Crypto, $2^{ème}$ Semestre 2013/2014

DATA ENCRYPTION STANDARD

Description du DES.

1. Introduction

DES (DATA ENCRYPTION STANDARD) est un cryptosystème permettant de crypter des messages de 64 bits (16 chiffres hexadécimaux de 8 octets), il a été mis au point en 1977. Bien que dépassé en termes de sécurité, il est à la base de plusieurs cryptosystèmes actuelles.
Pour effectuer le chiffrement, l'algorithme DES utilise une clé de 64 bits. En fait, seulement 56 bits sont utilisé pour l'encryptage, les 8 bits restants servent de contrôle de redondance cyclique (CRC). La vulnérabilité de DES réside essentiellement de la trop faible taille de l'espace des clefs (seulement 2^{56} clefs !), la puissance actuelle des ordinateurs permet une attaque brutale par exhaustion.
En cryptologie, le message que l'on veut coder est appelé message *clair*. Celui-ci est chiffré grâce à la clef du DES. Le résultat de cette procédure est le message *chiffré*. Pour retrouver le message d'origine, on déchiffre le texte chiffré à l'aide de la même clé DES. C'est un chiffrement dit *symétrique*. Les caractères ASCII utilisé en informatique peuvent être décrit comme une chaîne de 8 bits (c'est à dire des 0 et des 1). C'est cette chaine binaire qui va être cryptée. Si le message à chiffrer n'est pas de longueur un multiple de 64 bits, on complète le message avec quelques octets supplémentaires ajoutés à la fin du texte. L'encryptage se compose de deux étapes : la création de 16 sous-clés de 48 bits de long à partir de la clef de 64 bits et le chiffrement des messages clairs à l'aide des sous-clés créés.

2. La création de 16 sous-clefs à partir d'une clef

2.1. **Étape 1 : PC-1 permutation.** Après la conversion en 64 bits de la clé de 16 chiffres hexadécimaux. Celle-ci est permutée (en fait, ce n'est pas exactement une permutation car la sortie n'a plus que 56 bits) grâce à la table PC1 ci-dessous : la première valeur dans la table PC1 est égal à 57. Cela signifie que le 57ème bit de la clé donnée K devient le premier bit de la clé permutée K^+. Le 41ème bit de la clé originale devient la troisième bit de la clé permutée et ainsi de suite. Il est important de noter que seulement 56 bits de la clé d'origine sont utilisés dans la clé permuté. Les 8ème bits de chaque octet sont ignorés et servent pour un contrôle de parité.

Permutation PC1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Exemple, la clef hex : $B0DB0DB0DB0DB0D1$ s'écrit sur 64 bits :

$$1011000011011011000011011011000011011011000011010001.$$

La clef permutée K^+ vaut alors :

$$11011011100100100100100111010001001000100100001101101011$$

2.2. **Étape 2 : Split.** La prochaine étape consiste à scinder K^+ en deux moitiés gauche et droite de 28 bits : G_0 et D_0.
Dans notre exemple, cela donne :
K^+ : 11011011100100100100100111010001001000100100001101101011
G_0 : 1101101110010010010010011101
D_0 : 0001001000100100001101101011

2.3. **Étape 3 : Permutation circulaire.** Ensuite, on forme 16 couples (G_n, D_n) pour $1 \leq n \leq 16$ à partir de (G_0, D_0) par décalage d'un ou deux crans à gauche. Chaque paire (G_n, D_n) est formée à partir de la paire précédente (G_{n-1}, D_{n-1}) en passant le premier bit (ou les 2 premiers bits) de G_{n-1} et de D_{n-1} à la fin du bloc.
Cela donne pour notre exemple :
G_0 : 11011011100100100100100111101
D_0 : 00010010001001000001101101011
G_1 : 10110111001001001001001110111 : 1 décalage
D_1 : 00100100010010000011011010110
G_2 : 01101110010010010010011101111 : 1 décalage
D_2 : 01001000100100000110110101100
G_3 : 10111001001001001001111011101 : 2 décalages
D_3 : 00100010010000011011010110001
G_4 : 11100100100100100111011101110 : 2 décalages
D_4 : 10001001000011011010110000100
G_5 : 10010010010010011101110111011 : 2 décalages
D_5 : 00100100001101101011000100 10
G_6 : 01001001001001110111011011 10 : 2 décalages
D_6 : 10010000110110101100010010 00
G_7 : 00100100100111011101101110 01 : 2 décalages
D_7 : 01000011011010110001001000 10
G_8 : 10010010011101110111011001 00 : 2 décalages
D_8 : 00001101101011000100100010 01
G_9 : 00100100111011101101110010 01 : 1 décalage
D_9 : 00011011010110001001000100 10
G_{10} : 10010011101110110111001001 00 : 2 décalages
D_{10} : 01101101011000100100010010 00
G_{11} : 01001110111011011100100100 10 : 2 décalages
D_{11} : 10110101100010010001001000 01
G_{12} : 00111011101101110010010010 01 : 2 décalages
D_{12} : 11010110001001000100100001 10
G_{13} : 11101110110111001001001001 00 : 2 décalages
D_{13} : 01011000100100010010000110 11
G_{14} : 10111011011100100100100100 11 : 2 décalages
D_{14} : 01100010010001001000011011 01
G_{15} : 11101101110010010010010011 10 : 2 décalages
D_{15} : 10001001000100100000110110101
G_{16} : 11011011100100100100100111 01 : 1 décalage
D_{16} : 00010010001001000011011010 11

2.4. **Étape 4 : Permutation PC2.** On reconcatène chaque paire de données et on permute en fonction de la table PC2 pour former la table de sous-clés K_n. Les sous-clefs K_n ont toute 48 bits.

Permutation PC2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

On obtient donc : G_1D_1 : 10110111001001001001001110110010010001001000011011010110
Puis après permutation :
K_1 : 11111011010010010101110011101100000001001010 1000

De même :

K_2 : 000101110101110010111011000100111111010001100000
K_3 : 111111100100001110001000110100010001101001000000
K_4 : 000110101100111010001101100010000110110000011110
K_5 : 010110010011000100111110011011010101001010010000
K_6 : 101001001000110011101101100100010100000001101011
K_7 : 110100110110101000100110100001101001101000000100
K_8 : 101010001011111110110000100100000010011111110100
K_9 : 111000111111000000000110110001010111000000000101
K_{10} : 001010001000111111111011000100010001011101100
K_{11} : 111101000111000000111011101100001001100110000111
K_{12} : 101001111000011101110000000001100000011010110011
K_{13} : 110011100101101010010111010111110010100101000001
K_{14} : 001111111011001101011000001000101100000101011000
K_{15} : 000011100101010011101011010000011011010100000110
K_{16} : 000111011010001111010001000011001001001101011001

3. Chiffrement du message

Rappel des principaux ingrédients.

3.1. **Étape 1 : Permutation initiale.**

Permutation IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

3.2. **Étape 2 : Split.**

3.3. **Étape 3 : Diagramme de Feistel.** L'étape suivante est constituée de 16 itérations. Pour $1 \leq n \leq 16$ on calcule : $G_n = D_{n-1}$ et $D_n = G_{n-1} \oplus f(D_{n-1}, K_n)$. La fonction f prend deux blocs de données de 32 bits et 48 bits et produit un bloc de 32 bits. Plus précisément, la fonction f procède en 4 étapes :
1. Expansion de D_{n-1} sur 48 bits
2. XOR avec une sous-clé K_n
3. S-Boîte transformation
4. Permutation P

3.3.1. *Expansion.*

Expansion E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

3.3.2. *S-boite.*

S1 :

S1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2 :

S2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3 :

S3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4 :

S4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5 :

S5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6 :

S6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S7 :

S7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8 :

S8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

3.3.3. *Étape 4 : Permutation.*

S8 :

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

3.4. **Étape 4 : Permutation finale.**

Permutation IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25