UNIVERSITÉ LAVAL

# Cryptographie asymétrique

MOHAMED MEJRI

*Groupe LSFM*

*Département d'Informatique et de Génie Logiciel*

*Université LAVAL*

*Québec, Canada*

# Plan

- RSA
  - Introduction
  - Génération de clés
  - Cryptage/décryptage
  - Signature
  - Pourquoi ça marche ?
  - RSA en pratique
- Chiffrement d'El-Gamal sur $(\mathbb{Z}_P^*, \times)$
- Chiffrement d'El-Gamal sur les courbes elliptiques
- Cryptographie symétrique vs. asymétrique

# RSA : Introduction

RSA : Introduit par Rivest, Shamir et Adleman en 1978.

⇨ idée : Sous certaines conditions sur $e$, $d$ et $n$, on aura que pour tout $m < n$ :

$$\boxed{m^{ed} \bmod n \quad = \quad m}$$

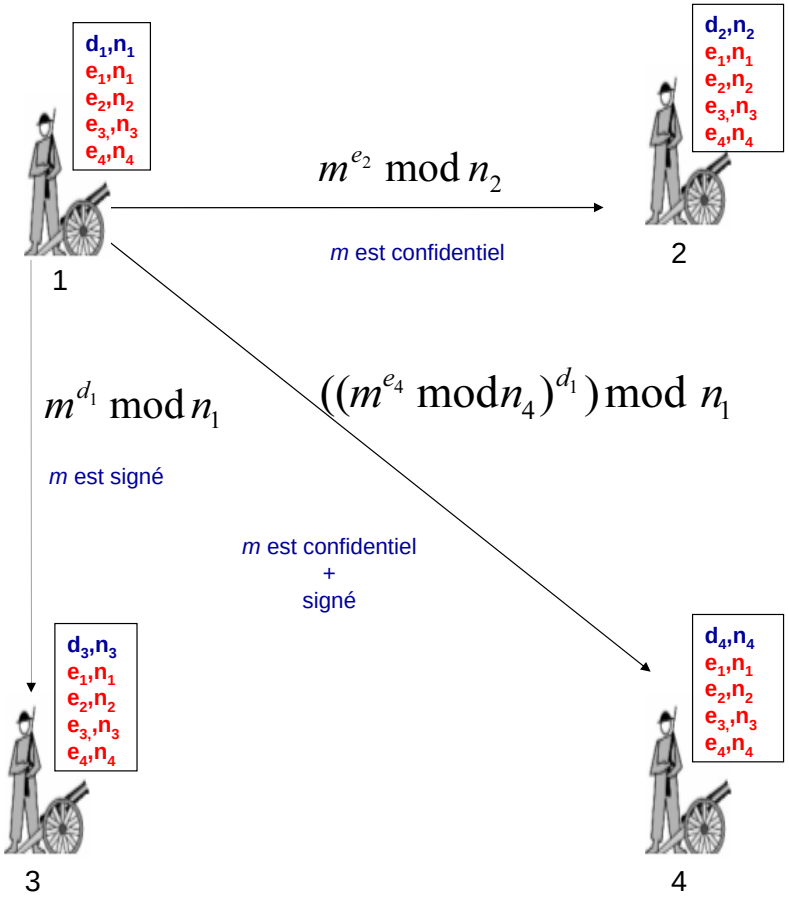
- Exemple : Si $e = 3$, $d = 7$ et $n = 33 \Rightarrow \forall\, m \in \{0, \ldots, 32\}$, alors

$$m^{21} \bmod 33 = m$$

- Encrypter un message $m$ : $\Rightarrow$ calculer $c = m^e \bmod n$
- Décrypter un message $c$ : $\Rightarrow$ calculer $c^d \bmod n$

$$\begin{aligned} c^d \bmod n &= (m^e \bmod n)^d \bmod n \\ &= m^{ed} \bmod n \\ &= m \end{aligned}$$

# RSA : Introduction



## RSA : Génération des clés

1. Choisir deux grands nombres entiers premiers $(p, q)$
2. Calculer leur produit $n = p * q$
3. Calculer $\Phi(n) = (p-1) * (q-1)$
4. Choisir e tel que : $pgcd(e, \Phi(n)) = 1$ et $e < \Phi(n)$
5. Calculer $d$ tel que $e * d \equiv 1 \ mod \ \Phi(n)$ ($d$ existe et il est unique)
6. Clé privée= $(n, d)$ (les valeurs de $p$, $q$ et $\Phi(n)$ sont gardées secrètes)
7. Clé publique= $(n, e)$

# RSA : Cryptage/Décryptage

➪ Cryptage : Pour chiffrer un message et l'envoyer à Bob, Alice a besoin d'avoir le message clair $m$ et la clé publique $(e_b, n_b)$ de Bob.

- $0 \leq m < n$, le message clair
- Le message chiffré est : $e_{k_{pb}}(m) = c = m^{e_b} \ mod \ n_b$

➪ Décryptage : Bob a besoin d'avoir le message chiffré $c$ et sa clé privée $(d_b, n_b)$ (correspondante à celle utilisée pour le chiffrement s'il en a plusieurs)

- $0 \leq c < n$, le message crypté
- Le message original est $d_{k_{pr}}(c) = m = c^{d_b} \ mod \ n_b$

# RSA : Signature

➪ Signature : Pour signer un message, Alice a besoin d'avoir le message clair $m$ et sa clé privée $(d_a, n_a)$

- $0 \leq m < n$, le message à signer
- Le message signé est : $e_{kpr}(m) = sig = m^{d_a} \ mod \ n_a$

➪ Vérification de la signature : Pour vérifier un message signé par Alice, Bob a besoin d'avoir le message $m$, sa signature $sig$ et la clé publique $(e_a, n_a)$ de Alice (celle correspondante à la clé privée utilisée pour la signature).

- Est ce que $c$ est le message $m$ signé par $k_{pr}$ ?
- La réponse est vraie ssi : $d_{k_{pb}}(c) = m = sig^{e_a} \ mod \ n_a$
- Autrement : $estValide := (m == sig^{e_a} \ mod \ n_a)$

# RSA : Exemple

- Alice veut envoyer à Bob le message $m = 4$

| *Alice* | | *Bob* |
|---|---|---|
| | | (1) $p = 3,\ q = 11$ |
| | | (2) $n = p * q = 33$ |
| | | (3)$\Phi(n) = (3-1) * (11-1) = 20$ |
| | | (4)$e = 3,\ pgcd(3, 20) = 1$ |
| $m = 4$ | $k_{pb} = (3, 33) \longleftarrow$ | (5)$d = e^{-1} = 7\ mod\ 20$ |
| $c = m^e\ mod\ 33$<br>$= 4^3\ mod\ 33$<br>$= 64\ mod\ 33$<br>$= 31$ | $\longrightarrow c = 31$ | $m = c^d\ mod\ 33$<br>$= 31^7\ mod\ 33$<br>$= 4$ |

# RSA : Exemple

➪ Alice veut envoyer à Bob le message $m = 688232687966668$

| *Alice* | | *Bob* |
|---|---|---|
| | | (1) $p = 47,\ q = 71$ |
| | | (2) $n = p * q = 3337$ |
| | | (3)$\Phi(n) = (47 - 1) * (71 - 1) = 3220$ |
| | | (4)$e = 79,\ pgcd(79, 3220) = 1$ |
| $m = 688232687966668$<br>$\Longrightarrow$ découpage<br>$m_1 = 688;\ m_2 = 232;$<br>$m_3 = 687;\ m_4 = 966;$<br>$m_5 = 668$ | $k_{pb} = (79, 3337) \longleftarrow$ | (5)$d = e^{-1} = 1019\ mod\ 3220$ |
| $c_1 = 688^{79} mod\ 3337 = 1570$<br>$c_2 2 = 232^{79} mod\ 3337 = 2756;$<br>$c_3 = 687^{79} mod\ 3337 = 2091;$<br>$c_4 = 966^{79} mod\ 3337 = 2276;$<br>$c_5 = 668^{79} mod\ 3337 = 2423$ | $\longrightarrow c_1 c_2 c_3 c_4 c_5$ | $m_1 = 1570^{1019} mod\ 3337 = 668$<br>$m_2 = 2756^{1019} mod\ 3337 = 232$<br>$m_3 = 2091^{1019} mod\ 3337 = 687$<br>$m_4 = 2276^{1019} mod\ 3337 = 966$<br>$m_5 = 2423^{1019} mod\ 3337 = 668$<br>$m = m_1 m_2 m_3 m_4 m_5$<br>$= 688232687966668$ |

## RSA : Pourquoi ça marche

– Pour que RSA fonctionne, il faut que pour tout $0 \leq m \leq n-1$ :

$$d_{kpr}(e_{kpb}(m)) = m$$

– Or

$$\begin{aligned} d_{kpr}(e_{kpb}(m)) &= (m^d \ mod \ n)^e \ mod \ n \\ &= (m^{d*e} \ mod \ n) \ mod \ n \\ &= m^{d*e} \ mod \ n \end{aligned}$$

– Donc, pour que RSA fonctionne, il suffit que :

$$m^{d*e} \ mod \ n = m \ mod \ n$$

– Puisque $d * e = 1 \ mod \ \Phi(n)$, donc il existe un entier $k$ tel que :

$$d * e = 1 + k * \Phi(n)$$

## RSA : Pourquoi ça marche

– On déduit que pour que RSA fonctionne, il suffit que :

$$m^{1+k*\Phi(n)} \; mod \; n = m \; mod \; n$$

– Pour atteindre cet objectif, il suffit de démontrer que :

$$m^{\Phi(n)} \; mod \; n = 1 \; mod \; n$$

– Définition (Totient d'Euler) : $\Phi(n)$ = nombre d'entiers positifs dans $\{1, \ldots, n-1\}$ et relativement premiers avec $n$.
  – Pour p premier, $\Phi(p) = p - 1$
  – Pour $p$ et $q$ premiers ($p \neq q$), $n = p * q$ et $\Phi(n) = \Phi(p) * \Phi(q) = (p-1) * (q-1)$
– Théorème (Euler) : Si $m$ et $n$ sont relativement premiers alors

$$m^{\Phi(n)} \equiv 1 \; mod \; n$$

## RSA : Pourquoi ça marche

- Question : Quoi faire si m et n ne sont pas relativement premiers ?
- Réponse : Le théorème d'Euler tient encore.
- Preuve :
  - Supposant que :

$$pgcd(m, n) = pgcd(m, p * q) \neq 1$$

  - Puisque $p$ et $q$ sont premiers, donc il existe un entier $r < p$, $r < q$ tel que :

$$(m = r * p) \text{ ou } (m = r * q)$$

  - Supposons que $m = r * p$ (la preuve sera similaire pour le cas où $m = r * q$), dans ce cas on a :

$$pgcd(m, q) = 1 \text{ (autrement } p \text{ et } q \text{ auront un diviseur commun)}$$

# RSA : Pourquoi ça marche

- Preuve (suite) :
  - Puisque $m$ et $q$ sont relativement premiers, donc d'après le théorème d'Euler, on aura :

$$m^{\Phi(n)} = m^{(p-1)(q-1)} = (m^{\Phi(q)})^{(p-1)} = 1 \; mod \; q$$

  - On déduit qu'il existe un entier $k$ tel que :

$$m^{\Phi(n)} = 1 + k * q$$

  - Puisque $m = r * p$, on déduit que :

$$m*m^{\Phi(n)} = m*(1+k*q) = m+m*k*q = m+r*p*k*q = m+r*k*n = m \; mod \; n$$

  - On déduit que :

$$m^{\Phi(n)} \equiv 1 \; mod \; n$$

## RSA en pratique

- Comment trouver deux grands nombres premiers $p$ et $q$ ?
  - Test probabiliste de primalité
- Comment trouver e premier avec $\Phi(n)$ ?
  - Algorithme d'Euclide
- Comment déterminer $d$ à partir de $e$ ?
  - Algorithme d'Euclide étendu
- Comment effectuer les gros calculs $x^k mod\ n$ ?
  - Algorithmes d'exponentiation

## RSA en pratique : test de primalité de Miller-Rabin

- On veut tester si $p$ (choisi au hasard) est premier ou non.
- On doit choisir $p$ impair, sinon il est clair qu'il n'est pas premier.
- Soit $p - 1 = 2^k * m$ avec $m$ est un nombre impair.
- Autrement dit $k$ est le plus grand nombre entier tel que $2^k$ divise $p - 1$.
- **Théorème :** Si p est premier, alors pour tout $a$ tel que $pgcd(p, a) = 1$, alors :

$$\begin{cases} a^m \equiv 1 \; mod \; p \\ \textbf{ou} \\ \exists \; r \in \{0, \ldots, k-1\} \mid a^{2^r m} \equiv -1 \; mod \; p \end{cases}$$

# RSA en pratique : test de primalité de Miller-Rabin

– **Corollaire :** S'il existe $a \in \{2, \ldots, p-1\}$ premier avec $p$ (pgcd(a,p)=1) tel que :

$$\begin{cases} a^m \not\equiv 1 \; mod \; p \\ \textbf{et} \\ \forall \; r \in \{0, \ldots, k-1\} \mid a^{2^r m} \not\equiv -1 \; mod \; p \end{cases}$$

Alors, $p$ **n'est pas premier** ( $a$ est appelé un **témoin**).

– **Théorème :** Si $a \in \{2, \ldots, p-1\}$ (une valeur choisie au hasard) n'est pas un témoin pour $p$ alors :

$$Pr(p \text{ n'est pas premier}) \leq 0.25$$

– **Conséquence :** Si, après avoir choisi aléatoirement $\{a_1, \ldots, a_t\} \subseteq \{2, \ldots, p-1\}$, aucune des ces valeurs n'est un témoin pour $p$ alors :

$$Pr(p \text{ est premier}) \geq 1 - (\tfrac{1}{4})^t \qquad (\text{ c-à.d. } Pr(p \text{ n'est pas premier}) \leq (\tfrac{1}{4})^t)$$

$< t = 5 : 1 - (\frac{1}{4})^t \approx 0.999 >; < t = 8 : 1 - (\frac{1}{4})^t \approx 0.99999 >$

## RSA en pratique : test de primalité de Miller-Rabin

- Question : Combien y en a-t-il de nombres premiers inférieurs à $n$ ?
- Réponse : $\approx \frac{n}{ln(n)} \Rightarrow$ il y a une infinité de nombres premiers dans $\mathbb{N}$
- Exemple : $\approx 10^{152}$ nombres premiers de $512$ bits ou moins $\Rightarrow$ plus que le nombre d'atomes dans l'univers $\approx 10^{77}$
- Question : Quelle est la probabilité pour qu'un nombre $x$, $x \leq n$, choisi au hasard soit premier ?
- Réponse : $Pr(x \text{ choisi au hasard soit premier} | x \leq n) \approx \frac{1}{ln(n)}$
- Exemple : $Pr(x \text{ choisi au hasard soit premier} | x \leq 2^{250}) \approx \frac{1}{75} \Rightarrow$ cela pourrait prendre du temps avant de trouver un nombre premier.

## RSA en pratique : algorithme d'Euclide

- Problème : trouver $e < \Phi(n)$ tel que : $pgcd(e, \Phi(n)$
- Solution :
  - Choisir un nombre (assez grand) $e < \Phi(n)$
  - Appliquer l'algorithme d'Euclide pour calculer $pgcd(e, \Phi(n))$
    - Si $pgcd(e, \Phi(n)) = 1$ on arrête
    - Sinon, on choisit une autre valeur de $e$ et on répète le même traitement.
- Remarque : L'algorithme d'Euclide est basé sur le fait suivant :
  - Si $a = q * b + r$ alors le $pgcd(a, b) = pgcd(b, r)$.
    - En effet, tout diviseur de $a$ et de $b$ est également un diviseur de $r$.

# RSA en pratique : algorithme d'Euclide

➪ $pgcd(a, b) = ?$

| Algorithme |
|---|
| $pgcd(a, b)$ |
| $x := a\,;\, y := b$ |
| While $y \neq 0$ |
| $r := x \ mod \ y\,;$ |
| $x := y\,;\, y := r\,;$ |
| end while |
| return(x) |

| Explication | |
|---|---|
| $r_0 = a$ | |
| $r_1 = b$ | |
| $r_0 = q_1 * r_1 + r_2$ | $0 < r_2 < r_1$ |
| $r_1 = q_2 * r_2 + r_3$ | $0 < r_3 < r_2$ |
| $\vdots$ | $\vdots$ |
| $r_i = q_{i+1} * r_{i+1} + r_{i+2}$ | $0 < r_{i+2} < r_{i+1}$ |
| $r_{n+1} = 0 \Rightarrow pgcd(a, b) = pgcd(r_n, 0) = r_n$ | |

# RSA en pratique : algorithme d'Euclide étendu

➪ Problème :

- Trouver d tel que $e * d = 1 \; mod \; \Phi(n)$
- Autrement, il suffit de trouver $d$ et $k$ tels que :

$$e * d + k * \Phi(n) = 1 = pgcd(e, \Phi(n))$$

➪ Solution :

- Algorithme d'Euclide étendu
- Étant donnés deux entiers $r_0$ et $r_1$, il permet de trouver deux autres entiers $s$ et $t$ tels que :

$$r_0 * s + r_1 * t = pgcd(r_0, r_1)$$

# RSA en pratique : algorithme d'Euclide étendu

⇨ Calcul de $s$ et $t$ :

- Soit $q_i$ le $i$-ème quotient du calcul de $pgcd(r_0, r_1)$
- Soit $m$ le nombre d'étapes que le calcul de $pgcd(r_0, r_1)$ prend.
- Alors $s = s_{m+1}$ et $t = t_{m+1} = r_1^{-1}$
- Avec :

| | | |
|---|---|---|
| $s_0 = 1$ | $t_0 = 0$ | |
| $s_1 = 0$ | $t_1 = 1$ | |
| $s_i = s_{i-2} - q_{i-1} * s_{i-1}$ | $t_i = t_{i-2} - q_{i-1} * t_{i-1}$ | $i = 2, 3, \ldots$ |

# RSA en pratique : Algorithme d'exponentiation

➪ Exemple : Supposant qu'on veut calculer $a^{17}$

- Au lieu de faire 16 multiplications

$$a^{17} = \underbrace{a \times \ldots \times a}_{17 \text{ fois}}$$

- On fait juste 5 multiplications

$$a^{17} = (((a^2)^2)^2)^2 \times a$$

➪ Généralisation : Pour calculer $a^m$, avec $a$ et $m$ des entiers positifs, on procède ainsi :

- Déterminer la représentation binaire de $m = b_k + b_{k-1} + \ldots + b_0$

$$m = \sum_{b_i \neq 0} 2^i$$

- Calculer

$$a^m = a^{\sum_{b_i \neq 0} 2^i} = \prod_{b_i \neq 0} a^{2^i}$$

# RSA en pratique : Algorithme d'exponentiation

➪ Algorithme : Calcul de $a^m \ mod \ n$ avec $m = b_k 2^k + b_{k-1} 2^{k-1} + \ldots + b_0 2^0$.

$z := 1$

pour $i = k$ jusqu'à 0 faire

  $z := z \times z \ mod \ n$

  si $b_i = 1$ alors $z := z \times a \ mod \ n$

return$(z)$

➪ Exemple : $m = 11 = 2^3 + 2 + 1$

| | | |
|---|---|---|
| $i = 3$ | $b_3 = 1$ | $z = z \times z \times a \ mod \ n$ |
| $i = 2$ | $b_2 = 0$ | $z = z \times z = a^2 \ mod \ n$ |
| $i = 1$ | $b_1 = 1$ | $z = z \times z \times a \ mod \ n = a^5 \ mod \ n$ |
| $i = 0$ | $b_0 = 1$ | $z = z \times z \times a \ mod \ n = a^{11} \ mod \ n$ |

## Sécurité de RSA

- Attaquer RSA revient à :
  - Trouver $x$ tel que $x^e = a \; mod \; n$ ($e$, $a$ et $n$ sont connus).
  - Factoriser un nombre assez grand : Trouver deux nombres premiers $p$ et $q$ tels que $p * q = n$ ($n$ est connu).
- Les meilleurs algorithmes connus pour résoudre ces problèmes ne sont pas polynomiaux.
- Il se peut que quelqu'un connaisse un algorithme très efficace pour factoriser des nombres entiers et qu'il le garde en secret ! Pour cette personne ou groupe de personnes RSA n'est pas de tout sécuritaire.
- Il se peut bien aussi que les lois de la mathématique interdisent la résolution de ce problème dans un temps raisonnable.

# Sécurité de RSA

- Compétition de factorisation RSA (1991-2007) :

| Compétition | Prix | Statut | Date de factorisation | Par |
|---|---|---|---|---|
| RSA-576 | USD 10 000 | Factorisé | 3 décembre 2003 | J. Franke et al. |
| RSA-640 | USD 20 000 | Factorisé | 2 novembre 2005 | F. Bahr et al. |
| RSA-704 | USD 30 000 | Annulé | 2 Juillet 2012 | Shi Bai, Emmanuel Thomé et Paul Zimmermann |
| RSA-768 | USD 50 000 | Factorisé | 15 janvier 2010 | Divers organismes |
| RSA-896 | USD 75 000 | Annulé | - | - |
| RSA-1024 | USD 100 000 | Annulé | - | - |
| RSA-1536 | USD 150 000 | Annulé | - | - |
| RSA-2048 | USD 200 000 | Annulé | | - |

source : https ://fr.wikipedia.org/wiki/Compétition_de_factorisation_RSA

RSA-768 = 1 230 186 684 530 117 755 130 494 958 384 962 720 772 853 569 595 334 792 197 322 452 151 726 400 507 263 657 518 745 202 199 786 469 389 956 474 942 774 063 845 925 192 557 326 303 453 731 548 268 507 917 026 122 142 913 461 670 429 214 311 602 221 240 479 274 737 794 080 665 351 419 597 459 856 902 143 413

RSA-768 = 33 478 071 698 956 898 786 044 169 848 212 690 817 704 794 983 713 768 568 912 431 388 982 883 793 878 002 287 614 711 652 531 743 087 737 814 467 999 489 x **36 746 043 666 799 590 428 244 633 799 627 952 632 279 158 164 343 087 642 676 032 283 815 739 666 511 279 233 373 417 143 396 810 270 092 798 736 308 917**

## Sécurité de RSA

Signature $s = m^d \ mod \ n$ ; vérification $m = s^e \ mod \ n$

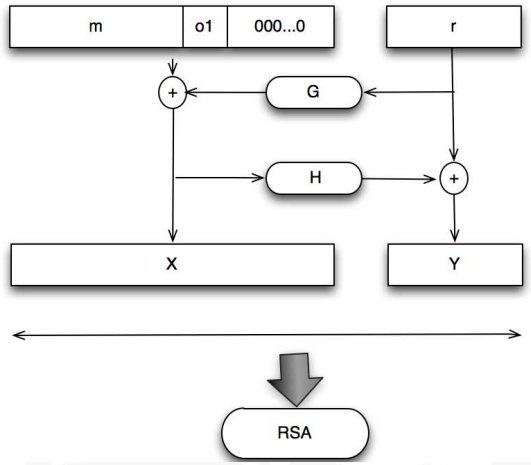
- Attaque sans messages : choisir $s$, calculer $m = s^e \ mod \ n$ (signature de $m$ est $s$)
- Attaque multiplicative : calculer $s_1 = m_1^d \ mod \ n$, $s_2 = m_2^d \ mod \ n$ alors $s_1 \times s_2$ est la signature de $m_1 \times m_2$

$$\begin{aligned} s_1 \times s_2 &= (m_1^d \ mod \ n) \times (m_2^d \ mod \ n) \\ &= (m_1 \times m_2)^d \ mod \ n \end{aligned}$$

- **Ne jamais chiffrer un message directement via RSA !**
- Déchiffrer certains messages : Si on a $c = m^e \ mod \ n$ et on sait que la taille de $m$ n'est pas grande (exemple une clé de 56 bits), alors un pirate peut calculer toutes les valeurs de $m_i^e \ mod \ n$ (avec $m_i$ parcourt tous les messages possibles de longueur 56 bits). La solution est d'allonger la taille de $m$ par un *padding* contenant une partie aléatoire.

# Sécurité de RSA

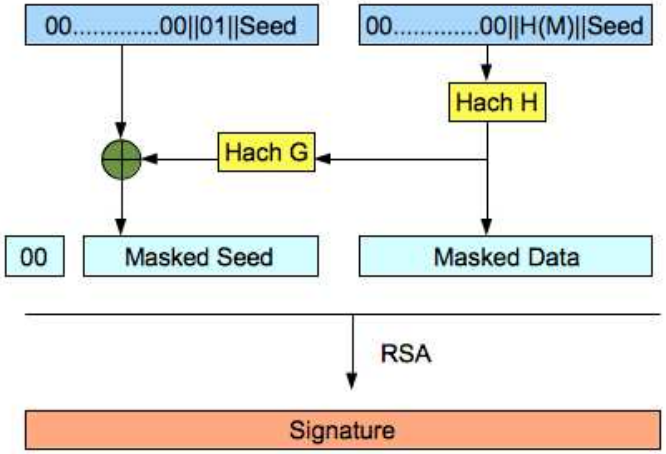
- Chiffrer un message : standard PKCS#1 v2.0 : AOEP (padding)




- En pratique $H$ et $G$ sont souvent remplacées par SHA256.
- Pour chiffrer, on calculer $c = RSA(OAEP(m))$ avec $OAEP(m) = X||Y$, $X = m \oplus G(r)$ et $Y = H(X) \oplus r$ ( $r$ aléatoire).
- Pour retrouver $m$, on déchiffre $c$, pour trouver $OAEP(m) = X||Y$, puis on calcule $r = H(X) \oplus Y$ et $m = X \oplus G(r)$
- Prouvé sécuritaire : Sans connaitre le $r$, si l'intrus nous envoie $m_1$ et $m_2$ et on lui retourne $c$ (chiffrement de l'un des deux), il ne peut pas dire si $c$ correspond à $m_1$ ou à $m_2$.

# Sécurité de RSA

- Signature en format standardisé PKCS#1 v2.1 : PSS




- En pratique $H$ et $G$ sont souvent remplacées par SHA256.
- On démontre que cette manière de faire est sécuritaire lorsque RSA, $H$ et $G$ respectent certaines conditions.

# Plan

- RSA
  - Introduction
  - Génération de clés
  - Cryptage/décryptage
  - Signature
  - Pourquoi ça marche ?
  - RSA en pratique
- Chiffrement d'El-Gamal sur $(\mathbb{Z}_P^*, \times)$
- Chiffrement d'El-Gamal sur les courbes elliptiques
- Cryptographie symétrique vs. asymétrique

# Chiffrement d'El-Gamal

⇨ Rappel :

↦ Définition d'un groupe : Soient $\mathcal{G}$ un ensemble d'éléments et $\circ$ un opérateur binaire. On dit que $(\mathcal{G}, \circ)$ forme un groupe si les quatre conditions suivantes sont respectées :

1. $a \in \mathcal{G}$ et $b \in \mathcal{G}$ alors $a \circ b \in \mathcal{G}$ (fermeture)
2. $a \in \mathcal{G}$, $b \in \mathcal{G}$ et $c \in \mathcal{G}$ alors $(a \circ b) \circ c = a \circ (b \circ c)$ (associativité)
3. Il existe un élément $e \in \mathcal{G}$ tel que pour tout $a \in \mathcal{G}$ on a :

$$a \circ e = e \circ a = a \text{ (élément neutre)}$$

4. Pour tout $a \in \mathcal{G}$, il existe $\overline{a} \in \mathcal{G}$ tel que : $a\overline{a} = e$ (inverse)

↦ Exemples :

– $(\mathbb{Z}, +)$ est un groupe.
– $(\mathbb{Z}, \times)$ n'est pas un groupe (problème d'inverse).

# Chiffrement d'El-Gamal

- Principe :
  - On suppose qu'on a un groupe $(\mathcal{G}, \circ)$ ou le problème de logarithme discret (DLP) est difficile :
    - Soient $n$ un entier et $g \in \mathcal{G} \Longrightarrow$ Calculer $n \otimes g = \underbrace{g \circ \ldots \circ g}_{n}$ est facile.
    - (DLP) : Étant données $n \otimes g$ et $g$ dans $\mathcal{G} \Longrightarrow$ Il est DIFFICILE de trouver $n$
    - (DHP) : Étant données $n_1 \otimes g$ et $n_2 \otimes g$ dans $\mathcal{G} \Longrightarrow$ Il est DIFFICILE de trouver $n_2 \otimes (n_1 \otimes g)$
    - (Conjoncture ) : DLP est equivalent à DHP.
  - Générer une paire de clés d'une entité $A$
    - Clé publique = $(n_a \otimes g, g)$ (fermeture : $n_a \otimes g$ est un élément du groupe)
    - Clé privée= $n_a$ ($n_a$ est un entier)
  - Encryption d'un message $m$ destiné à $A$
    - Calculer $B = n_b \otimes g$ et $K = n_b \otimes (n_a \otimes g)$ (fermeture : $K$ et $B$ sont des éléments du groupe)
      (Remarque : $K = n_a \otimes B$) (associativité de "$\circ$")
    - Envoyer $(B, K \circ m)$ (fermeture : $K \circ m$ est un élément du groupe)
  - Décryptyion par $A$
    - Calculer $\overline{K}$, l'inverse de $K$ ($K = n_a \otimes B$) dans $(\mathcal{G}, \circ)$ (Inverse : Tout élément est inversible)
    - Trouver $m = \overline{K} \circ K \circ m = e \circ m$ (inverse + élément neutre)
  - Intrus : Il doit résoudre le DHP pour trouver $m$ à partir de $g$, $n_a \otimes g$, $B$ et $K \circ m$

# Chiffrement d'El-Gamal sur $(\mathbb{Z}_p^*, \times)$

- Rappel sur $(\mathbb{Z}_p^*, \times)$ :
  - Définition : On dénote par $\mathbb{Z}_n^*$ l'ensemble des entiers dans $\{0, 1, \ldots, n-1\}$ et qui sont premiers avec $n$.
  - Exemple : $\mathbb{Z}_9^* = \{1, 2, 4, 5, 7, 8\}$

| $\times \ mod\ 9$ | 1 | 2 | 4 | 5 | 7 | 8 |
|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 4 | 5 | 7 | 8 |
| 2 | 2 | 4 | 8 | 1 | 5 | 7 |
| 4 | 4 | 8 | 7 | 2 | 1 | 5 |
| 5 | 5 | 1 | 2 | 7 | 8 | 4 |
| 7 | 7 | 5 | 1 | 8 | 4 | 2 |
| 8 | 8 | 7 | 5 | 4 | 2 | 1 |

  - Théorème : $(\mathbb{Z}_n^*, \times)$ est un groupe.
  - Lien avec le schéma précédent :
    - $(\mathcal{G}, \circ) = (\mathbb{Z}_n^*, \times)$
    - $n \otimes a = \underbrace{a \circ \ldots \circ a}_{n} = \underbrace{a \times \ldots \times a}_{n} = a^n \ mod\ p$
    - (DLP) : étant données $a$ et $b = a^x \ mod\ p \Longrightarrow$ Trouver $x$
    - (DHP) : étant données $(a^x \ mod\ p)$ et $(a^y \ mod\ p) \Longrightarrow$ Trouver $(a^x \ mod\ p)^y \ mod\ p$ (i.e : $a^{x \times y} \ mod\ p$)

# Chiffrement d'El-Gamal sur $(\mathbb{Z}_p^*, \times)$

➪ Rappel sur $(\mathbb{Z}_p^*, \times)$ :

- DLP et DHP sur $(\mathbb{Z}_p^*, \times)$ : $\mathbb{Z}_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

| $x$ | $2^x \, mod 11$ | $3^x \, mod 11$ | $4^x \, mod 11$ | $5^x \, mod 11$ | $6^x \, mod 11$ | $7^x \, mod 11$ | $8^x \, mod 11$ | $9^x \, mod 11$ | $10^x \, mod 11$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 2 | 4 | 9 | 5 | 3 | 3 | 5 | 9 | 4 | 1 |
| 3 | 8 | 5 | 9 | 4 | 7 | 2 | 6 | 3 | 10 |
| 4 | 5 | 4 | 3 | 9 | 9 | 3 | 4 | 5 | 1 |
| 5 | 10 | 1 | 1 | 1 | 10 | 10 | 10 | 1 | 10 |
| 6 | 9 | 3 | 4 | 5 | 5 | 4 | 3 | 9 | 1 |
| 7 | 7 | 9 | 5 | 3 | 8 | 6 | 2 | 4 | 10 |
| 8 | 3 | 5 | 9 | 4 | 4 | 9 | 5 | 3 | 1 |
| 9 | 6 | 4 | 3 | 9 | 2 | 8 | 7 | 5 | 10 |
| 10 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

- Remarques :
  - $b = a^x \ mod \ 11$ a une seule solution ssi $a$ est un générateur. Les solutions de $1 = 10^x \ mod \ 11$ sont $\{2, 4, 6, 8, 10\}$.
  - Le (DHP) est difficile à résoudre seulement si l'ordre de $a$ est grand. À partir de $10^x \ mod \ 11$ et $10^y \ mod \ 11$, on a une chance sur 2 de trouver $10^{x \times y} \ mod \ 11$.
- Définition (Ordre) : Soient $(\mathcal{G}, \circ)$ un groupe et $a$ un élément de $\mathcal{G}$. L'**ordre** de $a$ dans $(\mathcal{G}, \circ)$ est le plus petit entier $n$ tel que $\underbrace{a \circ \ldots \circ a}_{n} = e$.
- Définition (Générateur) : Un groupe $(\mathcal{G}, \circ)$ qui contient un élément $a$ tel que $ordre(a) = |\mathcal{G}|$ (ordre maximal) est dit **cyclique**. Les éléments de $\mathcal{G}$ qui ont un ordre maximal sont appelés **générateurs** ou éléments primitifs.

# Chiffrement d'El-Gamal sur $(\mathbb{Z}_p^*, \times)$

➪ Rappel :

- Exemple : Dans $(\mathbb{Z}_{11}^*, \times)$, $Ordre(3) = 5$ et $Ordre(2) = 10 = |\mathbb{Z}_{11}^*|$ (2 est un générateur).

| $3^x \ mod\ 11$ | $2^x \ mod\ 11$ |
|---|---|
| $3^1 = 3 \ mod\ 11$ | $2^1 = 2 \ mod\ 11$ |
| $3^2 = 9 \ mod\ 11$ | $2^2 = 4 \ mod\ 11$ |
| $3^3 = 27 \equiv 5 \ mod\ 11$ | $2^3 = 8 \ mod\ 11$ |
| $3^4 = 81 \equiv 4 \ mod\ 11$ | $2^4 = 16 \equiv 5 \ mod\ 11$ |
| $3^5 = 243 \equiv 1 \ mod\ 11$ | $2^5 = 32 \equiv 10 \ mod\ 11$ |
| | $2^6 = 64 \equiv 9 \ mod\ 11$ |
| | $2^7 = 128 \equiv 7 \ mod\ 11$ |
| | $2^8 = 256 \equiv 3 \ mod\ 11$ |
| | $2^9 = 512 \equiv 6 \ mod\ 11$ |
| | $2^{10} = 1024 \equiv 1 \ mod\ 11$ |

- Observation : Un générateur, génère tous les éléments du groupe : $2^i$, $i \in \{1, \ldots, 10\}$, générent tous les éléments de $\mathbb{Z}_{11}^*$.

| $i$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $2^i$ | 2 | 4 | 8 | 5 | 10 | 9 | 7 | 3 | 6 | 1 |

# Chiffrement d'El-Gamal sur $(\mathbb{Z}_p^*, \times)$

| Schéma Abstrait | Schéma Concret |
|---|---|
| ↦ Groupe $(\mathcal{G}, \circ)$<br>DLP : Trouver $n$ sachant que $n \otimes g$ et $g$ sont connues<br><br>DHP : Trouver $n_2 \otimes (n_1 \otimes g)$<br>sachant $n_1 \otimes g$ et $n_2 \otimes g$ sont connues | ↦ Groupe $(\mathbb{Z}_p, \times)$<br>DLP : Trouver $x$ sachant que $g^x \ mod\ p$ et $g$ sont connues<br>(g est un générateur $p$ est premier)<br>DHP : Trouver $g^{x \times y} \ mod\ p$<br>sachant $g^x \ mod\ p$ et $g^y \ mod\ p$ sont connues |
| ↦ Générer une paire de clés d'une entité $A$<br>Clé publique = $(n_a \otimes g, g, \mathcal{G})$<br>Clé privée= $n_a$<br>↦ Encryption d'un message $m$<br>Calculer $B = n_b \otimes g$ et $K = n_b \otimes (n_a \otimes g) = n_a \otimes B$<br>Envoyer $(B, K \circ m)$<br>↦ Décryptyion<br>Calculer $\overline{K}$, l'inverse de $K$ $(K = n_a \otimes B)$<br>Trouver $m = \overline{K} \circ K \circ m = e \circ m$ | ↦ Générer une paire de clés d'une entité $A$<br>Clé publique = $(g^{n_a} \ mod\ p, g, p)$<br>Clé privée= $n_a$ $\quad (n_a \in \{2, \ldots, p-1\})$<br>↦ Encryption d'un message $m$<br>Calculer $B = g^{n_b} \ mod\ p$ et $K = (g^{n_a} \ mod\ p)^{n_b} \ mod\ p = B^{n_a} \ mod\ p$<br>Envoyer $(B, K \times m \ mod\ p)$<br>↦ Décryptyion<br>Calculer $\overline{K}$, l'inverse de $K$ : $\overline{K} = B^{p-1-n_a}$<br>Trouver $m$ :<br>$m = \overline{K} \times K \times m \ mod\ p$<br>$= B^{p-1-n_a} \times B^{n_a} m \ mod\ p$<br>$= B^{p-1} \times m \ mod\ p = m$ (Th. Fermat) |

# Chiffrement d'El-Gamal sur $(\mathbb{Z}_p^*, \times)$

⇨ Comment trouver un générateur ?

- Quelques propriétés utiles des groupes cycliques : Soit $(\mathcal{G}, \times)$ un groupe cyclique.
  1. Pour tout $a \in \mathcal{G}$, $ord(a)$ divise $|\mathcal{G}|$ .
  2. pour un générateur $g$, $ord(g) = |\mathcal{G}|$
  3. Le nombre de générateurs de $(\mathcal{G}, \times)$ est $\Phi(|\mathcal{G}|)$.
  4. $Ord(a) = d$ alors les autres éléments de $\mathcal{G}$ ayant le même ordre sont $\{a^k \mid pgcd(k, d) = 1\}$
- Trouver un générateur : Conséquence de la propriété 1. : si $\mathcal{G} = \mathbb{Z}_p^*$ ($|\mathcal{G}| = |\mathbb{Z}_p^*| = p - 1$) et $p = 2 * q + 1$ avec $p$ et $q$ deux nombres premiers alors un élément $a$ dans $\mathbb{Z}_p^*$ est un générateur si $a^2 \not\equiv 1 \ mod \ p$ et $a^q \not\equiv 1 \ mod \ p$
- Exemple : $(\mathbb{Z}_{11}^*, \times)$
  - $p = 11 = 2 * 5 + 1$.
  - 2 est un générateur car $2^2 = 4 \not\equiv 1 \ mod \ 11$ et $2^5 = 32 \not\equiv 1 \ mod \ 11$
  - Il y a 4 générateurs dans $\mathbb{Z}_p^*$ car $\Phi(10) = (5 - 1) \times (2 - 1) = 4$
  - Les autres générateurs sont $\{2^k \ mod \ 11 \mid pgcd(k, 10) = 1\} = \{8, 7, 6\}$

# Chiffrement d'El-Gamal sur $(\mathbb{Z}_p^*, \times)$

➪ Exemple : Alice veut envoyer à Bob le message $m = 7$

| *Alice* | | *Bob* |
|---|---|---|
| | | (1) $p = 23,\ g = 7,\ a = 6$ |
| $m = 7$ | $(p = 23, g = 7, A = 4) \longleftarrow$ | (2) $A = g^a \ mod\ p = 4$ |
| $b = 3,\ B = g^b \ mod\ p = 21$<br><br>$c = A^b m \ mod\ p = 11$ | <br><br>$\longrightarrow (B = 21, c = 11)$ | $m = B^{p-1-6} c \ mod\ 23$<br>$= 21^{16} 11 \ mod\ 23$<br>$= 7$ |

# Signature El-Gamal

- clés
  - choisir un grand nombre premier $p$
  - choisir un générateur $g$ dans $Z_p^*$
  - choisir un entier $a$ entre 2 et $p-2$
  - clé publique =$(p, g, y = g^a \ mod \ p)$
  - clé privée= $a$
- Signer un message $m$
  - choisir aléatoirement $k$ tel que $k < p$ et $pgcd(k, p-1) = 1$
  - calculer $r = g^k \ mod \ p$ et $s = k^{-1}(H(m) - a \times r) \ mod \ (p-1)$
  - la signature de $m$ est $(r, s)$
- Vérifier la signature de $m$

$$y^r \times r^s \equiv^? g^{H(m)} \ mod \ p$$

# Signature El-Gamal

➪ Justification si $r = g^k \ mod \ p$ et $s = k^{-1}(H(m) - a \times r) \ mod \ (p-1)$,

on aura

$$\begin{aligned} y^r \times r^s \ mod \ p &= (g^a \ mod \ p)^r \times (g^k \ mod \ p)^{k^{-1}(H(m)-a\times r) \ mod \ (p-1)} \ mod \ p \\ &= g^{a\times r} \times g^{k\times k^{-1}(H(m)-a\times r) \ mod \ (p-1)} \ mod \ p \\ &= g^{a\times r} \times g^{(H(m)-a\times r) \ mod \ (p-1)} \ mod \ p \\ &= g^{a\times r} \times g^{H(m)-a\times r} \ mod \ p \\ &= g^{H(m)} \ mod \ p \end{aligned}$$

➪ $k$ est aléatoire

Si on choisit le même $k$ pour deux signatures, on sera en mesure de dévoiler le secret $a$

$s_1 = k^{-1}(H(m_1) - a \times r) \ mod \ (p-1)$ et $s_2 = k^{-1}(H(m_2) - a \times r) \ mod \ (p-1)$ donne $k = (s_1 - s_2)^{-1}(H(m_1) - H(m_2)) \ mod \ (p-1)$, on conclut que

$a = (r^{-1} \times (H(m_1) - s_1 \times k)) \ mod \ (p-1)$

# Signature DSA

Un standard de signature basé sur El-Gamal

- DSA (Digital Signature Algorithm)
- Proposé par le NIST comme DSS (Data Signature Standard)
- Adopté en 1994
- Utilisée par tous les agences et les département fédéraux de USA
- Beaucoup de critiques vis-à-vis de ce choix
  - Il ne permet l'encryption (seulement la signature)
  - Plusieurs attendaient aux choix de RSA (standard de facto)
  - Le processus de sélection n'était pas public (pas assez du temps pour l'analyser)
  - Le NIST a dit que c'est elle qu'elle a l'a conçu, puis elle a admis que le NSA l'a aidé, après elle a avoué que c'est l'¡uvre de NSA
  - NSA n'inspire pas confiance : crainte de porte arrière ("backdoor")
  - La taille originale du clé est petite (512 bits), mais NIST a rendu la taille variable 512, 1024

# Signature DSA

- clés
  - choisir un grand nombre premier $p$
  - choisir un grand nombre $q$ premier et diviseurs de $p-1$
  - choisir un générateur $g$ dans $Z_p^*$
  - choisir un entier $a$ entre 2 et $p-2$
  - clé publique =$(p, q, g, y = g^a \ mod \ p)$
  - clé privée= $a$
- Signer un message $m$
  - choisir aléatoirement $k$ tel que $k < q$
  - calculer $r = (g^k \ mod \ p) \ mod \ q$ et $s = k^{-1}(H(m) - a \times r) \ mod \ q$
  - la signature de $m$ est $(r, s)$
- Vérifier la signature de $m$

$w = s^{-1} \ mod \ q \qquad u = w \times H(m) \ mod \ q \qquad v = r \times w \ mod \ q$

$((g^u \times y^v) \ mod \ p) \ mod \ q =^? r$

# Signature DSA

- La signature est beaucoup plus rapide que El-Gamal et RSA

| | 512 bits | 768 bits | 1024 bits |
|---|---|---|---|
| Sign | 0.20 sec | 0.43 sec | 0.57 sec |
| Verify | 0.35 sec | 0.80 sec | 1.27 sec |

Résultats obtenus sur un ordinateur 80386 33 mHz

| | DSA | RSA |
|---|---|---|
| Signature | .03 sec | 15 sec |
| Verification | 16 sec | 1.5 sec |

- La vérification est par contre plus lente
- Les fonctions de hachage sont SHA, SHA-1

# Plan

- RSA
  - Introduction
  - Génération de clés
  - Cryptage/décryptage
  - Signature
  - Pourquoi ça marche ?
  - RSA en pratique
- Chiffrement d'El-Gamal sur $(\mathbb{Z}_P^*, \times)$
- Chiffrement d'El-Gamal sur les courbes elliptiques
- Cryptographie symétrique vs. asymétrique

# Chiffrement d'El-Gamal sur les courbes elliptiques

➪ Le groupe visé : $(\mathcal{G}, \circ) = (E(a, b, \mathbb{K}), +)$ : les éléments du groupes sont les points sur une courbe elliptique $E(a, b, \mathbb{K})$. L'opérateur du groupe, "+" est "l'addition" (à définir) de deux points de la courbe $E(a, b, \mathbb{K})$.

➪ Définition des éléments du groupe $E(a, b, \mathbb{K})$ :

➻ Qu'est ce qu'une courbe elliptique ? C'est un ensemble de points, notée par $E(a, b, \mathbb{K})$, défini de la manière suivante :

$$\{(x, y) \mid y^2 = x^3 + ax + b\} \cup \{O\}$$

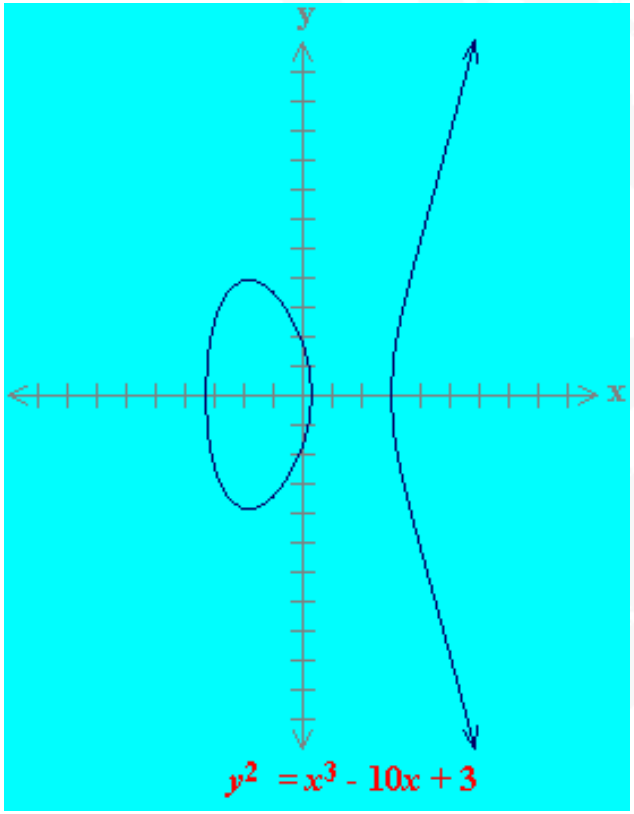
- Les éléments $a$, $b$, $x$ et $y$ sont dans un corps donné $\mathbb{K}$ (exemples : $(\mathbb{R}, +, \times)$, $(\mathbb{Q}, +, \times)$, $(\mathbb{Z}_p, +, \times)$ avec $p$ est premier, etc.)
- $O$ est un point "extra" qui va jouer le rôle de l'élément neutre du groupe visé.

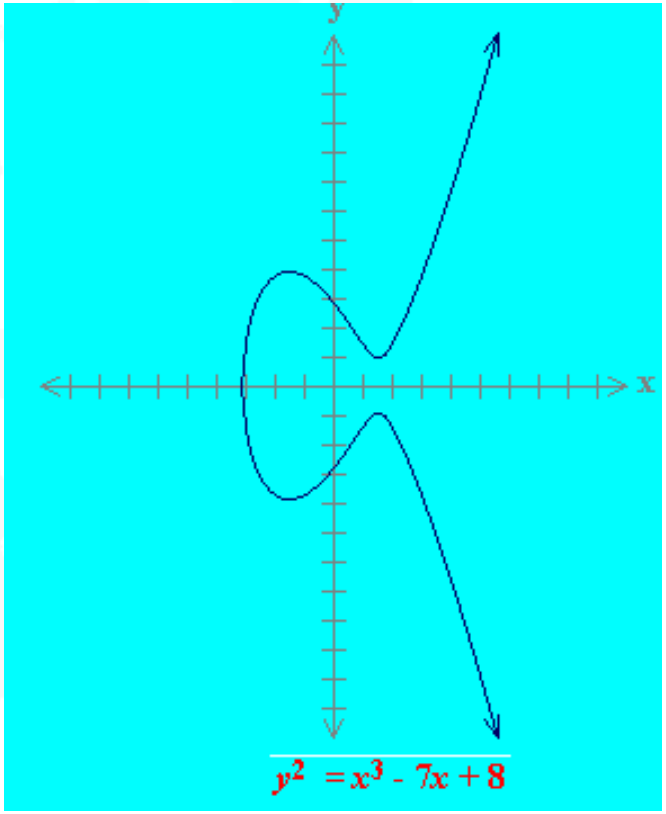
➻ Remarques :

- On s'intéresse aux courbes elliptiques n'ayant pas des racines multiples : $4a^3 + 27b^2 \neq 0$. Une condition nécessaire pour pouvoir construire le groupe visé.
- L'opérateur "$+$" du corps est différent de l'opérateur "+" du groupe visé. Par abus de langage, on confond les deux notations.
- Pour la cryptographie, on s'intéresse au courbes elliptiques sur des corps finis (corps de Galois).
- La forme générale de l'équation d'une courbe elliptique est $Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6$. Sous certaines conditions, on peut ramener cette équation, en faisant des changements de variables, à la forme simple : $y^2 = x^3 + ax + b$

# Chiffrement d'El-Gamal sur les courbes elliptiques

- Exemples (corps infinis) : Les figures suivantes donnent les courbes de $E(-10, 3, \mathbb{R})$ et $E(-7, 8, \mathbb{R})$.

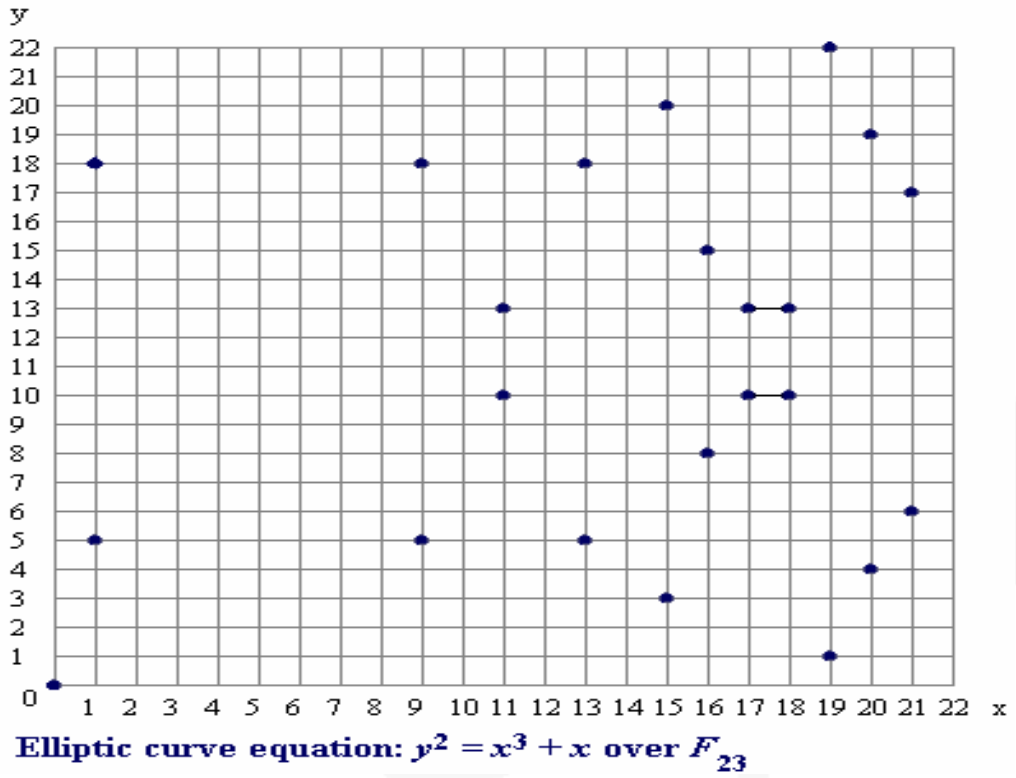



- Remarque : Toutes les images sur les courbes elliptiques proviennent du site : http ://www.certicom.com

# Chiffrement d'El-Gamal sur les courbes elliptiques

- Exemples (corps finis) : La figure suivante donne la courbe de $E(1, 1, \mathbb{Z}_{23})$.




- Remarques :
  - Il est difficile de retracer la forme de la courbe à partir des ses points.
  - Puisqu'on travaille dans $\mathbb{Z}_{23}$ tous les calculs se font modulo 23.

# Chiffrement d'El-Gamal sur courbes elliptiques

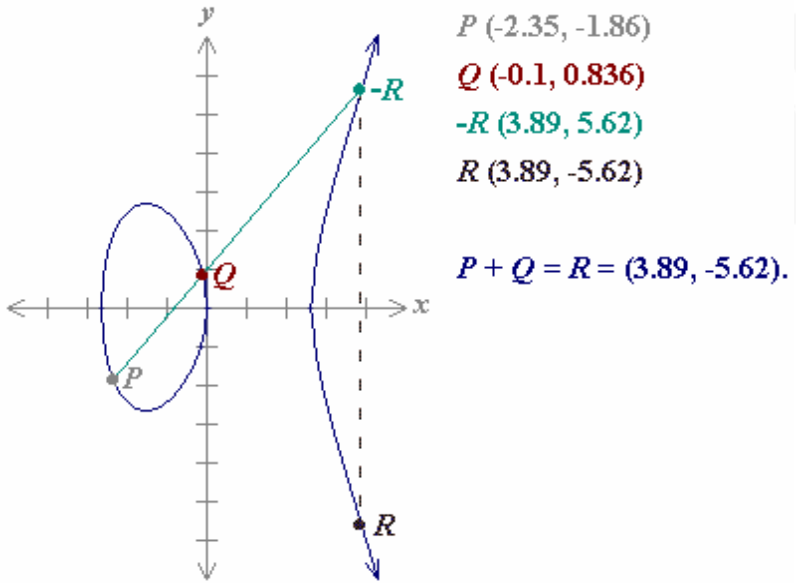
➪ Définition de l'opérateur + du groupe visé : Il s'agit de faire l'addition de deux points dans la courbe pour en obtenir un troisième.

➙ Comment additionner deux points ?

➙ Approche géométrique (corps infinis) :

– 1er cas : $P$ et $Q$ deux points distincts et non symétriques par rapport à l'axe des $x$ ($P \neq -Q$). L'addition $P$+$Q$ est le point $R$ sur la courbe tel que $-R$ (symétrie de $R$ par rapport à l'axe des $x$) est le point d'intersection entre la courbe et la droite qui passe par les deux points $P$ et $Q$ :

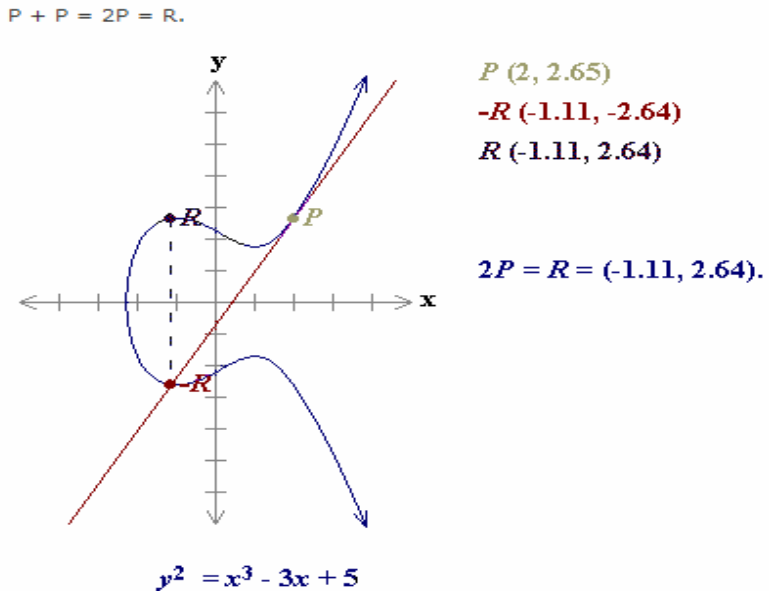
**Exemple :**



# Chiffrement d'El-Gamal sur les courbes elliptiques

– 2ème cas : $P$ et $Q$ deux points identiques. L'addition $P$+$P = 2P$ est le point $R$ sur la courbe tel que $-R$ est le deuxième point d'intersection (quand cela existe) de la courbe avec la tangente au point $P$.

**Exemple :**

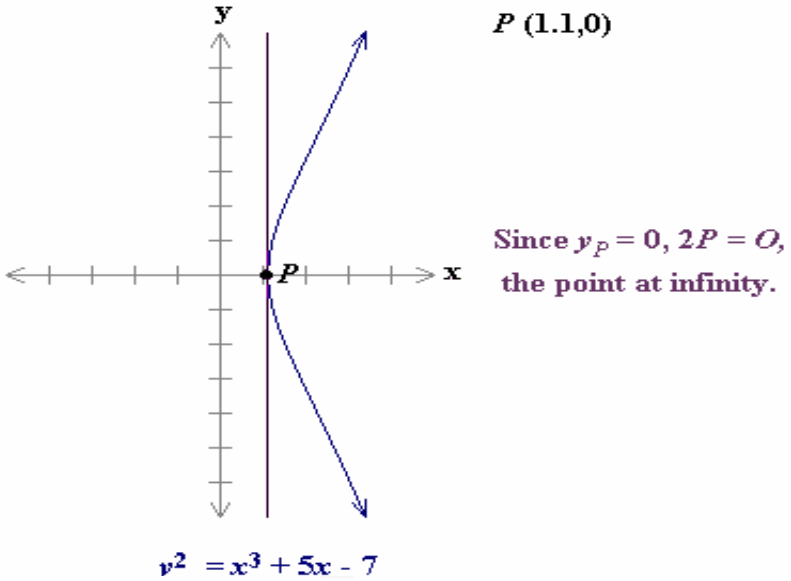



On note par $kP$ avec $k$ est un entier et $P$ un point par $\underbrace{P + \ldots + P}_{k}$.

# Chiffrement d'El-Gamal sur les courbes elliptiques

– 3ème cas : $P$ et $Q$ deux points identiques et l'intersection entre la courbe et la tangente en $P$ est un seul point : Dans ce cas $P$+$P = P = kP$.

**Exemple :**




– 4ème cas : $P$ et $Q$ sont symétriques par rapport à l'axe des $x$. Dans ce cas $P = -Q$ et $P+Q = O$.

↦ Remarque : L'approche géométrique n'est pas valide avec les corps finis.

# Chiffrement d'El-Gamal sur les courbes elliptiques

- Comment additionner deux points ? (suite)

→ Approche algébrique (valide pour les corps infinis et les corps finis) :

Soient $E(a, b, \mathbb{K})$ une courbe elliptique et $P_1 = (x_1, y_1)$ et $P_2 = (x_2, y_2)$ deux points de cette courbe avec $P_1 \neq -P_2$. Alors $P_1 + P_2 = P_3$ avec $P_3 = (x_3, y_3)$ est défini de la manière suivante :

$$\begin{cases} x_3 &= m^2 - x_1 - x_2 \\ y_3 &= m(x_1 - x_3) - y_1 \end{cases}$$

avec

$$m = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & si \quad P1 \neq P2 \\ \\ \frac{3x_1^2 + a}{2y_1} & si \quad P1 = P2 \end{cases}$$

Remarque : $m = \infty \Rightarrow P_1 + P_2 = (\infty, \infty) = O$

# Chiffrement d'El-Gamal sur les courbes elliptiques

**Addition (Cas particuliers) :**

$$\begin{aligned} O &= (\infty, \infty) \\ O+O &= O \\ P+O &= O + P = P \\ (x,y)+(x,-y) &= (x,y) - (x,y) = O \\ (x,-y) &= -(x,y) \end{aligned}$$

➪ Théorème : $(E(a,b,\mathbb{K}),+)$ est un groupe commutatif avec $O$ est l'élément neutre.

➪ Ordre d'un point :

Soit $P$ un point de $E(a,b)$. L'ordre de $P$ est le plus petit entier positif $i$ tel que

$$i * P = \underbrace{P + \ldots + P}_{i} = O$$

# Chiffrement d'El-Gamal sur les courbes elliptiques

➪ Résumé :

- ➙ Ce qui a été fait : Nous avons défini un groupe, $(E(a, b, \mathbb{K}), +)$, sur les courbes elliptiques.
- ➙ Ce qui reste à faire :
  - Voir si le DLP et DHP sont des problèmes difficiles sur ce groupe.
  - Faire la projection du schema abstrait sur ce groupe pour voir comment encrypter et décrypter.

➪ Le DLC sur les courbes elliptiques (ECDLP) : Étant donnés une courbe elliptique $E(a, b, \mathbb{K})$, deux points $P$ et $Q$ dans $E(a, b, \mathbb{K})$, et un entier $k$ tel que $P = k * Q = \underbrace{Q+ \ldots +Q}_{k}$. Trouver la valeur de $k$.

➪ Le DHP sur les courbes elliptiques (ECDHP) : Étant donnés une courbe elliptique $E(a, b, \mathbb{K})$, trois points $P$, $k_1 * P$ et $k_2 * P$, trouver le point $k_1 * (k_2 * P)$

➪ Exemple : Soit $E(9, 17, \mathbb{Z}_{23}) : Y^2 = X^3 + 9x + 17 \ (mod\ 23)$. Trouver la valeur de $k$ telle que $Q = k * P$ avec $Q = (4, 5)$ et $P = (16, 5)$

$$
\begin{array}{lll}
P = (16, 5) & 4P = (19, 20) & 7P = (8, 7) \\
2P = (20, 20) & 5P = (13, 10) & 8P = (12, 17) \\
3P = (14, 14) & 6P = (7, 3) & 9P = (4, 5)
\end{array}
$$

➪ ECDLP : C'est un problème très difficile (complexité exponentiel) quand l'ordre de $P$ est grand.

➪ Nombre de points dans $E(a, b, \mathbb{K})$ avec $\mathbb{K}$ est un corps fini qui contient $n$ éléments : Le nombre de points dans $E(a, b, \mathbb{K})$, noté par $\#E(a, b, \mathbb{K})$, est estimé par le théorème suivant (Théorème de Hasse : $|\#E(a, b, \mathbb{K}) - n - 1| < 2\sqrt{n}$

- ➙ Remarque : On démontre que pour tout $N$, $|N - n - 1| < 2\sqrt{n}$, ils existent a et b tels que $N = \#E(a, b, \mathbb{K})$

# Chiffrement d'El-Gamal sur les courbes elliptiques

| Schema Abstrait | Schema Concret |
|---|---|
| ↦ Groupe $(\mathcal{G}, \circ)$<br><br>DLP : Trouver $n$ sachant que $n \otimes g$ et $g$ sont connues<br><br>DHP : Trouver $n_2 \otimes (n_1 \otimes g)$<br>sachant $n_1 \otimes g$ et $n_2 \otimes g$ sont connues | ↦ Groupe $(E(a, b, \mathbb{Z}_p), +)$ $p$ premier ($\|p\| \geq 200$ bits)<br>(Remarque $\mathbb{Z}_p$ peut être remplacé par n'importe quel corps de Galois)<br>DLP : Trouver $k$ sachant que $k * P$ et $P$ sont connues<br>(P doit avoir un ordre assez grand)<br>DHP : Trouver $(k_1 \times k_2) * P$<br>sachant $P$, $k_1 * P$ et $k_2 * P$ sont connues |
| ↦ Générer une paire de clés d'une entité $A$<br>Clé publique = $(n_a \otimes g, g, \mathcal{G})$<br>Clé privée= $n_a$<br>↦ Encryption d'un message $m$<br>Calculer $B = n_b \otimes g$ et $K = n_b \otimes (n_a \otimes g) = n_a \otimes B$<br>Envoyer $(B, K \circ m)$<br>↦ Décryptyion<br>Calculer $\overline{K}$, l'inverse de $K$ ($K = n_a \otimes B$)<br>Trouver $m = \overline{K} \circ K \circ m = e \circ m$ | ↦ Générer une paire de clés d'une entité $A$<br>Clé publique = $(n_a * P; p, P, p)$<br>Clé privée= $n_a$ ( $n_a \in \{2, \ldots, p-1\}$)<br>↦ Encryption d'un message $m$<br>Calculer $B = n_b * P$ et $K = n_b * (n_a * P) = n_a * B$<br>Envoyer $(B, K+m)$<br>↦ Décryptyion<br>Calculer $\overline{K}$, l'inverse de $K$ : $\overline{K} = -K = -(n_a * B)$<br>Trouver $m$ :<br>$m = \overline{K}+K+m$<br>$= -n_a * B+n_a * B+m$ |

# Courbes elliptiques sur les corps finis

⇨ Comment choisir une courbe avec un point de base qui a un ordre assez grand ?

- Il est difficile de générer une "bonne" courbe elliptique avec un "bon" point de base $P$.
- Par bonne courbe elliptique, on veut dire une courbe qui résiste aux différentes attaques connues. Par exemple, il faut éviter les courbes $((E, a, b, \mathbb{Z}_p))$ avec $\#(E, a, b, \mathbb{Z}_p) = p$.
- Par un bon point de base, on veut dire un point qui a un ordre assez grand.
- L'ordre de n'importe quel point sur la courbe est un diviseur de $\#(E, a, b, \mathbb{K})$.
- Il existe des formules pour calculer le nombre de points dans certains types de courbes elliptiques. Par exemple si $p \equiv 2 \ mod \ 3$, $a = 0$ et $0 < b < p$ alors $\#(E, a, b, \mathbb{Z}_p) = p + 1$

# Courbes elliptiques sur les corps finis

- Remarque Plusieurs standards et recommendations sur le choix des courbes elliptiques ont été faits (voir http ://www.secg.org/collateral/sec2_final.pdf).
- Exemple :

$(E, a, b, \mathbb{Z}_p)$ : $y^2 = x^3 + ax + b \ mod \ p$

$$
\begin{aligned}
p &= FFFFFFFF\ FFFFFFFF\ FFFFFFFF\ FFFFFFFE\ FFFFAC73 \\
&= 2^{160} - 2^{32} - 2^{14} - 2^{12} - 2^9 - 2^8 - 2^7 - 2^3 - 2^2 - 1 \\
a &= 00000000\ 00000000\ 00000000\ 00000000\ 00000000 \\
b &= 00000000\ 00000000\ 00000000\ 00000000\ 00000007
\end{aligned}
$$

$$
\begin{aligned}
P_x &= 3B4C382C\ E37AA192\ A4019E76\ 3036F4F5\ DD4D7EBB \\
P_y &= 938CF935\ 318FDCED\ 6BC28286\ 531733C3\ F03C4FEE \\
Ord(P) &= 01\ 00000000\ 00000000\ 0001B8FA\ 16DFAB9A\ CA16B6B3
\end{aligned}
$$

$P = (P_x, P_y)$ est un bon point de base.

# Cryptographie asymétrique

➪ Quelques fameux problèmes :

- Problème de racine carrée modulo : Étant donnés $a$ et $n$, trouver $x$ tel que : $x^2 = a \ mod \ n$.
- Problème de logarithme discret : Étant donnés $g$, $a$ et $p$, trouver $x$ tel que $g^x = a \ mod \ p$.
- Problème de RSA : Étant donnés $e$, $a$ et $n$, trouver $x$ tel que $x^e = a \ mod \ n$.
- Problème de factorisation : Étant donné un entier positif $n \Rightarrow$ trouver des nombres premiers $p_1, \ldots, p_k$ et des entiers $\alpha_1, \ldots, \alpha_k$ tels que :

$$n = \prod_{i}^{k} p_i^{\alpha_i}$$

- Problème de Diffie-Hellman : Étant donnés $g^a \ mod \ n$ et $g^b \ mod \ n$, trouver $g^{ab} \ mod \ p$
- Problème de somme d'un sous-ensemble : Étant donnés un ensemble $S$ et un entier $m$, trouver $S' \subseteq S$ tel que :

$$\sum_{s \in S'} s = m$$

# Plan

- RSA
  - Introduction
  - Génération de clés
  - Cryptage/décryptage
  - Signature
  - Pourquoi ça marche ?
  - RSA en pratique
- Chiffrement d'El-Gamal sur $(\mathbb{Z}_P^*, \times)$
- Chiffrement d'El-Gamal sur les courbes elliptiques
- Cryptographie symétrique vs. asymétrique

# Cryptographie : Symétrique vs. Asymétrique

- Symétriques :
  - Avantages :
    - Rapidité (jusqu'à 1000 fois plus rapide que les solutions asymétriques)
    - Facilité d'implantation en "hardware"
    - Taille de la Clé : longueur de clé très réduite : 128 bits (= 16 caractères => mémorisable !) au lieu de 1024 bits pour des équivalents asymétriques
  - Inconvénients :
    - Le nombre de clé à gérer
    - Distribution des clés (authentification+confidentialité)
    - Certaines propriétés de sécurité importantes sont difficiles (ou impossible) à atteindre avec un système symétrique comme la signature

# Cryptographie : Symétrique vs. Asymétrique

- Asymétriques :
  - Avantages :
    - Le nombre de clés à distribuer est réduit relativement aux systèmes symétriques
    - La distribution de clé est simplifiée : On n'a besoin que de l'authentification
    - Permet de signer des messages facilement.
  - Inconvénients :
    - Taille des clés
    - La vitesse de chiffrement
- Problèmes propres aux deux techniques :
  - La gestion de clés par l'utilisateur reste le maillon le plus faible
  - Sécurité (normalement) basée sur des arguments empiriques plutôt que théoriques

# Cryptographie : Symétrique vs. Asymétrique

**NIST guidelines for public key sizes for AES**

| ECC KEY SIZE (Bits) | RSA KEY SIZE (Bits) | KEY SIZE RATIO | AES KEY SIZE (Bits) |
|---|---|---|---|
| 163 | 1024 | 1 : 6 | |
| 256 | 3072 | 1 : 12 | 128 |
| 384 | 7680 | 1 : 20 | 192 |
| 512 | 15 360 | 1 : 30 | 256 |

*Supplied by NIST to ANSI X9F1*

**Source:** http://www.certicom.com/index.php?action=sol,why_ecc_soft

**Remarques RSA vs ECC**

RSA se base sur la factorisation, un problème plus étudié que le logarithme discret sur lequel se base ECC.

Les meilleurs algorithmes pour résoudre la factorisation sont sous-exponentiels (avant les année 90, $2^{\sqrt{n}}$ avec $n$ est la taille de la clé, après les années 90, $2^{\sqrt[3]{n}}$, une énorme chute) alors que celui de logarithme discret est exponentiel ($2^n$ : comme une force brute)

ECC est devenu plus intéressant que RSA, surtout quand les ressources (mémoire + cpu) sont limitées