Organisation

- 6 cours (André Schrottenloher)
- 7 TDs (Clémence Chevignard)
- 1 DM
- 1 examen final

Matériel de cours (notes de cours, plan, slides, TDs, corrigés. . .) sur:
`https://andreschrottenloher.github.io/pages/teaching-2026.html`

E-mail:
andre(dot)schrottenloher(at)inria(dot)fr

(Je suis aussi sur Discord).

Introduction to Cryptography
Part I: Introduction – Defining Security

André Schrottenloher

Inria Rennes
Team CAPSULE

What is crypto today?

Definition

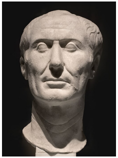
Protect **information** transmitted through **insecure channels** in presence of **adversaries** with the power to **listen** to and **corrupt** the transmitted messages.

This science borrows tools from:

- Information theory;
- Complexity theory and algorithms;
- Probabilities;
- Proof systems;
- (Computational) algebra;
- Quantum information theory.

History

Historical ciphers



Caesar

Al-Kindi (born 801 AD)

Caesar cipher

- Shift the alphabet by a fixed number
- Easy to break if you know the trick (only 26 possibilities, visible patterns...)

Substitution cipher

- Choose a **permutation of the letters** of your alphabet
- 26! possibilities
- Break by **frequency analysis**, **bigrams** and **probable words**

Pictures from Wikimedia Commons

Historical ciphers (ctd.)

Vigenère cipher

- Shift (like Caesar) but using a random, repeated keyword.
- Guess key length, analyze frequencies.

Rotor machines

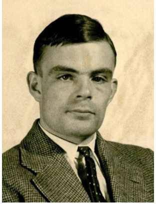
After the typewriter, encryption based on rotor machines (e.g., the Enigma family).

- Rotor encodes the key
- Typed symbol encrypted with the next symbol on the rotor
- Rotor moves as you type, changing the key each time

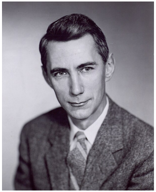
Cryptanalysis of Enigma

- First breaks in the 1930s by Polish cryptographers
- First "cryptologic bombs" used for cipher-breaking
- During the war: upgrade of the bombs by the British & the US, allowing to break the 4-rotor version

Dawn of modern cryptography (ca. 1950)



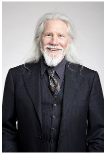
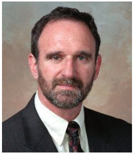
Turing: co-inventor of modern-day computer science, well-known for his code-breaking work during WWII



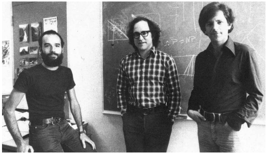
Shannon: information theory, information-theoretic security, and cipher design.

Pictures from Wikimedia Commons

Dawn of asymmetric cryptography (ca. 1970)

Diffie & Hellman: introduction of the Diffie-Hellman key-exchange, and mathematical foundations of public-key cryptography



Rivest, Shamir & Adleman: RSA cryptosystem (which became the most popular)

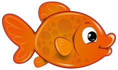
Credit: the Royal Society (Wikimedia Commons)

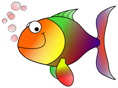
The modern era

- Network protocols (HTTPS, SSL, TLS, PGP, wifi, 5G)
- Encrypted messaging apps
- Hardware: credit cards, DVD, Blu-ray
- Anti-piracy software
- Electronic voting...

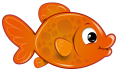
Principles

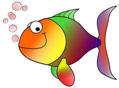
Cryptography = **protecting information** exchanged on **insecure channels** in the presence of **malicious, powerful adversaries**.





Cryptography = **protecting information** exchanged on **insecure channels** in the presence of **malicious, powerful adversaries**.





Security can be either:

- Proven unconditionally (**information-theoretic**)
- Reduced to **computational assumptions**

$\Longrightarrow$ more general in public-key crypto, more ad hoc in secret-key crypto.

What we want to achieve (important slide!)

- **Confidentiality**: the transmitted information remains secret $\Longrightarrow$ **encryption**
- **Authenticity**: guarantees that the transmitted information has indeed be sent by Alice (resp. Bob)
- **Integrity**: guarantees that the transmitted information has not been tampered with
- **Non-repudiation**: guarantees that parties cannot later deny being the author of a message

And nowadays, even more goals: multi-party secure computation, secret sharing, proofs of knowledge, computation on encrypted data, etc.

Modern-day crypto constraints

Designing secure cryptography is not easy, but what's most difficult is to make it secure **and** cost-efficient ("lightweight").

- **Latency:** the time to perform a key-exchange is counted in milliseconds;
- **Energy:** crypto on small, battery-operated devices has to use the minimal number of operations possible;
- **Circuit size:** crypto on embedded chips (e.g., smart cards) has to use the smallest possible circuits. This puts also constraints on key sizes.

Our goal nowadays is to **minimize computational resources** for a given **security level**.

Cryptography building blocks

Primitives

A primitive is a building block that offers a "low-level" functionality. Example: an asymmetric / symmetric cipher, a signature, a block cipher, stream cipher, etc.

Protocols

A protocol specifies an entire communication process. It makes use of primitives as "black boxes" (for example, you can use any block cipher).

The security of a protocol is **reduced** to the security of the primitives: if the primitives are secure, the protocol is secure.

Crypto design process

1. Some people design a primitive
2. They do **their own security analysis**
3. They **publish the result** and make **security claims**
4. Everybody else tries to cryptanalyze (and contradict the claims)
5. After some time, we gain **trust**, and some institution (ISO, IETF) may standardize the scheme

$$\text{Trust} = \int_{t=0}^{+\infty} \text{Cryptanalysis effort } dt$$

"Real" software only uses well-established, publicly audited, standardized designs (RSA, ECC, AES-GCM).

Perfect Security

Symmetric cipher

Let $\mathcal{K}$, $\mathcal{M}$, $\mathcal{C}$ be the **key space**, **plaintext space** and **ciphertext space**.

A **symmetric cipher** is a triple of PT algorithms $\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}$ with signature:

$$\begin{cases} \mathsf{KeyGen} & : \emptyset \to \mathcal{K} \\ \mathsf{Enc} & : \mathcal{K} \times \mathcal{M} \to \mathcal{C} \\ \mathsf{Dec} & : \mathcal{K} \times \mathcal{C} \to \mathcal{M} \end{cases}$$

and satisfying the **correctness property**:

$$\forall k \in \mathcal{K}, \forall m \in \mathcal{M}, \mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m \ .$$

We assume that all ciphertexts are accessible.

The algorithms $\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}$ are randomized, poly-time and public (per Kerckhoffs' principles).

Perfect security (= information-theoretic security)

A symmetric cipher is **perfectly secure** if:

- for any random variable M over $\mathcal{M}$;
- any message $m \in \mathcal{M}$;
- any ciphertext $c \in \mathcal{C}$:

$$\Pr\left[M = m | \mathrm{Enc}(\mathrm{KeyGen}, M) = c\right] = \Pr\left[M = m\right] \quad .$$

A symmetric cipher is **perfectly secure** if for any $m_1, m_2, c \in \mathcal{M} \times \mathcal{M} \times \mathcal{C}$:

$$\Pr_{k \leftarrow \mathrm{KeyGen}}\left[\mathrm{Enc}(k, m_1) = c\right] = \Pr_{k \leftarrow \mathrm{KeyGen}}\left[\mathrm{Enc}(k, m_2) = c\right] \quad .$$

Proof in TD.

The One-time Pad (Vernam's cipher)

$\mathcal{K} = \mathcal{M} = \mathcal{C} = \{0,1\}^n$
KeyGen: $k \hookleftarrow U(\mathcal{K})$
$\mathsf{Enc}(k, m) = m \oplus k$
$\mathsf{Dec}(k, c) = c \oplus k$

It's correct:

$$\forall k \in \mathcal{K}, \forall m \in \mathcal{M}, \mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m \ .$$

Lemma
The One-Time Pad has perfect security.

The One-time Pad (Vernam's cipher)

$\mathcal{K} = \mathcal{M} = \mathcal{C} = \{0,1\}^n$
KeyGen: $k \hookleftarrow U(\mathcal{K})$
$\mathrm{Enc}(k, m) = m \oplus k$
$\mathrm{Dec}(k, c) = c \oplus k$

It's correct:

$$\forall k \in \mathcal{K}, \forall m \in \mathcal{M}, \mathrm{Dec}(k, \mathrm{Enc}(k, m)) = m \quad .$$

Lemma
The One-Time Pad has perfect security.

Proof: let $K = \mathrm{KeyGen}()$

$$\forall m, c, \Pr[M = m | \mathrm{Enc}(K, M) = c] = \Pr[M = m | M \oplus K = c]$$
$$= \Pr[m \oplus K = c] = \Pr[K = m \oplus c] \quad .$$

K is uniform, so whichever c and m one has: $\Pr[K = m \oplus c] = 2^{-n}$.

The One-time Pad (ctd.)

The One-Time Pad is not a very practical cipher . . .

The One-time Pad (ctd.)

The One-Time Pad is not a very practical cipher ...

- You can only use the key once;
- How can you transmit such a key?

It would be much better to have a **small key** that you could somehow **expand**, i.e., a **stream cipher**.

$\Longrightarrow$ but can such a cipher have perfect security?

Shannon's theorem

Lemma

Perfect security implies $|\mathcal{K}| \geq |\mathcal{C}| \geq |\mathcal{M}|$.

Theorem

Let $\mathrm{KeyGen}, \mathrm{Enc}, \mathrm{Dec}$ be a symmetric cipher on $\mathcal{K}, \mathcal{M}, \mathcal{C}$ such that $|\mathcal{K}| = |\mathcal{M}| = |\mathcal{C}|$. It has perfect security **iff**:

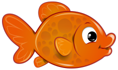
- Each key is chosen with probability $1/|\mathcal{K}|$
- For all $m \in \mathcal{M}, c \in \mathcal{C}$, there is a unique k such that $\mathrm{Enc}(m, k) = c$.

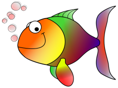
Proof in TD

What is an Adversary?

A cryptographic scheme

. . . has several participants nicknamed Alice, Bob, Charlie, etc. (In this course: Alice & Bob).





The **adversary** (Eve) may **listen to** or **modify** the exchanged communications between Alice and Bob.

- **Alice, Bob and Eve are algorithms**
- They are randomized
- Eve is a Monte-Carlo algorithm (makes errors)

Definition of security

An adversary can **always win** with some probability, for example they may guess the key correctly.

But an adversary is not successful unless they run in polynomial time, and succeed with large probability.

Definition

A scheme is secure (for some security notion) if any **efficient** adversary can only attack (this security notion) with **negligible** probability.

Let n be the **security parameter** of the scheme:

- efficient $= \mathrm{poly}(n)$ (PPT algorithm)
- negligible $= o(n^{-c})$ for any constant c, i.e., smaller than any inverse polynomial

Indistinguishability

Statistical indistinguishability

Definition

Let X, Y be two random variables on a set A. Their **statistical distance** is:

$$\Delta(X, Y) = \frac{1}{2} \sum_{a \in A} \left| \Pr[X = a] - \Pr[Y = a] \right| .$$

It's indeed a distance.

Two distributions D_0, D_1 are **statistically indistinguishable** if there is a negligible function negl such that: $\Delta(D_0, D_1) \leq \mathrm{negl}(n)$.

Computational indistinguishability

Two distributions are computationally indistinguishable if no **efficient algorithm** can distinguish from them.

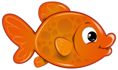
$\implies$ given access to **samples** of D, decide if $D = D_0$ or $D = D_1$.

We formalize this using **games.**

Distinguishing games

Let D_0, D_1 be two distributions over $\{0,1\}^n$. The distinguishing games G_0, G_1 are defined as follows.

The adversary $\mathcal{D}$ communicates with a challenger $\mathcal{C}$.



$$\xrightarrow{x \hookleftarrow D_b}$$

Return b'

- During the game $\mathcal{D}$ may perform a **query**: the challenger will return $x \hookleftarrow D_b$
- At the end $\mathcal{D}$ returns a bit b'

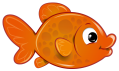
The **advantage** of $\mathcal{D}$ is:

$$\mathrm{Adv}(\mathcal{D}) = \left| \Pr\left[\mathcal{D} \xrightarrow{G_0} 1\right] - \Pr\left[\mathcal{D} \xrightarrow{G_1} 1\right] \right| .$$

$\mathcal{D}$ is a **distinguisher** if the advantage is non-negligible.

Second definition

In this second definition we use a **single game** G.



Choose $b \hookleftarrow U(0,1)$

$$\xrightarrow{\quad x \hookleftarrow D_b \quad}$$

Return b'

- **Initialization**: $\mathcal{C}$ chooses a bit $b \in \{0,1\}$ u.a.r.
- **Queries**: $\mathcal{C}$ will respond with $x \hookleftarrow D_b$
- **Finalization**: $\mathcal{D}$ will return a bit b'. If $b = b'$, $\mathcal{D}$ **wins** the game

$\mathcal{D}$ is a distinguisher if $\Pr[Win] \geq 1/2 + \varepsilon$ for some non-negligible ε.

Computational indistinguishability

D_0, D_1 are computationally indistinguishable if $\forall$ PPT adversary $\mathcal{D}$:

$$|\Pr\left[\mathcal{D} \xrightarrow{G_0} 1\right] - \Pr\left[\mathcal{D} \xrightarrow{G_1} 1\right]| \leq \text{negl}(n)$$

This is equivalent to: $\forall$ PPT adversary $\mathcal{D}$:

$$|\Pr[Win] - 1/2| \leq \text{negl}(n)$$

Proof of the equivalence by double reduction:

1. from a PPT distinguisher $\mathcal{A}$ for the first definition, create a PPT distinguisher for the second
2. conversely

Proof of the equivalence

Let $\mathcal{A}$ be a distinguisher for the **second** definition (single game). Let $\mathcal{A}'$ that runs in the **first** definition (two games) as follows: it simply runs $\mathcal{A}$.

We relate the advantage of $\mathcal{A}$ (in the second definition) with the one of $\mathcal{A}'$ (in the first definition) as follows:

$$\begin{aligned}
\mathrm{Adv}(\mathcal{A}') &= \left|\Pr\left[\mathcal{A}' \xrightarrow{G_0} 1\right] - \Pr\left[\mathcal{A}' \xrightarrow{G_1} 1\right]\right| \\
&= \left|\Pr\left[\mathcal{A} \to 1 | b = 0\right] - \Pr\left[\mathcal{A} \to 1 | b = 1\right]\right| \text{ (now running in } G) \\
&= \left|\,\Pr\left[b' = 1 | b = 0\right] - \Pr\left[b' = 1 | b = 1\right]\,\right| \\
&= \left|1 - \Pr\left[b' = 0 | b = 0\right] - \Pr\left[b' = 1 | b = 1\right]\,\right| \\
&= \left|1 - 2\Pr\left[b' = 0 \wedge b = 0\right] - 2\Pr\left[b' = 1 \wedge b = 1\right]\,\right| \\
&= \left|1 - 2\Pr\left[Win\right]\,\right| = \frac{1}{2}\mathrm{Adv}(\mathcal{A}) \quad .
\end{aligned}$$

Where we used the uniformity of b. So, if there exists a distinguisher for the first definition (non-negligible advantage), then there is one for the second, and conversely.

Relation between statistical & computational indistinguishability

Let X, Y be two random variables on a set A.

If the distributions X and Y are statistically indistinguishable ($\Delta(X, Y) = \mathrm{negl}(n)$), then they are computationally indistinguishable.

Proof sketch

Step 1: prove this for a **single-query** algorithm.

Let X, Y be two random variables on a set A.

For any (randomized) function $f : A \to B$, $\Delta(f(X), f(Y)) \leq \Delta(X, Y)$.

Proof sketch

Step 1: prove this for a **single-query** algorithm.

Let X, Y be two random variables on a set A.

For any (randomized) function $f : A \to B$, $\Delta(f(X), f(Y)) \leq \Delta(X, Y)$.

$\implies$ in particular the adversary $\mathcal{A}$ implements a function f into $\{0, 1\}$. Then we have:

$$\begin{aligned} \mathrm{Adv}(\mathcal{A}) &= |\Pr[f(X) = 1] - \Pr[f(Y) = 1]| \\ &= |\Pr[f(X) = 0] - \Pr[f(Y) = 0]| \\ \implies \mathrm{Adv}(\mathcal{A}) &= \Delta(f(X), f(Y)) \ . \end{aligned}$$

Proof sketch

Step 2: extend this to the multi-query case (in TD).

Recap

- The one-time pad has perfect security
- Perfect security implies large keys
- We use notions of **computational** security
- Indistinguishability: **statistical** implies **computational**
- Statistical indistinguishability is defined by the **statistical distance**
- Computational indistinguishability is defined by **games** (two equivalent definitions)