Mode Autonome Secure blocK (ASK) pour la Cryptographie à clé Secrète: Nouveau Mode Opératoire

Mohamed Sadek ALI-PACHA*– Naima HADJ-SAID – Adda ALI-PACHA – Aek HAOUAS

Laboratoire **SIMPA** (Signal-Image-Parole)

Université des *Sciences et de la Technologie d'Oran USTO,*
BP 1505 El M'Naouer Oran 31036 ALGERIE

* *Corresponding Author, Email: a.alipacha@gmail.com*

Résumé:

Un **mode opératoire** consiste en la description détaillée des actions nécessaires à l'obtention d'un résultat. Il décrit généralement le déroulement détaillé des opérations effectuées sur un poste fixe, mais il peut également décrire l'enchaînement des opérations de poste à poste. Un mode opératoire décrivant les enchaînements opératoires de poste à poste permet de définir :

- l'ensemble des postes de travail concernés par la réalisation d'un produit, d'une pièce élémentaire,
- les temps de passage prévus (alloués) à chaque poste,
- l'ordre logique d'intervention de chaque poste (machine, ou poste manuel),
- les conditions d'enchaînement, de déclenchement, des opérations successives,
- les moyens de transfert de poste à poste.

En cryptographie, un **mode d'opération** est la manière de traiter les blocs de texte clairs et chiffrés au sein d'un algorithme de chiffrement par bloc ou bien c'est la présentation d'une méthode de chaînage des blocs dans un chiffrement par blocs. Plusieurs modes existent possédant leur propre atout, certains sont plus vulnérables que d'autres et certains modes combinent les concepts d'authentification et sécurité.

Dans ce travail on essaie de faire la synthèse des modes opératoires de la cryptographie en bloc et de proposer une ***alternative pour faire le bon choix du*** *vecteur d'initialisation de certains de ces modes,* avec la suggestion d'un nouveau mode opératoire qu'on l'a baptisé mode **A**utonome **S**ecure bloc**K** (ASK).

Mots Clés : Modes opératoires, Data Encryption Standard, AES, Carte logistique, Ali-Pacha Generator.

1. Introduction :

La cryptographie à algorithmes symétriques [1] utilise la même clé pour les processus de codage et de décodage; cette clé est le plus souvent appelée "secrète" (en opposition à "privée") car toute la sécurité de l'ensemble est directement liée au fait que cette clé n'est connue que par l'expéditeur et le destinataire. Ce type de cryptographie fonctionne habituellement suivant deux procédés différents [1], le cryptage par blocs (DES, AES, Skipjack...); et le cryptage en continu "stream" (algorithme RC4) et font souvent appel à des algorithmes de génération de nombres aléatoires nécessitant de l'aléa.

Une bonne distribution statistique ne suffit pas en cryptographie. L'aléa cryptographique doit également être **imprédictible**.

Les utilisations de nombres aléatoires en cryptographie sont diverses [2]: génération de **clés**, génération de **vecteurs d'initialisation** en cryptographie symétrique, **randomisation** en cryptographie asymétrique. Des tests statistiques (FIPS 140) permettent de vérifier la bonne distribution statistique de l'aléa généré, mais aucun des tests existants ne garantit l'imprévisibilité d'une source de nombres aléatoires. Le caractère imprédictible de l'aléa cryptographique ne s'étudie pas sur la suite de bits générée mais sur l'algorithme de génération. D'une manière générale, il est recommandé de **retraiter** l'aléa issu d'une source aléatoire non maîtrisée pour en garantir l'imprédictibilité.

Les mécanismes cryptographiques présentés précédemment agissent sur des données de format bien défini. La définition des algorithmes de chiffrement par bloc induit une manière naturelle de chiffrer des blocs de taille fixe. Il existe cependant différentes manières d'utiliser ces primitives pour chiffrer des messages de taille quelconque, appelées **modes opératoires**.

Historiquement, les modes d'opération ont été abondamment étudiés pour leurs propriétés de propagation d'erreurs lors de divers scénarios de modification de données durant le chiffrement. Le mode le plus naturel consiste à découper le message en blocs de **n** bits, et à appliquer la fonction de chiffrement à chacun de ces blocs avec la clé de chiffrement **K**. Ce mode est appelé ECB (Electronic CodeBook). Il ne doit pas être utilisé, car les messages chiffrés obtenus laissent fuir de l'information sur les messages clairs correspondants.

En effet, deux blocs de clair identiques conduisent à deux blocs de chiffré identiques. Il y'a autres modes par bloc qui sont utilisés comme : Cipher Block

Chaining (CBC), Cipher FeedBack (CFB) ou Output FeedBack (OFB).

Dans cet article on essaie de faire la synthèse sur ces modes opératoires avec la suggestion d'un nouveau mode opératoire et de proposer une alternative pour faire le bon choix du vecteur d'initialisation de certains de ces modes.

2. Modes de chiffrement par blocs

Avant de s'intéresser à la construction de l'algorithme de chiffrement par blocs lui-même [3], il est utile de préciser qu'il existe plusieurs modes qui permettent d'enchaîner le chiffrement des différents blocs de taille n, m_i pour i variant de 0 à t−1, la fonction de chiffrement E_K s'appliquant alors à chacun des blocs. Il s'agit donc de chaîner les $c_i=E_K(m_i)$ avec en général m_{i+1} pour i variant de 0 à t−1. Où K désigne la clé utilisée par l'algorithme, E désigne le cryptage en lui-même, M (ou m, m_i) désigne le message en clair (c'est-à-dire un bloc) et C (ou c, c_i) le chiffré résultant.

Les quartes modes cités précédemment sont plus ou moins indépendants de l'algorithme choisi. Toutefois, tous les algorithmes ne permettent pas d'utiliser tout les modes possibles. Il existe des variantes du mode CBC. On peut citer : le mode CFB de l'anglais Cipher Feedback Block qui utilise un chiffrement à rétroaction, le mode OFB de l'anglais Output Feedback Block qui utilise un chiffrement à rétroaction de sortie. Pour mieux comprendre, voyons ces modes plus en détails.

2.1 Electronic CodeBook (ECB)

Ce mode, dictionnaire de codes, est le plus simple des modes. Il revient à crypter un bloc indépendamment des autres; cela permet entre autre de crypter suivant un ordre aléatoire (bases de données, etc...) mais en contre-partie, ce mode est très vulnérable aux attaques. On peut aussi l'utiliser pour pipeliner du hardware.

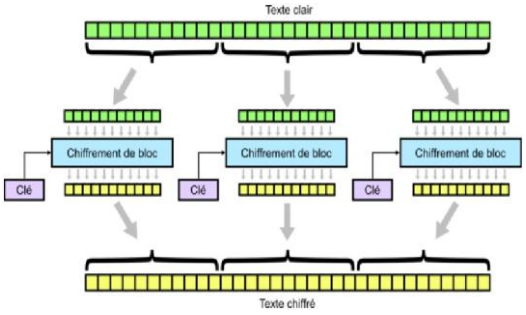


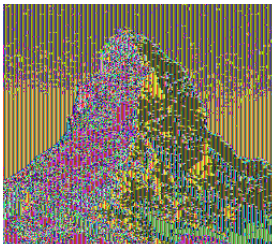
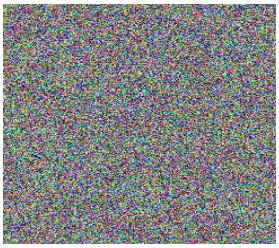
Figure 1: Mode ECB.

Il est par exemple possible de recenser tous les cryptés possibles (code books) puis par recoupements et analyses statistiques recomposer une partie du message original sans avoir tenté de casser la clé de chiffrement.

La vulnérabilité est encore plus flagrante sur une image. En effet, les images sont constituées de nombreuses redondances qui font que des blocs sont chiffrés de la même manière en mode ECB. Dans l'exemple ci-dessous, le chiffrement en ECB est réalisé sur des blocs de 4 pixels.

Image originale

On distingue très nettement les formes du Cervin ainsi que les séparations entre les blocs. Avec un mode plus sûr comme CBC ou CTR, l'image a un contenu aléatoire dont on ne peut tirer aucune information a priori.

Chiffré en mode ECB Chiffré en mode CBC

Cela ne veut toutefois pas dire que le chiffrement est sûr, des failles importantes peuvent également apparaître dans des schémas qui produisent des sorties aléatoires mais elles ne sont pas nécessairement liées au mode d'opération.

ECB a d'autres effets négatifs sur l'intégrité et la protection des données. Ce mode est sensible à des « attaques par répétition » : elles consistent à réinjecter dans le système des données identiques à celles interceptées auparavant. Le but est de modifier le comportement du système ou répéter des actions. Ce mode est pour ces raisons fortement déconseillé dans toute application cryptographique. Le seul avantage qu'il peut procurer est un accès rapide à une zone quelconque du texte chiffré et la possibilité de déchiffrer une partie seulement des données.

2.2 Cipher Block Chaining (CBC)

Un autre mode de chiffrement très employé est le mode CBC (enchaînement des blocs).

Il consiste à chiffrer le bloc i préalablement combiné par ou exclusif avec le chiffré du bloc précédent ainsi, $c_i = E_K(m_i \oplus c_{i-1})$ pour tout i de 1 à t, avec $c_0 = E_K(m_0 \oplus IV)$ où IV désigne un vecteur d'initialisation. C'est un bloc de données aléatoires qui permet de commencer le chiffrement du premier bloc et qui

fournit ainsi une forme de hasard indépendant du document à chiffrer. Il n'a pas besoin d'être lui-même chiffré lors de la transmission, mais il ne doit jamais être réemployé avec la même clé.

Dans ce mode, on applique sur chaque block un 'OU exclusif' avec le chiffrement du bloc précédent avant qu'il soit lui-même chiffré. De plus, afin de rendre chaque message unique, un vecteur d'initialisation (IV) est utilisé. Ce vecteur d'initialisation change à chaque session, et doit être transmis au destinataire. Par contre, il n'est pas nécessaire de le chiffrer avant de l'envoyer : il peut être connu de l'adversaire. Le schéma de base sera le suivant :

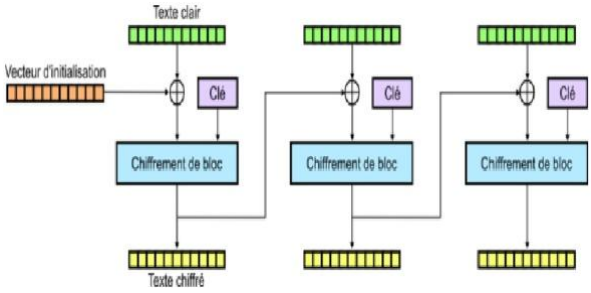


Figure 2: Mode CBC.

Le mode CBC introduit une dépendance entre deux cycles de cryptage : le chiffré obtenu au rang i-1 est utilisé pour obtenir le chiffré du rang i. Concrètement, ce chiffré c_{i-1} subit un XOR avec le bloc m_i. C'est d'ailleurs le mode le plus couramment utilisé. Il permet d'introduire une complexité supplémentaire dans le processus de cryptage en créant une dépendance entre les blocs successifs; autrement dit, le cryptage d'un bloc va être « d'une manière ou d'une autre » lié à ou aux blocs/chiffrés précédents.

2.3. Cipher Feedback (CFB)

Ce mode, chiffrement à rétroaction, et les suivants modes agissent comme un chiffrement par flux. Ils génèrent un flux de clés qui est ensuite appliqué au document original.

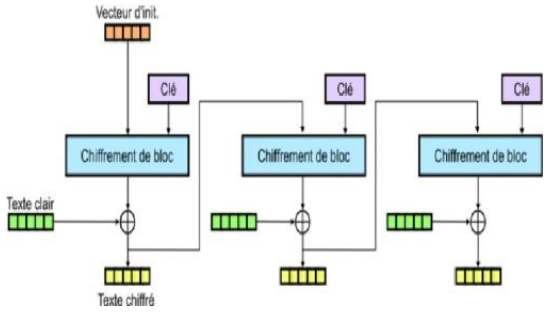


Figure 3: Mode CFB.

Ce mode est destiné aux blocks ciphers dans le but d'en autoriser une utilisation plus souple, qui s'apparente plus à celle des algorithmes en continu. On peut le considérer comme un intermédiaire entre les deux modes précédemment définis.

En effet, en partant d'un algorithme en bloc utilisant une longueur standard de n bits/blocs, le mode CFB va permettre de crypter des blocs dont la longueur pourra varier de n à 1 bits/blocs. Sachant que dans ce dernier cas, il serait plus économique en calculs d'utiliser directement un algorithme en continu. Quant au cas où la longueur est celle de l'algorithme (à savoir n), le schéma de CFB se simplifie et ressemble quelque peu à celui de CBC (à quelques nuances près).

Dans ce mode, le flux de clé est obtenu en chiffrant le précédent bloc chiffré. CFB est un chiffrement par flot. Son grand intérêt est qu'il ne nécessite que la fonction de chiffrement, ce qui le rend moins cher à câbler ou programmer pour les algorithmes ayant une fonction de chiffrement différente de la fonction de déchiffrement (exemple: AES).

2.4. Output Feedback (OFB)

Ce mode, chiffrement à rétroaction de sortie, est une variante de mode CFB précédemment abordé. Il est d'ailleurs parfois appelé internal feedback. Il présente beaucoup de problèmes de sécurité et il est peu conseillé sauf dans le cas où sa longueur est égale à celle de l'algorithme utilisé. Dans ce mode, le flux de clé est obtenu en chiffrant le précédent flux de clé.

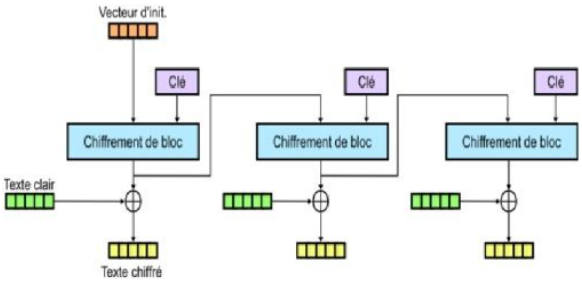


Figure 4: Mode CFB.

C'est un mode de chiffrement de flot qui possède les mêmes avantages que CFB. De plus, il est possible de le pré-calculer en chiffrant successivement le vecteur d'initialisation. Il n'est donc sûr que si la fonction de chiffrement alliée à la clé forme une bonne suite pseudo-aléatoire.

Ce mode est très fragile vis-à-vis d'une attaque au clair. En effet, à l'unique condition de connaitre le vecteur d'initialisation d'un message chiffré et de connaitre le clair d'un autre message chiffré, l'attaquant peut reconstituer aisément la chaine ayant chiffré le premier message et donc déchiffrer ce dernier. Cette fragilité se retrouve dans le mode CFB, à ceci près que seul le premier bloc du message peut être reconstitué de cette manière, l'attaquant a besoin de déchiffrer le message bloc à bloc, en fournissant à chaque fois l'ensemble des blocs précédents, de manière à récupérer la chaine ayant chiffré le bloc suivant (attaque au clair choisi).

2.5. CounTeR (CTR)

Dans le cas d'un chiffrement basé sur un compteur, c'est la valeur d'un compteur qui est chiffrée et qui produit un bloc pseudo-aléatoire qui sera ensuite xorer avec les blocs de message clair pour produire le chiffré correspondant : cette méthode correspond au célèbre "one-time-pad" et est illustrée à la figure 5.

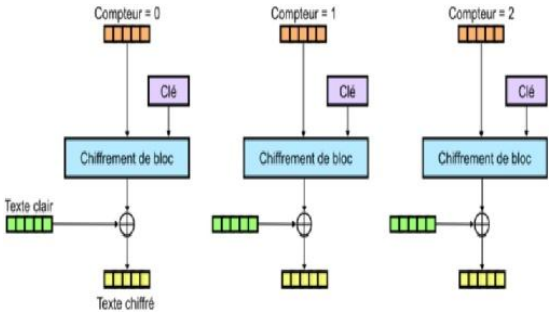


Figure 5: Mode CTR.

Dans ce mode, le flux de clé est obtenu en chiffrant les valeurs successives d'un compteur. Ce mode combine de nombreux avantages, car il permet le chiffrement par flot, est précalculable, permet un accès aléatoire aux données, est parallélisable et n'utilise que la fonction de chiffrement. Le compteur utilisé peut être une suite pseudo-aléatoire qu'il sera facile de retrouver à partir de la graine (vecteur d'initialisation), aussi la périodicité du compteur peut être un inconvénient.

Ce mode peut également être utilisé avec un vecteur d'initialisation qui, dans ce cas, est concaténé à la valeur du compteur.

2.6 CipherText Stealing (CTS)

Dans le mode du chiffrement avec vol de texte, applicable à un chiffrement par blocs (ECB, CBC, etc.), les deux derniers blocs sont partiellement combinés de façon à obtenir un message de même taille. Ici, exemple de CTS opérant sur un chiffrement en mode CBC.

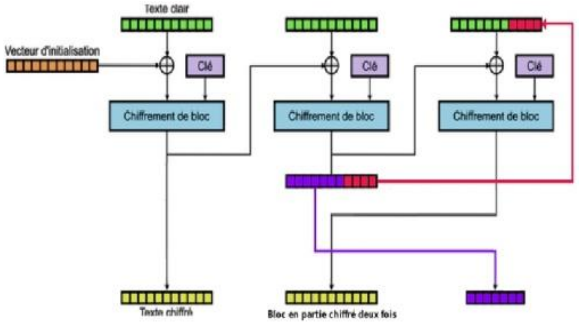


Figure 6: Mode CTS.

Les deux derniers blocs sont échangés et combinés en partie, ce qui nécessitera de les obtenir tous les deux pour en décrypter un. CTS n'est pas un mode de chiffrement par flot, mais permet d'éviter l'utilisation de bourrage dans les chiffrements par blocs, et donne une taille de message crypté égale à la taille du message clair. Il est très utilisé dans les protocoles ou formats ne supportant pas une taille quelconque.

3. Système Combiné (Sécurité + Authentification)

Aucun des modes ci-dessus ne protège l'intégrité du message c'est pourquoi on propose un système combiné entre sécurité et authentification. Il est généralement bien compris que lorsque des données sont chiffrées, il est presque toujours essentiel de fournir un tel mécanisme, car en son absence, les risques sont grands. Pour ce besoin, on peut utiliser un code d'authentification (HMAC) qui protègera le message chiffré et le vecteur d'initialisation.

A défaut, des modes d'opération sont conçus des algorithmes MAC spécifiques combinant sécurité et authentification, tel que, par exemple XCBC, IACBC, IAPM, OCB, EAX, CWC ou CCM.

3.1. HMAC :

Un **HMAC** [4, 5], de l'anglais *keyed-hash message authentication code* (code d'authentification d'une empreinte cryptographique de message avec clé), est un type de code d'authentification de message (CAM), et est un code accompagnant des données dans le but d'assurer l'intégrité de ces dernières, en permettant de vérifier qu'elles n'ont subi aucune modification, après une transmission par exemple.

Le concept est relativement semblable aux fonctions de hachage. Il s'agit ici aussi d'algorithmes qui créent un petit bloc authentificateur de taille fixe. La grande différence est que ce bloc authentificateur ne se base plus uniquement sur le message, mais également sur une clé secrète.

Tout comme les fonctions de hachage, les MAC n'ont pas besoin d'être réversibles. En effet, le récepteur exécutera le même calcul sur le message et le comparera avec le MAC reçu.

Le MAC assure non seulement une fonction de vérification de l'intégrité du message, comme le permettrait une simple fonction de hachage mais de plus authentifie l'expéditeur, détenteur de la clé secrète. Il peut également être employé comme un chiffrement supplémentaire (rare) et peut être calculé avant ou après le chiffrement principal, bien qu'il soit généralement conseillé de le faire avant.

La fonction HMAC est définie comme suit :

$$\mathrm{HMAC}_K(m) = h\Big((K \oplus opad) \,||\, h\big((K \oplus ipad) \,||\, m\big)\Big)$$

avec :

- h : une fonction de hachage itérative,
- K : la clé secrète complétée avec des zéros pour qu'elle atteigne la taille de bloc de la fonction h
- m : le message à authentifier,

- "||" désigne une concaténation et " $\oplus$ " un ou exclusif,
- *ipad* et *opad*, chacune de la taille d'un bloc, sont définies par : *ipad* = 0x363636...3636 et *opad* = 0x5c5c5c...5c5c. Donc, si la taille de bloc de la fonction de hachage est 512, *ipad* et *opad* sont 64 répétitions des octets, respectivement, 0x36 et 0x5c.

3.2. Compteur avec CBC-MAC,

CBC-MAC est l'un des algorithmes de MAC. Il est basé sur un chiffrement par bloc utilisé selon un mode d'opération CBC (*cipher block chaining*). Ce principe a été formulé en 1985 dans un standard du NIST (FIPS PUB 113, *Standard on Computer Data Authentication*).

Pour chiffrer, on découpe les données en blocs de taille adéquate (selon le chiffrement par bloc utilisé, au minimum un chiffrement par bloc de 32 bits). Les blocs sont chiffrés les uns après les autres, le résultat chiffré du bloc précédent est transmis au bloc suivant.

Soit $E_k(M_i)$ l'opération de chiffrement sur un bloc de données M_i avec la clé K.

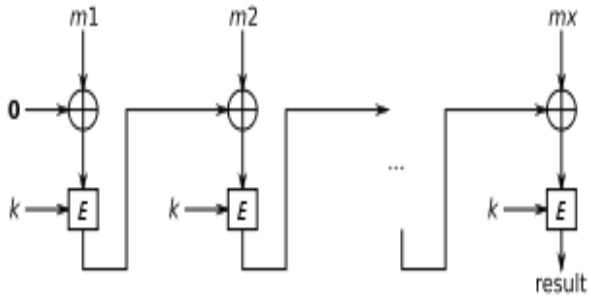


Figure 7: Schéma du CBC-MAC

Le chiffrement se fait comme suit :

1. on découpe les données en blocs de taille fixe M_0, M_1, …, M_{n-1} avec un remplissage selon la norme PKCS #7 pour le dernier bloc
2. on définit un vecteur d'initialisation $C_0 = 0$.
3. on traite les blocs au fur et à mesure : $C_{i+1} = E_k(C_i \oplus M_i)$

Le code d'authentification correspond à une partie du dernier bloc chiffré C_{n-1} (extraction de 16 à 64 bits dans le standard du NIST).

3.3 CCMP : Counter-Mode/CBC-Mac protocol

CCMP (*Counter-Mode/CBC-Mac protocol*) est une méthode de chiffrement définie dans le standard IEEE 802.11i. CCMP gère les clés et l'intégrité des messages. Le protocole utilise le chiffrement par bloc AES dans un mode d'opération de type "compteur" couplé à code d'authentification MAC (CBC-MAC). Le compteur sert à assurer un chiffrement sûr en évitant d'avoir un vecteur d'initialisation identique pour chaque message alors que le code d'authentification permet de vérifier que le message n'a pas été altéré.

La spécification de CCM prévoit un chiffrement utilisant des blocs de 128 bits. Deux paramètres doivent être définis :

- M, soit la taille du champ destiné à recevoir le code d'authentification. Des multiples de 2 octets (entre 4 et 16 octets) sont les valeurs autorisées. Le choix de la longueur dépend de la résistance désirée contre les attaques et la longueur du message final.
- L, la longueur du champ destiné à recevoir la taille du message. La valeur est comprise entre 2 et 8 octets et doit être adaptée en fonction de M.

Le compteur (*Packet Number*) passe à 48 bits. L'espace du vecteur d'initialisation est donc considérablement augmenté. Des clés temporaires de 128 bits sont extraites automatiquement de la clé principale et d'autres valeurs.

3.4 SAFER - Secure And Fast Encryption Routine

SAFER (**Secure And Fast Encryption Routine**) [6] est le nom d'une famille d'algorithmes de chiffrement par bloc conçus principalement par James Massey pour la société *Cylink Corporation en 1993* pour le ministère des affaires intérieures de Singapour. La première version, *SAFER K* et la révision *SAFER SK* partagent les mêmes principes pour le chiffrement mais le nombre de tours et le key schedule (préparation des clés) ont été renforcés.

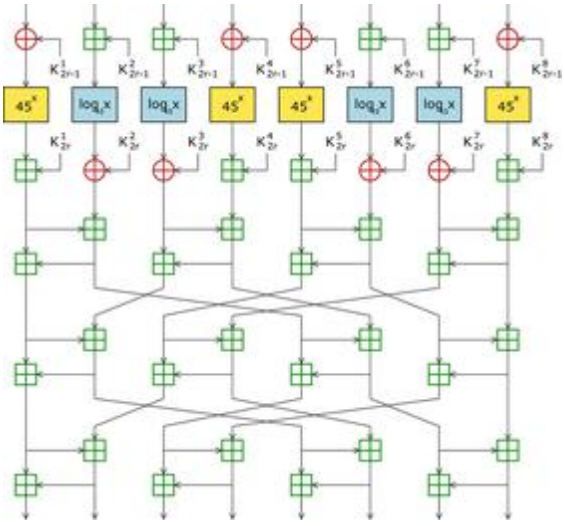


Figure 8: Schéma SAFER -

Tous ces chiffrements sont basés sur la même architecture de tour.

- **Nombre de tours :** 6 (64 bits) ou 10 (128 bits) pour SAFER-K (8 et 12 tours pour SAFER-SK).

Chaque tour est divisé en quatre stages :

- mixage de clé
- couche de substitution
- 2e mixage de clé
- couche de diffusion

A la première étape, le bloc en clair est divisé en 8 segments de 8 bits et les sous-clés issues du key schedule sont ajoutées via une addition modulo 256 ou un XOR. La couche de substitution consiste en deux S-Boxes, l'une étant l'inverse de l'autre. Leur contenu provient de l'exponentiation discrète de 45^x et du logarithme $x \cdot \log(45)$. Après un deuxième mixage, le stage de diffusion utilise une pseudo-transformation de Hadamard.

Ces algorithmes n'ont pas fait l'objet de brevets et peuvent être utilisés sans restriction.

3.5 Autres Modes Opératoires

1) OU exclusif-Chiffrement-OU exclusif : « Xor-Encrypt-Xor » (XEX) : Le chiffrement OUX est un chiffrement bit à bit [3] qui utilise les propriétés mathématiques de la fonction OU exclusif notamment cette égalité *(a⊕b)⊕b =a; a* sera le texte à chiffrer et *b* sera la clé de chiffrement.
2) « Tweaked CodeBook mode » (TCB) [7]

3) Liskov, Rivest, Wagner (LRW): LRW est utilisé par les logiciels [8] FreeOTFE, Bestcrypt et dm-crypt.

4) Xex-Tcb-Cts (XTS) : XTS est utilisé par [8] FreeOTFE, Bestcrypt, dm-crypt et Truecrypt.

4. Vecteur d'Initialisation : Solution Proposée

En cryptographie, un **vecteur d'initialisation** (en anglais *initialization vector* ou *IV*) est un bloc de bits combiné avec le premier bloc de données lors d'une opération de chiffrement. Il est utilisé dans le cadre des modes d'opération d'un algorithme de chiffrement symétrique par blocs ou pour un chiffrement par flux comme RC4. Dans certains crypto systèmes, le vecteur est généré de manière aléatoire puis transmis en clair avec le reste du message.

Dans ce qui suit, on va proposer une méthode de génération de ces vecteurs d'initialisations de façon à satisfaire la rondomization du vecteur et la possibilité de l'envoyer crypté.

4.1 Carte Logistique :

La carte logistique [9] est l'une des dynamiques très connues dans la théorie des systèmes non-linéaires et qui définit par l'équation (1) :

$$y_{k+1} = r.x_k(1 - x_k) \qquad (1)$$

Elle nous donne une explication parfaite pour un comportement d'un système dynamique. Ce système était utilisé en 1976 par le biologiste Robert May pour l'étude d'évolution de population des insectes ou :

- y_{k+1} : La génération à la venir qui est proportionnel a x_k.
- x_k : La génération précédente.
- r : Constante positive incorpore tous les facteurs reliés au reproductif, succès à la survie hivernale des œufs par exemple, etc.

Afin d'étudier ce système dynamique ainsi que certains modes asymptotiques particuliers, la première chose à faire est de tracer le graphe parabolique $y = r.x(1 - x)$, et le diagonale $y = x$.

L'opération que nous allons suivre pour tracer la forme itérative y_{k+1} en fonction x_k se résume simplement comme suite :

- À partir d'une valeur initiale x_0 de l'axe des abscisses, nous rejoignons la fonction par une verticale; la fonction prend la valeur $y_1=r.x_0(1-x_0)$,
- Par l'horizontale $y_1= r.x_0 (1-x_0)$ issue du point précédent, nous rejoignons la droite $y = x$;
- Nous représentons l'abscisse de ce point d'intersection par la droite verticale $x=x_0$; nous avons bien $y_1 = x_1$
- À partir de la valeur x_1 de l'axe des abscisses, nous rejoignons la fonction par une verticale; la fonction prend la valeur $y_2 = r.x_1(1 - x_1)$; et ainsi de suite.

On prenant $r = 3.9$ et $x_0 = 0.01$ pour la carte logistique, les opérations précédentes se retrouvent pour 60 itérations sur les graphiques de la figure 9.

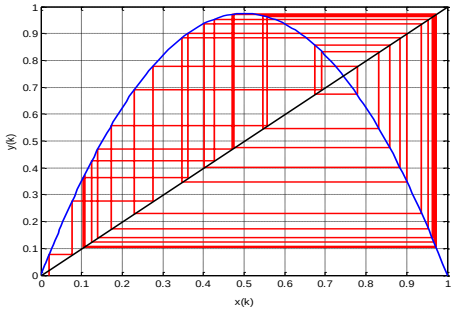


Figure 9a : Évolution de y_k en fonction de x_k

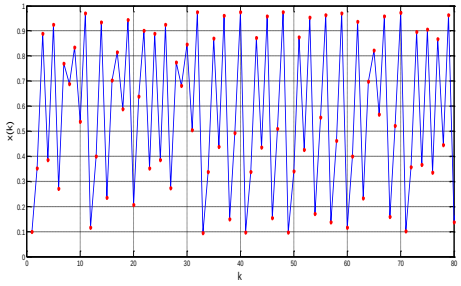


Figure 9b : Évolution de x(k) en fonction de k

Le système est purement chaotique et est très sensible aux petits changements sur la valeur de la condition initiale.

On peut noter, que les données qu'on va élire sont aléatoires, donc idéales pour un chiffrement parfait. On représente ces valeurs réelles avec la représentation des nombres à **virgule flottante** en utilisant la simple précision de la norme IEEE 754, c'est-à-dire (32 bits : 1 bit de signe, 8 bits d'exposant (-126 à 127), 23 bits de mantisse, avec bit 1 implicite).

Aussi on note, sachant la définition du chaos, que ce flux chaotique de données aléatoires a les caractéristiques suivantes :

1. Longue période,
2. Pas de répétitions,
3. Complexité linéaire locale,
4. Critères de non linéarité pour des fonctions booléennes.

4.2 Générateur de Vecteur d'Initialisation:

On va maintenant proposer un générateur des vecteurs d'initialisations pour générer les VI (figure 10).

Sur la base du générateur Ali-Pacha [10] en utilisant la carte logistique comme attracteur et ou les données chaotiques sont présentées sous la norme IEEE754.

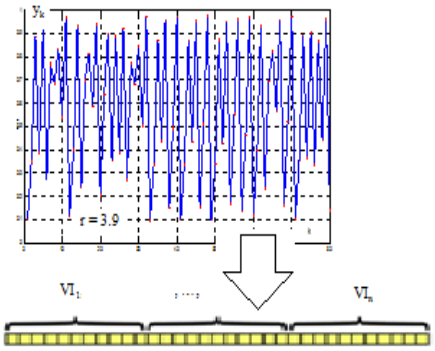


Figure 10: Générateur de Vecteur d'Initialisation

Au lieu de transmettre le vecteur initial en claire avec la clé, il y'a lieu d'augmenter le champ de la clé de chiffrement en une super clé de chiffrement comme il montre le tableau qui suit et transmettre cette super clé au destinataire par les procédés usuels :

Super Clé de Chiffrement			
Vecteur Initial			Clé de Chiffrement
r	t	X_0	K

Une fois le destinataire reçoit cette super clé de déchiffrement, il procède au calcul du vecteur initial et au déchiffrement du message crypté.

Il faut noter, que si la valeur de t est autre que 1 (sachant que t est toujours entier), les valeurs pseudo chaotiques de la carte logistique qui seront prises pour la génération des vecteurs initiaux sont les multiples de t, i.e. x_0, x_t, x_{2t}, …, x_{nt}.

4.3 Suggestion d'un Nouveau Mode Opératoire

De ce que nous avons vu auparavant, nous allons proposer un nouveau mode s'inspirant des trois modes opératoires suivants : CounTeR (CTR), Electronic CodeBook (ECB) et Cipher Block Chaining (CBC).

Dans ce mode, le message à chiffrer est subdivisé en plusieurs blocs qui sont chiffrés séparément les uns après les autres cela permet entre autre de les crypter, si on désire, suivant un ordre aléatoire (table de permutation au début de chiffrement).

Aussi, on recommande l'utilisation de générateur des vecteurs initiaux afin d'assurer le changement à chaque bloc/session du vecteur d'initialisation.

Le graphe de la figure 11 nous décrit la méthode de chaînage des blocs dans ce mode qui sera baptisé par : le Mode Autonome Secure blocK (ASK).

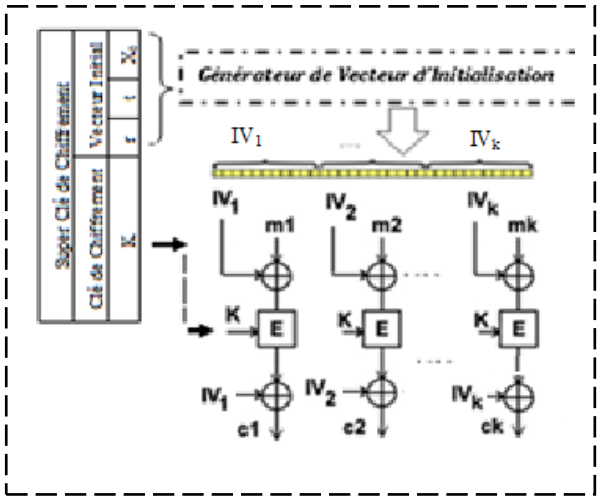


Figure 11: Autonome Secure blocK (ASK)

- Il s'agit du mode simple.
- Deux blocs avec le même contenu seront chiffrés des façons différentes.
- Vecteur d'initialisation change à chaque bloc/session.
- Il procure est un accès rapide à une zone quelconque du texte chiffré et la possibilité de déchiffrer une partie seulement des données.
- Il revient à crypter un bloc indépendamment des autres, c'est-à-dire c'est un mode parallélisable et on peut imaginer un

dispositif hardware ou le cryptage est pipeline et en parallèle.

- Permet entre-autre de crypter suivant un ordre aléatoire (table de permutation secrète au début de chiffrement.

5. Conclusion :

Nous avons présentés et fait une synthèse sur les méthodes de chaînage des blocs dans un chiffrement par bloc et par le chiffrement par flot.

La principale différence entre le chiffrement par bloc (en anglais block cipher) et le chiffrement par flot vient du découpage des données en blocs de taille généralement fixe (souvent une puissance de deux comprise entre 32 et 512 bits). Les blocs sont ensuite chiffrés les uns après les autres. Il est possible de transformer un chiffrement par bloc en un chiffrement par flot en utilisant un mode d'opération comme ECB (chaque bloc chiffré indépendamment des autres) ou CFB (on chaîne le chiffrement en effectuant un XOR entre les résultats successifs).

Nous avons vu d'une part que, tous les modes (à l'exception d'ECB) requièrent un 'Vecteur d'Initialisation : IV'. C'est un bloc de données aléatoires pour démarrer le chiffrement du premier bloc et fournir ainsi une forme de hasard indépendant du document à chiffrer. Il n'a pas besoin d'être lui-même chiffré lors de la transmission, mais il ne doit jamais être réemployé avec la même clé et d'autre part que, le mode ECB n'est pas sûr. Les modes CBC, OFB, CFB ou CTR doivent lui être préférés.

Aussi, on considérait que la protection de l'intégrité était un objectif à atteindre par des moyens complètements différents. Mais aujourd'hui il existe des modes d'opérations qui associent chiffrement et authentification de manière efficace, tel que, par exemple XCBC, IACBC, IAPM, OCB, CWC ou TCB.

Enfin, nous avons proposé dans ce travail :

I. Une méthode de génération des vecteurs d'initialisations de façon à satisfaire la rondomisation du vecteur,
II. Aussi, nous avons proposé un nouveau mode sécurisé : le mode **ASK** (**A**utonome **S**ecure bloc**K**), et
III. Suggérer l'augmentation de la clé de chiffrement en une super clé contenant outre la clé secrète les données nécessaires (r, t, x_0) pour le démarrage du générateur des vecteurs d'initialisations.

Références

[1] B. Schneier," Applied Cryptography-Protocols, Algorithms and Source Code in C", John Wiley & Sounds, Inc, New York, Second Edition, 1996.

[2] http://www.securite-informatique.gouv.fr/autoformations/cryptologie/co/cryptologie_CH03_2.html

[3] http://en.wikipedia.org/wiki/Block_cipher_modes_of_operation

[4] Mihir Bellare, Ran Canetti et Hugo Krawczyk, "*Message authentication using hash functions: The HMAC construction*", Appears in RSA Laboratories' CryptoBytes, Vol. 2, No. 1, Spring 1996.

[5] Mihir Bellare, Ran Canetti et Hugo Krawczyk, "*Keying Hash Functions for Message Authentication*", Advances in Cryptology (Crypto 96 Proceedings), Lecture Notes in Computer Science Vol. 1109, p. 1-15, N. Koblitz ed., Springer-Verlag, 1996.

[6] James Massey, Gurgen Khachatrian, Melsik Kuregian, "Nomination of SAFER++ as Candidate Algorithm for the New European Schemes for Signatures, Integrity, and Encryption (NESSIE)," Presented at the First Open NESSIE Workshop, November 2000.

[7] Debasis Gountia, Anil Kumar Swain, "A Novel Disk Encompression with Tweaked Code Book (DETCB)", International Journal of Computer Sciences and Technology: IJCST Vol. 2, Issue 2, June 2011.

[8] Wikipédia. "Disk encryption theory". http://fr.wikipedia.org/wiki/ Disk encryption theory.

[9] A. ALI-PACHA & N. HADJ-SAID & L. MERAH & M.S. ALI-PACHA, "Génération des Données Chaotiques de la Carte Logistique en vue de leur Application dans les Crypto Systèmes'', 10ième Colloque d'Algèbre Théorie des Nombres, Département de Mathématique - Faculté des Sciences Exactes - Université Mentouri Constantine, du 19 au 20 Octobre 2011.

[10] A. Ali-Pacha, N. Hadj-Said, A. M'Hamed and A.Belgoraf "Lorenz's attractor applied to the stream cipher (Ali-Pacha generator)", Chaos, Solitons & Fractals, Volume 33, Issue 5, pp 1762-1766. 2007.