

ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

ΑΣΚΗΣΗ 4

ΟΝΟΜΑ: ΣΑΒΒΑΣ ΓΕΩΡΓΙΟΥ

ΑΜ : 235037 / 1040803

ΕΡΩΤΗΜΑ Α

Λίστα τροποποιηθέντων αρχείων

- /usr/src/servers/vfs/protect.c

Τροποποιήσεις

/usr/src/servers/vfs/protect.c

Line	Code
95	printf("chmod: %s %o\n",fullpath,new_mode);

```
-bash-4.2# chmod 777 example.txt
chmod: example.txt 777
-bash-4.2# chmod 344 test.c
chmod: test.c 344
-bash-4.2#
```

ΕΡΩΤΗΜΑ Β

Λίστα τροποποιηθέντων αρχείων

- /usr/src/kernel/system/do_fork.c

Τροποποιήσεις

/usr/src/kernel/system/do_fork.c

Line	Code
110	printf("process forked\n");

Κατω απο το “ RTS_SET(rpc , RTS_VMINHIBIT); “ οπου ειναι η ενημερωση του process table του kernel που βρισκονται στο /usr/src/kernel/proc.h οι δηλωσεις.

```
http://wiki.minix3.org.  
process forked  
-bash-4.2# ls  
process forked  
.ashrc      .exrc      .ssh      create_patch  test.c  
.bash_history .lessht    changes.patch example.txt    test1  
.bashrc     .profile   core.149   test          test1.c  
-bash-4.2# ps  
process forked  
  PID TTY  TIME CMD  
  143 co   0:00 -bash  
  144 c1   0:00 getty  
  145 c2   0:00 getty  
  146 c3   0:00 getty  
  149 co   0:00 ps  
-bash-4.2# clang test1.c -o test1  
process forked  
chmod: /tmp/test1-00150a 100644  
process forked  
process forked  
chmod: test1 755  
-bash-4.2#
```

Οπως φαινεται στην εικονα , μια νεα διεργασια γινεται forked μετα απο το login , ενα ls , το ps και ενα compile με το clang.

ΕΡΩΤΗΜΑ Γ

Λίστα τροποποιηθέντων 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
- /usr/include/mycalls.h

Τροποποιήσεις

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

Line	Code
68	#define PROCNUM 69

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

Line	Code
83	do_procnum, /* 69 = number of running processes*/

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

Line	Code
59	int do_procnum(void);

/usr/src/servers/pm/misc.c

Line	Code
32	#include "glo.h"
67-76	<pre> / *===== =====* * do_procnum * *===== =====*/ int do_procnum(){</pre>

	<pre> int numberofproc = procs_in_use; //procs_in_use a variable in glo.h return numberofproc; } </pre>
--	---

/usr/include/mycalls.h

Line	Code
1-8	<pre> #include <lib.h> #include <unistd.h> int procnum(){ message m; return(_syscall(PM_PROC_NR,PROCNUM, &m)); } </pre>

Εφτιαξα ένα πρόγραμμα για τεστ του system call 69(procnum)

test69.c

```

#include <stdio.h>
#include <sys/types.h>
#include <mycalls.h>
#include <sys/stat.h>
#include <stdlib.h>

int main(int argc , char *argv[]){
    int numofproc = 0;
    numofproc = procnum();
    printf("%d\n" , numofproc);
}

```

```

-bash-4.2# ./test69
process forked
39
-bash-4.2# _

```

Το process forked είναι από το προηγούμενο ερώτημα. Το τυπώμα στην οθόνη είναι από το test69.c όπως φαίνεται πιο πάνω και όχι από το system call.

ΕΡΩΤΗΜΑ Δ

Λίστα τροποποιηθέντων 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
- /usr/include/mycalls.h

Τροποποιήσεις

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

Line	Code
69	#define FATHERPID 70

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

Line	Code
84	do_fatherpid, /* 70 = returns the pid of the father process*/

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

Line	Code
60	int do_fatherpid(int childpid);

/usr/src/servers/pm/misc.c

Line	Code
32	#include "glo.h"
78-95	<pre> / *===== ===== ====* * do_fatherpid * *===== ===== =====*/ int do_fatherpid(){</pre>

	<pre> int givenpid = m_in.m1_i1; int fathprocpid = 0; int i , fathindex; for(i=0; i<NR_PROCS; i++){ if(givenpid == mproc[i].mp_pid){ fathindex = mproc[i].mp_parent; fathprocpid = mproc[fathindex].mp_pid; break; } } return fathprocpid; } </pre>
--	--

/usr/include/mycalls.h

Line	Code
9-15	<pre> int fatherpid(int childpid){ message m; m.m1_i1 = childpid; return(_syscall(PM_PROC_NR,FATHERPID,&m)); } </pre>

Εφτιαξα ενα προγραμμα για τεστ του system call 70(fatherpid)

test70.c

<pre> #include <stdio.h> #include <sys/types.h> #include <mycalls.h> #include <sys/stat.h> #include <stdlib.h> int main(int argc , char *argv[]){ if(argc==2){ int givenpid = atoi(argv[1]); int ans; ans = fatherpid(givenpid); } } </pre>
--

```
        if(ans == 0 ){
            printf("process doesnt exists , returned value : %d \n" , ans);
        }else{
            printf("pid of process father , returned value : %d \n", ans);
        }
    }
}
```

```
-bash-4.2# clang test70.c -o test70
process forked
chmod: /tmp/test70-00195a 100644
process forked
process forked
chmod: test70 755
-bash-4.2# ./test70 8
process forked
pid of process father , returned value : 4
-bash-4.2# ./test70 66
process forked
process doesnt exists , returned value : 0
-bash-4.2#
```

Τα process forked όπως και το chmod είναι από τα προηγούμενα ερωτήματα. Το τυπώμα στην οθόνη είναι από το test70.c όπως φαίνεται πιο πάνω και όχι από το system call.

Η τιμή που επέστρεψε για το pid 8 είναι το pid 4 , ενώ για το 66 αφού δεν υπήρχε σαν pid , επέστρεψε 0.