Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 03

Course code: ICT-3110

Course title: Operating Systems Lab

Date of Performance :

Date of Submission : 14/09/2020

Submitted by

Name: Mahfuza Talukdar

ID:IT-18009

3rd year 1stsemester

Session: 2017-2018

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

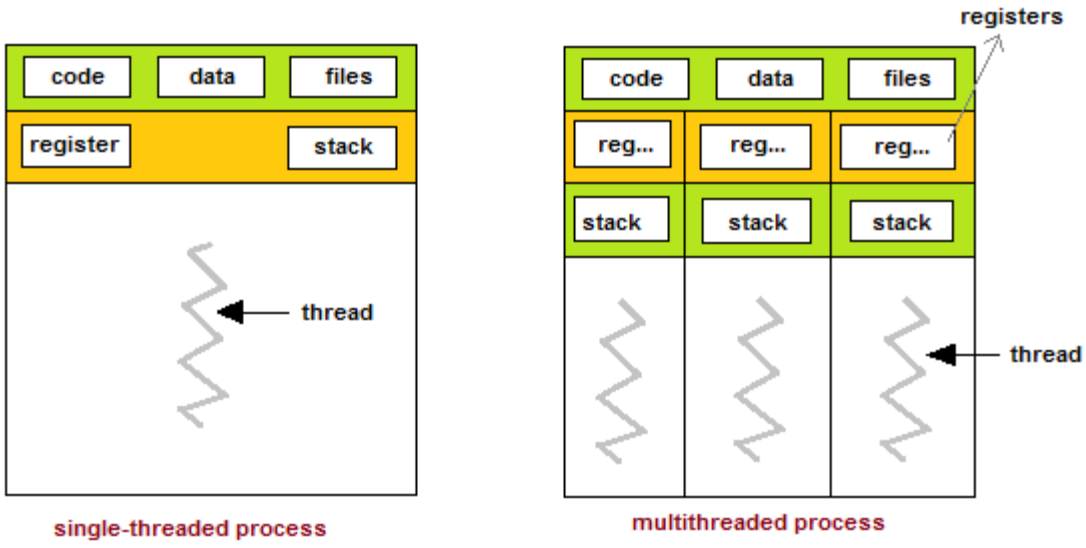
MBSTU.

Experiment No : 03

Experiment Name : Threads on Operating System.

Theory :

as a result threads shares with other threads their code section, Despite of the fact that a thread must execute in process, the process and its associated threads are different concept. Processes are used to group resources together and threads are the entities scheduled for execution on the CPU.
A thread is a single sequence stream within in a process. Because threads have some of the properties of processes, they are sometimes called *lightweight processes*. In a process, threads allow multiple executions of streams. In many respect, threads are popular way to improve application through parallelism. The CPU switches rapidly back and forth among the threads giving illusion that the threads are running in parallel. Like a traditional process i.e., process with one thread, a thread can be in any of several states (Running, Blocked, Ready or Terminated). Each thread has its own stack. Since thread will generally call different procedures and thus a different execution history. This is why thread needs its own stack. An operating system that has thread facility, the basic unit of CPU utilization is a thread. A thread has or consists of a program counter (PC), a register set, and a stack space. Threads are not independent of one other like processes data section, OS resources also known as task, such as open files and signals.



Types of Thread : In the operating system, there are two types of threads.

1. Kernel level thread.
2. User-level thread.

1.User-level thread :

The operating system does not recognize the user-level thread. User threads can be easily implemented and it is implemented by the user. If a user performs a user-level thread blocking operation, the whole process is blocked. The kernel level thread does not know nothing about the user level thread. The kernel-level thread manages user-level threads as if they are single-threaded processes?examples: Java thread, POSIX threads, etc.

Advantages of User-level threads

1. The user threads can be easily implemented than the kernel thread.
2. User-level threads can be applied to such types of operating systems that do not support threads at the kernel-level.
3. It is faster and efficient.
4. Context switch time is shorter than the kernel-level threads.
5. It does not require modifications of the operating system.
6. User-level threads representation is very simple. The register, PC, stack, and mini thread control blocks are stored in the address space of the user-level process.
7. It is simple to create, switch, and synchronize threads without the intervention of the process.

Disadvantages of User-level threads

1. User-level threads lack coordination between the thread and the kernel.
2. If a thread causes a page fault, the entire process is blocked.

2.Kernel-level thread :

The kernel thread recognizes the operating system. There are a thread control block and process control block in the system for each thread and process in the kernel-level thread. The kernel-level thread is implemented by the operating system. The kernel knows about all the threads and manages them. The kernel-level thread offers a system call to create and manage the threads from user-space. The implementation of kernel threads is difficult than the user thread. Context switch time is longer in the kernel thread. If a kernel thread performs a blocking operation, the Banky thread execution can continue. Example: Window Solaris.

Advantages of Kernel-level threads

1. The kernel-level thread is fully aware of all threads.
2. The scheduler may decide to spend more CPU time in the process of threads being large numerical.

3. The kernel-level thread is good for those applications that block the frequency.

Disadvantages of Kernel-level threads

1. The kernel thread manages and schedules all threads.
2. The implementation of kernel threads is difficult than the user thread.
3. The kernel-level thread is slower than user-level threads.

Implementation of threads :

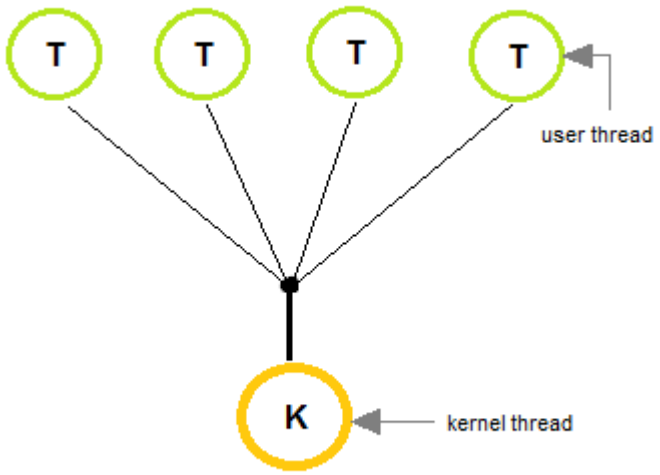
In general, user-level threads can be implemented using one of four models.

- Many-to-one
- One-to-one
- Many-to-many
- Two-level

All models maps user-level threads to kernel-level threads. A **kernel thread** is similar to a process in a non-threaded (single-threaded) system. The kernel thread is the unit of execution that is scheduled by the kernel to execute on the CPU. The term **virtual processor** is often used instead of kernel thread.

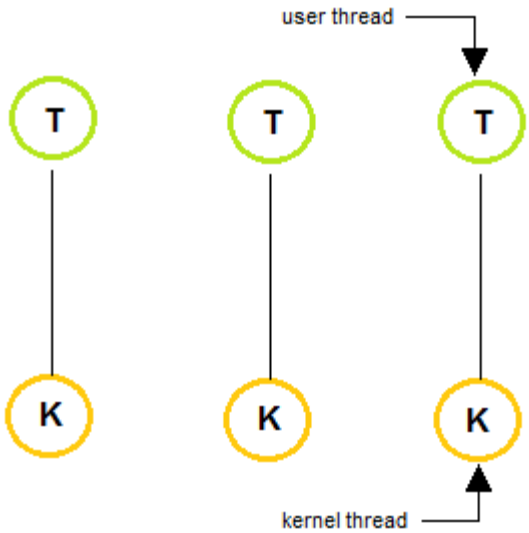
1.Many to One Model :

- In the **many to one** model, many user-level threads are all mapped onto a single kernel thread.
- Thread management is handled by the thread library in user space, which is efficient in nature.



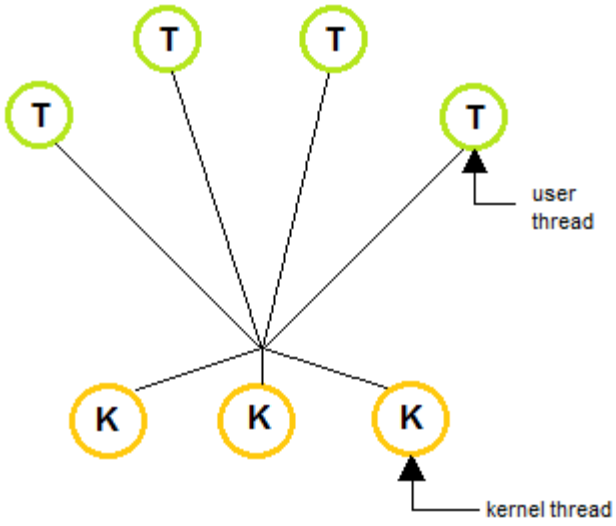
2.<u>One to One Model :</u>

- The **one to one** model creates a separate kernel thread to handle each and every user thread.
- Most implementations of this model place a limit on how many threads can be created.
- Linux and Windows from 95 to XP implement the one-to-one model for threads.

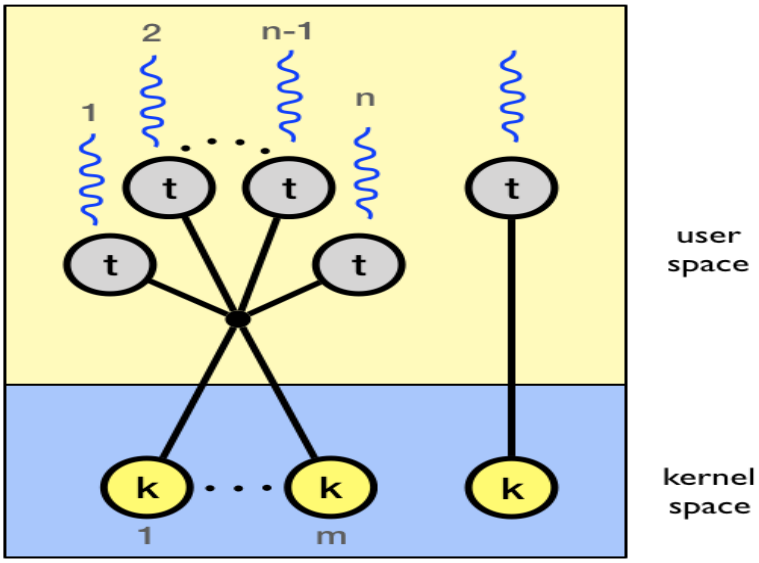


3.<u>Many to Many Model :</u>

- The **many to many** model multiplexes any number of user threads onto an equal or smaller number of kernel threads, combining the best features of the one-to-one and many-to-one models.
- Users can create any number of the threads.
- Blocking the kernel system calls does not block the entire process.
- Processes can be split across multiple processors.



4.<u>Two level</u> : The two-level model is similar to the many-to-many model but also allows for certain user-level threads to be bound to a single kernel-level thread.



Discussion : From this lab,we learn about Threads on Operating System. We see how the implementation of threads happen in Operating System. We can know more about it in future.