REPORT ON
P1: System Calls

TASK : SYSCALL get_access_level and get_access_level Implementation

Step # 1 //define accesslevel process attribute in task_structu structure in file sched.h:

Go to /usr/rep/src/reptilian-kernel/include/linux/

Open sched.h

Add a variable int accesslevel;

```
/*
 * New custom field accesslevel added for P1
 */
int                          accesslevel;
```

Step # 2 //initialize accesslevel attribute:

Go to /usr/rep/src/reptilian-kernel/include/linux

Open init_task.h

Initliaze the accesslevel attribute to 0 for all processes.

```
#define INIT_TASK(tsk)  \
{                                                    \
            INIT_TASK_TI(tsk)                                 \
.accesslevel = 0,                                \
}
```

Step # 3 //Implement syscall for accesslevel get and set functionality:

Go to /usr/rep/src/reptilian-kernel

create directory accesslevel

Define system call (syscall) function for access level get / set

sysaccesslevel.h

```
#ifndef ACCESSLEVEL
#define ACCESSLEVEL
        asmlinkage long sys_set_access_level(int pid, int new_level);
        asmlinkage long sys_get_access_level(int pid);
#endif
```

sysaccesslevel.c

```
//Implement syscall function using SYSCALL_DEFINEx function
SYSCALL_DEFINE2(set_access_level, int, pid, int, new_level)
SYSCALL_DEFINE1(get_access_level, int, pid)
```

Save sysaccesslevel.c and sysaccesslevel.h

Step # 4 //make accesslevel syscall functions

create a new Makefile in /usr/rep/src/reptilian-kernel/accesslevel

Add following line to compile sysaccesslevel.c file

```
obj-y:=sysaccesslevel.o
```

Edit Kernel Make File

Open /usr/rep/src/reptilian-kernel/Makefile

In kernel makefile, Add accesslevel directory to tell kernal that it need to look for sys_get_access_level and sys_set_access_level syscall in /accesslevel directory

```
core-y        += kernel/ certs/ mm/ fs/ ipc/ security/ crypto/ block/ accesslevel/
```

Step # 5 //Update syscall header file syscalls.h for new syscall functions

Go to /usr/rep/src/reptilian-kernel/include/linux

Open syscalls.h

Add following at the end of file before #endif

```
/*
 * syscall function added for P1 assignment
 */
asmlinkage long sys_set_access_level(int pid, int new_level);
asmlinkage long sys_get_access_level(int pid);
```

Step # 6 //Update syscall table file to add new syscall functions:

Goto /arch/x86/entry/syscalls

Open syscall_64.tbl

Add two new entries for newly added syscall (Take next available number)

```
335    common  set_access_level       __x64_sys_set_access_level
336    common  get_access_level       __x64_sys_get_access_level
```

Step # 7 //Recompile the kernel

TASK: CREATING LIBRARY functions to access newly implemented syscall get_access_level and set_access_level

Step # 1 // Implement set_access_level and get_access_level library functions

Create a directory accesslevel under /home/reptilian/p1/

Create new files as accesslevel.c and accesslevel.h

```
int set_access_level(int pid, int new_level)
{
        //return new_level on success and -1 otherwise
long temp = syscall(335, pid, new_level);
int temp2 = (int)temp;
return temp2;
}
```

```
int get_access_level(int pid)
{
//return access level on success, -1 if fails
printf("\nInside get_access_level library function pid : %d", pid);
long temp = syscall(336, pid);
int finally = (int)temp;
return finally;
}
```

(Library will expose set_access_level and get_access_level functions for any c program)

Step # 2 // Implement test harness functions:

```
int* retrieve_set_access_params (int pid, int new_level)
int* retrieve_get_access_params(int pid)
int interpret_set_access_result (int ret_value)
int interpret_get_access_result (int ret_value)
```

(Library will expose test harness functions for any c program)

TASK: User Program Implementation

getlevel.c

setlevel.c

accesstest.c

harnesstest.c

Above program has been reused for user level program and syscall function testing.

Makefile is used to create getlevel, setlevel, accesstest and harnesstest executable