6.857 Computer and Network Security: Lecture Notes (Draft)

Part 1 : Introduction

Version: September 15, 1998

This course is about computer and network security. It is intended primarily for seniors and first-year graduate students. Students should feel comfortable with computer systems, networks, algorithmic concepts and modular arithmetic. Topics covered in this course include:

Physical security: Computer site security. Magnetic media. Tempest.

Cryptography:

- **Unconditional security:** One-time pad. One-time authentication.
- **Secret-key cryptography:** DES, AES.
- **Public-key cryptography:** RSA, El-Gamal; digital signatures.
- **Crypto protocols:** Secret sharing, oblivious transfer, commitment, certificate authorities, zero–knowledge proofs, key exchange, key agreement. Pay TV security.
- **Crypto policy:** Key recovery, export regulations.

Computer Security: Authentication, authorization, auditing, computer viruses, covert channels, secure operating systems, smart cards, and biometric technologies like thumbprints and eyescans. We also will discuss formal models for computer security, modes of risk analysis, methods of attacks, and security testing.

Network Security and advanced applications: We have a special focus on the Internet. We will talk about the Internet Protocol and its security issues and study firewalls and network protocols like Kerberos. We also will look at the World Wide Web, electronic commerce, and electronic payments. In addition, we will study electronic voting and electronic intellectual property protection.

1 Security Objectives

We will learn how, given a computer system, we can identify and possibly solve its security problems. This means that we have to define what we mean by security. The computer system can be anything you may think of: a shared system, a Web server, a bank ATM, the cable-box on your TV, and so on. For each case, defining security means specifying a *security policy*—a set of desired goals. For example, in an electronic voting system, we decide to set up the system so that only registered

[0]Copyright ©1998 Ronald L. Rivest. Email: `rivest@mit.edu` Web: `http://web.mit.edu/6.857/www/home.html`
All rights reserved. May be freely reproduced for educational or personal use.

voters can vote; for a Web server, we may decide that clients accessing the server should pay a fee; in a shared computer system, only authorized users should be able to log in; and so on.

A security policy may be motivated by a desire to protect resources from attack or misuse (e.g. a password is required to login to a computer), to ensure correct operation (e.g. in an electronic voting system), or to comply with relevant law (e.g. requirements for protecting the confidentiality of patient information). European laws on individual privacy tend to be much more stringent than U.S. laws.

Once a security policy has been specified we need to implement *security mechanisms* to make sure that the desired goals are met. For example, a verification process for registered voters should ensure that only registered voters can vote. A payment protocol should ensure fee payments for clients accessing a Web server. A password scheme should give access to a shared system only to authorized users.

Once the security mechanisms are put in place, they need to be reviewed in order to assess possible *vulnerabilities*, i.e., weakness that leave the system open to attacks. Most systems are not bulletproof. Hostile parties who exploit weaknesses in order to create damage are known as *threats*.

If a vulnerability is identified, then a *safeguard* or *countermeasure* must be designed to eliminate the weakness.

1.1 Security Policy: Goals

There may be many different security goals depending on the purpose of the computer system. Many goals fit, however, into the folllowing three broad categories:

Confidentiality: Contents of a computer file or of a message sent across a network should not be accessible by unauthorized parties. (For example, a student's grades should be kept private, and he should have secure remote access to them.)

Integrity: Contents of a computer file or of a message sent across a network should not be modifiable by unauthorized users. For example, only professors should be able to modify grades, and no one should be able to alter them if they are transmitted across a network. Businesses are often much more worried about integrity than confidentiality.

Availability: The system should be available to authorized users. A *denial of service attack* is an attack that infringes upon availability of the system. For example, students should be able to view their own grades.

Other goals may also come up. For example, in an electronic payment protocol, anonymity of the payee might be required. Such an example does not seem to fit neatly into any of the categories above (although confidentiality is probably the closest).

Often a system may involve several parties with conflicting goals. In an electronic payment system, the interests of the bank, the customer, and the merchant may be different. Each party may have his own security requirements.

1.2 The Design of a System

The evolution of a security system usually involves a loop of three main steps:

Design: Starting from a given system purpose and associated security policy, an system design is created, and designers present an argument that the system design is secure.

Implementation: The system is implemented according to the specified design, and its security is tested and verified to the extent possible.

Breakdown: The security system fails. The process returns to design, possibly with revised system purpose and/or security policy.

Security mechanisms evolve over long periods of time. For example, the current design of bank vaults took years to evolve; they were improved as thieves found new ways to break into them. Similarly for the design of pay phones.

It is difficult to know precisely when security has been achieved because it is much harder to prove a negative (that no security hole exists) than it is to prove a positive (that a concrete security flaw exists). Also, a security mechanism can be proved correct, but doing so requires assumptions about what potential adversaries can do (passive attacks, active attacks, etc.). Such assumptions are sometimes difficult to pin down and often new ones arise as situations change.

1.3 Mechanisms

We will explore many possible security mechanisms. Some typical ones follow.

User awareness: Educating users about the security goals of a system and its possible risks is one of the most effective security mechanisms.

Physical protection: Locks and keys prevent unauthorized access to buildings where computer systems reside. Shredders make snooping in the trash harder. Degaussers can actually destroy the contents of a disk file, compared to a DELETE or ERASE command that typically just removes a pointer to the file.

Cryptography: As we will see later in the class, encryption and other cryptographic techniques can be used in different ways to enforce both confidentiality and integrity.

Access control: Only people authorized to have privileges to certain machines, files, etc., should have access to them. In UNIX, for example, access control lists specify which users can read, write, execute a given file.

Auditing: Recording all the security-relevant activities that take place in a system makes it possible to detect security breaches that could not be prevented.

Personnel policies: Many (most?) security problems are caused by "insiders"–authorized individuals who abuse the authority given to them. Careful hiring, education, and review can help prevent such incidents.

Redundancy: Duplicating data (backups) or equipment can reduce the risk of loss of unavailability.

Some security mechanisms are aimed at *prevention* of security breaches while others are aimed at their *detection*. The latter is particularly important since we never can be sure that we have ruled out all possible attacks. It is therefore important to detect something that has gone wrong and to take necessary countermeasures.

When there are multiple parties involved with different interests, it may be important to determine in whose interest a given mechanism is operating. For example, is a smart ATM card acting in the bank's interest, the user's interests, or both? Why should each party trust the mechanism?

1.4 Principles

The design of effective security mechanisms involves some general *security principles*.

Principle of least privilege: Give a user or process only privileges needed to perform task at hand. No more, no less. This extends to the principle of "separation of privilege:" when possible, require two users to cooperate to perform a task (e.g. dual locks required to fire a missile or open a safe).

Minimize number of trusted components: Identify which components of the system need to be trusted and aim to keep those small and simple. (Also called "economy of mechanism.")

Keep it simple: If the system can't be used easily, it won't be used. Here "simple" can mean simple from the users point of view—using familiar terminology and metaphors, etc.

Be skeptical: Don't accept people's claims that something is secure. Investigate for yourself.

Be paranoid: People put a lot of effort into attacking a system. At an invited lecture at a CRYPTO conference, Robert Morris said, "Never underestimate the amount of time and effort that an adversary will put into attacking your system."

Don't aim for perfection: Perfection is basically impossible. Instead, be prepared to practice risk management, to detect problems, to design backups and countermeasures, and to recover from attacks.

2 Readings

There is no textbook for this class. However, there are a few useful books that you may want to consider purchasing. The books provide a good background on many of the issues that we will study in 6.857. They also are useful reference tools for the subject matter in general beyond the course. We will give more references as time goes on. Books listed in decreasing order of recommendation for purchase for this course.

Applied Cryptography (Second Edition), by Bruce Schneier (Wiley, 1996). A popular introduction to the subject, with an extensive bibliography. Good for this class, as it also covers many applications.

em Secure Computing: Threats and Safeguards, by Rita C. Summers (McGraw-Hill, 1997). An excellent recent survey of principles and techniques for secure computing.

Computer-Related Risks, by Peter Neumann (Addison-Wesley, 1995). A collection of case studies on how systems go wrong. Very useful and mostly fun reading.

Cryptography, Theory and Practice, by Douglas Stinson (CRC, 1995). A good, solid textbook on cryptography.

Handbook of Applied Cryptography, by Alfred Menezes, et al (CRC, 1997). A very detailed reference book on cryptography.