SÉCURITÉ DES GÉNÉRATEURS PSEUDO-ALÉATOIRES

par

Robert Rolland

PARTIE I
GÉNÉRALITÉS

1. Introduction - Notations

1.1. Introduction. — Quand on s'intéresse à des notions de sécurité il faut distinguer plusieurs contextes :

(1) On peut s'intéresser à la sécurité d'un système donné, figé, dont tous les paramètres sont fixés, en particulier les tailles des objets qui interviennent. C'est une étude que nous avons appelée non-asymptotique, en opposition avec l'étude asymptotique dans laquelle le sytème dépend d'un paramètre qu'on fait tendre vers l'infini.

(2) On peut aussi s'intéresser à une famille de systèmes dépendant d'un paramètre (appelé paramètre de sécurité), et regarder ce qu'il se passe lorsque ce paramètre tend vers l'infini. C'est ce qu'il se passe lorsqu'on considère la famille de générateur pseudo-aléatoires Blum Blum Shub, basé sur un module N de taille k, et qu'on fait varier k. On dit alors qu'on fait une étude asymptotique. Dans une étude de ce type, toutes les données su système dépendent du paramètre de sécurité k.

Du point de vue de la formalisation de la sécurité, il faut d'une part définir le problème de sécurité qu'on se pose (par exemple dans notre cas le problème de distinguer une sortie d'un générateur pseudo-aléatoire d'une sortie purement aléatoire, ou encore connaissant les premiers termes d'une suite pseudo-aléatoire, prévoir le terme suivant) d'autre part il faut définir ce qu'est l'attaque. Nous considèrerons un attaquant comme un algorithme réalisable qui sort un résultat relatif au problème de sécurité posé. La notion d'algorithme réalisable est extrèmement importante car comme on travaille souvent sur des situations finies on peut décrire théoriquement des algorithmes qui cassent la sécurité des systèmes. Mais ces algorithmes ne sont pas réalisables en pratique à cause de la durée des calculs. On cherche donc à assurer non pas la sécurité parfaite, mais la sécurité calculatoire. L'attaquant sera donc considéré dans le cas non-asymptotique comme un algorithme probabiliste, dont le temps maximal d'exécution est borné par un nombre T, et dans le cas asymptotique, comme un algorithme probabiliste, dont le temps d'exécution est une fonction du paramètre de sécurité k, majorée par une fonction polynomiale de k.

À vrai dire, on ne prouve jamais une sécurité absolue, mais plutôt une sécurité qui repose sur la sécurité d'un autre système en qui on a confiance. Une preuve de la sécurité d'un système (S) consistera donc à démontrer que s'il existe un attaquant qui est capable de casser (S) alors grâce à cet attaquant on peut en construire un autre, capable de casser un système réputé sûr (par exemple de factoriser des nombres suffisamment grands). Bien entendu tous ces attaquants doivent s'exécuter en temps raisonnable.

Les types de sécurité, ainsi que les méthodes et techniques pour les étudier dépendent des secteurs de la cryptographie, même s'il y a des stratégies communes. Ainsi il faudra développer la théorie pour les générateurs pseudo-aléatoires, le chiffrement à clé secrète et ses modes d'utilisation, le chiffrement à clé publique, la signature. Ce qui nous intéresse ici est le secteur des générateurs pseudo-aléatoires. C'est ce que nous développerons ici, sous forme d'étude générale, à l'exclusion de toute preuve de sécurité de systèmes concrets. Ce dernier sujet sera

traité dans un autre exposé à partir du générateur de Blum Blum Shub et ses variantes.

1.2. Le plan de l'étude. —

(1) Étude non-asymptotique
   (a) Étude d'un générateur $f : \mathbf{U} \subseteq \{0,1\}^k \to \{0,1\}^l$
   (b) Étude d'une famille Γ de générateurs (k et l fixes)

(2) Étude asymptotique (l devient une fonction polynomiale en k et on fait tendre k vers $+\infty$)
   (a) Étude de la sécurité uniforme
   (b) Étude de la sécurité probabiliste

1.3. Notations. —

1.3.1. Typographie. — Dans la suite, les entiers seront notés par les lettres k, l, i, s, n, m. Les algorithmes seront notés $\mathcal{A}, \mathcal{B}, \mathcal{G}$. Les vecteurs de $\{0,1\}^m$ (m est un exposant entier quelconque) seront notés par X, Y. Par exemple $X = (x_1, x_2, \cdots, x_l)$. Un tel X est donc une suite de bits x_i. Les bits seront notés x_i, y_i, b. Les parties de $\{0,1\}^m$ seront notées en caractères droits, gras : $\mathbf{U}, \mathbf{Y}, \mathbf{Z}$. En particulier si $Y = (y_1, y_2, \cdots, y_l)$ représente un élément de $\{0,1\}^l$, alors $\mathbf{Y}$ est la partie $\{Y\}$ constituée du seul élément Y.

1.3.2. Description des algorithmes. — La flèche $\leftarrow$ désignera les opérations suivantes qui se distinguent dans le contexte :

– **l'affectation** d'une valeur à une variable. Exemples :

$$X \leftarrow (x_1, x_2, \cdots, x_l)$$

$$b \leftarrow 1$$

$$b \leftarrow \mathcal{A}(y_1, y_2, \cdots, y_l).$$

– **l'affectation au hasard** d'une variable suivant la loi de répartition uniforme. Exemples :

$$Y \leftarrow \{0,1\}^l$$

(on tire un vecteur binaire au sort suivant la loi uniforme)

$$b \leftarrow \{0,1\}$$

(On tire un bit au sort).

Les autres notations utilisées, qui concernent le déroulement des algorithmes ou des expériences aléatoires, sont classiques et se comprennent facilement.

1.4. Générateur pseudo-aléatoire. — Un générateur pseudo-aléatoire est une fonction f d'une partie $\mathbf{U} \subseteq \{0,1\}^k$ dans $\{0,1\}^l$ (où $k < l$) qui à un germe $X_0 \in \mathbf{U}$ (tenu secret) associe une suite finie de l bits :

$$(x_1, x_2, \cdots, x_l) = f(X_0).$$

Remarque 1.1. — En général la fonction f est construite grâce à un calcul récurrent qui permet de calculer successivement les bits x_i de $f(X_0)$. Un cas typique est celui où on dispose d'un fonction u de $\{0,1\}^k$ dans lui même qui permet de calculer par récurrence un état interne X_n à partir de la valeur initiale X_0 :

$$X_n = u(X_{n-1}),$$

ainsi que d'une fonction v qui à partir de X_n sort le bit x_n :

$$x_n = v(X_n).$$

On peut obtenir ainsi, les bits successifs de $f(X_0) = (x_1, \cdots, x_l)$. Cette façon de faire empêche, pourvu que les fonctions u et v soient convenablement construites, un attaquant qui connaît les t premiers bits $x_1, x_2, \cdots, x_t$ (mais pas le germe X_0) de pouvoir calculer facilement le bit x_{t+1}.

Exemple 1.2 **(Générateur $x^2 \mod n$ de Blum Blum Shub)**

Soit n un entier de Blum (c'est-à-dire un produit de deux nombres premiers p et q congrus à 3 modulo 4) ayant k bits (par exemple $k = 2048$). On définit à partir d'un germe X_0 ayant 128 bits (ici $\mathbf{U} = \{0,1\}^{128}$) la suite $X_i = X_{i-1}^2 \mod n$, et enfin $x_i = \mathrm{lsb}(X_i) = X_i \mod 2$. Ce générateur pseudo-aléatoire est le générateur BBS (de Blum Blum Shub).

1.5. Probabilités associées. — Nous définissons ici des probabilités associées à la donnée d'un générateur pseudo-aléatoire f, et nous établissons quelques formules simples, qui nous servirons par la suite, concernant ces probabilités.

Notons $P_{\mathbf{U}}$ la probabilité équirépartie sur $\mathbf{U}$, Π_j la probabilité équirépartie sur $\{0,1\}^j$ et Q_f la probabilité image par f de $P_{\mathbf{U}}$. Si $\mathbf{Y} \subseteq \{0,1\}^l$:

$$\Pi_l(\mathbf{Y}) = \frac{\#\mathbf{Y}}{2^l} \qquad Q_f(\mathbf{Y}) = P_{\mathbf{U}}\left(f^{-1}(\mathbf{Y})\right) = \frac{\#f^{-1}(\mathbf{Y})}{\#\mathbf{U}}.$$

Par la suite on sera amené à construire aléatoirement un élément $(y_1, \cdots y_l) \in \{0,1\}^l$ de la façon suivante :

Construction (C_s) **:**

(1) on fixe un entier s tel que $0 \leq s \leq l$;
(2) on tire au sort un entier $X_0 \in \mathbf{U}$ suivant la loi uniforme sur $\mathbf{U}$;
(3) on calcule $f(X_0) = (x_1, \cdots, x_l)$ dont on ne conserve que les s premiers bits $(y_1, \cdots, y_s)$ (avec $y_1 = x_1, \cdots, y_s = x_s$)
(4) on complète ces s bits par $l-s$ bits $(y_{s+1}, \cdots, y_s)$ pris au hasard dans $\{0,1\}^{l-s}$ suivant la loi uniforme.

On va donc introduire une probabilité adaptée à cette construction, c'est-à-dire la probabilité d'obtenir un $Y = (y_1, y_2, \cdots, y_l)$ lorsqu'on pratique la construction (C_s).

Pour tout entier $0 \leq s \leq l$ on définit sur $\{0,1\}^l$ la probabilité p_s suivante, définie sur les singletons par (on vérifie immédiatement que c'est bien une probabilité) :

$$p_s\Big(\{(y_1, y_2, \cdots, y_l)\}\Big) =$$
$$P_{\mathbf{U}}\Big(f^{-1}\left(\{(y_1, y_2, \cdots, y_s)\} \times \{0,1\}^{l-s}\right)\Big) \times \Pi_{l-s}\Big(\{(y_{s+1}, \cdots, y_l)\}\Big),$$

ce qui donne aussi, compte tenu de les définitions de $P_{\mathbf{U}}$, Π et Q_f :

$$(1) \qquad p_s\Big(\{(y_1, y_2, \cdots, y_l)\}\Big) = \frac{1}{2^{l-s}} Q_f\Big(\{(y_1, y_2, \cdots, y_s)\} \times \{0,1\}^{l-s}\Big).$$

Pour simplifier notons provisoirement $\mathbf{Y}$ l'événement :

$$\mathbf{Y} = \{Y\} = \{(y_1, y_2, \cdots, y_l\},$$

$\mathbf{Y}_s$ l'événement « les s premières composantes sont $y_1 \cdots y_s$ » :

$$\mathbf{Y}_s = \{(y_1, y_2, \cdots, y_s)\} \times \{0,1\}^{l-s},$$

et $\mathbf{Z}_{s+1}$ l'événement « la composante d'indice $s+1$ est y_{s+1} » :

$$\mathbf{Z}_{s+1} = \{0,1\}^s \times \{y_{s+1}\} \times \{0,1\}^{l-s-1}.$$

La formule (1), s'écrit alors :

$$p_s(\mathbf{Y}) = \frac{1}{2^{l-s}} Q_f(\mathbf{Y}_s). \tag{2}$$

De la définition d'une probabilité conditionnelle et de l'égalité :

$$\mathbf{Y}_s \cap \mathbf{Z}_{s+1} = \mathbf{Y}_{s+1},$$

on tire :

$$Q_f(\mathbf{Y}_s) \times Q_f(\mathbf{Z}_{s+1}|\mathbf{Y}_s) = Q_f(\mathbf{Y}_{s+1}),$$

ce qui donne en utilisant la formule (1) (ou la formule (2)) :

$$p_s(\mathbf{Y}) \times Q_f(\mathbf{Z}_{s+1}|\mathbf{Y}_s) = \frac{1}{2}\, p_{s+1}(\mathbf{Y}), \tag{3}$$

c'est-à-dire, en revenant aux notations antérieures :

$$\begin{aligned} &p_s\Big(\{(y_1, y_2, \cdots, y_l)\}\Big) \times \\ &Q_f\Big(\{0,1\}^s \times \{y_{s+1}\} \times \{0,1\}^{l-s-1} | \{(y_1, y_2, \cdots, y_s)\} \times \{0,1\}^{l-s}\Big) = \\ &\frac{1}{2}\, p_{s+1}\Big(\{(y_1, y_2, \cdots, y_l)\}\Big). \end{aligned} \tag{4}$$

Remarque 1.3. — Pour $s = 0$ on obtient $p_0 = \Pi_l$ (tous les bits sont tirés au sort suivant la loi uniforme). Pour $s = l$ on obtient $p_l = Q_f$ (tous les bits sont calculés selon le générateur pseudo-aléatoire).

PARTIE II
L'ÉTUDE NON-ASYMPTOTIQUE

2. La sécurité d'un générateur pseudo-aléatoire

Soit un générateur pseudo-aléatoire :

$$f : \mathbf{U} \subset \{0,1\}^k \to \{0,1\}^l.$$

On se fixe aussi un temps maximum d'exécution T. Soit $\mathcal{A}$ un algorithme probabiliste qui s'arrête en temps maximum $\leq T$ dont l'entrée est un élément de $\{0,1\}^l$ et la sortie un bit b.

Rappelons que la donnée d'un algorithme probabiliste est la donnée d'un algorithme non-déterministe ainsi que des probabilités définies pour chaque entrée e sur l'ensemble des exécutions qui peuvent se produire à partir de cette entrée. On notera $\mu_{\mathcal{A}}(e)$ la probabilité pour que l'entrée de l'algorithme $\mathcal{A}$ étant e, la sortie soit 1.

L'algorithme $\mathcal{A}$ étant fixé ainsi qu'un entier $0 \leq s \leq l$ on définit l'expérience aléatoire suivante, reliée à la construction (C_s) du paragraphe 1.5 :

$\mathbf{Expt}^{\mathbf{dist}}_{\mathbf{f,s}}(\mathcal{A})$
  $X_0 \leftarrow \{0,1\}^k$
  $X \leftarrow f(X_0)$
    (notation : $X = (x_1, \cdots, x_l)$)
  $Y_1 \leftarrow (x_1, x_2, \cdots, x_s)$
  $Y_2 \leftarrow \{0,1\}^{s-l}$
  $Y \leftarrow Y_1 || Y_2$
  $b \leftarrow \mathcal{A}(Y)$
  retour b
Fin.

Notons q_s la probabilité pour que le résultat b de l'expérience $\mathbf{Expt}^{\mathbf{dist}}_{\mathbf{f,s}}(\mathcal{A})$ soit 1. Avec les notations introduites :

$$q_s = \sum_{Y \in \{0,1\}^l} p_s(\mathbf{Y}) \mu_{\mathcal{A}}(Y). \tag{5}$$

Autrement dit, la probabilité q_0 est la probabilité de l'événement suivant : on tire un élément au hasard dans $\{0,1\}^l$, suivant la loi équirépartie, on lui applique l'algorithme $\mathcal{A}$, et on obtient 1. La probabilité q_l est quant à elle la probabilité de l'événement suivant : on tire au hasard un germe X_0 dans $\mathbf{U}$, on lui applique f pour obtenir un élément de $\{0,1\}^l$ qu'on soumet à l'algorithme $\mathcal{A}$, et on obtient pour résultat 1.

Définition 2.1. — L'avantage de l'algorithme $\mathcal{A}$ pour distinguer f est :

$$Adv_f^{dist}(\mathcal{A}) = |q_l - q_0|.$$

On définit alors la notion de (T,ϵ)-distingueur :

Définition 2.2. — Un (T,ϵ)-distingueur pour f est un algorithme probabiliste $\mathcal{A}$ qui s'exécute en temps maximal $\leq T$, qui a pour entrée un élément de $\{0,1\}^l$, qui produit un bit b et qui permet de distinguer entre la suite pseudo-aléatoire et la répartition uniforme, c'est-à-dire tel que :

$$Adv_f^{dist}(\mathcal{A}) > \epsilon.$$

On peut maintenant définir la (T,ϵ)-sécurité de f.

Définition 2.3. — Le générateur f est dit (T,ϵ)-sûr, s'il n'existe aucun (T,ϵ)-distingueur pour f. Autrement dit : tout algorithme probabiliste $\mathcal{A}$ de temps maximal d'exécution $t(\mathcal{A}) \leq T$ a un avantage plus petit que le nombre fixé ϵ :

$$Adv_f^{dist}(\mathcal{A}) \leq \epsilon.$$

Remarque 2.4. — Dans la définition de l'avantage on peut supposer que $q_l \geq q_0$, sinon on remplace $\mathcal{A}$ par l'algorithme complémentaire (qui renvoie 1 quand l'autre renvoie 0 et *vice versa*). Ceci permet de se passer si on veut de valeurs absolues.

3. L'imprévisibilité d'un générateur pseudo-aléatoire

Soit de nouveau un générateur pseudo-aléatoire :

$$f : \mathbf{U} \subset \{0,1\}^k \to \{0,1\}^l.$$

On se fixe aussi un temps maximum d'exécution T. Soit $1 \leq s < l$.

On considère un algorithme probabiliste $\mathcal{B}$ dont le temps maximal d'exécution est $\leq T$ et qui étant donnée une entrée de s bits renvoie un bit b.

On effectue l'expérience aléatoire suivante :

$\mathbf{Expt}_{\mathbf{f,s}}^{\mathbf{pred}}(\mathcal{B})$
$X_0 \leftarrow \{0,1\}^k$
$X \leftarrow f(X_0)$
(notation : $X = (x_1, \cdots, x_l)$)
$Y \leftarrow (x_1, x_2, \cdots, x_s)$
$b \leftarrow \mathcal{B}(Y)$
si $b = x_{s+1}$
alors retour 1
sinon retour 0
finsi
Fin.

Notons r_s la probabilité pour que le résultat de l'expérience $\mathbf{Expt}_{\mathbf{f,s}}^{\mathbf{pred}}(\mathcal{B})$ soit 1. Avec les notations introduites :

$$r_s = \sum_{Y \in \{0,1\}^l, y_{s+1}=1} p_s(\mathbf{Y}) \mu_{\mathcal{B}}(Y) + \sum_{Y \in \{0,1\}^l, y_{s+1}=0} p_s(\mathbf{Y}) \left(1 - \mu_{\mathcal{B}}(Y)\right).$$

Définition 3.1. — L'avantage de l'algorithme $\mathcal{B}$ pour prédire le $(s+1)^{\text{e}}$ bit calculé par f est :

$$Adv_{f,s}^{pred}(\mathcal{B}) = |r_s - \frac{1}{2}|.$$

On peut maintenant définir la notion de (T, s, ϵ)-extrapolateur.

Définition 3.2. — Soit f un générateur pseudo-aléatoire, $1 \leq s < l$. Un (T, s, ϵ)-extrapolateur $\mathcal{B}$ est un algorithme probabiliste dont le temps maximal d'exécution est $\leq T$, qui ayant en entrée les s premiers bits, d'une suite pseudo-aléatoire construite avec f, fournit en sortie 1 bit de telle sorte que :

$$Adv_{f,s}^{pred}(\mathcal{B}) > \epsilon.$$

On définit alors la notion de générateur pseudo-aléatoire (T, s, ϵ)-imprévisible.

Définition 3.3. — Soit f un générateur pseudo-aléatoire, et s un entier tel que $1 \leq s < l$. Le générateur f est dit (T, s, ϵ)-imprévisible, s'il n'existe aucun (T, s, ϵ)-extrapolateur.

4. Le théorème de Yao

Le théorème de Yao va relier la notion de sécurité avec la notion d'imprévisibilité du bit suivant. Nous commençons par l'exprimer ici sous forme non-asymptotique. Dans ce cas il donne lieu à deux résultats qui peuvent être considérés comme une condition nécessaire et une condition suffisante pour assurer la sécurité d'un générateur f.

Théorème 4.1. — *Soit un générateur aléatoire :*

$$f : \mathbf{U} \subset \{0,1\}^k \to \{0,1\}^l.$$

Un (T, s, ϵ)-extrapolateur permet de construire un $(T + c, \epsilon)$-distingueur (où c est une constante).

Démonstration. — Soit $\mathcal{B}$ un (T, s, ϵ)-extrapolateur. On définit un (T, ϵ)-distingueur $\mathcal{A}$ de la façon suivante :

$\mathcal{A}(x_1, x_2, \cdots x_l)$
  $b \leftarrow \mathcal{B}(x_1, x_2, \cdots x_s)$
  si $b = x_{s+1}$
    alors retour 1
    sinon retour 0
  finsi
Fin.

La probabilité pour que $\mathcal{A}(f(X_0)) = 1$ est donc $> 1/2 + \epsilon$, tandis que pour un éxhantillon $(y_1, y_2, \cdots, y_l)$ aléatoire dans $\{0,1\}^l$, la probabilité pour que $\mathcal{A}(y_1, y_2, \cdots, y_l) = 1$ est $1/2$. De plus, par rapport au temps d'exécution de $\mathcal{B}$, on rajoute juste le temps constant c nécessaire à la comparaison de b avec x_{s+1} (comparaison de deux bits). □

Théorème 4.2. — *Soit un générateur aléatoire :*

$$f : \mathbf{U} \subset \{0,1\}^k \to \{0,1\}^l.$$

Supposons que quel que soit $1 \leq s < l$*, il n'existe aucun* (T, s, ϵ)*-extrapolateur. Alors* f *est* $(T - (c_1 l + c_2), l\,\epsilon)$*-sûr (où* c_1 *et* c_2 *sont des constantes).*

Démonstration. — Supposons que f ne soit pas (T_1, η)-sûr. Alors il existe un algorithme distingueur $\mathcal{A}$ dont le temps d'exécution est $\leq T_1$ et qui possède un avantage $> \eta$. Reprenons la construction (C_s) definie au paragraphe 1.5, et utilisons les probabilités q_s introduites au paragraphe 2. Quitte à changer l'algorithme $\mathcal{A}$ en son complémentaire, on peut supposer que $q_l - q_0 > \eta$. De ce fait :

$$Adv_f^{dist}(\mathcal{A}) = q_l - q_0 =$$
$$(q_l - q_{l-1}) + (q_{l-1} - q_{l-2}) + \cdots + (q_s - q_{s-1}) + \cdots + (q_1 - q_0) > \eta.$$

Par suite, il existe un s tel que $(q_{s+1} - q_s) > \eta/l$. Définissons alors l'algorithme $\mathcal{B}$ suivant :

$\mathcal{B}(z_1, z_2, \cdots z_s)$
  $(z_{s+1}, \cdots, z_l) \leftarrow \{0,1\}^{l-s}$
  $b \leftarrow \mathcal{A}(z_1, \cdots, z_l)$
  si $b = 1$
    alors retour z_{s+1}
    sinon retour $\overline{z_{s+1}}$
  finsi
Fin.

Cet algorithme s'exécute en temps moindre que $T_1 + c_1 l + c_2$, où c_1 est le temps constant pour choisir au hasard 1 bit, et c_2 le temps constant pour retourner z_s ou $\overline{z_s}$ suivant la valeur de b. Nous allons montrer que c'est un $(T_1 + c_1 l + c_2, s, \eta/l)$-extrapolateur. Pour cela nous devons calculer la probabilité r_s pour que le résultat de l'expérience $\mathbf{Expt}_{\mathbf{f,s}}^{\mathbf{pred}}(\mathcal{B})$ soit 1. Nous allons donc injecter la définition de $\mathcal{B}$ que nous venons de donner à partir de $\mathcal{A}$ dans la définition du déroulement de l'expérience $\mathbf{Expt}_{\mathbf{f,s}}^{\mathbf{pred}}(\mathcal{B})$.

Nous obtenons donc en procédant de cette façon, l'expérience suivante, qui va nous fournir la manière de calculer la probabilité r_s dont nous avons besoin.

$\mathbf{Expt}_{\mathbf{f,s}}^{\mathbf{pred}}(\mathcal{B})$
$X_0 \leftarrow \{0,1\}^k$
$Y \leftarrow f(X_0)$
(notation : $Y = (x_1, \cdots, x_l)$)
$Y_s \leftarrow (x_1, x_2, \cdots, x_s)$
(Y_s est l'entrée de $\mathcal{B}$, qui ne connait que ces composantes)

•Début de l'injection de la définition de $\mathcal{B}$
$(z_{s+1}, \cdots, z_l) \leftarrow \{0,1\}^{l-s}$
$b_1 \leftarrow \mathcal{A}(x_1, \cdots, x_s, z_{s+1}, \cdots, z_l)$
si $b_1 = 1$
alors $b \leftarrow z_{s+1}$
sinon $b \leftarrow \overline{z_{s+1}}$
•Fin de l'injection

si $b = x_{s+1}$
alors retour 1
sinon retour 0
finsi
Fin.

On remarque que l'expérience a pour résultat 1 lorsque $b = x_{s+1}$, c'est-à-dire dans les deux cas suivants :

(1) $b_1 = 1$ et $z_{z+1} = x_{s+1}$;
(2) $b_1 = 0$ et $\overline{z_{s+1}} = x_{s+1}$.

Reprenons, en les adaptant aux notations de l'expérience précédente, les notations simplifiées déjà utilisées dans le paragraphe 1.5 : $\mathbf{Y}$ est l'événement $\{(x_1, \cdots, x_l)\}$, $\mathbf{Y}_s$ désigne l'événement « les s premières composantes sont $x_1, \cdots x_s$ » $\mathbf{Z}_{s+1}$ désigne l'événement « la composante d'indice $s+1$ est z_{s+1} ». Posons aussi $\nu_{\mathcal{A}}(Y) = 1 - \mu_{\mathcal{A}}(Y)$.

Remarquons que dans ces conditions, $Q_f(\mathbf{Z}_{s+1}|\mathbf{Y}_s)$ est la probabilité conditionnelle, lorsque Y est construit à partir d'un germe au hasard par le générateur pseudoaléatoire, que la composante d'indice s de Y (c'est-à-dire x_{s+1}) soit égale à z_{s+1}, sachant que les s premières composantes sont $(x_1, \cdots, x_s)$.

On a donc :

$$\begin{aligned} r_s &= \sum_{Y\in\{0,1\}^l} p_s(\mathbf{Y})\Big(Q_f\left(\mathbf{Z}_{s+1}|\mathbf{Y}_s\right)\mu_{\mathcal{A}}(Y) + Q_f\left(\overline{\mathbf{Z}_{s+1}}|\mathbf{Y}_s\right)\nu_{\mathcal{A}}(Y)\Big) \\ &= \sum_{Y\in\{0,1\}^l} p_s(\mathbf{Y})\Big(Q_f\left(\mathbf{Z}_{s+1}|\mathbf{Y}_s\right)\mu_{\mathcal{A}}(Y) + \left(1 - Q_f\left(\mathbf{Z}_{s+1}|\mathbf{Y}_s\right)\right)\nu_{\mathcal{A}}(Y)\Big). \end{aligned}$$

En utilisant la formule (4) on obtient :

$$\begin{aligned} r_s &= \frac{1}{2}\sum_{Y\in\{0,1\}^l} p_{s+1}(\mathbf{Y})\Big(\mu_{\mathcal{A}}(Y) - \nu_{\mathcal{A}}(Y)\Big) + \sum_{Y\in\{0,1\}^l} p_s(\mathbf{Y})\nu_{\mathcal{A}}(Y) \\ &= \frac{1}{2}\sum_{Y\in\{0,1\}^l} p_{s+1}(\mathbf{Y})\Big(2\mu_{\mathcal{A}}(Y) - 1\Big) + \sum_{Y\in\{0,1\}^l} p_s(\mathbf{Y})(1 - \mu_{\mathcal{A}}(Y)) \\ &= \frac{1}{2} + \sum_{Y\in\{0,1\}^l}\Big(p_{s+1}(\mathbf{Y}) - p_s(\mathbf{Y})\Big)\mu_{\mathcal{A}}(Y). \end{aligned}$$

Cette dernière égalité nous donne en utilisant la formule (5) :

$$r_s = \frac{1}{2} + q_{s+1} - q_s,$$

d'où :

$$r_s > \frac{1}{2} + \frac{\eta}{l}.$$

En posant $T_1 = T - (c_1 l + c_2)$ et $\eta = l\epsilon$ on a le résultat. □

5. La sécurité d'une famille de taille fixe de générateurs pseudo-aléatoire

Nous avons précédemment étudié le cas d'un générateur pseudo-aléatoire f. Cependant, même si on reste dans le cadre non-asymptotique, où k et l sont fixés, on est en pratique amené à étudier non pas un seul f, mais une famille (finie nécessairement puisque k et l sont fixes) Γ de fonctions f définies sur une partie $\mathbf{U}_f$ (qui peut varier avec f) de $\{0,1\}^k$ à valeurs dans $\{0,1\}^l$. C'est le cas par exemple des algorithmes de Blum Blum Shub : la taille étant fixée, le module N peut varier. On considère alors des algorithmes d'attaque, qui attaquent tous les générateurs de la famille.

La théorie peut se dérouler alors de plusieurs façons. Soit on peut regarder le « plus mauvais cas », et on se ramène alors à l'étude d'une seule fonction. Dans ce cas les résultats du paragraphe précédent s'appliquent directement. Soit on inclut le tirage au sort de la fonction f dans les expériences aléatoires qui permettent de définir l'avantage de l'attaquant.

Ainsi on remplace dans ce dernier cas les algorithmes $\mathcal{A}$ et $\mathcal{B}$ par des algorithmes dont l'entrée comporte en plus la fonction f. Les expériences aléatoires $\mathbf{Expt}^{\mathbf{dist}}_{\mathbf{f,s}}(\mathcal{A})$ et $\mathbf{Expt}^{\mathbf{pred}}_{\mathbf{f,s}}(\mathcal{B})$ sont remplacées par les expériences aléatoires $\mathbf{Expt}^{\mathbf{dist}}_{\mathbf{\Gamma,s}}(\mathcal{A})$ et $\mathbf{Expt}^{\mathbf{pred}}_{\mathbf{\Gamma,s}}(\mathcal{B})$ où on tire au sort non seulement le germe X_0, mais aussi la fonction f elle même :

$\mathbf{Expt}^{\mathbf{dist}}_{\mathbf{\Gamma,s}}(\mathcal{A})$
  $f \leftarrow \Gamma$
  $X_0 \leftarrow \{0,1\}^k$
  $X \leftarrow f(X_0)$
    (notation : $X = (x_1, \cdots, x_l)$)
  $Y_1 \leftarrow (x_1, x_2, \cdots, x_s)$
  $Y_2 \leftarrow \{0,1\}^{s-l}$
  $Y \leftarrow Y_1 || Y_2$
  $b \leftarrow \mathcal{A}(f, Y)$
  retour b
Fin.

$\mathbf{Expt}^{\mathbf{pred}}_{\mathbf{\Gamma,s}}(\mathcal{B})$
  $f \leftarrow \Gamma$
  $X_0 \leftarrow \{0,1\}^k$
  $X \leftarrow f(X_0)$
    (notation : $X = (x_1, \cdots, x_l)$)
  $Y \leftarrow (x_1, x_2, \cdots, x_s)$
  $b \leftarrow \mathcal{B}(f, Y)$
  si $b = x_{s+1}$
    alors retour 1
    sinon retour 0
  finsi
Fin.

On va noter maintenant $p_{f,s}$ (pour montrer que cette probabilité est spécifique à la fonction f), la probabilité qu'on avait appelé p_s dans le paragraphe précédent. De la même façon, en spécifiant ce qui dépend de f, on introduira, conformément aux notations du paragraphe précédent :

$$q_{f,s}, r_{f,s}, \mu_{\mathcal{A}}(f, Y), \mu_{\mathcal{B}}(f, Y).$$

Nous supposerons que f est tiré uniformément au sort dans Γ (mais ce n'est pas primordial). On pose $\gamma = 1/(\#\Gamma)$ (si on veut on peut prendre une probabilité autre $P_\Gamma(\{f\})$). En tenant compte du tirage au sort de f on obtient alors les probabilités des résultats des expériences sous la forme :

$$q_s = \gamma \sum_{f\in\Gamma} q_{f,s} = \gamma \sum_{f\in\Gamma} \sum_{Y\in\{0,1\}^l} p_{f,s}(Y)\mu_{\mathcal{A}}(f, Y),$$

$$r_s = \gamma \sum_{f\in\Gamma} r_{f,s}.$$

Nous constatons que toutes les définitions et les démonstrations du paraghraphe précédent se généralisent et que les théorèmes de Yao, avec les définitions qui découlent de la nouvelle situation, sont encore valides.

PARTIE III
L'ÉTUDE ASYMPTOTIQUE

6. Ne pas confondre

Attention, nous ne parlons ici que de machines de Turing uniformes, c'est-à-dire que la machine de Turing utilisée est la même quelque soit le paramètre de sécurité k et ne varie pas avec celui-ci.

La sécurité uniforme que nous décrivons par la suite, n'a rien à voir avec la notion d'uniformité au sens des machines de Turing. C'est seulement une notion qui précise que la qualité de la sécurité pour une valeur donnée du paramètre de sécurité k, ne dépend pas de la fonction f_k tirée au

sort dans la famille des générateurs pseudo-aléatoires admissibles pour le paramètre k.

7. La sécurité asymptotique uniforme

Nous allons maintenant considérer que nous disposons d'une famille probabiliste à un paramètre k de générateurs pseudo-aléatoires. On fixe avant toute chose, une fonction polynomiale $l(k)$. On dispose alors d'un algorithme $\mathcal{G}$ probabiliste polynomial, qui étant donné le paramètre de sécurité k fournit un générateur pseudo-aléatoire :

$$f_k : \mathbf{U}_k \subset \{0,1\}^k. \to \{0,1\}^{l(k)}.$$

Nous noterons $\mathcal{G}_k$ l'ensemble des générateur pseudo-aléatoires qui peuvent être renvoyés par $\mathcal{G}$, quand on lui fournit k.

Soit $S(k)$ un polynôme. Soit $\mathcal{A}$ un algorithme probabiliste, prenant en entrée k, un élément de $\mathcal{G}_k$, un élément de $\{0,1\}^{l(k)}$, et donnant en sortie un bit b, dont le temps maximum d'exécution est majoré par $S(k)$.

Remarque 7.1. — Attention, les algorithmes distingueurs qu'on avait appelé $\mathcal{A}$ dans la section 2, correspondent maintenant à $\mathcal{A}(k, f_k)$ où on a fixé les entrées k et f_k et où seul $Y \in \{0,1\}^{l(k)}$ reste une variable libre. Une remarque identique pourra être faite pour les algorithmes extrapolateurs $\mathcal{B}$.

Notons :

$$Adv_{\mathcal{G}}^{u-dist}(\mathcal{A}, k) = \max_{f_k \in \mathcal{G}_k} Adv_{f_k}^{dist}(\mathcal{A}(k, f_k)). \tag{6}$$

Définition 7.2. — Nous dirons que la famille donnée par $\mathcal{G}$ est uniformément sûre si pour tout polynôme $S(k)$ et pour tout algorithme probabiliste $\mathcal{A}$ qui s'exécute en temps polynomial majoré par $S(k)$, l'avantage $Adv_{\mathcal{G}}^{u-dist}(\mathcal{A}, k)$ est une fonction négligeable de k, c'est-à-dire que pour tout entier m :

$$\lim_{k \to +\infty} k^m Adv_{\mathcal{G}}^{u-dist}(\mathcal{A}, k) = 0.$$

Remarque 7.3. — C'est dans la formule (6) qui définit l'avantage à partir d'un max que réside l'uniformité. On ne permet donc pas d'entorse, pour une valeur de k, toutes les fonctions f_k doivent rentrer dans le moule.

On ne permet pas qu'il y en ait une qui ne soit pas sûre, même si sa probabilité d'être tiré par l'algorithme $\mathcal{G}$ est infime.

On développe une théorie analogue pour la notion de prédiction du bit suivant.

Soit $S(k)$ un polynôme. Soit $\mathcal{B}$ un algorithme probabiliste, dont le temps maximum d'exécution est majoré par $S(k)$, prenant en entrée k, un élément de $\mathcal{G}_k$, un entier s de l'intervalle $[1, l(k)-1]$, un élément de $\{0,1\}^s$, et donnant en sortie un bit b. dont le temps maximum d'exécution est majoré par $S(k)$. Notons :

$$Adv_{\mathcal{G}}^{u-pred}(\mathcal{B}, k) = \max_{f_k \in \mathcal{G}_k, s \in [1, l(k)-1]} Adv_{f_k, s}^{pred}(\mathcal{B}(k, f_k, s)).$$

***Théorème 7.4* (Yao).** — *La famille donnée par $\mathcal{G}$ est uniformément sûre, si et seulement si, pour tout polynôme $S(k)$, pour tout algorithme probabiliste $\mathcal{B}$ (dont les entrées et sorties sont comme il vient d'être dit) dont le temps maximum d'exécution est majoré par $S(k)$, l'avantage $Adv_{\mathcal{G}}^{u-pred}(\mathcal{B}, k)$ est une fonction négligeable de k.*

Démonstration. — C'est bien entendu une conséquence des théorèmes 4.1 et 4.2. Nous allons détailler la démonstration.

(1) **Partie directe.** Supposons l'existence d'un polynôme $S(k)$ et d'un algorithme probabiliste $\mathcal{B}$ de temps d'exécution majoré par $S(k)$ tel que $Adv_{\mathcal{G}}^{u-pred}(\mathcal{B}, k)$ ne soit pas une fonction négligeable de k. Alors on peut construire une suite $(k_i, f_{k_i}, s_i)_{i>0}$ où :

(a) $(k_i)_i$ est une suite stritement croissante tendant vers $+\infty$ de valeurs du paramètre de sécurité ;
(b) $f_{k_i} \in \mathcal{G}_k$, est un générateur pseudo-aléatoire associé au paramètre de sécurité k_i ;
(c) s_i est un indice de l'intervalle $[1, l(k)-1]$;
(d) m un exposant entier ;

de telle sorte que :

$$\lim_{i \to +\infty} k_i^m \, Adv_{f_{k_i}, s_i}^{pred}(\mathcal{B}(k_i, f_{k_i}, s_i)) = +\infty.$$

À partir de chaque algorithme de prédiction $\mathcal{B}(k_i, f_{k_i})$ on construit l'agorithme de distinction $\mathcal{A}(k_i, f_{k_i})$ comme il est dit dans le théorème 4.1. Si

$k \neq k_i$ ou $f \neq f_{k_i}$, prenons pour $\mathcal{A}(k, f)$ l'algorithme qui renvoie toujours 1. Fixons nous $S_1(k) = S(k) + c$ (c est la constante qui intervient dans le théorème 4.1), qui est bien un polynôme en k. Le théorème 4.1 nous permet de conclure que pour l'algorithme $\mathcal{A}$ qui s'exécute en temps majoré par $S_1(k)$ on a :

$$Adv_{\mathcal{G}}^{u-dist}(\mathcal{A}, k_i) \geq Adv_{f_{k_i},s}^{pred}(\mathcal{B}(k_i, f_{k_i}, s)),$$

donc que :

$$Adv_{\mathcal{G}}^{u-dist}(\mathcal{A}, k)$$

n'est pas une fonction négligeable de k.

(2) **Partie réciproque.** Si la famille donnée par $\mathcal{G}$ n'est pas uniformément sûre, alors il existe un polynôme $S(k)$ et un algorithme $\mathcal{A}$ de temps d'exécution majoré par $S(k)$ de telle sorte que :

$$Adv_{\mathcal{G}}^{u-dist}(\mathcal{A}, k)$$

ne soit pas une fonction négligeable de k. Comme dans la partie directe on peut donc construire une suite (k_i, f_{k_i}) et un entier m tels que :

$$\lim_{i \to +\infty} k_i^m \, Adv_{f_{k_i}}^{dist}(\mathcal{A}(k_i, f_{k_i})) = +\infty.$$

À partir des distingueurs $\mathcal{A}(k_i, f_{k_i})$, le théorème 4.2 nous permet de construire les prédicteurs $\mathcal{B}(k_i, f_{k_i}, s_i)$. Pour toute autre valeur de (k, f, s) que (k_i, f_i, s_i), on prend pour $\mathcal{B}(k, f, s)$, l'algorithme qui renvoie toujours 1. On a défini ainsi un algorithme $\mathcal{B}$, qui, d'après le théorème 4.2 s'exécute en temps majoré par le polynôme $S(k) + c_1 l(k) + c_2$ et tel que :

$$Adv_{\mathcal{G}}^{u-pred}(\mathcal{B}, k_i) \geq Adv_{f_{k_i}}^{dist}(\mathcal{A}(k_i, f_{k_i}))/l(k).$$

Soit $m' > \deg(l(k))$, alors :

$$\lim_{i \to +\infty} k_i^{m+m'} Adv_{\mathcal{G}}^{u-pred}(\mathcal{B}, k_i) = +\infty,$$

ce qui prouve que :

$$Adv_{\mathcal{G}}^{u-pred}(\mathcal{B}, k)$$

n'est pas une fonction négligeable de k.

□

8. L'étude asymptotique probabiliste

Soit $S(k)$ un polynôme. Soit $\mathcal{A}$ un algorithme probabiliste, prenant en entrée k, un élément de $\mathcal{G}_k$, un élément de $\{0,1\}^{l(k)}$, et donnant en sortie un bit b, dont le temps maximum d'exécution est majoré par $S(k)$. Considérons l'expérience aléatoire suivante :

$\mathbf{Expt}_{\mathcal{G}}^{\mathbf{p-dist}}(\mathcal{A},k)$
  $f_k \leftarrow \mathcal{G}(k)$
  $X_0 \leftarrow \mathbf{U}_k$
  $Y \leftarrow f_k(X_0)$
  $b \leftarrow \mathcal{A}(k, f_k, Y)$
  retour b
Fin.

que nous allons comparer à l'expérience :

$\widetilde{\mathbf{Expt}}_{\mathcal{G}}^{\mathbf{p-dist}}(\mathcal{A},k)$
  $Y \leftarrow \{0,1\}^{l(k)}$
  $b \leftarrow \mathcal{A}(Y)$
  retour b
Fin.

Nous définissons alors l'avantage de l'attaquant par :

$$Adv_{\mathcal{G}}^{p-dist}(\mathcal{A},k) = \left| Prob\left(\mathbf{Expt}_{\mathcal{G}}^{\mathbf{p-dist}}(\mathcal{A},k)\right) - Prob\left(\widetilde{\mathbf{Expt}}_{\mathcal{G}}^{\mathbf{p-dist}}(\mathcal{A},k)\right)\right|.$$

Définition 8.1. — Nous dirons que la famille donnée par $\mathcal{G}$ est probablement sûre si pour tout polynôme $S(k)$ et pour tout algorithme probabiliste $\mathcal{A}$ qui s'exécute en temps polynomial majoré par $S(k)$, l'avantage $Adv_{\mathcal{G}}^{p-dist}(\mathcal{A},k)$ est une fonction négligeable de k, c'est-à-dire que pour tout entier m :

$$\lim_{k\to+\infty} k^m Adv_{\mathcal{G}}^{p-dist}(\mathcal{A},k) = 0.$$

On développe une théorie analogue pour la notion de prédiction du bit suivant.

Soit $S(k)$ un polynôme. Soit $\mathcal{B}$ un algorithme probabiliste, dont le temps maximum d'exécution est majoré par $S(k)$, prenant en entrée k, un élément de $\mathcal{G}_k$, un entier s de l'intervalle $[1, l(k)-1]$, un élément de

$\{0,1\}^s$, et donnant en sortie un bit b. dont le temps maximum d'exécution est majoré par $S(k)$.

Faisons l'expérience suivante :

$\mathbf{Expt}_{\mathcal{G},\mathbf{s}}^{\mathbf{p-pred}}(\mathcal{A},k)$
$f_k \leftarrow \mathcal{G}(k)$
$X_0 \leftarrow \mathbf{U}_k$
$Y \leftarrow f_k(X_0)$
$Y_s \leftarrow Y$ (les s composantes de Y_s sont
les s premières composantes de Y)
$b \leftarrow \mathcal{B}(k, f_k, s, Y_s)$
retour $b_1 = (b = y_{s+1})$
retour b
Fin.

On définit alors l'avantage de $\mathcal{B}$ par :

$$Adv_{\mathcal{G}}^{p-pred}(\mathcal{B},k) = \max_{s\in[1,l(k)-1]} \left| Prob\left(\mathbf{Expt}_{\mathcal{G},\mathbf{s}}^{\mathbf{p-pred}}(\mathcal{A},k)=1\right) - \frac{1}{2}\right|.$$

On a alors une version du théorème de Yao adaptée à cette situation :

***Théorème 8.2* (Yao).** — *La famille donnée par $\mathcal{G}$ est probablement sûre, si et seulement si, pour tout polynôme $S(k)$, pour tout algorithme probabiliste $\mathcal{B}$ (dont les entrées et sorties sont comme il vient d'être dit) dont le temps maximum d'exécution est majoré par $S(k)$, l'avantage $Adv_{\mathcal{G}}^{p-pred}(\mathcal{B},k)$ est une fonction négligeable de k.*

Démonstration. — Le résultat est une conséquence de l'étude faite sur les familles de générateurs au paragraphe 5. □

9. Changement de sens de l'extrapolation

Dans le paragraphe 3 nous avons défini et utilisé la notion « d'extrapolateur à droite » c'est-à-dire qu'à partir des bits $(x_1,\cdots,x_s)$ l'algorithme extrapolateur prédit la valeur du bit x_{s+1} (prédiction du bit suivant). Toute l'étude peut être refaite à l'identique pour la notion « d'extrapolateur à gauche » c'est-à-dire, pour un algorithme qui, étant donnés les bits $(x_{s+1},\cdots,x_l)$, prédit le bit le bit x_s (prédiction du bit

précédent). En particulier, les théorèmes de Yao (version asymptotique ou non-asymptotique) restent valables pour les extrapolateurs à gauche.

---

27 février 2008

R. Rolland, Institut de Mathématiques de Luminy, Campus de Luminy, Case 907, 13288 MARSEILLE Cedex 9 • *E-mail :* robert.rolland@acrypta.fr
Url : http://www.acrypta.fr/~rolland