Cryptographie sur les courbes elliptiques et Dual_EC_DRBG

Yohann THEPAUT - Lucas CHARDONNET

Introduction

La cryptographie sur les courbes elliptiques est un ensemble de techniques qui utilisent les propriétés des courbes elliptiques pour avoir un système de chiffrement asymétrique.
Leur utilisation fut suggérée pour la première fois en 1985 par Neal Koblitz et Victor S. Miller, mais c'est à partir de 2004 qu'elles commenceront à être utilisées et standardisées.
Aujourd'hui, elles sont utilisées dans plusieurs algorithmes, notamment dans les signatures (ECDSA), l'échange de clés Diffie Hellman sur courbes elliptiques (ECDH), les générateurs de nombres pseudo-aléatoires cryptographiques (CSPRNG).

C'est sur cette dernière application que l'on va se pencher, notamment sur l'algorithme Dual_EC_DRBG et la potentielle vulnérabilité présente dans le générateur aléatoire.

Fonctionnement des courbes elliptiques

Avant de rentrer dans les détails, il est important de comprendre comment la cryptographie sur courbes les elliptiques fonctionne.

Les courbes elliptiques sont des courbes de la forme $y^2 = x^3 + ax + b$, avec $a \in R,\ b \in R$.

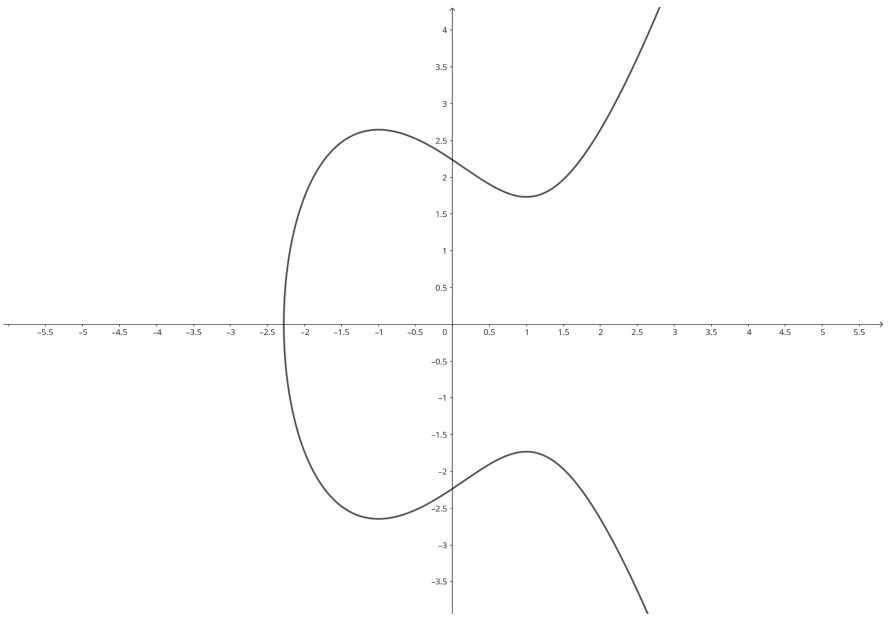

Figure 1 : Exemple de courbe elliptique $y^2 = x^3 - 3x + 5$

Sur cette courbe, on va pouvoir placer un point P, qui sera notre point de départ.

L'opération de base des courbes elliptiques est l'"addition". Par exemple, on peut "additionner" P à lui-même pour obtenir $2P$.
Pour calculer $2P$, il suffit de tracer la tangente à la courbe passant par P. De par la nature des courbes elliptiques, la tangente n'intersecte qu'un seul autre point de la courbe. On prend le symétrique de ce point par rapport à l'axe des abscisses et on obtient $2P$.

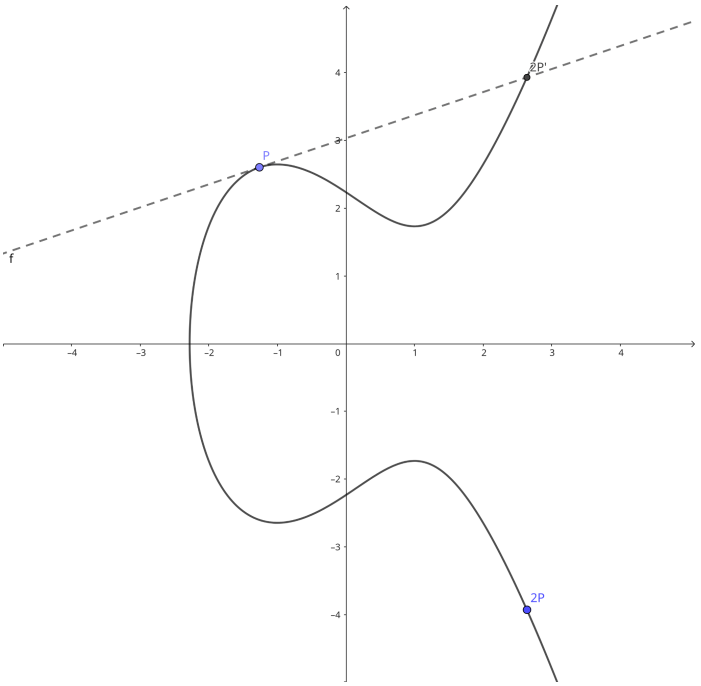

Figure 2 : Construction de $2P$

On peut continuer ainsi et additionner P et $2P$. Pour ce faire, on prend la droite passant par ces deux points, regarder où la droite coupe la courbe, prendre son symétrique et obtenir $3P$.

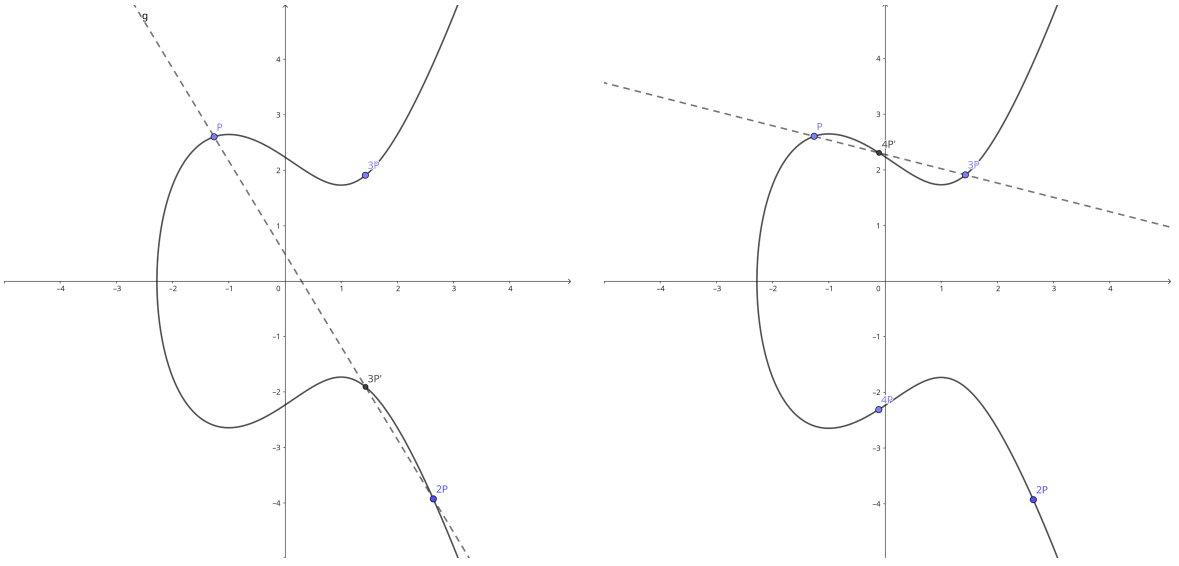

Figure 3 : Construction de $3P$ et $4P$

Une propriété fondamentale de cette opération est qu'il est difficile de trouver x en ayant les points P et xP.
A ce jour, aucun algorithme n'existe pour résoudre le problème du logarithme discret d'une courbe elliptique (ECDLP). [2]

Fonctionnement général d'un générateur pseudo aléatoire

Le fonctionnement général des générateurs pseudo-aléatoires basés sur un système d'état est le suivant :

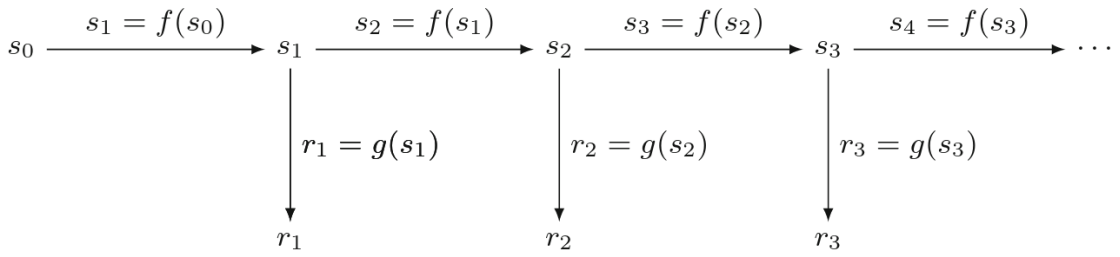

Figure 4 : Schéma du fonctionnement général d'un PRNG

Le générateur de nombres pseudo-aléatoires (PRNG) est constitué d'un état interne s qui se transforme à chaque tirage de nombre aléatoires, ainsi que de deux fonctions non réversibles f et g qui sont utilisées pour mettre à jour cet l'état interne et générer un nombre aléatoire en fonction de l'état, respectivement.

Dans la Figure 4, on observe l'état initial s_0 , que l'on peut aussi appeler graine.
Pour tirer un nombre, on fait une mise à jour de l'état s pour obtenir $s_{i+1} = f(s_i)$. À partir de ce nouvel état, on peut obtenir un nombre aléatoire r_i en calculant $r_i = g(s_i)$.

On note l'importance de la non réversibilité de f et de g.
Si g est inversible, il serait possible de retrouver l'état s_i , et ainsi deviner tous les prochains tirages.
Si f et g sont inversibles, il serait également possible de retrouver tous les précédents tirages.

Fonctionnement de Dual_EC_DRBG

L'algorithme Dual Elliptic Curve Deterministic Random Bit Generator (Dual_EC_DRBG) est un générateur de nombre pseudo-aléatoire cryptographique qui se base sur le même principe vu précédemment.

Il y a trois versions de cet algorithme qui utilisent trois courbes du standard "NSA Suite B". Chaque courbe est définie par :

- un nombre premier p,
- une courbe elliptique E sur un champ fini $F_p(Z/pZ)$,
- deux points P et Q sur E

La courbe la plus utilisée dans la suite “NSA Suite B” est la courbe P-256, où :

$p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$

$E: y^2 = x^3 - 3x + 5AC635D8AA3A93E7B3EBBD55769886BC651D06B0CC53B0F63BCE3C3E27D2604B_{16}$

$P_x = 6B17D1F2E12C4247F8BCE6E563A440F277037D812DEB33A0F4A13945D898C296_{16}$

$P_y = 4FE342E2FELA7F9B8EE7EB4A7C0F9E162BCE33576B315ECECBB606837BF51F5_{16}$

$Q_x = C97445F45CDEF9F0D3E05E1E585FC297235B82B5BE8FF3EFCA67C59852018192_{16}$

$Q_y = B28EF557BA31DFCBDD21AC46E2A91E3C304F44CB87058ADA2CB85151E610046_{16}$

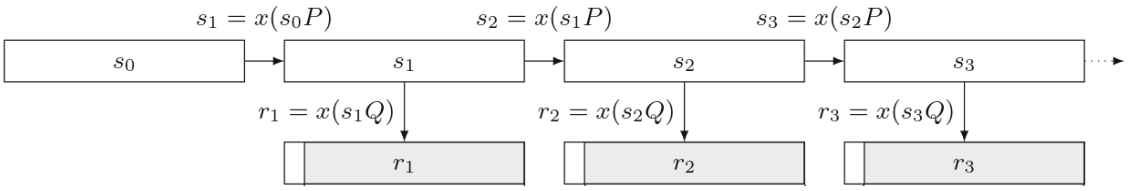

Figure 5 : Schéma du fonctionnement de Dual_EC_DRBG

La Figure 5 montre les opérations effectuées pour tirer un nombre aléatoire.
On note $x(A)$ la composante x du point A, et pour un point A sur F_p, la composante x de A appartient à F_p.
Ici, la fonction f pour la mise à jour de l’état correspond à $x(s_iP)$ et la fonction g pour le tirage du nombre aléatoire correspond à $x(s_iQ)$.

On note que pour P-256, les 16 premiers bits de poids fort de r_i sont retirés, ce qui fait que r est sur 240 bits.
Étant donné cette limite si l’on souhaite obtenir un nombre plus grand que 2^{240}, il suffit de générer plusieurs nombre et de concaténer leurs bits.

La raison pour laquelle les premiers bits de poids fort sont retirés, est que cela rend le processus encore plus compliqué à inverser. Retrouver sQ à partir des 240 derniers bits prend jusqu’à 2^{16} opérations.

La vulnérabilité de Dual EC

Dual_EC_DRBG a été développé par la National Security Agency (NSA) des États-Unis et fut progressivement introduit à partir de 2004. Cependant, il est devenu controversé en raison de préoccupations concernant d’éventuelles vulnérabilités insérées par la NSA.

En effet, si l'attaquant choisit P et Q tel que $P = dQ$, alors il connaît le logarithme discret de P sur la base Q. Un attaquant peut donc à partir de r_i trouver s_{i+1} et donc trouver tous les prochains nombres pseudo aléatoires.
Pour cela, il suffit de calculer : $d * r_i$. On sait que $r_i = s_i Q$ donc $d * r_i = d * s_i * Q = s_i * d * Q$. Mais comme $d * Q = P$, on a $d * r_i = s_i * d * Q = s_i * P = s_{i+1}$.

En 2007, Dan Sumo et Niels Ferguson ont montré lors d'une conférence que cette vulnérabilité peut exister, ce qui aurait dû mettre fin à son utilisation puisque P et Q sont des paramètres donnés par les standards de la NSA, et aucune explication n'a été fournie sur la génération de ces paramètres.
Ils ont vérifié de manière expérimentale que 32 octets de sortie sont suffisants pour identifier l'état interne du générateur pseudo-aléatoire. [3]

Dans une nouvelle version de Dual EC en 2007, qu'une autre vulnérabilité a été trouvée, où il était cette fois-ci possible de retrouver tous les états précédents à partir de r_i.

C'est plus tard, en 2013 que les Snowden Leaks ont révélé que la NSA a eu un accord de 10 millions de dollars avec l'entreprise RSA pour qu'ils implémentent Dual_EC_DRBG dans leurs standards.

Il y a donc une grande suspicion sur le fait que la NSA ait volontairement introduit cette backdoor pour pouvoir cracker ces systèmes et espionner leurs utilisateurs.
En conséquence, l'utilisation de Dual EC a été retirée des standards NIST en 2014, laissant place à d'autres algorithmes utilisant d'autres courbes et d'autres paramètres.

Références

1. Bernstein DJ, Lange T, Niederhagen R (2016) Dual EC: A Standardized Back Door. In: The new codebreakers essays dedicated to David Kahn on the occasion of his 85th birthday. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 256–281

2. Hankerson, D., Menezes, A. (2011). Elliptic Curve Discrete Logarithm Problem. In: van Tilborg, H.C.A., Jajodia, S. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA. https://doi.org/10.1007/978-1-4419-5906-5_246

3. Shumow D, Ferguson N (2007) On the Possibility of a Back Door in the NIST SP800-90 Dual Ec Prng. http://rump2007.cr.yp.to/15-shumow.pdf