



Signature Numérique de l'Information

Naima HADJ-SAID - Adda ALI PACHA – A. BELGORAF – B. BELMEKKI

Université des Sciences et de la Technologie d'Oran USTOBP 1505 El M'Naouer Oran 31036 ALGERIE

Tél / Fax: (213) 041 / 462685

E.Mail : nim_hadj@hayoo.fr

Summarise:

The signature on documents has been used for a long time to prove the identity of their author or at least the agreement of the signatory with content of the document. The invention of cryptography with public key made possible the use of another known process under the name of numerical signature. To sign the documents numerically, consists in adding some information with these documents, which proves to the recipient of the message that you are the sender of the message. Our work consists in developing the electronic signature algorithm DSS, and it is in this context that this communication is intended. The numerical signature is very much used for the commercial transactions.

Résumé :

La signature manuscrite sur des documents a été longtemps utilisée pour prouver l'identité de leur auteur ou du moins l'accord du signataire avec le contenu du document. L'invention de la cryptographie à clé publique a rendu possible l'utilisation d'un autre processus connu sous le nom de signature numérique. Signer les documents en numérique, cela consiste à rajouter une information à ses documents qui prouve au destinataire du message que c'est bien vous l'expéditeur du message. Notre travail consiste à développer l'algorithme de la signature électronique DSS et, c'est dans ce contexte que cette communication est destinée. La signature numérique est très utilisée pour les transactions commerciales.

Mots Clés : DSA, DSS, RSA, Signature Numérique, Empreinte Digitale, Fonction de Hachage, Clé Publique.

1. Introduction

L'algorithme de signature digitale DSA a été publié par le NIST (National Institute of Standards and Technology) pour la norme DSS (Digital Signature Standard), qui fait partie du projet Capstone du gouvernement des Etats Unis. Le DSS a été choisi en coopération avec le NSA (National Security Agency) pour être la norme d'authentification en mai 1994.

Le DSA est basé sur les propriétés des logarithmes discrets, et dérivé des systèmes de cryptage proposés par Schnorr et ElGamal. Il est destiné uniquement à l'authentification.

Pour l'algorithme DSA la génération de la signature est plus rapide que la vérification, tandis que pour l'algorithme RSA[6] c'est l'inverse. Le NIST prétend que l'avantage de DSA est la rapidité à signer, mais beaucoup d'utilisateurs [Site 1, Site 2, Site 3, Site 4] pensent que la vérification doit être l'opération la plus rapide.

La **vérification** de la signature suppose que le destinataire possède la clé de l'émetteur. La plupart du temps, ce dernier peut l'envoyer, incluse dans un certificat, en même temps que le message.

L'**authentification** des messages est également possible. La clé secrète de l'expéditeur peut être utilisée pour coder un message et l'envoyer avec l'original, Ceci créer une signature numérique du message, que le destinataire (ou n'importe qui d'autre) peut vérifier en utilisant la clé publique de l'expéditeur afin de la décoder. Ceci prouve que

l'expéditeur est bien la personne qui à l'origine du message, et que le message n'a pas été modifié dans l'intervalle par une autre personne, puisque l'expéditeur est le seul à posséder la clé secrète qui a fait la signature. Toute contrefaçon d'un message signé est impossible, de même l'expéditeur ne peut pas désavouer sa signature par la suite.

2. Models de Signatures

On peut signer ses documents de telle manière que le monde entier puisse vérifier en quelques instants que c'est vous qui à signer sans que personne ne puisse signer à votre place sans un calcul préalable faramineux. On peut aussi signer pour un correspondant unique sans que les autres puissent comprendre qui a signé.

1. La signature ne peut pas être falsifier.
2. Une signature est authentique.
3. Une signature n'est pas réutilisable.
4. Un document signer ne peut pas être inaltérable (ne pas modifier parés la signature).
5. Une signature ne peut pas être reniée (ne pas nier après avoir signer).

Dans la réalité ces caractéristiques ne sont pas toujours vrai car avec la tricherie la signature peut être falsifier ou transférer. Le document signer peut être modifié sans laisser de trace. D'autant plus que cela pose un problème si on travail avec des ordinateurs car l'image graphique de la signature manuscrite (scanner) est facile a recopier et a transférer à un autre document ainsi que le document.

Il existe deux grands modes de signatures [3, 5, 7].

1. Signature à l'aide d'un crypto système à clé secrète et avec un arbitre : l'émetteur A envoie le message chiffré avec sa clé K_A vers l'arbitre, ce dernier déchiffre le message avec K_A , et certifie la réception du message de l'émetteur. Ensuite il l'envoie le message et la certification chiffré avec K_B vers le récepteur B qui déchiffre le message avec K_B (obtient message clair + certification).

Cette méthode peut rendre la signature valable mais elle est trop coûteuse en temps pour l'arbitre qui joue le rôle d'intermédiaire entre tout paire de correspondant qui veulent s'échanger les documents signés ce qui produira l'encombrement de demande sur lui.

Même si l'arbitre est un programme on se pose la question comment l'entretenir ?

Ce programme doit être infaillible, le risque d'erreur doit être nul. Si sa base de données de clés secrètes venait d'être divulguée ou si quelqu'un modifie son code toute signature deviendrait inutilisable.

2. Dans le système cryptographique à clé publique chaque entité possède une clé privée K_S utilisée pour générer la signature et une clé publique k_B pour la vérification.

3. Empreinte Digitale d'un Document

Pour signer le message que vous expédiez, il suffit en effet de leur appliquer une fonction mathématique (appelée fonction de hachage) qui produit un résumé du message (code haché). Le résumé obtenu est propre à chaque message, à l'instar d'une empreinte digitale qui permet également d'identifier l'émetteur et de certifier le document.

Elle peut ensuite être chiffrée à l'aide de votre clé privée et annexée à votre message. C'est ce code qui constitue la **signature numérique**. Cette empreinte digitale (code haché) peut être séparée du document permettant ainsi de générer une base de données qui s'appuie sur le stockage des empreintes des fichiers et en associant une date à chaque empreinte interdisant ainsi l'accès aux fichiers.

Afin de signer de longs documents on utilise une fonction de hachage à sens unique pour générer une empreinte digitale qui sera signée à la place du document c'est à dire au lieu de signer le document tout entier on signe son empreinte.

Le destinataire du message peut ensuite vérifier que vous en êtes bien l'expéditeur en déchiffrant la signature numérique, au moyen de votre clé publique, pour obtenir le code haché. Le destinataire applique ensuite la même fonction de hachage au message reçu ; si les deux codes sont identiques, vous êtes bien l'expéditeur du message (non - répudiation) et le message n'a pas été altéré (intégrité).

Il y a deux méthodes utilisées pour la signature de l'empreinte:

• A l'aide de chiffrement

C'est l'œuvre de plusieurs algorithmes cryptographiques [1, 3, 4] comme celui du RSA, elle s'obtient en associant une opération de hachage et une opération de chiffrement, c'est à dire:

1. Création par l'émetteur d'un condensé du message par une opération de hachage.
2. Chiffrement asymétrique du condensé par la clé privée de l'émetteur. Ceci constitue la signature.
3. Envoi des deux informations (message et signature) au destinataire.

Le destinataire qui reçoit ces deux informations doit, pour vérifier la signature, procéder ainsi :

4. Calcul à nouveau du condensé du message par le même algorithme que celui utilisé par l'émetteur.
5. Déchiffrement de la signature en utilisant la clé publique de l'émetteur. Cela permet de reconstituer le condensé créé par l'émetteur.
6. Comparaison des deux informations condensées ainsi obtenues. Si elles sont identiques, seul l'émetteur (le processeur de la clé privée) a pu envoyer ce message.

• Sans l'aide de chiffrement

C'est une nouvelle technique créée pour que la signature soit propre sans avoir procédé à faire le chiffrement de l'empreinte. On utilise l'empreinte du document et la clé privée de l'émetteur pour faire un traitement mathématique. Le résultat de ce traitement (valeur numérique) sera la signature numérique du document. L'émetteur A calcule l'empreinte du document avec la fonction de hachage et signe, à l'aide de l'algorithme de la signature, l'empreinte avec sa clé secrète (K_S). Cette signature est associée au document original.

Le destinataire recevra le document avec sa signature, il calcule avec la fonction de hachage uniquement l'empreinte du document et vérifie, à l'aide de l'algorithme de la signature, l'empreinte avec la clé publique de A (K_P). Le résultat de ce traitement sera comparé avec la signature envoyée et s'ils sont égaux donc la signature est valide.

L'algorithme DSA (Digital Signature Algorithm) est un exemple de ce type de signature.

4. Algorithme de Hachage Sécurisé

L'utilisation de l'algorithme de hachage sécurisé [5] SHA (Secure Hash Algorithm) est requis pour toutes les applications fédérales et plus précisément pour toutes les opérations de signature digitale basées sur le standard DSS. Étant donné un message de longueur inférieure à 2^{64} bits, l'algorithme SHA produit un condensé de 160 bits de ce message, appelé condensé. Ce condensé est utilisé à la fois,

dans l'opération de signature électronique du message et dans le processus de vérification de cette signature.

Tout message M peut toujours, après un codage adéquat, être considéré comme une suite de bits. En pratique, on peut même supposer que la longueur l d'une telle suite est un entier strictement inférieur à 2^{64} .

La première étape de l'algorithme SHA est une étape de remplissage (*padding* en anglais) consistant à rajouter des bits au message M de telle sorte que la longueur du nouveau message (en bits) soit un multiple exact de 512.

- On ajoute d'abord un 1 à droite du message. Soit M' le message obtenu en rajoutant le bit 1 à la fin de M .
- On calcule ensuite l'entier m le plus petit tel que $(1 + l + 64 + m)$ soit un multiple entier de 512.
- On ajoute alors m bits tous égaux à 0 à la fin de M' . Soit M'' le message résultant et $PAD(M'')$ son expression en caractères hexadécimaux.
- Pour terminer le remplissage, on rajoute à la fin de $PAD(M'')$ les deux mots de 8 caractères hexadécimaux dérivés du mot de 64 bits codant la longueur initiale du message M . On notera N le message rempli résultant.

L'algorithme SHS opère sur des blocs de 512 bits. Un bloc de 512 bits peut être considéré comme une suite de 16 mots de 32 bits, un mot de 32 bits peut à son tour, être considéré comme une suite de 8 symboles hexadécimaux c'est-à-dire appartenant à l'ensemble $\{0,1,2,\dots,A,B,\dots,E,F\}$, cela on convertissant chaque groupe de 4 bits du mot en son équivalent hexadécimal.

Soit le mot $Y=(0001110101100111001010110011)$ qui est converti en une suite de 8 groupes à 4 bits qui donne, en finale la chaîne de 8 symboles hexadécimaux (0001) (1101) (0110) (0111) (0010) (1011) (0011), $Y=1D6572B3$.

SHA utilise 80 fonctions logiques définies sur des mots de 32 bits et 80 constantes de 32 bits. Les 80 fonction logique f_t produisant en sortie des mots de 32 bits et prennent chacune en entrée trois mots de 32 bits, B , C et D , leur expression est donnée par :

$$\begin{aligned} f_t(B,C,D) &= (B \wedge C) \vee (\sim B \wedge D) && \text{pour } 0 \leq t \leq 19; \\ f_t(B,C,D) &= B \text{ XOR } C \text{ XOR } D && \text{pour } 20 \leq t \leq 39; \\ f_t(B,C,D) &= (B \wedge C) \vee (B \wedge D) \vee (C \wedge D) && \text{pour } 40 \leq t \leq 59 \\ f_t(B,C,D) &= B \text{ XOR } C \text{ XOR } D && \text{pour } 60 \leq t \leq 79. \end{aligned}$$

L'expression en hexadécimale des 80 constantes k_t utilisées par l'algorithme SHA est donnée par

$$\begin{aligned} k_t &= 5A827999 && \text{pour } 0 \leq t \leq 19 \\ k_t &= 6ED9EBA1 && \text{pour } 20 \leq t \leq 39 \\ k_t &= 8F1BBCDC && \text{pour } 40 \leq t \leq 59 \\ k_t &= CA62C1D6 && \text{pour } 60 \leq t \leq 79 \end{aligned}$$

Pour calculer le condensé d'un message M , l'algorithme SHA opère sur le message rempli N (qui peut donc être considéré comme une suite de n blocs de 512 bits, $M_1, M_2, \dots, M_n$, où chaque M_i est lui-même considéré comme une suite de 16 mot de 64 bits) et traite en séquence chacun des blocs M_i .

Le traitement de chacun de ces blocs demandant 80 étapes. L'algorithme utilise deux mémoires tampons de 5 mots de 32 bits chacune et une suite de 80 mots de 32 bits. Traditionnellement, les mots de la première mémoire tampon sont notés A, B, C, D, E , et ceux de la deuxième H_0, H_1, H_2, H_3, H_4 . Les 80 mots de la suite sont notés $W_0, W_1, \dots, W_{79}$. Enfin une dernière mémoire tampon $TEMP$ de 32 bits est utilisée.

L'algorithme SHA est alors exécuté à la manière suivante :

Initialisation

Le contenu des mots H_i sont initialisés comme suit:

$$\begin{aligned} H_0 &= 67452301 \\ H_1 &= \text{EFCDA8B9} \\ H_2 &= 98BADCFE \\ H_3 &= 10325476 \\ H_4 &= \text{C3D2E1F0} \end{aligned}$$

Traitement

La procédure suivante est exécutée successivement pour $M_1, M_2, \dots, M_n$.

- Division du bloc M_1 en 16 mots de 32 bits, $W_0, \dots, W_{15}$, où W_0 est constitué des 32 bits les plus à gauche de M_1 .
- Pour $16 \leq t \leq 79$, soit :
 $W_t = S^1(W_{t-3} \text{ XOR } W_{t-8} \text{ XOR } W_{t-14} \text{ XOR } W_{t-16})$
- Poser $A := H_0, B := H_1, C := H_2, D := H_3, E := H_4$.
- Pour $0 \leq t \leq 79$, exécuter la boucle :
 $TEMP = S^5(A) + f_t(B,C,D) + E + W_t + k_t$;
 $E := D; D := C; C := S^{30}(B); B := A; A := TEMP$.
- Poser $H_0 := H_0 + A, H_1 := H_1 + B, H_2 := H_2 + C, H_3 := H_3 + D, H_4 := H_4 + E$.

A la fin du traitement de M_n , le condensé du message M est tout simplement la chaîne de 160 bits représentée par les mots H_0, H_1, H_2, H_3, H_4 .

5. Notions Mathématiques

La théorie des nombres est devenue depuis une vingtaine d'années un vivier dans lequel la science de la cryptologie est venue puiser [2]. La cryptographie classique manipule des caractères par contre la cryptographie moderne est orientée vers la manipulation des bits (chiffres en binaires), et utilise avec abondance des résultats des fonctions de l'arithmétique.

5.1 Nombres Premiers

Un nombre p est premier s'il admet exactement deux diviseurs, 1 et lui-même (1 n'est pas premier).

Il est facile de vérifier de tête que 7, 13 ou 31 sont des nombres premiers.

Mais quelle méthode adopter pour montrer que 4999 est premier ?

Essayer toutes les divisions de 4999 par D allant de 2 à 4998. Si aucune division ne tombe juste, alors on peut affirmer que 4999 est premier.

Comment créer une liste des nombres premiers ?

La méthode proposée par Eratosthène donne une solution (**Crible Eratosthène**). Ecrire tous les nombres de 2 à 1000. 2 est premier, on le souligne et on raye tous les multiples de 2. Le premier nombre non rayé est 3. Il est donc premier, on le souligne et on raye tous ses multiples. Etc. Arrivé à 32, qui supérieur à la racine carrée de 1000, on a terminé : Tous les nombres qui n'ont pas été rayés sont premiers.

La méthode devient inefficace avec des nombres très grands (par exemple pour une table des nombres premiers jusqu'à 10 000 000).

Existe-t-il des grands nombres premiers ?

La réponse a été fournie par Euclide qui affirme qu'un nombre est soit premier, soit admet un diviseur premier. S'il admet un diviseur premier, on effectue la division et on recommence avec le quotient obtenu (**critère de Lehmer** qui permet de savoir si un entier N est premier lorsque l'on connaît la décomposition en facteurs premiers de N-1).

En effet, si on suppose que cet ensemble est infini, il est composé de n nombres $p_1, p_2, \dots, p_n$ alors $N = p_1 * p_2 * \dots * p_n + 1$ est premier.

5.2 Congruences

Chacun sait que s'il est 3 heures, dans 2 heures il sera 5 heures. Normal puisque $3+2=5$. De même, s'il est 11 heures, dans 2 heures il sera 1 heure. Normal puisque $11+2=1$.

Comment arrive-t-on à ce résultat surprenant ?

En fait le reste de la division de 13 par 12 est 1 : les calculs d'heure se font modulo 12. On devrait en fait écrire $11+2=1 \pmod{12}$ ou encore $11+2=1 [2]$

a et b sont congrus modulo n s'ils ont un même reste par la division par n.

5.3 Petit Théorème de Fermat

Si p premier et p ne divise pas a (c'est à dire a et p premier entre eux). Alors $a^{p-1} = 1 [p]$.

Si p premier et a entier alors $a^p = a [p]$.

5.4 Fonctions à Sens Unique

La théorie de la calculabilité et de la complexité algorithmiques introduit une classe de fonctions dites à sens unique ou one way, dont l'existence est toujours une conjecture

La notion de sens unique signifie que tout algorithme de calcul de l'inverse donne un résultat au bout d'un temps extrêmement long (la complexité de l'algorithme est exponentielle). La rapidité ne dépend pas de la puissance du processeur qu'on utilise.

f est à sens unique $\Leftrightarrow$

$$\left\{ \begin{array}{l} \forall x \ y = f(x) \text{ est calculable rapidement} \\ x = f^{-1}(y) \text{ se calcule en temps très long} \end{array} \right.$$

C'est cette disproportion du temps de calcul de f et f^{-1} qui va être mise à profit pour faire bénéficier de la rapidité des processeurs de calculs les cryptographes, au détriment légitime des cryptanalystes.

5.5 Fonctions de Hachage à Sens Unique :

Une fonction de hachage H est la transformation d'un bloc d'informations m de taille variable en un bloc d'informations de taille fixe qui est la valeur de hachage. On a donc $h = H(m)$.

- H(x) doit être rapidement calculable pour tout x.
- H(x) ne doit pas être facilement inversible.
- H(x) doit être sans collisions.

5.6 Algorithme Discret

La fonction $a^p \pmod n$ de la variable p, est appelée exponentielle modulaire. L'algorithme de cette fonction se programme d'une manière récursive, en effectuant les deux opérations en même temps :

$$a^p \pmod n = \begin{cases} \bullet 1 & \text{si } p = 0 \\ \bullet (a^{p/2} \pmod n)^2 \pmod n & \text{si } p \text{ est pair} \\ \bullet (a(a^{p-1} \pmod n)) \pmod n & \text{si } p \text{ est impair} \end{cases}$$

Le calcul d'une exponentielle modulaire se fait en temps logarithmique par rapport aux données, c'est à dire très rapidement.

Leurs inverses, qui sont difficilement calculables, portent le nom de **logarithmes discrets**.

Etant donné un nombre premier P et F_Q un corps à Q éléments, où $Q = P^n$ est une puissance de ce nombre premier, étant donné d'autre part G un générateur et deux éléments quelconques non nuls X et Y, dans $]0, Q^{n-1}[$, il est :

- Très facile de calculer G^X dans F_Q .
- Infaisable calculatoirement de trouver un élément a de $[0, Q^{n-1}]$ tel que $G^a = Y$

L'entier a joue alors l'équivalent du logarithme de Y dans F_Q , d'où la terminologie employée du logarithme discret.

6. Standard de Signature Digitale DSS

Le DSA [3,5] utilise les paramètres suivants:

$$\begin{aligned} W &= (S')^{-1} \bmod Q \\ U1 &= (\text{SHA}(M') * W) \bmod Q \\ U2 &= (R' * W) \bmod Q \\ V &= ((G^{U1})(Y^{U2})) \bmod P \bmod Q \end{aligned}$$

61626380	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000
00000000	00000000	00000000	00000000	00000000
00000018	61626380	00000000	00000018	61626308
00000000	00000018	61626380	00000018	61626398
61626380	00000000	00000018	61626380	00000000
00000018	61626380	00000000	00000018	61626398
61626380	00000018	61626380	61626398	00000000
00000000	00000000	00000018	61626398	61626380
00000000	00000000	00000018	61626398	61626398

79

SIGNATURE NUMERIQUE DE L'INFORMATION.

61626380	00000018	61626380	00000018	61626398
61626398	61626398	61626398	61626380	00000018
61626398	61626380	00000018	61626380	00000000
00000000	00000000	00000018	61626380	00000000
00000000	00000000	00000018	61626380	00000018
61626398	61626380	00000018	61626380	00000000

♦ Les 80 itérations dans SHA conduisent au résultat suivant dans les registres A, B, C, D, E :

T	A	B	C	D	E
0	116fc33	67452301	7bf36ae2	98badcf	10325476
1	8990536d	116fc33	59d148c0	7bf36ae2	98badcfe
2	a1390f08	8990536d	c045bf0c	59d148c0	7bf36ae2
3	Cdd8e11b	a1390f08	626414db	c045bf0c	59d148c0
4	Cfd499de	cdd8e11b	284e43c2	626414db	c045bf0c
5	3fc7ca40	cf499de	f3763846	284e43c2	626414db
6	993e30c1	3fc7ca40	b3f52677	f3763846	284e43c2
7	9e8c07d4	993e30c1	b3f52677	ff1f290	f3763846
8	4b6ae328	9e8c07d4	664f8c30	ff1f290	b3f52677
9	8351f929	4b6ae328	27a301f5	664f8c30	ff1f290
10	Fbda9e89	8351f929	12dab8ca	27a301f5	664f8c30
11	63188fe4	60d47e4a	fbda9e89	27a301f5	12dab8ca
12	4607b664	63188fe4	7ef6a7a2	60d47e4a	12dab8ca
13	9128f695	4607b664	18c623f9	7ef6a7a2	60d47e4a
14	196bee77	9128f695	1181ed99	18c623f9	7ef6a7a2
15	20bdd62f	196bee77	644a3da5	1181ed99	18c623f9
16	Ed2ff4a3	20bdd62f	c65afb9d	644a3da5	1181ed99
17	565df73c	ed2ff4a3	c82f758b	c65afb9d	644a3da5
18	550b1e7f	565df73c	fb4bfd28	c82f758b	c65afb9d
19	Fe0f9e4b	550b1e7f	15977dcf	fb4bfd28	c82f758b
20	B4d4c943	fe0f9e4b	d542c79f	15977dcf	fb4bfd28
...	...	...	...	...	...
78	24ff7ed5	70286c07	f7f9f921	2e1b1690	86027578
79	9a1f95a8	24ff7ed5	dc0a1b01	f7f9f921	2e1b1690

On effectue ensuite :

$$\begin{aligned}
 h_0 &= 67452301 + 9a1f95a8 \\
 h_1 &= efcdab89 + 24ff7ed5 \\
 h_2 &= 98badcfe + dc0a1b01 \\
 h_3 &= 10325476 + f7f9f921 \\
 h_4 &= c3d2e1f0 + 2e1b1690
 \end{aligned}$$

Et l'empreinte numérique est finalement la concaténation $h_0h_1h_2h_3$:

0164b8a914cd2a5e74c4f7ff82c4d97f1edf880.

La signature du message sera :

R = 284868967 S = 147829589

La vérification de cette signature avec la clé publique correspondante $Y = 1284365718$, les mêmes paramètres et le même message "abc" nous donne le résultat suivant :

$$\begin{aligned}
 W &= 159792454 \quad U1 = 44430355 \\
 U2 &= 233151472 \quad V = 284868967.
 \end{aligned}$$

On remarque que $V = R = 284868967$ donc la signature est valide.

Si on essaye de modifier le contenu du message :

$$\begin{aligned}
 \text{Pour "aBc"} \quad W &= 159792454 \quad U1 = 300369226 \\
 \quad \quad \quad U2 &= 233151472 \quad V = 787599657
 \end{aligned}$$

On a $V \neq R$ donc la signature est invalide.

Pour "abc_" (_ représente un blanc après c)

$$\begin{aligned}
 W &= 159792454 \quad U1 = 285540440 \\
 U2 &= 233151472 \quad V = 193148431
 \end{aligned}$$

On a $v \neq R$ donc la signature est invalide.

La conversion en hexadécimal des messages 'abc', 'aBc' et 'abc_' (_ représente un blanc après c) donnent des valeurs différentes donc de point de vue binaire ces messages sont différents, même si l'on considère qu'ils portent la même information.

Autre Résultat : Identification :

Supposant qu'un individu nommé **M. kamel** signe numériquement (avec DSA) un chèque pour un autre individu **B. karim**, le format du message (contenu du chèque) sera :

Payez à Monsieur B.Karim la somme de 20.000 DA.,

Les paramètres utilisés sont :

$$P = 2031487547 \quad Q = 92340343 \quad G = 1868585202.$$

M. kamel possède une clé secrète $X = 90548893$ et une clé publique $Y = 1883055406$. Il génère $K = 54457712$ pour signer.

L'empreinte est :

79758d8e408626a43111c4feff04793b8402867a.

Alors la signature du message sera :

$$R = 39030301 \quad S = 92178625$$

B. karim va à la banque pour encaisser l'argent, le caissier entre la clé publique du signataire et les valeurs R, S pour vérifier la signature.

Résultat de la vérification avec les vraies paramètres :

$$\begin{aligned}
 W &= 33009854 \quad U1 = 56902802 \\
 U2 &= 84436945 \quad V = 39030301
 \end{aligned}$$

$V = R = 39030301$, alors la signature est valide.

Si **karim** triche, et joue à ajouter quelque zéro à la somme, le message sera par exemple : *Payez à Monsieur B.Karim la somme de : 20.000.000 DA*

L'empreinte change à : **7613e4ed9dc0055d7c73b8f3e45d66f90d09c30.**

donc la vérification donne une autre résultat.

$$\begin{aligned}
 W &= 33009854 \quad U1 = 23472820 \\
 U2 &= 84436945 \quad V = 22241998
 \end{aligned}$$

$V \neq R$, la signature est invalide

Ou un autre tiers, prend le chèque et le modifie-en :
Payez à Monsieur O.Ali la somme de: 900000000DA.
 L'empreinte change à:
15c60a11543e2f8635588ee5dcee35348b32f5a.

Et le résultat de la vérification donne:
 w=33009854 u1=73592670
 u2=84436945 v=38543254

$V \neq R$, la signature est invalide.

7.2 Image

Supposant que le message est une image. Soient deux images de l'EMIR ABDEKADER à signer avec les paramètres $P=2134277773$, $Q=16168771$, $G=1870579918$, l'émetteur A possède une clé secrète $X_A=12345678$, et une clé publique $Y_A=1031463729$ et signe ces messages avec $K=16111111$.



Image N°1

Image N°2

Empreinte de l'image N°1 est:
705342a4c6becd31d2bafc13713b0a5cb1075185

Empreinte de l'image N°2 est:
Eeb1cbf412efa41636b5fdead07a71352b344372

L'algorithme est conçu de telle manière que toute altération du message durant sa transmission produira, avec une probabilité très forte une empreinte différente de l'empreinte originale.

7.3 Unicité des Paramètres

L'utilisation de l'algorithme DSA tel qu'il est décrit par le NIST engendre une signature valide mais on suppose qu'il existe un problème de l'unicité de la signature.

Description du Problème

Etant données la signature DSA :

$$R = (G^k \bmod P) \bmod Q$$

$$S = (K^{-1}(H(M) + X \cdot R)) \bmod Q$$

$H(M)$ de longueur 160 bits est l'empreinte du message M. Le paramètre Q doit être : $Q > H(M)$

Donc l'utilisation d'un nombre inférieur à $H(M)$ comme paramètre Q peut produire deux messages différents ou plus qui ont des différentes empreintes mais partage la même signature S.

Solution Proposée

Pour remédier à ce problème on a choisi de découper l'empreinte en 20 sous-empreinte inférieur à Q et calculer S_i pour chacune d'elles (ajouté sa position lors du calcul). La concaténation des S_i donne l'élément S2 ajouté à la signature S1 et R.

Pour la vérification on calcul pour chaque S_i de la chaîne S2 les nombres ($W_i, U1_i, U2_i, V_i$), et on calcul en plus ($W, U1, U2, V$) pour S1. La vérification de la signature est lorsque tous les V_i égal à R.

Exemple :

On va signer le message :

M= "Il ne faut jamais dévoiler votre clé secrète.....!!!"

les paramètres $P=2147483587$, $Q=357913931$, $G=1049573710$ $K=123456$, la clé secrète $X=123456$.

La signature sera :

R : 69029973 S1 : 27511547

S2 :

00f3ea4d0562c38600131186026333fa04610bc5014b
 923d03e2ed24042a42cf00396aff01510cbc02376002
 04cebae903fe6ed704b3566e04668644032e058d03dd
 72a501b9416100ee6fce01c9b0de

La vérification avec la clé publique $Y=1437098679$, produira les résultats suivants:

pour S1 :

$W=5632320$ $U1=69131213$

$U2=68006546$ $V=69029973$

Et pour S2 :

$W[1]=62987643$ $U1[1]=27641225$

$U2[1]=24362059$ $V[1]=69029973$

$W[2]=63412707$ $U1[2]=39752766$

$U2[2]=21073308$ $V[2]=69029973$

$W[3]=58917095$ $U1[3]=39138999$

$U2[3]=22768995$ $V[3]=69029973$

$W[4]=47810401$ $U1[4]=54792821$

$U2[4]=85892665$ $V[4]=69029973$

$W[5]=36814462$ $U1[5]=82606327$

$U2[5]=60296579$ $V[5]=69029973$

$w[6]=31217838$ $U1[6]=70612968$

$U2[6]=76493629$ $V[6]=69029973$

$W[7]=71640493$ $U1[7]=25386901$

$U2[7]=24350348$ $V[7]=69029973$

$W[8]=15877073$ $U1[8]=21232463$

$U2[8]=17958362$ $V[8]=69029973$

$W[9]=82582719$ $U1[9]=13565181$

$U2[9]=77139507$ $V[9]=69029973$

$W[10]=64159955$ $U1[10]=90057102$

$U2[10]=14313353$ $V[10]=69029973$

$W[11]=53812289$ $U1[11]=3855441$

U2[11]=2358207 V[11]=69029973
W[12]=49836276 U1[12]=67706520
U2[12]=57222888 V[12]=69029973
W[13]=9900836 U1[13]=87493737
U2[13]=70912875 V[13]=69029973
W[14]=85622989 U1[14]=28199637
U2[14]=86686962 V[14]=69029973
W[15]=11563356 U1[15]=82275532
U2[15]=63772773 V[15]=69029973
W[16]=34863935 U1[16]=39626532
U2[16]=12531456 V[16]=69029973
W[17]=80236768 U1[17]=82002633
U2[17]=3742661 V[17]=69029973
W[18]=27163729 U1[18]=38829665
U2[18]=88151585 V[18]=69029973
W[19]=68122264 U1[19]=570676
U2[19]=91853058 V[19]=69029973
W[20]=44797431 U1[20]=37294447
U2[20]=20547823 V[20]=69029973

On remarque que tous les $V_i = V = 69029973 = R$ donc la signature est valide.

8. Conclusion

Le développement rapide des réseaux mondiaux et les immenses possibilités offertes par les transactions électroniques posent aujourd'hui de manière cruciale le problème de la protection des informations échangées sur les réseaux informatiques mondiaux, d'où l'utilisation des techniques cryptographiques et surtout les signatures numériques est devenue de plus en plus indispensables comme moyens de confidentialité, d'authentification et d'intégrité des données transmis.

L'étude sur les différents systèmes cryptographiques à clés secrète et surtout ceux à clés publique comme le **RSA**, **ELGAMEL** nous a permis de connaître et de détailler les différents paramètres qui entre dans les protocoles cryptographiques de signature, ainsi les fonctions de hachages utiliser pour générer l'empreinte digitale nécessaire pour la signature et sa vérification.

Le **DSA** qui est issu directement des principes élémentaires de l'arithmétiques des congruences. Son niveau de sécurité dépend de la difficulté de Logarithme discret. La combinaison de **DSA** avec le chiffrement engendre un outil qui peut assurer toute la confidentialité et l'authenticité.

La publication du **DSS** a en effet généré un nombre impressionnant de critique, en raison notamment de l'influence visible de la **NSA** (National Security Agency) dans la conception fine de l'algorithme, laissant penser au plus grand nombre que des limitations sécuritaires y auraient été volontairement introduites, afin d'en permettre une cryptanalyse aisée par la **NSA** elle-même. La même suspicion existe d'ailleurs, depuis l'origine, concernant les critères de conception du **DES** (Data Encryption Standard), qui n'ont jamais été publiés. En ce sens, **DSS** a reçu un accueil plutôt très défavorable de l'industrie ; celle-ci avait en effet misé sur une standardisation officielle de l'algorithme **RSA**, par ailleurs très souvent implémenté comme technique d'authentification et de signature digitale dans les réseaux informatiques classiques comme le prouve d'ailleurs la popularité grandissante du programme de chiffrement **PGP** (Pretty Good Privacy).

9. Références

- [1]: B.BECKETT, Introduction aux méthodes de la cryptologie, Edition MASSON 1990.
- [2]: J.DUBERTET, Initiation à la cryptographie, 2^{ème} Edition VEBERT, Avril 2000.
- [3]: B.SCHNEIER, Cryptographie appliquée, John Wiley & Sons, Inc 1994.
- [4]: X.MARCAULT, Compression et cryptage des données multimédias, Edition HERMES 1995.
- [5]: Techniques de l'ingénieur, traité informatique H 2000.
- [6]: K.BENAMAR et K MERINI, 'Etude et Implémentation de la Signature Numérique :DSA', Mémoire d'ingéniorat, institut informatique USTO 2001
- [7]: A.ALI PACHA, N.HADJ SAID, 'Sécurisation des données par le système cryptographique à clé publique RSA', Conférence Internationale sur la Productique CIP'01, CDTA, 9 au 11 Juin 2001 Alger

[Site1]: <http://pup.chez.com/cgi-bin/popup:pl//autre>.

[Site2]: <http://www.rsasecurity.com>.

[Site3]: <http://www.chez.com/nopb/crypto.html>.

[Site4]: <http://cuisung.unige.ch/memoires/Hugentobler/certnum.html>.