

(Corrigé du TD)

Partie-1 : Cryptologie classique

Exercice-1 : Attaques sur le cryptosystème de Cesar

On suppose l'association suivante :

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
q	r	s	t	u	v	w	x	y	z						
16	17	18	19	20	21	22	23	24	25						

Scénario KPA:

Ciphertext = gozih

on connaît la paire (plaintext,ciphertext)= (P,D)

Quel est la clé ?

Scénario CPA:

Tu peux choisir le plaintext que tu veux et obtenir le ciphertext correspondant.

Donner un exemple.

Quel est ton choix ? Et quel est donc la clé

Scénario CCA:

Tu peux choisir le ciphertext que tu veux et obtenir le plaintext correspondant.

Donner un exemple.

Quel est ton choix ? Et quel est donc la clé ?

KPA : $k = (D - P) \bmod 26 = 3 - 15 \bmod 26 = -12 \bmod 26 = 14$

CPA : on choisit le plaintext « A » comme entrée à la machine de cryptage, on obtient comme ciphertext « D » -> donc la clé est $(D-A) \bmod 26 = D=3$

CCA : on choisit comme ciphertext « A », comme entrée à la machine de décryptage, on obtient S. donc $k = (A-S) \bmod 26 = -S \bmod 26 = 8$

Exercice-2: Playfair

The Playfair Cipher

A	Z	I	WX	D
E	U	T	G	Y
O	N	K	Q	M
H	F	J	L	S
V	R	P	B	C

Utiliser la lettre spéciale Z pour séparer deux lettres égales. Et ajouter Z s'il reste une lettre individuelle à la fin.

On vous demande de Déchiffrer le ciphertext :

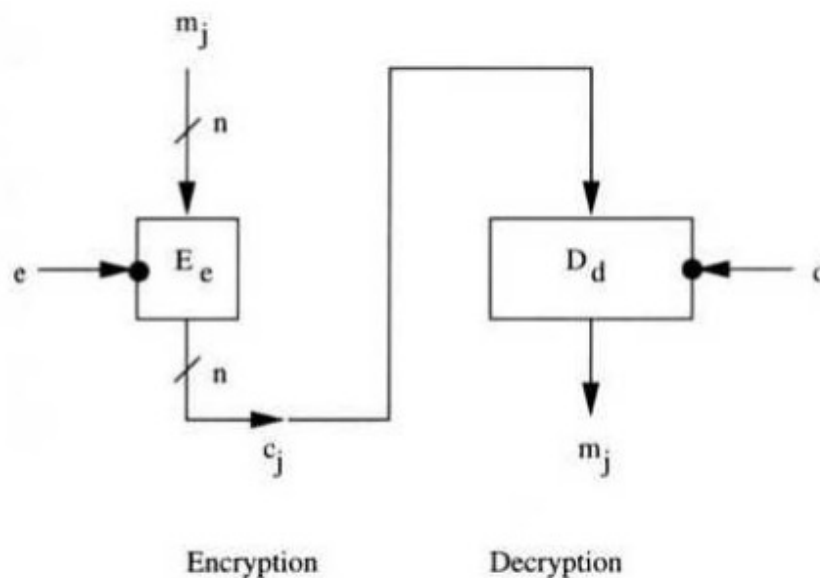
EJ DJ DJ EJ GA VO IE JY NK YV TI VO ZU

TH IS IS TH E W HE AT ST ONE C IP HE R

Partie-2 : Cryptologie moderne

Exercice-1 : ECB

On considère un cryptosystème de bloc qui applique une permutation à des vecteurs binaires de taille 4 en mode ECB.



La fonction de permutation π est définie comme suit :

$$b_1 b_2 b_3 b_4 \rightarrow b_{\pi(1)} b_{\pi(2)} b_{\pi(3)} b_{\pi(4)} \rightarrow b_2 b_3 b_4 b_1$$

On donne l'opération de permutation :

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{pmatrix} \rightarrow \pi(1)=2; \pi(2)=3; \pi(3)=4; \pi(4)=1$$

Soit le plaintext m :

$$\mathbf{m} = 101100010100101$$

- 1- Décrire mathématiquement ou avec un pseudo-code le fonctionnement de ECB

$$\mathbf{1) \quad c_i = E(m_i)}$$

- 2- Décomposer le plaintext en bloc de taille approprié. Faire du bourrage avec des zéros pour avoir des bloc de même taille

	B1	B2	B3	B4
M2	0	0	0	1
C2	1	0	0	0
M3	0	1	0	0
C3	0	0	1	0
M4	1	0	1	0
C4	0	1	0	1
M1	1	0	1	1

C1	1	1	0	1
----	---	---	---	---

- 3- Appliquer le mode ECB lors du chiffrement des blocs du plaintext

	B1	B2	B3	B4
M2	0	0	0	1
C2	1	0	0	0
M3	0	1	0	0
C3	0	0	1	0
M4	1	0	1	0
C4	0	1	0	1
M1	1	0	1	1
C1	1	1	0	1

- 4- Donner le ciphertext final

4) D'où le ciphertext final est : C= c1c2c3c4=

1	1	0	1	1	0	0	0	0	0	1	0	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

- 5- Appliquer le déchiffrement et vérifier avec le message original

5) D(0111 0010 1000 0101) = 1011 0001 0100 1010.

- 6- Considérer un plaintext formé par les mêmes blocs 1010, cette redondance est-elle propagée dans le ciphertext ?

6) Oui la redondance sera propagée ds le ciphertext : E(1010 1010 1010 1010) = 0101 0101 0101 0101

- 7- Si l'ordre des blocs des ciphertexts est modifié ? le décryptage de chaque bloc est-il possible ?

7) l'ordre des blocs dans ECB n'affecte pas le chiffrement/déchiffrement des blocs

- 8- Que pensez-vous de la sécurité d'ECB et dans quel application est-il approprié ?

8) ECB n'est pas si sécurisé. Il est adéquat seulement pour le chiffrement des messages très courts comme les IV ou les clés dans d'autres modes.

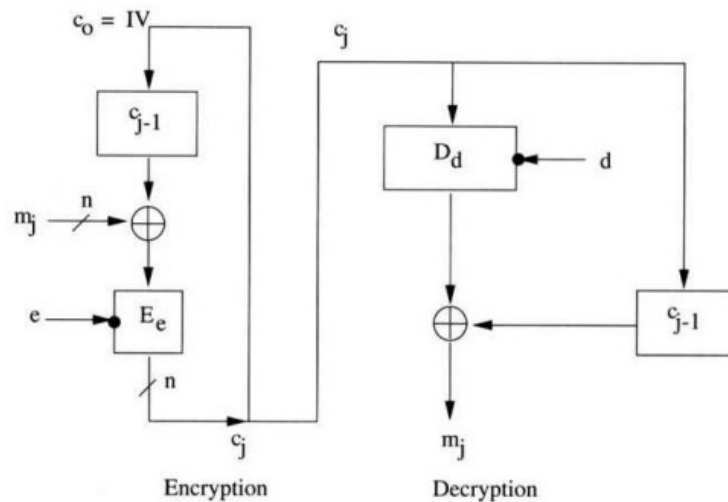
- 9- Le mode ECB est adéquat à la transmission des messages courts.**

10- **Openssl>enc -e -aes-128-ecb -in message.txt -out message.enc**

11- **Openssl>enc -d -aes-128-ecb -in message.enc -out message.dec**

Exercice-2 : CBC

On utilise la même clé, le même plaintext, la même opération mais en mode CBC. On donne IV=1010



1- Décrire mathématiquement ou avec un pseudo-code le fonctionnement de CBC.

1) Cryptage :

$C_0 = IV$; $c_j = E(c_{j-1} \oplus m_j)$ pour $1 \leq j \leq t$

Décryptage :

$C_0 = IV$; $m_j = c_{j-1} \oplus D(c_j)$ pour $1 \leq j \leq t$

2- Décomposer le plaintext en bloc de taille approprié. Faire du bourrage avec des zéros pour avoir des blocs de même taille

2) bourrage avec un seul 0 pour avoir des blocs de taille $n=4$. $m = 1011\ 0001\ 0100\ 1010$

$m_1 = 1011$; $m_2 = 0001$; $m_3 = 0100$; $m_4 = 1010$

3- Appliquer le mode ECB lors du chiffrement des blocs du plaintext

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \end{pmatrix}$$

Cryptage				
	B1	B2	B3	B4
M1	1	0	1	1
iv	1	0	1	0
M1 xor IV	0	0	0	1
C1	1	0	0	0
M2	0	0	0	1
M2 xor C1	1	0	0	1
C2	1	1	0	0
M3	0	1	0	0
M3 xor C2	1	0	0	0
C3	0	1	0	0
M4	1	0	1	0
M4 xor C3	1	1	1	0
C4	0	1	1	1

Décryptage				
	B1	B2	B3	B4
C1	1	0	0	0
D(c1)	0	0	0	1

iv	1	0	1	0
M1= D(c1) xor iv	1	0	1	1
C2	1	1	0	0
D(c2)	1	0	0	1
M2= D(c2) xor C1	0	0	0	1
C3	0	1	0	0
D(c3)				
C4	0	1	1	1
D(c4)				

C=c1c2c3c4=
1 0 0 0 1 1 0 0 0 1 0 0 0 1 1 1

- 4- Considérer un plaintext formé par les mêmes blocs 1011, cette redondance est-elle propagée dans le ciphertext ?

On suppose le message suivant : 1011101110111011101110111011

6) La redondance ne sera pas propagée :

E(1011 1011 1011 1011) = 0010 0011 0001 0101

- 7- Si l'ordre des blocs des ciphertexts est modifié ? Le décryptage de chaque bloc est-il possible ?

7) si l'ordre de blocs de ciphertext change ou les blocs de ciphertext sont remplacés par d'autres, alors le décryptage devient impossible. Ceci est un avantage de CBC par rapport ECB

- 8- Que pensez-vous de la sécurité de CBC et dans quelle application est-il approprié ?

8) sécurité améliorée parce que plus de confusion !

- 12- Si une erreur se passe dans le premier bloc du ciphertext. Etudier la propagation d'erreur sur le décryptage (dire quels sont les blocs affectés et les blocs intacts du plaintext).

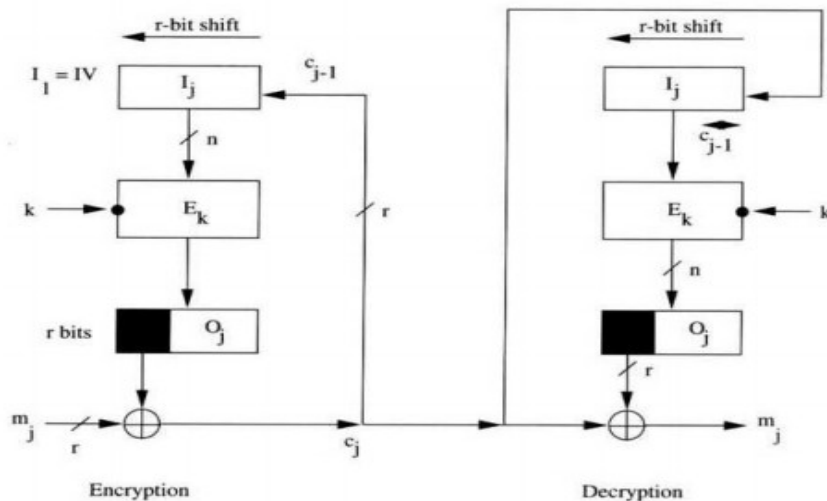
Il y a une propagation de l'erreur. Il faut réinitialiser le processus de décryptage par la ré-implication de IV.

- 10- Dire quelle application CBC est approprié

10) CBC peut être approprié pour le chiffrement de long messages tels que les messages multimédias

Exercice-3 : CFB

Dans CFB, on a besoin d'un IV et aussi un entier r avec $1 \leq r \leq n$. le plaintext sera décomposé en blocs de r . Et Initialisation d' $I_1 = IV$



Refaire l'exercice avec la même opération de permutation E, même plaintext et même **IV**, on donne aussi **r = 3**.

1)

Pour $1 \leq j \leq u$, alice fait le suivant

- $O_j = E(I_j)$;
- Extraire t_j qui est les r premiers bits de O_j ;
- $c_j = m_j \oplus t_j$
- $I_{j+1} = (2^r I_j + c_j) \bmod 2^n$. donc I_{j+1} est généré en supprimant les premiers r bits de I_j et en ajoutant c_j .
- Le ciphertext sera donc $c = c_1 c_2 c_3 \dots c_u$

Le décryptage est similaire :

Pour $1 \leq j \leq u$, bob fait le suivant

- $O_j = E(I_j)$;
- Extraire t_j qui est les r premiers bits de O_j ;
- $m_j = c_j \oplus t_j$
- $I_{j+1} = (2^r I_j + c_j) \bmod 2^n$. donc I_{j+1} est généré en supprimant les premiers r bits de I_j et en ajoutant c_j .
- Le plaintext sera donc $m = m_1 m_2 m_3 \dots m_u$

2) $m_1 = 101$; $m_2 = 100$; $m_3 = 010$; $m_4 = 100$; $m_5 = 101$

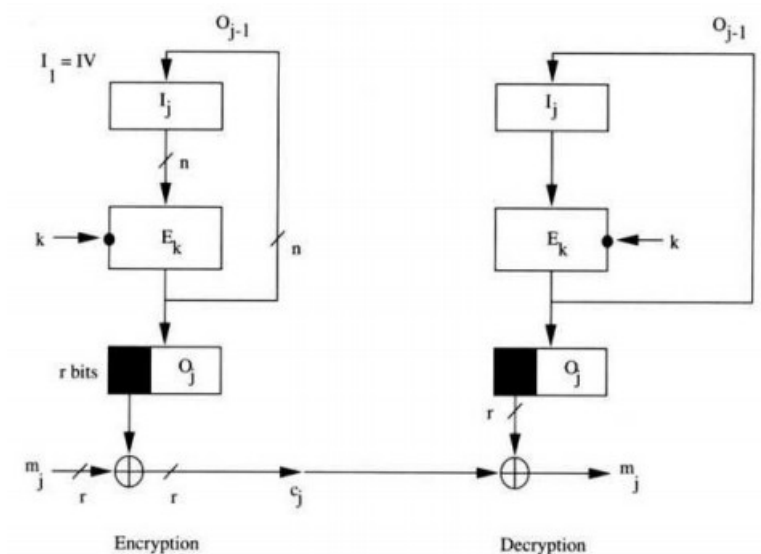
j	I_j	O_j	t_j	m_j	c_j
1	1010	0101	010	101	111
2	0111	1110	111	100	011
3	1011	0111	011	010	001
4	1001	0011	001	100	101
5	1101	1011	101	101	000

9) il ya propagation d'erreur aussi dans CFB puisque la partie erroné du ciphertext se trouve dans I_j . Il faut noter aussi que CFB ne peut être fonctionnel dans les cryptosystèmes à clé publique comme RSA puisque le récepteur utilise la clé publique aussi dans l'opération E comme l'émetteur.

10) Alice et Bob peuvent calculer t_{j+1} dès qu'ils connaissent le bloc c_j . Pour cela le bloc t_1 peut être calculé par Alice et Bob simultanément. Alice genere le ciphertext $c_1 = m_1 \oplus t_1$ et l'envoi à Bob. Le calcul de c_1 sera rapide puisque il est fait par un simple XOR. Alors Alice et Bob peuvent simultanément calculer le bloc t_2 , etc. CFB est donc approprié pour les longs messages mais il est plus rapide que CBC.

Exercice-4 : OFB

OFB est très similaire à CFB



Refaire l'exercice avec les mêmes Plaintext, clé, IV, r que CFB. Analyser la propagation d'erreur, la sécurité de OFB et sa rapidité.

1)

Pour $1 \leq j \leq u$, Alice fait le suivant

- $O_j = E(I_j)$;
- Extraire t_j qui est les r premiers bits de O_j ;
- $c_j = m_j \oplus t_j$
- $I_{j+1} = O_j$.
- Le ciphertext sera donc $c = c_1 c_2 c_3 \dots c_u$

Le décryptage est le même, seulement la troisième étape est remplacé par $m_j = c_j \oplus t_j$

2) Si un bit du ciphertext est reçu erroné alors le plaintext sera erroné exactement dans la même position. Donc il n ya pas de propagation d'erreur.

3) le bloc t_j dépend seulement du vecteur d'initialisation I_1 et de la clé k. il peut donc être calculé par Alice et bob simultanément. Ceci est plus rapide que CFB. Mais le problème est que le chiffrement d'un bloc de plaintext dans OFB ne depend pas des blocs précédents mais seulement par sa position (ordre). Pour cela la manipulation du ciphertext par un intrus est beaucoup plus facile en OFB qu'en CFB.

$m_1=101$; $m_2=100$; $m_3=010$; $m_4=100$; $m_5=101$

j	I_j	O_j	t_j	m_j	c_j
1	1010	0101	010	101	111
2	0101	1010	101	100	001
3	1010	0101	010	010	000
4	0101	1010	101	100	001
5	1010	0101	010	101	111

- Si la même clé est réutilisée pour le chiffrement de deux plaintexts, alors l'IV doit changer. Sinon, la même séquence t_j est régénérée et à partir de deux ciphertexts $c_j = m_j \oplus t_j$ et $c'_j = m'_j \oplus t_j$, l'intrus peut constater que $c_j \oplus c'_j = m_j \oplus m'_j$. Et donc il peut déterminer m'_j s'il connaît m_j .

Exercice-5 : une ronde DES

Donner le résultat après une seule ronde de cryptage de DES sur le plaintext p :

P=0123456789ABCDEF

Avec la clé K :

K= 133457799BBCDFF1

Le codage binaire de P=0123456789ABCDEF

0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	1
0	1	0	0	0	1	0	1
0	1	1	0	0	1	1	1
1	0	0	0	1	0	0	1
1	0	1	0	1	0	1	1
1	1	0	0	1	1	0	1
1	1	1	0	1	1	1	1

Après IP on obtient

1	1	0	0	1	1	0	0
0	0	0	0	0	0	0	0
1	1	0	0	1	1	0	0
1	1	1	1	1	1	1	1
1	1	1	1	0	0	0	0
1	0	1	0	1	0	1	0
1	1	1	1	0	0	0	0
1	0	1	0	1	0	1	0

Donc

$L_0 = 11001100000000001100110011111111,$

$R_0 = 11110000101010101111000010101010.$

L'écriture binaire de la clé K= 133457799BBCDFF1

0	0	0	1	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	1	0	1	1	1
0	1	1	1	1	0	0	1
1	0	0	1	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	1	1	1	1	1
1	1	1	1	0	0	0	1

On calcule la première clé intermédiaire :

$$C_0 = 1111000011001100101010101111$$

$$D_0 = 0101010101100110011110001111$$

$$C_1 = 1110000110011001010101011111$$

$$D_1 = 1010101011001100111100011110$$

Et donc

$$K_1 = 0001101100000010111011111111000111000001110010$$

En utilisant cette clé on a

$$E(R_0) \oplus K_1 = 011000010001011110111010100001100110010100100111$$

$$f_{K_1}(R_0) = 00100011010010101010100110111011$$

Et finalement

$$R_1 = 11101111010010100110010101000100$$

Exercice-6 : RSA

Soit l'ensemble des symboles $\{0, a, b, c\}$ avec l'identification :

0	a	b	c
0	1	2	3

Soit $p = 11$; $q = 23$

1- Générer les clés publiques et privée de RSA. Choisir comme clé publique e la plus petit possible.

1)

$$n = p \cdot q = 11 \times 23 = 253$$

$$\Phi(n) = (p-1)(q-1) = 10 \cdot 22 = 220$$

$e=3$ ($e=2$ a rejeter puisque $\text{pgcd}(2,220)=2$; $e=1$ n'est clairement pas un bon choix)

chercher d tel que $d = e^{-1} \bmod \Phi(n) = 3^{-1} \bmod 220$

Il faut trouver x telque $\Phi(n) = e \cdot x + 1 \rightarrow d = \Phi(n) - x = 220 - 73 = 147$

pour cela on applique l'alg d'euclide etendu :

$$220 = 3 \cdot 73 + 1$$

$$1 = 220 - 3 \cdot 73$$

$$-73 = 3^{-1} \bmod 220 = 147 \bmod 220$$

Donc $d = 147$

La clé publique $PU=\{e,n\}=\{3,253\}$

La clé privée $PR=\{d,n\}=\{147,253\}$

2- Donner la taille maximale d'un bloc du plaintext

2) on a $n=253$, la taille de bloc du plaintext $k = E[\log_4 253] = 3$ ($\log_z(x) = \ln(x)/\ln(Z)$)

Taille max texte clair(plaintext) = $E[\log_z(n)]$ avec Z dimension de l'espace textes clairs.

3- Donner la taille maximale d'un bloc de ciphertext

3) la taille maximale d'un bloc du ciphertext sera donc $k + 1 = 4$

4- Chiffrer le message «abb »

4) le message abb correspond à $(122)_4$ d'après le tableau. 122 correspond au nombre :

$$1 * 4^2 + 2 * 4^1 + 2 * 4^0 = (26)_{10}$$

Donc $P = 26$. le chiffrement de P est donné par $C = P^e \bmod n = 26^3 \bmod 253 = (119)_{10}$

Le nombre 119 correspond à :

$$119 = 1 * 4^3 + 3 * 4^2 + 1 * 4^1 + 3 * 4^0 = (1313)_4$$

Donc le nombre decimal 119 correspond au nombre dans la base 4 : 1313 qui correspond au message «acac ».

5- Effectuer le déchiffrement pour vérification.

5) le déchiffrement se fait comme suit :

le ciphertext acac correspond à 1313 dans la base 4 qui correspond au nombre :

$$1 * 4^3 + 3 * 4^2 + 1 * 4^1 + 3 * 4^0 = 119$$

$$M = C^d \bmod n = 119^{147} \bmod 253 = ?$$

Exponentiation rapide, on a $147 = 128 + 16 + 2 + 1 = (10010011)_2$

I	7	6	5	4	3	2	1	0
b_i	1	0	0	1	0	0	1	1
Exp	1	2	4	9	18	36	73	147
Res	119	246	49	82	146	64	146	26

Donc le plaintext en decimal c'est 26 qui s'écrit en base 4 sous la forme :

$$26 = 1 * 4^2 + 2 * 4^1 + 2 * 4^0$$

Donc $26 = (122)_4$ qui correspond d'après le tableau au plaintext « abb »

Exercice-9 : Diffie-hellman

Soit $p=17$, $g=3$ des clés globales partagés entre Alice et bob.

Alice choisit $a=7$, et Bob choisit $b=4$.

Compléter le protocole de Diffie-Hellman pour partager une clé secrète.

- Alice calcule sa clé publique $A = g^a \bmod p = 3^7 \bmod 17 = 11$ et envoie A à Bob
- Bob calcule sa clé publique $B = g^b \bmod p = 3^4 \bmod 17 = 13$ et envoie B à Alice
- Alice calcule la clé secrète par $K = B^a \bmod p = 13^7 \bmod 17 = 4$
- Bob calcule la clé secrète K par $K = A^b \bmod p = 11^4 \bmod 17 = 4$