ΕΡΓΑΣΙΑ ΜΕ ΕΝΤΟΛΕΣ

Λειτουργικά Συστήματα - Εργαστήριο

Γιωργος Σπαθουλας

Τμήμα Πληροφορικής με Εφαρμογές στην Βιοιατρική

Πανεπιστήμιο Θεσσαλίας

ΕΝΤΟΛΕΣ

Οι εντολές που χρησιμοποιούμε στο terminal μπορεί να είναι :

- Ένα εκτελέσιμο πρόγραμμα που υπάρχει στο directory **/usr/bin**
- Μία εντολή **ενσωματωμένη / built in** στο shell, πχ **cd**
- Ένα **shell function**, τα οποία αποτελούν μικρά script του shell
- Ένα **alias**, εντολές δηλαδή που ο χρήστης έχει αντιστοιχήσει σε άλλες εντολές

Η εντολή **which** επιστρέφει το είδος κάθε εντολής

```
[me@linuxbox ~]$ type type
type is a shell builtin
[me@linuxbox ~]$ type ls
ls is aliased to `ls --color=tty'
[me@linuxbox ~]$ type cp
cp is /bin/cp
```

ΒΟΗΘΕΙΑ ΓΙΑ ΤΙΣ ΕΝΤΟΛΕΣ

- Για να πάρουμε βοήθεια για μία ενσωματωμένη εντολή του shell χρησιμοποιούμε την σύνταξη **help command.name** πχ

```
[me@linuxbox ~]$ help cd
cd: cd [-L|[-P [-e]]] [dir]
Change the shell working directory.

Change the current directory to DIR.  The default DIR is the value of
the HOME shell variable.
```

- Για να πάρουμε βοήθεια για ένα εκτελέσιμο πρόγραμμα χρησιμοποιούμε την σύνταξη **command.name –help** πχ

```
[me@linuxbox ~]$ mkdir --help
Usage: mkdir [OPTION] DIRECTORY...
Create the DIRECTORY(ies), if they do not already exist.

  -Z, --context=CONTEXT (SELinux) set security context to CONTEXT
Mandatory arguments to long options are mandatory for short options
too.
  -m, --mode=MODE   set file mode (as in chmod), not a=rwx - umask
  -p, --parents     no error if existing, make parent directories as
                    needed
  -v, --verbose     print a message for each created directory
      --help        display this help and exit
      --version     output version information and exit
```

- Τα περισσότερα εκτελέσιμα προγράμματα παρέχουν και ένα εκτενές documentation που μπορούμε να δούμε με την εντολή **man**

```
[me@linuxbox ~]$ man ls
```

ΕΚΤΕΛΩΝΤΑΣ ΠΟΛΛΑΠΛΕΣ ΕΝΤΟΛΕΣ

- Είναι δυνατό να εκτελέσουμε περισσότερες από μία εντολές σε μία γραμμή με την χρήση του **συμβόλου ;**

```
command1; command2; command3...
```

```
[me@linuxbox ~]$ cd /usr; ls; cd -
bin  games    kerberos  lib64    local  share  tmp
etc  include  lib       libexec  sbin   src
/home/me
[me@linuxbox ~]$
```

ΔΗΜΙΟΥΡΓΩΝΤΑΣ CUSTOM ΕΝΤΟΛΕΣ

- Είναι δυνατό να αντιστοιχήσουμε περισσότερες από μία εντολές σε μία νέα εντολή που την ονοματίζουμε εμείς με την εντολή **alias**

```
[me@linuxbox ~]$ alias foo='cd /usr; ls; cd -'
```

- Με την εντολή **unalias** μπορούμε να καταργήσουμε αυτή την αντιστοίχηση

```
[me@linuxbox ~]$ unalias foo
[me@linuxbox ~]$ type foo
bash: type: foo: not found
```

- Με την εντολή **alias** χωρίς όνομα μπορούμε να δούμε την εν ενεργεία αντιστοιχήσεις

```
[me@linuxbox ~]$ alias
alias l.='ls -d .* --color=tty'
alias ll='ls -l --color=tty'
alias ls='ls --color=tty'
```

I/O REDIRECTION

- Οι εντολές που αναφέρθηκαν πιο πάνω συνήθως παράγουν κάποιο αποτέλεσμα
- Το αποτέλεσμα αυτό είναι είτε τα **αποτελέσματα** της εκτέλεσης της εντολής είτε καποια **μηνύματα κατάστασης ή λάθους**
- Τα αποτελέσματα στέλνονται σε ένα ειδικό αρχείο το **standard output (ή stdout)** ενώ τα μυνήματα λάθους σε ένα άλλο ειδικό αρχείο το **standard error (stderr)**
- Εξ' ορισμού και τα δύο αυτά αρχεία τυπώνονται στην οθόνη
- Επίσης κάποιες εντολές έχουν και ένα ειδικό αρχείο εισόδου το **standard input (stdin)** το οποίο εξ' ορισμού εισάγεται από το πληκτρολόγιο
- Το **I/O redirection** μας επιτρέπει να αλλάξουμε την **πηγή** του **stdin** και τον **προορισμό** των **stdout/stderr**

STDOUT REDIRECTION

- Μπορούμε να στείλουμε το **stdout** σε κάποιο αρχείο αντί για την οθόνη με το **σύμβολο >**

```
[me@linuxbox ~]$ ls -l /usr/bin > ls-output.txt
```

- Αν όμως η εντολή παράγει ένα μήνυμα λάθους τότε αυτό εμφανίζεται στην οθόνη

```
[me@linuxbox ~]$ ls -l /bin/usr > ls-output.txt
ls: cannot access /bin/usr: No such file or directory
```

- Εν προκειμένω παράγεται και ένα κενό αρχείο **ls-output.txt**, καθώς το **σύμβολο >** αντικαθιστά τα περιεχόμενα του αρχείου με το stdout, που στην περίπτωσή μας είναι κενό
- Αν θέλουμε να κάνουμε append στο τέλος του αρχείου χρησιμοποιούμε το **σύμβολο »**

```
[me@linuxbox ~]$ ls -l /usr/bin >> ls-output.txt
```

STDERR REDIRECTION

- Για να ανακατευθύνουμε το stderr πρέπει να το δηλώσουμε
- Το **shell** έχει αριθμήσει τα **stdin, stdout, stderr** με την σειρά **0,1,2**
- Οπότε η εντολή που ακολουθεί στέλνει το **stderr** στο αρχείο **ls-error.txt**

```
[me@linuxbox ~]$ ls -l /bin/usr 2> ls-error.txt
```

- Για να στείλουμε **ταυτόχρονα** και το **stdout** και το **stderr** μπορούμε να εκτελέσουμε

```
[me@linuxbox ~]$ ls -l /bin/usr > ls-output.txt 2>&1
```

- Μία άλλη εναλλακτική είναι το

```
[me@linuxbox ~]$ ls -l /bin/usr &> ls-output.txt
```

ΑΠΟΡΡΙΨΗ STDOUT/STDIN

- Υπάρχουν περιπτώσεις στις οποίες θέλουμε να μην εμφανιστούν πουθενά τα αποτελέσματα ή τα λάθη μίας εντολής
- Στο **linux** υπάρχει ένα ειδικό αρχείο που χρησιμοποιούμε για αυτό το σκοπό το **/dev/null** που ονομάζεται και **bit bucket**

```
[me@linuxbox ~]$ ls -l /bin/usr 2> /dev/null
```

Η ΕΝΤΟΛΗ CAT

- Με την εντολή **cat** στέλνουμε το περιεχόμενο του **stdin** στο **stdout**
- Σε περίπτωση που δώσουμε ως ορίσματα ένα ή περισσότερα αρχεία τότε τα περιεχόμενά τους συγχωνευόνται και στέλνονται στο **stdout**
- Οπότε για να τυπώσουμε το περιεχόμενο ενός αρχείου :

```
[me@linuxbox ~]$ cat ls-output.txt
```

- και για να συγχωνεύσουμε πολλά αρχεία σε ένα :

```
cat movie.mpeg.0* > movie.mpeg
```

ΧΡΗΣΗ PIPELINES

- Με την χρήση του **συμβόλου |** μπορούμε να στείλουμε το **stdout** μίας εντολής στο **stdin** μίας άλλης

```
command1 | command2
```

- Για να δω τα περιεχόμενα του **/usr/bin** με την εντολή **less** :

```
[me@linuxbox ~]$ ls -l /usr/bin | less
```

- Το **σύμβολο |** χρησιμοποιείται για να στείλουμε το **stdout** σε μία άλλη εντολή ενώ στο **σύμβολο >** χρησιμοποιείται για να στείλουμε το **stdout** σε ένα αρχείο

```
command1 > file1
command1 | command2
```

ΛΑΘΟΣ ΧΡΗΣΗ PIPELINES

- Ένα σύνηθες λάθος είναι :

```
command1 > command2
```

- Τι θα κάνουν οι παρακάτω εντολές ?

```
# cd /usr/bin
# ls > less
```

ΠΑΡΑΔΕΙΓΜΑΤΑ PIPELINES

- Συνήθως προσπαθούμε να αλλάξουμε κατά κάποιον τρόπο την έξοδο μίας εντολής
- Η παρακάτω εντολή ταξινομεί τα περιεχόμενα των δύο directories και μας τα παρουσιάζει με την εντολή **less**
- **ls /usr/bin | sort | less**
- Η εντολή **uniq** μετά από το **sort** εμφανίζει μόνο μία φορά κάθε γραμμή, έστω και αν αυτή εμφανίζεται παραπάνω από μία φορές
- ενώ με την χρήση του **διακόπτη -d** μας εμφανίζει μόνο τις γραμμές που εμφανίζονται πάνω από μία φορά

ΕΝΤΟΛΗ WC

- Με την εντολή wc μετράμε τις γραμμές τις λέξεις και τους χαρακτήρες του κειμένου

```
[me@linuxbox ~]$ wc ls-output.txt
 7902  64566 503634 ls-output.txt
```

- Με την χρήση των διακοπτών l w m μετράμε αντίστοιχα γραμμές, λέξεις και χαρακτήρες

ΕΝΤΟΛΗ GREP

- Με την εντολή grep αναζητούμε ένα pattern μέσα σε ένα αρχείο και τυπώνουμε τις γραμμές που το περιέχουν

```
grep pattern [file...]
```

- Αναζήτηση όλων των προγραμμάτων που περιέχουν την λέξη zip

```
[me@linuxbox ~]$ ls /bin /usr/bin | sort | uniq | grep zip
```

ΕΝΤΟΛΕΣ HEAD ΚΑΙ TAIL

- Με τις **εντολές head/tail** εκτυπώνουμε τις πρώτες/τελευταίες 10 γραμμές ενός αρχείου
- Με την χρήση του **διακόπτη -n** επιλέγουμε τον αριθμό γραμμών που θέλουμε να εμφανιστεί

```
[me@linuxbox ~]$ head -n 5 ls-output.txt
total 343496
-rwxr-xr-x 1 root root      31316 2007-12-05 08:58 [
-rwxr-xr-x 1 root root       8240 2007-12-09 13:39 411toppm
-rwxr-xr-x 1 root root     111276 2007-11-26 14:27 a2p
-rwxr-xr-x 1 root root      25368 2006-10-06 20:16 a52dec
[me@linuxbox ~]$ tail -n 5 ls-output.txt
-rwxr-xr-x 1 root root       5234 2007-06-27 10:56 znew
-rwxr-xr-x 1 root root        691 2005-09-10 04:21 zonetab2pot.py
-rw-r--r-- 1 root root        930 2007-11-01 12:23 zonetab2pot.pyc
-rw-r--r-- 1 root root        930 2007-11-01 12:23 zonetab2pot.pyo
lrwxrwxrwx 1 root root          6 2008-01-31 05:22 zsoelim -> soelim
```

- Με την χρήση του **διακόπτη -f** στην tail τυπώνονται οι τελευταίες γραμμές του αρχείου live

ΕΝΤΟΛΗ TEE

- Με την **εντολή tee** μπορούμε να τυπώσουμε ένα κείμενο αλλά ταυτόχρονα να το στείλουμε και σε μία άλλη εντολή

```
[me@linuxbox ~]$ ls /usr/bin | tee ls.txt | grep zip
bunzip2
bzip2
gunzip
gzip
unzip
zip
zipcloak
zipgrep
zipinfo
zipnote
zipsplit
```

ΕΝΤΟΛΗ SED

- Η εντολή sed είναι ένα ευέλικτο εργαλείο το οποίο επιτρέπει την επεξεργασία κειμένου

```
[me@linuxbox ~]$ echo "front" | sed 's/front/back/'
back
```

- Η παραπάνω εντολή αντικαθιστά την λέξη front με την λέξη back
- Μπορείτε να ορίσετε τις γραμμές για τις οποίες θα εκτελεσθεί η sed

```
[me@linuxbox ~]$ echo "front" | sed '1s/front/back/'
back
```

ΟΡΙΣΜΟΣ ΓΡΑΜΜΩΝ ΓΙΑ ΤΗΝ ΕΝΤΟΛΗ SED

Address	Description
n	A line number where *n* is a positive integer.
$	The last line.
/regexp/	Lines matching a POSIX basic regular expression. Note that the regular expression is delimited by slash characters. Optionally, the regular expression may be delimited by an alternate character, by specifying the expression with *\cregexpc*, where *c* is the alternate character.
addr1,addr2	A range of lines from *addr1* to *addr2*, inclusive. Addresses may be any of the single address forms above.
first~step	Match the line represented by the number *first*, then each subsequent line at *step* intervals. For example 1~2 refers to each odd numbered line, 5~5 refers to the fifth line and every fifth line thereafter.
addr1,+n	Match *addr1* and the following *n* lines.
addr!	Match all lines except *addr*, which may be any of the forms above.

ΠΙΘΑΝΑ COMMANDS ΓΙΑ ΤΗΝ ΕΝΤΟΛΗ SED

`=`	Output current line number.
`a`	Append text after the current line.
`d`	Delete the current line.
`i`	Insert text in front of the current line.
`p`	Print the current line. By default, `sed` prints every line and only edits lines that match a specified address within the file. The default behavior can be overridden by specifying the `-n` option.
`q`	Exit `sed` without processing any more lines. If the `-n` option is not specified, output the current line.
`Q`	Exit `sed` without processing any more lines.
`s/regexp/replacement/`	Substitute the contents of *replacement* wherever *regexp* is found. *replacement* may include the special character `&`, which is equivalent to the text matched by *regexp*. In addition, *replacement* may include the sequences `\1` through `\9`, which are the contents of the corresponding subexpressions in *regexp*. For more about this, see the discussion of *back references* below. After the trailing slash following *replacement*, an optional flag may be specified to modify the `s` command's behavior.
`y/set1/set2`	Perform transliteration by converting characters from *set1* to the corresponding characters in *set2*. Note that unlike `tr`, `sed` requires that both sets be of the same length.

ΕΠΑΝΑΛΗΠΤΙΚΗ ΕΚΤΕΛΕΣΗ SED

- Η εντολή sed αντικαθιστά μόνο την πρώτη εμφάνιση σε κάθε γραμμή

```
[me@linuxbox ~]$ echo "aaabbbccc" | sed 's/b/B/'
aaaBbbccc
```

- Με την χρήση της παραμέτρου g μπορείτε να την υποχρεώσετε να το κάνει για όλες τις εμφανίσεις

```
[me@linuxbox ~]$ echo "aaabbbccc" | sed 's/b/B/g'
aaaBBBccc
```