Merkle-Hellmanov sistem z nahrbtnikom

Merkle in Hellman sta leta 1978 predlagala ta sistem, že leta 1980 pa ga je razbil Shamir s pomočjo Lenstrinega algoritma za celoštevilčno programiranje (angl. integer programming).

Njegovo iterativno varianto pa je razbil malo kasneje Brickell.

Drugačen sistem z nahrbtnikom je predlagal Chor, razbil pa ga je Rivest.

Problem "podmnožica za vsoto"

Podatki: $I = (s_1, \ldots, s_n, T)$, *T je* **ciljna vsota**, *naravna števila* s_i *pa so* **velikosti**.

Vprašanje: *Ali obstaja tak binarni vektor* $\underline{x} = (x_1, \ldots, x_n)$, *za katerega velja* $\sum_{i=1}^{n} x_i s_i = T$?

Ta odločitveni problem je NP-poln:

– polinomski algoritem ni znan,
– isto velja tudi za ustrezen iskalni problem.

Ali za kakšno podmnožico problemov morda obstaja polinomskim algoritem?

Zaporedje $(s_1, \ldots, s_n)$ je **super naraščajoče**, če velja

$$s_j > \sum_{i=1}^{j-1} s_i \qquad \text{za} \;\; 2 \leq j \leq n.$$

Če je seznam velikosti super naraščajoč, potem lahko iskalno varianto zgornjega problema rešimo v času $O(n)$, rešitev $\underline{x}$ (če obstaja) pa je enolična.

Opišimo tak algoritem:

1. **for** $i = n$ **downto** 1 **do**
2.   **if** $T \geq s_i$ **then**
3. $T = T - s_i, \;\; x_i = 1$
4.   **else** $x_i = 0$
5. **if** $T = 0$ **then** $\underline{x} = (x_1, \ldots, x_n)$ je rešitev
6. **else** ni rešitve.

Naj bo $\underline{s} = (s_1, \ldots, s_n)$ super naraščajoč in

$$e_{\underline{s}} \,:\, \{0,1\}^n \longrightarrow \left\{0, \ldots, \sum_{i=1}^{n} s_i\right\}$$

funkcija, definirana s pravilom

$$e_{\underline{s}}(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i s_i.$$

Ali lahko to funkcijo uporabimo za enkripcijo?

Ker je $\underline{s}$ super naraščajoče zaporedje, je $e_{\underline{s}}$ injekcija, zgoraj opisani algoritem pa lahko uporabimo za dekripcijo.

Sistem **ni varen**, saj dekripcijo lahko opravi prav vsak.

Morda pa lahko transformiramo super naraščajoče zaporedje tako, da izgubi to lastnost in edino Bojan lahko opravi inverzno operacijo, da dobi super naraščajoče zaporedje.

Če napadalec Oskar ne pozna te transformacije, ima pred seboj primer (na videz) splošnega problema, ki ga mora rešiti, če hoče opraviti dekripcijo.

En tip takih transformacij se imenuje **modularna transformacija**. Izberemo si tak praštevilski modul p, da je

$$p > \sum_{i=1}^{n} s_i$$

ter število a, $1 \leq a \leq p-1$. Naj bo

$$t_i = \ as_i \bmod p, \qquad \text{za} \ \ 1 \leq i \leq n.$$

Seznam $\underline{t} = (t_1, \ldots, t_n)$ je javni ključ, ki ga uporabimo za enkripcijo, vrednosti a in p, ki definirata modularno transformacijo, pa sta tajni.

Zakaj smo si izbrali za p praštevilo?

Zakaj je bil ta sistem sploh zanimiv?

Primer: Naj bo

$$s = (2, 5, 9, 21, 45, 103, 215, 450, 946)$$

tajni super naraščajoči seznam velikosti.

Za $p = 2003$ in $a = 1289$ dobimo javni seznam velikosti

$$\underline{t} = (575, 436, 1586, 1030, 1921, 569, 721, 1183, 1570).$$

Anita zašifrira sporočilo $\underline{x} = (1, 0, 1, 1, 0, 0, 1, 1, 1)$:

$$y = 575 + 1586 + 1030 + 721 + 1183 + 1570 = 6665$$

ter ga pošlje Bojanu, ki najprej izračuna
$z = a^{-1}y \bmod p = 1643$ in nato
reši problem podmnožice zaporedja $\underline{s}$ za vsoto z.

6. poglavje

Sheme za digitalne podpise

- uvod (podpis z RSA sistemom)
- ElGamalov sistem za digitalno podpisovanje
- Digital Signature Standard
- napadi
- enkratni podpis
- podpisi brez možnosti zanikanja
- Fail-stop podpisi

Digitalni podpis je nadomestek za lastnoročni podpis pri elektronski izmenjavi in digitalnemu hranjeju podatkov.

Konceptualno se način zapisovanja informacij ni dramatično spremenil.

Medtem ko smo prej shranjevali in prenašali informacije na papirju, jih sedaj hranimo na magnetnih in drugih medijih ter jih prenašamo preko telekomunikacijskih sistemov (tudi brezžičnih).

Bistveno pa se je spremenila možnost kopiranja in spreminjanja informacij.

Zlahka naredimo na tisoče kopij neke informacije,
ki je shranjena digitalno, pri tem pa se
nobena ne razlikuje od originala.

Z informacijo na papirju je to precej težje.

Družba, v kateri so informacije
spravljene in prenašane v digitalni
obliki, mora poskrbeti za to,
da ne bo varnost informacij odvisna od
fizičnega medija, ki jih je zapisal ali prenesel.

Varnost informacij mora temeljiti izključno na
digitalni informaciji.

Eno izmed osrednjih orodij pri zaščiti informacij je **podpis**. Le-ta preprečuje poneverjanje, je dokaz o izvoru, identifikaciji, pričanju.

Podpis naj bi bil unikat vsakega posameznika, z njim se predstavimo, potrdimo, pooblastimo.

Z razvojem digitalne informacije moramo ponovno obdelati tudi koncept podpisa.

Ni več unikat, ki enolično določa podpisnika, kajti elektronsko kopiranje podpisa je tako lahko, da je skoraj trivialno na nepodpisan dokument pripeti poljuben podpis.

Potrebujemo protokole, ki imajo podobne lastnosti kot trenutni “papirni protokoli”.

Družba ima enkratno priložnost, da vpelje nove in učinkovitejše načine, ki nam bodo zagotovili varnost informacij.

Veliko se lahko naučimo iz dosedanjih sistemov, obenem pa moramo odpraviti tudi številne pomanjkljivosti.

Primerjava digitalnega in navadnega (lastnoročnega) podpisa:

- navadni podpis je fizično del podpisanega dokumenta;
- navadni podpis preverjamo s primerjanjem, digitalnega z algoritmom, katerega rezultat je odvisen od ključa in dokumenta;
- kopija digitalnega podpisa je identična originalu;
- digitalni podpis je odvisen od dokumenta, ki ga podpisujemo.

Sistem za digitalno podpisovanje je peterka $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$, za katero velja

1. $\mathcal{P}$ je končna množica sporočil,
2. $\mathcal{A}$ je končna množica podpisov,
3. $\mathcal{K}$ je končna množica ključev,
4. $\forall$ ključ $K \in \mathcal{K}$ obstaja algoritem za podpisovanje
$$\mathrm{sig}_K \in \mathcal{S}, \qquad \mathrm{sig}_K : \mathcal{P} \longrightarrow \mathcal{A}$$
in algoritem za preverjanje podpisa
$$\mathrm{ver}_K \in \mathcal{V}, \qquad \mathrm{ver}_K : \mathcal{P} \times \mathcal{A} \longrightarrow \{\mathrm{true}, \mathrm{false}\}.$$

Funkciji sig_K in ver_K imata to lastnost, da za vsako sporočilo $x \in \mathcal{P}$ in vsak podpis $y \in \mathcal{A}$ velja

$$\mathrm{ver}_K(x, y) = \begin{cases} \text{true}, & \text{če } y = \mathrm{sig}_K(x) \\ \text{false}, & \text{če } y \neq \mathrm{sig}_K(x) \end{cases}$$

Zahteve:

- algoritma sig_K in ver_K imata polinomsko časovno zahtevnost
- sig_K je znan le podpisniku
- ver_K je splošno znan
- računsko mora biti nemogoče ponarediti podpis

Primer: Algoritem RSA lahko uporabimo tudi za podpisovanje. Naj bo $n = pq$, kjer sta p in q praštevili.

Če je (n, d) skriti ključ, (n, e) pa javni, pri čemer je $de \equiv 1 \pmod{\varphi(n)}$, potem definiramo:

$$\mathrm{sig}_K(x) = d_K(x) = x^d \bmod n$$

$$\mathrm{ver}_K(x, y) = \text{true} \iff x = e_K(y) = y^e \bmod n$$

za $x, y \in \mathbb{Z}_n$.

Z zgornjim algoritmom je mogoče ponarediti podpis naključnih sporočil.

Ponarejevalec najprej izbere podpis y in nato izračuna

$$x \equiv y^e \pmod{n}.$$

Možnosti takega ponarejanja se izognemo z

- enosmernimi zgoščevalnimi funkcijami ali
- zahtevo, da ima sporočilo x določen pomen.

Pošiljanje podpisanih tajnih sporočil

Vrstni red šifriranja in digitalnega podpisovanja je pomemben.

1. Najprej podpisovanje:

$$x, \mathrm{sig}_{\mathrm{Anita}}(x) \longrightarrow e_{\mathrm{Bojan}}((x, \mathrm{sig}_{\mathrm{Anita}}(x))).$$

2. Najprej šifriranje $z = e_{\mathrm{Bojan}}(x)$,

   potem podpis $y = \mathrm{sig}_{\mathrm{Anita}}(z)$:

   Bojan prejme (z, y), odšifrira tajnopis $x = d_{\mathrm{Bojan}}(z)$ ter preveri podpis $\mathrm{ver}_{\mathrm{Anita}}(z, y)$.

V drugem primeru lahko napadalec Cene zamenja Anitin podpis s svojim:

$$y' = \mathrm{sig}_{\mathrm{Cene}}(z) \;\rightarrow (z, y') \;\rightarrow x = d_{\mathrm{Bojan}}(z),$$
$$\mathrm{ver}_{\mathrm{Cene}}(z, y')$$

in Bojan bo mislil, da je sporočilo prišlo od Ceneta.

Zato se priporoča najprej podpisovanje in nato šifriranje.

V primeru algoritma RSA je potrebno pri zaporednem podpisovanju in šifriranju paziti na velikosti modulov (*reblocking problem*).

Če je $n_{\text{Anita}} > n_{\text{Bojan}}$, se lahko zgodi, da Bojan ne bo mogel razvozlati sporočila. Naj bo

$$\begin{aligned}(n_{\text{Anita}}, e_{\text{Anita}}, d_{\text{Anita}}) &= (62894113, 5, 37726937),\\ (n_{\text{Bojan}}, e_{\text{Bojan}}, d_{\text{Bojan}}) &= (55465219, 5, 44360237).\end{aligned}$$

Anita podpiše sporočilo $x = 1368797$ in podpis zašifrira:

1. $s = x^{d_{\text{Anita}}} \bmod n_{\text{Anita}} = 59847900$,
2. $y = s^{e_{\text{Bojan}}} \bmod n_{\text{Bojan}} = 38842235$.

Bojan izračuna

1. $\widehat{s} = y^{d_{\text{Bojan}}} \bmod n_{\text{Bojan}} = 4382681$,
2. $\widehat{x} = \widehat{s}^{e_{\text{Anita}}} \bmod n_{\text{Anita}} = 54383568$.

Ker je $s > n_{\text{Bojan}}$, je $\widehat{x} \neq x = 1368797$.

Verjetnost tega dogodka je

$$\frac{n_{\text{Anita}} - n_{\text{Bojan}}}{n_{\text{Anita}}}.$$

Delitev shem za digitalno podpisovanje

1. Podpis je dodatek (ElGamal, DSA) sporočilu - sporočilo je možno rekonstruirati iz podpisa (RSA),
2. deterministični - nedeterministični,
3. enkratni - večkratni.

Različni sistemi za digitalno podpisovanje

- RSA
- ElGamal, DSS (*Digital Signature Standard*)
- Enkratni podpisi (*one-time signatures*)
- Slepi podpisi (*blind signatures*)
- Podpisi brez možnosti zanikanja (*undeniable signatures*)
- Skupinski podpisi (*group signatures*)
- Fail-Stop podpisi

ElGamalov sistem za digitalno podpisovanje

Za razliko od algoritma RSA je ElGamalov sistem namenjen predvsem digitalnemu podpisovanju, čeprav se ga da v posebnih primerih uporabiti tudi za šifriranje.

Podpis je nedeterminističen (odvisen od naključnega števila), torej sploh ni natanko določen.

Algoritem

Naj bo p takšno praštevilo, da je v $\mathbb{Z}_p$ težko izračunati diskretni logaritem in $\alpha \in \mathbb{Z}_p^*$ primitivni element.

Naj bo še $\mathcal{P} = \mathbb{Z}_p^*$, $\mathcal{A} = \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$ in

$$\mathcal{K} = \{(p, \alpha, a, \beta) : \beta \equiv \alpha^a \pmod{p}\}.$$

Število a je skrito (zasebno),

števila p, α in β pa so javno znana.

Podpisovanje: podpisnik s ključem $K = (p, \alpha, a, \beta)$ izbere naključno skrito število $k \in \mathbb{Z}^*_{p-1}$ in določi

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

kjer je

$$\gamma \equiv \alpha^k \bmod p$$

in

$$\delta \equiv (x - a\gamma)k^{-1} \bmod (p-1).$$

Preverjanje podpisa: (samo z javnimi p, α in β)

$$\mathrm{ver}_K(x, \gamma, \delta) = \mathrm{true} \iff \beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p.$$

Primer: Naj bo $p = 467, \alpha = 2$ in $a = 127$.
Potem je $\beta \equiv \alpha^a \bmod p = 132$. Recimo, da želimo podpisati $x = 100$, izbrali pa smo si tudi $k = 213$.
Podpis je enak (γ, δ), kjer je

$$\gamma \equiv 2^{213} \bmod 467 = 29$$

in

$$\delta \equiv (100 - 127{\cdot}29)\, 431 \bmod 466 = 51.$$

Pri preverjanju izračunamo

$$132^{29}\, 29^{51} \equiv 189 \bmod 467 \quad \text{in}$$

$$2^{100} \equiv 189 \bmod 467.$$

Zadnji vrednosti se ujemata, zato je podpis pravi.

Varnost ElGamalovega sistema za podpisovanje

Kako bi lahko ponaredili podpis, ne da bi vedeli za vrednost skritega števila a?

1. Za dano sporočilo x je potrebno najti tak par (γ, δ), da bo veljalo $\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$, torej
   - če izberemo γ: rabimo $\delta = \log_\gamma \alpha^x \beta^{-\gamma} \pmod p$,
   - če izberemo δ: glede na γ je potrebno rešiti enačbo $\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$,
   - hkrati računamo γ in δ (zaenkrat ni še nihče odkril hitrega postopka za reševanje zgornje enačbe).

2. Za podpis (γ, δ) je potrebno najti ustrezno sporočilo x:
$$x = \log_\alpha \beta^\gamma \gamma^\delta \pmod{p}.$$

3. Hkratno računanje x, γ in δ: naj bosta i in j takšni števili, da velja $0 \leq i, j \leq p-2$ in $D(j, p-1) = 1$. Potem števila
$$\begin{aligned} \gamma &\equiv \alpha^i \beta^j \pmod{p}, \\ \delta &\equiv -\gamma j^{-1} \pmod{(p-1)}, \\ x &\equiv -\gamma i j^{-1} \pmod{(p-1)} \end{aligned}$$
zadoščajo enačbi $\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod{p}$.

Primer: Če je $p = 467, \alpha = 2$ in $\beta = 132$, lahko z izbiro $i = 99$ in $j = 179$, dobimo veljaven podpis $(117, 41)$ za sporočilo 331.

4. Ali lahko pri veljavnem podpisu (γ, δ) za x najdemo še kakšen podpis za neko drugo sporočilo x'? Odgovor je "DA".

   Naj bodo h, i in j takšna števila, da zanje velja $0 \le h, i, j \le p - 2$ in $D(h\gamma - j\delta, p - 1) = 1$. Potem je par (λ, μ) veljaven podpis za x', kjer je

$$\begin{aligned} \lambda &= \gamma^h \alpha^i \beta^j \bmod p, \\ \mu &= \delta\lambda(h\gamma - j\delta)^{-1} \bmod (p-1), \\ x' &= \lambda(hx + i\delta)(h\gamma - j\delta)^{-1} \bmod (p-1). \end{aligned}$$

Nevarnosti pri napačni uporabi ElGamalovega sistema

1. Če naključno število k ne ostane skrito, lahko izračunamo
$$a = (x - k\delta)\gamma^{-1} \bmod (p-1).$$
2. Število k lahko uporabimo le enkrat, sicer ga je mogoče zlahka izračunati.

Digital Signature Standard

DSS je modifikacija ElGamalovega sistema za podpisovanje. Kot ameriški standard je bil predlagan leta 1991, sprejet pa leta 1994.

Algoritem: Naj bo p praštevilo velikosti L bitov, kjer je $512 \leq L \leq 1024$ in $64 \,|\, L$, q $160-$bitno praštevilo, da $q \,|\, p-1$, ter $\alpha \in \mathbb{Z}_p^*$ $q-$ti koren enote po modulu p. Definirajmo $\mathcal{P} = \mathbb{Z}_p^*$, $\mathcal{A} = \mathbb{Z}_q \times \mathbb{Z}_q$ in

$$\mathcal{K} = \{(p, q, \alpha, a, \beta) : \beta \equiv \alpha^a \pmod{p}\}.$$

Vrednosti p, q, α in β so javne, število a pa skrito.

Podpisovanje: podpisnik izbere naključno skrito število k, $1 \leq k \leq q-1$ in določi

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

kjer je

$$\gamma \equiv (\alpha^k \bmod p) \bmod q$$

in

$$\delta \equiv (x + a\gamma)\, k^{-1} \pmod{q}.$$

Za število δ mora veljati $\delta \not\equiv 0 \pmod{q}$.

Preverjanje podpisa: najprej izračunamo

$$e_1 \equiv x\delta^{-1} \quad \text{in} \quad e_2 \equiv \gamma\delta^{-1}.$$

Potem je

$$ver_K(x,\gamma,\delta) = \text{true}$$
$$\Updownarrow$$
$$(\alpha^{e_1}\,\beta^{e_2} \bmod p) \bmod q = \gamma.$$

Podobno kot pri ElGamalovi shemi je podpisovanje hitrejše od preverjanja (za razliko od RSA).

Prikrit kanal v algoritmu DSA

V algoritmu DSA obstaja prikrit kanal, ki omogoča:

(a) vključitev šifriranega sporočila v podpis, ki ga lahko prebere le tisti, ki pozna dodaten ključ;

(b) razkritje skritega ključa, brez vednosti njegovega lastnika.

Eno možnost za (a) si oglejmo na naslednji foliji, točko (b) pa prihranimo za domačo nalogo.

Primer: Izberimo n tajnih praštevil $p_1, \ldots, p_n$ in poskusimo v podpis skriti binarno zaporedje $b_1, \ldots, b_n$. Naključno število k izbiramo toliko časa, da za vsak $1 \leq i \leq n$ velja

$$b_i = 1 \Longrightarrow \gamma \text{ je kvadratni ostanek po modulu } p_i,$$

$$b_i = 0 \Longrightarrow \gamma \text{ ni kvadratni ostanek po modulu } p_i,$$

kjer je $\text{sig}_K(x, k) = (\gamma, \delta)$.

Napadi

Uganjevanje fraz, ki jih uporabljamo za gesla

primer	število znakov	zahtevnost	dolžina gesla	čas za razbijanje
mucka	5	25 (majhne črke)	12 bitov	40 minut
br1a9Az	7	62 (črke in številke)	24 bitov	22 let
TH,X1lb<V+	10	95 (znaki na tipkov.)	40 bitov	nedosegljivo

Če uporabimo angleško ali slovensko besedo, dobimo zaporedje s približno 1.3 biti entropije na en znak (t.j. prostor za besedo proti popolnoma naključnim znakom).

Napadi z grobo silo (angl. Brute Force Attack)

posameznik ima 1 PC in programsko opremo
($2^{17} - 2^{24}$ ključev/sek.),

majhna skupina, 16 PC ($2^{21} - 2^{28}$ ključev/sek.),

akademska omrežja, 256 PC ($2^{25} - 2^{32}$ ključev/sek.),

veliko podjetje z \$1.000.000 za strojno opremo
(2^{43} ključev/sek.),

vojaška obveščevalna organizacija z \$1.000.000.000 za strojno opremo in napredno tehnologijo
(2^{55} ključev/sek.).

Napadi z grobo silo

dolžina ključa (v bitih)	posamični napadalec	majhne skupine	razisko-valna omrežja	velika podjetja	vojaške obveščevalne službe
40	tedni	dnevi	ure	milisekunde	mikrosekunde
56	stoletja	desetletja	leta	ure	sekunde
64	tisočletja	stoletja	destletja	dnevi	minute
80	∞	∞	∞	stoletja	stoletja
128	∞	∞	∞	∞	tisočletja

Povprečen čas za napad z grobo silo

dolžina ključev (v bitih)	**število možnih ključev**	**potreben čas za eno šifriranje/μsek.**	**potreben čas za 10^6 šifriranj/μsek.**
32	$2^{32} = 4.3 \times 10^9$	$2^{31}\mu\text{sec} \approx 36$ min	≈ 2 milisek.
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55}\mu\text{sec} \approx 1142$ let	≈ 10 ur
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127}\mu\text{sec} \approx 5 \times 10^{24}$	$\approx 5 \times 10^{18}$ let

Napadi na PKS

Napadi na DSA

- Metoda Index Calculus ($p \approx 2^{1024}$)

- Pollardova ρ-metoda ($\sqrt{\pi q/2}, \quad q \approx 2^{160}$)

Napadi na ECDSA

- Pollardova ρ-metoda ($\sqrt{\pi n/2}, \quad n \approx 2^{160}$)

Programski napadi

MIPS računalnik lahko opravi 4×10^4 seštevanj točk na eliptični krivulji na sekundo.

(Ta ocena je precej konzervativna. Posebaj prirejeno integrirano vezje s frekvenco ure 40 MHz, ki opravlja operacije na eliptični krivulji nad obsegom $GF(2^{155})$ in lahko izvede 40.000 seštevanj na sekundo.)

Na osnovi tega zaključimo, da je število seštevanj na eliptični krivulji na $GF(2^{155})$ izvedeno na MIPS računalniku v času enega leta naslednje

$$(4 \times 10^4) \cdot (60 \times 60 \times 24 \times 365) \approx 2^{40}.$$

Spodnja tabela nam kaže kolikšno računsko moč potrebujemo za računanje problema diskretnega logaritma z uporabo Pollard ρ-methodo za različne vrednosti števila n. MIPS leto je ekvivalentno računski moči 1 MIPS računalnika, ki je na voljo eno leto.

velikost obsega (v bitih)	velikost števila n	$\sqrt{\pi n/2}$	MIPS let
155	150	2^{75}	3.8×10^{10}
210	205	2^{103}	7.1×10^{18}
239	234	2^{117}	1.6×10^{23}

Npr. če imamo na voljo 10.000 računalnikov z močjo 1.000 MIPS in je $n \approx 2^{150}$, potem je lahko problem diskretnega logaritma na eliptični krivulji rešen v 3.800 letih.

Prejšnjo tabelo je zanimivo primerjati s Odlyzko-vo tabelo, ki kaže kolikšno računsko moč potrebujemo za faktorizacijo celih števil s sedanjo verzijo splošnega NFS algoritma.

velikost števila n (v bitih)	MIPS let
512	3×10^4
768	2×10^8
1024	3×10^{11}
1280	1×10^{14}
1536	3×10^{16}
2048	3×10^{20}

Hardwarski napadi

Za bolj perspektiven napad (s strani dobro financiranega napadalca) na ECC, bi bilo potrebno narediti specializirano programsko opremo za paralelno iskanje na osnovi Pollard ρ-metode.

Van Oorschot and Wiener ocenjujeta: za $n \approx 10^{36} \approx 2^{120}$ bi računalnik z $m = 325.000$ procesorji (cena okoli 10 milijonov USD) lahko izračunal diskretni logaritem v približno 35 dneh.

Poudariti moramo, da računanje diskretnega logaritma na $E(\mathbb{Z}_p)$ v zgoraj omenjenih napadih odkrije **en sam** *zasebni ključ.*

M. Blaze, W. Diffie, R. Rivest, B. Schneier, T. Shimomura, E. Thompson, and M. Wiener, January 1996, (`http://theory.lcs.mit.edu/rivest/publications.html`) govorijo o minimalnih dolžinah ključev potrebnih za varen simetrični sistem (npr. DES ali IDEA):

> *Da bi zagotovili ustrezno zaščito proti najbolj resnim grožnjam (npr. velike komercialne ustanove in vladne agencije) mora ključ biti dolg vsaj 75 bitov. Za zaščito za naslednjih 20-let morajo ključi biti dolgi vsaj 90 bitov (pri tem upoštevamo pričakovano rast računske moči).*

Če posplošimo te zaključke na eliptične kripto-sisteme, mora biti praštevilo n, ki zagotovlja kratkoročno varnost, dolgo vsaj 150 bitov, za srednjeročno varnost pa vsaj 180 bitov.

Dolžina ključev

simetrične šifre (AES)	asimetrične (RSA, DSA, DH)	eliptične krivulje
40 bitov	274 bitov	80 bitov
56 bitov	384 bitov	106 bitov
64 bitov	512 bitov	132 bitov
80 bitov	1024 bitov	160 bitov
96 bitov	1536 bitov	185 bitov
112 bitov	2048 bitov	237 bitov
120 bitov	2560 bitov	256 bitov
128 bitov	3072 bitov	270 bitov

Digitalni podpisi v $\mathbb{Z}_p$ in na EC

grupa	$\mathbb{Z}_p^*$	$E(\mathbb{Z}_p)$
elementi	množica celih števil $\{1, 2, \ldots, p-1\}$	točke (x, y), ki zadoščajo enačbi eliptične krivulje E in še točka v neskončnosti
operacija	množenje po modulu p	seštevanje točk na eliptični krivulji
oznake	elementi: g, h množenje: $g \times h$ multiplikativni inverz: h^{-1} deljenje: g/h potenciranje: g^a	elementi: P, Q seštevanje: $P + Q$ nasprotna točka: $-Q$ odštevanje: $P - Q$ skalarno množenje točke: aP
problem diskretnega logaritma	Za dana $g, h \in \mathbb{Z}_p^*$ poišči tako celo število a da je $h = g^a \bmod p$.	Za dani točki $P, Q \in E(\mathbb{Z}_p)$ poišči tako celo število a da je $Q = aP$.

Grupe

Digital Signature Algorithm (DSA) eliptični analog ECDSA

DSA	ECDSA
1. Izberi praštevili p in q velikosti $2^{1023} < p < 2^{1024}$, $2^{159} < q < 2^{160}$, tako da $q \mid p-1$.	1. Izberi tako eliptično krivuljo E: $y^2 = x^3 + ax + b$ nad $\mathbb{Z}_q$, da je število $\lvert E(\mathbb{Z}_p)\rvert$ deljivo s praštevilom $n \approx 160$-bitov.
2. $t \in \mathbb{Z}_p^*$, izračunaj $g = t^{(p-1)/q} \bmod p$, potem je $g \neq 1$ in ima red q v $\mathbb{Z}_p^*$.	2. Izberi točko P na $E(\mathbb{Z}_q)$ katere red je praštevilo n.
3. Uporabi multiplikativno grupo $\{g^0, g^1, \ldots, g^{q-1}\}$	3. Uporabi aditivno grupo $\{\mathcal{O}, P, 2P, \ldots, (n-1)P\}$

Generiranje ključa pri DSA in ECDSA

DSA	ECDSA
1. Izberi naključno celo število $x \in [2, q-2]$, tj. **zasebni ključ**	1. Izberi naključno celo število $d \in [2, n-2]$, tj. **zasebni ključ**
2. Izračunaj $y = g^x \bmod p$, **javni ključ** je (p, q, g, y).	2. Izračunaj $Q = dP$, **javni ključ** je (E, n, q, Q).

DSA	ECDSA
q	n
g	P
x	d
y	Q

Podpisovanje sporočila m

DSA	ECDSA
1. Izberi naključno celo število $k \in [2, q-2]$.	1. Izberi naključno celo število $k \in [2, n-2]$.
2. Izračunaj $g^k \bmod p$, $r = (g^k \bmod p) \bmod q$, $0 \neq s = k^{-1}(h(m) + xr) \bmod q$.	2. Izračunaj $kP = (x_1, y_1)$, $r = x_1 \bmod n$, $0 \neq s = k^{-1}(h(m) + dr) \bmod n$.
Podpis je par (r, s).	

Preverjanje podpisa (r, s) sporočila m osebe A

DSA	ECDSA
1. Preskrbi si avtentično kopijo javnega ključa osebe A: (p, q, g, y)	(E, n, q, Q)
2. Izračunaj $s^{-1} \bmod p$ in $h(m)$, $u_1 = h(m)s^{-1} \bmod q$, $u_2 = rs^{-1} \bmod q$, $v = (g^{u_1} y^{u_2} \bmod p) \bmod q$.	2. Izračunaj $s^{-1} \bmod n$ in $h(m)$ $u_1 = h(m)s^{-1} \bmod n$, $u_2 = rs^{-1} \bmod n$, $u_1 P + u_2 Q = (x_0, y_0)$ in $v = x_0 \bmod n$.
Sprejmi podpis samo in samo če je $v = r$.	

SigGen z EC

Razvita je bila v **Certicom Corp., Kanada**, v sodelovanju s Schlumberger Smart Cards and Systems.

Uporablja Motorolin čip 68SC28:

- ROM 12.790 zlogov,
- EEPROM 8.112 zlogov,
- RAM 240 zlogov.

Vsebuje tehnologijo MULTIFLEX^{TM} ter tehnologijo eliptičnih krivulj $(CE)^2$, ki jo razvija podjetje Certicom Corp.

SigGen kartica je zelo prikladna za končnega uporabnika ter za proces prepoznavanja:

- je poceni,
- podpis je opravljen v pol sekunde,
- rabi samo 90 zlogov RAM-a,
- program ne zasede niti 4 KB.

Je edina pametna kartica, ki opravi digitalni podpis kar z obstoječim procesorjem.

Eliptični kripto-sistemi nudijo največjo moč glede na število bitov ključa med današnjimi javnimi kripto-sistemi.

Manjši ključi omogočajo

- manjše sistemske parametre,
- manjša potrdila z javnimi ključi,
- hitrejšo implementacijo,
- manjše zahteve po energiji,
- manjše procesorje,

itd.