Dixonov algoritem in kvadratno rešeto

$(x \not\equiv \pm y \pmod n, x^2 \equiv y^2 \pmod n \implies D(x-y,n) \neq 1)$

Sestavimo bazo faktorjev $\mathcal{B} = \{p_1, \ldots, p_B\}$, kjer so p_i "majhna" praštevila. Naj bo C malo večji kot B (npr. $C = B + 10$). Najdemo C kongruenc:

$$x_j^2 \equiv p_1^{\alpha_{1,j}} \times p_2^{\alpha_{2,j}} \times \cdots \times p_B^{\alpha_{B,j}} \pmod n, \quad 1 \leq j \leq C$$

Označimo $a_j := (\alpha_{1,j} \bmod 2, \ldots, \alpha_{B,j} \bmod 2)$.
Če najdemo podmnožico $\{a_1, \ldots, a_C\}$, v kateri se vektorji seštejejo v $(0, 0, \cdots, 0) \bmod 2$, potem bo produkt x_j uporabil vsak faktor iz $\mathcal{B}$ sodo mnogokrat.

Primer: $n = 15770708441$, $\mathcal{B} = \{2, 3, 5, 7, 11, 13\}$

$$\begin{aligned}
8340934156^2 &\equiv 3 \times 7 \pmod n & a_1 &= (0,1,0,1,0,0) \\
12044942944^2 &\equiv 2 \times 7 \times 13 \pmod n, & a_2 &= (1,0,0,1,0,1) \\
2773700011^2 &\equiv 2 \times 3 \times 13 \pmod n, & a_3 &= (1,1,0,0,0,1)
\end{aligned}$$

Iz $a_1 + a_2 + a_3 = (0,0,0,0,0,0) \bmod 2$ sledi

$$(8340934156 \times 12044942944 \times 2773700011)^2 \equiv$$
$$\equiv (2 \times 3 \times 7 \times 13)^2 \pmod n$$
oziroma $9503435785^2 \equiv 546^2 \pmod n$
in $D(9503435785 - 546, 15770708441) = 115759$.

- Linearno odvisnost med vektorji $\{a_1, a_2, \cdots, a_C\}$ poiščemo npr. z Gaussovo eliminacijo.
- $C > B + 1$: vendar imamo raje več različnih odvisnosti, da bo vsaj ena dala faktorizacijo.
- Števila x_j, za katere se da $x_j^2 \pmod n$ faktorizirati v $\mathcal{B}$, iščemo v množici $\{x_j = j + \lfloor \sqrt{n} \rfloor \mid j = 1, 2, ...\}$ z metodo **kvadratnega rešeta** (Pomerance).
- Če je $\mathcal{B}$ velik, je večja možnost, da se da neko število faktorizirati v $\mathcal{B}$, a potrebujemo več kongruenc, da najdemo linearno odvisnost. ($|\mathcal{B}| \approx \sqrt{e^{\sqrt{\ln n \, \ln \ln n}}}$).

Algoritmi za faktorizacijo v prak

Kvadratno rešeto	$O(e^{(1+o(1))\sqrt{\ln n \ln \ln n}})$
Eliptične krivulje	$O(e^{(1+o(1))\sqrt{\ln p \ln \ln p}})$
Številsko rešeto	$O(e^{(1.92+o(1))(\ln n)^{1/3}(\ln \ln n)^{2}}$

$o(1) \to 0$, ko $n \to \infty$
p - najmanjši praštevilski faktor n

V najslabšem primeru, ko je $p \approx \sqrt{n}$, imata kv
rešeto in eliptične krivulje približno enako
zahtevnost, sicer pa je boljše kvadratno rešeto

Faktorizacije velikih števil s kvadratnim rešetom:
($n = p \cdot q$, $p \approx q$)

69	cifer	1983	Davis, Holdredge, Simmons (sestavljen faktor od $2^{251} - 1$)
< 106	cifer	1989	Lenstra, Manasse + pomočniki
129	cifer	1994	Atkins, Graff, Lenstra, Leyland +600 pomočnikov

Fermatova števila:

$2^{2^{11}} - 1$ eliptične krivulje: 1988 (Brent)

$2^{2^9} - 1$ številsko rešeto:
1990 (Lenstra, Lenstra, Manasse, Pollard)

Prof. Vidav je leta 1997 zastavil naslednje vprašanje (morda tudi zato, da preveri trenutne moči namiznih računalnikov): poišči prafaktorje števila

$$10^{64} + 1$$

in namignil, da so vsi prafaktorji, če jih je kaj, oblike $128k + 1$.

Večina osebnih računalnikov z Mathematica
hitro najde en faktor:

1265011073

55-mestni ostanek pa povzroči težave.
V Waterlooju sem končno našel hiter računalnik
(cacr: Alpha ???) ter hitro programsko oprem
(glej `http://www.informatik.th-darmstadt.de/TI/LiDIA/`), ki je
kot 10-ih minutah našla še preostala prafakto

153431681888891378183 69

515217525265213267447869906815873

5. poglavje

Drugi javni kriptosistemi

- ElGamalovi kriptosistemi in Massey-Omura shema
- Problem diskretnega logaritma in napadi nanj
- Metoda velikega in malega koraka
- Pohlig-Hellmanov algoritem
- Index calculus
- Varnost bitov pri diskretnem logaritmu
- Končni obsegi in eliptične krivulje (ponovitev)
- Eliptični kriptosistemi
- Merkle-Hellmanov sistem z nahrbtnikom
- Sistem McEliece

Javna kriptografija

L. 1976 sta Whit **Diffie** in Martin **Hellman** predstavila koncept kriptografije z javnimi ključi.

Le-ta za razliko od simetričnega sistema uporablja dva različna ključa, **privatnega** in **javnega**.
V prejšnjem poglavju smo spoznali RSA (1978).

Taher ElGamal (1985): enkripcije z javnimi ključi in sheme digitalnih podpisov.

Varianta: algoritem za digitalni podpis
(**Digital Signature Algorithm – DSA**),
ki ga je prispevala vlada ZDA.

V razvoju javne kriptografije je bilo razbitih veliko predlaganih sistemov.

Le tri vrste so se ohranile in jih danes lahko smatramo za varne in učinkovite.

Glede na matematični problem, na katerem t
so razdeljene v tri skupine:

- **Sistemi faktorizacije celih števil**
  (Integer Factorization Systems)
  z RSA (Rivest-Adleman-Shamir)
  kot najbolj znanim predstavnikom,
- **Sistemi diskretnega logaritma**
  (Discrete Logarithem Systems),
  kot na primer DSA,
- **Kriptosistemi z eliptičnimi krivulja**
  (Elliptic Curve Cryptosystems).

Problem diskretnega logaritma v grupi G

za dana $\alpha, \beta \in G$, kjer je red elementa α enak n, najdi $x \in \{0, \ldots, n-1\}$, tako da je $\alpha^x = \beta$.

Število x se imenuje **diskretni logaritem** osnove α elementa β.

Medtem ko je diskretni logaritem (verjetno) težko izračunati (v splošnem), lahko potenco izračunamo hitro (primer enosmerne funkcije).

Problem diskretnega logaritma v grupi $\mathbb{Z}_p$

Trenutno ne poznamo nobenega polinomskega algoritma za DLP.

Praštevilo p mora imeti vsaj 150 mest (500 bitov), $p-1$ pa mora imeti vsaj en "velik" prafaktor.

ElGamalovi protokoli

Delimo jih v tri razrede:

1. protokoli za izmenjavo ključev,
2. sistemi z javnimi ključi,
3. digitalni podpisi.

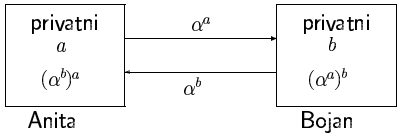
Te protokole lahko uporabimo s poljubno končno grupo G.

Osnovna razloga za uporabo različnih grup:

- operacije v nekaterih grupah so izvedene en
  v programih (software) in
  programski opremi (hardware)
  kot v drugih grupah,
- problem diskretnega logaritma je lahko v
  grupi zahtevnejši kot v drugi.

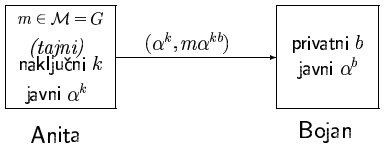
Naj bo $\alpha \in G$ in naravno število n red tega e
(t.j., $\alpha^n = 1$ in $\alpha^k \neq 1$ za vsak $k < n$).

1. Izmenjava ključev
(Diffie-Hellman)



Anita in Bojan si delita skupni element grupe: $(\alpha^a)^b = (\alpha^b)^a = \alpha^{ab}$.

2. ElGamalov kriptosistem javnih ključev
(dva ključa, asimetrični sistem)



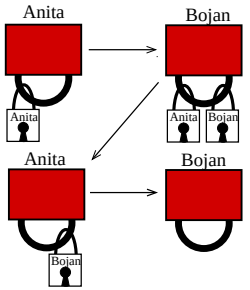
Če je $(y_1, y_2) = e_K(m,k) = (\alpha^k, m\alpha^{kb})$, potem je odšifriranje definirano z $d_K(y_1, y_2) = y_2(y_1^b)^{-1}$.

Sporočilo m lahko prebere le Bojan (s svojim b), ni pa nikjer rečeno, da mu ga je res poslala Anita (saj ni uporabila svojega privatnega ključa).

V javni kriptografiji smatramo, da nam javni del (npr. α^k, α^b) v ničemer ne pomaga pri iskanju tajnega/privatnega dela (npr. k, b).

(Digitalni podpis bo obravnavan v 6. poglavju.)

Massey-Omura shema



Zgled:
za G si izberemo grupo $GF(23)^*$.

Elementi obsega $GF(23)$ so: $0, 1, \ldots, 22$.

Definirajmo:
$a + b = r_1$, kjer je r_1 vsota $a + b$ mod 23.
$ab = r_2$, kjer je r_2 produkt ab mod 23.

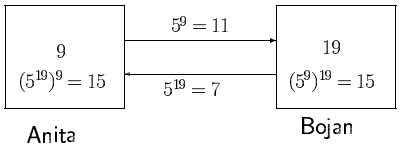
Primer: $12 + 20 = 32 = 9$, $8 \cdot 9 = 72 = 3$.

Multiplikativna grupa $GF(23)^*$

Elementi $GF(23)^*$ so elementi $GF(23)\backslash\{0\}$ in jih lahko generiramo z enim elementom:

$5^0 = 1$	$5^8 = 16$	$5^{16} = 3$
$5^1 = 5$	$5^9 = 11$	$5^{17} = 15$
$5^2 = 2$	$5^{10} = 9$	$5^{18} = 6$
$5^3 = 10$	$5^{11} = 22$	$5^{19} = 7$
$5^4 = 4$	$5^{12} = 18$	$5^{20} = 12$
$5^5 = 20$	$5^{13} = 21$	$5^{21} = 14$
$5^6 = 8$	$5^{14} = 13$	$5^{22} = 1$
$5^7 = 17$	$5^{15} = 19$	

Diffie–Hellmanov protokol v $GF(23)^*$



Anita in Bojan si sedaj delita skupen element $5^{9 \cdot 19} = 15$.

Log tabela

log	elt	log	elt	log	elt
0	1	8	16	16	3
1	5	9	11	17	15
2	2	10	9	18	6
3	10	11	22	19	7
4	4	12	18	20	12
5	20	13	21	21	14
6	8	14	13		
7	17	15	19		

Grupo G in generator α si izberemo tako, d
elementa α velik (s tem pa je velika tudi log t

Antilog tabela

elt	log	elt	log	elt	log
1	0	9	10	17	7
2	2	10	3	18	12
3	16	11	9	19	15
4	4	12	20	20	5
5	1	13	14	21	13
6	18	14	21	22	11
7	19	15	17		
8	6	16	8		

Algoritmi za računanje diskretnega logaritma

- Shankov algoritem (veliki korak – mali korak),
- Pollardov ρ-algoritem,
- Pohlig-Hellmanov algoritem,
- metoda "index calculus".

Danes si bomo ogledali samo prvega in zadnja dva.

Metoda veliki korak – mali korak:

$GF(23)^*$ z gen. 5: sestavi tabelo elementov $5^0, 5^5, 5^{10}, 5^{15}, 5^{20}$ in njihovih logaritmov.

element	1	20	9	19	12
logaritem	0	5	10	15	20

Izračunaj log(18): računaj $5 \times 18, 5^2 \times 18, \cdots$, vse dokler ne dobiš elementa iz tabele.
$5 \times 18 = 21$, $5^2 \times 18 = 13$, $5^3 \times 18 = 19$.
Iz tabele dobimo $\log(5^3 \times 18) = \log 19 = 15$.
Sledi $3 + \log 18 = 15$ oziroma $\log 18 = 12$.

$GF(89)^*$ z generatorjem 3: sestavi tabelo el
$3^0, 3^{10}, 3^{20}, \cdots, 3^{80}$ in njihovih algoritmov.

elt	1	42	73	40	78	72	87	5	32
log	0	10	20	30	40	50	60	70	80

Izračunaj log(36): računaj $3 \times 36, 3^2 \times 36,$
dokler ne dobiš elementa iz tabele.
$3 \times 36 = 19, 3^3 \times 36 = 82, 3^5 \times 36 = 26, 3^2 \times$
$3^4 \times 36 = 68, 3^6 \times 36 = 78.$
Iz tabele preberemo $\log(3^6 \times 36) = \log 78$
$6 + \log 36 = 40$ oziroma $\log 36 = 34$.

Čim daljša je tabela, ki jo sestavimo, tem dlje časa jo je treba računati (enkratni strošek), po drugi strani pa hitreje naletimo na element v krajši tabeli.

Običajno sestavimo tabelo velikosti $m = \lfloor\sqrt{|G|}\rfloor$ in za iskanje potrebujemo $O(m)$ čas.

Pollardov ρ algoritem (s Floydovim algoritmom za iskanje ciklov)

Ima isto časovno zahtevnost kot metoda veliki korak – mali korak, porabi pa le malo spomina.

Pohlig-Hellmanov algoritem

$$p - 1 = \Pi_{i=1}^{k} p_i^{c_i}$$

za različna praštevila p_i. Vrednost $a = \log_\alpha \beta$ je natanko določena po modulu $p - 1$.

Najprej izračunamo $a \bmod p_i^{c_i}$ za vsak $i = 1, \ldots, k$ in nato izračunamo $a \bmod (p-1)$ po kitajskem izreku o ostankih.

Predpostavimo, da je q praštevilo in c največje naravno število, za katero velja

$$p - 1 \equiv 0 \pmod{q^c}.$$

Kako izračunamo

$$x = a \bmod q^c, \quad \text{kjer je } 0 \le x \le q^c - 1?$$

Zapišimo x v številskem zapisu z osnovo q:

$$x = \sum_{i=0}^{c-1} a_i q^i, \quad \text{kjer je } 0 \le a_i \le q - 1.$$

Od tod dobimo

$$a = a_0 + a_1 q + \cdots + a_{c-1} q^{c-1} + s q^c,$$

kjer je s neko naravno število in $a = a_0 + Kq$
a_0 izračunamo iz naslednje identitete

$$\beta^{(p-1)/q} \equiv \alpha^{a_0 (p-1)/q} \bmod p.$$

Dokažimo slednjo kongruenco:

$$\begin{aligned}\beta^{(p-1)/q} &\equiv (\alpha^a)^{(p-1)/q} \bmod p \\ &\equiv (\alpha^{a_0+Kq})^{(p-1)/q} \pmod p \\ &\equiv \alpha^{a_0(p-1)/q}\,\alpha^{(p-1)K} \pmod p \\ &\equiv \alpha^{a_0(p-1)/q} \pmod p.\end{aligned}$$

Najprej torej izračunamo

$$\beta^{(p-1)/q} \pmod p.$$

Če je $\beta^{(p-1)/q} \equiv 1 \pmod p$, je $a_0 = 0$,
sicer pa zaporedoma računamo

$$\gamma = \alpha^{(p-1)/q} \pmod p, \quad \gamma^2 \pmod p, \quad \ldots,$$

vse dokler ne dobimo

$$\gamma^i \pmod p = \beta^{(p-1)/q} \pmod p$$

in je $a_0 = i$.

Sedaj moramo določiti $a_1, \ldots, a_{c-1}$
(če je $c > 1$). Naj bo

$$\beta_j = \beta\alpha^{a_0+a_1q+\cdots+a_{j-1}q^{j-1}} \pmod p,$$

za $0 \le j \le c-1$. Tokrat velja splošnejša identiteta

$$(\beta_j)^{(p-1)/q^{j+1}} \equiv \alpha^{a_j(p-1)/q} \pmod p,$$

ki jo dokažemo na enak način kot prejšnjo.

Za dani β_j ni težko izračunati a_j, omenim
rekurzijo

$$\beta_{j+1} = \beta_j\alpha^{-a_jq^j} \pmod p.$$

Za dano faktorizacijo števila n je časovna
zahtevnost Pohlig-Hellmanovega algoritma
$O(\sum_{i=0}^{k} c_i(\log n + \sqrt{p_i}))$ grupnih multiplikac

Primer: naj bo $p = 251$, potem je

$$n = p - 1 = 250 = 2 \cdot 5^3.$$

Naj bo $\alpha = 71$ in $\beta = 210$,
torej želimo izračunati $a = \log_{71} 210$.

Modul 2: $\gamma_0 = 1$,

$$\gamma_1 \equiv \alpha^{250/2} \equiv 250 \pmod p$$

in

$$\beta^{250/2} \equiv 250 \pmod p,$$

torej $a_0 = 1$ in $\log_{71} 210 \equiv 1 \pmod 2$.

Modul 5: $\gamma_0 = 1$,

$$\gamma_1 \equiv \alpha^{250/5} \equiv 20 \bmod p$$

in

$$\beta^{250/5} \equiv 149 \pmod p,$$

torej $a_0 = 2$. ...
$a_1 = 4 = \log_{20} 113$ in $a_2 = 2 = \log_{20} 149$,

$\log_{71} 210 \equiv 2 + 4 \cdot 5 + 2 \cdot 5^2 \equiv 72 \pmod{125}$.

Končno nam CRT da $\log_{71} 210 = 197$.

Metoda index calculus

$GF(23)^*$ z generatorem 5.
Izberi bazo 'majhnih' faktorjev: $B = \{-1, 2, 3$
in sestavi tabelo njihovih logaritmov:

elt	-1	2	3
log	11	2	16

Iščemo logaritem elementa β (Las Vegas).
Poišči 'gladko' potenco elementa β,
tj. β^x, ki se da razstaviti na faktorje iz B.

Izračunaj log(13): $13^2 = 169 = 2^3 \iff$
$\log 13^2 = \log 2^3 \iff 2\log 13 \equiv 3\log 2 \iff$
$2\log 13 \equiv 6 \pmod{22}$

Sledi $\log 13 \equiv 3$ ali $14 \pmod{22}$.
Preverimo $\log 13 = 14$.

Izračunaj log(14):
$14^3 = 2^3 7^3 = 2^3 \cdot 21 = 2^3 \cdot (-2) = -2^4$.
$3\log 14 = \log(-2^4) = \log(-1) + \log 2^4 = 11 + 4 \cdot 2 = 19$,
$\log 14 = \frac{19}{3} = 19 \cdot (-7) = (-3)(-7) = 21$.

Izračunaj log(15):
$15^3 = 3^3 \cdot 5^3 = 3^3 \cdot 2 \cdot 5 = (-1) \cdot 2 \cdot 3$,
$3\log 15 = \log(-1) + \log 2 + \log 3 = 11 + 2 + 16 = 29 = 7$,
$\log 15 = \frac{7}{3} = 7(-7) = -49 = -5 = 17$.

Izračunaj log(7):
$7^3 = 49 \cdot 7 = 3 \cdot 7 = 21 = (-1) \cdot 2$,
$3\log 7 = \log(-1) + \log 2 = 11 + 2 = 13$,
$\log 7 = \frac{13}{3} = 13 \cdot (-7) = 63 = -3 = 19$.

Še en primer: $GF(89)^*$ z gen. 3.

tabela logaritmov:

elt	-1	2	3	5
log	44	16	1	70

Izračunaj log(7):
$7^3 = 76 = 2^2 \cdot 19$, $7^5 = 3 \cdot 5^2$,
$5\log 7 = \log 3 + 2\log 5 = 1 + 2 \cdot 70 = 141 = 53$,
$\log 7 = \frac{53}{5} = 53 \cdot (-35) = 81$.

Izračunaj log(53):
$53^3 = 3 \cdot 23$, $53^5 = 2^2 \cdot 17$, $53^7 = 2 \cdot 3^2$,
$7\log 53 = \log 2 + 2\log 3 = 16 + 2 = 8$,
$\log 53 = \frac{18}{7} = 18 \cdot (-25) = 78$.

Metoda index calculus (splošno

1. Izberi bazo faktorjev $\mathcal{B} = \{p_1, \ldots, p_t\}$, tak
da dovolj veliko število elementov grupe G dov
razstaviti v $\mathcal{B}$.

2. Poišči $t + 10$ lineranih zvez z logaritimi el
iz $\mathcal{B}$:

izberi število $k < n$, izračunaj α^k in ga poskusi
kot

$$\alpha^k = \Pi_{i=1}^t p_i^{c_i} \iff k \equiv \sum_{i=1}^t c_i \log p_i \mod$$

3. Sestavi tabelo logaritmov elementov iz $\mathcal{B}$.

4. Izberi naključno število $k \in \{1, \ldots, n\}$,
izračunaj $\beta\alpha^k$ in ga poskusi zapisati kot

$$\beta\alpha^k = \Pi_{i=1}^t p_i^{d_i}.$$

Končno dobimo

$$\log_\alpha \beta = (\sum_{i=1}^t d_i \log_\alpha p_i - k) \mod n.$$

Obstajajo različni slučajni algoritmi za metodo Index calculus. Ob sprejemljivih predpostavkah je njihova časovna zahtevnost za pripravljalno fazo

$$\mathcal{O}\left(e^{1+o(1))\sqrt{\log p \log\log p}}\right),$$

za izračun vsakega posameznega logaritma pa

$$\mathcal{O}\left(e^{1/2+o(1))\sqrt{\log p \log\log p}}\right).$$