Fiche Exercices pratiques 13 – RSA jouet

Laurent Signac – CC-BY-SA – 08-06-21 1407 212bc868505c2f09d1db

Réalisez un ensemble de fonctions capables de chiffrer/déchiffrer un nombre avec l'algorithme RSA.

Les nombres que vous mettrez en jeu seront de taille raisonnable (quelques milliers maximum) : impossible de les utiliser en pratique avec RSA, mais suffisants pour comprendre le principe.

Vous pourrez vous inspirer du plan suivant pour réaliser vos fonctions :

1. fonction qui teste la primalité dun nombre (méthode naïve)
2. fonction qui cherche le plus petit nombre premier strictement supérieur à un nombre donné (utilise 1.)
3. fonction qui à partir de deux nombres donnés recherche deux nombres premiers (p et q) (utilise 2.)
4. fonction de calcul du pgcd (ou utilisation de celle de Python)
5. À partir de p et q et e_0, renvoie $e \geq e_0$ premier avec $z = (p-1)(q-1)$ (utilise 4.)
6. Calcul de l'inverse d'un nombre dans $z\mathbb{Z}$ (utilisation de l'algorithme d'Euclide étendu) : étant donnés z et e premier avec z, calculer d tel que $ed \equiv 1 \pmod{z}$. Une recherche exhaustive est envisageable mais ça ne fonctionnera que pour de petits nombres.
7. Empaqueter le tout dans une fonction à qui on donne trois nombres quelconques, recherche p et q premiers supérieurs aux deux premiers nombres, et renvoie les 3 couples : (e, n), (d, n), (p, q), avec e supérieur au 3e nombre.
8. Écrire une fonction qui prend en paramètre a, b, k et renvoie $a^b \pmod{k}$ (ou recherche si elle existe déjà dans Python).

Choisissez maintenant un nombre M plus petit que n, chiffrez le, puis déchiffrez le résultat pour vérifier que tout marche. Avec des nombres $$ et q à quatre chiffres, vous pouvez faire la vérification pour **tous** les M possibles.