Filters and useful UNIX commands.

Department of Computer Systems FIT, Czech Technical University in Prague

Motivation

- One email address of IT student ~ 0,1 USD.
- How to find a list of email addresses of IT students?

- Solution

```
ypcat passwd | cut -d: -f1 | awk '{print $0 "@fit.cvut.cz"}' | tr '\n' ','
```

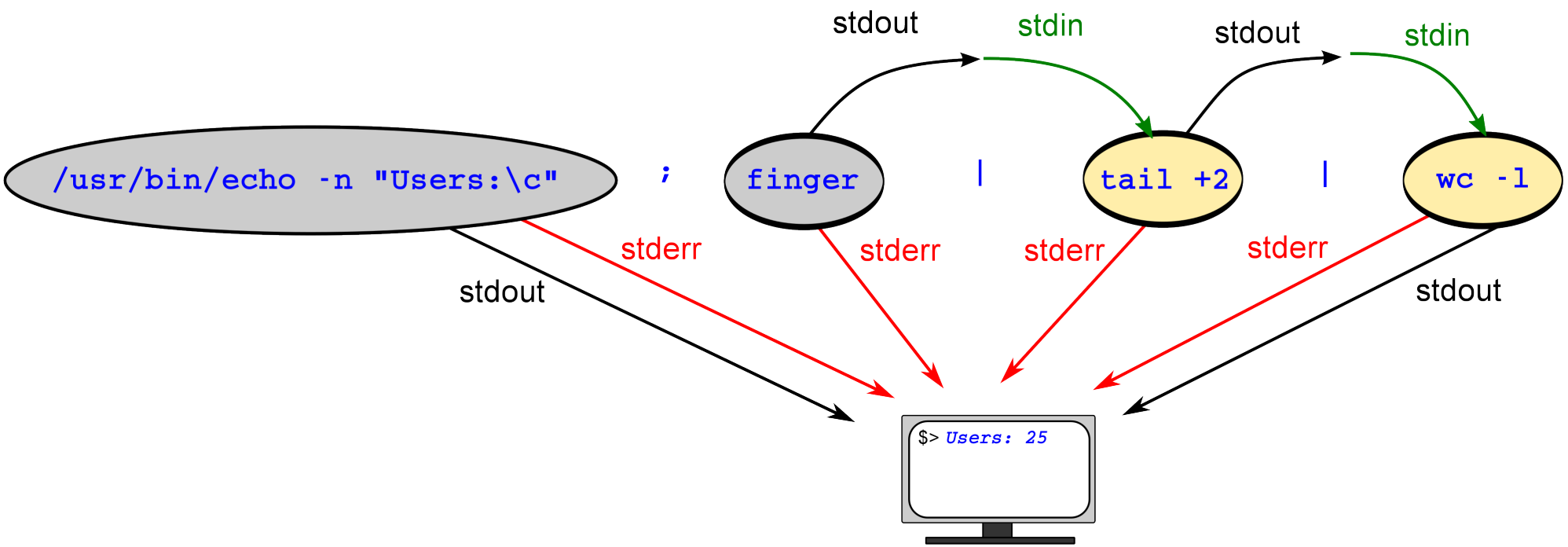
Motivation

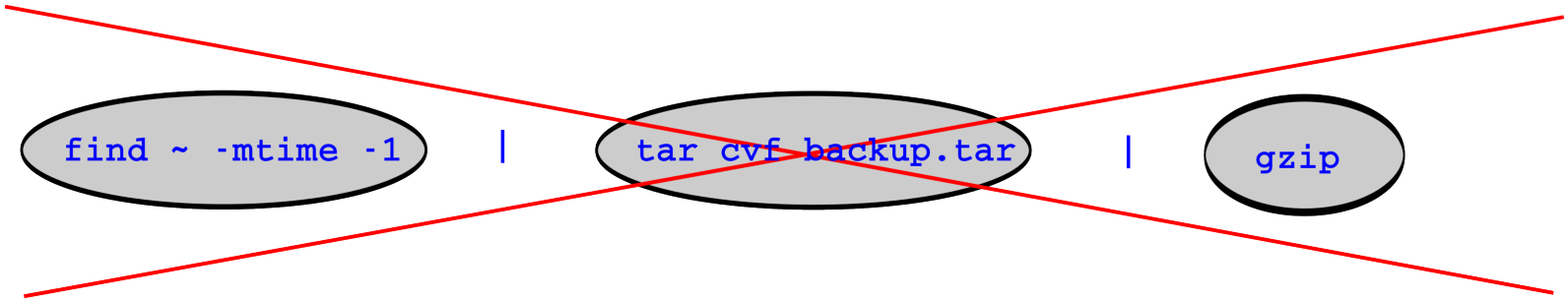
- Filtr
  - **It** is a “simple” program that gets most of its data from its standard input (the main input stream) and writes its main results to its standard output (the main output stream).
  - Examples. `head`, `tail`, `wc`, `cut`, `tr`, …

- UNIX filters are often used as elements of pipelines.
  - `ps -e -o rss,user,comm | sort -k1,1n | tail -1`
  - `du -ks /home/stud/* | sort -k1,1nr | head -10 | sed 's/\/.*\///'`

- Why use UNIX filters and not my own C program?
  - It is not proprietary solution
  - Anyone can simply modify the solution
  - Platform portability

Every application is not filter





```
tee [-a] [files]
```

- The tee utility will copy standard input to standard output, making a copy in zero or more files.

  -a appends the output to the files.

- **Examples:**

```
finger  |  tee a.txt b.txt

spell text.txt  |  tee -a typos
```

nl

```
nl [options] [files]
```

- Line numbering filter.

  -s´*sep*´ *sep* is the character(s) used in separating the line number.
  -bp´pattern´ Specifies which lines are to be numbered.

- **Examples:**

```
ls -l / | nl
ls -l / | nl -s´) ´
ls -l / | nl -bp´^-´
```

head

```
head [-k] [files]
```

- Utility copies the first *k* lines of files (*k*=10 is default).

- **Examples:**

```
ls -lct / | nl | head -4
```

tail

```
tail [options] [file]
```

- Utility prints the last lines of file.
  - -*k* begins printing at *k*-th item from end of file.
  - +*k* begins printing at *k*-th item from beginning of file (GNU: -n +*k*).
  - -f dosn't quit at the end of file (use CTRL-C to quit).
  - -r copies lines in reverse order.

- **Examples:**

```
ls -lct / | nl | tail
ls -lct / | nl | tail -5
ls -lct / | nl | tail +5
tail -f file.txt
```

```
wc [options] [files]
```

- Utility display a count of lines, words and characters in files.
  - -c counts bytes.
  - -w counts words.
  - -l counts lines.

- **Examples:**

```
ls -l / | tail +2 | wc -l
wc -l /etc/passwd
```

cut

```
cut options [files]
```

- Utility cut out selected fields of each line of a file.

  -c *list* specifies characters (e.g. 2-10,15,45-).

  -d *delim* defines the field delimiter (-f option only).

  -f *list* specifies fields separated in the file by a delimiter character.

- **Examples:**

```
ls -l  | cut -c2-10,15-23,54-
ypcat passwd | cut  -d: -f1,3,5
who | cut -d´ ´ -f1
```

paste

```
paste [options] files
```

- Utility merges corresponding or subsequent lines of files.

  -d*list* each character in *list* is an element specifying a delimiter character.

- **Examples:**

```
ypcat passwd | cut -d: -f1 > name.txt
ypcat passwd | cut -d: -f3 > uid.txt
ypcat passwd | cut -d: -f7 > shell.txt
paste -d´+-´ uid.txt name.txt shell.txt
```

split

```
split [options] file [name]
```

- Utility splits a file into pieces of given size with given name:

  name*aa*, **name*ab***, **name*ac***,...

  -b *size* splits a file into pieces n bytes in size

  -l *size* splits a file into pieces n lines in size

  -a length length of name suffix

- **Examples:**

```
split -b 10k /bin/ls ls
cat ls?? > ls
man ls | nl > ls.txt
split -l 10 -a 5 ls.txt ls
cat ls????? > ls.txt
```

uniq

```
uniq [options] [file]
```

- Utility reports or filters out repeated lines in a file.
  - -c precedes each output line with a count of the number of times the line occurred in the input.
  - -d suppresses the writing of lines that are not repeated in the input.
  - -u suppresses the writing of lines that are repeated in the input.

- **Examples:**

```
ps -e -o user | tail +2 | sort > names.txt
uniq names.txt
uniq -c names.txt
uniq -d names.txt
uniq -u names.txt
```

```
tr [options] string1 string2
```

- Utility copies the standard input to the standard output with substitution or deletion of selected characters. The string1 and string2 operands control translations that occur while copying characters.

  -d deletes all occurrences of input characters that are specified by string1.

  -s replaces instances of repeated characters with a single character.

- **Examples:**

```
cat file.txt | tr 'ABC' 'xyz'
cat file.txt | tr '[a-z]' '[A-Z]'
ls -l | tr -s ' ' | cut -d' ' -f2,9-
ls -l| tr -s ' ' '\012'
```

sort

```
sort [options] [files]
```

- Utility sorts lines of all the named files together and writes the result on the standard output.

  -f folds lower-case letters into upper case.

  -n sorts in arithmetic order.

  -M compares as months.

  -r reverses the sense of comparisons.

  -u identical lines in input file appear only one (uniq).

  -t*char* uses *char* as the field separator character.

  -k*start_field* [*type*][,*end_field*[*type*]]

  restricted sort key field definition

Examples:

```
ls -l / | sort
ls -l / | sort -k3
ls -l / | sort -k3,3
ls -l / | sort -k2,2n
ls -l / | sort -k2,2nr
ls -l / | sort -k6,6M
ls -l / | sort -k6,6M -k7,7n
ls -l / | sort -k6,6M | sort -k7,7n
ls -l / | sort -k6,6M -k7,7n -k8.2,8.3n -k8.5,8.6n
ypcat passwd | sort -t':' -k3,3n
```

cmp [options] file1 file2

- The utility compares two files.

  -s writes nothing for differing files and returns only exit status.

- **Examples:**

```
$ cmp  s1.txt s3.txt
s1.txt s3.txt differ: char 15, line 2
```

```
comm [-123] file1 file2
```

- The utility reads file1 and file2, which must be ordered in the current collating sequence, and produces three text columns as output: lines only in file1; lines only in file2; and lines in both files.

Examples:

```
comm  s1.txt s3.txt
comm -12 s1.txt s3.txt
```

```
diff [options] file1 file2
```

- Compare two files.
- The normal output contains lines of these forms:

  n1 **a** n3,n4

  n1,n2 **d** n3

  n1,n2 **c** n3,n4

  where n1 and n2 represent lines file1 and n3 and n4 represent lines in file2 (a,d,c are command for editor ed).

  -u produces a listing of differences with three lines of context(+, -).
  + lines added or changed in file2
  - removed and changed lines in file1

  -c produces a listing of differences with three lines of context. (+,-,!).

patch

```
patch [options] [files]
```

- The patch command reads a source (patch) file containing any of the three forms of difference (diff) listings produced by the diff(1) command (normal, context or in the style of ed(1)) and apply those differences to a file.

-b	saves a copy of the original contents of each modified file (with extension ".orig").
-i patchfile	reads the patch information from the file patchfile.
-R	reverses the sense of the patch script.

Examples:

```
diff -u f1 f2  > patch.diff

patch -b -i patch.diff f1

patch -b –R -i patch.diff f1
```

xargs

- Utility constructs argument lists and invokes following commands.

- **Examples:**

```
mkdir ~/sources
find ~ -type f -name "*.c" | xargs -I {} mv {} ~/sources/
```