1 Outils généraux

Exponentiation rapide

On considère l'algorithme suivant :

Algorithme 1 : Exp

```
Input : (x, n)
if n = 0 then
    Output : 1
if n = 1 then
    Output : x
if n est pair then
    y ← Exp(x, n/2)
    Output : y * y
else
    y ← Exp(x, n/2)
    Output : y * y * x
```

1. Montrez que quelque soient les entiers positifs x et n, l'algorithme Exp renvoie la valeur de x^n.

2. Donnez une borne sur le nombre de multiplications effectué par l'algorithme. En considérant l'écriture binaire de n, essayez de raffiner cette borne.

3. Soit k un paramètre de sécurité. Pour quelles valeur de n un attaquant polynomial peut-il calculer x^n : $n = k$, $n = k^2$, $n = 2^k$, $n = k^k$, $n = 2^{2^k}$?

Fonction de hachage

- Rappelez les trois propriétés que doit vérifier une fonction de hachage.
- Soit f la fonction $x \to x^2 \mod N$. Quelle propriétés vérifie f ?
  Soit $g : \{0,1\}^* \to \{0,1\}^n$ que l'on suppose résistante à la collision. Soit $h : \{0,1\}^* \to \{0,1\}^{n+1}$ définie par
  - $h(x) = 1|x$ si x est de longueur n.
  - $h(x) = 0|g(x)$ sinon

  Quelle propriétés vérifie h ?
- Quels autres implications ou non-implications pouvez-vous trouver.

- Soit $h : \{0,1\}^* \to \{0,1\}^n$. On suppose que pour un élément m aléatoire, la probabilité d'avoir une valeur donnée $h(m)$ est de $\frac{1}{2^n}$ (l'espace d'arrivée est équiprobable). Donner une attaque pour chaque propriété de h ainsi que sa complexité.

2 Protocoles de signature

Signature RSA

On considère le protocole de signature suivant :

- KeyGen $\to (pk = (N, e), sk = d)$
- Sign$(m, d) = \sigma = m^d \mod N$
- Verif$(pk, m, \sigma) = 1$ si $m = \sigma^e \mod N$

1. Montrer que ce protocole ne résiste pas à une forge existentielle dans un attaque sans message, i.e. un attaquant peut générer un couple (m, σ) valide.
2. Supposons que $N = 437$ et $e = 17$. Connaissant les deux couples $(m_1, \sigma_1) = (100, 156)$ et $(m_2, \sigma_2) = (2, 257)$, signez le message $m = 200$.
3. Montrer que ce protocole ne résiste pas à la forge universelle dans une attaque à messages choisis.

Signature El-Gamal

Definition 1. *Soit G un groupe fini et g un générateur de G. Le problème du logarithme discret est, étant donné un élément $h \in G$ de trouver x tel que $g^x = h$.*

Soit p un nombre premier et h une fonction de hachage. Soit g un élément générateur de $G = (\mathbb{Z}/p\mathbb{Z})^*$. On considère le protocole de signature suivant :

- KeyGen $\to (pk = (p, b, g), sk = a \xleftarrow{\$} [0, p-2])$ avec $b = g^a$
- Sign$(m, sk) = \sigma = (m, r, s)$ avec $r = g^k \mod p$ où k est un élément de $[2, p-2]$ différent à chaque utilisation et $s = k^{-1}(h(m) - ra) \mod p - 1$. Si $s = 0$ recommencer avec un autre k.
- Verif(m, σ, pk) retourne 1 si et seulement si $0 \leq r \leq p-1$ et $g^h(m) = b^r r^s \mod p$

1. Quels sont les propriétés que doit vérifier une signature ?

2. Montrer que la fonction de vérification est correcte, i.e. renvoie 1 si $\sigma = \text{Sign}(m, sk)$.
3. A priori, quel élément un attaquant doit calculer pour signer. Imaginer une attaque pour trouver cet élément.
4. Quel est la taille de p à choisir pour que le système de signature ne soit pas vulnérable à cette attaque.
5. Peut-on, en terme de temps de calcul, utiliser ce système en pratique ?

Importance de k

6. Montrer que si un élément même k est utilisé deux fois il est possible de retrouver la clef secrète a.

Importance de la condition $0 \leq r \leq p-1$

Supposons que le protocole ne vérifie pas cette condition. Soit (m, r, s) une signature valide. On veut se servir de cette signature pour signer un message m'. Pour cela on calcule

$$u = h(m')h(m)^{-1} \mod p-1 \text{ et } s' = su \mod p-1$$

On trouve ensuite r' tel que $r' = ru \mod p-1$ et $r' = r \mod p$.

7. A quelle condition ces calculs sont-ils possible ?
8. Vérifier que (m', r', s') est une signature valide.
9. En supposant que la fonction h est sans collision, comparer la sécurité du protocole à celle du logarithme discret.

Importance de la fonction de hachage

Supposons que l'on utilise pas de fonction de hachage ($h(m) = m$ dans le protocole), on va réaliser une autre falsification : Soient i, j des entiers dans $[0, p-2]$, on cherche r sous la forme $g^i b^j \mod p$

10. Montrer que la relation $g^h(m) = b^r r^s$ est équivalente à $g^{y-is} = b^{r+js} \mod p$.
C'est en particulier le cas si $y - is$ et $r + js$ sont congrus à $0 \mod p-1$
11. A quelle condition ce système admet-il une solution ? Déterminer (r, s, y) en fonction de i et j.
12. Pourquoi cette attaque est-elle moins puissante que la précédente ?

3 Un peu de cryptanalyse

Algorithme rho de Pollard

On cherche à trouver un diviseur non-trivial à un entier N. On suppose que n non premier.

1. Donner un algorithme (naïf) renvoyant un diviseur non-trivial de N. Quel est sa complexité ?
   On va maintenant analyser l'algorithme suivant :

Algorithme 2 : Pollard rho

```
Input : N, f où f est une fonction pseudo-aléatoire
1 a ←$ ℤ/Nℤ
2 (a, b) ← (a, a)
3 while do
4 |   (a, b) ← (f(a), f(f(b)))
5 |   d ← pgcd(a − b, n)
6 |   if 1 < d < n then
  |   |   Output : d
7 |   if d = n then
  |   |   Output : erreur
  |
```

   Le fait que f soit pseudo-aléatoire signifie que l'on peut considérer que les éléments $f(x)$ sont uniformément et aléatoirement distribués dans $\mathbb{Z}/N\mathbb{Z}$. Par exemple $f(x) = x^2 + 1 \mod N$.

2. Pour prouver la terminaison de l'algorithme montrer qu'à un certain moment dans la boucle $a = b$.
   Indication, poser $x_0 = a$ et $x_{n+1} = f(x_n)$ et montrer que x_n est périodique.

3. Prouver que dans le cas où l'algorithme ne retourne pas d'erreur, il retourne un diviseur non trivial de N.
   Remarque : en pratique on tombe rarement dans le cas d'erreur, dans ce cas il suffit de relancer avec une valeur de a différente.

4. En utilisant le paradoxe des anniversaires, donner la complexité moyenne de l'algorithme.
   Indication : considérer la suite (x_n) mais modulo p.