CS3510: Operating System – I
Theory Assignment - 1

MS17BTECH11013
P. Pavan Kalyan

1) There are two main modes in which a program can be executed: user mode and kernel mode. Depending on the type of code, the processor switches between two modes. Applications programs run in user mode and system calls will be executed in kernel mode. Program in user mode has no direct access to hardware resources. Programs in kernel mode had direct access to memory. It is also called privileged mode. User mode is a safer mode of operation compared to kernel mode because if a program running in kernel mode fails then entire system crashes. When program in user mode wants to access memory/ Resources then it uses system calls to enter into kernel mode and if the OS recognises that these requests are dangerous then they are forbidden and hence protects our data. User doesn’t need to have the knowledge on how the internal programs work so we can hide the implementation details from user and indirectly talk to the kernel mode through user mode. Hence user mode acts as an abstraction and we have limited capability when in user mode and thus it forms as a rudimentary form of protection system.

2)

a. Set value of timer.

Timer can be really important to determine whether a process should continue executing by the OS. so, modifying the value of it can lead it to run the process continuously. So, it should be a **privileged** instruction.

b. Read the clock.

As we are not modifying the data here. We are just accessing the data. So, it should **not be a privileged** instruction.

c. Clear memory.

This should be **privileged** instruction. If not, then a process can access data and can clear it. There may be a possibility that it can clear some important data like the OS code in memory.

d. Issue a trap instruction.

Trap is an interrupt to the OS to save the current process in it and run the new process (context switching). So, keeping it as a non-privileged instruction a program can generate trap instruction by itself many times and can run itself in the memory even if it is not important. So, it should be a **privileged** instruction.

3. The two difficulties that could arise are:

i. We can’t update the OS at all as it can’t be accessed even by itself. Even though it gives high security from hackers still if the OS designer makes a bug which can’t be modified and it might lead to corruption of data. Any new device whose device controllers need to be installed in the OS will be difficult and it’s not useful.

ii. The OS need to save passwords for user login in some other memory locations which can be easily accessed by some other user and can steal the data.

iii. All the instructions must be executed in one mode either user mode / kernel mode because we can’t access the OS to do tasks requiring resources when in user mode and if it’s in always in kernel mode then memory data becomes vulnerable.