Bash 101
Intro to Shell Scripting; Lightning

Updated: 2015-11-16

JP Vossen, CISSP
bashcookbook.com

http://www.jpsdomain.org/public/bash_101_lightning.pdf
http://www.jpsdomain.org/public/bash_101_lightning.odp

What is a “shell script?”

- Fundamentally just a list of commands to run
  - May use arguments, variables, various control logic and arithmetic to figure out what to run when
  - bash is integer only, other shells may not be
- Plain text file
- Used on CLI only
- Builds on:
  - The “Unix” tool philosophy
  - The “Unix” everything-is-a-file philosophy

Why should I care?

- You can write new commands!
  - Save time & effort and make life easier
  - E.g., if you always type in four commands to accomplish some task, type them once into an editor, add a “shebang” line and comments, save and set execute permissions. You now have a shell script!
- Automation
  - cron
- Consistency & Reliability
- (Process) Documentation
- One-liners

How do I get started?

- Fire up an editor
  - #!/bin/bash -
    echo 'Hello world, my first shell script!'
  - chmod +r script
- bash 'help' command!
  - 'help set' vs. 'man set'
- basic operation is invocation = you run stuff
- Most of a Linux system is run by shell scripts. They are everywhere, find some and read them.
  - Everything in /etc/init.d/
  - for i in /bin/*; do file $i | grep -q 'shell script' && echo $i; done *# You will be surprised!*

Positional Parameters

- “Main” script:

```
      $0        $1   $2    $3
  myscript  foo  bar  baz
```

- $# = number of parms
- $* = “$1 $2 $3” # a single string of all parms, separated by first character of $IFS (Internal Field Separator)
- “$@” = “$1” “$2” .. “$N” # For re-use later
- Reset inside a function
  - $1 = first arg to function, not script
  - But use $FUNCNAME instead of $0

Quotes

- The shell re-writes the line
- White space is a delimiter!
- Quoting
  - Use ' ' unless you are interpolating $variables, then use " "
  - echo 'foo'
  - echo "$foo"
  - grep 'foo' /path/to/file
  - grep "$regex" /path/to/file
- Except when it's not. Can make your head hurt.

Variables

- USE GOOD NAMES!!!
- No $ or spaces around = when assigning:

```
foo='bar'
foo="bar$baz"
```

- $ when referencing value:

```
echo "$foo"
```

- Append:

```
foo="$foo bar"
```

- Needs ${} if variable followed by [a-zA-Z_0-9]

```
foo="foo $bar baz"    # OK
foo="foo${bar}baz"    # $bar needs ${}
```

debugging

- DO NOT CALL YOUR TEST SCRIPT 'test'!
- PS4='+xtrace $LINENO: '
  - First character is duplicated to show nesting level, that's why I have '+' there
  - $LINENO should be in the default PS4 prompt!

- bash -n path/to/script # gross syntax check
- bash -x path/to/script # run-time debugging

- set -x & set +x # debug on / off
- set -v & set +v # verbose on / off

URLs, Wrap-up and Q&A

- URLs:
  - TONS of resources: http://www.bashcookbook.com/bashinfo/
  - These slides: http://www.jpsdomain.org/public/bash_101_lightning.pdf or http://www.jpsdomain.org/public/bash_101_lightning.odp
  - Bash vs. Dash: http://www.jpsdomain.org/public/2008-JP_bash_vs_dash.pdf and *aptitude install devscripts* then use *checkbashisms*
  - The sample script: http://www.jpsdomain.org/public/cdburn
  - STDIN, STDOUT, STDERR: http://en.wikipedia.org/wiki/Standard_streams
  - Revision Control: http://www.jpsdomain.org/public/Revision_Control_for_the_Rest_of_Us.pdf and (older) http://www.jpsdomain.org/public/PANTUG_2007-06-13_appd=Revision_Control=JP.pdf
  - Windows Shell Scripting (cmd.exe): http://www.jpsdomain.org/windows/winshell.html
  - BASH Prompt HOWTO: http://www.tldp.org/HOWTO/Bash-Prompt-HOWTO/index.html
  - Cygwin: http://www.cygwin.com/
  - UnxUtils: http://unxutils.sourceforge.net/
  - GNU Win32 ports: http://sourceforge.net/projects/gnuwin32/
  - Win32 Perl http://www.activestate.com/solutions/perl/
  - Mac; this is awesome http://examples.oreilly.com/0636920032731/Terminal_Crash_Course.pdf
- Questions?
- I'm on the PLUG list... jp@jpsdomain.org