CSCI 460 Operating Systems

Processes & Threads

Professor Travis Peters
Fall 2019

Some slides & figures adapted from Stallings instructor resources.

Some slides adapted from Adam Bates's F'18 CS423 course @ UIUC
https://courses.engr.illinois.edu/cs423/sp2018/schedule.html
Following Up on Questions About Boot Loaders

- BIOS is dead…
  - bootable-drive size limitations (MBR)
  - 16-bit mode / mem. limitations (1MB)
  - no networking pre-OS
  - + slow, etc.
- …long live UEFI! – [https://uefi.org](https://uefi.org)
  - boot from larger drives, run in 32-/64-bit mode, better UI, Secure Boot, networking/remote config.
  - \textit{UEFI is essentially a tiny OS!}
- Other relics…
  - CMOS vs. Flash/EEPROM, …
Goals for Today

• **Learning Objectives**
  • Understand the basic concept of threads and how they relate to the concept of a process

• **Announcements**
  • zyBook for OS is now accessible (optional)
  • *Coming Soon*…
    • 1st programming assignment — on basic sys programming + concurrency
Processes vs. Threads

- What is the difference between a process and a thread?
- Processes can be further divided in terms of their responsibilities:
  - 1st Part = resource ownership (process or task)
  - 2nd Part = scheduling/execution (thread or lightweight process)
- Multithreading — the ability of an OS to support multiple, concurrent paths of execution within a single process.
Processes vs. Threads

- Both provide independent execution sequences, but…
  - Processes…
    - each have their own private memory space
    - each have their own resources (protected access to processors, other processes (IPC), files, and I/O (devices and channels))
  - Threads…
    - run in a shared memory space (the process)
    - have their own execution state (Running, Ready, etc.), execution context (think PC), execution stack, some “thread local” storage, etc.

➡ Potentially many threads per process
Processes vs. Threads — **WHY**

> Threads (can) prevent a process from blocking entirely

![Figure 4.3 Remote Procedure Call (RPC) Using Threads](image-url)
Processes vs. Threads — **WHY**

It takes far less time to create, terminate, switch between, and communicate among threads as compared to processes!

Table 4.1  Thread and Process Operation Latencies (μs)

<table>
<thead>
<tr>
<th>Operation</th>
<th>User-Level Threads</th>
<th>Kernel-Level Threads</th>
<th>Processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Null Fork</td>
<td>34</td>
<td>948</td>
<td>11,300</td>
</tr>
<tr>
<td>Signal Wait</td>
<td>37</td>
<td>441</td>
<td>1,840</td>
</tr>
</tbody>
</table>
What is the difference between user threads and kernel threads?

Pros and cons?

User Threads
- Fast context switching
- Customized scheduling

Kernel Threads
- Each thread can make blocking I/O calls
- Can run concurrently on multiple processors
After Today, Be Sure to Review
Processes & Threads in UNIX & The Linux Kernel

### UNIX Process Structure

<table>
<thead>
<tr>
<th>Process ID</th>
<th>User IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal Dispatch Table</td>
<td>Memory Map</td>
</tr>
<tr>
<td>Priority</td>
<td>Signal Mask</td>
</tr>
<tr>
<td>Registers</td>
<td>STACK</td>
</tr>
<tr>
<td>File Descriptors</td>
<td>Processor State</td>
</tr>
</tbody>
</table>

---

**Figure 2.19** Linux Kernel Components

- signals
- system calls
- virtual memory
- processes & scheduler
- file systems
- network protocols
- char device drivers
- block device drivers
- network device drivers
- traps & faults
- physical memory
- interrupts
- CPU
- system memory
- terminal
- disk
- network interface controller

**Figure 4.13** Process Structure in Traditional UNIX and Solaris [LEWI96]
Activity: Think-Pair-Share Summary of Processes & Threads

1. Spend a few minutes alone summarizing everything you remember about processes, and the role of the OS in controlling them and managing resources.

   *Use whiteboards around the room to write on?!*

2. Spend a few minutes reviewing your summary with a neighbor.

   *Did you miss anything?*

3. Come back together and share.

   *What idea(s) seem to be the most critical to understand going forward?*

   *What is still unclear?*