1、查看隐藏文件

|  |
| --- |
| ls -al |

2、删除目录

|  |
| --- |
| rm -rf 目录名 |

3、查看linux时间

|  |
| --- |
| date |

4、查询安装程序(java)

|  |
| --- |
| rpm -qa |grep java |

5、查找可执行文件

|  |
| --- |
| which java |

6、重新加载配置

|  |
| --- |
| source ~/.bash\_profile  source .bash\_profile |

7、yum卸载软件

|  |
| --- |
| yum remove maven |

vi编辑命令模式下，用/搜索，下一个用n，上一个用大写N（shift+n或者大写锁定键）

用问号？从文章结尾处搜索

我们打开别的文件，发现也被高亮了，怎么关闭高亮？

nohlsearch是（no highlight search缩写）

set nohlsearch

简写：noh或者set noh

配置好环境变量 .bash\_profile /etc/profile等文件

source .bash\_profile

ssh -p 2222 user@host

ssh命令登录远程机器

假定经过风险衡量以后，用户决定接受这个远程主机的公钥。  
    　　Are you sure you want to continue connecting (yes/no)? yes  
系统会出现一句提示，表示host主机已经得到认可。  
    　　Warning: Permanently added 'host,12.18.429.21' (RSA) to the list of known hosts.  
然后，会要求输入密码。  
    　　Password: (enter password)  
如果密码正确，就可以登录了。

sudo -u userprofile -i

debian 安装软件用apt-get

apt-get update

apt-get install openssh-server -y

apt-get install net-tools

apt-get install vim

ssh命令

ssh -p 58958 userprofile@169.44.23.158

ssh长传文件命令

rz

sz 文件名

复制文件夹

cp -rf 源文件夹 目标地址

删除文件夹

cp -rf 文件夹

cp -rf elasticsearch-6.4.2-cp1/ ./elasticsearch-6.4.2-cp2

修改文件执行、读写权限

chmod [a|u|g]+[w|r|x] 文件

修改文件所有者

chown[[-f](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#ph3200wmh3)][[-h](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#d722dea943jani)][[-R](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#dh3240wmh3)] Owner [ :Group ] { File ... | Directory ... }  
chown[-R](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#dh3240wmh3)[[-f](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#ph3200wmh3)][[-H](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#d722dea001)| [-L](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#d722dea002)| [-P](http://study.chyangwa.com/IT/AIX/aixcmds1/chown.htm#d722dea003)] Owner [ :Group ] { File ... | Directory ... }

查看端口状态

|  |
| --- |
| **netstat -an | grep 3306   //查看所有3306端口使用情况** |

1、w命令查看当前用户

[root@localhost /]# w  
04:31:52 up  1:45,  2 users,  load average: 0.00, 0.01, 0.05  
USER    TTY        FROM            LOGIN@  IDLE    JCPU  PCPU  WHAT  
root      tty1                                02:46      1:45m  0.02s  0.02s  -bash  
root      pts/0    192.168.146.1    03:19      0.00s    0.10s  0.02s  w

第一行分别是：当前时间、从开机到现在有多久、负载均衡  
      第二行是：每列的说明  
      第三行是：root用户的登录信息

2、who命令查看主机上的用户

[root@localhost /]# who  
root    tty1          2017-04-19 02:46  
root    pts/0        2017-04-19 03:19 (192.168.146.1)  
【附】：查看当前登录用户是谁  
whoami  （字母之间没有空格）

3、lastlog命令查看主机上的用户

[root@localhost /]# lastlog | head -n 15  
Username        Port        From                      Latest  
root                  pts/0      192.168.146.1        Wed Apr 19 03:19:47 -0400 2017  
bin                                                                \*\*Never logged in\*\*  
daemon                                                        \*\*Never logged in\*\*  
adm                                                              \*\*Never logged in\*\*  
lp                                                                  \*\*Never logged in\*\*  
sync                                                              \*\*Never logged in\*\*

tar压缩命令

参数：

-c ：create 建立压缩档案的参数；

-x ： 解压缩压缩档案的参数；

-z ： 是否需要用gzip压缩；

-v： 压缩的过程中显示档案；

-f： 置顶文档名，在f后面立即接文件名，不能再加参数

1 将tgz文件解压到指定目录

tar   zxvf    test.tgz  -C  指定目录

比如将/source/kernel.tgz解压到  /source/linux-2.6.29 目录

tar  zxvf  /source/kernel.tgz  -C /source/ linux-2.6.29

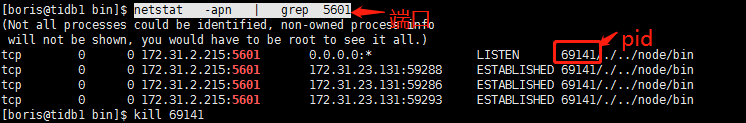
2 将指定目录压缩到指定文件

比如将linux-2.6.29 目录压缩到  kernel.tgz

 tar czvf   kernel.tgz   linux-2.6.29

查看端口占用情况

netstat -apn | grep 5601



杀掉进程

kill pid

日志分析：

查看日志|搜索cod免验证算法接口总耗时所在的行|用冒号分隔 在屏幕数据第二个字段|用秒分隔在屏幕输出第一个字段|排序

cat jolly-service-risk-busi.log | grep "cod免验证算法接口总耗时" | awk -F "：" '{print $2}' | awk -F "秒" '{print $1}' | sort |

cat jolly-service-risk-busi.log | grep "cod免验证算法接口总耗时" | wc -l

cat jolly-service-risk-busi.log | grep "cod免验证算法接口总耗时" | sort -k 14 19

cat jolly-service-risk-busi.log | grep "cod免验证算法接口总耗时" | awk -F '[：秒]' '$2>0.16 {print $2}' | wc -l

cat jolly-service-risk-busi.log | grep "cod免验证算法接口总耗时" | awk -F '[：秒]' '$2>0.16 {print $2}' | sort | wc -l

cat jolly-service-risk-busi.log | grep "cod免验证算法接口总耗时" | awk -F '[：秒]' '$2>0.16'

cat jolly-service-risk-busi.log | grep "cod免验证算法接口总耗时" | awk -F '[：秒]' '$2>0.16' | sort -k 2

根据分隔符分出字段，在取第二个字段的平均值，显示

grep "调用算法http耗时" jolly-service-risk-busi.log.copy |awk -F '[：秒]' '{sum += $2;} END { print "avg=" sum/NR}'

输出匹配字符串的行号

awk '/cod免验证算法接口总耗时/ {print NR}' jolly-service-risk-busi.log

跟grep -n ' cod免验证算法接口总耗时' jolly-service-risk-busi.log 相似

grep -niR '想要查找的字符' xxx

注：n，显示行号　R，查找所有文件包含子目录　i,忽略大小写 最后的“xxx”表示想要查找的文件名

grep -rn "hello,world!" ./  
  
./ : 表示路径为当前目录.  
  
-r 是递归查找  
  
-n 是显示行号

将一个文件的部分数据copy到另一个文件 ，相当于拆分文件

tail -n 10000 jolly-service-risk-busi.log>jolly-service-risk-busi.log.copy

也可以做合并

tail jolly-service-risk-busi.log>>jolly-service-risk-busi.log.copy

**日志分析查看——grep,sed,sort,awk运用**

https://blog.csdn.net/ultrani/article/details/6750434#content\_t4

# 在linux后台运行脚本的方法和命令

置顶 2018年05月03日 18:15:55 [izar](https://me.csdn.net/ruiyelp) 阅读数：6971 标签： [后台运行脚本](http://so.csdn.net/so/search/s.do?q=%E5%90%8E%E5%8F%B0%E8%BF%90%E8%A1%8C%E8%84%9A%E6%9C%AC&t=blog)[linux脚本](http://so.csdn.net/so/search/s.do?q=linux%E8%84%9A%E6%9C%AC&t=blog)[后台运行](http://so.csdn.net/so/search/s.do?q=%E5%90%8E%E5%8F%B0%E8%BF%90%E8%A1%8C&t=blog) 更多

个人分类： [linux](https://blog.csdn.net/ruiyelp/article/category/6742632)

版权声明：本文为博主izar原创文章，未经博主izar允许也可转载。 https://blog.csdn.net/ruiyelp/article/details/80184249

## 后台运行脚本

1. 执行脚本test.sh:./test.sh
2. 中断脚本test.sh：ctrl+c
3. 在1的基础上将运行中的test.sh，切换到后台并暂停：ctrl+z
4. 执行ctrl+z后，test.sh在后台是暂停状态（stopped）,使用命令：bg number让其在后台开始运行（“number”是使用jobs命令查到的 [ ]中的数字，不是pid）
5. **直接在后台运行脚本test.sh**:./test.sh &
6. 查看当前shell环境中已启动的任务情况：jobs
7. 将test.sh切换到前台运行：fg %number（”number”为使用jobs命令查看到的 [ ] 中的数字，不是pid）
8. 中断后台运行的test.sh脚本：先fg %number切换到前台，再ctrl+c；或是直接kill %number

以上两种在后台运行test.sh的方法，当遇到退出当前shell终端时，后台运行的test.sh也就结束了。这是因为以上两种方法使得test.sh在后台运行时，运行test.sh进程的父进程是当前shell终端进程，关闭当前shell终端时，父进程退出，会发送hangup信号给所有子进程，子进程收到hangup以后也会退出。所以要想退出当前shell终端时test.sh继续运行，则需要使用nohup忽略hangup信号。

1. **不中断的在后台运行test.sh**：nohup ./test.sh &（test.sh的打印信息会输出到当前目录下的nohup.out中）
2. 使用jobs可看到test.sh处于running状态
3. 使用ps -ef |grep test.sh可查看到正在运行的test.sh脚本进程
4. 退出当前shell终端，再重新打开，使用jobs看不到正在运行的test.sh，但使用ps -ef可以看到

在后台不中断的运行test.sh，可以使用nohup忽略hangup信号，或者使用setsid将其父进程改为init进程（进程号为1）

1. **不中断的在后台运行test.sh另一个命令**：setsid ./test.sh &
2. 使用ps -ef |grep test.sh可看到test.sh进程的父进程id为1

## 测试脚本

#！/bin/bash

int=1

while(( $int<=100 ))

do

echo $int

let "int++"

sleep 1

done

jps | grep [appname]

netstat -anp | grep [pid]

ping hostname, telnet hostname port

vim

gg:命令将光标移动到文档开头  
  
G:命令将光标移动到文档末尾

* 0 Move to the begining of the line
* $ Move to the end of the line
* 1G Move to the first line of the file
* G Move to the last line of the file
* nG Move to nth line of the file
* :n Move to nth line of the file
* H Move to top of screen
* M Move to middle of screen
* L Move to botton of screen

**一、tail命令语法**

tail [ -f ] [ -c Number | -n Number | -m Number | -b Number | -k Number ] [ File ]  
参数解释：  
-f 该参数用于监视File文件增长。  
-c Number 从 Number 字节位置读取指定文件  
-n Number 从 Number 行位置读取指定文件。  
-m Number 从 Number 多字节字符位置读取指定文件，比方你的文件假设包括中文字，假设指定-c参数，可能导致截断，但使用-m则会避免该问题。  
-b Number 从 Number 表示的512字节块位置读取指定文件。  
-k Number 从 Number 表示的1KB块位置读取指定文件。  
File 指定操作的目标文件名称  
上述命令中，都涉及到number，假设不指定，默认显示10行。Number前面可使用正负号，表示该偏移从顶部还是从尾部開始计算。  
tail可运行文件一般在/usr/bin/以下。

**二、tail命令使用方法演示例子**

1、tail -f filename  
说明：监视filename文件的尾部内容（默认10行，相当于增加参数 -n 10），刷新显示在屏幕上。退出，按下CTRL+C。

2、tail -n 20 filename  
说明：显示filename最后20行。

3、tail -r -n 10 filename  
说明：逆序显示filename最后10行。

补充：  
跟tail功能相似的命令还有：  
cat 从第一行開始显示档案内容。  
tac 从最后一行開始显示档案内容。  
more 分页显示档案内容。  
less 与 more 相似，但支持向前翻页  
head 仅仅显示前面几行  
tail 仅仅显示后面几行  
n 带行号显示档案内容

grep -n "2018-11-16" jolly-service-risk-busi.log

mysqldump  -h 10.142.170.177  -uusertag -pusertag##3  userprofile  表名 > 导出的文件名

grep

# [Linux后台运行命令，nohup和&的区别](https://www.cnblogs.com/chenkeyu/p/8544289.html)

&的意思是在后台运行， 什么意思呢？  意思是说， 当你在执行 ./a.out & 的时候， 即使你用ctrl C,  那么a.out照样运行（因为对SIGINT信号免疫）。 但是要注意， 如果你直接关掉shell后， 那么， a.out进程同样消失。 可见， &的后台并不硬（因为对SIGHUP信号不免疫）。

      nohup的意思是忽略SIGHUP信号， 所以当运行nohup ./a.out的时候， 关闭shell, 那么a.out进程还是存在的（对SIGHUP信号免疫）。 但是， 要注意， 如果你直接在shell中用Ctrl C, 那么, a.out进程也是会消失的（因为对SIGINT信号不免疫）

      所以， &和nohup没有半毛钱的关系， 要让进程真正不受shell中Ctrl C和shell关闭的影响， 那该怎么办呢？ 那就用nohua ./a.out &吧， 两全其美。

       如果你懂守护进程， 那么nohup ./a.out &颇有点让a.out成为守护进程的感觉。

查看软件安装位置

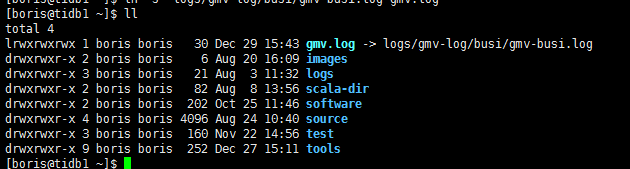
whereis redis

3.如果是rpm包安装的话，可以使用rpm -qal查询一下

rpm -qf /路径/程序名 可以查到一个程序属于哪个rpm包，然后rpm -ql那个包可以看到那个包的所有文件和路径等等。还可以解开成一个cpio文件等等之类的，仔细研究一下rpm的参数吧。

linux中的l类型文件，类似windows中的快捷方式。可以通过命令：ln -s 源文件 新文件。例如：

ln -s logs/gmv-log/busi/gmv-busi.log gmv.log



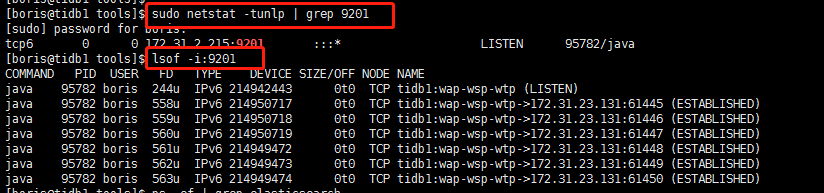
下载文件的命令很多 curl、prozilla、myget、axel

wget url

查看端口占用情况：

sudo netstat -tunlp |grep 端口号

lsof -i:端口号



后台启动nohup &使用

nohup ./tools/elasticsearch-6.4.2-cp1/bin/elasticsearch &