Université Paris 8 Année 2021–2022

M1 Mathématiques et applications, parcours ACC et CSSD

Cryptographie à clef publique – Devoir semaine 3

devoir du 11/02/2022
à rendre jusqu'au 18/02/2022

Le choix du langage de programmation est laissé libre, bien qu'il soit conseillé d'utiliser `python` pour sa facilité d'utilisation et sa manipulation aisée des grands entiers.

Exercice 1. 💻 Implantation du chiffrement de Rabin.

Question 1.– Implanter une fonction `xgcd(p, q)` qui retourne les coefficients de Bezout associés à deux entiers `p` et `q` (autrement dit, des entiers u et v tels que $up + vq = \text{pgcd}(p, q)$).

On rappelle que pour $p \equiv 3 \mod 4$, les racines carrées d'un carré $x \in \mathbb{Z}/p\mathbb{Z}$ sont $\pm x^{(p+1)/4}$. On rappelle également que l'on peut déduire les racines carrées modulo $n = pq$ de celles modulo p et q au moyen de l'isomorphisme des restes chinois.

Question 2.– Implanter une fonction `square_roots(x, p, q, n, u, v)` qui retourne la liste des racines carrées de x modulo n, où $n = pq$, avec p et q deux nombres premiers distincts congrus à 3 modulo 4, et où u et v sont les coefficients de Bezout associés à p et q. On supposera que x est bien un carré modulo n.

On rappelle dans les Algorithmes 1, 2 et 3 le fonctionnement du cryptosystème de Rabin, dans un mode que l'on qualifie d'**initial**.

Algorithme 1 : Génération de clés dans le cryptosystème de Rabin

Entrée : un paramètre de sécurité
Sortie : une paire de clés publique/privée
1 Choisir aléatoirement deux grands nombres premiers distincts p et q tels que $p \equiv q \equiv 3 \mod 4$. La taille de ces entiers est en accord avec le paramètre de sécurité.
2 Calculer $n = pq$, ainsi que les coefficients de Bezout u et v de p et q.
3 La **clé publique** est n, la **clé privée** est (p, q).

Algorithme 2 : Chiffrement dans le cryptosystème de Rabin

Entrée : la clé publique, un message $m \in \mathbb{Z}/n\mathbb{Z}$
Sortie : un chiffré $c \in \mathbb{Z}/n\mathbb{Z}$
1 Retourner $c = m^2 \mod n$.

Question 3.– Implanter les fonctions de chiffrement `encrypt_rabin_1(m, pk)` et de déchiffrement `decrypt_rabin_1(c, sk)` du cryptosystème initial de Rabin. On stockera dans les variables `pk` et `sk` les clés publiques et privées, et dans les variables `m` et `c` le clair et le chiffré.

Algorithme 3 : Déchiffrement dans le cryptosystème de Rabin

Entrée : la clé privée, un chiffré $c \in \mathbb{Z}/n\mathbb{Z}$
Sortie : une liste d'éléments de $\mathbb{Z}/n\mathbb{Z}$
1 Retourner la liste des racines carrées de c modulo n.

Question 4.– Écrire une procédure de test `TEST1()` qui démontre que votre implantation est valide. On vérifiera donc simplement que le message d'origine m figure dans la liste retournée par `decrypt_rabin_1(c, sk)`. Comme paramètres, on pourra prendre $p = 19$, $q = 23$ et tirer des messages aléatoires.

Pour pouvoir reconnaître quel est le bon message parmi la liste de racines carrées retournées par l'algorithme de déchiffrement, une idée est d'utiliser du **padding**, c'est-à-dire d'ajouter une série de zéros au message avant de chiffrer. Notons qu'ajouter des zéros (en binaire) à un entier correspond à le multiplier par une puissance de 2.

On décrit dans les Algorithmes 4 et 5 les modifications effectuées dans le cryptosystème de Rabin.

Algorithme 4 : Chiffrement dans le cryptosystème de Rabin avec padding

Entrée : la clé publique, un message m, un paramètre de padding ℓ
Sortie : un chiffré $c \in \mathbb{Z}/n\mathbb{Z}$
1 Calculer $m' = m \times 2^\ell$.
2 Retourner $c = (m')^2 \mod n$.

Algorithme 5 : Déchiffrement dans le cryptosystème de Rabin avec padding

Entrée : la clé privée, un chiffré $c \in \mathbb{Z}/n\mathbb{Z}$, un paramètre de padding ℓ
Sortie : un message m ou une erreur
1 Calculer la liste L des racines carrées de c modulo n.
2 Chercher dans L l'unique élément m' divisible par 2^ℓ.
3 S'il y en plusieurs, retourner une erreur.
4 Sinon, retourner $m = (m')/2^\ell$.

Question 5.– Implanter les fonctions de chiffrement `encrypt_rabin_2(m, pk, padding)` et de déchiffrement `decrypt_rabin_2(c, sk, padding)` du cryptosystème de Rabin **avec padding**.

Question 6.– Écrire une procédure de test `TEST2()` qui démontre que votre implantation est valide. On pourra prendre $p = 10007$, $q = 22247$ et un padding de 10, puis choisir le message m à chiffrer entre 0 et 217400 (pour que le padding de m ne dépasse pas n).

Bob décide de chiffrer un texte **lettre par lettre** et de l'envoyer à Alice. Pour cela, il encode chaque lettre de son texte par le caractère ASCII correspondant, puis il chiffre les entiers obtenus avec la version **avec padding** du cryptosystème de Rabin. On suppose qu'Alice choisit un padding de $\ell = 1500$ bits.

Sur la page web :

`www.math.univ-paris13.fr/~lavauzelle/teaching/2021-22/docs/CP/devoirs/devoir3.html`

vous trouverez

— la clé publique d'Alice (`pk.txt`) de taille approximativement 2048 bits,
— le message chiffré de Bob (`ciphertext.txt`), constitué de 5 lignes correspondant aux chiffrés de 5 lettres de l'alphabet, qui forment un mot.

Question 7.– Sans connaissance de la clé privée, attaquez le chiffré de Bob et retrouvez son message. *Indication : ne cherchez pas à factoriser n qui est de trop grand taille.*

Pour vous aider, vous trouverez la liste des caractères ASCII ici :

`https://www.rapidtables.com/code/text/ascii-table.html`