Chiffrement par flot

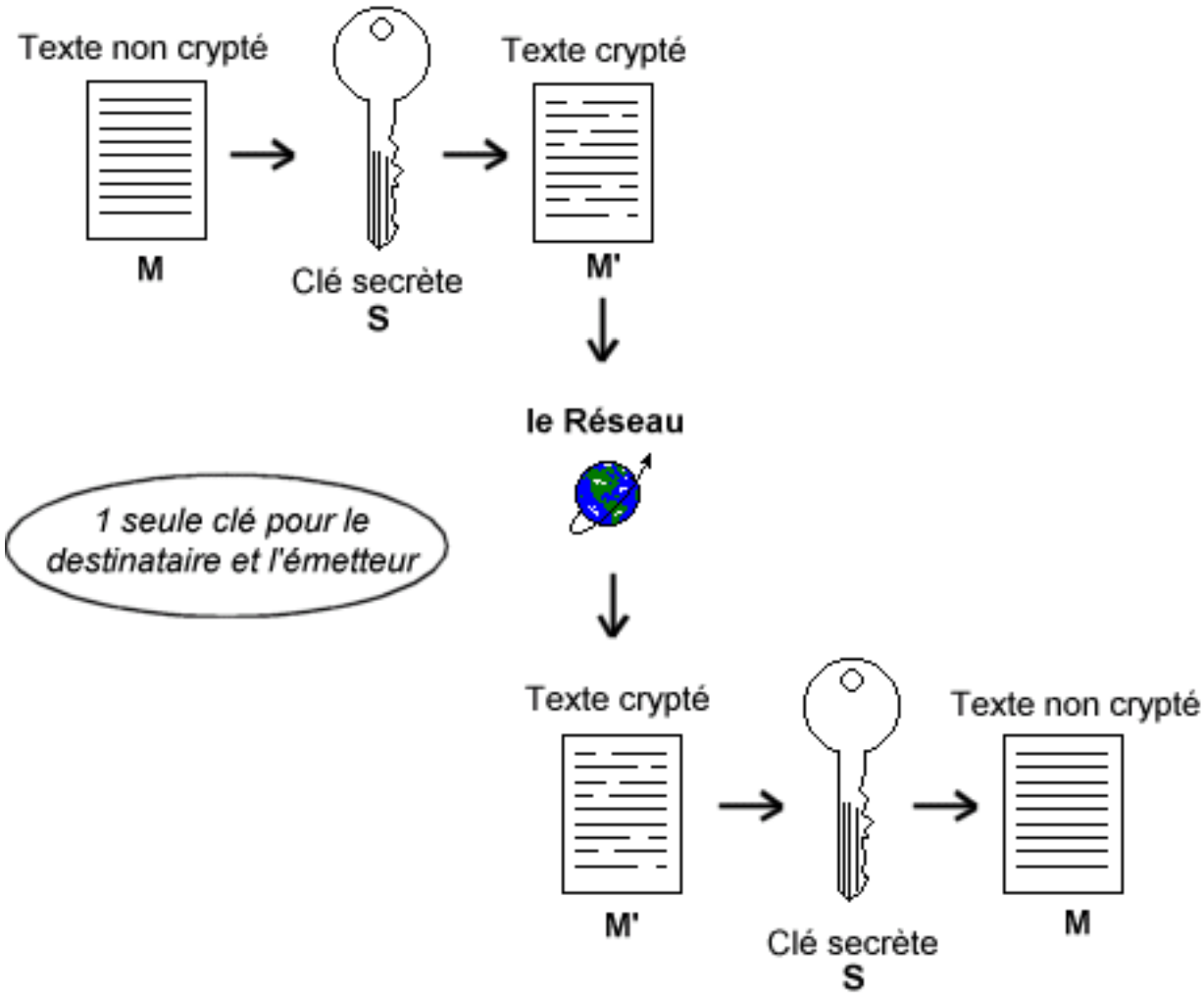
(Stream cipher)

October 12, 2012

Plan

Chiffrement à clef privée

On considère deux protagonistes Alice et Bob partageant une clef privée et secrète et s'envoyant un message chiffré:



Plan

Chiffrement de Vernam *(One Time Pad)*

La clef secrète d'Alice et Bob est $K = k_0 \dots k_{n-1}$ (générée aléatoirement)

- Chiffrement : Alice chiffre $M = m_0 m_1 \dots m_{n-1}$ en $C = c_0 \dots c_{n-1}$ où

$$c_i = m_i \oplus k_i$$

- Déchiffrement : Bob récupère le message clair M en calculant

$$m_i = c_i \oplus k_i \text{ pour } i = 0, \dots, n-1.$$

On vérifie que Bob retrouve bien M: pour tout $i = 0, \dots, n-1$

$$c_i \oplus k_i = (m_i \oplus k_i) \oplus k_i = m_i \oplus (k_i \oplus k_i) = m_i$$

Chiffrement de Vernam *(One Time Pad)*

La clef secrète d'Alice et Bob est $K = k_0 \dots k_{n-1}$ (générée aléatoirement)

- Chiffrement : Alice chiffre $M = m_0 m_1 \dots m_{n-1}$ en $C = c_0 \dots c_{n-1}$ où

$$c_i = m_i \oplus k_i$$

- Déchiffrement : Bob récupère le message clair M en calculant

$$m_i = c_i \oplus k_i \text{ pour } i = 0, \dots, n-1.$$

On vérifie que Bob retrouve bien M: pour tout $i = 0, \dots, n-1$

$$c_i \oplus k_i = (m_i \oplus k_i) \oplus k_i = m_i \oplus (k_i \oplus k_i) = m_i$$

Point important : la clef K est une utilisée une seule fois!

Chiffrement de Vernam - Sécurité inconditionnelle

Montrons que pour tout i, c_i prend la valeur 0 ou 1 avec une probabilité de $1/2$:

- On suppose m_i prend la valeur 0 avec une proba de p_i et 1 de $1 - p_i$.
- On suppose que k_i prend la valeur 1 et 0 avec une proba de $1/2$ chacun.
- m_i et k_i sont indépendants (ce qui fait sens vu que k_i est choisi indépendement de M).

Chiffrement de Vernam - Sécurité inconditionnelle

Montrons que pour tout i, c_i prend la valeur 0 ou 1 avec une probabilité de $1/2$:

- On suppose m_i prend la valeur 0 avec une proba de p_i et 1 de $1 - p_i$.
- On suppose que k_i prend la valeur 1 et 0 avec une proba de $1/2$ chacun.
- m_i et k_i sont indépendants (ce qui fait sens vu que k_i est choisi indépendement de M).

Alors

$$\begin{aligned} P(c_i = 0) &= P(\{m_i = 0 \text{ et } k_i = 0\} \cup \{m_i = 1 \text{ et } k_i = 1\}) \\ &= P(\{m_i = 0 \text{ et } k_i = 0\}) + P(\{m_i = 1 \text{ et } k_i = 1\}) \\ &= p_i \cdot \tfrac{1}{2} + (1 - p_i) \cdot \tfrac{1}{2} \\ &= \tfrac{1}{2} \end{aligned}$$

Donc la suite c_i ressemble à du bruit.

Chiffrement de Vernam - Sécurité inconditionnelle

- On voit que chacun des bits du message chiffré ne contient aucune information sur le message clair.
- En théorie de l'information cela se traduit en

$$H(M|C) = H(M)$$

où H est l'entropie d'une variable aléatoire.

- L'inconvénient majeur de cette méthode: la clef est aussi longue que le message envoyé! En pratique c'est quasi ingérable.

Chiffrement de Vernam - Sécurité inconditionnelle

- On voit que chacun des bits du message chiffré ne contient aucune information sur le message clair.
- En théorie de l'information cela se traduit en

$$H(M|C) = H(M)$$

où H est l'entropie d'une variable aléatoire.

- L'inconvénient majeur de cette méthode: la clef est aussi longue que le message envoyé! En pratique c'est quasi ingérable.

Plan

Schéma général - chiffrement

Idée : remplacer la suite aléatoire $k_i, i = 0, \ldots, n-1$, dans le chiffrement de Vernam, par une suite *pseudo-aléatoire* générée à partir d'une clef courte K.

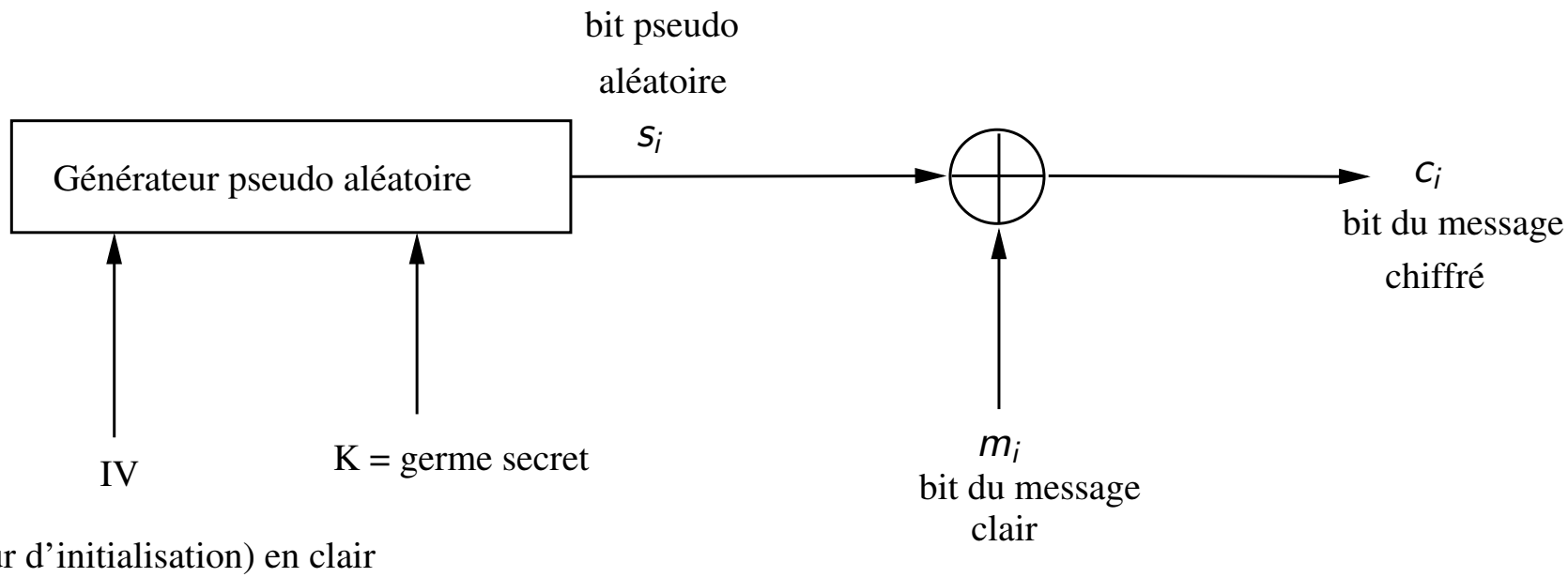
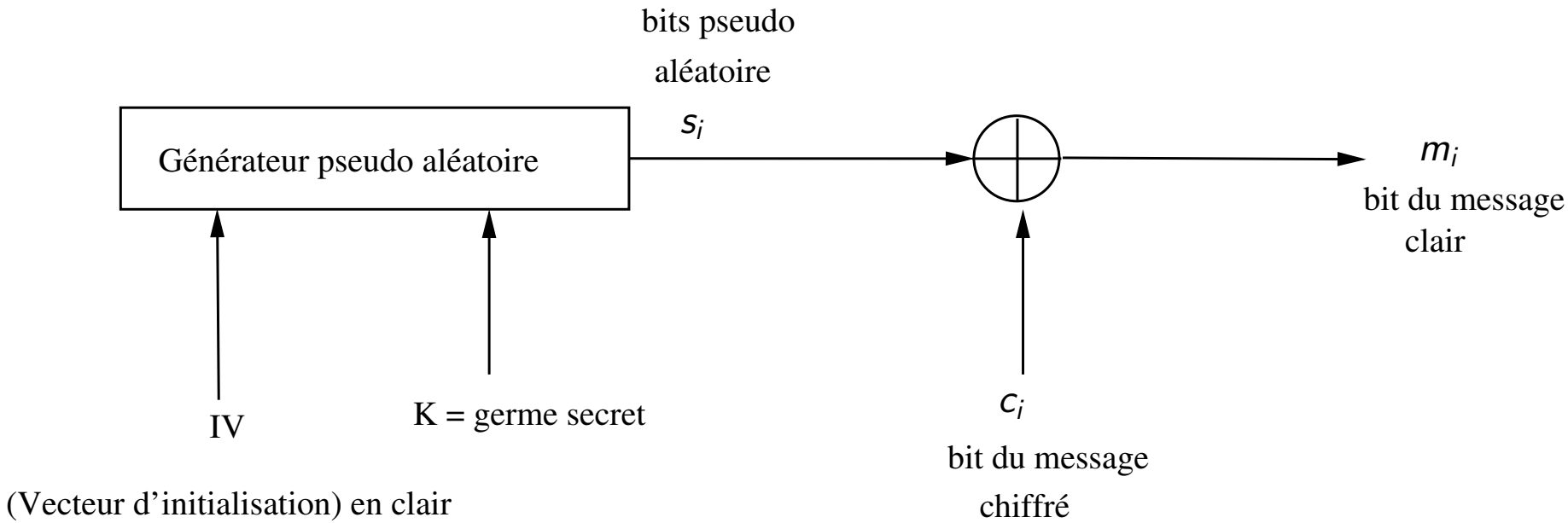


Schéma général - déchiffrement



Critère de Golomb

Soit une suite pseudo-alétoire $s_0, s_1, \ldots, s_{n-1}$:

1. A peu près le même nombre de 0 et de 1

$$\left| \sum_{i=0}^{n-1} (-1)^{s_i} \right| \leq 1$$

2. Une série est une succession de bits identiques entre deux bits opposés. Soit S l'ensemble des séries
   Il y a $|S|/2$ séries de longueur 1.
   Il y a $|S|/4$ séries de longueur 2.
   $\vdots$
   Il y a $|S|/2^{k+1}$ séries de longueur 2^k.

   Et pour chaque longueur de série il y a autant de série de 0 que de 1.

Critère de Golomb (suite)

3. La fonction d'auto-corrélation $C(\tau)$ prend deux valeurs suivant que $\tau = 0$ ou $\tau \neq 0$

$$C(\tau) = \sum_{i=0}^{n-1} (-1)^{s_i + s_{i+\tau}}$$

Générateur aléatoire cryptogaphique

Un générateur cryptographique utilisable pour la cryptographie doit:

- Génèrer des suites de bits satisfaisant les caractéristiques statistiques de suites vraiment aléatoires (critère de Golomb, autres tests statistiques comme le ξ^2, etc).
- Garantir que si un attaquant connait tout ou une partie de la suit chiffrante $s_0, s_1, \ldots, s_i, \ldots$, il est difficile, d'un point de vu quantité de calcul, de trouver la clef K ayant servi de germe.

Plan

Plan

Les registres à décalages linéaires (LFSR)

Une façon économe de construire une suite pseudo-aléatoire utilise une récurrence linéaire

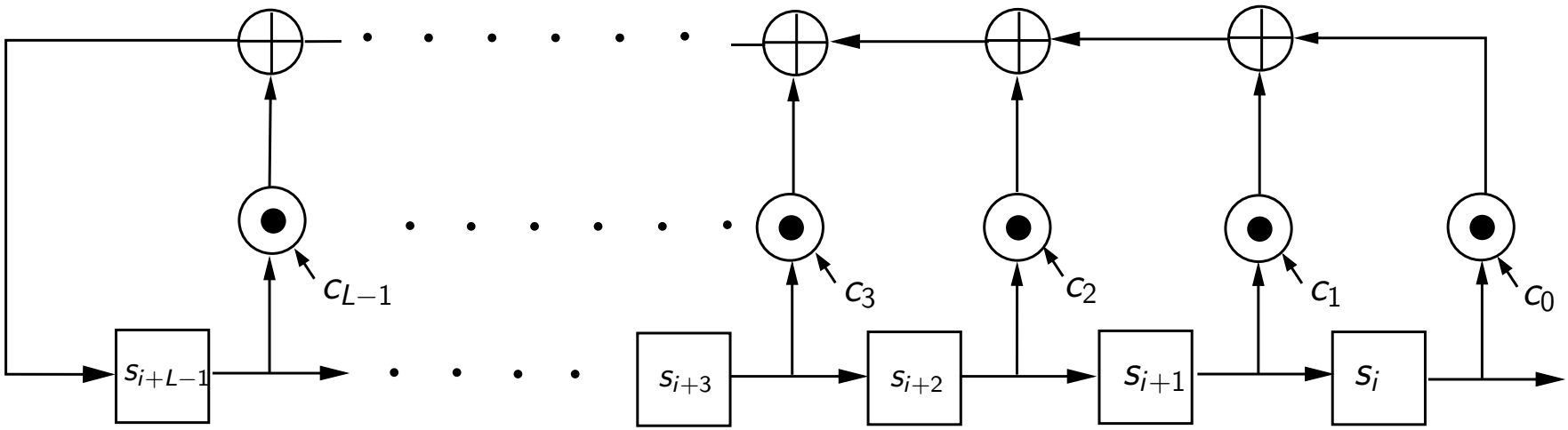
- Les L premiers bits sont $s_0, s_1, \ldots, s_{L-1}$
- Les bits suivants $s_L, s_{L+1}, \ldots, s_i$ se déduisent des L précédants bits grâce à la relation suivante:

$$s_{i+L} = \sum_{j=0}^{L-1} c_j \cdot s_{i+j}$$

Ces suites ont de bonnes propriétés statistiques : elle satisfont par exemple les critères de Golomb.

Les registres à décalages linéaires (LFSR)

Ces suites peuvent être générées avec un circuit séquentiel

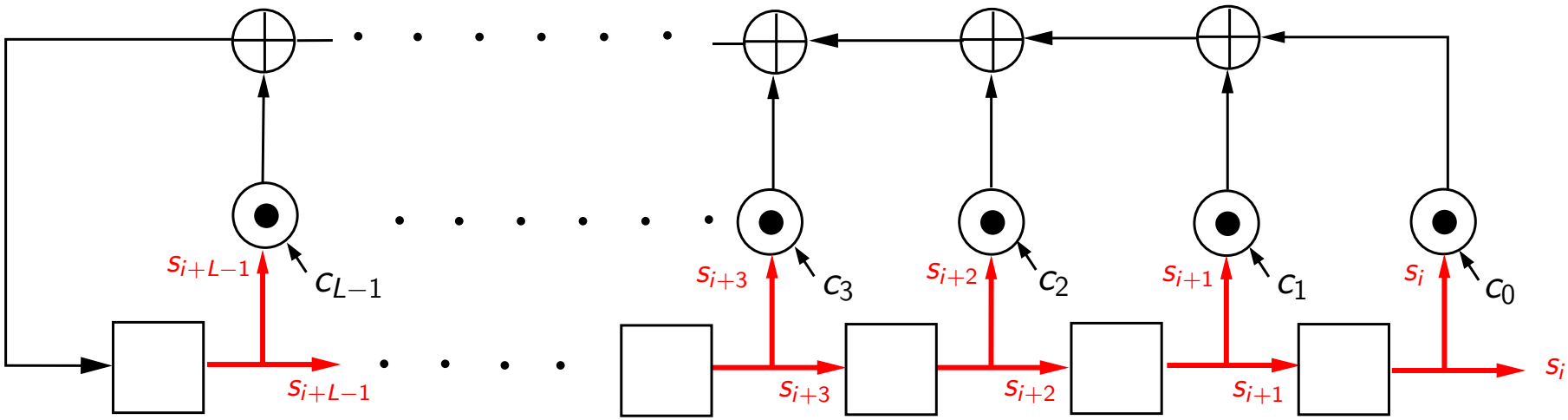


$\oplus$ =XOR, i.e., addition modulo 2

$\odot$ =AND, i.e., multiplication modulo 2

Les registres à décalages linéaires (LFSR)

Ces suites peuvent être générées avec un circuit séquentiel

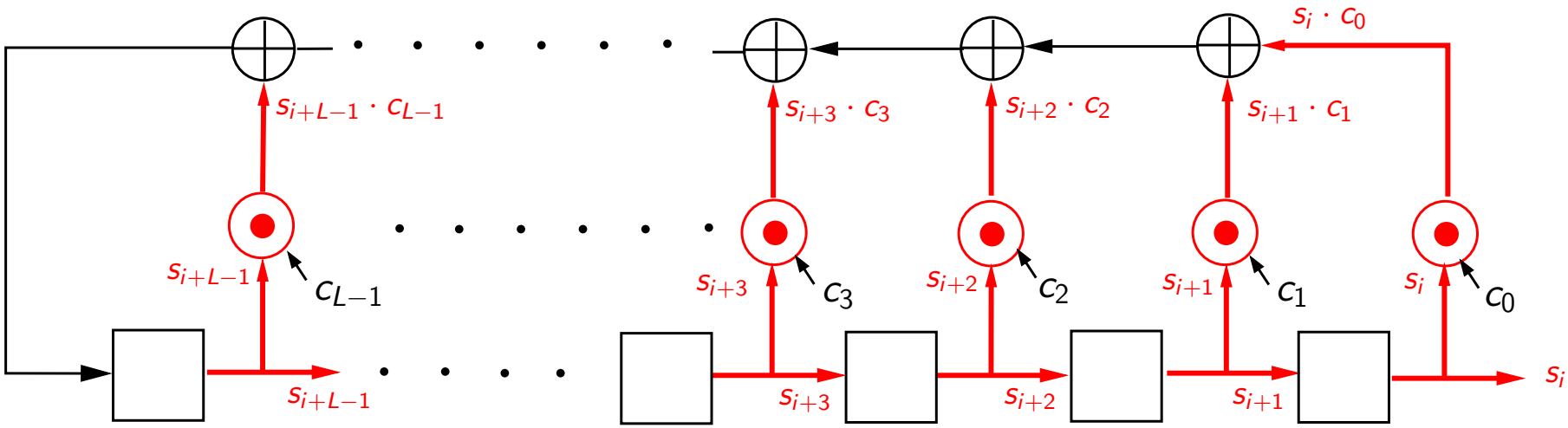


$\oplus$ =XOR, i.e., addition modulo 2

$\odot$ =AND, i.e., multiplication modulo 2

Les registres à décalages linéaires (LFSR)

Ces suites peuvent être générées avec un circuit séquentiel

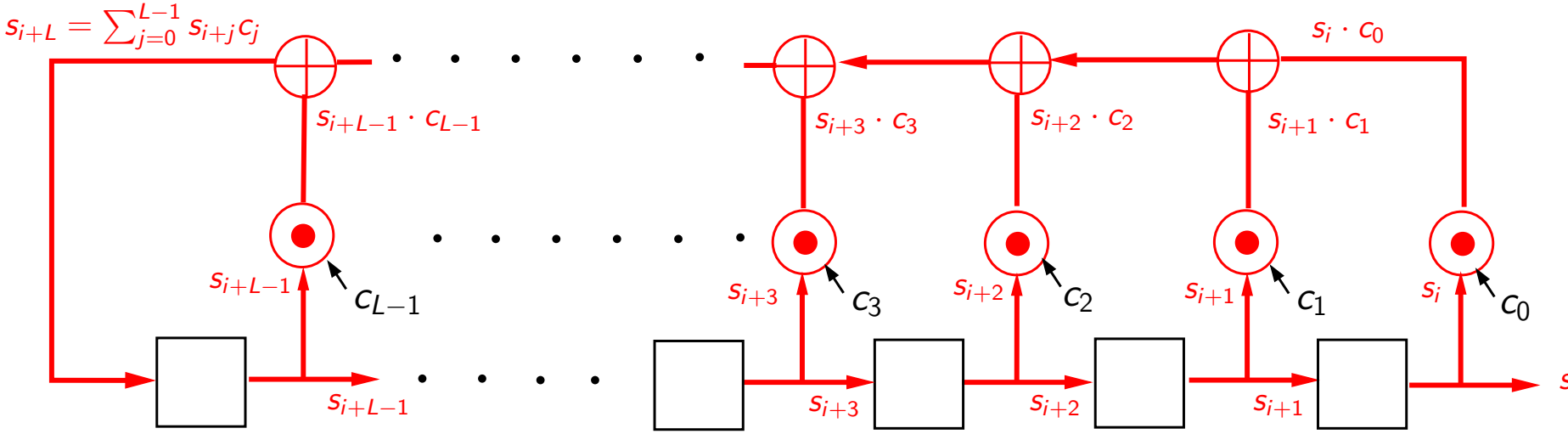


$\oplus$ =XOR, i.e., addition modulo 2

$\odot$ =AND, i.e., multiplication modulo 2

Les registres à décalages linéaires (LFSR)

Ces suites peuvent être générées avec un circuit séquentiel

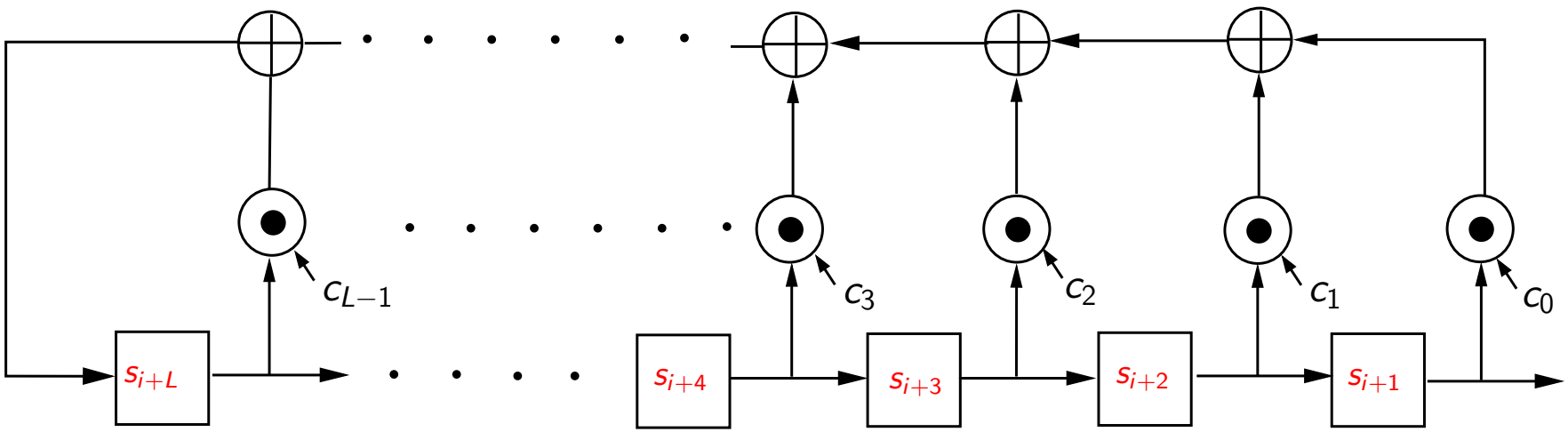


$\oplus$ =XOR, i.e., addition modulo 2

$\odot$ =AND, i.e., multiplication modulo 2

Les registres à décalages linéaires (LFSR)

Ces suites peuvent être générées avec un circuit séquentiel



$\oplus$ =XOR, i.e., addition modulo 2

$\odot$ =AND, i.e., multiplication modulo 2

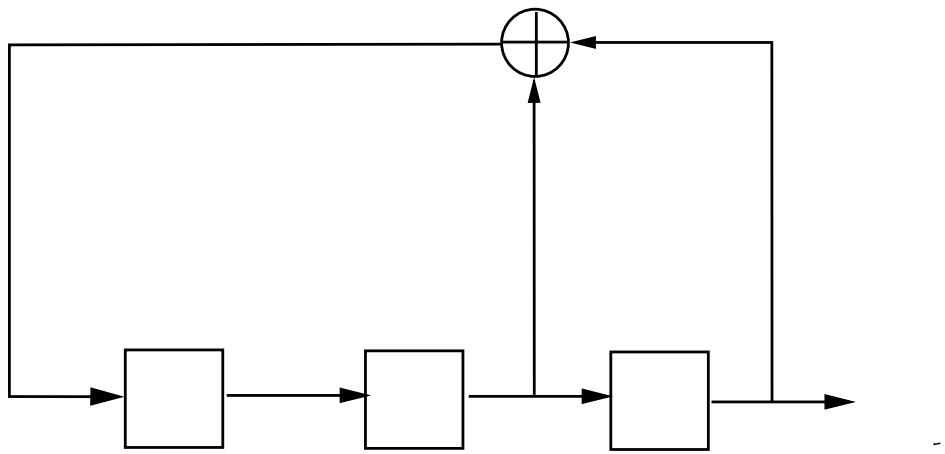
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous.



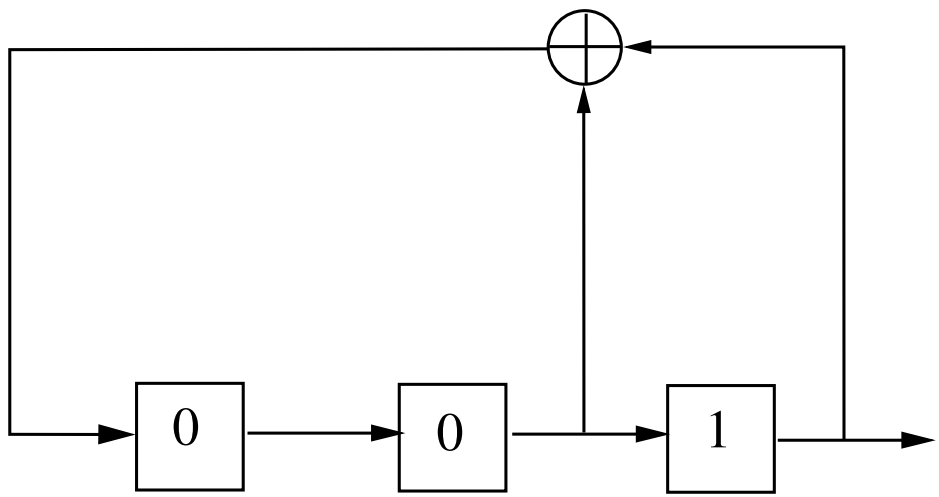
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



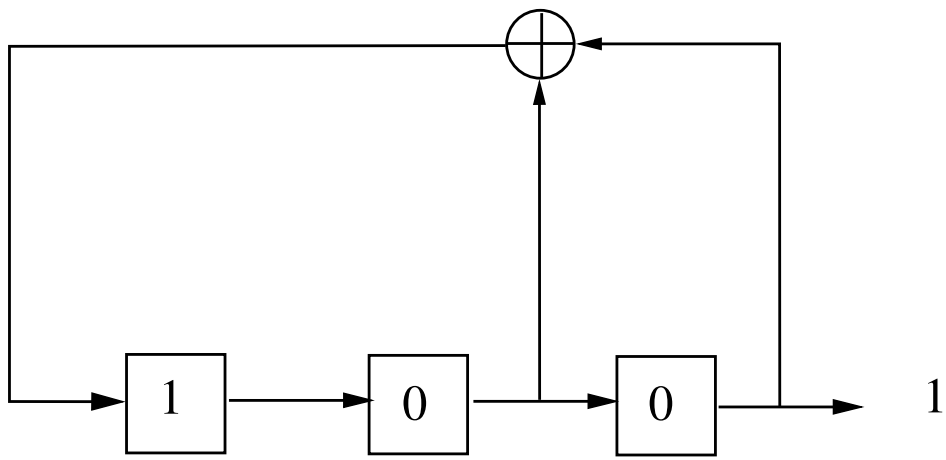
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



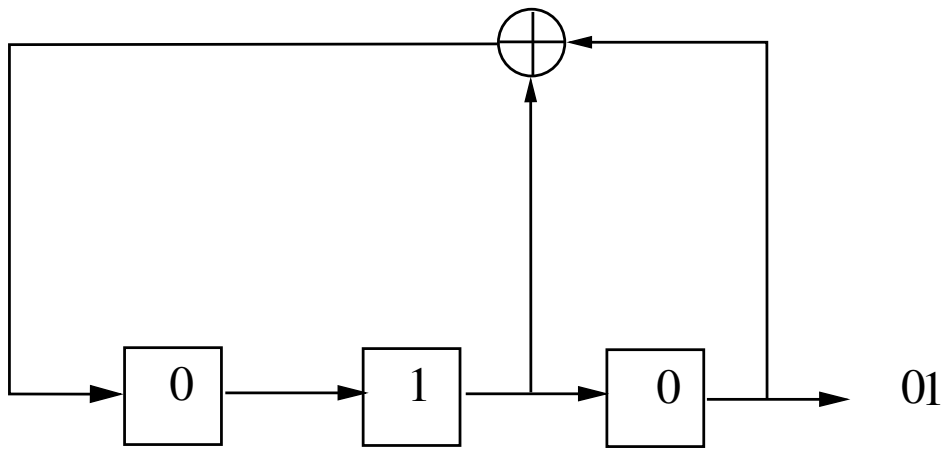
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



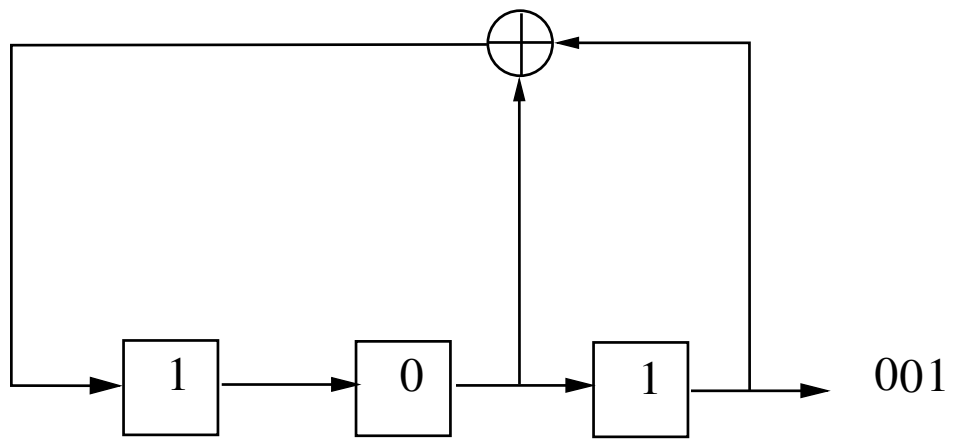
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



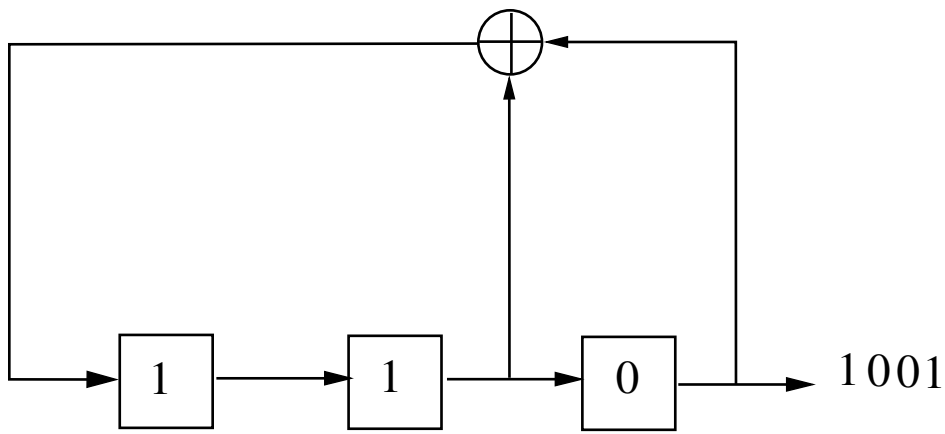
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



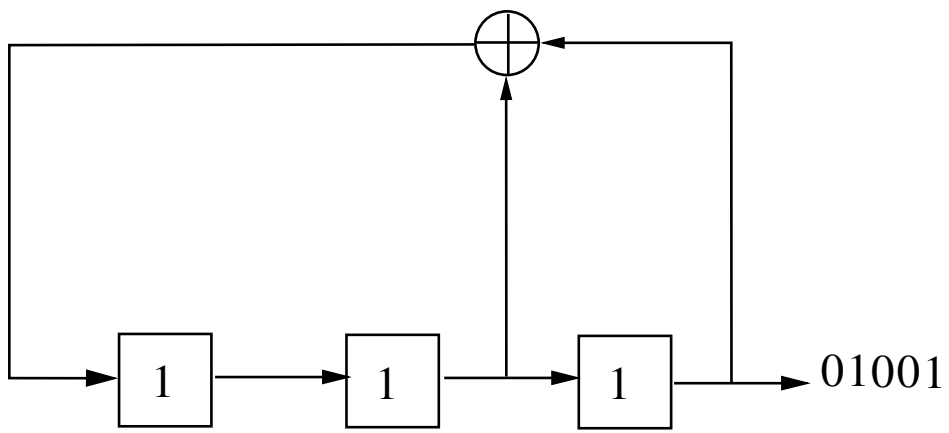
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



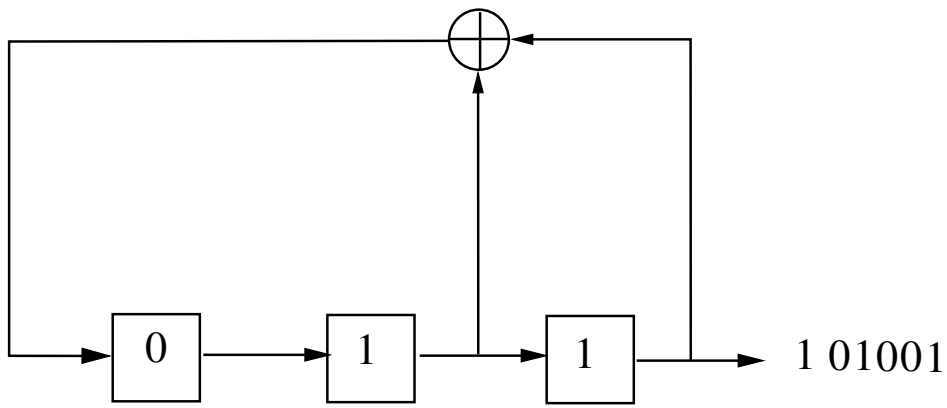
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



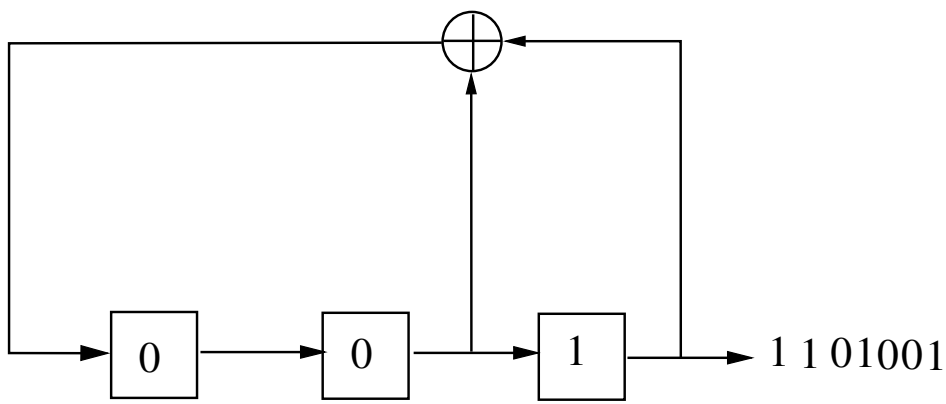
LFSR - Exemple

- On considère un LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

  i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

- Le circuit correspondant est ci-dessous. On l'initialise avec $s_2 = 0, s_1 = 0, s_0 = 1$.



Suite des états des registres

- Les états d'un LFSR satisfont la relation de récurrence:

$$\begin{bmatrix} s_{i+1} \\ s_{i+2} \\ s_{i+3} \\ \vdots \\ s_{i+L-1} \\ s_{i+L} \end{bmatrix}^t = \begin{bmatrix} s_i \\ s_{i+1} \\ s_{i+2} \\ \vdots \\ s_{i+L-2} \\ s_{i+L-1} \end{bmatrix}^t \cdot \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & c_0 \\ 1 & 0 & 0 & \cdots & 0 & c_1 \\ 0 & 1 & 0 & \cdots & 0 & c_2 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & c_{L-2} \\ 0 & 0 & 0 & \cdots & 1 & c_{L-1} \end{bmatrix}.$$

- Si $R_0 = [s_{L-1}, \ldots, s_0]$ est le registre initial, et si A est la matrice $L \times L$ ci-dessus, on a alors

$$R_i = R_0 \cdot A^i.$$

- La matrice A est inversible si et seulement si $c_0 = 1$, alors son déterminant vaut 1.

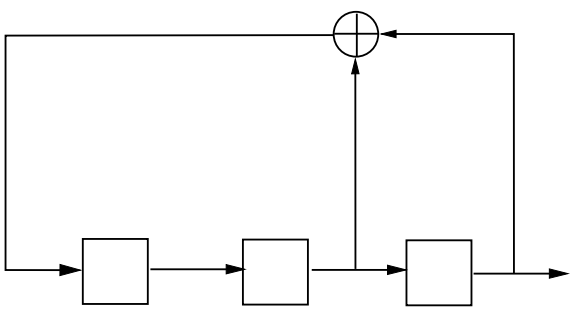
Exemple de suite de registre

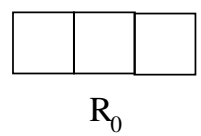
On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante





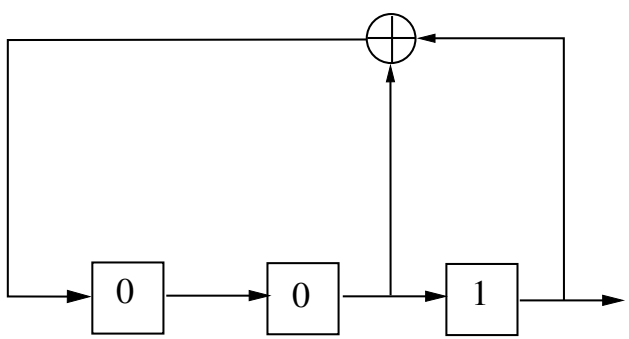
Exemple de suite de registre

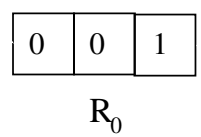
On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante





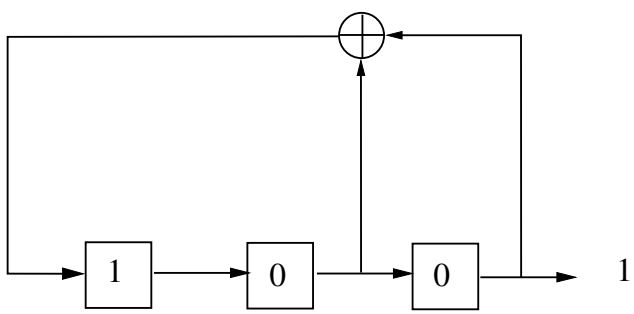
Exemple de suite de registre

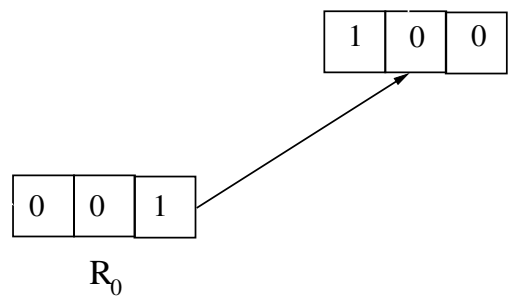
On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante





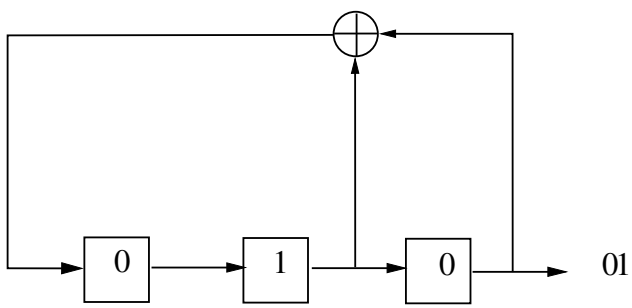
Exemple de suite de registre

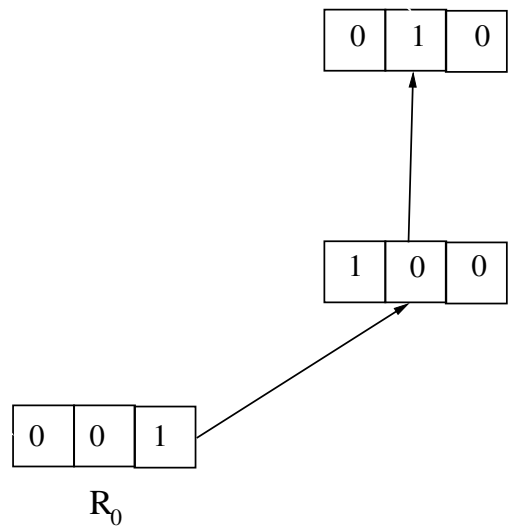
On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante





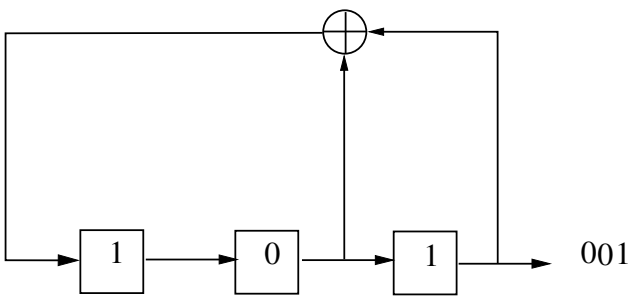
Exemple de suite de registre

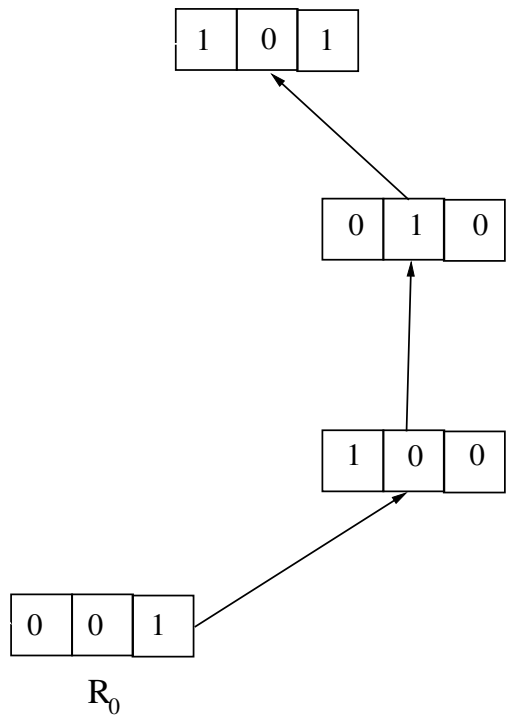
On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante





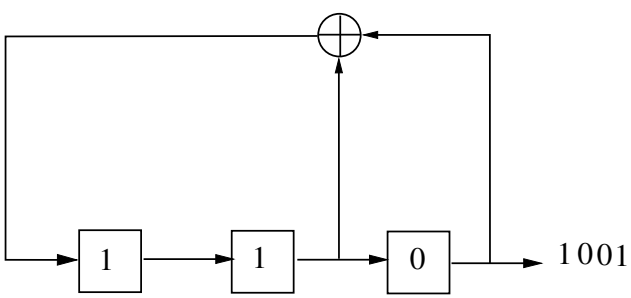
Exemple de suite de registre

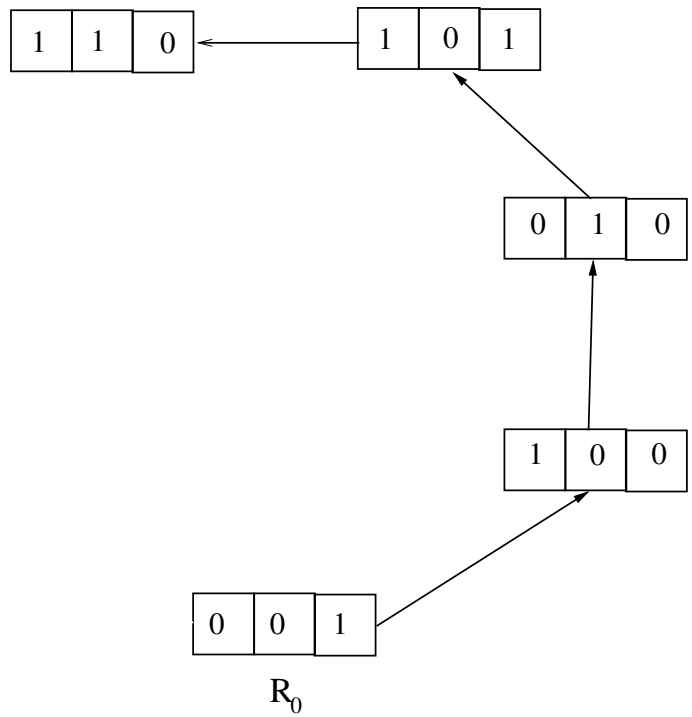
On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante





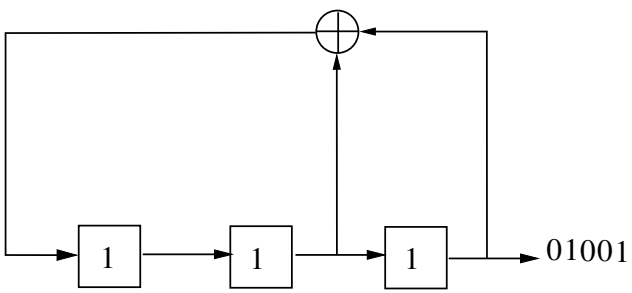
Exemple de suite de registre

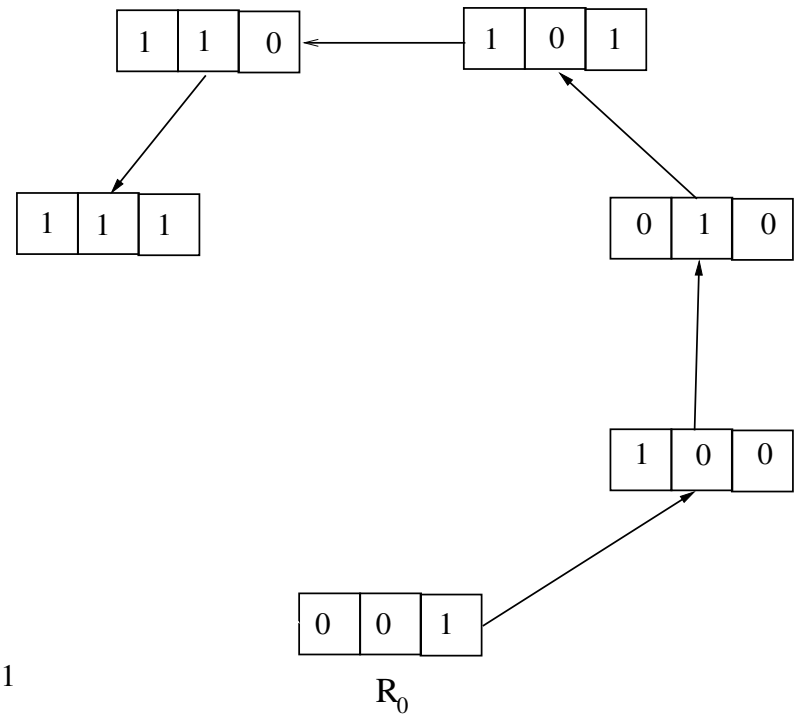
On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante





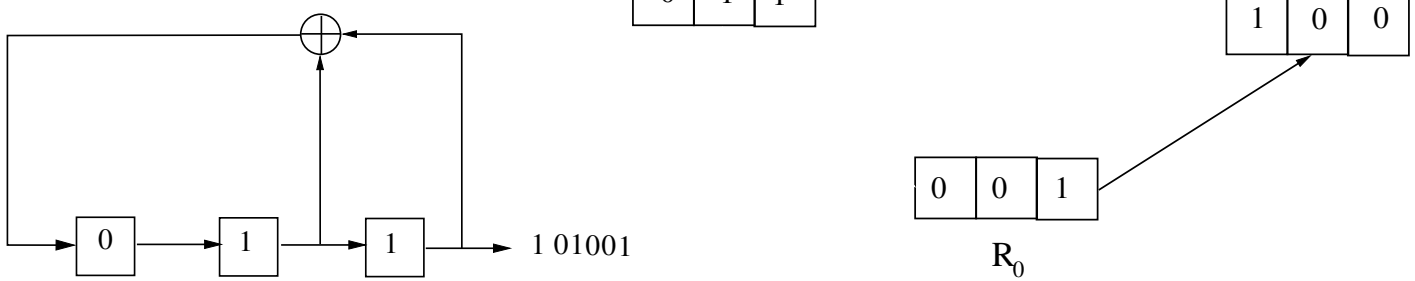
Exemple de suite de registre

On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante



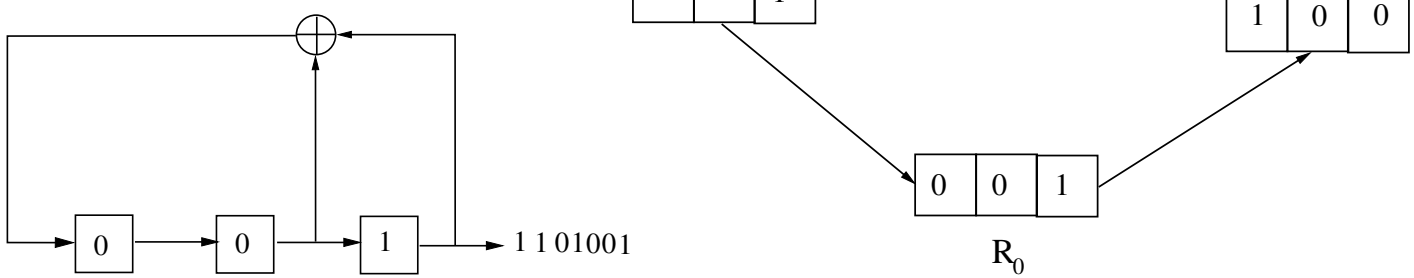
Exemple de suite de registre

On reprend le LFSR de longueur $L = 3$ avec la relation de récurrence

$$s_{i+3} = s_{i+1} + s_i$$

i.e. $c_0 = 1, c_1 = 1$ et $c_2 = 0$.

La suite d'état est la suivante



Suite ultimement périodique

Definition

Une suite $s_0, s_1, s_2, \ldots$ est dite ultimement périodique de type (u, r) si on

$$s_{i+r} = s_i \text{ pour tout } i \geq u$$

Si $u = 0$ on dit que s est périodique.

Suite ultimement périodique

Definition

Une suite $s_0, s_1, s_2, \ldots$ est dite ultimement périodique de type (u, r) si on

$$s_{i+r} = s_i \text{ pour tout } i \geq u$$

Si $u = 0$ on dit que s est périodique.

Exemple :

- La suite

$$00\,\underbrace{011}\,\overbrace{011}\,\underbrace{011}\,\overbrace{011}\ldots$$

  est ultimement périodique de type $(u = 2, r = 3)$.
- On peut vérifier qu'elle est engendrée par un LFSR de taille $L = 4$ et de récurrence $x_{i+4} = x_{i+3} + x_{i+2}$.

Borne sur la périodicité d'un LFSR

Theorem

Soit $s_0, s_1, s_2, \ldots$ un suite engendré par un LFSR de taille L alors elle est ultimement périodique de période $< (2^L - 1)$

Proof.

La suite des registes $R_0, R_1 \ldots, R_i, \ldots$ prends ses valeurs dans $\{0,1\}^L$ qui contient 2^L éléments.

- Si pour un $i \geq 0$ on a $R_i = 0$ alors pour tout $j \geq i$ on a aussi $R_j = 0$ et la suite est donc bien ultimement périodique de type $(i, 1)$.

- Sinon la suite de registre n'atteint jamais 0: donc pour $i < j <= 2^L - 1$ on a $R_i = R_j$, mais alors

$$R_{i+k} = A^k \cdot R_i = A^k \cdot R_j = R_{j+k}$$

et donc la suite est ultimement périodique de période $j - i <= 2^L - 1$.

LFSR de période maximale $(2^L - 1)$

Proposition

Soit un LFSR de taille L, on définit son polynome de rétroaction $f(x)$ *par*

$$f(x) = x^L + \sum_{i=0}^{L-1} c_i x^i$$

Alors le LFSR engendre des suites pseudo-aléatoire de période $2^L - 1$ *si et seulement si* $f(x)$ *est irréductible et primitif.*

Remarque

Pour un polynome $f(x)$ *irreductible:*
$f(x)$ *est primitif* $\iff$ *si* $x^i \mod f(x) \neq 1$ *pour* $i = 1, \ldots, 2^L - 2$.

Preuve de la proposition

On ne donne qu'une esquisse rapide de la preuve:

- Soit A la matrice $L \times L$ correspondant au LFSR alors $s_i, i = 0, 1, \ldots,$ est périodique de période $2^L - 1$ si et seulement si $A^i \neq Id$ pour $i < 2^L - 1$.
- On peut montrer que la matrice A est la matrice qui correspond à la multiplication par x modulo $f(x)$

$$x \cdot R(x) \mod (f(x)).$$

- La matrice A^i est la matrice de multiplication par x^i modulo $f(x)$
- Finalement $A^i \neq Id$ pour $1 \leq i < 2^L - 1$ équivaut à $x^i \neq 1$ pour $1 \leq i < 2^L - 1$.

Exemple : A est une matrice de multiplication

- Nous considerons le LFSR with $c_0 = 1, c_1 = 1$ et $c_2 = 0$, son polynome de rétroaction $f(x) = x^3 + x + 1$
- La matrice A est ici:

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

- On remarque que si $R = r_0 + r_1 x + r_2 x$ on a

$$A \cdot \begin{bmatrix} r_0 \\ r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} r_2 \\ r_0 + r_2 \\ r_1 \end{bmatrix}$$

qui sont biens les coefficients de

$$\begin{aligned} x \cdot R \mod f(x) &= (r_0 x + r_1 x^2 + r_2 x^3) \mod f(x) \\ &= r_2 + (r_0 + r_2)x + r_1 x^2 \end{aligned}$$

Exemple : A^2 est une matrice de multiplication

On calcule le carré de A

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \quad A^2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}.$$

Et on compare

$$A^2 \cdot \begin{bmatrix} r_0 \\ r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} r_1 \\ r_1 + r_2 \\ r_0 + r_1 \end{bmatrix}$$

avec

$$\begin{aligned} x^2 R \mod f(x) &= (r_0 x^2 + r_1 x^3 + r_2 x^4) \mod f(x) \\ &= r_1 + (r_2 + r_1)x + (r_0 + r_2)x^2 \end{aligned}$$

Exemple : LFSR de période maximale

- On prend $L = 4$ et le polynome de rétroaction $f(x) = x^4 + x + 1$.
- $f(x)$ est primitif: on calcule les puissance de x modulo $f(x)$

x^i	$x^i \mod f(x)$
1	1
x	x
x^2	x^2
x^3	x^3

x^i	$x^i \mod f(x)$
x^4	$x+1$
x^5	x^2+x
x^6	x^3+x^2
x^7	x^3+x+1

x^i	$x^i \mod f(x)$
x^8	x^2+1
x^9	x^3+x
x^{10}	x^2+x+1
x^{11}	x^3+x^2+x

x^i	$x^i \mod f(x)$
x^{12}	x^3+x^2+x+1
x^{13}	x^3+x^2+1
x^{14}	x^3+1
x^{15}	1

- La période est maximale: on calcule tous les état du registre jusqu'à obtenir un cycle:

Un LFSR n'est pas cryptographiquement sûr

- On suppose que l'on connait $2L$ bits consécutif de s

  $s_i, s_{i+1}, s_{i+2}, \ldots, s_{i+2L-1}$
- On peut calculer les c_i en résolvant le système

$$\begin{bmatrix} s_i & s_{i+1} & \cdots & s_{i+L-1} & s_{i+L-2} \\ s_{i+1} & s_{i+2} & \cdots & s_{i+L} & s_{i+L-1} \\ \vdots & \vdots & & \vdots & \vdots \\ s_{i+L-2} & s_{i+L-1} & \cdots & s_{i+2L-3} & s_{i+2L-4} \\ s_{i+L-1} & s_{i+L} & \cdots & s_{i+2L-2} & s_{i+2L-3} \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{L-2} \\ c_{L-1} \end{bmatrix} = \begin{bmatrix} s_{i+L} \\ s_{i+L+1} \\ \vdots \\ s_{i+2L+-2} \\ s_{i+2L-1} \end{bmatrix}$$

- On peut alors en déduire la matrice A
- Si $c_0 = 1$, la matrice A est inversible et on peut calculer en arrière $R_{i-k} = R_i \cdot A^{-k}$ jusqu'au germe (la clef).

Un LFSR n'est pas cryptographiquement sûr - Exemple

- On suppose que l'connaît une partie d'une suite pseudo-aléatoire générée par un LFSR de taille $L = 4$.

$$????\,????\,????\,0111\,1011$$

  Les ? sont les bits inconnus.

- Avec les $2 \cdot 4$ bits connus on peut calculer les coefficients générateurs c_0, c_1, c_2, c_3

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

.

Un LFSR n'est pas cryptographiquement sûr - Exemple

- On suppose que l'connaît une partie d'une suite pseudo-aléatoire générée par un LFSR de taille $L = 4$.

$$????\,????\,????\,0111\,1011$$

  Les ? sont les bits inconnus.

- Avec les $2 \cdot 4$ bits connus on peut calculer les coefficients générateurs c_0, c_1, c_2, c_3

$$\begin{bmatrix} 0&1&1&1 \\ 1&1&1&1 \\ 1&1&1&0 \\ 1&1&0&1 \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

.

Un LFSR n'est pas cryptographiquement sûr - Exemple

- On suppose que l'connaît une partie d'une suite pseudo-aléatoire générée par un LFSR de taille $L = 4$.

$$????\,????\,????\,0111\,1011$$

  Les ? sont les bits inconnus.

- Avec les $2 \cdot 4$ bits connus on peut calculer les coefficients générateurs c_0, c_1, c_2, c_3

$$\begin{bmatrix} 0&1&1&1 \\ 1&1&1&1 \\ 1&1&1&0 \\ 1&1&0&1 \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

.

Un LFSR n'est pas cryptographiquement sûr - Exemple

- On suppose que l'connaît une partie d'une suite pseudo-aléatoire générée par un LFSR de taille $L = 4$.

$$????\,????\,????\,0111\,1011$$

  Les ? sont les bits inconnus.

- Avec les $2 \cdot 4$ bits connus on peut calculer les coefficients générateurs c_0, c_1, c_2, c_3

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

.

Un LFSR n'est pas cryptographiquement sûr - Exemple

- On suppose que l'connaît une partie d'une suite pseudo-aléatoire générée par un LFSR de taille $L = 4$.

$$????\,????\,????\,0111\,1011$$

  Les ? sont les bits inconnus.

- Avec les $2 \cdot 4$ bits connus on peut calculer les coefficients générateurs c_0, c_1, c_2, c_3

$$\begin{bmatrix} 0&1&1&1 \\ 1&1&1&1 \\ 1&1&1&0 \\ 1&1&0&1 \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

.

Un LFSR n'est pas cryptographiquement sûr - Exemple

- On suppose que l'connaît une partie d'une suite pseudo-aléatoire générée par un LFSR de taille $L = 4$.

$$????\,????\,????\,0111\,1011$$

  Les ? sont les bits inconnus.

- Avec les $2 \cdot 4$ bits connus on peut calculer les coefficients générateurs c_0, c_1, c_2, c_3

$$\begin{bmatrix} 0&1&1&1 \\ 1&1&1&1 \\ 1&1&1&0 \\ 1&1&0&1 \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1&0&1&1 \\ 0&1&1&1 \\ 1&1&1&0 \\ 1&1&0&1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}.$$

Donne $c_0 = 1,\ c_1 = 1,\ c_2 = 1,\ c_3 = 1.$

Un LFSR n'est pas cryptographiquement sûr - Exemple

On cherche à compléter la suite ???? ???? ???? 0111 1011 généré avec $c_0 = 1,\ c_1 = 1,\ c_2 = 1,\ c_3 = 1$.

- On en déduit la matrice A et la matrice A^{-1}

$$A = \begin{bmatrix} 0\,0\,0\,1 \\ 1\,0\,0\,1 \\ 0\,1\,0\,1 \\ 0\,0\,1\,1 \end{bmatrix}^{-1} \quad \text{et } A^{-1} \begin{bmatrix} 1\,1\,0\,0 \\ 1\,0\,1\,0 \\ 1\,0\,0\,1 \\ 1\,0\,0\,0 \end{bmatrix}^{-1}$$

Un LFSR n'est pas cryptographiquement sûr - Exemple

On cherche à compléter la suite ???? ???? ???? 0111 1011 généré avec $c_0 = 1,\ c_1 = 1,\ c_2 = 1,\ c_3 = 1$.

- On en déduit la matrice A et la matrice A^{-1}

$$A = \begin{bmatrix} 0\,0\,0\,1 \\ 1\,0\,0\,1 \\ 0\,1\,0\,1 \\ 0\,0\,1\,1 \end{bmatrix}^{-1} \quad \text{et } A^{-1} \begin{bmatrix} 1\,1\,0\,0 \\ 1\,0\,1\,0 \\ 1\,0\,0\,1 \\ 1\,0\,0\,0 \end{bmatrix}^{-1}$$

- On obtient le début de la suite pseudo aléatoire:

$$[0,1,1,1] \cdot A^{-1} \quad = \quad [1,0,1,1]$$

Un LFSR n'est pas cryptographiquement sûr - Exemple

On cherche à compléter la suite ???? ???? ???? 0111 1011 généré avec $c_0 = 1,\ c_1 = 1,\ c_2 = 1,\ c_3 = 1$.

- On en déduit la matrice A et la matrice A^{-1}

$$A = \begin{bmatrix} 0&0&0&1 \\ 1&0&0&1 \\ 0&1&0&1 \\ 0&0&1&1 \end{bmatrix}^{-1} \quad \text{et } A^{-1} \begin{bmatrix} 1&1&0&0 \\ 1&0&1&0 \\ 1&0&0&1 \\ 1&0&0&0 \end{bmatrix}^{-1}$$

- On obtient le début de la suite pseudo aléatoire:

$$\begin{aligned} [0,1,1,1] \cdot A^{-1} &= [1,0,1,1] \\ [1,0,1,1] \cdot A^{-1} &= [1,1,0,1] \end{aligned}$$

Un LFSR n'est pas cryptographiquement sûr - Exemple

On cherche à compléter la suite ???? ???? ???? 0111 1011 généré avec $c_0 = 1,\ c_1 = 1,\ c_2 = 1,\ c_3 = 1$.

- On en déduit la matrice A et la matrice A^{-1}

$$A = \begin{bmatrix} 0\,0\,0\,1 \\ 1\,0\,0\,1 \\ 0\,1\,0\,1 \\ 0\,0\,1\,1 \end{bmatrix}^{-1} \quad \text{et } A^{-1} \begin{bmatrix} 1\,1\,0\,0 \\ 1\,0\,1\,0 \\ 1\,0\,0\,1 \\ 1\,0\,0\,0 \end{bmatrix}^{-1}$$

- On obtient le début de la suite pseudo aléatoire:

$$\begin{aligned} [0,1,1,1] \cdot A^{-1} &= [1,0,1,1] \\ [1,0,1,1] \cdot A^{-1} &= [1,1,0,1] \\ [1,1,0,1] \cdot A^{-1} &= [1,1,1,0] \end{aligned}$$

Un LFSR n'est pas cryptographiquement sûr - Exemple

On cherche à compléter la suite ???? ???? ???? 0111 1011 généré avec $c_0 = 1,\ c_1 = 1,\ c_2 = 1,\ c_3 = 1$.

- On en déduit la matrice A et la matrice A^{-1}

$$A = \begin{bmatrix} 0&0&0&1 \\ 1&0&0&1 \\ 0&1&0&1 \\ 0&0&1&1 \end{bmatrix}^{-1} \quad \text{et } A^{-1} \begin{bmatrix} 1&1&0&0 \\ 1&0&1&0 \\ 1&0&0&1 \\ 1&0&0&0 \end{bmatrix}^{-1}$$

- On obtient le début de la suite pseudo aléatoire:

$$\begin{aligned} [0,1,1,1] \cdot A^{-1} &= [1,0,1,1] \\ [1,0,1,1] \cdot A^{-1} &= [1,1,0,1] \\ [1,1,0,1] \cdot A^{-1} &= [1,1,1,0] \\ [1,1,0,1] \cdot A^{-1} &= [1,1,1,0] \end{aligned}$$

Un LFSR n'est pas cryptographiquement sûr - Exemple

On cherche à compléter la suite ???? ???? ???? 0111 1011 généré avec $c_0 = 1,\ c_1 = 1,\ c_2 = 1,\ c_3 = 1$.

- On en déduit la matrice A et la matrice A^{-1}

$$A = \begin{bmatrix} 0&0&0&1 \\ 1&0&0&1 \\ 0&1&0&1 \\ 0&0&1&1 \end{bmatrix}^{-1} \quad \text{et } A^{-1} \begin{bmatrix} 1&1&0&0 \\ 1&0&1&0 \\ 1&0&0&1 \\ 1&0&0&0 \end{bmatrix}^{-1}$$

- On obtient le début de la suite pseudo aléatoire:

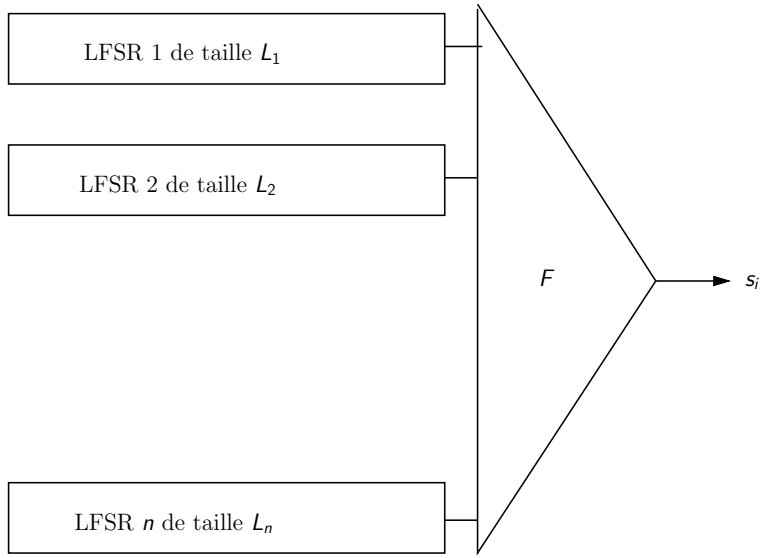
$$\begin{aligned} [0,1,1,1] \cdot A^{-1} &= [1,0,1,1] \\ [1,0,1,1] \cdot A^{-1} &= [1,1,0,1] \\ [1,1,0,1] \cdot A^{-1} &= [1,1,1,0] \\ [1,1,0,1] \cdot A^{-1} &= [1,1,1,0] \end{aligned}$$

Ce qui donne les 4 bits précédents :

???? 1111 0111 1011.

Plan

Fonction booléenne



- La fonction $F(x_1, \ldots, x_n)$ est une fonction booléenne

$$F : \{0,1\}^n \to \{0,1\}.$$

  donné souvent par explement par une table de vérité.
- Par exemple pour $n = 3$

$x_1x_2x_3$	000	001	010	011	100	101	110	111
$F(x_1, x_2, x_3)$	1	0	0	1	0	0	1	0

- La période de s_i satisfait $ppcm(r_1, \ldots, r_n)$ où r_i est la période du LFSR i.

Fonction booléenne

- Monôme: soit $b = (b_1, \ldots, b_n) \in \{0,1\}^n$ considérons un monôme

$$M_b(x_1, \ldots, x_n) = \prod_{i=1}^{n} (x_i + b_i + 1)$$

Ce mônome satisfait, pour $(\alpha_1, \ldots, \alpha_n)$ $in\{0,1\}^n$

$$M_b(\alpha_1, \ldots, \alpha_n) = \begin{cases} 0 \text{ si } (\alpha_1, \ldots, \alpha_n) \neq (b_1, \ldots, b_n) \\ 1 \text{ si } (\alpha_1, \ldots, \alpha_n) = (b_1, \ldots, b_n) \end{cases}$$

- Exemple : pour $n = 3$ et $b = (0,1,1)$ la table de vérité de $M_b = (x_1 + 1)x_2x_3$ est

$x_1x_2x_3$	000	001	010	011	100	101	110	111
$M_b(x_1, x_2, x_3)$	0	0	0	1	0	0	0	0

Forme normale algébrique

Décomposition : en sommant les monome correspondant au $b = (b_1, \ldots, b_n)$ tel que $F(b_1, \ldots, b_n) = 1$ et en développant chaque monome, on obtient la forme algébrique de f

$$F = \sum_{u \in \{0,1\}^n} F_u \prod_{i=1}^{n} x_i^{u_i} \text{ pour tout } u \; F_u \in \{0, 1\}$$

Cette expression est la *Forme Normale Algebrique* de f.

Exemple : forme normale algébrique

Exemple : pour $n = 3$ et f donnée par

$x_1x_2x_3$	000	001	010	011	100	101	110	111
$f(x_1, x_2, x_3)$	1	0	0	1	0	0	1	0

on a

$$F(x_1, x_2, x_3) = \overbrace{(x_1+1)(x_2+1)(x_3+1)}^{M_{000}} + \overbrace{x_1(x_2+1)(x_3+1)}^{M_{100}} + \underbrace{x_1x_2(x_3+1)}_{M_{110}}.$$

qui se simplifie en $F(x) = x_1x_2x_3 + x_1x_2 + x_2x_3 + x_2 + x_3 + 1$

Caractéristique requise pour les fonctions booléennes

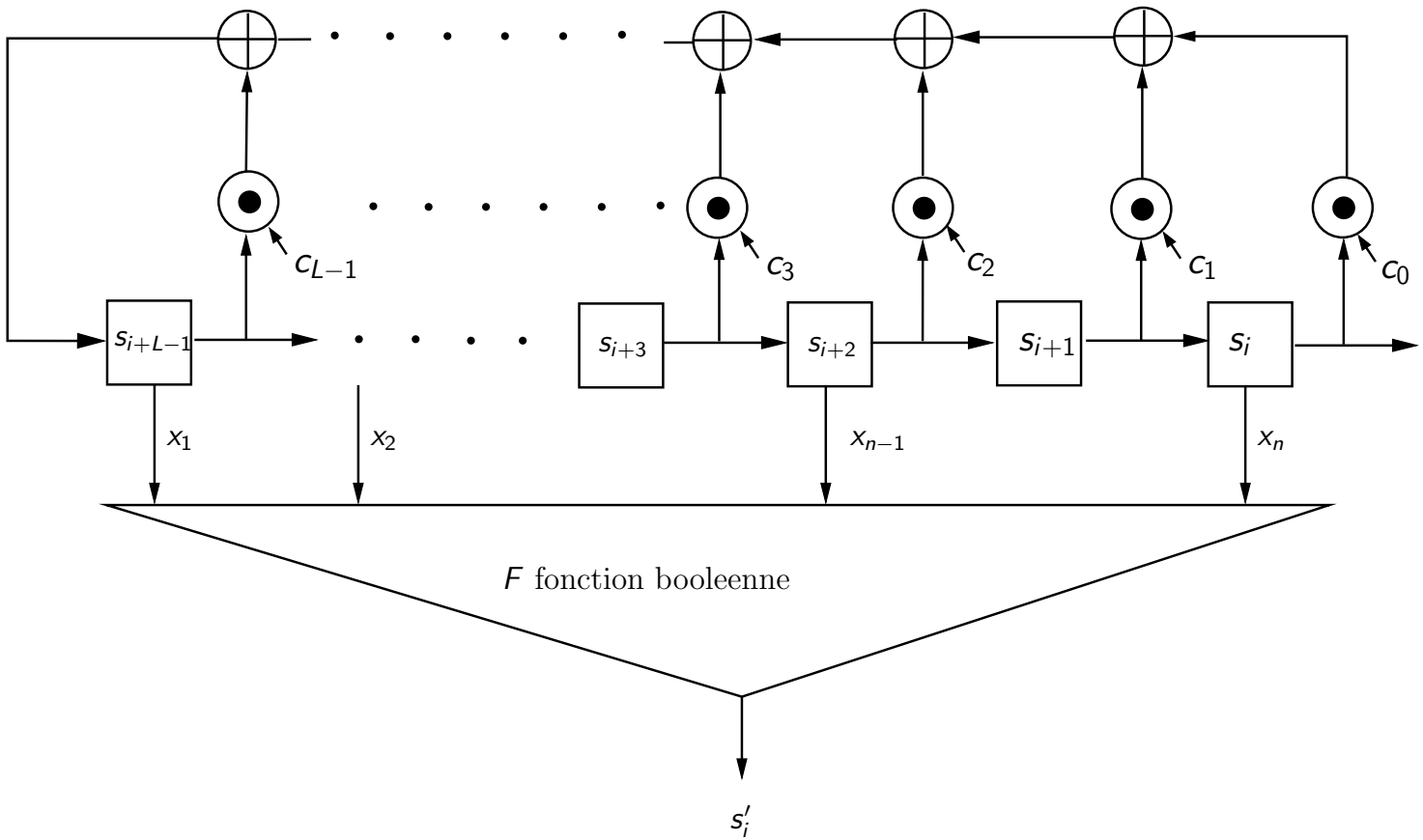
- Insensibilité aux attaques algébriques.
  - Les bits générés peuvent être mis sous forme d'équation en les inconnues $s_0, \ldots, s_L$.
  - Pour F de grand degré et "dense", ces equations sont aussi de grand degré et donc difficile à résoudre.

- Insensibilité aux attaques par corrélation.
  - Pour éviter les attaques par corrélation, F doit satisfaire des propriétés d'équilibre: elle doit dépendre *de la même manière* de chacun des n LFSR.

Plan

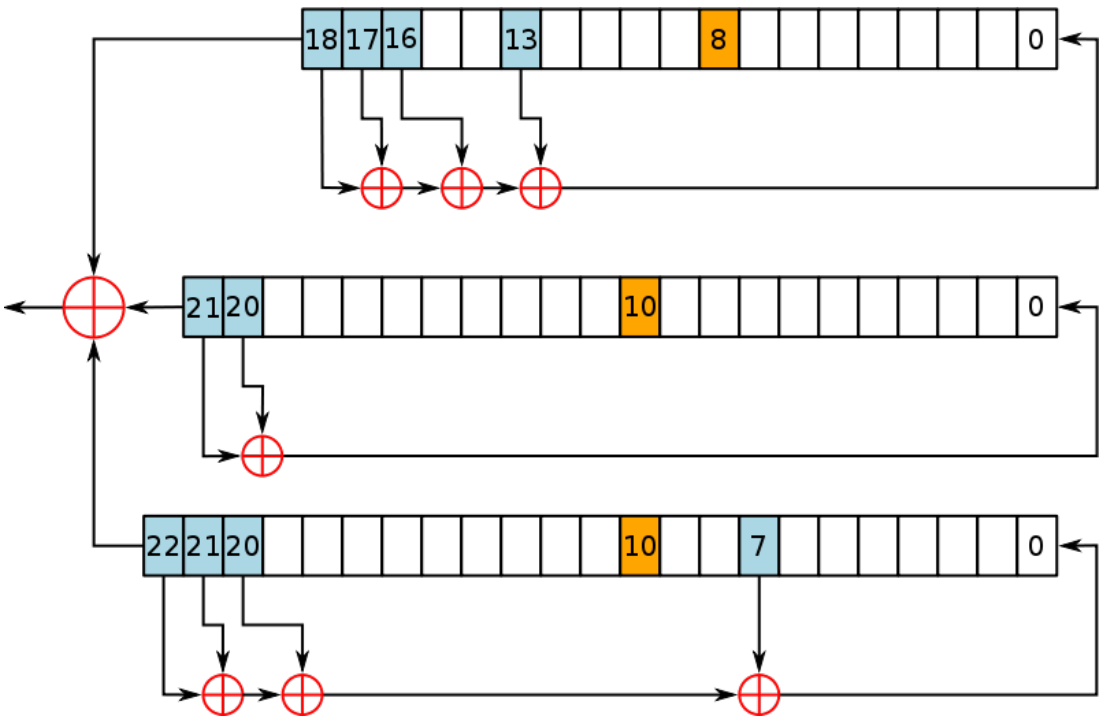
LFSR filtré

- Le générateurs contient un seul LFSR.
- Le bit générés est combiné par une fonction booléenne d'un sous ensemble de bits du registre.



Générateur avec contrôle d'horloge - algorithme A5/1

- Introduction d'irrégularité dans la mise à jour des LFSR.
- On contrôle le bit d'horloge d'un LFSR par un ou plusieurs bits des autres LFSRs.



A chaque cloc de l'horloge les registre sont shiftés, suivant les valeur de bits oranges.

Plan

Plan

RC4 - présentation

RC4 = (Rivest Cipher 4) est composé de 2 algorithmes

- L'algorithme KSA (Key schedule algorithm) qui initialise/randomise une fonction bijective $S : \{0, \ldots, N-1\} \to \{0, \ldots, N-1\}$.
  - En pratique $N = 256$.
- L'algorithme PRGA (Pseudo random generator algorithm) génere une suite aléatoire d'octet
  - L'octet aléatoire généré est un $S[j_i]$ et l'octect chiffré $c_i = m_i \oplus S[j_i]$.
  - L'indice j_i et la fonction S est mise à jour.

Remarque

C'est un algorithme proche d'Enigma : chiffrement par une substitution modifiée pour chaque nouveau caractère chiffré.

RC4 - KSA

L'algorithme de key schedule (KSA) initialise *aléatoirement* la fonction S.

```
KSA
Entrée : deux tableaux d'octet clef  et IV
Sortie : Une fonction S : [0,N-1] -> [0,N-1]
K := [IV | clef ]
L := longueur(K)
pour i de 0 à N
    S[i] := i
finpour
j := 0
pour i de 0 à 255
  j := (j + S[i] + K[i mod L]) mod 256
  échanger(S[i], S[j])
finpour
```

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0 + 0 + 4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0 + 0 + 4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$
1	4			3	0	5	6	7	$4 + 0 + 6 \mod 8 = 2$	$S[1] \leftrightarrow S[2]$

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0 + 0 + 4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$
1	4			3	0	5	6	7	$4 + 0 + 6 \mod 8 = 2$	$S[1] \leftrightarrow S[2]$
2	4	2		3	0		6	7	$2 + 1 + 2 \mod 8 = 5$	$S[2] \leftrightarrow S[5]$

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0 + 0 + 4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$
1	4			3	0	5	6	7	$4 + 0 + 6 \mod 8 = 2$	$S[1] \leftrightarrow S[2]$
2	4	2		3	0		6	7	$2 + 1 + 2 \mod 8 = 5$	$S[2] \leftrightarrow S[5]$
3		2	5		0	1	6	7	$5 + 1 + 4 \mod 8 = 1$	$S[3] \leftrightarrow S[1]$

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0 + 0 + 4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$
1	4			3	0	5	6	7	$4 + 0 + 6 \mod 8 = 2$	$S[1] \leftrightarrow S[2]$
2	4	2		3	0		6	7	$2 + 1 + 2 \mod 8 = 5$	$S[2] \leftrightarrow S[5]$
3		2	5		0	1	6	7	$5 + 1 + 4 \mod 8 = 1$	$S[3] \leftrightarrow S[1]$
4		2	5	4		1	6	7	$1 + 2 + 6 \mod 8 = 1$	$S[4] \leftrightarrow S[1]$

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0 + 0 + 4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$
1	4			3	0	5	6	7	$4 + 0 + 6 \mod 8 = 2$	$S[1] \leftrightarrow S[2]$
2	4	2		3	0		6	7	$2 + 1 + 2 \mod 8 = 5$	$S[2] \leftrightarrow S[5]$
3		2	5		0	1	6	7	$5 + 1 + 4 \mod 8 = 1$	$S[3] \leftrightarrow S[1]$
4		2	5	4		1	6	7	$1 + 2 + 6 \mod 8 = 1$	$S[4] \leftrightarrow S[1]$
5	0	2	5	4	3			7	$1 + 2 + 3 \mod 8 = 6$	$S[5] \leftrightarrow S[6]$

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0 + 0 + 4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$
1	4			3	0	5	6	7	$4 + 0 + 6 \mod 8 = 2$	$S[1] \leftrightarrow S[2]$
2	4	2		3	0		6	7	$2 + 1 + 2 \mod 8 = 5$	$S[2] \leftrightarrow S[5]$
3		2	5		0	1	6	7	$5 + 1 + 4 \mod 8 = 1$	$S[3] \leftrightarrow S[1]$
4		2	5	4		1	6	7	$1 + 2 + 6 \mod 8 = 1$	$S[4] \leftrightarrow S[1]$
5	0	2	5	4	3			7	$1 + 2 + 3 \mod 8 = 6$	$S[5] \leftrightarrow S[6]$
6	0	2		4	3	6		7	$6 + 1 + 3 \mod 8 = 2$	$S[6] \leftrightarrow S[2]$

Exemple d'exécution de KSA

Les deux opération de la boucle sont

```
j := (j + S[i] + K[i mod L]) mod 256
échanger(S[i], S[j])
```

On prend $N = 8$ et $K = [4, 6, 2, 4, 6, 3, 3, 7]$ et donc $L = 8$

Step(i)	0	1	2	3	4	5	6	7	$j = j + S[j] + K[i] \mod 8$	*Echange*
Init.	0	1	2	3	4	5	6	7	0	
0		1	2	3		5	6	7	$0+0+4 \mod 8 = 4$	$S[0] \leftrightarrow S[4]$
1	4			3	0	5	6	7	$4+0+6 \mod 8 = 2$	$S[1] \leftrightarrow S[2]$
2	4	2		3	0		6	7	$2+1+2 \mod 8 = 5$	$S[2] \leftrightarrow S[5]$
3		2	5		0	1	6	7	$5+1+4 \mod 8 = 1$	$S[3] \leftrightarrow S[1]$
4		2	5	4		1	6	7	$1+2+6 \mod 8 = 1$	$S[4] \leftrightarrow S[1]$
5	0	2	5	4	3			7	$1+2+3 \mod 8 = 6$	$S[5] \leftrightarrow S[6]$
6	0	2		4	3	6		7	$6+1+3 \mod 8 = 2$	$S[6] \leftrightarrow S[2]$
7	0	2	5	4	3	6			$2+5+7 \mod 8 = 6$	$S[7] \leftrightarrow S[6]$

RC4 - PRGA

PRGA génère des octets pseudo-aléatoire, et les ajoute aux caractères du message.

```
PRGA
i := 0
j := 0
tant_que générer une sortie:
    i := (i + 1) mod 256
    j := (j + S[i]) mod 256
    échanger(S[i], S[j])
    octet_chiffrement = S[(S[i] + S[j]) mod 256]
    result_chiffré = octet_chiffrement XOR octet_message
fintant_que
```

Exemple d'éxécution de PRGA

On considère le message $M = [100, 101, \ldots]$. On applique l'algorithme PRGA.

1. *Initialisation* $i = 0, j = 0$

k	0	1	2	3	4	5	6	7
$S[k]$	0	2	5	4	3	6	7	1

Exemple d'éxécution de PRGA

On considère le message $M = [100, 101, \ldots]$. On applique l'algorithme PRGA.

1. *Initialisation* $i = 0, j = 0$

k	0	1	2	3	4	5	6	7
$S[k]$	0	2	5	4	3	6	7	1

2. $i = 1, j = 0 + S[i] = 0 + 2$, on échange $S[1] \leftrightarrow S[2]$

k	0	1	2	3	4	5	6	7
$S[k]$	0			4	3	6	7	1

On chiffre le premier bloc de M. On a

$$octet\_chiffrement = S[(S[i] + S[j])mod256] = S[5 + 2] = S[7] = 1$$

et donc $C_1 = [100] \oplus [001] = 101$.

Exemple d'éxécution de PRGA

On considère le message $M = [100, 101, \ldots]$. On applique l'algorithme PRGA.

1. *Initialisation* $i = 0, j = 0$

k	0	1	2	3	4	5	6	7
$S[k]$	0	2	5	4	3	6	7	1

2. $i = 1, j = 0 + S[i] = 0 + 2$, on échange $S[1] \leftrightarrow S[2]$

k	0	1	2	3	4	5	6	7
$S[k]$	0			4	3	6	7	1

   On chiffre le premier bloc de M. On a

   $$octet\_chiffrement = S[(S[i] + S[j])mod256] = S[5 + 2] = S[7] = 1$$

   et donc $C_1 = [100] \oplus [001] = 101$.

3. $i = 2, j = 2 + S[2] = 2 + 5 = 7$, on échange $S[2] \leftrightarrow S[7]$

k	0	1	2	3	4	5	6	7
$S[k]$	0	5		4	3	6	7	

   On chiffre le deuxième bloc de M. On a
   $octet\_chiffrement = S[(1 + 2) \mod 256] = S[3] = 4$ et donc
   $C_2 = [101] \oplus [100] = 001$

Exemple d'éxécution de PRGA

On considère le message $M = [100, 101, \ldots]$. On applique l'algorithme PRGA.

1. *Initialisation* $i = 0, j = 0$

k	0	1	2	3	4	5	6	7
$S[k]$	0	2	5	4	3	6	7	1

2. $i = 1, j = 0 + S[i] = 0 + 2$, on échange $S[1] \leftrightarrow S[2]$

k	0	1	2	3	4	5	6	7
$S[k]$	0			4	3	6	7	1

   On chiffre le premier bloc de M. On a

   $$octet\_chiffrement = S[(S[i] + S[j])mod256] = S[5 + 2] = S[7] = 1$$

   et donc $C_1 = [100] \oplus [001] = 101$.

3. $i = 2, j = 2 + S[2] = 2 + 5 = 7$, on échange $S[2] \leftrightarrow S[7]$

k	0	1	2	3	4	5	6	7
$S[k]$	0	5		4	3	6	7	

   On chiffre le deuxième bloc de M. On a
   $octet\_chiffrement = S[(1 + 2) \mod 256] = S[3] = 4$ et donc
   $C_2 = [101] \oplus [100] = 001$

Exemple d'éxécution de PRGA

On considère le message $M = [100, 101, \ldots]$. On applique l'algorithme PRGA.

1. *Initialisation* $i = 0, j = 0$

k	0	1	2	3	4	5	6	7
$S[k]$	0	2	5	4	3	6	7	1

2. $i = 1, j = 0 + S[i] = 0 + 2$, on échange $S[1] \leftrightarrow S[2]$

k	0	1	2	3	4	5	6	7
$S[k]$	0			4	3	6	7	1

   On chiffre le premier bloc de M. On a

   $$octet\_chiffrement = S[(S[i] + S[j])mod256] = S[5 + 2] = S[7] = 1$$

   et donc $C_1 = [100] \oplus [001] = 101$.

3. $i = 2, j = 2 + S[2] = 2 + 5 = 7$, on échange $S[2] \leftrightarrow S[7]$

k	0	1	2	3	4	5	6	7
$S[k]$	0	5		4	3	6	7	

   On chiffre le deuxième bloc de M. On a
   $octet\_chiffrement = S[(1 + 2) \mod 256] = S[3] = 4$ et donc
   $C_2 = [101] \oplus [100] = 001$

Plan

WEP

- Sécurité dans les réseaux sans fil
- WEP : Wired Equivalent Protocol
- Ce protocole sécurise les données de la couche liaison pour les transmissions sans fil (WIFI) de la norme 802.11.

WEP

- Il présuppose l'existence d'une clé secrète entre les parties communicantes (la clé WEP) pour protéger le corps des frames transmises.
- L'utilisation du WEP est optionnel
- Il n'y a pas de protocoles de gestion de clé $\Rightarrow$ Une seule clé partagée par plusieurs utilisateurs.
- Cette clef sert de clef de chiffrement à toutes les sessions WEP.
- Le chiffrement se fait avec RC4.

WEP - Authentification

Alice s'authetifie auprès du serveur. Stratégie : protocole aléa-retour

- Le serveur choisit un aléa r de 128 bits et l'envoi à Alice.
- Alice chiffre chiffré avec la méthode précédente, $C' = r \oplus RC4(v, k)$
- Le serveur vérifie si C' = C (la valeur calculée par le serveur)

WEP : Structure des paquets

Un paquet/trame WEP comprend

- Une entête IEEE 802.11 de 30 octets (adresse MAC, etc.).
- Une IV de 24 bits.
- Données chiffrées avec RC4.
- Un code de détection d'erreur CRC sur 4 octets.

Entete IEEE 802.11 30 octets	IV 3 octets	Donnees < 2312 octets	CRC 4 octets

Attaque du WEP

Deux attaques statistiques contre RC4 avec des IVs

- FSM 2001 : des IV sont faibles et révèlent de l'information sur la clé à l'aide du premier octet de sortie.
- Amélioration de cette attaque par Hulton
  - utilisation des premiers octets de sortie
  - permet de réduire la quantité de données à capturer.
- Attaque de KoreK (2004) : généralisation des deux attaques précédentes + injection de paquets.
- On détermine le reste de la clé par recherche exhaustive

Actuellement il est assez facile de casser une clef WEP (1h pour l'injection de paquets et 1mn pour l'analyse statistique).

Plan

WPA

- WPA1. Utilise toujours RC4 mais corrige les défauts du WEP.
  - Utilisation de clef de session de 128 bits,
  - Authentification par hachage (802.1x pour l'authentification EAP (Extensive Authentication Protocol RFC 2284)).
  - Impossibilité de réutiliser un même IV avec la même clé Utilisation d'un contrôle d'intégrité du message (MIC) avec SHA-1
- WPA2, même chose que le WPA1 mais avec
  - l'AES en mode OFB pour le chiffrement,