

E0-256

Computer Systems Security

Vinod Ganapathy
Lecture 1: Introduction

Course administration

- Instructor: Vinod Ganapathy
vg@iisc.ac.in

Course Info

Everything you need for the course is at the class website:

<http://www.csa.iisc.ac.in/~vg/teaching/E0-256>

Syllabus

- Goal of the course:
 - To get you up-to-speed on the state-of-the-art in computer security
- The field is very dynamic: attacks and defences continuously evolve.
- Most textbooks are woefully out of date.
- So, we will read research papers.

Syllabus

- Computer security is a **VERY** broad area.
- Impossible to cover breadth in a single course:
 - We will focus on **some** system security issues
 - Memory-errors and sandboxes, Web, Cloud, IoT and other emerging areas
 - Even for these topics, we'll only see the tip of the iceberg. The realities are vast and deep. Goal is to give you a taste of what's there and encourage you to start looking.
 - Not even touching **cryptography** or **privacy** (see courses by Profs. Bhavana, Arpita, Sanjit)

Reading research papers



- Not written in your usual textbook style
- Each paper written in a slightly different style
 - Can take 3-4 hours to read a paper when you're starting off to truly understand and appreciate the ideas that it presents.
- One of the goals of the course is to teach you how to read research papers
 - Getting to the core ideas of the paper
 - Adding these core ideas to your “toolbox”

Reading research papers

- Try to read the paper before coming to class.
 - You may not understand everything, and that's okay. Just get a general feel for the topic that the paper covers.
- In class, we will discuss the context, background and technical details of the paper.
- Go home, and study the paper with your new-found understanding.

Reading research papers

The rules are the same, whether you're in kindergarten or doing your post-graduate studies

Before Reading	During Reading	After Reading
 <ul style="list-style-type: none">•What story clues are in the title and the pictures?•Is this story real or make believe? How do I know?•If this text is real, what will I learn?•What will the main character need or want?•Why do I want to read this story?•How do I picture the setting?	 <ul style="list-style-type: none">•What will happen next in the story?•How do I feel about the main character?•Why does the character act or feel a certain way?•Does the story or text make sense?•How will the story most likely end?•How does this story or text remind me of my life?	 <ul style="list-style-type: none">•How did the story or text make me feel?•What do I like or dislike about the story?•What is the main part of the story or text?•How have my feelings about the character changed?•How are the character's feelings or actions different at the end of the story?•What is the author trying to tell me?

Reading research papers

- Most papers that we will read are seminal research results
- Google the names of the authors
 - These are the big-players in computer systems security.
 - Learn about their research by perusing through other papers they have written
 - A good way to come up with project ideas

Other course-related stuff

- Weekly workload and what is expected.
- Evaluation components:
 - 2 HWs (not graded) + 3 Exams + Project
- Grading criteria
- Project details

**ALL THESE DETAILS ARE ON THE CLASS
WEB PAGE!**

Grading

What this means

If all of you do well, you all get A+ grades. The converse holds too :-)

WYSIWYG:

- No room for uncertainty due to “curving” issues.
- No back-and-forth discussions at end of semester about the grade you got
- Grades are non-negotiable

Score range	Letter grade
$90 \leq \text{score} \leq 100$	A+
$80 \leq \text{score} < 90$	A
$70 \leq \text{score} < 80$	B+
$60 \leq \text{score} < 70$	B
$50 \leq \text{score} < 60$	C
$40 \leq \text{score} < 50$	D
$0 \leq \text{score} < 40$	F

Exams

- All exams will be open-book. Will test deep understanding of material rather than mere knowledge of material
- We have a strict, no-nonsense policy against cheating. See the class webpage for details.

Project details

- Unique project this semester: Requires coordination of whole class.
- Going to try to mimic real-world software development.
- Entire class will be one big team, divided into multiple teams (each of size 2).
- You will all coordinate to build **ONE** big software system
- Most likely **mbed-tls**, which you will port **to Rust**, but I will confirm at the end of October once class size stabilizes

Project details: “Yes” answers

- Yes, it is going to be difficult
- Yes, there is going to be a ton of coding
- Yes, coordination between teams is going to be a big challenge
- Yes, it will require you to take initiative to organize yourselves into groups, create “program manager” roles, coordinate software development and deliver the software at the end of the semester.
- Yes, you’re going to have to learn good software engineering practices

Project details: “No” answers

- No, I will not give credit if your part is not working because someone else failed to deliver theirs
- No, I will not be there to resolve conflicts between teams – this is not kindergarten
- No, I am not here for overall coordination, that is your job as a class
- No, there will be no extensions to the final deliverable deadline
- **No, what you heard from your seniors last year is not going to help you this year**

Attending class

You are encouraged to attend every class. But you're all grown ups -- So no attendance policy will be enforced.

BUT, in each class, I will often cover material that is not in the papers, not in the slides, and not in any textbook. The exams may contain questions based on this material.

Class discipline

- Two basic common-courtesy rules
- Rules that ***you*** would expect ***me*** to follow as your instructor

Class discipline

Come to class on time

- Class starts at 11am, not 11:10am or 11:15am.
 - Unless otherwise noted, e.g., if there is an interesting department seminar before class (which I may ask you to attend as well).
- Ambling into class 10-15 minutes late is **unacceptable**

Contacting us

- ++ Attend and ask questions in class
- ++ Be curious. Seek out and read research papers on your own. I'm happy to provide pointers and you can contact me or my students (in the CSSL lab) for references to papers in topics that you're interested in pursuing further.
- ++ Some classes will be held live during the class hours.
- ++ Going forward, I will try to pre-record classes and put them on the webpage. I will be available during class hours for answering questions.

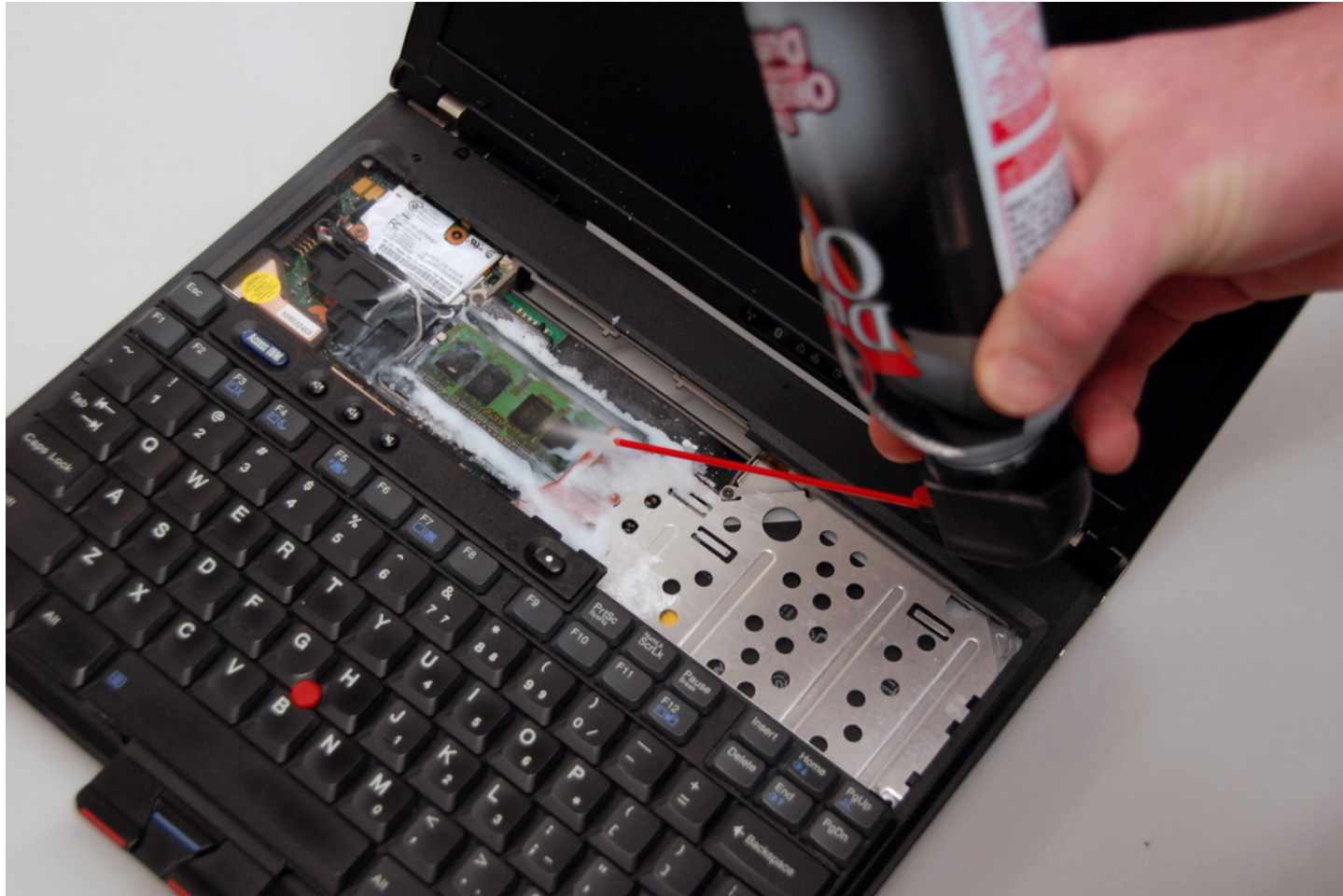
Computer security is the study of ...

- Weaknesses in systems and attacks against them.
- Defending against such attacks
- Pro-actively protecting data against various attacker models

Q1: Can you define a “secure system”?

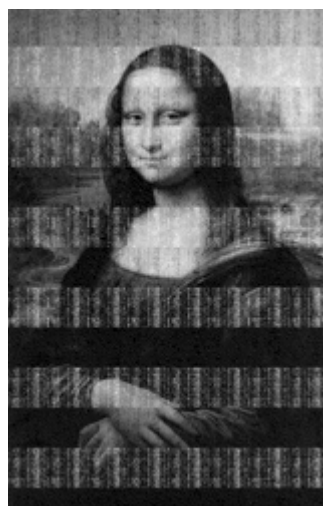
Q2: Real examples of “secure systems”?

A “cold boot” attack



Photos courtesy of: <https://citp.princeton.edu/research/memory/media/>
Slide #1-21

A “cold boot” attack



Photos courtesy of: <https://citp.princeton.edu/research/memory/media/>

Basic Components

- Confidentiality
 - Keeping data and resources hidden
- Integrity
 - Data integrity (integrity)
 - Origin integrity (authentication)
- Availability
 - Enabling access to data and resources

Goals of Security

- Prevention
 - Prevent attackers from violating security policy
- Detection
 - Detect attackers' violation of security policy
- Recovery
 - Stop attack, assess and repair damage
 - Continue to function correctly even if attack succeeds

Threat Models

- A **threat model** is a set of assumptions about the power of an adversary.
- The concept of a “secure system” is defined with respect to a threat model
- Examples of threat models?

Trusted Computing Base

- To prove that a system is “secure” against a particular adversary, we often make certain assumptions about portions of the system.
- Those constitute the **Trusted Computing Base (TCB)** of the system
- If the TCB is compromised, can no longer guarantee security of the system.

Trusted Computing Base

- Examples of the TCB on real systems?
- Given the TCB, is it reasonable for us to place trust in the TCB? Or is our trust misplaced?
 - How low can you go on a real system?

Policies and Mechanisms

- Policy says what is, and is not, allowed
 - This defines “security” for the site/system/*etc.*
- Mechanisms enforce policies
- Real-world example: locks on door.
- Composition of policies
 - If policies conflict, discrepancies may create security vulnerabilities

Policies and Mechanisms

- Policies
 - Unambiguously partition system states
 - Correctly capture security requirements
- Mechanisms
 - Assumed to enforce policy
 - Support mechanisms work correctly

Security Design Principles

- It is very hard to design systems that are resistant to attack.
- Even harder to prove that a system is secure. We make several assumptions:
 - Threat model
 - Trusted Computing Base
- Attackers are clever: Not subject to these assumptions and constantly improvise.
- Nevertheless, can we follow certain design principles that result in “more secure” systems?

1: Least Privilege

- A subject should be given only those privileges necessary to complete its task
 - Function, not identity, controls
 - Rights added as needed, discarded after use
 - Minimal protection domain

2: Fail-Safe Defaults

- Default action is to deny access
- If action fails, system as secure as when action began
- Blacklists versus Whitelists

3: Economy of Mechanism

- Keep it as simple as possible
 - KISS Principle
- Simpler means less can go wrong
 - And when errors occur, they are easier to understand and fix
- Interfaces and interactions
- Applies particularly to TCB design.

4: Complete Mediation

- Check every access
- Usually done once, on first action
 - UNIX: access checked on open, not checked thereafter
- If permissions change after, may get unauthorized access

5: Open Design

- Security should not depend on secrecy of design or implementation
 - Popularly misunderstood to mean that source code should be public
 - “Security through obscurity”
 - Does not apply to information such as passwords or cryptographic keys

6: Separation of Privilege

- Require multiple conditions to grant privilege
 - Separation of duty
 - Defense in depth

7: Least Common Mechanism

- Mechanisms should not be shared
 - Information can flow along shared channels
 - Covert channels
- Isolation
 - Virtual machines
 - Sandboxes

8: Psychological Acceptability

- Security mechanisms should not add to difficulty of accessing resource
 - Hide complexity introduced by security mechanisms
 - Ease of installation, configuration, use
 - Human factors critical here

Key Points

- Principles of secure design underlie all security-related mechanisms
- Require:
 - Good understanding of goal of mechanism and environment in which it is to be used
 - Careful analysis and design
 - Careful implementation