**Λειτουργικά Συστήματα 2023-2024**

**1η Δραστηριότητα**

Τσάμπρας Κωνσταντίνος

up1083865

* Άσκηση 1η

Στο παρακάτω αρχείο (user.h) βλέπουμε τους ορισμούς των κλήσεων του συστήματος που έχει στην διάθεσή του ο χρήστης:

Εικόνα που περιέχει κείμενο, ηλεκτρονικές συσκευές, στιγμιότυπο οθόνης, λογισμικό

Περιγραφή που δημιουργήθηκε αυτόματα

Ίσως το πιο απλό παράδειγμα χρήσης system calls από τα προγράμματα χρήστη είναι το αρχείο echo.c το οποίο χρησιμοποιεί την printf(). Αυτή με την σειρά της καλεί την write() κλήση συστήματος για να τυπώσει τους χαρακτήρες στην οθόνη.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, οθόνη, υπολογιστής

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

* Άσκηση 2η

Δημιουργούμε ένα αρχείο (fib.c) στον φάκελο user όπου θα έχει την απλή λειτουργία να τυπώνει όσους αριθμούς Fibonacci του ζητήσουμε. Χρειάζεται να υλοποιήσουμε και την συνάρτηση getint() χρησιμοποιώντας την κλήση συστήματος read(), ώστε να διαβάζουμε τον ακέραιο που μας δόθηκε.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Στην συνέχεια υλοποιούμε την main:

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, οθόνη

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

Αποτέλεσμα:

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

* Άσκηση 3η

Με χρήση της εντολής grep στο directory του kernel μπορούμε να βρούμε όλες τις αναφορές του uptime:

Εικόνα που περιέχει κείμενο, γραμματοσειρά, στιγμιότυπο οθόνης, λευκό

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, γραμματοσειρά, αριθμός, στιγμιότυπο οθόνης

Περιγραφή που δημιουργήθηκε αυτόματαΕικόνα που περιέχει κείμενο, γραμματοσειρά, αριθμός, στιγμιότυπο οθόνης

Περιγραφή που δημιουργήθηκε αυτόματα

Παρακάτω έχουμε την υλοποίηση της:

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

Βλέπουμε ότι αρχικά «δεσμεύει» την μεταβλητή tickslock ώστε να αποτρέψει άλλες διεργασίες από το να έχουν ταυτόχρονη πρόσβαση στην ίδια μεταβλητή. Στην συνέχεια αποθηκεύει την τιμή της μεταβλητής, «αποδεσμεύει» την tickslock και επιστρέφει την τιμή. Η τιμή που επιστρέφεται δεν είναι βέβαια σε δευτερόλεπτα αλλά σε πλήθος timing interrupts, στην περίπτωση του xv6 έχουμε 100 timing interrupts κάθε δευτερόλεπτο, άρα αν θέλουμε να βρούμε το uptime σε δευτερόλεπτα, απλά διαιρούμε τον αριθμό που μας επιστρέφει η uptime δια 100.

* Άσκηση 4η

Την συνάρτηση υλοποίησης των system calls εντοπίζουμε στο αρχείο syscall.c.

Αφού οριστούν οι system call functions και το array που αντιστοιχίζει τους δείκτες αυτών των συναρτήσεων σε αριθμούς από 1-21, ορίζεται η συνάρτηση syscall:

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

Αρχικά παίρνει τον αναγνωριστικό αριθμό (num) της συνάρτησης syscall που επιθυμούμε να καλέσουμε τον οποίο (σύμφωνα με την διαδικασία κλήσης συστήματος) έχουμε αποθηκεύσει στον καταχωρητή eax.

Στην συνέχεια ελέγχει αν ο αριθμός βρίσκεται στα αναμενόμενα όρια (από 1 έως το μήκος του array των system calls) και αν έχει δηλωθεί σωστά ο δείκτης στην αντίστοιχη συνάρτηση (π.χ. δεν είναι NULL pointer).

Αν ο έλεγχος επιτύχει, καλείται η system call με αναγνωριστικό αριθμό num. Το αποτέλεσμα της κλήσης αποθηκεύεται πάλι στον καταχωρητή eax σύμφωνα με το πρωτόκολλο.

Αν ο έλεγχος αποτύχει τυπώνεται αντίστοιχο μήνυμα και επιστρέφεται -1.

* Άσκηση 5η

1η αλλαγή: Προσθήκη στο αρχείο syscall.hΕικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

2η αλλαγή: Προσθήκη στο defs.h κάτω από το proc.c  
Εικόνα που περιέχει κείμενο, ηλεκτρονικές συσκευές, στιγμιότυπο οθόνης, λογισμικό

Περιγραφή που δημιουργήθηκε αυτόματα

3η αλλαγή: Προσθήκη στο user.h

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό

Περιγραφή που δημιουργήθηκε αυτόματα

4η αλλαγή: Προσθήκη κλήσης στο sysproc.c

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, οθόνη

Περιγραφή που δημιουργήθηκε αυτόματα

5η αλλαγή: usys.S

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, οθόνη, λογισμικό

Περιγραφή που δημιουργήθηκε αυτόματα

6η αλλαγή: syscall.c

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

7η αλλαγή: Γράφουμε την ίδια την συνάρτηση στο proc.c

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Έπειτα απλά δοκιμάζουμε να την καλέσουμε σε ένα πρόγραμμα.

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, γραμματοσειρά, στιγμιότυπο οθόνης, μαύρο

Περιγραφή που δημιουργήθηκε αυτόματα

* Άσκηση 6η

Σύμφωνα με την παραπάνω διαδικασία δημιουργούμε το system call. Αυτή την φορά η υλοποίηση γίνεται με την εντολή outw(0x604, 0x2000);

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Η κλήση γίνεται μέσω του προγράμματος χρήστη “halting” (halting.c):

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα

Αποτέλεσμα:

Εικόνα που περιέχει κείμενο, ηλεκτρονικές συσκευές, στιγμιότυπο οθόνης, οθόνη

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, γραμμή

Περιγραφή που δημιουργήθηκε αυτόματα

* Άσκηση 7η

Αυτή η άσκηση χρειάστηκε αρκετή προσπάθεια κυρίως γιατί δεν ήμουν σίγουρος αν ο τρόπος (τον οποίο ακολούθησα τελικά) είναι «σωστός». Δεν είμαι σίγουρος καν αν θεωρείται system call. Η υλοποίηση πάντως γίνεται στην syscall():

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, αριθμός

Περιγραφή που δημιουργήθηκε αυτόματα

Όπως φαίνεται παραπάνω, χρησιμοποιούμε μία static array από integers στο επίπεδο της syscall() όπου αποθηκεύουμε τον αριθμό κλήσεων τής κάθε syscall στην θέση που υποδεικνύει ο κωδικός της μέσα στο array. Όταν γίνεται η κλήση syscall με αριθμό 25, δεν πηγαίνουμε στην «κανονική ροή» ενός syscall, αλλά επιστρέφουμε κατευθείαν στον καταχωρητή την τιμή που πρέπει.

Στην «κανονική ροή» που προανέφερα χρειάστηκε προφανώς να προσθέσουμε το calls[num]=calls[num]+1 για να κρατάμε τον αριθμό των κλήσεων της κάθε syscall.

Ο λόγος που χρησιμοποιούμε static για το array είναι προφανώς για να μην γίνονται reset τα δεδομένα κάθε φορά που καλείται η συνάρτηση.

Ως αποτέλεσμα έχουμε το παρακάτω:



Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά

Περιγραφή που δημιουργήθηκε αυτόματα



Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, γραμματοσειρά, σχεδίαση

Περιγραφή που δημιουργήθηκε αυτόματα

* Άσκηση 8η

Με την πλέον γνωστή διαδικασία δημιουργούμε ένα νέο system call και παρακάτω έχουμε την υλοποίηση της συνάρτησης killrandom():

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, οθόνη

Περιγραφή που δημιουργήθηκε αυτόματα

Εικόνα που περιέχει κείμενο, στιγμιότυπο οθόνης, λογισμικό, ιστοσελίδα

Περιγραφή που δημιουργήθηκε αυτόματα

Ορίζουμε μια ψευδοτυχαία συνάρτηση και έπειτα διατρέχουμε την λίστα με τα processes και αποθηκεύουμε τα ids τους σε ένα array.

Στην συνέχεια προσπαθούμε να «σκοτώσουμε» μια τυχαία διεργασία επιλέγοντας ένα τυχαίο pid και στέλνοντας το στην kill(). Αν για κάποιο λόγο δεν δουλέψει την πρώτη φορά (πχ το process διαγράφηκε από το table) διαλέγουμε ένα άλλο τυχαίο pid και επαναλαμβάνουμε.