KRYPTOLOGIA

Keijo Ruohonen

2004

Sisältö

Esipuhe

Tämä moniste on tarkoitettu TTY:n kurssin ”73260 Kryptologia” perusmateriaaliksi. Monisteessa käydään läpi tärkeimmät nykyisen tiedonsalauksen tarvitsemat matemaattiset taustat sekä esitellään niiden soveltamista kryptauksessa sekä erilaisissa protokollissa. Esitys pohjautuu paljolti kirjoihin MOLLIN, STINSON ja SALOMAA.

Matematiikan ja kryptologian liitto on vanha, mutta tuli vahvemmin esille oikeastaan vasta toisen maailmansodan tehokkaiden kryptausmenetelmien ja niiden murtamisen yhteydessä (sitten kun nämä tulivat julkisiksi). Asian yleinenkin kiinnostavuus ilmenee vaikkapa siitä kirjoitettujen suurelle yleisölle tarkoitettujen (osin) fiktiivisten kirjojen runsaudesta.[1]

Kokonaan uuden vauhdin koko ala sai 1970-luvulla, jolloin otettiin käyttöön täysin avoin, nopea ja tehokas tietokoneille tarkoitettu kryptausmenetelmä DES sekä esiteltiin vallankumouksellinen julkisen avaimen kryptauksen idea. Sen jälkeen kryptologian sekä myös sen tarvitseman matematiikan—lähinnä eräiden lukuteorian ja algebran alojen—kehitys on ollut tavattoman nopeaa. Voidaankin sanoa, että lukuteorian ja algebran viimeaikainen suosio johtuu nimenomaan kryptologiasta. Usein tähän liitetään myös lähinnä teoreettiseen tietojenkäsittelytieteeseen kuuluva suhteellisen uusi laskennallisen vaativuuden teoria, mutta totuuden nimessä pitänee todeta, ettei sillä ole ollut kryptologiassa kovinkaan suurta merkitystä. Otollisia kryptauksessa käyttökelpoisia matemaattisia probleemoja kun ovat sellaiset, joita huippumatemaatikot ovat tutkineet jo niin kauan, että vain todella vaikeasti löydettävät teorian tulokset ovat avoinna. Kryptauksen murtaminen edellyttää silloin myös huomattavaa teoreettista läpimurtoa. Tällaisia probleemoja löytyy runsaasti nimenomaan lukuteorian ja diskreetin algebran aloilta.

Lukuteorian ja algebran tulokset ja algoritmit esitetään tässä monisteessa omissa luvuissaan ja pykälissään, useassa osassa. Keskeistä on lukuteorian ja algebrallisten struktuurien probleemojen lajittelu laskennallisesti ”helppoihin” ja ”vaikeisiin”. Edellisiä tarvitaan kryptauksessa ja dekryptauksessa sekä myös kryptosysteemien pystytyksessä, jälkimmäiset taas takaavat kryptosysteemin varmuuden.

Klassisia kryptosysteemejä—joihin myös DES on tyypiltään luettava—esitellään monisteessa vain muutamia (lisätietoa niistä löytyy vaikkapa viitteistä BAUER ja SALOMAA), pääpaino on moderneissa julkisen avaimen menetelmissä. Tämä ei suinkaan ole osoitus siitä, että niillä ei olisi käyttöä. Vaikka vanhojen klassisten menetelmien merkitys on kadonnut nopeasti[2], uusia klassistyyppisiä menetelmiä kehitetään jatkuvasti ja niillä on hyvin tärkeä rooli nopeassa massakryptauksessa. Monisteessa jätetään vähälle huomiolle myös monissa sovelluksissa niin tärkeä vuokryptaus. Yhden kurssin puitteissa aika on rajallinen. Kokonaan oma lukunsa on vielä kryptosysteemien oikea implementointi ja käyttö, johon tällaisessa matemaattisessa kurssissa ei voi kovinkaan paljon puuttua. Hyväkin kryptosysteemi kun voidaan helposti tehdä tehottomaksi huonolla implementoinnilla ja huolettomalla käytöllä.[3]

Kiitän tutkija Erkko Lehtosta, jonka kommentit ja tarkka silmä ovat parantaneet esitystä ja vähentäneet painovirheitä.

Keijo Ruohonen

[1] Esimerkkeinä Robert Harrisin *Enigma* sekä Neal Stephensonin mainio *Cryptonomicon.*

[2] Esimerkkinä mainittakoon vaikkapa se, että Yhdysvaltain maavoimien kenttämanuaali FM 34-40-2: *Basic Cryptanalysis* on nykyään julkisesti saatavana verkossa. Viite BAUER sisältää myös aikaisemmin salaiseksi luokiteltua materiaalia.

[3] Erinomainen tähän liittyvä teos on Bruce Schneierin *Secrets and Lies. Digital Security in a Networked World.*

Luku 1

JOHDANTO

Viestin *kryptauksella* tarkoitetaan sen salaamista siten, että satunnainen lukija tai salakuuntelija ei saa selville mitään viestin sisällöstä eli hän ei pysty *murtamaan* kryptausta. Alkuperäistä selväkielistä viestiä kutsutaan *selvätekstiksi* ja kryptattua taas *kryptotekstiksi*. Kryptauksessa tarvitaan ns. *avain,* ts. eräänlainen yleensä varsin mutkikas parametri, jolla kryptausta voidaan muuttaa. Jos kryptausmenettely pysyy pitkään tarkalleen samana, lisääntyvät murtomahdollisuudet käytännössä tuntuvasti. Eri käyttäjät tarvitsevat luonnollisesti vielä omat avaimensa.

Viestin vastaanottaja *dekryptaa* viestin, mitä varten hän tarvitsee oman avaimensa. Salakuuntelijalle sekä kryptausavain että dekryptausavain ovat arvokkaita, edellistä käyttäen hän pystyy lähettämään kryptattuja valeviestejä, jälkimmäisellä taas purkamaan (eli dekryptaamaan) viestit. Symmetrisissä kryptosysteemeissä avaimet ovat lisäksi usein samat.

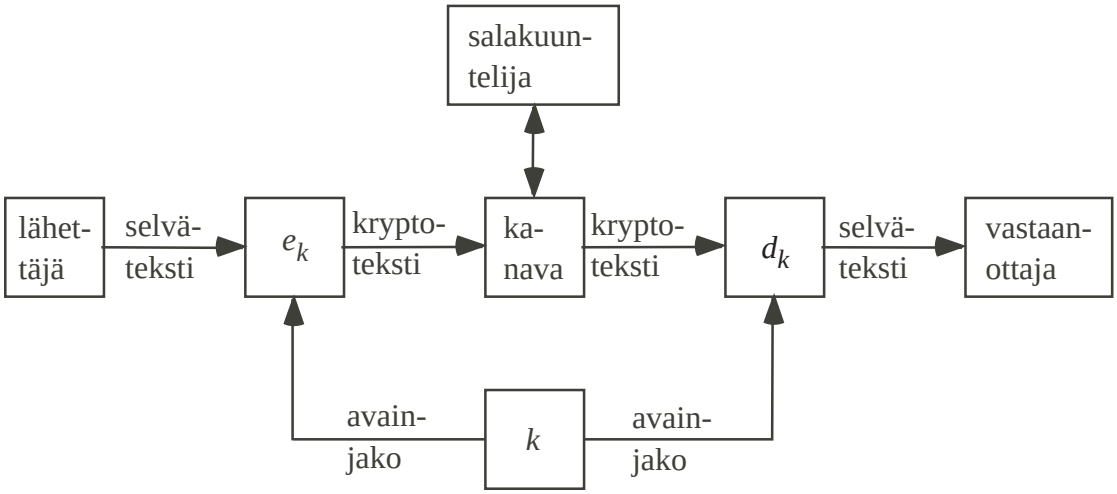
Kryptausmenettely voi kryptata symbolivirtaa jatkuvasti (*vuokryptaus*) tai lohkoihin jaettuna (*lohkokryptaus*). Lohkokryptauksessa lohkot voivat joissain tapauksissa olla eripituisia, kuitenkaan tiettyä lohkon maksimipituutta ei saa ylittää. Yleensä kuitenkin lohkot ovat samanpituisia. Jatkossa tarkastellaan vain lohkokryptausta. Tällöin riittää selvittää mielivaltaisen viestilohkon kryptaaminen/dekryptaaminen ja selvätekstinä voidaan pitää yhtä (mielivaltaista) viestilohkoa ja kryptotekstinä mielivaltaista viestilohkon kryptattua versiota.

Kryptausmenettely on *symmetrinen,* jos kryptaus- ja dekryptausavaimet ovat samat tai ainakin helposti saatavissa toinen toisistaan. *Epäsymmetrisessä* kryptauksessa kryptausavaimesta ei voi millään pienellä työmäärällä saada dekryptausavainta. Niinpä kryptausavain voidaan tällöin pitää jopa julkisena dekryptausavaimen pysyessä salaisena (*julkisen avaimen kryptaus;* vastaavasti symmetristä kryptausta kutsutaan *salaisen avaimen kryptaukseksi*). Symmetrisessä kryptauksessa pulmana on avaimien välitys salassa kaikille osapuolille, avaimia pitää vielä uusiakin aika ajoin.

Symmetrinen kryptaus (käyttäen samaa avainta) voidaan karakterisoida ns. *kryptosysteeminä* eli viisikkona (P, C, K, E, D), missä

- P on äärellinen *viestiavaruus* (selvätekstit).
- C on äärellinen *kryptotekstiavaruus.*
- K on äärellinen *avainavaruus.*
- kutakin avainta $k \in K$ kohti on *kryptausfunktio* $e_k \in E$ ja *dekryptausfunktio* $d_k \in D$. E on mahdollisten *kryptausfunktioiden avaruus* ja D mahdollisten *dekryptausfunktioiden avaruus.*
- jokaiselle viestille (lohkolle) w ja avaimelle k on $d_k(e_k(w)) = w$.

Ilmeisesti kryptausfunktion pitää olla *injektiivinen*, ts. se ei kryptaa kahta eri selvätekstiä samaksi kryptotekstiksi. Kryptauksessa voi kuitenkin olla mukana satunnaisuutta, jolloin kryptausfunktio voi eri kerroilla kryptata saman selvätekstin eri kryptotekstiksi (kyseessä ei siis tällöin ole varsinaisesti matemaattinen funktio). Injektiivisyydestäkin voidaan toisinaan tinkiä, jos kryptotekstiä vastaavia selvätekstejä on rajallinen määrä ja niistä löytyy se oikea pienellä vaivalla.



Kutakuinkin kaikki laajemmin käytetyt kryptausmenetelmät perustuvat lukuteorian tai algebran (ryhmäteoria, äärelliset kunnat, kommutatiivinen algebra) tuloksiin. Näihin perehdytään sitä mukaa kuin ne tulevat esille, tarvittavassa määrin.

Luku 2

EKSKURSIO LUKUTEORIAAN 1

2.1 Jaollisuus, tekijät, alkuluvut

Tietyt lukuteorian[1] käsitteet ja tulokset tulevat hyvin usein esille kryptologiassa, vaikka menettely sinänsä ei liittyisikään lukuteoriaan. Kaikkien kokonaislukujen joukkoa merkitään $\mathbb{Z}$:lla. Ei-negatiivisten kokonaislukujen eli ns. *luonnollisten lukujen* joukkoa $\{0, 1, 2, \dots\}$ merkitään $\mathbb{N}$:llä.

Kokonaislukujen yhteen- ja kertolasku ovat tuttuja vaihdannaisia ja liitännäisiä laskuoperaatioita, joiden identiteettialkiot ovat 0 ja 1. Muista myös liitäntälaki $x(y + z) = xy + xz$ sekä vastaluvun ja vähennyslaskun määrittelyt: $-x = (-1)x$ ja $x - y = x + (-1)y$. Kokonaislukujen *jakolasku* tarkoittaa seuraavanlaista operaatiota: Jaettaessa kokonaisluku x (*jaettava*) kokonaisluvulla $y \neq 0$ (*jakaja*) x saatetaan muotoon

$$x = qy + r,$$

missä kokonaisluku r on *jakojäännös*, joka toteuttaa ehdon $0 \leq r < |y|$. Kokonaisluku q on *osamäärä*. Lisäämällä x:ään toistuvasti $-y$:tä tai y:tä nähdään, että x on mahdollista kirjoittaa ko. muotoon. Jos voidaan kirjoittaa

$$x = qy,$$

missä q on kokonaisluku, sanotaan, että x *on jaollinen* y*:llä* tai että y *jakaa* x*:n* tai että y *on* x*:n tekijä*, merkitään $y \mid x$. Kokonaisluvun x ns. *triviaalit tekijät* ovat ± 1 ja $\pm x$. Mahdolliset muut tekijät ovat *ei-triviaaleja.*

Seuraavat jakolaskun ja jaollisuuden ominaisuudet ovat melko ilmeisiä:

(1) 0 on jaollinen millä tahansa kokonaisluvulla, mutta jakaa vain itsensä.

(2) ± 1 jakaa kaikki kokonaisluvut, mutta on jaollinen vain itsellään.

(3) Jos $y \mid x$ ja $x \neq 0$, niin $|y| \leq |x|$.

(4) Jos $x \mid y$ ja $y \mid z$, niin myös $x \mid z$.

(5) Jos $x \mid y$ ja $x \mid z$, niin myös $x \mid y \pm z$.

[1] *Lukuteoria* on nimenomaan kokonaislukujen teoriaa. Tosin lukuteoriasta on eri suuntiin laajenevia versioita. Mukaan voidaan ottaa algebralliset luvut, jolloin päästään *algebralliseen lukuteoriaan* (jolla on muuten paljonkin käyttöä kryptologiassa, ks. esimerkiksi KOBLITZ). Toisaalta lukuteoriaa voidaan tutkia käyttämällä muita matematiikan formalismeja. Esimerkiksi *analyyttinen lukuteoria* tutkii kokonaislukuja käyttäen matemaattisen analyysin menetelmiä (integraaleja, sarjoja, jne.).

(6) Jos $x \mid y$ ja z on kokonaisluku, niin $x \mid yz$.

(7) Jakolaskun tulos on yksikäsitteinen. Jos nimittäin

$$x = q_1 y + r_1 = q_2 y + r_2,$$

missä q_1, q_2, r_1, r_2 ovat kokonaislukuja ja $0 \leq r_1, r_2 < |y|$, niin y jakaa $r_1 - r_2$:n. Koska $|r_1 - r_2| < |y|$, tästä seuraa että $r_1 = r_2$ ja edelleen että $q_1 = q_2$.

Kokonaislukua, jolla on vain triviaalit tekijät, kutsutaan *jaottomaksi*. Jaoton kokonaisluku on *alkuluku*[2], jos se on ≥ 2. Ensimmäiset alkuluvut ovat

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, \ldots$$

2 on ainoa parillinen alkuluku. Eräs perustehtävä on testata onko luonnollinen luku alkuluku vai ei. Kokonaisluku, joka on ≥ 2 eikä ole alkuluku, on *yhdistetty luku.*

Lause 2.1. *Jos kokonaisluku x on itseisarvoltaan ≥ 2, niin sillä on alkulukutekijä eli ns.* alkutekijä.

Todistus. Jos $|x| \geq 2$, alkutekijä p löytyy seuraavalla algoritmilla:

1. Asetetaan $z \leftarrow x$.
2. Jos z on jaoton, tulostetaan $p = |z|$ ja lopetetaan.
3. Jos z ei ole jaoton, etsitään ei-triviaali tekijä u. Asetetaan $z \leftarrow u$ ja mennään kohtaan 2.

Menettely pysähtyy, sillä kohdan 3. kautta mentäessä aina $|z|$ pienenee ja on lopulta alkuluku, jolloin tullaan ulos kohdasta 2. □

Seuraus. *Alkulukuja on äärettömän monta.*

Todistus. Äärettömän pitkä jono alkulukuja voidaan saada aikaan seuraavalla jo muinaisten kreikkalaisten tuntemalla menettelyllä (joka ei tietenkään tuota kaikkia alkulukuja):

1. Asetetaan $\mathcal{P} \leftarrow 2$. (Tässä $\mathcal{P}$ on jonomuuttuja.)
2. Jos $\mathcal{P} = p_1, \ldots, p_n$, niin lasketaan $x = p_1 \cdots p_n + 1$. Huomaa, ettei mikään jonon $\mathcal{P}$ alkuluvuista jaa x:ää (jakolaskun yksikäsitteisyys).
3. Lauseen 2.1 mukaisesti x:llä on alkutekijä p, joka ei siis kuitenkaan ole mikään jonon $\mathcal{P}$ alkuluvuista. Etsitään jokin p, asetetaan $\mathcal{P} \leftarrow \mathcal{P}, p$ ja palataan kohtaan 2.

□

Alkulukuihin liittyviä perustehtäviä ovat mm. seuraavat:

(1) Laske (suuruusjärjestyksessä) n:s alkuluku.

(2) Laske (suuruusjärjestyksessä) n ensimmäistä alkulukua.

(3) Laske suurin (vast. pienin) alkuluku, joka on $\leq x$ (vast. $\geq x$).

(4) Laske alkuluvut, jotka ovat $\leq x$.

[2] Alkulukujen joukkoa merkitään toisinaan $\mathbb{P}$:llä.

Lause 2.2. *Kokonaisluku $x \neq 0$ voidaan etumerkkiä vaille kirjoittaa alkulukujen tuloksi, ns.* tekijöihinjako.

Todistus. Seuraava menettely tuottaa jonon alkulukuja, joiden tulo on $= \pm x$:

1. Asetetaan $\mathcal{T} \leftarrow \mathrm{NULL}$ (tyhjä jono).
2. Jos $x = \pm 1$, tulostetaan $\mathcal{T}$ ja lopetetaan. (Muista, että tyhjä tulo on $= 1$.)
3. Etsitään jokin x:n alkutekijä p (Lause 2.1), jolloin $x = py$. Asetetaan $\mathcal{T} \leftarrow \mathcal{T}, p$ sekä $x \leftarrow y$ ja mennään kohtaan 2.

Menettely pysähtyy, koska aina kohdan 3. kautta mentäessä $|x|$ pienenee ja on lopulta $= 1$, jolloin tullaan ulos kohdasta 2. (Erityisesti tulostus on tyhjä jono, jos $x = \pm 1$.) □

Myöhemmin todetaan vielä, että ko. tekijöihinjako on (tekijöiden järjestystä lukuunottamatta) yksikäsitteinen (ks. pykälä 2.3). Luonnollisesti eräs perustehtävä on kokonaisluvun tekijöihinjaon etsiminen. Laskennallisesti tämä on varsin vaativaa, ks. pykälä 6.3.

2.2 Kokonaisluvun esitys eri kannoissa

Tavallisin tapa esittää kokonaisluku on käyttää tuttua *desimaaliesitystä* eli 10-kantaista esitystä. Myös 2-kantainen eli *binääriesitys* on usein esillä, samoin 8-kantainen *oktaaliesitys* tai 16-kantainen *heksadesimaaliesitys.* Yleisen kannan esityksen antaa

Lause 2.3. *Jos $k \geq 2$, niin jokainen positiivinen kokonaisluku x voidaan esittää yksikäsitteisesti muodossa*

$$x = a_n k^n + a_{n-1} k^{n-1} + \cdots + a_1 k + a_0,$$

missä $0 \leq a_0, a_1, \ldots, a_n \leq k - 1$ ja $a_n > 0$. (Kyseessä on x:n k-kantainen esitys, $n + 1$ on esityksen pituus.*)*

Todistus. Esityksen eli jonon $a_n, a_{n-1}, \ldots, a_0$ antaa seuraava algoritmi:

1. Asetetaan $\mathcal{K} \leftarrow \mathrm{NULL}$ (tyhjä jono).
2. Jaetaan x kantaluvulla k: $x = qk + r$ (osamäärä q, jakojäännös r). Asetetaan $\mathcal{K} \leftarrow r, \mathcal{K}$ ja $x \leftarrow q$.
3. Jos $x = 0$, tulostetaan $\mathcal{K}$ ja lopetetaan. Muussa tapauksessa mennään kohtaan 2.

Koska kohdassa 2. aina x pienenee, menettely pysähtyy lopulta kohtaan 3.

k-kantainen esitys on yksikäsitteinen. Jos nimittäin

$$x = a_n k^n + a_{n-1} k^{n-1} + \cdots + a_1 k + a_0 = b_m k^m + b_{m-1} k^{m-1} + \cdots + b_1 k + b_0,$$

missä $0 \leq a_0, a_1, \ldots, a_n, b_0, b_1, \ldots, b_m \leq k - 1$ ja $a_n, b_m > 0$ ja $n \geq m$, niin ensinnäkin voidaan päätellä, että $n = m$ (eli että esityksen pituus on yksikäsitteinen). Jos olisi $n > m$, niin olisi myös

$$\begin{aligned} b_m k^m + b_{m-1} k^{m-1} + \cdots + b_1 k + b_0 &\leq (k-1)k^m + (k-1)k^{m-1} + \cdots + (k-1)k + k - 1 \\ &= k^{m+1} - 1 < k^{m+1} \leq k^n \\ &\leq a_n k^n + a_{n-1} k^{n-1} + \cdots + a_1 k + a_0, \end{aligned}$$

mikä on ristiriita. Siispä $n = m$. Samalla tavoin voidaan todeta, että $a_n = b_n$. Jos nimittäin vaikkapa $a_n > b_n$, niin

$$\begin{aligned} b_n k^n + b_{n-1}k^{n-1} + \cdots + b_1 k + b_0 &\leq (a_n - 1)k^n + (k-1)k^{n-1} + \cdots + (k-1)k + k - 1 \\ &= a_n k^n - 1 < a_n k^n + a_{n-1}k^{n-1} + \cdots + a_1 k + a_0, \end{aligned}$$

mikä on taas ristiriita. Edelleen voidaan samalla tavoin näyttää, että $a_{n-1} = b_{n-1}$, jne. □

Luvun 0 esitys on periaatteessa tyhjä jono missä tahansa kannassa. Koska tämä ei tietenkään käy, sovitaan, että 0:n esitys on 0. Konversio eri kantaesitysten välillä (ns. *kantamuunnos* eli *radix-muunnos*) on tietysti kokonaislukujen perustehtäviä.

Lause 2.4. *Positiivisen kokonaisluvun x k-kantaisen esityksen pituus on*

$$\lfloor \log_k x \rfloor + 1 = \lceil \log_k (x+1) \rceil,$$

missä $\log_k$ on k-kantainen logaritmi[3].

Todistus. Jos x:n k-kantainen esitys on $x = a_n k^n + a_{n-1}k^{n-1} + \cdots + a_1 k + a_0$, niin pituus on $s = n + 1$. Ilmeisesti $x \geq k^n$ ja toisaalta

$$x \leq (k-1)k^n + (k-1)k^{n-1} + \cdots + (k-1)k + k - 1 = k^{n+1} - 1 < k^{n+1}.$$

Koska siis $k^{s-1} \leq x < k^s$, niin $s - 1 \leq \log_k x < s$ eli

$$s = \lfloor \log_k x \rfloor + 1.$$

Toisaalta myös $k^{s-1} < x + 1 \leq k^s$, joten $s - 1 < \log_k (x+1) \leq s$ eli

$$s = \lceil \log_k (x+1) \rceil.$$

□

2.3 Suurin yhteinen tekijä ja pienin yhteinen jaettava

Kokonaislukujen x ja y *suurin yhteinen tekijä* (*s.y.t.*) on suurin kokonaisluku d, joka jakaa molemmat luvut, merkitään

$$d = \mathrm{syt}(x, y).$$

S.y.t. on olemassa, jos ainakin toinen luvuista x ja y on $\neq 0$. Huomaa, että s.y.t. on positiivinen. (Usein määritellään kuitenkin $\mathrm{syt}(0,0) = 0$.) Jos $\mathrm{syt}(x, y) = 1$, sanotaan että luvuilla x ja y *ei ole yhteisiä tekijöitä* tai että ne ovat *keskenään jaottomia*.

Lause 2.5. (Bezout'n lause) *Kokonaislukujen x ja y (joista ainakin toinen on $\neq 0$) s.y.t. d voidaan kirjoittaa muotoon*

$$d = c_1 x + c_2 y \quad \textit{(ns.} \text{ Bezout'n muoto}\textit{)},$$

missä c_1 ja c_2 ovat kokonaislukuja (ns. Bezout'n kertoimet*). Lisäksi, jos $x, y \neq 0$, voidaan olettaa, että $|c_1| \leq |y|$ ja $|c_2| \leq |x|$.*

[3]Muistathan, että logaritmien kannan vaihto sujuu kaavalla $\log_k x = \ln x / \ln k$. Tässä $\lfloor x \rfloor$ tarkoittaa x:n ns. *pohjaa*, ts. suurinta kokonaislukua, joka on $\leq x$. Vastaavasti $\lceil x \rceil$ tarkoittaa reaaliluvun x ns. *kattoa* eli pienintä kokonaislukua, joka on $\geq x$.

Todistus. Bezout'n muodon samoin kuin s.y.t.:nkin antaa seuraava ns. *(Yleistetty) Eukleideen algoritmi.* Tässä oletetaan, että $0 \leq x \leq y$, mikä ilmeisesti ei mitenkään rajoita tilannetta. Merkitään $\mathrm{SYT}(x, y) = (d, c_1, c_2)$.

(Yleistetty) Eukleideen algoritmi:

1. Jos $x = 0$, niin tulostetaan $\mathrm{SYT}(x, y) = (y, 0, 1)$ ja lopetetaan.

2. Jos $x > 0$, niin jaetaan y x:llä: $y = qx + r$, missä $0 \leq r < x$. Ilmeisesti $\mathrm{syt}(x, y) = \mathrm{syt}(x, r)$ (ajattele jaollisuuksia). Etsitään $\mathrm{SYT}(r, x) = (d, e_1, e_2)$. Silloin

$$d = e_1 r + e_2 x = e_1(y - xq) + e_2 x = (e_2 - e_1 q)x + e_1 y.$$

   Tulostetaan $\mathrm{SYT}(x, y) = (d, e_2 - e_1 q, e_1)$ ja lopetetaan.

Rekursio on päättyvä, koska $\min(r, x) < \min(x, y)$, ts. aina kutsuttaessa SYT ko. minimiarvo pienenee ja on lopulta $= 0$.

Ilmeisesti juuri ennen kuin tullaan kohtaan 1. on $y = qx$ ja $r = 0$ ja $d = x$, jolloin $c_1 = 1 \leq y$ ja $c_2 = 0 \leq x$. Toisaalta, jos $y = qx + r$ ja $d = e_1 r + e_2 x$, missä $|e_1| \leq x$ ja $|e_2| \leq r$, niin e_1 ja e_2 ovat erimerkkiset ja $|e_2 - e_1 q| = |e_2| + |e_1| q \leq r + xq = y$. □

Suoraan Bezout'n lauseesta näkee seuraavan tuloksen:

Seuraus. *Jos kokonaisluku z jakaa kokonaisluvut x ja y (joista ainakin toinen on $\neq 0$), niin se jakaa myös $\mathrm{syt}(x, y)$:n.*

Huomautus. *Tästä johtuen $\mathrm{syt}(x, y)$ määritelläänkin usein siten, että se on x:n ja y:n yhteinen tekijä, joka on jaollinen kaikilla näiden lukujen yhteisillä tekijöillä. Tämä johtaa samaan s.y.t.:n käsitteeseen. Vm. määritelmä soveltuu kuitenkin myös tilanteeseen $x = y = 0$ ja antaa em. kaavan $\mathrm{syt}(0, 0) = 0$.*

Toinen seuraus Bezout'n lauseesta on lukujen tekijöihinjaon yksikäsitteisyys (ks. Lause 2.2).

Lause 2.6. *Kokonaisluvun $x \neq 0$ tekijöihinjako on yksikäsitteinen.*

Todistus. Asetetaan vastaoletus: On kokonaisluku x, jolla on (ainakin) kaksi eri tekijöihinjakoa. Voidaan olettaa, että x on positiivinen ja pienin sellainen positiivinen kokonaisluku, jolla on vastaoletuksessa mainittu ominaisuus. Ilmeisesti $x \geq 2$ (sillä ykkösen ainoa tekijöihinjako on tyhjä tulo). Nyt x voidaan kirjoittaa tulomuotoihin

$$x = p_1^{i_1} p_2^{i_2} \cdots p_n^{i_n} = q_1^{j_1} q_2^{j_2} \cdots q_m^{j_m},$$

missä $p_1, \ldots, p_n$ ovat eri alkulukuja ja samoin $q_1, \ldots, q_m$ ovat eri alkulukuja ja $i_1, \ldots, i_n, j_1, \ldots, j_m$ ovat positiivisia kokonaislukuja. Itse asiassa edelleen tiedetään, että myös alkuluvut $p_1, \ldots, p_n$ eroavat alkuluvuista $q_1, \ldots, q_m$. Jos nimittäin esimerkiksi $p_1 = q_1$, niin kokonaisluvulla x/p_1 olisi myös kaksi eri tekijöihinjakoa ja se on pienempi kuin x. Ilmeisesti $\mathrm{syt}(p_1, q_1) = 1$, joten Bezout'n muodossa

$$1 = c_1 p_1 + c_2 q_1.$$

Mutta tästä seuraa, että

$$q_1^{j_1 - 1} q_2^{j_2} \cdots q_m^{j_m} = c_1 p_1 q_1^{j_1 - 1} q_2^{j_2} \cdots q_m^{j_m} + c_2 x,$$

josta havaitaan, että p_1 jakaa tulon $q_1^{j_1-1}q_2^{j_2}\cdots q_m^{j_m}$, ts.

$$q_1^{j_1-1}q_2^{j_2}\cdots q_m^{j_m} = p_1 z.$$

Koska z:lla ja $q_1^{j_1-1}q_2^{j_2}\cdots q_m^{j_m}$:llä on yksikäsitteiset tekijöihinjaot (ne ovat pienempiä kuin x), seuraa tästä, että p_1 on jokin alkuluvuista $q_1,\ldots,q_m$, mikä on ristiriita. Vastaoletus on siis väärä. □

Esitettäessä rationaaliluku muodossa x/y, on tapana että $\mathrm{syt}(x,y)=1$, ts. luku on *supistetussa muodossa.* Tämä on erityisen tärkeää suurilla luvuilla laskettaessa, jotta x ja y eivät pääsisi kasvamaan suuriksi. Supistettu muoto saadaan luonnollisesti jakamalla x ja y $\mathrm{syt}(x,y)$:llä, joten pitkissä laskuissa s.y.t. joudutaan etsimään toistuvasti.

On tärkeää huomata, että Bezout'n lauseessa mainittu kertoimia koskeva ehto on voimassa eo. Eukleideen algoritmin joka vaiheessa. Välitulokset eivät näin kasva suuriksi. Jotta toisaalta saadaan mielikuva siitä miten monta askelta algoritmi enintään ottaa, merkitään Y_n:llä pienintä sellaista positiivista kokonaislukua y, että jollekin kokonaisluvulle $0 \le x < Y_n$ Eukleideen algoritmi tarvitsee n askelta. Ilmeisesti $Y_0=1$, $Y_1=2$ ja $Y_2=3$.

Lause 2.7. (Lamén lause) *Y_n toteuttaa rekursiokaavan*

$$Y_n = Y_{n-1} + Y_{n-2},$$

ts. Y_n on ns. $n+1$:s Fibonaccin luku[4] *F_{n+1}. Lisäksi nimenomaan $\mathrm{syt}(F_n,F_{n+1})$:n laskeminen vaatii ne n askelta Eukleideen algoritmilla, kun $n>0$.*

Todistus. Laskettaessa $\mathrm{syt}(x,Y_n)$ Eukleideen algoritmilla tehdään jakolaskut

$$\begin{aligned} y_0 = Y_n = q_1x + r_1\ ,\ y_1 = x = q_2r_1 + r_2\ ,\ldots,\ y_{n-3} = r_{n-4} = q_{n-2}r_{n-3} + r_{n-2}\ ,\\ y_{n-2} = r_{n-3} = q_{n-1}r_{n-2} + r_{n-1}\ ,\ y_{n-1} = r_{n-2} = q_nr_{n-1}, \end{aligned}$$

joten

$$y_{n-1} \ge 2\ ,\ y_{n-2} \ge y_{n-1}+1\ ,\ y_{n-3} \ge y_{n-2}+y_{n-1}\ ,\ldots$$

Siispä pienin y_0 eli Y_n saadaan valitsemalla

$$y_{n-1} = 2\ ,\ y_{n-2} = y_{n-1}+1\ ,\ y_{n-3} = y_{n-2}+y_{n-1}\ ,\ldots,$$

jolloin $y_{n-j} = F_{j+1}$ ($j=1,2,\ldots,n$) ja erityisesti $Y_n = y_0 = F_{n+1}$ ja $x = y_1 = F_n$. □

Seuraus. *Jos x ja y ovat kokonaislukuja ja $0 \le x < y$, niin $\mathrm{syt}(x,y)$:n laskeminen Eukleideen algoritmilla vie enintään*

$$\left\lfloor \ln(\sqrt{5}y) \Big/ \ln\frac{1+\sqrt{5}}{2} \right\rfloor - 1$$

askelta.

Todistus. Jos askelia tarvitaan n, niin $y \ge Y_n = F_{n+1}$. Suoraan laskien voidaan todeta, että

$$F_n = \frac{1}{\sqrt{5}}\left(\frac{2}{\sqrt{5}-1}\right)^{n+1} - \frac{1}{\sqrt{5}}\left(\frac{-2}{\sqrt{5}+1}\right)^{n+1}.$$

Tämän voi myös tehdä Maple-ohjelmistolla:

[4] Fibonaccin luvut F_n ($n=0,1,2,\ldots$) määritellään sopimalla ensin, että $F_0=F_1=1$ ja jatkamalla määrittelyä rekursiokaavalla $F_n = F_{n-1}+F_{n-2}$ ($n=2,3,\ldots$).

```
> rsolve({F(n)=F(n-1)+F(n-2),F(0)=1,F(1)=1},F);
```

$$-2/5\sqrt{5}\left(-2\left(1-\sqrt{5}\right)^{-1}\right)^{n}\left(1-\sqrt{5}\right)^{-1}+2/5\sqrt{5}\left(-2\left(\sqrt{5}+1\right)^{-1}\right)^{n}\left(\sqrt{5}+1\right)^{-1}$$

Näytetään tätä käyttäen, että

$$F_n > \frac{1}{\sqrt{5}}\left(\frac{2}{\sqrt{5}-1}\right)^n \quad (n \geq 0).$$

Koska

$$\left(\frac{-2}{\sqrt{5}+1}\right)^{n+1} < \frac{2}{\sqrt{5}+1},$$

riittää näyttää, että

$$\left(\frac{2}{\sqrt{5}-1}\right)^{n+1} - \left(\frac{2}{\sqrt{5}-1}\right)^n \geq \frac{2}{\sqrt{5}+1}$$

eli että

$$\left(\frac{2}{\sqrt{5}-1}\right)^n \left(\frac{2}{\sqrt{5}-1}-1\right) \geq \frac{2}{\sqrt{5}+1}.$$

Tämä on kuitenkin selvä, sillä

$$\frac{2}{\sqrt{5}-1}-1 = \frac{\sqrt{5}-1}{2} = \frac{2}{\sqrt{5}+1} \quad \text{ja} \quad \frac{2}{\sqrt{5}-1} > 1.$$

Siispä

$$\left(\frac{2}{\sqrt{5}-1}\right)^{n+1} < \sqrt{5}F_{n+1} \leq \sqrt{5}y,$$

mistä otetaan puolittain logaritmit ja pohjat. □

Eukleideen algoritmin soveltaminen ei näin ole kovin työlästä, $\ln(\sqrt{5}y)$ on verrannollinen y:n pituuteen (Lause 2.4). Tarkemmin Eukleideen algoritmin laskennallista vaativuutta käsittelee esimerkiksi KNUTH.

Useamman kuin kahden kokonaisluvun $x_1, x_2, \ldots, x_N$ suurin yhteinen tekijä

$$d = \mathrm{syt}(x_1, x_2, \ldots, x_N)$$

määritellään samoin kuin kahdenkin, eli se on suurin kokonaisluku, joka jakaa kaikki luvut $x_1, x_2, \ldots, x_N$. Jälleen vaaditaan, että ainakin yksi luvuista on $\neq 0$, voidaan sopia, että $x_N \neq 0$. Tällainen s.y.t. voidaan laskea soveltamalla $N-1$ kertaa Eukleideen algoritmia:

Lause 2.8. $\mathrm{syt}(x_1, x_2, \ldots, x_N) = \mathrm{syt}(x_1, \mathrm{syt}(x_2, \ldots, x_N))$

$$= \mathrm{syt}(x_1, \mathrm{syt}(x_2, \mathrm{syt}(x_3, \ldots, \mathrm{syt}(x_{N-1}, x_N)\cdots)))$$

ja lisäksi s.y.t. voidaan kirjoittaa Bezout'n muodossa

$$\mathrm{syt}(x_1, x_2, \ldots, x_N) = c_1x_1 + c_2x_2 + \cdots + c_Nx_N.$$

Todistus. Merkitään lyhyyden vuoksi

$$d = \mathrm{syt}(x_1, x_2, \ldots, x_N) \quad \text{ja} \quad d' = \mathrm{syt}(x_1, \mathrm{syt}(x_2, \mathrm{syt}(x_3, \ldots, \mathrm{syt}(x_{N-1}, x_N)\cdots))).$$

Bezout'n lauseen nojalla

$$\mathrm{syt}(x_{N-1}, x_N) = e_1 x_{N-1} + e_2 x_N$$

ja edelleen

$$\mathrm{syt}(x_{N-2}, \mathrm{syt}(x_{N-1}, x_N)) = e_3 x_{N-2} + e_4 \mathrm{syt}(x_{N-1}, x_N) = e_3 x_{N-2} + e_4 e_1 x_{N-1} + e_4 e_2 x_N,$$

jne., eli lopulta nähdään, että

$$d' = c_1 x_1 + c_2 x_2 + \cdots + c_N x_N.$$

Tästä seuraa, että $d \mid d'$ ja siis $d \leq d'$. Toisaalta d' jakaa x_1:n ja s.y.t.:n

$$\mathrm{syt}(x_2, \mathrm{syt}(x_3, \ldots, \mathrm{syt}(x_{N-1}, x_N) \cdots)).$$

Viimemainittu s.y.t. jakaa x_2:n ja s.y.t.:n $\mathrm{syt}(x_3, \ldots, \mathrm{syt}(x_{N-1}, x_N) \cdots)$. Jne. Siispä d' jakaa kaikki luvut $x_1, x_2, \ldots, x_N$ ja näin ollen $d' \leq d$. Kaiken kaikkiaan päätellään, että $d = d'$. □

Jos lukujen $x_1, x_2, \ldots, x_N$ tekijöihinjaot ovat

$$x_i = \pm p_1^{j_{i1}} p_2^{j_{i2}} \cdots p_M^{j_{iM}} \quad (i = 1, 2, \ldots, N),$$

missä sovitaan, että $j_{ik} = 0$, mikäli alkuluku p_k ei ole x_i:n tekijä, niin ilmeisesti

$$\mathrm{syt}(x_1, x_2, \ldots, x_N) = p_1^{\min\{j_{11}, \ldots, j_{N1}\}} p_2^{\min\{j_{12}, \ldots, j_{N2}\}} \cdots p_M^{\min\{j_{1M}, \ldots, j_{NM}\}}.$$

Vaikeus tämän tuloksen käytössä on se, ettei tekijöihinjakoja yleisesti tunneta ja niiden etsiminen taas on työlästä.

Kokonaislukujen $x_1, x_2, \ldots, x_N$ *pienin yhteinen jaettava* (*p.y.j.*) on pienin positiivinen kokonaisluku, jonka kaikki luvut $x_1, x_2, \ldots, x_N$ jakavat, merkitään $\mathrm{pyj}(x_1, x_2, \ldots, x_N)$. Jotta p.y.j. olisi olemassa, pitää olla $x_1, x_2, \ldots, x_N \neq 0$. Ajatellen edellä ollutta tekijöihinjakoa on ilmeisesti

$$\mathrm{pyj}(x_1, x_2, \ldots, x_N) = p_1^{\max\{j_{11}, \ldots, j_{N1}\}} p_2^{\max\{j_{12}, \ldots, j_{N2}\}} \cdots p_M^{\max\{j_{1M}, \ldots, j_{NM}\}}.$$

Myös p.y.j. saadaan rekursiivisesti Eukleideen algoritmilla (ilman tietoa tekijöistä), sillä

Lause 2.9. $$\begin{aligned}\mathrm{pyj}(x_1, x_2, \ldots, x_N) &= \mathrm{pyj}(x_1, \mathrm{pyj}(x_2, \ldots, x_N)) \\ &= \mathrm{pyj}(x_1, \mathrm{pyj}(x_2, \mathrm{pyj}(x_3, \ldots, \mathrm{pyj}(x_{N-1}, x_N) \cdots)))\end{aligned}$$

ja

$$\mathrm{pyj}(x_1, x_2) = \frac{|x_1 x_2|}{\mathrm{syt}(x_1, x_2)}.$$

Todistus. Lauseen ensimmäinen kaava seuraa tekijöihinjakokaavasta, sillä p_k:n eksponentti $\mathrm{pyj}(x_1, \mathrm{pyj}(x_2, \ldots, x_N))$:ssä on $\max\{j_{1k}, \max\{j_{2k}, \ldots, j_{Nk}\}\}$ ja toisaalta

$$\max\{j_{1k}, \max\{j_{2k}, \ldots, j_{Nk}\}\} = \max\{j_{1k}, j_{2k}, \ldots, j_{Nk}\} \quad (k = 1, 2, \ldots, M).$$

Samoin toinen kaava seuraa tekijöihinjakokaavasta, sillä alkutekijän p_k eksponentti $x_1 x_2$:ssa on $j_{1k} + j_{2k}$ ja toisaalta

$$\max\{j_{1k}, j_{2k}\} = j_{1k} + j_{2k} - \min\{j_{1k}, j_{2k}\}.$$

□

Huomautus. *Tekijöihinjakokaavasta näkee, että myös useamman luvun s.y.t. on se näiden lukujen (positiivinen) yhteinen tekijä, jonka kaikki muut yhteiset tekijät jakavat, ja tätä käytetään usein määritelmänä. Vastaavasti näkee, että lukujen p.y.j. on se näiden lukujen (positiivinen) yhteinen jaettava, joka jakaa kaikki muut yhteiset jaettavat, ja tätäkin käytetään usein määritelmänä. Näiden vaihtoehtoisten määritelmien nojalla kirjoitetaan usein* $\mathrm{syt}(0, 0, \ldots, 0) = 0$ *ja myös* $\mathrm{pyj}(0, x_2, \ldots, x_n) = 0$.

2.4 Kongruenssilaskenta eli modulääarilaskenta

Kongruenssilaskennan idea on, että lasketaan vain kokonaislukujen jakojäännöksillä käyttäen (yhtä tai useampaa) kiinteää jakajaa, ns. *modulia* $m \geq 2$. Kongruenssilaskentaa kutsutaan usein myös *modulääarilaskennaksi.*

Sanotaan, että kokonaisluvut x ja y ovat *kongruentit modulo* m, merkitään

$$x \equiv y \mod m \quad \text{(ns. } \textit{kongruenssi}\text{)},$$

jos $x - y$ on jaollinen m:llä. Tämä luetaan myös "x on kongruentti y:n kanssa modulo m" tai vain "x on y modulo m". $x \equiv y \mod m$ sanoo, että jaettaessa x ja y modulilla m saadaan sama (jako)jäännös, ts. x ja y kuuluvat samaan (jako)*jäännösluokkaan* modulo m. Jokainen kokonaisluku kuuluu aina johonkin jäännösluokkaan modulo m ja vain yhteen sellaiseen.

Lause 2.10. (i) *Jos* $x \equiv y \mod m$ *ja* $u \equiv v \mod m$, *niin* $x + u \equiv y + v \mod m$.

(ii) *Jos* $x \equiv y \mod m$ *ja* $u \equiv v \mod m$, *niin* $xu \equiv yv \mod m$.

(iii) *Jos* $x \equiv y \mod m$ *ja* n *on positiivinen kokonaisluku, niin* $x^n \equiv y^n \mod m$.

Todistus. (i) Jos $x - y = km$ ja $u - v = lm$, niin $(x + u) - (y + v) = (k + l)m$.
(ii) Jos $x - y = km$ ja $u - v = lm$, niin $xu - yv = (x - y)u + (u - v)y = (ku + ly)m$.
(iii) Tämä seuraa (ii):sta. □

Näin ollen, jos ajatellaan jakojäännöksiä, voidaan laskea millä tahansa kokonaisluvuilla, joilla modulilla jaettaessa on kyseiset jakojäännökset, ja tulokset ovat samat, ts. edustajien valinnasta riippumattomat. Tietyt edustajajoukot, ns. *jäännössysteemit,* ovat kuitenkin usein esillä:

- *positiivinen jäännössysteemi* $0, 1, \ldots, m - 1$;
- *symmetrinen jäännössysteemi* (parittomalle m:lle)

$$-\frac{m-1}{2}\ ,\ -\frac{m-3}{2}\ ,\ \ldots\ ,\ -1\ ,\ 0\ ,\ 1\ ,\ \ldots\ ,\ \frac{m-3}{2}\ ,\ \frac{m-1}{2};$$

- *negatiivinen jäännössysteemi* $-m + 1, \ldots, -1, 0$.

Tavallisin on positiivinen jäännössysteemi. Yleisesti mitkä tahansa m kokonaislukua, joista mitkään eivät ole keskenään kongruentteja modulo m, muodostavat jäännössysteemin modulo m. Luvun x jäännöstä modulo m nimenomaan positiivisessa jäännössysteemissä merkitään jatkossa $(x,\mathrm{mod}\ m)$:llä.

Jakolasku ei yleisesti ole sallittua kongruensseissa, vaan vain seuraavassa mielessä:

Lause 2.11. $xu \equiv yu \mod m$ *on sama kuin kuin* $x \equiv y \mod m/\mathrm{syt}(u, m)$, *eli kongruenssista saa supistaa pois kokonaisluvun, mikäli samalla jakaa modulin sen ja supistettavan luvun s.y.t.:llä.*

Todistus. Lähdetään ensin liikkeelle siitä, että $xu \equiv yu \mod m$ eli että $(x - y)u = km$. Merkitään $d = \mathrm{syt}(u, m)$ ja $u = du_1$ sekä $m = dm_1$. Silloin ilmeisesti $\mathrm{syt}(u_1, m_1) = 1$ ja $m_1 = m/\mathrm{syt}(u, m)$ ja vielä $(x - y)u_1 = km_1$. Bezout'n lauseen mukaan $1 = c_1u_1 + c_2m_1$, josta seuraa että

$$x - y = c_1u_1(x - y) + c_2m_1(x - y) = (c_1k + c_2(x - y))m_1,$$

ts. että $x \equiv y \mod m/\mathrm{syt}(u, m)$.

Lähdetään sitten liikkeelle siitä, että $x \equiv y \mod m/d$ eli että $x - y = km/d$. Tästä seuraa, että $(x - y)d = km$ ja edelleen että $(x - y)u = u_1km$. Siispä $xu \equiv yu \mod m$. □

Erityisesti kokonaisluvun, jolla ei ole yhteisiä tekijöitä modulin kanssa, saa supistaa pois kongruenssista (moduliin koskematta).

Seuraus. *Jos* $\mathrm{syt}(x,m)=1$*, niin luvut* $y+kx$ *(*$k=0,1,\ldots,m-1$*) muodostavat jäännössysteemin modulo* m*, olipa* y *mikä tahansa kokonaisluku.*

Todistus. Lukuja on m kpl. Jos $y+ix \equiv y+jx \mod m$, missä $0 \le i,j \le m-1$, niin $ix \equiv jx \mod m$ ja Lauseen 2.11 nojalla $i \equiv j \mod m$. Siis $i-j=km$, mutta koska $0 \le i,j \le m-1$, tämä on mahdollista vain kun $k=0$ eli $i=j$. Luvut eivät siis ole kongruentteja keskenään. □

Samantapaisella tekniikalla nähdään välittömästi, että jos $\mathrm{syt}(x,m)=1$, niin x:llä on *inverssi modulo* m, ts. on sellainen kokonaisluku y, että

$$xy \equiv 1 \mod m.$$

Tällöin kirjoitetaan myös $x^{-1} \equiv y \mod m$ tai $1/x \equiv y \mod m$. Tällainen inverssi saadaan Eukleideen algoritmilla, sillä Bezout'n lauseen nojalla $1=c_1x+c_2m$ ja näin $x^{-1} \equiv c_1 \mod m$. Toisaalta, jos $\mathrm{syt}(x,m) \neq 1$, ei x:llä voi olla inverssiä modulo m, kuten helposti voi havaita.

Sellaiset jäännössysteemin luvut x, joille $\mathrm{syt}(x,m)=1$, muodostavat ns. *supistetun jäännössysteemin.* Vastaavia jäännösluokkia kutsutaan *alkuluokiksi* modulo m. Helposti voidaan todeta, että jos $x \equiv y \mod m$, niin $\mathrm{syt}(x,m)=\mathrm{syt}(y,m)$. Supistetuissa jäännössysteemeissä on näin aina yhtä monta lukua ja ne ovat pareittain kongruentteja modulo m. Supistetun jäännössysteemin modulo m lukujen lukumäärää kutsutaan *Eulerin (totientti)funktioksi*, merkitään $\phi(m)$. Se tarvitaan mm. RSA-kryptauksessa. Tavallisin on jälleen positiivisesta jäännössysteemistä muodostettu supistettu jäännössysteemi.

2.5 Jäännösluokkarenkaat ja alkukunnat

Kokonaisluvut jakautuvat m jäännösluokkaan sen mukaan ovatko ne kongruentteja i:n kanssa modulo m ($i=0,\ldots,m-1$). Luokkaa, johon kokonaisluku x kuuluu merkitään $\overline{x}$:lla. Huomaa, että silloin $\overline{x}=\overline{x+km}$, olipa k mikä tahansa kokonaisluku. Jäännösluokille voidaan määritellä peruslaskutoimitukset "edustajien välityksellä", ts.

$$\overline{x} \pm \overline{y} = \overline{x \pm y} \quad , \quad \overline{x} \cdot \overline{y} = \overline{x \cdot y} \quad \text{ja} \quad \overline{x}^n = \overline{x^n} \quad (n=0,1,\ldots).$$

Laskutoimituksen tulos ei riipu edustajan valinnasta, kuten on helppo todeta. Kokonaislukujen laskujen ominaisuudet siirtyvät jäännösluokille:

(1) $+$ ja $\cdot$ ovat liitännäiset sekä vaihdannaiset,

(2) osittelulait pätevät,

(3) jokaisella luokalla a on *vastaluokka* $-a$ ts. $a+(-a)=\overline{0}$ (jos $a=\overline{x}$, niin tietysti $-a=\overline{-x}$),

(5) $\overline{0}$ ja $\overline{1}$ "käyttäytyvät" kuten pitää, ts. $a+\overline{0}=a$ ja $a \cdot \overline{1}=a$, (ja lisäksi vielä $\overline{0} \neq \overline{1}$).

Algebrallisesti jäännösluokat modulo m muodostavat ns. *renkaan* (ks. pykälä 4.1 ja kurssi Algebra 1 tai Symbolinen analyysi 1), merkitään $\mathbb{Z}_m$ (*jäännösluokkarengas modulo* m).

Jos $\mathrm{syt}(x,m)=1$, niin jäännösluokalla $\overline{x}$, on myös *käänteisluokka* $\overline{x}^{-1}$, jolle $\overline{x} \cdot \overline{x}^{-1}=\overline{1}$. Jos taas $\mathrm{syt}(x,m) \neq 1$, niin tällaista käänteisluokkaa ei ole. Näin ollen kaikilla muilla jäännösluokilla kuin $\overline{0}$:lla on käänteisluokka tarkalleen siinä tapauksessa, että moduli m on alkuluku.

Tällöin jäännösluokkarengasta kutsutaan *alkukunnaksi*. Alkukunnissa on näin ollen käytössä myös jakolasku, ts. kertominen käänteisluokalla. Usein esiintyvä alkukunta on *binäärikunta* $\mathbb{Z}_2$, jonka alkiot ovat bitit $\overline{0}$ ja $\overline{1}$ (kirjoitetaan useimmiten ilman yläviivoja: 0 ja 1).

Jäännösluokkarenkaiden matriisien ja vektoreiden laskutoimitukset voidaan siirtää luonnollisella tavalla jäännösluokista muodostettujen vektorien ja matriisien laskutoimituksiksi. Käyttöön tulevat näin peruskursseilta tutut matriisien yhteen-, vähennys- ja kertolaskuoperaatiot sekä transponointi. Samoin neliömatriisien determinantit noudattavat tuttuja laskusääntöjä. Aivan samoin kuin peruskursseilla todetaan se, että neliömatriisilla on käänteismatriisi, jos sen determinantilla (joka on $\mathbb{Z}_m$:n jäännösluokka) on käänteisluokka. Huomaa, ettei riitä, että determinantti on $\neq \overline{0}$, sillä Cramerin sääntöä käyttäen käänteismatriisia muodostettaessa tarvitaan jakolasku determinantilla modulo m. Alkukunnissa sen sijaan tietysti riittää, että determinantti on $\neq \overline{0}$.

2.6 (Suurten) kokonaislukujen algoritmeja

Algoritmien tehokkuutta verrataan usein niihin tarvittavien perusaskelten lukumäärällä verrattuna syötelukujen (maksimi)pituuteen N. (Perusaskel voisi olla esimerkiksi numeraalien $0, 1, \ldots, 9$ yhteen-, vähennys- tai kertolasku.) Yleisin tällainen vertailunotaatio on ns. *O-notaatio*. Silloin $O(f(N))$ tarkoittaa kollektiivisesti sellaista funktiota $g(N)$, että jostain rajasta $N \geq N_0$ lähtien $|g(N)| \leq Cf(N)$, missä C on jokin vakio.

Yhteen- ja vähennyslasku

Tavallinen koulussa opittu yhteen- ja vähennyslasku ovat sellaisenaankin sopivia myös ohjelmoitaviksi.

Kertolasku

Tavallinen koulussa opittu kertolasku on sopiva tietokoneellekin, mutta se ei ole likellekään nopein mahdollinen. N- ja M-pituisten lukujen kertolasku vaatii tällä tavoin noin $O(NM)$ askelta, mikä voi olla paljon.

Karatsuban algoritmi on nopeampi kuin perinteinen algoritmi. Algoritmi on eräänlainen ”hajoita ja hallitse” -menetelmä. Positiivisten lukujen n ja m kertomiseksi (desimaaliesityksessä) kirjoitetaan ne ensin muotoon

$$n = a10^k + b \quad \text{ja} \quad m = c10^k + d,$$

missä $a, b, c, d < 10^k$ ja lukujen maksimipituus on $2k$ tai $2k - 1$. Toinen luvuista a ja c voi olla nolla, mutta eivät molemmat. (Ts. kirjoitetaan ainakin toinen luvuista 10^k-kantaisessa esityksessä.) Silloin

$$nm = (a10^k + b)(c10^k + d) = y10^{2k} + (x - y - z)10^k + z,$$

missä

$$x = (a + b)(c + d)\,, \quad y = ac \quad \text{ja} \quad z = bd,$$

ts. tarvitaankin vain kolme erillistä ”pitkää” kokonaislukujen kertolaskua (eikä neljä kuten äkkinäisempi luulisi). Kun nämä kolme kertolaskua

$$(a + b)(c + d)\,, \quad ac \quad \text{ja} \quad bd$$

edelleen suoritetaan samalla tavoin jakamalla kertolaskut kolmeen lyhyempään kertolaskuun, jne., jolloin lopulta päädytään kertotaulun käyttöön, saadaan varsinainen Karatsuban algoritmi (merkitään $\mathrm{TULO}(n,m) = nm$):

Karatsuban kertoalgoritmi:

1. Jos $n = 0$ tai $m = 0$, tulostetaan 0 ja lopetetaan.
2. Palautetaan tilanne sellaiseen, jossa kertoja ja kerrottava ovat positivisia:
   (2.1) Jos $n < 0$ ja $m > 0$ tai $n > 0$ ja $m < 0$, lasketaan $t = \mathrm{TULO}(|n|, |m|)$, tulostetaan $-t$ ja lopetetaan.
   (2.2) Jos $n < 0$ ja $m < 0$, lasketaan $t = \mathrm{TULO}(-n, -m)$, tulostetaan se ja lopetetaan.
3. Jos $n, m < 10$, katsotaan $\mathrm{TULO}(n,m)$ taulukosta ja lopetetaan.
4. Jos $n \geq 10$ tai $m \geq 10$, kirjoitetaan n ja m muotoon $n = a10^k + b$ ja $m = c10^k + d$, missä $a, b, c, d < 10^k$. (Desimaaliesityksessä tämä on helppoa.)
5. Lasketaan $\mathrm{TULO}(a+b, c+d)$, $\mathrm{TULO}(a,c)$ ja $\mathrm{TULO}(b,d)$, tulostetaan (helposti saatava)
$$\begin{aligned}\mathrm{TULO}(n,m) = 10^{2k}&\mathrm{TULO}(a,c) + 10^k(\mathrm{TULO}(a+b, c+d)\\ &- \mathrm{TULO}(a,c) - \mathrm{TULO}(b,d)) + \mathrm{TULO}(b,d)\end{aligned}$$
   ja lopetetaan.

Menettely pysähtyy, sillä kerrottavien lukujen maksimipituus pienenee noin puoleen joka kierroksella.

Jos kerrotaan Karatsuban algoritmilla kaksi N-pituista lukua ja merkitään $K(N)$:llä tarvittavien numeraaleja $0, 1, \ldots, 9$ koskevien peruslaskutoimitusten enimmäismäärää, niin ilmeisestikin saadaan $K(N)$:lle seuraava rekursiokaava

$$K(N) = \begin{cases} \alpha N + 3K(N/2), \text{ jos } N \text{ on parillinen} \\ \alpha N + 3K((N+1)/2), \text{ jos } N \text{ on pariton} \end{cases}, \quad K(1) = 1,$$

missä kerroin α saadaan tarvittavien yhteen- ja vähennyslaskujen määrästä (käytetystä algoritmista riippuen). Tietyn arvion tarvittavien perusoperaatioiden määrästä antaa

Lause 2.12. *Jos $N = 2^l$, niin $K(N) = (2\alpha+1)3^l - \alpha 2^{l+1} = (2\alpha+1)N^{\log_2 3} - 2\alpha N$.*

Todistus. Ilmeisesti arvo on oikea, kun $N = 1$. Jos arvo pätee, kun $N = 2^l$, se pätee myös kun $N = 2^{l+1}$, sillä

$$K(2^{l+1}) = \alpha 2^{l+1} + 3K(2^l) = \alpha 2^{l+1} + 3(2\alpha+1)3^l - 3\alpha 2^{l+1} = (2\alpha+1)3^{l+1} - \alpha 2^{l+2}.$$

□

Luonnollisesti lauseesta suurelle pituudelle N saatava perusoperaatioiden enimmäismäärä

$$(2\alpha+1)N^{\log_2 3} - 2\alpha N = O(N^{\log_2 3}) = O(N^{1.585})$$

on tuntuvasti pienempi kuin $O(N^2)$. Esimerkiksi jos $N = 100000$, niin $N^2/N^{\log_2 3} \approx 119$. Karatsuban menetelmästä on vieläkin nopeampia versioita, joissa kertolasku jaetaan useampaan kuin kahteen osaan, ks. esimerkiksi MIGNOTTE.

Nopeimmat kertolaskualgoritmit perustuvat ns. nopean Fourier'n muunnoksen (FFT) käyttöön, ks. esimerkiksi LIPSON tai CRANDALL & POMERANCE. Tällöin perusoperaatioiden lukumäärä on $O(N \ln N \ln(\ln N))$. (Ks. myös kurssi Fourier'n menetelmät.)

Jakolasku

Tavanomainen koulussa opetettu ns. ”pitkäjako” on siirrettävissä tietokoneelle, tosin siinä oleva arvausvaihe on jonkin verran hankala toteuttaa tehokkaasti, jos kantaluku on iso, ks. KNUTH. Perusoperaatioiden lukumäärä on $O(N^2)$, missä on N on jaettavan pituus.

Peruskursseilta tuttuun Newtonin menetelmään pohjautuva jakolasku on varsin tehokas (ks. moniste RUOHONEN, K.: Symbolinen analyysi). Katsotaan tässä toinen nopea jakoalgoritmi, *Karatsuban algoritmi.* Oletetaan, että jaettava n ja jakaja m on esitetty binäärijärjestelmässä. Lisäksi oletetaan, että n ja m ovat positiivisia. Tapaus, jossa n ja m ovat negatiivisia, on esimerkiksi suoraan palautettavissa $-n$:n jakamiseen $-m$:llä, vastaavasti muut tapaukset. Merkitään n:n pituutta N:llä ja m:n pituutta M:llä. Jakaminen 2:n potenssilla on suoritettavissa suoraan lähtien binääriesityksestä. Koska tapaukset $M = 1$, $N < M$ ja $N = M$ ovat helppoja, oletetaan, että $N > M \geq 2$. Jos kirjoitetaan

$$n = qm + r\ , \quad 0 \leq r < m,$$

niin merkitään $\mathrm{JAKO}(n, m) = (q, r)$. Jakaminen etenee eri vaiheissa.

(1) Tapauksessa $N \geq 2M$ lasketaan ensin $K = \lfloor N/2 \rfloor - M + 1$ ja palautetaan tapaus jakoon

$$\mathrm{JAKO}(n, 2^K m) = (q', r').$$

Tämä onnistuu, sillä jos kirjoitetaan $q = s2^K + t$, missä $0 \leq t < 2^K$, niin

$$n = (s2^K + t)m + r = s2^K m + (tm + r)$$

ja

$$tm + r < (2^K - 1)m + m = 2^K m.$$

Näin ollen pitää olla $q' = s$ ja $r' = tm + r$. Luvut t ja r saadaan sitten jakolaskusta $\mathrm{JAKO}(r', m) = (t, r)$, joka jää suoritettavaksi. (Mutta huomaa, että r':n pituus binääriesityksessä on enintään $M + K = \lfloor N/2 \rfloor + 1$.)

(2) Nyt voidaan olettaa, että $N \leq 2M - 1$. Jaetaan ensin m ”kahtia”. Merkitään $M = 2l + k$, missä $k = 0$ tai $k = -1$, ja kirjoitetaan m muotoon $m = c2^l + d$, missä $d < 2^l$. Huomaa, että silloin $c \geq 2^{l+k-1}$. Merkitään edelleen

$$\mu = \frac{d}{c2^l}.$$

Silloin (geometrinen sarja)

$$\frac{n}{m} = \frac{n}{c2^l + d} = \frac{n}{c2^l}\frac{1}{1+\mu} = \frac{n}{c2^l}(1 - \mu + \mu^2 - + \cdots) = \frac{n}{c2^l}(1 - \mu) + \epsilon_1$$

ja siirrytään laskemaan $\dfrac{n}{c2^l}(1 - \mu)$:tä.

(3) Jaetaan n ”kahtia”, ts. kirjoitetaan $n = a2^{2l} + b$, missä $b < 2^{2l}$. Silloin

$$\frac{n}{c2^l}(1 - \mu) = \frac{n}{c2^l}\frac{c2^l - d}{c2^l} = \frac{a}{c^2}(c2^l - d) + \frac{b}{c^2 2^{2l}}(c2^l - d) = \frac{a}{c^2}(c2^l - d) + \epsilon_2$$

ja siirrytään laskemaan $\dfrac{a}{c^2}(c2^l - d)$:tä.

(4) Tehdään jaot

$$\mathrm{JAKO}(a,c) = (q',r') \quad \text{ja} \quad \mathrm{JAKO}(2^l r' - q'd, c) = (q'', r'').$$

(Huomaa, miten näissä jaoissa jaettavien/jakajien pituus on pienentynyt ja että tässä tarvitaan myös kertolasku $q'd$.) Silloin

$$\begin{aligned} a(c2^l - d) &= q'2^l c^2 - q'cd + (c2^l - d)r' \\ &= q'2^l c^2 + (2^l r' - q'd)c - dr' \\ &= (q'2^l + q'')c^2 + r''c - dr' \end{aligned}$$

ja

$$\frac{a}{c^2}(c2^l - d) = q'2^l + q'' + \epsilon_3.$$

Kaiken kaikkiaan siis

$$\frac{n}{m} = q'2^l + q'' + \epsilon_1 + \epsilon_2 + \epsilon_3$$

Arvioidaan virheet ϵ_1, ϵ_2 ja ϵ_3. Ensinnäkin

$$0 \le \epsilon_1 \le \frac{n}{m}\mu^2 = \frac{nd^2}{mc^2 2^{2l}} < \frac{2^{2M-1}2^{2l}}{2^{M-1}2^{2l+2k-2}2^{2l}} = 2^{2-k} \le 8.$$

Edelleen, koska

$$\frac{b}{c2^l} < \frac{2^{2l}}{2^{l+k-1}2^l} = 2^{1-k} \le 4 \quad \text{ja} \quad \frac{c2^l - d}{c2^l} \le 1,$$

on $0 \le \epsilon_2 < 4$. Vielä

$$1 > \epsilon_3 = \frac{r''}{c} - \frac{d}{c}\frac{r'}{c} \ge -\frac{d}{c} > -2^{1-k} \ge -4.$$

Oikea osamäärä on näin jokin luvuista $q'2^l + q'' + i$ ($i = -3, -2, \ldots, 12$). Mikä niistä, selviää kertolaskulla ja muutamalla vähennyslaskulla.

Karatsuban jakoalgoritmi:

1. Jos $n = 0$, tulostetaan $(0,0)$ ja lopetetaan.

2. Palaudutaan tapaukseen, jossa jakaja ja jaettava ovat positiivisia:

   (2.1) Jos $n < 0$ ja $m > 0$, lasketaan $\mathrm{JAKO}(-n,m) = (q,r)$. Mikäli $r = 0$, tulostetaan $(-q,0)$, muuten tulostetaan $(-q-1, m-r)$, ja lopetetaan.

   (2.2) Jos $n > 0$ ja $m < 0$, lasketaan $\mathrm{JAKO}(n,-m) = (q,r)$, tulostetaan $(-q,r)$ ja lopetetaan.

   (2.3) Jos $n, m < 0$, lasketaan $\mathrm{JAKO}(-n,-m) = (q,r)$. Mikäli $r = 0$, tulostetaan $(q,0)$, muuten tulostetaan $(q+1, -m-r)$ ja lopetetaan.

3. Asetetaan $N \leftarrow$ jaettavan n pituus ja $M \leftarrow$ jakajan m pituus binäärijärjestelmässä.

4. Selvitetään triviaalit tapaukset:

   4.1 Jos $n < m$, tulostetaan $(0,n)$ ja lopetetaan.

   4.2 Jos $N = M$ ja $n \ge m$, tulostetaan $(1, n-m)$ ja lopetetaan.

5. Jos $N \geq 2M$, asetetaan $K \leftarrow \lfloor N/2 \rfloor - M + 1$, lasketaan jaot

$$\mathrm{JAKO}(n, 2^K m) = (q', r') \quad \text{ja} \quad \mathrm{JAKO}(r', m) = (t, r),$$

tulostetaan $(q'2^K + t, r)$ ja lopetetaan.

6. Asetetaan $l \leftarrow \lfloor (M+1)/2 \rfloor$ ja kirjoitetaan n ja m muotoihin

$$n = a2^{2l} + b \quad \text{ja} \quad m = c2^l + d,$$

missä $b < 2^{2l}$ ja $d < 2^l$.

7. Suoritetaan jako $\mathrm{JAKO}(a, c) = (q', r')$ ja kertolasku $q'd$ ja vielä jako

$$\mathrm{JAKO}(2^l r' - q'd, c) = (q'', r'').$$

8. Asetetaan $Q \leftarrow q'2^l + q''$ ja testataan kertolaskulla ja vähennyslaskuilla mikä luvuista $Q + i$ ($i = -3, -2, \ldots, 12$) on oikea osamäärä q. Lasketaan $r = n - qm$ (tämä voidaan liittää eo. testaukseen eikä vaadi kertolaskua), tulostetaan (q, r) ja lopetetaan.

Tämän algoritmin kompleksisuuden tarkempi arvio samaan tapaan kuin Karatsuban kertolaskualgoritmille on vaikeaa. Tulos on se, että jakolasku on oleellisesti vain vähän kompleksisempi kuin kertolasku (ks. MIGNOTTE).

Potenssiin korotus

Luvun a potenssiin korotus a^n peräkkäisillä kertomisilla on liian aikaaviepää, sillä siihen tarvitaan $|n| - 1$ kertolaskua, kun itse asiassa enintään $2\lfloor \log_2 |n| \rfloor$ kertolaskuakin riittää:

Venäläisten talonpoikien menetelmä:

1. Jos $n = 0$, tulostetaan potenssi 1 ja lopetetaan.
2. Jos $n < 0$, asetetaan $a \leftarrow a^{-1}$ sekä $n \leftarrow -n$.
3. Jos $n \geq 1$, muodostetaan n:n binääriesitys $b_j b_{j-1} \cdots b_0$, missä $j = \lfloor \log_2 n \rfloor$ (n:n pituus binäärilukuna miinus yksi, ks. Lause 2.4).
4. Asetetaan $i \leftarrow 0$ sekä $x \leftarrow 1$ ja $y \leftarrow a$.
5. Jos $i = j$, niin tulostetaan potenssi xy ja lopetetaan.
6. Jos $i < j$ ja
   - 6.1 $b_i = 0$, niin asetetaan $i \leftarrow i + 1$ sekä $y \leftarrow y^2$ ja mennään kohtaan 5.
   - 6.2 $b^i = 1$, niin asetetaan $x \leftarrow xy$ ja $y \leftarrow y^2$ sekä $i \leftarrow i + 1$ ja mennään kohtaan 5.

Algoritmin toimivuus seuraa välittömästi binääriesityksestä:

$$|n| = b_j 2^j + b_{j-1} 2^{j-1} + \cdots + b_1 2 + b_0$$

ja

$$a^{|n|} = a^{b_j 2^j} a^{b_{j-1} 2^{j-1}} \cdots a^{b_1 2} a^{b_0}.$$

Käytännössä lienee mukavaa laskea n:n binääriesitystä sitä mukaa, kun sitä tarvitaan, eikä kerralla alussa.

Itse asiassa tämä menettely toimii kaikenlaisille potensseille ja myös silloin kun kertolasku ei ole vaihdannainen, siis esimerkiksi polynomien tai matriisien potenssiinkorotuksessa. Laskettaessa potensseja modulo m pitää tulot tietysti redusoida koko ajan positiiviseen jäännössysteemiin modulo m, jotteivat välitulokset kasvaisi suuriksi. Näin voidaan laskea nopeasti hyvinkin korkeita modulääriisiä potensseja.

Nimi muuten tulee siitä, että venäläiset talonpojat käyttivät helmitaululla laskiessaan tätä menetelmää kertolaskuun, onhan $a \cdot n$ tulkittavissa a:n n:nneksi ”potenssiksi” yhteenlaskun suhteen. Algoritmi on ilmeisesti hyvin vanha.

Satunnaisluvun generointi

Satunnaisbittijonoja generoidaan yleisesti käyttäen p:nnen kertaluvun *siirtorekisteriä*[5] modulo 2:

$$r_i \equiv a_1 r_{i-1} + a_2 r_{i-2} + \cdots + a_p r_{i-p} \mod 2,$$

missä $a_1, a_2, \ldots, a_p$ ovat vakiobittejä (0 tai 1, $a_p = 1$). Aluksi tarvitaan tietysti siemenbitit $r_0, r_1, \ldots, r_{p-1}$. (Tässä lasketaan koko ajan positiivisessa jäännössysteemissä modulo 2 eli biteillä.) Saatava jono $r_p, r_{p+1}, \ldots$ ei tietenkään oikeasti ole mitenkään satunnainen, saadaanhan se täysin deterministisellä menettelyllä ja on periodinen (periodin pituus enintään 2^p). Kun valitaan kertoimet $a_1, a_2, \ldots, a_{p-1}$ sopivasti, saadaan jono kuitenkin käyttäytymään monessa mielessä satunnaisesti, periodi on pitkä jne., ks. esimerkiksi KNUTH. Yksinkertaisimmillaan melkein kaikki kertoimet ovat nollia.

Tyyppiä

$$r_i \equiv r_{i-q} + r_{i-p} \mod 2$$

olevat siirtorekisterit, missä p on alkuluku ja q valitaan sopivasti, tuottavat usein varsin hyviä satunnaisia bittejä. Seuraavassa taulussa on listattuna eräitä valintoja, missä luku q voidaan korvata myös luvulla $p - q$:

p	q (myös $p-q$ käy)	p	q (myös $p-q$ käy)
2	1	1279	216, 418
3	1	2281	715, 915, 1029
5	2	3217	67, 576
7	1, 3	4423	271, 369, 370, 649, 1393, 1419, 2098
17	3, 5, 6	9689	84, 471, 1836, 2444, 4187
31	3, 6, 7, 13	19937	881, 7083, 9842
89	38	23209	1530, 6619, 9739
127	1, 7, 15, 30, 63	44497	8575, 21034
521	32, 48, 158, 168	110503	25230, 53719
607	105, 147, 273	132049	7000, 33912, 41469, 52549, 54454

Nämä arvot on löydetty tietokone-etsinnällä.[6] Pienet p:n arvot eivät tietenkään ole kovin käyttökelpoisia.

[5] Klassinen viite on GOLOMB, S.W.: *Shift Register Sequences.* Aegean Park Press (1982)

[6] Artikkeliviitteet ovat ZIERLER, N.: On Primitive Trinomials Whose Degree is a Mersenne Exponent. *Information and Control* **15** (1969), 67–69 ja HERINGA, J.R. & BLÖTE, H.W.J. & COMPAGNER, A.: New Primitive Trinomials of Mersenne-Exponent Degrees for Random Number Generation. *International Journal of Modern Physics* **C3** (1992), 561–564.

Matriisimuodossa (ks. edellinen pykälä) siirtorekisteri on seuraavanlainen. Merkitään

$$\mathbf{r}_i = \begin{pmatrix} r_{i+p-1} \\ r_{i+p-2} \\ \vdots \\ r_i \end{pmatrix} \quad \text{ja} \quad \mathbf{A} = \begin{pmatrix} a_1 & a_2 & \cdots & a_{p-1} & a_p \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}.$$

$\mathbf{A}$ on ns. siirtorekisterin *oheismatriisi.* Silloin

$$\mathbf{r}_{i+1} \equiv \mathbf{A}\mathbf{r}_i \mod 2$$

ja näin ollen

$$\mathbf{r}_i \equiv \mathbf{A}^i \mathbf{r}_0 \mod 2 \quad (i = 0, 1, \ldots).$$

Matriisipotenssi $\mathbf{A}^i$ voidaan laskea nopeasti modulo 2 käyttäen venäläisten talonpoikien algoritmia. Niinpä, ehkä vähän yllättäen, voidaan varsin nopeasti laskea jonoa $r_p, r_{p+1}, \ldots$ ”etukäteen” hyvinkin pitkälle. Huomaa myös, että p:stä peräkkäisestä vektorista $\mathbf{r}_i$ voidaan ratkaista matriisi $\mathbf{A}$.

Satunnaisbittijonosta saadaan binääriesityksen kautta satunnaislukuja. Binääriesityksessään n-pituisia satunnaislukuja $s_0, s_1, \ldots$ saadaan jakamalla jono peräkkäisiin n bitin lohkoihin ja tulkitsemalla lohkot binääriluvuiksi. Matriisiesityksessä lasketaan $\mathbf{A}^n$ modulo 2 etukäteen, generoidaan jono

$$\mathbf{t}_j \equiv (\mathbf{A}^n)^j \mathbf{r}_0 \mod 2 \quad (j = 0, 1, \ldots)$$

ja (olettaen, että $p \geq n$) lasketaan

$$s_j = (2^{n-1}, 2^{n-2}, \ldots, 2, 1, 0, \ldots, 0)\mathbf{t}_j \quad (j = 0, 1, \ldots),$$

missä nollia on lisätty vaakavektoriin $p - n$ kpl.

Huomautus. *Kryptauksessa tarvittava satunnaisbittien/-lukujen generointi on varsin vaativaa. Huonosti generoidut satunnaisbitit kun edesauttavat kryptauksen purkamista tuntuvasti. Voidaankin sanoa, että satunnaislukujen generointi on viime aikoina edistynyt merkittävästi nimenomaan kryptauksen tarpeiden vuoksi. Yllä oleva siirtorekisterigeneraattori on riittävä ”tavallisiin” kryptaustehtäviin, varsinkin suurilla p:n arvoilla. Se on kuitenkin aivan liian helposti ennustettavissa ollakseen kryptologisesti vahva. Parempiakin menetelmiä on, mm. ns. Blum–Blum–Shub-generaattori, ks. pykälä 6.8. Ks. myös* GOLDREICH.

Luku 3

KLASSISIA KRYPTOSYSTEEMEJÄ

3.1 AFFINE. CAESAR

Jotta päästäisiin käyttämään edellisen luvun lukuteorian tuloksia, pitää selvätekstin symbolit ensin koodata luvuiksi tai jäännösluokiksi. Jos symboleja on M kpl, voidaan käyttää jäännösluokkia modulo M. Itse asiassa voidaan ajatella viesti kirjoitetuksi käyttäen näitä jäännösluokkia tai positiivisen jäännössysteemin lukuja.

Affiinissa kryptosysteemissä AFFINE kukin viestisymboli i (jäännösluokka modulo M esitettynä positiivisessa jäännössysteemissä) kryptataan seuraavasti:

$$e_{k_1}(i) = (ai + b, \bmod M).$$

Tässä a ja b ovat kokonaislukuja ja $\overline{a}$:lla on käänteisluokka $\overline{c}$ modulo M (ts. $\mathrm{syt}(a, M) = 1$). Kryptausavain k_1 muodostuu parista (a, b) ja dekryptausavain k_2 parista (c, b) (yleensä esitettyinä positiivisessa jäännössysteemissä). Dekryptausfunktio on

$$d_{k_2}(j) = (c(j - b), \bmod M).$$

Viestilohko on siis yhden pituinen. (Affiini kryptaus sopii näin ollen myös vuokryptaukseen.) Valittaessa a ja b positiivisesta jäännössysteemistä mahdollisten a:n arvojen lukumäärä on $\phi(M)$ (ks. pykälä 2.4), ja kaikkiaan kryptausavaimia on $\phi(M)M$ kpl. Avaimia on näin varsin vähän. Eräitä arvoja:

$$\phi(10) = 4\ ,\ \phi(26) = 12\ ,\ \phi(29) = 28\ ,\ \phi(40) = 16.$$

Erikoistapaus, jossa $a = 1$, tunnetaan *Caesar-kryptosysteeminä* CAESAR. Yleisempi kryptosysteemi, jossa

$$e_{k_1}(i) = (p(i), \bmod M)$$

ja p on kokonaiskertoiminen polynomi, ei ole juurikaan käyttökelpoisempi, avaimia on edelleen aika vähän (miksi?).

3.2 HILL. PERMUTATION. AFFINE-HILL. VIGENÈRE

Hillin[1] *kryptosysteemissä* HILL käytetään samaa symbolien koodausta jäännösluokiksi modulo M kuin AFFINEssa. Lohko kuitenkin muodostuu nyt d:stä jäännösluokasta ajateltuna d-vaakavektoriksi. Alkuperäinen d muuten oli 2. Kryptausavain on $d \times d$-matriisi $\mathbf{H}$, jolla on käänteismatriisi modulo M (ks. pykälä 2.5). Mainittu käänteismatriisi $\mathbf{H}^{-1} = \mathbf{K}$ modulo M taas on dekryptausavain.

[1] Lester S. Hill (1929)

Viestilohko

$$\mathbf{i} = (i_1, \ldots, i_d)$$

kryptataan nyt seuraavasti:

$$e_{\mathbf{H}}(\mathbf{i}) = (\mathbf{iH}, \bmod M).$$

Dekryptaus on vastaavasti

$$e_{\mathbf{K}}(\mathbf{j}) = (\mathbf{jK}, \bmod M)$$

(Tässä lasketaan koko ajan modulo M positiivisessa jäännössysteemissä.)

Kryptausavaimia on yhtä paljon kuin on kääntyviä $d \times d$-matriiseja modulo M. Tämä lukumäärä on vaikea laskea, yleensä avaimia kuitenkin on suhteellisen paljon, jos d on iso.

Erikoistapaus HILListä on PERMUTATION eli ns. *permutaatiokryptaus.* Siinä $\mathbf{H}$ on *permutaatiomatriisi,* ts. matriisi jonka jokaisella rivillä ja jokaisessa sarakkeessa on tarkalleen yksi ykkönen ja muut alkiot ovat nollia. Huomaa, että tällöin $\mathbf{H}^{-1} = \mathbf{H}^T$, ts. $\mathbf{H}$ on ortogonaalimatriisi. Permutaatiokryptauksessa viestilohkon symbolit permutoidaan tietyllä vakiopermutaatiolla, joka saadaan $\mathbf{H}$:sta.

Yleisempi kryptosysteemi on AFFINE-HILL eli *affiini Hillin kryptosysteemi.* Se eroaa HILListä siten, että kryptausavain k_1 on pari $(\mathbf{H}, \mathbf{b})$, missä $\mathbf{b}$ on vakio d-vaakavektori modulo M, ja dekryptausavain k_2 on vastaavasti pari $(\mathbf{K}, \mathbf{b})$. Tällöin

$$e_{k1}(\mathbf{i}) = (\mathbf{iH} + \mathbf{b}, \bmod M)$$

ja

$$e_{k2}(\mathbf{j}) = ((\mathbf{j} - \mathbf{b})\mathbf{K}, \bmod M).$$

Tästä puolestaan saadaan erikoistapauksena ns. *Vigenèren*[2] *kryptaus* VIGENÈRE valitsemalla $\mathbf{H} = \mathbf{I}_d$ ($d \times d$-identiteettimatriisi). (Tällainen $\mathbf{H}$:n valinta ei tietystikään käy HILLissä!) Vigenèren kryptauksessa viestilohkoon lisätään symboleittain d-pituinen avainsana modulo M.

3.3 ONE-TIME-PAD

Usein viestisymbolit koodataan (enintään) tietyn pituisiksi binääriluvuiksi (esimerkiksi ASCII-koodaus). Näin voidaan olettaa viestin olevan M-pituinen bittivektori. Jos viestin (maksimi)pituus tunnetaan ennalta ja kryptausta tarvitaan vain kerran, voidaan avaimeksi valita satunnainen M-pituinen bittivektori $\mathbf{b}$ (eli vektori modulo 2, ns. *kerta-avain*), joka kryptattaessa lisätään viestiin modulo 2. Tuloksena saatu kryptattu viestivektori on myös satunnainen (miksi?) eikä siitä mahdollinen salakuuntelija näin ollen saa ilman avainta mitään irti. Dekryptattaessa lisätään vastaavasti kryptattuun viestiin sama $\mathbf{b}$, sillä $2\mathbf{b} \equiv \mathbf{0} \mod 2$. Näin saadaan ns. *kerta-avainkryptaus* ONE-TIME-PAD.

3.4 Kryptanalyysi

Kryptanalyysin tarkoitus on murtaa kryptosysteemi, ts. löytää dekryptausavain (tai kryptausavain) tai ainakin kehittää menetelmä, jolla saadaan jotain tietoa ulos (joistain) kryptatuista viesteistä. Tällöin yleensä oletetaan, että kryptanalysoija on salakuuntelija tai muuten vihamielinen taho ja että kryptanalysoija tietää käytetyn kryptosysteemin (muttei sen avainta).

Kryptanalysoijan käytössä voi olla erilaista tietoa:

[2] Blaise de Vigenère (1523–1596)

(CO) vain jokin (satunnainen) kryptoteksti (*vain kryptoteksti,* cryptotext only),

(KP) jokin (satunnainen) selväteksti ja vastaava kryptoteksti (*tunnettu selväteksti,* known plaintext),

(CP) hänen valitsemansa selväteksti ja vastaava kryptoteksti (*valittu selväteksti,* chosen plaintext),

(CC) hänen valitsemansa kryptoteksti ja vastaava selväteksti (*valittu kryptoteksti,* chosen cryptotext).

Klassiset hyökkäysmenettelyt perustuvat usein *frekvenssianalyysiin,* ts. tietoon siitä että pitkissä kryptoteksteissä tietyt symbolit, symboliparit, symbolikolmikot, jne. esiintyvät tietyillä toisistaan eroavilla frekvensseillä. Frekvenssitauluja on tehty esimerkiksi tavallisesta englanninkielestä, amerikanenglannista, jne.

Huomautus. *Jos viesti pakataan ennen kryptausta, katoaa siitä frekvenssitietoa, ks. kurssi Informaatioteoria.*

Katsotaan esimerkkeinä eo. kryptosysteemien kryptanalyysejä.

AFFINE

Affiinissa kryptauksessa mahdollisia avaimia on yleensä vähän, joten CO-hyökkäyksessä ne voidaan käydä läpi yksi kerrallaan ja yrittää löytää todennäköinen selväteksti. Tämä ei tietenkään onnistu, jos viestissä ei ole mitään rakennetta, joka voidaan tunnistaa. Toisaalta pitkistä kryptoteksteistä voidaan etsiä frekvenssitaulujen mukaista rakennetta ja näin löytää KP-dataa, esimerkiksi yleisin yleensä esiintyvä symboli voitaisiin tunnistaa.

KP-hyökkäyksessä riittää löytää sellaiset kaksi viestisymboli-kryptosymboli-paria (i_1, j_1) ja (i_2, j_2), että $\mathrm{syt}(i_1 - i_2, M) = 1$. Tällaiset parit löytyvät yleensä pitkästä kryptotekstistä. Silloin matriisi

$$\begin{pmatrix} i_1 & 1 \\ i_2 & 1 \end{pmatrix}$$

on kääntyvä modulo M ja avain löytyy helposti:

$$\begin{pmatrix} j_1 \\ j_2 \end{pmatrix} \equiv \begin{pmatrix} i_1 & 1 \\ i_2 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} \mod M$$

eli

$$\begin{pmatrix} a \\ b \end{pmatrix} \equiv (i_1 - i_2)^{-1} \begin{pmatrix} 1 & -1 \\ -i_2 & i_1 \end{pmatrix} \begin{pmatrix} j_1 \\ j_2 \end{pmatrix} \mod M.$$

CP-hyökkäyksessä symboliparit (i_1, j_1) ja (i_2, j_2) voidaan suorastaan valita. CC-hyökkäyksessä riittää valita pitkä kryptoteksti.

AFFINEn helpon murrettavuuden takia se soveltuu vain kevyeen tiedon peittoon satunnaisilta lukijoilta.

HILL ja AFFINE-HILL

HILL-kryptosysteemissä avaimia on yleensä paljon, varsinkin jos d on iso. CO-hyökkäys ei näin ollen sellaisenaan onnistu helposti. Frekvenssianalyysiä soveltaen voidaan kuitenkin periaatteessa löytää KP-dataa, varsinkin jos d on pienehkö. KP-hyökkäyksessä riittää löytää sellaiset viestilohko-kryptolohko-parit $(\mathbf{i}_1, \mathbf{j}_1), \ldots, (\mathbf{i}_d, \mathbf{j}_d)$, että matriisit

$$\mathbf{S} = \begin{pmatrix} \mathbf{i}_1 \\ \vdots \\ \mathbf{i}_d \end{pmatrix} \quad \text{ja} \quad \mathbf{R} = \begin{pmatrix} \mathbf{j}_1 \\ \vdots \\ \mathbf{j}_d \end{pmatrix}$$

ovat kääntyviä modulo M. Huomaa, että itse asiassa riittää tietää toinen matriiseista kääntyväksi, toinen on sen jälkeen myös kääntyvä. Tietysti CP-hyökkäyksessä $\mathbf{S}$ on valittavissa suoraan ja CC-hyökkäyksessä taas $\mathbf{R}$. Jos $\mathbf{S}$ ja $\mathbf{R}$ tunnetaan, saadaan avain $\mathbf{H}$ helposti:

$$\mathbf{R} \equiv \mathbf{SH} \mod M \quad \text{eli} \quad \mathbf{H} \equiv \mathbf{S}^{-1}\mathbf{R} \mod M.$$

HILL on vaikea murtaa, ellei vähintään KP-dataa ole saatavilla, varsinkin jos d on iso (ja kryptanalysoija ei tiedä d:n arvoa). Toisaalta KP-hyökkäys ja varsinkin CP- tai CC-hyökkäys on helppo tehdä (dataa tarvitaan hyvin vähän), joten HILL ei käy vaativaan salaukseen.

AFFINE-HILL on HILLiä hieman vaikeampi murtaa. KP-hyökkäyksessä tarvitaan sellaiset viestilohko-kryptolohko-parit $(\mathbf{i}_1, \mathbf{j}_1), \ldots, (\mathbf{i}_{d+1}, \mathbf{j}_{d+1})$, että matriisit

$$\mathbf{S} = \begin{pmatrix} \mathbf{i}_1 - \mathbf{i}_{d+1} \\ \vdots \\ \mathbf{i}_d - \mathbf{i}_{d+1} \end{pmatrix} \quad \text{ja} \quad \mathbf{R} = \begin{pmatrix} \mathbf{j}_1 - \mathbf{j}_{d+1} \\ \vdots \\ \mathbf{j}_d - \mathbf{j}_{d+1} \end{pmatrix}$$

ovat kääntyviä modulo M. Huomaa jälleen, että itse asiassa riittää tietää toinen matriiseista kääntyväksi. CP-hyökkäyksessä $\mathbf{S}$ on valittavissa suoraan, CC-hyökkäyksessä $\mathbf{R}$. Jos $\mathbf{S}$ ja $\mathbf{R}$ tunnetaan, saadaan $\mathbf{H}$ helposti samoin kuin yllä. Kun $\mathbf{H}$ tunnetaan, saadaan $\mathbf{b}$ helposti.

VIGENÈRE

VIGENÈRE on ollut aikanaan niin paljon käytetty kryptosysteemi, että sen murtamista on mietitty paljon. Ensimmäinen toimenpide on etsiä d. Siihen on omia menetelmiään (d on VIGENÈREssä yleensä varsin iso). Sen jälkeen voidaan soveltaa frekvenssianalyysiä. Ks. STINSON tai SALOMAA tai BAUER.

ONE-TIME-PAD

Jollei avain ole kryptanalysoijan käytettävissä, on ONE-TIME-PAD mahdoton murtaa CO-hyökkäyksessä. Jos samaa avainta kuitenkin käytetään useita kertoja tullaan oleellisesti VIGENÈRE-salaukseen.

3.5 DES

3.5.1 Yleistä

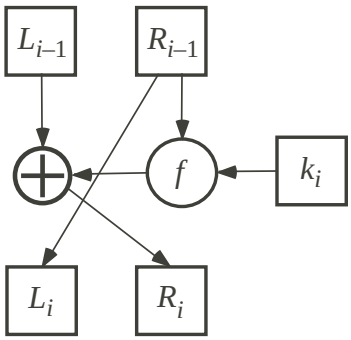
DES (Data Encryption Standard) on IBM:n 70-luvun alkupuolella kehittämä symmetrinen kryptosysteemi. (Se perustuu IBM:n aikaisemmin tekemään LUCIFER-systeemiin.) DES julkaistiin vuonna 1975 ja otettiin USA:ssa ”unclassified”-dokumenttien kryptausstandardiksi vuonna 1977. Sen jälkeen sitä on käytetty hyvin paljon eri yhteyksissä, myös kolminkertaisena 3-DES-systeeminä. DESin tapaisia kryptosysteemejä tunnetaan paljon: FEAL, IDEA, SAFER, RC5, BLOWFISH, jne., ks. MENEZES & VAN OORSCHOT & VANSTONE.

3.5.2 DESin määrittely

DES toimii bittisymboleilla, joten selväteksti- ja kryptotekstisymboleiksi voidaan ajatella $\mathbb{Z}_2$:n jäännösluokat (bitit) 0 ja 1. Selvätekstilohkon pituus on 64. Avain k on 56-bittinen. Se on yhteinen kryptauksessa ja dekryptauksessa. Pääpiirteissään DES toimii seuraavasti:

1. Selvätekstistä x muodostetaan bittijono x_0 permutoimalla x:n bitit tietyn kiinteän permutaation (ns. *alkupermutaation*) π_{ini} mukaan. Kirjoitetaan
$$x_0 = \pi_{\text{ini}}(x) = L_0 R_0,$$
missä L_0:ssa on x_0:n 32 ensimmäistä bittiä ja R_0:ssa loput.

2. Lasketaan jono $L_1R_1, L_2R_2, \ldots, L_{16}R_{16}$ iteroiden 16 kertaa seuraavaa menettelyä:
$$\begin{cases} L_i = R_{i-1} \\ R_i = L_{i-1} \oplus f(R_{i-1}, k_i), \end{cases}$$
missä $\oplus$ on bittikohtainen yhteenlasku modulo 2 (tunnetaan myös nimellä XOR-operaatio), f on funktio, joka annetaan myöhemmin, ja k_i on i:nnen iteraation avain, joka saadaan k:sta permutoimalla sen tietyt 48 bittiä tiettyyn järjestykseen. Havainnollisesti yksi iteraatioaskel on seuraavanlainen:



3. Sovelletaan alkupermutaation π_{ini} käänteispermutaatiota π_{ini}^{-1} (ns. *loppupermutaatio*) bittijonoon $R_{16}L_{16}$.

Vielä pitää antaa permutaatio π_{ini}, määritellä funktio f ja antaa avainjono $k_1, k_2, \ldots, k_{16}$, jotta kryptaus olisi määrätty.

Katsotaan ensin funktion f määrittely. f:n ensimmäinen argumentti R on 32-pituinen bittijono ja toinen argumentti K on 48-pituinen bittijono. f:n laskemiseksi menetellään seuraavasti:

1. Ensimmäinen argumentti R *laajennetaan laajennusfunktiolla* E. $E(R)$:ään otetaan R:n 32 bittiä, toistetaan niistä puolet ja permutoidaan. Bitit otetaan seuraavan taulukon mukaisesti, jota luetaan vasemmalta oikealle ja ylhäältä alas:

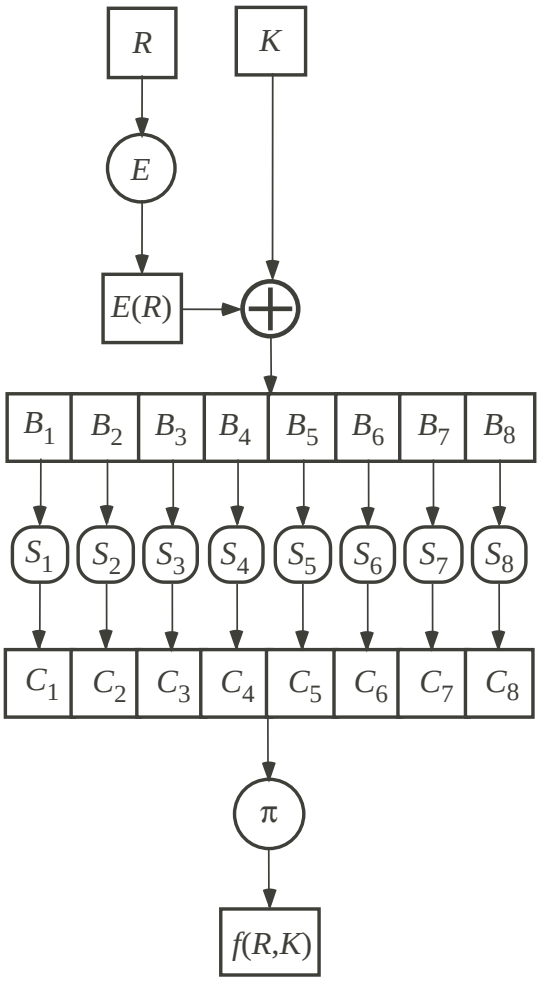
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

2. Lasketaan $E(R) \oplus K = B$ ja kirjoitetaan tulos kahdeksan 6-bittisen bittijonon katenaatioksi:
$$B = B_1B_2B_3B_4B_5B_6B_7B_8.$$

3. Seuraavaksi käytetään kahdeksaa ns. *S-laatikkoa* $S_1, \ldots, S_8$. Kukin S_i on kiinteä 4×16-taulukko, jossa esiintyy lukuja $0, 1, \ldots, 15$. Kun saadaan 6-pituinen bittijono
$$B_i = b_1b_2b_3b_4b_5b_6,$$
lasketaan $S_i(B_i) = C_i$ seuraavasti. Bitit b_1b_6 antavat erään rivin indeksin r binääriesityksen ($r = 0, 1, 2, 3$). Loput bitit $b_2b_3b_4b_5$ puolestaan antavat erään sarakkeen s indeksin binääriesityksen ($s = 0, 1, \ldots, 15$). (S_i:n rivit ja sarakkeet siis indeksoidaan nollasta lähtien.) Silloin $S_i(B_i)$ on S_i:n r:nnen rivin ja s:nnen sarakkeen risteyskohdassa olevan luvun binääriesitys, jonka alkuun lisätään tarvittaessa nollia, niin että saadaan neljä bittiä. Bittijonot C_i katenoidaan bittijonoksi
$$C = C_1C_2C_3C_4C_5C_6C_7C_8.$$

4. 32-pituinen bittijono C permutoidaan käyttäen kiinteää permutaatiota π. Näin saatu bittijono $\pi(C)$ on $f(R, K)$.

Havainnollisesti operaatio on seuraavanlainen:



Tässä voidaan huomata, että E ja π ovat lineaarisia operaatioita, ts. ne voitaisiin korvata bittivektorin kertomisella matriisilla. Toisaalta S-laatikot eivät ole lineaarisia, vaan epälineaarisia.

S-laatikoiden määrittelyt löytyvät kirjallisuudesta (esimerkiksi STINSON). Annetaan esimerkkinä S_2:

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

Permutaatiot π_{ini} ja π ovat seuraavat (annettuna samaan tapaan kuin E):

π_{ini} :

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

π :

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Avainjono $k_1, k_2, \ldots, k_{16}$ lasketaan iteratiivisesti seuraavasti:

1. Avain k annetaan muodossa, missä aina 7 bitin perään on lisätty pariteetintarkistusbitti. Tavussa on näin aina pariton määrä 1:siä ja avaimen pituus 64 bittiä. Mikäli pariteetintarkistus osoittaa, että avaimessa on virheitä, sitä ei oteta käyttöön. Jos taas virheitä ei ole, poistetaan pariteetintarkistusbitit, jolloin tullaan alkuperäiseen 56-bittiseen avaimeen. Avaimeen sovelletaan ensin kiinteää bittien permutaatiota π_{K1}. Kirjoitetaan

   $$\pi_{\text{K1}}(k) = C_0 D_0,$$

   missä C_0 ja D_0 ovat 28-pituisia bittijonoja.

2. Lasketaan jono $C_1D_1, C_2D_2, \ldots, D_{16}D_{16}$ iteroiden 16 kertaa seuraavaa menettelyä:

   $$\begin{cases} C_i = \sigma_i(C_{i-1}) \\ D_i = \sigma_i(D_{i-1}), \end{cases}$$

   missä σ_i on bittijonon syklinen siirto 1 tai 2 bittiä vasemmalle. Jos $i = 1, 2, 9, 16$, niin siirto on 1 bitti, muuten 2 bittiä.

3. Sovelletaan C_iD_i:hin kiinteää 48 bitin variaatiota π_{K2}. Näin saadaan $k_i = \pi_{\text{K2}}(C_iD_i)$.

Vielä pitää antaa permutaatio π_{K1} ja variaatio π_{K2}:

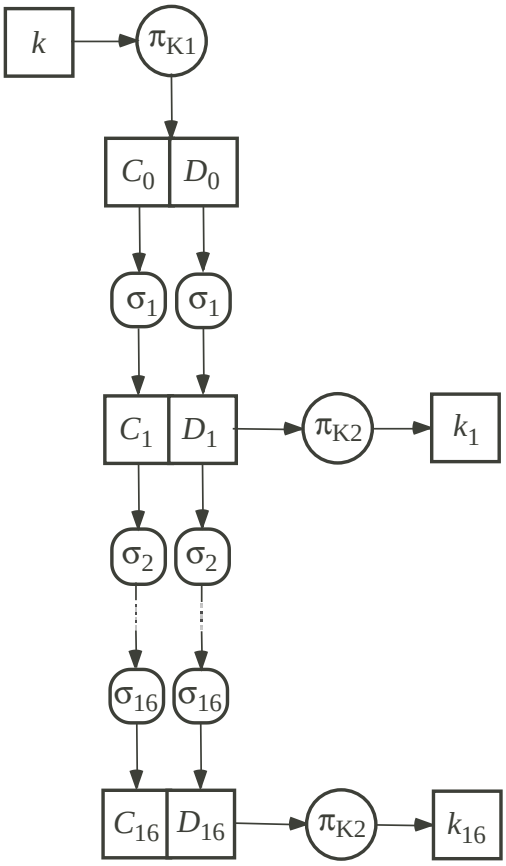
π_{K1} :

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

π_{K2} :

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Havainnollisesti avainten generointi on seuraavanlainen:



Dekryptaus sujuu oleellisesti samalla systeemillä, mutta käyttäen avainjonoa $k_1, k_2, \ldots, k_{16}$ päinvastaisessa järjestyksessä ja kääntäen permutaatiot. Tällöin

$$\begin{cases} L_{i-1} = R_i \oplus f(L_i, k_i) \\ R_{i-1} = L_i. \end{cases}$$

Tavallinen DESin käyttötapa on kryptata pitkä viesti lohko kerrallaan samalla avaimella, ns. *ECB-moodi* (electronic codebook). Toinen tapa, ns. *CBC-moodi* (cipher block chaining) on muodostaa aina viestilohkon w ja edellisen kryptolohkon c biteittäinen summa modulo 2 eli $w \oplus c$ ja kryptata se, käyttäen koko ajan samaa avainta. Aivan alussa tarvitaan tietty aloituslohko edellisen kryptolohkon tilalla. CBC-moodissa muutos jossain viestilohkossa aiheuttaa muutoksia sitä seuraavissa kryptolohkoissa. CBC-moodia voidaan näin käyttää *autentikointiin* eli ns. *MACiin* (message authentication code) seuraavasti. Aloituslohko voisi olla vaikka pelkästään 0-biteistä koostuva. Lähettäjällä on viestilohkoista $w_1, \ldots, w_n$ koostuva viesti ja hän laskee CBC-moodilla vastaavat kryptolohkot $c_1, \ldots, c_n$ käyttäen tiettyä salaista avainta k. Lähettäjä lähettää viestilohkot sekä c_n:n vastaanottajalle. Vastaanottajalla on myös avain k ja hän pystyy sitä käyttäen tarkistamaan onko c_n oikea.

Ns. *OFB-moodissa* (output feedback) käytetään DESiä avaimen muuntamiseen ja kryptaukseen ONE-TIME-PADin tapaista menettelyä. Lähtien tietystä ”alkuavaimesta” κ_0 saadaan avainvuo $\kappa_1, \ldots, \kappa_n$ kryptaamalla DESillä toistuvasti tätä avainta, κ_1 saadaan kryptaamalla κ_0. Kryptatessa käytetään jälleen koko ajan samaa salaista avainta k. OFB-moodista saadaan variantti, ns. *CFB-moodi* (cipher feedback), kun avainvuon avain κ_i muodostetaan kryptaamalla edeltävä kryptolohko. Jälleen κ_1 saadaan kryptaamalla aloituslohko. Tätä varianttia voidaan käyttää autentikointiin kuten CBC-moodiakin.

3.5.3 DESin kryptanalyysiä

DESin rakenteessa kaikki muu on lineaarista, eli toteutettavissa matriisikertolaskulla, paitsi S-laatikot. Jos S-laatikot olisivat affiinisia, eli toteutettavissa matriisikertolaskuilla ja vektorien yhteenlaskuilla, olisi DES oleellisesti sama kuin jokin AFFINE-HILL ja näin helpohkosti murrettavissa. S-laatikot eivät kuitenkaan ole affiinisia. Julkisuudessa esitetyt DESin S-laatikoiden suunnitteluperiaatteet[3] ovat seuraavat:

(1) S-laatikon kukin rivi on lukujen $0, 1, \ldots, 15$ permutaatio.

(2) S-laatikko ei ole syötteidensä affiininen funktio (eikä siis myöskään lineaarinen funktio). Itse asiassa vaaditaan, että mikään S-laatikon tulostusbitti ei ole ”lähellä” lineaarista syötebittien funktiota.

(3) Yhden bitin muuttaminen S-laatikon syötteessä aiheuttaa tulostuksessa ainakin kahden bitin muutoksen.

(4) S-laatikon tulostukset syötteillä x ja $x \oplus 001100$ eroavat ainakin kahden bitin osalta, olipa x mikä tahansa 6-pituinen bittijono.

(5) S-laatikon tulostukset syötteillä x ja $x \oplus 11b_1b_200$ eroavat, olipa x mikä tahansa 6-pituinen bittijono ja b_1 sekä b_2 mitä tahansa bittejä.

(6) Jokaista kuuden bitin jonoa $B = b_1b_2b_3b_4b_5b_6 \neq 000000$ kohti on 32 ($= 2^6/2$) eri syöteparia x_1, x_2, joille $x_1 \oplus x_2 = B$. Vastaavista 32 tulosteparista y_1, y_2 enintään kahdeksalla saa olla sama summa $y_1 \oplus y_2$.

DESin avaimia on

$$2^{56} = 72\,057\,594\,037\,927\,936$$

eli nykykäsityksen mukaan varsin vähän. Tämä antaa mahdollisuuden seuraavaan yksinkertaiseen KP-hyökkäykseen: Jos tunnetaan selväteksti w ja vastaava kryptoteksti c, kokeillaan läpi avaimia, kunnes löytyy avain, jolla saadaan aikaan tämä kryptaus. (Tällaisia avaimia voi kuitenkin olla useita.) Menettely ei vaadi muuta kuin aikaa ja nopeita prosessoreja ja on helposti rinnakkaistettavissa, muistia se vaatii minimaalisen määrän. DES on toteutettavissa tavattoman nopeina piireinä, joten varta vasten sen murtamiseen yo. tekniikalla suunnitellut prosessorit ovat mahdollisia.

CP-hyökkäys saadaan seuraavasti: Valitaan selväteksti w ja kryptataan se kaikilla mahdollisilla avainavaruuden avaimilla. Taulukoidaan tulokset. Jos nyt murrettavalla DESillä voidaan kryptata w ja saada vastaava kryptoteksti, niin taulukkoetsinnällä löytyy avain. Tämä metodi on tietysti edullinen vain, jos sitä käytetään useiden avainten etsimiseen, jolloin taulukkoa voidaan käyttää toistuvasti. Menettely ei vaadi lisäaikaa (taulukon laatimisen jälkeen) juuri lainkaan, mutta paljon muistia.

On myös menettelyjä, joissa aikaa voidaan vaihtaa muistiin, eräänlaisia yo. menettelyjen välimuotoja, ks. esimerkiksi STINSON. Parannetussa DESin versiossa AES (Advanced Encryption Standard) avaimia on (ainakin)

$$2^{128} = 340\,282\,366\,920\,938\,463\,463\,374\,607\,431\,768\,211\,456,$$

minkä katsotaan riittävän estämään yllä olevien tapaiset hyökkäykset. AES perustuu belgialaiskryptologien Joan Daemenin ja Vincent Rijmenin vuonna 1999 kehittämään kryptosysteemiin RIJNDAEL, josta enemmän seuraavassa luvussa.

[3] S-laatikoita ja muita epälineaarisia elementtejä käytetään muissakin kryptosysteemeissä. Näiden suunnitteluperusteita on selvittänyt mm. suomalaismatemaatikko Kaisa Nyberg, ks. myös Luku 4.

Pykälässä 3.4 esitetty AFFINE-HILLin (ja itse asiassa myös AFFINEn) KP-hyökkäys käytti murtamiseen selvätekstien erotuksia ja vastaavien kryptotekstien erotuksia (modulo M), joilla saatiin hävitetyksi affiinisuuden aiheuttama epälineaarisuus. Tällaista menettelyä kutsutaan *differentiaaliseksi kryptanalyysiksi.* Vastaavantapaista menettelyä voidaan soveltaa DESin KP- ja CP-hyökkäyksissä ja tietyssä määrin poistaa S-laatikoiden epälineaarisuuden vaikutuksia. Haittapuolena on suuri tarvittavien selväteksti-kryptoteksti-parien lukumäärä. Ks. esimerkiksi STINSON.

Lineaarinen kryptanalyysi taas pyrkii käyttämään hyväksi joidenkin syöte- ja tulostusbittien välisiä lineaarisia riippuvuuksia, jotka pitävät paikkansa vain osalle syötteistä. Tällaisia on DESissä eikä niitä ilmeisesti osattu alun perin välttää.

Luku 4

AES: RIJNDAEL

4.1 Ekskursio algebraan 1

4.1.1 Renkaat ja kunnat

Algebrallinen struktuuri koostuu joukosta A, jonka alkioille on määritelty yksi tai useampi laskuoperaatio ja näille laskusäännöt. Lisäksi yleensä jo(i)llekin A:n alkio(i)lle on varattu erikoisrooli.

Rengas on struktuuri $R = (A, \oplus, \odot, \mathbb{0}, \mathbb{1})$, jossa $\oplus$ on renkaan *yhteenlaskuoperaatio,* $\odot$ on renkaan *kertolaskuoperaatio,* $\mathbb{0}$ on renkaan *nolla-alkio* ja $\mathbb{1}$ on renkaan *ykkösalkio* (ja $\mathbb{0} \neq \mathbb{1}$). Lisäksi vaaditaan, että seuraavat ehdot ovat voimassa:

(1) $\oplus$ ja $\odot$ ovat vaihdannaisia operaatioita, ts. aina

$$a \oplus b = b \oplus a \quad \text{ja} \quad a \odot b = b \odot a.$$

(2) $\oplus$ ja $\odot$ ovat liitännäisiä operaatioita, ts. aina

$$(a \oplus b) \oplus c = a \oplus (b \oplus c) \quad \text{ja} \quad (a \odot b) \odot c = a \odot (b \odot c).$$

Liitännäisyydestä seuraa, että pitkät summa- ja tulolausekkeet voidaan suluttaa miten tahansa tuloksen muuttumatta. Usein ne kirjoitetaan täysin ilman sulkeita, esimerkiksi $a_1 \oplus a_2 \oplus \cdots \oplus a_k$ tai $a_1 \odot a_2 \odot \cdots \odot a_k$. Erityisesti otetaan käyttöön monikerta- ja potenssimerkinnät

$$ka = \underbrace{a \oplus \cdots \oplus a}_{k \text{ kpl}} \quad \text{ja} \quad a^k = \underbrace{a \odot \cdots \odot a}_{k \text{ kpl}}$$

(sekä $0a = \mathbb{0}$, $1a = a$, $a^0 = \mathbb{1}$ ja $a^1 = a$).

(3) $\mathbb{0} \oplus a = a$ ja $\mathbb{1} \odot a = a$ (huomaa, miten nämä sopivat yhteen eo. monikerta- ja potenssimerkintöjen kanssa).

(4) $a \odot (b \oplus c) = (a \odot b) \oplus (a \odot c)$ (osittuvuus).

(5) Jokaiselle alkiolle a on *vasta-alkio* $-a$, jolle $(-a) \oplus a = \mathbb{0}$. Vasta-alkiota käyttäen saadaan *vähennyslasku* $a \ominus b = a \oplus (-b)$ ja negatiivinen monikerta $(-k)a = k(-a)$.

Huomautus. *Tarkemmin sanoen tällainen R on ns.* vaihdannainen 1-rengas, *varsinainen rengas on algebrallisesti vieläkin yleisempi käsite. Ks. kurssi Algebra 1. Jatkossa renkaasta puhuttaessa kyseessä on siis aina juuri tällainen vaihdannainen 1-rengas.*

Jos lisäksi vielä seuraava ehto (6) pätee, on kyseessä *kunta:*

(6) Jokaiselle alkiolle $a \neq \mathbb{0}$ on *käänteisalkio* a^{-1}, jolle $a \odot a^{-1} = 1$. Käänteisalkiota käyttäen saadaan jakolasku $a/b = a \odot b^{-1}$ ja negatiivinen potenssi $a^{-k} = (a^{-1})^k$.

Tavallisesti sovitaan, että kerto- ja jakolaskut tehdään aina ennen yhteen- ja vähennyslaskuja, jolloin sulkuja voidaan jättää pois. Näistä ehdoista voidaan johtaa monet ”tutut” laskusäännöt, esimerkiksi

$$-(a \odot b) = (-a) \odot b \quad \text{ja} \quad \frac{a \odot b}{c \odot d} = \frac{a}{c} \odot \frac{b}{d}.$$

Jokainen kunta on siis myös rengas. Tuttuja renkaita, jotka eivät ole kuntia, ovat mm. kokonaislukujen rengas $\mathbb{Z}$ sekä erilaiset *polynomirenkaat*, esimerkiksi rationaali-, reaali-, kompleksi- ja kokonaiskertoimisten polynomien renkaat $\mathbb{Q}[x]$, $\mathbb{R}[x]$, $\mathbb{C}[x]$ ja $\mathbb{Z}[x]$. Näissä laskuoperaatiot ovat tutut $+$ ja $\cdot$, nolla-alkio on nolla ja ykkösalkio on 1. Myös $\mathbb{Z}_m$ (jäännösluokat modulo m) muodostavat renkaan, eli jäännösluokkarengas on rengas, ks. pykälä 2.5.

Tuttuja kuntia ovat *lukukunnat* reaalilukujen kunta $(\mathbb{R}, +, \cdot, 0, 1)$, rationaalilukujen kunta $(\mathbb{Q}, +, \cdot, 0, 1)$ ja kompleksilukujen kunta $(\mathbb{C}, +, \cdot, 0, 1)$ sekä reaalikertoimisten rationaalifunktioiden kunta $(\mathbb{R}(x), +, \cdot, 0, 1)$ ja *alkukunnat* $(\mathbb{Z}_p, +, \cdot, \overline{0}, \overline{1})$ (ks. pykälä 2.5). Näitä merkitään yleensä vain lyhyesti: $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{C}$, $\mathbb{R}(x)$ ja $\mathbb{Z}_p$.

4.1.2 Kuntien polynomirenkaat

Polynomit, jotka muodostetaan formaalisesti käyttäen kunnan F alkioita kertoimina, muodostavat ns. F:n *polynomirenkaan* $F[x]$. Polynomi kirjoitetaan tuttuun muotoon

$$p(x) = a_0 \oplus a_1 x \oplus a_2 x^2 \oplus \cdots \oplus a_n x^n \,, \quad \text{missä } \; a_0, a_1, \ldots, a_n \in F \; \text{ ja } \; a_n \neq \mathbb{0}.$$

Tavalliseen tapaan $F[x]$:n nollapolynomi samaistetaan F:n nolla-alkioon $\mathbb{0}$ ja vakiopolynomit vastaaviin F:n alkioihin. Polynomin $p(x)$ aste $\deg(p(x))$ määritellään edelleen tavalliseen tapaan korkeimman esiintyvän x:n potenssin eksponentiksi (yllä aste on n). Nollapolynomin asteeksi voidaan sopia vaikkapa -1. Korkeimman esiintyvän x:n potenssin kerrointa kutsutaan *johtavaksi kertoimeksi* (yllä a_n). Jos johtava kerroin on $= 1$, kyseessä on ns. *pääpolynomi*.

Polynomien yhteen- vähennys- ja kertolasku määritellään tavalliseen tapaan käyttäen kertoimia ja kunnan vastaavia laskuoperaatioita. Polynomeille $a(x)$ ja $m(x) \neq \mathbb{0}$ on määritelty yksikäsitteinen *osamäärä* $q(x)$ ja *jakojäännös* $r(x)$, ts. voidaan kirjoittaa

$$a(x) = q(x) \odot m(x) \oplus r(x) \,, \quad \deg(r(x)) < \deg(m(x)).$$

Käytännössä tällainen jakolasku voidaan suorittaa ”pitkänä jakolaskuna” paperilla, myös esimerkiksi Maple-ohjelmisto osaa jakaa polynomeja monissa kunnissa. $r(x)$:n sanotaan olevan $a(x)$:n *jäännös modulo* $m(x)$, merkitään $\overline{a(x)}$, vrt. vastaava käsite kokonaisluvuille pykälässä 2.4. $m(x)$ on ns. *moduli*. Käytännössä oletetaan modulin olevan vähintään astetta 1. Kaikki jäännökset modulo $m(x)$ muodostavat ns. *jäännösluokkarenkaan* eli *tekijärenkaan* $F[x]/\langle m(x) \rangle$. On helppo nähdä, samaan tapaan kuin kokonaisluvuille, että jäännökset modulo $m(x)$ voidaan antaa ja niillä voidaan laskea ”edustajan välityksellä”, ts.

$$\begin{aligned}
&\overline{a_1(x)} \oplus \overline{a_2(x)} = \overline{a_1(x) \oplus a_2(x)} \quad , \quad -\overline{a(x)} = \overline{-a(x)} \,, \\
&\overline{a_1(x)} \ominus \overline{a_2(x)} = \overline{a_1(x) \ominus a_2(x)} \quad , \quad k\overline{a(x)} = \overline{ka(x)} \,, \\
&\overline{a_1(x)} \odot \overline{a_2(x)} = \overline{a_1(x) \odot a_2(x)} \quad \text{ja} \quad \overline{a(x)}^k = \overline{a(x)^k}.
\end{aligned}$$

Jakojäännöstä $r(x)$ vastaavan jäännösluokan edustajia ovat ne polynomit $a(x)$, joiden jäännös modulo $m(x)$ on $r(x)$. Jäännösluokka voidaan myös samaistaa edustajiensa muodostamaan joukkoon. Näin ollen $F[x]/\langle m(x)\rangle$ on todellakin rengas.

Edelleen voidaan määritellä tekijä ja jaollisuus kuten pykälässä 2.1. Polynomia, jolla ei ole muita alempiasteisia tekijöitä kuin vakiopolynomit, kutsutaan *jaottomaksi.*

Aivan samalla tavoin kuin tehtiin osoitettaessa, että jokaisella muulla $\mathbb{Z}_p$:n alkiolla kuin nolla-alkiolla $\overline{0}$ on käänteisalkio, voidaan näyttää, että jokaisella muulla $F[x]/\langle p(x)\rangle$:n alkiolla kuin nolla-alkiolla $\overline{\mathbb{0}}$ on käänteisalkio olettaen, että moduli $p(x)$ on jaoton polynomi. Tätä varten tarvitaan $F[x]$:n polynomien suurin yhteinen tekijä sekä Eukleideen algoritmi.

$F[x]$:n polynomien $a(x)$ ja $b(x)$ (eivät molemmat nollapolynomeja) *suurin yhteinen tekijä* (s.y.t) on korkeinta astetta oleva oleva polynomi $d(x)$, joka jakaa tasan sekä $a(x)$:n että $b(x)$:n, merkitään $d(x) = \mathrm{syt}(a(x), b(x))$. Huomaa, että tällainen suurin yhteinen tekijä ei ole yksikäsitteinen, sillä jos $d(x) = \mathrm{syt}(a(x), b(x))$, niin myös esimerkiksi $-d(x)$ on $\mathrm{syt}(a(x), b(x))$. (Usein vaaditaankin lisäksi, että $d(x)$ on pääpolynomi.)

Lause 4.1. (Bezout'n lause) *Jos ainakin toinen polynomeista $a(x)$ ja $b(x)$ ei ole nollapolynomi, niin niiden s.y.t. $d(x)$ voidaan kirjoittaa muotoon*

$$d(x) = c_1(x) \odot a(x) \oplus c_2(x) \odot b(x) \quad \textit{(}\text{Bezout'n muoto}\textit{)}.$$

Todistus. Todistus on hyvin samantapainen kuin Lauseen 2.5. Merkitään $\mathrm{SYT}(a(x), b(x)) = (d(x), c_1(x), c_2(x))$ ja oletetaan, että $\deg(a(x)) \leq \deg(b(x))$. ***(Yleistetty) Eukleideen algoritmi*** on seuraava rekursio:

(Yleistetty) Eukleideen algoritmi polynomeille:

1. Jos $a(x) = \mathbb{0}$, niin tulostetaan $\mathrm{SYT}(a(x), b(x)) = (b(x), \mathbb{0}, \mathbb{1})$ ja lopetetaan.

2. Jos $a(x) \neq \mathbb{0}$ on vakiopolynomi, tulostetaan $\mathrm{SYT}(a(x), b(x)) = (a(x), \mathbb{1}, \mathbb{0})$ ja lopetetaan.

3. Jos $\deg(a(x)) \geq 1$, niin etsitään $b(x)$:n jäännös $r(x)$ modulo $a(x)$, ts. kirjoitetaan $b(x) = q(x) \odot a(x) \oplus r(x)$, missä $\deg(r(x)) < \deg(a(x))$. Etsitään sitten $\mathrm{SYT}(r(x), a(x)) = (d(x), e_1(x), e_2(x))$. Koska $d(x) = e_1(x) \odot r(x) \oplus e_2(x) \odot a(x)$, on silloin $d(x) = \mathrm{syt}(a(x), b(x))$ ja

   $$d(x) = (e_2(x) \ominus e_1(x) \odot q(x)) \odot a(x) \oplus e_1(x) \odot b(x).$$

   Tulostetaan $\mathrm{SYT}(a(x), b(x)) = (d(x), e_2(x) \ominus e_1(x) \odot q(x), e_1(x))$ ja lopetetaan.

Rekursio on päättyvä, koska $\min(\deg(r(x)), \deg(a(x))) < \min(\deg(a(x), \deg(b(x)))$, ts. aina kutsuttaessa SYT ko. minimiarvo pienenee. □

Jos $\mathrm{syt}(a(x), b(x))$ on vakio $f \neq \mathbb{0}$, niin Bezout'n muodosta saadaan jakamalla puolittain f:llä

$$\mathbb{1} = e_1(x) \odot a(x) \oplus e_2(x) \odot p(x).$$

Näin ollen $a(x)$:llä on tällöin inverssi $e_1(x)$ modulo $b(x)$ eli $\overline{a(x)}$:llä on $F[x]/\langle b(x)\rangle$:ssä käänteisalkio $\overline{e_1(x)}$. (Olettaen, että $\deg(b(x)) \geq 1$. Samalla tuli esitetyksi menetelmä, jolla käänteisalkion voi löytää.)

Erityisesti, jos $p(x)$ on jaoton vähintään astetta 1 oleva $F[x]$:n polynomi, niin $F[x]/\langle p(x)\rangle$ on kunta. Ko. kunnan alkiot on tapana kirjoittaa muotoon

$$c_0 \oplus c_1 x \oplus c_2 x^2 \oplus \cdots \oplus c_{n-1} x^{n-1},$$

missä $n = \deg(p(x))$ ja kertoimet $c_0, c_1, \ldots, c_{n-1}$ ovat F:n alkioita, eli oleellisesti n-vektoreiksi, joiden komponentit ovat F:ssä. Erityisesti huomataan, että jos $p(x)$ on ensimmäistä astetta, niin $F[x]/\langle p(x)\rangle = F$, ts. palataan takaisin alkuperäiseen kuntaan.

Esimerkki. *$\mathbb{R}[x]$:n jaottomat polynomit (vakioita lukuunottamatta) ovat tunnetusti joko ensimmäistä tai toista astetta. Edellisistä saadaan $\mathbb{R}$ ja jälkimmäisistä $\mathbb{C}$. Siis esimerkiksi $\mathbb{C} = \mathbb{R}[x]/\langle x^2+1\rangle$. $\mathbb{C}[x]$:n jaottomat polynomit ovatkin sitten vakioita tai ensimmäistä astetta.*

4.1.3 Äärelliset kunnat

Alkukunnat saatiin pykälässä 2.5 muodossa $\mathbb{Z}_p$ eli jäännösluokkina modulo jokin alkuluku p. Alkukunta on esimerkki *äärellisestä kunnasta*, mutta niitä on muitakin. Näiden saamiseksi valitaan jonkin alkukunnan $\mathbb{Z}_p$ polynomirenkaan $\mathbb{Z}_p[x]$ polynomi $P(x)$, joka on jaoton (pää)polynomi. Jäännökset modulo $P(x)$ muodostavat kunnan $\mathbb{Z}_p[x]/\langle P(x)\rangle$, jonka alkiot esitetään tavallisesti muodossa

$$c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1}x^{n-1},$$

missä $n = \deg(P(x))$ ja $c_0, \ldots, c_{n-1} \in \mathbb{Z}_p$. Tämä kunta on äärellinen, siinä on alkioita yhtä monta kuin on jäännöksiä modulo $P(x)$ eli p^n kpl.

Voidaan näyttää (sivuutetaan tässä), että kaikki äärelliset kunnat saadaan tällä tavoin (tietysti myös alkukunta $\mathbb{Z}_p$ itse). Äärellisen kunnan alkioiden lukumäärä on siis aina alkuluvun potenssi. Äärellisiä kuntia voidaan konstruoida monella tavalla, samaa astetta olevia jaottomia $\mathbb{Z}_p[x]$:n polynomejakin löytyy yleensä useampia, mutta aina äärellinen kunta, jossa on p^n alkiota, on rakenteeltaan samanlainen eli isomorfinen jonkin kunnan $\mathbb{Z}_p[x]/\langle P(x)\rangle$:n kanssa, missä $\deg(P(x)) = n$. Näin ollen äärellisiä kuntia, joissa on p^n alkiota, on olennaisesti vain yksi, ja sitä merkitään $\mathbb{F}_{p^n}$:llä tai $\mathrm{GF}(p^n)$:llä.[1] Jokaista mahdollista alkuluvun potenssia p^n kohti on edelleen olemassa $\mathbb{F}_{p^n}$, ts. jokaisesta polynomirenkaasta $\mathbb{Z}_p[x]$ löytyy kaikkia astelukuja $n \geq 1$ olevia jaottomia polynomeja.

Esimerkiksi $\mathbb{F}_{2^8}$:n konstruoimiseksi voidaan valita $\mathbb{Z}_2[x]$;n astetta 8 oleva jaoton polynomi $P(x) = 1 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8$. Tarkistetaan Maplella, että $P(x)$ todella on jaoton:

```
> Irreduc(1+x^3+x^4+x^5+x^6+x^7+x^8) mod 2;
```

$$true$$

$\mathbb{F}_{2^8}$:n alkiot ovat esitettävissä muodossa

$$b_0 + b_1 x + b_2 x^2 + b_3 x^3 + b_4 x^4 + b_5 x^5 + b_6 x^6 + b_7 x^7,$$

missä $b_0, \ldots, b_7$ ovat bittejä. Maplen `GF`-paketilla voidaan laskea äärellisissä kunnissa, tosin vähän kankeasti. Kokeillaan sitä $\mathbb{F}_{2^8}$:ssa:

```
> readlib(GF):
> G256:=GF(2,8,1+x^3+x^4+x^5+x^6+x^7+x^8):
> a:=G256[ConvertIn](x);
```

$$a := x$$

```
> G256['^'](a,1200);
```

$$x + x^2 + x^4$$

```
> c:=G256[inverse](a);
```

[1] "GF" = "Galois field" (*Galois'n kunta*).

$$c := x^2 + x^3 + x^4 + x^5 + x^6 + x^7$$

```
> G256[‘+‘](a,G256[‘^‘](c,39));
```

$$1 + x + x^3 + x^6 + x^7$$

Käsky `ConvertIn` muuntaa polynomin Maplen sisäiseen esitykseen. Ellei tiedä yhtään sopivaa jaotonta $\mathbb{Z}_p[x]$:n polynomia, etsii Maple kyllä jonkin sellaisen:

```
> G81:=GF(3,4):
> G81[extension];
```

$$2 + 2T^2 + T^4$$

Valinnan saa selville `extension`-käskyllä. Tässä siis saatiin tuloksena $\mathbb{Z}_3$[x]:n jaoton polynomi $\overline{2} + \overline{2}x^2 + x^4$.

Paitsi kryptologiassa, äärelliset kunnat ovat tavattoman tärkeitä virheitä korjaavassa koodauksessa. Niitä käsitelläänkin enemmän kursseissa Finite Fields ja Koodausteoria. Hyviä viitteitä ovat MCELIECE ja LIDL & NIEDERREITER ja myös GARRETT.

4.2 RIJNDAEL

RIJNDAEL-systeemissä lohkon pituus l_{B} ja avaimen pituus l_{K} ovat (toisistaan riippumatta) joko 128, 192 tai 256 bittiä. Jakamalla 32:lla saadaan luvut

$$N_{\mathrm{B}} = \frac{l_{\mathrm{B}}}{32} \quad \text{ja} \quad N_{\mathrm{K}} = \frac{l_{\mathrm{K}}}{32}.$$

Bittejä käsitellään 8-bittisinä tavuina. 8-bittistä tavua $b_0 b_1 \cdots b_7$ taas voidaan pitää äärellisen kunnan $\mathbb{F}_{2^8}$ alkiona.

Avain on tapana esittää $4 \times N_{\mathrm{K}}$-matriisina, jonka alkiot ovat tavuja, seuraavasti: Jos avain on tavuittain

$$k_{00} k_{10} k_{20} k_{30} k_{01} k_{11} k_{21} \cdots k_{3,N_{\mathrm{K}}-1},$$

niin vastaava matriisi on

$$\begin{pmatrix} k_{00} & k_{01} & k_{02} & \cdots & k_{0,N_{\mathrm{K}}-1} \\ k_{10} & k_{11} & k_{12} & \cdots & k_{1,N_{\mathrm{K}}-1} \\ k_{20} & k_{21} & k_{22} & \cdots & k_{2,N_{\mathrm{K}}-1} \\ k_{30} & k_{31} & k_{32} & \cdots & k_{3,N_{\mathrm{K}}-1} \end{pmatrix}.$$

Huomaa miten matriisin alkiot indeksoidaan nollasta alkaen. Vastaavasti, jos syötelohko on tavuittain

$$a_{00} a_{10} a_{20} a_{30} a_{01} a_{11} a_{21} \cdots a_{3,N_{\mathrm{B}}-1},$$

niin vastaava matriisi on

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & \cdots & a_{0,N_{\mathrm{B}}-1} \\ a_{10} & a_{11} & a_{12} & \cdots & a_{1,N_{\mathrm{B}}-1} \\ a_{20} & a_{21} & a_{22} & \cdots & a_{2,N_{\mathrm{B}}-1} \\ a_{30} & a_{31} & a_{32} & \cdots & a_{3,N_{\mathrm{B}}-1} \end{pmatrix}.$$

Kryptauksen aikana käsiteltävänä on joka vaiheessa jokin l_{B}-pituinen bittijono, ns. *tila*. Lohkon tapaan se esitetään tavuittain $4 \times N_{\mathrm{B}}$-matriisin muodossa:

$$\begin{pmatrix} b_{00} & b_{01} & b_{02} & \cdots & b_{0,N_{\mathrm{B}}-1} \\ b_{10} & b_{11} & b_{12} & \cdots & b_{1,N_{\mathrm{B}}-1} \\ b_{20} & b_{21} & b_{22} & \cdots & b_{2,N_{\mathrm{B}}-1} \\ b_{30} & b_{31} & b_{32} & \cdots & b_{3,N_{\mathrm{B}}-1} \end{pmatrix}.$$

Em. matriisien alkiot ovat 8 bitin tavuja, jotka ovat tulkittavissa kunnan $\mathbb{F}_{2^8}$ alkioiksi. Näin matriisit ovat ko. kunnan matriiseja. Toinen tapa tulkita matriiseja on ajatella näiden sarakkeet 4-pituisiksi kunnan $\mathbb{F}_{2^8}$ alkioiden jonoiksi. Nämä taas voidaan tulkita polynomirenkaan $\mathbb{F}_{2^8}[z]$ enintään astetta 3 olevien polynomien kertoimiksi (ylhäältä alas lukien), ts. tällaisiksi polynomeiksi. (Käytetään tässä polynomin muuttujana z:aa, jottei se sekaannu kunnan esityksessä käytettävään x:ään.) Jotta esitys olisi yksikäsitteinen, pitää $\mathbb{F}_{2^8}$:n konstruktiossa käyttää sovittua kiinteää jaotonta astetta 8 olevaa $\mathbb{Z}_2[x]$:n polynomia. RIJNDAELissa se on ns. ***RIJNDAEL-polynomi***

$$p(x) = 1 + x + x^3 + x^4 + x^8.$$

DESin tapaan RIJNDAELissa on tietty määrä N_{R} ns. *kierroksia*. Kierrosten lukumäärän antaa seuraava taulu:

N_{R}	$N_{\mathrm{B}} = 4$	$N_{\mathrm{B}} = 6$	$N_{\mathrm{B}} = 8$
$N_{\mathrm{K}} = 4$	10	12	14
$N_{\mathrm{K}} = 6$	12	12	14
$N_{\mathrm{K}} = 8$	14	14	14

Kullakin kierroksella, viimeistä lukuunottamatta, tehdään pseudokoodina esitettynä seuraavat operaatiot:

```
Round(State,RoundKey)
{
ByteSub(State);
ShiftRow(State);
MixColumn(State);
AddRoundKey(State,RoundKey);
}
```

Viimeisellä kierroksella jätetään pois sarakkeiden sekoitus:

```
FinalRound(State,RoundKey)
{
ByteSub(State);
ShiftRow(State);
AddRoundKey(State,RoundKey);
}
```

Salausavain laajennetaan ensin ja jaetaan sitten eri kierroksille. Tämä ja kierroksen eri operaatiot käydään läpi yksitellen seuraavissa pykälissä. Itse kryptaus koostuu sitten seuraavista vaiheista:

- Suoritetaan `AddRoundKey` syötelohkolle ja 0. kierrosavaimelle.
- $N_R - 1$ ”tavallista” kierrosta.
- Viimeinen kierros.

Dekryptattaessa käydään vaiheet läpi käänteisinä käänteisessä järjestyksessä.

4.2.1 Tavun muuntaminen (`ByteSub`)

Tässä operaatiossa kukin tilan (alussa syötelohkon) tavu b_{ij} muunnetaan seuraavasti:

1. Tulkitaan b_{ij} kunnan $\mathbb{F}_{2^8}$ alkioksi ja lasketaan sen käänteisalkio b_{ij}^{-1}. Nolla-alkion käänteisalkioksi sovitaan tässä se itse.

2. Avataan b_{ij}^{-1} kahdeksaksi bitiksi $b_0b_1b_2b_3b_4b_5b_6b_7$, merkitään

   $$b(x) = b_0 + b_1x + b_2x^2 + b_3x^3 + b_4x^4 + b_5x^5 + b_6x^6 + b_7x^7 \quad (\mathbb{Z}_2[x]\text{:n polynomi})$$

   ja lasketaan

   $$b'(x) \equiv b(x)(1 + x^4 + x^5 + x^6 + x^7) + (x + x^2 + x^6 + x^7) \mod 1 + x^8.$$

   Tulos

   $$b'(x) = b'_0 + b'_1x + b'_2x^2 + b'_3x^3 + b'_4x^4 + b'_5x^5 + b'_6x^6 + b'_7x^7$$

   tulkitaan tavuksi $b'_0b'_1b'_2b'_3b'_4b'_5b'_6b'_7$ (tai $\mathbb{F}_{2^8}$:n alkioksi). Jakaminen $\mathbb{Z}_2[x]$:ssä $1 + x^8$:lla on helppoa, sillä

   $$x^k \equiv x^{(k, \bmod 8)} \mod 1 + x^8.$$

Kohdan 2. operaatio voidaan ajatella myös matriisien avulla. Silloin tehdään $\mathbb{Z}_2$:ssa affiini muunnos

$$\begin{pmatrix} b'_7 \\ b'_6 \\ b'_5 \\ b'_4 \\ b'_3 \\ b'_2 \\ b'_1 \\ b'_0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

Dekryptauksessa operaatio tehdään käänteisessä järjestyksessä, $\mathbb{Z}_2[x]$:ssä on nimittäin

$$1 = \text{syt}(1 + x^4 + x^5 + x^6 + x^7, 1 + x^8),$$

joten polynomilla $1 + x^4 + x^5 + x^6 + x^7$ on inverssi modulo $1 + x^8$ (ja esiintyvä 8×8-matriisi on kääntyvä modulo 2). Ko. inverssi on $x^2 + x^5 + x^7$.

Tavun muuntaminen on kokonaisuutena epälineaarinen tavumuunnos, joka voidaan antaa yhtenä taulukkona, ns. *RIJNDAELin S-laatikkona*. Tällainen taulukko löytyy mm. MOLLINista.

4.2.2 Rivinsiirto (`ShiftRow`)

Tässä operaatiossa tilan matriisiesityksen rivien alkiot siirretään syklisesti vasemmalle seuraavasti:

siirto	0. rivi	1. rivi	2. rivi	3. rivi
$N_\mathrm{B} = 4$	ei siirtoa	1 alkio	2 alkiota	3 alkiota
$N_\mathrm{B} = 6$	ei siirtoa	1 alkio	2 alkiota	3 alkiota
$N_\mathrm{B} = 8$	ei siirtoa	1 alkio	3 alkiota	4 alkiota

Dekryptattaessa siirretään rivejä samat määrät syklisesti oikealle.

4.2.3 Sarakkeiden sekoitus (`MixColumn`)

Tässä muunnoksessa tilamatriisin sarakkeet tulkitaan enintään astetta 3 oleviksi polynomirenkaan $\mathbb{F}_{2^8}[z]$ polynomeiksi. Kukin sarake (polynomi) kerrotaan kiinteällä polynomilla

$$c(z) = c_0 \oplus c_1 z \oplus c_2 z^2 \oplus c_3 z^3$$

modulo $\mathbb{1} \oplus z^4$, missä (bitteinä)

$$\begin{aligned} c_0 &= 01000000\,(= x)\,, \\ c_1 = c_2 &= 10000000\,(= \mathbb{1})\,, \\ c_3 &= 11000000\,(= 1 + x). \end{aligned}$$

Jakaminen $\mathbb{F}_{2^8}[z]$:ssa polynomilla $\mathbb{1} \oplus z^4$ on erityisen helppoa, sillä

$$z^k \equiv z^{(k,\,\mathrm{mod}\ 4)} \mod \mathbb{1} \oplus z^4.$$

Dekryptattaessa jaetaan polynomilla $c(z)$ modulo $\mathbb{1} \oplus z^4$. Vaikkakaan $\mathbb{1} \oplus z^4$ ei ole $\mathbb{F}_{2^8}[z]$:n jaoton polynomi (se on $= (\mathbb{1} \oplus z)^4$), $c(z)$:llä on inverssi modulo $\mathbb{1} \oplus z^4$, sillä

$$\mathbb{1} = \mathrm{syt}(c(z), \mathbb{1} \oplus z^4).$$

Inverssi saadaan Eukleideen algoritmilla (työläs laskettava) ja se on

$$d(z) = d_0 \oplus d_1 z \oplus d_2 z^2 \oplus d_3 z^3,$$

missä

$$\begin{aligned} d_0 &= 01110000\,(= x + x^2 + x^3)\,, \\ d_1 &= 10010000\,(= 1 + x^3)\,, \\ d_2 &= 10110000\,(= 1 + x^2 + x^3)\,, \\ d_3 &= 11010000\,(= 1 + x + x^3). \end{aligned}$$

4.2.4 Kierrosavaimen lisäys (`AddRoundKey`)

Kierrosavain on yhtä pitkä kuin tilakin. Tässä operaatiossa kierrosavain lisätään biteittäin tilaan modulo 2. Käänteinen operaatio on sama.

4.2.5 Avaimen laajentaminen

Kierrosavaimet saadaan salausavaimesta laajentamalla se ja sitten valitsemalla sitten laajennetusta avaimesta eri kierroksille sopiva osa. Laajennetun avaimen bittipituus on $l_{\mathrm{B}}(N_{\mathrm{R}} + 1)$. Tavuihin jaettuna se voidaan esittää $4 \times N_{\mathrm{B}}(N_{\mathrm{R}} + 1)$-matriisina, jolla on siis $N_{\mathrm{B}}(N_{\mathrm{R}} + 1)$ kpl 4-pituista saraketta

$$\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_{N_{\mathrm{B}}(N_{\mathrm{R}}+1)-1}.$$

Jos $N_{\mathrm{K}} \leq 6$, lasketaan laajennettu avain seuraavan pseudokoodin osoittamalla tavalla:

```
KeyExpansion(word Key[NK] word W[NB*(NR+1)])
{
    for(i = 0; i < NK; i++)
        W[i] = Key[i];
```

```
        for(i = NK; i < NB*(NR + 1); i++)
        {
            temp = W[i - 1];
            if (i % NK == 0)
                temp = SubByte(RotByte(temp)) ^ Rcon[i / NK];
            W[i] = W[i - NK] ^ temp;
        }
    }
```

Tässä operaatio `SubByte` soveltaa RIJNDAELin S-laatikkoa kuhunkin sarakkeen alkioon (tavuun). Operaatio `RotByte` puolestaan tekee sarakkeen alkioissa yhden alkion syklisen siirron ylöspäin. Edelleen `^` on biteittäinen yhteenlasku modulo 2 vastintavuihin sovellettuina. Vektori $\mathbf{R}_{\mathrm{con}}(j)$ määritellään seuraavasti:

$$\mathbf{R}_{\mathrm{con}}(j) = \begin{pmatrix} x^{j-1} \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

(tässä x^{j-1} tulkitaan kunnan $\mathbb{F}_{2^8}$ alkioksi). Huomaa, ettei $\mathbf{R}_{\mathrm{con}}(j)$ riipu N_{K}:stä.

Mikäli $N_{\mathrm{B}} = 8$, menettely on hieman erilainen:

```
    KeyExpansion(word Key[NK] word W[NB*(NR+1)])
    {
        for(i = 0; i < NK; i++)
            W[i] = Key[i];

        for(i = NK; i < NB*(NR + 1); i++)
        {
            temp = W[i - 1];
            if (i % NK == 0)
                temp = SubByte(RotByte(temp)) ^ Rcon[i / NK];
            elseif (i % NK == 4)
                temp = SubByte(temp);
            W[i] = W[i - NK] ^ temp;
        }
    }
```

Ainoa ero on siis se, että kun $i \equiv 4 \mod N_{\mathrm{K}}$, $\mathbf{W}(i-1)$:een sovelletaankin ensin operaatiota `SubByte`.

Nyt i:nnen kierroksen kierrosavain saadaan sarakkeista $\mathbf{W}_{iN_{\mathrm{B}}}, \ldots, \mathbf{W}_{(i+1)N_{\mathrm{B}}-1}$ ($i = 0, 1, \ldots, N_{\mathrm{R}}$). Erityisesti N_{B} ensimmäisestä sarakkeesta saadaan aluksi tarvittava 0. kierroksen kierrosavain.

Huomautus. *Avaimen laajennus voidaan tehdä "etukäteen", kunhan vain salausavain tiedetään.*

4.2.6 Dekryptauksen muunnelma

Suoraan menetellen dekryptaus menee seuraavaa operaatioketjua:

```
AddRoundKey(State,RoundKey(NR));
InvShiftRow(State);
InvByteSub(State);

AddRoundKey(State,RoundKey(NR-1));
InvMixColumn(State);
InvShiftRow(State);
InvByteSub(State);

---

AddRoundKey(State,RoundKey(1));
InvMixColumn(State);
InvShiftRow(State);
InvByteSub(State);

AddRoundKey(State,RoundKey(0));
```

Operaatioiden järjestys voidaan kuitenkin myös kääntää. Ensinnäkin rivinsiirron ja tavun muuntamisen järjestyksellä ei ole väliä, edellinen operoi riveihin ja jälkimmäinen tavuihin. Sama koskee käänteisiä operaatioita. Toiseksi, operoinnit

```
AddRoundKey(State,RoundKey);
InvMixColumn(State);
```

voidaan korvata operoinneilla

```
InvMixColumn(State);
AddRoundKey(State,InvRoundKey);
```

missä `InvRoundKey` on `InvMixColumn(RoundKey)`. Näin dekryptauksessa voidaan myös edetä ketjua

```
AddRoundKey(State,RoundKey(NR));

InvByteSub(State);
InvShiftRow(State);
InvMixColumn(State);
AddRoundKey(State,InvRoundKey(NR-1));

InvByteSub(State);
InvShiftRow(State);
InvMixColumn(State);
AddRoundKey(State,InvRoundKey(NR-2));

---

InvByteSub(State);
InvShiftRow(State);
AddRoundKey(State,RoundKey(0));
```

joka muistuttaa kovasti kryptausta. Aivan kuten DESillekin, RIJNDAELin kryptaus ja dekryptaus ovat näin hyvin samanlaisia operaatioita.

4.3 RIJNDAELin kryptanalyysiä

RIJNDAEL on jo suunnitteluvaiheessa rakennettu kestämään likipitäen kaikkia tunnettuja hyökkäyksiä tämäntapaisille kryptosysteemeille.[2] Suunnittelijat Joan Daemen ja Vincent Rijmen selostivat laajasti RIJNDAELin konstruktioperiaatteita julkisessa dokumentissa DAEMEN, J. & RIJMEN, V.: *AES Proposal: Rijndael* (1999), jonka he ovat myös laajentaneet kirjaksi DAEMEN, J. & RIJMEN, V.: *The Design of Rijndael.* Springer–Verlag (2001). Mainittakoon vain lyhyesti, että erityisesti lineaarinen ja differentiaalinen kryptanalyysi on RIJNDAELissa tehokkaasti estetty eri muodoissaan.

Toisaalta RIJNDAEL on oikeastaan ainoa ”parempi” kryptosysteemi lajiaan, jonka (ainoa) S-laatikko on kirjoitettavissa suhteellisen yksinkertaiseen algebralliseen muotoon $\mathbb{F}_{2^8}$:ssa:

$$S(b) = s_0 \oplus \bigoplus_{i=1}^{8} (s_i \odot b^{255-2^{i-1}})$$

sopiville $\mathbb{F}_{2^8}$:n alkioille $s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8$. Tästä jatkaen on suhteellisen helppo saada koko kryptaukselle suora algebrallinen kaava! Onkin herännyt kysymys voiko tällaisia kaavoja laskennallisesti kääntää tehokkaasti. Myönteisessä tapauksessa RIJNDAEL olisi ilmeisesti murrettavissa. Asiaa on toistaiseksi tutkittu vain vähän.[3]

[2] Tässä on käytetty mm. suomalaismatemaatikon Kaisa Nybergin esittämiä ideoita. Ks. NYBERG, K.: Differentially Uniform Mappings for Cryptography. *Proceedings of Eurocrypt ’93. Lecture Notes in Computer Science* **765**. Springer–Verlag (1994).

[3] Ks. esimerkiksi FERGUSON, N. & SCHROEPPEL, R. & WHITING, D.: A Simple Algebraic Representation of Rijndael. *Proceedings of the 8th Annual Workshop in Selected Areas of Cryptography. Lecture Notes in Computer Science* **2259** (2001), 103–111.

Luku 5

JULKISEN AVAIMEN KRYPTAUS

5.1 Ekskursio algoritmien vaativuusteoriaan

Laskennallinen vaativuus eli *kompleksisuus* liittyy tehtävien laskennalliseen ratkaisemiseen tarvittaviin resursseihin verrattuna tehtävän kokoon. Tehtävän kokoa mitataan syötteen *pituudella* N, resurssit taas ovat yleensä *aika* (laskenta-askelten lukumäärä) tai *tila* (laskennan käyttämä maksimaalinen muistitila sopivasti mitattuna). Usein tehtävät ovat ns. *tunnistustehtäviä*, joissa ratkaisu on kyllä-vastaus. Hyvä viite klassisen kompleksisuusteorian osalta on HOPCROFT & ULLMAN.

Jotta kompleksisuus saadaan yhteismitalliseksi, pitää sopia tietty algoritmin matemaattinen malli, esimerkiksi Turingin koneilla laskeminen, ks. kurssi Automaattiteoria, Formaaliset kielet tai Matemaattinen logiikka. Algoritmimalleista on käytettävissä *deterministinen* versio, missä algoritmilla ei ole valinnanmahdollisuuksia, ja *epädeterministinen* versio, missä algoritmin seuraava askel saattaa olla valittavissa useista mahdollisista askelista. Jotta epädeterministisen algoritmin voitaisiin sanoa ratkaisevan tehtävän, pitää tehdä seuraavat oletukset:

- Algoritmi pysähtyy, valitaanpa askelet miten tahansa.
- Algoritmi voi pysähtyä tilaan, jossa se ei ole ratkaissut tehtävää.
- Kun algoritmi pysähtyy tilaan, jossa se antaa ratkaisun, niin ratkaisun pitää olla oikea (ratkaisuja voi tällöin olla useitakin).
- Tunnistustehtävissä tapaus, missä algoritmi ei anna yhtään kyllä-vastausta, tulkitaan ei-vastaukseksi.
- Jos tehtävänä on laskea funktion arvo, niin epädeterministisen algoritmin on jokaisella syötteellä annettava ratkaisu (funktion arvo).

Epädeterministinen algoritmi onkin useimmiten parhaiten ajateltavissa ratkaisun todennusmenetelmäksi, ei sen tuottamismenetelmäksi.

Kompleksisuutta tarkastellaan pääosin asymptoottisena, ts. suurten tehtävien osalta, eikä vakiokertoimella eroavia aika-/tilakompleksisuuksia eroteta toisistaan, lineaarinen kiihdytys ja tilan pakkaus kun ovat helppoja saada aikaan missä tahansa algoritmimallissa. Vaikka algoritmimallin valinnalla on selvä merkitys kompleksisuuteen, sillä ei ole oleellista merkitystä, ts. se ei muuta kompleksisuusluokkia, joihin tehtävät vaativuutensa perusteella jaetaan. Usein kompleksisuus annetaan O-notaatiomuodossa $O(f(N))$, ks. pykälä 2.6. Menemättä sen tarkemmin algoritmimalleihin, määritellään muutama keskeinen kompleksisuusluokka.

Aikakompleksisuusluokka $\mathcal{P}$ (*deterministisesti polynomiaikaiset tehtävät*) muodostuu niistä tehtävistä, joissa N-pituisen probleeman (syötteen) ratkaisemiseen deterministisellä algoritmilla kuluu enintään $p(N)$ askelta, missä p on jokin (tehtävästä riippuva) N:n polynomi. Esimerkiksi kokonaislukujen peruslaskutoimitukset ja s.y.t.:n lasku ovat $\mathcal{P}$:ssä (ks. Luku 2).

Aikakompleksisuusluokka $\mathcal{NP}$ (*epädeterministisesti polynomiaikaiset tehtävät*) taas muodostuu niistä tehtävistä, joissa N-pituisen probleeman ratkaisemiseen epädeterministisellä algoritmilla kuluu enintään $p(N)$ askelta, missä p on jokin (jälleen tehtävästä riippuva) N:n polynomi. Esimerkiksi kokonaislukujen yhdisteisyys on $\mathcal{NP}$:ssä: Arvataan (epädeterministisyys!) vain kaksi tekijää ($\neq 1$) ja tarkistetaan kertolaskulla onko arvaus oikea.

Aikakompleksisuusluokka co–$\mathcal{NP}$ (*komplementäärisesti epädeterministisesti polynomiaikaiset tehtävät*) taas muodostuu niistä tunnistustehtävistä, joiden komplementti on $\mathcal{NP}$:ssä. (Probleeman *komplementti* saadaan, kun kyllä- ja ei-vastaukset vaihdetaan keskenään.) Esimerkiksi alkuluvun tunnistus on co–$\mathcal{NP}$:ssä, sillä sen komplementti on yhdisteisyystestaus, joka on $\mathcal{NP}$:ssä. Ei ole kovin vaikea näyttää, että alkuluvun tunnistus on $\mathcal{NP}$:ssä, mutta huomattavasti vaikeampi, että se on $\mathcal{P}$:ssä, ks. pykälä 6.2.

Ilmeisesti $\mathcal{P} \subseteq \mathcal{NP}$ ja (tunnistustehtävien osalta) $\mathcal{P} \subseteq$ co–$\mathcal{NP}$. Onko kumpikaan näistä sisältymisistä aito, on avoin probleema (ja hyvin kuuluisa sellainen!). Yleisesti uskotaan, että kumpikin on aito. Myöskään ei tiedetä, onko (tunnistustehtävien osalta) kumpikaan yhtälöistä $\mathcal{NP} =$ co–$\mathcal{NP}$ tai $\mathcal{P} = \mathcal{NP} \cap$ co–$\mathcal{NP}$ voimassa, yleisesti kuitenkin uskotaan, ettei ole.

Tilakompleksisuusluokka $\mathcal{PSPACE}$ (*deterministisesti polynomitilaiset tehtävät*) muodostuu niistä tehtävistä, joissa N-pituisen probleeman (syötteen) ratkaisemiseen deterministisellä algoritmilla tarvitaan enintään $p(N)$ muistiyksikköä, missä p on jokin (tehtävästä riippuva) N:n polynomi. Esimerkiksi kokonaislukujen peruslaskutoimitukset ja s.y.t.:n lasku ovat tietysti $\mathcal{PSPACE}$:ssä.

Tilakompleksisuusluokka $\mathcal{NPSPACE}$ (*epädeterministisesti polynomitilaiset tehtävät*) taas muodostuu niistä tehtävistä, joissa N-pituisen probleeman ratkaisemiseen epädeterministisellä algoritmilla kuluu enintään $p(N)$ muistiyksikköä, missä p on jokin (jälleen tehtävästä riippuva) N:n polynomi. Ei ole kovin vaikea päätellä, että

$$\mathcal{NP} \subseteq \mathcal{PSPACE} = \mathcal{NPSPACE},$$

mutta sitä ei tiedetä onko sisältyminen aito.

Algoritmissa saattaa olla mukana ideaalisen satunnaisluvun generointi, jolloin algoritmi on *probabilistinen* eli *stokastinen.* Stokastinen algoritmi saattaa satunnaisesti epäonnistua, ts. se ei tuota lainkaan tulosta ja luopuu tehtävän ratkaisemisesta. Tällaisia algoritmeja kutsutaan *Las Vegas -algoritmeiksi.* Stokastinen algoritmi saattaa toisaalta tulostaa väärän ratkaisun, tällaisia algoritmeja kutsutaan puolestaan *Monte Carlo -algoritmeiksi.* Monte Carlo -algoritmeja vastaava polynomiaikainen kompleksisuusluokka on $\mathcal{BPP}$ (*rajoitetun todennäköisyyden polynomiaikaiset tehtävät*). Algoritmin pitää tällöin tuottaa oikea tulos vähintään todennäköisyydellä p, missä $p > 1/2$ on ennalta valittu syötteestä riippumaton luku.

Algoritmin tehtävä saattaa olla yhden tehtävän probleeman konvertoiminen toisen tehtävän probleemaksi, tällöin puhutaan *reduktiosta.* Jos tehtävä A saadaan redusoiduksi toiseksi tehtäväksi B käyttäen (deterministisessä) polynomiajassa toimivaa reduktiota, saadaan B:n polynomiaikaisesta algoritmista A:lle polynomiaikainen algoritmi. Tehtävän sanotaan olevan *$\mathcal{NP}$-kova,* jos jokainen $\mathcal{NP}$:n tehtävä voidaan redusoida siihen polynomiaikaisella algoritmilla. *$\mathcal{NP}$-kova* tehtävä on *$\mathcal{NP}$-täydellinen,* jos se on itse $\mathcal{NP}$:ssä. $\mathcal{NP}$-täydellinen tehtävä on siinä mielessä ”pahin mahdollinen”, että jos se voitaisiin näyttää deterministisesti polynomiaikaiseksi, niin kaikki $\mathcal{NP}$:n tehtävät olisivat $\mathcal{P}$:ssä ja $\mathcal{NP} = \mathcal{P}$. $\mathcal{NP}$-täydellisiä tehtäviä tunnetaan nykyään toistatuhatta (laskentatavasta riippuen enemmänkin).

Lause 5.1. *Jos jokin $\mathcal{NP}$-täydellinen tehtävä on $\mathcal{NP} \cap \text{co–}\mathcal{NP}$:ssä, niin $\mathcal{NP} = \text{co–}\mathcal{NP}$ (tunnistustehtävien osalta).*

Todistus. Oletetaan, että jokin $\mathcal{NP}$-täydellinen (tunnistus)tehtävä C on $\mathcal{NP} \cap \text{co–}\mathcal{NP}$:ssä. Tarkastellaan mielivaltaista $\mathcal{NP}$:ssä olevaa (tunnistus)tehtävää A. Koska C on $\mathcal{NP}$-täydellinen, on A redusoitavissa polynomiajassa C:hen. Näin ollen A:n komplementti on redusoitavissa polynomiajassa C:n komplementtiin, joka on myös $\mathcal{NP}$:ssä. Siispä A on $\text{co–}\mathcal{NP}$:ssä. A oli mielivaltaisesti valittu, joten $\mathcal{NP} \subseteq \text{co–}\mathcal{NP}$ ja välittömänä seurauksena myös $\text{co–}\mathcal{NP} \subseteq \mathcal{NP}$, ja siis $\mathcal{NP} = \text{co–}\mathcal{NP}$. □

Koska yleisesti uskotaan, että $\mathcal{NP} \neq \text{co–}\mathcal{NP}$, ei mikään $\mathcal{NP}$-täydellinen tunnistustehtävä näin olisi $\mathcal{NP} \cap \text{co–}\mathcal{NP}$:ssä.

Vanha tehtävien jako laskenta-ajan puolesta käytännössä mahdollisiin eli *selviäviin* (tractable) ja liian aikaaviepiin eli *selviämättömiin* (intractable) on se, että $\mathcal{P}$:n tehtävät ovat selviäviä ja muut selviämättömiä. Koska yleisesti uskotaan, että $\mathcal{NP} \neq \mathcal{P}$, olisivat $\mathcal{NP}$-täydelliset tehtävät selviämättömiä. Kryptologiassa on näin ollen luonnollista vaatia, että kryptaus- ja dekryptausfunktiot ovat $\mathcal{P}$:ssä. (On kuitenkin muistettava, että kryptaus saattaa sisältää stokastisia elementtejä.)

5.2 Julkisen avaimen kryptosysteemi

Julkisen avaimen kryptosysteemissä eli epäsymmetrisessä kryptosysteemissä on (ainakin) kaksi avainta: julkinen avain ja salainen avain (tai useampia sellaisia). Jotta salaisuus pysyisi, pitää olla laskennallisesti hyvin vaativaa laskea salainen avain julkisesta avaimesta lähtien. Julkinen avain voidaan jättää ”paikkaan”, josta kuka tahansa halukas voi sen ottaa käyttöön ja lähettää kryptatun viestin (jollekin) salaisen avaimen haltijalle. Tämän sinänsä yksinkertaisen idean esittivät Whitfield Diffie ja Martin Hellman vuonna 1976.[1]

Äkkiseltään tuntuisi olevan hyvä idea järjestää avaimet siten, että kryptanalyysi CO-datan ja julkisen avaimen avulla olisi laskennallisesti hyvin vaativa, esimerkiksi $\mathcal{NP}$-täydellinen. Melko ilmeisesti kyseinen kryptanalyysitehtävä on nimittäin $\mathcal{NP}$:ssä: Arvataan vain selväteksti ja kryptataan se julkisella avaimella. Vaikka kryptauksessa olisi mukana stokastisiakin elementtejä (satunnaisuutta), tämä menettely toimii, sillä satunnaiset valinnat voidaan myös arvata.

Mainittu kryptanalyysitehtävä voidaan myös pukea tunnistustehtäväksi, ns. *kryptotunnistukseksi:* ”Onko kolmikossa (w, k, c), missä k on julkinen avain, w kryptotekstiä c vastaava selväteksti?” Kryptotunnistus on $\mathcal{P}$:ssä, mikäli kryptaus on determinististä, joten sen saaminen kompleksisemmaksi edellyttää stokastistista kryptausta. Kovin pitkälle tällä tavoinkaan ei luultavasti päästä, sillä

Lause 5.2. *Jos jollekin kryptosysteemille kryptotunnistus on $\mathcal{NP}$-täydellinen, niin $\mathcal{NP} = \text{co–}\mathcal{NP}$.*

Todistus. Kryptotunnistustehtävä on ilmeisesti $\mathcal{NP}$:ssä (stokastiset osat voidaan arvata). Se on toisaalta myös $\text{co–}\mathcal{NP}$:ssä. Jos nimittäin c on kryptoteksti, niin sitä vastaa tarkalleen yksi selväteksti, muutoinhan dekryptaus ei onnistu. Arvataan nyt jokin selväteksti w' ja kryptataan se julkisella avaimella k. Jos tuloksena on c, verrataan w:tä ja w':a ja hyväksytään kolmikko (w, k, c), mikäli $w \neq w'$. Jos w':n kryptaus ei anna c:tä tai $w = w'$, menettely pysähtyy antamatta tulosta. Kryptotunnistus on siis $\mathcal{NP} \cap \text{co–}\mathcal{NP}$:ssä ja tulos seuraa Lauseesta 5.1. □

[1] Alkuperäisviite on DIFFIE, W. & HELLMAN, M.: New Directions in Cryptography. *IEEE Transactions on Information Theory* **IT–22** (1976), 644–654.

Näin ollen kryptotunnistus ei luultavastikaan voi käytännössä olla $\mathcal{NP}$-täydellinen. Tulos osoittaa myös sen, etteivät stokastiset kryptosysteemit liene huomattavasti parempia kuin deterministiset.

Julkisen avaimen systeemien yhteydessä on tapana puhua ns. *yksisuuntaisista funktioista:* funktio $y = f(x)$ on yksisuuntainen, jos y:n laskeminen x:stä on selviävä, mutta x:n laskeminen y:stä on selviämätön (ehkä jopa $\mathcal{NP}$-täydellinen). Jos julkisen avaimen systeemin kryptausfunktio on e_k, niin funktio $(c, k) = (e_k(w), k) = f(w, k)$ on ideaalisesti yksisuuntainen. (Huomaa, että koska julkinen avain k on aina saatavilla, se on mukana funktion arvossa.) Kiinteälle julkiselle avaimelle k vastaava salainen avain antaa toisaalta ns. *salaluukun*, jolla w voidaan laskea c:stä nopeasti. Salaluukun olemassaolo tietysti merkitsee sitä, ettei kryptausfunktio todellisuudessa ole yksisuuntainen, jos k on kiinteä.

Huomautus. *Salaluukun liittäminen $\mathcal{NP}$-täydelliseen tehtävään on osoittautunut pulmalliseksi. Käytännössä salaluukun mukaan ottaminen rajaa muuten $\mathcal{NP}$-täydellisestä tehtäväpiiristä osan, joka ei ole $\mathcal{NP}$-täydellinen (eikä usein edes kovin vaativa). Itse asiassa yhdestäkään kryptosysteemeihin liittyvästä funktiosta, jonka pitäisi ideaalisesti olla yksisuuntainen, ei ole voitu todistaa, että se sitä olisi. (Taustalla kummittelee tietysti $\mathcal{P} = \mathcal{NP}$-probleema.) Otollisia probleemoja, joiden varaan voi rakentaa hyvän kryptosysteemin, ovatkin sellaiset, joiden kompleksisuus on avoin. Tällöin systeemin murtaminen merkitsisi samalla teoreettista läpimurtoa kompleksisuusteoriassa. Kaikki tämä (ja myös Lause 5.2) tietää sitä, ettei kompleksuusteorialla ole aivan sellaista merkittävää roolia kryptologiassa kuin usein mainitaan. (Kryptografia mainitaan usein kompleksisuusteorian käytännön sovelluksena 'par excellence'.)*

Julkisen avaimen kryptosysteemeillä voidaan toteuttaa protokollia, joihin ei salaisen avaimen systeemeillä pystytä. Otetaan esimerkkinä *autentikointi* ja *allekirjoitus* (ks. myös pykälä 3.5.2 ja Luku 10). Jos B haluaa varmentaa, että viesti tulee A:lta, pitää viestissä olla mukana tietoa, joka riittävän yksiselitteisesti spesifioi A:n sen lähettäjäksi. Seuraavat vaatimukset ovat tällöin luonnollisia:

(i) Sekä A:n että B:n tulee pystyä suojautumaan valeviestejä vastaan. Ulkopuolinen taho C ei saa voida tekeytyä A:ksi.

(ii) A:n tulee pystyä suojautumaan B:n tekemiä väärennöksiä vastaan (joiden B väittää allekirjoitettuina tulleen A:lta).

(iii) A ei saa voida kieltää lähettäneensä viestiä, jonka hän tosiasiallisesti lähetti.

Merkitään e_{A}:lla ja e_{B}:llä A:n ja B:n (julkisia) kryptausfunktioita ja d_{A}:lla sekä d_{B}:llä vastaavia (salaisia) dekryptausfunktioita. Tässä oletetaan, että kryptaus on deterministinen. Menettely on nyt seuraava:

1. A lähettää viestin w B:lle muodossa $c = e_{\mathrm{B}}(d_{\mathrm{A}}(w))$.
2. B laskee: $e_{\mathrm{A}}(d_{\mathrm{B}}(c)) = e_{\mathrm{A}}(d_{\mathrm{A}}(w)) = w$. ($e_{\mathrm{A}}$ ja d_{A} ovat käänteisfunktioita.)

Ehdot (i) ja (iii) toteutuvat, sillä vain A tietää d_{A}:n. Viestissä tulee silloin olla tunnistettava oikeantyyppinen sisältö (muutoinhan viestillä ei ole välttämättä mitään merkitystäkään). Myös ehto (ii) toteutuu, sillä B:n olisi mahdotonta itse generoida oikeannäköistä viestiä, koska hän ei tiedä d_{A}:ta. Jos vain allekirjoitus on tärkeä, mutta ei viestin salaaminen, riittää, että A lähettää B:lle parin $(w, d_{\mathrm{A}}(w))$. Tämä yksinkertaisin versio autentikoinnista/allekirjoituksesta on haavoittuva, ja parempiakin protokollia on.

5.3 Reppusysteemin nousu ja tuho

Esimerkkinä edellisen pykälän kompleksisuuspäättelyjen vaikutuksista on tunnettu julkisen avaimen systeemi KNAPSACK[2] eli *reppusysteemi.*

Reppusysteemin pohjalla on ns. *reppuprobleema.* Sen syötteenä on $(\mathbf{a}, m)$, missä $\mathbf{a} = (a_1, a_2, \ldots, a_n)$ on positiivisten kokonaislukujen vektori ja m on positiivinen kokonaisluku (jossain lukujärjestelmässä esitettyinä). Tehtävänä on kirjoittaa m joidenkin $\mathbf{a}$:n komponenttien summaksi tai ilmoittaa, ettei tällainen kirjoitelma ole mahdollinen. Ts. tehtävänä on valita 0–1-luvut $c_1, c_2, \ldots, c_n$ siten, että

$$\sum_{i=1}^{n} c_i a_i = m$$

tai sitten ilmoittaa, ettei se ole lainkaan mahdollista. Tunnistustehtävässä pitää vain ilmoittaa onko valinta mahdollinen. Reppuprobleema on selvästi $\mathcal{NP}$:ssä: Arvataan vain $c_1, c_2, \ldots, c_n$ ja testataan onko arvaus oikea. Itse asiassa sen tiedetään olevan $\mathcal{NP}$-täydellinen.

KNAPSACKin kryptaus tapahtuu seuraavasti. Viestisymbolit ovat bittejä ja viestilohkon pituus on n. Viestilohko $w = b_1 b_2 \cdots b_n$ (bittijono) kryptataan luvuksi

$$c = e_k(w) = \sum_{i=1}^{n} b_i a_i.$$

Julkinen avain k on $\mathbf{a}$. Ilmeisesti tällainen kryptaus on $\mathcal{P}$:ssä. Kryptanalyysi lähtien c:stä ja $\mathbf{a}$:sta on näin $\mathcal{NP}$-täydellinen.

Ilman mitään apuneuvoja KNAPSACKin dekryptaus olisi sekin $\mathcal{NP}$-täydellinen. Salaluukku otetaan käyttöön lähtemällä liikkeelle yksinkertaisista reppuprobleemoista, jotka voidaan ratkaista $\mathcal{P}$:ssä, ja muuntamalla nämä sitten tavallisen (mielivaltaisen) reppuprobleeman valepukuun. Tämän vm. reppuprobleeman $\mathbf{a}$ julkaistaan julkisena avaimena. Salaluukkutietoa käyttäen voidaan reppuprobleema $(\mathbf{a}, c)$ palauttaa alkuperäiseen helposti ratkaistavaan muotoonsa ja näin dekryptata salattu viesti. Mutta tämä ei johda vahvaan kryptosysteemiin, ts. salaluukkua käyttäen ei voida saada valepukuista reppusysteemiä, jonka kryptanalyysi olisi $\mathcal{NP}$-täydellinen. Itse asiassa KNAPSACKin eri variantit onkin todettu vaarallisen heikoiksi eikä niitä juurikaan enää käytetä. Sangen tunnettu on ns. *Shamirin hyökkäys* perusKNAPSACKiä vastaan, ks. esimerkiksi SALOMAA.

5.4 Julkisen avaimen kryptaukseen sopivia tehtäviä

Reppuprobleeman tavoin ne tehtävätyypit, joille on löydetty käyttöä julkisen avaimen kryptauksessa, ovat yleensä lukuteorian tai algebran tehtäviä, usein alunperin hyvinkin abstrakteja ja puhtaan matematiikan piiriin luettuja. Tämä on tuonut monet aikaisemmin täysin puhtaan matematiikan piiriin kuuluviksi katsotut probleemat ja tulokset käytännön kryptosysteemin pohjaksi. Erityisesti algebrallisen lukuteorian sekä siihen liittyvän algebrallisten käyrien teorian sangen esoteerisiksi katsotut tulokset ja tehtävät ovat näin tulleet konkreettiseen ja laajaan käyttöön.

[2]KNAPSACK on yhtenä ensimmäisistä julkisen avaimen kryptosysteemeistä ”historiallisesti” merkittävä, alkuperäisviite on MERKLE, R. & HELLMAN, M.: Hiding Information and Signatures in Trapdoor Knapsacks. *IEEE Transactions in Information Theory* **IT–24** (1978), 525–530.

Eräitä esimerkkejä:

Kryptosysteemi	**Tehtävätyyppi**
RSA, RABIN	Kahden suuren alkuluvun aukikerrotun tulon tekijöihinjako
ELGAMAL, DIFFIE–HELLMAN, XTR	Diskreetin logaritmin laskeminen syklisessä ryhmässä
MENEZES–VANSTONE, CRANDALL	Logaritmin laskeminen elliptisen käyrän määrittämässä syklisessä ryhmässä
ARITHMETICA	Konjugaattiprobleema ryhmässä
NTRU	Lukuhilan pienimmän vektorin etsiminen
MCELIECE, NIEDERREITER	Algebrallis-geometrisen lineaarisen koodin (Goppa-koodin) dekoodaus

Näistä neljän ensimmäisen tarkkaa kompleksisuutta ei tiedetä, tehtävät ovat kuitenkin $\mathcal{NP}$:ssä. Lukuhilan pienimmän vektorin etsiminen ja lineaarisen koodin dekoodaus (ks. kurssi Koodausteoria) taas ovat tunnetusti $\mathcal{NP}$-täydellisiä tehtäviä, joten NTRUn, MCELIECEn ja NIEDERREITERin osalta tilanteen pitäisi olla samantapainen kuin KNAPSACKin (jota ne muutenkin kaukaisesti muistuttavat). Näistä onkin löydetty heikkouksia.[3] MCELIECEn tarvitsemien avainten suuri koko on vakavasti rajoittanut sen käyttöä. Sen sijaan NTRU on jossain määrin käytössä. ARITHMETICAn haittapuolena on sopivan ryhmän löytäminen (tähänastiset valinnat ovat osoittautuneet huonoiksi).

Jatkossa käsitellään systeemejä RSA, RABIN, ELGAMAL, DIFFIE–HELLMAN ja MENEZES–VANSTONE. Hyvä yleisesitys on esimerkiksi kirjassa GARRETT.

[3]Ks. esimerkiksi CANTEAUT, A. & SENDRIER, N.: Cryptanalysis of the Original McEliece Cryptosystem. *Proceedings of ASIACRYPT '98. Lecture Notes in Computer Science* **1514**. Springer–Verlag (2000).

Luku 6

EKSKURSIO LUKUTEORIAAN 2

6.1 Jäännösluokkien multiplikatiivinen rakenne

Palataan aluksi pykälässä 2.4 jo mainittuun Eulerin funktioon $\phi(n)$, joka ilmoitti kokonaisluvulle $n \geq 2$ niiden kokonaislukujen m, $1 \leq m < n$, lukumäärän, joille $\mathrm{syt}(m, n) = 1$, ja samalla alkuluokkien lukumäärän modulo n. Tavallisesti määritellään myös $\phi(1) = 1$.

Lause 6.1. (i) *Jos p on alkuluku ja $k \geq 1$, niin*

$$\phi(p^k) = p^{k-1}(p-1).$$

Erikoisesti $\phi(p) = p - 1$.

(ii) *Jos* $\mathrm{syt}(n, m) = 1$, *niin*

$$\phi(nm) = \phi(n)\phi(m) \quad (\phi\text{:}n \text{ multiplikatiivisuus}).$$

Todistus. (i) Luvuista $1, 2, \ldots, p^k$, joka p:s on p:llä jaollinen. Näin ollen p:llä jaottomia lukuja on $p^k - p^k/p = p^{k-1}(p-1)$ kpl.

(ii) Kirjoitetaan luvut $1, 2, \ldots, nm$ taulukoksi

$$\begin{array}{ccccc} 1 & 2 & 3 & \cdots & n \\ n+1 & n+2 & n+3 & \cdots & 2n \\ 2n+1 & 2n+2 & 2n+3 & \cdots & 3n \\ \vdots & \vdots & \vdots & & \vdots \\ (m-1)n+1 & (m-1)n+2 & (m-1)n+3 & \cdots & mn \end{array}$$

Tapaukset $n = 1$ ja $m = 1$ ovat selviä, joten voidaan olettaa, että $n, m \geq 2$. Jokaisessa sarakkeessa olevat luvut ovat keskenään kongruentteja modulo n. Lauseen 2.11 Seurauksen nojalla kunkin sarakkeen luvut taas muodostavat jäännössysteemin modulo m. Sellaisia sarakkeita, joissa olevien lukujen s.y.t. n:n kanssa on $= 1$, on $\phi(n)$ kpl. Kussakin tällaisessa sarakkeessa on $\phi(m)$ sellaista lukua, joiden s.y.t. m:n kanssa on $= 1$. Nämä luvut ovat ne, joiden s.y.t. nm:n kanssa on $= 1$, ja niitä on $\phi(n)\phi(m)$ kpl. □

Käyttäen luvun x alkutekijöihinjakoa

$$x = p_1^{i_1} p_2^{i_2} \cdots p_N^{i_N}$$

(ks. Lauseet 2.2 ja 2.6) saadaan lausetta käyttäen

$$\phi(x) = \phi(p_1^{i_1})\phi(p_2^{i_2}) \cdots \phi(p_N^{i_N}) = p_1^{i_1-1} p_2^{i_2-1} \cdots p_N^{i_N-1}(p_1-1)(p_2-1)\cdots(p_N-1).$$

Koska tekijöihinjako kuitenkin on vaativa operaatio, ei $\phi(x)$ ole tätä kautta laskettavissa käytännössä, ellei jakoa ole valmiina saatavilla. Tästä joka tapauksessa nähdään suoraan, että jos x on yhdistetty luku, niin $\phi(x) < x - 1$.

Keskeinen tulos RSAn määrittelyssä on

Lause 6.2. (Eulerin lause) *Jos* $\mathrm{syt}(x, m) = 1$*, niin*

$$x^{\phi(m)} \equiv 1 \mod m.$$

Todistus. Valitaan positiivisesta jäännössysteemistä modulo m alkuluokkien edustajat $j_1, j_2, \ldots, j_{\phi(m)}$ (eli supistettu jäännössysteemi). Silloin luvut $xj_1, xj_2, \ldots, xj_{\phi(m)}$ muodostavat myös supistetun jäännössysteemin, sillä Lauseen 2.11 Seurauksen mukaisesti ne eivät ole keskenään kongruentteja ja niiden s.y.t. m:n kanssa on $= 1$. Niinpä voidaan luvut $xj_1, xj_2, \ldots, xj_{\phi(m)}$ ja $j_1, j_2, \ldots, j_{\phi(m)}$ asettaa jossakin järjestyksessä pareittain kongruenssiin:

$$xj_k \equiv j_{i_k} \mod m \quad (k = 1, 2, \ldots, \phi(m)).$$

Kertomalla nämä kongruenssit puolittain keskenään saadaan

$$x^{\phi(m)} j_1 j_2 \cdots j_{\phi(m)} \equiv j_1 j_2 \cdots j_{\phi(m)} \mod m$$

ja, koska $\mathrm{syt}(j_1 j_2 \cdots j_{\phi(m)}, m) = 1$, supistamalla edelleen $x^{\phi(m)} \equiv 1 \mod m$. □

Välittömänä seurauksena saadaan

Lause 6.3. (Fermat'n pieni lause) *Jos* p *on alkuluku ja* x *ei ole* p*:llä jaollinen, niin*

$$x^{p-1} \equiv 1 \mod p.$$

Eulerin lause ja erityisesti Fermat'n pieni lause on usein avuksi laskettaessa kongruenssipotensseja modulo m. Paitsi että käytetään Venäläisten talonpoikien algoritmia, redusoidaan ensin eksponentti modulo $\phi(m)$.

Pienintä sellaista lukua $i \geq 1$ (jos olemassa), että $x^i \equiv 1 \mod m$, kutsutaan x:n ***kertaluvuksi*** modulo m. Seuraavassa muutama kertaluvun perusominaisuus

- Kertaluku on olemassa täsmälleen silloin, kun $\mathrm{syt}(x, m) = 1$, onhan tällöin ainakin $x^{\phi(m)} \equiv 1 \mod m$ (Eulerin lause). Toisaalta, jos $\mathrm{syt}(x, m) \neq 1$, on ilmeisesti myös $\mathrm{syt}(x^i, m) \neq 1$ ja näin ollen $x^i \not\equiv 1 \mod m$, aina kun $i \geq 1$.

- Jos $x^j \equiv 1 \mod m$, niin kertaluku i jakaa j:n. Muutoin $j = qi + r$, $1 \leq r < i$, (jakolasku) ja
$$x^r = x^r 1^q \equiv x^r (x^i)^q = x^{qi+r} = x^j \equiv 1 \mod m$$
ja i ei olisikaan pienin mahdollinen. Erityisesti (Eulerin lause) kertaluku jakaa $\phi(m)$:n.

- Jos x:n kertaluku modulo m on i, niin x^j:n kertaluku l modulo m on
$$\frac{i}{\mathrm{syt}(i, j)} = \frac{\mathrm{pyj}(i, j)}{j}$$
(ks. Lause 2.9). Tämä pitää paikkansa, koska
  - $i \mid jl$ ja $j \mid jl$, joten $\mathrm{pyj}(i, j) \mid jl$ eli $\mathrm{pyj}(i, j)/j$ on l:n tekijä, ja
  - $(x^j)^{\mathrm{pyj}(i,j)/j} \equiv 1 \mod m$, joten l jakaa $\mathrm{pyj}(i, j)/j$:n.

- Jos x:n kertaluku modulo m on i ja y:n kertaluku modulo m on j ja $\mathrm{syt}(i,j) = 1$, niin xy:n kertaluku modulo m on ij. Ensinnäkin

$$(xy)^i = x^i y^i \equiv y^i \mod m,$$

joten $(xy)^i$:n kertaluku modulo m on sama kuin y^i:n eli j. Mutta, jos xy:n kertaluku modulo m on k, niin $(xy)^i$:n kertaluku modulo m on $k/\mathrm{syt}(i,k)$. Näin ollen $j \mid k$. Vastaavasti todetaan, että $i \mid k$. Koska $\mathrm{syt}(i,j) = 1$, on näin ollen oltava $ij \mid k$. Toisaalta

$$(xy)^{ij} = (x^i)^j (y^j)^i \equiv 1 \mod m,$$

josta seuraa, että $k \mid ij$.

Jos g:n kertaluku modulo m on suurin mahdollinen eli $\phi(m)$ ja $1 \leq g < m$, niin g on ns. m:n *primitiivinen juuri.* Koska g:n potenssit

$$1, g, g^2, \ldots, g^{\phi(m)-1}$$

eivät tällöin ole keskenään kongruentteja (miksi?) ja niitä on $\phi(m)$ kpl, ne muodostavat itse asiassa supistetun jäännössysteemin. Seuraava primitiivisten juurten ominaisuus otetaan käyttöön todistuksetta[1].

Lause 6.4. *Luvulla $m \geq 2$ on primitiivisiä juuria tarkalleen siinä tapauksessa, että se on joko 2 tai 4 tai muotoa p^k tai $2p^k$, missä p on pariton alkuluku. Alkuluvulla on siis aina primitiivisiä juuria.*

Jos g on m:n primitiivinen juuri, niin luvuista

$$(g^i, \mod m) \quad (i = 1, 2, \ldots, \phi(m) - 1)$$

ne, joissa $\mathrm{syt}(i, \phi(m)) = 1$, ovat myös m:n primitiivisiä juuria ja itse asiassa tarkalleen kaikki m:n primitiiviset juuret. Näin ollen, jos luvulla m yleensä on primitiivisiä juuria, niin niitä on $\phi(\phi(m))$ kpl.[2] Erityisesti alkuluvulla p on primitiivisiä juuria $\phi(p-1)$ kpl.

Yllä olevasta saadaan välittömänä seurauksena seuraava tunnettu alkulukujen karakterisointi.

Lause 6.5. (Lucas'n kriteeri alkuluvuille) *Luku $p \geq 2$ on alkuluku täsmälleen silloin, kun on kertalukua $p-1$ modulo p oleva luku.*

Todistus. Jos p on alkuluku, sillä on primitiivinen juuri ja se on kertalukua $p-1$.

Jos taas on luku x, joka on kertalukua $p-1$ modulo p, niin p:n on oltava alkuluku. Muutoin nimittäin $\phi(p) < p-1$ eikä x:n kertaluku voi näin olla $p-1$, koska se on $\phi(p)$:n tekijä. □

Mainittakoon, ettei tunneta kovin tehokkaita algoritmeja primitiivisten juurten löytämiseksi, edes alkuluvuille. Jos tunnetaan $\phi(m)$:n tekijät, niin seuraava tulos antaa testin m:n primitiiviselle juurelle. Toisaalta jo $\phi(m)$:n laskeminen on vaativa tehtävä suurille m:n arvoille, tekijöihinjaosta puhumattakaan.

Lause 6.6. (Lucas'n kriteeri primitiiviselle juurelle) *Luku $1 \leq g < m$ on m:n primitiivinen juuri täsmälleen silloin, kun $\mathrm{syt}(g,m) = 1$ ja jokaiselle $\phi(m)$:n alkutekijälle q on $g^{\phi(m)/q} \not\equiv 1 \mod m$.*

[1] Todistus ei ole kovin vaikea, mutta aika pitkä. Se löytyy kutakuinkin mistä tahansa lukuteorian kirjasta, ks. esimerkiksi SIERPINSKI. Myös kryptologian kirjoista saattaa löytyä tämä todistus, ks. esimerkiksi KRANAKIS tai GARRETT.

[2] Tämä on syy, miksi kummallisen näköinen lauseke $\phi(\phi(m))$ esiintyy niin usein esimerkiksi kryptografiassa.

Todistus. Jos g on m:n primitiivinen juuri, niin ilmeisesti $\mathrm{syt}(g, m) = 1$ ja $g^{\phi(m)/q} \not\equiv 1 \mod m$ jokaiselle $\phi(m)$:n alkutekijälle q, koska g:n kertaluku on $\phi(m)$.

Jos taas $\mathrm{syt}(g, m) = 1$ ja $g^{\phi(m)/q} \not\equiv 1 \mod m$ jokaiselle $\phi(m)$:n alkutekijälle q, niin g:n kertaluku i jakaa $\phi(m)$:n, ts. $i = \phi(m)/l$. Jos $l = 1$, niin asia on selvä. Muu ei tule kysymykseen, sillä jos l:llä olisi alkutekijä q eli $l = qt$, niin silloin olisi

$$g^{\phi(m)/q} = (g^i)^t \equiv 1^t = 1 \mod m.$$

□

Koska m:n primitiiviselle juurelle g luvut $1, g, g^2, \ldots, g^{\phi(m)-1}$ muodostavat supistetun jäännössysteemin, niin jokaiselle luvulle x, jolla ei ole yhteisiä tekijöitä m:n kanssa, on täsmälleen yksi sellainen eksponentti y, $0 \leq y < \phi(m)$, että $g^y \equiv x \mod m$. Tätä eksponenttia kutsutaan x:n *indeksiksi* eli *diskreetiksi logaritmiksi* modulo m kannassa g. Ei tunneta tehokkaita algoritmeja diskreetin logaritmin laskemiseksi. (Tähän perustuu mm. kryptosysteemi ELGAMAL, johon tullaan myöhemmin.) Epädeterministinen polynomiaikainen algoritmi lähtien syötteestä (m, g, x) tietysti on: Arvataan vain indeksi y ja testataan onko se oikea. Potenssiinkorotus Venäläisten talonpoikien algoritmilla redusoiden tulosta modulo m on polynomiaikainen.

6.2 Alkulukujen testaus ja generointi

Esitetään ensin epädeterministinen algoritmi, ns. *Prattin algoritmi*[3] alkulukutestaukselle. Algoritmi on polynomiaikainen, mikä näyttää, että alkulukutestaus on $\mathcal{NP}$:ssä. Algoritmi perustuu Lucas'n kriteereille. Syöte on luku $n \geq 2$, jonka pituus binääriesityksessä on N. Merkitään algoritmin testiaskeleiden (ks. alla) lukumäärää $T(n)$:llä ja

$$\mathrm{PRATT}(n) = \begin{cases} \text{YES, jos } n \text{ on alkuluku} \\ \text{FAIL, jos testi ei tuota tulosta.} \end{cases}$$

Prattin algoritmi:

1. Jos $n = 2$ tai $n = 3$, tulostetaan ”YES” ja lopetetaan (0 testiaskelta).
2. Jos $n > 3$ ja on parillinen (jako 2:lla), tulostetaan ”FAIL” ja lopetetaan (0 testiaskelta).
3. Arvataan kokonaisluku x väliltä $1 \leq x \leq n - 1$.
4. Tarkistetaan, onko $x^{n-1} \equiv 1 \mod n$ Venäläisten talonpoikien algoritmilla ja jakolaskulla (1 testiaskel). Ellei ole, niin tulostetaan ”FAIL” ja lopetetaan.
5. Arvataan $n - 1$:n alkutekijät $p_1, \ldots, p_k$ (lista, jossa kukin oletettu alkutekijä esiintyy kertalukunsa osoittaman määrän kertoja; 0 testiaskelta). Näiden lukujen pituudet binäärijärjestelmässä ovat $P_1, \ldots, P_k$. Huomaa, että $P_1 + \cdots + P_k \leq N + k - 1$ ja että $2 \leq k \leq N$.
6. Tarkistetaan, kutsuen Prattin algorimia rekursiivisesti, että luvut $p_1, \ldots, p_k$ ovat todella alkulukuja (enintään $T(p_1) + \cdots + T(p_k)$ testiaskelta). Jos jokin $\mathrm{PRATT}(p_i) = \text{FAIL}$, tulostetaan ”FAIL” ja lopetetaan.
7. Tarkistetaan kertomalla, että todella $p_1 \cdots p_k = n - 1$ (1 testiaskel). Ellei näin ole, tulostetaan ”FAIL” ja lopetetaan.

[3] Alkuperäisviite on PRATT, V.R.: Every Prime has a Succint Certificate. *SIAM Journal on Computing* **4** (1976), 198–221.

8. Tarkistetaan, että $x^{(n-1)/p_i} \not\equiv 1 \mod n$ ($i = 1, \ldots, k$) Venäläisten talonpoikien algoritmilla ja jakolaskuilla (enintään k testiaskelta). Jos tämä pitää paikkansa, tulostetaan ”YES”, muutoin ”FAIL”.

$T(n)$:lle saadaan nyt seuraava rekursioepäyhtälö:

$$T(n) \leq 2 + k + \sum_{i=1}^{k} T(p_i)\,, \quad T(2) = 0\,, \quad T(3) = 0.$$

Tämän avulla voidaan löytää nyt $T(n)$:lle yläraja. Helposti nimittäin nähdään rekursiivisesti, että esimerkiksi $L(n) = 4\log_2 n - 4$ on tällainen yläraja, sillä $L(2) = 0$ ja $L(3) \geq 0$ ja

$$\begin{aligned} T(n) &\leq 2 + k + \sum_{i=1}^{k} L(p_i) = 2 + k + \sum_{i=1}^{k} (4\log_2 p_i - 4) \\ &= 2 + k + 4\log_2(p_1 \cdots p_k) - 4k = 2 - 3k + 4\log_2(n-1) \\ &\leq -4 + 4\log_2 n = L(n). \end{aligned}$$

Toisaalta kunkin testiaskeleen suorittamiseen kuluu $O(N^3)$ askelta (parempiakin arvioita olisi) ja $L(n)$ on verrannollinen N:ään (Lause 2.4). Siispä kokonaisaika on $O(N^4)$.

Eräät sangen käyttökelpoiset alkulukutestit ovat probabilistisia, ts. ne tuottavat (oikean) tuloksen vain tietyllä suurella todennäköisyydellä. Tällainen testi on esimerkiksi ns. ***Miller–Rabin-testi***[4]. Testi perustuu Fermat’n pieneen lauseeseen. Sen mukaisesti, jos n on alkuluku ja x on sellainen kokonaisluku, että $\mathrm{syt}(x, n) = 1$, niin $x^{n-1} \equiv 1 \mod n$. Kirjoitetaan n muotoon

$$n = 1 + 2^l m,$$

missä m on pariton. Jos n on pariton, niin $l \geq 1$ ja

$$0 \equiv x^{n-1} - 1 = x^{2^l m} - 1 = (x^{2^{l-1} m} - 1)(x^{2^{l-1} m} + 1) \mod n$$

ja koska n on alkuluku se jakaa joko $x^{2^{l-1} m} - 1$:n tai $x^{2^{l-1} m} + 1$:n. Jos n jakaa $x^{2^{l-1} m} - 1$:n, niin sama operaatio voidaan toistaa. Jne. Tästä päätellään, että joko jollekin luvulle $i = 0, 1, \ldots, l-1$

$$x^{2^i m} \equiv -1 \mod n$$

tai sitten, ellei näin ole, lopulta

$$x^m \equiv 1 \mod n.$$

Jos nyt jollekin sellaiselle kokonaisluvulle x, että $\mathrm{syt}(x, n) = 1$ ja $x^m \not\equiv \pm 1 \mod n$, onkin kaikille luvuille $i = 1, 2, \ldots, l-1$

$$x^{2^i m} \equiv 1 \mod n,$$

niin voidaan vain päätellä, että n ei olekaan alkuluku. (Samoin, kun kohdataan sellainen $i > 0$, että $x^{2^i m} \not\equiv \pm 1 \mod n$.) Toisaalta, kun kokeillaan useita lukuja, esimerkiksi tiettyjä ”pieniä” alkulukuja $x = 2, 3, 5, 7, 11, \ldots$, saadaan eräänlainen ”todiste” sille, että n on alkuluku. Itse asiassa tämä todiste voidaan saada erittäin varmaksi käyttämällä kyllin monta hyvin valittua lukua x. Myös todennäköisyysmielessä, ajatellen luvun x satunnaista valintaa väliltä $1 < x < n - 1$.

Seuraavassa oletetaan, että käytössä ovat (annetut tai satunnaiset) testiluvut $x_1, x_2, \ldots, x_k$.

[4]Alkuperäisviitteet ovat MILLER, G.L.: Riemann’s Hypothesis and Tests for Primality. *Journal of Computer and System Sciences* **13** (1976), 300–317 sekä RABIN, M.O.: Probability Algorithms. *Algorithms and Complexity* (J.F. Traub, toim.). Academic Press (1976), 35–36.

Miller–Rabin-alkulukutesti:

1. Jos n on parillinen, on asia helppo ja lopetetaan.
2. Jos n on pariton, asetetaan $l \leftarrow 0$ ja $m \leftarrow n - 1$.
3. Asetetaan $l \leftarrow l + 1$ ja $m \leftarrow m/2$.
4. Jos m on parillinen, mennään kohtaan 3. (Näitä kierroksia tarvitaan enintään $\lfloor \log_2 n \rfloor$ kpl.)
5. Asetetaan $j \leftarrow 0$.
6. Jos $j < k$, asetetaan $j \leftarrow j + 1$ ja $x \leftarrow x_j$. Muutoin tulostetaan ”ALKULUKU” (arveltu tieto) ja lopetetaan.
7. Jos $x^m \equiv 1 \mod n$ tai $\mathrm{syt}(x, n) = n$, niin mennään kohtaan 6. Jos taas $1 < \mathrm{syt}(x, n) < n$, tulostetaan ”YHDISTETTY LUKU” (varma tieto) ja lopetetaan. (Potenssilasku Venäläisten talonpoikien algorimilla, s.y.t. Eukleideen algoritmilla.)
8. Asetetaan $i \leftarrow 0$.
9. Jos $x^{2^i m} \equiv 1 \mod n$, tulostetaan ”YHDISTETTY LUKU” (varma tieto) ja lopetetaan. (Potenssilasku lähtien kohdan 7. potenssista peräkkäisillä neliöönkorotuksilla, välitulokset pitää säilyttää!)
10. Jos $x^{2^i m} \equiv -1 \mod n$, mennään kohtaan 6.
11. Jos $i = l - 1$, tulostetaan ”YHDISTETTY LUKU” (varma tieto) ja lopetetaan. Muutoin asetetaan $i \leftarrow i + 1$ ja mennään kohtaan 9.

Huomautus. *Tämä on ”bottom-up”-versio testistä. Siitä on myös ”top-down”-versio, jossa i vähenee (ks. esimerkiksi moniste* RUOHONEN, K.: Symbolinen analyysi*). Näillä ei liene suurta eroa nopeudessa.*

Testi ei siis ole ”vuorenvarma”. Sellaisia yhdistettyjä lukuja, joita testi arvelee alkuluvuiksi, kutsutaan *vahvoiksi valealkuluvuiksi* testilukujen $x_1, x_2, \ldots, x_k$ suhteen. Esimerkiksi $25\,326\,001 = 2\,251 \cdot 11\,251$ on vahva valealkuluku testilukujen 3 ja 5 suhteen. Kiinteälle k:n arvolle testin aikakompleksisuus on $O(N^3)$, kuten on helppo todeta.

Probabilistisena testi on Monte Carlo -tyyppiä. Voidaan osoittaa, että yhdelle satunnaisesti väliltä $1 < x < n - 1$ valitulle x:lle testi tuottaa väärän tuloksen enintään todennäköisyydellä $1/4$ (ks. alkuperäisviite RABIN tai esimerkiksi CRANDALL & POMERANCE tai KRANAKIS tai GARRETT). Toistamalla saadaan varmuus miten tahansa hyväksi. Muitakin Monte Carlo -tyyppisiä alkulukutestejä on olemassa, esimerkiksi ns. *Solovay–Strassen-algoritmi*, ks. esimerkiksi SALOMAA tai STINSON tai KRANAKIS. Alkulukutestien ”vanhaa aatelia” ovat *Adleman–Pomerance–Rumely-testi*[5] ja sen seurannaiset. Testi pohjautuu varsin pitkälle menevään algebralliseen lukuteoriaan ja on nopea, luvun n alkulukutestaus vie sillä enintään

$$O((\ln n)^{c \ln(\ln(\ln n))})$$

[5]Alkuperäisviite on ADLEMAN, L. & POMERANCE, C. & RUMELY, R.: On Distinguishing Prime Numbers from Composite Numbers. *Annals of Mathematics* **117** (1983), 173–206.

askelta, missä c on (pieni) vakio, eikä näin ollen ole aivan $\mathcal{P}$:ssä (mutta melkein, sillä kyllä $\ln(\ln(\ln n))$ kasvaa todella hitaasti). Toisaalta, sekä teoreettisesti että implementointia ajatellen, se on vaikea käsiteltävä. Ks. esimerkiksi KRANAKIS.

Viime aikojen suuria lukuteorian tuloksia on se, että alkulukujen tunnistus on $\mathcal{P}$:ssä. Tämän todistivat intialaiset Manindra Agrawal, Neeraj Kayal ja Nitin Saxena v. 2002 käsikirjoituksessaan PRIMES *is in* P. Algoritmin todistettu kompleksisuus on $O((\ln n)^8)$, mutta heuristisesti sille saadaan kompleksisuus $O((\ln n)^6)$. Kovin nopeita implementointeja ei kuitenkaan toistaiseksi ole, vaikka algoritmi on varsin lyhyt esittää.

Paitsi alkulukutestaus, on myös tietynpituisten alkulukujen generointi keskeinen tehtävä. N-pituinen alkuluku voidaan valita satunnaisesti valitsemalla ensin satunnainen N-pituinen kokonaisluku (ks. pykälä 2.6) ja sitten testaamalla (Miller–Rabin-testillä), onko se alkuluku. Tällainen alkulukujen generointitapa on tehokas. Jos nimittäin merkitään $\pi(x)$:llä niiden alkulukujen lukumäärää, jotka ovat $\leq x$, niin saadaan kuuluisa asymptoottinen arvio

Lause 6.7. (Alkulukulause) $\lim\limits_{x\to\infty} \dfrac{\pi(x)}{x/\ln x} = 1$

(Todistus on hankala!) Näin ollen suuruusluokkaa n olevista luvuista suurinpiirtein joka $\ln n$:s on alkuluku. Tämä on riittävän paljon, jotta alkulukujen satunnainen etsiminen sujuisi nopeasti.

Alkulukuja ja niiden testausta käsittelee monipuolisesti CRANDALL & POMERANCE.

6.3 Lukujen tekijöihinjako

Siitä, että alkulukutestaus on $\mathcal{P}$:ssä, seuraa varsin välittömästi, että lukujen tekijöihinjako on $\mathcal{NP}$:ssä: arvataan vain alkutekijät ja testataan ovatko ne alkulukuja. Vaikka alkulukutestaus on $\mathcal{P}$:ssä ja käytännössäkin varsin nopeaa, tekijöihinjako näyttää olevan hyvin vaativa tehtävä. Riittää esittää menetelmä, joka etsii luvulle $n \geq 2$ jonkin ei-triviaalin tekijän d, $1 < d < n$, tai sitten ilmoittaa n:n olevan alkuluvun. Sen jälkeen voidaan rekursiivisesti jatkaa luvuista d ja n/d. Tietysti kannattaa aloittaa alkulukutestillä, jonka jälkeen voidaan olettaa, ettei n ole alkuluku.

Seuraava tunnettu algoritmi löytää usein parittoman yhdistetyn luvun n tekijän, jos jollekin n:n alkutekijälle p luvulla $p-1$ ei ole tekijänään b:tä ylittäviä alkuluvun potensseja. Tästä ehdosta seuraa, että $p-1$ on $b!$:n tekijä (eikö vain?).

Pollardin $p-1$-algoritmi[6]:

1. Asetetaan $a \leftarrow 2$.
2. Iteroidaan j:n arvoille $j = 2, \ldots, b$ asetusta $a \leftarrow (a^j, \mathrm{mod}\ n)$.
3. Lasketaan $d = \mathrm{syt}(a-1, n)$.
4. Jos $1 < d < n$, tulostetaan tekijä d, muutoin tulostetaan ”FAIL”, ja lopetetaan.

Oletetaan, että p on n:n alkutekijä, joka toteuttaa mainitun ehdon. Kohdan 2. suorituksen jälkeen on ilmeisesti $a \equiv 2^{b!} \mod n$ ja siis myös $a \equiv 2^{b!} \mod p$. Fermat'n pienen lauseen nojalla $2^{p-1} \equiv 1 \mod p$. Kuten todettiin, $p-1 \mid b!$, joten myös $a \equiv 1 \mod p$. Näin ollen $p \mid a-1$ ja edelleen $p \mid d$. On mahdollista, että $a = 1$, jolloin tekijää ei löydy.

[6] Alkuperäisviite on POLLARD, J.M.: Theorems on Factorization and Primality Testing. *Proceedings of the Cambridge Philosophical Society* **76** (1975), 521–528. Algoritmia voidaan vähän varioida monin tavoin sen tehostamiseksi, tämä on aivan perusversio.

Algoritmin aikakompleksisuus on

$$O(bBN^2 + N^3),$$

missä N ja B ovat lukujen n ja b pituudet binäärijärjestelmässä, vastaavasti. Tästä nähdään, että b tulisi pitää varsin pienenä n:ään nähden, jotta algoritmi olisi nopea. Liian pieni b toisaalta karsii pois paljon alkutekijöitä ja algoritmi ei tuota tulosta.

Pollardin $p-1$-algoritmin (ja monien muidenkin algoritmien) tarkempi esitys ja analyysi löytyy mm. viitteistä RIESEL ja CRANDALL & POMERANCE. Pollardin $p-1$-algoritmia on yleistetty (mm. ns. *Elliptisten käyrien menetelmä* ja ns. *Williamsin* $p+1$*-algoritmi*) monin tavoin.

Hyvin perinteinen tekijän etsimisalgoritmi on ns. *Testijakoalgoritmi.* Siinä kokeillaan ensin tekijöitä 2 ja 3 ja sen jälkeen järjestyksessä muotoa $6k \pm 1$ olevia tekijöitä aina $\sqrt{n}$:ään asti. (Kokonaisneliöjuuri on nopea laskea, ks. seuraava pykälä.) Tällainen menetelmä on tietysti aikaaviepä. Testijako on tyypiltään ns. *seulamenetelmä.* Seulamenetelmiä on hyvin paljon tehokkaampia, mm. ns. *Neliöllinen seula* ja *Lukukuntaseula.* Tällä hetkellä nopeimmille algoritmeille on arvioitu seuraavat aikakompleksisuudet[7]:

Algoritmi	**Aikakompleksisuus**
Neliöllinen seula	$O\left(e^{(1+o(1))\sqrt{\ln n \ln(\ln n)}}\right)$
Elliptisten käyrien menetelmä	$O\left(e^{(1+o(1))\sqrt{2 \ln p \ln(\ln p)}}\right)$ (p on n:n pienin alkutekijä)
Lukukuntaseula	$O\left(e^{(1+o(1))(\ln n)^{1/3}(\ln(\ln n))^{2/3}}\right)$

6.4 Kokonaisneliöjuuri

Ei-negatiivisen kokonaisluvun n *kokonaisneliöjuuri*[8] on $\lfloor\sqrt{n}\rfloor$. Oletetaan että n on annettu binäärijärjestelmässä ja merkitään N:llä sen pituutta. Kokonaisneliöjuuren laskemiseen käy vaikkapa peruskursseilta tuttu *Newtonin menetelmä.* Yhtälön $f(x) = 0$ Newtonin iteraatio on

$$\frac{f(x_i)}{x_i - x_{i+1}} = f'(x_i) \quad \text{eli} \quad x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}.$$

Neliöjuurelle yhtälö on $x^2 - n = 0$ ja iteraatio siis

$$x_{i+1} = \frac{1}{2}x_i + \frac{n}{2x_i}.$$

Seuraavat ominaisuudet on helppo todeta.

- Jos $x \neq \sqrt{n}$ ja $x > 0$, niin
$$\frac{1}{2}x + \frac{n}{2x} > \sqrt{n}.$$
Iteraatio siis lähestyy neliöjuurta ”yläpuolelta”.

[7] Merkintä $o(1)$ tarkoittaa seuraavaa: $f(n) = o(1)$, jos $\lim_{n\to\infty} f(n) = 0$. Yleisemmin merkintä $o(g(n))$ tarkoittaa, että $f(n) = o(g(n))$, jos $\lim_{n\to\infty} f(n)/g(n) = 0$.

[8] Joissain teksteissä se on $\lceil\sqrt{(n)}\rceil$, joissain taas $\sqrt{n}$ kokonaisluvuksi pyöristettynä. Menettely yleistyy korkeampiin kokonaisjuuriin.

- Jos $x > \sqrt{n}$, niin
$$\frac{1}{2}x + \frac{n}{2x} < x.$$
Iteraatio on siis aidosti vähenevä.

Lähtöarvoksi valitaan vaikkapa $x_0 = 2^{\lceil N/2 \rceil}$. Koska $n < 2^N$, on $x_0 > \sqrt{n}$.

Mutta koska halutaan laskea kokonaisluvuilla, otetaankin iteraatioksi

$$y_{i+1} = \left\lfloor \frac{y_i^2 + n}{2y_i} \right\rfloor , \quad y_0 = x_0$$

ja käytetään kokonaislukujen kerto- ja jakolaskua. Eo. ominaisuuksista voidaan nyt päätellä, että jos tätä iteraatiota toistetaan kunnes $y_i < \sqrt{n} + 1$, niin y_i tai $y_i - 1$ on etsitty n:n kokonaisneliöjuuri. Mutta kuinka monta kertaa iteraatiota pitää toistaa ja tuottavatko toistot lopultakaan tulosta? Tämän selvittämiseksi otetaan käyttöön ”virhe”

$$\epsilon_i = y_i - \sqrt{n}.$$

Arvioidaan ensin ϵ_0:

$$\frac{\epsilon_0}{2\sqrt{n}} = \frac{y_0}{2\sqrt{n}} - \frac{1}{2} = \frac{2^{\lceil N/2 \rceil}}{2\sqrt{n}} - \frac{1}{2} \leq \frac{2^{\lceil N/2 \rceil}}{2 \cdot 2^{(N-1)/2}} - \frac{1}{2} \leq 1 - \frac{1}{2} = \frac{1}{2},$$

ja sitten yleisesti:

$$\epsilon_i = y_i - \sqrt{n} \leq \frac{y_{i-1}^2 + n}{2y_{i-1}} - \sqrt{n} = \frac{1}{2y_{i-1}}\epsilon_{i-1}^2 \leq \frac{1}{2\sqrt{n}}\epsilon_{i-1}^2.$$

Vm. arviota iteroimalla saadaan (kun merkitään lyhyyden vuoksi $a = \dfrac{1}{2\sqrt{n}}$)

$$\begin{aligned}\epsilon_i &\leq a\epsilon_{i-1}^2 \leq a \cdot a^2\epsilon_{i-2}^{2^2} \leq \cdots \leq a^{1+2+2^2+\cdots+2^{i-1}}\epsilon_0^{2^i} \\ &= a^{2^i-1}\epsilon_0^{2^i} = a^{-1}(a\epsilon_0)^{2^i} = 2\sqrt{n}\left(\frac{\epsilon_0}{2\sqrt{n}}\right)^{2^i} < 2\sqrt{n}\,2^{-2^i}.\end{aligned}$$

Jos nyt halutaan, että $\epsilon_i < 1$, niin riittää ottaa

$$2\sqrt{n}\,2^{-2^i} < 1 \quad \text{eli} \quad -2^i + \log_2(2\sqrt{n}) < 0 \quad \text{eli} \quad i > \log_2\left(1 + \frac{1}{2}\log_2 n\right).$$

Näin ollen $\log_2 N$:ään verrannollinen määrä iteraatioaskelia riittää. Kokonaisneliöjuuren laskeminen on siis selvästikin $\mathcal{P}$:ssä, aikakompleksisuus on $O(N^2 \ln N)$.

Newtonin menetelmä kokonaisneliöjuuren laskemiseksi:

1. Jos $n = 0$, tulostetaan 0 ja lopetetaan.
2. Asetetaan $y \leftarrow 2^{\lceil N/2 \rceil}$.
3. Iteroidaan asetusta $y \leftarrow q$, missä q on osamäärä jaettaessa $y^2 + n$ luvulla $2y$, kunnes $(y-1)^2 \leq n$.
4. Testataan neliöönkorotuksella kumpi luvuista y vaiko $y-1$ on kysytty kokonaisneliöjuuri, tulostetaan se ja lopetetaan.

6.5 Kiinalainen jäännöslause

Jos modulin tekijöitä tunnetaan, ts. voidaan kirjoittaa

$$m = m_1 m_2 \cdots m_k,$$

seuraa kongruenssista $x \equiv y \mod m$ luonnollisesti kongruenssit $x \equiv y \mod m_i$ ($i = 1, 2, \ldots, k$). Jos moduli on suuri luku, voi usein olla helpompaa laskea käyttäen näitä pienempiä moduleja. Näin voidaan tehdä aivan yleisestikin, jos tekijät $m_1, m_2, \ldots, m_k$ ovat pareittain keskenään jaottomia:

Lause 6.8. (Kiinalainen jäännöslause) *Jos modulit $m_1, m_2, \ldots, m_k$ ovat pareittain keskenään jaottomia, on täsmälleen yksi kokonaisluku x modulo $m_1 m_2 \cdots m_k$, joka toteuttaa kaikki k kongruenssia*

$$x \equiv y_i \mod m_i \quad (i = 1, 2, \ldots, k).$$

Todistus. Merkitään $M = m_1 m_2 \cdots m_k$ ja $M_i = M/m_i$ ($i = 1, 2, \ldots, k$), jolloin $\mathrm{syt}(M_1, M_2, \ldots, M_k) = 1$ ja $\mathrm{syt}(m_i, M_i) = 1$ ($i = 1, 2, \ldots, k$) (miksi?). Seuraava menetelmä tuottaa ratkaisun x (jos sellainen on!) ja näyttää myös, että ratkaisu on yksikäsitteinen modulo M:

CRT-algoritmi:

1. Kirjoitetaan Eukleideen algoritmia käyttäen $\mathrm{syt}(M_1, M_2, \ldots, M_k) = 1$ Bezout'n muotoon (ks. Lause 2.8)
$$1 = c_1 M_1 + c_2 M_2 + \cdots + c_k M_k.$$
2. Tulostetaan $x \equiv c_1 M_1 y_1 + c_2 M_2 y_2 + \cdots + c_k M_k y_k \mod M$. (Vaikkapa positiivisessa jäännössysteemissä.)

Menettely toimii, jos ratkaisu on, sillä kongruensseista $x \equiv y_i \mod m_i$ seuraa suoraan, että $c_i M_i x \equiv c_i M_i y_i \mod M$ ($i = 1, 2, \ldots, k$) ja puolittain yhteenlaskien edelleen että

$$x = 1 \cdot x = (c_1 M_1 + c_2 M_2 + \cdots + c_k M_k) x \equiv c_1 M_1 y_1 + c_2 M_2 y_2 + \cdots + c_k M_k y_k \mod M.$$

Vielä pitää vielä osoittaa, että ratkaisu on olemassa. Koska ilmeisestikin $M_i \equiv 0 \mod m_j$, jos $i \neq j$, ja toisaalta $1 = c_1 M_1 + c_2 M_2 + \cdots + c_k M_k$, niin $c_i M_i \equiv 1 \mod m_i$ ($i = 1, 2, \ldots, k$). Siispä

$$x \equiv c_1 M_1 y_1 + c_2 M_2 y_2 + \cdots + c_k M_k y_k \equiv y_i \mod m_i \quad (i = 1, 2, \ldots, k),$$

kuten pitääkin. □

Todistus antaa erään algoritmin lauseessa mainitun luvun x etsimiseksi. Muitakin algoritmeja tunnetaan, mm. ns. *Garnerin algoritmi*, joka on jonkin verran nopeampi (ks. esimerkiksi CRANDALL & POMERANCE).

Huomautus. *Tietyssä mielessä Kiinalainen jäännöslause antaa muotoa*

$$y = f_x(m) = (x, \mod m)$$

olevien funktioiden sovituksen (interpolaation) "pisteiden" (m_i, y_i) kautta, seikka, jota voidaan käyttää eräissä protokollissa. Kiinalainen jäännöslause on erittäin käyttökelpoinen muutenkin monissa yhteyksissä, kuten nähdään. Hyvä viite on DING & PEI & SALOMAA.

6.6 Modulääarinen neliöjuuri

Luvun x sanotaan olevan luvun y *neliöjuuri modulo* m (eli ns. *modulääarinen neliöjuuri*), jos

$$x^2 \equiv y \mod m.$$

Yleensä ko. neliöjuuri esitetään positiviisessa jäännössysteemissä. Välittömästi nähdään, että jos x on y:n neliöjuuri modulo m, niin samoin on $(-x, \mod m)$. Neliöjuuria on siis yleensä ainakin kaksi, mutta usein paljon enemmänkin.

Kaikilla luvuilla y ei välttämättä ole neliöjuuria modulo m. Lukua y, jolla on neliöjuuri(a) modulo m, sanotaan *neliönjäännökseksi* modulo m, ja lukua y, jolla ei ole neliöjuurta modulo m taas *epäneliönjäännökseksi* modulo m. Ilmeisesti ainakin 0 ja 1 ovat neliönjäännöksiä. Yleisessä tapauksessa sen selvittäminen, onko jokin luku neliön- vaiko epäneliönjäännös modulo m, on työlästä. Jos kuitenkin m on alkuluku, nopeita menetelmiä tunnetaan ja ne ovat tarpeen kryptologiassa, asiaan palataan Luvussa 10.

Jos y on neliönjäännös modulo m ja tunnetaan m:n tekijöihinjako

$$m = p_1^{i_1} p_2^{i_2} \cdots p_M^{i_M}$$

sekä jokin y:n neliöjuuri x_j modulo $p_j^{i_j}$ ($j = 1, 2, \ldots, M$), niin y:n neliöjuuria modulo m saadaan Kiinalaisen jäännöslauseen avulla. (Huomaa, että jos y on neliönjäännös modulo m. niin se on myös neliönjäännös modulo $p_j^{i_j}$, sillä jokainen y:n neliöjuuri x modulo m on myös sen neliöjuuri modulo $p_j^{i_j}$.) Ratkaistaan kongruenssiryhmästä

$$\begin{cases} x \equiv \pm x_1 \mod p_1^{i_1} \\ x \equiv \pm x_2 \mod p_2^{i_2} \\ \quad \vdots \\ x \equiv \pm x_M \mod p_M^{i_M} \end{cases}$$

x modulo m CRT-algoritmilla (tulos on yksikäsitteinen modulo $m = p_1^{i_1} p_2^{i_2} \cdots p_M^{i_M}$). Mikä tahansa 2^M:stä $\pm$-merkkivalinnasta on mahdollinen. Silloin

$$x^2 \equiv (\pm x_j)^2 \equiv y \mod p_j^{i_j}$$

ja siis $p_j^{i_j} \mid x^2 - y$ ($j = 1, 2, \ldots, M$). Koska $p_j^{i_j}$:t ovat keskenään jaottomia, on myös $m \mid x^2 - y$ eli $x^2 \equiv y \mod m$. Käymällä läpi kaikki neliöjuurien x_j valinnat (niitä voi olla useampiakin) sekä kaikki $\pm$-merkkiyhdelmät saadaan itse asiassa kaikki y:n neliöjuuret modulo m.

Tilanne palautuu siis neliöjuurten laskemiseen modulo alkuluku tai sellaisen potenssi. Neliöjuurten laskeminen modulo alkuluvun korkeampi potenssi on vähän vaikeampaa, eikä sitä tässä käsitellä (siihen tarvitaan ns. Henselin lemma, ks. esimerkiksi GARRETT). Sen sijaan neliöjuuret modulo alkuluku p ovat nopeasti laskettavissa ns. *Shanksin algoritmilla*. y:n neliöjuuria modulo p on aina tarkalleen kaksi, paitsi jos $y \equiv 0 \mod p$, sillä jos x on jokin neliöjuuri ja x' toinen, niin

$$x^2 \equiv y \equiv x'^2 \mod p \quad \text{eli} \quad (x - x')(x + x') \equiv 0 \mod p$$

ja joko $p \mid x - x'$ eli $x \equiv x' \mod p$ tai sitten $p \mid x + x'$ eli $x' \equiv -x \mod p$. Jos taas $y \equiv 0 \mod p$, niin ainoa neliöjuuri on 0, kuten on helppo nähdä.

Jos $p > 2$, niin ilmeisesti kaikki neliönjäännökset modulo p saadaan, kun otetaan lukujen $0, 1, \ldots, (p-1)/2$ neliöt modulo p. Nämä neliöt eivät ole keskenään kongruentteja modulo p (miksi?), joten neliönjäännöksiä on yhtä enemmän kuin epäneliönjäännöksiä (ja se yksi on 0). Se onko y neliön- vaiko epäneliönjäännös modulo p, voidaan selvittää nopeasti (tapaukset $p = 2$ ja $y \equiv 0 \mod p$ ovat tietysti triviaaleja).

Lause 6.9. (Eulerin kriteeri) *Jos p on pariton alkuluku ja $y \not\equiv 0 \mod p$, niin y on neliönjäännös modulo p täsmälleen silloin, kun*

$$y^{\frac{p-1}{2}} \equiv 1 \mod p.$$

(Moduläärinen potenssi lasketaan käytännössä nopeasti Venäläisten talonpoikien algoritmilla.)

Todistus. Jos y on neliönjäännös eli jollekin x:lle $y \equiv x^2 \mod p$, niin Fermat'n pienen lauseen nojalla $x^{p-1} \equiv 1 \mod p$ (koska $y \not\equiv 0 \mod p$, niin $\mathrm{syt}(x,p) = 1$). Siispä

$$y^{\frac{p-1}{2}} \equiv x^{p-1} \equiv 1 \mod p.$$

Jos taas $y^{\frac{p-1}{2}} \equiv 1 \mod p$, niin otetaan jokin primitiivinen juuri g modulo p. Silloin jollekin i:lle $y \equiv g^i \mod p$ (koska $\mathrm{syt}(y,p) = 1$) ja

$$g^{\frac{p-1}{2}i} \equiv y^{\frac{p-1}{2}} \equiv 1 \mod p$$

Mutta koska g:n kertaluku on $p-1$, on $(p-1)i/2$:n oltava jaollinen $p-1$:llä. Näin ollen i on parillinen ja y:llä on neliöjuuret $(\pm g^{i/2}, \mathrm{mod}\ p)$ modulo p. □

Jos p on muotoa $p = 4l-1$ eli $p \equiv 3 \mod 4$, niin Eulerin kriteerillä saadaan itse asiassa myös ne kaksi neliöjuurta. Ne ovat $(\pm y^{(p+1)/4}, \mathrm{mod}\ p)$, sillä

$$(\pm y^{(p+1)/4})^2 = y^{(p+1)/2} = y^{(p-1)/2} y \equiv y \mod p.$$

Tapaus $p = 4l+1$ onkin sitten, outoa kyllä, paljon mutkikkaampi ja siihen nimenomaan tarvitaan Shanksin algoritmia.

Ennenkuin mennään Shanksin algoritmiin, voidaan nyt todeta, että jos m:ssä ei ole tekijöinä alkuluvun potensseja (eli m on *neliötön*) ja m:n tekijöihinjako

$$m = p_1 p_2 \cdots p_M$$

tunnetaan, niin tilanne neliönjäännöksiä ja neliöjuuria modulo m ajatellen on helppo:

- y on neliönjäännös modulo m täsmälleen silloin, kun se neliönjäännös modulo kukin p_i ($i = 1, 2, \ldots, M$), ja tämä voidaan selvittää nopeasti Eulerin kriteerillä.
- Laskemalla (Shanksin algoritmilla) y:n neliöjuuret x_j modulo p_j, saadaan näistä eo. tavalla CRT-algoritmilla kaikki enintään 2^M kpl y:n neliöjuurta modulo m.

Vielä saadaan

Lause 6.10. *Jos m on pariton ja neliötön,* $\mathrm{syt}(y,m) = 1$ *(eli y ei ole jaollinen millään alkuluvuista p_j) ja y on neliönjäännös modulo m, niin y:n neliöjuuria modulo m on tarkalleen 2^M kpl, missä M on m:n alkutekijöiden lukumäärä.*

Todistus. Muutoinhan jollekin p_j:lle olisi $x_j \equiv -x_j \mod p_j$ eli $2x_j \equiv 0 \mod p_j$. Koska p_j on pariton, olisi silloin myös $x_j \equiv 0 \mod p_j$ ja edelleen $y \equiv x_j^2 \equiv 0 \mod p_j$. □

Seuraus. *Jos m on pariton ja neliötön, y on neliönjäännös modulo m ja x on jokin y:n neliöjuuri modulo m, niin y:n neliöjuuret modulo m ovat tarkalleen $(x\omega_i, \mathrm{mod}\ m)$ ($i = 1, 2, \ldots, 2^M$), missä M on m:n alkutekijöiden lukumäärä ja $\omega_1, \omega_2 \ldots, \omega_{2^M}$ ovat 1:n neliöjuuret modulo m.*

Huomautus. *Tämä kaikki on hyvin paljon siitä riippuvaa, että m:n tekijöihinjako tunnetaan. Jo siinä tapauksessa, että $M = 2$ eikä tekijöitä tunneta, sen selvittäminen, onko y neliönjäännös modulo m vai ei, ja mitkä positiivisessa tapauksessa ovat sen neliöjuuret modulo m, on hyvin työlästä. Edes se ei auta, että tunnetaan yksi neliöjuuripareista. Itse asiassa, jos tunnetaan y:n sellaiset neliöjuuret x_1 ja x_2 modulo $m = p_1p_2$, että $x_1 \not\equiv \pm x_2 \mod m$, niin luvut $\mathrm{syt}(m, x_1 \pm x_2)$ ovat alkuluvut p_1 ja p_2. Näiden havaintojen varassa ovat monet kryptosysteemit ja protokollat, mm. RSA ja RABIN.*

Ja sitten se Shanksin algoritmi:

Shanksin algoritmi:

1. Jos $p = 2$, tulostetaan $(y,\mathrm{mod}\ 2)$ ja lopetetaan. Jos $y \equiv 0 \mod p$, tulostetaan 0 ja lopetetaan.

2. Jos $y^{\frac{p-1}{2}} \not\equiv 1 \mod p$, ei y:llä Eulerin kriteerin nojalla ole neliöjuuria modulo p. Tulostetaan tämä tieto ja lopetetaan.

3. Jos $p \equiv 3 \mod 4$, tulostetaan $(\pm y^{(p+1)/4},\mathrm{mod}\ p)$ ja lopetetaan.

4. Jos taas $p \equiv 1 \mod 4$, kirjoitetaan $p - 1 = 2^s t$, missä t on pariton ja $s \geq 2$. Tämä saadaan aikaan peräkkäisillä 2:lla jakamisilla, joita tarvitaan enintään $\lfloor \log_2(p-1) \rfloor$ kpl.

5. Valitaan satunnaisesti luku u väliltä $1 \leq u < p$. Jos nyt $u^{\frac{p-1}{2}} \equiv 1 \mod p$, tulostetaan "FAIL" ja lopetetaan. (Eulerin kriteerin mukaan u on tällöin neliönjäännös modulo p ja jatkoa ajatellen u:ksi haluttaisiinkin epäneliönjäännös.) u:n valinta onnistuu näin ollen 50% todennäköisyydellä.

6. Asetetaan $v \leftarrow (u^t,\mathrm{mod}\ p)$. Silloin v:n kertaluku modulo p on 2^s. Jos nimittäin i on mainittu kertaluku, niin $it \mid 2^s t$ eli $i \mid 2^s$. Toisaalta $u^{t2^k} \not\equiv 1 \mod p$, jos $k < s$, muutoinhan olisi myös $u^{\frac{p-1}{2}} \equiv 1 \mod p$.

7. Asetetaan $z \leftarrow (y^{(t+1)/2},\mathrm{mod}\ p)$. Silloin $z^2 \equiv y \cdot y^t \mod p$. z on tietyssä mielessä "likimääräinen" y:n neliöjuuri modulo p ja sitä käyttäen voidaan löytää oikea neliöjuuri muodossa $x = (zv^{-l},\mathrm{mod}\ p)$.

8. Etsitään mainittu oikea neliöjuuri, ts. sellainen luku l, että

$$x^2 \equiv (zv^{-l})^2 \equiv y \mod p \quad \text{eli} \quad v^{2l} \equiv z^2 y^{-1} \equiv y^t \mod p.$$

   Tällainen luku on olemassa, sillä modulääriyhtälöllä $w^{2^{s-1}} \equiv 1 \mod p$ on w:n suhteen 2^{s-1} juurta[9] ja ne ovat $(v^{2j},\mathrm{mod}\ p)$ $(j = 0, 1, \ldots, 2^{s-1} - 1)$ ja $(y^t,\mathrm{mod}\ p)$ on yksi niistä. Luku l voidaan etsiä vaikkapa rekursiivisesti binäärimuodossa

$$l = b_{s-2}2^{s-2} + b_{s-3}2^{s-3} + \cdots + b_1 2 + b_0$$

   seuraavasti:

   8.1 Bitti b_0 löytyy, kun kongruenssi $v^{2l} \equiv y^t \mod p$ korotetaan puolittain potenssiin 2^{s-2}, sillä

$$b_0 = \begin{cases} 0, \text{ jos } (y^{t2^{s-2}}, \mathrm{mod}\ p) = 1 \\ 1 \text{ muuten.} \end{cases}$$

[9] Tässä tarvitaan polynomialgebrasta se tieto, että kunnan d-asteisella algebrallisella yhtälöllä on enintään d juurta. Ks. esimerkiksi kurssi Algebra 1 tai Symbolinen analyysi 1 tai jokin algebran kirja.

8.2 Bitti b_1 löytyy, kun kongruenssi $v^{2l} \equiv y^t \mod p$ korotetaan puolittain potenssiin 2^{s-3}, sillä

$$b_1 = \begin{cases} 0, \text{ jos } (y^{t2^{s-3}} v^{-b_0 2^{s-2}}, \mod p) = 1 \\ 1 \text{ muuten.} \end{cases}$$

Huomaa, miten tässä tarvitaan jo saatu b_0.

8.3 Saatujen bittien b_0 ja b_1 avulla etsitään vastaavasti seuraava bitti b_2, jne.

9. Tulostetaan $(\pm z v^{-l}, \mod p)$ ja lopetetaan.

Varsin helposti voi todeta algoritmin polynomiaikaiseksi ja se tuottaa tuloksen noin 50% todennäköisyydellä. Kyseessä on siis Las Vegas -tyyppinen stokastinen algoritmi.

6.7 Jacobin symboli

Kuten edellisessä pykälässä todettiin, kun annetaan erisuuret parittomat alkuluvut $p_1, p_2 \ldots, p_M$, jakaantuvat luvut x, joilla ei ole tekijänään mitään ko. alkuluvuista, eri luokkiin sen mukaan onko x neliön vai epäneliönjäännös modulo p_i ($i = 1, 2, \ldots, M$). Näitä luokkia on 2^M kpl ja, jos alkuluvut on annettu, on Eulerin kriteerin avulla helppo selvittää mihin luokkaan x kuuluu.

Jos annetaankin vain aukikerrottu tulo

$$n = p_1 p_2 \cdots p_M,$$

tehtävä muuttuu tavattoman paljon vaikeammaksi. Mainittuun tuloon voidaan siis eräällä tavalla kätkeä tieto siitä, mihin luokkaan x kuuluu, ja tieto paljastuu oleellisesti vasta, jos/kun alkuluvut annetaan. Toisaalta kaikki tieto, joka liittyy mainittuun luokkaan, ei ole kovin hyvin piilossa. Erityisesti lukujen x ja n ns. Jacobin symboli voidaan laskea nopeasti ja se sisältää tietyn määrän tietoa siitä, mihin luokkaan x kuuluu.

Jacobin symboli on klassinen lukuteorian apuneuvo, johon pääsemiseksi määritellään ensin ns. *Legendren symboli* alkuluvulle p:

$$\left(\frac{x}{p}\right) = \begin{cases} 0, \text{ jos } p \text{ jakaa } x\text{:n} \\ 1, \text{ jos } p \text{ ei jaa } x\text{:ää ja } x \text{ on neliönjäännös modulo } p \\ -1, \text{ jos } x \text{ on epäneliönjäännös modulo } p. \end{cases}$$

Jos $x \equiv y \mod p$, niin ilmeisesti $\left(\frac{x}{p}\right) = \left(\frac{y}{p}\right)$. Tapaus $p = 2$ on yksinkertainen, $\left(\frac{x}{2}\right) = 0$, jos x on parillinen, muuten $= 1$.

Lause 6.11. *Parittomalle alkuluvulle p on*

$$\left(\frac{x}{p}\right) \equiv x^{\frac{p-1}{2}} \mod p,$$

ts. Legendren symboli $\left(\frac{x}{p}\right)$ on $x^{\frac{p-1}{2}}$:n edustaja symmetrisessä jäännössysteemissä.

Todistus. Jos p jakaa x:n, niin asia on selvä. Jos taas p ei jaa x:ää, niin Fermat'n pienen lauseen nojalla

$$x^{p-1} - 1 = (x^{\frac{p-1}{2}} - 1)(x^{\frac{p-1}{2}} + 1) \equiv 0 \mod p$$

eli joka tapauksessa $x^{\frac{p-1}{2}} \equiv \pm 1 \mod p$. Eulerin kriteerin nojalla $+$-merkki tulee kyseeseen tarkalleen silloin, kun x on neliönjäännös modulo p. □

Legendren symboli on näin ollen helppo laskea moduläärisellä potenssiin korotuksella.

Jacobin symboli on Legendren symbolin yleistys mielivaltaiselle positiiviselle kokonaisluvulle n. Jos n:n tekijöihinjako on

$$n = p_1^{i_1} p_2^{i_2} \cdots p_M^{i_M},$$

niin

$$\left(\frac{x}{n}\right) = \left(\frac{x}{p_1}\right)^{i_1} \left(\frac{x}{p_2}\right)^{i_2} \cdots \left(\frac{x}{p_M}\right)^{i_M}.$$

Eräät perusominaisuudet seuraavat melko välittömästi tästä määritelmästä:

- $\left(\frac{x}{n}\right) = \left(\frac{y}{n}\right)$, jos $x \equiv y \mod n$.
- $\left(\frac{x}{1}\right) = 1$ (tyhjä tulo).
- $\left(\frac{x}{n}\right) = 0$ tarkalleen silloin, kun $\mathrm{syt}(x, n) \neq 1$.
- $\left(\frac{xy}{n}\right) = \left(\frac{x}{n}\right)\left(\frac{y}{n}\right)$ (*1. kertolaskukaava,* seuraa Lauseesta 6.11).
- Jos $\mathrm{syt}(x, n) = 1$, niin $\left(\frac{x^2}{n}\right) = 1$ ja $\left(\frac{x^2 y}{n}\right) = \left(\frac{y}{n}\right)$.
- $\left(\frac{x}{nm}\right) = \left(\frac{x}{n}\right)\left(\frac{x}{m}\right)$ (*2. kertolaskukaava*).
- Jos $\mathrm{syt}(x, n) = 1$, niin $\left(\frac{x}{n^2}\right) = 1$ ja $\left(\frac{x}{n^2 m}\right) = \left(\frac{x}{m}\right)$.
- Jos x on pariton, niin $\left(\frac{x}{2n}\right) = \left(\frac{x}{n}\right)$.
- $\left(\frac{1}{n}\right) = 1$, sillä 1 on neliönjäännös modulo mikä tahansa alkuluku.

Muitakin arvoja saadaan melko suoraan:

Lause 6.12. *Jos n on pariton, niin* $\left(\frac{-1}{n}\right) = (-1)^{\frac{n-1}{2}}$ *ja* $\left(\frac{2}{n}\right) = (-1)^{\frac{n^2-1}{8}}$.

Todistus. Jos n on pariton, on $n - 1$ parillinen ja edelleen $n^2 - 1 = (n + 1)(n - 1)$ on jaollinen 8:lla (molemmat tulontekijät ovat parillisia ja toinen jaollinen neljällä). Käytetyt eksponentit ovat siis kokonaislukuja. Tapaus $n = 1$ on triviaali, joten siirrytään tapaukseen $n > 1$.

Ensinnä voidaan todeta, että Lauseen 6.11 nojalla parittomalle alkuluvulle p

$$\left(\frac{-1}{p}\right) = (-1)^{\frac{p-1}{2}}.$$

Toiseksi, mille tahansa parittomalle luvulle n

$$n \equiv (-1)^{\frac{n-1}{2}} \mod 4$$

(n on silloin muotoa $4l \pm 1$). Siispä

$$\begin{aligned}\left(\frac{-1}{n}\right) &= (-1)^{i_1 \frac{p_1-1}{2}}(-1)^{i_2 \frac{p_2-1}{2}} \cdots (-1)^{i_M \frac{p_M-1}{2}} \\ &\equiv p_1^{i_1} p_2^{i_2} \cdots p_M^{i_M} = n \equiv (-1)^{\frac{n-1}{2}} \mod 4.\end{aligned}$$

Koska $1 \not\equiv -1 \mod 4$, ensimmäinen kaava seuraa tästä.

Toisen kaavan todistus on vähän pidempi. Ensinnä todetaan, että riittää todistaa kaava Legendren symboleille. Tämä seuraa siitä, että parittomille luvuille $m = 2l + 1$ ja $k = 2j + 1$ on

$$\frac{m^2k^2 - 1}{8} = 2(l^2 + l)(j^2 + j) + \frac{m^2 - 1}{8} + \frac{k^2 - 1}{8}$$

eli

$$\frac{m^2k^2 - 1}{8} \equiv \frac{m^2 - 1}{8} + \frac{k^2 - 1}{8} \mod 2$$

ja tämän seurauksena

$$(-1)^{\frac{n^2-1}{8}} = (-1)^{\frac{p_1^{2i_1} p_2^{2i_2} \cdots p_M^{2i_M} - 1}{8}} = (-1)^{i_1 \frac{p_1^2-1}{8}}(-1)^{i_2 \frac{p_2^2-1}{8}} \cdots (-1)^{i_M \frac{p_M^2-1}{8}}.$$

(Toistetaan kaavaa ja huomataan, että jos $a \equiv b \mod 2$, niin $(-1)^a = (-1)^b$.) Näin ollen voidaan olettaa, että $n = p > 2$ on alkuluku. Lasketaan ensin yhdellä tavalla tulo

$$\begin{aligned}(-1)^1 \cdot 1 \cdot (-1)^2 \cdot 2 \cdot (-1)^3 \cdot 3 \cdots (-1)^{\frac{p-1}{2}} \cdot \frac{p-1}{2} \\ &= ((p-1)/2)!(-1)^{1+2+3+\cdots+(p-1)/2} \\ &= ((p-1)/2)!(-1)^{\frac{p^2-1}{8}}.\end{aligned}$$

(Huomaa aritmeettisen sarjan summa

$$1 + 2 + 3 \cdots + \frac{p-1}{2} = \frac{1}{2}\frac{p-1}{2}\left(\frac{p-1}{2} + 1\right) = \frac{p^2-1}{8}.)$$

Koska ilmeisesti

$$(-1)^i i \equiv \begin{cases} i, \text{ jos } i \text{ on parillinen} \\ -i \equiv p - i, \text{ jos } i \text{ on pariton} \end{cases} \mod p,$$

voidaan laskea sama tulo toisella tavalla modulo p:

$$\begin{aligned}(-1)^1 \cdot 1 \cdot (-1)^2 \cdot 2 \cdot (-1)^3 \cdot 3 \cdots (-1)^{\frac{p-1}{2}} \cdot \frac{p-1}{2} \\ &\equiv 2 \cdot 4 \cdot 6 \cdots (p-1) = 2^{\frac{p-1}{2}}((p-1)/2)! \\ &\equiv \left(\frac{2}{p}\right)((p-1)/2)! \mod p\end{aligned}$$

(ks. Lause 6.11). Edelleen p ei ole $((p-1)/2)!$:n tekijä, joten supistamalla saadaan

$$(-1)^{\frac{p^2-1}{8}} \equiv \left(\frac{2}{p}\right) \mod p,$$

josta toinen kaava seuraa, sillä $1 \not\equiv -1 \mod p$. □

Paitsi eo. perustulokset, Jacobin symbolin laskun kannalta keskeinen on seuraava hyvin klassinen lukuteorian tulos:

Lause 6.13. (Neliönjäännösten resiprookkilaki) *Jos m ja n ovat parittomia positiivisia kokonaislukuja ja* $\mathrm{syt}(m,n)=1$*, niin*

$$\left(\frac{m}{n}\right)=(-1)^{\frac{(m-1)(n-1)}{4}}\left(\frac{n}{m}\right).$$

Ts., $\left(\frac{m}{n}\right)=\left(\frac{n}{m}\right)$*, paitsi jos* $m\equiv n\equiv 3 \mod 4$*, jolloin* $\left(\frac{m}{n}\right)=-\left(\frac{n}{m}\right)$.

Todistus. Todistus on varsin pitkä. Se löytyy kutakuinkin kaikista lukuteorian kirjoista ja joistakin kryptologian kirjoistakin, ks. esimerkiksi SIERPINSKI tai KRANAKIS tai GARRETT. □

Lause 6.12 ja Neliönjäännösten resiprookkilaki antavat Eukleideen algoritmin tapaisen rekursiivisen menetelmän Jacobin symbolin $\left(\frac{m}{n}\right)$ laskemiseksi:

Eukleideen algoritmi Jacobin symbolille:

1. Jos $\mathrm{syt}(m,n)\neq 1$, tulostetaan 0 ja lopetetaan. (S.y.t. lasketaan tietysti ”tavallisella” Eukleideen algoritmilla.)
2. Jos $m=1$ tai $n=1$, tulostetaan 1 ja lopetetaan.
3. Jos n on parillinen, lasketaan $\left(\frac{m}{n/2}\right)$, tulostetaan se ja lopetetaan.
4. Jos $m<0$, lasketaan ensin $\left(\frac{-m}{n}\right)$, tulostetaan $(-1)^{\frac{n-1}{2}}\left(\frac{-m}{n}\right)$ ja lopetetaan.
5. Jos m on parillinen, lasketaan ensin $\left(\frac{m/2}{n}\right)$, tulostetaan $(-1)^{\frac{n^2-1}{8}}\left(\frac{m/2}{n}\right)$ ja lopetetaan.
6. Edellisten kohtien jälkeen voidaan olettaa, että m ja n ovat >2 ja parittomia ja että $\mathrm{syt}(m,n)=1$. Jos $m>n$, suoritetaan jako $m=qn+r$, missä siis $0\leq r<n$, lasketaan ja tulostetaan $\left(\frac{r}{n}\right)$ ja lopetetaan. (Tulos on oikea, koska $r\equiv m \mod n$.)
7. Jos taas $m<n$, lasketaan ensin $\left(\frac{n}{m}\right)$, tulostetaan $(-1)^{\frac{(m-1)(n-1)}{4}}\left(\frac{n}{m}\right)$ ja lopetetaan.

Tälle algoritmille pätee hyvin Lamén lauseen tapainen arvio (ks. pykälä 2.3), mikä merkitsee, että Jacobin symbolit ovat nopeasti laskettavissa.

Laskien Legendren symboli on nopeasti selvitettävissä onko luku neliönjäännös modulo alkuluku p vai ei. Yhdistetyille luvuille tällaista nopeaa menetelmää ei tunneta ja Jacobin symboli ei yleisesti sano paljoa mitään neliönjäännöksistä. Jotain se kuitenkin sanoo. Nimittäin sen että jos

$$n=p_1p_2\cdots p_M$$

ja $\mathrm{syt}(x,n)=1$, niin

$$\left(\frac{x}{n}\right)=1$$

tarkalleen silloin, kun x on epäneliönjäännös modulo p_i parilliselle määrälle alkutekijöitä p_i. x voi tällöin olla neliönjäännös modulo n, ts.

$$\left(\frac{x}{p_1}\right) = \left(\frac{x}{p_2}\right) = \cdots = \left(\frac{x}{p_M}\right) = 1,$$

mutta ellei se sitä ole ja kuitenkin $\left(\frac{x}{n}\right) = 1$, niin x:ää kutsutaan *valeneliönjäännökseksi* modulo n.

Siirrytään sitten kryptologiassa tavalliseen tapaukseen, missä n on kahden erisuuren alkuluvun p ja q tulo ja $\mathrm{syt}(x, n) = 1$. Silloin x on

- neliönjäännös modulo n tarkalleen silloin, kun $\left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = 1$.
- valeneliönjäännös modulo n tarkalleen silloin, kun $\left(\frac{x}{p}\right) = \left(\frac{x}{q}\right) = -1$.
- epäneliönjäännös tarkalleen silloin, kun $\left(\frac{x}{p}\right) = -\left(\frac{x}{q}\right)$.

Lause 6.14. *Jos $n = pq$, missä p ja q ovat parittomia erisuuria alkulukuja, ja x on luku, jolle $1 \leq x < n$ ja $\mathrm{syt}(x, n) = 1$, niin*

(i) *puolella luvuista x on $\left(\frac{x}{n}\right) = 1$.*

(ii) *puolet niistä luvuista x, joille $\left(\frac{x}{n}\right) = 1$, on neliönjäännöksiä modulo n.*

(iii) *niistä luvuista x, joille $\left(\frac{x}{n}\right) = -1$, puolella on $\left(\frac{x}{p}\right) = -\left(\frac{x}{q}\right) = 1$.*

Todistus. Huomaa, että lukuja x on $\phi(n) = (p-1)(q-1)$ kpl eli neljällä jaollinen määrä. Valitaan luvut a, b ja c siten, että

$$\left(\frac{aq}{p}\right) = -1 \quad , \quad \left(\frac{bp}{q}\right) = 1 \quad , \quad \left(\frac{cp}{q}\right) = -1$$

ja merkitään

$$e = aq + bp \quad \text{ja} \quad d = aq + cp.$$

Silloin

$$\left(\frac{e}{p}\right) = \left(\frac{aq}{p}\right) = -1 \quad , \quad \left(\frac{e}{q}\right) = \left(\frac{bp}{q}\right) = 1 \quad \text{ja siis} \quad \left(\frac{e}{n}\right) = -1$$

sekä

$$\left(\frac{d}{p}\right) = \left(\frac{aq}{p}\right) = -1 \quad , \quad \left(\frac{d}{q}\right) = \left(\frac{cp}{q}\right) = -1 \quad \text{ja siis} \quad \left(\frac{d}{n}\right) = 1.$$

Nyt

$$\left(\frac{ex}{n}\right) = -\left(\frac{x}{n}\right),$$

ts. luvulla e kertoen modulo n Jacobin symboli vaihtuu. Näin ollen kohta (i) on tosi, sillä luvut x ovat luvut $(ex, \mathrm{mod}\ n)$, järjestys vain vaihtuu. Edelleen,

$$\left(\frac{dx}{n}\right) = \left(\frac{x}{n}\right),$$

ts. kertoen luvulla d modulo n Jacobin symboli ei vaihdu, mutta

$$\left(\frac{dx}{p}\right) = -\left(\frac{x}{p}\right) \quad \text{ja} \quad \left(\frac{dx}{q}\right) = -\left(\frac{x}{q}\right)$$

eli Legendren symbolit vaihtuvat. Tästä seuraa kohdat (ii) ja (iii). □

Kuten edellisessä pykälässä todettiin, jos $x \not\equiv 0 \mod p, q$, missä $n = pq$ ja p ja q ovat erisuuret parittomat alkuluvut, niin on tarkalleen neljä eri x:n neliöjuurta modulo n. Ne voidaan kirjoittaa muotoon $(\pm u, \mathrm{mod}\ n)$ ja $(\pm v, \mathrm{mod}\ n)$, missä $u \not\equiv \pm v \mod n$. Neliöjuuria u ja v, joille $u \not\equiv \pm v \mod n$, kutsutaan *erottuviksi* neliöjuuriksi modulo n. Myös todettiin, että jos tunnetaan erottuvat neliöjuuret modulo n, niin voidaan n jakaa tekijöihin. Erottuville neliöjuurille u ja v on ilmeisesti joko

$$u \equiv v \mod p \quad \text{ja} \quad u \equiv -v \mod q$$

tai sitten

$$u \equiv -v \mod p \quad \text{ja} \quad u \equiv v \mod q$$

(muutoinhan $u \equiv \pm v \mod n$ ja ne eivät olisi erottuvat). Tietyissä tilanteissa erottuvilla neliöjuurilla on myös eri Jacobin symbolit:

Lause 6.15. *Jos $n = pq$, missä p ja q ovat erisuuria alkulukuja ja $p \equiv q \equiv 3 \mod 4$, ja u ja v ovat erottuvat neliöjuuret modulo n, niin*

$$\left(\frac{u}{n}\right) = -\left(\frac{v}{n}\right).$$

Todistus. Jos $p = 4l + 3$, niin Lauseen 5.12 nojalla

$$\left(\frac{-1}{p}\right) = (-1)^{\frac{p-1}{2}} = (-1)^{2l+1} = -1$$

ja vastaavasti $\left(\frac{-1}{q}\right) = -1$. Näin ollen

$$\left(\frac{u}{n}\right) = \left(\frac{u}{p}\right)\left(\frac{u}{q}\right) = \left(\frac{\pm v}{p}\right)\left(\frac{\mp v}{q}\right) = -\left(\frac{v}{p}\right)\left(\frac{v}{q}\right) = -\left(\frac{v}{n}\right).$$

□

Seuraus. *Jos $n = pq$, missä p ja q ovat erisuuria alkulukuja ja $p \equiv q \equiv 3 \mod 4$ ja $\mathrm{syt}(x, n) = 1$, niin tarkalleen yksi x:n neliöjuurista modulo n on myös neliönjäännös modulo n. (Sitä kutsutaan x:n* pääneliöjuureksi *modulo n.)*

Todistus. Lauseen 6.15 nojalla jollekin x:n neliöjuurelle u on

$$\left(\frac{u}{n}\right) = 1 \quad \text{eli} \quad \left(\frac{u}{p}\right) = \left(\frac{u}{q}\right).$$

Erottuvalle neliöjuurelle v on silloin saman Lauseen nojalla $\left(\frac{v}{n}\right) = -1$ eikä se voi näin olla neliönjäännös modulo n. Lauseen 5.15 todistuksessa todettiin vielä, että

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

joten

$$\left(\frac{-u}{p}\right) = -\left(\frac{u}{p}\right) = -\left(\frac{u}{q}\right) = \left(\frac{-u}{q}\right).$$

Jompikumpi luvuista $\pm u$ on siis neliönjäännös modulo n, mutta vain toinen. □

6.8 Vahvat satunnaisluvut. Blum–Blum–Shub-generaattori

Kryptologisesti vahvoja satunnaislukuja tarvitaan mm. probabilistisissa kryptosysteemeissä, joissa kryptauksessa käytetään satunnaislukuja. Yhden ja saman viestin kryptaus voi näin ollen tuottaa eri kerroilla eri tuloksen. Myös monet protokollat käyttävät satunnaislukuja.

Monet muutoin hyviksi havaitut perinteiset satunnaislukugeneraattori, kuten pykälässä 2.6 esitetty siirtorekisterigeneraattori, ovat osoittautuneet kryptografian kannalta vaarallisiksi. Kryptologian tarpeet ovatkin käynnistäneet laajan pseudosatunnaislukujen tutkimuksen, niin teorian kuin käytännönkin tasolla.

Blum–Blum–Shub-generaattori[10] on yksinkertainen satunnaislukugeneraattori, jonka vahvuus on kytköksissä neliönjäännöstestiin. Koska tällä hetkellä ei tunneta nopeaa probabilististakaan algoritmia neliönjäännöstestiin, saati sitten determinististä, BBS-generaattoria pidetään vahvana kryptologisessa mielessä, ks. esimerkiksi STINSON.

BBS-generaattori perustuu Lauseen 6.15 Seuraukseen. Pääneliöjuuren otto neliönjäännöksestä modulo n tuottaa uuden neliönjäännöksen ja onkin näin neliönjäännösten permutaatio, ja samoin on sen käänteisoperaatio, neliönjäännöksen korottaminen neliöön modulo n. Tämä permutointi on niin tehokkaasti satunnaistava, että sitä voidaan käyttää satunnaislukugeneraattorina.

BBS-generaattori tuottaa jonon satunnaisbittejä. Generaattori tarvitsee RSAn tapaan kaksi jotakuinkin samanpituista alkulukua p ja q, joita ei paljasteta. Lisäksi vaaditaan Lauseen 6.15 ehto $p \equiv q \equiv 3 \mod 4$. Merkitään tavalliseen tapaan $n = pq$. Jos on tarkoitus tuottaa l satunnaisbittiä, menetellään seuraavasti:

Blum–Blum–Shub-generaattori:

1. Valitaan (satunnainen) neliönjäännös s_0 modulo n väliltä $1 \leq s_0 < n$. Satunnaisuus on tässä tärkeää. Jotkut valinnat nimittäin johtavat varsin lyhyeen jaksoon, ts. generaattori alkaa toistaa itseään varsin pian, mikä tietenkin on vakava puute. Asiaa käsittelee mm. alkuperäisartikkeli[10].

2. Toistetaan l kertaa rekursiota
$$s_i = (s_{i-1}^2, \mod n)$$
ja lasketaan bitit
$$b_i = (s_i, \mod 2) \quad (i = 1, 2, \ldots, l).$$

3. Tulostetaan $(b_1, b_2, \ldots, b_l)$ ja lopetetaan.

Huomautus. *Kryptologisesti vahvoilla satunnaislukugeneraattoreilla ja hyvillä kryptosysteemeillä on paljon yhteistä, itse asiassa monet kryptosysteemit voidaan muuntaa kryptologisesti vahvoiksi satunnaislukugeneraattoreiksi, ks. esimerkiksi* STINSON *tai asiaan liittyvä artikkeliviite* AIELLO, W. & RAJAGOPALAN, S.R. & VENKATESAN, R.: *Design of Practical and Provably Good Random Number Generators.* Journal of Algorithms **29** *(1998), 358–389.*

[10] Alkuperäisviite on BLUM, L. & BLUM, M. & SHUB, M.: A Simple Unpredictable Random Number Generator. *SIAM Journal on Computing* **15** (1986), 364–383.

Luku 7

RSA JA RABIN

7.1 RSAn määrittely

RSAn[1] salainen avain k_2 muodostuu kahdesta suuresta likimain samaa pituutta olevasta eri alkuluvusta p ja q sekä luvusta b (ns. *dekryptauseksponentti*), jolle

$$\mathrm{syt}(b, \phi(pq)) = \mathrm{syt}(b, (p-1)(q-1)) = 1.$$

Julkinen avain k_1 puolestaan muodostuu luvusta $n = pq$ (auki kerrottuna) sekä luvusta a (ns. *kryptauseksponentti*), jolle

$$ab \equiv 1 \mod \phi(n)$$

(huomaa, että b:llä on käänteisluku modulo $\phi(n)$). Kryptausfunktio on

$$e_{k_1}(w) = (w^a, \mod n)$$

ja dekryptausfunktio on

$$e_{k_2}(c) = (c^b, \mod n).$$

Jotta kryptaus toimisi, on viestilohkon oltava valmiiksi koodattu välillä $0 \leq w \leq n-1$ olevaksi kokonaisluvuksi. Sekä kryptaus että dekryptaus tapahtuvat Venäläisten talonpoikien algoritmia tms. käyttäen nopeasti. Seuraava aputulos on hyvin käyttökelpoinen:

Apulause. $a \equiv b \mod n$ *tarkalleen silloin, kun sekä* $a \equiv b \mod p$ *että* $a \equiv b \mod q$.

Todistus. $a \equiv b \mod n$ tarkalleen silloin, kun $n \mid a-b$ eli tarkalleen silloin, kun sekä $p \mid a-b$ että $q \mid a-b$. □

Pystytettäessä RSA-kryptausta, käydään läpi seuraavat askeleet:

1. Generoidaan satunnaiset halutun pituiset alkuluvut p ja q (ks. pykälä 6.2).
2. Kerrotaan p ja q luvuksi $n = pq$ ja lasketaan kertomalla $\phi(n) = (p-1)(q-1)$.
3. Etsitään sellainen satunnainen luku b väliltä $1 \leq b \leq \phi(n) - 1$, että $\mathrm{syt}(b, \phi(n)) = 1$, generoimalla satunnaisesti lukuja ko. väliltä ja testaamalla s.y.t.
4. Lasketaan b:n käänteisluku a modulo $\phi(n)$ Eukleideen algoritmilla.

[1] Alkuperäisviite on RIVEST, R.L. & SHAMIR, A. & ADLEMAN, L.: A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM* **21** (1978), 120–126.

5. Julkistetaan pari $k_1 = (n, a)$.

Todennetaan, että dekryptaus toimii. Ensinnäkin, jos $\mathrm{syt}(w, n) = 1$, niin Eulerin lauseen mukaan (jollekin luvulle l)

$$c^b \equiv (w^a)^b = w^{ab} = w^{1+l\phi(n)} = w(w^{\phi(n)})^l \equiv w \cdot 1 = w \mod n.$$

Jos taas $\mathrm{syt}(w, n) \neq 1$, niin on kolme vaihtoehtoa:

- $w = 0$. Nyt ilmeisesti
$$c^b \equiv (w^a)^b = 0^b = 0 \mod n.$$
- $p \mid w$, mutta $w \neq 0$. Nyt $w = pt$, missä $\mathrm{syt}(q, t) = 1$. Selvästi
$$c^b \equiv w^{ab} \equiv w \mod p.$$
Toisaalta Fermat'n pienen lauseen mukaan myös (jollekin luvulle l)
$$w^{ab} = w^{1+l\phi(n)} = w(w^{\phi(n)})^l = w(w^{(p-1)(q-1)})^l = w(w^{q-1})^{l(p-1)} \equiv w \cdot 1 = w \mod q.$$
Apulauseen nojalla $c^b \equiv w^{ab} \equiv w \mod n$.
- $q \mid w$, mutta $w \neq 0$. Käsitellään aivan vastaavasti kuin edellinen kohta.

Huomautus. *Mainittu ehto* $\mathrm{syt}(w, n) \neq 1$ *ei tiedä hyvää: Joko viesti on suoraan luettavissa (*$w = c = 0$*) tai sitten se sisältää tekijänään* p*:n tai* q*:n, jolloin Eukleideen algoritmilla saadaan* $\mathrm{syt}(w, n)$ *ja sitä kautta koko systeemi murretuksi. Samoin tietysti käy, jos* $\mathrm{syt}(c, n) \neq 1$*, mutta koska* n*:llä ei ole tekijänään alkuluvun korkeampia potensseja ja* $c \equiv w^a \mod n$*, on itse asiassa*

$$\mathrm{syt}(c, n) = \mathrm{syt}(w^a, n) = \mathrm{syt}(w, n).$$

7.2 Hyökkäyksiä ja puolustuksia

RSA on saatavissa hyvin varmaksi, mutta edellytyksenä on, että tiettyjä vaarallisia valintoja vältetään. (Huomaa, että julkisen avaimen systeemissä on KP-dataa aina saatavilla). Yo. Huomautuksessa jo oli eräs vältettävä tapaus, tosin varsin harvinainen. Muita huomioon otettavia asioita ovat seuraavat:

(A) Erotus $p-q$ ei saa olla itseisarvoltaan pieni! Jos nimittäin $p-q > 0$ on pieni, niin $(p-q)/2$ on pieni ja $(p+q)/2$ on (vähän) suurempi kuin $\sqrt{pq} = \sqrt{n}$ (totea!). Toisaalta

$$n = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2.$$

n:n jakamiseksi tekijöihin p ja q testataan yksi kerrallaan kokonaislukuja x lähtien $\lceil\sqrt{n}\rceil$:stä kunnes löytyy sellainen luku x, että $x^2 - n = y^2$ on neliö. Kun tämä x löytyy, saadaan välittömästi $p = x + y$ ja $q = x - y$. Koska n ei itse ole neliö, on $\lceil\sqrt{n}\rceil = \lfloor\sqrt{n}\rfloor + 1$, ja kokonaisneliöjuuren laskeminen on nopeaa, ks. pykälä 6.4.

(B) Valittaessa alkulukuja p ja q pitää myös $\phi(n)$:n tekijärakennetta pitää silmällä. Jos $\mathrm{syt}(p-1, q-1)$ on suuri, niin

$$u = \mathrm{pyj}(p-1, q-1) = \frac{(p-1)(q-1)}{\mathrm{syt}(p-1, q-1)}$$

(ks. Lause 2.9) on pieni. Toisaalta $\mathrm{syt}(a, u) = 1$ (miksi?) ja a:lla on käänteisalkio b' modulo u. Tämä b' toimii myös dekryptauseksponenttina. Tällöin nimittäin voidaan kirjoittaa $ab' = 1 + lu$ sekä $u = t(p-1) = s(q-1)$ joillekin luvuille l, t ja s ja Fermat'n pienen lauseen mukaan

$$c^{b'} \equiv w^{ab'} = w^{1+lu} = w(w^u)^l = w(w^{p-1})^{lt} \equiv w \cdot 1 = w \mod p.$$

(Tietysti $c \equiv w^a \mod p$.) Vastaavasti $c^{b'} \equiv w \mod q$ ja Apulauseen nojalla myös $c^{b'} \equiv w \mod n$. Jos u on paljon pienempi kuin $\phi(n)$, voidaan b' löytää kokeilemalla. Johtopäätöksenä on, että $p-1$:llä ja $q-1$:llä ei saisi olla suurta yhteistä tekijää.

(C) Myös pitää välttää tilannetta, jossa $\phi(n)$:llä on vain pieniä alkutekijöitä. Paitsi että tässä tilanteessa Pollardin $p-1$-algoritmilla (ja muilla vastaavilla) voidaan silloin yrittää jakaa n:ää tekijöihin, saattaa myös olla mahdollista käydä läpi kaikki $\phi(n)$-ehdokkaat f, joille $\mathrm{syt}(f, a) = 1$, laskea a:n käänteisluku modulo f, dekryptata jokin kryptoteksti ja näin löytää kokeilemalla $\phi(n)$. Huomaa, että jos $\phi(n) = (p-1)(q-1)$ ja n tunnetaan, saadaan niistä helposti p ja q toisen asteen yhtälön

$$(x-p)(x-q) = x^2 + (\phi(n) - n - 1)x + n = 0$$

juurina. Juuret

$$x_{1,2} = \frac{-\phi(n) + n + 1 \pm \sqrt{(\phi(n) - n - 1)^2 - 4n}}{2}$$

voidaan nimittäin löytää kokonaisneliöjuurta käyttäen varsin nopeasti.

(D) *Iteroidulla kryptauksella* voidaan toisinaan löytää selväteksti w, kun vastaava kryptoteksti c_0 tunnetaan. Tapaus, jossa $\mathrm{syt}(c_0, n) = \mathrm{syt}(w, n) \neq 1$ johtaa suoraan murtoon (vrt. eo. Huomautus), joten sivuutetaan se. Lasketaan rekursiivisesti jonoa

$$c_i = (c_{i-1}^a, \mod n) = (w^{a^{i+1}}, \mod n),$$

kunnes löytyy tunnistettava selväteksti w. Eulerin lauseen mukaan on nimittäin

$$a^{\phi(\phi(n))} \equiv 1 \mod \phi(n)$$

eli voidaan kirjoittaa $a^{\phi(\phi(n))} - 1 = l\phi(n)$. Toisaalta edelleen Eulerin lauseen nojalla

$$w^{a^{\phi(\phi(n))}} = w^{1+l\phi(n)} = w(w^{\phi(n)})^l \equiv w \cdot 1 = w \mod n,$$

joten ainakin $i = \phi(\phi(n)) - 1$ riittää. Tällaisen hyökkäyksen onnistumismahdollisuudet ovat pienet, jos $p-1$:llä ja $q-1$:llä on suuret alkutekijät p' ja q', vastaavasti, ja toisaalta myös $p'-1$:llä ja $q'-1$:llä on suuria alkutekijöitä, koska silloin pakosta myös $\phi(\phi(n))$ on suuri. (Ks. Lause 6.1.)

(E) Ilmeisesti pitää välttää liian pientä dekryptauseksponenttia, koska se voidaan silloin löytää kokeilemalla. (Myös pienestä kryptauseksponentista voi olla haittaa, vaikka dekryptauseksponentti olisi isokin.)

Huomautus. *(B)- ja (C)-kohdissa voidaan vaikeus välttää rajoittumalla ns.* turvallisiin alkulukuihin *eli* Germainin lukuihin p *ja* q*, ts. sellaisiin, että myös* $(p-1)/2$ *ja* $(q-1)/2$ *ovat alkulukuja. Valitettavasti vain tällaisten löytäminen on vaikeaa (eikä edes tiedetä onko niitä äärettömän monta!).*

Harmillisia ovat myös ns. *kiintopisteviestit*, ts. sellaiset viestit w, että

$$c = e_{k_1}(w) = w.$$

Ilmeisesti 0, 1 ja $n-1$ ovat tällaisia viestejä. Mutta niitä on yleensä enemmänkin!

Lause 7.1. *Kiintopisteviestejä on tarkalleen*

$$(1+\mathrm{syt}(a-1,p-1))(1+\mathrm{syt}(a-1,q-1)) \quad \textit{kpl.}$$

Todistus. Merkitään $l = \mathrm{syt}(a-1,p-1)$ ja $k = \mathrm{syt}(a-1,q-1)$ sekä valitaan jotkin primitiiviset juuret g_1 ja g_2 modulo p ja q, vastaavasti. Silloin g_1^{a-1}:n kertaluku modulo p on $(p-1)/l$ ja g_2^{a-1}:n kertaluku modulo q on $(q-1)/k$ (ks. pykälä 6.1). Näin ollen ainoat sellaiset luvut i väliltä $0 \le i < p$, että

$$(g_1^{a-1})^i \equiv 1 \mod p \quad \text{eli} \quad (g_1^i)^a \equiv g_1^i \mod p,$$

ovat luvut

$$i_j = j\frac{p-1}{l} \quad (j = 0,1,\ldots,l-1).$$

Vastaavasti ainoat luvut i väliltä $0 \le i < q$, joille $(g_2^i)^a \equiv g_2^i \mod q$, ovat luvut

$$h_m = m\frac{q-1}{k} \quad (m = 0,1,\ldots,k-1).$$

Ilmeisesti jokainen kiintopisteviesti w toteuttaa kongruenssit $w^a \equiv w \mod p,q$, ja kääntäen. Näin ollen tarkalleen kaikki kiintopisteviestit saadaan Kiinalaisen jäännöslauseen avulla $(l+1)(k+1)$ kongruenssiparista

$$\begin{cases} x \equiv g_1^{i_j}, 0 \mod p \\ x \equiv g_2^{h_m}, 0 \mod q \end{cases} \qquad (j = 0,1,\ldots,l-1; m = 0,1,\ldots,k-1).$$

□

Kiintopisteviestejä ei tietenkään saisi olla paljon. Koska sekä a että p ja q ovat käytännössä parittomia, on kiintopisteitä yleisesti ainakin $(1+2)(1+2) = 9$ kpl. Erityisen hankala on tilanne, jossa $p-1 \mid a-1$ ja $q-1 \mid a-1$. Silloin nimittäin kiintopisteviestejä on $(1+p-1)(1+q-1) = n$ kpl eli *kaikki viestit* ovat kiintopisteviestejä. Mikäli g_1 ja g_2 ovat tiedossa ja kiintopisteviestien määrä on suhteellisen pieni, voidaan ne etsiä etukäteenkin ja välttää niitä.

7.3 Kryptanalyysi ja tekijöihinjako

Vaikeus RSAn murtamisessa piilee siinä, ettei auki kerrotun n:n tekijöitä saada millään helpolla tavalla lasketuksi. Julkisessa avaimessa on myös kryptauseksponentti a. Seuraava tulos osoittaa, ettei a:stakaan saada millään helpolla tavalla lisätietoa. Ts. algoritmi A, joka n:stä ja a:sta laskee b:n, voidaan muuntaa probabilistiseksi algoritmiksi, millä voi jakaa nopeasti n:n tekijöihinsä.

Jos jotain kautta tunnetaan 1:n neliöjuuri $\omega \not\equiv \pm 1 \mod n$, niin tästä saadaan nopeasti lasketuksi n:n tekijät, kuten pykälässä 6.6 todettiin. Seuraava algoritmi käyttää tätä ideaa ja oletettua algoritmia A ja yrittää jakaa n:n tekijöihin:

1. Valitaan satunnaisesti viesti w, $1 \leq w < n$.
2. Lasketaan $d = \mathrm{syt}(w, n)$ Eukleideen algoritmilla.
3. Jos $1 < d < n$, tulostetaan d ja n/d ja lopetetaan.
4. Lasketaan b käyttäen algoritmia A ja asetetaan $y \leftarrow ab - 1$.
5. Jos y on pariton, mennään kohtaan 7.
6. Jos y on parillinen, asetetaan $y \leftarrow y/2$ ja mennään kohtaan 5. (Jos $ab - 1 = 2^s r$, missä r on pariton, kierretään tätä silmukkaa s kertaa.)
7. Lasketaan $\omega = (w^y, \mathrm{mod}\ n)$ Venäläisten talonpoikien algoritmilla.
8. Jos $\omega \equiv 1 \mod n$, tulostetaan ”FAIL” ja lopetetaan.
9. Jos $\omega \not\equiv 1 \mod n$, asetetaan $\omega' \leftarrow \omega$ ja $\omega \leftarrow (\omega^2, \mathrm{mod}\ n)$ ja mennään kohtaan 9. (Tätä silmukkaa joudutaan kiertämään enintään s kertaa, sillä $ab-1 = 2^s r$ on jaollinen $\phi(n)$:llä ja toisaalta Eulerin lauseen mukaan $w^{\phi(n)} \equiv 1 \mod n$.)
10. Lopulta saadaan sellainen 1:n neliöjuuri ω' modulo n, että $\omega' \not\equiv 1 \mod n$. Jos nyt $\omega' \equiv -1 \mod n$, tulostetaan ”FAIL” ja lopetetaan. Muutoin lasketaan $t = \mathrm{syt}(\omega' - 1, n)$, tulostetaan t ja n/t, ja lopetetaan.

Menettely on probabilistinen Las Vegas -tyyppinen algoritmi (kohta 1. on satunnainen). Voidaan osoittaa, että se tuottaa tuloksen ainakin todennäköisyydellä $1/2$ (ks. esimerkiksi STINSON tai SALOMAA).

Kaikesta ylläolevasta huolimatta ei ole pystytty näyttämään, että RSAn murtaminen välttämättä johtaisi n:n tekijöihinjakoon.

7.4 Osittaisen tiedon saaminen viestin biteistä

Vaikka itse viestin saaminen selville tuntuukin olevan vaikeaa, olisiko mahdollista saada selville osittaista tietoa viestistä, esimerkiksi onko viesti parillinen tai kummalla väleistä $0 \leq w < n/2$ vai $n/2 < w < n$ viesti on? (Tässä tietysti oletetaan, että n on pariton.) Näin saadaan kaksi tehtävää:

(1) Lähtien kryptotekstistä $c = e_{k_1}(w)$ laske w:n *pariteetti*

$$\mathrm{par}(c) = \begin{cases} 0, \text{ jos } w \text{ on parillinen} \\ 1, \text{ jos } w \text{ on pariton.} \end{cases}$$

(2) Lähtien kryptotekstistä $c = e_{k_1}(w)$ laske w:n *puolisko*

$$\mathrm{half}(c) = \begin{cases} 0, \text{ jos } 0 \leq w < n/2 \\ 1, \text{ jos } n/2 < w < n. \end{cases}$$

Nämä kaksi tehtävää eivät ole riippumattomat:

Apulause. *Funktiot* par *ja* half *liittyvät toisiinsa yhtälöillä*

$$\mathrm{half}(c) = \mathrm{par}((ce_{k_1}(2), \mathrm{mod}\ n)) \quad \textit{ja} \quad \mathrm{par}(c) = \mathrm{half}((ce_{k_1}(2^{-1}), \mathrm{mod}\ n)).$$

Todistus. Merkitään ensin

$$c' = (ce_{k_1}(2), \bmod n) = ((2w)^a, \bmod n).$$

Jos nyt $\mathrm{half}(c) = 0$, niin $0 \leq 2w < n$ eli $2w$ on c':a vastaava selväteksti ja $\mathrm{par}(c') = 0$. Jos taas $\mathrm{half}(c) = 1$, niin $n/2 < w < n$ eli $0 < 2w - n < n$. Tällöin siis $2w - n$ on c':a vastaava selväteksti ja se on pariton, joten $\mathrm{par}(c') = 1$.

Merkitään sitten

$$c'' = (ce_{k_1}(2^{-1}), \bmod n) = ((w2^{-1})^a, \bmod n).$$

Jos nyt $\mathrm{half}(c'') = 0$, niin $0 \leq (w2^{-1},\bmod n) < n/2$ eli c:tä vastaava selväteksti $w = 2(w2^{-1},\bmod n)$ on parillinen ja $\mathrm{par}(c) = 0$. Jos taas $\mathrm{half}(c'') = 1$, niin $n/2 < (w2^{-1},\bmod n) < n$ eli $0 < 2(w2^{-1},\bmod n) - n < n$. Tällöin c:tä vastaava selväteksti $w = 2(w2^{-1},\bmod n) - n$ on pariton ja $\mathrm{par}(c) = 1$. □

Näin ollen riittää tarkastella vain funktiota half. Lasketaan nyt luvut

$$c_i = \mathrm{half}((ce_{k_1}(2)^i, \bmod n)) = \mathrm{half}(((2^i w)^a, \bmod n)) \quad (0 \leq i \leq \lfloor \log_2 n \rfloor).$$

(Tässä tietysti $2^i w$ voidaan tarvittaessa korvata "oikealla" viestillä $(2^i w,\bmod n)$.) Ilmeisesti seuraavat loogiset ekvivalenssit pätevät:

$$c_0 = 0 \Leftrightarrow 0 \leq w < \frac{n}{2}$$

$$c_1 = 0 \Leftrightarrow 0 \leq w < \frac{n}{4} \text{ tai } \frac{n}{2} < w < \frac{3n}{4}$$

$$c_2 = 0 \Leftrightarrow 0 \leq w < \frac{n}{8} \text{ tai } \frac{n}{4} < w < \frac{3n}{8} \text{ tai } \frac{n}{2} < w < \frac{5n}{8} \text{ tai } \frac{3n}{4} < w < \frac{7n}{8}$$

jne.

w voidaan näin löytää binäärisellä etsinnällä $\lfloor \log_2 n \rfloor + 1$:llä askeleella.

Kaiken kaikkiaan voidaan päätellä, että algoritmi, joka laskee jommankumman funktioista par tai half, voidaan muuntaa algoritmiksi, jolla voidaan polynomiajassa dekryptata mielivaltainen kryptoteksti.

Huomautus. *Jos toisaalta tunnetaan jokin määrä dekryptausavaimen desimaaleja/bittejä, voidaan loput laskea, ks.* BONEH, D. & DURFEE, G. & FRANKEL, Y.: An Attack on RSA Given a Small Fraction of the Private Key Bits. *Proceedings of ASIACRYPT '98. Lecture Notes in Computer Science* **1514**. Springer–Verlag (2000).

7.5 Rabinin kryptosysteemi RABIN

Rabinin kryptosysteemissä[2] RABIN valitaan ensin kaksi eri paritonta alkulukua p ja q ja kerrotaan ne: $n = pq$. Lisäksi valitaan luku B väliltä $0 \leq B < n$. Julkinen avain on $k_1 = (n, B)$ ja salainen avain on $k_2 = (p, q)$. RABINin pystyttäminen on näin samantapaista kuin RSAn ja varsin helppoa.

Viestilohko w on kuten RSAssakin luku väliltä $0 \leq w < n$. Kryptaus- ja dekryptausfunktiot ovat

$$e_{k_1}(w) = (w(w + B), \bmod n) \quad \text{ja} \quad d_{k_2}(c) = (y - B2^{-1}, \bmod n),$$

[2] Alkuperäisviite on RABIN, M.O.: Digitized Signatures and Public-Key Functions as Intractable as Factorization. *MIT Laboratory for Computer Science Technical Report* **LCS/TR-212** (1979).

missä y on luvun $C = (B^2 2^{-2} + c, \mathrm{mod}\ n)$ neliöjuuri modulo n.[3] Huomaa, että C on neliönjäännös modulo n, sillä

$$B^2 2^{-2} + c \equiv B^2 2^{-2} + w^2 + Bw \equiv (w + B2^{-1})^2 \mod n,$$

josta myös näkyy, että dekryptaus ”toimii”. Moduläärisen neliöjuuren ottoon tarvitaan Shanksin algoritmi.

Koska dekryptauksessa kuitenkin on kyseessä neliöjuuren otto modulo n, niin, toisin kuin yleensä, tämä kryptausfunktio ei ole injektiivinen, ts. useaa selvätekstiä vastaa sama kryptoteksti. Jos nimittäin $\omega_1, \omega_2, \omega_3$ ja ω_4 ovat ne neljä 1:n neliöjuurta modulo n (ks. Lause 6.10 ja sen Seuraus), niin kaikki selvätekstit $(\omega_i y - B2^{-1}, \mathrm{mod}\ n)$ ($i = 1, 2, 3, 4$) vastaavat kryptotekstiä c. Nämä voidaan erottaa dekryptattaessa toisistaan vain sisällön perusteella. Tämä voi olla vaikeaa, jos selvätekstillä ei ole erikoista sisällön muotoa, esimerkiksi, jos kyseessä on pakattu data (ks. kurssi Informaatioteoria).

Toisin kuin RSAlle, RABINille dekryptauksen tiedetään olevan ekvivalentti n:n nopean tekijöihinjaon kanssa. Itse asiassa algoritmi A, joka dekryptaa RABINin, voidaan muuntaa Las Vegas -tyyppiseksi n:n tekijöihinjakoalgoritmiksi seuraavasti:

1. Valitaan satunnaisesti luku r väliltä $1 \leq r < n$.
2. Lasketaan $c = (r^2 - B^2 2^{-2}, \mathrm{mod}\ n)$. Huomaa, että tämä c on silloin selvätekstiä $w = (r - B2^{-1}, \mathrm{mod}\ n)$ vastaava kryptoteksti.
3. Dekryptataan c käyttäen algoritmia A. Tulos on jokin selväteksti u, mahdollisesti mutta ei välttämättä w.
4. Lasketaan $v = (u + B2^{-1}, \mathrm{mod}\ n)$. Silloin $v^2 \equiv r^2 \mod n$.
5. Jos $v \equiv \pm r \mod n$, tulostetaan ”FAIL” ja lopetetaan.
6. Lasketaan $s = \mathrm{syt}(v + r, n)$, tulostetaan s ja n/s, ja lopetetaan.

Jos kohdassa 5. on $v \not\equiv \pm r \mod n$, tiedetään, että v on jompikumpi niistä kahdesta muusta r^2:n neliöjuuresta modulo n. Silloin n on tulon $(v - r)(v + r)$ mutta ei kummankaan tulontekijän tekijä ja n:n tekijät ovat $\mathrm{syt}(v \pm r, n)$. Ks. Huomautus pykälässä 6.6. Voidaan osoittaa, että algoritmi menestyy todennäköisyydellä $1/2$ (ks. STINSON).

RABINin murtaminen CP-dataa käyttäen on siis vaikeaa. Jos kuitenkin käytettävissä on sopivaa CC-dataa, RABIN murtuu helposti (itse asiassa yo. algoritmilla!).

Huomautus. *Dekryptaus Shanksin algoritmilla on yleisesti hidasta. Jos kuitenkin rajoitutaan muotoa $4l - 1$ oleviin alkulukuihin p ja q, voidaan neliöjuuri modulo n laskea hyvin nopeasti Eulerin kriteerillä ja potenssiin korotuksella, kuten pykälässä 6.6 todettiin. Alkuperäisessä RABINissa tehtiinkin juuri näin. Tämä ei tiettävästi muuta n:n tekijöihinjakoa yhtään helpommaksi, joten murtovarmuus ei vähene. Toisaalta asymptoottisesti noin puolet alkuluvuista on mainittua muotoa, joten myös systeemin pystytys ei ole sen vaikeampaa.*

[3] Tässä esiintyvä 2:n inverssi on nimenomaan moduläärinen inverssi modulo n. Näille pätevät tutut laskusäännöt, esimerkiksi $2^{-2} + 2^{-2} \equiv 2^{-1} \mod n$.

Luku 8

EKSKURSIO ALGEBRAAN 2

8.1 Ryhmät

Ryhmä on struktuuri $G = (A, \odot, 1)$, jossa $\odot$ on laskuoperaatio (ns. *ryhmäoperaatio*) ja 1 on ns. ryhmän *identiteettialkio.* Lisäksi vaaditaan, että seuraavat ehdot ovat voimassa:

(1) $(a \odot b) \odot c = a \odot (b \odot c)$ ($\odot$ on liitännäinen eli assosiatiivinen).

(2) $a \odot 1 = 1 \odot a = a$.

(3) Jokaiselle alkiolle a on olemassa yksikäsitteinen alkio a^{-1}, ns. a:n *käänteisalkio* eli *inverssi,* jolle $a \odot a^{-1} = a^{-1} \odot a = 1$.

Edelleen vaaditaan luonnollisesti, että laskuoperaatio $a \odot b$ on määritelty kaikille alkioille a ja b ja että tulos on yksikäsitteinen. Ryhmäoperaatio luetaan usein ”kertaa” ja sitä kutsutaan *tuloksi.* Jos lisäksi vielä

(4) $a \odot b = b \odot a$ ($\odot$ on vaihdannainen eli kommutatiivinen),

niin sanotaan, että G on *vaihdannainen ryhmä* (eli *kommutatiivinen ryhmä* eli *Abelin ryhmä*).

Liitännäisyydestä johtuen voidaan kirjoittaa

$$a_1 \odot a_2 \odot \cdots \odot a_n$$

ilman sulkuja, tulos ei riipu sulutuksesta.[1] Edelleen merkitään (kuten pykälässä 4.1)

$$a^n = \underbrace{a \odot \cdots \odot a}_{n \text{ kpl}}, \quad a^{-n} = \underbrace{a^{-1} \odot \cdots \odot a^{-1}}_{n \text{ kpl}} \quad \text{ja} \quad a^0 = 1$$

ja tavanomaiset potenssilaskusäännöt pätevät. Potenssit voidaan myös laskea Venäläisten talonpoikien menetelmällä.

Huomautus. *Vaihdannaisia ryhmiä kutsutaan usein myös* additiivisiksi ryhmiksi. *Tällöin käytetään tavallisesti myös seuraavia additiivisia merkintöjä ja nimityksiä: Ryhmäoperaatiota merkitään $\oplus$:lla tai $+$:lla tms. ja kutsutaan* summaksi. *Usein se luetaan ”plus”. Identiteettialkiota taas kutsutaan* nolla-alkioksi, *ja merkitään 0:lla tai 0:lla tms. Käänteisalkiota a^{-1} kutsutaan* vasta-alkioksi *ja merkitään $-a$:lla. Potenssia a^n kutsutaan* monikerraksi *ja merkitään na:lla. Vrt. myös merkinnät pykälässä 4.1.*

[1] Tämä ei ole ihan helppo todistaa, jos ei kovin vaikeakaan, ks. vaikkapa moniste RUOHONEN, K.: Symbolinen analyysi.

Esimerkkejä:

- Tuttu ryhmä on $(\mathbb{Z}, +, 0)$ (kokonaisluvut ja yhteenlasku), merkitään lyhyesti vain $\mathbb{Z}$:lla. Käänteisalkiot ovat vastalukuja ja ryhmä on vaihdannainen.

- $(\mathbb{Z}_m, +, \overline{0})$ (jäännösluokat modulo m ja yhteenlasku) on myös vaihdannainen ryhmä, käänteisalkiot ovat vastaluokkia. Tätä kutsutaan *jäännösluokkaryhmäksi* ja merkitään lyhyesti vain $\mathbb{Z}_m$:llä.

- Ei-singulääriset reaalialkioiset $n \times n$-matriisit muodostavat matriisikertolaskun suhteen ryhmän $(\mathbb{R}^{n\times n}, \cdot, \mathbf{I}_n)$, joka ei ole vaihdannainen (ellei $n = 1$). Identiteettialkio on $n \times n$-identiteettimatriisi $\mathbf{I}_n$ ja käänteisalkiot ovat käänteismatriiseja.

- Jos merkitään $\mathbb{Z}_m^*$:llä alkuluokkia modulo m (ks. pykälä 2.4), niin $(\mathbb{Z}_m^*, \cdot, \overline{1})$ on vaihdannainen ryhmä, käänteisalkiot ovat käänteisluokkia. Huomaa, että kahden alkuluokan tulo on aina myös alkuluokka. Tätä kutsutaan $\mathbb{Z}_m$:n *yksiköiden ryhmäksi*, merkitään lyhyesti vain $\mathbb{Z}_m^*$, ja siinä on $\phi(m)$ alkiota (alkuluokkaa).

- Jokaisesta renkaasta $R = (A, \oplus, \odot, \mathbb{0}, \mathbb{1})$ (ks. pykälä 4.1) voidaan erottaa sen *additiivinen ryhmä* $R^+ = (A, \oplus, \mathbb{0})$. Jokaisesta kunnasta $F = (A, \oplus, \odot, \mathbb{0}, \mathbb{1})$ voidaan lisäksi erottaa sen *multiplikatiivinen ryhmä* $F^* = (A - \{\mathbb{0}\}, \odot, \mathbb{1})$.

Pienintä sellaista lukua $i \geq 1$ (jos olemassa), että $a^i = \mathbb{1}$, kutsutaan a:n *kertaluvuksi*. Kertaluvun perusominaisuudet ovat samat kuin luvun kertaluvun modulo m pykälässä 6.1 ja todistuksetkin ovat samat (itse asiassa kertaluku modulo m on sama kuin kertaluku ryhmässä $\mathbb{Z}_m^*$):

- Jos $a^j = \mathbb{1}$, niin a:n kertaluku i jakaa j:n.

- Jos a:n kertaluku on i, niin a^j:n kertaluku on

$$\frac{i}{\mathrm{syt}(i,j)} = \frac{\mathrm{pyj}(i,j)}{j}.$$

- Jos a:n kertaluku on i, niin $a^{-1} = a^{i-1}$.

- Jos vaihdannaisessa ryhmässä a:n kertaluku on i ja b:n kertaluku on j ja $\mathrm{syt}(i,j) = 1$, niin $a \odot b$:n kertaluku on ij.

- Äärellisen ryhmän alkioilla on aina kertaluku.

Jos äärellisen ryhmän $G = (A, \odot, \mathbb{1})$ koko on N ja jollekin alkiolle g

$$A = \{\mathbb{1}, g, g^2, \ldots, g^{N-1}\},$$

ts, kaikki ryhmän alkiot ovat g:n potensseja, niin ryhmää kutsutaan *sykliseksi ryhmäksi* ja g:tä sen *generaattoriksi*. Usein tällöin merkitään $G = \langle g \rangle$. Huomaa, että g:n kertaluvun on pakko silloin olla N (miksi?). Myös ääretön ryhmä voi olla syklinen, silloin vaaditaan, että

$$A = \{\mathbb{1}, g^{\pm 1}, g^{\pm 2}, \ldots\}.$$

Syklinen ryhmä on luonnollisesti aina vaihdannainen.

Ilmeisesti esimerkiksi $\mathbb{Z}$ ja $\mathbb{Z}_m$ ovat syklisiä, generaattoreina 1 ja $\overline{1}$. Jos on olemassa primitiivinen juuri modulo m, niin $\mathbb{Z}_m^*$ on syklinen, generaattorina mainittu primitiivinen juuri.

Huomautus. *Äärellinen syklinen ryhmä $\langle g \rangle$, jossa on N alkiota, on rakenteeltaan siis aivan samanlainen (eli isomorfinen) kuin $\mathbb{Z}_N$:*

$$g^i \odot g^j = g^{(i+j, \bmod N)}.$$

Laskeminen $\mathbb{Z}_N$:ssä on hyvin helppoa. Toisaalta laskeminen $\langle g \rangle$:ssä ei ole välttämättä ole ollenkaan helppoa, jos g^i:n ja i:n yhteyttä ei ole helppo laskea. Tätä käytetään lukuisissa kryptosysteemeissä, ks. seuraava luku. Asiaan palataan ns. diskreetin logaritmin yhteydessä.

Äärellisen kunnan $\mathbb{F}_{p^n}$ multiplikatiivinen ryhmä $\mathbb{F}_{p^n}^*$ on aina syklinen. Sen generaattoreita kutsutaan *primitiivisiksi alkioiksi.* Tämähän tuli jo todetuksi Lauseessa 6.4 alkukunnille $\mathbb{Z}_p$, joille generaattoreita kutsutaan myös primitiivisiksi juuriksi modulo p.

Jos $G = (A, \odot, \mathbb{1})$ on ryhmä ja myös $H = (B, \odot, \mathbb{1})$, missä B on A:n osajoukko, on ryhmä, niin H on G:n *aliryhmä.* Esimerkiksi $(2\mathbb{Z}, +, 0)$, missä $2\mathbb{Z}$ on parillisten kokonaislukujen joukko, on $\mathbb{Z}$:n aliryhmä. Tärkeitä aliryhmiä ovat *sykliset aliryhmät* eli *alkioiden generoimat aliryhmät:* Jos a:n kertaluku on i, niin a:n generoimassa aliryhmässä $\langle a \rangle$ on

$$B = \{\mathbb{1}, a, a^2, \ldots, a^{i-1}\}.$$

Jos taas a:lla ei ole kertalukua, niin

$$B = \{\mathbb{1}, a^{\pm 1}, a^{\pm 2}, \ldots\}.$$

On helppo todeta, että kyseessä on aliryhmä.

Jos $G_1 = (A_1, \odot_1, \mathbb{1}_1)$ ja $G_2 = (A_2, \odot_2, \mathbb{1}_2)$ ovat ryhmiä, niin niiden *suora tulo* on ryhmä

$$G_1 \times G_2 = (C, \otimes, (\mathbb{1}_1, \mathbb{1}_2)),$$

missä alkiojoukko on karteesinen tulo

$$C = A_1 \times A_2 = \{(a_1, a_2) \mid a_1 \in A_1 \text{ ja } a_2 \in A_2\}$$

ja operaatio $\otimes$ sekä käänteisalkion otto määritellään kaavoilla

$$(a_1, a_2) \otimes (b_1, b_2) = (a_1 \odot_1 b_1, a_2 \odot_2 b_2) \quad \text{ja} \quad (a_1, a_2)^{-1} = (a_1^{-1}, a_2^{-1}).$$

On helppo todeta, että näin määritelty $G_1 \times G_2$ on tosiaan ryhmä. Ideaa voidaan jatkaa, muodostaa kolmen ryhmän suoria tuloja $G_1 \times G_2 \times G_3$, jne. Todistamatta esitetään seuraava klassinen tulos, joka osoittaa, että ryhmillä $\mathbb{Z}_m$ voidaan oleellisesti karakterisoida kaikki äärelliset vaihdannaiset ryhmät suoraa tuloa käyttäen:

Lause 8.1. (Vaihdannaisten ryhmien kantalause) *Jokainen äärellinen vaihdannainen ryhmä on rakenteeltaan identtinen (eli isomorfinen) jonkin suoran tulon*

$$\mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_k}$$

kanssa. (Tässä pitää sopia, että tyhjä suora tulo vastaa triviaalia ryhmää $\{1\}$, niin saadaan sekin mukaan.)

8.2 Diskreetti logaritmi

Sykliselle ryhmälle voidaan määritellä (*diskreetti*) *logaritmi* kannan (generaattorin) g suhteen:

$$\log_g a = j \quad \text{tarkalleen silloin, kun} \quad a = g^j.$$

Äärellisessä syklisessä ryhmässä, jossa on N alkiota, sovitaan vielä, että $0 \leq \log_g a \leq N-1$.

Esimerkiksi $\mathbb{Z}$:ssa logaritmi on triviaali: Kanta on ± 1 ja $\log_{\pm 1} a = \pm a$. Ryhmässä $\mathbb{Z}_m$:kin se on helppo: Kanta on jokin $\overline{i}$, missä $\mathrm{syt}(i,m) = 1$, ja $\log_{\overline{i}} \overline{j} = (ji^{-1}, \mathrm{mod}\ m)$. Mutta jo $\mathbb{Z}_p^*$:n logaritmi on kaikkea muuta kuin triviaali suurille alkuluvuille p ja on osoittautunut erittäin työlääksi laskea. Myös eräiden muiden ryhmien diskreetit logaritmit ovat vaikeita laskea. Vaikka itse ryhmä G ei olisikaan syklinen (eikä diskreetti logaritmi siis olisi määritelty itse G:ssä), ovat sen syklisissä aliryhmissä logaritmit määriteltyjä.

Tarkastellaan lähemmin $\mathbb{Z}_p^*$:n logaritmia, jota usein kutsutaan myös *indeksiksi*. Tehtävänä on siis etsiä sellainen luku j väliltä $0 \leq j \leq p-2$, että $g^j \equiv b \mod p$, kun generaattori (primitiivinen juuri) g ja b on annettu, vaikkapa positiivisessa jäännössysteemissä desimaalilukuina. Selvästi tehtävä on $\mathcal{NP}$:ssä: Arvataan vain j ja testataan potenssiinkorotuksella (Venäläisten talonpoikien menetelmällä), että se on oikea. Deterministisesti j voidaan edelleen laskea yksinkertaisella etsinnällä (ja Venäläisten talonpoikien algoritmilla) noin ajassa $O(p(\ln p)^3)$ ja polynomitilassa. Laskemalla etukäteen (esiprosessointina) ns. *indeksitaulu*, ts. parit

$$(i, (g^i, \mathrm{mod}\ p)) \quad (i = 0, 1, \ldots, p-2)$$

lajiteltuna toisen komponentin mukaan, voidaan tehtävä ratkaista polynomiajassa ja -tilassa (indeksitaulu poislukien), mutta esiprosessointi vie nyt ylipolynomiaalisen ajan ja tilan. Eräänlaisen välimuodon tarjoaa

Shanksin pikkuaskel-jättiaskel-algoritmi:

1. Asetetaan $m \leftarrow \lceil \sqrt{p-1}\, \rceil$. Kokonaisneliöjuuri $\lfloor \sqrt{p-1} \rfloor$ on nopeasti laskettavissa ja

$$\lceil \sqrt{p-1} \rceil = \begin{cases} \lfloor \sqrt{p-1} \rfloor, \text{ jos } p-1 \text{ on neliö eli } p-1 = \lfloor \sqrt{p-1} \rfloor^2 \\ \lceil \sqrt{p-1}\, \rceil = \lfloor \sqrt{p-1} \rfloor + 1 \text{ muuten.} \end{cases}$$

2. Lasketaan parit

$$(i, (g^{mi}, \mathrm{mod}\ p)) \quad (i = 0, 1, \ldots, m-1) \quad (\textit{jättiaskeleet})$$

   ja lajitellaan ne toisen komponentin mukaan. Tuloksena on lista $\mathcal{L}_1$. Tähän tarvitaan Venäläisten talonpoikien menetelmä ja jokin *nopea* lajittelualgoritmi.

3. Lasketaan parit

$$(k, (bg^{-k}, \mathrm{mod}\ p)) \quad (k = 0, 1, \ldots, m-1) \quad (\textit{pikkuaskeleet})$$

   ja lajitellaan nekin toisen komponentin mukaan. Näin saadaan lista $\mathcal{L}_2$.

4. Etsitään listasta $\mathcal{L}_1$ pari (i, y) ja listasta $\mathcal{L}_2$ pari (k, z), joissa $y = z$.

5. Tulostetaan $(mi + k, \mathrm{mod}\ p-1)$ ja lopetetaan.

Jos mainitut parit löytyvät, niin saatu tulostus $j = (mi + k, \mathrm{mod}\ p-1)$ on oikea logaritmi, sillä silloin voidaan kirjoittaa $mi + k = t(p-1) + j$ ja

$$g^{mi} \equiv bg^{-k} \mod p \quad \text{eli} \quad b \equiv g^{mi+k} = (g^{p-1})^t g^j \equiv 1 \cdot g^j \equiv g^j \mod p.$$

Toisaalta algoritmi johtaa aina tulokseen, sillä jos $b \equiv g^j \mod p$ ja $0 \leq j \leq p-2$, niin j voidaan kirjoittaa jakolaskulla muotoon $j = mi + k$, missä $0 \leq k < m$, jolloin myös

$$i = \frac{j-k}{m} \leq \frac{j}{m} < \frac{p-1}{m} \leq \frac{p-1}{\sqrt{p-1}} = \sqrt{p-1} \leq m.$$

Pikkuaskel-jättiaskel-algoritmi voidaan implementoida (noin) aikaan $O(m)$ ja tilaan $O(m)$.

Muitakin algoritmeja diskreetin logaritmin laskemiseksi $\mathbb{Z}_p^*$:ssä on, mm. ns. *Pohlig–Hellman-algoritmi* ja ns. *Indeksilaskumenetelmä,* ks. esimerkiksi STINSON tai SALOMAA. Pohlig–Hellman-algoritmi on kohtuullisen nopea, mikäli $p-1$:llä on vain pieniä alkutekijöita. Kaikki nämä algoritmit voidaan myös yleistää $\mathbb{F}_{p^n}^*$:n diskreetin logaritmin laskuun, joka myös on hyvin työlästä.

8.3 Elliptiset käyrät

Geometrisesti *elliptisellä*[2] *käyrällä* tarkoitetaan 3. asteen käyrää, jonka yhtälö on

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

(Huomaa erikoinen kerrointen indeksointi, joka on perinteinen.) Lisäehtona on, että käyrä on sileä, ts. että puolittain osittaisderivoimalla saadut yhtälöt

$$\begin{aligned} a_1y &= 3x^2 + 2a_2x + a_4 \\ 2y + a_1x + a_3 &= 0 \end{aligned}$$

eivät toteudu käyrällä yhtaikaa. Geometrisesti tämä takaa, että käyrällä on tangentti jokaisessa pisteessä. Peruskursseilta tutun implisiittisen derivoinnin tuloksena nimittäin

$$y' = \frac{3x^2 + 2a_2x + a_4 - a_1y}{2y + a_1x + a_3}.$$

Kun sallitaan vaaka- ja pystysuuntaiset tangentit, niin ainoa tilanne, jossa tangenttia ei (välttämättä) ole, on se, että osoittaja ja nimittäjä ovat yhtaikaa nollia.

Alunperin elliptinen käyrä on tietysti reaalinen eli $\mathbb{R}^2$:ssa. Käyrää voidaan ajatella missä tahansa kunnassa $\mathbb{F}$ (ns. *kerroinkunta*), josta myös kertoimet silloin tulevat. Käyrä on silloin kaikkien niiden pisteparien (x, y) joukko, jotka toteuttavat yhtälön. Vaikka sileysehdolla ei ole tällöin (välttämättä) yhtään mitään ”geometrista” merkitystä, osoittautuu, että se on kuitenkin tärkeä. Merkitään jatkossa yksinkertaisuuden vuoksi kunnan $\mathbb{F}$ yhteen- ja vähennyslaskuja tuttuun tapaan $+$:lla ja $-$:lla.

Varsin yleisesti voidaan rajoittua hieman yksinkertaisempiin elliptisiin käyriin, nimittäin muotoa

$$y^2 = x^3 + ax + b$$

[2]Nimi tulee siitä, että eräät algebralliset funtiot $y = f(x)$, jotka liittyvät ellipsin kaarenpituuksien laskuun integroimalla, toteuttavat tällaisen 3. asteen yhtälön.

oleviin käyriin, missä yhtälöt

$$\begin{aligned} \mathbb{0} &= 3x^2 + a \\ 2y &= \mathbb{0} \end{aligned}$$

eivät toteudu käyrällä yhtaikaa (sileysehto). Eliminoimalla yhtälöistä

$$\begin{aligned} y^2 &= x^3 + ax + b \\ \mathbb{0} &= 3x^2 + a \\ 2y &= \mathbb{0} \end{aligned}$$

x ja y (ei ole kovin vaikeaa) nähdään, että tämä vastaa ehtoa

$$4a^3 + 27b^2 \neq \mathbb{0}.$$

Tällaisen käyrän erikoisuus on se, että se on symmetrinen x-akselin suhteen, ts. jos piste (x, y) on käyrällä, niin samoin on $(x, -y)$.

Poikkeuksia ovat kunnat, joissa aina $2z = \mathbb{0}$ (esimerkiksi kunnat $\mathbb{F}_{2^n}$) tai aina $3z = \mathbb{0}$ (esimerkiksi $\mathbb{F}_{3^n}$), kun z on kunnan alkio. Edellisissä muodot ovat

$$y^2 + ay = x^3 + bx + c \quad (\textit{ylisingulääärinen} \text{ tapaus})$$

ja

$$y^2 + xy = x^3 + ax^2 + b \quad (\textit{ei-ylisingulääärinen} \text{ tapaus})$$

ja jälkimmäisessä taas

$$y^2 = x^3 + ax^2 + bx + c.$$

(Lisäksi tarvitaan vastaavat sileysehdot.) Vaikkakin esimerkiksi kunnat $\mathbb{F}_{2^n}$ ovat hyvin tärkeitä kryptografiassa, jatkossa rajoitutaan yksinkertaisuuden vuoksi vain kuntiin, joissa em. muoto $y^2 = x^3 + ax + b$ (missä $4a^3 + 27b^2 \neq \mathbb{0}$) on mahdollinen.

Kauan on geometrisin perustein tiedetty, että reaaliselle elliptiselle käyrälle (eli sen pisteille) voidaan määritellä laskuoperaatio, joka tekee siitä vaihdannaisen ryhmän. Vastaava määritelmä voidaan tehdä myös muissa kunnissa, jolloin myös saadaan vaihdannainen ryhmä. Näitä ryhmiä kutsutaan yksinkertaisesti vain *elliptisiksi käyriksi*. Koska elliptisiä käyriä on hyvin paljon, saadaan tätä kautta runsaasti otollisia syklisiä aliryhmiä diskreettiin logaritmiin pohjautuvia kryptosysteemejä ajatellen.

Tarkastellaan ryhmäoperaatiota ensin havainnollisuuden vuoksi $\mathbb{R}^2$:ssa. Ryhmän identiteettialkio on jossain määrin keinotekoinen, se on y-akselin suunnassa äärettömyydessä oleva ”piste” O. Positiivinen ja negatiivinen äärettömyys samaistetaan. Kaikkien y-akselin suuntaisten suorien sovitaan leikkaavan tässä pisteessä O. Geometrisesti pisteiden P ja Q ryhmäoperaatio tuottaa pisteen $R = P \oplus Q$ sekä vastapisteen $-P$ seuraavan säännön mukaan:

1. Piirretään suora pisteiden P ja Q kautta. Jos $P = Q$, tämä suora on tangentti pisteessä P (sileys takaa, että tangentti on olemassa).

2. Mikäli piirretty suora on y-akselin suuntainen, on $R = O$.

3. Muutoin R on suoran ja käyrän leikkauspisteen peilikuva x-akselin suhteen. On mahdollista, että suora sivuaa käyrää P:ssä (eli leikkauspiste yhtyy P:hen), jolloin R on P:n peilikuva, tai Q:ssa (leikkauspiste yhtyy Q:hun), jolloin R on Q:n peilikuva.

4. $-P$ on P:n peilikuva x-akselin suhteen. Erityisesti $-O = O$.

Ilmeisesti operaatio $\oplus$ on vaihdannainen. Sääntöä sopivasti tulkiten nähdään välittömästi vielä, että $P \oplus O = O \oplus P = P$ (erityisesti $O \oplus O = O$) ja että $P \oplus -P = -P \oplus P = O$, kuten pitääkin.

Esimerkki. *Alla on Maple-ohjelmiston piirtämä kuva elliptisestä käyrästä*

$$y^2 = x^3 - 5x + 1.$$

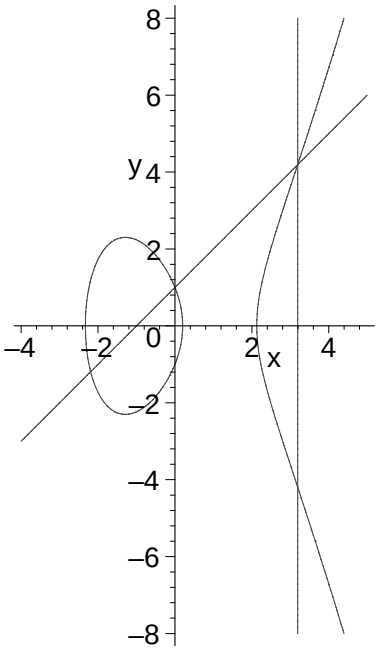
Kuvassa on myös laskettu käyrän pisteiden

$$P = ((1 - \sqrt{29})/2, (3 - \sqrt{29})/2) \quad \textit{ja} \quad Q = (0, 1)$$

ryhmäoperaatio. Tulos on

$$R = ((1 + \sqrt{29}/2, -(3 + \sqrt{29})/2).$$

Huomaa, miten käyrä on kahdessa osassa, joista toinen on suljettu ja toinen ääretön. Kaikki elliptiset käyrät eivät ole tällä tavoin kaksiosaisia.



Lasketaan operaation $P \oplus Q = R$ tulos yleisesti. Tapaukset $P = O$ ja/tai $Q = O$ ovat helpot. Jos pisteet ovat $P = (x_1, y_1)$ sekä $Q = (x_2, y_2)$, $P \neq Q$ ja $x_1 = x_2$, niin ilmeisesti $y_1 = -y_2$, joten $R = O$ eli $P = -Q$. Näinpä voidaan edelleen rajoittua tapauksiin, joissa joko $x_1 \neq x_2$ tai $P = Q$. Katsotaan ensin edellinen tapaus. Silloin P:n ja Q:n kautta kulkevan suoran parametriesitys on

$$\begin{cases} x = x_1 + (x_2 - x_1)t \\ y = y_1 + (y_2 - y_1)t. \end{cases}$$

Sijoitetaan tämä elliptisen käyrän yhtälöön $y^2 - x^3 - ax - b = 0$:

$$(y_1 + (y_2 - y_1)t)^2 - (x_1 + (x_2 - x_1)t)^3 - a(x_1 + (x_2 - x_1)t) - b = 0.$$

Vasen puoli on t:n 3. asteen polynomi $p(t)$. Jaetaan se polynomilla $t^2 - t$. Tulos on muotoa

$$p(t) = q(t)(t^2 - t) + r(t),$$

missä osamäärä $q(t)$ on 1. astetta ja $r(t) = \alpha t + \beta$ on jakojäännös. Arvo $t = 0$ vastaa pistettä P ja arvo $t = 1$ pistettä Q, jotka ovat käyrällä, ts. $p(0) = p(1) = 0$. Siispä myös $r(0) = r(1) = 0$ eli $\alpha = 0$ ja $\alpha + \beta = 0$, mutta tästä seuraa heti, että $r(x)$ on nollapolynomi ja $p(t) = q(t)(t^2 - t)$. Edelleen yhtälöstä $q(t) = 0$ saadaan se t:n arvo t_3, joka määrää kolmannen leikkauspisteen (x_3, y_3). Jakolaskutoimitus osoittaa, että

$$q(t) = (y_2 - y_1)^2 - 3x_1(x_2 - x_1)^2 - (x_2 - x_1)^3(t + 1)$$

ja näin

$$t_3 = \frac{(y_2 - y_1)^2}{(x_2 - x_1)^3} - \frac{2x_1 + x_2}{x_2 - x_1}.$$

Sijoittamalla tämä suoran parametriesitykseen saadaan

$$\begin{cases} x_3 = \lambda^2 - x_1 - x_2 \\ y_3 = \lambda(x_3 - x_1) + y_1, \end{cases}$$

missä

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

(piirretyn suoran kulmakerroin), ja lopulta

$$P \oplus Q = R = (x_3, -y_3).$$

Tässä voi olla, että $(x_3, y_3) = P$ tai $(x_3, y_3) = Q$. Huomaa, että on (x_3, y_3) aina määritelty.

Katsotaan vielä tapaus, jossa $P = Q = (x_1, y_1)$, ja lasketaan

$$P \oplus P = 2P = R.$$

Jos $y_1 = 0$, on käyrän tangentti ilmeisesti y-akselin suuntainen ja $R = O$ eli $-P = P$. Näin voidaan siirtyä tapaukseen $y_1 \neq 0$. Tangentin kulmakerroin on

$$y' = \frac{3x^2 + a}{2y}.$$

Pisteeseen P piirretyn tangenttisuoran parametriesitys on näin ollen

$$\begin{cases} x = x_1 + 2y_1 t \\ y = y_1 + (3x_1^2 + a)t. \end{cases}$$

Hyvin samanlaisen laskun tuloksena kuin edellä saadaan leikkauspistettä (x_2, y_2) vastaavalle parametrin t arvolle t_2 yhtälö

$$x_1 + 2y_1 t_2 = \frac{(3x_1 + a)^3}{4y_1^2} - 2x_1$$

ja edelleen

$$\begin{cases} x_2 = \lambda^2 - 2x_1 \\ y_2 = \lambda(x_2 - x_1) + y_1, \end{cases}$$

missä

$$\lambda = \frac{3x_1^2 + a}{2y_1}$$

(piirretyn suoran kulmakerroin), ja lopulta

$$2P = R = (x_2, -y_2).$$

Jälleen voi olla, että $P = (x_2, y_2)$. Myös nyt (x_2, y_2) on aina määritelty.

Nämä laskukaavat pätevät missä tahansa kunnassa, jossa elliptinen käyrä voidaan kirjoittaa muotoon $y^2 = x^3 + ax + b$ (missä $4a^3 + 27b^2 \neq 0$). Muissa kunnissa tarvitaan hieman erilaiset kaavat, ks. KOBLITZ.

Kaiken kaikkiaan todetaan, että vasta-alkion muodostaminen on helppoa (peilaus), ryhmäoperaatio $\oplus$ on vaihdannainen ja varsin helposti laskettavissa. Operaation liitännäisyys on kuitenkin vähän vaikea todistaa yo. kaavoista lähtien. Oikea maailma elliptisten käyrien ominaisuuksia ajatellen on ns. *projektiivinen geometria,* jossa myös itse ryhmäoperaatio luonnostaan tulee esille. Liitännäisyys $\mathbb{R}^2$:ssa seuraakin melko suoraan klassisista kolmannen asteen käyrien projektiivisen geometrian tuloksista. Seuraavassa vanhasta alan suomalaisesta klassikosta[3] tulos, josta liitännäisyys seuraa helposti:

> *"Jos kaksi suoraa a ja b leikkaa kolmannen asteen käyrän vastaavasti pisteissä $A_1, A_2, A_3; B_1, B_2, B_3$, ovat suorien A_1B_1, A_2B_2, A_3B_3 ja käyrän kolmannet leikkauspisteet C_1, C_2, C_3 keskenään samalla suoralla."*

Muissa kunnissa liitännäisyys pitää todistaa erikseen ja se on varsin työläs tehtävä. Huomaa, että muissa kunnissa myös vaihdannaisuus pitää todistaa erikseen, mutta tämä on aika helppoa. Molemmat ovat symbolisia identiteettejä, joten ne voidaan todentaa symbolisesti. Tehdään se Maple-ohjelmistolla. Ilmeisesti tapaukset, joissa ainakin yksi alkioista on O, ovat triviaaleja, joten ne voidaan jättää pois.

Aloitetaan vaihdannaisuudesta. Ensin määritellään ryhmäoperaatio

```
> eco:=proc(u,v)
  local lambda,xx,yy;
  lambda:=(v[2]-u[2])/(v[1]-u[1]);
  xx:=lambda^2-u[1]-v[1];
  yy:=lambda*(xx-u[1])+u[2];
  [xx,-yy];
  end:
```

ja sitten testataan

```
> A:=eco([x[1],y[1]],[x[2],y[2]]);
```

$$[\frac{(y_2-y_1)^2}{(x_2-x_1)^2} - x_1 - x_2, -(y_2-y_1)\left(\frac{(y_2-y_1)^2}{(x_2-x_1)^2} - 2\,x_1 - x_2\right)(x_2-x_1)^{-1} - y_1]$$

```
> B:=eco([x[2],y[2]],[x[1],y[1]]);
```

$$[\frac{(y_1-y_2)^2}{(x_1-x_2)^2} - x_2 - x_1, -(y_1-y_2)\left(\frac{(y_1-y_2)^2}{(x_1-x_2)^2} - 2\,x_2 - x_1\right)(x_1-x_2)^{-1} - y_2]$$

```
> normal(A-B);
```

$$[0, 0]$$

Todennetaan edelleen liitännäisyys tapauksessa, jossa ei ole mukana kahdentamisia.

```
> A:=eco([x[1],y[1]],eco([x[2],y[2]],[x[3],y[3]])):
> B:=eco(eco([x[1],y[1]],[x[2],y[2]]),[x[3],y[3]]):
> C:=numer(normal(A-B)):
> max(degree(C[1],y[1]),degree(C[1],y[2]),degree(C[1],y[3]),
      degree(C[2],y[1]),degree(C[2],y[2]),degree(C[2],y[3]));
```

[3] NYSTRÖM, E.J.: *Korkeamman geometrian alkeet sovellutuksineen.* Otava (1948).

$$11$$

Sijoitetaan käyrän määrittelevät yhtälöt:

```
> yhtalot:={seq(y[1]^(2*i)=(x[1]^3+a*x[1]+b)^i,i=1..5),
            seq(y[2]^(2*i)=(x[2]^3+a*x[2]+b)^i,i=1..5),
            seq(y[3]^(2*i)=(x[3]^3+a*x[3]+b)^i,i=1..5),
            seq(y[1]^(2*i+1)=y[1]*(x[1]^3+a*x[1]+b)^i,i=1..5),
            seq(y[2]^(2*i+1)=y[2]*(x[2]^3+a*x[2]+b)^i,i=1..5),
            seq(y[3]^(2*i+1)=y[3]*(x[3]^3+a*x[3]+b)^i,i=1..5)}:
> normal(subs(yhtalot,C));
```

$$[0, 0]$$

Termien lukumäärät kasvavat varsin suuriksi:

```
> nops(C[1]),nops(C[2]);
```

$$1082, 6448$$

Tarkistus käsipelissä olisi siis melkoinen työ, liitännäisyys voidaan tosin todistaa matemaattisesti. Katsotaan vielä liitännäisyys tapauksessa, jossa on mukana kahdennus. Riittää tarkistaa vain tapaus

$$P \oplus (Q \oplus Q) = (P \oplus Q) \oplus Q,$$

muut seuraavat tästä vaihdannaisuuden vuoksi. Määritellään ensin ko. kahdennus.

```
> ecs:=proc(u)
  local lambda,xx,yy;
  lambda:=(3*u[1]^2+a)/2/u[2];
  xx:=lambda^2-2*u[1];
  yy:=lambda*(xx-u[1])+u[2];
  [xx,-yy];
  end:
> A:=eco([x[1],y[1]],ecs([x[2],y[2]])):
> B:=eco(eco([x[1],y[1]],[x[2],y[2]]),[x[2],y[2]]):
> C:=numer(normal(A-B)):
> max(degree(C[1],y[1]),degree(C[1],y[2]),
      degree(C[2],y[1]),degree(C[2],y[2]));
```

$$15$$

Sijoitetaan käyrän määrittelevät yhtälöt:

```
> yhtalot:={seq(y[1]^(2*i)=(x[1]^3+a*x[1]+b)^i,i=1..7),
            seq(y[2]^(2*i)=(x[2]^3+a*x[2]+b)^i,i=1..7),
            seq(y[1]^(2*i+1)=y[1]*(x[1]^3+a*x[1]+b)^i,i=1..7),
            seq(y[2]^(2*i+1)=y[2]*(x[2]^3+a*x[2]+b)^i,i=1..7)};
> normal(subs(yhtalot,C));
```

$$[0, 0]$$

Elliptiset käyrät ovat ryhminä hyvin vaihtelevia. Vaihdannaisten ryhmien kantalause kertoo kuitenkin, että ne ovat jäännösluokkaryhmien suoria tuloja. Jos kerroinkunta on äärellinen, saadaan tarkempikin tulos:

Lause 8.2. (Casselsin lause) *Elliptinen käyrä yli äärellisen kunnan $\mathbb{F}_q$ on joko syklinen tai rakenteeltaan identtinen (eli isomorfinen) sellaisen kahden jäännösluokkaryhmän suoran tulon $\mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2}$ kanssa, että $n_1 \mid n_2, q - 1$.*

Ryhmän kokoa ajatellen tiedetään, että

Lause 8.3. (Hassen lause) *Jos elliptisessä käyrässä yli äärellisen kunnan $\mathbb{F}_q$ on N alkiota, niin*

$$q + 1 - 2\sqrt{q} \leq N \leq q + 1 + 2\sqrt{q}.$$

(Näiden lauseiden todistukset ovat jo varsin vaativaa algebrallista lukuteoriaa.[4]) Näin ollen elliptisessä käyrässä yli $\mathbb{F}_q$:n on suurin piirtein yhtä monta alkiota kuin $\mathbb{F}_q$:ssä. Tarkankin alkiomäärän laskemiseksi tunnetaan melko tehokkaita algoritmeja, ns. *Schoofin algoritmi*[5] ja sen seurannaiset, ks. KOBLITZ.

Ei ole helppoa löytää edes yhtä näistä monista alkioista. Itse asiassa ei tunneta polynomiaikaista determinististä algoritmia elliptisen käyrän alkioiden generoimiseksi äärellisissä kunnissa. Seuraava Las Vegas -tyyppinen algoritmi tuottaa käyrän alkion (positiivisessa jäännössysteemissä esitettynä) alkukunnassa $\mathbb{Z}_p$, missä $p > 3$:

1. Valitaan satunnainen luku x väliltä $0 \leq x < p$ ja asetetaan

   $$z \leftarrow (x^3 + ax + b, \bmod p).$$

   Hassen lauseen nojalla tämä tuottaa neliönjäännöksen z noin 50% todennäköisyydellä, sillä kustakin z:sta saadaan kaksi y:n arvoa (ellei $z = 0$).

2. Jos $z = 0$, tulostetaan $(x, 0)$ ja lopetetaan.

3. Jos $z^{(p-1)/2} \not\equiv 1 \mod p$, tulostetaan ”FAIL” ja lopetetaan. Eulerin kriteerin nojalla z on tällöin epäneliönjäännös modulo p.

4. Lasketaan Shanksin algoritmia käyttäen z:n neliöjuuret y_1 ja y_2 modulo p, tulostetaan (x, y_1) ja (x, y_2) ja lopetetaan.

Algoritmi on ilmeisesti polynomiaikainen ja se tuottaa tuloksen noin 25% todennäköisyydellä. (Muista, että Shanksin algoritmi tuottaa tuloksen noin 50% todennäköisyydellä.)

Huomautus. *Satunnaisesti etsimällä voidaan nyt löytää esimerkiksi sellainen elliptisen käyrän alkio $P \neq O$ ja (suuri) alkuluku r, että $rP = O$, jolloin P:n kertaluku on r (P:n kertaluvunhan on joka tapauksessa jaettava r). Syklinen aliryhmä $\langle P \rangle$ on silloin otollinen kryptografiaa ajatellen. Toinen (hidas) tapa on valita satunnainen alkio P ja testata sen kertaluku, jonka pitäisi tietysti olla suuri. Tähän käy eräs Pikkuaskel-jättiaskel-algoritmin versio. Asia on kuitenkin varsin monimutkainen eikä elliptisten käyrien käyttö kryptografiassa ole kovinkaan suoraviivaista. Ks. esimerkiksi* ROSING *tai* BLAKE & SEROUSSI & SMART.

Hyviä yleisiä viitteitä ovat KOBLITZ ja mm. SILVERMAN & TATE tai COHEN tai CRANDALL & POMERANCE.

[4] Ks. vaikkapa IRELAND, K. & ROSEN, M.: *A Classical Introduction to Modern Number Theory.* Springer–Verlag (1990), ks. myös CRANDALL & POMERANCE.

[5] Alkuperäisviite on SCHOOF, R.: Elliptic Curves over Finite Fields and the Computation of Square Roots mod p. *Mathematics of Computation* **44** (1985), 483–494. Algoritmi on hankala ja myös vaikea implementoida nopeaksi.

Luku 9

DISKREETTIIN LOGARITMIIN PERUSTUVIA KRYPTOSYSTEEMEJÄ

9.1 ElGamalin kryptosysteemi

ElGamalin kryptosysteemi[1] ELGAMAL voidaan pohjata mihin tahansa äärelliseen ryhmään $G = (A, \odot, 1)$, jonka syklisille aliryhmille $\langle a \rangle$ logaritmi on vaikea laskea. Tällaisia ryhmiä ovat esimerkiksi $\mathbb{Z}_p^*$ ja yleisemmin $\mathbb{F}_{p^n}^*$, erityisesti $\mathbb{F}_{2^n}^*$, sekä elliptiset käyrät yli äärellisten kuntien.

Julkinen avain on kolmikko

$$k_1 = (G, a, b),$$

missä $b = a^y$. Salainen avain on $k_2 = y$. Kryptaus on epädeterminististä, sitä varten valitaan satunnaisesti luku x väliltä $0 \leq x < l$, missä l on a:n kertaluku. (Jos l:ää ei haluta julkistaa tai sitä ei tiedetä, voidaan antaa suurempikin yläraja, esimerkiksi G:n alkioluku.) Kryptausfunktio on

$$e_{k_1}(w, x) = (a^x, w \odot b^x) = (c_1, c_2).$$

Viestilohko on siis tulkittava G:n alkioksi. Dekryptausfunktio on

$$d_{k_2}(c_1, c_2) = c_2 \odot c_1^{-y}.$$

Dekryptaus toimii, sillä

$$d_y(a^x, w \odot b^x) = w \odot b^x \odot (a^x)^{-y} = w \odot a^{xy} \odot a^{-xy} = w.$$

Ideana on ”peittää” w kertomalla se b^x:llä, x taas välitetään a^x:n kautta.

Jotta ELGAMAL voitaisiin pystyttää alkukunnan multiplikatiivisessa ryhmässä $\mathbb{Z}_p^*$, valitaan yhtaikaa sekä p että primitiivinen juuri a modulo p. Lisäksi on muistettava, että $p - 1$:llä tulisi olla suuria alkutekijöitä, jotta esimerkiksi Pohlig–Hellman-algoritmilla ei saataisi diskreettiä logaritmia lasketuksi nopeasti. Tämä käy seuraavasti:

1. Valitaan satunnainen suuri alkuluku q sekä pienempi satunnainen luku r, joka voidaan jakaa tekijöihin.

2. Jos $2qr + 1$ on alkuluku, asetetaan $p \leftarrow 2qr + 1$. (Huomaa, että $p - 1$:llä on silloin suuri alkutekijä q.) Muutoin mennään kohtaan 1.

[1] Alkuperäisviite on ELGAMAL, T.: A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory* **IT–31** (1985), 469–472. Tässä systeemissä tarkasteltiin $\mathbb{Z}_p^*$:n logaritmeja.

3. Valitaan satunnaisesti luku a väliltä $1 \leq a < p$.

4. Testataan Lucas'n kriteerillä onko a primitiivinen juuri modulo p vai ei. Tähän tarvittavat $p-1$:n alkutekijät, eli 2, q sekä r:n alkutekijät, ovat nyt helposti saatavilla.

5. Jos a on primitiivinen juuri modulo p, tulostetaan p ja a ja lopetetaan. Muutoin mennään kohtaan 3.

9.2 Diffie–Hellman-avainjakosysteemi

ELGAMAL sallii useiden osapuolien julkaista julkisen avaimensa saman systeemin puitteissa: Kukin valitsee vain oman y:nsä ja julkaisee vastaavan a^y:n. ELGAMAL onkin itse asiassa myöhempi modifikaatio eräästä vanhimmista julkisen avaimen systeemeistä, *Diffie–Hellman-avainjakosysteemistä* DIFFIE–HELLMAN.

Alkuasetelma on tässä sama kuin ELGAMALissa. Kukin osapuoli i valitsee jälleen satunnaisen luvun x_i väliltä $0 \leq x_i < l$ (tai joltain laajemmalta väliltä) ja julkaisee a^{x_i}:n. Osapuolten i ja j välinen yhteinen avain on silloin $a^{x_i x_j}$, jonka kumpikin voi laskea nopeasti julkaistusta tiedosta ja omasta salaisesta luvustaan.

Ilmeisesti DIFFIE–HELLMANin murtaminen on ekvivalentti seuraavan tehtävän ratkaisemisen kanssa:

DHP: Kun annetaan (G, a, b, c), laske $b^{\log_a c}$ (eli $c^{\log_a b}$).

Tämä tehtävä on ns. *Diffie–Hellman-probleema.* Diffie–Hellman-probleeman kompleksisuutta ei tiedetä, diskreetin logaritmin laskeminen luonnollisesti ratkaisee myös sen. Myös ELGAMALin dekryptaus on ekvivalentti Diffie–Hellman-probleeman kanssa, kuten on helppo todeta.

Uudempi hyvin nopea variantti Diffie–Hellman-tyyppisestä kryptosysteemistä saadaan eräiden äärellisien kuntien yksikköryhmissä, ns. *XTR-systeemi,* ks. LENSTRA, A.K. & VERHEUL, E.R.: The XTR Public Key System. *CRYPTO 2000. Lecture Notes in Computer Science* **1880**, Springer–Verlag (2000), 1–19.

9.3 Elliptisiin käyriin perustuvat kryptosysteemit

Elliptisen käyrän syklistä aliryhmää käyttäen voidaan pystyttää ElGamalin kryptosysteemi. Tällöin luonnollisesti ko. syklisessä aliryhmässä diskreetin logaritmin (tai Diffie–Hellman-probleeman) pitää olla vaikeasti ratkaistavissa. Valitettavasti vain eräissä elliptisissä käyrissä (ylisingulääriset elliptiset käyrät) yli äärellisten kuntien nämä probleemat ovat suhteellisen nopeasti ratkaistavissa ns. *Menezes–Okamoto–Vanstone-algoritmilla,* ja näitä pitää välttää, ks. KOBLITZ.[2] Todettakoon, että Pikkuaskel-jättiaskel-algoritmi sopii diskreetin logaritmin laskemiseksi elliptisissä käyrissä, ja samoin Pohlig–Hellman-algoritmi, mutta ne eivät aina ole nopeita.

Toinen vaikeus on se, että kun ELGAMAL äärellisille kunnille noin kaksinkertaistaa viestin pituuden (parikonstruktio), niin ELGAMAL elliptisille käyrille yli äärellisten kuntien noin nelinkertaistaa sen. (Elliptisessä käyrässähän on Hassen lauseen nojalla pisteitä suurin piirtein yhtä monta kuin kunnassa alkioita.) Tämä vältetään käyttämällä tehokkaampaa ELGAMALin

[2]Ikävä piirre on sekin, että juuri käytännön kannalta mukavat "bitteihin perustuvat" äärelliset kunnat $\mathbb{F}_{2^n}$ näyttäisivät olevan huonompia kuin muut. Ks. esimerkiksi GAUDRY, P. & HESS, F. & SMART, N.P.: Constructive and Destructive Facets of Weil Descent on Elliptic Curves. *HP Labs Technical Reports* **HPL–2000–10** (2000). Mitä pidemmälle matemaattisesti sangen vaativassa elliptisten käyrien teoriassa edetään sitä enemmän tällaisia heikkouksia luultavasti paljastuu.

varianttia, ns. *Menezes–Vanstone-systeemiä* MENEZES–VANSTONE. Julkinen avain on kolmikko $k_1 = (E, \alpha, \beta)$, missä E on elliptinen käyrä yli alkukunnan $\mathbb{Z}_p$ (missä $p > 3$), α on generoiva alkio E:n syklisessä aliryhmässä ja $\beta = a\alpha$. Salainen avain on $k_2 = a$. Viestilohko on $\mathbb{Z}_p$:n alkioiden pari (w_1, w_2) positiivisessa jäännössysteemissä esitettynä.

Kryptausfunktio määritellään seuraavasti:

$$e_{k_1}((w_1, w_2), x) = (y_0, y_1, y_2),$$

missä,

$$y_0 = x\alpha \quad , \quad y_1 = (c_1 w_1, \bmod p) \quad , \quad y_2 = (c_2 w_2, \bmod p),$$

x on satunnaisluku (vrt. ELGAMAL) ja luvut c_1 sekä c_2 saadaan, kun esitetään elliptisen käyrän piste $x\beta = (c_1, c_2)$ positiivisessa jäännössysteemissä. (x pitää valita siten, että $c_1, c_2 \not\equiv 0 \mod p$.) Dekryptausfunktio on puolestaan

$$d_{k_2}(y_0, y_1, y_2) = ((y_1 c_2^{-1}, \bmod p), (y_2 c_2^{-1}, \bmod p)).$$

Huomaa, että c_1 ja c_2 saadaan a:n avulla y_0:sta, sillä

$$ay_0 = ax\alpha = x\beta = (c_1, c_2).$$

Ideana on ELGAMALin tapaan käyttää elliptistä käyrää viestin ”peittämiseen”. ELGAMALin tapaan MENEZES–VANSTONE myös vain noin kaksinkertaistaa viestin pituuden, kaksi $\mathbb{Z}_p$:n alkiota kryptautuu neljäksi.

Huomautus. *Elliptisiin käyriin pohjautuvien kryptosysteemien etu esimerkiksi RSAhan verrattuna on se, että nykykäsityksen mukaan tarvittava avaimen koko on huomattavasti pienempi. Mainittakoon vielä Richard Crandallin patentoima ”nopea” elliptisten käyrien kryptosysteemi CRANDALL, joka perustuu erikoisten alkulukujen (ns.* Mersennen alkulukujen*) käyttöön.*

Luku 10

PROTOKOLLIA

10.1 Tiivistefunktiot ja tiivistelmät

Tiivistelmä on vakiopituinen viestiä ”kyllin tarkasti” kuvaava sana. Viesti saa tällöin olla hyvinkin pitkä. Menetelmä, joka antaa tiivistelmän, on *tiivistefunktio*. Koska mahdollisia tiivistelmiä on vähemmän kuin viestejä, tiivistefunktio ei ole injektiivinen, ts. se antaa tietyissä tilanteissa eri viesteille saman tiivistelmän. Tätä kutsutaan *törmäykseksi*. Jotta tiivistefunktio olisi käyttökelpoinen, pitää sen luonnollisesti olla nopeasti laskettavissa viestistä, mutta myös sellainen, ettei vastapuoli voi käyttää törmäyksiä mitenkään tehokkaasti hyväkseen. Tätä silmälläpitäen määritellään erilaisia käsitteitä:

- Tiivistefunktio h on viestille w *heikosti ei-törmäävä*, jos on laskennallisesti vaikeaa löytää toinen viesti w', jolle $h(w) = h(w')$. (Huomaa, ettei tämä ole aivan tarkka määritelmä, ts. ettei tässä ajatella laskennallista vaativuutta.)

- Tiivistefunktio h on *heikosti ei-törmäävä*, jos annetulle viestille w on laskennallisesti vaativaa löytää toinen viesti w', jolle $h(w) = h(w')$.

- Tiivistefunktio h on *vahvasti ei-törmäävä*, jos on laskennallisesti vaikeaa löytää eri viestit w ja w', joille $h(w) = h(w')$. Ts. jos on laskennallisesti vaikeaa löytää viesti w, jolle h *ei* ole heikosti ei-törmäävä. (Jälleen tämä määritelmä ei ole tarkka.)

- Tiivistefunktio h on *yksisuuntainen*, jos annetulle tiivistelmälle t on laskennallisesti vaativaa löytää viesti w, jolle $h(w) = t$.

Huomautus. *Muunkinlaisia nimityksiä käytetään. Heikosti ei-törmäävää tiivistefunktiota kutsutaan myös* toisen alkukuvan estäväksi, *vahvasti ei-törmäävää myös vain* ei-törmääväksi *ja yksisuuntaista myös* alkukuvan estäväksi.

Lause 10.1. *Jos viestiavaruus W ja tiivistelmäavaruus T ovat molemmat äärellisiä ja $\#W \geq 2\#T$, missä $\#$ tarkoittaa joukon alkiolukua, niin vahvasti ei-törmäävä tiivistefunktio h on yksisuuntainen. Ts. algoritmi A, joka kääntää h:n, voidaan muuntaa Las Vegas -tyyppiseksi probabilistiseksi algoritmiksi, joka löytää törmäyksen ainakin todennäköisyydellä $1/2$.*

Todistus. Merkitään M_w:llä niiden viestien joukkoa, joilla on sama tiivistelmä kuin w:llä, ja $\mathcal{D}$:llä kaikkien näiden joukkojen muodostamaa luokkaa. Silloin

$$\#\mathcal{D} = \#T \quad \text{ja} \quad \sum_{D \in \mathcal{D}} \#D = \#W.$$

Seuraava Las Vegas -algoritmi löytää törmäyksen (tai luopuu koko jutusta):

1. Valitaan satunnainen viesti $w \in W$.
2. Lasketaan tiivistelmä $t = h(w)$.
3. Etsitään algoritmia A käyttäen sellainen viesti w', että $h(w') = t$.
4. Jos $w' \neq w$, tulostetaan w ja w' ja lopetetaan. Muutoin tulostetaan ”FAIL” ja lopetetaan.

Pitää vain näyttää, että se toimii ainakin todennäköisyydellä $1/2$:

$$\begin{aligned} P(\text{törmäys löytyy}) &= \sum_{w \in W} \frac{\#M_w - 1}{\#M_w} \frac{1}{\#W} = \frac{1}{\#W} \sum_{D \in \mathcal{D}} \sum_{w \in D} \frac{\#D - 1}{\#D} \\ &= \frac{1}{\#W} \sum_{D \in \mathcal{D}} (\#D - 1) = \frac{1}{\#W} \left(\sum_{D \in \mathcal{D}} \#D - \sum_{D \in \mathcal{D}} 1 \right) = \frac{\#W - \#T}{\#W} \\ &\geq \frac{\#W - \#W/2}{\#W} = \frac{1}{2}. \end{aligned}$$

□

Jos mahdollisia tiivistelmiä on vähän, voidaan kokeilemalla löytää törmäyksiä: Valitaan vain k kpl satunnaisia viestejä $w_1, \ldots, w_k$, lasketaan tiivistelmät $t_i = h(w_i)$ ja testataan löytyiko törmäyksiä. Tätä yksinkertaista menetelmää kutsutaan *syntymäpäivähyökkäykseksi*[1]. Arvioidaan todennäköisyyksiä sille, että syntymäpäivähyökkäys onnistuu. Tällöin voidaan olettaa, että eri tiivistelmät esiintyvät (suurin piirtein) yhtä monta kertaa. (Muussa tapauksessa törmäyksien löytymisen todennäköisyys vain kasvaa.) Todennäköisyys sille, ettei törmäyksiä löydy, on ilmeisesti

$$P = \frac{n(n-1)(n-2)\cdots(n-k+1)}{n^k} = \left(1 - \frac{1}{n}\right)\left(1 - \frac{2}{n}\right)\cdots\left(1 - \frac{k-1}{n}\right),$$

missä n on tiivistelmien lukumäärä. Koska aina $1 - x \leq e^{-x}$ (funktion $e^{-x} + x - 1$ minimiarvo on 0), saadaan edelleen arvio

$$P \leq e^{-\frac{1}{n} - \frac{2}{n} - \cdots - \frac{k-1}{n}} = e^{-\frac{k(k-1)}{2n}}.$$

Näin ollen todennäköisyys sille, että löytyy törmäy(k)s(iä), on

$$Q = 1 - P \geq 1 - e^{-\frac{k(k-1)}{2n}}.$$

Näin saadaan yläarvio k:lle, jos Q on annettu:

$$-\frac{k(k-1)}{2n} \geq \ln(1 - Q)$$

eli

$$k^2 - k + 2n \ln(1 - Q) \leq 0$$

eli

$$k \leq \frac{1}{2}\left(1 + \sqrt{1 - 8n \ln(1 - Q)}\right).$$

[1]Nimi johtuu siitä, että jos joukossa on riittävästi ihmisiä, todennäköisyys sille, että ainakin kahdella on sama syntymäpäivä (vuoden päivä), on suuri. Koska $1.177\sqrt{365} \cong 22.49$, riittää, että joukossa on ainakin 23 ihmistä, jotta ainakin todennäköisyydellä $1/2$ löytyy samoja syntymäpäiviä.

Valitsemalla $Q = 1/2$ todetaan, että törmäys löytyy (ainakin) todennäköisyydellä $1/2$, jos

$$k \geq \frac{1}{2}\left(1 + \sqrt{1 + 8n \ln 2}\right) \simeq \sqrt{2n \ln 2} \simeq 1.177\sqrt{n}.$$

Näin esimerkiksi 40-bittiselle tiivistelmälle syntymäpäivähyökkäys onnistuu ainakin todennäköisyydellä $1/2$, jos k on vähän yli $2^{20} = 1\,048\,576$. Tiivistelmän pituuden pitäisi siis olla merkittävästi enemmän, esimerkiksi SHAssa (Secure Hash Algorithm, ks. esimerkiksi MENEZES & VAN OORSCHOT & VANSTONE) se on 160 bittiä.

Esimerkkinä yksinkertaisesta tiivistefunktiosta otetaan *Chaum–van Heijst–Pfitzmann-tiivistefunktio* h_{CHP}. Tätä varten tarvitaan sellainen alkuluku p, että myös $q = (p-1)/2$ on alkuluku (eli turvallinen alkuluku, ks. pykälä 7.2). Edelleen tarvitaan kaksi erisuurta primitiivistä juurta α ja β modulo p. Lisäksi oletetaan, että diskreetti logaritmi $\log_\alpha \beta$ ei ole helposti laskettavissa. Viesti muodostuu kahdesta välillä $0 \leq w \leq q$ olevasta luvusta w_1 ja w_2 ja

$$h_{\mathrm{CHP}}(w_1, w_2) = (\alpha^{w_1}\beta^{w_2}, \bmod p).$$

Voidaan osoittaa, että yhdenkin h_{CHP}:n törmäyksen löytäminen mahdollistaa diskreetin logaritmin $\log_\alpha \beta$ laskemisen nopeasti (ks. STINSON).

Huomautus. *CHP-tiivistefunktio on liian hidas ollakseen kovin käyttökelpoinen, monet muut tiivistefunktiot ovat paljon nopeampia laskea (MD4, MD5, SHA-1, jne.; ks.* STINSON *tai* MENEZES & VAN OORSCHOT & VANSTONE*). Pulmana on myös Germainin lukujen vaikea löydettävyys.*

10.2 Allekirjoitus

Allekirjoitussysteemillä tarkoitetaan viisikkoa (P, A, K, S, V), missä

- P on äärellinen *viestiavaruus.*
- A on äärellinen *allekirjoitusavaruus.*
- K on äärellinen *avainavaruus.* Kukin *avain* muodostuu parista $(k_{\mathrm{a}}, k_{\mathrm{t}})$, missä k_{a} on salainen *allekirjoitusavain* ja k_{t} on julkinen *todennusavain.*
- Kutakin allekirjoitusavainta k_{a} kohti on *allekirjoitusfunktio* $s_{k_{\mathrm{a}}} \in S$. Viestille w on $s_{k_{\mathrm{a}}}(w) = (w, u)$, missä u on viestin w *allekirjoitus.* S on kaikkien mahdollisten allekirjoitusfunktioiden avaruus.
- Kutakin todennusavainta k_{t} kohti taas on *todennusfunktio* $v_{k_{\mathrm{t}}} \in V$. V on kaikkien mahdollisten todennusfunktioiden avaruus.
- Jokaiselle viestille w ja avaimelle $(k_{\mathrm{a}}, k_{\mathrm{t}})$ on

$$v_{k_{\mathrm{t}}}(w, u) = \begin{cases} \mathrm{TRUE}, \text{ jos } s_{k_{\mathrm{a}}}(w) = (w, u) \\ \mathrm{FALSE} \text{ muuten.} \end{cases}$$

Useat kryptosysteemit ovat välittömästi muunnettavissa allekirjoitussysteemeiksi (ja itse asiassa ovat alunperin olleet allekirjoitussysteemejä!). Esimerkiksi RSAsta (ks. pykälä 7.1) saadaan allekirjoitussysteemi määrittelemällä

$$k_{\mathrm{a}} = (n, b) \quad \text{ja} \quad k_{\mathrm{t}} = (n, a)$$

sekä

$$s_{k_\mathrm{a}}(w) = (w, (w^b, \bmod n)) \quad \text{ja} \quad v_{k_\mathrm{t}}(w,u) = \begin{cases} \text{TRUE, jos } w \equiv u^a \mod n \\ \text{FALSE muuten.} \end{cases}$$

Vaihtoehtoisesti voidaan käyttää (yksisuuntaista) tiivistefunktiota h ja määritellä

$$s_{k_\mathrm{a}}(w) = (w, (h(w)^b, \bmod n)) \quad \text{ja} \quad v_{k_\mathrm{t}}(w,u) = \begin{cases} \text{TRUE, jos } h(w) \equiv u^a \mod n \\ \text{FALSE muuten.} \end{cases}$$

Tästä on etua, koska muuten pari $((x^a,\bmod n), x)$ on aina allekirjoitettu viesti (tosin ”satunnainen” sellainen).

ElGamalin kryptosysteemistä (ks. pykälä 9.1) saadaan allekirjoitussysteemi valitsemalla ryhmäksi $G = \mathbb{Z}_p^*$, missä p on suuri alkuluku, a:ksi primitiivinen juuri modulo p ja $b = (a^y,\bmod p)$. Todennusavain on nyt $k_\mathrm{t} = (p,a,b)$ ja allekirjoitussavain yksinkertaisesti $k_\mathrm{a} = y$. Allekirjoitusfunktio on $s_{k_\mathrm{a}}(w) = (w,c,d)$, missä

$$c = (a^x, \bmod p) \quad \text{ja} \quad d = ((w - yc)x^{-1}, \bmod p-1)$$

ja x on sellainen väliltä $1 \le x < p-1$ valittu satunnaisluku, että $\mathrm{syt}(x, p-1) = 1$ (jolloin x:llä on käänteisluku modulo $p-1$). Tällöin $xd = w - yc + k(p-1)$ jollekin luvulle k. Todennusfunktio puolestaan on

$$v_{k_\mathrm{t}}(w,c,d) = \begin{cases} \text{TRUE, jos } b^c c^d \equiv a^w \mod p \\ \text{FALSE muuten.} \end{cases}$$

Oikean allekirjoituksen todennus onnistuu, sillä Fermat’n pienen lauseen nojalla

$$b^c c^d \equiv a^{yc} a^{xd} = a^{yc+w-yc+k(p-1)} = a^w (a^{p-1})^k \equiv a^w \cdot 1 = a^w \mod p.$$

Allekirjoituksen väärentämiseksi pitäisi pystyä laskemaan c ja d ilman y:tä ja x:ää. Tällöin voidaan todeta seuraavaa:

- Jos valitsee ensin jonkin c:n ja yrittää saada vastaavan d:n, niin pitää laskea $\log_c(a^w b^{-c})$ modulo p. Huomaa, että koska $\mathrm{syt}(x, p-1) = 1$, myös c on primitiivinen juuri modulo p, ks. pykälä 6.1.

- Jos taas valitseekin ensin jonkin d:n ja yrittää sitten löytää vastaavan c:n, niin pitää ratkaista yhtälö
$$b^c c^d \equiv a^w \mod p.$$
Tällaisten yhtälöiden ratkaisemiseksi ei tunneta nopeita algoritmeja.

- Jos yrittää saada lähetetyksi jonkin allekirjoitetun viestin, vaikkapa vain satunnaisen, niin voisi yrittää valita ensin c ja d ja sitten etsiä jonkin sopivan w:n. Mutta tällöin pitäisi laskea $\log_a(b^c c^d)$ modulo p. (Muita tapoja saada aikaan satunnainen allekirjoitettu viesti kuitenkin on! Samoin on mahdollista yhden saadun allekirjoituksen turvin allekirjoittaa joitain muita satunnaisia viestejä.)

Huomautus. *ElGamalin allekirjoituksen modifikaatio DSS (Digital Signature Standard) on yleisemmin käytössä, ks. esimerkiksi* STINSON *tai* MENEZES & VAN OORSCHOT & VANSTONE.

10.3 Identifikaatio

Identifikaation tarkoitus on varmistaa tietyn osapuolen A henkilöllisyys tai identiteetti ja samalla estää muiden osapuolten esiintyminen A:na.

Esimerkkinä tällaisesta menettelystä esitetään tässä ns. *Schnorr-identifikaatio*[2]. Menettely vaatii luotetun osapuolen L, jonka toimiin kaikki luottavat. L valitsee ensin menettelyyn liittyvät eri parametrit seuraavasti:

1. Etsitään samaan tapaan kuin pykälässä 9.1 niin suuri alkuluku p (esimerkiksi $p > 2^{512}$), että diskreetin logaritmin laskeminen $\mathbb{Z}_p^*$:ssä on hyvin vaativaa, $p-1$:n suuri alkutekijä q (esimerkiksi $q > 2^{140}$), jolle q^2 ei ole $p-1$:n tekijä, sekä primitiivinen juuri g modulo p.

2. Valitaan $\mathbb{Z}_p^*$:n kertalukua q oleva alkio α. Kun tunnetaan primitiivinen juuri g modulo p, voidaan valita $\alpha = (g^{(p-1)/q}, \mathrm{mod}\ p)$, mutta muitakin mahdollisuuksia voi olla.

3. Valitaan ns. *turvaparametri* t, jolle $q > 2^t$ (esimerkiksi $t = 40$).

4. Valitaan allekirjoitusmenetelmä. Siihen liittyy (salainen) allekirjoitusfunktio s_{L} ja julkinen todennusfunktio v_{L}.

5. Vielä valitaan (yksisuuntainen) tiivistefunktio h_{L}.

Kaikki allekirjoitettava informaatio tiivistetään h_{L}:llä. Jatkossa jätetään tämä vaihe pois selvyyden vuoksi. Parametrit p, q ja α sekä v_{L} ja h_{L} ovat julkisia.

L antaa A:lle ns. *sertifikaatin* (eli *varmenteen*) seuraavasti:

Schnorr-sertifiointi:

1. L tarkistaa A:n henkilöllisyyden (tai identiteetin) ”perinteisillä” menetelmillä. Tämä esitetään sanana $\mathrm{id}(\mathrm{A})$.

2. A valitsee salaisen satunnaisen luvun a väliltä $0 \leq a < q$, laskee luvun

$$v = (\alpha^{-a}, \mathrm{mod}\ p)$$

   ja lähettää sen L:lle.

3. L generoi allekirjoituksen

$$S = s_{\mathrm{L}}(\mathrm{id}(\mathrm{A}), v)$$

   ja antaa A:lle sertifikaatin

$$\mathrm{cer}(\mathrm{A}) = (\mathrm{id}(\mathrm{A}), v, S).$$

A voi nyt todistaa B:lle henkilöllisyytensä (tai identiteettinsä) seuraavasti:

Schnorr-identifikaatio:

1. A valitsee satunnaisen luvun k väliltä $0 \leq k < q$ ja laskee luvun

$$\gamma = (\alpha^k, \mathrm{mod}\ p).$$

[2] Alkuperäisviite on SCHNORR, C.P.: Efficient Signature Generation by Smart Cards. *Journal of Cryptology* **4** (1991), 161–174.

2. A lähettää B:lle sertifikaattinsa $\mathrm{cer}(\mathrm{A})$:n ja γ:n.
3. B todentaa sertifikaatissa olevan allekirjoituksen tarkistamalla, että $v_{\mathrm{L}}(\mathrm{cer}(\mathrm{A})) = \mathrm{TRUE}$.
4. B valitsee satunnaisen luvun r väliltä $1 \leq r \leq 2^t$ ja lähettää sen A:lle. Tämä r on ns. *haaste*.
5. A laskee luvun $y = (k + ar, \mathrm{mod}\ q)$ ja lähettää sen B:lle.
6. B tarkistaa, että $\gamma = (\alpha^y v^r, \mathrm{mod}\ p)$.

Oikea identiteetti tulee oikein tarkistetuksi, sillä y on muotoa $y = k + ar + lq$ ja

$$\alpha^y v^r = \alpha^{k+ar+lq} v^r = \alpha^k (\alpha^q)^l \alpha^{ar} v^r \equiv \alpha^k \cdot 1 \cdot \alpha^{ar} \alpha^{-ar} \equiv \alpha^k \mod p.$$

(Muista, että α:n kertaluku modulo p on q.)

Ulkopuolinen (vihamielinen) taho U ei voi esiintyä A:na, ainakaan mitenkään helposti. U voisi esimerkiksi väärentää A:n sertifikaatin muodossa

$$\mathrm{cer}'(\mathrm{A}) = (\mathrm{id}(\mathrm{A}), v', S')$$

(missä $v' \neq v$). Koska käytetyn allekirjoitussysteemin oletetaan luonnollisesti olevan turvallinen, ei U kuitenkaan voi väärentää allekirjoitusta S ja B havaitsee väärän allekirjoituksen S'. Toinen tapa olisi, että U käyttää A:n oikeaa sertifikaattia. (Sertifikaattihan paljastuu aina, kun A sitä käyttää.) Mutta U ei voi kuitenkaan esiintyä A:na, sillä hän ei tiedä a:ta eikä voi näin ollen vastata oikein B:n haasteeseen, koska a:n laskeminen v:stä vaatii diskreetin logaritmin.

Lause 10.2. *Jos U tietää arvon γ, jota käyttäen hän pystyy esiintymään A:na ainakin todennäköisyydellä $1/2^{t-1}$, niin U pystyy myös laskemaan a:n polynomiajassa.*

Todistus. B:n mahdollisia haasteita on 2^t kappaletta, joten U pystyy vastaamaan ainakin kahteen ($2^t/2^{t-1} = 2$) haasteeseen r_1 ja r_2 oikein, ts. hän tietää eri luvut y_1 ja y_2, joille

$$\gamma \equiv \alpha^{y_1} v^{r_1} \equiv \alpha^{y_2} v^{r_2} \mod p$$

(muutenhan U ei pystyisi vastaamaan haasteisiin). Koska $v = (\alpha^{-a}, \mathrm{mod}\ p)$, on edelleen

$$\alpha^{y_1 - y_2 - a(r_1 - r_2)} \equiv 1 \mod p.$$

α:n kertaluku modulo p on q, joten $q \mid y_1 - y_2 - a(r_1 - r_2)$ eli

$$y_1 - y_2 \equiv a(r_1 - r_2) \mod q.$$

Koska $0 < |r_1 - r_2| < 2^t < q$ ja q on alkuluku, on $\mathrm{syt}(r_1 - r_2, q) = 1$. U voi näin laskea a:n:

$$a = ((y_1 - y_2)(r_1 - r_2)^{-1}, \mathrm{mod}\ q).$$

□

Huomautus. *Tämä lausekaan ei takaa, että Schnorr-identifikaatio on turvallinen, ts. että ulkopuolinen taho U ei, pyydettyään A:lta identifikaatiota polynomiaalisen määrän kertoja, pystyisi polynomiajassa laskemaan a:ta. Itse asiassa ei ole voitu todistaa, että se olisi turvallinen (mutta ei päinvastoinkaan!). Sen sijaan vähän mutkikkaampi variantti, ns.* Okamoto-identifikaatio, *on todistettavissa turvalliseksi, ks.* STINSON.

10.4 Arpominen

Satunnaisbitin arpominen on helppoa, jos on luotettu osapuoli, joka suorittaa arpomisen. Jos tällaista osapuolta ei ole, on arpominen silti mahdollista sopivaa protokollaa käyttäen.

Seuraavissa protokollissa[3] A arpoo B:lle satunnaisbitin. Aluksi vain B tietää tuloksen, mutta hän voi kertoa sen A:lle. Vaikka B ei kertoisi tulosta A:lle, hän ei silti voi muuttaa saamaansa bittiä (tai ilmoittaa A:lle väärää bittiä) ilman, että se jossain vaiheessa paljastuu A:lle. Näin B *sitoutuu bittiin,* jonka hän sai.

Ensimmäinen protokolla toimii seuraavasti (ks. pykälät 6.6–7):

1. A valitsee kaksi erisuurta isoa alkulukua p ja q ja lähettää B:lle niiden tulon $n = pq$.

2. B valitsee satunnaisen luvun u väliltä $1 < u < n/2$ ja lähettää A:lle neliön
$$z = (u^2, \bmod n).$$

3. A laskee ne neljä z:n neliöjuurta modulo n:
$$(\pm x, \bmod n) \quad \text{ja} \quad (\pm y, \bmod n).$$
Tämä on mahdollista, koska A tietää n:n tekijät. Valitaan x':ksi pienempi luvuista $(\pm x, \bmod n)$, ja vastaavasti y':ksi pienempi luvuista $(\pm y, \bmod n)$. Silloin u on jompikumpi luvuista x' tai y'.

4. A ei voi tietää kumpi luvuista x' tai y' on u, joten hän arvaa. A:n ei kannata lähettää arvaamansa lukua B:lle. (Sillä jos se ei olekaan u, niin B:llä on erottuvat neliöjuuret modulo n ja hän voi vaihtaa valitsemansa luvun u.) Sen sijaan A etsii oikealta lukien ensimmäisen bitin, jossa x':n ja y':n binääriesitykset eroavat, ja lähettää tämän bitin B:lle muodossa ”lukusi j:s bitti on $\ldots$ ”.

5. B kertoo A:lle oliko arvaus oikea (arvottu bitti on 1) vaiko väärä (arvottu bitti on 0). Vaikka B ei kertoisikaan arvonnan tulosta A:lle, hän on silti sitoutunut siihen eikä voi sitä muuttaa.

6. Lopuksi B paljastaa A:lle u:n ja A paljastaa n:n tekijöihinjaon. B ei voi huijata A:ta, sillä hän tietää vain toisen neliöjuurista x' ja y' (muutenhan B voisi jakaa n:n tekijöihin).

Huomautus. *Tässä kuten jatkossakin oletetaan, että valittaessa satunnaisesti luku u tms. ei saada sellaista lukua, että $\mathrm{syt}(u, n) \neq 1$. Tämä tapaushan on hyvin epätodennäköinen, jos n on suuri.*

Toinen protokolla on hieman erilainen:

1. B valitsee erisuuret isot alkuluvut p ja q ja lähettää niiden tulon $n = pq$ A:lle. B valitsee myös satunnaisen luvun a väliltä $1 \leq a < n$, jolle $\mathrm{syt}(a, n) = 1$ ja $\left(\frac{a}{n}\right) = 1$ (ks. Lause 5.14), ja lähettää senkin A:lle. (Jacobin symboli on nopea laskea.) Puolet mahdollisista luvuista a on neliönjäännöksiä modulo n, puolet valeneliönjäännöksiä.

2. A arvaa onko a neliönjäännös modulo n vai ei ja kertoo arvauksensa B:lle.

[3] Alkuperäisviite on BLUM, M.: Coin Flipping by Telephone. A Protocol for Solving Impossible Problems. *SIGACT News* (1981), 23–27.

3. B kertoo A:lle oliko arvaus oikea (arvottu bitti on 1) vaiko väärä (arvottu bitti on 0). B voi tarkistaa asian, sillä hän voi laskea nopeasti Legendren symbolit $\left(\frac{a}{p}\right)$ ja $\left(\frac{a}{q}\right)$. Jälleen, vaikka B ei kertoisikaan arvonnan tulosta A:lle, hän on silti sitoutunut siihen eikä voi sitä muuttaa.

4. Lopuksi B paljastaa A:lle n:n tekijät. B ei voi huijata A:ta, sillä A voi nyt puolestaan laskea Legendren symbolit $\left(\frac{a}{p}\right)$ ja $\left(\frac{a}{q}\right)$.

Yleistäen voidaan arpoa kokonaisluku annetulta väliltä arpomalla sen binääriesityksen bitit yksi kerrallaan (tarvittaessa lisätään esitykseen alkunollia).

Jälkimmäistä protokollaa voidaan yleistää seuraavaan tapaukseen. Arvotaan luvuista $1, 2, \ldots, N$ A:lle ja B:lle kummallekin k lukua siten, että kumpikin tietää omat lukunsa, mutta ei toisen lukuja. Lisäksi vaaditaan, että A ja B eivät voi saada yhtään samaa lukua. (Jos edellistä bitin arvontaa voidaan kutsua ”lantin heitoksi”, niin tätä voitaisiin kutsua ”korttien jakamiseksi”.) Seuraava protokolla[4] tekee tällaisen arvonnan:

1. A valitsee sellaiset erisuurten isojen alkulukujen parit p_i, q_i ($i = 1, 2, \ldots, N$), että $p_i \equiv q_i \equiv 3 \mod 4$.

2. A permutoi luvut $1, 2, \ldots, N$ satunnaisesti jonoksi $j_1, j_2, \ldots, j_N$, laskee tulot $n_i = p_i q_i$ ja liittää lukuun j_i luvun n_i ($i = 1, 2, \ldots, N$). Tämä kytkentä on kiinteä eikä A voi sitä muuttaa. Permutointi tarvitaan, ettei B voisi arvata kytkentää!

3. A koodaa kunkin luvun j_i vektoriksi $\mathbf{t}_i = (t_{i1}, t_{i2}, \ldots, t_{il})$, missä l on N:n binääriesityksen pituus ja t_{iu} on satunnaisesti väliltä $1 \leq t_{iu} < n_i$ valittu luku, jolle

$$\left(\frac{t_{iu}}{n_i}\right) = 1 \quad (u = 1, 2, \ldots, l).$$

   Lisäksi t_{iu} on neliönjäännös modulo n_i, jos j_i:n binääriesityksen u:s bitti on 1, ja muuten valeneliönjäännös modulo n_i. (Tarvittaessa lisätään esityksiin alkunollia.)

4. A lähettää B:lle luvut $n_1, n_2, \ldots, n_N$ sekä vektorit $\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N$.

5. Jakaakseen B:lle lukuja $i = 1, 2, \ldots, N$ A arpoo B:lle N lukua $x_1, x_2, \ldots, x_N$. A ei tiedä näitä lukuja, mutta B tietää ne.

6. B laskee luvut $(x_i^2 \mod n_i)$ ja Jacobin symbolit $\left(\frac{x_i}{n_i}\right)$ ($i = 1, 2, \ldots, N$).

7. B lähettää A:lle luvut $(x_i^2 \mod n_i)$. Hän lähettää A:lle myös Jacobin symbolit $\left(\frac{x_i}{n_i}\right)$, paitsi k:lle i:n arvolle $i = v_1, v_2, \ldots, v_k$, joille hän lähettääkin A:lle $-\left(\frac{x_i}{n_i}\right)$:n. A ei tiedä arvoja $v_1, v_2, \ldots, v_k$. (B siis ”haluaa” A:n listalta v_h:nnen luvun ($h = 1, 2, \ldots, k$), mutta ei voi etukäteen tietää mitä saa!)

8. A laskee kullekin saamallensa luvulle $(x_i^2 \mod n_i)$ neliöjuuret modulo n_i, valitsee niistä sellaisen, jonka Jacobin symboli vastaa B:n lähettämää (ks. Lause 5.15), ja lähettää nämä B:lle.

[4] Alkuperäisviite on GOLDWASSER, S. & MICALI, S.: Probabilistic Encryption. *Journal of Computer and Systems Science* **28** (1989), 270–299.

9. B:llä on nyt luvun $(x_{v_h}^2, \mathrm{mod}\ n_{v_h})$ erottuvat neliöjuuret modulo n_{v_h}. Näin ollen hän pystyy jakamaan n_{v_h}:n tekijöihin ja tätä kautta dekoodaamaan vektorin $\mathbf{t}_{v_h}$ vastaavaksi luvuksi j_{v_h}. Tämä on mahdollista, koska B pystyy selvittämään onko $t_{v_h u}$ neliönjäännös modulo n_{v_h} vai ei laskemalla Legendren symbolit

$$\left(\frac{t_{v_h u}}{p_{v_h}}\right) \quad \text{ja} \quad \left(\frac{t_{v_h u}}{q_{v_h}}\right).$$

   Näin B saa tietää jaossa saamansa k lukua $j_{v_1}, j_{v_2}, \ldots, j_{v_k}$, mutta A ei niitä tiedä.

10. B jakaa nyt vuorostaan A:lle k kpl jäljellä olevista $N-k$ luvusta ilmoittamalla vain ne A:lle käyttäen A:n lähettämää listaa. Hän ei tietenkään jaa A:lle itse saamiaan lukuja. Parempi tapa on, että B jakaa luvut A:lle aivan samalla tavoin kuin A jakoi B:lle edellä.

11. Lopuksi osapuolet paljastavat toinen toisilleen kaikki salaisina pidetyt luvut. Näin paljastuu esimerkiksi se, jos B onkin pyytänyt kerralla enemmän kuin k lukua ja näin saanut selville liikaa A:n alkulukupareja. Koska A arpoo B:lle N lukua $x_1, x_2, \ldots, x_N$ sitoen ne (sen sijaan, että B saa valita näitä lukuja, jolloin hän voisi vaihtaa niitä myöhemmin peitellessään jälkiään), A voi jälkikäteen tarkistaa B:n jaossa saamat luvut.

Jos N on iso ja k on iso, on menettely luonnollisesti aikaa viepä, mutta kaikki eri vaiheet ovat kuitenkin polynomiaikaisia. Menettely on yleistettävissä useamman kuin kahden osapuolen tapaukseen.

10.5 Salaisuuksien jakaminen

Jos t ja v ovat positiivisia lukuja ja $t \leq v$, niin (t, v)-*kynnyskaavio* on menettely, jolla salaisuus S jaetaan v:n osapuolen kesken siten, että mitkään $t-1$ osapuolta eivät saa salaisuudesta mitään selville, mutta mitkä tahansa t osapuolta saavat tietää sen kokonaan (*kynnys*).

Kynnyskaaviot toteutetaan usein käyttäen eräänlaista interpolointia. Tietty funktio $f_{p_1,\ldots,p_t}$ (ns. *interpolantti*) määräytyy täysin, kun sen parametrit $p_1, \ldots, p_t$ tunnetaan. Parametrit puolestaan saadaan, jos tunnetaan funktion arvot ainakin t:ssä eri pisteessä:

$$f_{p_1,\ldots,p_t}(x_i) = y_i \quad (i = 1, 2, \ldots, v; v \geq t)$$

Arvot missään $t-1$ pisteessä toisaalta eivät määrää parametrejä yksikäsitteisesti. Salaisuus S on funktio $f_{p_1,\ldots,p_t}$ (tai sen parametrit $p_1, \ldots, p_t$ tai vain osa niistä). Kullekin osapuolelle annetaan tiedoksi oma funktion arvo (ns. *osuus*). Tämän tekee salaa erillinen luotettu osapuoli, ns. *jakaja* J.

Eräs tapa valita interpolantti olisi käyttää Kiinalaista jäännöslausetta, ns. *Mignotten kynnyskaavio* (ks. esimerkiksi SALOMAA ja myös Huomautus pykälässä 6.5). Toinen tapa on käyttää polynomi-interpolanttia

$$p(x) = S + \sum_{j=1}^{t-1} p_{j+1} x^j.$$

Tätä kutsutaan *Shamirin kynnyskaavioksi*[5]. Se voidaan toteuttaa missä tahansa kunnassa $\mathbb{F}$, jossa on vähintään v alkiota. Tavallisin valinta on alkukunta $\mathbb{Z}_q$, missä $q > v$ on alkuluku. Salaisuus on $p(x)$:n vakiotermi S $(= p_1)$. Tunnetusti enintään astetta $t-1$ oleva polynomi määräytyy, kun sen arvot tunnetaan t:ssä eri pisteessä. Toisaalta polynomi ei määräydy yksikäsitteisesti,

[5]Alkuperäisviite on SHAMIR, A.: How to Share a Secret. *Communications of the ACM* **22** (1979), 612–613.

jos asteluku on $t-1$ ja pisteitä on vähemmän kuin t kpl. Erityisesti polynomin vakiotermi ei tällöin määräydy (ellei anneta arvoa nimenomaan pisteessä $x = \mathbb{0}$). Jos nimittäin vakiotermi S määräytyisi yksikäsitteisesti $t-1$:stä arvosta $y_i = p(x_i)$ eri pisteissä $x_i \neq \mathbb{0}$ ($i = 1, 2, \ldots, t-1$), niin saadaan yhtälöt

$$\frac{y_i - S}{x_i} = \sum_{j=1}^{t-1} p_{j+1} x_i^{j-1} \quad (i = 1, 2, \ldots, t-1),$$

joista määräytyvät myös loput parametrit $p_2, \ldots, p_t$. (Ja, kuten näkyy, S voi olla mitä vaan eli mitään tietoa S:stä ei tällöin välity.)

Itse interpolointi voidaan toteuttaa käyttäen lineaarista yhtälöryhmää (jonka matriisi on ns. Vandermonden matriisi) tai esimerkiksi Lagrangen interpolaatiota (ks. peruskurssit)

$$p(x) = \sum_{j=1}^{t} y_j \prod_{\substack{k=1 \\ k \neq j}}^{t} \frac{x - x_k}{x_j - x_k}.$$

Tällöin

$$S = p(\mathbb{0}) = p(x) = \sum_{j=1}^{t} y_j \prod_{\substack{k=1 \\ k \neq j}}^{t} \frac{x_k}{x_k - x_j}.$$

Pisteet, joissa $p(x)$:n arvot lasketaan, voidaan pitää julkisinakin (jolloin osuus olisikin vain ko. arvo). Tällöin S:n laskeminen on vain lineaariyhdelmän lasku osuuksista tunnetuin (ehkä etukäteen lasketuin) kertoimin.

Itse kaavio on seuraava:

Shamirin kynnyskaavio:

1. J valitsee kunnan $\mathbb{F}$ ja sen v eri alkiota $u_1, u_2, \ldots, u_v$ ($\neq \mathbb{0}$) ja antaa alkion u_i tiedoksi i:nnelle osapuolelle ($i = 1, 2, \ldots, v$). Kunta $\mathbb{F}$ ja alkiot $u_1, u_2, \ldots, u_t$ ovat julkisia.

2. Jos tarkoitus on jakaa salaisuus S (jokin $\mathbb{F}$:n alkio), niin J valitsee salaa (satunnaisesti) $t-1$ kpl $\mathbb{F}$:n alkioita: $p_2, \ldots, p_t$.

3. J laskee osuudet
$$w_i = S + \sum_{j=1}^{t-1} p_{j+1} u_i^j \quad (i = 1, 2, \ldots, v)$$
ja jakaa kullekin osapuolelle muilta salaa hänen osuutensa.

4. Kun osapuolet $i_1, i_2, \ldots, i_t$ haluavat tietää salaisuuden, he interpoloivat ja laskevat S:n. Esimerkiksi Lagrangen interpolaatiolla
$$S = \sum_{j=1}^{t} w_{i_j} \prod_{\substack{k=1 \\ k \neq j}}^{t} \frac{u_{i_k}}{u_{i_k} - u_{i_j}}.$$

Huomautus. *Salaisuuksien jakamista ei pidä sekoittaa hyvin samantapaiseen menettelyyn, ns.* tiedon hajautukseen, *missä hajautetaan tiedosto v osaan, joista mitkä tahansa t riittävät sen rekonstruoimiseen (nopeasti). Erona on se, että $t-1$ osaa voi hyvinkin antaa paljon tietoa*

tiedostosta, ei ehkä kuitenkaan aina kaikkea tietoa. Tiedon hajautus liittyy paljolti virheitä korjaaviin koodeihin[6] *(ks. kurssi Koodausteoria) ja hajautetut osat ovat paljon pienempiä kuin yo. osuudet. Alkuperäisviite on* RABIN, M.O.: *Efficient Dispersal of Information for Security, Load Balancing, and Fault Tolerance.* Journal of the ACM **36** *(1989), 335–348.*

10.6 Tiedostamaton tiedonsiirto

Osapuoli A haluaa siirtää salaisuuden osapuolelle B, kuitenkin niin, että salaisuus ei välttämättä siirry. B tietää tietenkin siirtyikö salaisuus vai ei, mutta A ei sitä tiedä. Itse asiassa A:n kannalta salaisuus siirtyy todennäköisyydellä $1/2$. Yksinkertainen protokolla tätä varten olisi seuraava. Tässä, kuten tavallista, n on kahden ison erisuuren alkuluvun p ja q aukikerrottu tulo. Salaisuus voidaan ajatella juuri näiksi alkuluvuiksi p ja q, varsinainen salaisuus olisi tällöin vaikkapa kryptattu RSAlla käyttäen n:ää. A siis tietää alunperin p:n ja q:n, B ei.

1. B valitsee luvun x väliltä $1 \leq x < n$ ja lähettää A:lle $(x^2, \mathrm{mod}\ n)$:n.

2. A laskee $(x^2, \mathrm{mod}\ n)$:n neljä neliöjuurta

$$(\pm x, \bmod n) \quad \text{ja} \quad (\pm y, \bmod n)$$

   modulo n ja lähettää yhden näistä B:lle. Koska A tietää n:n tekijät, hän voi tämän tehdä nopeasti. A ei voi kuitenkaan tietää mikä neliöjuurista x on.

3. B tarkistaa onko hänen A:lta saamansa neliöjuuri $\equiv \pm x \mod n$. Myönteisessä tapauksessa B ei saa salaisuutta. Muutoin B saa tietoonsa erottuvat neliöjuuret modulo n ja pystyy jakamaan n:n tekijöihin ja sitä kautta saamaan selville salaisuuden. A ei voi tietää saiko B salaisuuden vai, ellei B sitä kerro.

Toinen variantti tiedostamattomasta tiedonsiirrosta on sellainen, jossa A:lla on useita salaisuuksia $S_1, S_2, \ldots S_k$, joista yhden hän välittää (myy?) B:lle, kuitenkaan tietämättä minkä niistä. (Ts. A:n kannalta kunkin salaisuuden välittymisen todennäköisyys on $1/k$.) Protokolla, jolla tämä toteutetaan, voidaan rakentaa RSAn päälle seuraavasti.

1. A pystyttää k kpl RSA-systeemejä käyttäen alkulukupareja p_j, q_j. missä $p_j \equiv q_j \equiv 3 \mod 4$ ($j = 1, 2, \ldots, k$).

2. A lähettää B:lle kryptausavaimet (n_j, a_j), missä $n_j = p_j q_j$ ja a_j on kryptauseksponentti ($j = 1, 2, \ldots, k$). A lähettää B:lle myös salaisuudet kryptatussa muodossa

$$(S_j^{a_j}, \bmod n_j) \quad (j = 1, 2, \ldots, k).$$

3. B valitsee luvun x_j väliltä $1 \leq x_j < n_j$ ja laskee Jacobin symbolin $\left(\frac{x_j}{n_j}\right)$ sekä neliön $(x_j^2, \mathrm{mod}\ n_j)$ ($j = 1, 2, \ldots, k$).

4. Jos B haluaa salaisuuden S_i, hän lähettää A:lle neliöt $(x_j^2, \mathrm{mod}\ n_j)$ ja Jacobin symbolit $\left(\frac{x_j}{n_j}\right)$ ($j = 1, 2, \ldots, k$), paitsi arvolle $j = i$, jolle hän lähettääkin A:lle vastakkaismerkkisen Jacobin symbolin $-\left(\frac{x_i}{n_i}\right)$.

[6] Koodausteoriaan perustuvia salaisuuksien jakoprotokollia on myös useita, ks. esimerkiksi DING & PEI & SALOMAA.

5. A laskee saamilleen k:lle neliölle neliöjuuret ja palauttaa B:lle sellaiset neliöjuuret, joiden Jacobin symboli on B:n lähettämä. Vrt. edellisen pykälän arpomisprotokolla. A voi tämän tehdä nopeasti, sillä hän tietää tekijöihinjaot.

6. B:llä on nyt erottuvat neliöjuuret modulo n_i, joten hän pystyy laskemaan nopeasti tekijät p_i ja q_i ja saa selville salaisuutensa RSA-dekryptauksen jälkeen.

Huomattakoon, että B voi petkuttaa vaihtamalla useamman kuin yhden Jacobin symbolin merkin (tällaista kutsutaan *aktiiviseksi petkuttamiseksi*). Poikkeamatta säännöistä (*passiivinen petkuttaminen*) petkutus ei onnistu. Aktiivisen petkuttamisen estämiseksi voi valvojana olla mukana luotettu osapuoli L, jolle B lähettää luvut x_j ja n_j ja A puolestaan vastaanottamansa neliöt ja Jacobin symbolit.

10.7 Nollatietotodistukset

Interaktiivisessa todistussysteemissä kaksi osapuolta, *todistaja* P ja *todentaja* V, lähettävät toisilleen viestejä ja suorittavat laskuja saamiensa viestien perusteella (mukana voi olla satunnaislukujen generointia). P:n tavoite on saada V vakuuttuneeksi siitä, että hän tietää jonkin kohteen jonkin ominaisuuden (kohde voisi olla vaikkapa matemaattinen tulos ja ominaisuus sen totuus, mutta tietysti muunkinlainen). P:n tavoite toisaalta on olla välittämättä V:lle mitään muuta informaatiota kuin sen, että hän tietää ko. ominaisuuden. Tätä kutsutaan *nollatietotodistukseksi.*

Seuraava yksinkertainen protokolla sallii P:n todistaa, että hän tietää luvun $n = pq$ alkutekijät p ja q paljastamatta niitä.[7] Tässä oletetaan, että $p \equiv q \equiv 3 \mod 4$ ja tietysti, että p ja q ovat erisuuret ja kyllin isot.

1. V valitsee satunnaisesti luvun x väliltä $1 \leq x < n$ ja lähettää P:lle $(x^4,\mathrm{mod}\ n)$:n.

2. P kertoo V:lle $(x^2,\mathrm{mod}\ n)$. Huomaa, että tämä on se yksi ja ainoa neliöjuuri, joka on neliönjäännös modulo n (eli pääneliöjuuri, ks. pykälä 5.7) ja P:n on löydettävä se. V ei saa mitään uutta tietoa, hänhän voi laskea $(x^2,\mathrm{mod}\ n)$:n itsekin.

V voisi yrittää petkuttaa aktiivisesti valitsemalla sellaisen x:n, että $\left(\frac{x}{n}\right) = -1$, ja lähettämällä P:lle aluksi $(x^2,\mathrm{mod}\ n)$:n. Sitä kautta hän saa erottuvat neliöjuuret modulo n ja lopulta p:n ja q:n.[8] Tämä protokolla on siis sen varassa, että V ei petkuta.

Nollatietotodistuksen perusvaatimukset ovat

(I) Todennäköisyys, että P voisi petkuttaa V:tä, on hyvin pieni. (Jos P ei esimerkiksi itse asiassa tiedä matemaattisen tuloksen todistusta, vaikka niin väittää, niin hänen mahdollisuutensa petkuttaa V:tä ovat minimaaliset.)

(II) Jos P tietää ko. ominaisuuden, hän voi kiistatta todistaa sen V:lle.

(III) V ei saa P:ltä mitään informaatiota, mitä hän ei voisi hankkia itse ilman P:tä, tarvittaessa laskien polynomiajassa. (Tällöin V voisi itse asiassa simuloida protokollaa polynomiajassa ikäänkuin P:kin osallistuisi siihen, mutta ilman P:tä. Huomaa, että P:n laskujen kompleksisuudelle ei aseteta mitään rajaa. Simuloinnin on oltava niin tarkka, ettei sitä voida erottaa ”oikeasta” polynomiajassa laskien.)

[7] Alkuperäisviite on DAMGÅRD, I.B.: On the Existence of Bit Commitment Schemes and Zero-Knowledge Proofs. CRYPTO ’89. *Lecture Notes in Computer Science* **435**. Springer–Verlag (1990), 6–16.

[8] Tämä protokollan heikkous ei tunnu olevan yleisesti tiedossa. Se mainitaan kirjassa KOBLITZ (2000) harjoitustehtävänä ja sen keksi myös TTKK:n tutkija Erkko Lehtonen!

Ehdosta (III) huolimatta V voisi tietenkin saada pitkään laskien esiin lisää informaatiota, ehkä koko ominaisuuden. *Täydellisessä nollatietotodistuksessa* vaaditaankin (III):n sijasta vahvempi ehto

(III′) V ei saa P:ltä mitään informaatiota, mitä hän ei voisi hankkia itse ilman P:tä. (V laskee tässäkin polynomiajassa, mutta simuloinnin on nyt oltava identtinen "oikean" kanssa.)

Toisinaan yllä määriteltyä nollatietotodistusta kutsutaan *laskennalliseksi nollatietotodistukseksi,* erotukseksi täydellisestä nollatietotodistuksesta. Huomattakoon, että yo. ehdot eivät oikeastaan muodosta tarkkoja määritelmiä. Nämä määritelmät ovatkin huomattavasti mutkikkaampia, ks. esimerkiksi STINSON tai GOLDREICH. Ero laskennallisen ja täydellisen nollatietotodistuksen välillä on stokastisuudesta johtuvien jakaumien vertailussa: Täydellisessä nollatietotodistuksessa "oikeiden" ja simuloitujen jakaumien on oltava identtiset, laskennallisessa nollatietotodistuksessa taas vaaditaan vain, ettei jakaumia voi havaita erilaisiksi polynomiaikaisilla laskuilla.

Seuraava protokolla[9] antaa täydellisen nollatietotodistuksen sille, että x on neliönjäännös modulo n, missä $n = pq$ ja p sekä q ovat erisuuria isoja alkulukuja (olettaen, että $\mathrm{syt}(x, n) = 1$). (Tämä probleema on QUADRATICRESIDUES, todistus on jokin x:n neliöjuuri modulo n.)

1. Toistetaan seuraava k kertaa.

   1.1 P valitsee väliltä $1 \leq v < n$ satunnaisluvun v, jolle $\mathrm{syt}(v, n) = 1$, ja lähettää V:lle luvun $y = (v^2 \mathrm{,mod}\ n)$.

   1.2 V valitsee satunnaisesti bitin b (0 tai 1) ja lähettää sen P:lle.

   1.3 P laskee luvun $z = (u^b v \mathrm{,mod}\ n)$, missä u on x:n neliöjuuri modulo n, ja lähettää sen V:lle.

   1.4 V tarkistaa, että $z^2 \equiv x^b y \mod n$.

2. Jos tarkistus menee aina läpi kullakin k:lla kierroksella, V päättelee, että P todella tietää, että x on neliönjäännös modulo n.

Lause 10.3. *Yo. protokolla antaa täydellisen nollatietotodistuksen probleemalle* QUADRATICRESIDUES.

Todistus. Jos P ei tiedä x:n neliöjuurta, hän joutuu petkuttamaan ja lähettämään V:lle luvun $z = v$ ja joko luvun $y = (z^2 \mathrm{,mod}\ n)$ (paljastuu, jos $b = 1$) tai luvun $y = (z^2 x^{-1} \mathrm{,mod}\ n)$ (paljastuu, jos $b = 0$). Todennäköisyys, että P voisi paljastumatta petkuttaa koko ajan, on siis $1/2^k$, joka saadaan miten tahansa pieneksi. Jos taas P tietää neliöjuuren u, hän tietysti selviää testistä joka kerta.

V voi simuloida P:n osuutta protokollassa täydellisesti. Idea on, että V generoi kolmikoita (y, b, z), missä

$$y \equiv z^2 x^{-b} \mod n.$$

Näytetään, että jos V valitsee bitin b ja z:n aivan satunnaisesti, näillä kolmikoilla on identtinen jakauma verrattunaan "oikeaan" (jossa P on mukana ja valitsee satunnaisen v:n). Sanotaan, että (y, b, z) on *käypä,* jos

- $1 \leq y < n$ ja $\mathrm{syt}(y, n) = 1$,
- b on 0 tai 1 ja

[9] Alkuperäisviite on GOLDWASSER, S. & MICALI, S. & RACKOFF, C.: The Knowledge Complexity of Interactive Proof Systems. *SIAM Journal on Computing* **18** (1989), 186–208.

- $1 \leq z < n$ ja $z^2 \equiv x^b y \mod n$.

Tällaisia kolmikoita on $2\phi(n)$ kpl ja ne esiintyvät protokollassa yhtä todennäköisinä, kun P on mukana. (P:hän valitsee v:n satunnaisesti. Samoin V valitsee b:n arvot samalla todennäköisyydellä.) Simuloituja kolmikoita (y, b, z) on myös $2\phi(n)$ kpl, kun V valitsee z:n satunnaisesti väliltä $1 \leq z < n$ ja $\text{syt}(z, n) = 1$. Selvästi jokainen simuloitu kolmikko on käypä ja ne ovat simuloinnissa yhtä todennäköisiä. (V valitsee z:n ja b:n satunnaisesti.) Käyvät kolmikot ja simuloidut kolmikot ovat siis samat ja esiintyvät samalla todennäköisyydellä $1/(2\phi(n))$. □

Esitetään esimerkki myös (laskennallisesta) nollatietotodistuksesta. Probleemana on todistaa, että graafissa on ns. Hamiltonin piiri. *Graafi* muodostuu *pisteistä* ja niitä yhdistävistä *viivoista*. Kaikkien pisteiden välillä ei (yleensä) ole viivoja. *Hamiltonin piiri* on reitti, joka kiertää piirin graafin kaikkien pisteiden kautta, käyden kussakin pisteessä kerran ja palaten lähtöpisteeseen. Reitti kulkee viivoja pitkin. (Ks. kurssi Graafiteoria.) Sen selvittäminen, onko annetussa (sopivasti koodatussa) graafissa Hamiltonin piiri vai ei, on tunnettu $\mathcal{NP}$-täydellinen tunnistustehtävä (HAMILTONCIRCUIT). Seuraava protokolla[10] antaa nollatietotodistuksen tälle tehtävälle.

1. Toistetaan seuraava k kertaa. Syöte on tässä graafi G, jossa on pisteet $1, 2, \ldots, n$.

   1.1 P järjestää pisteet satunnaiseen järjestykseen ja lähettää näin saadun listan $v_1, v_2, \ldots, v_n$ (tarvittaessa koodattuna bittijonoksi) *kryptattuna* V:lle. P lähettää V:lle myös *alkioittain kryptattuna* $n \times n$-matriisin $\mathbf{D}$ (ns. *vieruspistematriisi*), missä lävistäjäalkiot ovat nollia ja

   $$\mathbf{D}_{ij} = \begin{cases} 1, \text{ jos pisteiden } v_i \text{ ja } v_j \text{ välillä on viiva} \\ 0 \text{ muuten,} \end{cases}$$

   kun $i \neq j$. (Itse asiassa riittää lähettää vain yläkolmio.) Kukin matriisin alkio on kryptattu omalla avaimellaan. Kryptauksen on oltava sitova, ts. P ei voi muuttaa graafia avaimia muuttamalla (vrt. pykälän 10.4 arpominen). Luonnollisesti kryptauksen oletetaan olevan tehokas, ts. polynomiajassa ei kryptatusta bitistä saa irti mitään.

   1.2 V valitsee satunnaisesti bitin b ja lähettää sen P:lle.

   1.3 Jos $b = 0$, P dekryptaa V:lle listan $v_1, v_2, \ldots, v_n$ ja koko matriisin $\mathbf{D}$ (lähettämällä V:lle dekryptausavaimet). Jos taas $b = 1$, P dekryptaa V:lle matriisista $\mathbf{D}$ vain n alkiota $\mathbf{D}_{i_1 i_2}, \mathbf{D}_{i_2 i_3}, \ldots, \mathbf{D}_{i_n i_1}$, missä pisteet $v_{i_1}, v_{i_2}, \ldots, v_{i_n}$ (tässä järjestyksessä) muodostavat Hamiltonin piirin (jolloin ko. alkiot ovat $= 1$).

   1.4 Jos $b = 0$, V tarkistaa, että hän sai oikean graafin (dekryptattuna lista $v_1, v_2, \ldots, v_n$ ilmoittaa pisteiden järjestyksen ja $\mathbf{D}$ antaa viivat). Jos taas $b = 1$, V tarkistaa, että saadut matriisin alkiot ovat $= 1$.

2. Jos tarkistus menee aina läpi kullakin k:lla kierroksella, V päättelee, että P todella tietää G:n Hamiltonin piirin.

Protokollassa kohdassa 1.1 mainitun kaltainen sitova kryptaus saadaan aikaan vaikkapa seuraavasti (menettelyjä on useita, ks. esimerkiksi STINSON). Tässä suuri alkuluku p ja primitiivinen juuri g modulo p ovat julkisia.

[10] Alkuperäisviite lienee BLUM, M: How to Prove a Theorem So No One Else Can Claim It. *Proceedings of the International Congress of Mathematicians 1986.* American Mathematical Society (1988), 1444–1451.

1. Alussa V valitsee ja välittää sitten P:lle satunnaisen luvun r väliltä $1 < r < p$. P ei voi nopeasti laskea diskreettiä logaritmia $\log_g r$ modulo p.

2. P valitsee satunnaisesti luvun y väliltä $0 \leq y < p-1$ (salainen avain) ja lähettää V:lle luvun $c = (r^b g^y, \mathrm{mod}\ p)$, missä b on kryptattava bitti. Jokainen $\mathbb{Z}_p^*$:n alkio on (positiivisessa jäännössysteemissä) sekä muotoa $(g^y, \mathrm{mod}\ p)$ että muotoa $(rg^y, \mathrm{mod}\ p)$, joten c ei siis paljasta mitään bitistä b (eli olipa b kumpi tahansa, c:n jakauma on sama). Toisaalta P ei voi vaihtaa bittiä b toiseksi vaihtamalla salaisen avaimen y':ksi, muuten on

$$g^y \equiv rg^{y'} \mod p \quad \text{tai} \quad rg^y \equiv g^{y'} \mod p$$

eli

$$r \equiv g^{\pm(y-y')} \mod p$$

ja P saisi tästä välittömästi $\log_g r$:n modulo p.

Lause 10.4. *Yo. protokolla antaa nollatietotodistuksen probleemalle* HAMILTONCIRCUIT.

Todistus. Jos P ei tiedä G:n Hamiltonin piiriä, hän voi petkuttaa vain silloin, kun hän saa bitin $b = 0$ (mutta ei jos hän saa bitin $b = 1$). Jos taas P tietää jonkin toisen n-pisteisen graafin G' Hamiltonin piirin, hän voi petkuttaa, kun hän saa bitin $b = 1$ (mutta ei jos hän saa bitin $b = 0$). Todennäköisyys, että P voisi petkuttaa koko ajan, on siis $1/2^k$, joka saadaan miten tahansa pieneksi. Jos taas P tietää G:n Hamiltonin piirin, hän tietysti selviää testistä joka kerta.

V voi simuloida protokollaa polynomiajassa myös ilman P:tä. V menettelee tällöin seuraavasti. V valitsee satunnaisen bitin b. Jos $b = 0$, V järjestää pisteet satunnaiseen järjestykseen ja kryptaa näin saamansa listan. Vielä V muodostaa matriisin $\mathbf{D}$ ja kryptaa sen. Jos taas $b = 1$, V kryptaakin vain jotkut (satunnaiset) alkiot $\mathbf{D}_{i_1 i_2}, \mathbf{D}_{i_2 i_3}, \ldots, \mathbf{D}_{i_n i_1}$, missä indeksit kulkevat syklisesti ja kukin indeksi esiintyy tarkalleen kaksi kertaa. (Täydellisyyden vuoksi V voi kryptata vielä muutakin, jotta saadaan oikea määrä kryptattua dataa.) Koska käytetty kryptaus on tehokas, ovat kryptatut alkiojonot hyvin ”samannäköisiä” riippumatta siitä muodostavatko ne Hamiltonin piirin vai ei. Ts. polynomiajassa laskien ei eroa saada näkyviin eikä esiintyviä jakaumia voida myöskään erottaa polynomiajassa laskien. Jakaumien ei silti tarvitse olla tarkalleen samat! □

HAMILTONCIRCUIT on $\mathcal{NP}$-täydellinen tehtävä, johon muut $\mathcal{NP}$:ssä olevat tunnistustehtävät voidaan polynomiajassa redusoida (ks. pykälä 5.1). V voi näin ollen tehdä tämän reduktion. Näin ollen pätee

Lause 10.5. *Jokaisen $\mathcal{NP}$:ssä olevan tunnistustehtävän positiiviselle ratkaisulle voidaan antaa nollatietotodistus.*

Arvellaan, ettei millekään $\mathcal{NP}$-täydelliselle tunnistustehtävälle voida antaa täydellistä nollatietotodistusta, ts. Lause 10.5 ei pidä paikkaansa, jos vaaditaan, että nollatietotodistus on täydellinen. Itse asiassa tiedetään Lausetta 10.5 yleisempikin tulos:

Lause 10.6. (Shamirin lause[11]) *Tunnistustehtävät, joiden positiivisille ratkaisuille on nollatietotodistus, ovat tarkalleen kaikki luokan $\mathcal{PSPACE}$ tunnistustehtävät.*

[11] Alkuperäisviite on SHAMIR, A.: IP = PSPACE. *Journal of the ACM* **39** (1992), 869–877.

Kirjallisuus

1. ADÁMEK, J.: *Foundations of Coding. Theory and Applications of Error-Correcting Codes with an Introduction to Cryptography and Information Theory.* Wiley (1991)

2. BAUER, F.L.: *Decrypted Secrets. Methods and Maxims of Cryptography.* Springer–Verlag (2002)

3. BLAKE, I. & SEROUSSI, G. & SMART, N.: *Elliptic Curves in Cryptography.* Cambridge University Press (2000)

4. BUCHMANN, J.: *Introduction to Cryptography.* Springer–Verlag (2001)

5. COHEN, H.: *A Course in Computational Algebraic Number Theory.* Springer–Verlag (1996)

6. CRANDALL, R. & POMERANCE, C.: *Prime Numbers. A Computational Perspective.* Springer–Verlag (2001)

7. DENNING, D.E.R.: *Cryptography and Data Security.* Addison–Wesley (1983)

8. DING, C. & PEI, D. & SALOMAA, A: *Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography.* World Scientific (1997)

9. GARRETT, P.: *Making, Breaking Codes: An Introduction to Cryptology.* Prentice–Hall (2001)

10. GOLDREICH, O.: *Modern Cryptography, Probabilistic Proofs, and Pseudorandomness.* Springer–Verlag (1999)

11. GOLDREICH, O.: *Foundations of Cryptography. Basic Tools.* Cambridge University Press (2001)

12. GOLDREICH, O.: *Foundations of Cryptography. Basic Applications.* Cambridge University Press (2004)

13. HOPCROFT, J.E. & ULLMAN, J.D.: *Introduction to Automata Theory, Languages, and Computation.* Addison–Wesley (1979)

14. KNUTH, D.E.: *The Art of Computer Programming Vol. 2: Seminumerical Algorithms.* Addison–Wesley (1998)

15. KOBLITZ, N.: *A Course in Number Theory and Cryptography.* Springer–Verlag (2000)

16. KOBLITZ, N.: *Algebraic Aspects of Cryptography.* Springer–Verlag (1998)

17. KONHEIM, A.G.: *Cryptography. A Primer.* Wiley (1981)

18. KRANAKIS, E.: *Primality and Cryptography.* Wiley (1986)

19. LIDL, R. & NIEDERREITER, H.: *Introduction to Finite Fields and Their Applications.* Cambridge University Press (1994)

20. LIPSON, J.D.: *Elements of Algebra and Algebraic Computing.* Addison–Wesley (1981)

21. MCELIECE, R.J.: *Finite Fields for Computer Scientists and Engineers.* Kluwer (1987)

22. MENEZES, A. & VAN OORSCHOT, P. & VANSTONE, S.: *Handbook of Applied Cryptography.* CRC Press (2001)

23. MEYER, C.H. & MATYAS, S.M.: *Cryptography: A New Dimension in Computer Data Security.* Wiley (1982)

24. MIGNOTTE, M.: *Mathematics for Computer Algebra.* Springer–Verlag (1991)

25. MOLLIN, R.A.: *An Introduction to Cryptography.* Chapman & Hall / CRC (2001)

26. MOLLIN, R.A.: *RSA and Public-Key Cryptography.* Chapman & Hall / CRC (2003)

27. RIESEL, H.: *Prime Numbers and Computer Methods for Factorization.* Birkhäuser (1994)

28. ROSEN, K.H..: *Elementary Number Theory.* Longman (2000)

29. ROSING, M.: *Implementing Elliptic Curve Cryptography.* Manning Publications (1998)

30. SALOMAA, A.: *Public-Key Cryptography.* Springer–Verlag (1996)

31. SCHNEIER, B.: *Applied Cryptography. Protocols, Algorithms, and Source Code in C.* Wiley (1996)

32. SIERPINSKI, W.: *Elementary Theory of Numbers.* Elsevier (1987)

33. SILVERMAN, J.H. & TATE, J.: *Rational Points on Elliptic Curves.* Springer–Verlag (1992)

34. STINSON, D.R.: *Cryptography. Theory and Practice.* Chapman & Hall / CRC (2002)

35. VAN DER LUBBE, J.C.A.: *Basic Methods of Cryptography.* Cambridge University Press (1998)

36. WAGSTAFF, S.S.: *Cryptanalysis of Number Theoretic Ciphers.* Chapman & Hall / CRC (2003)

Hakemisto