CS 3502
Operating Systems

Introduction

Kun Suo

Computer Science, Kennesaw State University

https://kevinsuo.github.io/

Outline

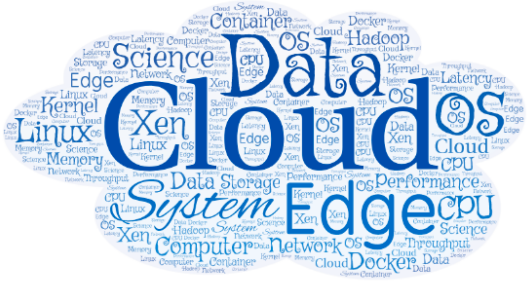
- Introduction & Basics
- Why study Operating Systems ?
- What to learn ?
- Course structure
- Course policy
- Course goals

Self Introduction

- Kun Suo, Ph.D.
  - Homepage, https://kevinsuo.github.io/

- Research interests:
  - Cloud computing and virtualization;
  - Operating systems, containers and kubernetes;
  - Software defined network (SDN) and network function virtualization (NFV)
  - Big data systems and machine learning systems

- Projects you may be interested in:
  - Several projects in Cloud & Data & Edge
  - https://kevinsuo.github.io/code-lab.html

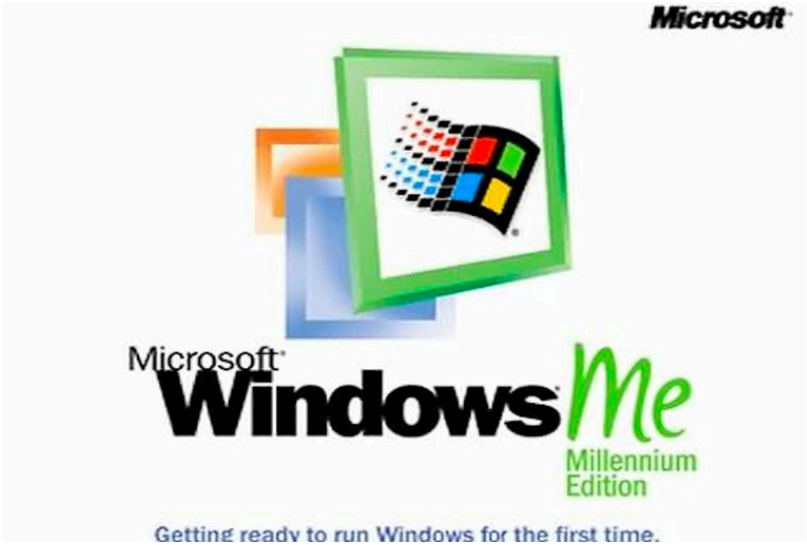


Now it's your turn

- Name, program/year, where from
- Your interests in Computer Science https://www2.eecs.berkeley.edu/Research/Areas/CS/
- What is the first OS your ever used? Current OS using? How many OSes you ever used (name them)?

If you are in the online course, introduce yourself in D2L, Discussions → Self-Introduction

Now it's your turn



Now it's your turn

Now it's your turn

Now it's your turn

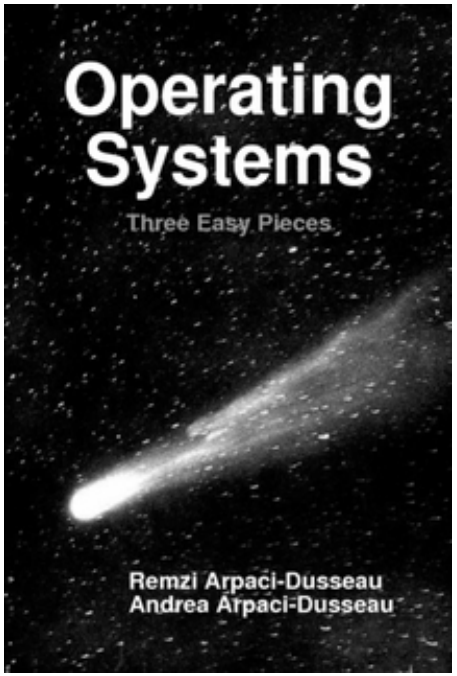
Now it's your turn

Course Information

- Instructor: Dr. Kun Suo
- Office: J-318
- Email: ksuo@kennesaw.edu
  - Only reply to e-mails that are sent from KSU student email accounts and title the course number [CS3502]
- Office Hours:
  - T/Th, 3pm-4pm
  - By appointment
- Course Materials
  - Homework assignments, lecture slides, and other materials will be posted in the webpage (https://kevinsuo.github.io/teaching.html) and D2L.

Reference Book

- "Operating Systems: Three Easy Pieces" by Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau:



  - Three pieces: virtualization, concurrency and persistence.
  - Free! (Separate PDFs for different chapters at http://pages.cs.wisc.edu/~remzi/OSTEP/
  - Hard copy option and single-PDF option are available for a fee.

Reference Book

Intro	Virtualization		Concurrency	Persistence	Appendices
Preface	3 *Dialogue*	12 *Dialogue*	25 *Dialogue*	35 *Dialogue*	*Dialogue*
TOC	4 Processes	13 Address Spaces code	26 Concurrency and Threads code	36 I/O Devices	Virtual Machines
1 *Dialogue*	5 Process API code	14 Memory API	27 Thread API code	37 Hard Disk Drives	*Dialogue*
2 Introduction code	6 Direct Execution	15 Address Translation	28 Locks code	38 Redundant Disk Arrays (RAID)	Monitors
	7 CPU Scheduling	16 Segmentation	29 Locked Data Structures	39 Files and Directories	*Dialogue*
	8 Multi-level Feedback	17 Free Space Management	30 Condition Variables code	40 File System Implementation	Lab Tutorial
	9 Lottery Scheduling code	18 Introduction to Paging	31 Semaphores code	41 Fast File System (FFS)	Systems Labs
	10 Multi-CPU Scheduling	19 Translation Lookaside Buffers	32 Concurrency Bugs	42 FSCK and Journaling	xv6 Labs
	11 *Summary*	20 Advanced Page Tables	33 Event-based Concurrency	43 Log-structured File System (LFS)	
		21 Swapping: Mechanisms	34 *Summary*	44 Flash-based SSDs	
		22 Swapping: Policies		45 Data Integrity and Protection	
		23 Complete VM Systems		46 *Summary*	
		24 *Summary*		47 *Dialogue*	
				48 Distributed Systems	
				49 Network File System (NFS)	
				50 Andrew File System (AFS)	
				51 *Summary*	

Prerequisites

- Computer basics that are supposed to covered in *(CS 3305) Data Structures* and *(CS 3503) Computer Organization and Architecture* course.

- C programming (code reading, kernel development and debugging). (Famous projects in C)

- Linux command line environment (compiling, Makefile, debugging, simple shell programming).

For C and Linux beginners

- C tutorial
  - https://www.tutorialspoint.com/cprogramming/
  - https://www.learn-c.org
  - https://www.cprogramming.com/tutorial/c-tutorial.html

- Linux tutorial
  - https://ryanstutorials.net/linuxtutorial/
  - http://www.ee.surrey.ac.uk/Teaching/Unix/
  - https://www.tutorialspoint.com/unix/

Project Environment

- Recommend project environment
  - VirtualBox + Ubuntu + Linux 5.0

Virtual machine

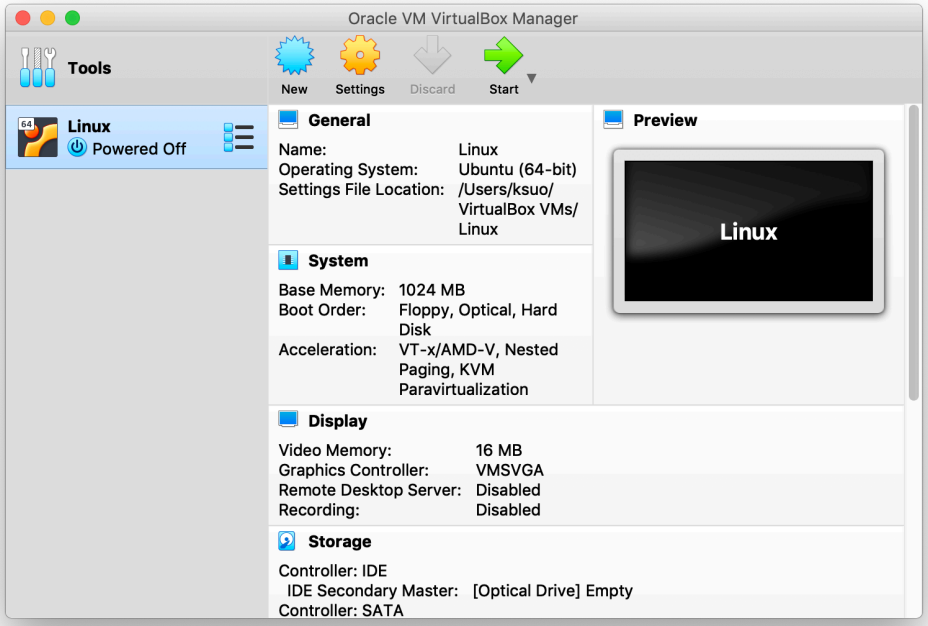
VM OS

VM OS Kernel

https://cdn.kernel.org/pub/linux/kernel/v5.x/

https://www.virtualbox.org/

https://ubuntu.com/download/desktop



Project Environment

- Recommend project environment
  - VirtualBox + Ubuntu + Linux 5.0

Virtual machine

VM OS

VM OS Kernel

https://www.virtualbox.org/

https://ubuntu.com/download/desktop

https://cdn.kernel.org/pub/linux/kernel/v5.x/

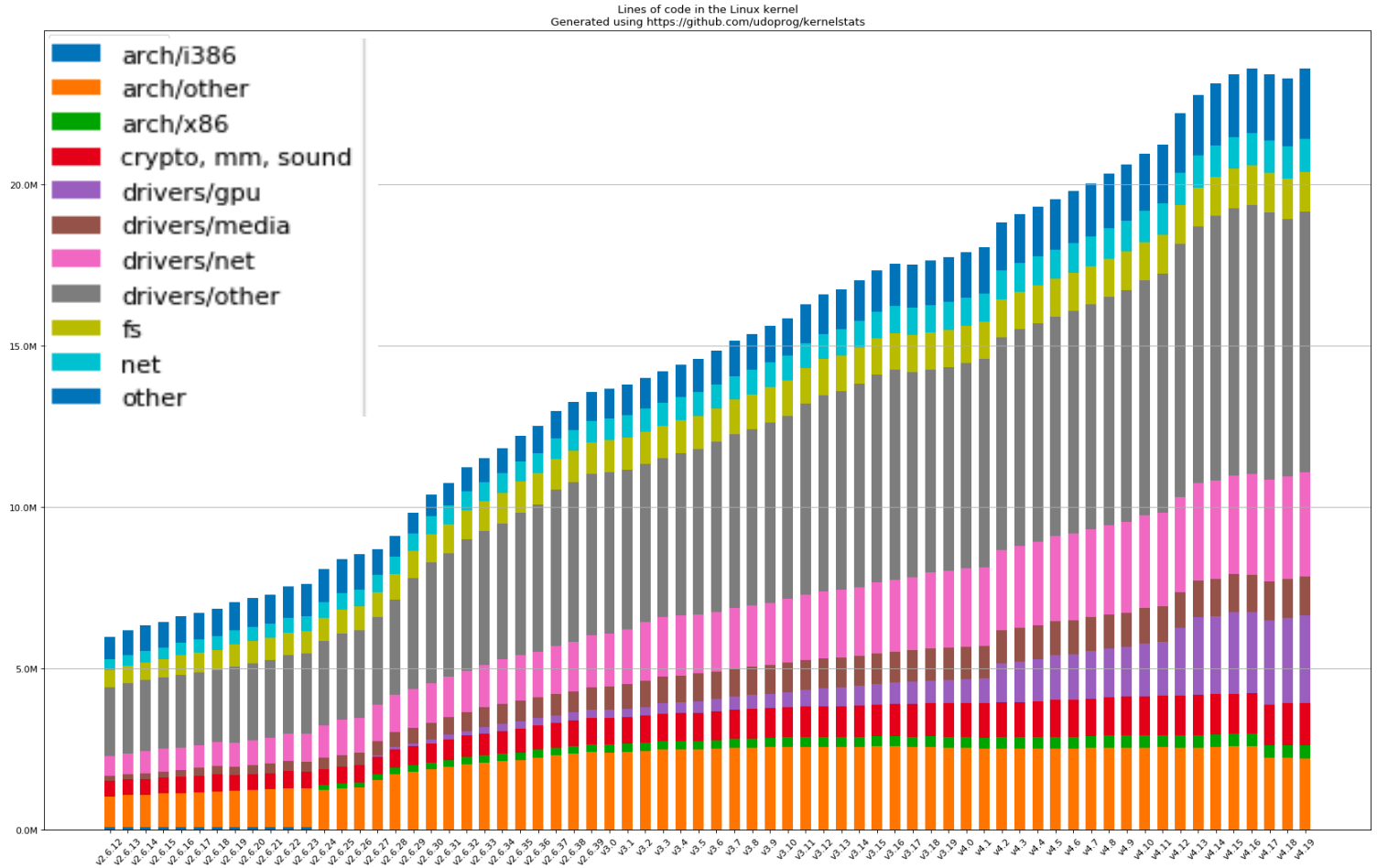
- New to VirtualBox?
  - https://oracle-base.com/articles/vm/virtualbox-creating-a-new-vm
  - https://www.youtube.com/watch?v=sB_5fqiysi4

Outline

- Introduction & Basics
- Why study Operating Systems ?
- What to learn ?
- Course structure
- Course policy
- Course goals

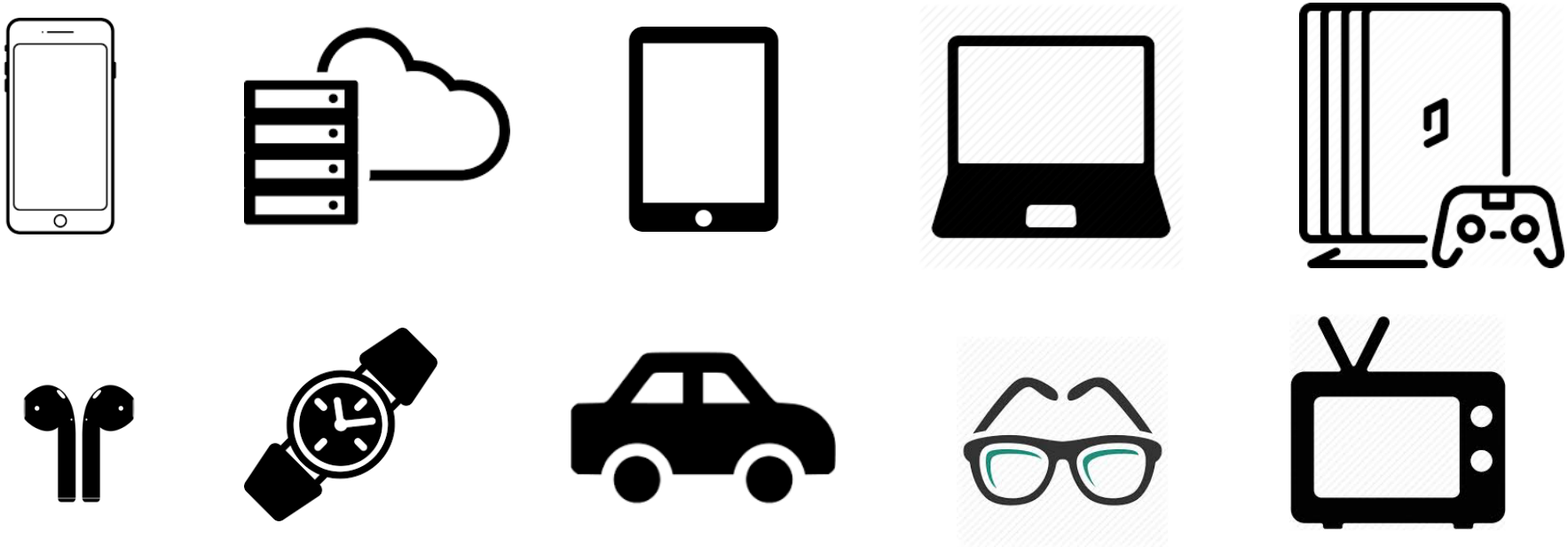
Why Study Operating Systems ?

- The most complex software
  - ~ 20+ million lines of code in Linux



Why Study Operating Systems ?

- The most fundamental software
  - OSs are almost everywhere, e.g., supercomputer, PC, phone…

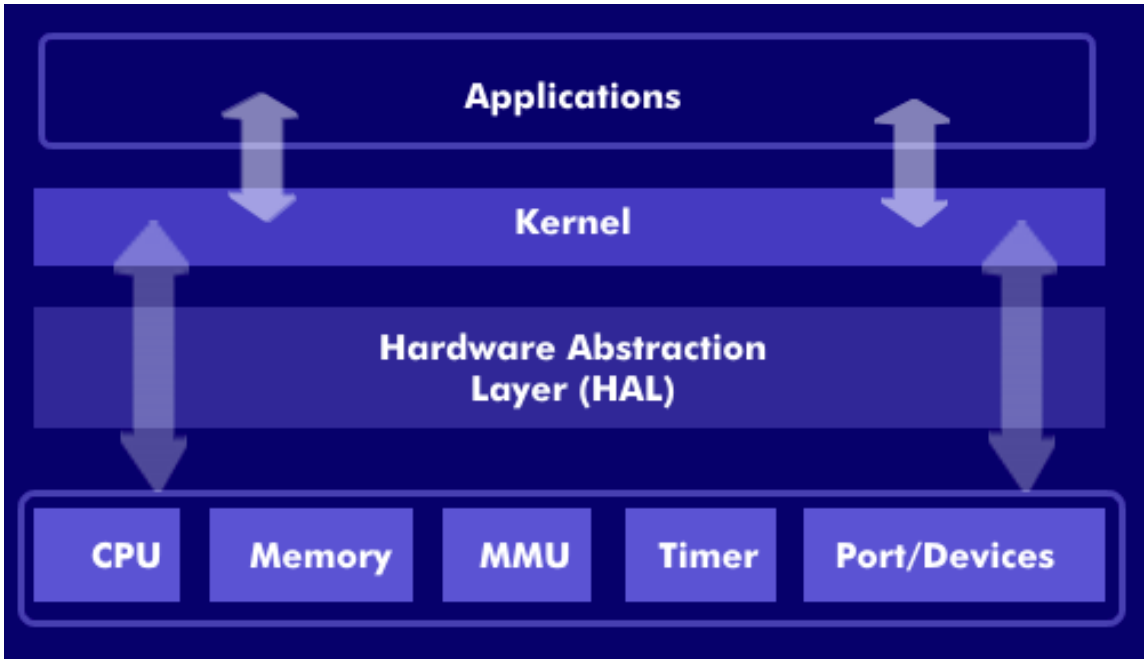


Why Study Operating Systems ?

- The most complex software
  - ~ 20+ million lines of code in Linux
- The most fundamental software
  - OSs are almost everywhere, e.g., supercomputer, PC, phone…
- By studying OS, you will
  - Learn how computers work
  - Gain a good understanding of OS with hardware and application
  - Learn about system design
    - Simplicity, portability, performance, and trade-offs

What to Learn in OSes?

1. Hardware abstraction
   - processes, threads, pages, files …

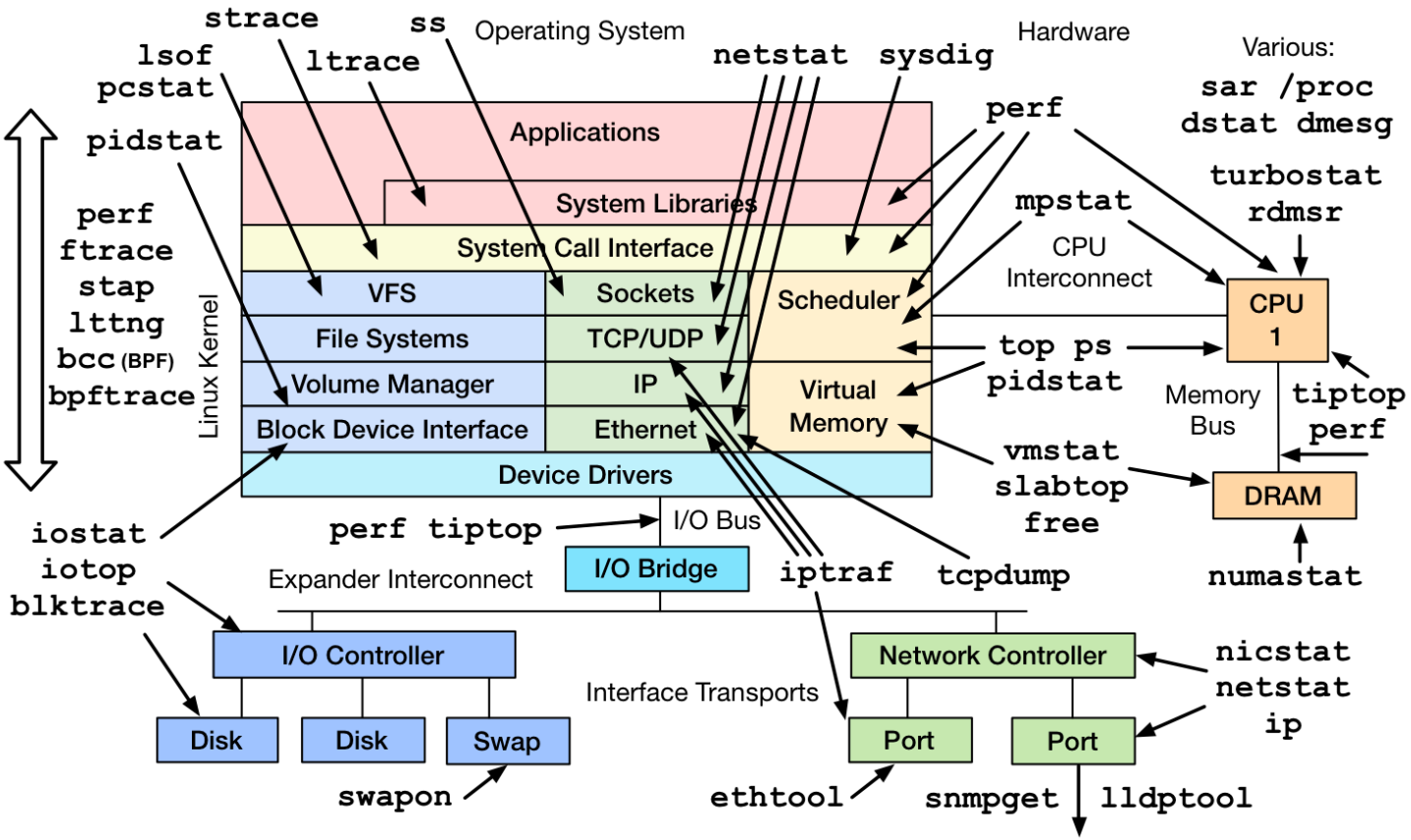


What to Learn in OSes?

2. Resource management

http://www.brendangregg.com/linuxperf.html

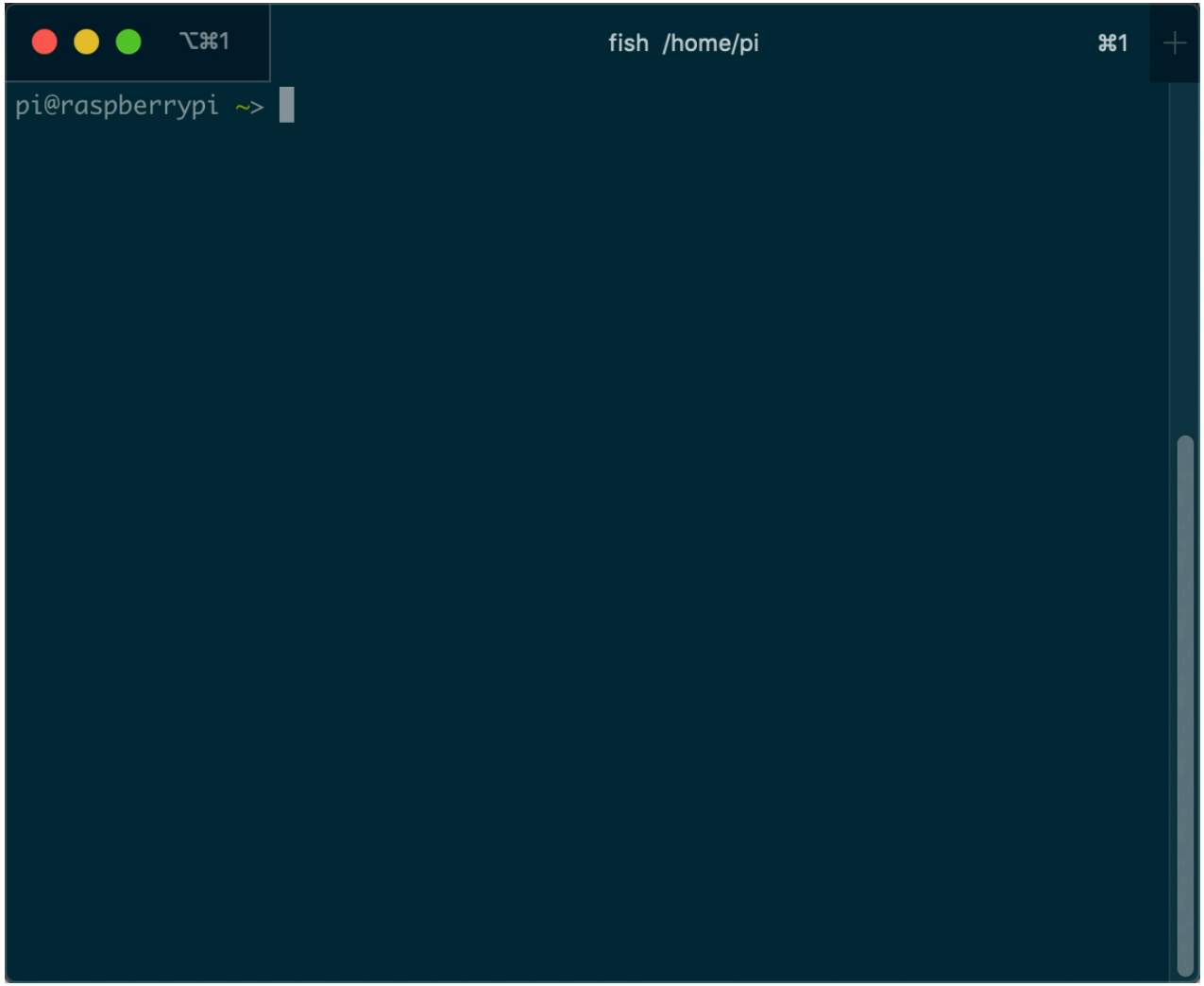
- process scheduling, memory management, file systems …



What to Learn in OSes?

2. Resource management
   - process scheduling, memory management, file systems …
   - E.g., nmon

http://nmon.sourceforge.net/pmwiki.php

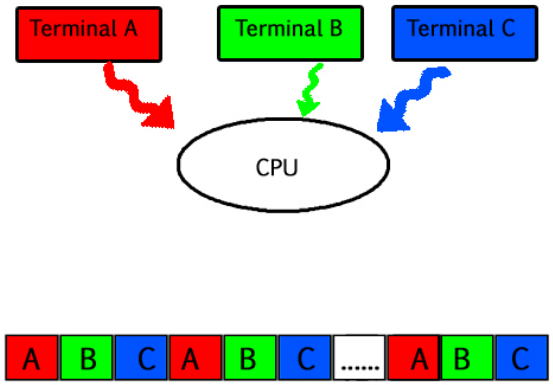


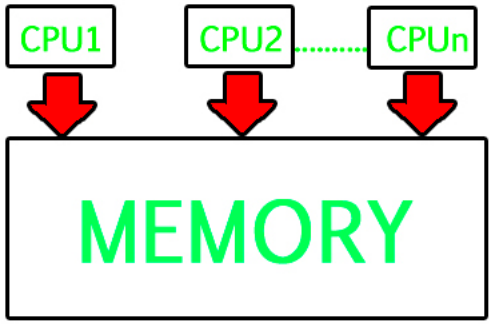
What to Learn in OSes?

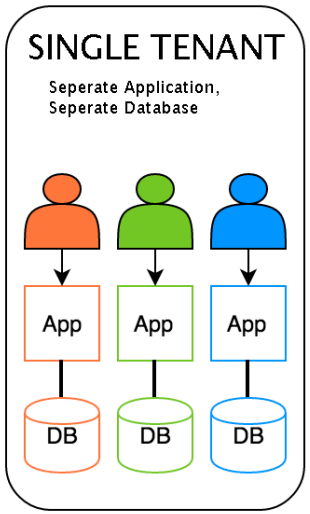
3. Coordination

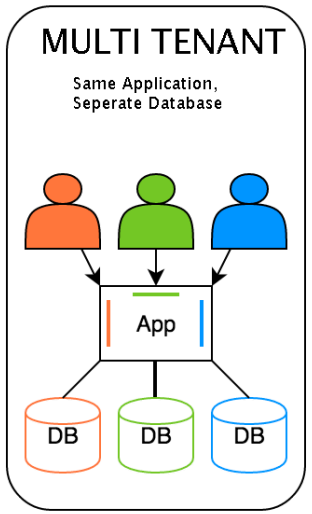
- Multiple programs and users
- Fairness vs. efficiency

Order
Period
Priority
Preemption
Fairness on different resources
…



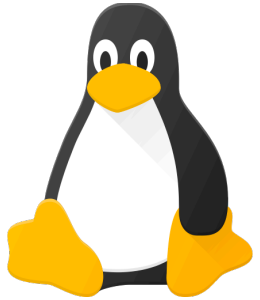






What to Learn in OSes?

- Hardware abstraction
  - processes, threads, pages, files …
- Resource management
  - CPU scheduling, memory management, file systems …
- Coordination
  - Multiple programs and users
  - Fairness and efficiency
- Case: Linux https://elixir.bootlin.com/linux/latest/source



Why Linux? Cloud and mobile.

https://www.cbtnuggets.com/blog/certifications/open-source/why-linux-runs-90-percent-of-the-public-cloud-workload

August 10, 2018 | Open Source - By Team Nuggets

Why Linux runs 90 percent of the public cloud workload

https://arstechnica.com/gadgets/2019/11/google-outlines-plans-for-mainline-linux-kernel-support-in-android/

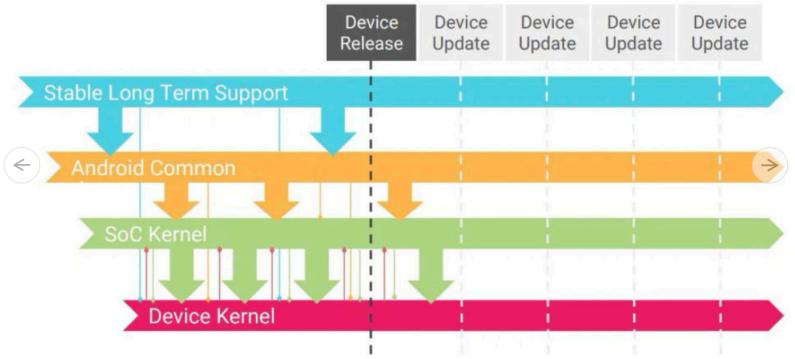
ars TECHNICA

ONE KERNEL TO BOOT THEM ALL —

Google outlines plans for mainline Linux kernel support in Android

Google wants less forking, more modularization for Android's Linux kernel.

RON AMADEO - 11/19/2019, 7:00 PM



What to Learn ?

Week	Topics	Homework/Project
Week 1	Introduction, OS Overview	
Week 2	Process	Project1
Week 3	Thread, Lab class project 1	
Week 4	Lock, Pthread	HW1
Week 5	CPU scheduling, Midterm Exam	Project2
Week 6	Lab class project 2, Memory	
Week 7	Page replacement, File system	HW2
Week 8	Miscellaneous in OS, Final exam	

http://pages.cs.wisc.edu/~remzi/OSTEP/

Course Structure

- Lectures
  - 3502/W01: D2L online
- Homework
  - 2 written assignments
- Projects
  - 2 programming assignments (platform Linux 5.0+)
- Exams (open books, open notes)
  - Midterm: D2L
  - Final: D2L

Course Policy

- Grading scale

Percentage	Grade
90 - 100	A
80 - 89	B
70 - 79	C
60 - 69	D
Below 60	F

Grading Policy (cont.)

- Grading percentage
  - In-class discussion and attendance: 5%
  - Homework assignments (x2): 10%
  - Projects (x2): 35%
  - Midterm: 20%
  - Final exam: 30%

Late submission policy: late submission will **not be accepted** and **no credits**

Academic Integrity

- Academic dishonesty
  - Cheating
  - Plagiarism
  - Collusion
  - The submission for credit of any work or materials that are attributable in whole or in part to another person
  - Taking an examination for another person
  - Any act designed to give unfair advantage to a student or the attempt to commit

Where to go for help ?

- Ask questions in class
- Ask questions outside class
  - Classmates and friends
- Attend office hours
  - Dr. Kun Suo: Tuesday/Thursday 3:00PM – 4:00PM, J-318
- Search on the web
  - Stand on the shoulder of giants

Fundamental Goals

1. Learning the concepts in OSes
   - Attend class on time
   - Ask questions if you have
   - Review the slides and learn from the internet
   - Working homework by your own

Fundamental Goals

2. Learning how to program with OS
   - Be able to design and implement well-structured system software, e.g., system calls
   - Learn how to use OS abstractions, e.g., process, thread, pages, files, …
   - Master how to use resources in OS, e.g., CPU, memory, disk, …
   - Learn how to debug and solve problems

Conclusion

- Why study Operating Systems ?
  - The most complex software
  - The most fundamental software
- What to learn ?
  - Hardware abstraction
  - Resource management
  - Coordination
- Course structure
- Course policy
- Course goals
  - Learning the concepts
  - Learning how to program with OS