规范的脚本构成

1,声明解释器

#!/bin/bash

2,注释

3,代码

bash优点 历史记录, 快捷键,tab,管道,重定向,别名

变量

1,自定义变量 UserTest

unset

2,环境变量 USER UID HOME SHELL PS1 PS2 PATH

env

set | grep XXX

3,位置变量 $1 $2 $3 .....

4,预定义变量 $0 $# $\* $$ $? $!

""

''

``

read -p "提示信息" x

stty -echo

stty echo

a=10 局部变量

export a

运算

expr $a + $b

expr 1 \\* 1

expr 10 % 3

$[] $(( ))

echo $[a+b]

echo $[1\*1]

let a=a+10

let a++

a--

a+=10

echo "1.1+1" | bc

echo "scale=3;10/3" | bc

条件测试:

1,字符串 == 是否相等 != 是否不相等

[ abc == abc ] //常量对比,echo $?得到结果0

[ abc == xyz ] //常量对比,echo $?得到结果非0

[ $USER == root ] //使用环境变量对比当前用户身份

逻辑判断

&& 且 前面任务成功,才执行后续任务

|| 或 前面任务失败,才执行后续任务

; 前面任务执行完毕后,继续执行后续任务,前后无逻辑关系

XX

XX2

XX3

XX ; XX2 ;XX3

[ $USER == root ] && echo 123 //如果当前用户是管理员,才输出123

[ $USER != root ] && echo 123 //如果当前用户不是管理员,才输出123

[ $USER == root ] || echo 123 //如果当前用户不是管理员,才输出123

休息10:05回

[ $USER == root ] && echo "我是管理员" || echo "我不是管理员"

-z 判断是否为空

使用 -z 优化创建用户的脚本,如果没有输入用户名直接回车则退出脚本

#!/bin/bash

read -p "请输入账户名" u

[ -z "$u" ] && exit

useradd $u

#!/bin/bash

yum list installed | grep vsftpd &> /dev/null

[ $? == 0 ] && exit

yum -y install vsftpd

-eq等于 -ne不等于 -gt大于 -ge大于等于 -lt小于 -le小于等于

编写脚本,每2分钟检查系统登录的用户数量,如果超过3人,则发邮件给管理员报警

#!/bin/bash

n=`who |wc -l`

[ $n -gt 3 ] && echo "报警,有人攻击服务器!!" | mail -s test root

或

[ $n -gt 3 ] && mail -s test root < a.txt //需要提前准备a.txt文件,文件的内容就是邮件内容

chmod u+x test3.sh //给脚本赋予x权限

crontab -e

\*/2 \* \* \* \* /opt/test3.sh //每2分钟执行一次脚本

rm /var/spool/mail/root //管理员的邮件,可以先删除此文件

文件测试, 加 ! 可取反测试

-e 是否存在,不分文件类别

[ -e /opt/abc ] //判断abc文件是否存在

-f 是否存在,且是普通文件

[ -f /opt/abc ] && echo "存在且是普通文件" || echo "文件不存在或者不是普通文件"

-d 是否存在,且是目录

[ ! -d /opt/xyz ] && mkdir /opt/xyz //测试目录xyz是否存在,如果不存在则创建

-r 是否可读 (对root无效)

-w 是否可写 (对root无效)

-x 是否可执行

== != -z ! -z (-n,测试时,变量加引号)

-eq -ne -gt -ge -lt -le

-e -d -f -r -w -x

&& || ;

if单分支

if 条件测试 ; then

命令序列

fi

if 条件测试

then

命令序列

fi

read -p "请输入用户名" u

if [ -z $u ] ; then

echo "必须输入用户名"

exit

fi

useradd $u

if双分支

if 条件测试 ;then

命令序列1

else

命令序列2

fi

ping命令结合选项增加工作效率

-c ping的次数

-i 每隔多久ping一次,单位是秒

-W 反馈结果的时间,如果不通时可加快反馈时间,单位是秒

#!/bin/bash

ping -c 3 -i 0.2 -W 1 172.25.0.11 &> /dev/null

if [ $? -eq 0 ];then

echo "通了"

else

echo "不通"

fi

#!/bin/bash

ping -c 3 -i 0.2 -W 1 $1 &> /dev/null

if [ $? -eq 0 ];then

echo "$1通了"

else

echo "$1不通"

fi

if多分支

if 条件测试 ;then

命令序列1

elif 条件测试;then

命令序列2

elif 条件测试;then

命令序列3

else

命令序列X

fi

猜对了

猜大了

猜小了

for 变量名 in 值列表

do

命令序列

done

for循环,执行多次相同任务时使用

基本用法:

#!/bin/bash

for i in a b //有a b两个值,循环两次

do

echo 123 //执行两次任务

done

利用for循环测试多台主机是否可以连通:

#!/bin/bash

for i in {1..15} //准备15个值,循环15次,第一次ping,

do 变量i的值是1,依次类推

ping -c 3 -i 0.2 -W 1 172.25.0.$i &> /dev/null

if [ $? -eq 0 ];then

echo "172.25.0.$i通了"

else

echo "172.25.0.$i不通"

fi

done

100

{1..100}

`seq 100`

while 条件测试

do

任务序列

done

while :

do

任务序列

sleep 1

done

#!/bin/bash

x=$[RANDOM%101]

a=0

while :

do

let a++

read -p "请输入一个数字(0-100)" n

if [ $x -eq $n ];then

echo "猜对了!猜了$a次"

exit

elif [ $n -lt $x ];then

echo "猜小了!"

else

echo "猜大了!"

fi

done