# Automating admin tasks using shell scripts and cron

Vijay Kumar Adhikari

vijay@kcm.edu.np

## How do we go?

- Introduction to shell scripts
- Example scripts
- Introduce concepts at we encounter them in examples
- Introduction to cron tool
- Examples

## Shell

- The “Shell” is a program which provides a basic human-OS interface.
- Two main ‘flavors’ of Shells:
  - sh, or bourne shell. It’s derivatives include ksh (korn shell) and now, the most widely used, bash (bourne again shell).
  - csh or C-shell. Widely used form is the very popular tcsh.
  - We will be talking about bash today.

# sh script syntax

- The first line of a *sh* script *must (should?)* start as follows:
  #!/bin/sh
  (shebang, http://en.wikipedia.org/wiki/Shebang )
  Simple unix commands and other structures follow.
- Any unquoted # is treated as the beginning of a comment until end-of-line
- Environment variables are $EXPANDED
- “Back-tick” subshells are executed and `expanded`

## Hello World script

```
#!/bin/bash
#Prints "Hello World" and exists
echo "Hello World"
echo "$USER, your current directory is $PWD"
echo `ls`
exit #Clean way to exit a shell script
----------------------------------------
```

To run

i. sh hello.sh

ii. chmod +x hello.sh
   ./hello.sh

## Variables

```
MESSAGE="Hello World“ #no $
SHORT_MESSAGE=hi
NUMBER=1
PI=3.142
OTHER_PI="3.142“
MIXED=123abc
new_var=$PI
echo $OTHER_PI # $ precedes when using the var
```

- Notice that there is no space before and after the ‘=‘.

## Variables cont…

```
#!/bin/bash
echo "What is your name?"
read USER_NAME # Input from user
echo "Hello $USER_NAME"
echo "I will create you a file called
   ${USER_NAME}_file"
touch "${USER_NAME}_file"
```

---------------------------------------

Exercise:

Write a script that upon invocation shows the time and date and lists all logged-in users. The script then saves this information to a logfile.

## Sample solution

```
#!/bin/bash
DATE_TIME = ‘date`
echo $DATE_TIME
USERS = `who`
echo $USERS
echo $DATE_TIME $USERS > log
exit
```

# Control Structures

- If

```
#!/bin/bash
T1=43
T2=43
T3=42
if [ $T1 = $T2 ];
then
 echo expression evaluated as true
else
 echo expression evaluated as false
fi
if [ $T1 = $T3 ];
then
 echo expression evaluated as true
else
 echo expression evaluated as false
fi
```

## Control Structures

- For loop

```
#!/bin/bash
for i in $( ls ); do
  echo item: $i
done
```

- While loop

```
#!/bin/bash
COUNTER=0
while [ $COUNTER -lt 10 ]; do
  echo The counter is $COUNTER
  let COUNTER=COUNTER+1
done
```

## Example – while loop

```
#!/bin/bash
while read f
  do
    case $f in
      hello) echo English ;;
      howdy) echo American ;;
      gday) echo Australian ;;
      bonjour) echo French ;;
      "guten tag") echo German ;;
      *) echo Unknown Language: $f ;;
    esac
done
```

## Useful file tests

**-d** $var - file is a directory
**-e** $var - file exists
**-f** $var - file is a file (i.e., not a directory)
**-L** $var - file is a symbolic link
**-p** $var - file is a named pipe
**-S** $var - file is a socket
**-o** $var - file is owned by the user
**-r** $var - user has read access
**-w** $var - user has write access
**-x** $var - user has execute access
**-z** $var - file is zero-length
All return True if correct

## When things go wrong.

-vx, set or bash

## Example - search

```
#! /bin/sh
f=$1            #first parameter passed to the script
for d in *
do
  if test -e $d/$f
    then
      echo FOUND: $d/$f
      exit
  fi
done
echo $f not found
```

## Example – simple one-liner

```
#!/bin/bash
find / -perm 0777 -print >`date
  +%Y-%m-%d`
```

## Example – route-backups

```
#!/bin/bash
TODAY=`date +%Y-%m-%d`
ACCOUNT=pch@npix.woodynet.pch.net
ssh $ACCOUNT show ip route > route.$TODAY
ssh $ACCOUNT show ip bgp > bgp.$TODAY
bzip2 *.$TODAY
```

## Example – Backup script

```
#!/bin/bash
SRCD="/home/"
TGTD="/var/backups/“
OF=home-$(date +%Y%m%d).tgz
tar -cZf $TGTD$OF $SRCD
exit
```

## Example – watch for some user

```
#!/bin/bash
case $# in
1) ;;
*) echo 'usage: watchfor username' ; exit 1
esac
until who | grep -s "$1" >/dev/null
do
  sleep 5
done
echo "$1 has logged in"
```

## Example ftp (non interactive)

```
#!/bin/sh
HOST=$1
USERNAME=$2
PASS=$3
FILE=$4
ftp –in <<EOF
open $HOST
user $USERNAME $PASS
bin
hash
prompt
dele $FILE
put $FILE
bye
EOF
echo "$FILE backed up successfully" | mail –s "backup”  "$USERNAME@$HOST"
```

# Example mysql-backup

```
#/bin/bash
HOST=$1; USER=$2; PASS=$3
FILENAME=`date +%Y%m%d-%H%M`
DIRNAME=/home/vijay/mysqldumpdir/
cd $DIRNAME
mysqldump -h$HOST -u$USER -p$PASS --
  all-databases > $FILENAME
bzip2 $FILENAME
```

## Example – delete old dir

```
#!/bin/bash

# wished time. older dirs will be deleted.
time="2005-07-10 00:00"

reffile=wipeout.ref.$RANDOM
touch -d "$time" $reffile
echo
echo Deletes all dirs that are older than $time
echo
find . -type d -maxdepth 1 -path './*' ! -newer $reffile | while read
    dir; do
  echo  rm -rf "$dir"
  rm -rf "$dir"
done
rm -f $reffile
```

```
#!/bin/sh

#Pings all the IPs in a /24 network
COUNT=0
X=1
while [ $X –lt 255 ]
do
 ping –c 1 "$1.$X"
 if [ $? = 0 ];
 then
   echo "$1.$X is alive"
   COUNT=$(($COUNT 1))
 fi
 X=$((X+1))
done
echo $COUNT hosts responded
```

## Crontab

- A crontab file contains instructions to the cron daemon of the general form: “run this command at this time on this date”.
- Each user has their own crontab, and commands in any given crontab will be executed as the user who owns the crontab.

## Crontab cont…

cron(8) examines cron entries once every minute

The time and date fields are:

| Field | allowed values |
|---|---|
| Minute | 0-59 |
| Hour | 0-23 |
| day of month | 1-31 |
| Month | 1-12 (or names, see below) |
| day of week | 0-7 (0 or 7 is Sun, or use names) |

A field may be an asterisk (*), which always stands for “first-last”.

## Examples

```
crontab -e


# run five minutes after midnight, every day
5 0 * * * $HOME/bin/daily.job >> $HOME/tmp/out

# run at 2:15pm on the first of every month -- output to be mailed
15 14 1 * * $HOME/bin/monthly

5 4 * * sun echo "run at 5 after 4 every sunday"
```

## Examples cont…

```
*/5 * * * * wget -q -O /dev/null http://classroom.kcm.edu.np/cron.php
1 0 * * * /root/backup_scripts/main 2> /root/backup_scripts/logs/lastlog >
    /dev/null
```

## Can you do this?

- Create a script that creates a zip archive of your public_html directory.
- Create a script that checks to see if a host is alive(responds to your ping request)
- Setup cron to run these scripts every 2 hours.

## References

- http://steve-parker.org/sh/sh.shtml
- http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.htm
- man 5 crontab

# Thank you

QUESTIONS?