## 系统相关

系统运行级别  
管道、通配符、变量取值、重定向、用户目录、命令换行（\）、单引号（常量）、反引号（命令行）、

var: various，系统一般运行时要改变的数据，大小经常变化。eg./var/log、/var/mail

usr: unix software resource，所有的程序都安装在这儿

home： 用户文件夹

dev： device

etc: environment configuration 全局设置

lib: library，开机会用到的函数库

media: 可删除的设备

mnt: mount,临时安装点

proc: process，伪文件系统，实际不存在的目录

sys: system，虚拟文件系统，不占硬盘容量

bin: binary，二进制可执行命令

boot： 开机会使用到的文件

opt: optional，第三方软件放置的目录

srv: service

## Shell 脚本

Server\_name=${5:-standalone} 判断第五个参数是否赋值，未赋值则默认为standalone

PackageN=`basename $package\_Name`

FileName=$(echo $PackageN | cut -d "." -f1) 注意变量赋值格式，两种效果一样

数值计算建议格式Var=$((计算式)) ，也可以declare –i var=计算式

判断式[ -e filename ]判断文件是否存在；-f存在且为文件；-d存在且为目录；-z字符串是否**为空**；-n字符串是否**非空；**

unzip -q -o $PackageN -d $FileName 解压缩且为安静模式

zip -rq $ZipName.zip un\_new\_"$WarName"/\* 生成zip包

nohup ./standalone.sh -c standalone-ha.xml & 后台运行

MD5=` md5sum $file ` 根据文件内容生成唯一的MD5码

sed 's/:\ /\//' delete.txt >> del\_list.txt 删除每一行“:/”

遍历处理目录下所有文件

for file in "$DesPath1"/\*

do

MD5=` md5sum $file `

echo $MD5 >> new\_"$WarName".md5

done

while read line

do

MD1=$(echo $line | awk '{print $1}')

done < new\_"$WarName".md5 从文件输入，逐行处理

$0~ "'$path1'" {print $0}

Grep 参数-I 忽略大小写；-v 反向选择

脚本不能用tab键对齐，涉及编码问题

Shell 脚本执行时 –x参数 将使用到的内容显示在屏幕上

## 常用命令：

Du //查看目录的总容量以及子目录的容量

mount /dev/cdrom /mnt/cdrom //挂载光驱到 /mnt/cdro m目录下

umount /mnt/cdrom //卸载光驱

file 文件名 //查看文件类型

find 目录名 –name “\*.c” //在某目录下查找文件为\*.c的文件

-type f //在目录下查找所有文件

chmod u+x g+r o-w 文件名 //为文件设置权限（用户，组，其他人）

chmod 744 文件名 //为文件设置权限（1-执行，2-写，4-读）

chown 用户 文件名 //把文件改为用户所有

ln 文件名 链接名 //为文件创建硬链接 相当于别名，不占空间

ln -s 文件名 链接名 //为文件创建软链接 相当于快捷方式，占空间

rpm –ivh 安装包名 //安装软件

rpm -e 软件名 //卸载软件

tar –jcv –f filename.tar.bz2

tar –zcv –f filename.tar.gzip

tar –jxv –f file.tar.bz2 –C directory 解压缩

init 0 //关机

init 6 //重启

ps –aux //查看进程

top //查看系统信息，类似任务管理器

kill 进程号 //中止某进程

kill -9 进程号 //按进程号强行中止某进程

jobs //查看后台程序运行

fg job号 //把后台命令调回前台

bg job号 //把命令调回后台

查看端口占用

netstat -apn 最后一栏是PID/PROGRAM NAME

ps -aux|grep pid

**standard input:<和<<**

<将原本应该由键盘输入的数据改由文件内容来输入

<<结束输入的意思，比如<<“stop”，则在输入stop之后自动终止输入

**重定向：>和>>**

>已存在则覆盖

>>文件最后添加

Tr 删除文件中的控制字符，或进行字符转换,

$ echo aaacccddd | tr -s [abc] 去重

Acddd

$ cat test.txt | tr -s ["\n"] 删除多余换行符

$ echo "Hello World I Love You" |tr [-t] [a-z] [A-Z] 所有小写字母转换成大写字母

$ echo "Hello World I Love You" |tr [:lower:] [:upper:] 亦可

$ cat test.txt | tr -d "[0-9][: ]" 删除所有的数字和空格冒号

$ cat test.txt|tr -cs "[a-z][A-Z]" "\n" 用换行符替换除字母之外所有字符，然后删除多余换行符

|  |
| --- |
| Cut:  Cut –d ‘分隔字符’ –f fields  Eg. echo $PATH|cut –d ‘:’ –f 3,5  Cut –c 字符范围；用于比较整齐的格式  Eg. export –c 12-  Grep:  Grep –c 计算找到目标字符串的次数  Grep –v 反向选择，即显示没有目标字符串的行  Grep –n 输出行号同时  Grep –i 忽略大小写 ignore  Grep --color=auto  Sort:  Sort –k field以某个字段排序  Sort –t ‘分隔符’ 默认Tab键  Sort –r 反向排序  Uniq:  Uniq –i 忽略大小写  Uniq –c 统计次数  通常配合sort，先排序再分组；因为单独的uniq 只能合并连续相同的关键字  Wc:  Wc –l 统计行 line  Wc –w 统计单词 word  Wc –m 统计字符  无参数时默认相当于-lwm  Sed:  Sed ‘s/old/new/’替换,注意**单引号，支持正则表达式**  Sed ‘2,5d’ 删除2到5行  Sed ‘2,5p’ 打印2到5行  Sed ‘2,5c’ ‘something’ 2到5行替换成something  Sed 后面如果要接超过2个以上的动作时，每个动作前面加-e参数  Awk:  **Awk ‘条件类型1{动作1} 条件类型2{动作2}……’ filename**  NF每一行的字段总数  NR 目前处理的是第几行  FS 目前分隔符，默认是空格或tab  Awk 中变量可以直接使用，不需要加$  Awk后续所有动作都是以***单引号***括住的  Awk ‘BEGIN {FS=”:”} $3<10 {print $1 “\t” $3}’  Eg.复杂例子  Awk ‘{if(NR==1) printf “%10s %10s %10s %10s,%10s\n”,$1,$2,$3,$4,”Total”}  NR>=2{total=$2+$3+$4  Printf “%10s %10d %10d %10d %10.2f\n”,$1,$2,$3,$4,total}’  Diff:  以行为单位比较文件差异  Diff –b 忽略一行当中仅有多个空白的区别  Diff –B 忽略空白行的区别  Diff –i 忽略大小写不同  Test:  -e 文件名是否存在  -f 文件名是否存在且为文件  -d 文件名是否存在且为目录  数值：  -eq 数值是否相等  -ne 数值是否不等  -gt n1大于n2  -lt n1小于n2  -ge 大于等于  -le 小于等于  字符串：  -z 字符串是否为空  -n 字符串是否非空  Str1=str2  Str1!=str2 |

## 编译环境

Gcc:编译C程序

G++：编译CPP程序

Make:根据预定义的makefile文件宏编译

### 关于makefile

一、Makefile的规则

target ... : prerequisites ...

command

...

target也就是一个目标文件，可以是Object File，也可以是执行文件。还可以是一个标签（Label），对于标签这种特性，在后续的“伪目标”章节中会有叙述。

prerequisites就是，要生成那个target所需要的文件或是目标。

command也就是make需要执行的命令。（任意的Shell命令）

这是一个文件的依赖关系，也就是说，target这一个或多个的目标文件依赖于prerequisites中的文件，其生成规则定义在command中。说白一点就是说，**prerequisites中如果有一个以上的文件比target文件要新的话，command所定义的命令就会被执行**。这就是Makefile的规则。也就是Makefile中最核心的内容。