Σκοπός

Η δεύτερη εργασία αποτελεί συνέχεια της πρώτης, μεταφέροντας τις βασικές αρχές συγχρονισμού αρχείων από τοπικά σε απομακρυσμένα συστήματα. Οι έννοιες του fss_manager, fss_console και worker διατηρούνται, επεκτείνονται όμως με sockets, threads, και condition variables. Η νέα υλοποίηση επιτρέπει τον απομακρυσμένο συγχρονισμό καταλόγων μέσω ενός ενδιάμεσου nfs_manager και πελατών nfs_client.

Συγκεκριμένα θα υλοποιήσετε ένα **Δικτυακό Σύστημα Συγχρονισμού Αρχείων (Network File System - NFS).** Το σύστημα αυτό θα διαχειρίζεται αυτόματα το συγχρονισμό αρχείων μεταξύ καταλόγων πηγής και καταλόγων προορισμού, θα δημιουργεί δηλαδή αντίγραφα αρχείων σε πραγματικό χρόνο. Η διαχείριση των αρχείων θα γίνεται με worker threads ενώ για το συγχρονισμό θα χρησιμοποιήσετε condition variables.

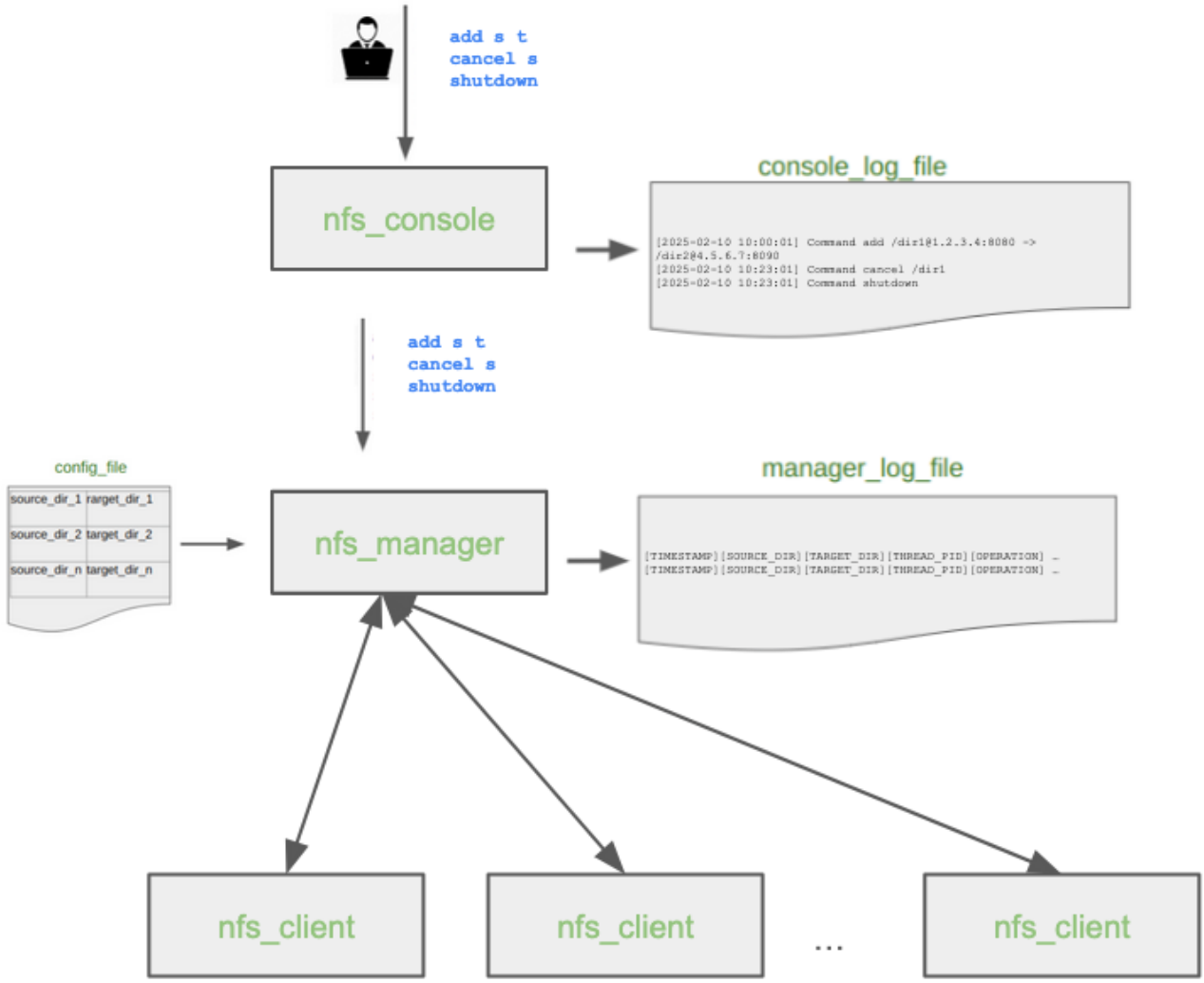
Η υλοποίησή σας πρέπει να περιλαμβάνει:

1. **nfs_manager**: Μια εφαρμογή διαχειριστή υπεύθυνο για την οργάνωση της διαδικασίας συγχρονισμού, Είναι ο αντίστοιχος διαχειριστής της προηγούμενης εργασίας (fss_manager), προσαρμοσμένος για χρήση σε δικτυακό περιβάλλον.»

2. **nfs_client**: μια εφαρμογή που αναλαμβάνει τη μεταφορά των αρχείων

3. **nfs_console**: Μία διεπαφή χρήστη για τη διαχείριση και την υποβολή ερωτημάτων στο σύστημα

Διάγραμμα Γενικής Επισκόπησης Συστήματος NFS



Εφαρμογή nfs_manager και worker processes

Εκτέλεση nfs_manager

Η εφαρμογή nfs_manager θα χρησιμοποιείται ως εξής:

```
./nfs_manager -l <manager_logfile> -c <config_file> -n <worker_limit>
              -p <port_number> -b <bufferSize>
```

όπου:

- Η παράμετρος `logfile` είναι το αρχείο καταγραφής του συστήματος.
- Η παράμετρος `config_file` είναι ένα αρχείο που περιέχει ζεύγη καταλόγων. Συγκεκριμένα, κάθε γραμμή περιέχει ένα ζεύγος, (`source_dir`, `target_dir`). Το `source_dir` υποδεικνύει τον κατάλογο του οποίου τα περιεχόμενα θα αντιγραφούν στον κατάλογο

`target_dir`. Το αρχείο config_file περιέχει ζεύγη source και target καταλόγων, διαχωρισμένα με ένα κενό. Υποθέτουμε πως τα source/target dirs δεν είναι αλληλοκαλυπτόμενα δηλαδή δεν θα υπάρχουν διαφορετικές γραμμές που να είναι sub-directories άλλων γραμμών.

- Η παράμετρος worker_limit είναι ο μέγιστος αριθμός νημάτων εργαζομένων (worker threads) που μπορούν να εκτελούνται ταυτόχρονα (default τιμή 5).

- Η παράμετρος `port_number` είναι το port στο οποίο θα ακούει ο nfs_manager για εντολές από το nfs_console.

- Η παράμετρος `bufferSize` είναι το μέγεθος (number of slots) ενός ενταμιευτή που θα κρατάει εργασίες συγχρονισμού που θα εκτελούνται από worker threads. Πρέπει να είναι > 0.

config_file
Το config file όπως είπαμε περιέχει ζεύγη από source/target directories. Το format κάθε γραμμής περιλαμβάνει τον host και το port με το οποίο θα υπάρχει η επικοινωνία καθώς επίσης και το αντίστοιχο directory. Για παράδειγμα:

```
/source1@123.10.10.20:8000 /source2@100.200.10.10:8080
...
```

Στο παραπάνω παράδειγμα η δουλειά του nfs_manager θα είναι να συγχρονίσει το directory `/source1` από τον host `123.10.10.20` που ακούει στο port `8000` με το directory `/source2` στον host `100.200.10.10` που ακούει στο port `8080`.

Προσοχή: Για λόγους ασφάλειας τα paths είναι relative από το directory που τρέχει ο κάθε nfs_client. Αν για παράδειγμα ένας nsf_client ξεκινάει μέσα στο directory /home/user/mydir τότε το /source1 πιο πάνω σημαίνει /home/user/mydir/source1 και όχι /source1 δηλαδή όχι από το root directory.

Υποδοχή νέων αιτημάτων: Ο nfs_manager υποδέχεται νέα αιτήματα για συγχρονισμό καταλόγων μέσω του nfs_console και μέσω του port που έχει δοθεί ως παράμετρος.

Αποθήκευση πληροφοριών σε εσωτερική δομή: Ο nfs-manager χρησιμοποιεί δομές δεδομένων sync_info_mem_store (δικής σας επιλογής) για την αποδοτική αποθήκευση και αναζήτηση πληροφοριών σχετικά με όλους τους παρακολουθούμενους καταλόγους. Η sync_info_mem_store (π.χ. hash table ή linked list) αποθηκεύει πληροφορίες ανά παρακολουθούμενο κατάλογο πχ (source_host, source_port, source_dir, target_host, target_port, target_dir, status, last_sync_time, active, error_count).

Διαχείριση Ορίου Εργαζομένων: Εάν έχει ήδη επιτευχθεί ο μέγιστος επιτρεπτός αριθμός worker threads εργαζομένων, ο διαχειριστής τοποθετεί επιπλέον εργασίες συγχρονισμού σε ουρά και τις εκτελεί όταν ελευθερωθούν εργαζόμενοι.

Λειτουργικότητα κατά την εκκίνηση

Ο nfs_manager στην αρχή δημιουργεί ένα socket στο port που δόθηκε ως παράμετρος στο command line στο οποίο θα δέχεται μηνύματα επικοινωνίας από το nfs_console. Ακολούθως, ετοιμάζει τις διάφορες δομές δεδομένων που θα χρειαστεί για το συγχρονισμό των καταλόγων. Επίσης ετοιμάζει και δημιουργεί ένα worker thread pool. Αυτά τα worker threads θα αναλάβουν τη δουλειά του συγχρονισμού ένα αρχείο τη φορά.

Προετοιμάζει επίσης τον συγχρονισμό καταλόγων που καθορίζονται στο `config_file`.
Η διαδικασία του συγχρονισμού είναι η εξής, για κάθε ζεύγος
`(source_dir@source_host:source_port, target_dir@target_host:target_port)`:

Αρχικά συνδέεται στο source_host:source_port όπου τρέχει ένας nfs_client και στέλνει μια εντολή:

```
LIST source_dir
```

Ο nfs_client διαβάζει αυτή την εντολή και επιστρέφει στον nfs_manager μία λίστα από τα αρχεία (ένα σε κάθε γραμμή και θεωρείστε πως δεν έχουμε υποκαταλόγους) που υπάρχουν μέσα στο directory source_dir. Το τέλος της λίστας σηματοδοτείται με μία τελεία, δηλαδή το χαρακτήρα '.'. Για παράδειγμα:

```
file1.txt
file2.txt
file3.txt
.
```

Εφόσον πάρει τη λίστα αποθηκεύει όλα αυτά τα αρχεία σε ένα buffer μαζί με την πληροφορία για το από πού προέρχονται (δηλαδή το source_host:source_port) και πού πηγαίνουν (δηλαδή το target_host, target_port, target_dir).

```
Τυπώνει στην οθόνη, στέλνει στο nfs_console, και γράφει στο manager-log-file
[2025-02-10 10:00:01] Added file: /dir1/file1@1.2.3.4:8080 ->
/dir2/file1@4.5.6.7:8090
```

Εφόσον υπάρχουν αρχεία στο buffer, κάθε worker thread δουλεύει ως εξής:

Παίρνει ένα entry από το buffer και ανοίγει socket connections στα source και target hosts. Κατόπιν, στέλνει την εντολή:

```
PULL /source_dir/file.txt
```

στον source nfs_client. Σε αυτό το σημείο ο nfs_client επιστρέφει πίσω στο thread το περιεχόμενο του file1.txt. Προσοχή το thread διαβάζει σε κάποιο buffer (δικό του όχι global) που δεν είναι απαραίτητο πως χωράει όλο το αρχείο οπότε θα πρέπει να κάνετε πολλαπλά reads.

Ταυτόχρονα το worker thread για κάθε chunk που διάβασε, στέλνει στο target_host:target_port nfs_client την εντολή:

```
PUSH /target_dir/file.txt chunk_size data
```

όπου chunk_size είναι το μέγεθος των data που περιμένει να **λάβει σε αυτό το request** (όχι το μέγεθος του αρχείου).

Όταν το chunk_size είναι -1 τότε σημαίνει πως το αρχείο θα ανοιχτεί για γράψιμο από την αρχή. Όταν το chunk_size είναι 0 τότε σημαίνει πως δεν υπάρχουν άλλα data και άρα έχουμε EOF οπότε το αρχείο μπορεί να κλείσει από τον target nfs_client.

Για λόγους απλότητας θεωρήστε ότι οι κατάλογοι ειναι flat και περιέχουν μόνο αρχεία και όχι υποκαταλόγους. Επίσης προσοχή τα threads θα πρέπει να έχουν δικά τους I/O buffers και θα πρέπει να κάνετε συγχρονισμό στην ουρά που έχει όλα τα requests των αρχείων με condition variables.

Λήψη εντολών από nfs_console

Αφού ξεκινήσει ο nfs_manager και έχοντας ξεκινήσει το συγχρονισμό καταλόγων που αναφέρονται στο `config_file`, ο nfs_server μπορεί να δεχτεί ασύχρονα εντολές στο port που του δόθηκε ως παράμετρος.

Για κάθε εντολή που δέχεται ο nfs_manager την επεξεργάζεται και την έξοδο:

1. τη στέλνει πίσω στο nfs_console μέσω του socket, και
2. την αποθηκεύει στο αρχείο καταγραφής (manager-log-file). Το αρχείο αυτό είναι διαφορετικό από το console-log-file της κονσόλας που θα αναφερθεί παρακάτω.

Εντολές που μπορεί να δεχθεί ο nfs_manager

`add <source> <target>`

Καταχωρεί όλα τα αρχεία του καταλόγου `<source>` για συγχρονισμό στον κατάλογο `<target>`.

```
Τυπώνει στην οθόνη, στέλνει στο nfs_console, και γράφει στο manager-log-file
[2025-02-10 10:00:01] Added file: /dir1/file1@1.2.3.4:8080 ->
/dir2/file1@4.5.6.7:8090
```

[Σημείωση: αν είναι ήδη καταχωρημένο το ζευγος `<source, target>` για συγχρονισμό, δεν ξεκινάει συγχρονισμό.]

```
Τυπώνει στην οθόνη και στέλνει στο nfs_console μόνο
[2025-02-10 10:00:01] Already in queue: /dir1/file1@1.2.3.4:8080
```

`cancel <source dir>`

Ακυρώνει τον υπόλοιπο συγχρονισμό του καταλόγου `<source dir>`. Αν δηλαδή έχουν γίνει ήδη κάποια αρχεία του καταλόγου synchronized ακυρώνει τα υπόλοιπα που είναι στην ουρά.

```
Τυπώνει στην οθόνη, στέλνει στο nfs_console, και γράφει στο manager-log-file
[2025-02-10 10:23:01] Synchronization stopped for /dir1@1.2.3.4:8080
```

Αν ο κατάλογος δεν παρακολουθείται τότε:

```
Τυπώνει στην οθόνη και στέλνει στο nfs_console μόνο
[2025-02-10 10:00:01]  Directory not being synchronized:
/dir1/file1@1.2.3.4:8080
```

shutdown

Σταματά τον nfs_manager, περιμένει τα ενεργά worker threads να τερματίσουν, επεξεργάζεται όσες εργασίες συγχρονισμού έχουν μείνει στην ουρά αναμονής και τερματίζει ομαλά:

```
Τυπώνει στην οθόνη και στέλνει στο nfs_console
[2025-02-10 10:23:02] Shutting down manager...
[2025-02-10 10:23:02] Waiting for all active workers to finish.
[2025-02-10 10:23:03] Processing remaining queued tasks.
[2025-02-10 10:24:01] Manager shutdown complete.
```

Worker threads:

Κάθε worker thread εκτελεί μια μεμονωμένη εργασία συγχρονισμού για ένα αρχείο.

Υλοποιηση εργασίας συγχρονισμού

Στην υλοποίησή σας, καθε worker πρέπει να χρησιμοποιεί χαμηλού επιπέδου I/O syscalls (π.χ., open, read, write, close) για την εργασία συγχρονισμού που επεξεργάζεται. (Σημείωση: **Απαγορεύεται** η χρήση κλήσεων όπως (`system("scp -r source dest"` ή `exec("rsync",...))`. Το worker thread πρέπει να ελέγχει την επιτυχία ή αποτυχία κάθε λειτουργίας που εκτελεί κατά τη διάρκεια του συγχρονισμού αρχείων. Αυτό γίνεται με δύο βασικούς τρόπους: 1) Έλεγχος των συναρτήσεων συστήματος που χρησιμοποιούνται για την επικονωνία και 2) Χρήση strerror() για περιγραφή σφάλματος.

Αν οποιαδήποτε από τις συναρτήσεις συστήματος επιστρέψει -1, τότε η strerror(errno) τυπώνει τον ακριβή λόγο της αποτυχίας.

Καταγραφή σε manager-log-file εργασίας συγχρονισμού

Ο nfs_manager γράφει στο manager-log-file μια εγγραφή που περιγράφει την εργασία συγχρονισμού που ολοκληρώθηκε από έναν worker. Το format της εγγραφής θα είναι ως εξής:

```
[TIMESTAMP] [SOURCE_DIR] [TARGET_DIR] [THREAD_PID] [OPERATION] [RESULT] [DETAILS]
```

όπου:

- `TIMESTAMP` είναι η ημερομηνία και ώρα του γεγονότος
- `SOURCE_DIR` είναι ο κατάλογος προέλευσης (`source`).
- `TARGET_DIR` είναι ο κατάλογος προορισμού (`target`)
- `THREAD_PID` είναι το pid του worker thread.
- `OPERATION` είναι ο τύπος συγχρονισμού: `PUSH` ή `PULL`.
- `RESULT` είναι το αποτέλεσμα της εργασίας συγχρονισμού: `SUCCESS` ή `ERROR`.
- `DETAILS` είναι λεπτομέρειες ή περιγραφή του αποτελέσματος.

Αρχικός Πλήρης Συγχρονισμός (Full Sync) από το `config_file`:

```
[2025-02-10 10:00:01] [/dir1/file1@1.2.3.4:8080] [/dir2/file1@4.5.6.7:8090]
[1234] [PULL] [SUCCESS] [10 bytes pulled]

[2025-02-10 10:00:02] [/dir1/file1@1.2.3.4:8080] [/dir2/file1@4.5.6.7:8090]
[1234] [PUSH] [SUCCESS] [10 bytes pushed]
```

Παράδειγμα Αποτυχίας σε Συγχρονισμό (Error Case):

```
[2025-02-10 10:00:01] [/dir1/file1@1.2.3.4:8080] [/dir2/file1@4.5.6.7:8090]
[1234] [PULL] [ERROR] [File: file1.txt - Permission Denied]
```

Εφαρμογή nfs_console

Οι χρήστες αλληλεπιδρούν με το σύστημα μέσω του **nfs_console**, το οποίο επικοινωνεί με τον nfs_manager.

Η εφαρμογή nfs_console παίρνει τα εξής ορίσματα

```
./nfs_console -l <console-logfile> -h <host_IP> -p <host_port>
```

όπου:

- Η παράμετρος `console-logfile` είναι το αρχείο καταγραφής της κονσόλας.
- Η παράμετρος `host_IP` είναι το όνομα μηχανήματος που τρέχει ο nfs_manager.
- Η παράμετρος `host_port` είναι το port στο οποίο θα ακούει ο nfs_manager για εντολές από το nfs_console

Μετά την εκκίνηση της nfs_console οι χρήστες μπορούν να εκτελούν τις παρακάτω εντολές (case sensitive):

- `add <source> <target>`: Προσθέτει ένα νέο ζεύγος καταλόγων προς συγχρονισμό.
- `cancel <source>`: Ακυρώνει τον συγχρονισμό του καταλόγου source.
- `shutdown`: Σταματά τη λειτουργία του διαχειριστή και των εργαζομένων.

Κατ΄ αντιστοιχία με τον nfs_manager το nfs_console χρησιμοποιεί ένα δικό του console-log-file διαφορετικό από αυτό του nfs-manager για να καταγράψει με απλό ομως τρόπο τις εντολές που εισάγονται από τον χρήστη πχ

```
[2025-02-10 10:00:01] Command add /dir1@1.2.3.4:8080 -> /dir2@4.5.6.7:8090
[2025-02-10 10:23:01] Command cancel /dir1
[2025-02-10 10:23:01] Command shutdown
```

Το nfs_console στέλνει τις εντολές στον nfs_manager μέσω ενός socket. Ο nfs_manager απαντά με τα αντίστοιχα μηνύματα κατάστασης, τα αποτελέσματα συγχρονισμού ή τυχόν σφάλματα πάνω από το socket από το οποίο το nfs_console τα διαβάζει και στη συνέχεια το nfs_console (προσοχή όχι ο nfs_manager)

- εμφανίζει στον χρήστη τα αποτελέσματα ή τυχόν σφάλματα και
- τα καταγράφει στο console-log-file του.

Το nfs_console τερματίζεται όταν ο χρήστης την κλείσει ή όταν εκτελεστεί η εντολή shutdown.

Παράδειγμα χρήσης:

```
$ ./nss_console -l console-logfile -h 12.28.28.9 -p 8035
> add /dir11.2.3.4:8080 -> /dir2@4.5.6.7:8090

[2025-02-10 10:00:01] Added file1: /dir1/file1@1.2.3.4:8080 ->
/dir2/file1@4.5.6.7:8090

> cancel /dir1

[2025-02-10 10:23:01] Synchronization stopped for /dir1@1.2.3.4:8080




> shutdown
[2025-02-10 10:23:02] Shutting down manager...
[2025-02-10 10:23:02] Waiting for all active workers to finish.
[2025-02-10 10:23:03] Processing remaining queued tasks.
[2025-02-10 10:24:01] Manager shutdown complete.
```

Εφαρμογή nfs_client

Η εφαρμογή nfs_client θα χρησιμοποιείται ως εξής:

```
./nfs_client -p <port_number>
```

όπου:

- Η παράμετρος `port_number` είναι το port στο οποίο θα ακούει ο nfs_client για εντολές

Ο nfs_client ξεκινάει και ακού στο συγκεκριμένο port. Ο nfs_client ακούει μόνο 3 είδη εντολών:

- `LIST source_dir` στέλνει πίσω στο host που επικοινώνησε μία λίστα (ένα σε κάθε γραμμή) με τα αρχεία που υπάρχουν στο συγκεκριμένο directory. Προσοχή το source_dir είναι local από εκεί που τρέχει ο nfs_client (δείτε και παρατήρηση στο κομμάτι του nfs_manager).
- `PULL /source_dir/file.txt` στέλνει πίσω στο host το συγκεκριμένο αρχείο εφόσον υπάρχει. Το στέλνει με το εξής φορμάτ:

  `<filesize><space><data…>`

  όπου filesize είναι το μέγεθος του αρχείου σε bytes, space είναι ένας κενός χαρακτήρας και data είναι τα δεδομένα. Εφόσον το αρχείο δεν υπάρχει ή έχει γίνει κάποιο λάθος τότε το filesize είναι -1. Σε αυτή την περίπτωση τα data θα περιέχουν το string λάθους.
- `PUSH /target_dir/file.txt chunk_size data` λαμβάνει από το socket data για το συγκεκριμένο αρχείο. Αν το chunk_size είναι -1 τότε το αρχείο θα πρέπει να γραφτεί από την αρχή εφόσον ήδη υπάρχει (δηλαδή να γίνει truncate). Σε αυτή την περίπτωση ο nfs_client κάνει append το κάθε chunk από data στο αρχείο. Αν το chunk size είναι 0 αυτό σημαίνει πως έχει γραφτεί όλο το αρχείο και επομένως μπορεί να το κλείσει.

Helpful tips

- Μπορείτε να υποθέσετε πως οι κατάλογοι θα είναι όλοι flat και θα περιέχουν μόνο αρχεία, όχι subdirectories.

- Φροντίστε να καθαρίζετε τα logfiles κατά την εκκίνηση των προγραμμάτων σας απο πιθανές προηγούμενες εκτελέσεις ώστε να μη δημιουργούνται προβλήματα λειτουργίας.

- Μπορείτε να θεωρήσετε ότι σε περίπτωση που υπάρχει μια εκδοση ενός αρχείου στο target dir μπορείτε απλώς να το κάνετε overwrite χωρίς να ελέγξετε χρονοσημάνσεις.

- Όλες οι εγγραφές ημερομηνίας/ώρας πρέπει να παραχθούν με μορφή

  ("%Y-%m-%d %H:%M:%S").

Διαδικαστικά

- Το πρόγραμμά σας θα πρέπει να γραφεί σε C (ή C++) και σας θα πρέπει να τρέχει στα Linux workstations του Τμήματος. Κώδικας που δε μεταγλωττίζεται εκεί, θεωρείται ότι δεν μεταγλωττίζεται. Δε θα γίνει αποδεκτή η εξέταση της εργασίας σε άλλον υπολογιστή.
- Για επιπρόσθετες ανακοινώσεις, παρακολουθείτε το forum του μαθήματος στο piazza.com.

Η πλήρης διεύθυνση είναι https://piazza.com/uoa.gr/spring2025/k24/home. Η παρακολούθηση του φόρουμ στο Piazza είναι υποχρεωτική.

- Ο κώδικάς σας θα πρέπει να αποτελείται από τουλάχιστον δύο (και κατά προτίμηση περισσότερα) διαφορετικά αρχεία. Η χρήση του separate compilation είναι επιτακτική και ο κώδικάς σας θα πρέπει να έχει ένα Makefile.

● Βεβαιωθείτε πως ακολουθείτε καλές πρακτικές software engineering κατά την υλοποίηση της άσκησης. Η οργάνωση, η αναγνωσιμότητα και η ύπαρξη σχολίων στον κώδικα αποτελούν κομμάτι της βαθμολογίας σας.
● Ο κώδικάς σας θα πρέπει να κάνει compile στα εκτελέσιμα nfs_manager, nfs_console, και nfs_client όπως **ακριβώς** ορίζει η άσκηση τα οποία θα λειτουργούν με τις παραμέτρους ακριβώς όπως προδιαγράφονται. **(penalty -10%).**

Τι πρέπει να παραδοθεί

Όλη η δουλειά σας θα παραδοθεί στο github repository που θα πρεπει να δημιουργήσετε ακολουθώντας τις οδηγίες στο κείμενο

https://docs.google.com/document/d/1iD7P3Zyfpg-stMUr2FpN98ciLM1K2noucHb-ojYygLQ/edit?usp=sharing

Να υποβάλετε μόνο κώδικα, Makefile, README και **όχι τα binaries**. Η άσκησή σας θα γίνει compile από την αρχή πριν βαθμολογηθεί. Επίσης μην υποβάλετε μεσα στο repo μεγάλα αρχεια για τεστ.
● Όποιες σχεδιαστικές επιλογές κάνετε, θα πρέπει να τις περιγράψετε σε ένα README (απλό text αρχείο) που θα υποβάλλετε μαζί με τον κώδικά σας. Το README χρειάζεται να περιέχει μια σύντομη και περιεκτική εξήγηση για τις επιλογές που έχετε κάνει στον σχεδιασμό του προγράμματός σας σε 1-2 σελίδες ASCII κειμένου. Συμπεριλάβετε την εξήγηση και τις οδηγίες για το compilation και την εκτέλεση του προγράμματός σας.

● Ο κώδικας που θα υποβάλετε θα πρέπει να είναι δικός σας. Απαγορεύεται η χρήση κώδικα που δεν έχει γραφεί από εσάς ή κώδικας που έχει γραφτεί με τη βοήθεια μηχανών τύπου chatGPT. Θα χρησιμοποιηθούν AI code detector για την αναγνώριση τέτοιου είδους προσπαθειών.

● Θα παρθεί snapshot των εργασιών από το github την ημέρα και ώρα λήξης της προθεσμίας. Κανένα commit μετα από αυτό δε θα γίνει δεκτό.

Υποχρεούστε να διατηρήσετε τον κώδικά σας ιδιωτικό στο github.

Τι θα βαθμολογηθεί
● Η συμμόρφωση του κώδικά σας με τις προδιαγραφές της άσκησης.
● Η οργάνωση και η αναγνωσιμότητα (μαζί με την ύπαρξη σχολίων) του κώδικα.
● Η χρήση Makefile και το separate compilation.

Άλλες σημαντικές παρατηρήσεις

- Οι εργασίες είναι ατομικές.
- Όποιος υποβάλλει / δείχνει κώδικα που δεν έχει γραφτεί από την ίδια/τον ίδιο **μηδενίζεται** στο μάθημα.

- Ακολουθήστε αυστήρα το format του log file και του config file που διαβάζετε **(penalty -20%).**
- Ακολουθήστε αυστηρά οτι ζητείται να εκτυπώνεται στην έξοδο. Τίποτα περισσότερο τίποτα λιγότερο. Οχι debug μηνύματα **(penalty -20%).**
- Αν και αναμένεται να συζητήσετε με φίλους και συνεργάτες το πώς θα επιχειρήσετε να δώσετε λύση στο πρόβλημα, αντιγραφή κώδικα (οποιασδήποτε μορφής) είναι κάτι που δεν επιτρέπεται. **Οποιοσδήποτε** βρεθεί αναμεμειγμένος σε αντιγραφή κώδικα απλά παίρνει μηδέν στο μάθημα. Αυτό ισχύει για όσους εμπλέκονται ανεξάρτητα από το ποιος έδωσε/πήρε κλπ. Τονίζουμε πως θα πρέπει να λάβετε τα κατάλληλα μέτρα ώστε να είναι προστατευμένος ο κώδικάς σας και να μην αποθηκεύεται κάπου που να έχει πρόσβαση άλλος χρήστης (π.χ., η δικαιολογία «Το είχα βάλει σε ένα github repo και μάλλον μου το πήρε από εκεί», δεν είναι δεκτή.)
- Το πρόγραμμά σας θα πρέπει να γραφτεί σε C ή C++. Μπορείτε να χρησιμοποιήσετε μόνο εντολές οι οποίες είναι διαθέσιμες στα μηχανήματα Linux του τμήματος. Πρόγραμμα που πιθανόν μεταγλωτίζεται ή εκτελείται στο προσωπικό σας υπολογιστή αλλά όχι στα μηχανήματα Linux του τμήματος, θεωρείται ότι δε μεταγλωτίζεται ή εκτελείται αντίστοιχα.