Chiffrement DES

Le système de chiffrement DES, aujourd'hui désuet, est un algorithme de chiffrement par bloc qui a longtemps été utilisé pour le chiffrement des mot de passe UNIX. Il est aussi utilisé dans (d'ancien) décodeur de HBO [1].

1970 : Horst Feistel et ses collègues d'IBM, travaillant avec un système d'exploitation appelé *DEMONSTRATION*, propose un système de chiffrement par bloc pour la banque en ligne. Le Système *DEMONSTRATION* ne supportant pas les noms long est communément appelé *DEMON*. D'où le nom de cette méthode de chiffrement : *lucifer*.

1973 : le bureau des standards américain (NBS) lance un appel à la création d'un algorithme de chiffrement utilisable par les entreprises.

1976 : après quelques "modifications" de la part de l'agence de sécurité nationale américaine (NSA), le système *lucifer* rebaptisé *DES* pour *data encryption system*, est sélectionné par le NBS.

1994 : Don Coppersmith avoue qu'en 1974, les concepteurs d'IBM avaient trouvé avant l'heure une méthode d'attaque (appelée *attaque-T*, genre d'attaque différentielle), permettant de casser DES.

1998 : la machine *deep crack* (ayant couté environ 200 000€) permet de casser DES en force brute en 56h.

99 : par l'intermédiaire de calcul distribué à travers le réseau, il a suffit de 22h pour casser DES en force brute (environ 100 000 machines connectées sur internet).

Dans tout ce chapitre nous ne travaillerons qu'en binaire.

0.1 Clef

La clef de DES est un nombre binaire sur 64 bits (8 octets) mais seul 56 bits sont actifs. En effet pour chaque octet un bit, le dernier, est réservé au contrôle de la clef. Il va permettre de s'assurer que l'octet comporte un nombre impaire de 1.

$$\begin{array}{ccccccc|c}
0&1&0&1&1&1&1&X\\
0&1&0&1&1&0&1&X\\
0&1&0&1&0&0&1&X\\
0&1&1&1&1&1&1&X\\
0&1&0&1&0&0&0&X\\
0&0&0&1&1&0&1&X\\
1&0&1&1&1&1&0&X\\
1&0&0&1&0&0&0&X
\end{array}
\quad\Longrightarrow\quad
\begin{array}{ccccccc|c}
0&1&0&1&1&1&1&0\\
0&1&0&1&1&0&1&1\\
0&1&0&1&0&0&1&0\\
0&1&1&1&1&1&1&1\\
0&1&0&1&0&0&0&1\\
0&0&0&1&1&0&1&0\\
1&0&1&1&1&1&0&0\\
1&0&0&1&0&0&0&1
\end{array}$$

Il n'y a donc que 56 bits de choix, ce que signifie, puisqu'un bit ne peut prendre que deux valeurs, qu'il y a 2^{56} clefs différentes. Ce qui représente environ 72 millions de milliard de clefs possible. Un ordinateur avec un algorithme permettant de tester une clef par seconde mettra donc environ 2 milliards d'année a tout tester.

0.2 Constante du chiffrement DES

La méthode de chiffrement DES peut être programmée dans une boite noire (dont on ne voit pas le code) avec un certain nombre de constate. Une attaque différentielle permet de ne pas prendre en compte la valeur des constantes utilisées dans cette boite. C'est pour cette raison que dans la pratique elles sont connues. Il y en a de trois natures : les fonctions de permutations, la fonction d'expansion et les matrices de substitutions.

Les fonctions de permutations. Ces fonctions que le notes dans un tableau vont permettre de mélanger les bits dans le mot. Par exemple la permutation $P = (4\ 1\ 3\ 2)$ se comprend par le placement en première position du quatrième bit, en seconde position du premier en troisième du troisième et en quatrième du second.

Si par exemple le message est $M =$ "1010" alors après application de la permutation P le nouveau message est $M' =$ "0110".

1. *Valar Morgulis*

Cinq permutations interviennent dans le système DES. Nous les notons en matrices bien qu'il faille les lire en ligne, nous n'avons simplement pas la place autrement.

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

La permutation initiale inverse qui fait simplement les association inverse de PI :

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

La permutation des rondes $\mathsf{P} = \begin{pmatrix} 16 & 7 & 20 & 21 & 29 & 12 & 28 & 17 \\ 1 & 15 & 23 & 26 & 5 & 18 & 31 & 10 \\ 2 & 8 & 24 & 14 & 32 & 27 & 3 & 9 \\ 19 & 13 & 30 & 6 & 22 & 11 & 4 & 25 \end{pmatrix}$

La première permutation des clefs

$$\mathsf{CP}_1 = \begin{pmatrix} 57 & 49 & 41 & 33 & 25 & 17 & 9 & 1 & 58 & 50 & 42 & 34 & 26 & 18 \\ 10 & 2 & 59 & 51 & 43 & 35 & 27 & 19 & 11 & 3 & 60 & 52 & 44 & 36 \\ 63 & 55 & 47 & 39 & 31 & 23 & 15 & 7 & 62 & 54 & 46 & 38 & 30 & 22 \\ 14 & 6 & 61 & 53 & 45 & 37 & 29 & 21 & 13 & 5 & 28 & 20 & 12 & 4 \end{pmatrix}$$

La seconde permutation des clefs

$$\mathsf{CP}_2 = \begin{pmatrix} 14 & 17 & 11 & 24 & 1 & 5 & 3 & 28 & 15 & 6 & 21 & 10 \\ 23 & 19 & 12 & 4 & 26 & 8 & 16 & 7 & 27 & 20 & 13 & 2 \\ 41 & 52 & 31 & 37 & 47 & 55 & 30 & 40 & 51 & 45 & 33 & 48 \\ 44 & 49 & 39 & 56 & 34 & 53 & 46 & 42 & 50 & 36 & 29 & 32 \end{pmatrix}$$

La fonction d'expansion va permettre de transformer un bloc de 32 bits en un bloc de 48 bits. Elle s'utilise comme les fonctions de permutation mais dupliquent en plus 16 bits.

$$E = \begin{pmatrix} 32 & 1 & 2 & 3 & 4 & 5 \\ 4 & 5 & 6 & 7 & 8 & 9 \\ 8 & 9 & 10 & 11 & 12 & 13 \\ 12 & 13 & 14 & 15 & 16 & 17 \\ 16 & 17 & 18 & 19 & 20 & 21 \\ 20 & 21 & 22 & 23 & 24 & 25 \\ 24 & 25 & 26 & 27 & 28 & 29 \\ 28 & 29 & 30 & 31 & 32 & 1 \end{pmatrix}$$

Les matrices de substitution sont aux nombres de 8 et sont des matrices de 4 lignes et 16 colonnes dont nous détaillerons l'utilisation le moment venu :

$$S1 = \begin{pmatrix} 14 & 4 & 13 & 1 & 2 & 15 & 11 & 8 & 3 & 10 & 6 & 12 & 5 & 9 & 0 & 7 \\ 0 & 15 & 7 & 4 & 14 & 2 & 13 & 1 & 10 & 6 & 12 & 11 & 9 & 5 & 3 & 8 \\ 4 & 1 & 14 & 8 & 13 & 6 & 2 & 11 & 15 & 12 & 9 & 7 & 3 & 10 & 5 & 0 \\ 15 & 12 & 8 & 2 & 4 & 9 & 1 & 7 & 5 & 11 & 3 & 14 & 10 & 0 & 6 & 13 \end{pmatrix}$$

$$S2 = \begin{pmatrix} 15 & 1 & 8 & 14 & 6 & 11 & 3 & 4 & 9 & 7 & 2 & 13 & 12 & 0 & 5 & 10 \\ 3 & 13 & 4 & 7 & 15 & 2 & 8 & 14 & 12 & 0 & 1 & 10 & 6 & 9 & 11 & 5 \\ 0 & 14 & 7 & 11 & 10 & 4 & 13 & 1 & 5 & 8 & 12 & 6 & 9 & 3 & 2 & 15 \\ 13 & 8 & 10 & 1 & 3 & 15 & 4 & 2 & 11 & 6 & 7 & 12 & 0 & 5 & 14 & 9 \end{pmatrix}$$

$$S3 = \begin{pmatrix} 10 & 0 & 9 & 14 & 6 & 3 & 15 & 5 & 1 & 13 & 12 & 7 & 11 & 4 & 2 & 8 \\ 13 & 7 & 0 & 9 & 3 & 4 & 6 & 10 & 2 & 8 & 5 & 14 & 12 & 11 & 15 & 1 \\ 13 & 6 & 4 & 9 & 8 & 15 & 3 & 0 & 11 & 1 & 2 & 12 & 5 & 10 & 14 & 7 \\ 1 & 10 & 13 & 0 & 6 & 9 & 8 & 7 & 4 & 15 & 14 & 3 & 11 & 5 & 2 & 12 \end{pmatrix}$$

$$S4 = \begin{pmatrix} 7 & 13 & 14 & 3 & 0 & 6 & 9 & 10 & 1 & 2 & 8 & 5 & 11 & 12 & 4 & 15 \\ 13 & 8 & 11 & 5 & 6 & 15 & 0 & 3 & 4 & 7 & 2 & 12 & 1 & 10 & 14 & 9 \\ 10 & 6 & 9 & 0 & 12 & 11 & 7 & 13 & 15 & 1 & 3 & 14 & 5 & 2 & 8 & 4 \\ 3 & 15 & 0 & 6 & 10 & 1 & 13 & 8 & 9 & 4 & 5 & 11 & 12 & 7 & 2 & 14 \end{pmatrix}$$

$$S5 = \begin{pmatrix} 2 & 12 & 4 & 1 & 7 & 10 & 11 & 6 & 8 & 5 & 3 & 15 & 13 & 0 & 14 & 9 \\ 14 & 11 & 2 & 12 & 4 & 7 & 13 & 1 & 5 & 0 & 15 & 10 & 3 & 9 & 8 & 6 \\ 4 & 2 & 1 & 11 & 10 & 13 & 7 & 8 & 15 & 9 & 12 & 5 & 6 & 3 & 0 & 14 \\ 11 & 8 & 12 & 7 & 1 & 14 & 2 & 13 & 6 & 15 & 0 & 9 & 10 & 4 & 5 & 3 \end{pmatrix}$$

$$S6 = \begin{pmatrix} 12 & 1 & 10 & 15 & 9 & 2 & 6 & 8 & 0 & 13 & 3 & 4 & 14 & 7 & 5 & 11 \\ 10 & 15 & 4 & 2 & 7 & 12 & 9 & 5 & 6 & 1 & 13 & 14 & 0 & 11 & 3 & 8 \\ 9 & 14 & 15 & 5 & 2 & 8 & 12 & 3 & 7 & 0 & 4 & 10 & 1 & 13 & 11 & 6 \\ 4 & 3 & 2 & 12 & 9 & 5 & 15 & 10 & 11 & 14 & 1 & 7 & 6 & 0 & 8 & 13 \end{pmatrix}$$

$$S7 = \begin{pmatrix} 4 & 11 & 2 & 14 & 15 & 0 & 8 & 13 & 3 & 12 & 9 & 7 & 5 & 10 & 6 & 1 \\ 13 & 0 & 11 & 7 & 4 & 9 & 1 & 10 & 14 & 3 & 5 & 12 & 2 & 15 & 8 & 6 \\ 1 & 4 & 11 & 13 & 12 & 3 & 7 & 14 & 10 & 15 & 6 & 8 & 0 & 5 & 9 & 2 \\ 6 & 11 & 13 & 8 & 1 & 4 & 10 & 7 & 9 & 5 & 0 & 15 & 14 & 2 & 3 & 12 \end{pmatrix}$$

$$S8 = \begin{pmatrix} 13 & 2 & 8 & 4 & 6 & 15 & 11 & 1 & 10 & 9 & 3 & 14 & 5 & 0 & 12 & 7 \\ 1 & 15 & 13 & 8 & 10 & 3 & 7 & 4 & 12 & 5 & 6 & 11 & 0 & 14 & 9 & 2 \\ 7 & 11 & 4 & 1 & 9 & 12 & 14 & 2 & 0 & 6 & 10 & 13 & 15 & 3 & 5 & 8 \\ 2 & 1 & 14 & 7 & 4 & 10 & 8 & 13 & 15 & 12 & 9 & 0 & 3 & 5 & 6 & 11 \end{pmatrix}$$

0.3 Logique propositionnelle

Le principe de chiffrement DES est très faible en terme de ressource de temps de calcul. Il n'y a que des substitutions, des conversions en binaire et des opérations de logique. Revenons sur les opérations de la logique.

0.3.1 Les trois opérations logiques élémentaires

Le *non* de la logique que l'on note $\neg p$ et dont la table est

p	$\neg p$
0	1
1	0

Le *ou* de la logique que l'on note $p + q$ et dont la table est

p	q	$p+q$
0	0	0
0	1	1
1	0	1
1	1	1

Le *et* de la logique que l'on note $p \times q$ ou plus simplement $p.q$ et dont la table est

p	q	$p.q$
0	0	0
0	1	0
1	0	0
1	1	1

On note 0 la proposition qui est toujours fausse, on l'appel la *contradiction*.
On note 1 la proposition qui est toujours vrai, on l'appel la *tautologie*.

0.3.2 Candimatica

Ces trois opérations satisfont certaines lois :

Proposition 0.3.1

Soient p, q et r des propositions

Commutativité. $p + q = q + p$, $p.q = q.p$

Associativité. $(p + q) + r = p + (q + r)$, $(p.q).r = p.(q.r)$

Neutralité. $p + 0 = p$, $p.1 = p$

Distributivité. $p.(q + r) = (p.q) + (p.r)$, $p + (q.r) = (p + q).(p + r)$

Idempotence. $p + p = p$, $p.p = p$

de Morgan $\neg(p + q) = (\neg p).(\neg q)$, $\neg(p.q) = (\neg p) + (\neg q)$

Absorption 1. $p + 1 = 1$, $p.0 = 0$

Tiers exclu. $p + (\neg p) = 1$

Involution. $\neg(\neg p) = p$

Contradiction. $p.(\neg p) = 0$

Absorption 2. $p + (p.q) = p$, $p.(p + q) = p$

Cette proposition se démontre en comparant par exemple les tables de vérité

Ou exclusif

Le *ou exclusif*, qui sera l'outil de la cryptographie DES, se comporte comme le *ou* classique à ceci près qu'il renvoie faux si les deux propositions sont vrais. On le note $\mathsf{p} \oplus \mathsf{q}$ et sa table de vérité est

p	q	$\mathsf{p} \oplus \mathsf{q}$
0	0	0
0	1	1
1	0	1
1	1	0

Théorème 0.3.2

1. $\mathsf{p} \oplus \mathsf{q} = (\mathsf{p} + \mathsf{q}).\neg(\mathsf{p}.\mathsf{q})$
2. $\mathsf{p} \oplus \mathsf{p} = 0$
3. $\mathsf{p} \oplus 0 = \mathsf{p}$
4. $\mathsf{p} \oplus 1 = \neg \mathsf{p}$
5. $\mathsf{p} \oplus \neg \mathsf{p} = 1$
6. $\mathsf{p} \oplus \mathsf{q} = \mathsf{q} \oplus \mathsf{p}$
7. $\mathsf{p} \oplus (\mathsf{q} \oplus \mathsf{r}) = (\mathsf{p} \oplus \mathsf{q}) \oplus \mathsf{r}$
8. $(\mathsf{p} \oplus \mathsf{q} = 0) \Leftrightarrow (\mathsf{p} = \mathsf{q})$
9. $\neg(\mathsf{p} \oplus \mathsf{q}) = (\neg \mathsf{p}) \oplus \mathsf{q} = \mathsf{p} \oplus (\neg \mathsf{q}) = (\neg \mathsf{p}) \oplus (\neg \mathsf{q})$
10. $(\mathsf{p} \oplus \mathsf{q} = \mathsf{r}) \Rightarrow (\mathsf{q} \oplus \mathsf{r} = \mathsf{p})$

Démonstration. Exercice □

0.4 Méthode de chiffrement

Création de 16 sous-clefs. La clef K est donnée sur 64 bits. On commence par supprimer les 8 bits de contrôle pour ne garder que les 56 bits utiles

$$\begin{array}{cccccccc}
0&1&0&1&1&1&1&0\\
0&1&0&1&1&0&1&1\\
0&1&0&1&0&0&1&0\\
0&1&1&1&1&1&1&1\\
0&1&0&1&0&0&0&1\\
0&0&0&1&1&0&1&0\\
1&0&1&1&1&1&0&0\\
1&0&0&1&0&0&0&1
\end{array}
\Rightarrow
\begin{array}{ccccccc|c}
0&1&0&1&1&1&1&0\\
0&1&0&1&1&0&1&1\\
0&1&0&1&0&0&1&0\\
0&1&1&1&1&1&1&1\\
0&1&0&1&0&0&0&1\\
0&0&0&1&1&0&1&0\\
1&0&1&1&1&1&0&0\\
1&0&0&1&0&0&0&1
\end{array}$$

On considère donc la clef de 56 bits $\mathsf{K} = 01011110101101010100101111110101000000110110111101001000$
On va appliquer la première permutation CP_1 :

CLEF =

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14
bit	0	1	0	1	1	1	1	0	1	0	1	1	0	1
Position	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
bit	0	1	0	1	0	0	1	0	1	1	1	1	1	1
Position	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**39**	**40**	**41**	**42**
bit	0	1	0	1	0	0	0	0	0	1	1	1	0	1
Position	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**
bit	1	0	1	1	1	1	0	1	0	0	1	0	0	0

$$CP_1 = \begin{pmatrix} 57 & 49 & 41 & 33 & 25 & 17 & 9 & 1 & 58 & 50 & 42 & 34 & 26 & 18 \\ 10 & 2 & 59 & 51 & 43 & 35 & 27 & 19 & 11 & 3 & 60 & 52 & 44 & 36 \\ 63 & 55 & 47 & 39 & 31 & 23 & 15 & 7 & 62 & 54 & 46 & 38 & 30 & 22 \\ 14 & 6 & 61 & 53 & 45 & 37 & 29 & 21 & 13 & 5 & 28 & 20 & 12 & 4 \end{pmatrix}$$

$CP_1[CLEF]$ =

Position	1_{57}	2_{49}	3_{41}	4_{33}	5_{25}	6_{17}	7_{9}	8_{1}	9_{58}	10_{50}	11_{42}	12_{34}	13_{26}	14_{18}
bit	1	1	0	0	0	0	0	0	0	0	0	1	1	1
Position	15_{10}	16_{2}	17_{59}	18_{51}	19_{43}	20_{35}	21_{27}	22_{19}	23_{11}	24_{3}	25_{60}	26_{52}	27_{44}	28_{36}
bit	1	1	0	1	0	0	1	0	0	0	1	1	1	1
Position	29_{63}	30_{55}	31_{47}	32_{39}	33_{31}	34_{23}	35_{15}	36_{7}	37_{62}	38_{54}	39_{46}	40_{38}	41_{30}	42_{22}
bit	0	0	1	0	1	1	1	1	0	1	0	0	1	0
Position	43_{14}	44_{6}	45_{61}	46_{53}	47_{45}	48_{37}	49_{29}	50_{21}	51_{13}	52_{5}	53_{28}	54_{20}	55_{12}	56_{4}
bit	0	1	0	1	1	0	1	0	1	1	1	1	1	1

On obtient alors le mélange : 11000000000111110100100011110010111101001001011010111111 On sépare la clef en deux partie de 28 bits :

$$G = 1100000000011111010010001111 \qquad D = 0010111101001001011010111111$$

On va alors réaliser le processus suivant pour obtenir 16 sous-clefs :

- Écraser G et D par leur décaler de 1 bit vers la gauche (le premier bit devenant le dernier).
- La clef K_1 est le résultat de la permutation par CP_2 de la concaténation de G et D.
- Écraser G et D par leur décaler de 1 bit vers la gauche.
- La clef K_2 est le résultat de la permutation par CP_2 de la concaténation de G et D.
- Écraser G et D par leur décaler de 1 bit vers la gauche.
- La clef K_3 est le résultat de la permutation par CP_2 de la concaténation de G et D.
- ...
- Écraser G et D par leur décaler de 1 bit vers la gauche.
- La clef K_16 est le résultat de la permutation par CP_2 de la concaténation de G et D.

La propriété de la permutation CP_2 est qu'elle supprime les bits 9, 18, 22, 25, 35, 38, 43 et 54 transformant le bloc de 56 bits en un bloc de 48 octets.

Dans notre exemple :

1. Décalage de 1 bit à gauche :

$$G = 1100000000011111010010001111 \quad \text{deviens} \quad G = 1000000000111110100100011111$$

$$D = 0010111101001001011010111111 \quad \text{deviens} \quad D = 0101111010010010110101111110$$

   On concatène G et D et on applique le mélange CP_2 (comme nous l'avons fait pour CP_1) pour obtenir :

$$K_1 = 111100110000010100011100101011111111000011101001$$

2. Décalage de 1 bit à gauche :

$$\mathsf{G} = 1000000000111110100100011111 \quad \text{deviens} \quad \mathsf{G} = 0000000001111101001000111111$$

$$\mathsf{D} = 0101111010010010110101111110 \quad \text{deviens} \quad \mathsf{D} = 1011110100100101101011111100$$

On concatène G et D et on applique le mélange CP_2 pour obtenir :

$$\mathsf{K}_2 = 101100010001111010101010011010001110111011011111$$

3. etc

Paquetage. On divise le message en paquet de 64 bits en complétant éventuellement les bits manquant par des 0 à la fin. Si par exemple le message est

$$\mathsf{M} = \begin{array}{l} 1101110010111011110001001101010 1 \\ 11100110111101111100001000110010 \\ 10011101001010110110101111100011 \\ 0011101011011111 \end{array}$$

Alors on le découpe en deux blocs :

$$\mathsf{M}_1 = \begin{array}{l} 11011100101110111100010011010101 \\ 11100110111101111100001000110010 \end{array} \qquad \mathsf{M}_2 = \begin{array}{l} 10011101001010110110101111100011 \\ 00111010110111110000000000000000 \end{array}$$

Les opérations qui suivent se font sur chacun des blocs, nous ne les illustreront qu'avec le bloc de 64 bits M_1.

Permutation initiale. On applique au bloc la permutation initiale PI donnée plus haut. On rappel que cette permutation initiale correspond à un mélange des bits :

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
bit	1	1	0	1	1	1	0	0	1	0	1	1	1	0	1	1
Position	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
bit	1	1	0	0	0	1	0	0	1	1	0	1	0	1	0	1
Position	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
bit	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	1
Position	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
bit	1	1	0	0	0	0	1	0	0	0	1	1	0	0	1	0

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

Ainsi, avec cette matrice de permutation, le premier bit du nouveau message est le bit 58, le second est le bit 50, le troisième le 42 et ainsi de suite. Le nouveau message deviens alors

Position	1_{58}	2_{50}	3_{42}	4_{34}	5_{26}	6_{18}	7_{10}	8_{2}	9_{60}	10_{52}	11_{44}	12_{36}	13_{28}	14_{20}	15_{12}	16_{4}
bit	0	1	1	1	1	1	0	1	1	0	1	0	1	0	1	1
Position	17_{62}	18_{54}	19_{46}	20_{38}	21_{30}	22_{22}	23_{14}	24_{6}	25_{64}	26_{56}	27_{48}	28_{40}	29_{32}	30_{24}	31_{16}	32_{8}
bit	0	0	1	1	1	1	0	1	0	0	1	0	1	0	1	0
Position	33_{57}	34_{49}	35_{41}	36_{33}	37_{25}	38_{17}	39_{9}	40_{1}	41_{59}	42_{51}	43_{43}	44_{35}	45_{27}	46_{19}	47_{11}	48_{3}
bit	0	1	1	1	1	1	1	1	1	0	1	1	0	0	1	0
Position	49_{61}	50_{53}	51_{45}	52_{37}	53_{29}	54_{21}	55_{13}	56_{5}	57_{63}	58_{55}	59_{47}	60_{39}	61_{31}	62_{23}	63_{15}	64_{7}
bit	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	0

Soit encore $\mathsf{PI}[\mathsf{M}_1] = 0111110110101011001111010010101001111111101100100000001111110010$

Gauche et droite. On note G la partie gauche de $\mathsf{PI}[\mathsf{M}_1]$ correspondant aux 32 premiers bits et D la partie droite correspondant aux 32 derniers.

$$\mathsf{G} = 01111101101010110011110100101010 \qquad \mathsf{D} = 01111111101100100000001111110010$$

Rondes. On va effectuer 16 *rondes*. Chacune de ces rondes suit le même schéma à ceci près qu'à la ronde k on utilisera le morceau K_k de la clef. Le schéma des rondes est le suivant :

1. On applique la fonction d'expansion au bloc D. On obtient un message $\mathsf{E}[\mathsf{D}]$ non plus sur 32 mais sur 48 bits que l'on regarde comme 12 blocs de 4 bits.
2. On calcul $\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_k$ (lors de la ronde k) en additionnant (exclusivement) avec le morceau de clef.
3. On découpe ensuite $\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_k$ en 8 blocs de 6 bits. Notons $\mathsf{B}_1, \ldots, \mathsf{B}_8$ ces 8 blocs. Le bloc $\mathsf{B}_i = x_1x_2x_3x_4x_5$ où les x_{machin} sont soit 0 soit 1. On considère l'entier n dont l'écriture binaire est x_1x_2. Il s'agit donc d'un entier entre 0 et 3. On considère m l'entier dont l'écriture binaire est $x_2x_3x_4$ ce qui correspond à un entier entre 0 et 15. On va remplacer B_i par le nombre, que l'on écrira en binaire, à l'intersection de la $(n+1)$-ième ligne et $(m+1)$-ième colonne de la matrice de substitution S_i. Il s'agit, par construction d'un bloc de 4 bits.
4. Notons $\mathbb{S}$ l'application successive des matrices de substitution, de sorte que l'on note $\mathbb{S}[\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_k]$ le message issue de l'itération précédente (en regroupant les 8 blocs de 4 bits). On applique alors la permutation des rondes. On note $\mathsf{P}[\mathbb{S}[\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_k]]$ le résultat
5. On remplace D par $\mathsf{P}[\mathbb{S}[\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_k]] \oplus \mathsf{G}$ et G par D.

Réalisation la première ronde.

1.

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
bit	0	1	1	1	1	1	1	1	1	0	1	1	0	0	1	0
Position	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
bit	0	0	0	0	0	0	1	1	1	1	1	1	0	0	1	0

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

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	32	1	2	3	4	5	4	5	6	7	8	9	8	9	10	11
bit	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1
Position	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
	12	13	12	13	14	15	16	17	16	17	18	19	20	21	20	21
bit	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
Position	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
	22	23	24	25	24	25	26	27	28	29	28	29	30	31	32	1
bit	0	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0

Ainsi $\mathsf{E}[\mathsf{D}] = 001111111111110110100100000000000011111110100100$

2. On réalise le *ou exclusif* avec la clef K_1 :

$$\begin{array}{cc} & 001111111111110110100100000000000011111110100100 \\ \oplus & 111110011000001010001110010101111111000011101001 \\ \hline & 110001100111111100101010010101111000111101001101 \end{array}$$

Ainsi $\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_1 = 110001100111111100101010010101111000111101001101$

3. On regarde $\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_1$ en 8 blocs de 6 bits.

$$\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_1 = 110001\ 100111\ 111100\ 101010\ 010101\ 111000\ 111101\ 001101$$

On va remplacer le bloc 110001 **à l'aide** S_1**.**

- Pour la ligne : le premier et dernier caractère forment 11 soit 3 en base 10.
- Pour la colonne : les autres caractères du bloc forment 1000 soit 8 en base 10.
- A l'intersection de la ligne $3+1$ et de la colonne $8+1$ de S_1 se trouve l'entier 5 qui, codé sur 4 bits, est 0101.

On va remplacer le bloc 100111 à l'aide S_2.

- Pour la ligne : le premier et dernier caractère forment 11 soit 3 en base 10.
- Pour la colonne : les autres caractères du bloc forment 0011 soit 3 en base 10.
- A l'intersection de la ligne $3+1$ et de la colonne $3+1$ de S_2 se trouve l'entier 1 qui, codé sur 4 bits, est 0001.

On va remplacer le bloc 111100 à l'aide S_3.

- Pour la ligne : le premier et dernier caractère forment 10 soit 2 en base 10.
- Pour la colonne : les autres caractères du bloc forment 1110 soit 14 en base 10.
- A l'intersection de la ligne $2+1$ et de la colonne $14+1$ de S_3 se trouve l'entier 14 qui, codé sur 4 bits, est 1110.

On va remplacer le bloc 101010 à l'aide S_4.

- Pour la ligne : le premier et dernier caractère forment 10 soit 2 en base 10.
- Pour la colonne : les autres caractères du bloc forment 0101 soit 5 en base 10.
- A l'intersection de la ligne $2+1$ et de la colonne $5+1$ de S_4 se trouve l'entier 11 qui, codé sur 4 bits, est 1011.

On va remplacer le bloc 010101 à l'aide S_5.

- Pour la ligne : le premier et dernier caractère forment 01 soit 1 en base 10.
- Pour la colonne : les autres caractères du bloc forment 1010 soit 10 en base 10.
- A l'intersection de la ligne $1+1$ et de la colonne $10+1$ de S_5 se trouve l'entier 15 qui, codé sur 4 bits, est 1111.

On va remplacer le bloc 111000 à l'aide S_6.

- Pour la ligne : le premier et dernier caractère forment 10 soit 2 en base 10.
- Pour la colonne : les autres caractères du bloc forment 1100 soit 12 en base 10.
- A l'intersection de la ligne $2+1$ et de la colonne $12+1$ de S_6 se trouve l'entier 1 qui, codé sur 4 bits, est 0001.

On va remplacer le bloc 111101 à l'aide S_7.

- Pour la ligne : le premier et dernier caractère forment 11 soit 3 en base 10.
- Pour la colonne : les autres caractères du bloc forment 1110 soit 14 en base 10.
- A l'intersection de la ligne $3+1$ et de la colonne $14+1$ de S_7 se trouve l'entier 3 qui, codé sur 4 bits, est 0011.

On va remplacer le bloc 001101 à l'aide S_8.

- Pour la ligne : le premier et dernier caractère forment 01 soit 1 en base 10.
- Pour la colonne : les autres caractères du bloc forment 0110 soit 6 en base 10.
- A l'intersection de la ligne $1+1$ et de la colonne $6+1$ de S_8 se trouve l'entier 7 qui, codé sur 4 bits, est 0111.

En conclusion nous obtenons le message :

$$\mathbb{S}[E[D] \oplus K_1] = 0101\ 0001\ 1110\ 1011\ 1111\ 0001\ 0011\ 0111$$

4. On applique la permutation des rondes $\mathbb{S}[E[D] \oplus K_1]$:

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
bit	0	1	0	1	0	0	0	1	1	1	1	0	1	0	1	1
Position	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
bit	1	1	1	1	0	0	0	1	0	0	1	1	0	1	1	1

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

Position	$\underset{16}{1}$	$\underset{7}{2}$	$\underset{20}{3}$	$\underset{21}{4}$	$\underset{29}{5}$	$\underset{12}{6}$	$\underset{28}{7}$	$\underset{17}{8}$	$\underset{1}{9}$	$\underset{15}{10}$	$\underset{23}{11}$	$\underset{26}{12}$	$\underset{5}{13}$	$\underset{18}{14}$	$\underset{31}{15}$	$\underset{10}{16}$
bit	1	0	1	0	0	0	1	1	0	1	0	0	0	1	1	1
Position	$\underset{2}{17}$	$\underset{8}{18}$	$\underset{24}{19}$	$\underset{14}{20}$	$\underset{32}{21}$	$\underset{27}{22}$	$\underset{3}{23}$	$\underset{9}{24}$	$\underset{19}{25}$	$\underset{13}{26}$	$\underset{30}{27}$	$\underset{6}{28}$	$\underset{22}{29}$	$\underset{11}{30}$	$\underset{4}{31}$	$\underset{25}{32}$
bit	1	1	1	0	1	1	0	1	1	1	1	0	0	1	1	0

Au final

$$\mathsf{P}[\mathbb{S}[\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_1]] = 10100011010001111110110111100110$$

5. Pour finir on réalise l'opération $\mathsf{P}[\mathbb{S}[\mathsf{E}[\mathsf{D}] \oplus \mathsf{K}_1]] \oplus \mathsf{G}$ qui deviendra le nouveau D et le nouveau G sera l'ancien D.

$$\begin{array}{c|cccccccccccccccccccccccccccccccc}
 & 1&0&1&0&0&0&1&1&0&1&0&0&0&1&1&1&1&1&1&0&1&1&0&1&1&1&1&0&0&1&1&0 \\
\oplus & 0&1&1&1&1&1&0&1&1&0&1&0&1&0&1&1&0&0&1&1&1&1&0&1&0&0&1&0&1&0&1&0 \\
\hline
 & 1&1&0&1&1&1&1&0&1&1&1&0&1&1&0&0&1&1&0&1&0&0&0&0&1&1&0&0&1&1&0&0
\end{array}$$

Et

$$\mathsf{G} = 01111111011001000000001111110010 \qquad \mathsf{D} = 11011110111011001101000011001100$$

On réitère alors ces rondes en changeant la clef à chaque tour.

Initialement.

$$\mathsf{G} = 01111101101010110011110100101010 \qquad \mathsf{D} = 01111111011001000000001111110010$$

A la fin de la ronde 1.

$$\mathsf{G} = 01111111011001000000001111110010 \qquad \mathsf{D} = 11011110111011001101000011001100$$

A la fin de la ronde 2.

$$\mathsf{G} = 11011110111011001101000011001100 \qquad \mathsf{D} = 01101111000010101101000101000010$$

A la fin de la ronde 3.

$$\mathsf{G} = 01101111000010101101000101000010 \qquad \mathsf{D} = 00010110110001011111000000000101$$

A la fin de la ronde 4.

$$\mathsf{G} = 00010110110001011111000000000101 \qquad \mathsf{D} = 11000010011110110010001010100101$$

A la fin de la ronde 5.

$$\mathsf{G} = 11000010011110110010001010100101 \qquad \mathsf{D} = 11011110001100011000100010010110$$

A la fin de la ronde 6.

$$\mathsf{G} = 11011110001100011000100010010110 \qquad \mathsf{D} = 00110000100011100000011111011101$$

A la fin de la ronde 7.

$$\mathsf{G} = 00110000100011100000011111011101 \qquad \mathsf{D} = 11011000000101101110110100111101$$

A la fin de la ronde 8.

$$\mathsf{G} = 11011000000101101110110100111101 \qquad \mathsf{D} = 10110001110000011011010001001001$$

A la fin de la ronde 9.

$$\mathsf{G} = 10110001110000011011010001001001 \qquad \mathsf{D} = 01100001011001111100011111101100$$

A la fin de la ronde 10.

$$\mathsf{G} = 01100001011001111100011111101100 \qquad \mathsf{D} = 00101111001001110101000100100100$$

A la fin de la ronde 11.

$$\mathsf{G} = 00101111001001110101000100100100 \qquad \mathsf{D} = 01000110000011010111100010111011$$

A la fin de la ronde 12.

$$\mathsf{G} = 01000110000011010111100010111011 \qquad \mathsf{D} = 00010111101001110010010111110111$$

A la fin de la ronde 13.

$$\mathsf{G} = 00010111101001110010010111110111 \qquad \mathsf{D} = 00101111100101010111011000111111$$

A la fin de la ronde 14.

$$\mathsf{G} = 00101111100101010111011000111111 \qquad \mathsf{D} = 01100111100101000101100001000001$$

A la fin de la ronde 15.

$$\mathsf{G} = 01100111100101000101100001000001 \qquad \mathsf{D} = 00110000110010100100001000011100$$

A la fin de la ronde 16.

$$\mathsf{G} = 00110000110010100100001000011100 \qquad \mathsf{D} = 11010101001001100001000100011010$$

Pour finir on recolle la partie gauche et droite et on fini avec le message

$$\mathsf{M}'_1 = 0011000011001010010000100001110011010101001001100001000100011010$$

Permutation initiale inverse. On applique finalement à ce message M'_1 la permutation initiale inverse qui fini alors par

$$\mathsf{PI}^{-1}[\mathsf{M}'_1] = 10001000000110110101000010001001111001011011000001001010010010000$$

Qui correspond donc au message chiffré. On réitère ensuite l'intégralité de ce processus à tous les blocs du message.