Lab No : 03

Name of the lab : Threads on Operating System

Name: Golam Kibria Tuhin ID:IT-18015

Objective: Threads, types of threads, how threads get implemented & worked on our OS.

Q.1 What is Thread?

Answer:

Is general words THREAD is the tiniest unit of the processing which can be participated in the OS. During this present era's OS , thread exists within a process, like a single process may contain multiple threads.

We can imagine multitasking as something that allows processes to run simultaneously while multithreading allows sub-processes to run concurrently.

When multiple task are running concurrently, this is known as multithreading, which is similar to multitasking. Basically, an OS with multitasking capabilities allows programs to run without being corrupted at the same time. Besides, a single program with multithreadingcapabilities allows individual sub-processes to run seemingly at the same time.

Q.2 Explain types of thread?

Answer:

Threads can be classified in following two ways especially on the basis of operating system. i.

User Level Threads

ii. Kernel Level Threads

User Level Threads

Inthis case, application manages thread management kernel is not aware of the existence of threads. The thread library contains code for creating and destroying threads, for passing message and data

between threads, for scheduling thread execution and for saving and restoring thread contexts. The application begins with a single thread and begins running in that thread.

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.

Kernel Level Threads

In this case, thread management done by theKernel. There is no thread management code in the application area. Kernel threads are supported directly by the operating system. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.

The Kernel maintains context information for the process as a whole and for individuals threads within the process. Scheduling by the Kernel is done on a thread basis. The Kernel performs thread creation, scheduling and management in Kernel space. Kernel threads are generally slower to create and manage than the user 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.

Q.3 Implementation of Threads?

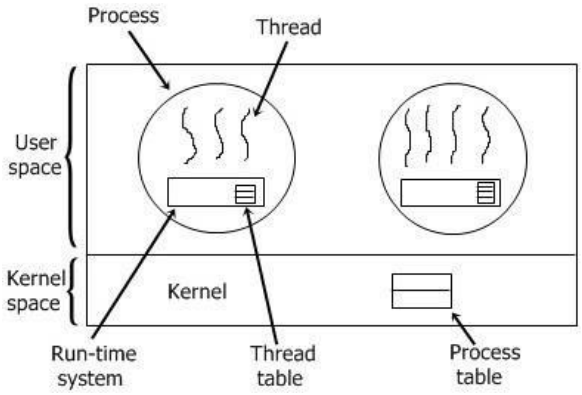
Answer :

To implement a threads package, there are the following two ways basic on the operating system.

i. Implementation in user space
ii. Implementation in kernel

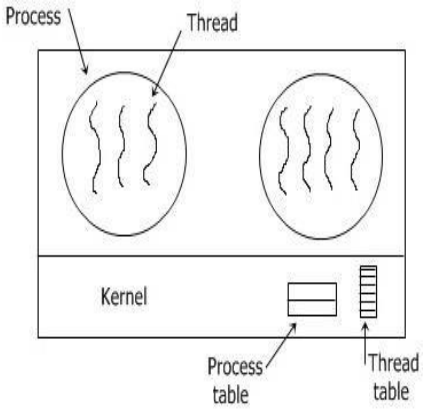
Threads Implementation in User Space:

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.

Whenever a thread wants to create a new thread or destroy an existing thread, then it makes a kernel call, which does the creation or destruction just by updating the kernel thread table. The thread table of the kernel holds each registers, state, and some other useful information of the thread . Here the information is the same as with the user-level threads .This information is a subset of the information that traditional kernels maintains about each of their single-threaded processes, that is, the process state.

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

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.