Fonctionnement d'AES

1 Présentation du système AES

Le système de chiffrement à **clé secrète AES** est un système basé sur le système Rijdndael construit par Joan Daemen et Vincent Rijmen.
Pour AES les blocs de données en entrée et en sortie sont des blocs de 128 bits, c'est à dire de 16 octets.
Les clés secrètes ont au choix suivant la version du système : 128 bits (16 octets), 192 bits (24 octets) ou 256 bits (32 octets).
On découpes les données et les clés en octets et **on les place dans des tableaux**. Les données comportent $t_d = 16$ octets $p_0, p_1, \cdots, p_{15}$ qui sont classés dans un tableau ayant 4 lignes et 4 colonnes. Le tableau est rempli colonnes par colonnes.
De même la clé est découpée en octets ($t_k = 16$, $t_k = 24$ ou $t_k = 32$ octets) $k_0, k_1, \cdots, k_{t_k-1}$. Ces octets sont aussi classés dans un tableau de 4 lignes et N_k colonnes ($N_k = 4$, $N_k = 6$ ou $N_k = 8$).

p_0	p_4	p_8	p_{12}
p_1	p_5	p_9	p_{13}
p_2	p_6	p_{10}	p_{14}
p_3	p_7	p_{11}	p_{15}

k_0	k_4	k_8	k_{12}	k_{16}	k_{20}	k_{24}	k_{28}
k_1	k_5	k_9	k_{13}	k_{17}	k_{21}	k_{25}	k_{29}
k_2	k_6	k_{10}	k_{14}	k_{18}	k_{22}	k_{26}	k_{30}
k_3	k_7	k_{11}	k_{15}	k_{19}	k_{23}	k_{27}	k_{31}

FIG. 1 – Données et clés (cas $N_k = 8$)

2 La structure générale

Le système AES effectue **plusieurs tours** d'une même composition de transformations.

2.1 Le nombre de tours

Suivant la version (la taille de la clé), ce nombre de tours noté n_r est différent. Le nombre n_r est donné dans le tableau suivant.

N_k	4	6	8
n_r	10	12	14

2.2 La clé de tour

À partir de la clé initiale K, le système crée $n_r + 1$ clés de tour ayant chacune 16 octets. Ces clés seront stockés dand un tableau unidimensionnel TK et seront notées

$$TK[0], TK[1], \cdots TK[n_r].$$

Nous verrons ultérierement comment sont calculées ces clés en fonction de la clé K du système.

2.3 Vue globale du fonctionnement

La procédure suivante décrit le fonctionnement global du système AES. Elle prend en entrée un tableau de données St (texte clair) qui est modifié par la procédure et renvoyé en sortie (texte chiffré).

Entrée : le tableau St et la clé K
Sortie : le tableau St modifié

$AES(St, K)$
début
$KeyExpansion(K, TK)$;
$AddRoundKey(St, TK[0]$;
pour $(i = 1 ; i < n_r; i++)$ $Round(St, TK[i])$;
$FinalRound(St, TK[n_r])$;
fin

Les procédures appelées $Round$ et $FinalRound$ sont elles-mêmes composées

Entrée : le tableau d'état St et une clé de tour T
Sortie : le tableau St modifié

$Round(St, T)$
début
$SubBytes(St)$;
$ShiftRows(St)$;
$MixColumns(St)$;
$AddRoundKey(St, T)$;
fin

Entrée : le tableau d'état St et une clé de tour T
Sortie : le tableau St modifié

$FinalRound(St, T)$
début
$SubBytes(St)$;
$ShiftRows(St)$;
$AddRoundKey(St, T)$;
fin

3 Les détails

3.1 la procédure SubBytes

Cette procédure est la seule transformation qui ne soit pas linéaire. C'est donc grâce à celle-ci que le système est résistant. Elle utilise une opération sur le corps fini à 256 éléments.

3.1.1 le corps fini à 256 éléments

Considérons le polynôme

$$P(X) = X^8 + X^4 + X^3 + X + 1.$$

Ce polynôme à coefficients dans le corps à 2 éléments $\mathbb{F}_2 = \{0, 1\}$ est irréductible sur ce corps.
Les éléments du corps à 256 éléments seront les octets

$$b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$$

considérés comme des polynômes

$$b(X) = b_7 X_7 + b_6 X^6 + b_5 X^5 + b_4 X^4 + b_3 X^3 + b_2 X^2 + b_1 X + b_0,$$

ce qui nous permet de définir les deux opérations suivantes :

addition

$$a_7a_6a_5a_4a_3a_2a_1a_0 + b_7b_6b_5b_4b_3b_2b_1b_0 = c_7c_6c_5c_4c_3c_2c_1c_0,$$

avec

$$c(X) = a(X) + b(X),$$

ce qui donne aussi

$$c_i = a_i \oplus b_i.$$

multiplication

$$a_7a_6a_5a_4a_3a_2a_1a_0 \times b_7b_6b_5b_4b_3b_2b_1b_0 = c_7c_6c_5c_4c_3c_2c_1c_0,$$

avec

$$c(X) = a(X) \times b(X) \mod P(X).$$

On a ainsi une structure de corps et donc tout élément non nul est inversible. Nous noterons g l'application de $\mathbb{F}_{256}$ dans $\mathbb{F}_{256}$ définie par

$$g(x) = \begin{cases} 0 \text{ si } x = 0 \\ x^{-1} \text{ sinon} \end{cases}$$

L'inverse d'un élément $b(X)$ se trouve par l'algorithme d'Euclide étendu.

3.2 La fonction affine f

Définissons $b = f(a)$ grâce à une matrice

$$\begin{pmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

La matrice carrée qui intervient est une matrice circulante, elle correspond donc à une multiplication de polynômes modulo $X^8 - 1$.

$$b(x) = \left((X^4 + X^3 + X^2 + X + 1) \times a(X) \mod (X^8 + 1)\right) + (X^6 + X^5 + X + 1).$$

On remarque que $g^{-1} = g$ et que f^{-1} est définie par

$$\begin{pmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

3.2.1 La procédure SubByte

On définit alors

$$s(a) = f\left(g(a)\right)$$

On a donc aussi

$$s^{-1}(b) = g\left(f^{-1}(b)\right).$$

La procédure $SubByte$ applique s à chaque octet de l'entrée St.

3.3 La procédure ShiftRows

La procédure consiste à opérer une rotation à gauche sur chaque ligne du tableau d'entrée. Le nombre de cases dont on décale la ligne i ($0 \leq i \leq 3$) est de i.
La transformation inverse est immédiate à calculer.

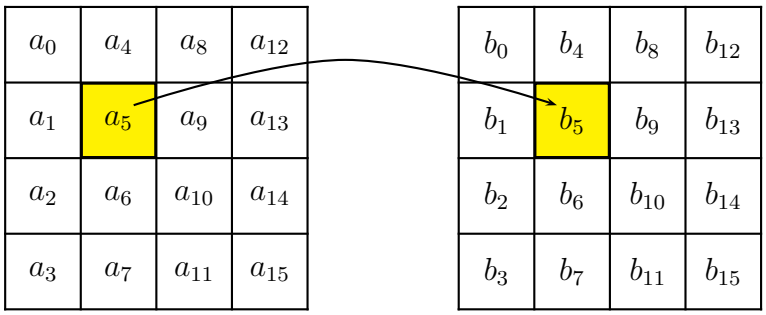

FIG. 2 – Transformation SubBytes

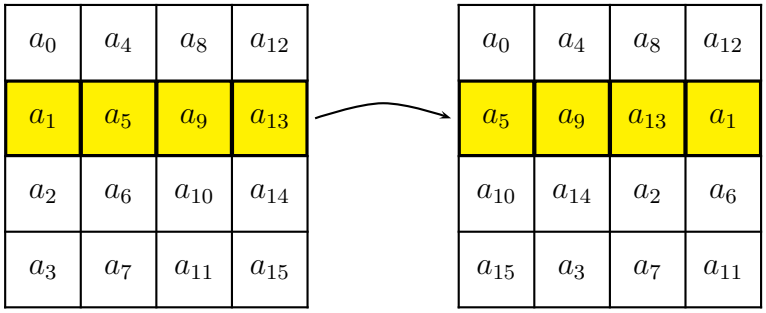

FIG. 3 – Transformation ShiftRows

3.4 La procédure MixColumns

La transformation $MixColums$ consiste à appliquer à chaque colonne du tableau des données une même transformation que nous allons décrire.
Considérons une colonne

$$c = (c_1, c_2, c_3, c_4)^t.$$

Les élément c_i sont des éléments de $\mathbb{F}_{265}$. Chaque colonne c est transformée en une colonne d grâce à la transformation linéaire suivante donnée par sa matrice dont les coefficients sont dans $\mathbb{F}_{256}$ et que nous écrivons comme des octets en hexadécimal :

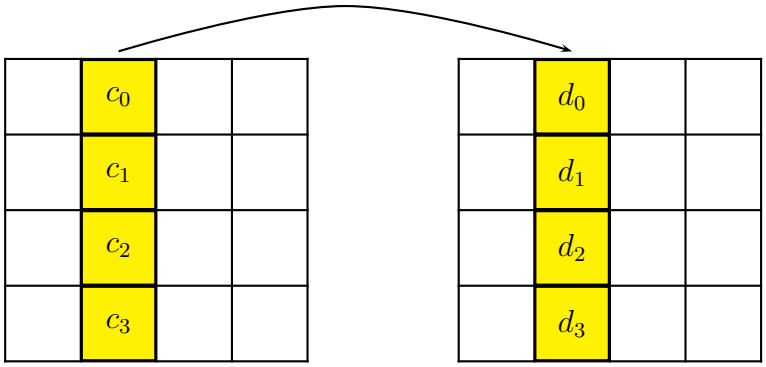

FIG. 4 – Transformation MixColums

$$\begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \times \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

Là encore la matrice utilisée est circulante. La transformation correspond en fait à une multiplication par un polynôme fixe

$$A(X) = 03.X^3 + 01.X^2 + 01.X + 02,$$

modulo $1 + X^4$:

$$d(X) = A(X) \times c(x) \mod X^4 + 1$$

où

$$c(X) = c_0 + c_1X + c_2X^2 + c_3X^3,$$

et

$$d(X) = d_0 + d_1X + d_2X^2 + d_3X^3.$$

Le polynôme $A(X)$ est premier avec $X^4 + 1$, il est donc inversible modulo $X^4 + 1$ est son inverse est

$$B(X) = 0B.X^3 + 0D.X^2 + 09.X + 0E.$$

On retrouve donc $c(X)$ à partir de $d(X)$ en effectuant le produit

$$c(X) = B(X) \times d(X) \mod X^4 + 1,$$

ou en effectuant le produit matriciel

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \times \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{pmatrix}.$$

3.5 La procédure AddRoundKey

La procédure AddRoundKey est très simple. Elle consiste à faire un ou exclusif entre les 128 bits de l'état St et les 128 bits de la clé de tour T. On obtient une nouvelle valeur de l'état.

$$St := St \oplus T.$$

3.6 La procedure KeyExpansion

La clé de chiffrement K stockée dans un tableau de 4 lignes et N_k colonnes ($N_k = 4, 6, 8$) est étendue en un tableau W ayant 4 lignes et $4 * n_r + 1$ colonnes. La clé de tour $TK[i]$ ($0 \leq i \leq n_r$) est donnée par les 4 colonnes $4 * i, 4 * i + 1, 4 * i + 2, 4 * i + 3$ du tableau W.
Il y a deux façons de construire le tableau W suivant que $N_k = 4, 6$ ou $N_k = 8$. La procédure de construction est nommée $ExpandedKey$.

1) Cas $N_k = 4$ ou $N_k = 6$.

Entrée : la clé K (sous forme de tableau)
Sortie : le tableau W

$ExpandedKey(K, W)$
début
pour $(j = 0; j < N_k; j++)$
  pour $(i = 0; i < 4; i++)$ $W[i, j] = K[i, j]$;
pour $(j = N_k; j < 4(n_r + 1); j++)$
  si $(j \mod N_k == 0)$
  alors
    $W[0, j] = W[0, j - N_k] \oplus s\,(W[1, j - 1]) \oplus RC[j/N_k]$;
    pour $(i = 1; i < 4; i++)$
      $W[i, j] = W[i, j - N_k] \oplus s\,(W[i + 1 \mod 4, j - 1])$;
  sinon
    pour $(i = 0; i < 4; i++)$
      $W[i, j] = W[i, j - N_k] \oplus W[i, j - 1]$;
  finsi ;
fin

La procédure utilise la fonction s sur les octets définie précédemment. Elle utilise aussi des constantes de $\mathbb{F}_{256}$, données par

$$RC[i] = \alpha^i,$$

où α est l'élément de $\mathbb{F}_{256}$ correspondant au polynôme X ($\alpha = 02$). L'élévation à la puissance i se fait dans le corps $\mathbb{F}_{256}$.

2) Cas $N_k = 8$.

Entrée : la clé K (sous forme de tableau)
Sortie : le tableau W

$ExpandedKey(K, W)$
début
pour $(j = 0; j < N_k; j++)$
  pour $(i = 0; i < 4; i++)$ $W[i, j] = K[i, j]$;
pour $(j = N_k; j < 4(n_r + 1); j++)$
  si $(j \mod N_k == 0)$
  alors
    $W[0, j] = W[0, j - N_k] \oplus s\,(W[1, j - 1]) \oplus RC[j/N_k]$;
    pour $(i = 1; i < 4; i++)$
      $W[i, j] = W[i, j - N_k] \oplus s\,(W[i + 1 \mod 4, j - 1])$;
  sinon si $(j \mod N_k == 4)$
    pour $(i = 0; i < 4; i++)$
      $W[i, j] = W[i, j - N_k] \oplus s\,(W[i, j - 1])$;
  sinon
    pour $(i = 0; i < 4; i++)$
      $W[i, j] = W[i, j - N_k] \oplus W[i, j - 1]$;
  finsi ;
fin