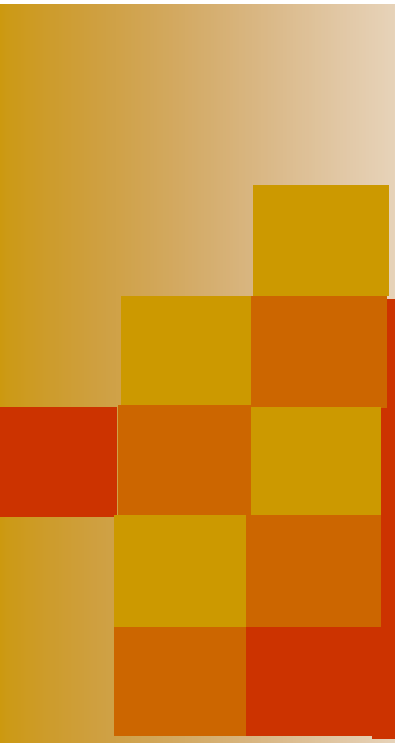


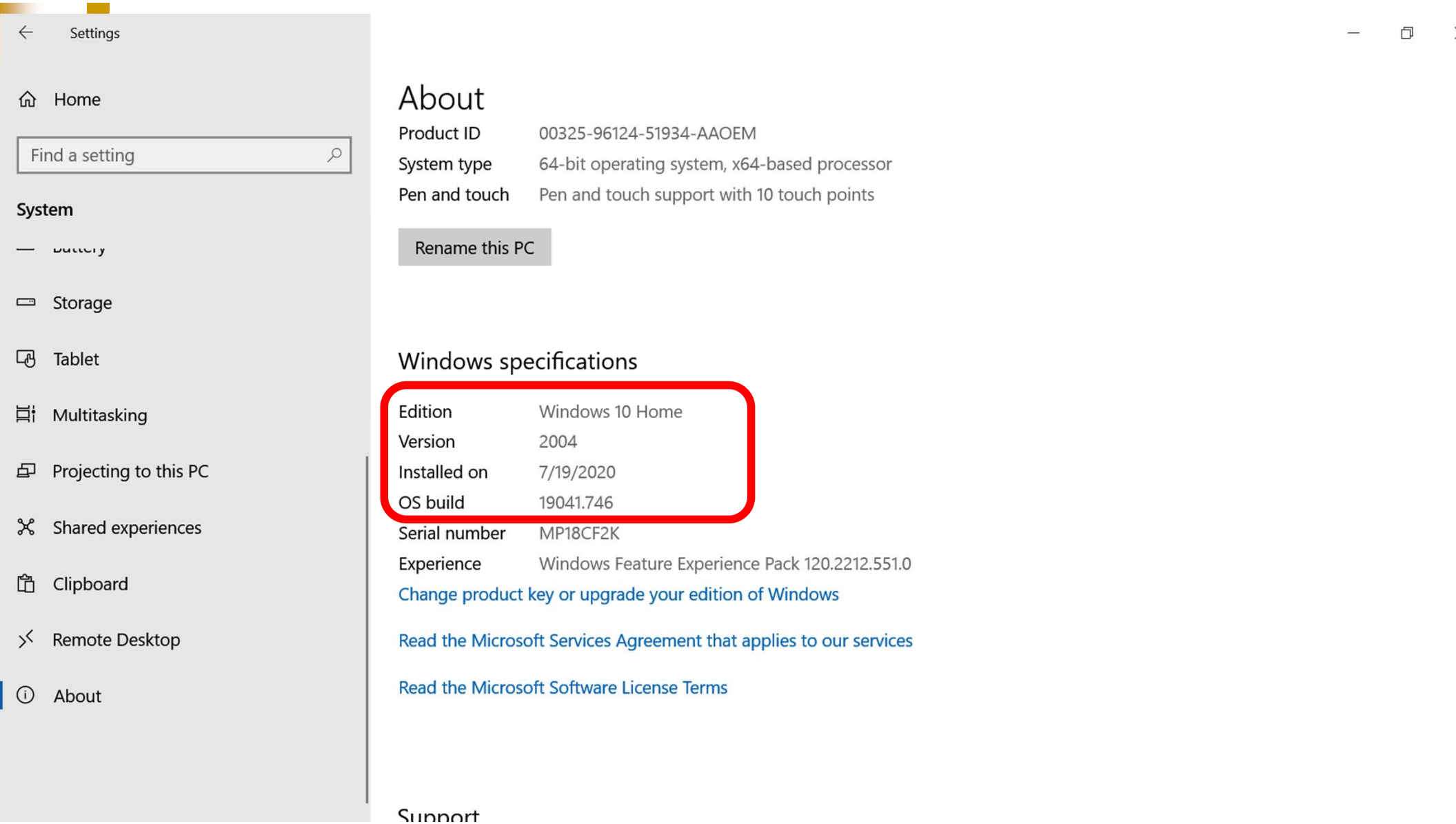
COSC 3360/6310
Monday, February 1

Announcements

- A ***sample input*** for the first assignment is now on MS Teams
  - **`input10.txt`**
    - Checked for correctness
- ***First quiz*** on ***February 8*** at ***4pm*** on ***Blackboard***
  - Will require Respondus lockdown browser and camera
  - Will cover everything that was presented up to and including Wednesday February 3 lecture.
  - Will post by Thursday afternoon a ***mock quiz***

Installing WSL on PC running Windows 10

- https://docs.microsoft.com/en-us/windows/wsl/install-win10
- Must
  - Have **Version 1903** or higher, with **Build 18362** or higher.
  - Know how to start PowerShell in administrative mode
    - Type PowerShell in search box at the left bottom corner of your screen
- Cut and paste the long PowerShell commands


Settings
Home
Find a setting
System
Storage
Tablet
Multitasking
Projecting to this PC
Shared experiences
Clipboard
Remote Desktop
About
About
Product ID 00325-96124-51934-AAOEM
System type 64-bit operating system, x64-based processor
Pen and touch Pen and touch support with 10 touch points
Rename this PC
Windows specifications
Edition Windows 10 Home
Version 2004
Installed on 7/19/2020
OS build 19041.746
Serial number MP18CF2K
Experience Windows Feature Experience Pack 120.2212.551.0
Change product key or upgrade your edition of Windows
Read the Microsoft Services Agreement that applies to our services
Read the Microsoft Software License Terms
Support

The four missions (continued)

Functions of an OS

- ***Four*** basic functions
  - To provide a better user interface
  - To manage the system resources
  - To protect users' programs and data
  - To let programs exchange information

Protecting users' data (I)

- Unless we have an isolated single-user system, we must prevent users from
  - Accessing
  - Deleting
  - Modifying

  without authorization other people's programs and data

Protecting users' data (II)

- Two aspects
  - Protecting user's files on disk
  - Preventing programs from interfering with each other
- Two solutions
  - Dual-mode CPUs
  - Memory protection

Historical Considerations

- Earlier operating systems for personal computers did not have any protection
  - They were single-user machines
  - They typically ran one program at a time
- Windows 2000, Windows XP, Vista and MacOS X are protected

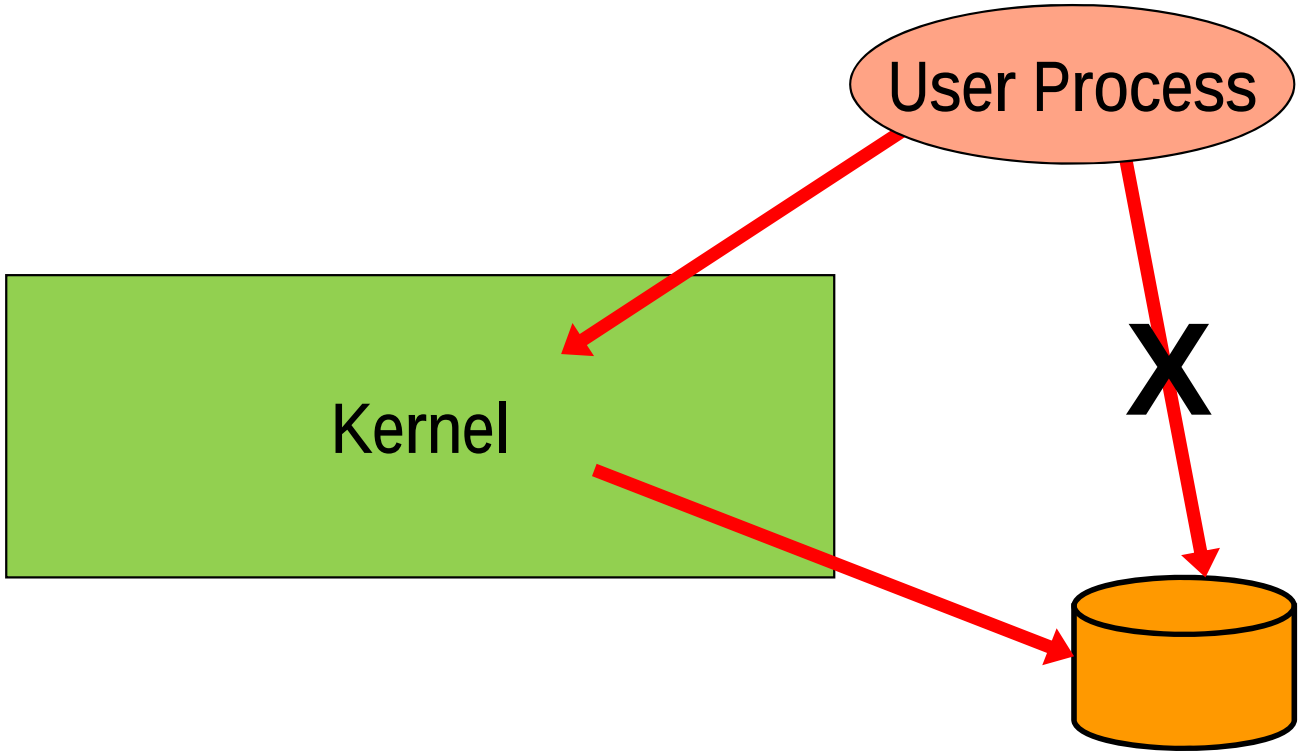
Protecting users' files

- Key idea is to prevent users' programs from directly accessing the disk
- Will require I/O operations to be performed by the kernel
- Make them ***privileged instructions***
  - Only the kernel can execute

Privileged instructions

- Require a ***dual-mode CPU***
- Two CPU modes
  - ***Privileged mode*** or ***executive mode***
    - Allows CPU to execute all instructions
  - ***User mode***
    - Allows CPU to execute only safe unprivileged instructions
- State of CPU is determined by a ***special bit***

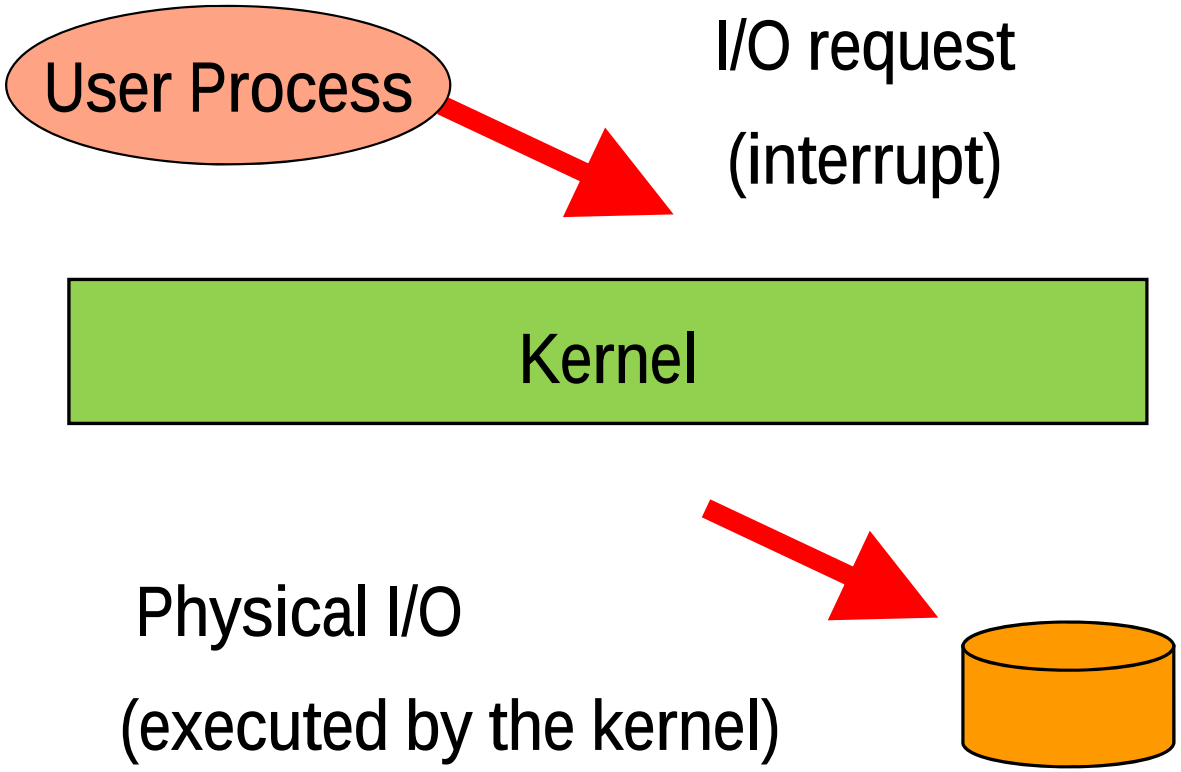
All disk/SSD accesses must go through the kernel



Switching between states

- User mode will be the default mode for all programs
  - Only the kernel can run in supervisor mode
- Switching from user mode to supervisor mode is done through an interrupt
  - Safe because the jump address is at a well-defined location in main memory

Performing an I/O



An analogy (I)

- Most UH libraries are ***open stacks***
  - Anyone can consult books in the stacks and bring them to checkout
- National libraries and the Library of Congress have ***closed stack collections***
  - Users fill a request for a specific document
  - A librarian will bring the document to the circulation desk

An analogy (II)

- ***Open stack collections***
  - Let users browse the collections
  - Users can misplace or vandalize books
- ***Closed stack collections***
  - Much slower access
  - Much safer

More trouble

- Having a dual-mode CPU is ***not enough*** to protect user's files
- Must also prevent rogue users from tampering with the kernel
  - Same as a rogue customer bribing a librarian in order to steal books
- Done through ***memory protection***

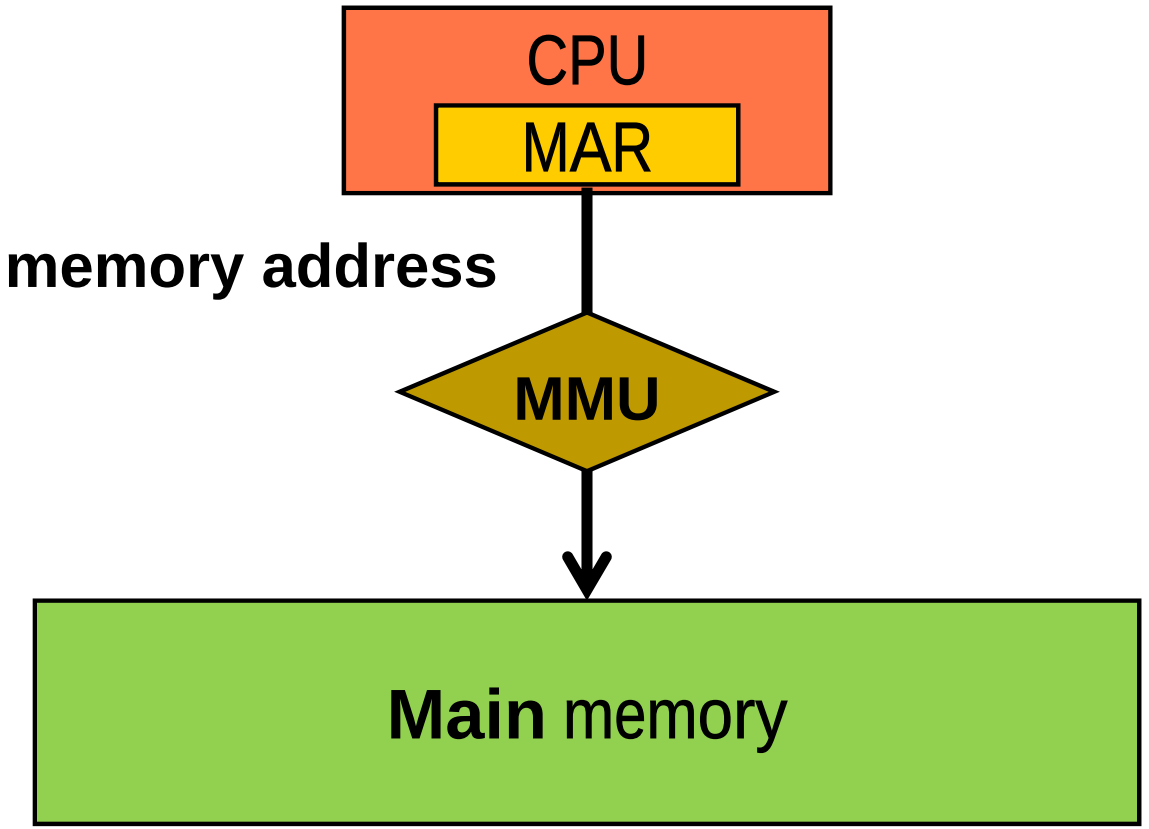
Memory protection (I)

- Prevents programs from accessing any memory location outside their own ***address space***
- Requires special ***memory protection hardware***
  - ***Memory Management Unit*** **(MMU)**
- Memory protection hardware
  - Checks ***every*** reference issued by program
  - Generates an interrupt when it detects a protection violation

Memory protection (II)

- Has additional advantages:
  - Prevents programs from corrupting address spaces of other programs
  - Prevents programs from crashing the kernel
    - Not true for device drivers which are inside the kernel
- Required part of any multiprogramming system

Memory protection (III)



Even more trouble

- Having both a dual-mode CPU and memory protection is not enough to protect user's files
- Must also prevent rogue users from booting the system with a ***doctored kernel***
  - ***Example:***
    - Can run Linux from a "live" CD Linux
    - Linux will read all NTFS files ignoring all restrictions set up by Windows

Inter-process communication

- Has become very important over the last thirty years
- Two techniques
  - Message passing
    - General but not very easy to use
  - Shared memory
    - Less general, easier to use but requires inter-process synchronization

ANOTHER VIEW

- Arpaci-Dusseau & Arpaci-Dusseau
  - Focus on services provided by OSes
- Three themes
  - Virtualization
  - Concurrency
  - Persistence

Virtualization

- The process abstraction
- Virtualizing the CPU:
  - Process scheduling
- Virtualizing the memory:
  - Memory management

Concurrency

- Threads
- Locks
- Semaphores

We will cover threads in the chapter on processes because they are essential to the client server model

Persistence

- The file system

Types of operating systems

Types of operating systems

- Already discussed:
  - Batch systems
  - Time-Sharing systems
- Will now introduce
  - Real-Time systems
  - Operating systems for multiprocessors
  - Distributed systems

Real-time systems

- Designed for applications with ***strict real-time constraints*** :
  - ***Process control***
  - ***Guidance systems***
  - Most ***multimedia applications***
- Must guarantee that critical tasks will ***always*** be performed within a specific time frame.

Hard RT systems

- Must guarantee that all deadlines will always be met
- ***Any failure*** could have ***catastrophic consequences***:
  - The reactor could overheat and explode
  - The rocket could be lost

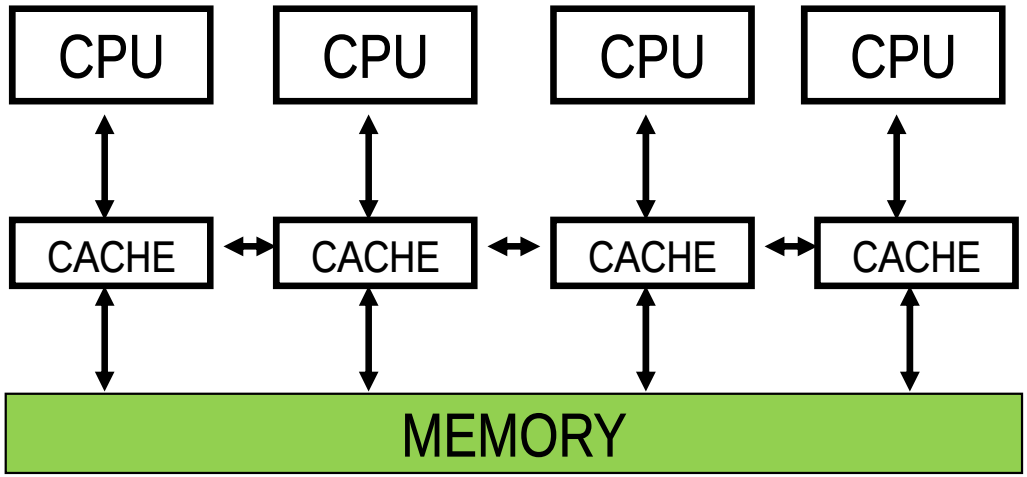
Soft RT systems

- Guarantee that most deadlines will be met
- A DVD decoder that miss a deadline will spoil our viewing pleasure for a fraction of a second

Observations

- Hard RT applications normally run on special RT OSes
- Soft RT applications can run on a regular OS
  - If the OS supports them
- Interactive and time-sharing systems are ***not*** RT systems
  - They attempt to provide a fast response time but do not try to meet specific deadlines

Multiprocessor operating systems



- Designed for ***multiprocessor architectures***
  - Several processors share the same memory

Leader/follower multiprocessing

- Single copy of OS runs on a dedicated core/processor
  - ***Leader*** or *master* (deprecated)
- Other cores/processors can only run applications
  - ***Followers*** or *slaves* (deprecated)
- Major advantage is ***simplicity***
  - Requires few changes
- Major disadvantage is ***lack of scalability***
  - Single copy of OS can become a ***bottleneck***

Symmetric multiprocessing

- Any core/processor can perform all functions
  - There can be multiple copies of the OS running in parallel
- Must prevent them from interfering with each other
  - Disabling interrupts will not work
  - Must add ***locks*** to all critical sections

The state of the art

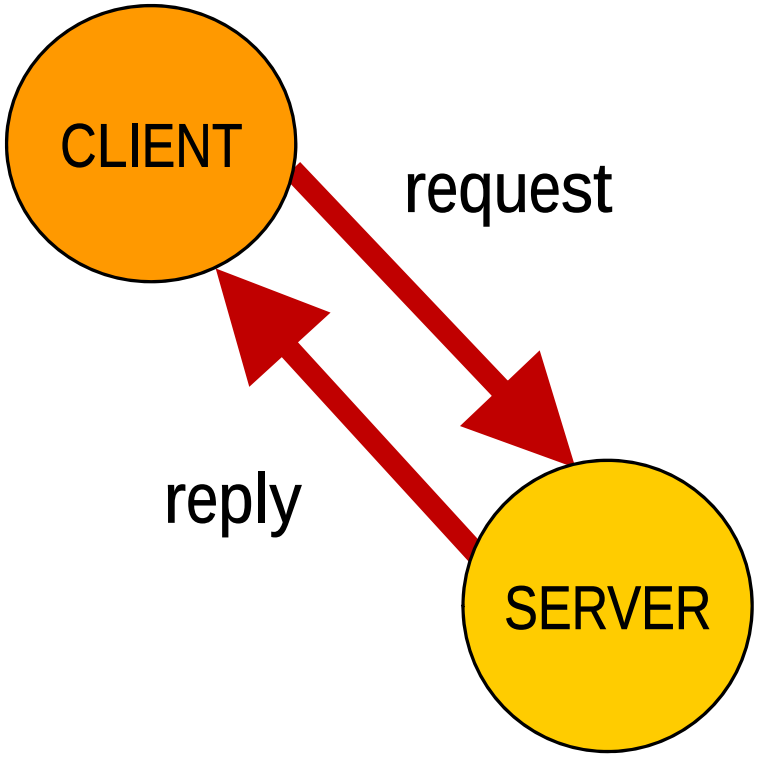
- Most computers now have ***multicore CPUs***
  - Sole practical way to increase CPU power
- Many have powerful GPUs
  - Highly parallel
- Using multicore architectures in an effective way is a huge challenge

Distributed systems

- Integrated networks of computers
  - ***Workstations*** sharing common resources (file servers, printers, …)
- Current trend is to leave systems very loosely coupled
  - Each computer has its own OS

Client /Server Model

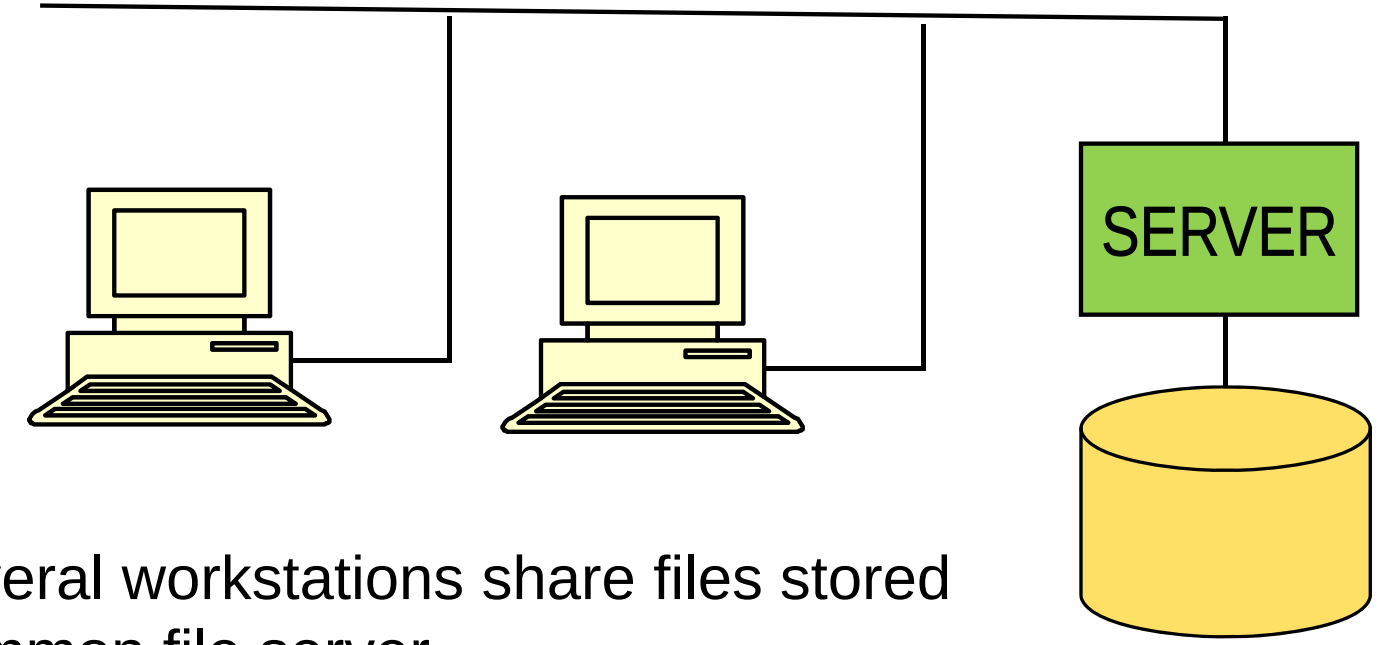
- Servers wait for requests from clients and process them
  - File servers
  - Print servers
  - Authentication servers



A typical sequential server

```
for (;;){
   //wait for request
   get_request(…);
   // process it
   process_request(…);
   // send reply
   send_reply(…);
} // forever
```

Network file system

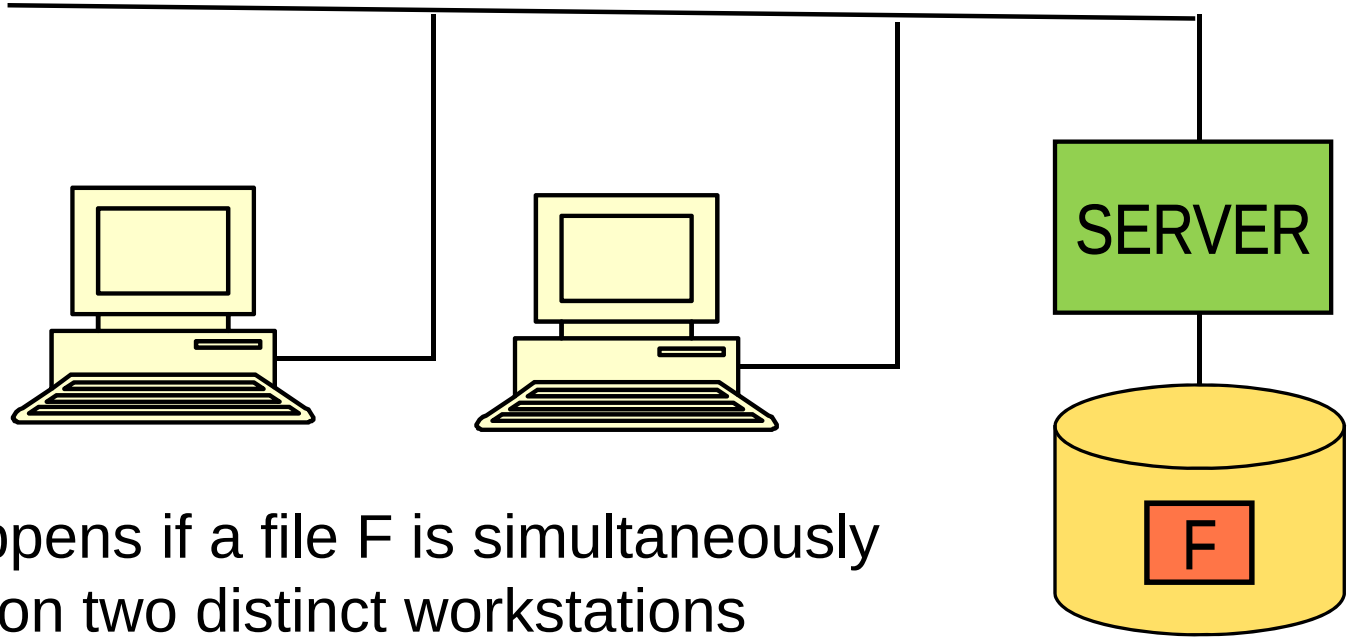


- Lets several workstations share files stored on a common file server

Performance Issues

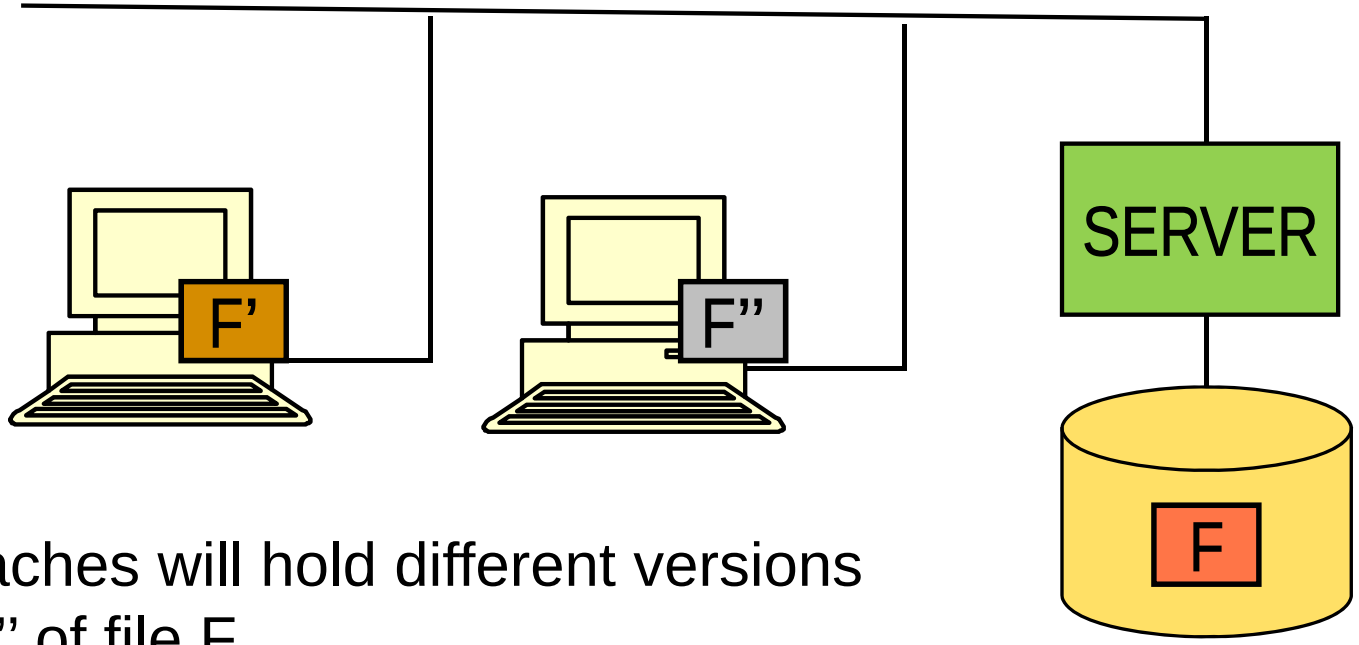
- ***Response time*** is the main issue
  - ***Network latency*** is now added to ***disk latency***
- Will attempt to mask these two latencies
  - Extensive ***client caching***
  - ***Works very well***

File consistency issues (I)



- What happens if a file F is simultaneously modified on two distinct workstations

File consistency issues (II)



- Client caches will hold different versions F' and F'' of file F

File Consistency Issues (III)

- Maintaining file consistency is a very important issue in distributed/networked file system design
- Different systems use different approaches
  - NFS from Sun Microsystems
  - AFS/Coda from CMU
  - …

Other distributed systems issues

- **Authenticating** users
  - *A problem in open networks*
- Making distributed systems as *reliable* as stand-alone systems
  - ***Replication*** of data and services
- Keeping the clocks of the machines more or less synchronized.