# Organization

- 6 lectures (André Schrottenloher)
- 7 TDs (Clémence Chevignard)
- 2 grades TBD

Course material (organization, lecture notes, slides, TDs...) on:
`https://andreschrottenloher.github.io/pages/teaching.html`

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

# Content of this course

- **Perfect security, rigorous definition of security** $\Longrightarrow$ this lecture
- Public-key cryptography: RSA
- Discrete logarithm problem, Diffie-Hellman key-exchange, ElGamal cryptosystem
- (LWE) (if time)
- Digital signatures
- Symmetric cryptography

# Introduction to Cryptography
# Part I: Introduction – Defining Security

André Schrottenloher

Inria Rennes
Team CAPSULE

1. History and Principles
2. Perfect Security
3. What is an Adversary?
4. Indistinguishability

# What is crypto today?

**Definition**

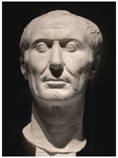
Protect **information** transmitted through **insecure channels** in presence of **adversaries** with the power to **listen** to and **corrupt** the transmitted messages.

This science borrows tools from:

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

# History and Principles

# 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.
- Cryptanalysis?

# Rotor machines

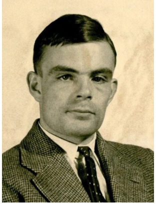
After the typewriter, encryption based on rotor machines (e.g., the Enigma family).

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

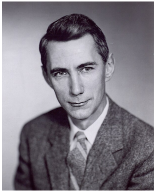
**Cryptanalysis of Enigma**

- 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 (Turing) & the US, allowing to break the 4-rotor version

# Dawn of modern cryptography (ca. 1950)



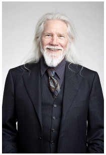
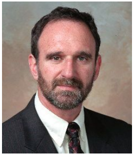
Turing: co-inventor of modern-day computer science, well-known for his code-breaking work during WWII



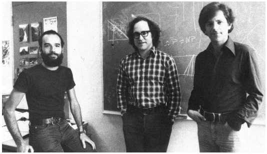
Shannon: information theory, information-theoretic security, and cipher design.

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

Crypto is now everywhere:

- Network protocols (HTTPS, SSL, TLS, PGP, wifi, mobile phone networks)
- Encrypted messaging apps
- Hardware: credit cards, DVD, Blu-ray
- Anti-piracy software

With applications beyond secure communication:

- Digital signatures
- Secure multi-party computation
- Electronic voting
- Proofs of knowledge

For (most of) these use cases there exists well-established, publicly audited, standardized designs (RSA, ECC, AES-GCM).

And even cryptocurrency.

# 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.

The security of a primitive relies on computational conjectures (different in symmetric / asymmetric crypto).

# 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

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

# Kerckhoffs' principles (1883)

**1.** Le système doit être **matériellement**, sinon mathématiquement, indéchiffrable

**2.** Il faut qu'il **n'exige pas le secret**, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi.

$\Longrightarrow$ a specification should be **public** (ex. ISO / IETF / NIST standard)

**3.** La clef doit pouvoir en être communiquée et retenue sans le secours de notes écrites, et être changée ou modifiée au gré des correspondants.

**4.** Il faut qu'il soit applicable à la correspondance télégraphique.

**5.** Il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes.

**6.** Enfin, il est nécessaire [...] que le système soit d'un usage facile [...].

# 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 $c \in \mathcal{C}$:

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

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

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

Proof in TD.

# The One-time Pad (Vernam's cipher)

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

It's correct:

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

**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 $K = \mathrm{KeyGen}()$

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

$K$ 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**:

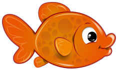
- Each key is chosen with probability $1/|\mathcal{K}|$
- For all $m \in \mathcal{M}, c \in \mathcal{C}$, there is a unique $k$ such that $\mathrm{Enc}(m, k) = c$.

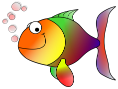
Proof in TD

# 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 / Turing machines
- The algorithms are **randomized**

# 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 $(t, \varepsilon)$-secure if any adversary running in time $t$ can attack it with probability at most $\varepsilon$.

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

"Attack" to be defined later!

# Indistinguishability

# Statistical indistinguishability

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.

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

$\implies$ this is a strong property. In practice, we need to relax it into the notion of **computational** indistinguishability.

# Computational indistinguishability

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

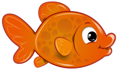
$\implies$ given access to **samples** of $D$, decide if $D = D_0$ or $D = D_1$.

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 $\mathcal{C}$.



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

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 $b'$

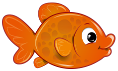
The **advantage** of $\mathcal{D}$ is:

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

$\mathcal{D}$ 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 $\varepsilon$.

# Computational indistinguishability

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

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

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

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

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

# Proof of the equivalence

Let $\mathcal{A}$ be a distinguisher for the **second** definition. Let $\mathcal{A}'$ that acts exactly like $\mathcal{A}$ in the game $G$.

$$\begin{aligned}
\mathrm{Adv}(\mathcal{A}') &= \Pr\left[b' = 1 | b = 0\right] - \Pr\left[b' = 1 | b = 1\right] \\
&= \left|1 - \Pr\left[b' = 0 | b = 0\right] - \Pr\left[b' = 1 | b = 1\right]\right| \\
&= \left|1 - 2\Pr\left[b' = 0 \wedge b = 0\right] - 2\Pr\left[b' = 1 \wedge b = 1\right]\right| \\
&= \left|1 - 2\Pr\left[Win\right]\right| \geq 2\varepsilon \ .
\end{aligned}$$

# Proof of the equivalence

Let $\mathcal{A}$ be a distinguisher for the **second** definition. Let $\mathcal{A}'$ that acts exactly like $\mathcal{A}$ in the game $G$.

$$\begin{aligned}
\mathrm{Adv}(\mathcal{A}') &= \Pr[b'=1|b=0] - \Pr[b'=1|b=1] \\
&= |1 - \Pr[b'=0|b=0] - \Pr[b'=1|b=1]| \\
&= |1 - 2\Pr[b'=0 \wedge b=0] - 2\Pr[b'=1 \wedge b=1]| \\
&= |1 - 2\Pr[Win]| \geq 2\varepsilon \ .
\end{aligned}$$

Let $\mathcal{A}'$ be a distinguisher for the **first** definition. Let $\mathcal{A}$ that acts exactly like $\mathcal{A}'$ in the games $G_0, G_1$.

$$\begin{aligned}
\Pr[Win] &= \Pr[b'=0 \wedge b=0] + \Pr[b'=1 \wedge b=1] \\
&= \frac{1}{2}(\Pr[b'=0|b=0] + \Pr[b'=1|b=1]) \\
&= \frac{1}{2}(1 + \mathrm{Adv}(\mathcal{A}')) \geq 1/2 + \varepsilon/2
\end{aligned}$$

We need to assume $\Pr[b'=1|b=1] \geq \Pr[b'=1|b=0]$: otherwise we modify $\mathcal{A}'$ to return $1 - b'$.

# Recap

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