CSE543 - Introduction to Computer and Network Security

Module: Introduction

Professor Trent Jaeger
Some bedtime stories ...
Some bedtime stories ...
Some bedtime stories …
to nightmares …

Figure 5: Items purchased separated into product category and customer age. The left half of each graph show orders from women, and the right half shows orders from men. Customers restricted to those who self-report age and sex.
College of Engineering network disabled in response to sophisticated cyberattack

Plans in place to allow teaching, research in the college to continue as University moves to recover
May 15, 2015

UNIVERSITY PARK, Pa. – The Penn State College of Engineering has been the target of two sophisticated cyberattacks conducted by so-called “advanced persistent threat” actors, University officials announced today. The FireEye cybersecurity forensic unit Mandiant, which was hired by Penn State after the breach was discovered, has confirmed that at least one of the two attacks was carried out by a threat actor based in China, using advanced malware to attack systems in the college.

In a coordinated and deliberate response by Penn State, the College of Engineering's computer network has been disconnected from the Internet and a large-scale operation to securely recover all systems is underway. Contingency plans are in place to allow engineering faculty, staff and students to continue in as much of their work as possible while significant steps are taken to upgrade affected computer hardware and fortify the network against future attack. The outage is expected to last for several days, and the effects of the recovery will largely be limited to the College of Engineering.
This course

• We are going to explore why these events are not *isolated*, *infrequent*, or even *unexpected*.

• Why are we doing so poorly in computing systems at protecting our users and data from inadvertent or intentional harm?
This course

• We are going to explore why these events are not *isolated*, *infrequent*, or even *unexpected*.

• Why are we doing so poorly in computing systems at protecting our users and data from inadvertent or intentional harm?

The answer: stay tuned!
This course ... 

- This course is a systems course covering general topics in computer and network security, including:
  - network security, software security, OS security, web security, cryptography, authentication, security protocol design and analysis, key management, intrusion detection, security policy, language-based security, cloud computing security, and other emerging topics (as time permits)
You need to understand ...

- How a Computer System *Really* Works
- Program Toolchains (E.g., gcc, binutils)
- Modern Operating Systems
- IP Networks
- Discrete Mathematics
- Basics of systems theory and implementation
  - E.g., File systems, distributed systems, networking, operating systems, ....
Goals

- My goal: to provide you with the tools to understand and evaluate research in computer security.
  - Basic technologies
  - Engineering/research trade-offs
  - How to read/understand security research papers

- This is going to be a hard course. The key to success is sustained effort. Failure to keep up with readings and projects will likely result in poor grades, and ultimately little understanding of the course material.

- Pay-off: security competence is a rare, valuable skill
Course Materials

• Website - I am maintaining the course website at
  ‣ http://www.cse.psu.edu/~tjaeger/cse543-f17/

• Course assignments, slides, and other artifacts will be
  made available on the course website.
  ‣ Assignment submissions and communications via Canvas

• Course textbook
  ‣ Information Security: Principles and Practice
    • Mark Stamp
Course Calendar

- The course calendar as all the relevant readings, assignments and test dates
- The calendar page contains electronic links to online papers assigned for course readings.
- Please check the website frequently for announcements and changes to the schedule. Students are responsible for any change on the schedule.
Grading

• The course will be graded on exams, projects, paper reviews and class participation in the following proportions:

  40% Projects
  20% Mid-term Exam
  30% Final Exam (comprehensive)
  10% Paper Reviews & Participation

• **NOTE:** Must do better than 50% average on each of exams and programming projects to pass the course
Exams

- Midterm and Final
  - Same Format
  - Short Answer Questions
    - What is X?
  - Conceptual Questions
    - Why is Y?
  - Constructions
    - How is Z?
- Time can be an issue
  - Answer the questions you know
- Final is worth far more than midterm
Projects

• **Goal:** Learn security concepts
• **Goal:** Learn research skills

• Two Projects (Individual)
  • Crypto and Passwords
  • Software Security
  • 50%+ on these

• Research Project (Team)
  • Research Topic - Find limitation in prior work and sketch out experiment - More Later…
Readings

• There are a large amount of readings in this course covering various topics. These assignments are intended to:
  ‣ Support the lectures in the course (provide clarity)
  ‣ Augment the lectures and provide a broader exposure to security topics.

• Students are **required** to do the reading!

• **About 10-20% of questions on the tests will be off the reading on topics that were not covered in class.** You better do the reading or you are going to be in deep trouble when it comes to grades.
Paper reviews

- Goal: Record key ideas and methods for later
- We will review one paper per week
Ethics Statement

• This course considers topics involving personal and public privacy and security. As part of this investigation we will cover technologies whose abuse may infringe on the rights of others. As an instructor, I rely on the ethical use of these technologies. Unethical use may include circumvention of existing security or privacy measurements for any purpose, or the dissemination, promotion, or exploitation of vulnerabilities of these services. Exceptions to these guidelines may occur in the process of reporting vulnerabilities through public and authoritative channels. Any activity outside the letter or spirit of these guidelines will be reported to the proper authorities and may result in dismissal from the class and/or institution.

• When in doubt, please contact the instructor for advice. Do not undertake any action which could be perceived as technology misuse anywhere and/or under any circumstances unless you have received explicit permission from Professor Jaeger.
Academic Integrity Policy

• See the EECS Department’s Academic Integrity Standards for CMPSC, CMPEN, and CSE Programming Courses

• http://www.eecs.psu.edu/students/resources/EECS-CSE-Academic-Integrity.aspx

• You must follow this policy
Academic Integrity Policy

• The Department of Computer Science and Engineering expects all student programming work assigned in a class to be completed independently by students (or by teams if permitted/required) and to consist of code designed and developed solely by the students. The use of any other code is not permitted unless the course instructor explicitly allows it and such code is clearly identified as coming from an external source and that source is credited. Students will never be given credit for code which they did not construct.

• The department uses software tools to identify similarities in code submitted by students. These tools differentiate between insignificant cosmetic differences (names used in code, the order of certain code elements) and significant structural similarities (algorithms, data organization). These tools give a percentage of common code between two submissions and identify this common code. We do not set a single, fixed percentage above which we automatically determine that an academic violation has occurred. Rather we rely on the expertise of the instructor to determine when similarities rise above what a reasonable person could expect two students working independently to construct.
Academic Integrity Policy

• For example, in an introductory course in which the programming assignments require relatively short solutions (i.e., less than 50 lines of code) we would expect to see similarities in student solutions rising to a significant percentage of the code. But in an advanced course in which programming projects may contain thousands of lines of code, only a small percentage may be similar but still constitute an academic integrity violation if the code in question was a significant/important aspect of the assignment and if the similarities found could not, in the opinion of the instructor, have been independently developed.

• Furthermore, in cases where student submissions have been found to contain significant portions of code found in online sources (e.g., a common code hosting site is GitHub), the determination of an academic integrity violation is essentially automatic.
What is security?

• Garfinkel and Spafford (1991)
  ‣ “A computer is secure if you can depend on it and its software to behave as expected.”

• Harrison, Ruzzo, Ullman (1978)
  ‣ “Prevent access by unauthorized users”

• Not really satisfactory – does not truly capture that security speaks to the behavior of others
  ‣ Expected by whom?
  ‣ Under what circumstances?
  ‣ What are the risks?
Risk

• At-risk valued resources that can be misused
  ‣ Monetary
  ‣ Data (loss or integrity)
  ‣ Time
  ‣ Confidence
  ‣ Trust

• What does being misused mean?
  ‣ Confidentiality
  ‣ Integrity
  ‣ Availability
  ‣ Privacy (personal)

• Q: What is at stake in your life?
Adversary

- An adversary is any entity trying to circumvent the security infrastructure
  - The curious and otherwise generally clueless (e.g., script-kiddies)
  - Casual attackers seeking to understand systems
  - Venal people with an ax to grind
  - Malicious groups of largely sophisticated users (e.g., chaos clubs)
  - Competitors (industrial espionage)
  - Governments (seeking to monitor activities)
Are users adversaries?

• Have you ever tried to circumvent the security of a system you were authorized to access?
• Have you ever violated a security policy (knowingly or through carelessness)?
Are users adversaries?

• Have you ever tried to circumvent the security of a system you were authorized to access?
• Have you ever violated a security policy (knowingly or through carelessness)?

This is known as the insider adversary!
Threats

• A threat is a specific means by which an adversary can put a system at risk
  ‣ An ability/goal of an adversary (e.g., eavesdrop, fraud, access denial)
  ‣ Independent of what can be compromised

• A threat model is a collection of threats that deemed important for a particular environment
  ‣ A collection of adversary(ies) abilities
  ‣ E.g., a powerful adversary can read and modify all communications and generate messages on a communication channel

• Q: What were risks/threats in the introductory examples?
  ‣ Slammer
  ‣ Yale/Princeton
  ‣ Estonia
Vulnerabilities (attack vectors)

- A **vulnerability** is a **flaw** that is accessible to an adversary who can exploit that flaw
- E.g., buffer overflow, file open w/ adversary name
- What is the source of a vulnerability?
  - Bad software (or hardware)
  - Bad design, requirements
  - Bad policy/configuration
  - System Misuse
  - Unintended purpose or environment
    - E.g., student IDs for liquor store
Attacks

- An **attack** occurs when an adversary attempts to **exploit** a vulnerability

- Kinds of attacks
  - Passive (e.g., eavesdropping)
  - Active (e.g., password guessing)
  - Denial of Service (DOS)
    - Distributed DOS – using many endpoints

- A **compromise** occurs when an attack is successful
  - Typically associated with taking over/altering resources
Principals

• **Principals** are expected system subjects
  ‣ Computers, agents, people, enterprises, …
  ‣ Depending on context referred to as: servers, clients, users, entities, hosts, routers, … - and some may be adversarial
  ‣ Security is defined with respect to these subjects
    • Implication: every principal may have unique view

• A **trusted third party**
  ‣ Trusted by all principals for some set of actions
  ‣ Often used as introducer or arbiter
Trust

• **Trust** refers to the degree to which a principal is expected to behave
  ‣ What the principal not expected to do?
    • E.g., not expose password
  ‣ What the principal is expected to do (obligations)?
    • E.g., obtain permission, refresh

• A **trust model** describes, for a particular environment, who is trusted to do what?

• Note: you make trust decisions every day
  ‣ Q: What are they?
  ‣ Q: Whom do you trust?
Security Model

• A security model is the combination of a trust and threat models that address the set of perceived risks
  ‣ The “security requirements” used to develop some cogent and comprehensive design
  ‣ Every design must have security model
    • LAN network or global information system
    • Java applet or operating system
• This class is going to talk a lot about security models
  ‣ What are the security concerns (risks)?
  ‣ Who are our adversaries?
  ‣ What are the threats?
  ‣ Who do we trust and to do what?
• Systems must be explicit to be secure.
A Security Model Example

• Assume we have a University website that hosts courses through the web (e.g., Canvas)
  ‣ Syllabus, other course information
  ‣ Assignments submissions
  ‣ Online grading

• In class: elements of the security model
  ‣ Principals (Trusted)
  ‣ Adversaries
  ‣ Risks
  ‣ Threats