Bash Script

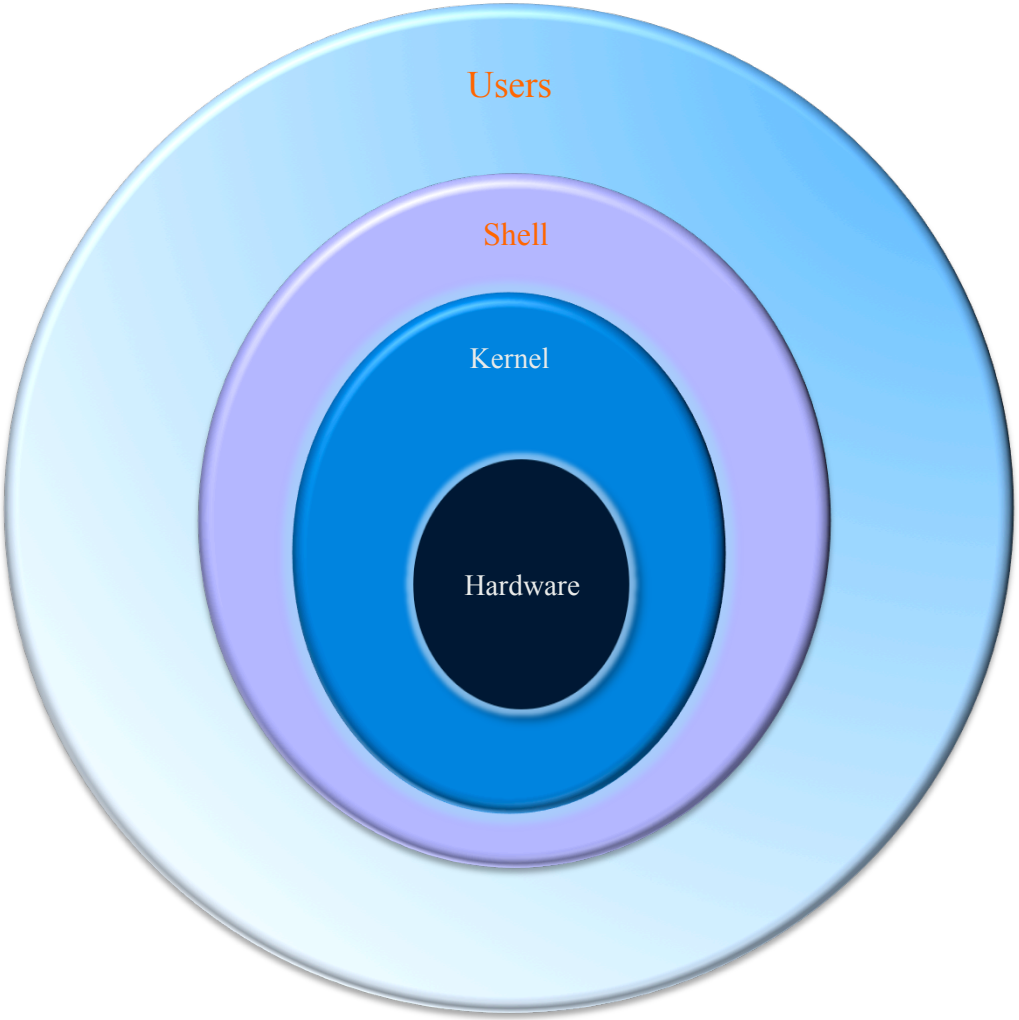
CIRC Summer School 2015

Baowei Liu

Command Lines VS. Bash Script

- Unix/Linux commands in a text file
- A series of commands executed in batch mode

Review of Linux



Linux and Shells

From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

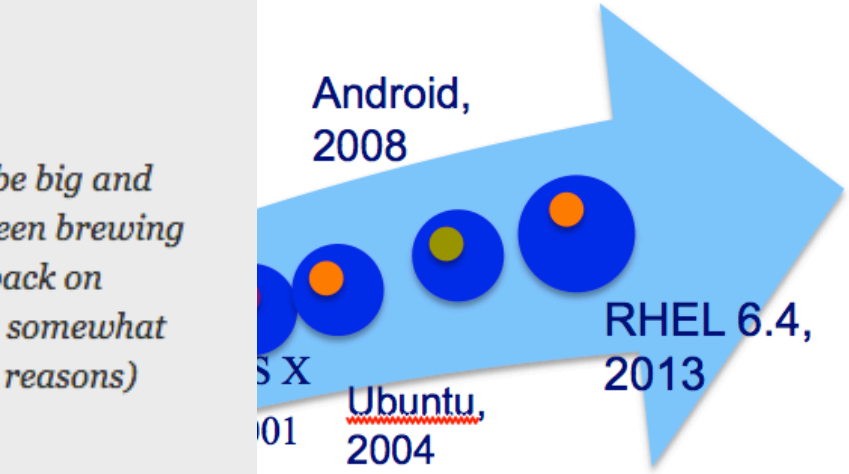
Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready. I'd like any feedback on things people like/dislike in minix, as my OS resembles it somewhat (same physical layout of the file-system (due to practical reasons) among other things).

I've currently ported bash(1.08) and gcc(1.40), and things seem to work. This implies that I'll get something practical within a few months, and I'd like to know what features most people would want. A are welcome, but I won't promise I'll implement them 😀

Linus (torvalds@kruuna.helsinki.fi)

PS. Yes – it's free of any minix code, and it has a multi-thr It is NOT protable (uses 386 task switching etc), and it pr will support anything other than AT-harddisks, as that's all I have :-(.



https://en.wikipedia.org/wiki/

```
cd /media/botwindata/repositories/astrobear_dev
changeSet=`hg heads | head -n 1`
currentRevision=`echo ${changeSet} | awk '{print $2}'`
oldRevision=`cat /home/bliu/mornitor_Revision/revision.old`

TITLE="New Revision < "${currentRevision}" > Available in Dev Repo"

if [ "$currentRevision" != "$oldRevision" ]; then
cat <<EOF |/usr/lib/sendmail -t -oi
To: $ADDRS
Reply-to: $SENDER
From: $SENDER
Subject: $TITLE

======================== AstroBEAR Reminder ============================

A new revision `echo ${currentRevision}` has been checked in to the our dev repo:
/media/botwindata/repositories/astrobear_dev.

========================================================================

EOF

fi
```

Examples Using Bash Script – Pipeline things

- File manipulation
- Wrappers

Shell Script VS. Other Script Language

- Easy to program: the commands and syntax are exactly the same as those directly entered at the command line. Quick start.
- Slow run speed comparing with other programming languages
- Not easy for some tasks like floating point calculations or math functions
- Not friendly to use: error messages/white space.

Linux Commands

- ls: list directory contents
- cd: change directory
- man: manual
- echo

Linux Command echo

- Display a line of text
- Example: echo hello world
- “…” or ‘…’

To Write a Bash Script

- An editor: vi emacs, nano,….
- Specify interpreter as bash: #!/bin/bash
- Some Linux commands
- Comments: # (single line)
- Set executable permission

File permissions and First Script

```
-rw-r-----@ 1 liu  staff  446317 Jan 20 14:08 TcshAndShScreenCapture.png
drwxr-xr-x@ 9 liu  staff     306 Jan 23 12:31 Tests
```

- Three scopes and permissions
- Bash script has to have execute permission to allow the operating system to run it.
- Check permissions: ls –l
- Add execute permission: chmod +x
- First script

Bash Variables

- Create a variable: name=value
- No data type
- No need to declare but can be declared with "declare command"
- No space allowed before and after =
- Use $ to refer to the value: $name

Environment Variables

- env
- $SHELL
- $PATH
- $LD_LIBRARY_PATH
- $RANDOM
- $USER

Variable Value

- Assign value: a=2
- Pass value: b=$a
- Display value: echo $a
- Multiple Variables
- Strong quoting & weak quoting

Assign Variable Value

- Parameter expansion ${}
- Command Substitution: $(), or `
- Arithmetic expansion: $((…))

Arithmetic Expression

- Arithmetic operators: + - * /
- Integer only
- Arithmetic Expansion ((…))
- Floating point calculation: other tools like bc, or awk

Basic calculator: bc

- An arbitrary precision calculator language
- Simple usage: echo $a+$b | bc
- Can use math library: echo “s(0.4)” | bc –l

Conditional Expression and if

- If condition
  then
    ….
  else
    ….
  fi

Conditional Expression

- Integers (Numeric Comparison): (( ))
- operators ==, !=,>,<,>=,<=
- You can use standard C-language operators inside (( ))
- white spaces are not necessary

Conditional Expression: Strings

- Compare strings: [[“$a” = “$b”]]
- operators = or ==, !=, >, < (careful!)
- White spaces around [[ ]] and operators are necessary!!

```
if [[ $a=$b ]]; then
 echo "$a=$b"
else
 echo "$a!=$b"
 fi
```

- -n (not null), -z(null)

Compound Operators

- &&, ||

```
if [[ … ]] && [[ … ]]
then
    ….
 fi
```

A c

- Old

Feature	new test [[	old test [
string comparison	>	\> (*)
	<	\< (*)
	= (or ==)	=
	!=	!=
integer comparison	-gt	-gt
	-lt	-lt
	-ge	-ge
	-le	-le
	-eq	-eq
	-ne	-ne
conditional evaluation	&&	-a (**)
	||	-o (**)

Compare Floating Point Numbers

- Use Basic Calculator: bc

  compare_results=`echo "$a>$b" | bc`

  double quotation are important!!
- Operators: ==, !=, >, >=, <, <=
- Convert to integer (Return 1 for True and 0 for False)
- Always check the command before using it!

Shell Expansions Review

- Parameter Expansion: $variable, ${variable}
- Arithmetic Expansion: $((expression))
- Command Substitution: $() or ``