Course Information

CS 838 — Applied Cryptography

Instructor: Thomas Ristenpart

Website: http://pages.cs.wisc.edu/ rist/cs838/
Cryptography usage

Did you use any cryptography

• today?
Did you use any cryptography

- today?
- over the last week?
Cryptography usage

Did you use any cryptography

- today?
- over the last week?
- over the Christmas break?
https invokes the Secure Socket Layer (SSL) communication security protocol to securely transmit your credit card number to the server

- SSL uses cryptography
Other uses of cryptography

- ATM machines
- On-line banking
- Remote login and file transfer using SSH
- X-Box, PlayStation 3
What is cryptography about?

Adversary: clever person with powerful computer

Goals:
- Data privacy
- Data integrity and authenticity
The goal is to ensure that the adversary does not see or obtain the data (message) $M$.

**Example:** $M$ could be a credit card number being sent by shopper Alice to server Bob and we want to ensure attackers don’t learn it.
The goal is to ensure that

- $M$ really originates with Alice and not someone else
- $M$ has not been modified in transit
Integrity and authenticity example

Adversary Eve might

- Modify “Charlie” to “Eve”
- Modify “$100” to “$1000”

Integrity prevents such attacks.
Medical databases

Doctor

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

Database

Get Alice $F_A$ $F_A'$

Put: Alice, $F_A'$

Alice $F_A$
Bob $F_B$

Alice $F_A'$
Bob $F_B$
Medical databases

Doctor

Reads $F_A$
Modifies $F_A$ to $F'_A$

Get Alice $F_A$

Put: Alice, $F'_A$

Database

<table>
<thead>
<tr>
<th>Alice</th>
<th>$F_A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>$F_B$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Alice</th>
<th>$F'_A$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>$F_B$</td>
</tr>
</tbody>
</table>

- Privacy: $F_A, F'_A$ contain confidential information and we want to ensure the adversary does not obtain them
Medical databases

Doctor

Reads $F_A$
Modifies $F_A$ to $F'_A$

Get Alice $F_A$

Put: Alice, $F'_A$

Database

<table>
<thead>
<tr>
<th></th>
<th>$F_A$</th>
<th>$F_B$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 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
What is cryptography about?

Adversary: clever person with powerful computer

Goals:
- Data privacy
- Data integrity and authenticity
Ideal World

Secure channel: Cannot see inside or alter content.
Secure channel: Cannot see inside or alter content.

All our goals would be achieved!
Secure channel: 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

\[ \mathcal{E} : \text{ encryption algorithm} \quad K_e : \text{ encryption key} \]

\[ \mathcal{D} : \text{ decryption algorithm} \quad K_d : \text{ decryption key} \]
Cryptographic schemes

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

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

$\mathcal{E}$: encryption algorithm
$\mathcal{D}$: decryption algorithm

$K_e$: encryption key
$K_d$: decryption key

Settings:

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

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

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

Our concerns:

• How to define security goals?
• How to design $\mathcal{E}$, $\mathcal{D}$?
• How to gain confidence that $\mathcal{E}$, $\mathcal{D}$ achieve our goals?
Why is cryptography hard?

- One cannot anticipate an adversary strategy in advance; number of possibilities is infinite.
- "Testing" is not possible in this setting.
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:

\[
\begin{array}{c|ccccc}
\sigma & A & B & C & D & \ldots \\
\hline
\pi(\sigma) & E & A & Z & U & \ldots \\
\end{array}
\]

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

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

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

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

Not very secure! (Common newspaper puzzle)
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

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

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

For any \( M \in \{0, 1\}^k \)

\[ \mathcal{E}_K(M) = K \oplus M \]

\[ \mathcal{D}_K(C) = K \oplus C \]
Shannon and One-Time-Pad (OTP) Encryption

\[ K_e = K_d = \begin{cases} \$ & \{0, 1\}^k \end{cases} \]

\( K \) chosen at random from \{0, 1\}^k

For any \( M \in \{0, 1\}^k \)

- \( E_K(M) = K \oplus M \)
- \( D_K(C) = K \oplus C \)

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
- Algorithm design
Classical Approach: Iterated design

Scheme 1.1
Scheme 1.1  →  bug!
Classical Approach: Iterated design

Scheme 1.1 $\rightarrow$ bug!
\[\downarrow\]
Scheme 1.2
Classical Approach: Iterated design

Scheme 1.1 $\rightarrow$ bug!

↓

Scheme 1.2 $\rightarrow$ bug!
Classical Approach: Iterated design

Scheme 1.1 $\rightarrow$ bug!
\[ \downarrow \]
Scheme 1.2 $\rightarrow$ bug!
\[ \downarrow \]
\[ \vdots \]
\[ \downarrow \]
Scheme 1.\(n\)
Classical Approach: Iterated design

Scheme 1.1 $\rightarrow$ bug!

↓

Scheme 1.2 $\rightarrow$ bug!

↓

\vdots

↓

Scheme 1.n $\rightarrow$ deploy
Classical Approach: Iterated design

Scheme 1.1 $\rightarrow$ bug!

\[ \Downarrow \]

Scheme 1.2 $\rightarrow$ bug!

\[ \Downarrow \]

\[ \vdots \]

\[ \Downarrow \]

Scheme 1.\(n\) $\rightarrow$ deploy $\rightarrow$ bug!
Good cryptography

- Understanding the goals: Formal adversarial models and definitions of security goals

- Beyond iterated design: Proof by reduction that a construction achieves its goal
Defining security

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

“What exactly is the security goal?”

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

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!
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($N$) \quad // $N$ 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** \( \text{Factor}(N) \)  // \( N \) 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
- Nor does anyone know now

Nobody today knows how to factor a 400 digit number in a practical amount of time.
Provable Security

Provide

- A scheme
- A proof of security

The proof should establish something like:

“The only way to break the scheme is to factor a large number”

or, put another way

“If an adversary breaks the scheme, it must have found a fast factoring algorithm.”
Being able to break scheme implies

- attacker has found a way to factor fast
- attacker is smarter than Gauss
- and smarter than all living mathematicians...

or

- the adversarial model was wrong!
Atomic Primitives or Problems

Examples:

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

Examples:

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

Features:

- Few such primitives
- Bugs rare
- 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, ...

Features:

- Few such primitives
- Bugs rare
- 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, ...
Goal: Solve security problem of direct interest.

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

Features:

- Lots of them
- Bugs common in practice
We typically design high-level primitives from atomic ones

History shows that the Transformer is usually the weak link:
- Atomic primitives secure, yet
- Higher level primitive insecure
Enables us to get transformers for which we can guarantee

Atomic primitive secure $\Rightarrow$ High-level primitive secure

I.e., If attacker breaks encryption scheme then they are smarter than Gauss.
Provable security in practice

Proven-secure schemes in use (SSL, SSH, IPSec, ...):

• HMAC
• OAEP
• ECIES
• ...
New uses for old mathematics

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.
Parties 1, 2, 3, . . . , n.
Party \( i \) has the integer \( x_i \in \{0, \ldots, M - 1\} \)

They want to know

\[
x = \frac{x_1 + \ldots + x_n}{n}
\]

but each party \( i \) wants to keep its own \( x_i \) private.
Cryptography beyond communication security

Parties 1, 2, 3, . . . , n.
Party $i$ has the integer $x_i \in \{0, \ldots, M - 1\}$

They want to know

$$x = \frac{x_1 + \ldots + x_n}{n}$$

but each party $i$ wants to keep its own $x_i$ private.

Usage:

$$x_i = \text{score of student } i \text{ on homework 1}$$

$$x_i = \text{vote of party } i \text{ for proposition } X \text{ on ballot}$$

\[ \vdots \]
Parties 1, 2, 3, ..., n.
Party $i$ has the integer $x_i \in \{0, \ldots, M - 1\}$
They want to know
\[ x = \frac{x_1 + \ldots + x_n}{n} \]
but each party $i$ wants to keep its own $x_i$ private.

**Trusted Party Solution:**
Cryptography beyond communication security

Parties 1, 2, 3, . . . , n. Party \( i \) has the integer \( x_i \in \{0, \ldots, M - 1\} \)

They want to know

\[
x = \frac{x_1 + \ldots + x_n}{n}
\]

but each party \( i \) wants to keep its own \( x_i \) private.

**Trusted Party Solution:**

**Secure Computation:** Allows us to accomplish objective without a trusted party, using only (secure) communication between parties.
Internet Gambling

Will you play?
Internet Gambling

Will you play?

Casino can cheat. It returns 😞, $T$ for some $T \neq g$
Will you play?

Casino can cheat. It returns 😞, \( T \) for some \( T \neq g \)

Crypto can fix this!
Security today

- Millions of dollars of loss due to credit-card fraud, phishing, identity theft, ...
- Lack of privacy: Enormous amounts of information about each of us is collected and harvested by businesses dedicated to this purpose

Cryptography is a central tool in getting more security and privacy
Cryptography in the real world

Central uses: SSL, SSH, TLS, IPSEC, ...
Cryptography in the real world

- Poor exposition: Incomplete, unclear scheme specifications in documents
- Lack of precise goal formulations
- Complex, unclear or incorrect schemes

Lack of cryptographic education and skill in workforce.
What you can get from this course

You can get the ability to

• Identify threats
• Evaluate security solutions and technologies
• Design high-quality solutions
• Write clear, complete scheme specifications
• Begin research in cryptography

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

- Lecture slides
- Course notes
- Research papers

No textbook.

All resources will be on course web page.
• Read course information sheet!
  Handout today and on course webpage.

• The course will require:
  – Homeworks
  – Short write-up on course project (5 pages or less)
  – Final presentation on course project (10-20 minutes)
  – Final discussion with me

• Grades will be based on my assessment of how well you did on the above endeavors. Try to learn something, have fun, and you’ll end up with a high grade.
Homeworks

- Homeworks must be written up individually. If a problem is discussed with others (in the class or otherwise), then the writeup should explicitly indicate this.
- Writeups are strongly encouraged to be typeset in LaTeX if you want me to read them
- Finding solutions on the Internet is not allowed
Projects

- In-depth investigation of some topic in applied or theoretical crypto
- Individual or small group
- Examples: analyse a proposed standard or implementation, insightful comments on a research paper, extend the OpenSSL codebase in some meaningful way, new cryptographic research result, etc.
- Short presentation to class at end of term
- Projects must be approved by me. 1 page proposal due February 8, meetings following week to discuss.
- Probably have short project progress meetings once a week during last 5 weeks of class
This is a theory course! Largely definitions and proofs, although of applied value.

Needed: undergraduate algorithms and theory of computation, some probability theory, a little calculus, and

**Mathematical Maturity**
Question: What is the cost of multiplying two $k$-bit numbers?
Warm-up

**Question:** What is the cost of multiplying two $k$-bit numbers?

**Answer:** $O(k^2)$

\[
\begin{array}{cccccccc}
1 & 0 & 1 & 1 & 1 & 0 \\
\times & & & & 1 & 0 & 1 \\
\hline
1 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
+ & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
\hline
1 & 1 & 1 & 0 & 0 & 1 & 1 & 0
\end{array}
\]
Question: I have a coin with probability $p$ of HEADS. I flip it $n$ times.

\[ \Pr[\text{at least one HEADS}] = \frac{51}{55} \]
Question: I have a coin with probability $p$ of HEADS. I flip it $n$ times.

$$\Pr[\text{at least one HEADS}] = pn$$

Because I flip $n$ coins and each has probability $p$ of being HEADS.
Question: I have a coin with probability $p$ of HEADS. I flip it $n$ times.

$$\Pr[\text{at least one HEADS}] = pn$$

WRONG! Why?

Say $p = \frac{1}{2}$ and $n = 3$. Then the “probability” is

$$pn = \frac{1}{2}(3) = \frac{3}{2} > 1 ??$$
Question: I have a coin with probability $p$ of HEADS. I flip it $n$ times.

$$\Pr[\text{at least one HEADS}] = pn$$

WRONG! Why?

Let $H_i$ be the event that the $i$-th flip is heads.

$$\Pr[H_i] = p \text{ for all } 1 \leq i \leq n$$

$$\Pr[\text{at least one HEADS}] = \Pr[H_1 \lor H_2 \lor \cdots \lor H_n]$$

but this is not equal to

$$\Pr[H_1] + \cdots + \Pr[H_n]$$
Example: $n = 2$

$$
\Pr [H_1 \lor H_2] = \Pr [H_1] + \Pr [H_2] - \Pr [H_1 \land H_2]
$$

Is there another way to compute

$$
\Pr [\text{at least one HEADs}]
$$
Question: I have a coin with probability $p$ of Heads. I flip it $n$ times.

$$\Pr[\text{at least one Heads}] = 1 - \Pr[\text{all Tails}]$$

$$= 1 - (1 - p)^n$$