

ΕΡΓΑΣΤΗΡΙΟ ΛΕΙΤΟΥΡΓΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
ΑΣΚΗΣΗ 2^η



ΠΑΝΕΠΙΣΤΗΜΙΟ
ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΟΝ/ΜΟ : Τριάντης Παναγιώτης
Α.Μ. : 5442
ΕΤΟΣ : 5^ο

Ερώτημα 1

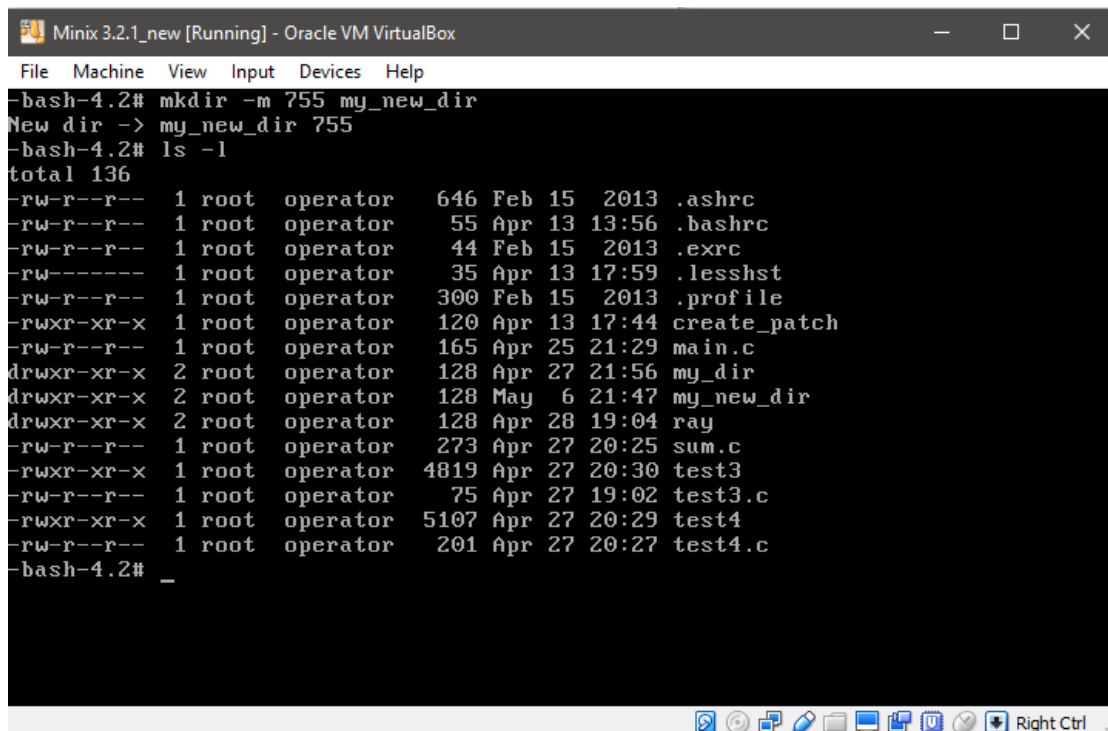
Το system call που πρέπει να τροποποιήσουμε για αυτό το ερώτημα είναι το `do_mkdir()` και βρίσκεται στο αρχείο `open.c`

Path: `usr/src/servers/vfs/open.c`

Στην γραμμή 611 προσθέτουμε την εξής εντολή:

Γραμμή	Εντολή
611	<code>printf("New dir -> %s %o\n",fullpath, bits - 16384);</code>

Screenshot



```
Minix 3.2.1_new [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
-bash-4.2# mkdir -m 755 my_new_dir
New dir -> my_new_dir 755
-bash-4.2# ls -l
total 136
-rw-r--r-- 1 root operator 646 Feb 15 2013 .ashrc
-rw-r--r-- 1 root operator 55 Apr 13 13:56 .bashrc
-rw-r--r-- 1 root operator 44 Feb 15 2013 .exrc
-rw----- 1 root operator 35 Apr 13 17:59 .lessht
-rw-r--r-- 1 root operator 300 Feb 15 2013 .profile
-rwxr-xr-x 1 root operator 120 Apr 13 17:44 create_patch
-rw-r--r-- 1 root operator 165 Apr 25 21:29 main.c
drwxr-xr-x 2 root operator 128 Apr 27 21:56 my_dir
drwxr-xr-x 2 root operator 128 May 6 21:47 my_new_dir
-rwxr-xr-x 2 root operator 128 Apr 28 19:04 ray
-rw-r--r-- 1 root operator 273 Apr 27 20:25 sum.c
-rwxr-xr-x 1 root operator 4819 Apr 27 20:30 test3
-rw-r--r-- 1 root operator 75 Apr 27 19:02 test3.c
-rwxr-xr-x 1 root operator 5107 Apr 27 20:29 test4
-rw-r--r-- 1 root operator 201 Apr 27 20:27 test4.c
-bash-4.2# _
```

Ερώτημα 2

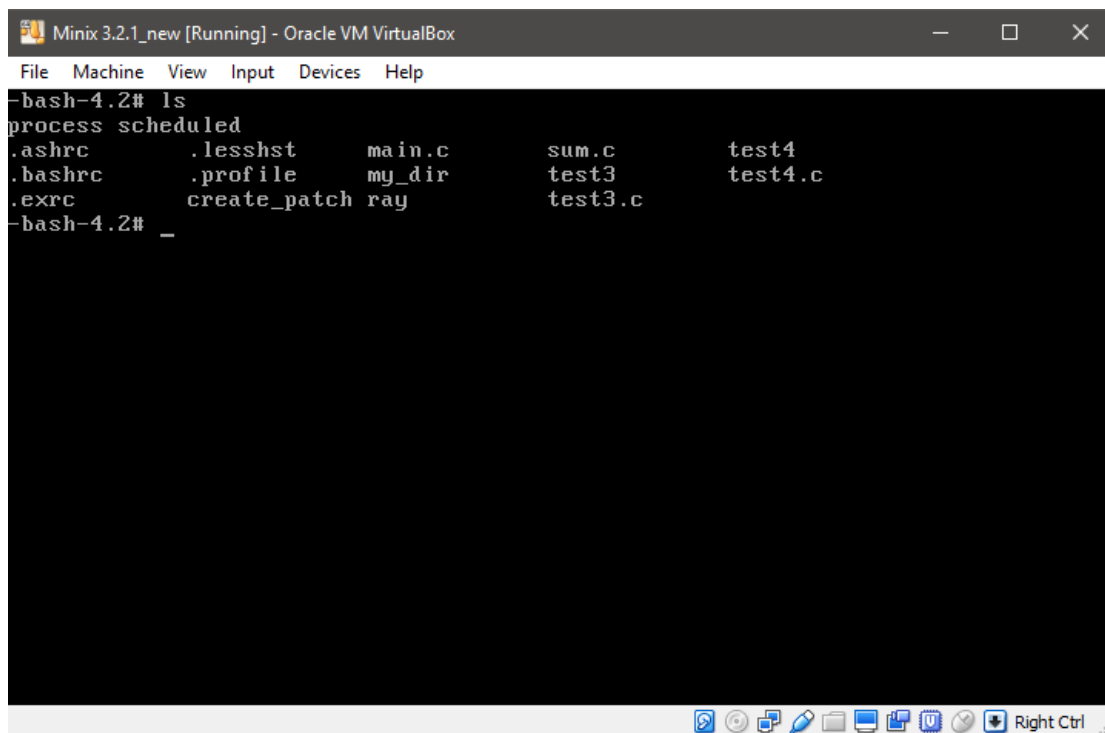
Το system call που πρέπει να τροποποιήσουμε για αυτό το ερώτημα είναι το `do_fork()` το οποίο βρίσκεται στο αρχείο `do_fork.c`

Path: `usr/src/kernel/system/do_fork.c`

Στην γραμμή 125 προσθέτουμε την εξής εντολή

Γραμμή	Εντολή
125	<code>printf("process scheduled\n");</code>

Screenshot



```
-bash-4.2# ls
process scheduled
.ashrc      .lessht     main.c      sum.c       test4
.bashrc     .profile    my_dir      test3       test4.c
.exrc       create_patch ray          test3.c
-bash-4.2# _
```

Ερώτημα 3

Λίστα τροποποιημένων αρχείων

- usr/src/include/minix/callnr.h
- usr/src/servers/ppm/table.c
- usr/src/servers/pm/proto.h
- usr/src/servers/pm/misc.c

Τροποποιήσεις

- usr/src/include/minix/callnr.h

Γραμμή	Εντολή
68	#define SYSCALL 69

- usr/src/servers/pm/table.c

Γραμμή	Εντολή
83	do_syscall, /* 69 = do_syscall */

- usr/src/servers/pm/proto.h

Γραμμή	Εντολή
--------	--------

- usr/src/servers/pm/misc.c: γραμμή 475+

```
int do_syscall()
{
    int i = 0;
    int counter = 0;
    /* Sum sys time of children */
    clock_t sumSys = 0;
    /* Sum user time of children */
    clock_t sumUser = 0;
    for(i = 0; i < NR_PROCS; i++){
        // Elegxoume an einai active to process
        if(mproc[i].mp_flags & IN_USE){
            counter++;
            sumSys += mproc[i].mp_child_stime;
            sumUser += mproc[i].mp_child_utime;
        }
    }
    printf("processes: %d\n",counter);
    printf("user time: %d\n",sumUser);
    printf("system time: %d\n",sumSys);
    return 0;
}
```

Έπειτα δημιουργήσαμε ένα νέο αρχείο στον φάκελο usr/src/include, syscalllib.h

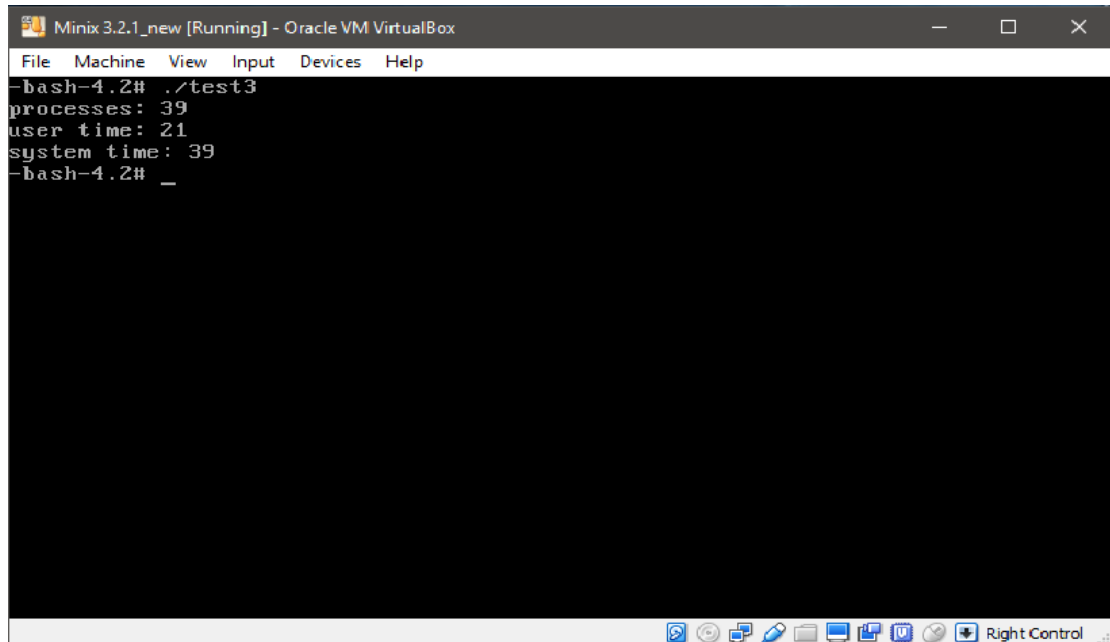
Γραμμή	Εντολή
1	#include <lib.h>
2	#include <unistd.h>
3	int do_syscall(){
4	message m;
5	return(_syscall(PM_PROC_NR,SYSCALL,&m));
6	}

Και τέλος φτιάξαμε ένα αρχείο test3.c το οποίο καλούσε την συνάρτηση do_syscall() που ορίσαμε στο παραπάνω αρχείο. Κάναμε compile το αρχείο με: clang -o test3 test3.c και προέκυψε το executable test3 το οποίο και τρέξαμε. Ο κώδικας του test3.c είναι :

```
#include <syscalllib.h>
```

```
int main(void)
{
    do_syscall();
    return 0;
}
```

Screenshot



Ερώτημα 4

Λίστα τροποποιημένων αρχείων

- usr/src/include/minix/callnr.h
- usr/src/servers/pm/table.c
- usr/src/servers/pm/proto.h
- usr/src/servers/pm/misc.c

Τροποποιήσεις

- usr/src/include/minix/callnr.h

Γραμμή	Εντολή
69	#define ACTIVE 70

- usr/src/servers/pm/table.c

Γραμμή	Εντολή
84	do_active,/* 70 = do_active */

- usr/src/servers/pm/proto.h

Γραμμή	Εντολή
--------	--------

- usr/src/servers/pm/misc.c: γραμμή 497+

```
int do_active()
{
    int i = 0;
    int found = 0;

    int mypid = (int)m_in.m1_i1;

    // Elegxoume oles tis diergasies gia na vroume afti me to mypid
    for(i = 0; i<NR_PROCS; i++)
    {
        if(mproc[i].mp_pid == mypid){
            found = 1;
            break;
        }
    }
    return (found);
}
```

Έπειτα δημιουργήσαμε ένα αρχείο activelib.h στο directory usr/src/include

Γραμμή	Εντολή
1	#include <lib.h>
2	#include <unistd.h>
3	m.m1_i1=mypid;
4	int do_active(int mypid){
5	message m;
6	m.m1_i1 = mypid;
7	return (_syscall(PM_PROC_NR,ACTIVE,&m));
8	}

Και τέλος φτιάξαμε ένα αρχείο test4.c το οποίο καλούσε την συνάρτηση do_active() που ορίσαμε στο παραπάνω αρχείο. Κάναμε compile το αρχείο με: clang -o test4 test4.c και προέκυψε το executable test4 το οποίο και τρέξαμε (εδώ προστέθηκε μια γραμμή που εκτυπώνει το αποτέλεσμα για να φανεί ότι δουλεύει σωστά ο κώδικας). Ο κώδικας του test4.c είναι :

```
#include <activelib.h>
#include <stdio.h>
int main()
{
```

```
    int num,found;  
    printf("Pid elegxou\n");  
    scanf("%d",&num);  
    found = do_active(num);  
    printf("elegxos:%d\n",found);  
    return (found);  
}
```

Screenshot

