Overview of Threads in Windows and Linux

Vaibhav Mehta (00005005)

Aakash Rawat (00D05002)

Shashank Pandit (00005011)

What are threads?

- Threads are "light weight processes".
- "Heavy-weight processes" have a significant amount of switching overhead. All the tables have to be flushed from the processor for each task switch and the only shared information is through pipes and shared memory.
- Threads reduce overhead by sharing fundamental parts. By sharing these parts, switching happens much more frequently and efficiently.

Types of threads:

- User-Space Threads
  - Avoids the kernel and manages the tables itself.
  - Threads change by calling switches or sending signals.
  - Starving of threads can occur.
  - Cannot take advantage of Multi-Processor Systems.
- Kernel-Space Threads
  - Implemented in the kernel using several tables.
  - Clockticks determine the switching times.
  - Can take advantage of Multi-Processors.

Threads in Linux:

- Threads in Linux are handled quite differently from most other operating systems due to the open source nature of Linux and it's configurability.
- Linux supports both kernel level and user level threading or their combination.
- Linux does not support multithreading as these are more likely to crash.
  - Several objects hidden from applications and shared inadvertently
  - Conflicts between some libraries
- Threads in linux are not reentrant. Hence, threads making various system calls will block other threads till call completion.

Threads in Windows :

- Windows NT is a multi-threaded environment in which each process contains at least one thread of execution.

- Most lengthy operations will take place in a separate background thread with low priority while the user continues to work in a separate, normal priority thread.

- Windows NT also will take advantage of a machine using multiple processors by running each thread on a separate CPU.

- Windows NT utilzies threads to help with Interprocess communications, Overlapped I/O, Security, and Synchronization.

Comparison:

- Linux kernel-space threads are processes and are not handled differently from fork'ed tasks (unlike most other multi-threaded OS's). Therefore, process scheduling, cancelling, suspending, etc. is controlled via signals.

- Linux is an inherrently more stable OS. Linus Torvalds defined a thread as "a context of execution" which means that only one process/thread table and one scheduler is needed. Obviously, this lends to stability because there is less risk of conflict.

References

1. http://www.northco.net/chenke/project/linux.html

2. http://linas.org/linux/threads-faq.html

3. http://www.icewalkers.com/opd/Computers/Programming/Threads/

4. http://home.iae.nl/users/mhx/i4threads.html