Table of contents

Basic Cryptography Concepts

- Introduction to the **fundamental concepts** of cryptography.
- Presents **security principles**, **system types**, and **attack models**.
- Includes **historical systems** and complementary techniques.

Kerckhoffs' Principle

- Fundamental principle: **security relies solely on the key**, not on the secrecy of the algorithm.
- The system must remain **secure even if the algorithm is public**.
- The key must be **easily modifiable** and the system **simple to use**.

- Explicit rejection of **security through obscurity**.

Ultra-summary

- Security based on the key
- Public algorithm
- No security through obscurity

Original version

Kerckhoffs' Principle

Auguste Kerckhoffs published in **1883** two articles defining **six principles** for military ciphers:

1. The system must be **practically, if not mathematically, undecipherable**.
2. It must **not require confidentiality** and remain secure even if it falls into enemy hands.
3. The **key** must be able to be **memorized, transmitted and modified easily**, without written notes.
4. The system must be **compatible with telegraphic communications**.
5. It must be **portable** and usable by **a single person**.
6. It must be **simple to use**, without complex procedures or excessive constraints.

Kerckhoffs stated as early as the **19th century** that security must be **mathematically demonstrable** and that **there is no security through obscurity**.

Classification of Encryption Systems

Unconditional Security

(*unconditional security / perfect secrecy*)

- Security **independent of computing power**.
- **Ciphertext** provides no information about the **plaintext**.
- Conditions: **key   message**, **never reused**.
- Mostly **theoretical** use.
- **Example**: *one-time pad*.

As hard as / equivalent / provable security

- Cryptanalysis **as difficult** as a **hard mathematical problem**.
- **RSA** and **Rabin** proven equivalent to factorization.
  - Demonstrated by **reduction** (*reduction proof*).
- Central but **controversial** concept.

Computational Security

(*computational security / practical security*)

- Security based on **unrealistic cost of attacks**.
- Most used category in practice.
- **Examples**: **AES**, **DES**, **IDEA**, **RC4**.

Ultra-summary

- **Unconditional**: perfect, theoretical (*one-time pad*).
- **Provable security**: equivalence to hard mathematical problem.
- **Computational**: secure in practice.

Original version

- **Unconditional security** (*unconditional security also called **perfect secrecy***):
  - The security of the encryption system **is not compromised by the computing power** intended for cryptanalysis.
  - This category relies on **information theory** published by **Shannon in 1949**.
  - More precisely, an encryption system is **unconditionally secure** if the probability of encountering a **plaintext x** after observing the **corresponding ciphertext y** is identical to the a priori probability of encountering plaintext x.
  - In other words, having **plaintext/ciphertext pairs (x,y)** provides **no help for cryptanalysis**.
  - A necessary condition for a system to be unconditionally secure is that the **key be at least the same size as the message** and, above all, that it **not be reused** to encrypt different messages.
  - This condition makes these systems **poorly suited to usual cryptographic needs** and reduces their domain of interest to a **theoretical framework**.
  - The classic example is the **one-time pad** invented in **1917** by **J. Mauborgne** and **G. Vernam**.

> - Theoretical foundations of unconditionally secure systems + other examples in [**Sti06**].
>
> - **As hard as / equivalent / provable security**
>   - When it can be proven that cryptanalyzing the algorithm is **as difficult as solving a reputedly hard mathematical problem**.
>   - For example **factorization of large numbers**, calculation of **square roots modulo a composite**, calculation of **discrete logarithms in a finite group**, etc.
>   - The **Rabin** and **RSA** algorithms (generic case^1) are "proven" **equivalent to factorization**.
>   - Such a proof is called a "reduction" (**reduction proof**).
>   - The notion of **provable security** is at the origin of an **important controversy** in the cryptographic world.
> - **Computational security** (*computational security also called **practical security***)
>   - An encryption system is in this category if the **computational effort needed to "break" it** using the best possible techniques is **beyond** (with a reasonable margin) the computing resources of a hypothetical adversary.
>   - The vast majority of symmetric encryption systems (**AES, DES, IDEA, RC4**, etc.) are in this category.

Entropy

- **Entropy** (Shannon, 1948) measures the **effective amount of information** contained in a message.
- **Conditional entropy** measures the uncertainty that remains about the **plaintext** after observing the **ciphertext**.

Properties

- $0 \leq H(X) \leq \log n$
- $H(X) = 0 \rightarrow$ no uncertainty
- $H(X) = \log n \rightarrow$ all outcomes equally probable

Interpretation

- Approximates the **number of bits needed** to encode X.
- **Redundancy** = difference between effective encoding and entropy.

Conditional Entropy

- $H(X \mid Y=y) = -\sum_x P(X=x \mid Y=y) \log P(X=x \mid Y=y)$
- $H(X \mid Y) = \sum_y P(Y=y) H(X \mid Y=y)$
- Measures the remaining uncertainty about the **plaintext** after observing the **ciphertext**.

Ultra-summary

- **Entropy**: amount of information in a message.
- **Conditional entropy**: uncertainty about plaintext after ciphertext.
- **Redundancy**: difference between effective encoding and entropy.

Original version

- An essential definition in cryptography is the **effective** amount of information contained in a message.
- For example, days of the week (*Monday*, …, *Sunday*) can intuitively be encoded as character strings of length ($\leq$ len("Wednesday")), i.e., ($8 \times 8 = 64$) bits. However, the effective amount of information of the variable *day of the week* can be optimally encoded on **3 bits** (since ($2^3 = 8$) is sufficient to represent the 7 possible variations).
- **Entropy** (Shannon, 1948) is the mathematical formalization of this definition.

Formal Definition

Let X be a random variable with a finite set of possible values $x_1, x_2, \ldots, x_n$, such that $P(X = x_i) = p_i$, with $0 \leq p_i \leq 1$ and $\sum p_i = 1$. The entropy of X, denoted $H(X)$, is defined by
$H(X) = -\sum_{i=1}^{n} p_i \log p_i = \sum_{i=1}^{n} p_i \log\left(\frac{1}{p_i}\right)$
By convention: $p_i \log p_i = 0$ if $p_i = 0$. All logarithms are in **base 2**.

Interpretation

- Approximation of the number of bits needed to encode the elements of X.
- **Redundancy** is the difference between the effective encoding and entropy.

Properties

1. $0 \leq H(X) \leq \log n$
2. $H(X) = 0 \iff \exists i : p_i = 1, p_j = 0 \ \forall j \neq i$
3. $H(X) = \log n \iff p_i = 1/n \ \forall i$

Conditional Entropy

- $H(X \mid Y = y) = -\sum_x P(X = x \mid Y = y) \log P(X = x \mid Y = y)$,
- $H(X \mid Y) = \sum_y P(Y = y) H(X \mid Y = y)$

Measures the uncertainty about X (plaintext) after having observed Y (ciphertext).

Attacks on Encryption Systems

- **Ciphertext-only**: Adversary has only the **ciphertext**.
- **Known-plaintext**: Adversary has **plaintext/ciphertext pairs**.
- **Chosen-plaintext**: Adversary **chooses** the **plaintext** and see the ciphertext (and tries to find the plaintext for other messages).
- **Adaptive chosen-plaintext**: **depends on** the **received** ciphertexts.
- **Chosen-ciphertext**: Adversary **chooses** the **ciphertext** and obtains the plaintext (aims to find the key).
- **Adaptive Chosen-ciphertext**: **Chosen-ciphertext** depends on the received plaintexts.

Oracles and Security Models

Random Oracles and Security Models

- **Random Oracle**: A theoretical "perfect" function that returns a uniform and random value for each new input, but remains deterministic for an input already seen.
- **ROM (Random Oracle Model)**: Mathematical proof framework using this ideal oracle as a substitute for hash functions.
- **Standard Model**: Framework where security relies solely on the adversary's computing power against real algorithms.
- **Limit**: A security proof in ROM does not guarantee absolute security in the real world (with SHA-256, etc.).

Original version

A **random oracle** is an abstract entity accessible to legitimate parties and adversaries.

- **Behavior**: It responds to input queries x with perfectly random responses $Orc(x)$.
- **Determinism**: The only exception lies in previously processed inputs $(x_1, x_2, \dots, x_n)$. If $x_1' = x_1$, then $Orc(x_1') = Orc(x_1)$.
- **Modeling**: It is modeled by a function $Orc : X \to Y$ where $\forall x \in X, \Pr(Orc(x) = y) = \frac{1}{|Y|}$.
- **Utility**: It behaves like an **"ideal" cryptographic hash function**, a valuable

tool for proving security in the **Random Oracle Model**.
- **Comparison**: The **standard model** limits adversaries by computational factors. A protocol secure in the random oracle model can become vulnerable if used with a "real" hash function (SHA-1, SHA-256).

Encryption, Decryption and Signature Oracles

- **Function**: Entities that perform operations (encrypt/sign) for the adversary using secret keys without ever revealing them.
- **Symmetric cryptography**: The oracle provides $E_k(x)$ or $D_k(y)$.
- **Asymmetric cryptography**: The oracle is crucial for private operations (decryption/signature), as public operations are already freely accessible.

Original version: Operational Oracles

An **encryption/decryption/signature oracle** is an abstract entity offering an "on-demand" service.

- **Key access**: It uses the **same keys as the legitimate owners** (symmetric and asymmetric systems) without disclosing them.
- **Symmetric primitives**: For a primitive E and a key k, it returns $y = E_k(x)$ or the corresponding plaintext x.
- **Public key systems**: The oracle is only needed for operations with the **private key** ($priv_k$).
  - **Decryption**: returns x such that $E'_{pubk}(x) = y$.
  - **Signature**: For a system S, it returns $y = S_{privk}(x)$.
- **Attacks**: The attack models using **chosen plaintext** (CPA) and **chosen ciphertext** (CCA) rely on making these oracles available to the adversary.

Indistinguishability and Semantic Security (IND-CPA)

- **Property**: An adversary must not be able to distinguish the ciphertexts of two different plaintext messages.

- **IND-CPA (Indistinguishability under Chosen Plaintext Attack)**: If the adversary guesses the correct message only with a probability of $1/2 + \epsilon$, the system is considered secure.
- **Semantic Security**: Equivalent to IND-CPA, it ensures that no useful information leaks from the ciphertext.

Original version: Semantic Security

Ciphertext indistinguishability guarantees the inability to distinguish the ciphertexts of given plaintexts.

- **Experiment (IND-CPA Security Game)**:

  1. The adversary chooses two messages M_0 and M_1.
  2. The oracle chooses a random index $i \in \{0, 1\}$ and returns $c_i = E_k(M_i)$.
  3. The adversary can perform other calculations or oracle calls.

- **IND-CPA Definition**: The system is secure if the adversary's advantage is **negligible** ($Prob = 1/2 + \epsilon$ with ϵ small).
- **Note**: In public key, the encryption oracle is useless because the adversary already possesses the public key. IND-CPA provides **semantic security**.

Probabilistic Encryption and OAEP

- **Problem**: Deterministic encryption allows **dictionary attacks** (comparison of known ciphertexts).
- **Solution**: Add randomness to the message before encryption so that $E(M)$ is different each time.
- **OAEP (Optimal Asymmetric Encryption Padding)**: Standard used with RSA. It combines the message P with a random number R via hash functions h and XOR operations ($\oplus$).

Original version: Determinism vs. Probabilism

Deterministic behavior (same inputs = same outputs) creates vulnerabilities.

- **Example**: If Alice sends "Yes" or "No", the adversary can compute $C_{yes} = E_{pub}(``Yes")$ and compare. They can create a **codebook** (dictionary) to identify messages without breaking the key.
- **Probabilistic encryption**: Adds randomness. The goal is semantic security for the public key.
- **OAEP**: Used in **RSA-PKCS1**. The text P is combined with randomness R:

  - $M_1 := P \oplus h(R)$
  - $M_2 := R \oplus h(M_1)$
  - Encryption applies to M_1 and M_2. During decryption, we recover $R = M_2 \oplus h(M_1)$, then $P = h(R) \oplus M_1$.

Ultra-summary

- **Random Oracle**: “Ideal” hash function (theoretical model).
- **CPA/CCA Oracles**: Simulate access to the secret key to test resistance.
- **IND-CPA**: Inability to distinguish two ciphertexts (Semantic Security).
- **Probabilistic Encryption**: Essential to counter codebooks (dictionary attacks).
- **OAEP**: Padding method adding the necessary randomness to RSA.

History of Cryptography and Unconditional Security

Historical Encryption Systems

Cryptography was for a long time limited to the sole pursuit of **confidentiality**. Historical systems are based on two fundamental principles: **substitution** and **transposition**.

- **Caesar Cipher** (mono-alphabetic substitution): Fixed letter shift. Very vulnerable to **frequency analysis**.
- **Vigenère Cipher** (polyalphabetic substitution): Uses a key to vary the shift. More complex, but breakable by identifying the key length.
- **Transposition Cipher**: Reorganization of the original text characters according to a permutation defined by a key.

Original version: Historical Cryptography

For centuries **confidentiality** was the only application of cryptography…

- **1st century BC, Caesar Cipher**: **Mono-alphabetic substitution encryption** $e_k(x) = (x + k) \pmod{26}$, $d_k(y) = (y - k) \pmod{26}$ where $x, y, k \in \mathbb{Z}_{26}$.
  - Example: E_1('hello') = 'ifmmp'.
  - **Cryptanalysis**: easy, based on **character frequency**.
- **16th century, Vigenère**: **Polyalphabetic substitution encryption** $e_k(x_1, \dots, x_n) = (x_1 + k_1, \dots, x_m + k_m, x_{m+1} + k_1, \dots) \pmod{26}$.
  - **Cryptanalysis**: find the **key length** m by identifying repeated ciphertext portions and analyzing separate blocks as in the Caesar Cipher.

- **Transposition Ciphers** (Porta, 1563): The key defines a **permutation** on the plaintext.
- These techniques are still the basis of modern encryption systems (ex: **Enigma**, qualified by W. Churchill as the secret weapon that won the war).

The One-Time Pad (OTP)

The **One-Time Pad** (OTP), or **Vernam cipher**, is the only system proven to be **unconditionally secure** (perfect secrecy).

- **Principle**: The message is combined with a key of the same length via the XOR operation ($\oplus$).
- **Unconditional Security**: Observation of the encrypted message provides no information about the plaintext message. Even an adversary with infinite computing power cannot break it.
- **Shannon's Constraints**: The key must be **as long as the message**, purely **random**, and **used only once**.
- **Key Reuse**: If a key is reused for two messages, an attacker can eliminate the key by XOR ($y_a \oplus y_b = x_a \oplus x_b$) and recover the plaintext messages.

Original version: The One-Time Pad

Let $n \geq 1$ and the spaces P, C, K such that $P, C, K = (\mathbb{Z}_2)^n$. The encryption and decryption operations of a **one-time pad** (Vernam Cipher) are: $E_k(x_i) = x_i \oplus k_i$ and $D_k(y_i) = y_i \oplus k_i$ for $1 \leq i \leq n$.

- **Unconditional security**: If k_i are random and independent, observation of ciphertexts does not help cryptanalysis. The **entropy** of X does not decrease: $H(X|C) = H(X)$.
- **Shannon's Theorem**: Necessary condition: $H(K) \geq H(X)$. The length of the **random key** must be at least as large as that of the plaintext.
- **Key reuse**: $y_a \oplus y_b = x_a \oplus x_b$. With low-entropy messages, the plaintexts and the key ($k = y_a \oplus x_a$) can be recovered.
- Vulnerable to the **Known Plaintext** attack (if the key is reused).
- Major problem: The **distribution and management** of large keys. Revived by **quantum cryptography** proposing confidential channels for unlimited-length key distribution.

Steganography

In contrast to cryptography, which makes the message unreadable, **steganography** conceals the very existence of the message.

- **Method**: Use a “subliminal channel” (an innocent medium like an image or a banal text).
- **Modern technique**: Insertion of data into the **least significant bits** (LSB) of multimedia files, allowing the hiding of large volumes of data without visible alteration.

Original version: Steganography

Steganography hides a message inside another. Constituent elements:

1. A different **physical or logical channel** (subliminal channel).
2. A **secret mechanism** to identify this channel.

- **Classic examples**: First letters of words in a text, invisible ink.
- **Modern example**: Use the **least significant bits** of the frames of a Photo CD.
- For a 2048x3072 image (RGB 24 bits), hiding a message using 1 bit allows storing **2.3 Mb** without deteriorating quality.

Ultra-summary

- **History**: Substitution (Caesar/Vigenère) and Transposition (permutation).
- **One-Time Pad**: Absolute security if the key is random, unique, and as long as the message ($H(K) \geq H(X)$).
- **Steganography**: Hiding the existence of the message (ex: LSB technique in images).