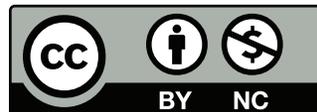


---

# COMS W4187: Security Architecture and Engineering

Prof. Steven M. Bellovin

<https://www.cs.columbia.edu/~smb/classes/f16>



---

## What is this Course?

- Security primitives
- Security architecture
- Security engineering
- How to think about security
- 👉 How to think about insecurity...
- *Not* 4180—complementary to it

---

# Security Primitives

- What are the basic mechanisms you can use to secure a system?
- What are the properties of these mechanisms?
- What is the *assurance* associated with them?

---

# Security Architecture

- How to put the pieces together
- How to spot the risky parts
- How to evaluate an architecture

---

# Security Engineering

- Putting the pieces together
- Tradeoffs
- Balancing cost, security, usability, acceptability, and more

---

## How to Think About Security

- Security is a property of the overall design
- You do *not* get security by sprinkling on crypto or by forcing people to change their passwords frequently
- Those can sometimes help—but bad guys go around strong security, not through it
- Security is a *systems property*

---

## How to Think About Insecurity...

- The bad guys don't follow the rules
- To understand how to secure a system, you have to understand what sort of attacks are possible
- Note that that is *not* the same as actually launching them. . .

---

## Course Structure

- Lecture format
- Syllabus subject to change to discuss current events
- Approximate grading percentages:

Homework	48%
Midterm	21%
Final	31%

Experience suggests that the exact percentages are not that important (and may change slightly)
- Grades will be posted on Courseworks
- Yes, I curve

---

## Readings

- *The Craft of System Security*, Sean Smith and John Marchesini, Addison-Wesley, 2008, ISBN 0-321-43483-8.
- *Thinking Security: Stopping Next Year's Hackers*, Steven M. Bellovin, Addison-Wesley, 2016, ISBN 0-13-427754-6, 0-13-427754-6.
- Some primary source material—I assume you all know how to use the library and/or electronic resources. (Hint: Google does not (yet?) have access to all of the world's knowledge.)
- Note: ACM and IEEE readings are often only easily available from the campus network.

---

## Logistics

- For grading issues, approach the TAs within two weeks; if you don't receive a satisfactory answer, contact me.
- For issues relating to *this class*, email [smb+4187@cs...](mailto:smb+4187@cs...)
- That lets me auto-sort class-related mail and keep better track of things

---

# Programming Assignments

- All programming homework *must* be done in C or C++ unless otherwise instructed. Don't bother asking for exceptions.
- Turn in a single `tar` file, including a Makefile; if necessary, include test data and a README file with execution instructions
- All programs *must* compile and run on Linux on the Google Cloud machines; zero credit for programs that don't compile. Note that this means you must be comfortable compiling and running code on Linux.
- Because most security problems are due to buggy code, there will be copious deductions for bugs or for inadequate documentation

---

## Again:

- Programs must compile and run on Linux on the Google Cloud machines
- You can do your initial development on any platform you want—but you *must* make sure that the submitted version works on Google Cloud

---

## Co-operation versus Dishonesty

- Discussing homework with others is encouraged—but all programs and written material *must* be individual work unless otherwise instructed.
- Please use appropriate file permission mechanisms to protect your homework. (Looking at other people's work is forbidden.)
- Zero tolerance for cheating
- See the department's honesty policy:  
<http://www.cs.columbia.edu/education/honesty> I will assume that you have all read it; you are in any event responsible for its terms and provisions.

---

## Using Open Source Programs

- Generally, you are free to use any binary software installed on the Google Cloud machines
- Generally, you are welcome to use any open source software if (a) all such files are in a separate subdirectory from anything you write; (b) you clearly identify the origin of all such code; and (c) it all compiles seamlessly if the grader just types 'make' in the parent directory.
- *Exception:* you may not use outside code that accomplishes the primary purpose of the assignment.
- If in doubt, ask me *first*.

---

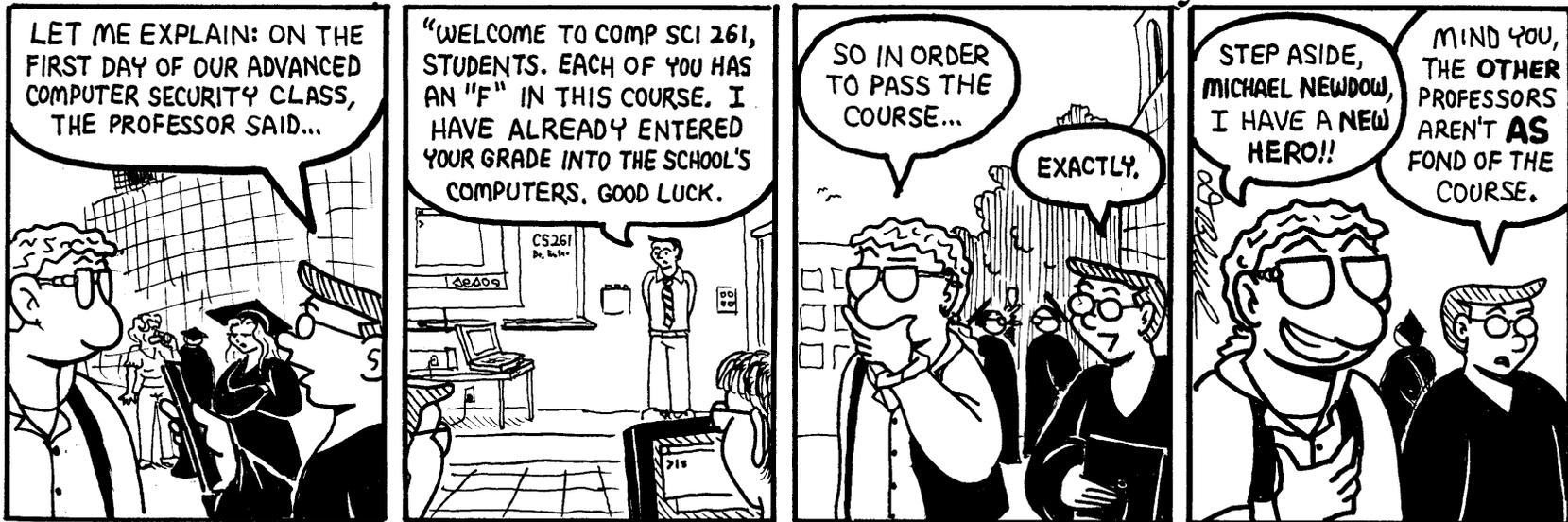
# The Ethics of Security

- Taking a computer security class is *not* an excuse for hacking
- “Hacking” is any form of unauthorized access, including exceeding authorized permissions
- The fact that a file or computer is not properly protected is no excuse for unauthorized access
- *If* the owner of a resource invites you to attack it, such use is authorized
- For more details, see  
[http://www.columbia.edu/cu/policy/network\\_use.html](http://www.columbia.edu/cu/policy/network_use.html)
- *Absolutely no Trojan horses, back doors, or other malicious code in homework assignments*

# Not How I Teach It!

*Nukees*

by Darren Bleuel



<http://www.nukees.com>

Copyright 2007 Darren Bleuel. All rights reserved.

(Used with permission; see <http://www.nukees.com>)

---

## Contacting Me

- Feel free to drop in during office hours.
- I'll announce changes on my home page
- I'm amenable to meeting other times, by appointment. You're welcome to drop in if my office door is open, but I reserve the right to ask you to come back later
- If you have any questions, please use email rather than telephone; I travel a lot and am not very reachable by phone

---

## Talking to Me

- Drop by just to talk (a good idea if you think you'll want me to write a recommendation. . .)
- You don't need to be in trouble to talk with me. . .
- If my office door is open, just walk in
- But—I travel too much

---

## Class Schedule

- Probable midterm date: October 19
- The final will be held on the date specified by the registrar. The current projection is December 21, 1:10-4:00—but that's **tentative**. Do *not* make any travel plans that involve leaving campus before that.

---

## TAs

- John Koh <koh@...>
- Prateek Gupta <pg2512@...>
- Haoxiang Xu <hx2185@...>

---

# Lectures

- I prepare slides for each class, and upload them shortly before class time
- Slides (and other information) are uploaded to my web page
- Well, occasionally they're uploaded shortly after class. . .

---

## Homeworks

- As noted, four homework assignments
- Homeworks are designed for practice, teaching, and evaluation
- Homeworks must be submitted electronically by the start of class
- Homeworks received later that day lose 5%, the next day 10%, two days late 20%, three days late 30%; after that, zero credit
- Exceptions granted only for *unforeseeable* events. Workload, day job, etc., are quite foreseeable.
- No grace period, no freebies
- Problems? See me *before* the due date

---

## Grade Arithmetic

- If four homeworks total 48%, each one is 12%
- If you miss one assignment, that's 12% off your final average
- That's generally more than a single letter grade
- Last year, an 88% was a B

---

# The Google Cloud

- All programs *must* run on the Google Cloud
- Programs that don't compile *on those machines* receive zero credit
- You will receive credentials and instructions for how to set up your login

---

# Responsibility

- You're all adults
- You're all responsible for your own actions
- If there's something missing, you have to tell me

---

## Practical Focus

- This is not a pure academic-style security course
- You'll be experimenting with real security holes
- A lot of (in)security is about doing the unexpected
- The ability to “think sideways” is a big advantage

---

## The First Question

*“What are you trying to protect, and against whom?”*

- Some assets are more valuable than others
- Different enemies have different goals
- Different enemies have different abilities

You don't want to spend too much—but you also don't want to spend too little.

---

## What is Security?

*Security is keeping unauthorized entities from doing things you don't want them to do.*

This definition is too informal...

---

# What is Security?

- Confidentiality
- Integrity
- Availability

---

# Confidentiality

- “The property that information is not made available or disclosed to unauthorized individuals, entities, or processes [i.e., to any unauthorized system entity].” [definitions from RFC 4949]. Not the same as *privacy*.
- Contrast with *privacy*: “The right of an entity (normally a person), acting in its own behalf, to determine the degree to which it will interact with its environment, including the degree to which the entity is willing to share information about itself with others.” Privacy is a reason for confidentiality
- The traditional focus of computer security

---

# Integrity

- **data integrity:** “The property that data has not been changed, destroyed, or lost in an unauthorized or accidental manner.”
- **system integrity:** “The quality that a system has when it can perform its intended function in a unimpaired manner, free from deliberate or inadvertent unauthorized manipulation.”
- Often of more commercial interest than confidentiality

---

# Availability

- “The property of a system or a system resource being accessible and usable upon demand by an authorized system entity, according to performance specifications for the system; i.e., a system is available if it provides services according to the system design whenever users request them.”
- Turning off a computer provides confidentiality and integrity, but hurts availability. . .
- Denial of service attacks are direct assaults on availability

---

## They Interact

- It's obvious that violations of integrity can be used to compromise confidentiality
- In some situations, violations of availability can be used that way as well

---

## Security is a System Problem

- You can't solve it one component at a time
- It's not a matter of adding crypto or using better passwords
- Firewalls don't do it, either
- All of these things help—but all of the components interact

---

## More Definitions

**vulnerability** An error or weakness in the design, implementation, or operation of a system

**attack** A means to exploit some vulnerability in a system

**threat** An adversary that is motivated and capable of exploiting a vulnerability

(Definitions from *Trust in Cyberspace*)

---

# Vulnerabilities

- The technical failing in a system
- The primary focus of most computer security classes
- If you can close the vulnerabilities, the threats don't matter
- Or do they?

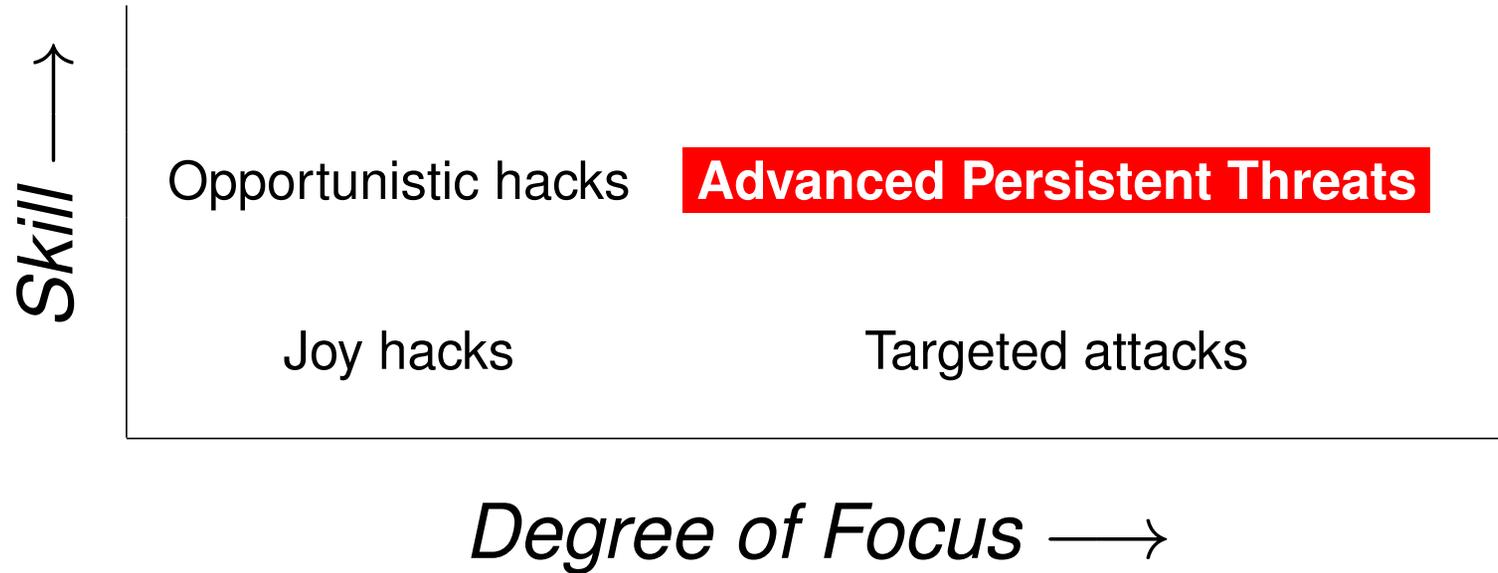
---

# Threats

- Different enemies have different abilities
- Teenage joy-hackers can't crack a modern cryptosystem
- Serious enemies can exploit the “three Bs”: burglary, bribery, and blackmail
- But are they motivated to attack you?
- You can't design a security system unless you know who the enemy is

---

## The Threat Matrix



---

## Types of Attackers

**Joy hackers** Beginners; doing it for fun; little knowledge or focus

**Opportunistic attackers** Often very good but will hit any vulnerable target

**Targetiers** Focuses on particular target; gathers background information, but not necessarily skilled

**APT** Skilled and focused.

---

## Advanced Persistent Threats

- The most dangerous attacker
- Still a wide range, from skilled criminals to some governments to very advanced cyberwarfare and cyberespionage activities—think Stuxnet
- The US? China? Russia? Israel? North Korea? (I'm blaming the “Andromedans”...)
- Caution: many corporations blame ordinary penetrations on APTs

---

## Don't Neglect the Human Element

“Humans are incapable of securely storing high-quality cryptographic keys, and they have unacceptable speed and accuracy when performing cryptographic operations. They are also large, expensive to maintain, difficult to manage, and they pollute the environment. It is astonishing that these devices continue to be manufactured and deployed, but they are sufficiently pervasive that we must design our protocols around their limitations.”

*Network Security: Private Communication in a Public World*, Kaufman, Perlman, and Speciner

---

# Engineering

- Sometimes, requirements are inconsistent and/or incomplete
- Conflicts:
  - Security versus cost
  - Security versus performance
  - Security versus acceptability and culture
  - Security versus usability
  - Security versus security!
- We'll discuss how to detect and analyze such conflicts

---

# Designing Security

- The problem is overconstrained
- Among the constraints are cost, human behavior, and ease of operation
- In the real world, realistic security is often far more important than theoretical security
- *What are you trying to protect against whom?*

---

## What this Course is About

- Thinking about security
- Mechanisms
- Threat analysis
- Security architecture
- Assurance
- In short, *engineering* secure systems