**THREADS**

A Thread is an instance within a process. Many threads can run at the same time in a CPU

**Each thread shares the following**:

1. Address space
2. Open Files
3. Signal Handler
4. Code
5. Data

**Each thread has unique**:

1. Thread ID
2. Stack
3. Register
4. Priority

Every time a process is created, it creates a thread known as single-threaded process. Threads can run in kernel and user mode

**Advantages of threads:**

1. Responsiveness means that a multithreaded program can allow a program to run even if part of it is blocked.
2. Resource sharing occurs when an application has several different threads of activity within the same address space. Threads share the resources of the process to which they belong. As a result, it is more economical to create new threads than new processes.
3. Finally, a single-threaded process can only execute on one processor regardless of the number of processors actually present. Multiple threads can run on multiple processors, thereby increasing efficiency

**Disadvantages of threads**:

1. Synchronization- Many threads running at the same time and accessing shared resources can create data that is non-consistent . Threads need to be synchronize to make sure that only one thread access the shared resource at a time. In POSIX threads the pthread-join( ) system call can be used to synchronize threads
2. If one thread in a program calls fork() does the new process duplicate all the threads or is the new process single-threaded? Unix systems supports both versions of fork() some processes duplicate the thread other processes are just single-threaded
3. Signaling Handle. Handling signals in a multithreaded system is complicated. Where a signal should be delivered?.
   1. Deliver the signal to the thread to which signals applies
   2. Deliver the signal to every thread in the process
   3. Deliver the signal to certain threads in the process

e.g. CTRL-C signal should be sent to all threads

**Threads Types**

1. POSIX threads pthreads . POSIX threads uses the following functions
   1. Pthread\_create() creates a new thread
   2. Pthread\_join() synchronizes threads . It makes the thread to wait for its child thread to complete
   3. Pthread-exit( ) to terminate a thread
   4. POSIX threads need the library <pthread.h>
   5. Each thread has a thread ID that is type pthread\_t
2. Windows Threads
3. Java Threads

**Thread Pools**

Thread pools is a set of threads created at process startup, placed into a pool and waiting for work. When a server receives a request, it takes a thread from the pool. Once the thread is done it returns to the pool

**Threads can be in different states:**

1. Initial
2. Running
3. Standby
4. Transition
5. Terminated
6. Idle