Universelle Verifizierbarkeit des Genfer E-Voting Systems

Dieses Dokument beinhaltet eine Übersicht der zu prüfenden Daten sowie einen Testkatalog der künftigen Verifikationsoftware

Projekt 2
Christian Wenger
Biel, 11. Juni 2018

Berner Fachhochschule
Hochschule für Technik und Informatik
BSc Informatik - Security

Inhaltsverzeichnis

1.Einleitung....2
2.Zu prüfende Daten....3
2.1.Vordefinierte Parameter....3
2.2.Das Bulletin Board....4
3.Testkatalog....7
3.1.Vollständigkeit....7
3.2.Integrität....9
3.3.Konsistenz....11
3.4.Evidenz....12
3.5.Authentizität....13
4.Fazit und Ausblick....13

1. Einleitung

Das Internet ist heute aus unserem Leben nicht mehr wegzudenken. Wir kaufen online ein, holen von Google was wir wissen wollen, ja sogar unser Bewerbungen können wir damit verschicken. Dieser Fortschritt machte auch keinen halt vor der Politik. So begann das Parlament im Jahr 2000 mit den Vorbereitungen für die elektronische Stimmabgabe kurz E-Voting. Zurzeit gibt es zwei vorhandene Systeme. Einmal jenes des Kanton Genf (CHVote) und dasjenige der Schweizerischen Post. Ein grosses Problem dieser Systeme ist die vollständige Nachvollziehbarkeit der einzelnen Schritten. Die sogenannte universelle Verifizierbarkeit ist ein Kernelement der bundesrechtlichen Anforderungen.

Im Rahmen der Projektarbeit 2 studierte ich das E-Voting System des Kanton Genf sehr genau. Als Grundlage diente mir dabei die Spezifikation[1] dieses Systems. In einem ersten Schritt erstellte ich eine Übersicht der öffentlichen vorhanden Daten. Diese werden dann in einem sogenannten Bulletin Bord gespeichert. Es haben jedoch nicht alle Teilnehmer vollen zugriff auf diese Daten.

In einem zweiten Schritt erstellte ich dann aus diesen Daten einen Testkatalog. Diese Tests haben den Zweck zu beweisen, dass alle Daten korrekt sind. Im Rahmen der Bachelor-Thesis werde ich dann ein Programm schreiben, welches diese Tests über die die öffentlichen Daten ausführt.

1 Haenni R, Koenig E. R, Locher P, Dubuis E, 2017. CHVote System Specification

2. Zu prüfende Daten

In diesem Kapitel werden alle Daten aufgeführt welche später für die Tests im Testkatalog verwendet werden. Jeder zu prüfende Parameter verfügt über ein eigenes Zeichen, eine Beschreibung und einen Wertebereich. Auch hier diente mir die Spezifikation als Grundlage.

2.1. Vordefinierte Parameter

Die nachfolgenden Daten sind nicht Teil des Bulletin Boards. Sie spielen für die Sicherheit des Systems jedoch eine entscheidende Rolle. Sie müssen auch nicht für jeden Wahlgang neu gewählt werden. Entscheidend sind die drei Parameter σ, τ und ε.
Der Parameter σ definiert wie viel Rechenpower nötig ist, damit eine Angreifer in polynomialer Zeit die Privatsphäre einer Stimme brechen könnte.
Der Parameter τ definiert wie viel Rechenpower nötig ist, damit eine Angreifer in gleicher weise die Integrität der Stimme brechen könnte.
Der Parameter ε definiert die Wahrscheinlichkeit, das eine Manipulation am System von einem aufrichtigen Teilnehmer entdeckt wird. Alle anderen Parameter können von diesen Drei abgeleitet werden. Zusätzlich habe ich noch die nötigen Zertifikate und dessen öffentliche Schlüssel für die Signaturen in dieser Tabelle aufgeführt. Denn auch diese müssen nicht bei jedem Wahlgang neu erstellt werden

Parameter	**Description**	**Range**
p	Modulo of encryption group $\mathbb{G}_q$	$p \in \mathbb{S}$
g, h	Independent generators of $\mathbb{G}_q$	$g, h \in \mathbb{G}_q \setminus \{1\}$
$\hat{p}$	Modulo of identification group $\mathbb{G}_{\hat{q}}$	$\hat{p} \in \mathbb{P}$
τ	Minimal integrity (bits)	$\tau \in \{4, 80, 112, 128\}$
σ	Minimal privacy (bits)	$\sigma \in \{4, 80, 112, 128\}$
λ	Security level	$\lambda \in \{0, 1, 2, 3\}$
ε	Deterrence factor	$\varepsilon \in \{0.99, 0.999, 0.9999, 0.99999\}$
ℓ	Hash lenght (bits)	$\ell \in \{8, 160, 224, 256\}$
L	Hash lenght (bytes)	$L \in \{1, 20, 28, 32\}$
L_M	Length of OT messages (bytes)	$L_M \in \{2, 40, 56, 64\}$
L_F	Length of finalization codes F_i (bytes)	$L_F \in \{1, 2, 3\}$
$\hat{q}$	Order of $\mathbb{G}_{\hat{q}}$	$\|\|\hat{q}\|\| \geqslant 2\tau$
$\hat{g}$	Generator of $\mathbb{G}_{\hat{q}}$	$\hat{g} \in \mathbb{G}_{\hat{q}} \setminus \{1\}$
p'	Modulo of prime field $\mathbb{Z}'_p$	$\|\|p'\|\| \geqslant 2\tau$
$\hat{q}_x$	Upper bound of secret voting credential x	$\|\|\hat{q}_x\|\| \geqslant 2\tau$
$\hat{q}_y$	Upper bound of secret confirmation credential y	$\|\|\hat{q}_y\|\| \geqslant 2\tau$
n_{max}	Maximal number of candidates	$n_{max} \geqslant 2$
(pk_{Admin}, C_{Admin})	Public key and certificate of election administrator	$pk_{Admin} \in \mathbb{G}_q$, $C_{Admin} \in \text{X.509}$
$((pk_{Auth_j}, C_{Auth_j}))$	Public key and certificate of election authority *j*	$pk_{Auth_j} \in \mathbb{G}_q$, $C_{Auth_j} \in \text{X.509}$, $j \geqslant 1$

2.2. Das Bulletin Board

Die nachfolgenden Daten sind als öffentlich zu betrachten. Wie schon erwähnt kann jedoch definiert werden, wer auf welche Daten zugriff bekommt. Wie der Namen schon ahnen lässt, kann man sich das Bulletin Board wie eine Pinnwand vorstellen. Jedoch dürfen nur die Autoritäten und Administratoren schreibend darauf zugreifen. Bei allen andern ist es zwingend nötig das sie nur lese Rechte haben.

Einer der wichtigsten Parameter ist die ID des Wahlgangs. Denn nur damit lässt sich sicherstellen, dass es keine Vermischungen der Wahlgang Daten gibt. Denn ohne könnte man denn Daten nicht ansehen zu welchem Wahlgang sie gehören. Deshalb muss diese ID mit jeder Nachricht mitgeschickt werden. Ausserdem müssen alle Nachrichten von den Autoritäten und Administratoren signiert werden. So kann man später noch feststellen wer die Nachricht geschickt hat.

Zur Übersicht habe ich die Daten den drei Phasen von der Spezifikation zugeordnet. Diese sind „Vor der Wahl“ (Pree-Election), „Während der Wahl“ (Election) und „Nach der Wahl“ (Post-Election).

Um die Länge von Matrizen darzustellen wird Folgende Definition verwendet:

$$\text{For } X = (x_{ij})_{n \times m}, \text{ then } |X| = (n, m)$$

2.2.1. Vor der Wahl

Parameter	Description	Range
U	Unique election event identifier to protect for election event mismatch.	$U \in \mathbf{A}^*_{\mathbf{ucs}}$
$\mathbf{n} = (n_j)$	Number of candidates in each election	$n_j \geqslant 2, \lvert\mathbf{n}\rvert \geqslant 1$
$\mathbf{c} = (C_i)$	Candidate description	$C_i \in \mathbf{A}^*_{\mathbf{ucs}}, \lvert\mathbf{c}\rvert \geqslant 2$
$\mathbf{k} = (k_j)$	Number of selection in each election	$k_j \geqslant 2, \lvert\mathbf{k}\rvert \geqslant 1$
$\mathbf{v} = (V_i)$	Voter descriptions	$V_i \in \mathbf{A}^*_{\mathbf{ucs}}, \lvert\mathbf{v}\rvert \geqslant 0$
$\mathbf{w} = (\omega_i)$	Assigned counting circles	$\omega_i \geqslant 1, \lvert\mathbf{w}\rvert \geqslant 0$
$\mathbf{E} = (_{ij})$	Eligibility matrix	$e_{ij}\mathbb{B}, \lvert\mathbf{E}\rvert \geqslant (0, 1)$
$\hat{\boldsymbol{D}} = (\hat{\boldsymbol{d}}_{\boldsymbol{j}})$	Public credentials of all voters	$\lvert\hat{\boldsymbol{d}}\rvert \geqslant 0, \lvert\hat{\boldsymbol{D}}\rvert \geqslant 1,$
$\mathbf{pk} = (pk_j)$	Public key for encryption	$pk_j \in \mathbb{G}_q, \lvert\mathbf{pk}\rvert \geqslant 1$
σ_1^{param}	Signature of full election parameters	$\sigma_1^{param} \in \mathbb{B}^\ell \times \mathbb{Z}_q$
σ_2^{param}	Signature of part of election params	$\sigma_2^{param} \in \mathbb{B}^\ell \times \mathbb{Z}_q$
σ_3^{param}	Signature of other part of params	$\sigma_3^{param} \in \mathbb{B}^\ell \times \mathbb{Z}_q$
$\mathbf{s}_{prep} = (\sigma_j^{prep})$	Signatures of public credentials	$\sigma^{prep} \in \mathbb{B}^\ell \times \mathbb{Z}_q, \lvert\mathbf{s}_{prep}\rvert \geqslant 1$
$\mathbf{s}_{kgen} = (\sigma_j^{kgen})$	Signatures of public keys	$\sigma^{kgen} \in \mathbb{B}^\ell \times \mathbb{Z}_q, \lvert\mathbf{s}_{kgen}\rvert \geqslant 1$

2.2.2. Während der Wahl

Diese Phase besteht aus 4 verschiedene Listen. Dabei gilt zu beachten das diese unvollständig sein können. Meine Aufgabe besteht also darin die Gültigen an Hand der ID des Abstimmenden zu finden und zu kontrollieren ob der Abstimmend in allen Listen genau einen gültigen Eintrag hat. Alle andern Einträge dürfen nicht gezählt werden.

Parameter	Description	Range
$\mathbf{A} = \langle (v, \alpha) \rangle$	List of ballots	$\lvert\mathbf{A}\rvert \geqslant 0$

v	Voter id	$v \geqslant 0$
$\alpha = (\hat{x}_v, \mathbf{a}, \pi_\alpha)$	Ballot	$\alpha \in \mathbb{Z}_{\hat{q}} \times (\mathbb{G}_q \times \mathbb{G}_q)^{k'_v} \times ((\mathbb{G}_{\hat{q}} \times \mathbb{G}_q^2) \times (\mathbb{Z}_{\hat{q}} \times \mathbb{G}_q \times \mathbb{Z}_q))$

$\hat{x}_v$	Voter's public credentials	$\hat{x}_v \in \mathbb{G}_{\hat{q}}$
$\mathbf{a} = (a_j)$	Encrypted selection of a voter	$a_j \in \mathbb{G}_q^2$, $\lvert\mathbf{a}\rvert \geqslant 0$
π_α	Proof of validity of ballot	$\pi_\alpha \in (\mathbb{G}_{\hat{q}} \times \mathbb{G}_q^2) \times (\mathbb{Z}_{\hat{q}} \times \mathbb{G}_q \times \mathbb{Z}_q)$

$\mathbf{B} = \langle (v, \beta_j, \sigma_{vj}^{cast}) \rangle$	List of OT responses and signature	$\lvert\mathbf{B}\rvert \geqslant 0$

β_j	OT response	$\beta_j \in \mathbb{G}_q^{k'_v} \times (\mathcal{B}^{L_M})^{nk'_v} \times \mathbb{G}_q$
σ_{vj}^{cast}	Signature of OT response	$\sigma_{vj}^{cast} \in \mathbb{B}^\ell \times \mathbb{Z}_q$

$\mathbf{C} = \langle (v, \gamma) \rangle$	Confirmation list	$\lvert\mathbf{C}\rvert \geqslant 0$

$\gamma = (\hat{y}_v, \pi_\beta)$	Confirmation	$\gamma \in \mathbb{G}_{\hat{q}} \times (\mathbb{G}_q \times \mathbb{Z}_{\hat{q}})$

$\hat{y}_v$	Confirmation credential	$\hat{y}_v \in \mathbb{G}_{\hat{q}}$
π_β	Proof knowledge of $y + y'$	$\pi_\beta \in \mathbb{G}_{\hat{q}} \times \mathbb{Z}_{\hat{q}}$

$\mathbf{D} = \langle (v, \delta_j, \sigma_{vj}^{\text{conf}}) \rangle$	List of finalization and signature	$\lvert\mathbf{D}\rvert \geqslant 0$

δ_j	Finalizations	$\delta_j \in \mathcal{B}^{L_F} \times \mathbb{Z}_q^2$
$\sigma_{vj}^{\text{conf}}$	Signatures of finalizations	$\sigma_{vj}^{\text{conf}} \in \mathbb{B}^\ell \times \mathbb{Z}_q$

2.2.3. Nach der Wahl

Parameter	Description	Range
$\mathbf{E}' = (\mathbf{e}_j)$	Mixed and re-encrypted ballot lists	$\mathbf{e}_j \in (\mathbb{G}_q^2)^N,\ N \geqslant 0,\ \lvert\mathbf{E}'\rvert \geqslant 1$
$\boldsymbol{\pi} = (\pi_j)$	Shuffle proofs	$\pi_j \in (\mathbb{G}_q^3 \times \mathbb{G}_q^2 \times \mathbb{G}_q^N) \times (\mathbb{Z}_q^4 \times \mathbb{Z}_q^N \times \mathbb{Z}_q^N) \times \mathbb{G}_q^N \times \mathbb{G}_q^N$ $N \geqslant 0, \lvert\boldsymbol{\pi}\rvert \geqslant 1$
$\mathbf{B}' = (\mathbf{b}'_j)$	Partial decrypted ballot lists	$\mathbf{e}_j \in (\mathbb{G}_q^2)^N,\ N \geqslant 0,\ \lvert\mathbf{B}'\rvert \geqslant 1$
$\boldsymbol{\pi}' = (\pi'_j)$	Decryption proofs	$\pi'_j \in (\mathbb{G}_q \times \mathbb{G}_q^N) \times \mathbb{Z}_q$ $N \geqslant 0, \lvert\boldsymbol{\pi}'\rvert \geqslant 1$
$\boldsymbol{V} = (v_{ij})$	Election result matrix	$v_{ij} \in \mathbb{B}, v_{ij} = \{\ 1 \text{ if vote } i \text{ contains } j\ \}$ $\lvert\mathbf{V}\rvert \geqslant (0, 2)$
$\boldsymbol{W} = (\omega_{ij})$	Counting circle matrix	$w_{ij} \in \mathbb{B}, \omega_{ij} = \{\ 1 \text{ if } i \text{ is assigned to } j\ \}$ $\lvert\mathbf{W}\rvert \geqslant (0, 1)$
σ^{tally}	Signature of tallying result	$\sigma^{tally} \in \mathbb{B}^\ell \times \mathbb{Z}_q$
$\mathbf{s}_{mix} = (\sigma_j^{mix})$	Signatures of mixed re-encryptions	$\sigma_j^{mix} \in \mathbb{B}^\ell \times \mathbb{Z}_q,\ \lvert\mathbf{s}_{mix}\rvert \geqslant 1$
$\mathbf{s}_{dec} = (\sigma_j^{dec})$	Signatures of partial decryptions	$\sigma_j^{dec} \in \mathbb{B}^\ell \times \mathbb{Z}_q,\ \lvert\mathbf{s}_{dec}\rvert \geqslant 1$

3. Testkatalog

Aus all den zu prüfende Daten galt es nun einen Testkatalog zu erstellen. Dieser ist in fünf Kategorien unterteilt, welche später im Detail erklärt werden. Das Endergebnis ist also die Summe aus allen fünf Ergebnissen der jeweiligen Kategorie.

Später in der Bachelor-Thesis muss ersichtlich sein. Welcher Test welchem Programm Code entspricht. Aus diesem Grund wurde jedem Test eine Eindeutige Nummer zugewiesen. Damit man diese Später im Programm Code verwenden kann.

Wie auch schon beim Bulletin Board wurden die Test auch hier teilweise den Vordefinierten Parameter und den 3 Phasen der Spezifikation zugewiesen. Diese ermöglicht eine bessere Übersicht über die Tests.

Um die Integritäts- und Konsistenz Tests durchführen zu können, müssen noch folgende Parameter definiert werden.

- $\omega = max(\mathbf{w})$
- $t = |\mathbf{n}|$
- $N_E = |\mathbf{v}|$
- $s = |\hat{\boldsymbol{D}}|$
- $n = \sum_{j=1}^{t} n_j$
- $N = |\mathbf{e}_1|$

3.1. Vollständigkeit

In diesem Teil wird überprüft, ob alle erforderlichen Daten vorhanden sind. Denn nur so kann man eine lückenlose Verifizierung gewährleisten. Es werden hier also einfach nochmal alle Daten aufgeführt. Jeder Test überprüft ob ein Parameter vorhanden ist oder nicht. Der Output ist also entweder Wahr oder Falsch.

3.1.1. Vordefinierte Parameter

3.1.1.1. Check for modulo of encryption group p

3.1.1.2. Check for independent generators g, h

3.1.1.3. Check for minimal integrity (bits) τ

3.1.1.4. Check for minimal privacy (bits) λ

3.1.1.5. Check for security level (bits) λ

3.1.1.6. Check for deterrence factor ε

3.1.1.7. Check for hash lengt (bits) ℓ

3.1.1.8. Check for hash lenght (bytes) L

3.1.1.9. Check for length of OT messages (bytes) L_M

3.1.1.10. Check for length of finalization codes F_i (bytes) L_F

3.1.1.11. Check for order of group $\hat{q}$

3.1.1.12. Check for generator of group $\hat{g}$

3.1.1.13. Check for modulo of prime field p'

3.1.1.14. Check for upper bound of secret voting credential $\hat{q}_x$

3.1.1.15. Check for upper bound of secret confirmation credential $\hat{q}_y$

3.1.1.16. Check for maximal number of candidates n_{max}

3.1.1.17. Check for certificate and public key of administrator (pk_{Admin}, C_{Admin})

3.1.1.18. Check for vector of certificate and public key of authorities $((pk_{Auth_j}, C_{Auth_j}))$

3.1.2. Vor der Wahl

3.1.2.1. Check for election identifier U

3.1.2.2. Check for vector of number of candidates in each election $\mathbf{n}$

3.1.2.3. Check for vector of candidate description $\mathbf{c}$

3.1.2.4. Check for vector of number of selection in each election $\mathbf{k}$

3.1.2.5. Check for vector of voter description $\mathbf{v}$

3.1.2.6. Check for vector of assigned counting circles $\mathbf{w}$

3.1.2.7. Check for eligibility matrix $\mathbf{E}$

3.1.2.8. Check for vector of public voter credentials $\hat{\mathbf{D}}$

3.1.2.9. Check for vector of public keys of Authorities $\mathbf{pk}$

3.1.2.10. Check for signature of full election parameters σ_1^{param}

3.1.2.11. Check for signature of part of election parameters σ_2^{param}

3.1.2.12. Check for signature of other part of election parameters σ_3^{param}

3.1.2.13. Check for vector of signature of public credentials $\mathbf{s}_{prep}$

3.1.2.14. Check for vector of signature of public keys $\mathbf{s}_{kgen}$

3.1.3. Während der Wahl

3.1.3.1. Check for ballot list $\langle(v, \alpha)\rangle$

3.1.3.1.1. For all elements check for voter ID and ballot (v, α)

3.1.3.2. Check for OT-response list $\langle(v, \beta_j, \sigma_{ij}^{cast})\rangle$

3.1.3.2.1. For all elements check for :
voter ID, OT-response and signature $(v, \beta_j, \sigma_{ij}^{cast})$

3.1.3.3. Check for confirmation list $\langle(v, \gamma)\rangle$

3.1.3.3.1. For all elements check for voter ID and confirmation (v, γ)

3.1.3.4. Check for finalization list $\langle(v, \delta_j, \sigma_{ij}^{cast})\rangle$

3.1.3.4.1. For all elements check for :
voter ID , finalization and signature $(v, \delta_j, \sigma_{ij}^{cast})$

3.1.4. Nach der Wahl

3.1.4.1. Check for mixed and re-encrypted Ballot lists $\mathbf{E}'$

3.1.4.2. Check for vector of Shuffle Proofs $\boldsymbol{\pi}$

3.1.4.3. Check for vector of partial decrypted Ballot lists $\mathbf{B}'$

3.1.4.4. Check for vector of decryption Proofs $\boldsymbol{\pi}'$

3.1.4.5. Check for election result matrix $\mathbf{V}$

3.1.4.6. Check for counting circle matrix $\mathbf{E}$

3.1.4.7. Check for signature of tallying result σ^{tally}

3.1.4.8. Check for vector of signatures of mixed re-encryptions $\mathbf{s}_{mix}$

3.1.4.9. Check for vector of signatures of partial decryption's $\mathbf{s}_{dec}$

3.2. Integrität

In diesem Teil wird die Integrität der Parameter geprüft. Es wird also geprüft ob die Parameter in sich schlüssig sind. Beispielweise wird geprüft ob sie sich im geforderten Wertebereich befinden. Der Output ist wieder entweder Wahr oder Falsch.

3.2.1. Vordefinierte Parameter

3.2.1.1. Check if $p \in \mathbb{S}$

3.2.1.2. Check if $g, h \in \mathbb{G}q \setminus \{1\}$

3.2.1.3. Check if $\hat{p} \in \mathbb{P}$

3.2.1.4. Check if $||p'|| \geqslant 2\tau$

3.2.1.5. Check if $||\hat{q}|| \geqslant 2\tau$

3.2.1.6. Check if $||\hat{q}_x|| \geqslant 2\tau$

3.2.1.7. Check if $||\hat{q}_y|| \geqslant 2\tau$

3.2.1.8. Check if $n_{max} \geqslant 2$

3.2.1.9. Check if $\tau \in \{4, 80, 112, 128\}$

3.2.1.10. Check if $\sigma \in \{4, 80, 112, 128\}$

3.2.1.11. Check if $\lambda \in \{0, 1, 2, 3\}$

3.2.1.12. Check if $\varepsilon \in \{0.99, 0.999, 0.9999, 0.99999\}$

3.2.1.13. Check if $\ell \in \{8, 160, 224, 256\}$

3.2.1.14. Check if $L \in \{1, 20, 28, 32\}$

3.2.1.15. Check if $L_M \in \{2, 40, 56, 64\}$

3.2.1.16. Check if $L_F \in \{1, 2, 3\}$

3.2.1.17. Check if $\hat{g} \in \mathbb{G}_{\hat{q}} \setminus \{1\}$

3.2.1.18. Check if $pk_{Admin} \in \mathbb{G}_q$, $C_{Admin} \in \mathrm{X.509}$

3.2.1.19. For all $j \in \{1, ..., s\}$, check if $pk_{Auth_j} \in \mathbb{G}_q$, $C_{Auth_j} \in \mathrm{X.509}$

3.2.2. Vor der Wahl

3.2.2.1. Check if $U \in A^*_{\mathbf{ucs}}$

3.2.2.2. Check if $\omega \geqslant 1$

3.2.2.3. Check if $t \geqslant 1$

3.2.2.4. Check if $N_E \geqslant 0$

3.2.2.5. Check if $k = \sum_{j=1}^{t} k_j$

3.2.2.6. Check if $s \geqslant 1$

3.2.2.7. For all $j \in \{1, ..., t\}$, check if $n_j \geqslant 2$

3.2.2.8. For all $j \in \{1, ..., t\}$, check if $k_j \geqslant 1$

3.2.2.9. For all $j \in \{1, ..., t\}$, $i \in \{1, ..., N_E\}$, check if $e_{ij} \in \mathbb{B}$

3.2.2.10. Check if $\sum_{j=1}^{t} e_{ij} \geqslant 1$

3.2.2.11. For all $i \in \{1, ..., n\}$, check if $C_i \in A^*_{\mathbf{ucs}}$

3.2.2.12. For all $i \in \{1, ..., N_E\}$, check if $V_i \in A^*_{\mathbf{ucs}}$

3.2.2.13. For all $i \in \{1, ..., N_E\}$, check if $\omega_i \in \{1, ..., \omega\}$

3.2.2.14. For all $j \in \{1, ..., s\}$, $i \in \{1, ..., N_E\}$, check if $\hat{d}_{ij} = (\hat{x}_{ij}, \hat{y}_{ij})$

3.2.2.15. For all $j \in \{1, ..., s\}$, check if $pk_j \in \mathbb{G}_q$

3.2.2.16. For all $i \in \{1, 2, 3\}$, check if $\sigma_i^{param} \in \mathbb{B} \times \mathbb{Z}_q$

3.2.2.17. For all $j \in \{1, ..., s\}$, check if $\sigma_j^{prep} \in \mathbb{B} \times \mathbb{Z}_q$

3.2.2.18. For all $j \in \{1, ..., s\}$, check if $\sigma_j^{kgen} \in \mathbb{B} \times \mathbb{Z}_q$

3.2.3. Während der Wahl

3.2.3.1. For all elements in $\mathbf{A}$, check if $v \in \{1, ..., N_E\}$

3.2.3.2. For all elements in $\mathbf{A}$, check if $\alpha = (\hat{x}_v, \mathbf{a}, \pi_\alpha)$

3.2.3.2.1. For all elements in α, check if $\hat{x}_v \in \mathbb{G}_{\hat{q}}$

3.2.3.2.2. For all elements in α, check if $a_j = (a_{j,1}, a_{j,2}) \in \mathbb{G}_q^2$

3.2.3.2.3. For all elements in α, check if $\pi_\alpha \in (\mathbb{G}_{\hat{q}} \times \mathbb{G}_q^2) \times (\mathbb{Z}_{\hat{q}} \times \mathbb{G}_q \times \mathbb{Z}_q)$

3.2.3.3. For all elements in $\mathbf{B}$ check if $v \in \{1, ..., N_E\}$

3.2.3.4. For all elements in $\mathbf{B}$ check if $\beta_j \in \mathbb{G}_q^{k'_v} \times (\mathcal{B}^{L_M})^{nk'_v} \times \mathbb{G}_q$

3.2.3.5. For all elements in $\mathbf{B}$ check if $\sigma_{vj}^{cast} \in \mathbb{B}^\ell \times \mathbb{Z}_q$

3.2.3.6. For all elements in $\mathbf{C}$ check if $v \in \{1, ..., N_E\}$

3.2.3.7. For all elements in $\mathbf{C}$ check if $\gamma = (\hat{y}_v, \pi_\beta)$

3.2.3.7.1. For all elements in γ, check if $\hat{y}_v \in \mathbb{G}_{\hat{q}}$

3.2.3.7.2. For all elements in γ, check if $\pi_\beta \in \mathbb{G}_{\hat{q}} \times \mathbb{Z}_{\hat{q}}$

3.2.3.8. For all elements in $\mathbf{D}$ check if $v \in \{1, ..., N_E\}$

3.2.3.9. For all elements in $\mathbf{D}$ check if $\delta_j \in \mathcal{B}^{L_F} \times \mathbb{Z}_q^2$

3.2.3.10. For all elements in $\mathbf{D}$ check if $\sigma_{vj}^{\text{conf}} \in \mathbb{B}^{\ell} \times \mathbb{Z}_q$

3.2.4. Nach der Wahl

3.2.4.1. For all $j \in \{1, ..., s\}$ check if $\mathbf{e}_j \in (\mathbb{G}_q^2)^{N_*}$

3.2.4.2. For all $j \in \{1, ..., s\}$ check if $\pi_j \in (\mathbb{G}_q^3 \times \mathbb{G}_q^2 \times \mathbb{G}_q^N) \times (\mathbb{Z}_q^4 \times \mathbb{Z}_q^N \times \mathbb{Z}_q^N)$

3.2.4.3. For all $j \in \{1, ..., s\}$ check if $\mathbf{b}'_j \in \mathbb{G}_q^N$ $\times \mathbb{G}_q^N \times \mathbb{G}_q^N$

3.2.4.4. For all $j \in \{1, ..., s\}$ check if $\pi'_j \in (\mathbb{G}_q \times \mathbb{G}_q^N) \times \mathbb{Z}_q$

3.2.4.5. For all $j \in \{1, ..., s\}$, $i \in \{1, ..., N\}$ check if $v_{ij} \in \mathbb{B}$

3.2.4.6. For all $j \in \{1, ..., s\}$, $i \in \{1, ..., N\}$ check if $\omega_{ij} \in \mathbb{B}$

3.2.4.7. For all $j \in \{1, ..., s\}$ check if $\sigma_j^{mix} \in \mathbb{B} \times \mathbb{Z}_q$

3.2.4.8. For all $j \in \{1, ..., s\}$ check if $\sigma_j^{dec} \in \mathbb{B} \times \mathbb{Z}_q$

3.2.4.9. Check if $\sigma^{tally} \in \mathbb{B} \times \mathbb{Z}_q$

3.3. Konsistenz

In diesem Teil wird geprüft ob die Parameter zu andern konsistent sind. Wir haben zu Beispiel zwei Vektoren welche die gleiche Länge haben sollten. Nun wird geprüft ob diese wirklich die gleiche Länge haben. Auch hier ist der Output Wahr oder Falsch.

3.3.1. Vordefinierte Parameter

3.3.1.1. Check if $p = 2q + 1$

3.3.1.2. Check if g, h are independent

3.3.1.3. Check if $\hat{p} = k\hat{q} + 1, k \geqslant 2$

3.3.1.4. Check if $\hat{q}_x \leqslant \hat{q}$

3.3.1.5. Check if $\hat{q}_y \leqslant \hat{q}$

3.3.2. Vor der Wahl

3.3.2.1. Check if $|\mathbf{n}| = t$

3.3.2.2. Check if $|\mathbf{k}| = t$

3.3.2.3. Check if $|\mathbf{E}| = (N_E, t)$

3.3.2.4. Check if $|\mathbf{c}| = n$

3.3.2.5. Check if $|\mathbf{v}| = N_E$

3.3.2.6. Check if $|\mathbf{w}| = N_E$

3.3.2.7. Check if $|\hat{\mathbf{D}}| = s$

3.3.2.8. For all $j \in \{1, ..., s\}$, check $|\hat{\mathbf{d}}_j| = N_E$

3.3.2.9. Check if $|\mathbf{pk}| = s$

3.3.2.10. Check if $p_{n+w} \prod_{j=1}^{k} p_{n-j+1} < p$

3.3.3. Während der Wahl

3.3.3.1. For all elements in $\mathbf{A}$, for all α, check if $|\mathbf{a}| = s$

3.3.3.2. For all elements in $\mathbf{B}$, for $\boldsymbol{\beta}_v = \{\beta_{v,1}, ..., \beta_{v,s}\}$, check if $|\boldsymbol{\beta}_{\boldsymbol{v}}| = s$

3.3.3.3. For all elements in $\mathbf{B}$, for $\mathbf{S}_{cast_v} = \{\sigma_{v,1}^{cast}, ...\sigma_{v,s}^{cast}\}$, check if $|\mathbf{S}_{cast_v}| = s$

3.3.3.4. For all elements in $\mathbf{D}$, for $\boldsymbol{\delta}_v = \{\delta_{v,1}, ..., \delta_{v,s}\}$, check if $|\boldsymbol{\delta}_{\boldsymbol{v}}| = s$

3.3.3.5. For all elements in $\mathbf{D}$, for $\mathbf{S}_{\mathrm{conf}_v} = \{\sigma_{v,1}^{\mathrm{conf}}, ...\sigma_{v,s}^{\mathrm{conf}}\}$, check if $|\mathbf{S}_{\mathrm{conf}_v}| = s$

3.3.4. Nach der Wahl

3.3.4.1. Check if $|\mathbf{E}'| = (N, t)$

3.3.4.2. For all $j \in \{2, ..., s\}$, check if $|\mathbf{e}_j| = N$

3.3.4.3. Check if $|\boldsymbol{\pi}| = s$

3.3.4.4. For all $j \in \{1, ..., s\}$, check if $|\boldsymbol{\pi}_j| = N$

3.3.4.5. Check if $|\mathbf{B}'| = s$

3.3.4.6. For all $j \in \{1, ..., s\}$, check if $|\mathbf{b}'_j| = N$

3.3.4.7. Check if $|\mathbf{V}| = (N, n)$

3.3.4.8. Check if $|\mathbf{W}| = (N, \omega)$

3.3.4.9. Check if $|\mathbf{s}_{mix}| = s$

3.3.4.10. Check if $|\mathbf{s}_{dec}| = s$

3.4. Evidenz

In diesem Teil wird geprüft ob die verschieden Kryptographischen Beweisen stimmen. Dazu wird eine Verifizierungsfunktion aufgerufen, die Wahr oder Falsch zurückgibt.

Es handelt sich hierbei um sogenannte Nicht-Interaktive Zero-Knowledge-Beweise. Sie dienen dazu Zu beweisen, dass man Kenntnis von gewissen Parameter/Schlüssel hat. Beispiel „poof of confirmation“ : Hier wir der Beweis erbracht, dass man sowohl das öffentliche „confirmation credential“ und auch das private „vote validity credential“ kennt.

3.4.1. Check proof of validity of ballot π_α

3.4.2. Check proof of confirmation π_β

3.4.3. For all $j \in \{1, ..., s\}$, check shuffle proof π_j

3.4.4. For all $j \in \{1, ..., s\}$, check decryption proof π'_j

3.5. Authentizität

In diesem Teil wird die Gültigkeit der Zertifikate und Signaturen überprüft.
Auch hier wird eine Verifizierungsfunktion aufgerufen, welche wiederum Wahr oder Falsch zurückgibt. Jedes Zertifikate Besitzt einen Gültigkeitszeitraum sowie einen öffentlichen Schlüssel. Mit dem öffentlichen Schlüssel kann dann die Signatur des Zertifikates überprüft werden.

3.5.1. Zertifikate

3.5.1.1. Check validity of Certificate of election administrator C_{Admin}

3.5.1.2. For all $j \in \{1, ..., s\}$, check validity of the Certificates of authorities C_{Auth_j}

3.5.2. Siganturen

3.5.2.1. Check signature of full election parameters σ_1^{param}

3.5.2.2. Check signature of part of election parameters σ_2^{param}

3.5.2.3. Check signature of other part of election parameters σ_3^{param}

3.5.2.4. Check signature of tallying result σ^{tally}

3.5.2.5. For all $j \in \{1, ..., s\}$, check signature of public credentials σ_j^{prep}

3.5.2.6. For all $j \in \{1, ..., s\}$, check signature of public keys σ_j^{kgen}

3.5.2.7. For all $j \in \{1, ..., s\}$, $i \in \{1, ..., N_B\}$ check signature of OT responses σ_{ij}^{cast}

3.5.2.8. For all $j \in \{1, ..., s\}$, $i \in \{1, ..., N_C\}$ check signature of finalization σ_{ij}^{fin}

3.5.2.9. For all $j \in \{1, ..., s\}$, check signatures of mixed re-encryption's σ_j^{mix}

3.5.2.10. For all $j \in \{1, ..., s\}$, check signatures of partial decryption's σ_j^{dec}

4. Fazit und Ausblick

Ich merkte bald, das die Zusammenstellung der Daten und deren Test gar nicht so einfach ist. Denn man musste sich immer wieder Gedanken machen, wie man sie darstellen will. Ausserdem kann es gut sein, das ich ein oder mehre Tests vergessen habe. Dieses Dokument ist also nicht in Stein gemeisselt. Es kann und wird wahrscheinlich noch Änderungen während der Bachelor-Thesis geben.

Als nächster Schritt gilt es sich für eine Programmiersprache zu entscheiden. Wahrscheinlich wird es Python oder Java. Beide haben keine Probleme mit einer Server-Client Umgebung. Dann werde ich mir die nötigen Design-Pattern suchen und eine Softwarearchitektur erstellen. Auch das Framework für die GUI-Programmierung muss noch gefunden werden.