CR TD AES

L’objectif du TD est de programmer les différentes fonctions qui permettent de chiffrer un message avec le chiffrement AES. Etant plus à l’aise avec le C++, j’ai pris l’initiative de programmer avec ce langage qui diffère peu du C. Voici les différentes étapes du programme :

1. Chiffrement en bloc

Chaque message de 128 octets doit être placé dans un tableau d’unsigned char 4x4 composé de 2 octets pas case du tableau. Nous illustrerons ce compte rendu avec l’exemple étudié en cas. Dans un souci pratique, j’ai rentré le message en dur dans le code mais j’avais également développé une fonction de saisie utilisateur.

Une image contenant texte

Description générée automatiquement

J’ai également réalisé une simple fonction d’affichage displayMatrice44 qui permet de visualiser tout ça sous forme d’un tableau de 4x4 :

Une image contenant texte

Description générée automatiquement

Et voici le résultat dans la console :

Une image contenant table

Description générée automatiquement

1. SubBytes

On réalise ensuite l’étape de SubBytes. Dans cette étape, on lit dans la S\_Box la valeur par laquelle il faut remplacer chaque case du tableau :

Une image contenant texte

Description générée automatiquement

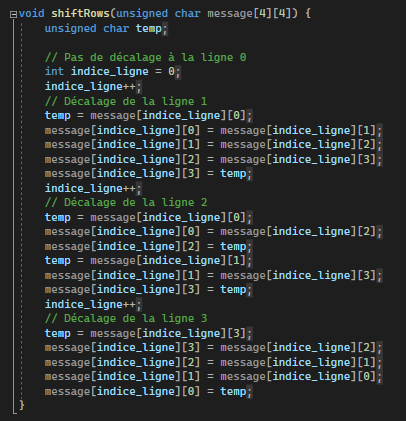
Le masque 0xF0 permet de récupérer la valeur hexadécimale de gauche qui après décalage à droite donnera la valeur dans l’hexadécimal de droite et on fait la même chose pour celui de droite qui sera le 2ème paramètre de la S\_Box. Puis on affiche le résultat :

Une image contenant texte

Description générée automatiquement

1. ShiftRows

On réalise ensuite l’étape de ShiftRows. Dans cette étape, on effectue une rotation de chaque ligne de la matrice 4x4, avec un décalage de 0 pour la première ligne, un décalage de 1 pour la seconde etc :



Une image contenant texte

Description générée automatiquement

1. MixColumns

On réalise ensuite l’étape de MixColumns. Dans cette étape, va reproduire le résultat général du produit matriciel entre l’entrée et et une matrice contenant les paramètres des Galois Fieds :

Une image contenant texte

Description générée automatiquement

Pour éviter de permet de l’information, nous devons ici tester après le premier XOR si le bit de poids fort est à 1 ou à 0 et lui appliquer un décalage avec sous sans filtre en fonction du cas, comme nous pouvons voir dans la boucle if du code précédent. On affiche le résultat :

Une image contenant texte

Description générée automatiquement

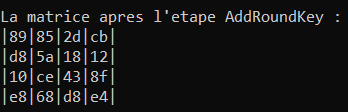
1. AddRoundKey

Pour finir, la dernière fonction est le AddRoundKey, c’est-à-dire qu’on regénère une nouvelle clé à partir du message et de la clé d’entrée comme ci-dessous

Une image contenant texte

Description générée automatiquement

On obtient donc la nouvelle clé key suivante :



**Voici donc les 4 fonctions quii composent un chiffrement AES. Je n’ai pas eu le temps de faire le Cipher qui va effectuer un addRoundKey dès le début puis exécute 9 fois les 4 fonctions présentées ci-dessous puis repasse par les fonctions ShiftRows, Subbytes et** AddRoundKey**.**

J’ai cependant également réalisé les fonctions de décodage qui sont simplament les fonctions inverses qui permettent de retrouver le message de départ. Les voici avec les résultats en sortie.

On repasse tout d’abord dans un AddRound Key pour revenir à la matrice précédent la première étape de AddRoundKey.

Une image contenant texte

Description générée automatiquement

La fonction MixColumns Inverse pourrait surement être optimisée. Cela donne le résultat suivant :

Une image contenant texte

Description générée automatiquement

Ensuite, on réalise un simple ShiftRows inverse, en décalant dans le sens inverse les valeur de chaque ligne.

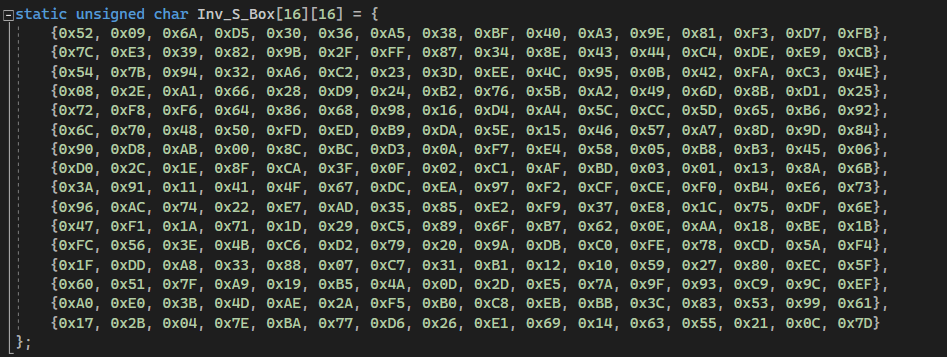
Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

Pour finir, il nous reste simplement à effectuer un SubBytes inverse. Pour cela, on va utiliser la S\_BoxInverse et remplacer chaque couple d’hexadécimal par sa correspondance dans les tableau.



Une image contenant texte

Description générée automatiquement

Et donc on retombe bien sur la matrice de départ :

Une image contenant texte

Description générée automatiquement