User Level v/s Kernel Level Threads | Operating System – M01 P12

This is a multipart blog article series, and in this series I am going to explain you the concepts of operating system. This article series is divided into multiple modules and this is the first module which consists of 12 articles.

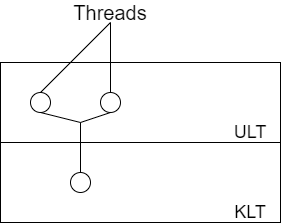
In this article you will the difference between “User level” and “Kernel level” threads in an operating system. Here, we are talking about “multiprocessing/multitasking” environment.

Remember one thing that, user level thread and kernel level thread always share code, file and data while they have different stack and registers, that’s why these are known as “Light weight processes.”

We will see the difference in between these with the help of a table.

|  |  |
| --- | --- |
| User level thread | Kernel level thread |
| User level threads are managed by user level libraries. It means that it is the responsibility of application to create user level thread. | Kernel level threads are managed by operating system. We have to use system calls for it. |
| User level threads are typically fast. | Kernel level threads are slower then user level threads. |
| Context switching is faster. | Context switching is slower. |
| If one user level thread performs blocking operations then entire process get blocked. | If one kernel level thread blocked, no affect on others. |

**Hybrid environment:**



* At present we use hybrid environment, which means that we have both user level thread and kernel level thread.
* Generally we use two user level threads and map it with one kernel level thread.
* We can map it by using “many to one”, “one to many”, “many to many” method. Just like Linux base operating system in general use “one to one” method, to map user level thread and kernel level thread.

Context switching time :- Process > KLT > ULT

This was all about kernel level thread and user level thread. Hope you liked it and learned something new from it.

If you have any doubt, question, queries related to this topic or just want to share something, then please feel free to contact me.