

# CS 3502

# Operating Systems

## Introduction

**Kun Suo**

Computer Science, Kennesaw State University

<https://kevinsuo.github.io/>

# Outline

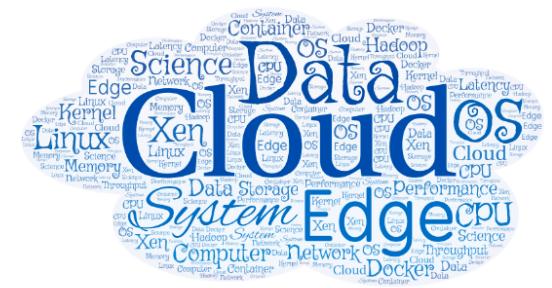
---

- 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 you 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



Windows 1  
1985



1992



1995



Windows XP  
2001



Windows Vista  
2006



Windows 7  
2009



Windows 8  
2012



Windows 10  
2015



# Now it's your turn

---



# Now it's your turn

 redhat	 MEPIS	 turboLinux	 LUNAR	 EvilEntity	 debian	 Vine Linux	 cAos/CentOS	 MiniKazit	 UTUTO
 archlinux	 m0n0wall	 jam	 Knoppix STD	 gentoo linux	 DeLi Linux	 Hiweed	 amlug	 slackware	 yellow dog linux
 Fedora	 LPG	 PLD	 SLAX	 COREL LINUX	 Progeny	 GEEKBOX	 BIGLINUX	 FREEDUC	 Lycoris
 EnGarde	 Mandrakelinux	 Beatrix	 Linspire	 Suse	 YOPER	 BearOps	 ASPLINUX	 kalango	
 Slackintosh	 Frugalware	 Foresight	 Mint	 PCLinuxOS	 Haydar Linux	 sabayon	 ubuntu	 JULEX	 blag



# Now it's your turn



Cupcake  
Android 1.5



Donut  
Android 1.6



Eclair  
Android 2.0/2.1



Froyo  
Android 2.2/2.2.3



Gingerbread  
Android 2.3/2.3.7



Honeycomb  
Android 3.0/3.2



Android 7.0  
**NOUGAT**



# Now it's your turn



# Course Information

---

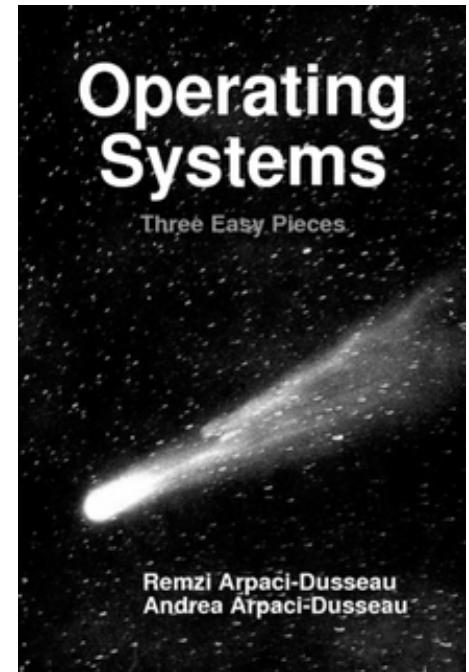
- Instructor: Dr. Kun Suo
- Office: J-318
- Email: [ksuo@kennesaw.edu](mailto:ksuo@kennesaw.edu)
  - Only reply to e-mails that are sent from KSU student email accounts and title the course number [CS3502]
- Office Hours:
  - T/Th, 3pm-4pm
  - By appointment
- Course Materials
  - Homework assignments, 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 <a href="#">Dialogue</a>	12 <a href="#">Dialogue</a>	25 <a href="#">Dialogue</a>	35 <a href="#">Dialogue</a>	<a href="#">Dialogue</a>
TOC	4 <a href="#">Processes</a>	13 <a href="#">Address Spaces</a> <small>code</small>	26 <a href="#">Concurrency and Threads</a> <small>code</small>	36 <a href="#">I/O Devices</a>	<a href="#">Virtual Machines</a>
1 <a href="#">Dialogue</a>	5 <a href="#">Process API</a> <small>code</small>	14 <a href="#">Memory API</a>	27 <a href="#">Thread API</a> <small>code</small>	37 <a href="#">Hard Disk Drives</a>	<a href="#">Dialogue</a>
2 <a href="#">Introduction</a> <small>code</small>	6 <a href="#">Direct Execution</a>	15 <a href="#">Address Translation</a>	28 <a href="#">Locks</a> <small>code</small>	38 <a href="#">Redundant Disk Arrays (RAID)</a>	<a href="#">Monitors</a>
	7 <a href="#">CPU Scheduling</a>	16 <a href="#">Segmentation</a>	29 <a href="#">Locked Data Structures</a>	39 <a href="#">Files and Directories</a>	<a href="#">Dialogue</a>
	8 <a href="#">Multi-level Feedback</a>	17 <a href="#">Free Space Management</a>	30 <a href="#">Condition Variables</a> <small>code</small>	40 <a href="#">File System Implementation</a>	<a href="#">Lab Tutorial</a>
	9 <a href="#">Lottery Scheduling</a> <small>code</small>	18 <a href="#">Introduction to Paging</a>	31 <a href="#">Semaphores</a> <small>code</small>	41 <a href="#">Fast File System (FFS)</a>	<a href="#">Systems Labs</a>
	10 <a href="#">Multi-CPU Scheduling</a>	19 <a href="#">Translation Lookaside Buffers</a>	32 <a href="#">Concurrency Bugs</a>	42 <a href="#">FSCK and Journaling</a>	<a href="#">xv6 Labs</a>
	11 <a href="#">Summary</a>	20 <a href="#">Advanced Page Tables</a>	33 <a href="#">Event-based Concurrency</a>	43 <a href="#">Log-structured File System (LFS)</a>	
		21 <a href="#">Swapping: Mechanisms</a>	34 <a href="#">Summary</a>	44 <a href="#">Flash-based SSDs</a>	
		22 <a href="#">Swapping: Policies</a>		45 <a href="#">Data Integrity and Protection</a>	
		23 <a href="#">Complete VM Systems</a>		46 <a href="#">Summary</a>	
		24 <a href="#">Summary</a>		47 <a href="#">Dialogue</a>	
				48 <a href="#">Distributed Systems</a>	
				49 <a href="#">Network File System (NFS)</a>	
				50 <a href="#">Andrew File System (AFS)</a>	
				51 <a href="#">Summary</a>	



# Prerequisites

---

- Computer basics that are supposed to be 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

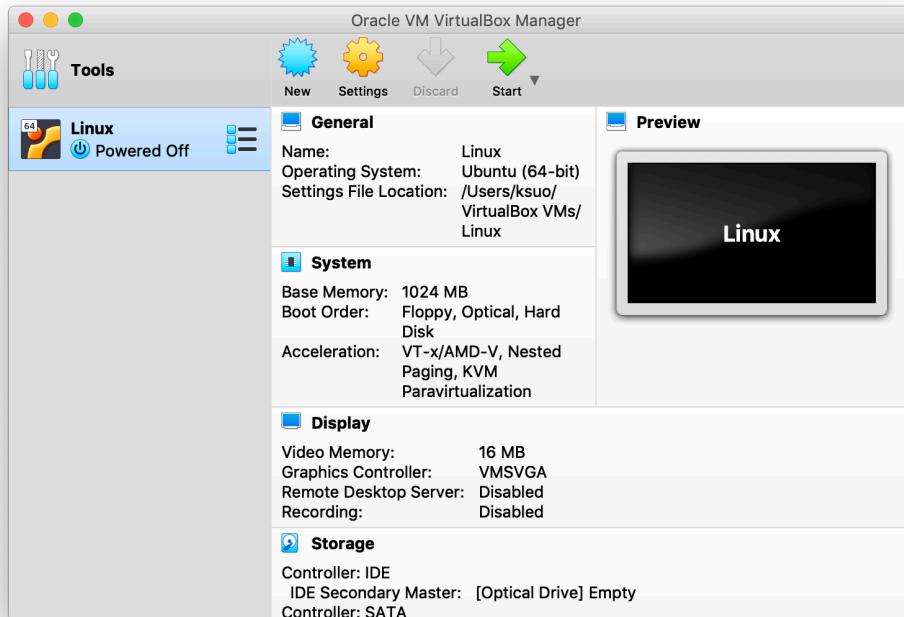
VM OS

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\\_5fqysi4](https://www.youtube.com/watch?v=sB_5fqysi4)

# Outline

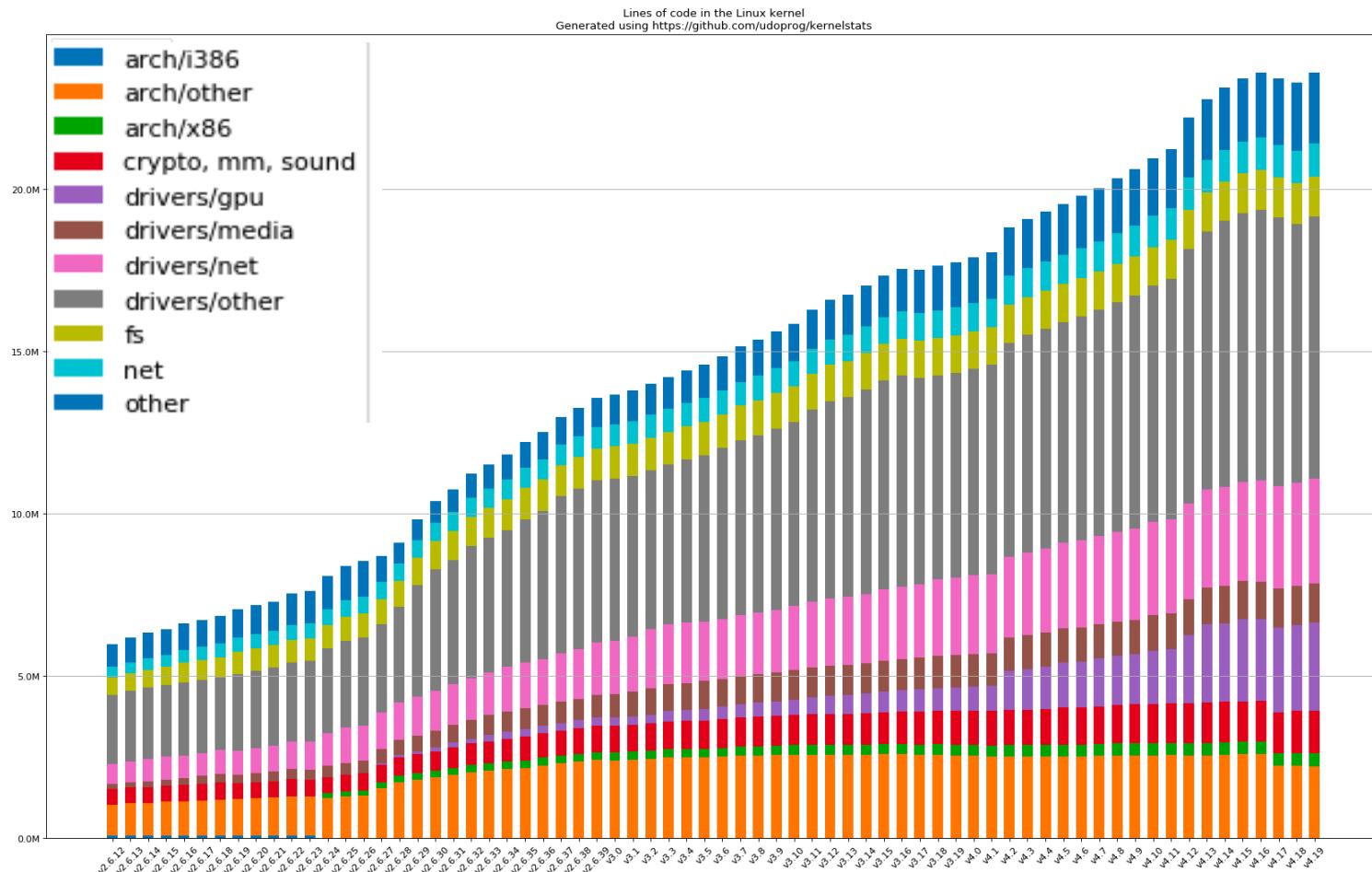
---

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



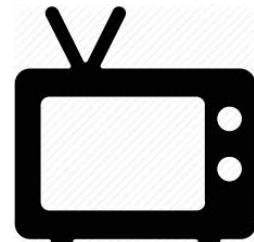
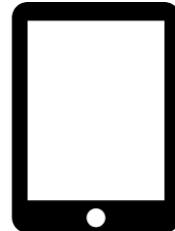
# Why Study Operating Systems ?

- 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 ?

---

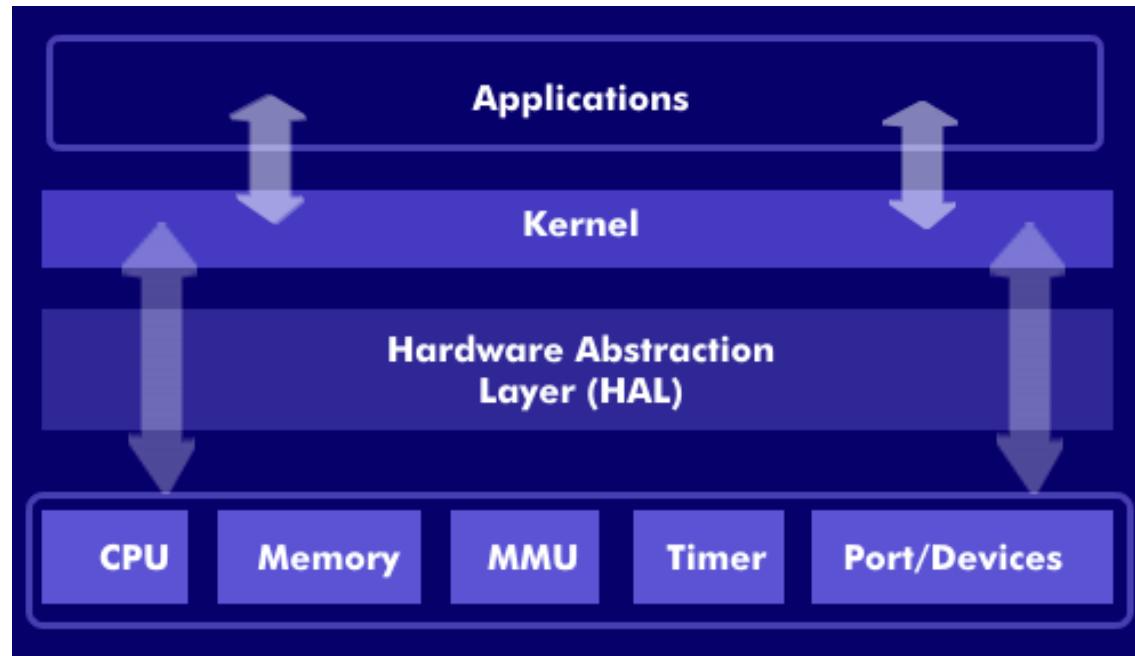
- The most complex software
  - ~ 20+ million lines of code in Linux
- The most fundamental software
  - OSs are almost everywhere, e.g., supercomputer, PC, phone...
- By studying OS, you will
  - Learn how computers work
  - Gain a good understanding of OS with hardware and application
  - Learn about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



# What to Learn in OSes?

## 1. Hardware abstraction

- processes, threads, pages, files ...

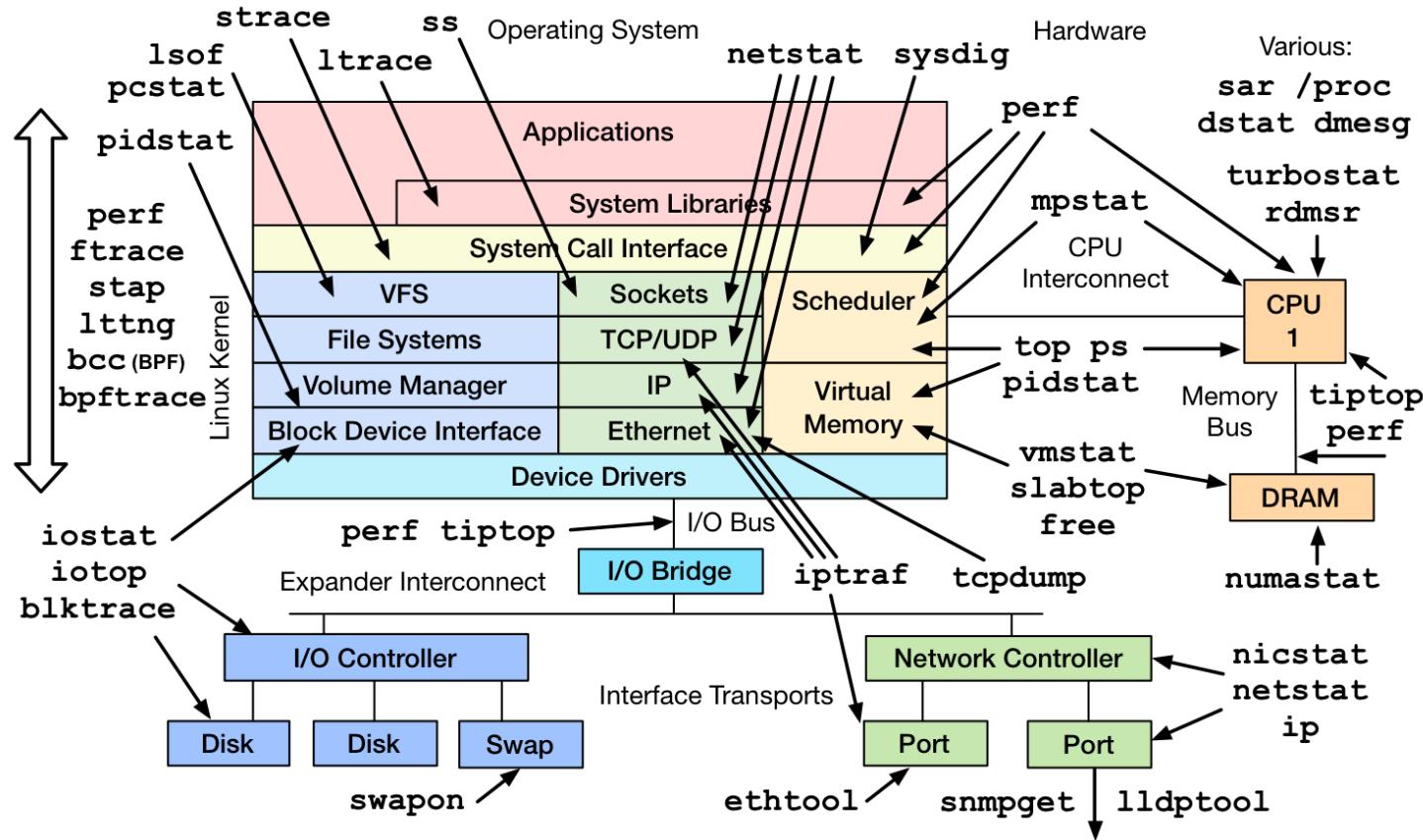


# What to Learn in OSes?

## 2. Resource management

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

- process scheduling, memory management, file systems ...

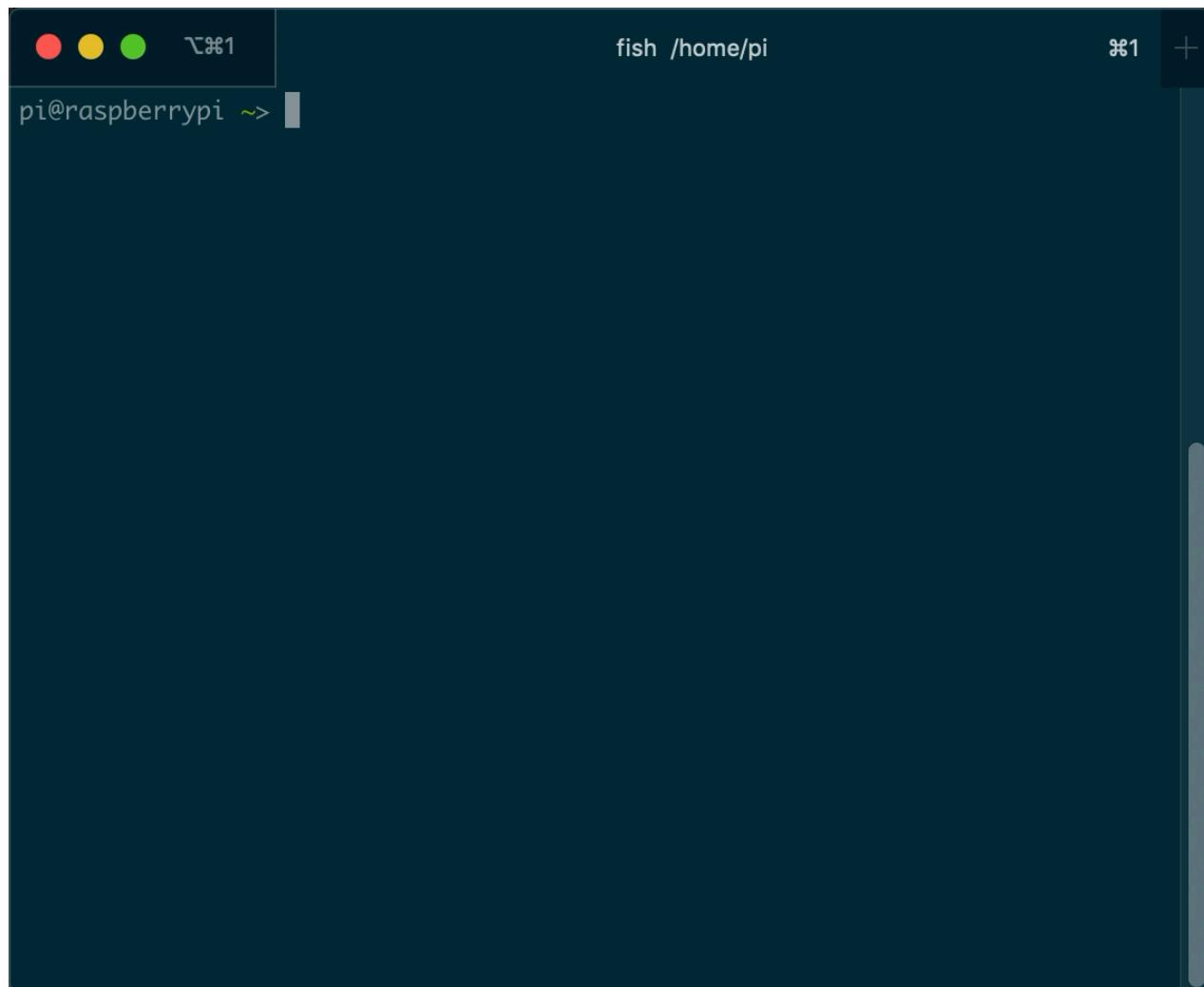


# What to Learn in OSes?

## 2. Resource management

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

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

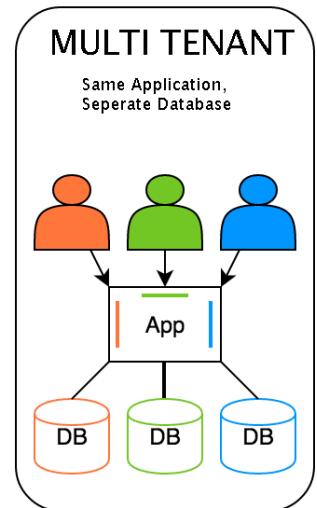
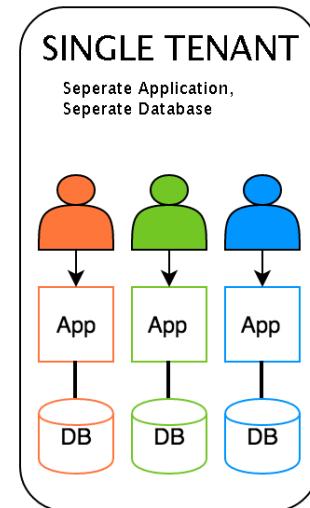
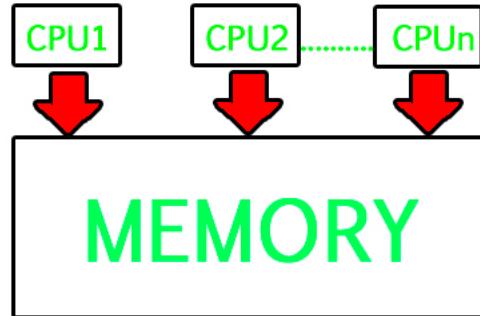
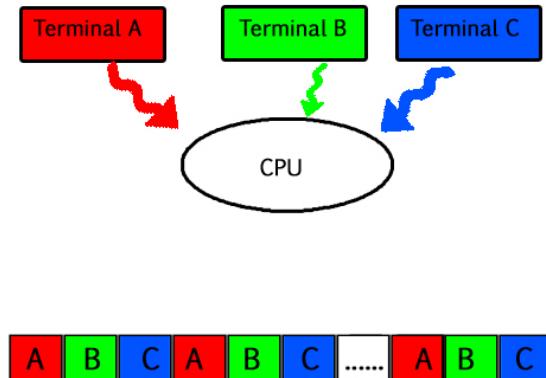


# What to Learn in OSes?

## 3. Coordination

- Multiple programs and users
- Fairness vs. efficiency

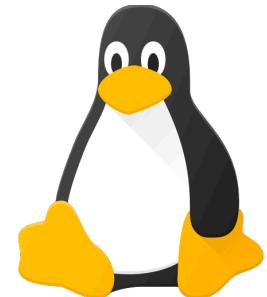
{ Order  
Period  
Priority  
Preemption  
Fairness on different resources  
...



# 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/>



# What to Learn ?

Week/Date	Topic	Chapters	Assignment	Project
1	Introduction and OS overview	1,2		
2	Processes	4, 5		
3	Thread	26, 27		
4	System call	5		Project 1
5	IPC and IRQ	29, 30, 31		
6	CPU scheduling	7, 10		
7	Deadlock and Pthread	28, 29, 30, 31	HW1	Project 2
8	Mid-term exam			
9	Memory	13, 14, 15, 16		
10	Page design and implementation	18, 19, 20, 21, 22		Project 3
11	File system	39, 40		
12	Storage	37, 41, 42, 43, 44		
13	I/O	36		
14	Virtualization and Cloud	23	HW2	Project 4
15	Conclusion	11, 24, 34, 51		
16	Final exam			

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



# Course Structure

---

- Lectures
  - 3502/01: T/R 11:00AM - 12:15PM, 3502/02: M/W 3:30PM-4:45PM
  - 3502/01: Atrium Building (J) - 151, 3502/02: Atrium Building (J) - 132
- Homework
  - 2 written assignments
- Projects
  - 4 programming assignments (platform Linux 5.0+)
- Exams (open books)
  - Midterm: in class, TBA.
  - Final: TBA



# 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 (x4): 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

