Course Information

CSE 107 — Introduction to Modern Cryptography

Instructor: Daniele Micciancio

Website: https://cseweb.ucsd.edu/classes/fa19/cse107-a
(not there yet, check again later tomorrow)

Slides and much course material: courtesy of Prof. Mihir Bellare. You can check/preview most of the material from last quarter webpage at https://cseweb.ucsd.edu/classes/sp19/cse107-a
Cryptography usage

*Did you use any cryptography today?*
Cryptography usage

- https invokes the TLS protocol
- TLS uses cryptography
- TLS is in ubiquitous use for secure communication: shopping, banking, Netflix, gmail, Facebook, ...
Secure messaging apps

WhatsApp, Signal, iMessage/FaceTime, Viber, Telegram, LINE, Threema, ChatSecure, KakaoTalk, ...

Use them!
Cryptography usage

Other uses of cryptography

- ATM machines
- Bitcoin
- Tor: Anonymous web browsing
- Google authenticator
- ...

11,748 android apps use cryptography (encryption), and 10,327 get it wrong [EBFK13]
What is cryptography about?

Adversary: clever person with powerful computer

Security goals:

- **Data privacy:** Ensure adversary does not see or obtain the data (message) $M$.

- **Data integrity and authenticity:** Ensure $M$ really originates with Alice and has not been modified in transit.
**Example: Medical databases**

**Doctor**
- Reads $F_A$
- Modifies $F_A$ to $F'_A$

**Database**
- Alice $F_A$
- Bob $F_B$
- Alice $F'_A$
- Bob $F_B$

- Get Alice $F_A$
- Put: Alice, $F'_A$
Example: Medical databases

Doctor

Reads $F_A$
Modifies $F_A$ to $F_A'$

Put: Alice, $F_A'$

Database

Get Alice $F_A$

Alice $F_A$
Bob $F_B$

Alice $F_A'$
Bob $F_B$

• Privacy: $F_A, F_A'$ contain confidential information and we want to ensure the adversary does not obtain them

• Integrity and authenticity: Need to ensure
  – doctor is authorized to get Alice’s file
  – $F_A, F_A'$ are not modified in transit
  – $F_A$ is really sent by database
  – $F_A'$ is really sent by (authorized) doctor
Cryptonium pipe: Cannot see inside or alter content.

All our goals would be achieved!
Cryptonium pipe: Cannot see inside or alter content.

All our goals would be achieved!

But cryptonium is only available on planet Crypton and is in short supply. 😞
Cryptographic schemes

$\mathcal{E}$: encryption algorithm  \hspace{1cm} \hspace{1cm} $K_e$: encryption key

$\mathcal{D}$: decryption algorithm  \hspace{1cm} \hspace{1cm} $K_d$: decryption key
Cryptographic schemes

\( \mathcal{E} \): encryption algorithm
\( \mathcal{D} \): decryption algorithm
\( K_e \): encryption key
\( K_d \): decryption key

**Algorithms:** standardized, implemented, public!
Cryptographic schemes

\[ E \text{: encryption algorithm} \quad K_e \text{: encryption key} \]
\[ D \text{: decryption algorithm} \quad K_d \text{: decryption key} \]

**Settings:**

- **public-key (asymmetric):** \( K_e \) public, \( K_d \) secret
- **private-key (symmetric):** \( K_e = K_d \) secret
Cryptographic schemes

$E$: encryption algorithm

$D$: decryption algorithm

$K_e$: encryption key

$K_d$: decryption key

*How do keys get distributed?* Magic, for now!
Cryptographic schemes

Our concerns:

- How to define security goals?
- How to design $E$, $D$?
- How to gain confidence that $E$, $D$ achieve our goals?
Cryptographic schemes

Computer Security: How does the computer/system protect $K_e/K_d$ from break-in (viruses, worms, OS holes, . . .)? (CSE 127,227)

Cryptography: How do we use $K_e$, $K_d$ to ensure security of communication over an insecure network? (CSE 107,207)
Why is cryptography hard?

- One cannot anticipate an adversary strategy in advance; number of possibilities is infinite.
- “Testing” is not possible in this setting.
Substitution ciphers/Caesar ciphers:

\[ K_e = K_d = \pi : \Sigma \rightarrow \Sigma, \text{a secret permutation} \]

e.g., \( \Sigma = \{A, B, C, \ldots\} \) and \( \pi \) is as follows:

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pi(\sigma) )</td>
<td>E</td>
<td>A</td>
<td>Z</td>
<td>U</td>
<td>\ldots</td>
</tr>
</tbody>
</table>

\[ \mathcal{E}_\pi(CAB) = \pi(C)\pi(A)\pi(B) \]
\[ = Z \ E \ A \]

\[ \mathcal{D}_\pi(ZEA) = \pi^{-1}(Z)\pi^{-1}(E)\pi^{-1}(A) \]
\[ = C \ A \ B \]

Not very secure! (Common newspaper puzzle)
Early history

Substitution ciphers/Caesar ciphers:

\[ K_e = K_d = \pi : \Sigma \rightarrow \Sigma, \text{a secret permutation} \]

e.g., \( \Sigma = \{A, B, C, \ldots\} \) and \( \pi \) is as follows:

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \pi(\sigma) )</td>
<td>E</td>
<td>A</td>
<td>Z</td>
<td>U</td>
<td>\ldots</td>
</tr>
</tbody>
</table>

\[ E_{\pi}(CAB) = \pi(C)\pi(A)\pi(B) \]
\[ = Z \ E \ A \]

\[ D_{\pi}(ZEA) = \pi^{-1}(Z)\pi^{-1}(E)\pi^{-1}(A) \]
\[ = C \ A \ B \]

Not very secure! (Common newspaper puzzle)
The age of machines

**Enigma**: German World War II machine

Broken by British in an effort led by **Turing**
Shannon and One-Time-Pad (OTP) Encryption

\[ K_e = K_d = \begin{align*}
K \leftarrow &\{0, 1\}^k \\
K \text{ chosen at random from } &\{0, 1\}^k
\end{align*} \]

For any \( M \in \{0, 1\}^k \)
- \( E_K(M) = K \oplus M \)
- \( D_K(C) = K \oplus C \)
Shannon and One-Time-Pad (OTP) Encryption

\[ K_e = K_d = K \overset{\$}{\leftarrow} \{0, 1\}^k \]

\[ K \text{ chosen at random from } \{0, 1\}^k \]

For any \( M \in \{0, 1\}^k \)
- \( \mathcal{E}_K(M) = K \oplus M \)
- \( \mathcal{D}_K(C) = K \oplus C \)

**Theorem (Shannon):** OTP is perfectly secure as long as only one message encrypted.

“Perfect” secrecy, a notion Shannon defines, captures mathematical impossibility of breaking an encryption scheme.

**Fact:** if \( |M| > |K| \), then no scheme is perfectly secure.
Security of a “practical” system must rely not on the impossibility but on the computational difficulty of breaking the system. (“Practical” = more message bits than key bits)

Rather than:

“It is impossible to break the scheme”

We might be able to say:

“No attack using $\leq 2^{160}$ time succeeds with probability $\geq 2^{-20}$”

I.e., Attacks can exist as long as cost to mount them is prohibitive, where Cost = computing time/memory, $$$
Security of a "practical" system must rely not on the impossibility but on the computational difficulty of breaking the system.

Cryptography is now not just mathematics; it needs to draw on computer science

- Computational complexity theory (CSE 105,200)
- Algorithm design (CSE 101,202)
The factoring problem

**Input:** Composite integer $N$

**Desired output:** prime factors of $N$

**Example:**

- Input: 85
- Output:
The factoring problem

Input: Composite integer \( N \)

Desired output: prime factors of \( N \)

Example:

Input: 85
Output: 17, 5
The factoring problem

Input: Composite integer $N$
Desired output: prime factors of $N$

Example:
  Input: 85
  Output: 17, 5

Can we write a factoring program?
The factoring problem

**Input:** Composite integer $N$

**Desired output:** prime factors of $N$

**Example:**
- Input: 85
- Output: 17, 5

Can we write a factoring program? Easy!

**Alg** Factor($N$)  // $N$ a product of 2 primes

For $i = 2, 3, \ldots, \lceil \sqrt{N} \rceil$ do
  If $N \mod i = 0$ then return $i$
The factoring problem

Input: Composite integer $N$
Desired output: prime factors of $N$

Example:
Input: 85
Output: 17, 5

Can we write a factoring program? Easy!

Alg Factor($N$) // $N$ a product of 2 primes
For $i = 2, 3, \ldots, \lceil \sqrt{N} \rceil$ do
  If $N \mod i = 0$ then return $i$

But this is very slow ...
Prohibitive if $N$ is large (e.g., 400 digits)
Can we factor fast?

- Gauss couldn’t figure out how
- Today there is no known algorithm to factor a 400 digit number in a practical amount of time.

Factoring is an example of a problem believed to be computationally hard.

**Note 1:** A fast algorithm MAY exist.

**Note 2:** A quantum computer can factor fast! One has not yet been built but efforts are underway ...

**Note 3:** Search for “post-quantum” cryptography well under way, see NIST Post-quantum standardization process, Lattice-Based Cryptography, etc.
Atomic Primitives or Problems

Examples:

- **Factoring**: Given large $N = pq$, find $p, q$
- **Block cipher primitives**: DES, AES, ...
- **Hash functions**: MD5, SHA1, SHA3, ...

Features:

- Few such primitives
- Design an art, confidence by history.

Drawback: Don't directly solve any security problem.
Atomic Primitives or Problems

Examples:

• **Factoring**: Given large $N = pq$, find $p, q$
• **Block cipher primitives**: DES, AES, ...
• **Hash functions**: MD5, SHA1, SHA3, ...

Features:

• **Few such primitives**
• **Design an art**, confidence by history.
Atomic Primitives or Problems

Examples:

- Factoring: Given large $N = pq$, find $p, q$
- Block cipher primitives: DES, AES, ...
- Hash functions: MD5, SHA1, SHA3, ...

Features:

- Few such primitives
- Design an art, confidence by history.

Drawback: Don’t directly solve any security problem.
Goal: Solve security problem of direct interest.

Examples: encryption, authentication, digital signatures, key distribution, 
...
Higher Level Primitives

Goal: Solve security problem of direct interest.

Examples: encryption, authentication, digital signatures, key distribution, ...

Features:

• Lots of them
We typically design high-level primitives from atomic ones
Defining security

A great deal of design tries to produce schemes without first asking:

“What exactly is the security goal?”

This leads to schemes that are complex, unclear, and wrong.

Being able to precisely state what is the security goal of a design is challenging but important.

We will spend a lot of time developing and justifying strong, precise notions of security.

Thinking in terms of these precise goals and understanding the need for them may be the most important thing you get from this course!
What does it mean for an encryption scheme to provide privacy?
What does it mean for an encryption scheme to provide privacy?

Does it mean that given $C = E_{K_e}(M)$, adversary cannot

- recover $M$?
- recover the first bit of $M$?
- recover the XOR of the first and the last bits of $M$?
- ...
Defining Security

What does it mean for an encryption scheme to provide privacy?

Does it mean that given $C = E_{K_e}(M)$, adversary cannot

- recover $M$?
- recover the first bit of $M$?
- recover the XOR of the first and the last bits of $M$?
- ... 

We will provide a formal definition for privacy, justify it, and show it implies the above (and more).
Cryptography in practice

Schemes designed via the principles we will study are in use (TLS, SSH, IPSec, . . .): HMAC, RSA-OAEP, ECIES, Ed25519, CMAC, GCM, . . .
Cryptography uses

- Number theory
- Combinatorics
- Modern algebra
- Probability theory
Hardy, in his essay *A Mathematician’s Apology* writes:

“Both Gauss and lesser mathematicians may be justified in rejoicing that there is one such science [number theory] at any rate, and that their own, whose very remoteness from ordinary human activities should keep it gentle and clean”

No longer: Number theory is the basis of modern public-key systems such as RSA.
Security today

- Server breaches, malware
- Compromise of people’s private information leading to identity theft, credit-card fraud, ...
- Lack of privacy: Information about us is collected and harvested
- Mass surveillance: Snowden Revelations

2017 Equifax breach exposed 143 million social security numbers.

Cryptography is a central tool in getting more security and privacy.
Computing on encrypted data

- Searchable encryption
- Homomorphic encryption
- multi-party computation
- garbled circuits
- ...
What you can get from this course

Be able to

• Identify threats
• Evaluate security solutions and technologies
• Design high-quality solutions
• Develop next-generation privacy tools
• ...

If nothing else, develop a healthy sense of paranoia!
Resources:

- Lecture *slides*, on webpage
- Lecture podcasts, podcast.ucsd.edu
- Lecture attendance/participation still mandatory!
- Homework solutions, handed out in discussion, NOT on webpage
- PlayCrypt: Software used for assignments
- Piazza, mandatory
- Instructor/TA/Tutor office hours

No textbook.
Read course information sheet!

- to be distributed on webpage
- hard copies will also be handed out in the next lecture (Tuesday)

Once you have read it, sign and turn in affirmation by due date.
Grade Computation

Denote your percentage scores on problem sets (homeworks), the midterms and the final by H,M,F respectively. Grade is determined by total score TS where

\[
RS = \frac{20}{100} \cdot H + \frac{40}{100} \cdot M + \frac{40}{100} \cdot F \\
TS = \frac{90}{100} \cdot RS + \frac{10}{100} \cdot DS.
\]

Here DS is your **Discretionary Score**:

- Its default value is your raw score RS
- It can go **up** due to class or discussion participation, impressing instructors, answering Piazza posts, ...
- It can go **down** due to requesting exceptions to policies, asking already answered administrative questions, requesting actions already denied by policies, asking for special consideration, ...

Homework announcements are made on Piazza.

Homeworks use our Python-based PlayCrypt software. (Currently being ported from python2.7 to python3.)

Midterm exams, during regular class time, dates TBA. Discussion is MANDATORY, homework solutions will be presented and/or distributed in discussion. (No formal attendance, but you are responsible for it.)

There will be 2 midterms, and 6-8 (possibly multi-part) homework assignments.
Looking at solutions from previous years of the course or finding them on the Internet is not allowed.

Exams (midterms and final) are open notes, but must be your own notes, and no other materials is allowed (problem sets or their solutions, your homeworks, PlayCrypt, calculators, computers, iPads, cellphones, ...)

Copying or using other people’s code / software is of course not allowed.
Grading

Strive for neat, mathematically precise and well-written solutions.

Quality of exposition will impact score.
This is a theory course! Largely definitions and proofs, although of applied value.

Pre-requisites: undergraduate algorithms (101) and theory of computation (105), some probability theory, and

**Mathematical Maturity**

If you got C+ or worse in CSE 101 or CSE 105, your chances of passing 107 are slim and it is recommended you not take it.

**127:** If you are good at programming and systems, did well in CSE 120

**107:** If you are good at theory and math, did well in CSE 105.
How to do well in CSE 107

Characteristics of the successful 107 student:

• More interested in learning than grades
• Likes challenges, does not give up easily
• Tries to understand *all* the materiel, not just some of it
• Questions are more often about the materiel (slides) than about the homework.
• Limited need, or requests for, examples.

If you take the course with the view that you only want to pass, you may fail. If you take it aiming to get an A and are willing to work for it, you may very well get one.
How to do well in CSE 107

**Doesn’t work:** Random access mode, in which you look at homework or quiz problem, then try to find something in slides that “matches” it.

**Works:** Sequential mode, where you first go through all the slides, sequentially, and make sure you understand the materiel, and THEN attempt homework and quizzes.

Some students ask for more examples. Try instead to understand the theory without the aid of examples. Then, everything looks easy.

Some students expect a **recipe for success:** “I am willing to work hard. Just tell me what to do!”

We can give no such recipe. Different people understand things in different ways and have different paths to success. Work to find your own path.
Sometimes we hear: “I understand the materiel but have trouble doing the homework or quizzes.”

This is an unproductive attitude. If you have trouble with homeworks and quizzes, then you don’t understand the materiel, by definition. If you think you do, you put yourself beyond help. Instead, identify the gaps in understanding. Then we can help you.
Is CSE 107 for me?

- Nobody is good at everything
- Everyone is good at something

Life, and college, are not about doing well at everything but at finding what you do well and what you like doing.

This course fits well with some (mostly people who like theory) but not others. Figure out where you fall QUICKLY and move on if the class is not a match for you.