Week 1: Lecture A
Course Intro & The Security Mindset

Tuesday, August 22, 2023
Reminders

- Be sure to join the course **Canvas**, **Gradescope**, and **Piazza**
  - See links at top of course page
  - [http://cs4440.eng.utah.edu](http://cs4440.eng.utah.edu)

- Trouble accessing? See me after class!
  - Or email me at: snagy@cs.utah.edu
Today’s Class

- Welcome to CS 4440 😊
- Course Overview
- The Security Mindset
  - Thinking like an attacker
  - Thinking as a defender
- Ethics and Academic Integrity
Course Staff

Stefan Nagy
Assistant Professor, KSoC
Email: snagy@cs.utah.edu
Office: Merrill Eng. 3446

Teaching Assistants

Mandy Mieu
Porter Smith
Trace Engel
About Me

Stefan Nagy
Assistant Professor, KSoC

Stefan Nagy
Assistant Professor, KSoC

Co-founder and Co-director:

SSG UTAH SOFTWARE SECURITY GROUP
SCHOOL OF COMPUTING | THE UNIVERSITY OF UTAH

Places I’ve been:

University of Utah, 2022–now
Virginia Tech, Ph.D. 2016–2022
Univ. of Illinois, B.S. 2012–2016

cs.utah.edu/~snagy
twitter.com/snagycs
@snagy@infosec.exchange
My Research: Fast Fuzzing

- Fastest techniques for fuzzing closed-source executable code
  - ZAFL (USENIX’21)
    - Make testing of closed-source code as fast as open-source
  - UnTracer (S&P’19), HeXcite (CCS’21)
    - Make testing of closed-source code faster than open-source
My Research: Fast Fuzzing

- **Winnie-AFL**: Winnie-AFL is a fork of WinAFL that supports fuzzing using a fork()-like API. For more details about Winnie, check out the NDSS paper.

- **bsod-kernel-fuzzing**: This repository contains the implementations described in "BSOD: Binary-only Scalable fuzzing Of device Drivers".

- **TrapFuzz**: Hacky support for (basic-block) coverage guided fuzzing of closed source libraries for honggfuzz.
My Research: Extending Fuzzing’s Reach

- Highly-configurable commodity software
  - Variability-induced bugs
  - Patch integrity vetting
  - Accelerating bug discovery
My Research: Extending Fuzzing’s Reach

- Highly-configurable commodity software
  - Variability-induced bugs
  - Patch integrity vetting
  - Accelerating bug discovery

- Expanding auditing of opaque software
  - Testing binary analysis tools
  - Automated interface recovery
  - Closed-source operating systems
Our work: systems and software security, binary analysis, fuzzing
The Utah Cybersecurity Club

Come hack with us!
Help us get to know you!
Help us get to know you!

- Throughout the course we’ll use **Poll Everywhere**
  - Use your laptop or phone to send-in your responses
  - Poll participation = **2.5%** of your grade

- To receive credit:
  - Log-in as your UID (e.g., u8675309)
  - Answer the following questions to give us some more info about you!
## Experience with Programming Languages

<table>
<thead>
<tr>
<th>Option</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>(A) HTML / JavaScript</td>
<td>0%</td>
</tr>
<tr>
<td>(B) C / C++</td>
<td>0%</td>
</tr>
<tr>
<td>(C) Python</td>
<td>0%</td>
</tr>
<tr>
<td>(D) Assembly</td>
<td>0%</td>
</tr>
<tr>
<td>(E) None of the above</td>
<td>0%</td>
</tr>
</tbody>
</table>
## Experience with Cybersecurity

<table>
<thead>
<tr>
<th>Option</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>0%</td>
</tr>
<tr>
<td>Some</td>
<td>0%</td>
</tr>
<tr>
<td>More</td>
<td>0%</td>
</tr>
<tr>
<td>I can ace the final now!</td>
<td>0%</td>
</tr>
<tr>
<td>Course Code</td>
<td>Course Name</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>CS 3500</td>
<td>Software Practice</td>
</tr>
<tr>
<td>CS 3505</td>
<td>Software Practice II</td>
</tr>
<tr>
<td>CS 3810</td>
<td>Computer Organization</td>
</tr>
<tr>
<td>CS 4400</td>
<td>Computer Systems</td>
</tr>
</tbody>
</table>
Experience with Tools

- Debuggers (e.g., GDB) 0%
- The Linux Terminal 0%
- Virtual Machines (e.g., VirtualBox) 0%
- Wireshark 0%
- Firefox or Chrome Dev Consoles 0%
Last Question

What do you hope to get out of this course?

And no, I don’t mean the grade that you want 😊
Course Overview
Course Goals

- **Critical Thinking**
  - How to think like an attacker
  - How to reason about threats and risks
  - How to balance security costs and benefits

- **Technical Skills**
  - How to protect yourself
  - How to manage and defend systems
  - How to design and program secure systems

- **Learning to be a security-conscious citizen**

- **Learning to be a L337 H4X0R... but an ethical one!**
## Topics

<table>
<thead>
<tr>
<th>Week 1</th>
<th>Week 2–4</th>
<th>Week 5–8</th>
<th>Week 9–12</th>
<th>Week 13–15</th>
<th>Week 16</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Course Intro &amp; The Security Mindset</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Principles, threat modeling, vulnerabilities, attacking versus defending; VM setup</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>P1: Communications Security</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Public- and private-key crypto, digital signatures, authentication, hashes, secure channels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>P2: Application Security</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Memory protection, sandboxing, virtual machines, software exploitation, malware, testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>P3: Web and Network Security</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IP, TCP, routing, net protocols, web architecture, web attacks, firewalls, intrusion detection</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>P4: New Frontiers in Security</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Side channels, hardware, ML security, reverse engineering, election security, policy, ethics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Course Wrap-up</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Careers in cybersecurity, the security ecosystem; the final exam</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Common Concerns

- Attendance required? **Yes.**
  - Standard lecture format:
    - ~20 minutes of review
    - ~55 minutes of new material

- Textbook is required? **No.**
  - However, “Intro. to Computer Security” (ISBN: 9780321512949) is recommended

- Midterm exam? **No.** Final exam? **Yes.**
  - Covers entire course material
  - Review session as final lecture
  - Similar to homework questions
Grading

- **10%** = weekly solo quizzes based on lectures
- **50%** = four Programming Projects (**12.5%** each)
- **35%** = Final Exam covering all course material
- **5%** = participation during lecture and on Piazza
Lecture Quizzes (10%)

- Weekly exercises to be completed **individually**
  - Designed to test your understanding of the lectures

- Released on **Gradescope** after Tuesday’s lecture
  - You may work until the following **Monday by 11:59 PM**
  - Strict deadline—**late submissions will not be accepted**

- **Lowest score** will be **dropped** at no penalty
Programming Projects (50%)

- Four projects completed in groups of **no more than two**
  - You can discuss your approaches with other groups
  - Must complete and submit **only within your group**

- **Topics:** Crypto, App security, Web security, Net security

- **Where to find and submit?**
  - Distributed via **course website** (we’ll announce when)
  - Upload your work (**one** per team) as tarball to **Canvas**
Project Teams

- Can work in **teams of up to two**
  - Find teammates on Piazza
  - Post on

- Why work with someone else?
  - Pair programming
  - Divide and conquer
  - Two sets of eyes to solve problems
  - Teaching others helps you learn more

- Yes, you are free to work solo...
  - But we encourage you to team up!
Project Lateness Policy

- **Course staff constraints:**
  - We want to return graded work promptly
  - Can’t discuss solutions until all work graded

- **Project lateness policy:**
  - 10% **penalty** for being late up to **two days past deadline**
  - **Will not accept after 48 hours** past the original deadline
  - Extensions made only under **extraordinary** circumstances

- **Please start early!** It is your responsibility to...
  - Turn in assignments **ahead** of the deadline
  - Ensure your submissions **work** as intended
Project Regrade Policy

- After grades posted, **regrade form** open for **one** week
  - We’ll distribute regrade forms via **Piazza**

- Valid regrade requests:
  - You have verified your solution is correct
    (i.e., we made an error in grading)

- Requests that will be **rejected**:
  - My code crashed, but I’ve now fixed it
  - I am looking for more partial credit
  - I submitted late without an extension
  - I missed the regrade request deadline

- **Your** responsibility to stay atop of this!
Project Collaboration Policy

- We encourage you to help each other learn!
  - You may give or receive help on key **high-level concepts**

- However, **all code** must only be written by **you or your team**

- Cheating is when you give/receive an **unfair advantage**. Examples:
  - Distributing your solutions (e.g., to GitHub, Chegg, CourseHero) = cheating
  - Copying code/solutions (e.g., from GitHub, Google, another team) = cheating
  - Copying code/solutions from AI tools (e.g., CoPilot, GPT, Bard, etc.) = cheating

- Violations = misconduct sanctions. **Don’t jeopardize your degree!**
Final Exam (35%)

- One exam covering all course material
- Questions similar to homework problems
- Final lecture will serve as a review session
- **Save the date:** 1–3PM on Monday, December 11
  - Late exams only for conflicts with other finals
Participation (5%)

- **Lecture** participation via PollEverywhere:
  - Three lecture absences allowed at zero penalty
  - We’ll track these internally—no need to notify us
  - To get credit, log-in as your UID (e.g., u8675309)
Participation (5%)

- **Lecture** participation via PollEverywhere:
  - Three lecture absences allowed at zero penalty
  - We’ll track these internally—no need to notify us
  - To get credit, log-in as your UID (e.g., u8675309)

- **Online** participation on course Piazza:
  - Make intellectual contributions to help others learn
  - Collaboration policies apply—**don’t share your code!**
Participation (5%)

- **Lecture** participation via Poll Everywhere:
  - Three lecture absences allowed at zero penalty
  - We'll track these internally—no need to notify us
  - To get credit, log-in as your UID (e.g., u8675309)

- **Online** participation on course Piazza:
  - Make intellectual contributions to help others learn
  - Collaboration policies apply—don’t share your code!

- How to **lose** points:
  - Frequently missing class, or not contributing online
  - Engaging in disruptive behavior or violating policies
Lectures

- Tuesdays and Thursdays
  - 2:00–3:20 PM at Warnock L105

- Take notes!
  - Studies show most effective if hand-written 😊

- Slides posted prior to each lecture
  - See “Schedule” on http://cs4440.eng.utah.edu

- Interrupt with questions, (relevant) stories

- Not recorded—come to lectures!
Office Hours

- **TA office hours (15 total hours)**
  - First-come/first-serve via **TA Queue**
  - Help with homework and projects
  - Go over solutions after deadlines

- **Professor’s office hours**
  - Help understanding lecture **concepts**
  - Administrative issues

- **Check the office hours calendar!**
  - [http://cs4440.eng.utah.edu](http://cs4440.eng.utah.edu)
  - Cancellations announced via **Piazza**
Communication

- **Course website:** your go-to resource for all things CS 4440
  - [http://cs4440.eng.utah.edu](http://cs4440.eng.utah.edu)

---

**CS 4440: Introduction to Computer Security**

This course teaches the security mindset and introduces the principles and practices of computer security as applied to software, systems, and networks. It covers the foundations of building, using, and managing secure systems. Topics include standard cryptographic functions and protocols, and threats and defenses for real-world systems.

This class is open to undergraduates. It is recommended that you have familiarity with topics like software engineering, software debugging, basic networking, computer organization, the web and databases, and the command-line terminal; and with languages such as Python, SQL, HTML, and C/C++. This course is weighted 3 credit hours.

**Learning Outcomes:** At the end of the course, students will be able to:

- Explain the objectives and functions of common communication, network, and software defenses.
- Understand how common vulnerabilities and implementation flaws weaken the security of a system.
- Apply the attacker and defender mindsets to model the threats faced by an arbitrary computer system.
- Identify attack surfaces of system layers and their impacts on confidentiality, integrity, and availability.
- Replicate real-world attacks to exploit flaws in basic communication, network, and software defenses.
- Judge the security of a system based on its access control policies, defense measures, and software.
New for Fall 2023: The CS 4440 Wiki

- Our aim is to lower the overall learning curve
- Resources to help you:
  - Tutorials
  - Cheat Sheets
  - Software documentation
New for Fall 2023: The CS 4440 Wiki

- Our aim is to lower the overall learning curve
- Resources to help you:
  - Tutorials
  - Cheat Sheets
  - Software documentation
- Contributions welcome!
  - Pages, ideas, fixes, etc.
  - Currently only editable by instructors, but eventually will be editable by you!
Summary

Course website .................. wiki, assignments, schedule, slides, office hours

Piazza ......................................................... questions, discussion, announcements

Gradescope .................................................... weekly post-lecture quizzes

Canvas ..................................................... project submission, course gradebook

Instructor email (snagy@cs.utah.edu) .................... administrative issues
Questions?
The Security Mindset
What does Computer Science impact?

Engineering?

Natural Sciences?

Math?

Philosophy?
What does Computer **Security** impact?

Math?  
Engineering?  
Natural Sciences?  
Philosophy?
What’s the difference?

Reliability does not equal Security
“Computer security studies how systems behave in the presence of an adversary.”

- The adversary...
- a.k.a. the attacker
- a.k.a. the bad guy
- An intelligence that actively tries to cause the system to misbehave.
Know thine enemy

- Motives?
  - Disruption
  - Espionage
  - Money

- Capabilities?
  - Denial of service
  - Code execution

- Degree of access?
  - Physical access
  - Root privileges
The Security Mindset

- **Thinking like a defender**
  - Know what you’re defending, and against whom
  - Weigh benefits vs. costs:
    - No system is ever completely secure.
  - Embrace “rational paranoia”

- **Thinking like an attacker**
  - Understand techniques for circumventing security
  - Look for ways security can break, not reasons why it won’t
High-level Approaches

Attacks

Defenses
Why study attacks?

- Identify vulnerabilities so they can be fixed
- Create incentives for vendors to be careful
- Learn about **new classes** of threats
  - Determine what we need to defend against
  - Help designers build stronger systems
  - Help users more accurately evaluate risk
“Insecurity”

- A hierarchy view

---

**Level-2 Problem: “Weakness”**
Factors that predispose systems to vulnerability

**Level-1 Problem: “Vulnerability”**
Specific errors that could be exploited in an assault.

**Level-0 Problem: “The Attack”**
Actual malicious attempt to cause harm.

---

“The Attack”
Assault = recipe, vulnerabilities are ingredients
Thinking like an Attacker

- Look for the weakest links
  - What is easiest to attack

- Identify assumptions that the security depends on
  - Are any assumptions false?
  - Can you render them false?

- Think outside the box!
  - Don’t be constrained by the system designer’s worldview

Practice thinking like an attacker:
For every system you interact with, think about what it means for it to be secure, and imagine how it could be exploited
Exercise

- What are some security systems that you interact with in everyday life?
Exercise

- What are some security systems that you interact with in everyday life?
  - Breaking into a University professor’s office after hours to alter your grade?
Thinking as a Defender

- Security policy
  - What are we trying to protect?
  - What properties are we trying to enforce?
- Threat model
  - Who are the attackers? Capabilities? Motivations?
  - What kind of attack are we trying to prevent?
- Risk assessment
  - What are the weaknesses of the system?
  - What will successful attacks cost us?
- How likely?
  - Countermeasures
  - Costs vs. benefits?
  - Technical vs. nontechnical?

The challenge is to think rationally and rigorously about risk.

Rational paranoia.
Security Policies

- What assets are we trying to protect?
- What properties are we trying to enforce?
  - Confidentiality
  - Integrity
  - Availability
  - Privacy
  - Authenticity
Threat Models

- Who are our adversaries?
  - Motives?
  - Capabilities?
  - Level of access?

- What kinds of attacks must we prevent?
  - Think like the attacker!

- Limits: kinds of attacks we should ignore?
  - Unrealistic versus unlikely
Security through... obscurity?

Common mistake:

- Trying to convince yourself the system is secure since attacker won't know X
Security through... obscurity?

Common mistake:
- Trying to convince yourself the system is secure since attacker won't know X

Better approach:
- Limit the assumptions that the security of your system depends upon
- Assume the attacker knows everything but a small bit of data (e.g., a key)
Assessing Risk

- Remember: *Rational* paranoia

- What would security breaches cost us?
  - Direct: money, intellectual property, safety
  - Indirect: reputation, future business, well being

- How likely are these costs?
  - Probability of attacks?
  - Probability of success?
Countermeasures

- Technical countermeasures
  - Bug fixes, more crypto, re-architecting, etc.

- Nontechnical countermeasures
  - Law, policy (government, institutional)
  - Procedures, training, auditing, incentives, etc.
Costs of Security

- No security mechanism is free

- Direct costs:
  - Design, implementation, enforcement, false positives

- Indirect costs:
  - Lost productivity, added complexity, time to market

- Challenge is to rationally weigh costs vs. risk
  - Human psychology makes reasoning about high cost, low probability events very difficult
Exercises

Should you lock your house/room door?

- Assets?
- Adversaries?
- Risk assessment?
- Countermeasures?
- Costs/benefits?
Using a credit card safely?

- Assets?
- Adversaries?
- Risk assessment?
- Countermeasures?
- Costs/benefits?
Secure Design

- Common mistake:
  - Trying to convince yourself that the system is secure as-is
Secure Design

- **Common mistake:**
  - Trying to convince yourself that the system is secure as-is

- **Better approach:**
  - Identify the weaknesses of your design and focus on correcting them
Secure Design

- **Common mistake:**
  - Trying to convince yourself that the system is secure as-is

- **Better approach:**
  - Identify the weaknesses of your design and focus on correcting them

- **Secure design is a process**
  - Must be practiced continuously
  - Very difficult to be retrofitted
Where to Focus Defenses

- Trusted components (aka Trusted Computing Base)
  - Parts that must function correctly for the system to be secure.

- Attack surface
  - Parts of the system exposed to the attacker

- **Complexity versus security** are inversely related
Other Principles

- Defense-in-Depth
  - Multiple layers of safeguards
  - Physical, technical, administrative

- Diversity
  - More moving parts = harder to attack
  - Conversely, harder to secure

- Maintainability
  - Minimize maintainer workload
  - Make fixes easy/fast to deploy
Exercise

- Preventing cheating on the CS 4440 Final Exam?
Security Testing

- Testing against requirements
  - What you have done up until now?
  - What are the correct requirements?

- Adversarial testing (my work)
  - Black-box testing
  - White-box testing
  - Gray-box testing

- Example: airport security

Red Team agents use disguises, ingenuity to expose TSA vulnerabilities
Learning from Failures

- Time-honored engineering practice
  - Especially important in security

- Identifying causes of failures
  - Where, how, why
  - First step of fixing

- What can failure teach us?
  - New kinds of attacks
  - New kinds of defenses
A Note on Ethics...
Laws and Ethics

- Don’t be evil!
  - Ethics requires you to refrain from doing harm
  - Always respect privacy and property rights
  - Otherwise, you will fail the course (and worse)

- Federal/state laws criminalize computer intrusion, wiretapping, or other abuse
  - Computer Fraud and Abuse Act (CFAA)
  - You can be sued or go to jail

- University policies prohibit tampering with campus or other systems
  - You can/will be disciplined and even expelled
Next time on CS 4440...

Python Tutorial and Course VM Setup
Bring your laptops, and pre-download your VM image!