VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
KOMPIUTERIJOS KATEDRA

Baigiamasis magistro darbas

Efektyvios šifravimo bei skaitmeninio parašo sistemos

Atliko: 2 kurso, 1 grupės studentas
Mindaugas Valkaitis (parašas)

Darbo vadovas:
doc. dr. Vilius Stakėnas (parašas)

Vilnius
2012

Turinys

Anotacija

Darbe atlikta naujos kartos dedikuotos pasirašymo *ir* šifravimo kriptosistemos, pavadintos Signcryption, teorinė apžvalga bei teorinis ir praktinis šios kriptosistemos efektyvumo palyginimas su populiarių viešojo rakto pasirašymo *arba* šifravimo kriptosistemų kompozicija. Palyginimui parinktos RSA (Rivest, Shamir, Adleman) kriptosistema, kurios saugumas paremtas didelių skaičių faktorizacijos uždavinio sprendimo sudėtingumu, ir ElGamalio kriptosistema, kurios saugumas paremtas diskretaus logaritmo problemos sprendimo sudėtingumu.

Kriptosistemų efektyvumas apibrėžiamas dviem parametrais:

1. Pranešimo pasirašymo, šifravimo, dešifravimo ir parašo patikrinimo operacijų trukmė,
2. Perduodamos perteklinės informacijos kiekis – pranešimo ilgio padidėjimas atlikus pasirašymo ir šifravimo operacijas

Taip pat pasiūlyti du efektyvūs Signcryption praktinio pritaikymo algoritmai: grupei skirto šifro be slenksčio ir su slenksčiu schemos.

Summary

This submission called “Efficient encryption and digital signature schemes” consists of three parts.

- In Part I theoretical analysis of popular public key cryptosystems RSA (Rivest, Shamir, Adleman) with security based on the large integer factorization problem and ElGamal with security based on the discrete logarithm problem, along with new cryptographic primitive termed as "signcryption" proposed by Y. Zheng which simultaneously fulfills both the functions of digital signature and public key encryption in a logically single step, and with a cost significantly smaller than that required by "signature followed by encryption" using popular public key cryptosystem composition is done. For the completeness of analysis description of supplemental algorithms and functions such as AES block cipher, SHA hash functions, HMAC keyed hash function is present.
- In Part II the results of the practical implementation done in Python programming language are analyzed. Effectiveness is described by two factors:
  - Total computation time of signing – encryption – decryption – verification operations;
  - Communication overhead – signed and encrypted message length increase compared to the original plaintext.
- In Part III two effective Signcryption implementation algorithms are proposed: secret sharing without threshold and (k, n) threshold schemes.

Results of analysis prove Signcryption being secure and extremely effective signature and encryption cryptosystem. It has very low requirements for the computational power as well as provides almost no data expansion. These properties make it perfect solution for battery-powered small devices such as smart cards, mobile phones and personal digital assistants (PDAs) and applications operating in resource-constrained environment such as contactless wireless identification tokens.

Įvadas

Bendravimas vis labiau persikelia į virtualią ir tuo pačiu metu nesaugią viešą erdvę. Vis dažniau susiduriama su užduotimi kaip efektyviai perkelti į elektroninę erdvę bendravimą, kuris pasižymėtų dviem svarbiom „popierinio pašto“ savybėm:

1. pranešimai turi būti *asmeniniai* – naudojant viešus nesaugius komunikacijos kanalus šią savybę užtikrina šifravimas,
2. turi būti galimybė laiškus *pasirašyti* – siunčiant pranešimus viešais nesaugiais komunikacijos kanalais jie turi būti pasirašyti skaitmeniniu parašu.

Šiuo metu šios užduotys patikėtos viešojo rakto kriptosistemoms, tačiau populiarios viešojo rakto kriptosistemos buvo kurtos ne abiems operacijoms kartu, o kiekvienai atskirai, dėl to, norint užšifruoti ir pasirašyti, naudojama elementari kompozicija: pirmą kartą algoritmas pritaikomas pasirašymui, antrą – šifravimui. Du kartus naudoti tą patį algoritmą nėra efektyvus užduoties sprendimas. Padėtį kiek pataiso maišos funkcijos – pirmą kartą (pasirašant) nebereikia algoritmo taikyti visam pranešimui, pakankama „pasirašyti“ santrauką, tačiau turi būti efektyvesnių užduoties sprendimų nei kompozicija. Vieną tokių būdų 1997 metais pasiūlė Y. Zhengas, kuris pateikė naujos kartos vientisą viešojo rakto šifruoto *ir* pasirašyto pranešimo sukūrimo kriptosistemą, kurią pavadino Signcryption [Zhe97].

Šio darbo tikslas – apžvelgti šiuo metu naudojamas klasikines viešojo rakto šifravimo ir skaitmeninio parašo sistemas bei naujos kartos Signcryption kriptosistemą ir atlikti dedikuotos pasirašymo *ir* šifravimo kriptosistemos efektyvumo palyginimą su pasirašymo *arba* šifravimo kriptosistemų kompozicija bei pasiūlyti praktinio pritaikymą naujos kartos Signcryption kriptosistemai.

Darbe apžvelgtos šios kriptosistemos:

1. RSA (Rivest, Shamir, Adleman) – klasikinė viešojo rakto pasirašymo *arba* šifravimo kriptosistema, kurios saugumas paremtas didelių skaičių faktorizacijos uždavinio sprendimo sudėtingumu,
2. ElGamalio – klasikinė viešojo rakto pasirašymo *arba* šifravimo kriptosistema, kurios saugumas paremtas diskretaus logaritmo problemos sprendimo sudėtingumu,
3. Signcryption – naujos kartos viešojo rakto pasirašymo *ir* šifravimo kriptosistema, realizuota modifikuotos ElGamalio skaitmeninio parašo schemos pagrindu.

Minėtos kriptosistemos apžvelgtos teoriškai, sukurta praktinė jų realizacija ir apžvelgti rezultatai bei palygintas jų efektyvumas, kuris apibrėžiamas dviem parametrais:

1. Pranešimo pasirašymo, šifravimo, dešifravimo ir parašo patikrinimo operacijų trukmė,
2. Perduodamos perteklinės informacijos kiekis – pranešimo ilgio padidėjimas atlikus pasirašymo ir šifravimo operacijas.

Taip pat apžvelgtos kriptosistemų realizacijoje naudotos papildomos funkcijos bei algoritmai, tokie kaip AES blokiniai šifrai, SHA maišos funkcijų šeima, HMAC kontrolinis parašas bei pasiūlyti du efektyvūs Signcryption praktinio pritaikymo algoritmai: grupei skirto šifro be slenksčio ir su slenksčiu schemos.

1 Blokiniai šifrai

Blokiniais šifrais vadinamos kriptosistemos, kurių šifravimo algoritmai transformuoja fiksuoto ilgio teksto žodžius (blokus) į tokio pat ilgio šifro žodžius. Raktas, valdantis šifravimo operaciją, irgi paprastai renkamas iš fiksuoto ilgio žodžių aibės [Sta07].

Blokiniai šifrai priklauso simetrinio rakto kriptosistemoms, tai yra tas pats raktas naudojamas tiek šifravimui, tiek dešifravimui, taigi tenkinama lygybė:

$$M=dec(enc(M, K), K).$$

Kur:

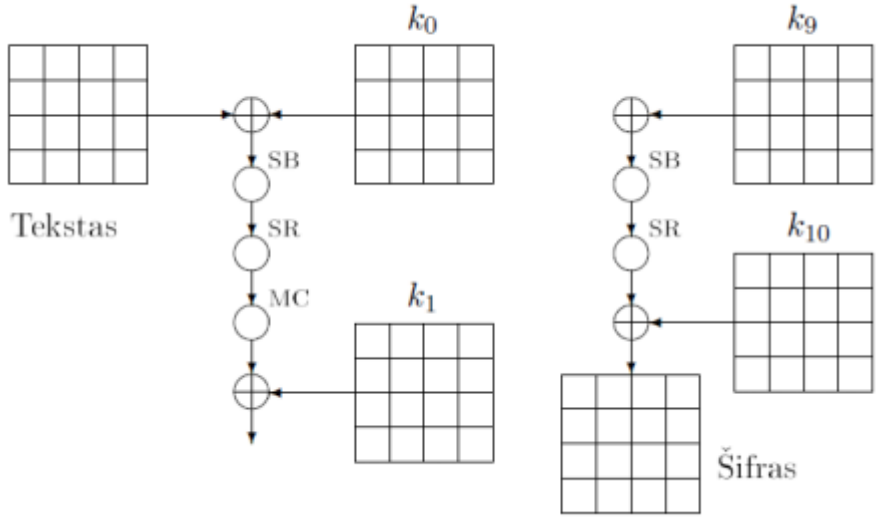
M – norima perduoti informacija,
enc – tam tikra šifravimo funkcija,
dec – tam tikra dešifravimo funkcija,
K – šifravimo/ dešifravimo raktas.

1.1 AES

Pažangus šifravimo standartas AES (Advanced Encryption Standard, angl.) yra elektroninių duomenų šifravimo specifikacija 2001 metais patvirtinta Nacionalinio Standartų ir Technologijų instituto (National Institute of Standards and Technology (NIST), angl.) kaip JAV Federalinis informacijos apdorojimo standartas (Federal Information Processing Standard (FIPS)). AES pakeitė anksčiau naudotą DES standartą.

AES standartas apibrėžia Rijndael (pavadintą „žaidžiant“ autorių, – belgų kriptografų Vincent Rijmen ir Joan Daemen, – pavardžių raidėmis) algoritmą – simetrinio blokinį šifrą, kuris šifruoja 128 bitų dydžio blokus, naudojant 128, 192 ir 256 bitų dydžio raktus [Fips197].

AES-128 (skaičius po pavadinimo nurodo naudojamo rakto dydį bitais) sudaro 10 vienodos struktūros žingsnių. Kiekvienai operacijai iš kriptosistemos bendro rakto sudaromas dalinis raktas. Šifravimas prasideda sudėties su pradžios raktu veiksmu. Pirmieji devyni žingsniai vienodi - atliekamos baitų keitimo (SB), eilučių postūmio (SR) ir stulpelių maišymo (MC) operacijos. Paskutiniame žingsnyje stulpelių maišymo nėra [Sta07]. Bendra algoritmo AES-128 schema pateikta *1 pav*.



1 paveiksliukas. Algoritmo AES-128 schema

2 Viešojo rakto kriptosistemos

Viešojo rakto kriptosistemos taip buvo pavadintos, nes jos paremtos dviejų raktų, kurių vienas yra slaptas (privatus), o kitas *viešas* panaudojimu. Be to šis pavadinimas leidžia jas aiškiai atskirti nuo tradicinių kriptosistemų, žinomų simetrinio rakto, bendros paslapties, slapto rakto, privataus rakto pavadinimais.

XX a. simetrinio šifravimo algoritmai buvo labai ištobulinti ir šiuo metu egzistuoja visa eilė efektyvių, didelį saugumo lygį užtikrinančių simetrinio rakto kriptosistemų, tačiau sparčiai vystantis telekomunikacijoms ir ypatingai internetui iškilo vadinama „raktų paskirstymo problema“ (key distribution problem, angl.). Jos esmė, kad prieš vykdant privatų duomenų apsikeitimą naudojantis privataus rakto kriptosistema viešais komunikacijos kanalais, turi būti įvykdytas *papildomas* privatus duomenų apsikeitimas – slaptų raktų pateikimas pranešimų siuntėjui ir gavėjui. Dažniausiai tam naudojamas kurjerių paštas. Tačiau šis sprendimas nepriimtinas elektroninio pašto sistemose, kurių pagrindiniai privalumai yra greitis ir maži kaštai.

1976 metais W. Diffie ir M. Hellman pasiūlė būdą, pavadintą ***viešojo rakto kriptosistema***, kuris leistų informacijos siuntėjui ir gavėjui apsikeisti šifravimo raktais naudojant viešus (t.y. nesaugius) komunikacijos kanalus, nekeliant pavojaus kriptosistemos saugumui [DH76]. Tam reikia, kad šifravimas ir dešifravimas būtų atliekami skirtingais raktais $\boldsymbol{K_{enc}}$ ir $\boldsymbol{K_{dec}}$, tokias, kad turint $\boldsymbol{K_{enc}}$ išskaičiuoti $\boldsymbol{K_{dec}}$ skaičiavimų prasme būtų neįmanoma (pvz. reikalautų 10^{100} operacijų). Tada šifravimo raktas $\boldsymbol{K_{enc}}$ galėtų būti skelbiamas viešai, neatskleidžiant dešifravimo rakto $\boldsymbol{K_{dec}}$. Tokiu būdu dviejų asmenų komunikacija galėtų būti vykdoma slaptai, kai siunčiama informacija šifruojama gavėjo ***viešuoju raktu*** $\boldsymbol{K_v = K_{enc}}$, o ją dešifruoti gali tik ***privataus rakto*** $\boldsymbol{K_p = K_{dec}}$ turėtojas. Taigi viešojo rakto kriptosistema privalo tenkinti lygybę:

$$M=dec(enc(M, K_v), K_p), K_v \neq K_p.$$

Kur:

M – norima perduoti informacija,
enc – tam tikra šifravimo funkcija,
dec – tam tikra dešifravimo funkcija,
K_v – šifravimui naudojamas viešasis raktas,
K_p – dešifravimui naudojamas privatus raktas.

2.1 RSA kriptosistema

1978 R. Rivest, A. Shamir, L. Adleman pasiūlė kriptosistemą, kurios saugumas pagrįstas faktorizacijos (skaičiaus skaidymo pirminiais daugikliais) uždavinio sudėtingumu [RSA78].

Autoriai pasiūlė tokią kriptosistemos realizaciją: norint užšifruoti pranešimą M, naudojant viešą raktą (e, n), kur e ir n laisvai parinkti sveiki teigiami skaičiai, reikia pranešimą paversti sveiku teigiamu skaičiumi tarp 0 ir n-1 (ilgesnius pranešimus reikia išskaidyti į blokų seką, kiekvieną sekos bloką paverčiant natūraliuoju skaičiumi, tenkinančiu paminėtą sąlygą).

Šifruotas pranešimas C gaunamas keliant pranešimą M e – uoju laipsniu moduliu n:

$$C \equiv \mathrm{E}(M) \equiv M^e \pmod{n}.$$

Dešifravimas atliekamas šifruotą pranešimą C keliant laipsniu d, taip pat moduliu n:

$$\mathrm{D}(C) \equiv C^d \pmod{n}.$$

Kaip matome šifravimo raktas yra natūraliųjų skaičių pora (e, n), o dešifravimo raktas yra natūraliųjų skaičių pora (d, n). Dešifravimo raktas pateikiamas viešai, šifravimo – saugomas paslaptyje.

Norint teisingai sudaryti raktus reikia:

1. Parinkti du didelius pirminius skaičius p ir q;
2. n gaunamas sudauginus p ir q, t.y.:
$$n = p \cdot q.$$
3. laisvai parinkti natūralųjį skaičių d, kuris būtų tarpusavyje pirminis su $(p\text{-}1) \cdot (q\text{-}1)$, t.y. tenkintų sąlygą $\gcd(d, (p\text{-}1) \cdot (q\text{-}1)) = 1$, čia gcd – didžiausias bendras daliklis (greatest common divisor, angl.);
4. naudojantis Euklido algoritmu rasti skaičių e, kad:
$$e \cdot d \equiv 1 \pmod{((p\text{-}1) \cdot (q\text{-}1))}.$$

Apskaičiavus n, e ir d, pirminius skaičius p ir q tikslinga sunaikinti, siekiant maksimizuoti kriptosistemos saugumą, nes jų daugiau nebeprireiks. Nors n pateikiamas viešai, p ir q radimas prilygsta faktorizacijos uždavinio sprendimui, būtent todėl šios kriptosistemos saugumas ir prilyginamas faktorizacijos uždavimo sprendimo sudėtingumui.

2.2 ElGamalio kriptosistema

1985 metais T. ElGamalis pasiūlė kriptosistemą, kurios saugumas pagrįstas tuo, kad dideliam pirminiam skaičiui p diskretaus logaritmo radimo uždavinį spręsti yra sunku. Be to iki šiol nėra įrodyta, kad kriptosistemos saugumas tapatus efektyvaus būdo spręsti diskretaus logaritmo uždavinį dideliems pirminiams skaičiams p radimui, t.y. net jei būtų rastas efektyvus metodas skaičiuoti diskrečius logaritmus nebūtinai reikštų kad šio tipo kriptosistemos tapo nebesaugios [Elg85].

T. ElGamalis savo kriptosistemoje panaudojo algoritmą, aprašytą Diffie – Hellman apsikeitimo raktais schemoje: tarkime A ir B nori apsikeisti paslaptimi K_{AB} bei tik A žino slaptą x_A, o tik B slaptą x_B. Tegul p yra didelis pirminis skaičius, o α primityvioji šaknis mod p, abu šie elementai viešai žinomi. Tada A apskaičiuoja $y_A \equiv \alpha^{x_A} \bmod p$ ir persiunčia jį B, analogiškai B apskaičiuoja $y_B \equiv \alpha^{x_B} \bmod p$ ir persiunčia jį A. Tada paslaptis K_{AB} apskaičiuojama taip:

$$K_{AB} \equiv \alpha^{x_A x_B} \pmod p \equiv y_A^{x_B} \pmod p \equiv y_B^{x_A} \pmod p.$$

Taigi, tarkime A nori perduoti B pranešimą m, kur $0 \le m \le p\text{-}1$ (jei pranešimas ilgesnis nei p, jis turi būti išskaidomas į blokus, kurie tenkina šią sąlygą). Visų pirma A pasirenka skaičių k, tokį, kad $0 \le k \le p\text{-}1$. Šis skaičius k bus naudojamas kaip A paslaptis x_A. Tada A apskaičiuoja „raktą“

$$K \equiv y_B^k \pmod p.$$

kur $y_B \equiv \alpha^{x_B} \bmod p$ viešai paskelbtas arba persiųstas B. Tada šifruotas pranešimas bus pora (c_1, c_2), kur

$$c_1 \equiv \alpha^k \pmod p, \qquad c_2 \equiv K \cdot m \pmod p.$$

Pažymėtina, kad šifruoto pranešimo ilgis dvigubai didesnis nei nešifruoto pranešimo.

Pranešimo dešifravimas vyksta dviem etapais. Visų pirma reikia nustatyti raktą K. B jį gali lengvai rasti, nes $K \equiv (\alpha^k)^{x_B} \bmod p \equiv c_1^{x_B} \bmod p$, o x_B žinomas tik B. Antras žingsnis – c_2 padalinus iš K gauti pranešimą m.

3 Skaitmeninis parašas

Atsiradusi galimybė telekomunikaciniais tinklais saugiai perduoti informaciją, sąlygojo naujo poreikio „pasirašyti“ dokumentus elektroniniu būdu atsiradimą. Skaitmeniniam parašui taip pat naudojama asimetrinių raktų pora $\boldsymbol{K_v}$ *ir* $\boldsymbol{K_p}$, o algoritmai turi savybę:

$$ver(M, y| K_v)=1,\ y= sig(M|K_p),\ K_v \neq K_p.$$

Kur:

M – tekstas, kuriam kuriamas parašas,
y – teksto M skaitmeninis parašas,
sig – tam tikra parašo kūrimo funkcija,
ver – tam tikra parašo tikrinimo funkcija,
K_p – parašo kūrimui naudojamas privatus raktas,
K_v – parašo tikrinimui naudojamas viešasis raktas.

Pažymėtina, kad nors informacijos kodavimui ir skaitmeniniam parašui naudojami asimetriniai kodavimo algoritmai su viešo ir privataus raktų pora, tai yra atskiros savarankiškos sistemos ir naudojamos skirtingos raktų poros.

Skaitmeninis parašas „atlieka“ tris funkcijas, aiškumo dėlei naudokime pavyzdžius:

- autentifikavimo (authentication, angl.) – kai skaitmeninis parašas yra susietas su virtualiu, – esančiu elektroninėje erdvėje, – asmeniu, pavyzdžiui Simu, teisingas (tenkinantis patikros lygtį) parašas užtikrina, kad pranešimo siuntėjas yra būtent susietasis vartotojas, t.y. virtualus asmuo Simas;
- vientisumo (integrity, angl.) – ši savybė užtikrina, kad siuntimo metu pranešimas nebuvo pakeistas: teisingas parašas užtikrina, kad jo turinys yra toks, kokį sukūrė siuntėjas. Taigi, jei pranešimas tenkina patikros lygtį naudojant Simo skaitmeninį parašą, galima teigti kad pranešimo turinys yra būtent toks, kokį sukūrė Simas;
- nepaneigiamumo (non-repudiation, angl.) – tai labai svarbi funkcija, susiejanti elektroninę erdvę su realiu pasauliu. Ši funkcija užtikrina, kad virtualus asmuo, su kuriuo yra susietas skaitmeninis parašas, ir asmuo realiame pasaulyje yra tapatūs, t.y. vartotojas, elektroninėje erdvėje prisistatantis Simu ir turintis susietą skaitmeninį parašą, realiame pasaulyje taip pat yra Simas, o ne Sandra.

Kaip matosi iš pateiktų funkcijų, skaitmeninio parašo sistemos nėra naudojamos pranešimo turiniui perduoti, todėl kyla klausimas ar tikslinga pasirašyti visą pranešimą, t.y. taikyti parašo kūrimo funkciją visam pranešimui, kuris gali būti labai ilgas? Pasirodo ne – pakanka panaudoti maišos funkciją ir pasirašyti pranešimo santrauką (plačiau žr. *4. Maišos funkcijos*).

3.1 RSA skaitmeninio parašo schema

Kadangi RSA kriptosistemoje šifruojama ir dešifruojama tuo pačiu algoritmu tik su skirtingais raktais, tai, norint RSA naudoti kaip skaitmeninio parašo schemą, nieko nereikia keisti, tačiau egzistuoja tam tikri praktinio realizavimo keblumai [Sta07].

Jei A viešasis RSA raktas yra $K_{V,A} = (n_A, e_A)$, o privatus $K_{P,A} = (n_A, d_A)$, tai pranešimo x parašą A gali sudaryti taip:

$$y = \text{sig}\,(x \mid K_{P,A}) \equiv x^{d_A} \pmod{n_A}.$$

parašo tikrinimas – pranešimo dešifravimas su A viešuoju raktu:

$$x \equiv y^{e_A} \pmod{n_A}.$$

Siekiant užtikrinti privatumą patartina, kad pranešimo gavėjas B taip pat naudotų RSA kriptosistemą, su savo raktais $K_{V,B} = (n_B, e_B)$, o privatus $K_{P,B} = (n_B, d_B)$. Šiuo atveju A galės užšifruoti pranešimą su B viešuoju raktu, taip užtikrinant privatumą.

Jei pranešimas pasirašomas siuntėjo A privačiu raktu ir šifruojamas gavėjo B viešuoju raktu gauname šifravimo ir skaitmeninio parašo sistemų kompoziciją, kuri gali būti dviejų tipų: „Šifravimas – po to – pasirašymas“ (encrytion – then – signature arba EtS, angl.) arba „Pasirašymas – po to – šifravimas“ (signature – then – encrytion arba StE, angl.), plačiau apie kompoziciją aprašyta *4 Šifravimas ir skaitmeninis parašas*, taigi A turi du pasirinkimus:

$$c_{StE}=\text{e}(\,\text{sig}(\,x \mid d_A) \mid e_B) \text{ ir } c_{EtS}=\text{sig}(\,\text{e}(\,x \mid e_B) \mid d_A).$$

Tačiau šie variantai nelygiaverčiai – tarkime $n_A > n_B$, tuomet norint užšifruoti parašą $y = \text{sig}\,(x \mid d_A)$, turi būti teisinga nelygybė $y < n_B$. Tačiau jei $n_A > n_B$, tai gali būti ir $y < n_B$. Tokiu atveju galime naudoti tik EtS schemą, kuri turi eilę trukumų, pavyzdžiui pasiduoda "žmogaus per vidurį" (man in the middle, angl.) tipo atakoms.

Šie keblumai lengvai išsprendžiami susitarus, kad kiekvienas RSA kriptosistemos naudotojas turės po dvi poras raktų: vieną skirtą pasirašymui, kitą – šifravimui. Be to sutariamas tam tikras slenkstis T, toks, kad pasirašymui skirtų raktų moduliai turi būti mažesni nei T, o šifravimui – didesni.

3.2 ElGamalio skaitmeninio parašo schema

1985 metais T. ElGamalis pasiūlė ne tik kriptosistemą, kurios saugumas pagrįstas diskrečiojo logaritmo radimo uždavinio sudėtingumu, bet ir skaitmeninio parašo schemą, kurios saugumas taip pat pagrįstas diskrečiojo logaritmo radimo uždavinio sudėtingumu. Pagal šią schemą skaitmeninio parašo tikrinimui naudojami tie patys viešieji raktai, kurie naudojami šifravimui – $y \equiv q^x \mod p$, kur p – didelis pirminis skaičius, $q < p$ – primityvioji multiplikatyviosios grupės $\mathbb{F}_p^*$ šaknis, y – viešasis raktas, x – privatus raktas [Elg85].

Pranešimo m ($0 \le m \le p$-1) pasirašymui vartotojas A pasinaudodamas privačiu raktu x turi galėti sukurti tokį parašą, kad visi vartotojai galėtų patikrinti jo autentiškumą naudodami viešąjį raktą y (kartu su viešais q ir p), tačiau nežinant x nebūtų įmanoma sukurti netikro parašo.

Pranešimo m parašas bus pora (r, s), kur $0 \le r, s \le p$-1, parinkta taip, kad tenkintų lygybę:

$$q^m \equiv y^r r^s \pmod{p}, \tag{1}$$

Pasirašymo procedūra atliekama trimis žingsniais:

1) Parenkamas atsitiktinis natūralusis skaičius k, toks, kad $0 \le k \le p$-1 ir $\gcd(k, p\text{-}1) = 1$;
2) Apskaičiuojamas r:

$$r \equiv q^k \pmod{p}, \tag{2}$$

3) Tada (1) galime perrašyti:

$$q^{m} \equiv q^{xr} \cdot q^{ks} \pmod{p}, \quad (3)$$

šią lygtį galime panaudoti *s* radimui žinodami, kad:

$$m \equiv x \cdot r + k \cdot s \pmod{(p-1)}, \quad (4)$$

arba

$$s \equiv (m - xr) \cdot k^{-1} \pmod{(p-1)}. \quad (5)$$

(4) lygtis turi sprendinį *s* tik tuomet, jei *k* parinktas taip, kad gcd(*k*, *p*-1) = 1.

Parašo tikrinimo procedūra labai paprasta: kadangi viešasis raktas yra (*y*, *q*, *p*), o pranešimo m parašas (*r*, *s*) tai į (1) lygtį įrašius *m*, *r*, *s* lygybė turi būti teisinga.

4 Maišos funkcijos

Maišos funkcijos (hash function, angl.) – tokios funkcijos, kurias pritaikius norimam duomenų blokui, – pavyzdžiui pranešimui, – grąžinama fiksuoto ilgio santrauka. Ideali maišos funkcijas turi pasižymėti šiomis savybėmis:

- vienpusiškumas (one-way property, angl.), ši savybė reiškia, kad:
  - bet kokiam pranešimui paskaičiuoti santrauką turi būti paprasta;
  - turint santrauką neįmanoma paskaičiuoti paties pranešimo;
- atsparumas kolizijoms[1] (collision resistance, angl.), ši savybė reiškia, kad:
  - neįmanoma pakeisti pranešimo taip, kad nepasikeistų santrauka;
  - neegzistuoja du skirtingi pranešimai su vienodomis santraukomis. Maišos funkcijos santrauka paprastai yra nuo 128 iki 512 bitų ilgio, t.y. santraukos priklauso baigtinei skaičių aibei, o pranešimų skaičius yra begalinis, todėl akivaizdu, kad kiekviena praktinė maišos funkcijos realizacija turi be galo daug kolizijų. Dėl šios priežasties praktikoje atsparumas kolizijoms reiškia, kad nors kolizijos tikrai egzistuoja, jų neįmanoma aptikti [FS05].

Kaip matosi iš pateiktų maišos funkcijos savybių, santraukos pasirašymas yra tapatus pranešimo pasirašymui, taigi taikyti parašo kūrimo funkciją visam pranešimui nėra tikslinga, pakanka pasirašyti pranešimo santrauką, Dėl šios priežasties skaitmeninio parašo sistemos yra neatsiejamos nuo kriptografinių maišos funkcijų.

4.1 SHA – 2 maišos funkcijų šeima

SHA-2 (secure hash algorithm, angl.) maišos funkcijų šeima, kurią sudaro SHA-224, SHA-256, SHA-384, SHA-512 funkcijos (skaičius prie pavadinimo nurodo kiek bitų ilgio santrauka sudaroma naudojant funkciją) buvo sudaryta Nacionalinės saugumo agentūros (National Security Agency (NSA), angl.) ir 2001 metais patvirtinta Nacionalinio Standartų ir Technologijų instituto (National Institute of Standards and Technology (NIST), angl.) kaip JAV Federalinis informacijos apdorojimo standartas (Federal Information Processing Standard (FIPS)) [Fips180]. SHA-2 šeimos maišos funkcijų savybių palyginimas pateiktas *1 lentelėje*.

SHA-224 ir SHA-256 naudoja šešias logines funkcijas, kiekviena iš funkcijų atlieka operacijas su 32 bitų žodžiais, žymimais *x*, *y* ir *z*. Kiekvienos funkcijos rezultatas – naujas 32 bitų žodis:

$\mathrm{Ch}(x, y, z) = (x \wedge y) \oplus (\neg x \wedge z)$,

[1] Kalbant apie maišos funkcijas kolizija vadinamas situacija, kai egzistuoja du skirtingi pranešimai m_1 ir m_2, tokie, kad $h(m_1) = h(m_2)$.

$\mathrm{Maj}(x, y, z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$,

$\sum_0^{\{256\}}(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$,

$\sum_1^{\{256\}}(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$,

$\sigma_0^{\{256\}} = ROTR^7(x) \oplus ROTR^{18}(x) \oplus ROTR^3(x)$,

$\sigma_1^{\{256\}} = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus ROTR^{10}(x)$.

Kur:

$\wedge$ – bitinė IR operacija,

$\vee$ – bitinė ARBA operacija,

$\oplus$ – bitinė XOR operacija,

$\neg$ – bitinė papildymo operacija,

<< – pastūmimo į kairę operacija, kur $x << n$ gaunamas atmetus n bitų iš žodžio x kairės bei užpildant n bitų iš dešinės nuliais,

>> – pastūmimo į dešinę operacija, kur $x >> n$ gaunamas atmetus n bitų iš žodžio x dešinės bei užpildant n bitų iš kairės nuliais,

$ROTR^n(x)$ – pasukimo į kairę operacija, kur x – w-bitų ilgio žodis, n natūralusis skaičius $0 \leq n \leq w$. $ROTR^n(x) = (x << n) \vee (x >> w - n)$.

SHA-384 ir SHA-512 funkcijų veikimas analogiškas, tik jos atlieka operacijas su 64 bitų žodžiais, o kiekvienos funkcijos rezultatas – naujas 64 bitų žodis.

Algoritmas	Pranešimo dydis (bitais)	Bloko dydis (bitais)	Žodžio dydis (bitais)	Pranešimo santraukos ilgis (bitais)
SHA-224	$<2^{64}$	512	32	224
SHA-256	$<2^{64}$	512	32	256
SHA-384	$<2^{128}$	1024	64	284
SHA-512	$<2^{128}$	1024	64	512

1 lentelė. SHA-2 šeimos maišos funkcijų savybių palyginimas

4.2 Kontrolinis parašas HMAC

Kriptografijoje HMAC (Hash-based Message Authentication Code, angl.) vadinama tam tikra schema, apskaičiuojanti pranešimo kontrolinį parašą naudojant maišos funkciją kartu su šifravimo raktu (Keyed Hash Function, angl.) [Rfc2104].

HMAC apibrėžimui reikia maišos funkcijos, kuri žymima H ir šifravimo rakto K. Pagrindinis reikalavimas maišos funkcijai – ji turi iteratyviai naudoti tam tikras funkcijas duomenų blokams, taigi aprašytoji SHA-2 šeimos funkcija puikiai tinka. Bloko dydis baitais žymimas B (SHA-224 ir SHA-256 atveju B=64, SHA-384 ir SHA-512 B=128), pranešimo santraukos dydis baitais žymimas L (28, 32, 48 ir 64 baitai atitinkamai SHA-224, SHA-256, SHA-384 ir SHA-512). Šifravimo raktas turi būti $L \leq K \leq B$, jei turimas raktas ilgesnis nei B, prieš jį naudojant reikia jį perskaičiuoti su maišos funkcija H – tokiu atveju raktas taps L dydžio.

Papildomai naudojami du žodžiai:

ipad = baitas 0x36 pakartotas B kartų,
opad = baitas 0x5C pakartotas B kartų.

Norint paskaičiuoti kontrolinio parašo HMAC reikšmę pranešimui m reikia atlikti šiuos veiksmus:

1. prie rakto *K* iš galo pridėti tiek nulių, kad jo ilgis būtų lygus *B* (t.y. jei rakto *K* ilgis yra 32 baitai, o *B*=64, *K* turi būti papildytas 32 nuliniais baitais 0x00),
2. atlikti XOR operaciją su raktu *K*, gautu atlikus (1) aprašytą operaciją, ir žodžiu *ipad*,
3. prie (2) žingsnyje gauto žodžio pridėti pranešimą *m*,
4. pritaikyti maišos funkciją *H* žodžiui, gautam (3) žingsnyje,
5. atlikti XOR operaciją su raktu *K*, gautu atlikus (1) aprašytą operaciją, ir žodžiu *opad*,
6. prie (5) žingsnyje gauto žodžio pridėti maišos funkcijos santrauką, gautą (4) žingsnyje,
7. pritaikyti maišos funkciją *H* žodžiui, gautam (6) žingsnyje.

Šią seką galima užrašyti kaip formulę:

$$H\,(K \text{ XOR } opad, H\,(K \text{ XOR } ipad, m)).$$

5 Šifravimas ir skaitmeninis parašas

Siekiant perkelti komunikaciją nuo įprasto pašto į elektroninę erdvę, reikia užtikrinti dviejų svarbių „popierinio pašto“ savybių perkėlimą: (a) pranešimai turi būti *asmeniniai* – naudojant viešus nesaugius komunikacijos kanalus šią savybę užtikrina šifravimas, ir (b) turi būti galimybė laiškus *pasirašyti* – siunčiant pranešimus viešais nesaugiais komunikacijos kanalais jie turi būti pasirašyti skaitmeniniu parašu.

Šiuo metu, siekiant užtikrinti šias dvi „popierinio pašto“ savybes, paprastai naudojama elementari pasirinktų viešojo rakto kriptosistemos ir skaitmeninio parašo schemos kompozicija, apie kurią jau užsiminta *3.1 RSA skaitmeninio parašo schema*. Kaip jau minėta, kompoziciją galima realizuoti dviem būdais:

1. „Šifravimas – po to – pasirašymas“ (encryption – then – signature arba *EtS*, angl.) – realizuojant šiuo būdu pranešimas pirma užšifruojamas, po to pasirašomas. Šio būdo pagrindinis privalumas – lengva pranešimų srauto kontrolė: jei patikrinus pranešimo autentiškumą jis nepasitvirtina, pranešimas tiesiog atmetamas, nevykdant jo dešifravimo. Šio būdo trūkumas – jis neatitinka kriptografijoje naudojamo Hortono principo, pagal kurį reikia autentifikuoti tai kas norėta pasakyti, o ne tai kas pasakyta, t.y. mes pasirašome ir tuo pačiu tvirtiname prasmingą atvirą tekstą, o ne jo užšifruotą atitikmenį – beprasmį simbolių rinkinį.
2. „Pasirašymas – po to – šifravimas“ (signature – then – encryption arba *StE*, angl.) – realizuojant šiuo būdu pranešimas pirma pasirašomas, po to užšifruojamas. Šio būdo privalumas – atitikimas aukščiau paminėtam Hortono principui. Trūkumas – prieš įsitikindami pranešimo autentiškumu mes turime jį dešifruoti.

Praktiniai šių kompozicijų pritaikymo niuansai paminėti *3.1 RSA skaitmeninio parašo schema*, be to egzistuoja dar visa eilė šių dviejų kompozicijos būdų palyginimų saugumo aspektu, kurie nusveria pasirinkimą antrojo būdo naudai, be to šiuolaikiniai šifravimo algoritmai yra pakankamai ištobulinti, o kompiuterinė techninė įranga pakankamai naši, kad galima būtų neatsižvelgti į šiek tiek didesnę apkrovą, tenkančią procesoriui, todėl praktikoje dažniausiai sutinkamos antruoju būdu sukurtos realizacijos [FS05].

1997 metais Y. Zhengas pateikė šifruoto ir pasirašyto pranešimo sukūrimo kriptosistemą, kuri ženkliai sumažina tokio tipo pranešimo kūrimo kaštus, lyginant su tradicinėmis StE schemomis. Pasiūlytoji kriptosistema atlieka ne vieną iš, o abi operacijas kartu, todėl galima teigti, kad pasiūlyta kriptosistema yra naujos kartos – atsiradusi natūralios evoliucijos,

siekiant optimizuoti praktikoje pasitaikančių uždavinių sprendimą, būdu. Anot autoriaus ši kriptosistema reikalauja 58% (50%, atitinkamai) mažiau skaičiavimo laiko ir 85% (91%, atitinkamai) sumažina šifruoto ir pasirašyto pranešimo ilgio padidėjimą lyginant su StE kriptosistemomis, kurių sudėtingumas pagrįstas diskrečiojo logaritmo radimo uždavinio sudėtingumu (faktorizacijos – sveikųjų skaičių skaidymo į pirminių skaičių sandaugą uždavinio sudėtingumu, atitinkamai). Teorinis šių teiginių (skaičių) patvirtinimas pateiktas *2 lentelėje*. Pasiūlytoji kriptosistema buvo pavadinta „*Signcryption*“ [Zhe97].

Schema	**Skaičiavimo sudėtingumas**	**Pranešimo pailgėjimas (bitais)**
StE schema RSA parašas ir šifravimas	EXP=2, HASH=1, ENC=1 (EXP=2, HASH=1, DEC=1)	$\|n_a\| + \|n_b\|$
StE schema DSS+ElGamal šifravimas	EXP=3, MUL=1, DIV=1 ADD=1, HASH=1, ENC=1 (EXP=2.17, MUL=1, DIV=2 ADD=0, HASH=1, DEC=1)	$2\|q\| + \|p\|$
StE schema Schnorr parašas + ElGamal šifravimas	EXP=3, MUL=1, DIV=0 ADD=1, HASH=1, ENC=1 (EXP=2.17, MUL=1, DIV=0 ADD=0, HASH=1, DEC=1)	$\|hash(\cdot)\| + \|q\| + \|p\|$
Signcryption SCS1	EXP=1, MUL=0, DIV=1 ADD=1, HASH=2, ENC=1 (EXP=1.17, MUL=2, DIV=0 ADD=0, HASH=2, DEC=1)	$\|KH.(\cdot)\| + \|q\|$
Signcryption SCS2	EXP=1, MUL=1, DIV=1 ADD=1, HASH=2, ENC=1 (EXP=1.17, MUL=2, DIV=0 ADD=0, HASH=2, DEC=1)	$\|KH.(\cdot)\| + \|q\|$

2 lentelė. StE ir Signcryption schemų palyginimas

Čia:
EXP = kėlimo laipsniu moduliu skaičius (trupmena nurodo vidutinę vertę),
MUL = daugybos moduliu operacijų skaičius,
DIV = dalybos moduliu operacijų (inversijų) skaičius,
ADD = sudėties arba atimties moduliu operacijų skaičius,
HASH = operacijų su maišos funkcija skaičius,
ENC = šifravimo operacijų, naudojant privataus rakto šifrą, skaičius,
DEC = dešifravimo operacijų, naudojant privataus rakto šifrą, skaičius,
Skliausteliuose pateikiamas operacijų skaičius vykdant dešifravimą ir parašo tikrinimą arba *Unsigncryption* funkciją.

6 Signcryption

6.1 Sutrumpintas ElGamalio skaitmeninis parašas

Kaip jau minėta *3.2 ElGamalio skaitmeninio parašo schema*, ElGamalio skaitmeninis parašas pranešimui m yra pora (r, s):

$$r \equiv q^k \pmod{p}, \quad (1)$$

$$s \equiv (hash(m) - xr) \cdot k^{-1} \pmod{(p-1)}. \quad (2)$$

Kur:

p – didelis pirminis skaičius,

q – primityvioji multiplikatyviosios grupės $\mathbb{F}_p^*$ šaknis,

k – natūralusis skaičius, toks, kad $0 \leq k \leq p\text{-}1$ ir $\gcd(k, p\text{-}1) = 1$,

x – privatus raktas,

hash – maišos funkcija.

Sutrumpintas ElGamalio skaitmeninis parašas sudaromas pakeitus r ir s skaičiavimą [Zhe98]:

1. Nustatome, kad $r = hash(d, m)$, kur $d = g^k \pmod p$, g natūralusis skaičius laipsnyje q moduliu p, $0 \leq d \leq p\text{-}1$,
2. s skaičiavimą pakeičiame taip:
    a. (1) $hash(m)$ pakeičiame į 1 ir paliekame r arba r pakeičiame į 1, o $hash(m)$ į r,
    b. pakeičiame formą iš $s = (...)/\ k$ į $s = k\ /\ (...)$.

Patikrinimui ar (r, s) yra pranešimo m parašas, reikia apskaičiuoti $d = g^k \pmod p$ su žinomais r, s, g, p ir y_A ir patikrinti ar $hash(d, m)$ lygus r.

0je pateikiamos sutrumpintos parašo schemos, pažymėtos SDSS1 ir SDSS2, kur:

1. pirma raidė „S“ reiškia sutrumpinimą (shortened, angl.),
2. parametrai p, q ir g sutampa su naudojamais DSS,
3. k yra natūralusis skaičius $[1, \ldots, q\text{-}1]$, x_A privatus raktas, $y_A = g^{x_A} \pmod p$ viešasis raktas,
4. $|t|$ nurodo skaičiaus t dydį (ilgį) bitais,
5. SDSS1 yra efektyvesnė nei SDSS2, nes antru atveju reikia atlikti papildomą daugybos moduliu veiksmą.

Pavadinimas	**Pranešimo *m* parašas (*r*, *s*)**	**Parašo tikrinimas**	**Parašo ilgis**
SDSS1	$r = hash(g^k \pmod p, m)$, $s = k/(r + x_A) \pmod q$.	$d = (y_A \cdot g^r)^s \pmod p$, tikrinama ar $hash(d, m) = r$.	$\|hash(\cdot)\| + \|q\|$
SDSS2	$r = hash(g^k \pmod p, m)$, $s = k/(1 + x_A \cdot r) \pmod q$.	$d = (y_A \cdot g^r)^s \pmod p$, tikrinama ar $hash(d, m) = r$.	$\|hash(\cdot)\| + \|q\|$

3 lentelė. Sutrumpintos skaitmeninių parašų schemos

6.2 Signcryption realizacija sutrumpinto skaitmeninio parašo schemos pagrindu

Vienas iš pagrindinių motyvų pasirenkant SDSS schemą Signcryption realizacijai yra tai, kad nors sutrumpintoje skaitmeninio parašo schemoje $g^x \bmod p$ nėra pateikiamas, jis gali būti lengvai apskaičiuojamas iš parašo (r, s) ir kitų viešų schemos parametrų.

Signcryption schemoje naudojami parametrai pateikti *4 lentelėje*.

Viešai pateikiami parametrai: p – didelis pirminis skaičius, q – primityvioji multiplikatyviosios grupės $\mathbb{F}_p^*$ šaknis, g – natūralusis skaičius laipsnyje q moduliu p, $[0 \leq d \leq p\text{-}1]$, *hash* – tam tikra maišos funkcija, kurios santrauka ne trumpesnė nei 128 bitai, *KH* – tam tikra kontrolinio parašo maišos funkcija, (E, D) – privataus rakto (simetriniai) šifravimo ir dešifravimo algoritmai.
Pranešimo siuntėjo A raktai: x_A – privatus siuntėjo A raktas, pasirinktas atsitiktiniu būdu iš $[1, ..., q\text{-}1]$, y_A – viešas siuntėjo A raktas ($y_A = g^{x_A} \pmod p$).

Pranešimo gavėjo B raktai:
x_B – privatus gavėjo *B* raktas, pasirinktas atsitiktiniu būdu iš [1, ..., q -1],
y_B – viešas gavėjo *B* raktas ($y_B=g^{x_B} \pmod p$).

4 lentelė. Signcryption schemoje naudojami parametrai

Metodo aprašyme *E* ir *D* žymimi privataus rakto (simetriniai) šifravimo ir dešifravimo algoritmai (pavyzdžiui aprašytasis AES). Pranešimo *m* šifravimas su raktu *k*, paprastai šifrų blokų grandinės, – CBC (cipher block chaining, angl.), – režimu, žymimas $E_k(m)$, šifruoto teksto *c* dešifravimas su raktu *k* žymimas $D_k(c)$. Žymėjimas $KH_k(m)$ naudojamas norint pažymėti kontrolinio parašo maišos funkcijos *KH* su raktu *k* panaudojimą.

Signcryption schemos realizacija labai paprasta: jei pranešimo siuntėjas *A* nori išsiųsti pranešimą gavėjui *B*, jam reikia atlikti šias operacijas:

1. Atsitiktinai pasirinkti x iš [1, ..., q -1] ir paskaičiuoti $k = hash(y_B^x \pmod p)$. Perskelti *k* pusiau į du raktus k_1 ir k_2 (jei naudojama maišos funkcija su 128 bitų santrauka, gaunami du 64 bitų ilgio raktai),
2. Paskaičiuoti $r= KH_{k_2}(m)$,
3. Paskaičiuoti
   $s= x / (r + x_A) \bmod q$, jei naudojama SDSS1 schema arba
   $s= x / (1 + x_A \cdot r) \bmod q$, jei naudojama SDSS2 schema,
4. Tada šifruotas pranešimas bus $c = E_{k_1}(m)$,
5. Nusiųsti pranešimo gavėjui *B* Signcryption schemos rezultatą, – šifruotą ir pasirašytą tekstą, – (c, r, s).

Unsigncryption schemoje pasinaudojama tuo, kad pranešimo gavėjas *B*, gali lengvai paskaičiuoti $g^x \bmod p$, turėdamas r, s, g, p. Gavęs iš siuntėjo *A* pranešimą (c, r, s) gavėjas *B* atlieka šiuos veiksmus:

1. Apskaičiuoja *k* iš r, s, g, p, y_A ir x_B:
   $k = hash((y_A \cdot g^r)^{s \cdot x_B} \pmod p)$ jei naudojama SDSS1 schema arba
   $k = hash((g \cdot y_A^r)^{s \cdot x_B} \pmod p)$ jei naudojama SDSS1 schema,
2. Perskelti *k* pusiau į du raktus k_1 ir k_2,
3. $m = D_{k_1}(c)$,
4. Priimti paranešimą *m* tik jei $KH_{k_2}(m)$ lygus r.

7 Praktinis patikrinimas

Praktiniam patikrinimui buvo sukurtos realizacijos Python (64-bitų 2.7.2 versija) programavimo kalba, naudojant kriptografinę biblioteką pyCrypto (2.5 versija). Bandymai buvo atliekami kompiuteryje su AMD Phenom™ II X2 550 (3.10GHz) 64- bitų procesoriumi, 4 GB operatyviosios atminties, Windows 7 Ultimate SP1 64-bitų operacine sistema.

Kiekvienai kriptosistemai buvo atlikta po 100 iteracijų kiekvienai rakto ir teksto dydžių porai. Skaičiuojama buvo su trim skirtingais rakto dydžiais: 1024 bitų, 1536 bitų ir 2048 bitų, šifruojami buvo 2048 baitų, 10240 baitų ir 51200 baitų ilgio pranešimai. Taigi kiekvienai kriptosistemai buvo atlikta po 900 iteracijų, bendrai atlikta 2700 iteracijų. Analizei naudojamas trukmės aritmetinis vidurkis sekundėmis, skliausteliuose pateikiamas eksperimentinio standartinio aritmetinio vidurkio nuokrypio santykis su aritmetiniu vidurkiu procentais:

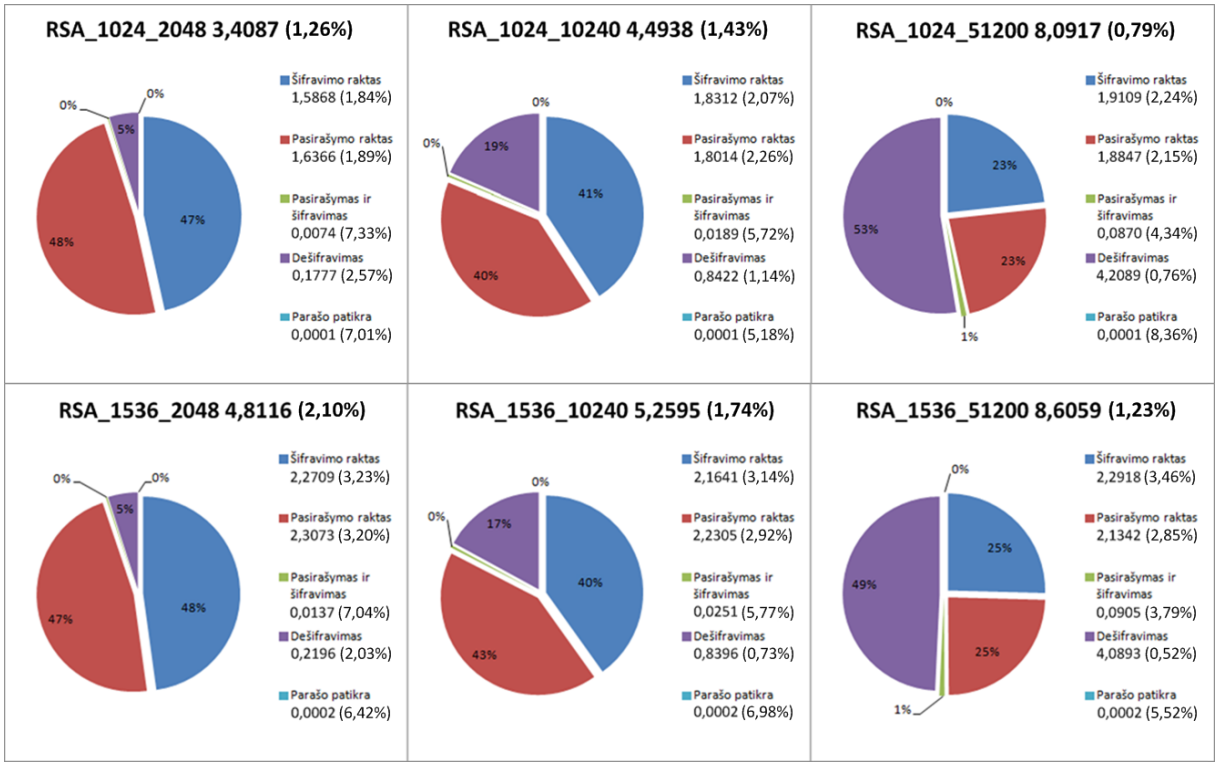
$$\sigma_t = \frac{1}{\bar{t}} \cdot \sqrt{\frac{1}{n} \cdot \left(\frac{\sum_{i=1}^{n}(t_i - \bar{t})^2}{n-1}\right)}, n=100.$$

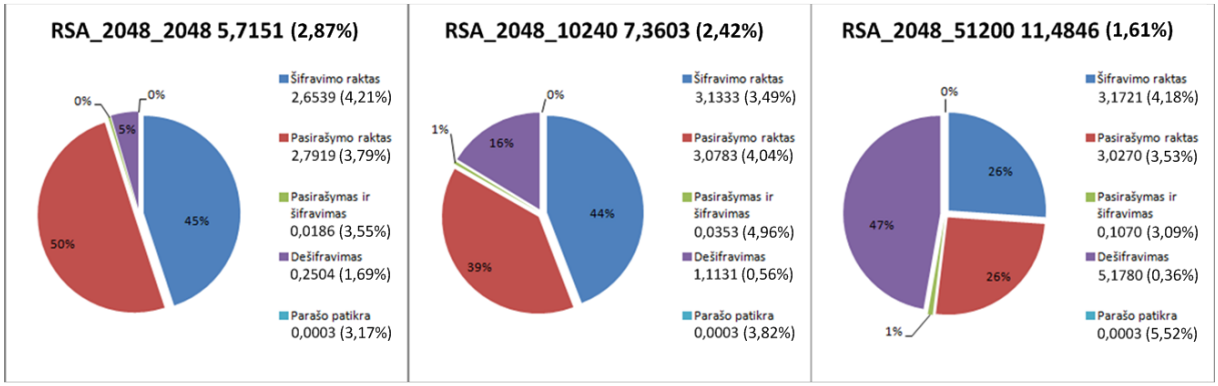
7.1 RSA kriptosistema

Viešojo rakto kriptosistemos negali šifruoti pranešimų, ilgesnių nei naudojamo rakto ilgis, todėl realizuotame algoritme pranešimas buvo skaidomas į blokus, kurių ilgis sutampa su naudojamu raktu: naudojant 1024 bitų dydžio raktą, pranešimas buvo skaidomas į 1024 bitų ilgio blokus, naudojant 2048 bitų raktą – į 2048 bitų ilgio blokus. Užšifruoti blokai jungiami į vientisą šifruotą pranešimą. Dešifravimo metu analogiškai – šifruotas pranešimas skaidomas į blokus, dešifravus bloką jis jungiamas į vientisą pranešimą.

2 paveiksliuke pateikiamos šifravimo ir dešifravimo bei parašo patikros trukmės, įskaitant šifravimo rakto radimo laiką. Kiekvienai rakto dydžio ir teksto ilgio porai pateikiamas atskiras blokas, kuriame:

- Antraštė sudaryta pagal principą AAA_BBBB_CCCCC, kur:
  - AAA – kriptosistemos pavadinimas;
  - BBBB – naudojamo rakto ilgis bitais;
  - CCCCC – šifruojamo pranešimo dydis baitais;
- Greta antraštės pateikiama operacijos trukmė sekundėmis;
- RSA kriptosistemai operacijos apima:
  - Šifravimo raktas – pirminių skaičių p ir q radimą, $n=p*q$ paskaičiavimą, sveiko skaičiaus $1 < e < \varphi(n)$ parinkimą bei $d=e^{-1} \pmod{\varphi(n)}$ paskaičiavimą. Viešas raktas yra pora (n, e), privatus – (n, d);
  - Pasirašymo raktas – apskaičiavimas analogiškas šifravimo rakto skaičiavimui;
  - Pasirašymas ir šifravimas – apima pranešimo santraukos panaudojant MD5 maišos funkciją radimą, santraukos pasirašymą bei santraukos ir parašo šifravimą;
  - Dešifravimas – apima šifruoto pranešimo dešifravimą bei pranešimo ir skaitmeninio parašo atskyrimą;
  - Parašo patikra – apima skaitmeninio parašo tikrinimo operaciją.

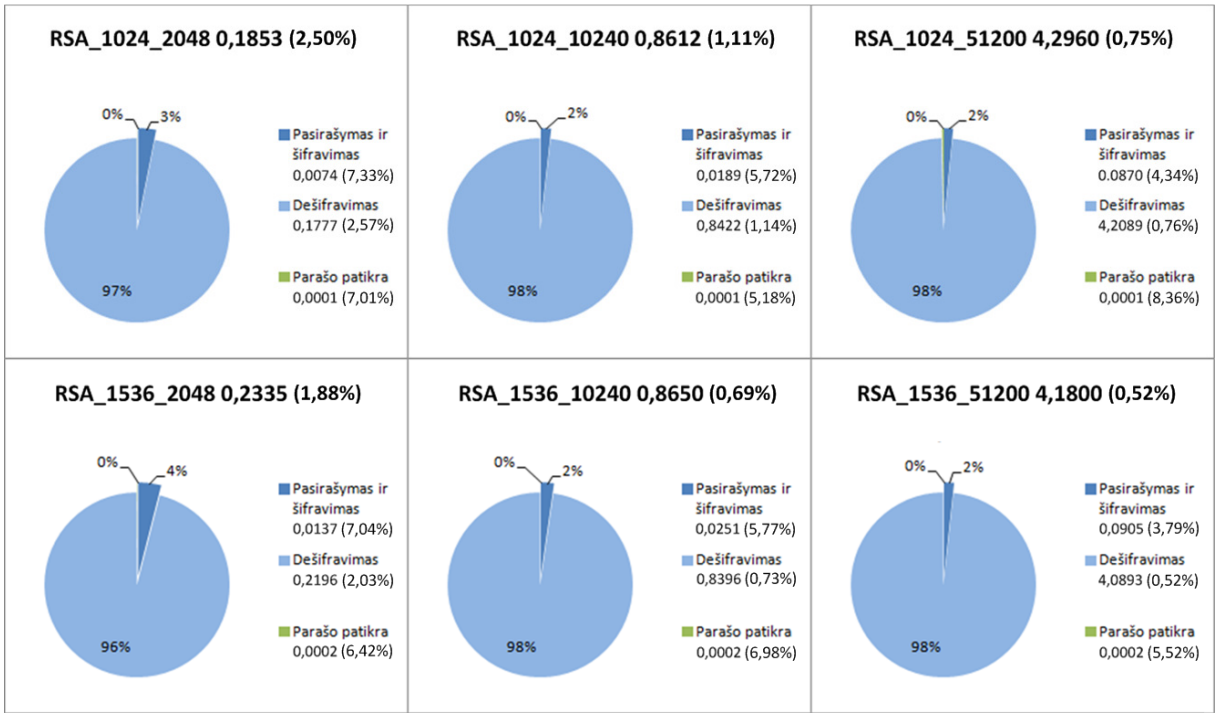


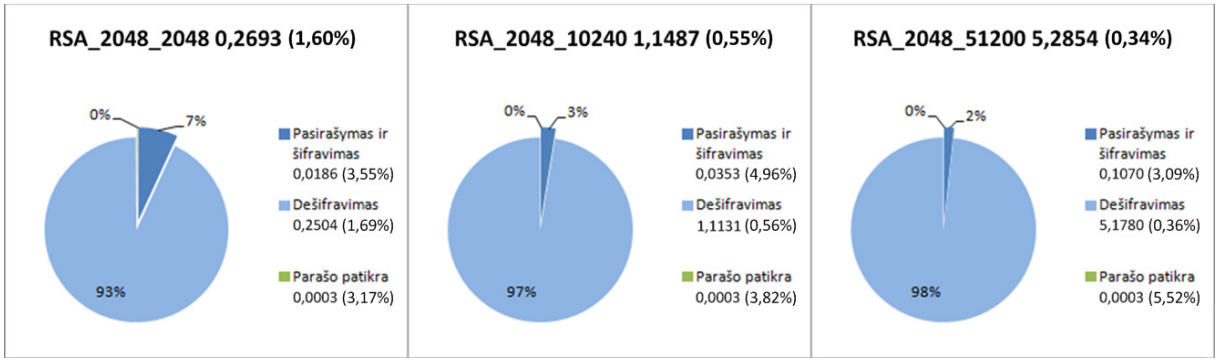


2 paveiksliukas. RSA kriptosistemos šifravimo ir dešifravimo bei parašo patikros trukmės, **įskaitant** šifravimo ir pasirašymo raktų generavimo trukmę, *sekundės*

Kaip matosi iš *2 paveiksliuko* RSA kriptosistemoje daugiausiai trunka raktų generavimo bei dešifravimo operacijos, be to, kad dešifravimo trukmę tiesiogiai įtakoja dešifruojamo pranešimo ilgis. Tuo tarpu šifravimo operacija atliekama labai sparčiai – ji trunka mažiau nei 1% visos operacijos trukmės.

3 paveiksliuke pateikiamos šifravimo ir dešifravimo bei parašo patikros trukmės, neįskaitant šifravimo rakto radimo laiko. Blokų struktūra ir reikšmės identiškos pateiktoms *2 paveiksliuke*.

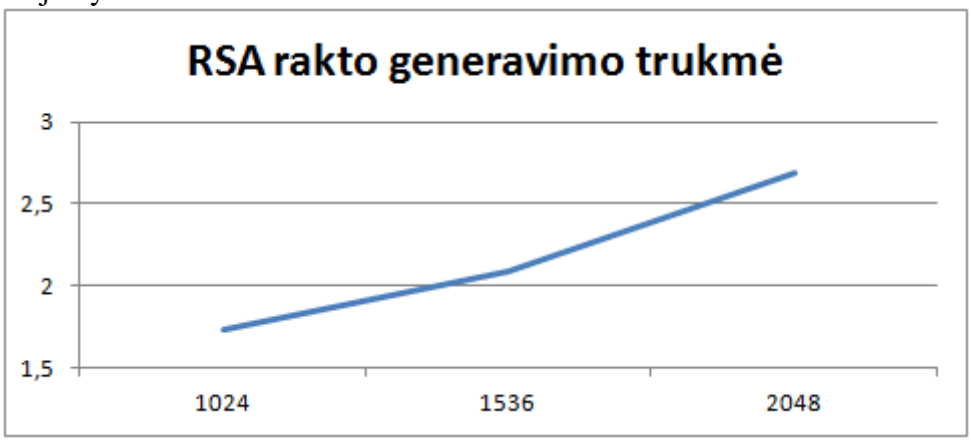




3 paveiksliukas. RSA kriptosistemos šifravimo ir dešifravimo bei parašo patikros trukmės, **neįskaitant** šifravimo ir pasirašymo raktų generavimo trukmės, *sekundės*

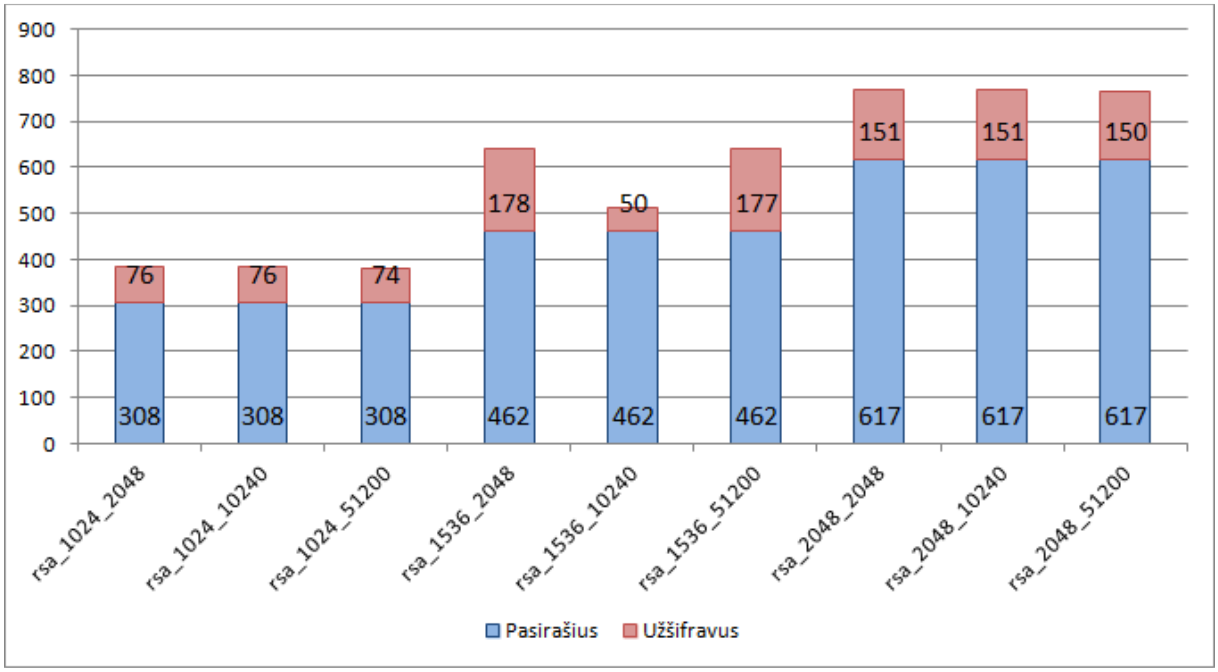
3 paveiksliuke matosi, kad pasirašymo ir šifravimo operacijos taip pat priklauso nuo pranešimo ilgio, tačiau didėjant pranešimo ilgiui šių operacijų trukmės augimas ženkliai mažesnis nei dešifravimo operacijos, dėl to joms tenkanti santykinė trukmė mažėja.

4 paveiksliuke matosi, kad RSA rakto generavimo trukmė tiesiogiai ir beveik tiesiškai priklauso nuo jo dydžio.



4 paveiksliukas. RSA kriptosistemos šifravimo rakto generavimo trukmė, *sekundės*

5 paveiksliuke pateikiamas pranešimo ilgėjimas pasirašant šifruojant. Iš jo matosi, kad pailgėjimas tiek pasirašant, tiek šifruojant tiesiogiai priklauso nuo naudojamo rakto dydžio. 1536 bitų dydžio raktu šifruojant 10240 baitų pranešimą dydžio padidėjimas mažesnis nei šifruojant 2048 ir 51200 baitų ilgio pranešimus yra dėl atsitiktinai sutampančių dydžių: skaidant pranešimą į 1536 bitų ilgio blokus 10240 baitų pranešimas skaidomas į 55,74 blokus, tai yra išnaudojama 74% paskutinio bloko ilgio, tuo tarpu 2048 ir 51200 baitų ilgio pranešimai skyla atitinkamai į 13,07 ir 269,07 blokus, t.y. išnaudojama tik 7% bloko ilgio, nors šifruojamas visas blokas dėl ko gaunamas papildomas nereikalingas pailgėjimas: $(\frac{26}{93} \approx \frac{50}{178})$.



5 paveiksliukas. Pranešimo ilgio padidėjimas naudojant RSA StE algoritmą, *baitai*

7.2 ElGamalio kriptosistema

Kaip minėta aprašant RSA kriptosistemą – viešojo rakto kriptosistemos negali šifruoti pranešimų, ilgesnių nei naudojamo rakto ilgis, todėl realizuojant ElGamalio kriptosistemą pranešimas taip buvo skaidomas į blokus, kurių ilgis sutampa su naudojamu raktu.

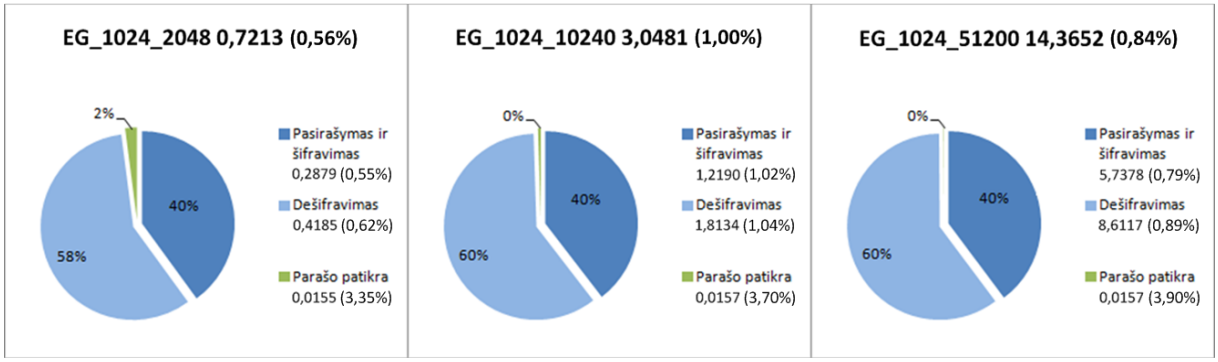
6 paveiksliuke pateikiamos šifravimo ir dešifravimo bei parašo patikros trukmės, įskaitant šifravimo rakto radimo laiką. Kiekvienai rakto dydžio ir teksto ilgio porai pateikiamas atskiras blokas, kuriame:

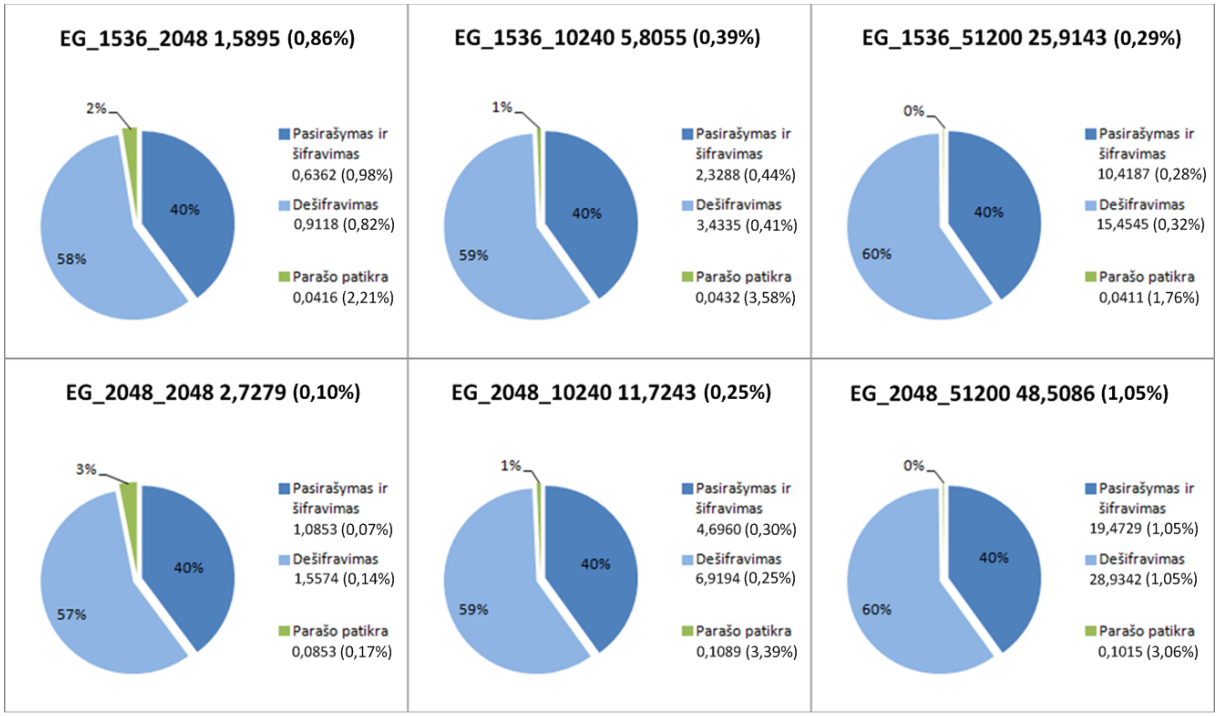
- Antraštė sudaryta pagal principą AAA_BBBB_CCCCC, kur:
  - AAA – kriptosistemos pavadinimo sutrumpinimas;
  - BBBB – naudojamo rakto ilgis bitais;
  - CCCCC – šifruojamo pranešimo dydis baitais;
- Greta antraštės pateikiama operacijos trukmė sekundėmis;
- ElGamalio kriptosistemai operacijos apima:
  - Šifravimo raktas – pirminio skaičiaus p, primityviosios multiplikatyviosios grupės $\mathbb{F}_p^*$ šaknies g radimą, sveiko skaičiaus $1 < x < p\text{-}1$ parinkimą bei $y=g^x \pmod p$ paskaičiavimą. Viešas raktas yra (p, g, y), privatus – (x);
  - Pasirašymo raktas – apskaičiavimas analogiškas šifravimo rakto skaičiavimui;
  - Pasirašymas ir šifravimas – apima pranešimo santraukos panaudojant MD5 maišos funkciją radimą, santraukos pasirašymą bei santraukos ir parašo šifravimą;
  - Dešifravimas – apima šifruoto pranešimo dešifravimą bei pranešimo ir skaitmeninio parašo atskyrimą;
  - Parašo patikra – apima skaitmeninio parašo tikrinimo operaciją.

6 paveiksliukas. ElGamalio kriptosistemos šifravimo ir dešifravimo bei parašo patikros trukmės, **įskaitant** šifravimo ir pasirašymo raktų generavimo trukmę, *sekundės*

Kaip matosi iš *6 paveiksliuko* ElGamalio kriptosistemoje raktų generavimo operacijos, kaip ir RSA kriptosistemoje, trunka ilgai, tačiau, skirtingai nei RSA kriptosistemoje, didėjant pranešimo ilgiui ilgėja ne tik dešifravimo, bet ir pasirašymo bei šifravimo operacijų trukmė.

7 paveiksliuke pateikiamos šifravimo ir dešifravimo bei parašo patikros trukmės, neįskaitant šifravimo rakto radimo laiko. Blokų struktūra ir reikšmės identiškos pateiktoms *6 paveiksliuke*.

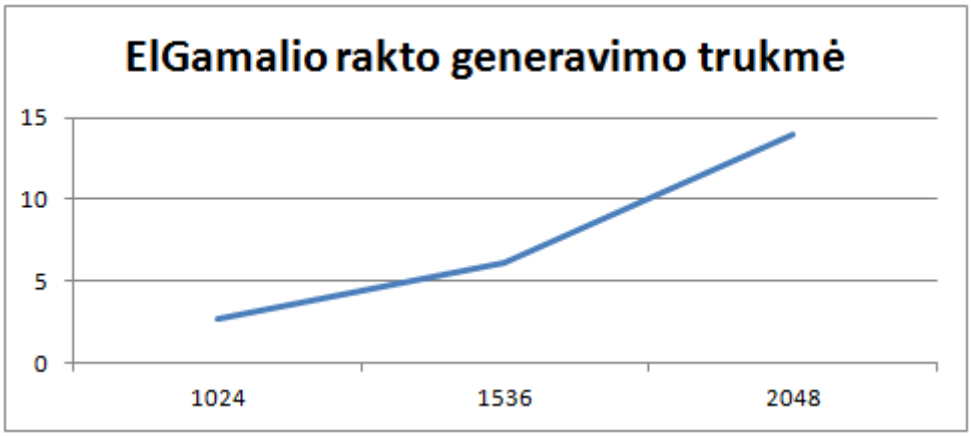




7 paveiksliukas. ElGamalio kriptosistemos šifravimo ir dešifravimo bei parašo patikros trukmės, **neįskaitant** šifravimo ir pasirašymo raktų generavimo trukmės, *sekundės*

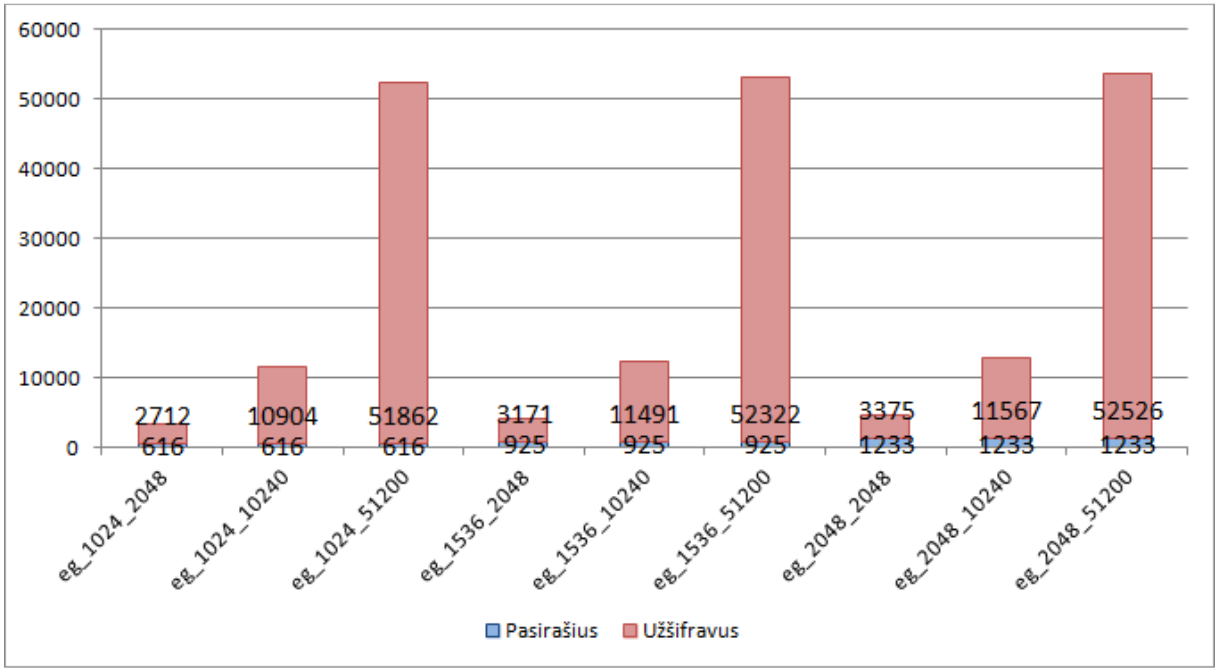
7 paveiksliuke matosi, kad pasirašymo ir šifravimo bei dešifravimo operacijų trukmė yra beveik tiesinėje priklausomybėje: nepriklausomai nuo naudojamo rakto dydžio bei pranešimo ilgio jų santykis beveik nekinta. Pažymėtina, kad dešifravimo operacijos trukmė visgi šiek tiek labiau ilgėja priklausomai nuo pranešimo ilgio, tuo tarpu parašo patikrinimo operacijos trukmė ilgėjant pranešimui nykstamai mažėja.

8 paveiksliuke matosi, kad ElGamalio rakto generavimo trukmė tiesiogiai priklauso nuo jo dydžio, be to auga ženkliai sparčiau nei didėja rakto dydis.



8 paveiksliukas. ElGamalio kriptosistemos šifravimo rakto generavimo trukmė, *sekundės*

9 paveiksliuke pateikiamas pranešimo ilgėjimas pasirašant šifruojant. Iš jo matosi, kad pailgėjimas tiek pasirašant, tiek šifruojant tiesiogiai priklauso nuo pranešimo ilgio ir šiek tiek nuo naudojamo rakto dydžio. Dėl ElGamalio kriptosistemos ypatybių šifruoto pranešimo ilgis dvigubėja, kadangi mes ne tik šifruojame, bet prieš tai ir pasirašome tai pritaikius StE algoritmą pailgėjimas yra dvigubas pranešimo ir parašo ilgis, parašas savo ruožtu – dvigubas taikomos maišos funkcijos santraukos ilgis.



9 paveiksliukas. Pranešimo ilgio padidėjimas naudojant ElGamalio StE algoritmą, *baitai*

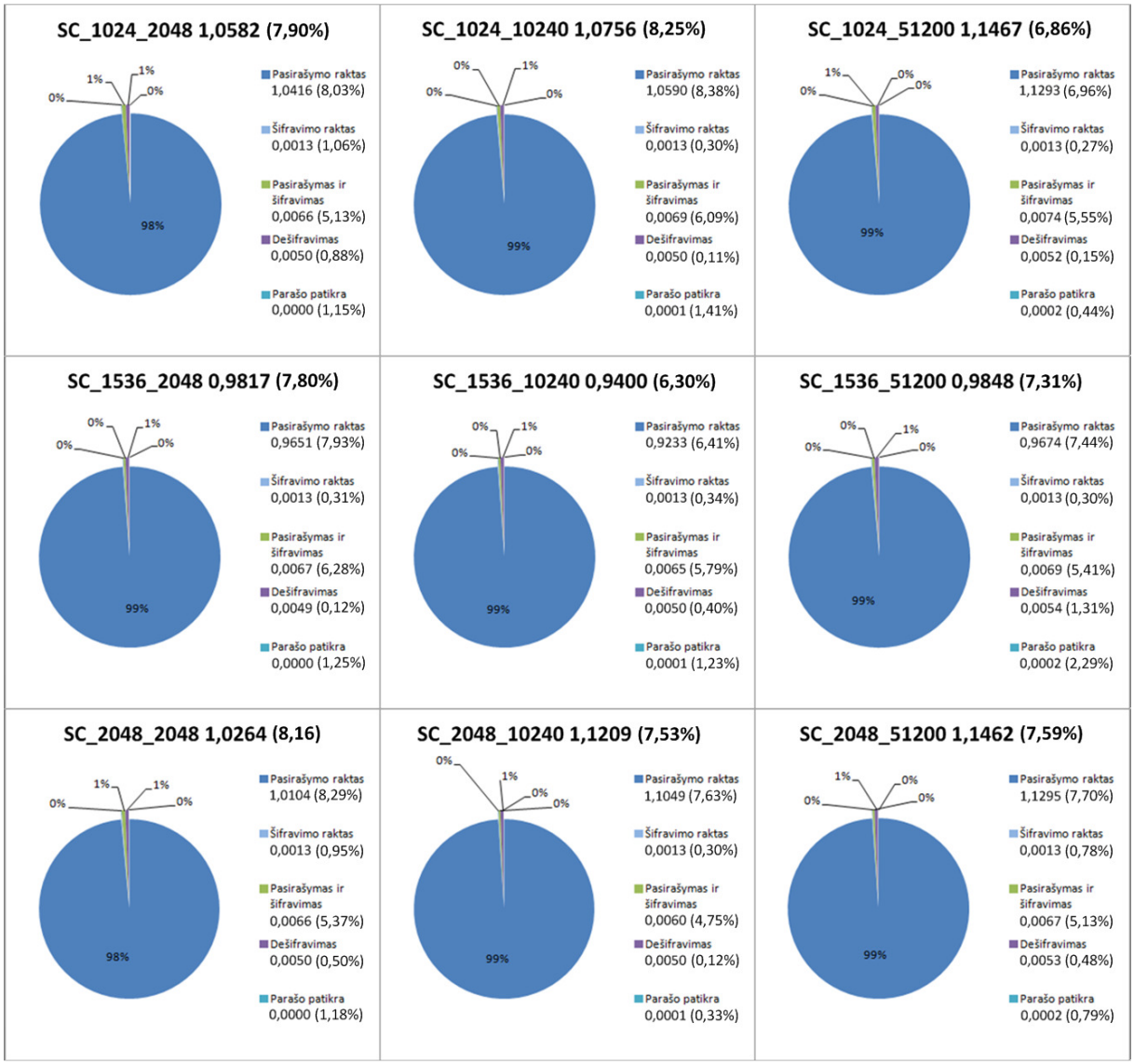
7.3 Signcryption kriptosistema

Signcryption kriptosistemoje šifravimas atliekamas naudojant blokinio šifro, – šioje realizacijoje AES-128, – algoritmą, kuriame jau yra realizuotas pranešimo skaidymas į atitinkamo ilgio blokus, dėl to skirtingai nei RSA ir ElGamalio sistemų atvejų pranešimas į blokus papildomai nėra skaidomas.

10 paveiksliuke pateikiamos šifravimo trukmės, įskaitant šifravimo rakto radimo laiką. Kiekvienai rakto dydžio ir teksto ilgio porai pateikiamas atskiras blokas, kuriame:

- Antraštė sudaryta pagal principą AAA_BBBB_CCCCC, kur:
  - AAA – kriptosistemos pavadinimo sutrumpinimas;
  - BBBB – naudojamo rakto ilgis bitais;
  - CCCCC – šifruojamo pranešimo dydis baitais;
- Greta antraštės pateikiama operacijos trukmė sekundėmis;
- Signcryption kriptosistemai operacijos apima:
  - Pasirašymo raktas – pirminio skaičiaus p, primityviosios multiplikatyviosios grupės $\mathbb{F}_p^*$ šaknies q radimą, sveiko skaičiaus $1 < g < p\text{-}1$ parinkimą, atsitiktinio skaičiaus $1 < x_A < p\text{-}1$ parinkimą (siuntėjo privatus raktas), atsitiktinio skaičiaus $1 < x_B < p\text{-}1$ parinkimą (gavėjo privatus raktas), $y_A = g^{x_A} \pmod p$ ir $y_B = g^{x_B} \pmod p$ paskaičiavimą (atitinkamai siuntėjo ir gavėjo viešieji raktai);
  - Šifravimo raktas – pranešimo 256 bitų santraukos panaudojant SHA-256 maišos funkciją radimas bei išskaidymas į du 128 bitų raktus k_1 ir k_2;
  - Pasirašymas ir šifravimas – apima pranešimo šifravimą blokiniu šifru AES-128, panaudojant raktą k_1, ir parametrų $r = hmac(k_2, m)$ bei $s = x/(r + x_A) \pmod q$ radimą;
  - Dešifravimas – apima dešifravimo rakto $k_d = hash((y_A \cdot g^r)^{s \cdot x_B} \pmod p)$, naudojant SHA-256 maišos funkciją radimą, 256 bitų k_d išskaidymą į du 128 bitų raktus k_{d1} ir k_{d2}, šifruoto pranešimo dešifravimą naudojant blokinį šifrą AES-128, naudojant raktą k_{d1};

- Parašo patikra – apima skaitmeninio parašo tikrinimo operaciją: tikrinimą ar teisinga lygybė $m \equiv hmac(k_{d2}, m)$.



10 paveiksliukas. Signcryption kriptosistemos šifravimo ir dešifravimo bei parašo patikros trukmės, **įskaitant** šifravimo ir pasirašymo raktų generavimo trukmę, *sekundės*

Kaip matosi iš *10 paveiksliuko* Signcryption kriptosistemoje visų operacijų trukmė, nepriklausomai nuo pranešimo ilgio bei rakto dydžio, praktiškai lygi pasirašymo rakto generavimo operacijos trukmei.

Įdomi detalė – šio bandymo rezultatų ribose trumpiausiai trunka šifravimo ir dešifravimo bei parašo patikros operacijos su vidutiniu, – 10240 baitų, – tekstu ir vidutinio dydžio, – 1536 bitų, – raktu. Taip yra todėl, kad, kaip jau minėta, operacijų ilgis praktiškai lygus pasirašymo rakto generavimo laikui, o pasirašymo rakto generavimas apima pirminio skaičiaus radimą – atsitiktinio didelio sveiko skaičiaus parinkimą ir Rabino – Millerio testo (Miller–Rabin primality test, angl.) pagalba tikrinimo ar šis skaičius yra pirminis, taigi kiek užtruks tokio skaičiaus radimas, o tuo pačiu ir rakto generavimas yra laimės dalykas: maksimalus skaičius operacijų (spėjimų) yra $100*(\log_2 u+1)$, kur u – didžiausias norimo dydžio bitais skaičius, taigi 1024 bitų dydžio skaičiui maksimalus operacijų (spėjimų) skaičius yra 102500 (1536 bitų –

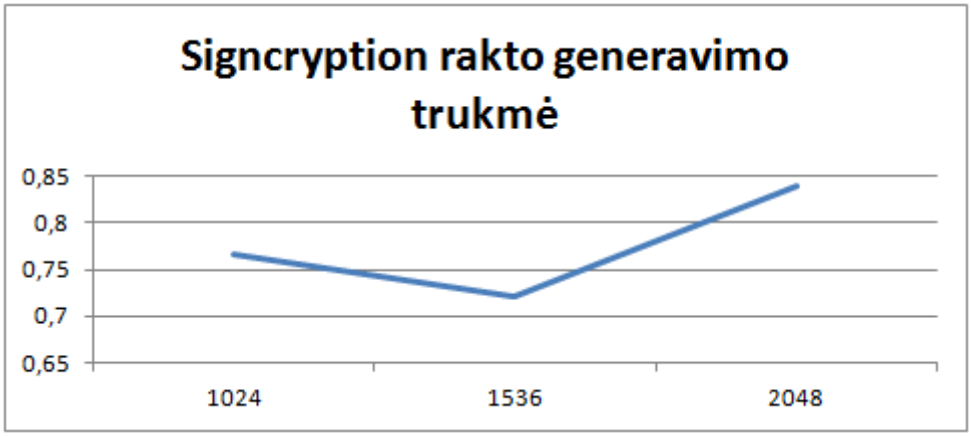
153700, 2048 bitų - 204900) [FS05]. Būtina pažymėti, kad šio bandymo rezultatas nėra dėsningumas – pakartotinai atlikus 100 iteracijų rezultatas gali pasikeisti.

11 paveiksliuke pateikiamos šifravimo ir dešifravimo bei parašo patikros operacijų trukmės, neįskaitant šifravimo rakto radimo laiko. Blokų struktūra ir reikšmės identiškos pateiktoms *10 paveiksliuke*.

11 paveiksliukas. Signcryption kriptosistemos šifravimo ir dešifravimo bei parašo patikros trukmės, **neįskaitant** šifravimo ir pasirašymo raktų generavimo trukmę, *sekundės*

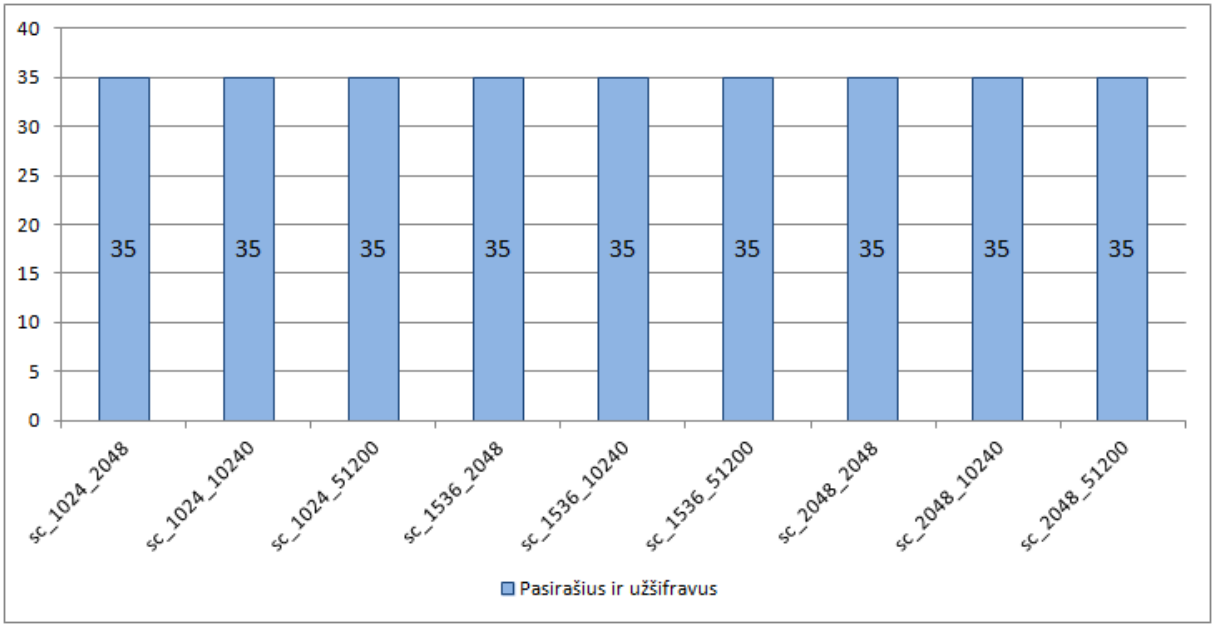
11 paveiksliuke matosi, kad tik parašo patikrą įtakoja pranešimo ilgis, bet neįtakoja rakto dydis. Visos kitos operacijos neturi aiškios priklausomybės nuo rakto dydžio ir pranešimo ilgio.

12 paveiksliuke matosi, kad Signcryption pasirašymo rakto generavimo trukmė nepriklauso nuo jo dydžio, kaip jau minėta ji priklauso nuo atsitiktinių skaičiaus generatoriaus.



12 paveiksliukas. Signcryption kriptosistemos pasirašymo rakto generavimo trukmė, *sekundės*

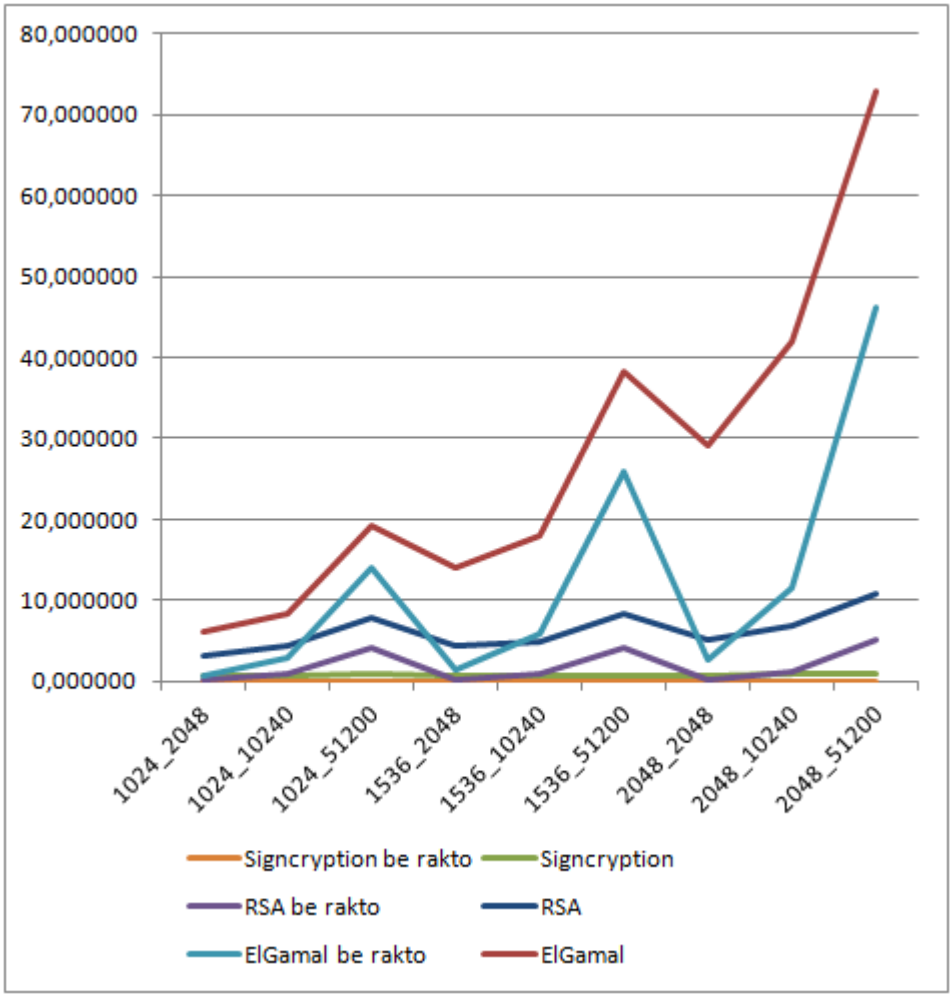
13 paveiksliuke pateikiamas pranešimo pailgėjimas naudojant Signcryption kriptosistemą. Šifravimas atliekamas blokiniu šifru AES-128, todėl šifruoto pranešimo ilgis nesiskiria nuo nešifruoto. 35 baitų padidėjimas, nepriklausomai nuo pranešimo ilgio ir rakto dydžio gaunasi, nes kartu su šifruotu pranešimu siunčiami parametrai *r* ir *s*, kurie yra atitinkamai 16 ir 19 (arba 15 ir 20 atitinkamai) baitų ilgio, jei *r* paskaičiavimui naudojamas HMAC kontrolinis parašas realizuotas MD5 maišos funkcijos pagrindu – būtent toks naudojamas bandomojoje realizacijoje.



13 paveiksliukas. Pranešimo ilgio padidėjimas naudojant Signcryption algoritmą, *baitai*

7.4 Kriptosistemų palyginimas

14 paveiksliuke pateikiama palyginamoji StE algoritmo trukmė įskaitant ir neįskaitant raktų generavimo laiką. Kaip matosi greičiausias algoritmas – Signcryption be rakto generavimo, ženkliai lėčiausias ElGamalio algoritmas su rakto generavimu.



14 paveiksliukas. Palyginamoji StE algoritmo trukmė įskaitant ir neįskaitant raktų generavimo laiką, *sekundės*

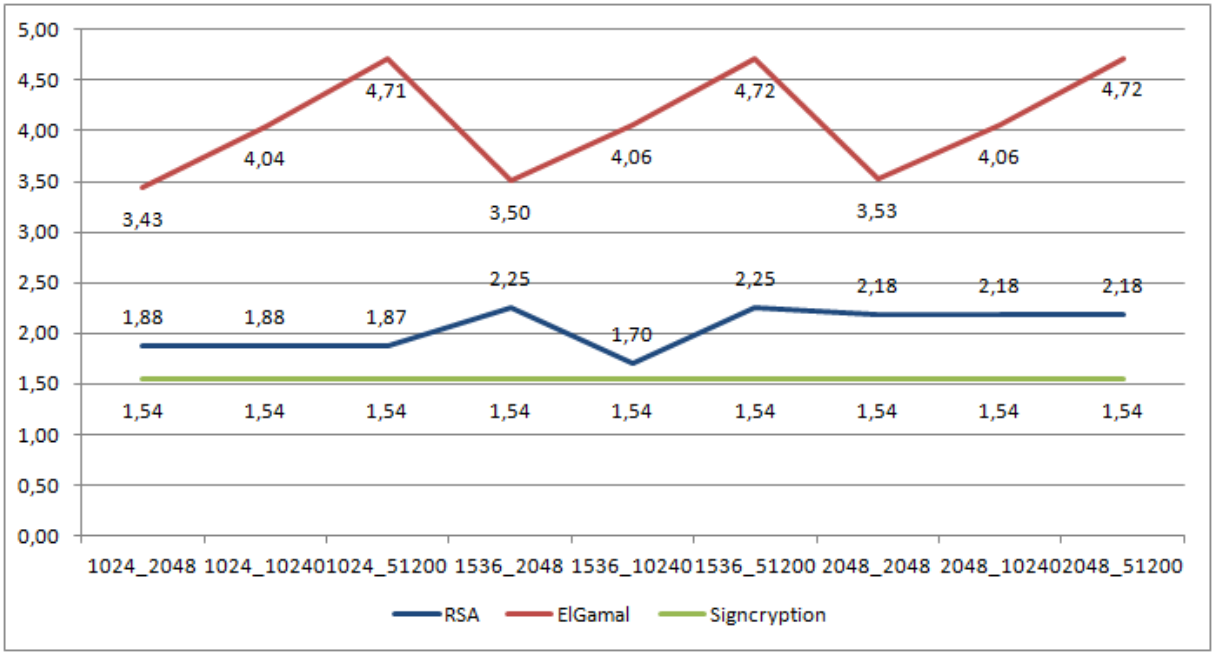
Kadangi algoritmų vykdymo trukmė skiriasi labai ženkliai: nuo 0,0111 iki 78,6836 sekundžių, reikšmių pateikimas grafike nėra pakankamai aiškus, todėl reikšmių skaitinės vertės pateikiamos *5 lentelėje*. Pateikiamas trukmės aritmetinis vidurkis 100-e iteracijų sekundėmis, skliausteliuose pateikiamas eksperimentinio standartinio aritmetinio vidurkio nuokrypio santykis su aritmetiniu vidurkiu procentais.

	Sign-cryption be rakto	Sign-cryption	RSA be rakto	RSA	ElGamal be rakto	ElGamal
1024_2048	0,0116 (2,94%)	1,0582 (7,90%)	0,1853 (2,50%)	3,4087 (1,26%)	0,7213 (0,56%)	6,5946 (2,89%)
1024_10240	0,0119 (3,52%)	1,0756 (8,25%)	0,8612 (1,11%)	4,4938 (1,43%)	3,0481 (1,00%)	8,9799 (2,17%)
1024_51200	0,0128 (3,22%)	1,1467 (6,86%)	4,2960 (0,75%)	8,0917 (0,79%)	14,3652 (0,84%)	20,0244 (1,23%)
1536_2048	0,0116 (3,59%)	0,9817 (7,80%)	0,2335 (1,88%)	4,8116 (2,10%)	1,5895 (0,86%)	15,0449 (3,16%)
1536_10240	0,0116 (3,30%)	0,9400 (6,30%)	0,8650 (0,69%)	5,2595 (1,74%)	5,8055 (0,39%)	19,5466 (2,60%)

1536_51200	0,0124 (3,06%)	0,9848 (7,31%)	4,1800 (0,52%)	8,6059 (1,23%)	25,9143 (0,29%)	39,4703 (1,34%)
2048_2048	0,0116 (3,09%)	1,0264 (8,16%)	0,2693 (1,60%)	5,7151 (2,87%)	2,7279 (0,10%)	30,6101 (2,88%)
2048_10240	0,0111 (2,58%)	1,1209 (7,53%)	1,1487 (0,55%)	7,3603 (2,42%)	11,7243 (0,25%)	46,4034 (2,75%)
2048_51200	0,0121 (2,85%)	1,1462 (7,59%)	5,2854 (0,34%)	11,4846 (1,61%)	48,5086 (1,05%)	78,6836 (1,82%)

5 lentelė. Palyginamoji StE algoritmo trukmė įskaitant ir neįskaitant raktų generavimo laiką, *sekundės*

15 paveiksliuke pateikiamas palyginamasis pranešimo pailgėjimas, palyginimui naudojant dešimtainio logaritmo reikšmes. X ašyje naudojamas žymėjimas AAAA_BBBBB, kur AAAA – rakto dydis bitais, BBBBB – pranešimo ilgis baitais. Kaip matosi didžiausias pranešimo pailgėjimas naudojant ElGamalio kriptosistemą, mažiausias – Signcryption.



15 paveiksliukas. Palyginamasis pranešimo ilgio padidėjimas ($\log_{10}$), *baitai*

8 Grupei skirto šifro padalijimo schema

Įprastose kriptografinėse schemose vientisas šifro raktas laikomas vienoje vietoje (kompiuteryje, žmogaus galvoje ir pan.), o asmuo, turintis priėjimą prie šio rakto, gali vienasmeniškai iššifruoti informaciją. Grupei skirto šifro padalijimo schema tai metodas, leidžiantis padalinti paslaptį tarp grupės asmenų taip, kad kiekvieno iš gavėjų turima paslapties dalis nesuteiktų jokios informacijos (galimybės atskleisti) apie paslaptį, tačiau susirinkusi visa grupė galėtų nesunkiai atskleisti paslaptį.

Vienas iš galimų grupei skirto šifro padalijimo schemos algoritmų gali būti toks:

- Paversti paslaptį sveiku skaičiumi S;
- Kiekvienam iš paslapties gavėjų, išskyrus vieną, įteikti atsitiktinį sveiką skaičių r_i;
- Vienam gavėjui duoti skaičių $s = S - r_1 - ... - r_i$;
- Tada susirinkusi visų gavėjų grupė nesunkiai galėtų paskaičiuoti S.

Grupei skirtų šifrų padalijimo schemos puikiai tinka itin svarbios informacijos saugojimui, kai sprendimo teisė priklauso ne nuo vieno, o nuo kelių asmenų, pavyzdžiui raketų paleidimo kodai, banko saugyklų slaptažodžiai ir pan.

8.1 Grupei skirto šifro Signcryption schema

Kaip nustatyta praktinių bandymų metu (žr. *7 skyrių*) Signcryption yra efektyvi aukšto saugumo lygio kriptosistema, idealiai tinkanti kai reikia pranešimo slaptumo (šifravimo), ir pranešimo autentiškumo bei vientisumo patvirtinimo (skaitmeninio parašo), esant mažiems skaičiavimo ir energijos ištekliams bei ribotam ir (arba) brangiam informacijos perdavimui.

Pasirodo grupei skirto šifro padalijimo schemą galima nesunkiai realizuoti Signcryption kriptosistemos pagrindu. Signcryption schemoje naudojami parametrai pateikti *6 lentelėje*.

Viešai pateikiami parametrai: p – didelis pirminis skaičius, q – primityvioji multiplikatyviosios grupės $\mathbb{F}_p^*$ šaknis, g – natūralusis skaičius $g=d^q \bmod p$, $[0 \le d \le p-1]$, *hash* – tam tikra maišos funkcija, kurios santrauka ne trumpesnė nei 128 bitai, *KH* – tam tikra kontrolinio parašo maišos funkcija, (E, D) – privataus rakto (simetrinis) šifravimo ir dešifravimo algoritmas (pvz. AES-128).
Pranešimo siuntėjo A raktai: x_A – privatus siuntėjo A raktas, pasirinktas atsitiktiniu būdu iš [1, ..., q -1], y_A – viešas siuntėjo A raktas ($y_A=g^{x_A} \pmod p$).
Pranešimo gavėjų B_i raktai: x_{B_i}– privatus i-tojo paslapties gavėjo B raktas, pasirinktas atsitiktiniu būdu iš [1, ..., q -1], y_{B_i}– viešas i-tojo paslapties gavėjo B raktas ($y_B = g^{x_B} \pmod p$), n – bendras paslapties gavėjų skaičius.

6 lentelė. Signcryption schemoje naudojami parametrai

Jei paslapties dalintojas A nori nusiųsti pranešimą m gavėjams B_i, kur i=[1, …, n], taip, kad tik visi gavėjai, susirinkę kartu galėtų jį perskaityti, jam reikia atlikti šias operacijas:

1. Atsitiktinai pasirinkti x iš [1, ..., q -1],
2. Paskaičiuoti gavėjų grupės viešąjį raktą $y_B=y_{B_1} \cdot \ldots \cdot y_{B_n} \pmod p$,
3. Paskaičiuoti $K = hash(y_B^x \pmod p)$,
4. Perskelti K pusiau į du raktus K_1 ir K_2,
5. Tada šifruotas pranešimas bus $c = E\ K_1(m)$,
6. Paskaičiuoti $r= KH\ K_2(m)$,
7. Paskaičiuoti
   $s= x / (r + x_A) \pmod q$, jei naudojama SDSS1 schema arba
   $s= x / (1 + x_A \cdot r) \pmod q$, jei naudojama SDSS2 schema (žr. *7 skyrių*),
8. Nusiųsti paslapties gavėjams B Signcryption schemos rezultatą, – šifruotą ir pasirašytą pranešimą, – (c, r, s). Išties c parametro (šifruoto pranešimo) netgi galima nesiųsti, o patalpinti jį sutartoje vietoje, pavyzdžiui dešifravimo įrenginyje.

Unsigncryption schemoje, kaip ir pačioje kriptosistemoje, pasinaudojama tuo, kad paslapties gavėjai B, gali lengvai paskaičiuoti $g^x \pmod p$, turėdami r, s, g, p. Gavę iš siuntėjo A pranešimą (c, r, s), kiekvienas gavėjas B atlieka šiuos veiksmus:

9. Apskaičiuoja K_i iš r, s, g, p, y_A ir x_{B_i}:
   $K_i = (y_A \cdot g^r)^{s \cdot x_{B_i}} \pmod p$ jei naudojama SDSS1 schema arba

$K_i = (g \cdot y_A^r)^{s \cdot x_{B_i}} (\mathrm{mod}\ p)$ jei naudojama SDSS2 schema (žr. *7 skyrių*),

Tada susirinkusi paslapties gavėjų grupė gali paskaičiuoti:

10. $K = hash(\prod_{i=1}^{n} K_i\ (\mathrm{mod}\ p))$,
11. Perskelti *K* pusiau į du raktus K_1 ir K_2,
12. Pranešimo dešifravimas $m = D\ K_1(c)$,
13. Patikrinti parašą ir priimti pranešimą *m* tik jei $KH\ K_2(m) \equiv r$.

Kaip matome norint padalinti paslaptį tarp *n* gavėjų Signcryption kriptosistemos algoritme praktiškai nieko nereikia keisti, tereikia:

a) *Signcryption* algoritme prieš skaičiuojant pagrindinį raktą *K* (žr. 3 algoritmo punktą) paskaičiuoti grupės viešąjį raktą, panaudojant visų paslapties gavėjų viešuosius raktus – atlikti papildomą daugybos modulyje operaciją.
Pastebėtina, kad nereikia nei skaičiuoti, nei siųsti jokių papildomų kintamųjų, taigi siunčiamos informacijos kiekis nepadidėja.

b) *Unsigncryption* algoritme taip pat atsiranda papildoma daugybos modulyje operacija $K = hash(\prod_{i=1}^{n} K_i\ (\mathrm{mod}\ p))$ (žr. 10 algoritmo punktą).

Norint padalinti kitą paslaptį, – nusiųsti kitą pranešimą, – tereikia atlikti visus veiksmus nuo pradžių, t.y. nuo 1 algoritmo punkto.

Be to grupei skirto šifro dalijimas pagal šią schemą pasižymi labai įdomiom savybėm:

1. Paslapties gavėjai privalės kiekvieną kartą susirinkti visi, net jei kiekvieną kartą bus siunčiamas tas pats pranešimas, nes parinkus kitą *x* pasikeičia ir visų parametrų, įskaitant K_i, reikšmės.
2. Susirinkę paslapties gavėjai gali drąsiai pateikti savo K_i reikšmę, nes ji kiekvieną kartą, nesvarbu ar bus siunčiamas tas pats pranešimas ar kitas, parinkus kitą *x* taip pat bus kita.
3. Pirmos dvi savybės suponuoja trečiąją: kiekvieną kartą siunčiant pranešimą **būtina** parinkti kitą *x* reikšmę, nes jei naudosime tą pačią, o kaip žinia parametrai *p*, *q*, *g*, x_A, y_A, x_{Bi}, y_{Bi} irgi yra tie patys, netgi siunčiant **kitą** pranešimą reikšmė K_i, nežiūrint į tai, kad jos apskaičiavimui bus naudojami kiti parametrai *r* ir *s*, irgi bus ta pati.

Tiesa šią savybę galima išnaudoti ir teigiamai: jei pavyktų sukurti įrenginį į kurį kiekvienas gavėjas įveda savo rakto dalį K_i, bet neatskleidžia jos kitiems gavėjams, tada gavėjams nebereikėtų kiekvieną kartą skaičiuoti savo rakto reikšmės, o jie galėtų turimą K_i naudoti panašiai kaip PIN kodą bankomatuose. Be to tada ir paslapties dalintojui nereikėtų parinkinėti jokių papildomų reikšmių (pvz. *x*) ir net skaičiuoti rakto, tereiktų paskaičiuoti *c*, *r* ir *s*.

Naudojant tam tikrą dešifravimo įrenginį, raktą *K* taip pat galima „paslėpti“ nuo gavėjų sukuriant kiek sudėtingesnę matematinę konstrukciją – perrašant 10-13 algoritmo žingsnius:

1. Vietoje 10-12 žingsnių palikti tik vieną: $m = D\ (hash(\prod_{i=1}^{n} K_i\ (\mathrm{mod}\ p))[:64])\ (c)$, kur [:64] nurodo, kad dešifravimo funkcija *D* turi naudoti maišos funkcijos sukurtos santraukos 1 – 64 bitus (jei raktas *K* yra 128 bitų, o atitinkamai K_1 ir K_2 yra 64 bitų);
2. 13 žingsnyje tikrinti ar $KH\ (hash(\prod_{i=1}^{n} K_i\ (\mathrm{mod}\ p))[64:])(m) \equiv r$, kur [64:] nurodo, kad kontrolinio parašo funkcija *KH* turi naudoti maišos funkcijos sukurtos santraukos 65 – 128 bitus (atitinkamai jei naudojamas *K* yra 128 bitų).
Šiuo atveju raktas *K* nebus niekur atvaizduojamas.

8.2 Shamiro (k, n) slenksčio grupei skirto šifro schema

8.1. skyriuje pateikta grupei skirto šifro schema puikiai tinka itin svarbios informacijos, kai atskleisti paslaptį gali tik **visi** gavėjai kartu, saugojimui. Tačiau egzistuoja situacijos kai visų paslapties gavėjų dalyvavimas nėra būtinas – pakanka **kvorumo**. Pavyzdžiui banke tam, kad išduoti paskolą, – pasirašyti paskolos suteikimo sutartį skaitmeniniu parašu, – reikia ne mažiau nei trijų valdybos narių pritarimo. Būtent tokią grupei skirto šifro schemą kai norint atskleisti paslaptį D, sudarytą iš n dalių (padalintą tarp n paslapties gavėjų) pakanka žinoti tik k dalių (privalo dalyvauti k paslapties gavėjų) 1979 pasiūlė A. Shamiras ir pavadino ją (k, n) slenksčio schema [Sha79].

Shamiro pasiūlyta schema paremta tuo, kad k-1 eilės daugianario aprašymui reikia k taškų. Taigi, norint padalinti paslaptį S pagal (k, n) slenksčio schemą, parenkamas pakankamai didelis pirminis skaičius p, $p > n$ bei skirtingi x_1, …, $x_n \in \mathbb{F}_p^*$. Šie skaičiai nėra slapti, paslaptis yra skaičius $S \in \mathbb{F}_p^*$. Dalintojas atsitiktinai parenka a_1, ..., a_{k-1} ir sudaro k-1 eilės daugianarį:

$$f(x)=S+a_1x^1 + a_2x^2 + \ldots + a_{k-1}x^{k-1} \in \mathbb{F}_p^*[x].$$

Kiekvienam paslapties gavėjui D_i (i=1 .. n) paskaičiuoja $S_i \equiv f(x_i) \pmod p$ ir įteikia porą (x_i, S_i). Tuomet susirinkę k ir daugiau dalyvių paslaptį S gali atkurti taip:

$$S \equiv \sum_{i=1}^{k} S_{i_j} \prod_{\substack{1\le t\le k \\ t\ne j}} \frac{x_{i_t}}{x_{i_t}-x_{i_j}} \pmod p.$$

Susirinkę k-1 ir mažiau paslapties gavėjų neturės pakankamai duomenų apskaičiuoti paslaptį S.

8.3 Grupei skirto šifro Signcryption schema su slenksčiu

Šioje schemoje naudojami tie patys parametrai pateikti *6 lentelėje*, kaip ir schemoje be slenksčio, tik papildomai kiekvienam paslapties gavėjui suteikiamas unikalus identifikatorius $ID_{B_i} \neq 0$, kuris pateikiamas viešai.

Tada jei paslapties dalintojas A nori nusiųsti pranešimą m gavėjams B_i, kur i=[1, …, n], taip, kad tik k gavėjų, susirinkę kartu galėtų jį perskaityti, jam reikia atlikti šias operacijas:

1. Atsitiktinai pasirinkti x iš [1, ..., q -1],
2. Atsitiktinai parinkti a_1, ..., a_{k-1} ir sudaryti k-1 eilės daugianarį:

$$f(z)=x+a_1z^1 + a_2z^2 + \ldots + a_{k-1}z^{k-1} \in \mathbb{F}_p^*[z].$$

3. Paskaičiuoti $x_i \equiv f(ID_{B_i})$, kurį nusiųsti i-tajam gavėjui,
   Toliau analogiškai kaip schemoje be slenksčio:
4. Paskaičiuoti viešąjį raktą $y=g^x \pmod p$,
5. Paskaičiuoti $K = hash(\ y^x \pmod p)$),
6. Perskelti K pusiau į du raktus K_1 ir K_2,
7. Tada šifruotas pranešimas bus $c = E\ K_1(m)$,
8. Paskaičiuoti $r= KH\ K_2(m)$,
9. Paskaičiuoti
   $s= x / (r + x_A) \pmod q$, jei naudojama SDSS1 schema arba
   $s= x / (1 + x_A \cdot r) \pmod q$, jei naudojama SDSS2 schema (žr. *7 skyrių*),
10. Nusiųsti paslapties gavėjams B Signcryption schemos rezultatą, – šifruotą ir pasirašytą pranešimą, – (c, r, s).

Unsigncryption schemoje ne mažiau nei k gavėjų iš siuntėjo A gavę savo x_i:

11. Randa x pagal formulę:

$$x \equiv \sum_{i=1}^{k} x_{i_j} \prod_{\substack{1 \le t \le k \\ t \ne j}} \frac{ID_{i_t}}{ID_{i_t} - ID_{i_j}}.$$

12. Apskaičiuoja K iš r, s, g, p, y_A ir x:
    $K = (y_A \cdot g^r)^{s \cdot x} \pmod p$ jei naudojama SDSS1 schema arba
    $K = (y_A^r \cdot g)^{s \cdot x} \pmod p$ jei naudojama SDSS2 schema (žr. *7 skyrių*),
13. Perskelia K pusiau į du raktus K_1 ir K_2,
14. Pranešimo dešifravimas $m = D\ K_1(c)$,
15. Patikrina parašą ir priima pranešimą m tik jei $KH\ K_2(m) \equiv r$.

Kaip matome, norint padalinti paslaptį pagal grupei skirto šifro Signcryption schemą su slenksčiu kaip ir paprastoje (be slenksčio) schemoje praktiškai nereikia keisti kriptosistemos algoritmo. Papildomos operacijos yra:

a) *Signcryption* algoritme reikia:
   i. papildomai sudaryti k-1 eilės daugianarį,
   ii. paskaičiuoti ir paslapties gavėjams nusiųsti jų paslapties dalis x_i,
   iii. atlikti papildomą laipsnio kėlimo modulyje operaciją – surasti parametrą y.
b) *Unsigncryption* algoritme susirinkę k ir daugiau paslapties gavėjų prieš dešifruojant pranešimą turi paskaičiuoti Lagranžo interpoliacinį daugianarį – x reikšmę.

Taip pat, kaip ir paprastoje (be slenksčio) schemoje, norint padalinti kitą paslaptį, – nusiųsti kitą pranešimą, – tereikia atlikti visus veiksmus nuo pradžių, t.y. nuo 1 algoritmo punkto.

Pasiūlyta schema taip pat pasižymi labai įdomia savybe – jos algoritme nenaudojami paslapties gavėjų viešieji raktai. Paslapties gavėjų viešuosius raktus galima panaudoti jiems persiunčiant nesaugiais komunikacijos kanalais paslapties dalis x_i, užšifruotas pasirinkta viešojo rakto kriptosistema, kad ir Signcryption.

Pasiūlyta schema taip pat atitinka Shamiro sąlygą: susirinkę k ir daugiau paslapties gavėjų gali nesunkiai apskaičiuoti x ir atskleisti paslaptį, tuo tarpu k-1 ir mažiau paslapties gavėjų nesugebės apskaičiuoti x, tuo pačiu ir atskleisti paslapties. Be to galima patikrinti siuntėjo skaitmeninį parašą – pranešimo tikrumą.

2002 Z. Zhang, M. Cai, J. Qu taip pat pasiūlė grupei skirto šifro Signcryption schemą su slenksčiu, tačiau mano siūloma schema yra pranašesnė, nes reikalauja 2 kėlimo laipsniu modulyje operacijų *Signcryption* metu ir tik 1 kėlimo laipsniu operacijos *Unsigncryption* metu, kai Z. Zhang et al. schemoje reikia 2 kėlimo laipsniu modulyje operacijų *Signcryption* metu ir 2 kėlimo laipsniu modulyje operacijų *kiekvienam gavėjui Unsigncryption* metu [ZCQ02].

8.4 Praktinis patikrinimas

Kaip ir kriptosistemų palyginime praktiniam siūlomų schemų patikrinimui buvo sukurtos realizacijos Python (64-bitų 2.7.2 versija) programavimo kalba, naudojant papildomas bibliotekas pyCrypto (2.5 versija), SciPy (0.10.1 versija) bei NumPy-MKL (1.6.1 versija).

Su kiekviena schema buvo atlikta po 100 bandymų atsitiktiniu būdu parenkant pranešimo ilgį (2048, 10240 arba 51200 baitų) bei rakto dydį (1024, 1536 arba 2048 bitų). Visais atvejais pranešimai buvo sėkmingai užšifruoti bei dešifravus sėkmingai atlikta skaitmeninio parašo patikra.

Schemoje su slenksčiu papildomai buvo atlikta 100 iteracijų bandant dešifruoti pranešimą su mažiau ir daugiau nei *k* pranešimo dalių bei įvairiais *n* (pranešimo gavėjų) ir *k* (reikalingų dešifravimui dalių) skaičiais. Atlikti bandymai patvirtino, kad schema veikia su įvairiais *k* ir *n* skaičiais bei kad su *k*-1 ir mažiau dalių dešifruoti pranešimo nepavyksta (skaitmeninio parašo patikra buvo nesėkminga) su *k* ir daugiau pranešimo dalių pranešimas sėkmingai dešifruojamas (skaitmeninio parašo patikra sėkminga).

Be to atliekant bandymus schemoje su slenksčiu buvo aptiktas papildomas reikalavimas praktinei realizacijai: kadangi algoritme naudojamas *x* yra sveikasis skaičius (long, angl.), o skaičiuojant Lagranžo interpoliacinį daugianarį laisvasis narys yra realusis skaičius (float, angl.), prieš galutinai parenkant atsitiktinį *x* reikia pasitikrinti ar interpoliuojant gaunama reikšmė sutampa su parinktąja. Atlikus 200 iteracijų kiekvieną kartą vidutiniškai reikėjo parinkti 2,1 (5,21%) *x* reikšmių, kad ši sąlyga būtų tenkinama (skliausteliuose pateiktas eksperimentinio standartinio aritmetinio vidurkio nuokrypio santykis su aritmetiniu vidurkiu).

Išvados ir rekomendacijos

Atlikus teorinę apžvalgą bei praktinio patikrinimo rezultatų palyginimą, pagal nustatytus efektyvumo kriterijus:

1. Pranešimo pasirašymo, šifravimo, dešifravimo ir parašo patikrinimo operacijų trukmę,
2. Perduodamos perteklinės informacijos kiekį, t.y. pranešimo ilgio padidėjimą atlikus pasirašymo ir šifravimo operacijas,

nustatyta, kad teorinės prielaidos, kuriomis paremtas Signcryption pranašumas, lyginant su įprastomis viešojo rakto kriptosistemomis, praktikoje pasitvirtina: pagal nustatytus kriterijus Signcryption kriptosistema ženkliai pranoksta kitas dvi nagrinėtas kriptosistemas. Atliktų bandymų rezultatai parodė, kad teorinės vertės: 58% (50% atitinkamai) mažiau skaičiavimo laiko ir 85% (91% atitinkamai) sumažintas šifruoto ir pasirašyto pranešimo ilgio padidėjimas lyginant su StE kriptosistemomis, kurių sudėtingumas pagrįstas diskrečiojo logaritmo radimo uždavinio sudėtingumu (faktorizacijos uždavinio sudėtingumu atitinkamai), ne tik pasitvirtina, bet netgi viršijamos: lyginant kriptosistemas su 1024, 1536, 2048 bitų dydžio raktais ir šifruojant 2048, 10240, 51200 baitų ilgio pranešimus, Signcryption algoritmui vidutiniškai reikia 93,91% (81,99% atitinkamai) mažiau skaičiavimo laiko, jei įskaitomas šifravimo ir pasirašymo raktų radimo laikas, 99,60% (97,77% atitinkamai) mažiau skaičiavimo laiko, jei neįskaitomas šifravimo ir pasirašymo raktų radimo laikas, ir gaunamas 99,59% (93,46% atitinkamai) sumažintas šifruoto ir pasirašyto pranešimo ilgio padidėjimas lyginant su StE kriptosistemomis, kurių sudėtingumas pagrįstas diskrečiojo logaritmo radimo uždavinio sudėtingumu, – ElGamalio, – (faktorizacijos uždavinio sudėtingumu atitinkamai, – RSA).

Nepaisant efektyvumo, Signcryption kriptosistema turi mažesnes pritaikymo galimybes nei kitos apžvelgtos kriptosistemos, nes skirtingai nei ElGamalio ar RSA kriptosistemos, Signcryption negali būti naudojama tik šifravimui arba tik pasirašymui, todėl ją taikyti galima tik ten, kur reikalingas ir pranešimo slaptumas (šifravimas), ir pranešimo autentiškumo bei vientisumo patvirtinimas (skaitmeninis parašas).

Skaičiavimo trukmė parodo algoritmo poreikį skaičiuojamajai galiai bei energijai, todėl Signcryption puikiai tinka nedidelės skaičiavimo galios, mažas energijos atsargas turintiems įrenginiams.

Itin mažas pranešimo ilgio padidėjimas gali būti lemiamu pranašumu kai yra ribotos informacijos perdavimo galimybės arba informacijos perdavimas yra brangus.

Nežiūrint į ženklų pranašumą tiek skaičiavimo greičio, tiek perteklinės informacijos kiekio aspektais, Signcryption saugumas yra ne mažesnis nei kitų apžvelgtų kriptosistemų, nes bandomoji realizacija sukurta modifikuotos ElGamalio skaitmeninio parašo schemos pagrindu bei naudoja to paties dydžio raktus, o šifravimui naudojamas AES-128 algoritmas, kurio saugumo lygis yra 128 bitai, kas prilyginama RSA kriptosistemos saugumui naudojant 3072 bitų dydžio raktą [Bar07].

Apibendrinant galima daryti išvadą, kad Signcryption yra aukšto saugumo lygio kriptosistema, idealiai tinkanti kai reikia pranešimo slaptumo (šifravimo), ir pranešimo autentiškumo bei vientisumo patvirtinimo (skaitmeninio parašo), esant mažiems skaičiavimo ir energijos ištekliams bei ribotam ir(arba) brangiam informacijos perdavimui, pavyzdžiui M–verslui (M-Commerce, angl.). Be to pateiktos grupei skirto šifro Signcryption schemos įrodo, kad Signcryption, išlaikant efektyvumą bei slaptumą ir autentiškumą, gali būti nesudėtingai pritaikoma kitose kriptografinėse schemose.

Literatūros sąrašas

[Bar07] E. Barker, „NIST Special Publication 800-57, Recommendation for Key Management - Part 1: General (Revised)“, National Institute of Standards and Technology, 2007.

[DH76] W. Diffie, M. E. Hellman, „New Directions in Cryptography“, IEEE Transactions on Information Theory. Nr. 22(6), 1976, pp. 644 – 654.

[DZ10] A. W. Dent, Y. Zheng. „Practical signcryption“, Springer – Verlag, Berlin Heidelberg, 2010.

[Elg85] T. ElGamal, „A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms“, IEEE Transactions on Information Theory. Nr. 31(4), 1985, pp. 469 – 472.

[FS05] Н. Фергюсон, Б. Шнайер, „Практическая криптография“, Вильямс, Москва, 2005.

[Fips180] „Secure Hash Standard (SHS). FIPS PUB 180-3“, National Institute of Standards and Technology, 2008.

[Fips197] „Announcing the ADVANCED ENCRYPTION STANDARD (AES). FIPS PUB 197”, National Institute of Standards and Technology, 2001.

[Rfc2104] HMAC: „Keyed-Hashing for Message Authentication“, IETF Tools, 1997.

[RSA78] R. Rivest, A. Shamir, L. Adleman, „A Method for Obtaining Digital Signatures and Public – Key Cryptosystems“, Communications of the ACM 21 (2), 1978, pp. 120 – 126

[Sha79] A. Shamir, „How to Share a Secret“. Communications of the ACM 22 (11), 1979, pp. 612 – 613.

[Sta07] V. Stakėnas, „Kodai ir šifrai“, Vilnius, 2007.

[ZCQ02] Z. Zhang, M. Cai, J. Qu, „Signcryption Scheme with Threshold Shared Unsigncryption Preventing Malicious Receivers“, Proceedings of 2002 IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering (TENCON '02), Vol. 1, 2002, pp. 196 – 199.

[Zhe97] Y. Zheng. „Digital Signcryption or How to Achieve Cost (Signature & Encryption) << Cost (Signature) + Cost (Encryption)“, Advances in Cryptology – Crypto'97, Lecture Notes in Computer Science, Vol. 1294, Springer – Verlag, Berlin Heidelberg, 1997, pp. 165 – 179. Pilna versija – URL: http://coitweb.uncc.edu/~yzheng/publications/files/signcrypt.pdf. 484 KB.

[Zhe98] Y. Zheng. Shortened „Digital Signature, Signcryption and Compact and Unforgeable Key Agreement Schemes“, a submission to IEEE P1363 Standard Specifications for Public Key Cryptography, 1998.