CSc 360
Operating Systems
Threads

Jianping Pan

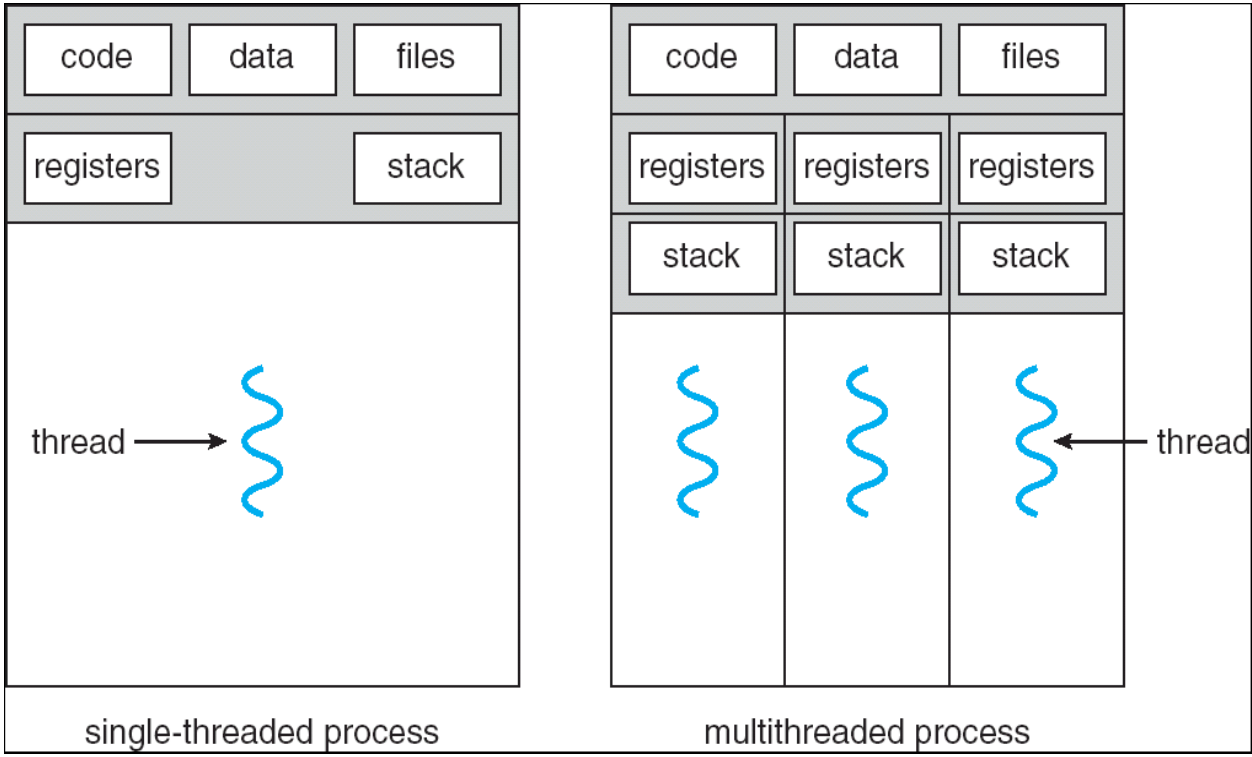
Summer 2015

Review: process

- Process
  - a running program + allocated resources
- Process scheduling
  - how to handle many processes (PCB)
- Process operation: play around in P1!
  - create processes and load a new program
- Process communication
  - shared memory vs message passing

Program, process, thread

- In one process
  - easy to share
- Btw processes
  - multitasking
- Best of both
  - thread
    - one process
    - multitasking



Q: browsers to use multi-process?

Threads

- Thread
  - a basic unit of CPU utilization
    - thread state, program counter, register set, stack
  - share with other threads in the same process
    - code, data, opened files, signals, etc
- Benefits
  - responsiveness: multithreading
  - resource sharing, efficiency, MP architectures

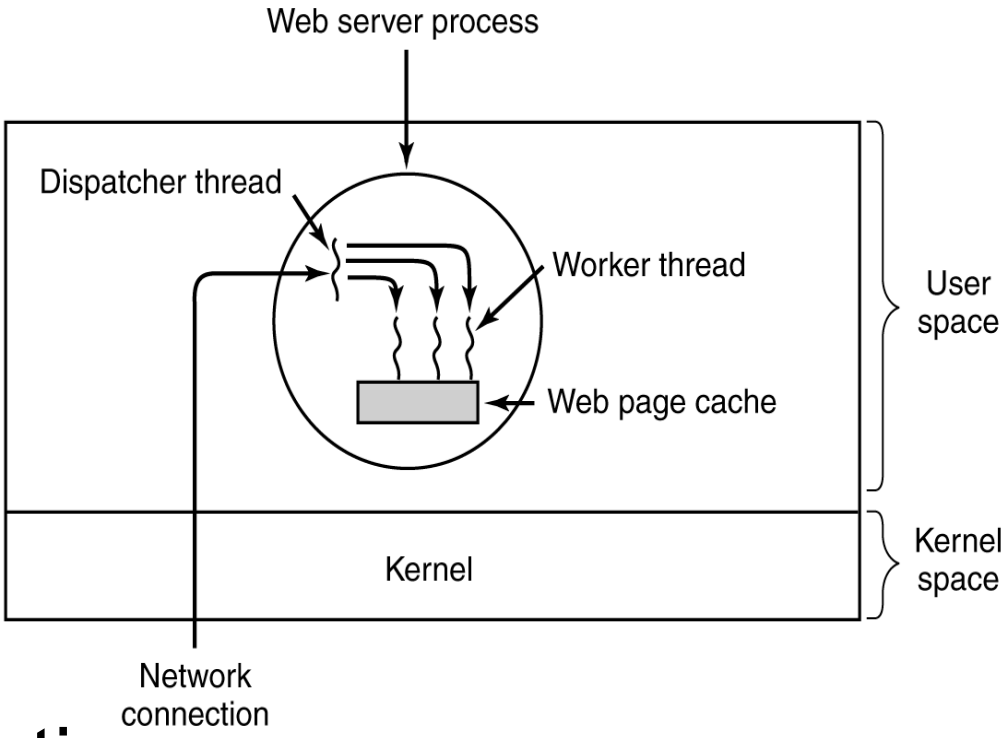
Q: potential problems?

Single-threaded Web server

- Web server with cache and disk
  - wait for a request
  - process the request
    - check cache; if hit, break
    - otherwise, retrieve from disk (relatively slow)
  - respond the request
- One request at a time
  - or create a new process on each request
    - expensive!

Multi-threaded Web server

- *Dispatcher* thread
  - wait for a request
  - handoff the request
- *Worker* threads
  - process the request
    - disk I/O
  - respond the request
- "Many" requests at a time

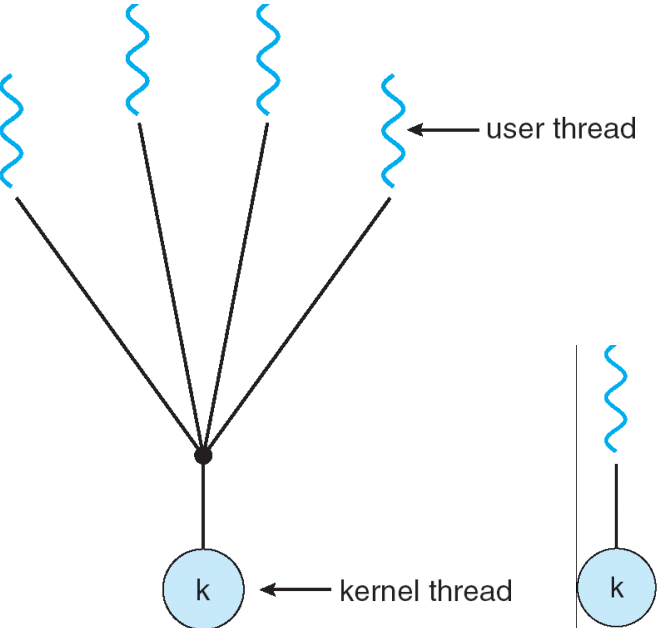


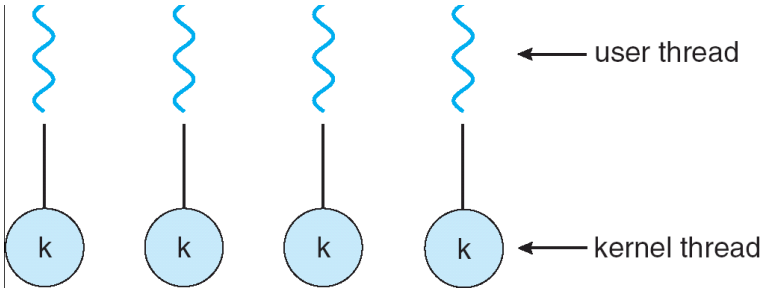
User vs kernel threads

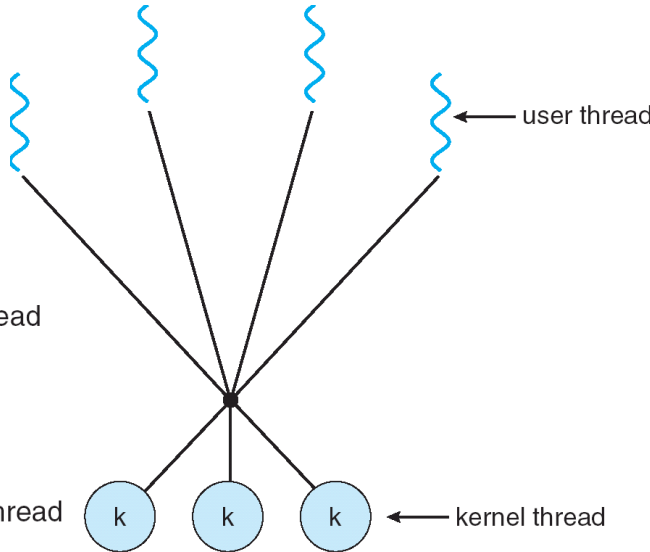
- User threads: e.g., pthread library
  - each process schedules its own threads
  - no context switch between these threads
  - a blocking call blocks the entire process
- Kernel threads: in almost all modern OS
  - kernel manages all threads
  - can pickup another thread if one blocks
- Hybrid approaches

Thread models

- User-kernel mapping
  - many-to-one: low cost, (lower) parallelism
  - one-to-one: high parallelism, (higher) cost
  - many-to-many: limited kernel threads

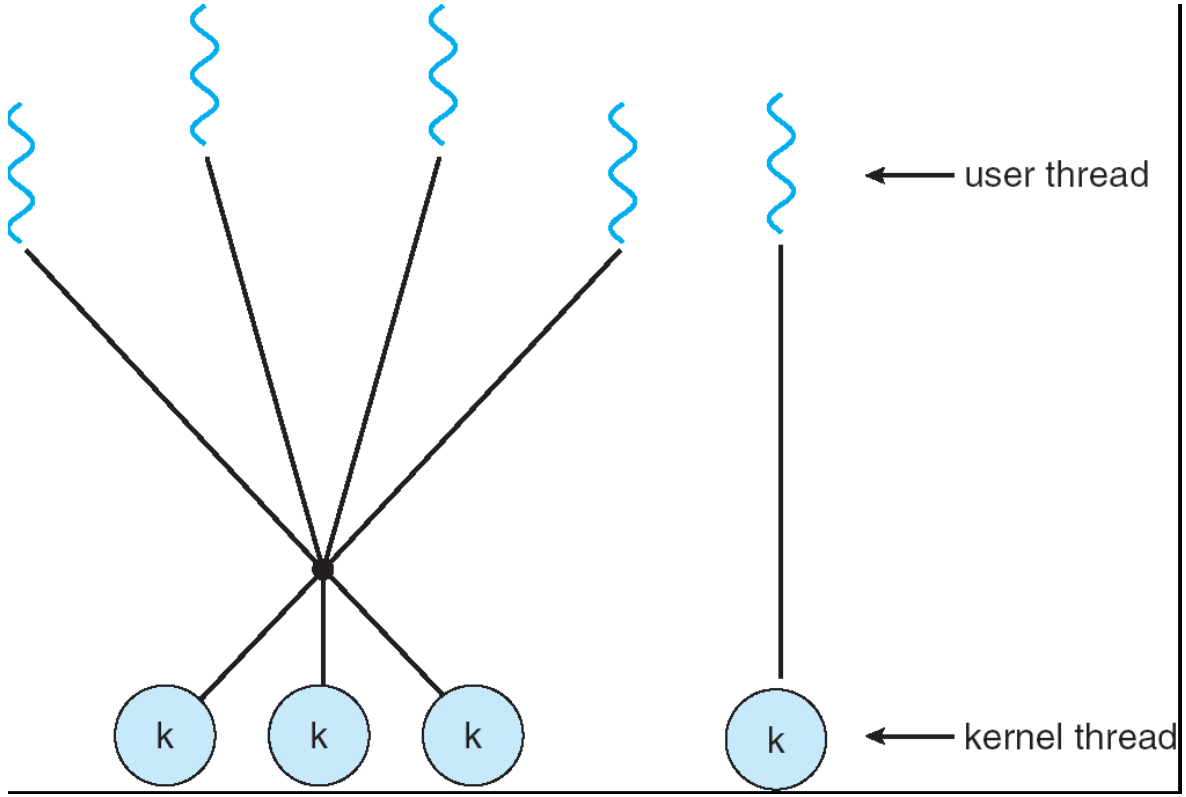






Thread models: more

- Two-level model



Threading issues

- When a new process is created
  - fork(), and usually then exec()
    - duplicate all threads or just the calling thread?
- When a signal to be delivered
  - signal: event notification to be handled
    - to all, some, or a specific thread?
- Thread pool
  - keep a pool of threads to be used
    - and reused

This lecture

- Thread
  - a basic unit of CPU utilization
  - user vs kernel-level threads
    - thread models
  - issues with threading (and more later)
- Explore further
  - thread support in your favorite OS
    - user vs kernel, thread model?

Next lecture

- Pthread
  - read OSC7 Chapter 4 (or OSC6 Chapter 5)
  - Pthread tutorial
    http://www.llnl.gov/computing/tutorials/pthreads/