Introduction to Information Security
14-741/18-631 Fall 2021
Unit 6, Lecture 2:
Management, Assurance, and Evaluation

Hanan Hibshi hhibshi@andrew

This Lecture's Agenda

- **Outline**
  - Management issues in security
  - Assurance in secure systems
  - Evaluation of secure systems
  - Case study
- **Objective**
  - By now, you should know
    - Some of the available security techniques (crypto, software engineering, …)
    - What you are allowed to use (public policy & law)
  - This lecture helps you get elements of answers to the following important questions
    - How do you go about organizing security?
    - How do you know when you are done?

Should Security Always be the Top Priority?

- Assume turnover = $10 M, net profit = $1 M but penalty (for security flaws) = $150 K

- Would you rather fix the security flaw…
- …or double the turnover, assuming gross profit margin remains the same, but penalty triples, and goes to $450K?

Security engineers have to understand the economics of each situation

Risk Management

- **Purpose of a business is profit**
- **Security mechanisms can make a difference in the risk/reward equation…**
- **Risk management is the science (art?) of finding the most profitable balance between risk and reward**
  - Security risk is only one of many risks
  - Political risk, economic risk, legal risk…

Organizational Issues

- **Complacency cycle**
  - Systems get very secure immediately after attack
  - People gradually loosen their guard
  - System gets attacked again
- **Interaction with reliability**
  - Unreliable systems make fraud easier
- **Solving the wrong problem**
  - Putting all your companies secret in a 10-ft walled bunker
  - While failing to deal with insiders (e.g., dishonest staff)
- **Incompetent or inexperienced security managers**
  - Security manager job = saying no all the time
  - May not be good for promotion
  - Turn around (in the US) = 7 months
- **Moral hazard**
  - Tempting employees by (for instance) systematically covering up breaches

- **Wasn't engineering a secure system challenging enough?**

- **How do we deal with management issues?**

Security Requirements Engineering

- **Requirements engineering here means**
  - Developing security policy
  - Obtaining agreement on security policy with system owner
    - The latter may be the harder part!
- **Generating a policy is not that much different from producing code**

Evolving Requirements

- **Hard (impossible) to predict future uses of a system**
- **Equally hard to predict future attacks on a system**
- **Requirements have to evolve over time**

- **Four major causes leading to changes in security requirements**
  - Bug fixing
  - Control tuning in response to better knowledge of attacks
  - Environment evolution
  - Organizational change

Why Requirements?

- **Flight 501 for Ariane 5 rocket**
  - Launched 4 June 1996
  - Software error
    - Data conversion from 64-bit float to 16-bit int -> operand error
    - Exception generated; not handled; propagated onto bus
    - Processor shut-down
  - Backup computer?
  - Code written in Ada

Ariane 501 failure was due to a requirements error: reuse specification problem

Bug Fixing

- **Monitoring**
  - Make it easy for customers to report bugs
  - Have someone monitor *bugtraq* and assorted security mailing-lists
- **Repair**
  - Have on-call skilled people readily available
- **Distribution**
  - Make sure it scales to your customers
  - Push or pull model? Centralized server vs. automated distribution?
- **Reassurance**
  - Have canned press releases ready

Proper bug fixing is not just good engineering practice, it is also good advertising and gives you competitive advantage

Control Tuning

- **Continuous audit of your system**
  1. Assess risks
  2. Design controls
  3. Monitor control performance
  4. Back to 1.

- **Work with auditors**

Environment Evolution

- **Environment changes all the time, and present new challenges**
  - E.g., the Internet made DDoS possible
    - threat was known for a long time, but purely theoretical
- **Tragedy of the commons**
  - Adding one sheep to the common reduces marginally the resource available for other sheep
  - People tempted to add more sheep of their own
  - Resource gets exhausted
- **Typical of situations that require collaboration between companies**
  - i.e., where there is no monopoly
  - Very frequent in practice!
- **Solution: control by consortium?**

Organizational Change

- **Understanding employees is key**
- **Force-feeding change results in disasters**
  - E.g., London Ambulance Service
- **From security perspective, smooth evolution generally preferable to radical changes**

Assurance and Evaluation

- **Assurance**
  - Prove that the system been tested thoroughly enough
    - How do you define enough?
  - "Our estimate of the likelihood that the system will not fail in a particular way"

- **Evaluation**
  - Prove that the system does work as intended (from a security/reliability perspective)
  - "The process of assembling evidence that a system meets, or fails to meet, a prescribed assurance target"

Both extremely hard!

Assurance Goals

- **Functionality**
- **Strength of mechanisms**
- **Implementation assurance**
- **Usability**

- **Problem: User's focus not aligned with vendor's focus!**
  - E.g., PC user wants high usability, high strength, medium assurance, simple functionality
  - Vendors focus on high functionality (where revenue is), low strength (not marketable), low assurance (costly), low usability (developers generate externalities)

Project Assurance

- **Security testing**
  - White box testing
    - Code, documentation, and product available
    - Look for obvious flaws, then for common flaws (e.g., buffer overruns), then for less common flaws (e.g., poor communication protocol design)
- **Formal methods**
  - E.g., BAN logic
- **Testing the tests**
  - Deliberately injecting faults in a program
    - E.g., inject 100 faults, tester finds 70 of them (missed 30); if tester found 70 other vulnerabilities, you can expect him/her to have missed 30 vulnerabilities you don't know about
  - Parallel processing (use several testers)

Process Assurance

- **Focuses on development team**
- **Use rules, e.g., "fix your own bugs"**
- **Use certifications models**
  - Capability Maturity Model (from CMU)
    - 5-level model to define evolution of process
      - Initial, repeatable, defined, managed, and optimizing
  - ISO 9001
- **Problem: hard to revoke certification, even when team changes**

Evaluation

- **"The process of assembling evidence that a system meets, or fails to meet, a prescribed assurance target"**
  - Convince your boss you've done a good job
  - Convince customers that rely on your product
- **Who evaluates?**
  - The relying party (customers)
  - Someone else (commercial licensed evaluation facility, or CLEF): Common Criteria

Evaluation by the Relying Party

- **Insurance assessments, independent verification and validation done by NASA, military evaluation criteria (e.g., orange book)**
- **Classification example**
  - C1: no protection
  - C2: discretionary access control by single users
  - B1: mandatory access control
  - B2: structured protection (= B1+formal model)
  - B3: security domains (= B2+minimal TCB)
  - A1: verification design (= B3+formal proof TCB spec. ⇔security policy)
- **Depending on assessment of a specific system, possibility of handling several levels of classification (e.g., unrestricted to classified)**

Common Criteria (CC)

- Independent model to evaluate security
- Tries to be a generalization of former schemes, like Orange Book classification
- Provide extensive list of things to check
- Management tool for keeping track of how threats are addressed

Limitations of Common Criteria

- **Do not take into account**
  - Administrative security
  - Crypto
  - Technical-physical (e.g., emanations security)
  - Evaluation methodology
  - How standards are used
- **Reevaluations of product outside of their scope**
- **Focus a lot (too much) on technical aspects of design**

A Case study inspired by a real attack

Further reading:

https://krebsonsecurity.com/2015/09/inside-target-corp-days-after-2013-breach/
https://krebsonsecurity.com/2014/02/target-hackers-broke-in-via-hvac-company/
http://www.bloomberg.com/news/articles/2014-03-13/target-missed-warnings-in-epic-hack-of-credit-card-data

An Online Store

- T is a retail chain that has hundreds of stores nationwide. T's CEO decided to invest in T's online store. A team of web application developers is hired because they win the bid by offering the lowest cost with the most functional features and assuring the CEO of the security of the product by promising to use HTTPS connections. Once the application is developed, an in-house testing team tested the functionality of the application (e.g., websites look great, servers use HTTPS, shopping cart works, check out works fine, purchases are connected to the inventory etc.).
- The website suffered an SQL injection attack and credit card numbers were stolen. After paying for the cost of the breach, the company lost all of its profit for that year.
- ***What could the CEO have done better in terms of management, assurance, and evaluation?***

A Cost-Saving Cooling and Heating Solution

- The CEO noticed that local companies that T hires to provide heating and cooling for stores are charging a lot of money to maintain the heating and cooling. Each routine on-site check is expensive. T identified an HVAC company H that will install sensors and actuators that can be read and controlled over the internet, and therefore can lower the cost of running the stores by paying less for climate control. To work with company H, T needs only to allow H's employees to access the sensors and actuators remotely. T's stores have a single network that connects not only T's point of sale (POS) systems, but also other servers.
- T's CEO is cautious this time and asks T's security team to evaluate the risk of allowing H's employees access to the network. The team evaluates the system against the PCI (payment card industry) data security standard and concludes that as long as network access is controlled this would be compliant with the PCI standard. Therefore, T hires H and sets up an account for H on T's network.
- ***Is the security analysis adequate? What is the problem with evaluating against a standard such as PCI and Common Criteria?***

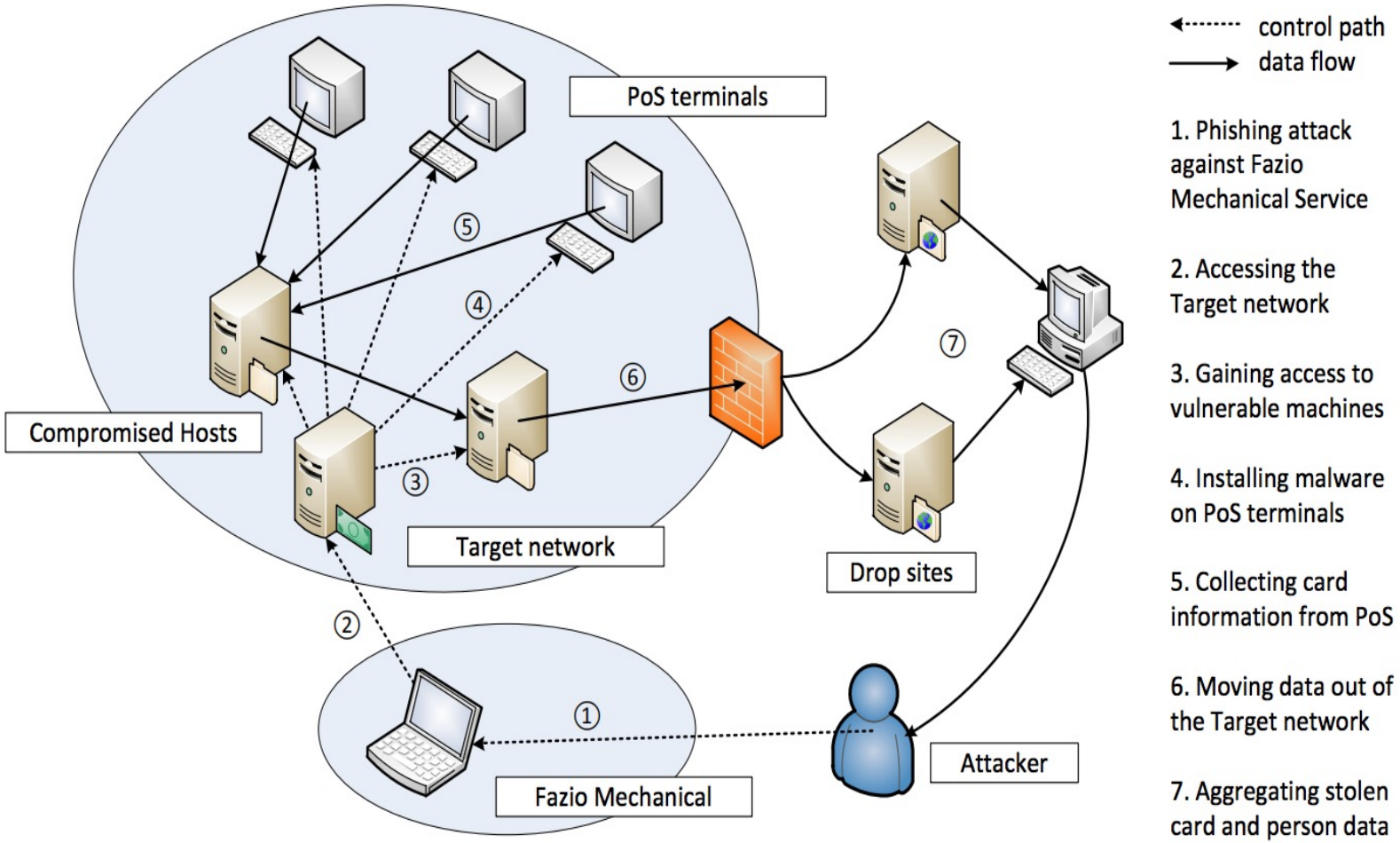
Paying More for Security with Growing Concerns

- T's business has grown significantly. Now T's network is populated with various servers, a small number of which run services that are accessible from remote affiliates. T's CEO worries that malware may be installed on computers in T's network and therefore decides to use fireeye's malware protection software, which will generate warnings to alert admins. T's security team is skeptical of fireeye even though fireeye's malware detection system received a Common Criteria certification. T's CEO decides to hire a team of specialists in Bangalore to monitor the alerts generated by the system. This decision is also met with oppositions from the security team located at the headquarters in the US, which prefers to work with a team based in the same location. The plan goes ahead anyway. The fireeye system is installed and employees are informed of the change. The security team at the headquarters is asked by the CEO to work with the team in Bangalore. The team in Bangalore, while working on installing the fireeye system, notices that several of T's servers are missing critical patches. The team informs the security team located at T's headquarters, which ignores the warning as this team sees their main task as securing the online sales sector.
- ***From the perspective of assurance evolution, why are unpatched servers common and why is this harmful?***

Attack

- The security team at Banglore receives warning messages from the fireeye software, which suggest that the computers in T's networks may be compromised. The team sends a warning to headquarters. The team at the headquarters doesn't know how to interpret the warning and is skeptical of the software to begin with, so ignores the warning. At the same time, T's customer service starts to receive calls from angry customers, claiming that their credit card numbers were stolen while they shopped at T and that they are seeing fraudulent charges. No team at T knows what is going on or how to respond. Further, T's CEO is convinced that T's system is compliant with PCI standards, so business runs as usual. Eventually, T receives a call from the FBI, and days later, finds out that they are under attack. T's CEO decides not to publicly release the number of cards that is compromised or any details about the attack. T lost $100 million+.
- ***What management issues do you see that could play into the hands of the attacker?***
- ***How do you propose to improve T's internal processes to make the infrastructure more secure?***
- ***What could T have done to respond better in this series of events?***

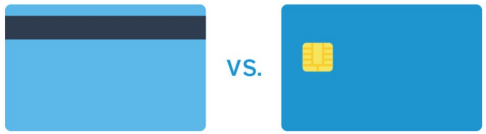
Attack – A Visual



Source: Shu, Xiaokui, Ke Tian, and Andrew Ciambrone. "Breaking the target: an analysis of target data breach and lessons learned." arXiv preprint arXiv:1701.04940 (2017).

After the Target Attack

- **Target payed $18.5 million multistate settlement,**
  - the largest ever for a data breach, to resolve state investigations of the 2013 cyber attack
  - affected more than 41 million of the company's customer payment card accounts
- **Target improved their security practices**
- **Nationwide impact**
  - The US was the last major market to still use the old-fashioned swiping system
  - The US transitioned in 2015 to using smart payment cards with chips (EMV standard)



Take Away Slide

- **Management is hard**
  - Security engineers can get inspiration from software engineering techniques, but their applicability is more restrained
  - Risk management is key – a perfectly secure system which is prohibitively expensive is of little help
- **Evaluation and assurance are harder**
  - Few standards (e.g., Common Criteria)
  - Trust in the evaluator is paramount
  - Open source (and its generalizations) seem promising approach