Κ24: Προγραμματισμός Συστήματος

Ονοματεπώνυμο: Κωνσταντίνος Πασχόπουλος

Αριθμός Μητρώου: 1115201500127

Όταν ένας καινούργιος client συνδέεται στο σύστημα πρώτα συγχρονίζεται με όλα τα ids που βρίσκονται μέσα στο common_dir. Όταν αυτός ο αρχικός συγχρονισμός ολοκληρωθεί αρχίζει να παρακολουθεί το common_dir για αλλαγές. Προκειμένου ο client να ελέγχει το common_dir για αλλαγές στα αρχεία χρησιμοποίησα το API της inotify. Ο κώδικας που χρησιμοποίησα για την inotify βασίζεται στις διαλέξεις του κύριου Δελή από το προηγούμενο εξάμηνο και στα παραδείγματα που υπάρχουν στο βιβλίο του Kerrisk.

Για να μην συμβαίνει άσκοπο waiting όταν συνδέεται ένας καινούργιος client οι clients κάνουν fork ένα καινούργιο παιδί (το οποίο τρέχει το πρόγραμμα syncing.c) το οποίο είναι υπεύθυνο για τον συγχρονισμό. Δηλαδή ολόκληρο το βήμα 4 συμβαίνει σε ένα καινούργιο παιδί κάθε φορά. Ο λόγος που το έκανα αυτό είναι έτσι ώστε να μην χρειάζεται ο client να περιμένει κάθε φορά ολοκληρωθεί ένας συγχρονισμός για να πάει στον επόμενο, μπορεί να τους κάνει όλους ταυτόχρονα. Όλα τα signals που στέλνουν να παιδία sender και receiver πάνε σε αυτό το καινούργιο process με αυτόν τον τρόπο είναι πιο εύκολο να βρω ποιο παιδί πρέπει να τερματίσω ή να επανεκκινήσω σε περίπτωση λάθους.

Όταν ένας client λάβει SIGINT ή SIGQUIT θα πρέπει να ελευθερώνει όλους του πόρους που χρησιμοποίησε και να σβήνει το mirror_dir του και το .id αρχείο του. Για να το υλοποιήσω αυτό χρησιμοποίησα ένα flag του οποίου η τιμή αλλάζει μόνο μέσα στον signal handler. Έτσι το πρόγραμμα ελέγχοντας την τιμή του flag καταλαβαίνει ότι πρέπει να ξεκινήσει τον τερματισμό του.

Το παιδί sender χρησιμοποιεί μία αναδρομική συνάρτηση για να διασχίσει ολόκληρο το input_dir και να στείλει όλα τα αρχεία και τους φακέλους που υπάρχουν μέσα. Κάθε φορά πριν την αποστολή ενός αρχείου ο sender γράφει 0 ή 1 στο pipe για να δείξει ότι το αρχείο που στέλνει πρόκειται για regular file ή directory αντίστοιχα.

Για να ελέγχω αν ένα παιδί receiver περιμένει πάνω από 30 δευτερόλεπτα να διαβάσει από το pipe χρησιμοποιώ την συνάρτηση alarm και έναν signal handler. Όταν ληφθεί ένα SIGALARM ο signal handler πάει και αυξάνει την μεταβλητή done και στέλνει signal στον πατέρα. Αν η μεταβλητή done δεν είναι 0 η συνάρτηση cleanup πάει και καθαρίζει την μνήμη

που έχω δεσμεύσει για τον buffer, κλείνει τον file descriptor του pipe και διαγράφει το pipe πριν κάνει exit.

Η συνάρτηση που γράφει στο logfile του κάθε client βρίσκεται στο my_functions.c.

Το script create_files.sh χρησιμοποιεί το πρόγραμμα randomString για να παράγει τυχαία αλφαριθμητικά. Το randomString είναι γραμμένο σε C και παίρνει ως ορίσματα τον ελάχιστο και μέγιστο αριθμό bytes που πρέπει να δημιουργήσει και χρησιμοποιώντας την συνάρτηση random() επιστρέφει ένα τυχαίο αλφαριθμητικό. Τα ονόματα των directories και των αρχείων πρώτα αποθηκεύονται σε πίνακες και μετά δημιουργούνται οι απαραίτητες ιεραρχίες με levels και round-robin.

Για να γίνει compilation υπάρχει το Makefile το οποίο κάνει separate compilation.

Σημείωση: Για να τρέξει σωστά το πρόγραμμα μου πρέπει οι φάκελοι common_dir, input_dir και mirror_dir να δίνονται με relative και όχι absolute path.