回顾：

1）循环结构

—— for列表式循环

for 变量名 in 值1 值2 .. ..

do

针对 $变量名 处理操作

done

for ((i=1;i<=10;i++))

do

。。 。。

done

—— while条件式循环 【until】

while 满足XX条件时

do

处理操作命令...

done

i=1

while 变量i的值满足XX条件时

do

处理操作命令...

let i++

done

2）case分支 —— 在变量值等于不同的情况时做不同操作

case 变量值 in

值1)

命令操作1 .. ..

;;

值2)

命令操作2 .. ..

;;

\*)

默认的命令操作

esac

3）Shell函数 —— 就是使用固定的名称来存放一批命令行

变量、命令别名、脚本、函数/内部命令

函数名() {

命令操作1

命令操作2 $1

.. ..

}

函数名

函数名 参数1 参数2

4）脚本的中断和退出

exit 退出值 —— 直接放弃剩余的所有操作，直接退出脚本

continue —— 放弃本次循环剩余的操作，继续下一次循环

break —— 放弃整个循环，继续执行done之后的操作

任务需求：编写一个机房巡检脚本 chkidc.sh

1）脚本在执行时需要提供一个文件作参数，这个文件内包含需要被检查的服务器IP地址列表

2）如果没有提供参数，应该提示正确用法，并退出脚本

3）正常操作为检查列表中每一台主机是否存活（ping通），只列出不存活的主机IP

!!!! 位置参数 # ./chkidc.sh idc1.txt

!!!! 提前 if 检查是否给参数，。。。。 exit

!!!! 使用for循环去取得文件内的每一个IP地址，通过 ping 检查

!!!! 如果能通 continue，否则列出不通的主机地址

#!/bin/bash

if [ $# -eq 0 ]

then

echo "Usage:$0 <IP\_LIST\_FILE>"

exit 1

fi

echo "以下服务器已失联："

for ip in $(cat $1)

do

if ping -c3 -i0.2 -W1 $ip &> /dev/null

then

continue

else

echo $ip

fi

done

今日目标：

1）字符串处理（系统指标提取/生成/配置内容替换）

2）扩展脚本技巧（Shell数组、expect交互登录）

3）正则表达式（用特殊的符号来表示不确定的字符串）

pgrep .

任务需求：

1）提取/etc/selinux/config文件的 ^SELINUX= 行保存到变量 A

2）用三种不同方法截取变量A的前7个字符、=号之后的所有字符

# A=$(grep ^SELINUX= /etc/selinux/config)

云服务器（虚拟机）的虚拟网卡的地址

任务需求：

1）生成一个随机的8位密码字串

# uuidgen | cut -b -8

2）编写脚本 /sbin/macgen，执行此命令后能生成一个mac地址，这个mac地址的前3组为 52:54:00，后3组要随机生成

echo "52:54:00:xx:yy:zz"

#!/bin/bash

echo "52:54:00:$(uuidgen |cut -b -2):$(uuidgen |cut -b -2):$(uuidgen |cut -b -2)"

字符串拼接方式：

#!/bin/bash

XX="52:54:00"

for i in 1 2 3

do

XX="$XX:$(uuidgen | cut -b -2)"

done

echo $XX

路径分割：

dirname 文档路径 —— 取得文档的存放位置

basename 文档路径 —— 取得文档的基本名称

vim编辑器的末行替换操作

:s /old/new/

:s /old/new/g

字串替换：

${var/old/new}

${var//old/new}

键盘上 # 在 % 左边

字串砍头：

${var#\*关键词}

${var##\*关键词} 【贪婪模式】

字串剁脚：

${var%关键词\*}

${var%%关键词\*} 【贪婪模式】

任务需求：

1）提取server0上Web服务的网页目录配置，保存到变量webroot

# webroot=$(grep ^DocumentRoot /etc/httpd/conf/httpd.conf)

2）采用剁脚方式处理变量webroot，把网页目录改成 /tedu

# echo "${webroot%% \*} /tedu"

3）采用字串替换方式处理变量webroot，把网页目录改成 /tedu

# echo ${webroot/"/var/www/html"/"/tedu"}

4）采用砍头方式处理变量webroot，只保留目录部分

# echo ${webroot#\* }

5）编写脚本ren1.sh，将当前目录下的 .txt文件的扩展名改为 .text

6）编写脚本ren2.sh，将当前目录下的 .text文件的扩展名改为 .txt

#!/bin/bash

for FILE in \*.text

do

mv $FILE ${FILE%.text\*}.txt

done

7）编写脚本ren3.sh，将当前目录下的 $1 文件的扩展名改为 $2

#!/bin/bash

for FILE in \*$1

do

mv $FILE ${FILE%$1\*}$2

done

Shell变量的初始值

${var:=初始值}

任务需求：

1）编写一个脚本，计算从1～x之间所有整数的和

2）x的值由脚本在运行时提示用户输入，若未输入则默认取值为 1

#!/bin/bash

read -p "Input an integer: " x

while [ "${i:=1}" -le "${x:=1}" ]

do

let sum+=$i

let i++

done

echo "sum=$sum"

变量：使用一个固定的名称来存放一个变化的值

数组：使用一个固定的名称来存放多个变化的值

星期袜/裤 ——

周一 ==》 红色

周二 ==》 橙色

.. ..

周日 ==》 紫色

定义数组：星期袜=(红色 橙色 .. .. 紫色)

调用数组元素：echo ${星期袜[0]}、echo ${星期袜[1]}

任务需求：

改写 chuquan.sh 脚本，利用数组简化代码

#!/bin/bash

quan=(石头 剪刀 布)

echo ${quan[$[RANDOM%3]]}

获取一个数组的元素的个数：

echo ${#数组名[@]}

获取一个变量的字符个数：

echo ${#变量名}

echo ${#数组名[0]}

expect预期交互 ——

<< 操作，用来指定一段文档的起始和结束标记

命令行 << EOF

line asdfasd

line asdfs

EOF

任务需求：编写脚本 ssh.sh

1）能够自动向远程的 server0、desktop0这两台主机部署文件

2）部署的文件内容为"TEDU.CN"，文件位于 /tmp/mike.txt

#!/bin/bash

for ip in server0 desktop0

do

expect << EOF

spawn ssh -o StrictHostKeyChecking=no root@$ip

expect "password" {send "redhat\r" }

expect "#" {send "echo TEDU.CN > /tmp/mike.txt\r" }

expect "#" {send "exit\r" }

EOF

done

echo "文件部署完毕"

任务需求：使用 egrep 完成下列操作

1）列出 /etc/resolv.conf 中包含 net（-i忽略大小写） 的行

2）列出 /etc/resolv.conf 中包含 nameserver 的行

3）列出 /etc/resolv.conf 中不包含 nameserver 的行（-v）

4）列出 /etc/resolv.conf 中以 # 开头或包含 domain 的行（|）

5）列出 /etc/resolv.conf 中包含 172 的行的行号（-n）

6）列出 /etc/resolv.conf 中包含 172 的行的数量（-c）

7）检查 /etc/resolv.conf 中是否已设置 172.25.254.254，如果是则显示yes，不要显示其他无关信息

# egrep -i net /etc/resolv.conf

# egrep nameserver /etc/resolv.conf

# egrep -v nameserver /etc/resolv.conf

# egrep '^#|domain' /etc/resolv.conf

# grep -n 172 /etc/resolv.conf

# grep -c 172 /etc/resolv.conf

# grep -q 172.25.254.254 /etc/resolv.conf && echo yes

正则表达式 —— 通用知识

通过特殊符号来描述一个字符串的特点

a

TEDU

10k+

10k?

任务需求：

1）提取网卡eth0的mac地址

# ifconfig eth0 | egrep -o '([0-9a-f]{2}:){5}[0-9a-f]{2}'

aa:bb:cc:dd:ee:ff

==》 ..:..:..:..:..:..

==》

[0-9a-f]{2}:[0-9a-f]{2}:[0-9a-f]{2}:[0-9a-f]{2}:[0-9a-f]{2}:[0-9a-f]{2}

==》

([0-9a-f]{2}:){5}[0-9a-f]{2}

2）提取网卡eth0的IP地址

# ifconfig eth0 | egrep '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'

aaa.bbb.ccc.ddd

==》.{1,3}\..{1,3}\..{1,3}\..{1,3}

==》[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}

==》([0-9]{1,3}\.){3}[0-9]{1,3}

匹配1开头、1结尾的IP地址字符串：

1[0-9]{0,2}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{0,2}1

+ --》匹配最少一次

? --》匹配最多一次

\* --》匹配任意多次

.\* --》匹配任意长度的任意字符

[ ] --》匹配范围内任意字符

{n,m} --》匹配前面的字符或字符串指定的次数（从n～m次）

任务需求：

1）找出 /etc/resolv.conf 文件中每一行的前3个字符

# egrep -o '^...' /etc/resolv.conf

# cat /etc/resolv.conf | cut -b -3

2）找出 yum info httpd 命令行结果中包含 http://。。。/ 的行

# yum info httpd | egrep 'http://.\*/'

转义（改变字符本来的含义）符号\

\b 代表单词边界

\. 只匹配一个 .

总结：

1）字符串处理（子串截取、替换、砍头、剁脚、初始值）

2）数组

3）expect预期交互

4）正则表达式

任务需求：

1）新建一个文档 reg.txt ，内容如下

12345@163.com

http://sina.com.cn

172.25.0.11

版权所有@2018

1.2.3.4.5.6.7.8

zhsan@tarena.com.cn

2）列出 reg.txt 文件中以数字开头的行

3）列出 reg.txt 文件中合法的电子邮件地址

4）列出 reg.txt 文件中合法的电子邮件地址