[Software Development]

Linux toolset (part C)

Davide Balzarotti

Eurecom – Sophia Antipolis, France

The Right Tool For the Right Job

Find files that contains some text	`find | grep`
Number of lines, words, bytes	`wc`
Print certain lines (by number)	`sed`
Print certain lines (by content)	`grep`
Extract patterns	`grep -o`
Remove lines that match a pattern	`grep -v`
Remove certain characters from all lines	`tr`
Remove/substitute a string/pattern	`sed`
Convert to lower/upper case	`tr`
Group lines (w/o frequency)	`sort | uniq -c`
Intersection, union, complements of files	`sort | uniq`

The Right Tool For the Right Job

Remove/keep some columns	`cut, awk`
Repeat a command for each file	`find -exec, find | xargs` `for loop`
Repeat a command for each line	`cat | while loop` `awk`
Repeat a command for each word	`fmt | while, tr | while, awk`
Insert a line	`sed`
Extract part of a file (from ... to)	`sed`
Anything that requires some math	`awk`
Working on record split on multiple lines	`awk`
Modify one or more columns	`awk`

Internet from the Command Line

- Many network- and web-related command lines tools
  - Text-based web browsers (`w3m, lynx, links`)
  - Non-interactive downloaders (`wget, curl`)
  - General input/output through sockets (`netcat`)
  - DNS query (`host, dig`)
  - Pinging, scanning, trace routing (`ping, arping, nmap, traceroute, hping3`)
  - Remote shell, secure copy (`ssh, scp`)
  - ....

Internet from the Command Line

- Many network- and web-related command lines tools
  - Text-based web browsers (`w3m, lynx, links`)
  - **Non-interactive downloaders** (`wget, curl`)
  - **General input/output through sockets** (`netcat`)
  - DNS query (`host, dig`)
  - Pinging, scanning, trace routing (`ping, arping, nmap, traceroute, hping3`)
  - Remote shell, secure copy (`ssh, scp`)
  - ....

wget

- `wget` download files from the web using either HTTP, HTTPS, or the FTP protocol

```
wget [options] url
```

- It can retrieve a single document or recursively download everything that is referred to from a remote resource
- Parameters
  - `-m` – mirror the entire web site
  - `-r` – recursively download all the linked documents
  - `-l n` – specify the max recursion depth to n
  - `-nd` – don't replicate the directory structure
    (saves all the file in the same directory)
  - `-k` – fix all the the links in the downloaded pages to make them suitable for offline browsing

wget

- `-c` – continue the download of a partially-downloaded document
- `--spider` – don't download the pages, just check that they are there
- `-w sec` – wait sec seconds between each download
- `-U agentstring` – send agentstring as user agent
- `-A pattern` – download only files matching the pattern
- `-R pattern` – do not download files matching the pattern
- `--limit-rate nk` – limit the speed to n KB/s

- Example: download all pdf files:

```
wget -r -l1 -nd -A"*.pdf" http://…url…/
```

CURL

- Very similar to `wget` but support more protocols:
  - FTP, FTPS, HTTP, HTTPS, SCP, SFTP, TFTP, TELNET, DICT, FILE, LDAP and LDAPS
- Curl fetches just the URLs that the user specifies
  - It does not contain any recursive downloading capability
  - It does not parse the HTML
- It supports file upload in HTTP and FTP
- It has a pretty complete cookie management

CURL

`-s` – silent (suppress the progress meter)

`-d data` – send the data in a POST request

`-A useragent` – specify the given useragent

`-F "field=@filename"` – upload a file

`-I` – fetch only the HTTP headers

`-c file` – write the cookies to file (in netscape format)

`-b file` – load the cookies from file

`-u username:password` – specify the authentication credentials

Example: submit a POST form, pretending to be a Mozilla browser

```
curl -s -A "Mozilla/5.0 (compatible; MSIE 7.01; Windows NT   5.0)"
  -d 'rid=value&submit=SUBMIT'
```

URL De-Shortening with CURL

```
curl -sI http://tinyurl.com/m3q2xt
```

```
HTTP/1.1 301 Moved Permanently
Location: http://en.wikipedia.org/wiki/URL_shortening
X-tiny: cache 0.00096607208252
Content-type: text/html
Date: Wed, 20 Oct 2010 09:09:13 GMT
Server: TinyURL/1.6
```

```
sed -n 's/Location: //p'
```

The TCP/IP Swiss Army Knife

- netcat (nc) is a tool to read and write data across TCP or UDP network connections
- It can be used as a generic network client:
  - `nc host port`
  - The standard input is redirected to the remote host
  - The data received from the network is sent to the standard output
- It can be used as a generic network server:
  - `nc -l -p port`
  - It listens for inbound connections to port `port` and then forwards the data as in client mode
  - It manages only one connection and then terminates

Copying Files with NetCat

- On the sender side:

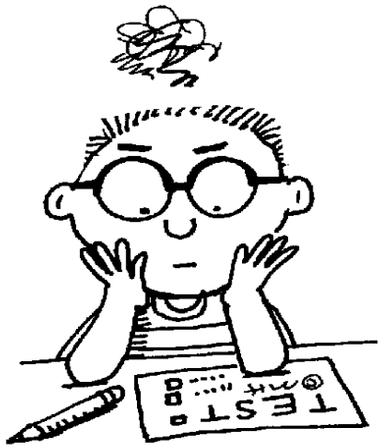
```
tar czf - dir | nc -q 10 -l 3333
```

- On the receiver side:

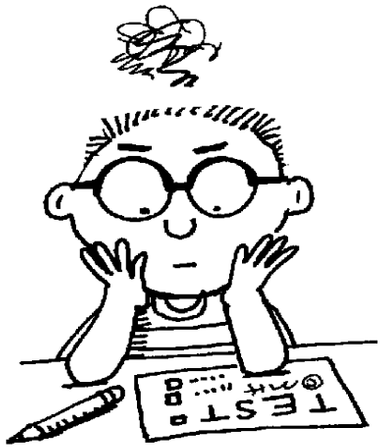
```
nc -w 10 remotehost 3333 | tar xzv
```

- The same can be done by having the received that listens and the sender that open the connection


TEST



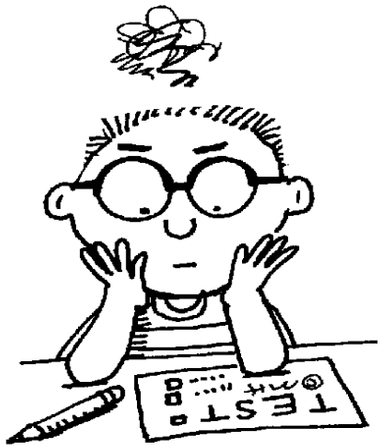
Print the 10 commands you use more frequently



Print the number of packages installed, grouped per calendar day

In a text file:

- Remove everything that is not an alphabetic character
- Fix duplicated words
- Print the longest word in a file



Print the paragraphs that contain a certain word

In a C header file:

- Update the copyright year in the header
- Replace the comment style from /* .. */ to //
- Move inline comments to the previous line

Using “`ls -l`” print the total file size grouped by file extension

So far...

- Something about the history
  - The origins, the Berkley era, the GNU system...
  - The people: Ken Thompson, Dennis Ritchie, Bill Joy, Richard Stallman, Linus Torvalds...
- Something about the philosophy
  - Write programs that do one thing and do it well
  - Programs are tools and like all good tools, they should be specific in function, but usable for many different purposes
  - Remember the pipe - the power is not in the tools, but in the way you combine them together
- Something about the shell and the command line tools
  - You can go a long way without writing a single line of code
  - Regular expressions are useful you in many situation

Scripts

- A way to execute sequences of terminal commands in batch mode
- Text files containing the commands to be executed by a particular interpreter
  - Scripts do not follow the compile-link-run cycle, but are instead interpreted or compiled "*just in time*" (JIT)
- (The distinction is not so clear nowadays)

- Many languages for many different purposes:
  - Shell (bash, C..) - the original, for command scripting
  - Awk – text processing
  - PHP – Web applications, server-side code
  - Javascript – Web client- & server-side code
  - Perl, Python, Ruby – general purpose languages

A Script under the Hood

```
#!/bin/awk -f

BEGIN { print " - START - "}
{ print $0}
END { print " - DONE -" }
```

A Script under the Hood

```
#!/bin/awk -f

BEGIN { print " - START - "}
{ print $0}
END { print " - DONE -" }
```

If the first line starts with #! the shell considers the rest of the line as the name of the interpreter to be used to run the file

Two ways to run the script

```
> awk -f test.awk
```

```
> chmod a+x test.awk
> ./test.awk
```

A World Full of Script

```
> find /bin /usr/bin/ /usr/local/bin -exec file -b {} \; |
grep -i script | sort | uniq -c
```

A World Full of Script

```
> find /bin /usr/bin/ /usr/local/bin -exec file -b {} \; |
grep -i script | sort | uniq -c

      1 a nickle script text executable
      5 a perl script text executable
      4 a python script text executable
    101 a /usr/bin/perl script text executable
      1 a /usr/bin/perl -t script text executable
      1 a /usr/bin/perl -wl script text executable
     44 a /usr/bin/perl -w script text executable
      1 a /usr/bin/perl -wT script text executable
      2 a /usr/bin/ruby1.8 script text executable
      1 awk script text executable
     35 Bourne-Again shell script text executable
    247 POSIX shell script text executable
     73 python script text executable
      1 setuid a /usr/bin/perl -T script text executable
```