
LATVIJAS UNIVERSITĀTE
ANNO 1919

LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

# ELIPTISKO LĪKŅU KRIPTOSISTĒMU PIELIETOJUMI

BAKALAURA DARBS

Autors: **Krists Magons**
Studenta apliecības Nr.: km10054
Darba vadītājs: profesors Dr. dat. Andris Ambainis

RĪGA 2016

# ANOTĀCIJA

IT nozares attīstība un daudzpusīgie interneta lietojumi prasa drošākus un efektīvākus datu aizsardzības risinājumus.

Darbā aprakstīta personalizētas, publiskās atslēgas kriptogrāfiskas sistēmas izstrāde, kas balstīta uz *Veierštrāsa* eliptiskajām līknēm pār galīgu pirmskaitļu lauku. Šādu kriptogrāfisku sistēmu fundamentālā drošība saistās ar algoritmiski sarežģīto diskrētā logaritma problēmu eliptiskajām līknēm.

Eliptisko līkņu bāzētas kriptogrāfiskas sistēmas, salīdzinājumā ar populāro RSA sistēmu, nodrošina augstāku drošības līmeni pie vienāda atslēgu garuma, tādējādi iespējami efektīvi datu aizsardzības risinājumi ar labāku veiktspēju.

Darbā tiek skaidroti vispārīgi eliptisko līkņu kriptosistēmu matemātiskie un datorzinātņu pamati, kas nodrošina praktiskus šādu sistēmu pielietojumus datu aizsardzībā un kriptogrāfijā.

**Atslēgas vārdi**: eliptiskās līknes, eliptisko līkņu kriptogrāfija, diskrētā logaritma problēma, kriptogrāfiskas sistēmas

# ABSTRACT

**Applications of elliptic curve cryptosystems**

Rapid development of the IT industry and growing use of Internet demand more efficient and reliable data protection solutions.

Within the framework of this research paper a personalized public-key cryptographic system design is proposed which is based on the mathematical properties of *Weierstrass* elliptic curves over finite prime fields. The fundamental security of such systems is based on the algorithmically hard discrete logarithm problem for elliptic curves.

Elliptic curve based cryptographic systems, compared to the popular RSA system provide a higher level of security in terms of key bit-length, so more efficient data protection solutions with better performance could be introduced.

The mathematical and computer science foundations of elliptic curve based cryptosystems are studied for practical applications of such systems.



**Key words**: elliptical curves, elliptic curve cryptography, discrete logarithm problem, cryptographic systems

# SATURA RĀDĪTĀJS

# APZĪMĒJUMU SARAKSTS

**ASCII** - *American Standard Code for Information Interchange*, kodu tabula.

**Asimetriskā kriptosistēma** - Kriptogrāfiska sistēma, kur katra operācija tiek kontrolēta ar vienu no divām atslēgām – privāto atslēgu vai publisko atslēgu. Neeksistē tāds algoritms, kas no publiskās atslēgas atrod privāto atslēgu polinomiālā laikā.

**Atslēga** - Parametrs, kas nosaka kriptogrāfiskas operācijas izpildi.

**DLPEL** - Diskrētā logaritma problēma eliptiskajām līknēm.

**DSA** - *Digital Signature Algorithm*, ciparu paraksta algoritmu standarts.

**ECDH** - *Elliptic Curve Diffie-Hellman*, atslēgu apmaiņas protokols.

**ECDSA** - *Elliptic Curve Digital Signature Algorithm*, ciparu paraksta algoritmu standarts eliptisko līkņu kriptosistēmām.

**NIST** - (ASV) *National Institute of Standards and Technology*.

**Privātā atslēga** - Publiskās atslēgas kriptosistēmu atslēga, kas nepieciešama informācijas atšifrēšanai un tiek turēta slepenībā.

**Publiskā atslēga** - Publiskās atslēgas kriptosistēmu atslēga, kas nepieciešama informācijas šifrēšanai un tiek brīvi izplatīta nedrošās saziņas vidēs.

**REST** - *Representational State Transfer*, tīmekļa programmatūras arhitektūras koncepts.

**RSA** - *(Rivest, Shamir, Adleman)* Viena no pasaulē populārākajām publiskās atslēgas kriptosistēmām.

**Sarežģīta skaitļošanas problēma** - Neeksistē algoritms, kurš visiem problēmas gadījumiem polinomiālā laikā atrod risinājumu.

**Simetriskā kriptosistēma** - Kriptogrāfiska sistēma, kur katra operācija tiek kontrolēta ar vienu atslēgu.

**Viegls skaitļošanas uzdevums** - Eksistē algoritms, kurš visiem uzdevuma gadījumiem polinomiālā laikā atrod risinājumu.

$\mathbb{F}_p$ - Galīgs pirmskaitļu lauks, ko nosaka pirmskaitlis $p$.

$\mathbf{E}(\mathbb{F}_p)$ - Eliptiskā līkne pār galīgu pirmskaitļu lauku.

$\#\mathbf{E}(\mathbb{F}_p)$ - Eliptiskās līknes pār galīgu pirmskaitļu lauku punktu kopas veidotās aditīvās grupas pakāpe.

$\langle \mathbf{G}, \circ \rangle$ - Eliptiskās līknes pār galīgu pirmskaitļu lauku punkta $G$ veidotā cikliskā apakšgrupa.

# IEVADS

Ievadā tiek izklāstīts vispārīgs situācijas apraksts, kas pamato darba mērķu aktualitāti.

Mūsdienu komunikācija globālajā tīmeklī fundamentāli nav iedomājama bez datu aizsardzības un kriptogrāfijas. Teju visas ikdienišķas operācijas, ko attālināti veicam internetā, tādas kā sensitīvu datu pārraide, autentifikācija un autorizācija dažādās sistēmās, datu autentiskuma pārbaudes, interneta banku lietošana, tiešsaistes līgumi utt., saistās ar dažādu kriptogrāfisku metožu pielietošanu datu aizsardzībai.

Tā kā internetam nav centralizētas pārvaldes un tā ekspluatācijas likumdošana ir specifiska konkrētā jurisdikcijā, katrs lietotājs atbild par savu datu aizsardzību internetā. Viena no iespējām, kā internetā pasargāt sensitīvu informāciju, ir lietot personalizētas kriptogrāfiskas sistēmas.

Uz jebkuru kriptogrāfisku sistēmu varam skatīties divējādi. No vienas puses saziņas dalībnieki vēlas izmantot publisku, nedrošu, bet ātru informācijas pārraides kanālu, lai efektīvi apmainītos ar konfidenciāliem datiem. Savukārt no otras puses ir kriptoanalītiķi, kas, iespējams, vēlas pārtvert saziņu un atklāt vai izmainīt tās laikā pārsūtīto informāciju [1]. Vienas vai otras minētās puses mērķu sasniegšana lielā mērā saistās ar pielietoto kriptogrāfisko sistēmu un pieejamajiem skaitļošanas resursiem. Skaidrs, ka saziņas dalībniekiem, uzlabojot kriptosistēmas drošību, attiecībā pret tās uzlaušanai nepieciešamajiem resursiem, kriptoanalītiķi veiks uzlabojumus metodoloģijā, kas vairāk vai mazāk efektīvi nodrošina konkrētās kriptogrāfiskās sistēmas atslēgu atrašanu.

Būtiski ievadā pieminēt, ka interneta lietošana saistās ar aizvien kompaktākām un energoefektīvām ierīcēm, kas uzliek papildu prasības pielietoto kriptogrāfisko sistēmu algoritmu veiktspējai, attiecībā uz pamatoperācijām, tādām kā atslēgu ģenerēšana un to verifikācija.

Tātad, kriptogrāfija nepārtraukti attīstās un prasa arvien sarežģītākus un efektīvākus risinājumus datu aizsardzībai. Šī darba ietvaros autors projektē personalizētu, praktiskai datu aizsardzībai pielietojamu kriptogrāfisku sistēmu, kas balstīta uz eliptisko līkņu pār galīgiem laukiem matemātiskajām īpašībām.

## Darba mērķis

Darba mērķis ir izstrādāt personalizētu eliptisko līkņu bāzētu kriptogrāfisku sistēmu.

Darbs ir tiešs turpinājums autora 2015. gada pavasarī izstrādātajam un aizstāvētajam kursa darbam ar nosaukumu "*Eliptisko līkņu kriptogrāfija*" un autora publikācijai konferencē *SOFSEM Student Reaserch Formum* ar nosaukumu "*Applications and Benefits of Elliptic Curve*

*Cryptografy*" [2], kur autors pēta vispārīgu asimetriskās kriptogrāfijas teoriju, eliptisko līkņu matemātiskos pamatus un salīdzina eliptisko līkņu bāzētas kriptosistēmas ar informācijas tehnoloģiju nozares *de facto* standartu RSA [2]. Minēto darbu zinātniskais vadītājs - profesors Rūsiņš Mārtiņš Freivalds.

Nosauktā darba mērķa sasniegšanai autors izvirza sekojošus uzdevumus:

- Turpināt eliptisko līkņu matemātisko īpašību un metožu izpēti
- Turpināt eliptisko līkņu bāzētu kriptosistēmu pamatalgoritmu izpēti
- Projektēt personalizētu eliptisko līkņu bāzētu kriptogrāfisku sistēmu
- Konstruēt projektētās sistēmas references implementāciju
- Eksperimentāli novērtēt references implementācijas kriptogrāfisko pamatalgoritmu veiktspēju konkrētā aparatūras un programmatūras kontekstā.

Kriptogrāfiskās sistēmas references implementācija tiek realizēta, sadarbojoties ar uzņēmuma *Exigen Services Latvia* (ESL) praktikantu programmu un tiek plānots, ka ESL praktikants un LU Datorikas fakultātes students Oļegs Pešudovs to aizstāvēs sava kvalifikācijas darba ietvaros. Uzņēmums ESL nodrošina visus tehniskos resursus, kas saistīti ar references implementācijas realizāciju, izvietošanu un uzturēšanu.

### Darba struktūra

Darbs strukturēts sešās pamata sadaļās, proti: *Ievads*, *Eliptisko līkņu bāzētu kriptogrāfisko sistēmu fundamentālie matemātiskie pamati*, *Eliptisko līkņu kriptogrāfiskās sistēmas*, *Personalizētas eliptisko līkņu kriptosistēmas izstrāde*, *Rezultāti* un *Secinājumi.*

Ievadā autors neformāli izklāsta situāciju un darba nepieciešamību, kā arī definē sasniedzamos mērķus. Sadaļā *Eliptisko līkņu bāzētu kriptogrāfisko sistēmu fundamentālie matemātiskie pamati* autors pēta eliptisko līkņu kriptogrāfisko sistēmu matemātiskos pamatus un metodes. Sadaļā *Eliptisko līkņu kriptogrāfiskās sistēmas* autors apraksta eliptisko līkņu kriptosistēmu fundamentālo drošību, pamatalgoritmus un vispārīgu salīdzinājumu ar RSA kriptosistēmu. Nodaļā *Personalizētas eliptisko līkņu kriptosistēmas izstrāde* autors izklāsta personalizētas eliptisko līkņu bāzētas kriptosistēmas implementāciju. Darbs tiek nobeigts ar īsu svarīgāko rezultātu izklāstu sadaļā *Rezultāti* un secinājumiem.

# 1. ELIPTISKO LĪKŅU BĀZĒTU KRIPTOGRĀFISKO SISTĒMU FUNDAMENTĀLIE MATEMĀTISKIE PAMATI

Jebkuru kriptogrāfisku sistēmu drošība fundamentāli tiek realizēta, izmantojot matemātiskās metodes. Šajā nodaļā tiek aplūkoti eliptisko līkņu bāzētu kriptogrāfisku sistēmu matemātiskie pamati – tiek skaidrots eliptisko līkņu jēdziens, to īpašības pār galīgiem pirmskaitļu laukiem un definētas matemātiskas operācijas, kas nodrošina attiecīgo kriptogrāfisko sistēmu pamatalgoritmu efektīvas realizācijas iespējas.

Strukturāli nodaļa sastāv no četrām sadaļām. Iesākumā tiek izklāstīta kubisko līkņu definīcija, singularitātes nozīme, tālāk tiek definēts eliptiskās līknes jēdziens un aprakstītas nozīmīgākās matemātiskās eliptisko līkņu īpašības.

## 1.1. Kubiskās līknes

Ja $K$ ir lauks un koeficienti $a, b \in K$, tad par kubisku līkni $K \times K$ veidotajā plaknē pār lauku $K$ sauksim sekojošu kortežu kopu [3]:

$$C = \{(x, y) | y^2 = x^3 + ax + b\} \subseteq K^2$$

Vienādojumu $y^2 = x^3 + ax + b$ sauc par kubisko *Veierštrāsa* vienādojumu [3].

Kubisko līkņu fundamentālās īpašības nosaka 3.pakāpes polinoms $p(x) = x^3 + ax + bx$ [3]. Saskaņā ar algebras pamata teorēmu, nosauktajam polinomam eksistē 3 saknes $\lambda_1, \lambda_2, \lambda_3$.

Apskata lauku $K', kuram\ K \subseteq K' un\ \{\lambda_1, \lambda_2, \lambda_3\} \subseteq K'$. Polinomu $p(x)$ pār lauku $K'$ iespējams faktorizēt formā $p(x) = (x - \lambda_1)(x - \lambda_2)(x - \lambda_3)$. Viegli redzēt, ka, attiecībā uz polinoma $p(x)$ saknēm $\lambda_1 .. \lambda_3$ , iespējami trīs dažādi gadījumi:

a) $\forall i, j \in \{1,2,3\}: \lambda_i \neq \lambda_j$ - visas $p(x)$ saknes ir atšķirīgas. Šādas kubiskas līknes sauc par gludām kubiskām līknēm (*smooth curve*) [3].

b) $\exists i, j \in \{1,2,3\}: \lambda_i = \lambda_j$ - eksistē divas $p(x)$ identiskas saknes. Šādas kubiskas līknes sauc par mezglotām kubiskām līknēm (*nodal curve*) [3].

c) $\forall i, j \in \{1,2,3\}: \lambda_i = \lambda_j$ - visas $p(x)$ saknes ir identiskas. Šādas kubiskas līknes sauc par smailām kubiskām līknēm (*cuspidal curve*) [3].

Aplūkosim kubisko līkņu piemērus pār lauku $K' = \mathbb{R}$:

Piemērs *gludai* kubiskai līknei pār lauku $\mathbb{R}$:

No reālo skaitļu lauka patvaļīgi izvēlamies dažādas $\lambda_1 .. \lambda_3$ vērtības : $\lambda_1 = 1, \lambda_2 = -\frac{1}{2} - \frac{\sqrt{5}}{2}\ \lambda_3 = \frac{\sqrt{5}}{2} - \frac{1}{2}$, tātad $p(x) = (x - 1)(x - (-\frac{1}{2} - \frac{\sqrt{5}}{2}))(x - (\frac{\sqrt{5}}{2} - \frac{1}{2}))$ un atbilstošais

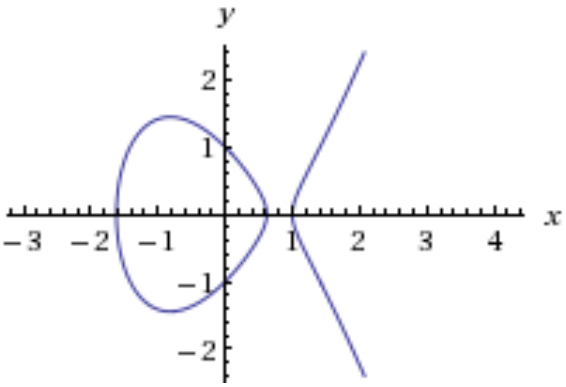
kubiskais *Veierštrāsa* vienādojums $y^2 = x^3 - 2x + 1$. Iegūtā *gluda* kubiska līkne redzama attēlā 1.1.1.

Piemērs *mezglotai* kubiskai līknei pār lauku $\mathbb{R}$:

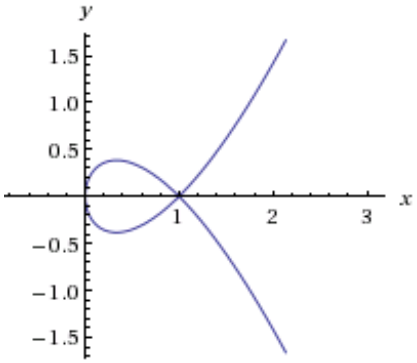
No reālo skaitļu lauka patvaļīgi izvēlamies divus skaitļus 0 un 1, definē $\lambda_1..\lambda_3$ vērtības šādā veidā: $\lambda_1 = 0, \lambda_2 = 1, \lambda_3 = 1$. $p(x) = (x-0)(x-1)(x-1) = x(x-1)^2$, atbilstošais kubiskais *Veierštrāsa* vienādojums $y^2 = x^3 - 2x^2 + x$. Iegūtā *mezglota* kubiska līkne redzama attēlā 1.1.2. Ievēro *mezglotas* kubiskās līknes veidoto raksturīgo grafika *mezglu* punktā (1,0)

Piemērs *smailai* kubiskai līknei pār lauku $\mathbb{R}$:

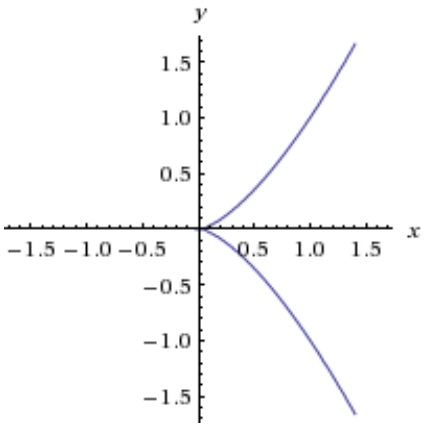
Definē $\lambda_1..\lambda_3$ vērtības šādā veidā: $\lambda_1 = 0, \lambda_2 = 0, \lambda_3 = 0$. $p(x) = (x-0)(x-0)(x-0) = x^3$, atbilstošais kubiskais *Veierštrāsa* vienādojums $y^2 = x^3$**.** Iegūta *smaila* kubiska līkne redzama attēlā 1.1.3. Ievēro *smailas* kubiskās līknes veidoto raksturīgo grafika *smaili* punktā (0,0).




*att. 1.1.1* ***Gluda kubiska līkne*** $y^2 = x^3 - 2x + 1$




*att. 1.1.2* ***Mezglota kubiska līkne*** $y^2 = x^3 - 2x^2 + x$




*att. 1.1.3* ***Smaila kubiska līkne*** $y^2 = x^3$

Redzams, ka visas kubiskās līknes pār reālo skaitļu lauku ir simetriskas attiecībā pret x-asi, jo, saskaņā ar *Veierštrāsa* vienādojumu, $y^2 = x^3 - 2x^2 + x \rightarrow y = \pm\sqrt{x^3 - 2x + 1}$

## 1.2. Kubisko līkņu singularitāte

Aplūkosim kubiskās līknes polinoma $p(x) = x^3 + ax + b$ diskriminantu reālo skaitļu laukā. Vispārīgajā gadījumā diskriminantu $\Delta$ 3.pakāpes, polinomam $g(x)$ definē šādā veidā [4]: Ja $g(x) = (x - \lambda_1)(x - \lambda_2)(x - \lambda_3) = x^3 - a_1x^2 + a_2x - a_3$ , kur $\lambda_i - g(x)$ sakne, tad

$$\Delta_{g(x)} = \prod_{i<j} (\lambda_i - \lambda_j)^2 = (\lambda_1 - \lambda_2)^2(\lambda_1 - \lambda_3)^2(\lambda_2 - \lambda_3)^2$$

$$= a_2^2(a_1^2 - 4a_2) + a_3(-4a_1^3 + 18a_1a_2 - 27a_3)$$

Tātad, $p(x)$ diskriminants $\Delta_{p(x)} = -4a_2^3 - 27a_3^2$, jo $a_1 = 0$

Vispārīgajā gadījumā iespējamas sekojošas kubiska polinoma $g(x)$ diskrimina $\Delta_{g(x)}$vērtības [5]:

- $\Delta_{g(x)} > 0$ Polinoma $g(x)$ saknes $\lambda_1 \dots \lambda_3$ ir savstarpēji atšķirīgas
- $\Delta_{g(x)} = 0$ : Polinoma $g(x)$ saknes $\lambda_1 \dots \lambda_3$ ir identiskas vai eksistē divas savstarpēji identiskas saknes $\lambda_i = \lambda_j$
- $\Delta_{g(x)} < 0$ : Viena no polinoma $g(x)$ saknēm $\lambda_i \in \mathbb{R}$, pārējās divas kompleksās saknes ir kompleksi saistīti skaitļi.

Pēc definētās formulas aprēķināsim diskriminantus polinomiem, kuri veido iepriekš aplūkotās kubiskās līknes pār reālo skaitļu lauku: $\Delta_{x^3-2x+1} = 5$, $\Delta_{x^3-2x^2+x} = \Delta_{y^2=x^3} = 0$. Redzam, ka *mezglotās* kubiskās līknes un *smailās* kubiskās līknes polinomu diskriminantu vērtības ir 0.

Ja kubiskās līknes $C$ 3.pakāpes polinoma $p(x)$ diskriminants $\Delta_{p(x)} = 0$, tad $C$ – singulāra kubiska līkne [6]. Autors secina, ka visas *mezglotās* kubiskās līknes un *smailās* kubiskās līknes ir singulāras kubiskās līknes, savukārt *gludās* kubiskās līknes ir nesingulāras kubiskās līknes.

Ja kubisku līkni C izsakām kā divu argumentu funkciju no līknes mainīgajām vērtībām $x$ un $y$ : $f(x, y) = 0$, kas ir atvasināma visā tās definīcijas apgabalā, tad C ir singulāra, ja eksistē $(x, y) \in dom\, f$ un atbilstošie parciālie atvasinājumi singulārajā punktā $(x, y)$ : $f'_x = f'_y = 0$, pretējā gadījumā C ir nesingulāra līkne [6].

Iepriekš aplūkotajos singulāro kubisko līkņu piemēros, sekojoši punkti ir singulāri: *mezglotai* līknei $y^2 = x^3 - 2x^2 + x$ punkts (1,0) un *smailai* līknei $y^2 = x^3$ punkts (0, 0). No attēliem redzam, ka šajos punktos nav iespējams definēt pieskares līniju[1].

[1] Taisni $y = kx + b$ sauc par līknes $f(x)$ pieskares līniju punktā $x_0$, ja taisne iet caur punktu $(x_0, f(x_0))$ un tās virziena koeficients $k = f'(x_0)$

Eliptisko līkņu bāzētu kriptogrāfisku sistēmu izstrādē praktiska nozīme ir nesingulārām jeb gludām kubiskām līknēm. Singulāru kubisku līkņu izmantošana kriptosistēmās rada būtiskus drošības riskus.

Būtisks fakts. Patvaļīgas kubiskas līknes $y^2 = x^3 + ax + b$ ne-singularitātes nepieciešamais un pietiekamais nosacījums: $4a^3 + 27b^2 \neq 0$ [7].

## 1.3. Eliptiskās līknes Veierštrāsa formā

Par eliptisko līkni Veierštrāsa formā $E$ pār lauku $K$ sauksim nesingulāras kubiskas līknes $C$ plaknē $K \times K$ veidoto punktu kopu, kurai pievienots specifisks elements $\mathcal{O}$ [7], [8]:

$$C = \{(x, y) | y^2 = x^3 + ax + b \wedge a, b \in K \wedge 4a^3 + 27b^2 \neq 0 \} \subseteq K^2$$

$$E = C \cup \{\mathcal{O}\}$$

Tātad no definīcijas secinām, ka eliptiskajām līknēm piemīt visas nesingulāru kubisko līkņu ģeometriskās īpašības.

Eliptisko līkņu definīcijā iekļautais elements $\mathcal{O}$ nodrošina, ka E kopas elementi attiecībā pret saskaitīšanas operāciju veido Ābela grupu, kurā elements $\mathcal{O}$ ir identitātes elements. $\forall P = (P_x, P_y) \in E: P + \mathcal{O} = \mathcal{O} + P = P$ [7].

Saturiski $\mathcal{O}$ apzīmē *punktu pie bezgalības*, kurā, projektīvās koordinātu sistēmās, krustojas divas paralēlas taisnes.

## 1.4. Eliptiskās līknes pār galīgu pirmskaitļu lauku $\mathbb{F}_p$

Galīgs lauks sastāv no galīga skaita lauka elementiem un tiem definētām binārām saskaitīšanas un reizināšanas operācijām, kas atbilst lauka aksiomām [9].

Ja p ir nepāra pirmskaitlis, tad pierādīts, ka veselie skaitļi pēc moduļa p veido lauku. $\mathbb{F}_p$- nepāra pirmskaitļa $p$ veidotais lauks ar pakāpi $\#\mathbb{F}_p = p$. Katram p eksistē tieši viens lauks $\mathbb{F}_p$, tomēr tā elementi var tikt reprezentēti dažādi [9]. Šajā darbā par $\mathbb{F}_p$ elementiem sauksim veselos skaitļus pēc moduļa p, tātad $\mathbb{F}_p = \mathbb{Z}_p = \{0,1,2, \dots, p-1\}$.

- $\mathbb{F}_p$ elementu saskaitīšanas operācijas definīcija: $\forall a, b \in \mathbb{F}_p : a + b \equiv c \ (mod\ p) \wedge c \in \mathbb{F}_p$
- $\mathbb{F}_p$ elementu reizināšanas operācijas definīcija: $\forall a, b \in \mathbb{F}_p : ab \equiv c \ (mod\ p) \wedge c \in \mathbb{F}_p$

Saskaņā ar lauka aksiomām, katram lauka elementam nepieciešams uzrādīt unikālus tā inversos elementus attiecībā pret saskaitīšanu un reizināšanu.

$\mathbb{F}_p$ elementu aditīvi inversos elementus atrod risinot kongruenci:

$\forall a \in \mathbb{F}_p : a + (-a) \equiv 0 \ (mod\ p)$, kur $-a \in \mathbb{F}_p$ – unikāls elementa $a$ aditīvi inversais elements.

$\mathbb{F}_p$ elementu multiplikatīvi inversos elementus atrod risinot kongruenci:

$\forall a \in \mathbb{F}_p : aa^{-1} \equiv 1\ (mod\ p)$, kur $a^{-1} \in \mathbb{F}_p$ – unikāls elementa $a$ multiplikatīvi inversais elements.

Ja $E$ ir eliptiska līkne un $\mathbb{F}_p$ ir nepāra pirmskaitļa $p$ veidots galīgs lauks, tad par eliptisko līkni pār galīgu pirmskaitļu laiku $E(\mathbb{F}_p)$ sauksim sekojošu kopu:

$$E(\mathbb{F}_p) = \{(x,y) \mid y^2 \equiv x^3 + ax + b\ (mod\ p) \wedge (x,y) \in \mathbb{F}_p^2\ , a,b \in \mathbb{F}_p \wedge 4a^3 + 27b^2 \not\equiv 0\ (mod\ p)\} \cup \{\mathcal{O}\}$$

kur $\mathcal{O}$ apzīmē projektīvo punktu pie bezgalības.

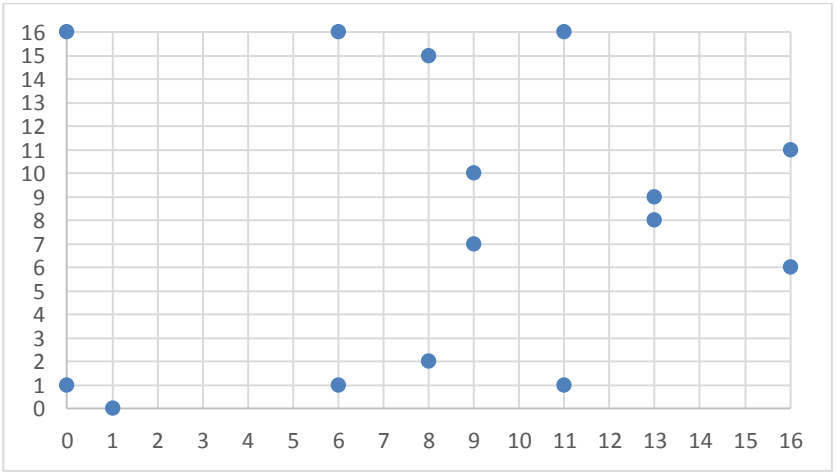
$E(\mathbb{F}_p)$ kopas elementus, $(x,y) \in \mathbb{F}_p^2$ pārīšus un elementu $\mathcal{O}$ sauc par eliptiskās līknes pār galīgu pirmskaitļu lauku punktiem.

### 1.4.1. $E(\mathbb{F}_p)$ kopas elementu aditīvā grupa

No definīcijas viegli redzēt, ka $E(\mathbb{F}_p)$ raksturo galīga, diskrētu punktu kopa. Aplūkosim jau vairākkārt pieminēto *gludo* kubisko līkni $y^2 = x^3 - 2x + 1$ (attēls 1.1.1). Jau tika parādīts, ka līkne ir nesingulāra, tātad atbilst eliptiskās līknes kritērijiem. Ērtai demonstrācijai izvēlas nelielu pirmskaitli p = 17 un definē lauku $\mathbb{F}_{17} = \{0,1,2,\dots,16\}, \#\mathbb{F}_{17} = 17$, iegūstam sekojošu eliptiskās līknes definīciju pār lauku $\mathbb{F}_{17}$:

$$y^2 = x^3 - 2x + 1(\mathbb{F}_{17}) = \{(x,y) | y^2 \equiv x^3 - 2x + 1\ (mod\ 17) \wedge (x,y) \in \mathbb{F}_{17}^2\} \cup \{\mathcal{O}\}$$

Risinot kongruenci $y^2 \equiv x^3 - 2x + 1\ (mod\ 17)$, iegūst 15 dažādus $\mathbb{F}_{17}^2$ pārīšus, kas atbilst $E(\mathbb{F}_p)$ punktiem $\mathbb{F}_{17} \times \mathbb{F}_{17}$ veidotajā plaknē pār lauku $\mathbb{F}_{17}$. Kongruences atrisinājumi redzami attēlā 1.4.1




*att. 1.4.1* ***Eliptiskās līknes $y^2 = x^3 - 2x + 1(\mathbb{F}_{17})$ plaknes punkti.***

Nozīmīgs fakts: $E(\mathbb{F}_p)$ kopas elementi veido komutatīvu (Ābela) grupu, attiecībā pret saskaitīšanas operāciju, ar identitātes elementu $\mathcal{O}$. Lai gan minētā fakta pamatojumi un

pierādījumi plaši pieejami literatūrā, autors iesaka interesentiem iepazīties ar J.S. Milne grāmatā "*Elliptic Curves*" [6], aprakstīto pierādījumu.

Eliptisko līkņu kriptogrāfisku sistēmu konstruēšanā nozīmīgs parametrs ir kopas $E(\mathbb{F}_p)$ punktu skaits jeb $E(\mathbb{F}_p)$ aditīvās grupas pakāpe, kuru apzīmēsim $\#E(\mathbb{F}_p)$. Minētais parametrs raksturo konkrētas kriptogrāfiskas sistēmas fundamentālo drošību, attiecībā uz nepieciešamo laiku diskrētā logaritma problēmas atrisināšanai. Tālāk aplūkosim pieejamos risinājumus, kā novērtēt $\#E(\mathbb{F}_p)$.

1. Triviāls secinājums. $E(\mathbb{F}_p)$ punktu skaits $\#E(\mathbb{F}_p) \leq 2p + 1$ [10]
2. *Hasses* nevienādība. $p + 1 - 2\sqrt{p} \leq \#E(\mathbb{F}_p) \leq p + 1 + 2\sqrt{p}$ [9]
3. *Frobeniusa* (*Frobenius*) endomorfisms $\varphi_p$ [11]

$\varphi_p: E(\overline{\mathbb{F}}_p) \to E(\overline{\mathbb{F}}_p), \varphi_p(x, y) = (x^p, y^p), \varphi_p(\mathcal{O}) = (\mathcal{O})$, kur $E(\overline{\mathbb{F}}_p)$ eliptiskā līkne pār lauka $\mathbb{F}_p$ algebrisko paplašinājumu $\overline{\mathbb{F}}_p$.[2]

*Frobeniusa* endomorfismam $\varphi_p$ piemīt būtiska īpašība [11], [12]:

$$\varphi_p^2 - t\varphi_p + p = 0, kur\ t \in \mathbb{Z}$$

Risinot šo vienādojumu, iespējams atrast $\#E(\mathbb{F}_p)$:

$$\#E(\mathbb{F}_p) = p + 1 - t, t \in \mathbb{Z}$$

Saskaņā ar *Hasses* nevienādību: $|t| \leq 2\sqrt{p}$

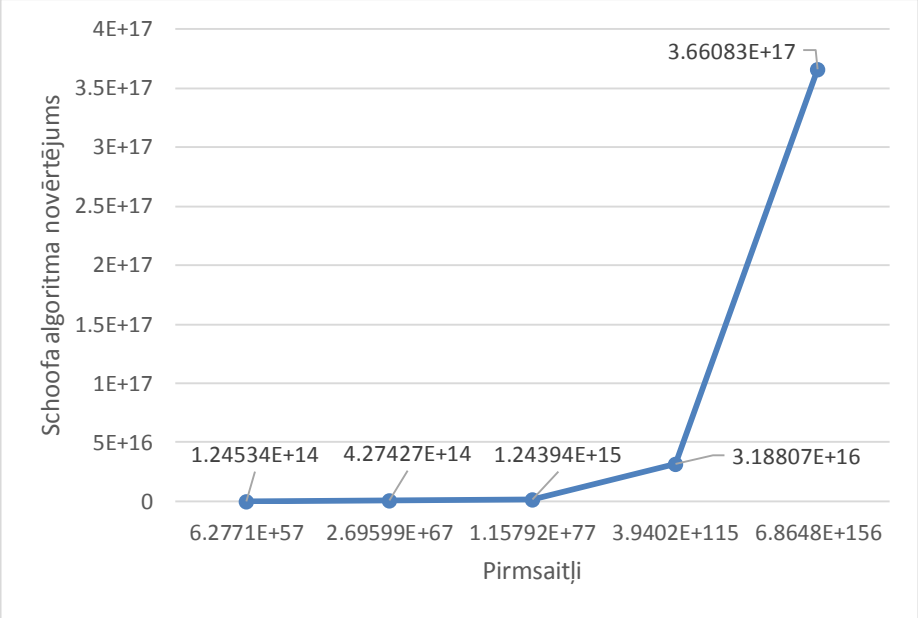
$$\forall P = (x, y) \in E(\overline{\mathbb{F}}_p): (x^{p^2}, y^{p^2}) + t(x^p, y^p) + p(x, y) = 0:$$

Algoritmiski $\#E(\mathbb{F}_p)$ iespējams noteikt, izmantojot *Hasses* nevienādību un veicot pilno pārlasi, tomēr eksistē efektīvāks paņēmiens – 1985. gadā *Schoofa* (*Schoof*) publicētais algoritms, kurš izmanto gan *Hasses* nevienādību, gan *Frobeniusa* endomorfisma īpašības un *Ķīniešu atlikumu* teorēmu. *Schoofa* algoritms strādā $O(\log^8 p)$ laikā [12] un ir viens no nozīmīgākajiem eliptisko līkņu kriptogrāfisko sistēmu fundamentālajiem algoritmiem, kas tiek lietots kriptogrāfiskās sistēmas domēna parametru ģenerēšanā un verifikācijā. Ar *Schoofa* algoritma matemātisko pamatojumu iespējams iepazīties literatūras avotā [12], tomēr praktiskai lietošanai autors iesaka iepazīties ar Schoofa algoritma pseidokodu un implementāciju valodā *Wolfram Language*, kas aprakstīta literatūras avotā [11].

Neskatoties, ka *Schoofa* algoritms strādā poli-logaritmiskā laikā, praktiska $\#E(\mathbb{F}_p)$ atrašana kriptogrāfijā lietotajām līknēm ir skaitļošanas resursu prasoša operācija [7], jo lauku veidojošie pirmskaitļi ir relatīvi lieli skaitļi ($p$ bitu garums $\geq 160\ bitiem$). *Schoofa* algoritma

[2] $\mathbb{F}_p \subset \overline{\mathbb{F}}_p, \forall a \in \overline{\mathbb{F}}_p: a\ algebrisks\ pār\ lauku\ \mathbb{F}_p \to \exists f(x) \in \mathbb{F}_p: \deg(f(x)) > 0 \wedge f(a) = 0$

ātrdarbības novērtējumu biežāk sastopamajiem eliptisko līkņu bāzētu kriptosistēmu pirmskaitļu laukiem(*NIST* standarts) var redzēt attēlā 1.4.2.




*att. 1.4.2* ***Schoofa alogoritma darbības laika novērtējums NIST pirmskaitļiem***

Īpašu interesi izraisa tādas eliptiskās līknes $E(\mathbb{F}_p)$, kurām $\#E(\mathbb{F}_p) = \#\mathbb{F}_p = p$. Šādas līknes sauc par pirmskaitļu lauka anomālām (*prime field anomalous*) eliptiskām līknēm [8]. Pirmskaitļu lauka anomālas eliptiskās līknes nav lietojamas kriptogrāfisku sistēmu konstruēšanā, jo eksistē polinomiāla laika metodes diskrētā logaritma problēmas risināšanai, piemēram, *Semaev-Smart-Satoh-Araki* metode [8]. Tātad, būtiski parādīt, ka konkrētā kriptogrāfiskā sistēmā netiek lietota pirmskaitļu lauka anomāla eliptiska līkne.

### 1.4.2. $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu saskaitīšanas operācijas algebriska definīcija

Kā jau tika noskaidrots, $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punkti veido grupu attiecībā pret punktu saskaitīšanu. Šajā nodaļā precīzi definēsim saskaitīšanas operāciju.

Iesākumā interesēsimies par eliptiskām līknēm pār reālo skaitļu lauku. Ja aplūko eliptisko līkni pār reālo skaitļu lauku $\mathrm{E}(\mathbb{R})$, tad izrādās, ka $\mathrm{E}(\mathbb{R})$ punktu saskaitīšanas operāciju var viegli definēt ģeometriski. Ģeometriska punktu saskaitīšanas operācijas definīcija ir raksturīga visām kubiskām līknēm, kurām punktu skaits $> 0$ [3]. Punktu saskaitīšanas ģeometriskā metode ir plaši aprakstīta literatūrā, piemēram, var ieteikt iepazīties ar metodes aprakstu populārzinātniskā rakstā, atsauce pieejama literatūras avotā [13]. Vispārīgam priekšstatam, divus $\mathrm{E}(\mathbb{R})$ punktus $P\ un\ Q$ ģeometriski iespējams saskaitīt sekojoši:

- Plaknē attēlo $\mathrm{E}(\mathbb{R})$ veidoto grafiku
- Atzīmē izvēlētos punktus $P\ un\ Q$, caur tiem novelk taisni.

- Novilktā taisne krusto $\mathrm{E}(\mathbb{R})$ veidoto grafiku līknes punktā $-R$
- Atrod punktam $-R \in \mathrm{E}(\mathbb{R})$ pret x-asi simetrisko punktu $R \in \mathrm{E}(\mathbb{R})$
- $P + Q = R$

Viegli pamanīt, ka iespējami šādi speciālgadījumi: $P = Q, P = -Q, P = \mathcal{O}$

$P = Q$

- Novelk $\mathrm{E}(\mathbb{R})$ pieskares taisni punktā $P$,
- pieskare krusto $\mathrm{E}(\mathbb{R})$ veidoto grafiku līknes punktā $-R$
- Atrod punktam $-R \in \mathrm{E}(\mathbb{R})$ pret x-asi simetrisko punktu $R \in \mathrm{E}(\mathbb{R})$
- $Q + Q = R$

$P = -Q$

- Saskaņā ar komutatīvas grupas aksiomu: $Q + (-Q) = (-Q) + Q = \mathcal{O}$

$P = \mathcal{O}$

- Saskaņā ar komutatīvas grupas aksiomu: $\forall Q \in \mathrm{E}(\mathbb{R}): Q + \mathcal{O} = \mathcal{O} + Q = Q$

$P = Q, P = \mathcal{O}$

- $\mathcal{O} + \mathcal{O} = \mathcal{O}$

Fakts, ka divu līknes $\mathrm{E}(\mathbb{R})$ punktu saskaitīšanas operāciju iespējams definēt ģeometriski ir pārsteidzošs, tomēr algoritmiskai punktu efektīvai saskaitīšanai nepieciešamas algebriskas metodes. Izrādās, ka līknes $\mathrm{E}(\mathbb{R})$ punktu saskaitīšana var tikt veikta algebriski, izmantojot sekojošas sakarības [8], [9]:

$\mathrm{E}(\mathbb{R})$ punktu saskaitīšanas algebriska definīcija:

- $\mathrm{P} = (\mathrm{x}_1, \mathrm{y}_1) \in E(\mathbb{R})$, P aditīvi inversais elements: $-\mathrm{P} = (\mathrm{x}_1, -\mathrm{y}_1) \in E(\mathbb{R})$,
- $\mathrm{Q} = (\mathrm{x}_2, \mathrm{y}_2) \in E(\mathbb{R})$, $\mathrm{Q} \neq -\mathrm{P}$
- $\mathrm{P} + \mathrm{Q} = \mathrm{R} = (\mathrm{x}_3, \mathrm{y}_3) \in E(\mathbb{R})$, kur

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda(x_1 - x_3) - y_1$$

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, ja\ P \neq Q \\ \dfrac{3x_1^2 + a}{2y_1}, ja\ P = Q \end{cases}$$

- $\forall (\mathrm{x}, \mathrm{y}) \in E(\mathbb{R}) : (\mathrm{x}, \mathrm{y}) + (\mathrm{x}, -\mathrm{y}) = \mathcal{O}$
- $\mathcal{O} + \mathcal{O} = \mathcal{O}$
- $\forall (\mathrm{x}, \mathrm{y}) \in E(\mathbb{R}) : (\mathrm{x}, \mathrm{y}) + \mathcal{O} = \mathcal{O} + (\mathrm{x}, \mathrm{y}) = (\mathrm{x}, \mathrm{y}) \in E(\mathbb{R})$

Izrādās, ka definētās eliptisko līkņu punktu saskaitīšanu aprakstošās algebriskās sakarības ir attiecināmas uz eliptiskām līknēm pār galīgiem pirmskaitļu laukiem $E(\mathbb{F}_p)$ [9].

$E(\mathbb{F}_p)$ punktu saskaitīšanas algebriska definīcija:

- $(x_1, y_1) \in E(\mathbb{F}_p), (x_2, y_2) \in E(\mathbb{F}_p), x_1 \neq x_{2,}, tad\ \ (x_1, y_1) + (x_2, y_2) = (x_3, y_3) \in E(\mathbb{F}_p)$,kur

$$x_3 \equiv \lambda^2 - x_1 - x_2\ (mod\ p)$$

$$y_3 \equiv \lambda(x_1 - x_3) - y_1 (mod\ p)$$

$$\lambda \equiv \frac{y_2 - y_1}{x_2 - x_1}\ (mod\ p)$$

- $(x_1, y_1) \in E(\mathbb{F}_p), y_1 \neq 0, (x_1, y_1) + (x_1, y_1) = (x_2, y_2)$,kur

$$x_2 \equiv \lambda^2 - 2x_1\ (mod\ p)$$

$$y_2 \equiv \lambda(x_1 - x_2) - y_1 (mod\ p)$$

$$\lambda \equiv \frac{3x_1^2 + a}{2y_1} (mod\ p)$$

- $\forall (\mathrm{x}, \mathrm{y}) \in E(\mathbb{F}_p) :\ (\mathrm{x}, \mathrm{y}) + (\mathrm{x}, -\mathrm{y}) = \mathcal{O}$
- $\mathcal{O} + \mathcal{O} = \mathcal{O}$
- $\forall (\mathrm{x}, \mathrm{y}) \in\ E(\mathbb{F}_p)\ : (\mathrm{x}, \mathrm{y}) +\ \mathcal{O} = \mathcal{O} + (\mathrm{x}, \mathrm{y}) = (\mathrm{x}, \mathrm{y}) \in E(\mathbb{F}_p)$

Var secināt, ka $E(\mathbb{F}_p)$ punktu saskaitīšanas operācija algoritmiski var tikt efektīvi realizēta, izmantojot relatīvi vienkāršas algebriskas sakarības. $E(\mathbb{F}_p)$ līknes punktu saskaitīšana ir eliptisko līkņu bāzētu kriptogrāfisko sistēmu fundamentāla operācija un nosaka atslēgu ģenerēšanas un apmaiņas algoritmu teorētisko efektivitāti.

### 1.4.3. $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu skalārās reizināšanas operācijas algebriska definīcija

Definēsim $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu reizinājumu ar naturālu skaitli šādā veidā [14]:

$$Ja\ \ P \neq \mathcal{O}\ un\ P = (x_1, y_1) \in E(\mathbb{F}_p), n \in \mathbb{N}_1,$$

tad $n \circ P \overset{def}{\Longleftrightarrow} P + P + \cdots + P = \sum_1^n P$, tātad līknes punkts $P$ tiek pieskaitīts sev $n$ reizes. $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu reizinājumu ar naturālu skaitli turpmāk sauksim par $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu skalāro reizinājumu.

Literatūrā [14] aprakstīti vairāki algoritmi efektīvai $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu skalārā reizinājuma realizācijai, kā viens no ātrākajiem algoritmiem minēts algoritms *Montgomery ladder*, kas skalāros reizinājumus atrod $o(\log(n))$ laikā, pie nosacījuma, ka punkti attēloti projektīvā koordinātu sistēmā.

Klasiski skalārā reizināšana bieži tiek realizēta, izmantojot *Double-And-Add* algoritmu. Algoritms plaši aprakstīts un analizēts grāmatā *Cryptography Theory And Practice*, atsauce literatūras sarakstā [7].

### 1.4.4. $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu ģenerētās cikliskās apakšgrupas

Teiksim, ka grupa $X$ ir cikliska, ja $X = \langle x \rangle, kur\ x \in X$ [15]. Vienkāršā veidā $X$ varam definēt, uzdodot $X$ elementa $x$ pakāpju kopu. Secinājums: visas $X$ apakšgrupas, kuras satur elementu $x$, satur visas $x$ pakāpes, tātad $\langle x \rangle$ ir mazākā $X$ apakšgrupa, kura satur elementu $x$ [15].

Viegli redzēt, ka Eliptiskās līknes $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punkta $P$ visu skalāro reizinājumu kopa ir cikliska, algebriska kopas $\mathrm{E}(\mathbb{F}_\mathrm{p})$ apakšgrupa, kuru apzīmēsim $\langle \mathrm{P}, \circ \rangle$, līknes punktu $P$ sauksim par $\langle \mathrm{P}, \circ \rangle$ primitīvo elementu, jeb ģenerējošo punktu.

*Lagranža* teorēma. *Jebkurai galīgai grupai G ar apakšgrupu H :* $\#G | \#H$

Saskaņā ar *Lagranža* teorēmu, $\mathrm{E}(\mathbb{F}_\mathrm{p})$ apakšgrupu pakāpes ir $\#\mathrm{E}(\mathbb{F}_\mathrm{p})$ dalītāji, tātad $\mathrm{E}(\mathbb{F}_\mathrm{p})$ apakšgrupu pakāpes varam iegūt, faktorizējot skaitli $\#\mathrm{E}(\mathbb{F}_\mathrm{p})$.

Ieviesīsim diskrētā eksponenta un diskrētā logaritma jēdzienus.

Ja $X$ ir grupa un $x \in X$, tad $x^n$, kur $n \in \mathbb{N}$ ir diskrētais eksponents pie bāzes $x$ pakāpē $n$ [16].

Diskrētā eksponenta operācijas inversā operācija ir diskrētā logaritma atrašana: dots elements $h$, atrast tādu $n \in \mathbb{N}$ (ja šāds $n\ eksistē$), kuram $h = x^n$. Skaitli $n$ sauksim par elementa $h$ diskrēto logaritmu pie bāzes $x$ [16]. Var lietot klasisko logaritmu notāciju:

$$n = \log_x h\,, ja\ h = x^n$$

Diskrētajiem eksponentiem un diskrētajiem logaritmiem piemīt būtiskas klasisko eksponentu un logaritmu īpašības, kas ir fundamentāli svarīgas eliptisko līkņu bāzētu kriptogrāfisku sistēmu konstruēšanā [16]:

- $x^{n+m} = x^n * x^m$
- $\log_x(h * j) \equiv \log_x(h) + \log_x(j) \quad (mod\ \#\langle X \rangle)$

Precizēsim diskrēto eksponentu un diskrēto logaritmu jēdzienus $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu veidotajās cikliskajās apakšgrupās.

Ja $\langle \mathrm{P}, \circ \rangle$ ir cikliska $\mathrm{E}(\mathbb{F}_\mathrm{p})$ apakšgrupa ar primitīvo elementu $\mathrm{P} \in \mathrm{E}(\mathbb{F}_\mathrm{p})$ un $x \in \langle \mathrm{P}, \circ \rangle$, tad par diskrēto eksponentu sauksim skalāro reizinājumu $n \circ x, kur\ n \in \mathbb{N}$ pie bāzes $x$ pakāpē $n$. Un attiecīgi diskrētais logaritms: $n = \log_x h\,, ja\ h = n \circ x$. Diskrēto logaritmu atrašanas sarežģītība ir eliptisko līkņu kriptogrāfisku sistēmu fundamentālais drošības pamats.

# 2. ELIPTISKO LĪKŅU KRIPTOGRĀFISKĀS SISTĒMAS

Iepriekšējā nodaļā tika apskatīti eliptisko līkņu bāzētu kriptosistēmu matemātiskie pamati. Šajā nodaļā tiks pētīti minēto sistēmu datorzinātņu pamati.

Nodaļa strukturāli sastāv no sešām sadaļām. Iesākumā tiks aprakstīta diskrētā logaritma problēma eliptiskajām līknēm un tās nozīme kriptogrāfiskajās sistēmās, tam sekos ieskats eliptisko līkņu kriptogrāfisku sistēmu domēna parametru uzdošanā un pamatalgoritmos, tādos kā atslēgu ģenerēšana un apmaiņa, ciparu parakstu izsniegšana un verifikācija. Nodaļas noslēgumā tiks sniegts vispārīgs eliptisko līkņu kriptosistēmu un RSA sistēmu salīdzinājums.

## 2.1. Eliptisko līkņu bāzētu kriptosistēmu fundamentālā drošība

Eliptisko līkņu bāzētas kriptogrāfiskas sistēmas fundamentālā drošība balstās uz algoritmiski sarežģīto diskrētā logaritma problēmu eliptiskajām līknēm [8].

Diskrētā logaritma problēma ir relatīvi sen pazīstama skaitļu teorijas problēma [16], tomēr praktisku pielietojumu kriptogrāfijā tā ieguva pēc slavenās *Diffie* un *Hellman* publikācijas 1976. gadā, kas izraisīja apvērsumu kriptogrāfijā un pavēra iespējas konstruēt asimetriskas kriptogrāfiskas sistēmas [17]. Minētajā publikācijā tika definēta interesanta problēma, ko sauc par *Diffie-Hellman* problēmu [17]:

$$G - galīga\ cikliska\ grupa\ ar\ primitīvo\ elementu\ g \in G.$$

$$Algoritmiska\ problēma: atrast\ g^{s_A s_B}, ja\ zināms\ g^{s_A} un\ g^{s_B}$$

Un saistītā diskrētā logaritma problēma [17]:

$$G - galīga\ cikliska\ grupa\ ar\ primitīvo\ elementu\ g \in G.$$

$$Algoritmiska\ problēma: a \in G\ atrast\ tādu\ s \in \mathbb{N}, ka: g^s = a$$

Vispārīgajā gadījumā nav zināms, vai *Diffie-Hellman* problēma ir efektīvi atrisināma, vispirms atrisinot diskrētā logaritma problēmu, tāpat nav zināms, vai eksistē tādas grupas, kurām *Diffie-Hellman* problēma ir vienkāršāks skaitļošanas uzdevums kā diskrētā logaritma problēma [17].

Tiek uzskatīts, ka diskrētā logaritma atrašana atsevišķām algebriskām grupām ir algoritmiski sarežģīts uzdevums, tomēr apgalvojumam neeksistē formāls pierādījums [8],[16],[17]. Vienas no šādām grupām, kurām diskrētā logaritma atrašana tiek uzskatīta par algoritmiski sarežģītu uzdevumu, ir eliptisko līkņu pār galīgu pirmskaitļu lauku punktu ģenerētās cikliskās apakšgrupas [8],[16].

Uzreiz jāpiemin, ka RSA kriptosistēmas fundamentālā drošība balstās uz skaitļu faktorizācijas problēmu, kurai arī neeksistē pierādījums par efektīvu (polinomiāla laika) algoritmu neesamību.

### 2.1.1. Diskrētā logaritma problēma eliptiskajām līknēm

Šobrīd diskrētiem logaritmiem vislielākā praktiskā nozīme ir eliptisko līkņu kriptogrāfijā [16]. Pirmās eliptisko līkņu kriptogrāfiskās shēmas, kuru fundamentālā drošība balstās uz diskrētā logaritma problēmu, saistās ar *Neal Koblitz* un *Victor Miller* 1985. gada pētījumiem [16]. Minētie autori ieteica izmantot eliptiskās līknes pār pirmskaitļu laukiem veidotās punktu grupas diskrētā logaritma bāzētu kriptogrāfisku sistēmu konstruēšanā [8].

Atgādināšu, ka pirmās diskrētā logaritma bāzētas kriptogrāfiskas sistēmas saistās ar 1976. gadā publicētajiem *Diffie* un *Hellman* pētījumiem, kas pavēra ceļu publiskās atslēgas kriptogrāfijai. Tāpat nozīmīgi ieguldījumi diskrētā logaritma bāzētu kriptogrāfisku sistēmu attīstībā saistās ar ēģiptiešu zinātnieka *Tahera Elgamal* pētījumiem asimetriskajā kriptogrāfijā [8].

Definēsim diskrētā logaritma problēmu eliptiskajām līknēm šādā veidā : Ja $\mathrm{E}(\mathbb{F}_\mathrm{p})$ ir eliptiska līkne pār galīgu pirmskaitļu lauku, un $\langle P,\circ\rangle$ ir šīs līknes punkta $P$ ģenerētā cikliskā apakšgrupa attiecībā pret punktu skalārās reizināšanas operāciju, tad par diskrētā logaritma problēmu eliptiskajā līknēm sauksim sekojošu uzdevumu:

$$Dots\ \mathrm{E}(\mathbb{F}_\mathrm{p})\ punkts\ Q, tāds, ka\ Q \in \langle P,\circ\rangle \subseteq \mathrm{E}(\mathbb{F}_\mathrm{p})$$

$$Atrast\ tādu\ s \in \mathbb{N}, ka\ s \circ P = Q$$

Aprakstītā problēmas definīcija atbilst mūsdienu eliptisko līkņu bāzētu kriptogrāfisku sistēmu fundamentālajiem drošības pamatiem [9].

No diskrētā logaritma problēmas eliptiskajām līknēm definīcijas varam izdarīt vairākus triviālus secinājumus.

Viegli redzēt, ja $Q \in \langle P,\circ\rangle$, tad vienmēr šāds naturāls $s$ atradīsies, jo $P$ ir $\langle P,\circ\rangle$ primitīvais elements: $\forall X \in \langle P,\circ\rangle\ \exists i \in \mathbb{N}: i \circ P = X$.

Ja $n = \#\langle P,\circ\rangle$, tad $s \leq n-1$, jo grupa $\langle P,\circ\rangle$ pēc tās definīcijas satur visu punkta $P$ skalāro reizinājumu kopu.

Būtiski piezīmēt, ka literatūrā sastopamas vairākas diskrētā logaritma problēmas eliptiskajām līknēm definīcijas, kas savstarpēji atšķiras ar algebrisko grupu un diskrētā logaritma bāzi, kas mūsu definīcijā ir līknes punkts un $\langle P,\circ\rangle$ ģenerējošais elements $P$. Vispārīgā veidā diskrētā logaritma problēmu var definēt parametru divniekam $(G,a)$, kur $G$ ir algebriska grupa, $a$ – diskrētā logaritma bāze [7]. Skaidrs, ka atkarībā no problēmas definīcijas (parametru divnieka īpašībām), atšķirsies problēmas atrisināmības sarežģītība.

### 2.1.2. Algoritmi diskrētā logaritma atrašanai

Jebkuras kriptogrāfiskas sistēmas drošība, attiecībā pret atslēgu uzlaušanas noturību, ir tiešā veidā saistīta ar algoritmu efektivitāti. Jo efektīvāks algoritms eksistē atslēgas atrašanai, jo lielāks atslēgas bitu garums ir nepieciešams, lai paildzinātu neautorizētu atslēgas atrašanu.

Šajā nodaļā vispārīgi aplūkosim šobrīd zināmos, efektīvākos algoritmus, kas risina diskrētā logaritma problēmu eliptiskajām līknēm. Interesanti, ka kompānija *Certicom Reasearch* izsludinājusi konkursu ar vērienīgu naudas balvu par efektīvāka algoritma izstrādi un uzrādīšanu [7].

Iesākumā aplūkosim *naivo* algoritmu. Saskaņā ar diskrētā logaritma problēmas eliptiskajām līknēm definīciju un pieņemot, ka mums ir zināma cikliskās apakšgrupas pakāpe $n = \#\langle P, \circ \rangle$, kas parasti ir viens no publiski pieejamajiem eliptisko līkņu bāzētas kriptosistēmas domēna parametriem, tad var apgalvot, ka ar *naivo* algoritmu, sliktākajā gadījumā jāapskata $n - 1$ iespējamās atslēgas. Tā kā literatūrā ieteikts izvēlēties apakšgrupas, kuru pakāpe $n \geq 2^{160}$ [7],[8], tad praktiska šāda *naivā* algoritma darbināšana atslēgas atrašanai nav piemērota(iespējama). Tālāk aplūkosim efektīvākus algoritmus diskrētā logaritma atrašanai.

#### *Index Calculus* algoritms

*Index Calculus* metode ir populārs sub-eksponenciāla laika algoritms diskrētā logaritma atrašanai galīgos pirmskaitļu laukos. Algoritms vispārīgā veidā balstās uz lineāru vienādojumu sistēmu sastādīšanu un diskrētā logaritma atrašanu, atrisinot sastādīto sistēmu [18].

Uzreiz jāpiezīmē, ka, lai gan literatūrā eksistē daudzi mēģinājumi vispārināt *Index Calculus* metodi attiecībā uz diskrētā logaritma problēmu eliptiskajām līknēm [8], tomēr līdz šim nav publiski pieejami pozitīvi rezultāti, kas attiecināmi uz patvaļīgu eliptisko līkņu kriptosistēmu [8],[9]. Jāpiebilst, ka eksistē neliela eliptisko līkņu kopa pār galīgu pirmskaitļu laukiem, kurai diskrētā logaritma problēma atrisināma vai atvieglojama ar *Index Calculus* metodi [9]. Piemēram, ja $\mathrm{E}(\mathbb{F}_\mathrm{p})$ cikliskās apakšgrupas pakāpe $n$ dala skaitli $\mathrm{p}^\mathrm{B} - 1, kur\ B \in [1; 100[$, tad diskrētā logaritma problēmu eliptiskajām līknēm iespējams reducēt uz klasisko diskrētā logaritma problēmu [9].

Lai izvairītos no *Index Calculus* metodes pielietošanas kriptogrāfisku sistēmu atslēgu atrašanā, kompānija *Certicom Reasearch* iesaka izvēlēties tādu cikliskās apakšgrupas pakāpi $n$, ka $p^B \not\equiv 1(mod\ n), kur\ B \in [1; 100[$

### *Pohlig-Hellman* algoritms

*Pohlig-Hellman* algoritms ir vispārīga metode diskrētā logaritma atrašanai cikliskās apakšgrupās ar pakāpi $n$. Algoritms balstās uz apakšgrupas pakāpes $n$ faktorizāciju pirmreizinātājos $n = \prod p_i$. Ja zināmi skaitļa $n$ pirmreizinātāji, tad diskrētā logaritma problēmu iespējams reducēt, apskatot apakšproblēmas skaitā, kas vienāds ar $n$ pirmreizinātāju skaitu, kurš ir mazāks vai vienāds ar $\log_2 n$ [8],[18]. Ja $n$ pirmreizinātāji ir nepietiekoši lieli skaitļi, tad *Pohlig-Hellman* algoritms var efektīvi atrisināt diskrētā logaritma problēmu [18] un atrast kriptogrāfiskās sistēmas atslēgu.

Ar *Pohlig-Hellman* algoritma analīzi un matemātisko pamatojumu var tuvāk iepazīties rakstā nr.18 no literatūras saraksta, kas ir salīdzinoši konspektīvs un pietiekami informatīvs metodes apraksts.

Lai gan algoritms tiek uzskatīts par efektīvu metodi diskrētā logaritma problēmas atrisināšanai, praksē ir relatīvi viegli aizsargāties pret uzbrukumiem, kas saistās ar *Pohlig-Hellman* algoritma praktisku pielietošanu kriptogrāfisku sistēmu atslēgu atrašanai. Lai sasniegtu maksimālo iespējamo aizsardzību pret uzbrukumiem, kuri realizē *Pohlig-Hellman* algoritmu, jāpārliecinās, ka cikliskās apakšgrupas pakāpe $n$ ir pirmskaitlis, vai vismaz tā dalītāji ir pietiekoši lieli pirmskaitļi.

### *Baby-step giant-step* algoritms

Tāpat kā *Pohlig-Hellman algoritms, Baby-step giant-step* metode risina diskrētā logaritma problēmu patvaļīgai cikliskai grupai $G$. Algoritms ir relatīvi vienkāršs un vispārīgi strādā šādā veidā: Dota cikliska grupa $G = \langle g \rangle$ un tās pakāpe $n$, aprēķina skaitli $m = \lceil \sqrt{n} \rceil$, atrod kopu ar šādiem grupas $G$ elementiem: $\{g^0, g^m, g^{2m}, \dots, g^{(m-1)m}\}$, sauktu par *Giant steps* [18]. Izskaitļo $h, hg^{-1}, hg^{-2}, \dots, hg^{-(m-1)}$, sauktus par *Baby steps*. Ja $h \in \langle g \rangle$, tad $h = g^{i+jm}$, kur $0 \leq i, j \leq m-1$ un *Giant steps* kopas elements $g^{jm}$ sakritīs ar *Baby steps* kopas elementu $hg^{-i}$ [16].

Metodes laika un telpas sarežģītība ir $O(\sqrt{\#G})$ [16]. Praktiski, algoritma darbības laiku nosaka grupas $G$ aritmētisko operāciju sarežģītība un laiks, kas nepieciešams, lai sakārtotu un salīdzinātu divu kopu elementus. Ja algoritma darbības laiku ir iespējams efektīvi optimizēt (piemēram, kopu kārtošanu aizstāt ar jaucējfunkciju izmantošanu), tad neeksistē efektīvas metodes, kā samazināt konkrētā algoritma darbībai nepieciešamo atmiņas apjomu [16].

Lai aizsargātos pret uzbrukumiem, kas saistās ar *Baby-step giant-step* metodes pielietošanu kriptogrāfiskas sistēmas atslēgas atrašanai, jānodrošina, ka minētā kriptosistēma pielieto ciklisku apakšgrupu ar pietiekoši lielu pakāpi. Kā jau minēts, praktiskos eliptisko līkņu

kriptosistēmu risinājumos, literatūrā tiek rekomendētas cikliskās apakšgrupas, kuru pakāpe $n \geq 2^{160}$.

### *Pollard* $\rho$-metode

*Pollard* $\rho$-metode orģināli izstrādāta diskrētā logaritma atrašanai galīgos pirmskaitļu laukos, tomēr var tikt vispārināta uz patvaļīgu ciklisku grupu $G$ [18]. Tāpat kā iepriekš aprakstītais *Baby-step giant-step* algoritms, *Pollard* $\rho$-metodes darbības laika novērtējums ir $O(\sqrt{\#G})$, tomēr šī algoritma darbībai nepieciešams ievērojami mazāks atmiņas apjoms [16]. Algoritms ir relatīvi sarežģītāks kā iepriekš aprakstītais, tomēr ļoti vispārīgi to var aprakstīt šādā veidā [16]:

Grupas $G = \langle g \rangle$ elementu kanonisko reprezentāciju kopu sadala 3 apakškopās $S_1, S_2, S_3$ ar aptuveni vienādu elementu skaitu, $x_0 = 1, h \in G, \forall i \in \mathbb{N}$:

$$x_{i+1} = \begin{cases} x_i h, ja\ x_i \in S_1 \\ {x_i}^2, ja\ x_i \in S_2 \\ x_i g, ja\ x_i \in S_3 \end{cases}$$

Tad $\exists a_i, b_i \in \mathbb{Z} : x_i = h^{a_i} g^{b_i}$, ja izdodas parādīt, ka eksistē tāds $i \geq 1$, ka $x_i = x_{2i}$, tad seko, ka $x_i = h^{a_i} g^{b_i} = h^{a_{2i}} g^{b_{2i}} = x_{2i}$, šādā situācijā diskrēto logaritmu iespējams atrast risinot lineāru vienādojumu sistēmu.

*Pollard* $\rho$-metode ir šobrīd labākais zināmais, praktiski pielietojamais algoritms diskrētā logaritma atrašanai patvaļīgai cikliskai grupai $G$ [8]. Ja par algoritma soli uzskata vienu $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punktu saskaitīšanas operāciju, tad diskrētā logaritma problēmu eliptiskajām līknēm ar *Pollard* $\rho$-metodi iespējams atrisināt $\frac{\sqrt{\pi n}}{2}$ soļos, kur $n$ līknes punkta $P$ ģenerētās cikliskās apakšgrupas pakāpe [8]. Turklāt parādīts, ka *Pollard* $\rho$-metodi iespējams efektīvi paralelizēt. Ar skaitļotāju, kuram ir $r$ skaitļošanas vienības (procesori), diskrētā logaritma problēma atrisināma $\frac{\sqrt{\pi n}}{2r}$ soļos [8], [18]. Ļoti interesants teorētisks novērtējums [8]: *Ar* $1 * 10^4$ *skaitļotājiem, kur katrs skaitļotājs atbilst* $1 * 10^3$ *MIPS (Millions of instructions per second) skaitļošanas jaudai, diskrētā logaritma problēmu eliptiskajām līknēm ar paralelizētu Pollard* $\rho$*-metodi iespējams atrisināt* $8{,}5 * 10\text{^}4$ *gados, ja cikliskās apakšgrupas pakāpe ir aptuveni* $2^{160}$.

Jāpiemin, ka eksistē dažādi publicēti rekordi attiecībā uz diskrētā logaritma risināšanu. Aplūkojot *Wikipedia* rakstu *Discrete logarithm records*, kas atrodams interneta adresē *https://en.wikipedia.org/wiki/Discrete_logarithm_records*, redzam, ka populārākā metode diskrētā logaritma problēmas eliptiskajām līknēm atrisināšanai ir *Pollard* $\rho$-metode.

Tāpat kā iepriekš, lai aizsargātos pret uzbrukumiem, kas saistās ar *Pollard* $\rho$-metodes pielietošanu kriptogrāfiskas sistēmas atslēgas atrašanai, jānodrošina, ka minētā kriptosistēma pielieto ciklisku apakšgrupu ar pietiekoši lielu pakāpi.

## 2.2. Sistēmas domēna parametri

Literatūrā eksistē vairāki eliptisko līkņu bāzētu kriptogrāfisku sistēmu parametru uzdošanas veidi, atkarībā no galīgā lauka un drošības prasībām. Būtiski, ka izvēlētais parametru reprezentācijas veids viennozīmīgi definē kriptosistēmu. Piedāvāju iepazīties ar diviem domēna parametru uzdošanas veidiem, kurus standartizējusi *Certicom Research* [9]. Pirmajā gadījumā aplūkosim eliptiskās līknes pār galīgu pirmskaitļu lauku $\mathrm{E}(\mathbb{F}_\mathrm{p})$, savukārt otrajā gadījumā - eliptiskās līknes pār galīgu bināru lauku $\mathrm{E}(\mathbb{F}_{2^\mathrm{m}})$.

Ja kriptogrāfiska sistēma izmanto *Veierštrāsa* līknes pār galīgu pirmskaitļu lauku $\mathbb{F}_\mathrm{p}$, to iespējams viennozīmīgi definēt ar sešinieku $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$ :

$$D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = (p, a, b, G, n, h), kur\ p - nepāra\ pirmsskaitlis, a, b \in \mathbb{F}_\mathrm{p}, G \in \mathrm{E}(\mathbb{F}_\mathrm{p}), n = \#\langle G, \circ \rangle \in \mathbb{N}, h = \frac{\#\mathrm{E}(\mathbb{F}_\mathrm{p})}{n} \in \mathbb{N}$$

Pirmskaitlis $p$ nosaka galīgu pirmskaitļu lauku $\mathbb{F}_\mathrm{p}$. Parametru komponenti $a, b$, saskaņā ar definīciju (skatīt nodaļu *Eliptiskās līknes pār galīgu pirmskaitļu lauku*), nosaka eliptisko līkni $\mathrm{E}(\mathbb{F}_\mathrm{p})$. Komponents $G = (G_x, G_y) \in \mathrm{E}(\mathbb{F}_\mathrm{p})$ - eliptiskās līknes $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punkts, kurš ģenerē ciklisku apakšgrupu $\langle G, \circ \rangle$ attiecībā pret eliptiskās līknes punktu skalārās reizināšanas operāciju. Parametrs $n$ saturiski nosaka līknes punktu skaitu cikliskajā apakšgrupā $\langle G, \circ \rangle$, kas sakrīt ar apakšgrupas pakāpi. Parametrs $h$ ir cikliskās apakšgrupas pakāpes $\#\langle G, \circ \rangle = n$ kofaktors.

Savukārt, ja kriptogrāfiska sistēma izmanto *Veierštrāsa* līknes pār galīgu bināru lauku $\mathbb{F}_{2^\mathrm{m}}$, to iespējams viennozīmīgi definēt ar septītnieku $D_{\mathrm{E}(\mathbb{F}_{2^\mathrm{m}})}$ :

$$D_{\mathrm{E}(\mathbb{F}_{2^\mathrm{m}})} = (m, P(x), a, b, G, n, h), kur\ m \in \mathbb{N}, P(x) - nereducējams\ polinomus\ ar\ pakāpi\ \deg(P(x)) = m;\ a, b \in \mathbb{F}_{2^\mathrm{m}}; G \in \mathrm{E}(\mathbb{F}_{2^\mathrm{m}}); n = \#\langle G, \circ \rangle \in \mathbb{N}; h = \frac{\#\mathrm{E}(\mathbb{F}_{2^\mathrm{m}})}{n} \in \mathbb{N}$$

Kā redzams, atšķirībā no $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$, parametra $p$ vietā kortežā $D_{\mathrm{E}(\mathbb{F}_{2^\mathrm{m}})}$ iekļauti divi domēna parametri $m, P(x)$, kas viennozīmīgi definē galīgu lauku. Aplūkosim sīkāk šos parametrus:

Parametrs $m \in \mathbb{N}$ nosaka elementu skaitu laukā, savukārt $P(x)$ ir monisks, laukā $\mathbb{F}_{2^\mathrm{m}}$ nereducējams polinoms ar koeficientiem no lauka $\mathbb{F}_2$ un sakni $\alpha \to P(\alpha) = 0$. $\alpha$ - $\mathbb{F}_{2^\mathrm{m}}$

primitīvais elements. Zināms, ka katram galīgam laukam eksistē vismaz viens primitīvais elements.

Abos aplūkotajos gadījumos, korteži $D_{\mathrm{E}(\mathbb{F}_{\mathrm{p}})}$, $D_{\mathrm{E}(\mathbb{F}_{2^{\mathrm{m}}})}$ definē minimālo parametru kopu, kas ļauj viennozīmīgi noteikt izvēlēto kriptosistēmu. Papildus nosauktajiem eliptisko līkņu bāzētu kriptogrāfisku sistēmu domēna parametriem, var tikt tik pievienoti parametri, kas raksturīgi konkrētai implementācijai. Piemēram, ja līknes parametri $a$ un $b$ tiek varbūtiski ģenerēti, parametru kopai var pievienot skaitli $s$, kuram pielietojot jaucējfunkciju $f$ tiek iegūtas $a$ un $b$ vērtības: $f(s) = \{a, b\}$. Šāds parametrs ļautu kriptosistēmas lietotājiem verificēt domēna parametru leģitimitāti.

Šajā darbā kriptosistēmas domēna parametri tiks definēti, izmantojot kortežu $D_{\mathrm{E}(\mathbb{F}_{\mathrm{p}})}$. Doto reprezentāciju autors izvēlējās, jo tā precīzi un uzskatāmi definē verificējamus kriptosistēmas atribūtus un īpašības pār galīgu pirmskaitļu lauku $\mathbb{F}_{\mathrm{p}}$.

Sīkāk par $D_{\mathrm{E}(\mathbb{F}_{\mathrm{p}})}$ ģenerēšanu lasīt nodaļā *Domēna parametru korteža* $D_{\mathrm{E}(\mathbb{F}_{\mathrm{p}})}$ *definēšana.*

## 2.3. Kriptogrāfisko atslēgu ģenerēšana

Eliptisko līkņu kriptosistēmas nodrošina asimetrisku kriptogrāfisko atslēgu ģenerēšanu un apmaiņu. Atbilstoši publiskās atslēgas kriptosistēmu konceptam, eliptisko līkņu bāzētās kriptogrāfiskās shēmās ir divi atslēgu modeļi – publiskā atslēga un privātā atslēga. Katram sistēmas lietotājam tiek piekārtots atslēgu pāris, kas sastāv no konkrēta lietotāja publiskās un privātās atslēgas.

Lietotājs izmanto privāto atslēgu, lai ģenerētu publisko atslēgu, atšifrētu ar savu publisko atslēgu nošifrētu informāciju, izveidotu ciparu parakstu un veiktu citas operācijas, ko nosaka konkrētā kriptogrāfiskā shēma un tās implementācija.

Lietotāja publiskā atslēga tiek izplatīta šifrētās saziņas partneriem un tiek izmantota informācijas šifrēšanai vai konkrēta lietotāja ciparu paraksta verifikācijai. Būtiski, ka publiskā atslēga netiek izmantota informācijas atšifrēšanas procedūrā un tā ir relatīvi droši izplatāma nedrošos saziņas kanālos, piemēram, internetā.

Lai kriptoanalītiķis varētu no eliptisko līkņu bāzētas kriptosistēmas publiskās atslēgas iegūt atbilstošo privāto atslēgu, ir jārisina diskrētā logaritma problēma eliptiskajām līknēm, kas, kā jau vairākkārt uzsvērts, ir algoritmiski sarežģīts skaitļošanas uzdevums.

Eliptisko līkņu kriptogrāfiskās sistēmās ar domēna parametriem $D_{\mathrm{E}(\mathbb{F}_{\mathrm{p}})}$ atslēgu primitīvi atbilst sekojošiem datu tipiem:

- Privātā atslēga : $k \in \mathbb{N}_1 < \#\langle G, \circ \rangle$

- Publiskā atslēga : $Q = (x_Q, y_Q) \in \langle G, \circ \rangle / \{\mathcal{O}\}$

Vispārīgi, eliptisko līkņu kriptogrāfisku sistēmu atslēgu primitīvu ģenerēšana notiek pēc sekojoša algoritma [9].

1. Definēta šifrētās saziņas partneru kopa $U$
2. Definēts sistēmas domēna parametru kortežs $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
3. Katram $u \in U$ piekārto atslēgu kortežu $(k, Q)$, kur $k$ ir lietotāja $u$ privātā atslēga un $Q$ – publiskā atslēga.
   3.1. Privātā atslēga $k$ ir vesels skaitlis, kas tiek varbūtiski izvēlēts no intervāla $[1, n-1]$ $kur\ n \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
   3.2. Publiskā atslēga $Q$ ir eliptiskās līknes $\mathrm{E}(\mathbb{F}_\mathrm{p})$ punkts, kas iegūts sekojoši:
   $Q = k \circ G, kur\ k\ ir\ iepriekšējā\ solī\ atrastā\ privātā\ atslēga\ un\ G \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$

## 2.4. Atslēgu apmaiņas algoritmi

Kriptogrāfisko atslēgu apmaiņas operācija nodrošina, ka divām saziņā iesaistītajām pusēm $A$ un $B$ tiek ģenerēta atslēga $S$, kas ir konkrēto lietotāju kopējais noslēpums un var tikt tālāk izmantota simetriskos vai pseidosimetriskos datu aizsardzības risinājums, piemēram, *ElGamal* kriptogrāfiskā shēma šifrētai ziņapmaiņai [8].

Eliptisko līkņu kriptogrāfiskās sistēmās ar domēna parametriem $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$, kopīgā noslēpuma $S$ primitīvs atbilst šādiem datu tipiem:

$$S = (x_s, y_s) \in \langle G, \circ \rangle / \{\mathcal{O}\}$$

Klasiska atslēgu apmaiņa eliptisko līkņu kriptogrāfiskajās sistēmās notiek saskaņā ar pielāgotu *Diffie-Hellman* protokolu ECDH (*Elliptic Curve Diffie-Hellman*) [8],[17]. ECDH algoritms diviem saziņas partneriem [8],[9]:

1. Definē saziņas partneru kopu $U = \{u, v\}$
2. Saziņas dalībnieki $u, v$ vienojas par kriptogrāfiskās sistēmas parametru kortežu $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
3. Definētajām $U$ un $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$ ģenerē kriptogrāfisko atslēgu primitīvus:
   3.1. $u \to (k_u, Q_u), v \to (k_v, Q_v)$
4. $u, v$ apmainās ar publiskajām atslēgām, izmantojot kļūdu tolerantu (kļūdu korekcija var tikt implementēta kriptogrāfiskajā sistēmā), potenciāli publisku, saziņas kanālu $I$:
   4.1. $u$ nosūta $v$ publisko atslēgu $Q_u$
   4.2. $v$ nosūta $u$ publisko atslēgu $Q_v$
5. $u$ verificē saņemto $Q_v'$, $v$ verificē saņemto $Q_u{}'$ attiecībā pret $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$ šādiem verifikācijas nosacījumiem [9]:

5.1. $Q_v' \neq \mathcal{O} = (x_{Q_v}, y_{Q_v}) \wedge Q_u' \neq \mathcal{O} = (x_{Q_u}, y_{Q_u})\ kur\ \mathcal{O} \in \mathrm{E}(\mathbb{F}_\mathrm{p})$

5.2. $n \circ Q_v' = n \circ Q_u' = \mathcal{O}, kur\ n \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})},\ \mathcal{O} \in \mathrm{E}(\mathbb{F}_\mathrm{p})$

5.3. $y_{Q_v}^2 \equiv x_{Q_v}^3 + x_{Q_v}a + b\ (mod\ p) \wedge y_{Q_u}^2 \equiv x_{Q_u}^3 + x_{Q_u}a + b\ (mod\ p), kur\ p, a, b \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$

5.4. Ja kāds no verifikācijas nosacījumiem ir aplams, atslēgu apmaiņu atkārto.

5.5. Ja visi verifikācijas nosacījumi ir patiesi, pieņem, ka $Q_v' = Q_v \wedge Q_u' = Q_u$.

6. $u, v$ aprēķina kopīgo noslēpumu $S$:

6.1. $u$ aprēķina $S_u = k_u \circ Q_v$

6.2. $v$ aprēķina $S_v = k_v \circ Q_u$

6.3. $u, v$ verificē $S_u, S_v$ :

6.3.1. Ja $S_u \neq \mathcal{O} \wedge S_v \neq \mathcal{O}$, pieņem ka $S = S_u = S_v$

6.3.2. Ja $S_u = \mathcal{O} \vee S_v = \mathcal{O}$, atkārto atslēgu apmaiņas algoritmu

## 2.5. Ciparu paraksti

Viens no būtiskākajiem modernas kriptogrāfijas uzdevumiem ir nodrošināt informācijas izcelsmes autentiskuma verifikāciju, kas tiek īstenota kā pārbaudāmu ciparu parakstu ģenerēšana un verifikācija. Eksistē vairākas standartizētas ciparu parakstu shēmas, kas balstās uz publisko atslēgu kriptogrāfiskajām sistēmām, tādām kā RSA un eliptisko līkņu kriptosistēmas. Kā vienu no populārākajām ciparu parakstu shēmām var minēt DSA (*Digital Signature Algorithm*) [7].

Vispārīgajā gadījumā ziņojuma $m$ autors $A$ vēlas ziņojumam pievienot ciparu parakstu, tā, ka ziņojuma saņēmējs $B$ var pārbaudīt $m$ autentiskumu.

Populārākā standartizētā (*NIST FIPS 186-4*) eliptisko līkņu kriptogrāfijas ciparu parakstu ģenerēšanas un verifikācijas metodoloģija ir ECDSA (*Elliptic Curve Digital Signature Algorithm*), kas ir DSA modifikācija eliptiskajām līknēm [7],[19].

ECDSA ir viens no populārākajiem eliptisko līkņu pielietojumiem IT industrijā. Pazīstamākās tehnoloģijas, ar relatīvi lielu lietotāju skaitu, kas praktiski pielieto ECDSA implementācijas ir *Bitcoin*, *SSH*, *TLS*, Austrijas e-pārvaldes sistēma [2].

Aplūkosim ECDSA ciparu parakstu ģenerēšanas un verifikācijas pamatalgoritmus.

### 2.5.1. ECDSA ciparu paraksta ģenerēšanas algoritms

Pieņemsim, ka saziņas puse $u$ vēlas pievienot ciparu parakstu ziņojumam $m \in \mathbb{N}$ un nosūtīt parakstītu ziņojumu saziņas pusei $v$.

Eliptisko līkņu kriptogrāfiskās sistēmās ar domēna parametriem $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$, ECDSA ciparu paraksta primitīvs atbilst sekojošiem datu tipiem:

$$\big(m,(r,s)\big), kur\ m,r,s \in \mathbb{N}$$

ECDSA ciparu paraksta ģenerēšanas algoritma apraksts (Autora pielāgots *FIPS 186-4* standarta variants [9], [19]).

1. Definē saziņas partneru kopu $U = \{u, v\}$
2. Saziņas dalībnieki $u, v$ vienojas par kriptogrāfiskās sistēmas parametru kortežu $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
3. Definētajām $U\backslash\{v\}$ un $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$ ģenerē kriptogrāfisko atslēgu primitīvus:
   3.1. $u \rightarrow (k_u, Q_u)$
4. Saziņas dalībnieks $u$ ziņojumam $m \in \mathbb{N}$ ģenerē ciparu parakstu:
   4.1. $u$ varbūtiski izvēlas veselu skaitli $t \in [1, n-1], kur\ n \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
   4.2. $Q' = t \circ G, = \big(x_{Q'}, y_{Q'}\big), kur\ G \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
   4.3. $r \equiv x_{Q'}\ (mod\ n), kur\ n \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
      4.3.1. Ja $r = 0$, atgriežas pirmajā solī
   4.4. $s \equiv t^{-1}(m + rk_u)\ (mod\ n)$
      4.4.1. Ja $s = 0$, atgriežas pirmajā solī
   4.5. Ziņojumam $m$ piekārto ciparu parakstu $m \rightarrow (r, s)$
5. Saziņas dalībnieks $u$ dalībniekam $v$ nosūta parakstītu ziņojumu $(m, (r, s))$

### 2.5.2. ECDSA ciparu parakstu verifikācijas algoritms

Pieņemsim, ka saziņas puse $v$ ir saņēmusi no autora $u$ parakstītu ziņojumu $(m', (r, s))$ un vēlas verificēt ziņojuma autentiskumu.

ECDSA ciparu parakstu verifikācijas algoritms [9]:

1. $v$ no $u$ saņem parakstītu ziņojumu $(m', (r, s))$
2. $v$ izskaitļo $e \equiv m's^{-1}\ (mod\ n), f \equiv m'r^{-1}\ (mod\ n)$
3. $R = e \circ G + f \circ Q_u, kur\ G \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
4. Ja $R = \mathcal{O} \rightarrow m' \neq m: \big(m', (r, s)\big)$ tiek noraidīts
5. $R = (x_R, y_R), l \equiv x_R\ (mod\ n), kur\ n \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
6. Ja $l = r \rightarrow m' = m: \big(m', (r, s)\big)$ tiek atzīts par autentisku
7. Ja $l \neq r \rightarrow m' \neq m: \big(m', (r, s)\big)$ tiek noraidīts

## 2.6. Salīdzinājums ar RSA kriptogrāfisko sistēmu

Tā kā mūsdienās viena no visvairāk pielietotajām kriptogrāfiskajām sistēmām ir RSA, nepieciešams parādīt, kādas ir eliptisko līkņu bāzētu kriptogrāfisku sistēmu priekšrocības un trūkumi, attiecībā pret RSA. Ievērojami plašāk šo jautājumu autors apskatījis savos iepriekšējos darbos [2], tomēr, vispārīgam ieskatam, salīdzināsim abu sistēmu fundamentālos drošības pamatus un atslēgu ģenerēšanas algoritmu ātrdarbību.

Jāmin, ka, atšķirībā no eliptisko līkņu kriptosistēmām, RSA kriptosistēmas matemātiskie pamati ir balstīti uz relatīvi vienkāršām skaitļu teorijas operācijām, kas pieļauj šādas sistēmas salīdzinoši vienkārši implementēt un uzturēt. Sīkāk par RSA kriptosistēmu var lasīt *A. Salomaa* grāmatā *Public-Key Cryptography* [1].

### 2.6.1. Sistēmu drošības salīdzinājums

Eliptisko līkņu kriptogrāfisko sistēmu fundamentālā drošība balstīta uz diskrētā logaritma problēmu eliptiskajām līknēm (DLPEL), kas skaidrota iepriekšējās nodaļās. RSA kriptogrāfiskās sistēmas fundamentālā drošība balstīta uz skaitļa faktorizācijas problēmu : *dots* $n$*, kas ir divu lielu pirmskaitļu* $p$ *un* $q$ *reizinājums, atrast* $p$ *un* $q$*, tādus, ka* $pq = n$ [1].

Tiek uzskatīts, ka gan DLPEL, gan skaitļa faktorizācijas problēma ir algoritmiski sarežģīti skaitļošanas uzdevumi, kuriem nav zināmi efektīvi, praktiski pielietojami algoritmi (eksistē efektīvi kvantu algoritmi). Tomēr parādīts, ka DLPEL ir šobrīd sarežģītāks uzdevums kā skaitļa faktorizācijas problēma, kura ir atrisināma sub-eksponenciālā laikā ar *General Number Field Sieve* algoritmu [7]. Kā jau noskaidrojām, vispārīgā gadījumā DLPEL ir atrisināma eksponenciālā laikā ar *Pollard* $\rho$-metodi.

Sub-eksponenciāla laika algoritma neesamība DLPEL atrisināšanai ir lielākā eliptisko līkņu kriptosistēmu priekšrocība attiecībā pret RSA. Praktiski tas nozīmē, ka, pie vienāda drošības līmeņa, attiecībā uz nepieciešamo laiku atslēgu uzlaušanai, eliptisko līkņu kriptosistēmu atslēgu bitu garumi, salīdzinājumā ar RSA atslēgu bitu garumiem būs ievērojami mazāki. Piemēram, RSA kriptosistēmai ar 1024 bitu garām atslēgām drošības ziņā ir ekvivalenta eliptisko līkņu kriptosistēma ar 163 bitu garām atslēgām [20]. Pieaugot drošības līmenim, atslēgu izmēru starpība pieaug eksponenciāli par labu eliptisko līkņu kriptosistēmām. Šāda, būtiska atslēgu bitu izmēru starpība paver plašas iespējas implementēt efektīvas un drošas eliptisko līkņu bāzētas kriptogrāfiskas sistēmas dažādās zema energopatēriņa mobilajās ierīcēs, iegultajās sistēmās, sensoru tīklos utt.

### 2.6.2. Atslēgu ģenerēšanas algoritmu veiktspējas salīdzinājums

Literatūrā eksistē vairāki pētījumi, kas salīdzina abu kriptosistēmu atslēgu ģenerēšanas ātrdarbību konkrētās skaitļotāju arhitektūrās. Vispārīgs secinājums: *Pie fiksēta drošības līmeņa, attiecībā uz laiku, kas nepieciešams, lai nesankcionēti atrastu kriptosistēmu atslēgas, eliptisko līkņu kriptosistēmu atslēgu ģenerēšanas algoritmi praktiski strādā ātrāk kā RSA atslēgu ģenerēšanas algoritmi.* [20],[21].

Vispārīgā gadījumā, pieaugot drošības līmenim, atslēgu ģenerēšanas algoritmu darbības laika starpība pieaug eksponenciāli [20].

Eksistē interesants pētījums [21], kura autori eksperimentāli parāda fundamentālu sakarību starp procesora vārda garumu un kriptosistēmu atslēgu garumu: *Relatīvā eliptisko līkņu kriptosistēmu veiktspēja, attiecībā pret RSA kriptosistēmu veiktspēju palielinās, ja arhitektūras vārda garums samazinās* [21].

Var izteikt intuitīvu hipotēzi, ka eliptisko līkņu kriptosistēmu atslēgu ģenerēšanas algoritmu darbības laika pārākums, attiecībā pret RSA atslēgu ģenerēšanas algoritmu, pie fiksēta drošības līmeņa, ir saistīts ar būtiski mazāku atslēgu bitu garumu, kas ļauj samazināt rēķināšanai nepieciešamo laiku.

Eliptisko līkņu atslēgu pāra ģenerēšanas algoritma laikietilpīgākā operācija ir punkta skalārā reizinājuma aprēķināšana, savukārt RSA sistēmā, tās ir divu lielu pirmskaitļu atrašana, pārbaude un *Eiklīda* algoritma izpilde pirmreizinātāju atrašanai.

Kā jau tika minēts, efektīva eliptisko līkņu skalārā reizinājuma operācijas algoritma ātrdarbības novērtējums ir $O(logn), kur\ n - \#\langle P, \circ \rangle$, savukārt sistēmā RSA, atslēgu ģenerēšanas sarežģītība aptuveni novērtējama kā $O((logm)^3), kur\ m - divu\ lielu\ pirmskaitļu\ p\ un\ q\ reizinājums$ [7].

# 3. PERSONALIZĒTAS ELIPTISKO LĪKŅU KRIPTOSISTĒMAS IZSTRĀDE

Iepriekšējās nodaļās tika aplūkoti eliptisko līkņu kriptogrāfisku sistēmu fundamentālie matemātiskie un datorzinātņu pamati, kas nodrošina praktiskas implementācijas iespējas dažādos datu aizsardzības risinājumos.

Šajā nodaļā tiek pētītas praktiskas eliptisko līkņu kriptosistēmu izstrādes problēmas, iespējas un aprakstīts praktiski realizēts klienta - servera arhitektūras bāzēts datu aizsardzības risinājums, kas balstīts uz eliptisko līkņu kriptosistēmu.

Nodaļā ir septiņas sadaļas, iesākumā tiek definēti personalizētu kriptogrāfisku sistēmu izstrādes mērķi, tālāk dots vispārīgs ieskats publisko atslēgu kriptosistēmu konstruēšanas problemātikā, kam seko sistēmas domēna parametru izvēles ieteikumi un specifiskas klienta – servera arhitektūras un pamatalgoritmu apraksts. Nobeigumā sniegti komentāri par risinājuma implementāciju un eksperimentāli veiktspējas novērtējumi testu sistēmā pie dažādiem domēna parametriem.

## 3.1. Personalizētu eliptisko līkņu kriptosistēmu izstrādes mērķi

Praksē sastopams liels skaits dažādu komerciālu un bezmaksas datu aizsardzības un kriptogrāfijas risinājumu, kas bāzēti uz dažādām kriptogrāfiskām shēmām, tai skaitā eliptisko līkņu kriptogrāfiju. Viennozīmīgi, gatavu sistēmu izmantošana datu aizsardzībai samazina izstrādes un uzturēšanas izmaksas un laiku, tomēr šāda pieeja nepiedāvā pilnībā personalizējamu risinājumu ar pilnu kontroli pār kriptogrāfiskajām shēmām, to pamatalgoritmiem un sistēmas domēna parametriem.

Mūsdienu IT risinājumu neveiksme bieži vien ir saistīta ar nepietiekamu lietotāju datu aizsardzības politiku, kā rezultātā konfidenciāla informācija nonāk trešajām pusēm. Viens no veidiem, kā uzlabot lietotāju un klientu datu aizsardzību, ir atbilstošas, mūsdienīgas kriptogrāfiskās sistēmas izstrāde, implementācija un regulāra uzturēšana.

Ja datu aizsardzība un kriptogrāfija noteikta par kāda projekta prioritāti, kas būtiski ietekmē vai nosaka pasūtītāja biznesa sfēru un intereses, tad ieteicams izvērtēt personalizētas kriptosistēmas izstrādes iespējas. Lai gan šāda pieeja būtiski paaugstina projekta izmaksas un izstrādes laiku, korekti projektēta un implementēta personalizēta kriptogrāfiska sistēma nodrošina augstākus drošības, veiktspējas un uzturamības rādītājus, salīdzinot ar gataviem datu aizsardzības risinājumiem, jo atbilst specifiskām konkrētā projekta prasībām.

Šī darba ietvaros autors projektē personalizētu kriptogrāfisku sistēmu, sekojošu mērķu sasniegšanai:

- Personalizēts, mūsdienīgs datu aizsardzības risinājums, kas nodrošina kriptogrāfisko atslēgu apmaiņu un šifrētu saziņu internetā starp serveri un klientu gala iekārtam, izmantojot eliptisko līkņu kriptogrāfijas metodes.
- Kriptosistēma ir atdalīta no klienta un servera lietotņu biznesa loģikas, saskarnes un datiem.
- Pilna izstrādātāja kontrole pār kriptogrāfiskās sistēmas domēna parametriem.
- Efektīvi kriptogrāfiski algoritmi, kas nodrošina to darbību zema enerģijas patēriņa mobilajās iekārtās, izmantojot tīmekļa tehnoloģijas.
- Kriptosistēma ir neatkarīga no tīmekļa datu pārraides protokoliem un tīkla infrastruktūras.

## 3.2. Ieskats publiskās atslēgas kriptosistēmu konstruēšanā

Autors *A. Salomaa* savā grāmatā *Public Key Cryptography* [1] definē vairākus soļus, kas nosaka vispārīgas publiskās kriptosistēmas konstruēšanas posmus. Iesākumā nepieciešams atrast algoritmiski sarežģītu problēmu $P$, kurai nav zināmu polinomiāla laika algoritmu, bet eksistē apakšproblēma $P_{easy}$, kura ir atrisināma polinomiālā laikā. $P_{easy}$ jāsajauc tādā veidā, ka rezultējošā problēma $P_{shuffle}$ vairs nelīdzinās $P_{easy}$, bet gan pamatproblēmai $P$. Problēmu $P_{shuffle}$ pasludina par sistēmas publisko atslēgu un paziņo algoritmu, kā pielietot $P_{shuffle}$ informācijas šifrēšanai. Informācija, kā no $P_{easy}$ iegūts $P_{shuffle}$tiek paturēta noslēpumā un izmantota nošifrētās informācijas atšifrēšanai. Lai atšifrētu šifrēto informāciju, legālais saņēmējs risina algoritmiski vienkāršo $P_{easy}$, savukārt nesankcionēts saņēmējs risina algoritmiski sarežģīto $P_{shuffle}$.

Viegli pamanīt, ka eliptisko līkņu publiskās atslēgas kriptogrāfiskajās sistēmās, $P$ ir diskrētā logaritma problēma eliptiskajām līknēm (DLPEL). $P_{easy}$ - atrast diskrētos eksponentus iespējams ar polinomiāla laika algoritmiem. $P_{shuffle}$ - lai atrastu diskrētos logaritmus, jārisina $P$.

Ļoti svarīgi piebilst, ka publiskās atslēgas kriptogrāfiskās sistēmas fundamentāli nav beznosacījumu drošas [7]. Pēc definīcijas, publiskā atslēga kopā ar kriptosistēmas domēna parametriem satur informāciju par privāto atslēgu. Aplūkosim divas kriptosistēmu drošību ietekmējošas problēmas, kas rodas no publiskās atslēgas definīcijas.

Ja vispārīgā gadījumā, lai nesankcionēti atrastu privāto atslēgu, jārisina algoritmiski sarežģītā $P_{shuffle}$, tad bieži vien eksistē efektīvi algoritmi specifiskiem $P_{shuffle}$ gadījumiem. Tipisks piemērs, jau minētajām galīga pirmskaitļu lauka anomālām eliptiskām līknēm DLPEL

var tikt atrisināta polinomiālā laikā, neskatoties, ka vispārīgā gadījumā DLPEL ir eksponenciāla laika problēma.

Pieejamie skaitļošanas resursi nemitīgi palielinās. Ar ātru skaitļotāju var ievērojami samazināt laiku, kas nepieciešams, lai nesankcionēti atrastu kriptosistēmu privātās atslēgas. Īpaši būtiski, ja problēmu $P$ iespējams efektīvi risināt, izmantojot paralelizāciju. Atgādināšu, ka šobrīd labākais DLPEL risināšanas algoritms ir *Pollard* $\rho$-metode, ko var sekmīgi paralelizēt.

Konstruējot personalizētas kriptogrāfiskas sistēmas, nosaukto problēmu dēļ ir ļoti svarīgi pareizi izvēlēties konkrētās sistēmas domēna parametrus un nemitīgi atjaunot tos.

## 3.3. Domēna parametru korteža $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$ definēšana

Kā jau vairākkārt uzsvērs, viena no būtiskākajām problēmām, kura jārisina, projektējot jebkuru eliptisko līkņu bāzētu kriptogrāfisku sistēmu, ir sistēmas domēna parametru definēšana. Kā jau minēts sadaļā *Sistēmas domēna parametri*, par eliptiskās līknes bāzētas kriptogrāfiskas sistēmas domēna parametriem $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$ sauc sešinieku :

$D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = (p, a, b, G, n, h), kur\ p - nepāra\ pirmsskaitlis, a, b \in \mathbb{F}_\mathrm{p}, G \in \mathrm{E}(\mathbb{F}_\mathrm{p}), n = \#\langle G, \circ \rangle \in \mathbb{N}, h = \frac{\#\mathrm{E}(\mathbb{F}_\mathrm{p})}{n} \in \mathbb{N}$

Korteža komponenšu vērtības nosaka kriptogrāfiskās sistēmas fundamentālos atribūtus un īpašības, piemēram, galīgo lauku, eliptisko līkni, ciklisko apakšgrupu, atslēgu bitu garumu, atslēgu ģenerēšanas un apmaiņas algoritmu efektivitāti, noturīgumu pret uzbrukumiem utt. Nekorekta kriptosistēmas parametru izvēle nozīmē nedrošu un neefektīvu datu aizsardzību.

Projektējot eliptisko līkņu bāzētu kriptogrāfisku sistēmu ir divas iespējas, kā definēt sistēmas domēna parametrus: izvēlēties kādu no standartizētiem risinājumiem vai definēt parametrus patstāvīgi.

### 3.3.1. Standartizēti domēna parametri

Eksistē vairāki eliptisko līkņu domēna parametru standarti. Arvien pieaugošā eliptisko līkņu kriptosistēmu izmantošana komerciālos risinājumos sekmējusi risinājumu standartizāciju. Tādas globāli atzītas standartu organizācijas kā *ANSI* (*American National Standards Institute*), *IEEE* (*Institute of Electrical and Electronics Engineers*), *ISO* (*International Standards Organization*) un *NIST* (*National Institute of Standards and Technology)* standartizējusi eliptisko līkņu bāzētu kriptosistēmu domēna parametrus [22]. Turpmāk tiek aplūkoti NIST standartizētie domēna parametri. NIST piedāvā izvēlēties standartizētos kriptosistēmu domēnu parametrus atkarībā no nepieciešamā kriptogrāfisko atslēgu garuma bitos. Visos piedāvātajos

parametru kortežos, līknes parametrs $a = -3$. Šāda izvēle tiek skaidrota ar skalārā reizinājuma algoritmu veiktspējas uzlabojumiem *Jacobi* koordinātu sistēmā [22]. Tāpat visos standarta parametru kortežos, kofaktors $h = \frac{\#\mathrm{E}(\mathbb{F}_\mathrm{p})}{n} = 1$, kas nozīmē, ka $\#\mathrm{E}(\mathbb{F}_\mathrm{p}) = \#\langle G, \circ \rangle$, tātad: $\forall P \in \mathrm{E}(\mathbb{F}_\mathrm{p}) : \exists i \in \mathbb{N} : i \circ G = P$

NIST standartizējusi šādus pirmskaitļus (standarts FIPS 186-4), kas izmantojami domēna parametru kortežos [19],[23]:

- $p_{192} = 2^{192} - 2^{26} - 1$
- $p_{224} = 2^{224} - 2^{96} + 1$
- $p_{256} = 2^{256} - 2^{224} + 2^{192} + 2^{26} - 1$
- $p_{384} = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$
- $p_{521} = 2^{521} - 1$

Redzams, ka visi ieteiktie pirmskaitļi ir *Mersēna* pirmskaitļi, kas uzlabo kriptogrāfisko algoritmu ātrdarbību [23]. Tiek apgalvos [22], ka minētie pirmskaitļi izvēlēti, jo to veidoto lauku pakāpju bitu garumi ir vismaz divreiz lielāki kā populārāko simetrisko kriptogrāfisko sistēmu atslēgu bitu garumi. Eksistē apgalvojums [22], ka simetriskas k-bitu bloku kriptosistēmas atslēgas atrašana, izmantojot pilno pārlasi, prasa aptuveni tik pat daudz laika, cik atrisināt diskrētā logaritma problēmu eliptiskajām līknēm ar *Pollard rho* algoritmu pār $\mathbb{F}_\mathrm{p}, kuram\ \log_2 \#\mathbb{F}_\mathrm{p} \approx 2k$. Literatūras avotā [22] iespējams iepazīties ar salīdzinājumu starp populāro simetrisko kriptosistēmu (*Triple-DES,AES*) atslēgu bitu garumiem un atbilstošajām eliptisko līkņu kriptosistēmu pirmskaitļu lauku pakāpēm.

NIST standartā FIPS 186-4 [19] rekomendētie domēna parametri pieejami 1.pielikumā.

### 3.3.2. $\mathrm{E}(\mathbb{F}_\mathrm{p})$ Domēna parametru ģenerēšana

Kriptogrāfiskajās sistēmās lietoto domēna parametru kopu iespējams ģenerēt, neizmantojot standartizētos risinājumus. Parametru ģenerēšana saistās ar implementējamai kriptogrāfiskai sistēmai izvirzītajām funkcionālām un nefunkcionālām prasībām, galvenokārt ar realizējamo drošības līmeni attiecībā uz nepieciešamo laiku diskrētā logaritma problēmas eliptiskajām līknēm atrisināšanai konkrētā implementācijā. Tāpat būtiski ir novērtēt pieejamos skaitļošanas resursus un citus sistēmas ierobežojumus.

Literatūrā ir pieejami vairāki domēna parametru primitīvu ģenerēšanas algoritmi. Vispārīgajā gadījuma var rīkoties sekojoši [9]:

Fiksē nepieciešamo kriptogrāfiskās sistēmas drošības līmeni $t \in \{80,112,128,192,256\}$, ko turpmāk sauksim par *kriptogrāfiskās sistēmas atslēgas bitu garumu*. Ja kriptogrāfiskās sistēmas atslēgas bitu garums ir $t$ biti, tad diskrētā logaritma problēmas eliptiskajām līknēm atrisināšana prasa aptuveni $2^t$ operācijas [8], [9].

Nosaka galīgo pirmskaitļu lauku $\mathbb{F}_p$. Atbilstoši fiksētajam atslēgas bitu garumam $t$ izvēlas pirmskaitli $p$ šādā veidā:

$\lceil \log_2 p \rceil = 2t$, ja $t \in ]80; 256[$

$\lceil \log_2 p \rceil = 521$, ja $t = 256$

$\lceil \log_2 p \rceil = 192$, ja $t = 80$

Nosaka *Veierštrāsa* līknes parametrus $a, b$. Varbūtiski izvēlas veselus skaitļus $a, b$, kas atbilst līknes vienādojumam un singularitātes nosacījumam:

$$E: y^2 \equiv x^3 + ax + b \ (mod\ p) \wedge 4a^3 + 27b^2 \not\equiv 0 \ (mod\ p)$$

Atrod ciklisko apakšgrupu ģenerējošo līknes punktu $G$:

1. Izmantojot *Schoof'a* algoritmu [12] aprēķina $\#\,\mathrm{E}(\mathbb{F}_p)$.
2. Atrod maksimāli lielu $\#\,\mathrm{E}(\mathbb{F}_p)$ pirmreizinātāju $n$, kas, saskaņā ar *Lagranža* teorēmu[3], atbilst līknes punktu skaitam cikliskajā apakšgrupā.
3. Aprēķina kofaktoru $h = \frac{\#\mathrm{E}(\mathbb{F}_p)}{n}$.
4. Varbūtiski izvēlās līknes punktu $G \in \mathrm{E}(\mathbb{F}_p)$, aprēķina skalāro reizinājumu $n \circ G$.
5. Ja $n \circ G = \mathcal{O}$, tad $G$ ir ģenerators cikliskai grupai $\langle G, \circ \rangle$, kuras pakāpe $\#\langle G, \circ \rangle = n$.
6. Ja $n \circ G \neq \mathcal{O}$, tad atkārto 4.soli.

Verificē iegūtos domēna parametrus. Domēna parametrus noraida un iniciē jaunu domēna parametru ģenerēšanu šādos gadījumos:

1. $\#\,\mathrm{E}(\mathbb{F}_p) = \#\mathbb{F}_p$, pierādīts, ka šādām kriptogrāfiskām sistēmām diskrētā logaritma problēma atrisināma polinomiālā laikā, parādot izomorfismu starp $\mathrm{E}(\mathbb{F}_p)$ un $\mathbb{F}_p$ aditīvu grupu [8],[10].
2. $n|r, r - pirmskaitlis, r < 2^{160}$ [8]. Piemēram, šādas kriptosistēmas nav drošas pret uzbrukumiem, kas realizē *Pohlig-Hellman* metodi.

Certicom Research [9] papildu iesaka noraidīt domēna parametrus šādus gadījumus:

[3] Lagranža teorēma. Jebkurai galīgai grupai $G$ ar apakšgrupu $H$ : $\#G|\#H$

3. $h > 2^{\frac{t}{8}}$
4. $p^B \equiv 1(mod\ n), kur\ B \in [1; 100[$, Šādas kriptosistēmas nav drošas pret uzbrukumiem, kas realizē *Index Calculus* metodi.
5. Skaitļiem $(n+1) \wedge (n-1)$ eksistē nepietiekami liels pirmreizinātājs $r$, tāds kā:

$$\log_n r \leq \frac{19}{20}$$

Iespējams definēt implementācijas specifiskus ierobežojumus, skatīt nodaļu $\mathrm{E}(\mathbb{F}_\mathrm{p})$ *Domēna parametru optimizācija.*

### 3.3.3. $\mathrm{E}(\mathbb{F}_\mathrm{p})$ Domēna parametru optimizācija

Jebkura teorētiskas kriptogrāfiskas sistēmas implementācija praktiski tiek lietota ierobežotā vidē. Piemēram, procesora ātrdarbība, atmiņas apjoms, saziņas kanāla caurlaidība un trokšņu varbūtība kā arī citi ierobežojumi. Kā jau uzsvērts, viena no būtiskākajām eliptisko līkņu priekšrocībām, attiecībā pret industrijas *de facto* standartu, RSA ir būtiski mazāks kriptogrāfisko atslēgu izmērs, kas pieļauj eliptisko līkņu bāzētas sistēmas izmantot maza strāvas patēriņa ierīcēs, piemēram, viedkartēs. Šādi lietojumi prasa rūpīgu kriptosistēmu domēnu parametru izvēli. Literatūras sarakstā pieejama atsauce nr. 21. uz interesantu korporācijas *Sun Microsystems* pētījumu par eliptisko līkņu kriptosistēmu implementāciju iespējām datorsistēmās ar 8 bitu procesoriem.

Piedāvāju iepazīties ar vispārīgiem eliptisko līkņu bāzētu kriptosistēmu domēna parametru optimizācijas ieteikumiem, kas balstīti uz autora publikācijā [2] uzskaitītajām tipiskajām eliptisko līkņu kriptosistēmu implementācijas problēmām.

**$\mathbf{E(\mathbb{F}_p)}$ izvēle**. $\mathrm{E}(\mathbb{F}_\mathrm{p})$ ieteicams izvēlēties tā, ka $\#\,\mathrm{E}(\mathbb{F}_\mathrm{p})$ - pirmskaitlis. Saskaņā ar jau minēto *Lagranža* teorēmu, seko, ka cikliskās apakšgrupas $\langle G, \circ \rangle$ pakāpe $\#\langle G, \circ \rangle = \#\,\mathrm{E}(\mathbb{F}_\mathrm{p})$. Šādā situācijā tiek izmantota visa pieejamā eliptiskās līknes punktu kopa, kas uzlabo attiecību starp drošību un veiktspēju, kā arī būtiski atvieglo domēna parametru verificēšanu.

**Pirmskaitļa p izvēle $\mathbb{F}_\mathbf{p}$ definēšanai**. Pierādīts [21], ka Mersēna pirmskaitļu izmantošana uzlabo kriptosistēmas veiktspēju. Praktiskais ieguvums ir atkarīgs no konkrētās kriptogrāfiskās sistēmas implementācijas un datorsistēmas arhitektūras un saistīts ar mūsdienu procesoru uzbūvi, kas ļauj efektīvi veikt aritmētiskas operācijas ar divnieka pakāpēm.

**Koordinātu sistēmas izvēle.** Eliptiskās līknes punktu skalāro reizināšanu iespējams efektīvi realizēt projektīvā koordinātu sistēmā [22]. Lauka operācijas, tādas kā elementu

saskaitīšana un reizināšana, var tikt būtiski paātrinātas, pārejot uz projektīvām koordinātu sistēmām.

Standarta projektīvā koordinātu sistēma:

Afīnais punkts $\left(\frac{X}{Z}, \frac{Y}{Z}\right)$ atbilst projektīvajam punktam $(X:Y:Z), Z \neq 0$,

Eliptiskās līknes vienādojums $y^2 = x^3 + ax + b$ Dekarta koordinātu sistēmā ir ekvivalents eliptiskās līknes vienādojumam $y^2z = x^3 - 3xz^2 + bz^3$ standarta projektīvajā koordinātu sistēmā.

*Jacobi* projektīvā koordinātu sistēma:

Afīnais punkts $\left(\frac{X}{Z^2}, \frac{Y}{Z^3}\right)$ atbilst projektīvajam punktam $(X:Y:Z), Z \neq 0$,

Eliptiskās līknes vienādojums $y^2 = x^3 + ax + b$ Dekarta koordinātu sistēmā ir ekvivalents eliptiskās līknes vienādojumam $y^2z = x^3 - 3xz^4 + bz^6$ *Jacobi* projektīvajā koordinātu sistēmā.

Kā jau minēju, NIST standartā eliptisko līkņu koeficients $a = -3$, kas atļauj standartizētās līknes pielietot projektīvās koordinātu sistēmās.

Eliptiskās līknes punktu dubultošana *Jacobi* koordinātu sistēmā:

Dots eliptiskās līknes punkts $P = (X_1:Y_1:Z_1) \in \mathrm{E}(\mathbb{F}_\mathrm{p})$

$$2P = 2(X_1:Y_1:Z_1) = (X:Y_3:Z_3), kur:$$

$$A = 4X * Y_1^2, B = 8Y_1^4, C = 3(X_1 - Z_1^2)(X_1 + Z_1^2), D = -2A + C^2$$

$$X_3 = D, Y_3 = C * (A - D) - B, Z_3 = 2Y_1 * Z_1$$

Ja kriptosistēmas implementācija nosaka biežas operācijas, kas saistās ar punktu dubultošanu jeb skalāru reizināšanu ar pāra skaitli, tad, saskaņā ar pētījumu [22], efektīvākā koordinātu sistēma ir *Jacobi* koodinātu sistēma. Minētais pētījums parāda, ka efektīva divu dažādu eliptiskās līknes punktu saskaitīšana var tikt realizēta, izmantojot jauktas koordinātu sistēmas.

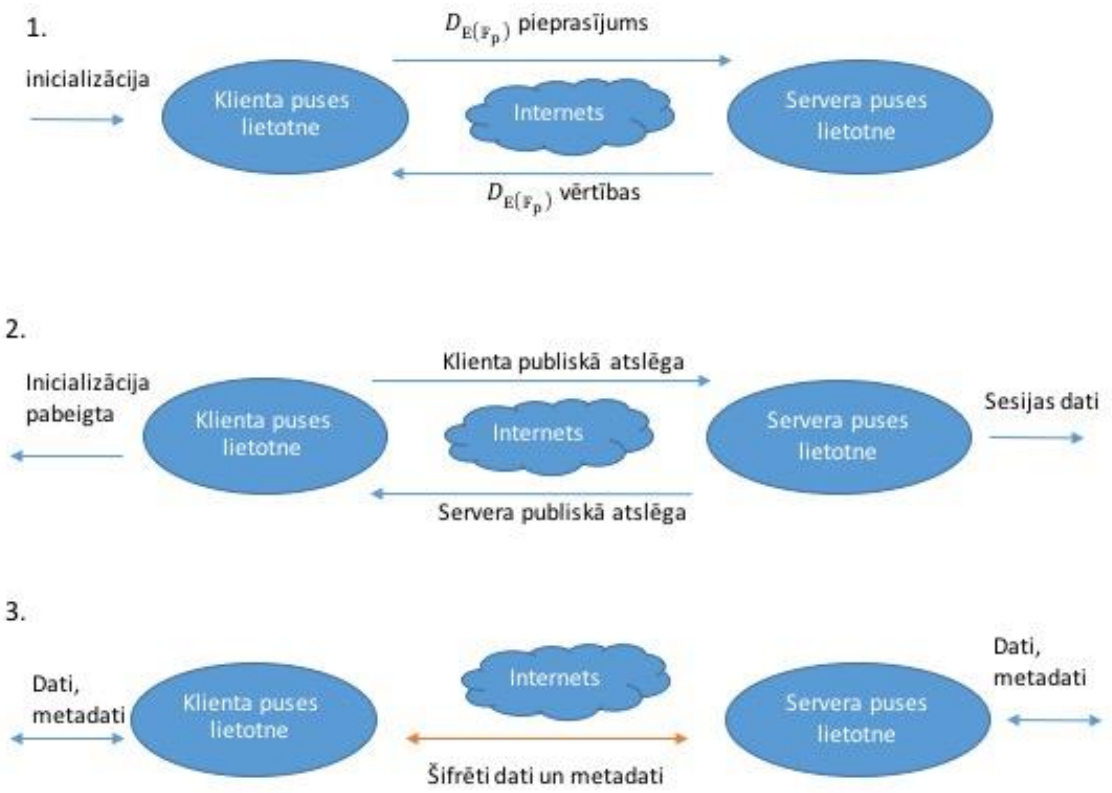
Var secināt, ka iespējams eksperimentāli optimizēt konkrētas kriptosistēmas operācijas, izmantojot dažādas koordinātu sistēmas.

## 3.4. Kriptogrāfiskās sistēmas vispārīga arhitektūra

Personalizēta kriptogrāfiska sistēma tiek izstrādāta atbilstoši klienta – servera arhitektūrai un nodrošina eliptisko līkņu kriptogrāfisko atslēgu un šifrētu teksta ziņojumu apmaiņu starp vienu serveri un vairākiem klientiem. Teksta ziņojumu šifrēšana notiek simetriski, par kriptogrāfisko atslēgu lietojot servera – klienta kopīgo noslēpumu, kas iegūts atslēgu apmaiņas procedūras laikā.

Izstrādātā kriptogrāfiskā sistēma sastāv no diviem komponentiem: klienta lietotne un servera lietotne. Komunikācija starp klienta lietotni un servera lietotni notiek nedrošā saziņas kanālā, ko tehniski nodrošina tīmekļa tehnoloģijas.

Shematisks savienojuma inicializācijas attēlojums redzams attēlā 3.4.1.




*att. 3.4.1* **Savienojuma inicializācija**

Uzsākot savienojumu, klienta puses lietotne sūta sistēmas domēna parametru pieprasījumu servera puses lietotnei. Kad saņemti kriptosistēmas domēna parametri, vienas komunikācijas sesijas ietvaros, tiek ģenerēti servera un klienta atslēgu primitīvi, kuri, pēc publisko atslēgu apmaiņas, tiek izmantoti, lai simetriski šifrētu sesijas ziņapmaiņu. Sistēma atbalsta patvaļīgus teksta ziņojumus alfabētā, ko definē *ASCII* kodu tabula.

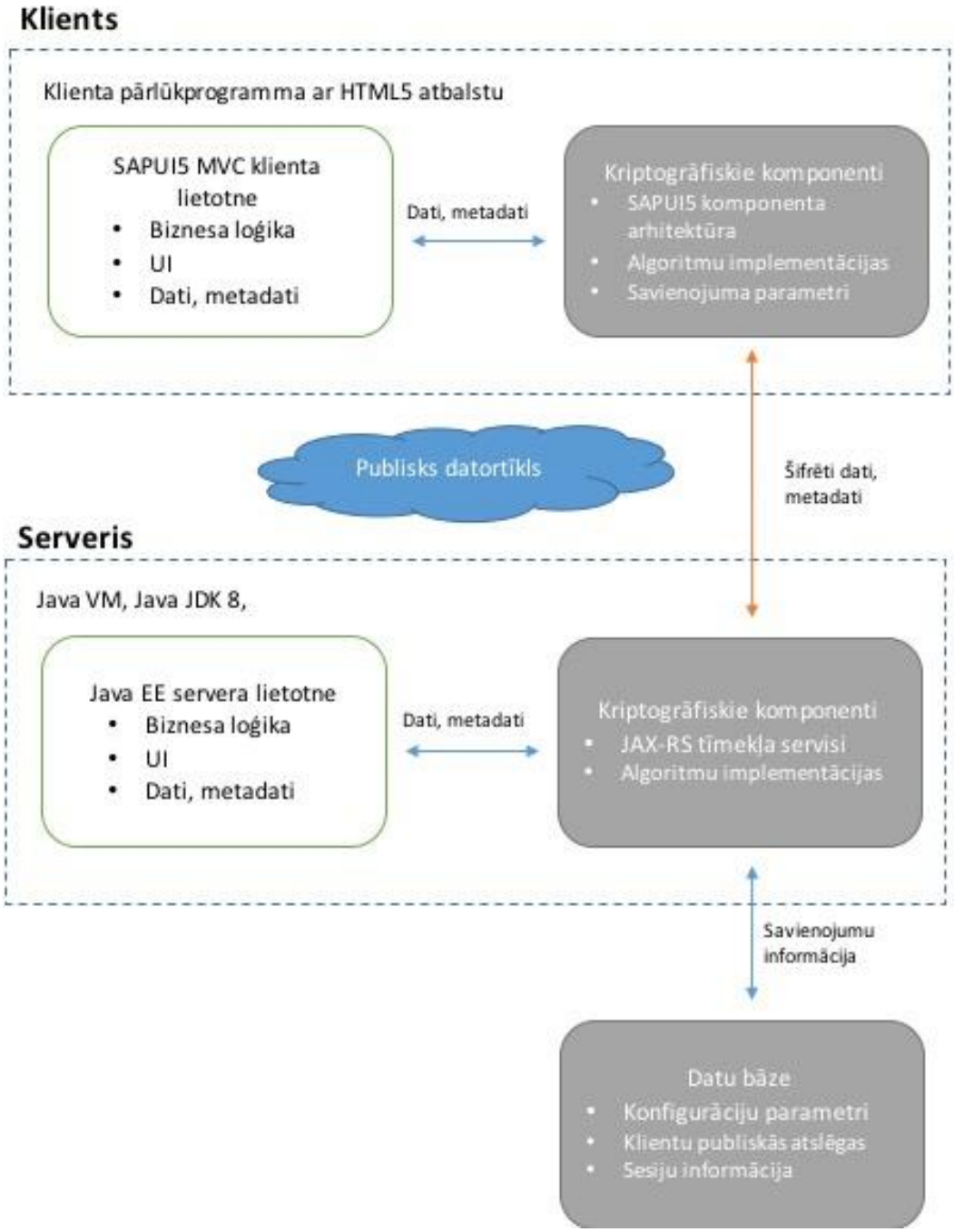
Klienta puses kriptogrāfiskie komponenti realizēti kā SAP biznesa lietotņu izstrādes ietvara *SAPUI5* modulis, kas nodrošina visas kriptogrāfiskās pamatoperācijas. Šifrēta ziņapmaiņa tehniski realizēta, izmantojot *REST* arhitektūrai atbilstošus tīmekļa servisus. Klienta puses kriptogrāfisko komponentu darbībai nepieciešama *HTML5* saderīga tīmekļa pārlūkprogramma ar ieslēgtu *JavaScript* koda izpildes atbalstu.

Servera puses kriptogrāfiskie komponenti realizēti kā Java EE (*Enterprise Edition*) lietotne, kas nodrošina *REST* pieprasījumu apstrādi, izmantojot *JAX-RS* implementāciju.

Klientu sesiju informācija, kas ietver saņemtās kriptogrāfiskās atslēgas un aktīvos sistēmas domēna parametrus, tiek glabātas relāciju datubāzē, kas pieejama, izmantojot drošu saziņas kanālu.

Servera puses lietotnes darbināšanai nepieciešams *Apache Tomcat* vai saderīgs tīmekļa servera risinājums un *Java* virtuālā mašīna.

Shematiski klienta un servera lietotņu komunikācija redzama attēlā 3.4.2.




*att. 3.4.2* **Kriptogrāfiskās sistēmas arhitektūra**

## 3.5. Kriptogrāfiskās sistēmas algoritmi

Personalizēta kriptogrāfiskā sistēma klientu – servera saziņai sastāv no klienta puses un servera puses kriptogrāfiskajiem komponentiem, kuriem ir kopīgi kriptogrāfiskie pamatalgoritmi, kas nosaka domēna parametru verifikācijas, atslēgu ģenerēšanas un apmaiņas procedūru un citas kopīgas operācijas.

Servera lietotnes specifiski kriptogrāfiski algoritmi ir domēna parametru ģenerēšana, atbilstoši prasītajam atslēgas bitu garumam un savienojuma pieprasījumu apstrāde.

Klienta lietotnes specifiski kriptogrāfiski algoritmi ir atslēgu apmaiņas procedūras inicializācija.

Servera un klienta lietotņu implementācijas realizējamas fundamentāli dažādās tehnoloģijās. Algoritmu aprakstā netiek ņemtas vērā platformas specifiskas prasības, piemēram, prasības aritmētiskajām operācijām ar veseliem skaitļiem, kuri binārajā pierakstā pārsniedz 64 bitus.

$\mathrm{E}(\mathbb{F}_p)$ punktu grupas saskaitīšanas un skalārās reizināšanas operācijas algoritmiski realizētas, saskaņā ar operāciju definīciju nodaļās 1.4.2 un 1.4.3.

Sadaļā aprakstītā informācija nav uztverama kā formāla izstrādājamās sistēmas prasību specifikācija, bet gan kā vispārīga informācija par sistēmā realizējamiem kriptogrāfiskajiem pamatalgoritmiem, kas izstrādāti, balstoties uz šajā darbā aprakstītajiem rezultātiem.

### 3.5.1. Kriptogrāfiskie pamatalgoritmi

Šajā nodaļā specificēti kriptogrāfiskie pamatalgoritmi, kas nodrošina izstrādājamās kriptogrāfiskās sistēmas darbību: domēna parametru verifikācija, atslēgu primitīvu ģenerēšana un verificēšana, publisko atslēgu apmaiņa, ziņojumu šifrēšana un atšifrēšana.

#### Domēna parametru verifikācijas algoritms *isValidDomainParameters*

Algoritms verificē domēna parametru kortežu $D_{\mathrm{E}(\mathbb{F}_p)}$.

Ievaddati:

- Domēna parametri $D_{\mathrm{E}(\mathbb{F}_p)} = (p, a, b, G, n, h)$
- $t \in \{80,112,128,192,256\}$ – aptuvenais atslēgas bitu garums

Izvaddati :

- $Response \in \{true, false\}$, ja $Response = true$, domēna parametri $D_{\mathrm{E}(\mathbb{F}_p)}$ ir derīgi, citādi – domēna parametri $D_{\mathrm{E}(\mathbb{F}_p)}$ nav derīgi

Algoritma apraksts:

$$f: (D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}, t) \rightarrow \{true, false\}$$

1. Izmantojot *Schoof'a* algoritmu [12] aprēķina $\#\mathrm{E}(\mathbb{F}_\mathrm{p})$
2. Pārbauda sekojošus nosacījumus
   a. $G \neq \mathcal{O} = (x_G, y_G)$
   b. $\mathrm{n} \circ G = \mathcal{O}$
   c. $y_G^2 \equiv x_G^3 + x_G a + b \ (mod\ p)$
   d. $4a^3 + 27b^2 \not\equiv 0 \ (mod\ p)$
   e. $\#\mathrm{E}(\mathbb{F}_\mathrm{p}) \neq \mathrm{p}$
   f. Ja $t \in ]80; 256[ : \lceil \log_2 p \rceil = 2t$
   g. Ja $t = 256 : \lceil \log_2 p \rceil = 521$
   h. Ja $t = 80 : \lceil \log_2 p \rceil = 192$
   i. $n | r, r - pirmskaitlis, r \geq 2^{160}$
   j. $h \leq 2^{\frac{t}{8}}$
   k. $p^B \not\equiv 1 (mod\ n), kur\ B \in [1; 100[$
   l. $(n+1) \wedge (n-1) | r, r - pirmskaitlis, tāds\ ka\ \log_n r > \frac{19}{20}$
3. Ja kāds no nosacījumiem a.-l. ir aplams, $Response = false$, citādi $Response = true$

Kriptogrāfisko atslēgu ģenerēšanas algoritms *generateKeyPair*

Algoritms ģenerē kriptogrāfisko atslēgu pāri $(k, Q)$, kur $k$ sistēmas entītijas privātā atslēga, $Q$ – sistēmas entītijas publiskā atslēga.

Ievaddati:

- Domēna parametri $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = (p, a, b, G, n, h)$

Izvaddati :

- $keyPair = (k, Q)$

Algoritma apraksts:

$$f\left(D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right) = (k, Q)$$

1. Varbūtiski izvēlas veselu skaitli $k$ no intervāla $[1, n-1]\ kur\ n \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
2. $Q = k \circ G, kur\ G \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
3. Ja is$ValidPublicKey\left(Q, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right) = false$, pāriet uz 1. soli.
4. Atgriež $keyPair = (k, Q)$

Publiskās atslēgas verifikācijas algoritms *isValidPublicKey*

Algoritms verificē publisko atslēgu attiecībā pret sesijas domēna parametriem.

Ievaddati:

- $Q$ – publiskā atslēga
- Domēna parametri $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = (p, a, b, G, n, h)$

Izvaddati :

- $Response \in \{true, false\}$, ja $Response = true$, publiskā atslēga $Q$ ir derīga, citādi – $Q$ nav derīga publiskā atslēga

Algoritma apraksts:

$$f: (Q, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}) \rightarrow \{true, false\}$$

1. Pārbauda sekojošus nosacījumus
   a. $Q \neq \mathcal{O} = (x_Q, y_Q), kur\ \mathcal{O} \in \mathrm{E}(\mathbb{F}_\mathrm{p})$
   b. $n \circ Q = \mathcal{O}, kur\ n \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})},\ \mathcal{O} \in \mathrm{E}(\mathbb{F}_\mathrm{p})$
   c. $y_Q^2 \equiv x_Q^3 + x_Q a + b\ (mod\ p), kur\ p, a, b \in D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
2. Ja kāds no nosacījumiem a.-c. ir aplams, $Response = false$, citādi $Response = true$

Atslēgu apmaiņas algoritms *getSharedSecret*

Algoritms aprēķina divu sistēmas entītiju kopīgo noslēpumu, kas tiek lietots kā simetriska datu šifrēšanas risinājuma atslēga.

Ievaddati:

- $Q_A$ – entītijas $A$ publiskā atslēga
- $k_B$– entītijas $B$ privātā atslēga
- Domēna parametri $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = (p, a, b, G, n, h)$

Izvaddati :

- Kopīgais noslēpums $S \in \mathbb{F}_\mathrm{p}$

Kļūdas:

- *nederīga entītijas A publiskā atslēga*
- *nederīgi sesijas dati*

Algoritma apraksts:

$$f: (Q_A, k_B, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}) \rightarrow \mathbb{F}_\mathrm{p}$$

1. Ja is$ValidPublicKey\left(Q_A, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right) = false$, beidz darbu ar kļūdu : *nederīga entītijas A publiskā atslēga*
2. Aprēķina $Q_S = k_B \circ Q_A$
3. Verificē $Q_S$: $Ja\ Q_S = \mathcal{O}$, beidz darbu ar kļūdu : *nederīgi sesijas dati*
4. $Q_S = (x_{Q_S}, y_{Q_S})$: $S = x_{Q_S}$
5. Atgriež $S$

Teksta šifrēšanas algoritms *encryptMessage*

Metode simetriski šifrē tekstu, izmantojot sūtītāja entītijas un saņēmēja entītijas kopīgo noslēpumu $S$ un simetrisku kriptogrāfisku funkciju $f_{encrypt}$, kas šeit netiek sīkāk specificēta.

Ievaddati:

- $m$ – šifrējamais teksts
- $S$ – sūtītāja un saņēmēja kopīgais noslēpums
- $f_{encrypt}$ – simetriska teksta šifrēšanas funkcija

Izvaddati :

- $s$ – šifrēts teksts $m$

Kļūdas:

- $f_{encrypt}$ *kļūda*

Algoritma apraksts:

$$f(m, S, f_{encrypt}) = \mathrm{s}$$

1. $\mathrm{s} = f_{encrypt}(m, S)$,
    a. Ja $f_{encrypt}$ atgriež kļūdu, beidz darbu ar kļūdu : $f_{encrypt}$ *kļūda*
2. Atgriež s

Teksta atšifrēšanas algoritms *decryptMessage*

Metode simetriski atšifrē kriptotekstu $s$, izmantojot sūtītāja entītijas un saņēmēja entītijas kopīgo noslēpumu $S$ un kriptogrāfisku funkciju $f_{decrypt}$, kas šeit netiek sīkāk specificēta.

Ievaddati:

- $s$ – šifrēts teksts $m$
- $S$ – sūtītāja un saņēmēja kopīgais noslēpums
- $f_{decrypt}$ – simetriska teksta šifrēšanas funkcija

Izvaddati :

- $m$ – atšifrēts kriptoteksts $s$

Kļūdas:

- $f_{decrypt}$ *kļūda*

Algoritma apraksts:

$$f(s, S, f_{decrypt}) = \text{t}$$

1. $\text{m} = f_{decrypt}(s, S)$,
    a. Ja $f_{decrypt}$ atgriež kļūdu, beidz darbu ar kļūdu : $f_{decrypt}$ *kļūda*
2. Atgriež m

### 3.5.2. Servera lietotnes algoritmi

Servera lietotne nodrošina varbūtisku domēna parametru ģenerēšanu, domēna parametru un sesijas pieprasījumu apstrādi. Servera lietotne uztur datu bāzi ar aktīvo klientu sesiju informāciju un domēna parametru kortežiem, kas atbilst dažādiem atslēgu bitu garumiem. Vienas sesijas ietvaros serveris un klients apstiprina kopīgus sistēmas domēna parametrus un ģenerē atslēgu primitīvus. Sesijai ir ierobežots derīguma termiņš, pēc kura sesija ir jāatjauno.

#### Domēna parametru ģenerēšanas algoritms *generateDomainParameters*

Algoritms ģenerē kriptosistēmas domēna parametrus, atbilstoši nepieciešamajam drošības līmenim attiecībā pret atslēgas bitu garumu.

Ievaddati:

- $t \in \{80,112,128,192,256\}$ – aptuvenais atslēgas bitu garums

Izvaddati :

- $D_{\text{E}(\mathbb{F}_\text{p})} = (p, a, b, G, n, h)$ – sistēmas domēna parametri

Algoritma apraksts:

$$f: \{80,112,128,192,256\} \rightarrow D_{\text{E}(\mathbb{F}_\text{p})}$$

1. Varbūtiski izvēlas pirmskaitli $p$ atkarībā no $t$ sekojošā veidā:
    a. $\lceil \log_2 p \rceil = 2t$, ja $t \in ]80; 256[$
    b. $\lceil \log_2 p \rceil = 521$, ja $t = 256$
    c. $\lceil \log_2 p \rceil = 192$, ja $t = 80$
2. No lauka $\mathbb{F}_\text{p}$ varbūtiski izvēlās koeficientus $a, b$
3. Izmantojot *Schoof'a* algoritmu [12] aprēķina $\#\,\text{E}(\mathbb{F}_\text{p})$
4. Atrod maksimāli lielu $\#\,\text{E}(\mathbb{F}_\text{p})$ pirmreizinātāju $n$
5. $h = \frac{\#\text{E}(\mathbb{F}_\text{p})}{n}$

6. Varbūtiski izvēlās līknes punktu $G \in \mathrm{E}(\mathbb{F}_\mathrm{p})$, aprēķina skalāro reizinājumu $n \circ G$
7. Ja $n \circ G \neq \mathcal{O}$, atkārto 6.punktu
8. Ja $isValidDomainParameters\left(D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}, t\right) = false$, pāriet uz 1. soli.
9. Atgriež $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$

Domēna parametru iestatīšanas algoritms *setDomainParameters*

Algoritms iestata kriptosistēmā lietoto domēna parametru kortežu $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$ atbilstoši atslēgas garumam $t$.

Ievaddati:

- $t \in \{80,112,128,192,256\}$ – aptuvenais atslēgas bitu garums
- Domēna parametri $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = (p, a, b, G, n, h)$ vai $\emptyset$

Izvaddati :

- $(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$

Kļūdas:

- *nederīgi domēna parametri*

Algoritma apraksts:

1. Ja $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} \neq \emptyset \; un \; isValidDomainParameters\left(D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}, t\right) = false$, beidz darbu ar kļūdu : *nederīgi domēna parametri*
2. Ja $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = \emptyset$:
    a. $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = generateDomainParameters(t)$
3. Saglabā datubāzē un atgriež $(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$

Domēna parametru pieprasījuma apstrādes algoritms *onParametersRequested*

Algoritms klientam $C$ laika momentā $d$ nosūta aptuvenajam atslēgas bitu garumam $t$ atbilstošo domēna parametru kortežu $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$. Ja pieprasījuma izpildes brīdī klientam $C$ ir aktīvs sesijas ieraksts, tas tiek dzēsts.

Ievaddati:

- $C$ – klienta identifikators
- $t \in \{80,112,128,192,256\}$ – aptuvenais atslēgu bitu garums

Izvaddati:

- Domēna parametri $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})} = (p, a, b, G, n, h)$

Kļūdas:

- *komunikācijas kļūda*

Algoritma apraksts:

$$f: \{80,112,128,192,256\} \rightarrow D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$$

1. Sistēmas datubāzē meklē $(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
    a. Ja atslēgas garumam t atbilst vairāki domēna parametru korteži, varbūtiski izvēlas vienu no tiem
2. Ja $\left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right) = \emptyset$
    a. $\left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right) = setDomainParameters(t, \emptyset)$
3. Laika momentā $d$ klientam $C$ nosūta $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
    a. Ja ziņojumu nosūtīt neizdevās, beidz darbu ar kļūdu : *komunikācijas kļūda*
4. Ja datubāzē eksistē pagaidu domēna parametru pieprasījuma ieraksts $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), d_1\right)$, to dzēš.
5. Ja datubāzē eksistē klienta $C$ sesijas ieraksts, to dzēš.
6. Datubāzē izveido domēna parametru pieprasījuma pagaidu ierakstu $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), d\right)$

Sesijas pieprasījuma apstrādes algoritms *onSessionRequested*

Algoritms apstrādā klienta $C$, ar publisko atslēgu $Q_C$, sesijas pieprasījumu. Ja atslēga $Q_C$ derīga, serveris izveido datu bāzē sesijas ierakstu ar derīguma termiņu $d$ un nosūta klientam $C$ servera publisko atslēgu $Q_S$ un atslēgas derīguma termiņu $d$ .

Ievaddati:

- $C$ - klienta identifikators
- $Q_C$- klienta $C$ publiskā atslēga

Izvaddati:

- $Q_s$ - servera publiskā atslēga
- $d \in \mathbb{N}_1$ – sesijas derīguma termiņš

Kļūdas:

- *domēna parametri nav atrasti*
- *nederīga klienta $C$ publiskā atslēga $Q_C$*
- *komunikācijas kļūda*

Algoritma apraksts:

$$f(C, Q_C) = (d, Q_s)$$

1. Datubāzē meklē domēna parametru pieprasījuma pagaidu ierakstu $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), d_1\right)$
    a. Ja$\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), d_1\right) = \emptyset$, beidz darbu ar kļūdu : *domēna parametri nav atrasti*
2. Ja is$ValidPublicKey\left(Q_C, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right) = false$, beidz darbu ar kļūdu : *nederīga klienta $C$ publiskā atslēga $Q_C$*
3. Ģenerē servera entītijas atslēgu pāri:
    a. $(k_S, Q_S) = generateKeyPair(D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
4. Laika momentā $d_2$ Klientam $C$ nosūta $(d_2, Q_s)$
    a. Ja ziņojumu nosūtīt neizdevās, beidz darbu ar kļūdu : *komunikācijas kļūda*
5. No datubāzes dzēš domēna parametru pieprasījuma pagaidu ierakstu $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), d_1\right)$
6. Datubāzē izveido sesijas ierakstu $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), Q_C, (k_S, Q_S), d_2\right)$

### Ziņojuma nosūtīšanas algoritms *sendMessageToClient*

Algoritms sūta klienta $C$ puses saņēmējam $Receiver$ šifrētu ziņojumu $s$.

Ievaddati:

- $C$ - klienta identifikators
- $Receiver$ - klienta puses saņēmēja identifikators ASCII alfabētā
- $m$ - ziņojuma teksts
- $Time$ - laika moments
- $f_{encrypt}$ – simetriska teksta šifrēšanas funkcija

Izvaddati:

- $\emptyset$

Kļūdas:

- *Komunikācijas kļūda*
- *Šifrēšanas kļūda*
- *Sesijas kļūda*

Algoritma apraksts:

1. Datu bāzē atrod sesijas ierakstu $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), Q_C, (k_S, Q_S), d\right)$
    a. Ja $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), Q_C, (k_S, Q_S), d\right) = \emptyset$, beidz darbu ar kļūdu : *Sesijas kļūda*

2. Ja $d < Time$, beidz darbu ar kļūdu : *Sesijas kļūda*
3. $S = getSharedSecret(k_S, Q_C, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
4. $s = encryptMessage(m, S, f_{encrypt}), Receiver' = encryptMessage(Receiver, S, f_{encrypt})$,
    a. ja $encryptMessage$ atgriež kļūdu, beidz darbu ar kļūdu : *Šifrēšanas kļūda*
5. Klientam $C$ sūta ziņojumu $(s, Receiver')$.
    a. Ja ziņojumu nosūtīt neizdevās, beidz darbu ar kļūdu : *Komunikācijas kļūda*

Ziņojuma saņemšanas algoritms *receiveMessageFromClient*

Algoritms saņem no klienta $C$ šifrētu ziņojumu $s$, atšifrē un paziņo servera puses sistēmai atšifrēto ziņojumu $m$.

Ievaddati:

- $C$ – klienta identifikators
- $s$ - ziņojuma m kriptoteksts
- $Time$ - laika moments
- $f_{decrypt}$ – simetriska teksta atšifrēšanas funkcija

Izvaddati:

- $\emptyset$

Kļūdas:

- *Atšifrēšanas kļūda*
- *Sesijas kļūda*

Algoritma apraksts:

1. Datu bāzē atrod sesijas ierakstu $\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), Q_C, (k_S, Q_S), d\right)$
    a. Ja$\left(C, \left(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right), Q_C, (k_S, Q_S), d\right) = \emptyset$, beidz darbu ar kļūdu : *Sesijas kļūda*
2. Ja $d < Time$, beidz darbu ar kļūdu : *Sesijas kļūda*

3. $S = getSharedSecret(k_s, Q_C, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
4. $m = decryptMessage(s, S, f_{decrypt}),\ Receiver' = decryptMessage(Receiver, S, f_{decrypt})$
    a. ja $decryptMessage$ atgriež kļūdu, beidz darbu ar kļūdu : *Šifrēšanas kļūda*
5. Paziņo $(C, m)$

### 3.5.3. Klienta puses algoritmi

Klienta puses lietotne nodrošina saziņas inicializāciju un sesijas pieprasījumu, kā arī ziņapmaiņu ar serveri.

#### Domēna parametru pieprasījuma algoritms *requestParameters*

Algoritms sūta serverim $Serv$ domēna parametru ar atslēgu bitu garumu $t$, pieprasījumu un atbildē gaida atbilstošo domēna parametru kortežu.

Ievaddati:

- $Serv$ - servera identifikators
- $t \in \{80,112,128,192,256\}$ – aptuvenais atslēgas bitu garums

Izvaddati:

- Domēna parametri $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$

Kļūdas:

- *Komunikācijas kļūda*
- *nederīgi domēna parametri* $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$

Algoritma apraksts:

1. Serverim $Serv$ sūta atslēgas bitu garumam $t$ atbilstošo domēna parametru pieprasījumu.
    a. Ja ziņojumu nosūtīt neizdevās, beidz darbu ar kļūdu : *komunikācijas kļūda*
2. Laika brīdi $r$ gaida servera $Serv$ atbildi ar atbilstošo $(\mathrm{t}, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
    a. Ja atbilde netika saņemta, beidz darbu ar kļūdu : *komunikācijas kļūda*
3. Ja $isValidDomainParameters\left(D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}, t\right) = false$, beidz darbu ar kļūdu : *nederīgi domēna parametri*
4. Ģenerē klienta puses atslēgu primitīvus:
    a. $(k_C, Q_C) = generateKeyPair(D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$

5. lokālajā atmiņā saglabā $((k_C, Q_C), D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
6. $requestSession(Serv\ , Q_C)$
7. Atgriež $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$

Sesijas pieprasījuma algoritms *requestSession*

Algoritms sūta serverim $Serv$ sesijas pieprasījumu, kas satur klienta publisko atslēgu $Q_C$ un atbildē gaida servera $Serv$ publisko atslēgu $Q_S$ un sesijas derīguma termiņu $d$.

Ievaddati:

- $Serv$ - servera identifikators
- $Q_C$ - klienta publiskā atslēga

Izvaddati:

- $(d, Q_s)$ - servera publiskā atslēga un tās derīguma termiņš

Kļūdas:

- *Komunikācijas kļūda*
- *Nederīga servera publiskā atslēga* $Q_s$

Algoritma apraksts:

1. Serverim $Serv$ sūta klienta publisko atslēgu $Q_C$.
    a. Ja ziņojumu nosūtīt neizdevās, beidz darbu ar kļūdu : *komunikācijas kļūda*
2. Laika brīdi $r$ gaida servera $Serv$ atbildi ar atbilstošo servera publisko atslēgu $Q_s$ un tās derīguma termiņu $d$
    a. Ja atbilde netika saņemta, beidz darbu ar kļūdu : *komunikācijas kļūda*
3. Lokālajā atmiņā atrod $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
4. Ja is$ValidPublicKey\left(Q_s, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}\right) = false$, beidz darbu ar kļūdu : *nederīga servera* $Serv$ *publiskā atslēga* $Q_s$
5. Atgriež un lokālajā atmiņā saglabā $(d, Q_s)$

Ziņojuma nosūtīšanas algoritms *sendMessageToServer*

Algoritms sūta serverim $Serv$ šifrētu ziņojumu $s$.

Ievaddati:

- $Serv$ - servera identifikators
- $m$ - ziņojuma teksts
- $(Q_S, d)$ - servera publiskā atslēga un derīguma termiņš

- $k_C$- klienta privātā atslēga
- $Time$ - laika moments
- $f_{encrypt}$ – simetriska teksta šifrēšanas funkcija

Izvaddati:

- $\emptyset$

Kļūdas:

- *Komunikācijas kļūda*
- *Šifrēšanas kļūda*
- *Nederīga servera publiskā atslēga* $Q_s$

Algoritma apraksts:

1. Ja $d < Time$, beidz darbu ar kļūdu : *Nederīga servera publiskā atslēga* $Q_s$
2. Lokālajā atmiņā atrod $D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
3. $S = getSharedSecret(k_C, Q_S, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
4. $s = encryptMessage(m, S, f_{encrypt})$
    a. ja $encryptMessage$ atgriež kļūdu, beidz darbu ar kļūdu : *Šifrēšanas kļūda*
5. Serverim $Serv$ sūta ziņojumu $s$.
    a. Ja ziņojumu nosūtīt neizdevās, beidz darbu ar kļūdu : *Komunikācijas kļūda*

### Ziņojuma saņemšanas algoritms *receiveMessageFromServer*

Algoritms saņem no servera $Serv$ šifrētu ziņojumu $s$, atšifrē un nodod atšifrēto ziņojumu $m$ saņēmējam $Receiver$.

Ievaddati:

- $s$ - ziņojuma m kriptoteksts
- $Time$ - laika moments
- $Receiver$ – šifrēts saņēmēja identifikators
- $f_{decrypt}$ – simetriska teksta atšifrēšanas funkcija

Izvaddati:

- $\emptyset$

Kļūdas:

- *Nederīga servera publiskā atslēga* $Q_s$
- *Atšifrēšanas kļūda*

Algoritma apraksts:

1. Lokālajā atmiņā atrod $(Q_S, d), (k_C, Q_C), D_{\mathrm{E}(\mathbb{F}_\mathrm{p})}$
2. Ja $d < Time$, beidz darbu ar kļūdu : *Nederīga servera publiskā atslēga* $Q_S$
3. $S = getSharedSecret(k_C, Q_S, D_{\mathrm{E}(\mathbb{F}_\mathrm{p})})$
4. $m = decryptMessage(s, S, f_{decrypt}),\ Receiver' = decryptMessage(Receiver, S, f_{decrypt})$
    a. ja $decryptMessage$ atgriež kļūdu, beidz darbu ar kļūdu : *Šifrēšanas kļūda*
5. Saņēmējam $Receiver'$ paziņo $m$

## 3.6. Kriptogrāfiskās sistēmas references implementācija

Kā jau ievadā minēts, balstoties uz šajā darbā aprakstītajiem rezultātiem un sadarbojoties ar autoru, personalizētu kriptogrāfisku datu aizsardzības risinājumu implementēja uzņēmuma *Exigen Services Latvia* praktikants un Latvijas Universitātes Datorikas fakultātes students Oļegs Pešudovs. Izstrādātā implementācija un tās dokumentācija tiks aizstāvēta kā kvalifikācijas darbs programmētāja kvalifikācijas iegūšanai.

Personalizētas eliptisko līkņu bāzētas kriptogrāfiskas sistēmas references implementācija realizēta kā klienta un servera puses programmatūras izstrādes ietvari, kas nodrošina kriptogrāfisko atslēgu apmaiņu un patvaļīgu teksta ziņojumu šifrēšanu un atšifrēšanu.

Izstrādātā kriptogrāfiskā sistēma nodrošina abstrakcijas slāni starp visas sistēmas biznesa loģiku, lietotāja saskarni, datiem un kriptogrāfiskajiem komponentiem. Implementēto risinājumu ir viegli integrēt dažādos klienta – servera risinājumos, kur nepieciešama publisko atslēgu apmaiņa un šifrēta saziņa internetā starp dažādiem sistēmu gala punktiem.

References implementācijas būtiskākās īpašības:

- Nodrošina publisko atslēgu apmaiņu un šifrētu saziņu internetā starp uzņēmuma serveri un lietotāju gala iekārtām, izmantojot eliptisko līkņu bāzētu kriptogrāfisku shēmu.
- Kriptogrāfiskās sistēmas implementācija nodrošina vieglu sistēmas domēna parametru iestatīšanu.
- Risinājums nodrošina papildu abstrakcijas slāni starp lietotāju datiem un internetā nosūtīto informāciju gan klienta pusē, gan servera pusē.
- Internetā pārsūtītā informācija tiek šifrēta, izmantojot izstrādāto risinājumu, neatkarīgi no tā vai tiek lietots HTTP vai HTTPS protokols.

- Izstrādātā kriptogrāfiskā sistēma vienā datu sūtījumā ļauj šifrētā veidā pārsūtīt līdz *5000* ASCII simbolu garu ziņojumu.
- Servera pusē risinājums atbalsta *Java EE* izstrādes tehnoloģiju un tā darbībai nepieciešams *Apache Tomcat* tīmekļa serveris un *Java* virtuālā mašīna.
- Klienta pusē risinājums atbalsta *JavaScript* lietotņu izstrādi, izmantojot *SAPUI5* ietvaru un to darbināšanu *HTML5* saderīgā tīmekļa pārlūkprogrammā.
- Klienta gala iekārtas var būt zema enerģijas patēriņa ierīces ar ierobežotu atmiņu un procesora veiktspēju.
- Pielietotās tehnoloģijas ļauj viegli lietot implementēto datu aizsardzības risinājumu dažādās aparatūras un programmatūras vidēs.
- Risinājumā ir iebūvēti veiktspējas kontroles rīki, kas ļauj novērtēt konkrētu kriptosistēmas domēna parametru efektivitāti mērķa sistēmās.

Darbs pie references implementācijas izstrādes un pilnveidošanas tiek turpināts.

## 3.7. Eksperimentāli veiktspējas novērtējumi

Lai aptuveni novērtētu izstrādātās kriptogrāfiskās sistēmas klienta puses un servera puses komponentu veiktspēju pie dažādiem domēna parametriem, tika mērīts atslēgu ģenerēšanas algoritmu izpildes laiks references implementācijā un izdarītas statistiskās hipotēzes par attiecīgo komponentu veiktspēju pie dotiem domēna parametriem.

Klienta puses un servera puses atslēgu primitīvu ģenerēšanas algoritmi tika izpildīti testu sistēmā ar konfigurācijas parametriem, kas doti 3.7.1 tabulā.

*3.7.1 tabula*

**Testu datorsistēmas konfigurācijas parametri**

| | |
|---|---|
| Procesors | *Intel(R) Core(TM) i5-4460 CPU @ 3.20GHz* |
| Operatīvā atmiņa | *16,0 GB* |
| Operētājsistēma | *Microsoft Windows 10 (build 10240), 64-bit* |
| Java | *1.7* |
| Tīmekļa serveris | *Apache Tomcat 7.0* |
| Pārlūkprogramma | *Google Chrome Version 50.0.2661.102 m* |
| SAP ietvars SAPUI5 | *1.36.9* |

Testos tika izmantoti 5 dažādi domēna parametru korteži, kas uzrādīti 1.pielikumā. Domēna parametri atbilst NIST standartā *FIPS 186-4* definētajiem eliptisko līkņu kriptosistēmu drošības līmeņiem $t \in \{80,112,128,192,256\}$ [19]. Kā jau minēts, ja atslēgas garums ir $t$ biti, tad diskrētā logaritma problēmas eliptiskajām līknēm atrisināšana prasa aptuveni $2^t$ operācijas.

Atslēgu primitīvu ģenerēšanas algoritma darbības laiks $\Delta T$ tika mērīts, fiksējot sistēmas laiku pirms algoritma izpildes $T_1$ un pēc algoritma izpildes $T_2$

$$\Delta T = T_2 - T_1$$

$T_1$ un $T_2$ klienta puses sistēmā tika fiksēti, izmantojot valodā *JavaScript* iebūvēto *Date* klases metodi *Date.getTime()*. Servera puses sistēmā $T_1$ un $T_2$ tika fiksēti izmantojot valodā *Java* iebūvēto *System* klases metodi *System.nanoTime()*.

Katram 1.pielikumā uzrādītajam domēna parametru kortežam tika mērīts klienta puses un servera puses atslēgu ģenerēšanas algoritmu darbības laiks 24 neatkarīgos mēģinājumos, kuru rezultāti uzrādīti 2.pielikumā. Visi laika mērījumi tika konvertēti milisekundēs ar precizitāti līdz 0 zīmēm aiz komata.

Pamatojoties uz veiktspējas eksperimenta laikā savāktajiem datiem, ar programmu *IBM SPSS Statistics 22* tika veikta *t-testa* statistikas analīze, ar mērķi noteikt klienta puses un servera puses komponentu vidējo atslēgu ģenerēšanai nepieciešamo laiku testa sistēmā, pie dotiem kriptosistēmas domēna parametram.

Rezultātā, pie būtiskuma līmeņa $\alpha = 0.05$, ar 95% pārliecību nevar noliegt hipotēzes, ka testa sistēmā, references implementācijas klienta puses un servera puses komponenti, kriptogrāfiskos atslēgu primitīvus, atbilstoši domēna parametriem $T-80, \ldots, T-256$ vidēji ģenerē laikos, kas uzrādīti tabulā 3.7.2. *T-testa* statistikas analīzes rezultāti uzrādīti 3.pielikumā.

*3.7.2 tabula*

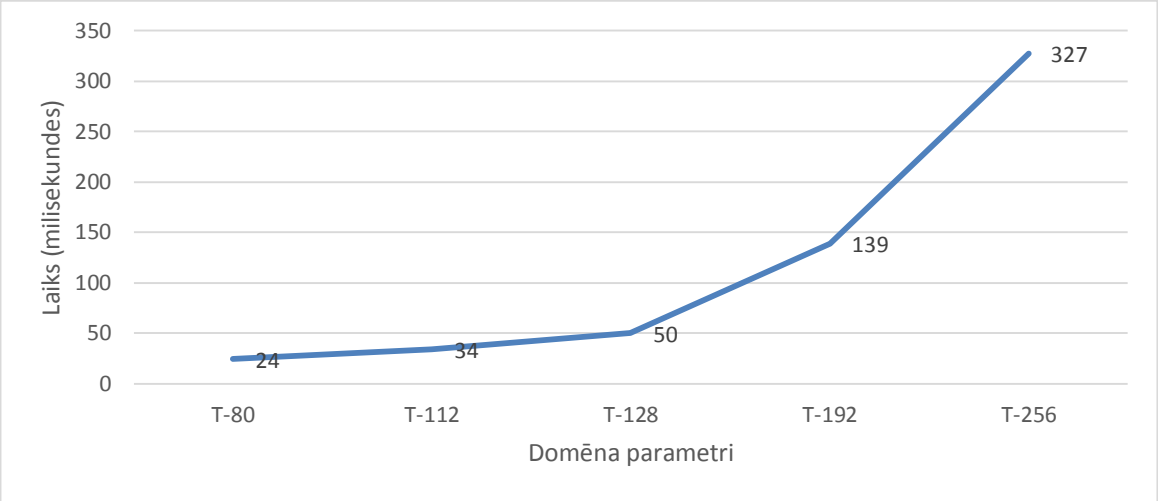
**Atslēgu primitīvu ģenerēšanai nepieciešamais laiks milisekundēs**

| T-80 | | T-112 | | T-128 | | T-192 | | T-256 | |
|---|---|---|---|---|---|---|---|---|---|
| Klients | Serveris | Klients | Serveris | Klients | Serveris | Klients | Serveris | Klients | Serveris |
| **24** | **6.5** | **34** | **7** | **50** | **8** | **139** | **12.5** | **327** | **17.5** |

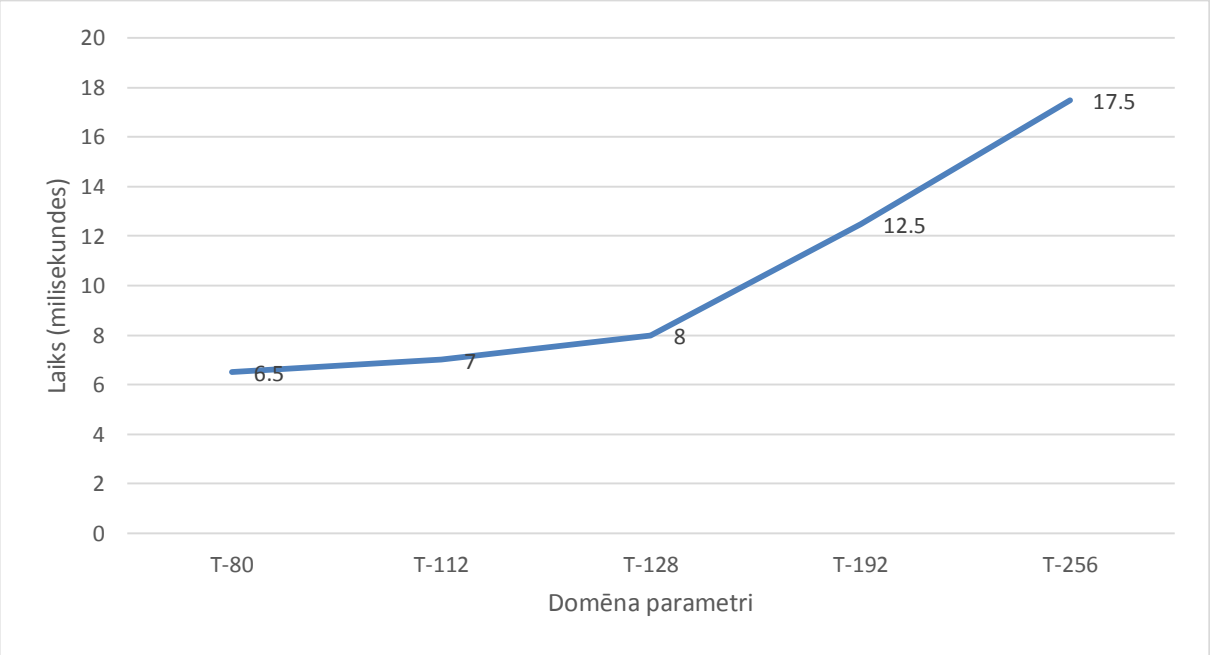
Veiktspējas eksperimenta rezultāti parāda statistiski nozīmīgu atšķirību starp klienta puses un servera puses atslēgu primitīvu ģenerēšanas algoritmu implementāciju darbības laiku pie vienādiem kriptosistēmas domēna parametriem.

Attēlos 3.7.1 un 3.7.2 var iepazīties ar veiktspējas eksperimentā iegūto sakarību starp dažādiem drošības līmeņiem atbilstošiem kriptosistēmas domēna parametriem un nepieciešamo laiku atslēgu ģenerēšanai servera un klienta puses kriptogrāfiskajos komponentos.

No grafikiem viegli pamanīt, ka klienta puses atslēgu ģenerēšanas algoritma implementācija, salīdzinot ar servera puses attiecīgo algoritmu, strādā 4-18 reizes lēnāk. Šāds rezultāts izskaidrojams ar fundamentāli atšķirīgajām tehnoloģijām, kas lietotas servera un klienta puses kriptogrāfisko komponentu realizācijā un darbināšanā.




*att. 3.7.1* **Klienta puses atslēgu primitīvu ģenerēšanas algoritma darbības laiks testu sistēmā**




*att. 3.7.2* **Servera puses atslēgu primitīvu ģenerēšanas algoritma darbības laiks testu sistēmā**

Veiktspējas eksperiments parāda, ka references implementācija ir praktiski darbināma ar mūsdienu drošības standartiem atbilstošiem kriptogrāfiskajiem domēna parametriem un nodrošina efektīvu kriptogrāfisko pamatalgoritmu izpildi klienta un servera puses komponentos. Var secināt, ka nepieciešami papildu klienta puses implementācijas optimizācijas iespēju pētījumi.

# REZULTĀTI

Darba ietvaros tika pētīti eliptisko līkņu asinhronās kriptogrāfijas fundamentālie matemātiskie un datorzinātņu pamati, kā arī tika izstrādāts viegli personalizējams, platformu un tīkla protokolu neatkarīgs, praktiski pielietojams datu aizsardzības risinājums, kas nodrošina efektīvu kriptogrāfisko atslēgu apmaiņu un šifrētu saziņu internetā starp serveri un klientiem.

Eliptisko līkņu kriptogrāfiskas sistēmas balstās uz algoritmiski sarežģīto diskrētā logaritma problēmu eliptiskajām līknēm (DLPEL). Labākie zināmie un praktiski pielietojamie DLPEL algoritmi, tādi kā *Pollard ρ*-metode, strādā eksponenciālā laikā. Salīdzinoši, šobrīd IT industrijā populārā RSA kriptosistēma balstās uz skaitļu faktorizācijas problēmu, kurai eksistē sub-eksponenciāla laika algoritmi, piemēram *General Number Field Sieve* algoritms. Šāda starpība zināmo algoritmu darba laika sarežģītībā nodrošina, ka pie vienāda drošības līmeņa, eliptisko līkņu kriptosistēmu atslēgu bitu garumi ir ievērojami mazāki kā RSA kriptosistēmas atslēgu bitu garumi. Mazāka izmēra atslēgas un relatīvi vienkāršas pamatoperācijas nodrošina būtiski ātrāku kriptosistēmu algoritmu izpildi, kas pieļauj implementēt drošas un efektīvas eliptisko līkņu kriptosistēmas zema enerģijas patēriņa ierīcēs.

Darba ietvaros liela nozīme tika pievērsta eliptisko līkņu kriptosistēmu domēna parametru izvēlei un verifikācijai. Drošas un efektīvas kriptogrāfiskas sistēmas pamats ir pareizi izvēlēti domēna parametri. Implementācijās iespējams izmantot gan standartizētus eliptisko līkņu kriptosistēmu domēna parametrus, gan arī tos var varbūtiski ģenerēt un verificēt. Darbā tika analizētas abas minētās iespējas.

Izstrādātās kriptogrāfiskās sistēmas pamatalgoritmi realizē *Diffie-Hellman* atslēgu apmaiņas protokolu eliptiskajām līknēm un sinhronu teksta šifrēšanas shēmu, kuras darbību nodrošina atslēgu apmaiņas rezultātā iegūts kopīgs noslēpums. Izstrādātā kriptogrāfiskās sistēmas arhitektūra paredz divus komponentus – klienta puses sistēmu un servera puses sistēmu, kuru savstarpējo komunikāciju internetā nodrošina tīmekļa tehnoloģijas. Minētie komponenti relatīvi vienkārši integrējami esošās tīmekļa bāzētās klienta – servera arhitektūras biznesa sistēmās, kas izmanto *SAPUI5* un *Java JAX-RS* tehnoloģijas, kā arī nodrošina pilnu izstrādātāja kontroli pār kriptosistēmas domēna parametriem.

Sadarbībā ar autoru, izstrādātās kriptogrāfiskās sistēmas references implementāciju realizēja un dokumentēja LU Datorikas fakultātes students sava kvalifikācijas darba ietvaros.

Tika veikti eksperimentāli references implementācijas kriptogrāfisko algoritmu veiktspējas novērtējumi, kuri ļauj izdarīt secinājumu, ka izstrādātā kriptogrāfiskā sistēma ir praktiski pielietojama datu aizsardzībai, izmantojot drošus kriptosistēmas domēna parametrus.

# SECINĀJUMI

Var secināt, ka eliptisko līkņu kriptogrāfiskās sistēmas ir relatīvi vienkārši un efektīvi implementējamas datu aizsardzības risinājumos, kas nodrošina informācijas apmaiņu nedrošos saziņas kanālos starp klienta – servera arhitektūras sistēmu komponentiem.

Salīdzinājumā ar RSA kriptosistēmu, eliptisko līkņu kriptosistēmu fundamentālie matemātiskie un datorzinātņu pamati ir relatīvi komplicētāki un mazāk pētīti, tomēr var secināt, ka mūsdienās eliptisko līkņu kriptosistēmas ir droša un efektīva alternatīva. Eliptisko līkņu kriptogrāfiskas sistēmas sekmīgi tiek pielietotas reālos liela mēroga IT risinājumos, tādos kā *Bitcoin*, *Austrijas e-ID* sistēma, *SSH* tīkla protokols un citos.

Triviāls, tomēr fundamentāls secinājums saistās ar eliptisko līkņu kriptosistēmu implementāciju drošību, kas ir tiešā veidā saistīta ar izvēlētajiem kriptosistēmas domēna parametriem. Eksistē nedrošu eliptisko līkņu klase, kura tiek regulāri atjaunota, pamatojoties uz uzlabojumiem eliptisko līkņu kriptogrāfijas metodoloģijā un pieejamo skaitļošanas jaudu.

Lai nodrošinātu kriptosistēmas drošību, regulāri jāatjauno sistēmas domēna parametri un tiem atbilstošās kriptogrāfiskās atslēgas. Var pielietot standartizētus domēna parametrus, tomēr šādā gadījumā risinājuma drošība saistās ar standarta izstrādātāju atbildību un mērķiem.

Jāpiebilst, ka reālos drošības risinājumos pielietojamu, mūsdienu drošības prasībām atbilstošu domēna parametru varbūtiska ģenerēšana un verifikācija saistās ar lielu skaitļošanas resursu patēriņu.

Izstrādājot datu aizsardzības risinājumu, tika secināts, ka reālo sistēmas drošību ietekmē daudz faktori, tādi kā izmantotās tehnoloģijas, aparatūra, programmatūra un cilvēciskais faktors. Turklāt varbūtība, ka risinājuma drošības barjera tiek apieta, atrisinot kriptogrāfiskās sistēmas fundamentālo problēmu, ir mazāka kā, piemēram, cilvēciskā faktora un privāto atslēgu fiziskas drošības negatīva ietekme.

Tika novērtos, ka izstrādātās sistēmas kriptogrāfisko pamatalgoritmu veiktspēja ir saistīta ar pielietotajām tehnoloģijām un izdarīti secinājumi, ka nepieciešamai tālāk pētījumi komponentu veiktspējas uzlabošanai. Īpašu interesi izraisa eliptiskās līknes punktu kompresijas iespējas un algoritmu optimizācijas iespējas, kas nodrošina aritmētiskas operācijas ar liela izmēra (192 – 521 biti) veseliem skaitļiem valodā *JavaScript*.

Nobeigumā var minēt, ka visi sākotnēji izvirzītie mērķi tika sasniegti. Bez šī pētījuma, par eliptisko līkņu kriptogrāfijas tēmu tika uzrakstīts un aizstāvēts autora kursa darbs *Eliptisko līkņu kriptogrāfija*, publicēts autora raksts *Applications and Benefits of Elliptic Curve Cryptografy,* kā arī sadarbībā ar autoru uzrakstīts kvalifikācijas darbs *Eliptiskās līknes bāzēta kriptogrāfiska ietvara izstrāde*.

# PATEICĪBAS

*Autors vēlas izteikt cieņpilnu pateicību darba vadītājiem profesoram Rūsiņam Mārtiņam Freivaldam un* profesoram Andrim Ambainim *par ieguldīto laiku* un darbu.

*Tāpat autors vēlas izteikt pateicību ģimenei un kolēģiem, par atbalstu un sapratni darba tapšanas laikā.*

# IZMANTOTĀ LITERATŪRA UN AVOTI


1. Salomaa A., *Public-Key Cryptography*: Springer, 1996
2. Magons, K.: Applications and Benefits of Elliptic Curve Cryptography, CEUR Workshop Proceedings, vol. 1548, pp. 32 -42, Harrachov, 2016
3. Szendrői B., *Cubic curves: a short survey*, Department of Mathematics, University of Utrecht, The Netherlands, 2005
4. Garrett P., *Discriminant of cubics*, University of Minnesota, 2013. [tiešsaiste]. [atsauce 05.20.2016.]. Pieejams internetā: http://www.math.umn.edu/~garrett/m/mfms/notes_2013-14/03b_cubic_discriminant.pdf
5. Blinn J.F., *How to Solve a Cubic Equation*. In: "*IEEE Computer Graphics and Application*, Los Alamitos, 2006, p.84–93
6. Milne J.S., *Elliptic Curves*. Princeton University Press, 1980
7. Stinson D. R., *Cryptography Theory And Practice. 3th edition*, Chapman & Hall/CRC, New York, 2006
8. Koblitz N., Menezes A., Vanstone S., *The State of Elliptic Curve Cryptography*. In: "*Towards a Quarter-Century of Public Key Cryptography*", Kluwer Academic Publishers, Boston, 2000, p.173–193
9. Brown D. R. L., *SEC 1: Elliptic Curve Cryptography*. Certicom Corp, 2009
10. Silverman H.S., *An Introduction to the Theory of Elliptic Curves.* University of Wyoming. 2006
11. McGee J. J., *Elliptic Curve Algorithms in Mathematica*, Wolfram Technology Conference 2007, Champaign, 2007
12. Schoof R., *Elliptic Curves Over Finite Fields and the Computation of Square Roots.* Mathematics of Computation 44, 483-494, Stor. Apr.,1985
13. *Elliptic Curve Cryptography: Elliptic Curve Cryptography: finite fields and discrete logarithms* [tiešsaiste]. [atsauce 20.05.2016.]. Pieejams internetā: http://andrea.corbellini.name/2015/05/23/elliptic-curve-cryptography-finite-fields-and-discrete-logarithms/
14. Rivain M., *Fast and Regular Algorithms for Scalar Multiplication over Elliptic Curves*, CryptoExperts, 2011 [tiešsaiste]. [atsauce 05.04.2016.]. Pieejams internetā: https://eprint.iacr.org/2011/338.pdf
15. Robinson J.S.D., *An Introduction to Abstract Algebra*. Walter de Gruyer GmbH & Co, 2003

16. Odlyzko A. , *Discrete logarithms: The past and the future*. In: "*Towards a Quarter-Century of Public Key Cryptography*", Kluwer Academic Publishers, Boston, 2000, p.129-145
17. Maurer U. M., Wolf S., *The Diffie–Hellman Protocol*. In: "*Towards a Quarter-Century of Public Key Cryptography*", Kluwer Academic Publishers, Boston, 2000, p.147-171
18. Giuliani K.J., *Attacks on the Elliptic Curve Discrete Logarithm Problem*. [tiešsaiste]. [atsauce 09.05.2016.]. Pieejams internetā: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.23.3890&rep=rep1&type=pdf
19. *Digital Signature Standard (DSA) FIPS PUB 186-4*, National Institute of Standards and Technology, Gaithersburg, 2013.
20. Arrendondo B., Jansma N., *Performance Comparison of Elliptic Curve and RSA Digital Signatures.* [tiešsaiste]. [atsauce 20.05.2016.]. Pieejams internetā: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.129.7139&rep=rep1&type=pdf
21. Gura N.,Patel A., Wander A., Eberle H., Shantz S. C., *Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs*. [tiešsaiste]. [atsauce 20.05.2016.]. Pieejams internetā: https://www.iacr.org/archive/ches2004/31560117/31560117.pdf
22. Brown M., Hankerson D., Lopez J., Menezes A. *Software Implementation of the NIST Elliptic Curves Over Prime Fields*. [tiešsaiste]. [atsauce 20.05.2016.]. Pieejams internetā: https://choucroutage.com/Papers/SideChannelAttacks/ctrsa-2011-brown.pdf
23. Bos J.W., Halderman J.A., Heninger N., Moore J., Naehrig M., Wustrow E., *Elliptic Curve Cryptography in Practice*. [tiešsaiste]. [atsauce 20.05.2016.]. Pieejams internetā: https://eprint.iacr.org/2013/734.pdf

# PIELIKUMI

1. pielikums.

NIST FIPS 186-4 domēna parametri

**T-80**

*p* 627710173538668076383578942320766641608390870039032496127 9

*n* 627710173538668076383578942317605901376719477318284228408 1

*a* 627710173538668076383578942320766641608390870039032496127 6

*b* 245515554600894381774029391519745178476910805816119123806 5

*Gx* 602046282375688656758213480587526111916698976636884684818

*Gy* 174050332293622031404857552280219410364023488927386650641

*h* 1

**T-112**

*p* 269599466671506397946670150870196306735579162600263081435100662988 81

*n* 269599466671506397946670150870196259404578077144243917216827223680 61

*a* 269599466671506397946670150870196306735579162600263081435100662988 78

*b* 189582862855666080004086685444939264155046809686793210757872346725 64

*Gx* 192779291135662930711103080346994880268319342194524401566497843520 33

*Gy* 199268087580344709701979743708887491842059919906039495376373431987 72

*h* 1

**T-128**

*p* 115792089210356248762697446949407573530086143415290314195533631308867097 8\
53951

*n* 115792089210356248762697446949407573529996955224135760342422259061068512 0\
44369

*a* 115792089210356248762697446949407573530086143415290314195533631308867097 8\
53948

*b* 410583637251521421293261297800472684091144410159937255548352563140394674 0\
1291

*Gx* 484395612939064517590525852527979142027629495260417479958440807170824046 3\
5286

*Gy* 361342509567497957985851279195878819566111066729850150718771982535684144 0\
5109

*h* 1

**T-192**

*p* 3940200619639447921227904010014361380507973927046544666794829340424572177147\
9687032904726608825893800186160697311231 9

*n* 3940200619639447921227904010014361380507973927046544666794690527962765939917\
1326356939895630815229491355443365394264 3

*a* 3940200619639447921227904010014361380507973927046544666794829340424572177147\
9687032904726608825893800186160697311231 6

*b* 2758019355995970587784901184038904809305690585636156852142870730198868924137\
0986086513626076488374510776543976123057 5

*Gx* 2624703509579968926862315674456698189185292349110921338781561590092551885477\
3805008902238805397571978665087247673208 7

*Gy* 8325710961489029985546751289520108179287853048861315594709205902480503199887\
4419224438643760392947333078086511627871

*h* 1

**T-256**

*p* 6864797660130609714981900799081393217269435300143305409394463459185543183397\
7656052122559640661454554977296311391480858037121987999716643812574028291117\
5057151

*n* 6864797660130609714981900799081393217269435300143305409394463459185543183397\
7655394245057746333217197532963996371363321113864768612440380340372808892707\
7005449

*a* 6864797660130609714981900799081393217269435300143305409394463459185543183397\
7656052122559640661454554977296311391480858037121987999716643812574028291117\
5057148

*b* 1093849038073734274511112390766805569936207598951683748994586394495953116157\
0735016013708737573759623248592132296706313309438452531591012912142327488478\
8985984

*Gx* 2661740802050217063228768716723360960729859168756973147706671368418802944997\
6427808491545080627771902352094241225065558662157113545570916814161637315895\
5999846

*Gy* 3757180025770020463545507224491183603594455134769762486694567779615544477447\
0556316691234405012945539562144444537289428522585666729196580810124344277578\
8376784

*h* 1

2. pielikums.

Veiktspējas mērījumu rezultāti

**Atslēgu primitīvu ģenerēšanas laika mērījumu rezultāti (milisekundes)**

| | T-80 | | T-112 | | T-128 | | T-192 | | T-256 | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Npk** | K | S | K | S | K | S | K | S | K | S |
| 1 | 25 | 6 | 33 | 7 | 54 | 9 | 150 | 13 | 347 | 17 |
| 2 | 23 | 6 | 32 | 8 | 48 | 8 | 135 | 12 | 323 | 17 |
| 3 | 24 | 6 | 33 | 7 | 50 | 8 | 136 | 13 | 321 | 18 |
| 4 | 24 | 7 | 33 | 7 | 50 | 8 | 139 | 13 | 327 | 18 |
| 5 | 23 | 6 | 32 | 8 | 47 | 8 | 133 | 13 | 319 | 19 |
| 6 | 23 | 7 | 34 | 7 | 52 | 8 | 139 | 13 | 315 | 18 |
| 7 | 22 | 6 | 32 | 7 | 51 | 8 | 134 | 12 | 312 | 18 |
| 8 | 24 | 7 | 31 | 7 | 50 | 7 | 146 | 14 | 319 | 18 |
| 9 | 24 | 8 | 33 | 6 | 50 | 8 | 145 | 14 | 332 | 17 |
| 10 | 24 | 6 | 34 | 7 | 48 | 8 | 138 | 13 | 324 | 17 |
| 11 | 24 | 6 | 32 | 7 | 51 | 8 | 134 | 13 | 325 | 17 |
| 12 | 24 | 7 | 31 | 7 | 51 | 7 | 134 | 12 | 322 | 17 |
| 13 | 24 | 6 | 32 | 7 | 50 | 8 | 142 | 12 | 331 | 17 |
| 14 | 25 | 7 | 31 | 7 | 50 | 7 | 135 | 12 | 322 | 17 |
| 15 | 23 | 6 | 30 | 8 | 52 | 8 | 133 | 13 | 326 | 18 |
| 16 | 24 | 6 | 32 | 7 | 49 | 8 | 134 | 12 | 340 | 17 |
| 17 | 25 | 7 | 33 | 7 | 51 | 8 | 132 | 12 | 373 | 18 |
| 18 | 23 | 7 | 33 | 7 | 49 | 8 | 138 | 12 | 328 | 19 |
| 19 | 23 | 7 | 33 | 8 | 51 | 8 | 133 | 12 | 323 | 18 |
| 20 | 24 | 8 | 33 | 7 | 50 | 8 | 135 | 12 | 337 | 18 |
| 21 | 23 | 6 | 41 | 7 | 48 | 9 | 149 | 12 | 317 | 18 |
| 22 | 24 | 6 | 40 | 7 | 53 | 8 | 161 | 13 | 313 | 18 |
| 23 | 24 | 6 | 40 | 7 | 47 | 8 | 145 | 13 | 313 | 17 |
| 24 | 25 | 6 | 39 | 7 | 48 | 8 | 138 | 12 | 340 | 18 |

K – Klienta puses sistēma

S – Servera puses sistēma

3. pielikums.
Veiktspējas mērījumu *t-testa*
statistika

**Hipotēzes pārbaudes rezultāti par klienta puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem $T - 80$**

| | Test Value = 24 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-80 Atslēgu primitīvu ģenerēšanas laiks klienta sistēmā | -1.310 | 23 | .203 | -.208 | -.54 | .12 |

**Hipotēzes pārbaudes rezultāti par servera puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem $T - 80$**

| | Test Value = 6.5 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-80 Atslēgu primitīvu ģenerēšanas laiks servera sistēmā | .000 | 23 | 1.000 | .000 | -.28 | .28 |

**Hipotēzes pārbaudes rezultāti par klienta puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem T − 112**

| | Test Value = 34 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-112 Atslēgu primitīvu ģenerēšanas laiks klienta sistēmā | -.597 | 23 | .556 | -.375 | -1.67 | .92 |

**Hipotēzes pārbaudes rezultāti par servera puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem T − 112**

| | Test Value = 7 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-112 Atslēgu primitīvu ģenerēšanas laiks servera sistēmā | 1.366 | 23 | .185 | .125 | -.06 | .31 |

**Hipotēzes pārbaudes rezultāti par klienta puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem T − 128**

| | Test Value = 50 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-128 Atslēgu primitīvu ģenerēšanas laiks klienta sistēmā | .000 | 23 | 1.000 | .000 | -.76 | .76 |

**Hipotēzes pārbaudes rezultāti par servera puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem $T - 128$**

| | Test Value = 8 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-128 Atslēgu primitīvu ģenerēšanas laiks servera sistēmā | -.440 | 23 | .664 | -.042 | -.24 | .15 |

**Hipotēzes pārbaudes rezultāti par klienta puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem $T - 192$**

| | Test Value = 139 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-192 Atslēgu primitīvu ģenerēšanas laiks klienta sistēmā | .058 | 23 | .954 | .083 | -2.90 | 3.07 |

**Hipotēzes pārbaudes rezultāti par servera puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem $T - 192$**

| | Test Value = 12.5 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-192 Atslēgu primitīvu ģenerēšanas laiks servera sistēmā | .624 | 23 | .539 | .083 | -.19 | .36 |

**Hipotēzes pārbaudes rezultāti par klienta puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem $T-256$**

| | Test Value = 327 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-256 Atslēgu primitīvu ģenerēšanas laiks klienta sistēmā | .015 | 23 | .988 | .042 | -5.61 | 5.69 |

**Hipotēzes pārbaudes rezultāti par servera puses sistēmas vidējo atslēgu primitīvu ģenerēšanas laiku testa datorsistēmā pie domēna parametriem $T-256$**

| | Test Value = 17.5 | | | | | |
|---|---|---|---|---|---|---|
| | t | df | Sig. (2-tailed) | Mean Difference | 95% Confidence Interval of the Difference | |
| | | | | | Lower | Upper |
| T-256 Atslēgu primitīvu ģenerēšanas laiks servera sistēmā | 1.282 | 23 | .213 | .167 | -.10 | .44 |

Bakalaura darbs „Eliptisko līkņu kriptosistēmu pielietojumi” izstrādāts LU Datorikas fakultātē.

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: _____ Krists Magons

Rekomendēju/nerekomendēju darbu aizstāvēšanai

Vadītājs: profesors Dr. dat. Andris Ambainis

_____ ___.05.2016.

Recenzents: profesors Dr. dat. Juris Vīksna

Darbs iesniegts Datorikas fakultātē ___.05.2016.

Dekāna pilnvarotā persona: _____

Darbs aizstāvēts bakalaura gala pārbaudījuma komisijas sēdē

_____

Komisijas sekretāre: _____