

# Secure Messaging: WhatsApp Alternative

**Bachelorarbeit**  
Institut für Computerwissenschaften  
Fakultät Naturwissenschaften  
Universität Salzburg

vorgelegt von  
**Thomas Hütter**

Betreuer: Ass.-Prof. Mag. Dr. Bernhard Collini-Nocker

Thomas Hütter  
Matrikelnummer: 1120239  
Wildenauer Straße 28  
4931 Mettmach

---

# Inhaltsverzeichnis

<b>Abbildungsverzeichnis</b>	<b>I</b>
<b>Abstract</b>	<b>II</b>
<b>1 Einführung</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Aktuelle Systeme . . . . .	2
1.2.1 WhatsApp . . . . .	2
1.2.1.1 Allgemein . . . . .	2
1.2.1.2 Funktionen . . . . .	2
1.2.1.3 Probleme . . . . .	3
1.2.2 Alternativen . . . . .	5
1.2.2.1 ThreeMa . . . . .	6
1.2.2.2 Wickr . . . . .	6
1.3 Allgemeine Sicherheitsprobleme . . . . .	7
1.4 Aufgabenstellung . . . . .	8
<b>2 Umsetzung “SecMes“</b>	<b>9</b>
2.1 Anforderungen an das System . . . . .	9
2.2 Konzept . . . . .	10
2.2.1 Schließfach erstellen . . . . .	10
2.2.2 Nachrichtenschlüssel . . . . .	11
2.2.3 Datenaustausch . . . . .	11
2.2.4 Schließfach verwenden . . . . .	12
2.2.5 Verschlüsselungsalgorithmen . . . . .	13
2.3 Implementierung . . . . .	13
2.3.1 Datenbank . . . . .	13
2.3.1.1 ER-Modell . . . . .	13
2.3.1.2 Timeout-Funktion . . . . .	15
2.3.2 Schlüsselstellen . . . . .	16
2.3.2.1 Diffie-Hellman . . . . .	16
2.3.2.2 Nachrichtenabfrage . . . . .	17
2.3.2.3 Datenerhaltung . . . . .	18
2.3.3 Verwendete Bibliotheken und Programme . . . . .	18

2.3.3.1	jQuery . . . . .	18
2.3.3.2	jQuery Mobile . . . . .	19
2.3.3.3	Zend . . . . .	20
2.3.3.4	Movable-Type . . . . .	20
2.3.3.5	XAMPP . . . . .	20
2.4	Design und Funktion . . . . .	20
2.5	Sicherheitsaspekte von SecMes . . . . .	27
2.6	Optimierungen . . . . .	28

**Literaturverzeichnis**

# Abbildungsverzeichnis

1.1	Accountverifizierung im alten System [gig13]	4
1.2	Accountverifizierung im neuen System [gig13]	5
2.1	Konzept des Kommunikationsweges	12
2.2	ER-Modell der Datenbank	14
2.3	Diffie-Hellman Implementierung	17
2.4	Seitengliederung in jQuery	21
2.5	Startseite	22
2.6	Schließfach öffnen	22
2.7	Schließfach erstellen	23
2.8	Schließfach erstellt	23
2.9	Entschlüsselte Nachricht	24
2.10	Verschlüsselte Nachricht	24
2.11	Einstellungen (Admin)	25
2.12	Adminfunktionen öffnen	26
2.13	Adminfunktionen verwenden	26

# Abstract

WhatsApp is a well known and frequently used instant messenger, which didn't make headlines for only positive reasons. Like a lot of other programs, it has to deal with some serious security issues. To mention one of the worst problems, for the first three years, WhatsApp didn't use any encryption for their text messages. But still, after years of developing, the application don't achieve the actual security standards.

This leads to the purpose of this bachelor thesis, which is to implement an instant messenger with focus on a new security concept, called SecMes. The first part of this paper takes a look at existing messenger, like WhatsApp, and analyses all aspects in relation to functionality and security. The results shows that topics like encryption standards, key management, anonymity, data integrity and authentication are the key points in matters of security.

Creating the concept is based on this information gathered in the first part. In simple terms, SecMes is a mixture between a locker and a mailbox, where users can put there messages and others can access them. All lockers are created by users, have a unique ID and a password, that is defined by the user. The password isn't used to encrypt the messages. The user himself is reasonable for the encryption, the system only gives him the opportunity to do it. So, the user has to define a key, which is used to encrypt the message communication. This key must be shared with the other users over a secure channel. There are two possibilities to do this. The first is to meet personally and the second with an implemented key exchange algorithm, called Diffie-Hellman. The second one can only be done by superusers of a locker, which are users, that know the locker password.

This concept is implemented as an instant messenger, that is based on HTML 5, which leads to a system and device independent usage.

# Kapitel 1

## Einführung

### 1.1 Motivation

Eines der aktuell wohl brisantesten Themen ist die Netzwerksicherheit. Von vielen Medien berichtet, wurde auch der breiten Masse ein Einblick in die Problematik des Abhörens, sowie des Datenschutzes gewährt. Doch trotz der Enthüllungen von Julian Assange oder Edward Snowden ist sich ein Großteil der Bevölkerung noch immer nicht dem Ernst der Lage bewusst. Laut einer Online-Umfrage von T-Online können sich 76 Prozent der Deutschen nicht vorstellen, dass sie persönliche Nachteile durch die NSA haben. [to13] Dass es zum Eindringen in die Privatsphäre keinen nationalen Sicherheitsdienst benötigt, sondern auch Einzelpersonen durchaus enorme Schäden anrichten können, stößt vermutlich auf noch weniger Verständnis. Über das Internet verbreiten sich Programme, wie Sniffer, etc., deren Anwendung so einfach ist, dass man dafür kein Expertenwissen mehr benötigt.

Eine Art von Anwendungen, die häufig in Verbindung mit unsicheren Systemen gebracht wird, sind Instant Messenger, dessen Nutzeranzahl mit dem Verkauf von Smartphones steigt. Ein Vorteil von diesen Nachrichtendiensten, die ihre Inhalte über das Internet Protokoll senden, ist, dass sobald eine Internetverbindung, sei es durch ein WLAN oder eine mobile Datenübertragung, besteht, ortsunabhängig kostenlose Nachrichten versendet werden können. Weiters sind meist zusätzliche Funktionen wie Gruppenchats oder das Übertragen von Medien (Fotos, Videos, etc.) ausschlaggebend für die Benutzung. Da die Informationen, wie bereits beschrieben über IP gesendet werden, besteht natürlich immer die Gefahr, dass die Pakete gelesen oder sogar verändert werden. Diesen Möglichkeiten sollte durch Verschlüsselung und anderen Sicherheitsmechanismen Einhalt geboten werden. Im nächsten Teilkapitel wird auf bestehende Systeme, deren technische Umsetzung, sowie deren Sicherheitsprobleme eingegangen.

## 1.2 Aktuelle Systeme

### 1.2.1 WhatsApp

#### 1.2.1.1 Allgemein

Beim Thema Instant-Messaging-Anwendungen kommt man nur schwer an WhatsApp vorbei. Entwickelt wurde der Nachrichtendienst von der WhatsApp Inc., 2009 gegründet von Brian Acton und Jan Koum. In einem nur sehr kurzen Zeitraum konnte der Messenger eine Vielzahl an Benutzern für sich gewinnen. Diese senden Ihre Nachrichten bis zu 27 Milliarden mal pro Tag über die mobile App, also mehr als die klassische SMS. [chi13] Der Erfolg ist neben der Einfachheit des Systems auch auf das Fehlen von Werbung, sowie der Unterstützung auf den meisten Plattformen zurückzuführen. Trotz vieler negativer Schlagzeilen bezüglich Sicherheit und unklarer Nutzungskosten, kann WhatsApp heute 300 Millionen Menschen aus aller Welt seine Benutzer nennen. [nt13]

#### 1.2.1.2 Funktionen

Die Basisfunktion von WhatsApp ist klarerweise das Senden von Text-Nachrichten zwischen verschiedenen Benutzern. Zusätzlich werden dem Anwender durch die App folgende Möglichkeiten geboten:

- **Gruppenchats:**  
Abhängig von der benutzten Version können Unterhaltungen mit bis zu 50 Personen gleichzeitig geführt werden.
- **Senden von Fotos und Videos:**  
Foto- und Videodateien, welche auf dem mobilen Gerät gespeichert sind, können in jeder Unterhaltung versendet werden.
- **Teilen von Standorten:**  
Mit der Freigabe der, durch GPS bestimmten, Positionsdaten kann man seinen Standort oder auch eine spezielle Lokalität preisgeben.
- **Sprachnachrichten:**  
Im Nachrichtenfenster kann per Knopfdruck eine Sprachnachricht aufgenommen und wie eine Textnachricht übertragen werden.

### 1.2.1.3 Probleme

Dass die Einfachheit des Programms auch starke Nachteile hat, sieht man in puncto Sicherheit. Schnell wurden Lücken gefunden, die sowohl das Mitlesen, als auch die Datenintegrität gefährdeten. Auch wenn mittlerweile viele Schwachstellen durch Updates behoben worden sind, darf der Sicherheit des Apps nicht all zu viel Vertrauen geschenkt werden.

Als Schlagworte können folgende Begriffe genannt werden:

- Verschlüsselung der Nachrichten und Benutzerdaten
- Benutzerauthentifizierung
- Passwörterstellung

Risiken zu Beginn:

- Erschreckend ist, dass bis zum August 2012 die Nachrichten der Benutzer im Klartext übertragen wurden, die Nummer, die als Benutzername verwendet wird, sogar noch länger. Das heißt, für Personen, die sich im selben Netzwerk befinden, war das Mitlesen der Nachrichten kein Problem.
- Um dem Benutzer eine schnelle und unkomplizierte Verwendung zu ermöglichen, gibt es keine Abfrage nach Benutzernamen und Passwort. Dies führt zu einer deutlichen Verminderung der Sicherheit im Falle eines physischen Diebstahls, da jeder, der die App öffnet, Nachrichten im Namen des echten Handybesitzers versenden kann.
- Die Passwörterstellung war weder in der Android-Version, bei der das Passwort aus der Seriennummer (IMEI) des Handys erstellt wurde, noch bei iOS-Version, wo die MAC-Adresse der WLAN Schnittstelle verwendet wurde, sicher. Da diese Daten ohne größere Probleme in Erfahrung gebracht werden können, kann man mit dieser Information und der Rufnummer eines Nutzers, Nachrichten unter seinem Namen empfangen und versenden. Der Algorithmus mit dem aus IMEI oder MAC das Passwort entsteht, ist im Internet zu finden. Der Nutzer merkt nicht primär, dass auf seinen Account zugegriffen wird, da er noch immer Nachrichten senden kann. Dass er keine mehr empfängt, wird ihm wohl erst nach einem gewissen Zeitraum auffallen. Kurzzeitige Abhilfe bringt ein Neustart der App, da man sich damit neu am Server anmeldet und somit wieder der aktuelle Benutzer für das System ist. Weil sich die Zugangsdaten jedoch nicht verändern, kann der Dritte jederzeit wieder den Account kapern. [gig13] Schematisch veranschaulicht dies die Abbildung 1.1.

Neuere Versionen: (Stand Mai 2013)

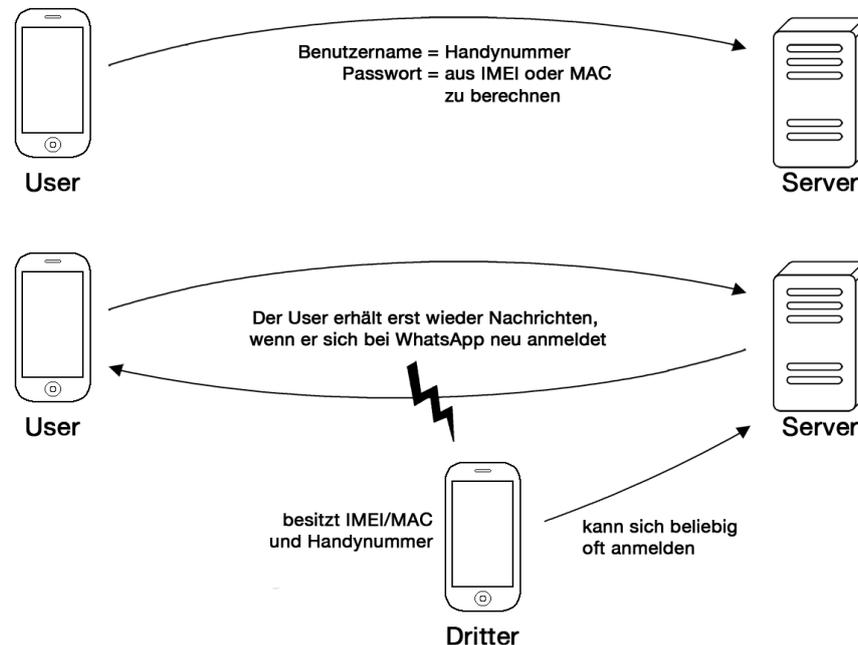


Abbildung 1.1: Accountverifizierung im alten System [gig13]

- In den aktuellen Versionen der verschiedenen Betriebssysteme wurden bereits einige Verbesserungen bezüglich der Sicherheit getroffen. So wird zwar mittlerweile ein RC4 Algorithmus zur Verschlüsselung der Nachrichten verwendet, der jedoch für ein- und ausgehende Nachrichten den selben Schlüssel verwendet. Will nun jemand eine Nachricht entschlüsseln, erhält er dadurch eine höhere Menge an Vergleichsnachrichten. [it13]
- Auch die Verifizierung funktioniert nicht mehr selbstständig über MAC-Adresse oder IMEI, sondern über folgenden Datenaustausch mit dem Server: [gig13]
  - Der Nutzer schickt dem Server seine Handynummer.
  - Der Server erstellt einen Bestätigungscode und schickt diesen per SMS an den Nutzer zurück.
  - Der Code wird nun direkt in die Anwendung eingegeben und an den Server gesendet.
  - Dieser generiert einmalig ein Passwort und sendet es dem Nutzer zurück, wo es lokal gespeichert wird.

Eine Übernahme des Accounts wird somit auch deutlich erschwert, aber nicht unmöglich. Im Gegensatz zum alten System benötigt ein Dritter anstatt der IMEI/-MAC nun physischen Zugriff auf das Smartphone des Opfers. Wie in Abbildung 1.2

zu sehen ist, sendet der Dritte zu Beginn die Handynummer an den WhatsApp-Server, der auf das Handy des Opfers den Bestätigungscode zurück sendet. Der Angreifer liest nun den Code aus und gibt ihn bei sich am Smartphone ein. Dieses kommuniziert mit dem Server und erhält das Passwort. Nun ist es nicht nur schwieriger den Account zu kapern, auch das Opfer merkt viel schneller, dass etwas nicht stimmt. Will er die App benutzen, merkt der Server das sein lokal gespeichertes Passwort nicht mehr gültig ist. Er bietet ihm an, ein neues zu generieren. Ab dem Zeitpunkt der Bestätigung dieses Angebots, kann der Dritte den erworbenen Account nicht mehr nutzen und müsste den Vorgang wiederholen.

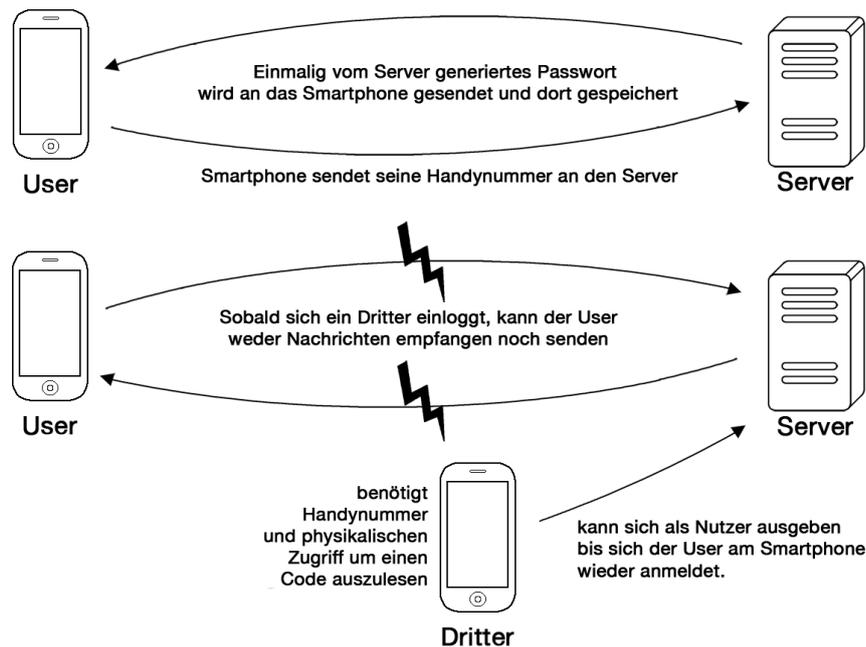


Abbildung 1.2: Accountverifizierung im neuen System [gig13]

Das noch immer viel zu verbessern ist, sieht man am im Juli 2013 erschienenen WhatsApp-Exploit "Priyanka", der die Namen aller Kontakte in eben dieses Wort umändert. Das Programm kann sich weder selbst verbreiten, noch installieren, dies funktioniert lediglich durch die Bestätigung der Priyanka Visitenkarte. [hei13]

### 1.2.2 Alternativen

Aufgrund der großen Nachfrage ist der Markt für Instant Messaging mittlerweile riesig. Egal ob iMessage, Skype, joyn oder WhatsApp, um nur einige zu nennen, der Benutzer kann stundenlang vergleichen und einen Dienst nach seinen Anforderungen auswählen.

---

Bezüglich der Sicherheit stehen jedoch zwei Vertreter immer öfter positiv im Vordergrund, Threema und Wickr.

### 1.2.2.1 ThreeMa

[thr13] Hier handelt es sich um ein Produkt der Firma Kasper Systems GmbH aus der Schweiz. Funktional versucht Threema Anschluss an seine Konkurrenten zu finden. Bis dato fehlen aber zum Beispiel noch der Gruppenchat und die Sprachnachrichten. Dafür wird sicherheitstechnisch um einiges mehr geboten, wobei hier die wesentlichen Punkte hervorgehoben werden.

Threema bietet seinen Kunden ein zweischichtiges Verschlüsselungsmodell an. Dies beinhaltet eine End-to-End Kommunikation zwischen den Teilnehmern, sowie eine zusätzliche Verschlüsselung zwischen dem App und dem Server. Durch diese zwei Schichten wird davon ausgegangen, dass die Übertragung sicher ist und das Problem höchstens darin bestehen kann, dass der Kommunikationspartner nicht die geschwünschte Person ist. Darum besitzt die App als Alleinstellungsmerkmal ein 3 stufiges Sicherheitssystem, indem die Vertrauenswürdigkeit des Unterhaltungspartners bewertet wird:

- Stufe 1: Man besitzt lediglich die Threema-ID eines anderen Nutzers
- Stufe 2: Die Daten des Kontaktes wurden zwischen Telefonbuch des Nutzers und dem Threema Server verglichen
- Stufe 3: Bei einem persönlichen Treffen der Unterhaltungspartner kann der QR-Code des anderen gescannt und somit weitgehend sichergestellt werden, dass es sich um die Person handelt, die man erwartet.

Durch Elliptic Curve Cryptography, einer Hashfunktion, XSalsa20, einem Message Authentication Code und Füllbits wird auch auf Verschlüsselung und Datenintegrität extremer Wert gelegt.

### 1.2.2.2 Wickr

[myw13] Einen anderen interessanten Ansatz verfolgt die in Kalifornien entwickelte App Wickr. Entwickelt wurde das Programm von einer Gruppe aus Sicherheitsexperten, die es sich als Ziel gesetzt haben, einen sicheren Instant Messenger zu erstellen. Neben den neuesten Verschlüsselungsstandards (AES256, ECDH521, RSA4096, TLS) wird dem Benutzer hier eine "Selbstzerstörungsfunktion" angeboten. Jeder Verwender kann über die Einstellungen einen Zeitraum wählen, der die Dauer zwischen dem Lesen des Empfängers und dem endgültigen Löschen der Nachricht (Selbstzerstörung) festlegt. Wie auch bei Threema wird angegeben, dass das Programm, keine Benutzerdaten speichert.

Wickr steht WhatsApp mit den möglichen Funktionen um nichts nach, im Gegenteil,

---

hier können auch PDFs versandt werden. Als zusätzliche Sicherheitsfunktion werden sogar alle Meta-Daten einer Datei vor dem Senden gelöscht. Einziger Kritikpunkt von Experten ist der fehlende Source Code, den die Entwickler auch nicht veröffentlichen wollen. [mas13]

## 1.3 Allgemeine Sicherheitsprobleme

Natürlich gibt es eine Vielzahl von weiteren Mitbewerbern auf diesem Markt, die sich durch unterschiedliche Stärken und Schwächen auszeichnen. Das Wort “sicher“ ist trotz der besten Verschlüsselungsalgorithmen immer mit Vorsicht zu genießen. Die wesentlichen Sicherheitsaspekte, die bei einem Instant Messenger beachtet werden sollen, sind:

- **Nachrichtenverschlüsselung:**  
Es ist sowohl auf ein sicheres Verfahren, als auch auf gute Schlüssel zu achten. Beim Thema Schlüssel spielt natürlich auch eine sichere Übertragung dieser, sowie deren Speicherung, eine wichtige Rolle.
- **Anonymität:**  
Weder der Server, ein Kommunikationspartner, noch im schlimmsten Fall eine dritte Person, sollen Zugriff auf persönliche Daten des Nutzers bekommen. Das heißt, es sollen weder private Daten bei der Anmeldung benötigt werden, noch auf lokale Informationen zugegriffen werden.
- **Integrität:**  
Die übertragene Information sollte korrekt beim Empfänger ankommen, sollte es doch zu einer Modifikation kommen, muss diese schnellstmöglich erkannt werden.
- **Authentifizierung:**  
Stellt in Konkurrenz mit der Anonymität, sowie der Usability, einen Kompromiss dar. Es sollen zwar keine persönlichen Daten gespeichert werden, jedoch sichergestellt, dass es sich um den gewünschten Anwender handelt. Darum muss abgewogen werden, wie oft und nach welchen Daten ein Benutzer abgefragt werden soll.

Zwei Punkte, die bei einem Großteil der Anwendungen zu kurz kommen, sind Anonymität und Authentifizierung. Grund dafür ist eine Steigerung des Anwendungskomforts für den Nutzer.

So sucht die App einfach nach persönlichen Daten (z.B. im Telefonbuch), um dem Anwender gleich alle möglichen Kontakte vorzuschlagen, die den Messenger ebenso verwenden. Auf die Authentifizierung des Nutzer wird verzichtet, um nicht bei jedem Öffnen einer App den Benutzernamen und das Passwort eingeben zu müssen. Aber auch eine Nachrichtenverschlüsselung ist nicht so selbstverständlich wie man annimmt, siehe WhatsApp.

---

## 1.4 Aufgabenstellung

Daraus ergibt sich folgende Aufgabenstellung für die Implementierung:

Ziel dieser Arbeit ist es, unter Beachtung der in Abschnitt 1.3 erarbeiteten sicherheitsrelevanten Punkte, ein System zur Nachrichtenübertragung zu konzeptionieren und als Anwendung zu implementieren. Das Programm soll als WebApp, basierend auf HTML5, umgesetzt werden, um eine systemunabhängige Benutzung, von Desktop-PCs bis hin zu mobilen Endgeräten, zu ermöglichen.

# Kapitel 2

## Umsetzung “SecMes“

Dieses Kapitel beschreibt das Konzept, die einzelnen Schritte der Umsetzung, sowie deren technische Details. “SecMes“ ist ein Akronym für “Secure Messaging“ und zugleich der Arbeitstitel des Projekts, der in weiterer Folge als Name dafür verwendet wird.

### 2.1 Anforderungen an das System

Ausgehend von Abschnitt [1.3](#) kann man folgende Anforderungen für die WebApp stellen:

- Es soll kein Zugriff auf lokale Daten erfolgen, weder gelesen noch geschrieben. (z.B.: Kontaktbuch oder das Ablegen von Schlüsseln oder Passwörter)
- Private Daten, wie E-Mail, Name oder Telefonnummer, vom Benutzer sollten nicht benötigt werden.
- Verwendung eines guten Verschlüsselungsstandards.
- Schlüssel müssen verändert, sicher ausgetauscht und gespeichert werden können.
- Eine Anmeldung soll vor jeder Benutzung der App durchgeführt werden.
- Keine permanente Speicherung von jeglicher Information, also Nachrichten, Benutzer und Schließfächer sollen nicht ewig existieren.

---

## 2.2 Konzept

Die Idee hinter der Anwendung kann man sich als eine Mischung zwischen Post- und Schließfach vorstellen. In diesem Fach findet der komplette Austausch von Nachrichten zwischen zwei oder mehreren Kommunikationsteilnehmern statt.

Der erste Schritt ist das Erstellen eines Schließfachs. Dieser und alle weiteren Aktionen werden in den folgenden Unterkapiteln erklärt.

### 2.2.1 Schließfach erstellen

Schließfächer werden immer von Benutzern erstellt, welche aufgrund des von ihnen gewählten Passwortes zukünftig auch als Admin innerhalb dieses Schließfachs fungieren. Beim Erzeugen werden prinzipiell 3 Daten benötigt:

- Schließfach-ID:  
Hierbei handelt es sich um eine eindeutige Kennung eines Schließfaches. Diese kann sowohl vom Benutzer eingegeben, als auch vom System generiert werden. Bei der Erstellung durch das System wird eine dreistellige Zufallszahl in Kombination mit einem Timestamp in folgender Form als ID verwendet.

$$ID = \text{Random}(999) + \text{Timestamp}(YmdHis)$$

Als Ergebnis wird immer ein 17-stelliger String geliefert, der dem Benutzer nach der Erstellung angezeigt wird, um sich diesen notieren zu können.

- Passwort:  
Das Passwort dient als Erkennungsmerkmal für den Schließfach-Admin. Bei diesem handelt es sich um eine spezielle Person im System, die man als Verwalter des von ihm erstellten Schließfaches bezeichnen kann. Er verfügt über diese speziellen Rechte:
  - Austauschen des Schlüssels innerhalb einer Konversation
  - Löschen des Nachrichtenverlaufs
  - Löschen des Schließfachs

Wichtig ist, dass mit Passwort nicht der Schlüssel gemeint ist, der für das Verschlüsseln der Nachrichten verwendet wird. Um eine Abfrage nach dem Passwort, auf welches eine SHA-256 Hash Funktion angewandt wurde, zu ermöglichen, wird es in der Datenbank abgelegt.

Der Schließfach-Admin kann dieses Passwort natürlich weitergeben, wenn er mehrere Admins für sein Schließfach haben will.

- Timeout:  
Mit dem Timeout definiert man einen weiteren Sicherheitmechanismus des Systems. Der angegebene Wert gibt eine Zeitspanne in Tagen an, die bestimmt, wie viel Zeit zwischen dem Eingehen der letzten Nachricht und dem Löschen eines Schließfachs verstreichen soll. Der Löschvorgang wird vom System durchgeführt und funktioniert automatisch.

### 2.2.2 Nachrichtenschlüssel

Hierbei handelt es sich im Vergleich zum Passwort um den Schlüssel, der für die Nachrichtenverschlüsselung verwendet wird. Sowohl die Schließfach-ID zum Öffnen des Schließfachs, als auch der Schlüssel, müssen jedem Teilnehmer, der verschlüsselt Nachrichten übertragen und lesen will, zur Verfügung stehen. Prinzipiell kann jeder einen Schlüssel festlegen, mit dem er seine Daten verschlüsselt und versucht, die anderen Nachrichten zu entschlüsseln. Die zentrale Person ist jedoch immer der Admin des Schließfachs, denn er hat die Möglichkeit seinen Schlüssel innerhalb der Anwendung mit den anderen Teilnehmern auszutauschen.

### 2.2.3 Datenaustausch

Prinzipiell gibt es drei verschiedene Daten, die ein Schließfach-Admin anderen zur Verfügung stellen kann. Dabei handelt es sich um die Schließfach-ID, den Nachrichtenschlüssel und das Passwort. Hier gibt es mehrere Möglichkeiten diese auszutauschen:

- Persönlich:  
Über den persönlichen Weg können natürlich alle Personen Informationen austauschen. Unter der Voraussetzung, dass man einen guten Ort des Treffens wählt, ist dies die sicherste Variante Daten auszutauschen. Dadurch können aufgrund der hohen Sicherheit alle Daten ausgetauscht werden, so erspart man sich eine zusätzliche digitale Übertragung.
- Digital
  - Im Programm:  
Für den Admin bietet der Messenger in den Einstellungen eine Funktion, die alle Kommunikationsteilnehmer auflistet und durch Auswählen eines Namens den Schlüsselaustausch via Diffie-Hellman (Kapitel 2.3.2.1) startet. Diese Funktion steht nur für den Nachrichtenschlüssel zur Verfügung. Zusätzlich kann in einer Unterhaltung natürlich auch die ID für ein anderes Schließfach ausgetauscht werden.

- Sonstige digitale Übertragungsmittel:  
Die Schließfach-ID kann über nicht sichere Kommunikationsmittel wie E-Mail, SMS oder anderen Programmen übertragen werden. Somit ist es dem Teilnehmer möglich der Unterhaltung beizutreten. Da er sich jetzt zwar in der Konversation befindet, aber noch keinen Schlüssel hat, muss dieser noch mit einer der oben genannten Methoden vom Schließfach-Admin übertragen werden.

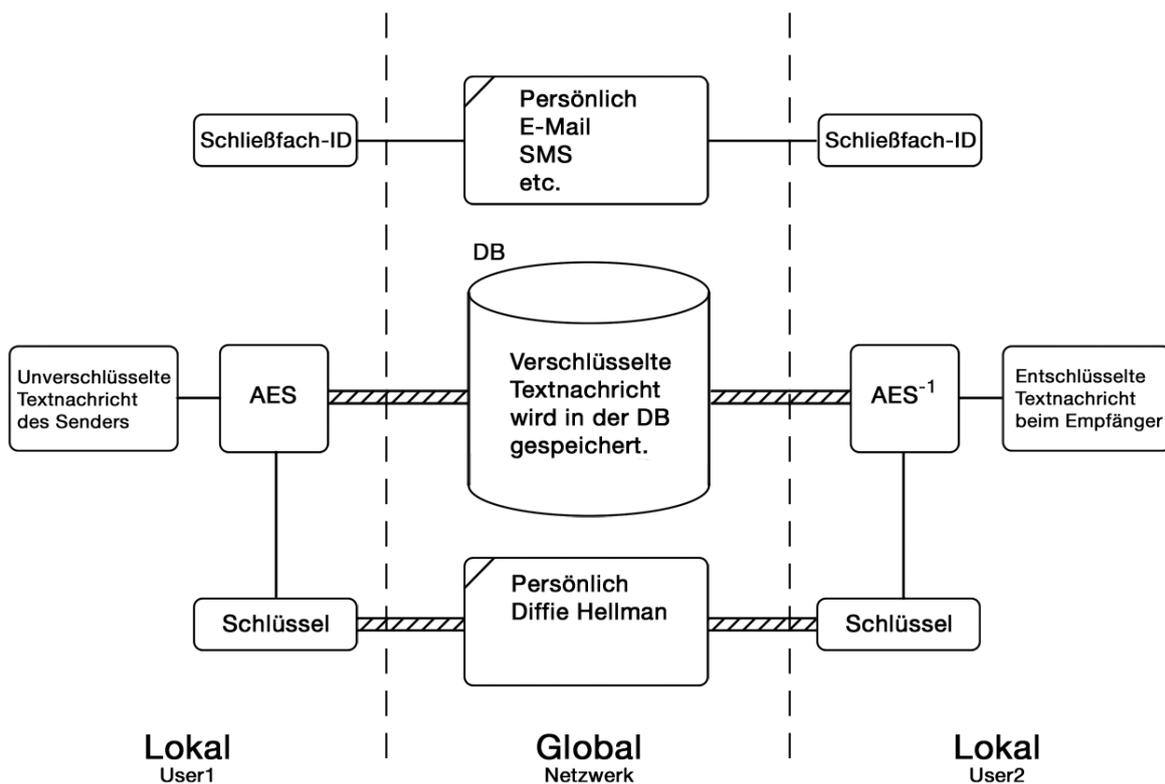


Abbildung 2.1: Konzept des Kommunikationsweges

### 2.2.4 Schließfach verwenden

Ein Teilnehmer erhält vom Admin die Schließfach-ID auf eine der in Abschnitt 2.2.3 beschriebenen Varianten. Besitzt ein Teilnehmer diese, kann er mit dieser und einem eindeutigen User-Namen der Kommunikation beitreten. Er sieht bereits jetzt alle eingetragenen Nachrichten und kann auch selber welche schreiben. Ohne den richtigen Schlüssel sieht er bei verschlüsselten Nachrichten jedoch nur kryptische Zeichen. Kennt er den Schlüssel bereits, so kann er diesen einfach in den Einstellungen eintragen, ansonsten

---

muss er auf einen Austausch durch den Schließfach-Admin warten. Nach dem Austausch oder der Eingabe kann nun eine sichere Übertragung der Nachrichten erfolgen.

### 2.2.5 Verschlüsselungsalgorithmen

Will nun einer der Nutzer eine Nachricht senden, so verschlüsselt sein Client die Nachricht mit dem ausgetauschten Schlüssel unter Verwendung einer AES Verschlüsselungsfunktion. Anschließend wird die nun verschlüsselte Nachricht zur Datenbank übertragen und dort in Abhängigkeit zum Schließfach gespeichert. Der Client des anderen Benutzers bemerkt die neue Nachricht, empfängt sie, entschlüsselt sie und zeigt sie am Display. Die Nachrichten sind also auf dem ganzen Übertragungsweg durch die Verschlüsselung gesichert.

## 2.3 Implementierung

Wie bereits in der Aufgabenstellung formuliert, soll eine Implementierung als WebApp erfolgen. In diesem Kapitel wird genauer auf die technische Umsetzung, die verwendeten Bibliotheken und das Design des Konzeptes aus [2.2](#) eingegangen.

### 2.3.1 Datenbank

#### 2.3.1.1 ER-Modell

Zu Beginn wird hier das ER-Modell der Datenbank dargestellt, da es das Konzept und die Struktur der Anwendung gut veranschaulicht. Ein zu komplexes System ist unübersichtlich und wäre dadurch vielleicht auch unsicher. Darum wurde bei “SecMes“ ein schlichtes, gut strukturiertes Datenmodell verwendet. In [Abbildung 2.2](#) sind die drei verwendeten Entitäten, samt ihren Attributen zu sehen.

- Locker:  
Das Schließfach ist sozusagen die Basis des ganzen Systems. Es besteht aus einer ID, einem Password und einem Timeout. Die Generierung der Schließfach-ID wurde bereits in [Kapitel 2.2.1](#) erklärt. Als mögliche Eingabedaten sind hier 32 Characterzeichen erlaubt. Für die Eindeutigkeit garantiert der Primärschlüssel. Die Bedeutung von Timeout wird in [Kapitel 2.3.1.2](#) erklärt.

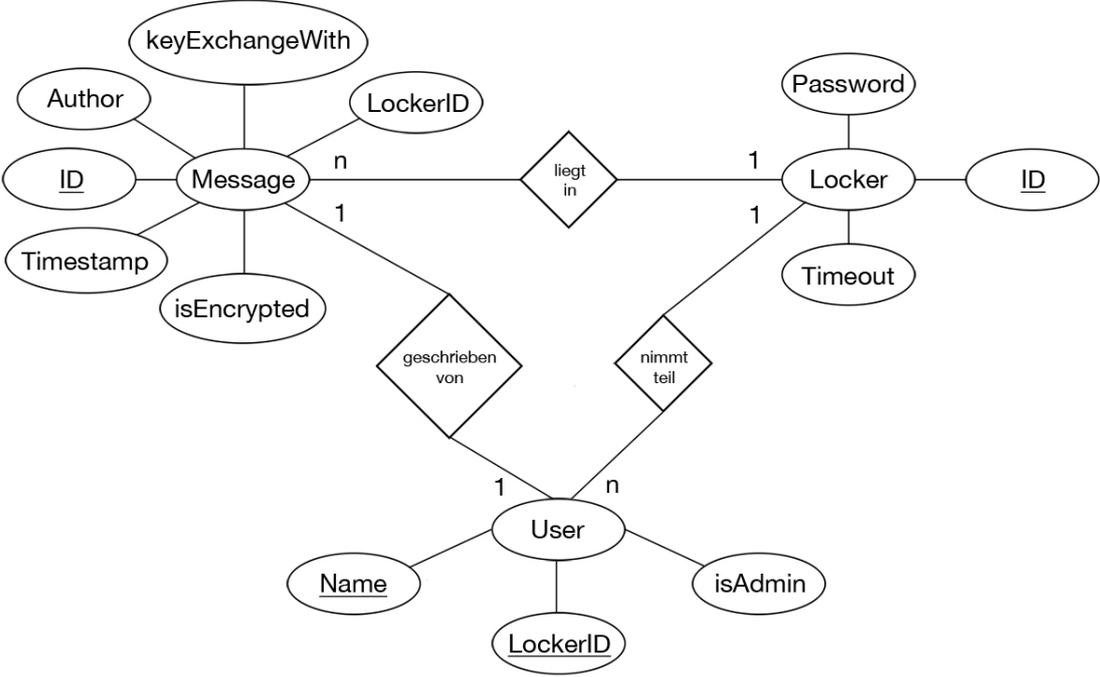


Abbildung 2.2: ER-Modell der Datenbank

- User:  
Wie man an dem Primärschlüssel erkennen kann, ist der Name pro Schließfach eindeutig. Sowohl der Name, als auch die LockerID haben eine Länge von 32 Zeichen. Das isAdmin Flag zeigt an, ob es sich bei einem Benutzer in dem jeweiligen Schließfach um einen Admin handelt. Wichtig ist hier, dass es mehrere Admins für ein Schließfach geben kann.
- Message:  
Neben einer fortlaufenden ID enthält eine Nachricht natürlich ihren Inhalt, der in dieser Implementierung mit 300 Zeichen begrenzt ist. Als zusätzliche Information wird ein Username als Author und der Zeitpunkt, an dem die Nachricht gesendet wurden, gespeichert. LockerID ist ein Fremdschlüssel auf das Schließfach, in welchem die Nachricht liegt. Somit kann keine Nachricht existieren, dessen Schließfach nicht vorhanden ist. Die beiden Attribute isEncrypted und keyExchangeWith sind Flags, die zur Verarbeitung im Code genutzt werden. isEncrypted zeigt an, ob der Absender eine Verschlüsselung verwendet hat. keyExchangeWith enthält den Namen eines Benutzers, für den die Nachricht im Rahmen des Schlüsselaustauschs bestimmt ist. Diese werden nicht im normalen Chatfenster angezeigt und von anderen Benutzern, die nicht am Austausch beteiligt sind, ignoriert.

### 2.3.1.2 Timeout-Funktion

Die im Konzept bereits erwähnte Timeout Funktion dient dazu, Schließfächer, welche seit einem gewissen Zeitraum keine Nachricht mehr erhalten haben zu löschen. Dieser Zeitraum wird vom Schließfach-Admin beim Erstellen des Schließfachs festgelegt. Generell stellt dies ein weiteres Sicherheitsmerkmal dar. Daten sollen und werden somit nicht permanent gespeichert.

Technisch wurde diese Funktion über sogenannte Events realisiert. Die hier verwendete MySQL Datenbank verfügt über diese Technik. Somit bietet es eine einfache Alternative zu einem PHP Skript, das in einem Cron Job am Server läuft.

Der Datenbank muss in der Init Datei mittels `event_scheduler=ON` mitgeteilt werden, dass sie Events abhandeln soll. Der verantwortliche Scheduler läuft permanent im Hintergrund der Datenbankanwendung und prüft, ob ein Event aufgetreten ist. Der eigentliche Event wird mit einem `CREATE EVENT` Befehl erstellt. Anschließend werden die Details gesetzt, ab wann und in welchen Abständen soll der Event ausgeführt und was soll gemacht werden. Der in “SecMes“ verwendete Event wird durch das SQL Statement in Listing 2.1 erzeugt. [Sit11]

Listing 2.1: Event erstellen

```

1 CREATE
2   EVENT `delete_locker`
3   ON SCHEDULE
4   EVERY 1 HOUR
5   DO
6   DELETE loc.* FROM Locker AS loc
7   WHERE id IN (
8     SELECT id
9     FROM (
10    SELECT l.id
11    FROM (
12     SELECT m.Lockerid AS id, MAX(m.Timestamp) AS ts, m.timeout AS tio
13    FROM (
14     SELECT me.Lockerid, me.Timestamp, lo.Timeout
15     FROM Message AS me, Locker AS lo
16     ORDER BY Timestamp DESC
17    ) AS m
18    GROUP BY m.LockerID
19    ) AS l
20    WHERE DATE_ADD(l.ts, INTERVAL l.tio DAY) < CURRENT_TIMESTAMP
21    ) AS del
22  )

```

## 2.3.2 Schlüsselstellen

### 2.3.2.1 Diffie-Hellman

Die Initialisierung und Schlüsselberechnung erfolgt über die im Zend Framework [2.3.3.3](#) enthaltene Klasse. Nach dem Einbinden dieser, müssen zuerst eine Primzahl  $p$  und ein Generator  $g$  gesetzt werden. Diese werden nun dem Konstruktor der Klasse übergeben. Mit `generateKeys()` wird nun ein zufälliger, privater Schlüssel ( $a$ ) für den Schließfach-Admin generiert und daraus ein öffentlicher Schlüssel ( $A$ ) berechnet. Anschließend wird ( $A$ ) in einer Nachricht, die einen Usernamen als Wert für `keyExchangeWith` gesetzt hat, in die Datenbank geschrieben.

Der Benutzer, der den Schlüssel empfangen soll, bemerkt beim Lesen der Nachrichten, dass sein Name im Feld `keyExchangeWith` steht. Sein Programm startet nun ebenfalls mit dem Schlüsselaustausch und führt die selben Initialisierungsschritte durch wie der Schließfach-Admin und erhält ( $b$ ) (privat) und ( $B$ ) (öffentlich). Dies resultiert mit einem Eintrag in der Datenbank, die dem Sender der vorherigen Nachricht zugeordnet ist. Der Schließfach-Admin, der auf die Antwort des Benutzers gewartet hat, empfängt die Nachricht und bekommt somit den öffentlichen Schlüssel ( $B$ ) seines Gegenübers.

Also jedem der beiden Teilnehmer steht nun der öffentliche Schlüssel des anderen zur Verfügung. Somit können nun beide durch die Diffie-Hellman Klasse den geheimen Schlüssel

( $K$ ) berechnen. Mit diesem und dem AES Algorithmus verschlüsselt der Schließfach-Admin nun seinen Nachrichtenschlüssel ( $N$ ) und schreibt ihn in die Datenbank. Der Austauschpartner empfängt ihn und entschlüsselt die AES Nachricht. Somit ist der Schlüsselaustausch vollzogen.

Die Nachrichten die beim Austausch entstehen, werden natürlich gleich nach dem Empfangen vom jeweiligen Benutzer wieder gelöscht. Grafik 2.3 illustriert nochmal den gesamten Ablauf inklusive der in der Datenbank abgelegten Werte.

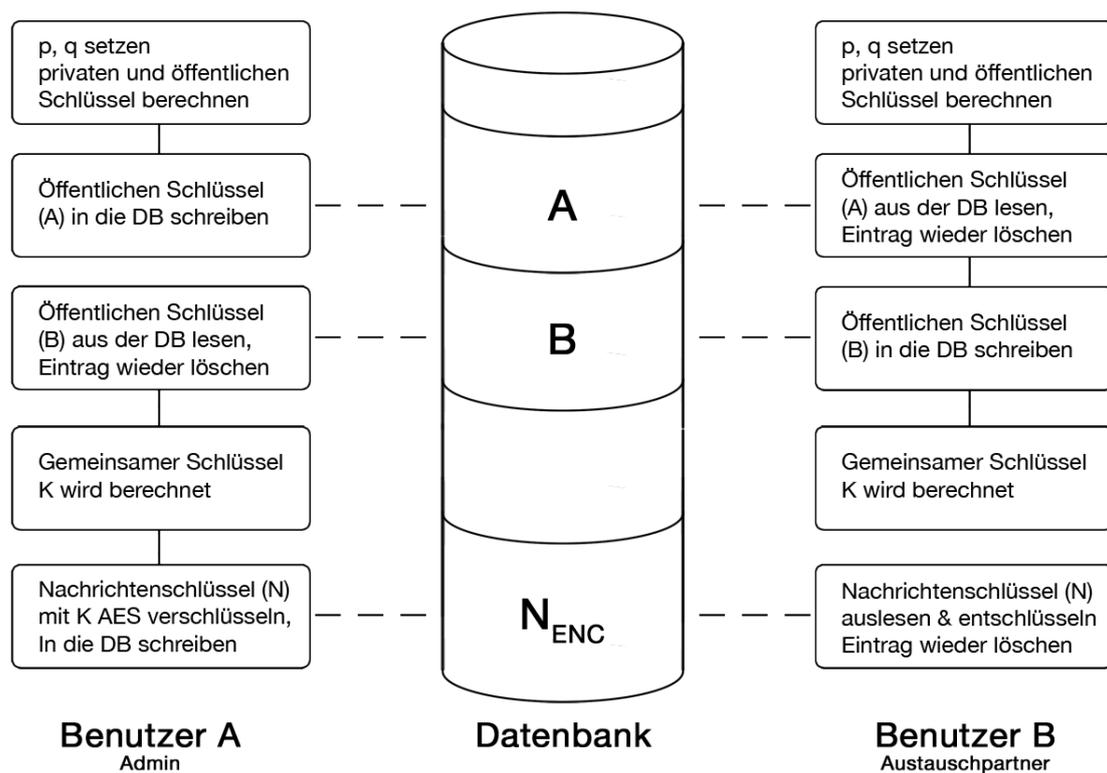


Abbildung 2.3: Diffie-Hellman Implementierung

### 2.3.2.2 Nachrichtenabfrage

In diesem Unterkapitel wird nebenbei auch der Unterschied zwischen einem Postfach und einem Instant Messenger deutlich. Während man bei einem Postfach davon ausgeht, dass man als Benutzer explizit die Nachrichten abrufen muss, redet man bei einem Instant Messenger eigentlich von einem Chat. Damit ist gemeint, dass man wie bei einer Unterhaltung sofort die Nachricht erhält, wenn der andere sie sendet. Bei "SecMes" wurde dies durch eine einfache Polling-Routine implementiert. Eine JavaScript-Funktion wird,

während man auf ein Schließfach zugreift, im Sekundentakt aufgerufen und prüft die Datenbank nach neuen Nachrichten. Jede dieser Nachrichten in der Datenbank hat eine inkrementell erhöhende Schließfach-ID. Das Script merkt sich lokal die höchste ID die es empfangen hat und prüft ab sofort nur noch auf Nachrichten mit einer höheren ID als der gespeicherten.

### 2.3.2.3 Datenerhaltung

Ab dem Zeitpunkt der Anmeldung müssen bei verschiedenen Seitenanforderungen lokale Daten zwischen diesen übergeben werden. Dabei handelt es sich, neben der Schließfach-ID und dem Usernamen, auch um vertrauliche Daten, wie den Schlüssel für die Nachrichten. Als Methode wurde hier eine Übertragung mittels POST-Variablen gewählt. Um die Daten durch eine zusätzliche Verschlüsselung zu schützen, wird HTTPS verwendet. Bei HTTPS handelt es sich um eine spezielle, sicherere Variante von HTTP. Durch das SSL-Handshake-Protokoll findet eine Identifizierung und Authentifizierung zwischen Client und Server statt. Der eigentliche Schlüsselaustausch findet, wie ebenso bei “SecMes“ zur Nachrichtenverschlüsselung verwendet, mit Hilfe eines Diffie-Hellman Algorithmus statt.[Wik13]

## 2.3.3 Verwendete Bibliotheken und Programme

### 2.3.3.1 jQuery

Bei jQuery handelt es sich um eine Javascript Bibliothek, die 2006 von John Resig veröffentlicht wurde. [jQu13b] Die Verwendung fällt unter die MIT-Lizenz<sup>1</sup>, was eine freie Nutzung ermöglicht, solange die Header in den Files vorhanden sind und nicht verändert werden. [jQu13c]

Hier folgt ein Überblick über die wichtigsten Funktionen, die auch im “SecMes“ Projekt verwendet wurden: [jQu13a]

- Zugriff und Manipulation des DOM:  
Im Beispiel 2.2 sieht man, wie dem Objekt `message_content`, also dem HTML-Element, das die Nachrichten enthält, eine neue Nachricht angefügt wird.

#### Listing 2.2: DOM Manipulation

```
1 $("#message_content").append(decrypted + "<br />");
```

<sup>1</sup><https://github.com/jquery/jquery/blob/master/MIT-LICENSE.txt>

- Event Handling:

jQuery bietet einfache Möglichkeiten, die Events einzelner Objekte abzufangen und ihre Funktionen zu bearbeiten. Im folgenden Beispiel ist die Funktion enthalten, die es ermöglicht eine Nachricht auch mit der Return-Taste abzusenden.

**Listing 2.3:** Event Handling

```
1  $("#message").bind("keypress", function(event) {
2      if(event.which == 13) {
3          $("#text_input_form").submit();
4          return false;
5      }
6  });
```

- Ajax Abfragen:

Diese Funktion ermöglicht das Absenden einer HTTP Abfrage, ohne eine vollständige Seite laden zu müssen. Hauptsächlich finden diese Abfragen statt, um interaktive Inhalte zu laden oder zu prüfen, wie das Empfangen von Nachrichten oder das Prüfen eines Formulars, wie im Beispiel unten der Fall ist. Hier wird in der Datenbank nachgesehen, ob ein Schließfachname bereits vergeben ist.

**Listing 2.4:** Event Handling

```
1  $.ajax({
2      type: "POST",
3      url: "insertLocker.php",
4      scriptCharset: "utf-8",
5      data: "check=1&timeout=" + $('#timeout').val() + "&locker_id=" +
6          $('#locker_id').val() + "&pass_hash=" + passHash,
7      success: function(isInserted) {
8          var arr = isInserted.split(",");
9          if(arr[0] == 1) { //everything's OK
10             $('#create_form #locker_id').attr('value', arr[1]);
11             $('#create_form #password').removeAttr('value', 'dummy');
12             $('#create_form').submit();
13         } else { //error occured
14             alert(arr[1]); //print error message
15         }
16     });
```

### 2.3.3.2 jQuery Mobile

jQuery Mobile ist ein weiteres Projekt der jQuery Foundation. Hierbei handelt es sich um eine Kombination aus der jQuery und der jQuery User Interface (UI) Bibliothek. Sie wurde entwickelt, um eine einfache Schnittstelle zum Erstellen der User Interfaces auf mobilen Geräten, sowie auf Desktop PC's anzubieten. [jQM13]

---

Vorteile sind eine umfassende, leicht verständliche API Dokumentation<sup>2</sup> und viele Werkzeuge auf der Homepage, die es auch Leuten mit wenig Programmierkenntnissen per Drag and Drop ermöglicht, mit jQuery Mobile zu arbeiten.

Wie auch die vorher im Kapitel 2.3.3.1 beschriebene Bibliothek, ist auch diese völlig Open Source und mit der MIT-Lizenz zu verwenden.

### 2.3.3.3 Zend

Das Zend Framework 2, wie es in “SecMes“ verwendet wird, ist ein Open Source Framework, welches eine Vielzahl an nützlichen Funktionen für PHP ab der Version 5.3 enthält. Entwickelt wurde es 2006 durch Zend Technologies, einer Firma aus Cupertino, USA. Benötigt wurde hauptsächlich die “Crypt“-Klasse, welche viele gängige Algorithmen wie SHA-256, RSA oder Diffie-Hellman implementiert hat.

### 2.3.3.4 Movable-Type

Auf der Website von Chris Veness, Movable Type Ltd, stellt der Spezialist für Datedesign und -management unter anderem seine JavaScript Methoden für AES<sup>3</sup> und SHA-256<sup>4</sup> zur allgemeinen und freien Verfügung. [Mov13] Diese werden zur clientseitigen Verschlüsselung der Nachrichten, sowie zum Bilden eines Hash-Wertes des Schließfach-Passwortes vor der Eintragung in die Datenbank verwendet.

### 2.3.3.5 XAMPP

Hierbei handelt es sich um eine Sammlung von freier Software, d.h. von Programmen unter der Verwendung der GNU General Public License. Dieses Packet beinhaltet jegliche Anwendungen und Systeme, die für die Umsetzung von “SecMes“ benötigt werden. So inkludiert es als Datenbanksystem MySQL, sowie den Webserver von Apache. Das gesamte Projekt wurde in XAMPP entwickelt und getestet. [XAM13]

## 2.4 Design und Funktion

Das Design ist schlicht gehalten, um den User nicht von den wesentlichen Funktionen der Instant Messengers abzulenken. Die verwendeten Elemente sind im Stil von mobilen Apps gehalten und stammen hauptsächlich aus der CSS-Datei von jQueryMobile.

---

<sup>2</sup>jQuery Mobile API: <http://api.jquerymobile.com>

<sup>3</sup><http://www.movable-type.co.uk/scripts/aes.html>

<sup>4</sup><http://www.movable-type.co.uk/scripts/sha256.html>

Zusammen mit der JavaScript Bibliothek von jQueryMobile kann man nun HTML Tags in sogenannte Pages aufteilen. Diese verschiedenen Seiten müssen nun nicht zwingend in unterschiedlichen HTML Dateien liegen. So definiert man einfach in einer Datei verschiedene DIV-Elemente, denen man mittels dem Attribute `page-role` Seitennamen zuweist, auf die untereinander verlinkt werden kann. Mit weiteren Flags können Funktionen wie z.B. Seitenübergänge festgelegt werden.

Diese Seiten werden intern noch in die drei gleichnamigen `data-role` Attribute `Header`, `Content` und `Footer` gegliedert, siehe Abbildung 2.4.

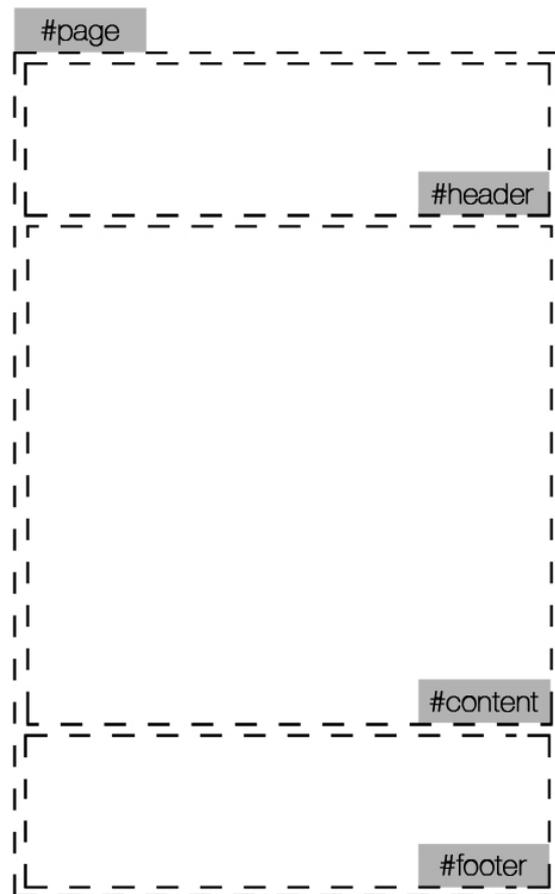


Abbildung 2.4: Seitengliederung in jQuery

“SecMes“ ist bezüglich User Interface in fünf verschiedene Bereiche aufgeteilt:

- Startseite und Schließfachanmeldung:  
 Abbildung 2.5: Auf der Startseite findet der Benutzer ein Textfeld, indem ihm kurz die Funktion von “SecMes“ erklärt wird. Darunter beginnt die Navigation, die ihm die Möglichkeit bietet ein Schließfach zu erstellen, zu öffnen oder zu verwalten. Hier erkennt man auch gleich deutlich die Trennung von Header, der den Namen der Anwendung enthält, und Content, hellgrauer Bereich mit Erklärung und Buttons.  
 Abbildung 2.6: Auf dem diesem Screenshot sieht man nun das Anmeldefenster. Also ist bereits ein Schließfach vorhanden, so kann man hier, durch die Eingabe von Schließfach-ID, Username und Passwort, auf dieses zugreifen. Im Header erscheint nun zusätzlich ein Home-Button, der auf die Startseite verweist.

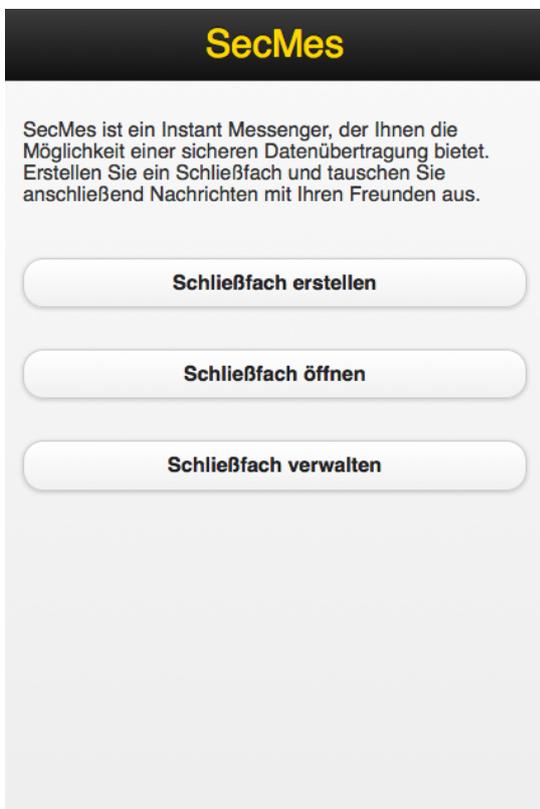


Abbildung 2.5: Startseite

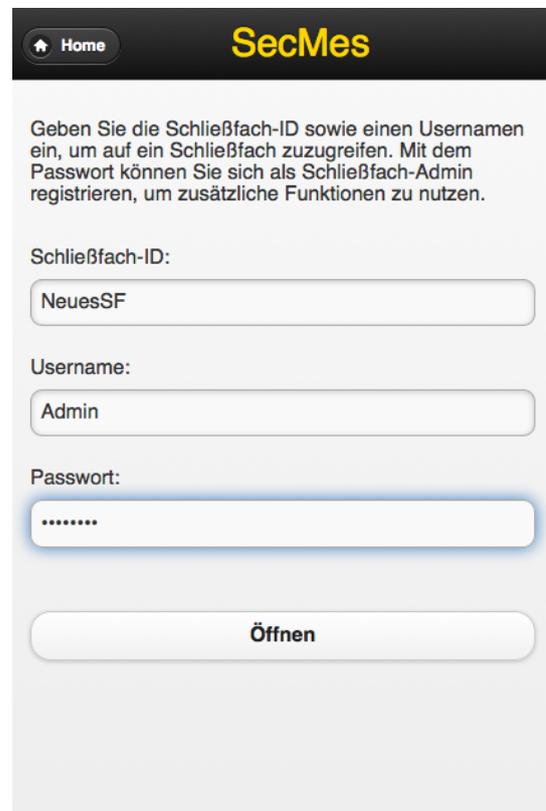


Abbildung 2.6: Schließfach öffnen

- Schließfacherstellung:  
 Abbildung 2.7: Wie bereits erwähnt, muss zuerst ein Schließfach erstellt werden. Dies geschieht hier durch die Eingabe der Schließfach-ID, einem Wert für Timeout, sowie dem Passwort. Der Benutzer hat hier für die Parameter freie Wahl, kann sich aber auch, wie in der Anwendung erklärt, eine ID generieren lassen.  
 Abbildung 2.8: Kam es zu keinen Komplikationen, das heißt die Daten wurden korrekt eingegeben und in die Datenbank geschrieben, so folgt nun dieses Bestätigungsfenster. Hier wird dem Nutzer nochmals die Schließfach-ID angezeigt, damit er sich diese auch merken oder aufschreiben kann. Mit dem Button “Zum Schließfach“ kommt der Ersteller zum Fenster in Abbildung 2.6, wo er nun auf sein Schließfach zugreifen kann.

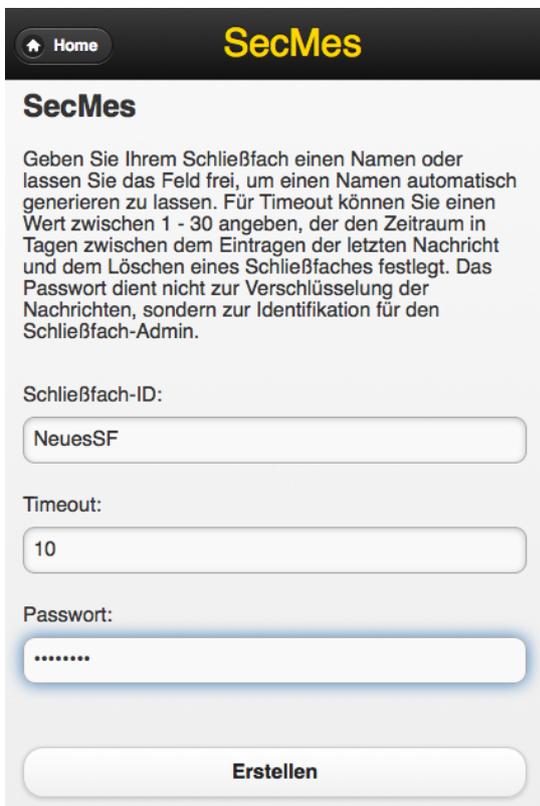


Abbildung 2.7: Schließfach erstellen

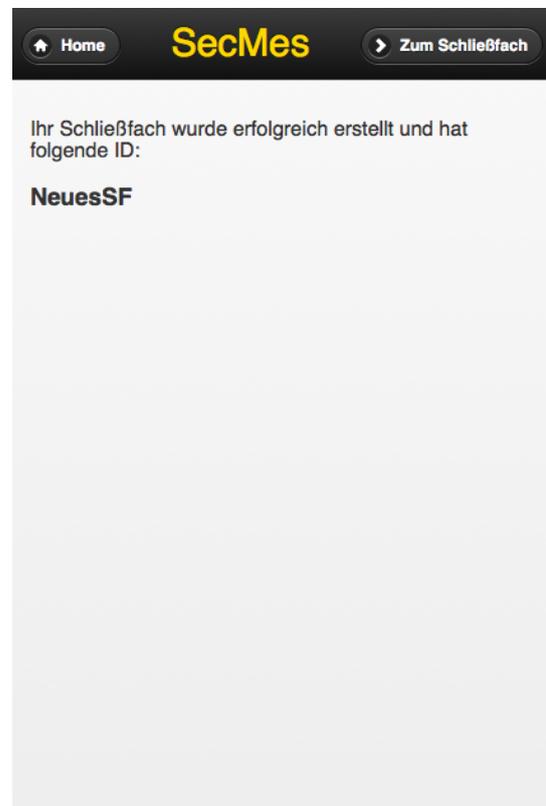


Abbildung 2.8: Schließfach erstellt

- Nachrichtenverlauf:  
 Nach der erfolgreichen Anmeldung befindet man sich nun im Schließfach. Hier sieht man ein typisches Szenario aus der Sicht von zwei verschiedenen Nutzern. Zum einen die Abbildung 2.9 aus der Sicht des Admins der den Nachrichtenschlüssel besitzt, zum anderen die eines Users ohne Schlüssel in Abbildung 2.10.  
 Hier enthält nun auch der Footer Elemente, da sich dort die Nachrichteneingabe und dessen Submit-Button befinden.

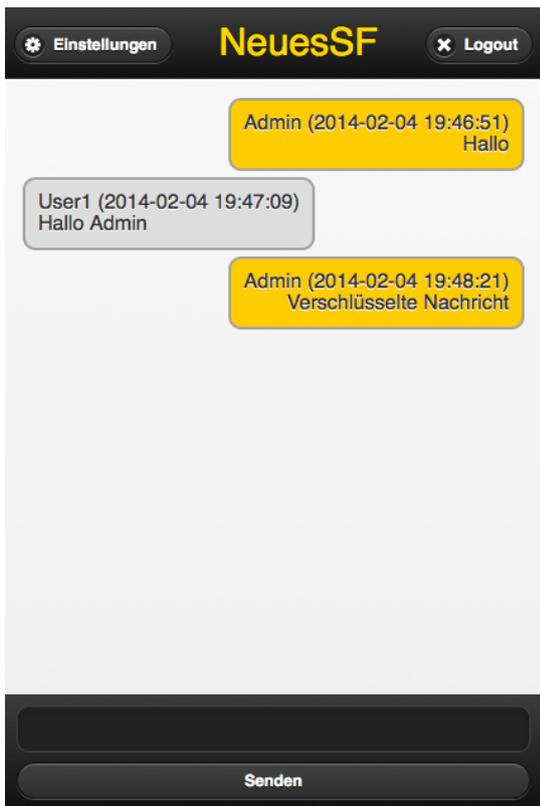


Abbildung 2.9: Entschlüsselte Nachricht



Abbildung 2.10: Verschlüsselte Nachricht

- Einstellungen:  
 Damit nun auch der User die verschlüsselten Nachrichten lesen kann, muss der Schließfach-Admin in sein Einstellungs-menü wechseln, siehe Abbildung 2.11. Hier kann er seinen Nachrichtenschlüssel festlegen, bzw. einen User aus der Liste darunter auswählen, um ihm seinen Schlüssel zu senden. (Im diesem Fall ist nur ein weiterer Benutzer außer dem Admin anwesend, User1.)

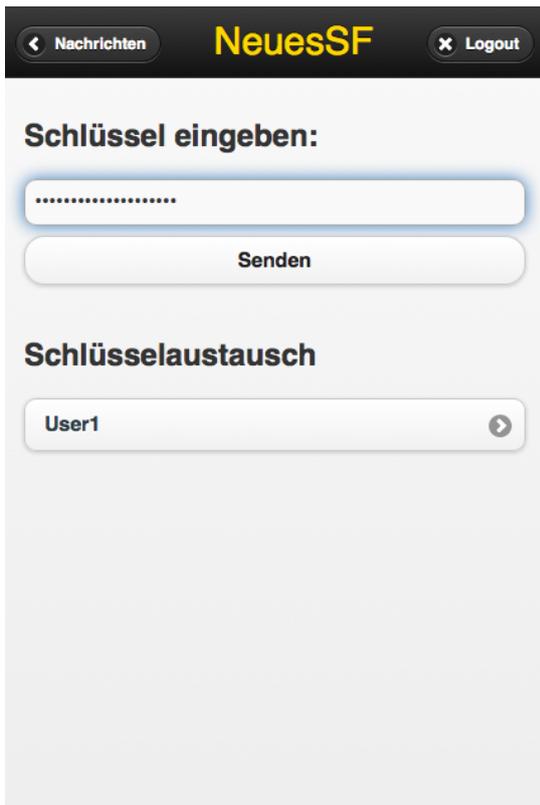


Abbildung 2.11: Einstellungen (Admin)

- Adminfunktionen:  
 Abbildung 2.12: Wählt der Benutzer im Startfenster "Schließfach verwalten", so muss er sich mit der Schließfach-ID und dem Passwort als Schließfach-Admin authentifizieren.  
 Abbildung 2.13: Bei korrekter Eingabe hat er nun Zugriff auf die Adminfunktionen "Schließfach löschen" und "Nachrichten löschen", dessen Funktionen selbsterklärend sind.

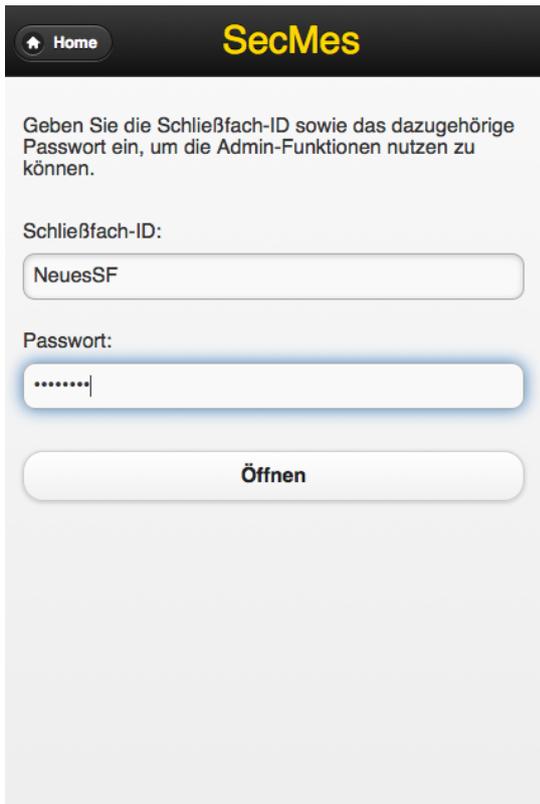


Abbildung 2.12: Adminfunktionen öffnen

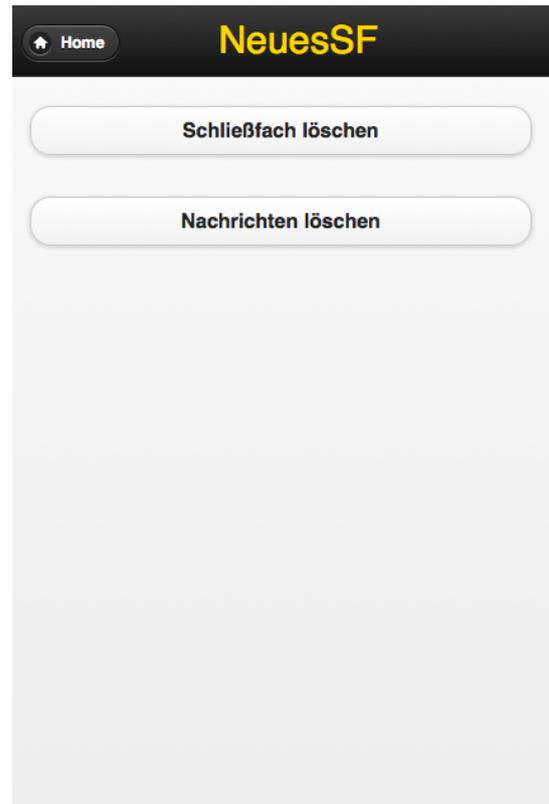


Abbildung 2.13: Adminfunktionen verwenden

---

## 2.5 Sicherheitsaspekte von SecMes

Nachdem die App nun vorgestellt wurde, bleibt die Frage: "Wodurch unterscheidet sich nun "SecMes" zu den bestehenden Systemen?" Beantwortet wird diese Frage basierend auf den Punkten in Kapitel 1.3.

- **Nachrichtenverschlüsselung:**  
Die Nachrichten werden, wie in vielen weiteren Messengern, mit einem aktuellen Verschlüsselungsstandard, nämlich AES, verschlüsselt. Der Unterschied folgt nun jedoch im Schlüsselaustausch. Hier wird dem Benutzer vergleichsweise viel Verantwortung übertragen. Bei anderen Systemen ist die Verschlüsselung für den Anwender sozusagen unsichtbar, wohingegen bei SecMes der Benutzer für die Verwendung, sowie den Austausch des Schlüssels zuständig ist. Das System bietet mit dem Diffie-Hellman Algorithmus lediglich eine Möglichkeit dafür an.
- **Anonymität:**  
Ein User existiert nur, solange er auf ein Schließfach zugreift. Hier kennzeichnet er sich durch seinen Nutzernamen aus, den er beliebig wählen kann. Das heißt er gibt in keiner Weise Informationen über seine Person preis. Des Weiteren nutzt SecMes keine Daten die lokal beim Anwender gespeichert sind. Es ist nicht nötig Kontakte über das Telefonbuch zu suchen, da man hier nicht über Personen Kontakt aufnimmt, sondern über Schließfächer.
- **Authentifizierung:**  
Eine Authentifizierung zwischen Nutzer und System findet hier nicht statt. Das System prüft bei jeder Anmeldung lediglich, ob der Nutzer, der auf ein Schließfach zugreifen will, ein Administrator ist, indem das Schließfach-Passwort abgefragt wird. Die zweite Authentifizierung muss der Schließfach-Admin selbst durchführen, da er prüfen muss, welchen Nutzern er seinen Nachrichtenschlüssel übertragen will.
- **Datenspeicherung:**  
Die Benutzer existieren, wie bereits erwähnt, nur solange sie auf ein Schließfach zugreifen. Aber auch die Schließfächer sind durch die Timeout-Funktion, sowie dem manuellen Löschen durch den Schließfach-Admin, zeitlich begrenzt in der Datenbank gespeichert.

---

## 2.6 Optimierungen

Wie in den meisten Programmen sind auch in “SecMes“ noch Optimierungen möglich. Auf zwei solcher Punkte wird hier kurz eingegangen:

- **Datenerhaltung:**  
Wie bereits in Kapitel 2.3.2.3 erklärt, funktioniert die “Speicherung“ von Daten durch ständige Übergaben von POST-Variablen.  
Dies ist trotz der Verwendung eines HTTPS Protokolls sicher ein verbesserungswürdiger Programmteil. Auch in Bezug darauf eine Sitzung (voller Zeitraum des Zugriffs auf ein Schließfach) sicherer zu machen, wäre hier vielleicht eine Verwendung von den in PHP enthaltenen Sessions eine mögliche Option. So könnte auch klarer zwischen Öffnen und Schließen eines Schließfachs am Client unterschieden werden.
- **Nachrichtenabfrage:**  
Die Nachrichtenabfrage mittels Polling findet einmal pro Sekunde statt und belastet das System dadurch nicht unwesentlich.  
Eine Variante die hier die Anzahl an Anfragen deutlich reduzieren würde, ist das sogenannte “Long Polling“. Hier wird im Vergleich zu einem normalen Polling nur eine Anfrage gesendet. Nun wird die Verbindung so lange aufrecht erhalten, bis die Abfrage, in diesem Fall eine neue Nachricht, ein positives Ergebnis zurück liefert. Somit wird nicht mehr einmal pro Sekunde, sondern einmal pro Nachricht eine Abfrage gestartet. Die Grundlegende Technologie stellt das aktuelle System bereits zur Verfügung, also mit PHP und jQuery wäre eine Umsetzung bereits möglich.

# Literaturverzeichnis

- [chi13] chip.de. WhatsApp-Rekord: 27 Milliarden Nachrichten am Tag. [http://www.chip.de/news/WhatsApp-Rekord-27-Milliarden-Nachrichten-am-Tag\\_62446082.html](http://www.chip.de/news/WhatsApp-Rekord-27-Milliarden-Nachrichten-am-Tag_62446082.html), 2013.
- [gig13] giga.de. WhatsApp: Sicherer, aber immer noch hackbar. <http://www.giga.de/apps/whatsapp-fuer-android/news/whatsapp-sicherer-aber-immer-noch-hackbar/>, 2013.
- [hei13] heise.de. Priyanka mischt WhatsApp auf. <http://www.heise.de/security/meldung/Priyanka-mischt-WhatsApp-auf-1915571.html>, 2013.
- [it13] iphone ticker.de. Hacker demonstriert: WhatsApp-Verschlüsselung unbrauchbar. <http://www.iphone-ticker.de/hacker-demonstriert-whatsapp-verschlueselung-unbrauchbar-55236/>, 2013.
- [jQM13] jQueryMobile About. <http://jquerymobile.com/about/>, 2013.
- [jQu13a] jQuery. <https://jquery.org/>, 2013.
- [jQu13b] jQuery History. <https://jquery.org/history/>, 2013.
- [jQu13c] jQuery License. <https://jquery.org/license/>, 2013.
- [mas13] mashable.com. Wickr: Can the Snapchat for Grown-Ups Save You From Spies? <http://mashable.com/2013/03/04/wickr/>, 2013.
- [Mov13] Moveable Type Ltd. <http://www.movable-type.co.uk>, 2013.
- [myw13] mywickr.com. Offizielle Webseite von Wickr. <https://www.mywickr.com/en/index.php>, 2013.
- [nt13] n tv.de. Über 300 Millionen Nutzer weltweit. <http://www.n-tv.de/technik/Whatsapp-versendet-Sprachnachrichten-article11134816.html>, 2013.

- 
- [Sit11] SitePoint MySQL Events. <http://www.sitepoint.com/working-with-mysql-events/>, 2011.
- [thr13] threema.ch. Offizielle Webseite von Threema. <https://threema.ch/de/index.html>, 2013.
- [to13] t online.de. Mehrheit fühlt sich durch NSA-Schnüffelei nicht bedroht. [http://www.t-online.de/nachrichten/specials/id\\_66315660/deutsche-fuehlen-sich-durch-nsa-abhoerskandal-nicht-bedroht.html](http://www.t-online.de/nachrichten/specials/id_66315660/deutsche-fuehlen-sich-durch-nsa-abhoerskandal-nicht-bedroht.html), 2013.
- [Wik13] Wikipedia HTTPS. [http://de.wikipedia.org/wiki/Hypertext\\_Transfer\\_Protocol\\_Secure](http://de.wikipedia.org/wiki/Hypertext_Transfer_Protocol_Secure), 2013.
- [XAM13] XAMPP. <http://www.apachefriends.org/de/xampp.html>, 2013.