

Linux command summary

Command	description
• ll	查看当前目录下的所有文件的权限
• ls -al	显示所有（包括隐藏的文件）文件
• date	查看系统时间
• pwd	显示当前目录
• mkdir/rmdir	新建或删除一个新的/空目录
• cd -	返回上一次 cd 的目录
• cat -n filename	查看当前文件内容，顺便输出行号
• mv file1 file2	将文件名 file1 改为 file2
• ctrl+z	暂停目前的命令
• ctrl+c	终止目前的命令
• ctrl+s	暂停屏幕的输出
• ctrl+q	恢复屏幕的输出
• jobs	查看当前有多少 job 在执行
• bg	查看当前后台执行的 job 情况
• fg % id	将后台运行的某个进程放到前台执行
• ps	查看当前执行的进程数
• kill jobid	关闭当前的某个进程
• ssh -X	登陆服务器，加 option X 表示服务器可以可视化
• su user	切换当前账户到 user 这个账户
• who	查看当前服务器登录的用户情况
• who am i	显示当前 local 用户登录信息
• exit	退出当前执行的账户
• xeyes	查看当前服务器是否可视化，如果可以就有 eyes 弹窗显示
• chmod xxx filename	修改文件的 permission（readable，writeable，executable）
• touch filename	当前目录下新建空文件 filename
• rm -rf	强制执行删除一个文件
• double tab	查看当前用户可执行的文件数或者操作数
• vimdiff file1 file2	比较当前两个文件的差异
• echo str	当前目录下查找含有特定字符的文件
• alias command='str'	对可执行文件设置 alias 方便经常执行调用
• more/less	一页一页地显示文件内容，向下翻页查看文件内容
• tail -f file	实时显示文件的状态，特别执行 job 时，查看 log 文件内容
• man command	查看 command 的操作指南，特别是 option 的 usage

About test

- `test -e demo.sh && echo "exist" || echo "not exist" #test the file demo.sh exist or not.`
- `test -z $filename && echo "you must input a filename"&& exit 0`
- `test ! -e $filename && echo "the filename does not exist" && exit 0`
- `test -f $filename && echo filetype="file is a filename"&& exit 0`
- `test -d $filename && echo filetype="file is a directory "&& exit 0`
- `test -r $filename && perm="readable"`
- `test -w $filename && perm="$perm writable"`
- `test -x $filename && perm="$perm executable" 1`

About file types

- Regular file `-frwxrwxrwx`
- ASCII file
- Binary file
- Date file
- Directory `drwxrwxrwx`
- Link `lrwxrwxrwx`
- Device/block file `brwxrwxrwx` /character file `crwxrwxrwx` /sockets file `srwxrwxrwx`
- Sockets
- FIFO, pipe

About vi/vim editors

➤ Regular mode

- `h/j/k/l` turn left/down/up/right
- `n<space>` turn right by n steps
- `G` turn to the bot
- `gg` turn to the top
- `/word` find the word downward
- `n` double find /word
- `N` double find /word reverse
- `?word` find the word upward
- `x, X` delete strings backward or forward
- `dd` delete the whole line
- `ndd` delete n lines downward
- `yy` copy the whole line
- `p, P` paste the copy context after/before this line.

- u rebuilt the last condition
- O turn to the head of line
- \$ turn to the end of line
- fa find the 1st string "a" forward
- 3fa fine the 3rd string "a" forward
- Ctrl+z 将当前的 vi,vim editor 放置到后台执行，回来用 fg
- Env 查看当前终端有关的环境变量
- tr 'a' 'b' 将查找结果中的 a 全部替换成 b

➤ Editor mode (insert mode)

- i change from regular to editor mode
- esc change from editor to regular mode
- ctrl+p auto polishing (自动补齐功能)

➤ Command mode

- : wq save change and exit
- : q! don't save change and exit
- : set nu/nonu set row numbers auto

➤ Multiple windows operate

- :sp filename in command mode
- Ctrl+w+(j/k) switch windows
- Ctrl+w+q exit multiple windows operate

About retarget data

- Com1 && com2 如果 com1 正确执行，则开始执行 com2，如果没有，com2 不执行
- Com1 || com2 如果 com1 正确执行，则 com2 不执行，如果没有，com2 执行

About pipe

- cut -d 'plits str' -f fields 查找出用 splits str 分割的变量的第 fields 的区域。
- cut -c fields 排列整齐的信息。
- grep [-acnv] --color=auto 'str' filename 按行查找文件中的指定字符串。
- xargs 由于很多 command 并不支持 pipe，所以加上 xargs 在前面可以读取 stdin.
- awk -F : '{print \$1 \$3 \$5}' 将得到的列表信息的第 1,3,5 列输出。
- > filename 将结果以文件 file 的形式保存，方便查看。

Linux setting usage

➤ Set display row number as default

1. `cp /usr/share/vim/vimrc ~/.vimrc` (if Redhot system, change usr to etc)
2. `vi ~/.vimrc`
3. go into editor mode
4. syntax on
5. set nu

FYI, check a blog link: <http://www.cnblogs.com/yjmyzz/p/4019783.html>

➤ Set open vim editor as default when using vi editor

1. `which vi vim`(display the directory)
2. `rm /bin/vi` or `mv /bin/vi /bin/vi/old`
3. `cp /usr/bin/vim /bin/vi` or `alias vi ='/usr/bin/vim'`

➤ Change strings in files by one time (very useful)

1. `sed -i "s/old str/new str/g" `grep old str -rl dir``

➤ About close process

1. jobs and kill % num
2. ps and kill jobid

Linux office usage

➤ Print in terminal

1.echo 自带换行符

Option: -n 标志可以忽略结尾换行符

-e 接受双引号字符串内的转义字符

2.Printf 没有自带换行符，可以通过加 “\n” 来添加换行

➤ get program's ID

Pgrep programName

➤ get environment variable

Cat /proc/\$PID/environ

➤ set variable value in shell

Var=value, should look out the diff between 'var=value' and 'var = value' (equal)

➤ set temporary environment variable

Export PATH

➤ set forever environment variable

Vi /etc/profile

Add "export "PATH""

Source /etc/profile or ./etc/profile

➤ get the length of a variable

Echo \${#var}

➤ 对 variable 的算术运算使用 let，但是需要注意的是 let 只能进行整数的操作，而 bc 可以用于浮点型的算术运算。

Var1=4;var2=5;let result=var1+var2

Echo "scale=2;var1*var2" | bc (将精度设置为 2 位小数)

Echo "obase=10;ibase=2; \$result" | bc (将 result 这个变量的值由输入的 2 进制转换为输出的 10 进制)

➤ get some terminal info (tput and stty tools)

1. 获取终端的行数和列数

Tput cols / tput lines

2. 打印当前终端名

Tput longname

...

➤ 获取设置日期和延时

1. date "+%d %B %Y"

2. 打印纪元时: date +%s

3. 将日期串输出为纪元时: date --date "Jan 20 2001" +%A

4. 检查一组命令花费的时间

Start=\$(date +%s)

End=\$(date +%s)

Diff=\$((end-start))

➤ 调试脚本

Bash -x script.sh / sh -x script.sh

➤ 函数格式

Function fname(){

statements

}

或

Fname(){

statements

```
}
```

导出函数: `export -f fname`

`Cmd;`

`Echo $?;` #输出命令 `cmd` 的返回值（退出状态）。如果命令成功退出，退出状态应该是 0，否则是非 0。

➤函数格式

`[condition] && action` #如果 `condition` 条件为真，执行 `action`

`[condition] || action` #如果 `condition` 条件为假，执行 `action`

`[condition1 -a condition2]` #逻辑与

`[condition1 -o condition2]` #逻辑或

比较符: `gt lt ge le eq ne`

`if [condition]` 可以写成 `test condition`

如果是字符串的比较，需比如: `[[$str1 = $str2]]`

➤命令乐趣-cat

`Cat file1 file2...`: 可以拼接多个文件一起显示

`Cat` 可以实现将输入文件的内容与标准输入拼接在一起

`Echo "hello" | cat - file1` “-”理解为标准输入文本的文件名。

`Cat` 可以实现将文本文件中的多余空行压缩到一行显示

`Cat -s file | tr -s '\n'` 可以将多个 `\n` 字符压缩成单个 `\n`

➤录制与回放 terminal 会话 `script`、`scriptreplay`

`Script -t 2> timing.log -a output.session`

Type commands;

Exit

其中 `timing` 用于储存时序信息，描述每个 `command` 在何时运行，`output` 用于存储命令输出，`-t` 选项用于将时序数据导入 `stderr`。2>则将 `stderr` 重定向到 `timing` 中。借助这两个文件，可以利用下面命令回放命令执行过程：

Scriptreplay timing.log output.session

Script 实现多用户之间视频通话

1.建立通话连接

Terminal1（广播员）：Mkfifo scriptfifo

Terminal2（听众）：cat scriptfifo

2.开始通话

Terminal1: script -f scriptfifo

Commands

Exit

➤命令乐趣-find

1.find base_path 查找当前目录以及子目录中所有文件和文件夹。

2.Find path -name "*.sh" -print 查找当前目录以及子目录中包含 sh 后缀的文件。

3.Find path \(-name "*.sh" -o -name "*.out" \) -print 查找多个类型文件。

4.Find path ! -name "*.sh" -print 反向匹配查找出后缀不是 sh 的文件。

5.Find path -maxdepth num1 -mindepth num2 -print 设定搜索的范围。

6.find path -type f-print 根据文件类型搜索

文件类型	类型参数
普通文件	F
符号链接	l
目录	d
字符设备	c
块设备	b
套接字	s
Fifo	p

7.基于访问时间、修改时间、变化时间的查找

`-atime -mtime -ctime`

`-amin -mmin -cmin`

Ex: `find . -type f -amin -7 -print` 打印出访问时间小于 7 分钟的所有文件。

8. 基于文件大小-size

`find . -type -size +2k -print` 打印出文件大小大于 2kb 的文件。

文件大

小

b	块
c	字节
w	字
k	千字节
M	兆
G	吉字节

9.删除匹配的文件 -delete

10.基于文件所有权的匹配 -perm

11.基于用户的文件的匹配 -user

Find `. -type -f -user kevin - print` 打印出用户 user 拥有的所有文件。

12.find 最强大的 option: -exec

`-exec` 可以与其他命令结合起来

Ex: `# find . -type f -user root -exec chown kevin {} \;`

`{}` 是一个特殊的字符串，与 `-exec` 结合使用，对于每一个匹配的文件，`{}` 会被替换成相应的文件名。

➤命令乐趣-xargs

管道可以将一个命令的 **stdout** 重定向到另一个命令的 **stdin**，但是有些命令只能以命令行参数的形式接受数据，而无法通过 **stdin** 接受数据流。**Xargs** 可以将标准输入数据转换成命令行参数，也可将单行或者多行文本输入转换成其他格式。

- 多行输入变成单行输入

`Cat file | xargs`

- 单行输入转换成多行输出

`Cat file | xargs -n 3` (每行由 3 个参数组成划分多行)

- 用 `xargs -d` 设定数据划分的指定定界符（默认 **xargs** 采用内部字段分割符 **IFS** 作为输入定界符）

