15-316: Software Foundations of Security and Privacy

Introduction

Frank Pfenning, Fall 2024

Course Staff

Myra Dotzel

Derek Duenas

Hemant Gouni

Ray Man

Jason Yao

This course

Security from a programmer's standpoint

We'll cover a range of concerns

- Safety: *The code will never do something we deem "bad"*
- Isolation: *Untrusted code and data can't affect important state*
- Information Flow: *Confidential data remains that way*
- Privacy: *Control over peoples' data and how it's used*
- Authorization: *Only designated actors/code can obtain rights*
- Trust: *Leverage a small base to establish trust in a complete system*

Recurring themes

- Ways of specifying computations that are secure, i.e. *policy*
- Ways of ensuring that code meets policy, i.e. *enforcement*
- Formal ways of connecting the two, i.e. *semantics*

Logic and languages

Why is this a logic and languages elective?

Precise ways to write down policies

- Types, logical formulas, domain-specific languages
- Often devised for correctness, perfect for security also

Enforcing them *rigorously*

- Static: verification, type checking
- Dynamic: runtime monitors, code instrumentation
- Either way, prove that the policy won't be violated

Learning objectives

After taking this course, you should:

- Be able to identify, formalize, and implement a range of practical security & privacy policies
- Understand the tradeoffs of different approaches to security & privacy, and how to use context-specific rationale to justify them
- Understand how general principles like least privilege, roots of trust, and complete mediation play a role in formulating and vetting defenses
- Be able to provide a formal, rigorous argument for several types of security mechanisms that are used in practice

Logistics

- Website: https://www.cs.cmu.edu/~15316
- Contact: Piazza
- Lecture: This room, same time, Tuesday/Thursday
  - Attendance expected, but not recorded
  - Piazza is not a substitute for attending lecture
- Submit everything to Gradescope

- Office hours
  - My office, Mondays from 2:00-3:20 or by appointment
  - TAs in Gates Commons, details posted soon

Grading

- Breakdown:
  - 40% written homework
  - 40% labs
  - 20% midterm exam

Final letter grades

- 90+ guarantees an A
- 80+ guarantees a B, etc.
- We reserve the right to move thresholds down

- 5 written homeworks
- 3 labs
- 1 midterm
- 0 final exams
- 5 late days for written homeworks
- 3 late days for labs
- At most 2 per hw/lab

Written homework (40%)

Focus on theory + fundamental skills

1 week to complete

Due dates for the semester are on the course website

Solo (but: whiteboard policy)

Grades based on:

- Correctness (obviously)
- Rigor
- Clarity

Rigor and clarity in this course:

- Formally state what you intend to show
- List any assumptions
- Show your steps, with appropriate justification

Labs (40%)

Translate theory into something that works

- Design a set of policies and a way of enforcing them
- Formalize both
- Implement them for the setting described in the lab
- Learn to use new tools in the process

Grades are based on:

- Correct functionality (evaluated mostly by autograder)
- Completeness, robustness of security mechanism
- Documentation, clarity of your solution

Complete the labs in pairs (recommended) or solo

- We can help with: understanding the setting/concepts, critiquing your approach, explaining strengths/limitations of tools
- We're less useful for: debugging code, leaking autograder tests, giving last-minute hints

Midterm Exam (20%)

In lecture slot and room, October 10

You need to understand the fundamentals

- No laptop, no notes
- Relevant reference sheet provided with exam

Final Exam (0%)

Last lab will have written and programming component

A different (better) kind of learning experience

Before Thursday

Make sure that you are enrolled in Piazza & Gradescope (not Canvas)

- See course pages at https://www.cs.cmu.edu/~15316 for signup links

Answer Piazza poll on programming language support

Read the course pages & syllabus, reach out if there are questions