Introduction II

Dmitri Loguinov
Texas A&M University

January 26, 2024
Multi-Threading

• Quiz next time (entire class)
  – My programming tutorial (pointers, bits ops, debugging, Windows datatypes)

• Threads execute concurrently as part of a process

• Benefits:
  – Allows for parallelism in a multiprocessor/multicore system
  – If a blocking call is made in one thread, other threads can continue executing

• Issues:
  – Memory is shared between threads, concurrent access requires proper synchronization
  – Order of execution of threads is non-deterministic
Multi-Threading 2

• Reasons for using multiple threads in hw #1
  – Web servers respond slowly (1-10 seconds/request)
  – While a thread is suspended waiting for connect() and recv(), other threads should be allowed to work

• Multiple threads achieve significant speed-up
  – You could run thousands of threads, but limit your testing to ~10 until you know it works correctly

• Common synchronization mechanisms
  – **Mutex** (mutual exclusion): allows only one thread access to critical section; others must wait
  – **Semaphore**: allows up to N concurrent threads
  – **Event**: binary (i.e., ON or OFF) signal
Multi-Threading 3

- Mutex usage
  - Any data structure (e.g., queue) or resource (e.g., screen or disk) modified by parallel threads needs to be protected
  - If not, inconsistencies (data corruption) may result

```c
CRITICAL_SECTION cs;
InitializeCriticalSection (&cs);

EnterCriticalSection (&cs); // lock
// critical section here ...
LeaveCriticalSection (&cs); // unlock
```

- Events
  - CreateEvent, WaitForSingleObject, CloseHandle

- Homework note: pass shared parameters to threads using a dedicated class instead of using global variables (see 463-sample.zip on course site)
Multi-Threading

• A semaphore has a numerical value \( s \) attached to it

• Wait on semaphore (operation P)
  - If \( s = 0 \), the semaphore suspends the calling thread
  - If \( s > 0 \), the thread is allowed access and \( s \) is set to \( s-1 \)

• Release semaphore (operation V)
  - If threads are waiting, unblock one of them and run it
  - Otherwise, increment \( s = s + 1 \)

```c
HANDLE sema = CreateSemaphore (...);
DWORD ret = WaitForSingleObject(sema, INFINITE); // wait
if (ret != WAIT_OBJECT_0)
  // report error

// critical section...

if (ReleaseSemaphore (sema, ...) == FALSE) // release
  // report error
```
Chapter 1: Roadmap

1.1 What is the Internet?
1.2 Network edge
1.3 Network core
1.4 Network access and physical media
1.5 Internet structure and ISPs
1.6 Delay & loss in packet-switched networks
1.7 Protocol layers, service models
1.8 History
Internet: Network of Networks

- Roughly hierarchical
  - In the center: “tier-1” ISPs (e.g., Sprint, AT&T, Verizon), national/international coverage
  - Treat each other as equals, do not pay for upstream bandwidth
  - Form the backbone of the Internet

Tier-1 providers also interconnect at public network access points (NAPs)
“Tier-2” ISPs: smaller (often regional) ISPs
- Connect to one or more tier-1 ISPs, possibly other tier-2 ISPs

Tier-2 ISP pays tier-1 ISP for connectivity to rest of Internet

Tier-2 ISP is customer of tier-1 provider

Tier-2 ISPs also peer privately with each other, or interconnect at NAPs
Internet Structure: Network of Networks

- “Tier-3” ISPs and local ISPs
  - Last hop (“access”) network (closest to end systems)

Local and tier-3 ISPs are customers of higher tier ISPs connecting them to rest of Internet.
Internet Structure: Network of Networks

- A packet passes through many networks!