Cryptographie à clé publique II

Bruno MARTIN,
Université Nice Sophia Antipolis

Plan

Un autre problème difficile

DLP : problème du logarithme discret de y en base g

Problème

INSTANCE : $g, y \in G$, *groupe fini*
QUESTION : *trouver* x *tel que* $g^x \equiv y$ *dans* G

ou, pour p un grand premier, g un générateur de $G = \mathbb{Z}_p^\star$,

$$g^x \equiv y \mod p \text{ et } x = \log_g(y) \mod p$$

Plus généralement, tout $y \in G$ possède un logarithme discret en base g ssi G cyclique de générateur g.

Exemple

Soit $G = \mathbb{Z}_7^\star$ groupe cyclique, d'ordre 6.

- en base 2, seuls $1, 2$ et 4 possèdent un logarithme discret ;
- en base g=3, on obtient le tableau :

nombre y	1	2	3	4	5	6
logarithme	6	2	1	4	5	3

Par exemple, pour nombre $= 1$ et log $= 6$, cela signifie que $\log_3 1 = 6$ (dans G), ce qu'on vérifie par $3^6 \mod 7 = 1$.

Calcul du logarithme discret – Shanks

S'applique à tout groupe fini G.
Complexité en temps $O(\sqrt{|G|}\log|G|)$ en espace $O(\sqrt{|G|})$
Idée : construire deux listes de puissances de g :

- une liste de petits pas $\{g^i : i = 0..\lceil\sqrt{n}\rceil - 1\}$ avec $n = |G|$
- une liste de pas de géant $\left\{y\left(g^{-\lceil\sqrt{n}\rceil j}\right) : j = 0..\lceil\sqrt{n}\rceil\right\}$.

Puis trouver un terme commun aux 2 listes. Ainsi,

$$g^{i_0} = y(g^{-j_0\lceil\sqrt{n}\rceil}) \text{ et } m = i_0 + j_0\lceil\sqrt{n}\rceil$$

Calcul du log. discret de groupes de faible cardinalité facile, opération difficile quand le cardinal de G croît.

Exemple

On travaille dans $\mathbb{Z}_{113}^{\times} =< 3 >$ d'ordre $n = 112$; $\sqrt{n} = r = 11$. On cherche le logarithme discret de $y = 57$ en base $g = 3$:
Liste (non ordonnée) des petits pas, forme (exposant, valeur) :

$$B = \{(0,1),(1,3),(2,9),(3,27),(4,81),\\ (5,17),(6,51),(7,40),(8,7),(9,21),(10,63)\}$$

Liste (non ordonnée) des pas de géant, forme (exposant, valeur) :

$$L = \{(0,57),(1,29),(2,100),(3,37),(4,112),(5,55),(6,26),\\ (7,39),(8,2),(9,3),(10,61),(11,35)\}$$

3 valeur commune aux 2 listes, engendré pour $i_0 = 1$ dans B et $j_0 = 9$ dans L. Le logarithme discret est $x = i_0 + r.j_0 = 100$.
Vérification : on calcule $g^x \mod 113 = 57$.


```
1  from sympy import *
2  g,n,r,y=3,113,11,57
3  B=[(i,pow(g,i,n)) for i in range(r)]
4  L=[(j,y*gcdex(pow(g,j*r,n),n)[0]\%n) for j in range
   (r+1)]
5  B=sorted(B, key=lambda x:x[1])
6  L=sorted(L, key=lambda x:x[1])
7  print(B,L)
```

```
[(0, 1), (1, 3), (8, 7), (2, 9), (5, 17), (9, 21), (3, 27), (7, 40), (6, 51), (10, 63), (4, 81)]
[(8, 2), (9, 3), (6, 26), (1, 29), (11, 35), (3, 37), (7, 39), (5, 55), (0, 57), (10, 61), (2, 100), (4, 112)]
```

```
1  1+r*9
```

100

Shanks

Algorithme :
Entrée : n le cardinal de G,g et $y \in G$
Sortie : logarithme discret de y en base g dans G. $r := \lceil\sqrt{n}\rceil$
Construire la liste $B := \{g^i : i = 0..\lceil\sqrt{n}\rceil - 1\}$
Construire la liste $L := \{y(g^{-\lceil\sqrt{n}\rceil j}) : j = 0..\lceil\sqrt{n}\rceil\}$
Trier les listes B et L selon un ordre sur les g^i et les g^{-rj}.
Trouver i_0 et j_0 tel que : $g^{i_0} = yg^{-rj_0}$.
RETURN$(i_0 + j_0 r)$.
Preuve :
Pour $r = \lceil\sqrt{n}\rceil$, on a :
Construction des listes : $r + 1 + r$ opérations de groupes : $O(r)$
Tri des listes : $O(r\log(r))$
Recherche d'un même élément dans deux listes triées : $O(\log(r))$ □

Plan

Le chiffre d'El Gamal [1]

Repose sur DLP.

1. choisir p premier t.q. DLP est difficile dans ${\mathbb{Z}_p}^\star$
2. choisir un générateur $\alpha \in {\mathbb{Z}_p}^\star$
3. choisir $2 \leq a < p-1$, la **clé privée**
4. calculer $\beta \equiv \alpha^a \mod p$
5. **clé publique :** p, α, β.

Chiffrer : $E : (x,k) \mapsto (y_1 = \alpha^k \mod p, y_2 = x\beta^k \mod p)$
pour k aléatoire secret de ${\mathbb{Z}_{p-1}}^\star$
Déchiffrer : $(y_1, y_2) \mapsto y_2 {y_1}^{-a})^{-1} \mod p$

ElGamal Rockstar

Fonctionnement

Alice écrit à Bob en utilisant $pk = (p, \alpha, \beta)$

$$E : (x,k) \mapsto (y_1 = \alpha^k \mod p, y_2 = x\beta^k \mod p)$$

k aléatoire secret de ${\mathbb{Z}_{p-1}}^\star$ (on a donc un chiffrement probabiliste).
Clair x est « masqué » par β^k.
Valeur de α^k transmise comme partie du chiffré comme y_1.

Bob, avec sk a, calcule $\beta^k = \alpha^{ak}$. Comme $\beta = \alpha^a$, il calcule $(\alpha^k)^a = \beta^k$. Reste à multiplier y_2 par $(\beta^k)^{-1} \mod p = x$.

Observons que le chiffré est deux fois plus long que le clair.

Exemple

Soit $p = 2579, \alpha = 2$, $a = 765$. $\beta = 2^{765} \mod 2579 = 949$.
Alice veut transmettre $x = 1299$ à Bob.
Elle choisit $k \in \mathbb{Z}^\star_{p-1} = 853$ et calcule (mod 2579) :

$$y_1 = 2^{853} = 435 \qquad y_2 = 1299.(949)^{853} = 2396$$

Bob reçoit $(435, 2396)$ et connait $a = 765$. Il calcule $(y_1)^a$ mod $p = 435^{765} \mod 2579 = 2424$, cherche l'inverse mod p par Euclide étendu : $(2424, 2579) = -599.2424 + 563.2579 = 1 \Leftrightarrow (\beta^k)^{-1} = -599 = 1980$.
Puis il calcule $2396.1980 \mod 2579 = 1299$.

Difficulté du logarithme discret

$G = \langle g \rangle$ d'ordre $p-1$, $\forall y \in G, \exists! x : g^x = y$ (on note $x = \log_g y$)
DLP : donnés g, y, calculer x dans G.

$\mathcal{G}$ algo polytime : entrée 1^n retourne $G = \langle g \rangle$ d'ordre $p-1$.
Expérience $\mathrm{DLog}_{A,\mathcal{G}}(n)$:

- lance $\mathcal{G}(1^n)$ pour avoir (G, p, g) (générateur de groupe)
- $y \xleftarrow{u} G$
- A reçoit (G, p, g, y) et retourne x
- résultat expérience 1 si $g^x = y$ sinon 0

Définition (Hypothèse DH)

DLP est difficile pour $\mathcal{G}$ si pour tout algo A PPT, il existe negl :
$Pr(DLog_{A,\mathcal{G}}(n) = 1) \leq negl(n)$

Les problèmes CDH et DDH permettent de construire de bons $\mathcal{G}$

Computational Diffie Hellman CDH

$$\langle g \rangle = G \quad y_1, y_2 \in G \quad DH_g(y_1, y_2) \stackrel{\Delta}{=} g^{\log_g y_1 \log_g y_2}$$
$$y_1 = g^x \quad y_2 = g^z \Rightarrow DH_g(y_1, y_2) = g^{xz} = (y_1)^z = (y_2)^x$$

CDH = calculer $DH_g(y_1, y_2)$ pour y_1, y_2 choisis au hasard

Si DLP relatif à $\mathcal{G}$ facile, CDH aussi
Si log discret difficile, CDH difficile ?

Decisional Diffie Hellman DDH

DDH revient à distinguer $DH_g(y_1, y_2)$ d'un élément aléatoire de G : pour $y_1, y_2 \xleftarrow{u} G$ et y' une solution, DDH revient à décider si $y' = DH_g(y_1, y_2)$ ou si $y' \xleftarrow{u} G$.

Définition

DDH est difficile relativement à $\mathcal{G}$ si, pour tout algo D PPT, il existe negl. tq

$$|Pr(D(G,p,g,g^x,g^y,g^z)=1) - Pr(D(G,p,g,g^x,g^y,g^{xy})=1)| \leq negl(n)$$

pour $\mathcal{G}(1^n)$ qui renvoie (G, p, g) et $x, y, z \xleftarrow{u} \mathbb{Z}_p$

Lemme utile

Le multiple d'un élément y tiré uniformément est unif. distribué.
Ou, y' ne contient pas d'information sur m.

Lemme

G groupe fini, $m \in G$ qcq. Les distrib. de probabilité relatives à :

- $y \xleftarrow{u} G$ *et* $y' := m \cdot y$
- $y' \xleftarrow{u} G$

sont identiques. Autrement dit, $\forall \hat{y} \in G, Pr(m \cdot y = \hat{y}) = 1/\sharp G$

$\hat{y} \in G$ qcq. Alors $Pr(m \cdot y = \hat{y}) = Pr(y = m^{-1} \cdot \hat{y})$.
Comme $y \xleftarrow{u} G$, la proba. pour que y soit un élément donné de G est $1/\sharp G$ □

ElGamal (rappel)

Le chiffre d'ElGamal Π est (mod p) :

- Gen :$1^n \mapsto \mathcal{G}(1^n) \mapsto (G, p, g)$, $\quad a \xleftarrow{u} \mathbb{Z}_p^\star$, $\quad \beta := g^a$; $pk = (G, p, g, \beta)$ et $sk = (G, p, g, a)$
- E : reçoit (pk, m) ; $k \xleftarrow{u} \mathbb{Z}_p^\star$; renvoie $c = (y_1, y_2) = (g^k, m\beta^k)$
- D : reçoit sk et $c = (y_1, y_2)$; renvoie $m := y_2(y_1^a)^{-1}$

Théorème

Si DDH est difficile pour $\mathcal{G}$, ElGamal est IND-CPA.

Comparer fonctionnement de Π à celui de Π' qui ressemble syntaxiquement à Π en remplaçant toutes ses sorties par des VA.

On montre IND-EAV plutôt que IND-CPA (équivalent).

IND-EAV : $\text{PubK}^{\text{EAV}}_{A,\Pi} \equiv \text{PubK}^{\text{CPA}}_{A,\Pi}$

1. Gen(1^n) produit (pk, sk)
2. A reçoit pk et retourne $m_0, m_1 \in M(pk)$ de même long.
3. $b \xleftarrow{u} \{0,1\}$; $c \leftarrow E_{pk}(m_b)$ et envoyer c à A
4. A retourne un bit b'
5. A réussit l'expérience (i.e. renvoie 1) ssi $b = b'$

Définition

Π est CPA-sûr si, pour tout adversaire A PPT, il existe negl(.) tq :

$$Pr(PubK^{CPA}_{A,\Pi}(n) = 1) \leq \frac{1}{2} + negl(n)$$

On pose : $\varepsilon(n) = Pr(\text{PubK}^{\text{EAV}}_{A,\Pi}(n) = 1)$ qu'il faut évaluer.

Le chiffre Π', pas déchiffrable mais convenable pour A

- Gen' =Gen rend $pk = (G, p, g, \beta)$ et $sk = (G, p, g, a)$
- E : donné (pk, m) ; $y, z \xleftarrow{u} \mathbb{Z}_p^\star$; rend $c = (y_1, y_2) = (g^y, mg^z)$

Par le lemme :

- y_2 unif. distribué sur G et indépendant de m
- y_1 est unif. distribué et indépendant de m.

On ne tire pas d'information sur m à partir de c :

$$Pr(\text{PubK}^{\text{EAV}}_{A,\Pi'}(n) = 1) = \frac{1}{2}$$

Algo D PPT qui résout DDH relativement à $\mathcal{G}$

D reçoit pk et c :$(G, p, g, g^x = \beta, g^y = y_1, g_3)$ avec $g_3 = \begin{cases} g^{xy} \\ g^z \end{cases}$ où $x, y, z \xleftarrow{u} \mathbb{Z}_p^\star$.

- $pk := (G, p, g, g^x = \beta)$ et appelle A pour obtenir m_0, m_1
- $b \xleftarrow{u} \{0,1\}$; $y_1 := g^y$ et $y_2 := m_b g_3$
- donne (y_1, y_2) à A qui renvoie b'
- résultat expérience 1 si $b' = b$ sinon 0

Deux comportements possibles pour D selon g_3

$g_3 = g^z$ A, appelé par D fonctionnera ĉ ds $\text{PubK}^{\text{EAV}}_{A,\Pi'}(n)$ sur un chiffré de la forme (g^y, mg^z). Donc

$$Pr(D(\text{pk}, g^y, g^z) = 1) = Pr(\text{PubK}^{\text{EAV}}_{A,\Pi'}(n) = 1) = 1/2$$

$g_3 = g^{xy}$ A, appelé par D fonctionnera ĉ ds $\text{PubK}^{\text{EAV}}_{A,\Pi}(n)$ sur un chiffré de la forme $(g^y, m(g^x)^y)$. Donc

$$Pr(D(\text{pk}, g^y, g^{xy}) = 1) = Pr(\text{PubK}^{\text{EAV}}_{A,\Pi}(n) = 1) = \varepsilon(n)$$

par hyp. DDH difficile pour $\mathcal{G}$ donc $\exists$ negl. tq negl$(n)\geq$

$$\left| \begin{array}{l} Pr(D(\text{pk}, g^y, g^z) = 1) - \\ Pr(D(\text{pk}, g^y, g^{xy}) = 1) \end{array} \right| = |1/2 - \varepsilon(n)| \Rightarrow \varepsilon(n) \leq 1/2 + \text{negl}(n)$$

Partage des paramètres

Dans la définition d'ElGamal, on demande aux sujets de lancer $\mathcal{G}$ pour engendrer G, p, g. En pratique, ces paramètres sont souvent engendrés une fois pour toute.
P.e. un admin système peut fixer ces paramètres pour un paramètre de sécurité donné n et tout le monde peut partager ces valeurs.

Dans un BSD sous `/etc/moduli`. Demander `man moduli`

```
DESCRIPTION The /etc/moduli file contains prime
numbers and generators for use by sshd in the
Diffie-Hellman Group Exchange key exchange method.
```


Objectifs des PKI

- **confidentialité :** message chiffré doit rester secret.
- **authentification :** assurance de l'authenticité (expéditeur/origine)
- **identification :** déclaration de son identité
- **intégrité :** message n'a pas été altéré durant la transmission
- **non répudiation :** l'expéditeur ne peut nier l'envoi du message

Techniques utilisées

- **signature :** moyen d'associer l'expéditeur à un message
- **certificat :** attestation (d'un tiers) confirmant une affirmation (d'identité)
- **tiers de confiance :** autorité qui délivre les certificats
- **estampillage :** ajout dates ou jetons $\rightarrow$ unicité du message

Où se tenir au courant ?

Tous les standards sont enregistrés dans les PKCS [1] qui se trouvent à `https://arxiv.org/abs/1207.5446`

> *These standards cover RSA encryption, Diffie-Hellman key agreement, password-based encryption, extended-certificate syntax, cryptographic message syntax, private-key information syntax, and certification request syntax, as well as selected attributes.*

1. Public Key Cryptographic Standards

Signatures

But traditionnel de la crypto : assurer la confidentialité.
Autre application : les signatures introduites par Diffie et Hellman.
But des signatures : garantir **intégrité** et **authentification**.
Signature dépend de l'id. du signataire et du contenu du message.

Signature empêche deux types de fraudes :

- la falsification de la signature par le destinataire ;
- la non-reconnaissance du message par l'expéditeur.

Utilisation électronique légale depuis la loi 2000-230 du 13/3/2000 Art.3 :L'écrit sur support électronique a la même force probante que l'écrit sur support papier

Cahier des charges

Le cahier des charges d'une signature $\Sigma(m)$ est :

- elle doit être calculable par le signataire pour tout message m ;
- tout individu (surtout le destinataire) peut la vérifier ;
- elle doit être impossible à falsifier ;
- l'expéditeur ne doit pouvoir affirmer que sa signature a été imitée.

Mécanisme général de signature Σ

Une signature est composée de 3 algos PPT :

- génération de clés noté **gen** (pk, sk) fonction de 1^n
- signature (privée) noté **sig** qui, pour une clé fixée sk, retourne une signature s pour un clair m ;

$$\text{sig}_{sk}(m) = s$$

- vérification (déterministe et publique) noté **ver** qui, à une clé fixée pk et pour tout couple clair/signature (m, s) va vérifier si la signature correspond bien au clair.

$$\text{ver}_{pk}(m, s) = \begin{cases} \text{vrai si } s = \text{sig}_{sk}(m) \\ \text{faux si } s \neq \text{sig}_{sk}(m) \end{cases}$$

Expérience Sig-forge$_{A,\Sigma}(n)$

Formalise l'attaque du faussaire (Adversaire change m et calcule une signature valide en usurpant l'identité de Bob).

L'expérience Sig-forge$_{A,\Sigma}(n)$:

1. Gen(1^n) produit (pk, sk)
2. A reçoit pk et l'accès à un oracle $\text{Sig}_{sk}(.)$[2]. A renvoie (m, s). Soit Q l'ensemble des messages pour lesquels A a eu recours à l'oracle
3. A réussit l'expérience (i.e. renvoie 1) ssi
   1. $\text{ver}_{pk}(m, s) = 1$ et
   2. $m \notin Q$

2. l'oracle fournit $\text{sig}_{sk}(m)$ à tout m choisi par A

Sécurité des signatures Sig-forge$_{A,\Sigma}(n)$

Définition

Un mécanisme de signature Σ est existentiellement infalsifiable pour une attaque adaptative à messages choisis si pour tout adversaire A PPT, il existe negl tq :

$$Pr(Sig\text{-}forge_{A,\Sigma}(n) = 1) \leq negl(n)$$

Signer avec RSA

Bob désire envoyer un message M signé à Alice. Paramètres RSA :

	Privés	Publics
Alice	d_A	n_A, e_A
Bob	d_B	n_B, e_B

Procédé de signature :

$$\text{sig}_{sk}(M) = M^{d_B} \mod n_B = S$$

Vérification :

$$\text{ver}_{pk}(M, S) = \text{vrai} \Leftrightarrow S^{e_B} \mod n_B \equiv M$$

Exemple d'envoi d'un message secret signé (RSAKE)

Comment Bob peut-il envoyer à Alice un message secret signé ?
Fonctions de chiffrement et de déchiffrement d'Alice et Bob :

	Privés	Publics
Alice	$D_A(C) = C^{d_A} \mod n_A$	$E_A(M) = M^{e_A} \mod n_A$
Bob	$D_B(C) = C^{d_B} \mod n_B$	$E_B(M) = M^{e_B} \mod n_B$

Bob envoie le message $C = E_A(D_B(M))$
Et Alice le déchiffre en $E_B(D_A(C))$
Pour cela, il faut que $M < n_B < n_A$.

Falsification sans message

Faussaire reçoit $pk = (n, e)$, choisit $s \in \mathbb{Z}_n^\star$ calcule $m := s^e \mod n$

(m, s) valide qui n'a pas été signé par le propriétaire de sk !

Faussaire n'a pas le choix de m mais... que se passe-t-il dans le cas d'une authentification par défi ?

Falsification d'un message choisi

Si le faussaire obtient 2 signatures du propriétaire de sk il peut signer un message m de son choix :

- $m_1 \xleftarrow{u} \mathbb{Z}_n^\star$
- $m_2 := m.m_1^{-1} \mod n$
- obtient s_1 et s_2, signatures de m_1 et m_2
- $s := s_1 s_2 \mod n$ signe m !

Vous acceptez de signer n'importe quoi ?
Et vous savez si votre protocole favori signe ce qu'on lui présente ?

C'est plus sûr en hachant !

- En remplaçant m par $h(m)$, la signature par RSA devient plus sûre (hashed RSA signature scheme).
- Il faut h résistante aux collisions, sinon, on trouve $m_1 \neq m_2$ avec $h(m_1) = h(m_2)$ qui donnent les mêmes signatures.
- Les 2 attaques précédentes échouent (du fait de CR de h)
- Utilisé en pratique ; sous certaines hypothèses, prouvé sûr avec SHA-1.

Signature par El Gamal

Soit p un nombre premier pour lequel DLP est difficile dans $\mathbb{Z}_p^\times$ et α un générateur de $\mathbb{Z}_p^\times$. Le message $M \in \mathbb{Z}_p^\times$ et sa signature est $(M, S) \in \mathbb{Z}_p^\times \times \mathbb{Z}_p^\times \times \mathbb{Z}_{p-1}$. L'ensemble des clés est $K = \{(p, \alpha, a, \beta) : \beta = \alpha^a \mod p\}$

Secrets	Publics
a	p, α, β

On choisit $k \in \mathbb{Z}_{p-1}^\times$ aléatoire et secret qui vérifie $\gcd(k, p-1) = 1$
On définit une signature comme :

$$\mathrm{sig}_K(M, k) = (\gamma, \delta)$$

pour

$$\gamma = \alpha^k \mod p \qquad \delta = (M - a\gamma)k^{-1} \mod (p-1)$$

Exemple

Soit $p = 467$ et $a = 127$. On a bien que $\gcd(a, p-1) = 1$. Soit $\alpha = 2$ un générateur de $\mathbb{Z}_p^\times$. On calcule

$$\beta = \alpha^a \mod p = 2^{127} \mod 467 = 132$$

Si Bob veut signer le message $M = 100$ pour la valeur aléatoire $k = 213$ qui est tq $\gcd(k, p-1) = 1$, il calcule $k^{-1} \mod p-1$ par Euclide étendu qui donne $k^{-1} = 431$ alors,

$$\gamma = \alpha^k \mod p = 2^{213} \mod 467 = 29$$

et

$$\delta = (M - a\gamma)k^{-1} \mod (p-1) = (100 - 127.29).431 \mod 466 = 51$$

Fonctionnement – Vérification

Pour $M, \gamma \in \mathbb{Z}_p^\times$ et $\delta \in \mathbb{Z}_{p-1}$, on définit

$$\text{ver}_K(M, \gamma, \delta) = \text{vrai} \Leftrightarrow \beta^\gamma \gamma^\delta \equiv \alpha^M \mod p$$

Si la signature est construite correctement, la vérification authentifie la signature car :

$$\begin{aligned} \beta^\gamma \gamma^\delta &\equiv \alpha^{a\gamma} \alpha^{k\delta} \mod p \\ &\equiv \alpha^M \mod p \end{aligned}$$

en utilisant le fait que $a\gamma + k\delta \equiv M \mod (p-1)$

Exemple

On vérifie la signature de $(100, 29, 51)$ *par* $\text{ver}_K(M, \gamma, \delta) = \text{vrai}$ *:*

$$\beta^\gamma \gamma^\delta \equiv \alpha^M \mod p \Leftrightarrow 132^{29} 29^{51} \equiv 2^{100} \mod p \equiv 189$$

Digital Signature Standard DSA – 1991

- Variante d'El Gamal qui diminue la taille de la signature.
- DSS-DSA proposé en 1991 par D.W. Kravitz (NSA) ; adopté en 1993.
- Un module de 512 bits d'El Gamal donne une signature de 1024 bits. DSA : avoir une signature plus courte.
- Par une astuce, DSA raccourcit les tailles en offrant une signature de 320 bits sur un message de 160 bits en impliquant un module de 512 bits. Astuce : travailler dans un sous-groupe de $\mathbb{Z}_p^\times$ de taille 2^{160}.

Fonctionnement de DSS

Soit p un premier de 512 bits, q facteur premier de 160 bits de $p-1$ et α racine q^e primitive de 1 modulo p tq DLP dans le sous-groupe engendré par α est difficile. Message $h(M) \in \mathbb{Z}_p^\times$; signature : $(h(M), S) \in \mathbb{Z}_p^\times \times \mathbb{Z}_q \times \mathbb{Z}_q$. L'ensemble des clés est $K = \{(p, q, \alpha, a, \beta) : \beta = \alpha^a \mod p\}$

Privé	Publics
a	p, q, α, β

Choisir $1 < k \leq q-1$ aléatoire et secret ; la signature est :

$$\text{sig}_K(h(M), k) = (\gamma, \delta)$$

pour

$$\gamma = \alpha^k \mod q \quad \delta = (h(M) + a\gamma)k^{-1} \mod q$$

Exemple

On choisit $q = 101$ et $p = 78q + 1 = 7879$. Une racine primitive de $\mathbb{Z}_{7879}$ est 3 et on peut prendre

$$\alpha = 3^{78} \mod 7879 = 170$$

On suppose que $a = 75$, on a :

$$\beta = \alpha^a \mod 7879 = 4567$$

Bob souhaite signer M d'empreinte $h(M) = 1234$ et choisit comme valeur de $k = 50$. $k^{-1} \mod 101 = 99$. On a

$$\gamma = (170^{50} \mod 7879) \mod 101 = 2518 \mod 101 = 94$$

et

$$\delta = (1234 + 75.94)99 \mod 101 = 97$$

La signature du message d'empreinte 1234 est alors $(94, 97)$

Vérification par DSS

Pour $h(M) \in \mathbb{Z}_p^\times$ et $\gamma, \delta \in \mathbb{Z}_q$, $\text{ver}_K(h(M), \gamma, \delta) = \text{vrai} \Leftrightarrow (\alpha^{e_1}\beta^{e_2} \mod p) \mod q = \gamma$

$$\text{pour} \begin{cases} e_1 & = & h(M)\delta^{-1} \mod q \\ e_2 & = & \gamma\delta^{-1} \mod q \end{cases}$$

NB : $\delta \not\equiv 0 \mod q$ car $\delta^{-1} \mod q$ est nécessaire à la vérification.
Si on a $\delta \equiv 0 \mod q$, il faut choisir une nouvelle valeur de k.

Exemple

La signature de 1234 est $(94, 97)$ *; vérification :*

$\delta^{-1} = 97^{-1} \mod 101 = 25$
$e_1 = 1234.25 \mod 101 = 45$; $e_2 = 94.25 \mod 101 = 27$

et $(170^{45}4567^{27} \mod 7879) \mod 101 = 2518 \mod 101 = 94$

Inconvénient

Procédé plus lent que RSA d'un facteur compris entre 10 et 40.
Générer des clés est plus rapide que pour RSA.
Autre inconvénient : clé de 512 bits trop petite. La taille des clés DSA passe à 1024 ou 2048 (et le sous-groupe à 224 ou 256 bits).
La fonction de hachage est aussi en train de changer dans FIPS 186-3 qui devra utiliser SHA-224/256/384/512.


T. ElGamal.
A public-key cryptosystem and a signature scheme based on discrete logarihms.
IEEE trans. on Info. Theory, 31(4) :469–472, 1985.