Kriptografija

13 Įvadas

Jei kompiuteris yra kasdienės jūsų veiklos įrankis, tikriausiai esate patyrę, kaip muša šaltas prakaitas pastebėjus, kad po jūsų skaitmeninių duomenų – dokumentų ir programų – archyvą pasišvaistyta piktavalio viruso ar neišmanėlio jaunesniojo brolio. O jeigu kas įveiktų banko duomenų apsaugos sistemą ir patvarkytų jūsų sąskaitas?

Taigi informacijos apsaugos reikšmė yra nepaprastai didelė ir vis didėja.

Galima teigti, kad ir kodavimo teorija, nagrinėta antrojoje knygos dalyje sprendžia informacijos apsaugos uždavinius. Iš tiesų, siuntėjas A prieš siųsdamas informaciją gavėjui B gali pasinaudoti kodavimo teorijos idėjomis ir parengti informaciją taip, kad tikimybė, jog B gaus neiškraipytą informaciją, padidėtų. Šitaip sprendžiamas kanalu perduodamos informacijos vientisumo išsaugojimo uždavinys, kai pažeidimų (iškraipymų) priežastis – atsitiktiniai fizinės prigimties trikdžiai.

O dabar įsivaizduokime, kad informacijos perdavimo kanalą gali kontroliuoti protingas moderniausia skaičiavimo technika ir naujausiomis žiniomis apsiginklavęs Z. Šio skyriaus paskirtis – aptarti informacijos apsaugos uždavinius, kurie tuomet iškyla.

13.1. Kriptologijos dėmenys

Kriptologija – mokslas apie matematines duomenų apsaugos priemones. Kas ją sudaro ir kaip keliami jos uždaviniai?

Apžvelkime gerokai supaprastintą padėtį, į kurią patenka šiuolaikinės informacinės visuomenės žmonės: A (Algis) siunčia informaciją B (Birutei), perdavimo kanalą kontroliuoja Z (Zigmas) ir jaučiasi padėties šeimininkas. Jis gali pasyviai stebėti perdavimo kanalą, skaityti siunčiamus pranešimus ir kaupti dosjė; jis gali veikti aktyviai – pakeisti dalį siunčiamos informacijos, o kartais – apsimesti A ir siųsti jo vardu pranešimus B arba apsimesti B. Taigi dėl Zigmo veiksmų gali būti pažeidžiamos šios siunčiamų duomenų savybės:

- slaptumas;
- vientisumas;
- autentiškumas.

Kaip to išvengti? Fizinė kanalo apsauga gali būti pernelyg brangi arba tiesiog neįmanoma. Kokiomis priemonėmis galėtumėte apsaugoti, pavyzdžiui, jūsų elektroninio pašto perdavimo kanalą?

Prisiminkime kodavimo teorijos konstrukcijas. Informacijos, kurią reikia perduoti, blokai keičiami specialaus kodo žodžiais. Taigi – galimų iškraipymų taisymo priemonės „įmontuojamos“ pačioje perduodamų duomenų struktūroje. Ar panašiai negali būti sprendžiami ir informacijos apsaugos uždaviniai, kuriuos kelia Zigmo egzistavimas?

Kriptografija bendriausiu požiūriu ir yra duomenų apsaugos uždavinių sprendimo matematiniais metodais mokslas. Pagrindiniai šios apsaugos tikslai yra trys: užtikrinti informacijos slaptumą, vientisumą ir autentiškumą. Tačiau šiuolaikinė kriptografija tuo toli gražu neapsiriboja. Naudojimasis kompiuteriniais tinklais ir duomenų bazėmis kelia daug naujų įdomių teorinių ir praktinių uždavinių. Tik pagalvokime, pavyzdžiui, kiek reikalavimų reiktų įgyvendinti ir numatyti atvejų, norint tinkamai organizuoti elektroninius rinkimus! Pirmiausia reikia išduoti biuletenius turintiems teisę balsuoti, užkirsti kelią įvairaus pobūdžio klastotėms, garantuoti balsavimo anonimiškumą, suteikti rinkėjui galimybę patikrinti, kad jo balsas tinkamai įskaitytas...

Jeigu A ir B savo ryšiui apsaugoti naudoja tinkamus kriptografijos metodus, tai Zigmui, net ir kontroliuojant perdavimo kanalą, turėtų būti sunku „prisibrauti“ prie siunčiamos informacijos prasmės, pakeisti ką nors ir likti nepastebėtam arba pasiųsti pranešimą A ar B vardu. Sunku, bet nėra neįmanoma! Z kanalu siunčiamiems informacijos srautams analizuoti irgi gali naudoti mokslinius metodus. Jis gali žinoti daugelį naudojamų kriptografinių algoritmų detalių (išskyrus slaptus parametrus – raktus) ir bandyti įveikti kriptografinę duomenų apsaugą. Jo metodų visuma irgi yra mokslas – kriptoanalizė. Apibendrinant galima teigti, kad kriptoanalizė – tai duomenų kriptografinių apsaugos algoritmų patikimumo vertinimo mokslas. Kriptografija siūlo, o kriptoanalizė vertina. Šiuo požiūriu Z nėra A ir B priešas, bet pagalbininkas. Nieko nėra blogiau už apsaugą, kurios patikimumas nėra įvertintas!

O sugretinę kriptografiją ir kriptoanalizę, gauname kriptologiją – mokslą apie duomenų apsaugą naudojant matematikos ir informatikos priemones:

$$\text{kriptologija} = \text{kriptografija} + \text{kriptoanalizė}.$$

Verta paminėti, kad dešinėje lygybės pusėje sau vietos ima reikalauti dar vienas dėmuo – steganografija. Internetu dabar siunčiame pačią įvairiausią informaciją: dokumentus, fotografijas, audio- ir videoinformaciją... Ar negalima siunčiamoje kokio nors obuolio fotografijoje paslėpti svarbų pranešimą, kurį kitaip būtų rizikinga siųsti? Tokių metodų kūrimo ir analizės sritis ir yra steganografija (steganos – slėpti, graphein – rašau (graik.). Šį žodį 1499 metais pirmą kartą pavartojo vokietis Johannes Trithemius, pirmojo spausdinto kriptografijos veikalo autorius. Tačiau jam steganografija reiškė ne tik

mokymą apie šifrus, bet ir apie įvairius okultinius su paslaptimis susijusius dalykus. Kriptografija – taip pat iš graikiškų žodžių sudarytas pavadinimas (krypto – paslėptas graik.) Šį terminą 1661 metais pirmą kartą panvartojo Johnas Williamsas. Kriptografinės ir steganografinės duomenų apsaugos metodai esmingai skiriasi. Kriptografija neslepia duomenų, bet paslepia jų struktūrą, t. y. prasmę. O steganografija paslepia pačius duomenis.

13.2. Šifrai ir parašai

Šifrai ir skaitmeniniai parašai yra duomenų slaptumo ir autentiškumo užtikrinimo priemonės. Patikslinkime šias sąvokas.

Norėdami, kad pašalinis asmuo neįžvelgtų siunčiamų duomenų prasmės, turime pertvarkyti tuos duomenis taip, kad tik teisėtas gavėjas galėtų atkurti pradinę duomenų struktūrą. Duomenų pertvarkymą, kurio tikslas – paslėpti prasmę, vadinsime šifravimu, o pradinės struktūros atkūrimą – dešifravimu. Pradinius duomenis, kuriems taikoma šifravimo operacija vadinsime nešifruotu arba atviru tekstu, o šifravimo operacijos rezultatą – šifru.

Pagrindinis šiuolaikinės kriptografijos reikalavimas – kad šifravimo ir dešifravimo operacijų rezultatai iš esmės priklausytų nuo tam tikrų dydžių, paprastai vadinamų raktais. Tai reiškia, kad net jeigu visos A ir B ryšio slaptumą saugančios sistemos detalės (išskyrus raktus) taptų žinomos Z, įveikti duomenų apsaugos sistemą jam neturėtų pasidaryti lengviau (mažai naudos bus žmogui, norinčiam paskambinti telefonu, jeigu mes jam paaiškinsime, kad reikia vieną po kito paspausti septynis telefono mygtukus su numeriais, bet nepasakysime kokius).

Duomenų apsaugos sistemą, naudojančią šifravimą, vadinsime kriptografine sistema arba tiesiog kriptosistema.

92 apibrėžimas. *Kriptografine sistema vadinsime trejetą* $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ *čia* $\mathcal{M}$ *– nešifruotų tekstų,* $\mathcal{K}$ *– naudojamų raktų ir* $\mathcal{C}$ *– šifrų aibės, kartu su šifravimo ir dešifravimo operacijomis*

$$e(\cdot|K_e) : \mathcal{M} \to \mathcal{C}, \quad d(\cdot|K_d) : \mathcal{C} \to \mathcal{M}, \quad \langle K_e, K_d \rangle \in \mathcal{K},$$

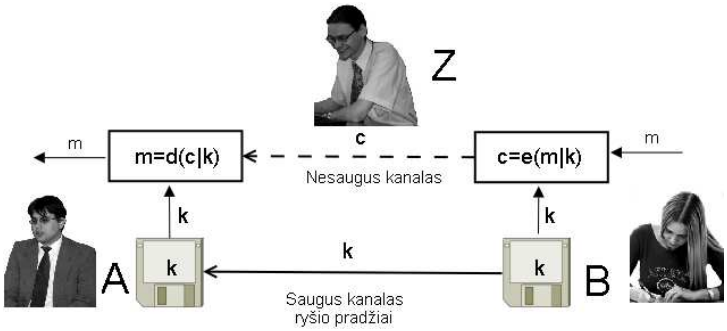
tenkinančiomis sąlygą

$$\text{kiekvienam } M \in \mathcal{M} \quad d(e(M|K_e)|K_d) = M.$$

Galime įsivaizduoti, kad kriptosistemoje naudojamas raktas sudarytas iš dviejų dalių: šifravimui skirto rakto K_e ir dešifravimui skirto rakto K_d.

Gali būti, kad abu raktai sutampa, t. y. $K_e = K_d$, arba, nors formaliai ir būdami skirtingi, lengvai gaunami iš vienas kito. Tokias kriptosistemas vadinsime simetrinėmis. Jeigu Birutė norėtų, kad Algis jai parašytų laišką ir užšifruotų jį simetrine kriptosistema, ji turi slapta nuo Zigmo perduoti Algiui raktą K_e ir jau tuomet laukti laiško. Taigi simetrinės kriptosistemos

veiklos pradžiai būtina nors ir trumpalaikė galimybė pasinaudoti visiškai saugiu kanalu raktui perduoti.



Ryšio apsauga su simetrine kriptosistema

Iki 1976 metų visos kriptosistemos buvo simetrinės. Tais metais du matematikai – Diffie ir Hellmannas[5] paskelbė naujos kartos kriptosistemų principus. Jų esmė tokia: nors dešifravimui skirtas raktas K_d yra susijęs su šifravimui skirtu raktu K_e, tačiau praktiškai, žinant K_e, neturi būti įmanoma per „tikrovišką“ laiką nustatyti K_d. Kaip tai gali būti įmanoma? Galbūt suprasti padės toks „hipotetinis“ pavyzdys.

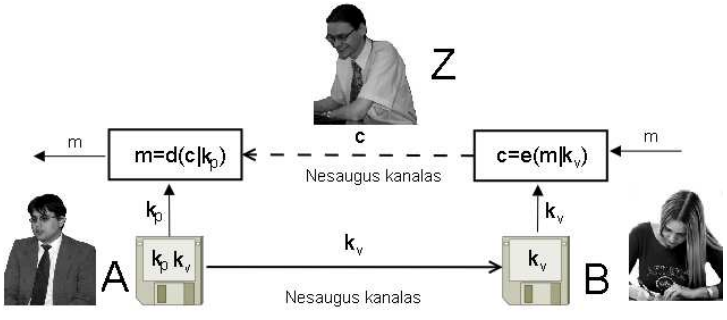
Tarkime, aš sukūriau kriptosistemą ir sudariau raktų porą:

$$K_e = 2, \quad K_d = a_n a_{n+1} \dots a_{n+99}; \qquad n = 10^{10^{10}}, \ \sqrt{K_e} = a_0, a_1 a_2 \dots,$$

čia a_i yra skaičiaus skleidimo begaline dešimtaine trupmena skaitmenys.

Ar, žinodami šifravimui skirtą raktą, greitai galėsite nustatyti, koks raktas naudojamas dešifravimui?

Taigi turėdama tokią kriptosistemą, Birutė gali perduoti šifravimui skirtą raktą K_e ir nesaugiu kanalu (pavyzdžiui, paskelbti savo tinklalapyje). Kadangi šifravimui skirtas raktas gali būti paskelbtas viešai, tokios kriptosistemos pradėtos vadinti viešojo rakto kriptosistemomis. Šifravimui skirtas raktas dažnai vadinamas viešuoju, o dešifravimui – privačiuoju raktu.



[5] W. Diffie, M. E. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, p. 644–654.

Ryšio apsauga su viešojo rakto kriptosistema

Viešojo rakto kriptosistemą galime palyginti su rakinama pašto dėžute. Visi gali mesti laiškus pro plyšį, t. y. naudotis viešuoju raktu, tačiau tik dėžutės savininkas gali atsirakinti ją ir išsiimti (dešifruoti) laiškus.

Dabar viešojo rakto kriptosistemų yra sugalvota daug. Pats įdomiausias dalykas – konstruojant šias sistemas, taikomi „gryniausios" matematikos srities – skaičių teorijos – rezultatai. Erdvė, kuri skyrė praktinius uždavinius sprendžiančius informatikus ir „grynuosius" matematikus, dirbančius skaičių teorijos srityje, tapo didelę įtaką šiuolaikinių ryšių raidai padariusio bendradarbiavimo vieta.

Viešojo rakto kriptosistemų atsiradimas išsprendė ir skaitmeninių parašų uždavinį. Mūsų parašas yra tam tikra grafinė informacija, kurią susiejame su dokumentu. Parašo tikrinimo algoritmas labai paprastas – lyginama su dokumentu. O kaip pasirašyti skaitmeninį dokumentą, t. y. nulių-vienetų srautą? Kaip pridėti tą papildomą mus autentifikuojančią informaciją, kad būtų galima patikrinti, kad ją tikrai sukūrėme mes, o ne kažkas už mus?

Sistemą, skirtą skaitmeninių duomenų autentiškumui įrodyti, vadinsime skaitmeninio parašo schema. Ją formaliai galime apibrėžti taip:

93 apibrėžimas. *Skaitmeninių parašų schemą sudaro aibės $\mathcal{M}, \mathcal{P}, \mathcal{K}$, čia $\mathcal{M}$ – tekstų aibė, $\mathcal{P}$ – skaitmeninių parašų aibė, $\mathcal{K}$ – raktų $K = \langle K_v, K_p \rangle$ aibė (K_p – privačioji, parašui sudaryti skirta rakto komponentė, K_v – viešoji, parašui tikrinti skirta rakto komponentė) ir parašų sudarymo bei tikrinimo algoritmų šeimos*

$$\begin{aligned} sig(\cdot|K_p) &: \mathcal{M} \to \mathcal{P}, \\ ver(\cdot|K_v) &: \mathcal{M} \times \mathcal{P} \to \{0,1\}. \end{aligned}$$

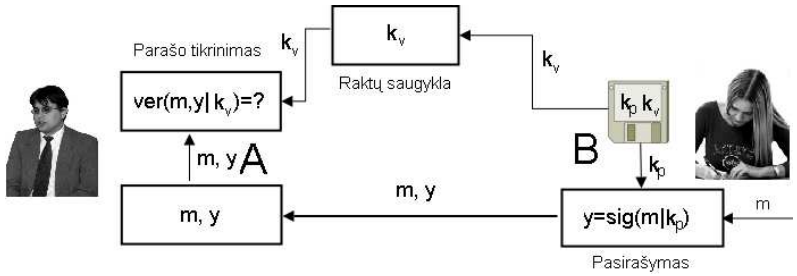
Parašo tikrinimo algoritmai turi savybę:

$$ver(x, sig(x|K_p)|K_v) = 1. \tag{76}$$

Jeigu $y \in \mathcal{P}$ pateikiamas kaip dokumento $x \in \mathcal{M}$ parašas, tai parašas pripažįstamas galiojančiu, jeigu $ver(x, y|K_v) = 1$, ir pripažįstamas negaliojančiu, jei $ver(x, y|K_v) = 0$.

Savybė (76) reiškia, kad tinkamai sudarytas parašas visada bus pripažįstamas. Tai būtina kiekvienos skaitmeninio parašo schemos savybė. Tačiau skaitmeninio parašo schema būtų bevertė, jeigu būtų nesunku klastoti parašus, t. y. jeigu iš viešojo rakto K_v būtų lengva nustatyti parašui sudaryti reikalingą privatųjį raktą K_p arba ir nežinant privataus rakto būtų galima parinkti poras $x \in \mathcal{M}, y \in \mathcal{P}$, tenkinančias lygybę $ver(x, y|K_v) = 1$.

Yra du būdai išvengti klastojimo. Viena vertus, galime tikrinimui skirtą raktą atskleisti tik tada, kai reikia patikrinti parašą ir daugiau jo nebenaudoti. Tokios skaitmeninių parašų schemos vadinamos vienkartinėmis. Arba galima naudoti viešojo rakto kriptosistemų konstrukcijas ir sudaryti skaitmeninio parašo schemą taip, kad, žinant K_v, praktiškai nebūtų įmanoma nustatyti K_p. Tada tuos pačius schemos raktus bus galima naudoti daugeliui dokumentų pasirašyti.



Skaitmeninio parašo schema

Apskritai skaitmeninio parašo schemą galime įsivaizduoti kaip kriptosistemą, kurioje raktai sukeisti vietomis: šifras (parašas) sudaromas naudojant privatųjį raktą, o šifras dešifruojamas (parašas tikrinamas) naudojant viešąjį raktą.

13.3. Algoritmai ir protokolai

> *Jeigu į savo butą įstatysite šarvuotas duris, bet ir toliau išeidami iš namų paliksite raktą po kilimėliu... patys spręskite, ar šarvuotos durys apsaugos jūsų turtą. Jeigu su draugu nutarėte susitikti dideliame svetimos šalies mieste, bet nesusitarėte dėl vietos – net ir detalus miesto žemėlapis nepadės. Panašiai ir su kriptografine apsauga: maža turėti saugius kriptografinius algoritmus, reikia gerai pagalvoti, kaip jie turi būti sujungti į visumą.*

Šifravimas, dešifravimas, parašo sudarymas, tikrinimas... – tai veiksmai, kuriuos gali atlikti vienas asmuo. Jis naudojasi patikimais įrankiais – kriptografiniais algoritmais. Tikrovėje niekas neatlieka šifravimo dėl paties šifravimo ir nekuria skaitmeninių parašų tiesiog šiaip sau (išskyrus galbūt tyrinėtojus, kuriems kriptografijos uždaviniai yra įdomesni ir svarbesni už visus jų rezultatų taikymus). Tikrovėje informacijos slaptumas ir autentiškumas yra svarbūs įvairių praktinių uždavinių sprendimo elementai. Tie tikrovės uždaviniai yra painūs ir sudėtingi, o jų konstruktyvus sprendimas (arba sprendimo sutrukdymas) gali rūpėti daugeliui. Pavyzdžiui, reikia naudojantis kompiuteriniais tinklais pasirašyti daugiašalę sutartį arba organizuoti šifruotą ryšį tarp daugelio dalyvių, neturint galimybės jiems saugiai perduoti

raktus arba organizuoti elektroninius rinkimus arba skaitmeninių pinigų sistemą...

Kad patikimi kriptografiniai įrankiai gali visiškai neatlikti savo paskirties, gerai parodo toks pavyzdys.

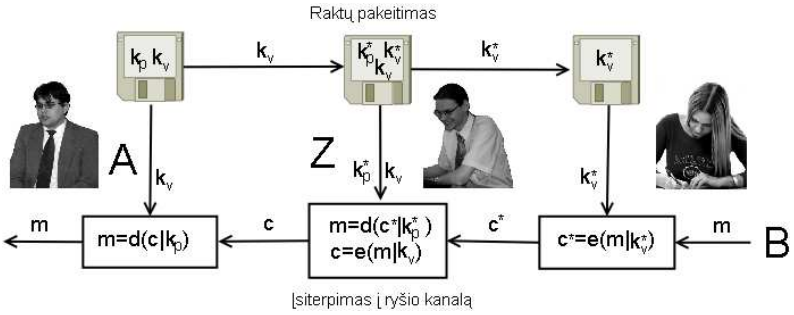
Tarkime, Algis ir Birutė nori palaikyti šifruotą ryšį, naudodamiesi kokia nors saugia viešojo rakto kriptosistema. Atrodo, kas gali būti paprasčiau: tegu A nusiunčia B šifravimui skirtą raktą $K_{e,A}$, B tegu nusiunčia raktą $K_{e,B}$ ir problema išspręsta! Galima saugiai susirašinėti. Tačiau prisiminkime Z, kurio galimybės kontroliuoti ryšio kanalą yra neribotos. Jis taip pat gali sudaryti tos pačios kriptosistemos raktų porą $K = \langle K_{e,Z}, K_{d,Z} \rangle$ ir pasielgti taip: perėmęs A siunčiamą raktą $K_{e,A}$ jis gali jį pakeisti savo raktu $K_{e,Z}$. Taigi užuot gavusi Algio raktą $K_{e,A}$, Birutė gaus $K_{e,Z}$. Galimybės nustatyti, kad raktas pakeistas nėra! Z taip pat gali pakeisti ir B siunčiamą raktą. Kai bus pasikeista raktais, A ir B bandys užmegzti „saugų“ ryšį. Tarkime, B ketina pasiųsti A šifruotą pranešimą M. Ji šifruoja: $C = e(M|K_{e,Z})$, ir siunčia A, bet pirmiausia laiškas patenka Z, kuris jį išsišifruoja ir vėl užšifravęs siunčia A:

$$M = d(C|K_{d,Z}), \quad C' = e(M|K_{e,A}).$$

A irgi išsišifruoja ir skaito laišką:

$$M = d(C'|K_{d,A}),$$

nė nenumanydamas, kad jis nebe pirmasis skaitytojas.



Įsiterpimo (man-in-the middle, angl.) ataka

Šis būdas įveikti kriptografinę apsaugą vadinamas „įsiterpimo“ (man in the middle, angl.) ataka. Zigmas net nebandė atakuoti patikimos kriptosistemos. Jis nelaužė šarvuotų durų, jis įlindo pro atvirą langą! Zigmas atliko ne kriptografinio algoritmo, bet naudojimosi juo taisyklių ataką.

Prieš pradėdami naudotis kriptosistema A ir B turėjo atlikti numatytus veiksmus, kurių tikslas – pasikeisti raktais.

Nustatytų veiksmų seka, kurią atlieka du ar daugiau asmenų, norėdami pasiekti tam tikrą tikslą, vadinama protokolu. Prekės įsigijimas, sąskaitos

atidarymas banke, įsidarbinimas, egzamino laikymas... – visur vieni protokolai.

Protokolas, kuriame, atliekant veiksmus, naudojami kriptografiniai algoritmai, yra kriptografinis protokolas. Kriptografiniai protokolai skiriasi nuo kasdienį gyvenimą reguliuojančių protokolų tuo, kad dažniausiai protokolo dalyviai neturi tiesioginio sąlyčio. Jūs negalite savo partneriui, su kuriuo norite naudodamiesi kompiuteriniais tinklais pasirašyti sutartį, tiesiog pažvelgti į akis ir, pasikliovę nuojauta, nuspręsti, ar jis neketina jūsų apgauti. Apgavystės galimybė turi būti paneigta pačioje protokolo struktūroje. Jis turi būti nedviprasmiškas ir išsamus, kita vertus, jį vykdydami, dalyviai neturėtų gauti vienas iš kito daugiau žinių, nei jų reikia numatytam tikslui pasiekti.

Kurti protokolus ir analizuoti jų patikimumą nėra lengva. Apskritai lengviau nustatyti, kad naudojamas protokolas nėra patikimas, nei įrodyti patikimumą. Juk saugumui paneigti pakanka nurodyti vieną spragą, o teigti, kad trūkumų nėra, nes jų nepastebėta – netikusi logika.

Kuriant protokolą, formuluojamas tikslas, kurį tuo protokolu siekiama pasiekti, nustatomi dalyviai, jų funkcijos ir tarpusavio ryšiai. Beveik visada reikia įvertinti grėsmes, kurios A ir B vykdomam protokolui kyla dėl Z egzistavimo. Kartais į protokolą įtraukiami tretieji pasiekti norimą tikslą padedantys asmenys.

Pavyzdžiui, jeigu A ir B gali pasinaudoti nesuinteresuoto ir patikimo trečiojo asmens J (Justo) pagalba, tai „įsiterpimo“ atakos pasikeičiant viešaisiais raktais galima išvengti. Tarkime, A ir B gali kreiptis į patikimą asmenį, turintį skaitmeninio parašo schemą. Tada A, norėdamas pasiųsti B savo viešąjį raktą ir norėdamas išvengti pakeitimo, gali pasiųsti J laišką su prašymu pasirašyti $K_{e,A}$. Gavęs iš J parašą ir jį patikrinęs

$$y = sig(K_{e,A}|K_{p,Z}), \quad ver(K_{e,A}, y|K_{v,Z}) = 1,$$

A gali siųsti B raktą $K_{e,A}$ kartu su parašu $y = sig(K_{e,A}|K_{d,Z})$ nebebijodamas, kad raktas bus pakeistas ir tai liks nepastebėta.

Tačiau trečiųjų asmenų pagalba gali brangiai kainuoti arba būti tiesiog neįmanoma. Geriausi protokolai yra tie, kurie įvairių apgavysčių galimybę panaikina dėl vidinės numatytų veiksmų struktūros.

13.4. Kriptosistemų saugumas

Vykdyti kriptografinio algoritmo kriptoanalizę reiškia atlikti jo atakas. Kokios tos atakos gali būti? Ką reiškia teiginys, kad kriptosistema yra saugi?

Teisėtas šifro C gavėjas dešifruoja jį, naudodamasis dešifravimui skirtu raktu K_d. Zigmas nežino šio rakto, todėl bando atkurti pradinį tekstą, naudodamasis šifru ir žiniomis apie naudojamą kriptosistemą. Kitaip tariant,

jis atlieka jos kriptoanalizę. Kuo jis gali naudotis? Žiniomis apie šifravimo ir dešifravimo operacijų struktūrą, kriptosistemos darbo duomenimis: šifrais ir juos atitinkančiais tekstais. Viskuo, išskyrus slaptuosius parametrus – raktus. Kriptoanalizės tikslas – naudojantis šifrais, atkurti juos atitinkančius tekstus, o galbūt – netgi nustatyti ir raktus.

Skaitmeninių parašų sistemos kriptoanalizė – tai būdų sudaryti galiojančius parašus nežinant privačiųjų raktų paieška, o taip pat – parašams sudaryti naudojamų raktų nustatymo galimybių analizuojant tekstus ir galiojančius jų parašus tyrimas.

Panagrinėkime kriptosistemų atakų rūšis. Jos skiriasi atakai naudojamų duomenų pobūdžiu. Tačiau visais atvejais daroma prielaida, kad kriptoanalitikas žino bendrąją kriptosistemos struktūrą: kaip veikia šifravimo ir dešifravimo algoritmai, kokia naudojamų raktų aibė.

Jeigu naudojamasi tik šifrais, tai ataka vadinama

- *pavienių šifrų ataka (ciphertext-only attack).*

Tokią ataką galime suformuluoti kaip uždavinį:

Duota: $C_1 = e(M_1|K_e), \ldots, C_i = e(M_i|K_e)$.

Rasti: arba $M_1, \ldots, M_i$ ir K_d, arba algoritmą, kuris bet kokiam $C_{i+1} = e(M_{i+1}|K_e)$ surastų M_{i+1}.

Jeigu pavyko gauti ne tik šifruotų, bet ir juos atitinkančių pradinių tekstų, kriptoanalitikas atlieka

- *teksto-šifro porų ataką (known-plaintext attack).*

Duota: $M_1, C_1 = e(M_1|K_e), \ldots, M_i, C_i = e(M_i|K_e)$.

Rasti: K_d arba algoritmą, kuris bet kokiam $C_{i+1} = e(M_{i+1}|K_e)$ nustatytų M_{i+1}.

Šifravimą mūsų laikais vykdo, žinoma, kompiuteriai. Gali būti, kad Zigmas turi galimybę pateikti šifravimo sistemai kokius nori tekstus ir gauti atitinkamus šifrus. Tada jis gali atlikti rimtą kriptosistemos išbandymą –

- *pasirinktų teksto-šifro porų ataką (chosen-plaintext attack).*

Duota: $M_1, C_1 = e(M_1|K_e), \ldots, M_i, C_i = e(M_i|K_e)$; čia pranešimus $M_1, \ldots, M_i$ pasirinko pats kriptoanalitikas.

Rasti: K_d arba algoritmą, kuris bet kokiam $C_{i+1} = e(M_{i+1}|K_e)$ nustatytų M_{i+1}.

Tačiau Zigmas gali turėti dar daugiau galimybių. Vieną kartą atlikęs pasirinktų teksto-šifro porų ataką, jis gali, atsižvelgdamas į kriptoanalizės rezultatus, pasirinkti naujus pranešimus ir vėl gauti jų šifrus. Ši ataka vadinama

- *adaptyvia pasirinktų teksto-šifro porų ataka (adaptive-chosen-plaintext attack).*

Kartais įmanoma ir tokia ataka: kriptoanalitikas pasirenka šifrus ir gauna juos atitinkančius pranešimus, tai –

- *pasirinktų šifrų ataka (chosen-ciphertext attack).*

Pavyzdžiui, jis gali įvairius pranešimus dešifruoti parašui tikrinti skirtu raktu ir bandyti gauti netiesioginės informacijos apie parašams sudaryti naudojamą raktą.

Kaipgi galima vertinti kriptosistemos saugumą? Yra keli požiūriai, ką laikyti saugia kriptosistema.

Kriptosistema vadinama **besąlygiškai saugia** (*unconditional security*), jei net ir turėdamas beribius skaičiavimo išteklius kriptoanalitikas negali be rakto iš šifro nustatyti, koks pranešimas buvo siųstas. Tai griežčiausias saugios kriptosistemos apibrėžimas.

Kriptosistema vadinama **saugia sudėtingumo teorijos požiūriu** (*complexity-theoretic security*), jei jos negali įveikti Zigmas, kurio skaičiavimo resursai leidžia jam taikyti tik polinominio laiko algoritmus (t. y. kai naudojamas laikas ir atmintis polinomiškai priklauso nuo įvedamų duomenų dydžio).

Sakoma, kad kriptosistemos **saugumas yra įrodomas** (*provable security*), jeigu galima įrodyti, kad sistemos įveikimas yra tolygus matematinio (dažniausiai skaičių teorijos) uždavinio, kuris laikomas sunkiu, sprendimui.

Kriptosistema vadinama **skaičiavimų požiūriu saugia** (*computational security*), jeigu pasiektas skaičiavimų resursų lygis yra pernelyg žemas, kad naudojant geriausias žinomas atakas, sistema būtų įveikta.

Pagaliau **ad hoc saugia**, arba euristiškai saugia, kriptosistema vadinama tokia sistema, kurios saugumą patvirtina tam tikri dažnai euristiniai argumentai. Suprantama, šis terminas tereiškia, kad specialistai atliko tam tikrą sistemos analizę, tačiau įveikti kriptosistemos nepavyko.

Įvertinti kriptosistemos saugumą – sudėtinga užduotis. Jokių bendrų receptų jai spręsti nėra. Kriptografo žinių, patirties ir talento niekas neatstos. Kriptografija yra modernus mokslas, tačiau, kaip ir senaisiais laikais, ir menas.

14 Klasikinė kriptografija

Kaip paslėpti teksto

$$M = m_1 m_2 \dots m_n, \quad \text{čia } m_i \in \mathcal{A} \text{ yra abėcėlės simboliai arba žodžiai,}$$

prasmę, išsaugant galimybę ją atkurti? Galima simbolius perstatyti vietomis, galima juos pakeisti kitais.

Tegu, pavyzdžiui, $\sigma : \{1, 2, \dots, n\} \to \{1, 2, \dots, n\}$ yra elementų perstata (abipus vienareikšmis atvaizdis). Naudodami ją kaip raktą, šifravimo operaciją galime apibrėžti taip:

$$e(M|\sigma) = m_{\sigma(1)} m_{\sigma(2)} \dots m_{\sigma(n)}.$$

Kitas būdas sudaryti šifrą – panaudoti keitinį. Jeigu $\mathcal{B}$ yra kokios nors kitos abėcėlės simbolių arba žodžių aibė ir $\lambda : \mathcal{A} \to \mathcal{B}$ yra injektyvus atvaizdis, tai, naudodamiesi juo kaip raktu, galime šifruoti taip:

$$e(M|\lambda) = \lambda(m_1)\lambda(m_2) \dots \lambda(m_n).$$

Galima teigti, kad kriptografijos istorija – tai tinkamų šifravimui perstatų ir keitinių konstravimo istorija.

Banaloka, žinoma, mintis! Reiškia maždaug tiek pat kaip teiginys, kad kompiuterių raida – tai tobulesnių skaičiavimo priemonių kūrimo istorija.

14.1. Skytalė ir kitos perstatos

Plutarchas savo „Įžymiųjų žmonių gyvenimuose“ rašydamas apie Spartos karaliaus Lisandro gyvenimą, mini, kad spartiečiai, norėdami jį parsikviesti iš karo žygių, pasiuntė skytalę. Skytalė – pirmasis istorijoje žinomas šifravimo įrenginys, kartu ir šifras.

Skytalė graikiškai reiškia lazdelę. Tačiau tuo pačiu žodžiu graikai vadino ir ilgą siaurą odinę juostelę su užrašytu tekstu. Rašydavo ant tokios juostelės taip: užvyniodavo ją ant lazdelės, kad sluoksniai būtų šalia vienas kito, ir rašydavo ant jos eilutė po eilutės vis pasukdami lazdelę. Užrašę visą tekstą, juostelę nuvyniodavo. Ant jos galėjai matyti pabiras raides, išdėstytas tarsi be jokios tvarkos. Jei skaitytum jas iš eilės – tik veltui gaištum laiką. Kad išryškėtų prasmė, juostelę reikia užvynioti ant tokio pat skersmens lazdelės.

Graikų istorikas Herodotas savo raštuose mini spartiečių šifravimo įrenginį – skytalę. Spartiečių skytalė – tai tiesiog lazdelė, ant kurios užvyniojama odos arba papiruso juostelė. Siaurą popieriaus juostelę užvynioję ant pieštuko, galite išbandyti, kaip toks prietaisas veikia.

Taigi skytalė yra šifravimo įrenginys. Siuntėjas ir gavėjas turi turėti po tokio pat skersmens lazdelę – tai šios šifravimo sistemos raktai. Naudodamasis raktu, siuntėjas išbarsto savo pranešimo raides išilgai juostelės – šifruoja pranešimą. Tekstas, kurį gautume nuosekliai skaitydami juostelėje užrašytas raides nuo pradžios iki galo, yra pranešimo šifras. Šifro gavėjas, naudodamasis savo raktu, vėl surenka šifro raides taip, kad išryškėtų siųstasis pranešimas – dešifruoja tekstą.

Ši šifravimo sistema sugalvota daugiau kaip prieš du tūkstančius metų. Tačiau ir dabartines šifravimo sistemas – jas pavadinome kriptosistemomis – sudaro tos pačios dalys: nešifruoti pranešimai, raktai ir šifrai, kurie gaunami, pagal tam tikras taisykles (algoritmus) pertvarkant pranešimus taip, kad nebūtų aiški jų prasmė. Panagrinėkime, kaipgi veikia skytalės kriptosistema. Visai nebūtina ieškoti lazdelės ir popieriaus juostos. Panagrinėję piešinį, greitai suprasime, kaip ji veikia.

M	Ū	S	Ų	Ž	E	M	Ė	J	A	U	N	A	D	A	R
B	U	S	I	R	P	A	S	M	U	S	E	L	D	O	R
A	D	O	A	L	D	O	R	I	J	O	A	D	R	I	J
O	A	D	A		K	A	Z	Y	S	B	I	N	K	I	S

Kai šifruojame naudodami skytalę, tarsi rašome tekstą į lentelę eilutė po eilutės, o šifrą sudarome skaitydami stulpelį po stulpelio: MBAOŪUDA...

Jeigu teksto raides rašysime į stačiakampės lentelės eilutes, o skaitysime stulpelis po stulpelio, tai gausime skytalės šifrą. Dešifruodami turime elgtis priešingai – šifro simbolius rašyti į stulpelius, o skaityti eilutę po eilutės.

Taigi skytalė apibrėžia tam tikrą teksto simbolių išbarstymo taisyklę, kurią galima paaiškinti naudojant lentelę. Tokią taisyklę nesunku apibendrinti. Pavyzdžiui, surašius tekstą į $m \times n$ matavimų lentelę (matricą) eilutė po eilutės, galima susitarti, kad šifras bus gaunamas perskaitant lentelę stulpelis po stulpelio tam tikra tvarka, kurią turi žinoti abu šifro vartotojai – šifruotojas ir dešifruotojas. Kriptografijos istorijoje buvo naudota įvairiausių tokios rūšies taisyklių. Šias operacijas galima matematiškai užrašyti pasinaudojant matricų veiksmais.

Štai pavyzdys. Tarkime, šifruojama taip: tekstas M surašomas į 3×4 matavimų matricą eilutė po eilutės, o tada skaitant stulpelis po stulpelio, surašoma į 4×3 matavimų matricą C, pirma perskaitant antrąjį, tada trečiąjį, paskui – ketvirtąjį ir pirmąjį stulpelius. Taigi tokio šifro raktas – stulpelių numerių seka $2 - 3 - 4 - 1$.

Tegu M yra pradinė teksto lentelė, C – šifras (4×3 matavimų lentelė), o K – 4×4 matavimų matrica, nusakanti M stulpelių skaitymo tvarką (šifro raktas):

$$K = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

Galime įsivaizduoti, kad teksto simboliai pakeisti sveikaisiais skaičiais, pavyzdžiui, simbolių eilės numeriais abėcėlėje. Tada

$$C = e(M|K) = K \cdot M^T,$$

čia M^T yra transponuota matrica; nulinio elemento ir simbolio sandauga laikoma lygia nuliui, o daugyba iš vieneto simbolio nekeičia.

Prisiminę matricų veiksmų savybes, galime ir dešifravimo operaciją užrašyti matematiškai:

$$M = d(C|K) = C^T (K^{-1})^T.$$

Žinoma, praeities laikų kriptografai matricų nedaugino. Pagrindiniai jų įrankiai – pieštukas ir popieriaus lapas, ant kurio rašydami raidžių virtines ieškodavo prasmės. Todėl senieji šifrai kriptografijos literatūroje dažnai ir vadinami „popieriaus ir pieštuko“ šifrais.

Štai dar vienas toks šifras, paremtas perstatomis. Jis nepelnytai vadinamas Fleissnerio kvadratų šifru, nes buvo aprašytas austrų armijos kapitono E. Fleissnerio (1843–1874) knygelėje. Tačiau iš tiesų jis buvo naudotas ir anksčiau.

Kapitono Fleissnerio idėja paprasta. Tarkime, mūsų pranešimą sudaro 16 simbolių. Jam užšifruoti pasigaminkime šešiolikos langelių kvadratą, išpjaukime jame keturis langelius (ne bet kaip!) ir prismeikime smeigtuku per centrą prie popieriaus lapo. Įrašę pirmąsias pranešimo raides į išpjautus lagelius, pasukime kvadratą 90° kampu prieš laikrodžio rodyklę. Jeigu langelius tinkamai išpjovėme, jie atidengs tuščias popieriaus vietas. Į jas vėl įrašykime keturias raides ir vėl pasukime kvadratą 90° kampu prieš laikrodžio rodyklę. Pasukę kvadratą paskutinį kartą, įrašysime likusias keturias pranešimo raides. Nuėmę kvadratą, ant popieriaus pamatysime į kvadratą surašytas pranešimo raides. Tai ir yra šifras. Galime jį pasiųsti nurašę raides eilutė po eilutės. Kad gavėjas galėtų iššifruoti šifrą, jis privalo turėti tokį pat kvadratą, kokiu naudojosi šifruotojas.

			L
A			
	U		
		K	

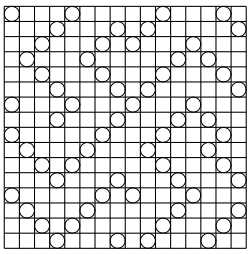
S			L
A			I
	U	M	
	E	K	

S	N		L
A		A	I
	U	M	U
J	E	K	

S	N	I	L
A	E	A	I
N	U	M	U
J	E	K	Ų

Perskaitę „užpildyto“ kvadrato eilutes, gauname tokį šifrą: SNILAEAI...

Žinoma, tokiam šifravimui galime naudoti ir didesnį kvadratą. Kiekviename $2n \times 2n$ langelių kvadrate galime taip išpjauti n langelių, kad, tris kartus $90°$ kampu pasukus kvadratą apie centrą, atsidengtų vis naujos popieriaus lapo vietos.



XVIII amžiuje naudoto šifro šablonas.

Nors šiam šifravimo būdui prigijo Fleissnerio vardas, idėja pernelyg paprasta, kad nebūtų kilusi ir kitiems. Dviem olandų tyrinėtojams pavyko iššifruoti XVIII amžiaus vidurio šifrą, rastą Olandijos valdytojo archyvuose; iššifruoti pavyko sudarius 16×16 dydžio Fleissnerio kvadratą, pavaizduotą brėžinyje. Taigi Fleissnerio sugalvotas šifras jau buvo naudotas šimtmečiu anksčiau![6]

Perstatas naudojantys šifravimo algoritmai nekeičia raidžių dažnių: ir nešifruotame tekste, ir jo šifre tos pačios raidės pasitaiko vienodai dažnai. Taigi sudarius pakankamai ilgo šifro raidžių dažnių lentelę ir lyginant ją su įvairių kalbų raidžių dažnių lentelėmis, galima ne tik nustatyti, kad šifravimui naudota perstata, bet ir sužinoti, kokia kalba parašytas pradinis nešifruotas tekstas.

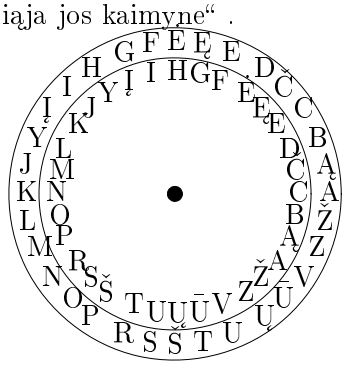
Nors perstatų naudojimas šifravimui labai sena idėja, ji praverčia ir šiandien konstruojant modernius šifrus. Juk kompiuteriai gali perstatų operacijas atlikti labai greitai, be to – taikyti jas milžiniško dydžio duomenų kiekiams.

[6] Karl de Leuw, Hans van der Meer. A Turning Grille from the Ancestral Castle of the Dutch Stadtholders. Cryptologia, XIX, 2, 1995, p. 153–165.

14.2. Keitiniai ir šifrai

Iš romėnų rašytojo Svetonijaus, gyvenusio maždaug 70–150 m., veikalo „Dvylikos cezarių gyvenimas“ žinome, kaip Julijus Cezaris rašė šifruotus laiškus Ciceronui...

Paprastą Julijaus Cezario šifravimo būdą galima nusakyti labai paprastai: „keisk abėcėlės raidę trečiąja jos kaimyne“ .



Cezario šifras su lietuviška abėcėle. Pagal skritulių kraštus viena po kitos išrašytos visos lietuvių kalbos abėcėlės raidės. Po didžiojo skritulio abėcėlės raidėmis antrajame skritulyje surašytos raidės, kurios abėcėlėje stovi trimis pozicijomis toliau – “ trečiosios kaimynės“ . Paskutiniųjų abėcėlės raidžių kaimynės yra pirmosios abėcėlės raidės.

Tarkime, didesnysis skritulys nejuda, o mažesnysis sukiojasi apie bendrą abiejų skritulių ašį. Galime pasukti mažąjį skritulį, kad po didžiojo skritulio raide A atsidurtų kita mažojo skritulio raidė. Gausime naują Cezario šifro variantą. Kiek yra abėcėlės raidžių, tiek yra mažojo skritulio padėčių. Kiekvieną iš jų galime nusakyti skaičiumi padalų, kuriuo prieš laikrodžio rodyklę pasuktas mažasis skritulys. Taigi gauname n šifravimo algoritmų, kurių raktai yra aibės

$$\mathcal{K} = \{0, 1, \ldots, n-1\}$$

skaičiai, čia n yra raidžių skaičius abėcėlėje.

Cezario šifravimo būdą patogu apibrėžti matematiškai, pakeitus abėcėlės raides skaičiais.

Tegu $\mathcal{A} = \mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ yra n raidžių turinti abėcėlė, pranešimų ir šifrų aibės sudarytos iš šios abėcėlės žodžių $\mathcal{M} = \mathcal{C} = \mathcal{A}^*$, o raktų aibė sutampa su abėcėle $\mathcal{K} = \mathcal{A}$. Tada Cezario šifravimo taisyklę galime užrašyti taip:

$$e(x|k) \equiv x + k \pmod{n},\ x \in \mathbb{Z}_n, \quad e(m_1 m_2 \ldots |k) = e(m_1|k)e(m_2|k)\ldots$$

Ši paprasta kriptosistema kriptografijoje vadinama Cezario kriptosistema. Raktų yra tiek pat kiek abėcėlės simbolių. Matematinę šifravimo algoritmo išraišką norisi apibendrinti. Imkime tokią raktų aibę:

$$\mathcal{K} = \{K = \langle k_1, k_2 \rangle : k_i \in \mathbb{Z}_n, (k_1, n) = 1\},$$

ir vieno simbolio šifrą apibrėžkime taip:

$$e(x|K) \equiv k_1 x + k_2 \pmod{n}.$$

Jeigu n yra pirminis, tai tokioje apibendrintoje Cezario kriptosistemoje bus $n \cdot (n-1)$ skirtingų raktų. Jie „valdo“ tik nedidelę dalį visų galimų keitinių $\mathbb{Z}_n \to \mathbb{Z}_n$; iš viso skirtingų keitinių yra $n!$

Cezario kriptosistemos apibendrinimu galime laikyti Lesterio Hillo šifrus, kuriuos jis paskelbė 1929 metais.[7]

Apibrėžkime Hillo kriptosistemos raktų aibę:

$$\mathcal{K}_{n,r} = \{K : K = (k_{ij}) \text{ yra } r\text{-osios eilės kvadratinė matrica } (\det(K), n) = 1\},$$

čia visi skaičia k_{ij} yra sveikieji, o $\det(K)$ žymi matricos determinantą. Dabar r ilgio žodžių šifrus galime apibrėžti taip:

$$e(m_1 m_2 \dots m_r|K) = c_1 c_2 \dots c_r, \quad c_1 c_2 \dots c_r = m_1 m_2 \dots m_r \cdot K,$$

o dešifravimo operaciją

$$d(c_1 c_2 \dots c_r|K) = m_1 m_2 \dots m_r, \quad m_1 m_2 \dots m_r = c_1 c_2 \dots c_r \cdot K^{-1}.$$

Atvirkštinę matricą K^{-1} reikia skaičiuoti, žinoma, moduliu n, o žodis daugi-namas iš matricos naudojantis vektoriaus ir matricos sandaugos apibrėžimu.

Jei $n = p$ yra pirminis skaičius, tai raktų aibėje $\mathcal{K}_{n,r}$ yra tiek matricų, kiek tiesinė erdvė $\mathbb{F}_p^r$ turi skirtingų bazių. Šį dydį nesunku apskaičiuoti.

115 teorema. *Jei p yra pirminis, tai*

$$|\mathcal{K}_{p,r}| = (p^r - 1)(p^r - p)(p^r - p^2) \cdots (p^r - p^{r-1}).$$

Jeigu šifravimas naudojant keitinį iš pradžių apibrėžiamas pavieniams abėcėlės simboliams, o šifruojant žodžius, jis taikomas kiekvienai raidei atskirai, tai pradinio teksto simbolių dažniai lygūs atitinkamiems šifro simbolių dažniams, taip pat – pradinio teksto simbolių porų dažniai lygūs atitinkamiems šifro simbolių porų dažniams ir t. t. Taigi naudojantis įvairių kalbų simbolių, jų porų, trejetų ir t. t. dažnių lentelėmis, iš šifro galima nustatyti ir kalbą, kuria buvo parašytas tekstas, ir patį keitinį.

Jeigu šifruojame (kaip Hillo šifro atveju) ne pavienius simbolius, bet m ilgio žodžius, tai ryšio tarp pavienių simbolių dažnių tekste ir šifre jau nebus, tačiau $m, 2m, \dots$ ilgio simbolių blokų dažniai bus tie patys. Žinoma, kai m yra didelis skaičius, tuos dažnius skaičiuoti ir lyginti yra sudėtinga.

Taigi kai šifruojami ne pavieniai žodžiai, bet m $(m > 1)$ ilgio žodžiai, ryšys tarp simbolių dažnių nešifruotame tekste ir šifre išnyksta. Yra dar viena keitinių, panaikinančių ryšį tarp simbolių dažnių, rūšis.

[7] L. Hill. Cryptography in an Algebraic Alphabet. The American Mathematical Monthly, **36**, 1929, June-July, p. 306–312.

Tegu $\mathcal{A}$ ir $\mathcal{B}$ yra nešifruoto teksto ir šifro abėcėlės. Abėcėlės $\mathcal{B}$ visų poaibių aibę žymėsime $\mathcal{P}(\mathcal{B})$.

Kiekvienam atviro teksto abėcėlės simboliui $a \in \mathcal{A}$ nurodysime homofonų aibę $k(a) \subset \mathcal{B}$, t. y. nurodysime skirtingai užrašomus simbolius, kuriuos šifre reikia „skaityti“ kaip a. Tokios kriptosistemos raktas yra funkcija

$$k : \mathcal{A} \to \mathcal{P}(\mathcal{B}),\ k(x) \cap k(y) = \emptyset,\ \text{jei } x \neq y,\ x, y \in \mathcal{A}.$$

Šifravimui dar reikia turėti būdą, kuriuo naudojantis, iš šifruojamo simbolio homofono būtų atsitiktinai parenkamas vienas jo elementas; visų elementų parinkimo tikimybės turėtų būti vienodos. Žymėdami atsitiktinai parinktą homofono $k(a)$ elementą $k(a, \omega)$, teksto šifrą apibrėšime taip:

$$e(m_1 m_2 \ldots | k) = k(m_1, \omega) k(m_2, \omega) \ldots$$

Naudodami homofoniniais keitiniais grindžiamus šifrus, galime panaikinti ryšį tarp atviro teksto simbolių dažnių lentelės bei šifro simbolių dažnių lentelės. Parinkdami funkciją k taip, kad homofono $k(a)$ elementų skaičius būtų proporcingas simbolio a dažniui atvirame tekste, galime pasiekti, kad šifruotame tekste visų simbolių pasirodymo dažniai būtų beveik vienodi.

Jei kriptosistemoje norime naudoti homofoninį keitinį, turime nuspręsti, kaip apibrėžti raktą, kad jį būtų patogu saugoti bei lengva juo naudotis. Manoma, kad pirmąkart homofoniniai keitiniai šifravimui buvo panaudoti 1401 metais slaptam Mantujos hercogystės ir Simeone'o de Crema susirašinėjimui. Priebalsiams buvo naudoti įprastiniai keitiniai, tačiau balsiams šifruoti – homofoniniai šifrai.

Įdomus homofoninių keitinių pavyzdys – Beale'o šifrai. Thomas Jeffersonas Beale'as buvo nuotykių ieškotojas. Jis paliko tris šifruotus tekstus apie Virdžinijos valstijos apylinkėse užkastą didelį lobį. Tai įvyko apytikriai 1820 metais. Antrąjį Beale'o šifrą, kuriame aprašomas lobis ir nurodoma, kad pirmajame šifre yra detalios instrukcijos, kaip jį surasti, 1880 metais iššifravo Jamesas Wardas. Beale'o raktas – JAV Nepriklausomybės deklaracija. Jis naudojosi šiuo tekstu taip. Įsivaizduokime, kad kokio nors dokumento ar knygos (Beale'o atveju – JAV Nepriklausomybės deklaracijos) žodžius sunumeravome. Žodžių, kurie prasideda raide „a“ numeriai tegu sudaro šios raidės homofonų aibę, žodžių, kurie prasideda raide „b“ numeriai sudaro „b“ homofonų aibę ir t. t. Jeigu norite – galite numeruoti ne žodžius, bet raides.

Puikus metodas!

14.3. Vigenere šifras

Šifras, naudojantis keitinį, keičia pradinio teksto abėcėlės $\mathcal{A}$ raides abėcėlės $\mathcal{B}$ raidėmis. Panaudoti šifrui ne vieną, bet kelias abėcėles $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_d$ – pati geriausia naujosios Europos kriptografijos idėja!

Viduramžių europiečiams kriptografija nerūpėjo. Tačiau prasidėjus naujiesiems laikams, suklestėjus pirmiausia Italijos miestams, kurie ėmė įnirtingai varžytis dėl įtakos, valdžios, naudos ir turto, atėjo metas ir politinei Europos kriptografijai. Italijos miestų valdžia turėjo savo žinioje šifrų specialistus, kurie triūsė šifruodami ir dešifruodami slaptus laiškus, tiek savo miesto, tiek svetimus – pagrobtus, užverbuotų agentų perduotus... Pavyzdžiui, Venecijoje XVI amžiaus viduryje buvo net trys šifrų sekretoriai – visas kriptografų skyrius! Kriptologijos meno buvo mokoma mokyklose, buvo rengiami kriptoanalizės konkursai, už nuopelnus dosniai atlyginama.

1555 metais kriptografo (sekretoriaus šifrų reikalams) pareigybę įsteigė ir popiežius. Apie 1580 metus popiežiams ėmė tarnauti įžymių kriptografų Argenti šeima.

Tuo laiku naudotus pranešimų prasmės slėpimo būdus galima suskirstyti į dvi grupes: kodus ir šifrus. Kodas reiškė išankstinį susitarimą keisti vienus žodžius ar frazes kitais žodžiais, frazėmis arba simboliais. Pavyzdžiui, vienname seniausių Vatikano tekstų, skirtų kriptografijai, nurodoma, kad tuometinėse popiežiaus kovose su Romos imperatoriumi dalyvavusių gelfų ir gibelinų partijas reikia vadinti egiptiečiais ir Izraelio vaikais, minint karalių, rašyti raidę A, popiežių – raidę D ir t. t. Kad gavėjas perskaitytų naudojant kodą parašytą laišką, jam turi būti iš anksto įteiktas pats kodas – naudojamų žodžių ir tikrosios jų prasmės atitikties lentelė.

Kitaip pranešimo prasmė slepiama naudojant šifrus. Iš anksto susitariama, kaip bus keičiamos laiško raidės. Jos gali būti keičiamos kitomis raidėmis, kokiais nors simboliais arba tiesiog skaičiais. Laiško gavėjas turi žinoti dviejų simbolių rinkinių atitikties taisyklę, kuria naudojamasi keičiant raides, t. y. raktą. Tokiam raktui perduoti Argenti pradėjo naudoti prasmingus žodžius (juos lengviau įsiminti!). Argenti idėja parodyta pavyzdžiu.

A	R	G	E	N	T	I	B	C	D	F	H	J
10	11	12	13	14	15	16	17	18	19	20	21	22
K	L	M	O	P	Q	S	U	V	W	X	Y	Z
23	24	25	26	27	28	29	30	31	32	33	33	34

Pirmojoje ir trečiojoje eilutėse užrašyta lotyniška abėcėlė, tačiau ne įprastine tvarka. Ji prasideda rakto žodžiu ARGENTI, o po jo surašytos likusios abėcėlės raidės eilės tvarka. Antrojoje ir ketvirtojoje eilutėse užrašyti skaičiai, kuriais šifruojant keičiamos abėcėlės raidės. Ši idėja buvo naudojama net ir po kelių šimtų metų. Štai Richardo Zorgės – žvalgo, Antrojo pasaulinio karo išvakarėse veikusio Japonijoje, – šifro pavyzdys.

S	U	B	W	A	Y
0	82	87	91	5	97
C	D	E	F	G	H
80	83	3	92	95	98
I	J	K	L	M	N
1	84	88	93	96	7
O	P	Q	R	T	V
2	85	89	4	6	99
X	Z	.	/		
81	86	90	94		

Anglų kalbos abėcėlė surašyta lentelės eilutėse pagal Argenti metodą, pradedant žodžiu SUBWAY. Frazės „a sin to er“ raidės pakeistos skaitmenimis, o likusios – dviženkliais skaičiais, pradedant 80. Taip padaryta norint pagreitinti šifro perdavimą: frazės „a sin to er“ raidės anglų kalbos tekstuose pasitaiko dažniausiai.

Apie 1466 metus Leonas Batista Alberti – žymus italų architektas, architektūros teoretikas, muzikantas... tiesiog tikras Renesanso laikų žmogus – parašė rankraštį apie kriptografiją, kurioje išdėstė itin svarbią tolimesnei europiečių kriptografijos raidai idėją.

Panašius šifravimo skritulius aprašė Leonas Batista Alberti (1404–1472) savo traktate, skirtame šiframs. Mažesnysis skritulys yra sukiojamas, taigi šifro abėcėlė yra kaitaliojama.

Prisiminkime du skritulius su abėcėlės raidėmis, kuriais naudojomės nagrinėdami Cezario šifrus. Tie skrituliai taip pat yra Alberti išradimas, aprašytas jo kriptografijos traktate. Alberti skrituliuose, žinoma, buvo užrašytos lotyniškos abėcėlės raidės; jei surašysime lietuviškos abėcėlės raides – esmė dėl to nepasikeis.

Tarkime, didesnysis skritulys nejuda, o mažesnysis gali sukiotis. Skritulių tarpusavio padėtis apibrėžia laiško raidžių keitimo (šifravimo) taisyklę: laiško raidę suradę didesniajame skritulyje, ją pakeičiame raide, kuri užrašyta po ja mažesniajame skritulyje. Naujiena yra toks Alberti pasiūlymas: „Užšifravę du ar tris laiško žodžius, mažąjį skritulį pasukime per vieną padalą ir kitus du ar tris žodžius užšifruokime naudodamiesi naująja skritulių padėtimi ir vėl pasukime mažąjį skritulį...“ Šifruojant šitokiu būdu, pasikartojančios laiško raidės jau nebėra keičiamos viena ir ta pačia raide. Kokia raide reikia pakeisti, tarkime, raidę A, priklauso nuo skritulių tarpusavio padėties. Taigi šifruojant naudojama nebe viena, bet daugelis abėcėlių. Suprantama, kad laiško siuntėjas ir gavėjas turi turėti vienodus Alberti skritulius ir būti susitarę dėl pradinės skritulių padėties ir mažojo skritulio sukiojimo taisyklės. Kaip nusakyti tokias taisykles, Alberti nenagrinėjo. Šis nedidelis veikalas, galėjęs iš esmės pakeisti visą tuometinę kriptografiją, liko neišspausdintas ir didelės įtakos nepadarė. Matyt, šifravimo su daugeliu abėcėlių idėja buvo pernelyg nauja tiems laikams.

Panaši idėja vėl išniro maždaug po šešiasdešimt metų. Ji kilo ne mažiau talentingam, tačiau visai kito būdo žmogui. Alberti mėgo pasaulietiško gyvenimo spalvas, o štai kitas Europos kriptografijos pradininkas labiau vertino vienuolyno celės ramybę ir tylą. Johannes Trithemius (1462–1516) kriptografijos istorijoje minimas kaip pirmosios išspausdintos knygos apie šifrus autorius. Daugiau kaip penkių šimtų puslapių jo knyga „Poligraphia“ buvo išleista daugelį kartų. Joje buvo aprašyta ir panaši kaip Alberti šifravimo su daugeliu abėcėlių idėja.

Trithemius šifruoti siūlė taip: pirmąją laiško raidę su pirmąja abėcėle, antrąją – su antrąja ir t.t. Taigi ir šiuo būdu šifruojant, pasikartojančios laiško raidės keičiamos įvairiai priklausomai nuo jų vietos laiške.

Būtų teisinga šifravimą naudojant daugelį abėcėlių pavadinti Alberti arba Tritemijaus vardu. Tačiau istorija susiklostė taip, kad tokiam šifrui prigijo visai kito žmogaus vardas. 1586 metais buvo išspausdinta Blezo de Vigenere knyga „Traktatas apie šifrus“ , kuriame taip pat buvo išdėstytas šifravimo su daugeliu abėcėlių būdas. Vigenere vardas kriptografijoje prigijo šifrui, kuriame naudojama keletas abėcėlių, o šifro raktas užrašomas žodžiu.

Pasirinkime kokį nors žodį, pavyzdžiui, MES. Panaudosime jį kaip Vigenere šifro raktą sakiniui „UPELĖ MUSĖ RANGOSI VIKRIAI“ .

Pasinaudosime lotyniškosios abėcėlės lentele, todėl raidę Ė keisime į E.

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
G H I J K L M N O P Q R S T U V W X Y Z A B C D E F
H I J K L M N O P Q R S T U V W X Y Z A B C D E F G
I J K L M N O P Q R S T U V W X Y Z A B C D E F G H
J K L M N O P Q R S T U V W X Y Z A B C D E F G H I
K L M N O P Q R S T U V W X Y Z A B C D E F G H I J
L M N O P Q R S T U V W X Y Z A B C D E F G H I J K
M N O P Q R S T U V W X Y Z A B C D E F G H I J K L
N O P Q R S T U V W X Y Z A B C D E F G H I J K L M
O P Q R S T U V W X Y Z A B C D E F G H I J K L M N
P Q R S T U V W X Y Z A B C D E F G H I J K L M N O
Q R S T U V W X Y Z A B C D E F G H I J K L M N O P
R S T U V W X Y Z A B C D E F G H I J K L M N O P Q
S T U V W X Y Z A B C D E F G H I J K L M N O P Q R
T U V W X Y Z A B C D E F G H I J K L M N O P Q R S
U V W X Y Z A B C D E F G H I J K L M N O P Q R S T
V W X Y Z A B C D E F G H I J K L M N O P Q R S T U
W X Y Z A B C D E F G H I J K L M N O P Q R S T U V
X Y Z A B C D E F G H I J K L M N O P Q R S T U V W
Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
Z A B C D E F G H I J K L M N O P Q R S T U V W X Y
```

Lentelę, kurioje surašytos atitinkamai paslinktos lotynų kalbos abėcėlės Tritemijus vadino „tabula recta“ .

Parašykime po šio sakinio raidėmis rakto raides taip:

```
UPELE MUSE RANGOSI VIKRIAI
MESME SMES MESMESM ESMESME
```

Šifruojame pirmąją raidę U: ieškome lentelės stulpelio, kuris prasideda raide U, ir eilutės, kuri prasideda rakto raide M; eilutės ir stulpelio sankirtoje parašyta raidė G, ja ir keičiame raidę U. Šifruodami sakinio raidę P, ieškome stulpelio, prasidedančio raide P, ir eilutės, prasidedančios raide E, jų sankirtoje yra raidė T, šifruodami raidę, naudojame lentelės eilutę, kuri prasideda raide S ir t. t. Taip gauname ir visą sakinio šifrą:

```
UPELE MUSE RANGOSI VIKRIAI
MESME SMES MESMESM ESMESME
GTWXI EGWW DEFSSLU ZAWVAMM
```

Šifravimui panaudojome tik tris „tabula recta“ eilutes – tiek, kiek yra rakto žodžio raidžių. Dešifruojant Vigenere šifrą, taip pat prireiks trijų eilučių (trijų abėcėlių). Šifruodami pirmąją raidę, naudojome taisyklę, kuri nusakoma pirmąja lentelės eilute ir eilute, prasidedančia raide M, t. y. raidė

A yra keičiama raide M. Dešifruodami pirmąją raidę, naudosime taisyklę, kuri raidę M keičia raide A. Ši taisyklė nusakoma pirmąja lentelės eilute ir eilute, kuri prasideda raide O.

Antrajai ir trečiajai šifro raidėms dešifruoti teks naudoti lentelės eilutes, prasidedančias raidėmis W ir I. Taigi dešifruodami galime elgtis panašiai kaip šifruodami, tik vietoj rakto MES turime naudoti raktą OWI:

```
GTWXI EGWW DEFSSLU ZAWVAMM
OWIOW IOWI OWIOWIO WIOWIOW
UPELE MUSE RANGOSI VIKRIAI
```

Vigenere kriptosistemą matematiškai galime apibrėžti visai panašiai kaip Cezario. Tegu $\mathcal{A} = \mathbb{Z}_n = \{0, 1, 2, \ldots, n-1\}$, $\mathcal{M} = \mathcal{C} = \mathcal{A}^*, \mathcal{K} = \mathcal{A}^d$. Šifravimo algoritmas su raktu $k = k_1 k_2 \ldots k_d$ veikia taip:

$$e(m_1 m_2 \ldots m_d m_{d+1} \ldots | k) = c_1 c_2 \ldots c_d c_{d+1} \ldots,$$
$$c_1 \equiv m_1 + k_1 \pmod{n},\ c_2 \equiv m_2 + k_2 \pmod{n},\ \ldots, c_d \equiv m_d + k_d \pmod{n},$$
$$c_{d+1} \equiv m_{d+1} + k_1 \pmod{n},\ \ldots$$

Taigi galime įsivaizduoti, kad Vigenere šifras gaunamas suskaidžius pradinį tekstą į d fragmentų (d – naudojamo rakto ilgis) ir kiekvieną fragmentą užšifravus atskiru Cezario šifru.

14.4. Vigenere šifro analizė

Vigenere šifras ilgai buvo laikomas tiesiog neįveikiamu. Tačiau kai šifro raktas trumpas, visai paprasta idėja padeda jį įveikti.

Šifrai su keletu abėcėlių iš pradžių buvo naudoti retai. Alberti, Tritemijaus ir Vigenere laikais slaptaraščiuose vyravo kodai. Šifrai su daugeliu abėcėlių buvo pernelyg sudėtingi kasdieniam naudojimui. Vartyti kodų knygas ir užrašinėti žodžius to meto slaptų laiškų rašytojams ir skaitytojams atrodė paprasčiau.

Tačiau tie, kas išmanė, kaip Vigenere šifrai veikia, pripažino, kad jie saugūs, netgi neįveikiami. Net XX amžiaus pradžioje buvo manoma, kad jeigu raktas nežinomas, tai nėra būdo jiems įminti. Tai buvo netiesa. Nedidelėje 1863 metais išleistoje knygelėje prūsų armijos majoras Kassiskis išdėstė itin paprastą Vigenere šifro analizės metodą, kuriuo dažnai įmanoma iššifruoti šifrą ir be rakto. Tačiau Kassiskio idėją mažai kas įvertino. Pats Kassiskis irgi nesiekė žymaus kriptologo šlovės. Tarnaudamas Prūsijos armijoje, laisvalaikiu gilinosi į kriptografiją, tačiau, parašęs minėtą 95 puslapių knygelę *„Die Geheimschriften und die Dechiffrir-kunst“*, nustojo ja domėtis ir, išėjęs į atsargą, atsidėjo archeologijai.

Kokį gi Vigenere šifro analizės būdą sugalvojo Kassiskis?

Tarkime, Vigenere raktą sudaro d skirtingų simbolių. Tada pirmąją teksto raidę šifruojame naudodami vieną abėcėlę („tabula recta“ eilutę),

pažymėkime ją A_1, antrąją raidę – naudodami abėcėlę A_2, ... d-ąją – naudodami abėcėlę A_d, $d+1$-ąją – naudodami abėcėlę A_1 ir t. t. Taigi galime įsivaizduoti, kad tekstą $M = m_1 m_2 \ldots$ sudaro d dalių $M_1, M_2, \ldots, M_d$

$$\begin{aligned} M_1 &= m_1 m_{1+d} m_{1+2d} \ldots \\ M_2 &= m_2 m_{2+d} m_{2+2d} \ldots \\ &\ldots\ldots \\ M_d &= m_d m_{2d} m_{3d} \ldots, \end{aligned}$$

o kiekviena dalis šifruojama naudojantis atskira abėcėle.

Kriptoanalitikas nežino nei rakto, nei jį sudarančių simbolių skaičiaus (rakto ilgio), tačiau turi ilgoką šifrą $C = c_1 c_2 \ldots$

Jeigu jis žinotų rakto ilgį d, galėtų turimą šifrą suskaidyti į d fragmentų

$$\begin{aligned} C_1 &= c_1 c_{1+d} c_{1+2d} \ldots \\ C_2 &= c_2 c_{2+d} c_{2+2d} \ldots \\ &\ldots\ldots \\ C_d &= c_d c_{2d} c_{3d} \ldots \end{aligned}$$

ir, naudodamasis raidžių dažnių lentelėmis, dešifruotų kiekvieną paprastu keitinių šifru užšifruotą fragmentą.

Taigi raktas, kuris „atrakina“ šifro raktą, yra jo ilgis! Kaip jo ieškoti? Į šį klausimą Kassiskis atsakė labai paprastai: ieškokite šifre pasikartojančių fragmentų!

Patyrinėkime paprastą šiek tiek dirbtinį pavyzdį. Šifruosime tekstą Vigenere šifru, kurio raktas yra šešių raidžių žodis LAUKAS:

```
PAVARGĘS VASARIS PUSNYNUOSE MIEGA IR VANDENYS SLŪGSO UŽŠALĘ
LAUKASLA UKASLAU KASLAUKASL AUKAS LA UKASLAUK ASLAUK ASLAUK
```

Žvilgtelėkime į pirmąjį ir antrąjį šifruojamo teksto žodžius. Juose yra tas pats skiemuo VA, o po juo abiejuose žodžiuose parašytos tos pačios rakto raidės UK. Vadinasi, ir šifre gausime du vienodus iš dviejų raidžių sudarytus fragmentus. Atstumas tarp jų lygus 6, t. y. rakto ilgiui! Taigi kriptoanalitikas, pastebėjęs šifre šiuos vienodus fragmentus ir suradęs, kiek raidžių juos skiria, sužinotų rakto ilgį! Žinoma, analizuodami tikrą šifrą, tokios lengvos sėkmės negalėtume tikėtis. Rastume daug vienodų šifro fragmentų, atstumai tarp jų irgi būtų įvairūs. Tačiau vis tiek gana dažnai pasitaikytų rakto ilgio kartotiniai!

Panagrinėkime pavyzdį. Užšifravę pačią pirmąją šio skyrelio skiltį Vigenere šifru su raktu VIETA (naudojama lietuviška 32 simbolių abėcėlė), gausime tokį šifrą:

PŠJJA FCAFE YMŽNA ŽOFĄL FĘMLP NIHŠI ŠYARO KIAZO RŠŪŽT VŠEGB
CBŽDT NŠŽŽM FŲENS FBCDG CŽIJE YIMFA FCVGA MDEĖA JMVNS FBELI
KOPDM CLUHI KICĮK LLEDŠ FPŪTI OEHTU ĘMSDU VYYŪĖ YŠĄŲU UAŲŽR
KMSĘG OEHĄT FŽKDK VCHDE KŠEHN VEHĮJ FZADV VBŽĘT FŪUZŲ ĮŽOCA
OŠŪNŽ NIZDN DDMŠO BHMNS RAŠŽT LCSTP RĘSTI PŪĄJA PUŽĮJ VZVDR
OŪEDT HDUĖA JCEMR LLYYA MBEKČ FIA

Suradę vienodas simbolių poras (digramas) šifre, nustatę atstumus tarp jų, o tada suskaičiavę, kiek atstumų dalijasi atitinkamai iš $2, 3, \ldots, 10$, sudarytume tokią lentelę:[8]

Dalikliai	2	3	4	5	6	7	8	9	10
Kiek atstumų dalijasi	27	23	10	30	15	1	6	7	10

Skaičiai iškalbingi – rakto ilgis be abejonės lygus 5.

14.5. Friedmano kappa testas

> *Neretai senųjų amžių matematikai garsėjo ir kaip geri kriptografai: F. Viète, J. Walis... Tačiau senųjų laikų kriptografija buvo veikiau menas, nei mokslas. Bet XX amžiuje jos ryšiai su matematika darėsi vis glaudesni ir glaudesni...*

Kassiskio testą vargu ar pavadintume matematiniu kriptoanalizės metodu. Matematikos jame tiek ir tėra – atstumų tarp fragmentų skaičiavimas ir daliklių nagrinėjimas.

Pirmasis matematinius metodus kriptoanalizei pradėjo taikyti Williamas Friedmanas (1891–1969). Jį kriptografijos istorikai pelnytai vadina vienu didžiausių visų laikų kriptoanalitikų. Jo žmona – Elizebeth Friedman irgi buvo žymi kriptografė. Kriptografija šie žmonės susidomėjo atlikdami gana keistus tyrimus turtingo amerikiečio Fabyano įkurtame Riverbanko tyrimų centre. Jie tyrinėjo, ar didžiojo Šekspyro kūrinių autorius kartais nėra tų laikų žymus filosofas Backonas.

JAV įsitraukus į Pirmąjį pasaulinį karą prireikė kriptoanalitikų pagalbos. Tokių specialistų JAV kariniuose sluoksniuose nebuvo. Užtat jų buvo Riverbanke. Šitaip iš Šekspyro raštų tyrinėtojo W. Friedmanas virto vienu žymiausių JAV kriptografijos specialistų.

Kriptoanalizės metodą, kurį šiame skyrelyje panagrinėsime, W. Friedmanas išdėstė savo veikale „Sutapimų indeksas“ („The Index of Coincidence“). Panagrinėsime, kaip jį galima taikyti Vigenere šifrui. Pats W. Friedmanas jį naudojo ir kitokių šifrų analizei.

Tarkime, teksto ir šifro abėcėlę $\mathcal{A}$ sudaro n simbolių (lietuvių kalbos abėcėlėje $n = 32$):

$$\mathcal{A} = \{a_1, a_2, \ldots, a_n\}.$$

[8] Vigenere šifrų analizei galite pasinaudoti Java įskiepiais šios knygos tinklalapyje.

Jeigu iš šios abėcėlės su grąžinimu rinktume dvi raides, tai tikimybė, kad abi raidės būtų a_1 lygi $\frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$, kad a_2 – tokia pati ir t. t. Taigi dydį

$$\kappa_0 = \frac{1}{n^2} + \frac{1}{n^2} + \ldots + \frac{1}{n^2} = \frac{1}{n}$$

galime suvokti kaip tikimybę gauti dvi vienodas raides, kai jas atsitiktinai su grąžinimu renkame iš abėcėlės. Vadinsime šį dydį atsitiktinio teksto sutapimų indeksu.

Pažymėkime $p_1, p_2, \ldots, p_n$ abėcėlės raidžių pasitaikymo tikimybes kokia nors kalba parašytame tekste.

Jeigu dabar dvi raides atsitiktinai rinktume iš kokio nors tikro teksto, tai tikimybė gauti vienodas būtų

$$\kappa_t = p_1^2 + p_2^2 + \ldots + p_n^2.$$

Šį skaičių vadinsime teksto sutapimų indeksu. Pavyzdžiui, lietuvių kalbos teksto sutapimų indeksas yra $\kappa_t \approx 0,069$.

Jeigu sugretintume surašydami į dvi eilutes du skirtingus, bet to paties ilgio N ir ta pačia kalba parašytus tekstus – kiek stulpelių iš vienodų raidžių gautume? Stulpelių iš vienodų raidžių būtų

$$\approx N \cdot \kappa_t.$$

O jeigu sugretintume dviejų tokių tekstų šifrus, gautus panaudojus vieną keitinį? Vienodų stulpelių skaičius būtų maždaug tas pats. O jeigu abu tekstai būtų užšifruoti tuo pačiu Vigenere šifru? Irgi tas pats! O jeigu skirtingais Vigenere šifrais? Nieko konkretaus negalime pasakyti, tačiau galima tikėtis, kad vienodų raidžių stulpelių būtų

$$\approx N \cdot \kappa_0.$$

Šiais samprotavimais ir remiasi Friedmano kappa testas.

Įsivaizduokime, kad kriptoanalitikui įteiktas Vigenere šifras, kuris buvo sudarytas naudojant, tarkime, raktą ABC, kurio kriptoanalitikas, aišku, nežino. Tarkime, tas šifras yra toks: $C = c_1c_2c_3c_4c_5c_6\ldots$ Galime įsivaizduoti, kad jis gautas naudojant raktą $abcabcabc\ldots$ Jeigu kriptoanalitikas nubrauktų pirmąjį šifro C simbolį, gautų šifrą, kuris sudarytas naudojant raktą $bcabcabc\ldots$ Jeigu jis suskaičiuotų, kiek yra vienodų raidžių stulpelių tokioje lentelėje

$$\begin{matrix} c_1 & c_2 & c_3\ldots \\ c_2 & c_3 & c_4\ldots \end{matrix},$$

matyt, gautų, kad jų yra maždaug κ_0 nuo viso stulpelių skaičiaus. Tą patį rezultatą gautų ir iš lentelės

$$\begin{matrix} c_1 & c_2 & c_3\ldots \\ c_3 & c_4 & c_5\ldots \end{matrix}.$$

O štai skaičiuodamas vienodų raidžių stulpelių dalį lentelėje

$$\begin{matrix} c_1 & c_2 & c_3 \dots \\ c_4 & c_5 & c_6 \dots \end{matrix},$$

jis turėtų gauti reikšmę, artimą skaičiui κ_t. Tada jis galėtų padaryti išvadą, kad rakto ilgis yra tikriausiai lygus 3.

Išbandykime kappa testą. Prisiminkime, kaip sėkmingai pavyko įspėti rakto ilgį, naudojantis Kassiskio testu. Dabar tam pačiam šifrui analizuoti pasinaudokime Friedmano metodu: sugretinkime pradinį šifrą ir šio šifro dalį, gautą atmetus pirmuosius d simbolių ($d = 1, 2, \dots$) ir apskaičiuokime, kokią dalį sudaro vienodų raidžių stulpeliai. Suapvalinę iki tūkstantųjų, gausime tokius skaičius:

Poslinkiai $d =$	1	2	3	4	5	6	7
Sutapimai	0,032	0,029	0,04	0,033	0,055	0,026	0,022

Taigi ir kappa testas rodo, kad rakto ilgis turėtų būti lygus 5.

Naudojantis sutapimų indeksais, netgi galima išvesti apytikslę Vigenere šifro rakto ilgio formulę.

Tarkime, pavyko sugauti N ilgio žinoma kalba parašyto teksto šifrą

$$C = c_1 c_2 \dots c_N.$$

Žinome sutapimų indeksus κ_0, κ_t. Tačiau šifro raidžių dažnių lentelė skiriasi nuo nešifruoto teksto. Kaip surasti šifro sutapimų indeksą κ_c, t. y. tikimybę, kad, atsitiktinai iš šifro rinkdami dvi raides, gausime vienodas? Šį dydį suskaičiuosime dviem būdais: naudodamiesi gautuoju šifru ir tikimybiniais samprotavimais.

Tegu simboliai $a_1, \dots, a_n$ pasikartoja šifre atitinkamai $m_1, \dots, m_n$ kartų. Įsivaizduokime, kad atsitiktinai pasirenkame du gauto šifro simbolius. Tikimybę, kad gausime du vienodus, galime skaičiuoti taip:

$$\kappa_c = \frac{1}{C_N^2} \sum_{i=1}^{n} C_{m_i}^2.$$

Dabar padarykime prielaidą, kad šifravimui panaudotas d ilgio raktas. Suskaidysime šifrą į d fragmentų:

$$\begin{matrix} c_1 & c_{1+d} & c_{1+2d} & \dots \\ c_2 & c_{2+d} & c_{2+2d} & \dots \\ \vdots & \vdots & \vdots & \vdots \\ c_d & c_{2d} & c_{3d} & \dots \end{matrix}.$$

Tarsime, kad N yra pakankamai didelis skaičius, tad kiekviename fragmente yra $\approx N/d$ simbolių. Kiekvienas fragmentas – Cezario šifras, tad galime manyti, kad tikimybė, jog du simboliai, parinkti iš to paties segmento, sutampa, lygi atviro teksto sutapimų indeksui κ_t. Dabar įsivaizduokime, kad atsitiktinai pasirenkame x, y – du gauto šifro simbolius. Skaičiuosime tikimybę, kad jie sutampa. Iš pradžių pažymėkime įvykius:

$$\begin{aligned} A &= \{\text{pasirinkti simboliai } x, y \text{ yra iš to paties segmento}\}, \\ A^c &= \{\text{pasirinkti simboliai } x, y \text{ yra iš skirtingų segmentų}\}. \end{aligned}$$

Pasinaudokime pilnos tikimybės formule

$$P(x = y) = P(x = y|A)P(A) + P(x = y|A^c)P(A^c).$$

Šioje lygybėje imsime:

$$P(x = y|A) = \kappa_t, \quad P(x = y|A^c) = \kappa_0$$

ir suskaičiuosime besąlygines tikimybes:

$$\begin{aligned} P(A) &= \frac{1}{C_N^2} \cdot \frac{N}{2} \cdot \Big(\frac{N}{d} - 1\Big), \\ P(A^c) &= \frac{1}{C_N^2} \cdot \frac{N(N - N/d)}{2}. \end{aligned}$$

Kadangi $P(x = y) = \kappa_c$, tai gausime formulę

$$\kappa_c = \frac{1}{C_N^2} \cdot \frac{N}{2} \cdot \Big(\frac{N}{d} - 1\Big)\kappa_t + \frac{1}{C_N^2} \cdot \frac{N(N - N/d)}{2}\kappa_0.$$

Iš šios lygybės jau galime surasti apytikslę formulę rakto ilgiui:

$$d \approx \frac{N(\kappa_t - \kappa_0)}{\kappa_c(N - 1) + \kappa_t - N\kappa_0}.$$

14.6. Nauji laikai – nauji iššūkiai

Su „juodųjų kambarių“ – Europos absoliutinių monarchijų kriptografų tarnybų laikais baigėsi ir „popieriaus ir pieštuko“ šifrų epocha. Mechanizmai keitė rankų darbą, o kartais ir galvos. Kokie tie pirmieji mechaniniai šifravimo prietaisai? Kokie nauji iššūkiai kriptografijai?

Žinome, kokį kelią nuėjo skaičiavimo technikos kūrėjai: nuo skaitytuvo (abako) iki kompiuterio. Kriptografijos istorijoje irgi galime nubrėžti tokį kelią: nuo pirmojo šifravimui skirto įrenginio (skytalės) iki to paties kompiuterio. Garbingą vietą skaičiavimo technikos istorijoje užsitarnavo mechaninių prietaisų – aritmometrų – kūrėjai: Pascalis, Leibnizas, Babbage'as ir kiti

išradėjai. Kriptografijoje irgi būta mechaninių prietaisų, nors ir gerokai menkiau žinomų. Po spartiečių jų skytalės tikriausiai niekas nenaudojo, Alberti skrituliai taip ir liko išradimu popieriuje...

Visai nebloga idėja, kaip mechanizuoti šifravimą ir dešifravimą XVIII amžiaus pabaigoje kilo Thomui Jeffersonui. Jis buvo trečiasis JAV prezidentas, o pirmojo prezidento – George'o Washingtono metais – valstybės sekretorius, taigi rūpinosi užsienio politika. Galbūt todėl jis ir susimąstė apie šifrus.

Jeffersono prietaisą sudaro trisdešimt šeši siauri vienodo skersmens ritiniai, sumauti ant ašies, apie kurią kiekvienas jų gali sukiotis. Ant kiekvieno ritinio šoninio paviršiaus surašytos abėcėlės raidės – vis kita tvarka. Taigi galime sakyti, kad ant kiekvieno ritinio surašyta vis kita abėcėlė. Ritinius galima numauti nuo ašies ir perstatyti kitaip. Kai visi ritiniai sumauti ant ašies, susidaro 36 raidžių eilutės. Šifruojama taip: pažymėkime vieną eilutę (eilutei fiksuoti galima pritaisyti liniuotę) ir, sukiodami ritinius, surinkime šioje eilutėje 36 (ar mažiau) raidžių tekstą. Visose kitose eilutėse irgi susidarys tekstai – tai šifrai. Galime pasiųsti bet kurį iš jų. Gavėjas, norėdamas iššifruoti šifrą, privalo turėti lygiai tokius pat ritinius, suvertus ant ašies tokia pat tvarka. Surinkęs fiksuotoje eilutėje šifro raides jis peržiūrės kitas eilutes. Vienoje iš jų jis perskaitys mūsų pasiųstą žinią.

Šis šifravimo prietaisas vadinamas Jeffersono ritiniu. Svarbu, kad siuntėjas ir gavėjas naudotųsi juo, sumovę ant ašies siauruosius ritinius su abėcėlėmis ta pačia tvarka. Taigi šios šifravimo sistemos raktas yra ritinių išdėstymo tvarka. Iš viso yra 36! skirtingų išdėstymų. Taigi raktų yra labai daug.

Žinoma, ritinių skaičius gali būti kitas. Galimos ir kitokios šifro sudarymo taisyklės. Pavyzdžiui, pusę šifro galima perskaityti iš vienos, kitą pusę – iš kitos eilutės.

Šis įrenginys buvo ne kartą išrastas pakartotinai. 1891 metais panašų prietaisą sugalvojo įžymus prancūzų kriptografas E. Bazeries, XX amžiaus pradžioje tokį prietaisą naudojo amerikiečiai.

Tiems laikams tai buvo labai saugus šifras. Tačiau juo nesinaudota. Galbūt ir todėl, kad, rūpindamasis sudėtingais to meto politikos reikalais, Jeffersonas primiršo savo išradimą. Panašiais šifravimo įrenginiais naudojosi karinis JAV laivynas ir vyriausybė antrajame XX amžiaus dešimtmetyje. Tačiau ne todėl, kad buvo prisimintas Jeffersono ritinys. Jis tiesiog buvo iš naujo išrastas!

Tačiau tikrasis kriptografijos prietaisų poreikis atsirado tada, kai Samuelis Morse 1844 metais pasiuntė pasauliui pranešimą apie naujos ryšių epochos pradžią. Žinoma, nei jis, nei kiti nemanė, kad prasidėjo kažkas nepaprasta. Samuelis Morse tiesiog išrado telegrafą.

Telegrafas pakeitė visas žmonių tarpusavio bendravimo „per nuotolį“ aplinkybes. Juk svetimo laiško skaitymas greičiau išimtis negu taisyklė, o žinią perduoti telegrafu be pašalinių žmonių pagalbos beveik neįmanoma. Taigi informacijos slaptumo klausimas tapo svarbus ne vien diplomatams ir kariškiams.

Pasikeitusi padėtis kėlė naujus uždavinius kriptografijai.

Vienas iš kriptografijos naujos raidos ženklų – vėl susidomėta šifrais. Tiesa, šifrais visada domėtasi. Tačiau jais domėtasi daugiau teoriškai, o praktikoje vyravo kodai. Sudaryti gerą kodą – specialisto darbas, o išmokyti juo naudotis galima bet kokį eilinį diplomatų atstovybės sekretorių. Tereikia įteikti jam kodų knygą ir paaiškinti, kaip ją vartyti. Kas kita šifrai. Šifruojant ir dešifruojant reikia atidžiai atlikti algoritmo žingsnius, o vienintelė perdavimo arba šifravimo klaida kartais gali visą šifrą paversti neiššifruojamu.

Tačiau telegrafo laikais šifrų pranašumai su kaupu kompensuoja visus nepatogumus. Svarbiausias dalykas, kad šifro raktą pakeisti yra labai lengva, o pakeisti kodą – anaiptol. Juk tektų perrašyti ištisą knygą, o įvykiai juk nelaukia.

Kokių gi šifrų reikia telegrafo epochai, kai didelė dalis svarbių pranešimų keliauja ne vokuose, ne ant brangaus popieriaus su vandens ženklais, bet, pavirtę taškais ir brūkšneliais, įveikia didžiulius atstumus ir pasiekia adresatą mechaninio prietaiso atspausdinti ant siauros popieriaus juostelės?

Pirmasis tai labai aiškiai suvokė ir suformulavo Augustas Kerckhoffas. Jis gimė 1835 metais Olandijoje, buvo senos ir žymios giminės atžala, todėl, matyt, ir visas jo vardas yra itin ilgas: Jean-Guillaume-Hubert-Victor-Francois-Alexandre-Auguste Kerckhoffs von Nieuwenhof. Jis buvo labai išsilavinęs žmogus: Liežo universitete įgijo net du mokslo laipsnius – iš kalbų ir tiksliųjų mokslų, dėstė įvairius dalykus, mokė kalbų ir rašė knygas – daugiausia filologijos.

Kriptografijai svarbus Kerckhoffo veikalas „La cryptographie militaire“ [9] buvo išspausdintas 1883 metais karo mokslams skirtame žurnale, o vėliau išleistas atskira knygele.

Kerckhoffas pabrėžia, kad veikiančioje armijoje atsirado būtinybė palaikyti nuolatinį šifruotą ryšį. Iš to išplaukia nauji reikalavimai naudojamoms šifravimo sistemoms. Kerckhoffas suformuluoja šešias sąlygas.

Pirma, jeigu šifravimo sistema gali būti įveikta, tai tik matematiškai (*le système doit être matériellement, sinon mathématiquement, indéchiffrable*).

Taigi šifruota informacija negali būti atskleista taip kaip iš dėlionės dalelių sudedamas paveikslas, t. y. sistemą galima įveikti tik atskleidus jos matematinius pagrindus.

Antra, sistema turi būti tokia, kad net ją turėdamas priešininkas negalėtų jos įveikti (*il faut qu'il n'exige pas le secret, et qu'il puisse sans inconvénient tomber entre les mains de l'ennemi*).

Trečia, sistemos raktas turi būti įsimenamas ir perduodamas jo neužrašius, jis turi būti keičiamas (*la clef doit pouvoir en être communiquée et retenue sans le secours de notes écrites, et être changée et modifiée au gré des corre-*

[9] Auguste Kerckhoffs. La cryptographie militaire. Journal des sciences militaires, vol. IX, Jan. 1883, p. 5—83, Feb. 1883, p. 161—191.

spondants).

Ketvirta, sistema turi būti pritaikyta telegrafo ryšiui (*il faut qu'il soit applicable à la correspondance télégraphique*).

Penkta, šifravimo sistema turi būti nešiojama ir naudojimuisi ja nereiktų daugelio žmonių (*il faut qu'il soit portatif, et que son maniement ou son fonctionnement n'exige pas le concours de plusieurs personnes*).

Šešta, sistema turi būti paprasta naudotis: neturi būti reikalinga nei proto įtampa, nei ilga taisyklių seka (*le systéme doit être d'un usage facile ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer*).

Jeigu ir reiktų ką pakeisti pritaikant šį „kriptografijos kodeksą" mūsų laikams – tai vietoj telegrafo minėti elektroninį ryšį.

Tačiau kriptografija Kerckhoffo laikais vis dar buvo kūdikystės amžiuje. Pats A. Kerckhoffas savo straipsnyje rašo, kad jį stebina mokyti žmonės, siūlantys šifravimo sistemas, kurias įveikti galima per pusvalandį. Kad kriptografijos reikšmė būtų suvokta, reikėjo sunkių išbandymų. Būtinybę kuo greičiau subręsti atskleidė Pirmasis pasaulinis karas.

14.7. Rotoriai ir Enigma

> *Enigma – šitaip vokiečiai vadino šifravimo įrenginius, kuriuos jie naudojo Antrojo pasaulinio karo mūšiuose. Enigma – tai mūšis, kurį laimėjo britų matematikai, dirbdami tyliuose Bletchley Park kambariuose...*

Telegrafu ir radijo ryšiu Pirmajame pasauliniame kare pradėta naudotis neturint beveik jokios patirties kriptografijos ir kriptoanalizės srityse. Tik prancūzai buvo kiek geriau pasiruošę. Šlovinga britų naujųjų laikų kriptografijos istorija prasidėjo tiesiog prie vieno stalo su vokiečių šifrų šūsnimi, už kurio susėdo keli karo laivininkystės koledžo dėstytojai, tikėdamiesi ką nors iš tos šūsnies išrausti. Kriptoanalitikų darbas niekada nebuvo toks svarbus kaip šio karo metu. Užtenka paminėti vien Zimmermano telegramos atvejį: britams iššifravus vokiečių karo ministro telegramą pasiuntiniui Meksikoje, vis dar delsęs JAV prezidentas W. Wilsonas apsisprendė dėl JAV dalyvavimo kare.

Tačiau karas baigėsi ir dėmesys kriptografijai (o taip pat ir pinigai) sumenko. Talentingas amerikiečių inžinierius Gilbertas Vernamas truputį pavėlavo. 1918 metais Vašingtone jis pateikė paraišką naujo šifravimo-dešifravimo prietaiso patentui gauti. Jo idėja apie telegrafu perduodamos informacijos šifravimą buvo puiki. Tuomet perdavimui buvo naudojamas Baudot'o kodas. Naudojantis šiuo kodu, kiekvienai raidei perduoti telegrafu reikia penkių trumpų laiko intervalų. Per kiekvieną intervalą įrenginys arba pasiunčia elektros impulsą, arba ne. Impulso perdavimą galime pažymėti vienetu, o jo nebuvimą nuliu. Tada kiekviena raidė bus koduojama vienetų ir nulių gretiniu,

kuris sudarytas iš penkių simbolių. Gavėjo įrenginys pagal šiuos perduodamus nulių-vienetų penketus turi atkurti perduodamas abėcėlės raides.

Tarkime, telegrafu reikia perduoti vieną raidę, t. y. tam tikrą nulių-vienetų penketą, pavyzdžiui, 01001. Vernamui kilo mintis: sudarykime elektrinę grandinę, kuri sudėtų jai perduodamas vienodo ilgio nulių-vienetų eilutes panariui, naudodamasi tokia sudėties lentele:

$$0 \oplus 0 = 0; \quad 0 \oplus 1 = 1; \quad 1 \oplus 0 = 1; \quad 1 \oplus 1 = 0.$$

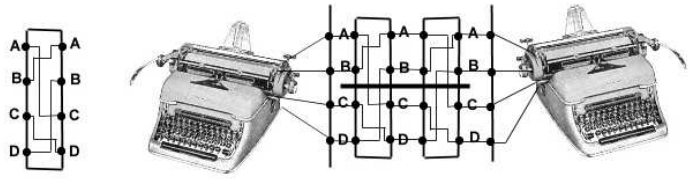
Jeigu tokiai grandinei perduosime eilutes $m = 01001$ ir $k = 11011$, tai grandinė sukurs eilutę $c = 10010$. Jeigu m yra pranešimas, kurį reikia perduoti, tai c yra šifras, gautas panaudojus raktą k. Kaip gavėjas gali iššifruoti gautą šifrą c? Jeigu jo įrenginyje bus tokia pati sudėties grandinė, tai, įvedęs į ją eilutes c ir k, gaus m. Taigi šifruojama ir dešifruojama visiškai vienodai ir šiuos veiksmus gali atlikti pats įrenginys. Tiesa, jeigu reikia perduoti ilgą pranešimą, tai ir rakto reikia ilgo. Taigi G. Vernamas pasiūlė šifravimui naudoti operaciją, kurią matematikai vadina sudėtimi moduliu 2, o informatikai – XOR operacija. Šią operaciją naudojome nagrinėdami klaidas taisančius kodus. Šiuolaikinėse kriptosistemose ji sutinkama kone kiekvianame žingsnyje.

Vernamo įrenginys galėjo padaryti perversmą ryšių technikoje. Tačiau taip neįvyko. Viena vertus, vyriausybės galvojo, kaip baigti kariauti, o ne iš naujo pradėti. Kita vertus, komercinės įstaigos buvo įpratusios naudotis kodų knygomis ir nepirko naujojo įrenginio. Svarbiausia šios puikios idėjos komercinės nesėkmės priežastis ta, kad ji atsirado anksčiau, nei subrendo nauji saugaus ryšio reikalavimai.

Kriptografinių prietaisų raida pasuko ne Vernamo nurodyta kryptimi. Pasirodė kažkas panašaus į Jeffersono ritinius, suvertus ant vienos ašies.

Įsivaizduokime diską, padarytą iš kietos elektrai nelaidžios medžiagos, kurio abiejose pusėse ratu pagal kraštą išdėstyti kontaktai. Kontaktų abiejose pusėse yra tiek, kiek raidžių abėcėlėje (lotyniškoje abėcėlėje 26). Taigi abiejose pusėse kiekviena abėcėlės raidė turi po kontaktą. Šie kontaktai tarpusavyje poromis sujungti laidais. Pavyzdžiui, raidės A kontaktas gali būti sujungtas su raidės C kontaktu, raidės B – su L ir t. t. Šis ritinys ir yra pagrindinė naujųjų kriptografijos prietaisų detalė. Jis vadinamas rotoriumi, jau pats vardas rodo, kad šifravimo sistemoje didelė reikšmė teikiama šio ritinio posūkiams apie pervertą ašį.

Panagrinėkime rotorių panaudojimo šifravimui idėją, kurią kone vienu metu patentavo amerikietis Edwardas Hughas Hebernas, olandas Alexanderis Kochas, švedas Arvidas Gerhardas Dammas ir vokietis Arthuras Scherbiusas.



Šifravimo įrenginio, naudojančio rotorius, veikimo principas. Tikrieji įrenginiai, žinoma, buvo sudėtingesni. Pavyzdžiui, impulsas, praėjęs elektros grandine nuo pirmojo iki paskutiniojo rotoriaus kontaktų, atsispindėjęs grįžtavo jau kita grandine į pirmąjį rotorių.

Įsivaizduokime, kad prie abiejų rotoriaus pusių kontaktų prijungtos dvi elektromechaninės spausdinimo mašinėlės. Jeigu paspausime kairiosios mašinėlės A raidės klavišą, tai į kairiosios rotoriaus pusės A raidės kontaktą bus perduotas srovės impulsas, kuris laidu pateks į dešiniosios rotoriaus pusės raidės C kontaktą, ir dešinioji spausdinimo mašinėlė atspausdins raidę C (jeigu, žinoma, raidės A kontaktas sujungtas su raidės C kontaktu). „Sukonstravome“ paprastą vienu abėcėlių keitiniu paremtą šifravimo sistemą, kuri nėra nei nauja, nei saugi. Jeigu be perstojo spaudysime kairiosios spausdinimo mašinėlės raidės A klavišą, dešinioji mašinėlė spausdins CCCCCC..... Dabar patobulinkime įrenginį.

Mūsų rotorių patalpinkime tarp dviejų skritulio formos nejudamų plokštelių su tokia pat tvarka kaip ant rotoriaus išdėstytais kontaktais: kairiosios plokštelės kontaktai liečiasi su rotoriaus kairiosios pusės kontaktais, dešiniosios plokštelės – su dešiniosios pusės. Spausdinimo mašinėles prijunkime prie atitinkamų plokštelių kontaktų. Šifravimo sistema nepasikeitė, tačiau šitaip mes įgijome galimybę sukioti mūsų rotorių nepriklausomai nuo prijungtų spausdinimo mašinėlių. Padarykime taip, kad, kiekvieną kartą paspaudus kairiosios spausdinimo mašinėlės klavišą, rotorius pasisuktų per 1/26 apskritimo, tarkime, laikrodžio rodyklės kryptimi. Paspaudus raidės A klavišą, antroji spausdinimo mašinėlė atspausdins raidę C, rotorius pasisuks, tačiau nejudamų plokštelių kontaktai vėl liesis su rotoriaus kontaktais, tačiau jau kitais, todėl antrą kartą paspaudus raidės A klavišą, antroji spausdinimo mašinėlė atspausdins nebe C, bet kokią nors kitą raidę! Jeigu vėl be perstojo spaudysime kairiosios spausdinimo mašinėlės raidės A klavišą, tai dešinioji mašinėlė spausdins raidžių seką, kuri ims kartotis tik po 26 raidės. Šis įrenginys realizuoja šifravimo sistemą, panašią į tą, kurią pasiūlė Trithemius. Panagrinėkime matematinę tokios sistemos su vienu rotoriumi struktūrą.

Tarkime, nejudamų plokštelių kontaktams abėcėlės raidės priskirtos eilės tvarka, t. y. jeigu spaudžiame raidės A klavišą, tai impulsas patenka į kairiosios plokštelės pirmąjį kontaktą, jeigu B – į antrąjį ir t. t. Analogiškai jeigu impulsas patenka į dešiniosios plokštelės pirmąjį kontaktą, tai šifrą spausdinanti mašinėlė išspausdina A, jeigu į antrąjį – B ir t. t. Sunumeruokime plokštelių kontaktus eilės tvarka, raidės A kontaktui priskirkime 1, raidės B

– 2 ir t. t. Tegu abėcėlėje yra n raidžių. Tais pačiais skaičiais $1, 2, \ldots, n$ sunumeruokime ir kairiuosius bei dešiniuosius rotoriaus kontaktus, abiejose pusėse numeravimas prasideda nuo tos pačios padėties. Tačiau rotoriaus kontaktai yra poromis sujungti. Sujungimus nurodysime keitiniu

$$\lambda = \begin{pmatrix} 1 & 2 & \ldots & n \\ i_1 & i_2 & \ldots & i_n \end{pmatrix}.$$

Šiuo keitiniu nurodoma, kad pirmasis kairiosios pusės kontaktas sujungtas su i_1 dešiniosios pusės kontaktu ir t. t.

Tarkime, plokštelių ir rotoriaus pradinė padėtis yra tokia, kad kairiosios plokštelės pirmasis kontaktas yra ties pirmuoju kairiosios rotoriaus pusės kontaktu, rotoriaus dešiniosios pusės pirmasis kontaktas yra ties pirmuoju dešiniosios plokštelės kontaktu. Taigi pirmosios plokštelės ir rotoriaus kontaktų bei rotoriaus ir antrosios plokštelės kontaktų atitiktį galime nusakyti tuo pačiu trivialiu keitiniu

$$\epsilon = \begin{pmatrix} 1 & 2 & \ldots & n \\ 1 & 2 & \ldots & n \end{pmatrix}.$$

Kairiosios ir dešiniosios plokštelių kontaktų atitiktį (šifro keitinį) galime užrašyti taip:

$$\sigma_0 = \lambda = \epsilon\lambda\epsilon.$$

Pasukime rotorių per vieną poziciją. Dabar kairiosios plokštelės ir kairiųjų rotoriaus kontaktų bei dešiniųjų rotoriaus ir dešiniosios plokštelės kontaktų atitiktį užrašysime jau kitais keitiniais:

$$\rho_1 = \rho = \begin{pmatrix} 1 & 2 & \ldots & n-1 & n \\ 2 & 3 & \ldots & n & 1 \end{pmatrix}, \quad \rho^{-1} = \begin{pmatrix} 2 & 3 & \ldots & n & 1 \\ 1 & 2 & \ldots & n-1 & n \end{pmatrix}.$$

Savo ruožtu šifravimo keitinys bus

$$\sigma_1 = \rho^{-1}\lambda\rho.$$

Pasukę rotorių per $2, 3, \ldots, m$ pozicijų, gausime plokštelių ir rotoriaus kontaktų atitikties keitinius $\rho^2, \rho^3, \ldots, \rho^m$ ir $\rho^{-2}, \rho^{-3}, \ldots, \rho^{-m}$. Vadinasi pasukę rotorių per m pozicijų, gauname tokį šifro keitinį:

$$\sigma_m = \rho^{-m}\lambda\rho^m, \quad m = 0, 1, 2, \ldots, \rho^n = \epsilon.$$

Taigi visų žingsnių šifravimo keitinius apibrėžia keitinys λ ir pradinė plokštelių ir rotoriaus kontaktų atitiktis.

Pavyzdžiui, jeigu keturių raidžių abėcėlės atveju plokštelių ir rotoriaus kontaktų pradinę padėtį nusako vienetinis keitinys ϵ, tai su rotoriaus kontaktų porų keitiniu λ gausime tokius keturių žingsnių šifravimo keitinius:

$$\lambda = \sigma_0 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 2 & 1 & 3 \end{pmatrix}, \ \sigma_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 2 & 3 \end{pmatrix},$$

$$\sigma_2 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 1 & 2 & 4 \end{pmatrix}, \ \sigma_3 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 3 & 2 \end{pmatrix}.$$

Patobulinkime įrenginį dar kartą. Imkime kitą rotorių (kairiosios ir dešiniosios pusių kontaktai sujungti poromis, tačiau kokia nors kita tvarka) ir užmaukime ant ašies. Tegu abu rotoriai liečiasi savo kontaktais: pirmojo rotoriaus dešiniosios pusės kontaktai liečia antrojo rotoriaus kairiosios pusės kontaktus, taigi pirmojo rotoriaus kairiosios pusės kontaktai liečia kairiosios nejudančios plokštelės kontaktus, o antrojo rotoriaus dešiniosios pusės kontaktai – dešiniosios plokštelės kontaktus. Spausdinimo mašinėlės lieka sujungtos su plokštelių kontaktais. Jeigu dabar nuspausime kairiosios spausdinimo mašinėlės raidės A klavišą, dešinioji atspausdins tikriausiai nebe C, bet kokią nors kitą raidę, pavyzdžiui, V. Padarykime taip, kad kiekvieną kartą, kai nuspaudžiamas kairiosios mašinėlės klavišas, antrasis rotorius pasisuka per 1/26 apskritimo laikrodžio rodyklės kryptimi, o kai šis rotorius po 26 posūkių grįžta į pradinę padėtį, per 1/26 apskritimo laikrodžio rodyklės kryptimi pasisuka pirmasis rotorius. Tada vėl sukiojasi antrasis rotorius, o pirmasis vėl pajuda tik po 26 antrojo rotoriaus posūkių. Panašiai juda elektros, vandens ar dujų apskaitos skaitiklių ritinėliai su skaitmenimis.

Jeigu nuolat spaudysime kairiosios mašinėlės raidės A klavišą, po kelių simbolių dešiniosios mašinėlės spausdinama raidžių seka ims kartotis? Po $26 \times 26 = 676$ simbolių. O jeigu panaudosime ne du, bet tris rotorius? Tada po $676 \times 26 = 17576$ simbolių. Net ir pasitelkus matematinius metodus bei kompiuterius, analizuoti tokį šifrą nėra lengva. Taigi rotorių sistema itin paprastai realizuoja šifrus su daug abėcėlių. Šifras priklauso nuo pradinės rotorių tarpusavio padėties, kurią galima keisti. Šitaip galima pasiekti, kad šifras priklausytų nuo rakto.

Matematiškai šifravimą su rotoriais galime aprašyti taip. Tarkime, turime tris rotorius ir kiekvieno rotoriaus kontaktai sunumeruoti tokia tvarka kaip nagrinėtu atveju, sujungtas kontaktų poras nusako keitiniai $\lambda_1, \lambda_2, \lambda_3$. Be to, tarkime, kad pradinėje padėtyje visų besiliečiančių kontaktų numeriai tie patys, t. y. kairiosios plokštelės pirmąjį kontaktą liečia pirmojo rotoriaus kairiosios pusės pirmasis kontaktas, jo dešiniosios pusės pirmąjį kontaktą liečia antrojo rotoriaus kairiosios pusės pirmasis kontaktas ir t.t. Koks šifravimo keitinys bus m-ajame žingsnyje, jei $m < n^3$? Išreikškime m n-ainėje

skaičiavimo sistemoje:

$$m = m_1 + m_2 n + m_3 n^2, \quad 0 \le m_i < n.$$

Tada kiek pagalvojus galima įsitikinti, kad šifravimo abėcėlės keitinys bus toks:

$$\sigma_m = \rho^{-m_3}\lambda_3\rho^{m_3}\rho^{-m_2}\lambda_2\rho^{m_2}\rho^{-m_1}\lambda_1\rho^{m_1}.$$

Rotorių principas panaudotas įžymiuosiuose vokiečių šifravimo įrenginiuose „Enigma“ . Antrojo pasaulinio karo metu vokiečiai naudojo kelis įrenginių variantus su trimis rotoriais, kuriuos buvo galima parinkti iš penkių rotorių rinkinio. Įrenginiai buvo puikūs, tačiau jie neatlaikė lenkų ir britų kriptoanalizės atakų. Štai tik dvi pavardės iš didingos Antrojo pasaulinio karo kriptoanalizės istorijos: Marijanas Rejewskis ir Alanas Turingas. Kodėl „Enigma“ pasidavė? Ne todėl, kad algoritmas buvo nesaugus, bet todėl, kad kiekvienos apsaugos sistemos saugumo lygis yra toks pat kaip pačios silpniausios jos grandies. Silpnoji „Enigmos“ praktinio naudojimo grandis – būtinybė susitarti dėl raktų (pradinių rotorių padėčių) ir vokiečių įprotis tai daryti.

15 Informacijos teorija ir kriptosistemos

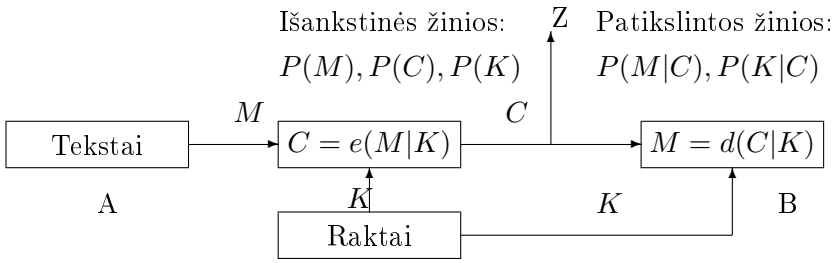
Ištisus šimtmečius kriptosistemų saugumas vertintas pasikliaujant tik subjektyvia šios srities autoritetų nuomone. Žinoma, patyrusių kriptoanalitikų nuomonė ir dabar yra svarbus argumentas. Kriptografijoje ir šiandien ne viską galima objektyviai pasverti ir įvertinti. Tačiau daug ką galima. Kokie gi tie moksliniai kriptosistemų saugumo vertinimo kriterijai?

15.1. Shannono modelis

C. Shannonas – informacijos teorijos pradininkas – yra ir teorinio kriptosistemų vertinimo pagrindų kūrėjas. Panagrinėkime jo sukurtą modelį.

Pirmasis teorinį kriptosistemų vertinimo pagrindą pabandė pateikti C. Shannonas. Jis pasinaudojo savo paties sukurtos informacijos teorijos sąvokomis. Informacijos teorijoje nagrinėjamos patikimo ryšio nepatikimais kanalais galimybės. Kada iš iškraipytos perdavimo kanale informacijos galima atkurti siųstąją? Šifravimą irgi galima interpretuoti kaip pradinės informacijos iškraipymą. Kada iš šifro (iškraipytos informacijos) galima atkurti pradinę, o taip pat – šifravimui naudotą raktą? Kitaip tariant – kaip galima kiekybiškai vertinti kriptosistemos atsparumą pavienių šifrų atakų atžvilgiu?

Panagrinėkime kriptosistema apsaugoto A ir B ryšio modelį, kurį naudojo C. Shannonas. Apsiribosime simetrinėmis kriptosistemomis $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$; tarsime, kad šifravimui ir dešifravimui naudojamas raktas yra tas pats: $K_e = K_d = K$.



C. Shannono šifruoto ryšio modelis. Kriptosistema tuo silpnesnė, kuo labiau gautas šifras patikslina kriptoanalitiko išankstines žinias apie siunčiamą tekstą ir panaudotą raktą.

Tarsime, kad kriptoanalitikui Z visos kriptosistemos komponentės yra gerai žinomos: jis žino, kokiai aibei priklauso šifruojami pranešimai, iš kokios aibės pasirenkami raktai ir kaip veikia šifravimo ir dešifravimo algoritmai. Jis taip pat turi neribotas galimybes stebėti ryšio kanalą bei atlikti skaičiavimus. Įsivaizduokime, kad jis laukia pirmojo siuntinio, kurį jis pasiruošęs įsirašyti į savo laikmenas. Pranešimas, kuris bus užšifruotas, jo požiūriu, yra atsitiktinio dydžio reikšmė. Žymėsime jį taip pat kaip galimų pranešimų aibę, t. y. $\mathcal{M}$. Kriptoanalitikas taip pat turi tam tikrą išankstinę informaciją, kokie pranešimai yra mažiau, kokie daugiau tikėtini, t. y. jis žino tikimybes

$$p(M) = P(\mathcal{M} = M).$$

Analogiškai naudojamą raktą irgi galime interpretuoti kaip atsitiktinį dydį $\mathcal{K}$, įgyjantį reikšmes su tikimybėmis $p(K)$. Natūralu daryti prielaidą, kad dydžiai $\mathcal{M}$ ir $\mathcal{K}$ yra nepriklausomi atsitiktiniai dydžiai. Šifruotą tekstą irgi galime suprasti kaip atsitiktinį dydį $\mathcal{C}$, su tikimybėmis $p(C)$ įgyjantį reikšmes iš galimų šifrų aibės; jis vienareikšmiškai apibrėžiamas su $\mathcal{M}, \mathcal{K}$ sąryšiu

$$e(\mathcal{M}|\mathcal{K}) = \mathcal{C}.$$

Taigi tikimybės $p(M), p(K), p(C)$ yra ta išankstinė informacija, kurią turi kriptoanalitikas Z prieš sugaudamas pirmąjį siunčiamą šifrą. Tarkime, kad to šifro C jis galų gale sulaukė. Tuomet jis gali patikslinti savo turimas žinias apie siunčiamą pranešimą, apskaičiuodamas sąlygines tikimybes

$$P(\mathcal{M} = M|\mathcal{C} = C), \quad M \in \mathcal{M}.$$

Gali būti, kad reikšmės liko tos pačios, tada Z iš gauto šifro neturės jokios naudos!

94 apibrėžimas. *Kriptosistemą $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ vadinsime besąlygiškai saugia tada ir tik tada, kai su visomis $\mathcal{M}$ ir $\mathcal{C}$ reikšmėmis M, C teisinga lygybė*

$$P(\mathcal{M} = M|\mathcal{C} = C) = P(\mathcal{M} = M).$$

Pastebėkime, kad šis apibrėžimas reiškia, kad atsitiktiniai dydžiai $\mathcal{M}, \mathcal{C}$ yra nepriklausomi. Tiesiog iš šio apibrėžimo ir sąlyginių tikimybių savybių išplaukia toks teiginys:

116 teorema. *Kriptosistema $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ yra visiškai saugi tada ir tik tada, kai su visomis $\mathcal{M}$ ir $\mathcal{C}$ reikšmėmis M, C teisinga lygybė*

$$P(\mathcal{C} = C | \mathcal{M} = M) = P(\mathcal{C} = C).$$

Ar egzistuoja visiškai saugios sistemos? Prisiminkime amžininkų neįvertintą G. Vernamo idėją.

Tegu $\mathcal{B} = \mathbb{F}_2 = \{0, 1\}$ – dvejetainė abėcėlė, o pranešimų, raktų ir šifrų aibės sudarytos iš šios abėcėlės n ilgio žodžių:

$$\mathcal{M} = \mathcal{K} = \mathcal{C} = \mathcal{B}^n.$$

Tegu šifravimo operacija – tiesiog žodžių sudėtis, kurią naudojome kodavimo teorijoje:

$$C = e(M|K) = M + K, \quad d(C|K) = C + K = M.$$

Jeigu visos rakto reikšmės yra vienodai galimos, t. y.

$$p(K) = \frac{1}{2^n},$$

tai tokia kriptosistema yra visiškai saugi. Iš tikrųjų, jeigu šifruojamas pranešimas yra M, tai šifras C bus gautas tik tuomet, kai bus naudojamas raktas $K = M + C$, taigi

$$P(\mathcal{C} = C | \mathcal{M} = M) = P(\mathcal{K} = M + C) = \frac{1}{2^n}.$$

Kita vertus, pasinaudoję pilnosios tikimybės formule, gauname

$$\begin{aligned} P(\mathcal{C} = C) &= \sum_{M \in \mathcal{M}} P(\mathcal{M} = M) P(\mathcal{C} = C | \mathcal{M} = M) \\ &= \sum_{M \in \mathcal{M}} P(\mathcal{M} = M) P(\mathcal{K} = M + C) = \frac{1}{2^n} \sum_{M \in \mathcal{M}} P(\mathcal{M} = M) = \frac{1}{2^n}. \end{aligned}$$

Taigi tokia kriptosistema yra visiškai saugi. Tikriausiai pastebėjote, kad analogišką visiškai saugią kriptosistemą galime sudaryti imdami vietoj dvejetainės abėcėlės bet kokį baigtinį kūną $\mathbb{F}_q$.

Tokia sistema yra visiškai saugi, jeigu raktas naudojamas tik vienam pranešimui šifruoti. Panagrinėkime atvejį, kai šios sąlygos nesilaikoma. Tarkime, raktas pakeičiamas užšifravus du pranešimus. Tada kriptoanalitikas gali savo analizę pradėti sulaukęs dviejų šifrų C_1, C_2 ir manyti, kad stebi kriptosistemą su pranešimų, šifrų ir raktų aibėmis

$$\mathcal{M} = \mathcal{C} = \mathcal{B}^{2n}, \quad \mathcal{K} = \mathcal{B}^n.$$

Prieš gaudamas šifrus kriptoanalitikas galbūt manė, kad visų 2^{2n} pranešimų tikimybės $P(M)$ yra teigiamos. Kokią informaciją apie siųstą pranešimą $M = M_1M_2$ kriptoanalitikui suteikia šifras $C = C_1C_2$? Sudėję abu šifrus, gauname

$$D = C_1 + C_2 = (M_1 + K) + (M_2 + K) = M_1 + M_2.$$

Taigi iš visų pranešimų, kurie galėjo būti siųsti, aibės kriptoanalitikas gali atmesti visus tuos, kurie neturi šios savybės, t. y. $M_1 + M_2 \neq D$. Tada nelygybė $P(M|C) > 0$ liks galioti tik 2^n visų aibės pranešimų! Gavęs šifrus kriptoanalitikas gerokai patikslino savo išankstines žinias.

Deja, sudėtingoms pranešimų aibėms visiškai saugios sistemos irgi yra neišvengiamai sudėtingos. Štaip galite interpretuoti šią teoremą.

117 teorema. *Jei kriptosistema yra visiškai saugi, tai $|\mathcal{K}| \geq |\mathcal{M}|$, t. y. raktų yra ne mažiau, negu pranešimų.*

Įrodymas. Pažymėkime M_i visus galimus pranešimus, C_j – visus galimus šifrus, kurių tikimybės teigiamos,

$$P(C_j|M_i) = P(\mathcal{C} = C_j|\mathcal{M} = M_j), \quad P(C_j) = P(\mathcal{C} = C_j).$$

Jeigu sistema visiškai saugi, tai su visais i, j

$$P(C_j|M_i) = P(C_j).$$

Tarkime, raktų yra mažiau negu pranešimų. Imkime kokį nors pranešimą M_i ir šifruokime jį visais raktais. Kadangi šifrų yra ne mažiau negu pranešimų, tai šifrai $e(M_i|K)$ nesutaps su visų galimų šifrų aibe, t. y. egzistuos šifras C_j, kad $e(M_i|K) \neq C_j$. Tai reiškia, kad $P(C_j|M_i) = 0$. Tačiau $P(C_j) > 0$. Gavome prieštarą. Kriptosistema negali būti visiškai saugi.

15.2. Kriptosistemos dydžių entropijos

Atsitiktinio dydžio entropija – tai neapibrėžtumo, kurį stebėtojas jaučia laukdamas dydžio reikšmės, matas. Geras įrankis kriptosistemos elementų savybėms reikšti!

Kriptosistemos saugumas priklauso nuo to, kiek netiesioginės informacijos apie atsitiktinį dydį $\mathcal{M}$ (ir $\mathcal{K}$) suteikia atsitikinis dydis $\mathcal{C}$. Informacijos kiekiams reikšti galime pasinaudoti entropijos sąvoka.

Su kriptosistema kriptoanalitikas sieja tris atsitiktinius dydžius $\mathcal{M}, \mathcal{K}, \mathcal{C}$, o neapibrėžtumą jų atžvilgiu prieš gaudamas šifrą ir jį gavęs gali reikšti naudodamas besąlygines ir sąlygines entropijas

$$H(\mathcal{M}),\ H(\mathcal{K}),\ H(\mathcal{C}),\ H(\mathcal{M}|\mathcal{C}),\ H(\mathcal{K}|\mathcal{C}).$$

Jeigu kriptosistema yra visiškai saugi, tai pranešimas ir šifras yra nepriklausomi atsitiktiniai dydžiai. Prisiminę entropijos savybes galime visiško saugumo sąlygą suformuluoti taip:

118 teorema. *Kriptosistema $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$ yra visiškai saugi tada ir tik tada, kai $H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M})$.*

Kriptosistemoje rakto slaptumas yra didesnis negu siunčiamų pranešimų slaptumas. Maždaug tokia yra šios teoremos prasmė.

119 teorema. *Kriptosistemos elementams $\mathcal{M}, \mathcal{K}, \mathcal{C}$ teisinga lygybė*

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{M}|\mathcal{C}) + H(\mathcal{K}|\mathcal{M}, \mathcal{C}).$$

Įrodymas. Panaudoję lygybę $H(X|Y) = H(X,Y) - H(Y)$ su $X = \mathcal{M}, Y = \mathcal{C}$, gausime

$$H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M}, \mathcal{C}) - H(\mathcal{C}).$$

Dar kartą pasinaudokime ta pačia lygybe tik dabar su $X = \mathcal{K}, Y = \langle \mathcal{M}, \mathcal{C} \rangle$:

$$\begin{aligned} H(\mathcal{K}|\mathcal{M}, \mathcal{C}) &= H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{M}, \mathcal{C}), \\ H(\mathcal{M}, \mathcal{C}) &= H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{K}|\mathcal{M}, \mathcal{C}). \end{aligned}$$

Taigi

$$H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{K}|\mathcal{M}, \mathcal{C}) - H(\mathcal{C}). \tag{77}$$

Analogiškai gauname

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{K}, \mathcal{C}) - H(\mathcal{C}) = H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{M}|\mathcal{K}, \mathcal{C}) - H(\mathcal{C}).$$

Pastebėję, kad iš $d(\mathcal{C}|\mathcal{K}) = \mathcal{M}$ išplaukia $H(\mathcal{M}|\mathcal{K}, \mathcal{C}) = 0$, paskutiniąją lygybę galime perrašyti taip:

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{M}, \mathcal{K}, \mathcal{C}) - H(\mathcal{C}). \tag{78}$$

Kad gautume teoremos tvirtinimą, belieka pasinaudoti gautuoju sąryšiu (77) lygybėje:

$$H(\mathcal{M}|\mathcal{C}) = H(\mathcal{K}|\mathcal{C}) - H(\mathcal{K}|\mathcal{M}, \mathcal{C}).$$

Skaičiuoti praktikoje naudojamų kriptosistemų dydžių entropijas yra sudėtinga. Panagrinėkime „žaislinį" pavyzdį. Ir toks pavyzdys padeda suvokti, kas vyksta sudėtingų sistemų atvejais.

Pavyzdys. Tarkime urnoje yra baltų, juodų ir raudonų rutulių. Jų yra atitinkamai $b = 5, j = 4, r = 3$. Atsitiktinai ištraukus rutulį, užrašoma pirmoji jo spalvos raidė: B, J arba R. Ta raidė – tai pranešimas. Tačiau tas pranešimas „šifruojamas", t. y. raidė keičiama kita. Šifravimui naudojami

šeši keitiniai $k_1, k_2, \ldots, k_6$, kurie parenkami su vienodomis tikimybėmis:

	B	J	R
k_1	B	J	R
k_2	B	R	J
k_3	J	B	R
k_4	J	R	B
k_5	R	B	J
k_6	R	J	B

Apskaičiuokime entropijas $H(\mathcal{M}), H(\mathcal{K}), H(\mathcal{M}|C), H(\mathcal{K}|C)$. Pranešimų B, J, R tikimybės lygios atitinkamai $\frac{5}{12}$, $\frac{4}{12}$ ir $\frac{3}{12}$, taigi

$$H(\mathcal{M}) = \frac{5}{12}\log_2\frac{12}{5} + \frac{4}{12}\log_2\frac{12}{4} + \frac{3}{12}\log_2\frac{12}{3} \approx 1.5546.$$

Nesunku rasti ir rakto entropiją;

$$H(\mathcal{K}) = \log_2 6 \approx 2.585.$$

Kiek daugiau tenka padirbėti skaičiuojant sąlygines entropijas

$$H(\mathcal{M}|\mathcal{C}) = \sum_C H(\mathcal{M}|\mathcal{C}=C)P(\mathcal{C}=C), \quad H(\mathcal{K}|\mathcal{C}) = \sum_C H(\mathcal{K}|\mathcal{C}=C)P(\mathcal{C}=C).$$

Šifrų tikimybes surasti nesunku, pasinaudojus pilnosios tikimybės formule:

$$P(\mathcal{C}=B) = P(\mathcal{C}=J) = P(\mathcal{C}=R) = \frac{5}{12}\cdot\frac{2}{6} + \frac{4}{12}\cdot\frac{2}{6} + \frac{3}{12}\cdot\frac{2}{6} = \frac{2}{6} = \frac{1}{3}.$$

Entropijai $H(\mathcal{M}|\mathcal{C}=C)$ apskaičiuoti reikia tikimybių $P(\mathcal{M}=M|\mathcal{C}=C)$. Suraskime, pavyzdžiui, $P(\mathcal{M}=B|\mathcal{C}=J)$:

$$\begin{aligned} P(\mathcal{M}=B|\mathcal{C}=J) &= \frac{P(\mathcal{M}=B,\mathcal{C}=J)}{P(\mathcal{C}=J)} = \frac{P(\mathcal{M}=B)P(\mathcal{C}=J|\mathcal{M}=B)}{P(\mathcal{C}=J)} \\ &= \frac{P(\mathcal{M}=B)\cdot\frac{2}{6}}{P(\mathcal{C}=J)} = P(\mathcal{M}=B). \end{aligned}$$

Kitais atvejais taip pat gautume, kad sąlyginės tikimybės lygios besąlyginėms:

$$P(\mathcal{M}=M|\mathcal{C}=C) = P(\mathcal{M}=M).$$

Taigi kriptosistema visiškai saugi ir $H(\mathcal{M}|\mathcal{C}) = H(\mathcal{M})$.

Tačiau sąlyginės raktų tikimybės nelygios besąlyginėms, pavyzdžiui,

$$P(\mathcal{K}=k_1|\mathcal{C}=B)=\frac{P(\mathcal{K}=k_1,\mathcal{C}=B)}{P(\mathcal{C}=B)}=\frac{\frac{1}{6}\cdot\frac{5}{12}}{\frac{1}{3}}=\frac{5}{24}.$$

Apskaičiavę visas tikimybes ir visas entropijas $H(\mathcal{K}|\mathcal{C}=C)$, gautume

$$H(\mathcal{K}|\mathcal{C})=2.5546,\quad H(\mathcal{K}|\mathcal{C})<H(\mathcal{K}).$$

Galima suvokti tokį reiškinį kad ir taip: kriptosistemos rakto entropija buvo „perteklinė", kriptosistema gali būti visiškai saugi ir su mažesne rakto entropija. Štai šiek tiek to pertekliaus ir nubyrėjo...

O dabar tarkime, kad vienas po kito iš urnos traukiami du rutuliai ir pagal jų spalvas sudaromas dviejų raidžių žodis. Žodžiui šifruoti naudojamas vienas iš šešių raktų. Raktai kaip ir anksčiau parenkami su vienodomis tikimybėmis. Dabar iš viso galimi devyni žodžiai, o raktų tik šeši. Taigi sistema visiškai saugi jau nebegali būti.

Kadangi rutuliai traukiami be grąžinimo, tai pranešimų tikimybes skaičiuojame taip:

$$P(\mathcal{M}=BB)=\frac{5\cdot4}{12\cdot11}=\frac{20}{132},\quad P(\mathcal{M}=BR)=\frac{5\cdot3}{12\cdot11}=\frac{15}{132},\ldots$$

Pranešimo besąlyginė entropija lygi

$$H(\mathcal{M})\approx3.0968.$$

Šifrų tikimybės yra dabartiniu atveju tokios:

$$\begin{aligned}P(\mathcal{C}=BB) &= P(\mathcal{C}=JJ)=P(\mathcal{C}=RR)=\frac{19}{198},\\ P(\mathcal{C}=BJ) &= \ldots=P(\mathcal{C}=RR)=\frac{47}{396}.\end{aligned}$$

Apskaičiavę gautume

$$\begin{aligned}&H(\mathcal{M}|\mathcal{C}=BB)=H(\mathcal{M}|\mathcal{C}=JJ)=H(\mathcal{M}|\mathcal{C}=RR)\approx1.433,\\ &H(\mathcal{M}|\mathcal{C}=BJ)=\ldots=H(\mathcal{M}|\mathcal{C}=RJ)\approx2.5533.\end{aligned}$$

Sąlyginė pranešimo entropija dabar yra

$$H(\mathcal{M}|\mathcal{C})=3\cdot H(\mathcal{M}|\mathcal{C}=BB)P(\mathcal{C}=BB)+6\cdot H(\mathcal{M}|\mathcal{C}=BJ)P(\mathcal{C}=BJ)\approx2.2308.$$

Taigi mūsų kriptoanalitikas, sugavęs šifrą, gauna maždaug vieną bitą informacijos apie siunčiamą užšifruotą pranešimą. Kaip jis tą bitą panaudos – ne mūsų reikalas.

15.3. Rakto įminimo taškas

Kuo ilgesnis šifras, tuo daugiau jame netiesioginės informacijos apie šifruotą pranešimą bei naudotą raktą. Kokio ilgio šifro užtenka, kad pavienio šifro ataka būtų įmanoma, t. y. kad, naudojantis juo, būtų galima nustatyti raktą? Įmanoma teoriškai, žinoma, nereiškia, kad praktiškai lengva tai padaryti.

Jeigu raktų aibė lieka ta pati, o pranešimų aibė didėja, tai kriptosistema darosi vis mažiau saugi. Tai pastebėjome ir iš ankstesnių skyrelių pavyzdžių. Panagrinėsime šią mintį detaliau.

Tarkime, siunčiami pranešimai yra parašyti lietuviškos 32 simbolių abėcėlės $\mathcal{A}$ raidėmis, o šifravimui naudojama Cezario kriptosistema, kurios raktai renkami su vienodomis tikimybėmis.

Jeigu šifruojamus pranešimus sudaro pavienės raidės, tai Cezario kriptosistema yra visiškai saugi.

Dabar tarkime, kad pranešimų aibę $\mathcal{M}_2$ sudaro dviejų raidžių lietuviški žodžiai. Gavęs šifrą $c = c_1c_2$, kriptoanalitikas gali bandyti visus raktus vieną po kito. Šitaip jis gautų 32 simbolių poras $m = d(c_1|k_i)d(c_2|k_i)$. Žinodamas, kokia kalba buvo parašytas tekstas, jis gali atmesti beprasmes kombinacijas ir gauti pranešimų bei raktų, kurie galėjo būti panaudoti, aibes.

Jeigu pranešimų aibę $\mathcal{M}_N$ sudaro N simbolių ilgio tekstai, kur N yra didelis skaičius, tai, sugavęs šifrą c_N ir išbandęs visus raktus, kriptoanalitikas ko gero gautų tik vieną galimą prasmingą tekstą $d(c_N|k_i)$. Taigi raktas vienareikšmiškai atkuriamas pagal šifrą: $\mathcal{K} = f(\mathcal{C}_N)$. Tačiau tada $H(\mathcal{K}|\mathcal{C}_N) = 0$, t. y. kriptosistemos slaptumas išnyksta, ji įmenama.

Šiame pavyzdyje padarėme prielaidą, kad kriptoanalitikas gali vertinti tekstus kaip prasmingus ir neprasmingus. Tačiau iš pavyzdžių matėme, kad kriptosistema silpnėja ir tada, kai prasmės faktoriaus nėra.

Nagrinėkime kokią nors kriptosistemą $\langle \mathcal{M}, \mathcal{K}, \mathcal{C} \rangle$, tegu šifrus sudaro abėcėlės $\mathcal{A}$ pavieniai simboliai. Interpretuodami $\mathcal{M}, \mathcal{K}, \mathcal{C}$ kaip atsitiktinius dydžius, galime apibrėžti jų entropijas $H(\mathcal{M})$, $H(\mathcal{K})$, $H(\mathcal{C})$. Kaip visada, tarsime, kad $\mathcal{M}, \mathcal{K}$ yra nepriklausomi, o $\mathcal{C} = e(\mathcal{M}|\mathcal{K})$. Taigi

$$H(\mathcal{M}, \mathcal{K}) = H(\mathcal{M}) + H(\mathcal{K}), \quad H(\mathcal{C}|\mathcal{M}, \mathcal{K}) = 0.$$

Tada

$$H(\mathcal{M}, \mathcal{K}, \mathcal{C}) = H(\mathcal{M}, \mathcal{K}) + H(\mathcal{C}|\mathcal{M}, \mathcal{K}) = H(\mathcal{M}) + H(\mathcal{K}).$$

Panaudoję šį sąryšį lygybėje (78), gausime

$$H(\mathcal{K}|\mathcal{C}) = H(\mathcal{M}) + H(\mathcal{K}) - H(\mathcal{C}). \tag{79}$$

Dabar apibrėžkime naujas kriptosistemas $\langle \mathcal{M}_N, \mathcal{K}, \mathcal{C}_N \rangle$, kur pranešimų ir šifrų aibės yra $\mathcal{M}_N = \mathcal{M}^N, \mathcal{C}_N = \mathcal{C}^N$, raktai renkami su tomis pačiomis tikimybėmis, o šifravimo procedūros apibrėžiamos taip:

$$e(m_1m_2 \dots m_N|K) = e(m_1|K)e(m_2|K) \dots e(m_N|K).$$

Tada šioms kriptosistemoms (79) lygybė irgi teisinga:

$$H(\mathcal{K}|\mathcal{C}_N) = H(\mathcal{M}_N) + H(\mathcal{K}) - H(\mathcal{C}_N).$$

Jau matėme, kad entropija $H(\mathcal{K}|\mathcal{C}_N)$ gali mažėti augant N.

95 apibrėžimas. *Jei N yra mažiausias skaičius, tenkinantis lygybę*

$$H(\mathcal{M}_N) + H(\mathcal{K}) - H(\mathcal{C}_N) = 0, \tag{80}$$

tai jį vadinsime rakto įminimo tašku.

Čia apibrėžtas rakto įminimo taškas angliškoje literatūroje vadinamas *unicity point*. Kartais vietoj lygybės reikalaujama, kad reiškinys būtų ne didesnis už 1. Jeigu $H(\mathcal{K}|\mathcal{C}_N) \leq 1$, tai iš esmės reiškia, kad raktą galima nustatyti naudojantis vien šifru.

Taigi rakto įminimo taškas – tai trumpiausio šifro, iš kurio galima nustatyti raktą, ilgis. Šį skaičių turėtume rasti iš (80) lygybės. Tačiau kaip tai padaryti? Pasielkime lyg fizikai: negalėdami išspręsti bendros lygties, suformuluokime sveikam protui priimtinas sąlygas, kurios palengvina rakto įminimo taško lygties sprendimą.

Visų pirma tarkime, kad mūsų pranešimus $\mathcal{M}_N$ generuoja šaltinis, turintis entropiją H. Tada galime manyti, kad dideliems skaičiams N

$$H(\mathcal{M}_N) = NH(\mathcal{M}) = NH.$$

Jei, pavyzdžiui, pranešimai yra kokios nors kalbos tekstai, tai H yra kalbos entropija, kurią galima suskaičiuoti naudojantis simbolių pasirodymo tikimybių lentelėmis. Toliau, tarkime, kad visi aibės $\mathcal{A}^N$ elementai turi vienodas galimybes pasirodyti kaip šifrai. Tokią savybę matėme nagrinėtuose paprastuose pavyzdžiuose. Tada

$$H(\mathcal{C}_N) = N \log_2 |\mathcal{A}|.$$

Pagaliau jei visi raktai renkami su vienodomis tikimybėmis, tai

$$H(\mathcal{K}) = \log_2 |\mathcal{K}|.$$

Su šiomis prielaidomis rakto įminimo taško lygtį (80) galima išspręsti:

$$N = \frac{\log_2 |\mathcal{K}|}{\log_2 |\mathcal{A}| - H}.$$

Gavome gerokai supaprastintos rakto įminimo lygties sprendinį. Jis pateikia praktikai reikalingą įvertį, rodantį, kada kriptosistemos naudojimas jau tampa nebesaugus. Lygties su nurodytomis sąlygomis sprendimas duoda tik apytikslę įminimo taško reikšmę[10], nei tikslus sprendimas, bet tai juk nėra didelis trūkumas.

[10] Tikslios lygtys gali iš viso neturėti sprendinių.

Rakto įminimo taškas nurodo, kokio ilgio šifrą reikia turėti, kad būtų efektyvi pavienio šifro ataka. Tačiau ši ataka atliekama vykdant perranką! Taigi praktiškai ją įvykdyti gali būti labai sudėtinga.

Panagrinėkime konkretų kriptosistemos pavyzdį. Imkime kokią nors skaičių aibės perstatą (bijekciją)

$$k : \{1, 2, , \ldots, d\} \to \{1, 2, , \ldots, d\}$$

ir apibrėžkime tokią pranešimo M, parašyto abėcėlės $\mathcal{A}$ simboliais, šifravimo taisyklę

$$\text{jei } M = m_1 m_2 \ldots m_d m_{d+1} m_{d+2} \ldots m_{2d}, \ldots,$$

$$\text{tai } e(M, k) = m_{k(1)} m_{k(2)} \ldots m_{k(d)} m_{d+k(1)} m_{d+k(2)} \ldots m_{d+k(d)} \ldots$$

Taigi pranešimas skaidomas d ilgio žodžiais ir jie vienas po kito šifruojami. Todėl galime manyti, kad teksto abėcėlė iš tikrųjų yra $\mathcal{A}^d$. Tada rakto įminimo taško reikšmė nurodys, kiek d ilgio šifro žodžių reikia raktui įminti. Kadangi raktų skaičius tokioje sistemoje yra lygus $d!$, tai iš rakto įminimo lygties gauname:

$$N = \frac{\log_2 d!}{\log_2 |\mathcal{A}^d| - H_d} = \frac{\log_2 d!}{d(\log_2 |\mathcal{A}| - H)},$$

čia H_d yra teksto, kurį interpretuojame kaip sudarytą iš d ilgio žodžių, entropija, o H – teksto, sudaryto iš pavienių simbolių, entropija, $H_d \approx dH$. Lietuviškoje abėcėlėje yra 32 raidės, todėl $\log_2 |\mathcal{A}| = 5$; lietuviškų tekstų entropija $H \approx 3$. Taigi įminimo taško reikšmė

$$N \approx \frac{\log_2 d!}{2d}.$$

Paskaičiavę su nedidelėmis d reikšmėmis (N reikšme imame mažiausią sveikąjį skaičių, didesnį už reiškinio reikšmę), gautume $N = 1$, kai $d \leq 8$, ir $N = 2$, kai $9 < d \leq 20$.

16 Blokiniai šifrai

Blokiniais šifrais vadinsime kriptosistemas, kurių šifravimo algoritmai transformuoja fiksuoto ilgio teksto žodžius (blokus) į tokio pat ilgio šifro žodžius. Raktas, valdantis šifravimo operaciją, irgi paprastai renkamas iš fiksuoto ilgio žodžių aibės.

Dvejetainė abėcėlė $\mathcal{B} = \{0, 1\}$ yra pati svarbiausia kriptografijoje, todėl tik ją ir naudosime.

96 apibrėžimas. *Kriptosistemą, kurios tekstų, šifrų ir raktų aibės yra sudarytos iš fiksuoto ilgio žodžių:* $\mathcal{M} = \mathcal{C} = \mathcal{B}^n, \mathcal{K} = \mathcal{B}^k$, *vadinsime blokine kriptosistema arba tiesiog – blokiniu šifru.*

Taigi blokinės kriptosistemos šifravimo ir dešifravimo taisyklės yra funkcijos, kurių argumentai – dvejetainių žodžių poros:

$$e(\cdot|\cdot), d(\cdot|\cdot) : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n.$$

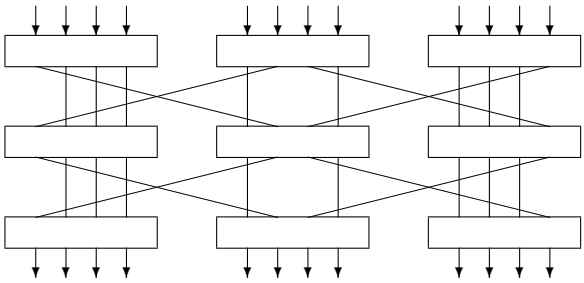
Turint blokinę kriptosistemą, reikia nuspręsti, kaip ja bus šifruojami ilgi duomenų srautai. Pasirinkti galima įvairiai. Tie būdai kriptografijoje vadinami blokinių šifrų naudojimo režimais. Svarbiausius iš jų aptarsime kiek vėliau.

Apskritai blokinės kriptosistemos gali būti tiek simetrinės, tiek viešojo rakto. Tačiau paprastai taip vadinamos simetrinės kriptosistemos.

16.1. Dvi schemos

Du svarbūs šiuolaikinių blokinių kriptosistemų projektavimo principai.

Teksto žodį galime pakeisti kitu (atlikti keitinį), galima sumaišyti jo simbolius (atlikti perstatą). Tokios pavienės transformacijos, žinoma, nesukurs saugaus šifro. Viena iš C. Shannono idėjų, kuria remiasi ir mūsų laikų kriptosistemų kūrėjai, yra labai paprasta: teksto blokui reikia taikyti tokią perstatų ir keitinių seką, kad kiekvienas gautojo šifro bitas priklausytų nuo kiekvieno teksto ir rakto bito, t. y. pakeitus net vienintelį teksto ar rakto bitą, šifro žodis labai pasikeistų. Šitaip sukonstruota daug blokinių šifrų: kol gaunamas šifras, teksto žodis praeina keletą ciklų (arba iteracijų), kur jam taikomos perstatos ir keitiniai. Kiekvieno ciklo transformacijas valdo daliniai raktai, gaunami iš kriptosistemos rakto pagal tam tikrą taisyklę. Tokia schema kriptografijoje vadinama keitinių-perstatų tinklu (Substitution-Permutation Network, (PSN) angl.).



Keitinių-perstatų tinklo schema. Dažnai cikle apdorojamas simbolių blokas skaidomas į mažesnius ir keitiniai taikomi pastariesiems. Struktūriniai schemos elementai, kurie skirti keitiniams atlikti, kriptografijoje vadinami S-dėžėmis.

Daugelio gerų blokinių šifrų konstrukcijoje taikoma Horsto Feistelio struktūrinė schema, kurią jis, dirbdamas IBM, panaudojo LUCIFER kriptosistemai. Feistelio struktūros blokiniuose šifruose transformacijos atliekamos su lyginio ilgio duomenų blokais.

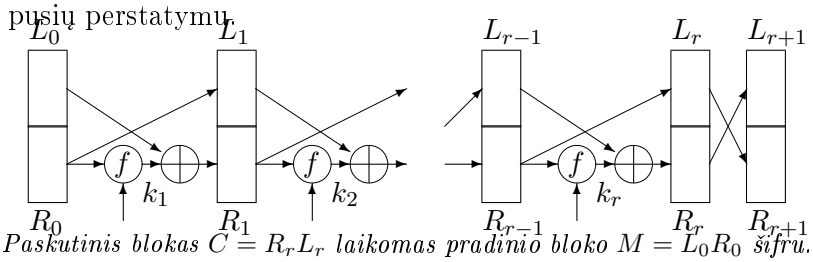
Tegu $M_0 = m_1 m_2 \dots m_n m_{n+1} \dots m_{2n}$ yra pradinis tekstas. Jo šifras gaunamas po r Feistelio iteracijų, kurias valdo daliniai raktai $k_1, k_2, \dots k_r$:

$$\begin{array}{ccccccccccc} & & \downarrow\ k_1 & & \downarrow\ k_2 & & & & \downarrow\ k_r & & \\ M = M_0 & \longrightarrow & M_1 & \longrightarrow & M_2 & \longrightarrow \dots \longrightarrow & M_{r-1} & \longrightarrow & M_r = C. \end{array}$$

Atliekant vieną iteraciją, blokas M_j dalijamas į dvi vienodo ilgio dalis – kairiąją ir dešiniąją – ir pertvarkomas taip:

$$M_i = L_i R_i \longrightarrow M_{i+1} = L_{i+1} R_{i+1}, \quad L_{i+1} = R_i,\ R_{i+1} = L_i \oplus f(R_i, k_{i+1}),$$

čia $\oplus$ žymi blokų sudėtį moduliu 2, o f yra funkcija, iš dvejetainės abėcėlės n ilgio žodžio ir rakto sukurianti naują n ilgio žodį. Naudinga r iteracijų grandinę papildyti dar vienu paprastu pertvarkiu: kairiosios ir dešiniosios blokų pusių perstatymu.



Paskutinis blokas $C = R_r L_r$ laikomas pradinio bloko $M = L_0 R_0$ šifru.

Taigi visa blokų pertvarkymo grandinė atrodo taip:

$$\begin{aligned} L_1 &= R_0,\ R_1 = L_0 \oplus f(R_0, k_1) \\ L_2 &= R_1,\ R_2 = L_1 \oplus f(R_1, k_2) \\ &\dots\dots\dots\dots\dots\dots\dots\dots \\ L_r &= R_{r-1}, R_r = L_{r-1} \oplus f(R_{r-1}, k_r) \\ L_{r+1} &= R_r, R_{r+1} = L_r. \end{aligned}$$

Kam gi reikalingas tas paskutinis žingsnis? Tuoj įsitikinsime, kad tai gudrus sumanymas.

Taigi po visų pertvarkymų gavome pradinio bloko $M = L_0R_0$ šifrą $C = L'_0R'_0 = R_rL_r$. Įsivaizduokime, kad tą patį pertvarkymų ciklą atliekame su C, tik raktus naudojame atvirkščia tvarka: $k_r, k_{r-1}, \ldots, k_1$. Atlikę pirmąjį žingsnį, gausime bloką $C_1 = L'_1R'_1$, čia

$$\begin{aligned} L'_1 &= R'_0 = L_r = R_{r-1}, \\ R'_1 &= L'_0 \oplus f(R'_0, k_r) = R_r \oplus f(R_{r-1}, k_r) \\ &= L_{r-1} \oplus f(R_{r-1}, k_r) \oplus f(R_{r-1}, k_r) = L_{r-1}. \end{aligned}$$

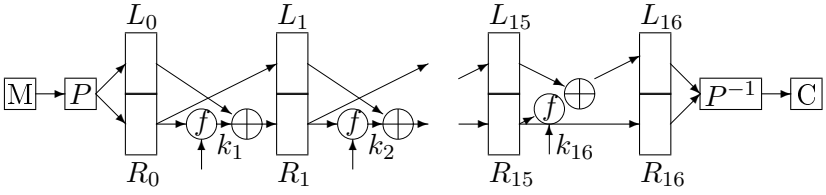
Taigi $C_1 = R_{r-1}L_{r-1}$. Atlikę r iteracijų, gautume $C_r = R_0L_0$, paskutiniame žingsnyje sukeitę bloko puses vietomis, gautume M. Šifravimo iteracijų seka tinka ir dešifravimui – tik raktus reikia naudoti „atbuline“ tvarka. Ir jokių reikalavimų funkcijai f!

16.2. DES

DES, arba Data Encryption Standard, yra kriptografijos mokslo brandos ženklas. Tai pirmoji kriptosistema valstybės institucijos įvertinta ir pripažinta šifravimo standartu.

1973 metais JAV vyriausybė nusprendė, kad metas standartizuoti informacijos srautų apdorojimo metodus. Užduotis parengti šiuos standartus teko Nacionaliniam standartų institutui (NBS – National Bureau of Standards). Vienas iš daugelio tikslų, kurie buvo keliami, – parengti duomenų šifravimo standartą. Buvo paskelbtas konkursas, bet jo rezultatai iškeltų sąlygų netenkino. Konkursas buvo pakartotas. Vieną iš pasiūlymų pateikė IBM. Siūloma kriptosistema buvo sukurta naudojantis H. Feistelio kriptosistemos LUCIFER idėjomis. Svarstymai truko pusantrų metų, o konkursas pasibaigė IBM pergale. Jų pasiūlymas tapo pirmuoju pasaulyje šifravimo standartu DES (Data Encryption Standard, patvirtinta 1976 metais).

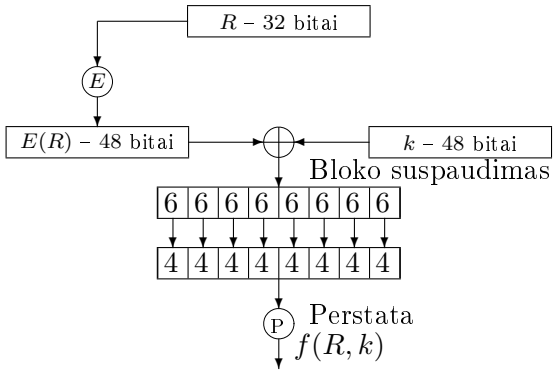
DES yra Feistelio struktūros blokinė kriptosistema, šifruojanti 64 bitų ilgio blokus ir naudojanti 56 bitų ilgio raktus. Jos struktūrinė schema yra tokia:



DES kriptosistemos schema. IP žymi tam tikrą pradinių duomenų perstatą, o IP^{-1} – šiai perstatai atvirkštinę.

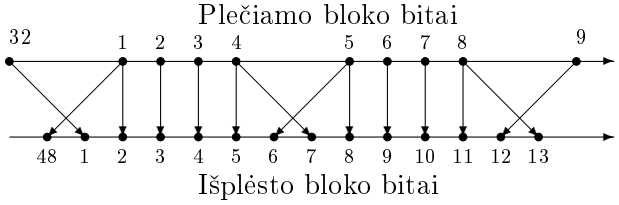
DES naudoja 16 iteracijų ir 56 bitų ilgio raktą. Formaliai rakto ilgis yra toks pat kaip ir blokų – 64 bitai. Tačiau aštuoni bitai sudarant dalinius

raktus nedalyvauja, jie yra tiesiog kontroliniai. Paskutinės iteracijos sudarytas blokas R_{16} sudaro kairiąją šifro pusę, L_{16} – dešiniąją (prisiminkime aptartąją Feistelio schemos pabaigą!), perstatos IP ir IP^{-1} yra fiksuotos, t. y. nepriklauso nuo rakto, todėl kriptosistemos saugumas nuo jų nepriklauso. Jos įtrauktos į schemą techniniais sumetimais – norėta, kad duomenys į DES mikroschemą būtų įkeliami greičiau. Taigi visa esmė glūdi f dėžėse. Kas gi jose vyksta?



Transformacijų, atliekamų DES f-dėžėje schema.

Pirmas faktas: į šią dėžę įvedamas 32 bitų ilgio duomenų blokas ir 48 bitų – dalinis raktas. Pirma operacija skirta duomenų blokui išplėsti iki tokio pat ilgio kaip raktas. Tai daroma paprastai – keletas bitų panaudojami du kartus.



Duomenų išplėtimo operacija DES dėžėje.

Tada ateina laikas pasinaudoti raktu. Išplėstas 48 bitų raktas tiesiog sudedamas moduliu 2 su daliniu raktu (XOR operacija) ir gaunamas naujas 48 bitų blokas. Tačiau iš dėžės turi išeiti 32 bitai, taigi reikia bloko ilgį sumažinti. Tiesiog nubraukti šešiolika bitų būtų prastas sprendimas. DES kūrėjai sugalvojo kitaip: 48 bitai paskirstomi po šešis ir kiekvienas šešetukas eina į savo dėžutę. Iš kiekvienos dėžutės išeina tik 4 bitai. Šitaip gaunamas reikalingo ilgio duomenų blokas. Kiekvienai iš aštuonių dėžučių DES kūrėjai sudarė 4×16 dydžio lentelę, kurios elementai – keturiais bitais užrašomi natūriniai skaičiai (taigi skaičiai nuo 1 iki 15). Jeigu į dėžutę įeina bitų šešetas $b_1 b_2 \ldots b_6$, tai skaičius $e = b_1 + 2b_6$ nurodo lentelės eilutės numerį, o

$s = b_2 + 2b_3 + 4b_4 + 8b_5$ – stulpelio. Vadinasi, iš šios dėžutės turi būti išvestas skaičius (keturi bitai) užrašytas e-ojoje eilutėje ir s-ajame stulpelyje! Štai šitaip veikia DES. Kam įdomu, kaip sudarytos tos lentelės bei kaip iš 56 bitų rakto sudaroma 16 dalinių raktų, pavartykite DES aprašymą. [11]

DES buvo naudojamas beveik trisdešimt metų. Ne tik naudojamas, tačiau ir nuodugniai tyrinėjamas. Konstrukcija pasirodė esanti tokia gera, kad iš esmės nebuvo rasta praktiškai reikšmingų saugumo spragų. Ir vis dėlto – mūsų dienomis DES nebegalima laikyti saugiu šifru. Kodėl? Nes gerokai išaugo kompiuterių galia. Paprastas raktų perrankos atakas, kurioms nepakako išteklių tuomet, kai DES buvo sukurtas, dabar jau galima vykdyti.

Perrankos ataka yra teksto-šifro porų ataka. Jeigu žinomas tam tikras kiekis nešifruotų tekstų $M_1, M_2, \ldots, M_r$ ir juos atitinkančių šifrų $C_1 = e(M_1|K)$, $C_2 = e(M_2|K), \ldots, C_r = e(M_r|K)$, gautų panaudojus nežinomą raktą K, tai galima bandyti visus galimus raktus ir tikrinti lygybes:

$$d(C_i|K) \stackrel{?}{=} M_i,\ i = 1, 2, \ldots, r;\ K \in \mathcal{K}.$$

Blogiausiu atveju tektų atlikti maždaug 2^{56} tokių tikrinimų. Tai didžiulis skaičius, tačiau dabartiniai kompiuteriai irgi labai galingi.

Tačiau ir su DES dar galima pasiekti tinkamą saugumo lygį, tiksliau su 3DES. Skaičius 3 reiškia, kad DES taikomas tris kartus:

$$C = e(d(e(M|K_1)|K_2)|K_1).$$

Jeigu $K_1 = K_2$, tai toks šifravimas tolygus DES šifravimui su vienu raktu. Kai $K_1 \neq K_2$, gauname algoritmą, kurio rakto ilgis yra 112 (raktų K_1 ir K_2 ilgių suma).

16.3. AES

1997 JAV Nacionalinis standartų ir technologijų institutas (NIST – National Institute of Standards and Technology) paskelbė naują konkursą garbingojo DES vietai užimti. Atsirado 15 kandidatų. Į finalą išėjo šie: MARS, RC6, Rijndael, Serpent ir Twofish.

O nugalėjo Rijndael. Šifro pavadinimas „Rijndael“ sudarytas sujungus jo kūrėjų – dviejų belgų kriptografų V. Rijmen ir J. Daemen – pavardžių skiemenis. Jų šifrui ir buvo suteiktas titulas AES (Advanced Encryption Standard). Standartas paskelbtas 2001 metais[12], po 5 metus trukusio vertinimo proceso. Tačiau tai nereiškia, kad tai vienintelis geras šifras iš penkių finalininkų. Kai kurie iš jų pagal atskirus kriterijus netgi geresni. Tačiau tų kriterijų

[11] FIPS 46-3, Data Encryption Standard (DES).http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[12] ADVANCED ENCRYPTION STANDARD (AES). Federal Information Processing Standards Publication 197 November 26, 2001.

buvo daug: saugumas, operacijų greitis, diegimo paprastumas, struktūrinės savybės... Iš kitų blokinių šifrų AES išsiskiria savo struktūros paprastumu bei „matematiškumu“ . Prisiminę DES dėžes, negalėtume paaiškinti, kokiais matematiniais dėsniais remtasi jas sudarant. O štai visos AES transformacijos labai paprastai aprašomos naudojantis matematiniais objektais. Tai privalumas, o galbūt ir trūkumas. Juk niekas negali užginčyti, kad sudėtingiems, bet tiksliai suformuluotiems matematiniams uždaviniams gali atsirasti ir elegantiški bei efektyvūs sprendimai. Kitaip tariant, naujos matematinės AES atakos... Tačiau kol kas tai tik svarstymai. Geriau panagrinėkime AES struktūrą.

AES kriptosistema atlieka veiksmus su baitais. Kiekvieną baitą, t. y. aštuonių bitų žodį $b = b_7b_6b_5b_4b_3b_2b_1b_0$, interpretuokime kaip erdvės $\mathbb{F}_{2,8}[x]$ daugianarį

$$b(x) = b_7x^7 + b_6x^6 + \ldots + b_1x + b_0.$$

Du tokius daugianarius sudėję, vėl gausime tos pačios erdvės daugianarį. Taigi du baitus galime sudėti; tiesą sakant, ta sudėtis – įprastinė žodžių sudėtis moduliu 2 (XOR operacija) ir tiek. Imkime aštunto laipsnio daugianarį

$$f(x) = x^8 + x^4 + x^3 + x + 1$$

ir apibrėžkime daugianarių iš $\mathbb{F}_{2,8}[x]$ (baitų) sandaugą, kaip tai darėme nagrinėdami kūnų plėtinius.

$$a(x) \times_f b(x) = a(x)b(x) \text{ dalybos iš } f(x) \text{ liekana.}$$

Taigi dabar turime du veiksmus su baitais: sudėtį ir daugybą. Su šiais veiksmais baitų aibė sudaro žiedą. O dabar gera naujiena – daugianaris $f(x)$ yra neskaidus virš kūno $\mathbb{F}_2$. Tai reiškia (žvilgtelėkite į daugianariams skirtą skyrelį), kad daugianarių erdvė $\mathbb{F}_{2,8}[x]$ su sudėties ir daugybos operacijomis sudaro kūną, turintį $2^8 = 256$ elementus. Šis kūnas yra izomorfiškas kūnui $\mathbb{F}_{2^8}$. Taigi veiksmus su baitais, kuriuos naudoja AES, galime interpretuoti kaip veiksmus su kūno $\mathbb{F}_{2^8}$ elementais.

DES kriptosistemą galima naudoti su fiksuotu duomenų bloko ir rakto ilgiu. AES yra šiuo požiūriu kiek lankstesnė: galima naudoti kelis kriptosistemos variantus su skirtingais rakto ilgiais ir skirtingu iteracijų skaičiumi. Kriptosistemos kūrėjai taip pat numatė galimybę naudoti ir įvairius duomenų bloko ilgius (128, 160, 192, 224 ir 256 bitų), tačiau standartizuotas tik variantas su 128 bitų ilgio duomenų blokais.

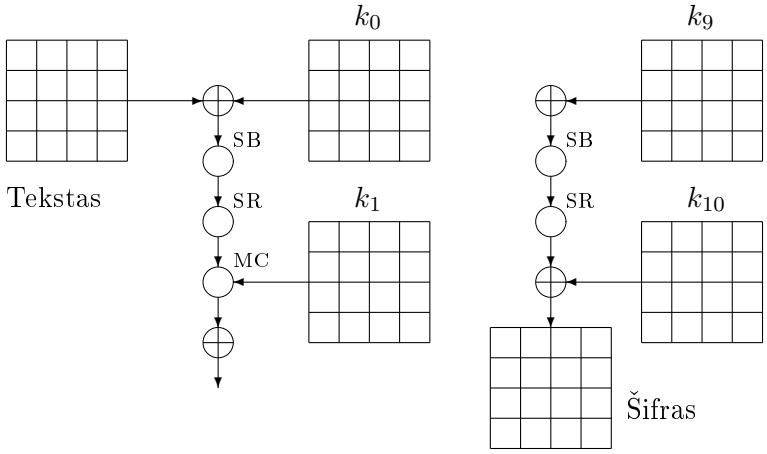
	Rakto ilgis	Bloko ilgis	Iteracijų skaičius
AES-128	128	128	10
AES-192	192	128	12
AES-256	265	128	14

Pradinių duomenų blokas prieš pradedant transformacijas surašomas į lentelę.

s_{00}	s_{01}	s_{02}	s_{03}
s_{10}	s_{11}	s_{12}	s_{13}
s_{20}	s_{21}	s_{22}	s_{23}
s_{30}	s_{31}	s_{32}	s_{33}

AES duomenų blokas; lentelės elementai – aštuonių bitų ilgio žodžiai (baitai).

Šifruojant su šia lentele atliekamos keturių rūšių operacijos: baitų keitimo, eilučių postūmio, stulpelių maišymo ir sudėties su iteracijos raktu. Dešifruojant vykdomos analogiškos (bet ne tokios pat) operacijos. Bendra AES-128 algoritmo schema atrodo taip:



AES-128 sudaro 10 vienodos struktūros žingsnių. Kiekvienai operacijai iš kriptosistemos bendro rakto sudaromas dalinis raktas. Šifravimas prasideda sudėties su pradžios raktu veiksmu. Pirmieji devyni žingsniai vienodi – atliekamos baitų keitimo (SB), eilučių postūmio (SR) ir stulpelių maišymo (MC) operacijos. Paskutiniame žingsnyje stulpelių maišymo nėra.

Baitų keitimo operacija kiekvieną lentelės baitą pakeičia nauju. Kiekvienas baitas (interpretuojamas kaip kūno $\mathbb{F}_{2^8}$ elementas) keičiamas atvirkštiniu, kuriam dar be to atliekama tam tikra nesudėtinga transformacija. Nulinis baitas neturi atvirkštinio, todėl jis nekeičiamas. Baito $b_0b_1b_2b_3b_4b_5b_6b_7$ keitimo baitu $b'_0b'_1b'_2b'_3b'_4b'_5b'_6b'_7$ operaciją galima užrašyti matricine forma

taip:

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

Baitų keitimo operacijos matricinė išraiška. Antroji matricos eilutė gauta iš pirmosios, atlikus jos elementų postūmį, analogiškai iš antrosios eilutės gaunama trečioji ir t. t.

Eilučių postūmio operacija yra pati paprasčiausia: duomenų eilutės baitai cikliškai perstumiami ir tiek.

s_{00}	s_{01}	s_{02}	s_{03}
s_{10}	s_{11}	s_{12}	s_{13}
s_{20}	s_{21}	s_{22}	s_{23}
s_{30}	s_{31}	s_{32}	s_{33}

$\longrightarrow$

s_{00}	s_{01}	s_{02}	s_{03}
s_{11}	s_{12}	s_{13}	s_{10}
s_{22}	s_{23}	s_{20}	s_{21}
s_{33}	s_{30}	s_{31}	s_{32}

Eilučių postūmio operacija

Kiek sudėtingesnė yra stulpelių maišymo operacija. Ji atliekama su kiekvienu duomenų lentelės stulpeliu.

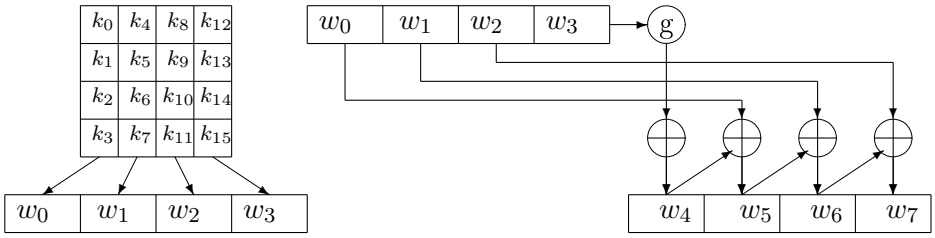
Šią operaciją galima apibrėžti naudojant daugianarių veiksmus arba matricinę lygybę

$$\begin{pmatrix} s'_{0c} \\ s'_{1c} \\ s'_{2c} \\ s'_{3c} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} s_{0c} \\ s_{1c} \\ s_{2c} \\ s_{3c} \end{pmatrix}.$$

AES stulpelių maišymo operacijos matricinė išraiška. Matricos elementus reikia interpretuoti kaip šešioliktaine sistema užrašytus baitus. Baitų daugyba atliekama interpretuojant juos kaip kūno $\mathbb{F}_{2^8}$ elementus.

Paskutinė – sudėties su raktu – operacija – tai paprasta matricų su elementais iš $\mathbb{F}_{2^8}$ sudėties operacija.

Štai ir viskas. Trūksta tik dalinių raktų sudarymo iš kriptosistemos rakto algoritmo ir galima pradėti programuoti!



Vienas rakto išplėtimo algoritmo ciklas. Pradinis AES raktas išsaugomas žodžiuose w_0, w_1, w_2, w_3, o naudojantis jais sukuriami dar keturi žodžiai

Pradinis AES 128 bitų raktas išsaugomas keturiuose 32 bitų žodžiuose w_0, w_1, w_2, w_3, naudojantis jais, sukuriami dar keturi žodžiai, naudojantis pastaraisiais – dar keturi... Iš viso sukuriami 44 žodžiai $w_i, i = 0, 1, \ldots, 43$. Pradinei sudėties su raktu operacijai panaudojami pirmieji keturi žodžiai, antrajai sudėčiai raktas sudaromas iš sekančių žodžių ir t. t. Rakto išplėtimo schemoje naudojama funkcija g. Ji veikia taip:

$$g(w) = SB(rot(w)) + R_j,$$

čia rot baitų postūmio operacija, $rot(b_0b_1b_2b_3) = b_1b_2b_3b_0$; SB – baitų keitimo operacija, apibrėžta anksčiau, o R_j – j-ojo žingsnio konstanta. Šios konstantos sudaromos pagal paprastą taisyklę (baitus dauginame interpretuodami juos kaip $\mathbb{F}_{2^8}$ elementus):

$$R_j = r_j000000, \quad r_1 = 01, \quad r_j = 2 \cdot r_{j-1}.$$

Dabar tai jau tikrai viskas apie AES. Kuo vadovavosi šios kriptosistemos kūrėjai, kodėl panaudojo tokias operacijas, o ne kitokias? Tikriausiai išbandė ne vieną. Pagrindinis kriterijus – kad kriptosistema atlaikytų visas žinomas atakas ir, žinoma, veiktų greitai. Struktūros paprastumas irgi svarbus kriterijus. Juk sudėtingas schemas sudėtinga ir analizuoti, vadinasi, sudėtinga ir vertinti.

16.4. Penki režimai

Turite gerą blokinį šifrą? Laikas nuspręsti, kam ir kaip jį naudoti...

Blokinė kriptosistema šifruoja ir dešifruoja vieną fiksuoto ilgio žodį:

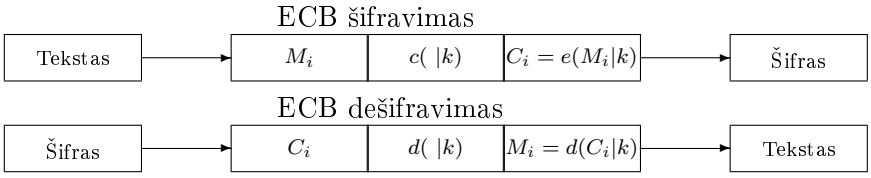
$$C = e(M|K), \; M = d(C|K) \quad M, C \in \{0,1\}^n.$$

Tačiau duomenis, kuriuos norime apsaugoti, sudaro daugybė žodžių. Kaip taikyti kriptosistemą tokiam srautui? Kriptografai žino keletą būdų.

Elektroninė kodų knyga (ECB, Electronic Codebook)

Tai pats paprasčiausias blokinės kriptosistemos naudojimo būdas. Savo duomenų srautą skaidote į vienodo ilgio žodžius ir juos vieną po kito šifruojate:

$$M_1 M_2 \ldots \mapsto C_1 C_2 \ldots, \quad C_j = e(M_j|K).$$



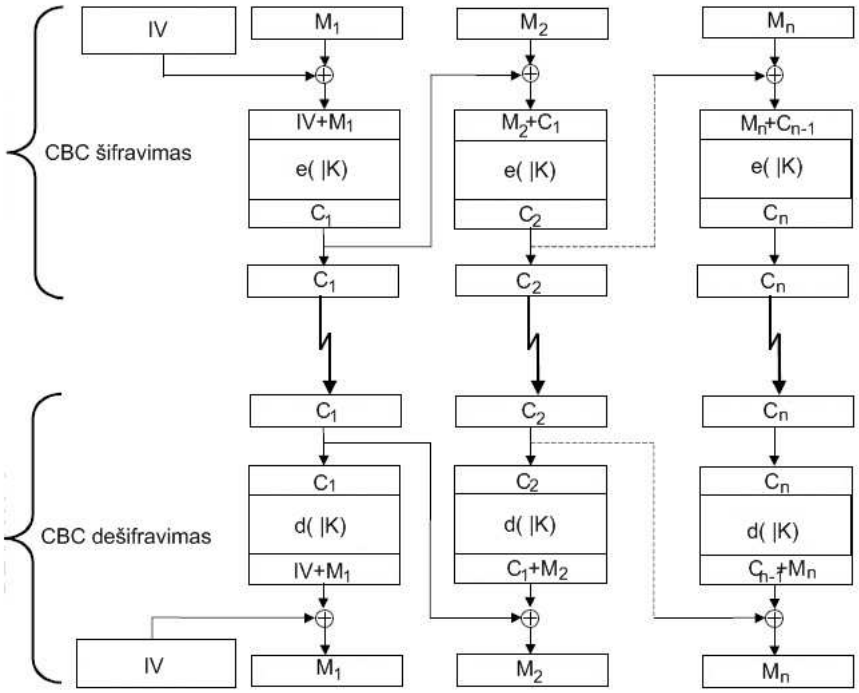
Duomenų šifravimas ir dešifravimas blokine kriptosistema ECB režimu.

Paprastumas – turbūt vienintelis šio režimo privalumas. O trūkumų daug. Pavyzdžiui, vienodi duomenų srauto žodžiai visada užšifruojami vienodai. Jeigu kas nors pašalins kelis šifro srauto žodžius – nepastebėsite. Žymūs kriptografai N. Fergusonas ir B. Shneieris savo knygoje „Praktinė kriptografija“ rašo taip: minime jį tik todėl, kad žinotumėte, kokio režimo niekada nereikia naudoti.

Šifrų blokų grandinės režimas (CBC, Cipher Block Chaining)

Naudojantis šiuo režimu, kiekvieno žodžio M_i šifras priklauso nuo anksčiau šifruotų žodžių:

$$\begin{aligned} C_1 &= e(M_1 \oplus IV|K),\ C_i = e(M_i \oplus C_{i-1}|K),\ i \geq 2, \\ M_1 &= d(C_1 \oplus K|IV), M_i = e(C_i|K) \oplus C_{i-1},\ i \geq 2. \end{aligned}$$



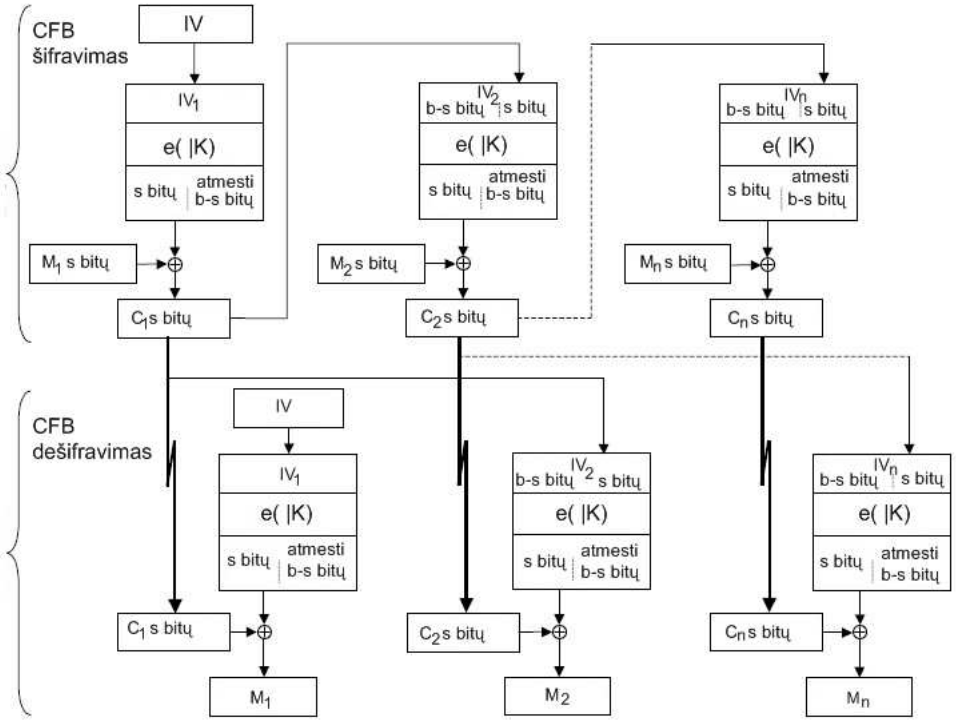
Duomenų šifravimas ir dešifravimas blokine kriptosistema CBC režimu.

Pirmajam blokui šifruoti naudojamas žodis IV – pradžios (kriptografai sako – inicializacijos) vektorius. Taigi šifro gavėjas turi žinoti ne tik raktą, bet ir inicializacijos vektorių.

Jeigu naudojamasi vienu iš šių režimų, tai šifravimo įrenginys perskaito visą šifruojamą bloką, paskui jį šifruoja ir tada pasiunčia kanalu arba įrašo į sudaromą šifro failą. Padidinto saugumo reikalavimų atveju gali būti neleidžiama perskaityti ir nors trumpam saugoti šifruojamą bloką. Duomenys turi būti skaitomi bitas po bito, bitai šifruojami ir iš karto siunčiami į kanalą. Tokiam atvejui, pavyzdžiui, tinka

Šifro atgalinio ryšio režimas (CFB, Cipher Feedback)

Naudojantis šiuo režimu duomenų srautas gali būti šifruojamas „blokeliais“ po s, $1 \leq s \leq n$, bitų, čia n yra naudojamo blokinio šifro bloko ilgis. Duomenys šifruojami sudedant moduliu 2 (kitaip tariant, atliekant XOR operaciją) s ilgio duomenų blokelius su tokio pat ilgio blokeliais, kuriuos generuoja kriptosistema. Darbo pradžiai taip pat reikalingas inicializacijos vektorius.



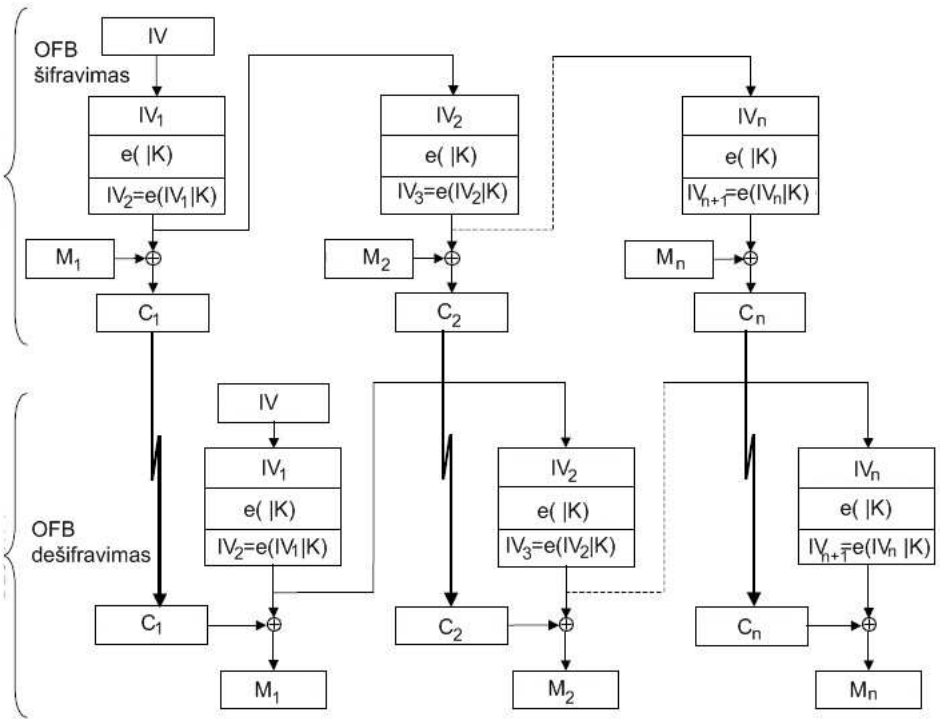
Duomenų šifravimas ir dešifravimas blokine kriptosistema CFB režimu.

Pirmajame žingsnyje blokinė kriptosistema šifruoja inicializacijos vektorių. Gautojo šifro s bitų panaudojama XOR operacijai, kiti atmetami. Tada pradinis inicializacijos vektorius pakeičiamas: pirmieji jo s bitų atmetami, o prie sutrumpinto vektoriaus prijungiama s gautojo šifro bitų. Galima įsivaizduoti, kad šifro bitai išstumia pirmuosius s inicializacijos vektoriaus bitų. Analogiškai inicializacijos bitai keičiami ir kituose žingsniuose. Dešifruojama lygiai taip pat kaip šifruojama. Tai net paprasčiau negu Feistelio šifre – net raktų tvarkos nereikia keisti.

Panašus į šį režimą yra

Srauto atgalinio ryšio režimas (OFB, Output Feedback)

Esminis skirtumas tik tas, kad srautas, kuris sudaromas XOR operacijai su šifruojamais duomenimis, generuoja pats save, taigi šifras nebenaudojamas.

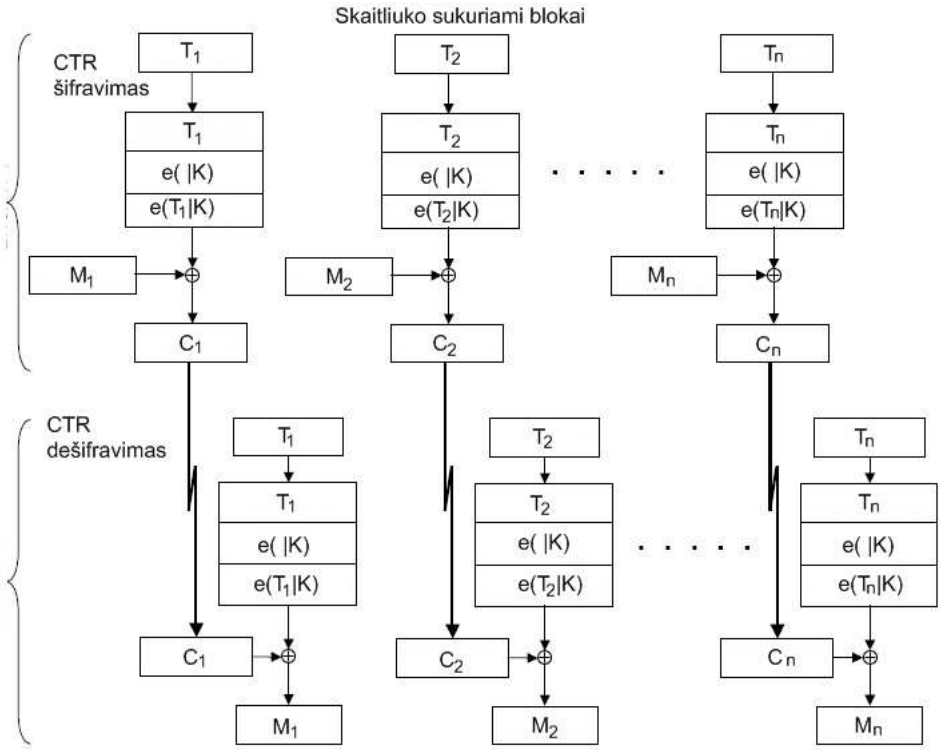


Duomenų šifravimas ir dešifravimas blokine kriptosistema OFB režimu.

Vienas šio režimo privalumų – bitų srautą, kuris naudojamas XOR operacijoje su duomenų srautu, galima generuoti iš anksto, t. y. neturint nei duomenų, nei šifro.

Skaitliuko režimas (CTR, Counter)

Šios paprastos schemos esmė tokia: blokinis šifras naudojamas blokams, kuriuos generuoja koks nors atskiras įrenginys (skaitiklis), šifruoti. Gautieji šifro žodžiai naudojami XOR operacijai su duomenų srautu.



Duomenų šifravimas ir dešifravimas blokine kriptosistema CTR režimu

Reikalaujama, kad skaitiklio generuojami blokai būtų skirtingi (žinoma, po daugelio žingsnių pasikartojimai neišvengiami). Paprasčiausias pavyzdys: dvejetainės skaičių $g, g+1, g+2, \ldots$ išraiškos. Šifravimas ir dešifravimas ir šiame režime yra vienodos operacijos. Tačiau reikia, kad šifro gavėjas turėtų taip pat veikiantį skaitiklį kaip siuntėjo.

Skaitytojas, žinoma, atkreipė dėmesį į tai, kad paskutinieji trys režimai priverčia blokinę kriptosistemą veikti kitaip. Iš tiesų, šie režimai iš blokinių kriptosistemų sukuria srautinius šifrus.

Režimų, žinoma, yra ir daugiau. Ir jūs galite sugalvoti savo režimą. Čia aptartieji režimai pasirinkti todėl, kad jie yra įtraukti į oficialų JAV Nacionalinio standartų ir technologijų instituto patvirtintą standartą[13].

[13]Morris Dworkin. Recommendation for BlockCipher Modes of Operation Methods and Techniques. NIST Special Publication 800-38A 2001 Edition.

17 Srautiniai šifrai

Blokinių kriptosistemų kūrėjai sprendžia klausimą: kaip transformuoti duomenų srautą, kad gautume šifrą? Srautinių kriptosistemų kūrėjai atsako į šį klausimą pačiu paprasčiausiu būdu.

Tegu $M = m_1 m_2 \ldots$ dvejetainės abėcėlės simbolių srautas, generuokime tokio pat ilgio rakto žodį $K = x_1 x_2 \ldots$ ir apibrėžkime M šifrą taip:

$$C = e(M|K) = c_1 c_2 \ldots, \quad c_i = m_i \oplus x_i,\ i = 1, 2, \ldots$$

Taigi teksto šifras – tai jo ir rakto srauto XOR operacijos rezultatas. Rakto simbolių srautas tiesiog paslepia teksto simbolius. Dešifravimas – ta pati XOR operacija, tik ją atlikti reikia su šifro ir rakto srautais:

$$M = d(C|K) = m_1 m_2 \ldots, \quad m_i = c_i \oplus x_i,\ i = 1, 2, \ldots$$

Tačiau šiuo atsakymu klausimai tik prasideda. Kaip sukurti tą rakto srautą K? Jeigu generuosime rakto srautą visiškai atsitiktinai ir naudosime jį tik vieną kartą, realizuosime Vernamo kriptosistemą. Tačiau tuomet bus reikalingas saugus kanalas raktui perduoti. Ar gavėjas negalėtų pats generuoti rakto srauto?

Taigi priartėjame prie pseudoatsitiktinių bitų srauto generavimo idėjos. Reikia sugalvoti algoritmą, kuris iš fiksuoto ilgio žodžio $k = k_1 k_2 \ldots k_m$ sukurtų reikiamo ilgio rakto srautą K:

$$k_1 k_2 \ldots k_m = k \longrightarrow K = x_1 x_2 \ldots$$

Šį srautą galėtume naudoti XOR operacijai su teksto simbolių srautu. Rakto srautas jau nebebus sudarytas iš atsitiktinai generuotų bitų, taigi jį vadinsime pseudoatsitiktinių bitų seka. Kuo ši seka panašesnė į tikrai atsitiktinę, tuo geriau.

Taigi svarbiausias srautinių šifrų kūrėjų rūpestis – kaip generuoti pseudoatsitiktinių bitų srautus?

17.1. Golombo pseudoatsitiktinės sekos

> *Tikrai atsitiktinę bitų seką gautume mėtydami simetrišką monetą ir užsirašydami rezultatus. Tai idealus, bet nepraktiškas būdas. Yra daug geresnių. Tačiau kaip nuspręsti, ar sudarytas bitų srautas yra panašus į atsitiktinį?*

Tarkime, sugalvojome kokį nors būdą generuoti nepriklausomų vienodai pasiskirsčiusių atsitiktinių dydžių $X_1, X_2, \ldots$

$$P(X_i = 0) = P(X_i = 1) = \frac{1}{2},$$

reikšmes. Naudodamiesi šiuo būdu, gavome n reikšmių

$$x_1x_2\dots x_n, \quad x_i \in \{0,1\}. \tag{81}$$

Gali pasitaikyti taip, kad, atlikę tiesiog tobulas dydžių reikšmių generavimo operacijas, mes gausime, pavyzdžiui, tokią bitų seką: $00011\dots11$. Ir niekas nepatikės, kad tai nepriklausomų atsitiktinių dydžių sekos reikšmės! Tačiau taip įvyks labai retai. Dažniausiai gausime sekas, turinčias tam tikras tipines savybes. Norėdami pagrįsti, kodėl tos savybės yra tipinės, turėtume formuluoti matematinius teiginius apie atsitiktinių dydžių tikimybes.

Pavyzdžiui, viena iš tokių tipinių savybių (už jos slypi svarbus tikimybių teorijos teiginys – didžiųjų skaičių dėsnis) yra tokia:

• *maždaug pusė (81) sekos elementų yra vienetai, kiti – nuliai.*

Vieną ar kelias tipines atsitiktinių dydžių reikšmių sekos savybes gali turėti labai „taisyklingos“ sekos. Pavyzdžiui, suformuluotą savybę turės seka, kuri sudaryta iš dviejų vienodo ilgio blokų: $00\dots0$ ir $11\dots1$. Todėl tiriant, ar bitų seka panaši į atsitiktinę, reikia pasitelkti daugiau kriterijų.

Pavadinkime r nulių grupę, „saugomą“ dviejų vienetų, (t. y. grupę $10\dots01$) 10_r1 bloku, o grupę $01\dots10$, kurioje nulių apsuptyje yra r vienetų, – 01_r0 vienetų bloku. Jeigu sekos pirmieji r nariai lygūs nuliui, o po jų seka vienetas, tai tokią seką vadinsime 0_r1 bloku; jeigu seka baigiasi r nuliais, tokią pabaigą vadinsime 10_r bloku. Analogiškai apibrėšime 1_r0 ir 01_r blokus. Blokus $10_r1, 0_r1, 10_r$ vadinsime tiesiog 0_r blokais. Analogiškai apibrėšime 1_r blokus.

Tarkime, N yra bendras (81) sekos blokų skaičius, t. y.

$$N = 0_1 \text{ skaičius } + 1_1 \text{ skaičius } + 0_2 \text{ skaičius } + \dots$$

Tada tipinėje sekoje

• *blokų skaičiai tenkintų lygybes:*

$$0_1 \textit{ skaičius } + 1_1 \textit{ skaičius } \approx \frac{N}{2}, \quad 0_2 \textit{ skaičius } + 1_2 \textit{ skaičius } \approx \frac{N}{2^2} \textit{ ir t.t.}$$

Jeigu sugretintume (81) su cikliškai pastumta seka

$$\begin{array}{ccccc} x_1 & x_2 & x_3 & \dots & x_n \\ x_t & x_{t+1} & x_{t+2} & \dots & x_{n+t-1} \end{array}$$

ir suskaičiuotume, kiek yra pozicijų su vienodais simboliais ir kiek su skirtingais, tai tipinėje sekoje

• *sutapimų ir nesutapimų būtų beveik po lygiai.*

Šios trys tipinės savybės panaudotos bandant apibūdinti bitų sekas, panašias į atsitiktinai generuotas.

97 apibrėžimas. *Tegu* $x_1x_2\dots x_n, \quad x_i \in \{0,1\}$, *yra bitų seka,* $\nu(1), \nu(0), \nu(0_r), \nu(1_r)$ *yra atitinkamai vienetų, nulių ir* r *ilgio blokų skaičiai*

šioje sekoje, o N – bendras blokų skaičius. Seką vadinsime pseudoatsitiktine Golombo seka, jeigu ji tenkina tokias sąlygas:

1) $|\nu(1) - \nu(0)| \le 1$;

2) $\nu(0_1) + \nu(1_1) \ge \dfrac{N}{2},\ \ldots,\ \nu(0_r) + \nu(1_r) \ge \dfrac{N}{2^r},\ \ldots$;

3) *dydis*

$$C(t) = \sum_{i=1}^{n}(2x_i - 1)(2x_{i+t} - 1), \quad t = 0, 1, \ldots, n-1,\ x_{k+n} = x_k, k \ge 1$$

įgyja tik dvi reikšmes: $C(0) = n,\ C(t) = m\ (t \ne 0)$.

Antrojoje sąlygoje nelygybes turi tenkinti, žinoma, tik sekoje egzistuojančių blokų kiekiai. Be to, dar pageidautina, kad būtų $\nu(0_r) \approx \nu(1_r)$. Trečioji sąlyga reikalauja, kad sutapimų ir nesutapimų skaičių skirtumas būtų nedidelis (lygus m). Golombo sąlygos yra gana griežtos. Tačiau jeigu seka tenkina griežtas sąlygas, tai ją nedrąsu vadinti atsitiktine. Golombo sąlygas tenkinančias sekas galima sukonstruoti. Pavyzdžiui, seka

011001000111101

yra pati tikriausia Golombo seka. Bendras blokų skaičius joje $N = 8$.

17.2. Statistiniai testai

Vertindami bitų seką, sprendžiame klausimą, ar ji tinka kriptografijos reikmėms, ar ne. Hipotezių tikrinimo terminologija ir metodai – geri įrankiai ieškant atsakymo.

Dauguma kriterijų (testų), kurie taikomi tiriamai bitų sekai, siekiant nustatyti, ar ji yra panaši į tipinę nepriklausomų atsitiktinių dydžių generuotą reikšmių seką, naudoja tam tikrus tikimybių teorijos rezultatus ir iš matematinės statistikos pasiskolintą hipotezių tikrinimo schemą. Trumpai aptarkime šio metodo esmę.

Tarkime, $X_1, X_2, \ldots, X_n$ yra „tikri“ nepriklausomi atsitiktiniai dydžiai, su vienodomis tikimybėmis įgyjantys reikšmes 0 ir 1. Sudaromas naujas atsitiktinis dydis (paprastai vadinamas tiesiog statistika)

$$T = T(X_1, X_2, \ldots, X_n),$$

kurio reikšmių pasiskirstymas paklūsta gerai žinomiems ir ištyrinėtiems dėsniams. Svarbiausi ir dažniausiai pasitaikantys iš jų – standartinis normalusis dėsnis $\mathcal{N}(0,1)$ ir $\chi^2(m)$ – chi-kvadrat dėsnis su m laisvės laipsnių (čia $m \ge 1$, taigi turime visą dėsnių šeimą). Visi šie dydžiai yra absoliučiai tolydūs, t. y. turi tikimybinius tankius – neneigiamas funkcijas, kuriomis reiškiamos su dydžiais susijusios tikimybės:

$$P(X<u) = \int_{-\infty}^{\infty} p_X(x)\mathrm{d}x, \quad p_X(x) = \frac{1}{\sqrt{2\pi}}\mathrm{e}^{-\frac{x^2}{2}},$$
$$P(Y<u) = \int_{0}^{\infty} p_Y(x)\mathrm{d}x, \quad p_Y(x) = \frac{1}{2^{n/2-1}\Gamma(n/2)}x^{n-1}\mathrm{e}^{-\frac{x^2}{2}},\ x>0,$$

čia $\Gamma(t)$ yra speciali funkcija (kai t natūrinis skaičius, tai $\Gamma(t)=(t-1)!$, todėl ją galima interpretuoti kaip faktorialo tęsinį). Dydis Y neįgyja neigiamų reikšmių, todėl $P(Y<u)=0$, kai $u<0$.

Sudarius geras savybes turinčią statistiką $T(X_1,X_2,\ldots,X_n)$, galima konstruoti testą, kurį taikant bitų sekai

$$\mathbf{x} = x_1x_2\ldots x_n, \tag{82}$$

priimama viena iš dviejų hipotezių:

H_0 : **x yra** tipinė dydžių $X_1,X_2,\ldots,X_n$ generuota seka,
H_1 : **x nėra** tipinė dydžių $X_1,X_2,\ldots,X_n$ generuota seka.

Jeigu priimama hipotezė H_0, sakome, kad seka praėjo testą, t. y. jos panašumas į atsitiktinai generuotą seką yra patvirtintas. Sprendimas priimamas apskaičiavus statistikos reikšmę

$$t = T(x_1,x_2,\ldots,x_n).$$

Testui taikyti dar reikia pasirinkti nedidelį teigiamą skaičių α, vadinamą reikšmingumo lygmeniu, (pavyzdžiui, $\alpha = 0,01; 0,005$) ir, naudojantis juo, sudaryti reikšmių aibę, į kurią patekus $t=T(x_1,x_2,\ldots,x_n)$, hipotezė H_0 yra atmetama. Ši sritis matematinėje statistikoje vadinama kritine sritimi.

Jeigu statistika T pasiskirsčiusi pagal standartinį normalinį dėsnį, tai hipotezė H_0 atmetama, jei

$$|T(x_1,x_2,\ldots,x_n)| \geq z_\alpha,$$

čia z_α randamas iš lygybės

$$P(|X|>z_\alpha)=\alpha, \quad X\sim\mathcal{N}(0,1).$$

Jeigu statistika T pasiskirsčiusi pagal chi-kvadrat dėsnį su m laisvės laipsnių, tai hipotezė H_0 atmetama, jei

$$T(x_1,x_2,\ldots,x_n) \geq z_{n,\alpha},$$

$z_{n,\alpha}$ randamas iš lygybės

$$P(X>z_{n,\alpha})=\alpha, \quad X\sim\chi^2(m).$$

Štai tokia statistinių testų ideologija. Reikia pabrėžti, kad apskritai statistikos $T(X_1, X_2, \ldots, X_n)$ pasiskirstymo dėsnis iš tiesų niekada nesutampa nei su standartiniu normaliuoju, nei su chi-kvadrat dėsniu. Tačiau kai n yra pakankamai didelis skaičius, skirstiniai yra panašūs, todėl, juos tiesiog sutapatinant, paklaidos neturi praktinės reikšmės.

O dabar liko išsiaiškinti, kaip konstruoti tas statistikas T, t. y. naujus gerus atsitiktinius dydžius iš nepriklausomų vienodai pasiskirsčiusių atsitiktinių dydžių sekos

$$X_1, X_2, \ldots, \quad X_i \in \{0,1\},\ P(X_i = 0) = P(X_i = 1) = \frac{1}{2}. \tag{83}$$

Statistinių testų iš tiesų yra sugalvota labai įvairių. Jei prireiks, teks paieškoti jiems skirtų publikacijų[14]. O mes aptarkime tik penkis.

Pavienių bitų testas

Tegu N_0 yra nulių skaičius atsitiktinių dydžių sekos (82) reikšmių aibėje, o N_1 – vienetų. Tada dydžio

$$T_1 = \frac{(N_1 - N_0)^2}{n}$$

pasiskirstymo dėsnis, kai $n \geq 10$ yra labai artimas dėsniui $\chi^2(1)$.

Taigi seka „praeis“ šį testą, jeigu joje vienetų ir nulių kiekiai bus panašūs. Galime sugretinti testą su pirmuoju Golombo reikalavimu. Kuris iš jų rodo daugiau tolerancijos?

Bitų porų testas

Tegu dydžių N_0, N_1 reikšmės yra tos pačios kaip pavienių bitų teste, o $N_{00}, N_{01}, N_{10}, N_{11}$ yra atitinkamai bitų blokų $00, 01, 10, 11$ kiekiai (82) reikšmių sekoje. Kadangi poros gali turėti vieną bendrą bitą, tai

$$N_{00} + N_{01} + N_{10} + N_{11} = n - 1.$$

Apibrėžkime statistiką

$$T_2 = \frac{4}{n-1}\left(N_{00}^2 + N_{01}^2 + N_{10}^2 + N_{11}^2\right) - \frac{2}{n}\left(N_0^2 + N_1^2\right) + 1.$$

Tikimybių teorija garantuoja, kad su $n \geq 21$ statistikos T_2 pasiskirstymo dėsnis artimas dėsniui $\chi^2(2)$.

Pokerio testas

Atliekant pavienių bitų testą, skaičiuojami nulių ir vienetų kiekiai reikšmių sekoje. Kodėl nepaskaičiavus, kiek kartų pasikartoja ilgesni žodžiai?

Fiksuokime m ($m < n$); visų skirtingų žodžių aibė yra $\mathbb{F}_2^m = \{a_1, a_2, \ldots, a_{2^m}\}$. Padalykime (82) reikšmių seką į $k = \left[\frac{n}{m}\right]$ m ilgio žodžių ir, peržiūrėję

[14] Pavyzdžiui, NIST rekomendacijos apie statistinių testų taikymą, žr. [66].

juos, nustatykime, kiek kartų pasitaiko žodžiai $a_1, a_2, \ldots, a_{2^m}$. Šių žodžių pasitaikymo kiekius pažymėkime $N_1, N_2, \ldots, N_{2^m}$. Dabar jau galime sudaryti statistiką

$$T_3 = \frac{2^m}{k} \sum_{i=1}^{2^m} N_i^2 - k.$$

Jeigu $k \geq 5 \cdot 2^m$, tai T_3 pasiskirstymo dėsnis artimas $\chi^2(2^m - 1)$.

Blokų testas

Antroji Golombo pseudoatsitiktinės sekos apibrėžimo sąlyga reikalauja, kad sekoje blokai pasirodytų gana dažnai. Pažymėkime F_r blokų 10_r1 skaičių (82) reikšmių sekoje, o G_r blokų 01_r0 skaičių. Pažymėkime

$$E_i = \frac{n - i + 3}{2^{i+2}}$$

ir k – didžiausią natūrinį skaičių, su kuriuo $E_k \geq 5$. Statistiką apibrėšime taip:

$$T_4 = \sum_{i=1}^{k} \left\{ \frac{(F_i - E_i)^2}{E_i} + \frac{(G_i - E_i)^2}{E_i} \right\}.$$

Statistikos T_4 pasiskirstymo dėsnis yra artimas $\chi^2(2k - 2)$.

Autokoreliacijos testas

Trečioji Golombo sąlyga kelia griežtą reikalavimą sutampančių bitų kiekiui, kada lyginame pradinę seką su paslinkta. Autokoreliacijos testas taip pat yra reikalavimas sutapimams, tačiau jį lengviau įvykdyti.

Tegu $1 \leq d \leq [n/2]$ ir

$$X(d) = \sum_{i=1}^{n-d+1} X_i \oplus X_{i+d-1}.$$

Dydis X_d yra lygus nesutampančių bitų skaičiui, kai lyginame atsitiktinių dydžių (83) reikšmių seką su ja pačia, paslinkta per $d-1$ poziciją į kairę (x_1 lyginamas su x_d). Jeigu $n - d \geq 10$, tai statistikos

$$T_5 = \frac{2X(d) - n + d}{\sqrt{n - d}}$$

pasiskirstymo dėsnis yra artimas standartiniam normaliajam dėsniui $\mathcal{N}(0, 1)$.

17.3. Tiesinių registrų sistemos

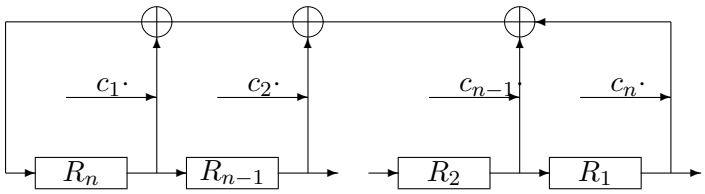
Vienas paprasčiausių instrumentų bitų srautams, panašiems į atsitiktinius, generuoti – tiesinių registrų įrenginys. Juo naudojamasi ir kodavimo teorijoje, ir kriptografijoje... todėl, kad juo paprasta naudotis.

Yra daug būdų generuoti bitų sekas, kurios tenkintų atsitiktinumo testus. Tikriausiai pats paprasčiausias ir geriausiai išnagrinėtas yra tiesinių registrų metodas.

Tarkime, kad sistemą sudaro n sujungtų tarpusavyje įrenginių, kuriuos vadinsime registrais ir žymėsime $R_1, \ldots, R_n$. Kiekvienas registras gali saugoti vieną informacijos bitą bei jį perduoti kitam registrui. Kiekvieno registro R_i turinį laiko momentu t žymėsime $x_i(t)$, $x_i(t) \in \{0,1\}$, $t = 0,1,2,\ldots$ Tegu pradinį sistemos būvį nusako vektorius

$$x(0) = \langle x_1(0), \ldots, x_n(0) \rangle.$$

Registrų sistemos turinio kitimą laikui bėgant nusakysime rekurenčiosiomis lygybėmis.



Tiesinių registrų sistema. Kiekviename žingsnyje atliekamas registrų bitų postūmis: pirmojo registro bitu papildomas generuojamas bitų srautas, antrojo registro bitas perrašomas į pirmąjį registrą, ..., n-ojo – į $n-1$-ąjį, o į n-ąjį registrą įrašoma registrų bitų tiesinė kombinacija.

Jei

$$x(t) = \langle x_1(t), \ldots, x_n(t) \rangle$$

yra sistemos padėtis laiko momentu t, tai sekančiame žingsnyje bus atlikti tokie veiksmai

$$x_i(t) = x_{i+1}(t), \quad 1 \le i \le n-1, \tag{84}$$

$$x_n(t+1) \equiv c_1 x_n(t) + \ldots + c_n x_1(t) \pmod 2, \tag{85}$$

čia $c_i \in \{0,1\}$, $1 \le i \le n$. Sakysime, kad $c_n \neq 0$, nes priešingu atveju registrų sistemą galėtume sutrumpinti. Tokia registrų sistema yra techniškai lengvai realizuojama: net ir daug registrų turintis įrenginys yra kompaktiškas ir dirba greitai.

Registrų sistemai veikiant, registrų turiniai nuolat keičiasi. Galime įsivaizduoti, kad kiekviename žingsnyje žodis $x(t)$ (dvejetainis vektorius) yra atvaizduo-

jamas į žodį $x(t+1)$. Atvaizdį galime užrašyti naudodami matricas taip:

$$x(t+1) = x(t)C, \quad C = \begin{pmatrix} 0 & 0 & \dots & 0 & c_n \\ 1 & 0 & \dots & 0 & c_{n-1} \\ 0 & 1 & \dots & 0 & c_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & c_1 \end{pmatrix}.$$

Ši lygybė apibrėžia tiesinį atvaizdį $\mathbb{F}_2^n \to \mathbb{F}_2^n$. Kadangi matrica yra neišsigimusi, tai atvaizdis yra abipus vienareikšmis, t. y. bijekcija.

Aptarę tiesinių registrų sistemos konstrukciją, panagrinėkime jos darbo rezultatą – pseudoatsitiktinę seką

$$x_1(0), x_1(1), x_1(2), ... \tag{86}$$

98 apibrėžimas. *Elementų seką $\{y_i\}$ $(i = 0, 1, \dots)$ vadinsime periodine, jeigu egzistuoja toks natūralusis skaičius p, kad su visais $i \geq 0$ teisinga lygybė $y_{i+p} = y_i$. Mažiausią natūralųjį skaičių p, su kuriuo visos lygybės teisingos vadinsime sekos periodu.*

Tiesinių registrų seka gali generuoti tik periodines sekas.

120 teorema. *Tiesinė n registrų sistema generuoja periodines sekas, kurių periodas yra ne didesnis už $2^n - 1$.*

Įrodymas. Jei $x(0)$ (pradinė registrų sistemos padėtis) yra nulinis vektorius, tai teoremos tvirtinimas akivaizdus. Todėl toliau sakysime, kad $x(0) \neq \langle 0, \dots, 0 \rangle$.

Pasinaudoję lygybe $x(s+1) = x(s)C$ t kartų, gausime

$$x(t) = x(t-1)C = x(t-2)C^2 = \dots = x(0)C^t.$$

Pažymėję $m = 2^n - 1$, galime tvirtinti, kad vektorių

$$x(0),\ x(0)C,\ ...,\ x(0)C^m$$

sekoje būtinai bus pasikartojimų, nes iš viso yra tik m skirtingų nenulinių n-mačių vektorių. Taigi galima rasti tokius s ir t, kad $0 \leq s < s+t \leq m$ ir

$$x(0)C^s = x(0)C^{s+t}.$$

Kadangi neišsigimusi matrica C^s turi atvirkštinę matricą C^{-s}, tai

$$x(t) = x(0)C^t = x(0)C^{s+t}C^{-s} = x(0).$$

Dabar su bet kokiu $r \geq 0$ gausime

$$x(r+t) = x(0)C^{r+t} = x(0)C^tC^r = x(t)C^r = x(0)C^r = x(r).$$

Tai rodo, kad generuojama seka yra periodinė, o jos periodas yra ne didesnis už $t, t \leq m = 2^n - 1$.

Ar visada galima sukonstruoti tiesinę n registrų sistemą, kuri generuotų maksimalaus periodo $m = 2^n - 1$ seką? Tikėtis teigiamo atsakymo į šį klausimą leidžia kad ir toks pavyzdys.

Pavyzdys. Nagrinėkime keturių registrų sistemą, kurioje

$$c_1 = c_4 = 1,\ c_2 = c_3 = 0,\quad x(0) = \langle 1, 0, 1, 0\rangle.$$

Paeiliui skaičiuodami sistemos būsenas, gausime: $x(1) = \langle 1, 1, 0, 1\rangle$, $x(2) = \langle 0, 1, 1, 0\rangle$, $x(3) = \langle 0, 0, 1, 1\rangle$ ir t. t., kol pasieksime $x(15) = x(0) = \langle 1, 0, 1, 0\rangle$. Taigi tokia registrų sistema generuoja maksimalaus periodo seką. Kokios gi tiesinių registrų sistemos generuoja tokias sekas? Pirmiausia prisiminkime vieną sąvoką.

99 apibrėžimas. *n-ojo laipsnio daugianaris $f(x) \in \mathbb{F}_2[x]$ vadinamas primityviuoju, jeigu jis yra neskaidus ir nėra jokio daugianario $x^d + 1$ su $d < 2^n - 1$ daliklis.*

Rasti n-ojo laipsnio primityviuosius daugianarius nėra paprastas algebros uždavinys. Tačiau žinoma, kad visiems natūraliesiems n jie egzistuoja.

100 apibrėžimas. *Tiesinės registrų sistemos, nusakytos (85) lygybėmis charakteringuoju daugianariu vadinsime daugianarį*

$$P_n(x) = 1 + c_1 x + \ldots + c_n x^n,\ c_n \neq 0.$$

O dabar – atsakymas į suformuluotą klausimą.

121 teorema. *Tiesinė registrų sistema generuoja maksimalaus periodo seką tada ir tik tada, kai jos charakteringasis daugianaris yra primityvus.*

Jeigu jau ketiname naudoti tiesinių registrų sistemą pseudoatsitiktinių bitų srautui generuoti, tai verta pasirūpinti, kad šio srauto periodas būtų maksimalus. O kaipgi su atsitiktinumo testais? Štai viena tiesinių registrų sistemos generuoto srauto savybė:

122 teorema. *Jeigu tiesinių registrų sistema generuoja maksimalaus periodo m seką, tai bet kuri šios sekos m ilgio atkarpa yra Golombo pseudoatsitiktinių bitų seka.*

Pasirodo, yra be galo daug ir kiek norima ilgų sekų, kurios tenkina griežtuosius Golombo reikalavimus! Be to, jas ne taip jau sudėtinga ir sukonstruoti.

Įrodysime, kad maksimalaus periodo tiesinių registrų sistemos generuota seka tenkina pirmąsias dvi Golombo sąlygas. Tai išplauks iš tokio teiginio:

123 teorema. *Tegu tiesinės registrų sistemos generuotos sekos periodas yra maksimalus ir lygus $m = 2^n - 1$. Tuomet bet kurioje šios sekos m ilgio atkarpoje yra:*

1. $2^{n-1}-1$ *nulių ir* 2^{n-1} *vienetų;*

2. 2^{n-t-2} *blokų* 10_t1 *ir tiek pat blokų* 01_t0 *kiekvienam* t, $1 \le t \le n-2$.

Įrodymas. n registrų sistemos padėtį kiekviename žingsnyje galima išreikšti n-ženkliu dvejetainiu skaičiumi, priklausančiu intervalui $[1, 2^n-1]$. Kiekvienas eilinis generuotos sekos elementas bus lygus pirmojo registro turiniui, t. y. šio dvejetainio skaičiaus pirmajam skaitmeniui. Kadangi registrų sistema generuoja maksimalaus periodo seką, tai ji turi pereiti visas galimas padėtis nuo 1 iki 2^n-1. Tačiau tarp šių skaičių yra $2^{n-1}-1$ lyginių (paskutinis bitas 0) ir 2^{n-1} nelyginių (paskutinis bitas 1). Iš čia išplaukia pirmasis teiginys.

Kad sekoje pasirodytų t ilgio vienetų blokas, reikia, kad kuriame nors žingsnyje registrų sistemos padėtis būtų $011\ldots10x_1\ldots x_{n-t-2}$, čia $x_i \in \{0,1\}$. Iš viso tokių padėčių yra 2^{n-t-2}. Taigi tiek t ilgio vienetų blokų ir bus viename sekos periode. Analogiškai randamas ir t ilgio nulių blokų skaičius.

Taigi maksimalaus periodo m ilgio sekoje bendras blokų $10_r1, 01_r0$ skaičius yra

$$2(2^{n-3}+2^{n-2}+\ldots+1) = 2^{n-1}-2.$$

Pridėję dvejetą (sekos pradžios ir pabaigos blokai), gauname, kad bendras blokų skaičius lygus $N = 2^{n-1}$. Dabar jau nesudėtinga įsitikinti, kad antroji Golombo sąlyga yra patenkinta.

Pažiūrėkime į (86) seką kriptoanalitiko akimis. Tarkime, kad pranešimas yra dvejetainė seka $m_1m_2\ldots$ Ją užšifravę, gausime šifrą $y_1y_2\ldots$, čia

$$y_i \equiv m_i + x_i \pmod 2, \quad i = 1, 2, \ldots$$

Žinodami m_i ir y_i, nesunkiai randame x_i:

$$x_i \equiv m_i + y_i \pmod 2.$$

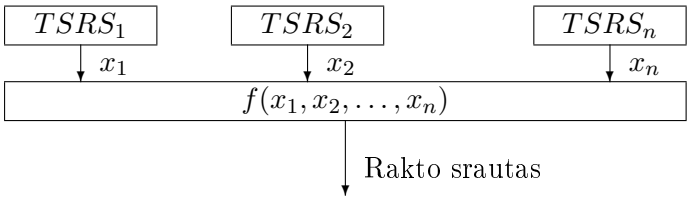
Suradę bet kuriuos $2n$ iš eilės einančius simbolius x_i, galėsime sudaryti n tiesinių lygčių su nežinomaisiais $c_1, \ldots, c_n$. Jas išsprendę rasime registrų sistemos koeficientus ir galėsime atkurti visą rakto srautą. Taigi kriptosistemą nesudėtinga įveikti turint pakankamai ilgą teksto ir jo šifro fragmentą.

Išvada. *Jeigu vienkartinio rakto kriptosistemoje raktas generuojamas tiesinių registrų sistema, tai pavienės teksto-šifro poros ataka tokia kriptosistema yra įveikiama.*

17.4. Iš paprastų – sudėtingesni

Keletas idėjų, kaip, jungiant paprastus pseudoatsitiktinių skaičių generatorius, galima sukurti sudėtingesnes rakto srauto generavimo sistemas.

Tiesinių registrų sistemos nėra vienintelis būdas generuoti pseudoatsitiktines bitų sekas. Tačiau tai lengviausiai realizuojamas būdas ir todėl dažnai naudojamas. Pagrindinis tokių sistemų trūkumas – generuoto srauto tiesiškumas, t. y. tiesinė generuoto simbolio priklausomybė nuo ankstesniųjų. Vienas iš būdų panaikinti tiesiškumą – kelių tiesinių registrų sistemų generatorių jungimas į sudėtingesnę sistemą.



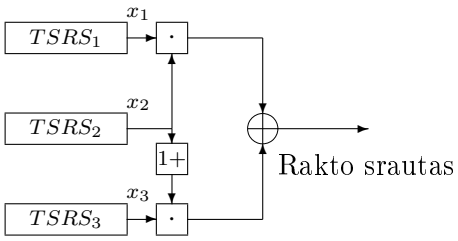
Kelių tiesinių registrų sistemų lygiagretus jungimas

Rakto srauto bitus generuoja Būlio funkcija $f(x_1, x_2, \ldots, x_n)$, ją visada galima užrašyti kaip n kintamųjų daugianarį. Pavyzdžiui, tris tiesinių registrų sistemas jungiančiame Geffe generatoriuje

$$f(x_1, x_2, x_3) = x_1 x_2 \oplus (1 \oplus x_2) x_3.$$

Jeigu Geffe schemoje visos sujungtos tiesinių registrų sistemos generuoja maksimalaus periodo sekas, o registrų kiekiai sistemose L_1, L_2, L_3 yra tarpusavyje pirminiai skaičiai, tai Geffe generatorius sukuria srautą, kurio periodas yra

$$(2^{L_1} - 1)(2^{L_2} - 1)(2^{L_3} - 1). \tag{87}$$



Geffe generatorius

Tačiau šioje schemoje yra vienas plika akimi nematomas trūkumas, kuris gali palengvinti tokios schemos kriptoanalizę.

Tarkime, turime pakankamai ilgą Geffe generatoriaus sukurtą rakto srauto fragmentą ir galime naudotis pačiu generatoriumi. Tikslas – sužinoti, kokie buvo registrų turiniai, kai buvo sudaromas rakto srautas, kurio fragmentą pavyko gauti. Sužinoję iš viso $L_1 + L_2 + L_3$ bitų, galėtume patys generuoti kokio tik reikia ilgio rakto srautą ir dešifruoti su juo gautą šifrą. Galima būtų tiesiog bandyti visus variantus: užpildžius registrus, generuoti raktų srautą ir lyginti jį su jau turimu. Iš viso blogiausiu atveju tektų atlikti (87)

tikrinimų. Tačiau kiekvieną iš trijų sistemų, pasirodo, galima „lukštenti“ skyrium. Pažymėkime $x_1(t), x_2(t), x_3(t)$ registrų sistemų generuotus bitus t žingsnyje, o $k(t)$ – iš jų gautą rakto srauto bitą. Tarkime, kad visos trys sistemos generuoja pakankamai „atsitiktinumo savybių“ turinčius srautus, pavyzdžiui, tarkime, kad $x_i(t)$ ir $x_j(t)$ kai, $i \neq j$, yra nepriklausomi ir

$$P(x_i(t) = 0) = P(x_i(t) = 1) = \frac{1}{2}.$$

Nesunku įsitikinti, kad

$$P(k(t) = x_1(t)) = P(x_2(t) = 1) + P(x_2(t) = 0)P(x_3(t) = x_1(t)) = \frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2} = \frac{3}{4}.$$

Taigi pirmosios sistemos generuotas bitas labai dažnai sutampa su rakto bitu. Tai teisinga ir trečiosios sistemos generuotam srautui:

$$P(k(t) = x_3(t)) = \frac{3}{4}.$$

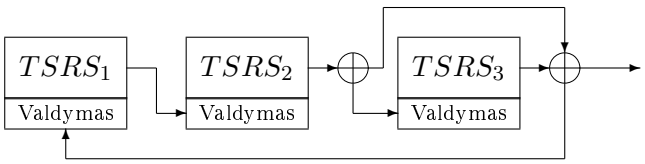
Nustačius šią savybę, schemos ataką galima vykdyti taip: užpildžius pirmosios sistemos registrus ir generavus srautą, reikia jį lyginti su rakto srautu ir tikrinti, ar sutapimų yra maždaug trys ketvirtadaliai. Jeigu taip – pirmosios registrų sistemos raktas įspėtas. Blogiausiu atveju prireiks $2^{L_1} - 1$ tikrinimų. Analogiškai galima nustatyti ir trečiosios sistemos pradines registruose saugotas reikšmes, o paskui ir antrosios sistemos. Taigi blogiausiu atveju prireiks

$$(2^{L_1} - 1) + (2^{L_2} - 1) + (2^{L_3} - 1)$$

tikrinimų, tai yra daug mažiau už (87) dydį.

Šis paprastas pavyzdys demonstruoja koreliacinių atakų, kurios taikomos srautinėms kriptosistemoms, esmę: jeigu yra priklausomybė tarp rakto srauto ir vieno schemos elemento generuoto srauto, tai tokią priklausomybę galima panaudoti daliai rakto bitų nustatyti.

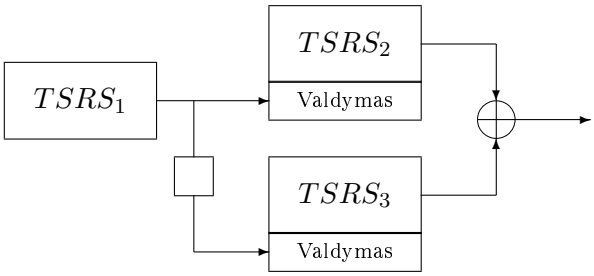
Panagrinėkime kelias nuoseklaus tiesinių registrų sistemų jungimo idėjas. Golmano generatorių grandinėje kiekviena sistema „klauso“ pirmesnės sistemos.



Golmano tiesinių registrų sistemų grandinė. Jeigu į tiesinės registrų sistemos „valdymo“ skyrių perduoto bito reikšmė lygi vienetui, sistema generuoja eilinį bitą ir atlieka registrų turinių postūmį; jeigu valdymo bitas lygus nuliui, registrų turiniai nepasikeičia. Rakto srautas – dviejų paskutinių registrų bitų suma.

Jeigu, pavyzdžiui, pirmoji sistema t žingsnyje generuoja $x_1(t) = 0$, tai antroji sistema, generavusi savo simbolį, nepakeičia registrų būsenų, jeigu $x_1(t) = 1$ – pakeičia. Analogišku būdu antroji sistema valdo trečiąją ir t.t.

Vienos registrų sistemos generuotas srautas gali būti naudojamas ne vienai, bet kelioms kitoms registrų sistemoms valdyti.



Pirmosios registrų sistemos generuotas srautas valdo antrosios ir trečiosios sistemų būsenų kaitą. Jeigu $x_1(t) = 1$, tai iš naujo užpildomi antrosios sistemos registrai, o trečiosios – lieka kaip buvę. Jeigu $x_1(t) = 0$, keičiasi trečiosios sistemos registrų turiniai, o antroji sistema „nepajuda“.

Dar viena dviejų sistemų derinimo idėja: pirmosios sistemos generuotą srautą galima naudoti antrosios srautui „praretinti“. Pavyzdžiui, jeigu pirmoji sistema generavo srautą

$$x_1 = 0100111001100011001\ldots,$$

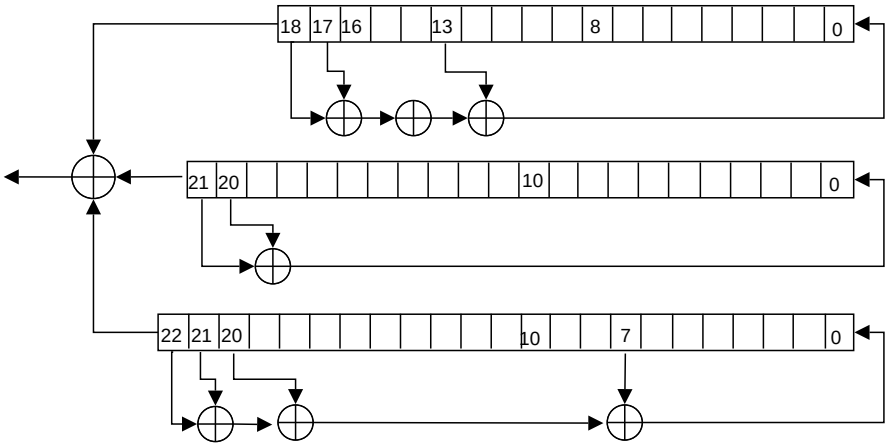
tai rakto srautas bus sudaromas iš antrosios sistemos generuoto srauto bitų:

$$k = x_2(2)x_2(5)x_2(6)x_2(7)x_2(10)x_2(11)x_2(15)x_2(16)x_2(19)\ldots$$

17.5. A5 ir Bluetooth E0

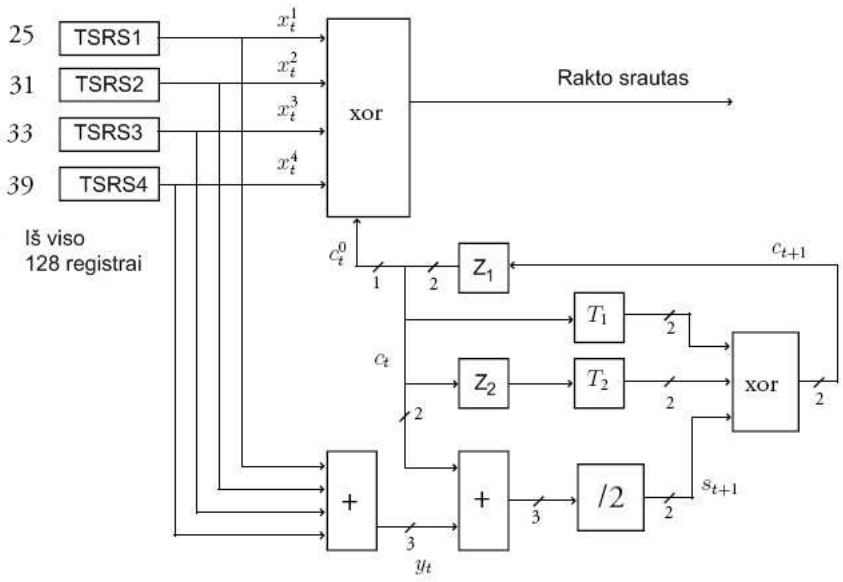
Operacijos su bitais srautinių šifrų sistemose vykdomos greitai ir nereikalauja didelių išteklių. Todėl jie naudojami tose ryšio priemonėse, kur reikia šifruoti daug ir greitai. Pavyzdžiui, bevieliam kompiuterių tinklui arba pokalbiams mobiliaisiais telefonais apsaugoti.

Srautinių šifrų yra daug ir įvairių. Panagrinėsime dvi šiuolaikiniuose ryšiuose naudojamas kriptosistemas. A5 sistema naudojama pokalbiams telefonu šifruoti, o Bluetooth E0 – bevielio kompiuterių tinklo protokoluose.



A5/1 kriptosistema

Srautinio šifro A5/1 rakto srautas sukuriamas atlikus sudėties moduliu 2 operaciją su bitais, kuriuos generuoja trys tiesinių registrų sistemos. Tačiau kiekviena iš šių sistemų nebūtinai kiekviename žingsnyje pakeičia savo būseną, t. y. iš naujo suformuoja registrų turinius. Sudarius rakto srauto bitą fiksuojami pirmosios sistemos aštuntojo registro, antrosios ir trečiosios sistemų – dešimtųjų registrų bitai. Pavyzdžiui, tegu šie bitai yra $0, 1, 0$. Nulių yra daugiau, todėl pirmosios ir trečiosios sistemų registrų turiniai yra pakeičiami, o antrosios – lieka kaip buvę. Jeigu visuose registruose būtų nuliai arba vienetai – visų trijų sistemų registrai būtų perkrauti.



Bluetooth E0 srautinio šifro schema

Bluetooth E0 rakto srauto bitai k_t gaunami atlikus sudėties moduliu operaciją su keturių tiesinių registrų generatorių pateiktais bitais ir vienu sistemos vidinės būsenos bitu:

$$k_t = x_t^1 \oplus x_t^2 \oplus x_t^3 \oplus x_t^4 \oplus c_t^0.$$

Kartu sudėjus iš viso yra 128 registrai, kiek jų kiekviename iš generatorių matyti iš jų charakteringųjų daugianarių, kurie visi yra primityvūs:

$$\begin{aligned} p_1(x) &= x^{25} + x^{20} + x^{12} + x^8 + 1, \\ p_2(x) &= x^{31} + x^{24} + x^{16} + x^{12} + 1, \\ p_3(x) &= x^{33} + x^{28} + x^{24} + x^4 + 1, \\ p_4(x) &= x^{39} + x^{36} + x^{28} + x^4 + 1. \end{aligned}$$

Taigi kiekviena iš keturių tiesinių registrų sistemų generuoja maksimalaus periodo pseudoatsitiktines sekas. Tiesinių registrų sistemos „išorėje“, dėžutėse Z_1 ir Z_2, saugomi keturi sistemos vidinės atminties bitai. Dėžutėje Z_1 saugoma bitų pora $c_t = c_t^0 c_t^1$, o dėžutėje $\mathbb{Z}_2$ – bitų pora $c_{t-1} = c_{t-1}^0 c_{t-1}^1$. Bitas c_t^0 naudojamas t žingsnio rakto srauto bitui sudaryti, o kiti – vidinei būsenai pakeisti. Vidinė būsena t žingsnyje pasikeičia taip: dėžutės Z_2 bitų pora $c_{t-1} = c_{t-1}^0 c_{t-1}^1$ pakeičiama į $c_t = c_t^0 c_t^1$, o į dėžutę Z_1 įkeliami du bitai $c_{t+1} = c_{t+1}^0 c_{t+1}^1$, gauti atlikus XOR operaciją įrenginyje X_2. Ši operacija atliekama su bitų porų trejetu. Pirmoji iš jų – per T_1 perduodama pora $c_t = c_t^0 c_t^1$. Antrąją iš dėžutės Z_2 perduotos poros $c_{t-1} = c_{t-1}^0 c_{t-1}^1$ sukuria įrenginys T_2. Jame bitų pora transformuojama taip:

$$T_2 : z_0 z_1 \mapsto z_2 z_0, \quad z_2 = z_0 \oplus z_1.$$

Daugiausia vargo dėl trečiosios poros, kurią pažymėsime $s_{t+1} = s_{t+1}^0 s_{t+1}^1$. Pirmiausia sudedant keturių registrų bitus kaip natūraliuosius skaičius, sudaromas trimis bitais užrašomas skaičius

$$y_t = x_t^1 + x_t^2 + x_t^3 + x_t^4.$$

Tada, pasinaudojus dėžutės Z_1 bitais, sudaromas skaičius $c_t = c_t^0 + 2c_t^1$ ir suskaičiuojama suma $y_t + c_t$. Paskutinė operacija jau sukuria trečiąją bitų porą:

$$s_{t+1} = s_{t+1}^0 s_{t+1}^1 = \left[\frac{y_t + c_t}{2} \right].$$

18 Sudėtingumo teorijos pradmenys

Dešifruodamas šifrą, teisėtas gavėjas naudojasi raktu ir sprendžia nesunkų uždavinį, o neteisėtas, bandydamas iššifruoti šifrą be rakto, – sunkų. Jeigu jums tokio paaiškinimo užtenka, galite šio skyriaus ir neskaityti. Tačiau jeigu norėtumėte sužinoti, pagal kokius požymius nesunkūs uždaviniai skiriami nuo sunkių – tada nepraleiskite ir šių puslapių. Tuo labiau kad vienoje nuošalioje šio skyriaus vietelėje yra paaiškinta, kaip galima užsidirbti milijoną dolerių.

18.1. Uždaviniai ir jų sprendimai

Uždavinį sudaro sąlyga ir klausimas. Sąlygą žinome, o atsakymo į klausimą – ne. Uždavinio sprendimas – tai planas, kuris visada nuo sąlygos duomenų nuveda iki atsakymo. Uždavinio sudėtingumas matuojamas šio plano ilgiu. Štai toks trumpas šio skyrelio turinys. Prisiminkite tai, jeigu paklysite matematiniuose tankumynuose!

Tiek teisėtas šifro gavėjas, tiek kriptoanalitikas dešifruodami sprendžia uždavinį: kokį pranešimą atitinka šifras? Gavėjui, turinčiam kriptosistemos raktą, ši problema nesunki, o kriptoanalitikui gali būti net teoriškai neišsprendžiama (kaip vienkartinio rakto kriptosistemos atveju). Šiame skyriuje nagrinėsime tik turinčius sprendimus uždavinius. Tačiau uždavinių būna įvairių: vieni turi vieną, kiti – kelis sprendimus. Kita vertus, ką reiškia išspręsti uždavinį? Tad iš pradžių pabandykime formalizuoti uždavinio ir jo sprendimo sąvokas.

Uždavinį paprastai sudaro sąlygos duomenys (dažnai vadinsime juos įeities duomenimis, santrumpa ID) ir užduotis arba klausimas (santrumpa U), į kurį reikia pateikti atsakymą. Tiek įeities duomenis, tiek atsakymą reikia užrašyti kokios nors abėcėlės simboliais. Dvejetainės abėcėlės $\mathcal{B} = \{0, 1\}$ šiam tikslui visiškai pakaks. Visų įmanomų šios abėcėlės žodžių aibę žymėsime, kaip visada, $\mathcal{B}^*$, $\mathcal{B}^* \times \mathcal{B}^*$ reiškia visų žodžių porų aibę.

101 apibrėžimas. *Uždaviniu vadinsime aibę $\pi \subset \mathcal{B}^* \times \mathcal{B}^*$, turinčią savybę: kiekvienam $\mathbf{x} \in \mathcal{B}^*$ atsiras $\mathbf{y} \in \mathcal{B}^*$, kad $\langle \mathbf{x}, \mathbf{y} \rangle \in \pi$. Jei $\langle \mathbf{x}, \mathbf{y} \rangle \in \pi$, tai $\mathbf{x}$ vadinsime įeities duomenimis, o $\mathbf{y}$ – atsakymu.*

Taigi įeities duomenys pateikiami dvinarės abėcėlės žodžiu. Kiekvienu konkrečiu atveju tokią duomenų pateiktį nebūna sudėtinga sudaryti. Akivaizdu, pavyzdžiui, kaip nulių-vienetų žodžiais užrašyti natūraliuosius skaičius, šiek tiek reikia pagalvoti, kaip pateikti orientuotus grafus. Gali atrodyti keista, kad bet kuris žodis įeina į uždavinį kaip įeities duomenys. Juk kad ir kokį, pavyzdžiui, grafų užrašymo būdą naudotume, ne visi dvinarės abėcėlės žodžiai reikš egzistuojančią struktūrą. Tačiau tokiais atvejais galime numatyti, kad atsakymas A reiškia, kad įeities duomenys neturi prasmės.

102 apibrėžimas. *Uždavinį $\pi \subset \mathcal{B}^* \times \mathcal{B}^*$ vadinsime funkcija, jei kiekvienam $\mathbf{x} \in \mathcal{B}^*$ atsiras vienintelis $\mathbf{y} \in \mathcal{B}^*$, kad $\langle \mathbf{x}, \mathbf{y} \rangle \in \pi$. Jei galimos tik dvi atsakymo reikšmės, funkciją vadinsime išsprendžiamumo uždaviniu.*

Išsprendžiamumo uždavinio atsakymo reikšmes interpretuosime kaip teigiamą bei neigiamą atsakymus T („taip“, arba galima žymėti tiesiog 1) ir N (ne, arba 0).

Visus uždavinius formaliai galima pertvarkyti į išsprendžiamumo uždavinius, t. y. kiekvienam uždaviniui galima rasti „orakulą“, kuris į pateikiamus klausimus atsakinėja tik „taip“ ir „ne“, bet iš tokių jo atsakymų jūs galite sukurti savo uždavinio sprendimą.

Apsiribosime natūraliojo skaičiaus skaidymo pirminiais daugikliais (faktorizacijos) uždavinio pavyzdžiu:

ID: *natūralusis skaičius n*;

U: *rasti netrivialų n daliklį arba nustatyti, kad tokio daliklio nėra.*

Šitaip suformuluotas uždavinys nėra, žinoma, išsprendžiamumo uždavinys. Tarkime, turime gerą algoritmą tokiam išsprendžiamumo uždaviniui spręsti:

ID: *natūralieji skaičiai n, k*;

U: *ar egzistuoja n daliklis m, tenkinantis nelygybę $m \geq k$?*

Pastarasis uždavinys yra išsprendžiamumo uždavinys, juk prašoma atsakyti tik „taip“ arba „ne“. Pasinaudosime šio uždavinio sprendimo algoritmu skaičiaus skaidiniui rasti.

Apsiribokime skaitiniu pavyzdžiu. Tarkime, reikia rasti netrivialų $n = 63$ daliklį m. Kadangi $n < 2^6$, tai daliklį, jeigu jis egzistuoja, galima užrašyti taip:

$$m = \epsilon_5 \cdot 2^5 + \epsilon_4 \cdot 2^4 + \epsilon_3 \cdot 2^3 + \epsilon_2 \cdot 2^2 + \epsilon_1 \cdot 2^1 + \epsilon_0,$$

čia $\epsilon_i \in \{0, 1\}$. Žymėkime $A(n, k)$ atsakymą, kurį duoda antrojo uždavinio algoritmas įeities duomenims n, k.

Kadangi $A(n, 2^5) = N$, tai galime teigti, kad $\epsilon_5 = 0$; $A(n, 2^4) = T$, tai $\epsilon_4 = 1$; $A(n, 2^4 + 2^3) = N$, todėl $\epsilon_3 = 0$; $A(n, 2^4 + 2^2) = T$, taigi $\epsilon_2 = 1$; $A(n, 2^4 + 2^2 + 2) = N$, $\epsilon_1 = 0$; pagaliau $A(n, 2^4 + 2^2 + 1) = T$, $\epsilon_0 = 1$. Ieškomas netrivialus daliklis toks:

$$m = 2^4 + 2^2 + 1 = 31.$$

103 apibrėžimas. *Išsprendžiamumo uždavinio $\pi \subset \mathcal{B}^* \times \mathcal{B}^*$ kalba vadinsime poaibį*

$$\mathcal{L}_\pi = \{\mathbf{x} : \mathbf{x} \in \mathcal{B}^*, \langle \mathbf{x}, T \rangle \in \pi\}.$$

Taigi išsprendžiamumo uždavinį galime interpretuoti kaip klausimą, ar žodis $\mathbf{x} \in \mathcal{B}^*$ įeina į kalbą $\mathcal{L}_\pi$.

Uždavinio sprendimą intuityviai suvokiame kaip tam tikrų nurodymų rinkinį, kuris valdo procesą, pasibaigiantį atsakymu (kartais nesėkme). Sudėtingumo teorijoje naudojama ne viena formali uždavinio sprendimo sam-

prata. Apsiribosime determinuotomis Turingo mašinomis. Mums užteks neformalaus jų apibrėžimo[15].

Galime įsivaizduoti, kad Turingo mašiną sudaro begalinė į kvadratėlius padalyta juosta ir slankiojanti skaitymo-rašymo galvutė. Kiekviename kvadrate gali būti įrašytas vienas abėcėlės $\mathcal{B}$ simbolis. Galvutė gali nuskaityti simbolį iš kvadrato, ties kuriuo ji stovi. Mašina gali būti vienoje iš padėčių, nusakomų aibe $Q = \{q_0, \ldots, q_m\}$. Mašina gali atlikti kelis paprastus veiksmus, kuriuos vadinsime elementariomis operacijomis:

- pakeisti nuskaitytą simbolį kitu $\mathcal{B}^*$ simboliu;
- paslinkti galvutę per vieną kvadratą į kairę ar į dešinę;
- pakeisti mašinos padėtį iš q_i į q_j.

Kuris iš šių veiksmų bus atliktas i-ajame žingsnyje, vienareikšmiškai lemia nuskaitytas simbolis ir tuometinė padėtis q_i.

Darbo pradžioje į juostos kvadratus nuo 1 iki n užrašomas pradinis žodis $\mathbf{x} \in \mathcal{B}^*$, $|\mathbf{x}| = n$, čia $|\mathbf{x}|$ – žodžio $\mathbf{x}$ ilgis. Mašina pastatoma į pradinę padėtį q_0, o jos galvutė ties pirmuoju kvadratu. Toliau mašina atlieka elementarias operacijas ir baigia darbą, patekusi į padėtį q_m. Juostos turinys tuo metu ir bus mašinos darbo rezultatas. Jį žymėsime $f(\mathbf{x})$ ir sakysime, kad Turingo mašina skaičiuoja funkciją $f : \mathcal{B}^* \to \mathcal{B}^*$. Mašinai M reikšmei skaičiuoti reikalingą elementariųjų operacijų skaičių žymėsime $t_{\mathrm{M}}(\mathbf{x})$.

104 apibrėžimas. *Turingo mašinos* M *darbo laiku vadinsime funkciją* $T_{\mathrm{M}} : \mathbf{N} \to \mathbf{N}$, *kurios reikšmė kiekvienam natūraliajam* n *yra lygi*

$$T_{\mathrm{M}}(n) = \max\{t : \quad \textit{egzistuoja } \mathbf{x} \in \mathcal{B}^*,\ |\mathbf{x}| = n,\ t_{\mathrm{M}}(\mathbf{x}) = t\}.$$

Taigi uždavinio (funkcijos, atskiru atveju – išsprendžiamumo uždavinio) sprendimą matematiškai suvoksime kaip atitinkamą Turingo mašiną.

Tą pačią funkciją $f(\mathbf{x})$ gali skaičiuoti įvairios Turingo mašinos priklausomai nuo to, koks algoritmas yra naudojamas. Konstruoti Turingo mašiną ir skaičiuoti jos darbo laiką – keblus darbas. Sudėtingesniems algoritmams patogiau konstruoti Turingo mašiną, kuri yra paprastesnių Turingo mašinų sistema (panašiai kaip programa, sudaryta iš paprogramių). Lengviau skaičiuoti ne Turingo mašinos elementarias operacijas, o įprastinių matematinių veiksmų kiekį. Pavyzdžiui, dviejų n dvejetainiais skaitmenimis užrašomų skaičių sudėčiai atlikti reikia n operacijų su simboliais. Kiekvienam tokiam veiksmui atlikti Turingo mašinai prireikia tam tikru dydžiu c_1 aprėžto elementariųjų operacijų kiekio. Galime tarti, kad dar reikia papildomai atlikti

[15] Matematinio griežtumo norintis skaitytojas turėtų pavartyti R. Lassaigne'o ir M. de Rougemont'o knygą „Logika ir algoritmų sudėtingumas“, kurią į lietuvių kalbą išvertė S. Norgėla, žr. [47].

ne daugiau kaip c_2 „techninių“ operacijų, pavyzdžiui, skaičių pabaigos ženklui nuskaityti. Taigi skaičių sudėčiai atlikti prireiks

$$T_{\mathrm{M}}(n) \le c_1 n + c_2$$

operacijų.

Panagrinėkime daugiau pavyzdžių. Vertindami operacijų skaičių, naudosime žymenį $\mathrm{O}(g)$, kuris reiškia tam tikrą dydį, ne didesnį kaip $c \cdot g$, čia $c > 0$ yra tam tikra konstanta, kurios tiksli reikšmė dažniausiai nėra svarbi.

1 pavyzdys. *Sveikųjų skaičių daugyba ir dalyba.* Tarkime, kad skaičiai $\mathbf{x} = x_1 \dots x_n$, $\mathbf{y} = y_1 \dots y_n$ užrašyti dvejetainėje sistemoje. Paeiliui daugindami x_i iš $y_1, y_2, \dots, y_n$, atliksime n^2 bitų daugybos operacijų. Gautas sandaugas pastumdami ir sudėdami atliksime dar maždaug n^2 operacijų. Taigi reikalingas bitų operacijų skaičius bus $T(n) = \mathrm{O}(n^2)$.

Jeigu reikia skaičių $\mathbf{x} = x_1 \dots x_n$ padalyti su liekana iš skaičiaus $\mathbf{y} = y_1 \dots y_m$ $(m \le n)$, tai, pagalvoję, kaip atliekamas „dalybos kampeliu“ algoritmas, įsitikinsime, kad reikalingas operacijų su bitais skaičius neviršys

$$T(n) = \mathrm{O}\left((n - m + 1) \cdot m\right) = \mathrm{O}(n^2).$$

Tačiau tiek dalybai, tiek daugybai galima sukonstruoti ir greitesnius algoritmus.

Panagrinėkime daugybos veiksmą. Kaip ir anksčiau $T(n)$ žymėsime kiekį elementarių operacijų, kurių reikia dviem skaičiams, užrašomiems n ilgio bitų žodžiu, sudauginti. Tarkime, kad reikia sudauginti skaičius $\mathbf{x} = x_1 \dots x_n$, $\mathbf{y} = y_1 \dots y_n$, užrašytus dvejetainėje sistemoje, ir n yra lyginis. Jeigu n nėra lyginis, visada galime pridėti gale nulį. Skaičius $\mathbf{x}$ ir $\mathbf{y}$ perskelsime pusiau:

$$\mathbf{x} = a2^{n/2} + b, \quad \mathbf{y} = c2^{n/2} + d.$$

Dabar sandaugą $\mathbf{x} \cdot \mathbf{y}$ galėsime skaičiuoti taip:

$$\mathbf{x} \cdot \mathbf{y} = (ac)2^n + (ac + bd - (a - b)(c - d))2^{n/2} + bd.$$

Taigi reikia apskaičiuoti tris sandaugas $(ac, bd, (a-b)(c-d))$, kuriose daugikliai yra ne ilgesni kaip $n/2$, bitų ir atlikti kelias sudėties operacijas. Kadangi daugyba iš 2^n yra labai paprasta, tai gausime nelygybę

$$T(n) \le 3T\Big(\frac{n}{2}\Big) + \mathrm{O}(n).$$

Tačiau perpus trumpesnių skaičių daugybai taip pat galime taikyti tą patį metodą, taigi

$$\begin{aligned} T(n) &\le 3\Big(3T\Big(\frac{n}{4}\Big) + \mathrm{O}\Big(\frac{n}{2}\Big)\Big) + +\mathrm{O}(n), \\ T(n) &\le 4 \cdot 3^k + \mathrm{O}\Big(n + 3 \cdot \frac{n}{2} + \ldots + 3^{k-1}\frac{n}{2^{k-1}}\Big), \end{aligned}$$

čia k yra natūrinis skaičius, $n/2^k \le 2$; galime imti $k = [\log_2 n]$. Kiek paskaičiavę gautume

$$3^k \le 3^{\log_2 n} = n^{\log_2 3}, \quad n + 3 \cdot \frac{n}{2} + \ldots + 3^{k-1}\frac{n}{2^{k-1}} \le \frac{2}{3} \cdot \left(\frac{3}{2}\right)^k < n^{\log_2 3}.$$

Taigi

$$T(n) = \mathrm{O}\big(n^{\log_2 3}\big).$$

Šis daugybos metodas yra kiek greitesnis už tiesioginį. Tačiau yra ir dar greitesnių. Palyginimui pasakysime, kad geriausiam iki šiol žinomam Schˋonhage-Strasseno algoritmui[16]

$$T(n) = \mathrm{O}(n \ln n \ln\ln n). \tag{88}$$

Natūraliojo skaičiaus dalybai iš kito skaičiaus irgi yra algoritmas, kurio veiksmų skaičiui teisingas (88) įvertis.

2 pavyzdys. *Patikrinti, ar natūralusis skaičius N yra pirminis.* Paprasčiausias (bet ne greičiausias!) būdas toks: patikrinti, ar N dalosi iš 2 ir visų nelyginių k, $3 \le k \le \sqrt{N}$. Tam blogiausiu atveju reikės apie $\sqrt{N}/2$ dalybos operacijų. Kadangi N dvejetainėje išraiškoje bus $n = [\log_2 N] + 1$ dvejetainių skaitmenų, tai reikalingą operacijų skaičių $T(n)$ galime vertinti taip:

$$T(n) = \mathrm{O}(\sqrt{N}) = \mathrm{O}\big(\mathrm{e}^{an}\big),$$

čia a – teigiama konstanta. Įvertis rodo, kad ilginant įeities duomenis skaičius operacijų, reikalingų „blogiausio“ atvejo sprendimui, didėja labai greitai. Tiesa, yra ir geresnių būdų šiam uždaviniui spręsti, žr. [3].

18.2. P ir NP uždavinių klasės

Vienus uždavinius sprendžiame nuosekliai atlikdami veiksmus iš pateikto sąrašo. Kitiems geriau taikyti tokį būdą: perskaičius sąlygą, nueiti miegoti arba žvejoti. Gal gera idėja kils netyčia, o tada jau uždaviniui išspręsti reikės visai mažų pastangų. Pirmuoju būdu sprendžiami **P** *klasės uždaviniai, o antruoju –* **NP**.

Ankstesniame skyrelyje išdėstytu požiūriu uždavinio sprendimas yra (determinuota) Turingo mašina. Ji visada realizuoja tam tikrą instrukcijų rinkinį, t. y. algoritmą. Dažnai vietoj Turingo mašinų kalbėsime tiesiog apie algoritmus.

Algoritmus klasifikuosime pagal jų Turingo mašinų darbo laiką.

105 apibrėžimas. *Algoritmas, kurio Turingo mašinos darbo laikas yra $T_{\mathrm{M}}(n)$, vadinamas polinominio laiko algoritmu, jei egzistuoja daugianaris*

[16] A. Schˋonhage, V. Strassen. Schnelle Multiplikation grosser Zahlen. Computing, 7(1971), p. 281–292.

p, kiekvienam $n \in \mathbf{N}$ tenkinantis nelygybę $T_{\mathrm{M}}(n) \leq p(n)$. Sakysime, kad uždavinys priklauso polinominio laiko uždavinių klasei **P**, *jei jos sprendimui žinomas polinominio laiko algoritmas*[17].

Paprastai laikomasi nuomonės, kad polinominio laiko algoritmas yra greitas ir jį galima praktiškai realizuoti. Todėl uždavinio priklausymas klasei **P** yra svarbi jo savybė. Tačiau praktiniam taikymui polinominis algoritmas gali būti net blogesnis už nepolinominį. Juk darbo laiko rėžius nustatančio daugianario laipsnis gali būti didelis!

Galima įsitikinti, kad pagrindinių aritmetinių bei palyginimo operacijų algoritmai priklauso klasei **P**. Todėl norint įrodyti, kad uždavinys priklauso klasei **P**, pakanka rasti jo sprendimo algoritmą, kuris turėtų polinominį laiką, išreikštą šių pagrindinių operacijų t.y sudėties, daugybos, palyginimo ir t. t. – skaičiumi. Kitaip sakant, nėra būtina kiekvieną kartą konstruoti Turingo mašiną. Prisiminkime ankstesnius pavyzdžius. Dviejų skaičių sandaugos, dalmens, liekanos ir bendrojo didžiausiojo daliklio radimas yra polinominio laiko uždaviniai. Tiktai visai neseniai buvo sugalvotas polinominio laiko algoritmas, nustatantis, ar duotasis skaičius yra pirminis, žr. [3].

O dabar panagrinėsime uždavinius, kuriems spręsti iki šiol nėra surasta polinominio laiko algoritmų. Tačiau tai būtų per daug platus ir neapibrėžtas tyrimo objektas. Todėl kalbėsime tik apie vadinamuosius nedeterminuoto polinominio laiko (NP) uždavinius. Apsiribosime išsprendžiamumo uždaviniais, nes į juos formaliai gali būti pertvarkyti kiti uždaviniai. Pradėsime vėl nuo pavyzdžių.

1 pavyzdys. *Patikrinti, ar natūralusis skaičius N yra sudėtinis.*

Jeigu skaitėme minėtą straipsnį apie polinominį algoritmą, tikrinantį, ar duotasis skaičius yra pirminis, tai ir šį uždavinį, žinoma, priskirsime **P** klasei. Juk klausimą, ar skaičius yra sudėtinis, galime pakeisti klausimu, ar jis pirminis.

Tarkime, kad to dar nesužinojome, todėl uždavinio nepriskiriame šiai klasei.

Taigi mums duotas didelis natūralusis skaičius N ir reikia pateikti atsakymą, ar jis sudėtinis. Tarkime, kad mes kažkaip sužinojome (galbūt susapnavome) problemos sprendimo raktą: kokį nors skaičiaus N daliklį d, $1 < d < N$. Aišku, kad šiuo atveju atsakymui gauti pakanka vienos dalybos operacijos, taigi jį išsprendžiame polinominio laiko algoritmu. Tačiau jeigu mums duotas skaičius nebūtų sudėtinis, tai gauti atsakymą per polinominį laiką nepadėtų jokie sapnai.

[17] Man patinka vietoj lotyniško žodžio „polinomas“ vartoti lietuvišką „daugianaris“. Jis skambus ir gerai perteikia prasmę. Tačiau termino „daugianarinis“ laikas pavartoti vis dėlto nesiryžau. Galima, žinoma, vartoti ilgą ir todėl nepatogų terminą, pavyzdžiui, „daugianariu apribotas laikas“. Tačiau tegu šiame skyrelyje bus „polinominis laikas“. Skaitytojams, kurie studijuos sudėtingumo teoriją iš kitomis kalbomis parašytų knygų bus lengviau atpažinti žinomas sąvokas (polynomial problem, algorithm angl.

Šis išsprendžiamumo uždavinys turi tokią savybę: jeigu į pateiktus duomenis atsakymas yra teigiamas, tai jį galima nustatyti per polinominį laiką, sužinojus tam tikrą papildomą informaciją (raktą), tačiau jeigu atsakymas yra neigiamas, tai surasti jį per polinominį laiką gali ir nepavykti. Tokius uždavinius vadinsime **NP** (nedeterminuoto polinominio laiko) uždaviniais.

Gautojo šifro dešifravimas paprastai būna **NP** klasės uždavinys: jei raktą žinome, galime jį išspręsti lengvai, jeigu ne – kartais sprendimas iš viso neįmanomas.

Pastebėsime, kad kiekvienas **P** klasės uždavinys yra ir **NP** klasės uždavinys.

2 pavyzdys. *Patikrinti, ar grafas yra Hamiltono, t. y. ar egzistuoja ciklas, jungiantis visas viršūnes ir einantis per kiekvieną iš jų tik po vieną kartą.*

Bendruoju atveju tai yra sudėtingas uždavinys, kuriam iki šiol nėra surastas polinominio laiko algoritmas. Akivaizdu, kad tai yra **NP** uždavinys. Jo raktas – Hamiltono ciklas.

Tačiau jeigu šį uždavinį suformuluotume šitaip: **ID:** *grafas* G;

U: *ar* G *nėra Hamiltono grafas?*

tai uždavinys jau nebepriklausytų klasei **NP**!

O dabar – kiek painokas matematinis **NP** uždavinio apibrėžimas. Primename, kad nagrinėjame tik išsprendžiamumo uždavinius, taigi klausimus galime interpretuoti taip: ar žodis $\mathbf{x} \in \mathcal{B}^*$ (įeities duomenys) priklauso uždavinio kalbai $\pi \subset \mathcal{B}^*$ (aibei duomenų, į kuriuos atsakymas yra teigiamas).

106 apibrėžimas. *Sakysime, kad* π *yra* **NP** *uždavinys (žymėsime* $\pi \in \mathbf{NP}$*), jei egzistuoja funkcija* $f : \mathcal{B}^* \times \mathcal{B}^* \to \{0, 1\}$ *tokia, kad*

1. $\mathbf{x} \in \pi$ *tada ir tik tada, kai egzistuoja toks* $\mathbf{y} \in \mathcal{B}^*$, *kad* $f(\mathbf{x}, \mathbf{y}) = 1$;
2. $f(\mathbf{x}, \mathbf{y})$ *skaičiuojama* $|\mathbf{x}|$ *atžvilgiu polinominio laiko algoritmu.*

Šiame apibrėžime žodis $\mathbf{y} \in \mathcal{B}^*$ ir yra uždavinio sprendimo raktas, kurį reikia kaip nors įspėti (kartais jis vadinamas sertifikatu).

Dabar įsitikinsime, kad **NP** uždavinių klasė turi savotišką struktūrą. Bet prieš tai dar viena nauja sąvoka.

107 apibrėžimas. *Sakoma, kad uždavinys* π_1 *yra polinomiškai pertvarkomas į uždavinį* π_2, *jei egzistuoja polinominio laiko funkcija* $\varphi : \mathcal{B}^* \to \mathcal{B}^*$, *tenkinanti sąlygą* $\mathbf{x} \in \pi_1$ *tada ir tik tada, kai* $\varphi(\mathbf{x}) \in \pi_2$.

Kitaip sakant, polinomiškai pertvarkyti uždavinį į kitą reiškia per polinominį laiką performuluoti klausimą, kad teigiamas atsakymas į jį reikštų taip pat teigiamą atsakymą į pirmąjį klausimą, o neigiamas – neigiamą.

Pavyzdžiui, visi žinome, kaip uždavinį

ID: *sveikieji skaičiai* a, b, c;

U: *ar lygtis* $ax^2 + bx + c = 0$ *turi du skirtingus sprendinius?*

galima pertvarkyti į uždavinį

ID: *sveikas skaičius* D;
U: *ar* D *teigiamas?*

124 teorema. *Jei* π_1 *yra polinomiškai pertvarkomas į* π_2 *ir* $\pi_2 \in \mathbf{P}$, *tai ir* $\pi_1 \in \mathbf{P}$.

Įrodymas. Tarkime, kad $\mathbf{x} \in \mathcal{B}^*$, $|\mathbf{x}| = n$, φ – polinominio laiko funkcija, pertvarkanti π_1 į π_2. Tada $|\varphi(\mathbf{x})| \leq g(n)$, čia g – daugianaris. Į klausimą, ar $\mathbf{x} \in \pi_1$, atsakysime patikrinę, ar $\varphi(\mathbf{x}) \in \pi_2$. Prisiminę, kad $\pi_2 \in \mathbf{P}$, galime teigti, kad šį tikrinimą galima atlikti algoritmu, kurio laikas neviršija $p(|\varphi(\mathbf{x})|) \leq q(n)$, čia p, q – daugianariai. Taigi $\pi_1 \in \mathbf{P}$.

Dabar jau galime suformuluoti vieną įžymią teoremą apie santykius **NP** uždavinių aibėje.

125 teorema. *(S. A. Cook, 1973) Egzistuoja* **NP** *uždavinys, į kurį polinomiškai galima pertvarkyti bet kurį kitą* **NP** *uždavinį.*

Toks universalus uždavinys vadinamas *pilnuoju* **NP** uždaviniu. Tiesą sakant, Cookas suformulavo šią teoremą kiek kitaip. Jis pateikė ir **NP** pilno uždavinio pavyzdį. Dabar tokių pavyzdžių žinoma labai daug. 1979 m. Garey ir Johnsonas pateikė katalogą [30], kuriame yra virš 3000 pilnųjų NP problemų iš skaičių teorijos, logikos, kombinatorikos ir automatų teorijos. Tarp jų yra ir mūsų jau nagrinėta grafo Hamiltono ciklo egzistavimo uždavinys.

Polinominių uždavinių aibė sudaro **NP** uždavinių aibės poaibį. Tiesą sakant, niekas pasaulyje nežino, ar **NP** uždavinių yra iš tikrųjų daugiau negu **P** klasės uždavinių! Juk, pavyzdžiui, niekas negali paneigti, kad kas nors nesuras polinominio laiko algoritmo Hamiltono ciklo radimo grafe uždaviniui spręsti. Jeigu tai įvyktų, būtų nustatyta, kad šis uždavinys priklauso klasei **P**, o tuo pačiu metu... kad visi **NP** uždaviniai sprendžiami polinominio laiko algoritmais, t. y. kad $\mathbf{P} = \mathbf{NP}$!

Prieš keletą metų Clay matematikos institutas Kembridže (JAV, Masachusetts), kurį finansuoja verslininkas Landonas T. Clay, paskelbė septynių matematinių uždavinių sąrašą. Atlygis už kiekvieno iš jų sprendimą – milijonas JAV dolerių. Vienas iš šių uždavinių formuluojamas itin paprastai:

įrodyti arba paneigti, kad $\mathbf{P} = \mathbf{NP}$.

Taigi milijoną dolerių galima uždirbti tyrinėjant paprasčiausius grafus.

18.3. Nepilnas NP pilnų problemų sąrašėlis

> **NP** *pilnų uždavinių žinoma labai daug. Ir iš nedidelio skyrelyje pateikiamo sąrašo susidarysite įspūdį apie tokių uždavinių įvairovę.*

Minėjome, kad **NP** pilnų uždavinių dabar jau žinoma keli tūkstančiai. Jų galima rasti visose diskrečiosios matematikos srityse. Tuo įsitikinsite peržvelgę ir šį nedidelį **NP** uždavinių rinkinį.

Visur $G = \langle V, E \rangle$ reiškia grafą, V – jo viršūnių aibė, E – briaunų aibė.

Viršūnių denginys (vertex cover)

ID: $G = \langle V, E\rangle$, $k \leq |V|$;

U: *ar egzistuoja* $V' \subset V, |V'| \leq k$, *kad kiekvienos briaunos viršūnė priklausytų* V'*?*

Grafo k-spalvinimas (graph k-colorability)

ID: $G = \langle V, E\rangle$, $k \leq |V|$;

U: *ar galima viršūnes nuspalvinti* k *spalvomis, kad bet kurios briaunos dvi viršūnės būtų nuspalvintos skirtingai?*

Vienspalvis trikampis (monochromatic triangle)

ID: $G = \langle V, E\rangle$;

U: *ar galima nuspalvinti visas grafo briaunas dviem spalvomis, kad iš tos pačios spalvos briaunų ir atitinkamų viršūnių negalėtume sudaryti vieno trikampio?*

Komercinis keliautojas (traveling salesman)

ID: *miestų aibė* C, *atstumai tarp miestų* $d(c_i, c_j)$, *skaičius* $B > 0$;

U: *ar egzistuoja būdas apeiti visus miestus, kad viso kelio ilgis neviršytų* B*?*

Aibių pakavimas (set packing)

ID: *baigtinės aibės poaibių rinkinys* C, *skaičius* $k, 0 < k \leq |C|$;

U: *ar rinkinyje* C *egzistuoja* k *poromis nesikertančių aibių?*

Aibių skaidymas (set splitting)

ID: *baigtinės aibės* S *poaibių rinkinys* C;

U: *ar galima* S *išskaidyti į dviejų nesikertančių aibių sąjungą* $S = S_1 \cup S_2$, *kad jokia* C *aibė nebūtų nei* S_1, *nei* S_2 *poaibis?*

Trumpiausias viršžodis (shortest common supersequence)

ID: *baigtinė abėcėlė* $\Sigma, R \subset \Sigma^*$ *ir natūralusis skaičius* k;

U: *ar egzistuoja žodis* $w \in \Sigma^*$, *kad* $|w| \leq k$ *ir kiekvienas žodis* $x \in R$ *būtų* w *fragmentas (dalis)?*

Skaidymas (partition)

ID: *baigtinė aibė* A *ir neneigiami sveikieji skaičiai (elementų svoriai)* $s(a), a \in A$;

U: *ar egzistuoja* $A' \subset A$, *kad*

$$\sum_{a \in A'} s(a) = \sum_{a \in A \setminus A'} s(a)?$$

Kvadratiniai lyginiai (quadratic congruences)

ID: *natūralieji skaičiai* a, b, c;

U: *ar egzistuoja natūralusis skaičius* $x, |x| < c$, *kad* $x^2 \equiv a \pmod{b}$?

Dalumo palyginimas (comparative divisibility)

ID: *du natūraliųjų skaičių rinkiniai* $\{a_1, \ldots, a_n\}$ *ir* $\{b_1, \ldots, b_m\}$;

U: *ar yra toks natūralusis skaičius* c, *kad skaičių iš pirmojo rinkinio, kuriuos dalo* c, *yra daugiau nei analogiškų skaičių iš antrojo rinkinio?*

Kvadratinės Diofanto lygtys (quadratic diophantine equations)

ID: *natūralieji skaičiai* a, b, c;

U: *ar egzistuoja natūralieji skaičiai* x, y *su sąlyga* $ax^2 + by = c$*?*

Kryžiažodžio konstravimas (crossword puzzle construction)

ID: *baigtinė abėcėlė* Σ*, baigtinė žodžių aibė* $W \subset \Sigma^*$*,* $n \times n$ *matrica* A*, kurios elementai yra vienetai;*

U: *ar galima iš* W *žodžių sukonstruoti kryžiažodį pagal* A *kaip šabloną (rašant raides į langelius, kurie pažymėti* 0*, o langelius, pažymėtus* 1*, laikant užtušuotais)?*

18.4. Tikimybiniai algoritmai

Deterministinio algoritmo taikytojas apie sunkų uždavinį galvoja taip: kadangi uždavinys sunkus, tai, norint gauti atsakymą, reiks daug dirbti. Tikimybinio algoritmo taikytojo požiūris kitoks: nors uždavinys sunkus, bet jeigu pasiseks, išspręsiu jį lengvai.

Tikimybinių algoritmų idėja labai paprasta, juos mes gyvenime neretai naudojame. Tarkime, reikia atsakyti į klausimą, ar ežere yra žuvų. Algoritmas paprastas – užmerkti meškerę ir palaukti. Galimos dvi baigtys: pagausime žuvį arba nepagausime. Pirmuoju atveju atsakymas į klausimą yra teigiamas ir visiškai teisingas, antruoju – atsakymas neigiamas, tačiau jo teisingumas yra tik tikėtinas, o ne absoliutus.

Sudarysime panašias savybes turintį algoritmą tokiam uždavinius spręsti:

ID: N *– nelyginis skaičius;*

U: *ar* N *sudėtinis?*

Pasinaudosime Fermat teorema (šį labai svarbų visai viešojo rakto kriptografijai teiginį įrodysime kitame skyriuje).

126 teorema. *Jei* N *– nelyginis pirminis skaičius,* $(a, N) = 1$*, tai* $a^{N-1} \equiv 1 \ (mod\ N)$.

Dabar galime tikrinti, ar N yra pirminis, taip:

- parinkime nelyginį $w, 1 < w < N$; jei $w|N$, tai N – sudėtinis skaičius;
- jei $(w, N) = 1$, skaičiuokime $w^{N-1} (\text{mod } N)$. Jei $w^{N-1} \not\equiv 1 (\text{mod } N)$, tai N sudėtinis, jei $w^{N-1} \equiv 1 (\text{mod } N)$, darome išvadą, kad N yra pirminis.

Aišku, kad teigdami, jog N pirminis, ne visada būsime teisūs. Galima atveju $w^{N-1} \equiv 1 \ (\text{mod } N)$ kartoti testą, parinkus kitą w reikšmę. Dažnai šitokiu būdu pavyksta „demaskuoti“ pirminiu apsimetusį sudėtinį skaičių. Deja, ne visada. Egzistuoja be galo daug skaičių[18] N, kurie su visais $(w, N) = 1$ tenkina sąlygą $w^{N-1} \equiv 1 \ (\text{mod } N)$. Mažiausias Carmichaelio skaičius yra 561. Tačiau visiems kitiems sudėtiniams skaičiams nepavyks išlikti neatpažintiems, jeigu tik testas bus pakankamai daug kartų pakartotas. Tai išplaukia iš tokios teoremos:

[18] Jie vadinami Carmichaelio skaičiais; R. D. Carmichael (1879–1967).

127 teorema. *Tegu N – nelyginis skaičius. Tada sąlyga $w^{N-1} \equiv 1 \ (mod\ N)$ teisinga arba visiems $w, (w,N) = 1, 1 \leq w < N$, arba daugiausiai pusei tokių w.*

Dabar apibrėžkime tikimybinio algoritmo sąvoką.

108 apibrėžimas. *Algoritmą išsprendžiamumo uždaviniui spręsti vadinsime tikimybiniu, jeigu įeities duomenims jo pateikiamas atsakymas $TAIP$ yra visada teisingas, o NE – teisingas su tikimybe, didesne už $1/2$.*

Teiginys *teisingas su tikimybe, didesne už* $1/2$, reiškia, kad, su įeities duomenimis, kuriems teisingas atsakymas yra NE, atlikus algoritmą visais įmanomais būdais, šis atsakymas bus gautas daugiau kaip pusę kartų. Taigi dėl Carmichaelio skaičių negalime mūsų algoritmo, pagrįsto Fermat teorema, pavadinti tikimybiniu. Juk jeigu pasitaikė taip, kad mūsų įeities duomuo yra Carmichaelio skaičius, tai tikimybė, kad algoritmas duos teisingą atsakymą, lygi nuliui!

Jeigu tikimybinio algoritmo naudojamo laiko kiekis yra polinominis įeities duomenų dydžio atžvilgiu, tai algoritmą vadinsime tikimybiniu polinominiu algoritmu, o jo sprendžiamą uždavinį – tikimybiniu polinominiu uždaviniu. Tikimybinių polinominių uždavinių klasė literatūroje žymima **RP** (*random polynomial* angl.).

Dabar sukonstruosime tikrą tikimybinį polinominį algoritmą sudėtinių skaičių atpažinimo uždaviniui spręsti.

Tegu N – nelyginis skaičius. Suraskime skaičius $s, t, (t,2) = 1$, kad būtų $N - 1 = 2^s t$. Parinkime $a, (a,N) = 1$ ir patikrinkime, ar $a^t \not\equiv 1 \pmod N$. Jei galioja lygybė, parinkime kitą a. Suskaičiuokime

$$a^t \pmod N,\ a^{2t} \pmod N,\ \ldots \quad a^{2^{s-1}t} \pmod N,\ a^{N-1} \pmod N. \quad (89)$$

Tarkime, N yra pirminis skaičius. Kiek vienetų yra sekoje (89)? Iš Fermat teoremos gauname, kad paskutinis šios sekos skaičius yra būtinai vienetas. Vienetas gali pasitaikyti ir anksčiau (pirmas skaičius tikrai ne vienetas). Tegu pirmasis sekos vienetas yra

$$a^{2^l t} \equiv 1 \pmod N,$$

tada prieš jį einantis skaičius $b \equiv a^{2^{l-1}t} \pmod N$ yra lyginio $x^2 \equiv 1 \pmod N$ sprendinys. Kadangi N yra pirminis, tai šis lyginys turi tik du sprendinius: 1 ir $N - 1$. Iš sąlygos $b \not\equiv 1 \pmod N$, gauname, kad $b \equiv N - 1 \pmod N$.

Taigi pirminiam skaičiui N teisingi tokie teiginiai:

1. paskutinis (89) sekos skaičius yra 1;
2. prieš pirmąjį (89) sekos vienetą stovi $N - 1$.

Dabar algoritmą galime aprašyti taip:

- surasti skaičius $s, t, (t, 2) = 1$, kad $N - 1 = 2^s t$;
- parinkti $1 < a < N$; jei $a|N$, tai N – sudėtinis;
- jei $a^t \equiv 1 \pmod{N}$, parinkti kitą a;
- sudaryti seką (89);
- jei bent vienas iš teiginių 1), 2) (89) sekai neteisingas, tai N – sudėtinis skaičius; jei abu teisingi – pirminis.

Šis algoritmas vadinamas Millerio (arba Millerio-Rabino) testu. Algoritmas polinominis. Ar jį galime pavadinti tikimybiniu, t. y. ar jis tenkina mūsų apibrėžimo sąlygas? Teigiamas atsakymas, žinoma, visada yra teisingas. Reikia įsitikinti, kad neigiamas atsakymas teisingas su tikimybe, didesne už 1/2. Tai išplaukia iš tokio teiginio:

128 teorema. *Tegu N – nelyginis sudėtinis skaičius, $4 < N$. Tada ne mažiau kaip $3(N-1)/4$ skaičių $a, 1 < a < N$, arba teiginys 1), arba 2) neteisingas.*

Taigi Millerio-Rabino testas – tikimybinis polinominis algoritmas. Tikimybė, kad sudėtinį N pripažinsime pagal šį testą pirminiu, yra ne didesnė už 1/4. Pakartojus testą k kartų, klaidingo sprendimo tikimybė bus ne didesnė už $1/4^k$. Matome, kad, pakartoję testą daug kartų, ją galima kiek norime sumažinti.

19 Skaičių teorijos sąvokos ir algoritmai

Viešojo rakto kriptosistemų, skaitmeninių parašų ir kitų moderniosios kriptografijos schemų pagrindas – veiksmai baigtinėse algebrinėse struktūrose. Tos struktūros – dalybos liekanų žiedai $\mathbb{Z}_n$, baigtiniai kūnai $\mathbb{F}_p$, $\mathbb{F}_{p^m}$... Jas tyrinėjo pačios „gryniausios“ matematikos atstovai, niekada turbūt nemanę, kad jų teiginiai bus pritaikyti labai praktiškiems tikslams.

Uždavinius sprendžiame naudodamiesi tam tikromis taisyklėmis – algoritmais. Jie nurodo, kokius veiksmus reikia atlikti su duomenimis, kad gautume rezultatą. Veiksmų skaičius priklauso nuo uždavinio duomenų dydžio. Kuo matuoti duomenų dydį? Natūrali idėja – dvejetainės abėcėlės simbolių skaičiumi, reikalingu duomenims užrašyti. Kai duomenų dydis auga (pavyzdžiui, skaičiuojame su vis didesniais skaičiais) veiksmų skaičius irgi didėja. Jeigu veiksmų skaičius, reikalingas algoritmui su n ilgio duomenimis atlikti neviršija tam tikro daugianario f reikšmės $f(n)$, toks algoritmas vadinamas polinominiu. Paprastai laikoma, kad uždavinys, kuriam spręsti žinome polinominį algoritmą, yra sprendžiamas greitai. Tačiau tai tik paviršutiniška, nors dažnai ir teisinga nuomonė. Algoritmų vertinimo pagal sudėtingumą uždaviniai nagrinėjami sudėtingumo teorijoje. Norintys į ją įsigilinti turėtų pavartyti, pavyzdžiui, [30], [47] knygas. Mes tik retkarčiais pabandysime

įvertinti algoritmui atlikti reikalingų veiksmų skaičių. Tokiuose įverčiuose patogu naudoti žymenį $f = \mathrm{O}(g)$, kuris reiškia, kad dydis f neviršija dydžio $c \cdot g$, čia $c > 0$ yra konstanta, kurios tiksli reikšmė paprastai nėra svarbi.

19.1. Euklido algoritmas

Bendrąjį didžiausiąjį dviejų skaičių daliklį Antikos graikai suvokė kaip bendrąjį skaičių matą. Euklido algoritmas jam rasti – nedaug matematinių kūrinių gali jam prilygti paprastumu, teorine ir praktine reikšme.

Kas yra bendrasis didžiausiasis dviejų natūrinių skaičių daliklis, visi žinome. Nuo jo ir pradėkime.

109 apibrėžimas. *Bendruuoju didžiausiuoju natūrinių skaičių a, b dalikliu vadinamas didžiausias skaičius, kuris dalija ir a, ir b. Bendrąjį didžiausiąjį daliklį žymėsime (a, b). Jeigu $(a, b) = 1$, šiuos skaičius vadinsime tarpusavyje pirminiais.*

Jeigu b dalijasi iš a, tai, žinoma, $(a, b) = b$.

Darni natūrinių skaičių dalumo teorija išdėstyta Euklido „Pradmenyse“. Visi skaičių teorijos vadovėliai, kuriuose teiginiai dėstomi nuosekliai ir išsamiai, ją pakartoja. Bet mums juk pirmiausia rūpi ne loginiai teorijos pagrindai, bet jos teiginiai, kuriuos kaip įrankius galima panaudoti kriptografijoje. Taigi praleiskime įrodymus tų teiginių, kurie tiesiog „akivaizdūs“. Pavyzdžiui,

jei $(a, b) = d$, tai $a = da', b = db'$ ir $(a', b') = 1$.

Skaičių dalybą su liekana jau aptarėme kodavimo teorijai skirtoje knygos dalyje. Jeigu $a > b$ yra natūralieji skaičiai, tai

$$a = qb + r, \quad 0 \le r < b.$$

Euklido algoritmas bendrajam didžiausiajam skaičių a, b dalikliui rasti remiasi teiginiu: $(a, b) = (b, r)$.

Jei $a > b$ – natūralieji skaičiai, tai, kartodami dalybos su liekana veiksmus, gausime:

$$\begin{aligned}
a &= q_0 b + r_0, \quad 0 < r_0 < b, \\
b &= q_1 r_0 + r_1, \quad 0 < r_1 < r_0, \\
r_0 &= q_2 r_1 + r_2, \quad 0 < r_2 < r_1, \\
&\cdots\cdots\cdots\cdots \\
r_{m-2} &= q_m r_{m-1} + r_m, \quad 0 < r_m < r_{m-1}, \\
r_{m-1} &= q_{m+1} r_m + r_{m+1}, \quad 0 < r_{m+1} < r_m, \\
&\cdots\cdots\cdots\cdots \\
r_{n-2} &= q_n r_{n-1} + r_n, \quad 0 < r_n < r_{n-1}, \\
r_{n-1} &= q_{n+1} r_n.
\end{aligned}$$

Tada paskutinė nelygi nuliui liekana ir yra bendrasis didžiausiasis daliklis:

$$(a,b) = (b,r_0) = (r_0,r_1) = \ldots = (r_{n-1},r_n) = r_n.$$

Štai ir visa Euklido algoritmo esmė.

Iš priešpaskutinės lygybės gausime:

$$(a,b) = r_n = r_{n-2} + (-q_n)r_{n-1}.$$

Įstatę į šią lygybę r_{n-1} išraišką per r_{n-2}, r_{n-3}, gausime bendrojo didžiausiojo daliklio išraišką per liekanas su mažesniais indeksais. Galų gale, šitaip kopdami į viršų, surasime sveikuosius x, y, kad

$$(a,b) = xa + yb.$$

Pastebėkime, kad jei $(a,b) = ur_m + vr_{m+1}$, tai $(a,b) = vr_{m-1} + (u - vq_{m+1})r_m$.

Pasinaudoję šia pastaba, galime ieškoti skaičių x, y taip. Surašykime Euklido algoritmo skaičiavimus tokioje lentelėje:

r_{i-1}, r_i	q_{i+1}	u_{i-1}, u_i
a, b	q_0	
b, r_0	q_1	
r_0, r_1	q_2	
$\ldots$	$\ldots$	$\ldots$
r_{m-1}, r_m	q_{m+1}	$v, u - vq_{m+1}$
r_m, r_{m+1}	q_{m+2}	u, v
$\ldots$	$\ldots$	$\ldots$
r_{n-2}, r_{n-1}	q_n	$1, -q_n$
r_{n-1}, r_n	q_{n+1}	

Trečiąjį stulpelį pildome nuo apačios į viršų. Jeigu paskutinė užpildyta eilutė yra tokia: $r_m, r_{m+1} \mid q_{m+2} \mid u; v$, tai į viršutinės eilutės trečiąjį stulpelį rašome skaičius $v; u - vq_{m+1}$. Su kiekvienos eilutės skaičiais $r_{i-1}, r_i, u_{i-1}, v_{i-1}$ teisinga lygybė

$$(a,b) = u_{i-1}r_{i-1} + v_{i-1}r_i.$$

Pirmosios eilutės skaičiai duoda lygybę

$$ax + by = (a,b).$$

Lentelėje pateiktas skaičiavimo pavyzdys su $a = 57, b = 10$.

$57; 10$	5	$3; -17;$	$1 = 57 \cdot 3 + 10 \cdot (-17)$
$10; 7$	1	$-2; 3;$	$1 = 10 \cdot (-2) + 7 \cdot 3$
$7; 3$	2	$1; -2;$	$1 = 7 \cdot 1 + 3 \cdot (-2)$
$3; 1$	3		

Panagrinėkime, kiek daugiausia Euklido algoritmo žingsnių reikia atlikti, kad surastume bendrąjį didžiausiąjį daliklį. Apibrėžkime skaičius

$$f_0 = 1,\ f_1 = 2,\ f_i = f_{i-1} + f_{i-2}\ (i \geq 2).$$

Kas nors, be abejonės, pastebėjo, kad tai – Fibonačio skaičiai. Įsitikinsime, kad mūsų dalybos liekanoms teisingos nelygybės:

$$r_n \geq f_0,\ r_{n-1} \geq f_1,\ \ldots,\ r_{n-m} \geq f_n,\ \ldots, r_0 \geq f_n,\ b \geq f_{n+1}.$$

Iš tikrųjų, nelygybės $r_n \geq f_0,\ r_{n-1} \geq f_1$ akivaizdžios. Jeigu teisingos nelygybės $r_{n-m+2} \geq f_{m-2}, r_{n-m+1} \geq f_{m-1}$, tai

$$r_{n-m} = q_{n-m+1} r_{n-m+1} + r_{n-m+2} \geq r_{n-m+1} + r_{n-m+2} \geq f_{m-1} + f_{m-2} = f_m.$$

Taigi teisinga nelygybė $b \geq f_{n+1}$. Tačiau savo ruožtu Fibonačio skaičiai tenkina nelygybes

$$f_m \geq \alpha^m, \quad \alpha = \frac{1+\sqrt{5}}{2}, \quad \alpha^2 = \alpha + 1.$$

Iš tikrųjų, nelygybės $f_0 \geq \alpha^0, f_1 \geq \alpha^1$ yra akivaizdžios. Jeigu teisingos nelygybės $f_{m-1} \geq \alpha^{m-1}, f_m \geq \alpha^m$, tai

$$f_{m+1} = f_m + f_{m-1} \geq \alpha^m + \alpha^{m-1} = \alpha^{m-1}(\alpha + 1) = \alpha^{m-1}\alpha^2 = \alpha^{m+1}.$$

Matematinės indukcijos metodu nelygybės Fibonačio skaičiams įrodytos. Tada

$$\alpha^{n+1} \leq f_{n+1} \leq b, \quad n \leq \log_\alpha b - 1.$$

Jeigu skaičiui a užrašyti reikia N dvejetainių skaitmenų, o vienam Euklido žingsniui atlikti reikalingų operacijų skaičių vertinsime dydžiu $\mathrm{O}(N^2)$, tai visam algoritmui reikalingų operacijų skaičių galime įvertinti dydžiu

$$\mathrm{O}(n \cdot N^2) = \mathrm{O}(\log_\alpha b \cdot N^2) = O(N^3).$$

Taigi bendrojo didžiausiojo daliklio radimo algoritmas yra polinominis; iš tikrųjų, vertinant operacijų skaičių tiksliau, galima gauti įvertį $O(N^2)$.

Skaičius α, kuris padėjo išspręsti uždavinį, yra labai įžymus: tai aukso pjūvio skaičius.

19.2. Apie žiedus $\mathbb{Z}_n$

Šio skyrelio teiginių atradėjai sudaro ryškiausią matematikų žvaigždyną. Lyginius sukūrė Gausas, Fermat ir Euleris įrodė svarbias skaičių teorijai (ir kriptografijai) teoremas ir, žinoma, vėl – Euklidas.

Jeigu sveikųjų skaičių a ir b skirtumas dalijasi iš n, rašome

$$a \equiv b \pmod{n}.$$

Tokį sąryšį vadiname lyginiu, skaitome: a lygsta b moduliu n. Toks sąryšis teisingas tada ir tik tada, kai, dalijant a ir b iš n, gaunama ta pati liekana.

Baigtiniams kūnams skirtame skyrelyje suformuluotas teiginys apie paprasčiausias lyginių savybes:

jei $a \equiv b \pmod{n}$ ir $c \equiv d \pmod{n}$, tai $a + c \equiv b + d \pmod{n}$;
jei $a \equiv b \pmod{n}$ ir c yra sveikasis skaičius, tai $ca \equiv cb \pmod{n}$;
jeigu $ca \equiv cb \pmod{n}$ ir $(c, n) = 1$, tai $a \equiv b \pmod{n}$.

Naudodamiesi šiomis savybėmis, galime atlikti veiksmus su lyginiais, t.y. iš vienų lyginių gauti kitus. Kiekvieną lyginį moduliu n, kurio abi pusės sudarytos iš skaičių ar simbolių, naudojant sudėties ir daugybos veiksmus, galime interpretuoti kaip dalybos liekanų žiedo $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ elementų sąryšį. Pakanka skaičius pakeisti jų dalybos liekanomis, o veiksmus – liekanų žiedo veiksmais. Primename, kad šiame žiede apibrėžtos tokios sudėties ir daugybos operacijos:

$$\begin{aligned} a +_n b &= a + b \text{ dalybos iš } n \text{ liekana}, \\ a \times_n b &= a \cdot b \text{ dalybos iš } n \text{ liekana}. \end{aligned}$$

Taip pat ir reiškinius su šiais veiksmais galime skaičiuoti naudodamiesi lyginių žymeniu ir savybėmis. Pavyzdžiui, apskaičiuokime $5^3 \times_9 7^6$:

$$5^3 \cdot 7^6 \equiv 25 \cdot 5 \cdot (49)^3 \equiv 7 \cdot 5 \cdot 4^3 \equiv (-1) \cdot 1 \equiv 8 \pmod{9}.$$

Taigi $5^3 \times_9 7^6 = 8$.

Dažniausiai veiksmams liekanų žiede žymėti vartosime įprastinės skaičių sudėties ir daugybos ženklus.

Kodavimo teorijoje svarbiausias veiksmas buvo žiedo elementų sudėtis ir daugyba, kriptografijoje, kaip netrukus pamatysime, – kėlimas laipsniu, t.y. daugelio vienodų elementų daugyba.

Kėlimo laipsniu operaciją žiede $\mathbb{Z}_n$ galime atlikti labai sparčiu „kėlimo kvadratu“ algoritmu. Panagrinėkime skaitinį pavyzdį. Tarkime, žiede $\mathbb{Z}_{11}$

reikia apskaičiuoti 10^{31}, t.y. reikia rasti skaičiaus $100\ldots0$ su 31 nuliu dalybos iš 11 liekaną. Kas ims dalyti – pražus. Skaičiuokime kitaip.

Užrašę 31 dvejetainėje skaičiavimo sistemoje, gausime:

$$31 = 1+1\cdot2^1+1\cdot2^2+1\cdot2^3+1\cdot2^4, \quad 10^{31} = 10^1\cdot10^2\cdot(10^2)^2\cdot((10^2)^2)^2\cdot(((10^2)^2)^2)^2.$$

Dabar skaičiuojame

$$10^2 \equiv 7 \pmod{31}, \quad 7^2 \equiv 18 \pmod{31},$$
$$18^2 \equiv 14 \pmod{31}, \quad 14^2 \equiv 10 \pmod{31},$$

taigi

$$10^{31} \equiv 10\cdot7\cdot18\cdot14\cdot10 \equiv 25 \pmod{31}.$$

Jeigu $n = p$ yra pirminis skaičius, tai žiedas $\mathbb{Z}_p$ yra kūnas; jį paprastai žymėdavome $\mathbb{F}_p$. Žinome, kad kiekvienas nenulinis šio kūno elementas a turi atvirkštinį, t.y. egzistuoja $b \in \mathbb{F}_p$, paprastai žymimas a^{-1}, kad

$$a \times_n b = 1, \quad \text{t.y. } a \times_n a^{-1} = 1.$$

O kaipgi bendruoju atveju?

110 apibrėžimas. *Tegu $n \geq 1$ yra natūralusis skaičius,*

$$\mathbb{Z}_n^* = \{m : 1 \leq m < n, (m,n) = 1\}.$$

Funkciją $\varphi(n) = |\mathbb{Z}_n^*|$, apibrėžtą natūraliųjų skaičių aibėje, vadinsime Eulerio funkcija.

Eulerio funkcijos $\varphi(n)$ reikšmė lygi mažesnių už n ir tarpusavyje pirminių su juo skaičių kiekiui. Pavyzdžiui,

$$\varphi(1) = \varphi(2) = 1, \ \varphi(3) = \varphi(4) = 2, \ldots$$

Jeigu p yra pirminis, tai nesunku suvokti, kad

$$\varphi(p) = p-1, \quad \varphi(p^m) = p^m - p^{m-1}. \tag{90}$$

O dabar imkimės atvirkštinių klausimo.

129 teorema. *Kiekvienas $\mathbb{Z}_n^*$ elementas turi atvirkštinį.*
Jeigu $a \in \mathbb{Z}_n^*$, tai

$$a^{\varphi(n)} \equiv 1 \ (mod\ n). \tag{91}$$

Įrodymas. Tegu

$$\mathbb{Z}_n^* = \{a_0, a_1, \ldots, a_{\varphi(n)}\}, \quad a_0 = 1,$$

ir $a \in \mathbb{Z}_n^*$. Pirmiausia reikia įsitikinti, kad visi skaičiai

$$a\cdot a_0, a\cdot a_1, \ldots, a\cdot a_{\varphi(n)} \tag{92}$$

atitinka skirtingus $\mathbb{Z}_n^*$ elementus, t.y. duoda skirtingas dalybos iš n liekanas. Padarę prielaidą, kad taip nėra, tuoj pat gautume prieštaravimą. Tačiau jeigu visi (92) atitinka skirtingus $\mathbb{Z}_n^*$ elementus, tai vienas atitinka $a_0 = 1$. Todėl atsiras toks $b \in \mathbb{Z}_n^*$, kad $a \times_n b = 1$.

Taigi (92) yra tie patys elementai $a_0, a_1, \ldots, a_{\varphi(n)}$, tik kitaip persirikiavę. Sudauginę juos visus ir pasinaudoję lyginių savybe, gausime

$$a^{\varphi(n)} \cdot a_0 \cdot a_1 \cdots a_{\varphi(n)} \equiv a_0 \cdot a_1 \cdots a_{\varphi(n)} \pmod{n}, \quad a^{\varphi(n)} \equiv 1 \pmod{n}.$$

Dar keletas pastabų apie šią teoremą. Iš tikrųjų įrodėme kiek daugiau negu teigėme. Įrodėme, kad daugybos veiksmo atžvilgiu aibė $\mathbb{Z}_n^*$ yra grupė.

Teiginys (91) skaičių teorijoje vadinamas Eulerio teorema. Vienas jos atvejis nusipelno atskiro vardo.

130 teorema. *(Mažoji Fermat teorema.) Jeigu p yra pirminis skaičius ir $(a, p) = 1$, tai*

$$a^{p-1} \equiv 1 \ (mod\ p).$$

Kriptografijai nepakanka žinoti, kokie elementai turi atvirkštinius, reikia mokėti juos greitai surasti. Tačiau greitas metodas jau yra!

Tegu $a \in \mathbb{Z}_n^*$, reikia rasti jo atvirkštinį. Kadangi $(a, n) = 1$, tai, pasinaudoję Euklido algoritmu, galime rasti sveikuosius skaičius x, y, kad būtų

$$ax + ny = 1.$$

Tačiau iš šios lygybės išplaukia lyginys $ax \equiv 1 \pmod{n}$. Tada x dalybos iš n liekana yra a atvirkštinis.

Pavyzdžiui, su $n = 57, a = 10$ iš lygybės $1 = 57 \cdot 3 + 10 \cdot (-17)$ gauname

$$10 \cdot (-17) \equiv 1 \pmod{57}, \quad 10 \cdot 40 \equiv 1 \pmod{57}, \quad 10^{-1} \equiv 40 \pmod{57}.$$

Kitas būdas surasti atvirkštinį – pasinaudoti Eulerio teorema: jei $(a, n) = 1$, tai

$$a^{\varphi(n)} \equiv 1 \pmod{n}, \quad a \cdot a^{\varphi(n)-1} \equiv 1 \pmod{n},$$

taigi $a^{-1} \equiv a^{\varphi(n)-1} \pmod{n}$.

O dabar dar patyrinėkime Eulerio funkciją. Kiekvieną natūralųjį skaičių n galime užrašyti pirminių skaičių laipsnių sandauga:

$$n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_t^{\alpha_t}, \quad p_1 < p_2 < \ldots < p_t,$$

čia p_i yra pirminiai skaičiai. Šis teiginys vadinamas pagrindine aritmetikos teorema. Nors tokiu pavidalu jis suformuluotas gana neseniai, juo naudojosi ir Euklido laikų matematikai. Žinodami visus pirminius skaičiaus n daliklius, galime jais pasinaudoti ir užrašyti Eulerio funkcijos išraišką.

131 teorema. *Jeigu $p_1 < p_2 < \ldots < p_t$ yra visi pirminiai skaičiaus n dalikliai, tai*

$$\varphi(n) = n\left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \cdots \left(1 - \frac{1}{p_t}\right). \tag{93}$$

Įrodymas. Pažymėkime $A = \{m : 1 < m \leq n, (m, n) > 1\}$. Tada $\varphi(n) = n - |A|$. Savo ruožtu jei $A_i = \{m : 1 < m \leq n, (p_i, n) = p_i\}$, tai $A = \bigcup_{i=1}^{t} A_i$. Taigi reikia apskaičiuoti, kiek skaičių yra šioje aibių sąjungoje. Pasinaudokime kombinatorine „pliuso-minuso“ taisykle:

$$\left|\bigcup_{i=1}^{t} A_i\right| = \sum_{i=1}^{t} |A_i| - \sum_{1 \leq i_1 < i_2 \leq t} |A_{i_1} \cap A_{i_2}| + \ldots + (-1)^{t+1} |A_1 \cap A_2 \cap \ldots \cap A_t|.$$

Aibių sankirtai $A_{i_1} \cap \ldots \cap A_{i_r}$ priklauso tie ne didesni už n skaičiai, kurie dalijasi iš pirminių skaičių sandaugos $p_{i_1} p_{i_2} \ldots p_{i_r}$. Tokių skaičių yra $n/(p_{i_1} p_{i_2} \ldots p_{i_r})$. Taigi

$$n - |A| = n - \sum_{i=1}^{t} \frac{n}{p_i} + \sum_{n \leq i_1 < i_2 \leq t} \frac{n}{p_{i_1} p_{i_2}} - \ldots + (-1)^t \frac{n}{p_1 p_2 \ldots p_t}. \tag{94}$$

Belieka įsižiūrėti ir įsitikinti, kad reiškiniai (93) ir (94) yra lygūs.

Pasinaudojus gautąja Eulerio funkcijos išraiška nesunku įrodyti tokią šios funkcijos savybę:

132 teorema. *Jeigu natūriniai skaičiai m, n yra tarpusavyje pirminiai, t.y. $(m, n) = 1$, tai*

$$\varphi(m \cdot n) = \varphi(m)\varphi(n). \tag{95}$$

Funkcijos, apibrėžtos natūraliųjų skaičių aibėje ir turinčios šią savybę, vadinamos multiplikatyviomis. Taigi Eulerio funkcija yra multiplikatyvi.

19.3. Kvadratiniai lyginiai

Štai tokia padėtis: didžiuliame ir sudėtingų sąryšių pilname realiųjų skaičių kūne kvadratines lygtis spręsti galime greitai, o paprastame dalybos iš skaičiaus liekanų kūne greito būdo nėra.

Kvadratines lygtis jau prieš kelis tūkstančius metų sprendė babiloniečių moksleiviai. Mūsų dienų moksleivius gelbsti diskriminantas: į jį pažiūrėjus galima pasakyti, ar lygtis turi sprendinių ir kokie jie.

O mes panagrinėkime kvadratinės lygties

$$x^2 + bx + c = 0$$

sprendimą žiede $\mathbb{Z}_n$, čia, žinoma, b, c yra sveikieji skaičiai. Kitaip tariant, panagrinėkime, su kokiomis x reikšmėmis lyginys

$$x^2 + bx + c \equiv 0 \pmod{n} \tag{96}$$

yra teisingas.

Jeigu pirminis p dalija n ir su kokia nors x reikšme (96) yra teisingas, tai su ta pačia reikšme bus teisingas ir lyginys

$$x^2 + bx + c \equiv 0 \pmod{p}. \tag{97}$$

Taigi galima pradėti nuo kvadratinių lygčių tyrinėjimo kūnuose $\mathbb{F}_p$ tikintis (ir pagrįstai!), kad, išnagrinėjus tokius atvejus, bus nesunku pereiti ir prie bendrojo.

Tarkime, kad $p > 2$ yra pirminis, ir nagrinėkime (97) lyginį. Atlikime keletą paprastų pertvarkių:

$$\begin{aligned}
&x^2 + bx + c \equiv 0 \pmod{p},\\
&4x^2 + 2(2x)b + b^2 \equiv b^2 - 4c \pmod{p},\\
&(2x + b)^2 \equiv b^2 - 4c \pmod{p},\\
&y^2 \equiv D \pmod{p},\ y = 2x + b,\ D = b^2 - 4c.
\end{aligned}$$

Taigi diskriminantas pasirodo ir čia. Jis lemia sprendinių skaičių, tačiau kaip – nelengva pasakyti.

Matome, kad svarbiausia išnagrinėti visų paprasčiausio lyginio

$$x^2 \equiv a \pmod{p}\ (a \neq 0) \tag{98}$$

atvejį. Žinome, kad kūne n-ojo laipsnio lygtis negali turėti daugiau kaip n sprendinių. Mūsų atveju padėtis tokia: jeigu sprendinių yra – tai jų lygiai du. Išties, jeigu x_0 tenkina lyginį, tai ir $x_1 = p - x_0$ tenkins.

Kada (98) lyginys turi sprendinį? Atsakyti į šį klausimą galima greitai, jeigu pasinaudosime gerais skaičių teorijos įrankiais.

111 apibrėžimas. *Tegu p yra pirminis skaičius. Legendre'o (Ležandro) simboliu vadinama funkcija*

$$\left(\frac{a}{p}\right) = \begin{cases} 0, & \text{jei } a \equiv 0 \ (mod\ p),\\ 1, & \text{jei lyginys } x^2 \equiv a \ (mod\ p),\ \text{turi sprendinių,}\\ -1, & \text{jei lyginys } x^2 \equiv a \ (mod\ p),\ \text{neturi sprendinių.} \end{cases}$$

Atrodo, kokia nauda iš tokio apibrėžimo? Ar jis nėra tik paprastas trijų atvejų ženklinimas skaičiais? Tačiau su skaičiais galima atlikti veiksmus. O veikiant visada galima ką nors pasiekti. Šiuo atveju veiksmų su Legendre'o simboliais efektyvumą lemia geros jų savybės.

133 teorema. *Su bet kokiu pirminiu skaičiumi p teisingos šios lygybės:*

$$\left(\frac{a^2}{p}\right) = 1, \qquad \left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right),$$
$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \ (mod\ p), \qquad \left(\frac{a+kp}{p}\right) = \left(\frac{a}{p}\right),$$
$$\left(\frac{-1}{p}\right) = (-1)^{(p-1)/2}, \qquad \left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}.$$

Ir dar vienas teiginys, kuris padeda skaičiuoti Legendre'o simbolius. Tai Gauso dėsnis – vienas iš pačių įstabiausių skaičių teorijos teiginių.

134 teorema. *Su bet kokiais skirtingais pirminiais skaičiais p, q teisinga lygybė*

$$\left(\frac{p}{q}\right) \cdot \left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}.$$

Kelios iš suformuluotų simbolio savybių visiškai akivaizdžios, kitų esmės lengvai neįžvelgsi. Įrodymus galite susirasti skaičių teorijos vadovėliuose.

O dabar išbandykime Legendre'o simbolių skaičiavimo techniką. Tarkime, mums reikia nustatyti, ar turi sprendinių lyginys

$$x^2 \equiv 46 \ (\mathrm{mod}\ 57).$$

Skaičiuokime naudodamiesi Legendre'o simbolio savybėmis:

$$\begin{aligned}\left(\frac{46}{57}\right) &= \left(\frac{2\cdot 23}{57}\right) = \left(\frac{2}{57}\right)\cdot\left(\frac{23}{57}\right) = \left(\frac{23}{57}\right)\\ &= (-1)^{\frac{(23-1)(57-1)}{4}}\left(\frac{57}{23}\right) = -\left(\frac{57}{23}\right) = -\left(\frac{2\cdot 23+11}{23}\right)\\ &= -\left(\frac{11}{23}\right) = -(-1)^{\frac{(11-1)(23-1)}{4}}\left(\frac{23}{11}\right) = \left(\frac{1}{11}\right) = 1.\end{aligned}$$

Taigi lyginys sprendinį turi. Kaip jį galima surasti? Tiesa yra tokia: greito metodo, tinkančio visiems pirminiams, nėra! Taigi belieka perrinkti pretendentus, tikintis, kad vienas iš dviejų sprendinių ilgai nesislapstys. Tiesa, yra vienas atvejis, kai sprendinį galima rasti skaičiavimais.

135 teorema. *Tegu pirminis skaičius p, dalijant iš 4, duoda liekaną 3, t.y. $p \equiv 3\ (mod\ 4)$. Jeigu lyginys $x^2 \equiv a\ (mod\ p)$ turi sprendinių, tai vienas iš jų yra*

$$x_0 \equiv a^{\frac{p+1}{4}}\ (mod\ p).$$

Įrodymas. Kadangi lyginys turi sprendinių, tai iš Legendre'o simbolio savybių gauname

$$\left(\frac{a}{p}\right) = 1, \quad a^{\frac{p-1}{2}} \equiv \left(\frac{a}{p}\right)\ (\mathrm{mod}\ p), \quad a^{\frac{p-1}{2}} \equiv 1\ (\mathrm{mod}\ p).$$

Tada

$$x_0^2 \equiv a^{\frac{p+1}{2}} \equiv a \cdot a^{\frac{p-1}{2}} \equiv a \pmod{p}.$$

Tai beveik viskas, ką galima pasakyti apie kvadratinių lyginių sprendimą pirminiu moduliu. Greito sprendimo būdo tiesiog nėra.

O dabar panagrinėkime sudėtinio modulio atvejį. Apsiribokime atveju, kai $n = pq$, čia p, q – du skirtingi pirminiai skaičiai. Surasti lyginio

$$x^2 \equiv a \pmod{n} \tag{99}$$

sprendinį reiškia rasti tokią x reikšmę v, kad $v^2 - a$ dalytųsi iš n. Tačiau tada ir p bei q dalytų $v^2 - a$, taigi reikšmė $x = v$ būtų abiejų lyginių

$$x^2 \equiv a \pmod{p}, \quad x^2 \equiv a \pmod{q} \tag{100}$$

sprendinys. Kita vertus, jeigu rastume vieną x reikšmę, tinkančią abiem (100) lyginiams, ji būtų ir (99) sprendinys. Tačiau spręsdami (100) lyginius atskirai, vargu ar galime tikėtis, kad gausime tą patį skaičių. Gautume, pavyzdžiui, kad pirmajam lyginiui tinka visi skaičiai m, tenkinantys sąlygą $m \equiv m_1 \pmod{p}$, o antrajam – visi skaičiai, tenkinantys sąlygą $m \equiv m_2 \pmod{q}$. Taigi reikia skaičiaus, kuris tenkintų abi sąlygas! Išspręsti šį uždavinį galime pasinaudoję dar vienu labai senu, bet kriptografijai labai naudingu įrankiu – kiniškąja liekanų teorema.

136 teorema. *Tegu $n_1, n_2, \ldots, n_t$ yra tarpusavyje pirminiai skaičiai, o $m_1, m_2, \ldots, m_t$ – bet kokie sveikieji skaičiai. Tegu $n = n_1 n_2 \cdots n_t$, o sveikieji skaičiai $a_1, a_2, \ldots, a_t$ tenkina sąlygas*

$$a_i \cdot \frac{n}{n_i} \equiv 1 \ (mod\ n_i), \quad i = 1, 2, \ldots, t.$$

Sudarykime skaičių

$$M = m_1 a_1 \cdot \frac{n}{n_1} + m_2 a_2 \cdot \frac{n}{n_2} + \ldots + m_t a_t \cdot \frac{n}{n_t}.$$

Tada šis skaičius tenkina visus lyginius $M \equiv m_i \ (mod\ n_i)$, $i = 1, 2, \ldots, t$.

Įrodymas. Įsitikinkime, pavyzdžiui, kad $M \equiv m_1 \pmod{n_1}$. Kadangi visi dėmenys skaičiaus M išraiškoje, išskyrus pirmąjį, dalijasi iš n_1, tai

$$M \equiv m_1 a_1 \cdot \frac{n}{n_1} \pmod{n_1}.$$

Tačiau sandaugą $a_1 \cdot \frac{n}{n_1}$ galime pakeisti vienetu, taigi $M \equiv m_1 \pmod{n_1}$.

Žinoma, pastebėjote, kad skaičiai a_i yra skaičių $\frac{n}{n_i}$ atvirkštiniai moduliu n_i. Kaip juos rasti, jau žinome.

O dabar sugrįžkime prie lyginių (99), (100). Tarkime, pirmojo lyginio sprendiniai yra $\pm x_1$, o antrojo – $\pm x_2$. Tada suradę skaičius a, b, kad

$$aq \equiv 1 \pmod{p}, \quad bp \equiv 1 \pmod{q},$$

galėsime pagal kinų liekanų teoremą sudaryti ir (99) sprendinius

$$x = \pm x_1 aq \pm x_2 bp.$$

Taigi gauname net keturis sprendinius. Taisyklė „sprendinių nėra daugiau nei lygties laipsnis“ galioja tik kūnuose!

19.4. Natūrinių skaičių skaidymas

Skaidyti natūraliuosius skaičius pirminiais daugikliais – sunkus skaičiavimo uždavinys. Tačiau tai nereiškia, kad skaidyti kiekvieną didelį skaičių yra sunku.

Tarkime, n yra didelis natūralusis skaičius. Uždavinys – surasti jo pirminius daliklius. Pirmiausia, kas ateina į galvą – bandyti jį dalyti iš pirminių, išrikiuotų didėjimo tvarka: $p_1 < p_2 \ldots$ Jeigu n yra sudėtinis skaičius, tai $n = n_1 n_2$ ir vienas iš skaičių n_1, n_2 yra ne didesnis už $\sqrt{n}$. Taigi ieškant pirminių daliklių tokiu elementarios perrankos būdu, tikrai neteks atlikti daugiau kaip $\sqrt{n}$ perrankų. Tačiau lyginant su dydžiu $\log_2 n$, nusakančiu bitų skaičių, reikalingą n užrašyti, perrankų gali tekti atlikti labai daug. Todėl toks algoritmas nėra efektyvus.

Kai nėra visiems atvejams tinkančių efektyvių sprendimo taisyklių, kliautis atsitiktine sėkme nėra blogas sprendimas. Panagrinėkime Pollardo sugalvotą skaičių skaidymo algoritmą, kuriame sėkmė vaidina svarbų vaidmenį.

Pollardo ρ metodas

Tarkime, n yra natūralusis skaičius, o p – mums nežinomas jo pirminis daliklis. Atsitiktinai pasirinkime skaičius $s_1, s_2, \ldots, s_m$. Kai m yra pakankamai didelis skaičius, tikėtina, kad atsiras du skaičiai s_i, s_j, kad $s_i \equiv s_j \pmod{p}$, tačiau $s_i \not\equiv s_j \bmod n$. Tada skirtumas $s_i - s_j$ dalysis iš p, bet nesidalys iš n. Bendrasis didžiausiasis skirtumo ir n daliklis $n_1 = (s_i - s_j, n)$ bus nelygus vienam, taigi tokiu atveju gautume skaidinį $n = n_1 n_2$ ir toliau galėtume ieškoti mažesnių skaičių n_1, n_2 pirminių daliklių. Tačiau ar toks būdas nėra toks pat neefektyvus kaip ir tiesioginė perranka? Iš tikrųjų, toks jis ir būtų, jeigu skaičiuotume visus bendruosius didžiausiuosius daliklius $(s_i - s_j, n)$. Viena gera idėja paverčia jį maždaug dukart spartesniu už tiesioginę daliklių perranką.

Visų pirma tarkime, kad atsitiktiniams skaičiams parinkti sudarėme tam tikrą funkciją $f : \mathbb{Z}_n \to \mathbb{Z}_n$, kurios reikšmės išsibarsčiusios aibėje $\mathbb{Z}_n$ gana „atsitiktinai“. Be to, tarkime, ši funkcija turi savybę: bet kokiam natūraliajam a iš $u \equiv v \pmod{a}$ seka $f(u) \equiv f(v) \pmod{a}$. Šią savybę turi, pavyzdžiui, daugianariai. Daugelis aukštesnio laipsnio daugianarių f neblogai tinka šiam metodui. Atsitiktinai pasirinkę $s_0 \in \mathbb{Z}_n$, kitas reikšmes skaičiuokime taip:

$$s_1 \equiv f(s_0) \pmod{n},\ s_2 \equiv f(s_1) \pmod{n},\ \ldots,\ s_m \equiv f(s_{m-1}) \pmod{n}.$$

Tarkime su skaičiais $0 \le i < j$ teisingas sąryšis $s_i \equiv s_j \pmod p$, čia p yra mums nežinomas pirminis n daliklis. Pažymėkime $k = j - i$, tada $j = i + k$, $s_i \equiv s_{i+k} \pmod n$. Panagrinėkime tokią teisingų lyginių lentelę:

$$\begin{array}{rcll} f(s_i) & \equiv & f(s_{i+k}) \pmod p & \text{arba} \quad s_{i+1} \equiv s_{i+1+k} \pmod p \\ f(s_{i+1}) & \equiv & f(s_{i+1+k}) \pmod p & \text{arba} \quad s_{i+2} \equiv s_{i+2+k} \pmod p \\ & \cdots & & \cdots \qquad \cdots \\ f(s_{i+k-1}) & \equiv & f(s_{i+k+k-1}) \pmod p & \text{arba} \quad s_{i+k} \equiv s_{i+2k} \pmod p \end{array}$$

Tačiau iš lyginių $s_i \equiv s_{i+k} \pmod p, s_{i+k} \equiv s_{i+2k} \pmod p$ gauname, kad $s_i \equiv s_{i+2k} \pmod p$. Samprotaudami toliau, gautume, kad lyginys $s_i \equiv s_{i+tk} \pmod p$ teisingas su bet kokiu natūraliuoju t. Jeigu yra toks t, kad $tk = i$, tai tada turime teisingą lyginį $s_i \equiv s_{2i} \pmod p$. Taigi daliklio paiešką galime atlikti taip: skaičiuojame s_i ir s_{2i} ir $d = (s_{2i} - s_i, n)$, kol gauname $d > 1$.

Skaičiavimus organizuoti patogu taip:

1. parenkame funkciją $f : \mathbb{Z}_n \to \mathbb{Z}_n$ ir atsitiktinį $r \in \mathbb{Z}_n$;
2. priskiriame reikšmes $a = r, b = r$;
3. skaičiuojame $a = f(a), b = f(f(b))$;
4. randame $d = (b - a, n)$; jei $d \neq 0, n$ – rastas netrivialus n daliklis; jei $d = 1$, kartojame 3) žingsnį.

Trečiajame žingsnyje skaičiuojami dydžiai s_i ir s_{2i}.

Išbandykime šį metodą, pavyzdžiui, su $n = 34571317791$. Pasirinkta funkcija $f(x) \equiv x^2 + 732 \pmod n$. Kelių skaičiavimo žingsnių rezultatai pateikti lentelėje

a	b	$(b - a, n)$
142107	20194400180	1
20194400180	21542042492	3
17674479849	10362019826	1
21542042492	16189403312	141

Jau antrajame žingsnyje suradome daliklį 3. Žinoma, kad skaičius dalijasi iš 3, galėjome nustatyti iš karto, naudodamiesi dalybos iš 3 požymiu: skaitmenų suma dalijasi iš 3. Taigi

$$34571317791 = 141 \cdot 34571317791 = 3 \cdot 47 \cdot 34571317791.$$

Jeigu bandytume skaidyti skaičių 34571317791 – nieko nelaimėtume. Šis skaičius yra pirminis.

Pollardo $p-1$ metodas

Pollardas sugalvojo dar vieną skaičių skaidymo metodą, kuriuo naudojantis kartais pirminį daliklį galima rasti labai greitai.

112 apibrėžimas. *Tegu m, B yra du natūriniai skaičiai. Skaičių m vadinsime B-glodžiu, jeigu visi jo pirminiai dalikliai yra ne didesni už B.*

Pollardo $p-1$ metodas skaičiui n skaidyti yra efektyvus tada, kai n turi pirminį daliklį, kuriam $p-1$ yra B-glodus su nedidele B reikšme. Kitaip tariant – kai bent vienam pirminiam dalikliui p visi skaičiaus $p-1$ pirminiai dalikliai yra maži. Todėl šis Pollardo metodas ir vadinamas $p-1$ metodu.

Tarkime, n yra skaičius, kurį reikia išskaidyti, o p – mums nežinomas pirminis daliklis. Mažoji Fermat teorema teigia: jei skaičius a nesidalija iš p, tai $a^{p-1} \equiv 1 \pmod{p}$. Tačiau tada ir su bet kokiu natūraliuoju skaičiumi t lygybė $a^{t(p-1)} \equiv 1 \pmod{p}$ taip pat teisinga. Metodo esmė yra tokia: darant prielaidą, kad $p-1$ sudarytas iš mažų pirminių daliklių, galima pabandyti sukonstruoti kokį nors $p-1$ kartotinį $N = t(p-1)$. Tada būtų teisinga lygybė $a^N \equiv 1 \pmod{p}$, t.y. $a^N - 1$ dalytųsi iš p. Kadangi ir n dalijasi iš p, tai galėtume Euklido algoritmu suskaičiuoti $(a^N - 1, n) = d$ ir rasti netrivialų n daliklį. Žinoma, skaičius $a^N - 1$ gali būti tiesiog milžiniškas, tačiau jo reikšmė skaičiuojant bendrąjį didžiausiąjį daliklį visai nereikalinga, pakanka laipsnius skaičiuoti moduliu n.

Parinkti skaičių N galima įvairiais būdais. Pavyzdžiui, galima pasirinkus B apibrėžti $N = BMK(2, 3, \ldots, B)$, čia $BMK(a_1, a_2, \ldots, a_k)$ žymi bendrąjį mažiausią skaičių kartotinį. Šio didelio skaičiaus reikšmę irgi nebūtina skaičiuoti iš anksto, pakanka surasti visus pirminius skaičius q, kurie yra mažesni už B ir reikšmę a^N apskaičiuoti palaipsniui. Štai tokio algoritmo aprašymas:

1. pasirenkamas skaičius $a_0 = a$ ir surandami visi pirminiai skaičiai $q_1, q_2, \ldots, q_h$, ne didesni už B;

2. atliekama h algoritmo žingsnių, i-ajame žingsnyje skaičiuojama

$$r_i = \left[\frac{\log B}{\log q_i}\right], \quad a_i \equiv a_{i-1}^{r_i} \pmod{n};$$

3. skaičiuojama $d = (a_i - 1, n)$; jeigu $d \neq 1, n$ – skaičiaus n daliklis rastas.

Pabandykime išskaidyti šiuo metodu gana didelį natūralųjį skaičių

$$n = 406222941815702227.$$

Galima iš pradžių pasirinkti nelabai didelį B, o tada, jeigu nepasiseka, skaičiuoti su didesniu. Imkime, pavyzdžiui, $B = 20$. Šis skaičius yra nedidelis,

todėl galime visų skaičių iki B bendrąjį mažiausiąjį kartotinį suskaičiuoti ir tiesiogiai:

$$N = 2^4 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 = 232792560.$$

Imkime, pavyzdžiui, $a = 2$; skaičiuodami gausime:

$$\begin{aligned} &2^{232792560} \equiv 327811457795757939 \pmod{n}, \\ &(327811457795757939 - 1, n) = 8893, \\ &n = 8893 \cdot 45678954437839. \end{aligned}$$

Žinoma, tokia greita sėkmė nusišypsos retai. Šįkart pasisekė todėl, kad iš anksto buvo parinktas geras skaičius.

Taigi natūraliųjų skaičių skaidymo uždavinys ne visada yra sudėtingas. Jeigu reikalingas sudėtinis skaičius, kurį būtų sunku išskaidyti dauginamaisiais, jo pirminiai dalikliai, sumažinti vienetu, neturi būti itin glodūs.

Efektyviausi šiuo metu žinomi skaičių skaidymo metodai naudoja skaičių laukų ir elipsinių kreivių savybes. Jų esmę paaiškinti būtų sudėtingiau nei nagrinėtųjų paprastų, bet dažnai greitai duodančių rezultatą Pollardo metodų.

19.5. Generuojantys elementai ir diskretieji logaritmai

Skaičiuodami su realiaisiais skaičiais, dažnai naudojamės apytikslėmis reikšmėmis. Tačiau baigtinėse grupėse apytikslis skaičiavimas tiesiog neturi prasmės, o paprastai formuluojami uždaviniai neturi paprastų ir efektyvių sprendimų. Logaritmo skaičiavimas baigtinėse grupėse – tokio uždavinio pavyzdys.

Tegu G yra baigtinė grupė, joje apibrėžta elementų daugyba. Jeigu elementus $a, b \in G$ sieja lygybė

$$a^x = b,$$

čia x yra sveikasis skaičius, tai jį pagal realiųjų skaičių pavyzdį natūralu pavadinti elemento b logaritmu pagrindu a. Tačiau verta tokį apibrėžimą patikslinti: būtų gerai, kad logaritmas būtų apibrėžtas vienareikšmiškai ir nereiktų papildomai nagrinėti, ar logaritmas duotuoju pagrindu egzistuoja, ar ne.

113 apibrėžimas. *Tegu G yra baigtinė ciklinė grupė, o $g \in G$ jos generuojantis elementas, t.y. toks elementas, kad*

$$G = \{g^0, g^1, \ldots, g^{N-1}\},$$

čia $N = |G|$ yra grupės eilė, t.y. jos elementų skaičius. Elemento y diskrečiuoju logaritmu pagrindu g vadinsime aibės $\mathbb{Z}_{N-1}$ skaičių x, su kuriuo $y = g^x$. Logaritmą žymėsime $x = \log_g y$.

Taigi diskrečiuosius logaritmus apibrėžėme tik ciklinėse grupėse, jų pagrindais gali būti tik generuojantys elementai. Ne visos baigtinės grupės yra ciklinės, tačiau ir ciklinės grupės ne retenybė. Pavyzdžiui, bet kokio baigtinio kūno $\mathbb{F}_{p^m}$ nenulinių elementų aibė $\mathbb{F}_{p^m}^*$ daugybos atžvilgiu sudaro ciklinę grupę. Dažniausiai kriptografijos reikmėms naudosime grupę $\mathbb{F}_p^* = \{1, 2, \ldots, p-1\}$.

Diskrečiojo logaritmo pagrindas turi būti generuojantis elementas. Kaip surasti bent vieną? Kadangi tų generuojančių elementų yra pakankamai daug, tai ieškojimas pasikliaujant sėkme nėra bloga išeitis. O nustatyti, ar pasirinktas elementas yra generuojantis, galime naudodamiesi tokiu kriterijumi:
tegu N yra ciklinės grupės G eilė, $g \in G$ jos elementas; jei $g^d \neq 1$ su visais netrivialiais N dalikliais d, tai g yra generuojantis elementas.

Panagrinėkime, pavyzdžiui, atvejį $G = \mathbb{F}_7, g = 2$. Kadangi grupės eilė $N = 6$, tai pakanka patikrinti, ar nei vienas iš laipsnių $2^2, 2^3$ nelygus vienetui. Kadangi $2^3 \equiv 1 \pmod 7$, tai $g = 2$ nėra generuojantis elementas. Tačiau $h = 3$ tenkina šį kriterijų, taigi yra generuojantis elementas.

Diskretieji logaritmai turi panašias savybes kaip ir logaritmai realiųjų skaičių aibėje. Tačiau yra vienas esminis skirtumas: apytikslis diskrečiųjų logaritmų skaičiavimas neturi prasmės, o efektyvių būdų surasti tikslias reikšmes kol kas niekas nesugalvojo.

137 teorema. *Tegu G yra ciklinė grupė, N – jos eilė, o g – generuojantis elementas. Tada*

1. *su visais $x, y \in G$ teisinga lygybė $\log_g(x \cdot y) \equiv \log_g x + \log_g y \ (mod\ N)$;*
2. *su visais $x \in G$ ir su visais sveikaisiais k teisinga lygybė $\log_g x^k \equiv k \log_g x \ (mod\ N)$.*

Teiginiai beveik akivaizdūs, panagrinėkime pirmąjį. Tegu $u = \log_g x, v = \log_g y, w = \log_g(x \cdot y)$, tada

$$x = g^u,\ y = g^v,\ x \cdot y = g^u \cdot g^v = g^{u+v} = g^w.$$

Tačiau iš laipsnių $g^a = g^b$ lygybės išplaukia $a \equiv b \pmod N$, taigi $u + v \equiv w \pmod N$.

Efektyvių metodų diskretiesiems logaritmams skaičiuoti nėra. Visada galima ieškoti diskrečiojo logaritmo reikšmės perrankos būdu: tikrinti galimas reikšmes, kol pasitaikys tikroji. Yra ir šiek tiek išradingesnių metodų, kurie, nors ir negarantuoja greitos sėkmės, kartais padeda rasti diskretųjį logaritmą gana greitai.

Shankso metodas

Šiuo metodu ieškoma elementų iš $\mathbb{F}_p^*$ diskrečiųjų logaritmų pagrindu g, čia p – pirminis skaičius. Bet kurio elemento $y \in \mathbb{F}_p^*$ diskretusis logaritmas

x yra skaičius, ne didesnis už $p-1$. Pažymėkime $m = [\sqrt{p-1}]+1$; padaliję x iš m su liekana, gautume

$$x = im + j, \quad 0 \le i < m,\ 0 \le j < m. \tag{101}$$

Taigi

$$g^{mi+j} = y, \quad g^{mi} = yg^{-j}.$$

Šia lygybe ir remiasi algoritmo idėja: galima iš anksto apskaičiuoti reikšmes ir susidaryti porų $\langle i, g^{mi}\rangle$, $i = 0, 1, \ldots, m-1$, lentelę; tada sudaryti kitą lentelę iš porų $\langle j, yg^{-j}\rangle$ ir ieškoti abiejose lentelėse įrašų su vienodomis antrosiomis komponentėmis, t.y. lygybės $g^{mi} = yg^{-j}$. Suradę atitinkamus i ir j, pagal (101) galime apskaičiuoti diskrečiojo logaritmo reikšmę. Blogiausiu atveju reiktų maždaug $\sqrt{p}$ skaičiavimų; ieškant logaritmo tiesioginės perrankos būdu, didžiausias tikrinimų skaičius, kurio gali prireikti, yra maždaug p. Šį metodą dar kartais vadina mažylio-milžino žingsnių metodu (baby-step-giant-step method). Mažais žingsneliais sudarome lenteles, randame i, j ir, žengę didelį žingsnį, apskaičiuojame diskrečiojo logaritmo reikšmę.

Pavyzdys. Apskaičiuokime skaičių $y_1 = 13, y_2 = 17, y_1, y_2 \in \mathbb{F}_{23}^*$ diskrečiuosius logaritmus pagrindu $g = 5$. Mūsų atveju $m = 5$; taigi lentelėse bus tik po penkias poras.

$i, j =$	g^{mi}	$y_1 g^{-j}$	$y_2 g^{-j}$
0	1	13	17
1	20	21	8
2	9	18	20
3	19	22	4
4	12	9	10

Iš lentelės matome

$$g^{2m} = y_1 g^{-4}, \quad g^{1m} = y_2 g^{-2},$$

taigi

$$\log_g y_1 = 2m + 4 = 14, \quad \log_g y_2 = m + 2 = 7.$$

Pollardas, kurio natūraliųjų skaičių skaidymo logaritmus nagrinėjome ankstesniame skyriuje, sugalvojo ir du diskrečiųjų logaritmų skaičiavimo metodus. Vienas vadinamas ρ-metodu, kitas – λ-metodu. Panagrinėsime pirmąjį, kuris primena ρ metodą skaičiams skaidyti.

Pollardo ρ-metodas

Reikia surasti skaičiaus $y \in \mathbb{F}_p^*$ diskretųjį logaritmą pagrindu g, čia g – generuojantis elementas. Metodo esmė tokia: vienas po kito skaičiuojami

elementai $x_{i+1} = f(x_i), x_i \in \mathbb{F}_p^*$, kad su atitinkamais a_i, b_i būtų teisinga lygybė

$$x_i \equiv y^{a_i} g^{b_i} \pmod{p},$$

čia skaičiai a_i, b_i irgi skaičiuojami vienas po kito. Jeigu gautume, kad su kokiais nors i, j teisinga lygybė

$$x_i \equiv x_j \pmod{p}, \quad y^{a_i} g^{b_i} \equiv y^{a_j} g^{b_j} \pmod{p}, \quad y^{a_i - a_j} \equiv g^{b_j - b_i} \pmod{p},$$

tai, pažymėję $x = \log_g y$, gautume lyginį

$$(a_i - a_j)x \equiv b_j - b_i \pmod{p}$$

reikšmei x rasti. Kaip ir naudojant panašų skaičių skaidymo metodą, daugelio tikrinimų $x_i \equiv x_j \pmod{p}$ galima išvengti skaičiuojant iš karto x_k ir x_{2k} bei tikintis, kad jie sutaps:

$$x_{k+1} = f(x_k), \quad x_{2k+2} = f(f(x_k)).$$

Jeigu $x_i = x_{2i}$, tai gautume tokią lygybę:

$$(a_i - a_{2i})x \equiv b_{2i} - b_i \pmod{p}.$$

Belieka apibrėžti sekų x_i, a_i, b_i skaičiavimo taisykles. Štai jos: $x_0 = 1$, $a_0 = b_0 = 0$,

$$x_{i+1} = \begin{cases} yx_i \pmod{p}, & \text{jei } 0 < x_i < p/3 \\ x_i^2 \pmod{p}, & \text{jei } p/3 < x_i < 2p/3 \\ gx_i \pmod{p}, & \text{jei } 2p/3 < x_i < p; \end{cases}$$

$$a_{i+1} = \begin{cases} 1 + a_i \pmod{p-1}, & \text{jei } 0 < x_i < p/3 \\ 2a_i \pmod{p-1}, & \text{jei } p/3 < x_i < 2p/3 \\ a_i \pmod{p-1}, & \text{jei } 2p/3 < x_i < p; \end{cases}$$

$$b_{i+1} = \begin{cases} b_i \pmod{p-1}, & \text{jei } 0 < x_i < p/3 \\ 2b_i \pmod{p-1}, & \text{jei } p/3 < x_i < 2p/3 \\ 1 + b_i \pmod{p-1}, & \text{jei } 2p/3 < x_i < p. \end{cases}$$

Lyginius $x_i \equiv y^{a_i} g^{b_i} \pmod{p}$ patikrinti nesudėtinga.

Pavyzdys. Tegu $p = 23$, tada $g = 5$ yra generuojantis elementas. Pabandykime Pollardo ρ-metodu surasti $x = \log_g y, y = 18$. Štai skaičiavimų

eiga:

i	x_i	a_i	b_i	x_{2i}	a_{2i}	b_{2i}
0	1	0	0	1	0	0
1	18	1	0	21	1	1
2	21	1	1	13	1	2
3	13	1	2	21	4	9
4	8	2	4	8	8	20

Taigi jau ketvirtajame žingsnyje pasisekė gauti lygybę $x_4 = x_8$. Dabar galime ieškoti ir logaritmo:

$$y^2g^4 \equiv y^8g^{20} \pmod{23}, \quad y^6 \equiv g^{-16} \pmod{23}, \quad 6x \equiv -16 \pmod{22},$$
$$3x \equiv -8 \pmod{11}, \quad 3x \equiv 3 \pmod{11}, \quad x \equiv 1 \pmod{11}.$$

Gauname du lyginio $6x \equiv -16 \pmod{22}$ sprendinius: $x = 1$ ir $x = 1 + 11 = 12$. Antroji reikšmė ir yra diskretusis logaritmas: $\log_5 18 = 12$.

Tačiau ne visada Pollardo metodo procesas baigiasi taip sėkmingai. Pavyzdžiui, skaičiuodami analogiškai su $y = 14$, gautume:

i	x_i	a_i	b_i	x_{2i}	a_{2i}	b_{2i}
0	1	0	0	1	0	0
1	14	1	0	12	2	0
1	12	2	0	6	4	0
2	6	4	0	15	5	0
3	15	5	0	15	5	0

Taigi gavome, kad sutampa ne tik $x_i = x_{2i}$, bet ir $a_i = a_{2i}, b_i = b_{2i}$. Iš to mums nėra jokios naudos. Tačiau galime pakartoti Pollardo algoritmą su kita pradine x_0 reikšme. Pavyzdžiui, imdami $x_0 = 3$, jau antrajame žingsnyje gauname:

$$x_2 = x_4, \quad a_2 = b_2 = 1, \quad a_4 = b_4 = 2,$$

ir logaritmą randame iš lyginio $yg \equiv y^2g^2 \pmod{23}$, arba $x \equiv -1 \pmod{22}$, $x = 21$.

20 Viešojo rakto kriptosistemos

Jeigu nėra saugaus būdo perduoti raktams, simetrinės kriptosistemos naudojimas ryšiui apsaugoti tiesiog neįmanomas. Kai didžiulių informacijos srautų judėjimas kompiuteriniais tinklais tapo kasdienybe, saugius ryšio kanalus

teko tiesiog užmiršti. Tiesa, diplomatinis bagažas egzistuoja ir dabar, tačiau dauguma iš mūsų nesame nei diplomatai, nei ruošiamės jais tapti. Taigi kompiuterių amžius iškėlė kriptografijai kone neįmanomą iššūkį: reikia šifruoti, bet saugiai perduoti raktų neįmanoma!

Galime prisiminti, kaip buvo atsakyta į klausimą: ar galima palaikyti labai patikimą ryšį nepatikimu kanalu, turint tik ribotus išteklius? C. Shannonas įrodė, kad tai įmanoma. Teigiamai atsakyta ir į klausimą apie šifravimą, kuriam nebūtini saugūs raktų perdavimo būdai.

Naująsias kompiuterių amžiaus kriptografijos kryptis 1976 metais pirmieji nurodė W. Diffie ir M. Hellman[19]. Idėja labai paprasta: kadangi neįmanoma sukurti tokios kriptosistemos, kurios šifravimui ir dešifravimui skirti raktai būtų nesusiję, tai reikia padaryti bent taip, kad, norint iš dešifravimo rakto nustatyti šifravimui skirtą raktą, tektų sugaišti tiek laiko, kad žmonių pasaulyje jis prilygtų kone amžinybei. Nereikia netgi amžinybės, pakanka, kad tokiam darbui prireiktų laiko, per kurį ir naudojamas raktas būtų pakeistas, ir užšifruotas pranešimas taptų nebesvarbus. Tada šifravimui skirtą raktą galima būtų paskelbti viešai, nebijant, kad iš jo bus greitai nustatytas dešifravimo raktas. Todėl tokia ideologija pagrįstos kriptosistemos ir vadinamos viešojo rakto kriptosistemomis.

Štai tokio viešo, bet saugaus paskelbimo pavyzdys. Tarkime, sugalvojau kriptosistemą, kurioje šifruojama naudojant raktą

$$\begin{aligned} k_e = \quad & 11739085287969531650666649599035831993898213898740 79 \\ & 90187848339927925066408945441063107863029870718402 3, \end{aligned}$$

o dešifravimo raktas – mažesnysis skaičiaus k_e pirminis daliklis. Teks gerokai pasidarbuoti jūsų programoms, kol nustatysite, kad

$$k_d = 10000000000000000000000000000000000000000000000151.$$

Neilgai trukus po Diffie ir Hellmano idėjų paskelbimo atsirado ir pirmosios viešojo rakto kriptosistemos. Jas sukūrė R. Merkle ir M. Hellmanas, o taip pat trijų autorių grupė: R. Rivestas, A. Shamiras ir L. Adlemanas. Jie laikomi viešojo rakto kriptografijos pionieriais. Tačiau būti laikomam pirmuoju – tai dar ne juo būti. Visai neseniai paaiškėjo, kad viešojo rakto kriptosistemų principai buvo sukurti maždaug dešimtmečiu anksčiau. Jų autoriai – Jungtinės Karalystės vyriausybinės organizacijos darbuotojai Jamesas Ellisas, Cliffordas Cocksas ir Malcolmas Williamsonas. Tačiau jų darbai buvo įslaptinti, todėl jie galėjo tik iš šalies stebėti, kaip laisvi akademinio sluoksnio žmonės plėtoja naujųjų laikų kriptografiją.

[19] W. Diffie, M. Hellman. New Directions in Cryptography. IEEE Transactions on Information Theory, vol. IT-22, Nov. 1976, p. 644–654.

20.1. Kuprinės ir šifrai

R. Merkle ir M. Hellmanas sukūrė viešojo rakto kriptosistemą, kuriai prigijo „kuprinės“ vardas[20]*. Ją verta panagrinėti dėl kelių priežasčių: viena vertus tai pirmoji viešojo rakto kriptosistema, antra – ją labai paprasta paaiškinti, trečia – tai kriptosistemos, kuri buvo įveikta pasitelkus ne didžiulius skaičiavimo išteklius, bet matematines idėjas, pavyzdys.*

„Kuprinės“ uždavinys formuluojamas taip:
ID: *natūraliųjų skaičių seka* $W = \{w_1, w_2, \ldots, w_n\}$ *ir skaičius* v.
U: *rasti skaičius* $x_i \in \{0, 1\}$, *kad*

$$v = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n,$$

arba nustatyti, kad tokių skaičių nėra.

Ši problema yra **NP** pilna, taigi kai įvesties duomenų (svorių) yra daug, ją spręsti sudėtinga. Tačiau kai „kuprinės“ svoriai w_i turi specialių savybių, „kuprinės“ uždavinys irgi gali būti greitai sprendžiamas.

114 apibrėžimas. *Sakysime, kad skaičiai* $w_1, w_2, \ldots, w_n$ *sudaro sparčiai didėjančią seką, jei visiems* $i > 1$ *teisinga nelygybė*

$$w_1 + w_2 + \ldots + w_{i-1} < w_i.$$

138 teorema. *Jei „kuprinės“ svoriai* $W = \{w_1, w_2, \ldots, w_n\}$ *sudaro sparčiai didėjančią seką, tai „kuprinės“ uždavinys sprendžiamas polinominiu algoritmu. Jeigu skaičių* v *galima išreikšti sistemos* W *svoriais, tai tokia išraiška yra vienintelė.*

Įrodymas. Tegu v – natūralusis skaičius. Ieškosime jo išraiškos

$$v = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n, \quad x_i \in \{0, 1\}.$$

Pirmiausia reikia patikrinti, ar $v \geq w_n$. Jeigu ši nelygybė teisinga, tai svorį w_n būtinai teks įtraukti į sumą, t. y. $x_n = 1$. Jeigu neteisinga, tai $x_n = 0$. Tada reikia lyginti skaičių $v - x_n w_n$ su w_{n-1} ir nustatyti x_{n-1} reikšmę. Matome, kad skaičių x_j reikšmės yra nustatomos vienareikšmiškai; taigi jei išraiška egzistuoja, tai ji vienintelė.

Uždavinio sprendimo algoritmą galima užrašyti taip:

1. $w := v, j := n$;
2. jei $w \geq w_j$, tai $x_j = 1$; jei $w < w_j$, tai $x_j = 0$; $w := w - x_j w_j$; $j := j - 1$;
3. jei $w = 0$, tai išraiška rasta; jei $w \neq 0, j = 0$, išraiška neegzistuoja; kitais atvejais reikia kartoti 2 žingsnį.

[20] R. C. Merkle, M. Hellman. Hiding Information and Signatures in Trapdoor Knapsacks. IEEE Transactions on Information Theory, v. 24, n. 5, 1978, p. 525–530.

Skaičiavimą sudaro ne daugiau kaip n žingsnių. Galime tarti, kad vienam žingsniui atlikti reikia $O(|v|)$ operacijų su bitais, čia $|v|$ žymi v užrašymui reikalingų bitų kiekį. Tada bendras operacijų skaičius bus

$$O(n|v|) = O((|w_1| + \ldots + |w_n|)|v|).$$

Taigi algoritmas yra polinominis.

Turint svorių sistemą $W = \{w_1, w_2, \ldots, w_n\}$, galima taip šifruoti nulių-vienetų blokus:

$$x_1 x_2 \ldots x_n \to c = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n.$$

Tačiau tenka pasukti galvą dėl dviejų dalykų. Kaip parinkti svorius, kad dviem skirtingiems blokams visada gautume du skirtingus skaičius c? Kaip dešifruoti c, t. y. kaip spręsti „kuprinės“ uždavinį, kad jis būtų sunkus kriptoanalitikui ir lengvas teisėtam šifro gavėjui?

Jei naudosime sparčiai augančią svorių sistemą, tiek šifravimo, tiek dešifravimo veiksmai bus atliekami sparčiai, tačiau ir kriptoanalitikas, ir teisėtas gavėjas turės vienodas galimybes.

Vieną iš sprendimo būdų 1976 metais pasiūlė Merkle ir Hellmanas.

Tegu $W = \{w_1, w_2, \ldots, w_n\}$ yra sparčiai didėjanti svorių sistema, p – skaičius, didesnis už visų svorių sumą (nebūtinai pirminis):

$$p > w_1 + w_2 + \ldots + w_n, \tag{102}$$

s, t – du tarpusavyje pirminiai su p skaičiai, tenkinantys sąlygą $st \equiv 1 \pmod{p}$. Vieną iš šių skaičių galime parinkti laisvai, o kitą surasti pasinaudoję Euklido algoritmu.

Algio privatųjį (slaptąjį) raktą sudaro svorių sistema W ir skaičius s, taigi $K_p =< W, s >$. Viešasis raktas bus svorių sistema

$$V = \{v_1, v_2, \ldots, v_n\}, \quad v_i \equiv w_i t \pmod{p}, \quad K_v = \langle V \rangle.$$

Ši svorių sistema nebėra sparčiai didėjanti, taigi galime tikėtis, kad paprasto algoritmo „kuprinės“ uždaviniui spręsti nėra. Pranešimas $M = m_1 m_2 \ldots m_n \in \{0,1\}^n$ bus šifruojamas taip:

$$C = e(M|K_v) = m_1 v_1 + m_2 v_2 + \ldots + m_n v_n.$$

Taigi šifras yra skaičius, ne didesnis už $n(p-1)$.

Kaip A gali dešifruoti C? Visų pirma jis turi suskaičiuoti

$$\begin{aligned} C_1 \equiv Cs \quad &\equiv m_1 s v_1 + m_2 s v_2 + \ldots + m_n s v_n \\ &\equiv m_1 w_1 + m_2 w_2 + \ldots + m_n w_n \pmod{p}. \end{aligned}$$

(102) sąlyga garantuoja, kad skirtingiems pranešimams M skaičiai C_1 irgi bus skirtingi. Kad surastų pranešimą M, Algis turi spręsti „kuprinės“

uždavinį su sparčiai didėjančių svorių sistema W ir skaičiumi C_1. Jau žinome, kad toks uždavinys sprendžiamas polinominiu algoritmu.

Deja, ši paprasta ir elegantiška kriptosistema turi didelį trūkumą. Pasirodė, kad iš viešojo rakto svorių sistemos V, pasinaudojus tam tikromis matematinėmis idėjomis įmanoma greitai rasti privatųjį raktą, t. y. sparčiai didėjančią svorių sistemą. Taigi ši kriptosistema nėra saugi. Buvo sugalvota įvairių kitų „kuprinės“ kriptosistemos variantų. Tačiau ir jie vienas po kito buvo įveikti. „Kuprinės“ tipo kriptosistemų ir jų analizės apžvalga yra pateikta straipsnyje[21].

„Kuprinės“ kriptosistema
Pranešimų aibė $\mathcal{M} = \{0,1\}^n$, šifrų aibė $\mathcal{C} \subset \mathbb{N}$. **Privatusis raktas**: $K_p = \langle W, s\rangle$, čia $W = \langle w_1, w_2, \ldots, w_n\rangle$ – sparčiai didėjanti svorių sistema; $w_1 + w_2 + \ldots + w_n < p, \quad (s,p) = 1$. **Viešasis raktas:** $K_v = \langle v_1, v_2, \ldots, v_n\rangle$, $v_i \equiv w_i s^{-1} \pmod p$. **Šifravimas:** $C = e(m_1 m_2 \ldots m_n \mid K_v) = m_1 v_1 + \ldots + m_n v_n$. **Dešifravimas:** $C_1 \equiv Cs \pmod p$, $C_1 = m_1 w_1 + \ldots + m_n w_n$, $m_1 \ldots m_n = d(C \mid K_p)$.

20.2. RSA

Ši trijų autorių – R. Rivesto, A. Shamiro ir L. Adlemano sukurta viešojo rakto kriptosistema yra pati populiariausia. Ja naudojamasi jau daugiau kaip dvidešimt metų. Dvidešimt metų trunkantys jos tyrinėjimai neatskleidė esminių saugumo spragų.

Šioje kriptosistemoje ir pranešimai, ir jų šifrai yra tos pačios aibės skaičiai.

Raktų parinkimas. Ryšio dalyvis A pasirenka du pirminius skaičius p, q ir sudaugina juos: $n = p \cdot q$. Dar reikia pasirinkti natūralųjį skaičių $e = e_A$, kad jis būtų tarpusavyje pirminis su $\phi(n) = (p-1)(q-1)$, t. y. $(e, (p-1)(q-1)) = 1$. Naudojantis Euklido algoritmu, reikia surasti skaičių $d = d_A$, kad būtų

$$e \cdot d \equiv 1 \pmod{(p-1)(q-1)}.$$

Dabar jau galima sudaryti raktus: viešąjį $K_v = \langle e, n\rangle$ ir privatųjį $K_p = \langle d\rangle$. Skaičius p, q geriausia iš viso ištrinti. Jų nebeprireiks, tačiau jeigu jie taptų žinomi kriptoanalitikui Z, jis surastų ir privatųjį raktą.

[21] Andrew M. Odlyzko. The Rise and Fall of Knapsack Cryptosystems. In: Cryptology and Computational Number Theory, Proceedings of Symposia in Applied Mathematics, vol. 42. American Mathematics Society, Providence, RI, 1990, p. 75–88. http://www.research.att.com/ amo/doc/arch/knapsack.survey.ps

Šifravimas. Pranešimai, kuriuos bus galima siųsti A, yra aibės $\mathcal{M} = \{1, 2, \ldots, n\}$ skaičiai. Šifravimas apibrėžiamas lygybe:

$$C = e(M|K_v) \equiv M^e \pmod{n}.$$

Dešifravimas. Dešifravimo algoritmas visiškai toks pat kaip ir šifravimo:

$$d(C|K_p) \equiv C^d \pmod{n}.$$

Įsitikinkime, kad dešifruojant visada gaunama $C^d \equiv M \pmod{n}$. Kadangi $n = pq$, tai pakanka įsitikinti, kad $C^d \equiv M \pmod{p}, C^d \equiv M \pmod{q}$.

Jei $M \equiv 0 \pmod{p}$, tai akivaizdu, kad $C \equiv 0 \pmod{p}$ ir

$$C^d \equiv 0 \equiv M \pmod{p}.$$

Tegu $(M, p) = 1$. Tada

$$C \equiv M^{ed} \equiv M^{1+t(p-1)(q-1)} \equiv M(M^{p-1})^{q-1} \pmod{p}.$$

Tačiau pagal Fermat teoremą $M^{p-1} \equiv 1 \pmod{p}$, taigi $C^d \equiv M \pmod{p}$. Aišku, kad analogiški samprotavimai tinka ir skaičiui q.

Kriptosistemos saugumas remiasi tuo, kad Z, norėdamas surasti privatųjį raktą d, turi spręsti lyginį $e \cdot d \equiv 1 \pmod{(p-1)(q-1)}$. Tačiau tam reikia žinoti $(p-1)(q-1)$. Kad šį skaičių surastų, Z turi išskaidyti n pirminiais daugikliais. Tai yra sunkus skaičiavimo uždavinys. Taigi kol efektyvus didelių pirminių skaičių skaidymo pirminiais daugikliais algoritmas nėra žinomas, tol RSA kriptosistema yra saugi. Gali būti, kad RSA galima įveikti ir be greito natūrinių skaičių skaidymo algoritmo, tačiau to niekas kol kas nežino.

RSA kriptosistema
Pranešimų ir šifrų aibės $\mathcal{M} = \mathcal{C} = \mathbb{Z}_n$, $n = pq$, skaičiai p, q yra pirminiai.
Privatusis raktas: $K_p = \langle d \rangle$, $(d, \varphi(n)) = 1, \varphi(n) = (p-1)(q-1)$.
Viešasis raktas: $K_v = \langle n, e \rangle$, $ed \equiv 1 \pmod{\varphi(n)}$.
Šifravimas: $C = e(M\|K_v) \equiv M^e \pmod{n}$.
Dešifravimas: $M = d(C\|K_p) \equiv C^d \pmod{n}$.

Norint RSA kriptosistema šifruoti, pavyzdžiui, lietuviškus tekstus, reikia susitarti, kaip jie bus verčiami skaičiais. Galima, pavyzdžiui, raides interpretuoti kaip skaičiavimo sistemos su pagrindu $N = 33$ skaitmenis, o tarpą tarp žodžių žymėti nuliu:

A	Ą	B	C	Č	D	E	Ę	Ė	F	G	H	I	Į	Y	J
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
K	L	M	N	O	P	R	S	Š	T	U	Ų	Ū	V	Z	Ž
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Tada kiekvieną žodį galėsime užrašyti skaičiumi, pavyzdžiui,

$$KNYGA = 17 \cdot 33^4 + 20 \cdot 33^3 + 15 \cdot 33^2 + 11 \cdot 33 + 1 = 20896096.$$

Galima ilgą tekstą skaidyti sakiniais ir pastaruosius keisti skaičiais. Galima tiesiog versti tekstą dvejetainių simbolių srautu, pastarąjį skaidyti blokais ir interpretuoti juos kaip natūraliųjų skaičių dvejetaines išraiškas.

20.3. RSA saugumas

Keletas paprastų RSA kriptosistemos atakų, kurių nesudėtinga išvengti.

Jau minėjome, kad esminių RSA kriptosistemos saugumo spragų kol kas neaptikta. Tačiau neatsargiai ja naudojantis, kriptoanalitikas gali įgyti galimybę surasti privatųjį raktą. Pavyzdžiui, jei pranešimas M nėra tarpusavyje pirminis su n, tai jo šifras C irgi turės su n netrivialų bendrą daliklį (bent vieną iš skaičių p, q) ir jį galima suskaičiuoti Euklido algoritmu. Tiesa, tikimybė, kad pranešimas bus p arba q kartotinis, labai nedidelė, viso labo tik

$$\frac{1}{p} + \frac{1}{q} - \frac{1}{pq}.$$

Reikia šiek tiek atsargumo parenkant pirminius skaičius p, q. Tarkime, pavyzdžiui, A pavyko surasti vieną, jo nuomone, pakankamai didelį pirminį skaičių p, pavyzdžiui, $p = 3138428376749$ ir jis nutarė kito pirminio ieškoti netoli p, t. y. tikrinti, ar $p + 2, p + 3, \ldots$ yra pirminiai. Neilgai trukus jis tokį pirminį surado ir, apskaičiavęs RSA modulį

$$n = 984973267632859020525139 1,$$

jį paskelbė. Kriptoanalitikas Z, turėdamas marias laisvo laiko ir gerų mokyklinės matematikos žinių, užsirašė paprastas lygybes

$$4n = (p+q)^2 - (p-q)^2, \quad (p+q)^2 = 4n + (p-q)^2$$

ir sukūrė paprastą programą, kuri tikrina, ar kartais kuris nors iš skaičių

$$4n + 2^2, 4n + 3^2, 4n + 4^2, \ldots$$

nėra pilnas kvadratas. Ilgai jo kompiuteriui dirbti nereikėjo:

$$4n + 110^2 = 6276856753608^2.$$

Dabar liko vieni niekai:

$$\begin{cases} p + q = 6276856753608, \\ p - q = 110 \end{cases}$$

ir $p = 3138428376749, q = 3138428376859$. Šio pavyzdžio moralas toks: jeigu norite saugumo, skaičius $|p - q|$ turi būti pakankamai didelis.

Atskirais atvejais gali būti sėkmingos ir kitokios atakos. Pavyzdžiui, efektyvią ataką galima atlikti, kai privačiojo rakto komponentė d yra maža palyginus su n.

139 teorema. *Jeigu $K_v = \langle n, e\rangle, K_p = \langle d\rangle$ yra RSA kriptosistemos raktai ir p, q, d tenkina sąlygas*

$$q < p < 2q, \quad d < \frac{1}{3} n^{\frac{1}{4}},$$

tai iš viešojo rakto polinominiu algoritmu galima rasti privatųjį.

Šią mažo privačiojo rakto ataką sugalvojo M. Wieneris[22]. Nesigilindami į teoremos įrodymą, panagrinėkime, kaip vykdoma privačiojo rakto paieška.

Kiekvieną paprastąją nesuprastinamą trupmeną $\frac{a}{b}$ $(a < b)$ galima užrašyti baigtine grandinine trupmena. Kaip tai daroma, pamatysite iš pavyzdžio. Tarkime, $a = 17,\ b = 57$. Pirmiausia atlikime didžiausiojo bendrojo daliklio ieškojimo Euklido algoritmu žingsnius:

$$\begin{aligned} 57 &= 3 \cdot 17 + 6, \quad \frac{57}{17} = 3 + \frac{6}{17} \\ 17 &= 2 \cdot 6 + 5, \quad \frac{17}{6} = 2 + \frac{5}{6}, \\ 6 &= 1 \cdot 5 + 1, \quad \frac{6}{5} = 1 + \frac{1}{5}. \end{aligned}$$

Skaičiaus skleidinys grandinine trupmena atrodys taip:

$$\frac{17}{57} = \frac{1}{\frac{57}{17}} = \frac{1}{3 + \frac{6}{17}} = \ldots = \frac{1}{3 + \cfrac{1}{2 + \cfrac{1}{1 + \frac{1}{5}}}}.$$

[22] M. Wiener. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory. **36**, 1990, p. 553–558.

Grandininę trupmeną visiškai apibrėžia skaičiai vardikliuose. Taigi kad nereiktų rašyti tokios ilgos grandinės, galime ją žymėti tiesiog

$$\frac{17}{57} = [3; 2; 1; 5].$$

Sumažindami koeficientų kiekį, sudarysime dar tris grandinines trupmenas:

$$t_1 = [3] = \frac{1}{3},\ t_2 = [3;2] = \frac{1}{3+\frac{1}{2}} = \frac{2}{7},\ t_3 = [3;2;1] = \frac{3}{10},\ t_4 = [3;2;1;5] = \frac{17}{57}.$$

Šias trupmenas vadinsime skaičiaus $\frac{17}{57}$ konvergentėmis. Paskutinioji konvergentė lygi pačiam skaičiui. Bet kokio skaičiaus konvergentės skaičiuojamos efektyviai.

Ryšys su RSA kriptosistema yra toks:

jeigu RSA dydžiai tenkina Wienerio teoremos sąlygas, tai kriptoanalitikui žinomos trupmenos $\frac{e}{n}$ vienos konvergentės vardiklis lygus privačiajam raktui d!

Įveikti RSA reiškia sugalvoti efektyvų būdą, kaip dešifruoti šifrą, kai dešifravimo raktas nežinomas. Jeigu turėtume efektyvų algoritmą kriptosistemos moduliui n skaidyti pirminiais daugikliais, surastume $\varphi(n)$, o tada jau ir privatųjį raktą. Taigi skaičių skaidymo uždavinio sprendimas duoda būdą įveikti ir RSA. Galbūt įmanoma RSA šifrą dešifruoti ir nenustačius privataus rakto? Ar toks „aplinkinis“ RSA įveikimo metodas taip pat duotų galimybę efektyviai skaidyti natūraliuosius skaičius pirminiais daugikliais? Tai nėra žinoma. Tačiau jeigu RSA būtų įveikiama nustatant privatųjį raktą, tai ir modulį būtų galima efektyviai skaidyti.

140 teorema. *Jeigu žinomi abu RSA kriptosistemos raktai $K_v = \langle n, e\rangle$ ir $K_p = \langle d\rangle$, tai n galima išskaidyti naudojant tikimybinį polinominį algoritmą.*

Įrodymas. Kadangi kriptoanalitikas žino ir e, ir d, jis gali apskaičiuoti $ed-1 = k\varphi(n)$. Kiekvienam skaičiui a, $(a,n) = 1$, teisingas lyginys $a^{ed-1} \equiv 1 \pmod{n}$. Tačiau gali būti, kad parinktajam a lyginys $a^m \equiv 1 \pmod{n}$ teisingas ir su mažesniu laipsnio rodikliu m.

Galime greitai nustatyti, iš kokio dvejeto laipsnio dalijasi $ed-1$, ir surasti tokią išraišką:

$$ed - 1 = 2^s t, \quad (2,t) = 1.$$

Parinkime $a, (a,n) = 1$, ir skaičiuokime:

$$a_0 \equiv a^t \pmod{n}, \quad a_1 \equiv a_0^2 \pmod{n}, \quad \ldots \quad a_i \equiv a_{i-1}^2 \pmod{n}, \ldots$$

Kuris nors iš elementų a_j bus lygus 1. Tarkime, v yra mažiausias indeksas, su kuriuo

$$a_{v-1} \not\equiv 1 \pmod{n}, \quad a_v \equiv 1 \pmod{n}.$$

Kadangi

$$a_v \equiv a^{2^v t} \equiv a_{v-1}^2 \,(\mathrm{mod}\, n), \quad \text{tai} \quad a_v - 1 \equiv (a_{v-1}-1)(a_{v-1}+1) \equiv 0 \,(\mathrm{mod}\, n).$$

Dabar galima bandyti sėkmę: sandauga $(a_{v-1}-1)(a_{v-1}+1)$ dalijasi iš n; jeigu nei vienas iš abiejų daugiklių nesidalytų iš n, tai vienas turėtų dalytis iš p, kitas iš q. Tada Euklido algoritmu surastume:

$$p = (a_{v-1}-1, n), \quad q = (a_{v-1}+1, n).$$

O jeigu abu daugikliai dalytųsi iš n? Tada tektų bandyti laimę su kitu a. Todėl šis algoritmas ir yra tikimybinis.

Taigi organizuojant grupės dalyvių tarpusavio ryšių apsaugą su RSA, reikia pasirūpinti, kad moduliai būtų skirtingi. Jeigu dviejų dalyvių moduliai bus vienodi, tai tas, kuris išmano kriptografiją, galės sužinoti savo kolegos privatųjį raktą, išskaidęs bendrąjį modulį pirminiais daugikliais (jeigu ne jis pats sukūrė savo raktus, tai to skaidinio ir pats nežino).

Bendras kelių vartotojų modulis kelia pavojų ir dėl kitos priežasties. Tegu dviejų ryšio dalyvių RSA viešieji raktai yra $K_{v,A} = \langle n, e_A\rangle, K_{v,B} = \langle n, e_B\rangle, (e_A, e_B) = 1$. Tarkime, kažkas pasiuntė jiems abiem to paties pranešimo šifrus:

$$C_1 \equiv M^{e_A} \,(\mathrm{mod}\, n), \quad C_2 \equiv M^{e_B} \,(\mathrm{mod}\, n).$$

Kriptoanalitikas Zigmas žino $K_{v,A}, K_{v,B}, C_1, C_2$. Atskleisti M jam visai nesunku: suradęs Euklido algoritmu skaičius a, b, su kuriais $ae_A + be_B = 1$ jis lengvai apskaičiuos

$$C_1^a C_2^b \equiv M^{ae_A + be_B} \equiv M \,(\mathrm{mod}\, n).$$

20.4. Pohligo-Hellmano ir Massey-Omura kriptosistemos

Tai dvi įdomios variacijos RSA kriptosistemos tema.

Jeigu kriptosistema yra viešojo rakto, tai nereiškia, kad šifravimui skirtą raktą būtina paskelbti. Jeigu nenorite – neskelbkite. Pavyzdžiui, perduokite RSA šifravimo raktą savo draugams saugiu būdu ir naudokitės RSA kaip paprasta simetrine kriptosistema. Jeigu daug šifruoti nereikia, toks viešojo rakto kriptosistemos naudojimas „ne pagal paskirtį“ visiškai tinkamas. Tačiau jeigu tenka šifruoti didelius duomenų srautus, geriau jau pasitelkti kokią nors įprastinę simetrinę kriptosistemą, nes ji tiesiog daug greitesnė.

O RSA naudojimo būdas nepaskelbiant raktų kriptografijoje netgi turi atskirą vardą – tai Pohligo-Hellmano simetrinė kriptosistema. Tiesa, dažniau taip vadinamas kiek pakeistas kriptosistemos variantas.

Pohligo-Hellmano kriptosistema

Pranešimai ir šifrai – aibės $\mathbb{Z}_p^*$ skaičiai, p – didelis pirminis skaičius.

Šifravimo ir dešifravimo raktai:
$K_e = e,\ K_d = d,\ ed \equiv 1 \pmod{p-1}$ – abu raktus gauna ir A, ir B.

Šifravimas: $C \equiv e(M|K_e) \equiv M^e \pmod{p}$.

Dešifravimas: $M \equiv d(C|K_d) \equiv C^d \pmod{p}$.

Pohligo-Hellmano kriptosistemos modulis p netgi gali būti paskelbtas, galimybių sužinoti e, d kriptoanalitikui toks paskelbimas nesuteiks. Susitarti dėl modulio A ir B gali ir nesaugiu kanalu, pavyzdžiui, telefonu. Skaičiai e ir d jiems turi būti įteikti saugiai. Įdomu, kad galima šia kriptosistema naudotis visiškai neperdavus skaičių e ir d! Žinoma, bent jau dešifravimo algoritmą teks pakeisti, todėl turėsime naują kriptosistemą. Tai Massey-Omura kriptosistema; iš tiesų tai tam tikras kriptografinis protokolas, suteikiantis simetrinei kriptosistemai viešojo rakto kriptosistemos savybes.

Massey-Omura kriptosistema

Pranešimai ir šifrai – aibės $\mathbb{Z}_p^*$ skaičiai, p – didelis pirminis.

Viešasis raktas: pirminis skaičius p.
Privatieji raktai: ryšio dalyviai A, B, naudodami p, sudaro savo privačiuosius raktus iš dviejų komponenčių:

$$K_A = \langle e_A, d_A\rangle, K_B = \langle e_B, d_B\rangle,$$
$$e_A d_A \equiv 1 \pmod{p-1}, e_B d_B \equiv 1 \pmod{p-1}.$$

Šifravimas: A šifruoja $C \equiv e(M|K_A) \equiv M^{e_A} \pmod{p}$ ir siunčia B.
Dešifravimas: B gavusi C, šifruoja $C_1 \equiv e(C|K_B) \equiv C^{e_B} \pmod{p}$ ir siunčia A. A gavęs C_1, dešifruoja $C_2 \equiv d(C_1|K_A) \equiv C_1^{d_A} \pmod{p}$ ir siunčia B. B, gavusi C_2, dešifruoja $d(C_2|K_B) \equiv C_2^{d_B} \equiv M \pmod{p}$.

Įsitikinkime, kad dešifruojama bus tikrai teisingai:

$$\begin{aligned} C_2^{d_B} &\equiv ((C_1)^{d_A})^{d_B} \equiv ((C^{e_B})^{d_A})^{d_B} \equiv (((M^{e_A})^{e_B})^{d_A})^{d_B} \\ &\equiv (M^{e_A d_A})^{e_B d_B} \equiv M^{e_B d_B} \equiv M \pmod{p}. \end{aligned}$$

RSA kriptosistemoje visi ryšio dalyviai savo viešuosius raktus turi paskelbti su skirtingais moduliais. Jeigu dviejų dalyvių moduliai vienodi, jie gali

nustatyti vienas kito privačiuosius raktus. Bendro modulio pavojaus nėra Massey-Omura kriptosistemoje. Įdomi šios kriptosistemos savybė – vienas asmuo gali paskelbti visiems ryšio dalyviams bendrą modulį. Tada bet kurie du dalyviai, naudodamiesi Massey-Omura protokolu, galės užmegzti šifruotą ryšį. Šia ypatybe pasinaudojama kai kuriuose kriptografiniuose protokoluose.

Iš ryšio kanalo, kuriuo naudojasi A ir B, kriptoanalitikas gali gauti šifrus C, C_1, C_2. Tačiau jam iš to mažai naudos, nes, pavyzdžiui, lyginyje

$$C \equiv M^{e_A} \pmod{p}$$

yra netgi du nežinomieji.

20.5. Rabino kriptosistema

Viešojo rakto kriptosistemų kūrėjai naudojasi skaičiavimo uždaviniais, kuriuos sunku spręsti. Vienas tokių uždavinių – kvadratinių lygčių sprendimas žieduose $\mathbb{Z}_n$. Šis uždavinys – Rabino kriptosistemos pagrindas.

Matėme, kad RSA kriptosistemos įveikimo uždavinys nėra sunkesnis už natūraliųjų skaičių skaidymo pirminiais daugikliais uždavinį. Tačiau nežinoma, ar jie ekvivalentūs, t. y. nežinoma, ar suradus efektyvų RSA kriptosistemos šifro dešifravimo be privataus rakto algoritmą, jį būtų galima panaudoti natūraliųjų skaičių skaidymo pirminiais daugikliais uždaviniui.

Šiame skyrelyje išnagrinėsime kriptosistemą, kurios „sulaužymo“ uždavinys sudėtingumo požiūriu ekvivalentus natūraliųjų skaičių skaidymo pirminiais daugikliais uždaviniui.

Raktų parinkimas. Parinkę du didelius pirminius skaičius p, q, iš jų sudarome privatųjį raktą $K_p = \langle p, q\rangle$ ir viešąjį raktą $K_v = \langle n\rangle$, $n = pq$.

Šifravimas. Pranešimų ir šifrų aibė ta pati: $\mathcal{M} = \mathcal{C} = \{0, 1, \ldots, n-1\}$. Šifravimui pasirinkime parametrą $a, 0 \le a < n$ (jis yra viešas). Tada šifras sudaromas taip:

$$c = e(m|K_v) \equiv m(m+a) \pmod{n}.$$

Dešifravimas. Dešifruojant reikia spręsti lyginį

$$x(x+a) \equiv c \pmod{n}.$$

Šį lyginį galima pertvarkyti į paprastesnį:

$$x(x+a) \equiv x^2 + 2\cdot 2^{-1}\cdot x\cdot a + (2^{-1}a)^2, -(2^{-1}a)^2 \equiv c \pmod{n}$$
$$y^2 \equiv d \pmod{n}, \quad y \equiv x + 2^{-1}a \pmod{n}, \quad d \equiv c + (2^{-1}a)^2 \pmod{n}.$$

Tačiau greito algoritmo lyginio $y^2 \equiv d \pmod{n}$ sprendiniui rasti taip pat nėra. Taigi kriptoanalitikas turi spręsti sudėtingą skaičiavimo uždavinį.

Sprendžiant variantų perrinkimo algoritmu, blogiausiu atveju prireiks maždaug n perrankos operacijų.

Privataus rakto savininkas gali vietoj lyginio $y^2 \equiv d \pmod{n}$ spręsti du lyginius $u^2 \equiv d \pmod{p}$ ir $v^2 \equiv d \pmod{q}$. Net ir perrinkimo algoritmu jis greičiau ras sprendinius, nes perrankų skaičius $p+q$ daug mažesnis skaičius už $n = pq$. Iš rastųjų sprendinių jis gali sudaryti lyginio $y^2 \equiv d \pmod{n}$ sprendinį, naudodamasis kiniškąja liekanų teorema. Tačiau ši teorema duoda net keturis sprendinius. Iš jų reikia atsirinkti vieną. Jeigu buvo šifruotas koks nors tekstas, tai mažai tikėtina, kad visi keturi sprendiniai reikš ką nors prasminga. Jeigu vertinti pranešimų pagal prasmę negalime, tada reikia kokio nors susitarimo, kaip atskirti tikrąjį pranešimą nuo „pašalinių“ . Pavyzdžiui, galima susitarti, kad tikrasis pranešimas turi baigtis tam tikra galūne.

Rabino kriptosistema

Pranešimai ir šifrai – aibės $\mathbb{Z}_n$, $n = pq$, skaičiai; čia p, q yra pirminiai, tenkinantys sąlygas $p, q \equiv 3 \pmod{4}$.

Privatusis raktas: $K_p = \langle p, q \rangle$.

Viešasis raktas: $K_v = \langle n \rangle$.

Šifravimas: $C = e(M|K_v) \equiv M^2 \pmod{n}$.
Dešifravimas: randami skaičiai u, v,

$$u \equiv M \pmod{p}, \quad v \equiv M \pmod{q},$$
$$u \equiv C^{(p+1)/4} \pmod{p}, \quad v \equiv C^{(q+1)/4} \pmod{q},$$

naudojantis kiniškąja liekanų teorema, randami keturi skaičiai M_i

$$M_1 \equiv u \pmod{p},\ M_1 \equiv v \pmod{q},$$
$$M_2 \equiv -u \pmod{p},\ M_2 \equiv v \pmod{q},$$
$$M_3 \equiv u \pmod{p},\ M_3 \equiv -v \pmod{q},$$
$$M_4 \equiv -u \pmod{p},\ M_4 \equiv v \pmod{q}$$

ir iš jų atrenkamas pranešimas $M = d(C|K_p)$.

Dešifruojant tenka spręsti lyginius pirminiais moduliais. Tai irgi gali būti nelengva. Tačiau ankstesniame skyrelyje įrodėme teiginį, kad atskiru atveju lyginio sprendinį galima surasti visai lengvai. Prisiminkime tą teoremą.

141 teorema. *Jei $p \equiv 3 \pmod{4}$ ir $u^2 \equiv d \pmod{p}$ turi sprendinį, tai vienas iš sprendinių yra $u_0 = d^{(p+1)/4}$.*

Taigi parinkus Rabino kriptosistemos pirminius iš progresijos $4m+3$, dešifravimui perrankos neprireiks, abu lyginio $u^2 \equiv d \pmod{p}$ sprendiniai gaunami panaudojus vieną kėlimo laipsniu veiksmą. Todėl kriptosistema dažniausiai ir naudojama su tokiais pirminiais.

Įveikti Rabino kriptosistemą reiškia sudaryti efektyvų algoritmą lyginiui $x^2 \equiv d \pmod{n}, n = pq$, spręsti. Jeigu sprendinys egzistuoja, tai jų yra net keturi. Gavę iš algoritmo sprendinius x_1, x_2, x_3, x_4, galime sudaryti dvi jų poras, pavyzdžiui, $x_1 \equiv -x_3 \pmod{n}, x_2 \equiv -x_4 \pmod{n}$. Tada su tam tikrais sveikaisiais skaičiais a, b, u, v bus teisingos lygybės:

$$x_1 \equiv vap + ubq \pmod{n}, \quad x_2 \equiv -vap + ubq \pmod{n},$$

čia p, q – skaičiaus n pirminiai dalikliai. Tuomet $x_1 + x_2 \equiv 0 \pmod{q}$, bet $x_1 + x_2$ nesidalija iš n. Todėl skaičiuodami didžiausiąjį bendrąjį skaičių n ir $x_1 + x_2$, daliklį surasime vieną iš n pirminių daliklių: $(x_1 + x_2, n) = q$.

Taigi Rabino kriptosistema yra viena iš nedaugelio viešojo rakto kriptosistemų, kurių saugumas yra įrodytas, t. y. matematiškai nustatyta, kad kriptosistemos įveikimas tolygus sudėtingo skaičiavimo uždavinio sprendimui.

20.6. Blomo-Goldwasserio tikimybinė kriptosistema

Dar viena kriptosistema, kurioje naudojamasi kvadratiniais sprendiniais. Ji vadinama tikimybine todėl, kad pranešimo šifras priklauso nuo atsitiktinio skaičiaus, kurį pasirenka šifruotojas.

Kriptosistemos raktai parenkami kaip Rabino kriptosistemoje.

Raktų sudarymas. Tegu p, q yra du pirminiai skaičiai su sąlyga $p, q \equiv 3 \pmod{4}$. Tada viešasis raktas $K_v = \langle n \rangle$, o privatusis – $K_p = \langle p, q \rangle$.

Šifravimas. Šifruojami bet kokio ilgio dvejetainės abėcėlės žodžiai, taigi $\mathcal{M} = \{0,1\}^*$. Šifruojamas pranešimas skaidomas į h ilgio fragmentus:

$$M = m_1 m_2 \dots m_t, \quad m_i \in \{0,1\}^h, \quad h = [\log_2 k], \quad k = [\log_2 n].$$

Šifruojama taip. Pasirinkus atsitiktinį skaičių $r \in \mathbb{Z}_n^*$, apskaičiuojamas kvadratas $x_0 \equiv r^2 \pmod{n}$. Kiekvienas fragmentas m_i šifruojamas atskirai:

$$x_i \equiv x_{i-1}^2 \pmod{n}, \quad c_i = e(m_i | K_v) = m_i \oplus y_i,$$

čia y_i yra žodis, sudarytas iš skaičiaus x_i dvejetainės išraiškos h paskutinių (mažiausiai reikšmingų) bitų. Baigus šifruoti visus blokus, dar surandamas skaičius $x_{t+1} \equiv x_t^2 \pmod{n}$ ir siunčiamas šifras

$$C = \langle c_1 c_2 \dots c_t, x_{t+1} \rangle.$$

Pastebėkime, kad visi skaičiai yra tarpusavyje pirminiai su n, taigi ir su p bei q.

Dešifravimas. Akivaizdu, kad, norint dešifruoti C, pakanka rasti x_0. Tada galima skaičiuoti kvadratus $x_i \equiv x_{i-1}^2 \pmod{n}$, surasti y_i ir dešifruoti pranešimo fragmentus: $m_i = c_i \oplus y_i$.

Skaičiui x_i rasti ir reikalingas x_{t+1}. Pirmiausia randami skaičiai $u \equiv x_0 \pmod{p}, v \equiv x_0 \pmod{p}$, o tada, naudojantis kiniškąja liekanų teorema, apskaičiuojamas ir x_0.

Kaip žinant x_{t+1}, surasti $u, u \equiv x_0 \pmod{p}$? Kadangi skaičiai x_i susieti lyginiais $x_i \equiv x_{i-1}^2 \pmod{n}$, tai jiems taip pat bus teisingi ir lyginiai $x_i \equiv x_{i-1}^2 \pmod{p}$. Taigi visi mūsų skaičiai $x_0, x_1, \ldots, x_{t+1}$ nesidalija iš p ir yra kvadratai moduliu p.

Šiek tiek paskaičiuokime:

$$x_{t+1}^{(p+1)/4} \equiv x_t^{(p+1)/2} \equiv x_t (x_t)^{(p-1)/2} \equiv x_t (x_{t-1})^{(p-1)} \equiv x_t \pmod{p}.$$

Taigi keldami x_{t+1} laipsniu $(p+1)/4$, galime rasti $x_t \pmod{p}$. Keldami x_t tuo pačiu laipsniu, galime rasti $x_{t-1} \pmod{p}$ ir t. t. Arba iš karto

$$x_0 \equiv (x_t)^{((p+1)/4)^{(t+1)}} \pmod{p}.$$

Skaičiuodami visų pirma galime surasti $a \equiv ((p+1)/4)^{(t+1)} \pmod{p-1}$, o tada apskaičiuoti $u \equiv x_{t+1}^a \pmod{p}$, t. y. $u \equiv x_0 \pmod{p}$.

Skaičiavimas su q, žinoma, yra analogiškas. Šifruotojas gali pasirinkti parametrą r. Nuo šio skaičiaus priklauso ir šifras. Taigi tas pats pranešimas su tuo pačiu raktu gali būti šifruojamas įvairiai.

20.7. ElGamalio kriptosistema

Ši kriptosistema, tiksliau tariant, įvairūs jos variantai, populiarumu rungiasi netgi su RSA. Jos kūrėjas Taheras ElGamalis yra vienas žymiausių mūsų laikų kriptografijos autoritetų. Kasdieną milijonai žmonių, naršydami po Internetą, naudojasi SSL protokolu, kuriame įdiegtos ElGamalio idėjos.

Nagrinėjome kriptosistemas, kurių saugumas pagrįstas sveikųjų skaičių skaidymo bei kvadratinių lyginių sprendimo uždavinių sudėtingumu. Laikas patyrinėti kriptosistemas, kuriose panaudojamas dar vienas sudėtingas skaičiavimo uždavinys – diskrečiojo logaritmo radimo.

Raktų sudarymas. Tegu p – didelis pirminis skaičius, kad rasti diskretųjį logaritmą moduliu p būtų sunku, g – primityvioji šaknis moduliu p, kitaip tariant – multiplikatyvios grupės $\mathbb{F}_p^*$ generuojantis elementas. Pasirinkime skaičių $a, 0 < a \leq p-1$ ir sudarykime viešąjį raktą K_v pranešimams šifruoti ir privatųjį dešifravimo raktą K_p:

$$K_v = \langle p, g, \beta \rangle, \quad \beta \equiv g^a \pmod{p}, \quad K_p = \langle a \rangle.$$

Šifravimas. Pranešimų aibė $\mathcal{M}$ – visi nenuliniai $\mathbb{F}_p^*$ elementai. Prieš šifruojant parenkamas skaičius $k \in \mathbb{F}_p^*$ ir pranešimo M šifras sudaromas taip:

$$e(M|K_v) = \langle C_1, C_2 \rangle = C, \quad C_1 \equiv g^k \pmod{p}, \quad C_2 \equiv M\beta^k \pmod{p}.$$

Dešifravimas. Šifro $C = \langle C_1, C_2 \rangle$ dešifravimas:

$$d(C|K_p) \equiv C_2(C_1^a)^{-1} \pmod{p}.$$

Nesunku patikrinti, kad dešifravimo procedūra tikrai veikia:

$$C_2(C_1^a)^{-1} \equiv M\beta^k(g^{ka})^{-1} \equiv Mg^{ak-ak} \equiv M \pmod{p}.$$

Jeigu kriptoanalitikui pavyktų iš lyginio $C_1 \equiv g^k \pmod{p}$ surasti $k = \log_g C_1$, tai jis be vargo nustatytų ir M. Tačiau kriptosistema nebūtų įveikta, nes kitą kartą šifruotojas panaudotų kitą k reikšmę. Žinoma, kriptosistema būtų visiškai įveikta, jeigu kriptoanalitikas apskaičiuotų diskretųjį logaritmą $a = \log_g \beta$. Tačiau bent kol kas diskrečiajam logaritmui skaičiuoti greitų būdų nėra.

Ar būtina kiekvienam šifruojamam pranešimui parinkti vis kitą k reikšmę? Panagrinėkime, kas gali atsitikti, jeigu du skirtingus pranešimus M ir M^* šifruotume ElGamalio kriptosistema su tuo pačiu k. Tada jų šifrai būtų

$$C = e(M|K_v) = \langle C_1, C_2 \rangle, \quad C^* = e(M^*|K_v) = \langle C_1, C_2^* \rangle.$$

Kriptoanalitikas kaipmat pastebėjęs, kad pirmosios komponentės sutampa galėtų apskaičiuoti

$$C_2^{-1}C_2^* \equiv (M\beta^k)^{-1}M^*\beta^k \equiv M^{-1}M^* \pmod{p}.$$

Šis lyginys nustato ryšį tarp pranešimų: jeigu vieną iš jų sužinotume, surastume ir kitą. O jeigu su tuo pačiu k būtų užšifruota ne du, bet keli šimtai pranešimų? Kad Z visus juos atskleistų, pakanka sužinoti vieno iš jų turinį.

ElGamalio kriptosistemoje skaičiuojama su kūno $\mathbb{F}_p$ multiplikatyviosios grupės elementais. Vietoj šios grupės galima naudoti bet kokią baigtinę ciklinę grupę, kurioje diskrečiojo logaritmo uždavinį yra sunku spręsti. Tokių grupių yra daug, tačiau yra ir tokių, kuriose diskretųjį algoritmą $\log_g x$ rasti vienas juokas. Štai tokios kriptografijai netinkamos ciklinės grupės pavyzdys: $\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$ su sudėties moduliu n veiksmu. Kokie elementai generuoja šią grupę ir kaip skaičiuojami diskretieji logaritmai?

Bendroji ElGamalio kriptosistemos schema
Pranešimai ir šifrai – baigtinės ciklinės grupės G, kurioje diskrečiojo logaritmo skaičiavimo uždavinys yra sunkus, elementai, t. y. $\mathcal{M} = \mathcal{C} = G$.
Privatusis raktas: $K_p = \langle a \rangle$, a natūralusis skaičius, $a < \lvert G \rvert$.
Viešasis raktas: $K_v = \langle G, g, \beta \rangle$, $\beta = g^a$.
Šifravimas: pranešimui $M \in G$ šifruoti atsitiktinai parenkamas natūralusis skaičius k, skaičiuojama $C_1 = g^k$ ir $C_2 = M\beta^k$. Sudaromas šifras $C = e(M\lvert K_v) = \langle C_1, C_2 \rangle$.
Dešifravimas: $d(C\lvert K_p) = C_2(C_1^a)^{-1} = M$.

Galime ElGamalio kriptosistemą sukonstruoti, pavyzdžiui, panaudoję kokio nors kūno $\mathbb{F}_{p^m}$ multiplikatyviąją grupę.

Šiuo metu labai intensyviai tyrinėjamos galimybės kriptografijai panaudoti baigtines grupes, susijusias su kreivėmis, kurių lygtys yra

$$y^2 = ax^3 + ax + b.$$

Šios kreivės vadinamos elipsinėmis. Naudojant tokias grupes tam pačiam saugumo lygiui kaip kriptosistemose su $\mathbb{F}_p$ garantuoti, pakanka trumpesnių raktų ir reikia mažiau skaičiavimų. Ši savybė yra labai svarbi diegiant kriptosistemas įrenginiuose su ribotais skaičiavimo resursais, pavyzdžiui, autentifikavimui naudojamose kortelėse.

20.8. McElliece'as: dešifravimas yra dekodavimas

Šifravimas yra duomenų pradinės struktūros „sugadinimas“ , dešifravimas – atstatymas. Panašūs veiksmai vyksta perduodant informaciją nepatikimu kanalu. Robertui McElliece'ui kilo mintis, kad perdavimo trūkumas – simbolių iškraipymai – tampa privalumu, kai norime pranešimus apsaugoti nuo tų, kam jie nėra skirti.

Kodavimo teorijos skyriuje aptarėme, kaip perdavimo klaidoms taisyti naudojami tiesiniai kodai. Jeigu klaidų skaičius neviršija pasirinktojo kodo galimybių, tai pagal gautąjį iškraipytą kodo žodį galime atkurti tą, kurį mums siuntė siuntėjas. Nagrinėjome visiems tiesiniams kodams tinkantį lyderių-sindromų metodą. Ar klaidų taisymas šiuo metu vyksta greitai? Kartais iš tikrųjų greitai, pavyzdžiui, dvejetainių Hammingo kodų atveju. Tačiau apskritai dekodavimo algoritmas – ne polinominis. Galima pasakyti dar daugiau – tiesinių kodų dekodavimas, t. y. siųstų žodžių radimas pagal gautuosius, kai iškraipymų kiekis nėra didesnis už to kodo taisomų klaidų skaičių, yra **NP** pilnas uždavinys.

Tačiau yra kodų, kurie specialiais kodų savybes panaudojančiais metodais dekoduojami greitai. Panašiai yra ir su „kuprinės uždaviniu“: nors

tai **NP** pilnas uždavinys, bet sparčiai didėjančių svorių sistemos atveju jis sprendžiamas polinominiu algoritmu.

Šiomis priešingybėmis – dekoduoti yra sunku, bet kartais lengva – ir pasinaudojama McElliece'o kriptosistemoje.

Raktų sudarymas. Tegu $\mathbf{C} \subset \mathbb{F}_q^n$ yra koks nors kodas, kuriam dekoduoti yra greitas algoritmas (pavyzdžiui, BCH kodų šeimos narys). Tegu kodo dimensija yra k, generuojanti matrica G, o taisomų klaidų skaičius t. Sudarykime dar dvi matricas: neišsigimusią $k \times k$ matricą S iš kūno $\mathbb{F}_q$ elementų ir $n \times n$ matricą P, kuri gaunama iš vienetinės, kokia nors tvarka sukeičiant stulpelius. Matrica P bus naudojama žodžio komponentėms sukeisti: jeigu $\mathbf{x} \in \mathbb{F}_q^n$, tai $\mathbf{x}P$ yra žodis, sudarytas iš tų pačių komponenčių, tik išdėstytų kita tvarka.

Dabar jau galime sudaryti raktus:

$$K_p = \langle S, G, P \rangle, \quad K_v = \langle G^* \rangle, \quad G^* = SGP.$$

Matrica G^* yra tam tikro tiesinio $[k, n]$ kodo $\mathbf{C}^* \subset \mathbb{F}_q^n$ generuojanti matrica. Tai „sugadintas“ geras kodas $\mathbf{C}$, taigi dekoduoti iškraipytus jo žodžius turėtų būti sudėtinga.

Šifravimas ir dešifravimas. Šifruojami pranešimai yra k simbolių ilgio žodžiai iš aibės $\mathcal{M} = \mathbb{F}_q^k$; šifravimą sudaro žodžio kodavimas $\mathbf{C}^*$ žodžiu ir t (arba mažiau) simbolių iškraipymas:

$$C = e(M|K_v) = MG^* + \mathbf{e}, \quad \mathbf{e} \in \mathbb{F}_q^n.$$

Klaidų žodis $\mathbf{e}$ parenkamas atsitiktinai ir jo svoris ne didesnis už t, t. y. jame ne daugiau kaip t nenulinių komponenčių.

Gavėjas iškraipytą žodį (šifrą) dešifruoja taip: apskaičiavęs $C_1 = CP^{-1} = (MS)G + \mathbf{e}P^{-1}$, jis gali manyti, kad C_1 yra kanalu siųstas, bet iškraipytas kodo $\mathbf{C}$ žodis $\mathbf{c} = (MS)G$. Svarbu, kad naujojo klaidų žodžio $\mathbf{e}P^{-1}$ svoris yra ne didesnis už t, todėl klaidas galima ištaisyti. Pritaikęs greitąjį dekodavimo algoritmą, jis gali surasti $M' = MS$ ir $M = M'S^{-1} = d(C|K_p)$.

McElliece'o kriptosistema

Sukonstruojamas tiesinis $[n,k]$ kodas $\mathbf{C} \subset \mathbb{F}_q^n$, taisantis t klaidų, kurio žodžiams dekoduoti yra greitai veikiantis algoritmas, G generuojanti kodo matrica. Pranešimų aibė $\mathcal{M} = \mathbb{F}_q^k$.

Privatusis raktas: $K_p = \langle S, G, P\rangle$, S neišsigimusi $k \times k$ matrica iš $\mathbb{F}_q$ elementų, P – n-os eilės matrica, gauta iš vienetinės, kokia nors tvarka perstačius stulpelius.

Viešasis raktas: $K_v = \langle G^*\rangle, G^* = SGP$.

Šifravimas: $C = e(M|K_v) = MG^* + \mathbf{e}, \mathbf{e} \in \mathbb{F}_q^n, w(\mathbf{e}) \leq t$, žodis $\mathbf{e}$ parenkamas atsitiktinai.

Dešifravimas: $C_1 = CP^{-1}$, žodyje C_1 ištaisius klaidas, randama

$$M' = MS, \quad M = d(C|K_p) = M'S^{-1}.$$

Štai ir viskas. Ši kriptosistema – viena pirmųjų viešojo rakto kriptosistemų. Esminių saugumo spragų kol kas niekas neįžvelgė. Tinkamai parinkus kodą šifravimo ir dešifravimo greičiu ji gali pranokti ir RSA. Tačiau yra ir keletas trūkumų, dėl kurių ji nėra populiari. Viena vertus, tinkamam saugumui garantuoti reikalingi dideli kodo parametrai. Pavyzdžiui, pats R. McElliece'as rekomendavo naudoti kodui iš dvejetainės abėcėlės žodžių

$$n = 1024, \quad k = 524, \quad t = 30.$$

Tada viešąjį raktą sudarys net $nk = 30720$ bitų! Kita vertus, šifro dydis gerokai viršija pranešimo. Tiesa, taip yra ir kitose kriptosistemose, pavyzdžiui, ElGamalio.

21 Skaitmeninių parašų schemos

Prieš šifruojant pranešimą viešojo rakto kriptosistema reikia jį užrašyti schemoje numatytu būdu. Kartais šifruojami dvejetainės abėcėlės žodžiai, kartais pranešimą reikia paversti skaičiumi.

Tą pradinį duomenų užrašymo veiksmą tenka atlikti ir prieš sudarant skaitmeninį parašą. Parengtus pasirašymui skaitmeniniu parašu duomenis vadinsime tekstais. Skaitmeninis teksto x parašas – tai tam tikri duomenys, sukurti naudojant tekstą bei privatųjį pasirašymui skirtą raktą. Tikrinant parašą, naudojamas tekstas x, jo parašas y ir viešasis parašui tikrinti skirtas raktas K_v. Parašas priimamas, jeigu šie trys dydžiai tenkina tam tikrą skaitmeninio parašo schemoje numatytą sąlygą.

115 apibrėžimas. *Skaitmeninių parašų schemą sudaro tekstų aibė $\mathcal{M}$, skaitmeninių parašų aibė $\mathcal{P}$, raktų $K = \langle K_v, K_p\rangle$ aibė $\mathcal{K}$ (čia K_p –*

privačioji parašui sudaryti skirta komponentė, K_v – viešoji parašui tikrinti skirta rakto komponentė) ir parašų sudarymo bei tikrinimo algoritmų šeimos

$$\begin{aligned} sig(\cdot|K_p) &: \mathcal{M} \to \mathcal{P}, \\ ver(\cdot|K_v) &: \mathcal{M} \times \mathcal{P} \to \{0,1\}. \end{aligned}$$

Parašo tikrinimo algoritmai turi savybę:

$$ver(x, sig(x|K_p)|K_v) = 1. \tag{103}$$

Jeigu $y \in \mathcal{P}$ pateikiamas kaip teksto $x \in \mathcal{M}$ parašas, tai parašas pripažįstamas galiojančiu, jeigu $ver(x, y|K_v) = 1$, ir pripažįstamas negaliojančiu, jei $ver(x, y|K_v) = 0$.

Taigi teksto $x \in \mathcal{M}$ skaitmeninis parašas yra $y = sig(x|K_p)$. Įprastiniai parašai tikrinami lyginant juos su pavyzdžiu, o skaitmeniniai – atliekant skaičiavimus, kurie parodo, ar x ir y tenkina tam tikrą matematinį sąryšį.

Daugelį viešojo rakto kriptosistemų galima paversti skaitmeninio parašo schemomis. Iš viešojo rakto K_v nustatyti privatųjį K_p praktiškai neįmanoma. Jeigu neįmanoma ir iš privačiojo rakto nustatyti viešąjį, tai tokią kriptosistemą galime paversti skaitmeninio parašo schema tiesiog paskelbdami dešifravimui skirtą raktą K_p ir paslėpdami viešąjį K_v.

Tada teksto parašas bus jo šifras, kurį gali sukurti tik turintis raktą K_v:

$$y = e(x|K_v).$$

Tikrinant parašą, pateikiama pora $\langle x, y\rangle$. Jeigu reikšmė $x' = d(y|K_p)$ sutampa su x, parašas priimamas. Jeigu x yra tekstas, kurį galime vertinti prasmės požiūriu, tai tikrinimui galima pateikti vien tik parašą y. Jeigu jis „prasmingai“ išsišifruoja, tai gautąjį tekstą galime laikyti pasirašytu to asmens, kurio raktą tikrinimui naudojome. Tikimybė, kad gerai išsišifruos be K_v sudarytas parašas y, labai maža.

Tačiau ne visų viešojo rakto kriptosistemų raktai K_v, K_p turi minėtą savybę. Pavyzdžiui, Rabino kriptosistemoje $K_p = \langle p, q\rangle$, o $K_p = \langle n\rangle, n = pq$. Aišku, kad, paskelbus pirminius p, q, slėpti jų sandaugą nebėra jokios prasmės. Tačiau ir tokiu atveju, kai privačiojo rakto negalima paskelbti, yra paprastas būdas paversti kriptosistemą skaitmeninio parašo schema. Galima dešifravimo algoritmo, pritaikyto pranešimui x, rezultatą $y = d(x|K_p)$ laikyti skaitmeniniu parašu. Tada, tikrinant parašą, pakaks įsitikinti, ar $x = e(y|K_v)$.

21.1. RSA skaitmeniniai parašai

Beveik nieko nereikia keisti RSA kriptosistemoje, jeigu norime ją naudoti kaip skaitmeninių parašų schemą. Vis dėlto, diegdami ją praktiškai, tam tikrų keblumų neišvengtume.

Kadangi RSA kriptosistemoje šifruojama ir dešifruojama tuo pačiu algoritmu tik su skirtingais raktais, tai, norint RSA naudoti kaip skaitmeninio parašo schemą, nieko nereikia keisti.

Jeigu A viešasis RSA raktas yra $K_{v,A} = \langle n_A, e_A \rangle$, o privatusis $K_{p,A} = \langle d_A \rangle$, tai pranešimo x parašą A gali sudaryti tiesiog taip:

$$y = sig(x|K_{p,A}) \equiv x^{d_A} \pmod{n_A},$$

ir siųsti B porą $\langle x, y \rangle$ arba tiesiog y. Parašo tikrinimas – dešifravimas su viešuoju raktu:

$$x \equiv y^{e_A} \pmod{n_A}.$$

Tikrinti A parašą ir skaityti pasirašytą tekstą galės visi, kas panorės. Kaip pasiekti, kad tik B galėtų perskaityti A laišką ir, patikrinusi parašą, būtų tikra, kad laišką atsiuntė tikrai A?

Patarkime B irgi susikurti RSA kriptosistemą. Tegu B raktai bus $K_{v,B} = \langle n_B, e_B \rangle$ ir $K_{p,B} = \langle d_B \rangle$. Dabar A, norėdamas siųsti ir šifruotą, ir pasirašytą laišką x, gali elgtis dvejopai: siųsti c_1 arba c_2:

$$c_1 = e(sig(x|d_A)|e_B), \quad c_2 = sig(e(x|e_B)|d_A).$$

Kurį būdą pasirinkti? Abu būdai ne tik nėra lygiaverčiai, bet vienas netgi gali neveikti. Tarkime, pavyzdžiui, $n_A > n_B$. Kad, sudarę parašą $y = sig(x|d_A)$, galėtume jį sėkmingai užšifruoti, turi būti teisinga nelygybė $y < n_B$. Tačiau $n_A > n_B$, todėl gali būti ir $y > n_B$. Tada šifruodami sugadinsime savo laišką. Taigi tokiu atveju reikia pirma šifruoti, o tada pasirašyti, t. y. siųsti c_2. Tačiau šis variantas irgi turi šiokį tokį trūkumą. Perėmęs siunčiamą c_2, kriptoanalitikas Z gali jį dešifruoti A viešuoju raktu, t. y. surasti $c = e(x|e_B)$, ir, jeigu jis irgi yra šios ryšių sistemos dalyvis, pasiųsti jį B savo vardu: $c_3 = sig(c|d_Z)$. Birutė bus kiek suklaidinta.

Tokių keblumų galima išvengti. Kiekvienam dalyviui sukurkime po du komplektus RSA raktų su skirtingais moduliais. Pirmoji raktų pora skirta skaitmeniniams parašams, o antroji – šifravimui. Jeigu parašams sudaryti skirtų raktų moduliai bus mažesni už tam tikrą nustatytą slenksčio reikšmę T, o šifravimui skirtų raktų moduliai didesni už T – išvengsime visų nesusipratimų pirma pasirašydami, o paskui šifruodami. Tada A, norėdamas pasiųsti pasirašytą ir šifruotą laišką B, turėtų siųsti $c_1 = e(sig(x|d_A^{(1)})|e_B^{(2)})$, čia $d_A^{(1)}$ yra A parašams sudaryti skirtas privatusis raktas, o $e_B^{(2)}$ – šifruotiems laiškams rašyti B viešasis raktas.

RSA skaitmeninio parašo schema
Tekstų aibė $\mathbb{Z}_n$, $n = pq$, p, q – du pakankamai dideli pirminiai skaičiai. **Privatusis parašams sudaryti skirtas raktas:** $K_p = \langle d \rangle$, $(d, \varphi(n)) = 1$. **Viešasis parašams tikrinti skirtas raktas:** $K_v = \langle n, d \rangle$, $ed \equiv 1 \pmod{\varphi(n)}$. **Parašo sudarymas:** tekstas $x \in \mathbb{Z}_n$, jo parašas $y \equiv x^d \pmod{n}$. **Parašo tikrinimas:** parašas priimamas, jeigu $y^e \equiv x \pmod{n}$ arba jeigu x neatsiųstas, įvertinus $y^e \pmod{n}$ pagal prasmę.

RSA schema turi multiplikatyvumo savybę: jei $y_1 = sig(x_1|K_{v,A}), y_2 = sig(x_2|K_{v,A})$, tai $y = y_1y_2$ yra pranešimo $x = x_1x_2$ parašas. Taigi Z turi galimybę iš dviejų A pasirašytų pranešimų sudaryti trečiojo pranešimo parašą. To galima išvengti, pavyzdžiui, pasirašinėjant ne pačius pranešimus, bet jų vaizdus, gautus naudojant visiems schemos dalyviams žinomą injekciją $R : \mathcal{M} \to \mathcal{M}_S$, jei tik ši funkcija parinkta taip, kad neturėtų multiplikatyvumo savybės, t. y. $R(x_1x_2) \neq R(x_1) \cdot R(x_2)$.

Tam tikruose kriptografiniuose protokoluose reikia, kad subjektas sukurtų skaitmeninį parašą, nematydamas paties teksto. Tai tarsi pasirašymas ant užklijuoto voko, kuriame įdėta kalkė ir pasirašymui parengtas dokumentas. Kalkė perkelia parašą nuo voko ant paties dokumento. Tokie parašai vadinami aklais parašais. Jų prireikia finansinėje kriptografijoje bei elektroninių rinkimų sistemose. Pavyzdžiui, rinkiminė komisija savo parašu turi patvirtinti, kad skaitmeninis balsavimo biuletenis yra galiojantis, tačiau neturi sužinoti, už ką rinkėjas balsavo.

Sukurti aklą parašą su RSA sistema labai paprasta. Tarkime, B nori, kad A sukurtų galiojantį teksto m RSA parašą, bet nepamatytų paties teksto. Tegu A raktai yra $K_{v,A} = \langle e_A, n_A \rangle$ ir $K_{p,A} = \langle d_A \rangle$. Parinkusi skaičių $r, (r, n) = 1$, B apskaičiuoja

$$x \equiv r^{e_A} m \pmod{n_A}$$

ir nusiunčia A pasirašyti. A pasirašo ir atsiunčia B parašą

$$z = sig(x|K_{p,A}) \equiv x^{d_A} \pmod{n_A}.$$

Dabar B skaičiuoja

$$y \equiv r^{-1}z \equiv r^{-1}x^{d_A} \equiv r^{-1}(r^{e_A}m)^{d_A} \equiv m^{d_A} \pmod{n_A}.$$

Taigi $y = sig(m|K_{p,A})$ yra galiojantis teksto m parašas.

21.2. Rabino skaitmeninis parašas

Pranešimo x Rabino skaitmeninis parašas – tiesiog lyginio $y^2 \equiv x \,(mod\ n)$ sprendinys. Šiek tiek keblumų sudaro tai, kad ne su visais x šį lyginį galima išspręsti.

Rabino kriptosistemoje privatųjį raktą sudaro pirminių skaičių pora $K_p = \langle p, q\rangle$, o viešąjį – jų sandauga $K_v = \langle n\rangle, n = pq$. Šifruojamų pranešimų aibė yra $\mathcal{M} = \mathbb{Z}_n$. Pranešimo x šifravimas – tiesiog kėlimas kvadratu

$$C = e(x|K_v) \equiv x^2 \pmod{n}.$$

Galime Rabino kriptosistemą paversti skaitmeninių parašų schema, kurioje parašo tikrinimas yra toks pat kėlimas kvadratu. Tegu $x \in \mathbb{Z}_n$ yra pranešimas, kurį norime pasirašyti. Naudodamiesi privačiuoju raktu, galime pabandyti išspręsti lyginį

$$y^2 \equiv x \pmod{n}. \tag{104}$$

Jeigu tai pavyko, tai gautą sprendinį y galime siųsti kaip pranešimo x parašą: $y = sig(x|K_p)$. Parašas bus priimtas, jeigu $y^2 \pmod{n}$ ir x sutaps.

Tačiau yra vienas keblumas: (104) lyginys ne su visais x yra išsprendžiamas. Kaip elgtis tuo atveju, kai sprendinio nėra? Žinome, kad nustatyti, ar lyginys išsprendžiamas, naudojantis Legendre'o simboliais galima labai greitai. Jeigu šifruojame kokį nors tekstą, galima bandyti jį kiek pakaitalioti, pavyzdžiui, įterpiant daugiau intervalų, šitaip tikintis, kad galų gale gausime x, kuriam sprendinys egzistuoja. Iš tikrųjų, ilgai vargti tikrai nereikės. Tačiau jeigu tokia empirinė paieška yra nepriimtina, tai reikia sugalvoti kokią nors funkciją $R : \mathbb{Z}_n \to Q_n$, čia $Q_n \subset \mathbb{Z}_n$ yra poaibis tų elementų x, su kuriais (104) lyginys turi sprendinį. Ši funkcija turi būti vieša. Tada pranešimo x skaitmeniniu parašu laikytume lyginio

$$y^2 \equiv R(x) \pmod{n}$$

sprendinį. Tikrinimui tektų pateikti porą $\langle x, y\rangle$, o tikrinant pirmiausia būtų apskaičiuojama reikšmė $R(x)$ ir tikrinama, ar lyginys

$$y^2 \equiv R(x) \pmod{n}$$

yra teisingas. Jeigu teisingas – parašas priimamas.

Kriptografai yra sukonstravę tokių funkcijų R. Tačiau tai padaryti nėra paprasta. Viena iš būtinų sąlygų tokiai funkcijai: lygtį $R(x) = r$ su žinomu r turi būti sunku spręsti. Jeigu būtų lengva, piktavalis Z galėtų siuntinėti prasmingus ir neprasmingus tekstus su mūsų parašais. Iš tikrųjų, parinkime bet kokį y ir raskime lygties $R(x) = y^2$ sprendinį x. Tada $y = sig(x|K_p)$ – parašas suklastotas!

Rabino skaitmeninio parašo schema
Pasirašomi tekstai – skaičiai $x \in \mathbb{Z}_n$, su kuriais lyginys $y^2 \equiv x \pmod{n}$ turi sprendinį; $n = pq$, p, q – du dideli pirminiai skaičiai.
Privatusis parašams sudaryti skirtas raktas: $K_p = \langle p, q \rangle$.
Viešasis parašams tikrinti skirtas raktas: $K_v = \langle n \rangle$.
Parašo sudarymas: teksto $x \in \mathbb{Z}_n$ parašas – lyginio $y^2 \equiv x \pmod{n}$ sprendinys.
Parašo tikrinimas: parašas priimamas, jeigu $y^2 \equiv x \pmod{n}$.

21.3. ElGamalio skaitmeninio parašo schema

Geras ElGamalio schemos idėjas ne vieną kartą panaudojo kitų skaitmeninio parašo sistemų kūrėjai. Schema gera ir tuo, kad ją galima įdiegti naudojant įvairias baigtines ciklines grupes.

Šios schemos saugumas priklauso nuo atitinkamo diskrečiojo logaritmo uždavinio sudėtingumo.

Raktų parinkimas. Tegu p – pirminis skaičius, pakankamai didelis, kad diskrečiojo logaritmo uždavinį multiplikatyvioje $\mathbb{F}_p$ grupėje, t. y. grupėje $\mathbb{F}_p^* = \{1, 2, \ldots, p-1\}$, būtų sunku spręsti. Tegu α yra šią grupę generuojantis elementas (primityvioji vieneto šaknis). Parinkę $a \in \mathbb{Z}_{p-1}$, skaičiuojame $\beta \equiv \alpha^a \pmod{p}$ ir sudarome viešąjį raktą $K_v = \langle p, \alpha, \beta \rangle$. Privatusis raktas sudarytas tik iš vienos komponentės: $K_p = \langle a \rangle$.

Parašų sudarymas. Pranešimai, kuriuos galima pasirašyti – aibės $\mathcal{M} = \mathbb{F}_p^*$ skaičiai, parašų aibė – $\mathcal{P} = \mathbb{F}_p^* \times \mathbb{Z}_{p-1}$. Pranešimo x parašui sudaryti pasirenkame atsitiktinį (slaptą) skaičių $k \in \mathbb{Z}_{p-1}^*$ (taigi $(k, p-1) = 1$) ir skaičiuojame:

$$\gamma \equiv \alpha^k \pmod{p}, \quad \delta \equiv (x - a\gamma)k^{-1} \pmod{(p-1)}.$$

Ši skaičių pora ir yra pranešimo x parašas: $sig(x|K_p) = \langle \gamma, \delta \rangle$.

Matome, kad pranešimo parašas priklauso nuo to, kokį atsitiktinį parametrą k parenkame. Taigi tam pačiam pranešimui gali būti sudaryta daug skirtingų, bet galiojančių parašų.

Parašų tikrinimas. Skaičių pora $y = \langle \gamma, \delta \rangle$ laikoma galiojančiu pranešimo x parašu tada ir tik tada, kai

$$\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod{p}. \tag{105}$$

Iš tikrųjų, jei y sudarytas tinkamai, tai $x \equiv a\gamma + k\delta \pmod{(p-1)}$, taigi

$$\beta^\gamma \gamma^\delta \equiv \alpha^{a\gamma + k\delta} \equiv \alpha^x \pmod{p}.$$

ElGamalio skaitmeninio parašo schema
Pasirašomų tekstų aibė $\mathcal{M} = \mathbb{F}_p^*$, čia p – didelis pirminis skaičius; parašų aibė $\mathcal{P} = \mathbb{F}_p^* \times \mathbb{Z}_{p-1}$.
Privatusis raktas: $K_p = \langle a \rangle, a \in \mathbb{Z}_{p-1}$.
Viešasis raktas: $K_v = \langle p, \alpha, \beta \rangle$, čia α – generuojantis elementas, $\beta \equiv \alpha^a \pmod p$.
Parašo sudarymas: pasirenkamas atsitiktinis $k, (k, p-1) = 1$ ir skaičiuojama: $\gamma \equiv \alpha^k \pmod p$, $\delta \equiv (x - a\gamma)k^{-1} \pmod{(p-1)}$, $\langle \gamma, \delta \rangle = sig(x\|K_p)$.
Parašo tikrinimas: parašas priimamas tada ir tik tada, kai $\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p$.

Kriptoanalitikas, norėdamas pranešimui x sudaryti be privačiojo rakto galiojantį parašą, turi parinkti γ, δ, kad būtų teisingas (105) lyginys. Galima pasirinkti, pavyzdžiui, γ ir ieškoti tinkamo skaičiaus δ. Tačiau

$$\gamma^\delta \equiv \alpha^x \beta^{-\gamma} \pmod p, \quad \delta \equiv \log_\gamma(\alpha^x \beta^{-\gamma}) \pmod{(p-1)},$$

t. y. tenka spręsti diskrečiojo logaritmo uždavinį.

Jeigu pasirenkamas δ ir iš (105) ieškoma γ, tai problema yra dar sudėtingesnė. Jeigu pasirinksime abi parašo komponentes γ, δ ir ieškosime pranešimo x, kuriam $y = \langle \gamma, \delta \rangle$ būtų tinkamas parašas, tai vėl teks ieškoti diskrečiojo logaritmo.

Tačiau vis dėlto galima parinkti (105) lyginį tenkinančius skaičius renkant juos kartu. Vienas būdas yra toks.

Pasirinkime skaičius i, j, tenkinančius sąlygą $0 \le i, j \le p-2, (j, p-1) = 1$. Dabar suskaičiuokime:

$$\gamma \equiv \alpha^i \beta^j \pmod p, \quad \delta \equiv -\gamma j^{-1} \pmod{p-1}, \quad x \equiv -\gamma i j^{-1} \pmod{p-1}.$$

Nesunku įsitikinti, kad tada $y = \langle \gamma, \delta \rangle$ yra galiojantis x parašas, t. y. skaičiai x, γ, δ tenkina (105) lyginį.

Skaičių k, kurį naudojame parašui sudaryti, geriausia, baigus skaičiuoti, iškart ištrinti. Jeigu jis taps žinomas kriptoanalitikui, tai jis, naudodamasis pranešimu x ir jo parašu $\langle \gamma, \delta \rangle$, nesunkiai suras slaptąjį skaičių a:

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

Toks pat pavojus kyla, jeigu du skirtingus pranešimus x_1 ir x_2 pasirašėme naudodami tą patį k: $\langle \gamma, \delta_1 \rangle = sig(x_1\|K_p)$, $\langle \gamma, \delta_2 \rangle = sig(x_2\|K_p)$. Tada iš lyginių

$$\beta^\gamma \gamma^{\delta_1} \equiv \alpha^{x_1} \pmod p, \quad \beta^\gamma \gamma^{\delta_2} \equiv \alpha^{x_2} \pmod p$$

gauname $\alpha^{x_1 - x_2} \equiv \gamma^{\delta_1 - \delta_2} \pmod p$. Kadangi $\gamma \equiv \alpha^k \pmod p$, tai nežinomam parametrui k rasti gauname lyginį $x_1 - x_2 \equiv k(\delta_1 - \delta_2) \pmod{p-1}$.

Tegu $d = (\delta_1 - \delta_2, p-1)$, suprastinę iš d, skaičiui k rasti gauname lyginį $x' \equiv k\delta' \pmod{p'}$, čia $x' = (x_1 - x_2)/d, \delta' = (\delta_1 - \delta_2)/d, p' = (p-1)/d$. Jeigu $d > 1$, tai gautąjį lyginį tenkina ne vienintelis skaičius $k, 1 \le k \le p-1$. Tačiau tikrąjį visada galima atsirinkti naudojantis sąlyga $\gamma \equiv \alpha^k \pmod{p}$.

Sužinojus k, jau aptartu būdu galima surasti ir a.

21.4. Schnorro skaitmeninio parašo schema

Tai variacija ElGamalio skaitmeninio parašo tema. Kriptografinėje literatūroje, o taip pat ir taikymuose galima surasti ir variacijų Schnorro skaitmeninio parašo tema.

Šios schemos saugumas taip pat remiasi diskrečiojo logaritmo uždavinio sudėtingumu.

Raktų sudarymas. Kaip ir ElGamalio schemoje, pirmiausia reikia parinkti didelį pirminį skaičių p. Toliau – surasti kokį nors pakankamai didelį pirminį q, kuris dalija $p-1$. Gali pasitaikyti, kad $p-1$ skaidinys bus sudarytas tik iš mažų pirminių daugiklių. Toks p Schnorro schemos kūrimui netiktų. Geriausia, ko galime tikėtis: $p - 1 = 2q$, čia q yra pirminis. Tokio pavidalo pirminiai skaičiai yra gera žaliava įvairioms kriptografinėms schemoms konstruoti. Jie vadinami saugiais pirminiais. Jų, žinoma, nėra daug, tačiau daug ir nereikia. Štai pirmasis saugių pirminių, didesnių už milijoną, penketukas:

$$1000919,\ 1001003,\ 1001159,\ 1001387,\ 1001447.$$

Radę pakankamai didelį q, suraskime q-osios eilės kūno $\mathbb{F}_p$ elementą α, t. y. tokį, kuriam $\alpha^q \equiv 1 \pmod{p}$ ir $\alpha^j \not\equiv 1 \pmod{p}$, jei $0 < j < q$. Pranešimų aibė $\mathcal{M} = \mathbb{F}_q$. Pasirinkę dar vieną skaičių $e, 0 < e < q$, jau galime sudaryti raktus:

$$K_p = \langle a \rangle, \quad K_v = \langle p, q, g, \beta \rangle, \quad \beta \equiv \alpha^{-a} \pmod{p}.$$

Parašo sudarymas ir tikrinimas. Parinkę skaičių $0 \le r < q-1$, skaičiuojame:

$$\gamma \equiv \alpha^r \pmod{p}, \quad \delta \equiv r + ax \pmod{q}, \quad sig(x|K_p) = \langle \gamma, \delta \rangle.$$

Parašas bus priimtas tada ir tik tada, kai teisingas lyginys

$$\alpha^\delta \beta^x \equiv \gamma \pmod{p}.$$

Schnorro skaitmeninio parašo schema
Pranešimų aibė $\mathcal{M} = \mathbb{F}_q$, čia q yra pirminis skaičiaus $p-1$ daliklis; p irgi yra pirminis.
Privatusis raktas: $K_p = \langle a \rangle, 0 < a < q-1$.
Viešasis raktas: $K_v = \langle p, q, \alpha, \beta \rangle$, $\alpha \in \mathbb{F}_p$ yra q-osios eilės elementas, $\beta \equiv \alpha^{-a} \pmod{p}$.
Parašo sudarymas: pasirašymui skirtas tekstas yra x; parinkus skaičių $0 \leq r < q-1$, skaičiuojama: $\gamma \equiv \alpha^r \pmod{p}, \delta \equiv r + ax \pmod{q}, sig(x\|K_p) = \langle \gamma, \delta \rangle$.
Parašo tikrinimas: pranešimo x parašas $sig(x\|K_p) = \langle \gamma, \delta \rangle$ priimamas tada ir tik tada, kai $\alpha^\delta \beta^x \equiv \gamma \pmod{p}$.

Nesunku įsitikinti, kad pagal taisykles sudarytas parašas bus visada priimtas. Sudarant parašą, reikia apskaičiuoti dydžius γ ir δ. Pirmasis dydis nesusijęs su pranešimu, todėl jį galima apskaičiuoti ir išsaugoti iš anksto. Antrajam dydžiui skaičiuoti reikia visai nedaug veiksmų – vienos daugybos ir sudėties moduliu q. Taigi parašui sudaryti nereikia daug skaičiavimo išteklių. Todėl tokią schemą galima diegti autentifikavimui naudojamose kortelėse su nedidelio galingumo mikroprocesoriais.

21.5. DSA

Šią parašo schemą savo reikšme galima palyginti su DES. DSA (Digital Signature Algorithm) yra pirmoji kriptografijos istorijoje vyriausybiniu lygiu pripažinta skaitmeninių parašų schema.

1991 metais JAV Nacionalinis standartų ir technologijos institutas pasiūlė skaitmeninių parašų schemą, kuri JAV buvo pripažinta skaitmeninių parašų standartu (DSA – Digital Signature Algorithm). Tai pirmoji vyriausybiniu lygiu pripažinta skaitmeninių parašų schema. Teoriniu požiūriu DSA yra ElGamalio skaitmeninio parašo variantas. Vienas iš privalumų, lyginant DSA su ElGamalio schema – DSA parašai yra trumpesni. Ankstesniame skyrelyje matėme, kad ElGamalio skaitmeniniai parašai gali būti net dvigubai ilgesni už patį pranešimą.

Raktų sudarymas. Parinkime du pirminius skaičius p, q, kad q dalytų $p-1$, t. y. $q|p-1$. Kad kriptosistema būtų saugi, diskrečiojo logaritmo uždavinys moduliu p turi būti sunkus. Rekomenduojama naudoti apie 512 bitų skaičių p ir apie 160 bitų skaičių q.

Tegu $\alpha \in \mathbb{F}_p^*$ yra q-osios eilės elementas. Parinkę $a \in \mathbb{F}_q$, apskaičiuojame $\beta \equiv \alpha^a \pmod{p}$, sudarome viešąjį raktą K_v ir slaptąjį K_p:

$$K_v = \langle p, q, \alpha, \beta \rangle, \quad K_p = \langle a \rangle.$$

Parašo sudarymas. Pranešimų, kuriuos bus galima pasirašyti, aibė yra $\mathcal{M} = \mathbb{F}_p^*$, parašų aibė – $\mathcal{P} = \mathbb{F}_q \times \mathbb{F}_q$. Pranešimo $x \in \mathcal{M}$ parašas sudaromas taip. Parinkę atsitiktinį $k \in \mathbb{F}_q^*$, skaičiuojame:

$$\gamma \equiv \alpha^k \pmod{p} \pmod{q}, \quad \delta \equiv (x + a\gamma)k^{-1} \pmod{q}, \quad sig(x|K_p) = \langle \gamma, \delta \rangle.$$

Be to, reikia pasirūpinti, kad būtų patenkinta sąlyga $q \not| \delta$. Jeigu $q|\delta$, reikia pasirinkti naują k ir vėl skaičiuoti parašą.

Parašo tikrinimas. Tegu reikia patikrinti, ar $y = \langle \gamma, \delta \rangle$ yra pranešimo x parašas. Iš pradžių skaičiuojame $e_1 \equiv x\delta^{-1} \pmod{q}, e_2 \equiv \gamma\delta^{-1} \pmod{q}$. Parašas pripažįstamas tada ir tik tada, kai

$$\alpha^{e_1}\beta^{e_2} \pmod{p} \equiv \gamma \pmod{q}. \tag{106}$$

Iš tikrųjų, jei parašas sudarytas teisingai, tai tikrindami gausime

$$\alpha^{e_1}\beta^{e_2} \equiv \alpha^{\delta^{-1}(x+a\gamma)} \pmod{p}.$$

Tačiau iš parašo sudarymo lygybės gauname $\delta^{-1}(x + a\gamma) \equiv k \pmod{q}$, taigi

$$\alpha^{e_1}\beta^{e_2} \equiv \alpha^k \pmod{p}.$$

Tačiau dešinioji pusė moduliu q lygi γ, taigi (106) lygybė teisinga.

Jeigu pasirinktasis pirminis skaičius p užrašomas 512 bitų ilgio žodžiais, o q – 160, tai 512 bitų ilgio teksto skaitmeninis parašas yra sudarytas iš maždaug $2 \times 160 = 320$ bitų.

Pavyzdys. Pasirinksime nedidelius p, q. Skaičius $p = 10007$ yra saugus pirminis, t. y. $p = 2q + 1$, čia $q = 5003$ irgi yra pirminis skaičius. Dabar reikia parinkti q-osios eilės elementą α. Iš pradžių suraskime generuojantį elementą moduliu p. Jų yra daug, pavyzdžiui, $\rho = 51$ yra vienas jų. Taigi galime imti $\alpha \equiv 51^2 \pmod{p}$, t. y. $\alpha = 2601$ yra q eilės elementas. Dabar parinkime a, pavyzdžiui, $a = 300$, tada $\beta \equiv \alpha^a \pmod{p}, \beta = 2774$. Taigi

$$K_v = \langle p, q, \alpha, \beta \rangle = \langle 10007, 5003, 51, 2774 \rangle, \quad K_p = \langle 300 \rangle.$$

Sudarykime pranešimo $x = 1111$ skaitmeninį parašą. Pasirinkime $k = 44$, $k^{-1} \equiv 1933 \pmod{q}$. Tada

$$\gamma \equiv 51^{44} \pmod{p}, \quad \gamma \equiv 8661 \pmod{p}, \quad \gamma \equiv 3658 \pmod{q},$$
$$\delta \equiv (1111 + 300 \cdot 3658) \cdot 1933 \pmod{q}, \quad \delta = 3476.$$

Sąlyga $(\delta, q) = 1$ patenkinta, taigi $sig(1111|K_p) = \langle 3658, 3476 \rangle$.

Tikrindami parašą, pirmiausia surandame $\delta^{-1} \equiv 2051 \pmod{q}$. Dabar skaičiuojame:

$$e_1 \equiv 1111 \cdot 2051 \pmod{q}, \quad e_1 = 2296, \quad e_2 \equiv 3658 \cdot 2051 \pmod{q}, \quad e_2 = 3061.$$

Toliau tikriname: $\alpha^{e_1}\beta^{e_2} \equiv 8661, \quad 8661 \equiv 3858 \pmod{q}$, parašas priimamas.

DSA

Pranešimų aibė $\mathcal{M} = \mathbb{F}_p^*$, parašų aibė – $\mathcal{P} = \mathbb{F}_q \times \mathbb{F}_q$, čia q yra pirminis $p-1$ daliklis.

Privatusis raktas: $K_p = \langle a \rangle,\ 0 < a < q-1$.

Viešasis raktas: $K_v = \langle p, q, \alpha, \beta \rangle$, $\alpha \in \mathbb{F}_p$ yra q-osios eilės elementas, $\beta \equiv \alpha^a \pmod{p}$.

Parašo sudarymas: pranešimui pasirašyti parenkamas skaičius $k \in \mathbb{F}_q^*$ ir skaičiuojama: $sig(x|K_p) = \langle \gamma, \delta \rangle$,

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

Turi būti patenkinta sąlyga $(\delta, q) = 1$.

Parašo tikrinimas: parašas pripažįstamas tada ir tik tada, kai

$$\alpha^{e_1}\beta^{e_2} \pmod{p} \equiv \gamma \pmod{q},$$
$$e_1 \equiv x\delta^{-1} \pmod{q}, e_2 \equiv \gamma\delta^{-1} \pmod{q}.$$

21.6. Nepaneigiami skaitmeniniai parašai

Tai schema, kurioje skaitmeninis B parašas tikrinamas jai „dalyvaujant“. Tačiau asmens dalyvavimas kriptografiniame protokole tiesiogine prasme yra beveidis: nepažiūrėsi į akis ir nenuspręsi, teisingai elgiasi ar sukčiauja. Vadinasi, reikalingi tam tikri matematiniai saugikliai...

Kartais pageidautina, kad, atliekant skaitmeninio parašo tikrinimą, dalyvautų ir parašo autorius. Pavyzdžiui, jeigu asmuo kreipiasi į banką savo sąskaitos tvarkymo klausimu, saugiau, jeigu informacija tikrinama jam dalyvaujant. Tai galima atlikti naudojantis klausimų-atsakymų protokolais (*challenge-and-response protocol*). Tokioje situacijoje svarbu, kad abi pusės (tikrintojas ir parašo autorius) laikytųsi protokolo taisyklių. Tačiau šiame tikrinimo procese abiejų pusių padėtys skirtingos. Tikrintojas remiasi tik vieša informacija, jei jis netinkamai atlieka protokolo veiksmus, autorius gali pareikalauti jį pakeisti ir pasiekti, kad jo parašas bus pripažintas. Parašo autorius atlieka veiksmus naudodamasis tik jam prieinama (slapta) informacija, taigi atlikdamas protokolo veiksmus, ne taip, kaip numatyta, jis gali pasiekti, kad jo parašas bus pripažintas negaliojančiu. Pavyzdžiui, šitaip jis gali neprisiimti kokių nors jo pasirašytų įsipareigojimų.

Chaum ir van Antverpeno nepaneigiamo parašo schemoje yra galimybė nustatyti, kad parašo autorius elgiasi protokolo vykdymo metu netinkamai, taigi bando paneigti savo parašą. Ši schema paskelbta 1989 metais.

Raktų parinkimas. Parinkime p – pakankamai didelį pirminį skaičių, kad diskrečiojo logaritmo uždavinys grupėje $\mathbb{F}_p^*$ būtų sunkus. Tegu q – pirminis skaičiaus $p-1$ daliklis, o $\alpha \in \mathbb{F}_p^*$ – q-osios eilės elementas. Parinkime skaičių $a, 1 \le a \le q-1$, ir suskaičiuokime $\beta \equiv \alpha^a \pmod p$. Dabar jau galime sudaryti raktus:

$$K_p = \langle a \rangle, \quad K_v = \langle p, \alpha, \beta \rangle.$$

Parašų sudarymas. Pranešimus reikia koduoti aibės

$$\mathcal{G} = \{\alpha^m : m = 0, \ldots, q-1\}$$

skaičiais. Parašai – taip pat šios aibės elementai, taigi $\mathcal{M} = \mathcal{P} = \mathcal{G}$. Jei $x \in \mathcal{M}$, tai $sig(x|K_p) \equiv x^a \pmod p$.

Parašo tikrinimas. Tarkime, A turi patikrinti, ar y yra B parašas. Tikrinimo protokolas vykdomas taip:

1. *A parenka atsitiktinius skaičius* $e_1, e_2 \in \mathbb{F}_q^*$, *skaičiuoja* $c \equiv y^{e_1}\beta^{e_2} \pmod p$ *ir siunčia B;*

2. *B skaičiuoja* $d \equiv c^{a^{-1}(\mathrm{mod} q)} \pmod p$ *ir siunčia A;*

3. *parašas priimamas tada ir tik tada, kai* $d \equiv x^{e_1}\alpha^{e_2} \pmod p$.

Jeigu parašas yra tikras, t. y. $y \equiv x^a \pmod p$, tai nesunku įsitikinti, kad jis bus priimtas. Iš tikrųjų,

$$d \equiv c^{a^{-1}(\mathrm{mod} q)} \equiv y^{e_1 a^{-1}}\beta^{e_2 a^{-1}} \equiv x^{e_1 a a^{-1}}\alpha^{e_2 a a^{-1}} \equiv x^{e_1}\alpha^{e_2} \pmod p.$$

Tačiau gali atsitikti, kad bus už teisingą priimtas ir negaliojantis parašas, t. y. toks, kuriam $y \not\equiv x^a \pmod p$. Tačiau tokia galimybė mažai tikėtina.

142 teorema. *Jei* $y \not\equiv x^a \pmod p$, *tai tikimybė, kad* y *bus pripažintas pranešimo* x *parašu, lygi* $1/q$.

Parašo tikrinimo protokolas gali duoti neigiamą atsakymą dviem atvejais: kai parašas iš tikrųjų netikras, t. y. $y \not\equiv x^a \pmod p$, arba B nesilaiko protokolo. Žinoma, ir tikrintojas A gali nesilaikyti protokolo, tačiau jis skaičiuodamas nesinaudoja jokia slapta informacija, taigi jei jis netinkamai elgiasi, gali būti pakeistas kitu.

Šioje parašo schemoje yra galimybė nustatyti, kada parašas yra tikrai netikras, o kada B mėgina (nesvarbu, ar sąmoningai ar dėl skaičiavimo klaidų) to parašo atsisakyti. Jeigu B siekia, kad tikrinimo protokolas duotų neigiamą rezultatą, tai 2-ajame žingsnyje ji pasirenka d „pažvelgusi į lubas“, t. y. bet kokį. Jeigu tikrinimo protokolas duoda neigiamą atsakymą, protokolas dar kartą kartojamas:

1. *jeigu* $d \not\equiv x^{e_1}\alpha^{e_2} \pmod p$ *A parenka atsitiktinius skaičius* $f_1, f_2 \in \mathbb{F}_q^*$, *skaičiuoja* $C \equiv y^{f_1}\beta^{f_2} \pmod p$ *ir siunčia B;*

2. *B skaičiuoja* $D \equiv C^{a^{-1}(mod\ q)} \pmod p$ *(arba vėl pasirenka D „nuo lubų“) ir siunčia A;*

3. *jeigu* $D \equiv x^{f_1}\alpha^{f_2} \pmod p$, *A parašą pripažįsta;*

4. *jeigu* $D \not\equiv x^{f_1}\alpha^{f_2} \pmod p$, *A laiko parašą klastote tada ir tik tada, kai*

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

priešingu atveju A apkaltina B protokolo nesilaikymu arba bandymu suklastoti svetimą parašą.

143 teorema. *Jei* $y \not\equiv x^a \pmod p$, *tačiau ir A, ir B laikosi protokolo, tai*

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

Įrodymas.
Apskaičiuokime dydį $(d\alpha^{-e_2})^{f_1} \pmod p$. Kadangi B laikosi protokolo, tai

$$(d\alpha^{-e_2})^{f_1} \equiv (c^{a^{-1}}\alpha^{-e_2})^{f_1} \equiv (y^{e_1a^{-1}}\beta^{e_2a^{-1}}\alpha^{-e_2})^{f_1} \equiv y^{e_1f_1a^{-1}} \pmod p.$$

Skaičiuodami $(D\alpha^{-f_2})^{e_1} \pmod p$, gautume lygiai tą patį. Taigi lyginys yra teisingas. Galbūt kiek keista, kad skaičiuodami niekur nepanaudojome sąlygos $y \not\equiv x^a \pmod p$. Jeigu ji nebūtų patenkinta, t. y. jeigu y tikrai būtų parašas, tai iki šio lyginio tikrinimo nė neprieitume – parašas būtų pripažintas galiojančiu jau anksčiau.

Ši teorema rodo, kad B negali būti apkaltinta nekaltai. Kokia gi tikimybė B apgauti A – įrodyti, kad jos galiojantis parašas yra klastotė, t. y. atsisakyti savo parašo? Tam ji turėtų atsiųsti ne pagal protokolo taisykles suskaičiuotus d, D, kad lyginys būtų vis dėlto teisingas. Tikimybė, kad tai pavyks yra nedidelė.

144 teorema. *Jei* $y \equiv x^a \pmod p$ *ir B parenka atsitiktinius* d, D, *kad*

$$d \not\equiv x^{e_1}\alpha^{e_2} \pmod p, \quad D \not\equiv x^{f_1}\alpha^{f_2} \pmod p,$$

tai tikimybė, kad

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

lygi $1/q$.

Taigi galimybė atsisakyti savo parašo yra menka.

21.7. Slaptieji kanalai skaitmeninių parašų schemose

Skaitmeninis parašas gali kartais būti ir šifras. Tuoj sužinosite, kaip tai padaryti.

Vienose skaitmeninio parašo schemose teksto parašas apibrėžtas vienareikšmiškai (pavyzdžiui, RSA). Kitose parašas priklauso nuo papildomai parenkamo dydžio. Parašas – tai tam tikri duomenys. Tikrinant parašą, su pranešimu ir tais duomenimis atliekami tam tikri skaičiavimai. Priklausomai nuo jų rezultatų parašas priimamas arba atmetamas. Tačiau tie duomenys gali būti kito pranešimo slaptavietė! Taigi skaitmeniniu parašu galima pasinaudoti kaip slaptu kanalu informacijai perduoti. Tokį slaptą kanalą galima įrengti beveik kiekvienoje skaitmeninio parašo schemoje, kurioje parašas nėra vienareikšmiškai apibrėžtas. Panagrinėkime, kaip tai galima padaryti su ElGamalio skaitmeninių parašų schema.

ElGamalio schemos raktai

$$K_p = \langle a \rangle, \quad K_v = \langle p, \alpha, \beta \rangle, \quad \beta \equiv \alpha^a \pmod{p},$$

čia $\alpha \in \mathbb{F}_p^*$ yra generuojantis elementas. Privatusis raktas sudarytas tik iš vienos komponentės: $K_p = \langle a \rangle$.

Pranešimo x parašas $sig(x|K_p) = \langle \gamma, \delta \rangle$ sudaromas pasirinkus skaičių $k \in \mathbb{Z}_{p-1}^*$ ir apskaičiavus

$$\gamma \equiv \alpha^k \pmod{p}, \quad \delta \equiv (x - a\gamma)k^{-1} \pmod{(p-1)}.$$

Parašas pripažįstamas, jeigu teisingas lyginys

$$\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod{p}. \tag{107}$$

Dabar tarkime, kad A veiksmai yra kontroliuojami ir jis negali pasiųsti B jokio nors menkiausią įtarimą keliančio pranešimo. Tarkime, kad šiek tiek anksčiau A sugebėjo pranešti B savo ElGamalio skaitmeninio parašo raktą $K_p = \langle a \rangle$. Norėdamas perduoti B slaptą pranešimą x, A ketina pasielgti taip: pasirašyti kokį nors nekaltą pranešimą x^* ir paskelbti jį kartu su parašu $y = sig(x^*|K_p) = \langle \gamma, \delta \rangle$. Parašas tenkins tikrinimo sąlygą, todėl nekels įtarimo jokiems kontrolieriams. Tačiau čia ir slypi gudrybė – paraše glūdės slaptasis pranešimas x, kurį B galės atskleisti žinodama $K_p = \langle a \rangle$.

Taigi A ketina sudaryti x^* parašą. Tarkime, $(x, p-1) = 1$ (jeigu taip nėra, galima x šiek tiek pakeisti). Tada parašą galima kurti su $k = x$. Jei

$$\gamma \equiv \alpha^x \pmod{p-1}, \quad \delta \equiv (x^* - a\gamma)x^{-1} \pmod{p-1},$$

tai $sig(x^*|K_p) = \langle \gamma, \delta. \rangle$. Tačiau žinodama pranešimą, parašą ir raktą, B gali surasti jai skirtą žinutę x:

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

Tiesa, B gali kilti šiokių tokių sunkumų, jei δ nėra tarpusavyje pirminis su $p-1$. Tada δ^{-1} neegzistuos. Tačiau tai nedideli sunkumai. Be to, jų išvengti gali padėti ir A, pasirūpindamas, kad būtų $(\delta, p-1) = 1$.

22 Maišos funkcijos

Skaitmeninių parašų schemos, kokias nagrinėjome, kone visiškai netinkamos praktiniam naudojimui! Iš tiesų, kiek reiktų atlikti sudėtingų skaičiavimų, kad pasirašytume, pavyzdžiui, 1Mgb dydžio tekstą! Todėl reikia ieškoti būdo, kaip tas nepraktiškas schemas paversti lengvai pritaikomomis. Išeitis gali būti paprasta: pasirašinėti reikia ne visus tekstus, bet fiksuoto ilgio santraukas. Taigi reikia tų santraukų sudarymo metodų.

Tarkime, tekstai, kuriuos reikia pasirašinėti, gali būti bet kokio ilgio dvejetainės abėcėlės žodžiai, o skaimeninio parašo schema sukuria tik n bitų ilgio žodžių parašus. Todėl reikalinga funkcija

$$h : \{0,1\}^* \to \{0,1\}^n,$$

sudaranti bet kokio pranešimo santrauką. Tokios funkcijos kriptografijoje vadinamos maišos funkcijomis (hash functions, angl.). Jeigu naudojama maišos funkcija, tai pranešimo x parašu laikome jo santraukos parašą $y = sig(h(x)|K_p)$. Gavęs porą $\langle x, y\rangle$, parašo tikrintojas pirmiausia suskaičiuoja $x' = h(x)$, o tada tikrina, ar y yra x' parašas. Parašo sudarymo ir tikrinimo veiksmų sąnaudos nepriklauso nuo pranešimo ilgio. Kadangi pranešimų yra daugiau negu santraukų, tai yra tekstų su vienodomis santraukomis. Tada šių tekstų skaitmeniniai parašai bus vienodi. Maišos funkcijų naudojimas, be abejonės, susilpnina skaitmeninio parašo schemos saugumą. Todėl, konstruojant maišos funkcijas reikia gerai apsvarstyti, ar parašo schema su šiomis funkcijomis išlieka praktiškai saugi.

22.1. Kokios maišos funkcijos yra saugios?

Kad maišos funkcija būtų saugi, ji turi tenkinti kelias sąlygas, kurias paprasta suformuluoti, bet sunku įgyvendinti.

Tarkime, $h : \mathcal{M} \to \mathcal{H}$ yra maišos funkcija, kurianti tekstų santraukas. Kokius reikalavimus ji turėtų tenkinti? Visų pirma, sudaryti santrauką turi būti nesudėtinga, t. y. funkcijos reikšmė $h(x)$ turi būti skaičiuojama efektyviu algoritmu. Tarkime, ir funkcijos pirmavaizdžio elementai gali būti randami greitai, t. y. yra efektyvus algoritmas duotam z, randantis lygties

$$h(x) = z, \quad z \in \mathcal{H},$$

sprendinį $x \in \mathcal{M}$. Tada kriptoanalitikas galėtų lengvai atlikti skaitmeninio parašo schemos su maišos funkcija ataką, visiškai neanalizuodamas parašo sudarymo algoritmų. Iš tiesų, išsaugojęs teksto ir jo parašo porą $\langle x, y\rangle, y = sig(h(x)|K_p)$, jis galėtų ieškoti lygties $h(u) = y$ sprendinių ir, radęs $u = x', x' \neq x$, pateikti porą $\langle x', y\rangle$ kaip naują tekstą su galiojančiu parašu. Kad tokios atakos tikimybė būtų maža, pirmavaizdžio elemento skaičiavimo uždavinys turi būti sudėtingas.

116 apibrėžimas. *Funkciją $h : \mathcal{M} \to \mathcal{H}$ vadinsime vienakrypte, jeigu jos reikšmės skaičiuojamos efektyviu algoritmu, o lygties $h(x) = z$ su žinomu z sprendiniams rasti efektyvaus algoritmo nėra.*

Efektyviais algoritmais paprastai vadiname polinominius algoritmus. Apibrėžimas aiškus, tačiau labai griežtas. Iš tiesų nežinome nei vienos funkcijos, kuri patenkintų jo reikalavimus! Tiesa, yra daug funkcijų, kurių reikšmės yra efektyviai skaičiuojamos, o pirmavaizdžio elementams rasti greitas būdas nežinomas. Šiandien nežinomas, o rytoj gal atsiras! Iš tiesų tos funkcijos, kurias vadiname vienakryptėmis tėra tik kandidatės į jas.

Taigi maišos funkcijos turėtų būti vienakryptės. Suformuluokime dar porą reikalavimų.

117 apibrėžimas. *Tegu $h : \mathcal{M} \to \mathcal{H}$ yra maišos funkcija. Jeigu $x, x^* \in \mathcal{M}, x \neq x^*$, bet $h(x) = h(x^*)$, tai šią porą vadinsime sutapimo pora, arba tiesiog sutapimu (collision, angl.). Funkciją vadinsime atsparia sutapimams, jeigu nėra efektyvaus algoritmo duotajam $x \in \mathcal{M}$, randančio $x^* \in \mathcal{M}, x^* \neq x$, kad $h(x) = h(x^*)$. Funkciją vadinsime labai atsparia sutapimams, jeigu nėra efektyvaus algoritmo, randančio kokią nors sutapimų porą.*

Angliškoje kriptografijos literatūroje funkcijos, atsparios sutapimams, vadinamos *weekly collision free*, o labai atsparios – *strongly collision free* funkcijomis.

Jeigu maišos funkcija nėra labai atspari sutapimams, tai galima tokia ją naudojančios skaitmeninio parašo schemos ataka. Tarkime, A sutinka pasirašyti sutartį s su tam tikrais įsipareigojimais. B norėtų, kad tie įsipareigojimai būtų griežtesni, todėl yra parengusi kitą sutarties variantą s^*. Tačiau A tikrai nesutiks jo pasirašyti. Tada B gali kiek padirbėti su pora s, s^*, nežymiai kaitaliodama tekstus, pavyzdžiui, įterpdama daugiau tuščių tarpų, ir skaičiuodama $h(s)$ ir $h(s^*)$. Jeigu maišos funkcija nėra labai atspari sutapimams, galbūt pavyks gauti $h(s) = h(s^*)$. Tada B gali pateikti pasirašymui tekstą s, o sprendžiant teisinius ginčus, pateikti s^* su galiojančiu A parašu.

Jeigu funkcija yra atspari sutapimams, tai ji nebūtinai yra vienakryptė. Tačiau jeigu ji nėra vienakryptė, tai sutapimui rasti yra efektyvus (tiesa, tikimybinis, t. y. priklausantis nuo sėkmės) būdas. Tokia yra teiginio, kurį tuoj suformuluosime ir įrodysime, prasmė.

145 teorema. *Tegu $h : \mathcal{M} \to \mathcal{H}$ yra maišos funkcija, pranešimų ir santraukų aibės yra baigtinės ir $|\mathcal{M}| \geq 2|\mathcal{H}|$. Jeigu yra efektyvus algoritmas lygties $h(x) = z$ sprendiniui x rasti, tai egzistuoja tikimybinis algoritmas, randantis sutapimą su tikimybe, ne mažesne už $1/2$.*

Įrodymas. Sąlyga $|\mathcal{M}| \geq 2|\mathcal{H}|$ išvertus į kasdienę kalbą reiškia, kad, užrašius pranešimą ir santrauką dvejetainės abėcėlės žodžiais, santrauka bus bent vienu bitu trumpesnė. Tikriausiai sutiksite, kad tai natūralus noras.

Sutapimų radimo algoritmą, kurį dabar nagrinėsime, net algoritmu nelabai tinka vadinti. Jis toks. Pažymėkime efektyviai randamą lygties $h(x) = z$

sprendinį $A(z)$. Sutapimo ieškosime taip: atsitiktinai parinkę $x \in \mathcal{M}$, skaičiuosime $y = h(x)$, o tada $x^* = A(y)$. Jeigu $x^* \neq x$, sutapimas rastas. Jeigu ne – veiksmus galime kartoti. Reikia įsitikinti, kad $P(x^* \neq x) \geq 1/2$.

Pažymėkime $\mathcal{M}_z = \{u : h(u) = z\}$. Skirtingiems z, z' aibės $\mathcal{M}_z, \mathcal{M}_{z'}$ arba nesikerta, arba sutampa. Šių aibių yra tiek, kiek gali būti skirtingų santraukų, pažymėkime šį skaičių t. Taigi

$$\mathcal{M}_i = \mathcal{M}_{z_i}, \quad \mathcal{M}_i \cap \mathcal{M}_j = \emptyset \ (i \neq j), \quad \bigcup_{i=1}^{t} \mathcal{M}_i = \mathcal{M}, \quad \frac{t}{|\mathcal{M}|} \leq \frac{1}{2}.$$

Kadangi, ieškant sutapimo, visi pranešimai pradiniame žingsnyje gali būti parinkti su vienodomis tikimybėmis, tai

$$\begin{aligned} P(x \neq x^*) &= \sum_{x \in \mathcal{M}} P(A(h(x)) \neq x) \frac{1}{|\mathcal{M}|} = \frac{1}{|\mathcal{M}|} \sum_{x \in \mathcal{M}} \frac{|\mathcal{M}_{h(x)}| - 1}{|\mathcal{M}_{h(x)}|} \\ &= \frac{1}{|\mathcal{M}|} \sum_{i=1}^{t} \sum_{x \in \mathcal{M}_i} \frac{|\mathcal{M}_i| - 1}{|\mathcal{M}_i|} = \frac{1}{|\mathcal{M}|} \sum_{i=1}^{t} (|\mathcal{M}_i| - 1) = 1 - \frac{t}{|\mathcal{M}|} \geq \frac{1}{2}. \end{aligned}$$

Jeigu maišos funkcijos reikšmių aibė turės nedaug elementų, tai daug tekstų turės tą pačią santrauką. Tada sutapimas gali pasitaikyti kriptoanalitikui kaip aklai vištai grūdas. Panagrinėkime, kokio ilgio turėtų būti santraukos, kad šis metodas neveiktų. Tiesa, kriptografijoje jis vadinamas ne aklos vištos metodu, bet gimtadienių ataka. Taip yra todėl, kad ataka siejama su žinomu elementarios tikimybių teorijos uždaviniu: kiek mažiausiai žmonių turi susirinkti, kad įvykio, jog atsiras gimusių tą pačią metų dieną, tikimybė būtų didesnė už 1/2? Beveik visi, bandantys atsakyti vadovaudamiesi „sveiku protu“, apsigauna. Atsakymas – 23.

Taigi panagrinėkime tokį sutapimo ieškojimo būdą: atsitiktinai pasirinkime skirtingus $x_1, x_2, \ldots, x_n$ ir apskaičiuokime $z_1 = h(x_1), z_2 = h(x_2), \ldots, z_n = h(x_n)$. Jeigu nors dvi reikšmės sutapo, sutapimas rastas.

Tegu iš viso santraukų yra m, t. y. $|\mathcal{H}| = m$. Padarykime prielaidą, kad toks santaukų skaičiavimas ekvivalentus atsitiktiniam jų pasirinkimui iš visos aibės su grąžinimu. Tada sutapimo tikimybę galime skaičiuoti spręsdami tokį uždavinį: kokia tikimybė, kad, iš urnos su m skirtingų rutulių paeiliui su grąžinimu traukdami n rutulių, nors vieną rutulį ištrauksime pakartotinai?

Lengviau suskaičiuoti tikimybę q, kad visi rutuliai bus skirtingi:

$$\begin{aligned} q &= \frac{m(m-1)\ldots(m-n+1)}{m^n} = \left(1 - \frac{1}{m}\right)\left(1 - \frac{2}{m}\right)\ldots\left(1 - \frac{n-1}{m}\right) \\ &\approx \exp\left\{-\frac{1}{m}\sum_{i=1}^{n-1} i\right\} \approx \exp\left\{-\frac{1}{2m}n^2\right\}. \end{aligned}$$

Jeigu sutapimo radimo tikimybė šiuo metodu yra ϵ, tai $q = 1 - \epsilon$ ir

$$m \approx \frac{-n^2}{2\ln(1-\epsilon)}.$$

Jeigu mūsų kriptoanalitikas gali atlikti sutapimų paiešką su milijardu pasirinktų tekstų, kiek turėtų būti santraukų, kad jo sėkmės tikimybė būtų tik $0,001$? Užrašant santraukas kaip dvejetainius žodžius, koks turėtų būti jų ilgis? Įstatę $n = 10^9, \epsilon = 10^{-3}$ ir kiek paskaičiavę gautume:

$$m \approx 5 \cdot 10^{20}, \quad \text{santraukų ilgis bitais} \geq 69.$$

Tokio ilgio santraukos yra pernelyg trumpos. Praktiškai dabar naudojamos $128, 256$ ir 512 bitų ilgio santraukos.

22.2. Blokiniai šifrai ir maišos funkcijos

Maišos funkcijas naudojame kaip pagalbinius skaitmeninių parašų schemų įrankius. O blokinius šifrus galime naudoti kaip pagalbinius įrankius maišos funkcijoms konstruoti!

Dažnai maišos funkcijos sudaromos taip, kad santrauka sukuriama atliekant iteracinio proceso žingsnius. Tarkime, turime tam tikrą funkciją

$$H : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n. \tag{108}$$

Tada, panaudoję ją, galime sukonstruoti maišos funkciją $h : \{0,1\}^* \to \mathcal{H}, \mathcal{H} = \{0,1\}^n$. Bet kokio ilgio tekstą $x \in \{0,1\}^*$ padalykime n ilgio žodžiais, jeigu reikia, papildydami paskutinįjį žodį iki reikiamo ilgio:

$$x = x_1 x_2 \ldots x_t, \quad x_i \in \{0,1\}^n.$$

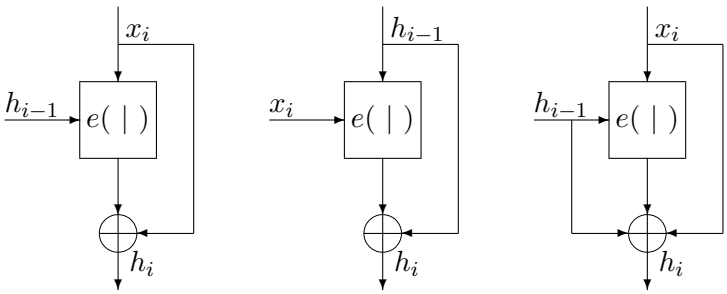
Tegu $h_0 \in \{0,1\}^n$ yra koks nors pradinis fiksuotas žodis. Apibrėžkime

$$h_i = H(h_{i-1}, x_i), \quad i = 1, 2, \ldots, t, \quad h(x) = h_t.$$

Šitaip sudarant santrauką „dalyvaus“ kiekvienas pranešimo simbolis. Tačiau kaip sudaryti (108) funkcijas? Panašios funkcijos kažkur matytos... Iš tiesų – jos naudojamos blokinėse kriptosistemose! Pavyzdžiui, jeigu blokinės kriptosistemos rakto ir bloko ilgiai sutampa, galime imti

$$H(u, v) = e(u|v),$$

t. y. šios funkcijos reikšmė – bloko u šifras panaudojant bloką v kaip raktą. Taigi blokinės kriptosistemos gali būti maišos funkcijų konstrukcijos pagrindas. Jas galima panaudoti ne vien tik minėtu būdu. Panagrinėkime kelis maišos funkcijų sudarymo naudojantis kriptosistemomis metodus.



Matyaso-Meyerio-Oseaso, Davieso-Meyerio ir Miyaguchi-Preneelio metodai maišos funkcijoms konstruoti iš blokinių kriptosistemų.

Brėžiniuose pavaizduotose schemose iteraciniai žingsniai apibrėžiami lygybėmis

$$\begin{aligned} h_i &= e(x_i|h_{i-1}) \oplus x_i \\ h_i &= e(h_{i-1}|x_i) \oplus h_{i-1} \\ h_i &= e(x_i|h_{i-1}) \oplus x_i \oplus h_{i-1}. \end{aligned}$$

Metodų, žinoma, yra ir daugiau. Pavyzdžiui, parinkę dydžiams $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2$ reikšmes iš $\mathbb{F}_2$, galime iteracinį žingsnį apibrėžti taip:

$$h_i = e(\alpha_1 x_i \oplus \alpha_2 h_{i-1}|\beta_1 x_i \oplus \beta_2 h_{i-1}) \oplus \gamma_1 x_i \oplus \gamma_2 h_{i-1}.$$

Ne su visais parametrų rinkiniais tokiu būdu gauname saugias maišos funkcijas.

22.3. SHA-1

Panagrinėjus praktiškai naudojamų maišos funkcijų konstrukcijas, susidaro įspūdis, kad jų kūrėjai veikiau kliovėsi savo intuicija nei kokiais nors objektyviais moksliškais kriterijais. Galite patys tuo įsitikinti.

Skaitmeninių parašų schemų naudojimas be saugių maišos funkcijų yra praktiškai neįmanomas. Nenuostabu, kad skaitmeninių parašų schemų kūrėjai susirūpino ir maišos funkcijų konstravimu. Vienas iš pirmųjų praktiniams taikymams skirtų maišos funkcijų kūrėjų – Ronaldas Rivestas. Jo idėjos panaudotos kuriant MD (Message Digest) maišos funkcijų šeimą.

Kita kriptografijoje gerai žinoma maišos funkcijų šeima – SHA (Secure Hash Algorithm) funkcijos. Funkcija SHA-1 buvo patvirtinta kaip standartinė skaitmeninio parašo DSA schemai. Dabar šios šeimos funkcijos žymimos nurodant, kiek bitų turi su funkcija sudaryta teksto santrauka: SHA-256, SHA-384 ir SHA-512.

Panagrinėkime maišos funkcijos SHA-1, sudarančios 160 bitų ilgio santraukas, veikimo principus.

Ši funkcija atlieka iteracinius 512 bitų ilgio blokų pertvarkius. Todėl pirmiausia tekstas, kurio santrauką reikia sudaryti, pailginamas (jeigu reikia), kad ilgis bitais būtų skaičiaus 512 kartotinis.

Santrauka formuojama penkiuose registruose A,B,C,D,E, kiekviename iš jų saugomi 32 bitai, taigi santraukos ilgis $5 \times 32 = 160$. Prieš pradedant darbą registrai užpildomi pradinėmis standartinėmis reikšmėmis. Pirmasis teksto 512 bitų ilgio blokas padalijamas į 16 žodžių po 32 bitus: $m[0], m[1], \ldots, m[15]$.

Su šiuo bloku bus atliekama 80 operacijų, kiekvienai operacijai reikalinga 32 bitų žodžių pora $K_t, W_t,\ t = 0, 1, \ldots 79$. Jos sudaromos taip:

$$K_t = \begin{cases} \left[2^{30}\sqrt{2}\right], & 0 \le t \le 19, \\ \left[2^{30}\sqrt{3}\right], & 20 \le t \le 39, \\ \left[2^{30}\sqrt{5}\right], & 40 \le t \le 59, \\ \left[2^{30}\sqrt{10}\right], & 60 \le t \le 79, \end{cases}$$

$$W_t = \begin{cases} m[t], & 0 \le t \le 15, \\ (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) \leftarrow\!\!\shortmid 1, & 15 < t < 80, \end{cases}$$

čia ir kitur $\leftarrow\!\!\shortmid r$ reiškia ciklinį postūmį į kairę per r bitų; skaičiai K_t užrašomi 32 bitų žodžiais. Kiekviename iš 80 žingsnių panaudojama sava funkcija

$$f_t : \{0,1\}^{32} \times \{0,1\}^{32} \times \{0,1\}^{32} \to \{0,1\}^{32}.$$

Štai jos visos, apibrėžtos naudojant logines OR, AND ir NOT operacijas su bitais:

$$f_t(x,y,z) = \begin{cases} (x \wedge y) \vee ((\neg x) \wedge z), & 0 \le t \le 19, \\ x \oplus y \oplus z, & 20 \le t \le 39, \\ (x \wedge y) \vee (x \wedge z) \vee (y \wedge z), & 40 \le t \le 59, \\ x \oplus y \oplus z, & 60 \le t \le 79. \end{cases}$$

O dabar jau galima nusakyti funkcijos darbą labai trumpai: kiekviename žingsnyje $t = 0, 1, \ldots, 79$ atliekami tokie veiksmai:

$$\begin{aligned} T &= (A \leftarrow\!\!\shortmid 5) + f_t(B,C,D) + E + W_t + K_t, \\ E &= D, \\ D &= C, \\ C &= B \leftarrow\!\!\shortmid 30, \\ B &= A, \\ A &= T. \end{aligned}$$

Sudėtingiausias veiksmas yra žodžių sudėtis: žodžiai interpretuojami kaip natūralieji skaičiai ir sudedami moduliu 2^{32}. Atlikus visas 80 iteracijų, registruose A, B, C, D, E esančios reikšmės naudojamos atliekant kito 512 bitų ilgio bloko pertvarkius. Tai, kas šiuose registruose atsiranda paskutiniojo bloko pertvarkymo pabaigoje, ir yra maišos funkcijos reikšmė.

Įveikti maišos funkciją reiškia išmokti efektyviai rasti sutapimus. Kartais pasklinda viena kita žinia, kad tokį sutapimą pavyko surasti. Tačiau tokie pavieniai sutapimai praktiniam schemų, kuriose šios maišos funkcijos naudojamos, saugumui grėsmės kol kas nekelia.

22.4. Aritmetinė maišos funkcija

Ar yra maišos funkcija, kurios saugumas kokiu nors būdu gali būti įrodytas? Viena kita atsiranda...

Panagrinėsime Chaumo, van Heijsto ir Pfitzmann sugalvotą maišos funkciją, kurios saugumą galime susieti su diskrečiojo logaritmo uždavinio sudėtingumu.

Tegu p yra saugus pirminis skaičius, t. y. $p = 2q + 1$, čia q irgi yra pirminis skaičius. Tegu $\alpha, \beta \in \mathbb{F}_p$ yra du generuojantys elementai.

Apibrėšime funkciją $h : \{0, 1, \ldots, q-1\} \times \{0, 1, \ldots, q-1\} \to \mathbb{F}_p$ taip:

$$h(x_1, x_2) \equiv \alpha^{x_1} \beta^{x_2} \pmod{p}.$$

Jeigu užrašytume funkcijos argumentus x_1, x_2 ir reikšmę $h(x_1, x_2)$ bitais, matytume, kad reikšmei užrašyti reikia maždaug dvigubai mažiau simbolių. Taigi h galime laikyti maišos funkcija. Tiesa, ji apibrėžta tik riboto ilgio duomenims, tačiau yra metodų, kaip tokias funkcijas pratęsti.

Šios funkcijos saugumo patvirtinimu galime laikyti tokį teiginį:

146 teorema. *Suradę funkcijos $h(x_1, x_2)$ sutapimą, t. y. argumentus $\langle x_1, x_2 \rangle \neq \langle x_1^*, x_2^* \rangle$ su sąlyga $h(x_1, x_2) = h(x_1^*, x_2^*)$, galime rasti $\log_\alpha \beta$.*

Taigi sutapimo radimas tolygus diskrečiojo logaritmo uždavinio atvejo sprendimui. Kadangi šis uždavinys laikomas sudėtingu, tai šia maišos funkcija galima pasikliauti.

Įrodymas. Tegu $\langle x_1, x_2 \rangle \neq \langle x_1^*, x_2^* \rangle$, tačiau

$$\alpha^{x_1} \beta^{x_2} \equiv \alpha^{x_1^*} \beta^{x_2^*} \pmod{p}.$$

Jeigu būtų $x_2 = x_2^*$, tai iš karto gautume, kad $x_1 = x_1^*$. Taigi galime daryti prielaidą, kad $x_2 \neq x_2^*$. Pažymėję $u = \log_\alpha \beta$, gausime

$$\alpha^{x_1 + u x_2} \equiv \alpha^{x_1^* + u x_2^*} \pmod{p}$$

arba

$$u(x_2 - x_2^*) \equiv x_1^* - x_1 \pmod{p-1}. \tag{109}$$

Pažymėkime $d = (x_2 - x_2^*, p-1)$. Jeigu $d = 1$, tai diskretųjį logaritmą gauname iš karto:

$$u \equiv (x_1^* - x_1)(x_2 - x_2^*)^{-1} \pmod{p}.$$

Jeigu $d = 2$, tai (109) turės du sprendinius, iš kurių atsirinksime, kuris mums tinka. Atvejo $d = q$ iš viso negali būti. Išties, kadangi $0 \leq x_2 \leq q-1, 0 \leq x_2^* \leq q-1$, tai $-(q-1) \leq x_2 - x_2^* \leq q-1$ ir šis skirtumas negali dalytis iš q. Teorema įrodyta.

23 Paslapties dalijimo schemos

Penki keliautojai rado didelės vertės brangakmenį. Kadangi kelionės vargai suartina, tai nesutarimų dėl netikėto radinio nekilo, kol jie sugrįžo. O sugrįžę jie nutarė brangakmenį laikyti seife, kol nuspręs, ką su juo veikti. Tačiau kasdienis gyvenimas jau nebe kelionė. Ar nekils kam nors noras pasisavinti brangenybę? Kaip padaryti, kad seifą jie galėtų atverti tik susirinkę visi? Vienas sprendimas – reikia, kad būtų penki užraktai su skirtingais raktais. O jeigu užraktas vienas, be to, atrakinamas surinkus, pavyzdžiui, dešimties skaitmenų kodą? Kaip nors paskirstyti skaitmenis po porą, kad kiekvienas žinotų tik savo? Tai įmanoma, tačiau nelabai saugu. Vienas iš tų penkių gal ir negalės apgauti likusių keturių, tačiau keturi vieną – visai lengvai. Juk susirinkus keturiems tektų patikrinti daugiausiai šimtą variantų, kad seifas atsivertų. Pažiūrėkime, kaip toks paslapties padalijimo uždavinys sprendžiamas kriptografijoje.

23.1. Shamiro paslapties dalijimo schemos su slenksčiais

Kartais paslapčiai atkurti būtina, kad dalyvautų visi, kartais pakanka, kad susirinktų ne mažiau kaip t dalyvių.

Paslapties padalijimo schema negali apsieiti be dalytojo(s). Tarkime, kad dalytojas yra D (Dalia arba Dalius), kuri(s) po paslapties dalijimo išvyksta kur nors labai ilgam. Paslapties dalių gavėjus žymėkime $D_1, D_2, \ldots, D_n$.

Jeigu paslaptis yra pakankamai ilgas dvejetainės abėcėlės žodis $S \in \{0,1\}^m$, o padalyti paslaptį reikia taip, kad tik visi dalyviai susirinkę galėtų ją atkurti, sprendimas gali būti labai paprastas. Dalytojas atsitiktinai parenka $n-1$-ą žodį $S_1, \ldots, S_{n-1} \in \{0,1\}^m$ ir apskaičiuoja

$$S_n = S \oplus S_1 \oplus S_2 \oplus \ldots \oplus S_{n-1}.$$

Dabar kiekvienas dalyvis D_i gauna savo paslapties dalį S_i. Susirinkę visi kartu gali atkurti paslaptį taip:

$$S = S_1 \oplus S_2 \oplus \ldots \oplus S_{m-1} \oplus S_n.$$

O štai Shamiro pasiūlyta schema, naudojanti lyginius.

Tegu m yra didelis natūrinis skaičius, o paslaptis – koks nors skaičius $S \in \mathbb{Z}_m$. Dalytojas D atsitiktinai parenka skaičius $S_1, S_2, \ldots, S_{n-1} \in \mathbb{Z}_m$ ir apskaičiuoja

$$S_n \equiv S - S_1 - S_2 - \ldots - S_{n-1} \pmod{m}.$$

Kiekvienas dalyvis D_i gauna savo paslapties dalį S_i. Jas visas sudėjus gaunama tikroji paslaptis:

$$S \equiv S_1 + S_2 + \ldots + S_n \pmod{m}.$$

Aišku, kad susirinkusiems $n-1$ dalyviui atspėti paslaptį taip pat sunku kaip ir tam vienam, neatėjusiam į susitikimą. Taigi sugadinti paslapties atkūrimo misteriją gali bet kuris neatvykęs į renginį dalyvis. Galima sugadinti ir kitaip – jeigu bent vienas iš dalyvių nurodytų savo paslapties dalį neteisingai, paslaptis būtų nesužinota, nesėkmės kaltininkas irgi nebūtų nustatytas.

Panagrinėkime, kaip bent iš dalies galima išvengti tokių nepatogumų.

118 apibrėžimas. *Tegu S yra paslaptis, o $S_1, S_2, \ldots, S_n$ yra jos dalys, kurias dalytojas D įteikė dalyviams $D_1, D_2, \ldots, D_n$. Šį paslapties padalijimą vadinsime paslapties dalybomis su slenksčiu t ($1 \le t \le n$), jeigu S galima atkurti tik iš ne mažiau kaip t bet kurių paslapties dalių.*

Abiejų nagrinėtų paslapties padalijimo schemų slenkstis lygus dalyvių skaičiui. Jeigu $t = 1$, tai paslapties dalijimo būdas irgi aiškus: $S_i = S$, t. y. kiekvienam įteikiama paslapties kopija. O kaip padalyti paslaptį, kai $1 < t < n$? Panagrinėsime Shamiro pasiūlytą metodą.

Dalytojas D parenka pakankamai didelį pirminį skaičių p, $p > n$, parenka skirtingus skaičius $x_1, x_2, \ldots, x_n \in \mathbb{F}_p$ ir kiekvienam dalyviui D_i įteikia x_i. Šie skaičiai nėra slapti, todėl galima, pavyzdžiui, visą skaičių sąrašą kartu su p paskelbti paslapties dalyboms skirtame tinklalapyje. Paslaptis yra skaičius $S \in \mathbb{F}_p$. Dalytojas atsitiktinai pasirenka skaičius $a_1, a_2, \ldots, a_{t-1}$ ir sudaro daugianarį

$$a(x) = S + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1} \in \mathbb{F}_p[x].$$

Paslapties dalis, kuri įteikiama dalyviui D_i, yra $S_i \equiv a(x_i) \pmod p$.

Paslaptį S, t. y. laisvąjį daugianario narį, galima nustatyti iš bet kurių t paslapties dalių $S_{i_1}, S_{i_2}, \ldots, S_{i_t}$. Iš tikrųjų, susirinkę t dalyvių gali pasinaudodami savo paslapties dalimis sudaryti t lygčių su t nežinomųjų sistemą

$$\begin{aligned}
&S + a_1 x_{i_1} + a_2 x_{i_1}^2 + \ldots + a_{t-1} x_{i_1}^{t-1} = S_{i_1},\\
&S + a_1 x_{i_2} + a_2 x_{i_2}^2 + \ldots + a_{t-1} x_{i_2}^{t-1} = S_{i_2},\\
&\ldots\ldots\ldots\ldots\ldots\ldots\\
&S + a_1 x_{i_t} + a_2 x_{i_t}^2 + \ldots + a_{t-1} x_{i_t}^{t-1} = S_{i_t}
\end{aligned}$$

su nežinomaisiais $S, a_1, a_2, \ldots, a_{t-1}$. Ši sistema moduliu p turi vienintelį sprendinį. Taigi t dalyvių visada gali surasti S.

Paslaptį galima apskaičiuoti ir iš karto, pasinaudojus Lagrange'o interpoliacine formule. Kadangi daugianario $a(x)$ laipsnis yra $t-1$, tai jį galime surasti žinodami t jo reikšmių, įgyjamų su skirtingomis argumento reikšmėmis. Tegu $v_i = a(u_i), i = 1, 2, \ldots, t$, čia $u_i \in \mathbb{F}_p$ yra skirtingi elementai. Tada

$$a(x) = \sum_{i=1}^{t} v_i \prod_{\substack{1 \le j \le t \\ j \ne i}} \frac{x - u_j}{u_i - u_j} \pmod p.$$

Skaičiuojant dalybą reikia suprasti, žinoma, kaip daugybą iš daliklių atvirkštinių elementų kūne $\mathbb{F}_p$. Jeigu įstatysime $x = 0$, gausime laisvąjį narį, t. y. paslaptį, kurią reikia atskleisti:

$$S \equiv \sum_{i=1}^{t} v_i \prod_{\substack{1 \le j \le t \\ j \ne i}} \frac{u_j}{u_j - u_i} \pmod{p}.$$

Taigi susirinkę t dalyvių $D_{i_1}, D_{i_2}, \ldots, D_{i_t}$ gali įstatyti į šią lygybę $v_i = S_{j_i}, u_i = x_{j_i}$ ir paslaptis bus sužinota.

Shamiro paslapties dalijimo schema su slenksčiu t

Paslapties dalijimas: D – dalytojas, $D_1, D_2, \ldots, D_n$ – dalyviai. D parenka pirminį p, skirtingus $x_1, x_2, \ldots, x_n \in \mathbb{F}_p$ ir D_i įteikia skaičių x_i. Pirminis p yra viešas, paslaptis – skaičius $S \in \mathbb{F}_p$.
D parenka skaičius $a_1, a_2, \ldots, a_{t-1}$, sudaro daugianarį

$$a(x) = S + a_1 x + a_2 x^2 + \ldots + a_{t-1} x^{t-1}$$

ir, apskaičiavęs paslapties dalis $S_i \equiv a(x_i) \pmod{p}$, įteikia jas D_i.

Paslapties atkūrimas: susirinkę t dalyvių $D_{i_1}, D_{i_2}, \ldots, D_{i_t}$ paslaptį gali atkurti taip:

$$S \equiv \sum_{i=1}^{t} S_{j_i} \prod_{\substack{1 \le k \le t \\ k \ne i}} \frac{x_{j_k}}{x_{j_k} - x_{j_i}} \pmod{p}.$$

Pastebėkime, kad Shamiro paslapties padalijimo schemą galime sukurti imdami vietoj $\mathbb{F}_p$ bet kokį baigtinį kūną $\mathbb{F}_q$.

Jeigu bent vienas iš susirinkusių dalyvių nurodys neteisingą savo paslapties dalį, paslaptis bus neatkurta. Tarkime, į paslapties atkūrimo „renginį" susirinko dalyviai $D_1, D_2, \ldots, D_{t+r}$ ($r \ge 2$), taigi dalyvių yra šiek tiek daugiau, negu iš tikrųjų reikia. Samprotavimai, žinoma, nepasikeistų, jeigu vietoje šios grupės susirinktų kita tokio pat dydžio dalyvių grupė.

Iš šių dalyvių paslapčių sudarykime žodį $\mathbf{c} = \langle S_1, S_2, \ldots, S_{t+r} \rangle$. Prisiminę, kaip dalytojas sukūrė šias paslapties dalis, galime užrašyti:

$$\mathbf{c} = S\langle 1, 1, \ldots, 1 \rangle + a_1 \langle x_1, x_2, \ldots, x_{t+r} \rangle + \ldots + a_{t-1} \langle x_1^{t-1}, x_2^{t-1}, \ldots, x_{t+r}^{t-1} \rangle.$$

Prisiminę kodavimo teoriją, galime padaryti išvadą:

$\mathbf{c}$ yra Reedo-Solomono kodo su abėcėle $\mathbb{F}_p$ žodis!

Šio kodo parametrai yra $[t+r, t, r+1]$, taigi kodas gali ištaisyti $v = [r/2]$ klaidų. Jeigu ne visi dalyviai nurodys teisingas savo paslapčių dalis, tai gausime ne žodį $\mathbf{c}$, bet iškraipytą žodį $\mathbf{d}$. Tačiau jeigu nesąžiningų dalyvių

nėra daugiau kaip v, tai, pritaikę klaidų taisymo algoritmą, ne tik nustatysime tuos, kurie melavo, bet ir sužinosime, kokias savo paslapčių dalis jie turėjo nurodyti!

23.2. Dar dvi schemos

Vienoje schemoje naudojamasi algebra, o kitoje – skaičių teorija. Jeigu tiksliau – kinų liekanų teorema.

Galima tarti, kad, atkuriant paslaptį, padalytą pagal Shamiro schemą su slenksčiu, sprendžiama t lygčių sistema su t nežinomųjų. Kiekvienas dalyvis į šią sistemą „atneša" savo lygtį. Taigi galima įsivaizduoti, kad, dalijant paslaptį, kiekvienas dalyvis gauna duomenis savo lygčiai – paslapties atkūrimo sistemos daliai – sudaryti.

Panagrinėkime Blakely paslapties padalijimo schemą, kurioje dalyviai gauna jau dalytojo sudarytas lygtis. Tegu p yra pakankamai didelis pirminis skaičius, paslaptis – skaičius $S < p$. Dalytojas atsitiktinai parenka skaičius $s_2, s_3, \ldots, s_t$, sudaro vektorių

$$\mathbf{s} = \langle s_1, s_2, \ldots, s_t \rangle, \quad s_1 = S,$$

ir pradeda dalyti paslaptį. Dalyvio D_j paslapties dalį jis sukuria taip: parenka atsitiktinius elementus $a_1^{(j)}, a_2^{(j)}, \ldots, a_{t-1}^{(j)}$, apskaičiuoja

$$c_j \equiv s_t - \sum_{i=1}^{t-1} a_i^{(j)} s_i \pmod{p}$$

ir įteikia dalyviui D_j lygtį

$$x_t \equiv c_j + \sum_{i=1}^{t-1} a_i^{(j)} x_i \pmod{p}.$$

Susirinkę t dalyvių gali sudaryti t lygčių su t nežinomųjų sistemą ir ją išsprendę rasti paslaptį $m_1 = S$. Tiesa, kad tai jiems pavyktų, t. y. kad sprendinys būtų vienintelis, sistemos koeficientų matrica turi būti neišsigimusi. Jeigu dalytojas nesinaudojo kokiu nors specialiu koeficientų parinkimo būdu, kartais taip gali ir nebūti. Shamiro paslapties padalijimo schema iš tiesų yra atskiras šios sistemos atvejis su garantija, kad bet kuri t lygčių sistema turės vienintelį sprendinį.

O dabar panagrinėkime paslapties padalijimo schemą, kurioje panaudojama senoji geroji kinų liekanų teorema. Metodą vadinsime autorių vardais: Asmutho-Bloomo paslapties padalijimo schema su slenksčiu.

Tegu n yra dalyvių skaičius, o t – slenksčio parametras. Dalytojas parenka $n+1$-ą skaičių $p < p_1 < \ldots < p_n$; p_i turi būti tarpusavyje pirminiai (kad būtų pirminiai nebūtina), be to, reikia, kad būtų patenkinta sąlyga:

$$p_1 p_2 \ldots p_t > p p_n p_{n-1} \ldots p_{n-t+2}. \tag{110}$$

Taigi t mažiausiųjų skaičių p_j sandauga turi būti didesnė už $t-1$ didžiausiųjų sandaugą. Visi šie skaičiai yra vieši.

Tegu skaičius S, $S < p$, yra paslaptis, kurią reikia padalyti. Pažymėkime $N = p_1 p_2 \dots p_t$. Kadangi N yra mažiausiųjų t sekos $p_1, p_2, \dots, p_n$ skaičių sandauga, tai, parinkę bet kuriuos kitus t skaičių, tikrai gausime daugiau.

Dabar parinkime bet kokį (geriau didelį) skaičių r, tenkinantį sąlygą $0 < r < N/p - 1$, ir sudarykime skaičių

$$S^* = S + rp, \quad S^* < S + N - p < N.$$

Taigi S^* yra mažesnis už bet kurių t skaičių $p_{i_1}, p_{i_2}, \dots, p_{i_t}$ sandaugą.

Paslapčiai atskleisti pakanka sužinoti S^*, nes $S \equiv S^* \pmod p$. Dalyvio D_j paslapties dalis sudaroma taip: $S_j \equiv S^* \pmod{p_j}$. Taigi kiekvienas dalyvis žino paslapties dalybos iš savo modulio liekaną.

Susirinkę dalyviai $D_{i_1}, D_{i_2}, \dots, D_{i_t}$ gali pasinaudoti kinų liekanų teorema ir surasti S^* dalybos iš $p_{i_1} p_{i_2} \dots p_{i_t}$ liekaną; tačiau S^* yra mažesnis už šią sandaugą skaičius, tai liekana ir bus šis skaičius.

Galbūt kilo klausimas, o kam reikia (110) sąlygos? Atsakymas: kad paslapties negalėtų atskleisti mažesnė nei t dalyvių grupė!

Asmutho-Bloomo paslapties padalijimo schema su slenksčiu t

Paslapties dalijimas: dalytojas D parenka skaičius

$$p < p_1 < p_2 < \ldots < p_n, \ (p_i, p_j) = 1, \ i \neq j,$$

$$p_1 p_2 \dots p_t > p p_n p_{n-1} \dots p_{n-t+2}.$$

Paslaptis – skaičius $S, S < p$. Dalytojas parenka $r, r < N/p - 1$, ir apskaičiuoja $S^* = S + rp$. Dalyviui D_i paskiriamas skaičius p_i ir saugiu kanalu perduodama paslapties dalis $S_i \equiv S^* \pmod p$.

Paslapties atkūrimas: t dalyvių, naudodamiesi kiniškąja liekanų teorema išsprendžia lyginių sistemą $x \equiv S_{i_j} \pmod{p_{i_j}}$ $(j = 1, 2, \dots, t)$ ir randa sprendinį $x = S^*$. Paslaptis $S \equiv S^* \pmod p$.

23.3. Leidimų struktūros

Kaip padalyti seifui atidaryti naudojamą slaptažodį, kad bet kuri darbuotojų pora galėtų jį atidaryti, išskyrus du labai gerus draugus?

Nagrinėtose paslapties padalijimo schemose visi dalyviai turi „vienodą svorį", t. y. nei vieno iš jų paslapties dalis nėra svarbesnė už kitų. Nesudėtinga paslaptį padalyti taip, kad, dalyvaujant svarbesniems dalyviams, pakaktų mažiau dalyvių nei numatyta. Pavyzdžiui, kai kuriems dalyviams galime duoti po daugiau paslapties dalių. Tačiau ir toks paslapties padalijimas kartais gali būti nepatikimas.

Tarkime, nuspręsta paslaptį padalyti n dalyviams, kad ją galėtų atkurti ne mažiau kaip trys, t. y. slenksčio parametras $t = 3$. Tačiau grupėje yra du svarbūs asmenys A ir B, kuriems mes norėtume suteikti daugiau galių. Galime jiems duoti po dvi paslapties dalis. Tada paslapčiai atverti pakaks A (arba B) ir dar vieno dalyvio. Tačiau A ir B dviese taip pat gali atverti paslaptį! O jeigu Algis ir Birutė – brolis ir sesuo?

Panagrinėkime paslapties padalijimo metodus, labiau atsižvelgiančius į dalyvių tarpusavio santykius.

119 apibrėžimas. *Tegu* $\mathbf{D} = \{D_1, D_2, \ldots, D_n\}$ *yra dalyvių aibė. Jos poaibių sistemą* $\mathcal{G}$ *vadinsime leidimų struktūra, jeigu kiekvienos aibės* $A \subset \mathcal{G}$ *viršaibis* B *(t. y. aibė, tenkinanti sąlygą* $A \subset B$*) taip pat įeina į* $\mathcal{G}$.

Akivaizdu, kad norint apibrėžti leidimų struktūrą $\mathcal{G}$, pakanka nurodyti tik tokią jos posistemę $\Gamma \subset \mathcal{G}$, kad su kiekviena $A \in \Gamma$ jos poaibiai nepriklausytų Γ. Tokią posistemę vadinsime leidimų struktūros branduoliu. Pavyzdžiui, visų poaibių, kurie turi lygiai po t elementų, sistema yra leidimų struktūros branduolys.

120 apibrėžimas. *Sakysime, kad paslapties padalijimo schema realizuoja leidimų struktūrą* $\mathcal{G}$*, jeigu paslaptį iš savo dalių gali atkurti tik tokie dalyviai, iš kurių sudarytas poaibis priklauso* $\mathcal{G}$.

Nesunku įsitikinti, kad norint realizuoti leidimų struktūrą $\mathcal{G}$ reikia paslaptį padalyti taip, kad ją galėtų atkurti poaibių iš branduolio Γ dalyviai, tačiau negalėtų atkurti tokie dalyviai, kurie nesudaro jokio B iš Γ viršaibio.

Bet kokią leidimų struktūrą galima realizuoti! Panagrinėkime labai paprastą metodą, kurį pasiūlė Benalohas ir Leichteris. Jų idėją suprasime panagrinėję pavyzdį.

Tegu $\mathbf{D} = \{D_1, D_2, D_3, D_4, D_5\}$ yra dalyvių aibė, o

$$G_1 = \{D_1, D_4, D_5\},\ G_2 = \{D_1, D_2, D_4\},\ G_3 = \{D_3, D_4\}$$

yra leidimų struktūros branduolys. Tegu paslaptis yra skaičius S. Pasinaudosime Shamiro paslapties padalijimo schema, kurioje paslapčiai atkurti reikia visų dalyvavimo.

Tegu n yra pakankamai didelis skaičius, $S < n$. Grupės G_1 dalyviams įteikime skaičius

$$S_{11},\ S_{14},\ S_{15} \equiv S - S_{11} - S_{14} \pmod{n},$$

čia S_{11}, S_{14} yra atsitiktinai parinkti skaičiai. Sudėję savo paslapčių dalis, šios grupės dalyviai visada galės atkurti paslaptį. Grupės G_2 dalyviams įteikime

$$S_{21},\ S_{22},\ S_{24} \equiv S - S_{21} - S_{22} \pmod{n},$$

o paskutiniosios grupės dalyviams

$$S_{33},\ S_{34} \equiv S - S_{33} \pmod{n}.$$

Paslaptis padalyta, leidimų struktūra realizuota. Kiekvienas dalyvis gavo tiek paslapties dalių, keliose branduolio grupėse jis dalyvauja: trečiasis tik vieną, o ketvirtasis tris. Taigi kuo asmuo „svarbesnis", t. y. į kuo daugiau grupių jis įtrauktas, tuo didesnis jo raktų ryšulys.

Galima pasistengti sudaryti leidimų struktūrą realizuojančią paslapties padalijimo schemą, kurioje kiekvienas dalyvis gauna po tokio paties dydžio paslapties dalį, tinkančią paslapčiai atkurti visose grupėse. Tačiau tikėtis, kad ją bus paprasta sudaryti, neverta.

Panagrinėkime Brickelio paslapties padalijimo schemą. Tegu dalyvių aibė yra $\mathbf{D} = \{D_1, D_2, \ldots, D_n\}$, o Γ – leidimų struktūros, kurią reikia realizuoti, branduolys. Tegu paslaptis vėl yra skaičius S. Dalytojas turi parinkti pakankamai didelį pirminį skaičių p, natūrinį skaičių d ir sukonstruoti funkciją $\delta : \mathbf{D} \to \mathbb{F}_p^d$, tenkinančią tokią sąlygą:

$$\langle 1, 0, \ldots, 0\rangle \in \mathcal{L}(\delta(D_{i_1}), \delta(D_{i_2}), \ldots, \delta(D_{i_r}))$$

tada ir tik tada, kai dalyvių aibė $\{D_{i_1}, D_{i_2}, \ldots, D_{i_r}\}$ yra kokios nors aibės iš branduolio viršaibis. Ši funkcija gali būti vieša. Čia $\mathcal{L}(\ldots)$ žymi tiesinį žodžių sistemos apvalką.

Šios funkcijos konstravimas yra sunkiausia schemos dalis. Bandymų kelias tikriausiai geriausias būdas tokiai funkcijai sudaryti.

Sudarius funkciją, paslaptis padalijama labai paprastai. Dalytojas, parinkęs $a_2, a_3, \ldots, a_d \in \mathbb{F}_p$ atsitiktinai, sudaro dar vieną vektorių

$$\mathbf{a} = \langle a_1, a_2, \ldots, a_d\rangle, \quad a_1 = S,$$

ir apskaičiuoja paslapties dalis $S_i \equiv \mathbf{a} \cdot \delta(D_i) \pmod p$. Ženklas $\cdot$ čia žymi skaliarinę vektorių sandaugą.

Dabar, tarkime, susirinko dalyviai, kurių sudaroma aibė G priklauso leidimų struktūros branduoliui arba yra jo viršaibis. Tada, išsprendę tiesinių lygčių sistemą, jie gali surasti skaičius c_i, kad būtų

$$\langle 1, 0, \ldots, 0\rangle = \sum_{D_i \in G} c_i \delta(D_i).$$

Jeigu padauginsime skaliariškai šią lygybę iš $\mathbf{a}$, gausime

$$\langle S, 0, \ldots, 0\rangle \equiv \sum_{D_i \in G} c_i(\mathbf{a} \cdot \delta(D_i)) \equiv \sum_{D_i \in G} c_i S_i \pmod p.$$

Tačiau dešiniąją lyginio pusę dalyviai gali apskaičiuoti! Taigi jie gali atkurti ir paslaptį.

Shamiro paslapties padalijimo schema su slenksčiu iš tikrųjų yra atskiras šios schemos atvejis. Funkcija $\delta : \mathbf{D} \to \mathbb{F}_p^t$ apibrėžiama taip:

$$\delta(D_i) = \langle 1, x_i, x_i^2, \ldots, x_i^{t-1}\rangle.$$

24 Kriptografiniai protokolai

Kriptografinės priemonės – kriptosistemos, skaitmeniniai parašai, maišos funkcijos ir kitos schemos – skirtos įvairiems duomenų apsaugos uždaviniams spręsti. Šie uždaviniai kyla labai konkrečiose aplinkose. Mums reikia apsaugoti ne duomenis apskritai, bet, pavyzdžiui, duomenis, kurie perduodami kompiuteriniais tinklais arba įrašomi į duomenų bazę. Taigi kriptografinės apsaugos priemonės tampa bendros informacinės sistemos dalimi, kurios efektyvumą lemia ne vienintelio asmens veiksmai. Todėl būtinos taisyklės, kurios numato, kokius veiksmus ir kokia tvarka turi atlikti asmenys, kad naudojama kriptografinė priemonė iš tikrųjų užtikrintų tokią duomenų apsaugą, kokią ji galėtų suteikti. Ta taisyklių visuma, numatanti asmenų (arba juos atstovaujančių kompiuterių ar programų), naudojančių kriptografinę schemą, veiksmų seką, ir vadinama kriptografiniu protokolu. Su kriptografiniais protokolais jau susidūrėme, pavyzdžiui, nagrinėdami nepaneigiamo parašo schemą.

24.1. Raktų paskirstymas

Norėtume susitarti su draugu dėl simetrinės kriptosistemos rakto, bet mūsų ryšio kanalas kontroliuojamas. Išeitis yra!

Viešojo rakto kriptosistemoms saugaus kanalo raktams perduoti nereikia. Tačiau šifravimo ir dešifravimo veiksmai naudojant viešojo rakto kriptosistemas atliekami daug lėčiau negu simetrinėse kriptosistemose. Galbūt galima suderinti abiejų kriptosistemų privalumus? Pavyzdžiui, naudojantis viešojo rakto kriptosistema, galima perduoti simetrinės sistemos raktus, o kai raktai perduoti – šifruoti ir dešifruoti greičiau veikiančia simetrine sistema.

Pirmąjį būdą simetrinės kriptosistemos raktams perduoti nesaugiu kanalu pasiūlė Diffie ir Hellmanas. Panagrinėkime jų protokolą.

A ir B nori sudaryti simetrinės kriptosistemos raktą, tačiau negali naudotis saugiu kanalu. Pirmiausia jie turi susitarti dėl pakankamai didelio pirminio skaičiaus p ir generuojančio elemento $g \in \mathbb{F}_p$. Tai jie gali padaryti nesaugiais kanalais. Tada kiekvienas pasirenka po skaičių: tarkime, A pasirinko skaičių $x_A, 0 < x_A < p-1$, o B – skaičių $x_B, 0 < x_B < p-1$. Dabar A skaičiuoja $X \equiv g^{x_A} \pmod p$ ir siunčia B, o B savo ruožtu – skaičiuoja $Y \equiv g^{x_B} \pmod p$ ir siunčia A. Tiek A, tiek B gali surasti tą patį skaičių ir naudoti jį kaip simetrinės kriptosistemos raktą:

$$K \equiv Y^{x_A} \equiv g^{x_A x_B} \pmod p, \quad K \equiv X^{x_B} \equiv g^{x_A x_B} \pmod p.$$

Diffie-Hellmano rakto nustatymo protokolas

Pasirengimas: A ir B viešu kanalu susitaria dėl pirminio skaičiaus p ir generuojančio elemento g. A pasirenka skaičių $x_A, 0 < x_A < p-1$, o B – skaičių $x_B, 0 < x_B < p-1$.
Raktų nustatymas:

1. A skaičiuoja $X \equiv g^{x_A} \pmod{p}$ ir siunčia B; B skaičiuoja $Y \equiv g^{x_B} \pmod{p}$ ir siunčia A.
2. A apskaičiuoja raktą $K \equiv Y^{x_A} \pmod{p}$, B apskaičiuoja raktą $K \equiv X^{x_B} \pmod{p}$.

Šį protokolą galime naudoti ir didesnės grupės bendram simetrinės kriptosistemos raktui nustatyti. Tegu, pavyzdžiui, A, B ir C nesaugiu kanalu nori sudaryti bendrą raktą. Kaip ir anksčiau, pirmiausia jiems reikia susitarti dėl pirminio skaičiaus p ir generuojančio elemento g ir pasirinkti po skaičių. Pažymėkime A, B ir C pasirinktus skaičius x_A, x_B ir x_C. Tada jiems reikia atlikti šiuos protokolo žingsnius:

1. A siunčia B $g^{x_A} \pmod{p}$, gauna iš C $g^{x_C} \pmod{p}$; B siunčia C $g^{x_B} \pmod{p}$, gauna iš A $g^{x_A} \pmod{p}$; C siunčia A $g^{x_C} \pmod{p}$, gauna iš B $g^{x_B} \pmod{p}$.
2. A siunčia B $g^{x_A x_C} \pmod{p}$, gauna iš C $g^{x_B x_C} \pmod{p}$; B siunčia C $g^{x_A x_B} \pmod{p}$, gauna iš A $g^{x_A x_C} \pmod{p}$; C siunčia A $g^{x_B x_C} \pmod{p}$, gauna iš B $g^{x_A x_B} \pmod{p}$.
3. kiekvienas, keldamas 2-ajame žingsnyje gautą skaičių laipsniu savo rodikliu, apskaičiuoja bendrą raktą $K \equiv g^{x_A x_B c_C} \pmod{p}$.

Trumpai protokolą galima apibūdinti taip: gavęs skaičių, dalyvis kelia jį laipsniu savuoju rodikliu ir siunčia toliau. Pakėlę laipsniu paskutinįjį kartą, visi dalyviai gauna tą patį skaičių – raktą.

Šifravimas simetrine kriptosistema galimas tik baigus rakto nustatymo protokolą. Hugeso raktų nustatymo protokolas suteikia galimybę A užšifruoti ir nusiųsti B pranešimą šiandien, o susitarimą dėl rakto nukelti rytdienai.

Tarkime, raktas, kurį panaudojo šifravimui A, sudarytas taip: $K \equiv g^x \pmod{p}$. Dėl pirminio p ir generuojančio elemento g turi būti susitarta anksčiau arba A gali šiuos dydžius perduoti B nesaugiu kanalu.

Gavusi šifrą, kurio kol kas negali iššifruoti, B pradeda raktų nustatymo protokolą:

1. B parenka skaičių $y, (y, p-1) = 1$, ir A siunčia $Y \equiv g^y \pmod{p}$;
2. A, gavęs Y, siunčia B $X \equiv Y^x \pmod{p}$;

3. B skaičiuoja $z \equiv y^{-1} \pmod{p-1}$ ir randa raktą $K \equiv X^z \pmod{p}$.

Jeigu raktui nustatyti naudojamas šis protokolas, tai A gali nusiųsti tą patį šifruotą pranešimą keliems draugams, o kai jie norės jį perskaityti, įvykdyti rakto nustatymo protokolą.

Jeigu grupė dalyvių $D_1, D_2, \ldots, D_n$ nori naudotis simetrine kriptosistema su atskiru kiekvienai porai raktu, jie gali susitarti dėl p ir g ir paskelbti savo grupės tinklalapyje skaičius

$$X_1 \equiv g^{x_1} \pmod{p}, X_2 \equiv g^{x_2} \pmod{p}, \ldots, X_n \equiv g^{x_n} \pmod{p},$$

čia $0 < x_i < p-1$ dalyvio D_i pasirinktas skaičius. Tada D_i, norėdamas rašyti šifruotą laišką D_j, gali naudoti kaip raktą skaičių $K_{ij} \equiv X_j^{x_i} \pmod{p}$.

Nors šie protokolai išsprendžia rakto nustatymo uždavinį, jie yra bejėgiai prieš „vidurio ataką“. „Vidurio atakos“ esmė tokia: kai A siunčia B skaičių $X \equiv g^{x_A} \pmod{p}$, Z gali jį perimti, išsaugoti ir pasiųsti B $X' \equiv g^{x_Z} \pmod{p}$. Tą patį Z gali padaryti ir su B siunčiamu skaičiumi $Y \equiv g^{x_B} \pmod{p}$. Kai raktų nustatymo protokolas bus užbaigtas, A turės bendrą su Z raktą $K_1 \equiv g^{x_A x_Z} \pmod{p}$, o B – bendrą su Z raktą $K_2 \equiv g^{x_B x_Z} \pmod{p}$. Tada Z galės gautus iš A laiškus iššifruoti su K_1 ir, vėl užšifravęs su K_2, siųsti B. Analogiškai Z galės elgtis su B laiškais. Taigi Z taps nepagaunamu ryšio kontrolieriumi, jeigu tik nesukels įtarimo delsdamas persiųsti laiškų.

Kovai prieš vidurio atakas reikalingos papildomos priemonės ir protokolai. Galima išvengti vidurio atakų, naudojant raktų nustatymo protokoluose skaitmeninius parašus.

24.2. Įrodymai, nesuteikiantys žinių

Tokius įrodymus kartai pateikia informatikai profesionalai. Jeigu paklausite, kaip dirbama su kokia nors sudėtinga programa, tikriausiai išvysite virtuozišką „grojimą“ klaviatūra, greitą meniu ir langų kaitą... Jums bus įrodyta, kokia gera ši programa, tačiau kažin ar daug suteikta žinių...

Įrodymai, nesuteikiantys žinių (Zero Knowledge Proofs (ZKP), angl.), – tai įrodymai, kuriais įrodinėtoja I (Irena) tikrintojui T (Tomui) įrodo, kad žino tam tikrus duomenis d, tačiau jų neatskleidžia. Dar daugiau, iš tos informacijos, kurią I suteikia T, jo (o taip pat ir pasyviai stebinčių protokolo eigą) žinios apie d nepasikeičia. Galime suprasti, kad tai iš tiesų įmanoma, prisiminę, kaip naudojamasi viešojo rakto kriptosistema. Tarkime, I sukūrė viešojo rakto kriptosistemą: paskelbė viešąjį raktą K_v ir išsaugojo savo kompiuteryje privatųjį K_d. T nori įsitikinti, ar kriptosistema, kurią sukūrė I, „tikra“, ar I nepaskelbė rakto K_v šiaip sau... T gali užšifruoti tekstą M ir pasiųsti I šifrą $C = e(M|K_v)$, prašydamas, kad I atsiųstų M. Jeigu Irena įvykdė prašymą, tai Tomas gali pripažinti tai kaip įrodymą, kad Irena tikrai žino dešifravimui skirtą raktą.

Taigi įrodymas, nesuteikiantis žinių, yra tam tikras protokolas, kuriam pasibaigus T įsitikina, kad I tikrai žino tai, ką teigia žinanti, tačiau iš duomenų, naudotų vykdant protokolą, jos žinių nustatyti negalima. Panagrinėkime keletą pavyzdžių.

Grafą G galime užrašyti nurodydami jo viršūnių aibę V ir viršūnių, kurios yra sujungtos briaunomis, porų aibę E. Šias viršūnių poras vadinsime tiesiog briaunomis.

Grafai $G_1 = \langle V_1, E_1\rangle$ ir $G_2 = \langle V_2, E_2\rangle$ vadinami izomorfiškais, jeigu $|V_1| = |V_2|$ ir egzistuoja abipus vienareikšmė viršūnių atitiktis $\sigma : V_1 \to V_2$, kad

$$\langle v_1, v_2\rangle \in E_1 \text{ tada ir tik tada, kai } \langle \sigma(v_1), \sigma(v_2)\rangle \in E_2.$$

Turint grafą $G = \langle V, E\rangle$, sudaryti jam izomorfiškus grafus nesunku: parinkime kokią nors viršūnių perstatą $\sigma : V \to V$ ir apibrėžkime naują briaunų aibę $E' = \{\langle \sigma(v_1), \sigma(v_2)\rangle : \langle v_1, v_2\rangle \in E\}$. Naujasis grafas $H = \langle V, E'\rangle$ bus izomorfiškas G. Žymėsime jį $H = \sigma(G)$. Savo ruožtu G galime gauti iš H keitiniu σ^{-1}, taigi $G = \sigma^{-1}(H)$.

Tačiau, turint du grafus su tiek pat viršūnių, nustatyti, ar jie yra izomorfiški – sudėtingas uždavinys.

Tarkime, Irena teigia, kad grafai $G_1 = \langle V, E_1\rangle$ ir $G_2 = \langle V, E_2\rangle$ yra izomorfiški; kadangi viršūnių skaičiai abiejuose grafuose tie patys, galime jas žymėti tais pačiais simboliais, pavyzdžiui, skaičiais. Taigi I teigia žinanti abipus vienareikšmę atitiktį $\sigma : V \to V$, „išsaugančią“ briaunas. Ji gali įrodyti Tomui, kad nemeluoja, neatskleisdama atitikties σ.

Įrodymo protokolas vykdomas taip

I: įrodysiu, kad žinau $\sigma : G_1 \to G_2$
1. I pasirenka atsitiktinę perstatą $\lambda : V \to V$ ir, sudariusi naują grafą $H = \lambda(G_1)$, siunčia jį T; 2. T atsitiktinai parenka skaičiaus $i \in \{1, 2\}$ reikšmę ir siunčia I; 3. jeigu $i = 1$, tai I atsiunčia perstatą $\pi = \lambda$, jeigu $i = 2$ – perstatą $\pi = \lambda \circ \sigma^{-1}$. 4. gavęs π, T tikrina, ar $H = \pi(G_i)$;

Kadangi grafai G_1 ir G_2 yra izomorfiški, tai naujai sudarytas grafas H yra izomorfiškas jiems abiem. T antrajame žingsnyje pareikalauja, jog I atskleistų arba H ir G_1, arba H ir G_2 izomorfizmą. Jeigu ji žino G_1 ir G_2 izomorfizmą, ji nesunkiai gali įvykdyti abu reikalavimus. Kokia galimybė apgauti tikrintoją, t. y. pasiekti, kad jis patikėtų nesamomis žiniomis?

Tikimybė, kad antrajame žingsnyje T pasirinks $i = 1$, lygi $\frac{1}{2}$. Tai ir yra apgavystės tikimybė, nes I atsakymui visai nereikia žinių apie G_1 ir G_2 izomorfizmą. Protokolą reikia kartoti. Jeigu jis kartojamas k kartų, tai apgavystės tikimybė lygi $\frac{1}{2^k}$. Labai svarbu, kad I negalėtų nuspėti, kokį skaičių atsiųs T. Jeigu, pavyzdžiui, ji tikrai žino, kad T siųs skaičių $i = 2$, ji protokolo pradžioje gali pasiųsti grafą $H = \lambda(G_2)$ ir, gavusi $i = 2$, siųsti tikrintojui $\pi = \lambda$.

Siuntinėti grafus ir atlikti su jais skaičiavimus nelabai patogu. Geriau siuntinėti skaičius. Panagrinėkime Fiato-Shamiro protokolą, kuriuo naudodamasi I įrodo, kad žino lyginio

$$x^2 \equiv a \pmod{n}, \quad n = pq,$$

sprendinį v, čia p, q yra pirminiai skaičiai, jų sandauga n paskelbiama viešai. Taigi I žino skaičių $v, v^2 \equiv a \pmod{p}$, ir nori jo neatskleisdama tai įrodyti T. Protokolas vykdomas taip:

I: įrodysiu, kad žinau $v, v^2 \equiv a$ **(mod** n**)**
1. I pasirenka atsitiktinį skaičių r ir siunčia T $b \equiv r^2 \pmod{n}$;
2. T atsitiktinai pasirenka $i \in \{0, 1\}$ ir siunčia I;
3. jeigu $i = 0$, I siunčia T skaičių $h = r$, jeigu $i = 1$, ji siunčia $h = rv$;
4. T tikrina, ar $h^2 \equiv a^i b \pmod{n}$; jeigu lyginys teisingas – įrodymą priima.

Jeigu I laikosi protokolo, T visada priims įrodymą. Iš tikrųjų, jei $i = 0$, tai T tikrina, ar $b \equiv r^2 \pmod{n}$. Jeigu $i = 1$, tai T tikrina, ar $ab \equiv (rv)^2 \pmod{n}$.

Apgavystės tikimybė ir čia lygi $\frac{1}{2}$. Jeigu $i = 0$, tai I atsakymui dydžio v nereikia. Tačiau jeigu ji iš tiesų nežinotų v, bet žinotų, kad T atsiųs skaičių $i = 2$, ji irgi galėtų apgauti. Prieš pirmąjį žingsnį ji galėtų pasirinkti bet kokį h, suskaičiuoti $b \equiv h^2 a^{-1} \pmod{n}$ ir nusiųsti šį skaičių pirmajame žingsnyje, o antrajame – h. Lyginys, kurį tikrins T, bus teisingas.

Kad apgavystės tikimybė būtų maža, protokolą reikia pakartoti kelis kartus.

Iš šių pavyzdžių matyti, kad žinios, kurių turėjimą įrodinėja I, yra sudėtingo skaičiavimo uždavinio sprendinys. Panagrinėkime dar vieną protokolą, kuriuo naudodamasi I įrodo, kad žino skaičiaus diskretųjį logaritmą.

Tegu p yra didelis pirminis skaičius, $g \in \mathbb{F}_p$ yra generuojantis elementas, $\beta \in \mathbb{F}_p$ – bet koks nenulinis elementas. Dydžiai p, g, β yra vieši. I žino diskretųjį logaritmą $u = \log_g \beta$ ir nori tai įrodyti T.

Protokolo eiga tokia:

I: įrodysiu, kad žinau u, $g^u \equiv \beta$ **(mod** p**)**
1. I pasirenka atsitiktinį skaičių $0 < r < p-1$ ir, apskaičiavusi $\gamma \equiv g^r \pmod{p}$, siunčia T; 2. T atsitiktinai pasirenka $i \in \{0,1\}$ ir siunčia I; 3. I skaičiuoja $h \equiv r + iu \pmod{p-1}$ ir siunčia T; 4. T tikrina, ar $g^h \equiv \beta^i \gamma \pmod{p}$. Jei lyginys teisingas, T priima įrodymą.

Apgavystės tikimybė šiame protokole ta pati kaip ankstesniuose. Ir šiame protokole svarbu, kad I negalėtų nuspėti, kokius skaičius atsiųs T.

24.3. Skaitmeniniai pinigai

Palyginkime du atsiskaitymo būdus. Pirmasis – atsiskaitymas parduotuvėje banko kortele. Antrasis toks: jūs rašote laišką broliui, seseriai ar tėvams, prašydami, kad jie iš tos metalinės dėžutės, kurioje laikote savo santaupas, paimtų tam tikrą sumą ir ją atiduotų A, kuriam esate skolingas. Ar skiriasi šie du būdai? Iš esmės ne!

Taigi banko kortelės su skaitmeniniais pinigais neturi nieko bendro. O kas gi yra tada tie skaitmeniniai pinigai? O kas yra popieriniai pinigai? Tam tikri dokumentai, kuriuos duodame pardavėjui, jis patikrina juos specialiu aparatu ir, ir nepaklausęs nei mūsų vardo, nei pavardės, leidžia pasiimti prekes. Vadinasi, skaitmeniniai pinigai turėtų būti tam tikri skaitmeniniai duomenys, kuriuos mes duodame pardavėjui, o jis – atitinkamai patikrinęs – priima ir nereikalauja mus identifikuojančių duomenų.

Jeigu norime sukurti skaitmeninius pinigus, turinčius pagrindines popierinių pinigų savybes, tačiau sudarytus vien tik iš skaitmenų (arba bitų, jeigu norite), reikia sugalvoti, kaip galima garantuoti tokius reikalavimus:

- autentiškumas: niekas negali pasiimti skaitmeninių pinigų mūsų vardu;
- vientisumas: sukurta pinigas negali būti pakeistas (prie 1 negali būti „prirašyta“ keletas nulių;
- tiesioginis mokėjimas: atsiskaitant skaitmeniniais pinigais, nereikia kreiptis į banką;
- saugumas: tie patys skaitmeniniai pinigai negali būti išleisti pakartotinai;

- anonimiškumas: atsiskaitant skaitmeniniais pinigais, neturi būti reikalinga informacija apie mokėtojus.

Reikalavimų daug ir jie sunkiai įgyvendinami. Juk, pavyzdžiui, kopijuoti skaitmeninius duomenis nereikia nei ypatingų prietaisų, nei sugebėjimų!

Panagrinėkime vieną kiek supaprastintą skaitmeninių pinigų sistemą ECash, kurią sukūrė kompanija DigiCash.

Taigi A nori gauti skaitmeninių pinigų ir jais atsiskaityti už prekes ar paslaugas. Žinoma, jis turi pirmiausia uždirbti tikrų pinigų ir atsidaryti sąskaitą banke B, kad galėtų šiuos tikruosius pinigus versti skaitmeniniais. Kad A vardu į banką, prašydamas skaitmeninių pinigų, negalėtų kreiptis Z, A turi naudotis kokia nors nustatyta autentifikavimo schema, pavyzdžiui, skaitmeniniais parašais. Skaitmeninis parašas būtinas ir bankui B, kad būtų galima patvirtinti išduotų skaitmeninių banknotų tikrumą. Tarkime, kad naudojama RSA skaitmeninių parašų schema. Naudojantis ja, galima kurti aklus parašus, duodant pasirašyti „užmaskuotus“ dokumentus. Protokoluose daugybos bei kėlimo laipsniu veiksmai atliekami atitinkamu moduliu.

ECash schemos skaitmeninių banknotų išdavimo protokolas

Tarkime, A nori gauti 100 EU skaitmeninį banknotą. Įdomu, kad jam ne tik nedraudžiama jį susikurti, bet jis netgi privalo pats tai padaryti.

1. A parengia n (banko nustatytą kiekį, pavyzdžiui, $n = 100$) skaitmeninių eilučių rinkinių $S_j = (I_{j1}, I_{j2}, ..., I_{jn})$, $j = 1, ..., n$; kiekvienoje eilutėje I_{jk} užrašyta A identifikuojanti informacija.

2. Kiekviena eilutė I_{jk} kaip paslaptis padalijama į dvi dalis (L_{jk}, R_{jk}).

3. A parengia n banknotų po 100 EU: $M_j = (m_j, (L_{jk}, R_{jk})_{k=1,...,n})$, čia m_j yra skirtingi banknotų identifikavimo numeriai, juose sutartu būdu nurodyta ir banknoto vertė.

4. A maskuoja banknotų numerius ir siunčia bankui $M_j^* = (z_j^e m_j, (L_{jk}, R_{jk})_{k=1,...,n})$, čia e yra banko B skaitmeninio parašo schemos viešasis raktas, z_j – A pasirinktas skaičius.

5. B atrenka $n-1$ atsiųstą banknotą (pvz., $M_1^*, \ldots, M_{99}^*$) ir pareikalauja, kad A nurodytų šiems banknotams kurti panaudotus maskuojančius daugiklius z_j.

6. B patikrina, ar atrinktieji banknotai sudaryti teisingai: ar visuose juose nurodytos vienodos vertės ir visi numeriai skirtingi.

7. Jeigu atskleistieji kvitai sudaryti teisingai, B pasirašo likusįjį ir siunčia A $((z_{100}^e m_{100})^d, (L_{100,k}, R_{100,k})_{k=1,...,n})$.

8. A pašalina maskuojamąjį daugiklį ir turi skaitmeninį banknotą $(m_{100}, (m_{100})^d)$ ir dar tam tikrą jo „priedą“ $I_{100} = (L_{100,k}, R_{100,k})_{k=1,...,n}$.

Įvykdyto protokolo rezultatas: A turi banko parašu patvirtintą 100 eurų vertės banknotą su numeriu, kurio bankas nežino. Kai bankas šį numerį sužinos, negalės jo susieti su A. Apskritai skaitmeninio banknoto struktūra turėtų būti sudėtingesnė; I_{100} irgi turėtų būti banko pasirašytas.

ECash mokėjimo protokolas

1. A pateikia pardavėjui P banknotą $(m_{100}, (m_{100})^d)$.
2. P tikrina banko B parašą $m_{100} = ((m_{100})^d)^e$.
3. P generuoja atsitiktinių bitų seką $b_1 b_2 \dots b_{100}$ ir perduoda A. Jeigu $b_i = 0$, A turi atskleisti $L_{100,i}$; jei $b_i = 1$, A turi atskleisti $R_{100,i}$.
4. P siunčia B $(m_{100}, (m_{100})^d)$ ir atskleistas $I_{100,i}$ dalis.
5. B taip pat patikrina parašą ir peržiūri savo duomenų bazę, ar pinigas su numeriu m_{100} dar nebuvo išleistas. Jeigu ne – perveda į P sąskaitą atitinkamą sumą, o į savo duomenų bazę įrašo atsiųstą informaciją. Jeigu jau išleistas – aiškinasi, kas kaltas.

Trečiojo žingsnio vykdymas reikalauja, kad A negalėtų sumeluoti atskleisdamas paslapčių dalis. Todėl ir jos turėtų būti patvirtintos banko parašu.

Kas gi atsitinka, jeigu B nustato, kad banknotas su tuo numeriu jau išleistas? Tuomet jis lygina atsiųstas paslapčių dalis su įrašytomis duomenų bazėje. Jeigu visos jos sutampa – pardavėjas P bando banknotą išgryninti pakartotinai. O jeigu nesutampa, tai banknotą bando pakartotinai panaudoti A. Tada iš pirmų nesutampančių paslapties dalių, tarkime, $L_{100,k}$ ir $R_{100,k}$, bankas nustato nesąžiningo kliento tapatybę. Gali atsitikti, kad visos atsiųstos paslapčių dalys sutaps su jau įrašytomis ne dėl pardavėjo kaltės, bet todėl, kad A bando skaitmeninį pinigą išleisti pakartotinai. Tačiau tokio atvejo tikimybė yra maža – tik 2^{-n}.

Ši schema įgyvendina visus minėtus skyrelio pradžioje reikalavimus skaitmeniniams pinigams. Tačiau, vykdant protokolo žingsnius, tenka atlikti gana daug skaičiavimų. Daugiausiai laiko sugaištama, bankui tikrinant atsiųstus banknotų variantus.

Yra ir kitokių skaitmeninių pinigų schemų. Visos jos yra gana sudėtingos. Tačiau kitaip ir būti negali. Norint praktiškai įdiegti aptartąją schemą, reiktų ją padaryti dar sudėtingesnę. Juk joje nenumatyta, pavyzdžiui, grąžos davimo galimybė.

24.4. Elektroniniai rinkimai

Balsavimo kabina su užuolaidėle – klasikinio saugaus balsavimo protokolo elementas. Kuo ją galima pakeisti, jeigu norėtume dalyvauti rinkimuose neatsitraukdami nuo savo kompiuterio?

Jeigu ketinate sukurti patikimą elektroninio balsavimo sistemą, teks pagalvoti, kaip įgyvendinti ne mažiau kaip skaitmeninių pinigų schemose sąlygų. Žinoma, jeigu pasitikite ryšio kanalais, rinkėjais ir rinkimine komisija, tai galite nesukti sau galvos. Jeigu ne – tada teks sugalvoti, kokių priemonių reikia imtis, kad

1. tik registruoti rinkėjai galėtų balsuoti;
2. kiekvienas rinkėjas galėtų balsuoti tik kartą;
3. niekas kitas negalėtų sužinoti, kaip rinkėjas balsavo;
4. kiekvienas rinkėjas galėtų įsitikinti, kad jo balsas įskaitytas;
5. joks rinkėjas negalėtų dubliuoti kito rinkėjo biuletenio;
6. niekas negalėtų pakeisti rinkėjo biuletenio.

Nors galima sukurti elektroninio balsavimo sistemą, kurioje nėra centrinės rinkimų komisijos, t. y. patys rinkėjai gali sekti, ar balsavimas vyksta korektiškai, protokolai supaprastės, jeigu tokią komisiją (toliau CRK) įsteigsime.

Paprasčiausias balsavimo protokolas galėtų būti toks. CRK paskelbia savo viešąjį raktą, skirtą biuleteniams šifruoti.

1. Rinkėjai šifruoja savo biuletenius viešuoju CRK raktu ir išsiunčia juos CRK;
2. CRK dešifruoja biuletenius, skaičiuoja ir skelbia rezultatus.

Toks balsavimas ne ką geresnis už balsavimą SMS žinutėmis įvairiuose televizijos renginiuose. Kas nori, gali balsuoti kiek nori kartų. Įgyvendinama tik paskutinė sąlyga: niekas negali pakeisti rinkėjo biuletenio (tačiau CRK gali jo neįskaityti).

Jeigu kiekvienam rinkėjui bus suteikta galimybė naudotis skaitmeninio parašo schema, galima naudoti geresnį protokolą.

1. Kiekvienas rinkėjas pasirašo biuletenį savo privačiuoju raktu.
2. Pasirašytą biuletenį rinkėjas šifruoja viešuoju CRK raktu ir jį išsiunčia CRK;
3. CRK dešifruoja biuletenius, tikrina parašus, skaičiuoja ir skelbia rezultatus.

Toks protokolas kur kas geresnis, tačiau CRK jame turi labai daug galios: ji žino, kas ir kaip balsavo, jeigu nori – gali neįskaityti dalies balsų.

Taigi vien skaitmeninių parašų neužtenka. Prisiminkime, kaip skaitmeninių pinigų sistemose naudojami aklieji parašai. Jie praverčia ir elektroninio balsavimo schemose. Štai dar vienas sudėtingesnis balsavimo protokolas.

1. Kiekvienas rinkėjas parengia n biuletenių rinkinių (pavyzdžiui, $n = 10$); kiekviename rinkinyje yra visiems balsavimo variantams parengti biuleteniai; rinkinyje kiekvienas biuletenis pažymėtas numeriu; skirtingų rinkinių numeriai yra skirtingi, jie renkami iš didelių skaičių aibės.

2. Rinkėjas maskuoja kiekvieną biuletenių rinkinį, parengdamas aklam parašui, ir išsiunčia CRK.

3. CRK tikrina, ar rinkėjas nebuvo atsiuntęs biuletenių anksčiau. Jeigu ne – atsitiktinai atrenka $n-1$-ą rinkinį ir paprašo atsiųsti šių rinkinių atskleidimo dydžius. Rinkėjas įvykdo prašymą, CRK patikrina, ar atskleistieji biuleteniai tinkamai sudaryti. Jeigu jie sudaryti pagal taisykles, CRK pasirašo paskutinį neatskleistąjį rinkinį ir nusiunčia rinkėjui.

4. Rinkėjas atskleidžia atsiųstąjį rinkinį ir gauna CRK parašu patvirtintus visų balsavimo variantų biuletenius.

5. Rinkėjas iš turimo rinkinio atrenka tą biuletenį, kuris atitinka jo valią, šifruoja jį viešuoju CRK raktu ir išsiunčia.

6. CRK dešifruoja biuletenį, patikrina savo parašą, perskaito numerį ir patikrina, ar jau priimtų biuletenių numerių sąraše tokio numerio nėra, skaičiuoja ir skelbia rezultatus, nurodydama biuletenių numerius ir kurio kandidato ar partijos naudai jie įskaityti.

Šis protokolas įgyvendina visus šešis anksčiau suformuluotus reikalavimus. Vis dėlto ir juo naudojantis gali kilti nesusipratimų. Rinkėjas gali patikrinti, ar jo balsas teisingai įskaitytas. Tačiau jeigu komisija suklydo arba neteisingai įskaitė tyčia, kaip rinkėjas gali tai įrodyti?

Galima sudaryti sudėtingesnius protokolus, kurie leidžia išvengti tokių situacijų. Galima suteikti galimybę rinkėjui, nepatenkintam savo balso įskaitymu ir norinčiam pakeisti savo nuomonę, tai padaryti. Elektroninis balsavimas pagal tokį protokolą galėtų padėti subalansuoti balsus ir išvengti pakartotinių rinkimų. Tačiau taip bus daroma, matyt, dar negreitai.

24.5. Pokeris telefonu

> *Algis ir Birutė kalbasi telefonu ir niekaip negali susitarti, kas iš jų turėtų grįžti namo anksčiau ir išvesti pasivaikščioti namuose paliktą šunį. Algis sako: „Metu monetą, jeigu ji atvirs herbu, sugrįšiu aš. Vienas, du, trys – atvirto skaičiusį' Ar tikrai Birutė niekaip negali patikrinti, kaip iš tikrųjų buvo?*

Monetos metimo protokolą telefonu garantuojant, kad baigtis bus paskelbta teisingai, tikrai galima surengti. Ir netgi daugeliu būdų. Telefonas, žinoma, čia tik dėl didesnio įspūdžio. Geriau už telefoną tokiems protokolams atlikti tinka elektroninis paštas. Panagrinėkime keletą.

1. A parenka didelį pirminį skaičių p ir praneša jį B.
2. B randa du generuojančius elementus $h, t \in \mathbb{F}_p$ ir siunčia A.
3. A parenka atsitiktinį skaičių x, skaičiuoja $y \equiv h^x \pmod{p}$ arba $y \equiv t^x \pmod{p}$ ir siunčia šį skaičių B.
4. B spėja, ar, skaičiuojant y, buvo naudotas h, ar t.
5. A patikrinimui siunčia B reikšmę x.

Galbūt geriau šį protokolą vadinti „įspėk, kurioje rankoje“ protokolu. Birutė turi galimybę pasirinkti ir pasitikrinti, ar neapsiriko. Tačiau galima jos rinkimąsi interpretuoti ir kaip monetos metimą. Jeigu ji įspėja – „moneta atvirto skaičiumi“, jeigu ne – „atvirto herbu“. Svarbu, kad baigties negali nei vienas iš dalyvių užginčyti.

O štai dar vienas protokolas, kuriame naudojamasi Pohligo-Hellmano simetrine kriptosistema. Prisiminkime, kaip ji veikia.

Parenkamas didelis pirminis skaičius p ir dar du skaičiai $k_1 = e, k_2 = d$, $e \cdot d \equiv 1 \pmod{p-1}$. Pranešimas m šifruojamas ir dešifruojamas taip:

$$c = e(m|k_1) \equiv m^{k_1} \pmod{p}, \quad m = d(c|k_2) \equiv c^{k_2} \pmod{p}.$$

„Monetos metimo“ telefonu protokolas naudojant Pohligo-Hellmano kriptosistemą

1. A ir B susitaria dėl bendro pirminio skaičiaus p ir kiekvienas pasirenka po porą kriptosistemos raktų: $K_A = \langle e_A, d_A \rangle, \quad K_B = \langle e_B, d_B \rangle$.
2. A užšifruoja dvi žinutes: m_1=„herbas“, m_2=„skaičius“

   $$c_1 \equiv m_1^{e_A} \pmod{p}, \quad c_2 \equiv m_2^{e_A} \pmod{p}$$

   ir siunčia B.
3. B pasirenka iš atsiųstųjų vieną žinutę c_i ir ją šifruoja $c_i' \equiv c_i^{e_B} \pmod{p}$, tada siunčia A.
4. A c_i' dešifruoja su raktu d_A $c_i^* \equiv (c_i')^{d_A} \pmod{p}$, išsaugo c_i^*, o taip pat pasiunčia B.
5. B iššifruoja $m_i \equiv (c_i^*)^{d_B} \pmod{p}$ ir praneša A, kas „iškrito“. Kad A galėtų patikrinti, ar B nemeluoja, B siunčia A savo raktą d_B.

Galima telefonu (arba elektroniniu paštu) netgi organizuoti lošimą kortomis, garantuojant, kad dalyviai negalės sukčiauti. Tokiam lošimui organizuoti reikia tokių dalinių protokolų: kortų dalijimo, kortų atskleidimo, lošimo korektiškumo tikrinimo.

Protokolui panaudosime tą pačią Pohligo-Hellmano simetrinę kriptosistemą. Iš pradžių A ir B turi susitarti dėl bendro pirminio skaičiaus p ir pasirinkti po porą raktų: $K_A = \langle e_A, d_A \rangle$, $K_B = \langle e_B, d_B \rangle$.

Kortų dalijimas

Tikslas: A ir B reikia duoti po k atsitiktinai parinktų kortų.

1. A užšifruoja žinutes-kortų pavadinimus $m_1, m_2, \ldots, m_n$, surašytus atsitiktine tvarka, ir siunčia B šifrus $c_i \equiv m_i^{e_A} \pmod{p}$.

2. B atrenka k skaičių c_i ir nusiunčia A. A iššifruoja c_i ir žino, kokios kortos jam teko.

3. B atrenka dar k kortų, jas užšifruoja: $c_i' \equiv c_i^{e_B} \pmod{p}$ ir siunčia A.

4. A iššifuoja gautas žinutes $c_i^* \equiv (c_i)'^{d_A} \pmod{p}$ ir siunčia atgal B.

5. B iššifruoja $m_i \equiv (c_i^*)^{d_B} \pmod{p}$ ir žino, kokios kortos B teko.

Protokolas pasibaigė – abu lošėjai gavo po k kortų, kokias kortas gavo priešininkas – nežino. Tačiau kiekvienas žino priešininko kortų šifrus. Galima pradėti lošti, t. y. vieną po kitos atskleisti kortas.

Kortos atskleidimas

Kai A daro ėjimą (atskleidžia kortą), siunčia partneriui porą $\langle m_i, c_i \rangle$, $c_i \equiv m_i^{e_A} \pmod{p}$. Jis negali pasiųsti tokios kortos, kurios negavo, nes visi jo kortų šifrai žinomi B. Tačiau kol kas B negali patikrinti, ar c_i tikrai yra m_i šifras.

Analogiškai atskleisdama savo kortas elgiasi ir B.

Lošimo korektiškumo tikrinimas

Kai lošimas baigiasi, partneriai turi pasikeisti raktais, kad galėtų patikrinti, ar buvo lošta tomis kortomis, kurias jie turėjo.

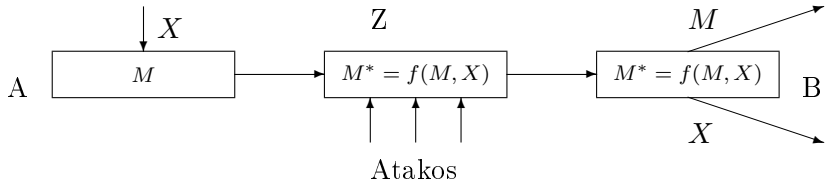
25 Quo vadis?

Kokia bus XXI amžiaus kriptografija? O gal jos iš viso nebereikės? Veltui spėliotume. Geriau pabandykime aptarti dvi duomenų apsaugos kryptis, kurios, be abejonės, bus plėtojamos XXI amžiaus kriptografijoje. Viena iš jų iš tiesų labai sena, senesnė už pačią civilizaciją. Na, o kita – vos dviejų dešimčių metų amžiaus.

25.1. Slėpimo menas

Žuvys, paukščiai ir žvėrys... – amžių amžius trunkanti kova už būvį išmokė juos pasislėpti, kad nepastebėtų priešai. Informacijos srautų judėjimo nepaslėpsi, tačiau galima jais pasinaudoti tarsi konteineriais su dvigubu dugnu ir perduoti nepastebėtus duomenis.

Jeigu kriptoanalitikas Z yra ne šiaip pasyvus kanalo stebėtojas, bet turi galią jį valdyti (pavyzdžiui, jeigu jūsų įstaigos darbuotojų elektroninis paštas yra kontroliuojamas), tai šifrai nepadės. Įtartino turinio siuntiniai bus tiesiog nesiunčiami. Tokiu atveju pranešimui apie negeroves pasiųsti galite pasinaudoti steganografija. *Steganos* graikiškai reiškia slėpti. Kriptografiniais metodais duomenys yra transformuojami, kad būtų paslėpta jų prasmė, tačiau pats duomenų siuntimas yra tikras faktas. Steganografiniais metodais slapta siunčiami duomenys paslepiami kituose, kurie kanalo kontrolieriams neturi kelti įtarimų. Taigi steganografija – tai sritis, kuri siūlo būdus, kaip vienus duomenis panaudoti kaip konteinerį kitiems duomenims siųsti.



Slapto pranešimo X steganografinio perdavimo schema.

Steganografiniai metodai buvo vienaip ar kitaip naudojami visuomet. Juos galima sąlyginai suskirstyti į dvi grupes: lingvistinius ir techninius. Lingvistiniai metodai kaip slapto pranešimo perdavimo konteinerį naudoja kalbą. Tekstai suteikia daug galimybių atlikti nedidelius pakeitimus, šitaip paslepiant reikalingus duomenis. Tarpų tarp žodžių skaičius, kiek nelygiai išdėstytos raidės – visa tai gali būti panaudota duomenims paslėpti.

F. L. Baueris savo knygoje „Decrypted secrets“ pateikia 1976 metais tuometinėje Vokietijos Demokratinėje Respublikoje išleisto kombinatorinės logikos vadovėlio puslapio nuotrauką. Gerai įsižiūrėję galime pastebėti, kad kai kurios raidės yra šiek tiek žemiau negu kitos (spausdinant mechanine spausdinimo mašinėle tai įprastas dalykas). Surašius visas tokias raides, galima perskaityti *„nieder mit dem sowjetimperialismus“* (šalin tarybinį imperializmą).

Viktorijos laikų Anglijoje buvo galima siuntinėti laikraščius paštu nemokamai. Žmonės sugalvojo pasinaudoti tuo žinioms perduoti: pakanka virš laikraščio straipsnių raidžių adata pradurti skylutes, kad, skaitant šitaip pažymėtas raides, susidarytų perduoti skirtas tekstas. Toks metodas buvo naudojamas netgi II pasaulinio karo metais ir po jo. Tik adatos skylutes pakeitė nematomo rašalo taškeliai.

Techninės steganografijos metodai duomenims paslėpti naudoja specialias technines priemones arba įrankius. Škotijos karalienei Marijai skirti laiškai buvo perduodami alaus bokaluose. Šio steganografinio metodo netobulumas irgi buvo viena iš priežasčių, dėl kurių karalienė buvo pasmerkta nukirsdinti.

Nematomas rašalas, mikrofilmai – kur kas geresnės priemonės, bet ir dėl jų daug kas nukentėjo.

Mūsų laikų steganografija – vienų skaitmeninių duomenų slėpimo kituose metodai. Kokie duomenys kaip konteineriai geriausiai tinka steganografijai? Žinoma, tokie, kuriuose nedidelio skaičiaus bitų pokyčiai sunkiai pastebimi – grafikos, audio- ir videoduomenys. Kiekvienam nuotraukos taškui (pikseliui) koduoti naudomi 24 bitai. Jei pakeisime vieną ar kelis jų – akis to nepastebės, tačiau tie pakeistieji gali būti panaudoti slaptam pranešimui (tekstui, vaizdui, garsui) perduoti. Yra visokių metodų ir tuos metodus realizuojančių programų.

Steganografinio metodo vertę lemia trys jo savybės: aptinkamumas (detectability, angl.), atsparumas (robustness, angl.), informacinė talpa (bit rate, angl.).

Steganografinis metodas yra tuo patikimesnis, kuo sunkiau, turint konteinerį (duomenis, kurie gali būti panaudoti kitiems paslėpti), nustatyti, ar jame yra kas nors, ar jis „tuščias“, t. y. ar jame paslėpti kiti duomenys, ar ne. Paslėptus duomenis galime interpretuoti kaip „triukšmą“, padidinantį entropiją. Atlikdamas konteinerio analizę, analitikas gali vertinti, ar turimų duomenų dydis yra būdingas tokio tipo duomenims. Pavyzdžiui, jeigu grynos spalvos 300×300 taškų kvadratas, kuriam saugoti jpg formatu turi užtekti 2 kilobaitų duomenų, siunčiamas kaip dvigubai daugiau duomenų užimantis failas, tai natūralu įtarti, kad kvadratas yra konteineris, kuriame kažkas yra.

Analitikas gali atlikti įtariamo konteinerio ataką, siekdamas jį „iškratyti“. Pavyzdžiui, paveikslą galima pasukti, pakeisti mastelį, atlikti kitas transformacijas, mažai keičiančias jo išvaizdą. Ar paslėpti duomenys nedings? Paslėptų duomenų savybė išsilaikyti po tam tikrų konteinerio modifikacijų ir yra metodo atsparumas.

Taip pat svarbu, kiek bitų įmanoma paslėpti, kad steganografinio metodo panaudojimą dar būtų sunku pastebėti. Slepiamų bitų kiekio ir konteinerio bitų kiekio santykiu galima nusakyti informacinę metodo talpą.

Steganografijai dera priskirti ir vandens ženklų (watermarks, angl.) kūrimo uždavinį. Vandens ženklas – tai liudijimas apie autorystę. Vandens ženklą popieriuje galime pamatyti pažvelgę į jį prieš šviesą. Skaitmeninis vandens ženklas – tai tam tikra informacija, siejama su duomenimis (kūriniu), kuri netrukdo juos naudoti nustatytiems tikslams (klausyti muzikinio kūrinio ar žiūrėti filmą), tačiau ji gali būti registruojama naudojant specialius analizės metodus. Tai šiek tiek panašu į skaitmeninius parašus, tačiau parašai paprastai pridedami kaip priedas (ir todėl gali būti nesunkiai pašalinami), o vandens ženklas turi būti „išlietas“ visame kūrinyje.

25.2. Kvantai ir bitai

Ženklai ant popieriaus, duomenys, perduodami elektros srovės impulsais, elektromagnetinėmis ar garso bangomis – visi šie kanalai turi

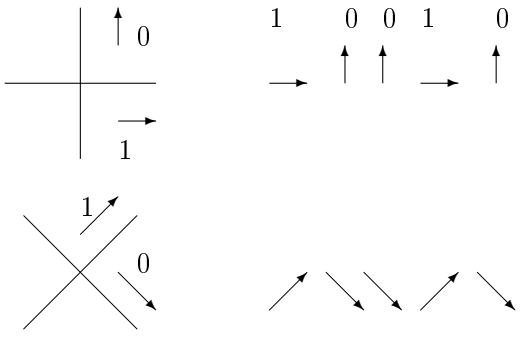
vieną bendrą savybę: duomenis galima nukopijuoti nepakeičiant originalų. Visai kas kita, jeigu duomenų bitus „neša“ fotonai...

Įsivaizduokime fotonus kaip nedalomas šviesos daleles. Idėja, kad reiškinių struktūros pagrindas – mažos dalelės, sena kaip mūsų civilizacija. Prisiminkime, kaip pasaulį aiškino Demokritas... Tačiau šviesos dalelės – fotonai – skiriasi nuo visų dalelių, kurias žmonės įsivaizdavo, tuo, kad jų elgesį valdo visai kitokie dėsniai. Tų dėsnių sąvadas dėstomas kvantinėje mechanikoje. Viena svarbiausių fotonų savybių yra tokia: negalima „išmatuoti“ fotonų nepakeičiant jų savybių. Fotonas prieš matavimą ir po jo – nebe toks pats!

Pirmąsias kvantinės mechanikos dėsnių panaudojimo kriptografijai idėjas išdėstė S. Wiesneris maždaug apie 1970 metus. Tačiau tuomet tos idėjos atrodė, matyt, pernelyg „beprotiškos“. S. Wiesnerio straipsnis buvo paskelbtas tik po gero dešimtmečio. Ir greitai sulaukė tęsinio: 1984 metais Ch. Bennettas (IBM) ir G. Brassard'as (Monrealio universitetas) pasiūlė pirmąjį kvantinės kriptografijos protokolą (pirmoji publikacija [6]). Kriptografijoje jis vadinamas BB84 protokolu. Šiame skyrelyje pabandysime išsiaiškinti, kaip jis vykdomas. Pamatysime, kad kvantinės kriptografijos rūpesčiai visai kitokie negu klasikinės.

Tačiau iš pradžių aptarkime, kaip bitams transportuoti galima panaudoti fotonus. Žinoma, teks labai supaprastinti fizinę realybę, tačiau mums reikia tik tų jos savybių, kurios panaudojamos kriptografijoje.

Pirmiausia įsivaizduokime fotoną kaip dalelę, kurią, naudojant tam tikrų filtrų sistemą, galima poliarizuoti, t.y. suteikti vieną iš dviejų savybių. Naudosime dvi tokių filtrų sistemas, kurias vadinsime atitinkamai + baze ir $\times$ baze. Fotonų poliarizacijas, kurias suteikia + bazė, žymėsime $\uparrow$ ir $\rightarrow$, o $\times$ bazės poliarizacijas – $\nearrow$ ir $\searrow$. Jeigu fotonui poliarizuoti naudota + bazė, sakysime, kad fotonas su poliarizacija $\uparrow$ „neša“ 0, o fotonas $\rightarrow$ „neša“ 1. Atitinkamai $\times$ bazės poliarizacijoms $\searrow$ ir $\nearrow$ irgi priskirkime 0 ir 1.



Tą patį bitų srautą gali „pernešti“ abiejų bazių poliarizuoti fotonai.

Tarkime, A nori perduoti bitų srautą B, pasinaudodamas poliarizuotais fotonais. Mums nesvarbi tikroji fizinė tų fotonų sklidimo terpė – jie

gali sklisti erdve, gali sklisti optiniu kabeliu, svarbu, kad sklistų. Tarkime, visiems fotonams poliarizuoti A pasinaudojo + baze. Gautų fotonų poliarizacijai matuoti turi būti naudojama viena iš dviejų bazių. Tarkime, B visus fotonus registruoja su ta pačia + baze. Visi jo poliarizacijų matavimai bus teisingi ir B gaus tą bitų srautą, kurį siuntė A. Tačiau toks ryšio kanalas niekuo ne geresnis už įprastinius nesaugius kanalus. Kriptoanalitikas Z, žinodamas, kad A ir B fotonams matuoti naudoja tą pačią + bazę, irgi ja pasinaudos: registruos visus A siunčiamus fotonus, užsirašys gautus bitus, o tada su ta pačia + baze generuos tokį pat fotonų srautą ir nusiųs B. Taigi Z bus nepastebimai įsiterpęs į ryšio kanalą, kaip paprastai būna klasikinėje kriptografijoje.

Aptarkime, kokio rezultato galima laukti, jeigu A poliarizavo fotoną su + baze, o B registravo su × baze. Tarkime, A pasiuntė fotoną su poliarizacija →, t.y. nori perduoti bitą, kurio reikšmė 1. Čia ir prasideda visas įdomumas: kvantinės mechanikos dėsniai sako, kad su vienodomis tikimybėmis bus registruotas fotonas su poliarizacijomis ↘ ir ↗ . Kitaip tariant, jeigu B nepataikė pasirinkti bazės, tai matavimo rezultatas bus toks pat, koks būtų paprasčiausiai metant simetrinę monetą!

O dabar jau žinome iš esmės viską, ko reikia, kad suprastume, kaip veikia kvantinės kriptografijos BB84 protokolas. Jo tikslas – perduoti tam tikrą bitų kiekį taip, kad siuntėjas A ir gavėjas B būtų užtikrinti, kad tie bitai nebuvo nukopijuoti Z. O tada galima iš perduotų bitų sudarytą žodį naudoti kaip įprastinės simetrinės kriptosistemos raktą. Juk geros simetrinės kriptosistemos yra labai saugios, jeigu saugiai perduodami raktai!

Kvantinis bitų perdavimo protokolas BB84

1. A rengiasi perduoti B bitų srautą $x_1x_2\ldots x_n$. A atsitiktinai pasirenka bazių seką $a_1a_2\ldots a_n$ ir perduoda B fotonus: bitą x_i „neša“ fotonas, poliarizuotas bazėje a_i.

2. B atsitiktinai pasirenka bazių seką $b_1b_2\ldots b_n$ ir, naudodamasis šiomis bazėmis, matuoja gaunamų fotonų poliarizacijas: i-asis fotonas matuojamas bazėje b_i. Pagal matavimų rezultatus B sudaro bitų seką $y_1y_2\ldots y_n$.

3. B susisiekia su A nesaugiu kanalu ir nurodo, kokią bazių seką $b_1b_2\ldots b_n$ ji buvo pasirinkusi matavimams.

4. A nurodo, kurie pasirinkimai buvo teisingi.

5. Jeigu $i_1, i_2, \ldots, i_k$ yra teisingai pasirinktų bazių numeriai, tai B sudaro iš atitinkamų bitų seką $\mathbf{y} = y_{i_1}y_{i_2}\ldots y_{i_k}$.

6. A ir B susitaria dėl tam tikro kiekio sekos $\mathbf{y}$ bitų, kuriuos reikia patikrinti: ar $x_{i_j} = y_{i_j}$? Jeigu visi jie sutampa, A ir B nutaria, kad Z nebuvo įsiterpęs į fotonų perdavimo procesą, ir, sutrumpinę $\mathbf{y}$,

išbraukdami tikrinant panaudotus bitus, naudojasi gautuoju žodžiu kaip saugiai perduotu raktu.

A siunčiami bitai	0	1	1	0	1	1	0	1	0	0
A bazės	+	×	×	+	×	+	×	+	×	×
A siunčiami fotonai	↑	↗	↗	↑	↗	→	↘	→	↘	↘
B bazės	+	+	×	+	+	+	×	×	+	×
B registruoti fotonai	↑	↑	↗	↑	→	→	↘	↘	→	↘
B registruoti bitai	0	0	1	0	1	1	0	0	1	0
Bendros bazės	+		×	+		+	×			×
Perduoti bitai	0		1	0		1	0			0
Tikrinami bitai	0			0						
Rakto srautas			1			1	0			0

BB84 protokolo pavyzdys: A saugiai perdavė B keturis bitus.

Kaip į šį protokolą gali įsiterpti Z? Jis gali įrengti savo filtrus ir matuoti fotonų poliarizaciją. Tačiau jis nežino, kokias bazes pasirinko A. Jeigu A fotonui poliarizuoti pasirinko + bazę ir Z pasirinko tą pačią bazę, tai, išmatavęs fotoną, jis pasiųs lygiai taip pat poliarizuotą fotoną, kokį gavo. Taigi tokiu atveju jo įsiterpimas tikrai liks nepastebėtas.

Dabar tarkime, kad A poliarizavo fotoną su + baze ir suteikė jam reikšmę 1. Jeigu Z matavimui naudos × bazę, tai jis gali registruoti tiek 1, tiek 0. Registravęs fotoną, jis lygiai tokį pat pasiųs B. Dabar B eilė pasirinkti bazę. Jeigu B pasirinks × bazę, tai trečiajame protokolo žingsnyje jam bus pasakyta, kad pasirinkimas klaidingas, ir šio žingsnio bitas bus išbrauktas. Taigi Z iš savo užrašų neturės jokios naudos. Jeigu B pasirinko tokią pat bazę kaip A, tai B įtrauks registruotą bitą į gautų bitų žodį. Jo reikšmė bus tokia pati, kokią jam suteikė Z, taigi tikimybė, kad ta reikšmė sutaps su A suteikta reikšme, yra $\frac{1}{2}$. Štai dėl ko reikalingas dalies bitų tikrinimas. Jeigu jo nebūtų, galėtų susidaryti tokia padėtis, kad B registruotų ne tokį pat bitų rinkinį, kokį siuntė A. Žinoma, jeigu tas bitų rinkinys bus naudojamas kaip simetrinės kriptosistemos raktas, apgaulė greitai išryškėtų – šifro, kurį sudarė B, A negalėtų iššifruoti. Tačiau verčiau užbėgti tokiai padėčiai už akių: jeigu bent vienoje tikrinamų bitų poroje jie nesutampa – Z buvo įsiterpęs ir protokolas nedavė rezultato. Jokio rezultato, žinoma, nepasiekė ir Z.

Štai toks protokolas ženklina kvantinės kriptografijos pradžią. Jau ir dabar yra praktinių jos taikymų.

O kas bus toliau? Nekantriai laukiame tęsinio...

26 Pastabos ir nuorodos

Kodavimo teorijai vos penkiasdešimt metų, o štai kriptografija – tūkstantmetė. Išsamiausią jos istoriją parašė D. Kahnas [42]. Daug istorinių faktų, o taip pat klasikinės (ir moderniosios) kriptografijos ir kriptoanalizės idėjų dėstoma F. L. Bauerio knygoje [5]. Kriptografijos istorija įdomiai dėstoma S. Singho knygoje [71]. Jeigu norėtumėte paskaityti apie kriptografijos istoriją lietuviškai – galiu pasiūlyti tik savo knygelę [72].

O dabar nebe apie kriptografijos istoriją, bet apie mokslą. Mokslinių kriptografijos metodų pradininkai – W. Friedmanas ir C. Shannonas. Galima teigti, kad C. Shannono darbu [69] prasidėjo kriptologijos matematinių metodų raida.

Šiuolaikinės kriptologijos uždavinius kelia informacinių technologijų naudojimo praktika. Duomenų perdavimas kompiuteriniais tinklais – štai pagrindinis uždavinių šaltinis. Svarbus įvykis kriptografijoje – pirmojo šifravimo standarto patvirtinimas [58]. DES (Data Encryption Standard) buvo intensyviai tyrinėjamas kone ketvirtį amžiaus, sukurta naujų svarbių kriptoanalizės metodų (skirtuminė kriptoanalizė [9], tiesinė kriptanalizė [49] ir kt.), tačiau šio standarto pakeitimo nauju priežastis – ne surastos kriptosistemos spragos, bet išaugusi skaičiavimo technikos galia. Po ilgai trukusio vertinimo proceso naujasis standartas įsigaliojo 2001 metais [2]. Kriptosistemos autoriai – belgų kriptografai – analizuoja šifro struktūrą savo knygoje [19]. Vykdomi nauji kriptosistemų vertinimo ir standartizavimo projektai, žr. pvz., NESSIE (New European Schemes for Signatures, Integrity and Encryption) projekto medžiagą [59], taip pat srautinių kriptosistemų tyrimo projektą eSTREAM [26].

Kriptologijos straipsniai skelbiami įvairiuose matematikos ir informatikos žurnaluose. Yra keletas specialių leidinių: leidžiamas atskiras kriptologijos istorijai, raidai ir įtakai skirtas žurnalas „Cryptologia“. „Journal of Cryptology“ – tikriausiai pagrindinis žurnalas, skirtas kriptografijos profesionalams. Nuo 2007 metų pradedamas leisti dar vienas leidinys „Journal of Mathematical Cryptology“, tai turėtų būti aukšto teorinio lygio straipsnių žurnalas. Tarptautinė organizacija IACR (International Association for Cryptologic Research) kasmet organizuoja kelias kriptologijai skirtas konferencijas (Crypto, Eurocrypt, Asiacrypt), jų darbai išleidžiami atskirais tomais. Skelbiama daug skaitmeninių publikacijų, pvz., portale „Cryptology ePrint-Archive“ (http://eprint.iacr.org/).

Viešojo rakto kriptosistemų idėjas pirmieji paskelbė W. Diffie, M. E. Hellmanas [22] ir nepriklausomai nuo jų R. Merkle [53]. R. Merkle ir M.E. Hellmanas sukūrė pirmąją viešo rakto kriptosistemą – „kuprinės“ kriptosistemą [54]. Vėliau paaiškėjo, kad jų schema nėra saugi. Viešojo rakto sistemą, kurios saugumo spragų nerasta iki šiol, pirmieji paskelbė R. Rivestas, A. Shamiras, L. Adlemanas [64]. RSA kriptoanalizės rezultatai apžvelgti straipsnyje [11].

1997 metais paskelbti faktai rodo, kad viešojo rakto kriptografijos idėjos kilo anglų slaptosioms tarnyboms dirbusiems kriptografams J. Ellisui, C. Cocksui, M. Williamsonui, žr. pav. [16].

Viešojo rakto kriptosistemoms tik 30 metų. Tačiau tai buvo sparčios raidos metai. Parašyta daug išsamių knygų, skirtų šiuolaikinės kriptografijos raidai. Visų pirma reikia paminėti pirmąjį išsamų naujųjų laikų kriptografijai skirtą B. Schneierio veikalą [70], o taip pat – solidų žinyną [52], kurio skaitmeninės kopijos platinamos laisvai. Parašyta ir vadovėlių, ir specialioms kriptografijos kryptims skirtų monografijų; paminėsime keletą, [18], [32], [46], [45], [79], [55], [73], [60], [74].

Praktiškai diegiant kriptografinius apsaugos metodus, pravers knyga, skirta kriptografinių schemų standartams [20].

Bibliografija

[1] N. Abramson. Information Theory and Coding. New York, McGraw-Hill, 1963.

[2] Advanced encryption standard (AES) FIPS-Pub. 197. November 26, 2001.

[3] M. Agrawal, N. Kayal, N. Saxena. PRIMES is in P. Annals of Mathematics, 160 (2004), p. 781–793.

[4] R. B. Ash. Information theory. New York: Dover 1990.

[5] F. L. Bauer. Decrypted secrets. Springer, 1997.

[6] C.H. Bennett, G. Brassard. Quantum cryptography: Public-key distribution and coin tossing. Proceedings of IEEE International Conference on Computers, Systems and Signal Processing, Bangalore, India, December 1984, p. 175 – 179.

[7] E. Berlekamp. Nonbinary BCH decoding. In: Proc. Int. Symp. on Info. Th. San Remo, Italy, 1967.

[8] C. Berrou, A. Glavieux, P. Thitimajshima. Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes. In: Proc. 1993 IEEE Int. Conf. on Communications, Geneva, Switzerland, 1993, p. 1064–1070.

[9] E. Biham, A. Shamir. Differential Cryptanalysis of the Data Encryption Standard. Springer Verlag, 1993.

[10] J. Bierbrauer, L. Kelly. Introduction to Coding Theory. CRC Press, 2005.

[11] D. Boneh. Twenty Years of Attacks on the RSA Cryptosystem. Notices of the AMS, vol. 46, 2, 1999, p. 203–213.

[12] R. Bose, D. Ray-Chaudhuri. On a Class of Error-Correcting Binary Codes. Inf. and Control, vol. 3, 1960, p. 68–79.

[13] M. Burrows, D. Wheeler. A block-sorting losseless data compression algorithm. Digital Systems Research Center report 124, Palo Alto, 1994.

[14] A. R. Calderbank. The art of signaling: fifty years of coding theory. IEEE Trans. Information Theory, 1998, IT-44 (6), p. 2561–2595.

[15] P. J. Cameron, J. H. van Lint. Graphs, codes and designs. Cambridge University Press, 1980.

[16] C. Cocks. A Note on Non-Secret Encryption. 1973. http://www.cesg.gov.uk/site/publications/index.cfm

[17] T. Cover, J. Thomas. Elements of Information Theory. New York: Wiley-Interscience, 1991, 2006.

[18] R. Crandall, C. Pomerance. Prime numbers. A computational perspective. Springer, 2001.

[19] J. Daemen, S. Borg, V. Rijmen. The Design of Rijndael: AES - The Advanced Encryption Standard. Springer-Verlag, 2002.

[20] A. Dent, C. Mitchell. User's Guide to Cryptography and Standards. Artech House, 2004.

[21] S. X. Descamps. A computational primer on block error-correcting codes. Springer, 2003. http://www.wiris.com/cc/

[22] W. Diffie, M. E. Hellman. New directions in Cryptography. IEEE Transactions on Information Theory, v. IT-22, n. 6, 1976, p. 644–654.

[23] Duomenų spūda.
Tinklalapis: http://www.data-compression.info/index.htm

[24] Elektroninis žurnalas „Entropy". http://www.mdpi.org/entropy/

[25] P. Elias. Coding for Noisy Chanels. IRE Conv. Rep., Pt. 4, p. 37–47, 1955.

[26] eSTREAM. http://www.ecrypt.eu.org/stream/

[27] N. Faller. An adaptive system for data compression. Record of the 7th Asilomar Conference on Circuits, Systems and Computers, p. 593–597.

[28] R. Gallagher. Variations on a theme by Huffman. IEEE Transactions on Information Theory, 1978, IT-24 (6), November, p. 668–674.

[29] R. Gallagher. Claude E. Shannon: A retrospective on his life, work, and impact. IEEE Trans. Information Theory, 2001, IT-47, p. 2681–2695.

[30] M. R. Garey, D. S. Johnson. Computers and Intractability. A Guide to the Theory of NP-Completeness, W. H. Freeman and Co., 1979.

[31] M. J. E. Golay. Notes on digital coding. Proceedings of the IRE, June 1949.

[32] O. Goldreich. Modern Cryptography, Probabilistic Proofs and Pseudorandomness. Springer, 1999.

[33] GAP - Groups, Algorithms, Programming. http://www-gap.mcs.st-and.ac.uk/

[34] R. M. Gray. Entropy and Information Theory. 2002. http://www-ee.stanford.edu/% gray/it.html

[35] R. W. Hamming. Error detecting and error correcting codes. The Bell System Technical Journal, 1950, No. 2, April, p. 147–160.

[36] R. W. Hamming. Coding and Information Theory. Prentice Hall, 1986.

[37] D. Hankersson et al. Introduction to Information Theory and Data Compression. CRC Press, 1997.

[38] A. Hockuenghem. Codes Correcteurs D'erreurs. Chiffres, vol. 2, 1959, p. 147–156.

[39] D. Huffman. A method for the construction of minimum redundancy codes. Proceedings of the IRE. 1952, 40(9), p. 1098–1101.

[40] W. C. Huffman, V. S. Pless. Fundamentals of error-correcting codes. Cambridge University Press, 2003.

[41] R. V. L. Hartley. Transmission of information. The Bell System Technical Journal, 7(3), 1928, p. 535–563.

[42] D. Kahn. The codebreakers. Scribner, 1967, 1996.

[43] G. J. Klir. Uncertainty and information. Foundations of Generalized Information Theory. Wiley&sons, 2006.

[44] D. E. Knuth. Dynamic Huffman coding. Journal of Algorithms. 1985, 6, p. 163–180.

[45] N. Koblitz. A course in number theory and cryptography. Springer, 1987.

[46] N. Koblitz. Algebraic Aspects of Cryptography. Springer, 1998.

[47] R. Lassaigne, M. Rougemont. Logika ir algoritmų sudėtingumas. Žara, Vilnius 1999.

[48] J. H. van Lint. Introduction to Coding Theory. Springer, 1982, 1999.

[49] M. Matsui. Linear Cryptanalysis Method for DES Cipher. EUROCRYPT 1993, p. 386–397.

[50] R. J. McEliece. The Theory of Information and Coding. Cambridge University Press, 2002.

[51] D. J. C. MacKay. Information Theory, Inference and Learning Algorithms. Cambridge University Press, 2003. http://www.inference.phy.cam.ac.uk/mackay/

[52] A. J. Menezes, P. C. van Oorshot, S. A. Vanstone. Handbook of applied cryptography. CRC Press, 1997. http://www.cacr.math.uwaterloo.ca/hac/

[53] R. C. Merkle. Secure communication over insecure channels. Communications of the ACM, v. 21, n. 4, 1978, p. 294–299.

[54] R. C. Merkle, M. Hellman. Hiding information and signatures in trapdoor knapsacks. IEEE Transactions on Information Theory, v. IT-24, n. 5, 1978, p. 525–530.

[55] R. A. Mollin. RSA and public-key cryptography. Chapman&Hall CRC, 2003.

[56] T. K. Moon. Error Correction Coding. Mathematical methods and algorithms. Wiley, 2005.

[57] D. Muller. Application of Boolean Switching Algebra to Switching Circuit Design. IEEE Trans. on Computers, vol. 3, Sept. 1974, p. 6-12.

[58] National Bureau of Standards, Data Encryption Standard, FIPS-Pub. 46. National Bureau of Standards, U. S. Department of Commerce, Washington D. C., January 1977.

[59] NESSIE, https://www.cosic.esat.kuleuven.be/nessie/

[60] R. Oppliger. Contemporary cryptography. Artech House, 2005.

[61] E. Prange. Cyclic Error-Correcting Codes in Two Symbols. Air Force Cambridge Research Center, Cambridge, MA, Tech. Rep. TN-58-156, 1958.

[62] I. Reed. A Class of Multiple-Error-Correcting Codes and a Decoding Scheme. IEEE Trans. Information Theory, vol. 4, Sept. 1954, p. 38–49.

[63] I. Reed, G. Solomon, Polynomial Codes over Certain Finite Fields. J. Soc. Indust. Appl. Math., vol. 8, 1960, p. 300-304.

[64] R. Rivest, A. Shamir, L. Adleman. A method of obtaining digital signatures and public-key cryptosystems. Communications of the ACM, v. 21, n. 2, 1978, p. 120–126.

[65] S. Roman. Coding and Information Theory. Springer, 1992.

[66] A. Rukhin et al. A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22, 2001. http://csrc.nist.gov/rng/SP800-22b.pdf

[67] D. Salomon. Data compression. The complete Reference. Springer, 1988, 2000, 2004.

[68] C.E. Shannon. A Mathematical Theory of Communication, The Bell System Technical Journal, XXVII, 1948, N. 3.

[69] C. E. Shannon. Communication theory of secrecy systems. Bell Syst. Tech. J. 28 (1949), p. 656-715.

[70] B. Schneier. Applied Cryptography. Willey, New York 1994.

[71] S. Singh. The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography. Anchor Books, 2000.

[72] V. Stakėnas. Šifrų istorijos. TEV, 2005.

[73] W. Stallings. Cryptography and Network Security. Principles and Practices, Fourth Edition, Prentice Hall, 2005.

[74] D. Stinson. Cryptography Theory and Practice. CRC Press, 1995, 2002, 2005.

[75] P. Sweeney, Error Control Coding: From Theory to Practice. John Wiley and Sons, 2002.

[76] H. van Tilborg. Fundamentals of cryptology. Kluwer, 2002.

[77] A. J. Viterbi. Convolutional Codes and Their Performance in Communication Systems. IEEE Trans. Com. Techn., vol. 19, n. 5, 1971, p. 751–771.

[78] S. Verdu. Fifty years of Shannon theory. IEEE Trans. Information Theory, 1998, IT-44 (6), p. 2057–2078.

[79] S. Wagstaff. Cryptanalysis of number theoretic ciphers. Chapman&Hall CRC, 2002.

[80] J. Ziv, A. Lempel. A universal algorithm for sequential data compression. IEEE Transactions on Information Theory, 1977, IT-24(5), p. 530–536.

Rodyklė

Įvertis
    Griesmerio, 168

Abipusės informacijos kiekis, 19
Algoritmas
    Euklido, 286
    kėlimo laipsniu moduliu, 289
    polinominio laiko, 278
    tikimybinis, 284
    Viterbi, 197
Ataka
    „gimtadienio“ , 337
    įsiterpimo, 207
    pavienių šifrų, 209
    adaptyvi pasirinktų teksto-šifrų porų, 209
    pasirinktų šifrų, 209
    pasirinktų teksto-šifrų porų, 209
    RSA bendro modulio, 312
    RSA mažo privačiojo rakto, 310
    teksto-šifrų porų, 209
    vidurio, 351

Bazė
    tiesinio poerdvio, 118

Daugianario svoris, 173
Daugianaris
    minimalusis, 129
    neskaidus, 121
    primityvusis, 128
    tiesinių registrų sistemos, 267
Diagrama
    entropijų sąryšių, 19
    kanalo, 67
    sąsūkų kodo būsenų, 196
    sąsūkų kodo grotelių, 197
Diskretusis logaritmas, 299
Dualus
    poerdvis, 119

Ekvivalentūs kodai, 104
Elementarieji pertvarkiai, 133
Elemento eilė, 126
Entropija
    atsitiktinio dydžio, 11
    Bernulio šaltinio, 20
    informacijos šaltinio, 15
    kalbos, 28
    pirmosios eilės Markovo šaltinio, 22
    sąlyginė, 16
    stacionaraus šaltinio, 25

Funkcija
    Eulerio, 290
    maišos aritmetinė, 341
    maišos atspari sutapimams, 336
    maišos iš blokinių kriptosistemų, 339
    maišos labai atspari sutapimams, 336
    maišos SHA-1, 339
    vienakryptė, 335

Gauso dėsnis, 294
Generuojantis elementas, 127

Hammingo atstumas, 77, 94

Išsprendžiamumo uždavinio kalba, 275
Idealas, 176
Įrodymas, nesuteikiantis žinių, 351

grafų izomorfiškumo, 352
Įvertis
Gilberto–Varšamovo, 101
Hammingo, 101
Singletono, 101

Jeffersono ritinys, 228

Kalbos pertekliškumas, 28
Kanalas
be atminties, 66
dvejetainis simetrinis, 67
simetrinis, 68
trinantis, 68
Kanalo talpa, 69
Kerckhoffo sąlygos, 229
Klaidų pliūpsnis, 190
Kodų sandauga, 166
Kodas, 30, 74
aritmetinis, 51
ASCII, 34
Baudot, 34
BCH, 184
Braille, 33
Burrowso-Wheelerio, 63
ciklinis, 177
daugianarių, 173
dekoduojamas, 35
dualus, 132
Fano, 45
Fire, 191
Golay, 150
Golombo, 57
Hadamardo, 110
Hammingo, 143
Huffmano, 46
Huffmano adaptyvus, 54
klaidas randantis, 95
klaidas taisantis, 95
Lempelio-Zivo, LZ78, 60
liekanų, 167
maksimalaus atstumo, 146
maksimalus, 100
momentinis, p-kodas, 36
Morse, 32
optimalus, 39
Polibijaus, 32
Reedo-Mullerio, 155
Reedo-Solomono, 147
Reedo-Solomono apibendrintas, 181
sąsūkų, 195
savidualus, 132
Shannono, 43
simplekso, 144
su kontroliniu simboliu, 136
sumažintas, 162
sutrumpintas, 161
tiesinis, 132
tobulas, 98
Unicode, 35
Kodo
dengimo spindulys, 97
koeficientas, 74
medis, 31
minimalus atstumas, 95
pakavimo spindulys, 97
plėtinys, 161
svorių skirstinys, 169
tandartinė lentelė, 139
Konstrukcija
uIu+v, 163
Kriptoanalizė, 208
Kriptografija, 202
Kriptografinis protokolas, 208
Kriptologija, 202
Kriptosistema, 203
įrodyto saugumo, 210
AES, 249
besąlygiškai saugi, 210, 236
Blumo-Goldwasserio tikimybinė, 316
DES, 247
ElGamalio, 318
kuprinės, 307
Massey-Omura, 313
McElliece'o, 320
Pohligo-Hellmano, 312
Rabino, 315

RSA, 308
saugi skaičiavimų požiūriu, 210
saugi sudėtingumo teorijos požiūriu, 210
saugi *ad hoc*, 210
simetrinė, 203
srautinė, 259
viešojo rakto, 204
Kroneckerio sandauga, 108

Legendre'o simbolis, 293

Maišos funkcijos sutapimas, 336
Matrica
generuojanti, 132
Hadamardo, 106
kanalo tikimybių, 66
kontrolinė, 136
Metodas
Pollardo ρ faktorizacijos, 296
Pollardo ρ logaritmo radimo, 301
Pollardo $p-1$, 298
Shankso, 300
steganografinis, 362

Nelygybė
Krafto-McMillano, 38

Paley konstrukcija, 109
Pasiskirstymas
asimptotiškai tolygus, 21
Protokolas
Diffie-Hellmano rakto nustatymo, 350
diskretaus logaritmo žinių įrodymo, 353
elektroninių rinkimų, 357
Fiato-Shamiro, 353
kvantinis bitų perdavimo, 364
monetos metimo, 358
pokerio, 359
Pseudoatsitiktinė Golombo seka, 260

Rakto įminimo taškas, 243
Režimas
šifrų blokų grandinės, 254
šifro atgalinio ryšio, 255
skaitiklio, 257
srauto atgalinio ryšio, 256
Rotoriai, 232
Rutulio tūris, 97

Schema
DSA skaitmeninio parašo, 331
ECash skaitmeninių pinigų, 355
ElGamalio skaitmeninio parašo, 326
Feistelio, 246
keitinių-perstatų tinklo, 245
nepaneigiamo skaitmeninio parašo, 331
paslapties dalijimo
Asmutho-Bloomo, 345
Blakely, 345
Brickelio, 348
pagal leidimų struktūrą, 347
Shamiro, 342
Rabino skaitmeninio parašo, 325
RSA aklo parašo, 324
RSA skaitmeninio parašo, 323
Shnorro skaitmeninio parašo, 328
skaitmeninių parašų, 205, 321
Sindromas, 140
Slaptasis kanalas parašo schemoje, 334
Statistinis testas
autokoreliacijos, 264
bitų porų, 263
blokų, 264
pavienių bitų, 263
pokerio, 263
Šaltinis
be atminties, 13
Bernulio, 13
Šaltinis
informacijos, 13
Šaltinis
Markovo, 13
Šaltinis
Markovo m-osios eilės, 14

Sutapimų indeksas, 225
Šifras
    A5/1, 272
    blokinis, 245
    Bluetooth E0, 273
    Cezario, 215
    Fleissnerio kvadratų, 213
    Hillo, 216
    homofonų, 216
    skytalės, 212
    Vernamo, 231
    Vigenere, 222

Taisyklė
    dekodavimo, 75
    didžiausio tikėtinumo, 76
    entropijų grandinės, 17
    idealaus stebėtojo, 76
    kodavimo, 29
    minimalaus atstumo, 77, 94
Tapatybė
    McWilliams, 169
Teorema
    Fermat, 291
    kiniškoji liekanų, 295
    Shannono atvirkštinė, 88
    Shannono dvinariam kanalui, 79
Testas
    Friedmano kappa, 224
    Kassiskio, 222
    Millerio-Rabino, 285
Tiesinė erdvė, 117
Tiesinių registrų sistema, 265
Tiesinis poerdvis, 117
Turingo mašina, 276

Uždavinys
    NP klasės, 280
    NP pilnasis, 281
    P klasės, 278
    išsprendžiamumo, 274
    kuprinės, 305