ΜΑΘΗΜΑ 5ο - Τερματικό II

0. Μερικές απλές εντολές
1. Ανάγνωση αρχείων και ανακατεύθυνση εισόδου & εξόδου εντολών
2. Αναζήτηση αρχείων
3. Συμβολικοί και σκληροί σύνδεσμοι (soft links & hard links)
4. Συμπίεση αρχείων
5. Διεργασίες (Processes)
6. Δικαιώματα αρχείων & καταλόγων
7. Αλλαγή δικαιωμάτων αρχείων & καταλόγων (`chmod`)
8. Αλλαγή σε άλλον χρήστη
9. Αλλαγή ιδιοκτησίας αρχείων (`chown`)
10. Διαχειριστές πακέτων: Δύο επίπεδα

---

0. Μερικές απλές εντολές

`whoami` (ποιος είμαι)
`who` (ποιοι είναι συνδεδεμένοι στο σύστημα, πότε συνδέθηκαν τελευταία φορά)
`last` (ποιοι συνδέθηκαν το τελευταίο διάστημα στο σύστημα)
`id greeklug` (πληροφορίες για τον χρήστη)
`uname -a` (πληροφορίες σχετικά με τον πυρήνα)
`lsb_release -a` (πληροφορίες σχετικά με την έκδοση της διανομής)

`du -ha` για εκτίμηση υπολογισμού χώρου των αρχείων
`h` - human readable
`a` - all files, not just directories

`file file1.txt` (Προσδιορισμός του τύπου αρχείου)

---

1. Ανάγνωση αρχείων και ανακατεύθυνση εισόδου & εξόδου εντολών

Τα προγράμματα (εντολές), παράγουν ένα αποτέλεσμα (μια **έξοδο**). Η έξοδος μπορεί να είναι:

- **Τα αποτελέσματα του προγράμματος**, δηλαδή τα δεδομένα που έχει σχεδιαστεί να παράγει το πρόγραμμα (**standard output** ή **stdout**, **file descriptor: 1**)

- **Μηνύματα κατάστασης και σφαλμάτων**, που μας ενημερώνουν για την κατάσταση εκτέλεσης του προγράμματος (**standard error** ή **stderr**, **file descriptor: 2**)

Για παράδειγμα, το `ls -l /home/greeklug` παράγει:

```
drwxr-xr-x 2 greeklug greeklug 4096 Apr  3 00:59 Desktop
drwxr-xr-x 8 greeklug greeklug 4096 Apr  3 20:00 Documents
drwxr-xr-x 4 greeklug greeklug 4096 Apr  3 19:54 Downloads
drwxrwxr-x 4 greeklug greeklug 4096 Apr  6 00:33 GreekLUG
drwxr-xr-x 2 greeklug greeklug 4096 Apr  3 00:59 Music
drwxr-xr-x 2 greeklug greeklug 4096 Apr  3 00:59 Pictures
drwxr-xr-x 2 greeklug greeklug 4096 Apr  3 00:59 Public
drwxr-xr-x 2 greeklug greeklug 4096 Apr  3 00:59 Templates
drwxr-xr-x 2 greeklug greeklug 4096 Apr  3 00:59 Videos
```

Το παραπάνω αποτέλεσμα (**stdout**), από προεπιλογή δεν αποθηκεύεται σε κάποιο αρχείο, αλλά προβάλλεται στην οθόνη.

Επίσης, οι εντολές λαμβάνουν είσοδο από έναν μηχανισμό που ονομάζεται **standard input** (**stdin**- **file descriptor: 0**), ο οποίος είναι συνδεδεμένος με το πληκτρολόγιο.

Με την **ανακατεύθυνση εισόδου/εξόδου (I/O redirection)** μπορούμε να αλλάξουμε πού πηγαίνει η έξοδος και από πού προέρχεται η είσοδος.

1α] Ανακατεύθυνση εξόδου (standard output), file descriptor: 1

Για να ανακατευθύνουμε την έξοδο σε κάποιο αρχείο αντί της οθόνης, χρησιμοποιούμε το σύμβολο `1>` . Για λόγους απλοποίησης το γράφουμε ως `>`

Παράδειγμα:

i]
`ls -l /home/greeklug > file-ls.txt` echo hello > hello.txt `Το` file-ls.txt `περιέχει τώρα το αποτέλεσμα της εντολής` ls -l /home/greeklug`

ii]
Αν τώρα θέλουμε να κάνουμε ανακατεύθυνση το αποτέλεσμα της εντολής `ls -l /home/member` πάλι στο ίδιο αρχείο `file-ls.txt` **που ήδη περιέχει** το output της εντολής `ls -ls /home/greeklug` τότε θα δούμε ότι με την εντολή:
`ls -l /home/member > file-ls.txt` σβήνεται το υπάρχον περιεχόμενο και γράφεται μόνο το νέο.

iii]
Για να προσθέσουμε περιεχόμενο (append) στο `file-ls.txt` τότε θα πρέπει να χρησιμοποιήσουμε το σύμβολο `>>` για την ανακατεύθυνση:
`ls -l /home/member >> file-ls.txt`
`echo how are you? >> hello.txt`

Μπορούμε να το διαβάσουμε χρησιμοποιώντας είτε την `cat` (προβολή αρχείων με μικρό περιεχόμενο) , είτε την `less` (χρησιμοποιείται σε αρχεία με μεγάλο περιεχόμενο), ή να προβάλλουμε μέρος των αρχείων (`head` , `tail`)

`cat file-ls.txt` (προβολή όλου του αρχείου)
`head file-ls.txt` (προβολή των πρώτων 10 γραμμών του αρχείου)
`head -15 file-ls.txt` (προβολή των πρώτων 15 γραμμών του αρχείου)
`tail file-ls.txt` (προβολή των τελευταίων 10 γραμμών του αρχείου)
`tail -15 file-ls.txt` (προβολή των τελευταίων 15 γραμμών του αρχείου)
`less file-ls.txt` (προβολή όλου του αρχείου σε pages)

1β] Ανακατεύθυνση σφαλμάτων (standard error), file descriptor: 2

Για να ανακατευθύνουμε το **standard error**, θα χρησιμοποιήσουμε τον **file descriptor** `2` , ώστε να αναγνωρίζεται από το shell.

Παράδειγμα:

Χρησιμοποιούμε την εντολή `ls -l /home/member` ώστε να προβάλλουμε κάτι στο οποίο δεν έχουμε πρόσβαση, ώστε να παραχθεί σφάλμα:
`ls: cannot open directory '/home/member/': Permission denied`

Θα χρησιμοποιήσουμε τον file descriptor `2` που συμβολίζεται ως `2>` για να στείλουμε το σφάλμα σε ένα αρχείο:
`ls -l /home/member 2> file-ls-error.txt`

1γ] Ταυτόχρονη ανακατεύθυνση εξόδου & σφαλμάτων

Μπορούμε να ανακατευθύνουμε εξίσου το **standard output** και **standard error** ταυτόχρονα, είτε σε διαφορετικά μέρη είτε σε ένα κοινό.

Παράδειγμα διαφορετικών ανακατευθύνσεων:
`ls /home/member > file-ls.txt 2> file-ls-error.txt`

Για την κοινή ανακατεύθυνση χρησιμοποιούμε το όρισμα `2>&1` στο τέλος της εντολής ώστε να συνδυάσουμε τις δύο εξόδους σε μία κοινή.
Πρακτικά αντιστοιχεί στην ανακατεύθυνση του standard error (2) στο ίδιο μέρος με το standard output (1).

Παράδειγμα κοινής ανακατεύθυνσης:
`ls /home/member > file-output.txt 2>&1`

1δ] Ανακατεύθυνση εισόδου (standard input), file descriptor: 0

Χρησιμοποιούμε το σύμβολο `<` για να ανακατευθύνουμε το standard input.

Παράδειγμα:
`cat < file1.txt`

Μπορούμε καλύτερα να καταλάβουμε το standard input μέσω ενός χαρακτηριστικού του shell που ονομάζεται pipelines (`|`)
Το `|` όταν χρησιμοποιείται ανάμεσα σε 2 εντολές, παίρνει το output της 1ης και το δίνει ως input στην 2η.

Παραδείγματα:
```
du -h /home/savvas/ | less
du -h /home/savvas/ | sort | less
ls /home/greeklug/Documents/ /home/greeklug/Downloads/ | sort | less
```

`sort` sorts in alphabetical order
`sort -n` sorts in numerical order
`sort -k3` sorts based on column 3

`grep` - print lines that match patterns
```
ls /home/greeklug/Documents/ /home/greeklug/Downloads/ | sort | grep ".txt"
```

```
diff file1.txt file2.txt
```

---

2. Αναζήτηση αρχείων

2α] locate

Για να χρησιμοποιήσουμε την locate, ανάλογα τη διανομή μας, κάνουμε εγκατάσταση τα πακέτα `mlocate` ή `plocate`.

Η εντολή **locate** χρησιμοποιεί μια βάση δεδομένων που δημιουργείται από ένα εργαλείο, το **updatedb**. Τα περισσότερα συστήματα Linux εκτελούν αυτόματα αυτή τη διαδικασία μία φορά την ημέρα μέσω cron ή υπηρεσίας συστήματος. Ωστόσο, μπορούμε να την ενημερώσουμε οποιαδήποτε στιγμή απλώς εκτελώντας το, προτού αναζητήσουμε κάτι με την locate:
```
sudo updatedb
```

Σύνταξη:
locate file/dir

Παράδειγμα
```
locate file.txt
locate greeklug | grep "2025"
locate greeklug-*
```

2β] find

Η εντολή **find** είναι ένα εξαιρετικά χρήσιμο και συχνά χρησιμοποιούμενο εργαλείο. Αναζητά αναδρομικά μέσα στο δέντρο του συστήματος αρχείων ξεκινώντας από έναν συγκεκριμένο κατάλογο και εντοπίζει αρχεία που πληρούν συγκεκριμένες συνθήκες. Ο προεπιλεγμένος κατάλογος αναζήτησης είναι πάντα ο τρέχων κατάλογος εργασίας.

Σύνταξη:
find /path/directory search_parametres (**Πρώτα ορίζω τον κατάλογο αναζήτησης και μετά τις παραμέτρους αναζήτησης**)

Options	Λειτουργία
`-name`	Ορισμός συγκεκριμένου ονόματος αρχείου/καταλόγου
`-type f`	Ορισμός τύπου αρχείου αναζήτησης ως αρχείο (file)
`-type d`	Ορισμός τύπου αρχείου αναζήτησης ως κατάλογο (directory)
`-ctime [+/-]n`	Ορισμός χρόνου δημιουργίας αρχείου (created). Το `n` αντιπροσωπεύει μέρες
`-atime [+/-]n`	Ορισμός χρόνου πρόσβασης/ανάγνωσης αρχείου (accessed). Το `n` αντιπροσωπεύει μέρες
`-mtime [+/-]n`	Ορισμός χρόνου μετατροπής/εγγραφής αρχείου (modified). Το `n` αντιπροσωπεύει μέρες
`-cmin [+/-]m`	Ορισμός χρόνου δημιουργίας αρχείου (created). Το `m` αντιπροσωπεύει λεπτά
`-amin [+/-]m`	Ορισμός χρόνου πρόσβασης/ανάγνωσης αρχείου (accessed). Το `m` αντιπροσωπεύει λεπτά
`-mmin [+/-]m`	Ορισμός χρόνου δημιουργίας αρχείου (created). Το `m` αντιπροσωπεύει λεπτά
`-size [+/-]n`	Ορισμός μεγέθους αρχείου
`-exec command {} ";"`	Εκτελεί μια εντολή σε κάθε αρχείο που βρίσκει

Options για χρήση με την find

- Σχετικά με το option `-atime n`, `-mtime n` κτλ, μπορούμε να καθορίσουμε χρονικές τιμές (**n**), **+n** ή **-n**.
- `n` σημαίνει ακριβώς τόσες μέρες, `+n` σημαίνει πάνω από τόσες μέρες και `-n` σημαίνει κάτω από τόσες μέρες
- Σχετικά με το option `-size` μπορούμε να καθορίσουμε μονάδες όπως bytes (**c**), kilobytes (**k**), megabytes (**M**), gigabytes (**G**) κ.λπ. Όπως και με τις χρονικές τιμές που αναφέρθηκαν παραπάνω, τα μεγέθη αρχείων μπορούν να είναι ακριβείς αριθμοί (**n**), **+n** ή **-n**.

Παραδείγματα:

`find ~/ -type f -name "*".txt`
Αναζητεί μέσα στο `/home/greeklug` όλα τα αρχεία .txt

`find /home/greeklug -type f -name greeklug*.txt`
Αναζητεί μέσα στο `/home/greeklug` όλα τα αρχεία .txt που ξεκινούν με το όνομα greeklug

και καταλήγουν σε .txt

`find /home -type d -name greeklug`
Αναζητεί μέσα στο `/home/` όλους τους καταλόγους με το όνομα greeklug

`find /home/greeklug/Videos -type f -size +1000M`
Αναζητεί μέσα στο `/home/greeklug/Videos` όλα τα αρχεία που έχουν μεγαλύτερο μέγεθος από 1000MB

`find /home/greeklug/.cache/mozilla/* -exec rm -r {} ";"`
Αναζητεί μέσα στο `/home/greeklug/.cache/mozilla/` όλα τα αρχεία και τα διαγράφει
(!) προσοχή με τις διαγραφές (!)

---

3. Συμβολικοί και σκληροί σύνδεσμοι (soft links & hard links)

3α] Δημιουργία και διαχείριση σκληρών συνδέσμων (hard links)

Χρησιμοποιούμε την εντολή `stat` για να δούμε **περισσότερες πληροφορίες** για ένα αρχείο, όπως τον **αριθμό inode** και τους **σκληρούς συνδέσμους** του:

`stat file.txt`

Το Inode είναι μια δομή δεδομένων σε μορφή μοναδικού αριθμού, στο οποίο περιέχεται πού είναι αποθηκευμένο το αρχείο ή κατάλογος στον δίσκο (σε ποια blocks), και **πληροφορίες**, όπως τα **δικαιώματα**, η **ώρα πρόσβασης**, η **ώρα τροποποίησης**, κ.ά.

Όταν δημιουργούμε ένα αρχείο, το σύστημα αρχείων ομαδοποιεί όλα τις πληροφορίες του αρχείου σε ένα Inode και το συνδέει με **σκληρό σύνδεσμο με το όνομα του αρχείου**.
Inode: 2130106 → σκληρός σύνδεσμος → file.txt

Για να φτιάξω έναν σκληρό σύνδεσμο (`file-hard.txt`) ενός υπάρχοντος αρχείου (`file.txt`) κάνω:
`ln /home/greeklug/file.txt /home/greeklug/Documents/file-hard.txt`

Με αυτό δημιουργώ ένα νέο αρχείο που μοιράζεται το ίδιο περιεχόμενο με το αρχικό αρχείο (προσοχή δεν μιλάμε για αντίγραφο αλλά δύο αρχεία που έχουν τα ίδια δεδομένα).
Μπορώ να τροποποιήσω οποιοδήποτε από τα δύο αρχεία και αυτόματα το περιεχόμενο, με βάση ότι είναι κοινό, θα υπάρχει και στα δύο αρχεία.

Εκτελώντας την εντολή `stat` , βλέπουμε ότι τα `file.txt` και `file-hard.txt` **μοιράζονται το ίδιο Inode**, αλλά έχουν **διαφορετικά ονόματα** και βρίσκονται σε **διαφορετικές τοποθεσίες**.
Τα δεδομένα αποθηκεύονται μόνο μία φορά, αλλά είναι προσβάσιμα από πολλές θέσεις.
`stat file.txt`
`stat file-hard.txt`

Αν διαγράψουμε το `file.txt`, το `file-hard.txt` δεν θα διαγραφεί, επειδή τα δεδομένα δεν διαγράφονται άμεσα. Το μόνο που αλλάζει είναι ότι το Inode θα έχει πλέον έναν σκληρό σύνδεσμο αντί για δύο.
Αν διαγραφεί και το `file-hard.txt`, τότε το Inode δεν θα έχει κανένα σύνδεσμο και τα δεδομένα θα χαθούν από το σύστημα αρχείων.

Περιορισμοί σκληρών συνδέσμων:

- Δεν μπορούμε να δημιουργήσουμε σκληρούς συνδέσμους για **καταλόγους (directories)** - μόνο για κανονικά αρχεία.
- Δεν μπορούμε να δημιουργήσουμε σκληρούς συνδέσμους **σε διαφορετικά συστήματα αρχείων**.

3β] Δημιουργία και διαχείριση συμβολικών συνδέσμων (soft links or symlinks)

Ένας **συμβολικός σύνδεσμος** είναι μια συντόμευση (shortcut) που δείχνει στο αρχείο, το οποίο με τη σειρά του δείχνει στο Inode:
soft link → αρχείο → Inode

Για να φτιάξω μια συντόμευση (`file-sym.txt`) ενός υπάρχοντος αρχείου (`file.txt`) κάνω:
`ln -s /home/greeklug/file.txt /home/greeklug/Documents/file-sym.txt`

Με αυτό δημιουργώ ένα νέο αρχείο `file-sym.txt` που `δείχνει` σε ένα υπάρχον αρχείο `file.txt`. Το αρχείο του συμβολικού δεσμού δεν περιέχει δεδομένα αλλά μέσω της συντόμευσης φορτώνει τα δεδομένα του αρχείου στόχου της συντόμευσης.
Συνεπώς, αν το αρχείο στόχος διαγραφεί, τότε ο συμβολικός δεσμός `σπάει` καθώς παραπέμπει σε κάποιο αρχείο που δεν υπάρχει πια!

Μπορούμε επίσης να δημιουργούμε **συμβολικούς συνδέσμους σε καταλόγους**, ή ακόμη και σε αρχεία/καταλόγους που βρίσκονται σε **διαφορετικά συστήματα αρχείων**.

Συνοπτικά:

- Ένας **σκληρός σύνδεσμος** συνδέει **όνομα αρχείου με δεδομένα** σε μια συσκευή αποθήκευσης.
- Ένας **συμβολικός σύνδεσμος** συνδέει **όνομα αρχείου (ή καταλόγου) με άλλο όνομα αρχείου (ή καταλόγου)**, το οποίο μετά δείχνει στα δεδομένα.

---

4. Συμπίεση αρχείων

Τα δεδομένα των αρχείων συχνά συμπιέζονται για εξοικονόμηση χώρου στον δίσκο και για μείωση του χρόνου μεταφοράς αρχείων πχ. μέσω δικτύου.
Το Linux υποστηρίζει διάφορες μεθόδους συμπίεσης, όπως:

Εντολή	Λειτουργία
`gzip`	Το πιο συχνά χρησιμοποιούμενο τύπος συμπίεσης σε Linux
`bzip2`	Παράγει αρχεία μικρότερου μεγέθους από αυτά που παράγει το **gzip**, αλλά απαιτεί περισσότερο χρόνο
`xz`	Το πιο αποδοτικό εργαλείο συμπίεσης από πλευράς χώρου σε Linux, με μεγαλύτερο χρόνο από άλλες συμπιέσεις
`zip`	Η πιο γνωστή μέθοδος συμπίεσης που υποστηρίζεται σε όλα τα λειτουργικά συστήματα. Δεν είναι τόσο αποδοτική όσο άλλες

Αυτές οι τεχνικές διαφέρουν ως προς την αποτελεσματικότητα (πόσο χώρο εξοικονομούν) και τον χρόνο που απαιτούν για τη συμπίεση ή αποσυμπίεση. Γενικά, όσο πιο αποδοτική είναι η μορφή συμπίεσης, τόσο περισσότερος είναι συνήθως ο χρόνος που χρειάζεται.

Επιπλέον, το εργαλείο **tar** χρησιμοποιείται συχνά για να ομαδοποιήσει αρχεία σε ένα αρχείο αρχειοθέτησης και στη συνέχεια να συμπιεστεί ολόκληρο το αρχείο.

Συμπίεση δεδομένων με χρήση του gzip (στηρίζεται στον αλγόριθμο DEFLATE)

Εντολή	Λειτουργία
`gzip *`	Συμπιέζει όλα τα αρχεία στον τρέχοντα κατάλογο. Κάθε αρχείο μετονομάζεται με κατάληξη **.gz**.
`gzip -r dir1`	Συμπιέζει όλα τα αρχεία στον κατάλογο `dir1`, καθώς και σε όλους τους υποκαταλόγους του.
`gunzip -r dir1`	Αποσυμπιέζει τα αρχεία που βρίσκονται μέσα στον κατάλογο `dir1` και σε όλους τους υποκαταλόγους του. Ισοδύναμο με `gzip -d`
`gunzip file.txt.gz`	Αποσυμπιέζει το αρχείο `file.txt.gz`
`zcat file.txt.gz`	Βλέπω τα περιεχόμενα του συμπιεσμένου αρχείου `file.txt.gz`, χωρίς να το αποσυμπιέσω

Συμπίεση δεδομένων με χρήση του bzip2 (στηρίζεται στον αλγόριθμο Burrows-Wheeler Transform)

Το **bzip2** παράγει μικρότερα σε μέγεθος αρχεία από το gzip, αλλά χρειάζεται περισσότερο χρόνο.

Εντολή	Λειτουργία
`bzip2 *`	Συμπιέζει όλα τα αρχεία στον τρέχοντα κατάλογο και τα μετονομάζει με κατάληξη **.bz2**.

Εντολή	Λειτουργία
`bunzip2 *.bz2`	Αποσυμπιέζει όλα τα αρχεία με κατάληξη **.bz2**. Ισοδύναμο με **bzip2 -d**.

Συμπίεση δεδομένων με χρήση του xz (στηρίζεται στον αλγόριθμο LZMA)

Το **xz** είναι το πιο αποδοτικό εργαλείο συμπίεσης ως προς τον χώρο, όμως είναι από τις πιο αργές μεθόδους συμπίεσης.

Εντολή	Λειτουργία
`xz *`	Συμπιέζει όλα τα αρχεία στον τρέχοντα κατάλογο και τα μετονομάζει με κατάληξη **.xz**.
`xz file.txt`	Συμπιέζει το `file.txt` σε `file.txt.xz` και διαγράφει το `file.txt`
`xz -d *.xz`	Αποσυμπιέζει όλα τα αρχεία με κατάληξη .xz

Χειρισμός αρχείων με χρήση του zip (στηρίζεται στον αλγόριθμο DEFLATE)

Το πρόγραμμα **zip** σπάνια χρησιμοποιείται για συμπίεση αρχείων στο Linux, αλλά μπορεί να χρειαστεί για να ανοίξουμε ή να αποσυμπιέσουμε αρχεία από άλλα λειτουργικά συστήματα, π.χ. Windows.

Εντολή	Λειτουργία
`zip /home/greeklug/dir1/greeklug.zip *`	Συμπιέζει όλα τα αρχεία μέσα στον κατάλογο `/home/greeklug/dir1/` , στο αρχείο `greeklug.zip`
`zip -r greeklug.zip ~`	Συμπιέζει τον προσωπικό κατάλογο και υποκαταλόγους στο `greeklug.zip`
`unzip greeklug.zip`	Εξάγει όλα τα αρχεία από το **greeklug.zip** στον τρέχοντα κατάλογο.
`unzip -l greeklug.zip`	Βλέπω τι περιέχει το αρχείο `greeklug.zip`

Αρχειοθέτηση και συμπίεση δεδομένων με χρήση του tar

Σε αντίθεση με το zip, συνήθως δεν χρησιμοποιούμε κατευθείαν τις εντολές `gzip` , `bzip2` , `xz` για την συμπίεση, καθώς συμπιέζουν μεμονωμένα τα αρχεία χωρίς να τα "συσκευάζουν" σε έναν φάκελο. Για να το πετύχουμε αυτό, χρησιμοποιούμε το `tar` σε συνδυασμό με τις παραπάνω μεθόδους συμπίεσης. Έτσι σε περιβάλλον Linux, συναντούμε συνήθως συμπιεσμένα αρχεία ως .tar.gz, .tar.bz2, .tar.xz κτλ.

Το **tar** σημαίνει "**t**ape **ar**chive" και επιτρέπει τη δημιουργία ή εξαγωγή αρχείων από ένα **αρχείο αρχειοθέτησης** (**tarball**). Παράλληλα, μπορεί να συμπιέσει ή να αποσυμπιέσει περιεχόμενα κατά τη δημιουργία ή εξαγωγή του αρχείου.

Σύνταξη της `tar` στο Linux:
$ tar επιλογές αρχείο-αρχειοθέτησης αρχείο
$ tar επιλογές αρχείο-αρχειοθέτησης /φάκελος

- **tar**: η εντολή.
- **επιλογές**: τροποποιούν τη συμπεριφορά της `tar`.
- **αρχείο-αρχειοθέτησης**: το όνομα του αρχείου που δημιουργούμε ή επεξεργαζόμαστε.
- **αρχείο ή φάκελος προς αρχειοθέτηση**: το περιεχόμενο που θέλουμε να συμπεριλάβουμε.

Εντολή	Λειτουργία
`tar -cvf greeklug.tar /home/greeklug/Documents`	Δημιουργία ενός tar με όνομα `greeklug.tar` που περιέχει όλα τα αρχεία μέσα στο `/home/greeklug/Documents`
`tar -cvf /home/greeklug/Desktop/greeklug.tar /home/greeklug/Documents`	Δημιουργία ενός tar με όνομα `greeklug.tar` , στην τοποθεσία `/home/greeklug/Desktop/` , που περιέχει όλα τα αρχεία μέσα στο `/home/greeklug/Documents`
`tar -tf greeklug.tar`	Προβολή των περιεχομένων του `greeklug.tar`
`tar -zcvf dir1.tar.gz dir1`	Δημιουργία tarball και συμπίεση χρησιμοποιώντας την `gzip` , των περιεχομένων του `dir1` με όνομα `dir1.tar.gz`
`tar -jcvf dir1.tar.bz2 dir1`	Δημιουργία tarball και συμπίεση χρησιμοποιώντας την `bzip2` , των περιεχομένων του `dir1` με όνομα `dir1.tar.bz2`
`tar -Jcvf dir1.tar.xz dir1`	Δημιουργία tarball και συμπίεση χρησιμοποιώντας την `xz` , των περιεχομένων του `dir1` με όνομα `dir1.tar.bz2`
`tar -xvf dir1.tar.gz`	Εξαγωγή του αρχείου `dir1.tar.gz` , στον τρέχον κατάλογο
`tar -xvf dir1.tar.gz -C /home/greeklug/Documents`	Εξαγωγή του αρχείου `dir1.tar.gz` μέσα στον κατάλογο `/home/greeklug/Documents`

Εντολή	Λειτουργία
`tar -rvf greeklug.tar file1.txt`	Προσθήκη του αρχείου `file1.txt` στο ήδη υπάρχον `greeklug.tar`
`tar -uvf greeklug.tar dir1`	Προσθήκη στο `greeklug.tar` μόνο των νέων αρχείων του `dir` που δεν είχαν προστεθεί προηγουμένως

Option	Περιγραφή
`-c`	(**create**) Δημιουργία νέου αρχείου αρχειοθέτησης
`-x`	(**extract**) Εξαγωγή από αρχείο
`-f`	(**file**) Ορισμός ονόματος αρχείου
`-v`	(**verbose**) Λεπτομερής έξοδος
`-A`	(**concatenate**) Συνένωση πολλών tarball
`-z`	(**gzip**) Συμπίεση με gzip
`-j`	(**bzip2**) Συμπίεση με bzip2
`-J`	(**xz**) Συμπίεση με xz
`-W`	(**verify**) Έλεγχος ακεραιότητας μετά την εγγραφή
`-t`	(**list**) Προβολή περιεχομένων
`-u`	(**update**) Προσθήκη μόνο των νέων αρχείων σε ήδη υπάρχων tarball
`-C`	(**directory**) Για εκτέλεση ενεργειών σε οριζόμενη τοποθεσία
`-r`	(**append**) Προσθήκη άλλων αρχείων σε ήδη υπάρχων tarball

---

5. Διεργασίες (Processes)

5α] Εισαγωγή

- Οι διεργασίες αποτελούν μέρος μιας εντολής ή προγράμματος που τρέχουν στο Η/Υ μας και απασχολούν πόρους του συστήματος (τον επεξεργαστή και την μνήμη RAM, τους δίσκους, την κάρτα δικτύου κτλ).
- Μια μόνο εντολή μπορεί να ξεκινήσει πολλές διεργασίες ταυτόχρονα. Ορισμένες διεργασίες είναι ανεξάρτητες μεταξύ τους, ενώ άλλες σχετίζονται. Μια αποτυχία σε μία διεργασία μπορεί, ή και να μη, επηρεάσει τις υπόλοιπες που εκτελούνται στο σύστημα.
- Το λειτουργικό σύστημα (και ειδικότερα ο πυρήνας) είναι υπεύθυνο για την κατανομή των πόρων του συστήματος σε κάθε διεργασία και για τη διασφάλιση της βέλτιστης λειτουργίας.

5β] Τύποι διεργασιών

Τύποι διεργασιών	Περιγραφή	Παράδειγμα
Διεργασίες χρηστών	Οι διεργασίες που ξεκινούν από τους χρήστες του συστήματος.	**bash**, **firefox**, **top**, **thunderbird**, **libreoffice**
Daemons/Services	Υπηρεσίες συστήματος που εκτελούνται συνεχώς. Πολλές από αυτές ξεκινούν κατά την εκκίνηση του συστήματος και στη συνέχεια περιμένουν ένα αίτημα από τον χρήστη ή το σύστημα που να υποδεικνύει ότι απαιτείται η υπηρεσία τους.	**httpd**, **sshd**, **libvirtd**, **cupsd**
Νήματα (threads)	Πρόκειται για εργασίες που εκτελούνται κάτω από μια κύρια διεργασία, μοιράζονται τη μνήμη και άλλους πόρους, αλλά προγραμματίζονται και εκτελούνται από το σύστημα ανεξάρτητα. Ένα μεμονωμένο νήμα μπορεί να τερματιστεί χωρίς να τερματιστεί ολόκληρη η διεργασία, και μια διεργασία μπορεί να δημιουργήσει νέα νήματα οποιαδήποτε στιγμή. Πολλά προγράμματα είναι multi-threaded.	**dconf-service**, **gnome-terminal-server**
Νήματα πυρήνα (Kernel Threads)	Εργασίες του πυρήνα που οι χρήστες έχουν ελάχιστο έλεγχο. Μπορεί να εκτελούν ενέργειες όπως η μεταφορά ενός νήματος από έναν επεξεργαστή σε έναν άλλο ή η διασφάλιση ότι οι λειτουργίες εισόδου/εξόδου προς τον δίσκο ολοκληρώνονται.	**kthreadd**, **migration**, **ksoftirqd**

Τύποι διεργασιών

5γ] Προβολή διεργασιών

Εντολή	Λειτουργία
`ps -u greeklug`	Προβολή των διεργασιών του χρήστη `greeklug`
`ps -e`	Προβολή όλων των διεργασιών
`pstree`	Προβολή όλων των διεργασιών σε μορφή "δέντρου"
`pstree -p`	Προβολή όλων των διεργασιών σε μορφή "δέντρου", με το PID
`w`	Προβολή των συνδεδεμένων χρηστών και τι κάνουν
`top`	δυναμική προβολή των διεργασιών ανά 3"
`top -d = 30`	δυναμική προβολή των διεργασιών ανά 30"
`htop`	διαδραστικό πρόγραμμα προβολής διεργασιών

Εντολές προβολής διεργασιών

- Ανά πάσα στιγμή, εκτελούνται πολλαπλές διεργασίες. Το λειτουργικό σύστημα τις παρακολουθεί αναθέτοντας σε καθεμία έναν μοναδικό αριθμό αναγνώρισης διεργασίας (**PID**). Το PID χρησιμοποιείται για την παρακολούθηση της κατάστασης της διεργασίας, της χρήσης της CPU, της χρήσης μνήμης, της ακριβούς τοποθεσίας των πόρων στη μνήμη κα.
- Τα νέα PID αποδίδονται με αύξουσα σειρά καθώς γεννιούνται νέες διεργασίες. Το PID 1 αντιστοιχεί στη διεργασία **init** (systemd).
- Μια κρίσιμη λειτουργία του πυρήνα, που ονομάζεται **προγραμματιστής εργασιών (scheduler)**, μετακινεί συνεχώς διεργασίες εντός και εκτός του επεξεργαστή (CPU). Μοιράζει το χρόνο χρήσης του επεξεργαστή με βάση τη σχετική προτεραιότητα της κάθε διεργασίας.

Κατάσταση διεργασιών	**Περιγραφή**
κατάσταση εκτέλεσης (running)	Διεργασίες που εκτελούν αυτή τη στιγμή εντολές σε έναν επεξεργαστή, ή περιμένουν να τις παραχωρηθούν ένα μερίδιο χρόνου για να εκτελεστούν
κατάσταση ύπνου (sleep)	Διεργασίες που περιμένουν κάτι να συμβεί για να συνεχίσουν, πχ να πληκτρολογήσει κάτι ο χρήστης
κατάσταση zombie	Διεργασίες που έχουν ολοκληρώσει την εκτέλεσή τους, αλλά παραμένουν στη λίστα διεργασιών επειδή η γονική διεργασία (parent process) δεν έχει διαβάσει την έξοδό τους

Κατάσταση διεργασιών

Ανάγνωση του αποτελέσματος της εντολής `top`

```
top - 12:31:13 up 2 days, 22:35,  1 user,  load average: 0,48, 0,37, 0,44
Tasks: 285 total,   1 running, 282 sleeping,   0 stopped,   2 zombie
%Cpu(s):  1,8 us,  0,7 sy,  0,0 ni, 97,6 id,  0,0 wa,  0,0 hi,  0,0 si,
0,0 st
MiB Mem :  15875,1 total,   4411,5 free,   3444,4 used,   8581,2
buff/cache
MiB Swap:   1956,0 total,   1397,7 free,    558,2 used.  12430,6 avail Mem
```

1η γραμμή: τρέχουσα ώρα, uptime, αριθμός συνδεδεμένων χρηστών, load average
Το **load average** είναι η μέση χρήση του επεξεργαστή από διεργασίες σε ένα συγκεκριμένο χρονικό διάστημα. Η εκτέλεση των διεργασιών μοιράζεται σε όλα τα νήματα του επεξεργαστή. Για παράδειγμα, αν ένας επεξεργαστής έχει 6 πυρήνες (cores) και 12 νήματα (threads), το load average είναι η ποσοστιαία χρήση του επεξεργαστή διά 12. Αν ένας επεξεργαστής έχει 4 νήματα και το load average είναι 4, τότε αυτό σημαίνει ότι το σύστημα

έχει πρόβλημα, καθώς ο επεξεργαστής χρησιμοποιείται κατά 100%!
Στο παραπάνω στιγμιότυπο, το load average σημαίνει ζητούσαν χρήση CPU το τελευταίο λεπτό 0,48 διεργασίες, 0,37 διεργασίες τα τελευταία 5' και 0,44 διεργασίες τα τελευταία 15'.

2η γραμμή: Συνολικός αριθμός των διεργασιών που είναι ενεργές.
3η γραμμή: Χρήση του επεξεργαστή από τον χρήστη (us), από τον πυρήνα (sy), ποσοστό διεργασιών με χαμηλή προτεραιότητα (ni), ποσοστό idle (id), ποσοστό διεργασιών που περιμένουν (wa).
4η γραμμή: Πληροφορίες και χρήση της μνήμης RAM.
5γραμμή: Πληροφορίες και χρήση της μνήμης Swap (χώρος στον δίσκο, που ενεργοποιείται όταν η μνήμη RAM δεν είναι αρκετή).

```
    PID  USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+
COMMAND
 213888 savvas    20   0   14576   5632   3456 R  15,4   0,0   0:00.04 top
   1855 savvas    20   0  898800  80404  43828 S   7,7   0,5   9:47.92
mintmenu        1 root      20   0   23112  13744   9136 S   0,0   0,1
0:13.45 systemd         2 root      20   0       0      0      0 S   0,0
0,0   0:00.13 kthreadd
```

- Ανά πάσα στιγμή, πολλές διεργασίες εκτελούνται (δηλαδή βρίσκονται στην ουρά εκτέλεσης) στο σύστημα. Ωστόσο, ένας επεξεργαστής μπορεί στην πραγματικότητα να εκτελεί μόνο μία εργασία τη φορά. Ορισμένες διεργασίες είναι πιο σημαντικές από άλλες, επομένως το Linux επιτρέπει τον καθορισμό και τη διαχείριση της προτεραιότητας των διεργασιών. Οι διεργασίες με υψηλότερη προτεραιότητα έχουν προνομιακή πρόσβαση στον επεξεργαστή.
- Η προτεραιότητα μιας διεργασίας μπορεί να καθοριστεί με τον ορισμό μιας **τιμής nice** (στήλη `NI` στο παραπάνω στιγμιότυπο), για τη συγκεκριμένη διεργασία.
- Όσο μικρότερη είναι η τιμή nice, τόσο υψηλότερη είναι η προτεραιότητα. Χαμηλές τιμές αποδίδονται σε σημαντικές διεργασίες, ενώ υψηλές τιμές σε διεργασίες που μπορούν να περιμένουν περισσότερο. Μια διεργασία με υψηλή τιμή nice απλώς επιτρέπει σε άλλες διεργασίες να εκτελεστούν πρώτες. Στο Linux, μια τιμή nice **-20** αντιπροσωπεύει τη μέγιστη προτεραιότητα και η τιμή **+19** τη χαμηλότερη.

5δ] Διαχείριση διεργασιών

- Μπορούμε να εκτελέσουμε μια διεργασία απευθείας από το τερματικό, πχ ανοίγουμε το πρόγραμμα geany
  `geany`
- Το πρόγραμμα τώρα είναι στο foreground και δεν μπορούμε να εκτελέσουμε/γράψουμε άλλη εντολή στο τερματικό.
- Πληκτρολογούμε `Ctrl + Z` για να βάλουμε την διεργασία σε παύση. Μέσα σε αγκύλη βλέπουμε το PID της (πχ PID = 2).

- Πληκτρολογώντας `bg` βάζουμε την διεργασία στο background και μπορούμε πάλι να εκτελέσουμε και άλλες εντολές στο τερματικό.
- Πληκτρολογώντας `fg` επαναφέρουμε την διεργασία στο foreground.
- Πληκτρολογώντας `sudo renice +10 2` αυξάνουμε το nice της διεργασίας 2 κατά 10, μειώνοντας την προτεραιότητά της.
- Πληκτρολογώντας `sudo kill -15 2` στέλνουμε αίτημα ώστε η διεργασία 2 να τερματιστεί με ευγενικό τρόπο, δίνοντάς της την ευκαιρία να καθαρίσει τους πόρους της και να αποθηκεύσει τυχόν δεδομένα αν χρειάζεται.
- Πληκτρολογώντας `sudo kill -9 2` τερματίζουμε βίαια την διεργασία 2.

---

6. Δικαιώματα αρχείων & καταλόγων

Σε ένα σύστημα Linux, μπορούν να συνδέονται ταυτόχρονα πολλοί χρήστες. Η πρόσβασή τους στηρίζεται σε ένα σύστημα **δικαιωμάτων** με βάση το οποίο παράγονται διάφορα επίπεδα πρόσβασης στα δεδομένα του συστήματος.

Έτσι για να μην μπορεί κάποιος **απλός** χρήστης να δει τα αρχεία κάποιου άλλου χρήστη, κάθε αρχείο και κατάλογος συνοδεύεται από συγκεκριμένα δικαιώματα, που ορίζουν ποιος χρήστης έχει πρόσβαση σε αυτά και αποτρέπουν τους υπόλοιπους από το να τα χειριστούνε.

Παράδειγμα
`cat /etc/sudoers`
`cat: /etc/sudoers: Permission denied`
Ως απλοί χρήστες, δεν μπορούμε να προβάλλουμε το αρχείο `sudoers`

`ls -lah /home/greeklug`

```
-rw-------  1 greeklug greeklug  368 Apr  8 19:04 .bash_history
-rw-r--r--  1 greeklug greeklug  220 Apr  3 00:43 .bash_logout
-rw-r--r--  1 greeklug greeklug 3,7K Apr  3 00:43 .bashrc
drwx------ 11 greeklug greeklug 4,0K Apr  3 16:39 .cache
drwxr-xr-x 16 greeklug greeklug 4,0K Apr  3 18:48 .config
drwxr-xr-x  2 greeklug greeklug 4,0K Apr  3 00:59 Desktop
-rw-r--r--  1 greeklug greeklug   27 Apr  3 00:59 .dmrc
drwxr-xr-x  8 greeklug greeklug 4,0K Apr  3 20:00 Documents
drwxr-xr-x  4 greeklug greeklug 4,0K Apr  3 19:54 Downloads
```

Πεδίο	**Έννοια**
`-rw-r--r--`	Δικαιώματα πρόσβασης στο αρχείο
`1`	Αριθμός hard link του αρχείου
`greeklug`	Ιδιοκτήτης (χρήστης στον οποίο ανήκει το αρχείο)
`greeklug`	Ομάδα (όνομα του group στο οποίο ανήκει το αρχείο)

Πεδίο	Έννοια
`3,7K`	Μέγεθος αρχείου
`Apr  3 00:43`	Ημερομηνία τροποποίησης
`.bashrc`	Όνομα αρχείου

- `-rw-r--r--` Όταν τα δικαιώματα ξεκινούν με `-` , σημαίνει ότι αυτό είναι αρχείο
- `drwxr-xr-x` Όταν τα δικαιώματα ξεκινούν με `d` , σημαίνει ότι αυτό είναι κατάλογος (directory)
- `lrwxrwxrwx` Όταν τα δικαιώματα ξεκινούν με `l` , σημαίνει ότι αυτό είναι συμβολικός σύνδεσμος (symlink)

Τα δικαιώματα ενός αρχείου ή καταλόγου αποτελούνται από τρεις τριάδες `rwxrwxrwx`

1. Η 1η τριάδα αφορά τα δικαιώματα του **ιδιοκτήτη** (user) του αρχείου ή καταλόγου (συνήθως αυτός που δημιούργησε το αρχείο ή κατάλογο).
2. Η 2η τριάδα αφορά τα δικαιώματα της **ομάδας** (group) στο οποίο ανήκουν κάποιοι χρήστες του συστήματος.
3. Η 3η τριάδα αφορά τα δικαιώματα για **όλους τους υπόλοιπους** (others) (άλλους χρήστες που δεν είναι ιδιοκτήτες του αρχείου, ούτε ανήκουν στο group).

Δικαίωμα	Αρχεία	Κατάλογοι
`r`	Επιτρέπει την ανάγνωση του αρχείου	Επιτρέπει την προβολή των περιεχομένων του καταλόγου, **αν είναι ενεργοποιημένο και το** `x`
`w`	Επιτρέπει να γράψουμε μέσα στο αρχείο	Επιτρέπει την δημιουργία, μετονομασία και διαγραφή αρχείων μέσα στον κατάλογο, **αν είναι ενεργοποιημένο και το** `x`
`x`	Επιτρέπει την εκτέλεση του αρχείου (πχ πρόγραμμα)	Επιτρέπει την πρόσβαση μέσα σε έναν κατάλογο, εφόσον έχουμε πρόσβαση ως ιδιοκτήτης ή μέσω της ομάδας

Παραδείγματα:

Α]
`-rw-r--r-- 1 greeklug greeklug 3,7K Apr 3 00:43 .bashrc`
Στο παραπάνω αρχείο `.bashrc` βλέπουμε:

1. `rw-` Ο χρήστης `greeklug` μπορεί να διαβάσει και να γράψει μέσα στο αρχείο
2. `r--` Όσοι ανήκουν στο group `greeklug` , μπορούν να διαβάσουν το αρχείο
3. `r--` Όλοι οι υπόλοιποι, μπορούν να διαβάσουν το αρχείο

Β]
`-rw-rw-r-- 1 greeklug greeklug 76 Apr 3 19:51 file35.txt`

Στο παραπάνω αρχείο `file35.txt` βλέπουμε:

1. `rw-` Ο χρήστης `greeklug` μπορεί να διαβάσει και να γράψει μέσα στο αρχείο
2. `rw-` Όσοι ανήκουν στο group `greeklug`, μπορούν να διαβάσουν και να γράψουν μέσα στο αρχείο
3. `r--` Όλοι οι υπόλοιποι, μπορούν να διαβάσουν το αρχείο

Γ]
`drwxr-x--- 17 greeklug greeklug 4,0K Apr 8 18:44 greeklug`
Στον παραπάνω κατάλογο `greeklug` βλέπουμε:

1. `rwx` Ο χρήστης `greeklug` μπορεί να προβάλλει και να επεξεργαστεί τα αρχεία μέσα στον κατάλογο `Documents`
2. `r-x` Όσοι ανήκουν στο group `greeklug`, μπορούν μόνο να προβάλλουν τα αρχεία μέσα στον κατάλογο `greeklug`
3. `---` Όλοι οι υπόλοιποι, δεν μπορούν ούτε να προβάλλουν τα περιεχόμενα του καταλόγου (δεν μπορούν να κάνουν τίποτα)

---

7. Αλλαγή δικαιωμάτων αρχείων & καταλόγων (`chmod`)

Αν είμαστε διαχειριστές ή **έχουμε δικαιώματα διαχειριστή** ή είμαστε οι **ιδιοκτήτες των αρχείων ή καταλόγων**, μπορούμε να αλλάξουμε τα δικαιώματα τους, ώστε να δώσουμε πρόσβαση σε άλλους χρήστες του συστήματος.

Αυτό μπορεί να γίνει με 2 τρόπους:

1] Αριθμητικό σύστημα (πιο βολικό)

Δικαίωμα	Αριθμός
`r`	`4`
`w`	`2`
`x`	`1`

`-rw-rw-r-- 1 greeklug greeklug 76 Apr 3 19:51 file35.txt`
Στο παραπάνω αρχείο `file35.txt` τα δικαιώματα του είναι:
`6` (4+2) για τον ιδιοκτήτη `greeklug`
`6` (4+2) για το group `greeklug`
`4` (4) για όλους τους άλλους
Συνεπώς, τα συνολικά δικαιώματα του αρχείου `file35.txt` είναι: `664`

Αν θέλουμε να κάνουμε το αρχείο εκτελέσιμο για τον ιδιοκτήτη και για το group `greeklug`, δηλαδή να αλλάξουμε τα δικαιώματα ώστε να φαίνεται:
`-rwxrwxr-- 1 greeklug greeklug 76 Apr 3 19:51 file35.txt`

τότε θα πρέπει τα δικαιώματα του να είναι:
`7` (4+2+1) για τον ιδιοκτήτη `greeklug`
`7` (4+2+1) για το group `greeklug`
`4` (4) για όλους τους άλλους
Συνεπώς, τα συνολικά νέα δικαιώματα του αρχείου `file35.txt` θα είναι: `774`

Αυτό γίνεται χρησιμοποιώντας την εντολή `chmod`:
`chmod 774 file35.txt`

2] Συμβολικό σύστημα

Σύμβολο	Έννοια
`u`	user (ιδιοκτήτης αρχείου)
`g`	group
`o`	others
`a`	all

Παραδείγματα:

- `chmod u+x script.sh` (Προσθέτουμε δικαιώματα εκτέλεσης του αρχείου script.sh, στον ιδιοκτήτη του)
- `chmod g+rw file.txt` (Προσθέτουμε δικαιώματα ανάγνωσης και εγγραφής του αρχείου file.txt, στο group)
- `chmod o=rw file.txt` (Δίνουμε μόνο δικαιώματα ανάγνωσης και εγγραφής του αρχείου file.txt, στους άλλους)
- `chmod u-x file.txt` (Αφαιρούμε δικαιώματα εκτέλεσης του αρχείου file.txt, από τον ιδιοκτήτη του)
- `chmod a=rwx file.txt` (Δίνουμε πλήρη δικαιώματα του αρχείου file.txt, σε όλους)
- `chmod o=rx dir1` (Δίνουμε μόνο δικαιώματα προβολής του φακέλου dir1 στους άλλους)

---

8. Αλλαγή σε άλλον χρήστη

8α] su

Μπορούμε να αλλάξουμε σε άλλον χρήστη με την εντολή:
`su -l otheruser`
πχ
`su -l member`
όπου το option `-l` σημαίνει login

Για την εκτέλεση της εντολής, το σύστημα μας ζητάει να πληκτρολογήσουμε **τον κωδικό του χρήστη που θέλουμε να συνδεθούμε** (`member`)

Έτσι, **αποσυνδεόμαστε από τον τρέχον χρήστη**, **ανοίγει νέο shell** και συνδεόμαστε ως νέος χρήστης (`member`) με το περιβάλλον του νέου χρήστη (**full login**)

Υπάρχει η δυνατότητα να αλλάξουμε με την εντολή `su member` όπου επίσης θα μας ζητήσει τον κωδικό του νέου χρήστη, ωστόσο **διατηρείται το περιβάλλον** του αρχικού μας χρήστη.

8β] `root` χρήστης

Σε όλες τις διανομές Linux υπάρχει ο... υπέρτατος διαγαλαξιακός... διαχειριστής που ονομάζεται `root` . Έχει πλήρη πρόσβαση σε όλο το σύστημα.

8β1] `sudo`

Σε αρκετές διανομές ο 1ος χρήστης ενός συστήματος έχει δικαιώματα διαχειριστή μέσω του ειδικού group `sudo` (sudoers).

Για να δούμε αν ανήκουμε στο group `sudo` :
`id username`
πχ
`id greeklug`
μας δίνει:
`uid=1000(greeklug) gid=1000(greeklug) groups=1000(greeklug),27(sudo)`
Βλέπουμε ότι ο χρήστης `greeklug` ανήκει στο group `sudo` και έχει user id: 1000

Σε διανομές που χρησιμοποιούν `sudo` , πχ όπως αυτές που στηρίζονται σε Ubuntu, μπορούμε βάζοντας το `sudo` μπροστά από την εντολή, να την εκτελέσουμε πληκτρολογώντας στη συνέχεια τον **κωδικό του user με τον οποίο είμαστε συνδεδεμένοι.**
Παράδειγμα:
`sudo chmod u-x file.txt`
Με την παραπάνω εντολή, **δεν αλλάζουμε χρήστη**, **ούτε shell**, απλά εκτελούμε την εντολή με τα δικαιώματα διαχειριστή που έχουμε.

Μια διαφορά του Linux με άλλα λειτουργικά συστήματα είναι ότι το σύστημα απαιτεί κάθε φορά από εμάς την εισαγωγή του κωδικού μας, για την εκτέλεση της εντολής (συνήθως δεν θα ζητηθεί για το επόμενο λεπτό, από τη στιγμή που θα εισαχθεί). Αυτό προσθέτει ένα επιπλέον επίπεδο ασφάλειας.

Γενικά χρησιμοποιούμε το `sudo` κάθε φορά που καλούμαστε να κάνουμε εργασίες σχετικά με την διαχείριση του συστήματος, πχ εγκατάσταση ενημερώσεων, εγκατάσταση/αφαίρεση προγραμμάτων, διαχείριση χρηστών, κτλ
Φυσικά, ένας χρήστης με δικαιώματα διαχειριστή έχει πρόσβαση και στα αρχεία όλων των υπόλοιπων χρηστών, διαχειριστών ή μη, αν δεν έχει γίνει κάποια επιπλέον ρύθμιση.

8β2] su `root` & `su -l root`

Σε διανομές που δεν χρησιμοποιούν `sudo` , πχ όπως στο Debian, μπορούμε να αλλάξουμε σε άλλον χρήστη με την εντολή:
`su root` ή την `su -l root`

Για την εκτέλεση της εντολής, το σύστημα μας ζητάει να πληκτρολογήσουμε **τον κωδικό του `root`**
Έτσι, **αποσυνδεόμαστε από χρήστης** `greeklug`, **ανοίγει νέο shell** και συνδεόμαστε ως χρήστης `root`

---

9. Αλλαγή ιδιοκτησίας αρχείων (`chown`)

Αν είμαστε διαχειριστές ή **έχουμε δικαιώματα διαχειριστή**, μπορούμε να αλλάξουμε την ιδιοκτησία ενός αρχείου ή καταλόγου.

Χρησιμοποιούμε την εντολή `chown` (ownership)

Αλλαγή μόνο ιδιοκτήτη:
Έστω ότι έχουμε τον παρακάτω κατάλογο:
`drwxrwxr-x 2 member member 4096 Apr 8 20:17 dir1`
και θέλουμε να αλλάξουμε την ιδιοκτησία του καταλόγου dir1 από τον τρέχον χρήστη στον χρήστη `greeklug` πληκτρολογούμε:
`sudo chown greeklug dir1`
Ο φάκελος θα αλλάξει ως εξής:
`drwxrwxr-x 2 greeklug member 4096 Apr 8 20:17 dir1`

`chown -R greeklug dir1/`
Αλλάζουμε την ιδιοκτησία του `dir1` και όλων των αρχείων/υποφακέλων του, σε χρήστη `greeklug` (το `-R` ή `--recursive` εφαρμόζει την αλλαγή αναδρομικά)

Αλλαγή ιδιοκτήτη και group:
Με την εντολή:
`sudo chown greeklug:greeklug dir1`
αλλάζουμε την ιδιοκτησία του καταλόγου σε χρήστη `greeklug` και το group σε `greeklug`

---

10. Διαχειριστές πακέτων: Δύο επίπεδα

Υπάρχουν 2 επίπεδα διαχείρισης λογισμικού: ένα εργαλείο χαμηλού επιπέδου (όπως το **dpkg** ή το **rpm**), που φροντίζει για τις λεπτομέρειες της αποσυμπίεσης των πακέτων, της εκτέλεσης σεναρίων και της σωστής εγκατάστασης του λογισμικού, ενώ ένα εργαλείο υψηλού επιπέδου (όπως το **apt**, το **dnf** ή το **zypper**) εργάζεται με **ομάδες πακέτων**, κατεβάζει πακέτα από τον πάροχο και **επιλύει τις εξαρτήσεις**.

Τις περισσότερες φορές, οι χρήστες χρειάζεται να χρησιμοποιούν **μόνο το εργαλείο υψηλού επιπέδου**, το οποίο φροντίζει να καλεί το εργαλείο χαμηλού επιπέδου όταν χρειάζεται. Η επίλυση εξαρτήσεων είναι ιδιαίτερα σημαντική δυνατότητα του εργαλείου υψηλού επιπέδου, καθώς αναλαμβάνει να βρει και να εγκαταστήσει αυτόματα κάθε απαραίτητο εξαρτώμενο πακέτο.

Linux Package Manager	Debian	SUSE	Red Hat
High Level Package Manager	apt	zypper	dnf
Low Level Package Manager	dpkg	rpm	rpm

2 επίπεδα διαχείρισης λογισμικού

Λειτουργία	Debian based	Red Hat based	Open Suse based	Arch based
Αναζήτηση πακέτου	`apt search nano`	`dnf search nano`	`zypper search nano`	`pacman -Ss nano`
Εγκατάσταση πακέτου (με εξαρτήσεις)	`apt install geany`	`dnf install nano`	`zypper install nano`	`pacman -S nano`
Αφαίρεση πακέτου (με εξαρτήσεις)	`apt remove nano`	`dnf remove nano`	`zypper remove nano`	`pacman -R nano`
Λήψη ενημερώσεων όλου του συστήματος	`apt update`	`dnf check-update`	`zypper refresh`	`pacman -Sy`
Αναβάθμιση όλου του συστήματος	`apt dist-upgrade`	`dnf upgrade`	`zypper update`	`pacman -Syu`

Εντολές διαχείρισης λογισμικού στις πιο γνωστές οικογένειες διανομών. Ως παράδειγμα χρησιμοποιείται το πρόγραμμα `nano`. Οι παραπάνω εντολές απαιτούν δικαιώματα διαχειριστή, επομένως χρειάζεται το `sudo` πριν την καθεμιά.

APT (Advanced Package Tool)

Ενημερώσεις

- `sudo -i`
- `apt update` (προβάλλονται οι νέες ενημερώσεις)
- `apt list --upgradable` (βλέπουμε αναλυτικά τι θα εγκατασταθεί στο σύστημά μας)
- `apt dist-upgrade` (γίνεται λήψη και εγκατάσταση των νέων ενημερώσεων)

Σε άλλες διανομές η εντολή είναι μία αντί για 2, πχ (dnf update)

Εγκατάσταση ενός πακέτου (παράδειγμα με git)

- `apt list git`
- `apt search git`
- `apt install git`
- `apt remove git`

Πώς προσθέτουμε ένα ppa - Personal Package Archive (παράδειγμα με LibreOffice)

- `sudo add-apt-repository ppa:libreoffice/ppa`
- `sudo apt update`
- `sudo apt dist-upgrade`

`ls -la /etc/apt/sources.list.d` (για να δούμε από ποιες πηγές κάνουμε λήψη τα πακέτα)
