Eisensteinova lema. *$p>2$ praštevilo, $p \nmid q \in \mathbb{N}$. Naj bo $A := \{2,4,6,\ldots,p-1\}$ in $r_a := qa \bmod p$ za $a \in A$. Potem je*

$$\left(\frac{q}{p}\right) = (-1)^{\sum_{a\in A} r_a}.$$

Dokaz: Za $a, a' \in A$, $a \neq a'$, ne more veljati $r_a(-1)^{r_a} = r_{a'}(-1)^{r_{a'}}$ oziroma $qa \equiv \pm qa' \pmod p$, saj bi od tod sledilo $a = \pm a'$, kar pa ni mogoče.

Opozorimo še, da so vsa števila $r_a(-1)^{r_a} \bmod p$ soda, torej pretečejo ravno vse elemente množice A.

Od tod dobimo

$$\prod a \equiv (-1)^{\sum r} \prod r \pmod p,$$

očitno pa neposredno iz definicije sledi tudi

$$q^{\frac{p-1}{2}} \prod a \equiv \prod r \pmod p.$$

Torej velja $q^{\frac{p-1}{2}} \equiv (-1)^{\sum r} \pmod p$ in po Eulerjevem kriteriju še

$$\left(\frac{q}{p}\right) \equiv q^{\frac{p-1}{2}} \pmod p. \quad \blacksquare$$

Oglejmo si Eisensteinov *dokaz Gaussovega izreka o kvadratni recipročnosti.* Očitno velja

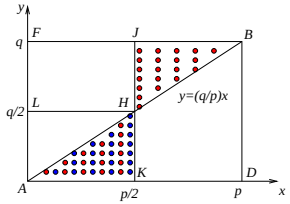
$$\sum qa = p \sum \left\lfloor \frac{qa}{p} \right\rfloor + \sum r\,.$$

Ker so elementi a vsi sodi in je p lih, velja

$$\sum r \equiv \sum \left\lfloor \frac{qa}{p} \right\rfloor \pmod 2$$

in zato iz Eisensteinove leme sledi

$$\left(\frac{q}{p}\right) = (-1)^{\sum \left\lfloor \frac{qa}{p} \right\rfloor}.$$



Vsota $\sum \left\lfloor \frac{qa}{p} \right\rfloor$ je enaka številu celoštevilčnih točk sodo x-koordinato, ki ležijo v notranjosti trikotnika ABD. Sedaj pa si oglejmo točke z x-koordinato večjo od $p/2$. Ker pa je $q-1$ sod, je parnost števila $\left\lfloor \frac{qa}{p} \right\rfloor$ točk z isto x-koordinato pod diagonalo AB enako številu točk z isto sodo x-koordinato nad diagonalo AB.

To pa je po drugi strani enako številu točk pod diagonalo AB z liho x-koordinato $p-a$ (bijektivna korespondenca med točkami s sodo x-koordinato v BHJ in liho x-koordinato v AHK). Od tod sledi, da ima vsota $\sum \left\lfloor \frac{qa}{p} \right\rfloor$ enako parnost kot številu μ celoštevilčnih točk v notranjosti trikotnika AHK, tj.

$$\left(\frac{q}{p}\right) = (-1)^{\mu}.$$

Če zamenjamo p in q, dobimo še število ν celoštevilčnih točk v notranjosti trikotnika AHL, kar nam da

$$\left(\frac{p}{q}\right) = (-1)^{\nu}$$

in skupaj s prejšnjo relacijo Gaussov izrek. $\blacksquare$

Še en Monte Carlo algoritem za testiranje sestavljenosti števil.

Miller-Rabinov test: *testiramo liho število n.*

1. *$n-1 = 2^k m$, kjer je m liho število,*
2. *izberemo naključno naravno število $a < n$,*
3. *izračunamo $b \equiv a^m \pmod n$,*
4. **if** $b \equiv 1 \pmod n$ **then** *n je praštevilo;* **exit;**
5. **for** $i = 0$ **to** $k-1$ **do**
   if $b \equiv -1 \pmod n$ **then** *n je praštevilo;* **exit;**
   else $b \equiv b^2 \pmod n$,
7. *število n je sestavljeno.*

Izrek: *Miller-Rabinov algoritem za problem sestavljenih števil je DA-naklonjen Monte Carlo algoritem.*

Dokaz: Predpostavimo, da algoritem odgovori "n je sestavljeno število" za neko praštevilo p. Potem je $a^m \not\equiv 1 \pmod n$.

Sledi $a^{2^i m} \not\equiv -1 \pmod n$ za $i \in \{0,1,\ldots,k-1\}$.

Ker je $n = 2^k m + 1$ praštevilo, iz Fermatovega izreka sledi

$$a^{2^k m} \equiv 1 \pmod n$$

in je $a^{2^{k-1}m}$ koren od 1 po modulu n.

Iz $x^2 \equiv 1 \pmod n$ oziroma $n | x^2 - 1 = (x-1)(x+1)$ sledi

$$x \equiv 1 \pmod n \quad \text{ali} \quad x \equiv -1 \pmod n$$

oziroma v našem primeru $a^{2^{k-1}m} \equiv 1 \pmod n$. Na isti način pridemo do

$$a^m \equiv 1 \pmod n,$$

kar je protislovje, saj bi algoritem v tem primeru odgovoril "n je praštevilo". $\blacksquare$

Za konec omenimo brez dokaza še, da je verjetnost napake Miller-Rabinovega algoritma kvečjemu 1/4.

Napadi na RSA

Odličen pregledni članek "Twenty Years of Attacks on the RSA kriptosysystem", je objavil Dan Boneh v *Notices of AMS*, Feb. 1999, pp. 203-212.

Mi bomo omenili le nekaj osnovnih napadov.

Če poznamo $\varphi(n)$ in n, dobimo p, q iz naslednjega sistema dveh enačb

$$n = pq \quad \text{in} \quad \varphi(n) = (p-1)(q-1).$$

Odšifrirni eksponent kriptosistema RSA

Trditev: *Vsak algoritem A, ki najde odšifrirni eksponent d, lahko uporabimo kot podprogram v probabilističnem algoritmu, ki najde faktorje števila n.*

Od tod sledi, da iskanje odšifrirnega eksponenta ni nič lažje kot problem faktorizacije.

Opozorilo: če "izgubimo" d, moramo poleg šifrirnega eksponenta zamenjati tudi modul n.

Naj bo $\varepsilon \in [0,1)$. **Las Vegas algoritem** je probabilističen algoritem, ki za dani primer problema, lahko *ne da odgovora* z verjetnostjo ε (se pravi, da konča s sporočilom "ni odgovora"). Če pa algoritem odgovori, potem je *odgovor gotovo pravilen*.

DN: Pokaži, da je povprečno pričakovano število ponovitev algoritma vse dokler ne dobimo odgovora, enako $1/(1-\varepsilon)$ (glej nalogo 4.15).

Če Las Vegas algoritem faktorizira število n z verjetnostjo vsaj ε in ga ponovimo m-krat, potem bo število n faktorizirano z verjetnostjo vsaj $1-\varepsilon^m$.

Trditev sledi iz algoritma, ki uporablja naslednje: za $n = pq$, kjer sta p, q lihi praštevili,

$$x^2 \equiv 1 \pmod{n}, \quad \text{tj.} \quad pq|(x-1)(x+1),$$

dobimo štiri rešitve; dve (trivialni) rešitvi iz enačb

$$x \equiv 1 \pmod{n} \ \text{ in } \ x \equiv -1 \pmod{n}$$

in s pomočjo kitajskega izreka o ostankih iz

$$x \equiv 1 \pmod{p}, \qquad x \equiv -1 \pmod{q}$$

in

$$x \equiv -1 \pmod{p}, \qquad x \equiv 1 \pmod{q}$$

še dve (netrivialni) rešitvi.

Algoritem za faktorizacijo z danim šifr. eksp. d

1. Izberi naključno naravno število $w < n$,
2. izračunaj $x = D(w, n)$,
3. **if** $1 < x < n$ **then exit**(uspeh $x = p$ ali $x = q$)
4. izračunaj $d = A(e, n)$ in zapiši $de - 1 = 2^s r$, r lih,
5. izračunaj $v = w^r \mod n$,
6. **if** $v \equiv 1 \pmod{n}$ **then exit**(neuspeh)
7. **while** $v \not\equiv 1 \pmod{n}$ **do** $v_0 = v$, $v = v^2 \mod n$
8. **if** $v_0 \equiv -1 \pmod{n}$ **then exit**(neuspeh)
   else izračunaj $x = D(v_0 + 1, n)$
   (uspeh: $x = p$ ali $x = q$) .

Naključne napake
(Boneh, DeMillo in Lipton, 1997)

Če uporabimo CRT in pride pri samo enem izmed C_p in C_q do napake, npr. C_p je pravilen, $\hat{C}_q$ pa ni, potem je $\hat{C} = t_p C_p + t_q \hat{C}_q$ očitno nepravilen podpis, saj je $\hat{C}^e \neq M \bmod N$. Vendar pa je

$$\hat{C}^e = M \bmod p, \quad \text{medtem, ko je} \quad \hat{C}^e \neq M \bmod q$$

in nam $D(n, \hat{C}^e - M)$ odkrije netrivialni faktor števila n.

Rabinov kriptosistem

Temelji na tem, da je težko najti faktorizacijo produkta dveh velikih praštevil p in q.

$n = pq, \quad p \neq q, \quad p, q \equiv 3 \pmod{4}, \quad \mathcal{P} = \mathcal{C} = \mathbb{Z}_n$

$$\mathcal{K} = \{(n, p, q, B);\ 0 \leq B \leq n-1\}$$

Za izbrani ključ $K = (n, p, q, B)$ naj bo:

$$e_K(x) = x(x+B) \bmod n,$$

$$d_K(y) = \sqrt{y + B^2/4} - B/2.$$

Javni ključ *je* (n, B), **zasebni ključ** *pa* (p, q).

Trditev: *Naj bo $\omega^2 \equiv 1 \pmod{n}$ netrivialen koren (kongruenca ima 4 rešitve: $1, -1$ in še dve netrivialni), in $x \in \mathbb{Z}_n$, potem velja:*

$$e_K(\omega(x + B/2) - B/2) = e_K(x).$$

Imamo 4 čistopise, ki ustrezajo tajnopisu $e_K(x)$:

$$x, \quad -x - B, \quad \omega\left(x + \frac{B}{2}\right) \ \text{ in } \ -\omega\left(x + \frac{B}{2}\right).$$

V splošnem se ne da ugotoviti, kateri je pravi.

Odšifriranje

Imamo tajnopis y in iščemo x, ki zadošča naslednji enačbi:

$$x^2 + Bx \equiv y \pmod{n}.$$

Poenostavimo: $x = x_1 - B/2$,

$$x_1^2 \equiv y + B^2/4 \pmod{n}, \qquad C = y + B^2/4.$$

Iščemo kvadratne korene enačbe $\boxed{x_1^2 \equiv C \pmod{n}}$.

To je ekvivalentno sistemu:

$$\boxed{\begin{array}{l} x_1^2 \equiv C \pmod{p} \\ x_1^2 \equiv C \pmod{q} \end{array}}$$

Eulerjev izrek:

$$C^{(p-1)/2} \equiv 1 \pmod{p}$$

$\downarrow$

predpostavka: $p \equiv 3 \pmod{4}$

$$\Rightarrow \quad (\pm C^{(p+1)/4})^2 \equiv C \pmod{p}$$

$$\boxed{\begin{array}{l} x_1 \equiv x_{1,2} \pmod{p} \\ x_1 \equiv x_{3,4} \pmod{q} \end{array}}$$

$\Rightarrow$ korena prve enačbe sta:

$$x_{1,2} = \pm C^{(p+1)/4}$$

korena druge enačbe pa:

$$x_{3,4} = \pm C^{(q+1)/4}$$

$\Downarrow$ KIO

x_1, x_2, x_3, x_4

Primer: $n = 77 = 7 \cdot 11, \quad B = 9$

$e_K(x) = x^2 + 9x \bmod 77$

$$d_K(y) = \sqrt{y + B^2/4} - B/2 = \sqrt{1+y} - 43 \pmod{77}$$

Tajnopis: $y = 22$. Poiskati moramo rešitve:

$$\boxed{\begin{array}{l} x^2 \equiv 23 \pmod{7} \\ x^2 \equiv 23 \pmod{11} \end{array}} \qquad \begin{array}{l} (x \equiv \pm 4 \pmod{7}) \\ (x \equiv \pm 1 \pmod{11}) \end{array}$$

Dobimo štiri sisteme dveh enačb z dvema neznankama, npr.:

$$x \equiv 4 \pmod{7}, \quad x \equiv 1 \pmod{11}$$

Po kitajskem izreku o ostankih velja:

$$x = 4 \cdot 11 \cdot (11^{-1} \bmod 7) + 1 \cdot 7 \cdot (7^{-1} \bmod 11).$$

Vse rešitve so:

$x_1 \equiv 67 \pmod{77}, \quad x_2 \equiv 10 \pmod{77},$
$x_3 \equiv 32 \pmod{77}, \quad x_4 \equiv -32 \pmod{77}.$

Odšifrirani tekst je:

$$\begin{aligned} d_K(y) = \; & 67 - 43 \bmod 77 = 24 \\ & 10 - 43 \bmod 77 = 44 \\ & 32 - 43 \bmod 77 = 66 \\ & 45 - 43 \bmod 77 = 2, \end{aligned}$$

vse štiri rešitve pa se zašifrirajo v 22.

Varnost Rabinovega kriptosistema

Hipotetični algoritem A za dekripcijo Rabinovega kriptosistema lahko uporabimo kot podprogram v algoritmu tipa Las Vegas za faktorizacijo števila n z verjetnostjo vsaj $1/2$.

1. Izberemo r, $1 \le r \le n-1$,
2. $y := r^2 - B^2/4 \bmod n \quad (y = e_K(r - B/2))$,
3. $x := A(y)$,
4. $x_1 := x + B/2 \qquad (x_1^2 \equiv r^2 \pmod{n})$,
5. če velja $x_1 \equiv \pm r \pmod{n}$, potem ni odgovora, sicer ($x_1 \equiv \pm\omega \cdot r \pmod{n}$, kjer je $\omega \equiv 1 \pmod{n}$ netrivialni koren) $D(x_1 + r_1, n) = p$ (ali q).

V zadnjem primeru $n \mid (x_1 - r)(x_1 + r)$, vendar $n \nmid (x_1 - r)$ in $n \nmid (x_1 + r) \quad \Rightarrow D(x_1 + r, n) \neq 1$.

Verjetnost, da uspemo v enem koraku:

Def: $r_1 \sim r_2 \Leftrightarrow r_1^2 \equiv r_2^2 \pmod{n}$ $(r_1, r_2 \neq 0)$.
To je ekvivalenčna relacija, ekvivalenčni razredi v $Z_n \backslash \{0\}$ imajo moč 4: $[r] = \{\pm r, \pm\omega r\}$.
Vsak element iz [r] nam da isto vrednost y.
Podprogram A nam vrne x, $[x] = \{\pm x, \pm\omega x\}$,
$r = \pm x$: 4 ni odgovora $\quad r = \pm\omega x$: dobimo odgovor.

Ker izberemo r slučajno, je vsaka od teh možnosti enako verjetna $\Rightarrow$ verjetnost, da uspemo, je $1/2$.

Algoritmi za faktorizacijo števil

Poskušanje
Število n delimo z vsemi lihimi števili do $\sqrt{n}$:

$i := 3$,
until $i \le \sqrt{n}$ **repeat**
if $i \mid n$, potem smo našli faktor,
else $i := i + 2$.

Algoritem je uporaben za manjše n (npr. $n \le 10^{12}$).
Časovna zahtevnost za k bitov je $2^{k/2-1}$ deljenj.

Metoda $p - 1$ (Pollard, 1974)

Podatki: n (lih, želimo faktorizirati) in B (meja)

Algoritem temelji na naslednjem preprostem dejstvu:

če je p praštevilo, ki deli n, in za vsako praštevilsko potenco q, ki deli $p-1$, velja $q \le B$, potem $(p-1)|B$!

Primer: $B = 9$, $p = 37$, $p - 1 = 36 = 2^2 \cdot 3^2$

$2^2 \le B, 3^2 \le B \Rightarrow 2^2 \cdot 3^2 | 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot 7 \cdot 8 \cdot 9$

Algoritem

Podatki: n, B
1. $a := 2$
2. $j = 2, ..., B$ $\quad (a \equiv 2^{B!} \pmod n)$
   $a := a^j \bmod n$ $\quad (\Rightarrow a \equiv 2^{B!} \pmod p)$
3. $d = D(a-1, n)$ $\quad$ (Fermat: $2^{p-1} \equiv 1 \pmod p$)
4. Če velja $1 < d < n$, je d faktor števila n (saj $p|d$) sicer ni uspeha (to se zgodi, kadar je $d=1$).

Če $B \geq \sqrt{n}$, vedno uspemo, vendar algoritem ni učinkovit.

Časovna zahtevnost

- $B-1$ potenciranj po modulu n, za vsako rabimo $2\log_2 B$ množenj po modulu n,
- največji skupni delitelj z Evklid. alg.: $\mathcal{O}((\log n)^3)$.

Skupaj $\mathcal{O}(B \log B (\log n)^2 + (\log n)^3)$, kar pomeni, da je za $B \approx (\log n)^i$ algoritem polinomski.

Primer: $n=143$, $B=4$, $a \equiv 2^{2\cdot 3\cdot 4} \equiv 131 \pmod{143}$. Torej je $a-1=130$ in od tod $D(130,143)=13$.

Za varen RSA izberemo $p = 2p_1+1$ in $q = 2q_1+1$, kjer sta p_1 in q_1 praštevili.

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

$$8340934156^2 \equiv 3\times 7 \pmod n \qquad a_1 = (0,1,0,1,0,0)$$
$$12044942944^2 \equiv 2\times 7\times 13 \pmod n, \quad a_2 = (1,0,0,1,0,1)$$
$$2773700011^2 \equiv 2\times 3\times 13 \pmod n, \quad a_3 = (1,1,0,0,0,1)$$

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 $x_j^2 \bmod n$ 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 praksi

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	$O(e^{(1.92+o(1))(\ln n)^{1/3}(\ln\ln n)^{2/3}})$

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

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

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

leto	stevilo	bitov	metoda	opombe
1903	$2^{67}-1$	67		F. Cole (3 leta ob ned.)
1988		250	QS	100 rac., e-posta
1994	RSA-129	425	QS	1600 rac. 8 mesecev
1999	RSA-155	512	NFS	300 del.p.+Cray; 5 mes.
2002	RSA-158	524	NFS	30 del.p.+Cray; 3 mes
2003	RSA-174	576	NFS	
2005	RSA-200	663	NFS	(55 let na eni del.p.)

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/Maple hitro najde en faktor:

1265011073

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

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
- 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, **zasebnega** 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 temeljijo, 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 krivuljami**
  (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.

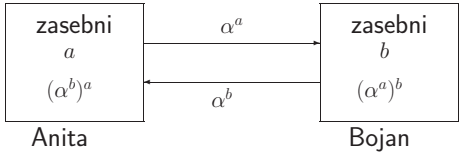
Osnovna razloga za uporabo različnih grup:

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

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

1. Izmenjava ključev

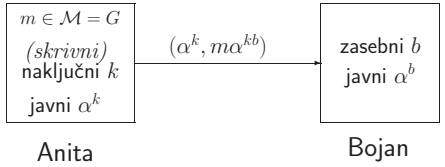
(Diffie-Hellman)



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)



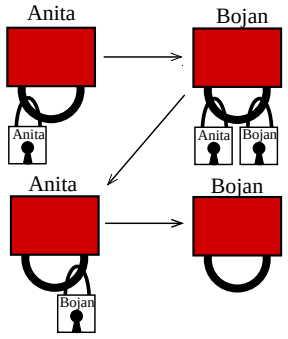
Če je $(y_1, y_2) = e_K(m, k) = (\alpha^k, m\alpha^{kb})$, potem je odšifriranje definirano z $d_K(y_1, y_2) = y_2(y_1^b)^{-1}$.

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

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

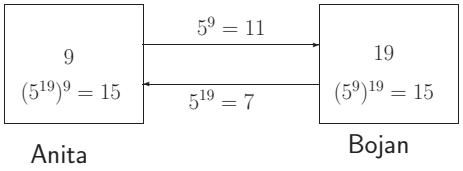
Primer: $12 + 20 = 32 = 9$, $8 \cdot 9 = 72 = 3$.

Multiplikativna grupa GF(23)*

Elementi GF(23)* so elementi GF(23)\{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, da je red elementa α velik (s tem pa je velika tudi log tabela).

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.
$5 \times 18 = 21, \quad 5^2 \times 18 = 13, \quad 5^3 \times 18 = 19.$
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 elementov $3^0, 3^{10}, 3^{20}, \cdots, 3^{80}$ 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 $3 \times 36, 3^2 \times 36, \cdots$, vse dokler ne dobiš elementa iz tabele.
$3 \times 36 = 19$, $3^3 \times 36 = 82$, $3^5 \times 36 = 26$, $3^2 \times 36 = 57$, $3^4 \times 36 = 68$, $3^6 \times 36 = 78$.
Iz tabele preberemo $\log(3^6 \times 36) = \log 78$.
Sledi $6 + \log 36 = 40$ 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 $O(m)$ časa.

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

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

Dokažimo slednjo kongruenco:

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

Najprej torej izračunamo

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

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

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

vse dokler ne dobimo

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

in je $a_0 = i$.

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

$$\beta_j = \beta \alpha^{a_0 + a_1 q + \cdots + a_{j-1} q^{j-1}} \mod p,$$

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, omenimo pa še rekurzijo

$$\beta_{j+1} = \beta_j \alpha^{-a_j q^j} \mod p.$$

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

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: $\gamma_0 = 1$,

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

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 $\log_{71} 210 = 197$.