Week 1: Lecture A

Course Intro & The Security Mindset

Tuesday, August 19, 2025

Reminders

- Be sure to join the course **Canvas** and **Piazza**
  - See links at top of course page
  - 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

Course Instructor

Stefan Nagy
Assistant Professor, KSoC

Email: snagy@cs.utah.edu
Office: Merrill Eng. 3446

Teaching Assistants

Alishia Seo

Alan Mo

Ayden Mcgonigal

Teagan Smith

About Me

Stefan Nagy

Assistant Professor, KSoC

cs.utah.edu/~snagy
twitter.com/snagycs
snagycs.bsky.social
@snagy@infosec.exchange

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

My Research Group

FUTURES³

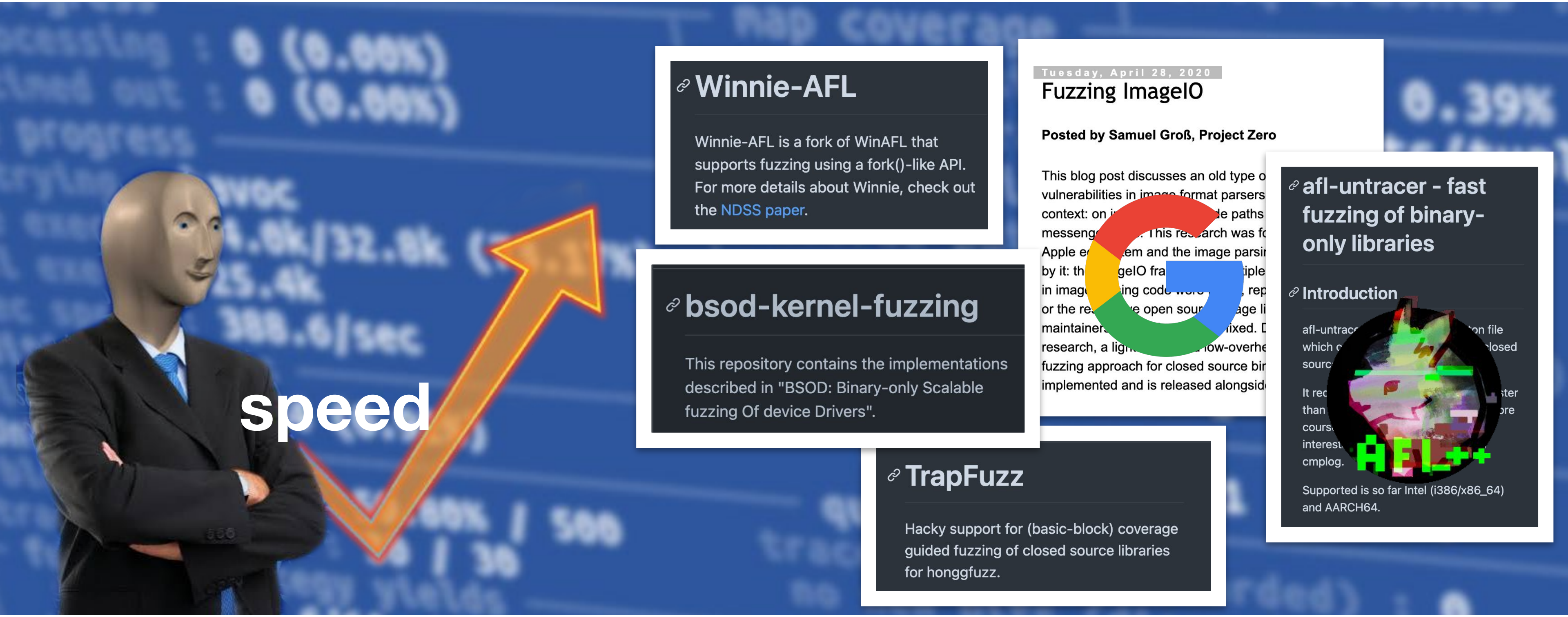
LAB FUTURE TECHNOLGY FOR USABLE, RELIABLE, & EFFICIENT SECURITY OF SOFTWARE & SYSTEMS

SCHOOL OF COMPUTING | THE UNIVERSITY OF UTAH | SALT LAKE CITY

Our work: systems and software security, binary analysis, fuzzing

FUTURES³

My Prior Research: Faster 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".

Tuesday, April 28, 2020

Fuzzing ImageIO

Posted by Samuel Groß, Project Zero

This blog post discusses an old type o vulnerabilities in image format parsers context: on i de paths messeng . This research was fo Apple e em and the image parsi by it: th geIO fra iple in image ing code were , rep or the re e open sour age li maintainer ixed. D research, a ligh low-overhe fuzzing approach for closed source bir implemented and is released alongside

TrapFuzz

Hacky support for (basic-block) coverage guided fuzzing of closed source libraries for honggfuzz.

afl-untracer - fast fuzzing of binary-only libraries

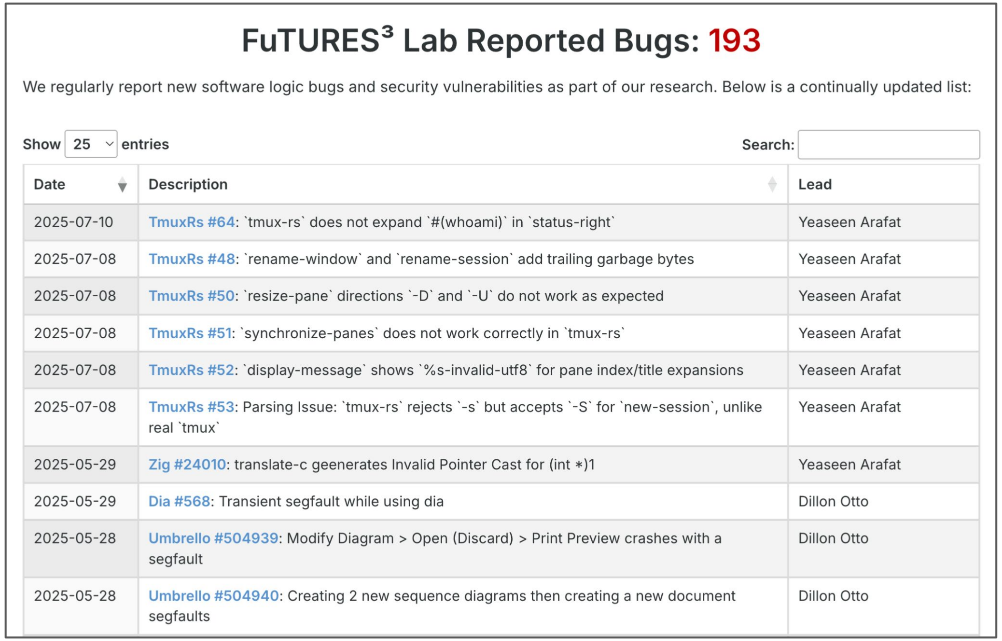
Introduction

afl-untrac on file which c closed sourc

It rec ster than ore cours interest cmplog.

Supported is so far Intel (i386/x86_64) and AARCH64.

My Current Research: More Fuzzing!



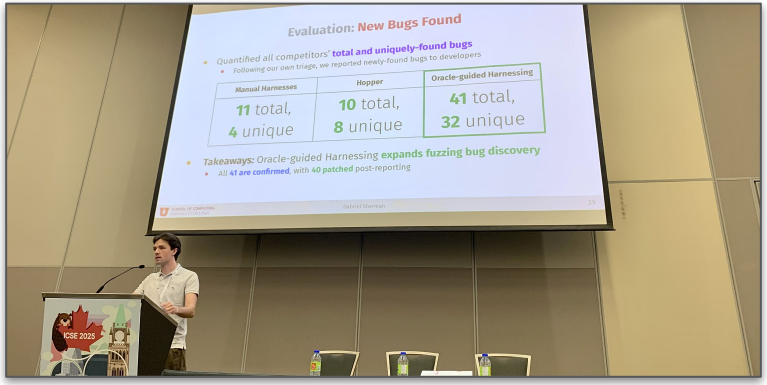
FuTURES³ Lab Reported Bugs: 193

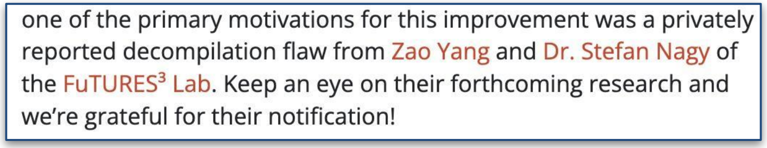
We regularly report new software logic bugs and security vulnerabilities as part of our research. Below is a continually updated list:

Show 25 entries Search:

Date	Description	Lead
2025-07-10	TmuxRs #64: `tmux-rs` does not expand `#(whoami)` in `status-right`	Yeaseen Arafat
2025-07-08	TmuxRs #48: `rename-window` and `rename-session` add trailing garbage bytes	Yeaseen Arafat
2025-07-08	TmuxRs #50: `resize-pane` directions `-D` and `-U` do not work as expected	Yeaseen Arafat
2025-07-08	TmuxRs #51: `synchronize-panes` does not work correctly in `tmux-rs`	Yeaseen Arafat
2025-07-08	TmuxRs #52: `display-message` shows `%s-invalid-utf8` for pane index/title expansions	Yeaseen Arafat
2025-07-08	TmuxRs #53: Parsing Issue: `tmux-rs` rejects `-s` but accepts `-S` for `new-session`, unlike real `tmux`	Yeaseen Arafat
2025-05-29	Zig #24010: translate-c geenerates Invalid Pointer Cast for (int *)1	Yeaseen Arafat
2025-05-29	Dia #568: Transient segfault while using dia	Dillon Otto
2025-05-28	Umbrello #504939: Modify Diagram > Open (Discard) > Print Preview crashes with a segfault	Dillon Otto
2025-05-28	Umbrello #504940: Creating 2 new sequence diagrams then creating a new document segfaults	Dillon Otto

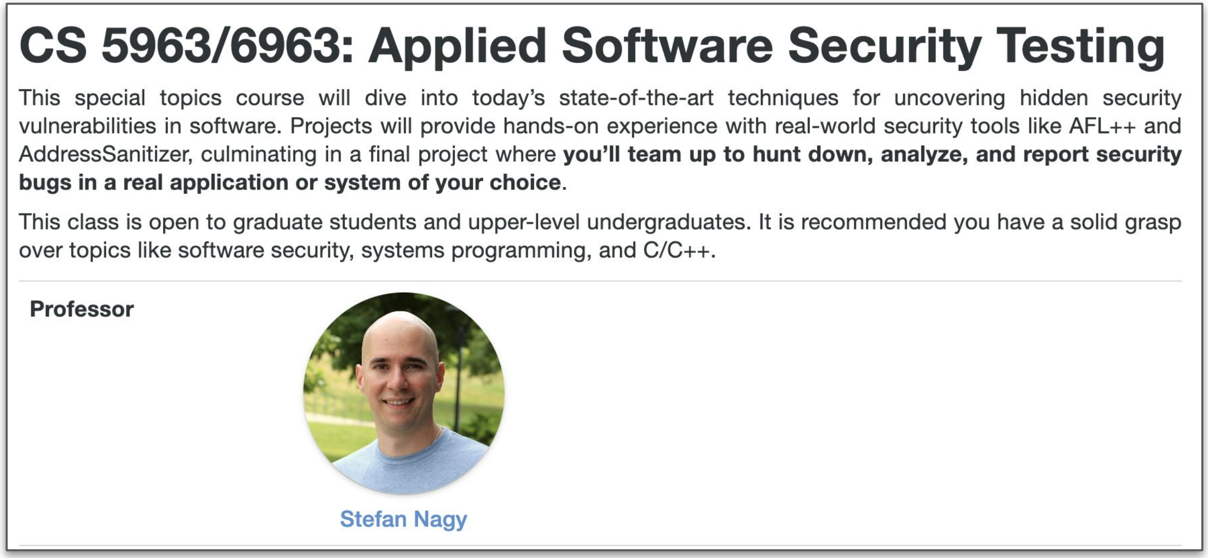
futures.cs.utah.edu/bugs


Evaluation: New Bugs Found
Quantified all competitors' total and uniquely-found bugs
Following our own triage, we reported newly-found bugs to developers
Manual Harnesses
11 total, 4 unique
Hopper
10 total, 8 unique
Oracle-guided Harnessing
41 total, 32 unique
Takeaways: Oracle-guided Harnessing expands fuzzing bug discovery
All 41 are confirmed, with 40 patched post-reporting
ICSE 2025



one of the primary motivations for this improvement was a privately reported decompilation flaw from Zao Yang and Dr. Stefan Nagy of the FuTURES³ Lab. Keep an eye on their forthcoming research and we're grateful for their notification!

My Other Courses



CS 5963/6963: Applied Software Security Testing

This special topics course will dive into today's state-of-the-art techniques for uncovering hidden security vulnerabilities in software. Projects will provide hands-on experience with real-world security tools like AFL++ and AddressSanitizer, culminating in a final project where **you'll team up to hunt down, analyze, and report security bugs in a real application or system of your choice**.

This class is open to graduate students and upper-level undergraduates. It is recommended you have a solid grasp over topics like software security, systems programming, and C/C++.

Professor

Stefan Nagy

cs.utah.edu/~snagy/courses/cs5963/

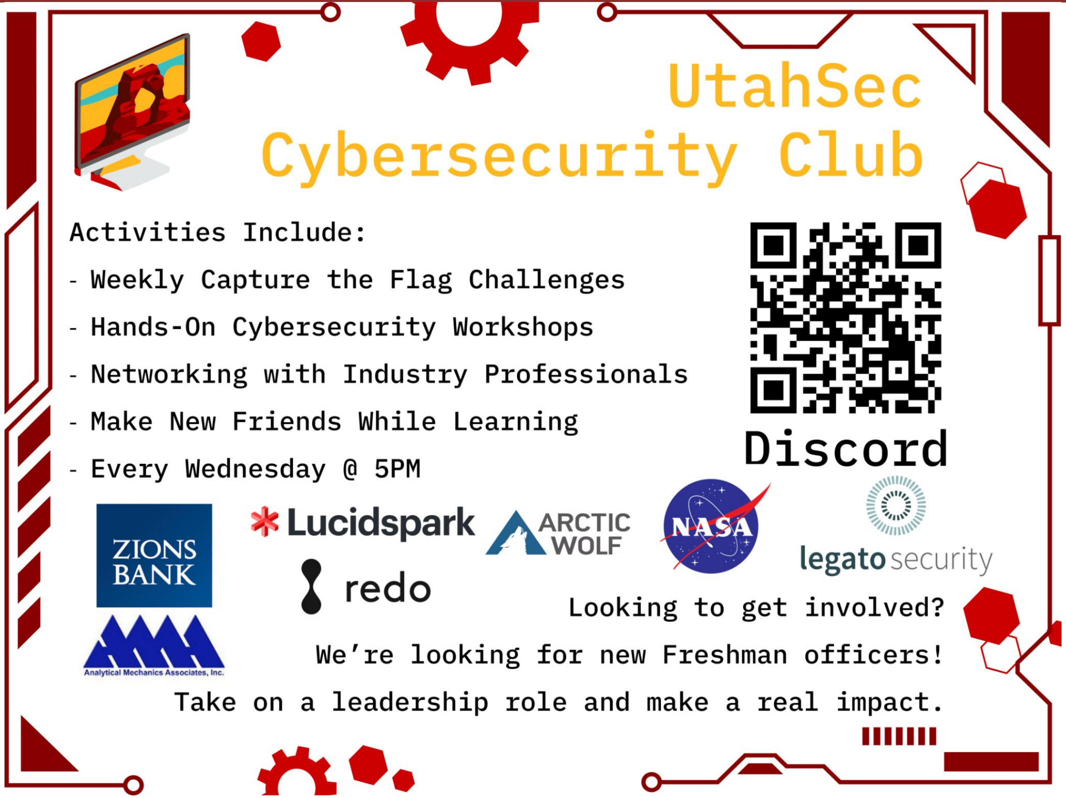
The Utah Cybersecurity Club

utahsec

Come hack with us!

utahsec.cs.utah.edu

The Utah Cybersecurity Club



UtahSec
Cybersecurity Club

Activities Include:

- Weekly Capture the Flag Challenges
- Hands-On Cybersecurity Workshops
- Networking with Industry Professionals
- Make New Friends While Learning
- Every Wednesday @ 5PM

Discord

Looking to get involved?

We're looking for new Freshman officers!

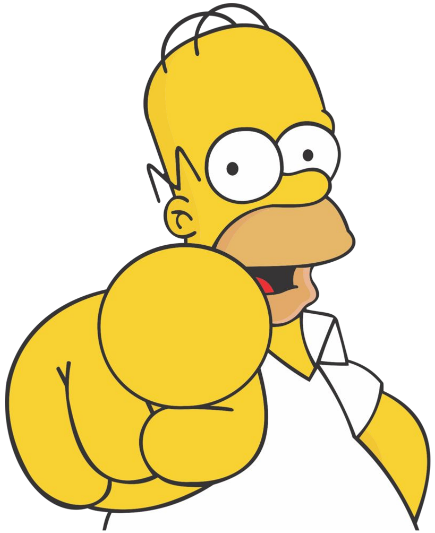
Take on a leadership role and make a real impact.

ZIONS BANK
Lucidspark
ARCTIC WOLF
NASA
legato security
redo
Analytical Mechanics Associates, Inc.

Help us get to know you!

Help us get to know you!

- Throughout lecture we'll use **Poll Everywhere**
  - Use your laptop to send-in your responses
  - Share location—we're checking you're here!
  - Poll participation = **5%** of your grade
- To receive credit:
  - Log-in via your UMAIL (e.g., `u8675309@utah.edu`)
  - We've automatically registered you (**if not, see me**)
- Answer the following questions to give us some more info about you!



Experience with Programming Languages

(A) HTML / JavaScript — 0%

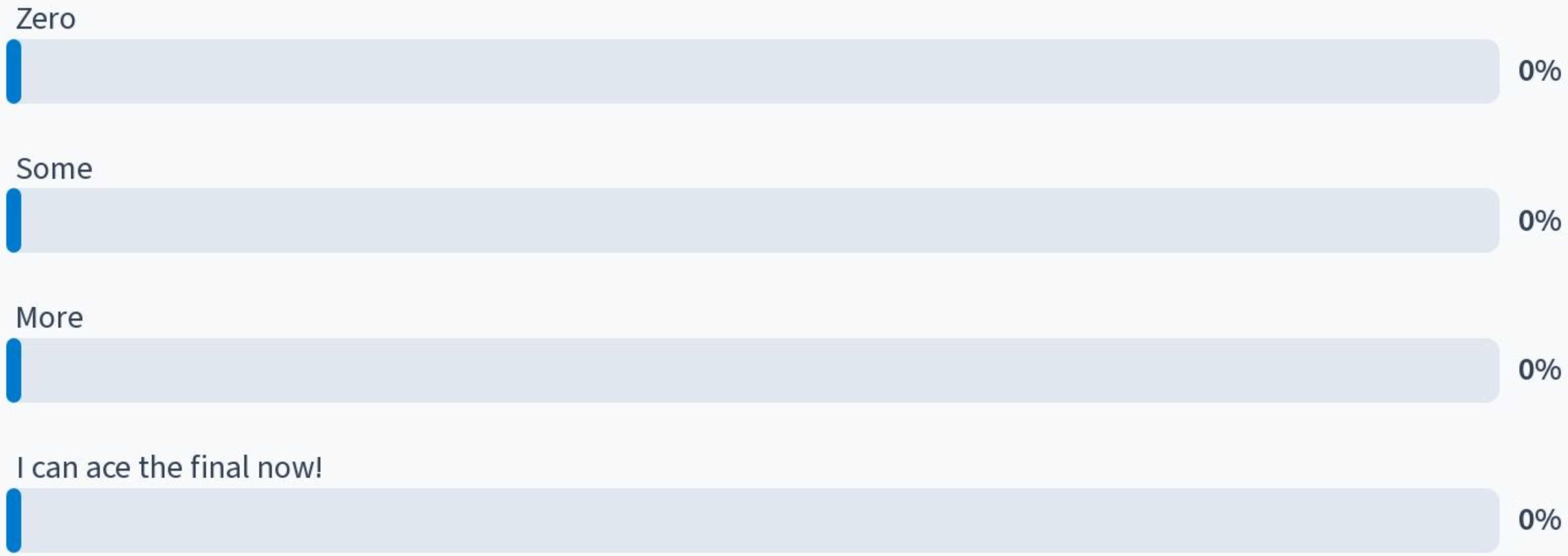
(B) C / C++ — 0%

(C) Python — 0%

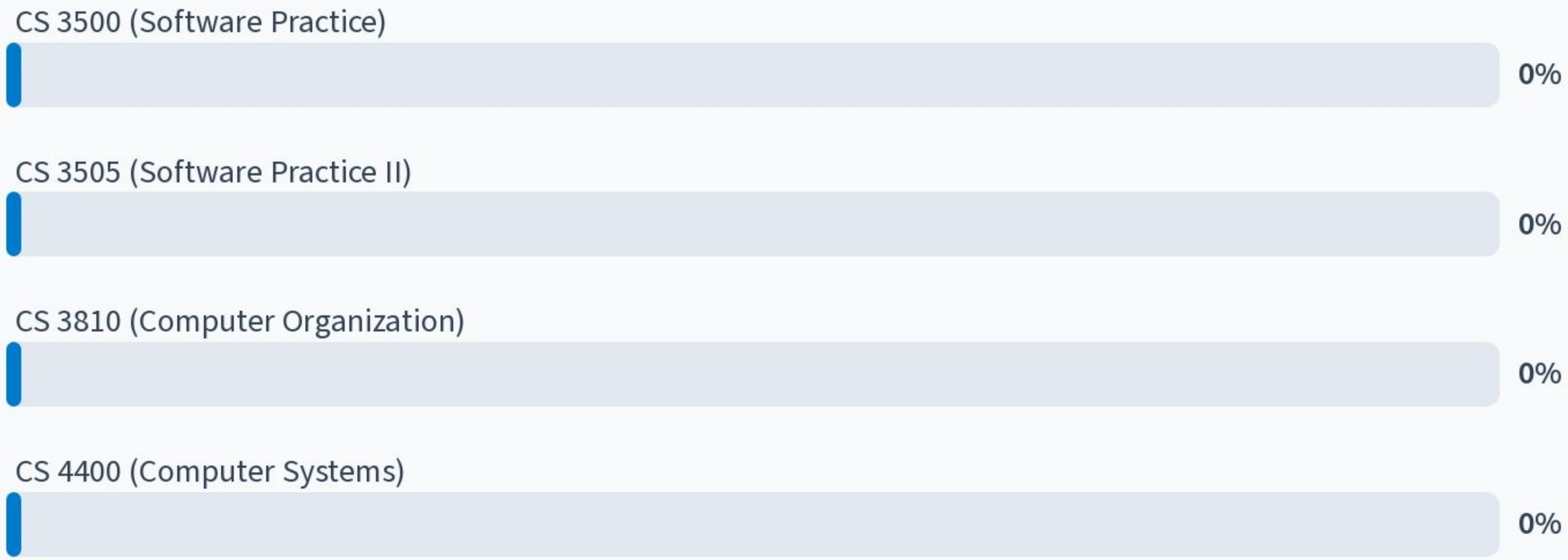
(D) Assembly — 0%

(E) None of the above — 0%

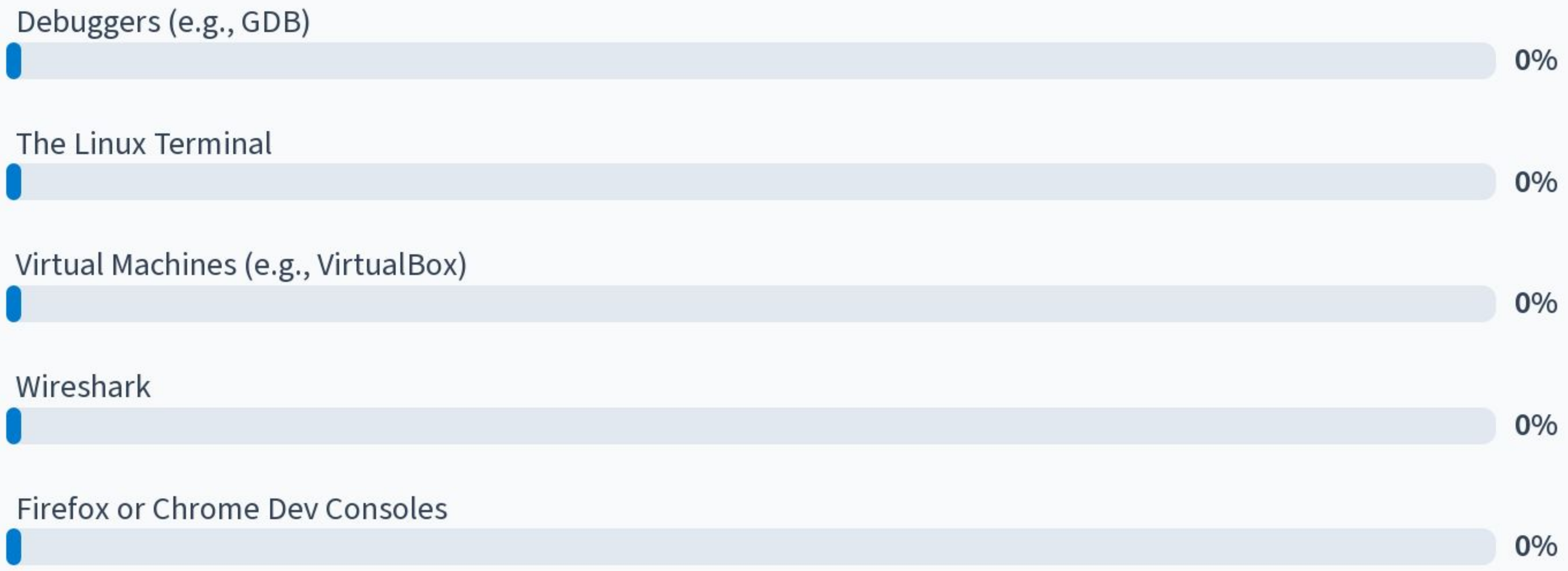
Experience with Cybersecurity



Courses Previously Taken



Experience with Tools



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

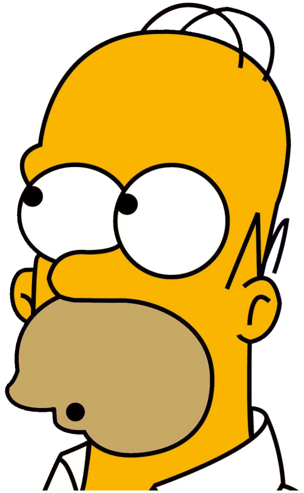
- **Critical Thinking**
  - How to think like an attacker
  - How to reason about threats and risks
  - How to balance security costs and benefits
- **Technical Skills**
  - How to protect yourself
  - How to manage and defend systems
  - How to design and program secure systems
- **Learning to be a security-conscious citizen**
- **Learning to be a** `L337 H4X0R`**... but an *ethical* one!**

Topics

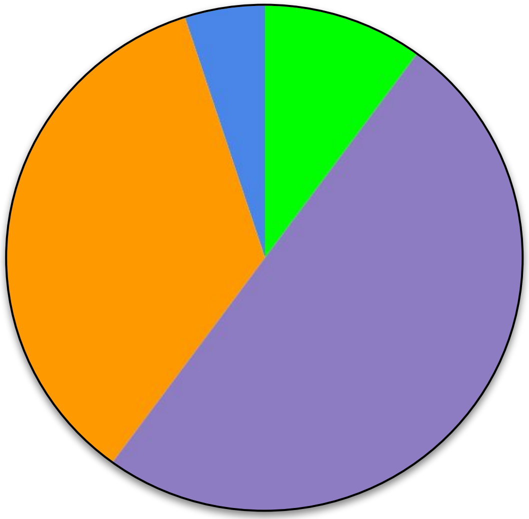
- Course Intro & The Security Mindset — **Week 1**
  - Principles, threat modeling, vulnerabilities, attacking versus defending; VM setup
- P1: Communications Security — **Weeks 2–4**
  - Public- and private-key crypto, digital signatures, authentication, hashes, secure channels
- P2: Application Security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

Common Concerns

- Attendance required? **Yes.**
  - Standard lecture format:
    - ~20 minutes of review
    - ~55 minutes of new material
- Textbook is required? **No.**
  - … but highly recommended!
  - We provide **6 free textbooks** on the site!
- Midterm exam? **No.** Final exam? **Yes.**
  - Covers entire course material
  - Review session as final lecture
  - Similar to in-class and quiz 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 poll exercises

Lecture Quizzes (10%)

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

- Released on **Canvas** 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

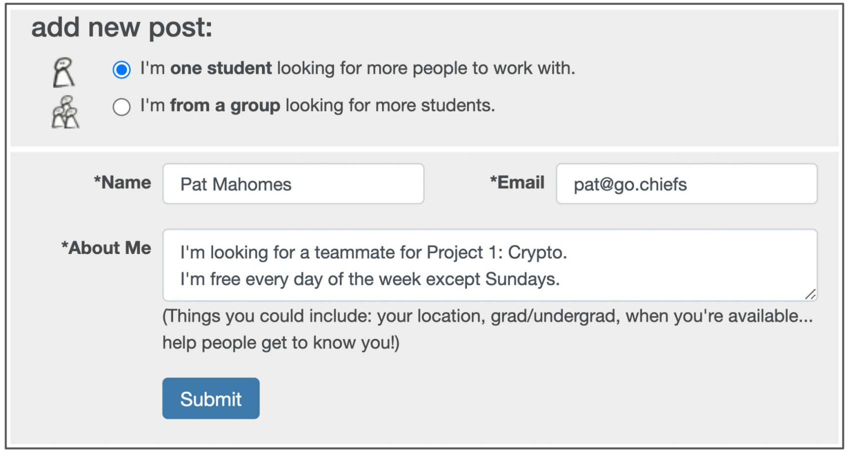
- Where to find and submit?
  - Distributed via **course website** (we'll announce when)
  - Upload your work (**one** per team) as tarball to **Canvas**

Project Teams

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

Search for Teammates! 12/21/22

- 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!



add new post:

[✓ I'm **one student** looking for more people to work with.]

I'm **from a group** looking for more students.

***Name**	Pat Mahomes	***Email**	pat@go.chiefs
***About Me**	I'm looking for a teammate for Project 1: Crypto. I'm free every day of the week except Sundays.		

(Things you could include: your location, grad/undergrad, when you're available... help people get to know you!)

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**

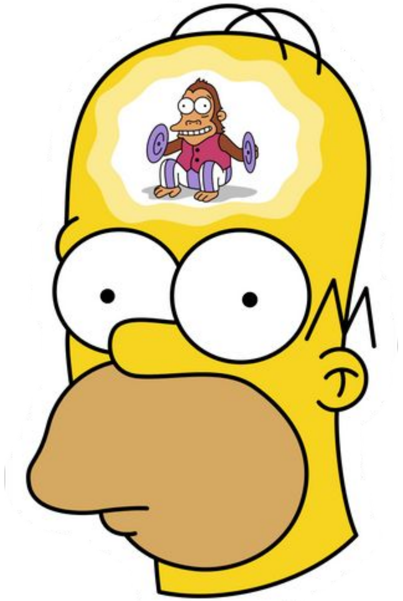
- Violations = misconduct sanctions. **Don't jeopardize your degree!**

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 **Wednesday, December 10**
  - Late exams only for conflicts with other finals

Participation (5%)

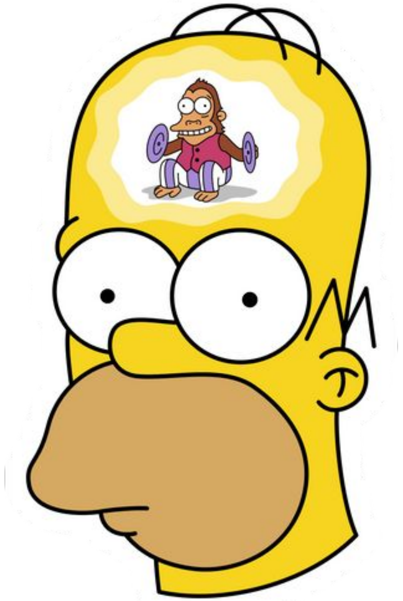
- **Lecture** participation via PollEverywhere:
  - **Three lecture absences allowed** at zero penalty
  - We'll track these internally—no need to notify us
  - Log-in as **your UMAIL** (e.g., `u8675309@utah.edu`)



Participation (5%)

- **Lecture** participation via PollEverywhere:
  - **Three lecture absences allowed** at zero penalty
  - We'll track these internally—no need to notify us
  - Log-in as **your UMAIL** (e.g., `u8675309@utah.edu`)

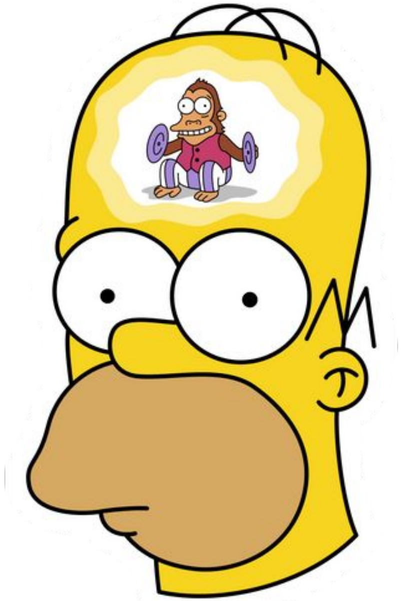
- **Online** participation on course Piazza:
  - Make intellectual contributions to help others learn
  - Collaboration policies apply—**don't share your code!**



Participation (5%)

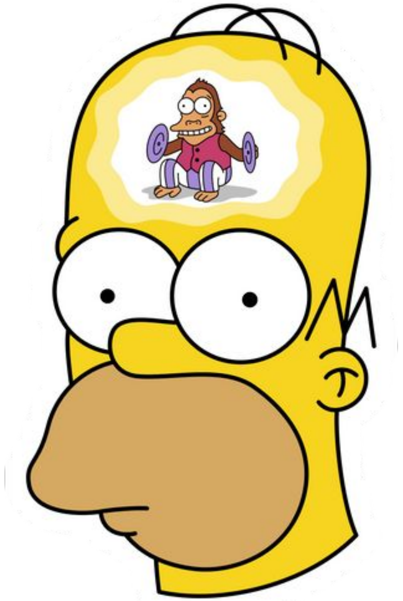
- **Lecture** participation via PollEverywhere:
  - **Three lecture absences allowed** at zero penalty
  - We'll track these internally—no need to notify us
  - Log-in as **your UMAIL** (e.g., `u8675309@utah.edu`)

- **Online** participation on course Piazza:
  - Make intellectual contributions to help others learn
  - Collaboration policies apply—**don't share your code!**
  - **Top-10 answerers** will receive **5pts extra credit**



Participation (5%)

- **Lecture** participation via PollEverywhere:
  - **Three lecture absences allowed** at zero penalty
  - We'll track these internally—no need to notify us
  - Log-in as **your UMAIL** (e.g., `u8675309@utah.edu`)
- **Online** participation on course Piazza:
  - Make intellectual contributions to help others learn
  - Collaboration policies apply—**don't share your code!**
  - **Top-10 answerers** will receive **5pts extra credit**
- How to **lose** points:
  - Frequently missing class, or not contributing online
  - Engaging in disruptive behavior or violating policies



Participation (5%)

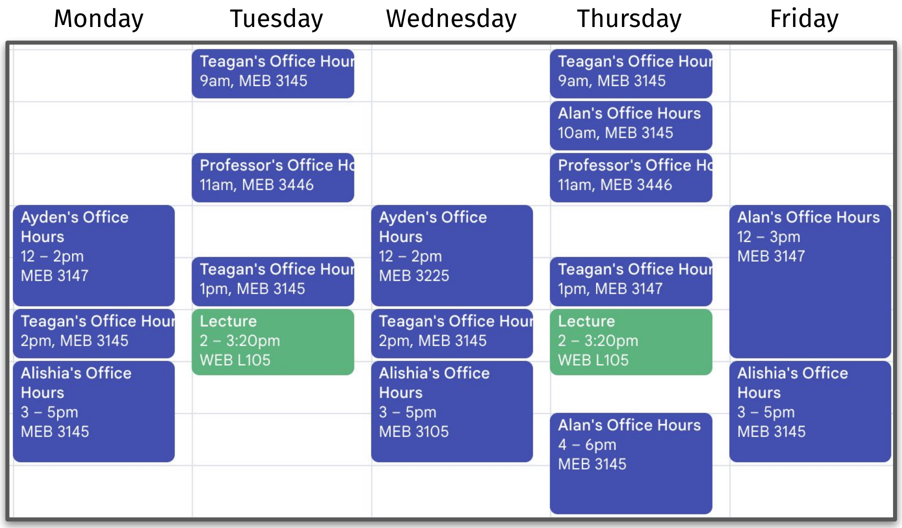
- Lectures where attendance will NOT be graded:
  - **Today's** introductory lecture
  - **Week 7B**, **Week 13A**, and **Week 13B**
    - Instructor out of town
    - Guest lectures planned
  - **Week 14B** (final review lecture)
- Participation total = **23 lectures**
  - **Three absences** dropped
  - We'll track these internally

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!**
  - … and pay attention, and take notes!
  - **Avoid distractions** like surfing the web, Discord, etc.

Office Hours

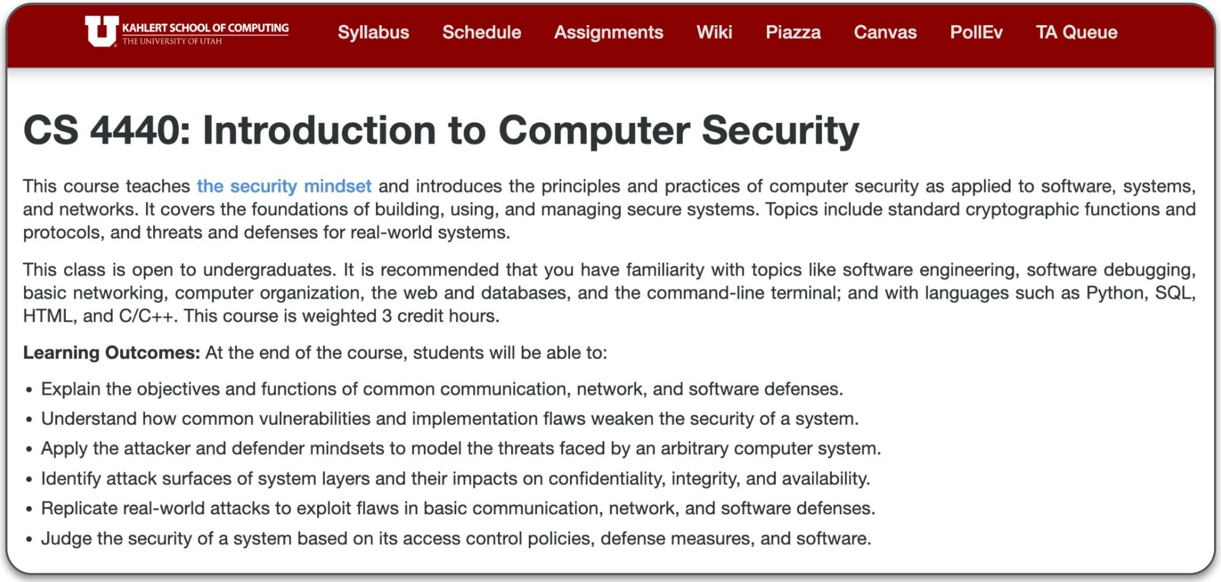
- TA office hours (**23 total hours**)
  - First-come/first-serve via **TA Queue**
  - Help with programming projects
- Professor’s office hours
  - Help understanding lecture material
  - Administrative or grading issues
- Check the office hours calendar!
  - http://cs4440.eng.utah.edu
  - Cancellations announced via **Piazza**



Monday	Tuesday	Wednesday	Thursday	Friday
	Teagan's Office Hour 9am, MEB 3145		Teagan's Office Hour 9am, MEB 3145	
			Alan's Office Hours 10am, MEB 3145	
	Professor's Office Hc 11am, MEB 3446		Professor's Office Hc 11am, MEB 3446	
Ayden's Office Hours 12 – 2pm MEB 3147		Ayden's Office Hours 12 – 2pm MEB 3225		Alan's Office Hours 12 – 3pm MEB 3147
	Teagan's Office Hour 1pm, MEB 3145		Teagan's Office Hour 1pm, MEB 3147	
Teagan's Office Hour 2pm, MEB 3145	Lecture 2 – 3:20pm WEB L105	Teagan's Office Hour 2pm, MEB 3145	Lecture 2 – 3:20pm WEB L105	
Alishia's Office Hours 3 – 5pm MEB 3145		Alishia's Office Hours 3 – 5pm MEB 3105		Alishia's Office Hours 3 – 5pm MEB 3145
			Alan's Office Hours 4 – 6pm MEB 3145	

Communication

- **Course website:** your go-to resource for all things CS 4440
  - 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.

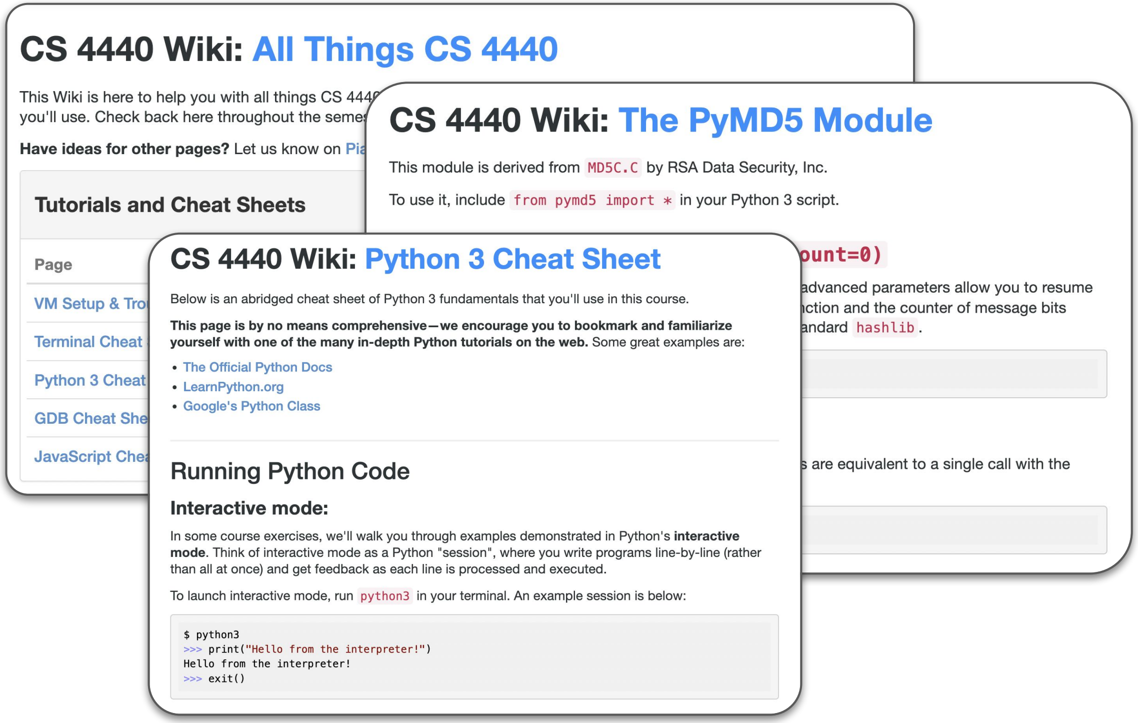
This class is open to undergraduates. It is recommended that you have familiarity with topics like software engineering, software debugging, basic networking, computer organization, the web and databases, and the command-line terminal; and with languages such as Python, SQL, HTML, and C/C++. This course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

The CS 4440 Wiki

- Our aim is to lower the overall learning curve
- Resources to help you:
  - Tutorials
  - Cheat Sheets
  - Software documentation
- Many more resources added since last Fall


CS 4440 Wiki: All Things CS 4440

This Wiki is here to help you with all things CS 444 you'll use. Check back here throughout the semes

Have ideas for other pages? Let us know on Pia

Tutorials and Cheat Sheets

Page

VM Setup & Tro

Terminal Cheat

Python 3 Cheat

GDB Cheat She

JavaScript Chea

CS 4440 Wiki: The PyMD5 Module

This module is derived from `MD5C.C` by RSA Data Security, Inc.

To use it, include `from pymd5 import *` in your Python 3 script.

ount=0)

advanced parameters allow you to resume nction and the counter of message bits andard `hashlib`.

s are equivalent to a single call with the

CS 4440 Wiki: Python 3 Cheat Sheet

Below is an abridged cheat sheet of Python 3 fundamentals that you'll use in this course.

This page is by no means comprehensive—we encourage you to bookmark and familiarize yourself with one of the many in-depth Python tutorials on the web. Some great examples are:

- The Official Python Docs
- LearnPython.org
- Google's Python Class

Running Python Code

Interactive mode:

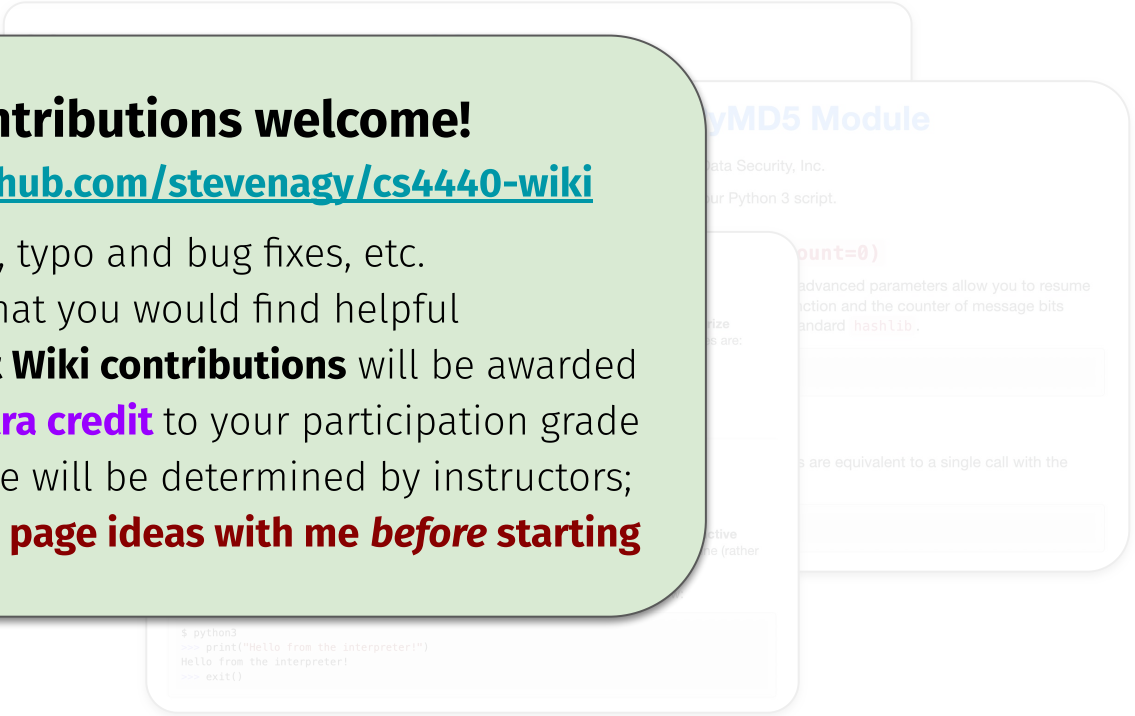
In some course exercises, we'll walk you through examples demonstrated in Python's **interactive mode**. Think of interactive mode as a Python "session", where you write programs line-by-line (rather than all at once) and get feedback as each line is processed and executed.

To launch interactive mode, run `python3` in your terminal. An example session is below:

```
$ python3
>>> print("Hello from the interpreter!")
Hello from the interpreter!
>>> exit()
```

The CS 4440 Wiki

- Our aim is to lower the overall learn
- Resources to
  - Tutorials
  - Cheat She
  - Software c
- Many more r added since l



yMD5 Module

ata Security, Inc.

ur Python 3 script.

ount=0)

advanced parameters allow you to resume
ction and the counter of message bits
andard hashlib.

s are equivalent to a single call with the

```
$ python3
>>> print("Hello from the interpreter!")
Hello from the interpreter!
>>> exit()
```

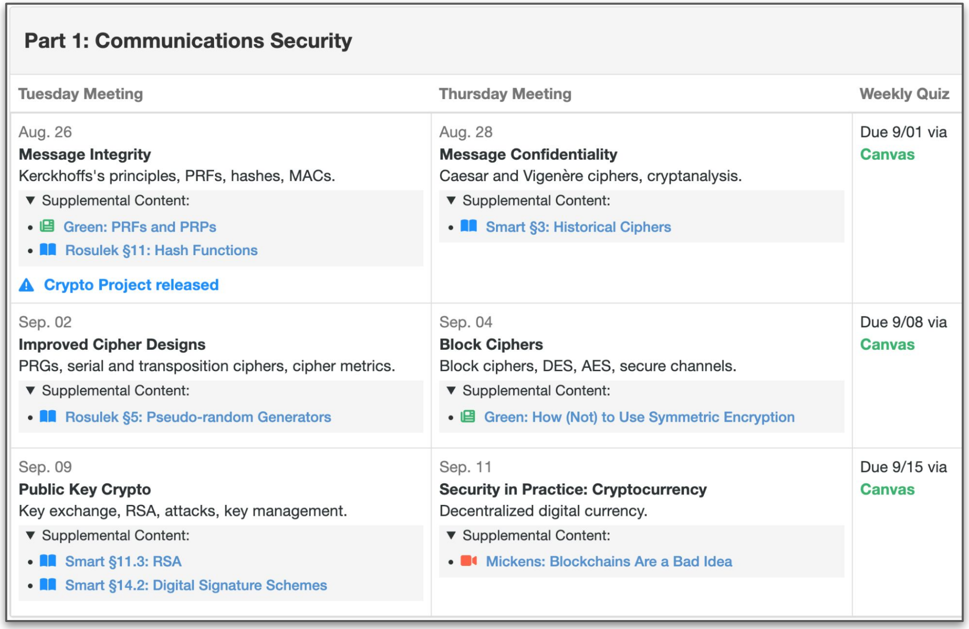
Contributions welcome!

https://github.com/stevenagy/cs4440-wiki

- Page ideas, typo and bug fixes, etc.
- Tutorials that you would find helpful
- **Significant Wiki contributions** will be awarded **1 point extra credit** to your participation grade
- Significance will be determined by instructors; must **clear page ideas with me *before* starting**

Supplemental Content

- To further help you learn, we've provided **supplemental content** relevant to every lecture topic
  - Short blog posts
  - Free textbook chapters
  - Fun podcasts or videos
- **Totally optional**—not required
  - … though we do recommend them as additional resources to lectures!
- To access, click the drop-down "▸" button beside each lecture

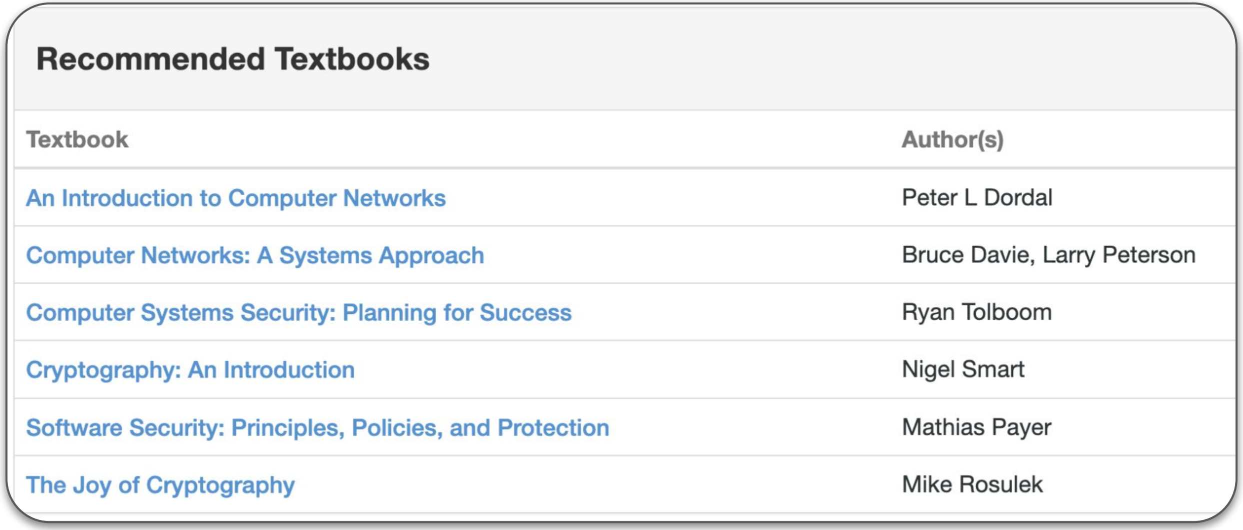


Part 1: Communications Security

Tuesday Meeting	Thursday Meeting	Weekly Quiz
Aug. 26 **Message Integrity** Kerckhoffs's principles, PRFs, hashes, MACs. ▼ Supplemental Content: • Green: PRFs and PRPs • Rosulek §11: Hash Functions ⚠ **Crypto Project released**	Aug. 28 **Message Confidentiality** Caesar and Vigenère ciphers, cryptanalysis. ▼ Supplemental Content: • Smart §3: Historical Ciphers	Due 9/01 via **Canvas**
Sep. 02 **Improved Cipher Designs** PRGs, serial and transposition ciphers, cipher metrics. ▼ Supplemental Content: • Rosulek §5: Pseudo-random Generators	Sep. 04 **Block Ciphers** Block ciphers, DES, AES, secure channels. ▼ Supplemental Content: • Green: How (Not) to Use Symmetric Encryption	Due 9/08 via **Canvas**
Sep. 09 **Public Key Crypto** Key exchange, RSA, attacks, key management. ▼ Supplemental Content: • Smart §11.3: RSA • Smart §14.2: Digital Signature Schemes	Sep. 11 **Security in Practice: Cryptocurrency** Decentralized digital currency. ▼ Supplemental Content: • Mickens: Blockchains Are a Bad Idea	Due 9/15 via **Canvas**

Free Online Textbooks

- We now make available several **freely-distributed textbooks** via the **Wiki**
  - Some textbook chapters are referenced as lecture-relevant Supplemental Content
  - Also **totally optional**—they are meant only as additional resources to help you learn



Recommended Textbooks

Textbook	Author(s)
An Introduction to Computer Networks	Peter L Dordal
Computer Networks: A Systems Approach	Bruce Davie, Larry Peterson
Computer Systems Security: Planning for Success	Ryan Tolboom
Cryptography: An Introduction	Nigel Smart
Software Security: Principles, Policies, and Protection	Mathias Payer
The Joy of Cryptography	Mike Rosulek

Summary

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

Piazza .... questions, discussion, announcements

PollEverywhere .... lecture participation

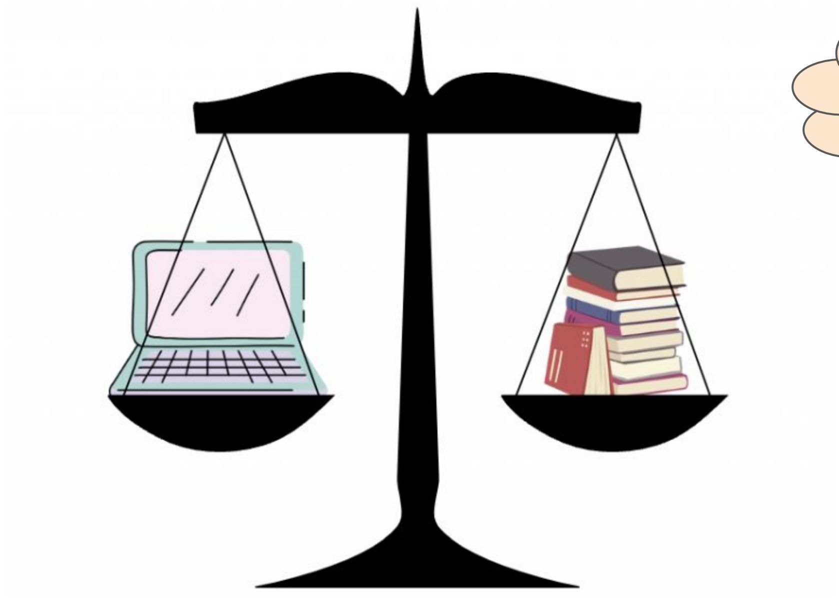
Canvas .... quizzes, project submission, course gradebook

Instructor email (snagy@cs.utah.edu**)** .... administrative issues

Questions?

The Security Mindset

What does Computer Science impact?



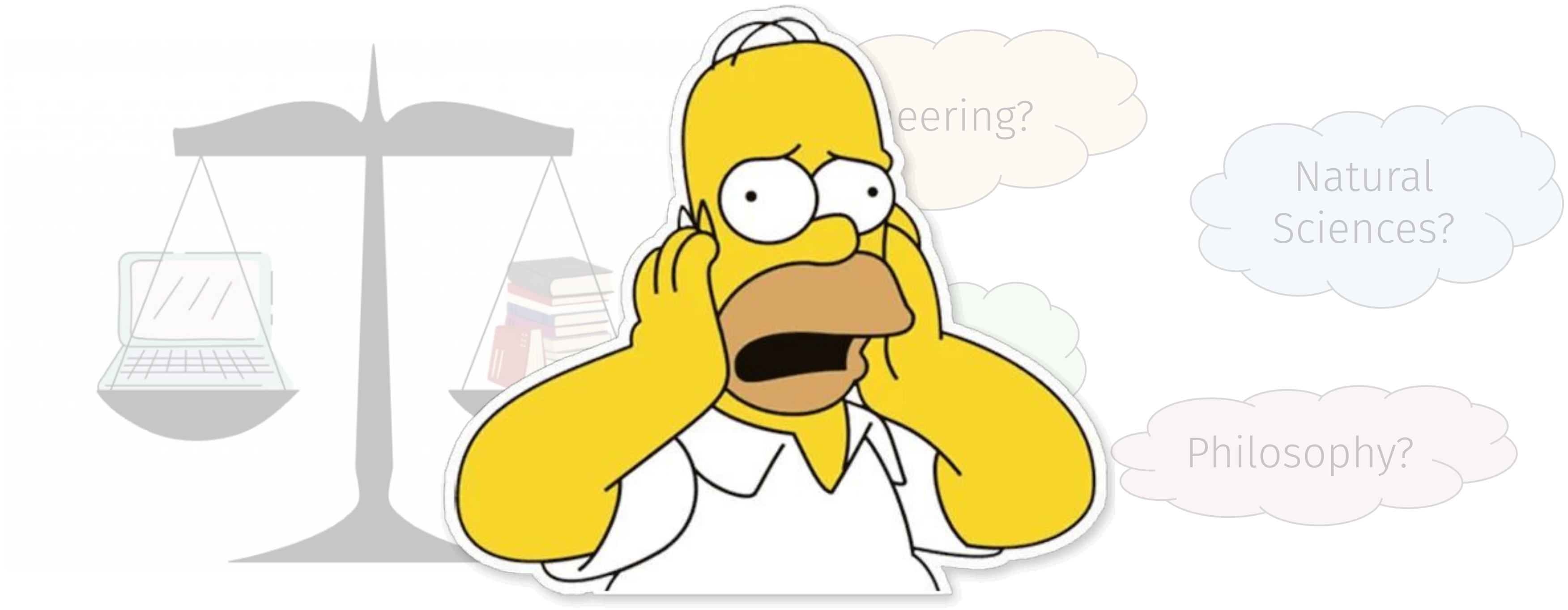
Engineering?

Natural Sciences?

Math?

Philosophy?

What does Computer Security impact?



Computers Nowadays…

- … are like wheelbarrows of orangutans
  - Think of every app, user, file as an orangutan
- **What could go wrong?**

Computers Nowadays…

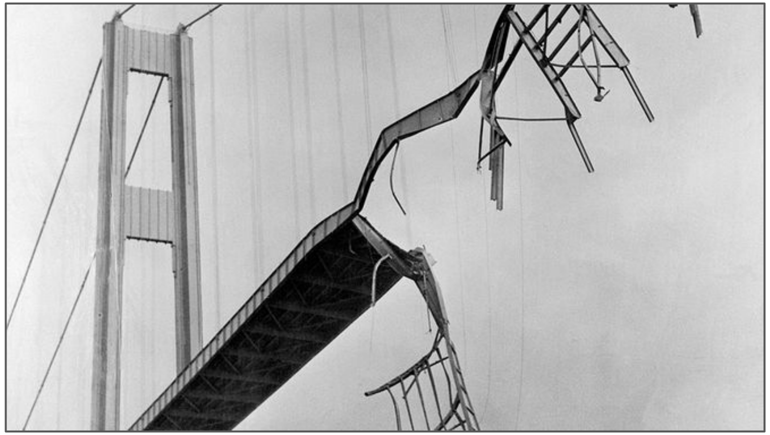
- … are like wheelbarrows of orangutans
  - Think of every app, user, file as an orangutan

- **What could go wrong?**
  - One might try to **throw another one off**
  - One is probably trying **to spy on another**
  - One **will bite you** and **steal your credit card**

Computers Nowadays…

- … are like wheelbarrows of orangutans
  - Think of every app, user, file as an orangutan
- **What could go wrong?**
  - One might try to **throw another one off**
  - One is probably trying **to spy on another**
  - One **will bite you** and **steal your credit card**
- **Call to action:** let's adjust our thinking based on the **possibility** of such risks
  - How we design new systems
  - How we permit user interaction
  - How we store sensitive information

What’s the difference?



Reliability **does not equal** **Security**

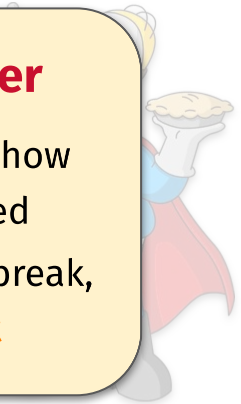
The Security Mindset

Attacks

Defenses

The Security Mindset

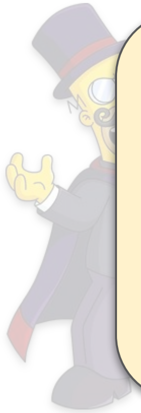
Attacks



Thinking like an attacker

- Understand **techniques** for how security can be circumvented
- Look for ways security can break, **not why (you think) it won't**

The Security Mindset

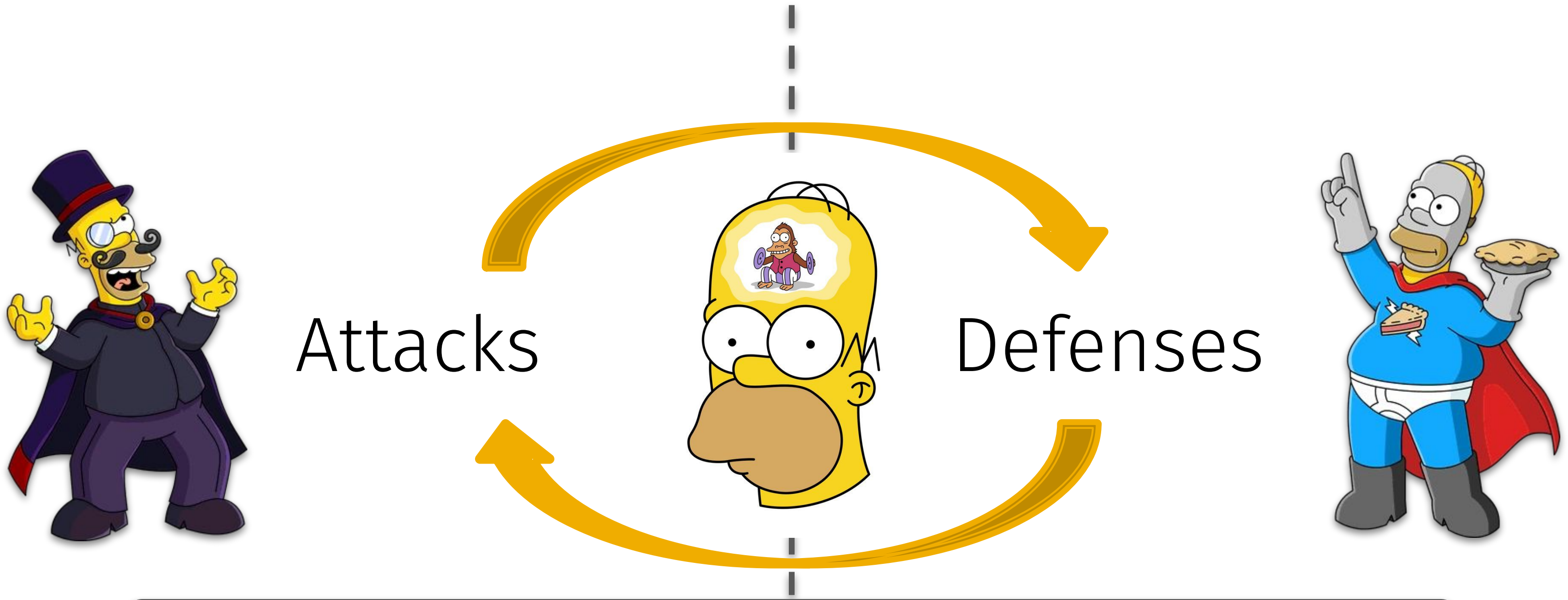


Thinking like a defender

- Know **what** you're defending, **whom** you're defending against
- Weigh benefits versus costs
- Embrace **"rational paranoia"**

Defenses

The Security Mindset



The Security Mindset: thinking as **both the attacker and defender**!

But first… know thy Adversary!

- Computer security studies how systems behave in the presence of an **adversary**
  - **???**

But first... know thy Adversary!

- Computer security studies how systems behave in the presence of an **adversary**
  - Independent / hobbyist hackers
  - “Script kiddies”
  - Cyber-criminal gangs
  - Nation-state government hackers
  - Disgruntled students (or professors)

- **Definition:** an intelligence that **actively tries to cause the system to misbehave**.

But first... know thy Adversary!

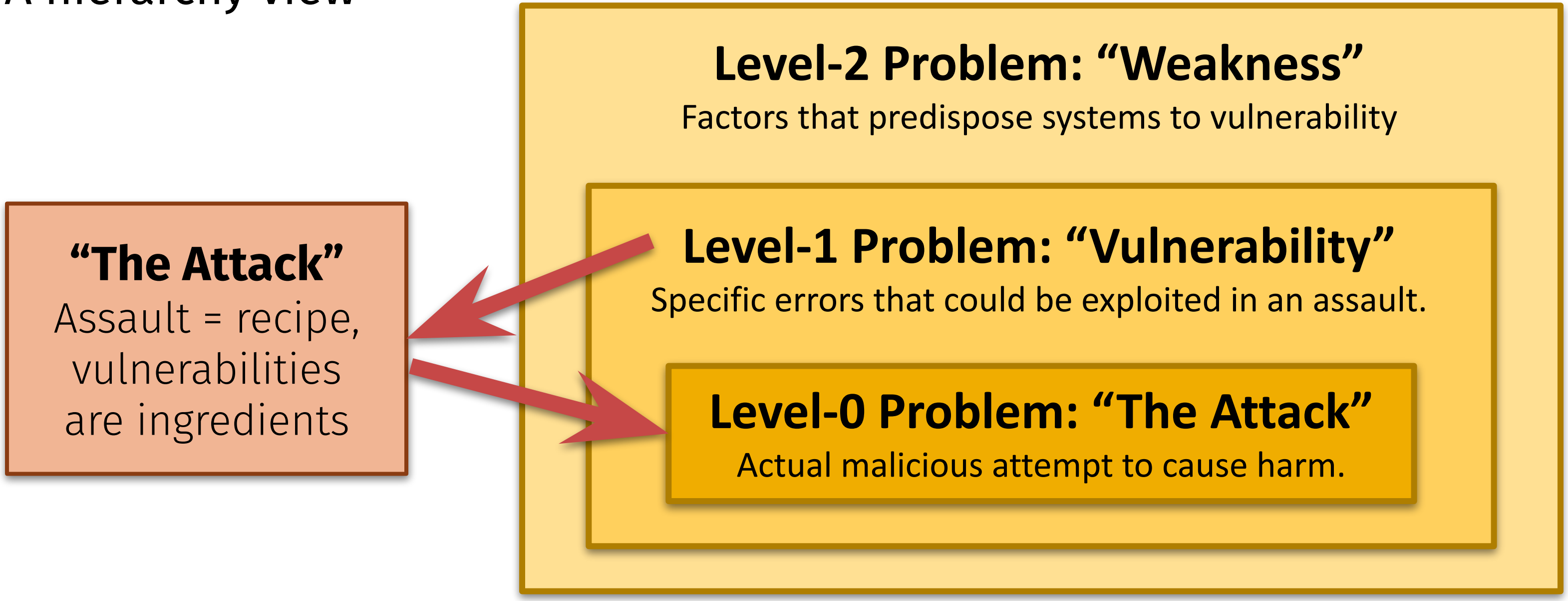
- **Motives?**
  - ???
- **Degree of access?**
  - ???
- **Capabilities?**
  - ???

But first... know thy Adversary!

- **Motives?**
  - Disruption
  - Espionage
  - Money
- **Degree of access?**
  - Physical access
  - Root privileges
- **Capabilities?**
  - Denial of service
  - Code execution

Attacks on Computer Systems

- A hierarchy view


"The Attack"
Assault = recipe, vulnerabilities are ingredients
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.

Why study attacks?

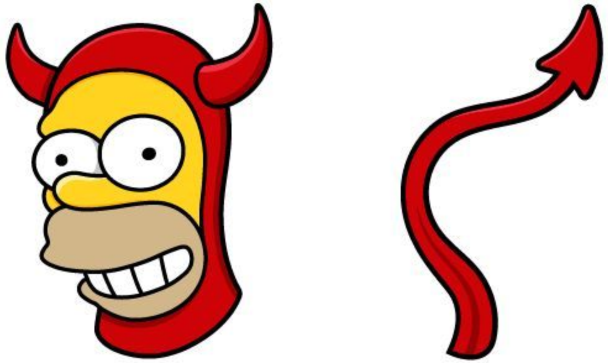
- **Why?**

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

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

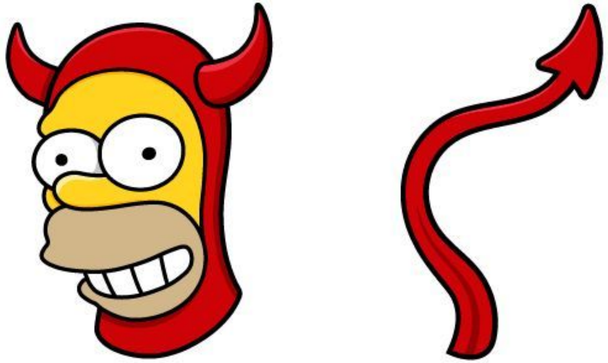


Thinking like an Attacker

- 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

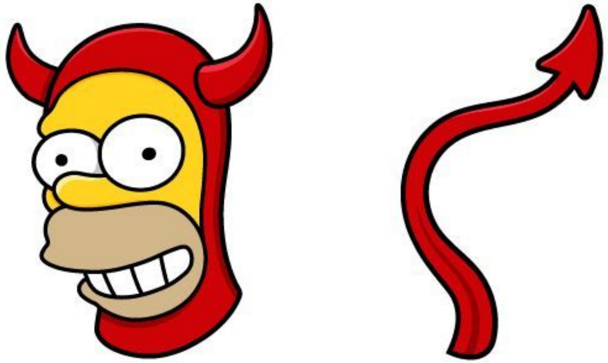
Practice thinking as an attacker:

For **each system you interact with**, think about what it means for it to be **secure**, and **imagine how it could be exploited**



Thinking like an Attacker

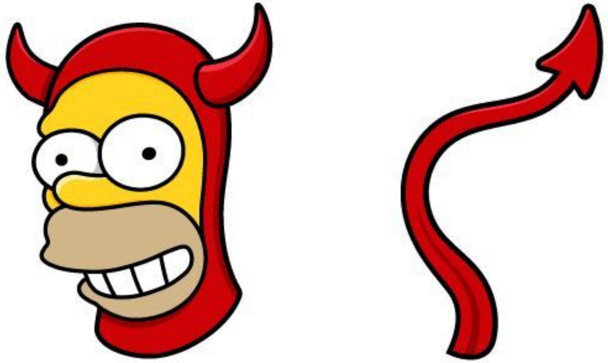
- **Exercise:** name some **security systems** that you interact with in everyday life



Thinking like an Attacker

- **Exercise:** name some **security systems** that you interact with in everyday life

- **Example:** the lock to **Prof. Nagy's office**
  - Breaking-in after hours to alter your grade?
  - **Weakest links?**
  - **Assumptions?**
  - **Circumventing?**



Thinking as a Defender

- **Security Policy**
  - What resources are we protecting?
  - What properties are we enforcing?
- **Threat Model**
  - Who will attack us? Capabilities? Motivations?
  - What types of attacks must we try to prevent?
- **Assessing Risk**
  - What are the system's weaknesses?
  - How will successful attacks hurt us?
- **Assessing Likelihood**
  - Countermeasures
  - Costs vs. benefits?
  - Technical vs. nontechnical?

Thinking as a Defender

- **Security Policy**
  - What resources are we protecting?
  - What properties are we enforcing?
- **Threat Model**
  - Who will attack us? Capabilities? Motivations?
  - What types of attacks must we try to prevent?
- **Assessing Risk**
  - What are the system's weaknesses?
  - How will successful attacks hurt us?
- **Assessing Likelihood**
  - Countermeasures
  - Costs vs. benefits?
  - Technical vs. nontechnical?

Rational paranoia:

Thinking **rigorously**, yet **realistically** about risk!

Security Policies

- What **resources** are we trying to protect?
  - **???**

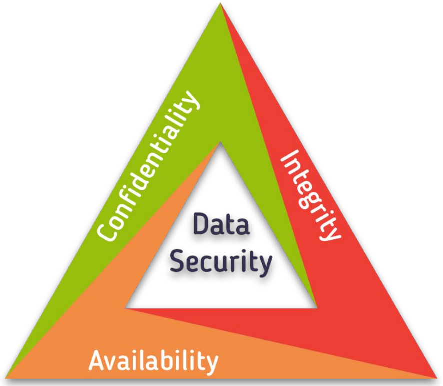
Security Policies

- What **resources** are we trying to protect?
  - Files
  - Programs
  - User data
  - NFTs?

- What **properties** are we trying to enforce?
  - ???

Security Policies

- What **resources** are we trying to protect?
  - Files
  - Programs
  - User data
  - NFTs?
- What **properties** are we trying to enforce?
  - Confidentiality
  - Integrity
  - Availability
  - Privacy
  - Authenticity



Threat Models

- Who are our **adversaries**?
  - Motives?
  - Capabilities?
  - Level of access?
- What **types of attacks** must we prevent?
  - Think like the attacker!
- **Limits:** kinds of attacks **we need ignore**?
  - Unrealistic versus unlikely

Assessing Risk

- **Remember: rational paranoia**
- How will a breach **harm us**?
  - **Direct harm:**
    - **???**

Assessing Risk

- **Remember: rational paranoia**
- How will a breach **harm us**?
  - **Direct harm:**
    - Money
    - Intellectual property
    - Physical safety
  - **Indirect harm:**
    - **???**

Assessing Risk

- **Remember: rational paranoia**

- How will a breach **harm us**?
  - **Direct harm:**
    - Money
    - Intellectual property
    - Physical safety
  - **Indirect harm:**
    - Reputation
    - Future business
    - Well being

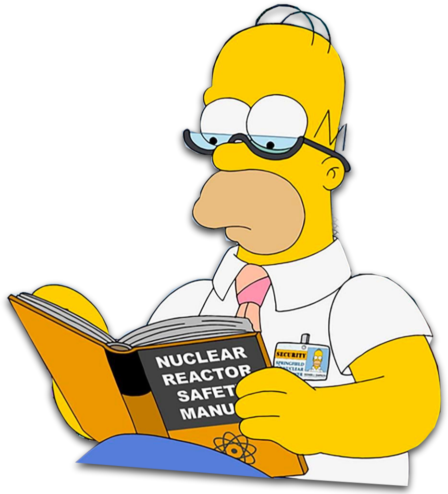
- How **likely** are these harms?
  - Attempts vs. successful attacks?

Countermeasures

- **Technical countermeasures**
  - Bug fixes, more crypto, re-architecting, etc.
- **Non-technical countermeasures**
  - Law, policy (government, institutional)
  - Procedures, training, auditing, incentives, etc.

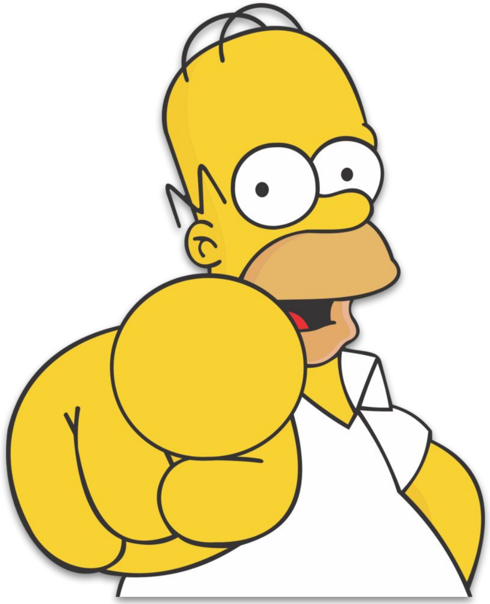
Costs of Security

- **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



Class Exercise

- **Using a credit card safely?**
  - Assets?
  - Adversaries?
  - Risk assessment?
  - Countermeasures?
  - Defense costs/benefits?



Security through... obscurity?

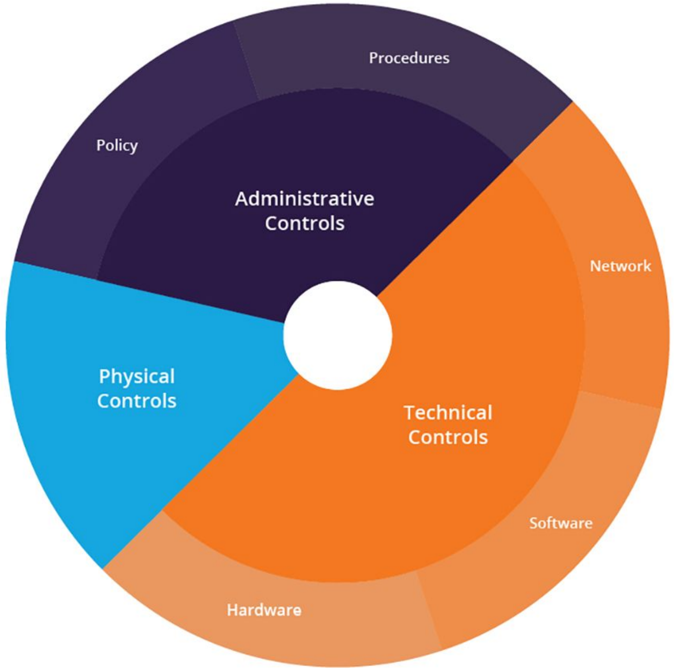
- **Common mistakes:**
  - Convincing yourself that a system is **already secure** in its current form
  - Convincing yourself a system is safe because attacker **won't know XYZ**

Security through... obscurity?

- **Common mistakes:**
  - Convincing yourself that a system is **already secure** in its current form
  - Convincing yourself a system is safe because attacker **won't know XYZ**
- **Better approach:**
  - **Limit key assumptions** that security of your system depends upon
  - Identify **any components exposed** to attackers and their weaknesses
  - Assume **attacker knows everything** but a small bit of data (e.g., a key)

Other Defense Principles

- **Defense-in-Depth**
  - Multiple layers of safeguards
  - Physical, technical, administrative
- **Component Diversity**
  - More moving parts = harder to attack
  - Conversely, harder to secure
- **Maintainability**
  - Minimize maintainer workload
  - Make fixes easy/fast to deploy



Testing Security

- Testing against **requirements**
  - How must the system behave?
  - What threats must be mitigated?
- **Adversarial testing** (my work)
  - Black-box testing
  - White-box testing
  - Gray-box testing
- **Example:** airport security

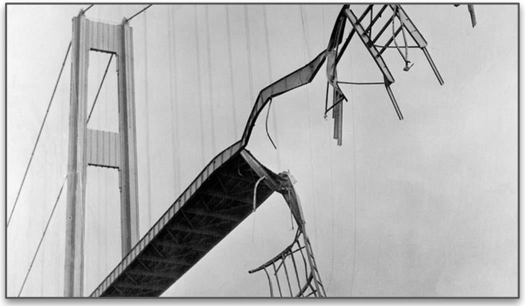
Red Team agents use disguises, ingenuity to expose TSA vulnerabilities

Learning from Failures

- **… a time-honored engineering practice!**
  - Especially important in security

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

- **What can failures teach us?**
  - New kinds of attacks
  - New kinds of defenses



Questions?

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**

Questions?

Next time on CS 4440...

Python Tutorial and Course VM Setup

Bring your laptops... and pre-download your VM image!