HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

INSTITUTE FOR
NETWORKED SOLUTIONS

Post-Quantum E-Voting

Bachelorarbeit

Abteilung Informatik

Hochschule für Technik Rapperswil

Herbstsemester 2019

Autoren:	Lukas Lätsch, Rolf Furrer, Romeo Spinas
Betreuer:	Prof. Dr. Andreas Steffen
Experte:	Dr. Ralf Hauser
Gegenleser:	Prof. Dr. Daniel Patrick Politze
Projektpartner:	INS, HSR
Abgabedatum:	10.01.2020

Zielsetzung
Die meisten Länder wenden gegenwärtig ein papierbasiertes Wahlsystem an. Der Kostenaufwand solcher Systeme ist relativ gross. Es gibt immer mehr Länder, die an einem E-Voting-System arbeiten. Die Kernpunkte eines E-Voting-Systems sind die Verifiability, Privacy und Transparency bestmöglich zu gewährleisten. Es soll ein zukunftssicheres E-Voting-System erstellt werden. Die folgenden Punkte heben das E-Voting-System besonders heraus:

- Der Verschlüsselungsalgorithmus soll gegenüber Quantum-Computern resistent sein, somit wird der Algorithmus immer noch sicher sein, nachdem Quanten-Computer genügend leistungsfähig sind.
- Die Stimmen können verschlüsselt aufsummiert werden, dadurch kennt der Stimmenzähler das Ergebnis und die Identität des Wählers nicht.
- Eine einzelne Partei sollte die Daten nicht alleine entschlüsseln können.

Das entwickelte E-Voting-System soll für Abstimmungen eingesetzt werden, wie zum Beispiel an Generalversammlungen oder auch an nationalen Abstimmungen. Es gibt eine Library, die die oben genannten Punkte und Kernpunkte erfüllt, basierend auf diesen wird in dieser Bachelorarbeit ein E-Voting-System entwickelt. Bei der Arbeit wird zudem auf den architektonischen Aufbau der Software geachtet, welcher für die Sicherstellung des Vertrauens in das Wahlsystem entscheidend ist.

Ergebnis
Die Einarbeitung in die Verschlüsselungstheorie und den Aufbau der Library wurde in der Studienarbeit [1] erarbeitet und darauf aufbauend ist in der Bachelorarbeit das Wissen vertieft worden. Aufbauend auf der Library ist erfolgreich ein E-Voting-System entwickelt worden, welches alle zentralen und die meisten optionalen Anforderungen abdeckt. Das E-Voting-System benötigt mindestens die drei eigenständigen Applikationen Coordinator, Bulletin Board und Trustee sowie einen Voter um abzustimmen. Die Komponenten sind mit mehreren JSON-Schnittstellen untereinander verbunden. Die Rechenoperationen auf dem Bulletin Board sind aufwendig, deswegen wurde von Beginn der Entwicklung an ein Augenmerk auf die Parallelisierung der Operationen gelegt.

Ausblick
Der Grundaufbau des E-Voting-Systems ermöglicht es, eine Abstimmung durchzuführen, jedoch ist das System noch nicht in einer realen Abstimmung einsetzbar. Die verschiedenen Teilapplikationen können sich untereinander noch nicht eindeutig identifizieren. Die Wahlberechtigung der Wähler zu prüfen, wird durch das E-Voting-System noch nicht unterstützt. Die abstimmungsbezogenen Daten können in der Zukunft anstatt in einer Datenbank in einer Blockchain abgespeichert werden. Es sind schon einige Optimierungen zur Beschleunigung des E-Voting-Systems unternommen worden, es könnte aber zusätzlich die Verteilung des Bulletin Boards auf mehrere Rechner implementiert werden.

Inhaltsverzeichnis

1. Einleitung

1.1 Übersicht

Abstimmungen werden in den meisten Ländern papierbasiert durchgeführt. Der Aufwand und die Kosten einer Durchführung einer Wahl sind substanziell. Eine Abstimmung elektronisch durchzuführen, E-Voting genannt, stellt aufgrund diverser Gründe eine Herausforderung dar. Die wichtigsten Gründe sind die folgenden:

- **Verifiability**: Es ist möglich die Abstimmungen zu verifizieren.
- **Privacy**: Das Stimmgeheimnis ist gewahrt.
- **Transparency**: Es kann nachvollzogen und überprüft werden, ob die Abstimmung korrekt durchgeführt wurde.

Um die oben erwähnten Punkte mit E-Voting garantieren zu können, muss besonders auf Folgendes geachtet werden:

- **Cryptography**: Es muss ein Verschlüsselungsalgorithmus verwendet werden, der Verifiability und Privacy garantieren kann.
- **Platform Security**: Die verwendeten Systeme, auf denen das E-Voting System betrieben wird, müssen sicher genug sein, um die Privacy zu garantieren.
- **Collaboration**: Es braucht verschiedene unabhängige Organisationen, um die Transparency sicherzustellen.

Es ist das Ziel dieser Bachelorarbeit, einen Verschlüsselungsalgorithmus einzusetzen, der die Verifiability, Privacy und Transparency erfüllt. Das nachfolgende Kapitel 1.2 beschreibt im Detail, was gefordert ist.

1.2 Aufgabenstellung

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

Bachelorarbeit 2019

Post-Quantum E-Voting

Studenten: Rolf Furrer, Lukas Lätsch, Romeo Spinas

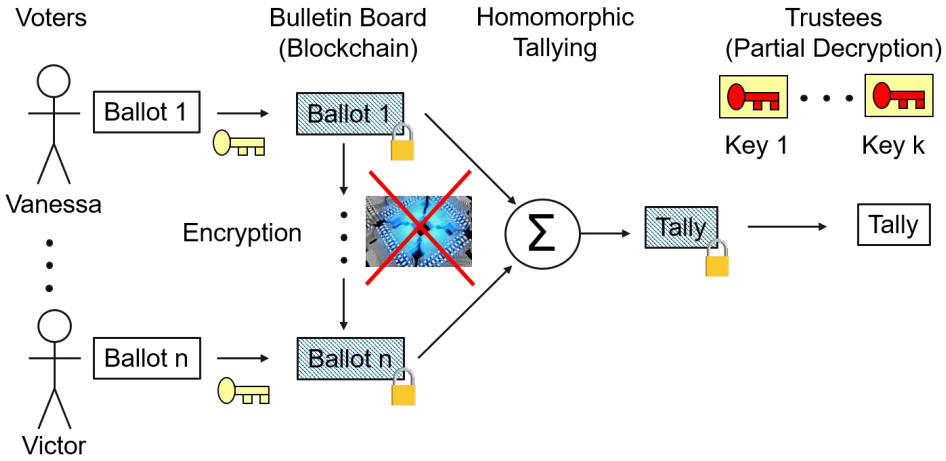
Betreuer: Prof. Dr. Andreas Steffen

Ausgabe: Montag, 16. September 2019

Abgabe: Freitag, 10. Januar 2020

Einführung

E-Voting über das Internet ist ein aktuelles Thema. Die zur Zeit in der Schweiz und anderen Ländern benutzten Systeme sind nicht sicher, weil die zentralen Server gehackt und damit die Endergebnisse manipuliert werden können. Seit mehr als zehn Jahren gibt es aber sichere Systeme, bei dem jeder Wähler auf einem durch eine Blockchain gesicherten Bulletin-Board jederzeit seine in verschlüsselter Form abgegebene Stimme verifizieren kann und auch die Berechnung des Gesamtresultats aus der Summe aller abgebenen Stimmen überprüfen kann. Dabei wird das Wahlgeheimnis vollständig gewahrt.



Zur Zeit werden diese Ende-zu-Ende verifizierbaren Systeme auf der Basis der RSA und ElGamal Public Key Verfahren realisiert. Es wird erwartet, dass ca. in zehn Jahren diese Algorithmen durch Quantencomputer geknackt werden können. Da die in der Blockchain öffentlich publizierten verschlüsselten Stimmzettel aus Datenschutzgründen mindestens 20-30 Jahre geheim bleiben müssen, sollte man schon jetzt die Realisierung von neuartigen E-Voting Systemen in Angriff nehmen, die resistent gegen Attacken durch Quantencomputer sind.

2016 wurde durch die Doktorandin Ilaria Chillotti ein solches Verfahren [1] beschrieben und 2019 dahin erweitert, dass der Schlüssel mit dem das Abstimmungsresultat entschlüsselt werden kann, auf mehrere unabhängige Parteinen (Trustees) verteilt werden kann [2]. Die dazu

benötigten neuartigen Kryptoalgorithmen werden in der MK-TFHE Open Source Library [3] zur Verfügung gestellt.

Die Grundlagen für das MK-TFHE Post-Quantum Wahlsystem wurden in einer vorausgegangenen Bachelorarbeit erarbeitet. In dieser Bachelorarbeit soll eine praktische Demo-Applikation entwickelt werden, mit der kleinere Abstimmungen durchgeführt werden können.

Aufgabenstellung

- Erabeiten einer verteilten Architektur, bestehend aus einer Voter-, Trustee-, Bulletin-Board und Coordinator-Applikation.
- Die Kommunikation zwischen den Komponenten soll auf RESTful Schnittstellen und die Datenstruktur auf JSON basieren.
- Unterstützung einer Ja/Nein Abstimmung und der Wahl eines Vertreters aus N Kandidaten, beide Varianten mit der Möglichkeit der Stimmenthaltung.
- Generierung der benötigten Schlüssel durch mehrere unabhängigeTrustees.
- Auflistung der abgegebenen verschlüsselten Stimmen durch das Bulletin-Board.
- Keine Verwendung von Public Key Signaturen für die gegenseitige Authentisierung, sowie keine Implementation einer Blockchain für das Bulletin Board.
- **Optional:** Mehrere Bulletin-Boards können parallel die homomorphe Aufsummierung der Stimmen vornehmen, um eventuelle Manipulationen aufzudecken.

Links

[1] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène: “An homomorphic LWE based E-voting Scheme”, 2016
https://ilachill.github.io/papers/CGGI16a-An_homomorphic_LWE_based_E-voting_Scheme.pdf

[2] Hao Chen, Ilaria Chillotti, and Yongsoo Song; Multi-Key Homomophic Encryption from TFHE, 2019
https://eprint.iacr.org/2019/116

[3] Ilaria Chillotti, MK-TFHE: Multi-Key Homomorphic Encryption from TFHE
https://github.com/ilachill/MK-TFHE

Rapperswil, 16. September 2019

Prof. Dr. Andreas Steffen

1.3 Vorarbeit Studienarbeit

Diese Bachelorarbeit baut auf dem Wissen der Studienarbeit[5] auf, in welcher die TFHE-Library [3], welche im Kapitel 1.5 näher beschrieben ist, von einer eher funktionalen C/C++ Library in eine objektorientierte C++ Library auf dem Standard C++17 umgeschrieben wurde. Dabei wurde darauf geachtet, dass die Private Keys nicht kopiert werden und dass sie im Speicher nach Verwendung überschrieben werden. Die umgeschriebene Library wurde dann verwendet, um eine Applikation mit verschiedenen lokalen Komponenten zu erstellen, welche eine Abstimmung durchführt. Die Applikation konnte auch parametrisiert werden.

Die TFHE-Library, auf welcher die Library der Studienarbeit[5] basiert, wurde ursprünglich für Berechnungen auf der Cloud konzipiert. Die Grundidee war, dass verschlüsselte zu berechnende Daten von einem Benutzer an die Cloud geschickt werden können, diese dort berechnet werden, ohne das sie entschlüsselt werden müssen. Danach konnten die Daten wieder an den Nutzer geschickt werden und dort entschlüsselt werden. Die TFHE-Library unterstützte keine funktionierende Implementierung der Aufsplittung der Teilschlüssel.

In der Bachelorarbeit wurde die überarbeitete TFHE-Library namens MK-TFHE-Library verwendet, die die oben genannte Thematik beseitigt. Es war möglich in der TFHE-Library, aus dem Cloud-Key, die Private Keys zurückzugewinnen. Bei der MK-TFHE-Library wurde eine strikte Aufteilung der LWE und TLWE Keys auf die einzelnen Trustees mittels der Multi-Key Erweiterung ermöglicht. Kurz vor Abschluss der Semesterarbeit wurde diese neue Version der Library veröffentlicht.
Die TFHE und MK-TFHE unterscheiden sich im Sourcecode erheblich, weshalb entschieden wurde, das verteilte E-Voting System für die Bachelorarbeit auf der MK-TFHE Library aufzubauen. Der Einsatz der MK-TFHE Library bringt folgende Vor- und Nachteile mit sich:

- Vorteile
  - Es musste keine Zeit aufgewendet werden, um die Unterschiede zwischen der alten und neuen Library zu analysieren und die Anpassungen auf die in der Studienarbeit[5] erarbeitete Library umzusetzen.
  - Es schleichen sich keine Fehler beim Umbau der Library ein.
- Nachteile
  - Anpassungen an der Library können mit einem neuen Pull-Request nicht schnell gelöst werden, da die Maintainer nur sehr zögerlich darauf reagieren.
  - Das Memory der Private Keys kann innerhalb der Library nicht gelöscht werden, da dies Änderungen an der Library benötigt.

1.4 Konkurrenzsysteme

Eine Wahl kann bereits heute durchgeführt werden. Eine herkömmliche Wahl auf Papier erfüllt die jetzigen Vorgaben und wird in der Schweiz seit dem 19. Jhd. eingesetzt. Einige E-Voting-Lösungen sind in der Testphase und werden in absehbarer Zeit wahrscheinlich eingeführt. In vielen Ländern sind zudem Wahlmaschinen im Einsatz, die im weitesten Sinne einen E-Voting-Client mit dessen Vor- und Nachteilen darstellen.

1.4.1 Papier

Die klassische Wahl auf Papier erfolgt in mehreren Schritten. Die Wahlunterlagen werden zusammengestellt und an die Wähler gesendet. Der Wähler authentisiert sich in persona oder mittels Unterschrift bei der Abgabe der Stimme. Durch die Urne werden die Stimmen anonymisiert. Wahlhelfer zählen die Stimmen aus und geben das Resultat bekannt.
Die Auszählung der Stimmen kann vom Wähler selbst nicht überprüft werden. Bei Verdacht auf Unstimmigkeiten müssen alle Stimmen nachgezählt werden.

1.4.2 E-Voting: Post

Die Schritte in einem E-Voting-Verfahren sind ähnlich wie jene in analogen Verfahren. Als Beispiel kann das E-Voting-System der Post [1] dienen. Nach der Konfiguration des Urnengangs werden Zugangs- und Prüfcodes für die Wähler erstellt und mit den Unterlagen zugesendet. Die Wähler stimmen mit den Zugangscodes ab. Die Urnen werden entschlüsselt, die Stimmen zusammen gezählt und die Resultate publiziert.
In der Vergangenheit konnte die Korrektheit der Auszählung in Tests nicht in allen Fällen bewiesen werden und eine Manipulation konnte dadurch nicht ausgeschlossen werden.

1.5 MK-TFHE Grundlagen

Die *Multi-Key fast Fully Homomorphic Encryption over the Torus* (MK-TFHE) ist eine Zusammensetzung der GSW (Gentry, Sahai, Waters) und der LWE (Learning with errors) Verschlüsselung. Der Name beinhaltet dabei verschiedene Hinweise auf ihre Eigenschaften auf welche teilweise in den folgenden Kapiteln eingegangen wird.

1.5.1 Torus

Logisch gesehen liegt der Wertebereich der Verschlüsselung zwischen $-\frac{1}{2}$ und $\frac{1}{2}$. Für die logischen Operationen ist es aber einfacher, sich den Bereich in Achtel einzuteilen. Das negative Ende des Wertebereiches wird dabei mit dem positiven Ende verbunden. So bildet sich ein Torus, der ähnlich einer Uhr, sich immer im Kreis dreht und den Wertebereich nie verlassen kann, da bei einem positiven Überlauf der Übertrag von der negativen Seite neu beginnt.

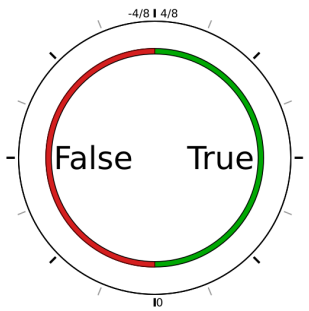


Abbildung 1.1: Torus

So ist beispielsweise $\frac{2}{8} + \frac{4}{8} = -\frac{2}{8}$. Visuell kann man sich das wie folgt vorstellen:

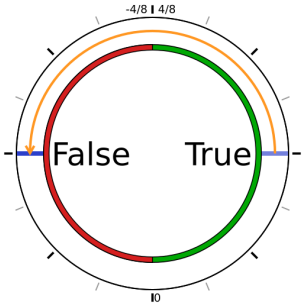


Abbildung 1.2: Torus Rechenbeispiel

Für die binären Operationen repräsentiert ein Torus einen binären Wert. Dabei gilt:

$$v(t) = \begin{cases} true & t > 0 \\ false & t <= 0 \end{cases}$$

1.5.2 Learning with Errors

Die Learning with Errors (LWE) Eigenschaft der Verschlüsselung dient der Sicherheit. Um ein Faktorisieren des Schlüssels zu verhindern, muss zu jedem verschlüsselten B-Element Rauschen addiert werden. So wird aus dem B kein exakter Wert sondern ein Wertebereich.

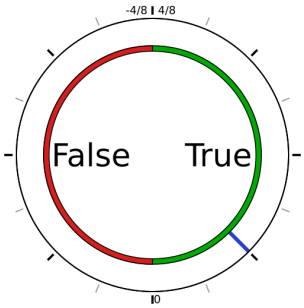


Abbildung 1.3: +0.25 Torus exakt

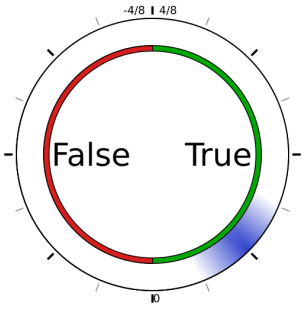


Abbildung 1.4: +0.25 Torus mit Rauschen

Das Problem dieser Eigenschaft ist, dass sich das Rauschen mit den logischen Operationen vergrössert. Das Rauschen darf einen Sechzehntel nicht überschreiten, um die binäre Entscheidbarkeit nicht zu verlieren. Es ist möglich, einen Key-Switch durchzuführen um das Rauschen zu entfernen, jedoch ist diese Operation sehr rechenaufwändig. Im Optimalfall sollten also möglichst wenig Key-Switches erfolgen.

1.5.3 Multi-Key

Die Multi-Key Eigenschaft erlaubt es, mit einem aus mehreren Teilschlüsseln generierten Schlüssel zu verschlüsseln. Dies ermöglicht es in einem Public-Private-Key-Verfahren Daten mit dem Public Key zu verschlüsseln, sodass nur alle Besitzer eines Private Keys gemeinsam Zugriff auf die Daten erhalten können. Eine Einzelperson allein kann die Daten nicht entschlüsseln.

1.5.4 Fully Homomorphic Gates

Homomorphe Verschlüsselung ist eine Form der Verschlüsselung, die es erlaubt, mit verschlüsselten Daten zu rechnen, sodass nach dem Entschlüsseln dasselbe Ergebnis vorliegt, wie bei derselben Rechenoperation mit den nicht verschlüsselten Daten.
Für das E-Voting System sind dabei vor allem die XOR- und AND-Gatter von Interesse, da diese für einen Halbaddierer benötigt werden.

1.5.4.1 XOR-Gatter

Zur Berechnung eines XOR-Gatters ist die Rechenoperation $XOR_{Konstante} + 2a + 2b = r$ notwendig. Daraus resultiert die folgende Wahrheitstabelle:

Binär			Torus		
a_b	b_b	$r_b = a_b \oplus b_b$	a_t	b_t	$r_t = \frac{1}{4} + 2a_t + 2b_t$
false	false	false	$-\frac{1}{8}$	$\frac{-1}{8}$	$-\frac{1}{4}$
true	false	true	$\frac{1}{8}$	$-\frac{1}{8}$	$\frac{1}{4}$
false	true	true	$-\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{4}$
true	true	false	$\frac{1}{8}$	$\frac{1}{8}$	$-\frac{1}{4}$

Tabelle 1.1: XOR Tabelle

$$true \oplus true = false \implies \frac{1}{4} + 2 * \frac{1}{8} + 2 * \frac{1}{8} = -\frac{1}{4}$$

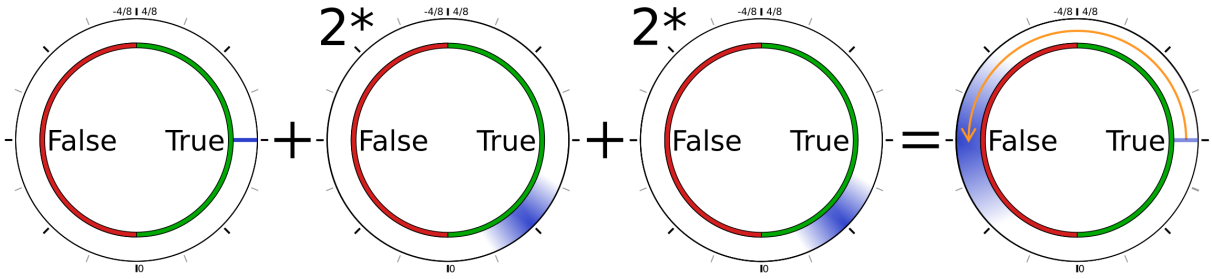


Abbildung 1.5: XOR Beispiel

1.5.4.2 OR-Gatter

Zur Berechnung eines OR-Gatters ist die Rechenoperation $OR_{Konstante} + a + b = r$ notwendig. Daraus resultiert die folgende Wahrheitstabelle:

Binär			Torus		
a_b	b_b	$r_b = a_b \vee b_b$	a_t	b_t	$r_t = \frac{1}{8} + a_t + b_t$
false	false	false	$-\frac{1}{8}$	$-\frac{1}{8}$	$-\frac{1}{8}$
true	false	true	$\frac{1}{8}$	$-\frac{1}{8}$	$\frac{1}{8}$
false	true	true	$-\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$
true	true	true	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{3}{8}$

Tabelle 1.2: OR Tabelle

$$true \vee true = true \implies \frac{1}{8} + \frac{1}{8} + \frac{1}{8} = \frac{3}{8}$$

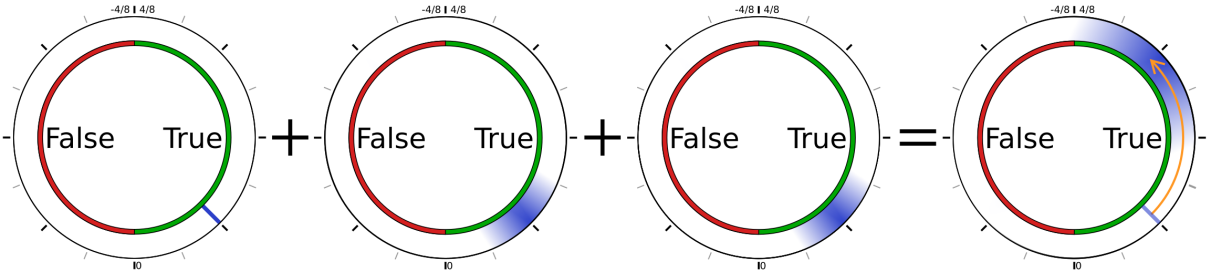


Abbildung 1.6: OR Beispiel

1.5.4.3 AND-Gatter

Zur Berechnung eines AND-Gatters ist die Rechenoperation $AND_{Konstante} + a + b = r$ notwendig. Daraus resultiert die folgende Wahrheitstabelle:

Binär			Torus		
a_b	b_b	$r_b = a_b \wedge b_b$	a_t	b_t	$r_t = -\frac{1}{8} + a_t + b_t$
false	false	false	$-\frac{1}{8}$	$-\frac{1}{8}$	$-\frac{3}{8}$
true	false	false	$\frac{1}{8}$	$-\frac{1}{8}$	$-\frac{1}{8}$
false	true	false	$-\frac{1}{8}$	$\frac{1}{8}$	$-\frac{1}{8}$
true	true	true	$\frac{1}{8}$	$\frac{1}{8}$	$\frac{1}{8}$

Tabelle 1.3: AND Tabelle

1.5.5 Definieren von Parametern und Schlüsselgenerierung

Als zu definierende Parameter gibt es bei der symmetrischen Verschlüsslung und Entschlüsselung lediglich die Schlüssellänge. Im folgenden Beispiel ist die Schlüssellänge 10 und sieht wie folgt aus:

$$key = [1,0,0,1,0,0,1,1,0,1]$$

1.5.6 Verschlüsselung von Daten

Zur Verschlüsselung müssen zunächst die Daten in ein LWE-Sample transformiert werden. Ein LWE-Sample besteht aus einem Vector (a) von der Grösse der Schlüssellänge an zufällig gewählten Zahlen von der Grösse des minimalen Int bis zum maximalen Int, für das Beispiel von $-\frac{4}{8}$ bis $\frac{4}{8}$:

$$a = [-\frac{2}{8}, \frac{4}{8}, -\frac{1}{8}, -\frac{3}{8}, \frac{4}{8}, -\frac{3}{8}, \frac{1}{8}, -\frac{3}{8}, \frac{2}{8}, -\frac{3}{8}]$$

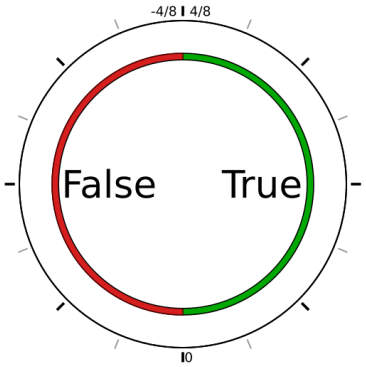


Abbildung 1.7: Torus Mapping

Zusätzlich gibt es ein Element B, das die Daten enthält. Dafür gibt es die folgende Zuweisung: eine logische Null (false) wird zu $-\frac{1}{8}$ und logisch Eins (true) wird zu $\frac{1}{8}$. Es soll die binäre Zahl 10 verschlüsseln, hätten wir beispielsweise die folgenden zwei LWE-Samples:

- Erstes LWE-Sample:

$$a = [-\frac{2}{8}, \frac{4}{8}, -\frac{1}{8}, -\frac{3}{8}, \frac{4}{8}, -\frac{3}{8}, \frac{1}{8}, -\frac{3}{8}, \frac{2}{8}, -\frac{3}{8}], b = \frac{1}{8}$$

- Zweites LWE-Sample:

$$a = [-\frac{4}{8}, \frac{1}{8}, -\frac{2}{8}, \frac{3}{8}, -\frac{3}{8}, -\frac{1}{8}, -\frac{2}{8}, \frac{4}{8}, -\frac{2}{8}, -\frac{1}{8}], b = -\frac{1}{8}$$

Nun wird noch das B-Element verschlüsselt. Hierfür werden die A-Komponenten mit dem Schlüssel multipliziert und zum B-Element addiert, wie nachfolgend illustriert wird.

$$B' = B + \begin{bmatrix} K_1 & K_2 & K_3 & K_4 & K_5 & K_6 & K_7 & K_8 & K_9 & K_{10} \end{bmatrix} * \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \\ A_7 \\ A_8 \\ A_9 \\ A_{10} \end{bmatrix}$$

In diesem Beispiel gäbe das die folgenden Samples:

- Erstes **verschlüsseltes** LWE-Sample:

$$a = [-\frac{2}{8}, \frac{4}{8}, -\frac{1}{8}, -\frac{3}{8}, \frac{4}{8}, -\frac{3}{8}, \frac{1}{8}, -\frac{3}{8}, \frac{2}{8}, -\frac{3}{8}], b = -\frac{2}{8}$$

- Zweites **verschlüsseltes** LWE-Sample:

$$a = [-\frac{4}{8}, \frac{1}{8}, -\frac{2}{8}, \frac{3}{8}, -\frac{3}{8}, \frac{-1}{8}, -\frac{2}{8}, \frac{4}{8}, -\frac{2}{8}, -\frac{1}{8}], b = 0$$

1.5.7 Entschlüsselung der Daten

Um die Daten wieder entschlüsseln zu können, muss nun die A-Komponente wieder von der B-Element abgezogen werden.

$$B = B' - \begin{bmatrix} K_1 & K_2 & K_3 & K_4 & K_5 & K_6 & K_7 & K_8 & K_9 & K_{10} \end{bmatrix} * \begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \\ A_7 \\ A_8 \\ A_9 \\ A_{10} \end{bmatrix}$$

So kommt man wieder zu den initialen LWE-Samples. Zum Schluss muss man entscheiden, ob es sich um eine logische Eins oder Null handelt. Dafür kann man nach $B > 0$ vergleichen, wobei $B > 0 \iff Eins$ und $B \leqslant 0 \iff Null$ entspricht.

1.5.8 Multi-Key TFHE

Diese Art der Verschlüsselung erlaubt es, mehrere Teilschlüssel zu verwenden, um eine Nachricht zu verschlüsseln. Dabei können verschiedene Parteien (P) Key-Vektoren generieren. Des Weiteren muss die A-Komponente P mal die Schlüssellänge (N) sein . Das B-Element wird wie folgt berechnet:

$$b' = b + \sum_{p=0}^{P} \sum_{n=0}^{N} key_p[n] * a[p * N + n]$$

Dies ermöglicht auch die Public-Private-Key-Verschlüsselung, da das B-Element für jede Partei vorberechnet werden kann und anstatt der Vektor Multiplikation auch das Vorberechnete B zu den Daten addiert werden kann.

1.5.9 Counter

Um eine Wahl zählen zu können, benötigt der Stimmenzähler pro Wahloption einen eigenen Zähler. Zudem muss der Stimmzettel in einen Vektor codiert sein, sodass alle Elemente null sind, ausser dasjenige am Index der gewählten Option. Somit sieht ein Stimmzettel mit 8 Optionen und der Wahl 6 wie folgt aus $[0, 0, 0, 0, 0, 1, 0, 0]$.
Bei der Initialisierung der Wahl wird pro Wahloption ein Vektor pro Option erstellt. Die Elemente dieses Vektors repräsentieren die Bits, welche zusammen eine binäre Zahl ergeben. Da verschlüsselte

Rechenoperationen sehr aufwendig sind, ist es sinnvoll, diesen Vektor mit nur einem Element zu initialisieren und ihn nach Bedarf zu vergrössern.
Die Aufsummierung der Stimmen kann mit einem Half-Adder erreicht werden, indem das A des ersten Bits als Input das Bit aus dem Stimmzettel entgegennimmt. Das B ist ein Feld im Vektor des Counters. Das Carry-Flag wird mit dem A des nächsten Bits verbunden während das Sum-Flag zum neuen Wert des Feldes im Vektor wird.

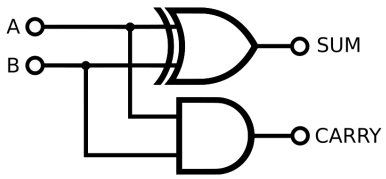


Abbildung 1.8: Half-Adder

A	B	SUM	CARRY
false	false	false	false
true	false	true	false
false	true	true	false
true	true	false	true

Tabelle 1.4: Half-Adder Tabelle

1.5.10 Codierung des Stimmzettel

Der Stimmzettel in unserem System ist eine binäre Codierung der Wahloptionen. Die Codierung ist ein dicht gepackter Code, bei dem jeder Wert eine gültige Wahloption oder die Stimmenthaltung repräsentiert. Bei drei Optionen werden somit zwei Bit benötigt. Bei der 0b00 bis 0b10 die Optionen eins bis drei sind, während die 0b11 die Stimmenthaltung ist. Dies schliesst aus, dass eine Stimme für mehrere Optionen abgegeben werden kann. Die binäre Repräsentation wird zum Versenden in LWE-Samples verschlüsselt, sodass das Stimmgeheimnis gewahrt werden kann.

1.5.11 Expander

Die Stimmenzähler erhalten von den Wählern einen dicht gepackten Code. Sie selbst benötigt jedoch eine Codierung, bei der dieselbe Anzahl Bits wie Optionen vorhanden sind. Zudem müssen alle Bits auf false gesetzt ausser dasjenige der gewählten Option. Für diese Umformung ist der Expander zuständig.
Zur Decodierung der Wahl wird zuerst ein Vektor der Länge der nächsten Zweierpotenz der Anzahl Optionen angelegt. Für sechs Wahloptionen ergibt sich so ein Vektor der Länge 8. Dieser Vektor wird mit Einsen initialisiert.

$$initial = [1, 1, 1, 1, 1, 1, 1, 1]$$

Als nächstes werden Filtervektoren erstellt. Diese Vektoren entsprechen derselben Länge, wie der initial Vektor. Es werden $log_2(anzahl\ optionen)$ Filtervektoren benötigt. Die Vektoren haben folgende Struktur:

$$filter_1 = [1,0,1,0,1,0,1,0], filter_2 = [1,1,0,0,1,1,0,0], filter_3 = [1,1,1,1,0,0,0,0]$$

Anschliessend wird bitweise XOR mit den Filtervektoren und der Stimme durchgeführt. Dabei wird der $filter_1$ mit dem Least-Significant-Bit der Stimme (LWE-Samples) und der $filter_3$ mit dem Most-Significant-Bit verrechnet. Für die Stimme 0b101 ergäbe das die Filter:

$$filter_1 = [0,1,0,1,0,1,0,1], filter_2 = [1,1,0,0,1,1,0,0], filter_3 = [0,0,0,0,1,1,1,1]$$

Wenn nun die Filtervektoren mit dem Initialenvektor bitweise AND-verknüpft werden, entsteht der von den Countern gewünschte Vektor:

$$res = initial \wedge filter_1 \wedge filter_2 \wedge filter_3$$

$$[0,0,0,0,0,1,0,0] = [1,1,1,1,1,1,1,1] \wedge [0,1,0,1,0,1,0,1] \wedge [1,1,0,0,1,1,0,0] \wedge [0,0,0,0,1,1,1,1]$$

Dieser Lösungsansatz ist für jede Anzahl an Optionen einsetzbar. Es gäbe noch effizientere Schaltungen, jedoch sind diese Spezialisierungen nur für je einen bestimmten Fall anwendbar.

2. Analyse

2.1 Funktionale Anforderungen

Die funktionalen Anforderungen wurden mit Use Cases abgesteckt.

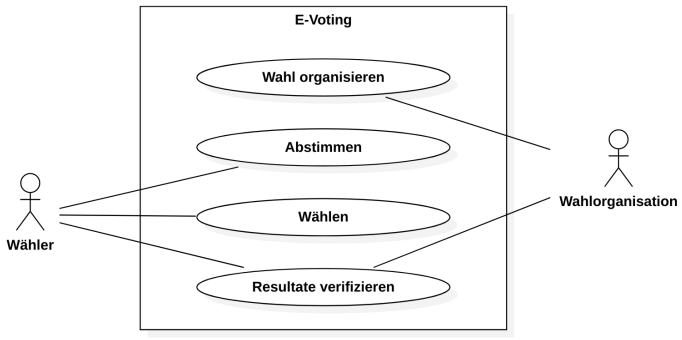


Abbildung 2.1: Use Cases

2.1.1 Wähler

Use Case ID:	1
Use Case Name:	Abstimmen
Actors:	Wähler
Beschreibung:	Ein Wähler möchte an einer Abstimmung teilnehmen. Er kann sich zwischen Ja, Nein und Enthaltung entscheiden. Er kann zwischen verschiedenen Kandidaten auswählen.
Preconditions:	Der Wähler ist wahlberechtigt.
Postconditions:	Die Stimme ist im Sinne des Wählers gezählt.
Normaler Ablauf:	1. Der Wähler möchte seine Stimme abgeben. 2. Die Wahlorganisation liefert eine Liste mit Ja, Nein oder Stimmenthaltung und eine Public Key Liste. 3. Der Wähler wählt eine der Möglichkeiten aus. 4. Der Wähler schickt den ausgefüllten und verschlüsselten Wahlzettel ab.
Priorität:	Mittel
Frequency of Use:	Jeder Wähler einmal
Business Rules:	Jeder Wähler darf nur einmal eine Stimme abgeben.

Tabelle 2.1: Usecase: Abstimmen

Use Case ID:	2
Use Case Name:	Wählen
Actors:	Wähler
Beschreibung:	Ein Wähler möchte an einer Wahl teilnehmen mit einer Auswahl von Kandidaten oder Optionen.
Preconditions:	Der Wähler ist wahlberechtigt.
Postconditions:	Die Wahl ist im Sinne des Wählers gezählt.
Normaler Ablauf:	1. Der Wähler möchte seine Wahl abgeben. 2. Die Wahlorganisation liefert eine Liste mit allen Kandidaten / Optionen und Stimmenthaltung sowie eine Public Key Liste. 3. Der Wähler wählt einen der Kandidaten / Optionen aus. 4. Der Wähler schickt den ausgefüllten und verschlüsselten Wahlzettel ab.
Priorität:	Mittel
Frequency of Use:	Jeder Wähler einmal
Business Rules:	Jeder Wähler darf nur einmal eine Wahl abgeben.

Tabelle 2.2: Usecase: Wählen

Use Case ID:	3
Use Case Name:	Resultate verifizieren
Actors:	Wähler, Wahlorganisation
Beschreibung:	Ein Wähler oder ein Kandidat möchte nach einer Wahl das Ergebnis verifizieren können.
Preconditions:	Die Wahl ist abgeschlossen und die Resultate veröffentlicht.
Postconditions:	-
Normaler Ablauf:	1. Der Wähler will die Korrektheit der Wahl überprüfen. 2. Der Wähler loggt sich in den Wahlclient ein. 3. Die Wahlorganisation liefert eine Statusanzeige zu seiner Stimme.
Priorität:	Mittel
Frequency of Use:	Nach Abschluss einer Wahl
Business Rules:	-

Tabelle 2.3: Usecase: Resultate Verifizieren

2.1.2 Wahlorganisation

Use Case ID:	4
Use Case Name:	Wahl organisieren
Actors:	Wahlorganisation
Beschreibung:	Wahlorganisation möchte eine Wahl duchführen.
Preconditions:	-
Postconditions:	-
Normaler Ablauf:	1. Die Wahlorganisation sammelt die Informationen über alle zur Wahl stehenden Abstimmungen und Wahlen. 2. Die Wähler werden von der Wahlorganisation über die Möglichkeiten informiert und erhalten die Wahl- bzw Stimmzettel. 3. Die Wahlorganisation bekommt die ausgefüllten Stimmzettel von den Wählern. 4. Die Wahlorganisation summiert die Stimmen verschlüsselt auf. 5. Die Wahlorganisation validiert das Resultat und veröffentlicht es.
Priorität:	Mittel
Frequency of Use:	Einmal pro Wahl
Business Rules:	-

Tabelle 2.4: Usecase: Wahl organisieren

2.2 Nichtfunktionale Anforderungen

In den nichtfunktionalen Anforderungen ist definiert, welche Eigenschaften das System haben muss oder sollte.

Privacy
Stimmen müssen nach dem Absenden anonym behandelt werden.

- Für Externe darf es nicht möglich sein, eine Stimme zu deanonymisieren.

Security
Die Datenintegrität soll soweit möglich sichergestellt werden.

- Das E-Voting System muss zur Verhinderung von Betrug mehrere unabhängige Interessensgruppen (z.B. Kantone, Gemeinden, Wahlbeobachter, Parteien, NGOs, Gewerkschaften) unterstützen, sodass die Integrität sichergestellt werden kann.
- Es braucht verschiedene unabhänige Stimmenzähler, um Manipulationen zu erkennen.
- Sensitive Daten müssen im Memory überschrieben werden.

Reliability
Das System muss eine hohe Zuverlässigkeit bieten.

- Während der Wahlperiode muss ein Wähler jederzeit in der Lage sein, seine Stimme abzugeben.
- Keine Stimme darf im System abhanden kommen.
- Die Stimmabgabe darf nicht manipuliert werden können.

Plattform
Der Zugang soll den Benutzern nicht unnötig erschwert werden.

- Das Stimmabgabeprogramm muss auf Personal-Computern mit GNU/Linux und Windows lauffähig sein.

Performance
Die Benutzer müssen die Stimmabgabe in sinnvoller Zeit abschliessen können, ein Warten des Benutzers soll vermieden werden.

- Das Stimmabgabeprogramm muss auf Personal-Computern innerhalb von 5 Sekunden übertragen werden können.

Bedienbarkeit
Es muss eine einfache Oberfläche zur Abgabe der Stimme haben. Die Benutzer sollen möglichst ohne Hilfe das System benutzen können.

- 90% der neuen Benutzer können die Wahl ohne externe Hilfe durchführen.

2.3 Umsysteme

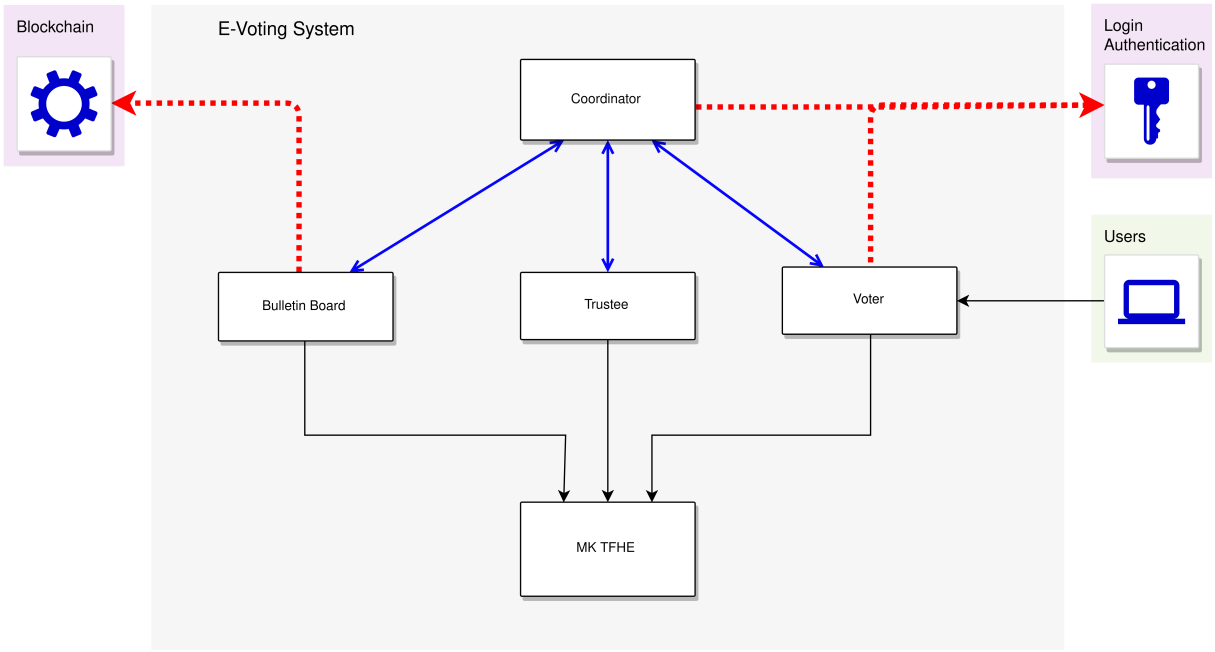


Abbildung 2.2: Externe Schnittstellen

2.3.1 User-Authentisierung

Das MK-TFHE E-Voting System ist selbst nicht in der Lage, die Stimmberechtigung der Benutzer zu verifizieren. Deshalb wird eine externe Schnittstelle benötigt, die über diese Information verfügt. Eine mögliche Option ist beispielsweise die Authentifizierung über die SwissID.

2.3.2 Blockchain

Bei einer Abstimmung ist Nachvollziehbarkeit eine wichtige Eigenschaft eines Systems. In der momentanen Implementation werden Datenbanken als Datenspeicher verwendet. Zukünftig soll zu diesem Zweck eine Blockchain eingesetzt werden, um die langfristige Persistenz der Daten sicher zu stellen. Diese Blockchain muss allerdings öffentlich sein und kann nicht Bestandteil unseres Systems sein. Eine Möglichkeit wäre beispielsweise die Datenspeicherung mit Ethereum über Smartcontracts zu lösen.

3. Design

3.1 Domainanalyse

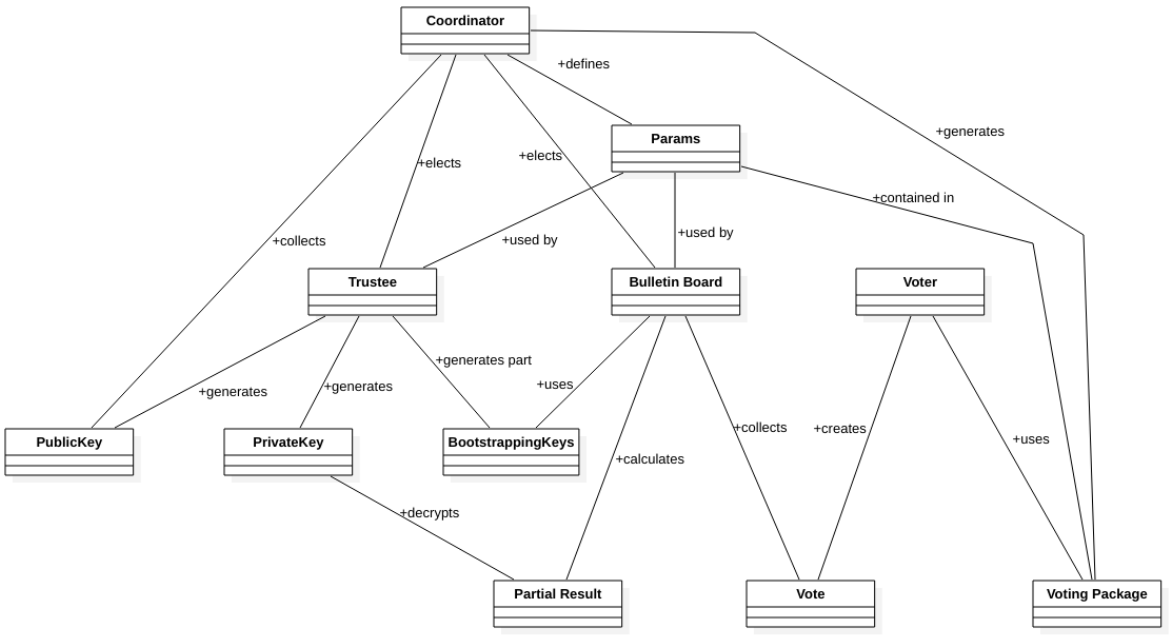


Abbildung 3.1: Domainanalyse

Bei der Domainanalyse wurden Erkenntnisse aus der Studienarbeit [5] beigezogen. Bei der Analyse ergaben sich vier Hauptkomponenten.

3.1.1 Coordinator

Der Coordinator wird durch den Organisator der Wahl betrieben, er ist verantwortlich für die Durchführung der Wahl. Er definiert die Wahloptionen, die Trustees und die Bulletin Boards. Nach der Abstimmung veröffentlicht der Coordinator die Wahlergebnisse. Wenn es mehrere Bulletin Boards gibt, entscheidet er, nach dem Prinzip des Byzantine Generals Problem, dass die Wahl nur bei einer 2/3 Mehrheit als gültig erklärt wird. Er ist zudem der Entrypoint für die Voter.

3.1.2 Voter

Die Voter sind stimmberechtigte Wähler, sie haben die Möglichkeit, an einer Abstimmung teilzunehmen. Zu diesem Zweck füllen sie einen leeren Stimmzettel (Vote) aus, wie im Kapitel 1.5 MK-TFHE erläutert kann dieser nur eine gültige Stimme enthalten. Der ausgefüllte Stimmzettel wird mit 12 öffentlichen Schlüsseln jedes Trustees verschlüsselt. Die Anzahl der verwendeten Schlüssel wird, wie in Anhang F erläutert, zu Beginn der Wahl aufgrund der Sicherheitsanforderungen gewählt. Der Voter muss dabei mindestens einem Trustee vertrauen.

3.1.3 Trustee

Die Trustees generieren als Vorbereitung auf eine Abstimmung je einen privaten Schlüssel und zahlreiche öffentliche Schlüssel. Zudem stellen alle Trustees gemeinsam den Bulletin Boards einen

Key-Switch-Key zur Verfügung. Wenn eine Wahl abgeschlossen ist, erhalten alle Trustees die Wahlresultate, die sie nur gemeinsam entschlüsseln können.

3.1.4 Bulletin Board

Das Bulletin Board sammelt alle Wahlzettel und addiert sie parallel auf. Es errechnet das Wahlresultat, das es aber nicht selber entschlüsseln kann. Es kann zur Verifizierung des Resultates mehrere Bulletin Boards geben.

3.2 Kommunikationsanalyse

Die Kommunikation findet zwischen den verschiedenen Komponenten statt, die im vorherigen Kapitel 3.1 Domainanalyse erwähnt wurden. Sie ist in die drei Phasen Vorbereitung, Durchführung und Abschluss unterteilt.

3.2.1 Vorbereitungen der Wahl

Eine Wahl muss wie folgt vorbereitet werden:

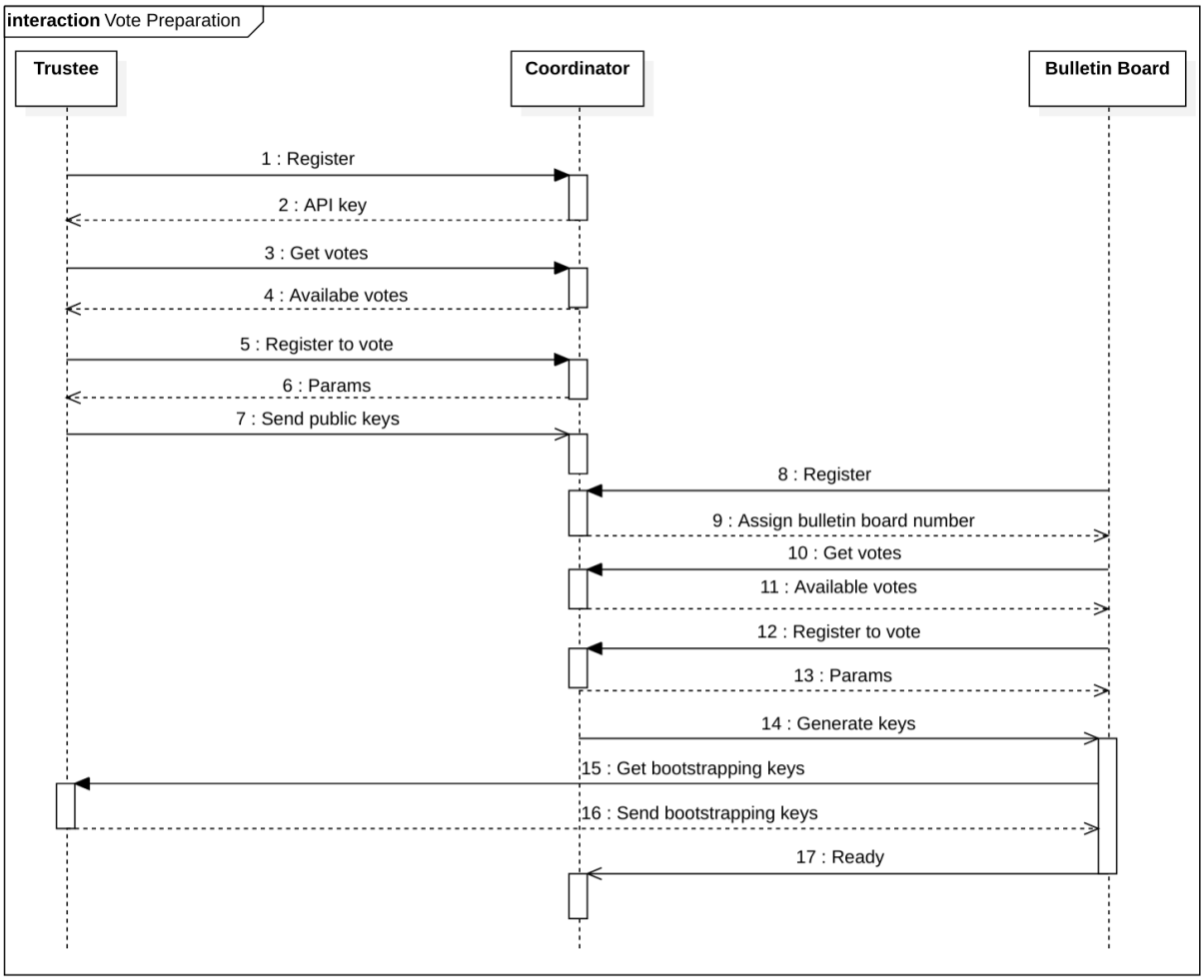


Abbildung 3.2: Kommunikationsdiagramm: Vorbereitungen der Wahl

1. Der Trustee registriert sich beim Coordinator mit Hostname und Port.
2. Der Coordinator erteilt dem Trustee eine eindeutige ID und einen API-Key.
3. Der Trustee fordert eine Liste der Wahlen an.
4. Eine Übersicht der Wahlen wird zum Trustee gesendet.
5. Der Trustee registriert sich für eine bestimmte Wahl.
6. Der Coordinator liefert die notwendigen Daten, wie zum Beispiel die Schlüssellängen. Der Trustee generiert die benötigten Schlüssel.
7. Die Trustees melden sich, wenn sie die Schlüsselgenerierung abgeschlossen haben. Der Coordinator stellt das Wahlpaket zusammen.
8. Analog zu den Trustees müssen sich auch die Bulletin Boards beim Coordinator registrieren (dies kann parallel zu 1 erfolgen).
9. Die Bulletin Boards erhalten eine eindeutige ID und einen API-Key.
10. Das Bulletin Board fordert eine Liste der Wahlen an.
11. Eine Übersicht der Wahlen wird zum Bulletin Board gesendet.
12. Das Bulletin Board registriert sich für eine bestimmte Wahl.
13. Der Coordinator liefert die notwendigen Daten.
14. Wenn die Schlüsselgenerierung der Trustees abgeschlossen ist, werden die Bulletin Boards aufgefordert ihre Schlüssel zu generieren.
15. Die Bulletin Boards fordern die Bootstrapping-Keys an.
16. Die Trustees senden die generierten Schlüssel.
17. Die Bulletin Boards melden dem Coordinator, dass sie zur Wahl bereit sind.

3.2.2 Durchführung der Wahl

Die Durchführung der Wahl verläuft wie folgt:

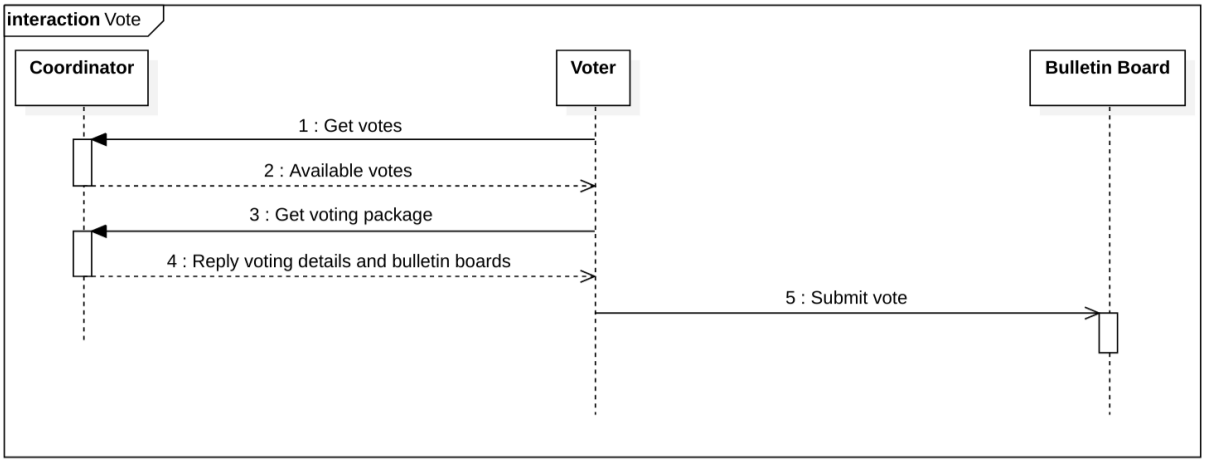


Abbildung 3.3: Kommunikationsdiagramm: Durchführung der Wahl

1. Ein Voter, der an einer Wahl teilnehmen will, kann sich die notwendigen Informationen beim Coordinator beschaffen.
2. Der Coordinater liefert Informationen zur Wahl sowie die Hostnamen und Ports der Bulletin Boards.
3. Der Voter sendet seinen ausgefüllten und verschlüsselten Stimmzettel den Bulletin Boards.

3.2.3 Abschluss der Wahl

Der Abschluss der Wahl wird wie folgt kommuniziert:

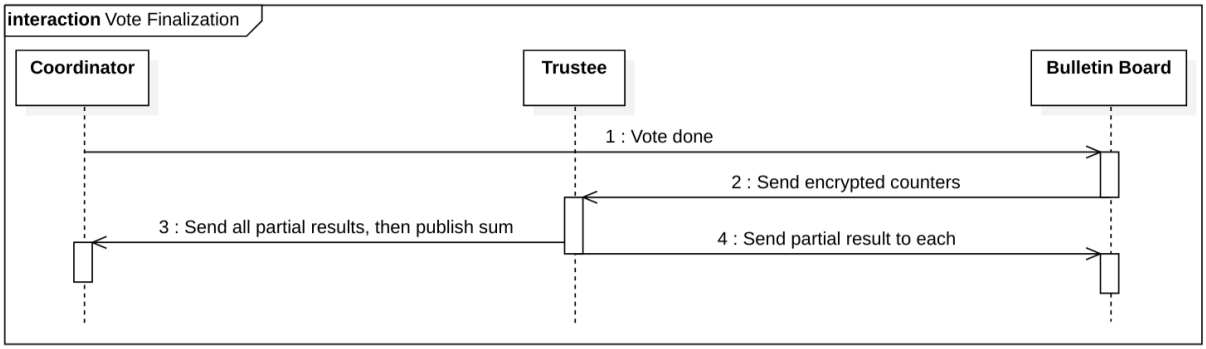


Abbildung 3.4: Kommunikationsdiagramm: Abschluss der Wahl

1. Nach Ablauf der Wahlfrist initiiert der Coordinator die Abschaltung des Systems.
2. Die Bulletin Boards senden ihre Daten an die Trustees zur Entschlüsselung.
3. Jeder Trustee entschlüsselt seinen Teil und sendet die Teilresultate zurück zu den Bulletin Boards.
4. Alle Trustees senden die Teilresultate an den Coordinator, dieser berechnet das Gesamtresultat.

4. Implementation

4.1 Evaluation Technologien

Um geeignete Technologien zu finden, die den Anforderungen genügen, wurde jeweils eine Technologieevaluation durchgeführt. Für die wichtigen Komponenten, wie die graphische Oberfläche des Voters und die RESTful-HTTP-Schnittstelle, wurde die Machbarkeit der verschiedenen Varianten mittels Beispielapplikationen getestet. Für die Auswahl weiterer Komponenten wurde auf die Vorkenntnisse der Teammitglieder und derzeitige 'best pactices' gesetzt.

4.1.1 GUI Framework

Für die Darstellung der anstehenden Wahlen und der Auswahl der Option benötigt der Voter eine Eingabemöglichkeit. Da die Hauptplattform Linux war, zogen GTK und Qt gegen die webbasierte Lösung React.js und das C++-Framework Webtoolkit, das eine Webseite ausliefert, ins Rennen. Die webbasierten Lösungen schieden wegen nicht ausreichenden Zufallszahlengeneratoren in JavaScript aus. Die Wahl fiel auf GTK, da mit der standardmässigen Assistant-Klasse eine Abarbeitung der Wahlen einfach zu realisieren war. Die Anbindung an das restliche Projekt war durch die näher am C++-Standard liegenden Klassen leichter möglich als in Qt, was den Entwicklungaufwand verringern sollte.

Anforderungen
Das GUI soll dem Voter ermöglichen, eine Abstimmung zu tätigen. Es ist essentiell, dass das Programm Zugriff auf genügend Entropie hat, sodass die eingerechneten Keys möglichst true-random ausgewählt werden. Das gewählte Framework muss folgende Views darstellen können:

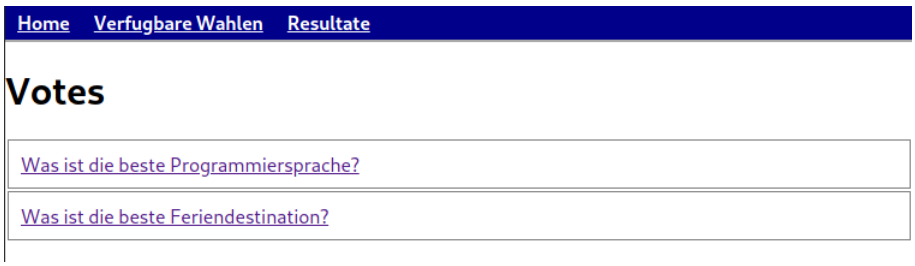


Abbildung 4.1: Entwurf 'Wahl einer Abstimmung'

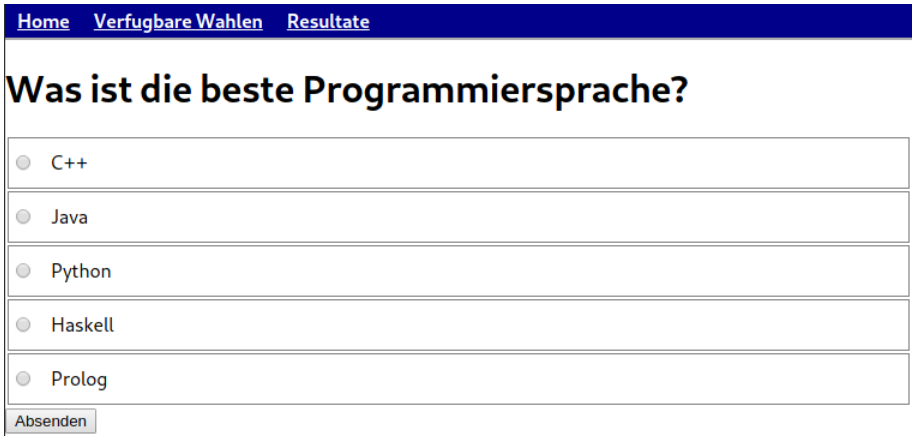


Abbildung 4.2: Entwurf 'Abstimmung'

1. Auswählen einer bestimmten Wahl
2. Den Abstimmungszettel ausfüllen

Es gelten folgende weitere Kriterien:

- Die Ansichten müssen aus einem JSON-Objekt[1] dynamisch generiert werden können.
- Die GUI-Applikation ist kompatibel mit Windows und Linux.
- Die Applikation funktioniert ohne die Installation von Third Party Anwendungen oder kann mithilfe eines Installers installiert werden.

Die optionalen Anforderungen müssen nicht eingehalten werden, jedoch wäre eine Unterstützung wünschenswert, da es damit einfacher ist, zusätzliche Features umzusetzten.

- Ein Wizard, der alle offenen Wahlen sequenziell durchführen kann.
- Das Framework ist aktuell und hat reguläre Git Commits.
- GUI-Elemente können in Komponenten gegliedert werden und sind wiederverwendbar.
- Das Framework trennt Daten, Funktion und Visuelles durch eine saubere Architektur (z.B. MVC).
- Das Framework ist einfach zu verstehen und es gibt eine gute Dokumentation sowie Beispiele.

4.1.1.1 Qt Library

Die Qt-Library steht unter der LGPL License. Diese Library bietet einen sehr grossen Umfang von Funktionen. Sie ermöglicht auch das Erstellen von GUIs. Ein GUI, das mit Hilfe der Qt Library erstellt wird, kann entweder mit dem älteren Qt-Widgets erstellt werden oder mit Qt Quick, welches sich seit Qt 5 in der Version 2 befindet.

Qt-Widgets
Qt-Widgets wird vor allem für Desktop GUIs verwendet. Der Funktionsumfang ist sehr gross. Mit dem QtCreator können GUIs graphisch erstellt werden. Die dynamischen Teile der GUI Elemente müssen im C++ Source Code erstellt werden. Qt ist eine Library, die oft eingesetzt wird. Zum Beispiel ist ein Grossteil der KDE Oberfläche in Qt geschrieben.

Qt Widgets basiert auf C++. Eine GUI Applikation die mit Qt geschrieben ist, läuft mit wenig Aufwand auf Windows, OSX und Linux. Es ist auch nicht allzu schwer, weitere Systeme zu unterstützen.

Qt-Widgets ist sehr solide, jedoch ist davon auszugehen, dass in Zukunft Qt-Widget durch Qt Quick abgelöst wird, und deswegen nicht mehr weiter entwickelt wird.

Qt Quick 2
Qt Quick 2 kann für Desktop und Mobile Applikationen verwendet werden. Es ist eine eher neuere Technologie im Vergleich zu Qt-Widgets. Qt Qucik 1 wurde im Jahr 2010 veröffentlicht.

Qt Quick 2 kann mit C++ verwendet werden. Es ist auch möglich, für einen Teil der Applikation JavaScript zu verwenden. Die Oberfläche wird bei Qt Quick in Qt Modelling Language (QML) geschrieben, welche deklarativ und CSS ähnlich ist. Qt Quick kann auch mit Qt3D verwendet werden,

[1] RFC 8259: https://tools.ietf.org/html/rfc8259

um Objekte in 3D zu rendern.

Der Einstieg in Qt Quick und QML ist herausfordernd und zeitaufwändig, da das Framework viele Möglichkeiten bietet.

4.1.1.2 Web Toolkit

Web Toolkit (Wt) ist ein Cross-Plattform Webframework, welches kostenfrei unter der GNU GPL oder kostenpfichtig unter anderer Lizenz nutzbar ist.

Wt ist in C++ geschrieben. Die Einbindung im Code wird über C++ Klassen, so genannte Widgets gelöst. Es generiert aus C++-Code HTML/Ajax-Code. Die Seiten werden im integrierten Server gehostet. Ein Webbrowser wird genutzt um das GUI anzuzeigen.

4.1.1.3 React

React oder auch ReactJs ist eine JavaScript Library, die durch Facebook und dessen Community gewartet wird.

Die Library basiert auf JavaScript.

Die zufällige Wahl der Schlüssel in JavaScript ist ein Problem, welches in Kapitel **??** näher untersucht wird. Eine Möglichkeit ist es, dies in WebAssembly zu lösen. Eine Onlinesuche hat jedoch ergeben, dass WebAssembly gesandboxed ist und dadurch keinen Zugriff auf eine Entropiequelle des darunterliegenden Systems zulässt.

4.1.1.4 GTK

GTK ist ein GUI-Toolkit, dass kostenfrei unter GNU LGPL einsetzbar ist.

GTK ist in C geschrieben. Die Einbindung in C++ ist mit gtkmm (auch LGPL) über C++-Klassen möglich. Die Fenster werden über das X-Window-System angezeigt.

4.1.1.5 JavaScript

Es soll die Qualität der Randomness von JavaScript abgeklärt werden, da laut dem Betreuer die Randomness von JavaScript nicht genügend ist. Zudem muss darauf geachtet werden, dass der Seed möglichst zufällig gewählt wird, da sonst von einem Angreifer die gewählten Public Keys durch die Voter Applikation mit wenig Aufwand nachvollzogen werden können.
Es ist eine Herausforderung online gute Quellen zu finden, welche die Qualität der Randomness, die JavaScript mitbringt, beschreibt. Die Webseite von David Bau erklärt die Thematik und bietet auch eine mögliche Lösung an. Eine Client JavaScript Library von ihm kann auf Github unter dem Link `https://github.com/davidbau/seedrandom` gefunden werden, welche unter der MIT License steht.

Entropy Accumulation
Das Sammeln von genügend Entropy mit JavaScript ist eine Herausforderung. Laut dem Paper *Symmetric Cryptography in Javascript*[4], besteht folgender Umstand: *"Since we extract 2 bits of entropy per mouse move sample, we need 128 samples to generate a 256-bit AES key. Our data show that the median time for the generator to be seeded to the default level of 128 samples is 9, 28 and 41 seconds on the survey, forum and blog, respectively."*. Wenn diese Aussage im Falle einer

JavaScript basierten Client Lösung auf die Voter Applikation angewendet wird, ist die Usability der Applikation stark beeinträchtigt, da man mehr als 40 Sekunden die Computermaus bewegen muss, bevor die benötigte Entropy vorhanden ist.
Um die Wartezeit zu verringern muss deshalb eine externe Entropy Quelle verwendet werden. Diese kann online durch einen Service zur Verfügung gestellt werden. Dabei muss dem Service vertraut werden können. Es gibt verschiedene Service-Anbieter. Man könnte diese auch kombinieren oder einen eigenen vertrauenswürdigen Service aufsetzen. Die erwähnte seedrandom.js Library würde diese Funktionen unterstützen. Dabei müsste man auch dem Author der Library vertrauen.

Beurteilung
Aufgrund der oben erwähnten Punkte besteht entweder:

- eine lange Wartezeit,
- man muss selbst einen Service aufbauen
- oder man müsste dem Author der seedrandom.js Library und deren Abhängigkeiten vertrauen.

Gemäss dieser Punkte ist es eher davon abzuraten, eine Client Implementation in JavaScript umzusetzen, es sei denn, die erwähnten Problematiken werden systematisch angegangen.

4.1.1.6 Gesamtbeurteilung

Im Kapitel 4.1.1.5 JavaScript ist eruiert worden, ob JavaScript eine geeignete Entropiequelle bietet. Technisch ist eine Lösung mit JavaScript möglich, jedoch gefährden die Lösungsansätze die Sicherheit des E-Voting Systems oder beeinträchtigen die Usability des GUI. Deswegen kommt React für eine Client-Applikation nicht in Frage.
Das Webkit wird im Browser gestartet, was mögliche Angriffsflächen darstellt, wenn der Zugriff nicht ausschliesslich auf den Localhost limitiert werden kann. Das Web Toolkit generiert eine lokale Webseite, welches sowohl die Nachteile einer lokalen Applikation als auch die Nachteile einer Webseite mit sich bringt. Deswegen kommt es ebenfalls nicht in Frage. Die Qt-Widget Variante wird möglicherweise in der Zukunft von Qt Quick abgelöst, was nicht ideal ist. Qt Quick zeichnet sich durch die architektonische Teilung von View, View Model und Daten aus. Beim Evaluieren der Variante war es sehr aufwendig eine Demo zu erstellen, da viele Methoden möglich sind. Aus den erwähnten Gründen sind Qt-Widgets und Qt Quick ausgeschlossen worden. GTK 3.0 hat in seiner einfachen Handhabung und den vorhandenen Konstrukten überzeugt, um einen Wizard zu erstellen. Der Wizard überzeugt auch visuell, wie in der Abbildung 4.3 zu sehen ist.

GTK wird auch weiterentwickelt. Die Version 4.0 wird im Jahr 2020 erwartet. Die Wahl für die GUI Library fällt deshalb auf GTK.

Abbildung 4.3: Entwurf einer Beispielapplikation einer Abstimmung mit GTK

Kriterium	Qt Widget	Qt Quick 2	Web Toolkit	React	GTK
Auswählen einer Wahl	Nicht getestet	Nicht getestet	Erfüllt	Erfüllt	Nicht getestet
Abgeben einer Wahl	Erfüllt	Erfüllt	Erfüllt	Erfüllt	Erfüllt
Dynamische Generierung aus JSON	Erfüllt	Erfüllt	Erfüllt	Erfüllt	Erfüllt
OS Kompatibilität	Erfüllt	Erfüllt	Erfüllt	Erfüllt	Erfüllt
Einfache Distribution	4	4	3	5	4
Ansicht des Status der Stimmen	Erfüllt	Erfüllt	Erfüllt	Erfüllt	Erfüllt
Voting Wizard	Erfüllt	Nicht getestet	Nicht getestet	Nicht getestet	Erfüllt
Das Framework ist aktuell	Erfüllt	Erfüllt	Erfüllt	Erfüllt	Erfüllt
Komponentenbildung	Erfüllt	Erfüllt	Erfüllt	Erfüllt	Erfüllt
Architektur des Framework	4	5	3	4	3
Einsteigerfreundlich	3	2	3	5	4

Tabelle 4.1: Vergleich der GUI-Frameworks/Libraries: Wobei 1 als niedrigster Punktzahl und 5 die Maximalpunktzahl ist

Die Tabelle 4.1 zeigt einen detalierteren Vergleich der GUI-Frameworks/Libraries auf. Nach der Implementierung der Beispielapplikation wurden die verschiedenen Technologien verglichen.

4.1.2 Web Client/Server

Die Verbindung der einzelnen Komponenten wird mit einer RESTful-HTTP-Schnittstelle realisiert. Für die Implementation dieser Schnittstellen wurde ein Framework benötigt. Als Kriterien für die Auswahl waren sowohl der Programmieraufwand als auch der Funktionsumfang miteinbezogen worden. Die für die C++ Komponenten war CppRestSdk die beste Lösung, da sowohl das Anbieten als auch das Konsumieren einer Schnittstelle durch das Framework abgedeckt wird. Der Programmieraufwand war überschaubar. Eine Library zur Konvertierung der Daten in JSON war ebenfalls enthalten.

4.1.2.1 Anforderungen

Bei der Auswahl des Frameworks ist es zentral, dass die nötigen Kriterien unterstützt werden, welche nachfolgend aufgeführt sind.

Das Framework

- stellt sicher, dass die MK TFHE Params in JSON parsen können,
- kann mit Listen von Integer (Keys) umgehen,
- kann HTTP POST Requests durchführen,
- unterstützt GNU/Linux,
- ist kostenfrei.

Die optionalen Anforderungen müssen nicht eingehalten werden, jedoch wäre eine Unterstützung wünschenswert, da es damit einfacher ist, zusätzliche Features umgesetzten.

Das Framework

- ist aktuell. Es hat zum Beispiel reguläre Git Commits,
- unterstützt eine verschlüsselte Verbindung mit mindestens TLS 1.2 Version,
- kann einfach in einen Docker Container eingebettet werden,
- kann ein kleines Docker Image wie Alpine Linux verwenden,
- unterstützt die Authentisierung auf der API.

4.1.2.2 Testaufbau

Der Coordinator und der Trustee befinden sich beide auf dem Localhost, so kann das Serialisieren und Deserialisieren getestet werden. Ebenfalls können die MK-TFHE Params verglichen werden.

Die Punkte 3 und 4 gemäss Abb. 4.4 sollen als POST-Request durchgeführt werden. Es soll dabei eine Zahl mitgeschickt werden. Das Read soll den HTTP Status Code 200 zurückmelden. Anschliessend muss die Zahl auf dem gegenüberliegenden Teilsystem verifiziert werden.

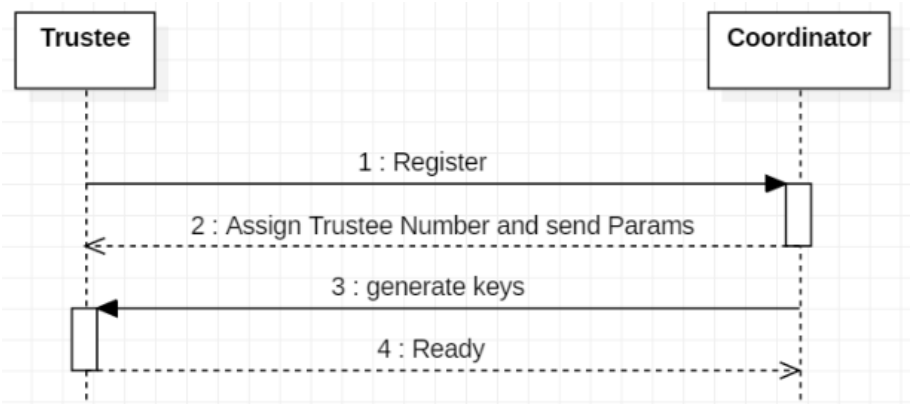


Abbildung 4.4: Testaufbau

Neben den Anforderungen sollen sich beim Evaluieren des Frameworks folgende Punkte notiert werden:

- Zeitdauer bis Technologiestack aufgesetzt wird.
- Aufwand, um den Aufbau umzusetzen.

4.1.2.3 C++ Web Framework

Das C++ Web Framework (CWF) ist ein modernes C++ Framework, welches unter der MIT License auf Github zur Verfügung gestellt wird. Die Projektwebseite von CWF kann unter https://www.cppwebframework.com/ erreicht werden.

Das CWF ist ein C++ Web Framework das mithilfe von Qt realisiert wurde. Der CWF Technologiestack wurde der Projektseite entnommen.

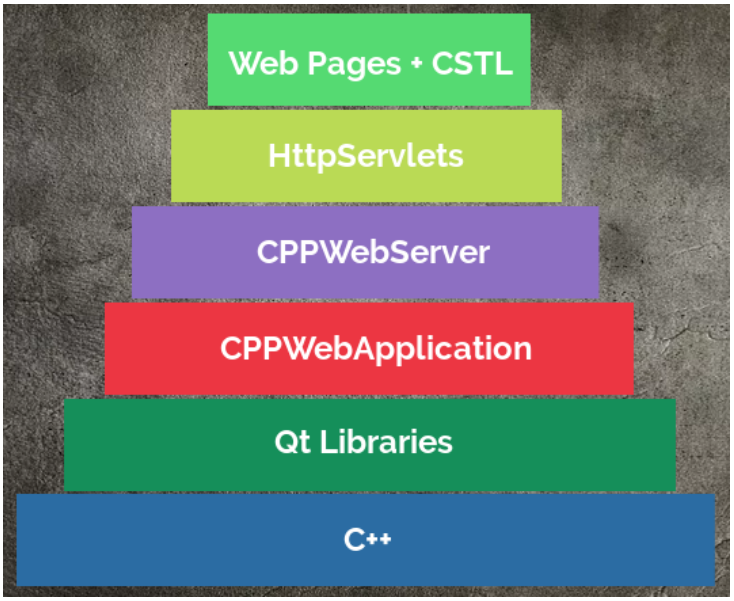


Abbildung 4.5: Technologie Stack C++ Web Framework

Wie in der Abbildung 4.5 ersichtlich ist, basiert das Framework hauptsächlich auf C++ und der Qt Library.

Installation

Die Installationsanleitung kann dem README.md des CPPWebFrameworks Github Repository entnommen werden, welches unter https://github.com/HerikLyma/CPPWebFramework erreichbar ist.

Bewertung

Das CWF macht einen schlanken Eindruck. Der Code für das Erstellen einer REST API ist sehr einfach und leserlich gehalten. Das Verarbeiten des JSON ist etwas aufwändiger. Es existieren eine Vielzahl von Lösungen die JSON Umwandlungen durchführen. In der Implementierung wurde die Umwandlung mit der in der Qt vorhandenen Funktionalität durchgeführt.
Einer der grössten Nachteile der Frameworks ist, dass die Community sehr klein ist und dadurch online wenige Informationen verfügbar sind.

4.1.2.4 CPP Rest SDK

Das CPP Rest SDK wurde von Mircrosoft erstellt. Detailiertere Informationen sind unter https://github.com/microsoft/cpprestsdk ersichtlich.

Das CPP Rest SDK ist in C++ geschrieben. Es baut auf Boost und Openssl auf.

4.1.2.5 Spring mit JNI

Auf Grund der vorhandenen Vorkenntnisse von Spring haben wir uns entschieden, ebenfalls Spring mit JNI zu testen.

Technologie

- C++ Shared Library
- JNI Java Nativ Interface
- Spring Java Web Framework

4.1.2.6 POCO

POCO C++ Libraries ist eine Ansammlung von Libraries, welche sich eignen, Backends für unterschiedliche Anforderungen zu entwickeln. Die Anzahl der Libraries ist gross, was eine sehr gute Grundlage schafft. Auf der Projektwebseite https://pocoproject.org unter Features ist dies ersichtlich. POCO gibt es in den POCO C++ Libraries und in der POCO PRO C++ Framework Version, welche kostenpflichtig ist. Die POCO C++ Libraries wurden unter der Boost Software License veröffentlicht.
Es gibt wenige Beispiele, wie man die POCO C++ Libraries nutzen kann. Ein Beispiel das gefunden worden ist unter https://github.com/edson-a-soares/poco_restful_webservice, wird von einem Contributor geführt. Es beinhaltet ein RESTful API, welches unter der Apache-2.0 License steht. Das README.md ist veraltet. Die Anleitung funktioniert nicht mehr und die Möglichkeit, in Docker zu deployen, nicht erwähnt wird, obschon es eine *docker-compose.yml* enthält. Der Docker Container kann gestartet werden, jedoch entstehen beim Starten mit *docker-compose up* viele Fehler.

Technologie

POCO baut auf diversen C++ Libraries auf. Die Libraries arbeiten gut mit der C++ Standard Library zusammen. Eine Übersicht der Unterteilung kann folgender Abbildung entnommen werden:

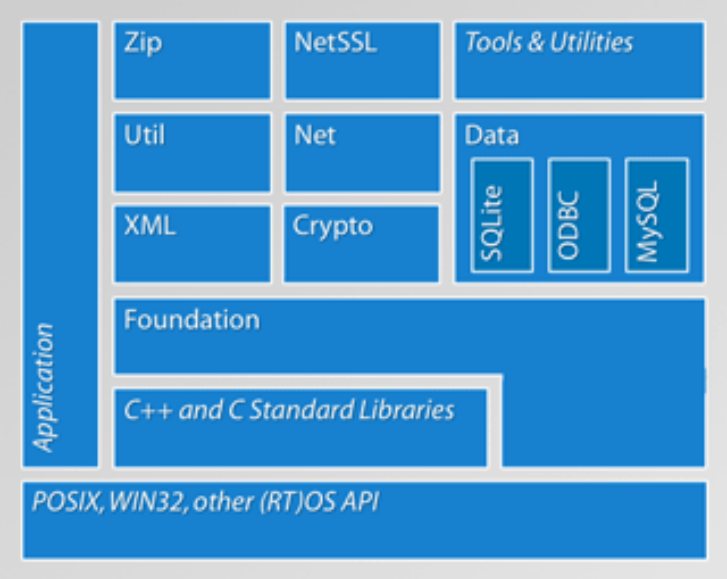


Abbildung 4.6: Unterteilung POCO Libraries

Bewertung

Die Anforderungen im Kapitel D.1 sind nicht beachtet worden. Dies hat den Grund, dass der Aufwand, sich in derart viele Libraries einzuarbeiten und damit einen Webserver zu erstellen, zu gross ist. Das oben erwähnte Beispiel beinhaltet mehr als 2000 Linien Code, was für einen kleinen RESTful Service aufwendig ist. Das Zeitfenster für die Bachelorarbeit ist zu klein, um eine Lösung mit POCO umzusetzen.

4.1.2.7 Gesamtbeurteilung

In der Tabelle werden die verschiedenen Frameworks verglichen:

Kriterium	CWF	CPP Rest SDK	Spring + JNI	POCO
Die MK TFHE Params in JSON parsen können.	2	5	3	1
Kann mit Listen von Integer (Keys) umgehen.	3	5	3	1
Kann HTTP POST Requests durchführen.	5	5	5	1
Das Framework unterstützt GNU/Linux.	5	5	5	1
Das Framework ist kostenfrei.	5	5	5	1
Das Framework ist aktuell, es hat zum Beispiel reguläre Git Commits.	4	5	5	1
Kann eine verschlüsselte Verbindung mit mindestens TLS 1.2 Version unterstützen.	5	5	5	1
Kann einfach in einen Docker Container eingebettet werden.	4	3	5	1
Es kann ein kleines Docker Image wie Alpine Linux verwendet werden.	5	3	5	1
Authentisierung auf die API ist unterstützt.	5	3	5	1

Tabelle 4.2: Vergleich der Technologien: Wobei 1 als niedrigster Punktzahl und 5 die Maximalpunktzahl ist

Beurteilung
Die beiden Frameworks CPP Rest SDK und Spring haben bei unseren Testapplikationen am besten abgeschnitten. Das Framework CPP Rest SDK haben wir für die Komponenten Trustee und Bulletin Board aus folgenden Gründen ausgewählt:

- die JSON Library ist integriert.
- die Rest Server-Komponente ist vorhanden.
- die Rest Client-Komponente ist vorhanden.

Spring + JNI ist für den Coordinator ausgewählt worden, weil keine Abhängigkeit zur MK-TFHE-Library besteht. In Spring gibt es viele praktische Libraries, welche erlauben, die benötigten Funktionalitäten einzubauen. Für den Coordinator haben wir uns aus folgenden Gründen für das Framework Spring entschieden:

- die JSON Library ist integriert.
- das Deployment auch mit Docker ist einfach.
- es ist betriebssystemunabhängig

4.2 Softwarearchitektur

4.2.1 Logische Architektur

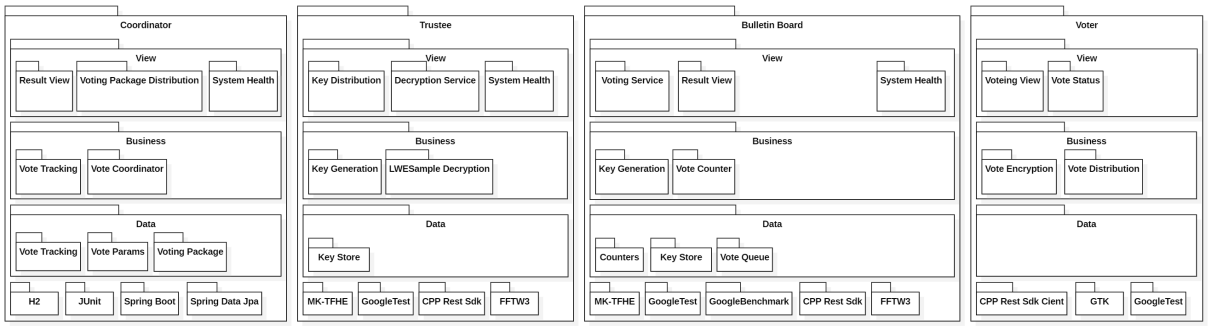


Abbildung 4.7: Architekturdiagramm

4.2.1.1 MK-TFHE-Library

Die MK-TFHE-Library ist unter https://github.com/ilachill/MK-TFHE ersichtlich. Es wird vom Branch extProduct_M1 der Library ausgegangen, da die anderen Branches nicht kompilierbar sind. Für die Umsetzung des E-Voting-Systems war es zusätzlich notwendig einige logische Operationen der alten MK-TFHE-Library in die neue Implementation zu überführen. Des weiteren enthält die MK-TFHE-Library alle Funktionen, die von allen Komponenten benötigt werden.
Die MK-TFHE-Library beinhaltet zusätzlich Interfaces, die eine Abstraktion der zu übertragenden Daten enthält. Die Daten werden dann in JSON-Objekte umgewandelt und per REST-Schnittstelle übertragen. In Tests können die Daten auch direkt übertragen werden.

4.2.1.2 Coordinator

Der Coordinator ist die zentrale Verwaltung dieses E-Voting-Systems. Als solches ist er primär verantwortlich für die Koordination aller Komponenten. Zusätzlich definiert er die Parameter, unter welchen die Abstimmung stattfindet.

4.2.1.3 Trustee

Der Trustee ist für die Schlüsselgenerierung und die Entschlüsselung der Stimmen zuständig. Die Schlüssel werden vor der Wahl mit der MK-TFHE-Library erzeugt. CPPRestSDK wird verwendet, um einen Server zu starten, der die Anfragen annimmt. Nach dem Start werden Keys generiert. Die Public-Keys werden nach der Generierung dem Coordinator gesendet. Während der Wahl muss der Trustee nicht erreichbar sein. Nach Abschluss der Wahl werden die Endresultate von den Bulletin Boards zu den Trustees gesendet. Die Resultate müssen mithilfe der MK-TFHE-Library von den Trustee entschlüsselt werden.

4.2.1.4 Bulletin Board

Die Bulletin Boards sind für das Zusammenzählen der Stimmen zuständig. Die Voter senden die Stimmen an alle Bulletin Boards. Über das CPPRestSDK werden die Stimmen entgegengenommen und in eine Warteliste hinzugefügt. Diese wird der Reihe nach abgearbeitet. Zuerst wird über den Coordinator geprüft, ob alle Bulletin Boards die Stimmen erhalten haben. Anschliessend werden die Stimmen über die MK-TFHE Library zu den Countern addiert. Am Ende der Wahl werden die verschlüsselten Counter mit CPPRestSDK den Trustees zum Entschlüsseln gesendet. Nach Abschluss der Wahl werden alle Stimmen, die sich nicht auf allen Bulletin Boards befinden, verworfen.

4.2.1.5 Voter

Der Voter wird als GTK-3 GUI-Applikation umgesetzt. Die Client-Applikation fordert die Daten vom Coordinator an. Der Client bekommt darauf ein Voting-Package in JSON. Nachdem der Voter seine Wahlzettel ausgefüllt hat, werden alle Wahlzettel im JSON Format an alle Bulletin Boards gesendet.

4.3 Deployment

Die Software ist in vier Komponenten aufgeteilt. Bei Anpassungen muss nur die Komponente gewechselt werden, die angepasst wurde.
Für die Serverkomponenten wird eine Containerlösung auf Basis von Docker verwendet. Das erleichtert das Verwalten und bringt eine gewisse Plattformunabhängigkeit. Dies dient auch als Grundlage für eine Skalierung des Systems. Für eine Auswertung im Fehlerfall werden die Logs mit dem Opensource-Tool Graylog an einem zentralen Ort zusammengetragen. Daduch muss nicht auf jeden Container einzeln zugegriffen werden, um die Logs zu sehen.

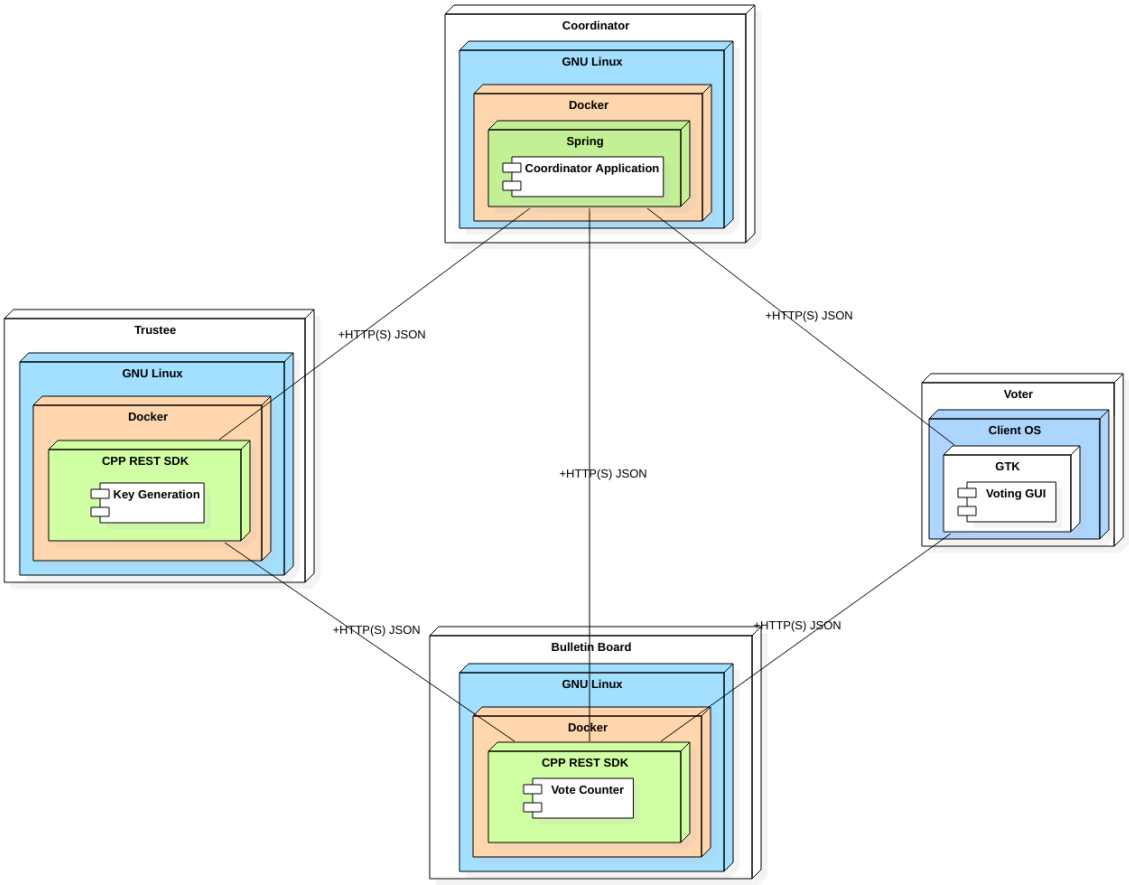


Abbildung 4.8: Deploymentdiagram

4.4 Entwicklung

4.4.1 Versionsverwaltung

Für die Versionsverwaltung wurde GitLab verwendet. Dabei wurde für jede Komponente (MK-TFHE, Trustee, Bulletin Board, Voter und Übergreifende Tests) ein eigenes Repository erstellt. Zusätzlich

ist das E-Voting-System Projekt als übergeordnetes Projekt mit Git-Submodules mit den zusammengehörigen Versionen zusammengefasst. Ebenfalls werden über das E-Voting-System Projekt externe Abhängigkeiten wie MK-TFHE, GelfCpp und CppRestSdk eingebunden.

4.4.2 Entwicklungsumgebung

Das Projekt besteht aus Sicht der Entwicklungsumgebung aus zwei Teilen. Zum einen gibt es, einen grösseren C++ Teil und zum anderen einen kleineren Java Teil.

4.4.2.1 C++

Im C++ Teil wurde CMake mit als Buildumgebung verwendet. Dies ermöglicht es, eine Strukturierung mit Libraries und Executeables zu erstellen, die einfach zu handhaben ist. Zusätzlich erlaubt die CLion IDE so, nur Teile des Projektes zu erstellen und zu testen.
CMake erlaubt es zusätzlich, unterschiedliche Konfigurationen (Dev, Live) einfach umzustellen.
Unter Windows ist es ebenfalls möglich, zu entwickeln. Jedoch wird dafür das WSL (Windows Subsystem for Linux) benötigt. Es ist dabei besser, die Debian Version von WSL zu verwenden, da es im Gegensatz zur Standard Ubuntu-Version über neuere Versionen der Libraries verfügt. Mit Hilfe von VcXsrv ist es auch möglich, das Voter-UI aus WSL in Windows anzuzeigen.

4.4.2.2 Java

Der Coordinator wurde als ein Maven-Projekt realisiert. Dies ermöglicht die einfache Versionsverwaltung und ein kontrolliertes Buildverfahren. Dieser Teil ist Plattform unabhängig, solange eine aktuelle JVM (Java Virtual Machine) vorhanden ist.

4.4.3 Continuous Integration

Zum Testen der Applikationen wird Jenkins verwendet. Ursprünglich wurde Jenkins für Java entwickelt, jedoch gibt es eine Vielzahl von Plugins, die verschiedene Programmiersprachen zulässt. Jenkins ermöglicht es, zusammenhängende Projekt zu testen, sodass nach dem Build einer Library auch die davon abhängigen Projekte getestet werden konnten.
Zusätzlich können mit Jenkins auch statische Analysen wie CppCheck und CheckStyle durchgeführt werden. Zudem sind Memoryanalysen mit Valgrind möglich.

4.4.4 Continuous Delivery

Alle Dienste auf unserem Testserver laufen in einem Dockercontainer. Dies ermöglicht es, uns die Teilkomponenten Coordinator, Trustee und Bulletin Board direkt als Buildstep im Jenkins in einen Container zu packen und zu deployen. Jedoch ist die Kommunikation zu den Komponenten durch die Firewall-Limitationen der HSR beeinträchtigt.

5. Resultate

Die definierten funktionalen Anforderungen werden vom E-Voting-System erfüllt. Auf dem Coordinator können Wahlen und Abstimmungen erstellt werden. Die nötigen Schlüssel werden auf der seperaten Komponente Trustee erstellt. Mit dem Voting-Client kann an einer Abstimmung teilgenommen werden. Das Bulletin Board zählt alle Stimmen verschlüsselt zusammen. Nach der Stimmabgabe kann der Status des Wahlzettels abgefragt werden und so die Addierung der eigene Stimme verifiziert werden.

5.1 Aufteilung Komponenten

Das E-Votingsystem besteht aus vier eigenständigen Komponenten. Sie sind über Webschnittstellen miteinander verbunden. Es können so, wie gefordert, mehrere Bulletin Boards die Stimmen zusammenzählen um Manipulation zu erkennen.

5.2 Benutzeroberfläche

Coordinator
Der Coordinator hat eine graphische Oberfläche zum Veröffentlichen der Wahlergebnisse. Die zusätzliche Administrationsseite ermöglicht das Erstellen einer Abstimmung und die zentrale Konfiguration der Wahlparameter.

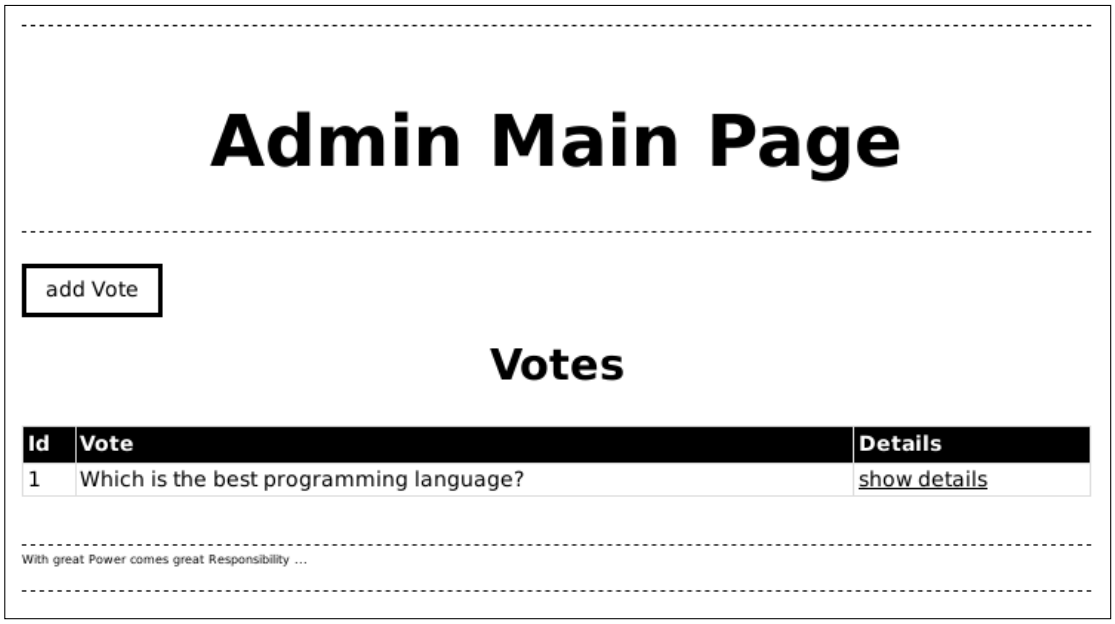


Abbildung 5.1: Voting-Client Hauptseite

Voting-Client
Der Voting-Client bietet eine übersichtliche graphische Oberfläche, diese ermöglicht eine einfache Stimmabgabe. Der Status der abgegebenen Stimme kann im Voting-Client überprüft werden. So kann der Wähler selbst erkennen, ob seine Stimme schon verarbeitet wurde.

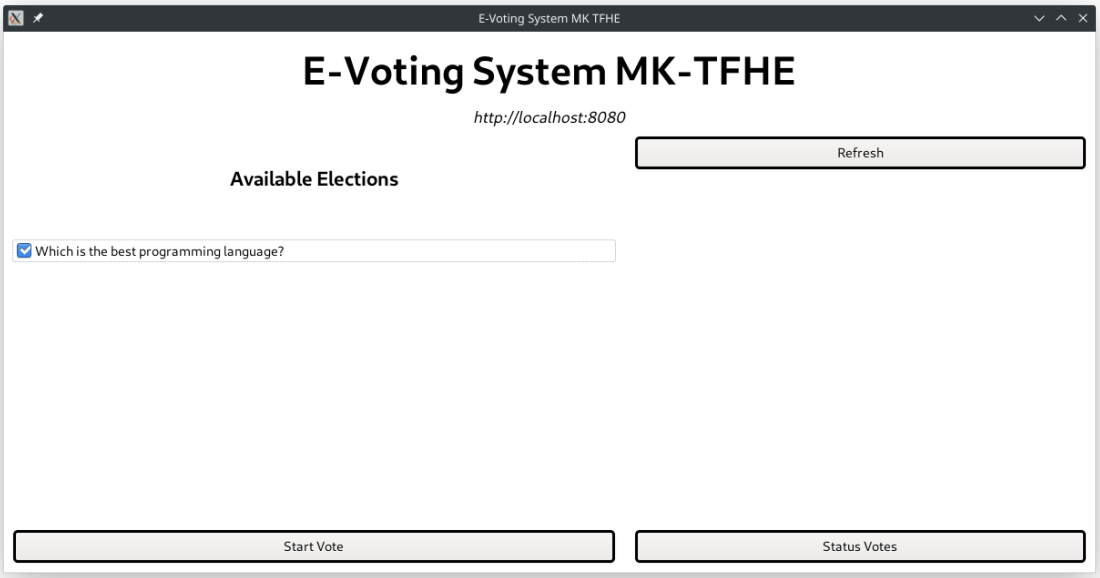


Abbildung 5.2: Voting-Client Hauptseite

5.3 Parallelisierung

Der Rechenaufwand wird auf dem Bulletin Board pararallel an die darunterliegende MK-TFHE-Library weitergegeben. Die Gateoperationen der Library haben alle etwa die gleiche Laufzeit.

AND	OR	XOR	NAND	NOR
1.135s	1.124s	1.130s	1.098s	1.126s

Tabelle 5.1: Ausführungszeiten Gateoperationen

Die deutliche Leistungssteigerung wird beim Zusammenzählen der Stimmen durch das parallele Ausführen der Gateoperationen erreicht. Auf die gleiche Weise konnte auch die Entschlüsselung beschleunigt werden. In der Standardkonfiguration mit drei Trustees und vier Wahloptionen ergeben sich auf den Bulletin Boards folgende Zeiten, bis das Endresultat bekannt gegeben werden kann:

Optimierung	8 Wähler (328 Gates)	16 Wähler (776 Gates)	32 Wähler (1800 Gates)	64 Wähler (4104 Gates)	128 Wähler (9224 Gates)
Parallel	1min	3min	7min	16min	35min
Sequenziell	6min	15min	36min	1h 20min	3h

Tabelle 5.2: Ausführungszeiten Abstimmung

5.4 Performance

Die Geschwindigkeit der Stimmenzählung hängt von verschiedenen Einflussfaktoren ab. Zum einen ist die Anzahl der Trustees zu beachten, da Schlüssel von jedem Trustee verwendet werden, steigt der Rechenaufwand mit mehreren Trustees (Abbildung 5.3) an. Zum anderen sind die möglichen Wahloptionen von Bedeutung, da es pro Operation einen unabhängigen Counter benötigt. Deswegen steigt der Rechenaufwand mit der Anzahl Wahloptionen (Abbildung 5.4). Des weiteren ist die Anzahl der Wähler (Abbildung 5.5) nicht zu vernachlässigen. Die Counter der verschiedenen Wahloptionen werden mit zunehmender Anzahl Wähler grösser, was zu Beginn den Rechenaufwand spürbar steigert.
Die Performance wurde auf einem Notebook mit Intel Core i7-6820HQ Prozessor und 16GiB Arbeitspeicher getestet. Der C++ Source-Code wurde als Release und ohne Optimuierungen kompiliert. Der GCC 9.2.0 Compiler wurde zum Kompilieren verwendet.

5.4.1 Anzahl Trustees

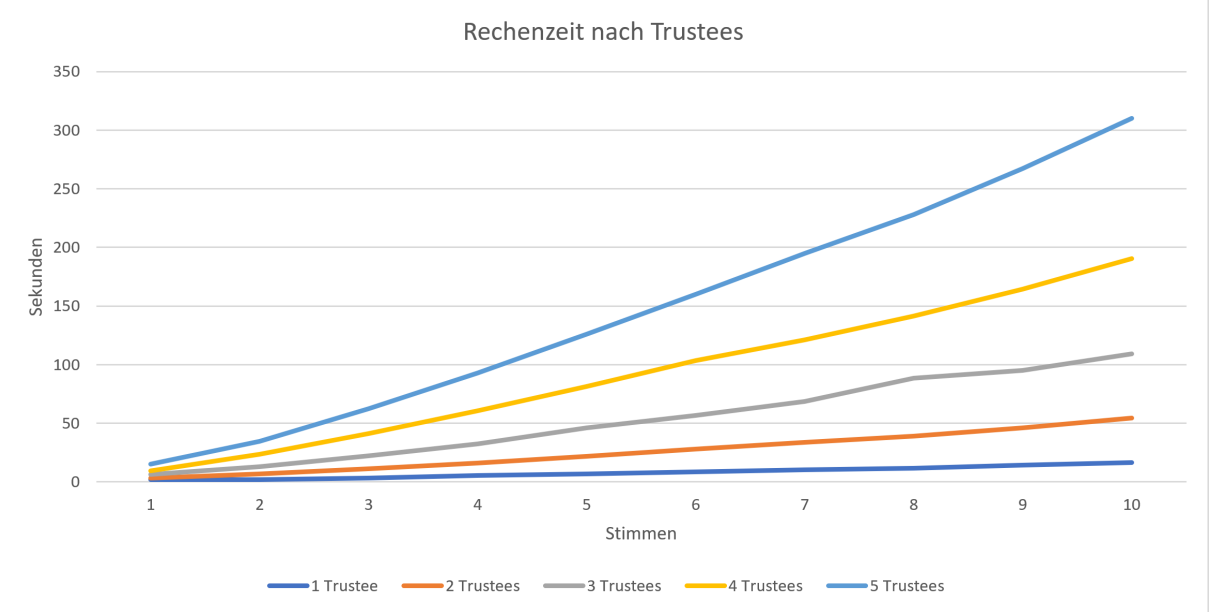


Abbildung 5.3: Ausführungszeiten unterschiedliche Anzahl Trustees bei 4 Wahloptionen

5.4.2 Anzahl Wahloptionen

Die Anzahl der Wahloptionen wirkt sich nicht linear auf die Ausführungszeit aus. Da der verwendete Algorithmus auf Zweierpotenzen optimiert ist, kann bei vier Optionen eine Steigerung der Geschwindigkeit beobachtet werden.

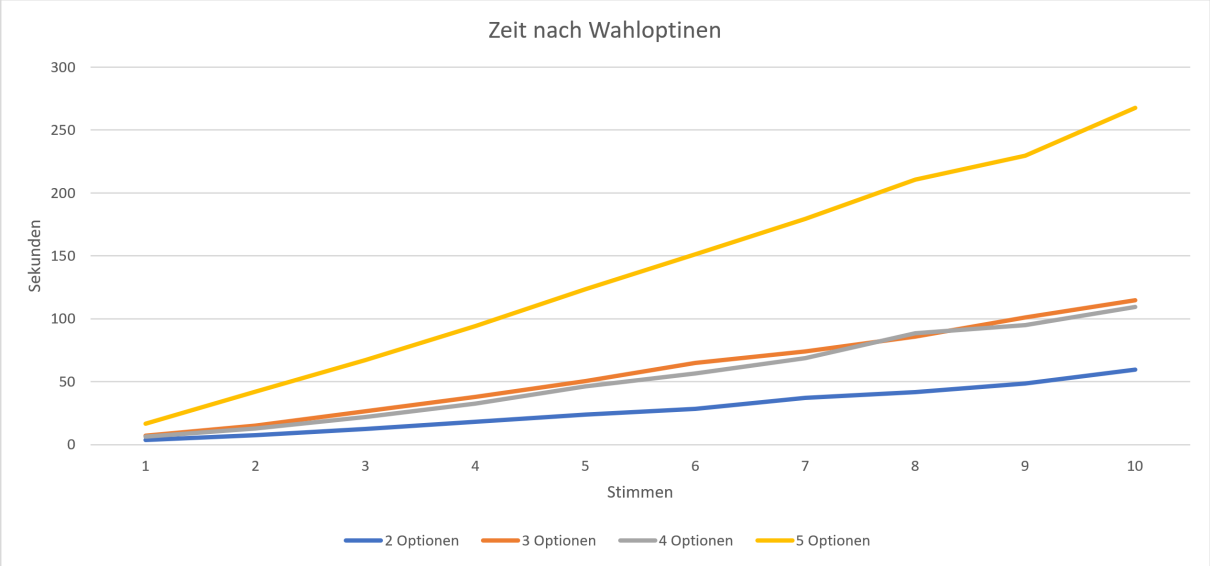


Abbildung 5.4: Ausführungszeiten unterschiedliche Anzahl Wahloptionen bei 3 Trustees

5.4.3 Anzahl Wähler

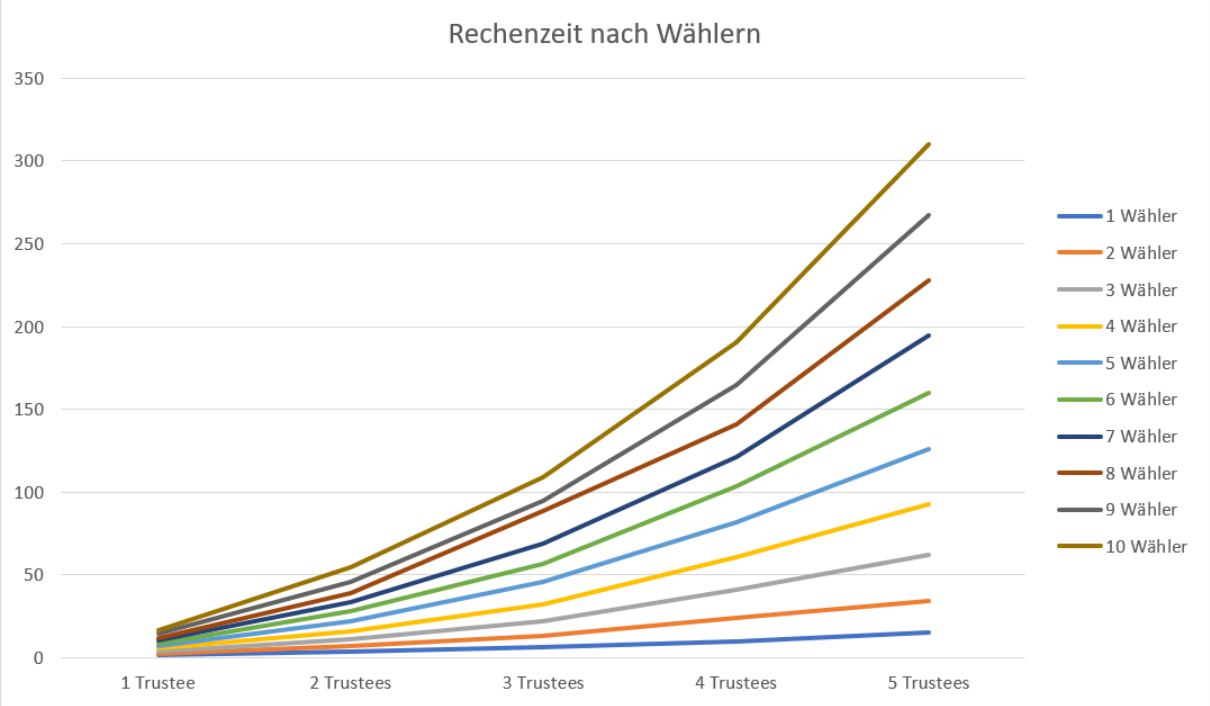


Abbildung 5.5: Ausführungszeiten unterschiedliche Anzahl Trustees und Wähler

5.4.4 Performance Schätzung

Für die Schätzung der Rechenzeit kann die folgende Formel angewendet werden.

$$Zeit(o,s) = 1.8sec * (s * (\lceil log_2(\lceil log_2(o) \rceil + 1) \rceil + 1 + 2^{\lceil log_2(o) \rceil} - o) + \sum_{n=1}^{s} \lceil log_2(s+1) \rceil))$$

Bei der Formel ist o die Anzahl Wahloptionen und s die Anzahl Stimmen. Zudem geht die Formel von 3 Trustees aus. Es wird ebenfalls angenommen, dass genügend Threads zur Verfügung stehen. Die Formel setzt sich aus den folgenden Komponenten zusammen.

- $1.8sec$: Zeit pro Gateoperation und Synchronisation zwischen den Gates und zusätzliche Operationen. Kann nach System variieren.
- $\lceil log_2(\lceil log_2(o) \rceil + 1) \rceil + 1$: Zeit im Expander die für das decodieren der Wahl benötigt wird.
- $2^{\lceil log_2(o) \rceil} - o$: Zeit die benötigt wird um im Fall das die Anzahl Optionen keine Zweierpotenz ist.
- $\lceil log_2(s+1) \rceil$: Zeit pro Stimme unter der Voraussetzung, dass bereits s Stimmen abgegeben wurden..

6. Ausblick

6.1 Verifizierung der Verbindungen

In der aktuellen Implementation haben die Komponenten untereinander keine Möglichkeit, die Identität einer anderen Komponente zu verifizieren. Deshalb ist es momentan ohne grossen Aufwand möglich eine Wahl zu sabotieren, indem man die Identität einer Komponente annimmt und fehlerhafte Nachrichten versendet. Dies kann durch den Einsatz von Zertifikaten verhindert werden.

6.2 Authentisierung / Autorisierung Voter

Wie bereits im Kapitel 2.3 Umsysteme erwähnt, benötigt das System zukünftig eine Möglichkeit zur Authentisierung und Autorisierung der Benutzer. Dies kann im Fall der Schweiz möglicherweise über die SwissID gelöst werden. Es ist aber sinnvoll, weitere Optionen zu prüfen, die beispielsweise für eine Generalversammlung einer Aktiengesellschaft eingesetzt werden könnten.

6.3 Nachvollziebarkeit mit Blockchain

Momentan werden für die Persistenz der Daten Datenbanken verwendet. Zwei zukünftige Anforderungen an das System sind Nachvollziehbarkeit und Unveränderbarkeit der Ergebnisse. Dies kann eine relationale Datenbank nur schwer erreichen. Eine öffentliche Blockchain hingegen bietet beide dieser Eigenschaften. Beispielsweise könnte dies über Ethereum Smart-Contracts gelöst werden.

6.4 Einheitliche UX auf allen Komponenten

Die Benutzeroberflächen sind aufgrund der Architektur auf verschiedene Komponenten verteilt. Die Benutzerführung ist auf den verschiedenen Oberflächen nicht einheitlich und deshalb für Personen ohne Vorkenntnisse nicht intuitiv. Vor einem produktiven Einsatz des Systems müssten verschiedene Komponenten benutzerfreundlicher gestaltet werden.

6.5 Recovery der Komponenten

Momentan dient die Persistenz primär der Nachvollziehbarkeit. Jedoch kann lediglich der Coordinator im Falle eines Absturzes seine Arbeit nach einem Neustart fortsetzen. Für den Einsatz des Systems ist es essenziell, dass Komponenten nach einem Absturz ihre Arbeit fortsetzen können. Der Trustee schreibt bereits jetzt alle generierten Informationen in eine SqLite Datenbank, jedoch fehlt eine Recovery-Strategie. Das Bulletin Board braucht zusätzlich noch eine Erweiterung der Persistenz. Es speichert lediglich die Stimmen, jedoch geht zur Zeit der KeySwitchKey bei einem Absturz verloren. Alternativ kann auch die Generierung eines neuen KeySwitchKeys geprüft werden.

6.6 Loadbalancing Bulletin Board

Die Verarbeitung der Stimmen ist durch verschiedene Optimierungen bereits wesentlich schneller. Jedoch ist die Verarbeitung für einen realen Einsatz mit ca. 15 Minuten für 64 Stimmen immer noch eher langsam. Eine weitere Optimierung wäre die Aufteilung des Bulletin Board auf mehrere Rechner, um so ein Loadbalancing der Stimmen zu ermöglichen. Dies wurde bereits als Variante (6.3 Variante Loadbalancer) in der Softwarearchitektur in Erwägung gezogen, es wurde aber nicht implementiert.

6.7 Private Keys im Speicher

Die MK-TFHE-Library ist zuständig für die Erstellung der Private Keys. In der Studienarbeit[5] wurde darauf geachtet, dass die Private Keys nur referenziert werden (keine Kopie), nach Gebrauch im Speicher mit Nullen überschrieben werden und nie geswappt werden. In der Bachelorarbeit wurde die bestehende MK-TFHE-Library benutzt, die oben genannten Anpassungen wurden nicht durchgeführt, da die gesamte Library nicht angepasst wurde. Die Sicherheit des E-Voting-Systems könnte verbessert werden, indem die genutzten Funktionen aus der MK-TFHE-Library herausgelöst werden. Wie in der Studienarbeit umgesetzt wurde, sollte der Source-Code in einen objektorientierten und aktuellen C++ Standard umgeschrieben werden unter Beachtung der drei erwähnten Punkte.

7. Abkürzungsverzeichnis

Glade Glade ist eine User Interface Designer für GTK, welcher XMLs generiert, welche dann im Code geladen und dann referenziert werden können.. 90

GSW Encryption by Craig **Gentry** and Amit **Sahai** and Brent **Waters**(GSW). 7

JVM Java Virtual Machine. 40

LWE Learning with errors. 6–8, 11–14, 86

Meson Meson ist ein Build System, das unter anderem für das builden von C++ Applikationen verwendet werden kann.. 90

MK-TFHE Multi-Key fast Fully Homomorphic Encryption over the Torus. 6, 7, 37, 39, 49, 86

NGO Nicht Regierunsorganisation. 18

Qt-Creator C++ IDE zum bequemen entwickeln von Qt-Applikationen. 77

SIMD Single Instruction, Multiple Data, sind Instruktionen, die vektoriell verarbeiten können.. 90

TFHE Fast Fully Homomorphic Encryption over the Torus. 6

TLWE Learning with errors over the Torus. 6

VcXsrv VcXsrv ist ein X-Server für Windows. 40

WSL Windows Subsystem for Linux. 40, 76, 80

8. Quellenverzeichnis

[1] Post CH AG. *Die E-Voting-Lösung der Post*. https://www.post.ch/-/media/post/evoting/dokumente/factsheet-e-voting.pdf. 2019.

[2] Hao Chen, Ilaria Chillotti und Yongsoo Song. *Multi-Key Homomophic Encryption from TFHE*. Cryptology ePrint Archive, Report 2019/116. https://eprint.iacr.org/2019/116. 2019.

[3] Ilaria Chillotti. *Towards efficient and secure Fully Homomorphic Encryption and cloud computing*. L'Université Paris-Sacla on Github. https://ilachill.github.io/papers/these_Ilaria_Chillotti_wo_acknowl.pdf. 2018.

[4] Dan Boneh Emily Stark Michael Hamburg. *Symmetric Cryptography in Javascript*. A whitepaper at the Annual Computer Security Applications Conference. 2009. URL: https://crypto.stanford.edu/sjcl/acsac.pdf.

[5] Romeo Spinas Lukas Lätsch Rolf Furrer. *Post-Quantum E-Voting Studienarbeit*. HSR E-Print. http://eprints.hsr.ch/id/eprint/816. 2019.

Illustrationsverzeichnis

Tabellenverzeichnis

A. Projektplan

A.1 Projektübersicht

Projektbeschreibung

Das Projekt beinhaltet die Umsetzung einer Post-Quantum E-Voting Plattform. Für die Implementation der Software verwenden wir eine homomorphe Verschlüsselung, die in der *Arbeit Towards efficient and secure Fully Homomorphic Encryption and cloud computing* [3] beschrieben wird. Zudem wird die Folgearbeit *Multi-Key Homomophic Encryption from TFHE*[2] mit einbezogen. Die Library kann unter Link `https://github.com/ilachill/MK-TFHE` erreicht werden.

Basierend auf der erwähnten Library soll ein E-Voting System entwickelt werden, welches folgende Teilsysteme umfasst:

- Coordinator
- Voter
- Bulletin Board
- Trustee

A.1.1 Zweck und Ziel

Zielsetzung

- Referenzimplementation der homomorphen Verschlüsselung für ein E-Voting System.
- Anwendung der im Studium gelernten Fähigkeiten.
- Software in Teilsysteme aufteilen.
- Kommunikation der Komponenten spezifizieren.

Persönliche Ziele

Für die Durchführung dieses Projektes haben wir uns folgende persönlichen Ziele vorgenommen:

- Uns mehr Erfahrung mit C++ aneignen.
- Uns in eine komplexe Materie einarbeiten.
- Ein besseres Verständnis für Software mit besonderen Sicherheitsanforderungen aneignen.

A.2 Projektorganisation

A.2.1 Organisationsstruktur

Das Projektteam besteht aus drei Entwicklern:

- Lukas Lätsch
- Rolf Furrer
- Romeo Spinas

Verantwortlichkeiten werden pro Task in der Iterationsplanung zugeteilt.

A.2.1.1 Externe Schnittstellen

Betreuer:

- Prof. Dr. Andreas Steffen

Es werden regelmässige Sitzungen mit dem Betreuer durchgeführt, um den Stand der Arbeit zu besprechen.

A.3 Management-Abläufe

A.3.1 Kostenvoranschlag

Zeitbudget

Die Bachelorarbeit hat am Montag, 16.09.2019 begonnen. Das Projekt soll gemäss der offiziellen Terminübersicht am 10.01.2020 fertiggestellt und die Dokumente sollen hochgeladen werden.

Es stehen 14 Wochen zur Verfügung, um die Bachelorarbeit durchzuführen. Dabei arbeitet jedes Mitglied 360 Stunden an diesem Projekt. Dies ergibt ein Zeitbudget für die 3 Mitglieder von 1080 Stunden.

A.3.2 Zeitliche Planung

Zeiterfassung

Die Zeitplanung und die Verwaltung der Arbeitspakete wird mithilfe der Software Redmine bewältigt. Die Planung wird laufend aktualisiert, um schnell auf Änderungen reagieren zu können. Der aktuelle Zeitplan im Redmine ist unter folgender URL abrufbar: https://redmine.lag-13.ch/projects/ba/issues/gantt

Genauigkeit

Die effektive Zeit wird im Redmine mittels einem Tracker erfasst, um genug präzise den Ressourcenverbrauch des Projektes nachzuführen.

A.3.3 Phasen / Iterationen

Phasendauer

Die 14 Wochen des Projektes werden in der Elaboration Phase in einwöchige Iterationen aufgeteilt. Die Construction Phase wird in zweiwöchige Iterationen unterteilt.

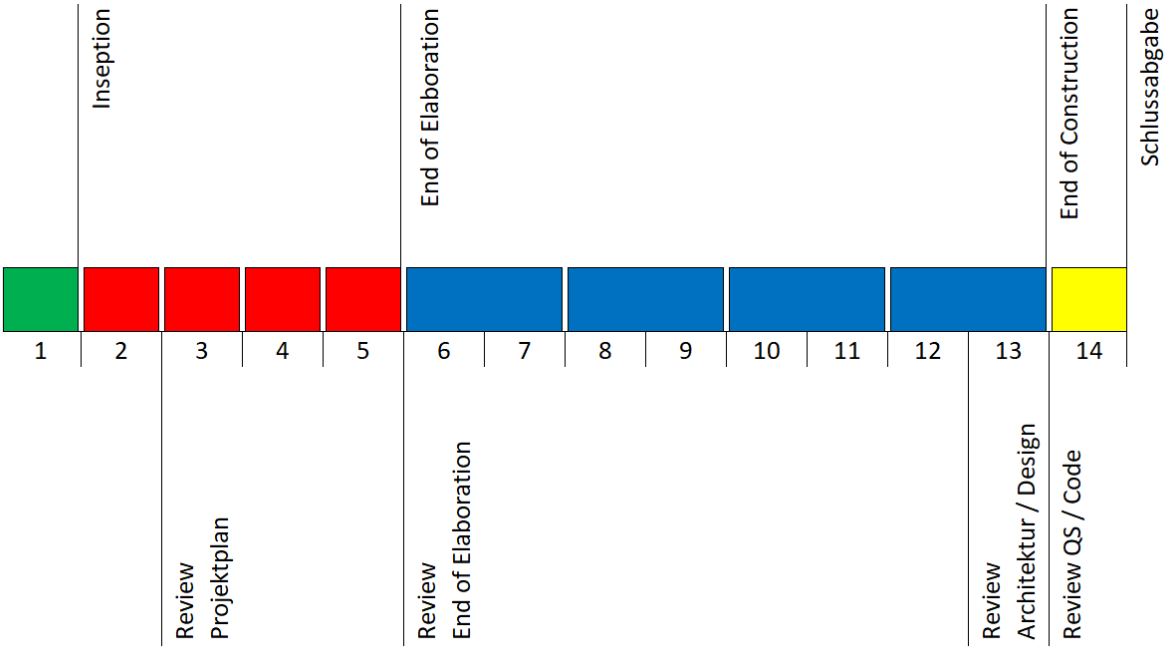


Abbildung A.1: Übersicht Iterationen und Reviews

A.3.4 Meilensteine

Übersicht

Die Meilensteine, welche in der folgenden Tabelle ersichtlich sind, werden angestrebt. Die Erzeugnisse werden mit dem verantwortlichen Betreuer besprochen.

Datum	Meilenstein	Beschreibung
29.09.2019	Projektplan abgeben	Projektplan erstellt/abgegeben
29.09.2019	Anforderungsspezifikation	Use Cases erfasst, nicht funktionale Anforderungen erfasst, Beschreibung Schnittstellen zu anderen Systemen, Domainmodell erstellt, Architektur festgelegt, Abuse-Cases erarbeitet
30.09.2019	Review Projektplan und Anforderungsspezifikation	
14.10.2019	Schnittstellen definiert	Definition der Rest interfaces Definition von DTOs
20.10.2019	End of Elaboration	Beispiel aufgeteilt in Teilapplikationen
21.10.2019	Review End of Elaboration	
04.11.2019	Coordinator Applikation	Coordinator Applikation netzwerkfähig
18.11.2019	Voter Applikation	Voter Applikation netzwerkfähig
02.12.2019	Trustee und Bulletin Board Applikation	Trustee Applikation netzwerkfähig Bulletin Board Applikation
09.12.2019	Review Architektur / Design	
15.12.2019	End of Construction	Testing abgeschlossen
16.12.2019	Review QS / Code	
06.01.2020	Abstract	Abgabe Abstract an den Dozenten für ein Review freigeben
10.01.2020	Abgabe aller verlangten Dokumente	Hochladen aller verlangten Dokumente auf archiv-i.hsr.ch Abgabe der Arbeit an den Dozenten bis 17:00 (Dozent Abgabe des Abstract)

Tabelle A.1: Meilensteine

A.3.5 Besprechung

Meetings

Die Projektmitglieder treffen sich mindestens einmal pro Woche. Die Meetings finden jeweils am Montag statt. Eine Standortbestimmung findet jeweils vor dem Treffen mit dem Dozenten statt und danach wird das weitere Vorgehen zusammen bestimmt.

Reviews

Die Meetings mit dem betreuenden Dozenten finden wenn möglich am Montag ab 15 Uhr statt, um ungeklärte Fragen zu besprechen und die folgenden Schritte zu planen. Zudem werden die Dokumente, die bei den Meilensteinen abgegeben werden, an diesen Meetings besprochen.

A.4 Risikomanagement

A.4.1 Risikomatrix

Nr	Titel	Beschreibung	Schaden pro Person [h]	Eintrittswahrscheinlichkeit	Gewichteter Schaden	Vorbeugung	Verhalten beim Eintreten
R1	Plattformabhängigkeit	Die verwendeten Libraries können nicht Plattform unabhängig eingesetzt werden.	8	Mittel	Mittel	Einrichte von Linux und Linuxänlichen Buildumgebungen (WSL)	Windows als Zilsystem ausschliessen
R2	Ressourcen	Die Laptops verfügen nicht über genügend Ressourcen, um beispielsweise den Secretkey zu genereieren.	16	Klein	Gross		Schlüssel auf einem Server generieren
R3	Inkorrekte Wahlresultate	Fehler im Code oder manipulierter Input führt zu falschen Wahlresultaten.	16	Mittel	Gross	Möglichst gründliche UnitTests und Abuse Cases	
R4	Anonymität	Kommunikation über unsichere Kanäle kann zu Verlust der Anonymität von Wählern führen.	4	Klein	Gross	Keine unverschlüsselte Nachrichten versenden	

Tabelle A.2: Risikomatrix

A.5 Arbeitspakete

Die Arbeitspakete des Projektes werden mit Redmine verwaltet. Alle Arbeitspakete werden geschätzt und priorisiert und dann in eine Iteration eingeplant.

A.6 Infrastruktur

A.6.1 Verwendete Infrastruktur

In der nachfolgenden Liste befindet sich die für die Umsetzung des Projektes benötigte Hardware und Tools.

Name	Verwendung
Redmine	Planung
Gitlab	Versionsmanagement
Buildserver (Debian, Jenkins)	Deploymnt / Testing
Laptop (2 Linux, 1 Windows)	Developement
WSL	Windows Subsystem for Linux (WSL) ist eine Umgebung zum Ausführen von Linux-Executables im ELF-Format in Windows 10
Cmake	Zum Generieren der Makefiles
Clion	C++ IDE

Tabelle A.3: Verwendete Infrastruktur

A.6.2 Verwendete Frameworks

Name	Verwendung
MK-TFHE	Dient als Grundlage, wird bei Bedarf angepasst
FFTW3	Fast Fourier transformation
Cute	Unit Testing

Tabelle A.4: Verwendete Frameworks

A.7 Qualitätsmassnahmen

Übersicht

Die folgende Tabelle gibt eine Übersicht über die festgelegten Qualitätsmassnahmen:

Massnahme	Zeitraum	Ziel
Backup aller Online-Tools	Einmal täglich	Kein Datenverlust, Konsistenz
Durchbesprechen der Dokumentation an Meetings	Jedes Meeting	Dokumentation enthält Konsens, keine Einzelmeinungen
Verwendung einer Projektmanagement-Software	Gesamtes Projekt	Zentralisierte Verwaltung von Pendenzen, Tickets, Zeiterfassung
Meilenstein-Snapshots	Jeder Meilenstein	Nachvollziehen des Ablaufs im Nachhinein möglich
Führen von Git-Repositories	Gesamtes Projekt	Versionierung, Sicherung
Implementation von Unit-Tests	Gesamtes Projekt	Korrektheit der Features
Durchführen von Code-Reviews	Nach Feature-Fertigstellung	Korrektheit und guter Stil im Code
Verwenden von Style-Guidelines	Gesamtes Projekt	Konsistenz im Code-Styling

Tabelle A.5: Qualitätsmassnahmen Übersicht

A.7.1 Dokumentation

Ablageorte

Die Dokumentationsdokumente der Post-Quantum E-Voting Arbeit, werden im LaTeX Format, in einem separaten GitLab Repository verwaltet.

Team-Konsens

Die Qualität der Dokumente wird in erster Linie dadurch sichergestellt, dass sämtliche Änderungen nachverfolgt und dann im jeweils nächsten Teammeeting (im Regelfall am folgenden Montagnachmittag) im Plenum besprochen und nachvollzogen werden. Auf diese Weise repräsentiert die Dokumentation immer den Konsens des gesamten Teams und nicht die Meinung von Einzelpersonen. Zudem wird das ganze Team über den aktuellen Stand des Projektes damit informiert. LaTeX Dokumente ermöglichen zusätzlich Änderungen einfach in einem Git-Diff nachzuverfolgen.

Snapshot

Beim Erreichen jedes Meilensteins wird ein Snapshot der gesamten Dokumentation als PDF erstellt und auf dem GitLab Repository hinterlegt, damit später der gesamte Prozessablauf noch nachvollzogen werden kann, sollte dies nötig sein.

A.7.2 Projektmanagement

Redmine

Das Projektmanagement wird, wie im Modul SSoftware Engineering 1"vorgeschlagen, mit dem Tool Redmine unterstützt, welches auf einem privaten Server in einem Docker-Container läuft und somit für alle Mitarbeitenden jederzeit erreichbar ist. Das Tool ist unter der Adresse https://redmine.

lag-13.ch/projects/ba erreichbar.

Verwendung

Redmine wird primär zum Erstellen von Arbeitspaketen, zum Nachvollziehen der bereits erledigten Aufgaben, zum Aufführen bekannter Bugs und Festhalten der aufgewendeten Zeit verwendet.

Backup

Täglich werden Backups des Inhaltes von Redmine vorgenommen, damit im Stör- oder Verlustfall der Schaden auf ein Minimum reduziert wird. Das Backup wird auf Dropbox eines Projektmitglieds abgelegt, ist also Off-site und somit nicht betroffen, falls ein Problem mit den privaten Server eintritt.

A.7.3 Entwicklung

Versionierung

Der Source Code der Bachelorarbeit wird mittels dem Git-Versionskontrollsystem verwaltet und befindet sich auf dem offiziellen GitLab Server. Architekturbedingt werden mehrere Git-Repositories erstellt, es wird für jede Teilapplikation dieses Projektes ein Repository erstellt, um eine klarere Trennung zu ermöglichen.

Backup

Das Repository wird auf jedem Notebook der Teammitglieder und noch bedingt auf den privaten PC aktuell gehalten, sodass mindestens drei Kopien der Repositories bestehen.

Projektautomation

Für die Builderstellung, Automatisierung, Testing und Continuous Integration (CI) wird das Tool Jenkins eingesetzt, das unter https://jenkins.lag-13.ch für alle Developer verfügbar ist. cMake (Cross-platform C++ Build Generator) wird verwendet, um die CI umzusetzen. Verantwortlich für Jenkins ist Lukas Lätsch.

A.7.3.1 Vorgehen

Vorgehensmodell

Die Entwicklung erfolgt nach dem in der Vorlesung SSoftware Engineering 1"betrachteten Mischmodell aus RUP und Scrum, bekannt als Scrum+. Die Scrum-Meetings finden jeweils am Ende einer Iteration als Teil des Montagnachmittag-Meetings statt. Eine Iteration hat die Länge von einer Woche in der Elaboration Phase und zwei Wochen in der Construction Phase.

A.7.3.2 Unit-Testing

Vorgehen

Jeder Developer ist selbst dafür verantwortlich, dass sein Code von Unit Tests ausreichend abgedeckt wird und führt die Tests und Coverage-Metrics während dem Development auch selbst lokal aus. Im C++ werden hierzu CUTE-Tests verwendet. Diejenigen Developer, die mit ihm am selben Modul arbeiten, reviewen diese Tests nach deren Fertigstellung und prüfen, ob eventuell noch Edge Cases nicht abgedeckt sind, Überlegungsfehler vorliegen oder weitere Problemstellungen vorhanden sind. Fehlgeschlagene Tests werden unter keinen Umständen ignoriert, sondern das Problem (sofern sinnvoll) behoben, bevor mit dem folgenden Feature begonnen wird.

Automation

Zusätzlich wird auf dem Jenkins-Server ebenfalls getestet, indem die gesamte erstellte Test-Suite (inkl. Integration Tests) dort ebenfalls ausgeführt wird. Dies geschieht jeweils, wenn gebuildet wird. Auch hier gilt: Kein Feature hat höhere Priorität als das Reparieren des Builds.

Verwendung

Für komplexe Features kann bei Bedarf das Prinzip des Test-Driven Development verwendet werden. Die erwünschten Resultate eines neuen Features werden vor der Implementation in Mikrotests festgelegt, erst danach wird die tatsächliche Implementation erstellt.

Tools

Die Test-Coverage der in Cevelop und CLion entwickelten Module wird mit der dementsprechenden IDE überprüft.

A.7.3.3 Code Reviews

Zeitpunkt

Code Reviews erfolgen regelmässig nach Fertigstellung eines Features von ausreichender Komplexität mit mindestens einem anderen Developer, welcher am selben Modul mitarbeitet. Der Autor des Features erklärt seinen Code den anderen Beteiligten, welche ihm Feedback und gegebenenfalls Verbesserungsvorschläge geben. Auf jeden Fall erfolgt ein Review am Ende einer Iteration. Nach Bedarf kann auch zwischendurch ein Review angesetzt werden.

Unterstützung

Sollten im Verlauf des Development an einer Stelle Schwierigkeiten auftreten, so wird bevorzugt auf Pair-Programming zurückgegriffen, wobei ebenfalls einer der Developer, die sich am gleichen Modul beteiligen, die Rolle des zweiten Augenpaares übernimmt.

Wenn spezifische unlösbare Probleme bestehen, wird bei Bedarf auf den Spezialisten an der Fachhochschule zurückgegriffen.

A.7.3.4 Code Style Guidelines

Die Richtlinien beinhalten folgende Punkte:

- Es soll der neuste Standard C++17 eingesetzt werden, wenn sich die Möglichkeit bietet.
- Die C++ Core Guidlines sollen dort eingesetzt werden, wo sie Sinn machen. Die Bewertung aus dem Modul C++ Advanced der einzelnen Guidlines wird höher gewertet.
- Die gelernten Richtlinien vom Fach C++ und C++-Advanced sollen vorgezogen werden.

A.7.4 Testen

Unit Tests

Unit Tests werden wie oben im dem Abschnitt A.7.3.2 Unit Testing beschrieben durchgeführt.

Integration Tests

Für Integration Testing wird Jenkins verwendet.

System Tests

System Tests werden von den Entwicklern jeweils am Ende einer Iteration anhand von Use Cases durchgeführt. Die Tests werden protokolliert.

B. Projektmanagement

B.1 Arbeitszeiten

Währen der Projektarbeit haben wir unsere Arbeitszeiten erfasst und kategorisiert. Aus den erfassten Daten konnten wir eine Statistik der Arbeitsstunden erstellen. Dabei haben wir die Arbeitszeiten in folgende Kategorien unterteilt:

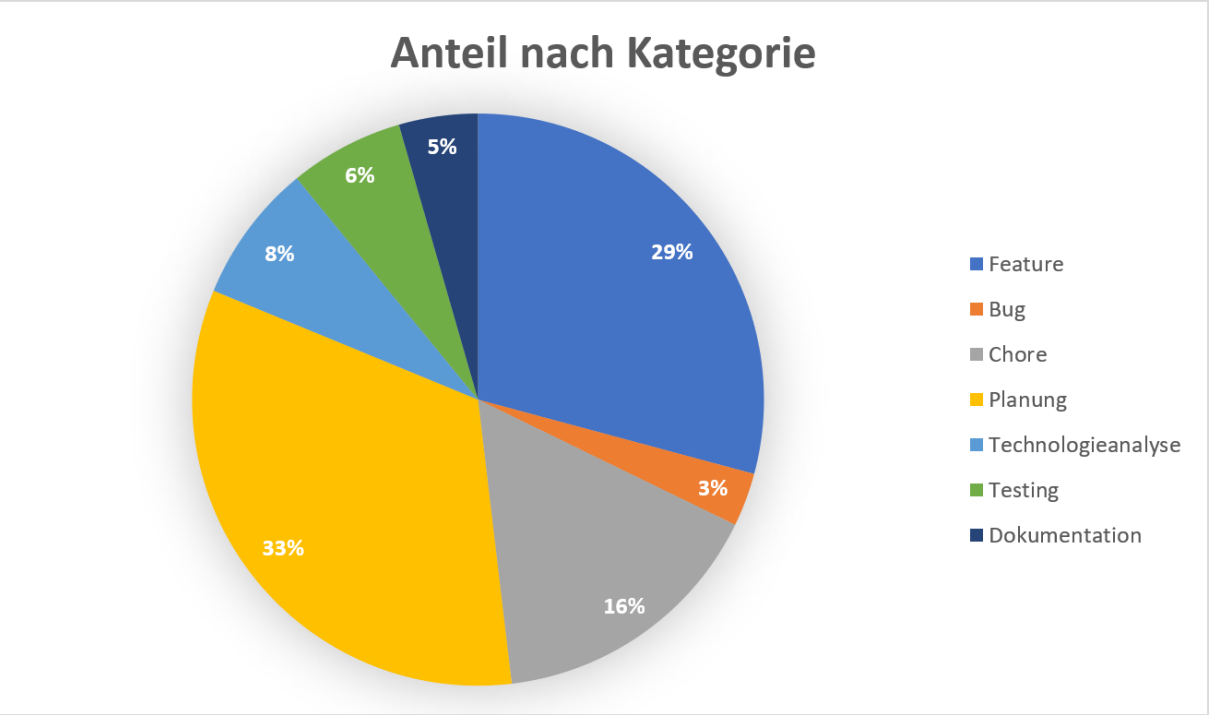


Abbildung B.1: Arbeitsaufteilung

Feature
Features beinhaltet die Softwareentwicklung sowie die Dokumentation.

Bug
Beheben von Fehlern in abgeschlossenen Arbeitspaketen.

Chore
Unterstützende Arbeiten wie Einrichtung der Entwicklungsumgebung.

Planung
Erarbeiten der funktionalen und nicht funktionalen Anforderungen sowie die Planung der Sprints, Sitzungen mit dem Betreuer und Reviews.

Technologieanalyse
Suchen und Testen geeigneter Libraries und Frameworks für den Einsatz in unserem System.

Testing
Testen der implementierten Features.

Dokumentation
Verfassen der Bachelor-Dokumentation.

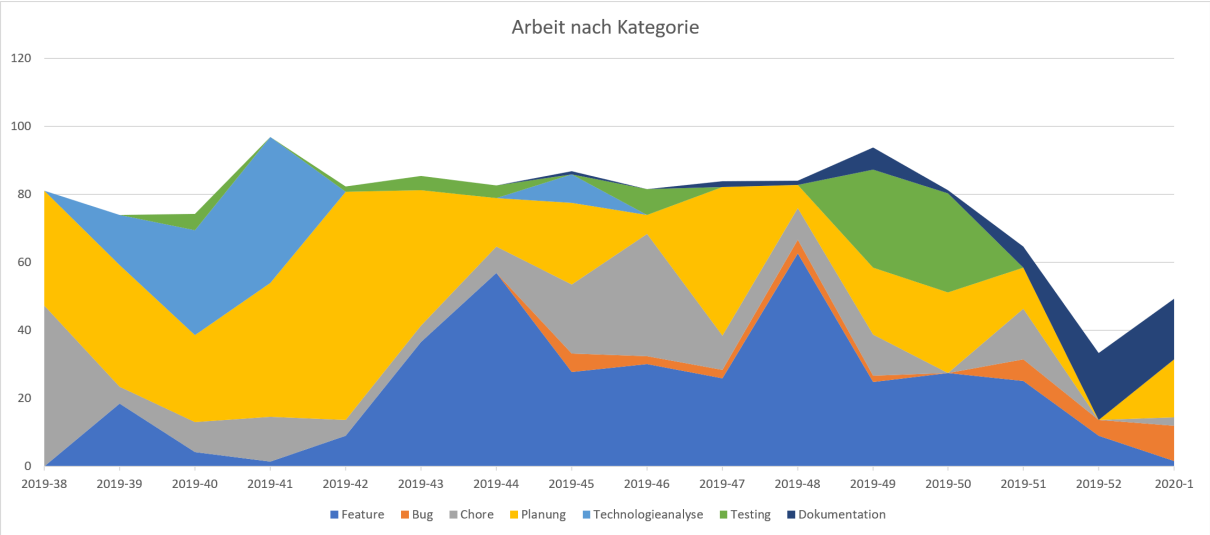


Abbildung B.2: Arbeitszeiten

Zeitschätzung

Vor Beginn eines Arbeitspaketen haben wir jeweils eine Zeitschätzung erfasst. Diese haben wir nach Abschluss des Pakets mit der tatsächlichen Zeit verglichen. In den meisten Fällen war die Abweichung im Rahmen von 15%. Zum Teil waren die Schätzungen zu hoch angesetzt und das Arbeitspaket wurden früher fertig. Etwas weniger Schätzungen waren zu niedrig angesetzt, wobei dort die Abweichung grösser war.

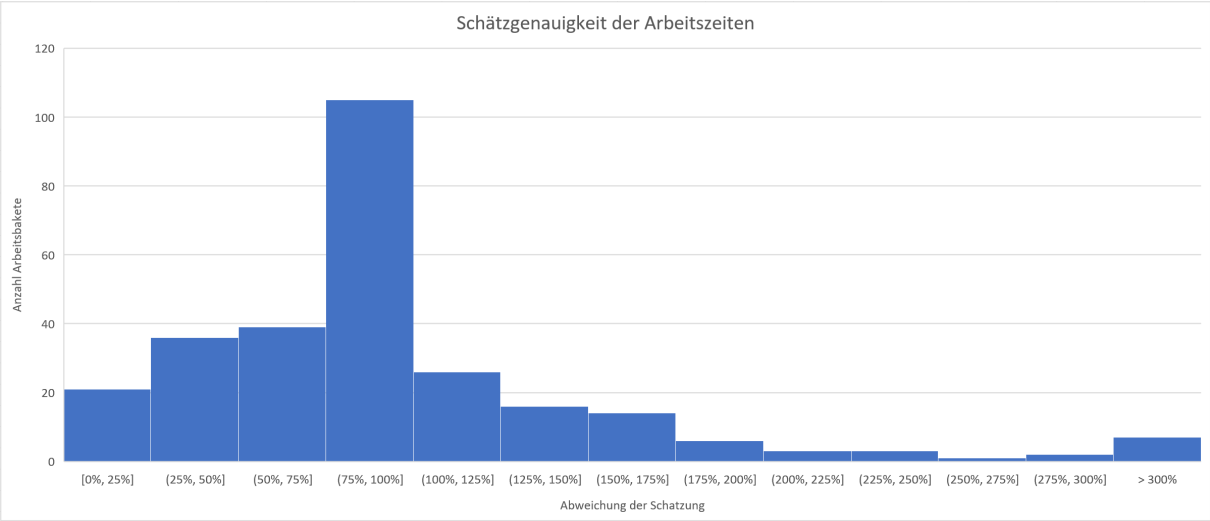


Abbildung B.3: Paket Schätzungen

C. Entwicklungsumgebung

Es gibt diverse Entwicklungsumgebungen zur Entwicklung von C++ Programmen. In den folgenden Unterkapiteln ist die Auswahl näher beschrieben.

C.1 Auswahl der Entwicklungsumgebung

In der Semesterarbeit wurde zuerst die IDE Cevelop verwendet, da sie in den C++ Modulen an der HSR verwendet werden. Cevelop basiert auf Eclipse CDT. Cevelop erweitert Elipse CDT um einige Plugins und die Integration des Test-Frameworks CUTE (Header-Only Library). Nach etwa der Hälfte der Semesterarbeit hat der Wechsel von Cevelop auf CLion aus folgenden Gründen stattgefunden:

- Das Projekt hat sich zu dieser Zeit vergrössert. Cevelop konnte nicht mehr alle Referenzen auflösen, was zur Anzeige von Fehlern führte. Der Compiler konnte die Referenzen auflösen und den Code fehlerfrei kompilieren.
- Der Buildserver hat das Projekt mit CMake gebildet. Lokal konnte man mit Eclipse und mit CMake builden. Die Usablity war jedoch nicht ideal. Mit CLion kann man dagegen die gleichen Buildkonfigurationen lokal und auf dem Buildserver verwenden.
- Cevelop hat Plugins, die auf Fehler aufmerksam machen, jedoch sind diese limitiert im Vergleich mit CLion, da CLion auch noch Clang-Tidy integriert. Clang-Tidy führt statische Analysen des Codes durch.
- In der Bachelorarbeit sind Google-Tests verwendet worden, diese werden in den CLion direkt integriert.
- Die Refactoring-Capabilities von CLion sind besser als die von Cevelop.

Die oben erwähnten Punkte haben dazu geführt, dass in der Bachelorarbeit von Beginn her CLion verwendet wurde.

C.2 CLion

CLion ist eine lizenzpflichtige Software von JetBrains. Für Studenten gibt es jedoch eine kostenfreie Lizenz.

C.2.1 Verwendung

CLion kann für Windows und Linux installiert werden, wobei Linux vorzuziehen ist, da C++ Libraries für das Projekt einfacher installiert werden können. Zudem funktioniert das Kompilieren der MK-TFHE Library auf Windows native nicht. Unter Windows ist es mithilfe von WSL auch möglich, CLion ohne allzu grosse Umstände zum Laufen zu bringen. Die Libraries werden dann auf dem WSL System installiert, was zum Beispiel ein Debian sein kann.

C.2.2 Tücken

CLion ist eine komfortable IDE, es gibt jedoch unter anderem folgende Tücken:

- Die IDE hat einen beachtlichen RAM Verbrauch. Sie braucht sonst auch viele rechnerische Ressourcen.
- Das Anzeigen von Syntax und Programmierfehlern dauert eine Weile. Somit besteht die Unsicherheit, ob wirklich ein Fehler besteht, oder ob die Ansicht noch nicht aktualisiert wurde.

- In den meisten IDEs kann die Wahl von Debug und Release Builds in der IDE selbst konfiguriert werden. Da CLion mit CMake arbeitet, muss dort der Buildtype angegeben werden.

Im Vergleich zu Cevelop und Qt-Creator ist CLion immer noch viel komfortabler, auch wenn man die oben erwähnten Tücken berücksichtigt.

D. Installationsanleitung

D.1 Anforderungen

D.1.1 Linux

Unter Debian / Ubuntu werden die folgende Pakete benötigt: cmake git libboost-atomic-dev libboost-thread-dev libboost-system-dev libboost-date-time-dev libboost-regex-dev libboost-filesystem-dev libboost-random-dev libboost-chrono-dev libboost-serialization-dev libwebsocketpp-dev openssl libssl-dev ninja-build libcpprest-dev libfftw3-dev build-essential gcc-9 g++-9 cpp-9 libtbb-dev libsqlite3-dev

Für andere Distributionen können die Namen unterschiedlich sein.

D.1.2 Windows

In Windows kann WSL verwendet werden, um die Komponenten zu verwenden. WSL ist ein Virtualisiertes Debian, somit sind die zu installierenden Pakete dieselben wie im vorherigen Kapitel.

D.2 Build

D.2.1 C++

Der Build an sich ist einfach, wenn alle Abhängigkeiten installiert sind, kann er mit den folgenden Befehlen gestartet werden:

```
1 mkdir build
2 cd build
3 cmake .. -DENABLE_TESTS=off -DENABLE_FFTW=on -DENABLE_NAYUKI_PORTABLE=off -
      DENABLE_NAYUKI_AVX=off -DENABLE_SPQLIOS_AVX=off -DENABLE_SPQLIOS_FMA=off -
      DBENCHMARK_ENABLE_TESTING=off -DGELFCPP_BUILD_TESTS=off
4 cmake --build .
```

Diese Befehle generieren alle C++ Teile des Projektes. Im Buildordner können nun folgende Programme gefunden werden.

- **Trustee**: build/trustee/server/mk-tfhe-trustee-server
- **Bulletinboard**: build/bulletinboard/server/mk-tfhe-bulletinboard-server
- **Voter**: build/voter/ui/mk-tfhe-voter-ui

D.2.2 Java

Der Coordinator ist ein Maven Projekt. Er kann mit dem Folgenden Befehl erstellt werden.

```
1 mvn install
```

Dieser Befehl erstellt zwei wichtige Komponenten, zum einen das Coordinator.jar in 'target/coordinator-[VERSION].jar' und zum anderen die benötigten Libraries im Ordner target/lib.

E. Bedienungsanleitung

E.1 Bedienung

E.1.1 Coordinator

E.1.1.1 Start

Der Coordinator muss als Erstes gestartet werden. Dies kann erreicht werden mit dem Befehl:

```
java -jar .\coordinator-[VERSION].jar
```

Wichtig ist dabei lediglich, dass der lib Ordner im selben Verzeichnis ist wie das Jar-File.

E.1.1.2 Erstellung einer Wahl

Eine neue Wahl kann unter */admin* erstellt werden. Die standard Login-Daten sind admin und a47S3x5de2hvsBraD8GXY8K8.
Wichtig ist, dass eine Wahl, nach dem sie erstellt ist, auch gestartet wird, bevor Trustees damit verbunden werden.

E.1.2 Trustee

Nach dem Start des Coordinators können Trustees hinzugefügt werden. Ein Trustee kann über das Executeable gestarted werden. Bei Bedarf kann der Port und der Hostname konfiguriert werden. Dies kann über das Setzen der Environment-Variablen gelöst werden.

```
export COORDINATOR_URI=https://coordinator.ba.lag-13.ch/
export SELF_HOSTNAME=trustee.ba.lag-13.ch
export SELF_PORT=42044
export VOTE_ID=1
./mk-tfhe-trustee-server
```

Dabei ist wichtig, dass der SELF_HOSTNAME und SELF_PORT dem externen Hostnamen und Port entspricht, da der Coordinator eine Verbindung zu diesen Hostnamen herstellen können muss.

E.1.3 Bulletin Board

Nach dem Verbinden aller Trustees können die Bulletin Boards hochgefahren werden. Die Schlüsselgenerierung der Trustees muss noch nicht abgeschlossen sein, aber die Trustees müssen dem Coordinator bekannt sein.
Der Start eines Bulletin Board ist identisch mit demjenigen des Trustees.

```
export COORDINATOR_URI=https://coordinator.ba.lag-13.ch/
export SELF_HOSTNAME=bulletinboard.ba.lag-13.ch
export SELF_PORT=43044
export VOTE_ID=1
./mk-tfhe-bulletinboard-server
```

E.1.4 Voter

Der Voter kann in der Theorie von Beginn an gestartet werden. Dies ist aber nicht sinnvoll, solange keine Wahlen bereit sind. Der Voter kann mit dem folgenden Befehl gestartet werden:

```
./mk-tfhe-voter-ui
```

Der Voter verfügt über zwei wichtige Funktionen. Erstens kann eine Wahl abgegeben werden und zweites kann der aktuelle Zustand der Wahl überprüft werden.

Die Voter-Applikation speichert seinen API-Key in der *apiKey.txt* Datei ab. Nach dem neustarten der Voter-Applikation wird der API-Key gelesen, sodass sich der Voter gegenüber dem Coordinator wieder authentifizieren kann. Der Vote-Status kann somit nach dem Neustart noch angezeigt werden. Es muss jedoch beachtet werden, wenn der Coordinator neugestartet wird, dass der API-Key nicht mehr gültig ist und deshalb zuvor die API-Key Datei gelöscht werden muss.

E.1.5 Docker

Alternativ zu den Executeables ist es auch möglich Dockerfiles vom Jenkins zu beziehen.

- **Coordinator**: https://jenkins.lag-13.ch/job/MK-TFHE%20Coordinator/lastSuccessfulBuild/artifact/coordinator.docker.tar
- **Trustee**: https://jenkins.lag-13.ch/job/MK-TFHE%20Trustee/lastSuccessfulBuild/artifact/trustee.docker.tar
- **Bulletinboard**: https://jenkins.lag-13.ch/job/MK-TFHE%20BulletinBoard/lastSuccessfulBuild/artifact/bulletinboard.docker.tar

Diese können dann wie folgt geladen werden:

```
docker load -i [File]
```

Anschliessend können sie mit den folgenden Befehlen gestartet werden:

```
docker run -d -p 8080:8080 coordinator
docker run -d -p 42043:42043 -e COORDINATOR_URI=[coordinator url] -e EXTERNAL_HOSTNAME=[
    hostname] -e EXTERNAL_PORT=[port] trustee
docker run -d -p 42143:42143 -e COORDINATOR_URI=[coordinator url] -e EXTERNAL_HOSTNAME=[
    hostname] -e EXTERNAL_PORT=[port] bulletinboard
```

Dabei ist zu beachten, dass EXTERNAL_HOSTNAME und EXTERNAL_PORT die Adresse sein muss, über welche die Komponente vom Koordinator aber auch von den Wählern erreichbar sein muss.

F. MK-TFHE Basics und Parameter

F.1 Parametrierung

Beim Erstellen einer Wahl werden verschiedene Parameter für die Verschlüsselung benötigt. Dieses Kapitel gibt eine kurze Erklärung, was die Parameter bewirken.

Trustees

Anzahl Trustees, die für eine Wahl benötigt werden.
Standard: 3.

Bulletin Boards

Anzahl Bulletin Boards, die für eine Wahl benötigt werden.
Standard: 3.

sn

Grösse des A-Vektors der LWE Samples. Wird in der MK-TFHE Library n genannt.
Standard: 500.

ln

Grösse des A-Vektors der TLwe Samples. Wird in der MK-TFHE Library N genannt.
Standard: 1024.

dg

Grösse der Gadget Matrix.
Standard: 8.

generatedKeys

Die Anzahl öffentlichen Schlüssel die ein Trustee generiert.
Standard: 8192.

encryptionKeys

Die Anzahl öffentlicher Schlüssel mit der eine Stimmen verschlüsselt wird. Dabei gibt es eine Abwägung zwischen der Grösse des Rauschens und der Sicherheit. In der nachfolgenden Grafik kann die Sicherheit einer Schlüsselkombination abgelesen werden:

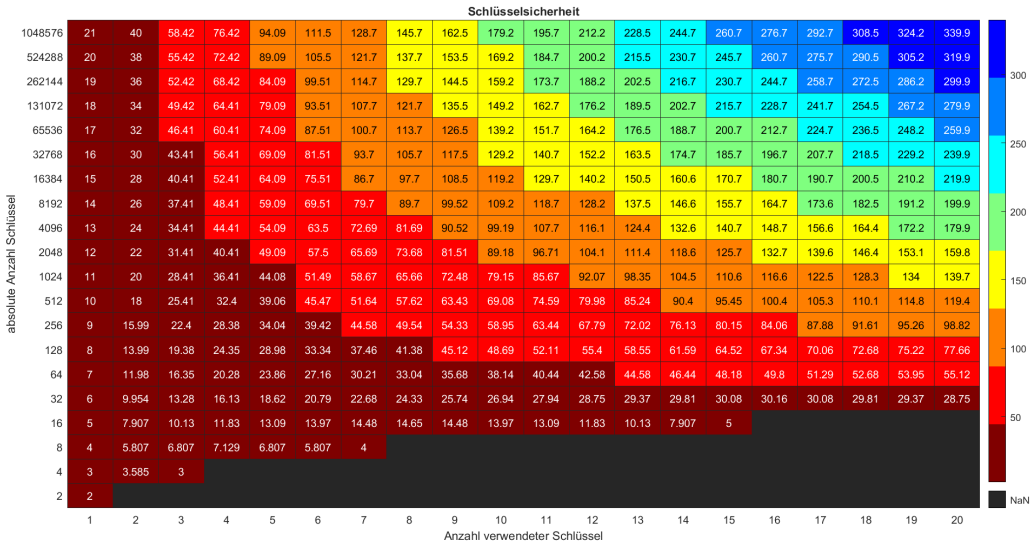


Abbildung F.1: Schlüsselsicherheit

Minimum bei 8192 Schlüssel und drei Trustees ist 12.

voteStart

Das Datum, ab dem Wähler wählen können.

voteEnd

Das Datum der letztmöglichen Stimmabgabe.

G. Selbstreflexion

Lukas Lätsch

Die Bachelorarbeit war für uns die Fortsetzung der Studienarbeit. Deshalb war die Einarbeitung in das Thema wesentlich einfacher als bei der Studienarbeit, da ein grosser Teil der Informationsbeschaffung bereits in der Studienarbeit geleistet worden war. Dennoch bin ich noch nicht vollumfänglich zufrieden mit meinem Verständnis der Verschlüsselung. Im Speziellen ist mir die Funktionsweise des Key-Switch-Algorithmus immer noch nicht klar. Da wir bei der Bachelorarbeit beschlossen hatten, die MK-TFHE Library nicht zu bearbeiten, konnten diese Algorithmen allerdings als Blackbox behandelt werden. Bei diesem Projekt haben wir uns dafür entschieden, sowohl C++ als auch Java zu verwenden. Dabei war es für mich interessant zu sehen, wo die verschiedenen Sprachen an ihre Grenzen kommen. Beispielsweise im Datalayer hat das Fehlen von Reflection in C++ einen wesentlichen Mehraufwand generiert. Im Allgemeinen war der Programmieraufwand in C++ grösser. Das liegt aber sicherlich auch an meinen Vorkenntnissen der beiden Sprachen. Die Durchführung des Projektes hat meiner Meinung nach gut funktioniert. Jedoch gäbe es für ein weiteres Projekt auch noch Verbesserungspotenzial. Beispielsweise hat die Verwendung von Scrum zwar funktioniert, jedoch besteht das Risiko, sich auf Details zu konzentrieren und dabei den Überblick zu verlieren. An einigen Stellen hätten wir viel Zeit sparen können, wenn wir vorausschauender geplant hätten. In Anbetracht der limitierten Zeit, bin ich mit den Ergebnissen des Projektes zufrieden. Wir haben einen funktionierenden Prototypen eines E-Voting Systems erstellt. Es gibt zwar noch verschiedene Verbesserungen, die wir im Kapitel Aussicht erwähnen und die notwendig sind, um ein marktfähiges Produkt zu erhalten. Die wichtigste Optimierung, die meiner Meinung nach gemacht werden müsste, ist eine Reduktion der Rechenzeit pro Stimme. Dies könnte beispielsweise erreicht werden, durch die Aufteilung des Bulletin Boards, sodass Loadbalancing der Votes möglich wird. Zusätzlich könnte der Key-Switch durch die Berechnung auf spezialisierter Hardware beschleunigt werden.

Rolf Furrer

In der Evaluation-Phase der Bachelorarbeit sind verschiedene Technologien ausgewertet worden. Dies waren Technologien für ein Web-Service und das GUI-Framework für den Voter-Client. Beim Web-Service wurde im Rückblick gesehen eine gute Wahl getroffen. Beim Coordinator wurde Java mit Spring verwendet und für die sonstigen Applikationen CPPRestSDK. Das GUI-Framework hätte aus jetziger Sicht besser gewählt werden können. Die Bewertung umfasste einige Punkte, jedoch nicht das Buildsystem einer der GUI-Applikation. Im offiziellen Tutorial und auch sonst wird die GUI Library gtkmm mit Meson gebildet und nicht mit CMake. Ein Vorteil ist, dass das Builden einer Applikation mit Meson einfacher ist, da gtkmm dafür optimiert ist. Meson unterstützt auch die einfache Integration von Glade Artefakten, was die Architektur sauberer und die Einfachheit der Gestaltung von statischen Inhalten vereinfacht. Meson hätte auch parallel in der Arbeit mit CMake verwendet werden können, jedoch ist der Aufwand grösser und, da Meson neu ist, war es fragwürdig, ob Jenkins Meson schon gut unterstützen wurde.

Die Wahl von Google-Test anstatt CUTE, war ein guter Entscheid, da CLion dies unterstützt und dadurch die Integration in die IDE nahtlos ist. Des Weiteren hat Google-Test mehr Features und unterstützt zum Beispiel Test-Fixtures.

In dieser Bachelorarbeit sind diverse Technologien eingesetzt worden, die Zeit des Projektes war jedoch beschränkt. Mit mehr Zeit oder auch mehr Erfahrung hätte die Wahl des JSON-Parsers (CPPRestSDK eingesetzt) besser getroffen werden können. Die Grösse der JSONs, im E-Voting System die geparst werden müssen, sind beachtlich, was bei der Evaluation nicht erwartet wurde. Das Parsen vom Voting-Package auf dem Voter-Client dauert beispielsweise eine Weile. Mit etwas mehr Zeit hätte zum Beispiel simdjson ausgewählt werden können. Simdjson unterstützt SIMD Instuktionen, was die Geschwindigkeit des Parsens, gemäss der Auswertung von simdjson (auf Github), beschleunigen würde.

Das E-Voting-System braucht viele Ressourcen, es braucht vor allem aber viel RAM. Die IDEs

(CLion und IntelliJ Ultimate Edition) von IntelliJ brauchen ebenfalls viel RAM. Das Arbeiten auf dem Notebook mit 16GB RAM war deswegen etwas umständlich, obwohl auch eine SWAP Partition verwendet wurde. Aufgrund der zuvor erwähnten Punkte ist es nicht möglich, das ganze E-Voting-System auf einem Notebook mit 16GB zu testen. Die Netzwerkfähigkeit schafft hier Abhilfe.
Das Arbeiten mit der MK-TFHE Library war nicht immer trivial. Es gab Situationen, in denen das Verstehen der dazugehörigen Doktorarbeit und der nachfolgenden Arbeiten sehr hilfreich gewesen wäre.
Das E-Voting-System ist ein grösseres Projekt im Vergleich zu den anderen Softwareprojekten im Studium. Es war eine bereichende Erfahrung, das Gelernte aus dem Studium in die Bachelorarbeit einfliessen lassen zu können. Das Projekt hat die Erfahrung mit dem Programmieren ebenso gestärkt. Die Arbeit im Team war produktiv und zielorientiert. An den Reviews hat man sich gegenseitig unterstützt, Probleme gelöst und architektonische Entscheide besprochen. Während der Arbeit gab es wenige Situationen, bei welchen nicht eine gemeinsame Meinung bestand. Wenn es so war, hat man zusammen immer einen guten Kompromiss gefunden.

Romeo Spinas
Insgesamt fand ich die Bachelorarbeit gut. Das Projekt mit Scrum zu machen, aber anfangs doch mehr zu planen, half uns die einzusetzenden Technologien zu evaluieren. Die Einschränkungen der bereits bestehenden MK-TFHE-Library waren absehbar, eine eigene Implementation der Funktionen hätte jedoch zu viel Zeit beansprucht. Das Arbeiten mit der bestehenden Library war nicht immer einfach, es war keine Parallelisierung vorhanden und in C++ muss man ohne Reflections auskommen. Diese Schwierigkeiten konnten jedoch überwunden werden. Die Arbeitsumgebung mit CLion und IntelliJ war nicht schlecht, aufgrund des hohen RAM-Verbrauchs unserer Applikation hat sich auch der Einsatz eines Jenkins-Build-Servers gelohnt. Das Zusammenspiel der verschiedenen Programmiersprachen war teilweise anstrengend. Da wir die Kommunikation zwischen den einzelnen Komponenten über eine JSON-Webschnittstelle realisiert haben, hat sich der Aufwand in Grenzen gehalten. Das Übersetzen von Daten in C++ zu JSON war generell anstrengend, eine andere Library hätte das etwas verbessert aber nicht behoben.
Die Bachelorarbeit ist zwar das grösste Projekt wärend des Studiums, jedoch bei Weitem zu wenig, um eine ganze E-Voting-Applikation auf Basis einer neuen Verschlüsselung zu schreiben. Wir haben uns auf einen kleinen funktionstüchtigen Teil konzentriert und das weniger Wichtige von Anfang an weggelassen. So können wir zum Schluss ein System präsentieren, das eine Abstimmung ermöglicht. Auch wenn noch einiges gemacht werden muss, um ein marktreifes Produkt zu erhalten, war es trotzdem interessant die Software im Team zu planen und zu programmieren.

H. Eigenständigkeitserklärung

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

Erklärung
Ich erkläre hiermit,

- dass ich die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde.
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.
- das ich keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt habe.

Ort, Datum:
Rapperswil, 8. Januar 2020

Lukas Lätsch

Rolf Furrer

Romeo Spinas

I. Software Lizenz

I.1 MK-TFHE

Die Library auf der das E-Voting-System aufgebaut ist, basiert auf der veröffentlichten MK-TFHE Library die unter https://github.com/ilachill/MK-TFHE erhältlich ist. Die Library hat eine Apache License, Version 2.0.

I.2 Boost

Die Boost-Library wird im E-Voting-System verwendet, diese hat die Boost Software License. Mehr Informationen können auf der Webseite https://www.boost.org/ nachgeschlagen werden.

I.3 FFTW3

Die FFTW3 Library, die in der ursprünglichen Library und im E-Voting-System verwendet wird, ist unter der GPL lizenziert. Die offizielle Webseite ist unter http://www.fftw.org/ erreichbar.

I.4 E-Voting-System

Unter Beachtung der oben genannten Lizenzen wird die das E-Voting-System unter die Apache License, Version 2.0 gesetzt.

I.5 Urheberrecht und Nutzungsrechte

Die Urheber- und Nutzungsrechte sind auf der folgenden Seite angefügt.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
FHO Fachhochschule Ostschweiz

Vereinbarung über Urheber- und Nutzungsrechte

1. Gegenstand der Vereinbarung
Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Bachelorarbeit Post-Quantum E-Voting von Lukas Lätsch, Rolf Furrer, Romeo Spinas unter der Betreuung von Prof. Dr. Andreas Steffen geregelt.

2. Urheberrecht
Die Urheberrechte stehen den Studenten zu.

3. Verwendung
Die Ergebnisse der Arbeit dürfen sowohl von den Studenten sowie von der HSR nach Abschluss der Arbeit verwendet und weiter entwickelt werden.

Rapperswil, den 8. Januar 2020

Lukas Lätsch

Rolf Furrer

Romeo Spinas

Prof. Dr. Andreas Steffen