CS2042 - Unix Tools

Fall 2010

Lecture 8

Hussam Abu-Libdeh

based on slides by David Slater

September 24, 2010

Compression & Archiving

- `zip` / `unzip`
  - Compress and archive (bundle) files into a single file.
  - A new compressed .zip file is created and the original files stay intact.
  - `zip <zipped_file_name> <files_to_compress>`
  - `unzip <zipped_file_name>`
  - Many options! E.g., add files to an existing zip, encrypt with a password ..etc

Compression & Archiving

- `gzip`
  - Compress files using Lempel-Ziv coding.
  - Does not bundle files, the compressed files will replace the original files.
  - `gzip <file_to_compress>`
  - `gunzip <compressed_file>`
- `bzip2`
  - Compress files using Burrows-Wheeler block sorting text compression algorithm and Huffman coding.
  - More efficient than `gzip` on most files, but a bit slower.
  - Like `gzip`, this is only a compression tool, and thus compressed files will replace the original files.
  - `bzip2 <file_to_compress>`
  - `bunzip2 <compressed_file>`

Tarballs!

- To archive multiple files together, we can use the "Tape Archive" utility (`tar`).
- `tar` bundles multiple files together into a single file (but does not compress them or replace them)
  - `tar -cf archive.tar foo bar`
    Create archive.tar from files foo and bar
  - `tar -xf archive.tar`
    Extract all files from archive.tar

Compressed Tarballs

- To compress a tarball we can pipe the outcome of `tar` to a tool like `gzip` or `bzip2`.
- However, `tar` has flags to automatically do this:
  - `-z` : compress using `gzip`
  - `-j` : compress using `bzip2`
  - `tar -czf archive.tar.gz foo bar`
    Creates a compressed file (archive.tar.gz) from files foo and bar
- **Naming convention:**
  - `archive.tar.gz` or `archive.tgz`: gzipped tarballs
  - `archive.tar.bz2` or `archive.tbz`: bzip2 tarballs
- Works with directories too!
  - `tar -czf cs2042.tgz cs2042/*`
    Creates a compressed file containing the directory and contents of cs2042 directory

You have the power!

We now have a variety of UNIX utilities at our disposal and it is time to learn about

scripting!

Scripting 101

Definition:

A script is very similar to a program, although it is usually much much simpler to write and are from source code (or byte code) via an interpreter. *Shell scripts* are scripts designed to run within a command shell like `bash`.

Scripts are written in a scripting language, like perl, ruby, python, sed or awk. They are then run using an interpreter. In our case, the scripting language and the interpreter are both **bash**.

The Shebang

All the shell scripts we'll see in this course begin the same way: with a **shebang** (#!). This is followed by the full path of the shell we'd like to use as an interpreter: `/bin/bash`

Example:

```
#!  /bin/bash
# This is the beginning of a shell script.
```

- Any line that begins with # (except the shebang) is a comment
- Coments are ignored during execution - they serve only to make your code more readable.

Setting Variables

Creating and setting variables within scripts works the same as in the shell

Example:

```
MYVAR="A new variable"

echo $MYVAR
A new variable
```

- NOTE: No spaces around the equal sign!!!!!!!

Simple Examples:

Example: hello.sh

```
#!  /bin/bash
echo "Hello World"
```

Now set your file permissions to allow execution:

Example:

```
chmod u+x hello.sh
```

And finally you can run your first shell script!

```
./hello.sh
Hello World!
```

Hello World - String Version

Lets modify slightly and use a variable:

Example: hello2.sh

```
#!  /bin/bash
STRING="Hello again, world!"
echo $STRING
```

Set your permissions and run:

```
chmod u+x hello2.sh && ./hello2.sh
Hello again, world!
```

A Backup Script

Here is something a little more practical - a simple script to back up all the files in your documents directory:

Example: backup.sh

```
#!  /bin/bash
tar -czf  /backups/cs2042.backup.tar.gz \
 /Documents/cs2042/
```

This script makes use of the `tar` archiving command:

Making Tarballs:

```
tar -c(z/j)f <dest_archive> <source>
tar -x(z/j)f <archive>
```

- `-c` version creates a new archive from a source file/dir
- `-x` extracts an existing archive to the current dir
- pick either `-z` or `-j` options (-z ⇒ .tar.gz , -j ⇒ .tar.bz2)

Backup Script With Date

Lets add the current date to the name of our backup file.

Example: backupwithdate.sh

```
#!  /bin/bash
tar -czf ~/backups/cs2042_$(date +\%d_\%m_\%y).tar.gz \
~/Documents/cs2042/
```

- Today, will write to a file named `cs2042_14_10_2009.tar.gz`
- Note the `\` at the end of the second line. This escapes the end of line character!