

ΕΠΛ232 – Προγραμματιστικές Τεχνικές και Εργαλεία

Εργαλεία UNIX II (Κεφάλαια 1-4 & 7, DAS-2ED)

Τμήμα Πληροφορικής, Πανεπιστήμιο Κύπρου

<http://www.cs.ucy.ac.cy/courses/EPL232>

Περιεχόμενο Διάλεξης

- **Εντολές UNIX** (*echo, touch*),
- **Ιδιοκτησία και Δικαιώματα Πρόσβασης** (*chmod, chgrp, chown, umask, suid, sgid, sticky bit*),
- **Προσθήκη Χρηστών** (*useradd & public/private keys*)
- **Κανονικές Εκφράσεις** (*grep, egrep*)



Απλές Εντολές UNIX

- Εντολή *echo* (επιλογές *-n*, *-e*)
 - εκτύπωση γραμμής κειμένου που δίνεται μαζί με newline
 - **Επιλογή *-n***
 - δεν εμφανίζει τη νέα γραμμή (newline)
 - **Επιλογή *-e***
 - επιτρέπει την ερμηνεία των backslash escapes

```
bash-3.1$ echo -e "This is a \nnew message"  
This is a  
new message
```

π.χ, \t, \n, κλπ



Απλές Εντολές UNIX

ls - list directory contents
-l use a long listing format

- Εντολή **touch** *<existing_filename>*
 - Ο χρόνος τροποποίησης του αρχείου ενημερώνεται με τον υφιστάμενο χρόνο.

```
bash-3.1$ ls -l test/test2/test2.txt
-rw-r--r-- 1 cspgcc1 cspg 18 Jan 24 10:42 test/test2/test2.txt
bash-3.1$
bash-3.1$
bash-3.1$ touch test/test2/test2.txt
bash-3.1$ ls -l test/test2/test2.txt
-rw-r--r-- 1 cspgcc1 cspg 18 Jan 30 17:14 test/test2/test2.txt
```

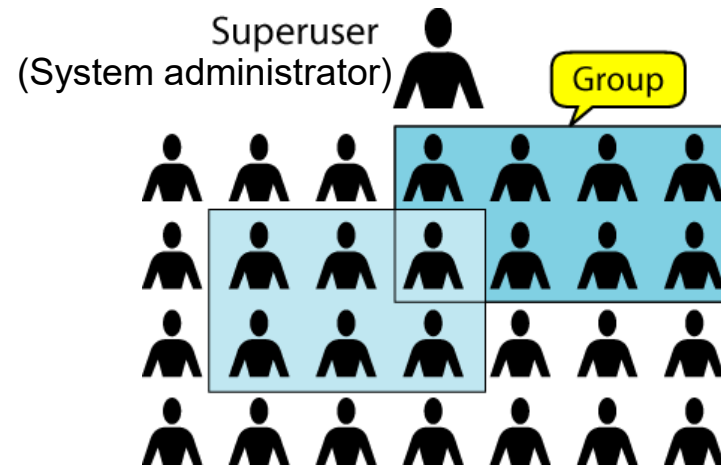
- Εντολή **touch** *<new_filename>*
 - Δημιουργεί ένα νέο, κενό αρχείο

```
bash-3.1$ ls -l new_file.txt
ls: new_file.txt: No such file or directory
bash-3.1$ touch new_file.txt
bash-3.1$ ls -l new_file.txt
-rw-r--r-- 1 cspgcc1 cspg 0 Jan 30 17:19 new_file.txt
```



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- **Χρήστης (User)**: οποιοσδήποτε έχει λογαριασμό στο UNIX σύστημα
- Οι χρήστες οργανώνονται σε **Ομάδες (Groups)**.
 - ένας ή περισσότεροι χρήστες μπορούν να ανήκουν σε **πολλαπλές ομάδες**.



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Για να βρεις πληροφορίες σε ποια/ες ομάδα/ες ανήκει ένας χρήστης:
 - Εντολή *groups* `<username>`

```
bash-3.1$ groups andarist
```

```
andarist : faculty ep1232
```

```
bash-3.1$ groups cchrys
```

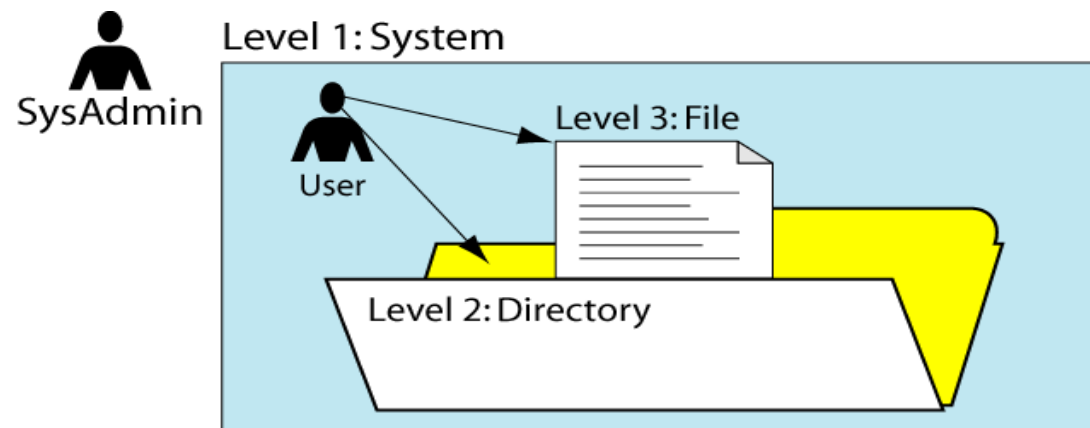
```
cchrys : tspecial ep1001 csphd visiting ep1371
```

- Σημείωση: Στο UNIX, κάθε χρήστης **ΠΡΕΠΕΙ** να είναι μέλος **τουλάχιστο μιας ομάδας** (αυτή που ορίζεται από το GID μέσα στο `/etc/passwd`)



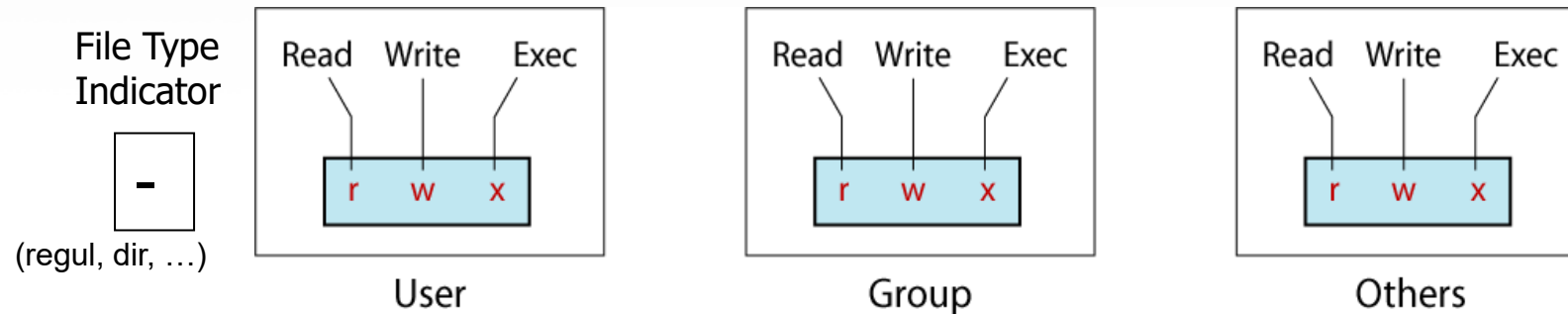
Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Επίπεδα Ασφάλειας:
 - Σύστημα, Κατάλογος, Αρχείο
 - Ασφάλεια Συστήματος: ελέγχεται από τον διαχειριστή του συστήματος (system administrator)
 - Κατάλογος και Αρχείο: ελέγχεται από το χρήστη στον οποίο ανήκει



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- **Κώδικας Δικαιωμάτων Πρόσβασης***



- **User (Χρήστης-Ιδιοκτήτης):** Ο δημιουργός του αρχείου
 - **Group (Ομάδα):** Σειρά από χρήστες που ομαδοποιούνται
 - **Others (Υπόλοιποι):** Οποιοσδήποτε λογαριασμός που δεν ανήκει στην Ομάδα αλλά ανήκει σε άλλη ομάδα
 - Τρεις τύποι δικαιωμάτων πρόσβασης:
 - **r** read
 - **w** write
 - **x** execute
 - **-** permission denied
- * `ls -al <file | directory>`



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Τι υποδηλώνει το κάθε είδος πρόσβασης;

Τύπος Πρόσβασης	Σημασία για <u>Αρχείο</u>	Σημασία για <u>Κατάλογο</u>
r (read)	View file contents	List directory contents
w (write)	Change file contents	- Change directory contents (create and remove files in that dir.)
x (execute)	Run executable file	- Access files explicitly (by name) in the given folder
-	Permission denied	Permission denied



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Δείκτης Τύπου Αρχείου

Χαρακτήρας	Δείκτης Τύπου Αρχείου
d	D irectory
b	B lock-type special file (π.χ., DVD, CDROM, DISK)
c	C haracter-type special file (π.χ., terminals, printers και networks)
l	S ymbolic L ink (A symbolic link is a reference to another file)
p	P ipe (connect the output of one process to the input of another)
s	S ocket (enables communication between two processes)
-	R egular file



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Ελέγχοντας τα Δικαιώματα:
 - Εντολή **ls -l <file_OR_dir_name>**

```
bash-3.1$ ls -lR test/ | head
test/:
total 4
-rw-r--r-- 2 cspgcc1 cspg 28 Jan 25 14:35 HardLinkToFile1.txt
lrwxrwxrwx 1 cspgcc1 cspg 20 Jan 24 10:50 SymbLinkToFile2.txt -> test/test2/test2.txt
drwxr-xr-x 4 cspgcc1 cspg 50 Jan 28 17:25 test1
drwxr-xr-x 2 cspgcc1 cspg 22 Jan 24 10:42 test2
```

recursive listing

total size of all files in blocks in the list (measured in 512B or 1024B)

Permissions, Hard-Links, User, Group, FileSize, Last Modified Date + Time, Filename

Σημείωση: Η εντολή head μπορεί να γίνει pipe σε άλλες εντολές. Στο πιο πάνω παράδειγμα, το αποτέλεσμα της εντολής LS είναι να εμφανίσει μόνο τα πιο πρόσφατα τροποποιημένα αρχεία ή φακέλους



Ιδιοκτησία και Δικαιώματα Πρόσβασης

```
$ ls -l /etc/passwd
```

```
-rw-r--r-- ... /etc/passwd → Regular file
```

```
$ ls -dl /
```

```
drwxr-xr-x 26 root root 4096 Sep 22 09:29 / → Directory
```

```
lrwxrwxrwx ... termcap -> /usr/share/misc/termcap → Symbolic Link
```

```
lrwxrwxrwx ... S03xinetd -> ../init.d/xinetd
```

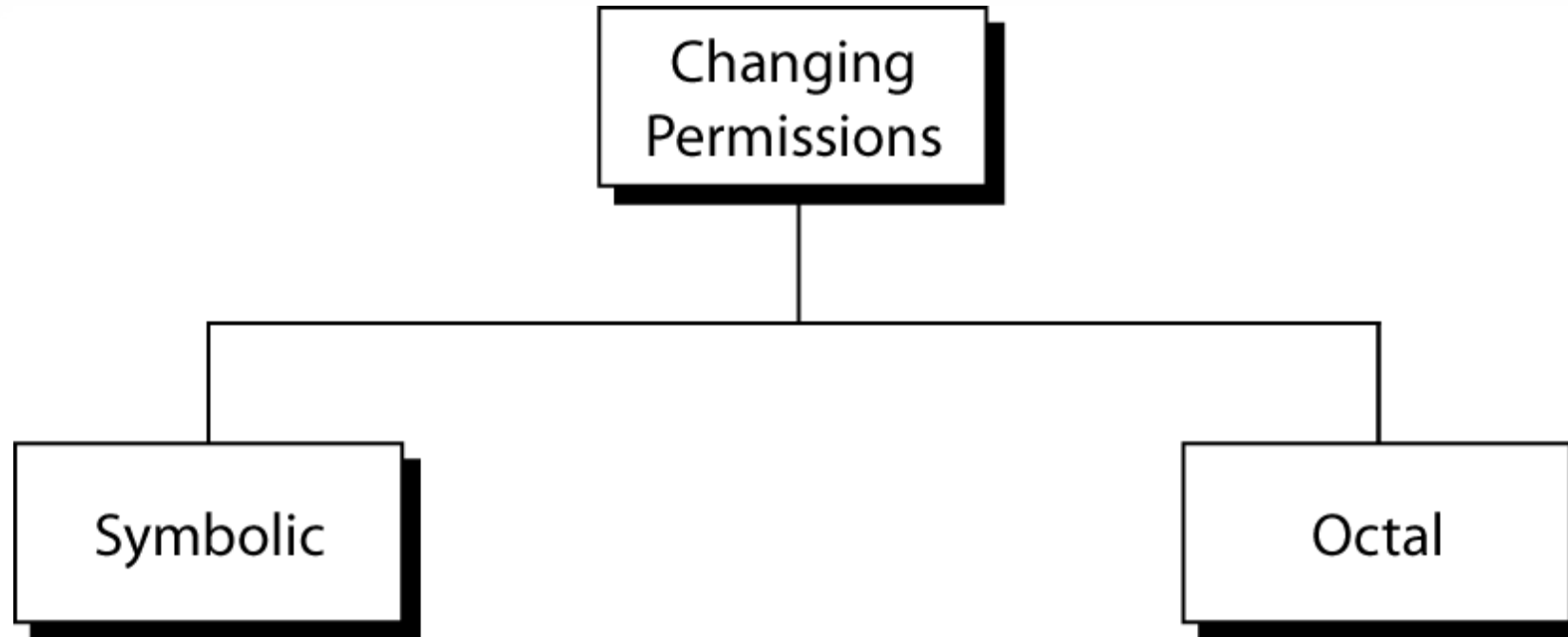
```
srwxrwxrwx /tmp/.X11-unix/X0 → Socket
```

```
prw-rw---- ... mypipe → Pipe
```



Ιδιοκτησία και Δικαιώματα Πρόσβασης

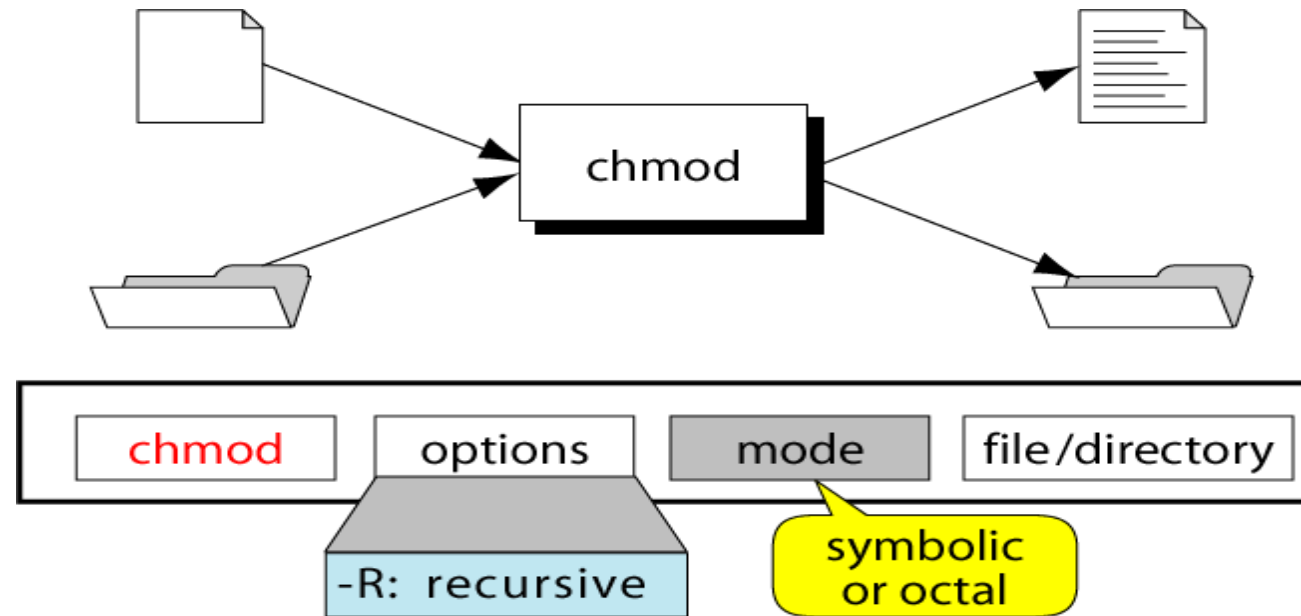
- Αλλαγή Δικαιωμάτων:



Ιδιοκτησία και Δικαιώματα Πρόσβασης

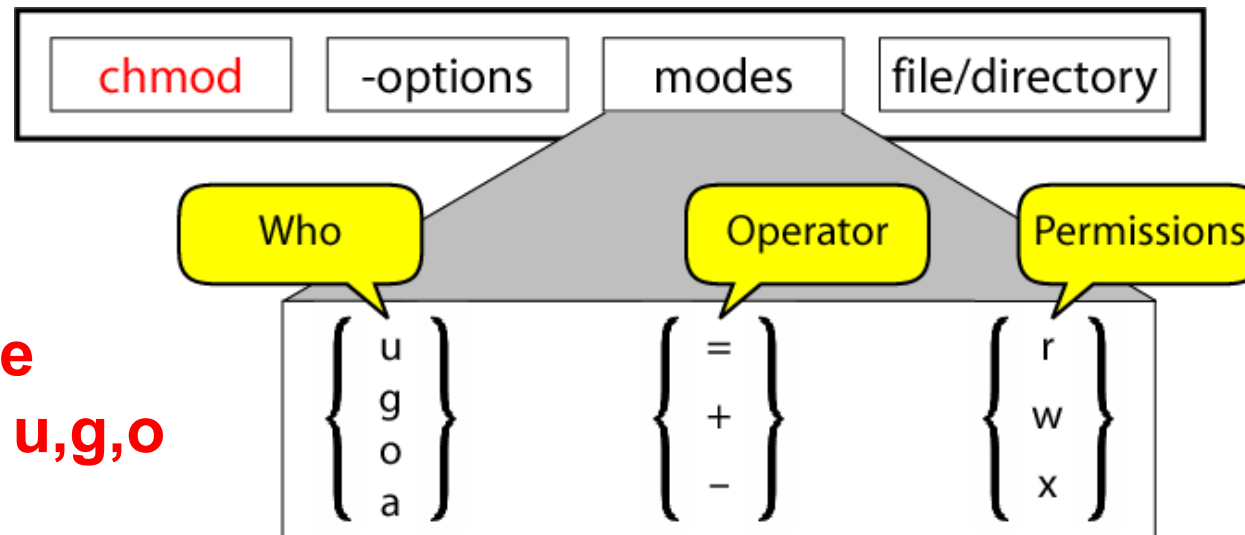
- Αλλαγή Δικαιωμάτων:

- Εντολή **chmod** → **μόνον ο ιδιοκτήτης (ο su)** (και άλλα μέλη της ομάδας κάτω από ορθά δικαιώματα μπορούν να το πράξουν!)



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Αλλαγή Δικαιωμάτων: Συμβολική μορφή
 - Εντολή *chmod*



Chmod +x file
> Adds +x to u,g,o

Example

```
chmod u=rwx,g+w,o-w memo.doc
```



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- **Αλλαγή Δικαιωμάτων: Συμβολική μορφή**
 - Εντολή *chmod*

chmod who operation permissions filename

u for user
g for group
o for others
a for all

+ for add
- for remove
= for assign
(set)

r for read
w for write
x for execute



chmod +x

```
$touch testfile
```

```
$ls -al testfile
```

```
-rw-r--r-- 1 andarist staff 0 Feb 14 11:27 testfile
```

```
$chmod +x testfile
```

```
$ls -al testfile
```

```
-rwxr-xr-x 1 andarist staff 0 Feb 14 11:27 testfile
```

```
$chmod = testfile
```

```
$ ls -al testfile
```

```
----- 1 andarist staff 0 Feb 14 11:27 testfile
```

```
$chmod 777 testfile
```

```
$ ls -al testfile
```

```
-rwxrwxrwx 1 andarist staff 0 Feb 14 11:27 testfile
```



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Αλλαγή Δικαιωμάτων: Συμβολική μορφή

- Παράδειγμα:

```
bash-3.1$ ls -l test.txt  
-rw-r--r-- 1 cspgcc1 cspg 0 Jan 30 19:38 test.txt
```

Αλλαγή των δικαιωμάτων πρόσβασης έτσι ώστε **όλοι να μπορούν να το διαβάζουν** και να το **εκτελούν** και **μόνο ο ιδιοκτήτης** και η **ομάδα** να **μπορούν να γράφουν** σ' αυτό (**rwX | rwX | r-X**):

```
bash-3.1$ chmod ug=rwx,o+x test.txt  
bash-3.1$ ls -l test.txt  
-rwxrwxr-x 1 cspgcc1 cspg 0 Jan 30 19:40 test.txt
```

- `chmod o= test.txt` → αφαιρεί τα δικαιώματα από O group



Ιδιοκτησία και Δικαιώματα Πρόσβασης

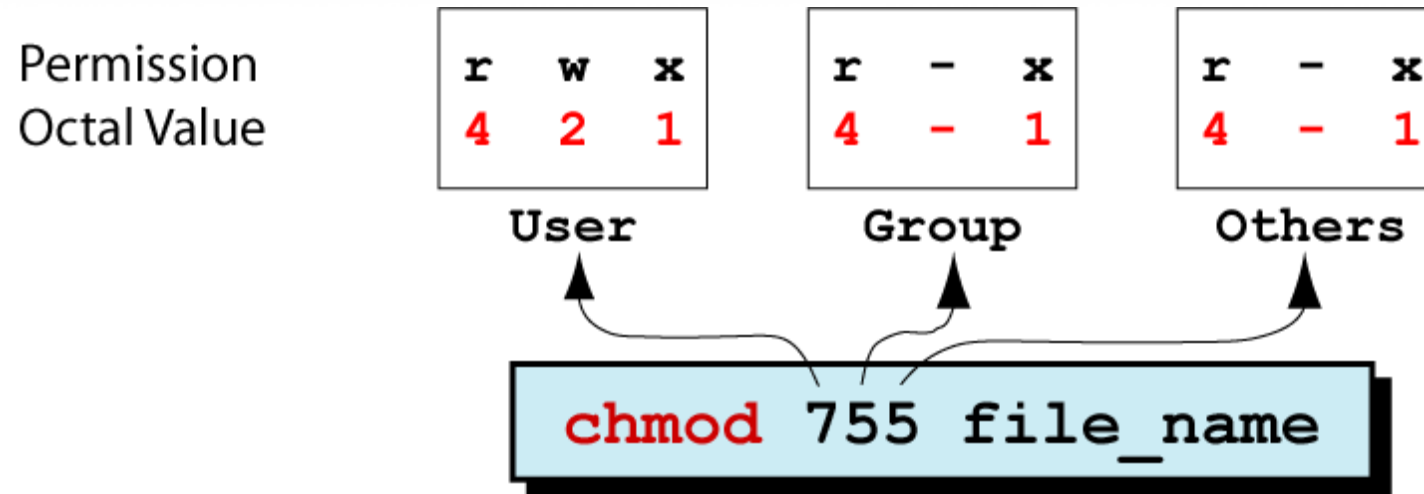
- Αλλαγή Δικαιωμάτων: Οκταδική μορφή

Βήμα	Εκτέλεση	Ρυθμίσεις
1	List the desired setting	<code>rwX rwX r-X</code>
2	Assign binary: 1 for access; 0 for no access	<code>111 111 101</code>
3	List octal values for the corresponding binary 1's	<code>421 421 401</code>
4	Convert the octal values to a 3-digit number	<code>775</code>
5	Write the command	<code>chmod 775 filename</code>



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Αλλαγή Δικαιωμάτων: Οκταδική μορφή



Πλήρεις δικαιώματα στον ιδιοκτήτη, *read* και *execute* στους υπόλοιπους

$rwX \ r-x \ r-x = 111 \ 101 \ 101 = 755$

read και *write* στον ιδιοκτήτη, και *read* μόνο στους υπόλοιπους

$rw- \ r-- \ r-- = 110 \ 100 \ 100 = 644$



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Αλλαγή Ιδιοκτησίας
 - Εντολή **chown** *<new_owner> <filename>*
 - Αλλαγή ιδιοκτησίας ενός αρχείου → **μόνον ο ιδιοκτήτης (και ο su) μπορεί να το πράξει!**
 - Με την αλλαγή, ο νέος ιδιοκτήτης είναι ο μόνος που μπορεί να δώσει τα δικαιώματα πίσω (και ο su).

chown root /directory

Change the owner of /directory to "root". The group of /directory is changed to root's default group (i.e., root).

chown root:staff /directory

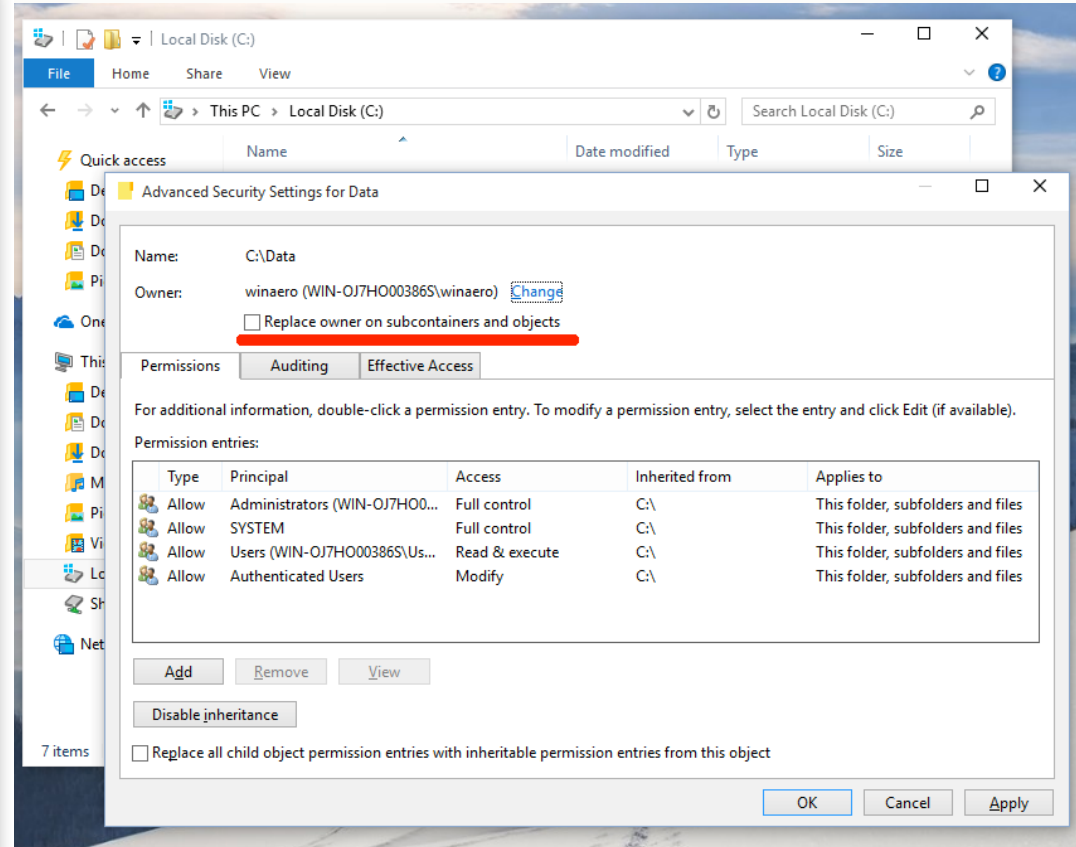
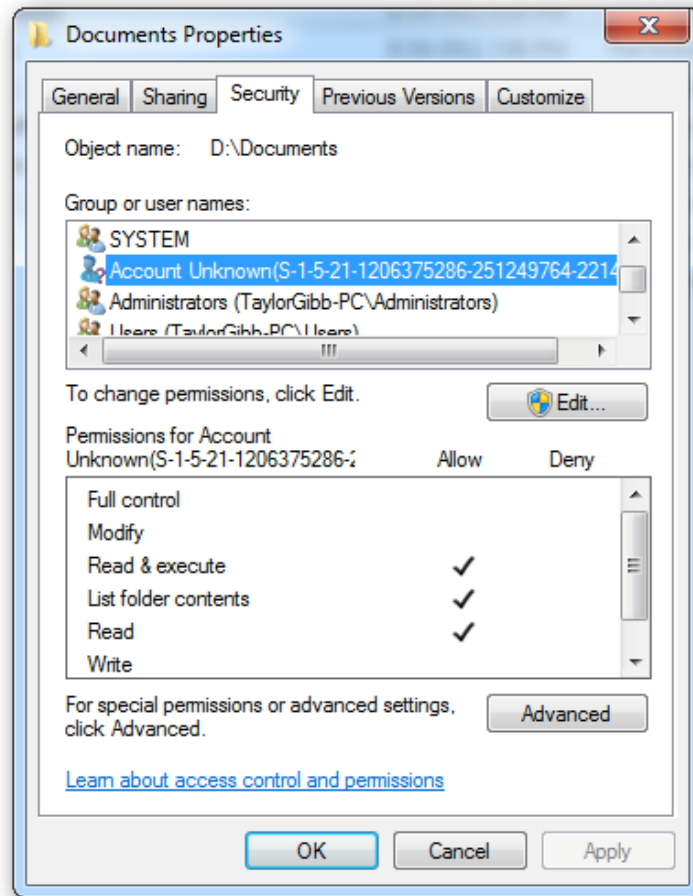
Likewise, but explicitly change the group of /directory to "staff". (recall that a user might belong to several groups)

chown -hR root /directory

Change the owner of /directory **recursively (-R)** to "root" (including the **traversal of symbolic links. Used (-h) to exclude them.**)



Fixing Orphaned Users Taking Ownership in Windows



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Αλλαγή Ομάδας

- Εντολή **chgrp** *<new_group>* *<filename>*

- Αλλαγή ομάδας (effective group) ενός αρχείου σε μια άλλη ομάδα στην οποία ανήκει ο χρήστης
 - Μπορεί να επιτευχθεί και με την chown

```
bash-3.1$ ls -l test.c
-rw-r--r-- 1 cchrys tspecial 55 Mar 17 2015 test.c
bash-3.1$ groups cchrys
cchrys : tspecial ep1001 csphd visiting ep1371
bash-3.1$ chgrp visiting test.c
bash-3.1$ ls -l test.c
-rw-r--r-- 1 cchrys visiting 55 Mar 17 2015 test.c
bash-3.1$ groups cchrys
cchrys : tspecial ep1001 csphd visiting ep1371
```

Δείτε Επόμενη Διαφάνεια για βασική & συμπληρωματική ομάδα!



Primary / Supplementary Groups (groups, id)

- Ανά πάσα στιγμή, ένας UNIX χρήστης φέρει ένα user id (uid) και ένα group id (gid).
- Παράλληλα, ένας χρήστης μπορεί να ανήκει και σε άλλα groups, το πρώτο (default) εκ των οποίων ονομάζεται **βασική ομάδα (Primary Group)**, ενώ τα υπόλοιπα **συμπληρωματικές ομάδες (Supplementary Groups)**

\$ groups # print the groups a user is in

```
faculty ep1371 ep1132 anyplacegrp ep1646 colloqgrp crowdgrp smartpgrp smartgrp smartlgrp  
smartnet tvmgrp
```

\$ id # print group names and their group IDs

```
uid=1280(andarist) gid=231(faculty)  
groups=231(faculty),306(ep1371),314(ep1232),348(anyplacegrp),411(ep1646),426(colloqgrp),  
446(crowdgrp),453(smartpgrp),466(smartgrp),483(smartlgrp),488(smartnet),505(tvmgrp)
```

\$ newgrp ep1371 # αλλαγή βασικής ομάδας

```
uid=1280(andarist) gid=306(ep1371)  
groups=231(faculty),306(ep1371),314(ep1232),348(anyplacegrp),411(ep1646),426(colloqgrp),  
446(crowdgrp),453(smartpgrp),466(smartgrp),483(smartlgrp),488(smartnet),505(tvmgrp)
```



Άντληση Ταυτοτήτων από Εξυπηρετητή Ταυτοποίησης LDAP - Εντολή getent

Παρουσίαση χρηστών (αντίστοιχο του /etc/passwd)

\$getent passwd (get the entries)

```
aandre28:*:2923:472:Andreou Andre:/home/students/cs/2011/aandre28:/bin/bash
acosta01:*:2776:472:Andri Costa:/home/students/cs/2011/acosta01:/bin/bash
ageorg35:*:2743:472:Anna Georgiou:/home/students/cs/2011/ageorg35:/bin/bash
```

						Login shell
					Home directory	
				GECOS fields (full name, etc.)		
			Primary Group id			
		User ID				
	Encrypted password indicator					

Username

Παρουσίαση στοιχείων ομάδας (αντίστοιχο του /etc/groups)

\$getent group cs11

```
cs11:*:472:
```



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Δικαιώματα Πρόσβασης αρχείων και καταλόγων κατά τη δημιουργία τους
 - Εντολή *umask* (Ορίζει τι δικαιώματα αφαιρούνται)
 - Τα εξ' ορισμού (προεπιλεγμένα) δικαιώματα ενός δημιουργηθέντος αρχείου ή καταλόγου ρυθμίζονται αρχικά χρησιμοποιώντας μια μεταβλητή **3-ψηφίων** σε οκταδικό σύστημα, που ονομάζεται *user mask*.
- Αυτό το *user mask* έχει ορισθεί από τον *system administrator* όταν δημιουργήθηκε ο λογαριασμός του κάθε χρήστη.
- Το *user mask* περιέχει τις ρυθμίσεις σε οκταδικό για τα δικαιώματα πρόσβασης που **ΑΦΑΙΡΟΥΝΤΑΙ** από το μέγιστο όταν ένας **κατάλογος** ή **αρχείο** δημιουργείται.



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Μέγιστα Δικαιώματα **κατά την δημιουργία**:
 - **Καταλόγου**: **777** (δηλ., rwxrwxrwx)
 - **Αρχείου**: **666** (δηλ., rw-rw-rw-)
 - Π.χ., με “touch a” τα δικαιώματα του a είναι “666 – umask”
- Γιατί 666 σε αρχεία και 777 σε καταλόγους;
 - Το UNIX προσπαθεί να συμπεριφέρεται έξυπνα όσον αφορά τα **execute δικαίωμα**.
 - Πρακτικά τα αρχεία θεωρούνται ότι δεν έχουν ΠΟΤΕ execute δικαιώματα κατά τη δημιουργία τους.
 - Αφαιρώντας εξ 'ορισμού το execute σε αρχεία επιτρέπει το σύστημα να έχει περισσότερη ασφάλεια.
 - Το execute σε καταλόγους δεν αφαιρείται εφόσον μπορεί να θέλουμε να παρέχουμε access by file name



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Παράδειγμα

```
bash-3.1$ umask
```

```
0022
```

→ Το επιπλέον 0 θα εξηγηθεί σε λίγο

```
bash-3.1$ touch test.txt
```

```
bash-3.1$ ls -l test.txt
```

```
-rw-r--r-- 1 cchrys tspecial 0 Jan 31 06:27 test.txt
```

```
bash-3.1$ mkdir test-perm
```

```
bash-3.1$ ls -ld test-perm/
```

```
drwxr-xr-x 2 cchrys tspecial 4096 Jan 31 06:28 test-perm/
```

File

Directory

666-022 = 644

777-022=755



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Παράδειγμα (συνέχεια)

```
bash-3.1$ umask 077 (group & others: NO permissions)
```

```
bash-3.1$ touch test2.txt → File: 666 - 077 = 600
```

```
bash-3.1$ ls -l test2.txt
```

```
-rw----- 1 cchrys tspecial 0 Jan 31 06:59 test2.txt
```

```
bash-3.1$ mkdir test2-perm → Directory: 777 - 077 = 700
```

```
bash-3.1$ ls -ld test2-perm (-d:show dir not content)
```

```
drwx----- 2 cchrys tspecial 4096 Jan 31 07:00 test2-perm
```

- Το **umask** ρυθμίζεται μια φορά και ισχύει μέχρι την αποσύνδεση του **session**.
 - Κάθε φορά που συνδεόμαστε (log in) στο σύστημα, το **umask** κρατά την προεπιλεγμένη του τιμή (αργότερα θα μιλήσουμε για το προφίλ)



Ειδικά Δικαιώματα Πρόσβασης

- Σε πολλές περιπτώσεις προκύπτει η ανάγκη για προσωρινά αναβαθμισμένα δικαιώματα.
 - Π.χ., ο χρήστης `andarist:faculty` να έχει δικαιώματα `root:root` κατά την εκτέλεση την εντολής `/usr/bin/passwd`
- Υπάρχουν οι ακόλουθοι τρόποι:
 1. Login as <newuser> (π.χ., `su - newuser`)
 2. Run command with sudo (π.χ., `sudo -u newuser /usr/bin/passwd`, requires the existence of configurations in `/etc/sudoers/`)
 3. Special Permissions on Executable (Ειδικά Δικαιώματα)
 - **Set User ID (SUID)** -- για εκτελέσιμα αρχεία
 - **Set Group ID (SGID)** -- για εκτελέσιμα αρχεία
 - **Sticky bit (STB)** -- για καταλόγους

Ειδικά Δικαιώματα Πρόσβασης

Παράδειγμα

- Συχνά, όταν εκτελείται μια εντολή από τον χρήστη, θα πρέπει να εκτελεστεί με ειδικά προνόμια για να ολοκληρώσει την εργασία της.
- Για παράδειγμα, όταν αλλάζετε τον κωδικό πρόσβασής σας με το **passwd** command, ο νέος σας κωδικός πρόσβασης αποθηκεύεται στο αρχείο **/etc/shadow**.
- Ως κανονικός χρήστης, δεν έχετε **read** ή **write** πρόσβαση σε αυτό το αρχείο για λόγους ασφαλείας, αλλά όταν αλλάζετε τον κωδικό πρόσβασής σας, πρέπει να έχετε δικαιώματα εγγραφής σε αυτό το αρχείο. Αυτό σημαίνει ότι το πρόγραμμα **passwd** πρέπει να σας δώσει πρόσθετα δικαιώματα, ώστε να μπορείτε να γράψετε στο αρχείο **/etc/shadow**.
- Επιπλέον δικαιώματα δίνονται σε προγράμματα μέσω ενός μηχανισμού γνωστού ως **Set User ID (SUID)** και **Set Group ID (SGID)** bits.
- Όταν εκτελείτε ένα πρόγραμμα που έχει το SUID bit enabled, κληρονομείτε τα δικαιώματα του κατόχου αυτού του προγράμματος. Προγράμματα που δεν έχουν το SUID bit ενεργό εκτελούνται με τα δικαιώματα του χρήστη που ξεκίνησε το πρόγραμμα.
- Αυτό συμβαίνει και με το SGID.
- Τα SUID και SGID bits εμφανίζονται με το γράμμα "s" αν η πρόσβαση είναι εγκεκριμένη.



Ειδικά Δικαιώματα Πρόσβασης: Αρχεία

- Ρύθμιση Ειδικών Δικαιωμάτων
 - Χρήση της εντολής *chmod* σε οκταδική μορφή:
 - π.χ. *chmod 7777 filename*

*Ενδέχεται να είναι
και στο τέλος*

suid	sgid	stb	r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1	4	2	1
7			7			7			7		
Special			user			group			others		



Ειδικά Δικαιώματα Πρόσβασης: Αρχεία

- Ειδικά Δικαιώματα: **Set User ID (SUID)**

- **SUID** επιτρέπει στους χρήστες να εκτελέσουν ένα αρχείο και να γίνουν **προσωρινοί ιδιοκτήτες** (Owners) του αρχείου (κατά τη διάρκεια της εκτέλεσης).

- Στο Linux / Unix αγνοείται για καταλόγους (μόνο για αρχεία)

- **Παράδειγμα:** Η εντολή *passwd* ή *ping* με ιδιοκτήτη τον *root* έχει τις ακόλουθες ειδικές ρυθμίσεις:

```
bash-3.1$ ls -l /usr/bin/passwd
```

```
-r-s--x--x 1 root root 21944 Feb 12 2006 /usr/bin/passwd
```

```
bash-3.1$ ls -al /bin/ping
```

```
-rwsr-xr-x 1 root root 40760 Sep 26 2013 /bin/ping
```

SUID

- Όταν ένας χρήστης εκτελεί την εντολή *passwd*, ο χρήστης γίνεται προσωρινά ο «*root*» χρήστης για **όσο τρέχει η εντολή** (δηλ., η διεργασία θα έχει τα **ίδια δικαιώματα** όπως αυτά του **ιδιοκτήτη** του αρχείου)

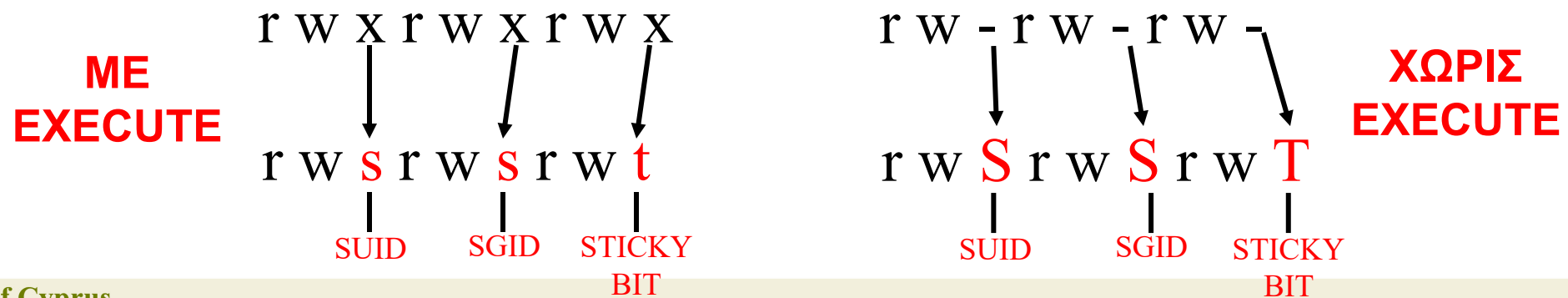
- Θα δούμε σε λίγο πως τίθεται η επιλογή «s»



Ιδιοκτησία και Δικαιώματα Πρόσβασης

• Παρουσίαση Ειδικών Δικαιωμάτων

- Η κατάσταση των δικαιωμάτων πρόσβασης που εμφανίζεται με την εντολή «*ls -l*» **δεν έχει ξεχωριστό τμήμα για τα ειδικά δικαιώματα σε πολλές υλοποιήσεις ☹️**.
- Επειδή τα ειδικά δικαιώματα απαιτούν συνήθως «*execute*», καλύπτουν/αντικαθιστούν το **δικαίωμα *execute*** στην παρουσίαση της εντολής «*ls -l*».



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Ειδικά Δικαιώματα: **Set Group ID (SGID)**
 - Η **SGID** κάνει τους χρήστες να **γίνουν μέλος της ομάδας** του γονικού καταλόγου
 - Χρήσιμο για δημιουργία κοινόχρηστης πρόσβασης σε άτομα από διαφορετικές ομάδες (π.χ., student, faculty)
 - Υπάρχει κάποια ομάδα “GALL” με μέλη τους A:G1, B:G2, C:G3 (username:group)
 - Ο κατάλογος /projects/ ανήκει στο GALL.
 - Ο A:G1 εκτελεί «touch file» (δημιουργία αρχείου)
 - Τώρα το **file** είναι του **A:G1** (δείτε ls -al file)
 - Ενώ εάν εκτελούσαμε το πιο κάτω πριν το touch:
 - chmod **2777** /projects/
 - Τότε το **file** θα είχε δικαιώματα **A:GALL**

Ιδιοκτησία και Δικαιώματα Πρόσβασης

Παράδειγμα SGID

```
$ls -ald anyplace/ # d shows directory not content
drwxrwsr-x 21 kgeorg10 anyplacegrp 4096 Jan  2 00:02 anyplace/
```

Πλέον, ότι αρχεία δημιουργούνται ανήκουν στο group anyplacegrp στο οποίο ανήκουν και οι δυο χρήστες (βολικό για αλλαγές και από τους δυο)

```
$ ls -al
total 180
drwsrwsr-x 21 kgeorg10 anyplacegrp 4096 Jan  2 00:02 .
drwxr-xr-x 68 root      root      8192 Sep 30 10:02 ..
drwsr-sr-x  2 andarist anyplacegrp 4096 Sep 19 14:57 architect
drwsr-sr-x  2 andarist anyplacegrp 4096 Sep 19 14:57 contact
-rwxrw-rw-  1 andarist anyplacegrp  181 Sep 18 13:55 contact.html
drwxrwxr-x  2 kgeorg10 anyplacegrp 4096 Nov 27 13:14 css
```



Ιδιοκτησία και Δικαιώματα Πρόσβασης

- Ειδικά Δικαιώματα: **Sticky Bit (STB)**

- Εάν εφαρμοστεί το *Sticky Bit* τότε η **διαγραφή αρχείων/καταλόγων** μπορεί να γίνει από **ένα χρήστη μόνο** σε αρχεία που έχει προσθέσει ο **ίδιος ο χρήστης**.

- Το *Sticky Bit* εκτελεί μια χρήσιμη λειτουργία στους καταλόγους, π.χ., στο /tmp (κοινόχρηστος χώρος)

- Παράδειγμα:

```
bash-3.1$ ls -ld /tmp
```

```
drwxrwxrwt 71 root root 16384 Jan 31 04:10 /tmp
```

→ Sticky Bit

- Εάν το /tmp ήταν απλά 777 τότε θα μπορούσε οποιοσδήποτε να διαγράψει καταλόγους/αρχεία άλλων χρηστών (όχι μόνο προσωπικούς) .



Κανονικές Εκφράσεις

- Μια **Κανονική Έκφραση (*Regular Expression*)** είναι ένα πρότυπο που περιγράφει ένα σύνολο συμβολοσειρών.
 - Σημαντική έννοια στα πλαίσια των **μεταγλωττιστών, θεωρίας υπολογισμού, αλλά και προγραμματιστικών γλωσσών.**
 - Ο βασικότερος τρόπος εκτέλεσης κανονικών εκφράσεων στο UNIX είναι με χρήση της εντολής **grep (ή egrep)**
 - Κανονικές εκφράσεις κατασκευάζονται χρησιμοποιώντας **μικρότερες εκφράσεις.**
- Στη Θεωρία Υπολογισμού είχαμε μάθει ότι οι κανονικές εκφράσεις (regular expressions, regexpr ή regex) χρησιμοποιούνται για την περιγραφή γλωσσών με απλά σύμβολα, το NULL και συνδυασμούς που προκύπτουν με εφαρμογή ένωσης (U), του αστεριού Κλήνου (Kleene Star) (*) ή και παρενθέσεων
 - Π.χ., $(a^*b^*)^*(abba)(a^*b^*)^*$

Εντολή *grep*

- Εντολή *grep* (επιλογές *-i*, *-n*, *-v*, *-w*, *-l*)
 - Ψάχνει σε αρχεία εισόδου για γραμμές που περιέχουν ένα πρότυπο που δίνεται και τις εμφανίζει στην έξοδο.

grep <options> <pattern> <filename>

• Παραδείγματα

bash-3.1\$ *grep -n root /etc/passwd*

-n option: show line number of matching lines
pattern
filename

```
1: root:x:0:0:root:/root:/bin/bash
12: operator:x:11:0:operator:/root:/sbin/nologin
```

-v option: invert match (show lines that don't contain the word bash)

```
bash-3.1$ grep -v bash /etc/passwd | grep -v nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
news:x:9:13:news:/etc/news:
```



Εντολή *grep*: Επιλογές (Options)

- ***grep* Options** (επιλογές *-i*, *-n*, *-v*, *-w*, *-l*)
 - **Επιλογή *-i* (*case-insensitive-match*)**
 - *Case-insensitive* - αγνοεί το διαχωρισμό κεφαλαίων και μικρών γραμμμάτων
 - **Επιλογή *-n* (*numeric-prefix*)**
 - εκτυπώνει την αρίθμηση των γραμμών στην έξοδο
 - **Επιλογή *-v* (*invert-match*)**
 - εμφανίζει τις γραμμές που ΔΕΝ ταιριάζουν με το πρότυπο που ψάχνουμε
 - **Επιλογή *-w* (*word-match*) – Ψάξιμο για λέξεις**
 - ταιριάζει το πρότυπο με ολοκληρωμένες λέξεις (δεν περιέχουν τους χαρακτήρες της τιμής περιβάλλοντος \$IFS, η οποία εξ' ορισμού περιέχει τα whitespace: newline / tab / space)
 - **Επιλογή *-l* (*latin-el*) (*filename-with-match*)**
 - Απόκρυψη της κανονικής ροής στο Stdout. Τύπωσε μόνο τα **ονόματα των αρχείων** (όχι τα αποτελέσματα) όπου γίνεται match



Εντολή *grep*: Κανονικές Εκφράσεις (Patterns)

- Η *grep* χειρίζεται δυο διαφορετικές εκδοχές **σύνταξης κανονικών εκφράσεων**: τις βασικές και εκτεταμένες.

- Οι **Βασικές κανονικές εκφράσεις (BRE)** μπορούν να χρησιμοποιηθούν από την εντολή *grep* (όμοιο με **GNU grep**)

- Όπως ορίστηκε από τον Ken Thompson στην αρχική υλοποίηση του για το UNIX.

- Οι **Εκτεταμένες Κανονικές Εκφράσεις (ERE)** μπορούν να χρησιμοποιηθούν από την εντολή *egrep*

- Π.χ., `egrep 'word1 | word2' filename`

- Ίδιο με το `grep -E 'word1|word2' filename`



Εντολή *grep*: Παραδείγματα

- Παραδείγματα (συνέχεια)

^ Match lines beginning with root

```
bash-3.1$ grep ^root /etc/passwd  
root:x:0:0:root:/root:/bin/bash
```

: Match lines ending with ":"

```
bash-3.1$ grep :$ /etc/passwd  
news:x:9:13:news:/etc/news:
```

\< means "Start of Word": Match lines that contain the word PATH.

```
bash-3.1$ grep export ~/.profile | grep '\<PATH'
```

```
export PATH
```

Any line containing export

```
#export PATH=$PATH:/usr/local/ns-allinone-2.1b6/bin
```

```
export PATH=$PATH:/hdd2/cchrys/ns-allinone-2.1b9/bin:/hdd2/cchrys/ns-allinone-  
2.1b9/tcl8.3.2/unix:/hdd2/cchrys/ns-allinone-2.1b9/tk8.3.2/unix
```

```
#export PATH=$PATH:/Backup/cchrys/OMNeTpp/omnetpp-2.0b5/bin
```



Εντολή *grep*: Παραδείγματα

- Παραδείγματα (συνέχεια)

Match lines containing word /

```
bash-3.1$ grep -w / /etc/fstab  
/dev/VolGroup00/LogVol100 /      ext3      defaults      1 1
```

[] Match any line that contains any of the characters yf

```
bash-3.1$ grep [yf] /etc/group
```

```
sys:x:3:root,bin,adm
```

```
tty:x:5:
```

*[^] Match any line that **does not** contain the following chars*

```
ftp:x:50:
```

```
nobody:x:99:
```

```
floppy:x:19:
```

```
nsnobody:x:65534:
```

```
xfs:x:43:
```



Παρένθεση περί Εισαγωγικών

- **Μονά εισαγωγικά (' ')**

- χρησιμοποιούνται για να διατηρήσουν τους χαρακτήρες που εμπερικλείονται σε μονά εισαγωγικά ως είναι (κυριολεκτικά)
- **Παράδειγμα**

```
bash-3.1$ echo '$HOME'  
$HOME
```

- **Διπλά Εισαγωγικά (" ")**

- χρησιμοποιούνται για να διατηρήσουν τους χαρακτήρες που εμπερικλείονται σε διπλά εισαγωγικά ως είναι (κυριολεκτικά), εκτός από το \$ (dollar sign), τα ` (backward single quotes) και το \ (backslash).
- **Παράδειγμα**

```
bash-3.1$ echo "$HOME"  
/home/faculty/andarist
```



Εντολή *grep*: Παραδείγματα

- Παραδείγματα (συνέχεια)

Case-insensitive matching of words beginning with i

```
bash-3.1$ ls -l | grep -i '\<i'
-rwx----- 1 cchrys tspecial 247605 Jan 12 2005 iasted1.pdf
-rwx----- 1 cchrys tspecial 180597 Jan 12 2005 iasted2.pdf
-rwx----- 1 cchrys tspecial 179801 Dec 30 2003 ICIS_chrysostomou.pdf
-rwx----- 1 cchrys tspecial 156920 Dec 30 2003 ICIS_pitsillides.pdf
drwxr-x--- 2 cchrys tspecial 4096 Aug 19 2002 idcc
drwxr-x--- 5 cchrys tspecial 4096 Jul 17 2001 IPE
```

Same but first filtering out directories (line starting with d)

```
bash-3.1$ ls -l | grep '^d' | grep -i '\<i'
drwxr-x--- 2 cchrys tspecial 4096 Aug 19 2002 idcc
drwxr-x--- 5 cchrys tspecial 4096 Jul 17 2001 IPE
```



Εντολή *grep*

- Παραδείγματα (συνέχεια) → Single, Double quotes important when set here ("\$HOME")

```
bash-3.1$ grep '^c.*h$' /usr/share/dict/words | more
```

```
cabbalah  
cable-stitch  
cablish  
caddish  
cadish  
caducibranch  
cafeneh  
cafh  
cailleach  
cailliach  
calabash  
calash  
calenturish  
calfish  
calipash  
caliph  
calligraph  
callipash  
calvish  
calycanth  
camelish  
cameograph  
camooch  
--More--
```

→ Start with C, one char (any), 0:M chars, end with h

```
bash-3.1$ grep '^c..h$' /usr/share/dict/words | more
```

```
cafh  
caph  
cash  
cath  
chih  
coch  
cosh  
coth  
croh  
csch  
cush
```

→ Start with C, two chars (any), end with h



Κανονικές Εκφράσεις

- Μια κανονική έκφραση μπορεί να χρησιμοποιήσει τους ακόλουθους *metacharacters (repetition operators)* – **Extended Regular Expression (ERE) – used with egrep**

Operator	Effect
.	Matches any single character (1:1)
?	The preceding item will be matched zero times or once (0:1)
*	The preceding item will be matched zero or more times (0:N)
+	The preceding item will be matched one or more times (1:N)
{N}	The preceding item is matched exactly N times (N:N)
{N,}	The preceding item is matched N or more times (N:-)
{N,M}	The preceding item is matched at least N times, but not more than M times (N:M)
-	Represents a range (e.g., [a-z] ή [^a-zA-Z])
^	Matches the empty string at the beginning of a line ; also represents the characters not in the range of list [^]
\$	Matches the empty string at the end of a line
\b	Matches the empty string at the edge of a word
\<	Matches the start-of-word (word that has a preceding whitespace)
\>	Matches the end-of-word (word that has a whitespace following)



Κανονικές Εκφράσεις (Regular Expressions)

- [] Match anything inside the square brackets for one character position once and only once, for example, [12] means match the target to either 1 or 2 while [0123456789] means match to any character in the range 0 to 9.
- The - (dash) **inside square brackets** is the 'range separator' and allows us to define a range, in our example above of [0123456789] we could rewrite it as [0-9]. You can define more than one range inside a list e.g. [0-9A-C] means check for 0 to 9 and A to C (but not a to c).
NOTE: To test for - inside brackets (as a **literal**) it must come first or last, that is, [-0-9] will test for - and 0 to 9.
- ^ The ^ (circumflex or caret) **inside square brackets** negates the expression (we will see an alternate use for the circumflex/caret **outside** square brackets later), for example, [^Ff] means anything except upper or lower case F and [^a-z] means everything except lower case a to z.
NOTE: Spaces, or in this case the lack of them, between ranges are very important.

Κανονικές Εκφράσεις (Regular Expressions)

? The ? (question mark) matches the **preceding character** 0 or 1 times only, for example, `colou?r` will find both `color` and `colour`.

***** The * (asterisk or star) matches the preceding character 0 or more times, for example, `tre*` will find `tree` and `tread` and `trough`.

+ The + (plus) matches the previous character 1 or more times, for example, `tre+` will find `tree` and `tread` but not `trough`.

{n} Matches the preceding character `n` times exactly, for example, to find a local phone number we could use `[0-9]{3}-[0-9]{4}` which would find any number of the form `123-4567`.

Note: The - (dash) in this case, because it is outside the square brackets, is a **literal**. Value is enclosed in braces (curly brackets).

{n,m} Matches the preceding character at least `n` times but not more than `m` times, for example, `'ba{2,3}b'` will find `'baab'` and `'baaab'` but NOT `'bab'` or `'baaaab'`. Values are enclosed in braces (curly brackets).

POSIX Character Classes

- Finally, certain named classes of characters are predefined within bracket expressions, as follows.

POSIX class	similar to	meaning
<code>[:upper:]</code>	<code>[A-Z]</code>	uppercase letters
<code>[:lower:]</code>	<code>[a-z]</code>	lowercase letters
<code>[:alpha:]</code>	<code>[A-Za-z]</code>	upper- and lowercase letters
<code>[:digit:]</code>	<code>[0-9]</code>	digits
<code>[:xdigit:]</code>	<code>[0-9A-Fa-f]</code>	hexadecimal digits
<code>[:alnum:]</code>	<code>[A-Za-z0-9]</code>	digits, upper- and lowercase letters
<code>[:punct:]</code>		punctuation (all graphic characters except letters and digits)
<code>[:blank:]</code>	<code>[\t]</code>	space and TAB characters only
<code>[:space:]</code>	<code>[\t\n\r\f\v]</code>	blank (whitespace) characters
<code>[:cntrl:]</code>		control characters
<code>[:graph:]</code>	<code>[^ [:cntrl:]]</code>	graphic characters (all characters which have graphic representation)
<code>[:print:]</code>	<code>[[:graph]]</code>	graphic characters and space

