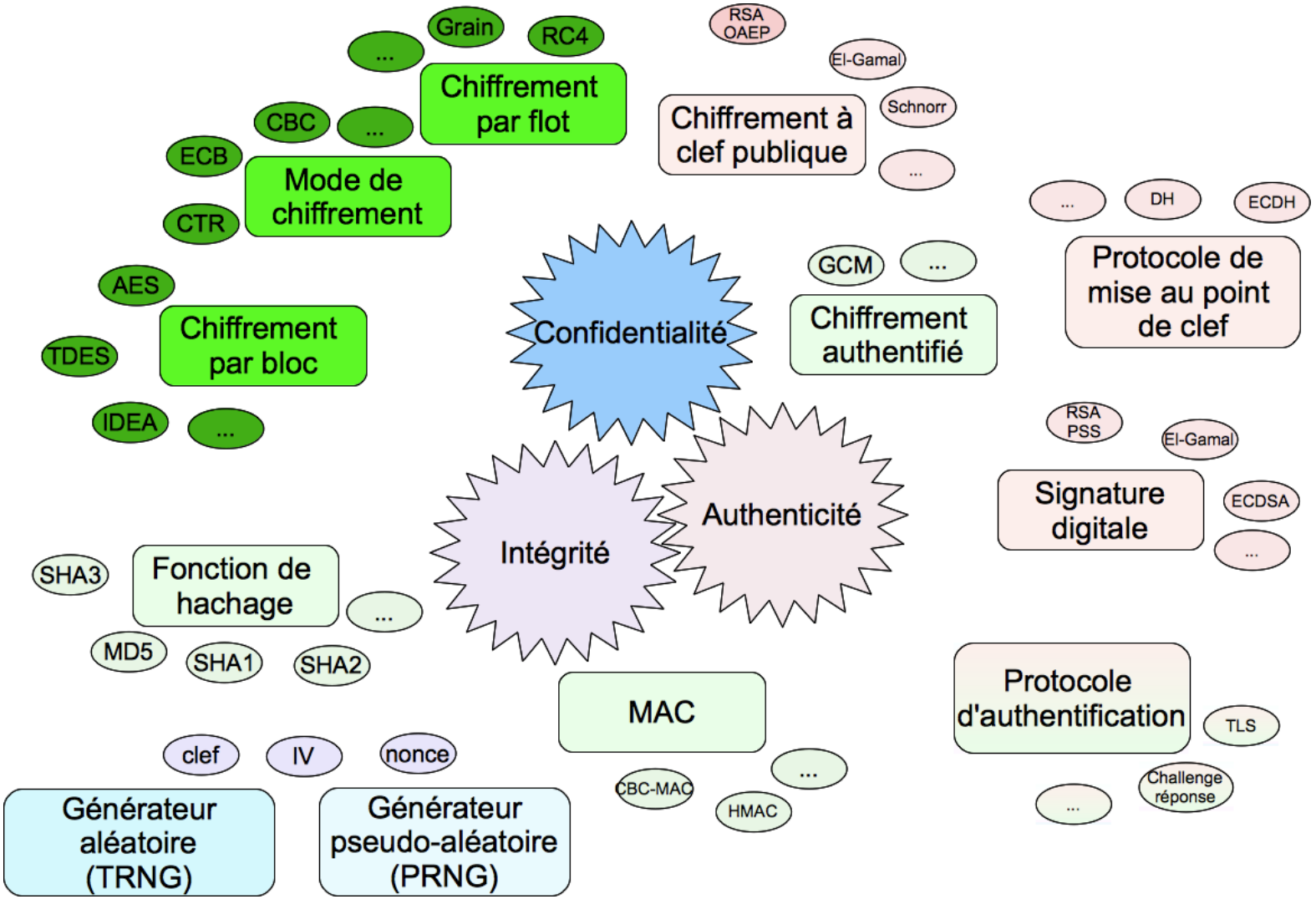
CRY 2024

Chiffrement Symétrique

Alexandre Duc

Vue Synoptique



Algorithme de Chiffrement Symétrique

Définition (Algorithme de Chiffrement Symétrique)

Un **algorithme de chiffrement symétrique** est une famille $E_K(.)$ de transformations paramétrée par une clef K :

$$E : \begin{cases} \mathcal{X} \times \mathcal{K} & \rightarrow & \mathcal{Y} \\ (X, K) & \mapsto & Y = E_K(X) \end{cases}$$

où $\mathcal{X}$ est l'espace des textes clairs, $\mathcal{Y}$ l'espace des textes chiffrés et $\mathcal{K}$ l'espace des clefs, et où X représente un texte clair, Y un texte chiffré et K une clef.

Algorithme de Chiffrement Symétrique

Pour chaque fonction de chiffrement

$$E : \begin{cases} \mathcal{X} \times \mathcal{K} & \to & \mathcal{Y} \\ (X, K) & \mapsto & Y = E_K(X) \end{cases}$$

il doit exister une fonction de déchiffrement correspondante

$$D : \begin{cases} \mathcal{Y} \times \mathcal{K} & \to & \mathcal{X} \\ (Y, K) & \mapsto & X = D_K(Y) \end{cases}$$

telle que, pour tout $(X, K) \in \mathcal{X} \times \mathcal{Y}$,

$$D_K(E_K(X)) = X.$$

En d'autres termes, un algorithme de chiffrement symétrique est une famille de **permutations** indexée par une clef.

Algorithme de Chiffrement Symétrique

- D'ordinaire, on a

$$\mathcal{X} = \mathcal{Y} = \{0,1\}^n$$
$$\mathcal{K} = \{0,1\}^\ell$$

où $n = 64, 128$ et $\ell = 64, \ldots, 256$.

Question

Combien de permutations a-t-on sur $\{0,1\}^n$?

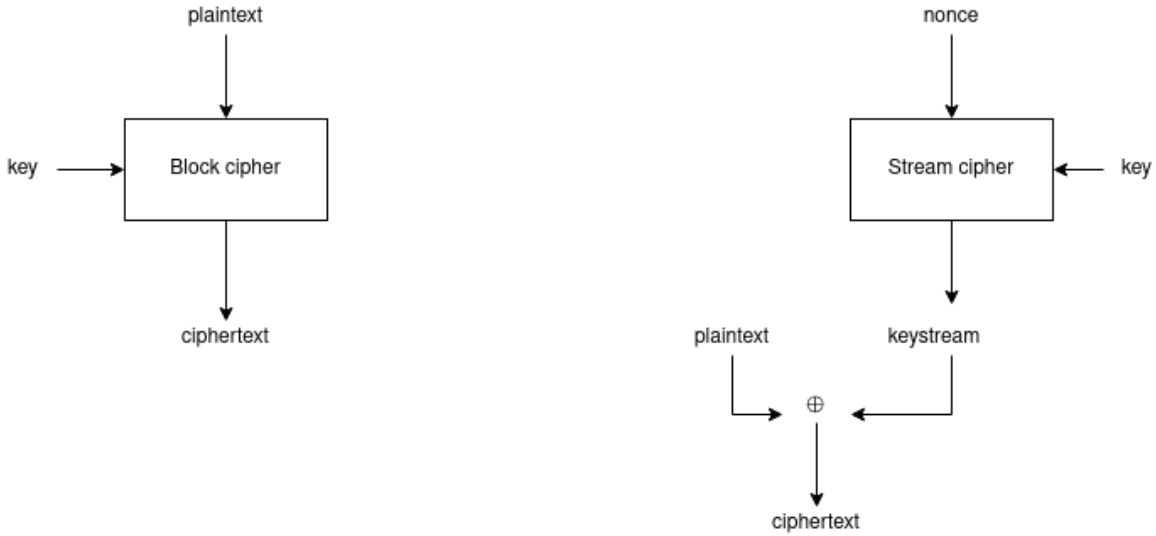
Chiffrement déterministe

- La définition du chiffrement symmétrique vue jusqu'à maintenant est **déterministe**.
- Sous une **même clef**, un **même texte clair** donnera le **même texte chiffré**.
- Or, la **clef** ne change **pas souvent** !
- Ceci est donc une **fuite d'information**.

Vecteurs d'initalisation et Nonces

- Nous ajoutons un **troisième paramètre** le **vecteur d'initialisation (IV)** à notre algorithme de chiffrement.
- Il sera **différent** pour chaque nouveau message chiffré (sous la même clef).
- Souvent **tiré aléatoirement** et **envoyé en clair** avec le texte chiffré.
- Appelé un **nonce** dans certaines situations (pour number used once).

Chiffrement par Blocs vs Chiffrement par Flot



- Un algorithme de **chiffrement par blocs** («block cipher») prend en entrée un bloc de texte clair et une clef et a en sortie un texte chiffré.
- Un algorithme de **chiffrement par flot** («stream cipher») prend en entrée un nonce et une clef et a en sortie un flot de bits qui sont ensuite XOR au texte clair.

Chiffrement par Blocs vs Chiffrement par Flot (2)

- Les algorithmes de chiffrement par blocs sont **déterministes** et ne peuvent chiffrer qu'un nombre **fixe** de bits.
- Besoin de **modes opératoires**.
- Un algorithme de chiffrement par flot est souvent plus **efficace**.
- La **sécurité** des algorithmes de chiffrement par flot est historiquement plus douteuse.

Modèles de Sécurité : Buts

- Récupérer la clef secrète.
- Décrypter un message.
- Récupérer un bit d'information
- ...

Modèles de Sécurité : Capacités

- **Attaque à texte chiffré** («ciphertext-only») : l'adversaire connaît uniquement un ou plusieurs textes chiffrés.
- **Attaque à texte clair connu** («known-plaintext») : l'adversaire connaît un certain nombre de textes clairs et leur textes chiffrés correspondants.
- **Attaque à texte clair choisi** («chosen-plaintext») : l'adversaire a obtenu un accès temporaire à la «machine» de chiffrement. Il peut donc choisir des textes clairs et obtenir les textes chiffrés correspondants.
- **Attaque à texte chiffré choisi** : («chosen-ciphertext») Identique à l'attaque précédente, mais au moyen d'une «machine» de déchiffrement.
- **Attaque à texte clair et chiffré choisi** : L'adversaire a accès à une machine de chiffrement **et** de déchiffrement.

Attaques avec Accès à un Oracle

- Lorsqu'un adversaire à accès à la "machine" de chiffrement, on dit qu'elle a accès à un **oracle** de chiffrement.
- Ces attaques peuvent être faite de manière **adaptative** si l'adversaire attend la réponse de l'oracle avant de faire une nouvelle requête.
- Ces attaques peuvent être aussi faite de manière **non-adaptative** si l'adversaire doit effectuer toutes ses requêtes en même temps.

Question

Quel est le sens de simuler un adversaire qui peut déchiffrer des messages ?

Sécurité

Attention

Un cryptosystème qui ferait fuiter même qu'un seul bit d'information est **cassé** et ne doit pas être utilisé.
En pratique, la simple capacité de pouvoir **distinguer** un texte chiffré d'un message aléatoire est déjà suffisant pour casser un système.

Algorithmes Linéaires

- Les **algorithmes de chiffrement linéaires** forment une famille qui comporte certains algorithmes historiques :
  - le chiffre de Vigenère ;
  - le chiffre de Hill ;
  - le chiffre par permutation ;
  - ...
- **Attention : Ces algorithmes sont totalement faibles et ne doivent jamais être utilisés en pratique !**

Fonction Linéaire

Définition (Fonction Linéaire)

Une fonction

$$f : \mathbb{Z}_m^n \to \mathbb{Z}_m^\ell$$

est **linéaire** si vous pouvez la décrire à l'aide d'une matrice M de taille $\ell \times n$ avec des cœfficients dans $\mathbb{Z}_m$ tel que $f(x) = Mx \bmod m$.

Fonction Affine

Une fonction affine $f : \mathbb{Z}_m^n \to \mathbb{Z}_m^\ell$ peut être décrite par une matrice M de type $\ell \times n$ avec des cœfficients dans $\mathbb{Z}_m$ ainsi qu'un vecteur $\boldsymbol{b} \in \mathbb{Z}_m^\ell$ tel que, pour tous les vecteurs $\boldsymbol{v} \in \mathbb{Z}_m^n$, on ait

$$f(\boldsymbol{v}) = (M\boldsymbol{v} + \boldsymbol{b}) \bmod m.$$

Les conditions nécessaires pour que f forme une permutation sont les suivantes :

- $\ell = n$
- Le déterminant de M doit être premier avec m.

Chiffre de Vigenère

Question

Comment peut-on casser le chiffre de Vigenère à l'aide d'une simple pair de texte clair connu ?

Chiffre de Hill

- L'espace des clefs est l'ensemble de toutes les matrices $n \times n$ inversibles possédant des cœfficients dans $\mathbb{Z}_m$.
- Le chiffrement d'un texte clair $\boldsymbol{x} \in \mathbb{Z}_m^n$ avec la clef $\boldsymbol{K} \in \mathbb{Z}_m^n \times \mathbb{Z}_m^n$ est défini par

$$E_{\boldsymbol{K}} : \boldsymbol{x} \mapsto \boldsymbol{y} = \boldsymbol{K}\boldsymbol{x} \pmod{m}$$

- Le déchiffrement d'un texte chiffré $\boldsymbol{y} \in \mathbb{Z}_m^n$ avec la clef $\boldsymbol{K} \in \mathbb{Z}_m^n \times \mathbb{Z}_m^n$ est défini par

$$D_{\boldsymbol{K}} : \boldsymbol{y} \mapsto \boldsymbol{x} = \boldsymbol{K}^{-1}\boldsymbol{y} \pmod{m}$$

Chiffre de Hill : Cryptanalyse

Question

Montrer comment casser le chiffre de Hill à l'aide de n paires de textes clairs connus.

Attaque des Chiffrements Affines

- Il est possible de décrire un chiffrement affine par l'opération $E : \boldsymbol{x} \mapsto \boldsymbol{K}\boldsymbol{x} + \boldsymbol{k} \pmod{m}$ où la clef est la paire $(\boldsymbol{K}, \boldsymbol{k})$.
- Si un adversaire possède $n+1$ paires de texte clair connu $(\boldsymbol{x}_0, \boldsymbol{y}_0), \ldots, (\boldsymbol{x}_n, \boldsymbol{y}_n)$, on peut écrire

$$\boldsymbol{y}_i - \boldsymbol{y}_0 = \boldsymbol{K}(\boldsymbol{x}_i - \boldsymbol{x}_0) \pmod{m}$$

  pour $1 \leq i \leq n$.
- De façon compacte, il est possible de réécrire ces relations sous forme matricielle, avec les colonnes de $\boldsymbol{X}$ formé des $\boldsymbol{x}_i$

$$\boldsymbol{K}\boldsymbol{X} = \boldsymbol{Y} \pmod{m}.$$

- Si $\det(\boldsymbol{X})$ est premier avec m, on peut alors retrouver

$$\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{X}^{-1} \pmod{m}$$

  dans un premier temps, puis $\boldsymbol{k}$ dans un second temps.

Recherche Exhaustive d'une Clef

- Il existe une attaque qu'il est toujours possible d'effectuer en théorie, la **recherche exhaustive de clef**.
- Le principe consiste simplement à essayer toutes les clefs possibles sur un texte chiffré jusqu'à que la bonne soit trouvée.
- Une recherche exhaustive de clef exige un **critère d'arrêt**.

Question

Quel est le nombre de clefs en moyenne à essayer pour des clefs de ℓ bits ? Et dans le pire cas ?

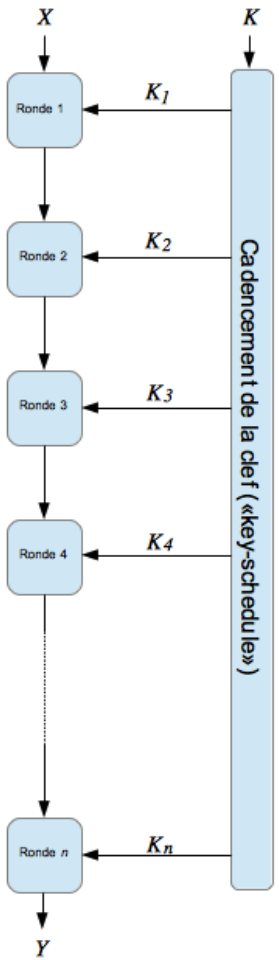
Tailles des Clefs symétriques

- En 2018, une taille de clef de 80 bits est généralement insuffisante. 128 bits est une taille considérée comme suffisante.
- La **loi de Moore** joue en défaveur de la sécurité d'un chiffrement : à budget égal, la puissance de calcul **double** environ tous les **18 mois**.

Chiffrement Itéré

- Une façon standard de construire un algorithme de chiffrement par bloc consiste à itérer une **fonction de ronde** un certain nombre de fois.
- Chaque fonction de ronde prend comme paramètre une **sous-clef**, qui est dérivée de la clef principale au moyen d'un algorithme de **cadencement de clef** («key-schedule»).

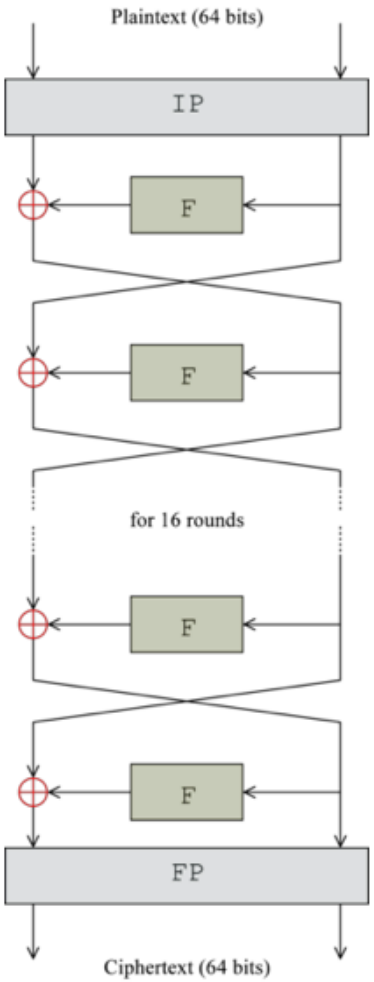
Chiffrement Itéré (2)

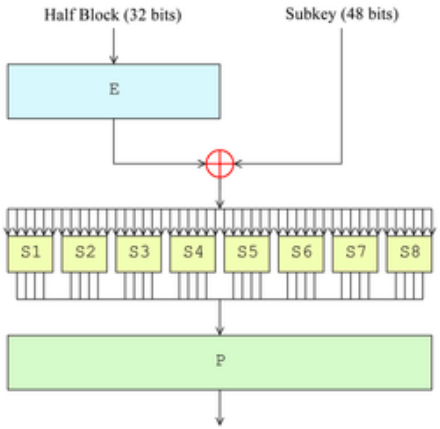


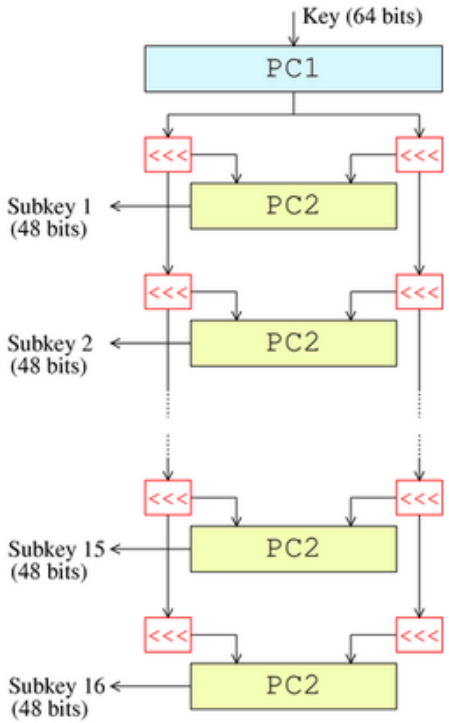
Data Encryption Standard

- Data Encryption Standard (DES)
  - Paramétré par une clef de 56 bits
  - Chiffre des blocs de données de 64 bits
  - Est bâti sur un **schéma de Feistel** à 16 rondes
- Conçu par une équipe d'IBM dans les années 1970, dérivé d'un algorithme appelé Lucifer, et «revu» par la NSA.
- Standard de chiffrement américain de 1977 à 2005.
- Souffre d'une taille de clef et de bloc trop petite.

Data Encryption Standard (2)







Source des illustrations : http://en.wikipedia.org/wiki/Data_Encryption_Standard

Data Encryption Standard (3)

- IP est une matrice de permutation sur $\mathbb{Z}_2$ de taille 64×64, et FP est son inverse.
- E est une fonction linéaire qui étend son entrée de 32 bits en une sortie de 48 bits, en dupliquant certains bits.
- P est une matrice de permutation sur $\mathbb{Z}_2$ de taille 32×32.
- PC1 est une fonction linéaire compressant son entrée de 64 bits en 56 bits.
- PC2 est une fonction linéaire compressant son entrée de 56 bits en 48 bits.

Data Encryption Standard (4)

- S1, ..., S8 sont des boîtes des substitution («S-box») prenant 6 bits en entrée et retournant 4 bits.

S_5		Middle 4 bits of input															
		0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Outer bits	**00**	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011

Source de l'illustration : http://en.wikipedia.org/wiki/Substitution_box

Triple-DES

- DES est encore largement utilisé sous la forme de **Triple-DES** :
  - Triple-DES à deux clefs : $Y = \mathrm{DES}_{K_1}(\mathrm{DES}^{-1}_{K_2}(\mathrm{DES}_{K_1}(X)))$.
  - Triple-DES à trois clefs : $Y = \mathrm{DES}_{K_3}(\mathrm{DES}^{-1}_{K_2}(\mathrm{DES}_{K_1}(X)))$.

Question

Pourquoi Triple-DES utilise-t-il la séquence «chiffrer-déchiffrer-chiffrer» plutôt que celle «chiffrer-chiffrer-chiffrer» ?

Advanced Encryption Standard

- Advanced Encryption Standard (AES)
- AES est un sous-ensemble de l'algorithme **Rijndael**, conçu par Joan Daemen et Vincent Rijmen à la fin des années 1990.
- Sélectionné après une compétition ouverte à tout le monde.
- Chiffre des données de 128 bits
- Supporte une clef de 128 bits (10 rondes), 192 bits (12 rondes) et 256 bits (14 rondes).
- Très efficace sur les plateformes embarquées, CPU standards, ainsi qu'en hardware.

Advanced Encryption Standard

Création d'un état initial

Un texte clair de 16 bytes $B_0, B_1, \ldots B_{15}$ est transformé dans la matrice (état) suivante :

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix}$$

Advanced Encryption Standard

Représentation binaire - polynomiale

- On utilise $\mathbb{Z}_2[x]/(x^8 + x^4 + x^3 + x + 1)$ pour représenter $\mathrm{GF}(2^8)$.
- On représente un byte $b_7 \dots b_0$ à l'aide du polynôme $b_7 x^7 + \cdots + b_1 x + b_0$ sur $\mathrm{GF}(2^8)$.
- Une **addition** est un simple XOR.
- **Une multiplication par** `0x01` ne nécessite pas d'actions.
- **Une multiplication par** `0x02` correspond à un shift vers la gauche ainsi qu'un XOR avec `0x1b` s'il y a un carry.
- **Une multiplication par** `0x03` correspond à un XOR des multiplications par `0x02` et `0x01`.

Advanced Encryption Standard

Opérations

Ronde initiale :

- `AddRoundKey`

Rondes internes (effectué ⟨nb_rondes -1⟩ fois) :

- `SubBytes`
- `ShiftRows`
- `MixColumns`
- `AddRoundKey`

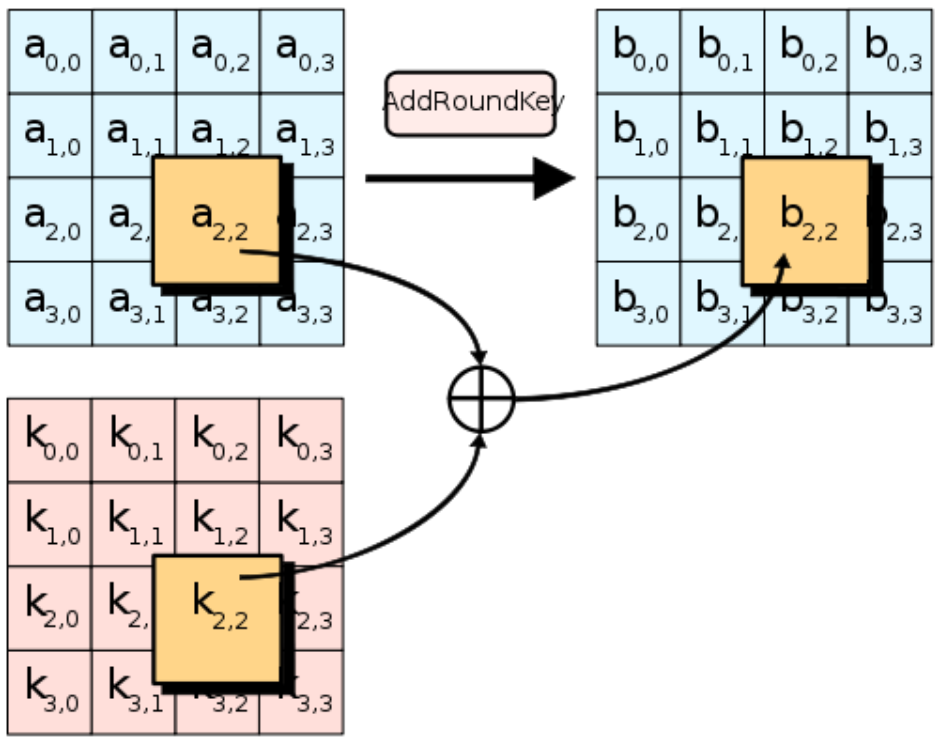
Ronde finale :

- `SubBytes`
- `ShiftRows`
- `AddRoundKey`

Advanced Encryption Standard

AddRoundKey

AddRoundKey est un simple XOR entre l'état courant et la sous-clef correspondante.



Source de l'illustration : http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

Advanced Encryption Standard

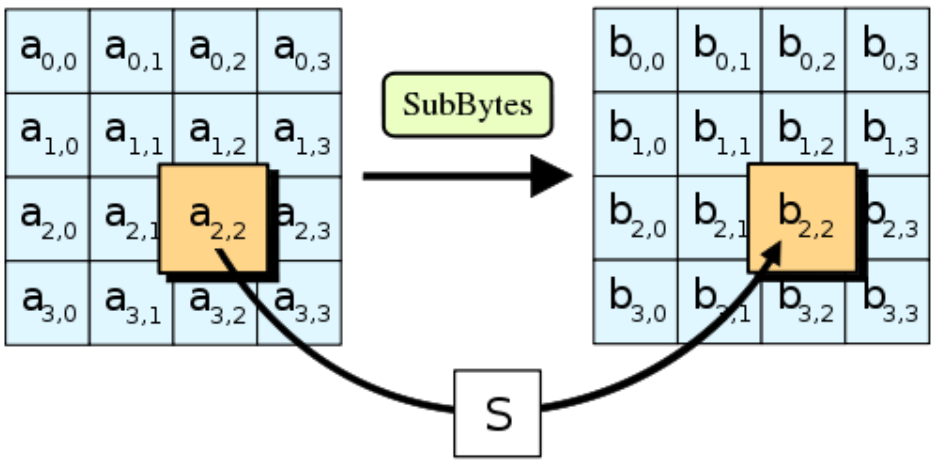
SubBytes

- SubBytes est une «boîte de substitution», c'est-à-dire une **permutation non-linéaire** transformant des entrées de 8 bits en une sortie de 8 bits.
- Mathématiquement, il s'agit de l'inversion dans $\mathrm{GF}(2^8)$ suivie de l'application affine suivante sur $\mathbb{Z}_2^8$ (les bytes sont représentés comme $b_7 \dots b_0$) :

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Advanced Encryption Standard

SubBytes



Source de l'illustration : http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

Advanced Encryption Standard

MixColumns

- `MixColumns` est une permutation sur les mots de 32 bits qui possède un haut pouvoir de **diffusion**.
- Cette permutation est définie comme une fonction linéaire sur $(\mathbb{Z}_2[x]/(x^8 + x^4 + x^3 + x + 1))^4$:

$$\boldsymbol{x} \mapsto \begin{pmatrix} x & x+1 & 1 & 1 \\ 1 & x & x+1 & 1 \\ 1 & 1 & x & x+1 \\ x+1 & 1 & 1 & x \end{pmatrix} \boldsymbol{x}$$

Advanced Encryption Standard

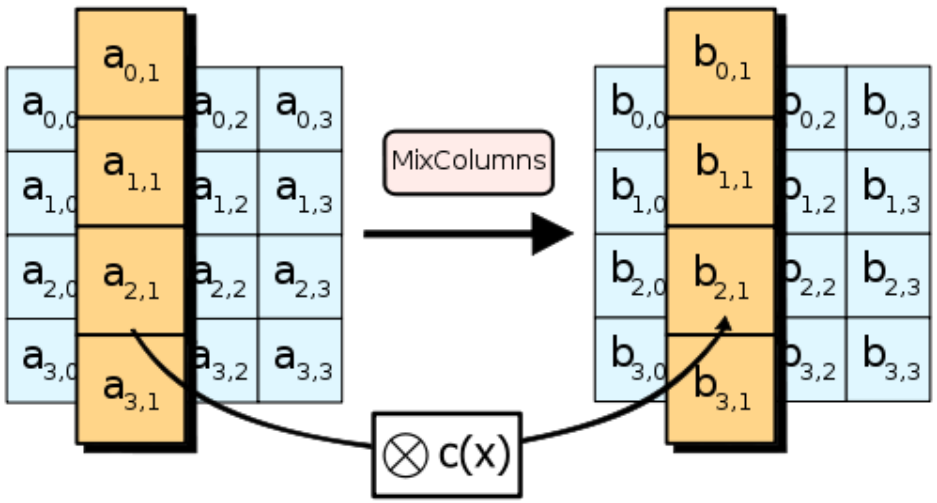
MixColumns

- On peut aussi représenter cette matrice de façon plus concise à l'aide de la notation par bytes :

$$\boldsymbol{x} \mapsto \begin{pmatrix} \texttt{0x02} & \texttt{0x03} & \texttt{0x01} & \texttt{0x01} \\ \texttt{0x01} & \texttt{0x02} & \texttt{0x03} & \texttt{0x01} \\ \texttt{0x01} & \texttt{0x01} & \texttt{0x02} & \texttt{0x03} \\ \texttt{0x03} & \texttt{0x01} & \texttt{0x01} & \texttt{0x02} \end{pmatrix} \boldsymbol{x}$$

Advanced Encryption Standard

MixColumns

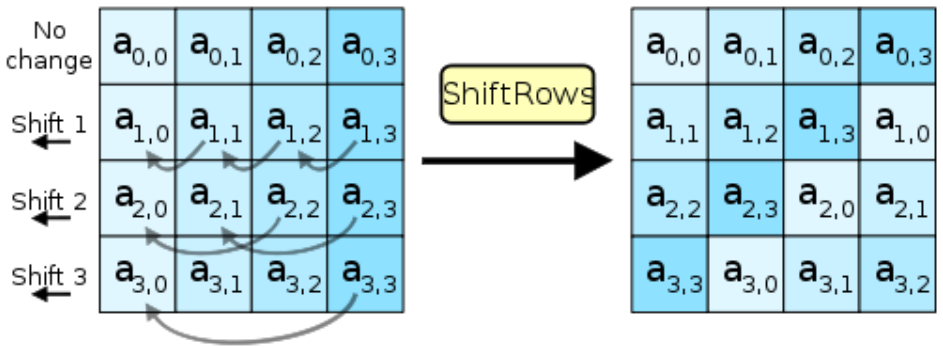


Source de l'illustration : http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

Advanced Encryption Standard

ShiftRows

- ShiftRows est une simple permutation d'octets définie sur les vecteurs de 4 octets.



Source de l'illustration : http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

1. Permutations et Chiffrement Symétrique
2. Algorithmes Linéaires
3. Chiffrement par Bloc
4. Chiffrement par Flot
   - Masque Jetable
   - RC4
   - ChaCha20

Masque Jetable («One-Time Pad»)

- Soit un message $\boldsymbol{x}$ de ℓ bits. On génère une chaîne $\boldsymbol{k}$ de ℓ bits de manière **uniformément aléatoire**, que l'on appelle «clef».
- On transmet la clef de manière sûre à Bob.
- On calcule le texte chiffré $\boldsymbol{y} = \boldsymbol{x} \oplus \boldsymbol{k}$, où $\oplus$ représente un XOR («ou exclusif»).
- Ce chiffrement est **parfaitement sûr**, et résiste à tout type d'adversaire, même ceux possèdant des capacités infinies de calcul ou un ordinateur quantique.
- Ce chiffrement est **affine** !

Chiffrement Parfaitement Sûr

Définition (Chiffrement Parfaitement Sûr)

Un chiffrement est appelé **parfaitement sûr** si le texte clair $\boldsymbol{X}$ et le texte chiffré $\boldsymbol{Y}$ sont statistiquement indépendants, c'est-à-dire si

$$\Pr[\boldsymbol{X} \mid \boldsymbol{Y}] = \Pr[\boldsymbol{X}].$$

- Rappel : par définition des probabilités conditionnelles, on a

$$\Pr[\boldsymbol{X} \mid \boldsymbol{Y}] = \frac{\Pr[\boldsymbol{X} \wedge \boldsymbol{Y}]}{\Pr[\boldsymbol{Y}]}.$$

Sécurité du Masque Jetable

Question

Que se passe-t-il lorsque la même clef est utilisée deux fois ou plus dans le cadre du masque jetable ?

Masque Jetable en Pratique - Inconvénients

- Une clef ne peut être utilisée qu'une seule fois ;
- le texte chiffré peut être manipulé si l'on n'utilise pas de mécanisme de contrôle d'intégrité ;
- la transmission de la clef pose de nombreux problèmes en pratique.
- De plus, générer une clef **parfaitement aléatoire** est une tâche difficile.

RC4

- Algorithme conçu par Ronald Rivest pour l'entreprise RSA Security (maintenant EMC).
- Sa définition a fui sur Internet en 1994.
- Cet algorithme génère un flux d'octets pseudo-aléatoires, qui peut être additionné à un flux de messages via un XOR.
- Il admet une clef de 40 à 2048 bits.
- Utilisé notamment dans SSL/TLS et dans WEP.

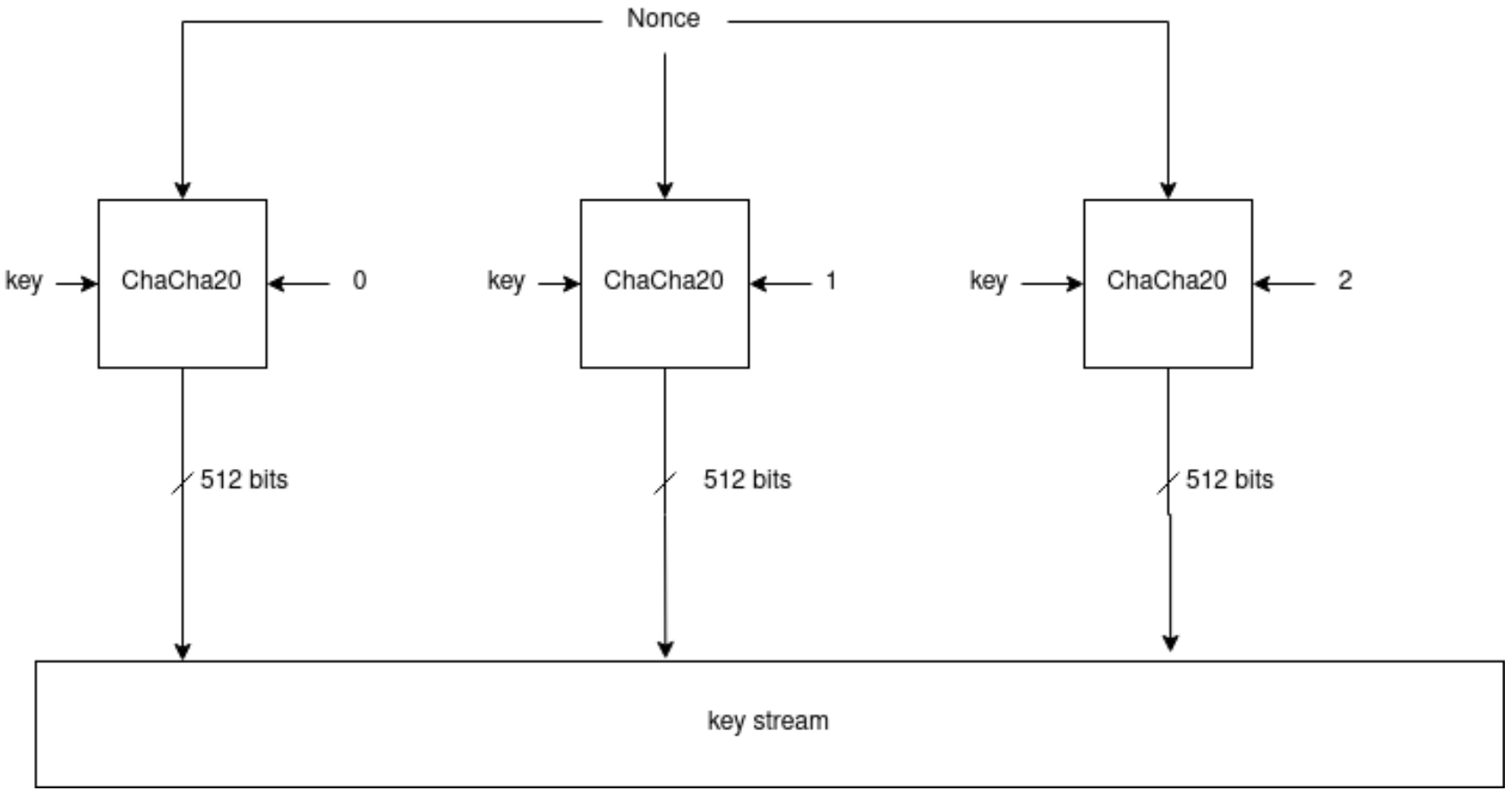
Sécurité de RC4

- Il n'est **plus recommandé d'utiliser RC4** en pratique :
  - **Attention :** Les premiers 256 octets du flux possèdent de nombreuses **faiblesses statistiques exploitables en pratique**.
  - **Attention :** RC4 n'utilise **pas de vecteur d'initialisation** ! Une clef ne doit donc être utilisée que pour chiffrer **un seul** message.

ChaCha20

- Inventé par Dan Bernstein comme une amélioration de Salsa20.
- Système de chiffrement par flots le plus utilisé actuellement.
- **Clefs** de 128 bits ou 256 bits.
- Utilise un **nonce** de 64 bits et un marqueur de **position** de 64 bits.
- Variante avec un **nonce** de 96 bits et un compteur de 32 bits.
- $f(\text{nonce}, \text{counter}, \text{key}) \rightarrow \{0,1\}^{512}$.
- Possibilité de chiffrer ou de déchiffrer un bloc de 512 bits sans devoir calculer tout le flux.

ChaCha20


Nonce
key
ChaCha20
0
key
ChaCha20
1
key
ChaCha20
2
512 bits
512 bits
512 bits
key stream

Etat de ChaCha20

- Etat de 512 bits.
- **Cons** : constante : “expand 32-byte k” en ASCII pour des clefs de 256 bits, “expand 16-byte k” en ASCII pour des clefs de 128 bits.
- **Clef** : Soit la clef soit la clef dupliquée (pour des clefs de 128 bits).
- Chaque élément est un **mot** de 32 bits.

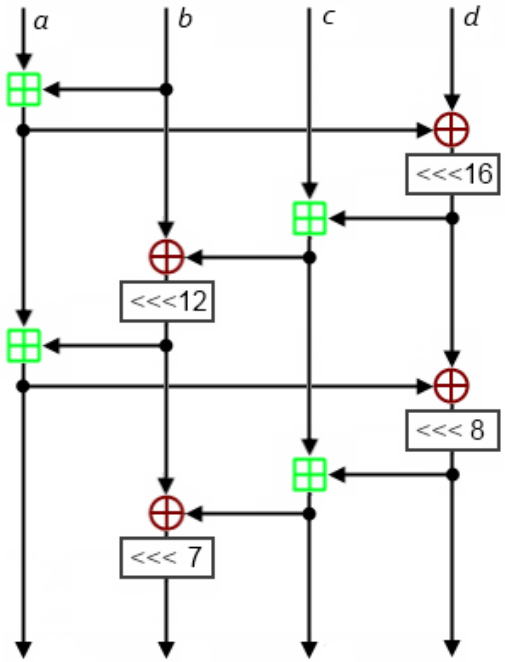
ChaCha20 Rounds

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \\ 12 & 13 & 14 & 15 \end{bmatrix}$$

- 10 **double rounds** composés de 8 **quarter-rounds**.
- Un double round est

```
// round impair
QR(0, 4,  8, 12)  // 1st column
QR(1, 5,  9, 13)  // 2nd column
QR(2, 6, 10, 14)  // 3rd column
QR(3, 7, 11, 15)  // 4th column
// Round pair
QR(0, 5, 10, 15)  // diagonal 1 (main diagonal)
QR(1, 6, 11, 12)  // diagonal 2
QR(2, 7,  8, 13)  // diagonal 3
QR(3, 4,  9, 14)  // diagonal 4
```

ChaCha20 Quarter-Rounds



- Construction **ARX** : addition, rotation, XOR.
- Trois opérations : $\oplus$ est un XOR, $\boxplus$ est une addition modulo 2^{32} et $<<<$ un shift cyclique.

Chiffrement Chacha20

- On initialise l'état avec la clef, le nonce et le compteur.
- On applique les 10 double rounds sur l'état.
- On additionne l'état initial à l'état final mot par mot (donc modulo 2^{32}) pour obtenir le flux.

Attention

Ne surtout pas oublier cette dernière étape (voir exo).

Solutions

Algorithme de Chiffrement Symétrique

Solution

Nous avons $2^n!$ permutations sur $\{0,1\}^n$. Pour pouvoir toutes les représenter, il nous faudrait donc $2^n!$ clefs.

Attaques avec Accès à un Oracle

Solution

Le raisonnement est le suivant : si l'on est capable de se protéger contre un adversaire qui peut déchiffrer des messages, on peut aussi se protéger contre des adversaires plus faibles qui récupèrent seulement un peu d'information sur chaque texte clair. C'est typiquement le type d'attaque effectuée lors d'une padding oracle attaque où l'on récupère un bit d'information sur le texte clair.

Chiffre de Vigenère

Solution

On récupère simplement la clef en soustrayant le texte chiffré au texte clair.

Sécurité du Masque Jetable

Si l'on réutilise une clef pour chiffrer deux textes clairs m_1 et m_2 et obtenir deux textes chiffrés c_1 et c_2, un adversaire peut calculer $c_1 \oplus c_2$ et obtenir $m_1 \oplus m_2$. Ceci peut donner énormément d'information à l'adversaire : nous avons quelque chose qui dépend uniquement des textes clairs et plus de la clefs.

Recherche Exhaustive de Clef

Solution

En moyenne, il faut essayer $2^{\ell-1}$ clefs. Dans le pire cas 2^{ℓ}.

Triple-DES

Solution

Pour des raisons de rétro-compatibilité. A l'époque, beaucoup de chips étaient capables de faire uniquement du chiffrement DES simple. Le but était de créer des chips capables de faire à la fois du DES simple et du triple-DES, ce qui est possible avec cette solution : pour obtenir du DES simple, il suffit d'y mettre 3 fois la même clef.