Calcul sécurisé - Attaque par faute sur DES

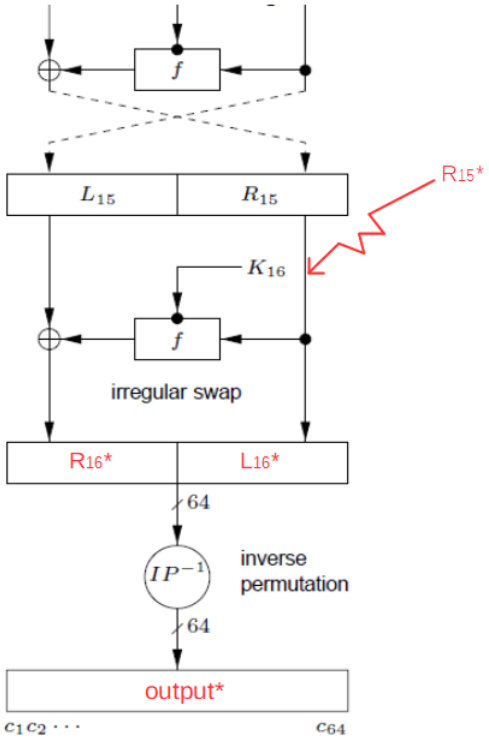
CAUMES Clément 21501810

Master 1 Informatique SeCReTs
www.github.com/Heisenberk/DFA-DES

Table des matières

1 Partie 1 : Attaque par faute sur le DES

Une attaque par faute consiste à changer le résultat d'un sous calcul afin d'obtenir une information secrète. Ce changement va donc produire volontairement une erreur. Cette attaque est physique car, pour modifier la valeur de certains bits, il est nécessaire d'agir physiquement sur les composants électroniques. Dans le cas du DES avec une attaque par faute sur la valeur de sortie R_{15} du 15^e tour, cela signifie que la valeur R_{15} va être changer par l'attaquant.



A partir de cette attaque, il est possible de retrouver la clé secrète utilisée par la victime 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). De plus, nous avons eu de la victime 32 chiffrés (toujours avec la même clé) et dont on a réussi à faire une attaque par faute. Ainsi, pour mener correctement à bien cette attaque, il faut trouver K_{16} puis en déduire K. Le détail de cette attaque sera montré par la suite.

2 Partie 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.

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

— étape 3 : éliminer les équations dont $P^{-1}(R_{16} \oplus R_{16}*)_{b_x \to b_y}$ vaut 0, ainsi que celle dont $S_z(E(R_{15}) \oplus K_{16})_{b_x \to b_y} = S_z(E(R_{15}*) \oplus K_{16})_{b_x \to b_y}$ puisqu'elles n'apporteront aucune information sur la portion de sous clé K_{16}.

— étape 4 : faire une attaque exhaustive sur la sortie de chaque boîte-S de $S_z(E(R_{15}) \oplus K_{16})_{b_x \to b_y}$: pour chaque élément, on déduit les possibles valeurs d'entrée de la boîte-S $E(R_{15}) \oplus K_{16}$. (En effet, avec la configuration non linéaire des boîtes-S, on a 4 valeurs d'entrée possibles par valeur de sortie.) 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}.

La complexité pour trouver K_{16} correspond à la recherche exhaustive de l'étape 4 :
$O(32 \times 8 \times 2^4 \times 4)$ car pour chaque chiffré faux (dans notre cas 32), on regarde pour chaque boîte-S (8 dans le DES) les 4 entrées possibles (1 sortie vaut 4 entrées différentes dans une boîte-S).
Donc on a une complexité de $O(2^5 \times 2^3 \times 2^4 \times 2^2) = O(2^{14})$ pour trouver K_{16}.

2.2 Question 2

Après avoir implémenter les fonctions "*attack_sbox*", "*find_intersection*" et "*find_K16*" du fichier *app/src/attack.c* fournis en annexe, on obtient pour chaque boîte-S les résultats suivants :

1. Boîte-S 1 :
   — Chiffré faux 1 : 0 4 5 10 14 19 20 24 25 30 34 39 ⟶ 12 portions de sous clés possibles
   — Chiffré faux 28 : 0 1 4 5 *a* *b* 16 17 24 25 34 35 ⟶ 12 portions de sous clés possibles
   — Chiffré faux 29 : 0 2 5 7 ⟶ 4 portions de sous clés possibles
   — Chiffré faux 30 : 0 1 4 5 9 d 20 23 24 27 ⟶ 10 portions de sous clés possibles
   — Chiffré faux 31 : 0 8 24 2c ⟶ 4 portions de sous clés possibles
   — Chiffré faux 32 : 0 e 10 1e 20 30 ⟶ 6 portions de sous clés possibles
   Portion de clé trouvée : 0 ⟶ 000000 en binaire
2. Boîte-S 2 :
   — Chiffré faux 24 : 2 3 10 11 12 13 1e 1f 20 21 30 31 34 35 ⟶ 14 portions de sous clés possibles
   — Chiffré faux 25 : 21 23 2d 2f ⟶ 4 portions de sous clés possibles
   — Chiffré faux 26 : 9 d 21 25 ⟶ 4 portions de sous clés possibles
   — Chiffré faux 27 : 10 14 18 1c 21 29 ⟶ 6 portions de sous clés possibles
   — Chiffré faux 28 : 4 f 14 1f 20 21 30 31 ⟶ 8 portions de sous clés possibles
   — Chiffré faux 29 : 1 4 7 17 1a 1f 21 24 27 37 3a 3f ⟶ 12 portions de sous clés possibles
   Portion de clé trouvée : 21 ⟶ 100001 en binaire
3. Boîte-S 3 :
   — Chiffré faux 20 : 2 3 ⟶ 2 portions de sous clés possibles
   — Chiffré faux 21 : 0 2 1d 1f ⟶ 4 portions de sous clés possibles
   — Chiffré faux 22 : 2 6 13 17 20 24 28 2c ⟶ 8 portions de sous clés possibles
   — Chiffré faux 23 : 2 7 a f 21 29 ⟶ 6 portions de sous clés possibles
   — Chiffré faux 24 : 2 d 12 1d 2d 3d ⟶ 6 portions de sous clés possibles
   — Chiffré faux 25 : 2 22 ⟶ 2 portions de sous clés possibles
   Portion de clé trouvée : 2 ⟶ 000010 en binaire

4. Boîte-S 4 :
   - Chiffré faux 16 : c d 10 11 14 15 18 19 20 21 26 27 30 31 32 33 ⟶ 16 portions de sous clés possibles
   - Chiffré faux 17 : 2c 2d 2e 2f 30 31 32 33 ⟶ 8 portions de sous clés possibles
   - Chiffré faux 18 : a e 31 35 ⟶ 4 portions de sous clés possibles
   - Chiffré faux 19 : 0 4 7 8 c f 31 39 ⟶ 8 portions de sous clés possibles
   - Chiffré faux 20 : e f 1e 1f 20 21 30 31 ⟶ 8 portions de sous clés possibles
   - Chiffré faux 21 : 4 5 10 11 24 25 30 31 ⟶ 8 portions de sous clés possibles

   Portion de clé trouvée : 31 ⟶ 110001 en binaire
5. Boîte-S 5 :
   - Chiffré faux 12 : 4 5 8 9 a *b* 1a 1b 28 29 3c 3d –> 12 portions de sous clés possibles
   - Chiffré faux 13 : 8 9 a *b* d f –> 6 portions de sous clés possibles
   - Chiffré faux 14 : 0 4 9 *b* d f 33 37 38 3c –> 10 portions de sous clés possibles
   - Chiffré faux 15 : 3 6 *b* e 13 15 1b 1d 35 3d –> 10 portions de sous clés possibles
   - Chiffré faux 16 : 3 6 a *b* 13 16 1a 1b 27 2a 37 3a –> 12 portions de sous clés possibles
   - Chiffré faux 17 : 6 *b* d 1f 26 2b 2d 3f –> 8 portions de sous clés possibles

   Portion de clé trouvée : b ⟶ 001011 en binaire
6. Boîte-S 6 :
   - Chiffré faux 8 : 4 5 14 15 18 19 ⟶ 6 portions de sous clés possibles
   - Chiffré faux 9 : 4 6 14 16 38 3a ⟶ 6 portions de sous clés possibles
   - Chiffré faux 10 : 0 4 20 21 24 25 29 2d ⟶ 8 portions de sous clés possibles
   - Chiffré faux 11 : 4 c ⟶ 2 portions de sous clés possibles
   - Chiffré faux 12 : 3 4 5 6 13 14 15 16 ⟶ 8 portions de sous clés possibles
   - Chiffré faux 13 : 0 4 1b 20 24 3b ⟶ 6 portions de sous clés possibles

   Portion de clé trouvée : 4 ⟶ 000100 en binaire
7. Boîte-S 7 :
   - Chiffré faux 4 : 2 3 8 9 1c 1*d* ⟶ 6 portions de sous clés possibles
   - Chiffré faux 5 : 1*d* 1f 24 26 ⟶ 4 portions de sous clés possibles
   - Chiffré faux 6 : 19 1*d* ⟶ 2 portions de sous clés possibles
   - Chiffré faux 7 : 6 7 e f 15 1*d* 30 32 34 38 3a 3c ⟶ 12 portions de sous clés possibles
   - Chiffré faux 8 : d e 1*d* 1e 25 28 2a 35 38 3a ⟶ 10 portions de sous clés possibles
   - Chiffré faux 9 : 0 2 4 c 14 18 1*d* 20 22 24 2c 34 38 3d ⟶ 14 portions de sous clés possibles

   Portion de clé trouvée : 1d ⟶ 011101 en binaire
8. Boîte-S 8 :
   - Chiffré faux 1 : 5 7 9 b c e 24 26 39 3b ⟶ 10 portions de sous clés possibles
   - Chiffré faux 2 : 8 b c f 20 24 28 2c 31 35 39 3d ⟶ 12 portions de sous clés possibles
   - Chiffré faux 3 : 10 18 31 34 35 39 3c 3d ⟶ 8 portions de sous clés possibles
   - Chiffré faux 4 : 8 9 18 19 29 39 ⟶ 6 portions de sous clés possibles
   - Chiffré faux 5 : 2 6 9 d 12 15 19 1a 22 26 29 2d 32 35 39 3a ⟶ 16 portions de sous clés possibles
   - Chiffré faux 32 : 38 39 ⟶ 2 portions de sous clés possibles

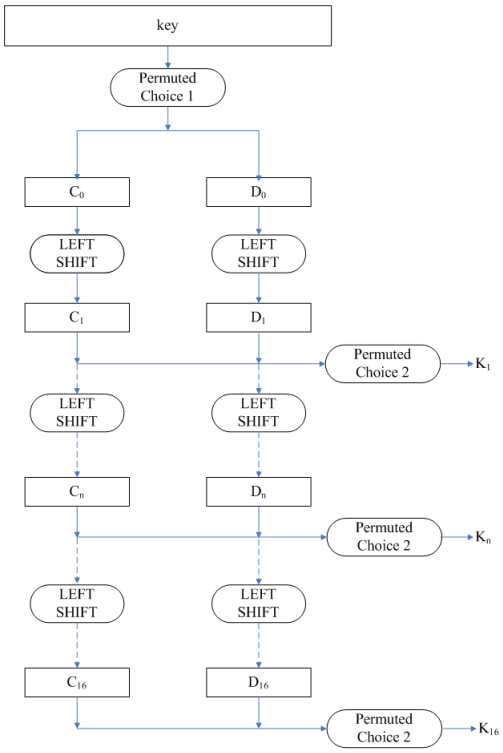
   Portion de clé trouvée : 39 ⟶ 111001 en binaire

On a donc en binaire : 000000 100001 000010 110001 001011 000100 011101 111001
soit 00000010 00010000 10110001 00101100 01000111 01111001 toujours en binaire.
On obtient en hexadécimal pour K_{16} : $K_{16} = 02\ 10\ B1\ 2C\ 47\ 79$

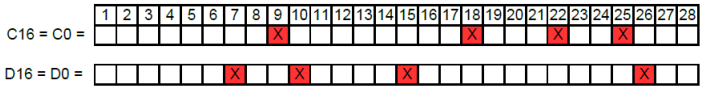
3 Partie 3 : Retrouver la clé complète du DES

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. Cela comporte plusieurs étapes :



— étape 1 : effectuer une permutation inverse $PC_2^{-1}(K_{16})$ afin d'obtenir C_{16} et D_{16}. 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}.

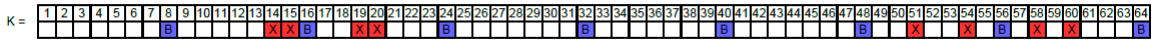


Sur ce schéma, les bits encore inconnus sont marqués par "X".

— étape 2 : déduire C_0 et D_0 à partir de C_{16} et D_{16}. En effet, $C_0 = C_{16}$ 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. Les 8 bits inconnus n'ont donc pas bougé de place dans C_0 et D_0.

— étape 3 : effectuer une permutation inverse $PC_1^{-1}(C_0||D_0)$ afin d'obtenir la clé finale sur 64 bits. On peut noter ici que les 8 bits inconnus sont mélangés dans la clé finale. De plus, 8

bits supplémentaires ont été rajoutés correspondant aux bits de parité. On a donc une clé finale sur 64 bits dont 16 bits sont inconnus.



Sur le schéma ci-dessous, "B" correspond aux bits de parité.

— étape 4 : retrouver les 8 bits inconnus par une recherche exhaustive : $2^8 = 256$ cas possibles. Pour chaque clé possible, on fait le chiffrement DES avec en entrée la clé possible et le message clair fourni. Si la clé testée est celle recherchée, alors le chiffré obtenu sera identique à celui du chiffré correct fourni.

— étape 5 : déduire les 8 bits de parité en s'assurant que chaque octet possède un nombre impair de bits à 1.

La complexité pour trouver K à partir de K_{16} est de $O(2^8)$ car à partir de K_{16}, 8 bits sont encore inconnus pour K sur 56 bits, d'où une recherche exhaustive de $O(2^8)$. A partir de cela, les 8 bits de parité (inconnus sur K de 64 bits) sont à déduire en temps constant.

On aura donc une complexité totale pour trouver K_{16} puis K de $O(2^{14} + 2^8) \simeq O(2^{14})$

3.2 Question 2

Après avoir implémenter les fonctions "$build\_C16\_D16$", "$build\_K56$, "$build\_K$", "set_parity_bits" et "$find\_K$" du fichier $app/src/attack.c$ fournis en annexe, on va effectuer les différentes étapes en détaillant :

— On a obtenu précédemment lors de la recherche de la sous clé : $K_{16} = 02\ 10\ B1\ 2C\ 47\ 79$.

Avec l'étape 1 et 2, on obtient :
- $C_0 = C_{16} = 0110\ 00100100\ 00010000\ 00000100 = 6241004$
- $D_0 = D_{16} = 0011\ 00000010\ 01001110\ 11001011 = 3024ECB$
- $C_0||D_0 = 62\ 41\ 00\ 43\ 02\ 4E\ CB$

Les 8 bits inconnus à chercher sont à 0.

— Avec l'étape 3, on obtient la clé DES sur 64 bits avec les 8 bits inconnus à 0 et les bits de parité également à 0 : $K_{64*avant\_recherche\_exhaustive} = 50\ 90\ 0C\ 18\ 02\ 8E\ D8\ 08$

— On réalise l'attaque exhaustive et on trouve la solution en ajustant également les bits de parité et on obtient la solution finale de la clé DES sur 64 bits en hexadécimal :
$K_{64} = 51\ 92\ 2C\ 19\ 02\ 8F\ FD\ 49$

4 Partie 4 : Fautes sur les tours précédents

On rappelle la loi de Moore qui dit que la puissance de calcul double tous les 18 mois. Pour rendre une attaque infaisable dans le monde civil, il faudrait 2^{80} opérations élémentaires.
De plus, nous avons trouvé une complexité de $0(2^{14})$ pour une attaque DFA sur la valeur de sortie R_{15}.

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

Pour ce type d'attaque, on obtient 2 paires d'égalités :

— $R_{15} = L_{14} \oplus f(K_{15}, R_{14})$ 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) (en rouge les valeurs inconnues et vert les valeurs 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 $O(32 \times 8 \times 2^4 \times 4) = O(2^5 \times 2^3 \times 2^4 \times 2^2) = O(2^{14})$.

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 réaliste.
- Pour l'attaque sur le 14^e tour, la complexité est de 2^{28}, qui est une attaque réaliste.
- Pour l'attaque sur le 13^e tour, la complexité est de 2^{42}, qui est une attaque réaliste.
- Pour l'attaque sur le 12^e tour, la complexité est de 2^{56}, qui est une attaque réaliste.
- Pour l'attaque sur le 11^e tour, la complexité est de 2^{70}, qui est encore une attaque réaliste.
- Pour l'attaque sur le 10^e tour, la complexité est de 2^{84}, qui est une attaque non réaliste de nos jours puisqu'elle est supérieure à $O(2^{80})$.

5 Partie 5 : Contre-mesures

Il existe plusieurs contre-mesures possibles contre ce type d'attaque par fautes sur le DES :

- installer une sorte de "bouclier" contre les perturbations extérieures. En effet, cette attaque par faute peut être réalisée en agissant physiquement sur les composants électroniques, tels que du laser, une hausse de température ou une modification du champ magnétique environnant. Ainsi, un bouclier physique permettrait de contrer ce type d'attaque. Il n'y aura aucun impact sur le temps de calcul par rapport à une implémentation non sécurisée puisqu'on ne change pas le logiciel. Malheureusement, installer ce dispositif coûte cher et est inadapté pour certains appareils.

- réaliser deux fois le calcul afin de vérifier que l'on obtient deux fois le même calcul. Cela permettrait d'annuler le calcul si une attaque de ce genre était effectuée. Il y aura un impact sur le temps de calcul : vu qu'on réalisera deux fois le même calcul, alors le temps de calcul va doubler par rapport à une implémentation non sécurisée.