IVXV auditirakenduse tööpõhimõte

Kokkuvõte

E-hääletamise raamistikus IVXV on oluline roll välisel andmeaudiitoril, kes kontrollib miksimistõendi ja lugemistõendi korrektsust ning veendub seeläbi, et elektroonilise hääletamise tulemus on arvutatud korrektselt. Dokumendis anname detailse ülevaate IVXV krüptoprotokollis defineeritud tõestuste verifitseerimisest, võimaldamaks välisel osapoolel ise auditirakendust luua.

Dokument ei käsitle individuaalset verifitseeritavust, hääle allkirjastamist ega registreerimisteenust.

Sisukord

Peatükk 1

Sõltumatu osapoole poolt auditeeritav e-hääletamine

E-hääletamise süsteemis IVXV saadab valija hääle e-urni krüpteeritult, kasutades avaliku võtme krüptograafiat. Hääl on krüpteeritud e-urni avaliku võtmega, krüptogrammil on konkreetse valija digitaalallkiri. Hääli talletatakse hääletamisperioodi vältel digitaalallkirjastatuna, mis lihtsustab e-urni tervikluse tagamist.

Anonüümimine on vahekiht häälte talletamise ja dekrüpteerimise vahel, mis peidab krüpteeritud hääle ja selle saatja vahelise seose. Anonüümimine on vajalik samm hääle salajasuse kaitsmiseks, mis peab leidma aset koostoimes hääletamistulemuse korrektsuse tõestamisega välisele osapoolele – triviaalsed meetodid ei tööta.

Verificatum[1] on miksimisel põhinev anonüümimistehnoloogia, mis, kasutades ElGamali krüptosüsteemi homomorfsuse omadust, muudab talletatud häälte esitust ning järjekorda. Verificatum permuteerib sisendhääled ainult talle teadaoleva juhusliku permutatsiooniga P ning juhuslikustab iga üksiku hääle ümber ainult talle teadaolevate uute juhuarvude r_i abil.

Vältimaks häälte muutmist miksimise käigus, väljastab Verificatum mitteinteraktiivse nullteadmustõestuse – *miksimistõendi* – järgmiste omaduste kohta:

- uue järjekorra loomisel kasutatud P teadmine,
- ümberjuhuslikustamisel kasutatud r_i teadmine,
- väljundhäälte arvutatus lähtudes sisendhäältest ning parameetritest P ja r_i.

Häälte miksimine võimaldab kasutada tõestatavat dekrüpteerimist ning anda iga sedeli dekrüpteerimisel koos avatekstiga ka mitteinteraktiivne nullteadmustõestus, mis kinnitab et

- dekrüpteerija valdab võtit, millega sedel oli krüpteeritud;
- dekrüpteeritud väärtused on arvutatud krüptogrammist kasutades salajast võtit.

Kõigi antud valimise sedelite avatekste ja dekrüpteerimistõestusi kogumina nimetame *lugemistõendiks*.

1.1 Tõestuste verifitseerimine

Verifitseeritava protokolli kasutamisel on mõtet, siis kui tõestusi ka tegelikult kontrollitakse. Kuna protokoll on avalik, siis saab programmeerida sõltumatud verifitseerijad, ning neile verifitseerijatele tuleb anda ligipääs anonüümimisel tekkivatele tõestustele.

Teisest küljest on oluline tagada, et igas anonüümivas süsteemi komponendis (nt. Verificatumi ühes instantsis) kasutatavad permutatsioonid ning ümberjuhuslikustamisel kasutatavad juhuarvud jäävad ainult nende komponentide teada - anonüümimissaladus ei tohi lekkida.

Sõltumatu verifitseerija võimaldab anda välisele osapoolele hääletamistulemuse korrektsuse tõestamiseks vajalikke andmeid – nii lugemistõendit kui ka allkirjastatud hääli, kartmata hääle salajasuse pärast ning lekitamata anonüümimissaldadust.

Nii miksimistõendi kui lugemistõendi verifitseerija realiseerimine on võimalik, aga mittetriviaalne – juba vea otsimine arvutustes nõuab teadmist krüptoprotokollide notatsioonist ja arvutamismehaanikast. Seetõttu anname dokumendis ka elementaarse sissejuhatuse nimetatud protokollide krüptograafilistesse alustesse.

Protokollist arusaamiseks tuleb see läbi arvutada, hea on seda teha inimmõistusele hoomatavate numbritega. Krüptoprotokollides annab juba näiliselt väike viga *alati* suure ebaõnnestumise.

Sõltumatu verifitseerija implementeerimisel on oluline roll testvektoritel, mille abil on võimalik kontrollida implementatsiooni korrektsus. Kuid implementatsiooni vastavus testvektoritele ei garanteeri implementatsiooni täielikku korrektsust, vaid välistab elementaarsed tehnilised vead. Implementatsiooni korrektsuse tagamine on sõltumatu auditirakenduse implementeerija ülesanne!

Peatükk 2

Auditeeritava e-hääletamise krüptograafilised alused

IVXV raamistik eeldab, et hääle salajasuse kaitsmiseks kasutatakse avaliku võtme krüptosüsteemi, millel on mitte-deterministlikkuse ja homomorfsuse omadused.

2.1 Avaliku võtme krüptosüsteem

Defineerime krüptosüsteemi kui algoritmide kolmiku

$$\varepsilon = (Gen_{enc}, Enc, Dec),$$

kus Gen_{enc} on võtmegenereerimisalgoritm, Enc on krüpteerimisalgoritm ning Dec on dekrüpteerimisalgoritm. Edasises huvitavad meid asümmeetrilised ehk avaliku võtme krüptosüsteemid.

Avaliku võtme krüptosüsteemi võtmegenereerimisalgoritm Gen_{enc} väljastab kahest komponendist – avalikust võtmest ja salajasest võtmest – koosneva k-bitise võtme:

$$(pk, sk) \leftarrow Gen_{enc}(1^k).$$

Tähistagu C kõigi krüptogrammide hulka ning M kõigi võimalike avatekstide hulka. Krüptogrammi $c \in C$, mis on saadud avateksti $m \in M$ krüpteerimisel avaliku võtmega pk, on võimalik dekrüpteerida ainult salajase võtmega sk, mis tagab ligipääsu avatekstile vaid salajase võtme valdajale. Kehtib seos

$$\forall m \in M : Dec(sk, Enc(pk, m)) = m.$$

Märgime, et praktikas on oluline salajase võtme kaitseks kasutatav meetod. Salajase võtme lekkimine teeb haavatavaks kogu vastava avaliku võtmega krüpteeritud informatsiooni. Seetõttu rakendatakse salajaste võtmete talletamisel mitmeid erinevaid turvameetmeid alates läviskeemidest, mis võimaldavad võtme kasutamist ainult mitmest osapoolest koosneva kvoorumi kogunemise korral, lõpetades riistvaraliste krüptomoodulitega. Antud dokumendi kontekstis ei ole need meetodid siiski olulised. Eeldame, et

salajane võti püsib salajasena ning audiitoril on ligipääs ainult avalikule informatsioonile.

2.1.1 Mitte-deterministlikkuse omadus

E-hääletamine eeldab hääle salajasuse tagamiseks kasutatavalt avaliku võtme krüptosüsteemilt mitte-deterministlikkuse omadust. Deterministlikuks nimetame krüptosüsteemi, mis kujutab sama võtme kasutamisel sama sõnumi alati samaks krüptogrammiks. Krüptosüsteemi deterministlikkus tähendab et kahe erineva valija poolt samale kandidaadile antud e-hääled on välisel vaatlusel identsed. Mitte-deterministlik krüptosüsteem kasutab krüpteerimisel juhuslikkust ning sama avaliku võtme pk kasutamisel kujutatakse sama sõnum m alati erinevaks krüptogrammiks c_i, eeldusel et krüpteerimisel kasutatav juhuslikkus r_i oli samuti erinev. Tähistagu R järgnevas kõigi juhuarvude hulka.

$$\begin{aligned} r_1 \leftarrow R, c_1 &= Enc(pk, r_1, m), \\ r_2 \leftarrow R, c_2 &= Enc(pk, r_2, m), \\ m_1 &= Dec(sk, c_1), \\ m_2 &= Dec(sk, c_2), \\ c_1 \neq c_2 &\iff r_1 \neq r_2, \\ m &= m_1 = m_2. \end{aligned}$$

Kehtib seos

$$\forall m \in M, \forall r \leftarrow R : Dec(sk, Enc(pk, r, m)) = m.$$

2.1.2 Homomorfsuse omadus

E-hääletamine eeldab hääle salajasuse tagamiseks kasutatavalt avaliku võtme krüptosüsteemilt homomorfsuse omadust, mis võimaldab teatud operatsioonide sooritamist krüptogrammidel, omamata ligipääsu salajasele võtmele.

Olgu meil kaks krüptogrammi

$$\begin{aligned} c_1 &= Enc(pk, r_1, m_1), \\ c_2 &= Enc(pk, r_2, m_2). \end{aligned}$$

Avaliku võtme krüptosüsteem on homomorfne binaarse tehte $\oplus$ suhtes, kui peavad paika järgmised võrdused

$$\begin{aligned} c_1 \oplus c_2 &= Enc(pk, r_1, m_1) \oplus Enc(pk, r_2, m_2) \\ &= Enc(pk, r_3, m_1 \oplus m_2) \\ &= c_3, \\ Dec(sk, c_3) &= m_1 \oplus m_2. \end{aligned}$$

E-hääletamise korral kasutatakse homomorfsuse omadust nii mitte-deterministliku krüptosüsteemiga krüpteeritud hääle ümberjuhuslikustamiseks kui ka nullteadmustõestuste konstrueerimisel.

Näide: ümberjuhuslikustamine – krüptogrammis sisalduva juhuarvu muutmine teiseks juhuarvuks, jättes samas krüpteeritud sõnumi muutmata. Olgu algne krüptogramm

$$c_1 = Enc(pk, r_1, m),$$

siis eeldades krüptosüsteemi homomorfsust binaarse tehte $\oplus$ suhtes ning ühikelemendi 1 olemasolu saame ümberjuhuslikustatud krüptogrammi

$$\begin{aligned} c_2 &= Enc(pk, r_1, m) \oplus Enc(pk, r_2, 1) \\ &= Enc(pk, r_3, m \oplus 1) \\ &= Enc(pk, r_3, m). \end{aligned}$$

Kehtivad järgmised võrdused.

$$\begin{aligned} c_1 &\neq c_2, \\ m_1 &= Dec(sk, c_1), \\ m_2 &= Dec(sk, c_2), \\ m &= m_1 = m_2. \end{aligned}$$

Ümberjuhuslikustamise korrektsust on võimalik tõestada osapoolele, kes ei kontrolli salajast võtit.

2.2 Rühmad

Avaliku võtme krüptosüsteem eeldab, et meil on meetod avaliku ja salajase võtme loomiseks, mis kirjeldab seose avaliku ja salajase võtme vahel, samas tagades, et avalikust võtmest salajase tuvastamine on praktiliselt teostamatu ülesanne.

Kaasaegsete avaliku võtme krüptosüsteemide turvalisus tugineb teatud matemaatiliste probleemide lahendamise keerukusele – näiteks RSA krüptosüsteem rajaneb eeldusel, et arvude lahutamine algteguriteks on raske ülesanne. ElGamal krüptosüsteemi turvalisus rajaneb diskreetse logaritmi probleemi keerukusele. Mõlemad probleemid on rühmateoreetilised probleemid ning rühma mõistel on oluline roll mh. IVXV auditirakenduse loomisel.

2.2.1 Rühma mõiste

Mittetühja hulka G nimetame *rühmaks*, kui tal on defineeritud üks kahekohaline algebraline tehe $\odot$ nii, et kehtivad järgmised omadused:

- assotsiatiivsus: $\forall g, h, i \in G : (g \odot h) \odot i = g \odot (h \odot i)$,

- ühikelemendi leidumine: $\exists 1 \in G, \forall g \in G : 1 \odot g = g \odot 1 = g$,
- pöördelemendi leidumine: $\forall g \in G, \exists g^{-1} \in G : g \odot g^{-1} = g^{-1} \odot g = 1$.

Rühma tehet nimetatakse kas *korrutamiseks* või *liitmiseks*. Saab tõestada, et iga rühma ühikelement on unikaalne ning et igal rühma elemendil leidub täpselt üks pöördelement.

2.2.2 Näiteid rühmadest

- Täisarvud moodustavad rühma $(\mathbb{Z}, +)$ liitmise suhtes.
- Reaalarvud moodustavad rühma $(\mathbb{R}, +)$ liitmise suhtes.
- Algebraline struktuur $(\mathbb{Z}, \cdot)$ ei ole rühm korrutamise suhtes, sest enamuse täisarvude pöördarv ei ole täisarv.
- Algebraline struktuur $(\mathbb{R}, \cdot)$ ei ole rühm korrutamise suhtes, sest nullil ei ole pöördarvu.
- Reaalarvud ilma nullita moodustavad rühma $(\mathbb{R} \setminus \{0\}, \cdot)$ korrutamise suhtes.

2.2.3 Kommutatiivsed rühmad

Rühma G nimetatakse *kommutatiivseks e. Abeli* rühmaks, kui tema tehe $\odot$ on kommutatiivne:

$$\forall h, g \in G : g \odot h = h \odot g.$$

Lõpliku rühma G *järk* $|G|$ on defineeritud kui selle rühma elementide arv.

Rühma G elemendi $g \in G$ järgu $ord(g)$ defineerime kui vähima positiivse täisarvu i, mille korral kehtib võrdus $g^i = g \odot g \odot \ldots \odot g = 1$.

On teada, et lõpliku rühma iga elemendi järk jagab rühma järku. See teadmine võimaldab rühma elemendi järku efektiivselt leida.

Kui rühma elemendi $g \in G$ korral kehtib $ord(g) = |G|$, siis nimetame elementi g rühma *moodustajaks*, rühma ennast *tsükliliseks* ja kirjutame $G = \langle g \rangle$.

2.2.4 Näide: $\mathbb{Z}_{11}^*$

Vaatleme kõiki võimalikke jääke, mis tekivad täisarvu jagamisel 11-ga:

$$\mathbb{Z}_{11} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

Defineerime tehte $\odot$ kui korrutamise *modulo* 11. Sellisel juhul moodustavad kõik hulga $\mathbb{Z}_{11}$ nullist erinevad elemendid rühma tehte $\odot$ suhtes.

$$\mathbb{Z}_{11}^* = \mathbb{Z}_{11} \setminus \{0\}$$

Rühma tehe $\odot$ on esitatud korrutustabelis joonisel 2.1.

$\odot$	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	1	3	5	7	9
3	3	6	9	1	4	7	10	2	5	8
4	4	8	1	5	9	2	6	10	3	7
5	5	10	4	9	3	8	2	7	1	6
6	6	1	7	2	8	3	9	4	10	5
7	7	3	10	6	2	9	5	1	8	4
8	8	5	2	10	7	4	1	9	6	3
9	9	7	5	3	1	10	8	6	4	2
10	10	9	8	7	6	5	4	3	2	1

Joonis 2.1: Rühma $\mathbb{Z}_{11}^*$ korrutustabel

Rühm $\mathbb{Z}_{11}^*$ on Abeli rühm, kuna korrutamine on kommutatiivne.
Rühma $\mathbb{Z}_{11}^*$ järk on $|\mathbb{Z}_{11}^*| = 10$.
Rühma $\mathbb{Z}_{11}^*$ elemendi 1 järk on $ord(1) = 1$.
Rühma $\mathbb{Z}_{11}^*$ elemendi 10 järk on $ord(10) = 2$.
Rühma $\mathbb{Z}_{11}^*$ elemendi 2 järk on $ord(2) = 10$.

$$2^1 = 2,$$
$$2^2 = 4,$$
$$2^3 = 8,$$
$$2^4 = 5,$$
$$2^5 = 10,$$
$$2^6 = 9,$$
$$2^7 = 7,$$
$$2^8 = 3,$$
$$2^9 = 6,$$
$$2^{10} = 1.$$

Elemendi 2 astmed moodustavad terve rühma $\mathbb{Z}_{11}^*$, teisisõnu – element 2 on rühma $\mathbb{Z}_{11}^*$ moodustaja ning rühm $\mathbb{Z}_{11}^*$ on tsükliline rühm.
Rühma $\mathbb{Z}_{11}^*$ elemendi 3 järk on $ord(3) = 5$.

$\odot$	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	**1**	2	**3**	**4**	**5**	6	7	8	**9**	10
2	2	4	6	8	10	1	3	5	7	9
3	**3**	6	**9**	**1**	**4**	7	10	2	**5**	8
4	**4**	8	**1**	**5**	**9**	2	6	10	**3**	7
5	**5**	10	**4**	**9**	**3**	8	2	7	**1**	6
6	6	1	7	2	8	3	9	4	10	5
7	7	3	10	6	2	9	5	1	8	4
8	8	5	2	10	7	4	1	9	6	3
9	**9**	7	**5**	**3**	**1**	10	8	6	**4**	2
10	10	9	8	7	6	5	4	3	2	1

Joonis 2.2: Rühma $\mathbb{Z}_{11}^*$ elemendi 3 poolt moodustatud alamrühma korrutustabel

$$3^1 = 3$$
$$3^2 = 9$$
$$3^3 = 5$$
$$3^4 = 4$$
$$3^5 = 1$$

Elemendi 3 astmed moodustavad rühma $\mathbb{Z}_{11}^*$ alamrühma, mille korrutustabel on esitatud osana rühma $\mathbb{Z}_{11}^*$ korrutustabelist joonisel 2.2. Moodustatud alamrühma järk on 5.

2.3 ElGamali krüptosüsteem

ElGamali krüptosüstem [2] on laialt levinud avaliku võtme krüptosüsteem, mis on mitte-deterministlik ja homomorfne krüptogrammide korrutamise suhtes.

2.3.1 Rühma parameetrid ja võtme genereerimine

Olgu p turvaline algarv, st algarv kujul $p = 2q + 1$, kus ka q on algarv. Eelneva tõttu on kõik alamrühmad kas järguga 1, 2, q või $2q$ ning $|\mathbb{Z}_p^*| = 2q$.

Olgu $g \in \mathbb{Z}_p^*$ selline, et $ord(g) = q$. Olgu $G = (\langle g \rangle, \odot)$.

Genereerime k-bitise võtme: $(pk, sk) \leftarrow Gen_{enc}(1^k)$.

1. $x \leftarrow \mathbb{Z}_q, h = g^x \mod p$,
2. $pk = ((p,q,g),h)$,
3. $sk = x$.

Avaliku võtme krüptosüsteemi turvalisuseks on vaja, et avalikust võtmest ei saaks tuletada salajast võtit. St, kui meil on antud $pk = ((p,q,g),h) = ((p,q,g),g^x)$, siis peab olema keeruline arvutada $sk = x$ väärtust.

Parameetrid p, q ja g määravad multiplikatiivse rühma $\mathbb{Z}_p^*$ ja tema tsüklilise alamrühma G, kus *diskreetse logaritmi ülesanne* on raske.

Diskreetse logaritmi ülesandena mõistame lõplikus rühmas G elementide $g, g^x \in G$ järgi väärtuse $x \in \mathbb{Z}$ leidmist.

Logaritmi leidmine pole mitte igas rühmas raske, näiteks rühma $(R \setminus \{0\}, \cdot)$ jaoks eksisteerivad lähendusmeetodid. Seni pole leitud efektiivset meetodit arvutamaks diskreetseid logaritme rühmas $(\mathbb{Z}_p^*, \cdot)$, kus p on suur (vähemalt $k = 2048$ bitti) *turvaline algarv*.

2.3.2 Krüpteerimine ja dekrüpteerimine

Olgu meil sõnum $m \in G$, ElGamali krüptosüsteemi salajane võti $sk = x$ ning ElGamali krüptosüsteemi avalik võti $pk = ((p,q,g),h = g^x)$, siis krüpteerime sõnumi m: $Enc(pk,r,m) = c$ järgmiselt:

1. $m \in G, r \leftarrow \mathbb{Z}_q$,
2. $u = g^r \mod p$,
3. $v = h^r \odot m \mod p$,
4. $c = (u,v)$.

Dekrüpteerime krüptogrammi c: $Dec(sk,c) = m$.

$$\begin{aligned} Dec(sk,c) &= Dec(x,(u,v)) \\ &= v \odot u^{-x} \\ &= h^r \odot m \odot (g^r)^{-x} \\ &= (g^x)^r \odot m \odot (g^r)^{-x} = m. \end{aligned}$$

ElGamali krüptosüsteem on mitte-deterministlik, sest krüpteerimine sõltub juhuarvust r. ElGamali krüptosüsteemi salajast võtit sk valdav osapool saab dekrüpteerida kõiki vastava avaliku võtmega pk krüpteeritud sõnumeid, samas on ühte konkreetset ElGamali krüptosüsteemi krüptogrammi võimalik avada ka siis kui on teada krüpteerimisel kasutatud juhuslikkus r, seda sõltumata võimalike avatekstide hulgast. Sisuliselt toimib r sõnumipõhise salajase võtmena, mis teeb lihtsaks e-hääle kontrollimisprotokolli:

$$\begin{aligned} Verify(r,c) &= Verify(r,(u,v)) \\ &= v \odot h^{-r} \\ &= h^r \odot m \odot h^{-r} = m. \end{aligned}$$

ElGamali krüptosüsteem eeldab avateksti kuulumist rühma G. Praktikas tähendab see vajadust suvaliste krüpteeritavate baidijadade teisendamiseks rühma G elemendiks (nt. 4.3).

2.3.3 Homomorfsus ja rerandomiseerimine

ElGamali krüptosüsteem on homomorfne krüptogrammide korrutamise suhtes. Kui võtame kaks ElGamali krüptosüsteemi krüptogrammi, siis on ka nende korrutis ElGamali krüptosüsteemi krüptogramm.

$$\begin{aligned} Enc(h,r_1,m_1) \odot Enc(h,r_2,m_2) &= (u_1,v_1) \odot (u_2,v_2) \\ &= (g^{r_1}, h^{r_1} \odot m_1) \odot (g^{r_2}, h^{r_2} \odot m_2) \\ &= (g^{r_1} \odot g^{r_2}, h^{r_1} \odot m_1 \odot h^{r_2} \odot m_2) \\ &= (g^{r_1+r_2}, h^{r_1+r_2} \odot (m_1 \odot m_2)) \\ &= Enc(h, r_1+r_2, m_1 \odot m_2) \end{aligned}$$

Kasutame ära homomorfsuse omadust ning teeme krüptogrammialuse tehte rühma ühikelemendiga.

$$\begin{aligned} Enc(h,r_1,m) \odot Enc(h,r_2,1) &= (u_1,v_1) \odot (u_2,v_2) \\ &= (g^{r_1}, h^{r_1} \odot m) \odot (g^{r_2}, h^{r_2} \odot 1) \\ &= (g^{r_1} \odot g^{r_2}, h^{r_1} \odot m \odot h^{r_2} \odot 1) \\ &= (g^{r_1+r_2}, h^{r_1+r_2} \odot (m \odot 1)) \\ &= Enc(h, r_1+r_2, m \odot 1) \\ &= Enc(h, r_1+r_2, m) \end{aligned}$$

Krüptogramm muutub, kuid sõnum jääb samaks!

2.3.4 Näide: ElGamal krüptosüsteemi kasutamine $p = 11$ korral

Paneme tähele, et $p = 11$ on algarv kujul $p = 2q+1$, $q = 5$. Alampeatükis 2.2.4 nägime, et rühma $\mathbb{Z}_{11}^*$ elemendi 3 järk $ord(3) = 5$. Seega võivad meie rühma G määravad parameetrid olla: $p = 11, q = 5, g = 3$.

$$G = \{1,3,4,5,9\}$$

$$\begin{aligned} 3^1 &= 3 \\ 3^2 &= 9 \\ 3^3 &= 5 \\ 3^4 &= 4 \\ 3^5 &= 1 \end{aligned}$$

Genereerime ElGamali krüptosüsteemi võtme. Valime juhuslikult salajase võtme $x \in \mathbb{Z}_5, x = 4$. Avalik võti on $h = g^x = 3^4 = 4$.

Olgu sõnum $m = m_1 = 3$ ja $m_2 = 5$. Olgu sõnum $m_3 = m_1 \odot m_2$. Tulenevalt homomorfsuse omadusest peab kehtima

$$3 \odot 5 = 3 \cdot 5 \mod 11 = 4.$$

Krüpteerime sõnumi m_1. Valime juhuslikult $r_1 \in \mathbb{Z}_5, r_1 = 4$.

$$c_1 = (3^4, 4^4 \odot 3) = (4, 3 \odot 3) = (4, 9).$$

Krüpteerime sõnumi m_2. Valime juhuslikult $r_2 \in \mathbb{Z}_5, r_2 = 3$.

$$c_2 = (3^3, 4^3 \odot 5) = (5, 9 \odot 5) = (5, 1).$$

Arvutame homomorfsuse omadust arvestades c_3.

$$c_1 \odot c_2 = (4,9) \odot (5,1) = (4 \odot 5, 9 \odot 1) = (9,9) = c_3.$$

Krüpteerime ühikelemendi 1: Valime juhuslikult $r_3 \in \mathbb{Z}_5, r_3 = 3$.

$$c_4 = (3^3, 4^3 \odot 1) = (5, 9 \odot 1) = (5, 9).$$

Rerandomiseerime krüptogrammi c_1.

$$c_5 = c_1 \odot c_4 = (4,9) \odot (5,9) = (4 \odot 5, 9 \odot 9) = (9,4).$$

Dekrüpteerime krüpteeritud sõnumid. Tuletame meelde, et $x = 4$ ja paneme tähele, et $-x = 1$.

Dekrüpteerime c_1: $Dec(x, c_1) = Dec(4, (4,9)) = 9 \odot 4^1 = 3$.
Dekrüpteerime c_2: $Dec(x, c_2) = Dec(4, (5,1)) = 1 \odot 5^1 = 5$.
Dekrüpteerime c_3: $Dec(x, c_3) = Dec(4, (9,9)) = 9 \odot 9^1 = 4$.
Dekrüpteerime c_5: $Dec(x, c_5) = Dec(4, (9,4)) = 4 \odot 9^1 = 3$.

Paneme tähele, et kehtib homomorfsuse omadus $Dec(x, c_1) \cdot Dec(x, c_2) = Dec(x, c_3)$ ning rerandomiseerimine $Dec(x, c_1) = Decc(x, c_5)$.

2.3.5 Ruutjäägid

Elementi $a \in \mathbb{Z}_p^*$ nimetatakse *ruutjäägiks* mooduli p järgi, kui leidub selline $x \in \mathbb{Z}_p^*$, et $x^2 = a \mod p$. Kõiki antud tingimust rahuldavaid elemente x nimetatakse elemendi a ruutjuurteks. Kui elemendil a ei ole ruutjuuri, siis nimetatakse seda elementi *mitteruutjäägiks*.

Suvalise elemendi $b \in \mathbb{Z}_p^*$ korral on võimalik tuvastada, kas tegu on ruutjäägiga, kasutades Euleri kriteeriumit.

Euleri kriteerium ütleb, et paaritu algarvu p korral on element $b \in \mathbb{Z}_p^*$ ruutjääk siis ja ainult siis, kui

$$b^{\frac{p-1}{2}} = 1 \mod p.$$

Euleri kriteeriumist lähtudes on defineeritud Legendre'i sümbol. Olgu p paaritu algarv ning a täisarv, mille korral $\gcd(a,p)=1$, siis on Legendre'i sümbol defineeritud järgmiselt:

$$\left(\frac{a}{p}\right)=\begin{cases}1 & \text{kui } a \text{ on ruutjääk,}\\ -1 & \text{kui } a \text{ on mitteruutjääk.}\end{cases} \tag{2.1}$$

ElGamali krüptosüsteemi semantilise turvalisuse huvides on oluline arvestada, et ElGamali krüptogramm lekitab informatsiooni selle kohta, kas avatekst oli ruutjääk või mitte, mis omakorda võib anda infot selle kohta, millise valiku valija tegi/ei teinud. Tagamaks semantilist turvalisust tuleb tagada, et kõik avatekstid oleks ruutjäägid [3].

Lähtme eeldusest, et ElGamali krüptosüsteemi aluseks oleva rühma parameeter p on kujul $p=2q+1$, kus ka q on algarv. Sellisel juhul kehtib võrdus $p=3 \mod 4$, mis garanteerib, et element 1 on ruutjääk ja element -1 ei ole ruutjääk. Kasutades Legendre'i sümbolit kontrollime, kas m on ruutjääk rühmas $\mathbb{Z}_p^*$. Kui m on ruutjääk, siis krüpteerime m, vastasel juhul krüpteerime m asemel rühma $\mathbb{Z}_p^*$ elemendi $-m$, mis on garanteeritult ruutjääk.

Dekrüpteerimisel saadud avateksti m korral kontrollime, kas kehtib $m \leq q$. Kui $m \leq q$, siis võtame avatekstiks $m_0 = m$, vastasel juhul $m_0 = -m$.

2.4 Nullteadmustõestused

Teadmustõestus on protokoll, mis võimaldab ühel osapoolel veenda teist mingi väite paikapidavuses. Nullteadmustõestus on teadmustõestuse erijuht, kus ainus informatsioon, mida teine pool saab, on veendumus väite paikapidavuses, ei midagi muud. Näiteks küsimus sellest, kuidas veenda valimise audiitorit, et e-hääled on dekrüpteeritud korrektselt, andmata talle samas valimise salajast võtit, saab vastuse nullteadmustõestuste abil. Nullteadmustõestused on ka miksimistõendi ja lugemistõendi sisu – soovime tõestada korrektset miksimist, samas miksimissaladust lekitamata, soovime tõestada korrektset dekrüpteerimist, samas salajast võtit lekitamata. Meie roll täna on olla nullteadmustõestustes *verifitseerija* ning IVXV (ja Verificatum) on *tõestaja*.

Tõestaja P soovib tõestada verifitseerija V'le, et tal on mingi teadmine w (*witness, tunnisti*) väitest x, seda tunnistit üle andmata. Nii P kui V teavad predikaati R, mille abil saab kontrollida, et w on väite x tunnisti. Väite x tunnisti w kontrollimine predikaadiga $R(x,w)=1$ toimub polünomiaalses ajas. Tõestajal on $R,x,w : R(x,w)=1$, selle tõestamiseks loob ta tõestuse π. Verifitseerijal on R,x,π.

2.4.1 Σ-protokoll

Klassikaline Σ-protokoll koosneb neljast osast:

1. P kinnistab (*commits*) mingi protokollis kasutatava väärtuse V'le,
2. V esitab P'le juhusliku väljakutse (*challenge*),
3. P vastab V väljakutsele uue protokollikohase arvutusega,

4. V kontrollib, et P vastus oleks kooskõlaline protokolli, algselt kinnistatud väärtuse (*commitment* e. *pühendumus*) ja juhusliku väljakutsega.

2.4.2 Schnorri protokoll

Schnorri protokollis [4] on P'l ElGamali krüptosüsteemii salajane võti w ning ta soovib selle teadmise fakti V'le tõestada, samas võtit lekitamata. Sisuliselt tõestatakse Schnorri protokolliga diskreetse logaritmi teadmist ja ElGamali krüptosüsteemi dekrüpteerimise tõestus on tegelikult diskreetse logaritmi teadmise tõestus.

Võtame meile tuttava rühma $G = (\langle g \rangle, \odot)$, kus $g \in \mathbb{Z}_p^*$, $p = 2q+1$, $ord(g) = q$ ning p ja q on algarvud.

$w \leftarrow \mathbb{Z}_q, h = g^w \mod p$.

Schnorri protokoll toimib järgmiselt:

1. Kinnistamine, P
   (a) P: $t \leftarrow \mathbb{Z}_q, y = g^t$
   (b) $P \rightarrow V$: y
2. Väljakutse, V
   (a) V: $c \leftarrow \mathbb{Z}_q$
   (b) $V \rightarrow P$: c
3. Vastus, P (w)
   (a) P: $s = t + w \cdot c \mod q$
   (b) $P \rightarrow V$: s
4. Kontroll, V (y)
   (a) V: $g^s = y \odot h^c$

Tõesti, kehtib võrdus:

$$g^s = g^{t+w \cdot c} = g^t \odot (g^w)^c = y \odot h^c.$$

2.4.3 Fiat-Shamiri heuristika

Schnorri protokoll on interaktiivne: ta eeldab, et V on protokolli täitmise ajal reaalajas kättesaadav ning genereerib juhusliku väljakutse. Praktikas soovime sageli tõestuse hilisemat verifitseerimist. Fiat-Shamiri heuristika on üks levinud tehnika interaktiivse tõestuse mitte-interaktiivseks tegemiseks.

Fiat-Shamiri heuristika eeldab kollisioonikindlat räsifunktsiooni: $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$. Kõik V sisendid asendatakse räsidega protokolli eelmiste sammude sisenditest:

- interaktiivse protokolli tõestuskäik oli $\pi = (g^t, c, t + w \cdot c)$,
- mitte-interaktiivse protokolli tõestuskäik on $\pi = (g^t, H(g^t), t + w \cdot H(g^t))$.

Fiat-Shamiri heuristika turvalisus ei ole tõestatud. Fiat-Shamiri heuristika võimalik ebaturvalisus ei ole kinnitust leidnud.

2.4.4 Deterministlik pseudojuhuslik generaator

Klassikalised räsifunktsioonid (nt. SHA-256) üldjuhul ei sobi Fiat-Shamiri heuristikas vahetuks kasutamiseks, kuna seavad suvalise pikkusega baidijadale vastavusse fikseeritud pikkusega baidijada (nt. 32 baiti), mis ei pruugi ühilduda rühma parameetritega.

Fiat-Shamiri heuristikale sobiv räsifunktsioon peab olema kollisioonikindel ning lähtudes sisendväärtusest väljastama täisarvu kindlast vahemikust. Sellise räsifunktsiooni saab koostada deterministliku pseudojuhusliku generaatori abil.

Deterministlik pseudojuhuslik generaator (DPRNG) on meetod näiliselt juhuslike baidijadade genereerimiseks, kasutades etteantud seemneväärtust. Sama seemneväärtuse korral genereerib meetod alati sama jada, kuid jada ise on statistiliselt peaaegu eristamatu tõeliselt juhuslikust jadast.

IVXV raamistik kasutab SHA2 põhist loenduriga DPRNG'd, kus räsifunktsioonist H ja seemnest *seed* lähtuv pseudojuhuslik jada on defineeritud järgmiselt:

$$DPRNG = H(1||seed)||H(2||seed)||...$$

Eesmärgiks on mingist seemnest lähtudes saada täisarv kindlas vahemikus. Naiivne meetod loeks DPRNGst vajaliku arvu baite ning võtaks jäägi ülemise tõkke järgi, kuid sellisel juhul tekib nn. *sampling bias* ning valim ei ole ühtlase jaotusega. Korrektne on kasutada tagasilükkamisega valimist (*rejection sampling*), mida kirjeldab algoritm 1.

Algorithm 1: Fiat-Shamiri väljakutse tagasilükkamisega valimine

Input: $q > 0$
Input: *seed*
Output: $0 \leq y < q$

```
1 DPRNGInit(seed);
2 while true do
3 |   y ← DPRNGRead(length of q in bytes);
4 |   if y < q then
5 |   |   return y
6 |   end
7 end
```

Tagasilükkamisega valimise meetodi eripära on, et DPRNG'd kasutades ei ole võimalik ette teada, palju baite tuleb enne sobiva täisarvu leidmist lugeda. Seetõttu ei ole optimaalne kasutada muutuva väljundpikkusega räsifunktsioone (nt. SHA3 SHAKE). Printsiibis on võimalik SHAKE realiseerimine selliselt, et väljundit saab käsitleda voona, praktikas ei ole enamikes SHAKE'i pakkuvates teekides voona kasutamine võimalik, mis suurendab oluliselt selle meetodi keerukust, seetõttu kasutab IVXV raamistik SHA2 põhist loenduriga DPRNG'd.

Peatükk 3

Hääletamistulemuse korrektsuse tõendamine IVXV raamistikus

3.1 Lugemistõend

IVXV raamistikus väljastatakse e-häälte dekrüpteerimisel lugemistõend – nullteadmustõestus, millega dekrüpteerija tõestab, et ta teab salajast võtit, mis on paariline valimiste avalikule võtmele. Täiendavalt tõestab dekrüpteerija, et seda salajast võtit kasutades avaneb antud krüptogramm avatekstiks.

ElGamali krüptosüsteemi kasutamisel on lugemistõend realiseeritav, kasutades Schnorri nullteadmustõestusel põhinevat protokolli diskreetse logaritmi teadmise tõestamiseks [4].

Olgu meil SHA2 põhisel loenduriga DPRNG'l baseeruv räsifunktsioon H ning krüptogramm

$$c = (u, v) = (g^r \bmod p, m \odot h^r \bmod p).$$

Dekrüpteerimine $PDec(c, sk) = (m, a, b, s)$ koos lugemistõendi väljastamisega kasutades Fiat-Shamiri heuristikut toimub järgmiselt:

1. P valdab salajast võtit x.
2. Dekrüpteerimine
   (a) $m = v \cdot u^{-x}$.
3. Kinnistamine
   (a) Valime t juhuslikult.
   (b) Arvutame sõnumipühendumuse $a = u^t$.

(c) Arvutame võtmepühendumuse $b = g^t$.

4. Väljakutse

   (a) Arvutame räsifunktsiooni H, kasutades väljakutset $k = H(seed)$.

5. Vastus

   (a) Arvutame vastuse $s = k \cdot x + t \mod q$.

   (b) Väljastame avateksti m ning lugemistõendi komponendid a, b, s.

Lugemistõendi kontrollimine $VerifyPDec(pk, c, m, a, b, s)$ toimub järgmiselt:

1. V valdab avalikku võtit $h = g^x$, krüptogrammi $c = (u, v)$, avateksti m ning lugemistõendi komponente a, b, s.

2. Kontroll

   (a) Arvutab väljakutse $k = H(seed)$.

   (b) Kontrollib võrdust $u^s = a \cdot (v/m)^k$.

   (c) Kontrollib võrdust $g^s = b \cdot h^k$.

Räsifunktsiooni lähtestamiseks vajalik seeme $seed = DECRYPTION|pk|c|m|a|b$ on järgmise ASN.1 andmestruktuuri DER-kodeering:

```
SEQUENCE ::= {
    NIPROOFDOMAIN       GENERAL STRING,
    pubkey              SubjectPublicKeyInfo,
    ciphertext          encryptedBallot,
    decrypted           OCTET STRING,
    msgCommitment       INTEGER,
    keyCommitment       INTEGER
}
```

Välja *NIPROOFDOMAIN* väärtus on sõne "DECRYPTION".

3.2 Miksimistõend

IVXV raamistik näeb ette võimaluse täiendava anonüümimismeetodi – krüptograafilise miksimise – kasutamiseks. Krüptograafiline miksimine on protsess, mis võtab sisendiks hulga krüpteeritud hääli B_1 ja annab väljundiks hulga krüpteeritud hääli B_2 ning miksimistõendi P_{mix} selliselt, et:

1. B_1 ja B_2 dekrüpteerimisel tekkivad avatekstide hulgad on samad;

2. ühegi krüpteeritud hääle kohta hulgast B_2 ei ole võimalik öelda, milline krüpteeritud hääl hulgast B_1 on temaga vastavuses ja vastupidi;

3. P_{mix} on matemaatiline tõestus tingimuse 1 täidetuse kontrollimiseks. P_{mix} kontrollimine on võimalik ilma hulkade B_1 ja B_2 vahelist vastavust avaldamata.

Sisuliselt tähendab krüptograafiline miksimine seda, et algsete krüpteeritud häälte asemel võime avada miksitud hääled, arvutada hääletamistulemuse ning kontrollida miksitud häälte dekrüpteerimisel saadud lugemistõendit. Kui saame kontrollida, et miksimisprotsessi sisendiks läksid õiged hääled, siis võime kogu protsessi lõpuks olla veendunud, et hääletamistulemus arvutati sisendist lähtudes korrektselt. Tänu miksimisele saab audiitor veenduda tulemuse korrektsuses, kuid ei saa mingit informatsiooni konkreetsete valijate eelistuste kohta.

Krüptograafilise miksimise rakendamine nõuab kasutatavalt krüptosüsteemilt ümberjuhulikustamise võimalust homomorfsuse omadust kasutades. Ümberjuhuslikustamise tulemusena saame kaks krüptogrammi, mille krüpteeritud sõnum on identne, kuid esitusviis erinev. Tõestus korrektsest ümberjuhulikustamisest on osa miksimistõendist.

Täitmaks krüptograafilisele miksimisele seatavat nõuet – vastavus sisendi ja väljundi vahel peab olema peidetud – rakendatakse miksimisel lisaks ümberjuhuslikustamisele ka segamist, mille tulemusena esitab miksimisprotsess oma väljundiks ümberjuhuslikustatud sisendkrüptogrammid teises järjekorras.

Miksimistõend on terviktõestus, mis näitab, et väljundhäälte hulk on saadud sisendhäälte hulgast korrektse ümberjuhuslikustamise ja segamise teel. Tõestus antakse nullteadmuse abil, mis tähendab, et me saame kinnituse väite paikapidavusest, kuid miksimisprotsessi siseinfot meile ei avalikustata. Kui lugemistõend antakse iga avatud hääle kohta eraldi, siis miksimistõend on alati üks tervik sõltumata miksitud häälte hulgast. Siiski tuleb silmas pidada, et miksimistõendi pikkus sõltub alati krüptogrammide arvust.

Peatükk 4

Algoritmid ja andmestruktuurid

Krüptograafilised protokollid on elektroonilise hääletamise auditeerimise aluseks, kuid nende praktikas rakendamisel on vaja lahendada ka mitmeid lihtsamaid tehnilisi probleeme nagu näiteks andmete kodeerimine ja talletamine sobival kujul. Anname ülevaate vajalikest andmevormingutest ja seotud algoritmidest.

4.1 Elektroonilise tahteavalduse vorming

Valija tahteavaldus avakujul eksisteerib valijarakenduses ning hiljem ka kontrollrakenduses, võtmerakenduses ja auditirakenduses. Tahteavaldus sisaldab nii valiku koodi ringkonnas, ringkonna EHAK-koodi kui ka valiku nimekirja nime ning konkreetse valiku nime nimekirjas. Kirjeldame tahteavalduse vormingu BNF kujul.

```
ehak-district = 1*10DIGIT
choice-no = 1*11DIGIT
district-choice = ehak-district '.' choice-no
choicelist-name = 1*100UTF-8-CHAR
choice-name = 1*100UTF-8-CHAR
ballot = district-choice '\x1F' choicelist-name '\x1F' choice-name
```

Valija tahteavaldust töödeldakse UTF-8 kodeeritud baidijadana.

Näide: Valija ringkonnast EHAK koodiga '9876' soovib hääletada kandidaadi nr. '999' poolt – 'Prževalski Hobune' erakonnast 'Kabjaliste selts'. Tahteavaldus esitatakse järgmise 43 sümbolilise sõnena:

```
'9876.999\x1FKabjaliste selts\x1FPrževalski Hobune'
```

UTF-8 kodeeritud esitus on järgmine 44 baidi pikkune jada (sümbol 'ž' kodeering on kahe baidi pikkune):

```
\x39 \x38 \x37 \x36 \x2e \x39 \x39 \x39
\x1F \x4b \x61 \x62 \x6a \x61 \x6c \x69
\x73 \x74 \x65 \x20 \x73 \x65 \x6c \x74
\x73 \x1F \x50 \x72 \xc5 \xbe \x65 \x76
\x61 \x6c \x73 \x6b \x69 \x20 \x48 \x6f
\x62 \x75 \x6e \x65
```

4.2 Avaliku võtme ja rühmaparameetrite vorming

Valija tahteavaldus avakujul krüpteeritakse valijarakenduse poolt valimise korraldaja genereeritud avaliku võtmega.

ElGamali krüptosüsteemi avalik võti kodeeritakse koos ElGamali krüptosüsteemi parameetritega ning konkreetset valimist iseloomustava identifikaatoriga. Krüptosüsteemi parameetrid on osaks algoritmi identifikaatori struktuurist, avalik võti on kodeeritud *SubjectPublicKeyInfo* struktuuri. Paneme tähele, et parameetrit q ei talletata. Kuna eeldame turvalise algarvu $p = 2 \cdot q + 1$ kasutamist, siis on q lihtsasti leitav: $q = (p-1)/2$. Võtme kasutaja peab olema veendunud selles, et q on tõesti algarv.

```
elGamalEncryption OBJECT IDENTIFIER ::= {
    { iso(1) org(3) dod(6) internet(1) private(4) enterprise(1)
      dds(3029) asymmetric-encryption(2) 1}
}

elGamal-Params-IVXV ::= SEQUENCE {
    p                   INTEGER,
    g                   INTEGER,
    election-identifier GeneralString
}

elGamalPublicKey ::= SEQUENCE {
    h                   INTEGER,
}

SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm           AlgorithmIdentifier,
    subjectPublicKey    BIT STRING
}
```

4.3 Elektroonilise tahteavalduse teisendamine rühma elemendiks

Valija tahteavalduse krüpteerimiseks võetakse UTF-8 kodeeringus struktuur *ballot* ning teisendatakse see ElGamali krüptosüsteemi parameetrite poolt kirjeldatud rühma elemendiks. Eeldame, et parameetri q minimaalne pikkus on 256 baiti. Sellisel juhul võib

struktuuri *ballot* pikkus olla maksimaalselt 253 baiti. Avakujul tahteavaldus pikendatakse parameetri q pikkuseni.

```
padded-ballot = '\x00' '\x01' *'\xff' '\x00' ballot
```

Näide: Võtame parameeter q pikkusega 64 baiti (NB! praktikas ebaturvaline). Pikendame avakujul tahteavalduse 44 baidi pealt 64 baidini:

```
\x00 \x01 \xff \xff \xff \xff \xff \xff
\xff \xff \xff \xff \xff \xff \xff \xff
\xff \xff \xff \x00 \x39 \x38 \x37 \x36
\x2e \x39 \x39 \x39 \x1F \x4b \x61 \x62
\x6a \x61 \x6c \x69 \x73 \x74 \x65 \x20
\x73 \x65 \x6c \x74 \x73 \x1F \x50 \x72
\xc5 \xbe \x65 \x76 \x61 \x6c \x73 \x6b
\x69 \x20 \x48 \x6f \x62 \x75 \x6e \x65
```

Pikendatud tahteavaldust interpreteeritakse kui tavajärjestusega (*big-endian*) positiivset täisarvu m.

Rühma elemendist elektroonilise tahteavalduse taastamiseks tuleb kõigepealt täisarvu m binaaresituse algusest eemaldada mustrile vastav täidis. Allesjäänud baidid kirjeldavad elektroonilist tahteavaldust UTF-8 kodeeringus (lõik 4.1).

```
padding = '\x00' '\x01' *'\xff' '\x00'
```

Märgime, et kuigi kirjeldatud meetodiga saame rühma elemendi, on meil turvalisuse huvides oluline teisendada see element ruutjäägiks 2.3.5.

4.4 Krüpteeritud sedeli vorming

Rühma $\mathbb{Z}_p^*$ ruutjäägina kodeeritud tahteavaldus krüpteeritakse vastavalt ElGamali krüptosüsteemi avaliku võtmega ning esitatakse ASN.1 andmestruktuurina *encryptedBallot*, mille DER-kodeering on krüpteeritud sedel ehk sisemine ümbrik topletümbriku skeemis.

```
elGamalEncryptedMessage ::= SEQUENCE {
    u           INTEGER,
    v           INTEGER
}

encryptedBallot ::= SEQUENCE {
    algorithm   AlgorithmIdentifier,
    cipher      ANY
}
```

4.5 Lugemistõendi vorming

Lugemistõend koosneb kõigist ühe valimise käigus krüptogrammidest c_i dekrüpteeritud tahteavaldustest f_i ning korrektset dekrüpteerimist kinnitavatest tõestustest.

Iga tõestus koosneb sõnumipühendumusest a, võtmepühendumusest b ning vastusest s. Tõestus esitatakse ASN.1 struktuurina $decryptionProof$ ja DER-kodeeritakse.

```
decryptionProof ::= SEQUENCE {
    msgCommitment      INTEGER,
    keyCommitment      INTEGER,
    response           INTEGER,
    intermediate_k     INTEGER OPTIONAL,
    intermediate_seed  INTEGER OPTIONAL
}
```

Krüptogramm c_i esitatakse ASN.1 struktuurina $encryptedBallot$ ja DER-kodeeritakse.

Tahteavaldus esitatakse UTF-8 kodeeringus andmestruktuurina $ballot$.

Lugemistõend esitatakse JSON vormingus. Lugemistõend sisaldab valimise identifikaatorit ning loendit kõigist dekrüpteerimistõestustest. Kõik binaarkujul andmed (s.t. DER-kodeeritud andmestruktuurid) BASE64 kodeeritakse. Tahteavaldus lisatakse lugemistõendisse sõnena.

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "definitions": {
        "proofs": {
            "type": "array",
            "items": {
                "type": "object",
                "required": ["ciphertext", "message", "proof"],
                "ciphertext": {
                    "type": "string"
                },
                "message": {
                    "type": "string"
                },
                "proof": {
                    "type": "string"
                },
                "additionalProperties": false
            },
            "additionalItems": false
        }
    },
    "type": "object",
    "properties": {
        "election": {
```

```
            "type": "string"
        },
        "proofs": {
            "$ref": "#/definitions/proofs"
        }
    },
    "required": [
        "election",
        "proofs"
    ],
    "additionalProperties": false
}
```

Näide lugemistõendist.

```
{
    "election" : "RK2051",
    "proofs" : [
        {
            "ciphertext" : "MIIDGTALBgkrBgEEAZdVAgEwggMIAo...",
            "message" : "9876.999\u001FKabjaliste selts\u001FPr\u00c5\u00beevalski Hobune",
            "proof" : "MIIEjQKCAYEA2MKKVs+j7gFwaCDPJummsGR..."
        },
        ...
    ]
}
```

4.6 Miksimistõendi vorming

Miksimistõendi kontrollimiseks kasutatakse algoritmi nagu on defineeritud Verificatumi verifitseerija realiseerimise manuaalis – `https://www.verificatum.org/files/vmnv-3.0.3.pdf`.

Märgime, et miksimistõendi koostamisel lisatakse krüptogrammile andmed valimiste, ringkonna, jaoskonna ja küsimuse identifikaatori kohta. Lisamiseks kodeeritakse vastav väli rühma elemendina, kasutades pimendamiseks juhuslikkust 0. Näitena, kui esialgu on krüptogramm $c_0 = (c_{00}, c_{01})$, kasutades avalikku võtit $pk = (g, y)$, siis Verificatumi sisendina kasutatakse laia krüptogrammi $C = (c_{id}, c_d, c_s, c_q, c_0)$, kus:

1. valimiste identifikaatori pseudokrüptogramm on antud kujul $c_{id} = (1, encode(id))$, kus funktsioon *encode* kodeerib sõne vastava rühma elemendina ja *id* on valimiste identifikaatori sõne.

2. ringkonna identifikaatori pseudokrüptogramm on antud kujul $c_d = (1, encode(d))$, kus d on ringkonna identifikaatori sõne.

3. jaoskonna identifikaatori pseudokrüptogramm on antud kujul $c_s = (1, encode(s))$, kus s on jaoskonna identifikaatori sõne.

4. küsimuse identifikaatori pseudokrüptogramm on antud kujul $c_q = (1, encode(q))$,
   kus q on küsimuse identifikaatori sõne.

Sellisel juhul defineeritakse laia krüptogrammile vastava avaliku võtmena $((g,1),(g,1),(g,1),(g,1),(g,y))$.

Peatükk 5

Krüpteeritud sedeli elutsükkel

5.1 Krüpteeritud sedeli loomine valijarakenduses

Valija kasutab valijarakendust oma tahte avaldamiseks elektroonilisel kujul ning selle tahteavalduse esitamiseks krüpteeritud sedelina. Töö toimub järgmise eeskirja alusel.

Protsessi sisendiks on valija valik kandidaatide hulgast ning elektroonilise hääletamise süsteemi avalik võti *pk*.

1. Valijarakendus esitab valija valiku elektroonilise tahteavalduse vormingus (lõik 4.1).
2. Valijarakendus laadib ElGamali krüptosüsteemi avaliku võtme, mis on esitatud DER-kodeeritud kujul (lõik 4.2).
3. Valijarakendus interpreteerib elektroonilist tahteavaldust rühma $\mathbb{Z}_p^*$ elemendina (lõik 4.3).
4. Valijarakendus garanteerib, et krüpteerimisele minev avatekst on ruutjääk (lõik 2.3.5).
5. Valijarakendus krüpteerib avateksti (lõik 2.3.2).
6. Valijarakendus kodeerib krüptogrammi krüpteeritud sedeli vormingus (lõik 4.4).

Protsessi väljundiks on krüptogramm *c* ning krüpteerimiseks kasutatud juhuslikkus *r*, mida töödeldakse edasi vastavalt IVXV protokollistikule.

5.2 Krüpteeritud sedeli kontrollimine kontrollrakenduses

Valija kasutab kontrollrakendust muuhulgas veendumaks, et valijarakenduse poolt loodud sedel on korrektselt ja valitud kandidaadi jaoks vormistatud ning krüpteeritud.

Protsessi sisendiks on kogumisteenuse vahendatud DER-kodeeritud krüpteeritud sedel *c*, valijarakenduse vahendatud juhuslikkus *r* ning elektroonilise hääletamise süsteemi avalik võti *pk*.

1. Kontrollrakendus DER-dekodeerib krüpteeritud sedeli ning saab krüptogrammi komponendid u ja v (lõik 4.4).

2. Kontrollrakendus laeb ElGamali krüptosüsteemi avaliku võtme, mis on esitatud DER-kodeeritud kujul, ning saab sealt rühma parameetrid (lõik 4.2).

3. Kontrollrakendus kasutab juhuslikkust r ja algoritmi $Verify$ avateksti m arvutamiseks (lõik 2.3.2).

4. Kontrollrakendus kontrollib, et avatekst m oleks ruutjääk (lõik 2.3.5).

5. Kontrollrakendus teisendab avateksti vajadusel mitteruutjäägiks, tulemuseks on avateksti esitav rühmaelement (lõik 2.3.5).

6. Kontrollrakendus teisendab rühmaelemendi binaaresituse valija elektrooniliseks tahteavalduseks (lõik 4.3).

7. Kontrollrakendus kontrollib elektroonilise tahteavalduse vastavust vormingunõuetele (lõik 4.1).

Protsessi väljundiks on valija elektrooniline tahteavaldus avakujul, mida on võimalik esitada valijale. Igas protsessi sammus võib toimuda viga või esineda mittevastavus, kõik need vead tähendavad kontrollimise ebaõnnestumist.

5.3 Krüpteeritud sedelite tõestatav dekrüpteerimine võtmerakenduses

Valimiskomisjon kasutab võtmerakendust elektrooniliste häälte hulga kokkulugemiseks. Anname ühe hääle koos tõestustega dekrüpteerimise protsessikirjelduse.

Protsessi sisendiks on DER-kodeeritud krüpteeritud sedel c. Eeldame ligipääsu elektroonilise hääletamise süsteemi salajasele võtmele sk.

1. Võtmerakendus DER-dekodeerib krüpteeritud sedeli ning saab krüptogrammi komponendid u ja v (lõik 4.4).

2. Võtmerakendus kasutab salajast võtit sk ja algoritmi $PDec$ avateksti m, sõnumipühendumuse a, võtmepühendumuse b ning vastuse s arvutamiseks (lõik 3.1).

3. Võtmerakendus kontrollib, et avatekst m oleks ruutjääk (lõik 2.3.5).

4. Võtmerakendus teisendab avateksti vajadusel mitteruutjäägiks, tulemuseks on avateksti esitav rühmaelement (lõik 2.3.5).

5. Võtmerakendus teisendab rühmaelemendi binaaresituse valija elektrooniliseks tahteavalduseks (lõik 4.3).

6. Võtmerakendus kontrollib elektroonilise tahteavalduse vastavust vormingunõuetele (lõik 4.1).

7. Võtmerakendus esitab nullteadmustõestuse korrektse dekrüpteerimise kohta ehk lugemistõendi vastavalt vormingule (lõik 4.5).

Protsessi väljundiks on valija elektrooniline tahteavaldus avakujul, mida on võimalik kasutada tulemuse väljaarvutamisel ning lugemistõend. Igas protsessi sammus võib toimuda viga või esineda mittevastavus, kõik need vead tähendavad dekrüpteerimise ebaõnnestumist ning sedel loetakse kehtetuks.

Täiendavalt võib jätkuval töötlemisel osutuda, et korrektselt vormistatud tahteavaldus ei kodeerinud ühtegi valija ringkonnas esitatud kandidaati. Ka sellisel, antud dokumendis mittekäsitletaval juhul loetakse sedel kehtetuks.

5.4 Hääletamistulemuse korrektsuse auditeerimine auditirakenduses

Audiitor kasutab auditirakendust lugemistõendi verifitseerimiseks.

Protsessi sisendiks on lugemistõend ja elektroonilise hääletamise süsteemi avalik võti *pk*.

1. Auditirakendus laadib ElGamali krüptosüsteemi avaliku võtme, mis on esitatud DER-kodeeritud kujul, ning saab sealt rühma parameetrid (lõik 4.2).
2. Auditirakendus dekodeerib lugemistõendi (lõik 4.5).
3. Iga lugemistõendis leiduva elektroonilise tahteavalduse f, krüptogrammi c, sõnumipühendumuse a, võtmepühendumuse b ning vastuse s korral teostab auditirakendus järgmised tegevused.
   (a) Auditirakendus kontrollib elektroonilise tahteavalduse f vastavust vormingunõuetele (lõik 4.1).
   (b) Auditirakendus interpreteerib elektroonilist tahteavaldust f rühma $\mathbb{Z}_p^*$ elemendina (lõik 4.3).
   (c) Auditirakendus garanteerib, et eeldatav avatekst m on ruutjääk (lõik 2.3.5).
   (d) Auditirakendus DER-dekodeerib krüpteeritud sedeli ning saab krüptogrammi komponendid u ja v (lõik 4.4).
   (e) Auditirakendus kasutab algoritmi $PDecVerify$ avalikku võtit *pk*, avateksti m ja krüptogrammi c sõnumipühendumuse a, võtmepühendumuse b ning vastuse s kontrollimiseks. (lõik 3.1).

Kirjandus

[1] Open verificatum. `https://www.verificatum.org/`.

[2] Taher El Gamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *Proceedings of CRYPTO 84 on Advances in Cryptology*, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[3] Mohamad El Laz, Benjamin Grégoire, and Tamara Rezk. Security analysis of elgamal implementations. In Pierangela Samarati, Sabrina De Capitani di Vimercati, Mohammad S. Obaidat, and Jalel Ben-Othman, editors, *Proceedings of the 17th International Joint Conference on e-Business and Telecommunications, ICETE 2020 - Volume 2: SECRYPT, Lieusaint, Paris, France, July 8-10, 2020*, pages 310–321. ScitePress, 2020.

[4] Claus-Peter Schnorr. Efficient Identification and Signatures for Smart Cards. In Gilles Brassard, editor, *CRYPTO*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252. Springer, 1989.