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 checkout path: Retrieve files locally

Svn update -r m path:

bring changes from svn into the working directory

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

$0 脚本启动名(包括路径)

$n 第n个参数,n=1,2,…9

$\* 所有参数列表(不包括脚本本身)

$@ 所有参数列表(独立字符串)

$# 参数个数(不包括脚本本身)

$$ 当前程式的PID

$! 执行上一个指令的PID

$? 执行上一个指令的返回值

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

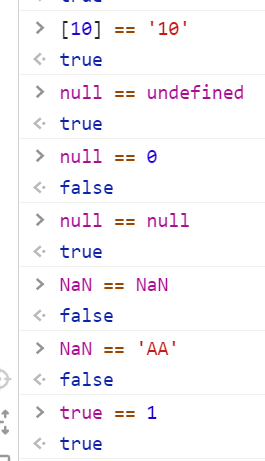
4/18/2023

查看后台进程作业ID的指令 jobs

Linux执行ls，会引起哪些系统调用 read

execve

fork



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。

4/16

在linux中，httpd.conf :网络服务器软件的配置文件,对WWW服务器进行访问、控制存取和运行等控制 。lilo.conf :由引导管理程序 lilo 读取的文件 inetd.conf :保存了系统提供internet服务的数据库并对其进行控制，如打开/关闭某项服务。 resolv.conf:域名解析器使用的配置文件

RHEL5系统安装DHCP服务器软件包后，系统中存在配置文件“/etc/dhcpd.conf”

touch /etc/nolgoin： 禁止所有普通用户登录

init3：完全的多用户状态（有NFS），登录后进入控制台命令行模式；

iptable -s INPUT -t tcp -s !127.0.0.1 -j DROP:屏蔽非127.0.0.1请求；

init6：系统正常关闭并重启，默认运行级别不能设为6，否则不能正常启动；

硬连接不能链接目录文件

链接分为硬链接和符号链接

硬链接就是让链接文件的i节点号指向被链接文件的i节点

Linux世界里的两句流行语：

1.一切皆文件：Linux中所有内容都是以文件的形式保存和管理；

2.有问题找男人：可以使用man来查看命令帮助手册，如man pwd

subnet 166.173.197.131 netmask 255.255.255.192{

range 166.173.197.128 166.173.197.191;

default-lease-time 600;

max-lease-time 7200;

}

对子网166.17\*\*\*7.131（其子网掩码为255.255.255.192，即网络号占高26位，主机号占低6位），该子网的网络地址为：166.17\*\*\*7.131，其主机范围为166.17\*\*\*7.128-166.17\*\*\*7.191.

现在对该可选择范围内申请主机号范围在166.17\*\*\*7.10-166.17\*\*\*7.107的IP地址；

该申请默认租用时间为600s，最大租用时间不超过7200秒

线程是程序的多个顺序的流动态执行

线程不能够独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制

线程是共享进程的地址空间

进程是 操作系统进行资源分配和调度的一个独立单位，而线程只是CPU调度和分派的基本单位

[ -z "" ] 命令判断双引号中变量的字符串长度是否为0，

[ -z "" ] && echo 0 || echo 1

^ ^ ^

命令1 命令2 命令3

本题即： 先判断 “” 变量中是否无值，成功

然后根据 && 符特性，当命令1执行成功时执行命令2，也就是执行echo 0

然后根据 || 符特性，前一个命令不能执行才执行后一个，命令2 echo 0

执行成功，所以命令3 echo 1 不执行

命令结束

新建一个管理员用户 admin，需要使用的参数

useradd -g -0 admin

-u 用户号 指定用户的用户号；因为系统用户的用户号为 0，故指定用户号为 0；

如果同时有 -o 选项，则可以重复使用其他用户的标识号，因为系统本身存在用户号为 0 的系统用户，故应该使用该参数。

查看某IP是否可达

telnet

ping

tracert

Linux 系统上同一个程序的多个进程实例共享一个 TCP 监听端口

每个进程都使用 SO\_REUSEPORT 选项,然后绑定同一个地址和端口

每个进程分别绑定不同的网卡地址的同一端口

第一个进程先绑定到监听地址端口, 然后 fork 子进程共享使用

chmod a 代表所有用户

存储用于创建用户目录的系统用户默认文件

/etc/skel

删除非空目录dirs的方法

rm -d dirs.

rm -r dirs.

! 为非运算

-a （and）表示并且，与运算

-o （or）表示或，或运算

top 获取本机cpu使用率

vi 编辑器强制存盘退出的指令(末行模式)

:q 不保存退出

:q! 直接强制退出，不写入修改的数据

:wq 强制性写入文件并退出。即使文件没有被修改也强制写入，并更新文件的修改时间（命令模式 ZZ）

:wq! 强制写入并退出（只有当操作者是文件所有者才可行）

uptime命令主要用于获取主机运行时间和查询linux系统负载等信息

vmstat命令可以查看查看cpu负载

线程同步的方式包括：互斥锁（或称互斥量Mutex)，读写锁(rdlock)，条件变量(cond)，信号量(Semophore)

临界区只是用来对共享资源的保护，临界区在任意时刻只允许一个线程对共享资源进行访问。如果有多个线程试图同时访问临界区，那么有线程进入后，其他线程试图访问时将被挂起，直到进入临界区的线程离开，临界区并不能保证线程的执行顺序，先到先得。

使用 dpkg 命令安装的软件为 .deb

Linux进程间通讯

UNIX套接字

信号量

无名管道

文件锁

共享内存

unix进程间的通信方式 （1）管道（Pipe）：管道可用于具有亲缘关系进程间的通信，允许一个进程和另一个与它有共同祖先的进程之间进行通信。 （2）命名管道（named pipe）：命名管道克服了管道没有名字的限制，因此，除具有管道所具有的功能外，它还允许无亲缘关系进程间的通信。命名管道在文件系统中有对应的文件名。命名管道通过命令mkfifo或系统调用mkfifo来创建。 （3）信号（Signal）：信号是比较复杂的通信方式，用于通知接受进程有某种事件发生，除了用于进程间通信外，进程还可以发送信号给进程本身；linux除了支持Unix早期信号语义函数sigal外，还支持语义符合Posix.1标准的信号函数sigaction（实际上，该函数是基于BSD的，BSD为了实现可靠信号机制，又能够统一对外接口，用sigaction函数重新实现了signal函数）。 （4）消息（Message）队列：消息队列是消息的链接表，包括Posix消息队列system V消息队列。有足够权限的进程可以向队列中添加消息，被赋予读权限的进程则可以读走队列中的消息。消息队列克服了信号承载信息量少，管道只能承载无格式字节流以及缓冲区大小受限等缺 （5）共享内存：使得多个进程可以访问同一块内存空间，是最快的可用IPC形式。是针对其他通信机制运行效率较低而设计的。往往与其它通信机制，如信号量结合使用，来达到进程间的同步及互斥。 （6）内存映射（mapped memory）：内存映射允许任何多个进程间通信，每一个使用该机制的进程通过把一个共享的文件映射到自己的进程地址空间来实现它。 （7）信号量（semaphore）：主要作为进程间以及同一进程不同线程之间的同步手段。 （8）套接口（Socket）：更为一般的进程间通信机制，可用于不同机器之间的进程间通信。起初是由Unix系统的BSD分支开发出来的，但现在一般可以移植到其它类Unix系统上：Linux和System V的变种都支持套接字。

chown（change owner）命令用于设置文件所有者和文件关联组的命令；

chgrp（change group）命令用于变更文件或目录的所属群组

linux的系统调用是指 是用户进程调用内核功能的接口

将文件xyz中的单词AAA全部替换为BBB

sed 's/AAA/BBB/g' xyz > xyz

:s/old/new 将当前行中查找到的第一个字符 “old” 串替换为 “new”

:s/old/new/g 将当前行中查找到的所有字符串 “old” 替换为 “new”

:#,#s/old/new/g 在行号 “#,#” 范围内替换所有的字符串 “old” 为 “new”

:%s/old/new/g 在整个文件范围内替换所有的字符串“old”为“new”

/etc/hosts 包含了主机名到IP地址的映射关系

netstat 查看TCP连接的命令

.bash\_logout：退出shell时，要执行的命令

.bash\_profile：每个用户都可使用该文件输入专用于自己使用的shell信息，当用户登录时，该文件仅仅执行一次!默认情况下

/etc/profile：此文件为系统的每个用户设置环境信息，当用户第一次登录时，该文件被执行

ext3增加了日志功能，即使在非正常关机后，系统也不需要检查文件系统

ext3文件系统能够极大地提高文件系统的完整性，避免了意外宕机对文件系统的破坏

ext2/ext3均不包含文件创建时间

arp –d 查看ARP缓存记录

用户磁盘配额配置文件 aquota.user 的默认访问权限是 600

硬链接 不可以跨文件系统 链接文件的i节点与被链接文件的i节点相同

4/17

| 是管道符，用于将两个命令隔开，管道符左边命令的输出会作为管道符右边命令的输入

> 是输出重定向，将左边内容重定向到右边指定的文件中

存在文本文件 /tmp/testfile，并且文件行数小于等于 10 行，则输出文件前 3 行

if [[ -f '/tmp/testfile' ]] && [[ `wc -l /tmp/testfile` -le 10 ]];then

head -3 /tmp/testfile

fi

虚拟内存是计算机系统内存管理的一种技术。它使得应用程序认为它拥有连续可用的内存，使得每个进程看到的内存空间一致。

crontab 命令用来定期执行程序。

时间配置段为 5 部分：f1 f2 f3 f4 f5，其中 f1 是表示分钟（0-59），f2 表示小时（0-23），f3 表示一个月份中的第几日（1-31），f4 表示月份（1-12），f5 表示一个星期中的第几天（0-6）。

当 f1 为 \* 时表示每分钟都要执行，f2 为 \* 时表示每小时都要执行程序，其余类推；

当 f1 为 a-b 时表示从第 a 分钟到第 b 分钟这段时间内要执行，f2 为 a-b 时表示从第 a 到第 b 小时都要执行，其余类推；

当 f1 为 \*/n 时表示每 n 分钟个时间间隔执行一次，f2 为 \*/n 表示每 n 小时个时间间隔执行一次，其余类推；

当 f1 为 a, b, c,... 时表示第 a, b, c,... 分钟要执行，f2 为 a, b, c,... 时表示第 a, b, c...个小时要执行，其余类推。

题中 \*/30 \* 3 \* \* 表示：每月的 3 号，每 30 分钟执行命令。

06 03 \* \* 3 表示每周 3 的 03 点 06 分执行

僵尸进程:死了没人管 孤儿进程:让init这一家给领养走了

孤儿进程：一个父进程退出，而它的一个或多个子进程还在运行，那么那些子进程将成为孤儿进程。

僵尸进程：一个进程使用fork创建子进程，如果子进程退出，而父进程并没有调用wait或waitpid获取子进程的状态信息，那么子进程的进程描述符仍然保存在系统中。这种进程称之为僵死进程。

/etc，etc 是 Etcetera（等等）的缩写，这个目录用来存放所有的系统管理所需要的配置文件和子目录；

/var，var 是 variable(变量) 的缩写，这个目录中存放着在不断扩充着的东西，我们习惯将那些经常被修改的目录放在这个目录下，包括各种日志文件；

/root，该目录为系统管理员，也称作超级权限者的用户主目录；

/bin，bin 是 Binaries（二进制文件）的缩写，这个目录存放着最经常使用的命令。

/dev，dev 是 Device（设备）的缩写，该目录下存放的是 Linux 的外部设备，在 Linux 中访问设备的方式和访问文件的方式是相同的

/lib，lib 是 Library（库） 的缩写这个目录里存放着系统最基本的动态连接共享库，其作用类似于 Windows 里的 DLL 文件。几乎所有的应用程序都需要用到这些共享库

图示

描述已自动生成

局域网的网络地址192.168.1.0/24，局域网络连接其它网络的网关地址是192.168.1.1。主机192.168.1.20访问172.16.1.0/24网络时，其路由设置正确的是

route 命令用于显示和操作IP路由表；

tracert 为 Windows 路由跟踪实用程序，可以用于确定 IP 数据包访问目标时所选择的路径；

ping 命令用于检测主机；

netstat 命令用于显示网络状态，利用 netstat 指令可以得知整个 Linux 系统的网络情况。 add –net 172.16.1.0 gw 192.168.1.1 netmask 255.255.255.0 metric 1

makefile文件保存了编译器和连接器的参数选项

主要包含了五个东西：显式规则、隐晦规则、变量定义、文件指示和注释

默认的情况下，make命令会在当前目录下按顺序找寻文件名为“GNUmakefile”、“makefile”、“Makefile”的文件， 找到了解释这个文件

linux防火墙iptabls拒绝所有客户端ping数据包的规则

iptables -A INPUT -s ! 127.0.0.1 -p icmp -j DROP

iptables -A INPUT -s 0.0.0.0 -p icmp -j DROP

内核包括的子系统

进程管理系统

内存管理系统

I/O管理系统

hosts文件包含了ip地址与主机名之间的映射，在没有域名解析服务器的情况下，系统上的所有网络程序都通过查询该文件来解析对应于某个主机名的ip地址

vi编辑器中，删除一行的命令是 dd

vi编辑 自下而上查找字符串“2006” ?2006

dns服务主配置文件 named.conf

产生死锁的四个必要条件：

（1） 互斥条件：一个资源每次只能被一个进程使用。

（2） 请求与保持条件：一个进程因请求资源而阻塞时，对已获得的资源保持不放。

（3） 不剥夺条件:进程已获得的资源，在使用完之前，不能强行剥夺。

（4） 循环等待条件:若干进程之间形成一种头尾相接的循环等待资源关系。

useradd 命令用于建立用户帐号；

usermod 命令用于修改用户帐号；

groupadd 命令用于创建一个新的工作组，新工作组的信息将被添加到系统文件中；

userdel 命令用于删除用户帐号。

init启动进程需要读取 /etc/inittab

当一个设置了SUID 位的可执行文件被执行时，无论谁来执行这个文件，他都暂时拥有文件所有者的权限

可以通过chmod u+s 命令设置SUID属性

passwd命令就是通过SUID机制，在任何用户修改密码时，有权限访问密码存储文件

ar 将.o文件打包成静态库文件（.a）

文件系统的文件数据读写性能与文件大小以及读写数据块的大小有关

应用程序可以用内存映射的方式访问文件中的数据

arp协议 地址解析协议

logcat -v time 打印log的同时打印系统时间

logcat -c 清除已有log

dd 命令用于读取、转换并输出数据，dd 可从标准输入或文件中读取数据，根据指定的格式来转换数据，再输出到文件、设备或标准输出；

fmt 命令用于编排文本文件，fmt 指令会从指定的文件里读取内容，将其依照指定格式重新编排后，输出到标准输出设备。若指定的文件名为 "-"，则 fmt 指令会从标准输入设备读取数据；

cut 从文本文件的每一行中截取指定内容的数据。

将当前目录下的归档文件 myftp.tgz 解压缩到 /tmp 目录下

tar xvzf myftp.tgz -C /tmp

-x：extract files from an archive 即从归档文件中释放文件；

-v：verbosely list files processed 即详细列出要处理的文件；

-z：filter the archive through gzip 即通过gzip解压文件；

-f：use archive file or device ARCHIVE 即使用档案文件或设备；

通常情况下解压 .tar.gz 和 .tgz 等格式的归档文件就可以直接使用 tar xvzf；

因为要解压到指定目录下，所以还应在待解压文件名后加上 -C（change to directory）参数

统计一个文件中"牛客"出现的行数

grep "牛客" 文件名 | wc -l

记录用户操作的历史记录

/home/.bash\_history

/etc/shadow 存放用户的密码密文

linux系统中的/var/log/lastlog日志记录了所有用户最后登录的记录，可以用lastlog命令查看

/var/run/utmp日志记录了正在登录本系统中的用户信息，可以用`last -f`命令查看

子进程保留了父进程的 环境变量 当前工作目录

ifconfig命令作用范围

配置本地回环地址

配置网卡的IP地址

激活网络适配器

Linux系统大致启动流程是 uEFI-> MBR -> grub -> initrd

后台运行可以有两种方式： 1.命令前加nohup # nohup \*.sh 2.命令后加& # \*.sh &

1 用户命令 5 文件格式描述 , 例如/etc/passwd n 新文档， 可能要移到更适合的领域。

o 老文档， 可能会在一段期限内保留。

l 本地文档， 与本特定系统有关的。

man 5 passwd 显示passwd文件的结构

dd命令 : 作用是用指定大小的块拷贝一个文件，并在拷贝的同时进行指定的转换。 参数：if 输入文件（或设备名称） of 输出文件（或设备名称） 将文件file1拷贝到文件file2中命令如下： dd if=file1 of=file2

Linux下通过哪个命令怎么查看中断 cat /proc/interrupts

如果cmd1成功执行，则执行cmd2命令 cmd1 && cmd2

vim

dd:删除当前行（0：返回行首）

ndd:删除光标行往下n行(含光标行)内容

dgg:删除光标当前行及以上内容

dG:删除光标当前行及以下内容

dH:删除当前页面第1行至光标行

p:粘贴到光标下一行

u:撤销一次操作

Ctrl + r:反撤销（一次）

linux内核锁

Semaphore

Spinlock

seqlock

head -n k # 打印前k行

head -n -k # 打印除最后k行外的所有内容

tail -n k # 打印最后k行

tail -n +k # 从k行开始打印

route 命令用于显示和操作IP路由表；

tracert 为 Windows 路由跟踪实用程序，可以用于确定 IP 数据包访问目标时所选择的路径；

ping 命令用于检测主机；

# 超级用户提示符，$ 是普通用户的提示符

设 umask 为 002，则新建立的文件默认权限是什么

假设原始权限为 mode，则最终建立文件的权限为：mode & ~umask。文件默认没有可执行权限。如果要计算权限，则需要将权限转换成二进制数据，然后进行上述运算，上述运算其实就是去掉某些权限。

文件的最大权限为 777，转换成字符串形式即 rwxrwxrwx，umask 为 002 代表的权限字符串形式为 -------w-

按照运算规则，即从 rwxrwxrwx 中去掉 -------w- 已有的权限，结果为 rwxrwxr-x，这就是目录创建的缺省权限；

对于文件来说，还需要去掉可执行权限，即 rw-rw-r--（664）

查找当前目录一个月（30天）以前大于 100M 的日志文件（.log）并删除

find . -name "\*.log" -mtime +30 -type f -size +100M | xargs rm -rf {};

find 命令用于查找文件：

. 在当前目录查找

-name 指定文件名，\*.log 表示后缀名为 .log，\* 是通配符，表示匹配任意字符串

-mtime 指定修改时间（以天为单位），+xx 表示修改时间大于 xx 天， -xx 表示修改时间小于 xx 天

-type 是指定文件类型，b 表示块设备文件，d 表示目录，c 表示字符设备文件，p 表示管道文件，l 表示符号链接文件，f 表示普通文件

-size 指定文件大小，+xx 表示文件大小大于 xx，-xx 表示文件大小小于 xx

xargs 把前一命令输入当作后一命令输出，通常配合管道使用。

交换分区的大小通常是 内存\*1.5或者2

子进程独有：进程号PID，不同的父进程号，自己的文件描述符和目录流的拷贝，不继承父进程的进程正文，数据，和其他锁定内存，不继承异步输出输入。

子进程继承父进程：UID GID，环境，堆栈，共享内存，打开文件的描述符，执行时关闭标志，信号控制设定，进程组号，当前工作目录，根目录，文件方式创建屏蔽字，资源限制，控制终端。

子父进程拥有独立的地址空间和PID。

经过fork()以后，父进程和子进程拥有相同内容的代码段、数据段和用户堆栈，就像父进程把自己克隆了一遍。事实上，父进程只复制了自己的PCB块。而代码段，数据段和用户堆栈内存空间并没有复制一份，而是与子进程共享。只有当子进程在运行中出现写操作时，才会产生中断，并为子进程分配内存空间。由于父进程的PCB和子进程的一样，所以在PCB中断中所记录的父进程占有的资源，也是与子进程共享使用的。

4/18

tracert命令用来探测路由经过

将脚本demo.sh的标准输出和标准错误输出重定向至文件demo.log

bash demo.sh &>demo.log

bash demo.sh >& demo.log

bash demo.sh >demo.log 2>&1

vi编辑文件时，要将某文本文件第1行到5行的内容复制到文件中的指定位置

将光标移到第1行，在vi命令模式下输入5yy,然后将光标移到制定位置，按p键

使用末行命令1，5y，然后将光标移到制定位置，按p键

maxfd 是指定要监视的文件描述符的范围，一般取监视的描述符数的最大值 + 1

关闭linux系统 halt

建立动态路由需要用到的文件有 /etc/gateways

软连接可以跨文件系统进行连接，硬链接不可以

统计日志中有多少个不同的ip登陆

cat catalina.log | awk -F ' ' '{print $3}' | sort | uniq -c | wc -l

cat catalina.log | awk '{print $3}' | sort -k1nr | uniq -c | wc –l

4/19/2023

www 的端口 80

和ftp的端口 21端口用于连接，20端口用于传输数据

vi gg到顶部；G = shift + g到底部

启动交换文件的指令是 swapon

sed命令用于行的新增/删除, 行的替换/显示, 搜寻并替换, 直接修改文件。 -n：使用安静(silent)模式。在一般 sed 的用法中，所有来自 STDIN 的数据一般都会被列出到屏幕上。但如果加上 -n 参数后，则只有经过 sed 特殊处理的那一行才会被列出来。 p：打印，将某个选择的数据印出来。通常 p 会与参数 sed -n 一起用 d:删除 a：新增

用 vi 打开一个文件，如何用字母 “new” 来代替字母 “old”

:1,$s/old/new/g

表示从第一行到最后一行寻找 old 字符串，并将该字符串取代为 new。

1,$ 表示从第一行到文件尾行；

s 表示替换；

/g 表示在全局文件中进行替换，省略时仅对每行第一个匹配串进行替换

vi编辑器环境中的任意时刻，选择“ESC”键后，编辑器将进入（ 命令）模式

select调用时会进行线性遍历,epoll采用回调函数机制,不需要线性遍历

select的最大连接数为FD\_SETSIZE

epoll较适用于有大量并发连接，但活跃连接不多的场景

epoll的效率不随FD数目增加而线性下降

epoll通过共享存储实现内核和用户的数据交互

内存屏障主要有：读屏障、写屏障、通用屏障、优化屏障、几种。

以读屏障为例，它用于保证读操作有序。屏障之前的读操作一定会先于屏障之后的读操作完成，写操作不受影响，同属于屏障的某一侧的读操作也不受影响。类似的，写屏障用于限制写操作。而通用屏障则对读写操作都有作用。优化屏障则用于限制编译器的指令重排，不区分读写。前三种屏障都隐含了优化屏障的功能

fork() 函数复制时将父进程的所以资源都通过复制数据结构进行了复制，然后传递给子进程，所以 fork() 函数不带参数； clone() 函数则是将部分父进程的资源的数据结构进行复制，复制哪些资源是可选择的，这个可以通过参数设定，所以 clone() 函数带参数，没有复制的资源可以通过指针共享给子进程

alias命令用于设置指令的别名

rm -i，-i 参数表示每次删除时交互式询问是否删除，删除前逐一询问确认。

awk把两个文件的合并成一个文件

BSS（Block Started by Symbol）通常是指用来存放程序中未初始化的全局变量和静态变量的一块内存区域。特点是:可读写的，在程序执行之前BSS段会自动清0。所以，未初始的全局变量在程序执行之前已经成0了。 数据段：数据段（data segment）通常是指用来存放程序中已初始化的全局变量的一块内存区域。数据段属于静态内存分配。 代码段：代码段（code segment/text segment）通常是指用来存放程序执行代码的一块内存区域。这部分区域的大小在程序运行前就已经确定，并且内存区域通常属于只读, 某些架构也允许代码段为可写，即允许修改程序。在代码段中，也有可能包含一些只读的常数变量，例如字符串常量等。 堆（heap）：堆是用于存放进程运行中被动态分配的内存段，它的大小并不固定，可动态扩张或缩减。当进程调用malloc等函数分配内存时，新分配的内存就被动态添加到堆上（堆被扩张）；当利用free等函数释放内存时，被释放的内存从堆中被剔除（堆被缩减） 栈(stack)：栈又称堆栈， 是用户存放程序临时创建的局部变量，也就是说我们函数括弧“{}”中定义的变量（但不包括static声明的变量，static意味着在数据段中存放变量）。除此以外，在函数被调用时，其参数也会被压入发起调用的进程栈中，并且待到调用结束后，函数的返回值也会被存放回栈中。由于栈的先进后出特点，所以栈特别方便用来保存/恢复调用现场。从这个意义上讲，我们可以把堆栈看成一个寄存、交换临时数据的内存区

Linux2.6 中，默认情况下一个进程最多能打开1024个文件