Prelegerea 1

Semnături electronice

1.1 Consideraţii generale

Vom lua în discuţie în această secţiune noţiunea de semnătură electronică (într-un mediu de calcul) precum şi diverse modalităţi de utilizare ale ei.

Orice semnătură pe un document autentifică acest document dar şi angajează în mod normal responsabilitatea semnatarului. Probleme practice legate de rapiditatea transmiterii unor documente care să fie certificate ca autentice prin semnătură au condus la necesitatea creerii de semnături electronice.

De exemplu, se ştie că majoritatea operaţiunilor şi tranzacţiilor bancare devin legal valide *numai după* ce ambele părţi au semnat formularele respective. Totuşi, dacă părţile sunt legate într-o reţea de calculatoare, ele vor adesea să faciliteze această operaţie care provoacă un mare consum de timp; solicită de aceea posibilitatea de a semna documentele folosind terminalele şi reţeaua aflată la dispoziţie.

Deci, apare următoarea problemă:

Cum se poate crea o semnătură într-un mediu de calcul ?

Deoarece calculatoarele acceptă informaţia numai în formă digitală, orice semnătură pusă în discuţie trebuie să aibă această formă. O semnătură (electronică sau olografă) trebuie să satisfacă următoarele condiţii:

- **Unică**: o anumită semnătură trebuie să poată fi generată numai de o singură persoană;
- **Neimitabilă**: nici o altă persoană nu va putea genera semnătura utilizatorului indicat; altfel spus, utilizatorii ilegali trebuie să rezolve probleme $NP-complete$ dacă vor să folosească o semnătură care nu le aparţine;
- **Uşor de autentificat**: orice destinatar legal şi orice arbitru (în cazul unor eventuale dispute) să poată stabili autenticitatea semnăturii (indiferent după ce interval de timp);
- **Imposibil de negat**: nici un utilizator legal să nu-şi poată nega propria semnătură, sub afirmaţia că nu este autentică;
- **Uşor de generat**.

Trebuie făcută totuşi distincţie între semnătura olografă şi cea digitală.

Iată câteva diferenţe notabile între cele două tipuri de semnături:

- O semnătură scrisă de mână este o confirmare fizică a unui document, cu ajutorul unei foi de hârtie care conţine două elemente: un mesaj (textul docu-mentului) şi o semnătură. O astfel de legătură între mesaje şi semnături nu este posibilă într-un mediu de calcul;
- O semnătură olografă este aceeaşi indiferent de document. Pentru semnăturile digitale însă, este esenţial ca ele să depindă atât de semnatar cât şi de conţinu-tul documentului;
- Orice copie a unui document electronic (inclusiv semnătura) este identică cu originalul. În schimb copia unui document pe hârtie este diferită ca valoare de original. Aceasta conduce la ideea că un document electronic nu este reutili-zabil. De exemplu, dacă *Bob* trimite lui *Alice* un cec în valoare de 10 milioane lei, banca nu va accepta onorarea sa decât o singură dată.

1.2 Protocoale de semnătură

Orice protocol de semnătură este format dintr-un algoritm de semnătură şi un algoritm de verificare. *Bob* semnează un mesaj x bazat pe un algoritm (secret) de semnătură sig. Rezultatul $sig(x)$ este apoi verificat de un algoritm public de verificare ver. Pentru orice pereche (x, y), algoritmul de verificare oferă un răspuns dicotomic (*adevărat* sau *fals*), după cum y este o semnătură autentică a lui x sau nu. Formal ([8]):

Definiţia 1.1 *Un* **protocol de semnătură** *este un cvintuplu* $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ *unde:*

1. $\mathcal{P}, \mathcal{A}, \mathcal{K}$ *sunt mulţimi finite, nevide, ale căror elemente se numesc* **mesaje, semnături** *şi respectiv* **chei***;*
2. *Există o aplicaţie biunivocă între* $\mathcal{K}$ *şi* $\mathcal{S} \times \mathcal{V}$*; anume, pentru fiecare* $K \in \mathcal{K}$ *există o pereche unică* (sig_K, ver_K) *unde* $sig_K : \mathcal{P} \to \mathcal{A}$, $ver_K : \mathcal{P} \times \mathcal{A} \to \{T, F\}$ *au proprietatea:*
$$\forall x \in \mathcal{P},\ \forall y \in \mathcal{A},\ ver_K(x, y) = T \iff y = sig_K(x)$$

Pentru fiecare $K \in \mathcal{K}$, funcţiile sig_K şi ver_K trebuie să fie calculabile în timp polinomial; ver_K este publică iar sig_K este secretă. Pentru $Oscar$, imitarea unei semnături a lui *Bob* pentru un mesaj x trebuie să fie imposibilă (din punct de vedere al complexităt ii calculului). Altfel spus, pentru un x dat, numai *Bob* este capabil să calculeze o semnătură y astfel ca $ver(x, y) = T$.

Bineînţeles, nici un protocol de semnătură nu este absolut sigur, deoarece $Oscar$ poate încerca – folosind funcţia publică de verificare ver – toate semnăturile y posibile ale unui mesaj x, până va găsi semnătura corectă.

Deci, dacă ar dispune de suficient timp, $Oscar$ poate totdeauna să contrafacă semnătura lui *Bob*.

Exemplul 1.1 *Un prim exemplu de semnătură este folosirea în acest scop a sistemului de criptare RSA. Se definesc*
$$\mathcal{P} = \mathcal{A} = Z_n, \quad \mathcal{K} = \{(n, p, q, a, b) \mid n = pq,\ p, q\ prime,\ ab \equiv 1\ (mod\ \phi(n))\}.$$
n şi b sunt publice, p, q, a sunt secrete.
Pentru $K = (n, p, q, a, b)$ se definesc:
$$sig_K(x) = x^a\ (mod\ n)$$
$$ver_K(x, y) = T \iff x \equiv y^b\ (mod\ n)$$

Aici Bob semnează un mesaj folosind cheia sa de decriptare din sistemul de criptare RSA; el este singurul capabil să genereze o semnătură corectă deoarece $d_K = sig_K$ este secretă. Funcţia de verificare utilizează funcţia de criptare e_K care este publică, deci oricine o poate verifica.

De remarcat că oricine poate genera o semnătură a lui *Bob* pentru un mesaj aleator x; *Oscar* poate alege un y şi calculează $x = e_k(y)$; atunci $y = sig_K(x)$.

Acest lucru poate fi prevenit folosind mesaje x cu suficient de multă redondanță (cu anumită semnificaţie). O altă modalitate de a evita acest atac este folosirea unor funcţii de dispersie (hash); vom studia această manieră în prelegerea următoare.

Să vedem cum pot fi combinate procedeele de semnătură şi criptare. Presupunem că *Alice* doreşte să trimită lui *Bob* un mesaj criptat şi semnat. Pentru un text clar x dat, *Alice* determină semnătura $y = sig_{Alice}(x)$, după care cifrează x şi y folosind cheia publică a lui *Bob* : $z = e_{Bob}((x, y))$.

Textul criptat z este transmis lui *Bob*. Acesta foloseşte cheia sa secretă d_{Bob} şi obţine (x, y). După aceasta, verifică semnătura lui *Alice* cu ajutorul cheii publice $ver_{Alice}(x, y)$.

Ce se întâmplă dacă Alice criptează înainte de a semna ? Ea va calcula $z = e_{Bob}(x)$, $y = sig_{Alice}(e_{Bob}(x))$ şi va trimite lui *Bob* mesajul (z, y). Acesta decriptează z, obţine x şi verifică y ca semnătură a lui z.

Pericolul constă în faptul că *Oscar* poate intercepta (z, y), înlocuieşte y cu propria sa semnătură y' şi transmite lui *Bob* mesajul (z, y').

Din acest motiv se recomandă folosirea semnăturii **înainte** de criptare.

1.3 Semnătura El Gamal

Fie p un număr prim (pentru care problema logaritmilor discreţi în Z_p este dificilă) şi $\alpha \in Z_p^* = Z_p \setminus \{0\}$ un element primitiv. Se ia:
$\mathcal{P} = Z_p^*, \quad \mathcal{A} = Z_p^* \times Z_{p-1}, \quad \mathcal{K} = \{(p, \alpha, a, \beta) \mid \beta = \alpha^a \ (mod\ p)\}$.
Valorile p, α, β sunt publice iar a este secret.
Pentru $K = (p, \alpha, a, \beta), k \in Z_{p-1}$ (secret) se defineşte:

$$sig_K(x, k) = (\gamma, \delta) \text{ unde}$$

$$\gamma = \alpha^k \ (mod\ p),\ \delta = (x - a\gamma)k^{-1} \ (mod\ p - 1).$$

Pentru $x, \gamma \in Z_p^*,\ \delta \in Z_{p-1}$ se defineşte

$$ver_K(x, \gamma, \delta) = T \iff \beta^\gamma \gamma^\delta \equiv \alpha^x \ (mod\ p)$$

Dacă semnătura este corectă, verificarea autentifică semnătura, deoarece:

$$\beta^\gamma \gamma^\delta \equiv \alpha^{a\gamma} \alpha^{k\delta} \ (mod\ p) \equiv \alpha^x \ (mod\ p)$$

(s-a folosit egalitatea $a\gamma + k\delta \equiv x \ (mod\ p - 1)$).

Protocolul de semnătură *El Gamal* a fost definit în 1985 ([4]). Ca o particularitate, el nu este determinist: pentru un mesaj dat pot exista mai multe semnături valide. Funcţia de verificare va trebui deci să accepte ca autentice toate aceste semnături.

Protocolul de semnătură *ElGamal* este descris pe pagina anterioară.

Bob calculează semnătura folosind cheia sa secretă a şi o valoare aleatoare secretă k (generată numai pentru semnarea mesajului x). Verificarea se realizează cu ajutorul cheii publice.

Exemplul 1.2 *: Să luăm $p = 467,\ \alpha = 2,\ a = 127$. Avem*

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

Dacă Bob doreşte să semneze mesajul $x = 100$ alegând valoarea $k = 213$ (de remarcat că $(213, 466) = 1$ şi $213^{-1} \ (mod\ 466) = 431$), va obţine

$$\gamma = 2^{213} \ (mod\ 467) = 29 \text{ şi } \delta = (100 - 127 * 29) * 431 \ (mod\ 466) = 51.$$

Pentru a verifica semnătura, calculăm

$$132^{29} * 29^{51} \equiv 189 \ (mod\ 467) \text{ şi } 2^{100} \equiv 189 \ (mod\ 467).$$

Semnătura este deci validă.

Să studiem securitatea protocolului de semnătură *El Gamal*.
Vom presupune că *Oscar* doreşte să falsifice semnătura pe mesajul x fără să ştie a.

- Dacă *Oscar* alege valoarea γ şi încearcă să găsească δ corespunzător, el va trebui să calculeze logaritmul discret $log_\gamma \alpha^x \beta^{-\gamma}$. Dacă strategia sa este inversă: să aleagă întâi δ şi să caute apoi γ, el va trebui să rezolve ecuaţia

  $$\beta^\gamma \gamma^\delta \equiv \alpha^x \ (mod\ p) \quad \text{de necunoscută } \gamma.$$

  Nu se cunoaşte încă o metodă pentru rezolvarea unei astfel de probleme.

- Dacă *Oscar* alege aleator şi pe δ şi caută să obţină x, el va ajunge din nou la problema logaritmului discret, adică la calculul $log_\alpha \beta^\gamma \gamma^\delta$.

Oscar poate totuşi să semneze un mesaj aleator x în felul următor:

Fie numerele întregi i, j $(0 \le i \le p-2,\ 0 \le j \le p-2,\ (j, p-1) = 1)$.

Se efectuează calculele:

$$\gamma = \alpha^i \beta^j \quad (mod\ p); \quad \delta = -\gamma j^{-1} \ (mod\ p-1); \quad x = -\gamma i j^{-1} \ (mod\ p-1)$$

(deoarece calculele sunt făcute modulo $p-1$, există j^{-1}).

(γ, δ) este o semnătură validă pentru x. Într-adevăr, se verifică

$$\beta^\gamma \gamma^\delta \equiv \beta^{\alpha^i\beta^j} (\alpha^i \beta^j)^{-\alpha^i\beta^j j^{-1}} \equiv \beta^{\alpha^i\beta^j} \alpha^{-ij^{-1}\alpha^i\beta^j} \beta^{-\alpha^i\beta^j} \equiv \alpha^{-ij^{-1}\alpha^i\beta^j} \equiv \alpha^{-\gamma i j^{-1}} \equiv \alpha^x$$

(toate calculele sunt făcute modulo p).

Să exemplificăm acest atac:

Exemplul 1.3 *Fie din nou $p = 467,\ \alpha = 2,\ \beta = 132$. Să presupunem că Oscar alege $i = 99,\ j = 179$ (deci $j^{-1} = 151\ (mod\ p-1)$); Oscar calculează:*

$$\gamma = 2^{99} 132^{179} = 117 \ (mod\ 467)$$
$$\delta = -117 \times 151 = 41 \ (mod\ 466)$$
$$x = 99 \times 41 = 331 \ (mod\ 466)$$

Deci $(117, 41)$ este o semnătură a mesajului 331, ceea ce se poate verifica imediat, calculând

$$132^{117} 117^{41} \equiv 303 \ (mod\ 467) \text{ şi } 2^{331} \equiv 303 \ (mod\ 467).$$

Semnătura este deci validă.

Să mai arătăm o modalitate prin care *Oscar* poate utiliza un mesaj semnat anterior de *Bob*. Să presupunem că (γ, δ) este o semnătură valida a lui x. *Oscar* poate semna atunci alte tipuri de mesaje:

Fie h, i, j numere întregi din intervalul $[0, p-2]$ cu $(h\gamma - j\delta, p-1) = 1$.

Calculăm:

$$l = \gamma^h\alpha^i\beta^j\ (mod\, p), \quad y = \delta l(h\gamma - j\delta)^{-1}\ (mod\, p-1),$$
$$x' = l(hx + i\delta)(h\gamma - j\delta)^{-1}\ (mod\, p-1)$$

unde $(h\gamma - j\delta)^{-1}$ este calculat modulo $p-1$. Se poate atunci verifica direct $\beta^l l^y \equiv \alpha^{x'}\ (mod\, p)$. Deci (l, y) este o semnătură validă a lui x'.

Aceste două strategii construiesc semnături valide, dar se pare că nu este posibil ca cineva să contrafacă semnătura unui mesaj ales de el, fără să rezolve o problemă de logaritmi discreţi. Din acest motiv se consideră că nu există slăbiciuni în protocolul de semnătură *El Gamal*.

Să arătăm în final două maniere de a sparge acest protocol de semnătură, atunci când este aplicat neglijent.

- Dacă întregul aleator k este cunoscut, se determină imediat
$$a = (x - k\delta)\gamma^{-1}\ mod\ (p-1)$$
Din acest moment, *Oscar*, ştiind a, poate calcula semnăturile la fel ca *Bob*.

- Dacă se utilizează acelaşi k pentru mai multe mesaje. Aceasta îi permite de asemenea lui *Oscar* să determine a. El va proceda astfel:

  Fie (γ, δ_i) semnăturile mesajelor $x_i,\ i = 1, 2$. Avem:
  $\beta^\gamma\gamma^{\delta_i} \equiv \alpha^{x_i}\ (mod\, p),\ i = 1, 2,$ deci $\alpha^{x_2 - x_1} \equiv \gamma^{\delta_2 - \delta_1}\ (mod\, p)$.

  Înlocuind $\gamma = \alpha^k$ se obţine ecuaţia de necunoscută k:
$$\alpha^{x_2 - x_1} \equiv \alpha^{k(\delta_2 - \delta_1)}\ (mod\, p),$$
  care este echivalentă cu $x_2 - x_1 \equiv k(\delta_2 - \delta_1)\ (mod\, p-1)$.

  Dacă se ia $d = (\delta_2 - \delta_1, p-1)$, din $d \mid (p-1)$ şi $d \mid (\delta_2 - \delta_1)$ rezultă $d \mid (x_2 - x_1)$.

  Dacă notăm:
$$x' = \frac{x_2 - x_1}{d} \qquad \delta' = \frac{\delta_2 - \delta_1}{d} \qquad p' = \frac{p-1}{d}$$
  ecuaţia devine $x' \equiv k\delta'(mod\, p)$. Cum $(\delta', p') = 1$, se poate determina $\epsilon = (\delta')^{-1}\ (mod\, p')$.

  Valoarea lui k verifică deci relaţia $k \equiv x'\ (mod\, p)$, ceea ce conduce la determinarea a d candidaţi la valoarea lui k, daţi de relaţia $k = x' + ip'(mod\, p),\ i = 0, \ldots, d-1$. Din aceste d valori posibile, soluţia se determină testând relaţia $\gamma \equiv \alpha^k\ (mod\, p)$

1.4 Variante ale protocolului de semnătură *ElGamal*

În general, un mesaj este criptat şi decriptat o singură dată, fiind necesară doar securitatea sistemului de criptare. În schimb, un document semnat – cum ar fi un contract – are o valoare juridică, şi este posibil ca autenticitatea sa să fie verificată chiar şi după mai mulţi ani. Este deci important să existe criterii de securitate mai severe pentru semnătura electronică decât pentru criptare. Cum siguranţa protocolului de semnătură *ElGamal* este echivalentă cu complexitatea problemei logaritmilor discreţi, este necesar să se folosească un modul p cât mai mare. Un p de 1024 biţi conduce însă la o semnătură *ElGamal* de 2048 biţi, ceea ce o elimină din multe aplicaţii (cum ar fi exemplu smart-cardurile).

1.4.1 Standard de semnătură electronică

Standardul de semnătură electronică (**DSS** – de la **D**_igital_ **S**_ignature_ **S**_tandard_) este o variantă a protocolului de semnătură _ElGamal_, cu proprietatea că reduce substanţial lungimea semnăturii. Protocolul de semnătură DSS este următorul:

> Fie p un număr prim de 512 biţi, q un factor de 160 biţi ai lui $p-1$ şi $\alpha \in Z_p^*$ o rădăcină primitivă de ordin q a unităt ii.
> Fie $\mathcal{P} = Z_p^*, \mathcal{A} = Z_q \times Z_q$ şi $\mathcal{K} = \{(p,q,\alpha,a,\beta) \mid \beta \equiv \alpha^a\ (mod\ p)\}$.
> Valorile p,q,α,β sunt publice, iar a este secretă.
> Pentru $K = (p,q,\alpha,a,\beta)$ şi pentru un număr (secret) k $(1 \le k \le q-1)$ se definesc:
>
> - $sig_K(x,k) = (\gamma,\delta)$ unde
> $$\gamma = (\alpha^k\ mod\ p)\ mod\ q \quad \delta = (x + a\gamma)k^{-1}\ mod\ q$$
> - Pentru $x \in Z_p^*,\ \gamma,\delta \in Z_q$ funcţia de verificare este definită
> $$ver_K(x,\gamma,\delta) = T \Longleftrightarrow (\alpha^{e_1}\beta^{e_2}\ mod\ p)\ mod\ q\ = \gamma$$
> unde $e_1 = x\delta^{-1}\ (mod\ q) \qquad e_2 = \gamma\delta^{-1}\ (mod\ q)$

Diferenţe între protocoalele de semnătură _El Gamal_ **şi** DSS:

a. DSS se distinge în primul rând de _El Gamal_ prin faptul că asigură o semnătură de 320 biţi pe un mesaj de 160 biţi, lucrând în Z_p cu p de 512 biţi. Aceasta permite lucrul într-un corp cu circa 2^{160} elemente. Ipoteza este aceea că în această bază, calculul logaritmilor discreţi este foarte dificil.

b. În definirea lui δ semnul $-$ s-a schimbat în $+$. Aceasta schimbă ecuaţia de verificare în: $\alpha^x\beta^\gamma \equiv \gamma^\delta\ mod\ p.$
Dacă $(x + \alpha\gamma, p-1) = 1$, atunci există $\delta^{-1}\ mod\ (p-1)$ şi această ecuaţie se poate scrie $\alpha^{x\delta^{-1}}\beta^{\gamma\delta^{-1}} \equiv \gamma\ (mod\ p)$.

c. (idee a lui _Schnorr_). Să presupunem că q este un număr de 160 biţi astfel încât $q \mid (p-1)$ şi $\alpha \in Z_p^*$ este o rădăcină primitivă de ordinul q a unitătii modulo p (pentru a găsi un astfel de număr, se ia o rădăcină primitivă $\alpha_0 \in Z_p$ şi se construieşte $\alpha = \alpha_0^{(p-1)/q}\ (mod\ p)$). În acest fel, β şi γ sunt de asemenea rădăcini de ordinul q ale unităţii. Deci exponenţii lui α,β,γ se pot reduce modulo q, fără a modifica ecuaţia de verificare de mai sus.

   Algoritmul de Schnorr foloseşte şi o funcţie de dispersie $h: \{0,1\}^* \to Z_q$ (a se vedea prelegerea următoare), pe baza căreia defineşte componentele semnăturii astfel:
$$\gamma = h(x\|\alpha^k), \qquad \delta = k + \alpha\gamma\ (mod\ q)$$
   Pentru $x \in \{0,1\}^*$ şi $\gamma,\delta \in Z_q$, procedura de verificare este
$$ver_K(x,(\gamma,\delta)) = T \Longleftrightarrow h(x\|\alpha^\delta\beta^{-\gamma}) = \gamma$$
   Relaţia de verificare este adevărată deoarece se obţine imediat $\alpha^\delta\beta^{-\gamma} = \alpha^k\ (mod\ p)$

Exemplul 1.4 _Fie_ $q = 101,\ p = 78q + 1 = 7879.$ _3 este rădăcină primitivă în_ Z_{7879}_, deci se poate lua_ $\alpha = 3^{78}\ (mod\ 7879) = 170$.

Dacă alegem de exemplu $a = 75$_, obţinem_ $\beta = \alpha^a\ (mod\ 7879) = 4567$.

Să presupunem că Bob doreşte să semneze mesajul $x = 1234$ *şi ia* $k = 50$ *(deci* $k^{-1}\ (mod\ 101) = 99$*); el va avea:*

$$\gamma = (170^{50}\ mod\ 7879)\ mod\ 101 = 2518\ (mod\ 101) = 94$$
$$\delta = (1234 + 75 \times 94) \times 99\ (mod\ 101) = 97$$

Semnătura $(94, 97)$ *a mesajului* 1234 *se verifică prin calcul:*

$\delta^{-1} = 97^{-1}\ (mod\ 1010 = 25,\ e_1 = 1234 \times 25\ (mod\ 1010 = 45,\ e_2 = 94 \times 25\ (mod\ 101) = 27$

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

Semnătura este deci validă.

În varianta Schnorr trebuie calculată valoarea $h(1234\|2518)$ *unde* h *este o funcţie de dispersie, iar* 1234 *şi* 2518 *sunt reprezentate în binar. Pentru a nu intra în detalii, să presupunem că* $h(1234\|2518) = 96$*. Atunci*

$$\delta = 50 + 75 \times 96\ (mod\ 101) = 79$$

şi semnătura este $(96, 79)$*.*

Ea este verificată calculând $170^{79}4567^{-96}\ (mod\ 7879) = 2518$ *şi verificând* $h(1234\|2518) = 96$*.*

O altă variantă a protocolului DSS este protocolul de semnătură DSA (*Digital Signature Algorithm*). Acesta a fost propus în 1991 şi adoptat ca standard de semnătură la 1 decembrie 1994 (după publicarea sa în Registrul Federal la 19 mai 1994). Singura modificare faţă de DSS constă în înlocuirea mesajului x (din calculul lui δ şi al lui e_1) cu $SHA-1(x)$, unde $SHA-1$ este o funcţie de dispersie standard.

Exemplul 1.5 *Să reluăm valorile lui* $p, q, \alpha, a, \beta, k$ *din Exemplul 1.4 şi să presupunem că Alice vrea să semneze amprenta* $SHA-1(x) = 22$*. Ea va calcula*

$$k^{-1}\ (mod\ 1010 = 50^{-1}\ (mod\ 1010 = 99,$$
$$\gamma = (170^{50}\ (mod\ 7879))\ (mod\ 1010 = 2518\ (mod\ 101) = 94 \text{ şi}$$
$$\delta = (22 + 75 \times 94) \times 99\ (mod\ 101) = 97.$$

Semnătura $(94, 97)$ *a amprentei* 22 *este verificată efectuând calculele;*

$$\delta^{-1} = 97^{-1}\ (mod\ 1010 = 25,$$
$$e_1 = 22 \times 25\ (mod\ 101) = 45, \qquad e_2 = 94 \times 25\ (mod\ 101) = 27$$

şi verificând că $170^{45}4567^{27}\ (mod\ 7879)\ (mod\ 101) = 2518\ (mod\ 101) = 94$*.*

Când DSS a fost propus în 1991, a avut mai multe critici. Astfel:

- Mulţi s-au arătat nemulţumiţi de impunerea mărimii de 512 biţi pentru modul; o mărime variabilă care să fie aleasă de beneficiari în funcţie de necesităt i ar fi fost mai convenabilă. Ca răspuns, $NIST$ a schimbat descrierea standardului pentru a permite alegerea ca modul a oricărui număr divizibil cu 64 având între 512 şi 1024 biţi. În octombrie 2001 NIST revine şi recomandă alegerea pentru p a unui număr prim de 1024 biţi.

- Semnăturile pot fi mult mai uşor generate decât verificate. Pentru comparaţie, în sistemul RSA un exponent mic de decriptare poate conduce la un protocol de verificare mult mai rapid decât semnătura. Obiecţia este ridicată din considerente practice, anume:
  - Un mesaj este semnat o singură dată, dar poate fi verificat de foarte multe ori de-a lungul timpului.
  - Pe ce tipuri de calculatoare sunt efectuate aceste protocoale de semnătură şi/sau verificare ? Cea mai mare parte a aplicaţiilor folosesc calculatoare de birou, cu resurse limitate, care comunică cu sisteme puternice de calcul. Trebuie dezvoltat deci un protocol care să oblige calculatorul de birou la cât mai puţine calcule.

Răspunsul dat de $NIST$ la această obiecţie a fost că este foarte puţin important ce calcul este mai simplu, având în vedere sistemele actuale de calcul, tot mai performante.

1.4.2 Protocolul de semnătură $ECDSA$

În 2000 protocolul $ECDSA$ (*Elliptic Curve Digital Signature Algorithm*) a fost aprobat sub numele $FIPS$ $186-2$. Şi el este o variantă a protocolului *ElGamal*, construit pentru funcţii eliptice. Să prezentăm o descriere a sa:

> Fie p un număr prim sau o putere a lui 2 şi E o curbă eliptică peste Z_p. Fie A un punct din E de ordin q (număr prim) astfel ca Problema Logaritmului discret pentru A să fie dificilă. Fie
> $\mathcal{P}= \{0,1\}^*,\ \mathcal{A}= Z_1^* \times Z_q^*, \quad \mathcal{K}= \{(p,q,E,A,m,B) \mid B = mA\}$,
> unde $m \in Z_{q-1}$. Valorile p, q, E, A, B sunt publice, m este cheia secretă.
> Pentru $K = (p,q,E,A,m,B)$ şi $k \in Z_q$ ales aleator, definim
> $$sig_K(x,k) = (r,s)$$
> unde
> $$kA = (u,v), \quad r = u\ (mod\ q), \quad s = k^{-1}(SHA_1(x) + mr)\ (mod\ q).$$
> (dacă $r \cdot s = 0$ se alege altă valoare aleatoare pentru k).
> Pentru $x \in \{0,1\}^*,\ r,s \in Z_q^*$ verificarea este definită prin
> $$w = s^{-1}\ (mod\ q), \qquad i = w \cdot SHA_1(x)\ (mod\ q),$$
> $$j = w \cdot r\ (mod\ q), \qquad (u,v) = iA + jB \qquad \text{şi}$$
> $$ver_K(x,(r,s)) = T \Longleftrightarrow u\ (mod\ q) = r$$

Exemplul 1.6 *Să considerăm curba eliptică* $y^2 = x^3 + x + 6$ *definită peste* Z_{11}. *Alegem parametrii protocolului de semnătură astfel:* $p = 11,\ q = 13,\ A = (2,7),\ m = 7$ *şi* $B = (7,2)$. *(a se vedea şi exemplul din prelegerea anterioară).*

Să presupunem că avem un mesaj x *cu* $SHA_1(x) = 4$ *şi Alice vrea să-l semneze folosind valoarea aleatoare* $k = 3$. *Ea va calcula*

$(u,v) = 3 \cdot (2,7) = (8,3), \qquad r = u\ (mod\ 13) = 8$ *şi* $s = 3^{-1} \cdot (4 + 7 \cdot 8)\ (mod\ 13) = 7.$

Deci semnătura este $(8,7)$.

Bob verifică această semnătură efectuând următoarele calcule:

$w = 7^{-1}\ (mod\ 13) = 2, \quad i = 2 \cdot 4\ (mod\ 13) = 8, \quad j = 2 \cdot 8\ (mod\ 13) = 3,$

$(u,v) = 8A + 3B = (8,3)$ *şi* $u\ (mod\ 13) = 8 = r.$

Deci semnătura este validă.

1.5 Protocoale de semnătură "One-time"

Ideea acestor protocoale se bazează pe faptul că funcţia care asigură semnătura este folosită pentru a semna un singur document, după care ea este abandonată (cu toate că poate fi verificată de un număr arbitrar de ori).

Poate cel mai cunoscut astfel de procedeu este **Protocolul de semnătură Lamport**:

Fie $k > 0$ un număr întreg şi $\mathcal{P} = \{0,1\}^k$. Dacă $f : Y \to Z$ este o funcţie neinversabilă, se alege $\mathcal{A} = Y^k$. Se selectează aleator $y_{i,j} \in Y,\ 1 \le i \le k,\ j = 0, 1$ şi fie $z_{i,j} = f(y_{i,j})$.
Cheia K este lista celor $2k$ valori y (secrete) şi a celor $2k$ valori z (publice).
Pentru $K = \{(y_{i,j}, z_{i,j}) \mid 1 \le i \le k,\ j = 0, 1\}$ se defineşte:

$$sig_K(x_1, \ldots, x_k) = (y_{1,x_1}, \ldots, y_{k,x_k})$$

şi

$$ver_K(x_1, \ldots, x_k, a_1, \ldots, a_k) = T \iff f(a_i) = z_{i,x_i}, 1 \le i \le k.$$

Conform acestui protocol, mesajul care trebuie semnat este un şir binar de lungime k. Fiecare bit, de valoare j $(j = 0, 1)$ este semnat prin $z_{i,j}$, unde fiecare $z_{i,j}$ este imaginea printr-o funcţie neinversabilă a unui $y_{i,j}$.

Verificarea constă în aplicarea lui f şi compararea rezultatului cu cheia publică.

Exemplul 1.7 *Fie* 7879 *un număr prim, iar* $3 \in Z_{7879}$ *un element primitiv. Se defineşte*

$$f(x) = 3^x \ mod\ 7879.$$

Dacă Bob doreşte să semneze un mesaj de trei biţi, el alege (secret) şase numere aleatoare

$$y_{1,0} = 5831 \quad y_{2,0} = 803 \quad y_{3,0} = 4285$$
$$y_{1,1} = 735 \quad y_{2,1} = 2467 \quad y_{3,1} = 6449$$

Calculează apoi imaginea lor prin funcţia f*:*

$$z_{1,0} = 2009 \quad z_{2,0} = 4672 \quad z_{3,0} = 268$$
$$z_{1,1} = 3810 \quad z_{2,1} = 4721 \quad z_{3,1} = 5731$$

Aceste numere z *sunt publice.*

Să presupunem că Bob vrea să semneze mesajul $x = (1, 1, 0)$. *Semnătura lui este*

$$(y_{1,1}, y_{2,1}, y_{3,0}) = (735, 2467, 4285)$$

Pentru a verifica semnătura, este suficient să se constate că:

$$3^{735} = 3810; \quad 3^{2467} = 4721; \quad 3^{4285} = 268,$$

toate calculele fiind realizate modulo 7879.

Oscar nu poate imita semnătura, deoarece f nu are inversă.

În plus, protocolul de semnătură nu poate fi utilizat decât pentru un singur mesaj: dacă dispune de două mesaje cu aceeaşi semnătură, *Oscar* poate imita semnătura unui nou mesaj (diferit de cele două).

De exemplu, dacă mesajele $(0, 1, 1)$ şi $(1, 0, 1)$ sunt semnate prin procedeul de sus cu $(y_{1,0}, y_{2,1}, y_{3,1})$ respectiv $(y_{1,1}, y_{2,0}, y_{3,1})$, se pot imediat semna mesaje cum ar fi $(1, 1, 1)$ sau $(0, 0, 1)$.

Deşi foarte simplu şi elegant, acest protocol nu este practic din cauza dimensiunii mari a semnăturii. Reluând exemplul de mai sus, o implementare sigură necesită un modul p de 512 biţi. Aceasta înseamnă că fiecare bit al mesajului are o semnătură de 512 biţi; avem deci o semnătură de 512 ori mai lungă decât mesajul !

De aceea a apărut o simplificare, care reduce lungimea semnăturii fără a diminua securitatea ei. Numit **Bos-Chaum**, acest protocol este definit astfel:

Fie $k > 0$ un număr întreg şi $\mathcal{P} = \{0,1\}^k$. Dacă n este un număr întreg cu proprietatea $2^k \le C_{2n}^n$, fie B mulţimea numerelor întregi din intervalul $[1, 2n]$ şi
$$\phi : \mathcal{P} \to \mathcal{B}$$
o funcţie injectivă în mulţimea $\mathcal{B}$ a părţilor lui B de n elemente.
Dacă $f : Y \to Z$ este o funcţie neinversabilă, fie $\mathcal{A} = Y^n$.
Se aleg aleator valorile $y_i \in Y,\ 1 \le i \le 2n$ şi fie $z_i = f(y_i)$.
Cheia K este lista celor $2n$ valori y (secrete) şi a celor $2n$ valori z (publice).
Pentru $K = \{(y_i, z_i) \mid 1 \le i \le 2n\}$, se definesc
$$sig_K(x_1, \ldots, x_k) = \{y_j \mid j \in \phi(x_1, \ldots x_k)\}$$
şi
$$ver_K(x_1, \ldots, x_k, a_1, \ldots, a_n) = T \Longleftrightarrow \{f(a_i) \mid 1 \le i \le n\} = \{z_j \mid j \in \phi(x_1, \ldots, x_k)\}$$

Avantajul protocolului **Bos-Chaum** este acela că scurtează semnătura. De exemplu, să presupunem că vrem să semnăm un mesaj de şase biţi ($k = 6$); cum $2^6 = 64$ şi $C_8^4 = 70$, putem lua $n = 4$. Aceasta permite semnarea mesajului cu numai patru valori y (în loc de şase la **Lamport**). De asemenea, şi cheia este mai scurtă, cu numai opt valori z faţă de 12 la semnătura Lamport.

Protocolul de semnătură Bos-Chaum necesită o funcţie injectivă ϕ care asociază fiecărei secvenţe de k biţi $x = (x_1, \ldots, x_k)$ o submulţime de n elemente. Un exemplu de algoritm simplu care realizează o astfel de asociere este:

$x \leftarrow \sum_{i=1}^{k} x_i 2^{i-1}$
$\phi(x) \leftarrow \emptyset$
$t \leftarrow 2n$
$e \leftarrow n$
while $t > 0$ **do**
  $t \leftarrow t - 1$
  if $x > C_t^e$ **then**
    $x \leftarrow x - C_t^e$
    $e \leftarrow e - 1$
    $\phi(x) \leftarrow \phi(x) \cup \{t + 1\}$.
      enddo

Dacă vrem să dăm o estimare generală a valorii lui n din protocolul Bos-Chaum, plecăm de la inegalitatea $2^k \le C_{2n}^n$ în care se evaluează $C_{2n}^n = \dfrac{(2n)!}{(n!)^2}$ cu formula lui Stirling, obţinându-se $2^{2n}/\sqrt{\pi n}$. După simplificări şi logaritmare în baza 2 se ajunge la relaţia:
$$k \le 2n - \frac{log_2 n\pi}{2}$$
Asimptotic, n este de ordinul lui $k/2$, deci protocolul de semnătură Bos-Chaum reduce mărimea semnăturii lui Lamport cu aproximativ 50%.

1.6 Semnături incontestabile

Semnăturile incontestabile au fost introduse de Chaum şi van Antwerpen în 1989. Ele prezintă câteva caracteristici. Astfel:

- Semnătura nu poate fi validată fără aportul semnatarului *Bob*. Aceasta îl protejează pe *Bob* de difuzarea fără consimtământ a unui document pe care se pretinde că l-ar fi semnat. Validarea se face urmând un protocol de întrebări şi răspunsuri.

- Pentru a evita ca *Bob* să-şi nege propria semnătură, există un *protocol de dezminţire* pe care *Bob* trebuie să-l urmeze pentru a arăta că o semnătură este falsă.

  Refuzul de a folosi acest protocol este o confirmare a autenticităţii semnăturii.

Deci, un protocol de semnătură incontestabilă este format dintr-o funcţie de semnă-tură, un protocol de verificare şi o procedură de dezminţire.

Algoritmul **Chaum-van Antwerpen** este:

Fie $p = 2q + 1$ un număr prim cu proprietatea că q este prim şi $\alpha \in Z_p^*$ un element de ordin q. Pentru $1 \leq a \leq q - 1$, se defineşte $\beta \equiv \alpha^a \ mod\, p$.
Fie G subgrupul de ordin q al lui Z_p generat de α.
Se definesc $\mathcal{P} = \mathcal{A} = G,\ \mathcal{K} = \{(p, \alpha, a, \beta) \mid \beta \equiv \alpha^a\ (mod\, p)\}$.
Valorile p, α, β sunt publice iar a este secretă.
Pentru $K = (p, \alpha, a, \beta), x \in G$ se defineşte $y = sig_K(x) = x^a \ mod\, p$.
Pentru $x, y \in G$, protocolul de verificare se efectuează astfel:

1. *Alice* alege aleator numerele $e_1, e_2 \in Z_q^*$;
2. *Alice* calculează $c = y^{e_1}\beta^{e_2} \ mod\, p$ şi-l trimite lui *Bob*;
3. *Bob* calculează $d = c^{a^{-1} \ mod\, q} \ mod\, p$ şi-l trimite lui *Alice*;
4. *Alice* admite autenticitatea lui y dacă şi numai dacă $d \equiv x^{e_1}\alpha^{e_2} \ mod\, p$.

A. Să explicăm întâi rolul lui p şi q în acest protocol. Calculele sunt efectuate în Z_p. Este necesar totuşi ca anumite calcule să fie făcute într-un subgrup al său de ordin prim (notat cu G). În particular este nevoie să calculăm inverse modulo q (ceea ce justifică de ce $q = card(G)$ trebuie să fie prim). Alegând $p = 2q + 1$ cu q prim, se asigură acest deziderat şi - în plus - dimensiunea lui G este maximă, lucru de dorit deoarece mesajele de semnat sunt elemente din G.

Să arătăm întâi cum admite *Alice* autenticitatea semnăturilor valide. În calculul de mai jos, exponenţii sunt reduşi modulo q.
$d \equiv c^{a^{-1}}\ (mod\, p) \equiv y^{e_1 a^{-1}}\beta^{e_2 a^{-1}}\ (mod\, p)$.
Cum $\beta \equiv \alpha^a\ (mod\, p),$ avem $\beta^{a^{-1}} \equiv \alpha\ (mod\, p)$.
De asemenea, din $y = x^a\ (mod\, p)$ rezultă $y^{a^{-1}} \equiv x\ (mod\, p)$.
Se ajunge deci la $d \equiv x^{e_1}\alpha^{e_2}\ (mod\, p)$.

Exemplul 1.8 *Fie* $p = 467$. 2 *este o rădăcină primitivă, deci* $2^2 = 4$ *este un generator al lui* G, *grupul reziduurilor patratice modulo* 467. *Vom lua deci* $\alpha = 4$.

Să presupunem $a = 101$*; avem* $\beta = \alpha^a \ mod\, 467 = 449$.

Bob semnează deci mesajul $x = 119$ *cu* $y = 119^{101} \ mod\, 467 = 129$.

Să presupunem că Alice vrea să autentifice semnătura y *şi că alege pentru asta* $e_1 = 38,\ e_2 = 397$. *Ea calculează* $c = 13$, *la care Bob răspunde cu* $d = 9$. *Alice verifică atunci relaţia* $119^{38}4^{397} \equiv 9\ (mod\, 467)$.

Semnătura este acceptată ca autentică.

B. Să arătăm acum că *Alice* nu poate accepta o semnătură falsă drept autentică decât cu o probabilitate neglijabilă.

Teorema 1.1 *Dacă $y \not\equiv x^a$ (mod p) atunci Alice admite pe y ca semnătură autentică a lui x cu probabilitate $1/q$.*

Demonstraţie: Se observă că orice întrebare c corespunde la exact q perechi (e_1, e_2) posibile (deoarece y şi β sunt elemente ale grupului G de ordin q prim şi în definiţia lui c, fiecare e_1 determină un e_2 unic). Când *Bob* primeşte c, el nu ştie ce pereche (e_1, e_2) a fost folosită pentru a-l construi. Vom arăta că dacă $y \not\equiv x^a \ (mod\ p)$, orice răspuns d nu poate fi consistent decât cu o singură pereche (e_1, e_2).

Deoarece α generează G, orice element din G se scrie ca o putere (unică modulo q) a lui α. Deci $c = \alpha^i, d = \alpha^j, x = \alpha^k, y = \alpha^m$ cu $i, j, k, m \in Z_q$ şi operaţiile aritmetice efectuate modulo p. Să considerăm sistemul:

$$c \equiv y^{e_1}\beta^{e_2} \ (mod\ p) \quad d \equiv x^{e_1}\alpha^{e_2} \ (mod\ p).$$

El este echivalent cu

$$i \equiv me_1 + ae_2 \ (mod\ q) \quad j \equiv ke_1 + e_2 \ (mod\ q).$$

Cum prin ipoteză $y \not\equiv x^a \ (mod\ p)$, rezultă $m \not\equiv ak \ (mod\ q)$.

Astfel, matricea sistemului modulo q admite un determinant nenul, deci sistemul are soluţie unică. Altfel spus, pentru orice $d \in G$, nu există răspuns corect la întrebarea c decât pentru un singur cuplu (e_1, e_2). Deci probabilitatea ca *Bob* să răspundă corect lui *Alice* în condiţiile teoremei este $1/q$. □

C. Să construim acum procedura de dezminţire. Ea foloseşte de două ori protocolul de verificare. Algoritmul este:

1. *Alice* alege aleator $e_1, e_2 \in Z_q^*$;
2. *Alice* calculează $c = y^{e_1}\beta^{e_2} \ (mod\ p)$ şi-l trimite lui *Bob*;
3. *Bob* calculează $d = c^{a^{-1} \ mod\ q} \ (mod\ p)$ şi-l trimite lui *Alice*;
4. *Alice* verifică $d \not\equiv x^{e_1}\alpha^{e_2} \ (mod\ p)$;
5. *Alice* alege aleator $f_1, f_2 \in Z_q^*$;
6. *Alice* calculează $C = y^{f_1}\beta^{f_2} \ (mod\ p)$ şi-l trimite lui *Bob*;
7. *Bob* calculează $D = C^{a^{-1} \ mod\ q} \ (mod\ p)$ şi-l trimite lui *Alice*;
8. *Alice* verifică $D \not\equiv x^{f_1}\alpha^{f_2} \ (mod\ p)$;
9. *Alice* admite că y este fals dacă şi numai dacă
$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \ (mod\ p)$$

Paşii $1-4$ şi $5-8$ corespund protocolului de verificare. Pasul 9 este *validarea consistenţei răspunsului*, care permite lui *Alice* să determine dacă *Bob* a calculat bine răspunsurile sale conform protocolului.

Exemplul 1.9 *Să luăm parametrii din exemplul anterior:* $p = 467,\ \alpha = 4,$
$a = 101,\ \beta = 449$. *Fie mesajul* $x = 286$ *cu semnătura (greşită)* $y = 83$.

Bob doreşte să dezmintă această semnătură.

Fie $e_1 = 45,\ e_2 = 237$ *primele valori alese de Alice. Ea calculează* $c = 305$ *şi Bob răspunde cu* $d = 109$. *Alice calculează atunci*

$$286^{45}4^{237}\ (mod\ 467) = 149.$$

Deoarece $149 \neq 109$, *Alice trece la pasul* 5 *al protocolului. Să presupunem că ea alege acum* $f_1 = 125,\ f_2 = 9$ *şi calculează* $C = 270$ *la care Bob răspunde cu* $D = 68$. *Alice calculează acum*

$$286^{125}4^{9}\ (mod\ 467) = 25.$$

Cum $25 \neq 68$, *Alice trece la pasul* 9 *şi efectuează testul de consistentă:*

$(109 \times 4^{-237})^{125} \equiv 188\ (mod\ 467) \qquad (68 \times 4^{-9})^{45} \equiv 188\ (mod\ 467)$

Acum Alice este convinsă că semnătura nu este valabilă.

Pentru final, mai trebuiesc arătate două elemente:

- *Bob* poate să o convingă pe *Alice* să invalideze o semnătură incorectă.
- *Bob* nu poate să o convingă pe *Alice* să invalideze o semnătură corectă decât cu probabilitate neglijabilă.

Teorema 1.2 *Dacă* $y \not\equiv x^a\ (mod\ p)$ *şi dacă Alice şi Bob urmează corect protocolul de dezminţire, atunci*

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1}\ (mod\ p).$$

Demonstraţie: Utilizând faptul că $d \equiv c^{a^{-1}}\ (mod\ p)$ şi $c \equiv y^{e_1}\beta^{e_2}\ (mod\ p)$, avem:

$(d\alpha^{-e_2})^{f_1} \equiv ((y^{e_1}\beta^{e_2})^{a^{-1}}\alpha^{-e_2})^{f_1}\ (mod\ p) \equiv y^{e_1f_1}\beta^{e_2a^{-1}f_1}\alpha^{-e_2f_1}\ (mod\ p)$
$\equiv y^{e_1f_1}\alpha^{e_2f_1}\alpha^{-e_2f_1}\ (mod\ p) \equiv y^{e_1f_1}\ (mod\ p)$.

Un calcul similar folosind $D \equiv C^{a^{-1}}\ (mod\ p),\ C \equiv y^{f_1}\beta^{f_2}\ (mod\ p)$ şi $\beta \equiv \alpha^a\ (mod\ p)$ arată că

$$(D\alpha^{-f_2})^{e_1} \equiv y^{e_1f_1}\ (mod\ p)$$

deci testul de consistentă de la pasul 9 reuşeşte. □

Să studiem acum cazul când *Bob* încearcă să dezmintă o semnătură validă. În acest caz *Bob* nu va urma protocolul, şi va construi d şi D fără să respecte procedura.

Teorema 1.3 *Să presupunem că* $y \equiv x^a\ (mod\ p)$ *şi Alice urmează procedura de dezminţire. Dacă* $d \not\equiv x^{e_1}\alpha^{e_2}\ (mod\ p)$ *şi* $D \not\equiv x^{f_1}\alpha^{f_2}\ (mod\ p)$ *atunci probabilitatea ca* $(d\alpha^{-e_2})^{f_1} \not\equiv (D\alpha^{-f_2})^{e_1}\ (mod\ p)$ *este* $1 - \dfrac{1}{q}$.

Demonstraţie: Să presupunem că avem (conform ipotezei):

$y \equiv x^a \quad d \not\equiv x^{e_1}\alpha^{e_2} \quad D \not\equiv x^{f_1}\alpha^{e_2} \quad (d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1}$

toate calculele fiind făcute modulo p. Vom arăta că se ajunge la o contradicţie.

Testul de consistenţă (pasul 9) se rescrie $D \equiv d_0^{f_1}\alpha^{f_2}\ (mod\ p)$ unde $d_0 = d^{1/e_1}\alpha^{-e_2/e_1}\ (mod\ p)$ nu depinde decât de paşii $1 - 4$ ai protocolului. Aplicând Teorema 1.1 se obţine că y este o semnătură validă a lui d_0 cu probabilitate $1 - \dfrac{1}{q}$. Dar, prin ipoteză, y este o semnătură validă a lui x. Deci, cu mare probabilitate vom avea $x^a \equiv d_0^a\ (mod\ p)$ adică $x = d_0$.

Pe de-altă parte, din $d \not\equiv x^{e_1}\alpha^{e_2}\ (mod\ p)$ rezultă $x \not\equiv d^{1/e_1}\alpha^{-e_2/e_1}\ (mod\ p)$, adică tocmai $x \neq d_0$, contradicţie. □

1.7 Protocol de semnătură fără eşec

O semnătură fără eşec oferă protecţie contra unui adversar atât de puternic încât poate contraface semnături. Protocolul de semnătură prezentat aici este construit de *Heyst şi Pedersen* în 1992. Este un tip de semnătură *one - time*, compus dintr-o funcţie de semnătură, o funcţie de verificare şi un protocol pentru **proba de autentificare**. Prezentarea sa în detaliu este:

Fie $p = 2q + 1$ un număr prim cu q prim, şi $\alpha \in Z_p^*$ un element de ordin q. Pentru $1 \le a_0 \le q - 1$ se defineşte $\beta = \alpha^{a_0}$ $(mod\ p)$. Valorile p, q, α, β sunt publice şi considerate fixe. Valoarea a_0 este secretă pentru toată lumea (inclusiv *Bob*). Fie $\mathcal{P} = Z_q$, $\mathcal{A} = Z_q \times Z_q$. O cheie este de forma $K = (\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$ unde $a_1, a_2, b_1, b_2 \in Z_q, \gamma_1 = \alpha^{a_1}\beta^{a_2}$ $(mod\ p)$ $\quad \gamma_2 = \alpha^{b_1}\beta^{b_2}$ $(mod\ p)$. γ_1, g_2 sunt publice, a_1, a_2, b_1, b_2 sunt secrete. Dacă $x \in Z_q$, se defineşte $sig_K(x) = (y_1, y_2)$ unde $y_1 = a_1 + xb_1$ $(mod\ q)$ $\quad y_2 = a_2 + xb_2$ $(mod\ q)$. Pentru $y = (y_1, y_2) \in Z_q \times Z_q$, avem $ver_K(x, y) = T \Longleftrightarrow \gamma_1\gamma_2^x \equiv \alpha^{y_1}\beta^{y_2}$ $(mod\ p)$

Se poate vedea direct că o semnătură corect construită este validată de funcţia de verificare. Rămâne de studiat problema de securitate şi de ce procedeul este fără eşec. Să stabilim întâi câteva proprietăţi importante ale cheilor.

Două chei $(\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$ şi $(\gamma_1', \gamma_2', a_1', a_2', b_1', b_2')$ sunt *echivalente* dacă

$$\gamma_1 = \gamma_1',\ \ \gamma_2 = \gamma_2'.$$

În fiecare clasă de echivalenţă sunt exact q^2 chei.

Lema 1.1 *Dacă K, K' sunt chei echivalente, atunci*

$$ver_K = T \Longleftrightarrow ver_{K'} = T.$$

Demonstraţie: Fie $K = (\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$ şi $K' = (\gamma_1, \gamma_2, a_1', a_2', b_1', b_2')$ cu
$\gamma_1 \equiv \alpha^{a_1}\beta^{a_2}$ $(mod\ p) \equiv \alpha^{a_1'}\beta^{a_2'}$ $(mod\ p)$ şi $\gamma_2 \equiv \alpha^{b_1}\beta^{b_2}$ $(mod\ p) \equiv \alpha^{b_1'}\beta^{b_2'}$ $(mod\ p)$.

Să presupunem că mesajul x este semnat cu $y = (y_1, y_2)$ folosind cheia K, unde

$$y_1 \equiv a_1 + xb_1\ (mod\ q) \quad y_2 \equiv a_2 + xb_2\ (mod\ q)$$

Să presupunem că verificarea lui y se face cu cheia K':
$\alpha^{y_1}\beta^{y_2} \equiv \alpha^{a_1'+xb_1'}\beta^{a_2'+xb_2'}$ $(mod\ p) \equiv \alpha^{a_1'}\beta^{a_2'}(\alpha^{b_1'}\beta^{b_2'})^x$ $(mod\ p) \equiv \gamma_1\gamma_2^x$ $(mod\ p)$.

Deci y se verifică şi cu cheia K'. $\square$

Lema 1.2 *Fie o cheie K şi $y = sig_K(x)$. Există exact q chei K' echivalente cu K astfel încât $y = sig_{K'}(x)$.*

Demonstraţie: Fie γ_1, γ_2 componentele publice ale lui K. Trebuie determinat numărul de quadrupluri (a_1, a_2, b_1, b_2) astfel încât

$$\gamma_1 \equiv \alpha^{a_1}\beta^{a_2}\ (mod\ p) \quad \gamma_2 \equiv \alpha^{b_1}\beta^{b_2}\ (mod\ p)$$
$$y_1 \equiv a_1 + xb_1\ (mod\ q) \quad y_2 = a_2 + xb_2\ (mod\ q).$$

Cum α generează Z_q^*, există exponenţii unici $c_1, c_2, a_0 \in Z_q$ astfel încât

$$\gamma_1 \equiv \alpha^{c_1}\ (mod\ p), \quad \gamma_2 \equiv \alpha^{c_2}\ (mod\ p), \quad \beta \equiv \alpha^{a_0}\ (mod\ p).$$

În acest fel, este necesar şi suficient să avem:

$$c_1 \equiv a_1 + a_0a_2\ (mod\ q) \quad c_2 \equiv b_1 + a_0b_2\ (mod\ q)$$

$$y_1 \equiv a_1 + xb_1\ (mod\ q) \quad y_2 \equiv a_2 + xb_2\ (mod\ q).$$

Matricea acestui sistem de ecuaţii cu necunoscutele a_1, a_2, b_1, b_2 are rangul 3 (determinantul este $\begin{vmatrix} 1 & a_0 & 0 & 0 \\ 0 & 0 & 1 & a_0 \\ 1 & 0 & x & 0 \\ 0 & 1 & 0 & x \end{vmatrix}$), deci sistemul are cel puţin o soluţie netrivială obţinută cu ajutorul cheii K şi – în plus – dimensiunea spaţiului soluţiilor este $4 - 3 = 1$, deci există exact q soluţii. □

Lema 1.3 *Fie o cheie K, $y = sig_K(x)$ şi $ver_K(x', y') = T$ pentru $x' \neq x$. Există atunci cel puţin o cheie K' echivalentă cu K astfel ca*

$$y = sig_{K'}(x), \quad y' = sig_{K'}(x')$$

Demonstraţie: Se face printr-un raţionament analog cu cel din lema precedentă. □

Din ultimele două leme putem trage următoarea concluzie: fiind dată o semnătură validă y a unui mesaj x, există exact q chei posibile care pot semna x. Pentru orice alt mesaj $x' \neq x$, aceste q chei produc semnături diferite ale lui x'. Se obţine astfel teorema următoare:

Teorema 1.4 *Fiind date $sig_K(x) = y$ şi $x' \neq x$, Oscar nu poate calcula $sig_K(x')$ decât cu probabilitate $\frac{1}{q}$.*

De remarcat că rezultatul acestei teoreme nu depinde de puterea de calcul a lui *Oscar*; securitatea provine din faptul că el nu poate distinge care din cele q chei posibile a fost utilizată.

Se poate explica acum noţiunea de semnătură fără eşec. S-a arătat că fiind dat un mesaj x semnat cu y, *Oscar* nu poate calcula semnătura y' a lui *Bob* pe un alt mesaj x'. Ar mai fi posibilitatea ca *Oscar* să poată calcula o semnătură $y" \neq sig_K(x')$ care să fie validă. Dar, dacă ea ajunge înapoi la *Bob*, acesta poate furniza cu probabilitate $1 - \frac{1}{q}$ o *probă de autentificare*; aceasta este valoarea $a_0 = log_\alpha \beta$, cunoscută numai de autor.

Să presupunem că *Bob* are o pereche $(x', y")$ astfel încât

$$ver_K(x', y") = T \text{ şi } y" \neq sig_K(x').$$

Avem $\gamma_1 \gamma_2^{x'} \equiv \alpha^{y_1"} \beta^{y_2"}\ (mod\ p)$ unde $y" = (y_1", y_2")$.

Bob poate calcula propria sa semnătură pentru x', pe care o notează $y' = (y_1', y_2')$ şi are $\gamma_1 \gamma_2^{x'} \equiv \alpha^{y'_1} \beta^{y'_2}\ (mod\ p)$.

Deci $\alpha^{y"_1} \beta^{y"_2} \equiv \alpha^{y'_1} \beta^{y'_2}\ (mod\ p)$. Scriind $\beta = \alpha^{a_0}\ (mod\ p)$ se obţine:

$$\alpha^{y"_1 + a_0 y"_2} \equiv \alpha^{y'_1 + a_0 y'_2}\ (mod\ p) \text{ de unde } y"_1 + a_0 y"_2 \equiv y'_1 + a_0 y'_2\ (mod\ q)$$

sau $y"_1 - y'_1 \equiv a_0(y'_2 - y"_2)\ (mod\ q)$. Evident $y'_2 \not\equiv y"_2$ deoarece $y"$ este un fals.

Deci $(y'_2 - y"_2)^{-1}\ (mod\ q)$ există şi avem:

$$a_0 = log_\alpha \beta = (y"_1 - y'_1)(y'_2 - y"_2)^{-1}\ (mod\ q).$$

Bineînţeles, în verificarea probei de autentificare s-a presupus că nici *Bob* nu poate calcula logaritmul discret $log_\alpha \beta$.

Ca o remarcă finală, acest procedeu este cu utilizare unică, deoarece cheia K a lui *Bob* poate fi uşor determinată după două folosiri.

Exemplul 1.10 *Să presupunem* $p = 3467 = 2 \times 1733 + 1$. *Numărul* $\alpha = 4$ *are ordinul* 1733 *în* Z_{3467}^*. *Dacă se ia* $a_0 = 1567$ *vom avea* $\beta = 4^{1567} \ (mod\ 3467) = 514$.

Reamintim că Bob cunoaşte α *şi* β *dar nu* a_0.

Să presupunem că Bob construieşte cheia sa cu $a_1 = 888,\ a_2 = 1024,$
$b_1 = 786,\ b_2 = 999$ *deci*

$\gamma_1 = 4^{888}514^{1024} \ (mod\ 3467) = 3405 \quad \gamma_2 = 4^{786}514^{999} \ (mod\ 3467) = 2281.$

În acest moment Bob este pus în prezenţa semnăturii false $(822, 55)$ *a mesajului* 3383. *Această semnătură este validă, deoarece condiţia de verificare este satisfăcută:*

$$3405 \times 2281^{3383} \equiv 2282 \ (mod\ 3467) \quad 4^{822}514^{56} \equiv 2282 \ (mod\ 3467).$$

Dar Bob ştie că aceasta nu este semnătura sa şi trebuie să dovedească acest lucru. El calculează propria sa semnătură:

$$(888 + 3383 \times 786 \ (mod\ 1733), 1024 + 3383 \times 999 \ (mod\ 1733)) = (1504, 1291)$$

după care evaluează logaritmul discret

$$a_0 = (822 - 1504)(1291 - 55)^{-1} \ (mod\ 1733) = 1567$$

care constituie probă de autentificare, şi arată că semnătura nu îi aparţine.

1.8 Exerciţii

1.1 *Să presupunem că Bob utilizează semnătura El Gamal şi semnează mesajele x_1, x_2 obţinând (γ, δ_1) respectiv (γ, δ_2) (cu aceeaşi valoare a lui γ în ambele sem-nături). Se consideră în plus că $(\delta_1 - \delta_2, p-1) = 1$.*

- *Arătaţi că aceste informaţii sunt suficiente pentru determinarea lui k;*
- *Arătaţi cum se poate sparge protocolul de semnătură;*
- *Presupunând $p = 3187$, $\alpha = 5$, $\beta = 25703$, efectuaţi calculul lui k şi a plecând de la semnăturile $(23972, 31396)$ pentru $x = 8990$ şi $(23972, 20481)$ pentru $x = 31415$.*

1.2 *Protocolul de semnătură El Gamal este implementat folosind $p = 31847$, $\alpha = 5$, $\beta = 26379$. Să se scrie un program care:*

- *Verifică semnătura $(20679, 11082)$ a mesajului $x = 20543$.*
- *Calculează exponentul secret a prin compromisul spaţiu - timp al lui Shanks. Apoi determină valoarea aleatoare k utilizată în semnătura lui x.*

1.3 *Bob utilizează procedeul de semnătură El Gamal ca în Exemplul 1.1: $p = 467$, $\alpha = 2$, $\beta = 132$. Să presupunem căel semnează mesajul $x = 100$ cu $(29, 51)$. Calculaţi semnătura falsă pe care o poate obţine Oscar cu $h = 102$, $i = 45$, $j = 293$. Autentificaţi semnătura rezultată cu funcţia de verificare.*

1.4 *Arătaţi că a doua metodă de atac din semnătura El Gamal furnizează o semnă-tură corectă care satisface funcţia de verificare.*

1.5 *Modificăm puţin protocolul de semnătură El Gamal. Cheia este construită astfel: Bob alege o rădăcină primtivă $\alpha \in Z_p^*$, un exponent secret a $(0 \le a \le p-2)$, $(a, p-1) = 1$ şi $\beta = \alpha^a$. Cheia este $K = (\alpha, a, \beta)$ unde α, β sunt publice, iar a este secretă. Fie $x \in Z_p$ un mesaj care trebuie semnat. Bob calculează semnătura $sig_K(x) = (\gamma, \delta)$ prin:*

$$\gamma = \alpha^k \ (mod\ p) \quad \delta = (x - k\gamma)a^{-1} \ (mod\ p-1).$$

Singura diferentă fată de semnătura El Gamal este calculul lui δ.

- *Descrieţi cum se poate verifica cu cheia publică a lui Bob o semnătură (γ, δ) pe un mesaj x;*
- *Descrieţi avantajul noului procedeu (fată de cel vechi) din punct de vedere al calculelor;*
- *Comparaţi pe scurt securitatea celor două protocoale.*

1.6 *Bob utilizează procedeul DSS cu $q = 101$, $p = 7879$, $\alpha = 170$, $a = 75$, $\beta = 4567$. Determinaţi semnătura lui Bob pe mesajul $x = 5001$ utilizând valoarea aleatoare $k = 49$, şi arătaţi cum poate fi verificată semnătura rezultată.*

1.7 *În protocolul de semnătură Lamport, Bob semnează două mesaje* x, x', *ambele de câte* k *biţi. Fie* $s = d(x, x')$ *numărul de coordonate în care diferă cele două mesaje. Arătaţi că Oscar poate semna* $2^s - 2$ *mesaje noi.*

1.8 *În protocolul de semnătură Bos-Chaum cu* $k = 6$, $n = 4$ *sunt semnate mesajele* $x = (0, 1, 0, 0, 1, 1)$, $x' = (1, 1, 0, 1, 1, 1)$. *Determinaţi noile mesaje pe care le poate semna Oscar, plecând de la semnăturile lui* x *şi* x'.

1.9 *În protocolul de semnătură Bos-Chaum, Bob semnează două mesaje* x, x'. *Fie* $s = card(\phi(x) \cup \phi(x'))$. *Arătaţi că Oscar poate semna acum* $C_s^n - 2$ *mesaje noi.*

1.10 *Bob utilizează protocolul de semnătură incontestabilă Chaum-van Antwerpen ca în Exemplul 1.7; deci* $p = 467$, $\alpha = 4$, $a = 101$, $\beta = 449$. *Să presupunem că Bob este confruntat cu semnătura* $y = 25$ *a mesajului* $x = 157$ *şi doreşte să arate că ea este falsă. Presupunând că Alice alege valorile aleatoare* $e_1 = 46$, $e_2 = 123$, $f_1 = 198$, $f_2 = 11$ *în protocolul de dezminţire, calculaţi întrebările* c, d *ale lui Alice şi răspunsurile* C, D *ale lui Bob; verificaţi că Alice admite dezminţirea.*

1.11 *Arătaţi că clasele de echivalentă de chei în protocolul de semnătură fără eşec Pedersen - van Heyst conţine* q^2 *chei.*

1.12 *Bob utilizează protocolul de semnătură fără eşec Pedersen - van Heyst cu* $p = 3467$, $\alpha = 4$, $a_0 = 1567$, $\beta = 514$ *(valoarea lui* a_0 *nu este cunsocută de Bob).*

- *Folosind faptul că* $a_0 = 1567$, *determinaţi toate cheile posibile* $K = (\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$ *astfel ca* $sig_K(42) = (1118, 1449)$.
- *Presupunem* $sig_K(42) = (1118, 1449)$ *şi* $sig_K(969) = (899, 471)$. *Fără a utiliza valoarea lui* a_0, *determinaţi valoarea lui* K *(cea utilizată în protocolul cu cheie one - time).*

1.13 *Bob foloseşte protocolul de semnătură fără eşec Pedersen - van Heyst cu* $p = 5087$, $\alpha = 25$, $\beta = 1866$. *Cheia este* $K = (5065, 5076, 144, 874, 1873, 2345)$. *Se presupune că Bob este confruntat cu semnătura* $(2219, 458)$ *contrafăcută pe mesajul* 4785.

- *Arătaţi că ea satisface condiţia de verificare, deci este validă.*
- *Arătaţi cum poate Bob să calculeze proba de autenticitate plecând de la această semnătură.*

Bibliografie

[1] J. N. Bos, D. Chaum - Provably unforgable signatures; Lecture Notes in Computer Science, $740(1993), 1 - 14$

[2] D. Chaum, H. van Antwerpen - Undeniable signatures; Lecture Notes in Computer Science, $435(1990), 212 - 216$

[3] W. Diffie, M.E. Hellman - Multiuser cryptographic techniques; AFIPS Conference Proceedings, $45(1976), 109 - 112$

[4] T. El Gamal - A public key cryptosystem and a signature scheme based on discrete algorithms; IEEE Trans on Inf. Theory, $31(1985), 469 - 472$

[5] E. van Heyst, T.P.Petersen - How to make efficient fail-stop signatures; Lecture Notes in Computer Science, $658(1993), 366 - 377$

[6] C. J. Mitchell, F. Piper, P. Wild - Digital signatures; Contemporary Cryptology, The Science of Information Integrity, IEEE Press, $(1992), 325 - 378$

[7] M. E. Smid, D. K. Branstad - Response to comments on the $NIST$ proposed digital signature standard; Lecture Notes in Computer Science, $740(1993), 76 - 88$

[8] D. Stinton - Cryptographie, Theorie and Practique, Int. Thompson Publishing (1995)

[9] Digital signature standard; national Bureau of Standards, FIPS Publications 186, 1994