

Cryptologie & Sécurité

Master 1 informatique

Université Claude Bernard Lyon 1

Fabien LAGUILLAUMIE

`fabien.laguillaumie@ens-lyon.fr`

`http://perso.ens-lyon.fr/fabien.laguillaumie`



Cryptologie & Sécurité

Master 1 informatique

Université Claude Bernard Lyon 1

Fabien LAGUILLAUMIE

`fabien.laguillaumie@ens-lyon.fr`

`http://perso.ens-lyon.fr/fabien.laguillaumie`



Chiffrement à flot

Le chiffrement de Vernam

Stream cipher et registres à décalage

Chiffrement par blocs

Chiffrements par blocs modernes

DES

AES

Modes opératoires

Stream ciphers from block ciphers

Stream ciphers from block ciphers

Message Authentication Codes - MAC

Fonctions de hachage



Chiffrement à flot

Le chiffrement de Vernam

Stream cipher et registres à décalage

Chiffrement par blocs

Chiffrements par blocs modernes

DES

AES

Modes opératoires

Stream ciphers from block ciphers

Stream ciphers from block ciphers

Message Authentication Codes - MAC

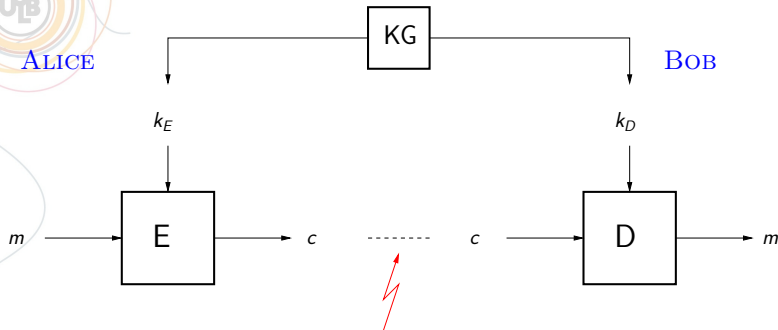
Fonctions de hachage

Confidentialité

Chiffrement symétrique

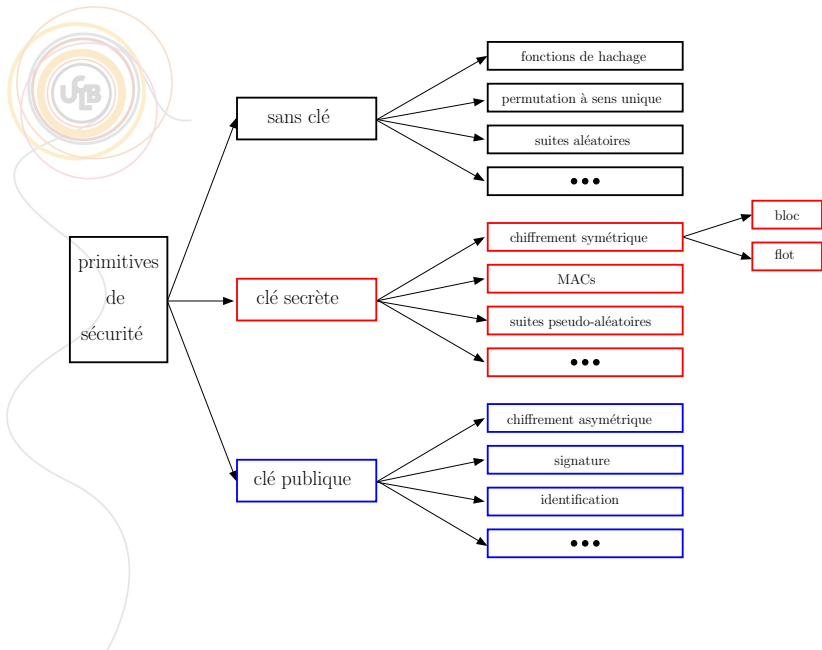
ALICE

BOB



$$k_E = k_D$$

- Encryption
- Decryption
- Key Generation





Chiffrement à flot

Chiffrement à flot

Chiffrement de Vernam

Cryptographie à clé secrète :

$$k_E = k_D$$

Un exemple fondamental : le *one-time pad*

Chiffrement de Vernam (1917) - masque jetable



Chiffrement à flot

Chiffrement de Vernam

Cryptographie à clé secrète :

$$k_E = k_D$$

Un exemple fondamental : le *one-time pad*

Chiffrement de Vernam (1917) - masque jetable

Le message : $m \in \{0,1\}^\ell$

$$m = m_1 m_2 \dots m_\ell$$

avec $m_i \in \{0,1\}$ pour $1 \leq i \leq \ell$.



Chiffrement à flot

Chiffrement de Vernam

Cryptographie à clé secrète :

$$k_E = k_D$$

Un exemple fondamental : le *one-time pad*

Chiffrement de Vernam (1917) - masque jetable

Le message : $m \in \{0, 1\}^\ell$

$$m = m_1 m_2 \dots m_\ell$$

avec $m_i \in \{0, 1\}$ pour $1 \leq i \leq \ell$.

La clé : $k \in_R \{0, 1\}^\ell$,

$$k = k_1 k_2 \dots k_\ell$$

avec $k_i \in \{0, 1\}$ pour $1 \leq i \leq \ell$.



Chiffrement à flot

Chiffrement de Vernam

Le chiffrement : $c \in \{0, 1\}^\ell$:

$$\begin{array}{rcccccc} m & = & m_1 & m_2 & \dots & m_\ell \\ \oplus & k & = & k_1 & k_2 & \dots & k_\ell \\ \hline c & = & c_1 & c_2 & \dots & c_\ell \end{array}$$

soit

$$c_i = m_i \oplus k_i \quad \forall 1 \leq i \leq \ell.$$

Chiffrement à flot

Chiffrement de Vernam

Le chiffrement : $c \in \{0, 1\}^\ell$:

$$\begin{array}{rcccccc} m & = & m_1 & m_2 & \dots & m_\ell \\ \oplus & k & = & k_1 & k_2 & \dots & k_\ell \\ \hline c & = & c_1 & c_2 & \dots & c_\ell \end{array}$$

soit

$$c_i = m_i \oplus k_i \quad \forall 1 \leq i \leq \ell.$$

Le déchiffrement :

$$\begin{array}{rcccccc} c & = & c_1 & c_2 & \dots & c_\ell \\ \oplus & k & = & k_1 & k_2 & \dots & k_\ell \\ \hline m & = & m_1 & m_2 & \dots & m_\ell \end{array}$$

soit

$$m_i = c_i \oplus k_i \quad \forall 1 \leq i \leq \ell.$$

Chiffrement à flot

Chiffrement de Vernam

En effet :

$$c \oplus k = m \oplus k \oplus k = m$$

Définition

Un chiffrement est dit parfait si l'on a

$$\Pr(M = m_0 \mid C = c_0) = \Pr(M = m_0).$$

Théorème

Si la clé k est tirée aléatoirement et uniformément parmi les chaînes binaires de longueur ℓ et n'est utilisée qu'une seule fois, le chiffrement de Vernam assure une confidentialité parfaite.

Chiffrement à flot

Chiffrement de Vernam

Preuve :

Idée : bruit blanc + message = bruit blanc

- ▶ distribution uniforme sur l'espace des clés $\Pr(K = k_0) = \frac{1}{2^\ell}$
- ▶
$$\Pr(C = c_0) = \sum_{m_0 \in \mathcal{M}} \Pr(M = m_0 \wedge K = m_0 \oplus c_0)$$

Chiffrement à flot

Chiffrement de Vernam

Preuve :

Idée : bruit blanc + message = bruit blanc

► distribution uniforme sur l'espace des clés $\Pr(K = k_0) = \frac{1}{2^\ell}$

►
$$\Pr(C = c_0) = \sum_{m_0 \in \mathcal{M}} \Pr(M = m_0 \wedge K = m_0 \oplus c_0)$$

$$= \frac{1}{2^\ell} \sum_{m_0 \in \mathcal{M}} \Pr(M = m_0)$$

Chiffrement à flot

Chiffrement de Vernam

Preuve :

Idée : bruit blanc + message = bruit blanc

► distribution uniforme sur l'espace des clés $\Pr(K = k_0) = \frac{1}{2^\ell}$


►
$$\Pr(C = c_0) = \sum_{m_0 \in \mathcal{M}} \Pr(M = m_0 \wedge K = m_0 \oplus c_0)$$

$$= \frac{1}{2^\ell} \sum_{m_0 \in \mathcal{M}} \Pr(M = m_0)$$

$$= \frac{1}{2^\ell}$$

Chiffrement à flot

Chiffrement de Vernam



$$\Pr(M = m_0 \mid C = c_0) = \frac{\Pr(M = m_0 \wedge C = c_0)}{\Pr(C = c_0)}$$

$$\text{Théorème de Bayes : } \Pr(A|B) = \frac{\Pr(A \wedge B)}{\Pr(B)}$$



Chiffrement à flot

Chiffrement de Vernam



$$\begin{aligned}\Pr(M = m_0 \mid C = c_0) &= \frac{\Pr(M = m_0 \wedge C = c_0)}{\Pr(C = c_0)} \\ &= \frac{\Pr(M = m_0 \wedge K = m_0 \oplus c_0)}{\Pr(C = c_0)}\end{aligned}$$

Théorème de Bayes : $\Pr(A|B) = \frac{\Pr(A \wedge B)}{\Pr(B)}$



Chiffrement à flot

Chiffrement de Vernam



$$\begin{aligned}\Pr(M = m_0 \mid C = c_0) &= \frac{\Pr(M = m_0 \wedge C = c_0)}{\Pr(C = c_0)} \\ &= \frac{\Pr(M = m_0 \wedge K = m_0 \oplus c_0)}{\Pr(C = c_0)} \\ &= \Pr(M = m_0) \frac{\Pr(K = m_0 \oplus c_0)}{\Pr(C = c_0)}\end{aligned}$$

Théorème de Bayes : $\Pr(A|B) = \frac{\Pr(A \wedge B)}{\Pr(B)}$



Chiffrement à flot

Chiffrement de Vernam


$$\begin{aligned}\Pr(M = m_0 \mid C = c_0) &= \frac{\Pr(M = m_0 \wedge C = c_0)}{\Pr(C = c_0)} \\&= \frac{\Pr(M = m_0 \wedge K = m_0 \oplus c_0)}{\Pr(C = c_0)} \\&= \Pr(M = m_0) \frac{\Pr(K = m_0 \oplus c_0)}{\Pr(C = c_0)} \\&= \Pr(M = m_0)\end{aligned}$$

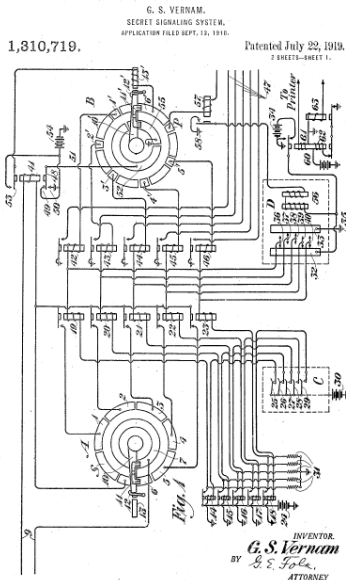


Théorème de Bayes : $\Pr(A|B) = \frac{\Pr(A \wedge B)}{\Pr(B)}$

Chiffrement à flot

Chiffrement de Vernam

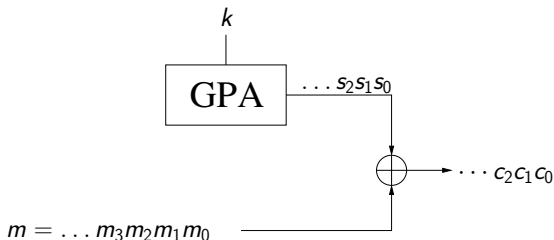
Le brevet :



Chiffrement à flot

Registres à décalage à réaction linéaire

Idée: faire du Vernam



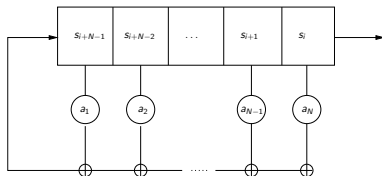
- ▶ k (la clé secrète) est petite (160 bits) et sert de graine à un **G**énérateur **P**seudo-**A**léatoire
- ▶ la suite produite (“suite chiffrente”) $(s_i)_{i \in \mathbb{N}}$ est additionnée bit-à-bit avec le message

Chiffrement à flot

Registres à décalage à rétroaction linéaire

Les ingrédients :

- ▶ des registres à décalage à rétroaction linéaire



$$P(X) = X^N + a_1 X^{N-1} + \dots + a_{N-1} X + a_N \in \mathbb{F}_2[X]$$

$$s_{i+N} = \sum_{j=1}^N a_j s_{i+N-j}$$

Registres à décalage à rétroaction linéaire

Exemples :

$$\begin{cases} P(X) = X^{10} + X^9 + X^7 + X^6 + X^3 + 1 \\ IV = 1001001001 \end{cases}$$

$$\begin{cases} P(X) = X^3 + 1 \\ IV = 001 \end{cases}$$

Polynôme primitif :

$$P(X) = x^{18} + x^{17} + x^{16} + x^{13} + 1$$

Registres à décalage à réaction linéaire

Definition

La *complexité linéaire* d'une suite $s = (s_i)_i \in \mathbb{N}$ est la longueur du plus petit LFSR permettant d'engendrer cette suite (noté $\Lambda(s)$).

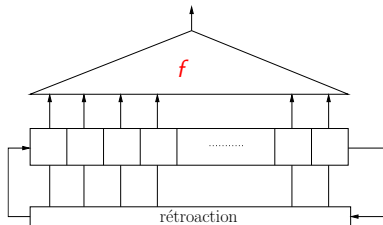
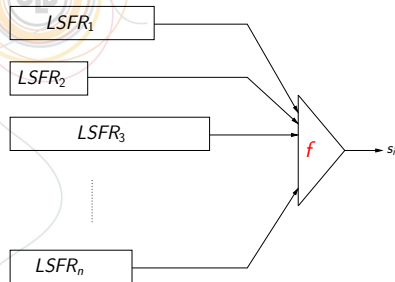
- ▶ Jim Massey montre en 1969 que l'algorithme de Berlekamp (décodage des BCH) permet de retrouver le polynôme caractéristique minimal d'une suite à partir de $\Lambda(s)$ bits.

\implies inutilisables directement en crypto.

d'où un autre ingrédient :

- ▶ des fonctions booléennes (avec bonnes propriétés)

Registres à décalage à réaction linéaire

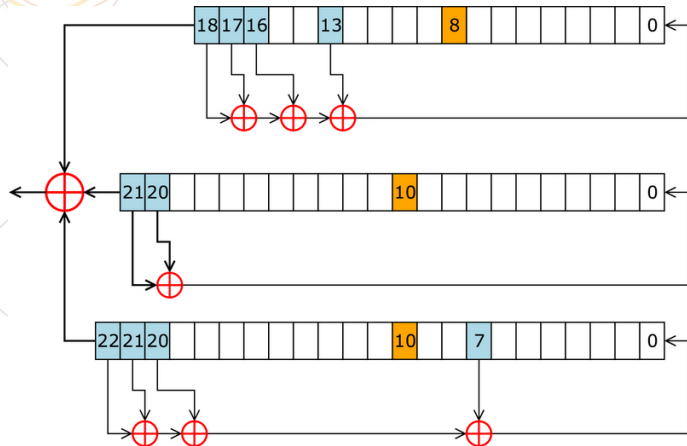


Remarques :

- ▶ très rapides et faciles à mettre en œuvre
- ▶ très répandus : A5/1 - GSM, E0 - Bluetooth...
- ▶ très durs à concevoir (eSTREAM – The ECRYPT Stream Cipher Project)

Registres à décalage à réaction linéaire

Suite chiffrante de A5/1 (GSM)



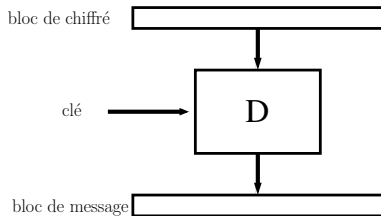
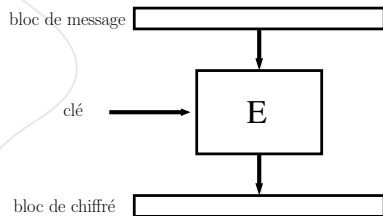


Chiffrement par blocs

Chiffrement par blocs


Definition

On désigne par **chiffrement par blocs** un chiffrement *symétrique* qui traite des *blocs* de longueur fixe (128 bits par exemple) de façon invariante.



Chiffrement par blocs

Definition

- 
- ▶ n = taille des blocs
 - ▶ ℓ = taille de la clé
 - ▶ recherche exhaustive à partir d'un couple clair/chiffré : $\mathcal{O}(2^\ell)$

Exemple : DES $n = 64$ et $\ell = 56 \rightsquigarrow 2^{56}$ déchiffrements

- ▶ AES, Blowfish, DES, Triple DES, FEAL, Serpent, Twofish...

Chiffrement par blocs

DES

Les chiffrements par blocs modernes :

Data Encryption Standard :

- ▶ 1972 – projet de recherche en crypto d'IBM
- ▶ 15 mai 1973 – appel d'offre du National Bureau of Standards US pour le DES
- ▶ 27 août 1974 – IBM répond avec un descendant de LUCIFER
- ▶ 15 janvier 1977 – négociations entre le NBS, la NSA et IBM : DES fût.

Chiffrement par blocs de 64 bits avec une clé de 64 bits (en fait 56 avec 8 bits de parité).

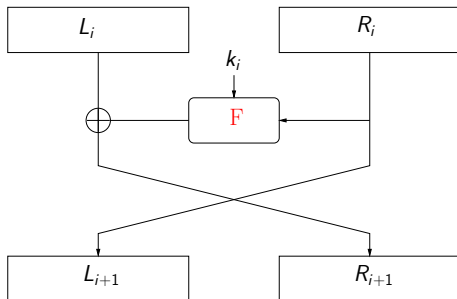
16 itérations d'un bloc de Feistel.

Chiffrement par blocs

DES

Le schéma de Feistel

fonction pseudo-aléatoire \rightsquigarrow permutation pseudo-aléatoire.

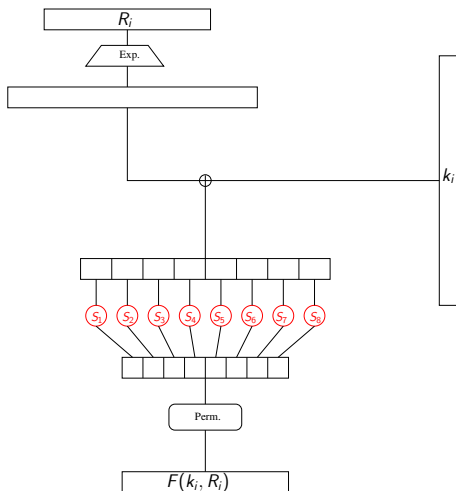


$$\begin{cases} L_{i+1} = R_i \\ R_{i+1} = F(k_i, R_i) \oplus L_i \end{cases} \iff \begin{cases} R_i = L_{i+1} \\ L_i = F(k_i, R_i) \oplus R_{i+1} \end{cases}$$

Chiffrement par blocs

DES

La fonction F et ses *boîtes* S (“substitution box”).



Chiffrement par blocs

DES

Les boîtes S contribuent à la confusion (cf. Shannon) et cassent la linéarité.

Entrée : $b_1 b_2 b_3 b_4 b_5 b_6 \rightsquigarrow b_1 b_6$ et $b_2 b_3 b_4 b_5$

$b_1 b_6 \backslash b_2 b_3 b_4 b_5$	0000	0001	...	1110	1111
00	0010	1100	...		
\vdots	\vdots				
11	1011				

S_5 :

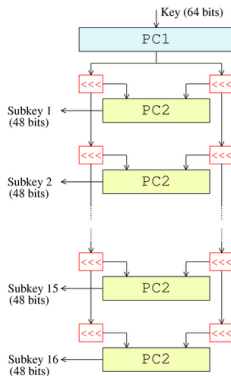
S_5		Middle 4 bits of Input															
		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Outer bits	00	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

Chiffrement par blocs

DES

Key schedule :

$PC_i =$ Permuted Choice i



Chiffrement par blocs

DES

Remarques :

- ▶ Pour une clé k fixée, DES définit une permutation de $\{0, 1\}^{64}$. Par ailleurs, l'ensemble des 2^{56} permutations n'est pas stable par composition
(chiffrement multiple \neq 1 chiffrement)
- ▶ 1ère attaque : en 1993 Matsui découvre la cryptanalyse linéaire (Henri Gilbert) et retrouve la clé avec 2^{43} couples clair/chiffré.
- ▶ 1998 DES Cracker - Electronic Frontier Fondation (250 000 \$) retrouve une clé en 3 jours
- ▶ 1999 Distributed Net 22h avec 10^5 PC + DES Cracker



Chiffrement par blocs

DES

Clé trop courte et chiffrement multiple :



- ▶ Recherche exhaustive naïve : 2^{2k}

Chiffrement par blocs

DES

Clé trop courte et chiffrement multiple :



- ▶ Recherche exhaustive naïve : 2^{2k}
- ▶ Problème : attaque Man-in-the-Middle
 - ▶ $c = E_{K_2}(E_{K_1}(m)) \implies D_{K_2}(c) = E_{K_1}(m)$
 - ▶ Calcul de $\alpha_j = E_j(m)$ pour les 2^k clés possibles + stockage (table de hachage)
 - ▶ Calcul de $\beta_j = D_j(c)$ pour les 2^k clés possibles + stockage (table de hachage)
 - ▶ $\beta_j = \alpha_i \rightsquigarrow (i, j)$

Chiffrement par blocs

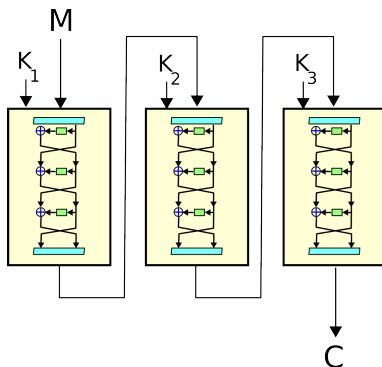
DES

► Complexité : $2 \times 2^k + 2^k$ en stockage

► 3DES :

$\text{DES}_{k_1}(\text{DES}_{k_2}(\text{DES}_{k_3}(m)))$ avec une clé de 168

$\text{DES}_{k_1}(\text{DES}_{k_2}^{-1}(\text{DES}_{k_1}(m)))$ avec une clé de 112 bits



Chiffrement par blocs



On oppose au *Feistel ciphers* les *product ciphers* :
combinaison de **substitutions** et de **transpositions**

- ▶ substitution \Rightarrow “confusion” (relation en entre clé et chiffré complexe)
- ▶ transposition \Rightarrow “diffusion” (propagation de la modification d'un bit)

Chiffrement par blocs

AES

Advanced Encryption Standard

- ▶ processus de normalisation lancé par le **National Institute of Standards and Technology** en 1997
- ▶ 15 candidats, 5 finalistes (3 US, 1 belge, 1 israelo-européen)
- ▶ the winner is... Rijndael (on prononce [rɛinda:l])
Joan Daemen & Vincent Rijmen (KUL - Belgique)

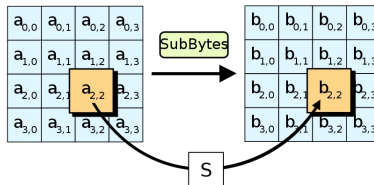
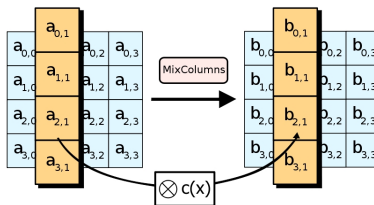
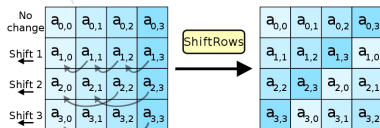
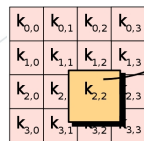
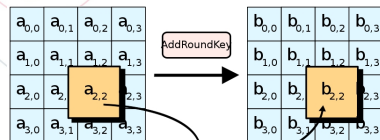


- ▶ cahiers des charges : 3 longueurs de clés (128, 192, 256 bits) et 3 longueurs de blocs
- ▶ Rijndael n'est **pas** un Feistel (*product cipher*)
- ▶ Succession de 10-12 ou 14 tours dans lesquels 4 opérations (dont 3 linéaires)

Chiffrement par blocs

AES

Les opérations



Modes opératoires

- ▶ Chiffrement par blocs de message de longueur quelconque.
 ↪ multiple de la taille des blocs

- ▶ On découpe le message en bloc de la longueur traitée

$$m = m_1 || m_2 || \dots || m_{k-1} || m_k$$

- ▶ Dernier bloc : *padding*

Modes opératoires

Exemples de padding

▶ RFC1321 (bit Padding)

- ▶ rajout d'un 1 suivi de 0 dans le dernier bloc
- ▶ rajout éventuel d'un bloc si le message a la bonne taille

Modes opératoires

Exemples de padding

▶ RFC1321 (bit Padding)

- ▶ rajout d'un 1 suivi de 0 dans le dernier bloc
- ▶ rajout éventuel d'un bloc si le message a la bonne taille

▶ ANSI X.923 (byte padding)

- ▶ rajout d'octets à 0
- ▶ dernier octet = nombre d'octets "padding"
- ▶ ...| DD DD DD DD DD DD DD DD | DD DD DD DD 00 00 00 04 |

Modes opérateurs

Exemples de padding

▶ RFC1321 (bit Padding)

- ▶ rajout d'un 1 suivi de 0 dans le dernier bloc
- ▶ rajout éventuel d'un bloc si le message a la bonne taille

▶ ANSI X.923 (byte padding)

- ▶ rajout d'octets à 0
- ▶ dernier octet = nombre d'octets "padding"
- ▶ ...| DD DD DD DD DD DD DD DD | DD DD DD DD 00 00 00 04 |

▶ ISO 10126 (byte padding)

- ▶ rajout d'octets aléatoires
- ▶ dernier octet : nombre d'octets "padding"
- ▶ ...| DD DD DD DD DD DD DD DD | DD DD DD DD 81 A6 23 04 |

Modes opérateurs

Exemples de padding

▶ RFC1321 (bit Padding)

- ▶ rajout d'un 1 suivi de 0 dans le dernier bloc
- ▶ rajout éventuel d'un bloc si le message a la bonne taille

▶ ANSI X.923 (byte padding)

- ▶ rajout d'octets à 0
- ▶ dernier octet = nombre d'octets "padding"
- ▶ ...| DD DD DD DD DD DD DD DD | DD DD DD DD 00 00 00 04 |

▶ ISO 10126 (byte padding)

- ▶ rajout d'octets aléatoires
- ▶ dernier octet : nombre d'octets "padding"
- ▶ ...| DD DD DD DD DD DD DD DD | DD DD DD DD 81 A6 23 04 |

▶ ISO 10126 (byte padding)

- ▶ rajout d'octets dont la valeur est le nombre d'octets rajouter :

- ▶

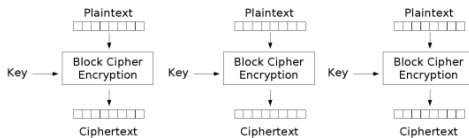
01			
02	02		
03	03	03	
04	04	04	04

Modes opérateurs



Modes opératoires

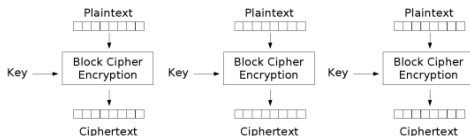
Electronic Code Book



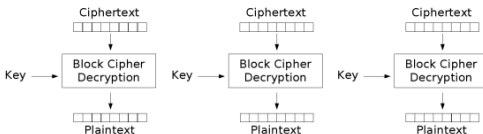
Electronic Codebook (ECB) mode encryption

Modes opératoires

Electronic Code Book



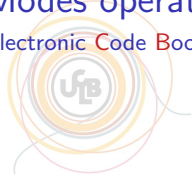
Electronic Codebook (ECB) mode encryption



Electronic Codebook (ECB) mode decryption

Modes opératoires

Electronic Code Book



c_i	$=$	$E_k(m_i)$
m_i	$=$	$D_k(m_i)$

- ▶ *Phantasy Star Online : Blue Burst* utilise **Blowfish** en mode ECB.
 - ▶ replay attacks pour gagner des points d'expérience
 - ▶ mécanisme d'échange de clé cassé aussi...

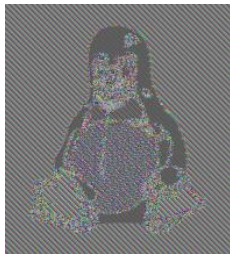
Modes opérateurs

Electronic Code Book



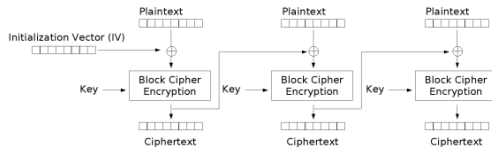
Modes opératoires

Electronic Code Book



Modes opératoires

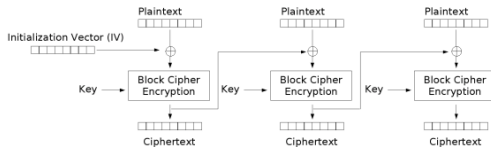
Cipher Block Chaining



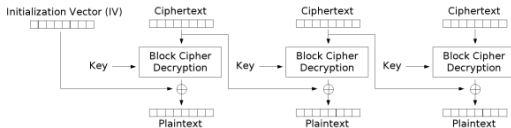
Cipher Block Chaining (CBC) mode encryption

Modes opératoires

Cipher Block Chaining



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

Modes opératoires

Electronic Code Book

- ▶ IV : vecteur d'initialisation
 - ▶ permet de rendre le chiffrement “aléatoire”
 - ▶ inutile de le garder secret
 - ▶ ne jamais le réutiliser avec la même clé

c_0	$=$	IV
c_i	$=$	$E_k(m_i \oplus c_{i-1})$
IV	$=$	c_0
m_i	$=$	$D_k(c_i) \oplus c_{i-1}$

- ▶ chiffrement séquentiel (non parallélisable)
- ▶ mode très utilisé

Modes opérateurs

Electronic Code Book



Modes opérateurs

Electronic Code Book



Chiffrement par blocs

Les algorithmes de chiffrement par blocs assurent la **confidentialité**.

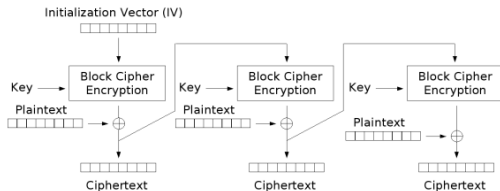
En tant que briques cryptographiques, ils permettent de créer :

- ▶ des générateurs pseudo-aléatoires
- ▶ des algorithmes de chiffrement à flot
- ▶ des MAC
- ▶ des fonctions de hachage,...

Chiffrement à flot

Stream ciphers from block ciphers

Cipher FeedBack

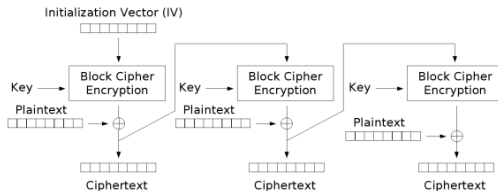


Cipher Feedback (CFB) mode encryption

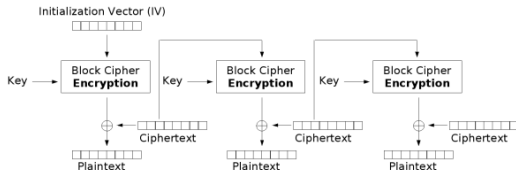
Chiffrement à flot

Stream ciphers from block ciphers

Cipher FeedBack



Cipher Feedback (CFB) mode encryption



Cipher Feedback (CFB) mode decryption

Chiffrement à flot

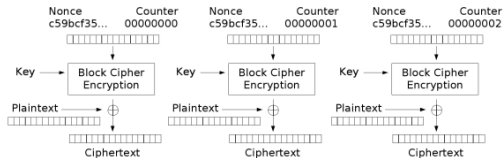
Stream ciphers from block ciphers

c_0	$=$	IV
c_i	$=$	$E_k(c_{i-1}) \oplus m_i$
IV	$=$	c_0
m_i	$=$	$D_k(c_{i-1}) \oplus c_i$

Chiffrement à flot

Stream ciphers from block ciphers

Counter

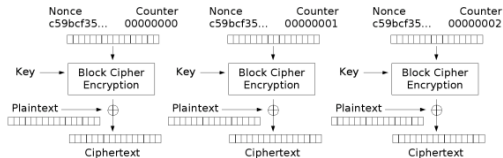


Counter (CTR) mode encryption

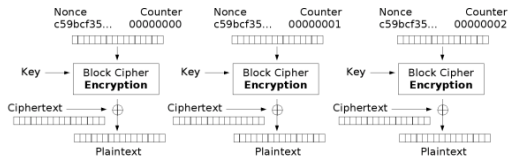
Chiffrement à flot

Stream ciphers from block ciphers

Counter



Counter (CTR) mode encryption



Counter (CTR) mode decryption



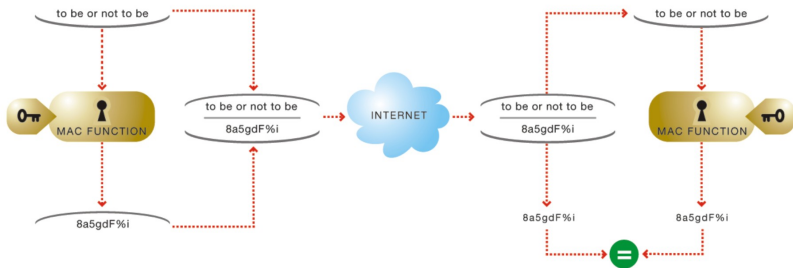
Message Authentication Codes MAC

MAC



- ▶ “petite” suite binaire permettant
 - ▶ d'*authentifier* un message
 - ▶ de garantir son *intégrité*
- ▶ calculé grâce à une clé
- ▶ contexte symétrique donc
 - ▶ ne convainc *que* la personne avec laquelle on partage la clé
- ▶ ce n'est *pas* une signature
 - ▶ pas de non-répudiation
 - ▶ pas de vérification universelle

MAC



MAC

Exemples :

- ▶ HMAC (keyed-Hash Message Authentication Code)

$$HMAC_k(m) = h(k \oplus opad || h(k \oplus ipad) || m)$$

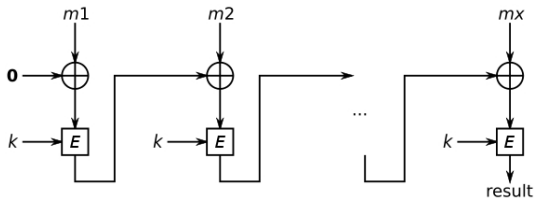
MAC

Exemples :

- ▶ HMAC (keyed-Hash Message Authentication Code)

$$HMAC_k(m) = h(k \oplus opad || h(k \oplus ipad) || m)$$

- ▶ CBC-MAC (Cipher Block Chaining Message Authentication Code)





Fonctions de hachage

Fonctions de hachage

Definition

Soit $\ell \in \mathbb{N}$.

Une fonction $h : \{0,1\}^* \longrightarrow \{0,1\}^\ell$ est une **fonction de hachage** si

- ▶ h est **à sens unique** (résistante à la préimage)
[étant donné $h(x)$ trouver x' tel que $h(x') = h(x)$ est difficile]
- ▶ h est **résistante à la seconde préimage**
[étant donné x et $h(x)$, trouver $x' \neq x$ tel que $h(x') = h(x)$ est difficile]
- ▶ h est **résistante aux collisions**
[trouver x et x' , $x \neq x'$, tels que $h(x) = h(x')$ est difficile]

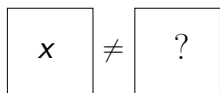
h crée une “empreinte” (un haché) de m .

Fonctions de hachage



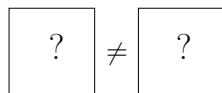
$h(x)$

Complexité : 2^ℓ



$h(x) = h(x')$

2^ℓ



$h(x) = h(x')$

$2^{\ell/2}$

Fonctions de hachage

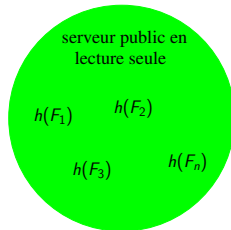
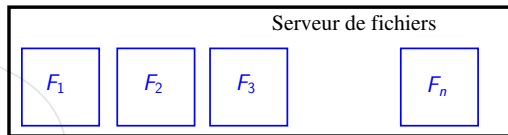
Résistance aux collisions :

$$h : \{0, 1\}^* \longrightarrow \{0, 1\}^\ell$$

Une collision est une paire (m_0, m_1) telle que

$$h(m_0) = h(m_1) \text{ et } m_0 \neq m_1$$

Pourquoi ?



Fonctions de hachage

Algorithme de recherche de collisions

1. Tirer aléatoirement $2^{\ell/2}$ messages $m_1, m_2, \dots, m_{2^{\ell/2}}$
2. Pour $i = 1$ à $2^{\ell/2}$ calculer $t_i = h(m_i)$
3. Chercher une collision $t_i = t_j$, sinon GOTO 1.

Fonctions de hachage

Algorithme de recherche de collisions

1. Tirer aléatoirement $2^{\ell/2}$ messages $m_1, m_2, \dots, m_{2^{\ell/2}}$
2. Pour $i = 1$ à $2^{\ell/2}$ calculer $t_i = h(m_i)$
3. Chercher une collision $t_i = t_j$, sinon GOTO 1.

PARADOXE DES ANNIVERSAIRES :

Soient $r_1, r_2, \dots, r_n \in \llbracket 1, B \rrbracket$ tirés uniformément et indépendamment :

si $n = 1.2 \times B^{1/2}$, alors $\Pr[\exists i \neq j : r_i = r_j] \geq 1/2$.

Fonctions de hachage

Algorithme de recherche de collisions

1. Tirer aléatoirement $2^{\ell/2}$ messages $m_1, m_2, \dots, m_{2^{\ell/2}}$
2. Pour $i = 1$ à $2^{\ell/2}$ calculer $t_i = h(m_i)$
3. Chercher une collision $t_i = t_j$, sinon GOTO 1.

- ▶ Nombres d'itérations avant de trouver : ~ 2
- ▶ Complexité en temps : $O(2^{\ell/2})$
- ▶ Complexité en espace : $O(2^{\ell/2})$

Fonctions de hachage

Des exemples :

- ▶ MD4 (128) MD5 (128)
- ▶ SHA-0 (160) SHA-1 (160)
- ▶ RIPEMD
- ▶ SHA-256
- ▶ Whirlpool
- ▶ AES

Fonctions de hachage

Des exemples :

- ▶ MD4 (128) ☠ MD5 (128) ☠
- ▶ SHA-0 (160) ☠ SHA-1 (160) théoriquement ☠
- ▶ RIPEMD ☠
- ▶ SHA-256 Sécurité précaire
- ▶ Whirlpool
- ▶ AES

Fonctions de hachage

Des exemples :

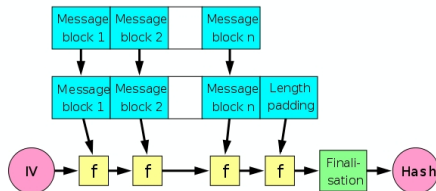
- ▶ MD4 (128) 💀 MD5 (128) 💀
- ▶ SHA-0 (160) 💀 SHA-1 (160) théoriquement 💀
- ▶ RIPEMD 💀
- ▶ SHA-256 Sécurité précaire
- ▶ Whirlpool
- ▶ AES

Concours SHA-3 :

- ▶ NIST 2008 : 64 soumissions dont 51 valides
- ▶ juillet 2009 : 14 candidats
- ▶ déc. 2010 : 5 finalistes (BLAKE, GRØSTL, JH, KECCAK, SKEIN)
- ▶ 2 oct. 2012 : vainqueur **KECCAK**
<http://keccak.noekeon.org/>

Fonctions de hachage

Construction de Merkle-Damgard : $m = m_1 m_2 \cdots m_n$



- ▶ $f : T \times M \longrightarrow T$ est une fonction de compression
- ▶ Théorème : si f est résistante aux collisions, h l'est aussi

Fonctions de hachage



PREUVE

Fonctions de hachage



PREUVE

Par contradiction : collision sur $h \implies$ collision sur f . □

Pour construire de bonnes fonctions de hachage, il faut construire de bonnes fonctions de compression.

Fonctions de hachage

Fonction de hachage et block cipher

Davies-Meyer & Matyas-Meyer-Oseas

