Lecture 1: Introduction
First off

- These are some crazy times. We’re going to do the best we can. You’re doing the best you can. We’ll get through it.

- Zoom and recording, managing interaction

- Grade status (letter vs P/NP)

- Some quick polls to see where we are
Course Information

▪ Professor: Stefan Savage
  - Lectures: Tu/Th 12:30-1:50 (via Zoom)
  - Discussion: W 3-3:50 (via Zoom)
  - Office Hours: F 11am-12pm (via Zoom)

▪ TAs
  - Ariana Mirian
  - Riley Hadden
  - Shuyi Ni
  - Simrandeep Singh
  - Xiaohan Fu

▪ Tutors
  - Patrick Liu
  - Thant Zaw

▪ Piazza
  - https://piazza.com/ucsd/spring2020/cse127/home

▪ Course Web Page
  - https://cseweb.ucsd.edu/classes/sp20/cse127-a/ (being updated)
About me

- I work at the intersection of computer security, networking and operating systems

- **Research**
  - I’m co-director of the Center for Networked Systems (CNS) on campus.
  - Lots of work on security measurement, ecrime, security of cyberphysical systems (esp cars and planes)

- **Policy**
  - National Research Council’s Cybersecurity Research group
  - Institute for Defense Analysis’ ISAT advisory group
  - National Science Foundation CISE Advisory Committee
  - Way too much time in D.C.; steering committee ACM Law+CS
  - I co-teach the graduate cybersecurity policy class in GPS

- **Industry**
  - Asta Networks (defunct anti-DDoS company)
  - Netsift (UCSD-originated worm defense company) -> Cisco
  - A fair amount of consulting... (data breaches, LE, Web companies)
Course Objectives

- A solid foundation of security concepts, backed by concrete examples

- Security mindset
  - How to think like an attacker/security engineer
  - Looking beyond the system’s intended functionality, to what it can be made to do

- Understanding how things work, how they break, and how to fix them
  - Technical details of vulnerabilities, attacks, and defenses

- Becoming a better engineer
  - Minimize number and severity of vulnerabilities you create
  - Understand the causes and impact of vulnerabilities that you are alerted to
  - Properly address vulnerabilities that are identified
What you will need to know

- C programming and bits of assembly
- OS (memory protection, address translation, threads)
- Some architecture (caches/TLBs) and networking (packets, connections)

- I’ll try to touch on some of these things, but you need to be prepared to learn on your own
  - In particular, we aren’t teaching any C/asm programming
Course Material

- Textbook: no mandatory textbook to buy
- Readings from:
  - Security Engineering by Ross Anderson (free online)
- Articles & Videos
  - Additional web-hosted content to be assigned
- Slides
  - Based on slides and notes from Kirill Levchenko, Alex Gantman, Deian Stefan, Nadia Heninger, Alex Dent, Vitaly Shamtikov, Robert Turner, and a host of others
Grading

- Homework assignments & projects: 30%
- Midterm: 30%
  - If midterm grade > 0
  - then midterm = max(midterm, final)
  - else midterm = 0
- Final: 40%
Rules

- Homework and assignments are *due on the date and time indicated*
  - May work in groups of 2 or individually (note: fate sharing for groups)
  - 7 *late days* to be allocated as you like, but that’s it

- **Regrades should be the exception**
  - We reserve the right to completely regrade your assignments

- **No Cheating**
  - Read and understand UC San Diego policy
    - [http://academicintegrity.ucsd.edu](http://academicintegrity.ucsd.edu)
  - Cheating includes not doing the assignment yourself, providing answers to others, etc.
  - Not ok to copy, translate, paraphrase, edit, etc. someone else’s work
  - If you are not sure if something is cheating, either ask or assume its cheating
  - We will report *all* suspected cheating cases to academic integrity
Ethics

- In this class you will learn how to attack the security of computer systems (and some physical systems)

- We learn attacks because it is needed to understand how to defend them

- You have an obligation to use this knowledge ethically
  - You **may not** attack others
    - In addition to unethical, may be a felony
  - Many good *legitimate* hacking challenges
    - [http://overthewire.org/wargames/](http://overthewire.org/wargames/) (wargames)
    - [https://challenges.re/](https://challenges.re/) (reverse engineering challenges)
    - [https://ctftime.org/ctfs](https://ctftime.org/ctfs) (Capture the Flag competitions)
18 U.S. CODE § 1030 - FRAUD AND RELATED ACTIVITY IN CONNECTION WITH COMPUTERS

- Whoever intentionally accesses a computer without authorization or exceeds authorized access, and thereby obtains information from any protected computer...

- Complex, but serious jail time...
What is security?

- **Merriam-Webster online dictionary:**
  
  Function: *noun*
  
  1: the quality or state of being **secure** : as *a*: freedom from danger: **SAFETY**  
  *b*: freedom from fear or anxiety  
  *c*: freedom from the prospect of being laid off <job security>

  2 *a*: something given, deposited, or pledged to make certain the fulfillment of an obligation  
  *b*: **SURETY**

  3: an instrument of investment in the form of a document (as a stock certificate or bond) providing evidence of its ownership

  4 *a*: something that **secures** : **PROTECTION**  
  *b* (1): measures taken to guard against espionage or sabotage, crime, attack, or escape  
  (2): an organization or department whose task is security
Computer security?

- Most of computer science is about providing *functionality*:
  - UX/UI
  - Software Architecture
  - Algorithms
  - Operating Systems/Networking/Databases
  - Compilers/PL
  - Microarchitecture
  - VLSI/CAD

- Computer security is *not* about functionality

- It is about how the embodiment of functionality behaves *in the presence of an adversary*

- Holistic property
  - “Software security is about integrating security practices into the way you build software, not integrating security features into your code” – Gary McGraw
Thinking like an attacker

▪ Look for the weakest links

▪ Identify the **assumptions** that security depends on. Are they guaranteed to always be true? Can you make them false?

▪ Think outside the box. Ignore the limited worldview of the system’s designers

▪ This is something you can start doing now and all the time. When you interact with a system (computerized or not) think about what that system depends on and how it might be exploited
History: two competing philosophies about securing systems

- **Binary** model  [secure vs insecure]
  - Traditional crypto and trustworthy systems
  - Assume adversary limitations X and define security policy Y
  - If Y cannot be violated without needing X then system is secure, else insecure
  - You know people are invoking some version of this model if they say “proof of security”, “secure by design” “trustworthy systems”

- **Risk management** model. [more secure vs less secure]
  - Most commercial software development
    (and much real-world security... e.g., terrorism)
  - Try to minimize biggest risks and threats
  - Improve security where most cost effective (expected value)
  - You know people are in this model if they use the words “risk”, “mitigation”, “defenses”, “resilience”, etc.
Classic example (binary model): perfect substitution cipher

\[ p_1 p_2 p_3 \ldots p_n \oplus b_1 b_2 b_3 \ldots b_n \]
\[ c_1 c_2 c_3 \ldots c_n \]

- Invited by combination of Vernam & Mauborgne (~1919)
- Choose a string of random bits the same length as the plaintext, XOR them to obtain the ciphertext.
- **Perfect Secrecy** (proved by Claude Shannon)
  - Probability that a given message is encoded in the ciphertext is *unaltered* by knowledge of the ciphertext
  - Proof: Give me any plaintext message and any ciphertext and I can construct a key that will produce the ciphertext from the plaintext. Zero information in ciphertext
Classic example (risk management): Concrete barricades

- Prevent incursion by car bombers
Problems with the binary model: Abstract design != Concrete artifact

- Many assumptions are brittle in real systems
  - Real artifacts fragile, imperfect, have bugs/limitations
    - Don’t do precisely what spec says or documentation says
    - E.g., what is an integer?
  - Large gap between abstraction and implementation
    - Example: secret key in chip used to decrypt data; key leaks via the current the chip draws for different operations

From Paul Kocher

Courtesy Oswald
Problems with the binary model: security evolution

- As engineers, we often delude ourselves into thinking that we understand our own creations
  - or that we can create complex systems to do only what we meant them to do

- But ... nobody knows how these systems really work
  - Complexity of computer systems is approaching complexity of biological organisms
    - 3 billion base pairs in human genome
    - 10+ billion transistors in modern CPUs

- Complex systems co-evolve with attacks against them
  - How we use systems, how we depend on them and how they might be attacked – all change over time
  - Systems deemed secure today may not be resilient to new threats
Problems with the risk management model:
One vulnerability can matter…
Problems with the risk management model: You never win

- Creates arms race – forced co-evolution

  Adversary invents new attack

  Defender creates new defense

- The best you can hope for is stalemate
Problems with the risk management model: How to measure

- It's fine to say security is a spectrum, but how to evaluate risk or reward?
  - How many units of security does your anti-virus product give you?

- Big question: how do we measure security?
  - How is this different from car safety?
  - Or drug safety?
Key meta issues in Security

- Policy
- Assets, Risks & Threats
- Value
- Protection
- Deterrence
- Identity & Reputation
Policy

- What is a bad thing?

- Remarkably tricky to define for known threats
  - The software on your computer likely has 100s of security options... How should you set them?
  - What might be a good security policy for who gets to access faculty salary data?

- Even harder for unknown threats
  - SPAM

- Should a highly privileged user have more rights on a system or less?
Assets, Risks & threats

- **Assets**
  - What you want to protect

- **Threats**
  - Actions likely to cause damage, harm or loss
  - Includes both kinds of attacks (e.g., virus, social engineering) and kinds of attackers (e.g., script kiddie vs state sponsored actor)
  - Need to reason about requirements of each threat (what capabilities does the attacker need) and what it enables (what harm might come? What motivations might drive such a threat)

- **Risk**
  - What is the potential likelihood of a something bad happening (i.e., what threats are likely)

- These tend to be well formalized in some communities (e.g. finance sector) and less in others (e.g. energy sector)

- We'll talk more about threat models next class...
Value

- What is the cost if the bad thing happens?
- What is the cost of preventing the bad thing?

- Example: credit card fraud
  - Who pays if someone steals your credit card # and buys a TV with it?

- Example: Permissive Action Links for nuclear weapons
Protection

- The mechanisms used to protect resources against threats
  - This is most of academic and industrial computer security

- Many classes of protections
  - Cryptographic protection of data
  - Software guards
  - Communication guards
  - User interface design (protect user against own limitations)

- Can be either proactive or reactive
Deterrence

- There is some non-zero expectation that there is a future cost to doing a bad thing
  - i.e. going to jail, having a missile hit your house, having your assets seized, etc
  - Criminal cost-benefit: \( M_b + P_b > O_{cp} + O_{cm} P_a P_c \) [Clark&Davis 95]
    - \( M_b \): Monetary benefit
    - \( P_b \): Psychological benefit
    - \( O_{cp} \): Cost of committing crime
    - \( O_{cm} \): Monetary cost of conviction
    - \( P_a \): Probability of getting caught
    - \( P_c \): Probability of conviction

- Need meaningful forensic capabilities
  - Audit actions, assign identity to evidence, etc
  - Must be cost effective relative to positive incentives
Switching gears: Identity

- Identity is implicit in virtually all security questions.... but we rarely think about it much

- We have strong intuitions however
  - How do you feel about “Black Unicorn” the cypherpunk?
  - How about A.S.L. von Bernhardi the investment banker?
Identity

- What is it?
  - One def: *The distinct personality of an individual regarded as a persisting entity; individuality* (courtesy Black Unicorn)
  - Another: *A unique identifier – distinguishing mark* (courtesy A.S.L. von Bernhardi)

- Is there a difference between an **identity** and an **identifier**?
  - Yes. Identifier is a concrete object (e.g., SSN, email address), identity is abstract
  - Allows naming; to establish an assertion about reputation

- Reputation?
  - A specific characteristic or trait ascribed to a person or thing: e.g., *a reputation for paying promptly*
  - Potentially a predictor of behavior, a means of valuation and as a means for third-party assessment

- Value comes from binding reputation and identifiers
- But how to make this binding?
Due diligence and trust

- **Due diligence**
  - Work to acquire multiple independent pieces of evidence establishing identity/reputation linkage; particularly via direct experience
  - Expensive

- **Trust**
  - *Reliance on something in the future; hope*
  - **Allows cheap form of due-diligence**: third-party attestation
  - Tricky
    - What is a third-party qualified to attest to?
    - Culturally informed/biased?
    - But scales well...
Homework

- Read:
  - *Reflections on Trusting Trust* by Ken Thompson
  - *This World of Ours* by James Mickens
    - [https://www.usenix.org/system/files/1401_08-12_mickens.pdf](https://www.usenix.org/system/files/1401_08-12_mickens.pdf)

- We will post first project next week
  - Getting comfortable with the debugger and project submission system
Next Lecture...

Security Foundations: Threat Models and Risk Analysis