Mawlana Bhashani Science and Technology University

Lab-Report

Lab Report No: 03

Lab Report Name: Threads on Operating System

Course code: ICT-3110

Course title: Operating System Lab

Date of Performance: 7/9/2020

Date of Submission:

Submitted by

Name: Afra Ibnat Tethye

ID: IT-18055

3rd year 1st semester

Session: 2017-18

Dept. of ICT

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT,

MBSTU.

Experiment Name: Threads on operating system.

Question-01:

What is Thread?

Answer:

A thread otherwise called a lightweight process (LWP) is a basic unit of CPU utilization, it comprises of a thread id, a program counter, a register set and a stack. It shares with other threads belonging to the same process its code section, data section, and operating system resources such as open files and signals. 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.

Question-02:

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. ii. Kernel Level Threads

User Level Threads :

User threads are supported above the kernel and are implemented by a thread library at the user level. Thread creation & scheduling are done in the user space, without kernel intervention. Therefore they are fast to create and manage blocking system call will cause the entire process to block

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:

Kernel threads are supported directly by the operating system .Thread creation, scheduling and management are done by the operating system. Therefore they are slower to create & manage compared to user threads. If the thread performs a blocking system call, the kernel can schedule another thread in the application for execution.

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.

Question-03:

Implementation of Threads?

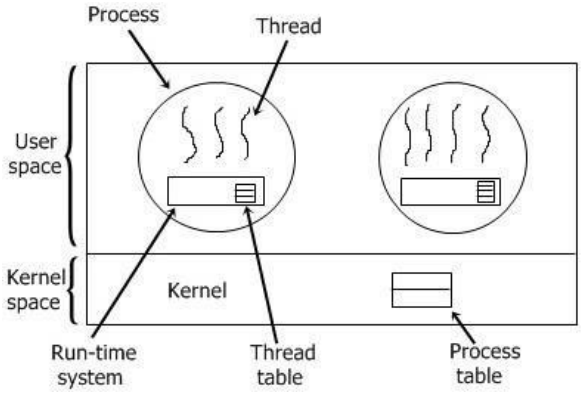
Answer:

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

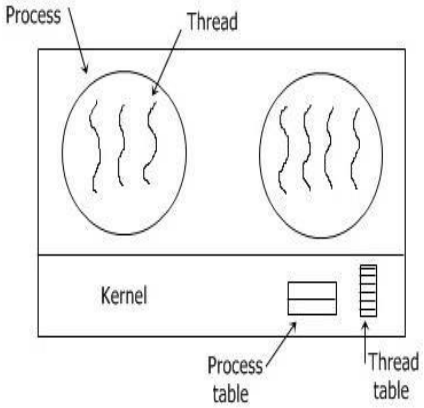
Threads Implementation in User Space:

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. Thethread 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.