Séance 7 - Travaux Dirigés
Modes, Hashage et Authentification

Yann Rotella

2026

Cette séance de TD est théorique et se fait seul.e.

Exercice 1. *Le padding.*
La notion de padding est extrêmement importante en cryptographie. Il est donc nécessaire de bien comprendre et réaliser cet exercice rapidement.

(1) Rappeler ce qu'est un bon padding

(2) Donner deux systèmes de padding, dont un doit faire intervenir la taille (en nombre de blocs) du message.

(3) *À la maison :* Implémenter le padding en question en Python ainsi que la fonction inverse de chacun de vos systèmes de padding.

Exercice 2. *Constructions et analyse des fonctions de hachage.*
Le but de cet exercice est de revenir sur les notions du cours et de vérifier que tout est bien compris.

(1) Dessiner les deux schémas possibles les plus simples qui permettent de construire une fonction de compression avec un chiffrement par bloc. Qu'est-ce qu'il ne faut surtout pas faire ?

(2) Qu'est-ce que l'attaquant connaît ?

(3) Dessiner les trois constructions vues en cours : Davies-Meyer, Matyas-Meyer-Oseas et Myaguchi Preneel

(4) On change un peu Davies-Meyer et on choisit la fonction de compression avec $h_i = E_{h_{i-1}}(m_{i-1}) \oplus h_{i-1}$. Expliquer pourquoi il ne faut pas faire ça.

(5) Reprenez la construction originale de Davies-Meyer. Montrez comment pour n'importe quel message m on peut trouver h tel que $E_m(h) \oplus h = h$. [1]

Exercice 3. *Attaque par insertion.*
On considère un chiffrement par bloc utilisant le mode opératoire `CTR`. Un attaquant parvient à intercepter un chiffré $c = (c_0, c_1, \dots)$, correspondant à un message $m = (m_0, m_1, \dots)$. L'attaquant connaît uniquement c, mais pas m, ni bien sûr la clé k ou la nonce.

On suppose que l'attaquant parvient à forcer la personne qui chiffre à re-chiffrer un message m' quasiment identique à m, mais avec un bloc de zéros insérés parmi les autres blocs. On suppose en outre que l'attaquant parvient à forcer ce deuxième chiffrement à être réalisé avec la même nonce. L'attaquant obtient donc un nouveau chiffré c'.

(1) Comment l'attaquant peut-il déterminer le bloc à partir duquel m et m' différent ?

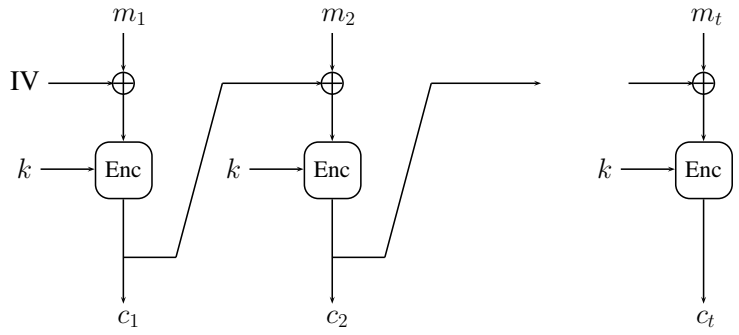
(2) Supposons que ce premier bloc différent ait pour indice i. Que vaut alors c'_i ? Comment l'attaquant peut-il en déduire m_i ?

(3) Montrer comment l'attaquant peut alors déduire les blocs suivants du message $m_{i+1}, m_{i+2}, \dots$.

(4) Que doit-on en conclure comme précaution sur l'utilisation de `CTR` ?

1. Cette propriété étrange ne casse pas complètement la sécurité de cette construction.

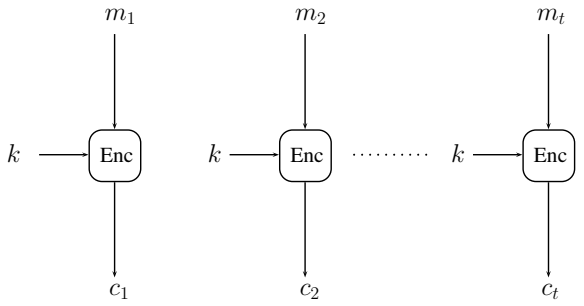
Exercice 4. *Cipher Block Chaining.*
Le mode de chiffrement CBC (Cipher Block Chaining) suit le schéma suivant



(1) Dessiner le schéma de déchiffrement correspondant à ce mode de chiffrement.
(2) À quoi sert le vecteur d'initialisation (IV) ? Doit-il rester secret ?
(3) On suppose que lors du chiffrement, ou pendant la transmission, un bloc a été altéré. Montrer que dans ce cas, lors du déchiffrement, seulement deux blocs seront déchiffrés incorrectement.
(4) **(Fuite d'information)** Qu'est-ce passe-t-il dans le cas où $c_i = c_j$ pour deux blocs chiffrés c_i et c_j distincts ?

Exercices complémentaires

Exercice 5. *Ce n'est même pas un mode - Electronic Code Book.*
Le mode de chiffrement ECB (Electronic Code Book) est le mode de chiffrement le plus simple : chaque bloc de données est chiffré indépendamment par la fonction de chiffrement, comme le montre la figure suivante :



(1) Quels sont les avantages principaux de ce mode ?
(2) Expliquer pourquoi ce mode opératoire n'est pas sûr.
(3) Jack, qui gagne 105 000 € par an a retrouvé l'entrée chiffrée qui lui correspond dans la base de données des salaires de son entreprise :

`Q92DFPVXC9IO`

Sachant que la fonction de chiffrement utilisée emploie des blocs de deux caractères et que le service informatique de son entreprise ne comprend aucun expert en cryptographie (entendre par là, utilise le mode ECB !), retrouver le salaire de Jane la patronne de Jack parmi le reste de la base de données :

`TOAV6RFPY5VXC9, YPFGFPDFDFIO, Q9AXFPC9IOIO, ACED4TFPVXIOIO, UTJSDGFPRTAVIO.`

Exercice 6. *Analyse de la construction en éponge.*
En 2007, Guido Bertoni, Joan Daemen, Michaël Peeters et Gilles Van Assche ont proposé la construction en éponge pour définir des fonctions de hachage à partir uniquement d'une permutation P publique.

On choisit une permutation P qui opère sur n bits. On fixe deux entiers r et c tels que $n = r + c$.

(1) Proposer un système de "padding" (bourrage) qui permet de transformer n'importe quel chaîne de bits dont la taille est un multiple de r, de telle sorte que ce padding soit injectif et qu'il soit inversible facilement.

À partir de maintenant, on suppose que chaque message m que l'on souhaite hacher est de taille un multiple de λr bits, pour $\lambda \in \mathbb{N}$.

Voici la procédure qui produit le haché de tout message m :

Algorithm 1 La fonction en éponge (simplifiée)

Input: Un message $m = m_0||m_1||\cdots||m_{\lambda-1}$ avec $|m_i| = r, \forall i$
Output: $h \in \{0,1\}^r$
  $S = 0^r||0^c$
  for $i = 0...\lambda - 1$ **do**
    $S = S \oplus (m_i||0^c)$
    $S = P(S)$
  end for
  return les r premiers bits de S

(2) Dessiner le schémas de la construction en éponge

(3) Montrer que si l'on applique pas la dernière permutation P (étape $S = P(S)$ pour la valeur $i = \lambda - 1$), alors il est très facile de construire une collision, de trouver une seconde pré-=image ou une pré-image, i.e. que la fonction de hachage ainsi définie n'est pas collision faible difficile, ni à collision forte difficile.

(4) On utilise maintenant correctement l'algorithme 1. Montrer qu'il existe une attaque en collision en approximativement $2^{r/2}$ applications de l'algorithme.

(5) Montrer qu'il existe une attaque en collision en approximativement $2^{c/2}$ applications de l'algorithme.

(6) On suppose maintenant qu'il est tout aussi coûteux de calculer P^{-1} que de calculer P. Montrer alors qu'il existe un algorithme qui trouve une pré-image pour un haché donné h en approximativement $2^{c/2}$ applications de P (et/ou) de P^{-1}.

(7) Pour que la fonction en éponge soit utilisable aujourd'hui, quelle(s) taille(s) propose-vous pour r, c et n ? Pensez-vous que choisir $P(\cdot) = \mathrm{AES}_k(\cdot)$ pour une certaine clef k est une bonne idée ?

(8) Proposez une version améliorée de la construction afin qu'aucune attaque en $2^{r/2}$ ne soit possible sans utiliser des propriétés spécifiques sur P.