CPSC 467: Cryptography and Computer Security

Michael J. Fischer

Lecture 1
August 28, 2013

Highlights from Syllabus

Course Overview

Symmetric Cryptography

Highlights from Syllabus

Expectations

Read the syllabus! Some highlights:

- Pay attention to policies on plagiarism, submitting your work, and electronics in class.
- Teaching assistant is Ewa Syta.
- The midterm and final exam are both scheduled. Keep those dates clear.
- You will use the Zoo for both programming and homework submissions.
- Do problem set 0!

Also, class attendance is strongly encouraged. Why?

Course Overview

What is this course about?

The course title is *Cryptography and Computer Security*.

Here are some definitions paraphrased from Wikipedia:

- *Cryptography* is the practice and study of techniques for secure communication in the presence of third parties (called adversaries). https://en.wikipedia.org/wiki/Cryptography
- *Computer security* is information security as applied to computers and networks. https://en.wikipedia.org/wiki/Computer_security
- *Information security* means defending information from unauthoriz perusal, inspection, recording or destruction.

  https://en.wikipedia.org/wiki/Information_security

Role of cryptography

Cryptography is to information security as locks are to personal security.

- Both are clever mechanisms that can be analyzed in isolation.
- Both can be effective when used in suitable contexts.
- Both comprise only a small part of the security picture.

Information security in the real world

Some goals of information security.

- Protection against data damage.
- Protection against theft of intellectual property.
- Protection against surveillance.
- Protection against unauthorized actions.
- Protection of constitutional privacy rights.
- Protection of freedom of information.

How is security achieved in the real world?

- **Prevention:** Physical barriers, locks, encryption, firewalls, etc.
- **Detection:** Audits, checks and balances.
- **Legal means:** Laws, sanctions.
- **Concealment:** Camouflage, steganography.

Threat examples

Some risks and possible countermeasures:

- Eavesdropping on private conversations: encryption.
- Unauthorized use of a computer: passwords, physical security.
- Unwanted email: spam filters.
- Unintentional data corruption: checksums and backups.
- Denial of service: redundancy, isolation.
- Breach of contract: nonrepudiable signatures.
- Data corruption: access controls, cryptographic hash functions.
- Disclosure of confidential data: access controls, encryption, physical security.

Principles of risk management

No such thing as absolute security.

Security goal: optimize tradeoff between cost of security measures and losses from security breaches.

Security risks can be lowered by

- Reducing exposure to attack.
- Reducing number of vulnerabilities.
- Reducing value to the attacker of a successful attack.
- Increasing the cost of a successful attack.
- Increasing the penalty for a failed attempt.

Focus of this course

This course is primarily focused on the use of cryptography in information security. It will cover:

1. Classical cryptography.
2. Formal definitions of cryptographic security.
3. Cryptographic primitives: private key cryptography, public key cryptography, pseudorandom numbers, MACs, cryptographic hash functions, digital signatures.
4. Practical implementations of cryptographic primitives.
5. Cryptographic protocols for multiparty problems: contract signing, oblivious transfer, zero-knowledge proofs, bit commitment, secret-splitting, and coin flipping.
6. Brief overview of real-world applications of cryptography such as SSH, SSL, WPA, encrypted email, PGP/GPG, etc.

Computer science, mathematics and cryptography

Cryptography cuts across both computer science and mathematics.

Computer science: Cryptographic algorithms must be implemented correctly and efficiently.

Mathematics: Underlies both algorithms and their analysis.

Many cryptographic primitives are based on:

- Number theoretic problems such as factoring and discrete log;
- Algebraic properties of structures such as elliptic curves.

Understanding and modeling security uses

- Probability theory and coding theory;
- Complexity theory.

Will explore in enough depth to provide insight for how algorithms work and why they are believed secure.

Organization of this course

Roughly organized around *cryptographic primitives*. For each one:

- What can be done with it? Study of cryptographic algorithms and protocols.
  [Primary reference: Trapp & Washington.]
- What are its properties? Modeling and analysis. Requires complexity theory, probability theory, and statistics.
  [Primary reference: Goldwasser & Bellare.]
- How is it built? Requires some mathematics, particularly number theory and algebra.
  [We'll cover needed math in limited depth.]
- How is it implemented? Requires attention to detail, especially to prevent accidental leak of secret information.
  [We'll do some implementations.]

What this course is not

This course is broad rather than deep.

- It will not go deeply into the mathematics underlying cryptosystems such as RSA, AES and elliptic curves.
- It will only briefly touch on *cryptanalysis*, the flip side of *cryptography*.
- It will not go deeply into real-world security protocols.
- It will not talk about security mechanisms for computer and network devices and applications such as firewalls, operating system access controls, detecting software security holes, or dealing with web security vulnerabilities.

Example primitive: Symmetric cryptography (informal)

A *symmetric cryptosystem* (sometimes called a *private-key* or *one-key* system) is a pair of efficiently-computable functions E and D such that

- $E(k, m)$ *encrypts* *plaintext message* m using key k to produce a *ciphertext* c.
- $D(k, c)$ *decrypts* ciphertext c using k to produce a message m.

Requirements:

Correctness $D(k, E(k, m)) = m$ for all keys k and all messages m.

Security Given $c = E(k, m)$, it is hard to find m without knowing k.

Application of a symmetric cryptosystem

Secret message transmission problem:
Alice wants to send Bob a private message m over the internet.

Eve is an *eavesdropper* who listens in and wants to learn m.

Alice and Bob want m to remain private and unknown to Eve.

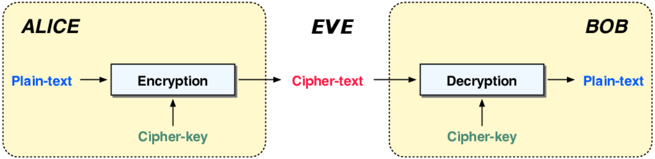


Image credit: Derived from image by Frank Kagan Gürkaynak,
`http://www.iis.ee.ethz.ch/~kgf/acacia/fig/alice_bob.png`

Solution using symmetric cryptography

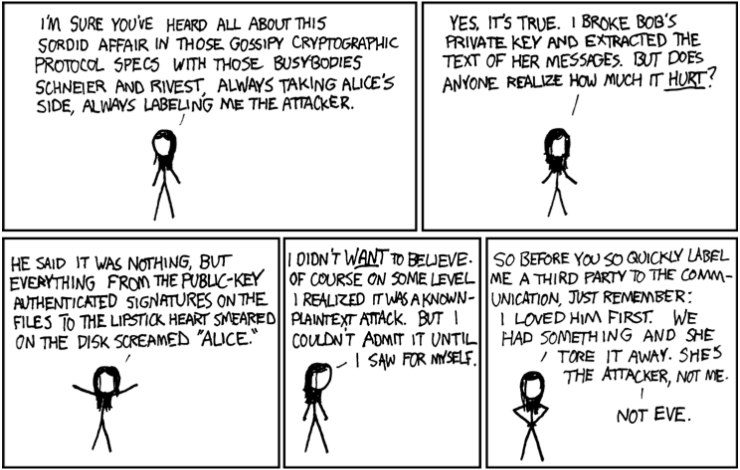
Protocol:

1. Alice and Bob share a common secret key k.
2. Alice computes $c = E(k, m)$ and sends c to Bob.
3. Bob receives c', computes $m' = D(k, c')$, and assumes m' to be Alice's message.

Assumptions:

- Eve learns nothing except for c during the protocol.
- The channel is perfect, so $c' = c$. The real world is not so perfect, so we must ask what happens if $c' \neq c$?
- Eve is a *passive eavesdropper* who can only read c but can not modify it.

Eve's side of the story



Cartoon by Randall Munroe, `https://www.xkcd.com/177/`

Requirements

What do we require of E, D, and the computing environment?

- Given c, it is hard to find m without also knowing k.
- k is not initially known to Eve.
- Eve can guess k with at most negligible success probability. (k must be chosen randomly from a large key space.)
- Alice and Bob successfully keep k secret. (Their computers have not been compromised; Eve can't find k on their computers even if she is a legitimate user, etc.)
- Eve can't obtain k in other ways, e.g., by social engineering, using binoculars to watch Alice or Bob's keyboard, etc.

Symmetric Cryptography

Symmetric cryptosystems (somewhat more formal)

A *symmetric cryptosystem* consists of

- a set $\mathcal{M}$ of *plaintext messages*,
- a set $\mathcal{C}$ of *ciphertexts*,
- a set $\mathcal{K}$ of keys,
- an *encryption* function $E : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$
- a *decryption* function $D : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$.

We often write $E_k(m) = E(k, m)$ and $D_k(c) = D(k, c)$.

Desired properties

Decipherability $\forall m \in \mathcal{M}, \forall k \in \mathcal{K}, D_k(E_k(m)) = m$. In other words, D_k is the left inverse of E_k.

Feasibility E and D, regarded as functions of two arguments, should be computable using a feasible amount of time and storage.

Security (weak) It should be difficult to find m given $c = E_k(m)$ without knowing k.

What's wrong with this definition?

This definition leaves three important questions unanswered?

1. What is a "feasible" amount of time and storage?
2. What does it mean to be "difficult" to find m?
3. What does it mean to not "know" k?

Practical considerations

These questions are all critical in practice.

1. E and D must be practically computable by Alice and Bob or the cryptosystem can't be used. For most applications, this means computable in milliseconds, not minutes or days.
2. The confidentiality of m must be preserved, possibly for years, after Eve discovers c. How long is long enough?
3. The only way to be certain that Eve does not know k is to choose k at random from a random source to which Eve has no access. This is easy to get wrong.