Compte rendu de l'attaque par fautes sur l'algorithme DES

Duprat Raphael

April 23, 2020

1 Introduction

Pour toute la réponse de l'exercice, nous noterons x' l'équivalent fauté d'une valeur x, et ne noterons pas systématiquement les permutations IP et IP^{-1}, lesquelles sont implicitement appliquées à chaque message clair que l'on souhaite chiffrer, et à chaque message chiffré que l'on souhaite déchiffrer.

2 Question 1

Le principe de l'attaque par faute et de changer la valeur d'un bit du message M que l'on souhaite chiffré avec une clé K tel que

$$DES(M, K) = C$$

$$DES\_faute(M, K) = C'$$

En injectant une faute, on obtient un chiffré C' différent, mais chiffré à partir du même message M et de la même clé K.

Le système DES est un réseau de Feistel de 16 tours organisé de la manière suivante : pour $1 \leq i \leq 16$, L_i la partie gauche du message (32 bits), R_i la partie droite du message (32 bits), K_i la sous clé (48 bits) générée à partir de K la clé complète

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

Avec

$$f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i))$$

$P : \{0,1\}^{48} \rightarrow \{0,1\}^{48}$ fonction de permutation,
$S : \{0,1\}^{48} \rightarrow \{0,1\}^{32}$ fonction de substitution par l'usage de 8 S-box
$E : \{0,1\}^{32} \rightarrow \{0,1\}^{48}$ fonction d'expansion.
L'attaque portera sur le dernier tour, avec pour objectif de retrouver la sous clé K_{16} de 48 bits générée à partir de la clé K complète de 64 bits. L'attaquant possède un message clair, ainsi que son équivalant chiffré, et 32 messages issus du même clair dont le chiffrement a été perturbé par une faute à la sortie de R_{15}

Pour le message donné et chaque message fauté, on a donc l'équation suivante :

$$R_{16} = L_{15} \oplus f(R_{15})$$

$$R'_{16} = L_{15} \oplus f(R'_{15})$$

$$\Leftrightarrow R_{16} \oplus R'_{16} = L_{15} \oplus L_{15} \oplus f(R_{15}) \oplus f(R'_{15})$$

$$\Leftrightarrow R_{16} \oplus R'_{16} = P(S(E(R_{15}) \oplus K_{16})) \oplus P(S(E(R'_{15}) \oplus K_{16}))$$

$$\Leftrightarrow R_{16} \oplus R'_{16} = P(S(E(R_{15}) \oplus K_{16}) \oplus S(E(R_{15}) \oplus K_{16}))$$

$$\Leftrightarrow P^{-1}(R_{16} \oplus R'_{16}) = S(E(R_{15}) \oplus K_{16}) \oplus S(E(R'_{15}) \oplus K_{16})$$

Nous remarquerons que l'influence de chaque Sbox sur le message chiffré est limité elle est observé en calculant $R_{15} \oplus R'_{15}$ pour obtenir la position de la faute (donc la S-box fautée) et $R_{16} \oplus R'_{16}$ pour obtenir les sous ensembles de 6 bits influencés par la S-box trouvée.

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

Ainsi, parmi les 32 chiffrés fautés obtenus, il est possible d'attaquer les S-boxes atteintes par le changement de bit comme indiqué dans le tableau. On doit alors résoudre l'équation pour chaque S-box S_i avec i de 1 à 8 :

$$P^{-1}(R_{16} \oplus R'_{16}) = S_i(E(R_{15}) \oplus K_{16}) \oplus S_i(E(R'_{15}) \oplus K_{16})$$

Avec $P^{-1}(R16 \oplus R16')$ de taille 4 bits (les 4 bits résultants de la Sbox que l'on attaque) et K_{16} de taille 6 bits (Les 6 bits ayant été XOR à 6 bits des messages $E(R_{15})$ et $E(R'_{15})$). Une recherche exhaustive nous donne toutes les clés possibles vérifiant l'équation, pour chaque Sbox. Le tout nécessite au plus $2 * 2^6$ opérations par message, soit 2^{12} opérations.

En comparant les clés obtenues pour chaque chiffrés, on obtient une unique clé de 6 bits qui a fonctionné pour toutes les équations de la S-box attaquée, ces 6 bits de clés sont donc ceux qui composent K_{16}

Une fois K_{16} reconstruit, il s'agit de retrouver la clé de taille 64 bits qui a généré K_{16}, pour cela on applique une recherche exhaustive sur les 8 bits manquants situés aux positions suivantes dans la clé de taille 64 : 14, 15, 19, 20, 51, 54, 58, 60

Il s'agit donc de tester les 2^8 possibilités pour retrouver la clé unique. Il n'est pas nécessaire de tester les bits de parité car ceux ci n'ont pas d'influence lors du chiffrement, on les laisse donc à 0 lors de la recherche exhaustive.

Cela fait, il ne reste qu'à ajouter les bits de parité, et nous aurons retrouvé la clé complète.

3 Question 2

Pour retrouver la clé, on applique exactement la méthode décrite en réponse à la question 1, celle qui consiste à résoudre l'équation

$$P^{-1}(R_{16} \oplus R'_{16}) = S_i(E(R_{15}) \oplus K_{16}) \oplus S_i(E(R'_{15}) \oplus K_{16})$$

En faisant une recherche exhaustive sur chaque fragment de 2^6 bits de la clé K_{16} (une recherche par S-box atteinte par la faute), conserver toutes les fragments de clés qui résolvent l'équation, puis ne garder que les fragments communs à chaque S-box
La clé de 48 bits retrouvée ainsi est :

$$010110001001001010001101000110000000010100011000$$

Cela nécessite 2^{12} opérations.

4 Question 3

Pour retrouver les 8 bits manquants, on souhaite appliquer l'opération suivante :

$$K = PC1^{-1}(PC2^{-1}(K_{16}))$$

Cependant $PC2^{-1}$ prend 48 bits en entrée et sort 56 bits, il manque donc 8 bits pour lesquels il faut trouver la position. Par analyse, on trouve les positions 14, 15, 19, 20, 51, 54, 58, 60. Il faut alors chercher une clé qui, appliquée au message clair via l'algorithme DES, donnera un chiffré identique à celui d'origine. Cela nécessite 2^8 opérations.

On obtient alors la clé suivante :

$$00110000\ 01000110\ 00101100\ 00001000\ 01100000\ 00011000\ 11010100\ 11100100$$

A laquelle on ajoute les bits de parité :

$$00110001\ 01000110\ 00101100\ 00001000\ 01100001\ 00011001\ 11010101\ 11100101$$

Ce qui donne en hexadécimal :

$$31\ 46\ 2C\ 08\ 61\ 19\ D5\ E5$$

La complexité de l'attaque complète est de 2^{12} + 2^8 opérations.

5 Question 4

Premièrement, pour identifier la position de la faute notée e, on peut calculer $D = L_{16} \oplus L'_{16}$ pour savoir quelles S-box de S_{15} ont données une sortie différentes et donc quelles S-box de S_{16} recevront une entrée différente . Ainsi, en vérifiant la propagation de la faute causée par D à travers la S-box S_{16} on peut retrouver le bit fauté en calculant $R_{16} \oplus R'_{16} \oplus e$

Ainsi, connaissant les différences avant et après l'application de S_{16} il devient possible de déduire exactement lesquelles des 2 S-boxes sont directement touchées par la faute et donc d'appliquer exactement la même attaque que pour le tour 15.

Cependant, cette méthode ne fonctionne pas pour le tour 13 et n'est pas réaliste au delà du 13ème. Suivre la propagation de l'erreur devient impossible après l'application d'une troisième sous clé.

6 Question 5

Une contre mesure possible est, une fois le message chiffré, de le déchiffrer et de vérifier que le résultat du déchiffrement est bien le message d'origine. Si les message sont différents, on déduit qu'une erreur a eu lieu et on ne donne pas le message chiffré en sortie. Cette solution double le temps de calcul.

Une autre contre mesure serait de XOR le message clair a une valeur IV générée aléatoirement dès le début de l'opération, ainsi chaque sortie chiffrée sera complètement différente et il sera impossible de déterminer l'influence de la faute. Le temps de calcul ajouté dépend de la méthode utilisée pour générer une valeur aléatoire. On pensera par exemple au générateur Yarrow inventé par Bruce Schneier, John Kelsey et Niels Ferguson pour un usage cryptographique, qui se base sur diverses sources d'entropie afin de générer une valeur aléatoire en un temps raisonnable.