

Κεφάλαιο 5

5 Άδειες πρόσβασης και ασφάλεια αρχείων

5.1 Άδειες πρόσβασης

Τα συστήματα αρχείων που βασίζονται στο UNIX, από την αρχή δημιουργίας του UNIX είχαν αυξημένες δυνατότητες σε σχέση με την ασφάλεια των δεδομένων, εφόσον επρόκειτο για ένα λειτουργικό σύστημα πολλαπλών χρηστών (multi user). Είναι προφανές ότι θα έπρεπε να δίνεται η δυνατότητα στους χρήστες να μπορούν να προστατεύουν τα προσωπικά τους αρχεία καθώς και στον διαχειριστή του συστήματος να μπορεί να προστατεύσει τα αρχεία του συστήματος από τους χρήστες. Για αυτόν τον σκοπό υπάρχει η έννοια των αδειών χρήσης των αρχείων. Οι άδειες πρόσβασης ορίζονται για όλους τους τύπους των αρχείων που αναφέρθηκαν στο Υποκεφάλαιο 3.5. Αυτό που μας ενδιαφέρει βέβαια σε πρώτη φάση είναι οι άδειες πρόσβασης για κανονικά αρχεία (regular files) και καταλόγους (directories).

- ✓ Οι άδειες πρόσβασης για ένα αρχείο «ορίζουν» τι μπορεί να γίνει με τα περιεχόμενα του αρχείου.
- ✓ Οι άδειες πρόσβασης για έναν κατάλογο «ορίζουν» τι μπορεί να γίνει με τα περιεχόμενα του καταλόγου, παραδείγματος χάριν αν μπορεί ο χρήστης να προσθέσει ή να διαγράψει αρχεία.

Όπως φαίνεται στο Σχήμα 5.1, μια από τις πληροφορίες που μας εμφανίζει η εντολή "ls -l" είναι οι άδειες πρόσβασης. Στην πρώτη στήλη, εκτός από τον τύπο του αρχείου, υπάρχουν ακόμη εννέα χαρακτήρες οι οποίοι εκφράζουν τις άδειες πρόσβασης του κάθε αρχείου. Οι άδειες πρόσβασης, βέβαια, όπως θα δούμε παρακάτω, έχουν νόημα μαζί με την πληροφορία ιδιοκτησίας του κάθε αρχείου (χρήστης και ομάδα).

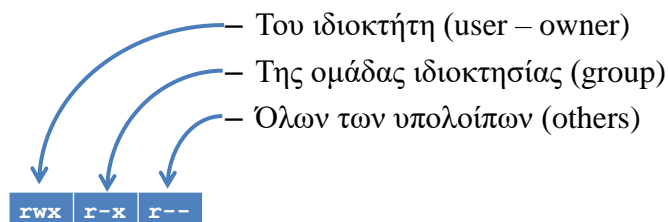
```
asidirop@asidirop:/tmp$ ls -l
total 84
-rw-r--r-- 1 asidirop asidirop 45539 Feb 28 19:32 a.xml
-rw----- 1 asidirop asidirop      0 Mar  5 10:06 bib2html513cd3
-rw-r--r-- 1 asidirop asidirop   148 Mar  5 10:06 bib2html513cd3.aux
-rw-r--r-- 1 asidirop asidirop  1118 Mar  5 10:06 bib2html513cd3.bbl
drwx----- 2 asidirop asidirop  4096 Jan  1  1970 orbit-asidirop
```

↑	↑	↑	↑	↑	↑	↑	↑	↑	↑
Τύπος Αρχείου	Άδειες πρόσβασης	Σύνδεσμοι	Ιδιοκτήτης Αρχείου	Ιδιοκτήτρια Ομάδα Αρχείου	Μέγεθος Αρχείου	Ωρα Τροποποίησης			Όνομα αρχείου

Σχήμα 5.1: Οι στήλες της ls -l.

Ο συμβολισμός ο οποίος ακολουθείται για την αναπαράσταση των αδειών χρήσης (με τον οποίο συμμορφώνεται και η εντολή "`ls -l`") αποτελείται από εννέα χαρακτήρες. Ουσιαστικά αυτοί οι εννέα χαρακτήρες χωρίζονται σε τρεις τριάδες χαρακτήρων. Κάθε τριάδα αφορά και διαφορετικό "target" στο οποίο αναφέρεται. Έτσι (Σχήμα 5.2):

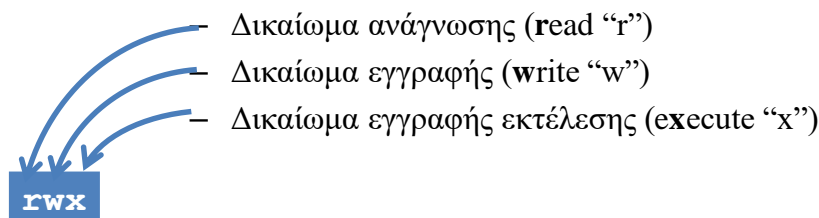
1. Η πρώτη τριάδα αφορά τις άδειες χρήσης που έχουν οριστεί για τον ιδιοκτήτη του αρχείου (συμβολίζεται ως "user").
2. Η δεύτερη τριάδα αφορά τις άδειες χρήσης που έχουν οριστεί για την ομάδα ιδιοκτησίας του αρχείου (συμβολίζεται ως "group").
3. Η τρίτη τριάδα αφορά όλες τις υπόλοιπες περιπτώσεις (συμβολίζεται ως "others").



Σχήμα 5.2: Οι ομάδες αδειών χρήσης.

Η κάθε τριάδα αποτελείται από τρεις χαρακτήρες, όπως στο Σχήμα 5.3. Για κάθε ομάδα δικαιωμάτων (τριάδα) ορίζονται τρία δικαιώματα:

- Δικαίωμα ανάγνωσης (read) που συμβολίζεται με το "r".
- Δικαίωμα εγγραφής (write) που συμβολίζεται με το "w".
- Δικαίωμα εκτέλεσης (execute) που συμβολίζεται με το "x".



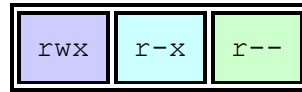
Σχήμα 5.3: Οι τύποι αδειών χρήσης.

Όταν πρόκειται για αρχεία:

- Δικαίωμα ανάγνωσης (read) σημαίνει ότι ο χρήστης που ανήκει στη συγκεκριμένη ομάδα δικαιωμάτων έχει το δικαίωμα να διαβάσει τα περιεχόμενα του αρχείου.
- Δικαίωμα εγγραφής (write) σημαίνει ότι ο χρήστης που ανήκει στη συγκεκριμένη ομάδα δικαιωμάτων έχει το δικαίωμα να μεταβάλει τα περιεχόμενα του αρχείου.
- Δικαίωμα εκτέλεσης (execute) σημαίνει ότι ο χρήστης που ανήκει στη συγκεκριμένη ομάδα δικαιωμάτων έχει το δικαίωμα να εκτελέσει το αρχείο, εφόσον βέβαια το αρχείο είναι κατάλληλου τύπου για εκτέλεση (εκτελέσιμο δυαδικό ή σενάριο - script).

Οι θέσεις των γραμμάτων είναι συγκεκριμένες. Όταν έχει αφαιρεθεί το αντίστοιχο δικαίωμα, στη θέση του αντίστοιχου γράμματος εμφανίζεται μια παύλα "-". Μερικά παραδείγματα είναι:

- `rwX`: ο αντίστοιχος χρήστης έχει τα δικαιώματα read-write-execute.
- `r-x`: ο αντίστοιχος χρήστης έχει τα δικαιώματα read-execute, αλλά όχι το write.
- `r--`: ο αντίστοιχος χρήστης έχει το δικαίωμα read, αλλά όχι τα write και execute.
- `-w-`: ο αντίστοιχος χρήστης έχει το δικαίωμα write, αλλά όχι τα read και execute !!!



Σχήμα 5.4: Παράδειγμα Αδειών Χρήσης.

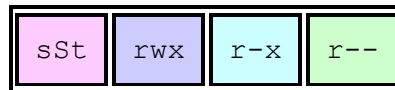
Στο Σχήμα 5.4 παρουσιάζεται ένα παράδειγμα. Οι παρακάτω συμβολισμοί σημαίνουν πως:

- Ο ιδιοκτήτης του αρχείου έχει όλα τα δικαιώματα για το αρχείο (`rwX`).
- Οι χρήστες που ανήκουν στην ίδια ομάδα με το αρχείο έχουν τα δικαιώματα “r” και “x”, αλλά όχι το “w”.
- Όλοι οι υπόλοιποι χρήστες έχουν το δικαίωμα “r”, αλλά όχι “w” και “x”.

5.1.1 Επιπλέον δικαιώματα

Υπάρχει ακόμη μια τριάδα δικαιωμάτων. Αυτή η τριάδα δεν έχει δική της θέση εμφάνισης, αλλά, εάν εμφανίζονταν, θα υπήρχε η εικόνα του Σχήματος 5.5. Αυτή η τριάδα αποτελείται από τα:

- `s`: δυνατότητα αλλαγής ταυτότητας χρήστη (set user id ή `setuid`).
- `S`: δυνατότητα αλλαγής ταυτότητας ομάδας (set group id ή `setgid`).
- `t`: δυαδικό ψηφίο μονιμότητας (sticky bit).



Σχήμα 5.5: Παράδειγμα Αδειών Χρήσης.

Τα δικαιώματα αυτά συνήθως δεν παρέχονται και κυρίως χρησιμοποιούνται για συγκεκριμένες περιπτώσεις:

- sticky bit σε κατάλογο: Εάν στον κατάλογο έχουν δικαίωμα εγγραφής-ανάγνωσης πολλοί χρήστες, τότε ένας χρήστης μπορεί να διαγράψει τα αρχεία άλλου. Εάν στον κατάλογο υπάρχει το sticky bit, τότε η πρόσβαση του χρήστη περιορίζεται μόνο στα αρχεία των οποίων είναι ιδιοκτήτης. Κλασικό παράδειγμα είναι ο φάκελος `/tmp`. Όπως φαίνεται παρακάτω στην περίπτωση του `/tmp`, ο χαρακτήρας “t” εμφανίζεται «επάνω» από το τελευταίο “x” και εννοείται ότι και το “x” για τους υπόλοιπους χρήστες (others) είναι ενεργοποιημένο.

```
asidirop@aetos:~$ ls -l /
total 101
drwxr-xr-x  2 root root  4096 Dec 16 04:01 bin
drwxr-xr-x  4 root root 1024 Feb 19 04:29 boot
drwxr-xr-x 12 root root 3000 Mar 11 11:30 dev
drwxr-xr-x 122 root root 12288 Mar 18 15:42 etc
drwxr-xr-x  5 root root  4096 Feb  3 2014 home
lrwxrwxrwx  1 root root   30 Jan 31 2014 initrd.img ->
boot/initrd.img-2.6.32-5-amd64
drwxr-xr-x 12 root root 12288 Mar  7 04:30 lib
```

```

drwxr-xr-x   4 root root 12288 Mar  7 04:30 lib32
lrwxrwxrwx   1 root root      4 Jan 31 2014 lib64 -> /lib
drwx-----  2 root root 16384 Jan 31 2014 lost+found
drwxr-xr-x   3 root root  4096 Jan 31 2014 media
drwxr-xr-x   8 root root  4096 Feb  3 2014 mnt
drwxr-xr-x   2 root root  4096 Jan 31 2014 opt
dr-xr-xr-x 482 root root      0 Feb 20 01:09 proc
drwxr-xr-x  14 root root  4096 Mar 18 13:30 root
drwxr-xr-x   2 root root  4096 Mar  7 04:30 sbin
drwxr-xr-x   2 root root  4096 Jul 21 2010 selinux
drwxr-xr-x   3 root root  4096 Jan 31 2014 srv
drwxr-xr-x  13 root root      0 Feb 20 01:09 sys
drwxrwxrwt  22 root root  4096 Mar 19 17:11 tmp
drwxr-xr-x  12 root root  4096 Feb  3 2014 usr
drwxr-xr-x  18 root root  4096 Feb  5 2014 var
lrwxrwxrwx   1 root root      27 Jan 31 2014 vmlinuz ->
boot/vmlinuz-2.6.32-5-amd64

```

- Σε προγράμματα του συστήματος, όταν υπάρχει το `setuid`, τότε η διεργασία που προέρχεται από το πρόγραμμα έχει το δικαίωμα να αλλάξει ιδιοκτήτη. Όταν ένας χρήστης εκτελεί ένα πρόγραμμα, τότε η διεργασία που προκύπτει έχει ως ιδιοκτήτη τον συγκεκριμένο χρήστη, άρα και τα δικαιώματα και τις άδειες του χρήστη. Αν όμως το πρόγραμμα που εκτελέστηκε έχει το `setuid`, τότε η διεργασία έχει το δικαίωμα να αλλάξει ιδιοκτήτη, π.χ. να γίνει `root`, άρα και η διεργασία αποκτά όλες τις άδειες και τα δικαιώματα του χρήστη `root`. Για να συμβεί αυτό, βέβαια, πρέπει να συνδυάζονται αρκετές συνθήκες. Παρόλα αυτά, η δυνατότητα αυτή είναι ένα λεπτό σημείο στον τομέα ασφάλειας του UNIX, για αυτό και υπάρχουν πολλές δικλείδες προστασίας, παράδειγμα οι εξωτερικοί ή δικτυακοί δίσκοι ορίζονται ως “`nosuid`”. Αυτό σημαίνει ότι ακόμη και αν ένα πρόγραμμα είναι μαρκαρισμένο ως “`setuid`”, αυτή η σήμανση θα αγνοηθεί από το λειτουργικό.
- Αντίστοιχα, σε προγράμματα που έχουν την άδεια “`setgid`”, που συμβολίζεται με “`S`”, η διεργασία που προκύπτει με την εκτέλεση αυτών των προγραμμάτων έχει το δικαίωμα να αλλάξει την ομάδα ιδιοκτησίας της (παράδειγμα να θέσει ως ομάδα το “`mail`” και να έχει το δικαίωμα να διαβάσει δεδομένα από τα αρχεία της υπηρεσίας ηλεκτρονικού ταχυδρομείου).

5.1.2 Αποθήκευση δικαιωμάτων

Εσωτερικά στο σύστημα αρχείων τα δικαιώματα αποθηκεύονται (όπως και όλα στον κόσμο των υπολογιστών) σε μια ακολουθία από bits. Τα bits αυτά συμπεριλαμβάνονται σε μια δομή η οποία ονομάζεται `inode`. Περισσότερα για τη δομή `inode` μπορείτε να βρείτε στις διαφάνειες της θεωρίας του μαθήματος καθώς και στα [4, 3, 5]. Για την αποθήκευση, λοιπόν, των δικαιωμάτων ενός αρχείου, απαιτούνται συνολικά 12 bits όπως στο Σχήμα 5.6.

sSt	rwX	r-x	r--
111	111	101	100

Σχήμα 5.6: Παράδειγμα αποθήκευσης Αδειών Χρήσης.

Κάθε bit αντιστοιχεί και σε ένα δικαίωμα. Όταν υπάρχει το δικαίωμα, τότε το αντίστοιχο bit είναι “1”. Όταν έχει αφαιρεθεί το δικαίωμα, τότε το αντίστοιχο bit είναι “0”. Επειδή η χρήση του δυαδικού συστήματος από τους χρήστες είναι δύσκολη, και επίσης η μετατροπή των τεσσάρων τριάδων από bits σε δεκαδικό

σύστημα είναι επίσης δύσκολη, για την αναπαράσταση αυτών των 12 bits χρησιμοποιείται πάντα το οκταδικό σύστημα. Έτσι, κάθε ομάδα δικαιωμάτων (3 bits) αντιστοιχεί σε ένα οκταδικό ψηφίο. Η προφανής αντιστοιχία δυαδικού με οκταδικό αριθμό φαίνεται στον Πίνακα 5.1. Το μονό οκταδικό ψηφίο συμπίπτει με το δεκαδικό, δηλαδή το "5" σε οκταδικό είναι ίσο με το "5" σε δεκαδικό. Όμως το "15" σε οκταδικό δεν είναι ίσο με "15" σε δεκαδικό, αλλά ίσο με "13".

Πίνακας 5.1: Αντιστοιχία Οκταδικού - Δυαδικού Συστήματος

8αδικό	2αδικό
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Δεδομένου ότι κάθε τριάδα από bits αντιστοιχίζεται σε ένα οκταδικό ψηφίο, για την αναπαράσταση των 12 bits, απαιτούνται τέσσερα οκταδικά ψηφία. Συνεπώς, το σύνολο των αδειών πρόσβασης αναπαριστάται με έναν τετραψήφιο οκταδικό αριθμό. Στον Πίνακα 5.2 παρουσιάζονται μερικά παραδείγματα. Για τα δικαιώματα sSt θεωρείται ότι απουσιάζουν και παραλείπονται, όμως συμπεριλαμβάνουμε τα αντίστοιχα bits στη δεύτερη στήλη του πίνακα.

Πίνακας 5.2: Παραδείγματα αναπαράστασης σε Οκταδικό και Δυαδικό Σύστημα

Σύμβολα	2αδικό	8αδικό
rwX rwx r--	000 111 111 100	0774
rw- --x ---	000 110 001 000	0610
--- --- ---	000 000 000 000	0000
-wX --- r--	000 011 000 100	0604
r-x --x -w-	000 101 001 010	0512

5.2 Σημασία δικαιωμάτων

Ποια είναι, όμως, ακριβώς η σημασία των δικαιωμάτων; Για την περίπτωση των αρχείων είναι σχεδόν προφανές και όπως αναφέρθηκε παραπάνω:

- Δικαίωμα ανάγνωσης (read) σημαίνει ότι ο χρήστης που ανήκει στην συγκεκριμένη ομάδα δικαιωμάτων έχει το δικαίωμα να διαβάσει τα περιεχόμενα του αρχείου.
- Δικαίωμα εγγραφής (write) σημαίνει ότι ο χρήστης που ανήκει στην συγκεκριμένη ομάδα δικαιωμάτων έχει το δικαίωμα να μεταβάλει τα περιεχόμενα του αρχείου.
- Δικαίωμα εκτέλεσης (execute) σημαίνει ότι ο χρήστης που ανήκει στην συγκεκριμένη ομάδα δικαιωμάτων έχει το δικαίωμα να εκτελέσει το αρχείο - εφόσον βέβαια το αρχείο είναι κατάλληλου τύπου για εκτέλεση (εκτελέσιμο δυαδικό ή σενάριο - script).

Όμως για την περίπτωση των καταλόγων δεν είναι προφανές. Για να κατανοήσουμε καλύτερα την χρήση των δικαιωμάτων σε καταλόγους θα πρέπει να μελετήσουμε λίγο την εσωτερική δομή ενός καταλόγου.

5.2.1 Εσωτερική δομή Συστήματος Αρχείων

Όπως αναφέρθηκε νωρίτερα (Παράγραφο 5.1.2), στο σύστημα αρχείων υπάρχει μια βασική δομή η οποία ονομάζεται i-node. Για κάθε αρχείο δημιουργείται ένα i-node, μέσα στο οποίο αποθηκεύονται όλες οι πληροφορίες σχετικές με το αρχείο, όπως ημερομηνία τροποποίησης, μέγεθος, ιδιοκτήτης, οι φυσικές διευθύνσεις των μπλοκ του δίσκου στις οποίες είναι αποθηκευμένα τα δεδομένα του αρχείου, οι άδειες χρήσης κ.ά. Κατά τη μορφοποίηση ενός δίσκου (format) δημιουργείται ένας πίνακας που αποτελείται από μερικά εκατομμύρια ή τρισεκατομμύρια i-nodes. Για κάθε αρχείο που δημιουργείται, δεσμεύεται και μια θέση στον πίνακα αυτόν, στην οποία θέση αποθηκεύονται οι πληροφορίες για το αρχείο.

Ένας κατάλογος στην ουσία είναι ένα αρχείο το οποίο περιέχει τη λίστα με τα ονόματα αρχείων που περιέχονται στον κατάλογο. Για κάθε αρχείο πρέπει να υπάρχει επίσης και η πληροφορία σε ποιο i-node αντιστοιχεί. Ουσιαστικά λοιπόν, ένας κατάλογος είναι ένα αρχείο που περιέχει την πληροφορία όπως παρουσιάζεται στον Πίνακα 5.3. Βέβαια, ανάλογα με τον Τύπο Συστήματος Αρχείων η δομή που παρουσιάζεται μπορεί να διαφέρει, ώστε να επιτρέπει γρήγορη αναζήτηση. Η ουσία, όμως, είναι ότι αποθηκεύονται οι πληροφορίες «όνομα αρχείου» και «αριθμός i-node».

Πίνακας 5.3: Παραδείγματα αποθήκευσης καταλόγου

Όνομα Αρχείου	i-node
.	134289164
..	100734149
file1	134289167
file2	134289171
dir7_test	134289172
file8.txt	134289173
lala.c	302018510

Όταν λοιπόν ο χρήστης δίνει την εντολή: `cat file1`, τότε ο πυρήνας, για να βρει τα περιεχόμενα του `file1`, πρώτα απ' όλα ψάχνει στον κατάλογο να βρει αν υπάρχει όντως αρχείο με το όνομα `file1`. Αν ναι, τότε διαβάζει από ποιο i-node θα βρει πληροφορίες για το αρχείο. Αφού «δει» ότι το `file1` αντιστοιχεί στο i-node "134289167", τότε ανοίγει τον πίνακα των i-nodes και διαβάζει το i-node στη θέση 134289167 του πίνακα. Με αυτόν τον τρόπο, έχει διαβάσει τις πληροφορίες που αφορούν το αρχείο. Μέσα σ' αυτές τις πληροφορίες είναι και η πληροφορία σε ποια blocks του δίσκου βρίσκονται τα δεδομένα του αρχείου. Διαβάζει τα συγκεκριμένα μπλοκ και «δίνει» τα δεδομένα στη διεργασία της εντολής `cat`. Έτσι, η `cat` κατάφερε να διαβάσει τα δεδομένα του `file1` και έπειτα να τα εκτυπώσει στο τερματικό.

Όταν, λοιπόν, δημιουργείται ένα αρχείο, πρέπει να γραφτεί στο αρχείο του καταλόγου μια νέα γραμμή που θα αφορά το νέο αρχείο. Όταν διαγράφεται από έναν κατάλογο ένα αρχείο, τότε στην πραγματικότητα διαγράφεται η γραμμή που αφορά το αρχείο από τον πίνακα που διατηρείται στον κατάλογο.

Ας θεωρήσουμε ότι ο κάθε κατάλογος περιέχει αυτόν τον δίστηλο πίνακα όπως φαίνεται στον Πίνακα 5.3.

- Το δικαίωμα "read" δίνει τη δυνατότητα να «διαβάσουμε» τα ονόματα αρχείων του καταλόγου από την πρώτη στήλη του πίνακα.
- Το δικαίωμα "write" δίνει τη δυνατότητα να «μεταβάλουμε» τα περιεχόμενα του καταλόγου, δηλαδή να προσθέσουμε ή να αφαιρέσουμε γραμμές στον πίνακα.

- Το δικαίωμα “execute” δίνει τη δυνατότητα να «διαβάσουμε» τα i-node numbers των αρχείων, δηλαδή, όπως φαίνεται σχηματικά, να διαβαστεί η δεύτερη στήλη του πίνακα. Επίσης, για να μπορεί ένας χρήστης να κάνει “cd” σε έναν κατάλογο, πρέπει να έχει και δικαίωμα “execute” στον κατάλογο.

Συνοψίζοντας με βάση τις ενέργειες:

- **Διαγραφή αρχείου ή καταλόγου:** Πρέπει να έχω “w” και “x” στον κατάλογο μέσα στον οποίο βρίσκεται το αρχείο προς διαγραφή, διότι:
 - ο Πρέπει να διαβάσει ο πυρήνας το i-node του αρχείου, ώστε να το μαρκάρει ως ελεύθερο. Συνεπώς, απαιτείται το “x” στον κατάλογο.
 - ο Πρέπει να μεταβληθούν τα περιεχόμενα του καταλόγου. Συνεπώς, απαιτείται το “w” στον κατάλογο.
 - ο Δεν επηρεάζουν τα δικαιώματα του αρχείου που θα σβήσω, διότι δεν θα διαβαστούν/μεταβληθούν τα περιεχόμενα του αρχείου. Συνεπώς, δεν χρειάζεται να έχω “w” ή “r” για το αρχείο.
- **Δημιουργία αρχείου ή καταλόγου:** Πρέπει να έχω “w” και “x” στον κατάλογο.
- **Εμφάνιση ονομάτων αρχείων:** Πρέπει να έχω “r” στον κατάλογο, για να δω τα ονόματα αρχείων.
- **Εμφάνιση πληροφοριών αρχείων** (πχ. με την `ls -l`): Πρέπει να έχω “r” και “x” στον κατάλογο.
- **cd στον κατάλογο:** Πρέπει να έχω “x” στον κατάλογο.

5.3 Ορισμός και μεταβολή δικαιωμάτων

Ο ορισμός ή μεταβολή των δικαιωμάτων ενός αρχείου ή καταλόγου μπορεί να γίνει είτε από τον ιδιοκτήτη του αρχείου, είτε από τον χρήστη root. Ακόμη και αν από τον ιδιοκτήτη του αρχείου έχουν δοθεί όλες οι άδειες προς τους υπόλοιπους χρήστες, ακόμη και αν έχουν δοθεί όλες οι άδειες και στον κατάλογο μέσα στον οποίο βρίσκεται το αρχείο, κανένας χρήστης δεν μπορεί να αλλάξει τα δικαιώματα παρά μόνο οι δυο που προαναφέρθηκαν.

Η μεταβολή των δικαιωμάτων ενός αρχείου μπορεί να γίνει χρησιμοποιώντας την εντολή: “chmod” (change mode) [2].

chmod [OPTIONS] δικαιώματα αρχεία

Μπορούμε στην εντολή να δώσουμε ως όρισμα ένα ή περισσότερα ονόματα αρχείων. Τα ονόματα αρχείων πρέπει να είναι τα τελευταία ορίσματα στην εντολή. Προφανώς μπορούν να χρησιμοποιηθούν και χαρακτήρες μπαλαντέρ. Η εντολή αναγνωρίζει τα δικαιώματα με δυο τρόπους. Είτε μπορεί να χρησιμοποιηθεί ο τετραψήφιος οκταδικός αριθμός, είτε κάποια σύμβολα που θα εξηγήσουμε παρακάτω. Όταν στον οκταδικό αριθμό παραλείπονται κάποια ψηφία, τότε εννοείται ότι αντιστοιχούν στο μηδέν. Στο παρακάτω παράδειγμα, οι εντολές ανά δυο είναι ισοδύναμες.

```
chmod 500 file1
chmod 0500 file1
chmod 4 file1
chmod 0004 file1
```

Για τους χρήστες που δεν τα πάνε καλά με τους αριθμούς, η chmod μπορεί να καταλάβει μια σειρά από σύμβολα που παρουσιάζονται στον Πίνακα 5.4:

Πίνακας 5.4: Σύμβολα που χρησιμοποιούνται στην εντολή *chmod*

Σύμβολο	Σημασία
+	πρόσθεσε τα δικαιώματα που ακολουθούν
-	αφαίρεσε τα δικαιώματα που ακολουθούν
=	θέσε ακριβώς τα δικαιώματα που ακολουθούν
r	δικαίωμα read
w	δικαίωμα write
x	δικαίωμα execute
u	user - ιδιοκτήτης του αρχείου
g	group - ομάδα του αρχείου
o	others - υπόλοιποι χρήστες
a	all - όλες οι ομάδες χρηστών
,	διαχωριστής

Ακολουθούν κάποια παραδείγματα:

```
chmod ug=rx,o=r file1
```

θα θέσει στο αρχείο τα δικαιώματα "r-xr-xr--".

```
chmod u=rwx,g=r,o= file1
```

θα θέσει στο αρχείο τα δικαιώματα "rwxr-----".

```
chmod a=r,u+w file1
```

θα θέσει στο αρχείο αρχικά τα δικαιώματα "r--r--r--" και μετά θα προσθέσει "wx" στον ιδιοκτήτη. Συνεπώς, το τελικό αποτέλεσμα θα είναι: "rwxr--r--".

Όταν χρησιμοποιούνται οι ενέργειες +/-, τότε τα δικαιώματα που προκύπτουν τελικά εξαρτώνται και από τα δικαιώματα που υπήρχαν πριν.

- Το + προσθέτει στα ήδη υπάρχοντα δικαιώματα. Προφανώς, αν υπάρχει ήδη το δικαίωμα, δεν προσθέτει κάτι.
- Το - αφαιρεί από τα υπάρχοντα. Προφανώς, αν δεν υπάρχει το δικαίωμα, δεν μπορεί να το αφαιρέσει.

```
chmod ug+w file1
```


Έστω το file1 έχει αρχικά τα δικαιώματα: "r-xr-xr--". Η παραπάνω εντολή θα προσθέσει στον ιδιοκτήτη και στην ομάδα το "w". Συνεπώς, τα τελικά δικαιώματα που θα αποκτήσει το αρχείο θα είναι: "rwxrwxr--".

Αν το file1 είχε αρχικά τα δικαιώματα: "rw-r-----", η ίδια εντολή θα προσθέσει μόνο στην ομάδα το "w", διότι ο ιδιοκτήτης το έχει ήδη. Συνεπώς, τα τελικά δικαιώματα που θα αποκτήσει το αρχείο θα είναι: "rw-rw-----".

Χρήσιμο όρισμα-σημαία στην εντολή chmod είναι το "-R" (recursively). Αυτή η σημαία οδηγεί την εντολή να εφαρμόσει τις αλλαγές στα δικαιώματα αναδρομικά σε ολόκληρη την ιεραρχία καταλόγων. Προφανώς, έχει νόημα, όταν ένα από τα ορίσματά της είναι κατάλογος. Εδώ να κάνουμε την παρατήρηση ότι οι περισσότερες εντολές (πχ οι cp, rm κ.ά.) δέχονται τη σημαία «αναδρομικά» με μικρό "-r". Για την chmod, όμως, το "-r" σημαίνει αφαίρεση του δικαιώματος "read". Συνεπώς, για να μην υπάρχει σύγχυση, η chmod δέχεται τη σημαία «αναδρομικά» ως "-R".

5.3.1 Προκαθορισμένα δικαιώματα και η umask

Όταν δημιουργείται ένας κατάλογος, τότε αυτός αποκτά όλα τα δικαιώματα, δηλαδή:

```
rwxrwxrwx (0777)
```

Όταν δημιουργείται ένα αρχείο, τότε και αυτό αποκτά όλα τα δικαιώματα εκτός από το δικαίωμα εκτέλεσης:

```
rw-rw-rw- (0666)
```

Το δικαίωμα εκτέλεσης απουσιάζει από τα αρχεία για λόγους ασφάλειας. Επίσης, το μεγαλύτερο ποσοστό αρχείων που υπάρχουν σε έναν υπολογιστή δεν είναι προγράμματα ώστε να εκτελεστούν, αλλά δεδομένα (εικόνες, βίντεο, κείμενο, κτλ.). Μια εγκατάσταση ενός πακέτου-προγράμματος, συνήθως περιλαμβάνει τη δημιουργία δυο-τριών εκτελέσιμων αρχείων και μερικών δεκάδων (ή και χιλιάδων) βοηθητικών αρχείων.

Όμως το παραπάνω είναι αυτοαναιρούμενο. Από τη μια απουσιάζει το δικαίωμα εκτέλεσης από τα αρχεία για λόγους ασφαλείας - το οποίο σημαίνει ότι δίνεται ιδιαίτερη βαρύτητα στην ασφάλεια - και από την άλλη με τη δημιουργία ενός αρχείου μπορούν να το μεταβάλλουν όλοι οι χρήστες, διότι όλοι οι χρήστες θα έχουν το δικαίωμα εγγραφής ("w").

Προφανώς, κάτι τέτοιο δεν είναι πρακτικό, διότι κάθε φορά που ένας χρήστης δημιουργεί ένα αρχείο θα πρέπει να το κλειδώνει αμέσως μετά. Από την άλλη, στο χρονικό διάστημα που θα μεσολαβήσει από τη στιγμή δημιουργίας του αρχείου μέχρι τη στιγμή «κλειδώματος» του αρχείου, μπορεί κάποιος άλλος χρήστης να προλάβει να μεταβάλει ή να διαβάσει τα δεδομένα του. Συνεπώς, εδώ το σύστημα μοιάζει να πάσχει στο θέμα της ασφάλειας.

Υπάρχει μια μεταβλητή του συστήματος (η οποία ορίζεται ανά διεργασία) και περιέχει την πληροφορία ποια δικαιώματα θα αφαιρούνται αυτόματα από τα αρχεία και τους καταλόγους που θα δημιουργήσει ένας χρήστης. Αυτή η μεταβλητή ονομάζεται umask και ο χειρισμός της γίνεται με την ομώνυμη εντολή [1, 4]. Για την αναπαράσταση του αριθμού umask χρησιμοποιούνται 4 οκταδικά ψηφία (12 bits). Το πρώτο οκταδικό ψηφίο δεν έχει νόημα να έχει τιμή διαφορετική από το μηδέν, διότι έτσι κι αλλιώς τα δικαιώματα sSt δεν δίνονται σε καμία περίπτωση εξ αρχής. Για λόγους, όμως, ομοιομορφίας χρησιμοποιούνται συνήθως τέσσερα ψηφία αντί των τριών που απαιτούνται.

Η εντολή umask χωρίς ορίσματα εμφανίζει την τιμή της μάσκας δικαιωμάτων (umask). Η εντολή umask με όρισμα έναν οκταδικό αριθμό θέτει νέα τιμή στη μάσκα δικαιωμάτων.

```

asidirop@antonis-PC:~/tmp/test$ umask
0022
asidirop@antonis-PC:~/tmp/test$ umask 777
asidirop@antonis-PC:~/tmp/test$ umask
0777
asidirop@antonis-PC:~/tmp/test$ touch file1
asidirop@antonis-PC:~/tmp/test$ ls -l
total 0
----- 1 asidirop asidirop 0 Mar 21 20:15 file1
asidirop@antonis-PC:~/tmp/test$ umask 055
asidirop@antonis-PC:~/tmp/test$ touch file2
asidirop@antonis-PC:~/tmp/test$ ls -l
total 0
----- 1 asidirop asidirop 0 Mar 21 20:15 file1
-rw--w--w- 1 asidirop asidirop 0 Mar 21 20:15 file2
asidirop@antonis-PC:~/tmp/test$ umask 044
asidirop@antonis-PC:~/tmp/test$ touch file3
asidirop@antonis-PC:~/tmp/test$ ls -l
total 0
----- 1 asidirop asidirop 0 Mar 21 20:15 file1
-rw--w--w- 1 asidirop asidirop 0 Mar 21 20:15 file2
-rw--w--w- 1 asidirop asidirop 0 Mar 21 20:55 file3
asidirop@antonis-PC:~/tmp/test$ umask 022
asidirop@antonis-PC:~/tmp/test$ touch file4
asidirop@antonis-PC:~/tmp/test$ ls -l
total 0
----- 1 asidirop asidirop 0 Mar 21 20:15 file1
-rw--w--w- 1 asidirop asidirop 0 Mar 21 20:15 file2
-rw--w--w- 1 asidirop asidirop 0 Mar 21 20:55 file3
-rw-r--r-- 1 asidirop asidirop 0 Mar 21 20:55 file4
asidirop@antonis-PC:~/tmp/test$

```

Στο παραπάνω παράδειγμα ακολουθίας εντολών, αρχικά εμφανίζουμε την τιμή της `umask` η οποία είναι 0022. Με την επόμενη εντολή, θέτουμε την `umask` σε 777 και επιβεβαιώνουμε ότι τέθηκε η σωστή τιμή. Εδώ παρατηρούμε ότι παρότι χρησιμοποιήσαμε τρία ψηφία κατά τον ορισμό τιμής, κατά την εμφάνιση χρησιμοποιούνται τέσσερα. Θέτοντας την `umask` σε 777 σημαίνει ότι θα αφαιρούνται όλα τα δικαιώματα από τα αρχεία που θα δημιουργήσουμε από εδώ και στο εξής (Σχήμα 5.7). Με την επόμενη εντολή δημιουργούμε το αρχείο "file1" και με την "ls -l" επιβεβαιώνουμε ότι από αυτό το αρχείο έχουν αφαιρεθεί κατά τη δημιουργία του όλα τα δικαιώματα.

Έπειτα, θέτουμε την `umask` σε 055. Αυτό σημαίνει ότι από τον ιδιοκτήτη δεν θα αφαιρείται κανένα δικαίωμα. Από την "ομάδα" όμως και από τους "υπόλοιπους" θα αφαιρούνται τα δικαιώματα "r" και "x" (Σχήμα 5.7).

Όμως, αφού δημιουργήσουμε το αρχείο file2 και αναμένουμε από τον ιδιοκτήτη να μην έχει αφαιρεθεί κανένα δικαίωμα, παρατηρούμε (με την `ls -l`) ότι το file2 δεν έχει στον ιδιοκτήτη το "x". Αυτό συνέβη, διότι, όπως αναφέραμε νωρίτερα, όταν δημιουργούνται αρχεία, δεν μπαίνει το δικαίωμα "x". Στην ουσία, λοιπόν, ακόμη και αν δεν είχαμε ορίσει να αφαιρείται το "x" από τους «υπόλοιπους» και την ομάδα, αυτό έτσι κι αλλιώς δεν θα υπήρχε. Αυτό το αποδεικνύουμε με την επόμενη εντολή, κατά την οποία θέτουμε το `umask` 044. Αυτό σημαίνει ότι από την «ομάδα» και τους «υπολοίπους» θα αφαιρείται το δικαίωμα "r". Δημιουργώντας το αρχείο "file3" βλέπουμε ότι τελικά και το file3 (που δημιουργήθηκε με `umask` 044) και το file2 (που δημιουργήθηκε με `umask` 055) έχουν τελικά τα ίδια δικαιώματα. Άρα, τελικά έχει νόημα το τελευταίο bit ("x") στο `umask`; Θα δώσουμε την απάντηση στο επόμενο παράδειγμα. Τέλος, επαναφέρουμε το `umask` στην τιμή 022, η οποία είναι μια συνηθισμένη τιμή για `umask` (όπως και η 077).

umask 777			
0	7	7	7
000	111	111	111
	rwX	rwX	rwX

umask 055			
0	0	5	5
000	000	101	101
	---	r-X	r-X

umask 044			
0	0	4	4
000	000	100	100
	---	r--	r--

umask 022			
0	0	2	2
000	000	020	020
	---	-w-	-w-

Σχήμα 5.7: Παράδειγμα umask.


Το επόμενο πείραμά μας είναι με την επίδραση της umask κατά τη δημιουργία καταλόγων. Επαναλαμβάνουμε ακριβώς τις ίδιες τιμές με το προηγούμενο παράδειγμα, αλλά αυτή τη φορά δημιουργούμε καταλόγους αντί αρχείων.


```
asidirop@antonis-PC:~/tmp/test$ umask 777
asidirop@antonis-PC:~/tmp/test$ mkdir dir1
asidirop@antonis-PC:~/tmp/test$ ls -l
total 4
d----- 2 asidirop asidirop 4096 Mar 22 21:12 dir1
asidirop@antonis-PC:~/tmp/test$ umask 055
asidirop@antonis-PC:~/tmp/test$ mkdir dir2
asidirop@antonis-PC:~/tmp/test$ ls -l
total 8
d----- 2 asidirop asidirop 4096 Mar 22 21:12 dir1
drwx-w--w- 2 asidirop asidirop 4096 Mar 22 21:12 dir2
asidirop@antonis-PC:~/tmp/test$ umask 044
asidirop@antonis-PC:~/tmp/test$ mkdir dir3
asidirop@antonis-PC:~/tmp/test$ ls -l
total 12
d----- 2 asidirop asidirop 4096 Mar 22 21:12 dir1
drwx-w--w- 2 asidirop asidirop 4096 Mar 22 21:12 dir2
drwx-wx-wx 2 asidirop asidirop 4096 Mar 22 21:12 dir3
asidirop@antonis-PC:~/tmp/test$ umask 022
asidirop@antonis-PC:~/tmp/test$ mkdir dir4
asidirop@antonis-PC:~/tmp/test$ ls -l
total 16
d----- 2 asidirop asidirop 4096 Mar 22 21:12 dir1
drwx-w--w- 2 asidirop asidirop 4096 Mar 22 21:12 dir2
drwx-wx-wx 2 asidirop asidirop 4096 Mar 22 21:12 dir3
```

```
drwxr-xr-x 2 asidirop asidirop 4096 Mar 22 21:13 dir4
asidirop@antonis-PC:~/tmp/test$
```

Η πρώτη τιμή στην *umask* (777), αφού δημιουργήσουμε τον κατάλογο *dir1*, έχει το ίδιο αποτέλεσμα με την περίπτωση δημιουργίας αρχείων. Δηλαδή το *dir1* δεν θα έχει κανένα δικαίωμα ενεργοποιημένο. Η δεύτερη τιμή της *umask* (055) έχει και αυτή την ίδια επίδραση στον κατάλογο *dir2* με το αρχείο *file2*. Από τον κατάλογο αφαιρούνται τα δικαιώματα “r” και “x” από την ομάδα και τους υπολοίπους. Η τρίτη όμως περίπτωση, δηλαδή η τιμή (044) δεν έχει την ίδια επίδραση στον κατάλογο *dir3* με το αρχείο *file3*.

umask 777				
Αρχικά	0	7	7	7
-	0	7	7	7
<hr/>				
=	0	0	0	0
	000	000	000	000
		---	---	---

umask 055				
Αρχικά	0	7	7	7
-	0	0	5	5
<hr/>				
=	0	7	2	2
	000	111	010	010
		rwX	-w-	-w-

umask 044				
Αρχικά	0	7	7	7
-	0	0	4	4
<hr/>				
=	0	7	3	3
	000	111	011	011
		rwX	-wX	-wX


umask 022				
Αρχικά	0	7	7	7
-	0	0	2	2
<hr/>				
=	0	7	5	5
	000	111	101	101
		rwX	r-X	r-X


Σχήμα 5.8: Παράδειγμα υπολογισμού επίδρασης της *umask* στη δημιουργία καταλόγων.


Ο κατάλογος `dir3` παίρνει τα δικαιώματα “w” και “x” στην ομάδα και στους υπολοίπους, ενώ το `file3` είχε μόνο το δικαίωμα “w”. Σε αυτήν την περίπτωση βλέπουμε τη διαφορά της επίδρασης χρήσης της τιμής (055) με την τιμή (044), κάτι το οποίο δεν ίσχυε στην περίπτωση των καταλόγων. Τέλος, η τιμή (022) δίνει διαφορετικά δικαιώματα στον κατάλογο `dir4` απ’ ότι στο αρχείο `file4`.


Απαιτείται προσοχή στην εκτίμηση της επίδρασης μιας τιμής της `umask` στις περιπτώσεις των αρχείων και καταλόγων. Στην περίπτωση των καταλόγων, τα δικαιώματα που προκύπτουν ως αποτέλεσμα μπορούν να υπολογιστούν κάνοντας μια απλή αφαίρεση στο οκταδικό σύστημα της τιμής της `umask` από την τιμή 777 (Σχήμα 5.8).

Αντιθέτως όμως, στην περίπτωση των αρχείων, η αφαίρεση από το (666) δεν θα δώσει σε όλες τις περιπτώσεις τα σωστά αποτελέσματα. Όπως δείχνουμε στο Σχήμα 5.9, μόνο δυο στις τέσσερις περιπτώσεις μας δίνει στο σωστό αποτέλεσμα.

umask 777				
Αρχικά	0	6	6	6
-	0	7	7	7
<hr/>				
=	-	1	1	1
	???	???	???	???
		???	???	???

umask 055				
Αρχικά	0	6	6	6
-	0	0	5	5
<hr/>				
=	0	6	1	1
	000	110	001	001
		rw-	--x	--x

umask 044				
Αρχικά	0	6	6	6
-	0	0	4	4
<hr/>				
=	0	6	2	2
	000	110	010	010
		rw-	-w-	-w-

umask 022				
Αρχικά	0	6	6	6
-	0	0	2	2
<hr/>				
=	0	6	4	4
	000	110	100	100
		rw-	r--	r--

Σχήμα 5.9: Παράδειγμα λάθους υπολογισμών επίδρασης της `umask` στη δημιουργία αρχείων.

Η καλύτερη μέθοδος είναι να υπολογίζουμε την αφαίρεση με βάση τα δικαιώματα. Αυτό θα μας δώσει το σωστό αποτέλεσμα στην περίπτωση των καταλόγων αλλά και στην περίπτωση των αρχείων.

Στο Σχήμα 5.10 παρουσιάζουμε τον τρόπο υπολογισμού που πρέπει να έχουμε στο μυαλό μας. Φυσικά, με την εκπαίδευση και την εμπειρία όλοι οι παρακάτω υπολογισμοί πραγματοποιούνται σχεδόν αυτόματα και ασυναίσθητα.


umask 777 (για καταλόγους)				
Αρχικά	---	rwX	rwX	rwX
-	---	rwX	rwX	rwX
<hr/>				
=	---	---	---	---
	000	000	000	000
		0	0	0


umask 777 (για αρχεία)				
Αρχικά	---	rw-	rw-	rw-
-	---	rwX	rwX	rwX
<hr/>				
=	---	---	---	---
	000	000	000	000
		0	0	0


umask 055 (για καταλόγους)				
Αρχικά	---	rwX	rwX	rwX
-	---	---	r-X	r-X
<hr/>				
=	---	rwX	-w-	-w-
	000	111	010	010
		7	2	2


umask 055 (για αρχεία)				
Αρχικά	---	rw-	rw-	rw-
-	---	---	r-X	r-X
<hr/>				
=	---	rw-	-w-	-w-
	000	110	010	010
		6	2	2

Σχήμα 5.10(α): Παράδειγμα σωστών υπολογισμών επίδρασης της umask στη δημιουργία αρχείων και καταλόγων.

umask 044 (για καταλόγους)				
Αρχικά	---	rwX	rwX	rwX
-	---	---	r--	r--
<hr/>				
=	---	rwX	-wX	-wX
	000	111	011	011
		7	3	3

umask 044 (για αρχεία)				
Αρχικά	---	rw-	rw-	rw-
-	---	---	r--	r--
<hr/>				
=	---	rw-	-w-	-w-
	000	110	010	010
		6	2	2

umask 022 (για καταλόγους)				
Αρχικά	---	rwX	rwX	rwX
-	---	---	-w-	-w-
<hr/>				
=	---	rwX	r-x	r-x
	000	111	101	101
		7	5	5

umask 022 (για αρχεία)				
Αρχικά	---	rw-	rw-	rw-
-	---	---	-w-	-w-
<hr/>				
=	---	rw-	r--	r--
	000	110	100	100
		6	4	4

Σχήμα 5.10(β): Παράδειγμα σωστών υπολογισμών επίδρασης της umask στη δημιουργία αρχείων και καταλόγων.

5.4 Ασκήσεις για εξάσκηση

Στόχος

Άδειες χρήσης. Ο επεξεργαστής κειμένου vi.

Ασκηση 1

Σε ένα τερματικό δοκιμάστε τα εξής:

1. Δείτε την τιμή της umask (μάσκα δικαιωμάτων)
2. Δημιουργήστε έναν κατάλογο με όνομα lab4. Ποια είναι τα δικαιώματα που περιμένετε να έχει ο κατάλογος με βάση την τιμή της umask; Δείτε τα δικαιώματα χρησιμοποιώντας την εντολή stat.
3. Σ' ένα νέο τερματικό, κάντε login στον aetos.it.teithe.gr.
4. Δείτε την τιμή της umask (στο απομακρυσμένο σύστημα)
5. Επαναλάβετε το βήμα δύο (2). Αλλάξτε τα δικαιώματα του lab4 σε 755.
6. Κάντε cd στο lab4 και εκτελέστε την εντολή: whoami>myid
7. Η παραπάνω εντολή θα εμφανίσει το όνομα του τρέχοντος χρήστη. Δεν θα το εμφανίσει όμως στο τερματικό, θα αποθηκεύσει τα αποτελέσματα της στο αρχείο με όνομα myid. Ποια δικαιώματα χρήσης έχει το αρχείο myid; Είναι αναμενόμενα με βάση την τιμή της umask;

Εντολή > αρχείο

Εκτελεί την εντολή. Τα αποτελέσματα όμως της εντολής δεν εμφανίζονται στο τερματικό αλλά αποθηκεύονται στο αρχείο. Αν το αρχείο υπάρχει ήδη, τότε διαγράφεται αυτόματα και ξαναδημιουργείται. Περισσότερα για την ανακατεύθυνση θα δούμε στο Κεφάλαιο 7.

8. Αλλάξτε το umask σε 000 και δημιουργήστε ένα αρχείο με όνομα public. Ποια είναι τα δικαιώματά του;
9. Χρησιμοποιώντας την chmod, αλλάξτε τα δικαιώματά του σε "rwxrw-r--".
10. Φτιάξτε τα παρακάτω αρχεία και χρησιμοποιώντας την chmod δώστε τους τα δικαιώματα που φαίνονται στον πίνακα:

file1	rwx---r--
file2	---rwx--x
file3	-w-r--r-x
file4	rwx-w-rw-
dir1	rwx--x--x
11. Από τον κατάλογο /bin αντιγράψτε στον τρέχοντα (lab4) το αρχείο ls και δώστε του το όνομα myls (αντιγράψτε την εντολή/εκτελέσιμο της ls).
12. Εκτελέστε την εντολή myls. Μπορείτε να την εκτελέσετε;
13. Εκτελέστε την εντολή ./myls. Μπορείτε να την εκτελέσετε;

14. Αφαιρέστε το δικαίωμα “x” από το `mys`. Μπορείτε να την εκτελέσετε;
15. Προσθέστε το δικαίωμα “x” και αφαιρέστε τα δικαιώματα “w”, “r”. Μπορείτε να την εκτελέσετε;

Όταν πληκτρολογούμε μια εντολή για εκτέλεση στο κέλυφος, το κέλυφος δεν ψάχνει να τη βρει στον τρέχοντα κατάλογο. Ψάχνει να βρει τα εκτελέσιμα αρχεία εντολών μέσα σε ένα σύνολο από καταλόγους. Αυτό το σύνολο καταλόγων ονομάζεται `$PATH`. Το `$PATH` είναι μια μεταβλητή περιβάλλοντος, περισσότερα για τις μεταβλητές περιβάλλοντος θα δούμε στο Κεφάλαιο 8. Αυτή η μεταβλητή είναι μια συμβολοσειρά (string) η οποία περιέχει διαδρομές καταλόγων, μέσα στους οποίους το κέλυφος θα ψάξει να βρει το εκτελέσιμο που αντιστοιχεί στην εντολή που έδωσε ο χρήστης. Μπορείτε να δείτε την τιμή αυτής της μεταβλητής με την εντολή:

```
echo $PATH
```

Μέσα σε αυτήν την συμβολοσειρά, οι διαδρομές χωρίζονται μεταξύ τους με τον χαρακτήρα «:». Για να εκτελέσουμε ένα πρόγραμμα, το οποίο είναι αποθηκευμένο σε φάκελο ο οποίος δεν περιλαμβάνεται στο `PATH`, θα πρέπει να πληκτρολογήσουμε την απόλυτη ή σχετική διαδρομή προς το εκτελέσιμο, πχ: το `./mys` είναι μια σχετική διαδρομή προς το αρχείο `mys` του τρέχοντος καταλόγου. Βλέποντας τον χαρακτήρα «/», το κέλυφος καταλαβαίνει ότι δόθηκε διαδρομή προς εκτελέσιμο και όχι μόνο το όνομα του εκτελέσιμου, επομένως, δεν ψάχνει στο `PATH`, αλλά ακολουθεί τη διαδρομή.

16. Η εντολή `which` με όρισμα το όνομα μιας εντολής, εμφανίζει την απόλυτη διαδρομή για το εκτελέσιμο της εντολής. Βρείτε ποιο είναι το εκτελέσιμο της εντολής `date` και της εντολής `man`.

Άσκηση 2

Σε ένα τερματικό δοκιμάστε τα εξής:

- Βρείτε χρησιμοποιώντας την εντολή `who` ποιοι χρήστες είναι συνδεδεμένοι αυτή τη στιγμή στον υπολογιστή `aetos.it.teithe.gr`.
- Χρησιμοποιώντας μπαλαντέρ (wildcards), βρείτε ποιοι φοιτητές έχουν στο home directory τους, κατάλογο με το όνομα `lab4` ο οποίος περιέχει αρχείο με το όνομα `myid`. Δείτε τα περιεχόμενα μερικών αρχείων “`myid`” των συμφοιτητών σας.
- Κλειδώστε το δικό σας αρχείο `myid` με τέτοιο τρόπο, ώστε οι υπόλοιποι χρήστες να μην μπορούν να δουν τα περιεχόμενά του. Χρησιμοποιήστε την εντολή `chmod`.
- Δώστε, στο αρχείο σας `myid`, το δικαίωμα μόνο του “write” σε όλους τους υπόλοιπους χρήστες (και group) και αφαιρέστε για αυτούς τους χρήστες τα δικαιώματα `read/execute`. Εννοείται ότι για τον εαυτό σας διατηρείτε τα δικαιώματα `read/write`.
- Για μερικά από τα αρχεία με το όνομα `myid` των υπολοίπων χρηστών, δοκιμάστε να εκτελέσετε την εντολή του τύπου:

```
(date;whoami) >> ~dimitris/lab4/myid
```

Όπου dimitris, το όνομα του χρήστη που εντοπίσατε ότι έχει το παραπάνω αρχείο. Η παραπάνω εντολή εκτελεί τις εντολές date (εμφανίζει ώρα) και whoami (εμφανίζει ποιος χρήστης είμαι). Τα αποτελέσματά τους, όμως, δεν εμφανίζονται στην οθόνη, αλλά γίνονται “append” στο αρχείο που ορίζουμε. Εάν δεν σας εμφάνισε το μήνυμα “Permission denied” τότε η εντολή εκτελέστηκε. Εάν σας εμφάνισε το παραπάνω μήνυμα τότε προφανώς δεν έχετε το δικαίωμα να γράψετε στο παραπάνω αρχείο. Δοκιμάστε την παραπάνω εντολή και σε αρχεία άλλων χρηστών μέχρι να καταφέρετε να γράψετε σε αρχεία δυο ή τριών τουλάχιστον συμφοιτητών σας.

```
Εντολή >> αρχείο
```

Εκτελεί την εντολή. Τα αποτελέσματα όμως της εντολής δεν εμφανίζονται στο τερματικό αλλά αποθηκεύονται στο αρχείο. Αν το αρχείο υπάρχει ήδη, τότε τα αποτελέσματα της εντολής γράφονται στο τέλος του αρχείου (append).

1. Κάθε φορά που δοκιμάζετε την παραπάνω εντολή σε ένα αρχείο, δοκιμάζετε αν μπορείτε να δείτε και τα περιεχόμενα του αρχείου με την εντολή cat. Ελέγξτε τις άδειες χρήσης του κάθε αρχείου.
2. Δοκιμάστε αν μπορείτε να αλλάξετε τις άδειες χρήσης ενός αρχείου myid που ανήκει σε άλλον χρήστη.
3. Πόσοι (και ποιοι) χρήστες κατάφεραν να «πειράξουν» το δικό σας αρχείο; Δείτε τα περιεχόμενά του.
4. Μετονομάστε το αρχείο σας σε myid2.

Ασκηση 3

Εξοικειώση με τον επεξεργαστή κειμένου vi. Δείτε το [Παράρτημα Α](#) για οδηγίες χρήσης του vi.

1. Εκτελέστε την εντολή:

```
vi myid2
```

Μπορείτε να τροποποιήσετε τα περιεχόμενα αυτού του αρχείου.

2. Εκτελέστε την εντολή:

```
vi
```

Εισάγετε κάποιο κείμενο και αποθηκεύστε το με όνομα της αρεσκείας σας στο φάκελο /tmp. Σημείωση: η εντολή “:w filename” στο filename μπορεί να δεχθεί είτε σχετική είτε απόλυτη διαδρομή για ένα όνομα αρχείου.

5.5 Αναφορές

- [1] GNU Linux Programmer's Manual: umask structure.
- [2] GNU coreutils Reference Manual: chmod command.
- [3] Andrew S. Tanenbaum, *Modern Operating Systems*, 3rd edition, page 279.

- [4] Brian Fox, Chet Ramey. *GNU Bash, General Commands Manual*.
- [5] Maurice J. Bach. *The Design of the Unix Operating System*, 1986.
- [6] WikiPedia. inode, February 2015. [<http://en.wikipedia.org/wiki/Inode>]