效率技的不是操作图形界面，而是命令行操作，因为命令行意味着自动化。如果你看过《你可能知道的Shell》以及《28个Unix/Linux的命令行神器》你就会知道Linux有多强大，这个强大完全来自命令行，于是，就算你不知道怎么去做一个环保主义的程序员，至少他们可以让你少熬点夜，从而有益于你的身体健康和性生活。下面是一个有点长的列表，正如作者所说，你并不需要知道所有的这些东西，但是如果你还在很沉重地在使用Linux的话，这些东西都值得你看一看。（注：如果你想知道下面及到的命令的更多的用法，你一定要man一点。对于一些命令，你可以需要先yum或apt-get来安装下，如果有什么问题，别忘了Google。如果你要Baidu的话，我仅代表这个地球上所有的生物包括微生物甚至细菌病毒和小强BS你到宇宙毁灭）

#### 基础

<ul>

<li>学习 Bash。你可以man bash来看看bash的东西，并不复也并不长。你用别的shell也行，但是bash是很强大的并且也是系统默认的。（学习zsh或tsch只会让在很多情况下受到限制）</li>

</ul>

<ul>

<li>学习 vim。在Linux下，基本没有什么可与之竞争的编译器（就算是一个Emacs或Eclipse的重度用户）。你可以看看《简明vim攻略》和《Vim的冒险游戏》以及《给序员的Vim速查卡》还有《把Vim变成一个编程的IDE》等等。</li>

</ul>

<ul>

<li>了解 ssh。明白不需要口令的用户认证（通过ssh-agent, ssh-add），学用ssh翻墙，用scp而不是ftp传文件，等等。你知道吗？scp 远端的时候，你可以按tab键来查看远端目录和文件（当然，需要无口令的用户认证），这都是bash的功劳。</li>

</ul>

<p>&nbsp;</p>

<ul>

<li>熟悉bash的作业管理，如：&，Ctrl-Z, Ctrl-C, jobs, fg, bg, kill，等等。当然，你也要知道Ctrl+\ (SIGQUIT) 和Ctrl+C (SIGINT) 的区别。</li>

</ul>

<ul>

<li>简单的文件管理：ls 和 ls -l (你最好知道“ls -l”的每一列的意思 less, head, tail 和 tail -f, ln 和 ln -s (你知道明白hard link和soft link的不同和优缺点), chown, chmod, du (如果你想看看磁盘的大小 du -sk \*), df, mount。当然，原作者忘了find命令。</li>

</ul>

<ul>

<li>基础的网络管理：ip 或 ifconfig, dig。当然，原作者还忘了如netstat, ping, traceroute, 等等</li>

</ul>

<ul>

<li>理解正则表达式，还有grep/egrep的各种选项。比如：-o, -A, 和 -B 这选项是很值得了解的。</li>

</ul>

<ul>

<li>学习使用 apt-get 和 yum 来查找和安装软件（前者的经典分发版是Ubuntu，后者的经典分发版是Redhat），我还建议你试着从源码编译安装软件。</li>

</ul>

<p>日常</p>

<ul>

<li>在 bash 里，使用 Ctrl-R 而不是上下光标键来查找历史命令。</li>

</ul>

<ul>

<li>在 bash里，使用 Ctrl-W 来删除最后一个单词，使用 Ctrl-U 来删除一行。请man bash后查找Readline Key Bindings一节来看看bash的默认热键，比如：Alt-. 把上一次命令的最后一个参数打出来而Alt-\* 则列出你可以输入的命令。 </li>

</ul>

<ul>

<li>回到上一次的工作目录： cd - &nbsp; (回到home是 cd ~) </li>

</ul>

<ul>

<li>使用 xargs。这是一个很强大的命令。你可以使用-L来限定有多少个命令，也可以用-P来指定并的进程数。如果你不知道你的命令会变成什么样，你可以使用xargs echo来看看会是什么样。当然， {} 也很好用。示例： </li>

</ul>

```
find . -name *.py | xargs grep some_function
cat hosts | xargs -l{} ssh root@{} hostname</pre>
```

<ul>

<li>pstree -p 可以帮你显示进程树。（读过我的那篇《一个fork的面试题》的人应该都不陌生） </li>

</ul>

<ul>

<li>使用 pgrep 和 pkill 来找到或是kill 某个名字的进程。（-f 选项很有用）。</li>

</ul>

<ul>

<li>了解可以发给进程的信号。例如：要挂起一个进程，使用 kill -STOP [pid]. 使用 man 7 signal 查看各种信号，使用kill -l 来查看数字和信号的对应表</li>

</ul>

<ul>

<li>使用 nohup 或 &nbsp;disown 如果你要让某个进程运行在后台。 </li>

</ul>

<ul>

<li>使用netstat -lntp来看看有侦听在网络某端口的进程。当然，也可以使用 lsof。 </li>

</ul>

<ul>

<li>在bash的脚本中，你可以使用 set -x 来debug输出。使用 set -e 来当有错误发生的时候abort行。考虑使用 set -o pipefail 来限制错误。还可以使用trap来截获信号（如截获ctrl+c） 。 </li>

</ul>

<ul>

<li>在bash 脚本中， subshells (写在圆括号里的) 是一个很方便的方式来组合一些命令。一个常用例子是临时地到另一个目录中，例如： </li>

</ul>

```
# do something in current dir
(cd /some/other/dir; other-command)
# continue in original dir</pre>
```

<ul>

<li>在 bash 中，注意那里有很多的变量展开。如：检查一个变量是否存在: \${name:?error message}。如果一个bash的脚本需要一个参数，也许就是这样一个表达式 input\_file=\${1:?usage: \$0 input\_file}。一个计算表达式： i=\$(( (i + 1) % 5 ))。一个序列： {1..10}。截断一个字符串： \${var%suffix} 和 {var#prefix}。 示例： if var=foo.pdf, then echo \${var%.pdf}.txt prints "foo.txt" .</li>

</ul>

<ul>

<li>通过 &lt;(some command) 可以把某命令当成一个文件。示例：比较一个本地文件和远程文件 /tc/hosts： diff /etc/hosts &lt;(ssh somehost cat /etc/hosts)</li>

</ul>

<ul>

- <li>了解什么叫 “here documents” ， 就是诸如 cat &lt;&lt;EOF 这样的东西。 </li>

</ul>

- <li>在 bash中， 使用重定向到标准输出和标准错误。如： some-command &gt;logfile 2&gt;&a p;1。另外， 要确认某命令没有把某个打开了的文件句柄重定向给标准输入， 最佳实践是加上 “&lt;/d v/null” ， 把/dev/null重定向到标准输入。 </li>

</ul>

- <li>使用 man ascii 来查看 ASCII 表。 </li>

</ul>

- <li>在远端的 ssh 会话里， 使用 screen 或 dtach 来保存你的会话。（参看《28个Unix/Linux的命令行神器》） </li>

</ul>

- <li>要来debug Web， 试试curl 和 curl -I 或是 wget 。 我觉得debug Web的利器是firebug， curl wget是用来抓网页的， 呵呵。 </li>

</ul>

- <li>把 HTML 转成文本： lynx -dump -stdin</li>

</ul>

- <li>如果你要处理XML， 使用 xmlstarlet</li>

</ul>

- <li>对于 Amazon S3， s3cmd 是一个很方便的命令（还有点不成熟） </li>

</ul>

- <li>在 ssh中， 知道怎么来使用ssh隧道。通过 -L or -D (还有-R) ， 翻墙神器。 </li>

</ul>

- <li>你还可以对你的ssh 做点优化。比如， .ssh/config 包含着一些配置：避免链接被丢弃， 链接新的ost时不需要确认， 转发认证， 以前使用压缩（如果你要使用scp传文件）： </li>

</ul>

```
TCPKeepAlive=yes
ServerAliveInterval=15
ServerAliveCountMax=6
StrictHostKeyChecking=no
Compression=yes
ForwardAgent=yes</pre>

</ul>



- <li>如果你有输了个命令行， 但是你改变注意了， 但你又不想删除它， 因为你要在历史命令中找到它但你也不想执行它。那么， 你可以按下 Alt-# ， 于是这个命令关就被加了一个#字符， 于是就被注释了。 </li>



</ul>



<p> <strong>数据处理&nbsp;&nbsp;&lt;/strong> </p>



- <li>了解 sort 和 uniq 命令 (包括 uniq 的 -u 和 -d 选项).</li>



</ul>



- <li>了解用 cut, paste, 和 join 命令来操作文本文件。很多人忘了在cut前使用join。 </li>



</ul>



- <li>如果你知道怎么用sort/uniq来做集合交集、并集、差集能极大地促进你的工作效率。假设有两

```

文本文件a和b已被 uniq 了，那么，用 sort/uniq 会是最快的方式，无论这两个文件有多大（sort 不被内存所限，你甚至可以使用 -T 选项，如果你的 /tmp 目录很小）

```
cat a b | sort | uniq &gt; c # c is a union b 并集
cat a b | sort | uniq -d &gt; c # c is a intersect b 交集
cat a b b | sort | uniq -u &gt; c # c is set difference a - b 差集
```

了解字符集相关的命令行工具，包括排序和性能。很多的Linux安装程序都会设置LANG 或是它和字符集相关的环境变量。这些东西可能会让一些命令（如：sort）的执行性能慢N多倍（注：就是你用UTF-8编码文本文件，你也可以很安全地使用ASCII来对其排序）。如果你想Disable那个i18n使用传统的基于byte的排序方法，那就设置export LC\_ALL=C（实际上，你可以把其放在 .bashrc）。如果这设置这个变量，你的sort命令很有可能会是错的。

了解 awk 和 sed，并用他们来做一些简单的数据修改操作。例如：求第三列的数字之和：awk { x += \$3 } END { print x }。这可能会比Python快3倍，并比Python的代码少三倍。

使用 shuf 来打乱一个文件中的行或是选择文件中一个随机的行。

了解sort命令的选项。了解key是什么（-t和-k）。具体说来，你可以使用-k1,1来对第一列排序 -k1来对全行排序。

Stable sort (sort -s) 会很有用。例如：如果你要想对两列排序，先是以第二列，然后再以第一，那么你可以这样： sort -k1,1 | sort -s -k2,2

我们知道，在bash命令行下，Tab键是用来做目录文件自动完成的事的。但是如果你想输入一个ab字符（比如：你想在sort -t选项后输入<tab>字符），你可以先按Ctrl-V，然后再按Tab键，可以输入<tab>字符了。当然，你也可以使用\$'\t'。

如果你想查看二进制文件，你可以使用hd命令（在CentOS下是hexdump命令），如果你想编二进制文件，你可以使用bvi命令（<http://bvi.sourceforge.net/>）

另外，对于二进制文件，你可以使用strings（配合grep等）来查看二进制中的文本。

对于文本文件转码，你可以试一下 iconv。或是试试更强的 uconv 命令（这个命令支持更高级Unicode编码）

如果你要分隔一个大文件，你可以使用split命令（split by size）和csplit命令（split by a pattern）。

## 系统调试

如果你想知道磁盘、CPU、或网络状态，你可以使用 iostat, netstat, top (或更好的 htop), 还有 stat 命令。你可以很快地知道你的系统发生了什么事。关于这方面的命令，还有iftop, iotop等（参《28个Unix/Linux的命令行神器》）

<ul>

<li>要了解内存的状态，你可以使用free和vmstat命令。具体来说，你需要注意“cached”的值，个值是Linux内核占用的内存。还有free的值。</li>

</ul>

<ul>

<li>Java 系统监控有一个小的技巧是，你可以使用kill -3 &lt;pid> 发一个SIGQUIT的信号给JV，可以把堆栈信息（包括垃圾回收的信息）dump到stderr/logs。</li>

</ul>

<ul>

<li>使用 mtr 会比使用 traceroute 要更容易定位一个网络问题。</li>

</ul>

<ul>

<li>如果你要找到哪个socket或进程在使用网络带宽，你可以使用 iftop 或 nethogs。</li>

</ul>

<ul>

<li>Apache的一个叫 ab 的工具是一个很有用的，用quick-and-dirty的方式来测试网站服务器的性负载的工作。如果你需要更为复杂的测试，你可以试试 siege。</li>

</ul>

<ul>

<li>如果你要抓网络包的话，试试 wireshark 或 tshark。</li>

</ul>

<ul>

<li>了解 strace 和 ltrace。这两个命令可以让你查看进程的系统调用，这有助于你分析进程的hang哪了，怎么crash和failed的。你还可以用其来做性能profile，使用 -c 选项，你可以使用-p选项来attach上任意一个进程。</li>

</ul>

<ul>

<li>了解用ldd命令来检查相关的动态链接库。注意：ldd的安全问题</li>

</ul>

<ul>

<li>使用gdb来调试一个正在运行的进程或分析core dump文件。参看我写的《GDB中应该知道的个调试方法》</li>

</ul>

<ul>

<li>学会到 /proc 目录中查看信息。这是一个Linux内核运行时记录的整个操作系统的运行统计和信，比如： /proc/cpuinfo, /proc/xxx/cwd, /proc/xxx/exe, /proc/xxx/fd/, /proc/xxx/smaps.</li>

</ul>

<ul>

<li>如果你调试某个东西为什么出错时，sar命令会有用。它可以让你看看 CPU, 内存, 网络, 等的统信息。</li>

</ul>

<ul>

<li>使用 dmesg 来查看一些硬件或驱动程序的信息或问题。</li>

</ul>

<p>作者最后加了一个免责声明：Disclaimer: Just because you&nbsp;&nbsp;<em>can</em>&nbsp;&nbsp;do something in bash, doesn' t necessarily mean you should. ;)</p>

<p>转自：http://coolshell.cn/articles/8883.html</p>