

DM 13

à rendre le lundi 29 janvier 2024

Méthode $p - 1$ de factorisation de Pollard

Partie A - Arithmétique

1. Montrer le **petit théorème de d'Euler** : Soit p, q deux nombres premiers distincts et $n = pq$.

a) Montrer que si a est premier avec p et q , alors $a^{(p-1)(q-1)} \equiv 1 [n]$.

b) Généralisation : Montrer que pour tout $a \in \mathbb{Z}$ et $k \in \mathbb{N}$, alors $a^{k(p-1)(q-1)+1} \equiv a [n]$.

c) Soit $e, d \in \mathbb{Z}$ tel que $ed \equiv 1 [(p-1)(q-1)]$. Montrer que pour tout $x \in \llbracket 0, n-1 \rrbracket$,

$$x \equiv x^{de} [n]$$

2. Combien de chiffre contient l'écriture binaire de m^k où m et k sont des nombres s'écrivant sur 1024 bits ? Sachant que l'univers observable contient de l'ordre de 10^{80} atomes, que pensez vous du calcul de m^k ?

3. Considérer l'algorithme suivant :

Prog

```

1 def algo(x, n):
2     r, m, p = 1, x, n
3     while p != 0:
4         if p % 2 == 1:
5             r = r * m
6             m, p = m * m, p // 2
7     return r

```

a) Suivre dans un tableau le contenu des variables ou expression lors de l'exécution de `algo(-2, 13)` :

r	m	p	p%2 (test lig.4)
...
⋮	⋮	⋮	⋮

Une ligne par boucle ; les valeurs étant données au début de chaque boucle.

b) Conjecturer la valeur retournée par `algo` en fonction de x et n .

c) Proposer une version récursive de cette fonction, `algo_R(x, n)`, et former le tableau suivant le contenu des variables lors de l'exécution de `algo_R(-2, 13)`.

4. En adaptant la fonction précédente, proposer une fonction `ERM(x, d, n)`, qui calcule $y \equiv x^d [n]$ en ne manipulant que des entiers inférieurs à n^2 : c'est l'algorithme de *l'exponentiation rapide modulaire*.

Définition – Entier b -friable

Soit $b \in \mathbb{N}$. On dit qu'un entier $k \in \mathbb{N}$ est b -friable si ses facteurs premiers sont inférieurs à b .

5. Soit k un entier b -friable et α le maximum de ses valuations p -adique. Montrer que k divise $(\alpha \times b)!$

6. Soit $n \in \mathbb{N}$ et $p \in \mathbb{P}$ un facteur premier de n et $B \in \mathbb{N}$ tel que $B!$ soit un multiple de $p - 1$. On pose a le reste de la division euclidienne de $3^{B!}$ par n .

- Montre que $a \equiv 3^{B!} [p]$.
- Justifier que $3^{p-1} \equiv 1 [p]$ et en déduire que $a \equiv 1 [p]$.
- En déduire que p est un diviseur de $a - 1$.
- En déduire que p divise $(a - 1) \wedge n$.

Partie B - Chiffrement RSA

R. Rivest (1947-)



A. Shamir (1952-)



L. Adelman (1945-)

⇒ Phase 1 - **Pour recevoir des messages**, Bob crée sa clé secrète et sa clé publique.

- Il choisit deux grands nombres premiers p et q dont il calcule le produit $n : n = pq$
- Il choisit un nombre d premier avec $(p - 1)(q - 1)$: c'est la *clé de chiffrement*.
- Il calcule e , la *clé de déchiffrement* telle que : $de \equiv 1 [(p - 1)(q - 1)]$
- Bob garde **secrets** p, q et e et **publie** ses "*coordonnées*" n et d .

⇒ Phase 2 - Alice **veut envoyer un message chiffré** à Bob.

- Elle regarde dans l'annuaire (public) et trouve les coordonnées n et d de Bob.
- Elle convertit son message en nombres plus petits que n (voir TP numérisation d'un fichier texte)
- Elle chiffre chaque nombre, x , en calculant $y \equiv x^d [n]$.
- Elle envoie ensuite les nombres y à Bob.

⇒ Phase 3 - Bob **déchiffre le message** d'Alice.

- Il déchiffre chaque nombre, y , en calculant $z \equiv y^e [n]$ (le résultat du 1c) donne que $z = x$).
- Il reconstitue le message à partir des nombres z .

On suppose posséder les fonction suivante :

- `numerise(f1, f2, n)` : elle considère un fichier de caractères, $f1$, le converti en bloc de nombres de $\llbracket 0, n - 1 \rrbracket$, et enregistre un nombre par ligne dans le fichier $f2$;
- `denumerise(f2, f3)` qui inverse le processus de la fonction `numerise` : qui considère un fichier contenant un entier par ligne, $f2$, le converti en texte qu'il enregistre dans le fichier $f3$.

7. Écrire une fonction `chiffre_RSA(f1, f3, d, n)` qui :

- considère le fichier texte $f1$, le numérise en entier de $\llbracket 0, n - 1 \rrbracket$ et enregistre le tout dans un fichier $f2$, un entier par ligne ;
- chiffre chaque entier du fichier $f2$, par la méthode RSA, avec la clé de chiffrement (d, n) ;

- enregistre, les entiers obtenus après chiffrement dans le fichier `f3`, un entier par ligne.
8. Écrire une fonction `dechiffrer_RSA(f1, f3, e, n)` qui :
- considère le fichier `f1`, contenant sur chaque ligne un entier de $\llbracket 0, n - 1 \rrbracket$;
 - déchiffre chaque entier, par le méthode RSA, avec la clé de déchiffrement (e, n) ;
 - enregistre, les entiers obtenus après déchiffrement dans le fichier `f2`, un entier par ligne ;
 - retranscrit en texte le fichier `f2` dans un fichier texte `f3`.
9. Traitons un cas simple afin de vérifier les fonctions précédentes : $n = 15770708441$, $d = 5$ le message à déchiffrer est contenu dans `secret1.txt`.
- a) Factoriser n à l'aide d'un petit programme PYTHON.
- b) Utilisant la fonction `Bezout` donnée, déterminer e , l'inverse de d modulo $(p - 1)(q - 1)$.
- c) Retrouver le message originel.

Partie C - Cryptanalyse par la méthode $p - 1$ de Pollard

La méthode $p - 1$ de Pollard est une méthode de factorisation qui consiste à identifier les facteurs premiers p tels que $p - 1$ soit friable.

En particulier, si p est un facteur premier de n tel que $p - 1$ est friable alors il existe un *petit* nombre B tel que $p - 1$ divise B et p divise $a - 1 \wedge n$ avec $a \equiv 3^{B!} [n]$.

Un algorithme est :

Entrées : $n \in \mathbb{N}$ et $B \in \mathbb{N}$ (une borne)
$k \leftarrow 1$
$a \leftarrow 3$
$d \leftarrow 1$
Tant que $d = 1$ et $k \leq B$ faire
$k \leftarrow k + 1$
$a \leftarrow a^k [n]$
$d \leftarrow (a - 1) \wedge n$
FinTantque
Sortie : d

Remarque : Si $d = 1$, alors la démarche n'est pas productive, on peut réessayer en remplaçant l'entier 3 par n'importe quel entier $g \in \llbracket 2, n - 1 \rrbracket$ ou augmenter la borne B .

Remarque : Si $a \neq 1$, alors on a trouvé un facteur de n , on peut poursuivre la factorisation de n en travaillant sur le quotient.

10. Donner une fonction `Pollard_p_1(n, B=10000)` qui recherche un facteur premier p de n dans le cas où $p - 1$ est friable avec B une borne, par défaut 10000, telle que $B!$ soit un multiple de $p - 1$.
- Exemple* `Pollard_p_1(15770708441, 180) → 135979`
11. Supposons avoir factorisé n ; on suppose donc connaître p, q et d . Que faut-il faire pour retrouver la clé de déchiffrement e ?
12. Appliquer la démarche de la question précédente pour trouver la clé de déchiffrement, e , associée à la clé de chiffrement ($d = 5, n = 2021$)
13. Utiliser la méthode $p - 1$ de Pollard pour décrypter le fichier `secret.txt` dont la clé de chiffrement est donnée dans le fichier `Pollard_inc.py`.

Attention ! L'utilisation de la cryptographie à usage privé ou professionnel est encadrée en France par l'Agence Nationale de la Sécurité des Systèmes d'Information : <https://www.ssi.gouv.fr>