Université de Paris Saclay

Calcul Sécurisé

Attaque par faute sur DES

Par :

Imad BOUKEDJANI 22305124

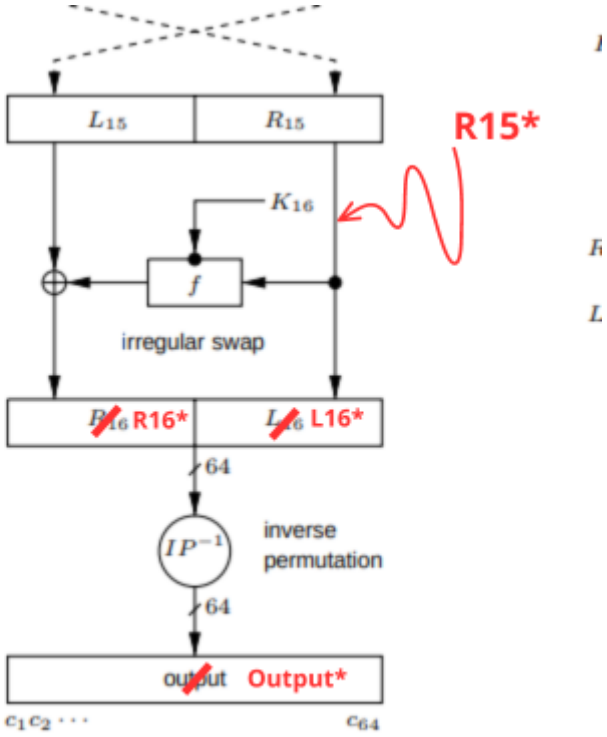
Lien Github :

https://github.com/lmaaaad/Attack-par-fautes-sur-DES

Table des matières

1 Attaque par faute sur le DES

Une attaque par faute implique de modifier le résultat d'un sous-calcul pour obtenir une information secrète. Ainsi, cette modification entraînera intentionnellement une erreur. Cette attaque est physique, car elle nécessite une intervention directe sur les composants électroniques pour altérer certaines bits. Dans le contexte du DES, une attaque par faute sur la valeur de sortie R_{15} du 15^e tour implique que l'attaquant altère volontairement la valeur de R_{15} .



À partir de cette attaque, la clé secrète utilisée par la victime peut être retrouvée à partir de la sous-clé K_{15}. On suppose ici que nous sommes l'attaquant et que nous avons réussi à obtenir de la victime le message clair associé à son message chiffré (avec une clé inconnue pour le moment, qui est à trouver). En outre, la victime a fourni 32 chiffres (toujours avec la même clé) et nous avons réussi à attaquer par erreur. Donc, afin de mener à bien cette attaque de manière adéquate, il est nécessaire de trouver K_{16} et en déduire K. La description de cette attaque sera présentée plus loin.

2 Application concrète

2.1 Question 1

Cette attaque par faute sur le dernier tour du DES comporte plusieurs étapes :

— Étape 1 : Trouver R_{15} à partir du chiffré juste et les $R_{15}*$ à partir des chiffrés faux. Pour cela, on fait une permutation initiale (qui annule la permutation finale IP^{-1}) pour trouver L_{16} et R_{16}. On fera de même pour $R_{15}*$ à partir des chiffrés faux. On peut désormais écrire les formules suivantes :

$R_{16} = L_{15} \oplus f(K_{16}, R_{15})$ et $L_{16} = R_{15}$ pour le chiffré juste.
$R_{16}* = L_{15} \oplus f(K_{16}, R_{15}*)$ et $L_{16}* = R_{15}*$ pour les chiffrés faux.

Le but ici est d'obtenir K_{16} : pour cela, on fait le XOR entre R_{16} et un $R_{16}*$. Ce qui nous donne l'équation suivante :

$R_{16} \oplus R_{16}* = L_{15} \oplus f(K_{16}, R_{15}) \oplus L_{15} \oplus f(K_{16}, R_{15}*)$
Les L_{15} s'annulent et on obtient : $R_{16} \oplus R_{16}* = f(K_{16}, R_{15}) \oplus f(K_{16}, R_{15}*)$

Or, $f(K_{i+1}, R_i) = P(S(E(R_i) \oplus K_{i+1})) = P(S_1(E(R_i) \oplus K_{i+1})_{b_1 \to b_6} || ... || S_8(E(R_i) \oplus K_{i+1})_{b_{43} \to b_{48}})$

— Étape 2 : Pour chaque chiffré faux (associé à son $R_{15}*$), établir 8 équations pour chaque boîte-S.

$$\begin{cases} P^{-1}(R_{16} \oplus R_{16}*)_{b_1 \to b_4} & = S_1(E(R_{15}) \oplus K_{16})_{b_1 \to b_4} \oplus S_1(E(R_{15}*) \oplus K_{16})_{b_1 \to b_4} \\ P^{-1}(R_{16} \oplus R_{16}*)_{b_5 \to b_8} & = S_2(E(R_{15}) \oplus K_{16})_{b_5 \to b_8} \oplus S_2(E(R_{15}*) \oplus K_{16})_{b_5 \to b_8} \\ P^{-1}(R_{16} \oplus R_{16}*)_{b_9 \to b_{12}} & = S_3(E(R_{15}) \oplus K_{16})_{b_9 \to b_{12}} \oplus S_3(E(R_{15}*) \oplus K_{16})_{b_9 \to b_{12}} \\ P^{-1}(R_{16} \oplus R_{16}*)_{b_{13} \to b_{16}} & = S_4(E(R_{15}) \oplus K_{16})_{b_{13} \to b_{16}} \oplus S_4(E(R_{15}*) \oplus K_{16})_{b_{13} \to b_{16}} \\ P^{-1}(R_{16} \oplus R_{16}*)_{b_{17} \to b_{20}} & = S_5(E(R_{15}) \oplus K_{16})_{b_{17} \to b_{20}} \oplus S_5(E(R_{15}*) \oplus K_{16})_{b_{17} \to b_{20}} \\ P^{-1}(R_{16} \oplus R_{16}*)_{b_{21} \to b_{24}} & = S_6(E(R_{15}) \oplus K_{16})_{b_{21} \to b_{24}} \oplus S_6(E(R_{15}*) \oplus K_{16})_{b_{21} \to b_{24}} \\ P^{-1}(R_{16} \oplus R_{16}*)_{b_{25} \to b_{28}} & = S_7(E(R_{15}) \oplus K_{16})_{b_{25} \to b_{28}} \oplus S_7(E(R_{15}*) \oplus K_{16})_{b_{25} \to b_{28}} \\ P^{-1}(R_{16} \oplus R_{16}*)_{b_{29} \to b_{32}} & = S_8(E(R_{15}) \oplus K_{16})_{b_{29} \to b_{32}} \oplus S_8(E(R_{15}*) \oplus K_{16})_{b_{29} \to b_{32}} \end{cases}$$

— Étape 3 : Éliminer les équations dont $P^{-1}(R_{16} \oplus R_{16}*)_{b_x \to b_y}$ vaut 0 puisqu'elles n'apporteront aucune information sur la portion de sous clé K_{16}.

— Étape 4 : Faire une attaque exhaustive(Brute-Force) sur la sortie de chaque boîte-S $S_z(E(R_{15}) \oplus K_{16})_{b_x \to b_y}$, pour chaque élément, on déduit les possibles valeurs d'entrée de la boîte-S $E(R_{15}) \oplus K_{16}$. Ainsi, on en déduit une possible K_{16}.

— Étape 5 : Pour chaque K_{16} possible, calculer $S_z(E(R_{15}*) \oplus K_{16})_{b_x \to b_y}$ et regarder si $P^{-1}(R_{16} \oplus R_{16}*)_{b_x \to b_y} = S_z(E(R_{15}) \oplus K_{16})_{b_x \to b_y} \oplus S_z(E(R_{15}*) \oplus K_{16})_{b_x \to b_y}$.

Si c'est le cas, alors K_{16} en question devient une portion de clé candidate pour le chiffré faux associé.

— Étape 6 : Pour chaque boîte-S, il y a une liste de clés candidates pour chaque chiffré faux qui agit sur la portion de sous clé. Trouver pour chaque boîte-S l'intersertion des portions de clés candidates. Il y aura donc 1 portion de clé (sur 6 bits) par boîte-S.

— Étape 7 : Concaténer les 8×6 bits pour obtenir la sous clé K_{16}.

Dans notre cas, le message clair, le chiffré correct et les 32 chiffrés fautés nous sont déja fournis, ce qui équivaut a l'étape 1. On poursuit alors les opérations explicitées précédemment jusqu'a obtenir les ensembles d'ensembles de S-Box possibles au cours de l'étape 4.

1. Boîte-S 1 : L'intersection des sous-ensembles vaut : 110011
2. Boîte-S 2 : L'intersection des sous-ensembles vaut : 111010
3. Boîte-S 3 : L'intersection des sous-ensembles vaut : 100010
4. Boîte-S 4 : L'intersection des sous-ensembles vaut : 111010
5. Boîte-S 5 : L'intersection des sous-ensembles vaut : 111011
6. Boîte-S 6 : L'intersection des sous-ensembles vaut : 110011
7. Boîte-S 7 : L'intersection des sous-ensembles vaut : 001001
8. Boîte-S 8 : L'intersection des sous-ensembles vaut : 110010

2.2 Question 2

On a donc en binaire : 1100111110101000101110101110111100110010011110010

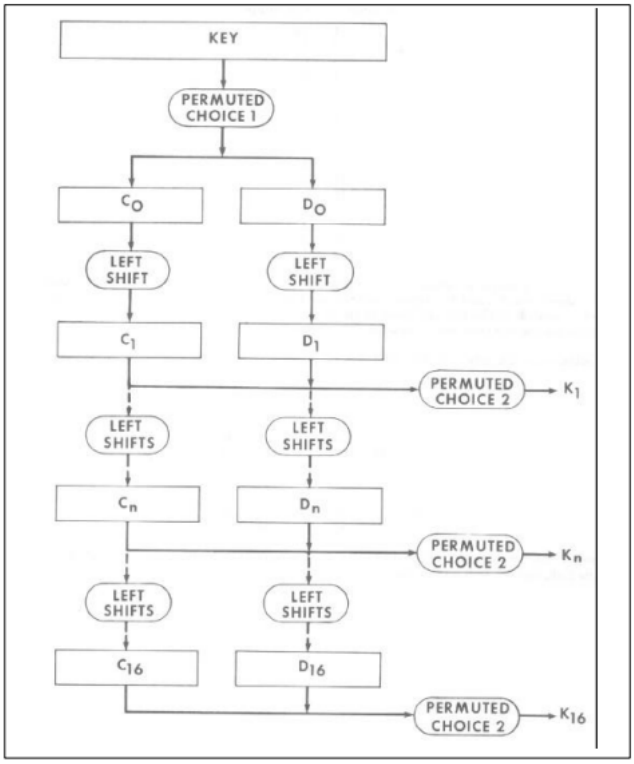
On obtient en hexadécimal pour K_{16} :$K_{16} = CF\ A8\ BA\ EF\ 32\ 72$

3 Retrouver la clé complète du DES

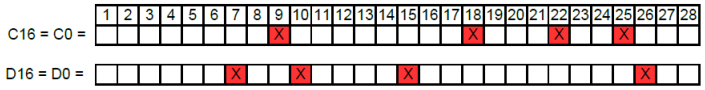
3.1 Question 1

Dans la question précédente, nous avons réussi à obtenir la clé secrète K_{16} qui contient 48 bits. En analysant le schéma de création des 16 sous-clés (de 48 bits chacunes) à partir

de la clé secrète (de 64 bits avec les 8 bits de parité), on peut en déduire la clé secrète a partir de : $K = PC^{-1}(PC^{-2}(K_{16}))$.

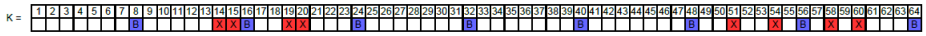


— Étape 1 : effectuer une permutation inverse $PC_2^{-1}(K_{16})$ afin d'obtenir C_{16} et D_{16}. (C et D sont des registres sur 28 bits) Il est important de noter que lors de cette permutation inverse, 8 bits sont inconnus. En effet, on passe de 48 bits à $2 \times 28 = 56$ bits. Il y a donc 8 bits qu'on ne peut déduire à partir de K_{16}.



Et on sait que $C_0 = C_{16}$ et $D_0 = D_{16}$ puisque la somme des shifts circulaires donne 28 qui correspond à la taille des blocs C_i et D_i , donc meme la taille des X ne changera pas

— Étape 2 : Retrouver les 8 bits inconnus par une recherche exhaustive : $2^8 = 256$ cas possibles , dans une boucle on effectuer aussi une permutation inverse $PC_1^{-1}(C_0||D_0)$ avec les cas possible afin d'obtenir la clé de 56 bits . apres cela, il nous reste les 8 bits de parité on nous assurant que chaque octet possède un nombre impair de bits à 1 . On on confirme les resultats avec une calculatrice en ligne (qui est fournie dans le sujet) jusqu'à ce que la clé nous donne le chiffré correct fourni .



```python
def recover_K(K16):

    K = reverse_PC2(K16)
    expected_output = str(plaintext)[2:].upper()

    # The parameters for the URL
    iv = '&iv=0000000000000000'
    input = '&input=' + str(cipher_corr)[2:].upper()
    mode = '&mode=ecb'
    action = '&action=Decrypt'
    output = '&output='

    for i in range(256):                #starting th
        key1 = replace_x_bits(K, i)
        key2 = reverse_PC1(key1)
        key3 = add_parity_bits(key2)

        key4 = hex(int(key3, 2))[2:].upper()
```

3.2 Question 2

En appliquant la méthode décrite précédemment, on obtient la valeur de clé secréte suivante :

$$K_{64} = 49\ F4\ 64\ F7\ 1C\ 9E\ 3E\ E3$$

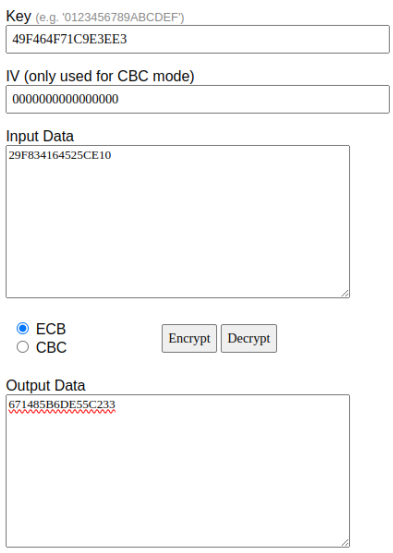


FIGURE 1 – Résultat Obtenue

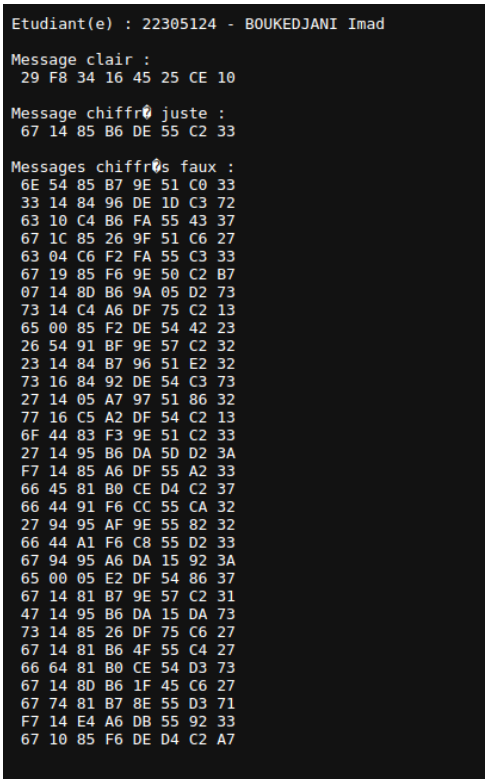


FIGURE 2 – Résultat Demandé

4 Fautes sur les tours précédents

4.1 Faute provoquée sur la valeur de sortie R_{14} du 14^e tour

On sait que :

— $R_{15} = L_{14} \oplus f(K_{15}, R_{14})$ et $L_{15} = R_{14}$
— $R_{15*} = L_{14} * \oplus f(K_{15}, R_{14}*)$ et $L_{15*} = R_{14}*$

On obtient donc les équations suivantes :
$R_{15} \oplus R_{15}* = L_{14} \oplus f(K_{15}, R_{14}) \oplus L_{14} \oplus f(K_{15}, R_{14}*) = f(K_{15}, R_{14}) \oplus f(K_{15}, R_{14}*)$

On décompose pour obtenir (E) (R_{15} **et** R_{15*} **et** R_{16} **et** R_{16*} **et** L_{16} **et** R_{16*} **sont connues**) :

$P^{-1}(R_{15} \oplus R_{15}*) = S_i(E(\ R_{14}) \oplus\ K_{15}) \oplus S_i(E(\ R_{14}*) \oplus\ K_{15})$

Or, $R_{14} = L_{15} = R_{16} \oplus f(K_{16}, L_{16})$ donne $P^{-1}(\ L_{15} \oplus R_{16})_{b_x \to b_y} = S_i(E(L_{16}) \oplus\ K_{16})_{b_x \to b_y}$ et
$R_{14}* = L_{15}* = R_{16} * \oplus f(K_{16}, L_{16}*)$ donne $P^{-1}(\ L_{15} * \oplus R_{16}*)_{b_x \to b_y} = S_i(E(L_{16}*) \oplus K_{16})_{b_x \to b_y}$.
On fait donc une attaque exhaustive et on déduit les valeurs possibles pour L_{15} et les $L_{15}*$. Cette attaque a donc une complexité de $O(2^{14})$.

Ensuite, pour chaque L_{15} et $L_{15}*$ possible, on fait une attaque de complexité de $O(2^{14})$ pour trouver K_{15} avec l'équation (E).
On a donc une complexité de $O(2^{14} \times 2^{14}) = O(2^{28})$ sur l'attaque par faute provoquée sur la valeur de sortie R_{14} du 14^e tour. Cette attaque reste **réaliste** car sa complexité est inférieure à $O(2^{80})$.

4.2 Faute provoquée sur la valeur de sortie R_i du i^e tour

Notons $O(2^a)$ la complexité de l'attaque DFA sur DES sur la valeur de sortie de R_{15}. Avec le paragraphe précédent, nous avons élaboré une attaque qui fonctionne sur n'importe quelle valeur de sortie de R_i en multipliant la complexité de l'attaque du tour précédent par $O(2^a)$. On obtient donc les résultats suivants :

— Pour l'attaque sur le 15^e tour, la complexité est de 2^{14}, qui est une attaque textbfréaliste.
— Pour l'attaque sur le 14^e tour, la complexité est de 2^{28}, qui est une attaque textbfréaliste.
— Pour l'attaque sur le 13^e tour, la complexité est de 2^{42}, qui est une attaque textbfréaliste.
— Pour l'attaque sur le 12^e tour, la complexité est de 2^{56}, qui est une attaque textbfréaliste.
— Pour l'attaque sur le 11^e tour, la complexité est de 2^{70}, qui est encore une attaque textbfréaliste.

— Pour l'attaque sur le 10^e tour, la complexité est de 2^{84}, qui est une attaque textbf-non réaliste de nos jours puisqu'elle est **supérieure à** $O(2^{80})$.

5 Contre-mesures

On distingue deux types de contre-mesures contre les attaques par injection de fautes. Les contre-mesures **algorithmiques** (qui utilisent les données calculées par la carte pour mettre en place un mécanisme de vérification) et les contre-mesures **physiques** (qui cherchent à protéger directement les composants de la carte d'une injection d'erreur).

— **Contre-mesures algorithmiques :** Comme expliqué précédemment, on cherche à mettre en place un mécanisme de vérification des calculs de la carte, cela en dégradant le moins possible le temps d'exécution de l'algorithme. La méthode la plus intuitive est de demander à la carte de calculer deux fois le même chiffrement et de comparer les chiffrés obtenus en sortie. Si ces chiffrés diffèrent, alors la carte refuse de renvoyer un quelconque résultat. Cette protection peut être valable car l'injection d'erreur est a priori aléatoire : il est assez peu probable que deux injections d'erreurs successives produisent la même erreur sur les mêmes bits. Ainsi, à quelques rares exceptions près, la carte ne renverra pas de chiffrés fautés : il n'est alors plus possible de mener l'attaque différentielle par injection de fautes. Le temps de calcul est alors deux fois celui du DES normal, ce qui reste acceptable étant donné qu'en pratique on utilise plutôt le triple DES (donc trois fois le temps d'un DES).

— **Contre-mesures Physiques :** Une idée de protection physique serait d'ajouter des sondes pour surveiller les tensions et les températures dans les différents composants de la carte. Si certains seuils de tension ou de température sont dépassés, la carte considère qu'elle n'est plus en état de fonctionnement normal et cesse son activité jusqu'à ce que les seuils redeviennent acceptables. Avec ce type de protection, le temps de calcul reste inchangé. Cependant, cette méthode n'est pas toujours adaptée et nécessite une réflexion préalable : il est crucial de s'assurer que des éléments extérieurs naturels (comme une température plus élevée dans un pays chaud) ne provoquent pas le dépassement des seuils, ce qui arrêterait le fonctionnement de la carte.