technische universität
dortmund

Diplomarbeit

Ein RSA-basiertes Konzept zum Erhalt der Privatsphäre in Social Network Services

Finn Siebert

18. April 2012

Betreuer:
Dr. Karsten Klein
Prof. Dr. Petra Mutzel

Fakultät für Informatik
Algorithm Engineering (Ls11)
Technische Universität Dortmund
http://ls11-www.cs.tu-dortmund.de

Abstract

In dieser Diplomarbeit wird ein soziales Netzwerk entworfen, welches die Privatsphäre der Anwender durch asymmetrische Verschlüsselung im Webbrowser schützt. Häufig werden die etablierten sozialen Netzwerke wegen ihres Mangels an Datenschutz kritisiert. Zudem findet auch der größte Teil der Kommunikation mit Emails und Instant Messenger unverschlüsselt statt. Unter anderem deshalb, weil die Installation von Kryptografiesoftware für normale Endverbraucher zu aufwendig ist. Hier wird daher ein Netzwerk entwickelt, das ohne Installation in fast jedem Webbrowser funktioniert und die Daten vor dem Versenden verschlüsselt.

Die prototypische Umsetzung pribook.com ist wie ein herkömmliches soziales Netzwerk mit einem zentralen Server und einer zentralen Datenbank aufgebaut. Aber die HTML-Datei, die beim Aufrufen der Seite heruntergeladen wird, enthält JavaScript, der die Nachrichten und persönlichen Informationen vor dem Versenden mit RSA verschlüsselt.

Inhaltsverzeichnis

Kapitel 1

Einleitung

In diesem Kapitel wird das Thema der Diplomarbeit motiviert und die Hintergründe erläutert, bevor der Aufbau der Arbeit geschildert wird.

1.1 Motivation und Hintergrund

Soziale Netzwerke sind gerade eines der populärsten Themen in der gesellschaftlichen Debatte mit einem Bezug zur Informatik. Oft wird die einfache Möglichkeit der Vernetzung gelobt, die zum Beispiel bei den demokratischen Bewegungen in Nordafrika eine Rolle gespielt haben soll. Dieser Gedanke wird im Abschnitt 1.1.1 weiter vertieft.

Zudem sind in den westlichen Ländern viele Menschen zwar bei den sozialen Netzwerken angemeldet, haben aber durchaus ein kritisches Bewusstsein dafür, dass ihre Daten für Werbezwecke missbraucht werden können und möglicherweise von Unbefugten einsehbar sind. In Abschnitt 1.1.2 wird daher die Problematik des Datenschutzes in sozialen Netzwerken genauer betrachtet.

Auch unabhängig von sozialen Netzwerken wird der Datenschutz zum Beispiel durch die Arbeit von Geheimdiensten immer wieder gefährdet. Dennoch können sich Techniken für den Datenschutz bei den Endverbrauchern nicht durchsetzten. Ein großes Problem der Kryptografie ist es, dass durchschnittliche Endbenutzer meistens daran scheitern Clients für PGP oder Ähnliches zu installieren und zu benutzen. So sind heute immer noch der größte Teil der E-Mails unverschlüsselt (Abschnitt 1.1.3).

Aus dieser Situation wird die Idee dieser Diplomarbeit abgeleitet, ein soziales Netzwerk zu entwickeln, welches für die Nutzer asymmetrische Verschlüsselung, möglichst einfach

zu bedienen und möglichst transparent, im Webbrowser realisiert. Um so möglichst vielen Menschen eine sichere Kommunikation zu ermöglichen. Der Aufbau der Diplomarbeit wird im Abschnitt 1.2 vorgestellt.

1.1.1 Soziale Netzwerke

Es vergeht kaum ein Tag, an dem nicht soziale Netzwerke in den Schlagzeilen der Medien stehen. Sie gehören mittlerweile zu den meistgenutzten IT-Anwendungen.

Der Branchenprimus unter den Netzwerken Facebook vermeldete am 23. September 2011 die Marke von 800 Millionen aktiven Nutzern geknackt zu haben [13]. Laut [13] melden sich „Rund 800 Millionen Mitglieder aus aller Welt“ mindestens einmal im Monat bei Facebook an. Alleine in Deutschland würden „knapp über 20 Millionen Anwender Facebook regelmäßig nutzen“. Im Schnitt würden alle Nutzer zusammen an einem Tag „mehr als 200 Millionen Fotos“ auf Facebook hochladen. Dies entspricht in nur einem Monat mehr als 6 Milliarden Bildern. Etwas konservativer schätzt Facebook die eigenen Nutzerzahlen im Emissionsprospekt für einen möglichen Börsengang ein, der am 1. Februar 2012 veröffentlicht wurde. Dort wird von „432 Millionen monatlich aktiven Usern“[29] gesprochen. Wobei „ungefähr fünf bis sechs Prozent aller Profile“ von Nutzern erstellt seine könnten, die bereits Profile haben.

Auch das soziale Netzwerk von Google, Google Plus hat nach eigenen Angaben bereits 100 Millionen Nutzer [10], die sich mindestens einmal im Monat einloggen und 50 Millionen, die sich sogar täglich anmelden. Auch viele weitere soziale Netzwerke haben viele Zugriffe. Exemplarisch lässt sich dies mit den Zugriffszahlen vom Dezember 2010 belegen, die von der COMPASS HEADING GmbH ermittelt wurden. So wurde die Seite Wer-kennt-Wen.de allein in Deutschland im Dezember 2010 5,1 Millionen mal besucht. Gefolgt von dem Netzwerk Stayfriends.de mit 3,8 Millionen Zugriffen und SchuelerVZ mit 3,5 Millionen. Es folgen die sozialen Netzwerke MeinVZ (3,9 Millionen), StudiVZ (2,9), Twitter.com(2,9), Xing.com(2,4) und linkedIn mit 0,76 Millionen Zugriffen aus Deutschland im Dezember 2010.

Auch die Zeit, die Nutzer mit sozialen Netzwerken verbringen ist enorm. Die Marktforschungsfirma comScore schätzt, dass im Schnitt „11,7 Prozent der gesamten Onlinezeit“ bei Facebook verbracht wird [13]. Insgesamt summiere sich die Nutzung von Facebook auf über 70 Milliarden online Minuten [13]. Das Wallstreet Journal veröffentlichte eine Studie von comScore der zufolge Facebook-Mitglieder im Schnitt 7,5 Stunden pro Monat auf Facebook verbringen [23].

Diese enormen Nutzerzahlen und die intensive Nutzung führen zu hohen Umsätzen und hohen Erwartungen. So konnte Facebook im Jahr 2011 3,71 Milliarden Dollar umsetzten [5] und bei einem Börsengang könnte das Unternehmen mit 75-100 Milliarden Dollar bewertet werden [5]. Doch nicht nur wegen der großen Verbreitung oder aus wirtschaftlicher Sicht macht eine Beschäftigung mit dem Thema soziale Netzwerke Sinn. Auch im gesellschaftlichen und politischen Bereich können soziale Netzwerke eine wichtige Aufgabe wahrnehmen.

So berichtet der tunesische Onlineaktivist Foetus: „Facebook war so etwas wie das Navigationssystem für diese Revolution“ [51]. Und beschreibt damit, die wichtige Rolle, die soziale Netzwerke bei der arabischen Revolution in Tunesien und den anderen arabischen Ländern eingenommen haben. Ohne die Straße gebe es zwar keine Revolution, aber wenn man Facebook dazu nehme, bekomme man „echtes Potenzial“ so Foetus weiter. Diese Einschätzung wird auch von Zenyep Tufekci geteilt, die Junior-Professorin an der University of North Carolina und Fellow am Berkman Center for Internet and Society der Harvard University ist. „Diese neuen, technischen Möglichkeiten erzeugen keine Opposition. Die Opposition war immer da. Aber sie haben den Unzufriedenen ermöglicht, sich auf eine neue Art und Weise zu organisieren“ [5], weiß sie über die Rolle von sozialen Netzwerken zu berichten.

1.1.2 Datenschutz in sozialen Netzwerken

Im vorigen Abschnitt wurde beschrieben, dass soziale Netzwerke zu den häufig genutzten IT-Anwendungen gehören, dass sie wirtschaftlich interessant sind und dass sie sogar wichtige gesellschaftliche Funktionen erfüllen können. In diesem Abschnitt wird nun gezeigt, dass es im Zusammenhang mit den sozialen Netzwerken aber ein eklatantes Problem gibt, nämlich den Datenschutz. Oben wurde geschrieben, dass kaum ein Tag vergeht, an dem nicht soziale Netzwerke in den Schlagzeilen stehen. Oftmals ist der Anlass dafür der mangelnde Datenschutz. So schreibt die Stiftung Warentest über Facebook:

„Facebook ist als größtes soziales Netzwerk eine gewaltige Datensammelmaschine. Die Daten über Freundschaften, Vorlieben und Verbindungen der Nutzer machen das Netzwerk ja gerade aus. Klar ist aber auch: Mit diesen Daten verdient Facebook Geld und nutzt sie für Werbezwecke.“ [43]

Welche Daten Facebook genau erhebt, ist ein Geschäftsgeheimnis. Neben den Daten, die vom Nutzer direkt eingegeben werden, schätzt die Stiftung Warentest, dass das Unternehmen aus San Francisco auch die folgenden Daten erfasst:

„ - die wichtigsten Daten des Computersystems, von dem aus ein Nutzer sich ins Netzwerk einloggt – bis hin zu IP-Adresse, Prozessortyp und Browserversion samt Plug-Ins und
- jeden Besuch auf Webseiten, auf denen ein „Gefällt mir"-Knopf installiert ist. Dabei wird auch die IP-Adresse übertragen. Sofern der Besucher der fremden Seite Facebook-Nutzer ist und sich vom selben Computer aus bereits bei Facebook eingeloggt hat, kann Facebook auch dessen Identität ermitteln."

Das unabhängige Landeszentrum für Datenschutz in Schleswig-Holstein (ULD) kommt sogar zu der Einschätzung, dass diese Datenauswertung gerade der „Gefällt mir"-Knöpfe gegen geltendes Recht verstößt. So schreibt das Landeszentrum in einer Pressemitteilung: „Nach eingehender technischer und rechtlicher Analyse kommt das ULD zu dem Ergebnis, dass derartige Angebote gegen das Telemediengesetz (TMG) und gegen das Bundesdatenschutzgesetz (BDSG) bzw. das Landesdatenschutzgesetz Schleswig-Holstein (LDSG SH) verstoßen."[45]

Doch nicht nur gegen den „Gefällt mir"-Knopf regt sich der Widerstand. Auch der sogenannte Freundesfinder, eine Funktion bei der Nutzer ihre privaten E-Mail-Adressbücher an Facebook übermitteln, erntet Kritik. Das Landgericht Berlin entschied am 06.03.2012:

„Auf Klage des Bundesverbandes der Verbraucherzentralen und Verbraucherverbände hat das Landgericht Berlin heute der Facebook Ireland Limited die Versendung entsprechender Anfragen an Dritte und die Verwendung eines unzureichenden Hinweises auf Datenimport bei der Registrierung sowie die Verwendung verschiedener Vertragsklauseln untersagt. Nach Auffassung des Landgerichts sind die entsprechende Werbepraxis von Facebook und die verwendeten Klauseln mit wettbewerbsrechtlichen Grundsätzen sowie den Regeln über allgemeine Geschäftsbedingungen nicht vereinbar." [32]

Datenschützer gehen davon aus, dass sich diese Datenschutzproblematik bei Facebook durch den erwarteten Börsengang in diesem Jahr weiter zuspitzt. Johannes Casper, Datenschutzbeauftrager von Hamburg geht davon aus, dass der Druck der Aktionäre wachsen und die Entscheidungen des Unternehmens möglicherweise zulasten des Datenschutzes beeinflussen werde. „Es steht zu befürchten, dass der Fokus in Zukunft auf Gewinnmaximierung gerichtet wird".[53]

Auch die meisten anderen sozialen Netzwerke stehen in der Kritik von Datenschützern. Exemplarisch soll dies hier am Beispiel Google belegt werden. Der größte Konkurrent von Facebook, Google mit seinem Netzwerk Google + erntete viel Gegenwehr, weil dieser zum 01. März die eigenen Datenschutzrichtlinien änderte. Die Verbraucherschutzgruppe EPIC reichte Klage gegen die Änderung ein. Mit der Begründung, dass Google so noch mehr

Daten über das Nutzerverhalten sammle und miteinander vernetze ohne das sich Google-Nutzer dagegen wehren könnten.[25]

Es gibt noch viele weitere Kritikpunkte am Datenschutz in sozialen Netzwerken, die hier nicht alle aufgelistet werden können. Es ist jedoch bereits so deutlich, dass der Datenschutz das größte Problem für soziale Netzwerke darstellt.

1.1.3 Datenschutz Allgemein

Nachdem in den oberen beiden Abschnitten dargelegt wurde, wie wichtig soziale Netzwerke sind und dass der Datenschutz das größte Problem für sie darstellt, sollen in diesem Abschnitt die allgemeinen Probleme des Datenschutzes angesprochen werden.

Nicht nur bei sozialen Netzwerken, sondern auch allgemein wird häufig ein mangelnder Datenschutz beklagt. Als Beispiel sei hier angeführt, dass am 25. Februar bekannt wurde, dass in Deutschland Geheimdienste im Jahr 2010 mehr als 37.000.000 Netzverbindungen überwacht haben, weil in Ihnen Schlagwörter wie „Bombe“ vorkämen [30]. Das Parlamentarische Kontrollgremium des Bundestages schreibt in seinem Bericht [15], dass die Kommunikation nach 16400 Begriffen durchsucht wurde. Wie viele Netzverbindungen insgesamt nach diesen Begriffen durchsucht wurden, ist nicht bekannt. Da es aber keine Aussagen dazugibt, muss davon ausgegangen werden, dass potenziell alle Verbindungen zumindest automatisch überwacht werden. Aus der Sicht des Autors dieser Arbeit ist es sehr fraglich, ob diese Praxis der Geheimdienste nicht gegen das Grundrecht auf informationelle Selbstbestimmung verstößt. Dieses Grundrecht wurde 1971 mit dem Volkszählungsurteil vom Bundesverfassungsgericht wie folgt begründet:

„Mit dem Recht auf informationelle Selbstbestimmung wären eine Gesellschaftsordnung und eine diese ermöglichende Rechtsordnung nicht vereinbar, in der Bürger nicht mehr wissen können, wer was wann und bei welcher Gelegenheit über sie weiß. Wer unsicher ist, ob abweichende Verhaltensweisen jederzeit notiert und als Information dauerhaft gespeichert, verwendet oder weitergegeben werden, wird versuchen, nicht durch solche Verhaltensweisen aufzufallen.[...] Dies würde nicht nur die individuellen Entfaltungschancen des Einzelnen beeinträchtigen, sondern auch das Gemeinwohl, weil Selbstbestimmung eine elementare Funktionsbedingung eines auf Handlungsfähigkeit und Mitwirkungsfähigkeit seiner Bürger begründeten freiheitlichen demokratischen Gemeinwesens ist. Hieraus folgt: Freie Entfaltung der Persönlichkeit setzt unter den modernen Bedingungen der Datenverarbeitung den Schutz des Einzelnen gegen unbegrenzte Erhebung, Speicherung, Verwendung und Weitergabe seiner persönlichen Daten voraus. Dieser Schutz ist daher von dem Grundrecht des Art. 2 Abs. 1 in Verbindung mit Art. 1 Abs. 1 GG umfasst. Das Grundrecht gewähr-

leistet insoweit die Befugnis des Einzelnen, grundsätzlich selbst über die Preisgabe und Verwendung seiner persönlichen Daten zu bestimmen.“ [8]

Im Urteil des BVG vom 2.März 2010 zur konkreten Ausgestaltung der Vorratsdatenspeicherung wird das Recht auf informationelle Selbstbestimmung weiter präzisiert. So lautet es in der Urteilsbegründung: „Zumal die Speicherung und Datenverwendung nicht bemerkt werden, ist die anlasslose Speicherung von Telekommunikationsverkehrsdaten geeignet, ein diffus bedrohliches Gefühl des Beobachtetseins hervorzurufen, das eine unbefangene Wahrnehmung der Grundrechte in vielen Bereichen beeinträchtigen kann.“ [9]

Dennoch gibt es Techniken und Programme, die die Menschen für den Datenschutz einsetzen könnten. Programme wie PGP oder GnuPG schützen den E-Mail-Verkehr vor Überwachung des Staates oder Netzbetreibers. Für Instantmessenger gibt es Off-the-Record (OTR) Messaging und für Voice-over-IP gibt es SRTP. Die Frage, die sich aufgrund dieser Situation stellt, ist warum diese Technologien nur in so geringem Maße verwendet werden.

Laut einem Artikel von 2001 benutzten weniger als 2,5% der Internetnutzer PGP[3]. Leider konnte keine neuere Zahl gefunden werden, aber der Autor dieser Arbeit geht davon aus, dass der Anteil heute eher deutlich geringer geworden ist. Im selben Artikel erläutert Phillip Zimmermann, der Erfinder von PGP, warum so wenige Menschen PGP verwenden: „Wir haben Probleme, PGP in der Breite der Gesellschaft zu verankern. Das Bewusstsein für den Schutz der Privatsphäre muss erhöht werden, aber ein wichtiger Faktor ist mit Sicherheit auch, wie einfach man es benutzen kann“.

Das Problem ist also, dass Techniken zum Datenschutz für normale Endverbraucher zu aufwendig zu betreiben sind. Schon die Installation von zusätzlichen Programmen auf dem eigenen Computer ist für viele Verbraucher zu umständlich. Hinzu kommt, dass für eine sichere Kommunikation alle Teilnehmer zusätzliche Software installieren müssen, was gerade für eine Gruppe von normalen Endverbrauchern eine nahezu unerfüllbare Voraussetzung ist. Ein weiterer wichtiger Faktor ist, dass die Unternehmen wie Facebook und Google ihre Gewinne mit den Nutzerdaten machen. Sie wären sehr leicht in der Position, die Kommunikation ihrer Kunden durch Verschlüsselung zu schützen. Da sie aber den Profit aus dem Zugriff auf diese Daten machen, steht für sie der Datenschutz der Nutzer im Konflikt zu ihren Gewinninteressen.

1.2 Aufbau der Arbeit

Nachdem in diesem ersten Kapitel das Thema der Arbeit motiviert und der Aufbau der Arbeit vorgestellt wird, geht es in Kapitel 2 um Ansätze, die ähnlich sind wie das in dieser

Diplomarbeit entwickelte System. Dort werden andere Projekte vorgestellt, die sich mit Datenschutz in sozialen Netzwerken beschäftigen. In Kapitel 3 werden die Grundlagen, die für die Realisierung des Projektes erforderlich sind, vorgestellt. Dies sind die Grundlagen der Kryptografie sowie eine genaue Beschreibung der Algorithmen AES, RSA und SHA.

In Kapitel 4 wird thematisiert, wie ein sicheres soziales Netzwerk aussehen kann. Zunächst werden Invarianten aufgestellt, die ein sicheres soziales Netzwerk einhalten muss und dann wird beschrieben, wie die einzelnen Elemente des Netzwerkes realisiert werden könnten. Pribook.com ist die Realisierung eines solchen sicheren sozialen Netzwerks. In Kapitel 5 wird der Aufbau von Pribook geschildert, sowie der Aufbau der Datenbank und die Realisierung der verschiedenen Anwendungsfälle.

Das 6. Kapitel ist ein Handbuch für die Nutzung von Pribook. Hier steht, wie die Implementierung für den Endbenutzer aussieht und wie die einzelnen Funktionen bedient werden. Aufbau, Ergebnisse und Interpretation eines empirischem Test von Pribook befindet sich in Kapitel 7. Schließlich wird in Kapitel 8 ein Fazit gezogen und ein Ausblick für weitere Arbeiten gegeben.

Kapitel 2

Ähnliche Ansätze

2.1 Übersicht

Es gibt eine ganze Reihe von Ansätzen um den Datenschutz in sozialen Netzwerken zu erhöhen. Viele Projekte und Implementierungen befassen sich mit der Problematik der Privatsphäre. Auf der Seite [21] wird ein Überblick über viele Projekte gegeben. In dem Artikel [2] werden verschiedene Alternativen zu den herkömmlichen Ansätzen vorgestellt. Im Groben lassen sich die Projekte in mehrere Kategorien einteilen.

2.1.1 Open Source

Projekte wie Buddypress [7] oder Crabgrass [11] setzten auf Open Source zu Verbesserung der Privatsphäre. Für beide Projekte ist der Quellcode der Software veröffentlicht und das sowohl für die Client- als auch für die Serverseite. So sind zum einen alle Prozesse bekannt, und die Nutzer müssen nicht befürchten, dass Daten im Hintergrund erhoben und missbraucht werden. Zum anderen kann jeder einen eigenen Server mit seinem eigenen sozialen Netzwerk aufsetzen. Die Software selbst enthält keine extra Verschlüsselung zum Schutz der Daten, aber auf den Servern, auf denen diese sozialen Netzwerke laufen, kann SSL eingesetzt werden, um so eine Server-Client-Verschlüsselung zu realisieren. Auch können auf dem Server bekannte Technologien eingesetzt werden, um die Daten auf den Festplatten zu verschlüsseln.

Zwar wird durch diese Art, der Datenschutz gegenüber den herkömmlichen Netzwerken deutlich erhöht, aber diese Implementierungen haben auch einige Nachteile. Immer noch kann ein Missbrauch der Daten durch den Betreiber des Servers nicht ausgeschlossen wer-

den und die Netzwerke stehen jeweils für sich alleine und kommen so auf keine sehr große Nutzerzahlen. Diese Art von Netzwerken ist also vor allem für Organisationen und feste Gruppen sinnvoll die gemeinschaftlich ein soziales Netzwerk nutzen wollen.

2.1.2 Verteilte Netzwerke

Die sogenannten verteilten Netzwerke wie Diaspora[16], friendica[22] oder Buddycloud[6] sind in der Regel auch Open Source. Im Unterschied zu Cabgrass und Buddypress ist aber vorgesehen, dass alle Server miteinander Kommunizieren und Nutzer, die bei verschiedenen Servern angemeldet sind, miteinander Nachrichten austauschen können. Als Beispiel für ein verteiltes System wird im Abschnitt 2.2 Diaspora vorgestellt. Da Diaspora auch eine Open Source Projekt ist, wird auf das Vorstellen eines Projektes aus der oben beschriebenen Open Source-Kategorie verzichtet.

2.1.3 Peer-to-Peer-Basierte Netzwerke

Bei den Peer-to-Peer-basierten Ansätze wie Safebook[40], Peerscape[35] oder Jappix[27] ist jeder Nutzer auch zugleich ein Knoten im Netzwerk. Auf zusätzliche Server wird komplett verzichtet. Dieser Ansatz hat den Vorteil, dass es in diesem Modell keinen Serverbetreiber mehr gibt, der die Daten potenziell missbrauchen könnte. Allerdings werden die Nachrichten im Klartext versendet, sodass sie zum Beispiel nicht gegen die Überwachung der der deutschen Geheimdienste geschützt sind (siehe Abschnitt 1.1.3). Bei einigen der Implementierungen ist es zudem nötig, dass sich Anwender extra Software zur Benutzung installieren. Dies ist gerade für soziale Netzwerke ein Hemmnis, da sie gerade von vielen Nutzern profitieren.

2.1.4 Sonstige

Das Netzwerk Secure Share[42] ist ein Framework für sicheren Chat und Dateienaustausch. Leider hat es den Nachteil, dass es auch die Installation von Software erfordert. Hushmail[26] ist ein Webmailclient, der Verschlüsselung im Webbrowser realisiert. Die Idee ist sehr ähnlich wie die in dieser Arbeit umgesetzte.

2.2 Diaspora - Beispiel für ein verteiltes Netzwerk

Als Diaspora oder Diaspora* wird die Software für das gleichnamige soziale Netzwerk bezeichnet. Die Software wurde in Ruby geschrieben und unter AGPL veröffentlicht. Derzeit sind laut Schätzungen ca. 370.000 Nutzer bei Diaspora aktiv [17]. Der Artikel [47] bietet eine gute Übersicht, darum lehnt die folgende Beschreibung an den Text an.

2.2.1 Idee

Diaspora unterscheidet sich von herkömmlichen sozialen Netzwerken aber nicht nur darin, dass der Quellcode veröffentlicht ist, sondern auch dadurch, dass es als verteiltes System aufgebaut ist. Es will dieselben Funktionen wie Facebook bieten, also schwarze Bretter für zeitversetzte Kommunikation, Chatfenster für Echtzeitkommuniktion und Schnittstellen, damit Drittanbieter zusätzliche Anwendungen programmieren können (Plug-Ins).

Jeder Anwender kann selbst einen Server, einen sogenannten Pod aufsetzten, auf dem seine privaten Daten liegen. Er hat aber auch die Möglichkeit sich bei einem bestehenden Pod anzumelden. Damit Nutzer von verschieden Pod's miteinander kommunizieren können, sind Nutzernamen aufgebaut wie eine E-Mail-Adresse oder eine Jabber-ID. Sie bestehen jeweils aus einem frei wählbaren, aber je Pod eindeutigem Benutzernamen gefolgt von der URL des Pods. Ein Beispiel wäre max.mustermann@geraspora.net.

2.2.2 Realisierung

Das Projekt wurde am 24. April 2010 von den vier Mathematik Studenten an der Universität von New York, Dan Grippi, Maxwell Salzberg, Rapheall Sofaer, Ilya Zhitomersky auf der Plattform Kickstarter vorgestellt. Es erlangte viel aufsehen, weil das Spendenziel von 10.000 Dollar weit überschritten wurde und letztendlich mehr als 200.000 Dollar zusammenkamen. Außerdem befand sich unter den Spendern auch der Erfinder von Facebook Mark Zuckerberg, der das Projekt als „coole Idee“ bezeichnete.

Der Quellcode des Projektes im Alphastadium wurde am 23. November 2010 veröffentlicht. Seit dem läuft auch eine Alphaversion von Diaspora. Im November 2011 wurde eine neue Version mit vergrößertem Funktionsumfang veröffentlicht darunter „Hashtag-Follow-Funktion, Direkt-Nachrichten, Like-Buttons für Status-Updates und ein Benachrichtigungs-System“ [14].

Wie oben bereits geschrieben will Diaspora die Daten schützen durch die Idee das Netzwerk als dezentrales verteiltes System zu realisieren. Jedem Nutzer ist es möglich, sich die Software von Diaspora zu kopieren und auf einem eigenen Server zum Laufen zu bringen und dann mit allen anderen Nutzern zu kommunizieren. Die Nutzer können selbst bestimmen, in welcher Form ihre Daten an andere Server weitergegeben werden. Zwischen den Servern werden die Daten mit GPG verschlüsselt und vom Server zum Client besteht eine SSL-Verbindung.

2.2.3 Nachteile

Theoretisch kann jeder Nutzer seinen eigenen Server aufbauen, in der Praxis ist dies aber wohl viel zu aufwendig. Laut [2] müssten, bevor auf einem Server Diaspora installiert werden kann, die folgenden Programme installiert sein: Ruby, SQLite3, OpenSSL, libcurl, ImageMagick, Git und Redis. Deshalb seien die meisten Benutzer nach wie vor darauf angewiesen, sich bei einem bestehenden Pod anzumelden und dem Betreiber dieses Pods zu vertrauen.

Doch selbst wenn die Installation deutlich einfacher wäre, so schließt eine Installation von Software immer viele Menschen aus, wie bereits in Kapitel 1 thematisiert wurde (Abschnitt 1.1.3). Auch stellt das Betreiben eines eigenen Servers, der über eine Standleitung verfügt, eine sehr hohe Hürde da. In der Praxis werden hier wohl Server von „Shared Webhosting“-Firmen gemietet, was bedeutet, dass auch diese Firmen theoretisch Zugriff auf die privaten Nutzerdaten haben. Ein durchschnittlicher Endverbraucher muss also darauf vertrauen, dass weder der Betreiber des eigenen Pods noch die Firma die den Server zur Verfügung stellt, die Daten missbraucht. Hinzukommen die Firmen und Betreiber von anderen Pods, wenn ein Nutzer auch mit Nutzern von anderen Pods kommuniziert. In der Praxis kann ein Endverbraucher also nicht zwingend davon ausgehen, dass seine persönlichen Daten bei Diaspora besser geschützt sind als bei einem der großen Anbieter wie Facebook oder Google.

2.3 Safebook - Beispiel für ein Peer-to-Peer-Netzwerk

In [12] wird Safebook als dezentrales Peer-to-Peer basiertes soziales Netzwerk vorgestellt. Jeder Nutzer ist gleichzeitig auch ein Knoten im Peer-to-Peer- Netzwerk, auf dem das soziale Netzwerk aufsetzt. Ein einzelner Nutzer ist von Ringen von Freunden umgeben, die seine privaten Informationen in verschlüsselter Form gespeichert haben. So werden nicht

nur die Inhalte der Kommunikation verborgen, sondern es wird auch vermieden, dass zum Beispiel Kommunikationsgraphen erstellt werden.

2.3.1 Idee

Safebook besteht aus drei Hauptkomponenten. Die erste Komponente ist der identification service zur Authentifizierung von Anwendern. Die zweite Komponente sind die konzentrischen Ringe um jeden Anwender - die sogenannten Matroschkas und die dritte Komponente ist das Peer-to-Peer-Netzwerk.

In Safebook hat jeder Nutzer zwei Identitäten, eine für seinen Knoten im Peer-to-Peer Netzwerk und eine für das soziale Netzwerk. Die Identität für den Peer-to-Peer Knoten wird nicht geschützt und ist den anderen Knoten im Peer-to-Peer Netzwerk ersichtlich. Der identification service ist eine zentrale Instanz, die den Nutzern bei ihrer Registrierung diese eindeutigen Identitäten zuweist. Trotz dieses zentralen Elements ist die Privatsphäre aber nicht gefährdet, da nur bei der Registrierung Kontakt zu dieser Stelle aufgenommen werden muss.

Die Matroschkas sind nach folgendem Schema aufgebaut. Der innerste Ring besteht aus Freunden, denen der Anwender vertraut. Bei ihnen sind die privaten Daten des Anwenders in verschlüsselter Form gespeichert und sie werden als mirrors bezeichnet. Die Freunde dieser Freunde bilden den 2. Ring um den Anwender und so weiter. Je nach Konfiguration ist ein Anwender von unterschiedlich vielen Ringen von Freunden umgeben. Die Knoten im äußersten Ring werden als enterypoints bezeichnet. Sie haben eine Information über die Identität des Nutzers gespeichert, und wie sie diesen über die Ringe erreichen können. Bei Echtzeitkommunikation werden die Nachrichten bis zum Anwender in der Mitte der Matroschka geleitet bei zeitversetzter Kommunikation reicht es, können die Nachrichten auch bei den mirrors gespeichert werden, falls der Knoten in der Mitte gerade offline ist.

2.3.2 Realisierung

Es gibt einen Client für Safebook, der von der Seite [40] heruntergeladen werden kann. Der Client ist in Python geschrieben und setzt sich aus vier Teilprogrammen zusammen. Dem Communication-Manager, verantwortlich für das Senden und Empfangen von Netzwerk-Paketen, dem Peer-to-Peer-Manager, der den Aufbau des Peer-to-Peer-Netzwerkes übernimmt, dem Matroschka-Manager, der für einen Anwender seiner Ringe von Freunden verwaltet und dem Benutzer-Manager, der die Benutzeroberfläche realisiert.

Dieser Prototype von Safebook stellt vorerst die folgenden Funktionen zur Verfügung:

- Registrierung
- Veröffentlichung von Informationen
- Einsehen von Informationen
- Kontaktanfragen und das bestätigen von Kontaktanfragen
- Nachrichtenverwaltung

2.3.3 Nachteile

Ein Nachteil dieses Ansatzes ist es, dass ein zusätzlicher Client installiert werden muss. Auch gibt der Autor zu erkennen, dass ein Peer-to-Peer basiertes Netzwerk zu langsam für praktische Anwendungen ist.

Kapitel 3

Grundlagen

In diesem Kapitel werden die Grundlagen erläutert, die für die Realisierung eines sicheren sozialen Netzwerkes notwendig sind, so wie es in den darauf folgenden Kapiteln entwickelt wird. Zunächst gibt es eine allgemeine Übersicht über die Kryptografie, dann folgen einzelne Betrachtungen der später verwendeten Verfahren RSA, AES und SHA.

3.1 Kryptographie

Kryptografie bezeichnet die Wissenschaft der Verschlüsselung beziehungsweise die Konzeption, Definition und Konstruktion von Informationssystemen, die unbefugtes Lesen und Verändern möglichst verhindern wollen. Als Schutzziele für Informationssysteme wird üblicherweise die Abwehr der folgenden drei Gefahren genannt:

1. Unbefugter Informationsgewinn, d. h. Verlust der Vertraulichkeit (confidentiality)
2. Unbefugte Modifikation von Informationen, d. h. Verlust der Integrität (integrity)
3. Unbefugte Beeinträchtigung der Funktionalität, d. h. Verlust der Verfügbarkeit (availability). [36]

Angewandt auf das Beispiel soziale Netzwerke können diese drei Gefahren mit den folgenden Beispielen verdeutlicht werden:

1. Clara, eine Mitarbeiterin beim Betreiber des sozialen Netzwerkes liest private Nachrichten zwischen Alice und Bob. Oder die Regierung eines autoritären Regimes bittet

oder zwingt den Netzwerkbetreiber zur Herausgabe von Nachrichten zwischen Oppositionellen, die sich zu Demonstrationen und Protesten verabredet haben.

2. Clara verändert die private Nachricht zwischen Alice und Bob. Sie fügt Beleidigungen in den Text ein, um Streit zwischen Alice und Bob zu produzieren. Oder das Regime verbreitet falsche Nachrichten, um eine Koordinierung der Oppositionellen zu erschweren und Zwietracht zu sähen.

3. Clara verhindert, dass Alice Nachrichten an Bob schickt. Im Beispiel des autoritären Regimes: An den Routern wird die IP-Adresse des sozialen Netzwerkes gesperrt, wie es zum Beispiel in China häufig vorkommt oder es wird sogar das ganze Internet ausgeschaltet, so wie in Ägypten in Januar 2011 geschehen.

Da im Falle eines sozialen Netzwerkes ein physischer Schutz des Systems nahezu unmöglich ist, kann Kryptografie eingesetzt werden, um die Schutzziele Vertraulichkeit und Integrität zu gewährleisten.

Zum Schutz der Vertraulichkeit kann ein sogenanntes Konzelationssystem (auch Verschlüsselungssystem oder Kryptosystem) eingesetzt werden. Allerdings schützen solche Systeme in der Regel nur die Vertraulichkeit der Inhalte von Nachrichten, nicht aber die Vertraulichkeit der Kommunikationsumstände. Also zum Beispiel bleibt ungeschützt, wer die Nachricht versendet, wer sie empfängt und wann sie versendet wird. Auch lassen sich in der Regel aus dem Chiffrat Rückschlüsse auf die Länge des Klartextes schließen.

Konzelationssysteme lassen sich nach Sicherheitsgrad und Schlüsselverteilung bewerten. Grundsätzlich gibt es mit dem One-Time-Pad ein Verschlüsselungsverfahren, das laut [24] aus informationstheoretischer Sicht sicher ist und nicht gebrochen werden kann. Vorausgesetzt ist ein echt zufälliger Schlüssel, der mindestens so lang ist, wie die Nachricht selbst. Da der Austausch von solch langen Schlüsseln aber sehr aufwendig ist, werden in der Praxis häufig Blockchiffre verwendet, die nur eine begrenzte Schlüssellänge brauchen. Dies sind unter anderem DES, Camellia, RC2, 3DES, FEAL, RC6, AES, Blowfish, Serpent, IDEA, Twofish, Skipjack, CAST, MARS und TEA. Sie benötigen nur eine Schlüsselgröße von 128, 192 oder 256 Bit das macht sie aus informationstheoretischer Sicht angreifbar, trotzdem werden einige von Ihnen als in der Praxis sicher angesehen. Grundsätzlich könnten viele dieser Algorithmen zur Realisierung eines sicheren sozialen Netzwerkes benutzt werden. Da das Kryptosystem Advanced Encryption Standard (AES) frei verfügbar ist, ohne Lizenzgebühren eingesetzt werden darf und eine sehr weite Verbreitung hat, wird es in der Referenzimplementierung verwendet und hier ausführlicher in Abschnitt 3.2 beschrieben.

Dennoch sollten die Teilnehmer des sozialen Netzwerkes auf einen Schlüsselaustausch auf einem geheimen Kanal verzichten können. Daher ist auch die Verwendung eines asymme-

trischen Verschlüsselungsverfahrens sinnvoll, bei dem jeder Benutzer nur seinen eigenen privaten Schlüssel geheim halten muss, aber der öffentliche Schlüssel öffentlich sein kann. Auch wenn solche Verfahren im Gegensatz zu den symmetrischen Blockchiffren wie AES sehr aufwendig sind und die Schlüsselerstellung lange dauert, lohnt sich ihr Einsatz. In der Referenzimplementierung wird das asymmetrische Verfahren RSA verwendet. Dieses ist weit verbreitet und wird unter anderem in der Emailverschlüsselung nach OpenPGP-Standard verwendet. Außerdem ist es frei verfügbar und es existiert eine Implementierung in JavaScript, die unter LGPL veröffentlicht ist(Abschnitt 3.3).

Authentifikationssysteme sollen das Schutzziel Integrität erreichen. Zur Authentifikation kann ebenfalls RSA verwendet werden. Außerdem wird zur Signatur und Authentifizierung eine kryptologische Hashfunktion verwendet. Eine Hashfunktion bildet einen String mit beliebiger Länge auf einen String mit fester Länge ab. Es wird eine kryptologische Hashfunktion verwendet, die sowohl eine Einwegfunktion ist, als auch kollisionsresistent. Kollisionsresistent bedeutet, dass es hinreichend unwahrscheinlich ist, das zwei verschiedene Strings auf denselben Hashwert abgebildet werden. Diese Funktion kann verwendet werden, um die Kenntnis eines Passwortes zu beweisen, ohne dass das Passwort verraten werden muss. Es gibt eine große Zahl von Hashfunktionen, die grundsätzlich für diese Aufgabe ausgewählt werden können. Als Beispiele seien hier MD4, MD5, SHA, RIPMED, HAVAL. TIGER, PANAMA, Whirlpool und SMASH genannt. Da SHA2 zu den am weitverbreitetsten Algorithmen gehört wurde dieser für die Referenzimplementierung gewählt und wird hier im Abschnitt 3.4 genauer erläutert.

3.2 Advanced Encryption Standard

Der Advanced Encryption Standard ist ein symmetrischer Blockchiffre. Er wurde im Oktober 2000 vom National Institute of Standard and Technology NIST als Standard vorgestellt [20]. Erfunden wurde der Algorithmus von Joan Daemen und Vincent Rijmen [28], die den Algorithmus zunächst Rijndael tauften. Die folgende Beschreibung des Algorithmus lehnt sich an, an [49].

Das Verfahren ist eine Blockchiffre die Blockgrößen von 128, 192 oder 256 Bit zulässt. Wobei AES nur eine Blockgröße von 128 Bit vorsieht, dies ist der einzige Unterschied zwischen Rijndael und AES. In Referenzimplementierung wird eine Blockgröße von 256 Bit verwendet und daher genaugenommen nicht AES, sondern Rijndael implementiert. Da es sich bei der Blockgröße aber nur um ein Detail handelt, werden in der Arbeit Rijndael und AES synonym verwendet. Die Schlüssellänge kann 128, 192 oder 256 Bit betragen. AES ist frei verfügbar und kann ohne Lizenzgebühren verwendet werden. In den USA ist

es zur Verschlüsselung von Dokumenten mit der höchsten Geheimhaltungsstufe zugelassen. Laut [46] wird AES verwendet von folgenden Anwendungen:

„AES wird u. a. vom Verschlüsselungsstandard IEEE 802.11i für Wireless LAN und seinem Wi-Fi-Äquivalent WPA2, bei IEEE802.16m (WiMAX), sowie bei SSH und bei IPsec genutzt. Auch in der IP-Telefonie kommt AES sowohl in offenen Protokollen wie SRTP oder proprietären Systemen wie Skype zum Einsatz. Mac OS X benutzt AES als Standardverschlüsselungsmethode für Disk-Images, außerdem verwendet der Dienst FileVault AES. Ebenso verwendete die transparente Verschlüsselung EFS in Windows XP ab SP 1 diese Methode. Außerdem wird der Algorithmus zur Verschlüsselung diverser komprimierter Dateiarchive verwendet, z. B. bei 7-Zip und RAR. In PGP und GnuPG findet AES ebenfalls einen großen Anwendungsbereich.“

3.2.1 Die Grundlagen für AES

Wie bereits erwähnt, handelt es sich bei AES um einen symmetrischen Block- Verschlüsselungsalgorithmus. Ein solcher zeichnet sich dadurch aus, dass zunächst ein geheimer Schlüssel gewählt wird. Der zu verschlüsselnde Text beziehungsweise die zu verschlüsselnden Daten werden in gleicht große Blöcke aufgeteilt. Jeder Block wird mithilfe des Schlüssels transformiert. Die so transformierten Blöcke bilden den Geheimtext.

Mit dem inversen Algorithmus und dem Schlüssel kann der Geheimtext wieder in den Klartext umgewandelt werden. Die Länge des Geheimtextes unterscheidet sich nur unwesentlich von der Länge des Klartextes, da die Transformation nicht die Größe der Blöcke verändert. Lediglich der letzte Block des Klartextes kann aufgefüllt werden, um die nötige Blockgröße zu erreichen.

In Rijndael werden die Operationen auf Bytes ausgeführt. Jedes Byte $a = a_7...a_0$ wird als ein Polynom interpretiert:

$a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x^1 +_a 0$

Für die Bytes werden die Operationen Addition und Multiplikation wie folgt definiert werden:

- Addition: Umgangssprachlich gesprochen funktioniert die Addition wie eine normale Addition nur ohne den Übertrag. Diese kann mit einer einfachen XOR-Verknüpfung der jeweiligen Koeffizienten realisiert werden.

Tabelle 3.1: Anzahl der Runden von AES nach Schlüsselgröße(k) und Blockgröße(b)

	b = 128	b = 192	b = 256
k = 128	10	12	14
k = 192	12	12	14
k = 256	14	14	14

- Multiplikation: Die zwei Polynome werden multipliziert und dann modulo eines festen Wertes $m(x) = x^8 + x^4 + x^3 + x + 1$ gerechnet: $(a(x) * b(x))modm(x))$. Eine solche Multiplikation kann durch die Schritweise Multiplikation $x * b(x)$ leicht errechnet werden.

Die beiden Operationen bilden zusammen mit der Menge der 256 Wörter von einem Byte Länge den Galoiskörper $GF(2^8)$

3.2.2 Der Aufbau von AES

AES wird in Runden durchgeführt. Das heißt, dass Teile des Algorithmus mehrmals ausgeführt werden. Die Rundenanzahl ist abhängig von der Blockgröße und von der Schlüsselgröße. Die Anzahl der Runden kann in der folgenden Tabele 3.1 abgelesen werden.

Die einzelnen Blocks werden als Bytes (8 Bit) und Worte (32 Bit) interpretiert. In der folgenden Matrix (Abbildung 3.1) wird ein Byte als Zelle $a_{i,j}$ und ein Wort als Spalte $(a_{0,j}, a_{1,j}, a_{2,j}, a_{3,j})$, dargestellt. Der genaue Ablauf von AES wird in Algorithmus 3.1, Algorithmus 3.2 und Algorithmus 3.3 als Pseudocode dargestellt. In der Referenzimplementierung wird eine Blockgröße von 256 Bit und eine Schlüsselgröße von 256 Bit verwendet. Der Algorithmus durchläuft also 14 Runden.

```
1: Rijndael(byte State, byte CipherKey)
2: {
3: KeyExpansion (CipherKey, ExpandedKey);
4: AddRoundKey(State, ExpandedKey);
5: for i = 1 to i < Nr do
6:    Round (State, ExpandedKey +Nb * i);
7:    FinalRound(State, ExpandedKey+Nb * Nr);
8: end for
9: }
```

Algorithmus 3.1: Rijndael

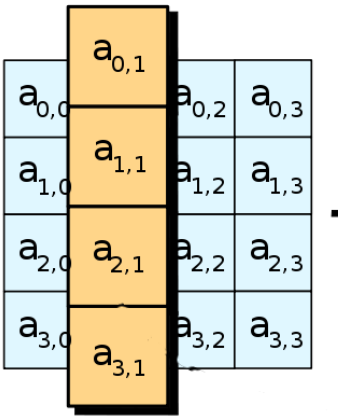


Abbildung 3.1: Ein Zustand von AES. Nach: MixColumns operation for AES, Aus: Wikimedia Commons, Urheber: User:Matt Crypto

```
1: Round (word State, word Roundkey)
2: {
3: ByteSub(State);
4: ShiftRow(State);
5: MixColumn(State);
6: AddRoundKey(State, Roundkey);
7: }
```

Algorithmus 3.2: Rijndael:Round

```
1: FinalRound (word State, word Roundkey)
2: {
3: ByteSub(State);
4: ShiftRow(State);
5: AddRoundKey(State, Roundkey);
6: }
```

Algorithmus 3.3: Rijndael:FinalRound

ByteSub()

ByteSub transformiert die einzelnen Bytes des gegebenen Blocks mit der Funktion Substitutionsbox. Die Substitutionsbox ist eine monoalphabetische Substitutionsmethode und gibt an welches Byte $a_{i,j}$ durch welches Byte $b_{i,j}$ zu ersetzen ist. Dieses Vorgehen wird Konfusion genannt. Sie kann als Tabelle gespeichert oder für jedes zu transformierende Byte neu berechnet werden. Die S-Box besteht aus den folgenden beiden Transformationen die Nacheinander auf das Byte angewendet werden:

1. Für das gegeben Byte $a = a_7...a_0 \neq 0$ wird das multiplikative Inverse zu der daszugehörigen Ploynomdarstellung gesucht. $f(x) = a_7x^7+a_6x^6+a_5x^5+a_4x^4+a_3x^3+a_2x^2+a_1x^1 +_a 0 \in GF(2^8)$ und $f^{-1}(x) = b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2b_1x^1 +_b 0$ Das Ergebnis der ersten Transformation ist $b = b_7...b_0$

2. Auf das Ergebnis der ersten Transformation wird die 2. Transformation angewendet.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} * \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

Das Ergebnis ist $c = c7...c0$

Die die Substitutionsbox ist invertierbar und bildet die nichtlineare Schicht. Laut [28] sorgt die Konstruktion der S-Box für nahezu „idealen Schutz vor differenzieller und linearer Kryptoanalyse."

Shiftrows()

In AES wird Shiftrows() zur Permutation eingesetzt. Der Zustand eines Blockes wird transformiert, in dem die Bytes aus den Zeilen je um die Anzahl der Zeilen nach links verschoben werden. $a_{i,j} = a_{i,j-i}$ Siehe Abbildung 3.2. Bei einer Blockgröße von 256 Bit (Wie in der Referenzimplementierung) wird die 3. und die 4. Spalte je um eine Position mehr verschoben. Also: $a_{i,j} = a_{i,j-i-1} Fall si > 2$

mixColumns()

In AES wird zur Diffusion das Verfahren mixColumns eingesetzt. Die Spalten eines Zustandes eines Blockes werden durch die folgende Matrixmultiplikation transformiert:

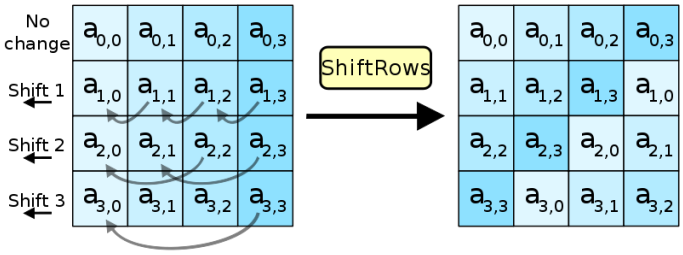


Abbildung 3.2: ShiftRows Operation von AES, Aus: Wikimedia Commons, Urheber: User:Matt Crypto

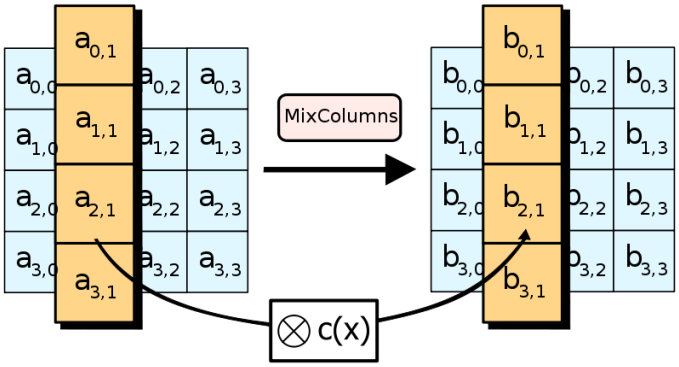


Abbildung 3.3: MixColumns Bild: MixColumns operation for AES, Aus: Wikimedia Commons, Urheber: User:Matt Crypto

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{pmatrix} * \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

Siehe Abbildung 3.3.

Die einzelnen Elemente werden als Bytes interpretiert auf die, die in den Grundlagen Spalten definierte Addition und Multiplikation angewendet wird. Zusammen bilden ShiftRows() und MixColumns() die lineare Schicht und sorgen für eine optimale Durchmischung der Bits eines Blocks.

AddRoundKey

Der Block wird Byteweise mit dem aktuellen Rundenschlüssel verschlüsselt. Dafür werden der Rundenschlüssel und der Block, die die gleiche Länge haben, Byteweise XOR verknüpft(Abbildung 3.4). AddRoundKey wird in jeder Runde angewendet und ist die einzige Operation, die vom Benutzerschlüssel abhängig ist. Aus dem Benutzerschlüssel werden durch die Schlüsselexpansion die Rundenschlüssel berechnet. Durch die Verknüpfung mit dem Rundenschlüssel von der ersten Runde und als letzter Schritt innerhalb einer Runde wirkt sich dieser auf jedes Bit der Rundenergebnisse aus. Es gibt im Verlauf der Ver-

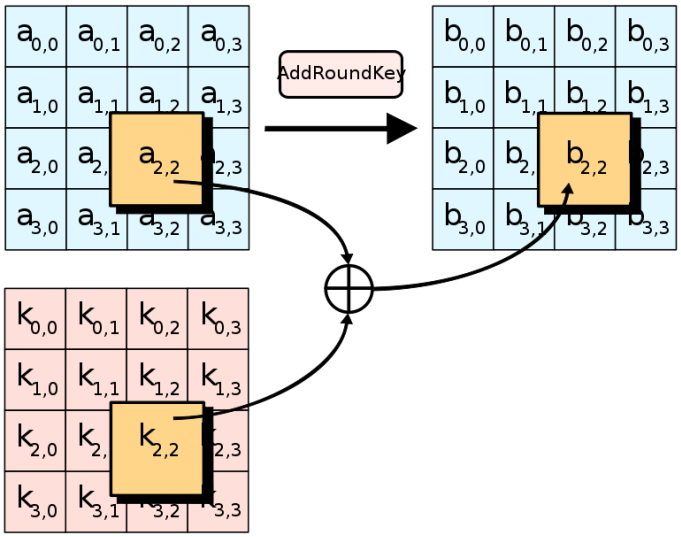


Abbildung 3.4: AddRoundKey Bild: AddRoundKey operation for AES, Aus: Wikimedia Commons, Urheber: User:Matt Crypto

schlüsselung eines Blocks keinen Schritt, dessen Ergebnis nicht in jedem Bit vom Schlüssel abhängig wäre.

Schlüsselexpansion

Der AES-Schlüssel, der vom Benutzer eingegeben wird, besteht je nach gewählter Schlüssellänge aus 4, 6 oder 8 Worten zu je 4 Bytes oder 32 Bit. Die Anzahl der Worte eines Schlüssels wird mit Nk bezeichnet. Da AddRoundKey in jeder Runde und der Vorrunde angewendet wird, muss der Schlüssel eine Länge von $Nr+1*nb$ Worten haben. Wobei Nr die Rundenanzahl und nb die Anzahl der Spalten in einem Block bezeichnet. Die Worte $w[i]$ des so entstanden Schlüssel werden der Reihe nach für AddRoundKey verwendet. Also für den Fall $Nb = 4$ wird beim ersten Aufruf von AddRoundKey der Rundenschlüssel $w[0]w[1]w[2]w[3]$ verwendet. Die Worte des erweiterten Schlüssels $w[0]...w[Nr]$ werden wie folgt berechnet. Die ersten Worte des erweiterten Schlüssels $w[0]-w[Nk-1]$ sind die Worte des Benutzerschlüssels. Die weiteren Wörter können durch die Formel angegeben werden:

Falls i durch Nk teilbar ist:
$w[i] = (SubWord(RotWord(w[i-1]))\, XOR\, ((Rcon[i/Nk])\, XOR\, w[i-Nk])$
sonst:
$w[i] = w[i-1]\, XOR\, w[i-Nk]$

RotWord()

Die Funktion RotWord() verschiebt die 4 Bytes eines Wortes um eine Position nach links.
$RotWord([a0, a1, a2, a3]) = [a1, a2, a3, a0]$ **SubWord()**
Die Funktion SubWord wendet auf jedes Byte eines gegeben Wortes die S-Box an. Also:

$SubWord([a0, a1, a2, a3]) = [S-Box(a0), S-Box(a1), S-Box(a2), S-Box(a3)]$ **Rcon[i]**
Die Rundenkonstane der I-ten Runde Rcon[i]() ist ähnlich wie die S-Box eine Abbildung. Für die genaue Berechnung siehe [28]. **Die Entschlüsselung**
Die Schritte der Verschlüsselung lassen sich alle leicht invertieren. Um AddRoundKey zu invertieren, reicht es den Rundenschlüssel blockweise rückwärts zu verwenden. Da XOR zu sich selbst invers ist. MixColum kann invertiert werden zu der Funktion invMixColumn in dem das folgende Polynom verwendet wird:

$$c^{-1}(x) = 0B_{hex}x^3 + 0D_{hex}x^2 + 09_{hex}x + 0E_{hex}$$

Da die Koeffizienten hier höher sind als in der normalen Funktion ist das Entschlüsseln von Nachrichten deutlich aufwendiger als das Verschlüsseln. Zum Invertieren von ShiftRow reicht es, in die andere Richtung zu schieben. So erhält man die Funktion InvShiftRow. InvByteSub kann aus der Tabelle abgelesen werden, die für ByteSub erstellt werden kann. Der genaue Ablauf der Entschlüsselung wird dargestellt durch Algorithmus **??**, Algorithmus 3.5 und Algorithmus 3.6.

```
1: invRijndael (byte State, byte CipherKey)
2: {
3: AddRoundKey(State, Roundkey);
4: for i = 1 to i < Nr − 1 do
5:    invRound(State, Roundkey);
6: end for
7: invFinalRound(State, Roundkey);
8: }
```

Algorithmus 3.4: invRijndael

```
1: invRound (word State, word Roundkey)
2: {
3: invByteSub(State);
4: invShiftRow(State);
5: invMixColumn(State);
6: AddRoundKey(State, InvMixColumn(Roundkey));
7: }
```

Algorithmus 3.5: Rijndael:invRound

```
1: invFinalRound (word State, word Roundkey)
2: {
3: invByteSub(State);
4: invShiftRow(State);
5: AddRoundKey(State, Roundkey);
6: }
```

Algorithmus 3.6: Rijndael:invFinalRound

3.2.3 Die Sicherheit von AES

Die Sicherheit von AES, ist anders als bei der Methode One-Time-Pad nicht beweisbar (Abschnitt 3.1).

„Das Problem jeder praktikablen Chiffriermethode ist jedoch, dass ihre Sicherheit (bis jetzt) nicht beweisbar ist. Man kann nur testen, ob die bisher bekannten Angriffsmethoden der Kryptanalyse versagen. Es gab zwar Warnungen wegen der besonders einfachen Struktur von AES, doch rational begründete Bedenken konnte niemand vorbringen: AES schien resistent gegenüber allen bekannten Angriffen.“ [50]

Obwohl bisher keine Angriffe mit praktischem Nutzen bekannt sind, werden in diesem Abschnitt einige Theorietische Angriffsmöglichkeiten vorgestellt.

Exhaustive Key Search
Der einfachste Angriff ist das Ausprobieren von allen möglichen Schlüsseln – die sogenannte Bruteforce Methode. Bei der Annahme, der Schlüssel sei echt zufällig gewählt und einer Schlüsselgröße von 128 Bit ergibt sich ein Schlüsselraum von $3,4 * 10^{38}, bei 192 Bit 6,2 * 10^{57} und bei 256 Bit 1,1 * 10^{77}$ möglichen Schlüsseln. Selbst wenn es eine Maschine gäbe, die den gesamten Schlüsselraum des DES Algorithmus (56 Bit) in nur einer Sekunde durchsuchen könnte, dann würde es bei Rijndael ungefähr 149.000 Billionen Jahre dauern, ehe der gesamte Schlüsselraum durchsucht ist.

Angriffe, die dieses Verfahren verkürzen können, werden als Shortcut Attacken bezeichnet. Diese Verfahren gelten als Kryptoanalyse und werden in die Kategorien known-plaintext (Der Angreifer ist in Besitz von Klar und Geheimtexten) choosen-plaintext (Es gibt den Geheimtext zu bestimmten Klartexten) und related Key unterteilt.

Der Begriff Security Margin gibt die Differenz an, zwischen der Anzahl der Runden eines Algorithmus und die Anzahl der Runden, nach denen der Algorithmus noch geknackt werden kann. Gibt es zum Beispiel einen Algorithmus der 10 Runden vorsieht und einen

Angriff, der erfolgreich gegen den Algorithmus nach 8 Runden eingesetzt werden kann, so beträgt die Security Margin für den Algorithmus 2.

Angriff durch differenzielle Kryptoanalyse
Die differenzielle Kryptoanalyse wurde 1990 von Eli Biham und Adi Shamir [18] vorgestellt. Sie befasst sich mit paaren von Geheimtexten, deren Klartexte Differenzen aufweisen. Diese Methode untersucht wie sich die Differenzen durch jede Runde des Algorithmus verändern. Nach vielen Versuchen wird sich ein Schlüssel mit hoher Wahrscheinlichkeit heruasstellen.
Angriffe durch lineare Kryptoanalyse
Die Lineare Kryptoanalyse wurde erstmals 1992 von Mitsuru Matsui und Atshuiro Yamagishi [**?**] eingesetzt. Es wird lineare Approximation verwendet, um das Verhalten eines Block-Verschlüsselungsalgorithmus zu erraten. Also die statistisch lineare Relation zwischen Klartext, Geheimtext und Schlüssel.Varianten dieser Angriffe sind die folgenden:

- Biclique-Angriff: [1]
- XSL-Angriff: [4]
- Impossible Differentials
- Square Attacks
- Collision Attacks

3.3 RSA

RSA wurde 1977 von Rives, Shamir und Adleman veröffentlicht [37] und beruht auf der Vorarbeit von Diffie und Hellman [**?**]. Es ist ein asymmetrisches Kryptografisches Verfahren, was sowohl zum Verschlüsseln als auch zum Signieren von Nachrichten eingesetzt werden kann. Die Beschreibung in diesem Abschnitt beruht hauptsächlich auf den oben genanten Artikeln so wie [48]. RSA verwendet ein Schlüsselpaar bestehend aus einem öffentlichen Schlüssel und einem privaten Schlüssel. Der private Schlüssel unterliegt der Geheimhaltung, während der öffentliche Schlüssel public sein kann und somit auch über unsichere Kanäle ausgetauscht werden darf. Das Verfahren beruht auf Einwegfunktionen, so das nur mit sehr großem Aufwand aus dem öffentlichen Schlüssel, der private berechnet werden kann.

3.3.1 Die Grundlagen für RSA

In diesem Abschnitt werden die Grundlagen erläutert, die für den Algorithmus von Bedeutung sind.

Einwegfunktionen
Eine Funktion $f(x)$ die leicht zu berechnen ist, aber deren Inverse $f^{-1}(x)$ schwer zu berechnen ist, wird Einwegfunktion genannt. Leicht meint in diesem Fall, dass die Funktion effizient also in Polynomialzeit berechenbar und somit in der Komplexitätsklasse P ist. Schwer bedeutet in diesem Kontext, dass es keinen probabilistischen Algorithmus gibt, der die Funktion in Polynomialzeit löst, also das das Problem echt schwerer ist als die Komplexitätsklasse BPP.

Bisher ist keine Funktion bekannt, für die sich die Einweg-Bedingung beweisen lässt. Auch die generelle Existenz von Einwegfunktionen ist nicht bewiesen. Zwar würde ein Beweis für die Existenz von Einwegfunktionen auch den Satz P $\neq$ NP beweisen, aber der Beweis P $\neq$ NP, erlaubt noch keine Rückschlüsse auf die Existenz von Einwegfunktionen.

Trotzdem gilt nach dem bisherigen Stand der Wissenschaft die Faktorisierung einer natürlichen Zahl als ein Beispiel für eine Einwegfunktion. Die Berechnung einer Zahl durch die Multiplikation von Primzahlen ist recht einfach, aber die Zerlegung der Zahl zurück in ihre Primzahlen ist sehr aufwendig. RSA nutzt eine sogenannte Falltürfunktion, eine spezielle Art der Einwegfunktion, die mithilfe einer Zusatzinformation leicht zurückgerechnet werden kann.

3.3.2 Der Aufbau von RSA

In diesem Abschnitt werden der Aufbau und der Ablauf des Algorithmus beschrieben.

Erzeugung des Schlüsselpaars Der öffentliche Schlüssel ist ein Zahlenpaar (e, N) und der private Schlüssel ist ein Zahlenpaar (d, N) Der Algorithmus3.7 gibt an, wie die Zahlenpaare berechnet werden. In der Referenzimplementierung werden Zahlen mit einer Größe von 1024Bit gewählt. Zum Erzeugen der Primzahlen in Schritt 1, werden in der Referenzimplementierung Zufallszahlen gebildet, die dann mit dem Miller-Rabin-Test auf Prim getestet werden. Wenn es sich um keine Primzahlen handelt, werden die Zahlen hochgezählt bis sich entsprechende Zahlen gefunden haben. Nach der Erzeugung der Schlüssel (e, N) und (d, N) werden die übrigen Werte p, q und FO nicht mehr gebraucht und sollten gelöscht werden.

1: Wähle zwei zufällige Primzahlen mit ausreichender Größe, die stochastisch unabhängig und ungleich sind. $p/neqq$L
2: Berechne: $N = p * q$
3: Berechne die Eulersche Funktion von N. $\varphi(N) = (p-1) \cdot (q-1)$
4: Wähle eine zu $\varphi(N)$ teilerfremde Zahl e, für die gilt $1 < e < \varphi(N)$
5: Berechne den Entschlüsselungsexponenten d als Multiplikativ Inverses von e bezüglich des Moduls $\varphi(N)$. Es soll also die folgende Kongruenz gelten: $e \cdot d \equiv 1 \mod \varphi(N)$

Algorithmus 3.7: RSA Schlüsselerstellung aus [48]

Verschlüsseln von Nachrichten
Um eine Nachricht K zu verschlüsseln, braucht der Absender den öffentlichen Schlüssel des Empfängers (e, N). K ist dabei eine natürliche Zahl, die kleiner als N sein muss. Textblöcke müssen zum Verschlüsseln also zunächst in Zahlen transformiert werden. Die verschlüsselte Nachricht C wird nach der folgenden Formel berechnet. $C \equiv K^e \mod N$

Entschlüsseln von Nachrichten
Eine verschlüsselte Nachricht C kann mit dem privaten Schlüssel (d, N) nach der folgenden Formel entschlüsselt werden. $K \equiv C^d \mod N$

Signieren von Nachrichten
Eine Nachricht kann vom Absender signiert werden, in dem er sie mit dem eigenen privaten Schlüssel (d, N) verschlüsselt und an die normale unverschlüsselte Nachricht hängt. Der Empfänger kann die Nachricht mit dem öffentlichen Schlüssel (e, N) des Absenders entschlüsseln und sie mit der unverschlüsselten Nachricht vergleichen. Bei Übereinstimmung, kann sich der Empfänger sicher sein, dass der Absender der Nachricht im Besitz des privaten Schlüssels ist und die Nachricht unterwegs nicht verändert wurde. In der Praxis wird für die Signatur nicht die gesamte Nachricht verschlüsselt, sondern nur ein Hashwert der Nachricht. Siehe dazu auch Abschnitt **??**.

Anwendung in der Praxis
Das oben beschriebene Verfahren wird so nicht in der Praxis und auch nicht in der Referenzimplementierung verwendet, da es noch eklatante Schwächen hat. RSA ist ein deterministischer Verschlüsselungsalgorithmus. Ein Angreifer kann also einen Text raten und dann mit dem chiffrierten Text vergleichen. Gerade, wenn die Möglichkeiten des Gesendeten gering sind, ist dieser Angriff sehr leicht durchführbar. Wenn beispielsweise unter einem Eintrag in einem sozialen Netzwerk eine Reihe von Bewertungen steht, von denen bekannt ist, dass sie ein „like“ oder ein „disslike“ codieren, so kann der Angreifer einfach beide Texte selbst verschlüsseln und erhält so die Erkenntnis über den Klartext.

In der Praxis werden solche Angriffe verhindert, in dem sogenannte Padding-Verfahren eingesetzt werden. Hierbei wird ausgenutzt, dass der Klartext kürzer ist als die größtmögliche Blockgröße. An den Klartext wird noch eine Zeichenfolge R gehangen, die nach einem bestimmten Muster aufgebaut ist und Angaben über die Länge des Klartextes enthält. Die Paddingverfahren Optimal Asymmetric Encryption Padding (OAEP) und Probabilistic Signature Scheme (PSS) nutzen zur weiteren Randomisierung des Klartextes selbst kryptographische Hashfunktionen. Diese sind unter idealisierten Annahmen und der RSA-Annahme beweisbar sicher. [38]

Da die Ver- und Entschlüsselung bei RSA sehr aufwendig ist, werden in der Praxis asymmetrische Konzelationssysteme mit symmetrischen Konzelaionssystemen wie AES zu hybriden Verfahren kombiniert. Dabei wird nur ein zufällig erzeugter Schlüssel mit RSA verschlüsselt und der Rest der Nachricht dann mit einem effizienteren Verfahren wie AES. Im Falle eines zufälligen AES Schlüssels, kann auch auf ein Paddingverfahren verzichtet werden, da der Suchraum der möglichen Schlüssel für einen praktischen Angriff zu groß ist.

3.3.3 Die Sicherheit von RSA

In diesem Abschnitt wird die Sicherheit von RSA diskutiert.

Faktorisierung Wie oben schon beschrieben, beruht die Sicherheit von RSA auf der Annahme, dass die Faktorisierung eine Einwegfunktion ist und somit für ausreichend große Zahlen nicht in praktikabler Zeit lösbar. Aus dem öffentlichen Schlüssel (e, N) lässt sich der private Schlüssel durch die Primfaktorzerlegung von $N = p * q$ berechnen.

Mit den heute bekannten Verfahren und der heute verfügbaren Hardware gilt eine Primfaktorzerlegung für Zahlen mit der Größe von 1024 oder 2048 Bit als unmöglich. Es ist aber nicht bewiesen, dass es keinen Algorithmus gibt, der dies in praktikabler Zeit lösen könnte. Im Dezember 2009 wurde eine RSA-Zahl mit 768 Bit länge faktorisiert [39]. Die Faktorisierung fand auf mehreren 100.000 Rechnern statt und das Forscherteam geht davon aus, dass es bei gleichbleibender Entwicklung in 10 Jahren möglich sein kann, einen 1024 Bit langen Schlüssel zu knacken. Es kann aber nicht ausgeschlossen werden, dass schon vorher ein Algorithmus oder ein Quantencomputer erfunden wird, der die Schlüssel viel schneller knacken kann. Es kann nicht einmal ausgeschlossen werden, dass es bereits heute Menschen gibt, die im Besitz solcher Technologien sind.

Weitere Angriffsmöglichkeiten Der modifizierte oder hybride RSA-Algorithmus gelten allgemein als sicher. Mögliche Angriffe in der Theorie bilden aber der Chosen-Ciphertext-Angriff , der Wiener Angriff und der Hastad Angriff siehe [48].

3.4 SHA

Wie oben bereits erwähnt wird der Secure Hash Algorithm (SHA) zur Signatur und Authentifizierung verwendet. SHA umfasst eine Gruppe kryptologischer Hashfunktionen, die sowohl Einwegfunktion, als auch kollisionsresistent sind. SHA wurde ursprünglich 1994 vom National Institute of Standards and Technology (NIST) und der NSA (National Security Agency) vorgestellt. Aufgrund von Schwächen der Verfahrens [41] wurde im August 2002 eine verbesserte Variante vorgestellt [19], Nun werden Daten auf größere Haswerte von 224, 256, 384 oder 512 Bit abgebildet. In der Referenzimplementierung wird SHA512 verwendet.

In [31] wird der genaue Ablauf von SHA512 beschrieben. Zunächst wird der String oder die Nachricht, von der ein Hashwert gebildet werden soll, mit einer Eins und hinreichend Nullen aufgefüllt, so, dass für die Länge l gilt: $l = (n/1024) - 64$ In die letzten 64 Bit wir die Länge des Strings codiert. Nun wird der SHA-2-Zustand initialisiert, welcher aus den acht 64-Bit-Worten A bis H besteht. Jeder 1024-Bit große Nachtrichtenblock wird nun in 80 Runden mit dem in Abbildung 3.5 dargestellten Algorithmus verarbeitet. Wobei gilt:

- Die Funktion Ch: $Ch(B, C, D) = (B \wedge C) \oplus (\neq B \wedge D)$
- Die Funktion $\Sigma 1$: $\Sigma 1(E) = ROTOR^{14}(E) \oplus ROTOR^{18}(E) \oplus ROTOR^{41}(E)$
- Die Funktion $ROTOR$ ist ein zyklische Rechtsverschiebung auf ein Wort
- Die Funktion MA: $Maj(B, C, D) = (B \wedge C) \oplus (B \wedge D) \oplus (C \wedge D)$
- Die Funktion $\Sigma 0$: $\Sigma 0(E) = ROTOR^{28}(E) \oplus ROTOR^{34}(E) \oplus ROTOR^{39}(E)$

Die Konstanten können in [31] nachgelesen werden. Nach allen 80 Runden wird der nächste Nachrichtenblock verarbeitet. Sind alle Blöcke verarbeitet, entspricht der resultierende Zustand dem Hashwert. Bisher sind keine Angriffe bekannt, die die Sicherheit von SHA512 in der Praxis gefährden[33].

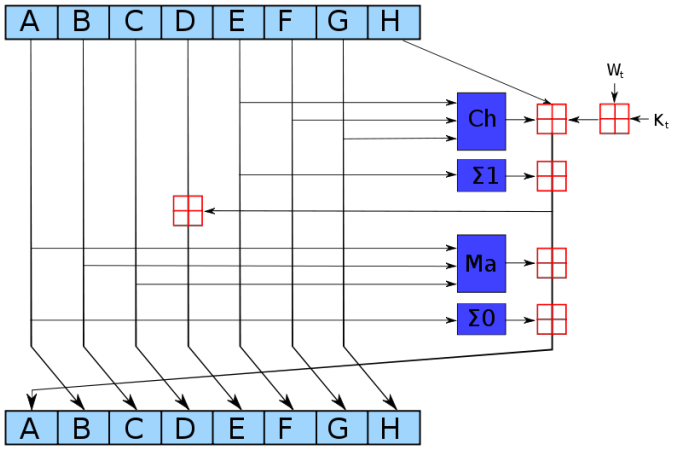


Abbildung 3.5: Aufbau einer Runde von SHA-2S. Nach: A schematic that shows the SHA-2 algorithm, Aus: Wikimedia Commons, Urheber: User:kockmeyer

Kapitel 4

Ein sicheres soziales Netzwerk

In diesem Kapitel wird beschrieben, wie ein soziales Netzwerk aussehen kann, welches die Privatsphäre seiner Anwender schützt und wie dies in der Referenzimplementierung pribook.com umgesetzt wird. Zunächst wird der grundsätzliche Aufbau geschildert, woraufhin die einzelnen Bereiche - Nachrichten, Freundschaften, Gruppen und Untergruppen – folgen. Des Weiteren werden Prämissen aufgestellt, unter welchen das System sicher ist und schließlich mögliche Angriffsszenarien diskutiert.

4.1 Invarianten

Für ein Social Network Service, der die Privatsphäre der Nutzer bewahrt werden im Rahmen dieser Diplomarbeit folgende Invarianten definiert:

Invariante 1: Verschlüsselung von Nutzer zu Nutzer

Der Schutz der Daten kann nur dann als gewährleistet angesehen werden, wenn die Informationen auf dem kompletten Weg vom Absender bis zum Empfänger verschlüsselt werden. Lösungen, die nur eine Verschlüsselung vom Nutzer bis zum Server vorsehen, sind nicht ausreichend, da es gerade die Betreiber der Server sind, deren Umgang mit den Nutzerdaten als kritisch angesehen wird [siehe Abschnitt 1.3]. Auch Peer-to-Peer Modelle, die auf einen zentralen Server verzichten, sind nicht ausreichend, da Informationen auch dort unterwegs verloren gehen können. Daher ist es sinnvoll, als Invariante für ein sicheres soziales Netzwerk zu fordern, dass alle Informationen von Nutzer zu Nutzer verschlüsselt werden.

Invariante 2: asymmetrische Verschlüsselung

Die Verwendung von symmetrischer Verschlüsselung würde den Austausch von Schlüsseln nötig machen. Dieser Austausch benötigt selbst einen sicheren Kanal und ist gerade wegen der Vielzahl von nötigen Austauschen extrem aufwendig. Darum sollte ein sicheres soziales Netzwerk ohne einen Schlüsselaustausch auskommen und asymmetrische Verschlüsselung verwenden.

Invariante 3: Bedienung im Webbrowser möglich.

Die Nutzbarkeit gegenüber den herkömmlichen Netzwerken darf nicht deutlich schlechter werden. Um die Hürden für die Teilnahme an einem Netzwerk möglichst gering zu halten und so möglichst vielen Menschen die Teilnahme zu ermöglichen, ist es notwendig, dass die Bedienung des Netzwerkes wie in den großen Netzwerken ausschließlich über den Webbrowser möglich ist. Eine Webseite garantiert auch die größtmögliche Plattformunabhängigkeit, was für Massenanwendungen wie soziale Netzwerke eine wichtige Voraussetzung ist. Zwar werden immer wieder Sicherheitslücken in Webbrowsern bekannt, aber da viele Nutzer durch eine Installation abgeschreckt werden, ist eine installationslose Bedienung im Webbrowser eine Voraussetzung für eine Massenanwendung für normale Endverbraucher. Siehe dazu auch Abschnitt 4.9.

4.2 Architektur

Die folgende Architektur kann die geforderten Invarianten realisieren.

Damit Daten schon sicher auf dem Client ver- und entschlüsselt werden können, muss das Netzwerk clientseitig mit einer Skriptsprache funktionieren, die der Webbrowser interpretiert. Aufgrund der großen Kompatibilität zu nahezu allen Webbrowsern und der verfügbaren Bibliotheken wurde für die prototypische Umsetzung JavaScript verwendet.

Die Anwender laden sich das Skript, zur Ver- und Entschlüsselung herunter, in dem sie die Webseite im Browser aufrufen. Um sicherzugehen, dass das korrekte Skript heruntergeladen wurde, können sie den Quellcode im Webbrowser überprüfen. Für praktische Zwecke reicht es aus, wenn dies von einigen Nutzern stichprobenartig ausgeführt wird. Dazu mehr im Abschnitt zu möglichen Angriffsszenarien.

Da davon ausgegangen wird, dass alle nicht öffentlichen Informationen schon auf dem Client verschlüsselt werden, entfallen auch die Hauptargumente für eine verteilte Serverarchitektur. Also ist eine Architektur mit einem zentralen Server vollkommen ausreichend. Pribook verwendet serverseitig eine MySQL-Datenbank und PHP für den Zugriff auf diese

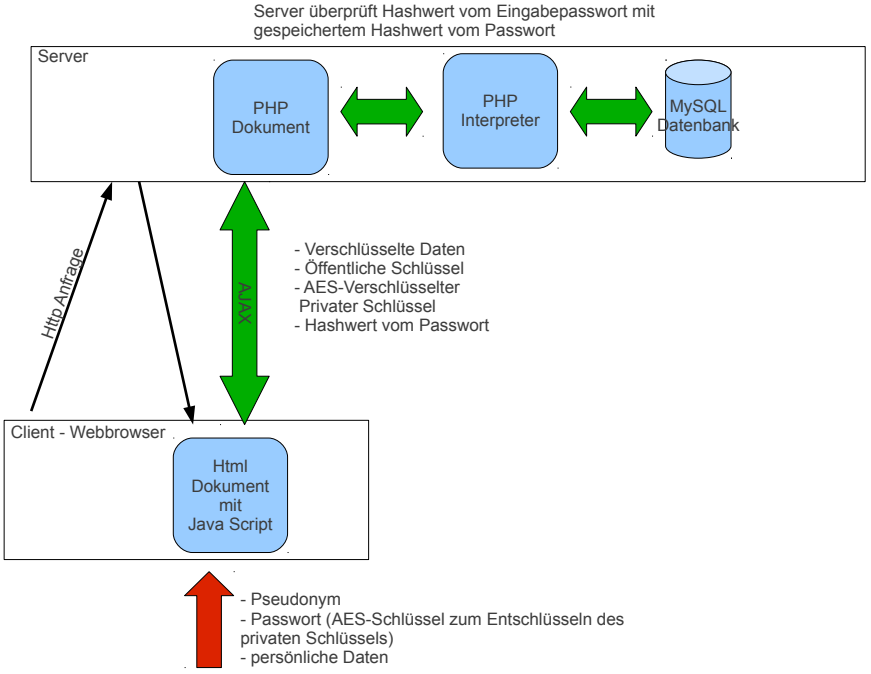


Abbildung 4.1: Client-Server Architektur

Datenbank, da diese Technologien weit verbreitet und auf vielen kommerziellen Servern standardmäßig installiert sind.

Insgesamt umfasst pribook.com ein HTML-Gerüst und mehrere JavaScript-Dateien, die beim Aufrufen der Seite in den Webbrowser geladen werden. Serverseitig umfasst das Projekt eine MySQL-Datenbank und einige PHP-Dateien, die den Zugriff auf die Datenbank ermöglichen. Siehe Abbildung 4.1.

4.3 Registrierung und Anmeldung

Jeder Nutzer eines sicheren sozialen Netzwerkes braucht ein asymmetrisches Schlüsselpaar. Dieses sollte direkt bei der Registrierung im Netzwerk auf dem Client erzeugt werden. Damit sich die Nutzer nicht den privaten Schlüssel merken müssen, kann er mit dem Passwort der Nutzer verschlüsselt und dann zusammen mit dem öffentlichen Schlüssel an den Server gesendet und dort gespeichert werden. Dies hat zur Folge, dass die Sicherheit von der Stärke des gewählten Passworts abhängt. Laut [44] kann ein gut gewähltes Passwort aber als sicher angesehen werden. Siehe dazu auch regeln für ein gutes Passwort in Abschnitt

?? und mögliche Angriffszenarien in Abschnitt 4.10. Die Eingabe eines Passwortes bei der Registrierung ist auch sinnvoll, damit sich Nutzer gegenüber dem Server authentifizieren können. Die Inhalte der Nachrichten würden zwar auch ohne Authentifizierung gegenüber dem Server für unautorisierte Nutzer verborgen bleiben, die Kommunikationsumstände jedoch nicht. Dafür wird bei der Registrierung auch ein Hashwert des Passwortes gebildet und an den Server geschickt. Um die gesamte Menge der Hashwerte von Passwörtern die auf dem Server liegen besser vor einem Wörterbuchangriff zu schützen, kann das Passwort, bevor der Hashwert gebildet wird, noch mit einem zufälligen String, einem sogenannten Salt konkateniert werden. Neben dem Passwort müssen Nutzer bei der Registrierung im Netzwerk noch ein Pseudonym angeben, um eindeutig identifizierbar zu sein. Das Pseudonym ist die einzige Information, die im Klartext gespeichert werden muss.

Nach erfolgreicher Registrierung kann eine Anmeldung wie folgt ablaufen: Nachdem die Webseite geladen ist, und gegebenenfalls vom Nutzer überprüft wurde, kann er sich mit Passwort und seinem Pseudonym anmelden. Das Skript schickt daraufhin einen Hashwert des Passwortes zusammen mit dem Pseudonym zum Server. Dieser überprüft den Hashwert und bei Erfolg sendet er den verschlüsselten Schlüssel des Nutzers zurück. Auf dem Client wird dann der private Schlüssel mit dem Passwort entschlüsselt.

In der Referenzimplementierung pribook.com wird bei der Registrierung ein RSA-Schlüsselpaar erstellt und der private Schlüssel mit AES und dem Passwort als Schlüssel verschlüsselt. Für den Hashwert des Passwortes wird ein zufälliger Salt erstellt, der mit dem Passwort konkateniert wird. Von diesem zusammengesetzten String wird ein SHA512-Wert gebildet, der zusammen mit dem Salt an den Server gesendet und dort gespeichert wird. Bei Verlust des Passwortes kann der private Schlüssel nicht wieder hergestellt werden.

4.4 Nachrichten

Kein soziales Netzwerk kommt ohne Nachrichten aus, die die Nutzer untereinander verschicken können. Dabei gibt es bei den verschiedenen Netzwerken ganz unterschiedliche Arten von Nachrichten. Die einfachste Form sind Nachrichten, die genau definierte Empfänger haben und meistens private Nachrichten genannt werden. Dann gibt es Nachrichten, die öffentlich sind, oder zumindest eine sehr große Anzahl von Empfängern haben. Welchen anderen Nutzern die Nachricht dann tatsächlich angezeigt wird, hängt oft von Abonnements oder statistischen Auswahlverfahren ab. Auch Profildaten werden meistens in Form von Nachrichten gespeichert.

Um Nachrichten mit einem asymmetrischen Verfahren zu verschlüsseln, brauchen sie immer wohldefinierte Empfänger. Um die Ver- und Entschlüsselung einer Nachricht zu beschleunigen und auch mit einer Vielzahl von Empfängern umzugehen, ist es sinnvoll, die Nachricht zunächst mit einem symmetrischen Verschlüsselungsverfahren – wie zum Beispiel AES – zu verschlüsseln. Für jeden Empfänger wird dann der AES-Schlüssel mit seinem öffentlichen RSA-Schlüssel verschlüsselt und an den Text gehangen. Zum einfacheren Verständnis werden diese asymmetrisch verschlüsselten AES-Schlüssel im weiteren Text und in der Referenzimplementierung Privacy Padlock oder kurz Pri genannt. Dadurch, dass zunächst ein zufälliger AES-Schlüssel mit fester länger erstellt wird, kann auch auf die Anwendung eines Paddingverfahrens verzichtet werden. Siehe Kapitel 3.

Beim Versenden einer Nachricht wird also zunächst ein zufälliger Schlüssel erstellt, mit dem dann die Nachricht mit AES verschlüsselt wird. Dann werden die öffentlichen Schlüssel der Empfänger und des Absenders aus der Datenbank geholt und mit ihnen werden die Pris erstellt. Schließlich wird die so verschlüsselte Nachricht an die Datenbank geschickt. Absender und Empfänger der Nachricht stehen so mit ihren Pseudonymen im Klartext in der Datenbank. Nur die Nachricht und die AES-Schlüssel sind verschlüsselt.

Der Overhead, der so für jede Nachricht entsteht, ist abhängig von der Anzahl der Empfänger und der gewählten Schlüsselgröße. Für jeden Empfänger und den Absender wird ein Pri an die Nachricht gehangen, der maximal so groß wie der öffentliche RSA-Schlüssel ist. Der Overhead ist unabhängig von der Größe der Nachricht, was vor allem bei dem Versenden von Dateien von Vorteil ist.

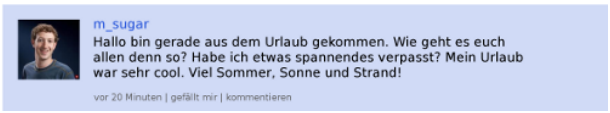


Abbildung 4.2: Statusnachricht im Klartext

Abbildung 4.3: Klartext wird mit symmetrischem Verfahren verschlüsselt

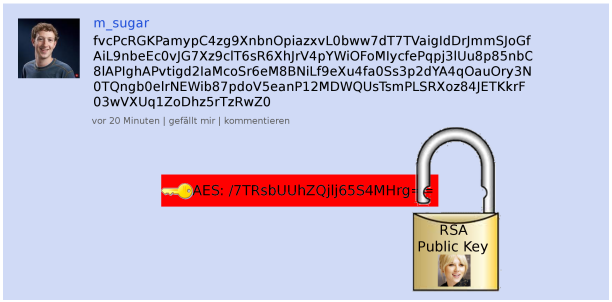


Abbildung 4.4: Der Schlüssel wird mit öffentlichem RSA-Schlssel verschlüsselt

Abbildung 4.5: Für jeden Freund wird der verschlüsselte Schlüssel an den Text gehangen

Nachdem eine verschlüsselte Nachricht, inklusive der zugehörigen Pris empfangen wurde, wird die Nachricht entschlüsselt. Dafür wird zunächst der verschlüsselte private Schlüssel des Anwenders aus der Datenbank geholt und mit dem Passwort des Anwenders entschlüsselt. Dann wird mit dem privaten Schlüssel, der passende Pri für den Benutzer entschlüsselt. Das Ergebnis ist der AES-Schlüssel, mit dem die Nachricht wieder in ihren Klartext verwandelt werden kann.

Der Nachricht können auch Dateien, zum Beispiel Videos oder Bilder angehangen werden. Die Dateien werden dann auch mit dem AES-Schlüssel verschlüsselt und an die Datenbank gesendet. Das Verfahren hat auch den Vorteil, dass die Dateien nur einmal in der Datenbank gespeichert werden müssen, auch wenn sie sehr vielen Nutzern zu Verfügung gestellt werden.

Auch Profilinformationen, die die Nutzer über sich in das Netz stellen, können genau wie die Nachrichten realisiert und verschlüsselt werden. Dafür ist nur nötig, dass für alle Profilinformationen auch Empfänger bestimmt werden. Anders als die Nachrichten werden sie jedoch den Empfängern nicht automatisch angezeigt beziehungsweise zugesendet, sondern nur dann, wenn die Empfänger auf das Profil des Nutzers klicken. Außerdem macht es bei den Profilinformationen auch Sinn, öffentliche Einträge zu erlauben, damit auch Nutzer die sich noch nicht kennen, finden oder kennenlernen können. Bei den etablierten sozialen Netzwerken gibt es noch Pinnwandeinträge, das sind Nachrichten, die bei der Ansicht eines Profils angezeigt werden, und in der Regel öffentlich sind.

Da es bei solchen Pinnwandeinträgen zu einem Konflikt kommen kann, über die Frage wer die Empfänger eines solchen Eintrags bestimmen darf, sollte in einem sicheren sozialen Netzwerk auf diese Funktion verzichtet werden. An der Stelle, wo sonst die Pinnwandeinträge stehen, können statt dessen Nachrichten zwischen den beiden Nutzern angezeigt werden.

In der Referenzimplementierung werden die Nachrichten wie hier beschrieben verschlüsselt. Bei der Registrierung wird ein 1024Bit großer RSA-Schlüssel generiert. Vor dem Erstellen einer Nachricht wird zunächst ein Zufälliger 16-Stelliger AES-Schlüssel generiert, der dann mit den öffentlichen Schlüsseln der Empfänger und des Absenders verschlüsselt wird. Diese Pris werden in Base64x codiert und an die Nachricht gehangen.

4.5 Freundschaftsbeziehungen

Ein elementarer Bestandteil von sozialen Netzwerken sind die Freundschaftsbeziehungen. Diese Beziehungen sind Verknüpfungen zwischen den Profilen der Nutzer, die in einigen Varianten noch mit Attributen wie Verwandtschaften belegt werden können. Listen dieser Beziehungen, sogenannte Freundeslisten, dienen den Nutzern dabei nicht nur als eine Adressliste, sondern, da sie in der Regel öffentlich sind, bieten sie auch anderen Nutzern die Funktion das Netzwerk nach Bekannten zu durchsuchen. In vielen Implementierungen von social network services werden auch standardmäßig die Nachrichten der Nutzer aus den Freundeslisten abonniert.

Da diese Freundschaften im hohen Maße die Privatsphäre berühren, sollte auch hier der Schutz bedacht werden. Die Struktur dieser Beziehungen ist jedoch deutlich Komplexer als es bei einfachen Nachrichten oder Profilinformationen der Fall ist, was die Verschlüsselung der Informationen problematisch gestaltet. Grundsätzlich können die Lösungen für dieses Problem in zwei Kategorien eingeteilt werden. Entweder können Freundesbeziehungen dadurch geschützt werden, dass sie nur an autorisierte Nutzer ausgegeben werden. Oder sie werden dadurch geschützt, dass sie wie Nachrichten verschlüsselt werden. Beide Lösungsansätze werden im Folgenden analysiert.

a) Verschlüsselung der Freundesbeziehungen

Grundsätzlich kann ein social network service so implementiert werden, dass die Freundesliste eines Anwenders, nicht durch eine Menge von Relationen in einer Datenbank, sondern durch einen einfachen Text repräsentiert wird. Dieser Text kann dann, genau wie eine Nachricht oder eine Profilinformation, verschlüsselt werden. Der Nutzer hat dann selbst auf die Liste Zugriff und kann entscheiden, welche anderen Nutzer noch Zugriff bekommen. Auch wäre es möglich, dass ein Anwender verschiedene Untermengen seiner Freunde bildet und für jede Menge einzeln entscheidet, welche anderen Nutzer auf sie zugreifen dürfen.

Ein solches Verfahren wäre aber aus zwei Erwägungen problematisch. Zum einen würde so mit der grundsätzlichen Funktionsweise von Freundesbeziehungen gebrochen. Zum anderen wäre die Geheimhaltung gegenüber dem Netzwerkbetreiber nur bedingt gegeben.

Das Wesen der Freundesbeziehungen sieht vor, dass eine Beziehung von beiden Seiten bestätigt wird. Ein Anwender kann erst dann einen anderen als Freund angeben, wenn dieser das vorher betätigt hat. Wenn die Beziehung allerdings verschlüsselt gespeichert ist, hat ein Anwender, der von einem anderen Anwender als Freund genannt wird, gar nicht zwingend Zugriff auf solch eine Beziehung. Auch der Netzwerkbetreiber kann nicht

überprüfen, ob einer Freundesbeziehung beide Seiten zugestimmt haben. Das Wesen der Freundesbeziehung wäre also grundlegend verändert.

Eine weitere wichtige Funktion der Freundeslisten, ist das durchsuchen des Netzwerkes nach weiteren Bekannten. Diese Funktion ist aber nur dann gegeben, wenn die meisten Nutzer ihre Freundeslisten für eine große Menge von anderen Nutzern freigeben. Dies könnte aber auch in der verschlüsselten Variante gewährleistet werden. Zum Beispiel indem eine Funktion zur Verfügung gestellt wird, die eine Freundesliste automatisch für eine große Menge für Nutzer verschlüsselt. Zum Beispiel für die Menge aller Freunde von Freunden.

Auch bei verschlüsselten Freundeslisten ist die Geheimhaltung gegenüber dem Netzwerkbetreiber nur bedingt gegeben. Da bei Nachrichten nur der Text, aber nicht die Absender und Empfänger verschlüsselt sind, ist der Netzwerkbetreiber in jedem Fall in der Lage Beziehungen zwischen den Nutzern zu erfassen. Dies gilt insbesondere dann, wenn von den Anwendern Funktionen genutzt werden, wie das Versenden von Nachrichten an alle Freunde. Nur in dem Fall, dass ein Anwender keine Nachrichten an einen Freund schickt, würde die Beziehung für den Netzwerkbetreiber verborgen bleiben.

b) Ausgeben der Freundesbeziehungen nur an autorisierte Nutzer.

Freundesbeziehungen können auch dadurch geschützt werden, dass sie nicht öffentlich, sondern nur von autorisierten Personen einsehbar sind. Diese Form des Schutzes ist mittlerweile auch in den etablierten sozialen Netzwerken üblich. Die Freundesbeziehungen werden dabei im Klartext als Relationen einer Datenbank repräsentiert, aber nur an autorisierte Anwender ausgegeben. Nutzer können selbstständig angeben, welche fremden Anwender autorisiert sind, ihre Freundeslisten einzusehen.

Dabei sind verschiedene Stufen der Granularisierung der Angaben denkbar. In den großen Netzwerken ist es gebräuchlich, dass den Nutzern die Wahl gegeben wird, ob ihre Freundesliste komplett öffentlich ist, oder ob nur eine bestimmte Nutzergruppe Zugriff hat. Zum Beispiel die Menge der Freunde oder die Menge der Freunde und der Freunde von Freunden. Aber grundsätzlich wäre es auch möglich, für jede Freundesbeziehung einzeln eine Menge von Nutzern zu bestimmen, die diese einsehen kann.

Ein Konflikt entsteht aber dadurch, dass Freundesbeziehungen zweiseitig sind. Angenommen, es gäbe eine Freundesbeziehung zwischen Person A und Person B. Wenn nun beide Personen konträre Angaben machen, ob Person C diese Beziehung sehen darf, tritt ein Konflikt auf.

Die etablierten Netzwerke lösen den Konflikt so, dass die Beziehung bei Ansicht des Profils von Person A angezeigt wird, aber bei der Ansicht von Profil B nicht. Durch diese Lösung

wird aber den Nutzern ein Teil der Selbstbestimmung über ihre Daten genommen. Daher ist es vorzuziehen, dass Freundschaften nur angezeigt werden, wenn beide Seiten dem zustimmen.

Referenzimplementierung

In der Referenzimplementierung werden die Freundesbeziehungen durch die zweite Variante, das Ausgeben der Freundesbeziehungen nur an autorisierte Nutzer, geschützt. Die Anwender können für jede Freundschaft auswählen, ob sie offen oder versteckt ist. Nutzer bekommen nur Freundschaften zwischen fremden Nutzern angezeigt, wenn diese von jeweils beiden beteiligten als nicht versteckt angegeben werden. Außerdem müssen Freundschaften von beiden Seiten bestätigt werden, bevor sie fremden Nutzern angezeigt werden. In der Datenbank wird dies durch eine Relation zwischen zwei Nutzern dargestellt, die das Attribut hide besitzt. Existiert eine Relation nur in eine Richtung, so wird dies als Freundschaftsanfrage gewertet. Existiert die Relation in beide Richtungen, so ist dies eine Freundschaft. Hat bei einem der beiden Einträge das Attribut hide den Wert 1, so wird die Freundschaft keinem fremden Nutzer angezeigt.

So sind die Beziehungen zwischen den Nutzern zwar für den Betreiber einsehbar, wie im obigen Abschnitt erläutert, kann dies jedoch wegen des Netzwerkaufbaus auch durch eine Verschlüsselung der Freundesliste nicht verhindert werden.

Es gibt keine Funktion, Nachrichten automatisch an alle Freunde zu versenden. Für diesen Anwendungsfall sind in der Referenzimplementierung Circles realisiert, eine Methode um Untermengen seiner Freunde zu organisieren.

Die vorliegende Realisierung könnte noch durch eine feinere Granulierung verbessert werden. So, dass die Anwender für jede Freundschaft genau bestimmen können, welche fremden Nutzer die Relation sehen können. Dabei muss aber darauf geachtet werden, dass das Userinterface weiterhin handhabbar bleibt.

Eine weitere Verbesserung wäre es, wenn die Freundschaftsbeziehungen mit verschlüsselbaren Attributen belegt werden können. So wäre es dem Netzbetreiber zwar ersichtlich, dass es eine Relation zwischen den Anwendern gibt, aber nur ausgewählte Nutzer könnten auch die Eigenschaft der Relation sehen. Der Einfachheit halber wurde in der Referenzimplementierung hierauf jedoch verzichtet.

4.6 Untergruppen von Freunden

Damit Nutzer ihre verschiedenen sozialen Statusgruppen organisieren können, bieten Netzwerke Funktionen wie Untergruppen oder Circles an. Sie bieten den Anwendern die Möglichkeit, Nachrichten oder private Daten einfach an eine große Menge von anderen Nutzern zu adressieren.

Den passiven Nutzern bleibt dabei verborgen, in welche Untergruppen sie sortiert werden. Wenn Nachrichten und Daten verschlüsselt werden, können Untergruppen auch zum Teilen von Schlüsseln verwendet werden. Dies ist sinnvoll, um fremden Nutzern auch im Nachhinein Zugriff auf Informationen zu geben.

Auch in der vorliegenden Umsetzung sind Untergruppen realisiert. Anwender können sogenannte Circles anlegen, ihnen fremde Nutzer zusortieren und Nachrichten an sie schicken. Damit Nachrichten und private Daten verschlüsselt gesendet werden können, teilen sich die Circle einen Schlüssel. In der Datenbank wird dafür, für jeden Circle und für jedes Circle-Mitglied ein AES-Schlüssel gespeichert, der jeweils mit dem öffentlichen Schlüssel des Circle-Mitglieds verschlüsselt wird.

Aus den Gründen, die schon im Abschnitt zu Freundesbeziehungen genannt werden, wird in der Referenzimplementierung darauf verzichtet, die Circles an sich zu verschlüsseln. Die Privatsphäre der Nutzer wird dadurch geschützt, dass nur sie selbst Zugriff auf ihre eigenen Circles haben. Den Nutzern, die in die Circles gruppiert werden, bleibt völlig verborgen, in welche Circles sie einsortiert sind. Wenn sie Nachrichten bekommen, die an einen Circle adressiert sind, so gibt die Datenbank als Empfänger immer nur @Circle aus, unabhängig von dem Namen des Circles.

In der Referenzimplementierung wurde auf eine Realisierung der Funktion „Löschen einzelner Nutzer aus einem Circle“ verzichtet. Bisher ist dies nur dadurch möglich, den Circle komplett zu löschen und einen neuen ohne den entsprechenden Nutzer zu erstellen. Sollen die alten Informationen weiterhin nutzbar sein, so müssen neu Schlüssel für die Nachrichten erstellt werden. Diesen Prozess zu automatisieren wäre eine mögliche Verbesserung für die vorliegende Umsetzung.

4.7 Gruppen

Eine weitere wichtige Funktion der social network services sind Gruppen. Anders als Circle oder Untergruppen sind Gruppen nicht auf einen einzelnen Anwender beschränkt, sondern

öffentlich, oder zumindest den Mitgliedern der Gruppen öffentlich. Sie bieten die Funktionen, Informationen über die Gruppe zu speichern und Nachrichten an alle Mitglieder der Gruppe zu senden.

Aus denselben Erwägungen wie bei den Freundesbeziehungen wird in der Referenzimplementierung auf eine komplette Verschlüsselung der Mitglieder verzichtet. Sie werden also auch im Klartext als Relation einer Datenbank gespeichert. Nutzer können die Aufnahme in eine Gruppe beantragen und in eine Gruppe eingeladen werden. Erst wenn es beide Relationen gibt, ist der Nutzer ein Mitglied der Gruppe.

Auch die Gruppen teilen sich einen Schlüssel. Dies hat den Vorteil, dass neuen Mitgliedern auch automatisch die alten Informationen der Gruppe zugänglich sind. Der Nachteil ist, dass Mitglieder nicht mehr aus einer Gruppe entfernt werden können. Dafür ist es in der Referenzimplementierung nötig, dass die Gruppe gelöscht und einen neue ohne das entsprechende Mitglied gegründet wird.

Genau wie ein Circle hat eine Gruppe einen AES-Key, der mit den öffentlichen Schlüsseln der Mitglieder verschlüsselt wird. Er wird bei der Einladung in eine Gruppe erstellt und in der Relation der Einladungen gespeichert. Da jedes Mitglied, sobald es einmal in Besitz des Schlüssels ist, auch Zugriff auf alle Informationen hat, ist auch jedes Mitglied berechtigt, weitere Mitglieder in die Gruppe einzuladen.

Eine sinnvolle Erweiterung der Referenzimplementierung wäre, genau wie bei den Circles, eine Funktion, die das Ausschließen eines Nutzers aus einer Gruppe automatisiert. Sie müsste neue Schlüssel für alle Mitglieder und für alle bisher versendeten Daten erstellen.

4.8 Schlüsselverwaltung

In einem System mit asymmetrischer Verschlüsselung ist es nötig, die öffentlichen Schlüssel zu verwalten. Ein Teilnehmer muss sich vor dem Versenden davon überzeugen, dass ein bestimmter öffentlicher Schlüssel auch wirklich dem Empfänger gehört. Ein Angreifer könnte sonst seinen eigenen Schlüssel für den Schlüssel des wahren Empfängers ausgeben und dann die Nachricht abfangen und entschlüsseln.

Um den Nutzern die gewohnte Benutzerfreundlichkeit aus den etablierten sozialen Netzwerken zu geben, kann eine Schlüsselverwaltung und Überprüfung auch weitgehend automatisiert passieren. Die Nutzer sollten nur einmal anklicken müssen, ob sie einem Schlüssel vertrauen und der Rest sollte automatisch funktionieren. Um dies zu realisieren, könnten Schlüssel, denen der Anwender vertraut, zentral im Netzwerk gespeichert werden. Dann

könnten die Schlüssel vor dem Verschlüsseln überprüft werden und nur bei einem falschen Schlüssel würde eine Fehlermeldung erscheinen und die entsprechende Nachricht nicht versendet.

Allerdings müsste ein Anwender auch die Schlüssel überprüfen, die im Netzwerk als diejenigen gespeichert sind denen er vertraut. Sonst könnte der Netzwerkbetreiber auch fälschlicherweise vorgeben ein Nutzer traue einem Schlüssel und so an die Nutzerdaten kommen. Zur automatischen Überprüfung könnten eine Hashfunktion und das Passwort verwendet werden. Wenn der Anwender einem öffentlichen Schlüssel vertraut, wird der Schlüssel mit dem Passwort konkateniert und davon ein Hashwert gebildet. Der Hashwert wird dann zusammen mit dem Schlüssel im Netzwerk gespeichert. Zur Überprüfung, ob das Netzwerk die richtigen Schlüssel gesendet hat, kann später einfach ein weiteres Mal der Hashwert gebildet und mit dem Wert vom Server verglichen werden. Der Pseudocode 4.1 soll dies verdeutlichen.

```
Wenn ein Nutzer dem öffentlichen Schlüssel (public_key) eines anderen Nutzers
(id_user) vertraut:
string hashcode = sha512(id_user + public_key + passwort)
sende an den Server (id_user, public_key, hashcode)

Zur Überprüfung:
hole vom Server (id_user, public_key, hashcode)
Wenn sha512(id_user + public_key + passwort) == hashcode
gebe ja aus;
sonst gebe nein aus ;
```

Algorithmus 4.1: So könnte eine automatische Schlüsselverwaltung realisiert werden

Aus Zeitgründen ist die automatische Überprüfung der Schlüssel aber noch nicht in der Referenzimplementierung umgesetzt worden. Dies zu nachzuholen wäre eine wichtige Verbesserung.

4.9 Prämissen

Bei der Benutzung eines sicheren sozialen Netzwerkes sollen die privaten Daten der Nutzer geschützt sein. Was konkret heißt, dass nur der Absender und die Empfänger Zugriff auf den Klartext von Nachrichten haben. Auch der Netzwerkbetreiber darf nicht in der Lage sein, auf diese Nachrichten zuzugreifen. Um bei der Nutzung der Referenzimplementie-

rung pribook.com von einer hinreichenden Sicherheit auszugehen, sind einige Prämissen notwendig.

1. Prämisse: Der Webbrowser ist hinreichend sicher.
Um auf eine Installation zu verzichten und den Anwendern gleichzeitig die gewohnt gute Benutzerfreundlichkeit zu geben, ist pribook.com als Webseite konzipiert. Daher ist die Prämisse für die Sicherheit von pribook, dass der Webbrowser, mit dem die Internetseite aufgerufen wird, hinreichend sicher ist.

Insbesondere wird bei der aktuellen Implementierung von pribook.com davon ausgegangen, dass die History des Webbrowsers hinreichend geschützt ist. Variablen wie das Passwort werden nicht explizit gelöscht. Es wird davon ausgegangen, dass der Browser selbstständig dafür sorgt, dass fremde Webseiten, die später aufgerufen werden, keinen Zugriff mehr auf die alten Variablen haben. Außerdem ist eine Prämisse, dass die Generierung von Zufallszahlen im Webbrowser mit JavaScript hinreichend gut funktioniert.

Zwar werden immer wieder Sicherheitslücken bei den gängigen Webbrowsern bekannt, es gibt jedoch quelloffene Internetbrowser, die auch regelmäßig aktualisiert werden. Für praktische Zwecke kann die Prämisse also als gegeben angesehen werden.

2. Prämisse: Das System ist hinreichend sicher.
Genauso ist die Sicherheit des Betriebssystems der Anwender, Voraussetzung für die Sicherheit des sozialen Netzwerkes. Die Sicherheit des Netzwerkes ist nicht gegeben, wenn das Betriebssystem, auf dem der Webbrowser läuft, von Schadsoftware befallen ist. Daher ist eine Prämisse, dass das Betriebssystem hinreichend sicher ist. Da es hier, genau wie bei den Webbrowsern, auch quelloffene Varianten gibt, für die auch regelmäßig Aktualisierungen angeboten werden, kann die Prämisse für praktische Zwecke erfüllt werden.

3. Prämisse: Der Quelltext wird geprüft.
Anders als herkömmliche Verschlüsselungsprogramme wie PGP, wird der Quellcode von pribook bei jedem Aufrufen der Seite neu geladen. Nachdem sich ein Nutzer also das erste Mal von der Korrektheit des Quelltextes überzeugt hat, muss er bei jedem Aufrufen erneut überprüfen, ob es sich tatsächlich noch um den von ihm überprüften Quellcode handelt.

Da die Sicherheit auch gegenüber dem Netzwerkbetreiber garantiert werden soll, reicht es nicht, von einer SSL-Verbindung zwischen Client und Server auszugehen. Denn der Netzwerkbetreiber könnte so, sehr einfach eine falsche Zeile Code senden, die zum Beispiel das Passwort des Nutzers im Klartext an den Server schickt.

Wenn die Verbindung zwischen Client und Server sicher ist, kann es für praktische Zwecke jedoch ausreichen, wenn der Quelltext nur stichprobenartig überprüft wird. Da der Netz-

werkbetreiber davon ausgehen muss, seine Glaubwürdigkeit und damit seine Nutzer zu verlieren, wenn nur einmal bemerkt würde, dass falscher Code von ihm versendet wird.

4. Prämisse: Das Passwort ist sicher.
Da der private RSA-Schlüssel mit dem Passwort der Benutzer verschlüsselt auf dem Server liegt, hängt die Sicherheit hier vom Passwort ab. Der Serverbetreiber kann versuchen, das Passwort mit einem Wörterbuchangriff oder einem Bruteforce-Angriff herauszubekommen. Um diese zu erschweren, wird in der Referenzimplementierung ein 16-stelliger zufälliger Salt verwendet und bei der Erstellung eines Passwortes darauf geachtet, dass es mindestens 8 Zeichen lang ist. Jedoch kann so nicht ausgeschlossen werden, dass die Benutzer sich unsichere Passwörter ausdenken.

5. Prämisse: Die Plug-ins sind sicher.
Die vorliegende Umsetzung verwendet mehrere Plug-ins. Die Sicherheit des gesamten Systems ist nur dann gegeben, wenn sich in diesen Plug-ins keine Sicherheitslücken befinden. Pribook verwendet JQuery, JQuery-Galleria, YAML und das Kryptografie Toolkit Titaniumcore. Eine weitere Voraussetzung für die Sicherheit ist, dass das Kryptografie Toolkit korrekt funktioniert.

4.10 mögliche Angriffsszenarien

Nach dem die Prämissen aufgestellt sind, werden hier mögliche Angriffe auf die Sicherheit der vorliegenden Implementierung diskutiert.

Der Serverbetreiber liest private Nachrichten aus Datenbank.
Alle Nachrichten, die die Nutzer nicht explizit als „öffentlich“ deklarieren, werden verschlüsselt und erst dann an den Server gesendet. Auch mit komplettem Zugriff auf die Datenbank kann also niemand unautorisiert Nachrichten lesen.

Der Serverbetreiber speist fremde Nachrichten ein oder manipuliert Daten.
In der bisherigen Referenzimplementierung ist ein solcher Angriff generell möglich. Jedoch könnte dies von Nutzern bemerkt werden. Spätestens durch die Kommunikation über einen anderen Kanal. Der Netzwerkbetreiber würde also so sein Vertrauen riskieren was solch eine Art des Angriffes für praktische Zwecke ausreichen unwahrscheinlich werden lässt.

Die bestehende Referenzimplementierung kann aber auch mit wenig Aufwand so modifiziert werden, dass mit den vorhandenen RSA-Algorithmen auch automatisch die Nachrichten signiert werden. Auch die Überprüfung der Signaturen könnte leicht automatisiert werden,

so, dass dem Nutzer automatisch angezeigt würde, sollte eine Nachricht manipuliert oder fälschlicherweise eingespeist sein.

Der Serverbetreiber löscht Nutzer oder Nachrichten.
Anders als bei Peer-to-Peer Varianten von sozialen Netzwerken ist es in der vorliegenden Implementierung möglich, dass Nutzer oder Nachrichten vom Netzwerkbetreiber gelöscht werden. Ein solcher Angriff kann aber nicht geschehn, ohne dass dies von den Anwendern bemerkt wird.

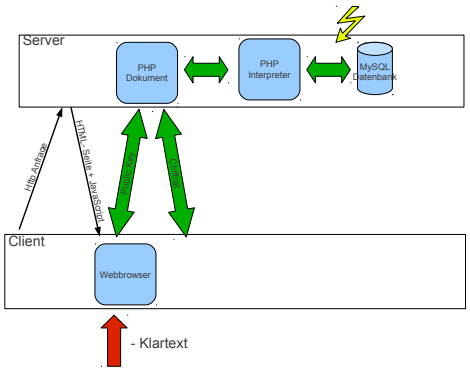


Abbildung 4.6: Angriff des Serverbetreibers

Der Serverbetreiber speist falschen Code ein.
Eine Prämisse ist es, dass der Quelltext vor dem Eingeben des Passwortes und des Benutzernamens überprüft wird. Nur so kann sichergestellt werden, dass auch wirklich nur verschlüsselte Daten zum Server gesendet werden. Wie schon bei den Prämissen erwähnt, kann es für praktische Zwecke aber hinreichend sein, wenn der Code nur stichprobenartig überprüft wird.

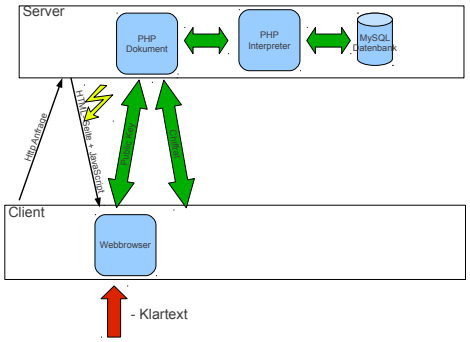


Abbildung 4.7: Server sendet falschen Code

Um die Handhabbarkeit der Referenzimplementierung hier zu verbessern, wäre eine mögliche Erweiterung, ein Browser-Plug-in zu entwickeln, welches automatisch den Code der Internetseite pribook.com überprüft. Der Anwender müsste sich dann nur einmal von der Richtigkeit des Quelltextes überzeugen und könnte dann einen Hashwert des Codes in das Plug-in eingeben. Bei jedem späteren Aufrufen der Seite würde

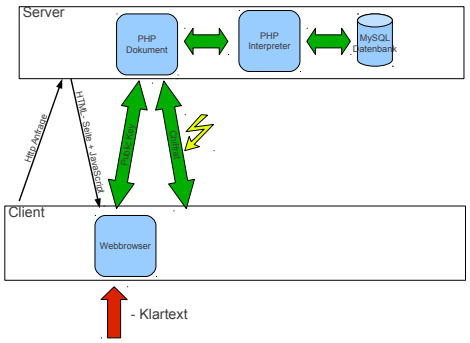


Abbildung 4.8: Codeinjection über Nutzerdaten

dann der aktuell geladene Code mit den Hashwerten verglichen und bei Übereinstimmung ein Okay-Signal ausgegeben.

Unbefugte rufen Metainformationen ab, oder manipulieren diese.
Bei jeder Abfrage, die ein Nutzer an den Server macht, muss er seinen Benutzernamen und einen Hashwert seines Passwortes an den Server schicken. Der Server überprüft den Hashwert und gibt die gewünschten Informationen nur bei erfolgreicher Prüfung aus. Nicht autorisierte Benutzer kommen also nicht an Informationen, die nicht für sie bestimmt sind. Sie können so auch keine Metainformationen, wie Freundschaften oder Empfänger von Nachrichten einsehen. Die Verbindung zwischen Server und Client kann auch leicht mit SSL verschlüsselt werden, umso ein Abfangen des Hashwertes unmöglich zu machen.

Der Serverbetreiber wertet Metainformationen aus.
Da nur die Inhalte der Nachrichten, nicht aber der Absender, die Empfänger sowie das Versendedatum verschlüsselt sind, kann der Netzwerkbetreiber auf diese Informationen zugreifen. Wie in den vorigen Abschnitten diskutiert, sind im pribook Freundesrelationen, Circle und Gruppen auch im Klartext in der Datenbank abgespeichert. Der Betreiber kann also auf alle diese Daten zugreifen.

Diese Daten können aber nur mit Pseudonymen und nicht mit persönlichen Profildaten verknüpft werden. In [52] werden Möglichkeiten thematisiert, von solchen Metadaten Rückschlüsse auf die Personen zu ziehen.

Falscher Code wird über die Nutzerdaten eingefügt (Code Injection).
Ein denkbarer Angriff, der sowohl vom Serverbetreiber als auch von den Nutzern durchführbar wäre, ist Code Injection über die Nutzerdaten. Nachdem der reguläre Code überprüft wurde und der Anwender sein Benutzername und das Passwort eingegeben hat, werden vom Server die Nachrichten für den Nutzer heruntergeladen. In diesen Nachrichten könnte sich auch Code verstecken.

Die Nachrichten und alle anderen Informationen, werden als JSON mit der JQuery-Funktion post() vom Server abgerufen und dann mit der JQuery-Funktion text() in die Internetseite eingebaut. Code, der sich in einem JSON-Objekt befindet, wird so nicht ausgeführt. Ein Angriff durch Code Injektion ist also nicht möglich.

Kapitel 5

Referenzimplementierung: pribook.com

In diesem Kapitel wird die Referenzimplementierung pribook.com vorgestellt. Zunächst wird der gesamte Aufbau, dann die Datenbank und schließlich die wichtigsten Anwendungsfälle erläutert.

5.1 Aufbau

Pribook ist nach der Architektur aufgebaut, wie sie in Abbildung 4.1 gezeigt wird. Pribook umfasst eine MySQL-Datenbank, die in Abschnitt 5.2 näher beleuchtet wird, sowie einige PHP-Dateien zum Zugriff auf die Datenbank sowie die Datei index.html. Die Datei index.html ist das HTML-Grundgerüst, das vom Webbrowser geladen wird, es inkludiert die JavaScript-Dateien pribook0.3.1.js und Cryption.js, die das eigentliche Programm darstellen. Außerdem inkludiert die Datei index.html die Bibliotheken JQuery, Galleria und die Kryptografie-Bibliothek Titaniumcore Project.

Die Kryptografie-Bibliothek Titaniumcore Project [34] von Atsushi Oka ist eine objektorientierte JavaScript-Bibliothek, die einige wichtige Verschlüsselungsalgorithmen realisiert. In Pribook werden davon die Algorithmen Rijndael, RSA und SHA512 benötigt.

Die Abbildung 5.1 zeigt den Zusammenhang der Dateien und ihre Methoden. Cryption.js und Pribook0.3.1.js werden über index.html inkludiert und zusammen mit dem HTML-Grundgerüst in den Webbrowser geladen und dann ausgeführt. Die PHP-Dateien liegen auf dem Server und werden mit einer AJAX-Abfrage gestartet und geben den Wiederga-

bewert als JSON-Object an den Client zurück. Es handelt sich nicht um objektorientierte Programmierung, da die Klassen nicht instanziiert werden. Dennoch wurde diese Art des Aufbaus und der Darstellung der Übersichtlichkeit halber gewählt.

Die Methoden der PHP-Dateien dienen dazu, in der Datenbank zu lesen und zu schreiben. Zum Beispiel speichert die Methode sendMessage() eine neue Nachricht in der Datenbank, während die Methode getMessage() die neusten Nachrichten an einen Nutzer aus der Datenbank holt. Die genauen Funktionsweisen der Methoden der PHP-Dateien sind nicht weiter von besonderer Bedeutung, weswegen sie hier nicht im einzelnen erläutert werden. In der Regel ergibt sich ihre Funktionsweise aus dem Kontext. Für genauere Informationen sei hier auf den Quelltext verwiesen.

Die Methoden der PHP-Dateien auf dem Server werden durch eine AJAX-Abfrage in Pribook0.3.1.js aufgerufen. Dabei wird als Parameter ein String mode übergeben, der den Namen der aufzurufenden Funktion enthält. Außerdem wird während jedes Funktionsaufrufes die Funktion authorization(string id_user, string pw_test) aufgerufen und nur bei erfolgreicher Authentifizierung die gewünschten Daten an den Client gesendet. Deshalb werden bei jedem Aufruf einer Funktion neben den Parametern der aufgerufenen Funktion auch die Parameter string mode, string id_user und string pw_test übergeben. Der Übersichtlichkeit halber wird bei dem Klassendiagramm aber darauf verzichtet. Eine Ausnahme bildet hierbei nur die Funktion getSalt() von der Datei Key.php. Da diese den Salt eines Nutzers wiedergibt, ist sie vor der Authentifizierung nötig und kann daher ohne Prüfung aufgerufen werden. Die AJAX-Aufrufe werden mit der Methode JQuery.post() aus der JQuery-Bibliothek realisiert. Als Wiedergabetyp wird stets ein Object in JSON (JavaScript Object Notation) erwartet, in dem das Ergebnis codiert ist. Zum Beispiel sieht der Aufruf der Funktion getMessage() von der Datei Message.php wie in Algorithmus 5.1 dargestellt aus.

Die genauen Funktionsweisen der wichtigsten Funktionen von Pribook0.3.1.js sowie von Cryption.js wird in Abschnitt 5.3 erläutert.

```
1: $.post( “Message.php“, {
2: id_user_1 : “Bob “,
3: no : 0,
4: pw_test : pw_test,
5: mode : “ getMessage “
6: }, Funktion( ), “json “);
```

Algorithmus 5.1: Aufruf der Methode getMessage()

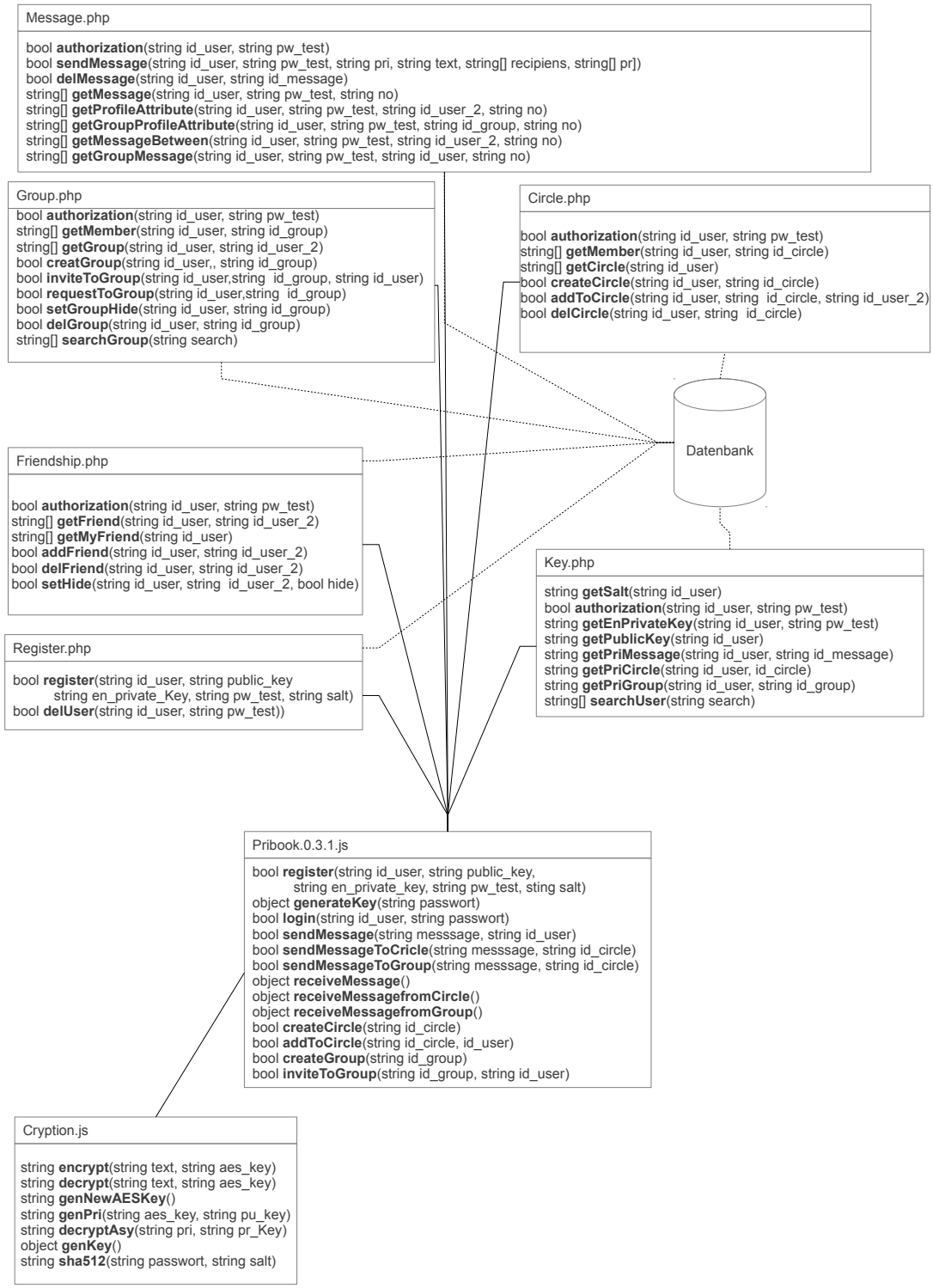


Abbildung 5.1: Übersicht der Dateien

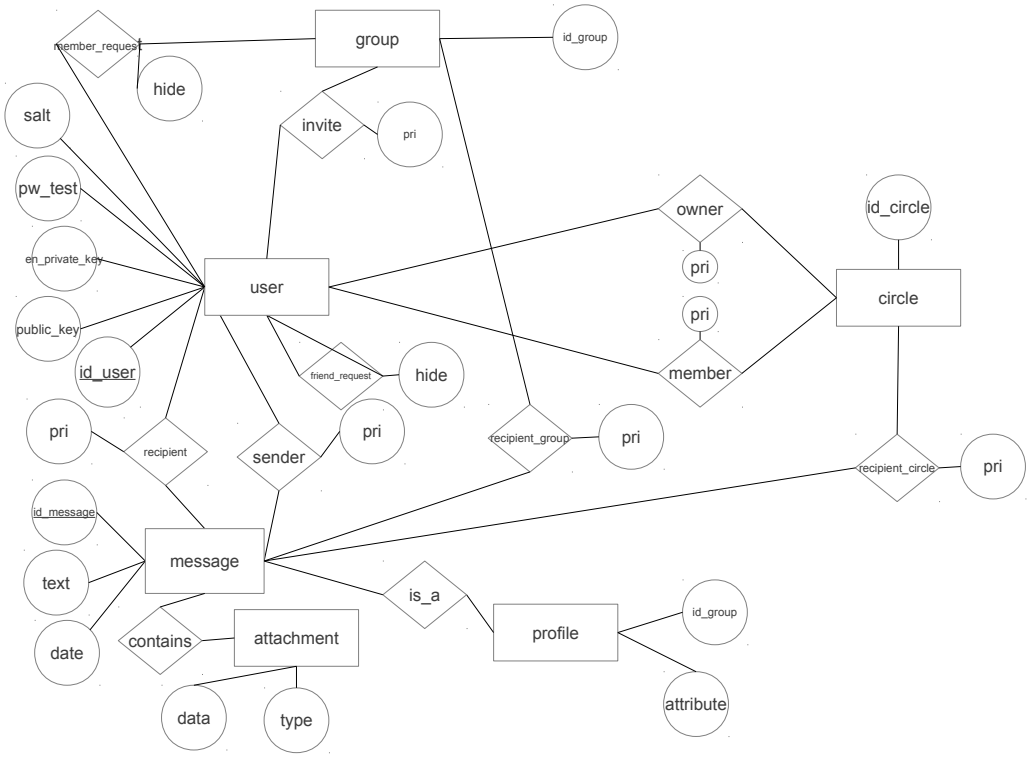


Abbildung 5.2: Datenbank als ER-Modell

5.2 Datenbank

Wie aus der Architektur in Abbildung 4.1 hervorgeht, werden alle Informationen zentral in einer Datenbank gespeichert. Dafür wird in der Referenzimplementierung eine zentrale MySQL-Datenbank verwendet. MySQL wurde gewählt, da es das weltweit am weitesten verbreitetste relationale Datenbankverwaltungssystem ist und als Open-Source-Software zur Verfügung steht. Die Datenbank umfasst die folgenden Entitäten user, message, attachment, profile, circle und group. In Abbildung 5.2 ist die Datenbank als ER Modell dargestellt.

5.2.1 Objekt user

Die Entität user repräsentiert die Benutzer des sozialen Netzwerkes und hat die folgenden Attribute.

- id_user: Beinhaltet eine eindeutige ID für den Nutzer, die er selbst bei der Registrierung wählen kann. Es ist der Key des Objektes user. Typ: varchar(255)

- salt: Beinhaltet einen Salt um einen Hashwert zu bilden. Typ: binary
- pw_test: Speichert einen SHA512-Wert vom Passwort des Nutzers und dem Salt. Typ: binary
- public_key: Speichert den öffentlichen Schlüssel des Nutzers, der bei der Registrierung generiert wird. Typ: binary
- en_private_key: Speichert den privaten Key des Nutzers in verschlüsselter Form, der bei der Registrierung verschlüsselt wird. Typ:binary

Das Objekt user hat die folgenden Relationen:

- member_request : Nutzer können Mitgliedsanfragen an Gruppen stellen. Und diese mit dem Attribut hide vor anderen Nutzern verstecken.
- invite: Nutzer können in Gruppen eingeladen werden. In dem Attribut pri wird der gemeinsame Schlüssel der Gruppe, für den Nutzer verschlüsselt, gespeichert.
- owner: Nutzer können einen Circle besitzen. In dem Attribut pri wird der gemeinsame Schlüssel des Circle, für den Nutzer verschlüsselt gespeichert.
- member: Nutzer können Mitglied in einem Circle sein. In dem Atrribut pri wird der gemeinsame Schlüssel des Circle, für den Nutzer verschlüsselt gespeichert.
- friend_request: Nutzer können an andere Nutzer eine Freundschaftsanfrage stellen und diese mit dem Attribut hide verstecken.
- sender: Nutzer können Nachrichten senden. Um sie nachher wieder lesen zu können, wird in dem Attribut pri der Schlüssel für die Nachricht, für den Nutzer verschlüsselt, gespeichert.
- recipient: Nutzer können Empfänger einer Nachricht sein. Um die verschlüsselte Nachricht lesen zu können, wird in dem Attribut pri der Schlüssel für die Nachricht, für den Nutzer verschlüsselt, gespeichert.

5.2.2 Objekt group

Das Objekt group repräsentiert die Gruppen im sozialen Netzwerk und umfasst das Attribut:

- id_group: Beinhaltet eine eindeutige ID für die Gruppe, die der Gruppengründer bei der Gruppengründung wählen kann. Es ist der Key des Objektes group. Typ: varchar(255)

Die Entität group umfasst die folgenden Relationen:

- recipient_group: Gruppen können Empfänger von Nachrichten sein. Um die verschlüsselte Nachricht lesen zu können, wird in dem Attribut pri der Schlüssel für die Nachricht, für die Gruppe Nutzer verschlüsselt, gespeichert.
- invite, member_request: Siehe user

5.2.3 Objekt circle

Das Objekt circle repräsentiert die Circles des sozialen Netzwerkes und umfasst das Attribut:

- id_circle: Beinhaltet eine ID für den Circle, die für je einen Circlebesitzer eindeutig ist. Die ID für den Circle wird bei der erstellung des Circles ausgewählt. Die ID ist der Key des Objektes circle. Typ: varchar(255)

Die Entität circle umfasst die folgenden Relationen:

- recipient_group: Circle können Empfänger von Nachrichten sein. Um die verschlüsselte Nachricht lesen zu können, wird in dem Attribut pri der Schlüssel für die Nachricht, für den Circle verschlüsselt, gespeichert.
- owner, member: Siehe user.

5.2.4 Objekt message

Das Objekt message repräsentiert die Nachrichten des sozialen Netzwerkes und umfasst die Attribute:

- id_message: Beinhaltet eine Nummer für die Nachricht die automatisch vergeben wird. Die ID ist der Key des Objektes message. Typ: int(255)
- text: Der Text der Nachricht. Typ: binary

- date: Das Datum der Nachricht, wird vom Server beim Eintreffen der Nachricht gesetzt.

Die Entität message umfasst die folgenden Relationen:

- recipient, sender: siehe user.
- recipient_group: siehe group.
- recipient_circle: siehe circle.
- contains: Nachrichten können einen Anhang haben.
- is_a: Profilinformationen sind Nachrichten.

5.2.5 Objekt attachment

Das Objekt attachment repräsentiert den Anhang von Nachrichten des sozialen Netzwerkes und umfasst die Attribute:

- data: Der Anhang. Typ: binary
- typ: Der Typ des Anhangs. Typ:varchar(255)

Die Entität attachment umfasst die folgende Relation:

- Contains: Siehe message

5.2.6 Objekt profile

Das Objekt profile repräsentiert die Profilinformationen. Es umfasst die folgenden Attribut:

- attribute: Der Name des Attributes für die Profilinformation. Typ: Binary.

Das Objekt hat die folgenden Relationen:

- id_group: Gruppe auf die sich das Attribut bezieht.
- is_a: Siehe message.

5.3 Funktionen

In diesem Abschnitt werden die wichtigsten Funktionen von Pribook0.3.1.js und Cryption.js erläutert. Dabei werden nur die Sachen von Interesse dargestellt und auf den Rest verzichtet. Insbesondere wird der Bereich des Userinterface hier vernachlässigt. Für genauere Informationen über die Gestaltung des Userinterface wird hier auf den Quelltext verwiesen. Des Weiteren wird die Unterscheidung zwischen Nachricht, Nutzerprofil und Gruppenprofil vernachlässigt. Da Gruppenprofil und Nutzerprofil auch Nachrichten sind (siehe Datenbank), werden in diesem Abschnitt nur Nachrichten behandelt.

Dieser Abschnitt ist in die Unterabschnitte Registrierung und Anmeldung, Nachrichten, Circle, Gruppen und Freundschaften unterteilt. In den Unterabschnitten werden die Funktionen für die jeweiligen Anwendungsfälle erläutert.

Im Pseudocode leiten sich die Namen der Variablen aus den Entitäten und Attributen aus der Datenbank ab und zeigen so gleich, wie der Inhalt der Variablen aussehen soll. Ein Aufruf der Methoden der PHP-Dateien wird durch [Name der PHP-Datei].[Name der Methode] repräsentiert

5.3.1 Registrierung und Anmeldung

In diesem Abschnitt lassen sich der Anwendungsfall Registrierung (register()), also die Erstellung eines neuen Nutzerkontos sowie der Anwendungsfall Anmeldung (login()) identifizieren.

Registrierung
Zur Benutzung von Pribook braucht jeder Nutzer einen Account. Dieser wird durch eine Registrierung erstellt. Zur Registrierung wird ein Name benötigt, der noch nicht vergeben ist sowie ein Passwort mit mindestens 8 Stellen. Ein 1024 Bit langes RSA-Schlüsselpaar wird erstellt. Der private Schlüssel wird mit dem Passwort verschlüsselt. Ein Salt wird als 16-stellige Zufallszahl gebildet. Zum späteren Authentifizieren wird ein Hashwert vom Passwort und dem Salt gebildet. Der Nutzer wird mit der ID, dem öffentlichen Schlüssel, dem verschlüsselten privaten Schlüssel sowie dem Hashwert vom Passwort an der Datenbank angemeldet. Siehe Algorithmus 5.2.

Anmeldung
Vor jeder Nutzung muss sich ein Nutzer mit seinem Passwort und seiner ID einloggen. Der Salt des Nutzers wird aus der Datenbank geholt. Es wird ein Hashwert vom Passwort und dem Salt gebildet. Mit der Methode Key.getEnPrivatKey wird der Hashwert zum

Eingabe: string passwort, string id_user

```
1: (string public_key, string private_key) = Cryption.generate(1024)
2: string en_privat_key = Cryption.encrypt(private_key, passwort)
3: string salt = Random(16)
4: string pw_test = Cryption.sha512(passwort + salt)
5: Register.register(id_user, public_key, en_private_key, pw_test)
```

Algorithmus 5.2: register() - Registrierung

Server gesendet und bei Übereinstimmung mit dem bei der Registrierung erstellten Hashwert wird der verschlüsselte private Schlüssel wird aus der Datenbank zurückgegeben. Der Hashwert wird nach der Anmeldung im lokalen Speicher behalten und bei allen folgenden Operationen zur Authentifizierung mitgesendet. Dies verhindert, dass unbefugte Nutzer die Kommunikationsumstände der verschlüsselten Daten oder die unverschlüsselten Daten, wie die Freundeslisten, einsehen können. Bei den nachfolgenden Funktionen wird diese Authentifizierung aber nicht jedes Mal angegeben. Auch der private Schlüssel wird für die folgenden Funktionen im lokalen Speicher gehalten. Siehe Algorithmus 5.3.

Eingabe: string passwort, string id_user

```
1: string salt = Key.getSalt(id_user)
2: string pw_test = Cryption.sha512(passwort + salt)
3: string en_private_key = Key.getEnPrivateKey(id_user, pwtest)
4: string private_key = Cryption.decrypt(en_private_key, passwort)
```

Algorithmus 5.3: login() - Anmeldung

5.3.2 Nachrichten

In der Rubrik Nachrichten gibt es die Anwendungsfälle Nachrichten Versenden (sendMessage()) und Nachrichten empfangen (getMessage()).

Nachricht versenden

Nach der Anmeldung können Nutzer Nachrichten verschicken. Dazu müssen sie den Namen (ID) des Empfängers sowie einen Text eingeben. Ein AES-Schlüssel wird aus 16 Zufallszahlen gebildet. Der Public Key des Nutzers wird aus der Datenbank geholt. Der AES-Key wird mit dem öffentlichen Schlüssel per RSA verschlüsselt. Der Text der Nachricht wird per AES-verschlüsselt. Die Nachricht wird in der Datenbank gespeichert. In der Referenzimplementierung können Nachrichten an beliebig viele Empfänger versendet werden. Das Verfahren wird dann einfach für jeden Empfänger wiederholt. Der Einfachheit halber wird hier jedoch nur das Verfahren für einen Empfänger erläutertet. Siehe Algorithmus 5.4.

Eingabe: string message, sting id_user

```
1: string aes_key = Random(16)
2: string public_key = Key.getPublicKey(id_user)
3: string pri = Cryption.genPri(aes_key, public_key)
4: string text = Cryption.encrypt(message, aes_key)
5: Message.sendMessage(text, id_user, pri)
```

Algorithmus 5.4: sendMessage() - Nachricht versenden

Nachrichten empfangen und entschlüsseln

Nach der Anmeldung ist der private Schlüssel im lokalen Speicher. Zum Empfangen von Nachrichten muss der Nutzer also keine weitere Eingabe machen. Eine empfangene Nachricht besteht aus dem verschlüsselten Text und dem angehängten Pri. Der AES-Schlüssel der Nachricht wird gewonnen, in dem der Pri mit dem privaten Schlüssel entschlüsselt wird. Die Nachricht wird durch das entschlüsseln des Textes mit dem AES-Schlüssel gewonnen. In der Referenzimplementierung werden gleich 10 Nachrichten auf einmal Empfangen und wie im Abschnitt Datenbank geschrieben, enthalten die Nachrichten noch das Datum, Absender und Empfänger. Der Einfachheit halber wird im Pseudocode aber auf diese Details verzichtet. Siehe Algorithmus 5.5.

Eingabe: string private_key, string id_user

```
1: (string text, string pri) = getMessage(id_user)
2: string public_key = Key.getPublicKey(id_user)
3: string message = Cryption.decrypt(text, aes_key)
```

Algorithmus 5.5: getMessage() - Nachricht empfangen und entschlüsseln

5.3.3 Circle

Circle dienen dazu verschiedene Untergruppen von Freunden zu organisieren siehe dazu auch Abschnitt 4.6. Der Unterpunkt Circle umfasst die Anwendungsfälle: Erstellung eines Circle (createCircle()), Nutzer in Circle einfügen (addToCircle()), Nachricht an Circle versenden (sendMessageToCircle()) und Nachricht an Circle empfangen (getMessageFromCircle()).

Erstellung eines Circle

Zur Erstellung eines Circle werden eine Circle-ID und die ID des Nutzers benötigt. Ein Nutzer kann keine zwei Circle mit derselben Circle-ID haben. Ein AES-Schlüssel wird aus 16 Zufallszahlen gebildet. Der Public Key des Nutzers wird aus der Datenbank geholt. Der

AES-Key wird mit dem öffentlichen Schlüssel per RSA verschlüsselt. Der Circle wird mit dem Pri in der Datenbank gespeichert. Siehe Algorithmus 5.6.

Eingabe: string id_user, string id_circle

```
1: string aes_key = Random(16)
2: string public_key = Key.getPublicKey(id_user)
3: string pri = Cryption.genPri(public_key, aes_key)
4: Circle.createCircle(id_user, id_circle, pri)
```

Algorithmus 5.6: createCirlce() - Erstellung eines Circle

Nutzer in Circle einfügen

Um einen andere Nutzer in einen Circle einzufügen wird die ID dieses Nutzers (id_user_member) die ID des Circles und die ID des Nutzers selbst gebraucht. Der Pri des Circle wird aus der Datenbank geholt. Der AES-Key des Circle wird mit dem privaten Schlüssel des Anwenders entschlüsselt. Der öffentliche Schlüssel des fremden Nutzers wird aus der Datenbank geholt. Der AES-Key des Cricles wird mit dem öffentlichen Schlüssel des fremden Users verschlüsselt. In der Datenbank wird die Relation vom Circle und Nutzer gespeichert. Siehe Algorithmus 5.7.

Eingabe: string id_user_owner, string id_user_member, string id_circle

```
1: string pri = getPriCircle(id_user_owner, id_circle)
2: string aes_key = Cryption.decryptAsy(pri, private_key)
3: string public_key = Key.getPublicKey(id_user_member)
4: string new_pri = Cryption.genPri(aes_key, public_key)
5: Circle.addToCircle(id_user_owner, id_circle, id_user_member, new_pri)
```

Algorithmus 5.7: addToCircle() - Nutzer in Circle einfügen

Nachrichten an Circle versenden

Zum Versenden einer Nachricht an einen Circle wird als Eingabe gebraucht: Die ID des Nutzers, die ID des Circle und die Nachricht. Ein AES-Schlüssel wird aus 16 Zufallszahlen gebildet. Der Pri des Circle wird aus der Datenbank geholt. Der AES-Key des Circle wird mit dem privaten Schlüssel des Anwenders entschlüsselt. Der AES-Key der Nachricht wird mit dem AES-Key des Circle verschlüsselt. Der Text der Nachricht wird per AES-verschlüsselt. Die Nachricht wird in der Datenbank gespeichert. Siehe Algorithmus 5.8.

Nachrichten an Circle empfangen und entschlüsseln

Direkt nach dem Einloggen werden auch Nachrichten empfangen, die an einen Circle adressiert sind, in dem der Nutzer Mitglied ist. Dafür wird nur die ID des Nutzers und sein pri-

Eingabe: string id_user, string id_circle, string id_message

1: string aes_key_message = Random(16)
2: string pri = Key.getPriCircle(id_user_owner, id_circle)
3: string aes_key = Cryption.decryptAsy(pri, private_Key)
4: string new_pri = .Cryption.encrypt(aes_key, aes_key_message)
5: string text = Cryption.encrypt(message, aes_key)
6: Message.sendMessage(text, id_circle, id_user, new_pri)

Algorithmus 5.8: sendMessageToCircle() - Nachrichten an Circle versenden

vater Schlüssel gebraucht die nach der Anmeldung verfügbar sind. Die Nachricht inklusive Pri, der ID des Circle und der ID des Besitzers des Circle wird aus der Datenbank abgerufen. Der Pri des Circle wird aus der Datenbank geholt. Der AES-Key des Circle wird mit dem privaten Schlüssel des Anwenders entschlüsselt. Der AES-Key der Nachricht wird mit dem AES-Key des Circle entschlüsselt. Die Nachricht wird mit ihrem AES-Schlüssel entschlüsselt. Hinweis: Voraussetzung ist natürlich, dass die abgerufene Nachricht eine Nachricht ist, die an einen Circle gesendet wird. In der realen Umsetzung werden immer gleich 10 Nachrichten abgerufen, die dann nacheinander entschlüsselt werden. Vor dem Entschlüsseln wird festgestellt, ob die Nachricht an einen Nutzer, eine Gruppe oder einen Circle adressiert ist, und wird dann auf die entsprechende weise entschlüsselt. Siehe Algorithmus 5.9.

Eingabe: string id_user, string private_key

1: (string text, string pri, string id_circle, string id_user_owner) = Message.getMessage(id_user)
2: string pri_circle = Key.getPriCircle(id_user_owner, id_circle, id_user)
3: string aes_key = Crytion.decryptAsy(pri, private_key)
4: string aes_key_message = Cryption.decrypt(pri_message, aes_key)
5: string message = Cryption.decrypt(message, aes_key_message)

Algorithmus 5.9: getMessageToCircle() - Nachrichten an Circle empfangen und entschlüsseln

5.3.4 Gruppen

In diesem Abschnitt werden die Funktionen für die folgenden Anwendungsfälle erläutert: Erstellung einer Gruppe (createGroup()), Nutzer in Gruppe einladen(inviteToGroup()), Nachricht an Gruppe versenden (sendMessageToGroup()), Nachricht an Gruppe empfangen (reciveMessageFromGroup()).

Erstellung einer Gruppe

Wenn ein Nutzer eine Gruppe erstellt, wird zunächst ein AES-Schlüssel aus 16 Zufallszahlen gebildet. Der Public Key des Nutzers wird aus der Datenbank geholt. Der AES-Key wird mit dem öffentlichen Schlüssel des Nutzers per RSA verschlüsselt. Die Gruppe wird mit dem Pri in der Datenbank gespeichert. Auf dem Server wird überprüft, ob es die Gruppe schon gibt, und dann eine Gruppe Einladung und eine Gruppenanfrage für den Nutzer erstellt. Siehe Algorithmus 5.10.

Eingabe: string id_user, string id_gruppe

```
1: string aes_key = Random(16)
2: string public_key = Key.getPublicKey(id_user)
3: string pri = Cryption.encrypt(public_key, aes_key)
4: Group.createGroup(id_user, id_group, pri)
```

Algorithmus 5.10: createGroup() - Pseudocode für die Erstellung einer Gruppe

Nutzer in Gruppe einladen

Um einen Nutzer in eine Gruppe einzuladen, müssen die ID des einzuladenden Nutzers (id_user_member) sowie die ID der Gruppe angegeben werden. Außerdem ist die ID des einladenden Nutzers sowie sein privater Schlüssel nötig. Der Pri der Gruppe wird aus der Datenbank geholt. Der AES-Key der Gruppe wird mit dem privaten Schlüssel des Anwenders entschlüsselt. Der öffentliche Schlüssel des einzuladenden Nutzers wird aus der Datenbank geholt. Mit diesem öffentlichen Schlüssel wird der AES-Key der Gruppe verschlüsselt. In der Datenbank wird die Relation vom Circle und Nutzer gespeichert. Siehe Algorithmus 5.11.

Eingabe: string id_user, string id_user_member, string id_group, string private_key

```
1: string pri = Key.getPriGroup(id_user_owner, id_group)
2: string aes_key = Cryption.decrypt(pri, private_key)
3: string public_key = Key.getPublicKey(id_user_member)
4: string new_pri = Cryption.encrypt(aes_key, public_key)
5: Group.inviteToGroup(id_group, id_user_member, new_pri)
```

Algorithmus 5.11: inviteToGroup() - Einladung eines Nutzers in eine Gruppe

Nachrichten an Gruppen versenden

Wie bei einer normalen Nachricht auch wird zunächst ein zufälliger 16-stelliger AES-Schlüssel gebildet und die Nachricht verschlüsselt. Um aus dem AES-Schlüssel einen Pri zu bilden, wird zunächst der AES-Schlüssel der Gruppe gebraucht. Dieser wird errechnet, in dem zunächst der Pri der Gruppe aus der Datenbank geholt wird und dann mit dem privaten Schlüssel des Nutzers entschlüsselt wird. Mit dem AES-Schlüssel der Gruppe wird

nun der AES-Schlüssel der Nachricht verschlüsselt und als Pri an die Nachricht gehangen. Siehe Algorithmus 5.12.

Eingabe: string id_user, string id_group, string message,

```
1: string aes_key_message = Random(16)
2: string text = Cryption.encrypt(message, aes_key)
3: string pri = Key.getPriGroup(id_user, id_group)
4: string aes_key_group = Cryption.decrypt(pri, private_Key)
5: string new_pri = Cryption.encrypt(aes_key_messege, aes_key_group)
6: Message.sendMessage(text, id_group, id_user, new_pri)
```

Algorithmus 5.12: sendMessageToGroup() - Nachrichten an Gruppen versenden

Nachrichten an Gruppe empfangen und entschlüsseln

Direkt nach dem Einloggen werden auch Nachrichten empfangen, die an eine Gruppe adressiert sind, in dem der Nutzer Mitglied ist. Dafür wird nur die ID des Nutzers und sein privater Schlüssel gebraucht die nach der Anmeldung verfügbar sind. Die Nachricht inklusive Pri und der ID der Gruppe wird aus der Datenbank abgerufen. Nun wird der Pri der Gruppe aus dem Netzwerk geholt. Dieser wird mit dem privaten Schlüssel des Nutzers entschlüsselt. Daraus entsteht der AES-Schlüssel der Gruppe mit diesem kann der AES-Schlüssel der Nachricht entschlüsselt werden. Mit diesem wird schließlich die Nachricht entschlüsselt. Hinweis: Voraussetzung ist natürlich, dass die abgerufene Nachricht eine Nachricht ist, die an eine Gruppe gesendet wird. In der realen Umsetzung werden immer gleich 10 Nachrichten abgerufen, die dann nacheinander entschlüsselt werden. Vor dem Entschlüsseln wird festgestellt, ob die Nachricht an einen Nutzer, eine Gruppe oder einen Circle adressiert ist, und wird dann auf die entsprechende weise entschlüsselt. Siehe Algorithmus 5.13.

Eingabe: string id_user, string private_key

```
1: (string text, string pri_message, string id_group) = Message.getMessage(id_user)
2: string pri_group = Key.getPriGroup(id_user, id_group)
3: string aes_key = Cryption.decryptAsy(pri_group, private_key)
4: aes_key_message = Cryption.decrypt(pri_message, aes_key)
5: string message = Cryption.decrypt( message, aes_key_message)
```

Algorithmus 5.13: receiveMessagefromGroup() - Nachrichten an Gruppe empfangen und entschlüsseln

Kapitel 6

Handbuch

Dieses Kapitel ist ein Handbuch für das soziale Netzwerk pribook.com. Hier wird beschrieben, wie die prototypische Umsetzung für den Endbenutzer aussieht und wie die einzelnen Funktionen bedient werden.

6.1 Einleitung

Pribook ist ein Social Network Service, der die Privatsphäre der Nutzer bewahrt. Mit Hilfe des asymmetrischen Verschlüsselungsverfahren RSA werden die Inhalte von Nachrichten und Profilinformationen vor dem Versenden verschlüsselt. Auf den Inhalt der Nachrichten können dann nur der Absender und die Empfänger zugreifen. Da die Daten schon im Webbrowser verschlüsselt werden, noch bevor sie an den Server geschickt werden, hat selbst der Netzwerkbetreiber keine Chance auf die Daten zuzugreifen.

Bei der Registrierung wird für jeden Nutzer ein RSA-Schlüsselpaar erstellt. Der private Schlüssel wird mit dem Passwort des Nutzers verschlüsselt und dann zusammen mit dem öffentlichen Schlüssel an das Netzwerk gesendet. So muss sich der Nutzer lediglich sein eigenes Passwort merken. Beim Verschicken einer Nachricht werden automatisch die öffentlichen Schlüssel der Empfänger vom Server geholt und die Nachricht noch vor dem versenden verschlüsselt. Empfangene Nachrichten werden mit dem privaten Schlüssel der Nutzer entschlüsselt, der selbst zunächst vom Server geholt und dann mit dem eingegebenen Passwort entschlüsselt wird.

Abbildung 6.1: pribook.com: Anmeldebildschirm

6.2 Voraussetzungen

Zum Starten von Pribook muss ein Webbrowser geöffnet und die URL http://www.pribook.com in die Adressleiste eingegeben werden. Pribook wurde mit den folgenden Webbrowsern getestet:

- Firefox 10.0
- Chromium 16.0

Die Nutzung von anderen Webbrowsern kann dazu führen, dass Pribook nicht richtig ausgeführt wird. Nach dem Aufrufen der URL lädt der Webbrowser den Anmeldebildschirm. Siehe Abbildung 6.1.

6.3 Registrierung

Um das Netzwerk nutzen zu können, braucht jeder Nutzer einen Account. Das Formular zum Registrieren wird durch einen Klick auf “oder erst registrieren“ im Anmeldebildschirm geöffnet. Siehe Abbildung 6.2. In das Formular müssen für die Registrierung ein Pseudonym und ein Passwort eingegeben werden. Das Pseudonym muss mindestens 3 Zeichen lang sein, und darf nur Buchstaben, Ziffern und Unterstriche enthalten. Umlaute sind nicht erlaubt. Um bei der Passworteingabe die Wahrscheinlichkeit eines Tippfehlers zu minimieren, wird eine doppelte Passworteingabe verlangt. Das Passwort muss mindestens acht Stellen haben. Damit das Passwort möglichst sicher ist, sollten die folgenden Regeln eingehalten werden:

Abbildung 6.2: pribook.com: Registrierung

- Das Passwort sollte Ziffern, Kleinbuchstaben, Großbuchstaben und Sonderzeichen enthalten.
- Das Passwort sollte möglichst zufällig sein und keine Worte, Namen oder Begriffe enthalten.

Eine sichere Möglichkeit, sich Passwörter auszudenken und zu merken ist laut [44] die Methode, die Anfangsbuchstaben, Sonderzeichen und Zahlen eines Satzes zu nehmen. Dafür muss ein Satz erfunden werden, in dem sowohl Zahlen als auch Sonderzeichen vorkommen. Ein Beispiel dafür ist der Satz: "Beim Italiener um die Ecke bestelle ich am liebsten die Nummer 28, Pizza mit Champignons." Die Anfangsbuchstaben, Ziffern und Sonderzeichen dieses Satzes, der relativ einfach zu Merken ist, bilden das Passwort. Der vorliegende Satz würde also zu dem Passwort: "BIudEbialdN28,PmC."

Achtung: Pribook.com schützt deine Daten. Das heißt aber auch, dass Passwörter bei Verlust nicht zurückgesetzt werden können. Bei Verlust des Passwortes sind alle Daten unwiderruflich verloren und können nicht mehr entschlüsselt werden.

Nach einem Klick auf "Schlüssel generieren" werden die Passwörter überprüft und bei Übereinstimmung und ausreichender Länge wird in RSA-Schlüsselpaar generiert. Sollten die Passwörter nicht den Voraussetzungen entsprechen, erscheint eine Fehlermeldung und der Nutzer kann die Passwörter erneut eingeben. Das Erstellen der Schlüssel kann je nach System und Webbrowser einige Minuten dauern. Nachdem das Schlüsselpaar erstellt wurde, wird der private Schlüssel mit dem eingebenden Passwort verschlüsselt. Anschließend werden alle Schlüssel – der private, der öffentliche und der verschlüsselte private Schlüssel angezeigt. Siehe Abbildung 6.3.

Nun kann sich der Nutzer durch einen Klick auf “Registrieren“ im Netzwerk registrieren. Sollte das Pseudonym schon vergeben sein, oder nicht den Voraussetzungen entsprechen, so gibt es jetzt eine Meldung und der Nutzer kann das gewählte Pseudonym ändern und erneut auf “Registrieren“ klicken. Für die Registrierung werden das Pseudonym, der öffentliche Schlüssel, der verschlüsselte private Schlüssel, ein Hashwert des Passwortes und ein zufällig gewählter Salt an den Server geschickt. Das Passwort und der private Schlüssel bleiben nur dem Nutzer sichtbar. Hat die Registrierung geklappt, erscheint eine entsprechende Meldung.

6.4 Anmeldung

Zur Anmeldung im Netzwerk reicht es, das bei der Registrierung gewählte Pseudonym zusammen mit dem Passwort im Anmeldebildschirm einzugeben und auf “anmelden“ zu klicken. Sollten Pseudonym und Passwort korrekt sein, erscheint eine Meldung, die den Anwender mit “Hallo“ begrüßt und der Hauptbildschirm öffnet sich. Ist Passwort oder Pseudonym falsch, dann gibt es eine Fehlermeldung und der Anmeldebildschirm bleibt sichtbar. Siehe Abbildung 6.4 .

6.5 Nachrichten

Direkt nach dem Anmelden oder durch einen Klick auf “Nachrichten“ befindet sich der Nutzer auf der Nachrichtenseite. Im oberen Teil befindet sich das Formular zum Versenden und im unteren Teil werden die neusten empfangenen und verschickten Nachrichten angezeigt.

6.5.1 Nachrichten versenden

Zum Versenden einer Nachricht wird das Pseudonym des Empfängers in das Feld “Senden an:“ und die Nachricht in das Feld “Nachricht“ eingegeben und auf “abschicken (verschlüsselt)“ geklickt. Die Nachricht wird daraufhin automatisch verschlüsselt und an den Server verschickt. Dafür wird automatisch ein zufälliger AES-Schlüssel generiert, mit dem die Nachricht verschlüsselt wird. Für die Empfänger und den Absender holt sich die Webanwendung selbstständig die öffentlichen Schlüssel vom Server und generiert damit die Pris(Privacy-Padlocks), die zusammen mit der Nachricht an das Netzwerk gesendet werden.

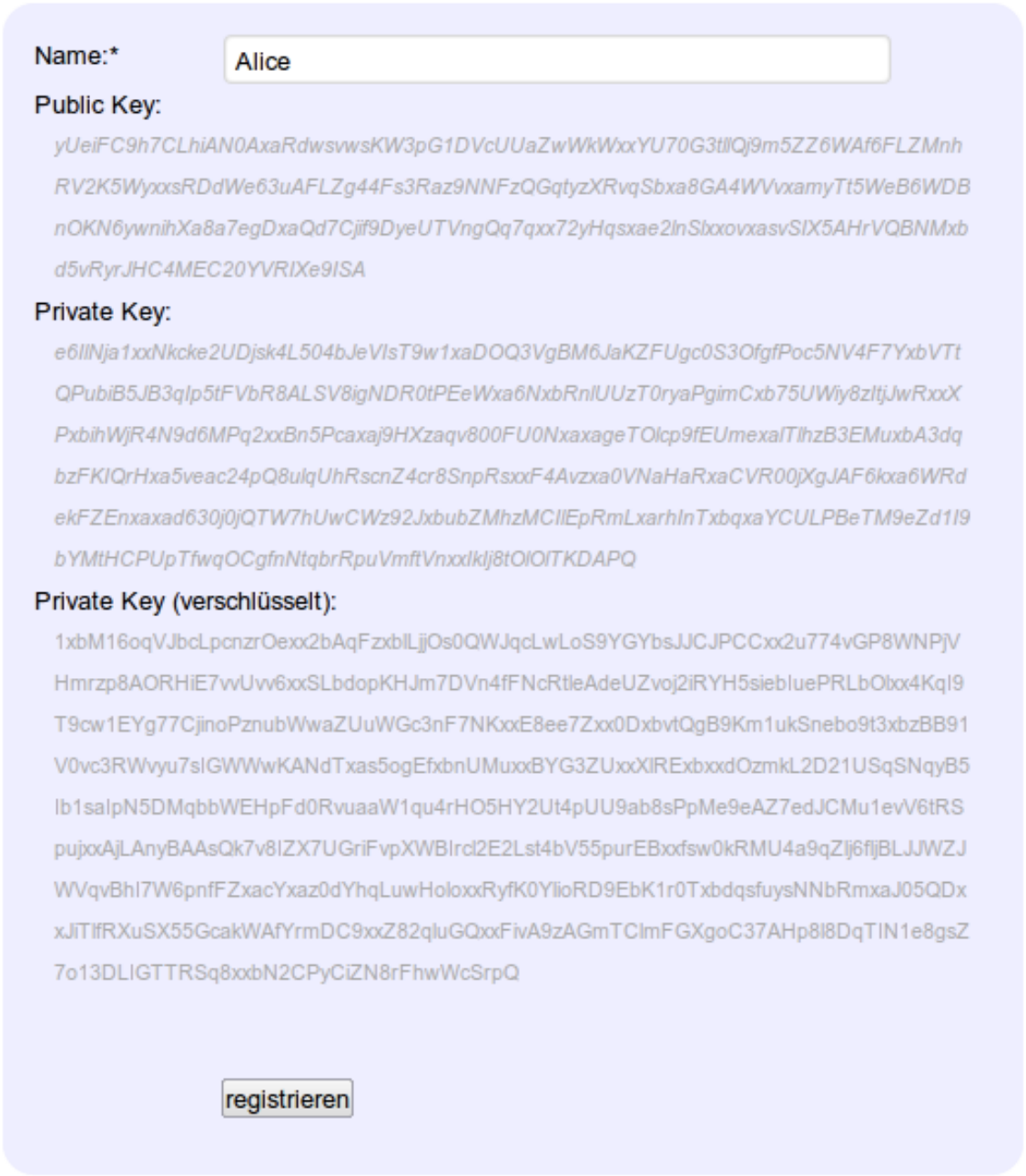


Abbildung 6.3: pribook.com: Erstellte Schlüssel

Abbildung 6.4: pribook.com: Hauptbildschirm

Wenn die Nachricht erfolgreich verschickt wurde, erscheint eine Meldung und sie wird bei den Nachrichten im unteren Teil der Seite angezeigt. Sollte es ein Problem geben, zum Beispiel, weil das Pseudonym des Empfängers falsch geschrieben ist, so erscheint eine Fehlermeldung.

Eine Nachricht kann auch an mehre Empfänger gleichzeitig gesendet werden. Dafür müssen die Pseudonyme durch ein Semikolon getrennt werden. Auch Gruppen und Circle können Empfänger einer Nachricht sein. Ein Gruppenname beginnt immer mit einer Raute "# " und ein Circle beginnt immer mit einem At "@ ". Eine korrekte Eingabe für die Empfänger, sähe also zum Beispiel folgender maßen aus: "Alice; Clara; #Gruppe1; @Friends; Bob" Die Nachricht würde in dem Fall dann an die Nutzer - Alice, Bob und Clara - sowie an den Circle "@Friends" und an die Gruppe "#Gruppe1" gesendet. Es ist auch möglich, Nachrichten Bilder anzuhängen. Dafür müssen vor dem Versenden im Feld "Anhang" Bilder vom lokalen Speicher ausgewählt werden. Diese werden auch automatisch mit demselben AES-Schlüssel wie die Nachricht verschlüsselt und können nach dem Absenden nur noch vom Absender und den Empfängern eingesehen werden.

6.5.2 Nachrichten empfangen

Die neuesten empfangenen Nachrichten werden im Hauptbildschirm angezeigt und automatisch entschlüsselt. Um nach neuen Nachrichten zu schauen, reicht ein Klick auf "Nachrichten" im oberen Menü. Zunächst werden lediglich die 10 neusten Nachrichten angezeigt. Mit einem Klick auf den Button "ältere Nachrichten anzeigen", der sich unterhalb der Nachrichten befindet, werden 10 weitere Nachrichten angezeigt. Dieser Vorgang lässt sich wiederholen, bis alle empfangenen und versendeten Nachrichten angezeigt werden. Im Startbildschirm werden die Nachrichten angezeigt, die der Nutzer selbst versendet hat, die er direkt empfangen hat oder die an eine Gruppe oder einen Circle gesendet wurden, deren Mitglied er ist.

6.5.3 Eine Nachricht

Eine Nachricht wird folgendermaßen dargestellt: Siehe Abbildung 6.5.

Links befindet sich der Name des Absenders gefolgt von der Nachricht. Oben links befindet sich ein Schlüsselsymbol, das anzeigt, dass die Nachricht verschlüsselt vom Server geholt - und für den Nutzer entschlüsselt wurde. Durch einen Klick auf dieses Symbol wird die verschlüsselte Nachricht angezeigt, so wie sie in der Datenbank gespeichert ist. Siehe Abbildung 6.6.

Abbildung 6.5: pribook.com: Eine Nachricht

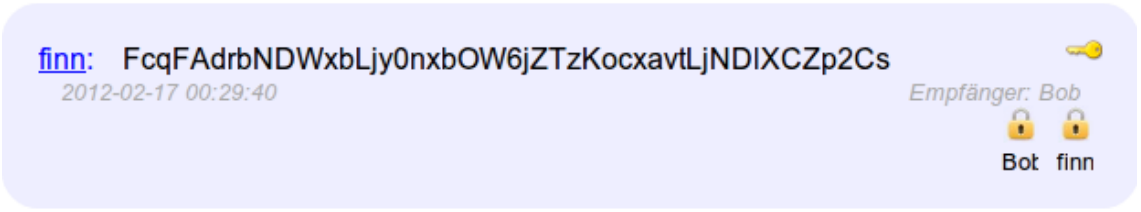


Abbildung 6.6: pribook.com: Eine verschlüsselte Nachricht

Unterhalb des Namens befindet sich links das Datum, an dem die Nachricht versendet wurde - oder genauer: an dem die Nachricht laut MEZ den Server erreicht hat. Alle Empfänger der Nachricht stehen rechts davon. Im unteren Teil der Nachricht werden die Privacy-Padlocks - oder kurz Pris - angezeigt. Nur Nutzern, für die ein solches Schloss an der Nachricht hängt, ist es möglich, mit ihrem Passwort die Nachricht zu lesen. Durch einen Klick auf die Schlosssymbole werden die Pris angezeigt, so wie sie in der Datenbank gespeichert sind.

6.5.4 Nachrichten löschen

Nutzer können alle Nachrichten löschen, die sie selbst verfasst haben. Sie werden dann vollständig aus der Datenbank entfernt und können nicht mehr wieder hergestellt werden. Auch die Nutzer, die die Nachricht bereits empfangen und gelesen haben, können danach nicht mehr auf sie zugreifen. Wenn ein Nutzer mit dem Mauszeiger über eine Nachricht geht, die er selbst verfasst hat, dann erscheint oben links ein kleines “x“. Ein Klick auf dieses “x“ reicht, um die Nachricht nach einer weiteren Sicherheitsabfrage zu löschen.

6.6 Suche

Durch einen Klick auf “Suchen“ im oberen Menü gelangt der Nutzer zum Suchmenü. Siehe Abbildung 6.7.

Abbildung 6.7: pribook.com: Suche

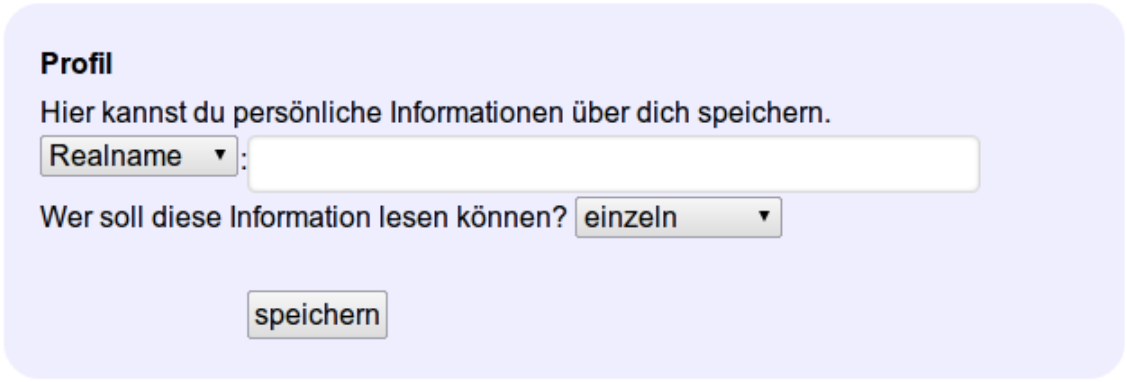


Abbildung 6.8: pribook.com: Profilmenü

Um andere User oder Gruppen zu finden, kann die Suche benutzt werden. Hier wird das Pseudonym des gesuchten Nutzers oder der Name der gesuchten Gruppe eingegeben und auf “suchen“ geklickt. Es werden sowohl die passenden Gruppen als auch die passenden Pseudonyme angezeigt. Es reicht, einen Teil des gesuchten Namens einzugeben. Es werden dann alle Pseudonyme und Gruppennamen angezeigt, die das eingegebene Wort enthalten. Klickt ein Anwender auf “suche“ ohne vorher etwas in das Suchfeld einzugeben, so werden die Pseudonyme aller angemeldeten Nutzer und die Namen aller Gruppen angezeigt.

6.7 Profildaten

Um Profildaten über sich einzugeben, muss ein Anwender im oberen Menü auf “Profil“ klicken. Es öffnet sich das Profilmenü. Siehe Abbildung 6.8.

6.7.1 Profildaten speichern

Zum Speichern einer neuen Profilinformation muss zunächst aus der linken Drop-down-Liste ein Attribut wie zum Beispiel “Hobbys“ ausgewählt werden. Die eigentliche Information – zum Beispiel “Schwimmen“ wird in das Textfeld daneben eingegeben. Zusätzlich muss für jede Profilinformation in der Drop-down-Liste unten rechts ausgewählt werden, wer diese Information lesen können soll. Zur Auswahl stehen “öffentlich“, die Circle und “einzeln“. Wird das Attribut “öffentlich“ gewählt, so wird die Information nicht verschlüsselt, sondern in Klartext an die Datenbank geschickt. Öffentliche Profilinformationen werden jedem Nutzer bei Ansicht des entsprechenden Profils angezeigt. Wird einer der Circle aus der Drop-down-Liste ausgewählt, so wird die Information für den Circle verschlüsselt und nur noch alle Mitglieder des Circles bekommen sie angezeigt und können sie entschlüsseln. Wird in der Drop-down-Liste “einzeln“ ausgewählt, so öffnet sich ein neues Textfeld. In dieses Textfeld können - genau wie bei normalen Nachrichten - einzeln die Empfänger eingegeben werden, die Information lesen können soll. Eine korrekte Eingabe sieht also genauso wie für normale Nachrichten aus. Die Eingabe: “Clara; @Friends; @Kollegen; @Verein“ wäre korrekt. Die Profilinformation wird dann für alle Empfänger verschlüsselt an den Server geschickt.

6.7.2 Profile

Der Profilbildschirm der verschiedenen Nutzer wird durch einen Klick auf deren Pseudonyme angezeigt. Siehe Abbildung 6.9.

Im oberen Teil des Profilbildschirms befindet sich der Freundschaftsindikator. Hier wird angezeigt, ob eine Freundschaft besteht und die Einstellungen der Freundschaft können verändert werden. Im mittleren Teil befinden sich die Profilinformationen des Nutzers sowie seine Freundschaften und seine Gruppen. Im unteren Teil werden Nachrichten angezeigt, die zwischen den beiden Nutzern versendet wurden.

6.7.3 Profilinformationen

Profilinformationen werden folgendermaßen dargestellt: Siehe Abbildung 6.10.

Links befindet sich das Attribut der Profilinformation Name gefolgt von der Information selbst. Wenn die Information nicht öffentlich gespeichert wurde, dann befindet sich oben links ein Schlüsselsymbol, das anzeigt, dass die Information verschlüsselt vom Server geholt - und für den Nutzer entschlüsselt wurde. Durch einen Klick auf dieses Symbol, wird die

Abbildung 6.9: pribook.com: Profil

Abbildung 6.10: pribook.com: Profilinformation

verschlüsselte Information angezeigt, so wie sie in der Datenbank gespeichert ist. Unterhalb des Attributs befindet sich links das Datum, an dem die Information erstellt wurde. Alle Empfänger, die die Information angezeigt bekommen und entschlüsseln können, stehen rechts davon. Im unteren Teil der Information werden die Privacy-Padlocks - oder kurz Pris - angezeigt. Nur Nutzern, für die ein solches Schloss an der Profilinformation hängt, ist es möglich, mit ihrem Passwort die Information zu lesen. Durch einen Klick auf die Schlösser Symbole werden die Pris angezeigt, so wie sie in der Datenbank gespeichert sind.

6.7.4 Profildaten löschen

Nutzer können ihre eigenen Profilinformationen jederzeit löschen. Die Profilinformationen werden dann vollständig aus der Datenbank entfernt und können nicht mehr wieder hergestellt werden. Auch die Nutzer, die die Profilinformation bereits eingesehen und gelesen haben, können danach nicht mehr auf sie zugreifen.
Wenn ein Nutzer mit dem Mauszeiger über eine seiner eigenen Profilinformationen geht, dann erscheint oben links ein kleines “x“. Ein Klick auf dieses “x“ reicht, um die Information nach einer weiteren Sicherheitsabfrage zu löschen.

6.8 Freundschaften

Nach dem Einloggen wird stets links im Bild die Freundschaftsliste angezeigt. Siehe Abbildung 6.4. Freundschaften sind Verknüpfungen zwischen den Nutzern, die beidseitig bestätigt werden müssen. Wurde eine Freundschaft von beiden Seiten bestätigt, so ist dies eine Freundschaft und wird im oberen Teil der Freundschaftsliste angezeigt. Eine Freundschaftsanfrage, die noch nicht bestätigt wurde, wird darunter unter “Anfragen von dir“ angezeigt. Freundschaftsanfragen an den Nutzer selbst, die er noch nicht bestätigt hat, werden darunter unter “Anfragen an dich“ angezeigt. Im Profilbildschirm eines Nutzers, werden alle Freundschaften, die beidseitig geschlossen wurden und nicht versteckt sind angezeigt.

6.8.1 Freundschaftsanfragen stellen

Um eine Freundschaftsanfrage zu stellen, muss auf das Pseudonym des entsprechenden Nutzers geklickt werden, damit sich der Profilbildschirm des Nutzers öffnet. Im oberen Bereich des Profilbildschirms befindet sich der Freundschaftsindikator. Hier muss auf “Freundschaft anbieten“ geklickt werden, um eine Freundschaft anzubieten.

6.8.2 Freundschaften kündigen

Eine Freundschaftsanfrage wird zurückgezogen, in dem im Freundschaftsindikator im Profilbildschirm auf “Freundschaft zurückziehen“ oder “zurückziehen“ geklickt wird.

6.8.3 Freundschaftsanfrage bestätigen

Wurde eine Freundschaftsanfrage an einen User gestellt, so kann dieser die Freundschaftsanfrage bestätigen, in dem er auf “bestätigen“ im Freundschaftsindikator klickt.

6.8.4 Freundschaftsanfragen verstecken

Freundschaftsanfragen können versteckt werden. Wenn eine Freundschaftsanfrage versteckt ist, so wird sie nur noch den beiden befreundeten Usern angezeigt. Es reicht, wenn die Freundschaftsanfrage von einem der beiden User versteckt wird.

6.9 Gruppen

Gruppen dienen dazu, dass Nutzer sich miteinander vernetzen können. Gruppen teilen sich einen Schlüssel, so das alle Gruppenmitglieder Nachrichten an die ganze Gruppe senden und empfangen können. Es können auch Gruppeninformationen gespeichert werden, die entweder öffentlich oder nur für die Gruppe sichtbar sind. Alle Mitglieder einer Gruppe können neue Leute in eine Gruppe einladen. Eingeloggte Nutzer bekommen stets unten links die Gruppen angezeigt, in denen sie Mitglied ist oder für die sie eine Einladung bekommen haben. Siehe Abbildung 6.4. Durch einen Klick auf “Gruppen“ im oberen Menü öffnet sich das Gruppenmenü Siehe Abbildung 6.11.

6.9.1 Gruppen gründen

Eine neue Gruppe kann gegründet werden, in dem unter “Neue Gruppe erstellen“ ein Gruppenname in die Textbox eingegeben und auf “gründen“ geklickt wird. Ist der Gruppenname noch nicht vergeben und entspricht der Voraussetzungen (Nur Ziffern, Unterstriche und Buchstaben, keine Umlaute) so wird die Gruppe jetzt erstellt und es gibt eine entsprechende Meldung. Andernfalls erscheint eine Fehlermeldung. Bei der Gründung einer

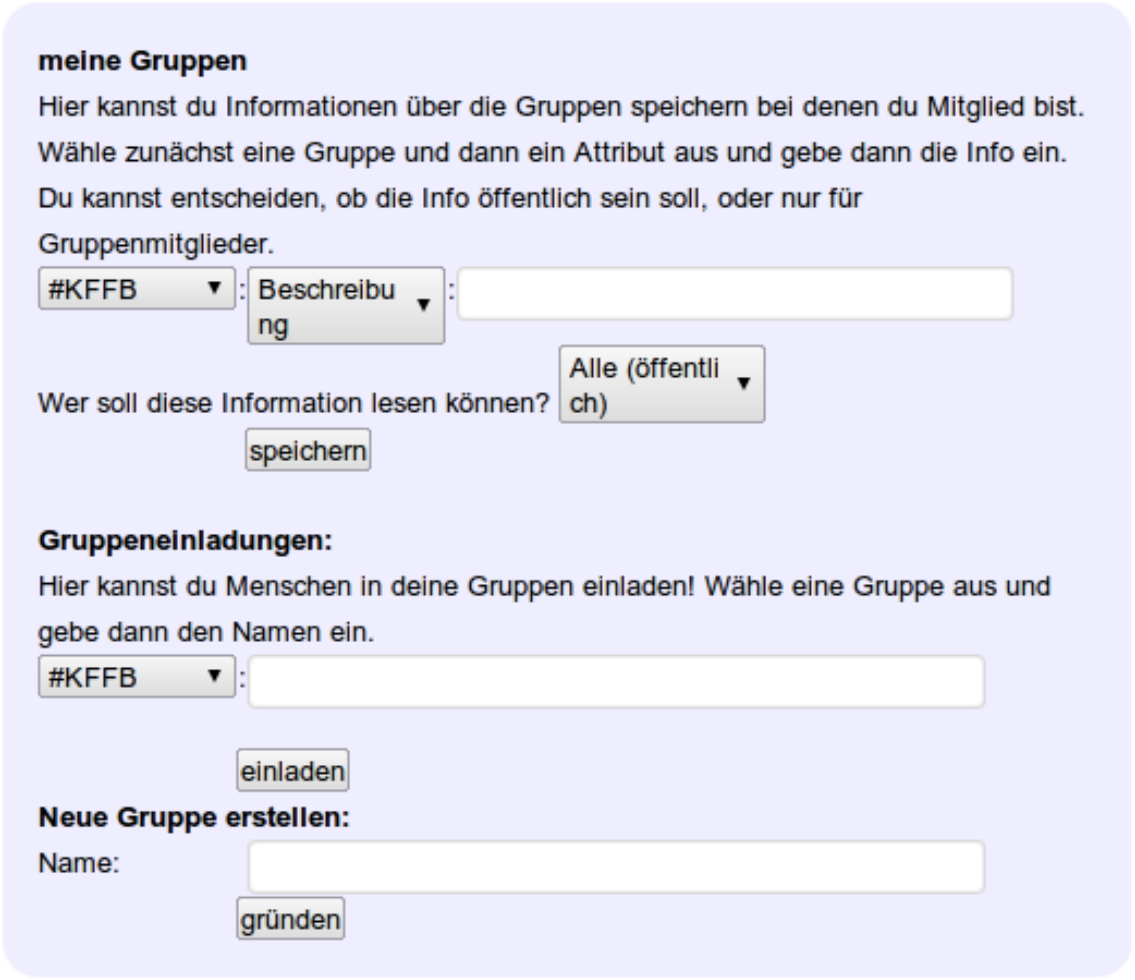


Abbildung 6.11: pribook.com: Gruppenmenü

Gruppe wird automatisch ein Schlüssel generiert, der für jedes Mitglied verschlüsselt in der Datenbank gespeichert wird.

6.9.2 Nutzer in Gruppen einladen

Im Gruppenmenü unter "Gruppeneinladungen" können andere Nutzer in die eigenen Gruppen eingeladen werden. Dazu wird aus der Drop-down-Liste die entsprechende Gruppe ausgewählt, in die Textbox das Pseudonym des Nutzers getippt, der eingeladen werden soll und dann auf "einladen" geklickt. Ein Nutzer, der in die Gruppe eingeladen wird, bekommt den Gruppenschlüssel und kann nun auf alle Nachrichten und Informationen zugreifen, die an die Gruppe adressiert sind. Nutzer können nur freiwillig wieder aus einer Gruppe austreten. Aber auch dann kann es sein, dass sie den Schlüssel gespeichert haben und damit die Nachrichten der Gruppe entschlüsseln können.

6.9.3 Gruppeninformationen speichern

Jedes Mitglied kann für seine Gruppen Informationen speichern. Diese können öffentlich sein, sodass sie nicht verschlüsselt werden und von allen Nutzern gesehen werden können. Dies kann zum Beispiel sinnvoll sein, um neue Leute für die Gruppe zu interessieren. Informationen können aber auch intern sein, dann werden sie mit dem Schlüssel der Gruppe verschlüsselt und nur noch die Gruppenmitglieder können darauf zugreifen. Gruppenmitglieder speichern neue Gruppeninformationen folgendermaßen. Zunächst wird im Gruppenmenü unter meine Gruppen in der ersten Drop-down-Liste die entsprechende Gruppe ausgewählt. In der zweiten Drop-down-Liste wird ein Attribut für die Information ausgewählt zum Beispiel "Beschreibung". Dann wird die Information in das Textfeld eingegeben, in der unteren Drop-down-Liste ausgewählt, ob die Information öffentlich oder intern sein soll und dann auf "speichern" geklickt.

6.9.4 Gruppeninformationen ansehen

Durch einen Klick auf einen Gruppennamen öffnet sich der Gruppenbildschirm. Siehe Abbildung 6.12.

Oben befindet sich der Mitgliedschaftsindikator, darunter die Gruppeninformationen, die für den eingeloggten Nutzer sichtbar sind. Darauf folgen die Mitglieder der Gruppe. Ist der eingeloggte Nutzer auch Mitglieder der Gruppe, so werden unten die Nachrichten angezeigt, die an die Gruppe gesendet wurden.

Abbildung 6.12: pribook.com: Gruppenprofil

6.9.5 Mitgliedschaft ein einer Gruppen beantragen

Um die Mitgliedschaft in einer Gruppe zu beantragen oder die Einladung in eine Gruppe zu bestätigen, muss der Gruppenbildschirm der entsprechenden Gruppe geöffnet und im Mitgliedsindikator auf “Mitgliedschaft beantragen“ oder “bestätigen“ geklickt werden.

6.9.6 Mitgliedschaft ein einer Gruppen löschen

Um die Mitgliedschaft in einer Gruppe zu löschen, muss der Gruppenbildschirm der entsprechenden Gruppe geöffnet und im Mitgliedsindikator auf “Mitgliedschaft kündigen“ geklickt werden. Mitgliedschaft verstecken. Mitgliedschaften in Gruppen können versteckt werden. Ist eine Mitgliedschaft in der Gruppe versteckt, so wird sie nur noch dem eingeloggten Nutzer und den Mitgliedern der Gruppe angezeigt. Zum Verstecken einer Gruppenmitgliedschaft muss der Gruppenbildschirm der entsprechenden Gruppe geöffnet und im Mitgliedsindikator unter “Mitgliedschaft verstecken?“ “Ja“ ausgewählt werden.

6.9.7 Nachrichten an Gruppen schicken

Gruppenmitglieder können Nachrichten an die gesamte Gruppe schicken in dem sie als Empfänger den Gruppennamen (Inklusive der “#“) eingeben.

6.10 Circles

Ein Nutzer kann seine Kontakte in verschiedene Circles aufteilen um dann verschiedene Informationen mit bestimmten Circles zuteilen. So kann der Nutzer zum Beispiel je einen Circle für seine Familie, seine Arbeitskollegen und seinen Sportverein erstellen und mit ihnen unterschiedliche Informationen teilen. Circles teilen sich einen Schlüssel und alle Nachrichten, die an einen Circle versendet werden, werden von allen Mitgliedern des Circles empfangen und können von diesen entschlüsselt werden. Die Circles eines Nutzers werden nach dem Einloggen stets rechts im Bild angezeigt. Siehe Abbildung 6.4 Das Circlemenü erscheint durch einen Klick auf “Circles“ im oberen Menü. Siehe Abbildung 6.13.

Abbildung 6.13: pribook.com: Circlemenü

6.10.1 Circle erstellen

Ein neuer Circle wird erstellt, in dem im Circlemenü unter "Neuen Circle erstellen" ein Name für einen Circle eingegeben und auf "erstellen" geklickt wird. Der Name eines Circles darf noch nicht für einen anderen Circle verwendet werden und nur Buchstaben, Zahlen, Unterstriche und keine Umlaute enthalten. Entspricht der Name den Voraussetzungen, erscheint eine Meldung und der Circle wird im rechten Bildschirmbereich angezeigt.

6.10.2 Nutzer zu einem Circle hinzufügen

Im Circlemenü wird unter "Person zu Circle hinzufügen" in der Drop-down-Liste der entsprechende Circle ausgewählt und dann das Pseudonym des Nutzers in die Textbox eingetragen und auf "hinzufügen" geklickt. Wenn das Hinzufügen erfolgreich war, erscheint das Pseudonym des hinzugefügten Nutzer im entsprechenden Circle auf der rechten Seite.

6.10.3 Einen Circle löschen

Um einen Circle zu löschen, muss man mit dem Mauszeiger über den Circle gehen, dann erscheint oben links im Circle ein kleines "x". Durch einen Klick auf dieses x wird der Circle nach einer Sicherheitsabfrage gelöscht.

Kapitel 7

Evaluierung

In diesem Kapitel wird die Evaluierung der Referenzimplementierung Pribook.com beschrieben. Pribook wurde durch eine Befragung evaluiert. Der Aufbau der Befragung wird in Abschnitt 7.1 geschildert, die Ergebnisse der Befragung werden in Abschnitt 7.3 geschildert und in Abschnitt 7.3 werden die Ergebnisse interpretiert.

7.1 Aufbau der Befragung

Da Pribook.com ein Produkt für normale Endverbraucher ist, ist es sinnvoll dieses auch von normalen Endverbrauchern testen zulassen. Daher wurde zur Evaluierung eine Befragung, die einen Test der prototypischen Umsetzung beinhaltet, ausgewählt.

Ziel der Befragung ist es, das Produkt auf die Kriterien zu testen, die für ein sicheres soziales Netzwerk in Kapitel 4 aufgestellt wurden. Wie in Kapitel 5 und 6 nachzulesen ist, und wie ein Blick auf pribook.com zeigt, werden die drei Invarianten, die in Kapitel 4 für ein sicheres soziales Netzwerk aufgestellt werden, realisiert. Die Nachrichten (und Profildaten) werden von Nutzer zu Nutzer verschlüsselt, es wird asymmetrische Verschlüsselung verwendet, sodass kein geheimer Schlüsselaustausch stattfinden muss, und die Bedienung funktioniert komplett im Webbrowser. Der Test mit potenziellen Endverbrauchern soll nun vor allem belegen, ob die Nutzung dadurch wirklich leichter wird und die folgenden Fragen beantworten:

- Schaffen es normale Endverbraucher sich bei pribook.com anzumelden und Nachrichten zu verschicken?

- Schaffen es mehr normale Endverbraucher sich bei pribook.com anzumelden und Nachrichten zu verschicken als PGP oder s/mime zu installieren und verschlüsselte Emails zu verschicken?
- Schaffen es Endverbraucher, die bisher nicht ausreichend Zeit oder Lust hatten sich mit einer sicheren Kommunikation auseinanderzusetzen, sich bei pribook.com anzumelden und Nachrichten zu verschicken?
- Schaffen es Endverbraucher, die bisher vermeiden brisante Inhalte über das Internet zu verschicken, sich bei pribook.com anzumelden und Nachrichten zu verschicken?
- Ist den Nutzern intuitiv klar, welche Daten wie geschützt sind?

Außerdem soll die Befragung die Thesen belgen, die zur Entwicklung eines sicheren sozialen Netzwerkes motivieren.

- Viele Menschen nutzen die sozialen Netzwerke regelmäßig.
- Viele Menschen kritisieren den Datenmissbrauch der etablierten sozialen Netzwerke.
- Viele Menschen wünschen sich eine sichere Alternative zu den sozialen Netzwerken.

Um diese Fragen zu beantworten beziehungsweise diese Thesen zu überprüfen, wurde als Form der Befragung ein Internetfragebogen ausgewählt. Diese Form der Befragung ermöglicht es, dass ohne großen Aufwand eine größere Anzahl an Personen an dem Test mitmachen kann. Außerdem bietet sich ein Onlinefragebogen an, da für die Befragung ein Test der Internetseite pribook.com nötig ist. Auch wenn eine webbasierte Befragung potenziell eine höhere Fehleranfälligkeit aufweist, als zum Beispiel eine persönliche Befragung, so überwogen aus Sicht des Autors dieser Diplomarbeit die Pro-Argumente, so, dass die Entscheidung zugunsten einer webbasierten Befragung ausfiel.

Der Fragebogen besteht aus zwei Teilen. Der erste Teil ist ein allgemeiner Teil, der noch vor dem Test abgefragt wird und den Nutzer nach seinem Surfverhalten sowie seiner Einstellung zu Datenschutz befragt. Dieser Teil dient auch dazu, dass der Fragebogen einen einfachen Einstieg hat und die Probanden nicht davor abschrecken ihn auszufüllen. Erst nachdem alle Fragen beantwortet sind, beziehungsweise „keine Angabe“ ausgewählt wurde, kann ein Proband auf „weiter“ klicken und bekommt den zweiten Teil des Tests angezeigt. Der zweite Teil besteht aus einer Erklärung, wie pribook.com zu testen ist und aus Fragen, die sich auf den Test beziehen. Für die Beantwortung der Fragen des zweiten Teils, sollten die Probanden die folgenden Test durchführen: 1. Melde dich bei Pribook an und schicke eine Nachricht an „Alice“. 2. Suche nach Alice, klick auf ihren Namen und biete ihr

eine Freundschaft an. 3. Biete auch Clara und Bob eine Freundschaft an. 4. Verstecke die Freundschaftsanfrage zwischen Bob und dir.

Die zwei Teile des Fragebogens bestehen jeweils aus 12 Fragen. Der erste Teil ist in drei Frageblöcke mit je vier Fragen unterteilt. Der erste Block enthält Fragen zur Nutzung von sozialen Netzwerken und Datenschutz in sozialen Netzwerken. Die Fragen sind jeweils als Aussagen formuliert und die Probanden können angeben, inwieweit sie den Aussagen zustimmten. Diese Skalierungsmethode ist eine nichtkomparative diskrete Ratingskala, die als Likertskala bezeichnet wird und sich vor allem eignet, da sie leicht zu verstehen und auch leicht zu erstellen ist. Der zweite Block fragt die Probanden nach ihrer bisherigen Nutzung von PGP und s/mime. Hierbei wird keine Likertskala verwendet sondern, die Nutzer können die Aussagen nur mit „ja“ und „nein“ beantworten. Da nicht vorausgesetzt werden kann, dass die Probanden die Begriffe PGP und s/mime kennen, und so fälschlicherweise etwas Falsches ankreuzen könnten, wird zusätzlich die Option „nicht sicher“ angeboten. Der letzt Block des ersten Teils enthält noch einmal 4 Fragen zum Thema Datenschutz im Allgemeinen und hat wieder eine Likertskala.

Der zweite Teil des Fragebogens beginnt mit der Erläuterung des Tests der Referenzimplementierung pribook.com und enthält dann 4 Fragen, ob die Durchführung der vier Aufgaben erfolgreich verlaufen ist. Hier haben die Probanden wieder die Möglichkeit „ja“, „nein“ oder „unsicher“ auszuwählen. Des Weiteren enthält der Block Auswahlmöglichkeiten für den Fall, dass einer der Test nicht funktioniert hat. Sollte einer der Test nicht funktioniert haben, so können die Probanden hier einen Grund dafür angeben. Es sind 5 Gründe vorgegeben, die die Probanden durch ein Häckchen setzen können. Außerdem gibt es eine Textbox, in die weitere Gründe geschrieben werden können. Der zweite Block des zweiten Teils enthält 6 Fragen, die zeigen sollen, ob die Nutzer intuitiv verstehen, welche Informationen wie geschützt werden. Die Fragen sind wieder als Aussagen formuliert und die Probanden können sich wieder zwischen „ja“, „nein“ und „unsicher“ entscheiden, je nachdem, ob sie die Aussagen für korrekt halten.

Der komplette Fragebogen ist in Anhang A abgebildet.

Die Befragung wurde mit einem kurzen Text beworben, der über Emailverteiler und in einem sozialen Netzwerk verbreitet wurde, und einen Link zu dem Onlinefragebogen enthielt. Der Onlinefragebogen war vom 23. Februar bis zum 20. März, also insgesamt 27 Tage erreichbar. Die Teilnehmenden hatten neben einer kurzen Beschreibung von pribook.com im Fragebogen auch die Möglichkeit auf das Handbuch zuzugreifen, so wie es in Kapitel 6 veröffentlicht ist.

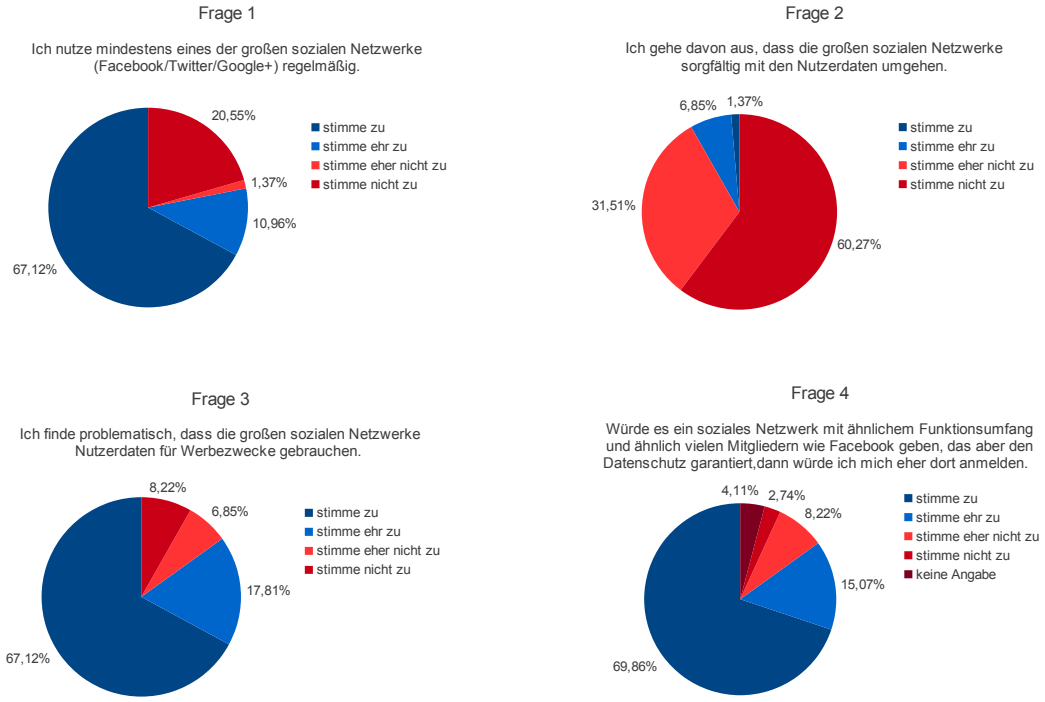


Abbildung 7.1: Beantwortung der Fragen Block1

7.2 Ergebnisse der Befragung

Bei dem Test haben insgesamt 73 Probanden mitgemacht, was nach Meinung des Autors eine ausreichende Anzahl für einen ersten Eindruck ist. Die Ergebnisse der Befragung sind in AnhangB zu finden.

Der ersten Aussage „Ich nutze mindestens eines der großen sozialen Netzwerke (Facebook/Twitter/Google+) regelmäßig.“ stimmen 49 Teilnehmer zu und weitere 8 Probanden stimmen dieser Aussage „eher zu“. Zusammen stimmen der Aussage also 57 Probanden oder 78% zu oder eher zu. Weniger als 9%, nämlich 6 Probanden stimmen der Aussage zu oder eher zu, die Betreiber der sozialen Netzwerke gingen sorgfältig mit den Nutzerdaten um. Mehr als 84% der Teilnehmenden findet problematisch, dass die sozialen Netzwerke Nutzerdaten für Werbezwecke gebrauchen. Ebenfalls etwas mehr als 84% geben an, eher ein soziales Netzwerk benutzen zu wollen, in dem der Datenschutz garantiert wird. Siehe Abbildung 7.1.

Im zweiten Block ging es um die Nutzung von Emailverschlüsselung. Nur 30,1% geben an in der letzen Woche in der Lage gewesen zu sein, verschlüsselte Emails nach PGP-Standard zuempfangen. Wohingegen nur 23% angeben in der letzten Woche auch in der

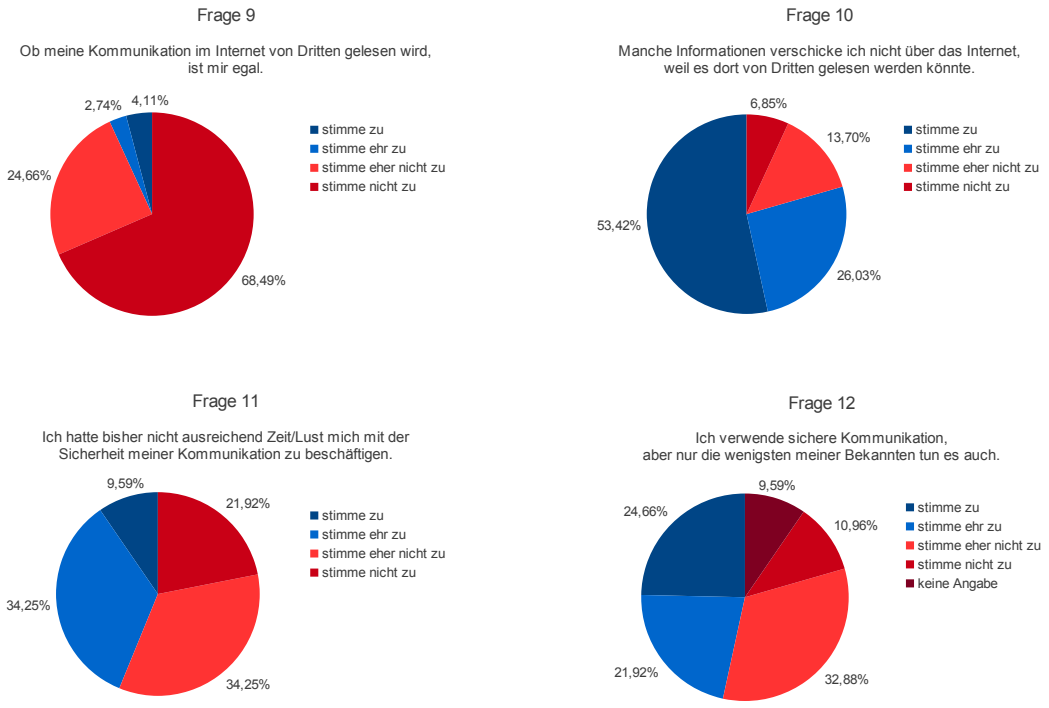


Abbildung 7.2: Beantwortung der Fragen Block3

Lage gewesen zu sein auch Emails nach PGP-Standard zuversenden. Emails nach S/MIME-Standard konnten laut Angaben 23% der Probanden empfangen und 24% der Probanden versenden.

Im dritten Block wurde die Einstellung zu Datenschutz im Allgemeinen abgefragt. Nur 6,8% der Befragten geben an, dass es Ihnen egal sei (Stimme zu oder stimme eher zu), ob ihre Daten im Internet von dritten gelesen werde. Insgesamt 68 der Probanden oder 93,1% gaben hierzu an, dieser Aussage nicht zu zustimmen oder eher nicht zuzustimmen. Immerhin 79,5% stimmen der Aussage zu (oder eher zu), dass sie es vermeiden brisante Daten über das Internet zu verschicken. Die Aussage, „Ich hatte bisher nicht genügend Zeit/Lust mich mit der Sicherheit meiner Kommunikation zu beschäftigen.“ teilen nur 43% der Befragten. Ebenfalls 43% der Befragten geben an, dass sie sichere Kommunikation verwenden, dies aber nur wenige ihrer Bekannten auch tun. Siehe Abbildung 7.2.

Der Test von Pribook.com folgt im zweiten Teil der Befragung und der erste Block des zweiten Teils fragt ab, ob die Benutzung erfolgreich verlaufen ist. Für den ersten Test sollten sich die Probanden bei Pribook.com anmelden und eine Nachricht (Die automatisch verschlüsselt wird) an „Alice“ verschicken. 60 Probanden oder 82,2% geben an, diesen Test erfolgreich durchgeführt zu haben. 3 Probanden oder 4,1% Prozent geben an, den Test

nicht erfolgreich durchgeführt zu haben wobei 2 Probanden (2,7%) angeben, sich nicht sicher zu sein, ob der Test erfolgreich war. 8 Probanden (11%) machen keine Angaben. Den zweiten Test, eine Freundschaft mit Alice zu schließen, werten 61 Teilnehmende (83,6%) als erfolgreich. Genauso viele wie den dritten Test, eine Freundschaft mit Bob und Clara zu schließen, als erfolgreich werten. Den 4. Test hingegen werten nur 56 der Befragten als erfolgreich durchgeführt, was 76,7% entspricht. In diesem Test sollte die Freundschaftsanfrage mit Bob versteckt werden.

Von den Befragten, die nicht angeben alle Tests erfolgreich durchgeführt zu haben, geben 5 oder 6,8% an „Keine Lust mehr an der weiteren Test Durchführung gehabt zu haben." Einer der Probanden gibt an, dass die Erstellung des Schlüssels zu lange gedauert hat. Ebenfalls einer der Teilnehmenden, der nicht alle Test erfolgreich durchgeführt hat gibt an, dass es eine Fehlermeldung gab und die Seite nicht wie im Handbuch beschrieben reagiert habe. 5 (6,8%) kreuzten an, die Bedienung sie nicht intuitiv und werde auch durch das Handbuch nicht erklärt.

Der 5. Block des Fragebogens besteht aus 4 Aussagen, die sich darauf beziehen ob und wie die Daten bei pribook.com gesichert werden. Die Befragten mussten Angeben, ob die Aussagen richtig oder falsch sind, um so zu zeigen, ob ihnen intuitiv klar ist, wie pribook funktioniert. Die erste Aussage, der Netzwerkbetreiber könne sich die Nachricht ansehen, die die Befragten an Alice verschickt haben, erkennen 40 der Befragten richtigerweise als falsch an (54,8%). Nur 8 Probanden oder 11% halten die Aussage für richtig. 25 Probanden waren sich unsicher oder machten keine Angaben. Bei der zweiten Aussage, der Netzwerkbetreiber könne sehen, dass eine Nachricht vom Account der Probanden an Alice versendet wurde, liegen 45,2% der Probanden mit „ja" richtig, 11% liegen falsch und 32 machen keine Angaben oder sind sich nicht sicher. „Der Netzwerkbetreiber kann sehen, dass ich mit Alice befreundet bin." bewerten 58,9% richtig, 8,3% falsch und 32% gar nicht. Lediglich 31,5 % liegen bei der Bewertung der Aussage richtig, der Netzbetreiber können die (versteckte) Freundschaft mit Bob sehen. Hier liegen 43,2% falsch und 24% machen keine Angaben oder sind sich nicht sicher. 50 Probanden geben richtigerweise an, dass ihre Freundschaft mir Alice für alle Nutzer öffentlich ist (8 falsch, 15 ohne Angabe) und sogar 51 wissen richtigerweise, dass die versteckte Freundschaft mit Bob nicht allen anderen Nutzern öffentlich ist (8 falsch, 14 ohne Angabe).

Des Weiteren enthielt der Fragebogen noch ein freies Textfeld, in dem die Probanden angeben sollten: „Sonstige Anmerkungen: Was gefällt dir gut, was gefällt die schlecht? Sind dir Fehler aufgefallen? Was würdest du gerne für Features bei pribook haben?". Insgesamt nutzten 28 der Befragten die Möglichkeit ein Kommentar zu hinterlassen. Alle Kommentare befinden sich im Anhang **??**. Mehrmals wurde positiv der bessere Datenschutz gegenüber den herkömmlichen sozialen Netzwerken gelobt. So schreibt ein Nutzer: „Es macht den

Anschein, dass es sicherer ist als z.B. Facebook, das gefällt mir gut.“ Mehrere Befragte geben an, sich ein besseres Userinterface zu wünschen. Ein Kommentar hierzu lautet zum Beispiel: „Natürlich unansehnlich im Vergleich zu fb...“. Außerdem wird in mehreren Kommentaren die Funktion zum Verstecken von Freundschaften kritisiert: „Beim anklicken der Ja/Nein Anzeige für das verstekcen der Freunschadftsanzeige fehlte mir noch ein Button für Änderungen speichern o.ä. hab ich erstmal länger gescuht, da mir nicht klar war, dass durch bloßes anklicken des Buttons die Einstellung quasi gespeichert ist :)“[sic]. Dies deckt sich auch mit der Auswertung der Fragen. Öfter wird auch angemerkt, dass ein tolles Feature das Versenden von Fotos wäre („Features: Fotos“), obwohl dies schon während des Tests möglich war.

7.3 Interpretation der Befragung

Die Thesen, die in der Einleitung aufgestellt wurden, um die Entwicklung eines sicheren sozialen Netzwerkes zu motivieren, konnten durch die Befragung bestätigt werden. Soziale Netzwerke sind eine viel gebrauchte IT-Anwendung. Diese These wird durch die beantwortung der ersten Frage bestätigt, wo 78% der Aussage zustimmen oder eher zustimmen, sie sein regelmäßig in den großen sozialen Netzwerken aktiv. Viele Menschen kritisieren den Datenmissbrauch der etablierten sozialen Netzwerke. Dies wird durch die Tatsache bestätigt, dass der Aussage, „ich gehe davon aus, dass die großen sozialen Netzwerke sorgfältig mit den Nutzerdaten umgehen“ nur 9 % zustimmen oder eher zustimmen. Außerdem geben 84% an es problematisch zu finden, dass die großen sozialen Netzwerke Nutzerdaten für Werbezwecke gebrauchen. 84% wünschen sich daher eine sichere Alternative zu den sozialen Netzwerken.

Die Befragung sollte des weiteren zeigen, ob es einfacher ist, sich bei Pribook.com anzumelden, um verschlüsselte Nachrichten zu verschicken und zu empfangen, oder ob die einfachste Möglichkeit nach wie vor ist, PGP oder S/MIME zu installieren. Dazu ist es notwendig zunächst festzustellen, wie viele der Probanden überhaupt in der Lage sind verschlüsselte Emails zu verschicken und zu empfangen. Das Ergebnis ist, dass 22 (30,1%) der Probanden angeben, dass sie verschlüsselte Emails mit PGP oder S/MIME verschicken und empfangen können. Da aber 60 (82,2%) Probanden den Test erfolgreich absolviert haben, sich bei pribook.com anzumelden und eine verschlüsselte Nachricht zu verschicken, ist dies offensichtlich deutlich einfacher für normal Endverbraucher. 7.3

Nun könnte eingewendet werden, dass die 51 Probanden, die bisher keine Emailverschlüsselung verwendet haben, dies aus anderen Gründen vermieden haben und nicht nur, weil es ihnen, zu aufwendig ist (siehe Kapitel 1). Allerdings geben von diesen 51 Probanden

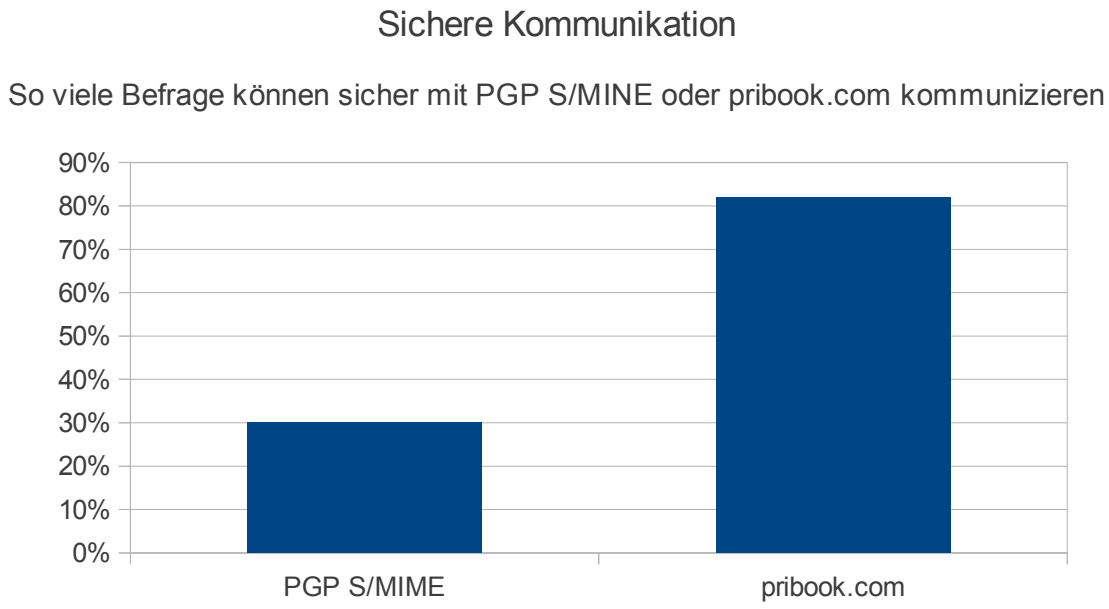


Abbildung 7.3: Balkendiagramm

nur 3 an, dass sie der Aussage zustimmen oder eher zustimmen, es sei ihnen egal, ob ihre Kommunikation im Internet von Dritten gelesen wird. Immerhin 38 der 51 geben an, manche Informationen nicht über das Internet zu verschicken, weil es dort von Dritten gelesen werden könnte. Alle der 51 Probanden (100%) geben an, dass sie entweder darauf verzichten wichtige Informationen über das Internet zu versenden oder ihnen zumindest die Sicherheit der Kommunikation nicht egal ist. Somit ist klar, dass alle Probanden, die bisher keine sichere Kommunikation benutzen trotzdem ein Interesse an sicherer Kommunikation haben. Offenbar ist die Einschätzung von Phillip Zimmermann, „ein wichtiger Faktor ist mit Sicherheit auch, wie einfach man es benutzen kann“ richtig. Die Ergebnisse der Umfrage geben ein erstes Indiz dafür, dass pribook.com schon in der prototypischen Umsetzung der einfachste Weg ist, verschlüsselte Nachrichten zu verschicken und zu empfangen.

Die Ergebnisse lassen auch vermuten, dass die erfolgreiche Durchführung des ersten Tests nicht von den Vorkenntnissen der Probanden abhängt. Verantwortlich dafür, dass 13 Probanden nicht angegeben haben den Test erfolgreich durchgeführt zu haben, könnten auch Bug's sein, die in der prototypischen Version noch aufgetreten sind. Ein Indiz hierfür ist die Tatsache, dass sich die Quote derer, die den Test erfolgreich bestanden haben, obwohl sie zustimmen oder eher zustimmen „bisher nicht ausreichend Zeit/Lust gehabt zu haben, um sich mit der Sicherheit ihrer Kommunikation zu beschäftigen“ sich nicht deutlich von

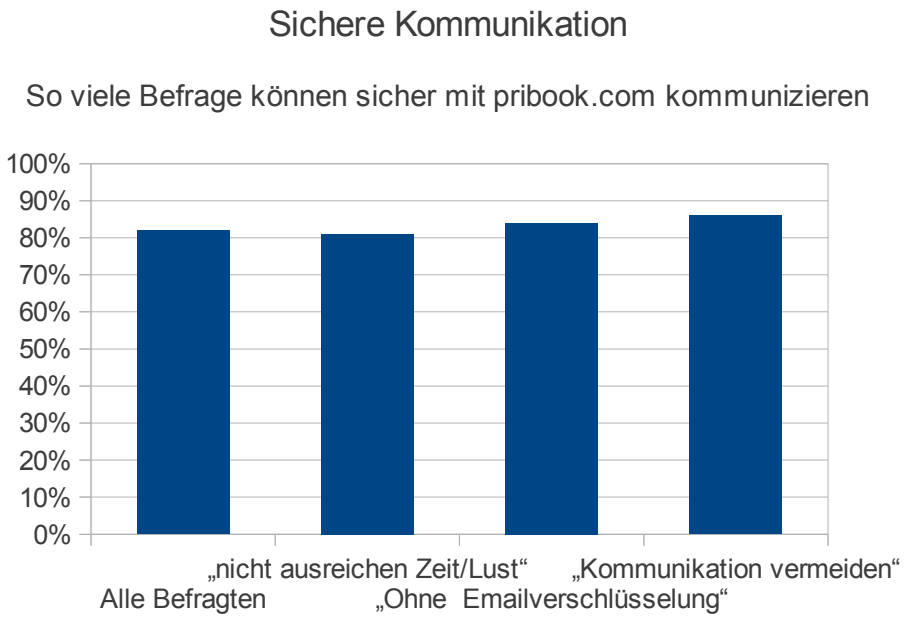


Abbildung 7.4: Balkendiagramm

der Quote alle Befragten unterscheidet (26 von 32 = 81%). Ähnlich sieht es bei den Befragten aus, die bisher keine Emailverschlüsselung verwenden. Von ihnen konnten 43 der 51 (84,3%) erfolgreich den 1. Test durchführen. Auch von den Menschen, die angeben es bisher zu vermeiden brisante Informationen über das Internet zu versenden schaffen den 1. Test 50 von 58 (86%). Siehe Abbildung 7.4.

Schließlich sollte die Umfrage Informationen darüber geben, ob den Nutzern intuitiv klar ist, ob und wie ihre Daten geschützt sind. Die Beantwortung der ersten beiden Fragen, des 5. Blockes zeigt, dass zwar nur 54,7% oder 45,2% der Befragten richtig einschätzen wie die Daten genau verschlüsselt werden, aber auch nur je 11% bei der Beantwortung der Fragen falsch liegen. Hier müsste die Quote noch deutlich verbessert werden. Eine Möglichkeit könnte es sein, dass Userinterface umzugestalten, und Grafiken zu Erläuterung einzufügen. Bei der Freundschaftsbeziehung sieht es ähnlich aus, offenbar ist den meisten Nutzern klar, dass sie im Klartext gespeichert wird. 58,9% liegen bei der Beantwortung der Frage richtig und nur 8,3% explizit falsch (32% machen keine Angaben oder sind unsicher). Auch wissen die meisten Nutzer, dass eine Freundschaft für alle anderen Nutzer einsehbar ist (50 richig, 8 falsch, 15 ohne Angabe). Dennoch müsste dieser Zusammenhang den Nutzern noch deutlicher gemacht werden.

Die Funktion Freundschaften zu verstecken wurde von den Befragten offenbar nicht intuitiv vertanden. Nur 31,5% geben richtigerweise an, dass die Beziehung nach wie vor in der Datenbank im Klartext gespeichert ist und daher vom Netzwerkbetreiber eingesehen werden kann. Hier liegen sogar 43,2% falsch und 24% machen keine Angaben oder sind sich nicht sicher. Dieser Wert deckt sich auch mit der Beantwortung der Frage, ob der vierte Test, die Freundschaft mit Bob zu verstecken, erfolgreich ausgeführt wurde. Dieser Test ist von den vier Testen am wenigsten erfolgreich ausgefallen hier geben nur 76,7% an ihn erfolgreich durchgeführt zu habe. Auch wurde in einigen Kommentaren die “Verstecken“-Funktion kritisiert. Die Funktion sollte also auch überarbeitet werden. Eine Möglichkeit wäre es, im Userinterface deutlicher zu machen, was „verstecken“ genau heißt. Es wäre auch denkbar die Funktion komplett zu entfernen.

Kapitel 8

Fazit

In dieser Diplomarbeit wurde ein soziales Netzwerk entworfen, welches die Privatsphäre der Anwender durch asymmetrische Verschlüsselung im Webbrowser schützt. Die Referenzimplementierung pribook.com bietet die Möglichkeit verschlüsselte Nachrichten zu versenden und das ohne Installation, systemübergreifend und einfach. Es bietet die wichtigsten Funktionen von sozialen Netzwerken und weist nur die, bei einem Prototyp üblichen, kleineren Bugs auf. Die Ergebnisse der Befragung sind ein erstes Indiz dafür, dass pribook.com die einfachste Möglichkeit ist, um verschlüsselt zu kommunizieren. Es sollten noch die weiteren Vorschläge aus der Diplomarbeit bezüglich der Sicherheit umgesetzt werden, wie SSL, ein Browser-Plug-in und eine Schlüsselverwaltung. Dann könnte die Idee von pribook.com, erweitert um die vielen Funktionen der etablierten sozialen Netzwerke, eine echte Alternative darstellen.

An dieser Stelle soll die Diplomarbeit kurz zusammengefasst und ein Ausblick gegeben werden.

Zu Begin der Diplomarbeit wurde zunächst motiviert, warum die Entwicklung eines solchen Netzwerkes sinnvoll ist. Es wurde beschrieben, dass soziale Netzwerke heute viel genutzte IT-Anwendungen sind, die aber häufig wegen ihres Mangels an Datenschutz kritisiert werden. Außerdem wurde beschrieben, dass der Datenschutz auch an anderen Stellen gefährdet wird und trotzdem der größte Teil der Kommunikation zum Beispiel über Emails unverschlüsselt stattfindet. Es wurde weiter dargelegt, dass dies unter anderem daran liegt, weil die Installation von Kryptografiesoftware für normale Endverbraucher zu aufwendig ist.

Im weiteren Verlauf wurden Projekte vorgestellt, die ebenfalls den Datenschutz in sozialen Netzwerken verbessern sollen. Es wurden Projekte vorgestellt, die auf Open Source, verteil-

te Architekturen, Peer-to-Peer Netzwerke oder Verschlüsselung zum besseren Schutz der Privatsphäre setzten. Es wurde aber auch beschrieben, dass es bisher kein Projekt gibt, was Verschlüsselung auf dem kompletten Weg vom Sender bis zum Empfänger gewährt und gleichzeitig ohne eine Installation von zusätzlicher Software auskommt.

Nun wurden Invarianten aufgestellt, wie ein sicheres soziales Netzwerk aussehen sollte und wie dies umgesetzt werden könnte. Als Invarianten wurden festgelegt:

- Verschlüsselung auf dem kompletten Weg vom Sender zum Empfänger
- Asymmetrische Verschlüsselung, die einen öffentlichen Schlüsselaustausch möglich macht
- Bedienung im Webbrowser für eine möglichst einfache Teilnahme, ohne Installation von zusätzlicher Software und weitgehend systemunabhängig

Des Weiteren wurde beschrieben, wie diese Invarianten realisiert werden könnten. Nämlich durch ein soziales Netzwerk, dessen HTML-Datei, die beim Aufrufen der Seite heruntergeladen wird, JavaScript enthält, der die Nachrichten und persönlichen Informationen vor dem Versenden mit RSA verschlüsselt. Sonst kann das soziale Netzwerk wie ein herkömmliches soziales Netzwerk mit einem zentralen Server und einer zentralen Datenbank aufgebaut sein. Für die Verschlüsselung muss für jeden Anwender bei der Anmeldung ein RSA-Schlüsselpaar erstellt und der öffentliche Schlüssel an den Server gesendet und in der Datenbank gespeichert werden. Damit die Nutzer sich bei der Anmeldung nur ihr Passwort merken müssen, kann der private Schlüssel mit dem Passwort der Anwender verschlüsselt und dann auch im Netzwerk gespeichert werden. Wird eine Nachricht verschickt, so wird zunächst der öffentliche Schlüssel des Empfängers aus der Datenbank geholt und dann der Text mit diesem verschlüsselt und erst dann an den Server gesendet. Wenn sich der Empfänger mit seinem Passwort einloggt, wird zunächst sein privater Schlüssel vom Server heruntergeladen und lokal mit seinem Passwort entschlüsselt. Dann können auch die verschlüsselten Nachrichten mit dem privaten Schlüssel entschlüsselt werden.

Außerdem wurde dargelegt, dass auch Profilinformationen, Statusnachrichten und Gruppeninformationen auf diese Weise verschlüsselt werden können, wohingegen es sinnvoll ist, Freundschaftsrelationen und Gruppen im Klartext zu speichern. Zuvor wurden bereits die Grundlagen für die dieses Netzwerk, die Verschlüsselungsalgorithmen RSA, AES und der Hashalgorithmus SHA vorgestellt und erläutert.

Anschließend wurde die Referenzimplementierung pribook.com vorgestellt, die das zuvor beschriebene soziale Netzwerk realisiert. Es wurde der Aufbau der Datenbank erläutert

und eine Übersicht über den Quellcode gegeben. Anschließend wurde in einem Handbuch dargelegt, wie pribook für den Endbenutzer zu bedienen ist.

Schließlich wurde eine Befragung vorgestellt, mit der pribook.com evaluiert wurde. Die Fragebögen der 73 Teilnehmer konnten ein erstes Indiz dafür geben, dass Pribook.com die einfachste Möglichkeit ist, um verschlüsselt zu kommunizieren. Außerdem wurden die Thesen die zum Projekt motiviert haben, unter anderem, dass sich viele Menschen einen besseren Datenschutz für soziale Netzwerke wünschen, bestätigt.

Insgesamt konnte das Projekt weitgehend wie geplant realisiert werden. Pribook.com ist ein soziales Netzwerk, dass den Nutzern erlaubt mit RSA verschlüsselte Nachrichten zu verschicken. Dabei funktioniert es ohne Installation von zusätzlicher Software, systemübergreifend und relativ einfach. Die Ergebnisse der Befragung sind ein erstes Indiz dafür, dass Pribook.com die einfachste Möglichkeit ist, um verschlüsselt zu kommunizieren. Pribook.com bietet den Nutzern die wichtigsten Funktionen von sozialen Netzwerken, die nur die, bei einem Prototyp üblichen, kleineren Bugs aufweisen:

- Registrierung und Anmeldung
- Das Verschicken von persönlichen Nachrichten
- Schließen von Freundschaften
- Das Verstecken von Freundschaften
- Das Speichern von Profilinformationen
- Das Versenden von Nachrichten an viele Empfänger (Statsunachrichten)
- Das Verwalten von Freunden mit Circles
- Das Gründen von Gruppen
- Das Versenden von Fotos

Aus der Perspektive der Sicherheit sollten in Zukunft noch die weiteren Vorschläge umgesetzt werden, so wie sie schon in der Diplomarbeit beschrieben werden:

- SSL-Verschlüsselung zwischen Server und Client um Kommunikationsumstände und Authentifizierungen vor Dritten zu schützen.
- Automatische Überprüfung von vertrauten öffentlichen RSA-Schlüsseln mithilfe von Hashfunktion und Passwort.

- Ein Plug-in für einen gängigen Webbrowser, das automatisch den Quelltext überprüft.
- Eine genaue Überprüfung, inwieweit Sicherheitslücken im Webbrowser die Sicherheit von pribook.com gefährden.

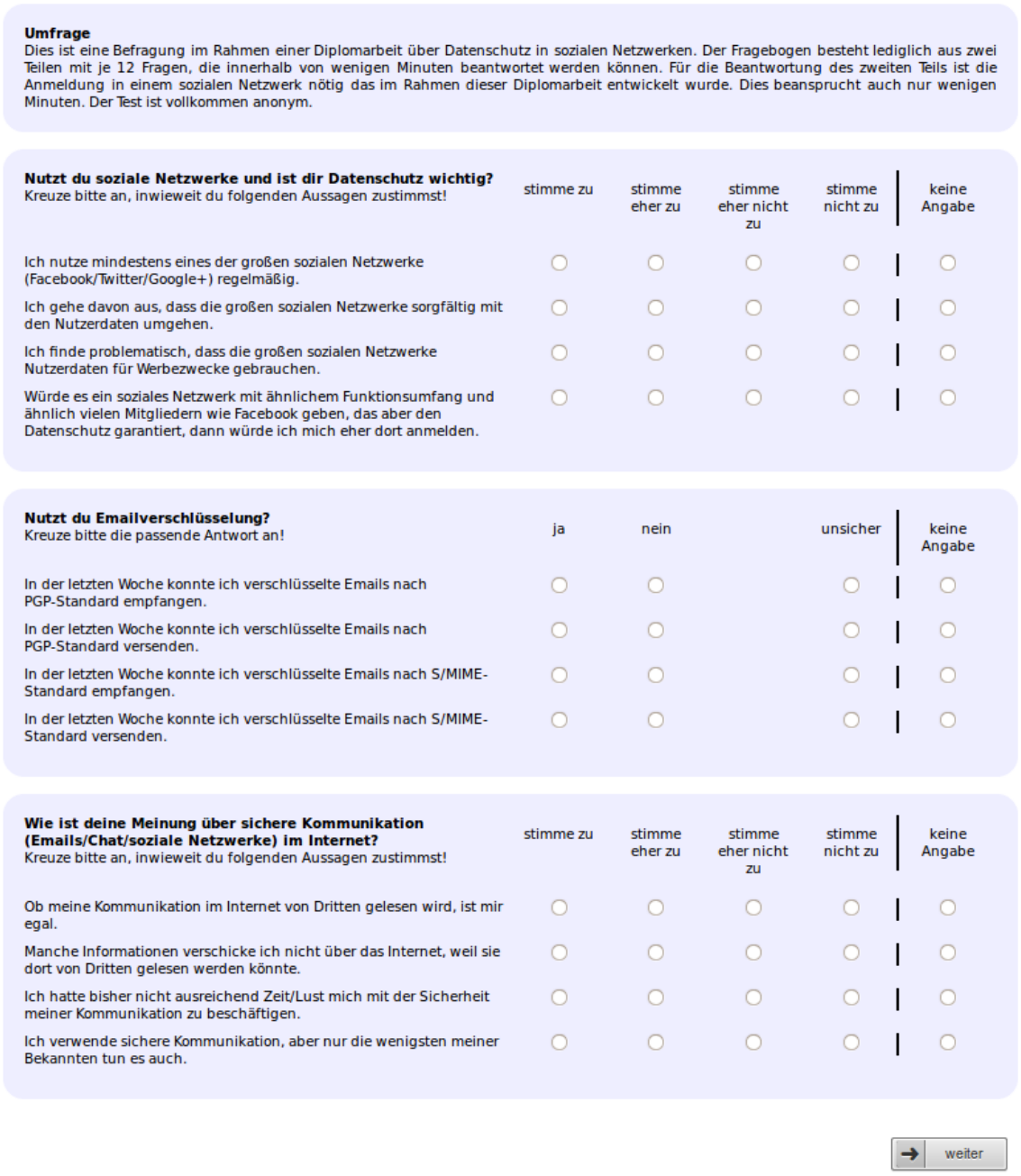
Um in der Praxis eine echte Alternative darzustellen, können noch einige Verbesserungen am sozialen Netzwerk vorgenommen werden:

- Die Ver- und Entschlüsselung dauert noch zu lange. Sie könnte zum Beispiel mit Web Workern parallelisiert werden so, dass sie weniger störend ist oder sogar auf mehrere Prozessoren verteilt wird.
- Auch pribook.com könnte grundsätzlich als verteiltes System mit mehreren Servern aufgebaut werden.
- Um mit den etablierten sozialen Netzwerken mitzuhalten, ist die Realisierung von noch weiteren Funktionen nötig. Zum Beispiel, das Kommentieren und Weiterleiten von Nachrichten, das versenden von Videos und so weiter.

Vielleicht kann die Idee dieser Diplomarbeit ein Anstoß sein, für eine der vielen Gemeinschaften, die zum Zeil haben ein alternatives und sicheres soziales Netzwerk zu entwickeln. Da eine sichere Kommunikation ein wichtiges Bedürfnis von vielen Menschen ist, könnte eine Gemeinschaft, die ein solches Netzwerk weiter entwickelt vielleicht sogar eine echte Alternative zu den etablierten Netzwerken darstellen.

Anhang A

Fragebogen



Umfrage
Dies ist eine Befragung im Rahmen einer Diplomarbeit über Datenschutz in sozialen Netzwerken. Der Fragebogen besteht lediglich aus zwei Teilen mit je 12 Fragen, die innerhalb von wenigen Minuten beantwortet werden können. Für die Beantwortung des zweiten Teils ist die Anmeldung in einem sozialen Netzwerk nötig das im Rahmen dieser Diplomarbeit entwickelt wurde. Dies beansprucht auch nur wenigen Minuten. Der Test ist vollkommen anonym.

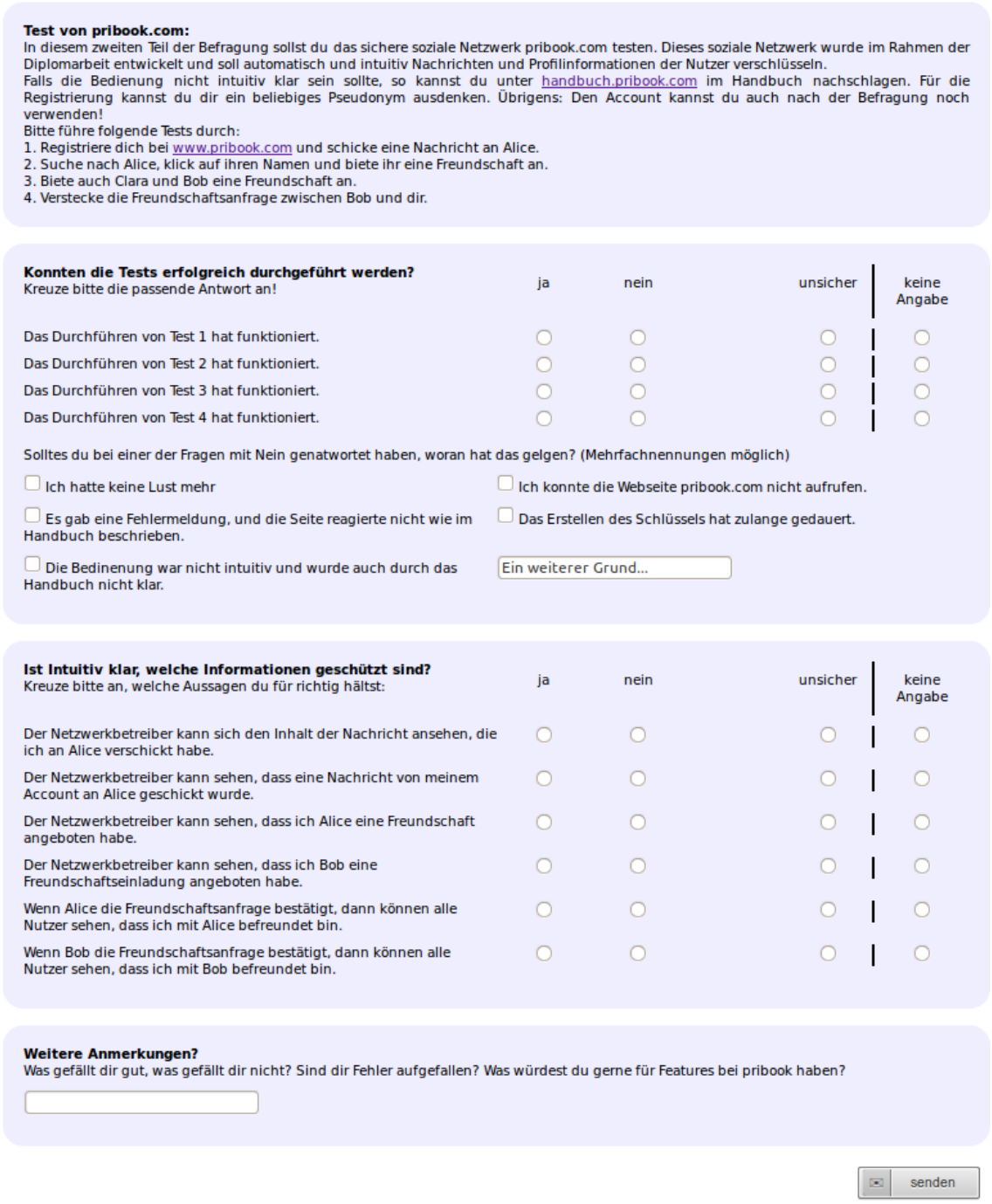
Nutzt du soziale Netzwerke und ist dir Datenschutz wichtig? Kreuze bitte an, inwieweit du folgenden Aussagen zustimmst!	stimme zu	stimme eher zu	stimme eher nicht zu	stimme nicht zu	keine Angabe
Ich nutze mindestens eines der großen sozialen Netzwerke (Facebook/Twitter/Google+) regelmäßig.	○	○	○	○	○
Ich gehe davon aus, dass die großen sozialen Netzwerke sorgfältig mit den Nutzerdaten umgehen.	○	○	○	○	○
Ich finde problematisch, dass die großen sozialen Netzwerke Nutzerdaten für Werbezwecke gebrauchen.	○	○	○	○	○
Würde es ein soziales Netzwerk mit ähnlichem Funktionsumfang und ähnlich vielen Mitgliedern wie Facebook geben, das aber den Datenschutz garantiert, dann würde ich mich eher dort anmelden.	○	○	○	○	○

Nutzt du Emailverschlüsselung? Kreuze bitte die passende Antwort an!	ja	nein	unsicher	keine Angabe
In der letzten Woche konnte ich verschlüsselte Emails nach PGP-Standard empfangen.	○	○	○	○
In der letzten Woche konnte ich verschlüsselte Emails nach PGP-Standard versenden.	○	○	○	○
In der letzten Woche konnte ich verschlüsselte Emails nach S/MIME-Standard empfangen.	○	○	○	○
In der letzten Woche konnte ich verschlüsselte Emails nach S/MIME-Standard versenden.	○	○	○	○

Wie ist deine Meinung über sichere Kommunikation (Emails/Chat/soziale Netzwerke) im Internet? Kreuze bitte an, inwieweit du folgenden Aussagen zustimmst!	stimme zu	stimme eher zu	stimme eher nicht zu	stimme nicht zu	keine Angabe
Ob meine Kommunikation im Internet von Dritten gelesen wird, ist mir egal.	○	○	○	○	○
Manche Informationen verschicke ich nicht über das Internet, weil sie dort von Dritten gelesen werden könnte.	○	○	○	○	○
Ich hatte bisher nicht ausreichend Zeit/Lust mich mit der Sicherheit meiner Kommunikation zu beschäftigen.	○	○	○	○	○
Ich verwende sichere Kommunikation, aber nur die wenigsten meiner Bekannten tun es auch.	○	○	○	○	○

weiter

Abbildung A.1: Fragebogen Teil1



Test von pribook.com:
In diesem zweiten Teil der Befragung sollst du das sichere soziale Netzwerk pribook.com testen. Dieses soziale Netzwerk wurde im Rahmen der Diplomarbeit entwickelt und soll automatisch und intuitiv Nachrichten und Profilinformationen der Nutzer verschlüsseln.
Falls die Bedienung nicht intuitiv klar sein sollte, so kannst du unter handbuch.pribook.com im Handbuch nachschlagen. Für die Registrierung kannst du dir ein beliebiges Pseudonym ausdenken. Übrigens: Den Account kannst du auch nach der Befragung noch verwenden!
Bitte führe folgende Tests durch:
1. Registriere dich bei www.pribook.com und schicke eine Nachricht an Alice.
2. Suche nach Alice, klick auf ihren Namen und biete ihr eine Freundschaft an.
3. Biete auch Clara und Bob eine Freundschaft an.
4. Verstecke die Freundschaftsanfrage zwischen Bob und dir.

Konnten die Tests erfolgreich durchgeführt werden?
Kreuze bitte die passende Antwort an!

	ja	nein	unsicher	keine Angabe
Das Durchführen von Test 1 hat funktioniert.	○	○	○	○
Das Durchführen von Test 2 hat funktioniert.	○	○	○	○
Das Durchführen von Test 3 hat funktioniert.	○	○	○	○
Das Durchführen von Test 4 hat funktioniert.	○	○	○	○

Solltes du bei einer der Fragen mit Nein genatwortet haben, woran hat das gelgen? (Mehrfachnennungen möglich)

☐ Ich hatte keine Lust mehr
☐ Ich konnte die Webseite pribook.com nicht aufrufen.
☐ Es gab eine Fehlermeldung, und die Seite reagierte nicht wie im Handbuch beschrieben.
☐ Das Erstellen des Schlüssels hat zulange gedauert.
☐ Die Bedinenung war nicht intuitiv und wurde auch durch das Handbuch nicht klar.
Ein weiterer Grund...

Ist Intuitiv klar, welche Informationen geschützt sind?
Kreuze bitte an, welche Aussagen du für richtig hältst:

	ja	nein	unsicher	keine Angabe
Der Netzwerkbetreiber kann sich den Inhalt der Nachricht ansehen, die ich an Alice verschickt habe.	○	○	○	○
Der Netzwerkbetreiber kann sehen, dass eine Nachricht von meinem Account an Alice geschickt wurde.	○	○	○	○
Der Netzwerkbetreiber kann sehen, dass ich Alice eine Freundschaft angeboten habe.	○	○	○	○
Der Netzwerkbetreiber kann sehen, dass ich Bob eine Freundschaftseinladung angeboten habe.	○	○	○	○
Wenn Alice die Freundschaftsanfrage bestätigt, dann können alle Nutzer sehen, dass ich mit Alice befreundet bin.	○	○	○	○
Wenn Bob die Freundschaftsanfrage bestätigt, dann können alle Nutzer sehen, dass ich mit Bob befreundet bin.	○	○	○	○

Weitere Anmerkungen?
Was gefällt dir gut, was gefällt dir nicht? Sind dir Fehler aufgefallen? Was würdest du gerne für Features bei pribook haben?

senden

Abbildung A.2: Fragebogen Teil2

Anhang B

Ergebnisse der Befragung

Kommentare der Befragung:

“- Name der Person oben aufs Profil schreiben – damit ich auch weiß wessen Seite ich betrachte, es ist mir uch leider unklar wie ich mir Gruppen suchen kann (erstellen ist jedoch intuitiv klar)“

“Natürlich unansehnlich im Vergleich zu fb und die ganze Bild-, Video- und Linkfuntionen wie auch Notizzeugs und Pinnwände etc. fehlen scheinbar. Das macht die Sache erst interessanter als Emails. “

“Hi Finn! Wenn du das quasi wie facebook aufziehen musst, dann sollte es möglich sein, dass man Vor- und Nachnamen angeben kann, damit andere einen auch finden. Das geht aber scheinbar nicht, weil keine Leerzeichen enthalten sein dürfen, oder? “

“Ich würde mir an deiner Stelle noch einmal Gedanken über den Namen machen - ich nehme an, pribook soll Englisch ausgesprochen werden, also ““preibook““. Auf Englisch hört sich das exakt an wie das Verb ““to pry““, was so viel bedeutet wie ““herumschnüffeln““, ““neugierig sein““, ““ausspähen““ - also ein grob unpassender Name für dein Projekt. “

“Ich wusste nicht was ich mit dem Key machen sollte, der mir zugeteilt wurde. Muss man den speichern? Muss man den irgendwann wieder eingeben¿‘

“Es macht den Anschein, dass es sicherer ist als z.B. Facebook, das gefällt mir gut.“

“nettes Projekt. Foto-Upload müsste natürlich unterstützt werden um eine Alternative darzustellen ;)“

Tabelle1

Ich nutze mindestens eines der großen sozialen Netzwerke (Facebook/Twitter/Google+) regelmäßig.

rb1-1	stimme zu	49	67,12%
rb1-2	stimme ehr zu	8	10,96%
rb1-3	stimme eher nicht zu	1	1,37%
rb1-4	stimme nicht zu	15	20,55%

Ich gehe davon aus, dass die großen sozialen Netzwerke sorgfältig mit den Nutzerdaten umgehen.

rb2-1	stimme zu	1	1,37%
rb2-2	stimme ehr zu	5	6,85%
rb2-3	stimme eher nicht zu	23	31,51%
rb2-4	stimme nicht zu	44	60,27%

Ich finde problematisch, dass die großen sozialen Netzwerke Nutzerdaten für Werbezwecke gebrauchen.

rb3-1	stimme zu	49	67,12%
rb3-2	stimme ehr zu	13	17,81%
rb3-3	stimme eher nicht zu	5	6,85%
rb3-4	stimme nicht zu	6	8,22%

Würde es ein soziales Netzwerk mit ähnlichem Funktionsumfang und ähnlich vielen Mitgliedern wie Facebook geben, das aber den Datenschutz garantiert, dann würde ich mich eher dort anmelden.

rb4-1	stimme zu	51	69,86%
rb4-2	stimme ehr zu	11	15,07%
rb4-3	stimme eher nicht zu	6	8,22%
rb4-4	stimme nicht zu	2	2,74%
rb4-5	keine Angabe	3	4,11%

In der letzten Woche konnte ich verschlüsselte Emails nach PGP-Standard empfangen.

rb21-1	ja	22	30,14%
rb21-2	nein	28	38,36%
rb21-4	unsicher	20	27,40%
rb21-5	keine Angabe	3	4,11%

In der letzten Woche konnte ich verschlüsselte Emails nach PGP-Standard versenden.

rb22-1	ja	17	23,29%
rb22-2	nein	33	45,21%
rb22-4	unsicher	20	27,40%
rb22-5	keine Angabe	3	4,11%

In der letzten Woche konnte ich verschlüsselte Emails nach S/MIME-Standard empfangen.

rb23-1	ja	18	24,66%
rb23-2	nein	30	41,10%
rb23-4	unsicher	22	30,14%
rb23-5	keine Angabe	3	4,11%

In der letzten Woche konnte ich verschlüsselte Emails nach S/MIME-Standard versenden.

rb24-1	ja	17	23,29%
rb24-2	nein	32	43,84%
rb24-4	unsicher	21	28,77%
rb24-5	keine Angabe	3	4,11%

Seite 1

Abbildung B.1: Ergebnisse der Befragung

Tabelle1

Ob meine Kommunikation im Internet von Dritten gelesen wird, ist mir egal.

rb31-1	stimme zu	3	4,11%
rb31-2	stimme ehr zu	2	2,74%
rb31-3	stimme eher nicht zu	18	24,66%
rb31-4	stimme nicht zu	50	68,49%

Manche Informationen verschicke ich nicht über das Internet, weil es dort von Dritten gelesen werden könnte.

rb32-1	stimme zu	39	53,42%
rb32-2	stimme ehr zu	19	26,03%
rb32-3	stimme eher nicht zu	10	13,70%
rb32-4	stimme nicht zu	5	6,85%

Ich hatte bisher nicht ausreichend Zeit/Lust mich mit der Sicherheit meiner Kommunikation zu beschäftigen.

rb33-1	stimme zu	7	9,59%
rb33-2	stimme ehr zu	25	34,25%
rb33-3	stimme eher nicht zu	25	34,25%
rb33-4	stimme nicht zu	16	21,92%

Ich verwende sichere Kommunikation, aber nur die wenigsten meiner Bekannten tun es auch.

rb34-1	stimme zu	18	24,66%
rb34-2	stimme ehr zu	16	21,92%
rb34-3	stimme eher nicht zu	24	32,88%
rb34-4	stimme nicht zu	8	10,96%
rb34-5	keine Angabe	7	9,59%

Melde dich bei Pribook an und schicke eine Nachricht an „Alice“.
Das Durchführen von Test 1 hat funktioniert.

rb201-1	ja	60	82,19%
rb201-2	nein	3	4,11%
rb201-4	unsicher	2	2,74%
rb201-5	keine Angabe	8	10,96%

Suche nach Alice, klick auf ihren Namen und biete ihr eine Freundschaft an.
Das Durchführen von Test 2 hat funktioniert.

rb202-1	ja	61	83,56%
rb202-2	nein	3	4,11%
rb202-4	unsicher	1	1,37%
rb202-5	keine Angabe	8	10,96%

Biete auch Clara und Bob eine Freundschaft an.
Das Durchführen von Test 3 hat funktioniert.

rb203-1	ja	61	83,56%
rb203-2	nein	3	4,11%
rb203-4	unsicher	1	1,37%
rb203-5	keine Angabe	8	10,96%

Verstecke die Freundschaftsanfrage zwischen Bob und dir.
76%Das Durchführen von Test 4 hat funktioniert.

rb204-1	ja	56	76,71%
rb204-2	nein	4	5,48%
rb204-4	unsicher	5	6,85%
rb204-5	keine Angabe	8	10,96%

Seite 2

Abbildung B.2: Ergebnisse der Befragung 2

Tabelle1

Der Netzwerkbetreiber kann sich den Inhalt der Nachricht ansehen, die ich an Alice verschickt habe.

rb211-1	ja	8	10,96%
rb211-2	nein	40	54,79%
rb211-4	unsicher	17	23,29%
rb211-5	keine Angabe	8	10,96%

Der Netzwerkbetreiber kann sehen, dass eine Nachricht von meinem Account an Alice geschickt wurde.

rb212-1	ja	33	45,21%
rb212-2	nein	8	10,96%
rb212-4	unsicher	24	32,88%
rb212-5	keine Angabe	8	10,96%

Der Netzwerkbetreiber kann sehen, dass ich mit Alice befreundet bin.

rb213-1	ja	43	58,90%
rb213-2	nein	6	8,22%
rb213-4	unsicher	16	21,92%
rb213-5	keine Angabe	8	10,96%

Der Netzwerkbetreiber kann sehen, dass ich mit Bob befreundet bin.

rb214-1	ja	23	31,51%
rb214-2	nein	25	34,25%
rb214-4	unsicher	17	23,29%
rb214-5	keine Angabe	8	10,96%

Alle Nutzer können sehen, dass ich mit Alice befreundet bin.

rb215-1	ja	50	68,49%
rb215-2	nein	8	10,96%
rb215-4	unsicher	7	9,59%
rb215-5	keine Angabe	8	10,96%

Alle Nutzer können sehen, dass ich mit Bob befreundet bin.

rb216-1	ja	8	10,96%
rb216-2	nein	51	69,86%
rb216-4	unsicher	6	8,22%
rb216-5	keine Angabe	8	10,96%

Solltest du bei einer Frage mit Nein geantwortet haben, gib bitte die Gründe an!
(Mehrfachnennungen möglich)
Ich hatte keine Lust mehr

cb2051-0	ja	68	93,15%
cb2051-1	nein	5	6,85%

Ich konnte die Webseite pribook.com nicht aufrufen.

cb2052-0	ja	73	100,00%
cb2052-1	nein	0	

Das Erstellen des Schlüssels hat zu lange gedauert.

cb2053-0	ja	72	98,63%
cb2053-1	nein	1	1,37%

Es gab eine Fehlermeldung und/oder die Seite reagierte nicht wie im Handbuch beschrieben.

Cb2054-0	ja	72	98,63%
cb2054-1	nein	1	1,37%

Die Bedienung war nicht intuitiv und wurde auch durch das Handbuch nicht klarer.

cb2055-0	ja	68	93,15%
cb2055-1	nein	5	6,85%

Seite 3

Abbildung B.3: Ergebnisse der Befragung 3

“Sobald ich eine Freundschaft anbiete, erscheint die Meldung ““Freundschaft erfolgreich geschlossen““. Dies ist nach meinem Verständnis aber erst nach Bestätigung des anderen der Fall“

“ein übersichtliches design mit einfacher bedienbarkeit á la applebenutzeroberfläche.“

“Das Privatsphäre an erster Stelle steht ist super! Die Aufmachung sollte nur noch der Idee gerecht werden¡‘

“Ich finde die Idee super und halte sie für zukunftstauglich. In der Beta-Phase wäre ein aufwändiges Design zu viel verlangt und kleine Macken (Zeilenumbrüche in Nachrichten werden nicht dargestellt; Nachrichtenformular bleibt ausgefüllt stehen, nachdem Nachricht versendet wurde ...) sind absolut verzeihbar. Wäre toll, wenn aus dem Projekt mehr würde und dann auch solche Dinge in Angriff genommen würden. Viel Erfolg weiterhin¡‘

“passwort war zu kurz, Fehler wurde gemeldet, dann Endlosschleife bei RSA Verschlüsselung“

“Anzeige ob Freundschaftsanfrage versteckt ist oder nicht wär gut, Benutzermäßig nich gan einsehbar“

“Mein Problem mit Faceboom ist nicht in erster Linie, dass der Netzbetreiber mitlesen kann, sondern dass meine Informationen auf den Facebook-Servern liegen. Trotzdem ein interessanter Ansatz, Respekt¡‘

“Beim anklicken der Ja/Nein Anzeige für das verstekcen der Freunschadftsanzeige fehlte mir noch ein Button für ““Änderungen speichern o.ä.““ hab ich erstmal länger gescuht, da mir nicht klar war, dass durch bloßes anklicken des Buttons die Einstellung quasi gespeichert ist :)“

“Ist ok“ “das rad wurde bereit erfunden“

“Für ein prototyp ist Gut. Datenschutz und Datensicherheit sind oft heiße Themen“

“Farben sind nciht schön. Logo auch nciht. ansonsten gute idee, es muesste aber optisch ansprechender und wohlgeordneter werden und mit mehr symbolen versehen.“

“Design könnte verbessert werden und es sollte leichter sein Gruppen beizutreten“

“Fragen teilweise unpräzise.“

“Mir gefällt nicht, dass ich mich explizit anmelden muss -> Zu hohe partizipationshürde sowie grundsätzliches Profilingpotential. :-D“

“public/private key wird unter normalen http übermittelt?! “

“Der Browser sollte beim erzeugen des Schlüssels vielleicht nicht unbedingt einfrieren. Das wirkt sehr unprofessionell. Die Sicherheit der Daten ist nicht intuitiv klar. Woher weiß ich, dass es wirklich sicher ist¿‘

“Es gibt einen Unterschied zwischen ““Der Netzwerkbetreiber kann [...]““ und ““Der Netzwerkbetreiber behauptet, dass er ... (nicht) kann““. Wie kann der Durchschnittsnutzer dem Netzwerkbetreiber vertrauen? Kann der Nutzer nur der Ausführung von Code zustimmen, der öffentlich gemacht wurde und unabhängig überprüft wurde? Wem kann der Nutzer für so eine Überprüfung vertrauen? Wie ist sichergestellt, dass keine Man-in-the-middle-Angriffe stattfinden, und zwar z.B. auf den übermittelten JavaScript Code¿‘

“Diese Pop-up-Fenster nach dem Versenden einer Nachricht sind etwas nervig.“

“Features: Fotos“

“Hinweis zur Beantwortung von intutiv klar sein welche Infos geschützt sind - die unsicheren Antworten sind bei mir unsicher weil ich nicht weiß ob du auch meine Freundeslisten und Liste verschickter Nachrichten verschlüsselst.Tolle neue Features: Andockung an Facebook, dass man über pribook verschlüsselte Nachrichten an Facebook-Freunde schreiben kann“

Abbildungsverzeichnis

Algorithmenverzeichnis

Literaturverzeichnis

[1] Andrey Bogdanov, Dmitry Khovratovich, Christian Rechberger: *A new method for known plaintext attack of FEAL cipher*, 2011.

[2] Bager, Jo: *Private Treffpunkte. Diaspora und andere Facebook-Alternativen. — In: c't. Nr. 5/2012. S. 136–139 (hier: S. 137).*, 2012.

[3] Bendrath, Ralf: *PGP - die ersten zehn Jahre — Telepolis*. `http://www.heise.de/tp/artikel/7/7175/1.html`, 2012. [Online; Stand 26. März 2012].

[4] Bernstein, Daniel J.: *Cache-timing attacks on AES*, 2005.

[5] Bleich, Holger: *Facebooks Börsengang nun offiziell — heise online*. `http://heise.de/-1426890`, 2012. [Online; Stand 26. März 2012].

[6] buddycloud: *buddycloud*. `http://buddycloud.com/`, 2012. [Online; Stand 26. März 2012].

[7] Buddypress: *Buddypress - Social networking, in a box*. `http://buddypress.org/`, 2012. [Online; Stand 26. März 2012].

[8] BUNDESVERFASSUNGSGERICHT [Hrsg.]: *Volkszählungsurteil - Aktenzeichen: 1 BvR 209, 269, 362, 420, 440, 484/83*. 1983.

[9] BUNDESVERFASSUNGSGERICHT [Hrsg.]: *Pressemitteilung Nr. 11/2010 vom 2. März 2010*. 2010.

[10] Cloer, Thomas: *Google nennt neue Nutzerzahlen zu Google+ — computerwoche.de*. `www.computerwoche.de/netzwerke/web/2506487/`, 2012. [Online; Stand 26. März 2012].

[11] Crabgrass: *Crabgrass - Software libre for social organizing*. `http://crabgrass.riseuplabs.org/`, 2012. [Online; Stand 26. März 2012].

[12] Cutillo, Leucio Antonio, Refik Molva und Thorsten Strufe: *Safebook: Feasibility of transitive cooperation for privacy on a decentralized social network*. In: *WOWMOM*, Seiten 1–6. IEEE, 2009.

[13] derStandard.at: *800 Millionen User nutzen Facebook monatlich — derStandard.at.* http://derstandard.at/1316733436483/Wachstum-800-Millionen-User-nutzen-Facebook-monatlich, 2012. [Online; Stand 26. März 2012].

[14] derStandart.at: *Diaspora lädt zu überarbeiteter Alpha-Version.* http://derstandard.at/1319182744096/Netzwerk-Diaspora-laedt-zu-ueberarbeiteter-Alpha-Version, 2012. [Online; Stand 26. März 2012].

[15] Deutscher Bundestag [Hrsg.]: *Bericht über die Durchführung sowie Art und Umfang der Maßnahmen, Drucksache 17/8639.* 2012.

[16] Diaspora*: *Diaspora.* https://www.joindiaspora.com/, 2012. [Online; Stand 26. März 2012].

[17] DIASPORA: *How many users are in the DIASPORA network?* https://diasp.eu/stats.html, 2012. [Online; Stand 26. März 2012].

[18] Eli Biham, Adi Shamir: *Differential cryptanalysis of DES-like cryptosystems*, 1991.

[19] FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION: *Secure Hash Standard (SHS)*, 2008.

[20] Federal Information Processing Standards Publication 197: *Announcing the ADVANCED ENCRYPTION STANDARD (AES)*, 2001.

[21] fellowship of fsfe: *Cloud Computing — fsfe.* http://wiki.fsfe.org/CloudComputing?highlight=%28cloudcomputing%29, 2012. [Online; Stand 26. März 2012].

[22] friendica: *friendica - because somebody has to stand up for the people of the internet.* http://friendica.com/, 2012. [Online; Stand 26. März 2012].

[23] futurezone.at: *Nutzer nur 3,3 Minuten pro Monat auf Google+ — futurezone.at - Technology News.* http://futurezone.at/digitallife/7692-nutzer-nur-3-3-minuten-pro-monat-auf-google.php, 2012. [Online; Stand 26. März 2012].

[24] HANNON, C. E. S: *Communication Theory of Secrecy Systems*, 1949.

[25] Hasselmann, Silke: *Google-Nutzer werden noch gläserner — tagesschau.de.* http://www.tagesschau.de/ausland/google414.html, 2012. [Online; Stand 26. März 2012].

[26] Hushmail: *Hushmail - Free Email with Privacy.* `http://secushare.org/`, 2012. [Online; Stand 26. März 2012].

[27] jappix project: *Jappix - your own social cloud.* `https://project.jappix.com/`, 2012. [Online; Stand 26. März 2012].

[28] Joan Daemen, Vincent Rijmen: *The Rijndael Block Chipher*, 1999.

[29] Kannenberg, Axel: *Facebook ändert Börsenprospekt — heise online.* `http://heise.de/-1466175`, 2012. [Online; Stand 26. März 2012].

[30] Kirsch, Christian: *Geheimdienste überwachten 37 Millionen Netzverbindungen — heise online.* `http://heise.de/-1442867`, 2012. [Online; Stand 26. März 2012].

[31] Kryptos - Fakultät für Informatik Uni Oldenburg: *Secure Hash Algorithm 2*, 2012. [Online; Stand 29. März 2012].

[32] Landgericht Berlin: *Landgericht Berlin: Facebook unterliegt der Verbraucherzentrale in Wettbewerbsprozess.* `http://www.berlin.de/sen/justiz/gerichte/kg/presse/archiv/20120306.1545.367067.html`, 2012. [Online; Stand 26. März 2012].

[33] Lucks, Stefan: *Zur Sicherheit kryptographischer Hashfunktionen und digitaler Signaturen*, 2005.

[34] Oka, Atsushi: *Titaniumcore Project - JavaScript Cryptography Toolkit*, 2011. [Online; Stand 29. März 2012].

[35] Peerscape: *Peerscape.* `http://www.peerscape.org/`, 2012. [Online; Stand 26. März 2012].

[36] Pfitzmann, Andreas: *Sicherheit in Rechnernetzen: Mehrseitige Sicherheit in verteilten und durch verteilte Systeme*, 2010.

[37] R.L. Rivest, A. Shamir, L. Adleman: *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems*, 1977.

[38] RSA Laporatories: *What is OAEP?*, 1994.

[39] RSA Laporatories: *RSA-768 is factored!*, 2010.

[40] Safebook: *Safebook - Secure Social Network.* `http://www.safebook.us`, 2012. [Online; Stand 26. März 2012].

[41] Schneier, Bruce: *SHA-1 Broken*, 2005. [Online; Stand 29. März 2012].

[42] Secure Share: *Secure Share.* `http://secushare.org/`, 2012. [Online; Stand 26. März 2012].

[43] TEST.DE: *Was Facebook alles erfährt — Stiftung Warentest.* http://www.test.de/themen/computer-telefon/meldung/Soziale-Netzwerke-und-Datenschutz-Was-Facebook-alles-erfaehrt-4271957-4271979/, 2012. [Online; Stand 26. März 2012].

[44] TISCHNER, FLORIAN: *SicheresPasswort.com.* http://www.sicherespasswort.com/, 2012. [Online; Stand 10. April 2012].

[45] UNABHÄNGIGES LANDESZENTRUM FÜR DATENSCHUTZ SCHLESWIG-HOLSTEIN: *ULD an Webseitenbetreiber: "Facebook-Reichweitenanalyse abschalten ".* https://www.datenschutzzentrum.de/presse/20110819-facebook.htm, 2012. [Online; Stand 26. März 2012].

[46] WIKIPEDIA: *Advanced Encryption Standard — Wikipedia, Die freie Enzyklopädie.* http://de.wikipedia.org/w/index.php?title=Advanced_Encryption_Standard&oldid=100523636, 2012. [Online; Stand 26. März 2012].

[47] WIKIPEDIA: *Diaspora (Software) — Wikipedia, Die freie Enzyklopädie*, 2012. [Online; Stand 28. März 2012].

[48] WIKIPEDIA: *RSA-Kryptosystem — Wikipedia, Die freie Enzyklopädie*, 2012. [Online; Stand 29. März 2012].

[49] WILKIN, CHRISTIAN: *Der Algorithmus des "Advanced Encryption Standard"*, 1999.

[50] WOBST, REINHARD: *AES unter Beschuss*, 2002. [Online; Stand 26. März 2012].

[51] WOLFGANG STIELER, BEN SCHWAN: *heise online — Facebook war "Navigationssystem "für arabische Revolution.* http://heise.de/-1351335, 2012. [Online; Stand 26. März 2012].

[52] WONDRACEK, GILBERT, THORSTEN HOLZ, ENGIN KIRDA und CHRISTOPHER KRUEGEL: *A Practical Attack to De-Anonymize Social Network Users.* 2010.

[53] ZIADINOVIC, DUSAN: *Facebook-Börsengang: Datenschutz contra Profit — heise online.* http://heise.de/-1428227, 2012. [Online; Stand 26. März 2012].

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet sowie Zitate kenntlich gemacht habe.

Dortmund, den 18. April 2012

Finn Siebert