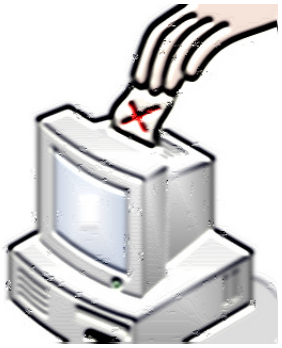
Anonyme Bestätigung der Unterrichtsbeurteilung



Studienarbeit 1

von

Stefan Hardegger
Thomas vogler

Sommersemester 2007

Betreuer:
Prof. Dr. Andreas Steffen

Hochschule für Technik Rapperswil
Fachbereich Informatik
Institut für Internet Technologien und Applikationen

Kapitel 1

Abstract

1.1 Zielsetzung

Seit ein paar Jahren können die Studierenden mit Hilfe der HSR Q-Feedback Webseite den Unterricht bewerten. Leider wird diese Möglichkeit zu wenig genutzt, weil eine ausreichende Anonymisierung nicht gewährleistet wird. Um den Rücklauf zu erhöhen, könnte das Ausfüllen der Fragebogen zur Pflicht gemacht werden. Dies bedingt, dass der Q-Feedback Server eine anonymisierte Bestätigung an die Studierende abgibt, die keine Kopplung zwischen dem Fragebogen und der Identität zulässt.

1.2 Ergebnisse

Im Rahmen dieser Studienarbeit wurde zu Beginn eine Übersicht über die bestehenden Verfahren erarbeitet. Ein Überblick über die rechtliche Lage, die Anforderungen an ein Internet-Wahlsystem und die mathematischen Grundlagen bilden die Kernpunkte.
Anschliessend wurde eine Demo-Version eines webbasierten Prototyps entwickelt. Dieser basiert auf einem Paper von Cramer, Gennaro und Schoenmakers aus dem Jahre 1997. Die eingebauten Vorteile sind:

- Privacy wird durch Verschlüsslung der Stimme gewährleistet.
- Verteilte Authoritäten verhindern Missbrauch durch eine einzelne Auszählinstanz.
- Signierter Nachrichtenaustausch ermöglicht es Fehler zu erkennen
- Effizienter Datenverkehr

1.3 Ausblick

Es sind Erweiterungen möglich, beispielsweise die Authentisierung der Studierenden mit einem Public Key Verfahren. Ein Zero Knowledge Proof Verfahren und blinde Signaturen können eingebaut werden. Des Weiteren können Funktionen zum Bulletin Board entwickelt werden, das für die Speicherung der Abstimmungen zuständig ist. Sämtliche Funktionen können in die Webplattform integriert werden.

In einer möglichen Diplomarbeit werden wir uns diesen interessanten Punkten widmen.

Inhaltsverzeichnis

Kapitel 2

Management Summary

2.1 Ausgangslage

2.1.1 Motivation

Seit ein paar Jahren können die Studierenden mit Hilfe der HSR Q-Feedback Webseite den Unterricht bewerten. Leider wird diese Möglichkeit nur von 20 - 30% der Studierenden genutzt, weil eine ausreichende Anonymisierung nicht gewährleistet wird. Das Vertrauen in die Applikation wurde bei einer Live-Demo im Jahre 2006 empfindlich geschwächt. Vor dem Informatikstudiengang wurde die eindeutige Verknüpfung zwischen dem ausgefüllten Fragebogen und dem Studenten gezeigt. Der Name des Studenten schmückte ein ausgefülltes Formular, was eine grosse Verletzung der Anonymität darstellt. Dieser gravierende Fehler ist bei vielen Studenten bekannt und sie trauen diesem System nicht.
Die grundsätzlichen Verfahren zur Anonymisierung von elektronischen Wahlen und Abstimmungen wurden schon vor Jahren durch den Pionier David Chaum postuliert. Weitere Mathematiker und Kryptologen fügten wichtige Inhalte hinzu. Das Paper "A Secure and Optimally Efficient Multi-Authority Election Scheme" von Ronald Cramer, Rosario Gennaro und Berry Schoenmakers vom Jahre 1997 beschreibt ein Optimales Verfahren zum Datenschutz, universellen Nachweisbarkeit und Robustheit[CGS97].
Im Rahmen dieser Studienarbeit erarbeiteten wir eine Übersicht über die bestehenden Verfahren und erstellten eine Demoversion einer anonymen Bestätigung. Die Kopplung zwischen den einzelnen Elementen wurde auf ein Minimum reduziert. Die Daten einer Wahl werden verschlüsselt gespeichert und werden über mehrere Authoritäten ausgezählt.

2.1.2 Vergleichbare Arbeiten

Die Thematik der digitalen Wahlen stösst auf ein breites Interesse. Das EU-Parlament möchte die Möglichkeit zur Wahl über das Internet salonfähig machen und damit Europa zusammenwachsen lassen. Der Kanton Zürich hat ein Pilotprojekt über E-Voting von der Firma UniSys entwickeln und testen lassen. In Deutschland werden ebenfalls ähnliche Projekte vorangetrieben.
Leider funktionieren diese Systeme auf dem Prinzip von "Security by obscurity". Dies bedeutet, dass die Architektur geschützt wird und keine technischen Spezifikationen an die Öffentlichkeit gelangen. Die mangelnde Transparenz und unklaren Methoden zur Ergebnissermittlung erschweren die Vertrauensbildung in der Wahlbevölkerung.

2.1.3 Ziele

Die Thematik um Anonymisierung interessiert uns sehr und wir sind uns über diese Herausforderung bewusst. Verschiedene Gebiete können wir in dieser interdisziplinären Arbeit verschmelzen.

Um den Rücklauf einer Umfrage zu erhöhen, könnte das Ausfüllen der Fragebogen zur Pflicht gemacht werden. Dies bedingt, dass der Q-Feedback Server eine anonymisierte Bestätigung an die Studierenden abgibt, die keine Kopplung zwischen dem Fragebogen und der Identität zulässt.
Die Demoversion AQFeedback bietet einen ersten Einblick an der HSR in E-Voting Verfahren.

2.2 Vorgehen

2.2.1 Planung und Arbeit

Die Arbeit gliederten wir in zwei Teile. Zuerst wurde vor allem theoretisches Wissen erarbeitet und im späteren Verlauf nahm die Entwicklung der Applikation mehr Zeit in Anspruch. Der Übergang geschah fliessend, sobald das theoretische Wissen in die Demoversion eingebaut werden konnte, wurde dies vorgenommen.
Die Funktionalität wurde modular entwickelt. Beispielsweise bestand eine Umfrage zu Beginn nur aus einer Stimme mit einer Wahlmöglichkeit "Ja" oder "Nein". In weiteren Schritten wurde eine Wahl von 1 aus n Antworten entwickelt, die Daten vom Client an den Server gesendet und in einer Datenbank gespeichert. Schliesslich übernimmt ein Auszählskript die Aufgabe zur Berechnung des Resultates.

2.2.2 Involvierte Personen

Betreuer:	Prof. Dr. Andreas Steffen
Studenten:	Stefan Hardegger
	Thomas Vogler

2.2.3 Ergebnisse

Zu Beginn erarbeiteten wir eine Übersicht über die bestehenden Verfahren. Ein rechtlicher Überblick, die Anforderungen an ein Internet-Wahlsystem und die mathematischen Grundlagen bildeten die Kernpunkte.
Die Demo-Version des webbasierten Prototyps AQFeedback wurde entwickelt. Die eingebauten Vorteile sind:

- Privacy wird durch Verschlüsslung der Stimme gewährleistet.
- Verteilte Authoritäten verhindern Missbrauch durch eine einzelne Auszählinstanz.
- Signierter Nachrichtenaustausch ermöglicht es Fehler zu erkennen
- Effizienter Datenverkehr

Der Prototyp wurde in Java entworfen und besteht aus den Teilprojekten Client (Applet), Server, Generator und Authority. Als Web-Applikationsserver wurde Apache Tomcat und als Datenbankserver PostgreSQL gewählt. Die Gestaltung und Logik der Webpage wurde mit Java Servlets und Applets gelöst, wobei das User Interface bewusst einfach gehalten worden ist. Die Kommunikation zwischen den einzelnen Elementen kann ohne grossen Aufwand mit SSL/ TLS Verschlüsselung gesichert werden. Folgende Funktionalität wurde implementiert:

- Client: Anzeige der Umfragen mit einem signierten Applet.
- Client: Ausfüllung und Übermittlung der Umfragen an den Server.
- WebServer: Zuständig für die Bereitstellung der Webseiten und Applets.
- WebServer: Authorisierung der Clients.

- WebServer: Empfang der Umfragen und Speicherung in Datenbank
- Generator: Erstellung von Umfragen.
- AuthorityInit: Initialisierung eines Umfrage-Systems.
- Auszähler: Auszählung der Umfragen mit x beliebigen Authoritäten.
- Datenbank: Speicherung der Daten in einer Datenbank (PostgreSql).
- Generell: Sichere Kommunikation durch Verschlüsselung der Daten mit TLS/SSL.

Die Folgende Grafik zeigt den Datenfluss. Des Weiteren ist erkennbar, wo was erledigt wird und welche Daten behandelt werden müssen.

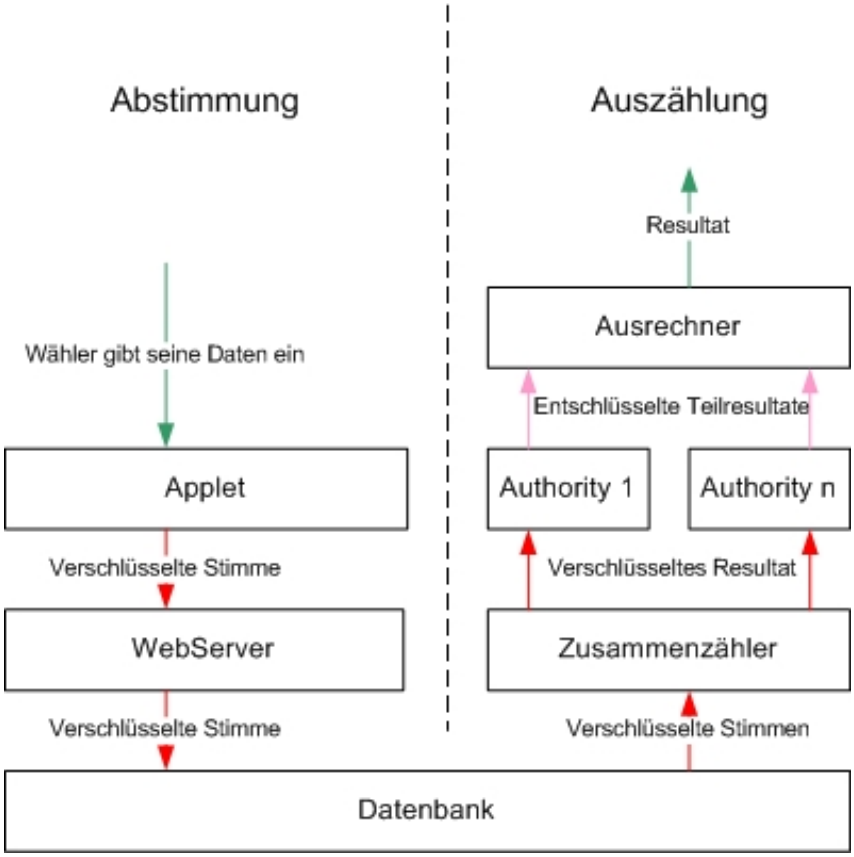


Abbildung 2.1: Datenfluss einfach

1. Der Benutzer startet den Webbrowser und öffnet die Homepage. Er baut eine SSL/TLS Verbindung mit dem Server auf.
2. Er authentifiziert sich und wählt auf dem Webserver eine Umfrage aus, für die er eine Berechtigung besitzt.
3. Die Umfrage wird im Browser dargestellt und der Wähler füllt seinen "Zettel" aus.
4. Das Applet verschlüsselt die Daten und erstellt eine Bestätigung für den Wähler.
5. Das Applet sendet die Wahl zum Webserver.

6. Der Webserver greift über die ODBC Schnittstelle und einer gesicherten Verbindung auf den Datenbank - Server zu und legt die relevanten Daten ab.
7. Nach Ablauf einer Umfrage werden die verschlüsselten Stimmen zusammengefasst.
8. verschiedene Authoritäten, das sind mindestens t aus einer Menge von n, entschlüsseln die Daten und errechnen Teilresultate.
9. Diese Teilresultate werden auf dem Ausrechner zusammengefasst, das Resultat kann publiziert und in die Datenbank gespeichert werden.

2.2.4 Lernpunkte

In dieser Arbeit haben wir sehr viel gelernt. Primär waren das Themen über elektronische Wahlsysteme, die dahinter liegenden mathematischen Verfahren zur Verschlüsselung und Anonymisierung. Unser Wissen in Webapplikationen und Java konnten wir erweitern. Zum ersten Mal haben wir in einem Projekt LaTeXzur Dokumentation benützt.

2.2.5 Ausblick

Es sind Erweiterungen möglich, beispielsweise die Authentisierung der Studierenden mit einem Public Key Verfahren. eine Zero Knowledge Verfahren und blinde Signaturen können eingebaut werden. Des Weiteren können Funktionen zum Bulletin Board entwickelt werden, das für die Speicherung der Abstimmungen zuständig ist. Sämtliche Funktionen können in die Webplattform integriert werden und die ausgetauschten Meldungen müssen überprüft werden. Die Usability und die grafische Gestaltung ist optimierbar.
In einer allfälligen Diplomarbeit werden wir uns diesen interessanten Punkten widmen.

Kapitel 3

Aufgabenstellung

3.1 Einführung

Seit ein paar Jahren können die Studierenden mit Hilfe der HSR Q-Feedback Webseite den Unterricht bewerten. Leider wird diese Möglichkeit meist nur von 20-30% der Studierenden genutzt. Um den Rücklauf zu erhöhen, könnte man das Ausfüllen des Online-Fragebogens zur Pflicht machen. Dies bedingt, dass der Q-Feedback Server eine anonymisierte Bestätigung an die Studierenden abgibt, die keine Kopplung zwischen dem soeben ausgefüllten Fragebogen und der Identität zulässt. Die Studierenden sollen aber dennoch eine persönliche Bestätigung erhalten, die sie z.B. zur Gutschrift der Kreditpunkte für das zu bewertende Modul vorweisen können.

3.2 Aufgabenstellung

Die grundsätzlichen Verfahren zur Anonymisierung von elektronischen Wahlen und Abstimmungen wurden schon vor Jahren durch den Pionier David Chaum postuliert. Im Rahmen dieser Studienarbeit soll zuerst eine Übersicht über die bestehenden Verfahren erarbeitet werden und anschliessend eine Demoversion einer web-basierten anonymen Bewertung mit persönlicher Bestätigung erstellt und getestet werden.

3.3 Links

A Secure and Optimally Efficient Multi-Authority Election Scheme
ftp://ftp.inf.ethz.ch/pub/crypto/publications/CrGeSc97.pdf

Rapperswil, 2. April 2007

Prof. Dr. Andreas Steffen

Kapitel 4

Einführung und Übersicht zu E-Voting

E-Voting oder auch Vote électronique bezeichnet die elektronische Wahl oder Abstimmung über das Internet. Zur Übermittlung des Wahl- oder Stimmentscheids können je nach Gestaltung auch andere elektronische Kommunikationssysteme wie das Mobilfunknetz genutzt werden.

4.1 Motivation

Seit ein paar Jahren können die Studierenden mit Hilfe der HSR Q-Feedback Webseite den Unterricht bewerten. Leider wird diese Möglichkeit nur von 20 - 30% der Studierenden genutzt, weil eine ausreichende Anonymisierung nicht gewährleistet wird. Das Vertrauen in die Applikation wurde bei einer Live-Demo im Jahre 2006 empflich erschüttert. Vor dem Informatikstudiengang wurde die eindeutige Verknüpfung zwischen dem ausgefüllten Fragebogen und dem Studenten gezeigt. Der Name des Studenten schmückte das ausgefüllte Formular, was eine grosse Verletzung der Anonymität darstellt. Dieser gravierende Fehler ist bei vielen Studenten bekannt.

Die grundsätzlichen Verfahren zur Anonymisierung von elektronischenWahlen und Abstimmungen wurden schon vor Jahren durch den Pionier David Chaum postuliert. Weitere Mathematiker und Kryptologen fügten wichtige Inhalte hinzu.

Die entwickelte webserverbasierte Demo-Version beruht auf der Veröffentlichung "A Secure and Optimally Efficient Multi-Authoriy Election Scheme" von Ronald Cramer, Rosario Gennaro und Berry Schoenmakers [CGS97]. Das Papier beschreibt ein Verfahren um eine Abstimmung oder Wahl anonymisiert durchzuführen. Das Schema ist sehr robust und die universelle Nachweisbarkeit ist gewährleistet. Die Privacy soll durch eine Verschlüsselung der Stimme gewährleistet werden, da ein Entschlüsselungsverfahren eingesetzt wird in welchem die verschlüsselten Stimmen zuerst kumuliert und nur das Endresultat entschlüsselt werden kann. Die Sicherheit beruht auf der Voraussetzung des diskreten Logarithmuses. Das Schema erfüllt alle Anforderungen ausser die "receipt freeness" wird nicht gewährleistet. Ein grosses Problem elektronischer Wahlen ist die Unüberprüfbarkeit (receipt-freeness). Selbst wenn ein Wähler mit einem Angreifer kooperiert, darf es ihm nicht möglich sein, beweisen zu können, wie er gewählt hat, da er sonst erpressbar oder bestechlich wird [MBC01].

4.2 Rechtlicher Rahmen

4.2.1 bisherige Entwicklung von E-Voting

Mit der Motion "Nutzung der Informationstechnologie für die direkte Demokratie"[M01] vom 9. Mai 2000 wurde der Bundesrat beauftragt einen Bericht über die Chancen und Risiken der E-Demokratie zu erstellen. Seit diesem Datum liess er für die digitale Demokratie und als Spezialgebiet für digitale Wahlen diverse Gutachten und Möglichkeitsstudien erstellen.
Folgende Anforderungen wurden in einem ersten Paper "Gutachten zum E-Voting" [L01] behandelt.

1. Akzeptanz und Selektivität elektronischer Stimmabgabe
2. Möglichkeiten und Folgen der E-Kommunikation und zusätzlicher Behördeninformation im Abstimmungs- und Wahlprozess
3. Beeinflussung der Stimmabgabe und Legitimationskraft des Urnengangs
4. Beeinflussung der Wahl- und Abstimmungsergebnisse durch E-Voting
5. Unterschreibung von Volksinitiativen und Referenden

Im Bundesratsbericht "Bericht über den Vote électronique - Chancen, Risiken und Machbarkeit elektronischer Ausübung politischer Rechte" [Br02] vom 9. Januar 2002 antwortet der Bundesrat auf die E-Voting betreffenden Teilaspekt der Motion. Darin wird die Realisierbarkeit des Vote électronique unter der Voraussetzung der "Lösung komplexer und schwieriger Sicherheitsprobleme" grundsätzlich für möglich gehalten. Zu diesem Zeitpunkt konnte nicht ausgeschlossen werden, dass diese Probleme lösbar seien. Die Einführung der elektronischen Stimmabgabe hätte das Funktionieren des demokratischen Systems gefährdet.

E-Voting wird dabei als Fortführung einer Entwicklung gesehen, die auf die Vereinfachung der Stimmabgabe zielt und schon mit der Einführung der brieflichen Stimmabgabe angegangen wurde. Dahinter steht die Hoffnung, dass durch Abbau von Hindernissen der Abwärtstrend bei der Stimmbeteiligung [L99] aufzuhalten oder zu mildern sei. Neben dieser Komponente sieht der Bundesrat E-Voting als Teil seiner E-Government-Strategie, mit der versucht wird den Staat und die direkte Demokratie an die Informations- und Kommunikationsgesellschaft anzupassen. Die Schweiz soll eine Führungsrolle in Europa einnehmen.

Die Auslandschweizer-Organisation (ASO) ist ein starker Befürworter des E-Voting, da die briefliche Stimmabgabe aus dem Ausland manchmal unzuverlässig ist und Auslandsschweizer durch E-Voting "die selben Voraussetzungen wie die Inlandschweizer in der Ausübung ihrer politischen Rechte" erhielten [ASO06]. Zudem bekommen viele das Stimm-Material zu spät, um es rechtzeitig zurückzusenden.

Nach der Durchführung von Pilotprojekten in Neuenburg, Genf und Zürich, die als Erfolge betrachtet wurden, hält der Bundesrat in seinem "Bericht über die Pilotprojekte zum Vote électronique vom 31. Mai 2006" [Br06] an einer schrittweisen Einführung von E-Voting fest. Mit dem Bericht gilt auch die Abklärung der Chancen und Risiken des Vote électronique als abgeschlossen.

4.2.2 Sicherheitsanforderung aus Sicht des Bundesrates

Der Bundesrat nennt folgende Punkte als Sicherheitsanforderung für das E-Voting [Br02]. "Elektronische Abstimmungssysteme müssen namentlich folgenden Sicherheitsanforderung genügen:"

1. Elektronisch abgegebene Stimmen dürfen weder abgefangen noch verändert oder umgeleitet werden können.
2. Vom Inhalt elektronisch abgegebener Stimmen dürfen Dritte keine Kenntnis erlangen können.
3. Nur stimmberechtigte Personen können teilnehmen.
4. Jede stimmberechtigte Person hat eine und nur eine Stimme.
5. Der Datenschutz muss gewährleistet sein.
6. Im Falle einer Panne darf keine bereits abgegebene elektronische Stimme verloren gehen.

Für den Bundesrat ist Transparenz und Nachvollziehbarkeit einer Wahl oder Abstimmung nicht relevant. Dieser wichtige Punkt wurde unsererseits missachtet.

4.2.3 Stimmgeheimnis

Aus den gesetzliche Bestimmungen in: SR 161.11 Verordnung über die politischen Rechte [SR161], können folgende Punkte für die digitalen Wahlen gefunden werden. Diese sind im Abschnitt 6a "Pilotversuche mit elektronischer Stimmabgabe" verankert. Folgende Punkte sind speziell erwähnenswert.

Art. 27f Verschlüsselung: 1 Die Massnahmen zur Wahrung des Stimmgeheimnisses müssen sicherstellen, dass elektronische Stimmen bei den zuständigen Behörden anonymisiert zur Auszählung eintreffen und nicht zurückverfolgt werden können.

Art. 27g Stimmgeheimnis: 1 Es sind sämtliche geeigneten Massnahmen zu treffen, damit ausgeschlossen werden kann, dass zwischen einer Stimme in der elektronischen Urne und der Person, die sie abgegeben hat, eine Verbindung hergestellt werden kann.

Art. 27l Technischer Stand: 1 Die bei den zuständigen Behörden eingesetzten technischen Komponenten, die Software, die Aufbau- und die Ablauforganisation werden vor jedem Urnengang nach neustem Stand der Technik beurteilt.

Art. 27h Weitere Massnahmen zur Sicherung des Stimmgeheimnisses: 2 Abgegebene Stimmen müssen in der elektronischen Urne anonymisiert gespeichert werden. Die Anordnung der gespeicherten Stimmen darf keinen Rückschluss auf die Reihenfolge des Stimmeneingangs ermöglichen.
4 Auf dem zur Stimmabgabe verwendeten Gerät muss die Stimme nach der Übermittlung durch den Stimmberechtigten unverzüglich ausgeblendet werden. Die verwendete Wahl- oder Abstimmungssoftware darf keinen Ausdruck der tatsächlich abgegebenen Stimme zulassen.

4.3 Technische Anforderungen

Der Deutsche Peter Wilm hat sich in "Elektronische Wahlen, Eine Informationsbroschüre für den Wahlbürger" [W04] sehr stark mit dem Thema auseinandergesetzt. Die Anforderungen hat er in folgende Kapitel unterteilt:

- Wahlgeheimnis
- Korrektheit des Ergebnisses
- Client-Rechnersicherheit
- Verfügbarkeit
- Transparenz

In vielen Punkten stimmt er mit der schweizerischen Gesetzgebung und dem Bundesrat überein. Abweichend, aber genau so wichtig sind seine Foderungen nach Transparenz. Dazu schreibt er:

> Soll die Legitimation der gewählten Volksvertreter in den Augen der Wähler nicht durch den Einsatz eines Onlinewahlsystems leiden, so muss dessen Funktionsweise mindestens ebenso transparent sein wie die des jetzigen Systems.
>
> Dabei geht es nicht darum, ob jeder Bürger tatsächlich jeden Aspekt des Systems nachvollzogen hat die Frage ist, ob er es könnte. Sicherlich ist eine formale Bauartzulassung des Bundesinnenministeriums notwendig, bei der das Ministerium eine Reihe von Gutachten einholt. Dies ist jedoch nicht ausreichend, soll wirkliches Vertrauen in der Bevölkerung aufgebaut werden. Dies kann nur mit einer rückhaltlosen Offenlegung jedes Systemdetails geschehen.

4.4 Bisherige Q-Feedback Plattform

Wie in der Motivation beschrieben, bestand oder besteht immer noch eine fixe Verbindung zwischen einer Abstimmung und einem Studenten. Da eine ausreichende Anonymisierung nicht gewährleistet wird, füllen nur 20-30% der Studierenden eine Umfrage aus. Über die bestehenden Sicherheitsmerkmale und technischen Anforderungen konnten wir nur wenige Infos in Erfahrung bringen.

4.5 Vergleichbare Systeme

Die Thematik der digitalen Wahlen stösst auf ein breites Interesse. Das EU-Parlament möchte die Möglichkeit zur Wahl über das Internet salonfähig machen und damit Europa zusammenwachsen lassen. Der Kanton Zürich hat ein Pilotprojekt über E-Voting von der Firma UniSys entwickeln und testen lassen. In Deutschland werden ebenfalls ähnliche Projekte vorangetrieben.
Leider funktionieren diese Systeme auf dem Prinzip von "Security by obscurity". Dies bedeutet, dass die Architektur geschützt wird und keine technischen Spezifikationen an die Öffentlichkeit gelangen. Die mangelnde Transparenz und unklaren Methoden zur Ergebnissermittlung erschweren die Vertrauensbildung in der Wahlbevölkerung.

4.5.1 Zürcher E-Voting

Das von der Firma Unisys entwickelte System für den Kanton Zürich wurde in der Infoweek vom 10.10. 2005 vorgestellt [PK05]. Im genannten Dokument wird das System erläutert und der Leser kommt den Eindruck vermittelt, dass sämtliche Vorgaben des Bundes und Sicherheitsspezifische Aspekte beachtet worden sind.
In den Zürcher Gemeinden Bertschikon, Bülach und Schlieren wurde die Plattform erfolgreich getestet. Ebenfalls wird es an der Universität Zürich seit 2004 für StuRa-Wahlen eingesetzt. Die Bürger haben die Möglichkeit, entweder über das Internet oder per SMS abzustimmen.
Bei einem Test mit Hilfe des Demo-Zuganges sind folgende Unklarheiten, Risiken und Fragen aufgetaucht.

Geburtsdatum: Warum muss der Wähler sein Geburtsdatum eingeben. Wozu wird das Geburtsdatum benötigt? Besteht eine Verbindung des Wählers zu seiner Abstimmung?

Url: Die URL scheint anfällig zu sein für einen XSS-Angriff. [V07]

Transparenz: Es gibt weder eine Beschreibung der Architektur noch über die eingesetzten Verschlüsselungsverfahren und mathematischen Grundlagen zu elektronischen Wahlen.

Bei einer telefonischen Nachfrage wurde uns versichert, dass alles sicher sei und unsere Bedenken unnötig seien. Eine eindeutigere Antwort bekamen wir nicht.

4.5.2 Deutsches i-Vote

Die Bundesrepublik Deutschland setzt sich ebenfalls mit E-Voting auseinander. Die Forschungsgruppe Internetwahlen hat in den Jahren 1999 und 2000 das Voting-System i-vote [For04] entwickelt. Dieses wird von der Firma ivl GmbH Leverkusen weiterentwickelt.
Das System setzt blinde Signaturen ein und legt sehr grossen Wert auf die Praxistauglichkeit. Die rechtlichen Aspekte und die Realisierbarkeit spielt eine entscheidende Rolle. Im Unterschied zur schweizerischen Gesetzgebung ist im Signaturgesetz die Verwendung von Smart-Cards vorgesehen. Mit dem System i-vote wurden bereits eine Reihe von Testwahlen durchgeführt.Das Projekt der Forschungsgruppe wird vom Bundesministerium für Wirtschaft und Technologie (BMWI) massgeblich gefördert.

Kapitel 5

Technische Grundlagen

5.1 ElGamal Verschlüsselung

Das ElGamal Verschlüsselungsverfahren[ElG85] ist ein asymmetrisches Verschlüsselungsverfahren, welches 1985 von Taher ElGamal erstmals beschrieben wurde. Seine Sicherheit besteht in der Annahme, dass diskrete Logarithmen nicht zurückgerechnet werden können.
Hier folgt nun ein kleines Beispiel einer Schlüsselgenerierung, Verschlüsselung und Entschlüsselung.

5.1.1 ElGamal-Schlüsselerzeugung

Für die Erzeugung eines ElGamal-Schlüsselpaares werden benötigt.

Beschreibung	Bezeichnung	Beispielwert
Hohe Primzahl	P	47
Hoher Primfaktor zu P üblicherweise $P = 2q + 1$	q	23
Generator (Prim zu P)	g	2
Geheimnis $0 < a < P - 2$	s	18
Öffentlicher Schlüssel $g^a \bmod p$	β	25

Der Schlüsselerzeuger behält das Geheimnis s für sich und verteilt den Öffentlichen Schlüssel, bestehend aus P, q, g und β.

5.1.2 ElGamal-Verschlüsselung

Um eine Nachricht zu verschlüsseln, so dass nur der Besitzer des Geheimnis die Nachricht entschlüsseln kann, muss die Nachricht in Elemente m eingeteilt werden, sodass $1 < m_n < q - 1$ ist. Ausserdem sollte eine Zufallzahl α generiert werden. Das Verschlüsselte Element besteht aus den zwei Elementen X und Y, welche wie folgt berechnet werden.

$$X = g^{\alpha} \bmod p$$

$$Y = m * \beta^{\alpha} \bmod p$$

Als Beispiel verschlüsseln wir hier nun eine Nachricht mit dem Nachrichtenwert $m = 14$. Als Zufallszahl wird $\alpha = 9$ verwendet.

Bezeichnung	Berechnung	Wert
X	$2^9 \bmod 47$	42
Y	$14 * 25^9 \bmod 47$	28

5.1.3 ElGamal-Entschlüsselung

Um aus den Werten X und Y die ursprüngliche Nachricht zu rekonstruieren wird folgende Formel verwendet:

$$m = \frac{Y}{X^s} \bmod p = Y * (X^s)^{-1} \bmod p$$

Mit unseren Werten X und Y eingesetzt errechnet sich der Ausgangswert:

Bezeichnung	Berechnung	Wert
m	$\frac{28}{42^{18}} \bmod 47$	14

5.2 Secret Sharing

Als "Secret Sharing" bezeichnet man den allgemeinen Vorgang ein Geheimnis in mehrere Teilgeheimnisse zu verteilen. Dies lässt sich auf verschiedene Arten realisieren und viele Kryptographen und Mathematiker haben sich mit diesem Problem auseinandergesetzt. Der Begriff "Secret Sharing" lässt sich grundsätzlich in zwei Kategorien einteilen.

5.2.1 (n,n)-Schwellwert Schema

Das einfachste Schema ist das (n,n)-Threshold Scheme oder auf Deutsch Schwellwert Schema, welches besagt, dass ein Schlüssel in n Teile zerlegt werden kann und für seine Rekonstruktion alle n Teile benötigt werden. Das simpelste Beispiel ist die Aufsummierung aller Teilgeheimnisse zum Geheimnis.

$$s = \sum_{i=1}^{n} s_i$$

Wie bereits erwähnt, ist die Rekonstruktion des Geheimnis s von allen Teilgeheimnissen abhängig. Geht ein Teilgeheimnis verloren lässt sich s nicht wiederherstellen.

5.2.2 (t,n)-Schwellwert Schema

Um den Verlust des Geheimnisses zu verhindern wurden verschiedene (t,n)-Schwellwerte Schemen erarbeitet. Die Idee dahinter ist, dass für die Rekonstruktion des Geheimnis t aus n Teilgeheimnissen benötigt werden um das Geheimnis wiederherzustellen. Bei einem (3,15)-Schema werden zum Beispiel 15 Teilgeheimnisse generiert, aus welchen beliebig 3 oder mehr ausgewählt werden können um das Geheimnis zu rekonstruieren. Die bekanntesten (t,n)-Schemas sind "Shamir's Secret Sharing" [Sha79] oder "Blackley's Secret Sharing"[Bla79] welche beide unabhängig voneinander 1979 vorgestellt wurden.
In unserer Arbeit wird "Shamir's Secret Sharing"' verwendet, welches von dem israelischen Kryptographen Adi Shamir entwickelt wurde. Das System basiert auf der Idee der Lagrange-Interpolation. Um das Geheimnis zu schützen wird ein Polynom des Grades t-1 generiert wobei die Stelle x^0 das Geheimnis bildet.

Beispiel mit t = 3 und n = 6:

Wir wollen das Geheimnis s = 1234 verteilen. Wir wählen zwei zusätzliche Zufallswerte sodass wir ein Polynom 2. Grades erhalten.

$$f(x) = 94x^2 + 166x + 1234$$

Wir erkennen das $f(0)$ das Geheimnis s zurückgeben wird. Aus diesem Polynom werden nun 6 Stützpunkte generiert (typischerweise 1-6) dessen Werte dann die Teilgeheimnisse bilden.

$$(1, 1494); (2, 1942); (3, 2578); (4, 3402); (5, 4414); (6, 5614)$$

Mit der Lagrange-Interpolation kann das Polynom nun mit 3 dieser Stützpunkten generiert werden. Die Durchführung einer Rekonstruktion wird im Kapitel "Projektspezifische Theorie und Beispiele" unter Abschnitt "Auszählung einer Abstimmung" durchgespielt.

5.3 A Secure and Optimally Efficient Multi-Authority Election Scheme

Die Arbeit von Ronald Cramer, Rosario Gennaro und Berry Schoenmakers[GGS97] beschreibt eine mögliche Umsetzung einer digitalen Wahl. Diese besitzt folgende Eigenschaften:

- Anonymität - Durch die Verschlüsselung der Stimme kann der Inhalt der Stimme zwar verifiziert, sein Inhalt jedoch nicht bestimmt werden.
- Bulletin Board - Die abgegebenen Stimmen werden auf einem Bulletin Board veröffentlicht. Nur der Wähler kann die Korrektheit seiner Stimme überprüfen
- Verteilte Authorities - Das Geheimnis wird durch "Shamir's Secret Sharing" auf verschiedene Authorities verteilt.

Argumente:

- Eine Verschlüsselung der Stimmen macht nur Sinn, solange gewährleistet werden kann, dass nicht eine einzelne Person das Geheimnis besitzt und damit einzelne Stimmen entschlüsseln oder sogar verändern kann.
- Eine abgegebene Stimme muss auf einen korrekten Inhalt überprüft werden können, ohne dessen Inhalt offenzulegen.
- Die Abgegebenen Stimmen müssen für alle öffentlich zu begutachten sein. Die Anonymität erfordert jedoch, dass deren Inhalt nicht erkannt werden darf.

Kapitel 6

Projektspezifische Theorie und Beispiele

In diesem Kapitel wird der Ablauf einer Abstimmung einer Abstimmung von Anfang bis Ende beschrieben und anhand eines fortlaufenden Beispiels veranschaulicht. Der Ablauf ist Grundsätzlich in 4 Stufen aufzuteilen und besteht aus:

- Generierung der Authorities und des Public Key
- Generierung einer Umfrage
- Durchführung der Abstimmung
- Auszählung der Abstimmung

Zur Erleichterung werden entsprechend kleine Beispielswerte benutzt. Als Parameter der ElGamal-Verschlüsselung werden die selben Wert wie im Kapitel "ElGamal-Schlüsselerzeugung" verwendet.

6.1 Generierung der Authorities

6.1.1 Theorie

Zur Generierung der Authorities werden sowohl die Parameter zur ElGamal-Schlüsserzeugung benötigt als auch die Angabe für die Anzahl der Authorities. Für die Generierung der "Shared Secrets" wird ausserdem der Schwellwert benötigt, also die für die Rekonstruktion des Geheimnis benötigte minimale Anzahl von Authorities.

Shared Secrets

Jede Authority bestimmt für sich eine Menge von Zufallswerten deren Anzahl dem Schwellwert entspricht und generiert daraus ein Polynom des Grades t-1. Jede Authority errechnet für die restlichen Authorities die Stützpunkte (typischerweise die Punkte $1..n$) und teilt diese den anderen Authorities mit. Die einzelnen Stützpunkte werden aufsummiert und der entsprechenden Authority zugewiesen.

Öffentlicher Schlüssel

Zur Generierung des öffentlichen Schlüssel der Abstimmung muss zuerst jede Authority für sich persönlich einen öffentlichen Schlüssel bestimmen. Das Initial-Geheimnis der Authority ist der

Schnittpunkt seine Polynoms mit der Y-Achse ($f(x = 0)$). Der Öffentliche Schlüssel der Abstimmung resultiert aus dem Produkt aller öffentlichen Schlüssel der Authorities.

$$\beta_A = \prod_{i=1}^{n} \beta_i$$

6.1.2 Beispiel

Als Beispiel folgt nun die Generierung von 3 Authorities mit dem Schwellwert 2:

Authority	Initialwerte	Öffentlicher Schlüssel β_i
A_1	5 - 2	32
A_2	6 - 3	17
A_3	7 - 4	34
Öffentlicher Schlüssel β		25

Aus den Initialwerten errechnet sich nun das Polynom sowie die Stützpunkte. Die einzelnen Stützpunkte zusammenaddiert ergeben die Shared Secrets

Authority	Polynom	x=1	x=2	x=3
A_1	$f(x) = 2x + 5$	7	9	11
A_2	$f(x) = 3x + 6$	9	12	15
A_3	$f(x) = 4x + 7$	11	15	19
Shared Secrets		4	13	22

6.2 Generierung einer Umfrage

6.2.1 Theorie

Bei der Generierung einer Umfrage werden für alle Kritierien/Fragen entsprechende Werte m definiert. Dabei gilt $0 < m < p - 2$. Hierbei wird ist das Verfahren bei einer einfachen Ja-Nein Wahl einfacher als bei einer Auswahl aus mehreren Werten.

6.2.2 Ja/Nein Abstimmung

Bei einer Ja//Nein-Abstimmung ist die Generierung von Abstimmungswerten einfach. So gilt für die Stimme "JA" der zufällig generierte Stimmwert m und für die Stimme "Nein" der enstprechende inverse Wert $1/m$.

6.2.3 1 aus n - Abstimmung

In einer Umfrage bei welcher aus mehreren Auswahlmöglichkeiten gewählt werden soll wird für jede einzelne Auswahlmöglichkeit ein entsprechender Wert m_n bestimmt. Die Ausgewählte Möglichkeit n erhält dann den Wert m_n falls dieser gewählt wurde oder $1/m_n$ falls er nicht gewählt wurde. Dieses System ermöglicht auch Abstimmungen mit Mehrfachauswahl.

6.2.4 Beispiel

Wir generieren hier ein Kriterium mit 3 möglichen Auswahlmöglichkeiten. Dazu wählen wir zufällige Stimmwerte m und berechnen deren inversen Wert.

Kriterium	1	2	3
Stimmwerte m	11	12	13
Stimmwerte $\frac{1}{m}$	30	4	29

6.3 Durchführung einer Abstimmung

6.3.1 Theorie

Der Benutzer tätigt seine Wahl/Bewertung und gibt diesen in den Client ein. Für jede Option in jedem Kriterium wird nun verschlüsselt:

- m - Wenn die Option gewählt wurde
- $\frac{1}{m}$ - Wenn die Option nicht gewählt wurde

Zusätzlich wird für jede Stimme (Gruppe von verschlüsselten Stimmwerten für ein Kriterium) und über den gesamten Stimmzettel ein MD5-Hash generiert. Die Optionen und die Hashes werden über einen Kanal an das Bulletin-Board gesendet und in der Datenbank gespeichert. Anhand der Hashes kann überprüft werden ob bei der Übertragung Fehler aufgetreten sind. Dem Benutzer wird eine Teilnahmebestätigung angezeigt. Er hat die Möglichkeit diese zu speichern und die Daten und Hashes mit den gespeicherten Daten auf dem Bulletin Board abzugleichen.

6.3.2 Beispiel

Ausfüllen der Frage. Als Zufallswert α für die ElGamal Verschlüsselung wird pro Stimmabgabe nur ein Wert benutzt. Normalerweise sollte für jede einzelne Verschlüsselung ein eigenes α gewählt werden. Zum einfacheren Verständnis wird darauf jedoch verzichtet. Wir geben hier 3 Stimmen ab. Einmal wählen wir die 2. Option und zweimal die 3. Option.

Stimme ($\alpha = 9$)

	1	2	3
Wahl		x	
Stimmwerte	30	12	29
X	42	42	42
Y	13	24	11

Stimme 2 ($\alpha = 10$)

	1	2	3
Wahl			x
Stimmwerte	30	4	13
X	37	37	37
Y	43	12	39

Stimme 3 ($\alpha = 11$)

	1	2	3
Wahl			x
Stimmwerte	30	4	13
X	27	27	27
Y	41	18	35

6.4 Auszählung einer Abstimmung

6.4.1 Theorie

Die Auszählung der in der Datenbank gespeicherten Stimmen geschieht mit Hilfe der Lagrange-Interpolation, für welche wir bei der Generierung der Authorities ja bereits die "Shared Secrets" berechnet haben. Um die Auszählung durchzuführen müssen natürlich die ElGamal-Parameter bekannt sein. Ausserdem müssen genügend Authorities anwesend sein um den Schwellwert zu erfüllen. Mit jeder Authority die zusätzlich Anwesend ist, kann die Authenzität der Authorities gewährleistet werden und fehlerhafte oder korrupte Authorities erkannt und ignoriert werden.
Die Authorities können das Geheimnis wiederherstellen in dem für jede Authority den dazugehörigen Lagrange-Koeffizienten λ_i generiert wird:

$$\lambda_i = \prod \frac{l}{l-i}$$

Der Lagrange-Koeffizient variiert je nachdem wie viele und welche Authorities anwesend sind. Das Geheimnis kann mit folgender Formel wiederhergestellt werden:

$$s = \sum \lambda_i * s_i$$

Für die Auszählung der Stimmen gibt es zwei Möglichkeiten.

Einzelne Auszählung

Die erste Möglichkeit besteht darin, dass jede einzelne Stimme an die Authorities verteilt wird und von dieser mit dessen "Shared Secret" entschlüsselt wird sodass $w_i = x^{s_i}$. Die Resultate werden danach mit den entsprechenden Lagrange-Koeffizienten potenziert und können dann anhand des Stimmwertes als Ja- oder Nein-Stimme identifiziert und gezählt werden.

$$m = \frac{y}{\prod {w_i}^{\lambda_i}} = \frac{y}{x^s}$$

Der Vorteil an dieser Auszählungsform ist, dass das Geheimnis nicht Rekonstruiert werden muss, da die eigentliche Entschlüsselung in jeder Authority einzeln geschieht. Als Nachteil zu bewerten ist die Tatsache, dass die Stimmen einzeln bekannt werden und daher Theoretische ein Rückschluss auf den Wähler möglich wäre.

Summierte Auszählung

Die zweite Möglichkeit greift dem zuvor indem die Stimmen bereis vor der Auszählung zusammengezählt werden. Dies ist Möglich, da die Stimmwerte für Ja und Nein invers zueinander sind. So werden zuerst alle verschlüsselten Werte zusammengezählt sodass:

$$X_{tot} = \prod x_i$$

$$Y_{tot} = \prod y_i$$

X_{tot} wird wie in der ersten Variante an alle Authorities gesendet und von diesen entschlüsselt: $W_i = {X_{tot}}^{s_i}$. Diese werden ebenfalls mit deren Lagrange-Koeffizienten potenziert und miteinander multipliziert. Als Resultat der Berechnung erhalten wir den Wert m^T.

$$\frac{Y_{tot}}{\prod {W_i}^{\lambda_i}} = \frac{Y_{tot}}{{X_{tot}}^{s}} = m^T$$

T repräsentiert dabei dem Wert der aufsummierten Stimmen wobei für $Ja = 1$ und $Nein = -1$ gilt. Dementsprechend gilt für die Anzahl Stimmen l:

- Nur Nein-Stimmen: $T = -l$
- Mehr Nein- als Ja-Stimmen: $-l < T < 0$
- Gleichviel Nein- wie Ja-Stimmen: $T = 0$
- Mehr Ja- als Nein-Stimmen: $0 < T < l$
- Nur Ja-Stimmen: $T = l$

Da sich T aus m^T wegen des Logarithmusproblems nicht direkt berechnet werden kann, muss nacheinander m^{-l} bis m^l durchgeprobt werden. Aus effizienztechnischen Überlegungen kann man bei einer Ausgeglichenen Abstimmung davon ausgehen das T nahe bei 0 liegen wird. Daher ist es für die meisten Fälle wahrscheinlich effizienter, wenn man mit der Berechnung bei 0 beginnt und in gleichmässigen Schritten in positiver und negativer Richtung testet.
Bei einer "1 aus n"- Abstimmung kommt noch die Tatsache hinzu, dass für jede abgegebene positive Stimme $n - 1$ negative Stimmen abgegeben werden. Daraus lässt sich schliessen, dass beim Probedurchlauf für jeden Schritt in die positive Richtung, $n - 1$ Schritte in die negative Richtung ausprobiert werden müssten.

6.4.2 Beispiel

Zuerst wollen wir Beweisen, dass die Rekonstruktion der Authorities mit Hilfe der Lagrange-Interpolation korrekt abläuft und funktioniert. Daher werden wir für alle möglichen Kombinationen von Authorities das Geheimnis konstruieren. Wir berechnen als erstes für die verschiedene Koeffizienten für die verschiedenen Kombinationsmöglichkeiten:

Authorities	λ_i	$\lambda_i \bmod q$
A_1 - A_2	$\frac{2}{1}$ - $\frac{1}{-1}$ - 0	2 - 22 - 0
A_2 - A_3	0 - $\frac{3}{1}$ - $\frac{2}{-1}$	0 - 3 - 21
A_1 - A_3	$\frac{3}{2}$ - 0 - $\frac{1}{-2}$	13 - 0 -11
A_1 - A_2 - A_3	$\frac{3}{1}$ - $\frac{3}{-1}$ - $\frac{1}{1}$	3 - 20 - 1
Shared Secrets β		4 - 13 - 22

Anhand dieser Koeffizienten und den "Shared Secrets" können wir das Secret mit einer beliebigen Kombination aus 2 oder 3 Authorities wiederherstellen:

Authorities	$\sum s_i * \lambda_i$	Geheimnis s
A_1 - A_2	$(4 * 2 + 13 * 22 + 22 * 0)$	18
A_2 - A_3	$(4 * 0 + 13 * 3 + 22 * 21)$	18
A_1 - A_3	$(4 * 13 + 13 * 0 + 22 * 11)$	18
A_1 - A_2 - A_3	$(4 * 3 + 13 * 20 + 22 * 1)$	18

In unserem Beispiel haben wir in 3 Stimmen je 3 Optionen bewertet. Die Auszählung geschieht mit den Authorities A_1 und A_3. Für diese kennen wir die Koeffizienten 13, 0 und 11 welche wir bereits errechnet haben. Wir werden beide Aufgezeigten Möglichkeiten zur Auszählung durchspielen, also zuerst alle Stimmen einzelnen auszählen.

Einzelne Auszählung

Als erstes zählen wir die Option 1 aus, für welche wir die folgenden (x,y)-Werte erhalten haben: (42,42),(37,16) und (17,24). Die beiden Authorities berechnen nun die Werte $w_i = x^{s_i}$.

Stimme	$w_1 = x^4$	$w_3 = x^2 2$	$\frac{y}{\prod {w_i}^{\lambda_i}}$	M
$V_1 = (42, 13)$	$42^4 \bmod 47 = 14$	$42^2 2 \bmod 47 = 28$	$\frac{13}{(14^{13} * 28^{11})} \bmod 47$	30
$V_2 = (37, 43)$	$37^4 \bmod 47 = 36$	$37^2 2 \bmod 47 = 14$	$\frac{37}{(36^{13} * 14^{11})} \bmod 47$	30
$V_3 = (27, 41)$	$27^4 \bmod 47 = 12$	$27^2 2 \bmod 47 = 7$	$\frac{27}{(12^{13} * 7^{11})} \bmod 47$	30

Wir erhalten hier 3-mal den Wert 30 welcher $\frac{1}{m}$ entspricht. Daraus können wir folgern, dass drei Mal nicht gewählt worden ist. Als erstes zählen wir die Option 2 aus:

Stimme	$w_1 = x^4$	$w_3 = x^2 2$	$\frac{y}{\prod {w_i}^{\lambda_i}}$	M
$V_1 = (42, 24)$	$42^4 \bmod 47 = 14$	$42^2 2 \bmod 47 = 28$	$\frac{24}{(14^{13} * 28^{11})} \bmod 47$	12
$V_2 = (37, 12)$	$37^4 \bmod 47 = 36$	$37^2 2 \bmod 47 = 14$	$\frac{12}{(36^{13} * 14^{11})} \bmod 47$	4
$V_3 = (27, 18)$	$27^4 \bmod 47 = 12$	$27^2 2 \bmod 47 = 7$	$\frac{18}{(12^{13} * 7^{11})} \bmod 47$	4

Bei der zweiten Option erhalten wir einmal Mal den Stimmwert $m = 12$ und zweimal den Stimmwert $\frac{1}{m} = 4$. Hier wurde also die Option einmal gewählt.

Stimme	$w_1 = x^4$	$w_3 = x^2 2$	$\frac{y}{\prod {w_i}^{\lambda_i}}$	M
$V_1 = (42, 11)$	$42^4 \bmod 47 = 14$	$42^2 2 \bmod 47 = 28$	$\frac{11}{(14^{13} * 28^{11})} \bmod 47$	29
$V_2 = (37, 39)$	$37^4 \bmod 47 = 36$	$37^2 2 \bmod 47 = 14$	$\frac{39}{(36^{13} * 14^{11})} \bmod 47$	13
$V_3 = (27, 35)$	$27^4 \bmod 47 = 12$	$27^2 2 \bmod 47 = 7$	$\frac{35}{(12^{13} * 7^{11})} \bmod 47$	13

Zweimal lautet bei dieser Entschlüsselung das Resultat $m = 13$. Daher wissen wir das zweimal Option 3 gewählt worden ist.

Summierte Auszählung

Für die summierte Auszählung werden als erstes alle (x,y) einer Option miteinander multipliziert.

Option	$X = \prod x_i$	$Y = \prod y_i$
Option 1	$42 * 37 * 27 \bmod 47 = 34$	$13 * 43 * 41 \bmod 47 = 30$
Option 2	$42 * 37 * 27 \bmod 47 = 34$	$24 * 12 * 18 \bmod 47 = 14$
Option 3	$42 * 37 * 27 \bmod 47 = 34$	$11 * 39 * 35 \bmod 47 = 22$

Wir erhalten für die Option 1 das Wertepaar (X,Y) = (34,30), für Option 2 (34,14) und für Option 3 (34,22). Die Authorities berechnen nun wiederum den Broadcast $W_i = X^{s_i]}$ welcher danach zusammengezählt wird.

Stimme	$W_1 = X^4$	$W_3 = X^{22}$	$\frac{Y}{\prod W_i^{\lambda_i}}$	m^T
$V_1 = (34, 30)$	$34^4 \bmod 47 = 32$	$18^22 \bmod 47 = 18$	$\frac{30}{(32^{13} * 18^{11})} \bmod 47$	22
$V_2 = (34, 14)$	$34^4 \bmod 47 = 32$	$34^22 \bmod 47 = 18$	$\frac{14}{(9^{13} * 4^{11})} \bmod 47$	4
$V_3 = (34, 22)$	$34^4 \bmod 47 = 32$	$34^22 \bmod 47 = 18$	$\frac{22}{(9^{13} * 4^{11})} \bmod 47$	13

Wir haben nun für jede Option das Resultat m^Terhalten. Für jede Stimme muss nun durch Erprobung T bestimmen werden. In unserem Fall mit 3 abgegebenen Stimmen sind wir auf 7 möglich Werte beschränkt: -3, -2, -1, 0, 1, 2 oder 3. Wir erproben nun für jede Stimme die möglichen Werte.

Option	m^0	m^1	m^2	m^3	m^{-1}	m^{-2}	m^{-3}	T
*Option*1	1	11	27	15	30	7	22	-3
*Option*2	1	12	3	36	4	16	17	-1
*Option*3	1	13	28	35	29	42	43	1

Wir haben nun die Werte T erhalten und sehen nun das Verhältnis zwischen Ja und Nein Stimmen.

- -3: 3 Mal Nein - 0 Mal Ja
- -1: 2 Mal Nein - 1 Mal Ja
- -2: 1 Mal Nein - 2 Mal Ja

Wir haben nun das Resultat der Abstimmung obwohl zu keiner Zeit das Geheimnis oder der Stimmwert einer einzelnen Stimme bekannt geworden ist.

Kapitel 7

Umsetzung

7.1 Übersicht

In diesem Kapitel wird der Aufbau und Ablauf unseres Prototypen erklärt und aufgezeigt. Neben der Datenbank, welche in fast allen Abläufen eine Rollespielt, ist das Kapitel in die gleichen vier Teile eingeteilt wie das Kapitel "Projektspezifische Theorie und Beispiele". Neben dem Applet und der Webapplikation haben wir die bedienung der restlichen Programme so simpel wie möglich gehalten und besonders das UserInterface beschränkt sich auf das nötigste.

Die Folgende Grafik zeigt den Weg einer Stimme durch das Sytem. Des Weiteren ist erkennbar, wo was erledigt wird und welche Daten behandelt werden müssen.

Abbildung 7.1: Datenfluss einfach

1. Der Benutzer startet den Webbrowser und öffnet die Homepage. Er baut eine SSL/TLS Verbindung mit dem Server auf.
2. Er authentifiziert sich und wählt auf dem Webserver eine Umfrage aus, für die er eine Berechtigung besitzt.
3. Die Umfrage wird im Browser dargestellt und der Wähler füllt seinen "Zettel" aus.
4. Das Applet verschlüsselt die Daten und erstellt eine Bestätigung für den Wähler.
5. Das Applet sendet die Wahl zum Webserver.
6. Der Webserver greift über die ODBC Schnittstelle und einer gesicherten Verbindung auf den Datenbank - Server zu und legt die relevanten Daten ab.
7. Nach Ablauf einer Umfrage werden die verschlüsselten Stimmen zusammengefasst.
8. verschiedene Authoritäten, das sind mindestens t aus einer Menge von n, entschlüsseln die Daten und errechnen Teilresultate.
9. Diese Teilresultate werden auf dem Ausrechner zusammengefasst, das Resultat kann publiziert und in die Datenbank gespeichert werden.

7.2 Datenbank

Zentrales Element des AQFeedback Systems ist eine PostgreSQL-Datenbank, welche die Informationen über Personen, Umfragen und Ergebnisse speichert.
Die folgenden Synonyme werden abwechselnd gebraucht und kommen meist aus der Englisch-Deutschen Übersetzung hervor.

Survey Umfrage: Bezeichnet den Namen einer Umfrage

Question Kriterium: Bezeichnet den Namen eines Kritierums, das zu einer Umfrage gehört.

Option Kriteriumbewertung: Jede Question besteht aus mehreren Bewertungen, die von $--$ bis $++$ gehen.

Vote Abstimmung: Die eingereichten Abstimmungen werden hier abgelegt.

Ballot Stimme: Eine Abstimmung besteht meist aus mehrere Stimmen.

Value Stimmwert: Jede Stimme hat verschiedene Stimmwerte.

7.2.1 Referentielle Integrität

Die referentielle Integrität befasst sich mit der Korrektheit zwischen Attributen von Relationen und der Erhaltung der Eindeutigkeit ihrer Schlüssel.
Die Beziehungen werden zuvor in einem Datenbanksystem festgelegt. Das Datenbanksystem wird dann diese Beziehungen zwischen den Relationen garantieren.
Die Referentielle Integriet wird bei der Vorstellung der Tabellenstruktur angesprochen.

7.2.2 Datenbank Diagramm

Die Datenbankstruktur wurde mit dem Programm Visual Paradigm erstellt. Die verschiedenen Tabellen, Spalten und Beziehungen werden in den nächsten Kapiteln ausführlich erklärt.

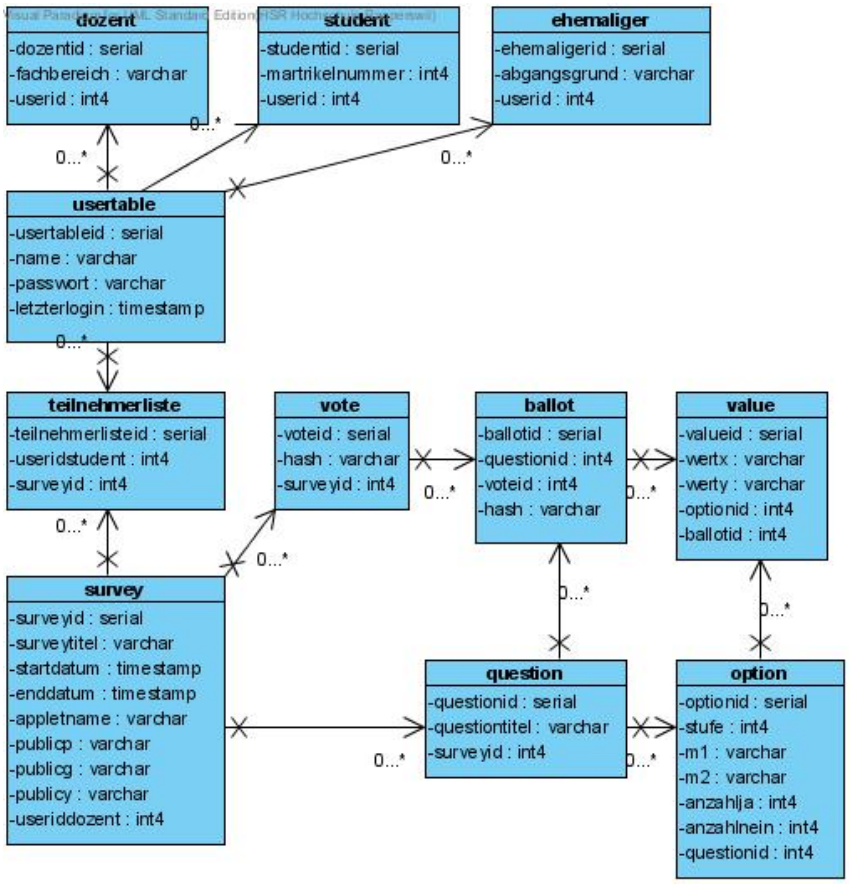


Abbildung 7.2: Datenbank Struktur als Klassendiagramm

7.2.3 Tabellenstruktur

In jeder Tabelle wird beim Erstellen von neuen Datensätzen automatisch eine neue Id über die dazugehörende Sequence eingefügt. Die Id ist in jeder Tabelle primär. Manche der eingebauten Spalten werden in der Demo-Version nicht benützt, sollen trotzdem die Möglichkeit für weitere Funktionalität zeigen. Die Foreign Keys stehen am Schluss einer Beschreibung und heissen gleich wie in der referenzierten Tabelle.
subsubsectionUserTables

```
CREATE TABLE Usertable (
  usertableId    INTEGER PRIMARY KEY DEFAULT nextval('SUsertable'),
  name           VARCHAR(20),
  passwort       VARCHAR(20),
  letzterLogin   TIMESTAMP
);
```

In der Usertabelle werden alle neu registrierten Benutzer abgelegt. Die Id ist Primärattribut wird intern zur Referenzierung von Personen benötigt. In der Zwischentabelle "Teilnehmerliste"

wird die Verbindung zwischen der Person und der Umfrage verknüpft. Die Id wird bei der Webapplikation immer mitgegeben und bei sämtlichen Anfragen wird sie geprüft.
Der Name und das Passwort sind selbsterklärend. Das Passwort wird in der momentanen Form im Plaintext abgelegt.
Das Feld letzterLogin wird in der produktiven Umgebung noch nicht gebraucht. Sobald sich der User erfolgreich authentifiziert hat wird das aktuelle Datum in das Feld geschrieben. Damit könnte eine Funktion erstellt werden, die inaktiven Usern eine Mitteilung zukommen lässt, dass sie aktuelle Umfragen ausfüllen sollten. Nach einer bestimmten Zeit könnten die User automatisch gelöscht werden.

```
CREATE TABLE Student (
   studentId          INTEGER PRIMARY KEY DEFAULT nextval('SStudent'),
   martrikelNummer    INTEGER,
   userId             INTEGER
);
ALTER TABLE Student ADD CONSTRAINT fk_StudentUser
  FOREIGN KEY (userId) REFERENCES Usertable;

CREATE TABLE Dozent (
   dozentId           INTEGER PRIMARY KEY DEFAULT nextval('SDozent'),
   fachbereich        VARCHAR(30),
   userId             INTEGER
);
ALTER TABLE Dozent ADD CONSTRAINT fk_DozentUser
  FOREIGN KEY (userId) REFERENCES Usertable;

CREATE TABLE Ehemaliger (
   ehemaligerId       INTEGER PRIMARY KEY DEFAULT nextval('SEhemaliger'),
   abgangsgrund       VARCHAR(200),
   userId             INTEGER
);
ALTER TABLE Ehemaliger ADD CONSTRAINT fk_EhemaligerUser
  FOREIGN KEY (userId) REFERENCES Usertable;
```

Die User können bestimmten Rollen zugeordnet werden. Diese sind Student, Dozent und Ehemaliger und werden von Usertable vererbt. Im Demo-System ist dies nicht möglich, da in einer produktiven Umgebung die Rollen und auch die User meist aus einer anderen Datenbank oder LDAP Verzeichnis gemanagt werden können. Mit einem SQL Skript kann die Zuordnung erfolgen, hat aber keine Auswirkung auf das System. Die Foreign Keys in den vererbten Tabellen enthalten die userId. Die Vererbungsstruktur der Tabellen:

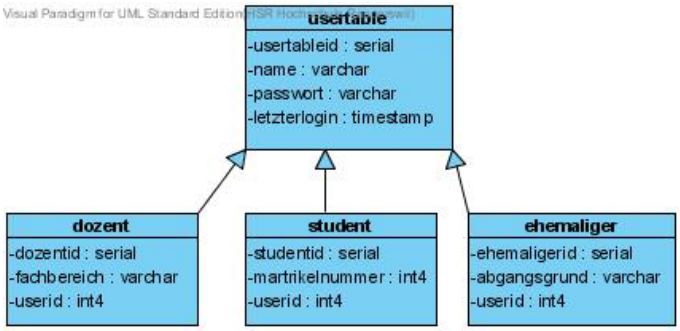


Abbildung 7.3: Datenbank Vererbungsstruktur der User

7.2.4 Survey

```
CREATE TABLE Survey (
  surveyId        INTEGER PRIMARY KEY DEFAULT nextval('SSurvey'),
  surveyTitel     VARCHAR(50),
  startdatum      TIMESTAMP,
  enddatum        TIMESTAMP,
  appletName      VARCHAR(30),
  publicP         VARCHAR(700),
  publicG         VARCHAR(700),
  publicY         VARCHAR(700),
  userIdDozent    INTEGER
);

ALTER TABLE Survey ADD CONSTRAINT fk_SurveyUserIdDozent
  FOREIGN KEY (userIdDozent) REFERENCES Usertable;
```

Die Tabelle Survey, Umfrage, ist ein zentrales Element des Aqfeedback Systems.
Im surveyTitel wird der Name einer Umfrage gespeichert. Die Länge ist auf 50 Zeichen begrenzt Die Start- und End-Daten sind für spätere Funktionalitäten gedacht. Die Umfrage könnte automatisch freigegeben werden wenn der Start Termin erreicht ist und automatisch geschlossen werden. Je nach Einstellung könnte dann ein Prozess gestartet werden, der das Ergebnis berechnet. Die Berechnung des Ergebnisses braucht bei einer Verschlüsselung von 2048 Bit relativ lange und müsste ohne Dozenten ausgelöst werden können.
Die Öffentlichen Zahlen P, G und Y werden als Text-Dump gespeichert. Die Java Big-Integer lassen sich sehr einfach in Strings konvertieren und in der Datenbank abspeichern.

7.2.5 Teilnehmerliste

```
CREATE TABLE TeilnehmerListe (
  teilnehmerListeId  INTEGER PRIMARY KEY DEFAULT nextval('STeilnehmerListe'),
  userIdStudent      INTEGER,
  surveyId           INTEGER
);
ALTER TABLE TeilnehmerListe ADD CONSTRAINT fk_TeilnehmerUserIdStudent
  FOREIGN KEY (userIdStudent) REFERENCES UserTable;
ALTER TABLE TeilnehmerListe ADD CONSTRAINT fk_TeilnehmerSurvey
  FOREIGN KEY (surveyId REFERENCES Survey;
```

Die Teilnehmerliste ist die Verknüpfung zwischen der Usertable und einer Umfrage. Wenn eine neue Umfrage erstellt wird, wird für jeden User ein Datensatz zur Umfrage hinzugefügt. Die Teilnehmerliste wird in einem späteren Zeitpunkt nicht mehr dynamisch erstellt. Der Dozent kann nach der Erstellung der Umfrage die Studenten einzeln in die Teilnehmerliste hinzufügen.

7.2.6 Question und Option

```
CREATE TABLE Question (
  questionId      INTEGER PRIMARY KEY DEFAULT nextval('SQuestion'),
  questionTitel   VARCHAR(100),
  surveyId        INTEGER
);
ALTER TABLE Question ADD CONSTRAINT fk_QuestionSurvey
  FOREIGN KEY (surveyId) REFERENCES Survey;
```

Jede Umfrage besitzt mindestens 1 Question, im Normalfall sind es aber mehrere. Die Zugehörigkeit zu einer Umfrage geschieht über das Feld surveyId. Der Question-Titel bestitzt eine Maximallänge von 100 Zeichen.

```
CREATE TABLE Option (
  optionId INTEGER PRIMARY KEY DEFAULT nextval('SOption'),
  stufe          INTEGER,
  m1             VARCHAR(700),
  m2             VARCHAR(700),
  anzahlJa       INTEGER,
  anzahlNein     INTEGER,
  questionId     INTEGER
);
ALTER TABLE Option ADD CONSTRAINT fk_OptionQuestion
  FOREIGN KEY (questionId) REFERENCES Question;
```

Jede Question hat 5 Optionen. Diese sind $--$, $-$, neutral, $+$ und $++$. Diese Angaben werden im Feld stufe als Zahl gespeichert.
Jede Option bekommt einen Ja und einen Nein- Wert, welche als m1 und m2 gespeichert werden. Diese Zahlen werden in der Java Applikation als BigInteger behandelt und in der Datenbank als Textfolgen abgespeichert.
Nach der Abstimmung schreibt der Zusammenzähler das Resultat der Anzahl Ja und Nein- Stimmen in die dafür vorgesehenen Felder.
Die Verknüpfung zur Question wird mit dem Constraint fk_OptionQuestion hergestellt.

7.2.7 Vote, Ballot und Value

```
CREATE TABLE Vote (
  voteId         INTEGER PRIMARY KEY DEFAULT nextval('SVote'),
  hash           VARCHAR(32),
  surveyId       INTEGER
);
ALTER TABLE Vote ADD CONSTRAINT fk_VoteSurvey
  FOREIGN KEY (surveyId) REFERENCES Survey;
```

Wenn der Student eine Umfrage ausgefüllt und seine verschlüsselten Daten versendet hat werden sie in der Tabelle Vote gespeichert.
Der Hashwert der generierten Vote wird zur Konsistenzprüfung gebraucht. Auf dem Bulletin Board kann jeder Student seinen Hashwert überprüfen mit den Werten auf seiner Quittung vergleichen. Diese Quittung bekommt er nach dem Absenden der Umfrage. Der Constraint verbindet eine ausgefüllte Vote mit der dazugehörenden Survey.

```
CREATE TABLE Ballot (
  ballotId       INTEGER PRIMARY KEY DEFAULT nextval('SBallot'),
  hash           VARCHAR(32),
  questionId     INTEGER,
  voteId         INTEGER
);
ALTER TABLE Ballot ADD CONSTRAINT fk_BallotVote
  FOREIGN KEY (voteId) REFERENCES Vote;
ALTER TABLE Ballot ADD CONSTRAINT fk_BallotQuestion
  FOREIGN KEY (questionId) REFERENCES question;
```

Jedes einzelne Kriterium bekommt eine Signatur. Diese wird wiederum bei der Aufnahme der gesamten Vote überprüft.
Es sind zwei Verknüpfungen auf andere Tabellen notwendig. Die Stimme muss wissen zu welcher Abstimmung sie gehört und welche Question damit verbunden ist.

```
CREATE TABLE Value (
  valueId          INTEGER PRIMARY KEY DEFAULT nextval('SValue'),
  wertX            VARCHAR(700),
  wertY            VARCHAR(700),
  optionId         INTEGER,
  ballotId         INTEGER
);
ALTER TABLE Value ADD CONSTRAINT fk_ValueBallot
  FOREIGN KEY (ballotId) REFERENCES Ballot;
ALTER TABLE Value ADD CONSTRAINT fk_ValueOption
  FOREIGN KEY (optionId) REFERENCES Option;
```

Jede Stimme hat genau gleich viele Values (Stimmwerte) wie ein Kriterium Optionen hat. Im entwickelten System sind dies fünf Values.
Jeder Value bekommt zwei Werte, nämlich den mit ElGamal verschlüsselten Stimmwert (X,Y).
Wiederum sind zwei Constraints vorhanden, der erste stellt die Verbindung zur Ballot her und der andere zur Option.

7.2.8 Sequencen

Da jede Tabelle eine Spalte mit dem Wert Id besitzt, wurde eine Sequence für jede erstellt. Die Namensgebung ist der Tabellenname und ein vorangestelltes "S". Der Increment ist auf 1 gestellt und beginnt je nach Tabelle mit einer anderen Zahl. Damit kann im Demo-System einfacher eine Assoziation von der Zahl zum dazugehörigen Datensatz gemacht werden. Beispielsweise gehört die Zahl 1002 automatisch zu einem User. Erst bei über 1000 Datensätzen pro Tabelle wird die Ordnung gestört.

Numerierung

Die Sequencen beginnen mit folgenden Startzahlen:

- SUsertable START WITH 1000
- SStudent START WITH 2000
- SDozent START WITH 3000
- SEhemaliger START With 4000
- SSurvey START WITH 5000
- STeilnehmerListe START WITH 6000
- SQuestion START WITH 7000
- SOption START WITH 8000
- SVote START WITH 9000
- SBallot START WITH 10000
- SValue START WITH 11000

7.2.9 Stored Procedures

Durch den Einsatz von Stores Procedures konnten die SQL- Statements in Java einfach gehalten werden. Die Kommunikation reduziert sich ebenfalls auf ein Minimum.
Die Sprache PL/pgSQL wurde gewählt und ist eine ladbare prozedurale Sprache für das PostgreSQL-Datenbanksystem. Der Vorteil gegenüber Standard SQL sind:

- Kontrollstrukturen zur Sprache SQL hinzufügt
- komplexe Berechnungen können ausgeführt werden
- alle benutzerdefinierten Typen, Funktionen und Operatoren stehen zur Verfügung vom Server wird der Code als TRUSTED eingestuft
- einfache Anwendung um Funktionen und Triggerprozeduren zu schreiben

Add-Procedures

Die Add-Procedures werden benötigt um die ID eines neu erstellten Datensatzes zu bekommen. Bei der Erstellung eines neuen Datensatzes wird die Id von der Sequence automatisch generiert. Auf der Java-Seite wird diese Id meist gebraucht um eine Objekt-Struktur aufzubauen und beispielsweise die Applets generieren zu können.
Folgende Procedures verhalten sich wie oben beschrieben.

- addSurvey
- addQuestion
- addOption
- addVote
- addBallot
- addValue

Die Procedures addVote wird stellvertretend erklärt.

```
CREATE OR REPLACE FUNCTION addVote(varchar, integer)
RETURNS INTEGER as \$VoteId\$
DECLARE ret integer;
begin
  INSERT INTO Vote( hash, surveyId)
  VALUES (\$1, \$2);

  SELECT voteid INTO ret FROM Vote WHERE hash= \$1;
return ret;
end; \$VoteId\$ language plpgsql;
```

Der Name der Funktion wird aus dem Wort "add" und dem Tabellennamen zusammengesetzt. Als Parameter werden die Variablentypen angegeben. bei dieser Funktion sind es zwei Werte, der erste ist der Hash der Vote, der zweite die Survey-Id.
Der neue Datensatz wird in die Datenbank gespeichert. Die neu erstellte Id wird mit einem Select Statement ausgelesen, in den Wert $VoteId$ kopiert und an das Java-Programm zurückgegeben.

setAppletName-Procedure

Die letzte Stored Procedure setzt den Namen eines Applets.

```
CREATE OR REPLACE FUNCTION setAppletName(integer,varchar)
RETURNS INTEGER as \SurId\
DECLARE ret integer;
begin
  UPDATE SURVEY SET appletname=\$2 WHERE surveyId = \$1;
return \$1;
end; \$SurId\$ language plpgsql;
```

7.2.10 Datenbankzugriff

Der Datenbankzugriff geschieht über die JDBC-Schnittstelle von PostgreSQL. Jedes Projekt, das die Datenbank benötigt, hat ein Subpaket mit dem Namen db. Die Darin enthaltene Klasse DbAqfeedback verbindet das jeweilige Projekt mit der Datenbank. Der Aufbau ist bei allen Projekten sehr ähnlich.

Einen kleinen Unterschied gibt es bei dem Datenbankobjekt des Web-Servers. Damit die Initialisierungsvariablen einfach übergeben werden können, wurde ein spezielles Servlet erstellt, das die Parameter aus der Configuration der Webapplikation, der Web.xml liest. Die Klasse wird mit den Werten aus dem AqfInit Servlet initialisiert. Dies sind die Variablen driver, database, user und password. In den anderen Projekten werden diese Daten in Properties-Dateien geschrieben.

Schematischer Aufbau

Ein Programm möchte auf der Datenbank eine Abfrage ausführen. Das folgende Schema zeigt den Aufbau.

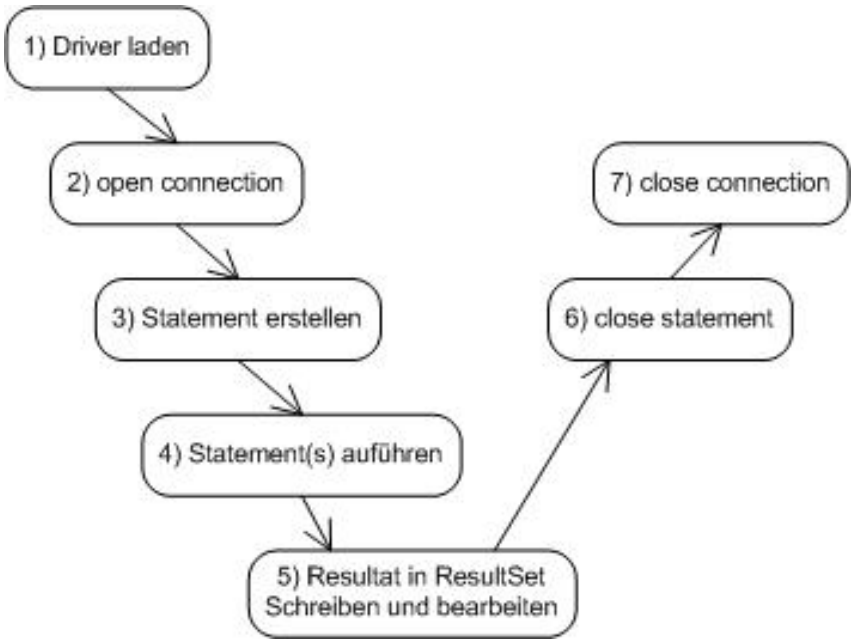


Abbildung 7.4: Datenbank Ablauf eines Zugriffs

1. Der JDBC- Treiber von PostgreSQL wird geladen.

2. Die Verbindung zur Datenbank wird mit einem Connection Objekt hergestellt.
3. Es wird mindestens ein Statement für den Zugriff erstellt. CallableStatements werden für Stored Procedure und PreparedStatements für "vor-kompilierten Statements" gebraucht.
4. Das Statement wird ausgeführt. Die DML Anweisung werden durch PreparedStatements ausgeführt.
5. Das Ergebnis wird in ein ResultSet geschrieben. Bei einer Stored Procedure wird in der Demo meist nur ein Integer zurückgegeben. Dieser wird nicht in ein ResultSet geschrieben, sondern als Zahl behandelt.
6. Das Statement wird geschlossen
7. Die Verbindung wird geschlossen.

Während der Elaboration- und Testphase wurden zwei verschiedene Möglichkeit für das Öffnen der Datenverbindung behandelt:

1. Die Datenbankverbindung wird geöffnet, sobald ein das DbAqfeedback erstellt wird und im Finalizer wieder geschlossen. Der Vorteil ist, dass die Verbindung nur ein mal geöffnet werden muss und keine Zeit verloren geht. Der Nachteil ist, dass die Verbindung offen ist auch wenn dies nicht erforderlich ist.
2. Die Datenbankverbindung wird bei Gebrauch geöffnet. Der Vorteil an dieser Lösung ist, dass die Verbindung nur aufgebaut wird, wenn dies notwendig ist. Der negative Punkt ist, dass das öffnen und schliessen Zeit braucht.

Im Verlauf entschieden wir uns für Variante zwei. Die benötigte Zeit für das Öffnen und Schliessen einer Verbindung ist unter einer Sekunde. Ab dem zweiten Durchgang ist die Performance noch höher. Zudem wollten wir nicht, dass auf dem Webserver die Verbindung immer steht. Ein Beispiel aus der Realität soll dies untermauern. Der Mensch öffnet den Kühlschrank nur, wenn er etwas hineinstellen oder rausholen möchte und schliesst ihn nachher. Es gibt überhaupt keinen Grund, warum der Kühlschrank ständig offen sein soll. Zudem verbraucht er dann mehr Strom.

Das folgende Codebeispiel zeigt den oben erwähnten Ablauf ohne Exception Handling

```
Class.forName("org.postgresql.Driver" );   // 1
Connection con = DriverManager.getConnection(
  "jdbc:postgresql://127.0.0.1:5432/aqfeedback", "user", "password"); // 2
PreparedStatement stmt = con.prepareStatement("Select * from Ballot"); // 3
ResultSet rs = stmt.executeQuery(); // 4, 5
stmt.close(); // 6
con.close(); // 7
```

7.3 Generierung der Authorities

Der Konstruktor der Klasse Authority empfängt folgende Werte:

- ID - Zur Identifikation und Bestimmung der Stelle in der Gruppe
- Anzahl Authorities - Wird benötigt zur Generierung der Shared Secrets
- Schwellwert - Wird benötigt zur Generierung der Shared Secrets
- Initialwerte - Die Liste der Intialwerte zur Generierung des Polynoms
- ElGamal Parameter - Werden benötigt zur Generierung von PublicKey und Shared Secrets

Das Skript erstellt für jede Authority eine Liste mit zufälligen Initialwerten und übergibt diese dem Konstruktor.

```
private void initAuthorities() {
  SecureRandom r = new SecureRandom();
  authorities = new LinkedList<Authority>();
  List<List<BigInteger>> a1 = new LinkedList<List<BigInteger>>();

  //Initialwerte werden generiert
  for(int i = 0; i<N; i++){
    List<BigInteger> tempList = new LinkedList<BigInteger>();
    for(int j = 0; j < T; j++){
      tempList.add(j,new BigInteger(Q.bitLength(),r));
    }
    a1.add(tempList);
  }
  //Authorities werden initialisiert
  for(int i = 0; i<N; i++){
    Authority temp = new Authority(i,"Auth"+i+1,T,N,a1.get(i),P,Q,G);
    authorities.add(temp);
  }
}
```

Die Authorites und der Öffentliche Schlüssel werden Serialisiert und in separate Dateien geschrieben.

```
private void calcPublicKey() {
  BigInteger publicKey= BigInteger.ONE;
  //Der Public Key wird aus den verschiedenen Authorities
  //ausgelesen und zusammengezählt
  for(Authority auth: authorities){
    publicKey = publicKey.multiply(auth.getPublicKey());
    publicKey = publicKey.mod(P);
  }
  //ElGamalPublicKey-Objekt wird anhand der Parameter und des
  //Public Keys erstellt
  ElGamalPublicKey pubKey = new ElGamalPublicKey(P,2,publicKey,Q);

  //Objekt wird serialisiert und ausgeschrieben
  PublicKeyFileWriter pkWriter = new PublicKeyFileWriter();
  pkWriter.writeKey(pubKey);
}
```

Für die Generierung der Shared Secrets lässt sich aus jeder Authority die Funktion getPolynomial(int i) aufrufen, welche den Stützwert an der Position i zurückgibt. Für jede Authority, wird der entsprechende Stützpunkt aus allen Authority geladen und gespeichert.

```
private void calcSharedSecrets() {
  sharedSecrets = new LinkedList<BigInteger>();

  //calc shared keys
  for(int i = 0; i< N; i++){
    for(int j = 0; j< N; j++){
      authorities.get(i).setSharedSecret( authorities.get(i).getSharedSecret().add(authorit
    }
    authorities.get(i).setSharedSecret( authorities.get(i).
      getSharedSecret().mod(Q));
```

```
    sharedSecrets.add(i,authorities.get(i).getSharedSecret());
  }
}
```

7.4 Generierung einer Umfrage

Die Umfrage-Generierung geschieht über ein Script. Als erstes muss der serialisierte PublicKey geladen werden. Der Umfrage-Titel und der Umfrage-Initiator müssen angegeben werden. Danach können beliebig viele Kriterien hinzugefügt werden. Für jedes Kriterium werden zufällig die Stimmwerte m und deren inversen Werte $\frac{1}{m}$ bestimmt.

```
private static void addQuestion() {
        //Lädt den Text des Kriteriums
  String response = JOptionPane.showInputDialog(null, "Enter criteria");

  SecureRandom r = new SecureRandom();
  List<Option> optionList = new LinkedList<Option>();

  //Es werden Stimmwerte für die Optionen generiert
  for(int i = 0; i < numOfOptions; i++){

    int bitLength = publicKey.getModulus().bitLength();
    Option o;
    BigInteger m1 = new BigInteger(bitLength,r);
    while(m1.compareTo(publicKey.getModulus()/*-2*/)>0
          || m1.equals(BigInteger.ZERO)){
      m1 = new BigInteger(bitLength,r);
    }
    try{
      BigInteger m2 = m1.modInverse(publicKey.getModulus());
      o = new Option(m1,m2,i);
      optionList.add(o);
    }
  }
  Question q = new Question(++numOfQuestions,response,optionList);
  survey.addQuestion(q);
}
```

Der Aufbau der Umfrage besteht aus 3 Klassen:

- Survey - Repräsentiert die Umfrage und besteht aus einer Menge von Question-Objekten.
- Question - Repräsentiert ein Kriterium und besteht aus einer Menge von Option-Objekten.
- Option - Repräsentiert eine auswählbare Option. Beinhaltet die Werte m und $\frac{1}{m}$

Um die Generierung abzuschliessen werden die Werte in die Datenbank geschrieben und es wird eine INI-Datei generiert welche ebenfalls die Umfrage-Daten beinhaltet. Mit dieser INI-Datei wird ein Java-Applet generiert und in einem Verzeichnis des Webservers abgelegt. Das Applet wird ausserdem signiert, was dem Benutzer dessen Herkunft und Integrität garantieren soll.

7.5 Durchführung einer Abstimmung

7.5.1 Webserver - Webapplikation aqfeedback

Der Webserver wurde als dynamisches Web Projekt in Java erstellt. Die einzelnen Pageinhalte werden mit der Servlet Technologie von Java erstellt. In der Elaborationphase haben wir die neuere JSP Technik evaluiert. Wir entschieden uns dennoch für die Servlets, da die Kommunikation aus einem Applet sehr einfach zu implementieren ist.
Aufbau der Applikation:

- aqfeedback
  - META-INF
    - MANIFEST.MF
  - umfragen
    - diverseApplets.jar
  - WEB-INF
    - classes
      - ch/hsr/aqfeedback/server
      - ch/hsr/aqfeedback/server/db
      - ch/hsr/aqfeedback/servlets
      - ch/hsr/aqfeedback/server/klassen nach Paket geordnet.class
      - extLib/log4j-1.2.14.jar
      - extLib/postgresql-8.2-505.jdbc3.jar
      - Log4J.properties
    - web.xml
  - index.html

Wenn der Webserver aus Eclipse gestartet wird baut er ein anderes Webapps-Verzeichnis auf als auf dem normalen Tomcat Server.
Das Tomcat Verzeichnis ist im server.xml des Tomcats angegeben. Im Default befindet es sich im Ordner /Apache Software Foundation/tomcat 5.5/webapps.

In der Eclipse Umgebung befindet sich das Verzeichnis für die Webapplikationen in
$/{Workspace}/.metadata/.plugins/org.eclipse.wst.server.core/tmp0/

Nun Definieren wir für Eclipse.
${catalina.base} = /${Workspace}/${Project-Space}/.metadata/.plugins/org.eclipse.wst.server.core/tmp0
Für Tomcat
${cataina.base} = /${tomcatinstallationsorder}

Folgende Dateien des Projektes müssen in die Pseudo-Struktur von Eclipse oder TomCat Installation eingebunden werden.
Diese Werte können geändert werden, müssen in einem solchen Fall in den jeweiligen Initial-Dateien (web.xml,*.properties) angepasst werden.

SSL/TLS Certificate Keystore
${catalina.base}/.keystore
Infos zur Installation von Tomcat SSL [T06] sind in den zusätzlichen Referenzen beschrieben.
Wenn der Tomcat als Windows Service installiert worden ist befindet sich im Default die .keystore Datei unter Windwo im Pfad /Dokumenten und Einstellungen/LocalService/.

Log4J properties File
${catalina.base}/webapps/aqfeedback/WEB-INF/classes/Log4J.properties

Die LogDatei wird in das folgende Verzeichnis gespeichert.
${catalina.base}/logs/AqfeedbackLog.log

Die benötigten Jar-Archive für PostgreSQL und Log4j sind in folgendem Verzeichnis zu speichern.
${catalina.base}/webapps/aqfeedback/WEB-INF/classes/extLib

Umfrage JAR's Die Umfragen befind
en sich im Context Path der Webapplikation im Ordner /umfragen/.
${req.getContextPath()}/umfragen/

7.5.2 Webserver - Beans

JavaBeans sind Software-Komponenten für die Programmiersprache Java. Sie werden in der Softwareentwicklung als Container zur Datenübertragung verwendet. JavaBeans entwickelten sich aus der Notwendigkeit heraus, GUI-Klassen einfach instanzieren und übertragen zu können.
In der Webapplikation benötigen wir diverse Beans.

ServerException

Bestimmte Server Exceptions werden von dieser Klasse aufgefangen, welche von Exception abgeleitet ist.

Surveyentry

Dies ist das zentrale Bean für eine Umfrage. Während der Client Session wird es für den Aufbau der Umfragen gebraucht, für die der Anwender eine Berechtigung besitzt.
Es werden folgende privaten Variablen festgelegt, auf die mit öffentlichen Methoden zugegriffen werden kann:

- String umfrageTitel: Umfragetitel einer Survey
- Timestamp start: Das Startdatum einer Survey
- Timestamp ende: Das Enddatum einer Survey
- String appletName: Der Dateiname eines damit verbunden Applets

UserEntry

Dieses J2EE Bean stellt eine UserEntry auf dem Webserver dar. Nicht alle Variablen können frei gesetzt werden, manche können nur bei der Erstellung des Objektes gesetzt werden. Die Benutzergruppe ist ein enum-Wert mit den Inhalten für keine Zugehörigkeit, Student, Dozent oder Ehemaliger. Während der ganzen Session wird auf das Objekt zugegriffen und es stellt das zentrale Sicherheitsmodul dar. Die Variablen sind:

- int userId: Die UserId des Benutzers. Kann gesetzt werden wenn er vom System authentifiziert worden ist.
- String name: Der UserName des Benutzers.
- int loginFailure: Die Anzahl der erfolglosen Login-Versuche.

- Benutzergruppe benutzergruppe: Zuordnung zur Gruppe Keine, Student, Dozent, Ehemaliger;

UserException

Bestimmte User-Exceptions werden von dieser Klasse aufgefangen. Sie ist wiederum von Exception abgeleitet.

7.5.3 Webserver - Servlet Struktur

Jedes Servlet ist von javax.servlet.http.HttpServlet abgeleitet. Je nach Einsatzgebiet wurde die Post oder die Get Methode implementiert.

Log4j und AqfInit

Die Servlets Log4j und AqfInit sind Init-Servlets. Mit dem ersten können höchst flexibel die Logging Eigenschaften von Log4j gesetzt werden. In der web.xml wird der Dateiname des Properties Files angegeben. Das Applet wird automatisch beim Start des Servers geladen. Es kann nicht vom Benutzer angesprochen werden.
Das zweite Initalisierungsapplet AqfInit besitzt die Variablen für die Datenbank. Aus dem web.xml werden die Zugangsdaten zur PostgreSql Datenbank gelesen. Die Daten können gelesen werden aus jedem weiteren Applet. Des Weiteren wird der Pfad zu den einzelnen Umfrage Dateien gesetzt. Der Benutzer hat keine Möglichkeit, diese Variablen autonom lesen zu können.

UserSession Servlets

Die Servlets ServerLogin, ServerLogut, ServerRegister, ServerSurvey, ServerSurveySelection, ServerUserAdmin stellen den HTML- Content für eine Benutzer Session zur Verfügung. Die Servlets haben je nach Gebrauch eine /Get oder /Post Methode. Das Zustandsdiagramm zeigt die verschiedenen Verknüpfungen der Servlets.

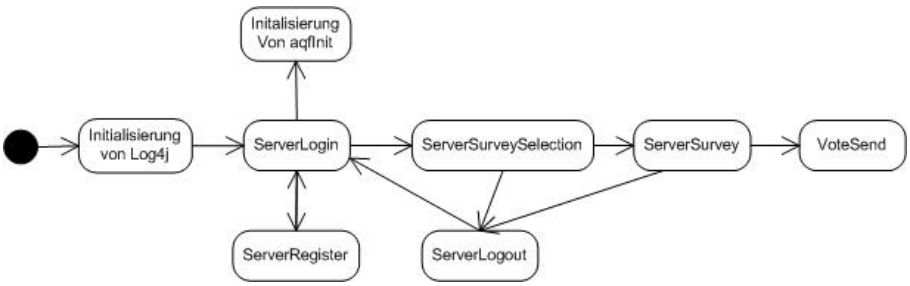


Abbildung 7.5: Servlets: Zustandsdiagramm

ServerLogin Stellt die Umgebung für den Benutzerlogin dar.

ServerLogut Der angemeldete Benutzer kann sich hier abmelden.

ServerRegister Ein neuer Benutzer findet eine Oberfläche für die Registrierung.

ServerSurveySelection Stellt die Tabelle der Umfragen dar, für die der Benutzer die Rechte hat.

ServerSurvey Ein Applet wird in dieser Maske eingebunden. Ist zentrales Element für das Ausfüllen der Umfrage. Das Applet kommuniziert mit dem Servlet VoteSend.

ServerUserAdmin Hier kann die Benutzergruppenzuteile angesehen und geändert werden.

VoteSend Die verschlüsselten Daten werden vom Applet an das VoteSend Servlet gesendet. Die Daten werden verifiziert und in die Datenbank gespeichert.

Für die ausführliche Dokumenten der Servlet Klassen wird auf die Java Dokumentation und den begelegten Source Code verwiesen.

7.5.4 Applet

Das Applet lädt das INI-File und rekonstruiert die Objektstruktur mit den Klassen Survey, Question und Option. Anhand dieser Struktur generiert das Applet das Eingabeformular, über welches der Benutzer seine Wahl eingeben kann. Beim Klick auf den Button "Senden" beginnt das Applet die Stimmwerte zu verschlüsseln. Die Verschlüsselten Daten werden analog der Umfrage in einer eigenen Objetkstruktur abgelegt.

```
//Die einzelnen Kriterien werden durchschritten
for(ButtonGroup bGroup: buttonGroups){
  Ballot ballot = null;
  Enumeration enumer = bGroup.getElements();

  //Für jeden RadioButton eines Kriteriums
  while (enumer.hasMoreElements()) {
    Value val = null;
    JRadioButton rb_tmp = (JRadioButton) enumer.nextElement();

    //Name beinhaltet QuestionID und Stelle in der ButtonGroup
    String[] selections = rb_tmp.getName().split("#");

    if(ballot == null){
      ballot = new Ballot(Integer.parseInt(selections[0]));
    }

    //Stimmwert wird geladen
    BigInteger number = survey.getNumber(selections[0],
        selections[1],
        rb_tmp.isSelected());
    ElGamalEncryptor cryptor = new ElGamalEncryptor(settings);
    val = cryptor.encrypt(number);
    val.setStufe(Integer.parseInt(selections[1]));
    ballot.add(val);
  }
  vote.addBallot(ballot);
}
```

- Vote - Repräsentiert eine Stimmabgabe und besteht aus einer Menge von Ballot-Objekten.
- Ballot - Repräsentiert die verschlüsselten Stimmwerte eines Kriteriums und besteht aus Value-Objekten.
- Value - Repräsentiert einen verschlüsselten Stimmwert

Nach der Generierung der Objektstruktur werden über alle Ballot-Objekte und über das gesammte Vote-Objekt Hashes generiert. Das Vote-Objekt wird serialisiert und zusammen mit dessen Hash an ein Servlet auf dem Webserver übertragen. Auf dem Server wird erneut ein Hash generiert und mit dem mitgesandten Hash verglichen um zu überprüfen ob ein Übertragungsfehler oder eine

Manipuation stattgefunden hat. Wenn kein Fehler vorliegt werden die Objekte in die Datenbank geschrieben.
Das Applet generiert eine Quittung und zeigt diese dem Benutzer in einem neuen Fenster. Der Benutzer hat die Möglichkeit diese zu kopieren und zu speichern um diese später mit den auf dem Server gespeicherten Daten zu vergleichen.
Anmerkung: In unserer Demo-Version ist einem Benutzer möglich mehrere Stimmen abzugeben. Eine entsprechende Überprüfung findet nicht statt.

7.6 Auszählung einer Abstimmung

Für die Auszählung einer Umfrage werden 3 Dinge benötigt:

- ElGamal Parameter - Werden aus der PublicKey-Datei geladen
- Authorities - Es werden genügend Authorities benötigt, so dass der Schwellwert erfüllt ist
- Daten - Werden aus der Datenbank ausgelesen

In unserem Skript werden zuerst der PublicKey und die benötigten Authorities geladen. Der Benutzer kann eine beliebige Menge von Authorities laden sofern der Schwellwert erreicht oder überschritten ist. Da die Authorities in beliebiger Reihenfolge geladen werden kann, muss sie vom Skript noch geordnet werden. Für jede Option wird die Funktion countOption() aufgerufen in welcher das Resultat berechnet wird:

```
public void countOption(Option option){
  BigInteger m1 = option.getM1();
  BigInteger m2 = option.getM2();

  //Berechne set
  //Bitliste um zu erkennen welche Authorities
  //an der Auszählung teilnehmen
  int set = 0;
  for(Authority auth: authorities){
    if(auth!=null){
      set += java.lang.Math.pow(2.0, auth.getId());
    }
  }
  System.out.println("set:"+set);

  //Shared Secrets werden geladen
  List<BigInteger> sharedSecrets = new LinkedList<BigInteger>();
  for(Authority auth: authorities){
    sharedSecrets.add(auth.getSharedSecret());
  }

  List<BigInteger> interpolationList = new Vector<BigInteger>(N);
  BigInteger result= BigInteger.ZERO;

  //Interpolationsliste wird berechnet und
  //mit den SharedKeys verrechnet
  interpolationList = lagrange(set,q);
  scalar(result,interpolationList,sharedSecrets);

  //Stimmen werden geladen und im Attribut xTot und yTot gespeichert
  loadVotes(option);

  //Calculate Broadcast
```

```
  System.out.print("Broadcast:     ");
  List<BigInteger> broadcast = new LinkedList<BigInteger>();
  //Each Authority calculates Broadcast W_i = xTot^s_i
  for(Authority auth: authorities){
    broadcast.add(auth.getCalc(xTot));
  }

  System.out.println();
  BigInteger exp = exponential(interpolationList, broadcast);
  result = yTot.multiply(exp);
  result = result.mod(p);

  //Anzahl Ja-Stimmen werden berechnet
  float res = checkResult(result,m1).floatValue();
  float numberOfVotes = numOfVotes.floatValue();
  float endRes = numberOfVotes/2 + res/2;

  option.setAnzahlStimmen((int)endRes);
}
```

Die Berechnung von T aus m^T geschieht in der Funktion checkResult(). Es werden zwei IndexVariabeln definiert. Die Variable posIndex testet die positiven Werte, die Variable negIndex testet die negativen Werte. Sobald der korrekte Wert gefunden wurde wird das Testen abgebrochen und das Resultat zurückgegeben.

```
private BigInteger checkResult(BigInteger result, BigInteger m1) {

  boolean resultFound = false;
  BigInteger posIndex = BigInteger.ONE;
  BigInteger negIndex = BigInteger.ZERO;
  BigInteger endResult = BigInteger.ZERO;
  while(!resultFound || posIndex > numOfQuestions){
    //Suche in positivier Richtung
    endResult = m1.modPow(posIndex, p);
    if(endResult.equals(result)){
      return posIndex;
    }

    //Suche in Negativer Richtung
    endResult = m1.modPow(negIndex, p);
    if(endResult.equals(result)){
      return negIndex;
    }

    posIndex = posIndex.add(BigInteger.ONE);
    negIndex = negIndex.subtract(BigInteger.ONE);
  }
  return null;
}
```

Die Resultate werden in die Objekt-Struktur geschrieben und können herausgelesen werden.

Kapitel 8

Use Cases

8.1 Übersicht

Unsere Applikation beschränkt sich im Grunde genommen auf drei Dienste: Die Generierung und das Bereitstellen von Umfragen für die Unterrichtsbeurteilung, das Anzeigen des Umfrage-GUI und die Übermittlung der Wahl an den Server, sowie zu guter letzt die Auszählung der Stimmen.

Zu diesen drei Diensten kommen noch die Authentisierung sowie die mögliche Ansicht der abgegebenen Stimmen auf dem Server. Daraus resultieren folgende fünf Use-Cases:

- UC01: Login des Benutzers
- UC02: Umfrage einrichten
- UC03: Umfrage ausführen (ausfüllen und übertragen)
- UC04: Bulletin Board einsehen
- UC05: Auswertung der Umfrage auslösen
- UC06: Auswertung der Umfrage einsehen

8.2 Use Cases

8.2.1 UC01: Login des Benutzers

User Actions	System Responses
Aufrufen der Website	
	Anzeigen des Anmeldeformulars
Eingabe von Benutzernamen und Passwort	
	Verifizierung der Eingabe
	Anzeigen der Verfügbaren Umfragen

8.2.2 UC02: Umfrage einrichten

User Actions	System Responses
Auswahl Umfrage einrichten	
	Anzeigen des Einrichtungs-Formular
Ausfüllen des Formulars	
	Verifizieren der Eingabe
	Generierung des Applets
	Anzeigen der Umfragen

8.2.3 UC03: Umfrage ausführen (ausfüllen und übertragen)

User Actions	System Responses
Auswählen der auszufüllenden Umfrage	
	Server: Bereitstellen des Applets
	Client: Ausführen des Applets Client: Anzeigen der Umfrage
Ausfüllen der Umfrage	
	Client: Verschlüsselung der Umfrage Client: Generierung der Signatur Client: Übertragung an Server
	Server: Entgegennahme der Daten Server: Speichern der Stimme Server: Markieren Benutzers in der Liste Server: Anzeigen der Umfrage

8.2.4 UC04: Bulletin Board einsehen

User Actions	System Responses
Auswählen der anzuzeigenden Umfrage	
	Anzeige abgegebenen Stimmen

8.2.5 UC05: Auswertung der Umfrage auslösen

User Actions	System Responses
Auswählen der auszuwertenden Umfrage	

Kapitel 9

Schlussfolgerung

9.1 Was haben wir erreicht

Wir haben uns in die Materie des E-Voting, besonders in das Paper von Gennaro, Cramer und Schoenmaker eingearbeitet. Der erstellte Prototyp ermöglicht es uns eine Wahl nach [CGS97] mit folgenden Eigenschaften durchzuführen.

- Initialisierung von Authorities mit Secret Sharing
- Durchführung einer Wahl mit verschlüsselten Werten
- Auszählung der Wahl

Zusätzlich haben wir als Infrastruktur:

- Webserver und Servlets für Authentisierung und Authorisierung der Wähler
- Applet als Abstimmungsclient
- Skript für die Erstellung einer neuen Umfrage

Die Infrastruktur ermöglicht die beliebige Erstellung von Benutzern und Umfragen. Der Prototyp eignet sich um das Schema des Papers zu verstehen und nachzuspielen.

9.2 Was haben wir nicht erreicht

Leider haben wir eigene Aspekte des Papers und der generellen Anforderungen nicht realisieren können:

- Verifizierung der Stimme durch Zero Knowledge Proof
- Einsicht in die abgegebenen Stimmen auf dem Bulletin Board
- User Management: Zuordnung von Umfrage zu User und User - UserRolle

Das Bulletin Board liesse sich mit relativ wenig Aufwand realisieren, da die Daten in der Datenbank gespeichert sind und nur in angemessener Form in einem Servlet geladen werden müssten.

9.3 Ausblick

In das System kann beispielsweise die Authentisierung der Studierenden mit einem Public Key Verfahren integriert werden. Ein Zero Knowledge Verfahren und blinde Signaturen können eingebaut werden. Des Weiteren können Funktionen zum Bulletin Board entwickelt werden, das für die Speicherung der Abstimmungen zuständig ist. Sämtliche Funktionen können in die Webplattform integriert werden und die ausgetauschten Meldungen müssen überprüft werden. Die Usability und die grafische Gestaltung ist optimierbar.

In einer allfälligen Diplomarbeit werden wir uns diesen interessanten Punkten widmen.

Kapitel 10

Projektmanagement

10.1 Übersicht

10.1.1 Zweck und Ziel

Wie in der Aufgabenstellung beschrieben, wird unser Team eine Übersicht über die bestehenden Verfahren erarbeiten und anschliessend eine Demoversion einer web-basierten anonymen Bewertung mit persönlicher Bestätigung erstellt und getestet werden.
Dieser Projektplan ist über die ganze Projektdauer gültig, wird bei Änderungen aber angepasst und auf die Ziele ausgerichtet.

Die Erarbeitung der Thematik um digitale Wahlen beinhaltet folgende Punkte.

- Grundlagen über die Verschlüsselungstheorien
- Verfahren zur Anonymisierung von elektronischen Wahlen und Abstimmungen
- Hintergrundinformationen über produktive Systeme

Bei der Demoversion kann folgende Funktionalität eingebaut werden, wobei die Priorität von oben nach unten abnimmt. Die genannte Funktionalität stellt einen ersten Grob-Entwurf der Anforderungs-Spezifikationen dar und dient diesbezüglich als Grundlage.

- Web-basierte Applikation
- Erstellung von Umfragen
- Auswertung einer Umfrage
- Einsehen von Auswertungen
- Speicherung der Daten in einer Datenbank
- Verschlüsselung der Daten
- Sichere Kommunikation
- Bulletin Board ansehen

10.1.2 Annahmen und Einschränkungen

Wir gehen von einem Aufwand von 16 Stunden pro Woche und Team-Mitglied aus. Daraus resultiert ein Gesamt-Aufwand von ca. 450 Stunden während 14 Wochen. Aufgrund der Planung sollte

dieser Aufwand ausreichend sein, sodass am Schluss eine lauffähige Version vorliegt.
Die Abweichungen von dieser Planung sollten maximal 4 Stunden pro Mitglied und Woche sein. Diese Abweichungen sollten sich über das gesamte Projekt mitteln.

10.2 Projekt Organisation

Das Team besteht aus 2 Mitgliedern und ist demokratisch organisiert. Bei Unstimmigkeiten wird ein gemeinsamer Konsens gesucht.
Es sind regelmässige Besprechungen geplant, in denen auch die Beratungen und Reviews stattfinden. Zusätzlich werden laufend Meetings vereinbart und jedes Team-Mitglied erarbeitet ein gewisses Pensum selbständig. Zur Kommunikation ausserhalb der Meetings stehen die bereits von allen Team-Mitgliedern verwendeten Kommunikationsmittel wie E-Mail, Telefon, MSN zur Verfügung.

10.3 Management Abläufe

10.3.1 Projektplan

Eine detaillierte Aufstellung über die Planung wird im separaten Dokument Projektplan.xls gezeigt. Dieses Dokument enthält die Aufteilung in Phasen, die Arbeitsbereiche und Themengebiete im Projekt und die personelle Planung.

10.3.2 Iterationsplanung

Aus dem separaten Dokument Projektplan.xls sind folgende Phasen und Iterationen ersichtlich

- 02.04 - 15.04 Inception
- 16.04 - 29.04 Elaboration Iteration 1
- 30.04 - 13.05 Elaboration Iteration 2
- 14.05 - 03.06 Construction Iteration 1
- 04.06 - 24.06 Construction Iteration 2
- 25.06 - 01.07 Transition
- 02.07 - 08.07 Reserve

Die Meilenstein-Termine sind jeweils am Freitag von einer abzuschliessenden Iteration.
Die Besprechungen mit Herrn Dr. Steffen werden am Mittwoch morgen von 8 bis 9 Uhr abgehalten. Bei Terminkollisionen werden Ausweichmöglichkeiten gesucht.

10.4 Risiko Management

10.4.1 Management Risiken

Krankheit, Unfall, ausserordentliche Verpflichtungen oder andere nicht vorhersehbare Zwischenfälle können zu Ausfällen führen, die nicht eingeplant sind. Diese Fälle müssen unmittelbar nach auftreten dem anderen Team-Mitglied kommuniziert werden, damit dieser die notwenidgen Massnahmen ergreifen kann. Dies sind primär

- Übernehmen einzelner Tätigkeiten für die Ausfallzeit

- Anpassen des Zeitplans
- Anpassen der geplanten Ziele, wo notwendig.
- Kontaktaufnahme mit dem Betreuer, falls durch teaminterne Massnahmen nicht lösbar.

Das Risiko fehlerhafter Planung schätzen wir relativ hoch ein. Bedingt dadurch, dass viele Details bei Projektbeginn noch nicht bekannt sind, rechnen wir damit, dass sich in der Planung noch diverse Verschiebungen ergeben können.
Sind geplante Features nicht in der vorgesehenen Zeit fertig zu stellen, müssen die Anforderungen entsprechend zurückgestuft werden.

Die Meilenstein- und Iterationsplanung soll nicht mehr geändert werden.

10.4.2 Technische Risiken

Es besteht das Risiko, dass Laufzeitprobleme entstehen, wenn die notwendigen Algorithmen zuviel Laufzeit in Anspruch nehmen. Sämtliche Algorithmen müssen ihre Arbeit möglichst unabhängig und autonom erledigen. Der Benutzer darf davon nicht aufgehalten werden.

Es ist denkbar, dass Probleme im Zusammenhang mit verwendeten Technologien entstehen. Dazu zählen vor allem Technologien aus der Kryptographie. Um dieses Risiko zu minimieren stehen verschiedene Informations-Ressourcen zur Verfügung.

Das Risiko von Ausfällen von Servern, Laptops oder anderen technischen Einrichtungen, die für das Projekt relevant sind können insofern minimiert werden, da alle Team-Mitglieder jeweils aktuelle Kopien der Sourcen und der Dokumente auf ihren persönlichen Laptops halten und somit immer sämtliche Daten zur Verfügung haben.

Datenverluste sind durch die gegebene Redundanz vermieden. Die Programm-Sourcen befinden sich auf einem SVN-Server und werden laufend auf die persönlichen Laptops abgeglichen. Mit allen weiteren Dokumenten (administrative Dokumente) wird gleich verfahren.

10.5 Infrastruktur

Die Demo-Version wird in Java programmiert. Die dazu notwenige Infrastruktur setzt sich wie folgt zusammen:

- Persönliche Notebooks der Team-Mitglieder und Desktops im Studienarbeitszimmer
- Entwicklungsumgebung Eclipse WTP (web tools platform)
- Apache Tomcat für den Webserver
- Datenbank Server (Läuft auf dem gleichen Rechner)
- Subversion-Projekt zur Verwaltung der Sourcen und Dokumente
- Photoshop zur Erstellung der Grafiken

Sämtliche Infrastruktur und Software steht den Team-Mitgliedern bereits zur Verfügung. Die Team-Mitglieder sind dadurch unabhängig von Lokalitäten (HSR) und installierter Software. Durch die Verwendung von Subversion ist ein permanenter Abgleich gewährleistet. Der Subversions-Server befindet sich nicht im Netz der HSR.

10.6 Qualitätsmassnahmen

Folgende Massnahmen sollen zur Gewährleistung guter Qualität beitragen:

Regelmässige Team-Meetings
Zum Austausch und gegenseitiger Abstimmung sind die regelmässigen Team-Meetings für beide Team-Mitglieder obligatorisch.

Dokumenten-Pflege
Sämtliche Dokumente werden kontinuierlich auf aktuellem Stand gehalten.

Komplikationen kommunizieren
Komplikationen jeglicher Art werden immer unmittelbar sämtlichen Team-Mitgliedern kommuniziert.

Codesmell
Wir entwickeln einen guten Codesmell. Durch Refactoring, Dokumentation und weitere Massnahmen sind die Codes entsprechend sauber zu halten. Die Erstellung der Funktionalität hat aber Priorität.

Prototyping
Mittels Prototyping können angewendete Technologien und Algorithmen getestet und optimiert werden.

Verwendung von Repository
Bietet den Team-Mitgliedern die Möglichkeit, immer auf die gleichen Grundlagen zurückzugreifen.

Unified Process / Beta Testing
Wir halten uns an die Richtlinien des Unified Processe.
Durch Beta-Tests werden Feedback, Resultate und Erfahrungen von Beta-Testern gesammelt und in die Umsetzung eingearbeitet.

10.7 Auswertung

Aufgrund grosser Ist- Soll Abweichungen während den ersten vier Wochen wurde die Zeiterfassung eingestellt. Die Planung weicht sehr stark ab, da wir für das Studium der Unterlagen und des Hauptskripts wesentlich mehr Zeit beansprucht haben. Dies ist aus der persönlichen Zeiterfassung zu entnehmen.

Die anschliessende Taskbearbeitung erstellten wir aufgrund der Prioritätsliste der Anforderungen. Bis Mitte der Hälfe des Semesters haben wir uns mit den Grundlagen beschäftigt und erst in den letzten vier- fünf Wochen verstanden wir die Theorie soweit, sodass wir uns an die Entwicklung der Demoversion wagen konnten. Dem entsprechend erhöhte sich der Aufwand. Die Vorgeschlagene Zeit von 225 Stunden pro Teammitglied wurde übertroffen.

Kapitel 11

Erfahrungsberichte

11.1 Stefan Hardegger

11.1.1 Thema der Arbeit

Das Thema der Arbeit hat mich von Anfang an sehr interessiert. Das Thema von elektronischen Abstimmungen und den möglichen Ansätzen ist sehr interessant und steckt gerade in der Umsetzung noch in den Kinderschuhen. Elektronische Abstimmungen werden in den nächsten Jahren immer häufiger eingesetzt und von daher ist es sicher von Vorteil wenn man die Grundlegenden Mechanismen, Probleme und Ansätze dahinter kennt.
Ich (und mein Kollege bestimmt auch) wurde zu Begin der Arbeit von der Komplexität und den mathematischen Formeln regelrecht erschlagen und ich brauchte viel Zeit und Geduld um die Vorgänge nach und nach zu verstehen. Gerade während der Umsetzung als Java-Applikation hatte ich einige Erleuchtungen. Vorgänge wie die Lagrange-Interpolation oder "Shamir's Secret Sharing" sind für mich nun logisch und verständlich. Daher kann ich sagen, dass ich neben der Erfahrung der Implemenation des Prototypen auch einige Mathematische und Kryptographische Vorgänge erlernt habe.

11.1.2 Planung und Umsetzung

Wir haben die Zeit, welche wir für die Erarbeitung des Papers und den dahinterliegenden Mechanismen investieren mussten ganz klar unterschätzt. Darunter hat vor allem die Planung und Durchführung der Entwicklung des Prototypen gelitten. Wir haben zu spät mit der Umsetzung des Prototypen begonnen und sind dementsprechend knapp damit fertig geworden. Erst knapp eine Woche vor Abgabe konnten wir eine ganze Abstimmung von Anfang (Initialisierung von Authorities) bis Ende (Auszählen der Stimmen) durchspielen. Sehr überrascht war ich, dass die Umstellung von 6bit auf 2046bit Schlüsseln ohne jegliche Probleme von statten ging und die Auszählung zwar einige Zeit länger dauerte, jedoch fehlerlos von statten ging.

11.1.3 Dokumentation und LaTeX

Ebenfalls zu wenig Zeit eingeplant wurde für die Dokumentation. Vor allem da wir beide noch nie zuvor mit LaTeXgearbeitet hatten, fiel die Einarbeitung relativ mühsam aus. Nach der Einarbeitung habe ich LaTeXjedoch schätzen gelernt, denn vor allem die Darstellung von mathematischen Formeln (welche in diesem Dokument nicht selten anzutreffen sind) gestaltet sich sehr einfach.

11.2 Thomas Vogler

11.2.1 Thema der Arbeit

Ich kann mich genau an die Stunde im Jahre 2006 erinnern. Nachdem das QFeedback installiert worden ist, hatten viele Informatiker Zweifel, dass irgend etwas mit die Privacy nicht stimmen kann. Bei einer Live-Demo wurden unsere Bedenken bestätigt. Die Verbindung zwischen einer Umfrage und dem Benutzer war vorhanden.
Als ich vor einem halben Jahr die Ausschreibung von Herrn Steffen gelesen habe, wusste ich, was meine Priorität eins für die Studienarbeit ist. Das Thema um digitale Wahlen interessiert mich sehr. Die Thematik wird immer wichtiger und viele Gemeinden möchten diese aussergewöhnliche Methode zur Abstimmung einsetzen. Die Bearbeitung der Thematik hat sehr viel Spass gemacht, obwohl wir viele ratlose Stunden hatten und manchmal keinen Lösungsansatz hatten.

11.2.2 Planung und Umsetzung

Zu Beginn der Arbeit hatten wir sehr grosse Mühe mit dem Skript "A Secure and Optimally Efficient Multi-Authority Election Scheme, 1997" von R. Cramer, R. Gennaro und B. Schoenmakers. Die vielen mathematischen Formeln, die allgemein sehr schwer verständlich gehaltene englische Sprache verhinderten ein rasches Bearbeiten der Theorie. Viele Stunden brüteten wir über einzelnen Sätzen. Die Komplexität haben wir stark überschätzt. Mit der Hilfe von Herrn Steffen konnten wir wichtige Erkenntnisse erarbeiten.
Die Programmierung der einzelnen Projekte nahm mehr Zeit in Anspruch als wir geplant haben. Die Verbindung zwischen den einzelnen Elementen hat uns vor unvorhergesehene Probleme gestellt. Die letzten vier Wochen waren wohl die intensivsten Projektwochen an der HSR. Viele Erkenntnisse haben wir erst während dieser Zeitspanne erarbeiten können. Die Zeiterfassung haben wir ab der fünften Woche verworfen, da sie einfach nicht mehr projektbezogen war. Der anvisierte Aufwand von rund 225 Stunden pro Mitglied haben wir zu 100 Prozent erreicht.

In dieser Arbeit habe ich sehr viel gelernt. Primär waren das Themen über elektronische Wahlsysteme, die dahinter liegenden mathematischen Verfahren zur Verschlüsselung und Anonymisierung. Mein Wissen in Webapplikationen und Java konnte ich erweitern.

11.2.3 Dokumentation und LaTeX

Zum ersten Mal habe ich mit LaTeXgearbeitet. Dieses Dokumentationstool ist äusserst stark, die Einarbeitung ist aber wesentlich höher als bei einem Microsoft Word. Die Vorteile von Latex sind umfassend. Beispielsweise die Einfache Darstellung von mathematischen Formeln, die flexible Struktur und Möglichkeit zum Input von neuen Daten oder die sauberen Darstellungsmöglichkeiten haben mich überzeugt.

11.2.4 Teamarbeit

Die Arbeit mit Stefan Hardegger und Herrn Steffen war sehr angenehm. Stefan und ich konnten uns sehr gut ergänzen. Beispielsweise hatten wir lange keine Konflikte in Subversion, da er vor allem am Abend und ich meist etwas später in der Nacht gearbeitet haben. Es war wie ein Outsourcing Projekt, praktisch 24 Stunden pro Tag wurde in den letzten paar Wochen gearbeitet. Mein Dank gilt Herrn Steffen, der uns beim Verständnis der Theorie stark geholfen hat.

Abbildungsverzeichnis

Anhang A

Verzeichnisse

A.1 Glossar

Das Glossar wurde mit Hilfe von [Wiki] und [Comp] erstellt.

Authority	Repräsentiert einen Stimmenzähler. Ist der Besitzer eines Shared Secrets.
El Gamal	Das Elgamal-Kryptosystem ist ein Schema zur Verschlüsselung, das auf dem mathematischen Problem des diskreten Logarithmus beruht.
Kriterium	Eine Menge von Abstimmungswerten. Repräsentiert ein zu bewertendes Kriterium resp. eine "1 aus n"-Wahlmöglichkeit.
Question	siehe Kriterium
Secret Sharing	Die Aufteilung eines Geheimnisses in mehrere Werte. In Kontext dieses Projektes typischerweise "Shamir's Secret Sharing" welches die Rekonstruktion eines Geheimnis mit Hilfe der Lagrange-Interpolation vornimmt.
Shared Secret	Ein Teilstück eines Geheimnis.
Survey	siehe Umfrage
Umfrage	Eine Menge von Kriterien und dazugehörigen Abstimmungswerten.

A.2 Abkürzungsverzeichnis

JSP	Java Server Pages sind Webseiten, die mittels der Programmiersprache Java erzeugt werden und dabei auf dem Server ausgeführt werden.
DML	Die Data Manipulation Language (DML) ist derjenige Teil einer Datenbanksprache, der verwendet wird, um Daten zu lesen, zu schreiben, zu ändern und zu löschen.
LDAP	LDAP ist die Abkürzung von Lightweight Directory Access Protocol, einem Protokoll, das für den Zugriff auf Informationssammlungen entwickelt wurde.

Literaturverzeichnis

[ASO06] Auslandscheizer Organisation, Noch kein e-voting bei Parlamentswahlen, http://www.swissinfo.org/ger/swissinfo.html?siteSect=881sid=7162450

[Bla79] G.R. Blakley, Safeguarding Cryptographic Keys, Proceedings of the National Computer Conference, American Federation of Information Processing Societies, Vol. 48, 1979, 313-317

[Br02] Bericht über den Vote électronique Chancen, Risiken und Machbarkeit elektronischer Ausübung politischer Rechte

[Br06] Bericht über die Pilotprojekte zum Vote électronique

[CGS97] R. Cramer, R. Gennaro, B. Schoenmakers. A Secure and Optimally Efficient Multi-Authority Election Scheme, 1997

[Comp] Computerlexikon.Com-Teams, Computerlexikon, http://www.computerlexikon.com

[ElG85] Taher ElGamal, "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", IEEE Transactions on Information Theory, v. IT-31, n. 4, 1985

[For04] Forschungsgruppe Internetwahlen: Strategische Initiative in der Bundesrepublik Deutschland: Wählen im Internet, 2004. http://www.internetwahlen.de Stand: 17.01.2004

[L99] C. Longchamp, Trends in der Stimm- und Wahlbeteiligung in der Schweiz, http://www.polittrends.ch/polittrends/beteiligung.html

[L01] W. Linder, Gutachten zum e-voting, Universität Bern, 2001

[MBC01] Emmanouil Magkos and Mike Burmester and Vassilis Chrissikopoulos. Receipt-freeness in Large-scale Elections without Untappable Channels. In B. Schmid et al., editor. First IFIP Conference on E-Commerce, EBusiness, E-Government (IEEE), Seiten 683-694, 2002.

[M01] http://search.parlament.ch/cv-geschaefte?gesch_id=20003190

[PK05] E. Prader, D. Knöri, So funktioniert das Zürcher E-Voting, Infoweek 10.10.2005

[Sha79] A.Shamir, How to share a secret. Communications of the ACM, 22(11):612-613, 1979

[SR161] Schweizerische Gesetzgebung, Verordnung über die politischen Rechte, http://www.admin.ch/ch/d/sr/161_11/index.html

[T06] Apache tomcat dokument zur Installation von ssl, T06.html

[V07] S. Vetsch, [Zuerich] Anregung zu neuem Thema (E-Voting), http://sonne.alt-f4.ch/pipermail/zuerich/2007-February/001013.html

[W04] P. Wilm, Elektronische Wahlen- Eine Informationsbroschüre für den Wahlbürger,2004

[Wiki] freiwillige Autoren, Enzyklopädie, http://de.wikipedia.org

Anhang B

weitere Dokumente

B.1 Dokumente des Projektes

AQfeedback.pdf	Dokumentation der Studienarbeit

B.2 Source-Code

Der Sourcecode wurde mit Java Doc dokumentiert. An dieser Stelle steht der Verweis auf das externe Verzeichnis mit den Dokumentationen.
Im Order src befinden sich sämtliche Java SourceCodes.

B.3 CD-Inhalt

- Beispiele
  - readme.txt
  - Beispiel 1 (Testumgebung mit 6bit Schlüssel)
  - Beispiel 2 (Testumgebung mit 2048 bit Schlüssel)
- Dokumentation
  - AqFeedback.pdf
  - Aufgabenstellen.doc
  - externe Dokumente
    - Referenzen
  - Latex
- Installationsdateien
  - Apache Tomcat
  - Postgressql
- Java Doc
- SourceCode
  - applet
  - aqfeedback
  - authority
  - servers
  - surveygenerator