

Table 4.1 Thread and Process Operation Latencies (μ s) [ANDE92]

Operation	User-Level Threads	Kernel-Level Threads	Processes
Null Fork	34	948	11,300
Signal Wait	37	441	1,840

Table 4.2 Relationship Between Threads and Processes

Threads:Processes	Description	Example Systems
1:1	Each thread of execution is a unique process with its own address space and resources.	Traditional UNIX implementations
M:1	A process defines an address space and dynamic resource ownership. Multiple threads may be created and executed within that process.	Windows NT, Solaris, Linux OS/2, OS/390, MACH
1:M	A thread may migrate from one process environment to another. This allows a thread to be easily moved among distinct systems.	Ra (Clouds), Emerald
M:N	Combines attributes of M:1 and 1:M cases.	TRIX

Table 4.3 Windows Process Object Attributes

Process ID	A unique value that identifies the process to the operating system.
Security Descriptor	Describes who created an object, who can gain access to or use the object, and who is denied access to the object.
Base priority	A baseline execution priority for the process's threads.
Default processor affinity	The default set of processors on which the process's threads can run.
Quota limits	The maximum amount of paged and nonpaged system memory, paging file space, and processor time a user's processes can use.
Execution time	The total amount of time all threads in the process have executed.
I/O counters	Variables that record the number and type of I/O operations that the process's threads have performed.
VM operation counters	Variables that record the number and types of virtual memory operations that the process's threads have performed.
Exception/debugging ports	Interprocess communication channels to which the process manager sends a message when one of the process's threads causes an exception.
Exit status	The reason for a process's termination.

Table 4.4 Windows Thread Object Attributes

Thread ID	A unique value that identifies a thread when it calls a server.
Thread context	The set of register values and other volatile data that defines the execution state of a thread.
Dynamic priority	The thread's execution priority at any given moment.
Base priority	The lower limit of the thread's dynamic priority.
Thread processor affinity	The set of processors on which the thread can run, which is a subset or all of the processor affinity of the thread's process.
Thread execution time	The cumulative amount of time a thread has executed in user mode and in kernel mode.
Alert status	A flag that indicates whether the thread should execute an asynchronous procedure call.
Suspension count	The number of times the thread's execution has been suspended without being resumed.
Impersonation token	A temporary access token allowing a thread to perform operations on behalf of another process (used by subsystems).
Termination port	An interprocess communication channel to which the process manager sends a message when the thread terminates (used by subsystems).
Thread exit status	The reason for a thread's termination.

Table 4.5 Linux clone () flags

CLONE_CLEARID	Clear the task ID.
CLONE_DETACHED	The parent does not want a SIGCHLD signal sent on exit.
CLONE_FILES	Shares the table that identifies the open files.
CLONE_FS	Shares the table that identifies the root directory and the current working directory, as well as the value of the bit mask used to mask the initial file permissions of a new file.
CLONE_IDLETASK	Set PID to zero, which refers to an idle task. The idle task is employed when all available tasks are blocked waiting for resources.
CLONE_NEWNS	Create a new namespace for the child.
CLONE_PARENT	Caller and new task share the same parent process.
CLONE_PTRACE	If the parent process is being traced, the child process will also be traced.
CLONE_SETTID	Write the TID back to user space.
CLONE_SETTLS	Create a new TLS for the child.
CLONE_SIGHAND	Shares the table that identifies the signal handlers.
CLONE_SYSVSEM	Shares System V SEM_UNDO semantics.
CLONE_THREAD	Inserts this process into the same thread group of the parent. If this flag is true, it implicitly enforces CLONE_PARENT.
CLONE_VFORK	If set, the parent does not get scheduled for execution until the child invokes the <i>execve()</i> system call.
CLONE_VM	Shares the address space (memory descriptor and all page tables).