Technische Universität München
Fakultät für Informatik

Master's Thesis in Informatik

Entwurf und Realisierung eines Sicheren Verzeichnisdienstes für Digitale Zertifikate

Daniel Straub

Technische Universität München

Fakultät für Informatik

Master's Thesis in Informatik

Entwurf und Realisierung eines Sicheren Verzeichnisdienstes für Digitale Zertifikate

Design and Implementation of a Secure Directory Service for Digital Certificates

Autor	Daniel Straub
Aufgabensteller	Prof. Dr.-Ing. Georg Carle
Betreuer	Dr. Matthias Wachs
Datum	27. April 2016

Informatik VIII
Lehrstuhl für Netzarchitekturen und Netzdienste

Ich versichere, dass ich die vorliegende Arbeit selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Garching b. München, 27. April 2016

Unterschrift

Zusammenfassung

Sichere Kommunikation spielt in der heutigen Welt eine wichtige Rolle. So gibt es auch für sichere E-Mail-Kommunikation zwei verschiedene Verfahren, OpenPGP und S/MIME. Die korrekte Verwendung dieser Verfahren erfordert vom Benutzer einen vergleichweise hohen Aufwand und setzt technisches Wissen voraus. Verantwortlich hierfür ist hauptsächlich der in beiden Verfahren erforderliche Austausch von Zertifikaten, der häufig über Verzeichnisdienste stattfindet. Damit verbunden sind zwei wichtige Probleme: die Authentizität eines Zertifikats und der Schutz der Privatsphäre des Zertifikateigentümers. Die Authentizität des Zertifikats ist erforderlich, um eine sichere Kommunikation zu gewährleisten. Die Veröffentlichung eines Zertifikats über einen Verzeichnisdienst macht das Zertifikat und alle damit verbundenen Informationen einer großen Gruppe von Personen zugänglich. Dadurch wird die Privatsphäre des Betroffenen verletzt. Um die Verwendung sicherer E-Mail-Kommunikation zu erleichtern, sollte ein Verzeichnisdienst die Authentizität von Zertifikaten und die Privatsphäre der Benutzer garantieren.

Aus diesem Grund ist das Ziel dieser Arbeit das Sicherstellen der Authentizität von Zertifikaten und der Schutz der Privatsphäre von Zertifkateigentümern, bei über Verzeichnisdienste veröffentlichte Zertifikate. Hierfür werden aktuelle Probleme bei der Verwendung von Verzeichnisdiensten untersucht, um Anforderungen an einen Verzeichnisdienst zu finden, der die erkannten Probleme löst. Das Ergebnis der Arbeit ist ein Entwurf und die anschließende Implementierung eines Verzeichnisdienstes, der die Authentizität von Zertifikaten sicherstellt und den Benutzern den Schutz ihrer Privatsphäre ermöglicht.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BC	Bouncy Castle
CA	Certificate Authority
CRL	Certificate Revocation List
DANE	DNS-Based Authentication of Named Entities
DNS	Domain Name System
GNS	GNU Name System
HKP	HTTP Keyserver Protocol
JPA	Java Persistence API
LDAP	Lightweight Directory Access Protocol
MIME	Multipurpose Internet Mail Extensions
PKI	Public Key Infrastructure
PKS	Public Key Server
WoT	Web of Trust

Kapitel 1

Einleitung

Immer mehr Leute beschäftigen sich mit dem Thema Sicherheit und Vertraulichkeit von Kommunikation, insbesondere seit der Enthüllung der Überwachungsprogramme verschiedener Geheimdienste [1]. Für die sichere E-Mail-Kommunikation im speziellen gibt es zwei Standards: *OpenPGP* und *S/MIME*. Die Verwendung dieser Standards zur sicheren Kommunikation, erfordert ein gewisses Maß an Aufwand und Know-how [2]. Aus diesem Grund verwendet der Durchschnittsnutzer, trotz seiner Sorge um Sicherheit und Privatsphäre, keine sichere Kommunikation.

Sowohl OpenPGP als auch S/MIME basieren auf Zertifikaten und asymmetrischer Kryptographie. Kommunikationspartner müssen Zertifikate austauschen, die das öffentliche Schlüsselmaterial enthalten. Erst das öffentliche Schlüsselmaterial ermöglicht eine sichere Kommunikation mit dem Eigentümer des zugehörigen privaten Schlüssels. Der Austausch von Zertifikaten, welche das öffentliche Schlüsselmaterial beinhalten, bereitet allerdings neue Probleme: Wenn der Austausch des Zertifikats nicht persönlich durchgeführt werden kann, dann müssen andere Wege für den Austausch gefunden werden. Deshalb werden Zertifikate in der Regel über Verzeichnisdienste veröffentlicht, wovon sie von anderen Benutzern abgerufen werden können. Sogenannte Public Key Server (PKS) erfüllen diese Aufgabe bei OpenPGP. Bei S/MIME werden dagegen X.500 Verzeichnisdienste, wie zum Beispiel LDAP Server verwendet.

Bei beiden Verfahren ist es ein Problem die *Authentizität* eines Zertifikats sicherzustellen, also die Tatsache, dass ein Zertifikat einer bestimmten Person gehört. Das betrifft insbesondere die PKS. OpenPGP und S/MIME gehen zwei verschiedene Wege, um diese Authentizität zu gewährleisten: Sogenannte Zertifizierungsstellen (engl. Certificate Authority (CA)) garantieren die Authentizität eines S/MIME Zertifikats, indem sie Zertifikate für Benutzer ausstellen und deren Identität dabei prüfen [3]. Das Vertrauen in ein Zertifikat bedeutet somit das Vertrauen in die ausstellende CA. OpenPGP löst das Problem der Authentizität auf anderem Weg: Anstatt durch zentrale CAs wird die Authentizität durch das sogenannte Web of Trust (WoT) sichergestellt [4]. In beiden Fällen

ist entweder das Vertrauen in andere Personen erforderlich oder eigener Aufwand zur Sicherstellung der Authentizität notwendig.

Ein weiteres Problem sind die in Zertifikaten enthaltenen persönlichen Informationen, wie beispielsweise Name oder Adresse der Eigentümer. Diese Daten sind nach der Veröffentlichung für jeden einsehbar und können vom Besitzer des Zertifikats auch nicht wieder zurückgenommen werden. OpenPGP Zertifikate ermöglichen durch die enthaltenen Signaturen zusätzlich die Erstellung eines sozialen Graphen. Das bedeutet einen starken Eingriff in die Privatsphäre.

1.1 Aufgabenstellung und Zielsetzung

Die vorliegende Arbeit untersucht aktuelle Probleme bezüglich Authentizität und Privatsphäre beim Austausch von Zertifikaten über Verzeichnisdienste.

Das Ziel der Arbeit ist die Realisierung eines Zertifikatsverzeichnisdienstes, welcher die Authentizität von Zertifikaten sicherstellt und die Privatsphäre der Beteiligten im Vergleich zu aktuellen Lösungen erhöht.

Zur Erreichung des Ziels ist die Beantwortung folgender Fragen erforderlich:

- Welche aktuellen Probleme und Ansätze gibt es, bezüglich der Authentizität von Zertifikaten und der Privatsphäre Beteiligter, beim Austausch von Zertifikaten über Verzeichnisdienste?
- Was sind die notwendigen Anforderungen an einen sicheren und die Privatsphäre schützenden Zertifikatsverzeichnisdienst?
- Wie muss ein Entwurf für einen Zertifikatsverzeichnisdienst aussehen, welcher die erarbeiteten Anforderungen erfüllt?
- Auf welche Art kann der erstellte Entwurf umgesetzt werden?

Das Ergebnis dieser Arbeit wird ein Systementwurf, mit dessen Hilfe die beschriebenen Probleme hinsichtlich der Privatsphäre und Authentizität gelöst werden können. Eine prototypische Implementierung basierend auf dem Enwurf soll die Umsetzbarkeit zeigen. Mit Hilfe der gewonnenen Erkenntnisse in dieser Arbeit soll die Verwendung von sicherer E-Mail-Kommunikation vereinfacht und verbessert werden, indem die Authentizität von Zertifikaten sichgerstellt und die Privatsphäre der Benutzer von Verzeichnisdiensten geschützt wird.

1.2 Gliederung der Arbeit

In Kapitel 2 werden zunächst die für diese Arbeit erforderlichen Grundlagen beschrie-

ben. Im Anschluss werden in Kapitel 3 aktuelle Probleme beim Austausch von S/MIME und OpenPGP Zertifikaten über Verzeichnisdienste hinsichtlich der Authentizität und Privatsphäre untersucht. Hierfür werden aktuell verwendete Verzeichnisdienste und verwandte Ansätze betrachtet, um ein tieferes Verständnis für die Probleme zu erhalten. Darauf aufbauend erarbeiet Kapitel 4 die Anforderungen für einen Zertifikatsverzeichnisdienst, um die untersuchten Probleme zu lösen. Anhand der erarbeiteten Anforderungen wird anschließend in Kapitel 5 ein geeigneter Systementwurf erstellt, der als Grundlage für die in Kapitel 6 beschriebene Implementierung dient. Abschließend wird die gefundene Lösung in Kapitel 7 evaluiert und deren Vor- und Nachteile untersucht.

Kapitel 2

Grundlagen

In diesem Kapitel werden die Grundlagen für sichere E-Mail-Kommunikation erläutert. Diese sind für das Verständnis des in dieser Arbeit behandelten Themas relevant.

2.1 E-Mail-Architektur

Die Kommunikation über E-Mail basiert auf den E-Mail-Adressen, die als Identifikation innerhalb des E-Mail-Systems dienen. Eine E-Mail-Adresse besteht aus zwei Teilen, aus dem lokalen Teil und der Domäne, die durch das Zeichen @ getrennt werden: lokal@domäne. Wird eine Nachricht an eine E-Mail-Adresse geschickt, wird diese über E-Mail-Server an den für die in der Adresse genannte Domäne verantwortlichen Server weitergereicht. Dort angelangt kann der Empfänger die Nachricht herunterladen und lesen. Das E-Mail-System ist dadurch ein offenes System, in dem jeder der im Besitz einer Domäne ist einen eigenen E-Mail-Server integrieren kann.

2.2 Struktur von E-Mail-Nachrichten und MIME

Die Struktur einer E-Mail-Nachricht ist in RFC 5322 [5] definiert und gliedert sich grundsätzlich in die zwei Teile *Header* und *Body*. Im Header sind verschiedene Metadaten enthalten, die Informationen über die Nachricht enthalten. Dafür existieren verschiedene Header-Felder, die am Beginn einer Nachricht stehen. Jede Zeile enthält ein Header-Feld im folgenden Format:

```
Name: Wert
```

Beispiele für solche Header sind *From* (die Absenderadresse), *Date* (der Zeitpunkt des Versands) oder *Subject* (der Betreff der Nachricht).

Im Anschluss folgt der *Body*, welcher die eigentliche Nachricht enthält. Ursprünglich konnte dieser Teil nur ASCII Text enthalten. Die Multipurpose Internet Mail Extensions (MIME) sind eine Erweiterung des RFC 2822 und ermöglichen unter anderem das Senden von unterschiedlichem Inhalt über das Medium E-Mail. Hierfür führt der Standard das *content-type*-Feld ein, über welches die Art des Inhalts einer E-Mail beschrieben wird. Zusätzlich kann eine E-Mail aus mehreren Teilen (MIME-Parts) bestehen, deren Inhalt jeweils über das *content-type*-Feld beschrieben wird. Der Inhalt eines solchen MIME-Teils kann zum Beispiel normalen Text (content-type text/plain), HTML (text/html) oder ein Bild (image/jpeg) enthalten. Ein MIME-Teil kann wiederum weitere MIME-Teile enthalten.

MIME ist die Grundlage für die Erweiterung von E-Mail-Kommunikation mit Protokollen zur sicheren E-Mail-Kommunikation.

2.3 Sichere E-Mail

Während die Kommunikation zwischen E-Mail Servern heute in der Regel verschlüsselt ist, trifft dies für die Nachricht selbst nicht zu. Das bedeutet, dass E-Mails nicht mitgelesen werden können, während sie auf dem Transportweg sind. Im Gegensatz dazu ist einem Dritten das Lesen der Nachricht aber sehr wohl möglich, wenn er Zugriff auf einen der am Transportweg beteiligten E-Mail-Server hat. Dort liegen die E-Mails unverschlüsselt vor, und sind somit nicht vor dem Zugriff Dritter geschützt. Noch gravierender ist: anstatt den E-Mail-Verkehr nur mitzulesen, ist es ebenso möglich Nachrichten zu blockieren oder sogar den Inhalt zu verändern.

Um diese Probleme anzugehen und die E-Mail-Kommunikation sicherer zu gestalten, wurden die beiden Protokolle S/MIME und OpenPGP entwickelt. S/MIME integriert sich in die MIME und für OpenPGP gibt es ebenfalls eine Standardisierung zur Integration in die MIME (PGP/MIME, siehe [6]). Mit ihnen ist es möglich das Mitlesen von Nachrichten zu verhindern und zusätzlich eine Abänderung einer Nachricht zu erkennen.

Sichere Kommunikation hat verschiedene Aspekte der Datensicherheit, welche von ihr erfüllt werden können (vgl. [7]):

Vertraulichkeit ist gewährleistet, wenn der Inhalt der Kommunikation nur von berechtigten Personen einsehbar ist.

Integrität der Kommunikation besagt, dass einmal abgeschickte Informationen nicht mehr nachträglich verändert werden können.

Authentizität der Kommunikation ist sichergestellt, wenn die behauptete Identitäten der Kommunikationspartner gleich der tatsächlichen Identitäten ist.

Verbindlichkeit der Kommunikation bedeutet, dass eine Aussage vom Urheber nicht abgestritten werden kann.

In den folgenden Abschnitten wird ein kurzer Überblick über die beiden Protokolle gegeben.

2.3.1 OpenPGP

OpenPGP basiert auf der proprietären Software PGP, welche 1991 von P. Zimmermann entwickelt wurde, um den Menschen die Möglichkeit zu geben ihre Privatsphäre zu schützen. [8] Zum Schutz von sensiblen E-Mails verwendet OpenPGP ein asymmetrisches Verschlüsselungsverfahren. Der öffentliche Schlüssel ist dabei im sogenannten *Public Key* enthalten, welcher außerdem weitere Informationen über den Kommunikationspartner enthält. Darunter sind unter anderem die sogenannten User IDs, welche aus einer Kombination aus Name und Emailadresse bestehen. Mit Hilfe des öffentlichen Schlüssels ist es möglich, dem Eigentümer des zugehörige privaten Schlüssels verschlüsselte Nachrichten zu schicken. Weiterhin sieht OpenPGP vor mit Hilfe des öffentlichen Schlüssels Signaturen prüfen zu können. Damit garantiert OpenPGP neben der Vertraulichkeit auch Authentizität, Integrität sowie Verbindlichkeit von E-Mails. [9]

OpenPGP sieht vor, dass in Public Keys enthaltene Informationen durch Signaturen bestätigt werden. Ist eine Information mit einer Signatur versehen, so bestätigt der Aussteller der Signatur, dass die Information korrekt ist. Generell kann jeder eine Information eines Public Keys signieren, um die Authentizität der Information zu bestätigen. Ist die persönliche Überprüfung der Authentizität eines Public Keys nicht möglich, können vorhandene Signaturen der Feststellung der Authenizität dienen: Gibt es eine Signatur von einer Person, deren Public Key bereits als authentisch verifiziert wurde und der man vertraut, so kann über deren Signatur eine Vertrauenskette gebildet werden. Je kürzer die Kette und je mehr Ketten vorhanden sind, umso höher ist auch die Wahrscheinlichkeit, dass das Schlüsselmaterial authentisch ist. Weil sich viele solcher Vertrauensketten finden lassen und es sich bei OpenPGP um kein hierarchisches Vertrauensmodell handelt, enstand der Name WoT. [4] [8]

2.3.1.1 OpenPGP Public Key

Der sogenannte OpenPGP Public Key oder auch OpenPGP Zertifikat enthält Informationen, welche es ermöglichen dem Eigentümer verschlüsselte Nachrichten zu senden. Im folgenden wird der Aufbau eines solchen OpenPGP Zertifikats beschrieben [9].

Ein OpenPGP Zertifikat besteht aus einer Ansammlung an Paketen. Jedes dieser Pakete enthält unterschiedliche Informationen über das Zertifikat.

Ein OpenPGP Zertifikat ist in Version 4 wie folgt aufgebaut:

Public-Key Packet Dieses Paket enthält Informationen über den öffentlichen Schlüssel. Darunter sind die Version, das Erstellungsdatum, der zugrundeliegende Verschlüsselungsalgorithmus und das Schlüsselmaterial selbst.

Revocation Signature Im Anschluss an das *Public-Key Packet* sind beliebig viele Rückrufsignaturen zu finden, welche das Schlüsselmaterial ungültig machen.

User ID Packet Es folgen ein oder mehrere *User ID Packets*, die jeweils eine Benutzerkennung im Format name-addr aus RFC 2822 enthält.

Signature Packet Auf jedes *User ID Packet* folgt eine beliebige Anzahl an Signaturen. Der jeweilige Unterzeichner bestätigt mit einer Signatur, dass voranstehende Benutzerkennung die Person identifiziert, welche den an erster Stelle stehenden öffentlichen Schlüssel besitzt. Eine solche Signatur wird berechnet über den zugehörigen öffentlichen Schlüssel und dem Inhalt der Benutzerkennung.

User Attribute Packet Auf die *User ID Packets* folgt eine beliebige Anzahl dieser Pakete. Sie enthalten ebenfalls Informationen über den Eigentümer des Zertifikats, können aber beliebige Daten, wie beispielsweise Adresse oder Bild enthalten.

Attribute Signature Packet Wie bei den *User ID Packets* wird auch jedes *User Attribute Packet* von einer beliebigen Anzahl an Signaturen gefolgt.

Subkey Packet Im Anschluss an die persönlichen Informationen kann eine beliebige Anzahl an Subschlüsseln stehen. Diese können unterschiedliche Verwendungszwecke haben.

Subkey Signature Packet Jedem Subschlüssel folgt eine Signatur, welche die Zugehörigkeit zum öffentlichen Schlüssel bestätigt.

Revocation Signature Optional folgt jedem Subschlüssel eine Rückrufsignatur, welche die Ungültigkeit des Subschlüssels festhält.

2.3.2 S/MIME (X.509)

S/MIME verwendet wie OpenPGP ein asymmetrisches Verschlüsselungsverfahren und verwendet hierfür X.509 Zertifikate. Diese sind vergleichbar mit den Public Keys aus OpenPGP, mit dem Unterschied im Aufbau des Formats.

Insgesamt bietet S/MIME die gleiche Funktionalität wie OpenPGP: Vertraulichkeit, Authentizität, Integrität und Verbindlichkeit von E-Mails. [10]

Die Authentizität eines X.509 Zertifikats wird vom Aussteller, der sogenannten CA, überprüft und durch eine Signatur bestätigt. Somit fällt im Gegensatz zu OpenPGP die Verantwortung für die Prüfung der Authentizität nicht dem Anwender zu. Ausgewählte CAs erhalten den Sonderstatus einer Root-CA. Deren Zertifikate, auch Root-Zertifikate

genannt, werden als vertrauensvoll eingestuft und sind im System hinterlegt. Über die Signatur eines Zertifikats wird versucht eine Kette bis zu einem Root-Zertifikat zu bilden. Kann eine solche Kette gebildet werden, wird die Authentizität des Zertifikats über die Kette der Signaturen überprüft. [11] [12]

2.3.2.1 S/MIME Zertifikat

Im folgenden wird der für diese Arbeit relevante Teil des Aufbaus eines S/MIME Zertifikats beschrieben, wie er in RFC 5280 [3] definiert ist. Die beschriebene Struktur ist auch in Abbildung 2.1 zu sehen.

Generell besteht ein S/MIME Zertifikat aus drei Teilen:

Zertifikat Dieser Teil enthält alle Informationen bezüglich des Zertifikats.

Zertifikatsunterzeichnungs-Algorithmus Dieses Feld definiert den für die Signatur verwendeten Algorithmus.

Signaturwert des Zertifikats Dieses Feld enthält die Signatur der CA, mit der sie die Gültigkeit der im Zertifikat enthaltenen Informationen bestätigt. Die Signatur wird basierend auf dem kompletten ersten Teil (*tbsCertificate*) berechnet.

Der signierte Teil des Zertifikats enthält verschiedene Informationen. Die folgende Aufzählung beschreibt nur die für diese Arbeit relevanten Felder des Zertifikats und ist unvollständig.

Version Die Version des Zertifikats.

Seriennummer Eine Seriennummer, die durch die CA vergeben wurde. Die Nummer muss eindeutig sein, für alle von dieser CA ausgestellten Zertifikate.

Aussteller Enthält den Aussteller des Zertifikats. Über diesen wird versucht eine Kette bis zu einem Root-Zertifikat zu bilden.

Gültigkeit Beschreibt den Zeitraum, in welchem das Zertifikat Gültigkeit besitzt.

Inhaber Hier sind Informationen zum Eigentümer des Zertifikats enthalten, wie beispielsweise Name oder Adresse.

Angaben zum öffentlichen Schlüssel des Inhabers Enthält das öffentliche Schlüsselmaterial des Zertifikats. Dazu gehören der verwendete Algorithmus und der öffentliche Schlüssel.

Erweiterungen Hier können weitere Informationen enthalten sein. Unter anderem werden E-Mail-Adressen in den Erweiterungen abgelegt.

Unter den verfügbaren Erweiterungen für X.509 Zertifikate ist für S/MIME hauptsächlich die Erweiterung *Zertifikatsgegenstand-Alternativ-Name* von Bedeutung. Diese Erwei-

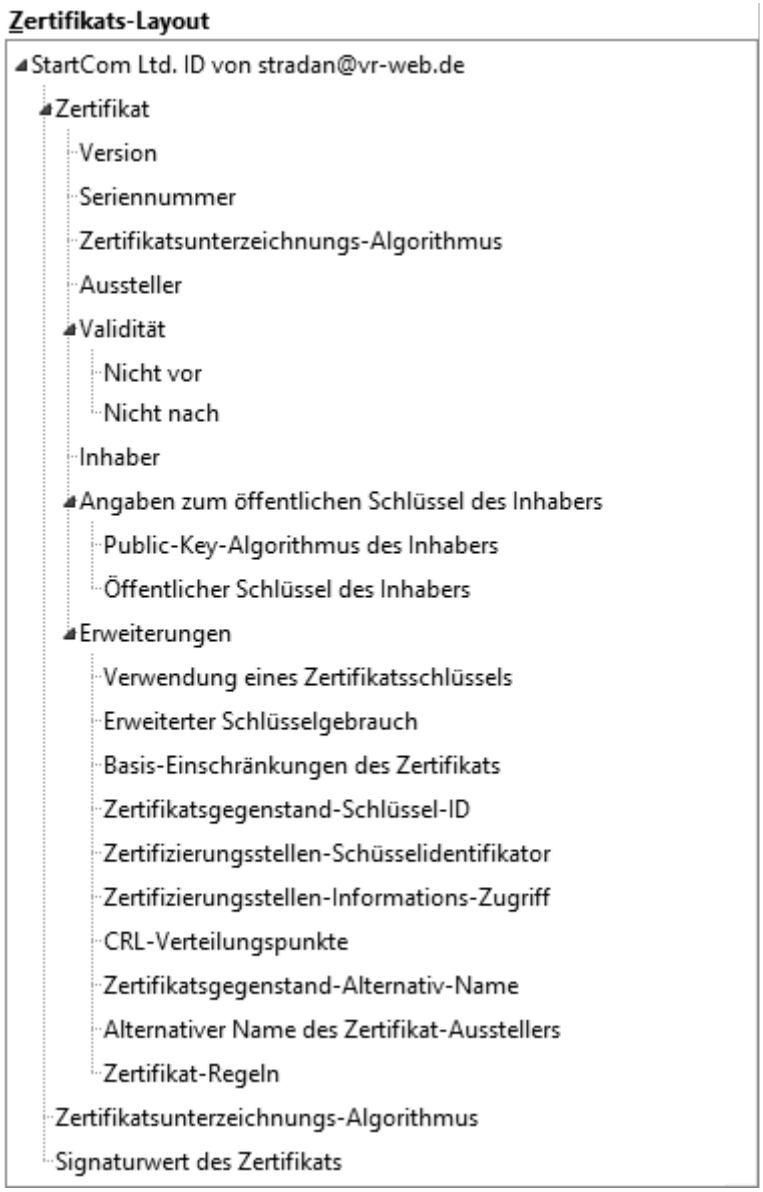

Abbildung 2.1: Beispiel für die Struktur eines S/MIME Zertifikats

terung dient dazu, weitere Identitäten an das Zertifikat zu binden. E-Mail-Adressen werden mit Hilfe dieser Erweiterung an ein Zertifikat gebunden, wenn es für S/MIME verwendet wird.

2.4 Zusammenfassung

Obwohl die beiden Verfahren für sichere E-Mail-Kommunikation nicht untereinander kompatibel sind, basieren sie dennoch beide auf Zertifikaten. Der Enwturf eines Verzeichnisdienstes, welcher beide Arten von Zertifikaten unterstützt, ist aufgrund der Ähnlichkeit der Verfahren gut vorstellbar.

Kapitel 3

Problemanalyse

Im Folgenden wird zuerst aktuelle Stand der Technik untersucht und dort vorhandene Probleme analysiert. Die Betrachtung alternativer Ansätze soll klären, ob bereits Lösungen für die erkannten Probleme existieren und ob diese zur Umsetzung sicherer E-Mail-Kommunikation geeignet sind. Basierend auf diesen Erkenntnissen werden im folgenden Kapitel die Anforderungen an ein System erarbeitet, welches die erkannten Probleme löst.

3.1 Einführung

Damit die Kommunikation über die beiden Protokolle OpenPGP und S/MIME sicher ist, ist es notwendig das Grundprinzip der Protokolle zu kennen. Beide Verfahren basieren auf der asymmetrischen Verschlüsselung. Sobald das dafür notwendige Schlüsselmaterial zwischen zwei Kommunikationspartnern ausgetauscht wurde, kann eine sichere Kommunikation stattfinden. Der entscheidende Punkt ist damit der Austausch des Schlüsselmaterials. Dessen Authentizität und Gültigkeit muss während des Austauschs sichergestellt sein.

Erfolgt die Übergabe des Schlüsselmaterials persönlich oder über einen bereits bekannten gesicherten Kanal, stellt die Frage der Authentizität und Gültigkeit kein Problem dar. Steht ein solcher Kanal hingegen nicht zur Verfügung, so muss die Authentizität und Gültigkeit über andere Wege sichergestellt werden, wie in 2.3.1 und 2.3.2 erläutert.

Die Bereitstellung von Schlüsselmaterial kann über verschiedene Arten geschehen, wie beispielsweise über die eigene Webseite. Sowohl S/MIME als auch OpenPGP definieren aber einen speziellen Kanal dafür: S/MIME beruht dabei auf dem X.500 Standard, welcher einen Verzeichnisdienst beschreibt. Die X.500 konforme Implementierung Lightweight Directory Access Protocol (LDAP) [13] hat sich inzwischen etabliert und wird häufig zur Bereitstellung von S/MIME-Zertifikaten verwendet. Dagegen baut OpenPGP auf dem

HTTP Keyserver Protocol (HKP) auf. Sogenannte *PKS* implementieren das Protokoll HKP und werden zur Veröffentlichung von Public Keys verwendet. [14]

3.2 Stand der Technik

Inzwischen hat sich herausgestellt, dass die genannten Dienste zur Veröffentlichung und Authentifizierung von Schlüsselmaterial bei OpenPGP und S/MIME einige Nachteile und Schwächen haben. Im Folgenden werden die wichtigsten Probleme beschrieben und erklärt.

3.2.1 Web of Trust

Das Web of Trust ist der Mechanismus zur Sicherstellung der Authentizität von Public Keys. Aus diesem Grund muss sich der Anwender auf das WoT verlassen, wenn keine andere Möglichkeit zur Überprüfung der Authentizität zur Verfügung steht. Die Tatsache, dass jede Person beliebige Public Keys erstellen kann, auch im Namen anderer, zeigt dass das Vertrauen in das WoT mit Vorsicht zu sehen ist. Sinnvoll wäre deswegen eine verlässliche Authentifizierungsmöglichkeit der Public Keys. [15]

3.2.2 Certificate Authorities

Die CAs sorgen bei der Signierung eines X.509 Zertifikats für die Überprüfung der Authentizität bestimmter im Zertifikat eingetragener Daten. Bei S/MIME Zertifikaten betrifft das in der Regel die E-Mail-Adresse. Es wird überprüft, ob der Beantragende Zugriff auf die E-Mail-Adresse hat. Die Überprüfung erfolgt in verschiedenen Stufen, wobei auch weitere Informationen wie beispielsweise Name oder Adresse geprüft werden können. Je genauer die Prüfung, umso teurer ist das Ausstellen eines Zertifikats durch eine CA. Das Problem bei der Prüfung durch eine CA ist, dass dieser vertraut werden muss. Eine korrupte oder kompromitierte CA kann, wie bereits geschehen [16], beliebige Zertifikate ausstellen, welche dann von allen Anwendungen so lange als gültig anerkannt werden, bis ein entsprechendes Update der sogenannten Certificate Revocation List (CRL) veranlasst wird. Die CRL listet alle ungültigen Zertifikate auf und dient der Erkennung ungültiger Zertifikate. Problematisch ist die Tatsache, dass eine CRL immer den Zustand zu einem bestimmten Zeitpunkt in der Vergangenheit zeigt und damit keinen Echtzeitschutz vor ungültigen Zertifikaten bieten kann.

3.2.3 Public Key Server

PKS verfügen über keinen Authentifizierungsmechanismus zur Überprüfung der dort hochgeladenen OpenPGP Zertifikate. Dadurch ist das Hochladen beliebiger und somit auch falscher Zertifikate möglich. Ein Benutzer, welcher nach dem Zertfikat einer bestimmten Person sucht, kann aus diesem Grund nicht davon ausgehen, dass ein gefundenes Zertifikat auch tatsächlich der gesuchten Person gehört. Weiterhin ist Handhabung von Aktualisierungen problematisch: Jeder kann Zertifikate von anderen Personen aktualisieren. Der Gedanke dabei ist, dass beispielsweise Signaturen für Benutzerkennungen hochgeladen werden können. Problem ist allerdings, dass der Eigentümer des Zertifikats eine solche Signatur garnicht möchte, da er dann mit anderen Personen in Verbindung gebracht werden kann.

Die meisten Probleme sind im Bezug auf Privatsphäre und Datenschutz zu finden. Der Grund hierfür liegt in der Endgültigkeit der Daten auf einem PKS. Einmal gespeicherte Public Keys lassen sich nicht mehr löschen. Dadurch lassen sich persönliche Daten, die in einem Public Key enthalten sind, nicht mehr entfernen und sind somit für die Öffentlichkeit zugänglich. Auf diese Art können beispielsweise Spammer durch den Abruf von Public Keys sehr einfach an gültige E-Mail-Adressen kommen. Zusätzlich enthalten die Public Keys wie in 2.3.1 beschrieben Signaturen zur Bestätigung der Authentizität. Über die Signaturen lässt sich ein sozialer Graph generieren, wodurch Beziehungen zwischen Personen erkannt werden können, wodurch ebenfalls die Privatsphäre beeinträchtigt wird. [17]

Ebenso problematisch ist das Protokoll HKP welches zur Kommunikation mit PKSs benutzt wird. HKP basiert auf unverschlüsseltem HTTP, weshalb Anfragen generell mitgelesen werden können. Aus der Analyse der Anfragen kann ein Dritter deshalb erkennen, welche Public Keys angefragt werden und somit mit welchen Personen jemand in Kontakt steht. Verbesserung bringt hier die mit TLS abgesicherte Version des Protokolls HKPS, welche von einigen PKS unterstützt wird. Die Benutzung des sicheren Protokolls muss allerdings konfiguriert werden und steht nicht bei allen PKS zur Verfügung.

3.2.4 LDAP Server

Die Situation bei S/MIME ist etwas anders. Während ein PKS für jeden zugänglich ist, sind LDAP Server meist nur innerhalb der jeweiligen Organisation zugänglich. Daraus entsteht für S/MIME sogar ein noch grundlegenderes Problem als bei OpenPGP: Die Verfügbarkeit der S/MIME Zertifikate ist nicht gewährleistet. Auch der öffentliche Zugang zu einem LDAP Server löst das Problem nicht, da jeder einzeln konfiguriert werden müsste, um Zugriff auf die Zertifikate zu erhalten.

3.3 Alternative Ansätze

Die angesprochenen Probleme sind bereits länger bekannt, weshalb es nicht verwundert, dass bereits Alternativen zu den bisherigen Methoden geschaffen wurden. In den folgenden Abschnitten werden alternative Ansätze untersucht und bewertet, welche versuchen die in 3.2 beschriebenen Probleme zu lösen.

3.3.1 DANE

DNS-Based Authentication of Named Entities (DANE) ist eine Erweiterung für DNS und nutzt dabei die mit DNSSEC eingeführten geschützten Einträge als Möglichkeit zur Speicherung von Informationen. Die beiden Varianten DANE/OPENPGPKEY und DANE/SMIMEA wurden entwickelt, um die Authentizität und Zugänglichkeit von Zertifikaten sicherzustellen. Bei DANE/OPENPGPKEY wird zu einer E-Mail-Adresse einfach ein Eintrag im Domain Name System (DNS) hinterlegt, welcher den Public Key enthält. So kann der korrekte und authentifizierte Public Key einfach über eine DNS Anfrage abgerufen werden. Auf fast die gleiche Weise arbeitet DANE/SMIMEA. Alternativ kann hier aber anstatt des öffentlichen Schlüssels auch nur ein Hashwert abgelegt werden. [18]

Die Aufgabe der Authentifizierung fällt dadurch den jeweiligen Domaininhabern zu, da nur ihnen die Möglichkeit eines Eintrags in DNSSEC gegeben ist. Das bedeutet, dass das Vertrauen mit DANE bei den jeweiligen E-Mail-Providern liegt, welche die entsprechenden DANE/OPENPGPKEY und DANE/SMIMEA Einträge verwalten.

Somit hat DANE bezüglich der Authentizität von Schlüsselmaterial die gleichen Probleme, wie sie bereits bei den CAs existieren. Zusätzlich hat DNSSEC den Nachteil, dass Anfragen generell unverschlüsselt sind. Eine Beobachtung des Netzwerkverkehrs ermöglicht Dritten, anhand der Auswertung der Anfragen, Beziehungen zwischen Personen zu erkennen.

Zweifel der Authentizität bleiben also auch bei DANE erhalten, obwohl der Ansatz im Fall von OpenPGP eine deutliche Verbesserung bewirkt. Im Fall von S/MIME verschiebt sich lediglich das Vertrauen von den CAs auf die Mailprovider. Bezüglich Datenschutz und Privatsphäre ist DANE keine große Verbesserung verglichen mit den PKS gelungen. Der einzige Vorteil zeigt sich in der Möglichkeit des Löschens der Einträge, wodurch der Zugang zu persönlichen Daten rückgängig gemacht werden kann.

Größtes Problem von DANE ist allerdings die schlechte Verbreitung von DNSSEC, die eine breite Nutzung bisher verhindert [19] [20].

3.3.2 Sicherer Keyserver

Eine andere Alternative ist die Erweiterung der bisherigen PKS durch zusätzliche Funktionen. In diesen Bereich fällt das PGP Global Directory. Im Vergleich zu anderen PKS bietet es einige Mechanismen zur Verifizierung der E-Mail-Adressen in Public Keys. Bevor ein Public Key veröffentlicht werden kann, versendet das PGP Global Directory Verifizierungsmails an alle genannten E-Mail-Adressen des Public Keys. Erst wenn jede E-Mail-Adresse bestätigt wurde, wird der Public Key im Verzeichnis veröffentlicht. Zusätzlich wird dieser Vorgang alle sechs Monate wiederholt, um nicht mehr verwendete Schlüssel auszusortieren. [21]

Die Verifizierung der E-Mail-Adressen ist eine gute Methode, um die Authentizität eines Public Keys zu überprüfen. Allerdings besteht beim PGP Global Directory, wie bei den anderen PKS, das Problem der Zugänglichkeit zu persönlichen Daten durch Dritte.

3.3.3 Dezentrale Domain Name Systems

Während DANE auf DNS basiert, gibt es auch alternative dezentrale DNS wie zum Beispiel Namecoin oder das GNU Name System (GNS). Der Grund für die Entwicklung dezentraler DNSs ist die mit DNS verbundene Macht über das Internet für diejenigen, die das DNS kontrollieren. Der dezentrale Ansatz verhindert das. [22] [23]

Es stellt sich die Frage, ob ein solches dezentrales DNS als Verzeichnis für Zertifikate Vorteile gegenüber DANE mit sich bringt. Ein Vorteil liegt darin, dass die Einträge persönlich verwaltet werden, wodurch eine Änderung durch Dritte nicht möglich ist. Namecoin sieht das Speichern von Public Key Hashes sogar bereits vor, wie auf den Wiki Seiten beschrieben (https://wiki.namecoin.info/index.php?title=Identity). Daten, welche in einem dezentralen DNS abgelegt werden, sind an eine frei wählbare Id gebunden. Aus diesem Grund besteht weiterhin das Problem, dass abgelegte Daten nicht authentifiziert sind.

Im Vergleich zu DANE ist der dezentrale Ansatz für die Authentizität von Zertifikaten von Nachteil und bietet keine Verbesserung gegenüber einem PKS.

3.4 Zusammenfassung

Zusammenfassend lässt sich folgendes feststellen: Die Authentizität von Zertifikaten und der Schutz der enthaltenen persönlichen Informationen wird von bisherigen Lösungen nur teilweise oder garnicht gewährleistet. Generell gute Ansätze zur Sicherstellung der Authentizität von Zertifikaten sind bei DANE (3.3.1) oder beim, um zusätzliche Funktionalität erweiterten Keyserver (3.3.2) zu finden. Erstaunlicherweise bietet keine der untersuchten Lösungen einen Mechanismum zum Schutz persönlicher Daten.

Im speziellen konnten folgende Probleme erkannt werden, welche im nächsten Kapitel gelöst werden sollen:

- Es findet in den meisten Fällen keine Authentifizierung des Zertifikatsbesitzers statt, weshalb die Übereinstimmung von Besitzer und Eigentümer des Zertifikats ungeprüft bleibt und eine sichere Kommunikation nicht garantiert werden kann.
- Weiterhin hat der Eigentümer des Zertifikats keine Kontrolle über die in den Zertifikaten enthaltenen Daten, wodurch die Privatsphäre gefährdet wird.
- Im Fall OpenPGP sind außerdem unauthorisierte Aktualisierungen von Zertifikaten durch andere Benutzer möglich.

Kapitel 4

Anforderungsanalyse

4.1 Systembeschreibung

Das zu entwerfende System ist ein Verzeichnisdienst für Zertifikate. Im speziellen sollen sowohl OpenPGP- als auch S/MIME-Zertifikate im System gespeichert werden. Die im System abgelegten Zertifikate sollen Benutzern des Systems zur Verfügung gestellt werden. Dabei ist zwischen internen und externen Benutzern zu unterscheiden. Interne Benutzer sind am System registrierte und authentifizierte Benutzer. Alle anderen nicht authentifizierten Benutzer sind externe Benutzer. Im Fall von registrierten Benutzern wird im folgenden von internen Benutzern, im anderen Fall von externen Benutzern gesprochen.

Die Zertifikate sollen von Benutzern in das System hochgeladen und veröffentlicht werden können. Zum Schutz der Privatsphäre soll der Benutzer den Zugriff auf das Zertifikat einschränken können, so dass nur bestimmten Nutzergruppen das Einsehen des Zertifikats ermöglicht wird. Die Authentizität der abgelegten Zertifikate soll vom System gewährleistet werden. Für vom System bereitsgestellte Zertifikate bedeutet dies, dass der jeweilige Eigentümer des Zertifikats Zugriff auf die im Zertifikat genannten Emailadressen hat. Außerdem soll es Benutzern möglich sein ihre Zertifikate zu aktualisieren, zurückzurufen und zu löschen.

Des weiteren soll das System an ein Zertifikatsmanagement angebunden werden können. Das Zertifikatsmanagement ist dabei eine Zertifikate ausstellende Komponente, welche authentische Zertifikate an das System weiterreicht. Zertifikate, welche vom Zertifikatsmanagement an das System übergeben wurden, sollen einem Benutzer zugeordnet, aber noch nicht veröffentlicht werden.

Benutzer sollen signierte OpenPGP-Zertifikate im System ablegen können, die dann dem Eigentümer zur Verfügung gestellt werden.

4.2 Stakeholderanalyse

Im Folgenden werden alle Interakteure des Systems definiert.

4.2.1 Interner Benutzer

Der interne Benutzer agiert ausschließlich authentifiziert mit dem System. Ziel des internen Benutzers ist es, seine Zertifikate im System zu verwalten: Er kann neue Zertifikate hochladen und veröffentlichen. Alte Zertifikate können vom internen Benutzer aktualisiert, zurückgerufen oder gelöscht werden.

Weiterhin ist dem internen Benutzer das Einschränken des Zugriffs auf seine Zertifikate möglich. Zusätzlich können von ihm bei OpenPGP-Zertifikaten einzelne Signaturen sowie Benutzerkennungen ausgeblendet werden.

Er hat die Möglichkeit auf veröffentlichte Zertifikate anderer Benutzer zuzugreifen. Im Fall von OpenPGP kann er signierte Zertifikate für andere hochladen. Unterschriften seiner eigenen Zertifikate durch andere Benutzer werden ihm zur Verfügung gestellt und können durch ihn veröffentlicht werden.

Ein interner Benutzer kann sich vom System abmelden und wird so zu einem externen Benutzer (siehe 4.2.2).

4.2.2 Externer Benutzer (nicht authentifizierter Benutzer)

Im Gegensatz zum internen Benutzer, agiert der externe Benutzer nicht authentifiziert mit dem System. Er kann Zertifikate interner Benutzer anfragen und einsehen, sofern sie für externe Nutzer veröffentlicht wurden. Von ihm signierte OpenPGP Zertifikate interner Benutzer kann er im System hochladen.

Des weiteren kann sich der externe Benutzer beim System registrieren, um sich anschließend am System einloggen zu können. Durch das Einloggen wird er zum internen Benutzer (siehe 4.2.1).

4.2.3 Admin

Der Admin überwacht die Funktionsfähigkeit des Systems. Dafür kann er Kennzahlen, die den aktuellen Zustand des Systems beschreiben, anzeigen. Anhand dieser Kennzahlen kann der Admin auf eventuelle Störungen im System reagieren oder präventive Maßnahmen vornehmen.

4.2.4 Operator

Ein Operator dient der Wartung der Zertifikate. Ungültig gewordene oder kompromittierte Zertifikate werden durch ihn als ungültig markiert oder gelöscht. Zusätzlich kann er den Verzeichnisdienst konfigurieren und auf diese Weise das Verhalten des Dienstes beeinflussen.

4.2.5 Verzeichnisdienst - Schwesterinstanz

Bei der Schwesterinstanz handelt es sich um eine parallel laufende Instanz des Systems. Schwesterinstanzen synchronisieren ihre Daten über ein Pushverfahren. Eine solche Instanz kann verwendet werden, um Zertifikate organisationsübergreifend auszutauschen. Das bringt zwei Vorteile mit sich: Zum einen müssen Benutzer nicht auf verschiedenen Instanzen des Verzeichnisdienstes nach Zertifikaten suchen. Und zum anderen können so Zertifikate weitergegeben werden, ohne sie uneingeschränkt zugänglich machen zu müssen.

4.2.6 Verzeichnisdienst - anderer Dienst

Das System soll kompatibel mit bereits existierenden Verzeichnisdiensten sein. Das Ziel ist im speziellen die Anbindung der Protokolle LDAP sowie HKPS, worüber die Nutzung der öffentlichen Schnittstelle des Systems ermöglicht werden soll.

4.2.7 Zertifikatsmanagement

Wie bereits unter 4.1 beschrieben, soll das System an ein Zertifikatsmanagement angebunden werden. Das Zertifikatsmanagement stellt Zertifikate aus. Dadurch ist es eine Zertifikatsquelle und stellt dem System Zertifikate zur Verfügung. Es kann außerdem Zertifikate aktualisieren, zurückrufen und löschen.

Um die Authentizität der Zertifikatsquelle zu garantieren authentifiziert sich diese am System bevor weitere Aktionen ausgeführt werden.

4.3 Anwendungsfälle

4.3.1 Registrieren

Name Benutzerkonto erstellen

Beschreibung Ein Benutzer registriert ein neues Benutzerkonto mit Eingabe von Benutzerdaten am System.

Akteure Externer Benutzer

Vorbedingungen Der Benutzer ist nicht eingeloggt.

Basisfluss

1 Benutzer wählt registrieren aus.

2 Benutzer gibt Emailadresse und Passwort ein.

3 Benutzer bestätigt die Eingabe.

Fehlerfluss

4 Ist die Emailadresse bereits an ein Benutzerkonto gebunden wird dieser aufgefordert eine andere Emailadresse zu verwenden. Weiter bei 2.

Nachbedingungen Das neue Benutzerkonto wurde im System angelegt.

4.3.2 Einloggen

Name Einloggen

Beschreibung Ein Benutzer authentifiziert sich am System mit seinen Benutzerdaten

Akteure Interner Benutzer, Operator, Admin, Schwesterinstanz, Zertifikatsmanagement

Vorbedingungen Der Benutzer ist nicht eingeloggt.

Basisfluss

1 Benutzer gibt Benutzername und Passwort ein.

2 Benutzer bestätigt die Eingabe.

Alternativfluss

1a Benutzer authentifiziert sich mit Hilfe eines X.509 Zertifikats.

Fehlerfluss

3 Gibt es zum für die Kombination Benutzername und Passwort kein Benutzerkonto, wird eine Fehlermeldung ausgegeben. Weiter bei 1.

Nachbedingungen Der Benutzer ist eingeloggt.

4.3.3 Ausloggen

Name Ausloggen

Beschreibung Ein am System eingeloggter Benutzer loggt sich aus.

Akteure Interner Benutzer

Vorbedingungen Der Benutzer ist eingeloggt.

Basisfluss

1 Benutzer klickt auf abmelden.

Nachbedingungen Der Benutzer ist nicht eingeloggt.

4.3.4 Eigenes Zertifikat hochladen

Name Eigenes Zertifikat hochladen

Beschreibung Der Benutzer lädt sein Zertifikat in das System hoch.

Akteure Interner Benutzer

Vorbedingungen Der Benutzer ist eingeloggt.

Basisfluss

1 Der Benutzer wählt *Zertifikat hochladen* aus.

2 Der Benutzer wählt sein Zertifikat aus und bestätigt dies.

3 Bestätigungscodes werden mit den jeweiligen öffentlichen Schlüsseln verschlüsselt an die im Zertifikat angegebenen Emailadressen geschickt.

4 Der Benutzer gibt für jede Emailadresse den erhaltenen Bestätigungscode ein.

Fehlerfluss

3a Die hochgeladene Datei ist kein Zertifikat von bekanntem Format und es wird eine Fehlermeldung angezeigt. Weiter bei 2.

3b Das Hochladen schlägt fehl und es wird eine enstprechende Fehlermeldung angezeigt. Weiter bei 2.

4a Die Bestätigungscodes sind falsch oder der Benutzer bricht den Vorgang ab. Eine entsprechende Fehlermeldung wird angezeigt.

Nachbedingungen Das hochgeladene Zertifikat wurde im Benutzerkonto gespeichert. Die Authentizität der Emailadressen und des zum Zertifikat gehörigen privaten Schlüssel wurde überprüft.

4.3.5 Zertifikat hochladen

Name Zertifikat hochladen

Beschreibung Der Benutzer lädt ein Zertifikat in das System hoch.

Akteure Zertifikatsmanagement, interner Benutzer

Vorbedingungen Der Benutzer ist eingeloggt.

Basisfluss

1 Das Zertifikatsmanagement lädt das Zertifikat unter Angabe des Benutzerkontos hoch.

2 Bestätigungscodes werden mit den jeweiligen öffentlichen Schlüsseln verschlüsselt an die im Zertifikat angegebenen Emailadressen geschickt.

3 Der Benutzer des zugehörigen Benutzerkontos meldet sich am System an und gibt für jede Emailadresse den erhaltenen Bestätigungscode ein.

Fehlerfluss

1a Die hochgeladene Datei ist kein Zertifikat von bekanntem Format und es wird eine entsprechende Fehlermeldung zurückgegeben.

3a Die Bestätigungscodes sind falsch. Eine entsprechende Fehlermeldung wird angezeigt. Weiter bei 3

Nachbedingungen Das hochgeladene Zertifikat wurde im Benutzerkonto gespeichert. Die Authentizität der Emailadressen und des zum Zertifikat gehörigen privaten Schlüssel wurde überprüft.

4.3.6 Zertifikat veröffentlichen und Zugriff einschränken

Name Zertifikat veröffentlichen und Zugriff einschränken

Beschreibung Der Benutzer veröffentlich ein eigenes Zertifikat und stellt es bestimmten Benutzern des Systems zur Verfügung.

Akteure Interner Benutzer

Vorbedingungen Benutzer ist angemeldet und Zertifikat wurde authentifiziert.

Basisfluss

1 Der Benutzer wählt beim zu veröffentlichen Zertifikat eine Sichtbarkeitsstufe.

Alternativfluss

1a Der Benutzer wählt beim Zertifikat die Sichtbarkeitsstufe *privat* aus, um eine Veröffentlichung rückgängig zu machen.

1b Der Benutzer wählt beim Zertifikat *Sichtbarkeit einstellen* aus.

2b Der Benutzer wählt bei den verfügbaren Elementen des Zertifikats jeweils eine Sichtbarkeitsstufe.

Nachbedingungen Das Zertifikat und dessen Elemente sind entsprechend der gewählten Sichtbarkeitsstufen veröffentlicht.

4.3.7 Zertifikat aktualisieren

Name Zertifikat aktualisieren

Beschreibung Der Benutzer aktualisiert eines seiner Zertifikate.

Akteure Interner Benutzer, Zertifikatsmanagement

Vorbedingungen Der Benutzer ist eingeloggt.

Basisfluss

1 Der Benutzer wählt beim zu aktualisierenden Zertifikat *aktualisieren* aus.

2 Der Benutzer wählt das aktualisierte Zertifikat aus und lädt dieses hoch.

3 Bestätigungscodes werden mit den jeweiligen öffentlichen Schlüsseln verschlüsselt an die im Zertifikat angegebenen Emailadressen geschickt.

4 Der Benutzer gibt für jede Emailadresse den erhaltenen Bestätigungscode ein.

Alternativfluss Alternate Flows

1a Schritt entfällt.

2a Das Zertifikatsmanagement lädt das aktualisierte Zertifikat hoch. Weiter bei 3.

Fehlerfluss

2b Das hochgeladene Zertifikat ist nicht gültig und ein Fehler wird gemeldet.

Nachbedingungen Das alte Zertifikat wurde mit dem hochgeladenen Zertifikat ersetzt und im Benutzerkonto gespeichert. Die Authentizität der Emailadressen und des zum Zertifikat gehörigen privaten Schlüssels wurde überprüft.

4.3.8 Zertifikat zurückziehen

Name Zertifikat zurückziehen

Beschreibung Der Benutzer zieht ein Zertifikat zurück und erklärt es damit für ungültig.

Akteure Interner Benutzer, Operator, Zertifikatsmanagement

Vorbedingungen Der Benutzer ist im System eingeloggt.

Basisfluss

1 Der Benutzer wählt beim zurückzuziehenden Zertifikat *zurückziehen* aus.

Alternativfluss

1a Das Zertifikatsmanagement zieht ein Zertifikat zurück.

Nachbedingungen Das Zertifikat ist als zurückgezogen markiert.

4.3.9 Zertifikat löschen

Name Zertifikat löschen

Beschreibung Der Benutzer entfernt eines seiner Zertifikate vom System.

Akteure Interner Benutzer, Operator, Zertifikatsmanagement

Vorbedingungen Der Benutzer ist im System eingeloggt.

Basisfluss

1 Der Benutzer wählt beim zu löschenden Zertifikat *löschen* aus.

Alternativfluss

1a Das Zertifikatsmanagement löscht ein Zertifikat.

Nachbedingungen Das Zertifikat ist nicht mehr im System vorhanden.

4.3.10 Zertifikat für Benutzer anfragen

Name Zertifikat für Benutzer anfragen

Beschreibung Der Benutzer fragt ein Zertifikat für einen am System registrierten Benutzer an.

Akteure Interner Benutzer, externer Benutzer

Basisfluss

1 Der Benutzer gibt im Suchfeld einen Suchbegriff ein.

2 Der Benutzer findet in der Ergebnisliste das gewünschte Zertifikat und kann weitere Aktionen ausführen.

Fehlerfluss

2a Der Benutzer findet das gewünschte Zertifikat nicht in der Ergebnisliste. Weiter bei 1.

Nachbedingungen Der Benutzer hat das gewünschte Zertifikat gefunden und kann weitere Aktionen ausführen.

4.3.11 OpenPGP Zertifikat eines anderen Benutzers aktualisieren

Name OpenPGP Zertifikat eines anderen Benutzers aktualisieren

Beschreibung Der Benutzer aktualisiert ein OpenPGP Zertifikat eines anderen Benutzers

Akteure Interner Benutzer, externer Benutzer

Basisfluss

1 Der Benutzer wählt beim zu aktualisierenden Zertifikat *aktualisieren* aus.

2 Der Benutzer wählt das aktualisierte Zertifikat aus und bestätigt.

Fehlerfluss

3 Das hochgeladene Zertifikat entspricht nicht dem zu aktualisierenden Zertifikat und eine Fehlermeldung wird angezeigt.

Nachbedingungen Der Eigentümer des Zertifikats wird über die Aktualisierung informiert und erhält die Möglichkeit diese zu veröffentlichen.

4.3.12 Signaturen eines eigenen OpenPGP Zertifikats löschen oder übernehmen

Name Signaturen eines eigenen OpenPGP Zertifikats löschen oder übernehmen

Beschreibung Der Benutzer löscht einzelne Signaturen seines OpenPGP Zertifikats.

Akteure Interner Benutzer

Vorbedingungen Der Benutzer ist im System eingeloggt.

Basisfluss

1 Der Benutzer wählt bei einem eigenen OpenPGP Zertifikat *Signaturen verwalten* aus.

2 Der Benutzer wählt bei der zu löschenden Signatur *löschen* aus.

Alternativfluss

2a Der Benutzer wählt bei einer durch einen anderen Benutzer hinzugefügten Signatur *übernehmen* aus.

Nachbedingungen Das Zertifikat ist mit den aktualisierten Signaturen im System gespeichert.

4.4 Angreifermodelle

Der zu entwerfende Verzeichnisdienst soll, wie bereits in Kapitel 4.1 beschrieben, die Authentizität der Zertifikate sowie die Privatsphäre der Benutzer gewährleisten. Im Hinblick auf die im System vorhandenen Zertifikate bedeutet das, dass die jeweiligen Benutzer sowohl Zugriff auf die in den Zertifikaten genannten Emailkonten, als auch auf den zum Zertifikat gehörigen privaten Schlüssel haben müssen. Um diese Authentizität garantieren zu können, muss auch die Integrität der Daten sichergestellt sein.

Zum Schutz der Privatsphäre ist es den Benutzern möglich, Zertifikate mit unterschiedlichem Umfang an persönlichen Daten für unterschiedliche Benutzergruppen zu veröffentlichen. Des weiteren muss dafür auch die Tatsache, welche Zertifikate von Benutzern angefragt werden, verborgen bleiben.

Im folgenden werden verschiedene Angriffsszenarien erläutert, gegen die das System im Bezug auf die genannten Aspekte geschützt wird.

4.4.1 Außenstehender Angreifer

Dieser Angreifer kann als externer Benutzer die nicht authentifizierte Schnittstelle des Systems benutzen. Weiterhin ist es ihm möglich den Netzwerkverkehr mitzulesen.

Über die nicht authentifizierte Schnittstelle ist es dem Angreifer nicht möglich Daten im System zu ändern. Das Erstellen eines neuen Benutzerkontos ist ihm möglich, dadurch können aber keine schon vorhandenen Daten im System geändert werden. Somit ist die Integrität und damit auch die Authentizität der Daten sichergestellt.

Persönliche Daten kann der Angreifer nur in dem Umfang einsehen, wie der jeweilige Benutzer dies für externe Benutzer erlaubt. Um das Mitlesen des Netzwerkverkehrs zu verhindern, wird die Kommunikation zwischen Benutzern und dem System verschlüsselt.

4.4.2 Angreifer als Teil des Systems

Ein Angreifer, welcher Teil des Systems ist, hat je nach ihm gewährten Zugriffsrechten verschiedene Angriffsmöglichkeiten.

4.4.2.1 Angreifer ist interner Benutzer

Als interner Benutzer steht dem Angreifer die interne Schnittstelle des Systems zur Verfügung. Der Angreifer kann eigene Zertifikate hochladen und erhält erweiterten Zugriff auf Zertifikate.

Die Integrität der bereits vorhandenen Daten bleibt aber erhalten, da dem Angreifer das Ändern der Daten nicht möglich ist. Durch die Authentifizierung neuer Zertifikate kann der Angreifer nur authentische Daten zum System hinzufügen, wodurch auch die Authentizität der Daten im System gewährleistet ist (siehe auch 5.3.2.2).

Persönliche Daten kann der Angreifer nur in dem Umfang einsehen, wie der jeweilige Benutzer dies für interne Benutzer oder selbst definierte Benutzergruppen erlaubt.

4.4.2.2 Angreifer ist Operator

Hat der Angreifer die Zugriffsrechte eines Operators, so verfügt dieser über die Fähigkeit beliebige Zertifikate zu löschen oder zurückzuziehen.

Um die Privatsphäre der Benutzer zu gewährleisten, kann der Operator zwar alle Zertifikate sehen, jedoch keine Details dieser Zertifikate, welche persönliche Daten eines Benutzers enthalten.

Das Löschen von Zertifikaten hat keine Auswirkung auf die Authentizät der Daten im System. Ebenso beeinflusst das Zurückziehen eines Zertifikats nicht die gewünschte Authentizität der Zertifikate, weil weder öffentliche Schlüssel noch in den Zertifikaten enthaltene Emailadressen geändert werden.

4.4.2.3 Angreifer ist Admin

Ein Angreifer mit Rechten eines Admins kann Metadaten des Systems einsehen.

Eine Änderung der Daten im System ist mit dieser Zugriffsrolle nicht möglich, die Integrität und damit die Authentizität der Daten im System bleibt damit gewährleistet. Um die Privatsphäre der Benutzer nicht zu gefährden müssen die Metadaten die dem Admin zur Verfügung gestellt werden anonymisiert oder zusammengefasst sein. Dadurch ist auch dem Angreifer die Möglichkeit verwährt auf private Daten anderer Benutzer zuzugreifen.

4.5 Funktionale Anforderungen

Um die beschriebenen Funktionalitäten bieten zu können, muss das System die im Folgenden beschriebenen funktionalen Anforderungen erfüllen.

Damit Zertifikate im System hinterlegt werden können, muss es die Funktionalität bieten ein Zertifikat zu *persistieren*. Hierfür müssen die beiden Formate OpenPGP und X.509 gelesen werden können, um die in den Zertifikaten enthaltenen Metainformation speichern zu können. Weiterhin muss ein Zertifikat in seiner Originalform gespeichert werden, um dieses später anfragenden Benutzern bereitzustellen. Die so gespeicherten Zertifikate müssen für anfragende Benutzer *bereitgestellt* werden. Dies soll einmal in Form von Metadaten geschehen, die das Zertifikat beschreiben. Außerdem soll das Zertifikat auch in binärer Form bereitgestellt werden, um dem Benutzer das Zertifikat als solches zur Verfügung zu stellen. Um die Privatsphäre der Benutzer zu schützen und um veraltete Daten zu entfernen, muss ein einmal gespeichertes Zertifikat wieder aus dem System *gelöscht* werden können. Zusätzlich müssen Zertifikate *aktualisierbar* sein. Das System benötigt dafür eine Möglichkeit alte Zertifikatsdaten durch neue ersetzen zu können.

Im Blick auf die Privatsphäre der Benutzer gibt es die folgenden funktionale Anforderungen. Der *Zugriff* auf gespeicherte Zertifikate und darin enthaltene Daten muss *einschränkbar* sein. Dem Besitzer soll so ermöglicht werden, nur bestimmten Personen Zugriff auf seine Daten zu gewähren. Soweit das jeweilige Zertifikatsformat das unterstützt, soll ein Zertifikat auch entsprechend der gewählten Zugriffseinschränkung *gefiltert* werden. Auf diese Weise können Benutzer den Zugriff auf bestimmte persönliche Daten feingradiger einschränken.

Für OpenPGP Zertifikate soll außerdem die Möglichkeit bestehen, dass *Signaturen für Zertifikate anderer Benutzer im System abgelegt* werden können. Hierfür muss eine Schnittstelle zum hochladen der signierten Zertifikate zur Verfügung gestellt werden. Im Anschluss müssen die *Signaturen* dem betreffenden Benutzer *zur Verfügung gestellt* werden, damit dieser die Signatur übernehmen kann.

Eine weitere Funktionalität ist die Bereitstellung von Metadaten des Systems, welche den aktuellen Systemstatus widerspiegeln und eine Unterstützung bei der Fehlerbehebung sowie -prävention bieten.

4.6 Technische Anforderungen

In diesem Kapitel werden die technischen Anforderungen an das System spezifiziert und erläutert.

Ein großes Augenmerk gilt der Authentizität aller im System vorhandenen Daten. Diese

soll dem Benutzer die Verwendung von Emailverschlüsselung erleichtern, indem sie den Austausch von Zertifikaten absichert. Wichtig ist die Sicherstellung der Authentiztät der Zertifikate: Der Benutzer, welcher das Zertifikat im System ablegt, muss Zugriff auf den privaten Schlüssel des Zertifikats, sowie auf die mit dem Zertifikat abgedeckten Emailadressen haben. Erst dann darf das Zertifikat für andere Benutzer freigegeben werden. Um die Authentizität sicherzustellen, sollen Authentifizierungstokens verwendet werden, welche an die jeweiligen E-Mail-Adressen des Zertifikats versandt werden. Die Authentifizierungstokens werden durch den Benutzer eingeben, wodurch das Zertifikat authentifiziert wird.

Ebenso große Bedeutung hat die Sicherstellung der Privatsphäre. Um diese zu gewährleisten muss die Kommunikation zwischen Benutzer und System verschlüsselt sein, wie bereits in 4.4.1 festgestellt.

Für die Persistierung der Daten ist eine Datenbank erforderlich. Sie muss die anfallenden Zertifikate und damit verbundene Daten speichern.

4.7 Datenanalyse

Im folgenden Abschnitt werden die im System anfallenden Daten analysiert, welche zur Bereitstellung der in den vorherigen Kapiteln beschriebenen Funktionen notwendig sind.

Die anfallenden Daten können in zwei Bereiche unterteilt werden: Benutzerdaten und Zertifikatsdaten. Zertifikatsdaten sind Daten, welche im direkten Zusammenhang mit einzelnen Zertifikaten stehen und den Inhalt der Zertifikate beschreiben. Benutzerdaten sind Informationen über den Benutzer und dessen Präferenzen im System, auch im Bezug auf seine Zertifikate.

4.8 Zusammenfassung

In diesem Kapitel wurden die folgenden entscheidenden Anforderungen an einen sicheren und die Privatsphäre schützenden Zertifikatsverzeichnisdienst gefunden:

- Es muss eine Authentifizierung der Zertifikate über die zugehörigen E-Mail-Adressen stattfinden. Als geeignetes Verfahren sollen hierfür Authentifizierungstokens verwendet werden.
- Zum Schutz der Privatsphäre von Zertifikatseigentümern muss der Zugriff auf persönliche Daten eingeschränkt werden können. Ein Zugriffsystem, das den Zugriff auf die Daten in verschiedenen Stufen einschränken lässt ist hierfür umzusetzen.

- Eine Änderung von OpenPGP Zertifikaten, durch andere Personen als den Eigentümer selbst, darf nur mit dessen Zustimmung geschehen. Hierfür ist der Eigentümer bei einer Änderung zu Benachrichtigen, woraufhin dieser die Änderung übernehmen oder ablehnen kann.

Kapitel 5

Systementwurf

Anhand der in Kapitel 4 erarbeiteten Anforderungen wird in diesem Kapitel ein System entworfen. Dem Top-Down-Entwurf folgend, wird zuerst die grobe Architektur beschrieben. Darauf aufbauend werden die einzelnen Komponenten und die erforderlichen Schnittstellen im Detail erarbeitet.

5.1 Übersicht

Das System kann, wie in Abbildung 5.1, generell in drei Bereiche aufgeteilt werden:

Zertifikatsverzeichnisdienst Diese Komponente vereint die Funktionalität zur Zertifikatsverwaltung. Sie bietet verschiedene Schnittstellen an, über welche Benutzer Zugriff auf die Funktionen des Verzeichnisdienstes erlangen.

Grafische Benutzerschnittstelle Die grafische Schnittstelle ermöglicht dem Benutzer eine einfache Interaktion mit dem Zertifikatsverzeichnisdienst.

Datenbank Die Datenbank dient der Persistierung aller Daten, welche für die Ausführung des Zertifikatsverzeichnisdienstes notwendig sind.

Bevor die einzelnen Komponenten genauer beschrieben werden, wird im folgenden Abschnitt die Handhabung von Zertifikaten im System definiert.

5.2 Handhabung von Zertifikaten

5.2.1 Lebenszyklus von Zertifikaten

Zur Verwaltung der Zertifikate im System müssen verschiedene Zustände eingeführt werden, welche den aktuellen Status eines Zertifikats beschreiben.

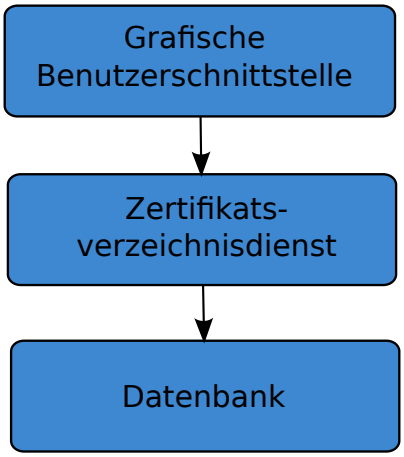

Abbildung 5.1: Systemüberblick

Neu Ein Zertifikat erhält diesen Status, sobald es im System abgelegt wird. Ein solches Zertifikat ist nur für den zugehörigen Besitzer oder einen Operator sichtbar. Ein Zertifikat in diesem Zustand kann gelöscht werden oder die Authentifizierung kann gestartet werden.

Authentifizierung ausstehend Wird für ein Zertifikat die Authentifizierung eingeleitet, geht es in den Zustand *Authentifizierung ausstehend* über. Die Authentifizierung wird in zwei Fällen eingeleitet: Wenn ein Zertifikat neu ist oder aber aktualisiert wird. In diesem Zustand verbleibt es so lange, bis es wie in 5.3.2.2 beschrieben authentifiziert wurde.

Authentifiziert Nachdem ein Zertifikat authentifiziert ist, kann es vom Besitzer veröffentlicht werden. Hierfür legt der Besitzer für das Zertifikat eine in 5.2.2 definierte Sichtbarkeitsstufe fest.

Veröffentlicht Ein Zertifikat mit dem Status *veröffentlicht* kann je nach Sichtbarkeitsstufe von anderen Benutzern abgerufen werden.

Ungültig Ein Operator oder der Besitzer eines Zertifikats, kann dieses als ungültig markieren. Mit diesem Status kann ein Zertifikat zwar noch im System gefunden, aber nicht mehr abgerufen werden.

Abbildung 5.2 zeigt die oben beschriebenen Zusammenhänge und Übergänge zwischen den Zuständen eines Zertifikats.

5.2.2 Sichtbarkeit von Zertifikaten

Wie in 4.1 beschrieben, soll der Zugriff auf Zertifikate einschränkbar sein. Zur Umsetzung dieser Zugriffskontrolle, sollen für Zertifikate verschiedene Sichtbarkeitsstufen

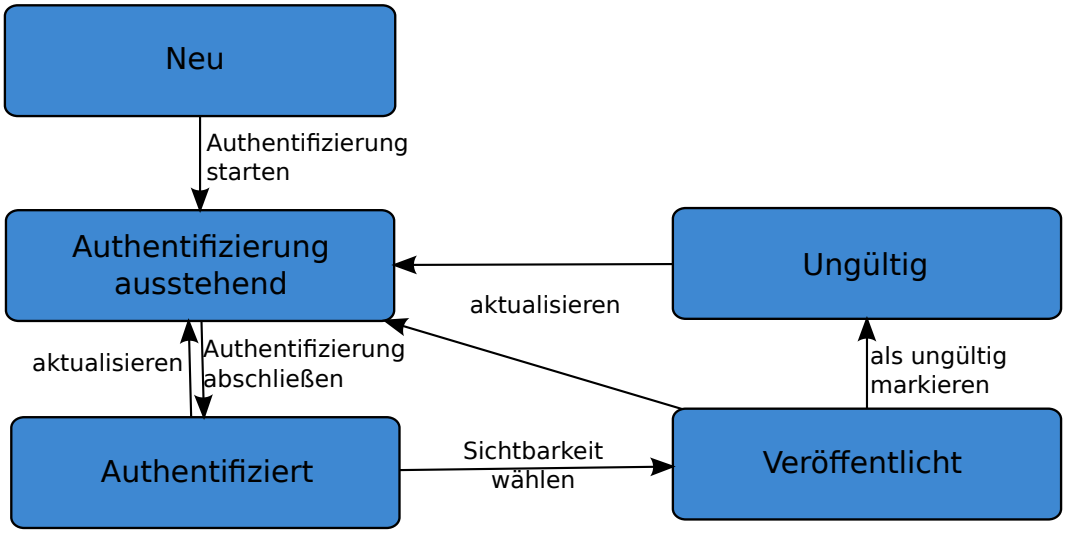

Abbildung 5.2: Lebenszyklus eines Zertifikats

eingeführt werden. Die unterschiedlichen Sichtbarkeitsstufen definieren wer Zugriff auf ein Zertifikat oder Details von Zertifikaten hat.

Öffentlich Diese Sichtbarkeitsstufe erlaubt jeder Person Zugriff auf das entsprechende Zertifikat.

Intern Die Sichtbarkeit *intern* schließt alle nicht authentifzierten Benutzer vom Zugriff aus. Damit haben nur interne Benutzer und das Zertifikat abzurufen.

Privat Diese Sichtbarkeitsstufe erlaubt den Zugriff nur für den Besitzer des Zertifikats.

Benutzerdefiniert Um dem Benutzer stärkere Kontrolle über die Vergabe von Zugriffsrechten auf seine Zertifikate zu geben, gibt es eine weitere Sichtbarkeitsstufe: *benutzerdefiniert*. Über diese wird dem Benutzer die Möglichkeit gegeben, den Zugriff auf ein Zertifikat für einzelne interne Benutzer oder Benutzergruppen freizugeben. Umgesetzt werden kann dies beispielsweise über Wildcardaddressen.

5.2.3 Sichtbarkeit von OpenPGP Zertifikaten

Für OpenPGP Zertifikate können die Sichtbarkeitsstufen auch für Benutzerkennungen und einzelne Signaturen verwendet werden (vgl. 4.2.1).

Um die Sichtbarkeit von Benutzerkennungen und Signaturen kontrollieren zu können, müssen diese entsprechend aus dem Zertifikat entfernt werden. Das heißt, wenn ein Benutzer ein OpenPGP Zertifikat abruft, welches Signaturen oder Benutzerkennungen enthält, die für ihn nicht sichtbar sind, so müssen diese aus dem Zertifikat entfernt werden. Die Aneinanderkettung von Paketen bei OpenPGP Zertifikaten, wie in 2.3.1.1

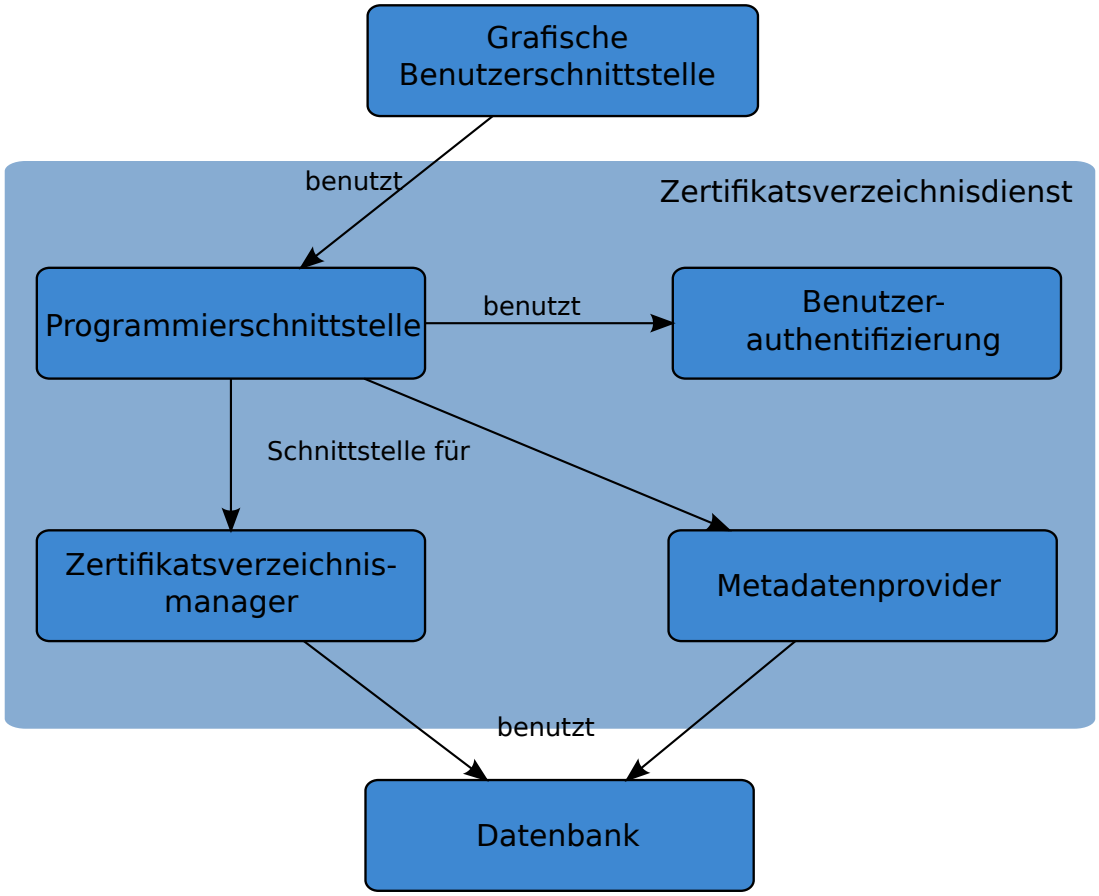

Abbildung 5.3: Aufbau des Zertifikatsverzeichnisdienstes

beschrieben, ermöglicht dies, indem entsprechend die *User ID Packets* und *Signature Packets* aus dem Zertifikat entfernt werden. Hierfür wird das Zertifikat vor der Herausgabe an einen Benutzer vom Zertifikatsparser gefiltert (siehe 5.3.2.1).

5.3 Zertifikatsverzeichnisdienst

Basierend auf Kapitel 4 lässt sich der Zertifikatsverzeichnisdienst, wie in Abbildung 5.3, in mehrere Komponenten unterteilen.

Benutzerauthentifizierung Damit Benutzer eigene Zertifikate speichern und den Zugriff darauf einschränken können, ist eine Benutzerauthentifizierung erforderlich. Sie ist für die Zuweisung der Rolle eines Benutzers innerhalb des Systems verantwortlich.

Zertifikatsverzeichnismanager Die Funktionen zur Verwaltung der Zertifikate können als Bibliothek gekapselt werden. Diese Komponente wird im weiteren Verlauf

als Zertifikatsverzeichnismanager bezeichnet.

Metadatenprovider Für den Administrator (vgl. Abschnitt 4.2.3) müssen Metadaten zur Überwachung der Funktionsfähigkeit des Systems bereitgestellt werden. Diese Funktionalität kann vom Rest des Systems abgekapselt werden. Die entsprechende Komponente wird Metadatenprovider genannt.

Programmierschnittstelle Der Zertifikatsmanager wird als Bibliothek konzipiert. Auf diese Weise kann der Zertifikatsverzeichnismanager von anderen Anwendungen benutzt werden. Für die Anbindung an das Zertifikatsmanagement (siehe 4.2.7) benötigt das System eine Programmierschnittslelle nach außen. Basierend auf der dieser Schnittstelle ist auch die Erstellung einer webbasierte Oberfläche für das System möglich.

5.3.1 Benutzerauthentifizierung

Funktionen des Systems sollen basierend auf Nutzerrollen ausgeführt werden, um Zugriffsrechte auf diese Funktionen zuteilen zu können. Aus der Stakeholderanalyse (siehe 4.2) ergeben sich verschiedene Rollen, die ein Benutzer im System einnehmen kann. Die folgenden Rollen sollen verwendet werden.

Benutzer Diese Rolle ermöglicht es einem Benutzer das System als Zertifikatsverzeichnisdienst zu nutzen (vgl. 4.2.1).

Admin Diese Rolle erlaubt Zugriff auf die Metadaten für die administrative Überwachung des Systems (vgl. 4.2.3).

Operator Diese Rolle erlaubt Zugriff auf operative Funktionen wie in 4.2.4 beschrieben.

System Diese Rolle erlaubt erweiterten Zugriff auf Zertifikate, wie zur Synchronisation von Daten erforderlich (vgl. 4.2.5 und 4.2.7).

Nimmt ein Benutzer keine der genannten Rollen ein, so steht ihm nur eine eingeschränkte Funktionalität zur Verfügung (vgl. 4.2.2). Jeder nicht-authentifizierte Zugriff am System soll mit eingeschränkter Funktionalität durchgeführt werden. Das betrifft auch Zugriffe durch externe Verzeichnisdienste (vgl. 4.2.6).

5.3.2 Zertifikatsverzeichnismanager

Der Zertifikatsverzeichnismanager verwaltet alle Zertifikate und bietet die damit verbundenen Funktionen als Bibliothek an. Die erforderlichen Funktionen zur Verwaltung der Zertifikate können noch in weitere Subkomponenten aufgeteilt werden, welche in Abbildung 5.4 dargestellt werden.

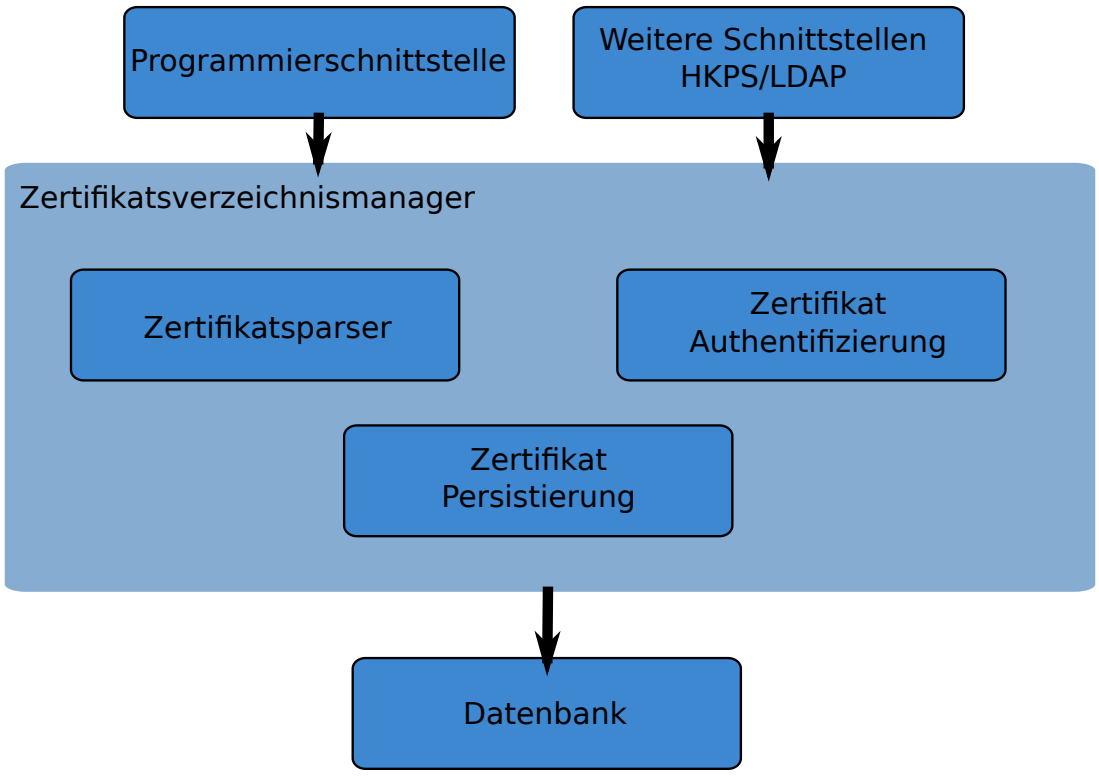

Abbildung 5.4: Aufbau des Zertifikatsverzeichnismanagers

Die Funktionen des Zertifikatsverzeichnismanagers sind die in Abschnitt 4.5 beschriebenen.

5.3.2.1 Zertifikatsparser

Um Zertifikate in binärer Form einlesen und wieder ausgeben zu können, verwendet der Zertifikatsverzeichnismanager eine Subkomponente, die im folgenden Zertifikatsparser genannt wird. Der Zertifikatsparser soll X.509 Zertifikate und OpenPGP Zertifikate in deren üblichen binären Formen einlesen und in eine zur Verarbeitung geeignete Struktur umgewandelt werden. Diese Struktur soll innerhalb des Systems genutzt werden. Wird ein Zertifikat angefragt, so benutzt der Zertifikatsverzeichnismanager den Zertifikatsparser, um das Zertifikat in standardisierter Form auszugeben. Im Fall von openPGP Zertifikaten wird das Zertifikat an dieser Stelle noch entsprechend der gesetzten Sichtbarkeitslevel gefiltert. Außerdem ermöglicht der Zertifikatsparser das Hinzufügen einzelner Signaturen, die nicht durch den Eigentümer eines Zertifikats gespeichert wurden (vgl. 5.3.2.3 und 4.5). Für die Verarbeitung von OpenPGP und X.509 Zertifikaten existieren bereits verschiedene Bibliotheken. Aus diesem Grund bietet sich die Verwendung einer bereits existierenden Bibliothek für das Lesen und Verarbeiten der Zertifikate an.

5.3.2.2 Zertifikat Authentifizierung

Wie bereits in 4.6 beschrieben, sollen zur Authentifizierung von Zertifikaten Authentifizierungstokens verwendet werden. Die entsprechende Funktionalität soll in der Subkomponente *Zertifikat Authentifizierung* gekapselt werden. Soll ein Zertifikat authentifiziert werden, wird der Zertifikatsverzeichnismanager das Zertifikat an die *Zertifikat Authentifizierung* weitergeben. Dort werden für die einzelnen E-Mail-Adressen zufällige Tokens generiert und dann mit dem Zertifikat verschlüsselt an die jeweilige E-Mail-Adresse gesendet. Die *Zertifikat Authentifizierung* bietet auch Funktionen zur Überprüfung eines Tokens an.

5.3.2.3 Zertifikat Persistierung

Zur Persistierung von Zertifikaten soll ebenfalls eine Subkomponente verantwortlich sein. Sie soll vor allem dazu dienen, das Persistieren der Zertifikate zu kapseln, um eine Anbindung an verschiedene Datenbanksysteme zu ermöglichen. Persistiert werden sollen neben der binären Form eines Zertifikats, alle für das System relevanten Informationen, die im Zertifikat enthalten sind. Darunter fallen:

Typ Typ des Zertifikats: OpenPGP oder S/MIME.

Benutzerkennungen Die zum Zertifikat gehörigen Benutzerkennungen mit den E-Mail-Adressen.

Identifikator ID des Zertifikats. Bei S/MIME die Seriennummer und bei OpenPGP der Fingerabdruck.

Signaturen Nur bei OpenPGP die Signaturen für das Zertifikat. Signaturen, die nicht durch den Zertifikatseigentümer hochgeladen werden, werden separat gespeichert. Der Eigentümer kann diese Signaturen dann übernehmen, worauf sie im entsprechenden Zertifikat gespeichert wird.

Zusätzlich fallen noch weitere das Zertifikat betreffende Informationen an:

Benutzer Der Benutzer, welchem das Zertifikat gehört.

Status Der Status des Zertifikats (vgl. 5.2.1).

Sichtbarkeit Die Sichtbarkeit des Zertifikats (vgl. 5.2.2).

Sichtbarkeit von Benutzerkennungen Bei OpenPGP fallen auch Informationen zur Sichtbarkeit einzelner Benutzerkennungen an.

Tokens Authentifizierungstokens für noch nicht authentifizierte E-Mail-Adressen.

5.3.2.4 Zugriffslevel

Um die Zugriffsbeschränkungen für Zertifikate umsetzen zu können, werden verschiedene Zugriffslevel für ausgeführte Funktionen innerhalb des Zertifikatverzeichnismanagers benötigt. Folgende Zugriffslevel ermöglichen die beschriebenen Einschränkungen:

Privat ist zu verwenden, wenn der Zugriff auf eigene Zertifikate erfolgt. Alle Daten und Funktionen können verwendet werden.

Intern ist für den Zugriff auf fremde Zertifikate zu verwenden, wenn der Benutzer am System authentifiziert ist. Mit diesem Level kann nur auf Daten zugegriffen werden, welche mit der Sichtbarkeitsstufe *intern* oder *öffentlich* veröffentlicht sind. Zusätzlich kann auch auf Daten mit der Sichtbarkeitsstufe *benutzerdefiniert* zugegriffen werden, wenn dem Benutzer der Zugriff explizit oder über eine Benutzergruppe gewährt wird.

Öffentlich ist zu verwenden, wenn keine Authentifizierung am System stattgefunden hat. Der Zugriff ist lediglich auf Daten mit der Sichtbarkeitsstufe *öffentlich* möglich.

System wird ausschließlich für systeminterne Zugriffe benutzt und ermöglicht uneingeschränkten Zugriff auf Daten und Funktionen.

Operator wird für Zugriffe durch Benutzer mit der Rolle Operator verwendet.

Tabelle 5.1: Kennzahlen zu den Zugriffen am System

Kennzahl	Beschreibung
Allgemeine Zugriffsdichte	Beschreibt die Anzahl an Zugriffen pro Zeiteinheit.
Zugriffsdichte lesend	Beschreibt die Anzahl lesender Zugriffe pro Zeiteinheit.
Zugriffsdichte schreibend	Beschreibt die Anzahl schreibender Zugriffe pro Zeiteinheit. Schreibende Zugriffe sind alle Zugriffe, welche Änderungen an gespeicherten Daten vornehmen oder Daten hinzufügen/löschen.
Anzahl aktiver Benutzer	Beschreibt die Anzahl an Benutzern, welche im Moment das System benutzen.

Die entsprechenden Zugriffslevel sind, anhand der Rolle eines Benutzers und dem Zugriffsobjekt, wie oben beschrieben zu bestimmen.

5.3.3 Metadatenprovider

In diesem Abschnitt werden Metadaten beschrieben, welche zur Überwachung der Funktionalität des Systems genutzt werden können.

5.3.3.1 Anzahl Zugriffe

Als wichtige Kennzahl dient die Anzahl an Zugriffen pro Zeiteinheit. Anhand der Größe dieser Zahl kann die Auslastung des Systems gemessen und wenn notwendig Maßnahmen zur Erhöhung der Leistung getroffen werden.

Sinnvoll ist auch eine Aufschlüsselung der Art der Zugriffe, wie zum Beispiel die Anzahl angefragter und gespeicherter Zertifkate. In Tabelle 5.1 sind die verschiedenen bereitgestellten Kennzahlen aufgelistet. Um diese Daten sinnvoll zu sammeln, ist die Verwendung des Entwurfsmusters Beobachter von Vorteil. Der Zertifikatsverzeichnismanager informiert den Metadatenprovider über ein Ereignis, wenn dieses Eintritt. Auf diese Weise kann die Anzahl und Art der Zugriffe effizient gesammelt werden.

Die gesammelten Daten sollen aufbereitet werden und über verschiedene Zeiträume abgefragt werden können. Dadurch kann ein Administrator die Entwicklung der Auslastung verfolgen.

5.3.3.2 Daten

Um die Auslastung des Speichers bewerten zu können, sind auch Kennzahlen zu den im System vorhandenen Daten releveant. Die Daten hierfür können vom Metadaten-

Tabelle 5.2: Kennzahlen zu im System vorhandenen Daten

Kennzahl	Beschreibung
Registrierte Benutzer	Die Anzahl am System registrierter Benutzer.
Gespeicherte Zertifikate	Die Anzahl gespeichter Zertifikate.
Gespeicherte OpenPGP Zertifikate	Die Anzahl gespeicherter OpenPGP Zertifikate.
Gespeicherte S/MIME Zertifikate	Die Anzahl gespeicherter S/MIME Zertifikate.
Authentifizierte Zertifikate	Die Anzahl an Zertifikaten, welche den Status *authentifiziert* haben.
Veröffentlichte Zertifikate	Die Anzahl an Zertifikaten, welche den Status *veröffentlicht* haben.
Ungültige Zertifikate	Die Anzahl an Zertifikaten, welche den Status *ungültig* haben.

provider aus der Datenbank ausgelesen und ausgewertet werden. Tabelle 5.2 zeigt die vom Metadatenprovider bereitzustellenden Kennzahlen dieser Art.

5.4 Schnittstellen

Zur Verwendung des Systems muss eine Schnittstelle zur Verfügung gestellt werden, welche die Funktionen des Zertifikatsverzeichnismanagers zugänglich machen. Zusätzlich soll existierenden Verzeichnisdiensten über die Protokolle LDAP und HKPS das Anfragen von Zertifikaten möglich sein (siehe 4.2.6). Diese Schnittstellen werden in den Folgenden Abschnitten beschrieben.

5.4.1 Programmierschnittstelle

Alle vom Zertifikatsverzeichnismanager angebotenen Funktionen sollen über eine Schnittstelle zugänglich gemacht werden, damit diese von außen durch verschiedene Anwendungen benutzt werden können. Sie soll im speziellen vom Zertifikatsmanagement (4.2.7) und für die Bereitstellung einer webbasierten Oberfläche verwendet werden. Zusätzlich zu den Funktionen, welche der Zertifikatsverzeichnismanager anbietet, soll über die Programmierschnittstelle auch die Authentifizierung der Benutzer möglich sein.

5.4.2 Interoperabilität

Der Verzeichnisdienst soll interoperabel mit anderen, für Zertifikate üblicherweise verwendeten Verzeichnisdiensten sein. Hierfür ist die Unterstützung der im Folgenden beschriebenen Protokolle und Funktionen vorgesehen.

5.4.2.1 LDAP

S/MIME Zertifikate sollen über LDAP abgerufen werden können, um Kompatibilität zu aktuellen Systemen zu erreichen. Die Zertifikate sollen über das in RFC 4523 ([24]) definierte Schema angefragt werden. Die LDAP Schnittstelle soll Anfragen nach E-Mail-Adressen unterstützen und entsprechend RFC 4523 zur Anfrage passende Zertifikate zurückliefern.

5.4.2.2 HKPS

OpenPGP Zertifikate sollen über HKPS abgerufen werden können. Das Protokoll ist im RFC Draft *draft-shaw-openpgp-hkp-00.txt* beschrieben. [25] Da nur die öffentliche Schnittstelle des Systems genutzt werden darf, sind über die HKPS Schnittstelle nur folgende Funktionen verfügbar:

Abrufen von Zertifikaten Es können alle Zertifikate abgerufen werden, welche die Sichtbarkeitsstufe öffentlich haben.

Aktualisieren von Zertifikaten Aktualisierte Zertifikate können über die HKPS Schnittstelle an das System weitergereicht werden. Von den erhaltenen Daten werden aber ausschließlich neue Signaturen im System übernommen und wie unter 4.5 beschrieben verarbeitet.

5.5 Grafische Benutzerschnittstelle

Zur Verwendung des Systems durch Benutzer wird eine webbasierte grafische Benutzerschnittstelle angeboten. Sie basiert auf der in 5.4.1 definierten Programmierschnittstelle. Die verschiedenen Zugriffststufen der Schnittstelle sind durch die Authentifikation eines Benutzers erreichbar. Entsprechend der Rolle 5.3.1 stehen in der grafischen Schnittstelle die jeweiligen Funktionen zur Verfügung.

5.6 Synchronisationsmechanismus

Damit Zertifikate auch für Benutzer anderer Instanzen des Systems verfügbar sind, werden Zertifikate zwischen konfigurierten Instanzen synchronisiert. Sollen Zertifikate zwischen zwei Instanzen synchronisiert werden, so muss in beiden Instanzen die jeweils andere Instanz als Schwesterinstanz konfiguriert werden. Werden dann Änderungen an Zertifikaten vorgenommen, so wird diese Änderung an alle konfigurierten Schwesterinstanzen weitergegeben. Um alle Zertifikate zu erhalten, wird der komplette Bestand an Zertifikaten beim Start einer Instanz abgefragt.

Bei der Synchronisation der Zertifikate gibt es zwei Modi, welche ebenfalls konfiguriert werden können:

Öffentliche Synchronisation Hier werden ausschließlich Zertifikate synchronisiert, welche die Sichtbarkeitsstufe *öffentlich* haben.

Interne Synchronisation Zusätzlich zu Zertifikaten mit der Stufe *öffentlich* werden auch Zertifikate mit der Stufe *intern* synchronisiert.

5.7 Zusammenfassung

In diesem Kapitel wurde ein Systementwurf erstellt, der die Anforderungen an einen sicheren und die Privatsphäre schützenden Verzeichnisdienst erfüllt. Für die Authentifizierung der Zertifikate ist die in Abschnitt 5.3.2.2 beschriebene Komponente verantwortlich. Mit Hilfe von Authentifizierungstokens, welche an die E-Mail-Adressen eines Zertifikats geschickt werden, werden durch die Komponente im System abgelegte Zertifikate authentifiziert. Der Zertifikatsparser (5.3.2.1) wiederum ermöglicht den Schutz der Privatsphäre, indem die Zertifikate vor der Herausgabe entsprechend gesetzter Sichtbarkeitsstufen gefiltert werden. Zusätzlich wird über den Zertifikatsparser auch die Möglichkeit bereitgestellt, Signaturen zu OpenPGP Zertifikaten hinzuzufügen. Auf diese Weise können Aktualisierungen von Zertifikaten durch andere Benutzer separat gespeichert und vom Eigentümer individuell übernommen werden.

Kapitel 6

Implementierung

Zur Bewertung des Systementwurfs wird dieser implementiert. Die verwendeten Techniken und die Art der Umsetzung wird in den folgenden Abschnitten beschrieben. Dieses Kapitel dient zusammen mit Kapitel 5 als Dokumentation und kann als Grundlage für die Weiterentwicklung der Implementierung verwendet werden.

6.1 Vorgaben

Für die Implementierung gab es folgende Vorgaben:

Programmiersprache Als Programmiersprache wurde ursprünglich Java in der Version 1.5 festgelegt. Der Grund hierfür war die voraussichtliche spätere Verwendung des Systems innerhalb einer Oracle Datenbank über sogenannte *Stored Procedure Calls*. Das bedeutet, dass der Java Code von der Oracle Datenbank ausgeführt wird. Die dort verwendete Version der Java Virtual Machine entspricht der Version 1.5. Die Anforderung, dass die Implementierung mit den *Stored Procedures* kompatibel sein muss, wurde später während der Entwicklung verworfen.

Kryptobibliothek In verwandten Projekten an der TUM im Lehrstuhl für Netzarchitekturen und Netzdienste bestanden gute Erfahrungen mit der Kryptobibliothek Bouncy Castle. Aus diesem Grund sollte die Bibliothek bei entsprechender Eignung auch bei Umsetzung des Zertifikatverzeichnisdienstes Einsatz finden.

6.2 Java Spring

Aufgrund der vielen Möglichkeiten die das Spring Framework zur Entwicklung von Webservices bietet, wurde es als Basis für die Implementierung verwendet. [1] Die be-

[1] Informationen zu Spring sind unter http://spring.io/ zu finden.

nutzten Funktionen des Frameworks werden im Folgenden beschrieben.

Spring MVC Das Spring MVC [2] (Model View Controller) Framework bietet Funktionen zur Dependency Injection sowie zum Erstellen von Webapplikationen. Für die Umsetzung der Programmierschnittstelle kann mittels Spring MVC eine REST Schnittstelle (Representational State Transfer) erstellt werden. Basierend auf der REST Schnittstelle bietet Spring MVC viele unterstützende Funktionen zur Erstellung einer grafischen Benutzerschnittstelle. Spring MVC wird in der Version 4.2.4 verwendet.

Spring Boot Spring Boot [3] ermöglicht die einfache Erstellung eigenständiger Webapplikationen. Unterstützende Funktionen sind, neben integriertem Webserver, die Möglichkeit zur automatischen Konfiguration von Spring und die Konfiguration der Applikation über Property-Dateien. Spring Boot wird in Version 1.3.1 verwendet.

Spring Security Das Framework Spring Security [4] bietet in Kombination mit Spring MVC die Möglichkeit der Benutzerauthentifizierung. Für die Authentifizierung werden viele verschiedene Technologien unterstützt. Das Framework übernimmt für die Implementierung die in Abschnitt 5.3.1 beschriebene Funktionalität. Spring Security wird in Version 4.0.3 verwendet.

Spring Data JPA Das Projekt Spring Data Java Persistence API (JPA) [5] ermöglicht das persistieren von Objekten in JPA basierten Datenspeichern. In der Implementierung wird Spring Data JPA in Kombination mit MySQL zum Persistieren von Daten verwendet. Spring Data wird in Version 1.9.2 verwendet.

6.3 Bouncy Castle

Bouncy Castle (BC) [6] bietet viele kryptografische Funktionen. Darunter auch die Verarbeitung von S/MIME und OpenPGP Zertifikaten, sowie das Verschlüsseln und Entschlüsseln mit Zertifikaten. Mit BC lassen sich auch Benutzerkennungen und Signaturen von OpenPGP Zertifikaten bearbeiten. Mit diesen Voraussetzungen, ist das Authentifizieren der Zertifikate (vgl. 5.3.2.2) und das Einschränken des Zugriffs auf OpenPPG Zertifikate (vgl. 5.2.3) wie beschrieben möglich. Für die Implementierung wird daher BC in der Version 1.54 zum Verarbeiten von S/MIME und OpenPGP Zertifikaten benutzt.

[2] Projektseite für Spring MVC: http://projects.spring.io/spring-framework/
[3] Projektseite für Spring Boot: http://projects.spring.io/spring-boot/
[4] Projektseite für Spring Security: http://projects.spring.io/spring-security/
[5] Projektseite für Spring Data JPA: http://projects.spring.io/spring-data-jpa/
[6] Projektseite für Bouncy Castle: http://bouncycastle.org/

6.4 Projektüberblick

Das Projekt gliedert sich in die folgenden vier Module:

cms-ds-certificates Dieses Modul enthält benötigte Funktionalität bezüglich OpenPGP und S/MIME Zertifikaten, die auf BC basieren. Darunter ist zum Beispiel das Verschlüsseln von Nachrichten mit Hilfe eines Zertifikats oder die Bearbeitung von OpenPGP Zertifikaten.

cms-ds-persistence Das Persistenzmodul enthält die Datenstrukturen und eine Anbindung an eine Datenbank als persistierenden Speicher. Die Umsetzung der Anbindung basiert auf Spring Data JPA.

cms-ds-service Das Servicemodul enthält die eigentliche Funktionalität des Zertifikatsverzeichnisdienstes. Die Klasse CertificateManager dient als Schnittstelle für alle angebotenen Funktionen des Dienstes.

cms-ds-web Das Webmodul stellt über Spring MVC neben einer REST Schnittstelle, welche auf die Klasse CertificateManager aufbaut, eine Weboberfläche bereit. Über die REST Schnittstelle erfolgt auch die Authentifizierung am System. Die Art der Authentifizierung wird über Spring Security konfiguriert.

Die beschriebenen Module und Technologien lassen sich, wie in Abbildung 6.1, auf den in Kapitel 5 beschriebenen Systementwurf abbilden.

6.5 Umgesetzte Funktionen

Zur Entwicklung wurde der Ansatz des vertikalen Protyping gewählt, um die implementierten Funktionen in vollem Umfang bewerten zu können. Im Folgenden werden die für die Implementierung umgesetzten Funktionen aufgelistet. Die umgesetzten Funktionen sind soweit nicht anders beschrieben auf die Nutzerrolle *Benutzer* (siehe 5.3.1) beschränkt. Abbildung 6.2 zeigt die Startseite Zertifikatverzeichnisdienstes für einen authentifizierten Benutzer. Wie die Verwendung der einzelnen Funktionen möglich ist, wird mit Hilfe der Abbildung beschrieben.

Speichern von Zertifikaten Das Hinzufügen neuer Zertifikate ist im unteren Bereich *Add certificate* möglich. Dafür muss zuerst die Art des Zertifikats ausgewählt werden. Danach muss die Datei, welche das Zertifikat enthält ausgewählt werden. Über *Submit* wird das Zertifikat vom System gespeichert. Ins System hochgeladene Zertifikate sind im Bereich darüber (*My certificates*) in Tabellenform dargestellt.

Sichtbarkeit von Zertifikaten Die Sichtbarkeit eines Zertifikats kann in der Tabelle in der Spalte *Visibility* eingestellt werden. Für die Implementierung wurden hierfür die Sichtbarkeitsstufen *privat* (PRIVATE), *intern* (INTERNAL) und *öffentlich*

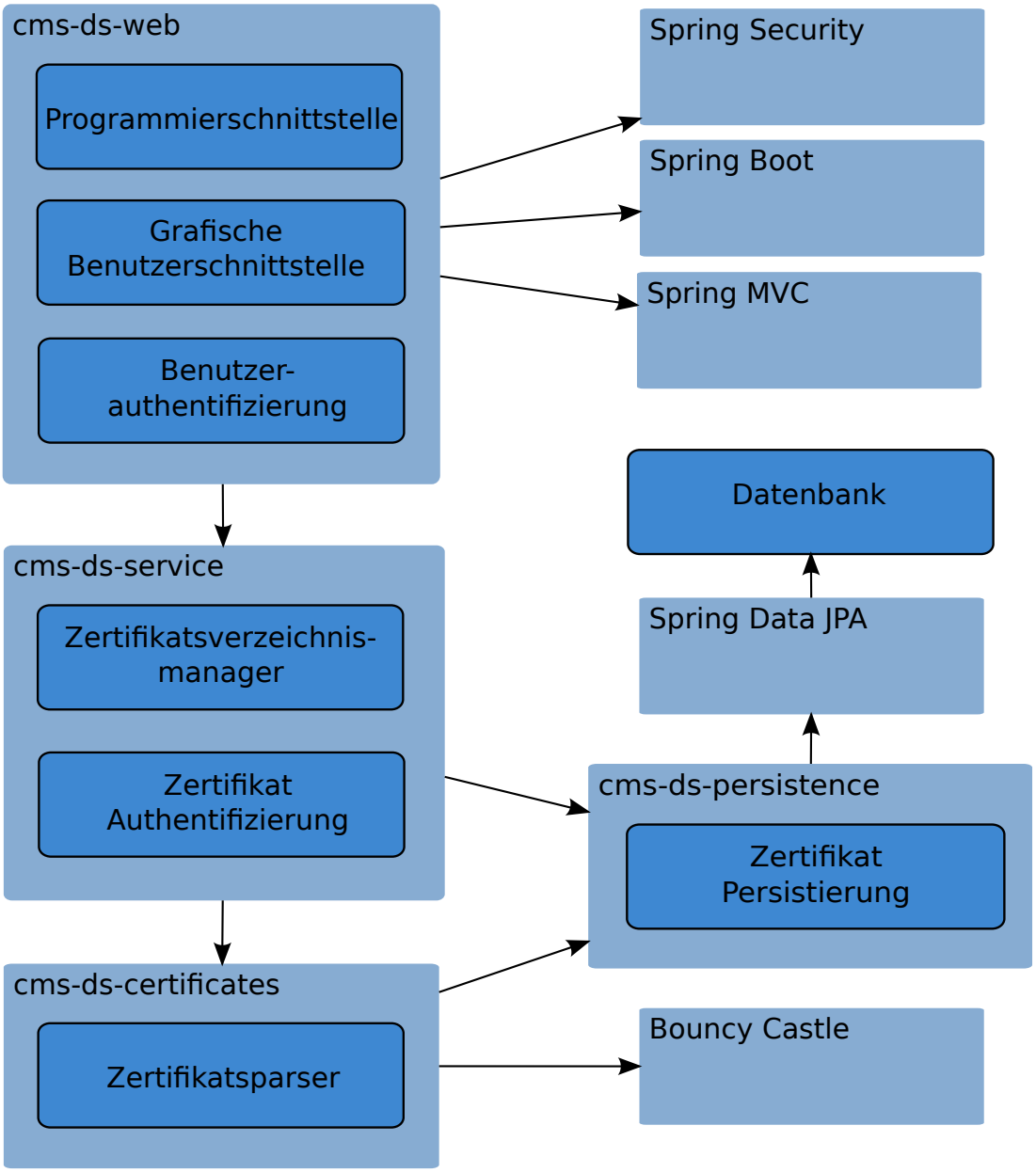

Abbildung 6.1: Ein Überblick über die Module mit Zuordnung zu im Systementwurf definierten Komponenten des Systems. Abhängigkeiten zwischen Modulen und verwendeten Technologien sind durch Pfeile dargestellt.

Abbildung 6.2: Startseite Zertifikatverzeichnisdienstes für authentifizierten Benutzer

(PUBLIC) umgesetzt (vgl. 5.2.2). Die benutzerdefinierte Sichtbarkeitsstufe steht noch nicht zur Verfügung. Die Auswahl einer Sichtbarkeitsstufe wird sofort im System übernommen.

Authentifizierung von Zertifikaten Mit Zertifikaten lassen sich verschiedene Aktionen durchführen. Diese Aktionen sind in der Tabelle rechts unter *Actions* zu finden. Die Authentifizierung eines neuen Zertifikats ist über das Umschlagsymbol zu erreichen. Dort muss für alle E-Mail-Adressen der Authentifizierungstoken eingegeben werden. Dieser wird per E-Mail an die jeweilige Adresse versandt. Alternativ kann die Authentifizierung auch direkt über den in der E-Mail enthaltenen Link erreicht werden.

Sichtbarkeit von Benutzerkennungen Bei OpenPGP Zertifikaten können auch Sichtbarkeitsstufen für die einzelnen Benutzerkennungen festgelegt werden. Diese Funktion ist über das dann verfügbare Zahnradsymbol erreichbar. Die möglichen Sichtbarkeitsstufen sind ebenso *privat*, *intern* und *öffentlich*.

Aktualisieren von Zertifikaten Die Aktualisierung eines Zertifikats kann über das Symbol mit Pfeil nach oben gestartet werden. Im Anschluss muss die Datei des aktualisierten Zertifikats ausgewählt werden. Über *Submit* wird das Zertifikat aktualisiert.

Herunterladen von Zertifikaten Das Herunterladen eines Zertifikats wird über das Symbol mit Pfeil nach unten gestartet. Der Browser speichert dann das Zertifikat am gewünschten Speicherort ab.

Löschen von Zertifikaten Über das X kann ein Zertifikat gelöscht werden. Alle das Zertifikat betreffenden Daten werden dann aus dem System gelöscht.

Suchen von Zertifikaten Über den Menüeintrag *Search* wird die Suche nach Zertifikaten geöffnet. Nach der Eingabe der Suchwörter wird die Suche durch die

Entertaste gestartet.

6.6 REST Schnittstelle

In diesem Abschnitt ist die implementierte REST Schnittstelle beschrieben. Werden von der Schnittstelle Informationen zu Zertifikaten zurückgegeben, so erfolgt dies über das folgende JSON-Modell eines Zertifikats.

```
[
    {
        "type":<Typ>,
        "ownerAccount":<Benutzername>,
        "id":<Zertifikat_ID>,
        "state":<Status>,
        "visibilityLevel":<Sichtbarkeitsstufe>,
        "userIdsMap":{
            <Benutzerkennung>:{
                "visibilityLevel":<Sichtbarkeitsstufe>,
                "authenticated":<true|false>
                "signatures":{
                  "id":<Signatur_ID>
                  "visibilityLevel":<Sichtbarkeitsstufe>
                }
            }
        }
    }
]
```

Tabelle 6.1 beschreibt die über die REST Schnittstelle bereitgestellten Funktionen.

6.7 Installation

Für die Implementierung wurde das Build-Tool Maven verwendet, um eine einfache Installation zu ermöglichen.

Zur Installation muss Maven in der Versiuon 3.3.1 oder höher installiert sein. Nach dem auschecken des Git-Repositories kann der Zertifikatverzeichnisdienst über den Befehl

```
mvn package
```

Tabelle 6.1: REST Schnittstelle des Systems

Pfad	Parameter	Methode	Rückgabe
/login	username, password	POST	-
Meldet einen Benutzer am System an. Im Anschluss können entsprechend der Rolle des Benutzers alle Funktionen genutzt werden.			
/logout	-	POST	-
Meldet einen Benutzer vom System ab.			
/getUserCertificates	user	GET	application/json: Liste von Zertifikaten.
Gibt alle Zertifikate des angegebenen Benutzers zurück.			
/search	searchString	POST	JSON: Liste von Zertifikaten.
Gibt alle Zertifikate zurück, welche zum Suchtext passen.			
/addCertificate	type, file	POST	-
Speichert ein Zertifikat des angegebenen Typs im System. Der im Moment angemeldete Benutzer wird als Besitzer festgelegt.			
/updateCertificate	id, file	POST	-
Ersetzt das Zertifikat mit der gegebenen Id durch die neue Zertifikatsdatei.			
/setCertificateVisibility	id, visibility	GET	-
Setzt die Sichtbarkeitsstufe des Zertifikats mit der gegebenen Id. Visbility kann einen der folgenden Werte enthalten: PRIVATE, INTERNAL, PUBLIC			
/setUserIdVisibility	id, userId, visibility	POST	-
Setzt die Sichtbarkeitsstufe der übergebenen Benutzerkennung des Zertifikats mit der gegebenen Id. Visbility kann einen der folgenden Werte enthalten: PRIVATE, INTERNAL, PUBLIC			
/deleteCertificate	id	POST	-
Löscht das Zertifikat mit der gegebenen Id.			
/downloadCertificate	id	GET	application/octet-stream
Gibt das Zertifikat mit der gegebenen Id als asc-Datei (OpenPGP) oder pem-Datei (S/-MIME) zurück.			
/authenticateUserId	id, userId, token	POST	-
Authentifiziert die Benutzerkennung des Zertifikats mit der gegebenen Id unter Verwendung des Authentifizierungstokens.			

aus dem Wurzelverzeichnis des Repositories heraus gebaut werden. Zum Ausführen des Zertifikatverzeichnisdienstes muss der folgende Befehl gestartet werden:

```
java -jar cms-ds-web/target/cms-ds-web-1.0-SNAPSHOT.jar
```

In diesem Fall speichert der Dienst alle anfallenden Daten im Hauptspeicher. Beim Beenden des Dienstes gehen dann alle gespeicherten Daten verloren. Alternativ kann auch das Spring Data JPA mitgestartet werden, um eine vorhandene Datenbank als persistenten Speicher zu verwenden. Hierfür wird beim Start einfach ein Parameter angehängt:

```
java -jar cms-ds-web/target/cms-ds-web-1.0-SNAPSHOT.jar
    -Dspring.profiles.active=database
```

Die Konfiguration des Dienstes wird im nächsten Abschnitt detailliert beschrieben.

6.8 Konfiguration

Der Zertifikatverzeichnisidenst wird über zwei Property-Dateien konfiguriert. Die Datei *datastore.properties* erlaubt die Konfiguration der zu verwendenden Datenbank, wenn der Dienst mit dem Spring Profil *database* gestartet wurde (siehe 6.7). Die Datei *cms-ds.properties* ermöglicht das Konfigurieren des Dienstes hinsichtlich verschiedener weiterer Aspekte.

6.8.1 cms-ds.properties

Über die *cms-ds.properties* können die Einstellungen für die Webapplikation vorgenommen werden. Im folgenden werden die möglichen Optionen erläutert.

Diese Optionen legen die maximale Größe der Http Anfragen fest. Dadurch begrenzen sie die maximale Größe, die ein Zertifikat haben darf.

```
multipart.maxFileSize=16MB
multipart.maxRequestSize=16MB
```

Diese Optionen ermöglichen die Konfiguration des E-Mail-Servers, der für das Versenden der Authentifizierungsmails verwendet wird. Die dargestellten Optionen sind für die Konfiguration eines SMTP Servers ausreichend. Weitere und detaillierter Informationen sind in der Dokumenation der Java Mail API unter https://javamail.java.net/nonav/docs/api/ zu finden.

```
#email configuration
mail.username=
```

```
mail.password=
mail.smtp.host=localhost
mail.smtp.auth=false
mail.smtp.starttls.enable=true
mail.smtp.port=587
```

Der Inhalt der Authentifizierungsmails kann über die nachstehenden Optionen konfiguriert werden.

```
#email content
directoryservice.authentication.email.template=
directoryservice.authentication.email.subject=
directoryservice.authentication.email.sender-address=
directoryservice.authentication.email.sender-name=
directoryservice.hostname=localhost
```

Das Template definiert den Inhalt der Authentifizierungsmails. Der Token wird an Stelle des Platzhalters *<token>* eingetragen. Auf die gleiche Weise wird *<email>* durch die zu authentifizierende E-Mail-Adresse und *<link>* durch den Authentifizierungslink ersetzt. Die Angabe bei der Option *directoryservice.hostname* ist wichtig, da der hier angegebene Hostname im Authentifizierungslink verwendet wird.

Der Webserver kann über die folgenden Optionen konfiguriert werden.

```
server.port=8443
server.port.http=80
server.ssl.key-store=classpath:keystore.p12
server.ssl.key-store-password=cms-ds
server.ssl.keyStoreType=PKCS12
```

server.port ist der Port unter welchem der Verzeichnisdienst erreichbar sein soll. Die Verbindung wird mit dem Zertifikat verschlüsselt, welches im angegebenen Keystore abgelegt ist. Zusätzlich müssen das Passwort für den Keystore und der Typ des zu verwendenden Keystores angegeben werden. Die Option *server.port.http* macht den HTTP Port der Webapplikation konfigurierbar. Anfragen an diesen Port werden umgeleitet an den Port *server.port*, über welchen eine verschlüsselte Kommunikation stattfindet.

6.8.2 datastore.properties

Die *datastore.properties* ermöglicht das Konfigurieren der zu verwendenden Datenbank:

```
spring.datasource.url = jdbc:mysql://localhost:1337/cms-ds
spring.datasource.username = cms-ds
```

```
spring.datasource.password = cms-ds
```

Die URL gibt den Pfad zur Datenbank und die Art Datenbank an. Im Beispiel handelt es sich um eine MySQL Datenbank, wobei das zu verwendende Schema *cms-ds* heißt. Der benötigte Benutzername und das Passwort müssen ebenfall angegeben werden.

Die Option *show-sql* ermöglicht das Ein- und Ausschalten von Logeinträgen beim Ausführen von SQL Befehlen.

```
spring.jpa.show-sql = true
```

Damit die Datenbank entsprechend des mit Hilfe von Spring Data JPA definierten Modells angelegt werden kann, müssen dem konfigurierten Benutzer die folgenden Zugriffsrechte auf das konfigurierte Schema gewährt werden:

- Select, Insert, Update und Delete für das Speichern von Daten.
- Create, Alter, References und Drop für das initiale Erstellen des Datenbank Modells.

Weitere Konfigurationsmöglichkeiten und Details sind unter https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-sql.html zu finden.

6.9 Zusammenfassung

Die wichtigen Funktionen, die den Zertifikatsverzeichnisdienst zu einem sicheren und die Privatsphäre schützenden System machen, konnten mit Hilfe der Kryptographiebibliothek BC umgesetzt werden:

Authentifizierung von Zertifikaten Die Authentifizierung erfolgt durch das Versenden von Authentifizierungstokens an die entsprechenden E-Mail-Adressen, die mit Hilfe von BC mit dem jeweiligen Zertifikat verschlüsselt sind. Die erforderliche Eingabe der Tokens stellt die Authentizität des Zertifikats sicher.

Schutz der Privatsphäre Der Zugriff auf Zertifikate wird über die Sichtbarkeitsstufen eingeschränkt. Im Fall von OpenPGP werden mit Hilfe von BC, entsprechend der Sichtbarkeitsstufen, einzelne persönliche Daten aus den Zertifikaten gefiltert. BC ermöglicht auch das Übernehmen einzelner Signaturen, die von anderen Benutzern bereitgestellt wurden.

Kapitel 7

Diskussion und Bewertung der Ergebnisse

Die ursprüngliche Motivation für das Design und den Entwurf des Prototypen für einen Zertifikatsverzeichnisdienst ist die Verbesserung der Verfahren OpenPGP und S/MIME hinsichtlich zweier Aspekte: Privatsphäre und Sicherheit. In den folgenden Abschnitten werden die Ergebnisse aus den vorangehenden Kapiteln anhand dieser Aspekte bewertet.

7.1 Privatsphäre

Ein Ziel dieser Arbeit ist die Verbesserung der Privatsphäre. Der entworfene Zertifikatsverzeichnisdienst setzt beim Schlüsselaustausch an, um Einfluss auf den Grad der Privatsphäre zu nehmen.

7.1.1 Verschlüsselte Kommunikation

Beim in dieser Arbeit entwickelten Zertifikatsverzeichnisdienst wird die Verbindung mit dem Zertifikatsverzeichnisdienst verschlüsselt. Dies erhöht die Privatsphäre hinsichtlich Außenstehender, welche die Netzwerkkommunikation mitlesen. Das Verschlüsseln der Verbindung verhindert die folgenden Punkte:

Erkennen von Beziehungen zwischen Personen Anhand der Abrufe von Zertifikaten könnte ein Außenstehender erkennen, welche Personen untereinander in Kontakt treten. Die Verschlüsselung der Verbindung verhindert das.

Abgreifen persönlicher Daten Weiterhin könnten anhand mitgelesener Anfragen persönliche Daten abgegriffen werden. Vor allem persönliche Daten, welche in den übertragenen Zertifikaten enthalten sind, wären dann für den Mithörer sichtbar. Durch die Verschlüsselung der Kommunikation wird auch dieses Szenario verhindert.

7.1.2 Zugriffskontrolle für Zertifikate

Ein weiterer Punkt, welcher die Privatsphäre schützen soll, ist die Möglichkeit den Zugriff auf Zertifikate einzuschränken. Hierfür bestehen im entworfenen Zertifikatsverzeichnisdienst zwei Mechanismen zur Verfügung: Das Einschränken der Sichtbarkeit sowie das Löschen von Zertifikaten. Die genannten Mechanismen schützen die Privatsphäre hinsichtlich folgender Punkte:

Sozialer Graph Der Zugriff auf viele OpenPGP Zertifikate ermöglicht aufgrund des Konzepts des WoT das Erstellen eines sozialen Graphen. Durch die Möglichkeit, die Sichtbarkeit einzelner Signaturen eines OpenPGP Zertifikats einzuschränken, kann dies erschwert oder sogar verhindert werden. Enthalten die Zertifikate keine Signaturen mehr, ist das Herstellen von Verbindungen zwischen verschiedenen Zertifikaten nicht mehr möglich.

Persönliche Daten Die vielen in Zertifikaten enthaltenen persönlichen Informationen sind bei herkömmlichen Methoden zur Veröffentlichung von Zertifikaten meist ungeschützt und für jeden abrufbar. Zusätzlich ist die Möglichkeit, die einmal veröffentlichten Daten wieder zu löschen nur in wenigen Fällen gegeben. Das Einschränken der Sichtbarkeit von Zertifikaten erhöht die Kontrolle über die persönlichen Daten. Zusammen mit der Funktion Zertifikate aus dem Zertifikatsverzeichnisdienst entfernen zu können, ist es möglich den Zugriff auf persönliche Daten auch im Nachhinein einzuschränken. Vor allem die Struktur von OpenPGP Zertifikaten ermöglicht auf diese Weise eine feingradige Kontrolle über die persönlichen Daten.

Aktualisierung von Zertifikaten OpenPGP ermöglicht auch anderen Benutzern das Aktualisieren von Zertifikaten, beispielsweise bei Signaturen für Benutzerkennungen. Der entwickelte Dienst verhindert ungewollte Änderungen durch andere Personen. Signaturen können weiterhin durch andere Benutzer hinzugefügt werden, allerdings steht dem Eigentümer des Zertifikats die Entscheidung zu, ob die Signatur veröffentlicht oder entfernt wird.

7.1.3 Bewertung

Der entworfene Zertifikatsverzeichnisdienst verbessert im Vergleich zu anderen Lösungen den Schutz der Privatsphäre hinsichtlich der Kontrolle über persönliche Daten. Ungeachtet dessen bleiben aber andere, die Privatsphäre betreffende Probleme bestehen: Die eigentliche E-Mail-Kommunikation, welche nach dem Schlüsselaustausch erfolgt, eröffnet Außenstehenden weiterhin die Möglichkeit persönliche Daten abzugreifen. Das liegt unter anderem daran, dass nur der Nachrichtenteil der E-Mail verschlüsselt wird. Unverschlüsselt bleiben beispielsweise der Betreff der E-Mail, sowie Absender und Empfänger. Zum Lösen dieser Probleme muss an anderer Stelle angesetzt werden.

7.2 Sicherheit

7.2.1 Authentifizierung von Zertifikaten

Das zweite Ziel der Arbeit geht Probleme bei der Sicherheit der Verfahren OpenPGP und S/MIME an. Hierfür werden die im Zertifikatsverzeichnisdienst gespeicherten Zertifikate authentifiziert, indem sichergestellt wird, dass der Besitzer des Zertifikats auch der Eigentümer ist. Bei den meisten herkömmlichen Lösungen zur Veröffentlichung von Zertifikaten, findet keine Authentifizierung der Zertifikate statt. Das im entworfenen Dienst eingesetzte Verfahren der Authentifizierungstoken, welche an die E-Mail-Adressen der Zertifikate versandt werden, sichert die Vertraulichkeit der Kommunikation.

Auf diese Weise kann verhindert werden, dass falsche Zertifikate die sichere Kommunikation gefährden.

7.2.2 Bewertung

Mit Hilfe des entworfenen Zertifikatsverzeichnisdienstes, kann die Vertraulichkeit der Kommunikation über OpenPGP oder S/MIME zusätzlich abgesichert werden. Dies geschieht, indem beim Schlüsselaustausch die Authentizität des Schlüsselsmaterials gewährleistet wird.

Eine Absicherung ist allerdings nicht garantiert, sofern ein Dritter Zugriff auf die E-Mail-Konten des betroffenen Zertifikats hat. Abgeschwächt wird das Problem nur dadurch, dass eine Authentifizierung auch am Zertifikatsverzeichnisdienst erforderlich ist.

Kapitel 8

Zusammenfassung

Das Ziel der Arbeit, die Realisierung eines Zertifikatsverzeichnisdienstes, welcher die Authentizität von Zertifikaten sicherstellt und die Privatsphäre der Beteiligten Benutzer schützt, konnte umgesetzt werden.

Anhand der untersuchten Probleme aktueller Verzeichnisdienste wurden die Anforderungen an einen sicheren und die Privatsphäre schützenden Zertifikatsverzeichnisdienst erarbeitet und darauf aufbauend ein Systementwurf entwickelt, der diese Anforderungen erfüllt. Abschließend wurde gezeigt, wie der Systemenwurf umgesetzt werden kann.

8.1 Ausblick

Das Thema sichere Kommunikation wird auch in Zukunft ein wichtiges Thema sein. Aus diesem Grund ist es wichtig, dass die Verfahren für sichere E-Mail-Kommunikation weiterhin verbessert oder neue Verfahren entwickelt werden. Vor allem die Sicherstellung der Authentizität von Zertifikaten ist von großer Bedeutung. Hierfür bietet sich die Untersuchung anderer, in dieser Arbeit nicht bearbeiteten Methoden, wie beispielsweise die Verwendung des neuen elektronischen Personalausweises zur Authentifizierung an.

Die Privatsphäre der Benutzer spielt ebenso eine wichtige Rolle. In dieser Arbeit wurde dargestellt, wie die Privatsphäre der Benutzer von Verzeichnidiensten für den Austausch von Zertifikaten geschützt werden kann. Dieser Schutz muss noch weiter ausgebaut werden, indem auch weitere Teile der Zertifikate durch Einschränkung des Zugriffs geschützt werden können. Darunter fallen zum Beispiel der Schutz von Adressdaten oder Bilder, die in dieser Arbeit nicht betrachtet wurden.

Eines der größten Probleme sicherer E-Mail-Kommunikation ist die geringe Benutzerfreundlichkeit aktueller Lösungen. Diese Arbeit leistet in dieser Hinsicht nur einen

geringen Beitrag. Damit sichere E-Mail-Kommunikation stärkere Verbreitung findet, ist es wichtig die Benutzerfreundlichkeit zu erhöhen.

Es ist offensichtlich, dass noch viele Aspekte durch zukünftige Arbeiten untersucht werden können, um die Verfahren für sichere Kommunikation zu verbessern.

Literaturverzeichnis

[1] P. Beuth, “Snowden Enthüllungen: Alles wichtige zum NSA Skandal,” 2013. [Online]. Available: http://www.zeit.de/digital/datenschutz/2013-10/hintergrund-nsa-skandal

[2] S. Ruoti, J. Andersen, D. Zappala, and K. E. Seamons, “Why johnny still, still can’t encrypt: Evaluating the usability of a modern PGP client,” *CoRR*, vol. abs/1510.08555, 2015. [Online]. Available: http://arxiv.org/abs/1510.08555

[3] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,” RFC 5280 (Proposed Standard), Internet Engineering Task Force, May 2008, updated by RFC 6818. [Online]. Available: http://www.ietf.org/rfc/rfc5280.txt

[4] P. Feisthammel, “Explanation of the web of trust of pgp,” 2004. [Online]. Available: http://www.rubin.ch/pgp/weboftrust.en.html

[5] P. Resnick, “Internet Message Format,” RFC 5322 (Draft Standard), Internet Engineering Task Force, Oct. 2008, updated by RFC 6854. [Online]. Available: http://www.ietf.org/rfc/rfc5322.txt

[6] M. Elkins, D. D. Torto, R. Levien, and T. Roessler, “MIME Security with OpenPGP,” RFC 3156 (Proposed Standard), Internet Engineering Task Force, Aug. 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3156.txt

[7] J. Eberspächer, *Sichere Daten, sichere Kommunikation / Secure Information, Secure Communication*. Springer, 1994.

[8] Network Associates, Inc., Ed., *An Introduction to Cryptography*. [Online]. Available: ftp://ftp.pgpi.org/pub/pgp/6.5/docs/english/IntroToCrypto.pdf

[9] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer, “OpenPGP Message Format,” RFC 4880 (Proposed Standard), Internet Engineering Task Force, Nov. 2007, updated by RFC 5581. [Online]. Available: http://www.ietf.org/rfc/rfc4880.txt

[10] B. Ramsdell and S. Turner, “Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification,” RFC 5751 (Proposed Standard),

Internet Engineering Task Force, Jan. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5751.txt

[11] Microsoft Corporation, Ed., *Understanding S/MIME*. [Online]. Available: https://technet.microsoft.com/en-us/library/aa995740%28v=exchg.65%29.aspx

[12] Zytrax Inc., Ed. [Online]. Available: http://www.zytrax.com/tech/survival/ssl.html

[13] K. Zeilenga, "Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map," RFC 4510 (Proposed Standard), Internet Engineering Task Force, Jun. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4510.txt

[14] C. Kirsch, "S/MIME vs. OpenPGP: Eine Entscheidungshilfe," 2001. [Online]. Available: http://2014.kes.info/archiv/online/01-01-60-SMIMEvsOpenPGP.htm

[15] F. Hammerl, "Decentralized Key Servers Are Dead," 2015. [Online]. Available: https://blog.whiteout.io/2015/07/10/trust-in-a-decentralized-system/

[16] O. Williams, "Google to drop China's CNNIC Root Certificate Authority after trust breach," 2015. [Online]. Available: http://thenextweb.com/insider/2015/04/02/google-to-drop-chinas-cnnic-root-certificate-authority-after-trust-breach/#gref

[17] "secure-email - Overview of projects working on next-generation secure email." 2015. [Online]. Available: http://youbroketheinternet.org/secure-email

[18] C. Strotmann, "DNSSEC und DANE: Hilfestellung zur Mail-Verschlüsselung," 2015. [Online]. Available: http://www.heise.de/netze/artikel/DNSSEC-und-DANE-Hilfestellung-zur-Mail-Verschluesselung-2619026.html

[19] H. Böck, "Bessere Transportverschlüsselung zwischen Mail-servern," 2016. [Online]. Available: http://www.golem.de/news/smtp-sts-bessere-transportverschluesselung-zwischen-mailservern-1603-119955.html

[20] M. Ermert, "IETF-Tagung: Neue Vorschläge zum Sichern des Mail-transports," 2016. [Online]. Available: http://www.heise.de/netze/meldung/IETF-Tagung-Neue-Vorschlaege-zum-Sichern-des-Mailtransports-3163818.html

[21] PGP Corporation, Ed., *Global Directory FAQ*. [Online]. Available: https://keyserver.pgp.com/vkd/VKDHelpPGPCom_de.html

[22] M. Schanzenbach, "The GNU Name System," 2012. [Online]. Available: https://gnunet.org/gns

[23] H. Kalodner, M. Carlsten, P. Ellenbogen, J. Bonneau, and A. Narayanan, "An empirical study of Namecoin and lessons for decentralized namespace design," 2015. [Online]. Available: http://www.econinfosec.org/archive/weis2015/papers/WEIS_2015_kalodner.pdf

[24] K. Zeilenga, “Lightweight Directory Access Protocol (LDAP) Schema Definitions for X.509 Certificates,” RFC 4523 (Proposed Standard), Internet Engineering Task Force, Jun. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4523.txt

[25] D. Shaw, “The OpenPGP HTTP Keyserver Protocol (HKP),” 2003. [Online]. Available: http://tools.ietf.org/html/draft-shaw-openpgp-hkp-00