KAUNO TECHNOLOGIJOS UNIVERSITETAS

MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

Vytautas Poderis

Matricinio laipsnio šifro funkcijos sprendinių aibės tyrimas

Magistro darbas

Vadovas
prof. dr. Eligijus Sakalauskas

KAUNAS, 2014

KAUNO TECHNOLOGIJOS UNIVERSITETAS

MATEMATIKOS IR GAMTOS MOKSLŲ FAKULTETAS

TAIKOMOSIOS MATEMATIKOS KATEDRA

TVIRTINU
Katedros vedėjas
doc. dr. N.Listopadskis

Matricinio laipsnio šifro funkcijos sprendinių aibės tyrimas

Taikomosios matematikos magistro baigiamasis darbas

Vadovas
(parašas) prof. dr. E.Sakalauskas
2014 06 01

Recenzentas
doc. dr. A.Aleksa
2014 06 01

Atliko
FMMM-2 gr. stud.
(parašas) V. Poderis
2014 05 30

KAUNAS, 2014

KVALIFIKACINĖ KOMISIJA

Pirmininkas: Juozas Augutis, profesorius (VDU)

Sekretorius: Eimutis Valakevičius, profesorius (KTU)

Nariai:

Arūnas Barauskas, dr., direktoriaus pavaduotojas (UAB „Danet Baltic“)

Vytautas Janilionis, docentas (KTU)

Zenonas Navickas, profesorius (KTU)

Kristina Šutienė, docentė (KTU)

Jonas Valantinas, profesorius (KTU)

Poderis V. Analysis of matrix power cipher function sets of solutions: Master‘s work in applied mathematics / supervisor dr. prof. Sakalauskas E.; Department of Applied mathematics, Faculty of Fundamental Sciences, Kaunas University of Technology. – Kaunas, 2014. – 67 p.

SUMMARY

Cryptography is the practice and study of techniques for secure communication in the presence of third parties. More generally, it is about constructing and analyzing protocols that overcome the influence of adversaries and which are related to various aspects in information security such as data confidentiality, data integrity, authentication, and non-repudiation. In cryptography, a cipher (or cypher) is an algorithm for performing encryption or decryption - a series of well-defined steps that can be followed as a procedure.

Cryptanalysis is the study of analyzing information systems in order to study the hidden aspects of the systems. Cryptanalysis is used to breach cryptographic security systems and gain access to the contents of encrypted messages, even if the cryptographic key is unknown, i.e., it is the study of how to crack encryption algorithms or their implementations. To ensure that confidentiality is robustly provided, it is essential to investigate the security of a new matrix power cipher against a cryptanalytic attacks.

In this paper we present the implementation analysis of the results of cryptanalytic attack to the matrix power cipher. The investigation of the power cipher function sets of solutions shown that it is possible to group all the possible secret keys into different groups. Hence the set of the all keys could be reduce by the number which is numerable from the parameters of cipher.

The main contribution of this work is the secret key clone’s theorem. It proves that there are a number of key clones which are able to work in a cipher as well as the main key. It’s changed the safe level of parameters which provide the unbreakable cipher on reasonable time. At the end of this paper there are a several kind of code for future detailed research.

TURINYS

LENTELIŲ SĄRAŠAS

PAVEIKSLŲ SĄRAŠAS

ĮVADAS

Esame tiesiogiai ir betarpiškai priklausomi nuo elektroninės terpės saugumo. Ženkliai augant technologijų ir mokslo raidai, sąlyginai visa asmeninė, juridinė ar visuomeninė informacija yra saugoma ar dubliuojama elektroniniu būdu. Šia informacija yra nuolatos dalinamasi koregavimo, atnaujinimo ar kitais tikslais. Eilę metų iš eilės informacijos saugumo incidentų skaičius nuolatos auga, nors daugumoje atvejų nutekinta informacija galbūt ir nėra ypatingai reikšminga, tačiau į tai būtina reaguoti.

Šiandien naudojamoms informacinio saugumo priemonėms dažnai trūksta integracijos, jos veikia tarsi atskiros programos. Tokia apsauga dažniausiai yra neveiksminga, nes konfidencialios informacijos vagysčių atvejai yra sudėtingi ir įgyvendinami naudojant daugelį metodų.

Papildoma duomenų saugumo priemonė – šifravimas. Šifravimo pagalba yra užtikrinamas informacijos perdavimo saugumas tarp dviejų šalių. Egzistuoja daugybė skirtingų šifrų, kuriuos galime skirstyti pagal saugumą, optimalumą, paprastumą ar pagal tai kokie matematiniai uždaviniai yra sprendžiami. Jie suteikia skirtingas saugumo ir efektyvumo galimybes. Plačiausiai yra naudojami simetriniai šifrai, kuriuose informacijos siuntėjai ir gavėjai naudoja tą patį slaptąjį užšifravimo ir iššifravimo raktą.

Natūralu, kad populiariausi šifrai sulaukia daugiausia dėmesio ir yra nuolat išmėginami, t.y. ne tik saugumo kūrėjų pastangomis, bet ir kenkėjų atakomis. Nėra garantuota, kad plačiai naudojami šifrai yra visiškai saugūs, tiesiog nėra paviešinti metodai ar sistemos šiems šiframs sukompromituoti. Todėl atsižvelgus į naujausius kriptoanalizės metodus, siekiama sukurti naujus šifrus, užtikrinančius nepaneigiamą saugumą.

Naujų kriptografinių algoritmų kūrimas apima dvi šakas – kriptografiją (sistemų ir protokolų kūrimas) ir kriptoanalizę (mėginimas sukompromituoti šifrą). Tinkamai atlikta kriptoanalizė padeda rasti šifro trūkumus. Šifras yra korektiškas, kai bet kokius duomenis yra galima užšifruoti ir teisingai iššifruoti su tuo pačiu pasirinktu slaptuoju raktu, arba paprastu būdu išskaičiuojamu raktu. Tai būtinas kiekvieno šifro reikalavimas.

Kiekvienu unikaliu atveju šifro stiprumas turi būti adekvatus informacijos svarbai. Logiška, jog sudėtingesnis šifras sunaudos daugiau sistemos resursų, veiksmai užims daugiau laiko. Taigi optimalus turimų resursų paskirstymas pagal poreikį ir galimybes priverčia surasti ribą, su kokiais minimaliais parametrais šifras bus tinkamas, t.y. per atitinkamą laiką nebus paviešinta informacija.

Šiuo darbu siekiama išnagrinėti naujo Matricinio laipsnio (ML) šifro saugumą atliekant kriptoanalizę. Algebrinė kriptoanalizė prieš ML šifrą yra daugiau nei pakankamai sudėtinga, tačiau net sėkmingai įvykdyta ataka – išspręstas suformuotas uždavinys, negarantuoja šifrogramos atskleidimo. Todėl atliekamas sprendinių aibės tyrimas, siekiant išsiaiškinti, ko galima tikėtis atlikus algebrinę kriptoanalizę, pilno perrinkimo ataką ir pan. Taikant pilno perrinkimo ideologiją, ištiriamos galimų sprendinių aibės, iš gautų rezultatų nustatomos šifro savybės.

Darbo tikslas ir uždaviniai

Pagrindinis darbo tikslas – ištirti Matricinio laipsnio šifro funkcijos sprendinių aibę.

Suformuluotam tikslui buvo sprendžiami šie uždaviniai:

1. Suformuojamas ir aprašomas kriptoanalizės atakos planas.
2. Įvykdoma pilno perrinkimo ataka.
3. Ištiriama gautoji sprendinių aibė.
4. Iškeliamos hipotezės dėl sprendinių aibės savybių.
5. Tiriami atakos metu gauti duomenys ir jų analizės rezultatai.
6. Suformuojami teiginiai, savybės ir teoremos pagal gautus rezultatus.
7. Nustatomos šifro savybės.

Tyrimų metodika ir struktūra

Tyrimui atlikti programinės įrangos pagalba yra imituojamas pilnas galimų rezultatų perrinkimas. Jo metu sukaupti duomenys yra analizuojami ir jiems iškeliamos hipotezės. Tyrimas pristatomas nuosekliais punktais, tai sukuria analizės eigos tęstinumą, leidžia pagrįsti kiekvienos hipotezės kilmę. Tyrimo metu iš gautų rezultatų suformuojami teiginiai, savybės ir teorema.

1 BENDROJI DALIS

1.1 ŠIFRAVIMO ĮDĖJOS IR SISTEMŲ METODAI

Kriptologija – tai mokslas apimantis duomenų slėpimą, apsaugojimą ir saugų dalinimąsi jais. Pagrindinės kryptys – kriptografija ir kriptoanalizė. Kriptografijos tikslas sukurti kriptografinių sistemų schemas, protokolus ir paruošti juos naudojimui.

Kriptoanalizės – surasti algoritmų trūkumus, silpnas vietas, kurios leistų atstatyti ar suklastoti duomenis, nežinant saugumą užtikrinančių sistemų parametrų. Kiekvienas naujai kuriamas šifras turi būti kuo detaliau išnagrinėtas abiejų šakų, pasitelkiant žinomus metodus, problemas.

Duomenų privatumui užtikrinti yra naudojamas informacijos šifravimas. Šifravimas – tai būdas atvirą informaciją paversti uždara ir atvirkščiai. Šifruoti galima praktiškai bet kokią informaciją, tiek duomenų bylas, tiek atskirus duomenų bazės įrašus. Taikoma ir asmeniniams, ir visiškai slaptiems duomenims apsaugoti nuo nesankcionuoto panaudojimo.

Pagrinde visas šifravimo sistemas galima suskirstyti į dvi pagrindines grupes – simetrinius ir asimetrinius šifrus. Simetrinėse kriptografinėse sistemose, duomenų užšifravimui ir iššifravimui naudojamas tas pats slaptasis raktas, arba jis gali būti paprastai išskaičiuojamas. Asimetrinėse sistemose visuomet yra slaptųjų raktų pora, kurią sudaro matematiškai susiję privatusis ir viešasis raktai. Vienas iš šių raktų gali būti viešai paskelbiamas. Tekstogramą užšifravus viešuoju raktu iššifruoti bus įmanoma tik naudojant privatųjį raktą.

Būtent ši savybė asimetrinius šifrus ir išpopuliarino, tačiau jie turi ir trūkumų - šie šifrai yra keliomis eilėmis lėtesni už simetrinius ir nėra labai praktiški dideliems duomenų kiekiams šifruoti. Taigi simetrinio rakto kriptografija yra greitesnė ir paprastesnė, o patys raktai - trumpesni. Vienas akivaizdžiausių simetrinio šifro trūkumų yra – rakto perdavimas duomenų gavėjui, kuris turi būti betarpiškai saugus. Raktų perdavimas nėra triviali problema.

Matricinio laipsnio šifro algoritmas kurtas laikantis A.Kerckhoffs principu (A.Kerckhoffs, XIX a.), kuris teigia, kad kriptografinė sistema turi būti saugi, net jei apie ją yra žinoma viskas, išskyrus jos slaptąjį raktą. Šio principo galiojimas leidžia plačiau nagrinėti saugumo spragas, suteikdamas daugiau informacijos fiktyviai atakai, nuolatos tobulinti patį šifrą tretiesiems asmenims. Tačiau pasaulinėje praktikoje fiksuojami atvejai, kai slapti kriptografiniai algoritmai yra naudojami įvertinus juos organizaciniai priklausančių kriptoanalitikų grupių atliktu darbu. Remiantis minėtu principu, darbe atliekama Matricinio laipsnio šifro kriptoanalizė.

1.2 SIMETRINIO ŠIFRO SISTEMOS

Simetriniai šifrai yra de-facto naudojami didelių duomenų masyvų ar srautų šifravimui. Savo ruožtu yra skiriamos dvi simetrinių šifrų grupės: srautiniai ir blokiniai.

Srautiniai šifrai informaciją šifruoja po vieną bitą (arba baitą) ir neturi jokių apribojimų duomenų ilgiui. Tai leidžia šiuos šifrus panaudoti realaus laiko duomenų srautų šifravimui įvedant minimalius pavėlinimus.

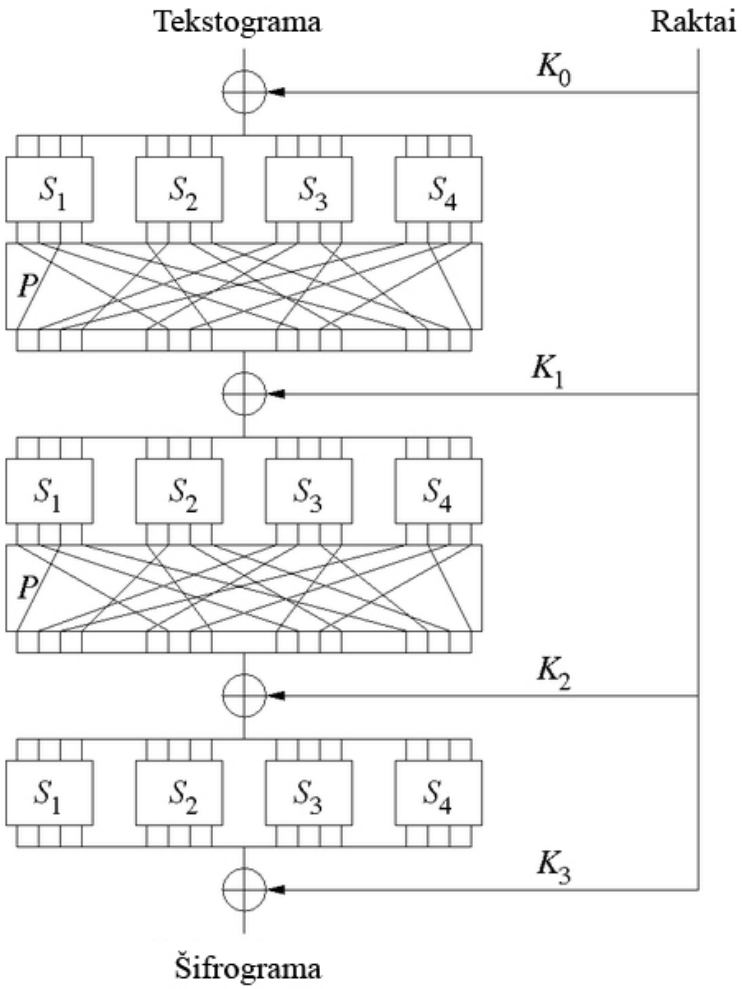
Blokiniai šifrai operuoja fiksuoto ilgio duomenų blokais - jiems veikti reikia turėti duomenų bloką, kuris ir bus šifruojamas ar iššifruojamas. Naudojant blokinius šifrus realaus laiko duomenims šifruoti sukuriamas vėlinimas, kurio metu duomenys kaupiami buferyje, šifruojami ir išsiunčiami tolesniam apdorojimui. Taigi blokiniai šifrai labai gerai tinka didelės apimties failams šifruoti.

Remiantis paprasčiausiais klasikiniais šifrais: sukeitimo (*angl*. substitution) šifru, kai kiekvienas simbolis ar jų kombinacija yra pakeičiama atitinkamu kiekiu kitų simbolių ir tam sudaroma šifrų knygos ar sukeitimo lentelės ir perstatymo (*angl*. transposition) šifru, kai tekstogramos simboliai ar jų grupės yra sukeičiamos vietomis su kitais simboliais ar grupėmis pagal nurodytas taisykles, yra sudaromi sukeitimų ir perstatymų tinklai.

Sukeitimų ir perstatymų tinklai buvo pasiūlyti C.Shannon (C.Shannon, 1949) siekiant išvengti kriptoanalizės, pagrįstos statistine analize. O pagal korektiško šifro principus, geras šifras turi visiškai paslėpti tekstogramos statistines savybes. Remiantis tinklais, užtikrinama šifruojamos informacijos sumaišymas ir paskleidimas. Paskleidimo mechanizmu siekiama, kad vieno tekstogramos bito pokytis turėtų įtakos visiems šifrogramos bitams. Sumaišymų siekiama ryšius tarp šifrogramos statistinės informacijos ir šifravimo rakto padaryti kuo sudėtingesnius.

Sukeitimų ir perstatymų tinklai (SP tinklai) sudaryti iš kelių sukeitimų (S blokų) ir perstatymo blokų (P blokų) lygių, einančių vienas paskui kitą. Siekiant pasiekti geresnių sumaišymo rezultatų pravartu derinti sukeitimų blokus su perstatymo blokais. Sukeitimai duomenis sumaišo lokaliai, o perstatymai juos sujungia ir paskleidžia.

S blokas, užtikrinantis duomenų sumaišymą, laikomas optimaliu, kai pakitus vienam įvesties bitui, vidutiniškai pasikeičia ne mažiau kaip pusė išvesties bitų. Taigi kiekvienas išvesties bitas turi priklausyti nuo visų įvesties bitų. Tuo tarpu P blokas išsklaido S bloko atliktą lokalų išskaidymą. Pateikiamas elementarus 16 bitų tekstogramos 3 iteracijų užšifravimo pavyzdys leisiantis lengviau suprasti pačią idėją:



1.1. pav. Suketimo ir perstatymo blokai

Žinomiausi blokiniai šifrai:

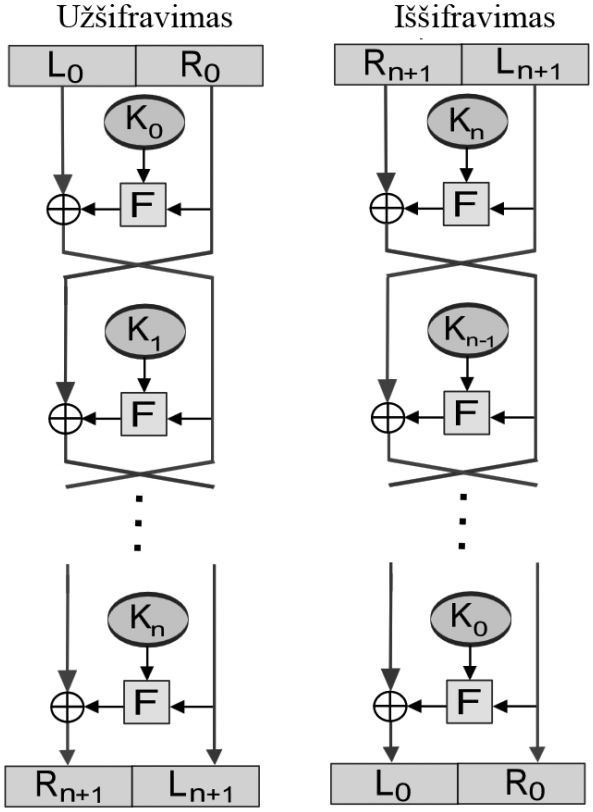
- *Data Encryption Standard (DES) / IBM 1977*
- *International Data Encryption Algorithm (IDEA) / Xuejia Lai and James Massey 1991*
- *RC5 / Ron Rivest 1994*
- *Advanced Encryption Standard (AES) / Vincent Rijmen, Joan Daemen 1998*
- *Blowfish / Bruce Schneier 1993 ir kt.*

Sukeitimo ir perstatymo tinklais bei H.Feistel šifru dažniausiai remiasi nauji šiuolaikiniai šifrai, kurie yra kuriami absorbuojant jau išanalizuotas klasikines architektūras. H.Feistel šifro idėja – blokinio šifro architektūra, kurioje duomenys dalinami į dvi lygias dalis ir vienoje iteracijoje pertvarkomi tik vienos dalies duomenys. Tai palengvina blokinio šifro sudarymą naudojant įvairias šifravimo funkcijas F. Svarbiausia šio šifro savybė ta, kad visada gaunamas abipusiškai vienareikšmis atvaizdis, kad ir kokia būtų iteracinė funkcija.

H.Feistel šifruose sukeitimą sudaro duomenis paveikianti funkcija F, kuri palyginus su sukeitimų ir perstatymų tinklų schema, atitinka S bloką priklausantį nuo slaptojo rakto. Ši funkcija turi būti kriptografiškai saugi ir kokybiškai paslėpti tekstogramos ir šifravimo rakto statistinius požymius, o jos

apverčiamumas, vienkryptiškumas, vienareikšmiškumas ar kitos savybės nėra būtinos. Be to funkcijai nesant vienareikšmei, vis tiek užtikrinamas vienareikšmis šifrogramos iššifravimas. Perstatymą (P bloką) sudaro gauto S bloko rezultato sukeitimas su dešiniąja duomenų dalimi.

Pateikiamas H.Feistel šifro pavyzdys:



1.2.pav. H.Feistel šifro pavyzdys

čia F – iteracinė funkcija, K – raktai, L_n ir R_n tai du S blokai.

Yra nustatyta: pakanka trijų iteracijų, kad H.Feistel šifras su kriptografiškai saugia iteracine funkcija F tarptų pseudo-atsitiktiniu keitiniu. Šifruojant keturiomis ir daugiau iteracijų, šifras tampa „stipriu“ pseudoatsitiktiniu keitiniu (Luby & Rackoff, 1988). Vėlesniuose darbuose buvo įrodyta, kad H.Feistel šifro saugumui užtikrinti pakanka septynių iteracijų (Patarin, 2003).

Kai kurios H.Feistel šifro modifikacijos sėkmingai realizuotose kituose šifruose:

- Dalijimą pusiau galima pakeisti dalijimu į keturias, aštuonias ar daugiau dalių.
- Vietoje sudėties moduliu 2 galima naudoti kokią nors kitą operaciją. Vienintelė būtina šios operacijos sąlyga yra reguliarumas, o komutatyvumas ir asociatyvumas nebūtini.
- Duomenų bloką galima padalinti į skirtino dydžio dalis.
- Į H.Feistel šifro schemą galima įtraukti papildomus apverčiamus S blokus.

Tačiau bendras šifravimo sistemos saugumas priklauso ne tik nuo paties algoritmo. Skirtingoms to paties tekstogramos bloko šifrogramos gauti blokiniai šifrai gali būti naudojami įvairiais režimais. Išskiriami penki pagrindiniai blokinių šifrų šifravimo rėžimai (Oppliger, 2005):

- Elektroninės šifrų knygos rėžimas (*angl*. Electronic codebook – ECB): pranešimo blokai šifruojami nepriklausomai vienas nuo kito.
- Šifro bloko grandininis režimas (*angl*. Cipherblock chaining – CBC): pirmiausia pranešimo bloko bitai sudedami moduliu 2 su ankstesnio etapo šifravimo rezultato bitais, ir tik tuomet šifruojami.
- Grįžtamojo ryšio šifro režimas (*angl*. Cipher feedback – CFB): pagal slaptąjį raktą ir jau užšifruotus blokus generuojama pseudoatsitiktinių bitų seka ir sumuojama moduliu 2 su pranešimu.
- Grįžtamojo ryšio išvesties režimas (*angl*. Output feedback – OFB): pagal raktą ir jau sugeneruotus pseudoatsitiktinius bitų blokus generuojama bitų seka ir sudedama moduliu 2 su pranešimu.
- Skaitiklio režimas (*angl*. Counter – CTR): pagal raktą ir skaitiklį (bloko indeksą) generuojama pseudoatsitiktinių bitų seka ir sudedama moduliu 2 su pranešimu.

ECB atveju kiekvienas blokas šifruojamas atskirai ir kenkėjas gali atkurti prasmingą tekstogramą iš atskirų žinomų tekstogramos ir šifrogramos blokų porų dėl perduodamos tam tikros informacijos, todėl šis režimas laikomas mažiausiai saugus. Siekiant eliminuoti problemas ir buvo sukurti paskesni režimai, kurie skiriasi pagal spartumą, tikslumą.

1.3 PAGRINDINĖS MATEMATINĖS PRIEMONĖS

1.3.1 ALGEBRINĖS STRUKTŪROS

Beveik visų algebrinių struktūrų algoritmų realizuotų kompiuteriuose operacijos yra atliekamos bitais (elementarus informacijos matavimo vienetas), todėl su tikslu įdarbinti procesorius maksimaliai, naudojamasi algebrinėmis struktūromis. Pagrindinės kriptografijoje naudojamos algebrinės struktūros yra baigtiniai žiedai Z_n ir laukai $GF(p^n)$. Tačiau veiksmams tarp elementų apibrėžti moduline sveikųjų skaičių sudėtimi ir daugyba bendru atveju nėra tinkama.

Kriptografijoje pagrinde naudojamas daugianarių liekanų laukas $Z_p[x]/(f_n(x))$, kur $f_n(x)$ yra n-tos eilės neredukuojamas daugianaris, t.y. lauko $GF(p^n)$ elementai yra vieno kintamojo daugianariai ir jų koeficientai apibrėžti virš Z_p, o visų daugianarių eilė neviršija $n-1$. Sudėties ir daugybos operacijos tarp daugianarių yra apibrėžiamos pasirinkto tinkamo neredukuojamo daugianorio $f_n(x)$ moduliu.

Sudėties operacija tarp daugianarių vykdoma pagal Z_p taisykles sudedant daugianarių koeficientus. Daugybos operacija vykdoma elementariai, kiekvienas iš daugianarių elementų yra

dauginamas iš visų kito daugianario elementų paeiliui, gauti rezultatai sudedami sutraukiant panašiuosius narius ir apskaičiuojama liekana moduliu neredukuojamo daugianario pagrindu.

Aprašyto lauko $GF(p^n)$ elementus galime sužymėti numeriais (skaitmenimis), kurių tarpusavio veiksmai bus lygūs daugianarių veiksmų pagal taisykles gauto rezultato numeris. Akivaizdu, kad atlikus analogiškus veiksmus žiede Z_{p^n} negausime izomorfinio vaizdavimo. Tačiau kriptografijoje to ir yra siekiama – panaikinti galimybę keisti veiksmų eiliškumą.

1.3.2 VEKTORINĖS BŪLIO IR VIENKRYPTĖS FUNKCIJOS

Simetrinėje kriptografijoje sukeitimo blokai yra vienas iš pagrindinių ir vienintelių netiesiškumo šaltinių. Kiekvieną S bloką galima nagrinėti kaip n kintamųjų Būlio funkciją F su m komponenčių. $F: GF(2)^n \to GF(2)^m$. Bet kurią Būlio funkciją vienareikšmiškai aprašo jos teisingumo lentelė arba jos algebrinė normalinė forma (ANF) (Pommerening, 2005).

Šifravimo funkcijos paveiktų išvesties duomenų pokytis įvesties duomenų atžvilgiu priklauso nuo slaptųjų raktų, naudojamų užšifravimo metu. Ši svarbi savybė turi būti tenkinama ir tai apibūdina lavinis kriterijus:

Apibrėžimas. Sakoma, kad funkcija tenkina lavinį kriterijų (angl. avalanche criterion), jei pakeitus vieną įvesties bitą, pasikeičia vidutiniškai pusė išvesties bitų.

Apibrėžimas. Sakoma, kad funkcija tenkina griežtą lavinį kriterijų (angl. strict avalanche criterion) jei pakeitus vieną įvesties bitą, kiekvienas išvesties bitas pasikeičia su 0,5 tikimybe.

Kriptografiniu požiūriu saugus S blokas turi tenkinti šiuos kriterijus:

- Koreliacinį imunitetą (angl. correlation immunity – CI) (Daemen, Govaerts & Vandewalle, 1995): šis kriterijus rodo įėjimo ir išėjimo signalų koreliacijos laipsnį.
- Sklidimo kriterijų (angl. propagation criterion) (Carlet, 2009): tai apibendrintas griežtas lavinis kriterijus, kuriuo įvertinami Būlio funkcijos reikšmių pokyčiai, atsižvelgiant į dalies argumentų pokytį.
- Netiesiškumą, kuris rodo funkcijos atstumą iki afiniųjų funkcijų poerdvio (Carlet, 2011).

Kriptografijoje taip pat plačiai naudojamos ir vienkryptės funkcijos. Populiariausios: pirminių skaičių sandaugos funkcijos, eksponentė baigtinėse struktūrose ir kt. Su šiomis funkcijomis yra susieta ir pirminių skaičių sandaugos išskaidymo ir diskretinio logaritmo problemos.

Apibrėžimas. Tarkime turime funkciją f, kuri $\boldsymbol{f}: \boldsymbol{A} \to \boldsymbol{B}$. Sakoma, kad f yra vienkryptė funkcija jei bet kuriam $\boldsymbol{x} \in \boldsymbol{A}$ galima lengvai apskaičiuoti $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$ ir bet kuriam atsitiktinai parinktam $\boldsymbol{y} \in \boldsymbol{B}$, labai sunku apskaičiuoti tokį $\boldsymbol{x} \in \boldsymbol{A}$, kad $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{y}$. (Garey & Johnson, 1979)

Atskira vienkrypčių funkcijų klasė yra landinės vienkryptės funkcijos, kai funkciją galima efektyviai apgręžti, žinant tam tikrą slaptą informaciją.

Apibrėžimas. Tarkime turime funkciją f, kuri $\boldsymbol{f}: \boldsymbol{A} \to \boldsymbol{B}$. Sakoma, kad f yra landinė vienkryptė funkcija jei f yra vienkryptė funkcija ir egzistuoja tam tikra slapta informacija s, kurią žinant bet kuriam atsitiktinai pasirinktam $\boldsymbol{y} \in \boldsymbol{B}$, lengva apskaičiuoti tokį $\boldsymbol{x} \in \boldsymbol{A}$, kad $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{y}$. (Goldreich, 2003)

Simetrinio šifravimo atveju: atstatyti iš šifrogramos pradinę tekstogramą galima tik su slaptaisiais parametrais. Be slaptųjų raktų šis uždavinys praktiškai nėra išsprendžiamas.

1.4 MATRICINIO LAIPSNIO FUNKCIJA

Matricinio laipsnio (toliau – ML), funkcija yra paremta vienos matricos kėlimu laipsniu, kai laipsnio rodiklis yra matrica. Ši funkcija yra tam tikras diskrečiosios eksponentės baigtinėse grupėse apibendrinimas praplečiant ją virš matricų aibės, tačiau saugumo atžvilgiu funkcija skiriasi nuo diskrečiosios eksponentės, nes nėra paremta klasikine diskretaus logaritmo problema (*angl*. discrete logarithm problem – DLP).

ML funkciją galima interpretuoti kaip matricų grupės veiksmą kitoje matricų aibėje. Funkcija *f* yra apibrėžiama $m \times m$ eilės matricomis virš baigtinio lauko $GF(2^n)$ ir atlieka vaizdavimą iš $GF(2^n)^{m \times m}$ į $GF(2^n)^{m \times m}$. Šios funkcijos apibrėžimo sritis yra tam tikras $GF(2^n)^{m \times m}$ poaibis, kuris susideda iš matricų, neturinčių nulinių elementų. Tas poaibis žymimas $\boldsymbol{M}$, o funkcijos reikšmių sritis taip pat $\boldsymbol{M}$. Tuomet ML funkcija gali būti apibrėžta kaip vaizdavimas $f: \boldsymbol{M} \to \boldsymbol{M}$. Pagal nustatytą ML funkcijos apibrėžimo sritį, įvesties matricose negali būti nulinių elementų. Todėl esant poreikiui, kad ML funkcija dirbtu su bet kokiais duomenimis, suformuojamas transformavimo etapas.

Sudarome laipsnio matricų grupę M_G, sudarytą iš neišsigimusių $m \times m$ eilės matricų virš žiedo $Z^*_{2^n-1} = \{1, \dots, 2^n - 2\}$, visos jos turi sau atvirkštines matricas. Grupės operacija apibrėžiama kaip įprasta matricų daugyba naudojant $Z^*_{2^n-1}$ aritmetiką. Pažymime $M_G \subset Z^{m \times m}_{2^n-1}$.

Sudaryta ML funkcija atitinka matricų grupės M_G veiksmus aibėje *M*. Bendru atveju įprastinė matricų daugyba nėra komutatyvi, todėl ML funkcija išskaidoma į dvi atskiras funkcijas: matricinio laipsnio funkcija iš kairės ir matricinio laipsnio funkcija iš dešinės. ML funkcija iš kairės atlieka vaizdavimą $f_{L-}: \boldsymbol{M_G} \times \boldsymbol{M} \to \boldsymbol{M}$, o matricinio laipsnio funkcija iš dešinės - $f_{-R}: \boldsymbol{M} \times \boldsymbol{M_G} \to \boldsymbol{M}$. Pati ML funkcija tada gali būti apibrėžta kaip šių funkcijų kompozicija:

$$f_{L,R} = f_{L-} \circ f_{-R} = f_{-R} \circ f_{L-} \tag{1.1}$$

Formuojamas S blokas paremtas ML funkcija, su įvesties ir išvesties duomenų aibėmis: *D* – įvesties aibė, *C* – išvesties aibė. Aibę D sudaro $m \times m$ eilės matricos virš $GF(2^{n-1})$, t.y. $D = GF(2^{n-1})^{m \times m}$, atitinkamai - $C = GF(2^n)^{m \times m}$. Laikoma, kad S bloko funkcija *F* atlieka savarankišką šifravimo operaciją, t.y. atlieka vaizdavimą $F: D \to C$. Funkcija *F* konstruojama kaip injektyvus (*vienas elementas į vieną*) vaizdavimas $F: GF(2^{n-1})^{m \times m} \to GF(2^n)^{m \times m}$, siekiant užtikrinti vienareikšmišką atvirkštinį vaizdavimą F^{-1} duomenų iššifravimui.

Kaip minėta ankščiau, siekiant kad S blokas dirbtu su bet kokiais įvesties duomenimis, atliekama transformacija pašalinanti nulinius elementus iš matricų, veikianti papildomos funkcijos g_K pagalba. Taigi S bloko funkcija yra ML ir g_K funkcijų, apibrėžtų matricomis $K \in Z_{2^n-1}^{m\times m}$ ir $L, R \in M_G$, kompozicija. Tuomet matricinio laipsnio S bloko švesties matricos D šifravimo procedūra į išvesties matricą C formaliai aprašoma taip:

$$C = F(D) = f_{L,R}\big(g_K(D)\big) \tag{1.2}$$

Sukurto S bloko saugumas nėra paremtas klasikine DLP, kadangi naudojamų baigtinių laukų eilės yra sąlyginai nedidelės. Saugumas paremtas apibendrinta matricų dekompozicijos problema (*angl*. matrix decomposition problem – MDP). Kuomet yra žinoma įvesties ir išvesties matricų pora, mėginimas nustatyti slaptųjų raktų matricas prilygsta kelių kintamųjų daugianarių (*angl*. multivariate polynomial – MP) lygčių sistemos sprendimo sudėtingumui. MP uždaviniai net kelių kintamųjų kvadratinių lygčių sistemų sprendimas virš bet kokio lauko priklauso NP pilnajai klasei (Garey & Johnson, 1979; Patarin & Goubin, 1997).

1.4.1 ML FUNKCIJOS KONSTRAVIMAS

Pagal praeitame skyriuje apibrėžtus reikalavimus, bet kurios matricoms $L, R \in M_G$ ir $X \in M$ turi egzistuoti matricos $Y, Z \in M$, su kuriomis (įvedami pažymėjimai „$\triangleright$“ ir „$\triangleleft$“):

$$Y = f_{L-}(X) = L \triangleright X \tag{1.3}$$

$$Z = f_{-R}(X) = X \triangleleft R \tag{1.4}$$

Toliau taikomi ekvivalentumai: $L\ \{l_{ij}\};\ X\ \{x_{ij}\};\ R\ \{r_{ij}\};\ Y\ \{y_{ij}\};\ Z\ \{z_{ij}\}$ ir suformuojamos matricų grupės M_G operacijos $L \triangleright X = Y$ ir $X \triangleleft R = Z$. Atitinkamai Y ir Z matricų elementai:

$$y_{ij} = \prod_{s=1}^{m} x_{sj}^{l_{is}} \tag{1.5.}$$

$$z_{ij} = \prod_{t=1}^{m} x_{it}^{r_{tj}} \tag{1.6.}$$

l_{is} ir r_{tj} priklauso $Z_{2^n-1}^*$ t.y. jie yra sveikieji skaičiai, o x_{ij} yra $GF(2^n)$ elementai, todėl dauginimo ir kėlimo laipsniu veiksmai yra atliekami naudojant Galua lauko $GF(2^n)$ aritmetiką.

Pačių veiksmų struktūra ML funkcijos yra panaši į matricų daugybos, tik daugybos veiksmas yra pakeičiamas laipsnio kėlimu, o sudėties veiksmas – daugybos. Tiksliau: ML funkcija iš kairės apskaičiuodama vieną išvesties matricos elementą ima vieną eilutę iš raktų matricos L ir vieną stulpelį iš matricos X. ML funkcija iš dešinės eilutę ima iš matricos X, o stulpelį iš raktų matricos R.

Elementarus pavyzdys ML funkcijos veikimo:

$$Y = L \triangleright X = \begin{pmatrix} l_{11} & l_{12} \\ l_{21} & l_{22} \end{pmatrix} \triangleright \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} = \begin{pmatrix} {x_{11}}^{l_{11}} \cdot {x_{21}}^{l_{12}} & {x_{12}}^{l_{11}} \cdot {x_{22}}^{l_{12}} \\ {x_{11}}^{l_{21}} \cdot {x_{21}}^{l_{22}} & {x_{12}}^{l_{21}} \cdot {x_{22}}^{l_{22}} \end{pmatrix}$$

$$Z = X \triangleleft R = \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \triangleright \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} = \begin{pmatrix} {x_{11}}^{r_{11}} \cdot {x_{12}}^{r_{21}} & {x_{11}}^{r_{12}} \cdot {x_{12}}^{r_{22}} \\ {x_{21}}^{r_{11}} \cdot {x_{22}}^{r_{21}} & {x_{21}}^{r_{12}} \cdot {x_{22}}^{r_{22}} \end{pmatrix}$$

Raktų matricos elementai yra naudojami kaip baigtinio laiko elementų laipsniai. Algebrinė struktūra: bet kurį baigtinio lauko elementą pakėlus laipsniu lygiu lauko eilei, gausime tą patį elementą (*analogija į Fermą mažąją teoremą*). Arba keldami vienetu mažesniu laipsniu gausime vienetinį lauko elementą. Dėl šios priežasties, visi veiksmai su laipsniais atitinka žiedo, kurio eilė vienetu mažesnė negu baigtinio lauko, veiksmus.

Apibendrinant visos ML funkcijos išvesties matricą pažymėję C

$$f_{L-}(X) = L \rhd X = {}^{L}X \tag{1.7}$$

$$f_{-R}(X) = X \lhd R = X^{R} \tag{1.8}$$

$$f_{LR}(X) = {}^{L}X^{R} = C \tag{1.9}$$

Galima pastebėti, kodėl reikalingas apribojimai įvesties matricoms, t.y. nulių panaikinimas, išreikšdami bet kurį C elementą taip:

$$\prod_{t=1}^{m}\prod_{s=1}^{m} x_{st}^{l_{is}r_{tj}} = c_{ij} \quad i,j = 1,2,\dots,m \tag{1.10}$$

Kriptografiniu požiūriu labai svarbi ML funkcijos savybė: kiekvienas išvesties matricos elementas priklauso nuo visų įvesties matricos elementų. Pateikiamas bendras ML funkcijos taikymo pavyzdys:

$$C = \left({}^{L}X\right)^{R} = \begin{pmatrix} {x_{11}}^{l_{11}r_{11}} \cdot {x_{21}}^{l_{12}r_{11}} {x_{12}}^{l_{11}r_{21}} \cdot {x_{22}}^{l_{12}r_{21}} & {x_{11}}^{l_{11}r_{12}} \cdot {x_{21}}^{l_{12}r_{12}} {x_{12}}^{l_{11}r_{22}} \cdot {x_{22}}^{l_{12}r_{22}} \\ {x_{11}}^{l_{21}r_{11}} \cdot {x_{21}}^{l_{22}r_{11}} {x_{12}}^{l_{21}r_{21}} \cdot {x_{22}}^{l_{22}r_{21}} & {x_{11}}^{l_{21}r_{12}} \cdot {x_{21}}^{l_{22}r_{12}} {x_{12}}^{l_{21}r_{22}} \cdot {x_{22}}^{l_{22}r_{22}} \end{pmatrix}.$$

1.5 MATRICINIO LAIPSNIO S BLOKO KONSTRAVIMAS

S bloko formavimo idėja, kaip ir buvo aprašyta ankstesniuose skyriuose, yra įvesties matricų transformacija. Pati ML funkcija negali būti ir S bloku. Toliau padaliname ML S bloko konstravimą į bloką vienai šifro iteracijai ir pakartotinį bloko naudojimą, esant n iteracijų.

1.5.1 VIENOS ITERACIJOS ŠIFRO S BLOKAS

Apibrėžta ML funkcija naudoja $GF(2^n)$ aritmetiką, todėl negalime sudaryti izomorfizmo į kitą aibę, kuri būtų didesnė ir tai leistu nenaudoti nulinio elemento. Bet kurio baigtinio lauko eilė yra kurio nors pirminio skaičiaus laipsnis, todėl pasirinkus $GF(2^n)$ lauką rasti kitą lauką su 2^{n+1} elementais praktiškai yra neįmanoma. Todėl S bloke įvedama funkcija g_K, kuri atlieka injektyvų vaizdavimą, priklausantį nuo slaptojo rakto $K \in Z_{2^n-1}^{m\times m}$, t.y. visi įvesties matricos elementai priklausantys $GF(2^{n-1})$ atvaizduojami į nenulinius $GF(2^n)$ elementus. Formuluotė:

$$g_K: GF(2^{n-1})^{m\times m} \to M \subset GF(2^n)^{m\times m}.$$

Funkcija yra primityvi, su dviem saugumo parametrais, tam, kad įvedimas nereikalautų papildomų veiksmų skaičiaus. Tačiau ji vis tiek padidina duomenų srautą, t.y. išvesties matrica bus m^2 bitų didesnė. Struktūra paremta sumos operacija žiede Z_{2^n} ir bet kuriai įvesties matricai $D \in GF(2^{n-1})^{m\times m}$:

$$g_K(D) = D + K + \mathbf{1} = X$$

*(**1** - $m \times m$ dydžio vienetinė matrica)*

Bendru atveju K matrica yra laisvai pasirenkama, todėl vienetinė matrica užtikrina nulinių elementų pašalinimą. Toliau pateikiamas įrodymas, kad po transformacijos, t.y. funkcijos rezultatas priklausys aibei **M**.

Lema. Dviejų nepriklausomų $Z_{2^{n-1}}$ elementų suma bus ne mažesnė nei 0 ir neviršys $2^n - 2$.

Įrodymas. Nagrinėjami du ribiniai atvejai: maksimalus ir minimalus sumos rezultatas. Tegu du, nepriklausomai pasirinkti atsitiktiniai elementai $x, y \in Z_{2^{n-1}}$. Tada:

$$\min(x + y) = \min(x) + \min(y) = 0 + 0 = 0$$

$$\max(x + y) = \max(x) + \max(y) = 2^{n-1} - 1 + 2^{n-1} - 1 = 2^n - 2$$

Tiek $\min(x + y)$ tiek $\max(x + y)$ priklauso Z_{2^n}, o visi kiti galimi sumų rezultatai bus tarp šių dviejų reikšmių ir taip pat priklausys Z_{2^n}.

Funkcijoje g_K sumuojamos trys reikšmės: du aibės $Z_{2^{n-1}}$ nariai ir vienetas. Todėl rezultate mažiausias galimas matricos elementas bus vienetas, o didžiausias $2^n - 1$, t.y. visi elementai tikrai priklausys Z_{2^n} ir tarp jų nebus nulių. Taigi $X \in \mathbf{M}$.

Taigi, ML S blokas yra priklausomas nuo trijų slaptųjų raktų – matricų L,R ir K. Kaip ir buvo minėta L ir R privalo turėti atvirkštines matricas, K jokių apribojimų neturi. S bloko funkcija laikome injektyviu vaizdavimu iš $GF(2^{n-1})^{m \times m} \rightarrow \boldsymbol{M}$ ir patį ML S bloką išreiškiame taip:

$$F(D) = f(g_K(D)) = f(X) = {}^{L}X^{R} = C \tag{1.11}$$

$$F(D) = {}^{L}(D + K + \mathbf{1})^{R} = C$$

Selektyviai kiekvieną išvesties matricos elementą galima pasiekti taip:

$$\prod_{t=1}^{m} \prod_{s=1}^{m} (d_{st} + k_{st} + 1)^{l_{is} r_{tj}} = \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{is} r_{tj}} = c_{ij} \tag{1.12}$$

$$i, j = 1, 2, \dots, m$$

Iššifravimo etape naudojama ML S bloko atvirkštinė funkcija F^{-1}. Tiesioginė F funkcija yra dviejų kitų funkcijų kompozicija, todėl privalu sudedamąsias funkcija taip pat apversti ir taikyti atvirkštine eiliškumo tvarka:

$$F^{-1}(C) = {g_K}^{-1}(f^{-1}(C)).$$

Pačios ML funkcijos atvirkštinei funkcijai prireiks ir L ir R raktų atvirkštinių matricų.

$$f^{-1}(C) = {}^{L^{-1}}C^{R^{-1}}.$$

Transformacijos funkcijai g_K atvirkštinė funkcija gaunama apgręžiant sumos operaciją:

$${g_K}^{-1}(X) = (X - K - \mathbf{1}) = D$$

ML atvirkštinė funkcija atlieka vaizdavimą iš M į M, o ${g_K}^{-1}$ vaizdavimą $\boldsymbol{M} \rightarrow GF(2^{n-1})^{m \times m}$.

Teorema. ML S bloko $F(D) = {}^{L}(D + K + \mathbf{1})^{R} = C$ atvirkštinė funkcija yra

$$F^{-1}(C) = {}^{L^{-1}}(C)^{R^{-1}} - K - \mathbf{1} = D$$

su bet kuriomis matricomis $L, R \in \boldsymbol{M}_G$, $K, D \in GF(2^{n-1})^{m\times m}$ ir $F(D) = C \in \mathbf{M}$.

Įrodymas. Kadangi $\boldsymbol{M}_G$ yra matricų grupė, todėl bet kurioms matricoms $L, R \in \boldsymbol{M}_G$ egzistuos tokios matricos $L^{-1}, R^{-1} \in \boldsymbol{M}_G$, su kuriomis galios tapatybės:

$$RR^{-1} = R^{-1}R = \boldsymbol{I} = LL^{-1} = L^{-1}L$$

$$I \in Z_{2^n-1}^{m\times m} \text{ (\textit{vienetinė matrica})}$$

Pagal ML S bloko funkcijos apibrėžimą jos atvirkštinę galima išreikšti taip:

$$F^{-1}(C) = {}^{L^{-1}}(C)^{R^{-1}} - K - \mathbf{1} = {}^{L^{-1}}({}^{L}(D + K + \mathbf{1})^{R})^{R^{-1}} - K - \mathbf{1}$$

Toliau remdamiesi ML funkcijų asociatyvumu ir ML savybėmis: $(X^R)^{R^{-1}} = X$ ir $H({}^{L}X^{R}) \neq {}^{L}(H(X))^{R}$ gauname:

$${}^{L^{-1}L}(D + K + \mathbf{1})^{RR^{-1}} - K - \mathbf{1} = \mathrm{D} + \mathrm{K} + \mathbf{1} - \mathrm{K} - \mathbf{1} = \mathrm{D}.$$

Analogiškai selektyvus matricos elemento:

$$\left(\prod_{t=1}^{m}\prod_{s=1}^{m} {c_{st}}^{l'_{is}r'_{tj}}\right) - k_{ij} - 1 = x_{ij} - k_{ij} - 1 = d_{ij}$$

$$l'_{is}r'_{tj} \text{ - \textit{atvirkštiniai elementai.}}$$

Išvada: su korektiškai parinktais raktais ir pertvarkytais slaptaisiais raktais, visuomet gausime matrica D, kurios visi elementai bus mažesni vienu bitu nei C elementai.

1.5.2 MATRICINIO LAIPSNIO S BLOKAS ITERACINIAME ŠIFRE

Patį ML S bloką galime naudoti kaip pagrindinę funkcija iteraciniame šifre. Vidinę funkciją g_K nagrinėjant kaip sudėtines matricines funkcijas, t.y. kiekvieną matricos elementą veikiančias funkcijas. Nustatome, jog kiekviena iš jų atlieka po dvi sudėties operacijas žiede Z_{2^n}, todėl jos nėra ekvivalenčios nei vienai laipsninei funkcijai ir nėra distributyvios daugybos atžvilgiu lauke GF(2^n). Pakartotinis ML S bloko funkcijos panaudojimas nebus ekvivalentus vienai ML funkcijai su atitinkamais raktais. Tačiau naudojant ML S bloką pakartotinai išvesties matrica su kiekviena iteracija užims m^2 bitų daugiau nei tarpinės įvesties. Taigi po t iteracijų, matrica bus didesne tm^2 bitų.

Vidinės transformacinės funkcijos g_K tikslas buvo pašalinti iš įvesties matricos nulinius elementus. Kadangi pritaikius ML funkciją nulinių reikšmių tarp matricos elementų neatsiranda, todėl pakartotinai atlikti transformacija yra nelogiška. Todėl atsisakant funkcijos g_K, įvedamas papildomas saugumo parametras – H funkcija.

H funkcija turėtų būti neasociatyvi su ML funkcija ir ji neturėtų generuoti nulinių reikšmių. Todėl pasirenkamos sudėtinės funkcijos h, kurios yra bijektyvios vektorinės Būlio funkcijos neekvivalenčios laipsninėms funkcijoms. Kriptografiniu požiūriu pageidautina, kad šios funkcijos turėtų kuo aukštesnį netiesiškumą, kuo žemesnį diferencialinį vienodumą ir aukštą algebrinį laipsni. t iteracijų šifravimo procedūra aprašoma taip:

$${}^{L_t}H_t(\ldots({}^{L_3}H_3({}^{L_2}H_2({}^{L_1}(\,g_K D)^{R_1})^{R_2})^{R_3})\ldots)^{R_t} = C$$

1.5.3 MATRICINIO LAIPSNIO ŠIFRAS

ML šifras yra simetrinis blokinis iteracinis šifras. Įvesties duomenys turi būti $m \times m$ eilės matricos virš $GF(2^{n-1})$, išvesties - $m \times m$ eilės matricos virš $GF(2^n)$. Slaptieji saugumo parametrai – $m \times m$ eilės matricos: K virš $Z_{2^{n-1}}$, L ir R matricos iš M_G. ML S blokas, atliekantis nulinių elementų pašalinimo iš įvesties matricų funkciją, taikomas tik pirmoje iteracijoje. Vėliau, esant sąlygai, jog nulinių reikšmių neatsiras, naudojama bijektyvi matricinė funkcija H. Iteracinė schema:

1.6 KRIPTOANALIZĖ

Kriptoanalizė – tai kriptologijos šaka nagrinėjanti šifro išsprendimo galimybes, t.y. nagrinėjami įvairūs užšifruotos informacijos korektiško nuskaitymo būdai, nežinant slaptųjų parametrų arba bent jų dalies. Tai ne tik matematinė sritis, siekianti rasti silpnąsias šifro algoritmo vietas, ar naujo šifro kūrimo atveju siekianti išmėginti protokolą, bet apimanti ir kitus šifro nulaužimo variantus. Pvz.: pilno perrinkimo, statistinės informacijos rinkimo, trečiosios šalies įterpimą į algoritmą ir t.t.

Dažniausiai šifro algoritmai yra kuriami viešai, dėl ribotų resursų, todėl taikomas Kerkhofo dėsnis, kad šifravimo ir iššifravimo sistemų saugumas priklauso tik nuo slaptųjų raktų. Taigi dažnas atvejis, kai kenkėjas žino kokia šifravimo sistema yra naudojama, t.y. algoritmo matematinį variklį.

Kriptografinės atakos tipą galime nustatyti pagal tai, kokios yra kenkėjo galimybės ir tikslai. Detaliau: pagal tai kokios yra kenkėjo galimybės, t.y. turimus skaičiavimo pajėgumus ir turimos informacijos pobūdį ir kiekį, ir pagal kenkėjo tikslus, t.y. pagal tai ko kenkėjas siekia. Šiuo atveju kenkėjo tikslai gali būti įvairūs, tačiau pagrindiniai tai paslapties – užšifruotos tekstogramos atskleidimas ar slaptųjų raktų – parametrų atskleidimas. Pagal galimybes išskiriamos šių atakų variantai:

- Šifrogramos ataka (*angl*. ciphertext-only attack): primityviausia ataka, kuomet kenkėjas surenka tik paslaptis ar jų dalis – šifrograma. Nieko nėra žinoma nei apie pradinį tekstą, nei apie slaptuosius parametrus.
- Žinomos tekstogramos ataka (*angl*. known-plaintext attack): kenkėjas turi tiek pradinį paslaptį – tekstą ir šifrogramą. Turėdamas pirminį ir galutinį tekstą mėgina surasti slaptuosius parametrus, tinkančius surinktiems duomenims.
- Pasirinktos tekstogramos ataka (*angl*. chosen-plaintext attack): kenkėjas turi prieiga prie šifravimo sistemos, t.y. gali užšifruoti savo pasirinktą pradinį tekstą ir gauti šifrograma. Slaptieji parametrai nėra žinomi, tačiau kenkėjas gali išmėginti įvarius variantus, siekiant juos atskleisti ar sukompromituoti.

- Pasirinktos šifrogramos ataka (*angl*. chosen-ciphertext attack): kenkėjas turi prieigą prie šifravimo sistemos, o tiksliau prie iššifravimo, jei tai atskiros sistemos. Ir analogiškai kaip Pasirinktos tekstogramos atakoje, kenkėjas iššifruoja visus pasirinktas paslaptis – šifrogramos, gaudamas korektiškas pradines paslaptis – tekstus, tačiau vis tiek nežino slaptųjų parametrų.

Visos šios atakos gali būti vykdomos lygiagrečiai su perrinkimo ataka. Tegu visų galimų slaptųjų raktų aibė bus K. jei kenkėjas turi n vienetų skaičiavimo priemonių, t.y. procesorių pajėgumai, kiekiai, sistemos. Tegu vieno iš aibės K elementų korektiškumo algoritmo atžvilgiu patikrinimas užtrunka *t* laiko, tuomet viena atakos realizacija vidutiniškai truks $|K|\mathrm{t}/2\mathrm{n}$. Tačiau čia ir atsiranda atakos trūkumas – kenkėjas turi turėti galimybę patikrinti slaptojo rakto teisingumą. Dažniausiai šita ataka vadinama Pilno perrinkimo ataka (*angl*. Brute-force attack).

1.6.1 DIFERENCIALINĖ KRIPTOANALIZĖ

Pačią diferencialinę kriptoanalizę pasiūlė Eli Biham & Adi Shamir (1991). Pats svarbiausias atsparumo šiai atakai matas yra tekstogramų ir šifrogramų porų skaičius, reikalingas kriptoanalizei. Šių porų skaičius yra atvirkščiai proporcingas maksimaliai diferencialinei tikimybei (angl. differential probability – DP) (Nyberg&Knudsen, 1995; Lai, Massey&Murphy, 1991).

Diferencialinė analizė yra vienas iš efektyviausių kriptoanalizės įrankių prieš standartines blokines simetrines kriptosistemas. Plačiąja prasme, tai analizinis metodas, kuris tiria kaip įvesties duomenų pokytis paveikia šifrogramą, o blokinių kriptosistemų atveju siekiama atrasti neatsitiktinius pokyčius transformacijų etape, kurie parodo kaip keičiasi pradiniai duomenys. Pažymime, jog tiesinės arba afinios operacijos nesudaro skirtumų arba jų pakeitimus galima nuspėti.

Bitų perstatymo operacija, perrikiuojanti x bitus į $P(x)$, taip pat perrikiuoja ir jų skirtumus, kuriuos nagrinėjant galima pastebėti, kad į duomenis įterpto vieno iš etapo tarpinis raktas – slaptasis parametras gali būti ir neanalizuojamas.

Pavyzdžiui, tegu kurio nors etapo raktas sudedamas su iki jo atėjusia pradinio teksto verte moduliu 2, ($y = x \oplus k$), tai šifruojant sekančio etapo porą (x^*, y^*) gaunama ($y^* = x^* \oplus k$).

Įrodoma, jog įterpimo operacijos skirtumas tarp etapų nepriklauso nuo tarpinių raktų, todėl skaičiuojant šifravimo skirtumus, galima nenagrinėti įterpimo operacijos.

$$\Delta y = y \oplus y^* = (x \oplus k) \oplus (x^* \oplus k) = \Delta x$$

Skaičiuojant netiesinių šifravimo operacijų, pvz.: S blokų ar kitos netiesinės operacijos skirtumus, galima pastebėti: jei įvesčių skirtumas lygūs 0, tai ir išvesties skirtumai bus nuliniai. Kai įvesties skirtumas nenulinis, reikalinga statistinė analizė siekiant išsiaiškinti skirtumų priklausomybę nuo įvesties skirtumų. Tačiau tiksliai pasakyti, kokie bus išvesties skirtumai yra sudėtinga.

Netiesinei operacijai, tegu S blokui, pritaikius diferencialinę kriptoanalizę sudaroma kiekvieno iš bloko skirtumų skirstinio lentelė (angl. difference disribution table). Lentelėje eilutės žymi įvesties skirtumus, o stulpeliai – visus galimus išvesties skirtumus. Skirtumų lentelė užpildoma įvesties ir išvesties porų atitinkančių konkrečius skirtumus skaičiumi. Tegu įvesties ilgis *4* bitai, tai bus 2^4 skirtingų tekstogramų. Šiuo atveju bus ir *16* skirtingų skirtumų, todėl pačių skirtumų skirstinio lentelę sudarys *16* eilučių, o stulpelių skaičius priklausys nuo išvesties bitų skaičiaus. Siekiant korektiškai užpildyti lentelę perrenkami visi variantai: visos įvesties duomenų poros, turinčios konkretų skirtumą. S bloko atveju – apskaičiuojamos visos galimos bloko išvestys su visomis tekstogramomis ir visi galimi įvesties bei išvesties skirtumai. Jei išvestis *4* bitų, tai iš viso reikės apskaičiuoti *128* įvesčių ir *128* išvesčių skirtumus, nes kiekviena skirtumą atitinka *8* poros.

Diferencialinė kriptoanalizė nusako susidarančių skirtumų galimų pokyčių charakteristikas įvairiose šifro iteracijose. Kiekvienai charakteristikai priskiriamas tekstogramos skirtumas, kuriam ji jį prognozuoja kitiems etapams (iteracijoms). Tekstogramų poros, kurių pradinis skirtumas ir šifravimo funkcijos išvesties skirtumas yra teisingai nuspėtas atitinkamos charakteristikos, yra vadinamos geromis arba teisingomis (*angl*. right pairs), likusios blogomis arba klaidingomis (*angl*. wrong pairs).

Tikimybė, kad charakteristika korektiškai nurodys skirtumą, t.y. kad paimta atsitiktinė pora, turinti reikiamą skirtumą, bus gera arba teisinga, priklauso nuo kiekvieno S bloko (ar dalinės operacijos) ryšių tarp įvesties ir išvesties skirtumų. Viso šifro bendra tikimybė, kad su pasirinktu tekstogramos skirtumu bus gautas reikiamas šifrogramos skirtumas, lygus visų raundų tikimybių sandaugai, jei visos tikimybės yra nepriklausomos. Kitu atveju gaunama tik tikimybės aproksimacija (*apytikslė reikšmė*). Taigi diferencialinė tikimybė yra didžiausia iš tikimybių, kad atsitiktinai parinkta tekstograma, turinti fiksuotą skirtumą, bus atvaizduota į šifrogramą, turinčią atitinkamą fiksuotą skirtumą. Turint priešpaskutinio blokų sukeitimo etapo tarpinių duomenų korektišką skirtumą, įmanoma nustatyti nežinomą raktą – slaptuosius parametrus, pasitelkus statistinę analizę.

Diferencialinė kriptoanalizė yra konkrečios tekstogramos ataka, vykdoma šiais etapais:

1) Duomenų rinkimo metu reikalaujama užšifruoti didelį kiekį tekstogramų porų, kurių skirtumai parenkami pagal nustatytą skirtumų kitimo charakteristiką.
2) Duomenų analizės metu išgaunamas slaptasis parametras – raktas iš gautų šifrogramų tyrimo.

Tegu diferencialinė tikimybė yra *p*, t.y. tikėtina, jog *p* dalis porų bus geros-tinkamos. Tokiu būdų priešpaskutiniame raunde *p* dalies porų skirtumai atitiks charakteristiką. Paprasčiausias (neefektyvus) būdas nustatyti paskutinio etapo raktą – išbandyti visus galimus variantus. Kiekvieno galimo rakto tinkamumui patikrinti, visoms šifrogramoms atliekamas vienas iššifravimo raundas ir apskaičiuojami kiekvienos poros duomenų skirtumai iki paskutinio raundo įvesties. Teoriškai, kai bus taikomi neteisingi raktai – slaptieji parametrai, charakteristikose nurodyti duomenų skirtumai turėtų pasirodyti retai, o

nurodžius teisingą raktų porą – atitinkami skirtumai sutaps su charakteristikų nurodytu skirtumu ne mažiau kaip *p* kartų.

Norint diferencialinei kriptoanalizei gauti pakankamai daug tinkamų tekstogramos ir šifrogramos porų, paprastai reikia $k/(1/p)$ (*p – atakai naudojamos charakteristikos tikimybė*) pasirinktų tekstogramų porų. Šis užšifruotų duomenų (porų) kiekis gali būti labai didelis, todėl dažniausiai duomenų surinkimo etapas yra sudėtingesnis už duomenų analizės etapą. Pavyzdžiui DES (*angl*. Data Encryption Standard) šifro kriptoanalizės atveju reikia 2^{47} pasirinktų tekstogramų blokų (Heys, 2001). Dėl itin didelio pasirinktų tekstogramų skaičiaus ataka gali būti neprasminga, nes didesnė atakos skaičiavimų dalis perduodama užpultajai šaliai, kuri turi užšifruoti daug tekstogramų, kad kenkėjas galėtų realizuoti savo ataką.

Šifro atsparumą diferencialinei kriptoanalizei galima įvertinti ir nagrinėjant atskirų jo S blokų charakteristikas. S blokai yra vektorinės Būlio funkcijos, todėl diferencialinių tikimybių skaičiavimą apibrėšime per jas.

Funkcijos $f: GF(2^u) \to GF(2^v)$ diferencialu vadinama tokia pora $(\alpha, \beta) \in GF(2^u) \times GF(2^v)$, kuri su tam tikru $x \in GF(2^u)$ tenkina lygybę $f(x) \oplus f(x \oplus \alpha) = \beta$. Dydis α vadinamas įvesties skirtumu, o β – išvesties skirtumu.

Diferencialo (α, β) diferencialinė tikimybė DP(α, β) funkcijos *f* atžvilgiu yra apibrėžiama tokiu būdu:

$$DP_f(\alpha, \beta) = 2^{-u} \cdot \#\{x \mid f(x) \oplus f(x \oplus \alpha) = \beta\} \tag{1.13.}$$

čia # žymi aibės kardinalumą (aibės galią).

Funkcijos *f* maksimali diferencialinė tikimybė:

$$DP_f = max_{\alpha \neq 0, \beta} DP_f(\alpha, \beta) \tag{1.14.}$$

Jei funkcija *f* yra priklausoma nuo rakto k iš raktų aibės K, tai analogiškai (1.14.) galima apibrėžti diferencialinę tikimybę $DP_f[k]$. Iš $DP_f[k]$ galima gauti funkcijos f vidutinę arba tikėtiną didžiausią diferencialinę tikimybę (EDP) (Daemen & Rijmen, 2007):

$$EDP_f = 2^{-\#k} \sum_{k \in K} DP_f[k] \tag{1.15.}$$

EDP_f yra vidurkis gautas su visais galimais raktais. Čia ir toliau tekste, jei nepaminėta kitaip, laikoma kad raktai yra tolygiai pasiskirstę.

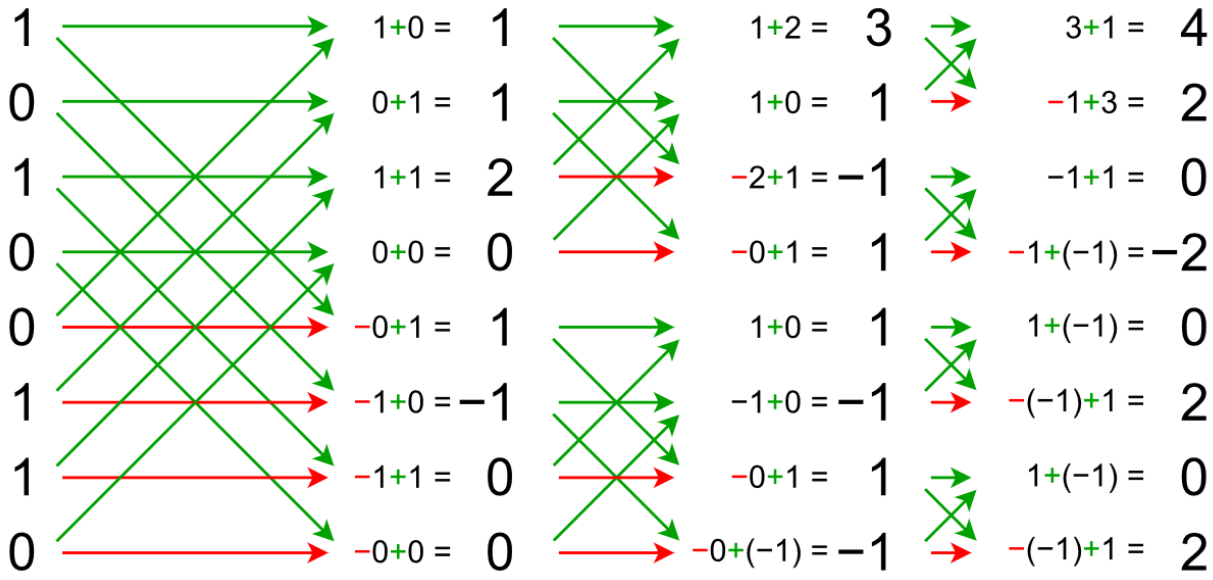
Dažnai nagrinėjant diferencialines tikimybes naudojamas ir funkcijų diferencialinio vienodumo (*angl*. uniformity) apibrėžimas (Nyberg, 1993).

2.7. ***Apibrėžimas****. Funkcija $f: GF(2^u) \to GF(2^v)$ yra vadinama diferencialiai k-vienoda (angl. k-uniform), jei k yra aibės $\{x \mid f(x) \oplus f(x \oplus \alpha) = \beta,\ x \in GF(2^u)\}$ maksimalus kardinalumas su visais $0 \neq x \in GF(2^u)$ ir $\beta \in GF(2^v)$, t.y. lygtį $f(x) \oplus f(x \oplus \alpha) = \beta$ tenkina nedaugiau kaip k sprendinių.*

Iš čia galima nustatyti ryšį tarp funkcijos vienodumo ir jos diferencialinės tikimybės, t.y. κ - vienodos funkcijos *f* diferencialinė tikimybė yra lygi

$$DP_f = \frac{\kappa}{2^u} \tag{1.16.}$$

Funkcijos diferencialinį vienodumą efektyviai galima nustatyti naudojantis teisingumo lentele ir Walsh transformacija (*angl*. Walsh transformation) (Pommerening, 2005; Nefas, 2007).



1.3. pav. Walsh transformacijos pavyzdys

Kuo mažesnis yra funkcijos diferencialinis vienodumas, tuo ji suteikia geresnį atsparumą diferencialinei kriptoanalizei. Ilgą laiką buvo manoma, kad geriausių bijektyvių vektorinių Būlio funkcijų vienodumas yra *4*, jei jų koordinačių skaičius yra lyginis. Tačiau itin retais atvejais galima gauti ir funkcijas, kurių vienodumas yra *2* (Dillon, 2009).

Kaip ir buvo minėta, kuriant naują šifrą visuomet yra taikoma kriptoanalizė su tikslu išsiaiškinti, ar šifras yra korektiškas. Kūrėjai turi įrodyti, jog diferencialinė ataka negali būti taikoma. Siekiant apsisaugoti nuo diferencialinės atakos, privalu kad maksimalioji diferencialinė tikimybė būtų kuo įmanoma mažesnė. Taigi geriausios charakteristikos (ar skirtumo) tikimybė p yra apribojama taip, kad skaičius $1/p$ būtų didesnis už tam tikrą saugumo slenksti (pagal šifro saugumo reikalavimą) arba didesnis net už tekstogramų aibės galią. Tokiu atveju net perrinkus visą tekstogramų aibę neįmanoma įvykdyti atakos. Šios ribos nustatymas leidžia įrodyti šifro saugumą diferencialinės kriptoanalizės atžvilgiu (Kang, et al., 2001; Hong, et al., 2000).

Galimi įvairūs diferencialinės atakos patobulinimo būdai:

- kelių skirtingų charakteristikų sujungimas į vieną (Biham & Shamir, 1991)
- sutrumpinti skirtumai (Knudsen, 1995)
- aukštesnių eilių skirtumai (Knudsen, 1995)

Taip pat yra tokių atakų, kurios skirtumus pasitelkia kaip sudedamą šias dalis ir juos sujungia įvairiais būdais:

- bumerango (Wagner, 1999)
- sustiprinta bumerango (Kelsey, Kohno&Schneier, 2000)
- stačiakampio atakos (Biham, Dunkelman & Keller, 2002).

Taikant diferencialinę kriptoanalizę, privalu suprasti, kad įvesties skirtumų sklaida šifro vidinėse šifro iteracijose – etapuose tampa sunkiai prognozuojama, jei nėra žinomi visi netiesiniai šifro elementai ir prieš tai neapskaičiuojamos jų skirtumų skirstinio lentelės. Todėl buvo sukurtas AES patobulinimas, kuris vietoje atvirkštinės funkcijos S bloko naudojo pseudoatsitiktinius S blokus, priklausančius nuo slaptojo rakto (Kazlauskas & Kazlauskas, 2009). Autoriai teigia, kad nežinant šifro struktūros, diferencialinė kriptoanalizė yra negalima ir AES šifras su priklausančiu nuo rakto S bloko yra atsparus šiai atakai. Priklausančius nuo raktų S blokus turi ir anksčiau sukurtas šifras Twofish (Schneier, et al., 1999). Dar viena DES šifrui pritaikyta ataka yra tiesinė kriptoanalizė (Matsui, 1994). Šios atakos pagrindinis tikslas yra sudaryti šifro tiesines aproksimacijas ir jas naudojant mėginti atstatyti slaptuosius raktus pagal turimas tekstogramų ir šifrogramų poras.

Tiek tiesinė tiek diferencialinė atakos naudoja skirtingas šifro (S blokų) charakteristikas, tačiau iš tikrųjų egzistuoja ryšys tarp šių kriptoanalizių (Chabaud & Vaudenay, 1995). Panašiai šios dvi atakos susietos ir vadinamojoje dekoreliacijos teorijoje (Vaudenay, 2003).

1.6.2 ALGEBRINĖ KRIPTOANALIZĖ

Algebrinė analizė paremta tuo, kad bet kurį blokinį šifrą ar atskiras jo sudėtines dalis galima aprašyti algebrinių lygčių sistema, susiejančia tekstogramos, šifro raktų ir šifrogramos duomenų bitus (Meier, Pasalic & Carlet, 2004). Pirmosios algebrinės kriptoanalizės idėjos pasirodė 1983 m. (Schaumüller-Bichl, 1983), nors pačios idėjos autorius yra moderniosios kriptologijos pradininkas Shannon, kuris teigė, kad korektiško šifro nulaužimui reikia tiek pat pastangų, kiek ir išspręsti atitinkamo dydžio sudėtingų lygčių sistemą (Shannon, 1949). Susidomėjimas algebrinę kriptoanalize sugrįžo ~2002 m., kai Kurtua ir Piepžikas paskelbė straipsnį, kuriame sudarė AES šifro algebrinių lygčių sistemą.

Pagal šifro algoritmą sudarius lygčių sistemą ir turint vieną ar daugiau įvesties bei šifruotų duomenų porų, galima ieškoti gautos lygčių sistemos sprendinio ir nustatyti šifravimo raktą. Priešingai nei tradicinei diferencialinės kriptoanalizės atakai, algebrinei atakai nereikia daug tekstogramų ir šifrogramų porų. Jai užtenka vienos ar kelių porų, o tai sąlyginai ženkliai didina algebrinės kriptoanalizės efektyvumą.

Naujų šifrų kūrimas kardinaliai keičiasi, nes seni blokinių šifrų saugumo postulatai keičiasi (Courtois, Dabraize & Garrido, 2005; Courtois, 2005), todėl buvo apibrėžtos tezės:

- Šifro sudėtingumas nedidėja eksponentiškai šifravimo raundų skaičiaus atžvilgiu.
- Algebrinei kriptoanalizei pakanka vienos ar keleto tekstogramų ir šifrogramų porų.

Nors ir šifrai realizuoti laipsninėmis šifravimo funkcijomis, tokiomis kaip Gold, Kasami, atvirkštinė ir kt., pasižymi pakankamu atsparumu paprastoms atakoms, tačiau juos gali pažeisti išskirtinai taikoma algebrinė kriptoanalizė. (Cheon & Lee, 2004). Šios funkcijos, aprašomos gana paprastomis algebrinėmis lygtimis, tampa potencialiai jautrios atakai.

AES šifras yra beveik optimalus tiesinei ir diferencialinei kriptoanalizei (Baigneres & Vaudenay, 2006), tačiau jis gali būti aprašytas paprastomis lygtimis. Kol kas šifras nėra sukompromituotas, tačiau yra vykdomas projektas NESSIE ištirti šifro atsparumą algebrinei kriptoanalizei. AES šifre vienintelis netiesinis elementas yra atvirkštinio elemento funkcija (Daemen & Rijmen, 2002), todėl lygčių sistema yra sąlyginai nesudėtinga ir gali būti išreikšta kvadratinių daugianarių lygčių sistema. (Courtois & Pieprzyk, 2002). Tačiau kol kas nėra nei vieno viešai žinomo sprendimo metodo, kuris būtų įgalus išspręsti lygčių sistemas, savo sudėtingumu neviršydamas procese apibrėžtų skaičiavimo galimybių.

Algebrinių lygčių sistema sudaryta pagal AES šifrą yra neišsprendžiama, nes nežinomųjų skaičius yra didesnis nei pačių lygčių, tačiau buvo pasiūlytas metodas, kaip padidinti lygčių skaičių iki reikiamo (Courtois & Pieprzyk, 2002). Gaunamos netiesiškai priklausomos lygtys, todėl jų sprendiniui rasti pritaikoma ištiesinimo metodai bei efektyvus Gauso algoritmas. Papildomų lygčių sudarymas pagrįstas vadinamaisiais XL (*angl*. eXtended Linearization) arba patobulintais metodais kaip XSL (*angl*. eXtended Sparse Linearization) ir kt. (Cid & Leurent, 2005; Courtois & Pieprzyk, 2002; Courtois, Klimov, Patarin & Shamir, 2000). XSL atakos metu sistemoje esantys polinomai dauginami iš specialiai parinktų monomų (vienanarių), kurių eilė neviršija apibrėžto slenksčio. Termai parenkami taip, kad atliekant šią daugybą atsirastu kuo daugiau papildomų termų, jau esančių kitose lygtyse.

Tokiu būdu atliekama tam tikra optimizacija, siekiant išvengti daugelio papildomų kintamųjų atsiradimo. Atliekant ištiesinimo procedūrą, gaunama tiesinių lygčių sistema, tačiau jos turi būti tinkamos ištiesinimui. Tyrinėjant algebrinių atakų veiksmingumą pasitelkus ankščiau minėtus XL ir XSL atakas ir jų modifikacijas, pastebėta, kad atakos veiksmingumas priklauso ne tik nuo kintamųjų ir monomų skaičiaus, bet ir nuo balanso tarp lygčių skaičiaus ir termų skaičiaus, esančių tose lygtyse. Nustatyti du algebrinio imuniteto (AI) kriterijai, susieti su dviem XSL atakų versijomis (Courtois, Debraize & Garrido, 2005):

$$\Gamma(r,n,t) = \left(\frac{t}{n}\right)^{\frac{t}{r}}$$

$$\Gamma(r,n,t) = \left(\frac{t-r}{n}\right)^{\frac{t-r}{n}}$$

r – lygčių skaičius,

n – kintamųjų skaičius,

t – monomų skaičius;

Taikant algebrinę kriptoanalizę sudaromos lygtys yra daugianarės kvadratinės (*angl*. MQ – multivariate quadratic), o jų sudarymas vadinamas MQ ataka. Atsitiktinių lygčių MQ sistema priklauso NP pilnųjų uždavinių sprendimui. (Garey & Johnson, 1979), kurie praktiškai ir viešai dar nėra išsprendžiami. Manoma, kad algebrinei atakai atsparus S blokas turi tenkinti nelygybę $\Gamma > 2^{32}$. remiantis šiuo kriterijumi buvo ištirtos žinomos laipsninės funkcijos ir pasiūlyti būdai naujų S blokų, su optimaliu algebriniu imunitetu, konstravimui (Nawaz, Gupta & Gong, 2009).

Beveik tobulai netiesinių (*angl*. almost perfect nonlinera) funkcijų atsparumas tradicinei kriptoanalizei yra beveik optimalus. Tačiau šių funkcijų atsparumas algebrinei kriptoanalizei yra labai prastas (Cheon & Lee, 2004). Iš to galime daryti išvada, kad **tos** eksponentinės funkcijos, kurios pasižymi gerais kriterijais tiesinės ir diferencialinės kriptoanalizių atžvilgiu yra netinkamos ir nepasipriešinančios algebrinės kriptoanalizės atžvilgiu.

1.7 ML ATSPARUMAS KRIPTOANALIZEI

1.7.1 TEORINIS ATSPARUMAS DIFERENCIALINIAI KRIPTOANALIZEI

Kaip ir buvo aptarta ankstesniuose skyriuose, kiekvienas ML funkcijos išvesties matricos elementas priklauso nuo visų įvesties matricos elementų. Nagrinėjant ML funkcijos atsparumą diferencialinei kriptoanalizei, ši funkcija išskaidoma į atskiras sudedamąsias dalis, kurios individualiai apskaičiuoja išvesties matricos elementus. Tegu dalinė ML funkcija žymime f_{ij} ir aprašome taip:

$$f_{ij}(X) = \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{is} r_{tj}} = c_{ij} \qquad i, j = 1, 2, \dots, m$$

Šios dalinės funkcijos atitinka daugialaipsnes (*angl*. multi-power) funkcijas, kurios yra kvadratinės kintamųjų l_{is} ir r_{tj} atžvilgiu. Visoje ML funkcijoje iš viso yra m^2 tokių sudėtinių funkcijų. Atskira f_{ij} funkcija yra vaizdavimas iš M į $GF(2^n)/\{0\}$, todėl ji taip pat atitinka tam tikrą S bloką.

Siekiant įvertinti tikėtinos diferencialinės atakos tikimybę, nagrinėjamas ribinis atvejis, kai turimas tik tai vieno bito pokytis įvesties duomenyse ir taikomas minimalus aktyvių S blokų skaičius. Tiriant iteracinių šifrų atsparumą diferencialiniai kriptoanalizei netiesė funkcija (šiuo atveju S blokas) vadinama aktyvia, jei jos įvesties duomenų pokytis yra nenulinis.

Tegu įvesties matricos pokytis ΔX yra $m \times m$ dydžio matrica virš GF(2^n). Kai fiksuojamas vieno bito pokytis tegu ΔX_1, atitinkamai tik vienas pokyčio matricos elementas bus ne nulinis. ML funkcijoje vieno įvesties bito pokytis daro poveikį visiems išvesties matricos elementams. Pažymime, kad vieno bito pokytis Δx suteikiamas elementui x_{uv}. Tada skaičiuojant vieną išvesties matricos lementą, pasikeis tik vienos laipsninės funkcijos rezultatas, o kitos eksponentės nesikeis:

$$f_{ij}(X \oplus \Delta X_1) = (x_{uv} \oplus \Delta x)^{l_{is} r_{tj}} \prod_{t=1}^{m} \prod_{s=1}^{m}{}_{\langle s \neq u, kai\ t=v \rangle}\ x_{st}^{l_{is} r_{tj}} = (x_{uv} \oplus \Delta x)^{l_{is} r_{tj}} \cdot c_{ij}$$

Atlikus apibrėžtas daugybos operacijas gaunamas tiesinio pobūdžio pokytis – pakitęs laipsninės funkcijos rezultatas yra dauginamas iš konstantos c, kuri gaunama sudauginus nepakitusias eksponentes.

Taigi pakeitus vieną įvesties bitą išvesties matricos elemento pokytį lemia laipsninė funkcija ir afinioji transformacija, kuri nesuteikia atsparumo prieš kriptoanalizę.

Skaičiuojant kitus išvesties matricos elementus, bus nagrinėjamas tas pats vieno bito pokytis, kurį apibrėžia aptarta laipsninė funkcija. Visos ML funkcijos pokytį lems m^2 skirtingų laipsninių funkcijų.

ML funkcijoje elementų laipsniu rodikliai gaunami sudauginus po viena elementą iš raktų matricų virš žiedo $Z^*{}_{2^{n-1}}$. Matricos generuojamos tokios, kad neturėtų nulinių elementų ir turėtų sau atvirkštinę matricą. Todėl atlikdami veiksmus su bet kuriais elementais, gautas rezultatas priklausys $[1; 2^n - 2]$ ir visi jie bus vienodai pasiskirstę.

Fiksuojamas įvesties pokytis ΔX, norint gauti konkretų pokyti $\Delta f(X)$ išvesties rezultate, reikia, kad kiekviena sudėtinė ML funkcija f_{ij} įgytų atitinkamą pokytį $\Delta f_{ij}(x_{ij})$. Visos laipsninės funkcijos gali būti skirtingos ir nepriklausomos viena nuo kitos, todėl tikimybė kad ML funkcijos išvestyje bus konkretus pokytis $\Delta f(X)$ galima išreikšti kaip atskirų sudėtinių ML funkcijų f_{ij} atitinkamų pokyčių tikimybių sandaugą:

$$DP_f = P\big(f(X \oplus \Delta X) \oplus f(X) = \Delta f(X)\big) =$$

$$= \prod_{i=1}^{m}\prod_{j=1}^{m} P\left(f_{ij}\big(x_{ij} \oplus \Delta x_{ij}\big) \oplus f_{ij}\big(x_{ij}\big) = \Delta f_{ij}\big(x_{ij}\big)\right)$$

Ribiniu atveju, kai turime tik vieno bito pokyti ΔX_1 šią tikimybę galime išreikšti tokiu būdu:

$$DP_f(\Delta X_1) = \prod_{i=1}^{m}\prod_{j=1}^{m} P\left((x_{uv} \oplus \Delta x)^{l_{iu}r_{vj}} \cdot c_{ij}) \oplus f_{ij}\big(x_{ij}\big) = \Delta f_{ij}\big(x_{ij}\big)\right) =$$

$$= \prod_{i=1}^{m}\prod_{j=1}^{m} P\left((x_{uv} \oplus \Delta x)^{lr_{ij}}) \oplus f_{ij}\big(x_{ij}\big) = \Delta f_{ij}\big(x_{ij}\big)\right)$$

čia $lr_{ij} \in Z^*_{2^n-1}$ ir žymi l_{iu} ir r_{vj} sandaugą. Daugybos iš konstantų c_{ij} galima ir nevertinti, nes jos diferencialinei tikimybei neturi įtakos (Nyberg & Knuden, 1995).

Tikimybė $P\left((x_{uv} \oplus \Delta x)^{lr_{ij}}) \oplus f_{ij}\big(x_{ij}\big) = \Delta f_{ij}\big(x_{ij}\big)\right)$ yra apytiksliai lygi vienos laipsninės funkcijos, kurios laipsnio rodiklis yra lr_{ij} diferencialinei tikimybei $DP_f(lr_{ij})$. Įvesties elementų apibrėžimo sritys gali skirtis, todėl ribiniu atveju, kai turime vieno bito pokytį įvestyje, ML funkcijos diferencialinė tikimybė yra apytiksliai lygi m^2 laipsninių funkcijų diferencialinių tikimybių sandaugai.

Labiausiai tikėtina ML funkcijos diferencialinę tikimybę su vieno bito pokyčiu įvestyje, galima išreikšti tikėtinų laipsninių funkcijų tikimybių sandauga:

$$EDP_f(\Delta X_1) \approx E\left(\prod_{i=1}^{m}\prod_{j=1}^{m} DP_l\big(lr_{ij}\big)\right) = \prod_{i=1}^{m}\prod_{j=1}^{m} EDP_l\big(lr_{ij}\big)$$

Bendru atveju iš raktų matricų paimti elementai – laipsnio rodikliai yra nepriklausomi, todėl vidurkio operatorių galime perkelti prieš daugybos operacijas. Tuomet vidutinė laipsninių funkcijų diferencialinė tikimybė yra vienoda. Pažymėjus: $EDP_l(lr_{ij}) = EDP_l$

$$EDP_f(\Delta X_1) \approx \prod_{i=1}^{m}\prod_{j=1}^{m} EDP_l = {EDP_l}^{m^2}$$

Apribojame ML funkcijos tikėtiną diferencialinę tikimybę, kai pokytis gali būti didesnis nei 1 bitas:

$$EDP_f \leq EDP_f(\Delta X_1) \approx {EDP_l}^{m^2}$$

ML S bloko su transformacija g_K funkcijoje F papildomų netiesinių elementų nėra lyginant su pradine ML funkcija, todėl diferencialinė tikimybės skaičiavimas bus analogiškas, pridedant vienos laipsninės funkcijos diferencialinę tikimybę. ML S bloko funkciją išskaidžius į sudėtines Fij funkcijas, vienos sudėtinės funkcijos pokytį galima išreikšti analogiškai:

$$F_{ij}(X \oplus \Delta X_1) = \left((x_{uv} + k_{uv} + 1) \oplus \Delta x\right)^{l_{iu}r_{vj}} \cdot c_{ij}$$

Perėjus prie diferencialinių tikimybių, iš šios išraiškos negausime įprastos laipsninės funkcijos diferencialinės tikimybės. Ši tikimybė yra susieta su funkcija, kuri yra injektyvus vaizdavimas iš $GF(2^{n-1})$ į $GF(2^n)$. Bendru atveju šią funkciją išreikšti galima taip:

$$ap(x) = (x + k + 1)^l$$
$$x, k \in GF(2^{n-1})$$
$$l \in Z^*_{2^n-1}$$

Vienos tokios funkcijos diferencialinę tikimybę žymėsime $DP_{ap}(k, l)$, nes ji priklauso ir nuo konkrečių parametrų *k* ir *l*.

Viso ML S bloko diferencialinė tikimybė su vieno bito pokyčiu:

$$DP_f(\Delta X_1) = \prod_{i=1}^{m}\prod_{j=1}^{m} P\left(\left(\left((x_{uv} + k_{uv} + 1) \oplus \Delta x\right)^{l_{iu}r_{vj}} \cdot c_{ij}\right) \oplus F_{ij}(x_{ij}) = \Delta F_{ij}(x_{ij})\right) =$$

$$= \prod_{i=1}^{m}\prod_{j=1}^{m} DP_{ap}(k_{uv}, lr_{ij})$$

Bendru atveju tikėtina ML S bloko funkcijos diferencialinę tikimybę apribojame analogiškai kaip ML funkcijos atveju:

$$EDP_{SBF} \leq EDP_{ap}^{m^2}$$

EDP_{ap} yra $ap(x)$ tipo funkcijų vidutinė diferencialinė tikimybė visų parametrų *k* ir *l* atžvilgiu. Teorinė šios tikėtinos diferencialinės tikimybės išraiška nėra žinoma, todėl pasirinkus konkretų baigtinio lauko dydį reikia apskaičiuoti visų galimų variantų vidurkį arba rasti jo įvertį.

1.7.2 ATSPARUMAS PRIEŠ ALGEBRINĘ KRIPTOANALIZĘ

Algebrinės kriptoanalizės sudėtingumas yra tiesiogiai susijęs su atvirkštinio uždavinio sprendimo sudėtingumu. Šiuo atveju atvirkštinis uždavinys formuojamas per funkcijos F apgręžimą. Tegu turima pradinė tekstograma D ir užšifruota šifrograma C, o slaptųjų raktų matricos L, R ir K, su kuriomis funkcija F yra korektiška, nėra žinoma. Tuomet ML S bloko atvirkštinis uždavinys:

$$(L, R, K) = F_{alg}^{-1}((D, C)) \qquad (1.17.)$$

Vienkryptiškumo (landinės vienkryptės funkcijos) savybė yra labai svarbi kriptografijoje. Remiantis keliais fundamentaliais teoriniais rezultatais VKF egzistavimas reiškia ir pseudoatsitiktinio skaičių generatoriaus egzistavimą ir atvirkščiai (Yao, 1982; Hastad, Impagliazzo, Levin & Luby, 1999). Tokios šifravimo funkcijos išvestį atskirti nuo atsitiktinės įmanoma tik su tikimybe lygia 0,5. Todėl ir stengiamasi konstruoti funkcijas, kuo artimesnes VK funkcijoms. Dažnai pasitaikantis būdas tokiai konstrukcijai yra NP pilnosios sudėtingumo klasės naudojimas ir polinominio redukavimo procesas (Garey & Johnson, 1979).

Kai dirbame su sąlyginai mažos eilės lauku $GF(2^n)$ ir operacijų lenteles galima išsaugoti kompiuterio operatyviojoje atmintyje, tai pagal (1.12)

$$\prod_{t=1}^{m}\prod_{s=1}^{m}(d_{st}+k_{st}+1)^{l_{is}r_{tj}} = \prod_{t=1}^{m}\prod_{s=1}^{m}x_{st}^{l_{is}r_{tj}} = c_{ij} \quad i,j = 1,2,\dots,m$$

formulę matoma, kad tiesioginis ML S bloko funkcijos apskaičiavimas yra atliekamas polinominio laiko algoritmu, paremtu *m* ir *n* atžvilgiu. Vieno elemento apskaičiavimui reikalingi $2m^2$ sudėties ir m^2 daugybos operacijų žiede, m^2 kėlimų laipsniu ir m^2 daugybų operacijų lauke.

Visos operacijos yra ekvivalenčios ir saugomos lentelėse, todėl iš viso reikalinga atlikti $5m^2$ operacijų, o visa išvesties matrica sunaudos $5m^4$ operacijų. Asimptotinis ML S bloko funkcijos skaičiavimo laikas yra $O(m^4n)$ eilės (polinominis). Taigi sukonstruotas S blokas tenkina pirmąją VKF (vienkryptės funkcijos) savybę.

Iš 1.12. formulės sudaroma lygčių sistema. Tegu yra žinomos kelios, arba minimaliai - viena pora įvesties ir išvesties matricų porų (D, C), o rasti norima slaptąsias raktų matricas L, R ir K. Šios lygčių sistemos sprendimo sudėtingumas padeda įvertinti pačio ML S bloko apgręžimo funkcijos sudėtingumą.

Tiesioginis m2 lygčių sistemos sprendimo būdas nėra žinomas, todėl atliekamos transformacijos. $GF(2^n)$ multiplikacinės grupės generatorius α išreiškiamas n-1 laipsnio daugianariu virš žiedo Z^2:

$$\alpha = 0\cdot x^{n-1} + 0\cdot x^{n-2} + \cdots + 0\cdot x^2 + 1\cdot x + 0\cdot x^0$$

kurio atžvilgiu galima apskaičiuoti abiejų 1.12. išraiškos pusių diskretinį logaritmą. Gaunama m^2 lygčių sistema:

$$\sum_{t=1}^{m}\sum_{s=1}^{m} l_{is}r_{tj}\ \log_{\alpha}(d_{st}+k_{st}+1) = \log_{\alpha}c_{ij} \quad i,j = 1,2,\dots,m \qquad (1.18.)$$

Kai lauko $GF(2^n)$ parametras n nėra didelis (>8), DLP problema gali būti efektyviai išsprendžiama. Didesnių parametrų, prie kurių šifro saugumas būtų paremtas DLP, parinkti negalime dėl praktiškumo ir šifro greitumo.

Įvedamas keitinys $h_{ij} = \log_\alpha c_{ij}$ ir $z_{st} = \log_\alpha(d_{st} + k_{st} + 1)$, kurio pagalba paslepiama priklausomybė nuo k_{st}. Jei z_{st} būtų rastas, kintamojo k_{st} radimas būtų trivialus: $\alpha^{z_{st}} - d_{st} - 1 = k_{st}$. Sumažinus nežinomųjų skaičių ir gaunama m^2 kelių kintamųjų daugianarių lygčių sistema:

$$\sum_{t=1}^{m}\sum_{s=1}^{m} l_{is} r_{tj} z_{st} = h_{ij}\,, \quad i,j = 1,2,\dots,m. \tag{1.19.}$$

Šioje sistemoje yra 3m2 nežinomųjų $\{l_{is}\}$ $\{r_{tj}\}$ $\{z_{st}\}$ ir m4 kubinių monomų, visi monomai (vienanariai) yra trečios eilės. Ši sistema nėra išretinta kaip AES šifro atveju ir specialūs tokių sistemų sprendimo algoritmai kaip XL ar XSL negali būti taikomi. (Courtois & Pieprzyk, 2002; Courtois, klimov, Patarin & Shamir, 2000).

Išsprendus 1.19. lygčių sistemą ML S blokas būtų sukompromituotas, todėl jo saugumo pagrindimui yra vertinamas sistemos sprendimo sudėtingumas.

Taikant grubią perrinkimo ataka (*angl*. brutal force attack) galima atspėti dalį lygčių sprendinių, t.y. atspėti L', R' ir Z'. Tačiau tai negarantuoja šifro sukompromitavimo, nes sprendinių aibė $(\mathrm{L}, \mathrm{R}, \mathrm{Z})$ konkrečiai (D, C) porai gali būti pakankamai didelė, o visiškam šifro *nulaužimui* reikia rasti tikrąsias slaptųjų raktų matricas. Apibendrinant, ši sistema yra nepilnai apibrėžta (*angl*. underdefined system of equations) ir galimi daugiau nei vienas sprendinys. Šios atakos sudėtingumas yra $O((2^n-1)^{3m\cdot m})$ eilės, nes trijų spėjamų raktų elementai gali įgyti $2^n - 1$ skirtingų reikšmių.

Pritaikant spėjimo ir apskaičiavimo ataką (*angl*. guess and determine attack) yra tikimybė atspėti dvi matricas iš Z, L arba R matricų ir lygčių sistemą supaprastinti iki tiesinių lygčių sistemos. Tada trečioji matrica gali būti apskaičiuota $O(m^6 n)$ eilės laiko algoritmu. Visos atakos bendras sudėtingumas - $O((2^n-1)^{2m\cdot m} m^6 n)$. Kai lauko fiksuojame parametrą $n = 8$, operacijų skaičiuojamumo riba 2^{80} būtų viršyta kai $m \geq 3$.

Jeigu būtų įvestas dar vienas keitinys - $u_{sti} = l_{is} z_{st}$, sumažėtų turimos MP lygčių sistemos kintamųjų skaičius. Atliktus tokia transformaciją gaunama kelių kintamųjų kvadratinė lygčių sistema (*angl*. mulivariante quadratic equation - MQ) virš Z_{2^n-1}:

$$\begin{cases} l_{is} z_{st} = u_{sti}, \qquad i,j = 1,2,\dots,m. \\ \sum_{t=1}^{m}\sum_{s=1}^{m} u_{sti} r_{tj} = h_{ij}\,, \quad i,j = 1,2,\dots,m. \end{cases}$$

Naujų kintamųjų skaičius yra lygus termų skaičiui - m^3. Prie pradinės sistemos pridedamas toks pats skaičius kvadratinių lygčių ir gaunama $m^2 + m^3$ lygčių sistemą su $3m^2 + m^3$ nežinomųjų, $m^3 + m^4$ kvadratinių termų ir m^3 tiesinių termų. Taigi akivaizdu, kad lygčių sistemos dydis tiesiogiai

priklauso nuo lygčių sistemos parametro m. Tegu $m = 4$, tuomet MQ sistema turi 80 lygčių ir 112 nežinomųjų.

Bendru atveju aptarta lygčių sistema yra apibrėžta virš žiedo, o ne baigtinio lauko. Taigi MQ sistemos sprendimas yra sudėtingesnis, nei MQ sistemos virš baigtinio lauko, nes ne visi žiedo elementai gali turėti atvirkštinius elementus daugybos atžvilgiu.

Bendra sprendiminė MQ problema virš bet kurio lauko yra NP pilnojoje klasėje (Garey & Johnson, 1979; Patarin & Goubin, 1997). Universaliausias ir kol kas labiausiai pritaikytas algoritmas spręsti tokias sistemas yra Grobnerio bazių algoritmas (Buchberger, 1985; Albrecht, 2010). Yra laikoma, kad Grobnerio bazių algoritmas yra nepraktiškas atsitiktinai sugeneruotoms MQ sistemoms virš $GF(2^n)$ kai lygčių ir nežinomųjų skaičius viršija 80 (Faugere, 2003). Polinominės eilės metodai nepilnai apibrėžtų MQ sistemų sprendimui (Hashimoto, 2009) taip pat neveiks efektyviai nes sistemose nėra pakankama viršijamas nežinomųjų skaičius palyginus su lygčių skaičiumi. Nors analizuojama lygčių sistemos struktūra priklauso nuo parametrų ir ji nėra atsitiktinai sugeneruota, kol kas nebuvo nustatyti jokie efektyvūs būdai sistemai spręsti.

Aprašytos lygčių sistemos, tiek paprasta, tiek transformuota į MQ, sudarytos su sąlygą, kad turima tik viena pora įvesties ir išvesties matricų.

Pirmos sistemos atveju, jei būtų įvesta papildoma pora, sistemos lygčių skaičius padvigubėtų, bet nežinomųjų skaičius nedidėtų taip sparčiai. Kiekvienai papildomai įvesties ir išvesties matricų porai nežinomųjų $\{l_{is}\}, \{r_{tj}\}$ skaičius nesikeičia ir išlieka pastovus. Nežinomųjų $\{z_{st}\}$ skaičius didėja, nes yra susietas su konkrečiais įvesties matricos elementais. Nagrinėjant p matricų (D, C) porų gauname analogišką lygčių sistemą su pm^2 lygčių, $2m^2$ nežinomųjų $\{l_{is}\}, \{r_{tj}\}$, pm^2 nežinomųjų $\{z_{st}\}$ ir pm^4 kubinių monomų.

Transformuojant paprastą sistemą į MQ virš žiedo Z_{2^n-1} gauname $p(m^2 + m^3)$ lygčių su $m^2 + p(m^2 + m^3)$ nežinomųjų. Išvada – skirtumas tarp nežinomųjų ir lygčių skaičiaus išlieka m^2, nesvarbu kiek įvesties ir išvesties matricų porų yra žinoma.

Tačiau esant situacijai, kai antrosios ar paskesnės eilės (D, C) porų matricų D pokyčiai tarpusavyje nėra ženklus, t.y. dalis matricų elementų sutampa, galima pasiekti situaciją, kai lygčių atsiranda daugiau nei naujų nežinomųjų. Taip galima sudaryti lygčių sistemą kuri jau būtų pilnai ar netgi perteklinai apibrėžta. Tačiau su kiekviena nauja pora kvadratinių lygčių sistema padidės $m^2 + m^3$ lygtimis ir gautoji sistema kai $m \geq 4$ jau bus per didelė, kad ją būtų galima išspręsti per priimtiną laiką.

Pagrindiniai ML S bloko saugumo parametrai yra matricų eilė m, o sudarytą lygčių sistemą transformavus į MQ sistemą virš Z_{2^n-1} svarbus ir baigtinio lauko dydis, priklausantis nuo n. Tam kad spėjimo ir apskaičiavimo ataka būtų neefektyvi sistemos parametrai turi atitikti šiuos saugumo kriterijus: $m \geq 4$ ir $n \geq 3$, tuomet atakos sudėtingumas būtų $O(2^{103})$

ML S bloko apgręžimo problema siekiant nustatyti slaptuosius parametrus, polinominio laiko algoritmu yra suvedama į neišsprendžiamą MQ sistemą, todėl tikėtina jog šifro blokas yra atsparus algebrinei kriptoanalizei.

2 TIRIAMOJI DALIS

(1) Tyrimo prasmė.

Naujos šifravimo sistemos kūrimo proceso neatsiejama dalis yra kriptoanalizė. Jos pagalba atskleidžiamos nesaugios šifro vietos, nustatomi papildomi saugumo parametrai, įvertinami atakos metu gauti rezultatai.

Matricinio laipsnio šifrui taikant algebrinę kriptoanalizę, kaip aprašyta 1.6.2 skyriuje, sudaroma kelių kintamųjų kvadratinė lygčių sistema (*angl*. mulivariante quadratic equation - MQ) virš Z_{2^n-1}.

Viešai nėra paskelbta apie realizuotus optimalius ir stabiliai veikiančius MQ lygčių sprendimo algoritmus. Tačiau keletas universitetų yra sudarę darbo grupes, kurios kuria programines įrangas, pritaikytas spręsti įvairius algebros uždavinius. Šiuo metu yra vystomi bandomieji algoritmai, tačiau jie nėra paprastai realizuojami, efektyvūs ar universalūs uždavinių atžvilgiu. Keletas atvirojo kodo algebros uždavinius galinčių spręsti programų:

- Vašingtono universiteto - SAGA (*nuoroda*: http://www.sagemath.org/index.html)
- Sidnėjaus universiteto – MAGMA (*nuoroda*: http://magma.maths.usyd.edu.au/magma/)

Su programinės įrangos MAGMA kūrėjais buvo užmegztas kontaktas, siekiant gauti prieigą prie programinės įrangos ir esamų algoritmų. Tačiau universitetas yra laikinai suspendavęs viešą priėjimą prie paties projekto. Vietoje to, jų tinklapyje yra realizuota programos aplikacija, kuri geba kompiliuoti užduodamą kodą ir atlikti skaičiavimus, jeigu jie trunka ne daugiau kaip 120 sekundžių (*nuoroda*: http://magma.maths.usyd.edu.au/calc/).

Įvertinus komplikuotą situaciją su programine įranga ir esamų realizuotų panašaus tipo algoritmų sudėtingumą, nuspręsta tyrimą pakreipti kita linkme. Prieš pradedant kurti algoritmą gebantį spręsti ML šifro atveju susidarančias MQ lygčių sistemas, yra prasminga išsiaiškinti, ko galima tikėtis iš rezultatų. Aktualu, kiek priklausomai nuo parametrų gali egzistuoti sprendinių. Jei egzistuos daugiau nei vienas sprendinys, ar bus realu rasti ieškomą tikrąjį sprendinį. Todėl pagrįstai yra reikalingas Matricinio laipsnio šifro funkcijos sprendinių aibės tyrimas. Tyrimui atlikti pasinaudosime pilno perrinkimo ataka.

(2) Aplinka.

Tyrimo metu imituojama grubi pilno perrinkimo ataka (*angl*. brutal force attack) prieš Matricinio laipsnio šifrą. Apsibrėžiame atakai priimtinas sąlygas:

- Yra žinomas šifro algoritmas.

- Turima galimybė kiek norima kartų naudotis algoritmu, t.y. vykdyti užšifravimo operaciją, keičiant slaptuosius raktus, pradines tekstogramas.
- Nėra atakos trukmės laiko apribojimų.
- Tiriamas vienos iteracijos šifras.
- Neanalizuojama g_K transformacijos funkcija.

(3) Sistema.

- Darbinis procesorius - keturių branduolių Intel®Core™ i5 CPU 760@ 2.80 GHz 2.79 GHz
- Laikinoji atmintis – 4,00 GB RAM
- Operacinė sistema – Windows 8.1 Pro 64 bit

(4) Programinė įranga

Tyrime naudojami programos algoritmai realizuoti daugiaplatformės programinės įrangos MATLAB (*Version 7.11.0.584 (R2010b) - 64-bit*) aplinkoje, kodas pateiktas prieduose. Programos tikslas yra įvykdyti pilno perrinkimo ataką, analizuoti gautų rezultatų aibes ir atlikti kitus veiksmus aprašytus tyrimo eigoje.

Norint paruošti kompiuterį darbui su programine įranga, reikia įkelti *MLcryp.m* failą į MATLAB programos darbinę aplinką. Tuomet paleidus MATLAB programinę įrangą užtenka sukompiliuoti kodą (*įkeltą failą*) ir darbiniame lauke kreiptis komanda *MLcrypt*. Toliau programa pateikia siūlomus atlikti veiksmus, kurių trumpiniai yra aprašyti ir detalizuoti iš eilės šiame darbe pateikto tyrimo etapuose.

```
>> MLcryp
 - Raktų generavimas "1"
 - Visų tekstogramų šifravimas "2"
 - Perrinkimo ataka "3"
 - Visų M šifrvimas viena R aibe "4"
 - Visų galimų raktų sugrupavimas "5"
 - Tyrimas su kitais parametrais (3x3) "6"
fx
```

Pagalbiniai failai saugomi kartu su pagrindiniu programinės įrangos failu: *LaipsnioT.txt* ir *DaugybosT.txt* - tai sudaromo daugianarių liekanų lauko operacijų lentelės, naudojamos Matricinio laipsnio šifro realizavime (*jų sudarymas aprašomas (7) ir (8) skiltyje*).

(5) Matematinė struktūra.

Tegu matrica A yra pradinė tekstograma. Tuomet praleidžiama g_K transformacija, panaikinanti nulinius elementus pradinėje tekstogramoje, apibrėžiama taip:

$$\left(\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} + \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} + \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}\right) = \begin{bmatrix} z_{11} & z_{11} \\ z_{11} & z_{11} \end{bmatrix}$$

Matrica $Z = \{z_{ij}\}$ yra transformuota pradinė matrica $A = \{a_{ij}\}$. Akivaizdu, jog turėdami pradinę matricą A ir turėdami matricą Z, transformacijos etapo matricą $D = \{d_{ij}\}$ rasime nesunkiai. Todėl toliau bus kalbama apie Z kaip pradinę matricą, o tyrimo metu bus generuojamos matricos be nulinių reikšmių.

Toliau apibrėžiama, kaip šifro ML funkcija paveikia pradinę tekstogramos matricą:

čia l ir r yra slaptųjų parametrų – laipsnių matricų L ir R elementai.

$$^{\mathrm{L}}Z^{\mathrm{R}} = \begin{bmatrix} {z_{11}}^{l_{11}} \cdot {z_{21}}^{l_{12}} & {z_{12}}^{l_{11}} \cdot {z_{22}}^{l_{12}} \\ {z_{11}}^{l_{21}} \cdot {z_{21}}^{l_{22}} & {z_{12}}^{l_{21}} \cdot {z_{22}}^{l_{22}} \end{bmatrix}^{R}$$

$$^{\mathrm{L}}Z^{\mathrm{R}} = \begin{bmatrix} \left({z_{11}}^{l_{11}} \cdot {z_{21}}^{l_{12}}\right)^{r_{11}} \left({z_{12}}^{l_{11}} \cdot {z_{22}}^{l_{12}}\right)^{r_{21}} & \left({z_{11}}^{l_{11}} \cdot {z_{21}}^{l_{12}}\right)^{r_{12}} \left({z_{12}}^{l_{11}} \cdot {z_{22}}^{l_{12}}\right)^{r_{22}} \\ \left({z_{11}}^{l_{21}} \cdot {z_{21}}^{l_{22}}\right)^{r_{11}} \left({z_{12}}^{l_{21}} \cdot {z_{22}}^{l_{22}}\right)^{r_{21}} & \left({z_{11}}^{l_{21}} \cdot {z_{21}}^{l_{22}}\right)^{r_{12}} \left({z_{12}}^{l_{21}} \cdot {z_{22}}^{l_{22}}\right)^{r_{22}} \end{bmatrix}$$

$$^{\mathrm{L}}Z^{\mathrm{R}} = \begin{bmatrix} {z_{11}}^{l_{11}r_{11}} \cdot {z_{21}}^{l_{12}r_{11}} {z_{12}}^{l_{11}r_{21}} \cdot {z_{22}}^{l_{12}r_{21}} & {z_{11}}^{l_{11}r_{12}} \cdot {z_{21}}^{l_{12}r_{12}} {z_{12}}^{l_{11}r_{22}} \cdot {z_{22}}^{l_{12}r_{22}} \\ {z_{11}}^{l_{21}r_{11}} \cdot {z_{21}}^{l_{22}r_{11}} {z_{12}}^{l_{21}r_{21}} \cdot {z_{22}}^{l_{22}r_{21}} & {z_{11}}^{l_{21}r_{12}} \cdot {z_{21}}^{l_{22}r_{12}} {z_{12}}^{l_{21}r_{22}} \cdot {z_{22}}^{l_{22}r_{22}} \end{bmatrix}$$

$$\begin{bmatrix} {z_{11}}^{l_{11}r_{11}} \cdot {z_{21}}^{l_{12}r_{11}} {z_{12}}^{l_{11}r_{21}} \cdot {z_{22}}^{l_{12}r_{21}} & {z_{11}}^{l_{11}r_{12}} \cdot {z_{21}}^{l_{12}r_{12}} {z_{12}}^{l_{11}r_{22}} \cdot {z_{22}}^{l_{12}r_{22}} \\ {z_{11}}^{l_{21}r_{11}} \cdot {z_{21}}^{l_{22}r_{11}} {z_{12}}^{l_{21}r_{21}} \cdot {z_{22}}^{l_{22}r_{21}} & {z_{11}}^{l_{21}r_{12}} \cdot {z_{21}}^{l_{22}r_{12}} {z_{12}}^{l_{21}r_{22}} \cdot {z_{22}}^{l_{22}r_{22}} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

$$C = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} - \text{šifrograma}$$

Selektyviai šifrogramos matricos elementas gaunamas:

$$\prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{is}r_{tj}} = c_{ij}$$

Toliau pasirenkami ML šifro sistemos parametrai $m = 2$ ir $n = 3$. (*m – sistemoje naudojamų matricų dydis* $m \times m$*, n - lauko parametras* $GF(2^n)$). Pagal juos suformuojamas laukas $GF(2^3)$, o neredukuojamu daugianariu pasirenkamas $x^3 + x + 1$ (*lauką sudaro daugianarių liekanos*).

Toliau sudaromos lauko operacijų lentelės ir užkoduojamos skaitmenimis, tai leis paprasčiau jomis naudotis programiškai. Apibrėžiama papildoma XOR operacijos lentelė:

2.1 lentelė

XOR operacijos lentelė

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

(6) Sudėties operacija lauke.

Sudėties operacija lauke vykdoma sudedant daugianarių koeficientus prie vienodų *x* laipsnių pagal Z_2 taisyklę arba verčiant daugianarius į bitinę išraišką ir dirbant su XOR operacija. Pateikiami abu pavyzdžiai:

a) $(x^2 + x + 1) + (x + 1) = x^2 + (x + x) + (1 + 1) = x^2 + 0 + 0 = x^2$

b) $(x^2 + x + 1) + (x^2 + x + 1) = (x^2 + x^2) + (x + x) + (1 + 1) = 0 + 0 + 0 = 0$

Bitinė išraiška:

a) $(x^2 + x + 1) + (x + 1) \xrightarrow[BITS]{} |1\ 1\ 1| + |0\ 1\ 1| \xrightarrow[XOR]{} |1\ 0\ 0| \xrightarrow[POLYNOM]{} x^2$

b) $(x^2 + x + 1) + (x^2 + x + 1) \xrightarrow[BITS]{} |1\ 1\ 1| + |1\ 1\ 1| \xrightarrow[XOR]{} |0\ 0\ 0| \xrightarrow[POLYNOM]{} 0$

Tokiu būdu užpildoma visa lentelė:

2.2 lentelė

Daugianarių liekanų lauko sudėties operacijų lentelė

+	**0**	**1**	**x**	**x+1**	**x^2**	**x^2+1**	**x^2+x**	**x^2+x+1**
0	0	1	x	x+1	x^2	x^2+1	x^2+x	x^2+x+1
1	1	0	x+1	x	x^2+1	x^2	x^2+x+1	x^2+x
x	x	x+1	0	1	x^2+x	x^2+x+1	x^2	x+1
x+1	x+1	x	1	0	x^2+x+1	x^2+x	x^2+1	x^2
x^2	x^2	x^2+1	x^2+x	x^2+x+1	0	1	x	x+1
x^2+1	x^2+1	x^2	x^2+x+1	x^2+x	1	0	x+1	x
x^2+x	x^2+x	x^2+x+1	x^2	x^2+1	x	x+1	0	1
x^2+x+1	x^2+x+1	x^2+x	x^2+1	x^2	x+1	x	1	0

2.3 lentelė

Sudėties operacijų lentelė užkoduota skaitmenimis

+	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	1	2	3	4	5	6	7
1	1	0	3	2	5	4	7	6
2	2	3	0	1	6	7	4	3
3	3	2	1	0	7	6	5	4
4	4	5	6	7	0	1	2	3
5	5	4	7	6	1	0	3	2
6	6	7	4	5	2	3	0	1
7	7	6	5	4	3	2	1	0

(7) Daugybos operacija lauke.

Šiame daugianarių liekanų lauke daugybos operacija vykdoma kaip įprasta aritmetinė daugianarių daugybos operacija, tiesiog rezultate skaičiuojama liekana apibrėžtu neredukuojamu daugianariu $x^3 + x + 1$ arba analogiškai kaip ir sudėties operacijoje: (i) transformuojama į bitinę išraišką, (ii) taikoma daugybos stulpeliu ideologija, (iii) taikoma XOR operacija.

Pateikiami abu pavyzdžiai:

a) $(x^2+x+1)\cdot(x+1) = x^3+x^2+x+x^2+x+1 = \left(x^3+x+1\right)+(x^2+x^2)+x =$

$$= \left\{\begin{matrix} x^3+x+1 \bmod (x^3+x+1)=0 \\ x^2+x^2 \underset{Z_2}{\to} 0 \end{matrix}\right\} = x$$

b) $(x^2+x+1)\cdot(x^2+x+1) = x^4+x^3+x^2+x^3+x^2+x+x^2+x+1 =$

$= \left(x^3+x+1\right)+(x^2+x^2)+(x^4+x^2+x)+x^3 =$

$= \left(x^3+x+1\right)+(x^2+x^2)+x(x^3+x+1)+x^3 =$

$$= \left\{\begin{matrix} x^3+x+1 \bmod (x^3+x+1)=0 \\ x^2+x^2 \underset{Z_2}{\to} 0 \\ x\left(x^3+x+1\right) \bmod (x^3+x+1) = \mathrm{x}\cdot 0 = 0 \\ pridedame: x+1+x+1 \underset{Z_2}{\to} 0 \end{matrix}\right\} = x^3+x+1+x+1 = x+1$$

arba

a) $(x^2+x+1)\cdot(x+1) \underset{BITS}{\longrightarrow} |1\ 1\ 1| \times |0\ 1\ 1| \to$

$$\begin{array}{ccccc} & & 1 & 1 & 1 \\ & & 0 & 1 & 1 \\ \hline & & 1 & 1 & 1 \\ & 1 & 1 & 1 & \\ 0 & 0 & 0 & & \\ \hline 0 & 1 & 0 & 0 & 1 \end{array}$$

Tada gautą atsakymą redukuojame pasirinkto daugianario bitine išraiška naudodami XOR:

$$\begin{array}{cccc} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ \hline 0 & 0 & 1 & 0 \end{array}$$

0 0 1 0 atitinka x.

b) $(x^2+x+1)\cdot(x^2+x+1) \underset{BITS}{\longrightarrow} |1\ 1\ 1| \times |1\ 1\ 1| \to$

$$\begin{array}{ccccc} & & 1 & 1 & 1 \\ & & 1 & 1 & 1 \\ \hline & & 1 & 1 & 1 \\ & 1 & 1 & 1 & \\ 1 & 1 & 1 & & \\ \hline 1 & 0 & 1 & 0 & 1 \end{array}$$

Redukuojame pasirinkto daugianario paslinkta bitine išraiška naudodami XOR:

$$\begin{array}{ccccc} 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 & 1 \end{array}$$

0 0 1 1 atitinka $x+1$.

Tokiu būdu užpildoma visa lentelė:

2.4 lentelė

Daugianarių liekanų lauko daugybos operacijų lentelė

×	**0**	**1**	**x**	**x+1**	**x^2**	**x^2+1**	**x^2+x**	**x^2+x+1**
0	0	0	0	0	0	0	0	0
1	0	1	x	x+1	x^2	x^2+1	x^2+x	x^2+x+1
x	0	x	x^2	x^2+x	x+1	1	x^2+x+1	x^2+1
x+1	0	x+1	x^2+x	x^2+1	x^2+x+1	x^2	1	x
x^2	0	x^2	x+1	x^2+x+1	x^2+x	x	x^2+1	1
x^2+1	0	x^2+1	1	x^2	x	x^2+x+1	x+1	x^2+x
x^2+x	0	x^2+x	x^2+x+1	1	x^2+1	x+1	x	x^2
x^2+x+1	0	x^2+x+1	x^2+1	x	1	x^2+x	x^2	x+1

2.5 lentelė

Daugybos operacijų lentelė užkoduota skaitmenimis

×	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	3	1	7	5
3	0	3	6	5	7	4	1	2
4	0	4	3	7	6	2	5	1
5	0	5	1	4	2	7	3	6
6	0	6	7	1	5	3	2	4
7	0	7	5	2	1	6	4	3

(8) Kėlimu laipsniu interpretacija lauke.

Šifro algoritme naudojama laipsnio funkcija. Tačiau atsižvelgus į tai, kad šifre nenaudojama elementari algebra $3 \cdot 3 = 9$ ar $3^2 = 9$, o atliekamas daugybos veiksmas lauke: 3×3 ar laipsnio atveju - $3 \times 3 \times 3$ naudojantis apibrėžta daugybos operacija, suformuojama kėlimo laipsniu lentelę remiantis daugybos operacijos lentele:

2.6 lentelė

Kėlimu laipsniu operacijos lentelė

^	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	1	1	1	1	1	1	1
2	2	4	3	6	7	5	1
3	3	5	4	7	2	6	1
4	4	6	5	2	3	7	1
5	5	7	6	3	4	2	1
6	6	2	7	4	5	3	1
7	7	3	2	5	6	4	1

(7), (8) ir (9) punktuose pateiktų skaičiavimų automatizavimas pateiktas Priede Nr.2

(9) Pasiruošimas atakai.

Atakos idėja: tegu yra turima pradinė tekstograma ir jos šifrograma, tuomet mėginama šifruoti pradinę tekstogramą su visais įmanomais slaptaisiais raktais L ir R. Tokiu būdu siekiama surasti tikrąją

raktų porą, kuri sugeneruotų identišką šifrogramą. Tai pilno perrinkimo ataka, todėl tam tikslui sugeneruojama visų galimų slaptųjų raktų matricų aibė, kurią sudaro neišsigimusios (*turinčios atvirkštinę sau*) $m \times m$ dydžio matricos. Raktų matricos elementai priklauso $Z_{2^n-1}^* = \{1, \dots, 2^n - 2\}$.

(10) Metodas. Visų galimų matricų sugeneravimas.

Programinėje įrangoje į metodą kreiptis komanda „1“.

```
Su visais i11 nuo 1 iki 6 vykdyti
  Su visais i12 nuo 1 iki 6 vykdyti
    Su visais i21 nuo 1 iki 6 vykdyti
      Su visais i22 nuo 1 iki 6 vykdyti
      LL1← [ i11 i12; i21 i22]
      Jei LL1 determinantas nelygus 0
      A(:,:,c3) ←LL1
      c3←c3+1
```

(11) Rezultatai. Visų galimų matricų sugeneravimas.

1) Maksimalus generuojamų matricų elemento dydis šiuo atveju - 6.
2) Sugeneruotų matricų skaičius $6^4 = 1296$.
3) Atrinktų neišsigimusių matricų skaičius - 1212.

Sugeneruotų raktų matricų kaupimui sukuriamas trimatis masyvas A. Trimačio masyvo dimensijos:

1. koordinatė - matricos eilutės numeris
2. koordinatė - matricos stulpelio numeris,
3. koordinatė – matricos eilės numeris masyve.

Generavimas trunka ~ 2.64 sekundės.

(12) Pilno perrinkimo ataka.

Imituojama ataka, kurios metu atliekamas pilnas galimų raktų perrinkimas. Raktų poros renkamos iš duomenų masyvo A, t.y. su kiekvienu iš 1.212 vnt. raktu yra išmėginamas poroje kiekvienas raktas iš dubliuotos aibės raktų - 1.212 vnt. Taigi iš viso gaunama 1.468.944 skirtingų kombinacijų, kurias sudaro skirtingos raktų L ir R poros. Akivaizdu, kad perrenkant visus raktus bus rasta ir tikroji raktų pora. Nustatoma laiko kaupimo funkcija, siekiant įvertinti pilno perrinkimo trukmę.

Tikslui pasiekti suformuojami du ciklai ir sukuriamas unikalus metodas veiksmams tarp lauko elementų atlikti. *T_daugybos* ir *T_laipsnio* funkcijos nėra kreipimaisi į metodus su parametru. Tai apibrėžtos matricos (*duomenys imami iš pagalbinių failų LaipsnioT.txt ir DaugybosT.txt*), kurių elementai yra rezultatai atlikto veiksmo tarp eilutės ir stulpelio indekso reikšmių. Pvz.: kreipimasis į *T_laipsnio* matricą su indeksais 2 ir 4 grąžins matricos elementą esantį antroje eilutėje, ketvirtame

stulpelyje, o reikšmė bus lygi lauko elemento „2“ reikšmei pakelta ketvirtuoju laipsniu pagal apibrėžtas lauko operacijas.

(13) Metodas. Pilno perrinkimo ataka.

Programinėje įrangoje į metodą kreiptis komanda „3“.

```
Su visais i nuo 1 iki <A masyvo dydžio> vykdyti
LL1←A(:,:,i)

    darbine_m(1,1) ←
    T_daugybos(T_laipsnio(pradine_m(1,1),LL1(1,1)),T_laipsnio(pradine_m(2,1),LL1(1,2)));
    darbine_m(1,2) ←
    T_daugybos(T_laipsnio(pradine_m(1,2),LL1(1,1)),T_laipsnio(pradine_m(2,2),LL1(1,2)));
    darbine_m(2,1) ←
    T_daugybos(T_laipsnio(pradine_m(1,1),LL1(2,1)),T_laipsnio(pradine_m(2,1),LL1(2,2)));
    darbine_m(2,2) ←
    T_daugybos(T_laipsnio(pradine_m(1,2),LL1(2,1)),T_laipsnio(pradine_m(2,2),LL1(2,2)));

pradine_m2←darbine_m;
darbine_m – išvalomas;

Su visais j nuo 1 iki <A masyvo dydžio> vykdyti
RR1←A(:,:,i)

    darbine_m(1,1) ←
    T_daugybos(T_laipsnio(pradine_m2(1,1),RR1(1,1)),T_laipsnio(pradine_m2(1,2),RR1(2,1)));
    darbine_m(1,2) ←
    T_daugybos(T_laipsnio(pradine_m2(1,1),RR1(1,2)),T_laipsnio(pradine_m2(1,2),RR1(2,2)));
    darbine_m(2,1) ←
    T_daugybos(T_laipsnio(pradine_m2(2,1),RR1(1,1)),T_laipsnio(pradine_m2(2,2),RR1(2,1)));
    darbine_m(2,2) ←
    T_daugybos(T_laipsnio(pradine_m2(2,1),RR1(1,2)),T_laipsnio(pradine_m2(2,2),RR1(2,2)));
Jei šifruota matrica lygi tikrai šifrogramai
        C(:,:,b) ←LL1;
        D(:,:,b) ←RR1;
        b=b+1;
```

(14) Rezultatai. Pilno perrinkimo ataka.

1) Iš viso išanalizuota 1.468.944 vnt. skirtingų raktų porų.
2) Nustatyta, jog įskaitant tikrąją raktų porą, iš viso egzistuoja **528 vnt.** raktų porų tinkančių užšifruoti pradinę tekstogramą ir gauti korektišką šifrogramą.
3) Pilnas perrinkimas truko 23 minutes ir 14 sekundžių.

(15) Teiginys.

Vienai pradinei tekstogramai užšifruoti Matricinio laipsnio šifru egzistuoja 528 vnt. (*tai apie 0.0359% iš 1.468.944 galimų porų*) slaptųjų raktų porų, su kuriomis bus gautos identiškos šifrogramos.

Tuo atveju, jei pradinė tekstograma (įskaitant, jei yra vykdoma, g_K transformaciją) turi vieną eilutę, stulpelį ar įstrižainę, kurią sudaro vienodi elementai, egzistuoja 630 vienetų raktų porų (*tai apie 0.0429% iš 1.468.944 galimų porų*) slaptųjų raktų porų, su kuriomis bus gautos identiškos šifrogramos.

Tuo atveju, jei pradinė tekstograma (įskaitant, jei yra vykdoma, g_K transformaciją) yra sudaryta iš vienodų elementų, egzistuoja 3.600 vnt. (*tai apie 0. 2451% iš 1.468.944 galimų porų*) slaptųjų raktų porų, su kuriomis bus gautos identiškos šifrogramos.

Pabrėžiama, kad 528 vienetų raktų porų tinka užšifruoti tik vienai, tyrimo metu naudotai matricai.

(16) Iškeliama hipotezė.

Remiantis gautais rezultatais, dėl vienai pradinei tekstogramai tinkamų, t.y. formuojančių identiškas šifrogramas, pakankamai didelio raktų kiekio, iškeliama hipotezė, ar ML šifras yra netik teoriškai bet ir praktiškai korektiškas - negalima kolizija ir nesusiformuoja dvi vienodos šifrogramos šifruojant skirtingas pradines tekstogramas.

Atliekama tokia analizė: su viena atsitiktinai parinkta raktų pora yra užšifruojamos visos galimos pradinės tekstogramos ir ieškoma, ar atsitiktinai nesusiformuos ekvivalenčios šifrogramų matricos. Iš viso patikrinama $7^4 = 2.401$ vnt. sugeneruotų matricų. Generuojant matricas neįtraukti „0“ reikšmės elementai, imituojant jau atliktą g_K transformaciją.

(17) Metodas. Visų pradinių tekstogramų šifravimas viena raktų pora.

Programinėje įrangoje į metodą kreiptis komanda „2“.

- *Atliekamas matricų generavimas analogiškas kaip ir raktų generavimo etape, užduodant elementų ribas [1 ... 7].*
- *Pabaigus generavimą, paleidžiamas ciklas, kurio metu visos sugeneruotos matricos yra šifruojamos.*
- *ML šifro taikymas analogiškas kaip ir pilno perrinkimo etape, tik panaikinami ciklai paduodantys skirtingas raktų poras (šifruojama su viena raktų pora).*
- *Visos užšifruotos matricos išsaugomos į trimatį duomenų masyvą CLO.*
- *Suformuojami du ciklai, kurių pagalba vykdoma vienodų šifrogramų paieška:*
  Su visais i nuo 1 iki <CLO masyvo dydžio> vykdyti
  Su visais j nuo (i+1) iki <CLO masyvo dydžio> vykdyti
  Tikrinama ekvivalentiškumas tarp CLO(:,:,i) ir CLO(:,:,j).
- *Radus vienodas matricas į ekraną išvedamas atsakymas su matricų numeriais.*

(18) Rezultatai. Visų pradinių tekstogramų šifravimas viena raktų pora.

- Tyrimas atliktas keletą kartų su skirtingomis raktų poromis.
- Vidutiniškai patikrinimas trunka apie 5.7 sekundės.
- Vienodų šifrogramų matricų nebuvo rasta.

(19) Teiginys.

Su vienetine slaptųjų raktų L ir R pora ML šifre užšifruojamos bet kurios pradinės tekstogramos šifrograma yra vienetinė ir šifruojant bet kurią kitą pradinę tekstograma nebus gauta ekvivalenti šifrograma.

(20) Hipotezė.

Atsižvelgus į teiginį, dėl pradinės tekstogramos ir šifrogramos vientisumo, suformuojama hipotezė: jeigu vienai matricai užšifruoti tinka 528 vnt. raktų porų, ar yra tikimybė, jog iš 528 vnt. raktų porų, bent viena, išskyrus tikrąją raktų porą, tiks korektiškai užšifruoti kitą pradinę tekstogramą.

Formaliai: tegu A - visų galimų pradinių tekstogramų matricų aibė, RL - visų galimų raktų porų matricų aibė. Tuomet $RL_i \subset RL$ yra poaibis, kurį sudaro visos raktų poros tinkančios užšifruoti pradinę tekstogramos matricą $a_i \in$ A ML šifru ir gauti tą pačią šifrogramą. Pažymima, kad $rl_i^{\alpha} \in RL_i$ yra tikroji raktų pora. Iškeliama hipotezė, ar egzistuoja tokia raktų pora $rl_i^{\beta} \in RL_i$, kuri būtų korektiška ir betarpiškai teisinga rakto rl_i^{α} kopija, t.y. $rl_i^{\alpha} \leftrightarrow rl_i^{\beta}$.

Tyrimui atlikti suformuojamas naujas metodas, kuris šifruos visas galimas pradinės tekstogramos matricas, kurių yra $7^4 = 2.401$ vnt. su vienai iš matricų tinkančių raktų porų aibe, kurią sudaro 528 elementų. Tikslas išsiaiškinti, kaip unikaliai pradinei tekstogramai tinkantys tikrųjų raktų klonai elgsis su kitomis pradinėmis tekstogramomis A/a_i .

(21) Metodas. Visų pradinių tekstogramų šifravimas viena raktų aibe.

Programinėje įrangoje į metodą kreiptis komanda „3“.

- *Pradinių tekstogramų matricų generavimas analogiškas kaip ir raktų generavimo etape, užduodant elementų ribas [1 ... 7].*
- *Pabaigus generavimą, paleidžiamas ciklas, kurio metu visos sugeneruotos matricos yra šifruojamos.*
- *Šifravimas vyksta paleidžiant du ciklus, kaip ir ML šifro pilno perrinkimo etape.*

Struktūra:

1) *Sugeneruojamos visos galimos pradinės tekstogramos.*
2) *Paleidžiamas ciklas, paduodama pirma sugeneruota matrica.*
3) *Paduota matrica užšifruojama su tikrąją raktų pora.*
4) *Paduota matrica yra šifruojama su nustatyta raktų aibe (528 vnt.).*
5) *Skaičiuojama kiek raktų iš 528 vnt. yra tinkami šiai matricai, t.y. užšifravo identiškai kaip ir tikroji raktų pora.*
6) *Kartojama nuo 2) punkto.*

(22) Rezultatai.

- Atliktas tyrimas parodė, jog iš vienai matricai tinkančių 528 vnt. raktų matricų porų, kitoms pradinėms tekstogramų matricoms yra tinkami 48 vnt. skirtingų raktų porų iš visų 528 vnt.

Tiksliau: tegu kiekvienai matricai $a_i \in$ A egzistuoja raktų matricų porų poaibis $RL_i \subset RL$, o matricai $a_j \in$ A egzistuoja raktų matricų porų poaibis $RL_j \subset RL$. Tuomet poaibiai RL_iir RL_j turi minimaliai 48 bendrus raktus, tačiau kintant i ir j reikšmėms, nebus gaunamos tos pačios bendrų raktų poros, nors jų kiekis bus tas pats, t.y. 528 vnt. raktų išsibarstymas nėra apibrėžiamas.

- Užfiksuota **6 vnt. išskirčių**, t.y. raktų porų kurios tiko **visoms matricoms**. Detalizuojame keletą rezultatų (*paryškinta tikroji raktų pora*):

2.6 lentelė

Slaptųjų raktų klonų lentelė (1 var.)

C(:,:,1) D(:,:,1)	ans = 1 1 1 2 ans = 5 1 2 1	C(:,:,345) D(:,:,345)	ans = 2 2 2 4 ans = 6 4 1 4	C(:,:,689) D(:,:,689)	ans = 3 3 3 6 ans = 4 5 3 5
C(:,:,1027) D(:,:,1027)	ans = 4 4 4 1 ans = 3 2 4 2	**C(:,:,1371)** **D(:,:,1371)**	**ans =** **5 5** **5 3** **ans =** **1 3** **6 3**	C(:,:,1715) D(:,:,1715)	ans = 6 6 6 5 ans = 2 6 5 6

2.7 lentelė

Slaptųjų raktų klonų lentelė (2 var.)

C(:,:,14) D(:,:,14)	ans = 1 2 4 5 ans = 2 6 5 6	**C(:,:,111)** **D(:,:,111)**	**ans =** **2 4** **1 3** **ans =** **1 3** **6 3**	C(:,:,258) D(:,:,258)	ans = 3 6 5 1 ans = 3 2 4 2
C(:,:,271) D(:,:,271)	ans = 4 1 2 6 ans = 4 5 3 5	C(:,:,418) D(:,:,418)	ans = 5 3 6 4 ans = 6 4 1 4	C(:,:,515) D(:,:,515)	ans = 6 5 3 2 ans = 5 1 2 1

- Pastebėjus, jog šios grupės raktų poros turi **proporcingumą**, mėginame rasti bendrą daugiklį arba išskaičiavimo būdą analiziniu būdu:

2.8 lentelė

Išrikiuotų slaptųjų raktų klonų lentelė (1 var.)

ans = **5 5** **5 3** **ans =** **1 3** **6 3**	ans = 1 1 1 2 ans = 5 1 2 1	ans = 2 2 2 4 ans = 6 4 1 4	ans = 3 3 3 6 ans = 4 5 3 5	ans = 4 4 4 1 ans = 3 2 4 2	ans = 6 6 6 5 ans = 2 6 5 6

2.9 lentelė

Išrikiuotų slaptųjų raktų klonų lentelė (2 var.)

ans = **2 4** **1 3**	ans = 1 2 4 5	ans = 3 6 5 1	ans = 4 1 2 6	ans = 5 3 6 4	ans = 6 5 3 2
ans = **1 3** **6 3**	ans = 2 6 5 6	ans = 3 2 4 2	ans = 4 5 3 5	ans = 6 4 1 4	ans = 5 1 2 1

- Susidarome pagalbinę daugybos operacijos lentelę moduliu 7.

2.10 lentelė

Daugybos operacijos lentelė moduliu 7

*	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

- Nustatome, kad tikroji raktų pora ir klonuotas raktas turi ryšį:

2.11 lentelė

Tikrojo rakto ir jo klono ryšio lentelė

Tikroji raktų pora	Klonuotas raktas
ans = **2 4** **1 3**	ans = 1 2 4 5

*	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	**4**	5	6
2	0	2	4	6	**1**	3	5
3	0	3	6	2	**5**	1	4
4	0	4	1	5	**2**	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

- Nustatomas bendras daugiklis – 4.
- Taip peržiūrimos visos matricos ir gauti rezultatai atskleidė, jog daugiklių parinkimas nėra atsitiktinis. Pateikiama bendra daugiklių schema:

2.12 lentelė

Bendra rakto ir jo klono ryšių lentelė (1 var.)

ans = **5 5** **5 3** **ans =** **1 3** **6 3**	ans = 1 1 1 2 ans = 5 1 2 1	ans = 2 2 2 4 ans = 6 4 1 4	ans = 3 3 3 6 ans = 4 5 3 5	ans = 4 4 4 1 ans = 3 2 4 2	ans = 6 6 6 5 ans = 2 6 5 6
Daugiklis	5 3	6 6	4 2	3 5	2 4
Daugiklių sandauga	15	36	8	15	8

2.13 lentelė

Bendra rakto ir jo klono ryšių lentelė (2 var.)

ans = **2 4** **1 3** **ans =** **1 3** **6 3**	ans = 1 2 4 5 ans = 2 6 5 6	ans = 3 6 5 1 ans = 3 2 4 2	ans = 4 1 2 6 ans = 4 5 3 5	ans = 5 3 6 4 ans = 6 4 1 4	ans = 6 5 3 2 ans = 5 1 2 1
Daugiklis	4 2	5 3	2 4	6 6	3 5
Daugiklių sandauga	8	15	8	36	15

Lentelėse 2.12 ir 2.13. matome, kaip tikroji raktų pora skiriasi nuo savo klonuotų raktų porų parinktomis daugiklių poromis. Nustatyta, kad tų daugiklių tarpusavio sandauga moduliu 7 yra lygi vienetui. (pvz.: $8\ mod\ 7 = 15\ mod\ 7 = 36\ mod\ 7 = 1$). Jokios kitos daugiklių poros virš žiedo $Z^*_{2^n-1}$ neegzistuoja su tokia pačia savybe.

Taigi galime daryti išvadą, jog egzistuoja raktų klonai, gebantis identiškai užšifruoti visas pradines tekstogramų matricas. Suformuojama teorema:

(23) Teorema. Korektiškų raktų kopijų.

Tegu taikomas Matricinio laipsnio šifras $D = \{x_{i,j}\}$ – pradinė tekstogramos matrica virš $GF(2^{n-1})$, slaptasis parametras - neišsigimusių raktų matricų $L = \{l_{i,j}\}$ ir $R = \{r_{i,j}\}$ poros, apibrėžtos virš žiedo $Z^*_{2^n-1} = \{1, \dots, 2^2 - 2\}$, C – šifrogramos matrica virš $GF(2^n)$. Visos matricos yra $m \times m$ eilės.

Suformuojami daugikliai $x, y \in Z_{2^n-1}$, tenkinantys sąlygą: $(x \cdot y) mod (2^n - 1) = 1$, kurių kiekis yra lygus atvirkštinių elementų skaičiui žiede Z_{2^n-1}, apskaičiuojamam Oilerio funkcijos (angl. Euler's totient function) pagalba: $\varphi(2^n - 1) = (2^n - 1) \cdot \prod_{p|(2^n-1)} \left(1 - \frac{1}{p}\right)$.

Tuomet, egzistuoja slaptųjų raktų matricų poros kopijos $L \cdot x = L'$ ir $R \cdot y = R'$, su kuriomis: $f_{L,R}(D) = f_{L',R'}(D) = C$.

(24) Įrodymas.

Tegu šifrogramos elemento išraiška:

$$\prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{is} r_{tj}} = c_{ij}$$

Tuomet su raktų kopijomis $L' = \{l'_{i,j}\}$ ir $R' = \{r'_{i,j}\}$ galioja lygybė:

$$\prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{is} r_{tj}} = \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l'_{is} r'_{tj}} = c_{ij}$$

Tuomet:

$$\prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l_{is} r_{tj}} = \prod_{t=1}^{m} \prod_{s=1}^{m} x_{st}^{l'_{is} r'_{tj}}$$

Sulyginami laipsnių rodikliai

$$l_{is} r_{tj} = l'_{is} r'_{tj}$$

Panaudojame daugiklius $x, y \in Z_{2^n-1}$ ir pagal teoremos sąlygą, $l_{i,j} \cdot x = l'_{ij}$ ir $r_{i,j} \cdot y = r'_{ij}$, gauname:

$$(l_{is} \cdot x)(r_{tj} \cdot y) = l'_{ij} r'_{ij}$$

$$l'_{ij} r'_{ij} = l_{is} r_{tj} \cdot (x \cdot y)$$

Akivaizdu, jog lygybė $l'_{ij} r'_{ij} = l_{is} r_{tj}$ galioja, kai $x \cdot y = 1$.

Atsižvelgus į tai, jog operacijos tarp raktų matricų yra apibrėžtos kaip įprasta matricų daugyba naudojant Z_{2^n-1} aritmetiką, pagrindžiame, jog žiede Z_{2^n-1} sandauga $x \cdot y$ lygi vienetui, kai x ir y yra vienas kitam atvirkštiniai elementai.

(25) Pavyzdys Nr.1

Atvirkštinių elementų skaičius žiede Z_7 yra lygus $\varphi(7) = 7\left(1 - \frac{1}{7}\right) = 6$

2.14 lentelė

Žiedo moduliu 7 atvirkštiniai elementai

	0	**1**	**2**	**3**	**4**	**5**	**6**
0	0	0	0	0	0	0	0
1	0	**1**	2	3	4	5	6
2	0	2	4	6	**1**	3	5
3	0	3	6	2	5	**1**	4
4	0	4	**1**	5	2	6	3
5	0	5	3	**1**	6	4	2
6	0	6	5	4	3	2	**1**

(26) Pavyzdys Nr.2

Teorema išmėginama su didesnės eilės matricomis ir kitu lauku. Nustatomi parametrai: $m = 3$ ir $n = 4$. (m – sistemoje naudojamų matricų dydis $m \times m$, n - lauko parametras $GF(2^n)$). Pagal juos suformuojamas laukas $GF(2^4)$, o neredukuojamu daugianariu pasirenkamas $x^4 + x^3 + x^2 + x + 1$ (*lauką sudaro daugianarių liekanos*).

Kai sistemos parametrai didėja, kartu sudėtingėja daugybos ir laipsnio operacijų lentelių sudarymas. Todėl visas procesas buvo automatizuotas MATLAB programinės įrangos pagalba. Visa tai pateikta Priede Nr.2.

2.15 lentelė

Daugybos operacijų lentelė užkoduota skaitmenimis II var.

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

2.16 lentelė

Kėlimu laipsniu operacijos lentelė II var.

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

Pagal užduotus parametrus atvirkštinių elementų skaičius žiede Z_{15} yra lygus

$$\varphi(15) = 15\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right) = 8$$

2.17 lentelė

Žiedo moduliu 15 atvirkštiniai elementai

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

2.18 lentelė

Lauko GF(2^4) tyrimas

pradine_matrica =	Užšifruota matrica	L_matrica =	R_matrica =
4 15 12	13 11 2	1 11 2	12 5 3
8 12 12	15 1 7	1 1 4	4 13 7
14 5 2	13 6 11	12 4 6	2 8 9

2.19 lentelė

Lauko GF(2^4) klonuoti raktai

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

(27) Hipotezė.

Jeigu iš visų 528 atrinktų raktų porų yra sudaroma 6 narių grupė, t.y. pagrindinė raktų pora ir 5 jos klonai, kaip patvirtina (23) punkto teorema, tuomet iškeliama hipotezė, ar yra įmanoma sugrupuoti visus 528 raktus į tokiais pačiais ryšiais susietas grupes.

Tyrimui atlikti suformuojamas metodas, kuris sugrupuoja visas atrinktas raktų matricas pagal daugiklius apibrėžtus žiede, t.y. imama daugiklių pora, pasirenkama iš eilės galimų raktų pora ir atlikus

daugybos veiksmą moduliu 7 ieškoma į grupę tinkanti kita matricų pora. Jei hipotezė apie sugrupavimą yra teisinga, turėtų būti nustatytos 88 grupės.

(28) Metodas. Programinėje įrangoje į metodą kreiptis komanda „3“.

- *Suformuojami daugiklių duomenų masyvai:*

vekt_daug1←[2 3 4 5 6] ir vekt_daug2← [4 5 2 3 6]

- *Formuojamas ciklas: su visais i nuo 1 iki 88 vykdyti*

Paimama raktų pora ir tikrinama ar nėra priskirta nulinė reikšmė pirmajam matricos elementui (tai požymis, kad matrica jau yra priskirta kažkuriai grupei). Jei matrica nėra priskirta, jai bus suformuojama jos grupė.

- *Formuojamas ciklas: su visais i nuo 1 iki 5 vykdyti*

Paimamas daugiklis iš masyvo ir atliekamas daugybos ir moduliu 7 veiksmas su pirmąją matrica.

- *Formuojamas ciklas: su visais i nuo 1 iki 528 vykdyti*

Ieškoma atitinkama matrica priskyrimui į grupę, t.y. bus tokios reikšmės kaip pirmoji matrica padauginta iš daugiklio ir redukuota moduliu 7.

- *Jei matricų pora randama, jų pirmiesiems elementams priskiriamos nulinės reikšmės.*

Toliau vykdomas ciklas keičiant daugiklius. Pabaigus jį, pereinama prie kitus pradinės matricos paėmimo.

(29) Rezultatai.

Tyrimas parodė, jog hipotezė yra teisinga ir visas tinkančias matricas galima suskirstyti į pagal taisykles nustatytus daugiklius. Dalis rezultatų (*skaičiai yra matricų numeriai masyve*):

<...>
36 170 310 321 461 595
37 178 295 336 453 594
38 177 297 334 454 593
39 179 296 335 452 592
<...>

(30) Teiginys.

Turint šifro algoritmą ir jo parametrus, prieš vykdydami pilno perrinkimo ataką galime pasiruošti raktų porų grupes. Jei yra aktualu perrinkimo trukmė, visų galimų raktų porų aibę galima sumažinti 6 kartus tiriamu atveju, arba $2^n - 2$ karto bendruoju atveju.

DISKUSIJA

Nuo tiriamų matricų dimensijų veiksmų specifika šifre nesikeičia, didėja atliekamų veiksmų skaičius. Tačiau vykdant pilno perrinkimo ataką, trukmė priklauso nuo galimų sprendinių aibės dydžio. Darbe analizuojamos 2x2 dimensijos matricos. Pasirinkus 3x3 dydį, peržengiama atliekamų veiksmų kiekio riba per priimtiną laiką, dirbant su asmeniniu kompiuteriu.

Atlikus pilno perrinkimo ataką, gaunamas santykis, kuris parodo: kiek iš visų galimų slaptųjų raktų aibėje esančių matricų yra tinkamos. Tačiau šio dydžio negalime pritaikyti didesnės dimensijos matricomis, nes aibės generavimo etape yra atrenkamos neišsigimusios matricos ir tikėtina, jog ženkliai išaugusi variantų aibė turės mažesnį procentą tinkamų raktų.

Analizės metu, kuomet yra tikrinamos visos galimos pradinės tekstogramos, su vienai iš tekstogramų tinkančia raktų aibe, paaiškėjo, kad kiekviena iš visų tekstogramų galėjo būti užšifruotos su 48 raktais iš 528 vnt. raktų tinkančių vienai iš tekstogramų. Tačiau verta pabrėžti, jog iš 48 vnt. raktų matricų tinkančių vienai tekstogramai, kitai tiks minimaliai 6 vnt. raktų (klonų), tačiau atsitiktinai gali atsirasti sutapimų ir bendrų raktų bus daugiau nei 6 vnt.

Todėl jei nebūtų buvus pilnavertė galimybė prieiti prie šifravimo sistemos kiek norima kartų, raktų klonavimo savybė nebūtų atrasta. Nes vykdant visų galimų pradinių tekstogramų analizę, paaiškėjo, jog egzistuoja raktų klonai tinkantys be išimčių visoms tekstogramoms. Ši savybė yra unikali ir pakeis šifro saugumo parametrų vertinimą keliais aspektais.

Jeigu saugumo tikimybė buvo vertinama kaip 1 tinkamos raktų poros atspėjimo tikimybė iš visų galimų, tuomet dabar prie tiriamųjų parametrų, tikimybė turėtų būti vertinama kaip 6 tinkamų raktų porų atspėjimo tikimybė. Taigi tinkamų sprendinių aibė padidėja 6 kartus. Tai reikšmingas pokytis, atsižvelgus į tai, jog įveikiama atspėjimo tikimybė yra pakitusi nuo 2^{-80} iki 2^{-128}.

Kitu atveju raktų sugrupavimo savybė galėtų ženkliai sutrumpinti pilno perrinkimo laiką. Įsivaizduojama situacija, tegu yra žinomi šifro parametrai. Šifras naudojamas greitam ir dažnam duomenų šifravimui, todėl sistema nėra sudėtinga. Palengvintu atveju - tegu vienos iteracijos. Peršifravimo periodo saugumo riba yra nustatoma mažesnė nei pilnos perrinkimo aibės pertikrinimo laikas. Tačiau pasiruošti šiai atakai galima sugrupavus visus galimus raktus į grupes (tiriamu atveju iš 528 vnt. į 88 vnt.). Nors tai nepakeis atspėjimo tikimybės, tačiau laiko prasme tikrinti reikia ženkliai mažiau elementų.

IŠVADOS

1. Tyrimas atliktas su 2×2 dydžio matricomis. Padidinus parametrą m bent iki 3, visų galimų raktų matricų aibė padidėtų nuo $1.212^2 = 1.468.944$ iki $39.471.305^2 = 1.557.983.918.403.020$. Normaliomis sąlygomis vien sugeneruoti visas galimas matricas truko 10 valandų 45 minutes. Išskaičiuoti visų jų galimų porų tinkamumą, užtruktu daugiau nei 2^{18} metų. Akivaizdu, jog tokiems procesams atlikti privalu padalinti skaičiavimus skirtingiems procesams ir naudoti super-kompiuterius. Tačiau Priede Nr.3 yra paruoštas programinis kodas, išskaidantis matricas į dalis, ir gebantis atlikti sinchronizuotai padalintą tyrimą.
2. Priklausomai nuo pasirinktų ML šifro sistemos parametrų: m (sistemoje naudojamų matricų dydžio $m \times m$) ir n (lauko parametro $GF(2^n)$) priklauso visų galimų raktų matricų aibės dydis - $(2^n - 2)^{2m}$, iš visos aibės vidutiniškai vienai tekstogramai korektiškai šifruoti egzistuoja apie 0.0359% raktų porų, su kuriomis bus gautos identiškos šifrogramos. Jei pradinė tekstograma (įskaitant, jei yra vykdoma, g_K transformaciją) turi vieną eilutę, stulpelį ar įstrižainę, kurią sudaro vienodi elementai, egzistuoja *apie 0.0429% galimų raktų porų. J*ei pradinė tekstograma (įskaitant, jei yra vykdoma, g_K transformaciją) yra sudaryta iš vienodų elementų, egzistuoja *0. 2451% galimų raktų porų*.
3. Su vienetine slaptųjų raktų L ir R pora ML šifre užšifruojamos bet kurios pradinės tekstogramos šifrograma yra vienetinė ir šifruojant bet kurią kitą pradinę tekstograma nebus gauta ekvivalenti šifrograma.
4. Tegu taikomas Matricinio laipsnio šifras $D = \{x_{i,j}\}$ – pradinė tekstogramos matrica virš $GF(2^{n-1})$, slaptasis parametras - neišsigimusių raktų matricų $L = \{l_{i,j}\}$ ir $R = \{r_{i,j}\}$ poros, apibrėžtos virš žiedo $Z^*_{2^n-1} = \{1, \dots, 2^2 - 2\}$, C – šifrogramos matrica virš $GF(2^n)$. Visos matricos yra $m \times m$ eilės. Suformuojami daugikliai $x, y \in Z_{2^n-1}$, tenkinantys sąlygą: $(x \cdot y) mod(2^n - 1) = 1$, kurių kiekis yra lygus atvirkštinių elementų skaičiui žiede Z_{2^n-1}, apskaičiuojamam Oilerio funkcijos (angl. Euler's totient function) pagalba: $\varphi(2^n - 1) = (2^n - 1) \cdot \prod_{p|(2^n-1)} \left(1 - \frac{1}{p}\right)$. Tuomet, egzistuoja slaptųjų raktų matricų poros kopijos $L \cdot x = L'$ ir $R \cdot y = R'$, su kuriomis: $f_{L,R}(D) = f_{L',R'}(D) = C$.
5. Turint šifro algoritmą ir jo parametrus, prieš vykdydami pilno perrinkimo ataką galime pasiruošti raktų porų grupes. Jei yra aktualu perrinkimo trukmė, visų galimų raktų porų aibę galima sumažinti $2^n - 2$ karto bendruoju atveju.

REKOMENDACIJOS

1. Atlikti išplėstinę analizę, remiantis šiame darbe pasiektais rezultatais, su kelių iteracijų Matricinio laipsnio šifru.
2. Užduoti super-kompiuteriui arba kompiuteriniam tinklui perskaičiuoti tyrimą su didesnės dimensijos matricomis (kodas realizuotas prieduose).
3. Pagal gautus rezultatus, nustatyti kaip kinta galimų raktų kiekis, didėjant matricoms.

LITERATŪRA

1. Anderson, R.; Biham, E. (1996). Two practical and provably secure block ciphers: BEAR and LION. FSE 1996. LNCS 1039, psl. 113-120. Springer Berlin Heidelberg.
2. Bard, G. (2009). Algebraic Cryptanalysis. Springer.
3. Biryukov, A.; Canniere, C.D. (2003). Block ciphers and systems of quadratic equatations. FSE 2003, LNCS 2887 (psl. 274-289). Springer-Verlag.
4. Birkhoff, G.; Bartee, T. (1974). Modern applied algebra. McGraw-Hill.
5. Bernstein, D.J. (2005). Understanding brute force. IL 60607-7045. The university of illinois at Chicago.
6. Burnett, Mark; Foster, James C. (2004). Hacking the Code: ASP.NET Web Application Security. Syngress. ISBN 1-932266-65-8.
7. Chabaud, F.; Vaudenay, S. (1995). Links between differencial and linear cryptanalysis. Advance in Cryptology-EUROCRYPT'94, LNCS, vol. 950 (psl. 356-365). New York: Springer–Verlag.
8. Cid, C. (2004). Some Algebraic Aspects of the Advanced Encryption Standard. Fourth Conference on the Advanced Encryption Standard (AES4) (psl. 58-66). Springer-Verlag.
9. Cid, C.; Murphy, S.; Robshaw, M. (2006). Algebraic Aspects of the Advanced Encryption. Berlin, Heidelberg, New York: Springer Verlag.
10. Courtois, N. (2005). General Principles of Algebraic Attacks and New Design Criteria for Cipher Components. Advanced Encryption Standard – AES, LNCS 3373 (psl. 67-83).
11. Courtois, N; Pieprzyk, J. (2002). Cryptanalysis of Block Ciphers with Overdefined Systems of Equations. Asiacrypt 2002, LNCS 2501 (psl. 267-287). Springer.
12. Courtois, N.; Bard, G.; Wagner, D. (2008). Algebraic and Slide Attacks on KeeLoq. FSE 2008, LNCS 5086 (psl. 97-115). Springer.
13. Courtois, N.; Klimov, A.; Patarin, J.; Shamir, A. (2000). Efficient Algorithms for Solving Overdefined Systems of Multivariante Polynomial Equations. Eurocrypt 2000, LNCS 1807 (psl. 392-407). Springer.
14. Daemen, J.; Knudsen, L.; Rijmen, V. (1997). The Block Cipher Square. Fast Software Encryption. LNCS 1267, psl. 149-165. Spinger-Verlag.
15. Gallian, J. (2010). Contemporary Abstract Algebra, 7 edition. Brooks Cole.
16. Garey, M.; Johnson, D. (1979). Computers and Intractability. A Guide to the Theory of NP-Completeness. San Francisco: Freemann.
17. Hastad, K.; Impagliazzo, R.; Levin, L.; Luby, M. (1999). A pseudorandom generator from any one-way function. Siam journal on Computation, 28(4), 1364-1396.
18. Heys, H. (2001). A tutorial on Linear and Differential Cryptanalysis. Technical Report CORR 2001-17, University of Waterloo, Centre for Applied Cryptographic Research, Department of Combinatorics and Optimization.
19. Kang. J.; Hong, S.; Lee, S.; Yi, O.; Park, C.; Lim, J. (2001). Practical and provable security against differential and linear cryptanalysis for substitunion-permuatation networks. ETRI Journal, 23 (4), 158-167.

20. Lidl, R.; Niederreiter, H. (1983). Finite Fields, Encyclopedia of Mathematics and its Applications, vol. 20. Addison-Wesley.
21. Luksys, K.; Nefas, P. (2008). Matric Power S-Box Analysis. “Information Science And Computing”, book 4 “Advanced Studies in Software and Knowledge Engineering”, (psl. 97-102).
22. Luksys, K.; Sakalauskas, E.,; Venckauskas, A. (2012). Implementation Analysis of Matrix Power Cipher in Embedded Systems. Electronics and Electrical Engineering (2 (118)), (psl. 95-98).
23. Luksys, K. (2013). Matricinio laipsnio šifras ir jo analizė. Daktaro disertacija. Kaunas: KTU.
24. Nyberg, K.; Knudsen, L. (1995). Provable security against a differentianl attack. Journal of Cryptology, 8 (1), 27-38.
25. Owens, J.; Matthews, J. (2008). A Study of Passwords and Methods Used in Brute-Force SSH Attacks. Clarkson Univesity. Postdam, New York.
26. Sakalauskas, E.; Luksys, K. (2007). Matrix Power S-Box Construction. Cryptology ePrint Archive: Report, no.214 (2007), http://eprint.iacr.org/2007/214.
27. Sakalauskas, E.; Luksys, K. (2012). The Matrix Power Function and its Application to Block Cipher S-box Construction. International Journal of Innovative Computing, information and Control, 8(4), 2655-2664.
28. Tavares, S. E.; Heys, H.M. (1995). Avalanche Characteristics of Subsitution-Permutation Encryption Networks. IEEE Transactions on Computers, 44(9), 1131-1139.
29. van der Varden, B. (1976). Algebra. Maskva: Nauka.
30. Wiener, M.J. (2004). The full cost of cryptoanalytic attacks, Journal of Cryptology 17, 105-124, ISSN 0933-2790.

PRIEDAS NR.1. PROGRAMOS KODAS

```
darbo_tipas=input(' - Raktų generavimas "1" \n - Visų tekstogramų šifravimas "2" \n -
Perrinkimo ataka "3" \n - Visų M šifrvimas viena R aibe "4" \n - Visų galimų raktų
sugrupavimas "5" \n - Tyrimas su kitais parametrais (3x3) "6" \n');
%_____________________________________________________________________%
if(darbo_tipas==1)
%_____________Matricu generavimas_____________________________________%
c1=input('Įveskite maksimalų raktu matricos elemento dydį: \n');
clear A  LL1; c2=1; c3=1; tic
hhh = waitbar(0,'Generuojamos visos galimos raktu matricos...');
for i11 = 1:c1
    for i12 = 1:c1
        for i21 = 1:c1
            for i22 = 1:c1
                 LL1=[i11 i12; i21 i22];
                 c2=c2+1;
                    if det(LL1)~=0;
                        A(:,:,c3)=LL1;
                        c3=c3+1;
                    end
             waitbar(c2/c1^4)
            end
        end
    end
end
delete(hhh);
display('Iš viso ištirta matricų:'); disp(c2-1);
display('Atrinktos matricos, kurių DET ~= 0:'); disp(c3-1);
display('Generavimas užtruko:'); disp(toc);
save('C:\Users\202020140114b\Documents\MATLAB\LL1.mat', 'A');
beep;
end
%_____________________________________________________________________%
if(darbo_tipas==2)
%_____________Visu matricu sifravimas vienu raktu_____________________%
c1=input('Įveskite maksimalų tekstogramos matricos elemento dydį: \n');
t1=1; tic
 for i11 = 1:c1
     for i12 = 1:c1
         for i21 = 1:c1
             for i22 = 1:c1
                 CC1=[i11 i12; i21 i22];
                 CL(:,:,t1)=CC1;
                 t1=t1+1;
             end
         end
     end
 end
 save('C:\Users\202020140114b\Documents\MATLAB\CL.mat', 'CL');
    Cs=size(CL,3);
    T_laipsnio=load('LaipsnioT.txt');
    T_daugybos=load('DaugybosT.txt');
    L_matrica=[2 4; 1 3];
    R_matrica=[1 3; 6 3];
    CLO=zeros(2,2,Cs);
    LL1=L_matrica;
    RR1=R_matrica;
    for o = 1:Cs
    CC1=CL(:,:,o);

darbine_matrica(1,1)=T_daugybos(T_laipsnio(CC1(1,1),LL1(1,1)),T_laipsnio(CC1(2,1),LL1(1,2))
);

darbine_matrica(1,2)=T_daugybos(T_laipsnio(CC1(1,2),LL1(1,1)),T_laipsnio(CC1(2,2),LL1(1,2))
);

darbine_matrica(2,1)=T_daugybos(T_laipsnio(CC1(1,1),LL1(2,1)),T_laipsnio(CC1(2,1),LL1(2,2))
);
```

```
darbine_matrica(2,2)=T_daugybos(T_laipsnio(CC1(1,2),LL1(2,1)),T_laipsnio(CC1(2,2),LL1(2,2))
);
    CC2=darbine_matrica;
    darbine_matrica=[];

darbine_matrica(1,1)=T_daugybos(T_laipsnio(CC2(1,1),RR1(1,1)),T_laipsnio(CC2(1,2),RR1(2,1))
);

darbine_matrica(1,2)=T_daugybos(T_laipsnio(CC2(1,1),RR1(1,2)),T_laipsnio(CC2(1,2),RR1(2,2))
);

darbine_matrica(2,1)=T_daugybos(T_laipsnio(CC2(2,1),RR1(1,1)),T_laipsnio(CC2(2,2),RR1(2,1))
);

darbine_matrica(2,2)=T_daugybos(T_laipsnio(CC2(2,1),RR1(1,2)),T_laipsnio(CC2(2,2),RR1(2,2))
);
    CLO(:,:,o)=darbine_matrica;
    darbine_matrica=[];
    end
    for oP = 1:Cs
        for oP2 = (oP+1):Cs
            if isequal(CLO(:,:,oP), CLO(:,:,oP2))==true;
            display('Rastos dvi vienodai užšifruotos matricos'); disp(oP); disp(oP2);
            end
        end
    end
    display('Nebuvo vienodai užšifruotų matricų naudojant tuos pačius raktus');
    CLOs=size(CLO,3); display('Patikrint matricų kiekis: '); disp(Cs); disp(CLOs);
    display('Trukmė: '); disp(toc);
    beep;
end
%_____________________________________________________________________%
if(darbo_tipas==3)
%_____________Perrinkimo ataka_____________________________________%
clear A;
load('C:\Users\202020140114b\Documents\MATLAB\LL1.mat');
T_laipsnio=load('LaipsnioT.txt');
T_daugybos=load('DaugybosT.txt');
L_matrica=[5 2; 3 4];    %load('LaipsnisL.txt');
R_matrica=[4 3; 6 5];    %load('LaipsnisR.txt');
pradine_matrica=[1 3; 4 3];
sifruota_matrica=[3 7; 5 3];

MasyvoDydis=size(A,3);
%reserve=(MasyvoDydis^2); E=zeros(2,2,reserve); F=zeros(2,2,reserve);
EF=zeros(2,2,reserve);
clear C D LL1 RR1; %E F EF
C=zeros(2,2,MasyvoDydis); D=zeros(2,2,MasyvoDydis); % 1.5983e+003
a=0; c4=MasyvoDydis; b=1; b2=1;

tic
hhh2 = waitbar(0,'Vykdomas perrinkimas');

for i = 1:c4
    LL1=A(:,:,i);
    darbine_matrica(1,1) =
T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(1,1)),T_laipsnio(pradine_matrica(2,1),LL1(1,
2)));
    darbine_matrica(1,2) =
T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(1,1)),T_laipsnio(pradine_matrica(2,2),LL1(1,
2)));
    darbine_matrica(2,1) =
T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(2,1)),T_laipsnio(pradine_matrica(2,1),LL1(2,
2)));
    darbine_matrica(2,2) =
T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(2,1)),T_laipsnio(pradine_matrica(2,2),LL1(2,
2)));
    pradine_matrica2=darbine_matrica;
    darbine_matrica=[];
```

```
    for j = 1:c4
        RR1=A(:,:,j);
             if isequal(LL1, L_matrica)==true && isequal(RR1, R_matrica)==true;
             display('Tarp visų sugeneruotų raktų yra ir tikroji raktų pora'); disp(i);
disp(j);
             end
        a=a+1;
    darbine_matrica(1,1) =
T_daugybos(T_laipsnio(pradine_matrica2(1,1),RR1(1,1)),T_laipsnio(pradine_matrica2(1,2),RR1(
2,1)));
    darbine_matrica(1,2) =
T_daugybos(T_laipsnio(pradine_matrica2(1,1),RR1(1,2)),T_laipsnio(pradine_matrica2(1,2),RR1(
2,2)));
    darbine_matrica(2,1) =
T_daugybos(T_laipsnio(pradine_matrica2(2,1),RR1(1,1)),T_laipsnio(pradine_matrica2(2,2),RR1(
2,1)));
    darbine_matrica(2,2) =
T_daugybos(T_laipsnio(pradine_matrica2(2,1),RR1(1,2)),T_laipsnio(pradine_matrica2(2,2),RR1(
2,2)));
                if isequal(sifruota_matrica, darbine_matrica)==true;
                    C(:,:,b)=LL1;
                    D(:,:,b)=RR1;
                    b=b+1;
    %else E(:,:,b2)=LL1; F(:,:,b2)=RR1; EF(:,:,b2)=darbine_matrica; b2=b2+1;
                end
    waitbar(a/c4^2)
    darbine_matrica=[];
    end
    pradine_matrica2=[];
end

    save('C:\Users\202020140114b\Documents\MATLAB\C.mat', 'C');
    save('C:\Users\202020140114b\Documents\MATLAB\D.mat', 'D');
  % save('C:\Users\202020140114b\Documents\MATLAB\E2.mat', 'E');
  % save('C:\Users\202020140114b\Documents\MATLAB\F2.mat', 'F');
  % save('C:\Users\202020140114b\Documents\MATLAB\EF2.mat', 'EF');
display('Iš viso patikrinta variantų: '); disp(a)
display('Tinkantys variantai: '); disp(b-1)
display('Procentaliai: '); disp((b-1)/a*100)
display('Skaičiavimai užtruko: '); disp(toc)
beep;
delete(hhh2);
end
%_____________________________________________________________________%
if(darbo_tipas==4)
%_____________Visų matricų šifravimas viena raktų aibe_____________%
load('C:\Users\202020140114b\Documents\MATLAB\CL.mat');
load('C:\Users\202020140114b\Documents\MATLAB\C.mat');
load('C:\Users\202020140114b\Documents\MATLAB\D.mat');
T_laipsnio=load('LaipsnioT.txt');
T_daugybos=load('DaugybosT.txt');
L_matrica=[2 4; 1 3];    %load('LaipsnisL.txt');
R_matrica=[1 3; 6 3];    %load('LaipsnisR.txt');

CLsize=size(CL,3); CDsize=528; %size(C,3);
TTF=zeros(1267728,1); a=1; ff=1;

hhh3 = waitbar(0,'Vykdomas šifravimas visų matricų... '); tic
    for o = 1:CLsize
        TT1=CL(:,:,o);
        LL1=L_matrica;
        RR1=R_matrica;
    darbine_matrica(1,1) =
T_daugybos(T_laipsnio(TT1(1,1),LL1(1,1)),T_laipsnio(TT1(2,1),LL1(1,2)));
    darbine_matrica(1,2) =
T_daugybos(T_laipsnio(TT1(1,2),LL1(1,1)),T_laipsnio(TT1(2,2),LL1(1,2)));
    darbine_matrica(2,1) =
T_daugybos(T_laipsnio(TT1(1,1),LL1(2,1)),T_laipsnio(TT1(2,1),LL1(2,2)));
    darbine_matrica(2,2) =
T_daugybos(T_laipsnio(TT1(1,2),LL1(2,1)),T_laipsnio(TT1(2,2),LL1(2,2)));
    TT2=darbine_matrica;
```

```
    darbine_matrica=[];
    darbine_matrica(1,1) =
T_daugybos(T_laipsnio(TT2(1,1),RR1(1,1)),T_laipsnio(TT2(1,2),RR1(2,1)));
    darbine_matrica(1,2) =
T_daugybos(T_laipsnio(TT2(1,1),RR1(1,2)),T_laipsnio(TT2(1,2),RR1(2,2)));
    darbine_matrica(2,1) =
T_daugybos(T_laipsnio(TT2(2,1),RR1(1,1)),T_laipsnio(TT2(2,2),RR1(2,1)));
    darbine_matrica(2,2) =
T_daugybos(T_laipsnio(TT2(2,1),RR1(1,2)),T_laipsnio(TT2(2,2),RR1(2,2)));
    TTMa=darbine_matrica;
    darbine_matrica=[];
    TT2=[];
    b=1;
    b2=1;
    %display('Matricos Nr.'); disp(o);
    for i = 1:CDsize
    LL1=C(:,:,i);
    RR1=D(:,:,i);
    darbine_matrica(1,1) =
T_daugybos(T_laipsnio(TT1(1,1),LL1(1,1)),T_laipsnio(TT1(2,1),LL1(1,2)));
    darbine_matrica(1,2) =
T_daugybos(T_laipsnio(TT1(1,2),LL1(1,1)),T_laipsnio(TT1(2,2),LL1(1,2)));
    darbine_matrica(2,1) =
T_daugybos(T_laipsnio(TT1(1,1),LL1(2,1)),T_laipsnio(TT1(2,1),LL1(2,2)));
    darbine_matrica(2,2) =
T_daugybos(T_laipsnio(TT1(1,2),LL1(2,1)),T_laipsnio(TT1(2,2),LL1(2,2)));
    TT2=darbine_matrica;
    darbine_matrica=[];
    darbine_matrica(1,1) =
T_daugybos(T_laipsnio(TT2(1,1),RR1(1,1)),T_laipsnio(TT2(1,2),RR1(2,1)));
    darbine_matrica(1,2) =
T_daugybos(T_laipsnio(TT2(1,1),RR1(1,2)),T_laipsnio(TT2(1,2),RR1(2,2)));
    darbine_matrica(2,1) =
T_daugybos(T_laipsnio(TT2(2,1),RR1(1,1)),T_laipsnio(TT2(2,2),RR1(2,1)));
    darbine_matrica(2,2) =
T_daugybos(T_laipsnio(TT2(2,1),RR1(1,2)),T_laipsnio(TT2(2,2),RR1(2,2)));
    TT2=[]; a=a+1;
          if isequal(TTMa, darbine_matrica)==true;
          TTF(ff,1)=i;
          ff=ff+1;
          end

   waitbar(a/(CLsize*CDsize))
   darbine_matrica=[];
   end
   pradine_matrica2=[];
   matrixKeys(o,1)=ff-1;
   end
   display('Skaičiavimai užtruko: ');  disp(toc);

   TTF(TTF==0)=[];
   sCale = sortrows(tabulate(TTF), -2);  %compute sorted frequency table
   tOp = sCale(1:10, 1:2)                %take the top 10

   save('C:\Users\202020140114b\Documents\MATLAB\TTF.mat', 'TTF');
   save('C:\Users\202020140114b\Documents\MATLAB\MK.mat', 'matrixKeys');
   delete(hhh3);
end
%_____________________________________________________________________%
if(darbo_tipas==5)
%_____________Visų galimų raktų sugrupavimas_____________________%
    load('C:\Users\202020140114b\Documents\MATLAB\C.mat');
    load('C:\Users\202020140114b\Documents\MATLAB\D.mat');

    clear GRP Ck Dk Ts Ts2 Pal GS1 GS2 GS1a GS2a;
    for val = 1:1212  %528
        Ck(:,:,val)=C(:,:,val);
        Dk(:,:,val)=D(:,:,val);
    end
    vekt_daug1=[2 3 4 5 6];
    vekt_daug2=[4 5 2 3 6];
```

```
    Ts=size(vekt_daug1,2);
    Ts2=size(Ck,3);
    tic
    y1=1;
    for Pal = 1:(Ts2/Ts+1)  %grupės numeris
        dNr=1;
            if (Ck(1,1,y1)~=0 && Dk(1,1,y1)~=0)
                GRP(Pal, dNr)=y1;
                GS1=Ck(:,:,y1);
                GS2=Dk(:,:,y1);
                Ck(1,1,y1)=0;
                Dk(1,1,y1)=0;
                dNr=dNr+1;
                for o = 1:Ts          %rakto numeris
                    daugiklis1=vekt_daug1(o);
                    daugiklis2=vekt_daug2(o);
                    GS1a=GS1*daugiklis1; GS1a=mod(GS1a,7);
                    GS2a=GS2*daugiklis2; GS2a=mod(GS2a,7);
                    for y2 = 1:Ts2
                        if (Ck(1,1,y2)~=0 && Dk(1,1,y2)~=0)
                            if isequal(Ck(:,:,y2), GS1a)==true && isequal(Dk(:,:,y2),
GS2a)==true;
                                GRP(Pal, dNr)=y2;
                                Ck(1,1,y2)=0;
                                Dk(1,1,y2)=0;
                                dNr=dNr+1;
                            end
                        end
                    end
                end
            end
           y1=y1+1;
    end
    toc
    beep
end

%_____________________________________________________________________%
if(darbo_tipas==6)
%_____________Tyrimas su kitais parametrais_____________________%

T_laipsnio=load('LaipsnioT2.txt');
T_daugybos=load('DaugybosT2.txt');
pradine_matrica=[1 11 3; 6 9 4; 10 4 1];
L_matrica=[1 11 2; 1 1 4; 12 4 6];
R_matrica=[12 5 3; 4 13 7; 2 8 9];
%sifruota_matrica=[6 4 1; 4 6 2; 2 5 3];

LL1=L_matrica*2; LL1=mod(LL1,15);
RR1=R_matrica*8; RR1=mod(RR1,15);

%LL1=L_matrica;
%RR1=R_matrica;

darbine_matrica(1,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(1,1)),T_laipsnio(pradine_matrica(
2,1),LL1(1,2))),T_laipsnio(pradine_matrica(3,1),LL1(1,3)));
darbine_matrica(1,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(1,1)),T_laipsnio(pradine_matrica(
2,2),LL1(1,2))),T_laipsnio(pradine_matrica(3,2),LL1(1,3)));
darbine_matrica(1,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,3),LL1(1,1)),T_laipsnio(pradine_matrica(
2,3),LL1(1,2))),T_laipsnio(pradine_matrica(3,3),LL1(1,3)));

darbine_matrica(2,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(2,1)),T_laipsnio(pradine_matrica(
2,1),LL1(2,2))),T_laipsnio(pradine_matrica(3,1),LL1(2,3)));
darbine_matrica(2,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(2,1)),T_laipsnio(pradine_matrica(
2,2),LL1(2,2))),T_laipsnio(pradine_matrica(3,2),LL1(2,3)));
```

```
darbine_matrica(2,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,3),LL1(2,1)),T_laipsnio(pradine_matrica(
2,3),LL1(2,2))),T_laipsnio(pradine_matrica(3,3),LL1(2,3)));

darbine_matrica(3,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(3,1)),T_laipsnio(pradine_matrica(
2,1),LL1(3,2))),T_laipsnio(pradine_matrica(3,1),LL1(3,3)));
darbine_matrica(3,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(3,1)),T_laipsnio(pradine_matrica(
2,2),LL1(3,2))),T_laipsnio(pradine_matrica(3,2),LL1(3,3)));
darbine_matrica(3,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,3),LL1(3,1)),T_laipsnio(pradine_matrica(
2,3),LL1(3,2))),T_laipsnio(pradine_matrica(3,3),LL1(3,3)));

darbine_matrica;
pradine_matrica=darbine_matrica;
darbine_matrica=[];

darbine_matrica(1,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),RR1(1,1)),T_laipsnio(pradine_matrica(
1,2),RR1(2,1))),T_laipsnio(pradine_matrica(1,3),RR1(3,1)));
darbine_matrica(1,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),RR1(1,2)),T_laipsnio(pradine_matrica(
1,2),RR1(2,2))),T_laipsnio(pradine_matrica(1,3),RR1(3,2)));
darbine_matrica(1,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),RR1(1,3)),T_laipsnio(pradine_matrica(
1,2),RR1(2,3))),T_laipsnio(pradine_matrica(1,3),RR1(3,3)));

darbine_matrica(2,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(2,1),RR1(1,1)),T_laipsnio(pradine_matrica(
2,2),RR1(2,1))),T_laipsnio(pradine_matrica(2,3),RR1(3,1)));
darbine_matrica(2,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(2,1),RR1(1,2)),T_laipsnio(pradine_matrica(
2,2),RR1(2,2))),T_laipsnio(pradine_matrica(2,3),RR1(3,2)));
darbine_matrica(2,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(2,1),RR1(1,3)),T_laipsnio(pradine_matrica(
2,2),RR1(2,3))),T_laipsnio(pradine_matrica(2,3),RR1(3,3)));

darbine_matrica(3,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(3,1),RR1(1,1)),T_laipsnio(pradine_matrica(
3,2),RR1(2,1))),T_laipsnio(pradine_matrica(3,3),RR1(3,1)));
darbine_matrica(3,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(3,1),RR1(1,2)),T_laipsnio(pradine_matrica(
3,2),RR1(2,2))),T_laipsnio(pradine_matrica(3,3),RR1(3,2)));
darbine_matrica(3,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(3,1),RR1(1,3)),T_laipsnio(pradine_matrica(
3,2),RR1(2,3))),T_laipsnio(pradine_matrica(3,3),RR1(3,3)));

display('Užšifruota matrica'); disp(darbine_matrica);

end
```

PRIEDAS NR.2. LAUKO SUDARYMO AUTOMATIZAVIMAS

GF(2^n) lauko neredukuojama daugianarį galima surasti generatoriaus pagalba, tinklapis: http://theory.cs.uvic.ca/gen/poly.html (formuoja bitine išraiška). Kadangi galioja komutatyvumas skaičiuojama tik pagrindinės daugianarių sąveikos. Atlikti daugybos veiksmus galime MATLAB programinėje įrangoje, naudodamiesi komanda *expand*, kuri tinkamai išskirsto dauginamuosius, pvz.:

```
syms x;
a=((x^3));

disp ('2'); disp(expand((x)*a));
disp ('3'); disp(expand((x+1)*a));
disp ('4'); disp(expand((x^2)*a));
disp ('5'); disp(expand((x^2+1)*a));
disp ('6'); disp(expand((x^2+x)*a));
disp ('7'); disp(expand((x^2+x+1)*a));
disp ('8'); disp(expand((x^3)*a));
disp ('9'); disp(expand((x^3+1)*a));
disp ('10'); disp(expand((x^3+x)*a));
disp ('11'); disp(expand((x^3+x+1)*a));
disp ('12'); disp(expand((x^3+x^2)*a));
disp ('13'); disp(expand((x^3+x^2+1)*a));
disp ('14'); disp(expand((x^3+x^2+x)*a));
disp ('15'); disp(expand((x^3+x^2+x+1)*a));
```

Rezultatų redukavimas atliekamas rankiniu būdų, tačiau patartina iš anksto išsiskaičiuoti aukštesnių eilių redukcijas.

		$x^4+x^3+x^2+x+1$			
	x	1	1	1	
		1	**1**	**1**	
1	**1**	1	1	1	1
2	**x**	x	x	x	2
3	**x+1**	x+1	x+1	x+1	3
4	**x^2**	x^2	x^2	x^2	4
5	**x^2+1**	x^2+1	x^2+1	x^2+1	5
6	**x^2+x**	x^2+x	x^2+x	x^2+x	6
7	**x^2+x+1**	x^2+x+1	x^2+x+1	x^2+x+1	7
8	**x^3**	x^3	x^3	x^3	8
9	**x^3+1**	x^3+1	x^3+1	x^3+1	9
10	**x^3+x**	x^3+x	x^3+x	x^3+x	10
11	**x^3+x+1**	x^3+x+1	x^3+x+1	x^3+x+1	11
12	**x^3+x^2**	x^3+x^2	x^3+x^2	x^3+x^2	12
13	**x^3+x^2+1**	x^3+x^2+1	x^3+x^2+1	x^3+x^2+1	13
14	**x^3+x^2+x**	x^3+x^2+x	x^3+x^2+x	x^3+x^2+x	14
15	**x^3+x^2+x+1**	x^3+x^2+x+1	x^3+x^2+x+1	x^3+x^2+x+1	15
	x	2	2	2	2
		x	**x**	**x**	**x**
2	**x**	x^2	x^2	x^2	4
3	**x+1**	x^2+x	x^2+x	x^2+x	6
4	**x^2**	x^3	x^3	x^3	8
5	**x^2+1**	x^3+x	x^3+x	x^3+x	10
6	**x^2+x**	x^3+x^2	x^3+x^2	x^3+x^2	12
7	**x^2+x+1**	x^3+x^2+x	x^3+x^2+x	x^3+x^2+x	14
8	**x^3**	x^4	x^4	x^3+x^2+x+1	15
9	**x^3+1**	x^4+x	x^4+x	x^3+x^2+1	13
10	**x^3+x**	x^4+x^2	x^4+x^2	x^3+x+1	11
11	**x^3+x+1**	x^4+x^2+x	x^4+x^2+x	x^3+1	9
12	**x^3+x^2**	x^4+x^3	x^4+x^3	x^2+x+1	7
13	**x^3+x^2+1**	x^4+x^3+x	x^4+x^3+x	x^2+1	5
14	**x^3+x^2+x**	x^4+x^3+x^2	x^4+x^3+x^2	x+1	3
15	**x^3+x^2+x+1**	x^4+x^3+x^2+x	x^4+x^3+x^2+x	1	1
	x	3	3	3	3
		x+1	**x+1**	**x+1**	**x+1**
3	**x+1**	x^2+x+x+1	x^2+1	x^2+1	5
4	**x^2**	x^3+x^2	x^3+x^2	x^3+x^2	12
5	**x^2+1**	x^3+x^2+x+1	x^3+x^2+x+1	x^3+x^2+x+1	15

6	**x^2+x**	$x^3+x^2+x^2+x$	x^3+x	x^3+x	10
7	**x^2+x+1**	$x^3+x^2+x^2+x+x+1$	x^3+1	x^3+1	9
8	**x^3**	x^4+x^3	x^4+x^3	x^2+x+1	7
9	**x^3+1**	x^4+x^3+x+1	x^4+x^3+x+1	x^2	4
10	**x^3+x**	$x^4+x^3+x^2+x$	$x^4+x^3+x^2+x$	1	1
11	**x^3+x+1**	$x^4+x^3+x^2+x+x+1$	$x^4+x^3+x^2+1$	x	2
12	**x^3+x^2**	$x^4+x^3+x^3+x^2$	x^4+x^2	x^3+x+1	11
13	**x^3+x^2+1**	$x^4+x^3+x^3+x^2+x+1$	x^4+x^2+x+1	x^3	8
14	**x^3+x^2+x**	$x^4+x^3+x^3+x^2+x^2+x$	x^4+x	x^3+x^2+1	13
15	**x^3+x^2+x+1**	$x^4+x^3+x^3+x^2+x^2+x+x+1$	x^4+1	x^3+x^2+x	14
	×	4	4	4	4
		x^2	**x^2**	**x^2**	**x^2**
4	**x^2**	x^4	x^4	x^3+x^2+x+1	15
5	**x^2+1**	x^4+x^2	x^4+x^2	x^3+x+1	11
6	**x^2+x**	x^4+x^3	x^4+x^3	x^2+x+1	7
7	**x^2+x+1**	$x^4+x^3+x^2$	$x^4+x^3+x^2$	$x+1$	3
8	**x^3**	x^5	x^5	1	1
9	**x^3+1**	x^5+x^2	x^5+x^2	$1+x^2$	6
10	**x^3+x**	x^5+x^3	x^5+x^3	$1+x^3$	9
11	**x^3+x+1**	$x^5+x^3+x^2$	$x^5+x^3+x^2$	$1+x^3+x^2$	13
12	**x^3+x^2**	x^5+x^4	x^5+x^4	x^3+x^2+x	14
13	**x^3+x^2+1**	$x^5+x^4+x^2$	$x^5+x^4+x^2$	x^3+x	10
14	**x^3+x^2+x**	$x^5+x^4+x^3$	$x^5+x^4+x^3$	x^2+x	6
15	**x^3+x^2+x+1**	$x^5+x^4+x^3+x^2$	$x^5+x^4+x^3+x^2$	x	2
	×	5	5	5	5
		x^2+1	**x^2+1**	**x^2+1**	**x^2+1**
5	**x^2+1**	$x^4+x^2+x^2+1$	x^4+1	x^3+x^2+x	14
6	**x^2+x**	$x^4+x^3+x^2+x$	$x^4+x^3+x^2+x$	1	1
7	**x^2+x+1**	$x^4+x^3+x^2+x^2+x+1$	x^4+x^3+x+1	x^2	4
8	**x^3**	x^5+x^3	x^5+x^3	$1+x^3$	9
9	**x^3+1**	$x^5+x^2+x^3+1$	$x^5+x^2+x^3+1$	x^2+x^3	12
10	**x^3+x**	$x^5+x^3+x^3+x$	x^5+x	$1+x$	3
11	**x^3+x+1**	$x^5+x^3+x^2+x^3+x+1$	$x^5+x^4+x^2+x+1$	$1+ x^3$	9
12	**x^3+x^2**	$x^5+x^4+x^3+x^2$	$x^5+x^3+x^2$	$1+x^3+x^2$	13
13	**x^3+x^2+1**	$x^5+x^4+x^2+x^3+x^2+1$	$x^5+x^4+x^3+1$	x^2+x+1	7
14	**x^3+x^2+x**	$x^5+x^4+x^3+x^3+x^2+x$	$x^5+x^4+x^2+x$	x^3	8
15	**x^3+x^2+x+1**	$x^5+x^4+x^3+x^2+x^3+x^2+x+1$	x^5+x^4+x+1	x^3+x^2+1	13
	×	6	6	6	6
		x^2+x	**x^2+x**	**x^2+x**	**x^2+x**
6	**x^2+x**	$x^4+x^3+x^3+x^2$	x^4+x^2	x^3+x+1	11
7	**x^2+x+1**	$x^4+x^3+x^2+x^3+x^2+x$	x^4+x	x^3+x^2+1	13
8	**x^3**	x^5+x^4	x^5+x^4	x^3+x^2+x	14
9	**x^3+1**	$x^5+x^2+x^4+x$	$x^5+x^2+x^4+x$	x^3	8
10	**x^3+x**	$x^5+x^3+x^4+x^2$	$x^5+x^3+x^4+x^2$	x	2
11	**x^3+x+1**	$x^5+x^3+x^2+x^4+x^2+x$	$x^5+x^3+x^4+x$	x^2	4
12	**x^3+x^2**	$x^5+x^4+x^4+x^3$	x^5+x^3	x^3+1	9
13	**x^3+x^2+1**	$x^5+x^4+x^2+x^4+x^3+x$	$x^5+x^2+x^3+x$	x^2+x^3+x+1	15
14	**x^3+x^2+x**	$x^5+x^4+x^3+x^4+x^3+x^2$	x^5+x^2	$1+x^2$	5
15	**x^3+x^2+x+1**	$x^5+x^4+x^3+x^2+x^4+x^3+x^2+x$	x^5+x	$1+x$	3
	×	7	7	7	7
		x^2+x+1	**x^2+x+1**	**x^2+x+1**	**x^2+x+1**
7	**x^2+x+1**	$x^4+x^3+x^2+x^2+x^3+x+x^2+x+1$	$x^4+x^3+x^2+1$	x	2
8	**x^3**	$x^5+x^4+x^3$	$x^5+x^4+x^3$	x^2+x	6
9	**x^3+1**	$x^5+x^2+x^4+x+x^3+1$	$x^5+x^2+x^4+x+x^3+1$	1	1
10	**x^3+x**	$x^5+x^3+x^4+x^2+x^3+x$	$x^5+x^4+x^2+x$	x^3	8
11	**x^3+x+1**	$x^5+x^3+x^2+x^4+x^2+x+x^3+x+1$	x^5+x^4+1	x^3+x^2+x+1	15
12	**x^3+x^2**	$x^5+x^4+x^4+x^3+x^3+x^2$	x^5+x^2	x^2+1	5
13	**x^3+x^2+1**	$x^5+x^4+x^2+x^4+x^3+x+x^3+x^2+1$	x^5++x+1	x	2
14	**x^3+x^2+x**	$x^5+x^4+x^3+x^4+x^3+x^2+x^3+x^2+x$	x^5+x^3+x	x^3+x+1	11
15	**x^3+x^2+x+1**	$x^5+x^4+x^3+x^2+x^4+x^3+x^2+x+x^3+x^2+x+1$	$x^5+x^3+x^2+1$	x^3+x^2	12
	×	8	8	8	8
		x^3	**x^3**	**x^3**	**x^3**
8	**x^3**	x^6	x^6	x	2
9	**x^3+1**	x^6+x^3	x^6+x^3	$x+x^3$	10
10	**x^3+x**	x^6+x^4	x^6+x^4	x^3+x^2+1	13
11	**x^3+x+1**	$x^6+x^4+x^3$	$x^6+x^4+x^3$	x^2+1	5
12	**x^3+x^2**	x^6+x^5	x^6+x^5	$x+1$	3
13	**x^3+x^2+1**	$x^6+x^5+x^3$	$x^6+x^5+x^3$	x^3+x+1	11
14	**x^3+x^2+x**	$x^6+x^5+x^4$	$x^6+x^5+x^4$	x^3+x^2	12
15	**x^3+x^2+x+1**	$x^6+x^5+x^4+x^3$	$x^6+x^5+x^4+x^3$	x^2	4
	×	9	9	9	9
		x^3+1	**x^3+1**	**x^3+1**	**x^3+1**
9	**x^3+1**	$x^6+x^3+x^3+1$	x^6+1	$x+1$	3
10	**x^3+x**	$x^6+x^4+x^3+x$	$x^6+x^4+x^3+x$	x^2+x+1	7
11	**x^3+x+1**	$x^6+x^4+x^3+x^3+x+1$	x^6+x^4+x+1	x^3+x^2+x	14
12	**x^3+x^2**	$x^6+x^5+x^3+x^2$	$x^6+x^5+x^3+x^2$	x^3+x^2+x+1	15

13	x^3+x^2+1	$x^6+x^5+x^3+x^3+x^2+1$	$x^6+x^5+x^2+1$	$x+x^2$	6
14	x^3+x^2+x	$x^6+x^5+x^4+x^3+x^2+x$	$x^6+x^5+x^4+x^3+x^2+x$	x	2
15	x^3+x^2+x+1	$x^6+x^5+x^4+x^3+x^3+x^2+x+1$	$x^6+x^5+x^4+x^2+x+1$	x^3+x+1	11
	×	10	10	10	10
		x^3+x	x^3+x	x^3+x	x^3+x
10	x^3+x	$x^6+x^4+x^4+x^2$	x^6+x^2	$x+x^2$	6
11	x^3+x+1	$x^6+x^4+x^3+x^4+x^2+x$	$x^6+x^3+x^2+x$	x^3+x^2	12
12	x^3+x^2	$x^6+x^5+x^4+x^3$	$x^6+x^5+x^4+x^3$	x^2	4
13	x^3+x^2+1	$x^6+x^5+x^3+x^4+x^3+x$	$x^6+x^5+x^4+x$	x^3+x^2+x	14
14	x^3+x^2+x	$x^6+x^5+x^4+x^4+x^3+x^2$	$x^6+x^5+x^3+x^2$	x^3+x^2+x+1	15
15	x^3+x^2+x+1	$x^6+x^5+x^4+x^3+x^4+x^3+x^2+x$	$x^6+x^5+x^2+x$	x^2+1	5
	×	11	11	11	11
		x^3+x+1	x^3+x+1	x^3+x+1	x^3+x+1
11	x^3+x+1	$x^6+x^4+x^3+x^4+x^2+x+x^3+x+1$	x^6+x^2+1	x^2+x+1	7
12	x^3+x^2	$x^6+x^5+x^4+x^3+x^3+x^2$	$x^6+x^5+x^4+x^2$	x^3	8
13	x^3+x^2+1	$x^6+x^5+x^3+x^4+x^3+x+x^3+x^2+1$	$x^6+x^5+x^4+x^3+x+x^2+1$	$x+1$	3
14	x^3+x^2+x	$x^6+x^5+x^4+x^4+x^3+x^2+x^3+x^2+x$	x^6+x^5+x	1	1
15	x^3+x^2+x+1	$x^6+x^5+x^4+x^3+x^4+x^3+x^2+x+x^3+x^2+x+1$	$x^6+x^5+x^3+1$	$x+x^3$	10
	×	12	12	12	12
		x^3+x^2	x^3+x^2	x^3+x^2	x^3+x^2
12	x^3+x^2	$x^6+x^5+x^5+x^4$	x^6+x^4	x^3+x^2+1	13
13	x^3+x^2+1	$x^6+x^5+x^3+x^5+x^4+x^2$	$x^6+x^3+x^4+x^2$	1	1
14	x^3+x^2+x	$x^6+x^5+x^4+x^5+x^4+x^3$	x^6+x^3	$x+x^3$	10
15	x^3+x^2+x+1	$x^6+x^5+x^4+x^3+x^5+x^4+x^3+x^2$	x^6+x^2	$x+x^2$	6
	×	13	13	13	13
		x^3+x^2+1	x^3+x^2+1	x^3+x^2+1	x^3+x^2+1
13	x^3+x^2+1	$x^6+x^5+x^3+x^5+x^4+x^2+x^3+x^2+1$	x^6+x^4+1	x^3+x^2	12
14	x^3+x^2+x	$x^6+x^5+x^4+x^5+x^4+x^3+x^3+x^2+x$	x^6+x^2+x	x^2	4
15	x^3+x^2+x+1	$x^6+x^5+x^4+x^3+x^5+x^4+x^3+x^2+x^3+x^2+x+1$	x^6+x^3+x+1	x^3+1	9
	×	14	14	14	14
		x^3+x^2+x	x^3+x^2+x	x^3+x^2+x	x^3+x^2+x
14	x^3+x^2+x	$x^6+x^5+x^4+x^5+x^4+x^3+x^4+x^3+x^2$	$x^6+x^4+x^2$	x^3+1	9
15	x^3+x^2+x+1	$x^6+x^5+x^4+x^3+x^5+x^4+x^3+x^2+x^4+x^3+x^2+x$	$x^6+x^4+x^3+x$	x^2+x+1	7
	×	15	15	15	15
		x^3+x^2+x+1	x^3+x^2+x+1	x^3+x^2+x+1	x^3+x^2+x+1
15	x^3+x^2+x+1	$x^6+x^5+x^4+x^3+x^5+x^4+x^3+x^2+x^4+x^3+x^2+x+x^3+x^2+x+1$	$x^6+x^4+x^2+1$	x^3	8

Laipsnio kėlimą apskaičiuojami pakartotinai judėdami po aukščiau sudarytą daugybos operacijos lentelę: MATLAB pagalba, pvz.:

```
<...>
B(7,1)=T2_daugybos(7,1);
B(7,2)=T2_daugybos(7,7);
B(7,3)=T2_daugybos(7,B(7,2));
B(7,4)=T2_daugybos(7,B(7,3));
B(7,5)=T2_daugybos(7,B(7,4));
B(7,6)=T2_daugybos(7,B(7,5));
B(7,7)=T2_daugybos(7,B(7,6));
B(7,8)=T2_daugybos(7,B(7,7));
B(7,9)=T2_daugybos(7,B(7,8));
B(7,10)=T2_daugybos(7,B(7,9));
B(7,11)=T2_daugybos(7,B(7,10));
B(7,12)=T2_daugybos(7,B(7,11));
B(7,13)=T2_daugybos(7,B(7,12));
B(7,14)=T2_daugybos(7,B(7, 13));
B(7,15)=T2_daugybos(7,B(7,14));
<...>
```

PRIEDAS NR.3. PILNO PERRINKIMO ATAKA 3X3

```
darbo_tipas=input(' - Visų galimų matricų generavimas "1"\n ');
if(darbo_tipas==1)
%MATRICŲ GENERAVIMAS IR IŠSAUGOJIMAS Į FAILĄ
c1=input('Įveskite generuojamų elementų eilę (max dydį) \n'); %7
tic
clear KK1 Kp1 Kp2 Kp3 Kp4 Kp5 Kp6 Kp7 Kp8 Kp9 Kp10 Kp11 Kp12 Kp13 Kp14 Kp15 Kp16;
%bom=(7^9);
Kp1=zeros(3,3,2500000);
Kp2=zeros(3,3,2500000);
Kp3=zeros(3,3,2500000);
Kp4=zeros(3,3,2500000);
Kp5=zeros(3,3,2500000);
Kp6=zeros(3,3,2500000);
Kp7=zeros(3,3,2500000);
Kp8=zeros(3,3,2500000);
Kp9=zeros(3,3,2500000);
Kp10=zeros(3,3,2500000);
Kp11=zeros(3,3,2500000);
Kp12=zeros(3,3,2500000);
Kp13=zeros(3,3,2500000);
Kp14=zeros(3,3,2500000);
Kp15=zeros(3,3,2500000);
Kp16=zeros(3,3,2500000);
display('Vietos rezervavimas');
toc
tic
c2=1; c3=1; p1=1; p2=1; p3=1; p4=1; p5=1; p6=1; p7=1; p8=1; p9=1; p10=1; p11=1; p12=1;
p13=1; p14=1; p15=1; p16=1;
hhh = waitbar(0,'Generuojamos visos galimos matricos...');
for i11 = 1:c1
for i12 = 1:c1
for i13 = 1:c1
for i21 = 1:c1
for i22 = 1:c1
for i23 = 1:c1
for i31 = 1:c1
for i32 = 1:c1
for i33 = 1:c1
    KK1=[i11 i12 i13; i21 i22 i23; i31 i32 i33];
    c3=c3+1;
        if det(KK1)~=0;
        if (c2<2500001); Kp1(:,:,p1)=KK1; p1=p1+1;
        elseif (c2>2500000) && (c2<5000001);  Kp2(:,:,p2)=KK1;  p2=p2+1;
        elseif (c2>5000000) && (c2<7500001);  Kp3(:,:,p3)=KK1;  p3=p3+1;
        elseif (c2>7500000) && (c2<10000001);  Kp4(:,:,p4)=KK1;  p4=p4+1;
        elseif (c2>10000000) && (c2<12500001);  Kp5(:,:,p5)=KK1;  p5=p5+1;
        elseif (c2>12500000) && (c2<15000001);  Kp6(:,:,p6)=KK1;  p6=p6+1;
        elseif (c2>15000000) && (c2<17500001);  Kp7(:,:,p7)=KK1;  p7=p7+1;
        elseif (c2>17500000) && (c2<20000001);  Kp8(:,:,p8)=KK1;  p8=p8+1;
        elseif (c2>20000000) && (c2<22500001);  Kp9(:,:,p9)=KK1;  p9=p9+1;
        elseif (c2>22500000) && (c2<25000001);  Kp10(:,:,p10)=KK1;  p10=p10+1;
        elseif (c2>25000000) && (c2<27500001);  Kp11(:,:,p11)=KK1;  p11=p11+1;
        elseif (c2>27500000) && (c2<30000001);  Kp12(:,:,p12)=KK1;  p12=p12+1;
        elseif (c2>30500000) && (c2<32500001);  Kp13(:,:,p13)=KK1;  p13=p13+1;
        elseif (c2>32500000) && (c2<35000001);  Kp14(:,:,p14)=KK1;  p14=p14+1;
        elseif (c2>35000000) && (c2<37500001);  Kp15(:,:,p15)=KK1;  p15=p15+1;
        elseif (c2>37500000) && (c2<40000001);  Kp16(:,:,p16)=KK1;  p16=p16+1;
        end
        c2=c2+1;
        end
        waitbar(c2/c1^9)
end
end
end
end
end
end
end
```

```
end
end
delete(hhh);
display('Iš viso ištirta matricų:'); c3-1
display('Atrinktos matricos, kurių DET ~= 0:'); c2-1
display('Generavimas užtruko:'); toc
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K1.mat', 'Kp1');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K2.mat', 'Kp2');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K3.mat', 'Kp3');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K4.mat', 'Kp4');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K5.mat', 'Kp5');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K6.mat', 'Kp6');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K7.mat', 'Kp7');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K8.mat', 'Kp8');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K9.mat', 'Kp9');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K10.mat', 'Kp10');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K11.mat', 'Kp11');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K12.mat', 'Kp12');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K13.mat', 'Kp13');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K14.mat', 'Kp14');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K15.mat', 'Kp15');
save('C:\Users\202020140114b\Documents\MATLAB\KKK\K16.mat', 'Kp16');

beep;
end

if(darbo_tipas==2)

pradine_matrica=[1 2 3; 6 5 4; 3 4 1];
L_matrica=[1 3 2; 6 1 4; 1 1 6];
R_matrica=[2 5 3; 4 3 4; 2 3 3];
sifruota_matrica=[6 4 1; 4 6 2; 2 5 3];
T_laipsnio=load('LaipsnioT.txt');
T_daugybos=load('DaugybosT.txt');
a=0; b=1;
load('C:\Users\202020140114b\Documents\MATLAB\KKK\K1.mat');
MasyvoDydis=size(Kp1,3);
hhh2 = waitbar(0,'Vykdomas perrinkimas');
tic

for i = 1:MasyvoDydis
    LL1=Kp1(:,:,i);
darbine_matrica(1,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(1,1)),T_laipsnio(pradine_matrica(
2,1),LL1(1,2))),T_laipsnio(pradine_matrica(3,1),LL1(1,3)));
darbine_matrica(1,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(1,1)),T_laipsnio(pradine_matrica(
2,2),LL1(1,2))),T_laipsnio(pradine_matrica(3,2),LL1(1,3)));
darbine_matrica(1,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,3),LL1(1,1)),T_laipsnio(pradine_matrica(
2,3),LL1(1,2))),T_laipsnio(pradine_matrica(3,3),LL1(1,3)));

darbine_matrica(2,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(2,1)),T_laipsnio(pradine_matrica(
2,1),LL1(2,2))),T_laipsnio(pradine_matrica(3,1),LL1(2,3)));
darbine_matrica(2,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(2,1)),T_laipsnio(pradine_matrica(
2,2),LL1(2,2))),T_laipsnio(pradine_matrica(3,2),LL1(2,3)));
darbine_matrica(2,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,3),LL1(2,1)),T_laipsnio(pradine_matrica(
2,3),LL1(2,2))),T_laipsnio(pradine_matrica(3,3),LL1(2,3)));

darbine_matrica(3,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),LL1(3,1)),T_laipsnio(pradine_matrica(
2,1),LL1(3,2))),T_laipsnio(pradine_matrica(3,1),LL1(3,3)));
darbine_matrica(3,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,2),LL1(3,1)),T_laipsnio(pradine_matrica(
2,2),LL1(3,2))),T_laipsnio(pradine_matrica(3,2),LL1(3,3)));
darbine_matrica(3,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,3),LL1(3,1)),T_laipsnio(pradine_matrica(
2,3),LL1(3,2))),T_laipsnio(pradine_matrica(3,3),LL1(3,3)));
```

```
pradine_matrica=darbine_matrica;
darbine_matrica=[];

for j = 1:MasyvoDydis
            RR1=Kp1(:,:,j);
a=a+1;
darbine_matrica(1,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),RR1(1,1)),T_laipsnio(pradine_matrica(
1,2),RR1(2,1))),T_laipsnio(pradine_matrica(1,3),RR1(3,1)));
darbine_matrica(1,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),RR1(1,2)),T_laipsnio(pradine_matrica(
1,2),RR1(2,2))),T_laipsnio(pradine_matrica(1,3),RR1(3,2)));
darbine_matrica(1,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(1,1),RR1(1,3)),T_laipsnio(pradine_matrica(
1,2),RR1(2,3))),T_laipsnio(pradine_matrica(1,3),RR1(3,3)));

darbine_matrica(2,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(2,1),RR1(1,1)),T_laipsnio(pradine_matrica(
2,2),RR1(2,1))),T_laipsnio(pradine_matrica(2,3),RR1(3,1)));
darbine_matrica(2,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(2,1),RR1(1,2)),T_laipsnio(pradine_matrica(
2,2),RR1(2,2))),T_laipsnio(pradine_matrica(2,3),RR1(3,2)));
darbine_matrica(2,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(2,1),RR1(1,3)),T_laipsnio(pradine_matrica(
2,2),RR1(2,3))),T_laipsnio(pradine_matrica(2,3),RR1(3,3)));

darbine_matrica(3,1) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(3,1),RR1(1,1)),T_laipsnio(pradine_matrica(
3,2),RR1(2,1))),T_laipsnio(pradine_matrica(3,3),RR1(3,1)));
darbine_matrica(3,2) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(3,1),RR1(1,2)),T_laipsnio(pradine_matrica(
3,2),RR1(2,2))),T_laipsnio(pradine_matrica(3,3),RR1(3,2)));
darbine_matrica(3,3) =
T_daugybos(T_daugybos(T_laipsnio(pradine_matrica(3,1),RR1(1,3)),T_laipsnio(pradine_matrica(
3,2),RR1(2,3))),T_laipsnio(pradine_matrica(3,3),RR1(3,3)));

                if isequal(sifruota_matrica, darbine_matrica)==true;
                    KrL(:,:,b)=LL1
                    KrR(:,:,b)=RR1
                    b=b+1;
                end
    waitbar(a/MasyvoDydis^2)
    darbine_matrica=[];
    end
    pradine_matrica2=[];
    save('C:\Users\202020140114b\Documents\MATLAB\KKK\KrL.mat', 'KrL');
    save('C:\Users\202020140114b\Documents\MATLAB\KKK\KrR.mat', 'KrR');

end
display('Iš viso patikrinta variantų: '); disp(a)
display('Tinkantys variantai: '); disp(b-1)
display('Procentaliai: '); disp((b-1)/a*100)
display('Skaičiavimai užtruko: '); disp(toc)
beep;
zzz=0;
delete(hhh2);
end
```