Introduction à la Cryptographie

6. Cryptographie asymétrique (2)

Cécile Pierrot, Chargée de Recherche INRIA Nancy
cecile.pierrot@inria.fr

Supports de E. Thomé

Telecom Nancy, 2A ISS – 2021

Vu la dernière fois

- Principe général de la clé publique.
- La primitive RSA.
- Ses propriétés désagréables. Le padding.
- Difficulté de la factorisation.

Plan

Plan

S'appuyer sur un problème mathématique

L'exemple de RSA : on s'appuie sur un problème mathématique : la factorisation.

Il y a beaucoup d'autres problèmes mathématiques qui peuvent donner lieu à des protocoles cryptographiques.

Exemple dans un groupe cyclique (fini).

- Une opération facile : la puissance.
- Une opération difficile : le logarithme discret.

Groupes

Soit $(G, \cdot, 1)$ un groupe cyclique fixé, connu de tout le monde.
G est un groupe, donc :

- Il y a une opération notée $\cdot$.
- Il y a un élément neutre noté 1.
- Chaque élément x a un symétrique x^{-1} t.q. $x \cdot x^{-1} = 1$.
- G est cyclique : $\exists g, G = \{g^i\}$.

On peut aussi noter additivement : $(G, +, 0)$: La loi est notée $+$.
L'élément neutre est noté 0. Le symétrique de x est l'opposé $-x$.

Un bon groupe pour la crypto satisfait un cahier des charges.

Condition 1 : Pas trop de bits pour écrire les éléments.
Exemple : $\lceil \log_2 n \rceil + \epsilon$ bits pour G de cardinal n.

Condition 2 : cf plus tard.

Condition 3 : cf plus tard.

Exponentiation

Dans un groupe, on peut calculer :

- Le produit de deux éléments.
  [Condition 2 : pour pas trop cher ! $\mathrm{M}(G) \leq O((\log n)^2)$].
- La puissance d'un élément : a^k (additivement le multiple).

Si $\#G = n$, alors k est défini modulo n (pourquoi ?).

Exponentiation

Dans un groupe, on peut calculer :

- Le produit de deux éléments.
  [Condition 2 : pour pas trop cher ! $\mathrm{M}(G) \leq O((\log n)^2)$].
- La puissance d'un élément : a^k (additivement le multiple).

Si $\#G = n$, alors k est défini modulo n (pourquoi ?).

Coût pour calculer a^k :

Exponentiation rapide :

- Un carré pour chaque bit de k.
- Une multiplication par bit non nul.

Logarithme discret

Le logarithme discret de $a \in G$ est l'élément $k \in \mathbb{Z}/n\mathbb{Z}$ tel que :

$$g^k = a.$$

C'est l'opération inverse de l'exponentiation.

Cadre additif : étant donné A, trouver k tel que $A = kg$.

Difficulté :

- Dans certains groupes, c'est trivial. Exemple :

Logarithme discret

Le logarithme discret de $a \in G$ est l'élément $k \in \mathbb{Z}/n\mathbb{Z}$ tel que :

$$g^k = a.$$

C'est l'opération inverse de l'exponentiation.

Cadre additif : étant donné A, trouver k tel que $A = kg$.

Difficulté :

- Dans certains groupes, c'est trivial. Exemple : $(\mathbb{Z}/n\mathbb{Z}, +)$.
- En général, c'est très difficile.

Condition3 : on veut que le log discret soit difficile.

Plan

À quoi ça sert

L'asymétrie entre exponentiation et logarithme discret sert à fabriquer des protocoles.

But du jeu :

- Tâche facile = exponentiation = tâche de l'acteur honnête.
- Tâche difficile = log discret = tâche de l'attaquant.

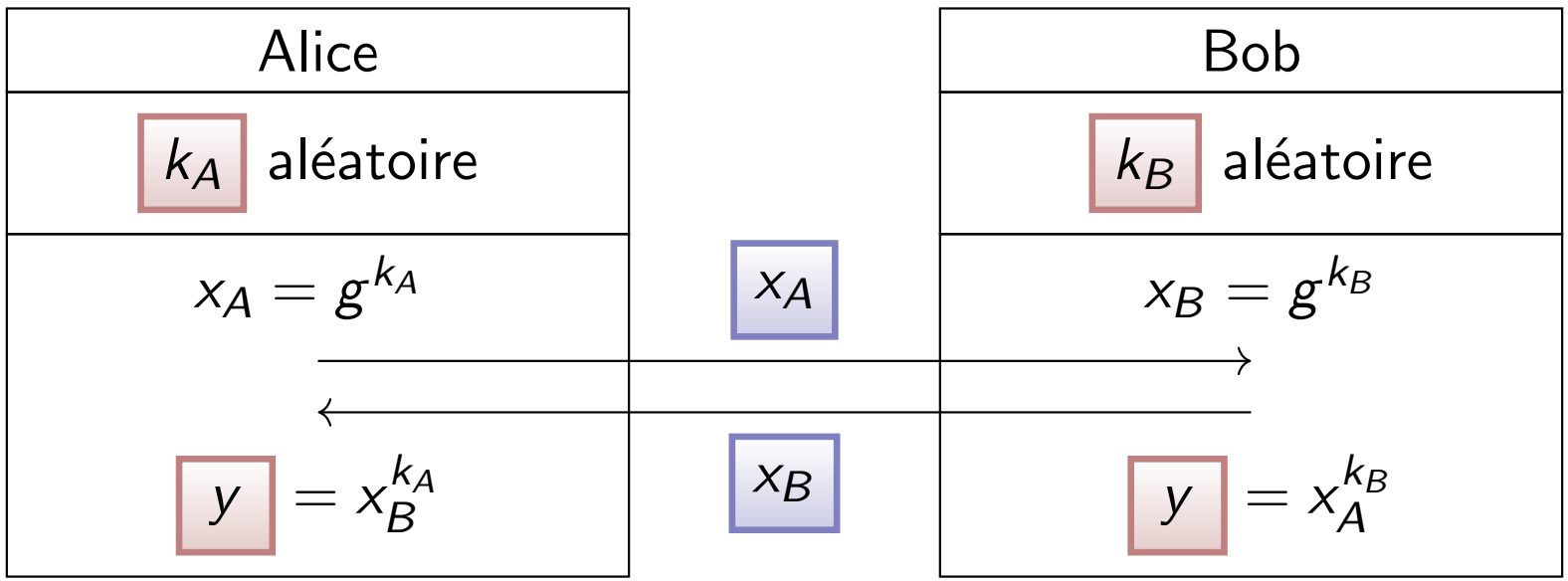
On étudie plusieurs primitives :

- Échange de clés.
- Chiffrement.
- Signature.

Diffie-Hellman

Primitive essentielle : l'échange de clés.

Donnée publique : un groupe $G = \langle g \rangle$, avec $\#G = n$.



Alice et Bob peuvent ensuite utiliser y comme clé de chiffrement pour leur communication (avec AES par exemple).

Sécurité de Diffie-Hellmann

Un espion potentiel doit retrouver $g^{k_A k_B}$ à partir de g^{k_A} et g^{k_B} (problème CDH).

- Résultats partiels d'équivalence avec $g^x \rightsquigarrow x$.
- $g^x \rightsquigarrow x$: problème du logarithme discret.
- Le choix d'un groupe adéquat est crucial.

Chiffrement : ElGamal

Donnée publique : un groupe $G = \langle g \rangle$, avec $\#G = n$.

Alice choisit au hasard une clé secrète $s \in \mathbb{Z}/n\mathbb{Z}$, et publie $h = g^s$.

Chiffrement : ElGamal

Donnée publique : un groupe $G = \langle g \rangle$, avec $\#G = n$.

Alice choisit $s \in_R \mathbb{Z}/n\mathbb{Z}$, et publie $h = g^s$.

Chiffrement : ElGamal

Donnée publique : un groupe $G = \langle g \rangle$, avec $\#G = n$.

Alice choisit $s \in_R \mathbb{Z}/n\mathbb{Z}$, et publie $h = g^s$.

Quand Bob souhaite envoyer un message m à Alice :

- Bob calcule au hasard $r \in \mathbb{Z}/n\mathbb{Z}$, et envoie $(g^r, h^r m)$.
- Alice recoit (u, v), et calcule $v \cdot u^{-s} = m$.

Tâche de l'attaquant : $(g^r, g^s) \rightsquigarrow h^r = g^{rs}$ (CDH).

Attention !

Bob doit être sérieux dans son choix d'aléa.
Le cryptosystème d'ElGamal est très demandeur d'aléa !

Signature : DSA

Donnée publique :

- Un groupe $G = \langle g \rangle$, avec $\#G = n$.
- Une fonction arbitraire $\phi : G \to \mathbb{Z}/n\mathbb{Z}$ « pas trop moche ».
- Une fonction de hachage H.

Alice : clé secrète $s \in_R \mathbb{Z}/n\mathbb{Z}$, clé publique $h = g^s$.

Signature de m : couple $(u \in \mathbb{Z}/n\mathbb{Z}, v \in \mathbb{Z}/n\mathbb{Z})$, v premier avec n, tels que :

$$\phi(g^{H(m)v^{-1}} h^{uv^{-1}}) = u.$$

Alice la calcule avec :

- Choix de $k \in_R \mathbb{Z}/n\mathbb{Z}$,
- $u = \phi(g^k)$, et $v = k^{-1}(H(m) + su)$.

Signature DSA

Job d'Alice :

- $k \in_R \mathbb{Z}/n\mathbb{Z}$. — Tirage aléatoire.
- g^k. — Une exponentiation dans G.
- $u = \phi(g^k)$. — Une application de ϕ.
- $H(m)$. — Fonction de hachage.
- $H(m) + s \cdot u$. — Produit dans $\mathbb{Z}/n\mathbb{Z}$.
- $k^{-1} \mod n$. — Inversion dans $\mathbb{Z}/n\mathbb{Z}$.
- $v = k^{-1}(H(m) + su)$. — Produit dans $\mathbb{Z}/n\mathbb{Z}$.

Job de Bob (vérifier) :

- $H(m)v^{-1}$ — Hachage, inverse, produit.
- uv^{-1} — produit.
- $g^{H(m)v^{-1}}$ — Une exponentiation dans G.
- $h^{uv^{-1}}$ — Une exponentiation dans G.
- $\phi(g^{H(m)v^{-1}} h^{uv^{-1}})$ — Un produit dans G, et une comparaison.

DSA = une horreur ?

C'est trop compliqué ?

- Oui : je suis incapable de me souvenir par cœur des formules.
- Pas tant : je sais où se trouve l'info, c'est ça qui compte.
- Compliqué à décrire n'est pas incompatible avec rapide !

Pour implanter DSA, il faut savoir :

- Calculer dans $\mathbb{Z}/n\mathbb{Z}$.
- Calculer dans G.
- Appliquer les fonctions ϕ et H.
- Faire un tirage aléatoire.

NB : Signature = éléments de $\mathbb{Z}/n\mathbb{Z}$, pas de G. Parfois nettement plus court à écrire.

Plan

Choix de groupes

On a mentionné des groupes trop simples : $G = (\mathbb{Z}/n\mathbb{Z}, +, 0)$.

Groupes plus résistants :

- Groupes multiplicatifs de corps finis : $\left(\left(\mathbb{Z}/p\mathbb{Z}\right)^*, \cdot, 1\right)$.
- Courbes elliptiques.

Ces groupes se distinguent par la taille de groupe nécessaire pour contrer les attaques sur le logarithme discret.

- Décrire un peu les groupes.
- Décrire un peu les attaques.

Éléments de comparaison : vitesse et sécurité par rapport à RSA.

Know your enemy

Les algorithmes de calcul de logarithmes discret (log. di) mettent en danger les protocoles.

- Un attaquant qui résout le pb du log. di.
  ≪ casse ≫ Diffie-Hellman.

Know your enemy

Les algorithmes de calcul de logarithmes discret (log. di) mettent en danger les protocoles.

- Un attaquant qui résout le pb du log. di. $\ll$ casse $\gg$ Diffie-Hellman.
- Un attaquant qui résout le pb du log. di. $\ll$ casse $\gg$ ElGamal.

Know your enemy

Les algorithmes de calcul de logarithmes discret (log. di) mettent en danger les protocoles.

- Un attaquant qui résout le pb du log. di. $\ll$ casse $\gg$ Diffie-Hellman.
- Un attaquant qui résout le pb du log. di. $\ll$ casse $\gg$ ElGamal.
- Un attaquant qui résout le pb du log. di. $\ll$ casse $\gg$ DSA.

Un algorithme de calcul de log. di. :

- s'applique à une famille particulière de groupes.
- a une complexité qui dépend de la taille du groupe.

Certains algorithmes sont plus spécifiques que d'autres.
La bonne taille de groupe dépend de la famille.

Préambule : la réduction de Pohlig-Hellman

Soit $(G, \cdot, 1)$ avec $\#G = n$.

- Supposons $n = p_1 \dots p_k$.
- On veut résoudre le pb du log. di. : calculer $x = \log a \in \mathbb{Z}/n\mathbb{Z}$.

Théorème des restes chinois

Calculer x = Calculer x mod p_1 + Calculer x mod p_2 + $\cdots$
Calculer x mod p_k .

Réduction de Pohlig-Hellman

Le coût du pb du log. di. dans G est au plus égal au coût du pb du log. di. dans les sous-groupes de G de cardinal $p_1, \dots, p_k$.

Moralité :

- La taille compte.
- Mais surtout, le plus grand facteur premier compte !

Les algorithmes «génériques»

Les algorithmes les moins spécifiques s'appliquent à tous les groupes.

Énoncé général : $G = \langle g \rangle$ connu, $\#G = n$; $a = g^x$, trouver x.

Exemple des « pas de bébé, pas de géant » :

- Soit $C = \lceil \sqrt{n} \rceil$. x est inconnu, mais on sait qu'il existe une écriture $x = Cy + z$, avec $y, z \leq C$.
- Soit $\mathcal{B} = \{1, g, g^2, \ldots, g^C\}$, et $\mathcal{G} = \{ag^{-C}, ag^{-2C}, \ldots, ag^{-C^2}\}$.
- Il existe $\beta \in \mathcal{B}$ et $\gamma \in \mathcal{G}$ tels que $\beta = \gamma$.
- Calculer l'intersection de ces ensembles.
- Complexité :

Les algorithmes «génériques»

Les algorithmes les moins spécifiques s'appliquent à tous les groupes.

Énoncé général : $G = \langle g \rangle$ connu, $\#G = n$; $a = g^x$, trouver x.

Exemple des « pas de bébé, pas de géant » :

- Soit $C = \lceil \sqrt{n} \rceil$. x est inconnu, mais on sait qu'il existe une écriture $x = Cy + z$, avec $y, z \leq C$.
- Soit $\mathcal{B} = \{1, g, g^2, \ldots, g^C\}$, et $\mathcal{G} = \{ag^{-C}, ag^{-2C}, \ldots, ag^{-C^2}\}$.
- Il existe $\beta \in \mathcal{B}$ et $\gamma \in \mathcal{G}$ tels que $\beta = \gamma$.
- Calculer l'intersection de ces ensembles.
- Complexité : $O(C)$ en temps, et

Les algorithmes «génériques»

Les algorithmes les moins spécifiques s'appliquent à tous les groupes.

Énoncé général : $G = \langle g \rangle$ connu, $\#G = n$; $a = g^x$, trouver x.

Exemple des « pas de bébé, pas de géant » :

- Soit $C = \lceil \sqrt{n} \rceil$. x est inconnu, mais on sait qu'il existe une écriture $x = Cy + z$, avec $y, z \leq C$.
- Soit $\mathcal{B} = \{1, g, g^2, \ldots, g^C\}$, et $\mathcal{G} = \{ag^{-C}, ag^{-2C}, \ldots, ag^{-C^2}\}$.
- Il existe $\beta \in \mathcal{B}$ et $\gamma \in \mathcal{G}$ tels que $\beta = \gamma$.
- Calculer l'intersection de ces ensembles.
- Complexité : $O(C)$ en temps, et $O(C)$ en mémoire.

Ce qui compte c'est le terme principal : on retient : $O(\sqrt{n})$.

Il existe des algorithmes en $O(\sqrt{n})$ en temps, et $O(1)$ en mémoire.

Théorème : aucun algorithme générique ne peut faire mieux.

S'il n'y a pas mieux

Si pour une famille de groupes, seuls les algos génériques s'appliquent :

- Alors pour $\#G = n = 2^m$, il faut $O(2^{m/2})$ pour un log. di.
- Pour imposer 2^{128} comme boulot à l'attaquant, il faut au moins $n \approx 2^{256}$. Sous réserve qu'il n'y ait pas de meilleur algorithme !

Corps finis

Un objet mathématique pas trop compliqué : $\mathbb{Z}/p\mathbb{Z}$.

- $\mathbb{Z}/p\mathbb{Z}$ est un corps.
- Les éléments non nuls forment un groupe.
- On veut faire de la crypto dans $(\mathbb{Z}/p\mathbb{Z}^*, \cdot, 1)$.
- Plus généralement, idem avec n'importe quel corps fini.

Avantages : • Ça fait pas trop trop mal à la tête.
Inconvénients : • Dommage, le pb du log. di. est dur, mais pas si dur que ça.

Coût du calcul de log discret dans $(\mathbb{Z}/p\mathbb{Z})^*$:

- $\exp(O((\log p)^{1/3}))$.
- Algorithme du crible algébrique. Très, très compliqué.
- Record de calcul connu aujourd'hui (2020) : 795 bits. (idem RSA !)

Taille de clés pour les corps finis

Pour les corps finis, pb du log. di. se résout en $\exp(O((\log p)^{1/3}))$.

- Si $p - 1 = \#G = 2^m$, il faut $\exp(O(m^{1/3}))$ pour résoudre le pb.
- Pour imposer 2^s comme boulot à l'attaquant, il faut m de l'ordre de s^3.
- Retenez : c'est pareil que la taille de clé pour RSA.
- La réalité : le pb du log. di est un tout petit peu plus complexe que RSA, surtout quand l'extension grandit.

Courbes elliptiques

Courbes elliptiques (ECC) sur les corps finis :

- Un peu de maths à connaître.
- Très simple au final, à la fois en matériel et logiciel.
- Log discret très difficile. L'espion a la vie rude : Aucun algorithme connu pour le log. di. autre que les génériques.

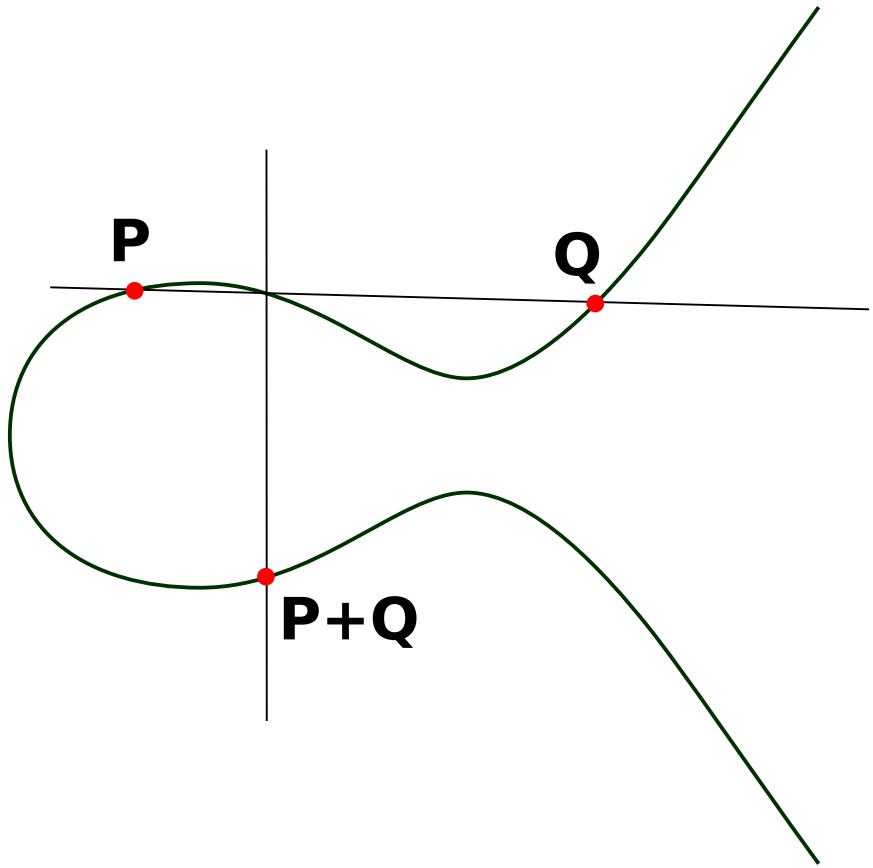
Une clé ECC : petit, mais costaud !

Pour imposer 2^{128} comme boulot à l'attaquant :

- Clé ECC : 256 bits.
- Clé RSA : 3072 bits au moins.
- Clé DSA, avec $G = (\mathbb{Z}/p\mathbb{Z})^*$: 3072 bits au moins.

Loi de groupe

Courbe elliptique : solutions de $y^2 = x^3 + ax + b$ sur $\mathbb{Z}/p\mathbb{Z}$.



Attention, on note additivement !

Une clé RSA


```
AAAAB3NzaC1yc2EAAAADAQABAAACAQCx1mguXe0GfGySnV0/CXb5PYPUwgkF
/ZRoLsbwENh8SQFXHYgC4PjxC3zlW5VowQQRgacUvT3Vfed837VPahmjjThR
6VkeoqB34yU2I06H5s6UXa4+epiKcNQNL2+7ePkGZZSPtwTMohdxMPSy17kf
IsmvaXMrnWSyjo5ggimoMAbENrF4EtVgDP8STPKDvVNuB2dEGBjJKDME2HZ3
dVX6rGv2CU2njajXNPalhZd1mCC+CBv5+67tgGPbWXT8x0JXqWso4qrSL+py
hAkq65R2HIdnUHBWPZrq7oxIV1Gp4DbRR70sb9xdbdTcxZbh5Xf/Y0lnHYY2
LaAzXCE8rKuUMiKISCYydBPGPjUsPuFbAsz6qOZ/lQTO5opK7vlmjLxfAhuo
uoymV5laqM0SonBCY8T2QUkG8iAt8ckn35QSS8OI48VuTWcgT6K2DmzXeuQ4
VnULp+h0VlLByqjrJbK8eRfbH2gloqwfJbUWJUkdjPElLlQygnMAOl8YcDTL
G9W1iuMbcSAVtZ3Qx+9bGUmibR2YVJyhhiWS+zM/VwhkEfmRQng7BepYg3jn
fQn90OTjRm2uj93CQkMvvQhMglZEIFCPCGOmJc7PBwItp7HczUrKzEBYJ3IR
mLYZFZkHyXmlA/bvy3LCOszah1sDie0uiIGIuJocaAH6skkkTrxbqw==
```

Une clé ECC

```
AAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBB7SKdUBQ
oNt8sHcrafwO2S/X/ojzuEHSEXJOAjiQRz7Tieczf+TV9oCadjkyDBTYZ5zq
TCbelHILDJNCp+aVOI=
```

Plan

Résumé

Beaucoup de choix à faire.

- Objectifs : confidentialité, authenticité, intégrité.
- Primitives : Échange de clé, Chiffrement, Signature.
- Algorithmes : clé privée, clé publique, hachage.

(et ce n'est qu'un petit morceau !)

Choix guidés par :

- Une exigence de sécurité globale. Exemple 2^{128}.
- Considérations de performance.
- Considérations de déploiement.

Performances

Performances de chiffrement : niveau de sécurité 2^{128}.

- Clé privée, AES-128 : ≈ 1 Go/s sur un cœur.
- Clé publique, RSA-3072 : 20 Mo/s (chiffrement $e = 3$), 50 Ko/s (signature).
- Clé publique, ECC-256 : 400 Ko/s (chiffrement ou signature).

Moralité :

- Échange de clés pour fabriquer une clé de session, puis AES.
- La pression en terme d'efficacité reste souvent sur le symétrique : quelques octets chiffrés en asymétrique, puis quelques mégaoctets avec AES.

Les courbes elliptiques se généralisent.