Architecture et Systèmes

Stefan Schwoon

Cours L3, 2014/15, ENS Cachan

Introduction to the Unix shell

The *shell* (`sh`, `bash`, ...) is a systems program, i.e. a “low-level” application, not far removed from the OS.

Provides a rich, text-based interface for managing processes.

Easy access to other systems programs that implement OS functionalities, in particular operations on files.

User types commands at the prompt.

Shell executes the command and waits for its completion before accepting the next command.

Simple shell commands

Standard form of a shell command:

```
program_name  arg1  arg2  ...
```

Spaces separate the program name and the arguments.

Example: `echo Bonjour`

Command: `echo` (simply outputs its arguments)

Example: `cat foo.txt`

Command: `cat` (shows contents of file)

Example: `ll`

Command: `ls` (`ll` is an alias for `ls -l`)

If an argument itself contains a space, enclose it in quotes:

```
cat "my file"
```

Combining multiple commands

`cmd1 ; cmd2`: execute first cmd1, then cmd2

`for i in list; do cmds; done`: execute cmds once for each element of list; set `i` to value of list element

Working with files and directories

Directories organized in tree-like fashion.

Notion of a current directory (initially, the user's home directory).

Filenames are interpreted relative to the current directory.

`toto` means a file named toto in the current directory.

`foo/toto` means a file named toto in the directory foo, which is contained in the current directory

`../toto` means toto in the directory above

`./toto` is equal to `toto` (`.` is the current directory)

`/home/schwoon/toto`: path from the root directory

Combine at will, e.g. `../../bin/toto`

Hint: TAB key auto-completes filenames

Directories

List the content of a directory:

ls: e.g., `ls` (current dir), `ls /bin` (list given dir)

Variant: `ls -l` (additional infos)

Change the current directory:

cd: e.g., `cd bin`, `cd ..`

Special forms: `cd` (back to home), `cd -` (back to previous)

Create a new directory: mkdir: e.g., `mkdir mydir`

Remove a directory: rmdir: e.g., `rmdir mydir` (only if empty)

Dealing with files

General remark: More details on each command can be obtained using man or info, e.g., man ls.

cat: Show content of file(s), e.g., cat toto tata

less: Show content of file, with paging

mv: Rename file(s) or move it/them to another directory.

cp: Duplicate file(s), copy to another directory

rm: Remove file(s)

Wildcards

Wildcards can be used to specify multiple files at the same time.

Question mark (`?`) means an arbitrary character.

> e.g., `cat t?t?` may display files `toto`, `titi`, `tito`, if they exist.

Star (*) means an arbitrary number of arbitrary characters.

> e.g., `mv *.c backup/` would move `foo.c`, `toto.c` etc to the backup directory.

Commands for text manipulation

sort: sort lines of text in a file

grep: search for pattern in a file and output only those lines

wc: count number of words, lines etc

uniq: eliminate duplicate lines

cut: display only certain parts of a line

head: display first couple of lines of a file

tail: display last lines of a file

Many useful options, consult man pages!

Redirection

Most commands work that we discussed above work on files. But, if no file is specified, they read from the so-called standard input. By default, the standard input is the keyboard, but it may be changed to something else, for instance a file.

General format: `cmd < file`: take input from file rather than console

Example: `cat < toto` (not so useful here, but with other commands)

Similarly, commands print their results to something called standard output, which is by default the console. It may be changed to a file.

General format: `cmd > file`: output to file rather than console

Example: `echo foo > toto` (will overwrite previous contents of toto)

Variant: `echo foo >> toto` (append to file)

Redirect error messages, e.g. suppress them completely: `cmd 2> /dev/null`

Pipes

Pipes are a special kind of redirection that connects the output of one program to the input of another.

General format: `cmd1 | cmd2`

Example: `grep word file | less`

Pipes can be combined multiple times, e.g. `sort file | uniq | less`