Introduction to Operating Systems at OU

CS 3113, Fall 2019

** Includes slide material from Silberschatz, Galvin and Gagne (2018)
What is an Operating System?
What is an Operating System?

A program that acts as an intermediary between a user of a computer and the computer hardware

…a User can really be a person, an application program or another computer
What are the Goals of an OS?
What are the Goals of an OS?

Operating system goals:
• Execute user programs and make solving user problems easier
• Make the computer system convenient to use
• Use the computer hardware in an efficient and secure manner
Computer System Structure

Computer system can be divided into four components:

- **Hardware** – provides basic resources
  - CPU, memory, I/O devices

- **Operating system**
  - Controls and coordinates use of hardware among various applications and users

- **Application programs** – define the ways in which the system resources are used to solve computing problems
  - Word processors, compilers, web browsers, database systems, video games

- **Users**
  - People, other computers
Computer System Structure

user 1

user 2

user 3

... user n

compiler

assembler

text editor

... system and application programs
database system

operating system

computer hardware
What Operating Systems Do?
What Operating Systems Do?

It depends …

• Users want convenience, ease of use, and good performance
• Individual users don’t necessarily care about resource utilization
• A shared computer such as mainframe or minicomputer must keep all users happy
• Users of dedicate systems, such as workstations, have dedicated resources, but frequently use shared resources from servers
• Handheld computers are resource poor, and are optimized for usability and battery life
• Some computers have little or no user interface, such as embedded computers in devices and automobiles
Operating System Definition

- OS is a resource allocator
  - Manages all hardware resources
  - Decides between conflicting requests for efficient and fair resource use

- OS is a control program
  - Controls execution of programs to prevent errors and improper use of the computer

- OS provides abstractions
  - Hides the details of the hardware
  - Provides an interface that allows a consistent experience for application programs and users
Operating System Definition

What are common abstractions provided by the OS?
Operating System Definition

What are common abstractions provided by the OS?

• A program has exclusive access to the CPU(s) and other hardware devices
• A program has unbounded access to memory
• Directories and files
• Reliable communication between programs and computers
• No errors in: execution, communication, device interaction
Example: Hard Disks

What are they and what do they provide?
Structure of a Hard Disk

A = Track (Red)
B = Sector (Slice)
C = Sector Track
D = Cluster
Structure of a Hard Disk

• A sector of the disk is a sequence of bits
  • Encoded by orientation of localized magnetic fields
• Disk controller provides sector read/write operations
• Disk design varies
  • Sector size
  • Number of platters
  • Speed of spindle rotation
  • Protocol for the computer talking to the disk
• Job of the device driver: handle low level abstractions
• The OS then lays a file system on top
Files

How is a file stored on a disk?
Files

How is a file stored on a disk?

• Depending on size, a file is allocated some number of sectors (blocks)

• While data within a block is contiguous, the different blocks that are used to represent a file are not necessarily contiguous

• Need some way of tracking which blocks belong to a file and what their order is (e.g., inodes)
A File System

Allows us to:

• Represent a set of files
• Organize these files in a useful way
  • Most common today: directory trees

Modern systems also enable:

• FS distributed across multiple disks and multiple computers
• Redundancy and automatic recovery
Example: Random Access Memory

What is RAM and how is it organized?
Example: Random Access Memory

What is RAM and how is it organized?

- Data stored in a sequence of bytes
- Each byte has an address: 0, 1, 2, ...
- **Random**: can access any byte as fast as any other byte
RAM for Processes

• Process: executing program
• What does memory look like from the perspective of a process?
RAM for Processes

What does memory look like from the perspective of a process?

• Also addressable

• An individual variable is assigned to “live” in one or more bytes

• Overlay data structures (with some hardware support):
  • Stack: storage for locally defined variables
  • Heap: storage for dynamically allocated variables
OS Role in Memory

- Providing processes with the memory that they need as they are executing
  - A process doesn’t usually need all of its memory available all of the time.
  - We can take advantage of this!
- Allowing multiple processes to co-exist safely
  - Usually, each has exclusive access to its own memory
  - But: memory can be shared across processes, if needed
- Providing buffering for I/O activities
Your Background

• Programming
  • Control structures & primitive data types
  • Substantial design and debugging experience
  • Building abstractions

• Algorithms and Theory
  • Building space and time efficient data structures

• Hardware
  • Computer Organization
Operating Systems

• The study of Operating Systems brings together your background in programming, algorithms, theory and hardware!

• And sets you up to study bigger things:
  • Databases
  • Communication networks
  • High performance computing
  • Embedded systems
CS 3113: Coverage

Mix of theory and practice:
• Systems-level programming in C
• *nix system calls
• File Systems: Properties and Implementation
• Processes and Threads: Pipes, Concurrency and Synchronization
• I/O and Process Scheduling
• Security
• Virtual Machines
Projects

The practice of OS requires real practice

• The projects are designed to exercise your algorithm and low-level programming skills
• Five 2-3 week projects over the semester
• You will need this time
• Projects are done individually
Project Procedures

• Programming, testing and debugging on a standard Linux instance running on your own laptop and/or desktop
  • Virtualbox to host the Linux instance
  • Practice using command line tools and available editor(s)

• Submission to Gradescope
  • Connected through Canvas
  • Automatic testing
  • We will provide coding feedback here, too
Mechanics
Our Assumptions About You

• Data Structures and Computer Organization
• C/C++
  • Control structures & primitive data types
  • Substantial design and debugging skill
• Computer system:
  • Laptop or equivalent for class
  • Laptop or desktop for outside work (note that we have Linux boxes in DEH 115)
Course Information

• Course web page: https://cs.ou.edu/~fagg/classes/cs3113
  • Includes schedule and syllabus
• Canvas: announcements, assignments, grading
Grading Distribution

• 5 Projects: 40%
• Exams: 30% (one midterm and a final)
• Homework: 15% (keep N-1 highest)
• In-class exercises: 15% (keep M-1 highest)

Official grades will be posted in Canvas
Homework Assignments

• Short-term (1-week)
• Mix of coding, algorithm simulation and short answers
Homework and In-Class Exercises

• Dropping the lowest of your grades is intended to mitigate unexpected situations, such as illness
• Even if you are late on an assignment, you should still take the time to complete it
  • They offer important experience that reinforces the lecture & readings and they prepare you for the exams
Projects

Detailed coding exercises that allow us to cover algorithms and data structures to very low level programming issues

• Projects will build on each other
• No project grades are dropped
Due Dates

• In-class exercises: due when asked for in class
• Homework and projects: due at 11:45pm on the date noted on the schedule
• Projects may be turned in late:
  • Up to 24 hours: 10% grade penalty
  • Up to 48 hours: 20% grade penalty
Grading Questions

• The graded assignment should be first brought to the person who graded it
• All grading questions must be brought to our attention within one week of when the item was returned
• Check your grades on Canvas
Honors College Students

This course can be taken for honors credit!
• You need to declare this week
• Faster pace on projects + additional project
Proper Academic Conduct

Discussion about any topic with the instructors and/or TAs is always fine

• We can also look at code!
Proper Academic Conduct

Coding assignments (projects and coding homework assignments):

• Discussion about solutions with classmates is allowed
• Looking at network resources is allowed
• You must document these discussions / resources (classmates and network)

• But: no looking at or copying code solutions for the assignments
Proper Academic Conduct

Collaborative in-class exercises:
• We expect you to interact in small groups
Proper Academic Conduct

Homework assignments, in-class exercises and exams:

• Unless otherwise specified: the work must be your own: no looking at or copying solutions from other students or from the net
Proper Academic Conduct

Code:

- Sharing solutions is penalized to the same degree as receiving solutions
- Make sure that your computer and account are properly protected. Use a secure password
- Do not give out access to your account or your computer system
- Do not leave printouts or mobile drives around a laboratory where others might access them
Proper Academic Conduct

• Programming projects will be checked by software designed to detect collaboration
• This software is extremely effective and has withstood repeated reviews by the campus judicial processes
Conduct Violations

• Upon the first documented occurrence of inappropriate collaboration, we will report the academic misconduct to the Campus Judicial Coordinator. The procedure to be followed is documented in the University of Oklahoma Academic Integrity Code

• The appeals process for both admonitions and full complaints is described at:
  • [http://integrity.ou.edu/](http://integrity.ou.edu/)
Laptop Policy

In Class:
• May be used to program along with the lecture or to take notes
• In-class assignments will require them

If you are using your laptop in a way that distracts people around you in class, you will be asked to leave.
Time Commitment

• You will be developing software in this class
  • Time spent $\propto$ grade
  • Start early. You don’t know how long it will take

• Plan before coding

• Write your own tests
  • Many of our tests will be hidden from you!

• Should rethink taking this class while taking another heavy class
Getting the Most out of Class

• Read materials ahead of time
• Ask questions
• Learn names of your fellow students (and use them)
• Participate in class discussions
• Attend class
• At the end of the semester, we should know your name
A Bit of Neuroscience …

• Your brain integrates information and problem solves over time

• Cramming assignments and studying into a very small number of sessions (especially when up against a deadline) works against this

• Instead, plan to block out time to work a little bit on the reading and the assignments every day or every other day

… Your brain will thank you
Appropriate Classroom Conduct

Key rule: **Respect**

- Yourself
- Your peers
- The teaching team
  - Keep in mind: we are human, too, and we have many obligations
Teaching Team

There are 98 of you and only three members of the teaching team

• Between the three of us, we only have 40 paid hours per week for this course

• We have to be efficient about the use of our time
  • Many aspects of the course are automated (e.g., grading projects, homework and exams) so we can maximize our time with you
  • Please take steps to help us with this
How to Find Me

Dr. Andrew H. Fagg:  DEH 243  andrewhfagg@gmail

• Office hours are still to be announced
• Appointments can also be made

• The TAs and I can be reached simultaneously: cs3113@googlegroups.com
How to find the TAs

All TA office hours in DEH 115 (computer lab)
  Dorian Selimovic: Dorian.Selimovic-1@ou.edu
  Gregory Maddra: Gregory.J.Maddra-1@ou.edu

William Kerber Teaching Scholars: DEH 115
This and Next Week…

Reading and next classes: see the schedule!
• High-level view of OS
• OS Internals
• C Programming and Linux
COMPUTER HOLY WARS

HOLD IT RIGHT THERE, BUDDY.

THAT SCRUFFY BEARD... THOSE SUSPENDERS... THAT SMUG EXPRESSION...

YOU'RE ONE OF THOSE CONDESCENDING UNIX COMPUTER USERS!

HERE'S A NICKEL, KID. GET YOURSELF A BETTER... OS
Computer System Organization

Common bus structure:

• One or more CPUs, device controllers connect through common bus providing access to shared memory
• Concurrent execution of CPUs and devices competing for memory cycles
I/O devices and the CPU execute concurrently
• Each device controller is in charge of a particular device type
• Data sent to or received from the device are stored in a local buffer
• CPU moves data between these local buffers and main memory
• When a device controller completes an I/O operation, it informs the CPU by causing an interrupt
Interrupts

An operating system is **interrupt driven**

- An interrupt transfers control from the currently executing program to the appropriate interrupt service routine
- Interrupt architecture must save the address of the interrupted instruction, as well as the register state
- A **trap** or **exception** is a software-generated interrupt caused either by an error or a user request
Interrupt Timeline

- CPU: user process executing
- I/O device: idle, transferring
- I/O interrupt processing

Timeline:
- I/O request
- Transfer done
- I/O request
- Transfer done
I/O Structure

• User program does not have direct access to the devices (it is prevented explicitly!)

• Instead, a request for access is made to the OS through the use of a system call
  • Special function that is able to access the kernel-level data structures and I/O system

• After I/O starts, control returns to user program without waiting for I/O completion
Storage Definitions

• Bit: contains a value of 0 or 1
• Byte: 8-bits. Fundamental unit of memory
• Word: multiple bytes (system dependent)
  • In modern laptops: 8 bytes
• $2^{10}$ bytes: kilobyte
• $2^{20}$ bytes: megabyte
• $2^{30}$ bytes: gigabyte
• $2^{40}$ bytes: terabyte
Storage Types (some)

• Main memory – only large storage media that the CPU can access directly
  • Random access, typically volatile

• Secondary storage – extension of main memory that provides large nonvolatile storage capacity
  • Hard disks – rigid metal or glass platters covered with magnetic recording material
  • Disk surface is logically divided into tracks, which are subdivided into sectors

• Solid-state disks – faster than hard disks, nonvolatile
  • Various technologies. Expensive relative to hard disks
Storage-Device Hierarchy
Storage Hierarchy

- Storage systems organized in hierarchy. Each level involves trade-offs:
  - Speed
  - Cost
  - Volatility

- **Caching** – copying information into faster storage system
  - Main memory can be viewed as a cache for secondary storage
Caching

Information in use copied from slower to faster storage temporarily

• Important principle, performed at many levels in a computer (in hardware, operating system, software)
• Faster storage (cache) checked first to determine if information is there
  • If it is, information used directly from the cache (fast)
  • If not, data copied to cache and used from there
• Cache management is an important design choice
  • Including: cache size and replacement policy
Direct Memory Access

• Used for high-speed I/O devices able to transmit information at close to memory speeds
• Device controller transfers blocks of data from buffer storage directly to main memory without CPU intervention
• Only one interrupt is generated per block, rather than the one interrupt per byte
Data Flow in a Modern Computer

- A thread of execution
- A cache
- CPU (*N)
- I/O request
- Data
- Interrupt
- DMA
- Memory
- Instructions and data
- Data movement
- Instruction execution cycle
- Device (*M)
Computer-System Architecture

• 15 years ago: most systems used a single general-purpose processor
  • Most systems (even today) also have special-purpose processors
• Multiprocessors systems have grown in use and importance
  • Also known as parallel systems, tightly-coupled systems
  • Advantages include:
    • Increased throughput
    • Economy of scale
    • Increased reliability – graceful degradation or fault tolerance
• Two types:
  • Asymmetric Multiprocessing – each processor is assigned a specific task.
  • Symmetric Multiprocessing – each processor performs all tasks
Multiprocessing Architectures

Symmetric Multiprocessor: loosely coupled, multiple chips

Multi-Core Processors: tightly coupled, single chip
Clusters

Cluster: large number of coordinated computers

• Programs can execute in parallel across multiple computers
  • Number of computers can scale with demand
  • High Performance Computing (HPC) clusters: 1000s of nodes

• A single computer can potentially be used by many programs
  • More efficient use of hardware

• Provides redundancy in the face of hardware failure

Andrew H. Fagg: CS3113
Making Efficient Use of a CPU

• **Multiprogramming:**
  • Switch between programs as CPU becomes idle (e.g., if a process is waiting for I/O)
  • Scheduling processes is relatively straight-forward

• **Multitasking:**
  • Switch quickly between processes automatically
  • Allows processes to appear like they are responding in real time (at least to a user)
  • Scheduling processes and their memory use is a challenge
Protection with Processor Modes

• Dual-mode operation allows OS to protect itself and other system components
  • User mode and kernel mode
  • Mode bit provided by hardware
  • Provides ability to distinguish when system is running user code or kernel code
  • Some instructions designated as privileged, only executable in kernel mode
  • System call changes mode to kernel, return from call resets it to user

• Increasingly CPUs support multi-mode operations
  • For example: virtual machine manager (VMM) mode for guest VMs
System Calls

System calls allow a user program to request services from the kernel
- Including I/O and process management services
Processes

• A process is a program in execution. It is a unit of work within the system. Program is a passive entity, process is an active entity.

• Process needs resources to accomplish its task, including: CPU, memory, I/O, files

• Process termination requires OS to reclaim of any reusable resources

• Single-threaded process has one program counter specifying location of next instruction to execute
  • Process executes instructions sequentially, one at a time, until completion

• Multi-threaded process has one program counter per thread

• Typically system has many processes, some user, some operating system running concurrently on one or more CPUs

Andrew H. Fagg: CS3113
Process Management Activities

The operating system is responsible for the following activities in connection with process management:

• Creating and deleting both user and system processes
• Suspending and resuming processes
• Providing mechanisms for process synchronization
• Providing mechanisms for process communication
• Providing mechanisms for deadlock handling
Storage Management

• OS provides uniform, logical view of information storage
  • Abstracts physical properties to logical storage unit
  • These physical properties include: access speed, capacity, data-transfer rate, access method (sequential or random)

• File-System management
  • Files usually organized into directories
  • Access control on most systems to determine who can access what
  • OS activities include
    • Creating and deleting files and directories
    • Primitives to manipulate files and directories
    • Mapping files onto secondary storage
    • Backup files onto stable (non-volatile) storage media
# Performance of Various Levels of Storage

<table>
<thead>
<tr>
<th>Level</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>registers</td>
<td>cache</td>
<td>main memory</td>
<td>solid state disk</td>
<td>magnetic disk</td>
</tr>
<tr>
<td>Typical size</td>
<td>&lt; 1 KB</td>
<td>&lt; 16MB</td>
<td>&lt; 64GB</td>
<td>&lt; 1 TB</td>
<td>&lt; 10 TB</td>
</tr>
<tr>
<td>Implementation technology</td>
<td>custom memory with multiple ports CMOS</td>
<td>on-chip or off-chip CMOS SRAM</td>
<td>CMOS SRAM</td>
<td>flash memory</td>
<td>magnetic disk</td>
</tr>
<tr>
<td>Access time (ns)</td>
<td>0.25 - 0.5</td>
<td>0.5 - 25</td>
<td>80 - 250</td>
<td>25,000 - 50,000</td>
<td>5,000,000</td>
</tr>
<tr>
<td>Bandwidth (MB/sec)</td>
<td>20,000 - 100,000</td>
<td>5,000 - 10,000</td>
<td>1,000 - 5,000</td>
<td>500</td>
<td>20 - 150</td>
</tr>
<tr>
<td>Managed by</td>
<td>compiler</td>
<td>hardware</td>
<td>operating system</td>
<td>operating system</td>
<td>operating system</td>
</tr>
<tr>
<td>Backed by</td>
<td>cache</td>
<td>main memory</td>
<td>disk</td>
<td>disk</td>
<td>disk or tape</td>
</tr>
</tbody>
</table>