Sichere und benutzerfreundliche Schlüsselverteilung auf Basis von QR-Codes

Kevin Körner, Thomas Walter

Fachbereich Informatik, Arbeitsbereich Informationsdienste
Eberhard Karls Universität Tübingen
Wächterstraße 76
72074 Tübingen
kevin.koerner@uni-tuebingen.de
thomas.walter@uni-tuebingen.de

Abstract: Datenschutzalgorithmen im Webumfeld und darüber hinaus basieren auf der Nutzung geheimer Schlüssel. Diese sind einerseits durch Kombination heterogener Zeichen höchstmöglich zu sichern, andererseits müssen die Schlüssel zur Nutzung auf allen Endgeräten des Nutzers verfügbar sein. Praktische Ansätze wie die Verteilung von Schlüsseln über E-Mail oder der Einsatz gut merkbarer Wörterbuchpasswörter sind zwar benutzerfreundlich, jedoch bezüglich Vertraulichkeit und Sicherheit fragwürdig. Die Problematik der benutzerfreundlichen Schlüsselverteilung wird durch den wirtschaftlichen Erfolg von mobilen Endgeräten zusätzlich interessant. Zum einen steigt die Anzahl an Personen mit mehr als einem (mobilen) Endgerät, zum anderen bieten mobile Endgeräte neue Technologien für die Nutzerinteraktion. In jedem Fall ist eine Auseinandersetzung mit benutzerfreundlichen Methoden zum Schlüsselaustausch zwischen beliebigen (mobilen) Endgeräten sinnvoll. Das vorliegende Dokument diskutiert einen Ansatz zur Verteilung von Schlüsseln zwischen unterschiedlichen (mobilen) Endgeräten. Hierbei führen wir das Konzept der Schlüsselverteilung über zweidimensionale Barcodes am Beispiel des QR-Codes ein, diskutieren potentielle Schlüsselspeicher-Architekturen und erörtern praktische Anwendungsszenarien.

1 Einleitung

Aus der Privatisierung des WorldWideWeb Anfang der 1990er Jahre ist eine Vielzahl an Diensten zur Kommunikation und zum automatisierten Datenaustausch über unsichere Netzwerke entstanden; beispielsweise Chat, Online-Banking, E-Mail und Cloud-Computing. Um Abhörsicherheit, Nachrichtenintegrität und Authentifizierung der Kommunikationspartner auf unsicheren Kanälen zu gewährleisten, existieren diverse Sicherheitsprotokolle (siehe auch [Lam04]). Diese basieren dabei auf der Geheimhaltung von Schlüsseln: Entweder eines allen Kommunikationspartnern bekannten symmetrischen Schlüssels oder der privaten Schlüssel bei asymmetrischen Verfahren. Bei letzteren besteht zudem noch das Problem, dass die Identität der Kommunikationspartner vor dem initialen Nachrichtenaustausch sichergestellt werden muss.
Dementsprechend müssen die Dienstnutzer in jedem Fall über einen sicheren Kanal In-

formationen austauschen, bevor sie gesichert über einen unsicheren Kanal kommunizieren können. Die Autoren von [LV99] erörtern hierbei, weshalb es notwendig ist, die verwendeten Schlüssel immer aufwendiger und größer zu gestalten. Andernfalls steigt die Wahrscheinlichkeit eines praktikabel durchführbaren Brute-Force-Angriffs mit steigender Rechnerleistung. Lampson erörtert in [Lam04] zudem die Herausforderungen an sichere computerbasierte Kommunikation und arbeitet heraus, weshalb Computer in der, wie er sie bezeichnet, „echten Welt“ nicht als sicher angesehen werden dürfen.
Mit steigender Schlüsselkomplexität steigt jedoch auch der Bedarf an benutzerfreundlichen und sicheren Ansätzen zum Austausch der initial benötigten Informationen. Die Entwicklung solcher Ansätze hängt der Entwicklung von Datensicherheitsmethoden unseres Erachtens hinterher. Wir stimmen den Autoren von [WT98] dahingehend zu, dass es ein Zusammenspiel zwischen Benutzerfreundlichkeit bei Datenschutztechnologien sowie deren Akzeptanz seitens der Nutzer gibt: Je aufwändiger eine Technologie für deren Nutzer ist, desto geringer ist ihre Akzeptanz.
Ende des ersten Jahrzehnts des 21. Jahrhunderts etablierten sich zudem die Technologien für mobile Endgeräte im privaten Umfeld; insbesondere Smartphones und Tablets. Aktuelle Studien (z.B. [AB13]) sagen voraus, dass die Anzahl der verwendeten Smartphones sich in den kommenden Jahren deutlich steigern wird. Für die Nutzer stellt sich daraus resultierend die Frage, wie benötigte Schlüssel auf sicherem Weg zwischen Endgeräten verteilt werden können. Insbesondere lohnt eine Betrachtung der neuen Interaktionsmöglichkeiten mit den mobilen Endgeräten, die von konventionellen Techniken nicht unterstützt werden.

Im vorliegenden Dokument präsentieren wir ein Konzept zur benutzerfreundlichen Verteilung von Schlüsseln zwischen (mobilen) Endgeräten unter Verwendung von QR-Codes. In Kapitel 2 wird der von uns aufgefasste aktuelle Forschungsstand erörtert. Die grundlegenden Architekturkomponenten diskutieren wir in Kapitel 3, gefolgt von potentiellen Schlüsselspeicher-Architekturen in Kapitel 4. Kapitel 5 diskutiert die präsentierte Idee und beschäftigt sich mit praktischen Anwendungsszenarien. In Kapitel 6 folgt abschließend eine Zusammenfassung des Dokuments sowie ein Ausblick auf künftige Forschungen in diesem Bereich.

2 Themenbezogene Arbeiten

2.1 Diffie-Hellman-Schlüsselaustausch

Eines der bekanntesten Schlüsselaustauschprotokolle ist 1976 von Diffie und Hellman in [DH76] eingeführt worden. Dieses erlaubt es Kommunikationspartnern, symmetrische Schlüssel sicher über einen unsicheren Kanal auszutauschen. Unberechtigte Dritte können die für den Schlüsselaustausch verwendeten Nachrichten mithören, ohne aus diesen auf den ausgetauschten Schlüssel schließen zu können. Ein essentieller Nachteil des Verfahrens ist die Anfälligkeit gegen einen Man-in-the-middle-Angriff aufgrund der fehlenden Authentifizierung der Kommunikationspartner (vgl. [Lam04]). Ansätze wie digitale Signaturen oder Message Authentication Codes (MACs) können mit dem Diffie-Hellman-

Verfahren kombiniert werden, um die Sicherheitslücke zu beheben. Für die Kommunikationspartner bedeutet dies jedoch zusätzlichen Organisationsaufwand, wie den Einsatz vertrauenswürdiger Public-Key-Infrastrukturen (vgl. [M+99]) oder den Kommunikationspartnern bekannte Geheimnisse. Unser Konzept kann hierbei mit geringerem Aufwand zum vertrauenswürdigen ad hoc Schlüsselaustausch verwendet werden.

2.2 Off-the-Record-Messaging (OTR)

Off-the-Record-Messaging (OTR) ist ein Protokoll zum sicheren initialen Schlüsselaustausch zwischen Kommunikationspartnern und wurde 2004 in [BGB04] eingeführt. Das Protokoll verwendet dabei das Diffie-Hellman-Schlüsselaustauschverfahren zum Schlüsselaustausch und stellt die Identität der Kommunikationspartner über MACs sicher. Eine weitere in OTR umgesetzte Anforderung besteht darin, dass von aufgedeckten Schlüsseln nicht auf andere für Kommunikationen verwendete Schlüssel geschlossen werden kann; beispielsweise von ausgetauschten Langzeitschlüsseln auf kurzlebige Sitzungsschlüssel. Unser Konzept erlaubt es Kommunikationspartnern ad hoc Schlüssel und eindeutige Identifizierungsmerkmale für die Authentifizierung in OTR auszutauschen.

2.3 QR-Codes

Nach eigenen Angaben (siehe [DW14]) wurde die QR-Code-Technologie erstmalig 1994 vom japanischen Unternehmen DENSO WAVE als Erweiterung des eindimensionalen Barcodes um eine zweite Dimension veröffentlicht. Die zweite Dimension ermöglicht es auf derselben Fläche deutlich mehr Zeichen in einem Code zu kodieren als in einem eindimensionalen Barcode. Nach [DW14] hängt die Anzahl an in einem QR-Code kodierbaren Zeichen dabei von der ihn erzeugenden Anwendung ab. Die Anzahl reduziert sich dabei durch verschieden ausgeprägte Fehlerkorrektur-Stufen, die den Ausgleich fehlerhafter Übertragungen erlauben. Der Standard [ISO00] ermöglicht beispielsweise 2953 8-Bit-Zeichen bei niedrigster Fehlerkorrektur-Stufe in einem QR-Code zu kodieren. Zudem erlauben es Positionsinformationen innerhalb des QR-Codes aus beliebigen Aufnahmepositionen die korrekte QR-Code-Darstellung zu errechnen. Insbesondere mobilen Endgeräten erleichtert dies die Aufnahme und Weiterverarbeitung dieser Codes.
Anwendungen wie [AK08] oder [TH14] nutzen die genannten Vorteile für eine möglichst benutzerfreundliche Überführung von Informationen auf mobile Endgeräte. Diverse Bibliotheken sind auf diese Weise aus Projekten entstanden, welche es Anwendungsentwicklern erleichtern die QR-Code-Funktionalitäten in ihre Applikationen zu integrieren; beispielsweise [DT14]. Nachteile von QR-Codes werden in [K+10] ausführlich erörtert; zum Beispiel die erschwerte visuelle Verifizierbarkeit von QR-Code-Inhalten. Dies vereinfacht bekannte Angriffe wie SQL-Injection sowie Social Engineering und Phishing.
Trotz des intuitiven Ansatzes Schlüssel über QR-Codes zwischen (mobilen) Endgeräten zu verteilen, haben wir bisher bei unserer Nachforschung keine wissenschaftlichen Projekte

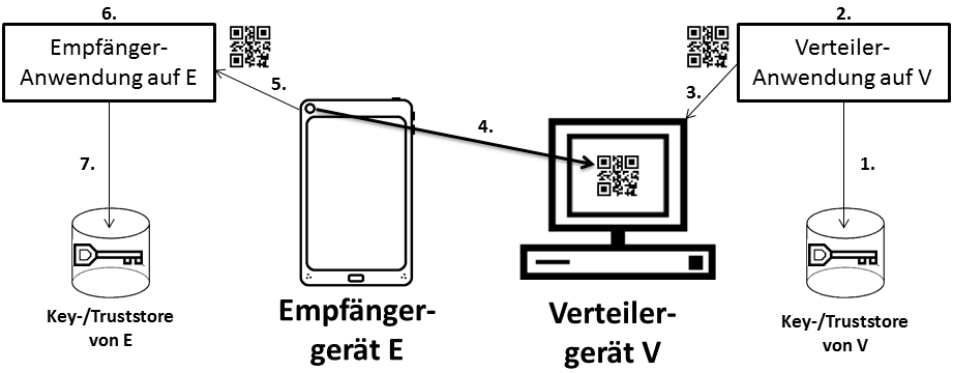

Abbildung 1: Konzept zum Schlüsselaustausch über QR-Codes

oder Veröffentlichungen zu der Thematik gefunden. Die proprietäre Anwendung *Threema* ([TH14]) bietet jedoch eine Möglichkeit öffentliche Schlüssel über QR-Codes zu verifizieren sowie private Schlüssel über solche zu verteilen.

3 Systemarchitektur

Kommunikationspartner müssen zusätzlichen Organisationsaufwand investieren, sofern sie etablierte Methoden zum sicheren Schlüsselaustausch verwenden möchten. Insbesondere gehen solche Protokolle davon aus, dass die Partner vor dem initialen Schlüsselaustausch ein gemeinsames Geheimnis ausgetauscht haben. Für den Austausch ist es daher unvermeidlich, dass alle Kommunikationspartner das identische Geheimnis zur Verfügung haben. Dies ist jedoch kritisch zu sehen, insbesondere wenn zwischen der Geheimnisvereinbarung und dem Schlüsselaustausch ein längerer Zeitraum liegt. Zudem müssen die Kommunikationspartner sicherstellen, dass die gewählten Geheimnisse nicht von Dritten mitgehört, erraten oder gefälscht werden können. Die im Folgenden präsentierte Architektur reduziert diese Probleme, indem der Schlüsselaustausch ohne zeitliche Verzögerung erfolgen kann und kein zusätzlich vereinbartes Geheimnis zwischen den Kommunikationspartnern erforderlich ist.

3.1 Systemkomponenten

Das von uns vorgeschlagene Konzept zur Schlüsselüberführung ist in Abbildung 1 dargestellt. Nachfolgend werden die Komponenten sowie deren Funktionen und Voraussetzungen an die (mobilen) Endgeräte erörtert. Dabei unterscheiden wir die verwendeten (mobilen) Endgeräte in ein Schlüssel verteilendes Gerät V und ein Schlüssel empfangendes Gerät E.

Keystore: Der Keystore ist ein gesicherter Speicherort auf dem Endgerät, der für die Aufbewahrung und Organisation von geheimen beziehungsweise privaten Schlüsseln verwendet wird; beispielsweise in pkcs12-Dateien oder in einer Datenbank. Der Keystore muss eine abgesicherte Schnittstelle besitzen, über die autorisierte dritte Anwendungen auf die hinterlegten privaten Schlüssel zugreifen können. Weiterhin ist sicherzustellen, dass in den Keystore ausschließlich vom Nutzer verifizierte Schlüssel gespeichert werden.

Truststore: Der Truststore ist ein gesicherter Speicherort auf dem Endgerät, der für die Aufbewahrung und Organisation von vertrauenswürdigen öffentlichen Schlüsseln verwendet wird; beispielsweise in crt-Dateien oder in einer Datenbank. Der Truststore muss eine abgesicherte Schnittstelle besitzen, über die autorisierte dritte Anwendungen auf die hinterlegten Schlüssel zugreifen können. Weiterhin ist sicherzustellen, dass in den Truststore ausschließlich vom Nutzer verifizierte öffentliche Schlüssel gespeichert werden.

Verteiler-Anwendung: Die Verteiler-Anwendung muss auf V verfügbar sein. Ihre Aufgabe ist es, aus der Textdarstellung eines Schlüssels den zugehörigen QR-Code zu erzeugen und diesen grafisch sichtbar darzustellen. Hierfür muss V eine grafische Ausgabe besitzen; beispielsweise einen Bildschirm. Zusätzlich kann die Verteiler-Komponente eine Passworteingabe bereitstellen, die es ermöglicht den Schlüssel vor der Überführung in die QR-Code-Darstellung zu verschlüsseln. Ohne Einschränkung gehen wir nachfolgend davon aus, dass zu verteilende Schlüssel bereits auf V gespeichert sind; zum Beispiel in V's Truststore oder Keystore.

Empfänger-Anwendung: Die Empfänger-Anwendung muss auf E verfügbar sein. Ihre Aufgabe ist es, QR-Code-Darstellungen aufzunehmen und aus diesen die kodierten Schlüssel zu berechnen. Hierfür muss E eine Bildaufnahme-Funktion besitzen; beispielsweise eine Kamera. Zusätzlich kann die Empfänger-Anwendung eine Passworteingabe bereitstellen, die es ermöglicht, einen übertragenen verschlüsselten Schlüssel zu entschlüsseln. In Abhängigkeit des übertragenen Schlüssels muss der Nutzer die Möglichkeit haben, den empfangenen Schlüssel entweder in E's Truststore oder in E's Keystore zu überführen.

Eine Aufteilung der Funktionen auf mehrere Anwendungen, die auf den (mobilen) Endgeräten installiert sind, ist empfehlenswert, jedoch nicht zwingend notwendig. Ebenso ist eine Anwendung realisierbar, in welche alle beschriebenen Softwarekomponenten integriert sind. In beiden Fällen kann ein (mobiles) Endgerät sowohl als verteilendes Gerät V als auch als empfangendes Gerät E Nutzung finden.

3.2 Schlüsselverteilung

Das Konzept der Schlüsselübertragung über QR-Codes ist in Abbildung 1 dargestellt. Es ist sowohl für öffentliche, private und symmetrische Schlüssel anwendbar. Möchte ein Nutzer einen auf Gerät V hinterlegten öffentlichen oder privaten Schlüssel an Gerät E

weitergeben, so greift er mit der Verteiler-Anwendung auf Vs Trust- beziehungsweise Keystore zu und wählt den zu verteilenden Schlüssel (1). Zusätzlich kann die Verteiler-Anwendung über eine Passworteingabe verfügen, um den ausgewählten Schlüssel für die Übertragung zu verschlüsseln. Aus dem (verschlüsselten) Schlüssel erstellt die Verteiler-Anwendung die QR-Code-Darstellung (2) und stellt diese auf Vs grafischer Ausgabe dar (3).
E fotografiert mit seiner Bildaufnahme-Funktion den QR-Code von Vs grafischer Ausgabe ab (4) und leitet diesen an die Empfänger-Anwendung weiter (5). Diese ermittelt den im QR-Code kodierten (verschlüsselten) Schlüssel (6). Sofern der Schlüssel verschlüsselt ist, muss Es Nutzer zusätzlich das zuvor verwendete Passwort für die Entschlüsselung des übertragenen Schlüssels eingeben. Dies kann beispielsweise direkt in der Empfänger-Anwendung geschehen. Anschließend speichert diese den übertragenen Schlüssel über die gesicherten Schnittstellen in Es Key- beziehungsweise Truststore (7). Nun kann der Schlüssel auch von Es Anwendungen verwendet werden.

4 Sicherung der gespeicherten Schlüssel

Die vorgestellte Architektur muss sicherstellen, dass ausschließlich berechtigte (dritte) Anwendungen auf die hinterlegten Schlüssel zugreifen können. Folgend werden die hierfür unseres Erachtens relevanten Ansätze präsentiert und diskutiert. Der Begriff *Schlüsselspeicher* steht in diesem Abschnitt stellvertretend sowohl für Trust- als auch Keystore.

4.1 Eigenverantwortliche Schlüsselverwaltung

Jede Anwendung kann neben ihren eigentlichen Funktionen einen eigenen Schlüsselspeicher betreiben. Dementsprechend ist jede Anwendung für die Verwaltung der von ihr benötigten Schlüssel eigenverantwortlich. Dies reduziert die Anzahl der im Schlüsselspeicher hinterlegten Schlüssel im Vergleich zu einem zentralisierten. Hieraus ergibt sich der Vorteil, dass im Falle eines korrumpierten Schlüsselspeichers die Schlüssel anderer Anwendungen nicht betroffen sind. Unserer Meinung nach ist es aus organisatorischer sowie aus datenschutzrechtlicher Sicht empfehlenswert, Schlüssel nur an den Stellen zu hinterlegen, an denen sie benötigt werden. Schlüsselsuche und fehlerhafte Schlüsselwahl kann somit zumindest verringert werden, was die Benutzerfreundlichkeit erhöht.
Aus der dezentralen Schlüsselhaltung ergeben sich jedoch auch diverse Nachteile. Anwendungsentwickler sind hierbei für die Implementierung eigener Verteiler- und Empfänger-Funktionalität verantwortlich oder müssen hinreichend geschützte Schnittstellen mit diesen realisieren. Zudem muss die Anwendung sich selbst um die Schlüsselorganisation kümmern, wie das Einfügen, Abrufen und Entfernen sowie der Schutz von Schlüsseln. Dieser Zusatzaufwand kann für die Entwicklung neuer Dienste und Anwendungen hinderlich sein. Zwar können Anwendungsentwickler hierfür Bibliotheken verwenden, deren Änderungen und Sicherheitsanpassungen jedoch von den Entwicklern kontinuierlich

in die Anwendung integriert werden müssen. Aus Nutzersicht bedeutet dies zusätzlichen Aufwand: Man muss alle Anwendungen auf allen (mobilen) Endgeräten aktualisieren, die diese Bibliotheken verwenden.
Ein aus Nutzersicht weitaus größerer Nachteil am dezentralen Ansatz ist die Verteilung von Schlüsseln, die in mehreren Anwendungen auf demselben (mobilen) Endgerät benötigt werden. In diesem Fall muss der Schlüssel durch mehrere Empfänger-Funktionen auf dasselbe (mobile) Endgerät überführt werden. Mit steigender Anzahl an Anwendungen steigt dabei für den Nutzer der organisatorische Aufwand. Besonders problematisch ist diese Herangehensweise, wenn ein Schlüssel ausgetauscht werden muss; beispielsweise wenn ein privater Schlüssel gestohlen wird. In diesen Fällen ist der Schlüssel zwangsläufig in allen ihn verwendenden Anwendungen auszutauschen. Folglich steigt mit wachsender Anzahl an Anwendungen zum einen der Aufwand für den Nutzer und zum anderen das Risiko, dass der Austausch an einer Stelle vergessen wird.
Das bereits erwähnte Threema ([TH14]) beinhaltet einen eigenen Speicher für die öffentlichen Schlüssel vertrauenswürdiger Personen und speichert den eigenen privaten Schlüssel.

4.2 Zentrale Schlüsselverwaltung

Der entgegengesetzte Ansatz zur dezentralen Schlüsselverwaltung ist, unabhängig von den Schlüssel nutzenden Anwendungen, das Betreiben einzelner zentraler Schlüsselspeicher. Die Extremform hierbei ist ein einzelner zentraler Schlüsselspeicher, der von allen Anwendungen genutzt werden kann. Ein Hauptvorteil aus Nutzersicht liegt im reduzierten Aufwand bei der Überführung von Schlüsseln, die in mehreren Anwendungen verwendet werden. Ebenso ist die Anpassung im Falle eines Schlüsseltausches einfacher: Der Nutzer muss den Schlüssel ausschließlich im zentralen Schlüsselspeicher aktualisieren. Die Folge ist eine Reduzierung des Risikos, dass Nutzer den Schlüsselaustausch in Anwendungen vergessen und in diesen mit dem veralteten oder unsicheren Schlüssel unbemerkt weiterarbeiten.
Aus technischer Sicht ist es vorteilhaft, dass sowohl Verteiler- als auch Empfänger-Funktionen über auf dem (mobilen) Endgerät zentrale Anwendungen bereitgestellt werden können. Diese können dadurch direkt als Teil des Schlüsselspeichers entwickelt und somit die Interaktion zwischen Verteiler-, Empfänger- und Schlüsselspeicher-Funktion gesichert werden. In jedem Fall ist die Aktualisierung der zentralen Verteiler-, Empfänger- oder Schlüsselspeicher-Funktion für den Nutzer weniger aufwendig, wenn sich verwendete Bibliotheken ändern, da die mit den Schlüsseln arbeitenden Anwendungen von diesen unabhängig sind.
Falls unberechtigte Dritte Zugriff auf einen zentralen Schlüsselspeicher erhalten, so sind alle dort gespeicherten Schlüssel als gestohlen anzusehen. Dies ist im Vergleich zur Aufteilung der Schlüssel auf mehrere unabhängige Schlüsselspeicher nachteilig. Hierbei handelt es sich um einen problematischen Gesichtspunkt, da alle Daten der mit dem Schlüsselspeicher interagierenden Anwendungen als gestohlen, manipuliert oder unsicher betrachtet werden müssen. Dadurch ist ein erheblicher Aufwand für den Nutzer unvermeidbar, um die Integrität der Daten zu prüfen, neue Schlüssel für die Anwendungen zu erstellen sowie

zu verteilen und die Daten unverzüglich neu zu verschlüsseln. Insbesondere wenn Dritte Zugriff auf Schlüssel aus kollaborativen Projekten bekommen haben, sind nicht nur die Daten des angegriffenen (mobilen) Endgeräts verloren, sondern auch die der Projektpartner. Dies muss an alle Projektpartner kommuniziert werden und es ist eine in der Gruppe durchführbare Lösung für die zuvor genannten Aufgaben zu finden.
Im zentralisierten Ansatz muss berücksichtigt werden, dass dritte Anwendungen mit dem Schlüsselspeicher interagieren können müssen. Erfahrungsgemäß ist es sinnvoll, standardisierte Schnittstellen für die Anwendungskommunikation zu verwenden. Der vom Schlüsselspeicher verwendete Standard ist dabei möglichst nicht zu verändern, da Änderungen an der Schnittstelle Änderungen an den zugreifenden Anwendungen nach sich ziehen. Dies verursacht wiederum Aktualisierungsaufwand für den Nutzer. Da die Schlüssel als kritische Elemente angesehen werden, muss der Schlüsselspeicher seine Schnittstellen vor unautorisiertem Zugriff schützen; zum Beispiel über eine Passwortabfrage.

4.2.1 Betriebssystemintegration

Eine mögliche Implementierungsart der zentralen Schlüsselverwaltung ist die Integration in das Betriebssystem des (mobilen) Endgeräts. Dies ermöglicht die Erstellung einer optimierten Gesamtanwendung aus Verteiler, Empfänger und Schlüsselspeicher und sie kann direkt in das Betriebssystem integriert werden. Nutzer müssen somit keiner dritten Anwendung für die Aufbewahrung ihrer Schlüssel vertrauen. Das Betriebssystem des (mobilen) Endgeräts ist zwangsläufig als vertrauenswürdig einzustufen, andernfalls dürften die Daten nie unverschlüsselt auf diesem vorliegen. Zudem ermöglicht die Integration, den Zugriff auf die Schlüssel vom Betriebssystem schützen zu lassen. Beispielsweise ist es denkbar, den Zugriff auf den Schlüsselspeicher auf Anwendungen zu beschränken, die vom Betriebssystemhersteller zertifiziert sind.
Nachteilig an dieser Art der Integration ist die Notwendigkeit, dass alle Anwendungen die mit den Schlüsseln arbeiten, mit den Betriebssystemschnittstellen interagieren können müssen. Daraus resultiert, dass die Anwendungen nicht betriebssystemübergreifend entwickelt werden können.

4.2.2 Externe Schlüsselspeicher

Zentrale Schlüsselspeicher können auch von Drittanbietern entwickelt und auf das Betriebssystem des (mobilen) Endgeräts installiert werden. Dies erlaubt es, mehrere zentrale Schlüsselspeicher auf einem Endgerät bereitzustellen, die für das zu verwaltende Schlüsselmaterial optimiert sind. Die Aufteilung der Schlüssel auf mehrere zentrale Schlüsselspeicher reduziert zumindest die Problematik eines erfolgreich angegriffenen Speicherorts.
Schlüsselspeicher können von externen Entwicklern auch in betriebssystemübergreifenden Technologien entwickelt werden. Dies erlaubt es Nutzern ihnen bekannte Anwendungen auf mehreren (mobilen) Endgeräten zu verwenden, auch wenn diese unterschiedliche Be-

triebssysteme haben.
Der Ansatz ist hinsichtlich des Zugriffs auf den Schlüsselspeicher nachteilig, da jeder Schlüsselspeicherentwickler seinen eigenen Schnittstellenstandard definieren kann. Daraus entsteht eine starke Kopplung zwischen den Schlüssel nutzenden Anwendungen und den Schlüsselspeichern. Weiterhin muss der Nutzer dem Entwickler des Schlüsselspeichers dahingehend vertrauen, dass er die aufbewahrten Schlüssel nicht weitergibt oder über „Hintertüren“ ausliest. Auch die eigentlich vorteilhafte Verwendung von betriebssystemübergreifenden Technologien ist kritisch zu betrachten, da nicht sichergestellt ist, ob ein Endgerät benötigte Grundlagen bereitstellt.

4.3 Aktualisierende Schlüsselverwaltung

Die Trennung der Verteiler- und Empfänger-Funktionalität vom Schlüsselspeicher ermöglicht eine aktualisierende Schlüsselverwaltung. Wie bei der dezentralen Schlüsselverwaltung kann jede Anwendung ihren eigenen Schlüsselspeicher betreiben, der über Schnittstellen der Anwendung zugreifbar ist. Dies erlaubt es, einer zentralen Empfänger-Anwendung in einem Schritt mehrere dezentrale Schlüsselspeicher zu aktualisieren; beispielsweise die Überführung eines benötigten Schlüssels in mehrere Anwendungen. Ebenso kann eine zentrale Verteiler-Anwendung die zu verteilenden Schlüssel aus mehreren dezentralen Schlüsselspeichern beziehen. Durch die zentrale Implementierung der Empfänger- und Verteiler-Anwendung auf dem (mobilen) Endgerät müssen neue Versionen ausschließlich an einer Stelle eingepflegt werden. Gleichzeitig können die Anwendungshersteller eigene Schlüsselspeicher implementieren und für ihre Anwendungsfälle optimieren.
Bei aktualisierender Schlüsselverwaltung sind standardisierte Schnittstellen zwischen Verteiler- beziehungsweise Empfänger-Funktionalität und den Schlüsselspeicher-Anwendungen erforderlich. Zudem müssen die Schnittstellen vor unautorisiertem Zugriff geschützt werden, was für den Nutzer Aufwand bedeutet, beispielsweise indem er für jede Schlüsselspeicher-Anwendung ein Zugriffspasswort vergibt.

5 Diskussion des Konzepts

5.1 Aufwand

Aufgrund der direkten Interaktion zwischen den (mobilen) Endgeräten benötigen die austauschenden Partner kein zusätzliches Geheimnis und keinen unsicheren Kanal wie E-Mail-Verkehr für den Schlüsselaustausch. Für den vertraulichen Austausch von öffentlichen Schlüsseln werden zudem keine digitalen Zertifikate und Public-Key-Infrastrukturen benötigt. Letztere können jedoch zusätzlich für die Gültigkeitsprüfung der öffentlichen Schlüssel genutzt werden.
Für den Verteilenden beschränkt sich der Aufwand darauf, auf seinem (mobilen) Endgerät

die Verteiler-Anwendung zu starten und den Schlüssel auszuwählen. Der Empfänger muss ausschließlich mit der Empfänger-Anwendung auf seinem (mobilen) Gerät das Display des Verteilenden-Geräts fotografieren. Hierbei sind die Positionsinformationen im QR-Code hilfreich, da die Position des Empfänger-Geräts somit weitestgehend vernachlässigbar ist. Zur eindeutigen Zuordnung zwischen Schlüssel und Verwendungszweck beziehungsweise Schlüssel und Person ist es empfehlenswert, dass der Empfänger Metadaten über den Schlüssel in sein Gerät einträgt.
Die direkte Interaktion ist auch der Hauptnachteil des Verfahrens. Um Schlüssel austauschen zu können, müssen die Kommunikationspartner eine Möglichkeit haben die Display- und Fotofunktionen ihrer (mobilen) Endgeräte einzusetzen; beispielsweise indem einander persönlich treffen. Unserer Meinung nach ist dies für das Verfahren jedoch nicht nachteilig, da die Möglichkeit im persönlichen Gespräch ad hoc Schlüssel austauschen zu können deutlich vorteilhafter ist, als zeitverzögert aufwendigere/weniger sichere Verfahren zu nutzen. Es ist zudem aus technischer Sicht zu hinterfragen, ob QR-Codes wie in [ISO00] spezifiziert über genügend Kapazität für das zu übertragende Schlüsselmaterial verfügen. Insbesondere asymmetrische Verschlüsselungsverfahren benötigen möglichst große Schlüssel, um ausreichend viele und hinreichend sichere Schlüssel bereitzustellen (siehe hierzu auch [LV99]). Alternative Darstellungsformen, die über ausreichend Kapazität verfügen oder eine Aufteilung des Schlüssels auf mehrere QR-Codes sind hierbei von Nutzen. Alternativen zum QR-Code sind hierfür beispielsweise verknüpfbare DataMatrix-Codes wie in [ISO06] spezifiziert.

5.2 Sicherheit

Durch die direkte Interaktion zwischen den (mobilen) Endgeräten ist der Schlüsselaustausch als sicher anzusehen. Fehler während der Übertragung können von der Empfänger-Anwendung anhand der QR-Code-Prüfsummen entdeckt werden. Damit erhält der Empfänger direkt Rückmeldung über den Erfolg und der Austausch kann gegebenenfalls wiederholt werden. Für geheime und private Schlüssel besteht die Möglichkeit diese vor ihrer Darstellung als QR-Code zusätzlich über Passwörter zu schützen. Kann die QR-Code-Darstellung nicht hinreichend geschützt werden, so können unberechtigte Dritte den geheimen Schlüssel nicht aus dem QR-Code ermitteln.
Die Implementierungen von Trust- und Keystores definiert die Sicherheit der gespeicherten Schlüssel auf den (mobilen) Endgeräten. Die Aufbewahrung von Schlüsseln ist jedoch eine Grundsatzfrage für Datenschutzsysteme. In Kapitel 4 haben wir diesbezüglich Vor- und Nachteile von Implementierungen diskutiert.

5.3 Anwendungsgebiete

Unser Ansatz erlaubt es, einer Person ohne zusätzlichen Aufwand ihre geheimen und privaten Schlüssel zwischen ihren (mobilen) Endgeräten zu verteilen. Diese Verteilung erfolgt dabei gesichert und weitestgehend automatisiert. Die Person muss nicht zeichenwei-

se Passwörter von einem Gerät in ein anderes übertragen, sondern es bedarf ausschließlich einer Fotografie des QR-Codes. Zudem ist dieser Übertragungsweg sicherer als der Versand von Passwörtern beispielsweise in E-Mails.
Ebenfalls vereinfacht der Einsatz den Austausch von Schlüsseln zwischen unterschiedlichen Kommunikationspartnern. Über zentral bereitgestellte QR-Code-Darstellungen können Gruppen einfach ihre Schlüssel untereinander verteilen. Beispielsweise kann ein Vortragender seinen öffentlichen Schlüssel auf sicherem Weg an seine Zuhörer verteilen, indem er sie in seine Präsentation einfügt. Dies erlaubt die vertrauenswürdige Übertragung von öffentlichen Schlüsseln ohne Public-Key-Infrastruktur.
Ähnlich zum Web-of-Trust-Modell kann ein (öffentlicher) Schlüssel über die transitive Vertrauensrelation ausgetauscht werden. Vertraut Person B einer Person A, so überführt sie den Schlüssel von A's Endgerät auf das von B. Benötigt Person C A's Schlüssel und Vertraut C Person B, kann der Schlüssel ebenfalls einfach von B's Endgerät auf C's übertragen werden. Diese Herangehensweise ist insbesondere dann von Interesse, wenn A und C sich nicht persönlich treffen können.
Ein weiteres Beispiel ist die Verteilung von Schlüsseln innerhalb einer Gruppe über Printmedien. Im Vorfeld eines Gruppentreffens kann ein administratives Gruppenmitglied die von der Gruppe zu verwendenden Schlüssel erstellen und in ihre QR-Code-Darstellung überführen. Diese können mit Zusatzinformationen ausgedruckt und beim Gruppentreffen verteilt werden. Während des Treffens haben alle Gruppenmitglieder die Möglichkeit, die geheimen Schlüssel auf ihre Endgeräte zu überführen und nach Abschluss des persönlichen Treffens werden die Ausdrucke vernichtet. Die Erstellung des Printmediums erfordert anfänglich zwar Zusatzaufwand, die sichere Überführung auf die (mobilen) Endgeräte der Teilnehmer ist jedoch benutzerfreundlich und sicher.

6 Zusammenfassung und Ausblick

Integrität, Vertraulichkeit und Identifikation werden im Webumfeld auf Basis von geheimen Schlüsseln sichergestellt. Dabei müssen gewählte Schlüssel einerseits möglichst sicher gewählt sein, andererseits werden sie auf allen sie verwendenden (mobilen) Endgeräten des Nutzers benötigt. Mit steigender Schlüsselgröße und -komplexität sowie wachsender Anzahl an Endgeräten, die mit den Schlüsseln arbeiten, steigt der Aufwand für die Nutzer ihre Schlüssel zu verteilen. Das im vorliegenden Dokument vorgestellte Konzept erlaubt es, Schlüssel zwischen (mobilen) Endgeräten unter Verwendung von QR-Codes auszutauschen. Wir haben dabei in Kapitel 3 die benötigten funktionellen Einheiten und in Kapitel 4 potentielle Schlüsselspeicher-Architekturen diskutiert. Abschließend haben wir das vorgestellte Konzept hinsichtlich Aufwand für die Anwender und Sicherheit für das Schlüsselmaterial betrachtet und ausgewählte Anwendungsfälle vorgestellt. Die Zielstellung des vorgestellten Konzepts ist hierbei in erster Linie der einfache, ad hoc mögliche und vertrauliche Schlüsselaustausch bei persönlichen Treffen von Personen und die Möglichkeit für einen Nutzer seine geheimen Schlüssel möglichst unkompliziert zwischen seinen (mobilen) Endgeräten zu verteilen.
Basierend auf dem vorgestellten Konzept entwickelt unsere Arbeitsgruppe derzeit proto-

typisch eine Anwendung. Sobald die Implementierung abgeschlossen ist, werden wir eine Evaluation über eine große Bandbreite an Arbeitsgruppen durchführen, um damit die von uns angenommene Nutzerfreundlichkeit zu überprüfen. Hierfür sind noch Anwendungen zu erstellen, die auf Trust- und Keystore unseres Protoypen zugreifen; beispielsweise mobile E-Mail-Clients oder Anwendungen, die mit Daten auf Cloud-Strukturen interagieren.

Literatur

[AB13] Ericsson AB. Ericsson Mobil Report: On The Pulse Of The Networked Society, November 2013.

[AK08] Hend S. Al-Khalifa. Utilizing QR Code and Mobile Phones for Blinds and Visually Impaired People. In *Computers Helping People with Special Needs*, Seiten 1065–1069. Springer Berlin Heidelberg, 2008.

[BGB04] Nikita Borisov, Ian Goldberg und Eric Brewer. Off-the-record Communication, or, Why Not to Use PGP. In *Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society*, Seiten 77–84. ACM, 2004.

[DH76] W. Diffie und M. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[DT14] The ZXing Developer-Team. Zxing, open source library to read 1D/2D barcodes. Webseite: "https://github.com/zxing/zxing/"[Zuletzt Aufgerufen: Mai 2014].

[ISO00] International Organization for Standardization. Information Technology - Automatic Identification and Data Capture Techniques - Bar code symbology - QR Code. ISO/IEC 18004:2000, 2000.

[ISO06] International Organization for Standardization. Information technology - Automatic identification and data capture techniques - Data Matrix bar code symbology specification. ISO/IEC 16022:2006(E), 2006.

[TH14] Threema GmbH. Threema - Seriously secure mobile messaging. Webseite: "https://threema.ch/de/"[Zuletzt Aufgerufen: Mai 2014].

[DW14] DENSO WAVE INCORPORATED. History of QR Code. Webseite: "http://www.qrcode.com/en/history/"[Zuletzt Aufgerufen: Mai 2014].

[K+10] Peter Kieseberg et al. QR Code Security. In *Proceedings of the 8th International Conference on Advances in Mobile Computing and Multimedia*, MoMM '10, Seiten 430–435. ACM, 2010.

[Lam04] Butler W. Lampson. Computer Security in the Real World. *Computer*, 37(6):37–46, 2004.

[LV99] Arjen K. Lenstra und Eric R. Verheul. Selecting Cryptographic Key Sizes. *Journal of Cryptology*, 14:255–293, 1999.

[M+99] M. Myers et al. RFC2560 X.509 Internet Public Key Infrastructure, Online Certificate Status Protocol, 1999.

[WT98] Alma Whitten und J. D. Tygar. Usability of Security: A Case Study. In *CMU-CS-98-155*, December 1998.