Algorithmes arithmétiques pour la cryptologie — 24 octobre 2016, Paris

Discussion on RSA, DSA and ECDSA

Razvan Barbulescu
CNRS et IMJ-PRG

Niveau de sécurité

Définition

On dit qu'un cryptosystème offre la sécurité s si la meilleure attaque connue requière 2^s opérations élémentaires.

Utilisation

1. On peut comparer la vitesse des différents cryptosystèmes en les réglant à la même sécurité.
2. Si on utilise ensemble de la cryptographie symétrique et asymétrique on peut les régler au même niveau de sécurité.

La loi de Moore

À cause de l'évolution des ordinateurs (loi de Moore), le même niveau de sécurité est considérer suffisant à un moment donné mais trop faible quelques années plus tard. En 2015, les principaux niveaux de sécurité sont:

- 80 bits
- 128 bits
- 256 bits.

Taille des clés RSA (1/2)

Complexité

Le meilleur algorithme pour factoriser des clés RSA est le crible algébrique (NFS).

- sa complexité est $L_N(1/3, c)^{1+o(1)}$ avec $c = \sqrt[3]{64/9} \approx 1.923$; le terme $o(1)$ est problématique pour extrapoler;
- selon un travail de Lenstra et Verheul (Selecting cryptographic key sizes, 2001), le terme $o(1)$ est petit pour les tailles cryptographique et sa dérivée est négligeable, donc on peut extrapoler sur des petits intervalles.
- il est raisonnable d'utiliser le modèle de complexité $\kappa L_N(1/3, c)$ pour une constante κ à déterminer expérimentalement.

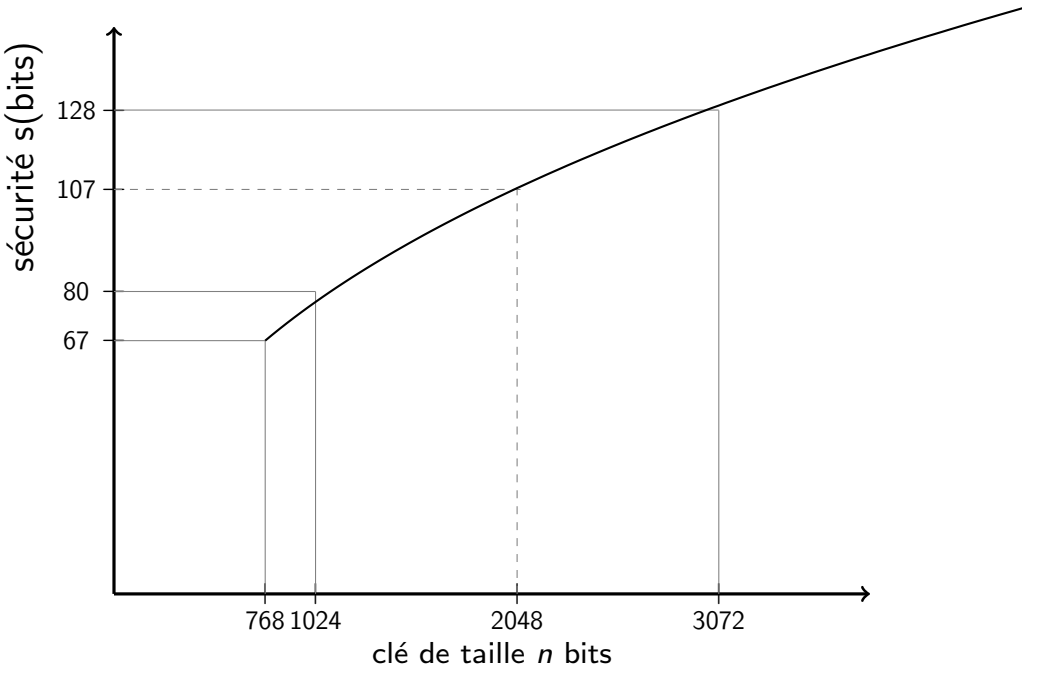
Records de factorisation

- RSA challenge, 768 bits;
- 2009, équipe commune à Nancy, Lausanne, Bonn, Tokyo, Amsterdam et Redmond;
- coût: 2000 années CPU sur des coeurs de 3GHz:

$$\log_2(3G \cdot 2000 \cdot 3.15e7) \approx 66.7.$$

RSA 768 offre une sécurité de 67 bits.

Taille des clés RSA (2/2)



Formule d'extrapolation

$$2^s = 2^{67} \frac{L_{2^n}(1/3, c)}{L_{2^{768}}(1/3, c)}$$

Recommandations gouvernementales

Mécanisme

- La NIST (National Institute of Standards and Technology) émet des spécifications, e.g. "Federal Information Processing Standards Publication 186-4" et ne valide que produits conformes à ses recommandations.
- L'ANSSI (Agence nationale de la sécurité des systèmes d'information) ne valide que les produits conformes avec le "RGS", issue tous les 2 ans.
- ENISA (European Union Agency for Network and Information Security) émet également des recommandations.

Référentiel général de sécurité version 2.0 (2014): clés RSA

1. (RègleFact-1) La taille minimale du module est de 2048 bits, pour une utilisation ne devant pas dépasser l'année 2030. (L'application d'un paradigme fondamental de la cryptographie, qui consiste à dimensionner les systèmes non pas en se plaçant juste à la limite des capacités d'attaquants connus mais en s'imposant une marge de sécurité, milite pour l'emploi de modules d'au moins 2048 bits, même si aucun module de 1024 bits n'a été officiellement factorisé à ce jour. Par conséquent, nous considérons que l'emploi de modules de 1024 bits constitue une prise de risque incompatible avec des critères de sécurité raisonnables.)
2. (RègleFact-2) Pour une utilisation au-delà de 2030, la taille minimale du module est de 3072 bits.

Autres tailles de clé

DSA (algorithme de signature digitale, basé sur le log discret dans $\mathbb{F}_p$)

Même algorithme (NFS), avec les mêmes paramètres, et ainsi la même taille de clé (RegLogp-1 et 2).

ECDSA (Elliptic curves discrete logarithm)

- le meilleur algorithme en pratique est Pollard's rho de complexité $\kappa 2^{n/2}$;
- le record actuel sur courbes elliptiques à coefficients sur un corps premier correspond à $n = 113$ dans $\approx 2^{60}$ opérations, donc $\kappa \approx 1$.
- Le RGS de l'année 2014 recommande $n = 256$ pour un niveau de sécurité de 128 bits (RègleECp-2).

Taille des clés des couplages

Les couplages ne sont pas standardisés actuellement.

Requière la difficulté de deux problèmes

1. Log discret en $\mathrm{GF}(p^n)$;
2. Log discret sur courbes elliptiques.

Degré de plongement n

Le coût du chiffrement dépend de la courbe elliptique (donc $\log p$) mais aussi du degré de plongement n. Il faut donc équilibrer le niveau de sécurité.

Taille des clés des couplages

Les couplages ne sont pas standardisés actuellement.

Requière la difficulté de deux problèmes

1. Log discret en GF(p^n);
2. Log discret sur courbes elliptiques.

Degré de plongement n

Le coût du chiffrement dépend de la courbe elliptique (donc $\log p$) mais aussi du degré de plongement n. Il faut donc équilibrer le niveau de sécurité.

- pour 80 bits de sécurité on prend $\log p \approx 160$ et $\log(p^n) \approx 1024$, donc $n = 6$;
- pour 128 bits de sécurité on prend $\log p \approx 256$ et $\log(p^n) \approx 3072$, donc $n = 12$.

Special number field sieve (SNFS)

Definition

Pour chaque d, un entier N est d-SNFS s'il existe une base de numération $m \in \mathbb{N}$ telle que $N < m^{d+1}$ et les chiffres de N en base m sont bornées par une constante absolue C.

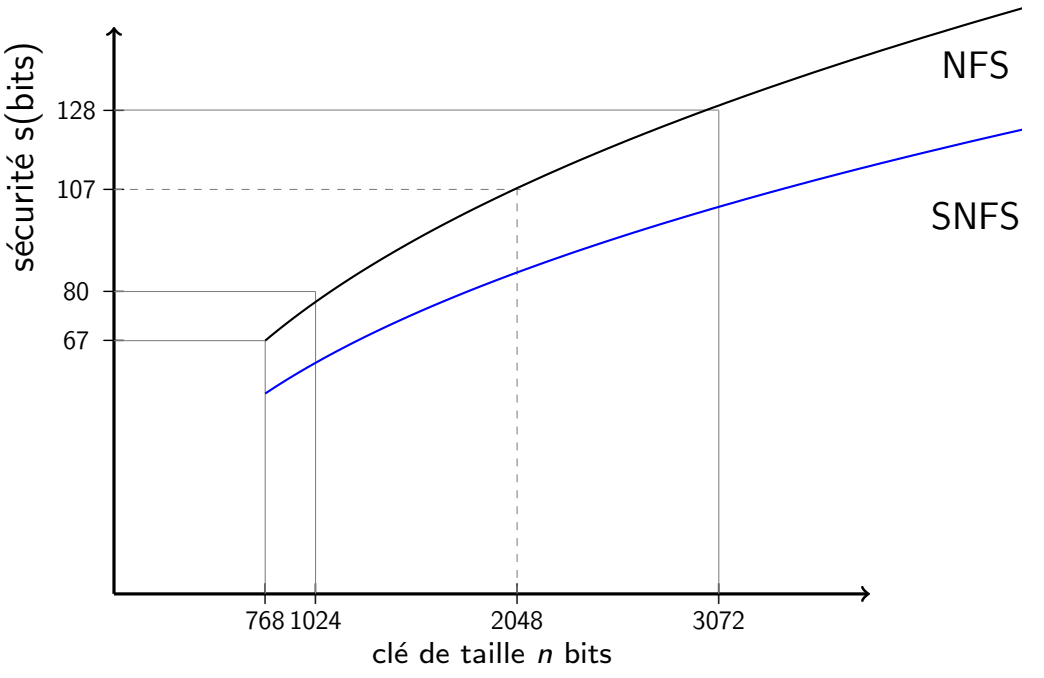
Example

- tous nombre N est 1-SNFS et $\lfloor \log N \rfloor$-SNFS, mais les nombres 6-SNFS ou 7-SNFS sont rares: C^7 au lieu de 2^x nombres de x bits.
- les nombres de type $2^n \pm 1$ sont d-SNFS pour tout d petit:

Conséquences

- Soit d le degré du polynôme f dans NFS, en pratique 6 ou 7. Les algorithmes NFS pour factorisation respectivement logarithmes discrets a une complexité $L_N(1/3, \sqrt[3]{\frac{32}{9}})$ quand on l'entrée (N ou p) est d-SNFS. À comparer avec la complexité de NFS de $L_N(1/3, \sqrt[3]{\frac{64}{9}})$.
- Aoki et al. (2007) ont factorisé un entier 6-SNFS de 1039 bits en 2^{63} opérations.

Difficulté de factoriser des modules SNFS



Formule d'extrapolation

$$2^s = 2^{63} \frac{L_{2^n}(1/3, c_{\mathrm{SNFS}})}{L_{2^{1024}}(1/3, c_{\mathrm{SNFS}})}$$

Conclusion

Si en 1985 on avait voulu choisir un groupe pour le protocole d'échange de clé de Diffie-Hellman à 80 bits de sécurité on aurait eu deux choix :

1. $(\mathbb{Z}/p\mathbb{Z})^*$. La meilleur algorithme était le Calcul d'indice (Index Calclus) de complexité $L_p(1/2, \sqrt{2})$. On aurait choisi un premier de 400 bits (car $L_{2^{400}}(1/2, \sqrt{2}) \approx 2^{80}$). Comme le Calcul d'indice n'est pas plus rapide dans le cas SNFS, on aurait pu considéré un premier de petit poids de Hamming. Aujourd'hui cela prend 32 heures CPU (http://cado-nfs.gforge.inria.fr/) pour le casser. Ça serait encore plus facile à casser si p a un petit poids de Hamming.

2. Une courbe elliptique $E(\mathbb{F}_p)$. Le meilleur attaque à l'époque était Pollard rho et on aurait choisi $\log_2 p = 160$. Pour des raisons d'efficacité on aurait pris p de faible poids de Hamming. Aujourd'hui la courbe serait toujours sûre car le record actuel est de 113 bits. Il n'existe pas (encore) d'algorithme plus rapide pour le cas où p a un faible poids de Hamming.