PROJECT_ 2 (GROUP_2 | HENIL | VISESH | SHIVAM)

IMPLEMENTATION OF FOUR SYSTEM CALLS AND RESULTS

1) **Int trapcounter** : Here we implemented the do_trapcounter.c program in the /usr/src/minix/kernel/system.

```
CODE : #include "kernel/system.h"
#include <minix/endpoint.h>

#if USE_TRAPCOUNTER

static int times_called = 0;

int do_trapcounter(struct proc *caller, message *m_ptr) {
    times_called = times_called+1;
    printf("/usr/src/minix/kernel/system/do_trapcounter.c \n");
    m_ptr->m_pm_lsys_trapcounter.num = times_called;
    printf("minix/system/Kernel do_trapcounter %d", times_called);
    return OK;
}

#endif
```

2) /usr/src/minix/lib/libsys/sys_trapcounter.c

```
#include "syslib.h"

int sys_trapcounter(endpoint_t proc_ep, int *trapcounter_count)
{
   message m;
   int ret = _kernel_call(SYS_TRAPCOUNTER, &m);
   if (ret != OK)
{
   printf("libsys error in kernel call %d\n", ret);
   return ret;
}
   *trapcounter_count = m.m_pm_lsys_trapcounter.num;
   printf(" /usr/src/minix/lib/libsys/sys_trapcounter %d",
   m.m_pm_lsys_trapcounter.num);
   return(OK);

}
```

3) /usr/src/minix/servers/pm/trapcounter.c

```
#include "pm.h"
#include <minix/callnr.h>
#include <minix/com.h>
#include <minix/syslib.h>
```

```c
#include "glo.h"
#include "mproc.h"

int do_trapcounter()
{
 printf("usr/src/minix/servers/pm/trapcounter.c \n");
 register struct mproc *sending_proc = mp;
 int inc = 0;
 int r = sys_trapcounter(sending_proc->mp_endpoint, &inc);
 if(r != OK){
 printf("Error in kernel inside trapcounter.c [%d]\n", r);
 return r;
}
 printf("Success from trapcounter.c %d", inc);
 return inc;
}
```

4) Usr/src/minix/lib/libc/sys

```c
#include<sys/cdefs.h>
#include"namespace.h"
#include<lib.h>
#include<string.h>
#include<signal.h>

#ifdef __weak_alias
__weak_alias(kill, _kill)
#endif

int trapcounter()
{
message m;
printf("/usr/src/minix/lib/libc/sys/trapcounter.c");
memset(&m, 0, sizeof(m));
return (_syscall(PM_PROC_NR, PM_TRAPCOUNTER, &m));
}
```

II) **Void inittrapcounter:** Here we implemented the do_inittrapcounter.c program in the /usr/src/minix/kernel/system.

```c
#include "kernel/system.h"
#include <minix/endpoint.h>

#if USE_INITTRAPCOUNTER

static int times_called = 0;

int do_inittrapcounter(struct proc *caller, message *m_ptr)
{
```

```
    printf("/usr/src/minix/kernel/system/do_inittrapcounter.c \n");
    m_ptr->m_pm_lsys_trapcounter.num = times_called;
    printf("minix/system/Kernel do_inittrapcounter %d", times_called);
    return OK;
}

#endif
```

2) /usr/src/minix/lib/libsys/sys_inittrapcounter.c

```
#include "syslib.h"

int sys_inittrapcounter(endpoint_t proc_ep, int *trapcounter_count)
{
   message m;
   int r = _kernel_call(SYS_INITTRAPCOUNTER, &m);
   printf("VAlue of r %d\n", r);
   *trapcounter_count = m.m_pm_lsys_trapcounter.num;
   printf(" /usr/src/minix/lib/libsys/sys_trapcounter %d",
   m.m_pm_lsys_trapcounter.num);
   return OK;
}
```

3) /usr/src/minix/servers/pm/inittrapcounter.c

```
#include "pm.h"
#include <minix/callnr.h>
#include <minix/com.h>
#include <minix/syslib.h>
#include "glo.h"
#include "mproc.h"

int do_inittrapcounter()
{
 printf("usr/src/minix/servers/pm/inittrapcounter.c \n");
 register struct mproc *sending_proc = mp;
 int inc = 0;
 int r = sys_inittrapcounter(sending_proc->mp_endpoint, &inc);
 if(r != OK){
 printf("Error in kernel inside inittrapcounter.c [%d]\n", r);
 return r;
}
 printf("Success from inittrapcounter.c %d", inc);
 return inc;
}
```

4) Usr/src/minix/lib/libc/sys

```
5) #include<sys/cdefs.h>
   #include"namespace.h"
   #include<lib.h>
   #include<string.h>
   #include<signal.h>

   #ifdef __weak_alias
   __weak_alias(kill, _kill)
   #endif

   int inittrapcounter()
   {
   message m;
   printf("innittrapcounter m %d\n",m);
   memset(&m, 0, sizeof(m));
   printf("/usr/src/minix/lib/libc/sys/inittrapcounter.c");
   return (_syscall(PM_PROC_NR, PM_INITTRAPCOUNTER, &m));
   }
```

- Now for the remaining the three system call, I have created the zip file for that,