# TP 3 : Chiffrement asymétrique El Gamal

## Chiffrement d'El Gamal

Le chiffrement d'El Gamal est un protocole de cryptographie asymétrique inventé par Taher El Gamal en 1984 et qui repose sur le problème du logarithme discret.

### Génération des clefs.

La première étape du schéma de chiffrement consiste à produire une paire de clefs : la clef publique, et la clef secrète (ou clé privée). La première servira à chiffrer les messages et la deuxième à les déchiffrer.
Pour générer sa paire de clefs, il faut d'abord choisir un nombre **premier** $p$ et un générateur $g$ pour $\mathbb{Z}/p\mathbb{Z}$.
Ensuite il faut choisir un élément s de $\mathbb{Z}/p\mathbb{Z}$ qui constituera la clef privée, et calculer $h = g^s \mod p$.
Pour terminer, la clé publique est $(p, g, h)$.

### Chiffrement

Pour chiffrer un message $M$ encodé comme un entier de $\mathbb{Z}/p\mathbb{Z}$ avec la clé publique $(p, g, h)$, il faut tirer au hasard un aléa $r$ dans $\mathbb{Z}/p\mathbb{Z}$ et calculer $C_1 = g^r \mod p$ et $C_2 = (M \cdot (h^r \mod p)) \mod p$.
Ainsi le chiffré $C$ est composé de ces deux nombres : $C = (C_1, C_2)$.

### Déchiffrement

Ayant accès à $C = (C_1, C_2)$ et à la clé privée $s$, pour déchiffrer il suffit de calculer :

$$\Big(C_2 \times ((C_1)^s \mod p)^{-1} \mod p\Big) \mod p$$

pour retrouver le message $M$.

**Exercice 1** (Génération des clefs). Écrivez une fonction Python3 `Clefs(k)` qui permet de générer une paire de clefs El Gamal, avec un module compris entre $2^k$ et $2^{k+1} - 1$.
La taille compte ! Plus vos clés seront de grands nombres, plus elles seront sûres. Le « paramètre de sécurité » est donc le nombre de bits du module $p$, autrement dit son logarithme binaire $k$.
Vous aurez peut-être l'usage de la fonction `randprime` de la bibliothèque `sympy` (éventuellement à installer en tapant `pip install sympy` dans un terminal) et de la fonction `randint` de la bibliothèque `random`. Pour calculer $x^n \mod p$ vous devrez utiliser `pow(x, n, mod=p)` car `(x**n)%p` ne marche pas pour les grands nombres. Attention, ce n'est **pas** une fonction de la bibliothèque `math` ! !

1. Choisissez comme module un nombre premier $p$ au hasard compris entre $2^k$ et $2^{k+1} - 1$.
2. Choisissez comme générateur un nombre $g$ au hasard compris entre 2 et $p - 1$.

   **Remarque :** Votre « générateur » n'en est peut-être pas un. Pour gagner du temps, on ne vous demande pas de le vérifier : la procédure de chiffrement/déchiffrement marche encore, mais votre clé sera plus facile à casser, car il y aura plusieurs valeurs possibles pour $s$, en plus de celle qui sera vraiment votre clé secrète.
3. Choisissez comme secret un nombre $s$ au hasard compris entre 2 et $p - 1$.
4. Calculez $h$.

**Exercice 2.** Écrivez une fonction Python3 `Chiffrer(...)` qui permet de chiffrer un nombre $M$ avec le chiffrement d'El Gamal.

**Exercice 3.** Écrivez une fonction Python3 `Dechiffrer(...)` qui permet de déchiffrer un chiffré $(C_1, C_2)$ avec le chiffrement d'El Gamal.
Vous pouvez calculer l'inverse de $x$ modulo $p$ (s'il existe) par `pow(x, -1, p)`.

**Exercice 4.** Fondamentalement, le chiffrement d'El Gamal s'applique à des nombres. Mais très souvent, les messages qu'on souhaite chiffrer sont plutôt des *textes* ! Pour chiffrer des textes, il faut donc passer par une étape préalable de *numérisation* des caractères qui le composent.

1. Écrivez une fonction Python3 `Encode(texte)` qui convertit une chaîne de caractères `texte` en un nombre entier. Il y a plusieurs manières de faire cela, voici celle que nous proposons ici :
   (a) Chaque caractère est encodé par son code ASCII décimal, composé de 1, 2 ou 3 chiffres. Étant donné un caractère `c`, le code ASCII correspondant est obtenu par `ord(c)`.
   (b) Les codes ASCII des caractères sont transformés en `str`, puis ramenés sur 3 chiffres en ajoutant si nécessaire des 0 à gauche.
   (c) Les chaînes représentant les codes ASCII sur 3 chiffres sont concaténées les unes à la suite des autres pour obtenir une seule (grande) chaîne, ensuite transformée en `int`.
2. Écrivez une fonction Python3 `Decode(nombre)` effectuant l'opération réciproque : étant donné un nombre entier `nombre`, retrouver la chaîne de caractères `texte` qu'il représente.
   Pour cela, utilisez la fonction `chr`. Étant un code ASCII donné `num`, le caractère associé est obtenu par `chr(num)`.
   Pour récupérer les codes ASCII des différents caractères, vous voudrez peut-être utiliser le reste (%) et le quotient (//) de la division par 1000.
3. Testez vos fonctions d'encodage/décodage. On doit avoir, pour tout message $M$ :

   `M=Decode(Encode(M))`

**Exercice 5.** À ce stade, vous pouvez chiffrer/déchiffrer des messages entre vous.

- Un voisin vous communique sa clé publique. Vous pouvez donc maintenant lui envoyer des messages chiffrés.
- Choisissez un message `texte` ((très) court ! puisque le nombre encodant votre message ne doit pas dépasser son module). Convertissez-le en `nombre`. Chiffrez ce nombre. Envoyez le cryptogramme résultant à votre voisin.

- Votre voisin déchiffre le cryptogramme. Puis il décode le `nombre` obtenu, et retrouve votre `texte`.

**Exercice 6** (Bonus)**.** Pour pouvoir échanger des messages aussi longs que vous voulez, écrivez une fonction qui découpe les entiers trop longs pour en faire plusieurs messages.

**Exercice 7** (Exponentielle)**.**

1. Écrire une fonction `exp_basique(x,n,p)` qui, pour trois entiers positifs $x, n, p$ donnés en entrée, calcule de manière basique (sans utiliser la bibliothèque `math`) l'exponentielle modulaire

$$x^n \mod p$$

2. L'algorithme d'*exponentielle rapide* permet également de calculer le nombre

$$x^n \mod p$$

mais en utilisant l'algorithme récursif suivant :

$$\text{puissance}(x,\, n) = \begin{cases} x, & \text{si } n = 1 \\ \text{puissance}(x^2,\, n/2), & \text{si } n \text{ est pair} \\ x \times \text{puissance}(x^2,\, (n-1)/2), & \text{si n est impair} \end{cases}$$

Écrire une fonction `exp_rapide(x,n,p)` implémentant cet algorithme.

3. Calculer le nombre suivant, avec les deux méthodes :

$$123456789^{123456789} \pmod{987654321} = 598987215$$

Que constatez-vous ?