

Cryptographie à clé publique – Feuille de TD 1

22/01/2024

Le corrigé de certains exercices sera disponible à l'adresse suivante :

<https://lvz1.fr/teaching/2023-24/cp.html>

(★) exercice fondamental (★★) pour s'entraîner (★★★) pour aller plus loin  sur machine

Exercice 1. (★) Application du protocole de Diffie–Hellman.

On exécute le protocole de Diffie–Hellman dans le groupe multiplicatif $G = \mathbb{F}_p^\times$ avec les paramètres suivants :

- $p = 19$,
- le générateur du groupe est $g = 2$,
- la valeur secrète d'Alice est $a = 4$,
- la valeur secrète de Bob est $b = 6$.

Question 1.– Quelle est la valeur commune obtenue par Alice et Bob ?

On souhaite maintenant exécuter le protocole de Diffie–Hellman dans le sous-groupe des résidus quadratiques de \mathbb{F}_p^\times , que l'on note QR_p^\times . Pour cela on garde la valeur $p = 19$.

Question 2.– L'élément $g = 2$ reste-t-il un générateur de QR_p^\times ?

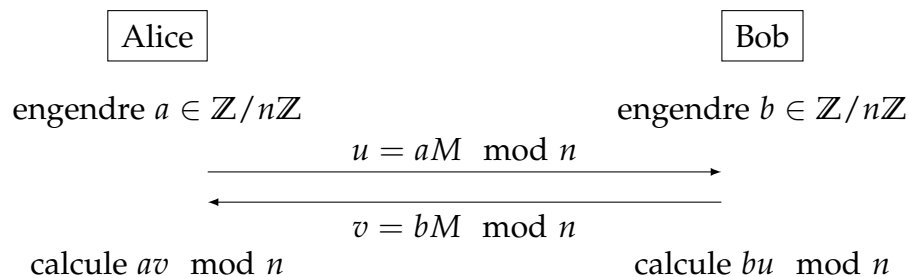
Question 3.– Démontrer que 5 est un générateur de QR_p^\times .

Question 4.– Exécuter le protocole de Diffie–Hellman dans QR_p^\times avec les valeurs suivantes :

- $p = 19$,
- $g = 5$,
- la valeur secrète d'Alice $a = 4$,
- la valeur secrète de Bob est $b = 6$.

Exercice 2. (☆☆) Protocole de Diffie–Hellman dans $(\mathbb{Z}/n\mathbb{Z}, +)$.

On considère une variante additive du protocole de Diffie–Hellman dans le groupe fini $G = (\mathbb{Z}/n\mathbb{Z}, +)$ où n est un très grand nombre entier. Pour cela, on fixe un générateur $M \neq 0$ du groupe G , que l'on publie. Dans cette version « additive », le protocole devient donc :



Question 1.– Donner une caractérisation simple des générateurs du groupe additif $(\mathbb{Z}/n\mathbb{Z}, +)$. Puis, proposer un exemple de générateur qui convient pour tout entier $n \geq 2$.

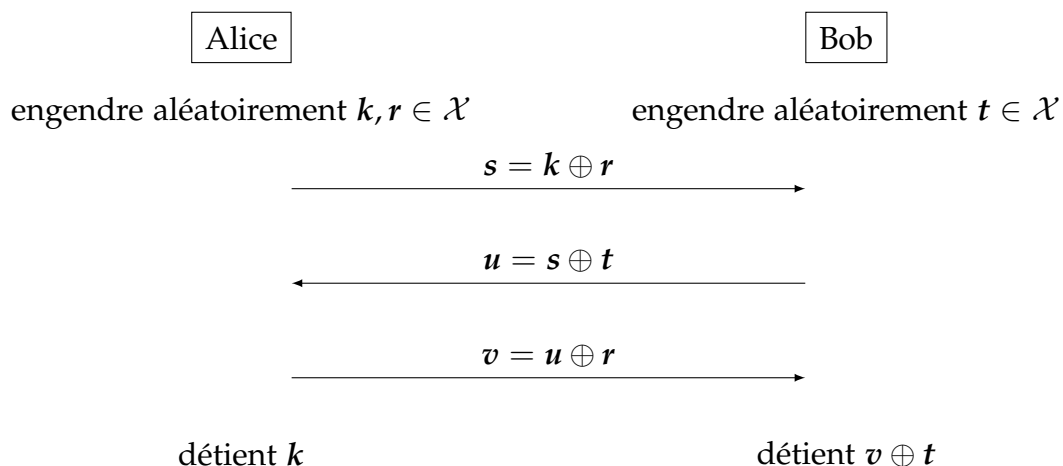
Question 2.– Vérifier que le protocole est **valide**, c'est-à-dire que lorsqu'ils suivent le protocole ci-dessus, Alice et Bob détiennent bien un secret commun.

Question 3.– Rappeler un algorithme qui calcule l'inverse modulaire dans $\mathbb{Z}/n\mathbb{Z}$ (autrement dit, qui effectue l'opération $x \mapsto x^{-1} \pmod n$). La complexité de cet algorithme est-elle polynomiale en $\log n$?

Question 4.– Selon vous, la variante du protocole de Diffie–Hellman proposée dans cet exercice est-elle sûre ? Justifier.

Exercice 3. (☆☆) Échange de clefs et masque jetable.

On note $\mathcal{X} = \mathbb{F}_2^n$ l'ensemble des chaînes de bits de longueur n , munies de l'opération de xor bit-à-bit, notée \oplus . On considère le protocole d'échange de clefs suivant.



Question 1.– Vérifier qu'Alice et Bob détiennent bien un secret commun.

Question 2.– Le protocole est-il sûr ? Justifier.

Exercice 4. () Problèmes DH et sqDH.**

Soit G un groupe cyclique de générateur g . On rappelle que le problème calculatoire de Diffie–Hellman dans G est :

CDH : étant donné (g, g^a, g^b) , calculer g^{ab} .

On définit maintenant le problème de « Diffie–Hellman carré » dans G comme :

sqDH : étant donné (g, g^a) , calculer g^{a^2} .

Question 1.– Expliquer en quoi sqDH peut être vu comme un sous-problème de CDH.

Question 2.– Supposons que l’on détienne un algorithme A qui résout efficacement sqDH. Démontrer qu’à partir d’une instance (g, g^a, g^b) de CDH, on peut calculer g^{2ab} .

Pour simplifier, on suppose ici que l’ordre r du groupe G est un entier impair et connu.

Question 3.– On dit que x est un carré dans G s’il existe un $y \in G$ tel que $x = y^2$. Décrire un algorithme qui calcule une racine carrée d’un carré $x \in G$.

Question 4.– En conclure que CDH se réduit à sqDH (dans le cas où l’ordre r est impair et connu).

Exercice 5. (**) Implantation de Diffie–Hellman dans \mathbb{QR}_p^\times .

Dans cet exercice on suppose que p est un nombre premier de taille importante **tel que $p' = (p - 1)/2$ est également un nombre premier**. Un tel nombre premier p est appelé **nombre premier sûr**, tandis que p' est un **nombre premier de Sophie Germain**.

Question 1.– Trouver dans les bibliothèques `random` et `math` de python comment :

- tirer uniformément un entier entre 1 et x ,
- calculer efficacement une puissance modulaire,
- calculer efficacement un inverse modulaire.

Question 2.– Écrire une fonction `is_square(x, p)` qui teste si un entier x est un résidu quadratique modulo p .

Question 3.– Écrire une fonction `random_QR_generator(p)` qui retourne un générateur aléatoire du groupe des résidus quadratiques modulo p . On rappelle que l'on suppose que $p' = (p - 1)/2$ est également un nombre premier.

Question 4.– Écrire les fonctions suivantes du protocole de Diffie–Hellman :

- une fonction `system_parameters(t)` qui définit les paramètres du protocole : un nombre premier p aléatoire de taille t bits tel que $p' = (p - 1)/2$ est aussi premier, et le générateur aléatoire g de \mathbb{QR}_p^\times ,
- une fonction `random_secret(p)` qui définit la valeur secrète x engendrée par un participant au protocole,
- une fonction `to_send(x, g, p)` qui déduit de la valeur secrète x et du générateur g la valeur à transmettre à l'autre participant,
- une fonction `shared_value(gy, x, p)` qui déduit de la valeur reçue $gy (= g^y)$ et de la valeur secrète x , la valeur commune aux deux participants.

Pour l'implantation de la fonction `system_parameters(t)`, on pourra notamment s'aider de la fonction `Crypto.Util.number.getPrime(t)` de la bibliothèque `Crypto.Util`, voir :

<https://pycryptodome.readthedocs.io/en/latest/src/util/util.html#Crypto.Util.number.getPrime>

Question 5.– En choisissant une taille $t = 64$ (exemple-jouet), tester vos fonctions en exécutant les actions successives d'Alice et Bob dans le protocole de Diffie–Hellman.