Operating Systems, CS-UH 3010

Fall 2019

Instructor:	Azza Abouzied azza@nyu.edu	Lectures:	MW 11:50 AM – 1:05 PM @ CR 006
TA:	Nabil Rahiman nabil.rahiman@nyu.edu	Labs:	S 2:55 PM – 4:10 PM @ A2 004

Course Description: The operating system is a computer's chief manager overseeing interactions between users, applications, shared software and hardware resources. This course covers the fundamentals of operating system design and implementation. Lectures present the central ideas and concepts such as synchronization, deadlock, process management, storage and memory management, file systems, security, protection, networking and virtualization. Assigned readings and programming assignments illustrate the manifestation of these concepts in real modern operating systems and future research ones.

Course Page: bit.ly/nyuadOS

Course textbook:

- Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne. **Operating System Concepts**. 9th. Wiley, 2014.

Reference textbooks:

- Andrew S. Tanenbaum and Herbert Bos. **Modern Operating Systems**. 4th. Pearson, 2014.
- Robert Love. **Linux Kernel Development**. 3rd. Addison-Wesley Professional, 2010.

Getting help: Email any questions to the Piazza class forum for the most immediate help from instructors or peers. I'm available for an hour after every class or at other times by appointment.

Learning Outcomes:

- Explain basic abstraction techniques employed by operating systems
- Explain trade-offs made by particular operating system designs
- Understand components of an OS by implementing different ones through a series of five, challenging and programming-intensive labs in Assembly and C.
- Synthesize a set of design principles that are useful for building large systems
- Analyze, critique and debate research articles on system design
- Study how one can break down fundamental assumptions about hardware, process behavior, trust, etc studied earlier in the course to construct novel and interesting operating systems

Teaching Methodologies:

- *Lectures*: In class lectures will cover fundamental OS design concepts. Most lectures have *in-class interactive exercises* and students are expected to participate. For practical demonstrations of an OS, we will focus on the Linux operating system.
- *Readings*: The course schedule lists sections of the course textbook as well as research papers that students should read prior to class. By keeping on top of the readings, you will make the best use of lecture time: you can clarify concepts you found difficult to understand and you can better participate in class discussions and exercises.
- *Collaborative work*: You will work with your peers to complete your programming-intensive labs and to understand and present an assigned research paper. Effective team work is crucial for developing large software systems.

- *Labs, Tutorials & Design Reviews*: You will complete a series of five to six programming-intensive group labs[1] to build the following OS components: bootloader, a non-preemptive kernel, a preemptive kernel, inter-process communication, a virtual memory system and possibly a file system if time permits. By implementing the building blocks of a working kernel, you apply the concepts learned in class. Professor Kai Li at Princeton University has kindly shared these labs with us. A condition of using the Princeton code base is not to distribute/share the skeleton code or your solutions. All assignment resources are therefore distributed through NYU Drive. Solutions must be submitted securely and you should not publish your solutions online.

  For some labs, we will run tutorials a few days after release. Design reviews are 15 minute meetings that you will schedule with your instructors a week after lab release. Design reviews consist of lab milestones that you need to complete as well as open-ended design discussions. **Design reviews are graded.**

- *Class Presentations & Discussions*: **Paper Cuts** is a debate between two student groups on a single research paper. Each group gives a 10 minute intro on the paper and then each group takes a for/against position and the debate begins. Your stance will be chosen randomly. This requires each group not only to examine the paper but also to examine secondary sources such as online commentary, preceding and follow-on work. You will also evaluate the paper's experimental methods. You will argue for or against the paper in a style similar to what a conference program committee does. Points will be awarded for dynamic presentations and convincing arguments. Imagine your fellow classmates and instructors are a star OS/systems development team and your goal is either to make us implement the system described in the paper or move away to something else.

- *Critiques*: For each of the assigned Paper Cuts research papers, students are expected to write a one to two page critique where they summarize the most important contributions of the paper and describe its strengths and weaknesses. By writing critiques, you will gain a more thorough understanding of research developments and appreciate the complexities and nuances of OS designs and trade-offs.

- *Review Notes & Questions*: Every few weeks, I will post a set of review notes and questions for the material covered. You can self-assess your understanding of the material by answering the questions.

Grading Policy:

5-6 Programming Labs	50%
Midterm 1	15%
Midterm 2	15%
Paper Cuts	10%
Paper Critiques	10%
Bonus and Participation	≈5%

In general, an 80% or above is within the A range, 70%-80% is within the B range and 60%-70% is within the C range. Typically, marks are not curved.

[1] Kai Li and University of Tromsø. **COS 318 Operating System Projects**. URL: https://www.cs.princeton.edu/courses/archive/fall16/cos318/projects.html.

Course Schedule:

This is a tentative schedule. We may spend more or less time on a certain topic.

Week	Lectures, Readings, Case Studies, Assignments
1	**Overview** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 1, 2 *Reading:* Ritchie and Thompson, "The UNIX Time-sharing System" *Optional Reading:* Mickens, "The Night Watch" *Assignment:* 0 Preparation Lab **The Boot Process** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Sec 3.1-3.3 *Assignment:* 1 The Bootloader (2 weeks)
2	**Communicating with the Kernel: System Calls, Interrupts & Exceptions** *Reading:* Love, *Linux Kernel Development*, Chp 5, 7 *Optional Reading:* Love, *Linux Kernel Development*, Chp 8 *Tutorial:* Lab 1 **Processes & Threads** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 3, 4 *Assignment:* Design Review for Lab 1
3	**Concurrency Control: Synchronization Primitives** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 5 **Concurrency Control: Semaphores and More; Deadlocks** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 7 *Reading:* Birell, *An Introduction to Programming with Threads* *Optional Reading:* Love, *Linux Kernel Development*, Chp 10 *Assignment:* 2 The Non-Premeptive Kernel (3 weeks)
4	**Process Scheduling** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 6 *Tutorial:* Lab 2 *Assignment:* Design Review A for Lab 2
5	**Preemption** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 7 *Reading:* Love, *Linux Kernel Development*, Chp 4 **Inter Process Communication (IPC)** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Sec 3.4-3.6 *Tutorial:* Lab 2 *Assignment:* Design Review B for Lab 2
6	**Memory Management** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 8 **Virtual Memory & Paging** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 9 *Optional Reading:* Love, *Linux Kernel Development*, Chp 12, 15 *Assignment:* 3 Kernel with Preemptive Scheduling (2 weeks)
7	**Paging & TLB** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 9 Midterm 1 *Assignment:* Design Review for Lab 3

8	**Device Drivers** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 13 **Block Devices** *Reading:* Love, *Linux Kernel Development*, Chp 14 *Optional Reading:* Bornholt et al., "A DNA-Based Archival Storage System" *Assignment:* 4 Inter-process Communication (IPC) (2 weeks)
9	**File Systems - Interface** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 11 *Reading:* Love, *Linux Kernel Development*, Chp 13 **File Systems - Implementation** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 12 *Assignment:* Design Review for Lab 4
10	**Caching, Journaling & Recovery** *Reading:* Love, *Linux Kernel Development*, Chp 16 *Reading:* Marshall K. McKusick et al., "A Fast File System for UNIX" **Log Structured File Systems** *Reading:* Rosenblum and J. K. Ousterhout, "The Design and Implementation of a Log-structured File System" *Optional Reading:* Ghemawat, Gobioff, and Leung, "The Google File System" *Optional Reading:* Thomson and Abadi, "CalvinFS: Consistent WAN Replication and Scalable Metadata Management for Distributed File Systems" *Assignment:* 5 Virtual Memory (3 weeks)
11	**Distributed Systems** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 17 *Reading:* Lamport, "Paxos Made Simple" *Optional Reading:* Corbett et al., "Spanner: Google's Globally Distributed Database" *Tutorial:* Lab 5 *Assignment:* Design Review A for Lab 5
12	**Virtualization** *Reading:* Silberschatz, Galvin, and Gagne, *Operating System Concepts*, Chp 16 *Optional Reading:* Adams and Agesen, "A Comparison of Software and Hardware Techniques for x86 Virtualization" *Assignment:* Design Review B for Lab 5
13	**Paper Cuts:** Hu, Liu, and Huang, "A Case for Lease-Based, Utilitarian Resource Management on Mobile Devices" *Assignment:* Written critiques for first paper cuts **Paper Cuts:** Ge et al., "Time Protection: The Missing OS Abstraction" *Assignment:* Written critiques for second paper cuts
14	**Paper Cuts:** Tsai and Sanchez, "Compress Objects, Not Cache Lines: An Object-Based Compressed Memory Hierarchy" *Assignment:* Written critiques for third paper cuts Midterm 2

Additional Course Readings:

Paper Cuts:
Recent Papers

- Qian Ge et al. **Time Protection: The Missing OS Abstraction**. In: *Proceedings of the Fourteenth EuroSys Conference 2019*. EuroSys '19. Dresden, Germany: ACM, 2019, 1:1–1:17. URL: http://doi.acm.org/10.1145/3302424.3303976.

- Yigong Hu, Suyi Liu, and Peng Huang. **A Case for Lease-Based, Utilitarian Resource Management on Mobile Devices**. In: *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. ASPLOS '19. Providence, RI, USA: ACM, 2019, pp. 301–315. URL: http://doi.acm.org/10.1145/3297858.3304057.
- Po-An Tsai and Daniel Sanchez. **Compress Objects, Not Cache Lines: An Object-Based Compressed Memory Hierarchy**. In: *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. ASPLOS '19. Providence, RI, USA: ACM, 2019, pp. 229–242. URL: http://doi.acm.org/10.1145/3297858.3304006.
- Min Hong Yun and Lin Zhong. **Ginseng: Keeping Secrets in Registers When You Distrust the Operating System**. In: *NDSS*. 2019. URL: https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_01A-2_Yun_paper.pdf.

Papers from past years

- Jian Xu and Steven Swanson. **NOVA: A Log-structured File System for Hybrid Volatile/Non-volatile Main Memories**. In: *Proceedings of the 14th Usenix Conference on File and Storage Technologies*. FAST'16. Santa Clara, CA: USENIX Association, 2016, pp. 323–338. URL: http://dl.acm.org/citation.cfm?id=2930583.2930608.
- Silas Boyd-Wickizer, Austin T. Clements, et al. **An Analysis of Linux Scalability to Many Cores**. In: *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*. OSDI'10. Vancouver, BC, Canada: USENIX Association, 2010, pp. 1–16. URL: http://dl.acm.org/citation.cfm?id=1924943.1924944.
- Silas Boyd-Wickizer, Haibo Chen, et al. **Corey: An Operating System for Many Cores**. In: *Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation*. OSDI'08. San Diego, California: USENIX Association, 2008, pp. 43–57. URL: http://dl.acm.org/citation.cfm?id=1855741.1855745.
- Galen C. Hunt and James R. Larus. **Singularity: Rethinking the Software Stack**. In: *SIGOPS Oper. Syst. Rev.* 41.2 (Apr. 2007), pp. 37–49. URL: http://doi.acm.org/10.1145/1243418.1243424.

Other:

- James C. Corbett et al. **Spanner: Google's Globally Distributed Database**. In: *ACM Trans. Comput. Syst.* 31.3 (Aug. 2013), 8:1–8:22. URL: http://doi.acm.org/10.1145/2491245.
- James Mickens. **The Night Watch**. In: *;login: logout: The Usenix Magazine* 17.7 (Nov. 2013), pp. 5–8. URL: https://www.usenix.org/system/files/1311_05-08_mickens.pdf.
- Keith Adams and Ole Agesen. **A Comparison of Software and Hardware Techniques for x86 Virtualization**. In: *SIGARCH Comput. Archit. News* 34.5 (Oct. 2006), pp. 2–13. URL: http://doi.acm.org/10.1145/1168919.1168860.
- Vijayan Prabhakaran, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau. **Analysis and Evolution of Journaling File Systems**. In: *Proceedings of the Annual Conference on USENIX Annual Technical Conference*. ATEC '05. Anaheim, CA: USENIX Association, 2005, pp. 8–8. URL: http://dl.acm.org/citation.cfm?id=1247360.1247368.
- Maurice Herlihy and J. Eliot B. Moss. **Transactional Memory: Architectural Support for Lock-free Data Structures**. In: *SIGARCH Comput. Archit. News* 21.2 (May 1993), pp. 289–300. URL: http://doi.acm.org/10.1145/173682.165164.
- Mendel Rosenblum and John K. Ousterhout. **The Design and Implementation of a Log-structured File System**. In: *ACM Trans. Comput. Syst.* 10.1 (Feb. 1992), pp. 26–52. URL: http://doi.acm.org/10.1145/146941.146943.
- Andrew D. Birell. **An Introduction to Programming with Threads**. 1989. URL: https://birrell.org/andrew/papers/035-Threads.pdf.
- Marshall K. McKusick et al. **A Fast File System for UNIX**. In: *ACM Trans. Comput. Syst.* 2.3 (Aug. 1984), pp. 181–197. URL: http://doi.acm.org/10.1145/989.990.
- Dennis M. Ritchie and Ken Thompson. **The UNIX Time-sharing System**. In: *Commun. ACM* 17.7 (July 1974), pp. 365–375. URL: http://doi.acm.org/10.1145/361011.361061.
- James Bornholt et al. **A DNA-Based Archival Storage System**. In: *SIGPLAN Not.* 51.4 (Mar. 2016), pp. 637–649. URL: http://doi.acm.org/10.1145/2954679.2872397.
- John Ousterhout et al. **The RAMCloud Storage System**. In: *ACM Trans. Comput. Syst.* 33.3 (Aug. 2015), 7:1–7:55. URL: http://doi.acm.org/10.1145/2806887.
- Alexander Thomson and Daniel J. Abadi. **CalvinFS: Consistent WAN Replication and Scalable Metadata Management for Distributed File Systems**. In: *Proceedings of the 13th USENIX Conference on File and Storage Technologies*. FAST'15. Santa Clara, CA: USENIX Association, 2015, pp. 1–14. URL: http://dl.acm.org/citation.cfm?id=2750482.2750483.

- Abhishek Verma et al. **Large-scale Cluster Management at Google with Borg**. In: *Proceedings of the Tenth European Conference on Computer Systems*. EuroSys '15. Bordeaux, France: ACM, 2015, 18:1–18:17. URL: http://doi.acm.org/10.1145/2741948.2741964.
- Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. **The Google File System**. In: *SIGOPS Oper. Syst. Rev.* 37.5 (Oct. 2003), pp. 29–43. URL: http://doi.acm.org/10.1145/1165389.945450.
- Marshall Kirk McKusick and Gregory R. Ganger. **Soft Updates: A Technique for Eliminating Most Synchronous Writes in the Fast Filesystem**. In: *Proceedings of the Annual Conference on USENIX Annual Technical Conference*. ATEC '99. Monterey, California: USENIX Association, 1999, pp. 24–24. URL: http://dl.acm.org/citation.cfm?id=1268708.1268732.

Academic Integrity: As set forth in NYU Abu Dhabi's Academic Integrity Policy, the relationship between students and faculty at NYU Abu Dhabi is defined by a shared commitment to academic excellence and is grounded in an expectation of fairness, honesty, and respect, which are essential to maintaining the integrity of the community. Every student who enrolls and everyone who accepts an appointment as a member of the faculty or staff at NYU Abu Dhabi agrees to abide by the expectation of academic honesty.

The full policies and procedures relating to Academic Integrity may be found on the NYUAD Student Portal.