

# Security Engineering and Security Objectives

UT CS361S

Spring 2021

Lecture Notes

# What is it?

- “[It] is about building systems to remain dependable in the face of ...”
  - Malice
  - Error
  - Mischance.
- “As a discipline, it focuses on the...”
  - Tools
  - Processes
  - Methods

# The Goal

- Confidentiality, Integrity, Availability (CIA Triad)
- For new systems:
  - Design security
  - Implement security
  - Test security
- For existing systems:
  - Adapt them for increased security
  - Adapt them as their *environment* evolves

# Key Observation

and covertness. But many systems fail because their designers protect the wrong things, or protect the right things but in the wrong way.

**Anderson, Ch 1, p. 4**

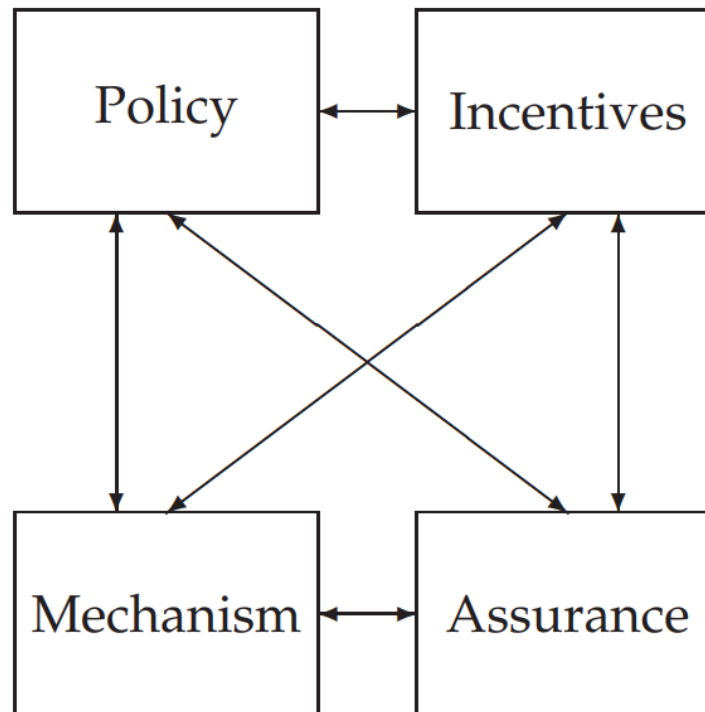
# Understand This

So if you're the owner of the company, don't fall into the trap of believing that the only possible response to a vulnerability is to fix it, and distrust the sort of consultant who can only talk about 'tightening security'. Often it's too tight already, and what you really need to do is just focus it slightly differently.

**Anderson, Ch 25, p. 816**

# A Framework

- Policy
- Mechanism
- Assurance
- Incentives



**Figure 1.1:** Security Engineering Analysis Framework

# Grasp the Context

- SECURITY IS ABOUT CONTEXT (Repeat after me)
- What does it mean when you say “system X is secure”?
  - Secure against *whom*?
  - Secure under *what conditions*?
  - Are we even protecting what matters?!
- Take voting security
  - Who are the potential attackers?
  - How does the context change if a nation decides to be the attacker?

# Anderson's Examples

- Bank
- Military
- Hospital
- Home



# Some Definitions

- System – Tech + Auxiliary Tech + Staff + Users + etc.
- Subject – Physical “person”
- Principal – Entity in the system
- Identity – Unique label attached to a unique principal
- Trusted – Failure results in compromise
- Trustworthy – Failure is unlikely

# IAAA

- Identity – Unique label for a unique principal
- Authentication – Validation of the principal's identity
- Authorization – Permissions granted the principal
- Accountability – Metering and auditing of principal
- (Message Authenticity – Integrity + Freshness)

# Start with Policy

- "...a succinct statement of a system's protection strategy"  
(Anderson ch1 p. 15)
- Examples:
  - Each credit must be matched by an equal and opposite debit
  - All transactions over \$1,000 must be authorized by two managers
- Practice:
  - What are the security policies for TLS?

# CIA (Not the Spies)

- Confidentiality – Cannot be read
- Integrity – Cannot be altered
- Availability – Cannot be interrupted

# *Then figure out mechanism*

- This is where most security people like to start
- But really we only need mechanism to enforce policy
- Some mechanisms aren't even technical (e.g., legal)
- MUST understand *threat model*

# Assurance

- Just how strong/resilient/comprehensive is the mechanism?
- Requires a solid understanding of the threat model
- Applications at every stage!
  - Design – solid security engineering principles
  - Implementation – coding practices, development processes
  - Testing – adversarial, comprehensive assessment

# Software Quality: Therac 25

- <http://sunnyday.mit.edu/papers/therac.pdf>
- Computer controlled radiation medical therapy machine
- Between 6/'85 and 1/'87, it overdosed 6 people (3 died)
- The problems were primarily software failures

# Race Condition Bug

- Real time operating system gathers details from the UI
- Setting the bending magnets takes 8 seconds
- Checks for data edits (in real time) as it is setting the magnets
- However, cleared variable mean subsequent edits are not recorded (but show up in UI)



# Overflow bug

- Error-checking and integrity checking code protects software
- One variable would perform a check if the value was non-zero
  - But the variable was just 8 bit
  - Every 256<sup>th</sup> check would overflow back to zero
- When the tech hit “set” when this overflow happened would allow full, maximum exposure

# Understanding Failures

- Everything fails.
- Everything.
- Don't be like AECL ("It can't fail that way...")
  - I recently had engineers of a client say the exact same things
  - They couldn't understand why I thought their software would fail
- How will *your* software fail?
  - You have to ensure that you fail safely
  - Some failures can never be tolerated; those features may need to be removed
  - Related: Make sure you use *fail safe defaults*

# Incentives

- Anderson's example of airport security
- What motivates the behavior?
- What is "Security Theater?"
- Everyone should learn a little game theory
  - Read up on Prisoner's dilemma
  - Understand "mechanism design"
  - Anderson's "Moral Hazard" (Chapter 25)

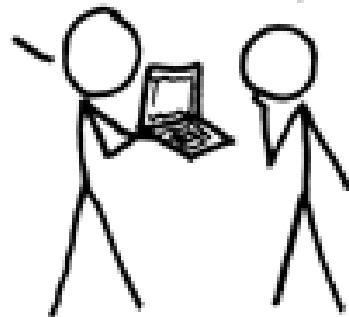
# Illustrations

## A CRYPTO NERD'S IMAGINATION:

HIS LAPTOP'S ENCRYPTED.  
LET'S BUILD A MILLION-DOLLAR  
CLUSTER TO CRACK IT.

BLAST! OUR  
EVIL PLAN  
IS FOILED!

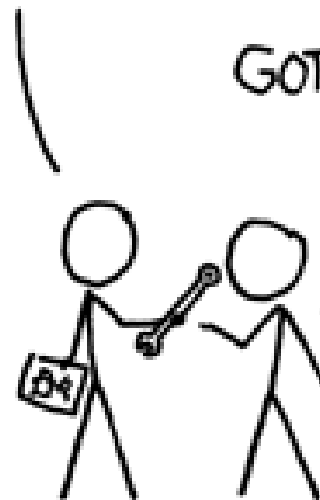
NO GOOD! IT'S  
4096-BIT RSA!



## WHAT WOULD ACTUALLY HAPPEN:

HIS LAPTOP'S ENCRYPTED.  
DRUG HIM AND HIT HIM WITH  
THIS \$5 WRENCH UNTIL  
HE TELLS US THE PASSWORD.

GOT IT.



# Security Principles

- Least privilege
- Minimize attack surface
- Defense in depth
- Separation of duties and responsibilities
- Crowdsourcing
- Open systems
- Fail Safe/Fail Secure

# Psychology

- Much of computer security rests on **psychology**
- Phishing, email scams, etc all depend on user psych
- Security mechanisms avoided because they clash with psych

# Social Engineering: *Pretexting*

- Pretexting: phone for info pretending to be someone else
- The goal is to private information, credentials, etc
- Attacks of this sort are often conducted in stages
  - Attacker starts by getting non-sensitive information
  - Attacker uses non-sensitive information to convince others of his deception
- One health institution had 30 such calls a week!

# Social Engineering: Phishing

- Phishing: Sending an email that appears to be authentic to get private info
- Easy to create an email that looks authentic.
  - Especially modern emails with HTML and media
  - The resources are often online, so the phishing email simply points to them
  - Email is easily forgible
- Example attacks:
  - Tell user they need to change password (and enter old password first)
  - Tell user they need to update profile including SSN



# Understanding Human Bias

- Humans are not rational
- Humans are designed with a bias toward action
  - If we thought about everything we'd never do anything
  - We're programmed to act without thinking
- Examples of bias:
  - We're more afraid of dying in a plane crash than a car crash

# When Emotion Takes Over

- When human logic/thinking ends, emotions take over
- If we don't know explicitly what to do, we respond emotionally
- So, sometimes education has limited value
  - Bad guys will always learn how to exploit what the users don't know
- One solution is safe defaults (FAIL SAFE/FAIL SECURE!)
  - "Our bank will never, ever send email"

# Abuses of Authority

- Read carefully the book's examples of how people behave
  - Under someone else's authority
  - When they have authority
- Also, people do not like to admit they make mistakes
  - “Hustlers” take advantage of this
- AGAIN, you cannot design assuming the user is dispassionate and rational

# CAPTCHAs

- Good case study!
  - Combine psychology, usability, and system design nicely
  - Designed around what humans do well that computers do not
  - “Completely Automated Public Turing Test to Tell Computers and Humans Apart”
  - Thanks Alan Turing!



# Network Security

- Everyone wants a secure network. But how?
- ***“Whoever thinks his problem can be solved using cryptography, doesn’t understand his problem and doesn’t understand cryptography.”***
  - — Attributed by Roger Needham and Butler Lampson to Each Other
- This is what we’re going to study this semester!