**Lab No : 03**

**Name of the Lab : Threads on Operating System**

**Name: Mehedi Hasan Mithun**

**ID: IT-17029**

**Threads on Operating System**

**Q1. What is Thread?**

**Answer:**

A thread is a small set of instructions designed to be scheduled and executed by the CPU independently of the parent pocess.For example, a program may have an open thread waiting for a specifiq event to occur or running a separate job, allowing the main program to perform other tasks. A program is capable of having multiple threads open at once and will either terminate or suspend them after a task is completed, or the program is closed.

**Q2. Explain types of thread?**

**Answer:**

Threads are implemented in two ways.

* User Level Threads
* Kernel Level Threads

**User Level Threads**

User level threads is managed by user.User level threads are faster to create and manage. Implemented by a thread\_library user level.User level thread can run on any orerating system.

Advantages of user level Threads:

* Thread switching does not require Kernel mode privileges.
* User level thread can run on any operating system.
* Scheduling can be application specific in the user level thread.
* User level threads are fast to create and manage.

Disadvantages of user level Threads:

* In a typical operating system, most system calls are blocking.
* Multithreaded application cannot take advantage of multiprocessing.

**Kernel Level Threads**

Kernel level Threads are mainly managed by operating system. Kernel level Threads are slower to create and manage.operating system support directly to kernel threads.

Advantages of Karnel level Threads:

* Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
* If one thread in a process is blocked, the Kernel can schedule another thread of the same process.
* Kernel routines themselves can multithreaded.

Disadvantages of Karnel level Threads:

* Kernel threads are generally slower to create and manage than the user threads.
* Transfer of control from one thread to another within the same process requires a mode switch to the Kernel.

**Q3. Implementation of Threads?**

**Answer :**

There are two ways of implementation of Threads..

* Implementation in user space
* Implementation in kernel

**Threads Implementation in User Space:**

In this model of implementing the threads package completely in user space, the kernel don't know anything about them.The advantage of implementing threads package in user space is that a user-level threads package can be implemented on an OS that doesn't support threads.All of these implementations have the same general structure as illustrated in the figure given below.



**Threads Implementation in Kernel:**

In this method of implementing the threads package entirely in the kernel, no any run-time system is need in each as illustrated in the figure given below.



In this, there is no any thread table in each process. But to keep track of all the threads in the system, the kernel has the thread table.

In addition to these, to keep track of processes, the kernel also maintains the traditional process table.

**Other two methods are as follows:**

* Hybrid implementation
* Scheduler activation

**Hybrid implementation:**

In this implementation, there is some set of user-level threads for each kernel level thread that takes turns by using it.

**Scheduler activation:**

The objective of this scheduler activation work is to replicate the working or function of kernel threads, but with higher performance and better flexibility which are usually related to threads packages which are implemented in userspace.

**Conclusion:**

Through this lab we learn about the importance of threads in Operating system. Different threads type, how they worked and what they served to make easier our UI & how they get implemented. The blessing of multitasking is only make possible by thread. Concurrent access of any program can performed on OS by both user &karnel basis. So we learn both of those process.