Cryptographie à clef publique

Boly SECK

<boly.seck@univ-st-etienne.fr>

d'apres un cours de mon directeur de thèse

Pierre-Louis CAYREL

<pierre.louis.cayrel@univ-st-etienne.fr>

Ecole superieure Polytechnique (ESP), Dakar/Universite Jean Monnet, Saint-Etienne

2021/2022

Séance 2

Utilisation et sécurité du RSA

Utilisation du RSA

- RSA algorithme lent (comparé aux algorithmes symétriques)
- Fonction puissance, à sens unique avec trappe, facile à calculer dans un sens (chiffrement).

$$f_{e,n}(x) = x^e \mod n$$

- Algorithme de calcul : « square and multiply »
- Notations :
  - $|e|$ taille en bits de e
  - $H(e)$ poids de Hamming de e (nombre de 1)
- Coût de l'algorithme « square and multiply » :

$$|e| \text{ « square »et } H(e) \text{ « multiply »}$$

Soit $O(|e| + H(e))$ multiplications. ($\frac{3}{2}|e|$ en moyenne)

Optimisation du chiffrement

- But : réduire le temps de chiffrement
- Degré de liberté : paramètre public e
- Idée : prendre e avec un poids de Hamming faible.
- Exemples : $\underbrace{3\ (2^1+1)}_{PEM,PKCS\#1}$, $17\ (2^4+1)$, $\underbrace{65537\ (2^{16}+1)}_{X.509,PKCS\#1}$.
- Risque avec des tailles de e trop petite (voir la suite)
- Amélioration déchiffrement : déchiffrer $\bmod p$ et $\bmod q$ et reconstruire avec le théorème chinois.
  $\Rightarrow$ gain en temps d'un facteur 4.

Rappels

- Casser RSA $\Leftrightarrow$ factoriser n
- Recommandations : $|n| \geq 768$ bits
- Question :

  « Sachant qu'on ne sait pas factoriser (en un temps humainement raisonnable) des nombres de plusieurs milliers de bits, est ce que le RSA est sûr ? »

- Deux réponses :
  - Oui : En temps qu'algorithme et pour le moment
  - Pas forcément : en tant que protocole (facteur humain)
- Bruce Schneier :

  « Il ne suffit pas d'utiliser le RSA, les détails comptent »

Exemples de faiblesse d'utilisation

Attaque par module commun :

Caractéristiques

Donnons le même n à tout le monde. (facilité de gestion des paramètres secrets)
Choix de e_i et d_i différents pour chaque utilisateur.
En principe e_1 et e_2 premiers entre eux.
Chiffrement du même message $\mathcal{M}$. $\begin{cases} C_1 = \mathcal{M}^{e_1} \mod n \\ C_2 = \mathcal{M}^{e_2} \mod n \end{cases}$
Bézout : $r \times e_1 + s \times e_2 = 1$ avec $r \leq 0$ par exemple.
Calcul de $G = C_1^{-1} \mod n$ (Euclide étendu).
Puis calcul de

$$\begin{aligned} G^{-r} \times C_2^s \mod n &= C_1^r \times C_2^s \mod n \\ &= (\mathcal{M}^{e_1})^r \times (\mathcal{M}^{e_2})^s \mod n \\ &= \mathcal{M}^{e_1 \times r + e_2 \times s} \mod n \\ &= \mathcal{M} \end{aligned}$$

Exemples de faiblesse d'utilisation

Attaque par exposant **commun faible**

Caractéristiques

Même exposant public $e = 3$ (taille faible).
Même message $\mathcal{M}$ envoyé à des clients différents avec des clefs différentes (les moduli).
Attaque en prenant la racine cubique réelle.

$e = 3$, $n_1 = 143$, $n_2 = 391$, $n_3 = 899$. (Les n_i premiers entre eux deux à deux, forcément ! pourquoi ?)
Message $\mathcal{M} = 135$ (forcément plus petit que le plus petit des moduli).
$C_1 = 60$, $C_2 = 203$ et $C_3 = 711$.

Attaquant :

1. Calcul de x_1, x_2 et x_3 tels que : $\begin{cases} x_1 n_2 n_3 \equiv 1 \mod n_1 \Rightarrow x_1 = -19 \\ n_1 x_2 n_3 \equiv 1 \mod n_2 \Rightarrow x_2 = -62 \\ n_1 n_2 x_3 \equiv 1 \mod n_3 \Rightarrow x_3 = 262 \end{cases}$
2. Construction de $C = C_1 x_1 n_2 n_3 + C_2 n_1 x_2 n_3 + C_3 n_1 n_2 x_3 \mod n_1 n_2 n_3$ alors $\forall i,\ C \equiv C_i \mod n_i \equiv \mathcal{M}^e \mod n_i$ et $C = 2460375 = \mathcal{M}^3$
3. $\mathcal{M} = C^{1/3} = 2460375^{1/3} = 135$

Exemples de faiblesse d'utilisation

Attaque par petit exposant commun (plus forcément le même message chiffré)

Principe

e petit et commun.
Chiffrement de $\frac{e(e+1)}{2}$ messages linéairement **dépendants** avec des clés publiques différentes.
Attaque possible du système.

S'il n'y a pas autant de messages ou qu'ils sont linéairement indépendants, il n'y a pas de problème.

Exemples de faiblesse d'utilisation

Attaque par petit exposant de déchiffrement.

- Motivation d'utiliser un petit exposant de déchiffrement : accélerer le temps de déchiffrement.

Attaque de Michael WIENER

Cas où $|d|$ est le quart de la longueur du module et e inférieur à n. N'arrive que si e est petit.

Principe basé sur le développement en fraction continue de $\frac{2e}{n}$.

Parades, recommandations

- Casser RSA$\Leftrightarrow$ factoriser n
  - Choix de $n = pq$ assez grand.
  - Choix de p et q aléatoires de manière à résister aux meilleures algorithmes de factorisation connus à ce jour (GNFS : Méthode du crible de corps de nombre généralisée)
- Attention à l'utilisation de protocoles impliquant RSA !
  - Ne partager jamais n parmi un groupe d'utilisateurs
  - Combler les messages par remplissage d'aléas avant de les chiffrer. $\mathcal{M}$ doit avoir la même taille que n.
  - Choisir une grande valeur pour d
- La sécurité est une chaîne dont la solidité est celle de son maillon le plus faible.
- Autre famille d'attaques (voir les signatures numériques) à texte chiffré choisi impliquant les signatures numériques basées sur le protocole RSA.

Attaques matérielles

- Supposons que l'on ait pris toutes les mesures précédentes, est-on protégé ?
- Rappel : protégé contre quoi ?
- P. Kocher : Attaques par canaux auxiliaires (« side channel attacks »).
  - Timing-attack
  - SPA : Simple Power Analysis
  - DPA : Differential Power Analysis
- Conclusion : Attention (selon le niveau de paranoïa et de menaces réelles) aux algorithmes de haut niveau utilisés, à l'implantation matérielle, la technologie utilisée (portes consommant la même quantité de courant), les conditions d'utilisation (cage de Faraday) !

Autres algorithmes de cryptographie asymètrique

Cryptosystème de Rabin

Problème mathématique

Trouver des racines carrées modulo un nombre composé (n) est équivalent à factoriser n.

1. Choisir deux nombres premiers p et q (clé secrète) tels que $p \equiv q \equiv 3 \mod 4$.
2. $n = pq$: clé publique
3. Chiffrement : $C = \mathcal{M}^2 \mod n$
4. Déchiffrement : Trouver les racines carrées de C modulo n (Combien y en a-t-il ?)
5. L'une de ces racines est le message clair.

Déchiffrement de Rabin

- Calcul de $\pm m_p = C^{\frac{p+1}{4}} \mod p$
- Calcul de $\pm m_q = C^{\frac{q+1}{4}} \mod q$
- Calcul de $a = q \times (q^{-1} \mod p)$ et $b = p \times (p^{-1} \mod q)$
- Calcul de

$$\begin{cases} \mathcal{M}_1 = (am_p + bm_q) \mod n \\ \mathcal{M}_2 = (am_p - bm_q) \mod n \\ \mathcal{M}_3 = (-am_p + bm_q) \mod n \\ \mathcal{M}_4 = (-am_p - bm_q) \mod n \end{cases}$$

- L'un des 4 $\mathcal{M}_i$ est le message cherché.
- Probabilité forte que les trois autres soient complètement incompréhensibles.
- Attention ! La fonction de chiffrement n'est donc pas injective (plusieurs textes clairs différents peuvent donner un même texte chiffré).

Cryptosystème d'El Gamal

- Problème mathématique : Résolution du logarithme discret dans un corps fini.

Problème du logarithme discret dans $\mathbb{Z}/p\mathbb{Z}$

Soit p un nombre premier.
Soit g un élément générateur de $(\mathbb{Z}/p\mathbb{Z})^*$ (qui est cyclique).
Étant donné, $y \in (\mathbb{Z}/p\mathbb{Z})^*$ trouver x tel que $g^x = y \mod p$ est appelé problème du logarithme discret dans $\mathbb{Z}/p\mathbb{Z}$.

- Génération des clés :
  1. Bob choisit $a \in \{1, \ldots, n-2\}$ et calcule $A = g^a \mod p$
  2. Bob publie A, g, p
  3. Bob garde secret a

Chiffrement El-Gamal

Chiffrement

Alice veut envoyer le message $\mathcal{M}$ à Bob.

1. Alice récupère la clé publique de Bob (A, g, p).
2. Alice choisit au hasard un nombre $k \in \{1, \ldots, n-2\}$ tel que k et $p-1$ soient premiers entre eux.
3. Alice calcule :

$$\begin{cases} y_1 &= g^k \mod p \\ y_2 &= \mathcal{M} \cdot A^k \mod p \end{cases}$$

4. Alice envoie $C = (y_1, y_2)$ à Bob.

Remarque : la taille du chiffré est le double de la taille du clair...

Déchiffrement

- Bob reçoit le message chiffré $C = (y_1, y_2)$.
- Bob déchiffre en calculant : $y_2 \cdot (y_1^a)^{-1} \mod p$

Sécurité de El-Gamal

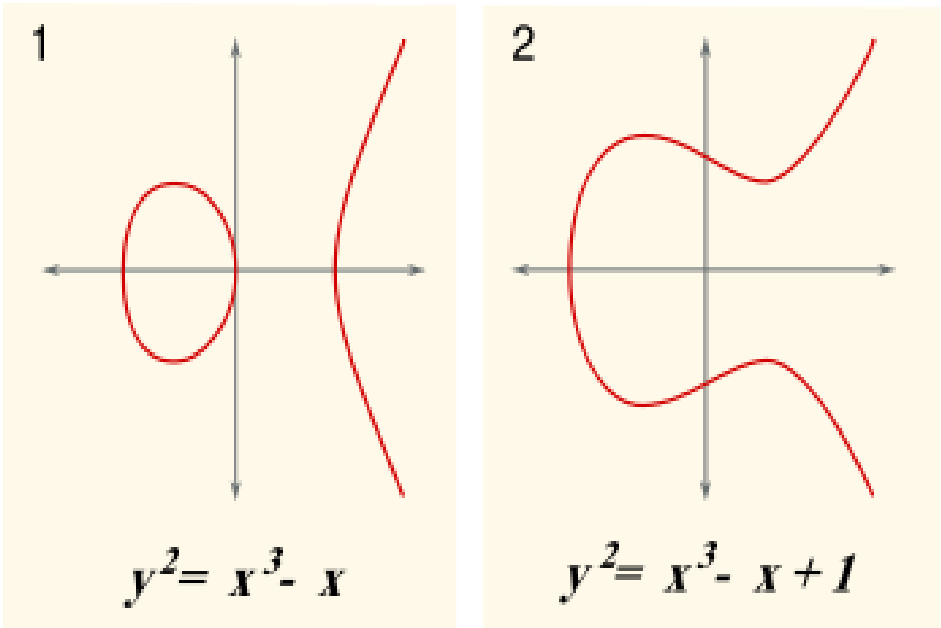
- Résoudre le logarithme discret permet de casser El-Gamal
- On ne sait pas prouver la réciproque
- Recommandations (cas de $(\mathbb{Z}/p\mathbb{Z})^*$) :
  - Choisir p, nombre premier aléatoire de 1024 bits pour résister aux méthodes connues de résolution du problème du logarithme discret dans $(\mathbb{Z}/p\mathbb{Z})^*$.

Cryptographie asymétrique : les limitations

- Taille des clés : plusieurs milliers de bits.
- Calculs de « haut niveau »$\Rightarrow$ temps de chiffrement élevé
- Performance croissante des méthodes de résolution des problèmes mathématiques (factorisation, log discret, . . .) $\Rightarrow$ augmentation de la taille des clés.
- Idée : chercher des objets mathématiques qui ont une structure de groupe (la plupart des algorithmes sont généralisables pour des groupes) mais dans lesquels les algorithmes connus sont moins efficaces.

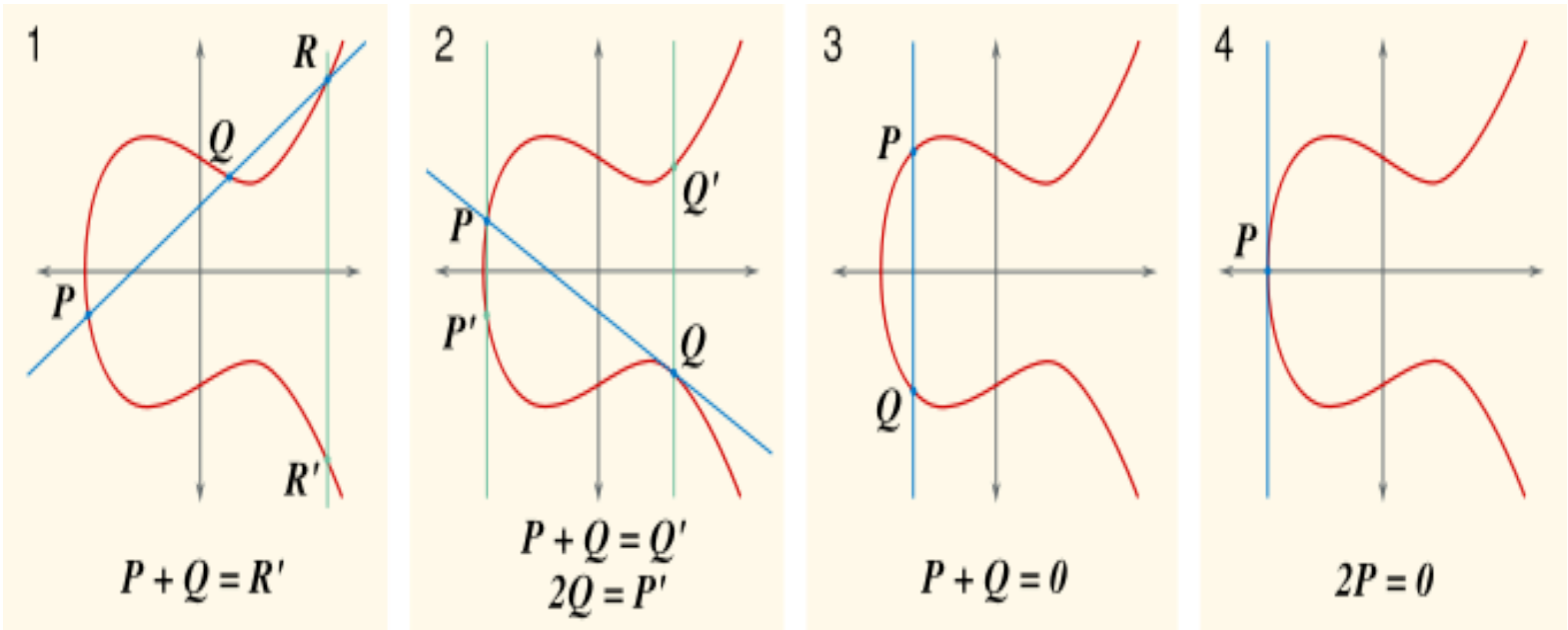
Courbe elliptique sur les corps finis

- $\mathbb{F}$ corps fini à q éléments, $a, b \in \mathbb{F}$
- Équation générale (forme réduite) : $y^2 = x^3 + ax + b$.



Structure de groupe sur courbe elliptique

- Définition de la somme de points sur une courbe elliptique



Avantages

- Diminution de la taille des clés : RSA-1024 bits $\approx$ ECC-160 bits
- Construction des corps finis peut permettre l'accélération des calculs de bases. Exemple : Corps fini à 2^n éléments vu comme $(\mathbb{Z}/2\mathbb{Z})/P_n(X)$ où P_n polynôme irréductible de degré n.
- Algorithme de résolution du log discret plus compliqué dans le cas des courbes elliptiques.
- Contrepartie : $\mathcal{P}+\mathcal{Q}$ implique de nombreux calculs sur les coordonnées (cf : formules)

Coordonnées ($x \neq x'$)

Soit $\mathcal{P}=(x,y)$ et $\mathcal{Q}=(x',y')$, alors $\mathcal{R}=\mathcal{P}+\mathcal{Q}$ a pour coordonnées :

$$\begin{aligned}\mathcal{R} &= (x'',y'') \\ &= \left(\left(\frac{y-y'}{x-x'}\right)^2 - x - x', -\frac{y-y'}{x-x'}\left(\left(\frac{y-y'}{x-x'}\right)^2 - x - x'\right) - \frac{x'y-xy'}{x-x'}\right)\end{aligned}$$

Systèmes de coordonées

- Représentation des coordonnées des points : cartésiennes, projectives, jacobiennes, jacobiennes modifiées
- But : améliorer les performances en temps de calcul.
- Exemple : entre 12 et 20 (selon les représentations) multiplications modulaires par bit d'exposant.
  - RSA-1024 : $1,5 \times 1024$ multiplications 1024 bits
  - ECC-160 : $1,5 \times 16 \times 160$ multiplications 160 bits
  - $T(MulMod_{1024}) \approx 6,4^2 \times T(MulMod_{160})$
  - $T(RSA-1024) \approx 62914,56 \times T(MulMod_{160})$
  - $T(ECC-160) \approx 3840 \times T(MulMod_{160})$
- A niveau de sécurité équivalent, RSA 16 fois plus lent que ECC.

D'autres problèmes mathématiques $\Rightarrow$ d'autres cryptosystèmes

Problème du sac à dos

Peut-on remplir un sac à dos ne pouvant pas supporter plus d'un certain poids avec des objets ayant chacun un poids et une valeur, en ayant pour objectif de maximiser la valeur sans dépasser le poids ?

- Problème *NP* complet sous sa forme décisionnelle. (Il n'existe pas de solution générale répondant par oui ou par non à la question)
- A la base du chiffrement de Merkle-Hellman (cassé par Shamir en 1984).
- Variante (Chor-Rivest 1988) pas encore cassée.

Problème du décodage

Comment peut on décoder un code correcteur d'erreur linéaire général ?

- Chiffrement de McEliece