Cryptography Basic Notions

- Kerckhoffs' principle
- Cryptosystem classification based on complexity
- Entropy
- Classification of cryptosystems attacks
- Random oracles and encryption oracles
- Deterministic and probabilistic encryption
- Historic cryptosystems
- The *One-Time* Pad
- Steganography

Kerckhoffs' principle

Auguste Kerckhoffs wrote in 1883[1] two journal articles formulating six design principles for military ciphers:

1. **The system must be practically, if not mathematically, indecipherable.**
2. **It should not require secrecy, and it should not be a problem if it falls into enemy hands.**
3. It must be possible to communicate and remember the key without using written notes, and correspondents must be able to change or modify it at will.
4. It must be applicable to telegraph communications.
5. It must be portable, and should not require several persons to handle or operate.
6. Lastly, given the circumstances in which it is to be used, the system must be easy to use and should not be stressful to use or require its users to know and comply with a long list of rules.

Kerckhoffs' states already in the XIX century that the system should be **mathematically proven** and that there is no **security through obscurity !**

1. Auguste Kerckhoffs, *"La cryptographie militaire" Journal des sciences militaires*, vol. IX, pp. 5–83, January 1883, pp. 161–191, February 1883.

Classification des systèmes de cryptage

- **Sécurité inconditionnelle** (*unconditional security* aussi appelée *perfect secrecy*):
  - La sécurité du système de cryptage n'est pas compromise par la puissance de calcul destinée à la cryptanalyse.
  - Cette catégorie s'appuie sur la théorie de l'information publiée par Shannon en 1949.
  - Plus précisément, un système de cryptage est *inconditionnellement sûr* si la probabilité de rencontrer un *plaintext* `x` après l'observation du *ciphertext* correspondant `y` est identique à la probabilité *à priori* de rencontrer le plaintext `x`. En d'autres termes, le fait de disposer de couples plaintext/ciphertext `(x,y)` ne constitue aucune aide pour la cryptanalyse.
  - Une condition nécessaire pour qu'un système soit inconditionnellement sûr est que la clé soit au moins de la même taille que le message et, surtout, qu'elle ne soit pas réutilisée pour encrypter des messages différents.
  - Cette condition rend ces systèmes peu adaptés aux besoins cryptographiques habituels et réduit leur domaine d'intérêt à un cadre théorique.
  - L'exemple classique est le *one-time pad* inventé en 1917 par J.Mauborgne and G. Vernam.
  - Fondements théoriques des systèmes inconditionnellement sûrs + d'autres exemples dans [Sti06].

Classification des systèmes de cryptage (II)

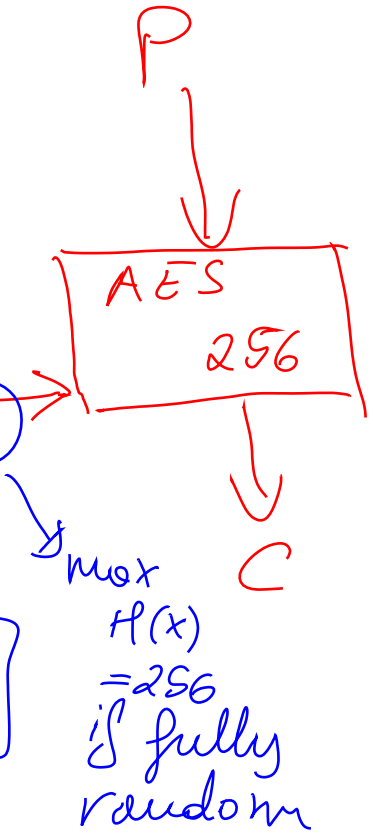
- ***As hard as* / équivalent / *provable security***
  - Lorsqu'on peut prouver que la cryptanalyse de l'algorithme est aussi difficile que de résoudre un problème mathématique réputé difficile.
  - Par exemple *la factorisation de grands nombres, le calcul de racines carrées modulo un "composite", le calcul de logarithmes discrets dans un groupe fini*, etc. (voir chapitre sur la Cryptographie Asymétrique).
  - L'algorithme de Rabin et RSA (cas générique[1]) sont "prouvés" équivalents à la factorisation. Une telle preuve s'appelle de "réduction" (*reduction proof*).
  - La notion de *provable security* est à l'origine d'une importante controverse dans le monde cryptographique[2].
- **Sécurité calculatoire** (*computational security* aussi appelé *practical security*)
  - Un système de cryptage est dans cette catégorie si l'effort calculatoire nécessaire à le "casser" en utilisant les meilleures techniques possibles est au delà (avec une marge raisonnable) des ressources de calcul d'un adversaire hypothétique.
  - La grande majorité de systèmes de cryptage symétriques (AES,DES, IDEA, RC4, etc.) sont dans cette catégorie.

1. *Breaking RSA Generically is Equivalent to Factoring*. D. Aggarwal and U.Maurer. Eurocrypt 2009.
2. *Another Look at Provable Security*. Neil Koblitz and Alfred Menezes. Cryptology ePrint Archive 2004.

Entropie

measure: quality of information

- Une définition essentielle dans la cryptographie est la quantité d'information "effective" contenue dans un message.
- Par exemple, les jours de la semaine ("lundi",..., "dimanche") peuvent intuitivement être encodés comme des chaînes de caractères de longueur ≤ len("mercredi"), soit 8 x 8 = 64 bits. Cependant, la quantité d'information effective de la variable "jour de la semaine" peut être encodée de manière optimale sur 3 bits (car $2^3 = 8$ est suffisant pour représenter les 7 variations possibles).
- L'**entropie** (Shannon, 1948) est la formalisation mathématique de cette définition:
- Soit X une variable aléatoire avec un ensemble fini de valeurs possibles $x_1, x_2,..., x_n$ avec probabilité $P(X=x_i) = p_i$, avec $0 \le p_i \le 1 \ \forall$ i, t.q. $1 \le i \le n$ et t.q. $\sum_{i=0}^{n} p_i = 1$. L'**entropie** de X, H(X) est définie comme: $H(x) = -\sum_{i=1}^{n} p_i \log p_i = \sum_{i=1}^{n} p_i \log(1/p_i)$ avec (par convention) $p_i \cdot \log p_i = p_i \cdot \log(1/pi) = 0$ si $p_i = 0$. Tous les logs à base 2.
- Cette formule permet d'approximer le nombre de bits qui sont nécessaires pour encoder les éléments dans X. La **redondance** est la différence entre le codage effectif d'une pièce d'information et son entropie. Le langage naturel (anglais) a une entropie de 1.3 par lettre et, donc, une redondance de 6.7 bits avec un codage à 8 bits !



H(x) ≥ 0

Entropie (II)

Propriétés:

(i) $0 \leq H(X) \leq \log n$

(ii) $H(X) = 0$ ssi $\exists$ i t.q. $p_i = 1$ et $p_j = 0$, $\forall$ $j \neq i$ (c.à.d. il n'y a pas d'incertitude sur le résultat)

(iii) $H(X) = \log n$ ssi $p_i = 1/n$ $\forall$ i $1 \leq i \leq n$ (c.à.d. tous les résultats sont équiprobables).

Exemple:

L'entropie de la variable "jours de la semaine" en admettant que toutes les valeurs sont équiprobables serait: $H(\text{jours de la semaine}) = \sum_{i=1}^{7} \frac{1}{7} \log 7 = \log 7 = 2{,}807$

Entropie conditionnelle de X étant donné Y = y:

$$H(X|Y=y) = -\sum_{x} P(X=x|Y=y) \cdot \log(P(X=x|Y=y))$$

Entropie conditionnelle de X étant donné Y:

$$H(X|Y) = \sum_{y} P(Y=y) \cdot H(X|Y=y)$$

L'entropie conditionnelle mesure le degré d'incertitude qui reste sur X (le *plaintext*) après avoir observé Y (le *ciphertext*).

Attaques sur les systèmes de cryptage

- Attaque *ciphertext-only*: L'adversaire (ou le cryptanalyste) essaye de trouver la clé ou le plaintext à partir de l'observation du ciphertext seul. Un système de cryptage vulnérable à cette attaque n'offre aucune sécurité.
- Attaque *known-plaintext*: L'adversaire a des couples plaintext/ciphertext à disposition. Cette attaque est à peine plus facile à mettre en place que le *ciphertext- only.*
- Attaque *chosen-plaintext*: L'adversaire peut choisir le plaintext et obtenir le ciphertext correspondant, le but étant de retrouver du plaintext à partir des ciphertext observés.
- Attaque *adaptive chosen-plaintext*: Il s'agit d'une attaque *chosen-plaintext* ou le choix sur les plaintexts peut dépendre des ciphertexts reçus lors des requêtes précédentes.
- Attaque *chosen-ciphertext*: L'adversaire choisit le ciphertext et obtient le plaintext correspondant. L'objectif de cette attaque étant normalement de trouver la clé.
- Attaque *adaptive chosen-ciphertext*: Il s'agit d'une attaque *chosen-ciphertext* ou le choix sur les ciphertexts peut dépendre des plaintexts reçus lors des requêtes précédentes.

Random Oracles

- A random oracle is an abstract entity (available to legitimate parties and adversaries) that responds to queries containing a given input `x` with perfectly random responses `Orc(x)`.
- The only exception to this behaviour resides in the previously processed inputs $(x_1, x_2, x_3, ..., x_n)$ where the random oracle will provide the same response than the previous query so that `if x1' = x1 then Orc(x1') = Orc(x1)` which means that the random oracle is *deterministic* with previously processed inputs.
- A random oracle can be modeled as a mathematical function `Orc: X` $\rightarrow$ `Y` where $\forall$ `x` $\in$ `X` we have that `Pr(Orc(x) = y) = 1/Sizeof(Y)` $\forall$ `y` $\in$ `Y.`
- A random oracle behaves like an "ideal" cryptographic hash function and as such is a valuable tool to prove security assertions under the so called *random oracle model*[1]. The previous point ensures that all the possible hash function outputs (*digests*) are *equiprobable*.
- The classical case where adversaries are bounded by computational factors is called the *standard model*.
- A cryptographic protocol that is proven secure in the random oracle model may result insecure when replacing the random oracle with a "real life" hash function as `SHA-1`, `SHA-256`, etc.

---

1. M. Bellare and P. Rogaway, *Random oracles are practical: A paradigm for designing efficient protocols*, In Proceedings of the 1st ACM Conference on Computer and Communications Security (1993), 62 -73.

Encryption, Decryption and Signing Oracles

- An encryption/decryption/signing oracle is also an abstract entity that will perform the corresponding operations as an "on demand" service to all legitimate and illegitimate parties.
- This entity will use (without disclosing them) the same keys as the legitimate key owners for both symmetric and asymmetric cryptosystems returning the authentic plaintext, ciphertexts or signed documents!
- Assuming E is a symmetric encryption primitive and k is the secret symmetric key, the encryption oracle will return $y = E_k(x)$ for any given input plaintext x and the decryption oracle will return x such as $E_k(x) = y$ for any given input ciphertext y.
- In Public Key Cryptosystems the oracle is only necessary for secret key operations (whether is signing or decrypting) since public key operations are openly available.
- As a result, assuming E' is a public key encryption system and $pubk$ and $privk$ the public and private keys, the decryption oracle will return x such as $E'_{pubk}(x) = y$ for any given input ciphertext y.
- Similarly, assuming S is a public key digital signature system and $pubk$ and $privk$ are respectively the validation and signing keys, the signing oracle will return y such as $S_{privk}(x) = y$ for any given input x (the data to be signed).
- *Chosen-plaintext* and *(adaptive) chosen-ciphertexts* attacks are normally modeled on the assumption that these oracles are available to the adversary.

Indistinguishable Ciphertexts and Semantic Security

- *Ciphertext Indistinguishability* is a property of cryptographic protocols that ensures that an adversary will be unable to distinguish between different ciphertexts for given plaintexts.
- Let's assume an adversary with polynomial time computing capability and unlimited access to an encryption oracle with secret key `k`, proceeds to the following steps:
  - He generates two plaintexts M_0 and M_1 and sends them to the encryption oracle.
  - The encryption oracle picks an index i at random $i \in \{0,1\}$ and sends the corresponding ciphertext $c_i = E_k(M_i)$ to the adversary.
  - The adversary is free to perform any additional computations including calls to the encryption oracle for any other M_j with $j \notin \{0,1\}$
- We say that a cryptosystem is *indistinguishable under chosen plaintext attacks* (***IND-CPA***) if such an adversary has a negligible advantage over random guessing to successfully picking the right index i (Prob = 1/2 + ε with ε small and bound).
- It should be noted that for public key cryptosystems the presence of an encryption oracle is unnecessary since encryptions involves public keys and consequently can be easily performed by the adversary.
- Cryptosystems featuring IND-CPA are also said to provide ***semantic security***[1].

1. S. Goldwasser, S. Micali. *Probabilistic encryption and how to play mental poker keeping secret all partial information*. Proc. 14th Symposium on Theory of Computing: 365–377. 1982.

Problems of Deterministic Encryption

- Cryptographic algorithms display a deterministic behaviour since encryption and decryption operations yield the same results over identical inputs.
- This may result in clear weaknesses in terms of *ciphertext indistinguishability* if an encryption oracle is available or if public key cryptosystems are used.
- As an example, let's assume Alice sends a simple message (i.e. 'yes' or 'no') to Bob encrypted with Bob's public key. If the adversary can guess the *semantics* of the message, he could easily compute the corresponding ciphertexts $C_{yes} = E_{Bobpubkey}(\text{'Yes'})$ and $C_{non} = E_{Bobpubkey}(\text{'Non'})$.
- If no extra random is added before encryption both ciphertexts are clearly distinguishable with Prob = 1 and, as a result, plaintexts would be disclosed without knowledge of Bob's private key.
- The adversary can even build a *codebook* of known plaintexts made of semantically sound messages and compare them with observed ciphertexts for eventual matches without cryptanalysing the algorithm!
- This problem also applies to symmetric cryptosystems provided that an encryption oracle is available and that no random Initialization Vector (IV) is used (more on this subject in the section describing block ciphers).

Probabilistic Encryption

- Consists in adding randomness to the cryptosystem by processing the plaintext *before* applying the encryption function. As a result, when dealing with multiple encryption instances the same plaintext will result in different ciphertexts.
- Building semantically sound codebooks becomes useless for adversaries. The final aim being to obtain *ciphertext indistinguishability* and *semantic security* for public-key cryptosystems.
- Initial probabilistic encryption approaches[1] had unpractical message expansion factors.
- The most commonly used solution is *Optimal Asymmetric Encryption Padding*[2] (OAEP) where the initial plaintext P is combined with a hashed random number R as follows:

  $M_1 := P \oplus h(R)$ and $M_2 := R \oplus h(M_1)$.

  M_1 and M_2 are then encrypted: $C_1 = E_{pubk}(M_1)$ and $C_2 = E_{pubk}(M_2)$ and sent.

  Upon decryption R is computed: $R = M_2 \oplus h(M_1)$ and then P: $P = h(R) \oplus M_1$.
- The security proofs provided in the initial OAEP publication have been questioned in recent papers[3]. OAEP is the basis of RSA-PKCS1 encryption standard.

---

1. S. Goldwasser, S. Micali. *Probabilistic encryption and how to play mental poker keeping secret all partial information*. Proc. 14th Symposium on Theory of Computing: 365–377. 1982.
2. M. Bellare, P. Rogaway. Optimal Asymmetric Encryption -- How to encrypt with RSA. Extended abstract in Advances in Cryptology - Eurocrypt '94 Proceedings, Lecture Notes in Computer Science Vol. 950, A. De Santis ed, Springer-Verlag, 1995.
3. D Brown, *Unprovable Security of RSA-OAEP in the Standard Model*, IACR eprint no 2006/223, http://eprint.iacr.org/2006/223. 2006

Systèmes de Cryptage Historiques

- Pendant des siècles la confidentialité a été la seule application de la cryptographie...
- I av. JC, *Caesar Cipher*: Cryptage à substitution mono-alphabétique

  $e_k(x) = (x + k) \bmod 26,\ d_k(y) = (y - k) \bmod 26$

  $x, y, k \in Z_{26}$

  - Exemple: E_1('bonjour') = 'cpokpws'
  - Cryptanalyse: facile basée sur la fréquence des caractères
- XVI siècle, *Vigenère*: Cryptage à substitution polyalphabétique

  $e_k(x_1,...x_n) = (x_1+k_1,...,x_m+k_m,x_{m+1}+k_1,...x_n+k_i) \bmod 26$

  $d_k(y_1,...y_n) = (y_1-k_1,...,y_m-k_m,y_{m+1}-k_1,...y_n-k_i) \bmod 26$

  $(x_1,...,x_n,y_1,...y_n,k_1,...,k_m) \in Z_{26}$

  - Cryptanalyse: trouver la taille m de la clé en identifiant les portions de ciphertext répétées et ensuite analyser les blocs séparés comme dans le *Caesar Cipher*
- *Transposition Ciphers* (*Porta*, 1563): La clé définit une *permutation* sur le *plaintext.*
- Autres exemples: cf. [Sti95] et [Men97].
- A noter que ces techniques sont toujours à la base des systèmes de cryptage actuels.
- *W. Churchill* à propos des Services d'Intelligence Britanniques capables de cryptanalyser les systèmes de cryptage allemands (*Enigma*): "*that secret weapon that won the war*"

Le *One-Time Pad*

Soit $n \geq 1$ et les espaces P,C,K resp. des plaintexts, ciphertexts et clés possibles tels que: P, C, $K = (Z_2)^n$. Soient $x = (x_1,x_2,...,x_n) \in X$, $y = (y_1,y_2,...,y_n) \in Y$ et $k =(k_1,k_2,...,k_n) \in K$. Alors, les operations d'encryption et decryption d'un *one-time pad* (aussi appellé *Vernam Cipher*) sont définies comme suit:

$$E_k(x_i) = x_i \oplus k_i \qquad 1 \leq i \leq n$$

$$D_k(y_i) = y_i \oplus k_i \qquad 1 \leq i \leq n$$

- Si les k_i sont choisis de manière indépendante et aléatoire, le one-time pad est *unconditionellement sûr* contre des attaques *ciphertext-only* ce qui veut dire que le fait d'observer des ciphertexts n'est d'aucune aide pour la cryptanalyse, ou dans d'autres termes, que l'entropie de X n'est pas diminuée après l'obervation des ciphertexts, soit: $H(X|C) = H(X)$. ($H(X)$ = fonction d'entropie, $H(X|C)$ = entropie conditionnelle)
  Ceci reste vrai même si l'adversaire a des ressources de calcul infinies!
- Problème: Shannon a prouvé qu'une condition nécessaire pour qu'un système de cryptage à clé symétrique soit inconditionnellement sûr est que $H(K) \geq H(X)$. En admettant que les composants d'une clé de m bits sont aléatoires et équiprobables, on a que $H(K) = m$ et donc que $m \geq H(X)$. Donc pour satisfaire l'hypothèse de Shannon indépendamment de l'entropie de X, il faut que la longueur de la clé aléatoire soit au moins aussi grande que celle du plaintext!

One-Time Pad (II)

- Une première conséquence de cette propriété est que la clé ne peut (même partiellement) être réutilisée. Voyons le résultat de la réutilisation de la même clé sur deux plaintexts différents:

$$E_k(x_a) = y_a = x_a \oplus k$$

$$E_k(x_b) = y_b = x_b \oplus k$$

$$y_a \oplus y_b = x_a \oplus k \oplus x_b \oplus k = x_a \oplus x_b \text{ !!!}$$

  ce qui avec des plaintexts de faible entropie permet de retrouver les deux plaintexts et même de calculer la clé k car:

$$k = y_a \oplus x_a$$

- Ceci signifie que le one-time pad est vulnerable à l'attaque *known plaintext*, même si ceci n'a pas d'importance si une nouvelle clé est regénérée pour chaque nouveau message.
- La contrainte sur la longueur de la clé pose un problème évident dans la distribution et la gestion des clés. De ce fait l'utilisation réelle du one-time pad est très peu sinificative.
- L'avènement de la *cryptographie quantique* proposant des canaux confidentiels de distribution de clés de longueur ilimitée a rélancé l'intérêt du one-time pad mais l'application de ces techniques aux réseaux classiques des télécommunications est pour le moment impossible.

Stéganographie

- Au lieu de rendre le message intelligible comme la cryptographie, la **stéganographie** (*steganography*) cache un message à l'intérieur d'un autre à l'aide de techniques diverses. Les deux éléments constituants de toute solution stéganographique sont les suivants:
  - Un canal physique ou logique différent de celui qui, de toute évidence, transporte l'information (aussi appelé *canal subliminal*).
  - Une indication ou un mécanisme secret permettant d'identifier le canal et d'accéder aux informations qu'il transporte.
- Exemples des solutions stéganographiques classiques:
  - Former un message secret en utilisant les premières lettres de tous les mots d'un texte.
  - Utiliser une encre invisible pour cacher les parties secrètes d'un texte
- [Way93][1] propose une solution stéganographique novatrice: former un message secret en utilisant les *least significant bit*s des *frames* présents sur un *Kodak Photo CD*. Pour une image de 2048x3072 avec une profondeur RGB de 24 bits, ceci donne 2.3 Mb !, en utilisant seulement 1 des 24 bits (ce qui ne détériore pas la qualité de l'image originale).

1.[Way93]: Wayner, P. *Should Encryption be Regulated?* BYTE Review, 1993.