ls -a 所有文件

ls -l 详细属性

more一页一页显示，更多参数

tail -f 查阅正在改变的文件

uptime 时间 系统运行时长 负载

uname -a 操作系统信息

locate file 在数据库内找，比find快

ctrl+c 中断，无后台

ctrl+z 暂停，后台挂起

tar -T 范本文件，让tar解开的文件

-v或--verbose 显示指令执行过程

-c或--create 建立新的备份文件

-t或--list 列出备份文件的内容

-w或--interactive 遭遇问题时先询问用户

-x或--extract或--get 从备份文件中还原文件

-f<备份文件>或--file=<备份文件> 指定备份文件

-z或--gzip或--ungzip 通过gzip指令处理备份文件

-k或--keep-old-files 解开备份文件时，不覆盖已有的文件

Whois 在数据库找信息

Dig 域名查询

Dig [@server] name type ,server 是域名服务器的名称或 IP 地址, type 指示所需的查询类型 ANY、A、MX、SIG 等

Dig -x 逆向域名查询

Extra commands

PATH - colon separated list of pathnames to search for commands

GCC 的编译过程可以划分为四个阶段：预处理（Pre-Processing）、编译（Compiling）、汇编（Assembling）以及链接（Linking）

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | .c | C 源文件 | .s/.S | 汇编语言源文件 | | .C/.cc/.cxx/.cpp | C++ 源文件 | .o/.obj | 目标文件 | | .h | C/C++ 头文件 | .a/.lib | 静态库 | | .i/.ii | 经过预处理的 C/C++ 文件 | .so/.dll | 动态库 | |

-pipe ：使用管道而不是中间文件

-E ：仅执行预处理（不要编译、汇编或链接）。

-S ：只编译（不汇编或链接）。

-c ：编译和汇编，但不链接。

-o <file> ：指定输出文件。

-g 编译时额外：1. 创建调试符号表，符号表包含了程序中使用的变量名称的列表。

2. 关闭所有的优化机制，以便程序执行过程中严格按照原来的C代码进行

如果没有可执行文件中的调试信息，你就无法在gdb中看到代码。在编译的时候，生成调试信息，该程序可以被调试器调试

/

编译 hello.c，默认输出 a.out

gcc hello.c

/

编译 hello.c 并指定输出文件为 hello

gcc hello.c -o hello

/

只执行预处理，输出 hello.i 源文件

gcc -E hello.c -o hello.i

/

只执行预处理和编译，输出 hello.s 汇编文件

gcc -S hello.c

/

也可以由 hello.i 文件生成 hello.s 汇编文件

gcc -S hello.i -o hello.s

/

只执行预处理、编译和汇编，输出 hello.o 目标文件

gcc -c hello.c

/

也可以由 hello.i 或 hello.s 生成目标文件 hello.o

gcc -c hello.i -o hello.o

gcc -c hello.s -o hello.o

/

由 hello.o 目标文件链接成可执行文件 hello

gcc hello.o -o hello

/

查看所有文件描述

$ file \*

/

查看 hello 文件描述

$ file hello

/

把f1上传到svn上步骤，Svn add f1; svn commit f1; svn revert f1

Revert是回溯到原版本

显示的命令放入到xxx.txt 文件中 ls > xxx.txt

显示的命令追加到xxx.txt中 ls>>xxx.txt

清屏 clear

cd . 当前路径

cd .. 上级目录

cd ../.. 上级目录的上级目录

cd - 回到上次所在路径

cd~ 回根目录

grep "ntfs" xxx.txt

grep -n "ntfs" xxx.txt 在哪一行

grep -v "ntfs" xxx.txt 不包括ntfs的行

grep "^ntfs" xxx.txt 以ntfs开头的

grep "ntfs$" xxx.txt 以ntfs结尾的

mv 111.txt laowang/ 移动到laowang/文件夹中

find /dir -name "name"

sudo find /dir -name "name" 获取权限

sudo 是允许系统管理员让普通用户执行一些或者全部的root命令的一个工具

关机 shutdown -h

重启 reboot

同时打开两个终端 Ctrl+Shift +t

su shuaige 切换到帅哥账号

sudo passwd shuaige 设置帅哥的密码

extra2

Svn commands:

svn commit -m "LogMessage" xxx.txt :Committing changed files to the repository

Svn help: all svn commands

Svn checkout path: Retrieve files locally

Svn update -r m path:

If no directory or file follows svn update, the files in the directory and subdirectories are updated to the latest version by default.

-r: Specifies which version to update to. Example: svn up -r 200 123.txt (reverts the local 123.txt file to version 200)

Svn log x.txt: Used to display the modification history of this file

svn diff -r m:n xxx.txt

Compare the differences for version m and version n.

-r: The two version numbers to compare. Example: svn di -r 2:3 123.txt (to see the difference between version 2 and version 3 of 123.txt).

svn delete path -m "DeleteMessage"

Remote delete: svn delete svn://192.168.1.1/project\_v1/123.php -m "delete test file" (using this way will delete the file in the repository directly after deletion).

Commit delete: first svn delete 123.php, then svn ci -m "delete file" (this is the recommended way to use).

Tlb = Translation lookaside buffer

export -p: List all environment variables given to the program by the shell

export x=7: Define environment variables and assign values

echo: The function of the echo command is to write content to standard output.

for loop

for i in {2,4,6,8};do echo $i;done Output: 2

4

6

8

for i in {1..5};do echo $i;done Output: 1

2

3

4

5

for i in user{1,4,6};do echo $i;done Output: user1

user4

user6

for i in {2,4,20};do echo $(($i\*3));done Output: 6

12

60

/

sum=0

for i in {1..10};do let sum+=$i;done

echo "$sum"

Output: 55

/

[root@localhost sed\_test]# for i in `ls`;do echo $i;done

linshi.txt

passwd\_bak.txt

sort\_test.txt

[root@localhost sed\_test]# for i in \*.txt;do echo $i;done

linshi.txt

passwd\_bak.txt

sort\_test.txt

[root@localhost sed\_test]# for i in 2 3 6 8;do echo $i;done

2

3

6

8

echo $$ Returns the PID of the login shell

echo $? Returns the status of the previous command, 0 means no error, any other value indicates an error

echo $\* displays all arguments passed to the script as a single string, unlike position variables, this option can have more than 9 arguments

echo $! Returns the process ID number of the last process running in the background

echo $\_ is the last argument to save the previously executed command

cut command cuts bytes, characters and fields from each line of the file and writes these bytes, characters and fields to standard output

cut -f2 -d\| file: cut each line by `|` and get the second column

cut -b : Segmentation in bytes These byte positions will ignore multi-byte character boundaries unless the -n flag is also specified

sort command is used to sort the contents of a text file

sort is sorted in ascending order by default

sort [-bcdfimMnr][-o<output file>][-t<separating characters>][+<start field>-<end field>][--help][--verison][file][-k field1[,field2]]

-d Sort by alphabet, numbers and space characters, ignoring all other characters.

-M Sort the first 3 letters according to the abbreviation of the month.

-n Sort by the value of number.

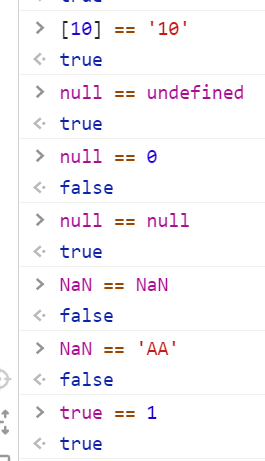
-u means unique and the output is de-duplicated.

-r reverse order

sort -r number.txt -o number.txt : Write the sorted result to the original file number.txt

cat [-AbeEnstTuv] [--help] [--version] fileName

-n Number all output lines starting from 1



chmod

u user

g group

o others

a all

+ add permission

* Removal of permissions

= assign permissions

内存泄露（memory leak）

是指程序在申请内存后，无法释放已申请的内存空间，

内存溢出（out of memory）

是指程序在申请内存时，没有足够的内存空间供其使用。

buffer overflow

用超出常规长度的字符数来填满一个域

超过的那部分字符将会覆盖与数组相邻的其他变量的空间，使变量出现不可预料的值

在bash中，通常使用3个整数来表示标准输入（0）、标准输出（1）和标准错误（2）。如果要把标准错误输出到文件中，可以使用

cat No\_exist.txt 2> tt.txt

这时在tt.txt文件中就会出现标准错误cat: No\_exist.txt: No such file or directory。同样的道理，我们可以将标准错误重导向输出为标准输出，2>&1比如

$ cat No\_exist.txt

cat: No\_exist.txt: No such file or directory

$ cat No\_exist.txt 2>&1

cat: No\_exist.txt: No such file or directory

虽然它们在终端上输出的内容看起来没有什么区别，但是它们的身份是不一样的，第一个是以标准错误的形式输出的，而第二个是标准输出。