Cryptographie Asymétrique

description d'un nouvel algorithme

Remy AUMEUNIER

18 janvier 2019

Le chiffrement asymétrique est apparu en 1976,avec la publication d'un ouvrage sur la cryptographie publié par Whitfield Diffie et Martin Hellman mais aussi par Ralph Merkle à la même époque.Le cryptosystème asymétrique utilise 2 clefs :une clef publique et une clef privée,ou secrète.Lorsque 2 personnes (nommées par convention Alice et Bob)veulent échanger des informations via un canal ouvert ou publique,Alice publie une clef publique, Bob code son message avec la clef public d'Alice et met à disposition d'Alice le résultat du chiffrement.Puis Alice avec sa clef privée récupère les informations codées par Bob

État de l'art Les algorithmes de cryptographie asymétrique peuvent être regroupés en 4 grandes familles.Les plus connus sont les cryptogrammes de type RSA. Cet algorithme a été décrit,en 1977 par Ronald Rivest,Adi Shamir et Leonard Adleman.Il y a aussi les cryptogrammes à courbe elliptique proposés de manière indépendante,par NealKoblitz et Victor Miller en 1985. Une courbe elliptique est un cas particulier de courbe algébrique avec laquelle on peut faire une addition,ce qui permet de définir un échange de clefs de type Diffie-Hellman.Il y a aussi le chiffrement El Gamal qui est un algorithme de cryptographie asymétrique fondé sur le problème du logarithme discret créé par l'Égyptien Taher Elgamal, doctorant de l'université de Stanford.Puis pour finir il y a aussi plusieurs cryptosystème basé sur le célèbre problème du sac à dos

Préambule J'ai souhaité, dès le début que mon algorithme de cryptographie asymétrique soit basé sur un savoir à la manier d'un Otp. Après de multiples recherches et essais infructueux malgré une certaine réussite en mathématiques, je me suis retrouvé confronté au dernier th de Fermat ou théorème de Fermat-Wiles pour mettre en musique mon OTP asymétrique.

$$x^n + y^n = z^n$$

$$z^n - y^n = x^n$$

$$z^n - y^n = (z-y)\cdot(z^n + z^{(n-1)}\cdot y + z^{(n-2)}\cdot y^2 + \cdots + z^2\cdot y^{(n-2)} + z\cdot y^{(n-1)} + y^n)$$

Cette relation, appréhendée avec de l'arithmétique modulaire

$$b^n < (a-b), b^n = a^n mod(a-b)$$

devient.

$$(b+c)^n < (a-b)$$

$$(b+c)^n = b^n + c \cdot \frac{a^n}{(a-b)} mod(a-b-c)$$

1 Présentation de l'algorithme

À partir de cette relation $(b+c)^n = b^n + c \cdot \frac{a^n}{(a-b)} mod(a-b-c)$ il devient possible de crée un algorithme de crypto asymétrique

— Alice publie deux valeurs.

$$publicAlice_1 = \frac{a^n}{(a-b)}$$

$$publicAlice_2 = (a-b)$$

— Bob publie une valeur : publicBob

$$b_1 = (publicAlice_1 mod(publicAlice_2 - n_1))$$

$$b_2 = (publicAlice_1 mod(publicAlice_2 - n_2))$$

$$b_3 = (publicAlice_1 mod(publicAlice_2 - n_3))$$

$$b_4 = (publicAlice_1 mod(publicAlice_2 - n_4))$$

$$...$$

$$b_n = (publicAlice_1 mod(publicAlice_2 - n_n))$$

$$publicBob = b_1 - b_2 + b_3 - b_4$$

— Alice calcule la somme des entiers $(n_1 - n_2 + ... - n_n)$ de Bob avec la clef privée d'Alice b et un coefficient $\frac{n!}{(n-p)! \cdot p!}$

$$\frac{(b+n_1)^n - b^n}{n_1} = \frac{a^n}{(a-b)} mod(a-b-n_1)$$

$$publicBob = \frac{(b+n_1)^n - b^n}{n_1} - \frac{(b+n_2)^n - b^n}{n_2} + \frac{(b+n_3)^n - b^n}{n_3} - ...$$

$$publicBob = c_1 \cdot b^{(n-2)} \cdot (n_1 - n_2 + n_3 - n_4 + ...) + c_2 \cdot b^{(n-3)} \cdot (n_1^2 - n_2^2 + n_3^2 - n_4^2 ...) + c_3 ...$$

$$\frac{publicBob}{c_1 \cdot b^{(n-2)}} = (n_1 - n_2 + n_3 - n_4 + ...) + c_2 \cdot b^{(n-3)} \cdot (n_1^2 - n_2^2 + n_3^2 - n_4^2 ...) + c_3 ...$$

$$\lfloor \frac{publicBob}{c_1 \cdot b^{(n-2)}} \rfloor = (n_1 - n_2 + n_3 - n_4 + ...)$$

2 Étude de sécurité

— Alice

$$publicAlice_1 = \frac{a^n}{(a-b)}$$

$$publicAlice_2 = (a-b)$$

— Bob

$$publicBob = \sum_{k=0} -1^k \cdot \frac{(b+n_k)^n - b^n}{n_k} = \sum_{k=0} -1^k \cdot (\frac{a^n}{(a-b)} mod(a-b-n_k))$$

— Alice

$$\lfloor \frac{publicBob}{\frac{n!}{(n-2)!\cdot 2!} \cdot b^{(n-2)}} \rfloor = \sum_{k=0} -1^k \cdot n_k$$

Pour évite tout attaque de $publicAlice_1$ et $publicAlice_2$ ou l'utilisation de la relation $\frac{a^n}{(a-b)} mod(a-b) = n \cdot b^{(n-1)}$ je propose la modification suivante

$$publicAlice_1 = \frac{c}{(a-b)} = \frac{q \cdot a^n + r}{(a-b)}$$

$$publicAlice_2 = (a - b + bruitPriveeAlice)$$

Puis Alice, calcul la clef privée de Bob

$$\lfloor \frac{publicBob}{q \cdot \frac{n!}{(n-2)!\cdot 2!} \cdot b^{(n-2)}} \rfloor$$

La modification de $publicAlice_2$ ne perturbe pas le calcule parce que le bruit est de bonne taille et Bob à un **nombre pair** d'éléments dans la somme. La modification de $publicAlice_1$ introduit une constante qui ne peut pas être complètement aléatoire. Par contre l'arithmétique modulaire permet de multiples variations dans l'écriture de $publicAlice_1$ Ces variations non pas vocation a être publique donc oui on peut en théorie retrouvé b mais pour cela il faut connaître la forme de

$$publicAlice_1 = \frac{a^n}{(a-b)}$$

$$publicAlice_1 = \frac{q \cdot a^n + r}{(a-b)}$$

$$publicAlice_1 = \frac{a^n + a^m}{(a-b)}$$

$$publicAlice_1 = \frac{....}{(a-b...)}$$

Bob peut lui aussi introduire un bruit :

$$publicBob = bruitPriveeBob + \sum_{k=0} -1^k \cdot \frac{(b+n_k)^n - b^n}{n_k}$$

Cela devrait permettre d'avoir plusieurs sommes possibles dans le cadre d'une attaque par force brute de publicBob puisque qu'il n'y a aucune indication sur le nombre d'éléments qui compose la somme, sous réserve de démonstration.

3 Application numérique

Sous linux ctrl+alt+T puis bc -l avec un (copier/coller). Sous Windows installer un terminal linux à partir du Windows store choisir Ubuntu par exemple puis bc -l et (copier/coller)

```
remy@....:~$ bc -l
bc 1.06.95
Copyright 1991-1994,..., 2004, 2006 Free Software Foundation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type 'warranty'.
##########Alice##########
scale=0
n=5;q=101^500;bruitalice2=7777777777777

a=9876543211^300+5
b=1234567891^40
c=q*a^n+123456789

publicalice1=c/(a-b)
publicalice2=a-b-bruitalice2
##########Bob##########
#la clef privee de bob a pour valeur(n1-n2+n3-...-nm) m est pair
scale=0
n1=2443;n2=3656;n3=17325;n4=17;bruitpriveebob=141413215144263611
(n1-n2+n3-n4)
16095
b1=(publicalice1%(publicalice2-n1))
b2=(publicalice1%(publicalice2-n2))
b3=(publicalice1%(publicalice2-n3))
b4=(publicalice1%(publicalice2-n4))

publicbob=b1-b2+b3-b4+bruitpriveebob
##########Alice#######
#alice calcul la clef privee de bob (n1-n2+n3-n4+...-nm)
scale=0
define f(x) {if (x>1){return x*f(x-1)};return 1}
coef=f(n)/(f(n-2)*2)

publicbob/(q*coef*b^(n-2))
16095
##########################
```

4 Attaque par force brut

Cette attaque consiste à extraire b. Je fais l'hypothèse que l'attaquent connaît la structure par exemple

$$publicAlice_1 = \frac{c}{(a-b)} = \frac{q \cdot a^n + r}{(a-b)}$$

$$publicAlice_2 = (a - b + bruitPrieeAlice)$$

$$\left(\frac{(publicAlice_1 \cdot (publicAlice_2 - x_1)) - x_2}{x_3}\right)^{1/x_4} = a$$

donc $(x_1 \cdot x_2 \cdot x_3 \cdot x_4)$ Ce qui revient parcourir n fois q.

Un attaquant peut aussi recherche à reconstruire *publicBob* sachant que le nombre d'éléments est inconnues, mais pair. Cela implique une complexité en $O(n!)$ la seule véritable attaque possible puisqu'un attaquant ne peut pas connaître les structures de $publicAlice_1, publicAlice_2$

5 Remarque

Pourquoi cela ne fonctionne pas avec mes valeurs.C'est à cause de l'arithmétique modulaire, de manier plus simple vous définissez vos valeurs pour que $(b+n_1)^n = b^n + n_1 \cdot (\frac{a^n}{(a-b)} mod(a - b - n_1))$ une fois définie a,b,n,n_1 ne pas oublier que $(b + n_1)^n < (a - b)$ vous introduisez le ou les bruits et coefficient .Cela implique qu'Alice définit ses valeurs, pour une taille max .Dit différemment Bob ne peut pas envoyer plus de 2^n bits .Cela peut être vu comme une limitation ou une indication lie aux clefs.$publicAlice_1$,$publicAlice_2$,$valeurMax = 2^{1024}$

6 Conclusions

La création d'un algorithme de cryptographie asymétrique et loin d'être trivial, il n'en existe que très peu ,**réellement exploitable** 2 peut être 3.Cet algorithme a était déposer et bénéficie d'une protection judiciaire. Pour de plus amples informations, veuillez me contacter les droits peuvent être vendu de manier total, ou sous forme de licences.

Références

[1] *Whitfield Diffie et Martin Hellman.* (en) « *New directions in cryptography* ». *IEEE Transactions on Information Theory, vol. 22, no 6, 1976, p. 644-654*

[2] *Liste d'algorithmes de cryptographie asymétrique.*(fr)
lien wikipedia