

Masterarbeit

**Mehrparteien-Signaturen für elektronische
Dokumente im Bereich der Telemedizin**

Jannic Hartwecker
Dezember 2013

Gutachter:

Prof. Dr. Jan Jürjens

Dr. Wolfgang Deiters

Technische Universität Dortmund
Fakultät für Informatik
Lehrstuhl 14
<http://ls14-www.cs.tu-dortmund.de>

In Kooperation mit:
Fraunhofer-Institut für Software- und
Systemtechnik ISST
<http://www.isst.fraunhofer.de>

Eidesstattliche Versicherung

Name, Vorname

Matr.-Nr.

Ich versichere hiermit an Eides statt, dass ich die vorliegende Bachelorarbeit/Masterarbeit* mit dem Titel

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Ort, Datum

Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -)

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird gfls. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin“) zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Ort, Datum

Unterschrift

Abstract

The aim of this master's thesis is the proposal of a multi-party contract signing system that allows multiple participants to sign an electronic document collaboratively. Therefore a multi-party contract signing protocol is proposed, which generates a complete signature for a given document, only if all participants have signed their signature share. Even one missing signature share should prevent the creation of a complete signature. This last requirement rules out any method, where the participants sign one after another and distribute the partial signed electronic document. As an additional countermeasure for fraud and attacks, the proposed system should facilitate biometric features. Out of the possible biometric features, in this thesis biometric handwritten signatures will be used. The proposed system should be usable to sign electronic protocols of teleconferences held with HealthTelKon.

Zusammenfassung

In dieser Masterarbeit wird ein Mehrparteien-Signatursystem entworfen, welches es mehreren Teilnehmern erlaubt, gemeinschaftlich elektronische Dokumente zu signieren. Dazu soll ein Mehrparteien-Signaturprotokoll entworfen werden, das nur dann eine Signatur für ein Dokument erzeugt, wenn alle Parteien ihren Signaturanteil generiert haben. Fehlt auch nur ein einziger Signaturanteil, kann keine Gesamtsignatur erzeugt werden. Diese letzte Anforderung schließt somit Verfahren aus, bei denen die Parteien ein Dokument nacheinander signieren und elektronisch versenden. Denn hier würden gültige Einzelsignaturen vor Beendigung des Signaturverfahrens vorliegen. Als zusätzlichen Schutz, soll das zu entwerfende System zusätzlich durch Biometrie gegen Angriffe und Fälschungen geschützt werden. Als biometrisches Merkmal sollen hierfür handgeschriebene biometrische Unterschriften genutzt werden. Das entworfene System soll genutzt werden können, um Protokolle von Telekonferenzen der Plattform HealthTelKon zu signieren.

Inhaltsverzeichnis

1	Einleitung	1
1.1	Motivation	1
1.2	Zielsetzung	2
1.3	Gliederung	2
2	Grundlagen	5
2.1	Kryptografie	5
2.1.1	Symmetrische Kryptosysteme	5
2.1.2	Asymmetrische Kryptosysteme	6
2.1.3	Hashverfahren	6
2.1.4	Digitale Signaturen	7
2.2	Rahmenbedingungen in Deutschland	7
2.2.1	Gesetzliche Regelungen und Anforderungen für elektronische Signaturen	8
2.2.2	Wichtige Begriffe der Informationssicherheit	11
2.2.3	Sichere Algorithmen nach dem Signaturgesetz der Bundesrepu- blik Deutschland	12
2.2.4	Gesundheitstelematik in Deutschland	14
2.2.5	Arten des Signaturerstellungsprozesses	15
2.2.6	Braunschweiger Regeln zur Archivierung mit elektronischen Signaturen im Gesundheitswesen	17
2.3	HealthTelKon	19
2.3.1	Ablauf einer Konferenz	20
2.3.2	Technische Details	20
2.3.3	Protokollierung	20
3	Verwandte Arbeiten	23
3.1	Biometrie	23
3.1.1	Biometrische Merkmale	24
3.1.2	Probleme beim Umgang mit Biometrie	25
3.1.3	Leistungsmetriken	26
3.1.4	Templates	27

Inhaltsverzeichnis

3.1.5	Biometrische Unterschriften	28
3.1.6	Biometrisches Matching	29
3.1.7	Hardware	31
3.2	Secret-Sharing	32
3.2.1	Shamir's Secret Sharing	32
3.3	Mehrparteien-Signaturen	33
3.3.1	Multi-Party Contract Signing	33
3.4	Fazit	34
4	Anforderungsanalyse	37
4.1	Problemstellung	37
4.2	Funktionale Anforderungen	37
4.3	Nicht-funktionale Anforderungen	39
4.4	Anwendungsfälle	41
5	Entwurf	49
5.1	Entwurf der Systemarchitektur	49
5.1.1	Desktopanwendung versus Webanwendung	49
5.1.2	Webservice Technologien	52
5.1.3	Entwurf der Komponenten	55
5.2	Protokolle	61
5.2.1	Mehrparteien-Signaturprotokoll	61
5.2.2	Biometrische Schlüsselfreigabe	65
5.2.3	Biometrischer Abgleich mit Dynamic Time Warping (DTW)	69
5.3	Interaktion mit HEALTHTELKON	71
5.3.1	Automatisierte Bereitstellung	71
5.3.2	Status einer Mehrparteien-Signatur	72
6	Prototypische Implementierung	73
6.1	Verwendete technische Hilfsmittel für die Implementierung	73
6.1.1	Systemkonfiguration	73
6.1.2	Werkzeuge	73
6.1.3	Wacom Tablet	75
6.2	Implementierung der Komponenten	77
6.2.1	HYDrOUS Business Logic	77
6.2.2	HYDrOUS Biometric Matcher	82
6.2.3	HYDrOUS Biometric Signature Extension	83
6.3	Probleme während der Implementierung	84
6.3.1	WACOM Low Level SDK	84
6.3.2	JAVA Applet	84

7	Evaluation	85
7.1	Biometrischer Abgleich	85
7.2	Erfüllung der funktionalen und nicht-funktionalen Anforderungen . . .	87
7.3	Angriffe	88
7.3.1	Brute-Force Angriff	88
7.3.2	Man-in-the-Middle-Angriff	89
7.3.3	Replay-Angriff	89
7.3.4	Perfect-Forward-Secrecy	90
7.3.5	Kompromittierung von Benutzer PCs	92
7.3.6	Kompromittierung der Server	93
8	Zusammenfassung	95
9	Ausblick	97
10	Abbildungsverzeichnis	99
11	Quelltextverzeichnis	101
12	Tabellenverzeichnis	103
13	Literaturverzeichnis	105

Inhaltsverzeichnis

Abkürzungsverzeichnis

ARGE Arbeitsgemeinschaft

ASR Automatic Speech Recognition

BGB Bürgerliches Gesetzbuch

BMWi Bundeswirtschaftsministerium

BSI Bundesamt für Sicherheit in der Informationstechnik

CCESigG Competence Center für die Elektronische Signatur im Gesundheitswesen
e.V.

CRUD Create Read Update Delete

CSS Cascading Style Sheets

DTW Dynamic Time Warping

ECMAScript European Computer Manufacturers Association Script

EER Equal Error Rate

EFA ELEKTRONISCHE FALLAKTE

EGK elektronische Gesundheitskarte

FAR False Acceptance Rate

FRR False Rejection Rate

FTP File Transfer Protocol

gematik Gesellschaft für Telematikanwendungen der Gesundheitskarte

GPG GNU Privacy Guard

GPL GNU General Public License

HBA elektronischer Heilberufsausweis

HTML Hypertext Markup Language

HTTP Hypertext-Transfer-Protocol

HYDrOUS Handwritten Multiparty Document Signature

ISO International Organization for Standardization

JAX-RS JAVA API for RESTful Web Services

JNLP Java Network Launching Protocol

JSON JavaScript Object Notation

KV Kassenärztliche Vereinigung

LGPL GNU Library General Public License

MIT Massachusetts Institute of Technology

MPCS Multi-Party Contract Signing

ORM Object-relational mapping

PDF Portable Document Format

PKI Public Key Infrastructure

REST Representational State Transfer

RSA Rivest Shamir Adleman

SAK Signaturanwendungskomponente

SGB Sozialgesetzbuch

SHA Secure Hash Algorithm

SigG Signaturgesetz

SigV Signaturverordnung

SOAP ursprünglich Simple Object Access Protocol

SSEE Sichere Signaturerstellungseinheit

TTP Trusted Third Party

URI Uniform Resource Identifier

URL Uniform Resource Locator

USB Universal Serial Bus

VPN Virtual Private Network

WSDL Web Services Description Language

XML eXtensible Markup Language

ZPO Zivilprozessordnung

1 Einleitung

In der Einleitung soll die Motivation für das in dieser Masterarbeit behandelte Problem gegeben werden. Die Zielsetzung gibt dann an, wie das Probleme gelöst werden soll. Danach wird die Gliederung dieses Dokumentes beschrieben.

1.1 Motivation

In der heutigen Zeit ziehen immer mehr Menschen in Ballungsgebiete und die medizinische Versorgung von Patienten konzentriert sich eben auf diese Ballungsgebiete. Aufgrund von Wettbewerb und stetig steigendem Bildungsniveau in Deutschland müssen sich auch Mediziner in einer Disziplin spezialisieren, um konkurrenzfähig zu bleiben. Aus diesem Umstand ergeben sich für Patienten Vorteile, da sie für medizinische Diagnostik und Behandlung Experten aufsuchen können. Trotz dieser Verbreitung von Fachärzten ist die räumliche Nähe nicht für jeden Patienten gegeben. Durch technische Hilfsmittel und die globale Vernetzung durch das Internet ist es jedoch möglich Expertenwissen mittels *Telekonferenzen* überall auf der Welt verfügbar zu machen.

Für diesen Zweck wurde am FRAUNHOFER ISST das Projekt HEALTHTELKON entwickelt, das es entfernten Teilnehmern ermöglicht Telekonferenzen durchzuführen und dabei eine Agenda von Patienten zu besprechen. Gestützt wird dieses durch elektronische Fallakten, mit denen die Behandlungen eines Patienten dokumentiert werden. Die Ergebnisse der so geführten Telekonferenzen sollen festgehalten und in elektronische Protokolle überführt werden. In diesen Protokollen befinden sich dann neben den digitalisierten Patientendaten auch Kommentare der Konferenzteilnehmer zur Diagnose und Behandlung.

Die aus den Telekonferenzen entstandenen Protokolle sollen eine Grundlage für die Behandlung von Patienten sein. Es muss also sichergestellt werden, dass diese sensiblen Daten sowohl von allen Teilnehmern einer Konferenz als legitim erachtet werden, als auch Änderungen der Protokolle zu einem späteren Zeitpunkt nicht unbenutzt bleiben. Da die Teilnehmer zu keinem Zeitpunkt räumlich zusammentreffen, muss durch ein sorgfältig gewähltes Protokoll auch eine Signierung von entfernten Standorten erfolgen können. Die Protokolle müssen also elektronisch signiert werden, damit spätere Manipulation ausgeschlossen (*Integrität*) ist und die Konferenzteilneh-

1 Einleitung

mer eine Teilnahme nicht abstreiten können (*Authentizität*). Insbesondere ist eine Signatur aller Teilnehmer erforderlich, damit die Protokolle gemeinschaftlich signiert werden können.

1.2 Zielsetzung

Das Hauptziel dieser Masterarbeit ist der Entwurf eines Systems, welches es mehreren Teilnehmern, im Folgenden *Parteien* genannt, erlaubt Mehrparteien-Signaturen für elektronische Dokumente zu erstellen. Dazu soll ein Mehrparteien-Signaturprotokoll entworfen werden, das nur dann eine Signatur für ein Dokument erzeugt, wenn alle Parteien ihren Signaturanteil generiert haben. Fehlt auch nur ein einziger Signaturanteil, kann keine Gesamtsignatur erzeugt werden. Diese letzte Anforderung schließt somit Verfahren aus, bei denen die Parteien ein Dokument nacheinander signieren und elektronisch versenden. Denn hier würden gültige Einzelsignaturen vor Beendigung des Signaturverfahrens vorliegen. Als zusätzlichen Schutz soll das zu entwerfende System zusätzlich durch Biometrie gegen Angriffe und Fälschungen geschützt werden. Als biometrisches Merkmal sollen hierfür handgeschriebene biometrische Unterschriften genutzt werden.

1.3 Gliederung

In *Kapitel 2* werden zunächst die kryptografischen Grundbausteine erklärt, die für Verschlüsselung und elektronische Signaturen benötigt werden. Danach werden die gesetzlichen Rahmenbedingungen für kryptografische System in Deutschland beschrieben. Insbesondere werden elektronische Signaturen, empfohlene Algorithmen und Informationen zur Gesundheitstelematik in Deutschland eingeführt. Abschließend wird ein Überblick zum Projekt HEALTHTELKON gegeben.

In *Kapitel 3* werden die für diese Masterarbeit relevanten Arbeiten behandelt. Dazu wird zunächst auf Biometrie und damit verbundene Problemstellungen im Allgemeinen eingegangen, um dann handgeschriebene biometrische Unterschriften im Detail zu betrachten. Neben der Biometrie ist das sogenannte *Secret-Sharing* eine mögliche Lösungskomponente für gemeinschaftliches Handeln mit Daten und wird in diesem Kapitel erklärt. Abschließend wird das sogenannte *Multi-Party Contract Signing* eingeführt, dessen Prinzip für diese Masterarbeit relevant ist.

Kapitel 4 stellt die Anforderungsanalyse an das in dieser Masterarbeit zu entwickelnde System dar. Die Anforderungen werden als funktionale und nicht-funktionale Anforderungen beschrieben. Aus diesen werden dann Anwendungsfälle konstruiert.

Der Entwurf des Systems wird in *Kapitel 5* geschildert. Hierbei werden die zugrundeliegenden technischen Konzeptionsfragen geklärt und der Entwurf der System-

komponenten beschrieben. Danach wird das Mehrparteien-Signaturprotokoll und das Protokoll für die biometrische Schlüsselfreigabe modelliert. Am Ende des Entwurfskapitels wird die Interaktion mit HEALTHTELKON skizziert.

In *Kapitel 6* wird die Implementierung des Systems auf technischer Ebene vorgestellt. Hierzu wird auf technische Hilfsmittel (Software & Hardware) eingegangen und die Implementierung wichtiger Komponenten dargestellt.

In *Kapitel 7* wird das implementierte System evaluiert. Ziel soll eine Beurteilung der Leistung, der Sicherheit und des Schutzes vor potenziellen Angriffen sein.

Die Zusammenfassung in *Kapitel 8* zieht ein Fazit der Masterarbeit.

Zuletzt wird in *Kapitel 9* ein Ausblick auf weiterführende Arbeiten gegeben.

1 Einleitung

2 Grundlagen

In diesem Kapitel wird auf die Grundlagen für diese Arbeit eingegangen. Neben den eingesetzten Technologien der Biometrie und Kryptografie, werden auch die gesetzlichen Regelungen berücksichtigt, die beim Umgang mit elektronischen Signaturen einzuhalten sind. Zusätzlich wird auf die Gesundheitstelematik in Deutschland eingegangen und zuletzt das Projekt HEALTHTELKON vorgestellt.

2.1 Kryptografie

Wird heute von Kryptografie gesprochen, ist die Wissenschaft zur Wahrung von Informationssicherheit gemeint. Bestandteil von Informationssicherheit ist aber auch die sogenannte Verschlüsselung, eine Disziplin mit der es möglich ist Klartext mit kryptografischen Geheimnissen so zu verschleiern, dass sie nicht ohne erheblichen Aufwand oder das passende kryptografische Geheimnis wieder lesbar gemacht werden können. Die moderne Kryptografie setzt dafür auf mathematische Probleme, deren Lösung ohne Zusatzinformationen (Geheimnisse) nicht lösbar sind. Informationen werden in der Kryptografie heute nicht mehr als Folge von Zeichen (z. B. Buchstaben oder Ziffern) sondern als deren Repräsentation als Bitstring behandelt, wodurch jegliches (binäre) Dateiformat ebenfalls verschlüsselt werden kann. In dieser Masterarbeit werden die in [BSI13] empfohlenen Algorithmen und Schlüssellängen / Hashwertlängen verwendet.

2.1.1 Symmetrische Kryptosysteme

Symmetrische Kryptosysteme sind solche, die zum Verschlüsseln und Entschlüsseln das gleiche kryptografische Geheimnis nutzen.

$$Dec(Enc(msg, secret), secret) = msg$$

Symmetrische Kryptosysteme werden in der Regel dort genutzt, wo große Datenmengen verschlüsselt werden müssen, da sie aufgrund ihrer mathematischen Komplexität im Vergleich zu asymmetrischen Kryptosystemen um ein Vielfaches schneller zu berechnen sind. Bei symmetrischen Kryptoverfahren wird unterschieden zwischen den Verfahren *Blockchiffren* und *Stromchiffren*. Blockchiffren werden blockweise verschlüsselt und benötigen daher eine feste Blockgröße. Stromchiffren verschlüsseln

2 Grundlagen

Daten kontinuierlich, indem sie aus Schlüsseln eine pseudozufällige Zeichenkette mit der gleichen Länge wie der eigentliche Datenstrom generieren.

2.1.2 Asymmetrische Kryptosysteme

Asymmetrische Kryptosysteme ermöglichen aufgrund ihrer kryptografischen Funktionsweise die Möglichkeit ein Paar von kryptografischen Schlüsseln zu nutzen. Oft wird auch von Public-Key-Kryptosystemen gesprochen, da jeweils ein Schlüssel eines Schlüsselpaares öffentlich bekannt gegeben werden kann und der andere kryptografische Schlüssel geheim gehalten wird. Der Vorteil liegt im flexibleren Einsatz, da kein gemeinsames Geheimnis ausgetauscht werden muss.

$$Dec(Enc(msg, key_{private}), key_{public}) = msg$$

$$Enc(Dec(msg, key_{public}), key_{private}) = msg$$

So kann bei einem Nachrichtenaustausch der öffentliche Schlüssel eines Empfängers genutzt werden, um Nachrichten zu verschlüsseln. Eine Entschlüsselung kann der Empfänger dann mit seinem geheimen privaten Schlüssel durchführen. In umgekehrter Richtung kann ein Absender eine Nachricht mit seinem geheimen (privaten) Schlüssel signieren und der designierte Empfänger mit dem öffentlichen Schlüssel des Absenders die eingetroffene Nachricht validieren.

In modernen kryptografischen Protokollen wird oft eine Kombination aus symmetrischer und asymmetrischer Verschlüsselung eingesetzt, bei der nur die symmetrischen Schlüssel mittels eines asymmetrischen Schlüsselpaares verschlüsselt werden. So kann man die Vorteile beider Systeme miteinander vereinen.

2.1.3 Hashverfahren

Unter Hashverfahren versteht man Funktionen, die Eingaben auf einen (kleineren) Bildbereich abbilden. Die wichtigste Eigenschaft dieser Abbildungen ist, dass es ohne erheblichen Aufwand nicht möglich sein soll aus der Ausgabe auf eine Eingabe zu schließen. *Hashfunktionen* sind somit sogenannte *Einwegfunktionen*.

2.1.1 Definition. Eine Funktion $f : E \rightarrow A$ heißt *Hashfunktion*, wenn $|E| \geq |A|$ und gilt:

$$\begin{array}{ll} f(x) = y & \text{effizient berechenbar,} \\ f^{-1}(y) = x & \text{\textbf{nicht} effizient berechenbar.} \end{array}$$

Aus diesen Eigenschaften folgt ebenso, dass es nicht effizient möglich sein soll einen Funktionswert x' zu finden, sodass gilt:

$$\forall x \exists x' : \{x \neq x' \mid f(x) = f(x')\} \quad (\text{Kollisionsresistenz})$$

Beim Auffinden einer solchen Konstellation spricht man von *Kollisionen*, welche bei Hashverfahren nicht ausgeschlossen werden können, da eine Abbildung auf einen kleineren Bildbereich immer einen Informationsverlust mit sich bringt. Ein Ziel von Hashfunktionen ist es insbesondere diese Kollisionen zu vermeiden, indem zwei sehr ähnliche Eingaben einen möglichst verschiedenen Hashwert besitzen (*Chaos*), jeder Hashwert gleich oft „getroffen“ wird (*Gleichverteilung*) und jeder mögliche Hashwert durch Eingaben erreicht werden kann (*Surjektivität*).

2.1.4 Digitale Signaturen

Digitale Signaturen sind die Kombination von asymmetrischer Kryptografie (Unterabschnitt 2.1.2) und Hashverfahren (Unterabschnitt 2.1.3). Da die asymmetrische Verschlüsselung im Vergleich zu symmetrischer Verschlüsselung langsam ist und Signaturen nicht so lang wie die Nachricht selbst sein sollten, werden digitale Signaturen mit Hilfe von Hashwerten der zu signierenden Daten erstellt.

1. Eingabedaten D bestimmen
2. Hashwert $H(D)$ berechnen
3. Hashwert signieren $Sig = Enc(H(D), signer_{private})$
4. Signierte Nachricht (D, Sig) versenden

Der designierte Empfänger muss dann wie folgt vorgehen:

1. Empfange signierte Nachricht (D, Sig)
2. Berechne $H(D)'$
3. Entschlüssele Sig mit $H(D) = Dec(Sig, signer_{public})$
4. Prüfe ob $H(D) = H(D)'$

Üblicherweise wird bei der Nutzung von digitalen Signaturen nicht von öffentlichem Schlüssel und privatem Schlüssel gesprochen. Stattdessen wird der private Schlüssel als *Signatur Schlüssel* (§ 2, Nr.4, SigG) und der öffentliche Schlüssel als *Signaturprüfungsschlüssel* (§ 2, Nr.5, SigG) bezeichnet.

2.2 Rahmenbedingungen in Deutschland

Für den Einsatz von elektronischen Signaturen und damit verbundenen kryptografischen Verfahren gelten in Deutschland gesetzliche Regelungen. Sie legen fest, wie elektronische Signaturen vor einem Gericht bewertet werden können und welche Algorithmen für solche Vorhaben geeignet sind. Speziell für das Gesundheitswesen gibt

2 Grundlagen

es bis auf wenige Ausnahmen keine Regelungen, die sich von den allgemein gültigen Gesetzen abheben. Als Leitfaden gibt der Competence Center für die Elektronische Signatur im Gesundheitswesen e.V. (CCESigG) deswegen Empfehlungen.

2.2.1 Gesetzliche Regelungen und Anforderungen für elektronische Signaturen

Zur Umsetzung dieser Masterarbeit sind sogenannte „elektronische Signaturen“ nötig, welche als das digitale Pendant zu handgeschriebenen Unterschriften angesehen werden. Ihre Sicherheit basiert auf modernen Kryptografieverfahren, welche auf schwer berechenbaren mathematischen Problemen basieren. Um elektronische Signaturen mit Beweiskraft zu erzeugen, müssen außerdem das Umfeld der Generierung und die verwendeten Zertifikate bestimmten (gesetzlichen) Regelungen genügen.

Elektronische Signaturen sind vor dem deutschen Gesetz im Rahmen des Signaturgesetzes¹ (Signaturgesetz (SigG)), sowie der Signaturverordnung festgelegt. Jedoch gibt es anders als bei nicht-digitalen Unterschriften verschiedene Arten von elektronischen Signaturen, die auch jeweils andere Beweiskraft besitzen.

Diese Arten elektronischer Signaturen sind gestaffelt in einfache, fortgeschrittene, sowie qualifizierte Signaturen. Einfache elektronische Signaturen sind hierbei die beweisschwächsten Signaturen, fortgeschrittene elektronische Signaturen haben weitere technische Merkmale und Voraussetzungen, die erfüllt werden müssen. Qualifizierte elektronische Signaturen sind, wenn nicht für einen konkreten Zweck ausgeschlossen, das Äquivalent zur Unterschrift auf Papierdokumenten.

(Einfache) elektronische Signaturen

Nach § 2, Abs. 1, SigG sind „[...] elektronische Signaturen Daten in elektronischer Form, die anderen Daten beigefügt oder logisch mit ihnen verknüpft sind und die zur Authentifizierung dienen [...]“. Weitere Anforderungen werden an diese Form elektronischer Signaturen nicht gestellt, sodass jegliche Form der Zuordnung zu einem Absender als (einfache) elektronische Signatur angesehen werden kann. Vor Gericht unterliegt eine (einfache) elektronische Signatur der *freien Beweiswürdigung* (§ 286, ZPO²) und wird als Beweis nur dann akzeptiert, wenn fundierte Begründungen das Gericht von der Echtheit überzeugen können. Generell sollten (einfache) elektronische Signaturen also nur zur Kennzeichnung des Autors genutzt werden [SKB⁺10].

¹Signaturgesetz vom 16. Mai 2001 (BGBl. I S. 876), das zuletzt durch Artikel 4 des Gesetzes vom 17. Juli 2009 (BGBl. I S. 2091) geändert worden ist

²Zivilprozessordnung in der Fassung der Bekanntmachung vom 5. Dezember 2005 (BGBl. I S. 3202; 2006 I S. 431; 2007 I S. 1781), die durch Artikel 4 des Gesetzes vom 11. März 2013 (BGBl. I S. 434) geändert worden ist

Beispiel: Versendet man eine E-Mail und gibt seinen *Namen als Kopfzeile* an oder fügt eine *textuelle Signatur* an den Inhalt der E-Mail an, so gilt beides als eine elektronische Signatur.

Fortgeschrittene elektronische Signaturen

Fortgeschrittene elektronische Signaturen sind (einfache) elektronische Signaturen, welche sich in den folgenden vier Punkten abgrenzen.

Nach § 2, Abs. 2, SigG müssen sie [...]

1. „*ausschließlich dem Signaturschlüssel-Inhaber zugeordnet (sein) [...]*”
2. „*die Identifizierung des Signaturschlüssel-Inhabers ermöglichen [...]*”
3. „*mit Mitteln erzeugt werden, die der Signaturschlüssel-Inhaber unter seiner alleinigen Kontrolle halten kann [...]*”
4. „*mit den Daten, auf die sie sich beziehen, so verknüpft (sein) [...], dass eine nachträgliche Veränderung der Daten erkannt werden kann [...]*”

Durch die so definierten Einschränkungen erhalten fortgeschrittene Signaturen mehr Beweiskraft, jedoch unterliegen sie auch der *freien Beweiswürdigung* (§ 286, ZPO). Somit sind sie vor dem Gesetz nicht äquivalent zu handgeschriebenen Unterschriften auf Dokumenten. Im wesentlichen müssen für ein solches Äquivalent weitere technische Details wie Verschlüsselungsalgorithmen, Hashalgorithmen und Rahmenbedingungen für die (sichere) Aufbewahrung getroffen werden.

Beispiel: Versendet man eine E-Mail und signiert diese mit einem Verfahren wie GNU Privacy Guard (GPG)³, welches auf asymmetrischer Kryptografie basiert, wird ein Schlüsselpaar genutzt, das *einem Benutzer (bzw. bestimmten E-Mail Adressen dieses Benutzers) zugeordnet* ist (Punkte 1 bis 3). Nachträgliche Veränderungen einer so signierten Nachricht fallen mit einer sehr hohen Wahrscheinlichkeit auf, da die Signatur stets aus einem *Hashwert der eigentlichen Nachricht* besteht, welche mit dem (privaten) *Signaturschlüssel* verschlüsselt wird (Abb. 2.1a).

³GnuPG <http://www.gnupg.org>

2 Grundlagen

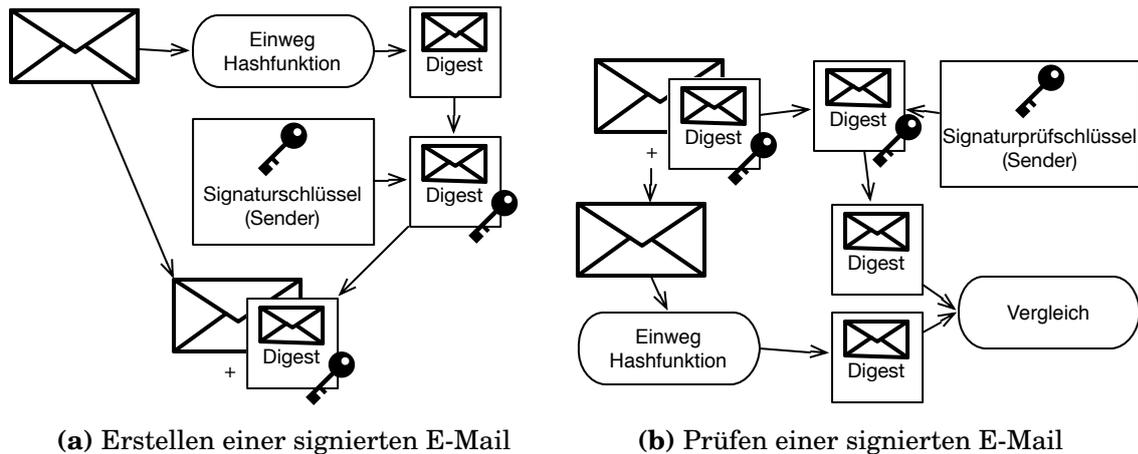


Abbildung 2.1: Beispiel GPG Verfahren für E-Mail

Zur Überprüfung wird erneut ein Hashwert der eigentlichen Nachricht erzeugt, welcher mit dem mittels öffentlichem *Signaturprüf Schlüssel* entschlüsseltem Hashwert in der Signatur verglichen wird (Punkt 4) (Abb. 2.1b).

Qualifizierte elektronische Signaturen

Qualifizierte elektronische Signaturen (§ 2, Abs. 3, SigG) sind fortgeschrittene Signaturen (§ 2, Abs. 2, SigG), die

1. „auf einem zum Zeitpunkt ihrer Erzeugung gültigen *qualifizierten Zertifikat* beruhen [...]“
2. „mit einer sicheren Signaturerstellungseinheit erzeugt werden“.

Erläuterung: Qualifizierte Signaturen sind prinzipiell wie fortgeschrittene Signaturen zu nutzen, jedoch bieten sie zusätzlich qualifizierte Zertifikate um kryptografische Schlüsselpaare eindeutig und rechtlich bindend einer Identität zuzuordnen.

Qualifizierte Zertifikate (§ 2, Abs. 7, SigG) sind Zertifikate (§ 2, Abs. 6, SigG), welche „die Voraussetzungen des § 7, SigG erfüllen und von Zertifizierungsdiensteanbietern ausgestellt werden, die mindestens die Anforderungen nach den §§ 4-14, SigG oder § 23, SigG und der sich darauf beziehenden Vorschriften der Rechtsverordnung nach § 24 erfüllen[...]“.

In § 7, SigG werden die erforderlichen neun Angaben von qualifizierten Zertifikaten beschrieben. Sie enthalten

- den Namen des Signaturschlüssel-Inhabers,
- den zugeordneten Signaturprüf Schlüssel,

- die eingesetzten kryptografischen Algorithmen,
- die laufende Nummer des Zertifikates,
- Beginn/Ende der Gültigkeit des Zertifikates,
- Name und Staatszugehörigkeit des Zertifizierungsdienstanbieters,
- Angaben über Beschränkung bei der Nutzung des Zertifikates,
- Angaben, dass es sich um ein qualifiziertes Zertifikat handelt und
- (optionale) Attribute des Signaturschlüssel-Inhabers.

In den weiteren Paragraphen (§§ 4-14, SigG) werden Anforderungen an Zertifizierungsdienstleister zum Betrieb beschrieben. Sie umfassen § 5 Vergabe von qualifizierten Zertifikaten, § 6 Unterrichtungspflicht, § 8 Sperrung von qualifizierten Zertifikaten, § 9 Qualifizierte Zeitstempel, § 10 Dokumentation, § 11 Haftung, § 12 Deckungsvorsorge, § 13 Einstellung der Tätigkeit und § 14 Datenschutz.

Sichere Signaturerstellungseinheit (SSEE) (§ 2, Abs.10, SigG) sind „[...] Software- oder Hardwareeinheiten zur Speicherung und Anwendung des Signaturschlüssels, die mindestens die Anforderungen nach § 17 oder § 23 [...] SigG [...] und der sich darauf beziehenden Rechtsverordnung nach § 24 erfüllen und für qualifizierte elektronische Signaturen bestimmt sind [...]“. Die „Technische Sicherheit von Produkten für qualifizierte elektronische Signaturen“ (§ 17, SigG) beschreibt wie Signaturanwendungskomponenten funktionieren müssen.

„Sichere Signaturerstellungseinheiten [...] müssen gewährleisten, dass der Signaturschlüssel erst nach Identifikation durch Besitz und Wissen oder durch Besitz und ein oder mehrere biometrische Merkmale angewendet werden kann“ (§ 15 (1), Abs.1, Signaturverordnung (SigV) ⁴).

2.2.2 Wichtige Begriffe der Informationssicherheit

Das BSI beschreibt die vier folgenden kryptografischen Grundziele⁵ (Tab. 2.1) beim Einsatz von Kryptoprodukten. Diese vier Grundziele sollten bei der Implementierung von Systemen, die mit sensiblen Daten und Identitäten arbeiten stets berücksichtigt werden.

⁴Signaturverordnung vom 16. November 2001 (Bürgerliches Gesetzbuch (BGB)). I S. 3074), die zuletzt durch Artikel 1 der Verordnung vom 15. November 2010 (BGBl. I S. 1542) geändert worden ist

⁵M 3.23 Einführung in kryptografische Grundbegriffe <https://www.bsi.bund.de/ContentBSI/grundschutz/kataloge/m/m03/m03023.html> (Stand: 11.04.13)

2 Grundlagen

Vertraulichkeit	Keine unbefugte dritte Partei soll an die Inhalte von Dateien gelangen
Nicht-abstreitbarkeit (Verbindlichkeit)	Der Versand (oder Erhalt) einer Nachricht kann zu einem späteren Zeitpunkt nicht mehr abgestritten werden.
Integrität	(Unbefugte) Manipulation einer Nachricht, muss entdeckt werden können.
Authentizität	Teilnehmer einer Kommunikation können ihre Identität zweifelsfrei beweisen. Der Absender kann zweifelsfrei zeigen, dass eine Nachricht von ihm stammt.

Tabelle 2.1: Kryptografische Grundziele

2.2.3 Sichere Algorithmen nach dem Signaturgesetz der Bundesrepublik Deutschland

Gemäß § 3, SigG gibt die Bundesnetzagentur für Elektrizität, Gas, Telekommunikation, Post und Eisenbahnen jährliche Empfehlungen zu geeigneten Algorithmen zur Erfüllung der Anforderungen nach § 17, Abs.1–3, SigG in Verbindung mit Anlage 1, Abs.I, Nr.2, SigV [Bun13]. Die folgenden Empfehlungen gelten bis Ende 2019.

Geeignete Hashfunktionen (2013)

Mindestens bis *Ende 2019* sind die Hashfunktionen der **SHA-2** (*Secure Hash Algorithm*) Familie

- *SHA-256*
- *SHA-384*
- *SHA-512*
- *SHA-512/256*

für die Erstellung qualifizierter Zertifikate (Paragraph 2.2.1) geeignet.

Die Vorgängerfunktionen der *SHA-1* Familie sollten jedoch seit 2010 nicht mehr genutzt werden, da Angriffe existieren. Faktisch sind *SHA-2* Hashfunktionen somit die einzig sicheren Hashfunktionen im Jahre 2013. In 2014 soll jedoch der Nachfolger *SHA-3* (*Keccak*) als neuer Standard vorgestellt werden⁶, wodurch eine Alternative zu *SHA-2* entsteht.

⁶National Institute of Standards and Technology (NIST): SHA-3 Standardization http://csrc.nist.gov/groups/ST/hash/sha-3/sha-3_standardization.html (Stand: 12.09.2013)

Geeignete Signaturverfahren

RSA Das *RSA*-Verfahren (**R**ivest, **S**hamir und **A**dleman) ist ein asymmetrisches Verschlüsselungsverfahren, welches als Signaturverfahren genutzt werden kann. Seine Sicherheit basiert auf dem Faktorisierungsproblem für ganze Zahlen in ihre Primfaktoren. Die empfohlene Schlüssellänge für den öffentlichen Parameter $n := p \cdot q$ sind **2048 Bit** (mindestens 1976 Bit).

$$\begin{array}{c}
 \text{öffentlich} \\
 \underbrace{} \\
 RSA \quad \underbrace{n, e} \\
 \underbrace{p, q, d} \\
 \text{privat}
 \end{array}$$

Die beiden privaten Parameter p und q (Primzahlen) sollten von ähnlicher Größenordnung

$$\epsilon_1 < |\log_2(p) - \log_2(q)| < \epsilon_2, \text{ mit } \epsilon_1 \approx 0,1 \text{ und } \epsilon_2 \approx 30$$

sein und separat von einander zufällig gewählt werden. Der zweite öffentliche Parameter e wird relativ prim zur Eulerschen ϕ -Funktion von n gewählt.

$$\begin{aligned}
 \phi(n) &= (p - 1) \cdot (q - 1) \\
 \text{ggT}(e, \phi(n)) &= 1, \text{ empfohlen mit } 2^{16} + 1 \leq e < \phi(n)
 \end{aligned}$$

Der dritte private Parameter d wird als die multiplikative Inverse von e modulo $\phi(n)$ gewählt.

$$e \cdot d \equiv 1 \pmod{\phi(n)}$$

Als Standardverfahren zur Erstellung digitaler Signaturen mit *RSA* wird das Signaturschema *RSASSA-PSS* empfohlen. Das ältere Schema *RSASSA-PKCS1-v1-5* sollte nur noch bis einschließlich 2015 (bzw. 2017 für Zertifikatssignaturen) eingesetzt werden.

DSA Der *Digital Signature Algorithm(us)* ist eine Variation des *ElGamal*-Signaturverfahrens, dessen Sicherheit auf dem Problem zur Lösung diskreter Logarithmen basiert. Bei *DSA* ist der öffentliche Schlüssel (p, q, g, A) und der private Schlüssel a .

Der Parameter p sollte eine Bitlänge von mindestens **2048 Bit** und der Parameter q mindestens **256 Bit** (bzw. 224 Bit bis Ende 2015) haben. Beide Parameter sind Primzahlen, die nach probabilistischen Primzahltests höchstens mit einer Wahrscheinlichkeit von 2^{-100} zusammengesetzt sein dürfen. Der Parameter g ist der Generator und $A = g^a \pmod p$ der verblendete Exponent [Bis09].

2 Grundlagen

ECDSA Die auf elliptischen Kurven basierenden *ECDSA (Elliptic Curve Digital Signature Algorithm)* Verfahren haben im Vergleich zu den vorherigen deutlich kürzere Schlüsselparameter.

Für die elliptischen Kurven $E(F_p)$ und $E(F_{2^m})$ werden für den Parameter p **keine Einschränkungen** getroffen, für den Parameter q jedoch mindestens **250 Bit** (bzw. 224 Bit bis Ende 2015).

2.2.4 Gesundheitstelematik in Deutschland

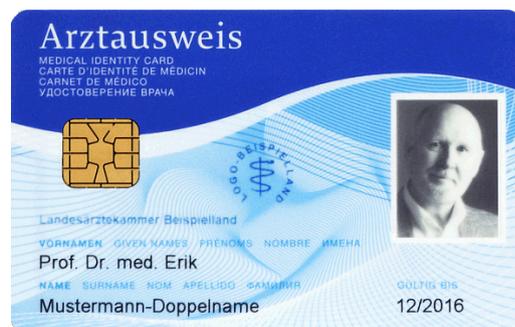
Die Gesundheitstelematik in Deutschland wird technisch von zwei Vereinigungen vertreten. Die *Gesellschaft für Telematikanwendungen der Gesundheitskarte (gematik)* setzt die gesetzlichen Anforderungen für den elektronischen Umgang mit Patientendaten um und die KV Telematik ARGE ermöglicht Leistungserbringern im Gesundheitswesen mit ihren Produkten die Nutzung von Telematikanwendungen.

Gesellschaft für Telematik (*gematik*)

Die gematik ist einem gesetzlichen Auftrag nach § 291b, SGB 5⁷ verpflichtet. Eine Aufgabe in diesem Gesetz ist die Einführung der elektronischen Gesundheitskarte (EGK) (Abb. 2.2a) für Patienten, die Grunddaten zur Person (§ 291 Abs. 2, SGB 5), enthält und als Transportmedium für weitere medizinische Daten (ärztliche Verordnungen (*eRezept*), Notfallversorgungsdaten, elektronischer Arztbrief, Unverträglichkeiten für bestimmte Medikamente, Daten über in Anspruch genommene Leistungen und deren Kosten) geeignet ist.



(a) *Elektronische Gesundheitskarte* [Wik12]



(b) *Heilberufsausweis* [Bun08]

Abbildung 2.2: Muster der SmartCards in der Gesundheitstelematik

⁷Sozialgesetzbuch (SGB) Fünftes Buch (V) - Gesetzliche Krankenversicherung - (Artikel 1 des Gesetzes v. 20. Dezember 1988, BGBl. I S. 2477), zuletzt geändert durch Art. 4 Abs. 3 G v. 20.4.2013 I 868

KV Telematik ARGE

„Die KV Telematik ARGE ist die Telematik-Arbeitsgemeinschaft der bundesdeutschen Kassenärztlichen Vereinigung. Im deutschen Gesundheitswesen übernimmt sie die Verantwortung zur Schaffung von telematischen Infrastrukturen für den sicheren Datentransfer zwischen medizinischen Leistungserbringern.“⁸

Die KV Telematik ARGE übernimmt somit die Schaffung einer Telematik Infrastruktur, wie sie zur Nutzung der *elektronischen Gesundheitskarte* erforderlich ist. Als Pendant zur *elektronischen Gesundheitskarte* für Personen im Gesundheitswesen, wird der elektronischer Heilberufsausweis (HBA)⁹ (Abb. 2.2b) eingeführt, mit dem auf Patientendaten nach (§ 291a Abs. 5, SGB 5) Zugriff gewährt werden darf. Erst durch den bundesweiten Einsatz einer solchen SmartCard werden Telematikanwendungen, wie *eRezept*, elektronische Arzneimitteldokumentation und der elektronische Arztbrief möglich. Die wesentlichen Funktionen des HBA sind Authentifizierung (Unterabschnitt 2.2.2), digitale Signatur (Unterabschnitt 2.2.1), sowie Verschlüsselung und Entschlüsselung (Abschnitt 2.1).

2.2.5 Arten des Signaturerstellungsprozesses

Das *Bundesamt für Sicherheit in der Informationstechnik (BSI)* definiert in den technischen Reports [BSI07a, BSI07b] verschiedene Arten des Signaturerstellungsprozesses (Abb. 2.3) mit dem *HBA* und beschreibt technische Voraussetzungen und Abläufe der Signaturerstellung.

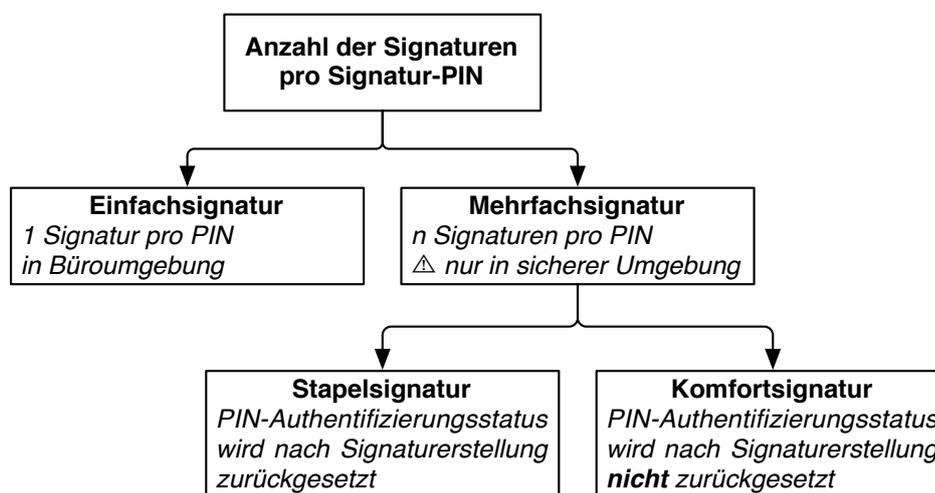


Abbildung 2.3: Übersicht der Signaturverfahren [BSI07a, BSI07b]

⁸KV-Telematik Internetauftritt <http://www.kv-telematik.de/kv-telematik-arge/profil/> (Stand 23.05.13)

⁹KV-Telematik Information zum *HBA* <http://www.kv-telematik.de/lesegeraete-karten/karten/heilberufsausweis-hba/> (Stand 23.05.13)

2 Grundlagen

Einfachsignatur

Eine Einfachsignatur wird dann genutzt, wenn nach einer Authentifizierung des Unterzeichners nur ein einzelnes Dokument unterzeichnet werden soll. Die *SSEE* (Paragraph 2.2.1) genehmigt nach erfolgreicher Authentifizierung höchstens die Erstellung einer Signatur.

Mehrfachsignatur

Soll mehr als ein Dokument nach einer einzelnen Authentifizierung gegen die *SSEE* durchgeführt werden, spricht man von Mehrfachsignaturen. Durch das höhere Risiko zum Missbrauch wird bei diesem Verfahren ein „sicherer Bereich“ [BSI07a, S.7] für die *SSEE* vorausgesetzt, der z. B. keinen Publikumsverkehr hat oder stets unter Beobachtung einer vertrauenswürdigen Person steht.

Beispiel: Ein Arzt möchte mehrere Rezepte für Patienten unterzeichnen. Angenommen er möchte 10 Rezepte unterzeichnen und die *SSEE* gestattet ihm bis zu 10 Signaturen nach einer Authentifizierung, so kann er 10 Rezepte gleichzeitig signieren. Möchte er hingegen 11 Rezepte signieren, muss er sich zweimal gegen die *SSEE* authentifizieren.

Stapelsignatur

Das *Bundesamt für Sicherheit in der Informationstechnik (BSI)* definiert Stapelsignaturen als Variante der Mehrfachsignatur mit einige Besonderheiten. Soll eine Stapelsignatur durchgeführt werden, muss diese unverzüglich nach Authentifizierung gegen die *SSEE* erfolgen, da die Authentifizierung sonst ausläuft. Per Definition müssen alle Dokumente auf dem Stapel direkt nacheinander signiert werden.

Beispiel: Ein Arzt möchte mehrere Rezepte für Patienten unterzeichnen. Diese liegen fertig ausgefüllt bereit und müssen von ihm nur noch signiert werden. Er wählt in seiner *Signaturanwendungskomponente (SAK)* die 5 Dokumente aus, authentifiziert sich mittels PIN und unterzeichnet die Dokumente als einen Stapel. Würde der Arzt in diesem Fall einen weiteren Stapel mit 5 Dokumenten unterzeichnen wollen, müsste er sich erneut authentifizieren, da für jede Stapelsignatur eine Authentifizierung nötig ist. Hätte der Arzt jedoch direkt alle 10 Dokumente für die Stapelsignatur ausgewählt, wäre eine Authentifizierung ausreichend gewesen. Voraussetzung dafür ist, dass die *SSEE* Mehrfachsignaturen mit bis zu 10 Signaturen zulässt.

Komfortsignatur

Das *BSI* definiert Komfortsignaturen als Variante der Mehrfachsignatur, die „komfortabler“ als die Stapelsignatur gestaltet ist. Bei einer Komfortsignatur muss vor der

Erstellung einer oder mehrerer Signaturen eine Authentifizierung mit PIN stattfinden, wobei die Zeitspanne nach der eine Authentifizierung ungültig wird von der *SAK* (bzw. der Person) selbst gewählt werden kann. Sollen in diesem gewählten Zeitraum Signaturen erstellt werden, muss jedoch vor jeder Signatur eine Willenserklärung (z. B. mittels USB-Token oder biometrischen Merkmalen) am Authentisierungsterminal erteilt werden, wobei der *HBA* stets in der *SSEE* verbleiben muss.

Beispiel: Ein Arzt behandelt mehrere Patienten und kann (bzw. möchte) beim Ausstellen von Rezepten keine Zeit für die Eingabe des länglichen PIN Codes für die Authentifizierung mit seinem *HBA* verlieren. In seinem (verschlossenen) Büro hat er daher seinen *HBA* im Kartenterminal und sich mit dem PIN gegenüber der *SSEE* authentifiziert. Am Empfang stellt er nun die Rezepte für seine Patienten über Komfortsignaturen aus, indem er sich unkompliziert bei jedem Rezept mittels Fingerabdruck-Scan authentifiziert und so seine Willenserklärung zur Signatur abgibt.

2.2.6 Braunschweiger Regeln zur Archivierung mit elektronischen Signaturen im Gesundheitswesen

Die „Braunschweiger Regeln zur Archivierung mit elektronischen Signaturen im Gesundheitswesen“, entworfen durch den Verein CCESigG, stellen eine Empfehlung für Einrichtungen im Gesundheitswesen dar [SKB⁺10, S.11f]:

1. Generelle Verwendung archivgeeigneter Dateiformate (z. B. PDF/A) sowie qualifizierter elektronischer Signaturen und Zeitstempel mit Anbieterakkreditierung durch die Bundesnetzagentur (nachfolgend als akkreditierte Signatur bzw. akkreditierter Zeitstempel bezeichnet).
2. Akkreditierte Signatur originär elektronischer Dokumente, für die gesetzliche Regelungen, die Schriftform fordern (grundsätzlich kann die Schriftform – unterschriebenes Papierdokument – gemäß § 126a, Abs.1, BGB durch die elektronische Form ersetzt werden).
3. Akkreditierte Signatur für Dokumente zur externen Verwendung und für interne Dokumente, die einen besonders hohen Stellenwert (z. B. Beweisinteresse) haben.
4. Akkreditierter („Eingangs-“) Zeitstempel für Dokumente externer Einsender (kann auch durch Regel Nr. 6 umgesetzt werden).
5. Geeignetes Authentifizierungsverfahren für alle sonstigen Dokumente.
6. Zeitnahe Archivierung der Dokumente, Protokoll- und Verifikationsdaten in einem revisionssicheren Archiv mit akkreditiertem („Archiv-“) Zeitstempel, in jedem Fall innerhalb von maximal 24 Stunden nach Erstellung oder Erhalt.

2 Grundlagen

7. Absicherung des Betriebs des elektronischen Archivs nach dem Stand der Technik durch Umsetzung allgemein anerkannter Regelungen und Normen (z. B. ISO 2001, BSI) – im Idealfall Nachweis durch Zertifikat.
8. Hash- und Signaturerneuerungen gemäß den Vorgaben der Bundesnetzagentur; Datei- und Medienkonvertierungen gemäß den Empfehlungen der BMWi-Studie TransiDoc.
9. Generelle Vermeidung von Medienbrüchen. Falls dennoch ersetzendes Scannen erforderlich ist:
 - a) Aufbewahrung der Originaldokumente, für die gesetzliche Regelungen die Schriftform fordern.
 - b) Verwendung eines abgesicherten Scanverfahrens nach dem Stand der Technik mit akkreditierter Signatur und / oder akkreditiertem Zeitstempel durch qualifiziertes eigenes Personal oder einen geeigneten externen Dienstleister.
 - c) Sicherstellung des uneingeschränkten Fortbestandes des Versicherungsschutzes.
10. Dokumentation und Handlungsanweisungen hinsichtlich der Verfahren, des Einsatzes der Signatur und weitergehender Regelungen (Verantwortlichkeiten, Datenschutz, Aktenstruktur, etc.) in einer Archivordnung.

In Abb. 2.4 wird eine Übersicht der Signaturverfahren in Kombination mit Zeitstempel und Authentifizierung gezeigt, wobei die Piktogramme wie folgt zu deuten sind:

-  Verwendung eines geeigneten Authentifizierungsverfahrens empfohlen
-  keine elektronische Signatur notwendig
-  Verwendung eines einfachen elektronischen Zeitstempels
-  Verwendung einer fortgeschrittenen elektronischen Signatur gemäß § 2, Abs.2, SigG empfohlen
-  Verwendung eines qualifizierten elektronischen Zeitstempels gemäß § 2, Abs.14, SigG empfohlen
-  Verwendung einer qualifizierten elektronischen Signatur gemäß § 2, Abs.3, SigG empfohlen
-  Weiterhin Verwendung der Papierform empfohlen

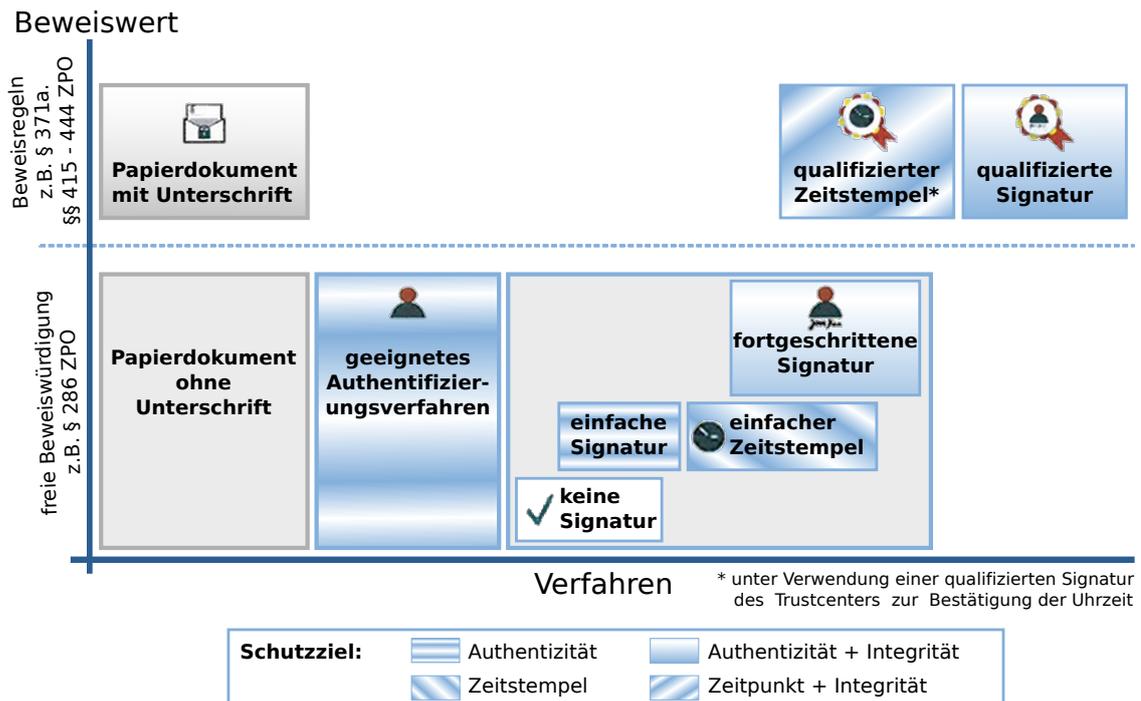


Abbildung 2.4: Einstufung der verwendeten Signaturverfahren bezüglich des Beweiswertes [SKB⁺10, S.18]

Im Detail gehen die Autoren in [SKB⁺10, S.45ff] dabei auf verschiedene Dokumente im Gesundheitswesen ein und geben zu den verschiedenen Dokumenten Empfehlungen ab.

Beispiel: Konsile sollten ähnlich wie Leistungsanforderungen gehandhabt werden. Es sollte eine Zuordnung zu einem Arzt sichergestellt werden. Somit wird mindestens eine Authentifizierung empfohlen, die durch qualifizierte Zeitstempel (bessere Nachverfolgbarkeit) und/oder fortgeschrittene Signaturen bestärkt werden kann.

2.3 HealthTelKon

Ziel des TELEMEDIZIN REPOSITORY ist die Förderung neuer Dienste, die sowohl Patienten als auch deren Leistungserbringer bei Erhaltung oder Wiederherstellung der Gesundheit unterstützen. Ein konkreter Anwendungsfall sind interdisziplinäre Konferenzen. In diesen Konferenzen wird von Spezialisten (der beteiligten Kliniken), basierend auf den erhobenen Befunden die weitere Behandlungsstrategie nach neuesten Erkenntnissen aus der Forschung unter Berücksichtigung vorgegebener Leitlinien festgelegt, um den Patienten die beste Genesung und Prognose anzubieten.

Die im Rahmen dieser Masterarbeit entwickelte Lösung wird zunächst für eben solche Telekonferenzen eingesetzt. Die Anwendung HEALTHTELKON [HO13] ist ein vom FRAUNHOFER ISST entwickeltes Telekonferenzsystem, welches im Rahmen

2 Grundlagen

des Projektes TELEMEDIZIN REPOSITORY, gefördert durch die Landesregierung NRW und den europäischen Fond für regionale Entwicklung, entwickelt wurde.

2.3.1 Ablauf einer Konferenz

Die mittels HEALTHTELKON durchgeführten Konferenzen laufen wie folgt ab:

1. Patient vorstellen: Name, Geburtsdatum, Geschlecht, Haupt-/Nebendiagnose und die aktuelle Therapie
2. Medizinische Daten besprechen: Anhand der medizinischen Daten den Erkrankungsstand diskutieren
3. Therapieempfehlungen und weitere Diskussion: Vorschläge für weitere Behandlungen besprechen. Grundlage sind bisherige medizinische Informationen über den Patienten
4. Weitere Behandlungsschritte festlegen: Die Konferenzteilnehmer einigen sich auf weitere Behandlungsschritte
5. Protokollierung: Der Konferenzmanager protokolliert Kommentare und Ergebnisse (Therapieempfehlungen) der Besprechung
6. Protokoll unterzeichnen: Nachdem alle Patientenfälle besprochen wurden, wird das Protokoll ausgedruckt und von allen Teilnehmer unterzeichnet

2.3.2 Technische Details

Auf der technischen Seite bietet HEALTHTELKON zahlreiche Möglichkeiten, die Telekonferenzen interaktiver zu gestalten. So können Bildschirminhalte und digitale Dokumente mit Patientendaten mit allen Konferenzteilnehmer geteilt werden. Sensible Patientendaten werden hierbei den Konferenzteilnehmer über die ELEKTRONISCHE FALLAKTE (EFA)¹⁰ in eigens erstellte Konferenzakten bereitgestellt. Konferenzakten können feingranular mit Zugriffsrechten versehen werden, die einzelne Ärzte oder auch Institutionen und deren Mitarbeiter berechtigen können.

2.3.3 Protokollierung

Bei Abschluss der Konferenz wird automatisch ein Konferenzprotokoll im PDF-Format erstellt und in der Konferenzakte gespeichert. Für die automatische Erstellung des Protokolls werden die in der Telekonferenz elektronisch vorliegenden Informationen genutzt:

¹⁰Verein elektronische FallAkte <http://www.fallakte.de> (Stand: 22.05.13)

- Konferenzinformation
- Konferenzmanager
- Konferenzteilnehmer
- Patientenbesprechungen (inklusive medizinische Informationen z. B. Diagnose und aktuelle Therapie)
- Kommentare, Therapiebeschlüsse

Da die Beschlüsse der Konferenz verbindlich sind, ist eine Signierung des Protokolls durch die teilnehmenden Ärzte erforderlich.

2 Grundlagen

3 Verwandte Arbeiten

Für die Implementierung eines Mehrparteien-Signatursystems mit biometrischen Unterschriften, werden diverse Komponenten benötigt. In diesem Kapitel werden die verwandten wissenschaftlichen Arbeiten und Verfahren beschrieben unter deren Berücksichtigung der Entwurf von biometrischen und kryptografischen Systemkomponenten durchgeführt wird.

3.1 Biometrie

Das Themengebiet der Biometrie versucht die einzigartigen Eigenschaften eines Menschen zu vermessen und auszuwerten. Wurden diese Eigenschaften (biometrische Merkmale) einmal aufgezeichnet, können sie dazu genutzt werden, Menschen (theoretisch) eindeutig zu identifizieren bzw. zu authentifizieren.

Zur Authentifizierung von Menschen können aber auch nicht-biometrische Merkmale wie Passwörter verwendet werden. In verbreiteten kryptografischen Systemen ist die Frage nach dem Besitz eines Merkmales zentral, denn der Besitzer kann sich so gegenüber dem System authentifizieren. In der Regel setzen kryptografische Systeme auf lange und zufällige kryptografische Schlüssel, die sich Menschen für gewöhnlich nicht einprägen können. Stattdessen werden diese kryptografischen Schlüssel in geschützter Form auf SmartCards oder Computersystemen hinterlegt, wo sie durch Eingabe von kürzeren Passwörtern freigegeben werden.

Somit ist das schwächste Glied in der Sicherheitsarchitektur das Passwort, bzw. der Umgang mit Passwörtern durch Menschen. In vielen Fällen neigen Menschen dazu, einfache Passwörter (z. B. Namen, Geburtsdaten oder „passwort“ als Passwort) zu wählen oder ihre Passwörter auf Papier zu notieren. Durch kluges Erraten oder Entwendung von einem Stück Papier kann so der Besitzer schnell wechseln. Ein weiteres Problem ist die Nichtabstreitbarkeit (Unterabschnitt 2.2.2), die durch Passwörter nicht gewährleistet werden kann, da ein Passwort auch bereitwillig Dritten zugänglich gemacht werden kann.

An dieser Stelle bieten biometrische Merkmale einen erheblichen Vorteil, da der jeweilige Besitzer sie nicht verlieren oder vergessen kann. Der Besitzer muss für eine biometrische Authentifizierung stets physikalisch bei der Eingabe anwesend sein.

3 Verwandte Arbeiten

Ebenso ist ein Erraten von biometrischen Merkmalen aufgrund ihrer Komplexität sehr unwahrscheinlich, teuer und zeitaufwändig.

3.1.1 Biometrische Merkmale

In diesem Abschnitt soll ein Überblick über die gängigsten biometrischen Merkmale gegeben werden. Um einen Vergleich der Merkmale zu ermöglichen, werden 7 Kriterien betrachtet (Tab. 3.1).

Kriterium	Beschreibung
Universalität	Haben alle Menschen dieses Merkmal?
Klarheit	Wie klar können Menschen anhand dieses Merkmales unterschieden werden?
Beständigkeit	Ändert sich dieses Merkmal mit der Zeit?
Aufzeichnung	Wie gut bzw. einfach kann dieses Merkmal aufgezeichnet werden?
Leistung	Wie schnell und akkurat ist dieses Merkmal?
Akzeptanz	Wollen Menschen dieses Merkmal nutzen?
Umgehung	Wie einfach kann dieses Merkmal umgangen bzw. gefälscht werden?

Tabelle 3.1: Vergleichskriterien biometrischer Merkmale

In [UPPJ04, S.949] werden die Kriterien für einige biometrische Merkmale durch die Autoren bewertet (Tab. 3.2), wodurch die Vorteile und Nachteile der einzelnen biometrischen Merkmale angedeutet werden. Aus den Daten ist in erster Linie zu

Merkmal	Universalität	Klarheit	Beständigkeit	Aufzeichnung	Leistung	Akzeptanz	Umgehung
Gesicht	H	L	M	H	L	H	H
Fingerabdruck	M	H	H	M	H	M	M
Handgeometrie	M	M	M	H	M	M	M
Iris	H	H	H	M	H	L	L
Tastenanschlag	L	L	L	M	L	M	M
Unterschrift	L	L	L	H	L	H	H
Sprache	M	L	L	M	L	H	H

H = hoch, M = mittel, L = niedrig

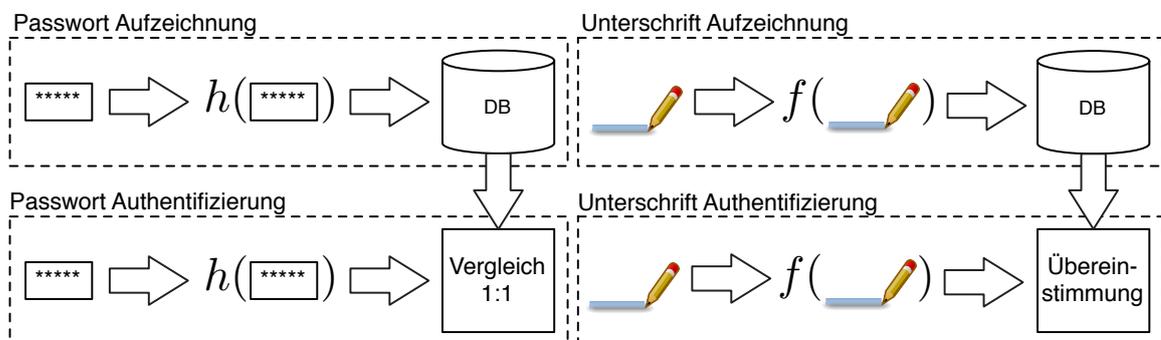
Tabelle 3.2: Vergleich biometrischer Merkmale [UPPJ04, S.949]

erkennen, dass kein biometrisches Merkmal perfekt ist und eine Evaluation der Anforderungen des designierten Einsatzzweckes erfolgen sollte.

So kann zwar mit einem Iris-Scan eine sichere, beständige und klare Erkennung von Personen durchgeführt werden, doch kann das Einlesen selbst unangenehm sein und wird deswegen von Benutzern nicht akzeptiert. Betrachtet man Fingerabdrücke, so ist die Akzeptanz ungleich höher, ohne in den übrigen Kriterien deutlich hinter einen Iris-Scan zu fallen. Im direkten Vergleich zu den beiden genannten ist die Unterschrift nach der Beurteilung der Autoren wenig geeignet. Hinterfragt man jedoch diese Bewertung, kann die geringe Universalität dadurch erklärt werden, dass nicht jeder Mensch schreiben kann. Betrachtet man dieses Kriterium aber mit Medizinern als Benutzer, trifft das Gegenteil zu. Schon dieser einfache Fall zeigt, wie notwendig die vorherige Evaluation ist.

3.1.2 Probleme beim Umgang mit Biometrie

Der Einsatz von Biometrie hat selbstverständlich nicht nur Vorteile gegenüber dem Einsatz von Passwörtern zur Authentifizierung. Statt Passwörter auf eine exakte Übereinstimmung mit einem hinterlegten Passwort zu prüfen, kann bei biometrischen Merkmalen nur ein Abgleich (*Matching*) mit *Templates* durchgeführt werden (Abb. 3.1). Sind die biometrischen Merkmale ähnlich zu den Templates, kann eine Authentifizierung erfolgen.



(a) Authentifizierung mit Passwortvergleich (b) Authentifizierung mit Übereinstimmung

Abbildung 3.1: Authentifizierungsablauf

Beim Matching mit Templates können verschiedene Faktoren eine Authentifizierung erschweren, die bei Verfahren mit festen Passwörtern nicht auftreten können. Im folgenden werden drei übliche Hürden beschrieben [UPPJ04, S.949].

Präsentation (Erfassung): Die biometrischen Merkmale eines Benutzers, können durch diverse Faktoren bei der Präsentation beeinflusst werden.

3 Verwandte Arbeiten

Beispiel: Möchte ein Benutzer seinen Fingerabdruck zur Authentifizierung nutzen, wird dieser räumliche Finger auf eine 2-Dimensionale Ebene übertragen. Bei dieser Übertragung gehen zwangsläufig Informationen verloren und Schwankungen beim Anpressdruck oder der Feuchtigkeit an den Fingerkuppen verfälschen die Ergebnisse.

Reproduzierbarkeit: Auch wenn bei der Datenerfassung penibel auf identische Bedingungen geachtet wird, können biometrische Merkmale sich von Mal zu Mal unterscheiden.

Beispiel: Menschen altern und ihr Körper verändert sich dabei (z. B. Falten oder Bartwuchs). Auch abrupte Veränderungen durch Unfälle (z. B. Handschrift durch Verletzungen) oder Krankheiten (z. B. Stimme bei Erkältung) können einen menschlichen Körper verändern.

Datenrepräsentation: Die Datenrepräsentation von biometrischen Merkmalen stellt eine besondere Herausforderung dar, da entschieden werden muss, welche Details eines biometrischen Merkmales für Vergleiche gespeichert werden sollen und welche als nicht relevant erachtet werden. Bei dieser Entscheidung gehen Informationen verloren und können zu einem späteren Zeitpunkt nicht mehr für Vergleiche genutzt werden. Werden hier die falschen oder zu wenige Daten gewählt, können zu einem späteren Zeitpunkt zwei ähnliche biometrische Merkmale nicht mehr voneinander unterschieden werden, wodurch eine klare Authentifizierung unmöglich wird.

3.1.3 Leistungsmetriken

Die biometrische Authentifizierung und die biometrische Verifikation haben viele gemeinsame Gesichtspunkte, da für eine erfolgreichen Authentifizierung zunächst eine Verifikation erfolgen muss. Eine der größten Herausforderungen bei biometrischen Merkmalen ist es die FAR (*Falschakzeptanzrate*, engl. *false acceptance rate*) und FRR (*Falschrückweisungsrate*, engl. *false rejection rate*) so gering wie möglich zu halten. Dabei verhalten sich FRR und FAR antiproportional, sodass eine niedrige FAR, als Wirkung eine hohe FRR erzeugt. Bei identischem FRR und FAR spricht man von der EER (*Gleichfehlerrate*, engl. *equal error rate*), die als Indikator für die Güte eines biometrischen Systems gesehen wird (Abb. 3.2).

Trotz zahlreicher Forschungsarbeiten existiert kein 100% exaktes Verfahren zur Verifikation von biometrischen Unterschriften. Dies liegt in der Natur von biometrischen Merkmalen (Unterabschnitt 3.1.2) und muss stets berücksichtigt werden. In der Literatur werden jedoch zahlreiche Annäherungen (um 2-5% FAR und FRR) entworfen und diskutiert. Meist unterscheiden sich die betrachteten globalen bzw. lokalen Merkmale, die Bewertungsfunktionen und die Trainingsfunktionen [KY05, KMA09].

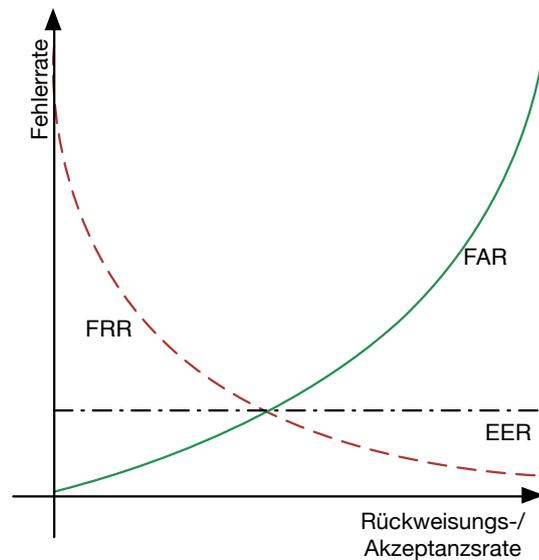


Abbildung 3.2: Zusammenhang der Leistungsmetriken

3.1.4 Templates

Um entscheiden zu können, ob ein biometrisches Merkmal authentisch ist, muss ein Vergleich mit sogenannten Templates erfolgen. Zur Erstellung von Templates existieren mehrere Wege, die bestimmte Vorteile, aber auch Nachteile, mit sich bringen.

Die trivialste Form von Templates ist eine umfassende Speicherung der aufgezeichneten biometrischen Merkmale. Durch ein solches Vorgehen behalten die Templates alle Informationen und können somit auch die besten Ergebnisse beim Matching erzielen. Dies ist jedoch höchst riskant, da ein solches „Rohdaten“-Template Rückschlüsse auf das biometrische Merkmal zulässt oder es komplett darstellt. Das Risiko rührt daher, dass Templates für jeden Benutzer in einer Datenbank abgelegt werden müssen. Könnten Unbefugte auf eine solche Datenbank Zugriff erlangen, wären sie in der Lage das biometrische Merkmal zu fälschen. Noch gravierender ist jedoch die Auswirkung auf den Benutzer hinter dem biometrischen Merkmal, da dieser sich z. B. eine neue Unterschrift angewöhnen müsste, was verglichen mit den Auswirkungen auf andere biometrische Merkmale noch harmlos wirkt.

Neben der erhöhten Gefahr durch Diebstahl können diese „Rohdaten“-Templates sehr groß werden, wodurch Speicherung und Matching aufwändiger werden. Eine Lösung dieses Problems ist die Spezifikation von relevanten Daten, die aus den Rohdaten abgeleitet werden (z. B. Globale Merkmale). Der Nachteil eines solchen Vorgehens ist der Informationsverlust und die daraus resultierende geminderte Leistung beim Matching. Dieser Leistungsverlust kann nur durch eine sorgfältige Auswahl von relevanten Merkmalen reduziert werden [MCN09].

Cancelable Biometrics

Einen Schritt weiter geht die kündbare Biometrie (*Cancelable Biometrics*) [RCB01], deren Ziel echte Einwegfunktionen zum Abbilden von Rohdaten sind. Durch solche Einwegfunktionen kann von den Templates nicht mehr auf die ursprünglichen biometrischen Merkmale geschlossen werden. Im Falle eines Diebstahles kann ein Template ohne weitere Folgen für ungültig erklärt und ersetzt werden. Die Leistung von solchen Verfahren ist jedoch schlechter, da mehr Toleranz beim Matching von biometrischen Merkmalen nötig wird [MCN09] [UPPJ04, S.953].

3.1.5 Biometrische Unterschriften

Aus der Menge der biometrischen Merkmale wird in dieser Masterarbeit die biometrische Unterschrift betrachtet. Biometrische Unterschriften werden mit sogenannten Unterschriftenpads erfasst, die mehr Informationen als nur die reine 2D-Grafik einer Unterschrift aufzeichnen. Werden diese erweiterten Informationen erhoben, wird von *On-line Verfahren* gesprochen. Bei *Off-line Verfahren* hingegen können auch bereits auf Papier vorliegende Unterschriften digitalisiert werden (eingescannt, abfotografiert) und zum Einsatz kommen.

Signaturlänge	Initiale Richtung	Stiftanhebung
Horizontale Position	Gesamtgeschwindigkeit	X Beschleunigung
Vertikale Position	X Geschwindigkeit	Y Beschleunigung
Druck	Y Geschwindigkeit	Log. Radius der Kurven
Tangentenwinkel (Pfad)	Gesamtbeschleunigung	Stiftazimut

Tabelle 3.3: Lokale Merkmale (Auswahl)

ØDruck	ØGeschwindigkeit	ØBeschleunigung
Min. Druck	Min. Geschw. (X,Y,Gesamt)	Min. Beschl. (X,Y,Gesamt)
Max. Druck	Max. Geschw. (X,Y,Gesamt)	Max. Beschl. (X,Y,Gesamt)
Var. Druck	Var. Geschwindigkeit	Var. Beschleunigung
Max.-Min. Druck	Quotient Geschw. (Ø÷Max.)	Punkt Max. Beschl.
Punkt Max. Druck	# Punkte mit pos X,Y-Geschw.	Seitenverhältnis (W:H)
Schreibdauer	Zeitpunkt max. Geschw.	Unterschrifthöhe (H)
# Punkte	# Stift abgesetzt	Unterschriftbreite (W)

Tabelle 3.4: Globale Merkmale (Auswahl)

Eine biometrische Unterschrift, die nach dem On-line Verfahren aufgezeichnet wird, bietet neben den Rohdaten *X/Y-Koordinate*, *Druck* und *Zeit* auch weitere abgeleitete

dynamische Merkmale. Diese werden als *lokale Merkmale* (zum Zeitpunkt der Erfassung erhoben, siehe Tab. 3.3) oder *globale Merkmale* (nach Erfassung der kompletten Unterschrift erhoben, siehe Tab. 3.4) kategorisiert [GM08].

In ISO/IEC 19794-7 [ISO07] wird ein Standard für Austauschformate von biometrischen Unterschriften beschrieben, der als Mindestvoraussetzung die *X,Y-Koordinate* und den zugehörigen *Zeitpunkt* umfasst.

3.1.6 Biometrisches Matching

Das biometrische Matching beschreibt den Prozess der Bestimmung der Übereinstimmung zwischen zwei biometrischen Templates. Üblicherweise wird eines bei der Ersterfassung (*Enrollment*) aufgezeichnet und ein weiteres bei der Durchführung einer biometrischen Authentifizierung (oder Verifizierung).

Dynamic Time Warping

Für das Matching biometrischer Unterschriften, die im *On-Line Verfahren* aufgezeichnet werden, hat sich das Verfahren der dynamischen Zeitverzerrung (*DTW*) als vielversprechender Ansatz erwiesen [WMD⁺13].

Erstmals wurde DTW in der automatischen Spracherkennung (*ASR*) eingesetzt [Vin68], bei der unterschiedlich lange Merkmalsvektoren menschlicher Sprache gegen Referenzmuster verglichen werden sollen. DTW ist jedoch nicht auf die Domäne der Spracherkennung beschränkt und kann für beliebige Wertreihen, die über einen Zeitraum erhoben werden, eingesetzt werden.

DTW-Algorithmus Gegeben seien zwei Zeitreihen X und Y der Länge m bzw. n ,

$$X = x_1, x_2, \dots, x_m$$

$$Y = y_1, y_2, \dots, y_n$$

zu denen ein Verzerrungspfad (*warping path*) W bestimmt werden soll,

$$W = w_1, w_2, \dots, w_K \text{ mit } \max(m, n) \leq K < (m + n)$$

Die Elemente des Verzerrungspfades $w_k = (i, j)_k$ entsprechen Indexpaaren einer $m \times n$ -Kostenmatrix, die paarweise für jeden Zeitpunkt die Distanzen der Zeitreihen enthält (Abb. 3.3). Damit die Berechnung der Pfade möglich ist muss der Verzerrungspfad beide Zeitreihen von Beginn in der Kostenmatrix Zelle $(1, 1)$ enthalten und am Ende beider Zeitreihen in der Kostenmatrix Zelle (m, n) enden. Zur Vermeidung unnötig

3 Verwandte Arbeiten

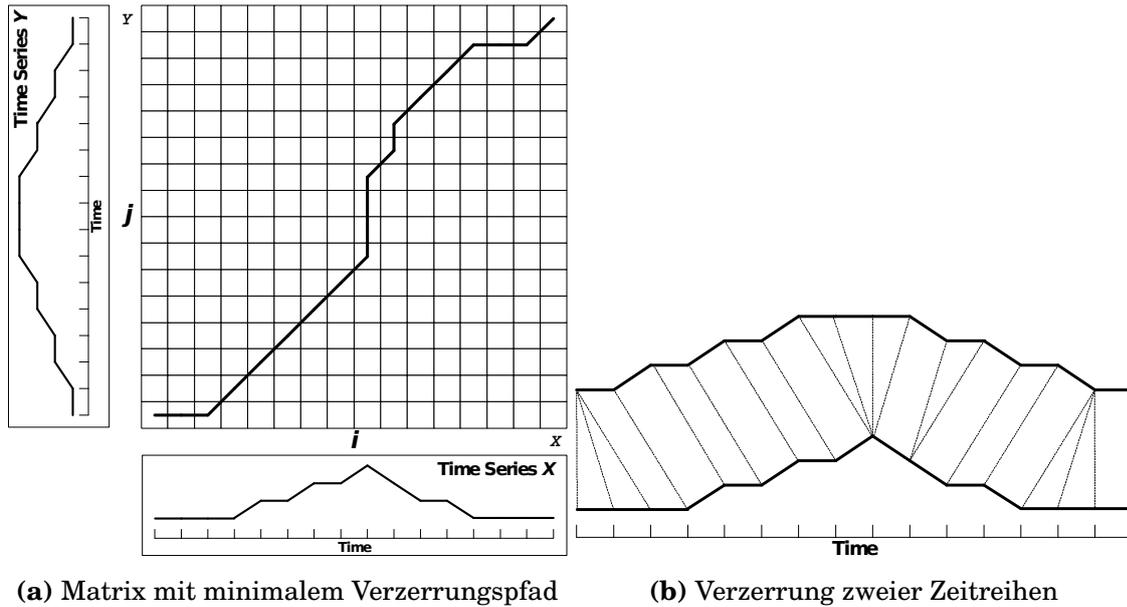


Abbildung 3.3: *Dynamic Time Warping* [SC07]

langer Pfade, muss der Verzerrungspfad monoton steigen, also nicht rückwärts in der Zeit gehen, aber trotzdem jeden einzelnen Zeitpunkt der beiden Zeitreihen enthalten:

$$w_1 = (1, 1)$$

$$w_K = (m, n)$$

$$w_k = (i, j), w_{k+1} = (i', j') \quad i \leq i' \leq i + 1, \quad j \leq j' \leq j + 1$$

Die Distanz der beiden Zeitreihen berechnet sich dann als Summe der Distanzen der Matrixzellen auf dem Verzerrungspfad

$$Dist(X, Y) = \sum_{k=1}^K dist(w_k)$$

Zur Minimierung dieser Distanz, muss der Verzerrungspfad möglichst auf Matrixzellen verweisen, welche niedrige Distanzen aufweisen und trotzdem noch einen Pfad von $(1, 1)$ nach (m, n) beschreiben. Beim *Dynamic Time Warping* wird dies über dynamische Programmierung gelöst

$$dist(i, j) = \begin{cases} d(x_1, y_1), & i = j = 1 \\ d(x_i, y_j) + \min(dist(i-1, j-1), dist(i-1, j), dist(i, j-1)), & i > 1, j > 1 \\ \infty & \text{sonst.} \end{cases}$$

wobei $d(x_i, y_j)$ meist als euklidischer Abstand $\|x_i - y_j\| = \sqrt{(x_i - y_j)^2}$ gewählt wird.

Kritik und Lösungsansätze Der DTW-Algorithmus ist in der erwähnten Form zwar exakt und liefert die geringste Verzerrungsdistanz, ist dabei aber nicht effizient. Da immer die komplette Kostenmatrix berechnet und im Speicher gehalten werden muss ist die Komplexität $O(m * n)$ (bzw. $O(n^2)$ bei ähnlich langen Zeitreihen).

Trotz der Bedingungen an den Verzerrungspfad, werden möglicherweise uninteressante (Teil-)Pfade berechnet, die sich von der Hauptdiagonalen stark entfernen. In der Literatur wird im wesentlichen auf drei Wegen versucht die Effizienz von *Dynamic Time Warping* zu steigern [SC07]:

- *Beschränkung* – Radius des *Warpwindow* auf einen Bereich um die Hauptdiagonale einschränken
- *Reduktion* – Abstrakte Version der Originaldaten, um „interessante“ Pfade zu erkennen und eine Suche auf Originaldaten gezielt durchführen zu können
- *Indizierung* – Einsatz *unterer Schranken*, um die Anzahl der DTW Durchgänge zu reduzieren (interessant für Clustering und Klassifikation)

FastDTW Eine Variante von *Dynamic Time Warping* ist FastDTW [SC07], die auf *Reduktion* der Daten in Kombination mit *Beschränkung* der berechneten Matrixzellen basiert.

Die Grundlage des Algorithmus ist ein Multilevel-Ansatz, bei dem die Zeitreihen vergrößert werden und zunächst für diese groben Versionen ein Verzerrungspfad berechnet wird. Sukzessiv nutzt der Algorithmus die gröberen Versionen, um die feinere Kostenmatrix nur in der Umgebung des groben Verzerrungspfades zu füllen und dann die feineren Verzerrungspfade zu berechnen (Abb. 3.4).

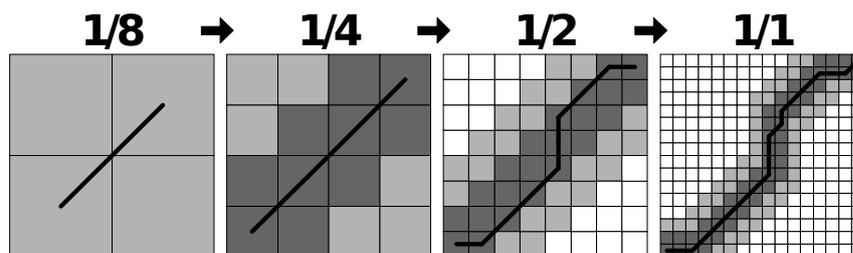


Abbildung 3.4: Kostenmatrizen der reduzierten Zeitreihen (grob → original) [SC07]

Die Komplexität kann durch dieses Vorgehen im Worst-Case auf $O(N * (4 * r + 7))$ also $O(N)$ abgeschätzt werden, wenn r (Suchradius) $\ll N$ ist.

3.1.7 Hardware

Die Erfassung von biometrischen Daten erfordert bei den meisten Merkmalen geeignete Sensoren, damit die aufgezeichneten Templates verschiedenen Qualitätsansprüchen

genügen. Diese Qualitätsansprüche steigern unter anderem die Beweiskraft der biometrischen Merkmale als Identifikationsmittel (vgl. Unterabschnitt 2.2.1). Vor dem Gesetz sind biometrische Daten immer Gegenstände der freien Beweiswürdigung, weswegen keine juristischen Voraussetzungen für eingesetzte Sensoren und Lesegeräte existieren. Um die Beweiskraft zu erhöhen, sind genauere (z. B. höhere Auflösung, höhere Abtastrate) biometrische Daten zu bevorzugen.

3.2 Secret-Sharing

Die *Geheimnisteilung* (*Secret-Sharing*) stellt ein Verfahren dar, mit dem ein Geheimnis D in n Stücke geteilt werden kann. Diese n Geheimnisanteile D_1, \dots, D_n werden nun an verschiedene Personen verteilt, damit

1. k dieser Personen gemeinsam das Geheimnis D sichtbar machen können und
2. $k - 1$ oder weniger dieser Personen keinerlei Rückschlüsse auf D ziehen können

Ein solches Schema wird als (k, n) -*Schwellwert-Schema* bezeichnet, da die Anzahl von (ehrlichen) Teilnehmer beim Wiederherstellen des Geheimnisses einen festgelegten Schwellwert überschreiten muss.

Beispiel: In einer Forschungseinrichtung werden streng geheime Dokumente in einem Tresor gelagert. Die Forschungseinrichtung hat fünf Abteilungen, welche jeweils einen Abteilungsleiter haben. Die Abteilungsleiter einigen sich darauf den Tresor nur mit mindestens drei Kollegen gemeinsam öffnen zu können, da in der Vergangenheit bereits Dokumente entwendet wurden. Soll der Tresor nun geöffnet werden müssen mindestens drei der fünf Abteilungsleiter ihren Geheimnisanteil preisgeben, damit die Zahlenkombination des Tresors rekonstruiert werden kann.

3.2.1 Shamir's Secret Sharing

In [Sha79] wurde bereits 1979 ein solches *Schwellwert-Schema* vorgestellt. Gegeben sei ein *explizit* als ganze Zahl repräsentiertes Geheimnis. Das Verfahren basiert auf linearen Gleichungen, in denen das Geheimnis *implizit* repräsentiert wird. Zunächst werden n lineare Gleichungen für das Geheimnis D aufgestellt, von denen t ausreichen, um D zu berechnen, aber $t - 1$ Anteile zu beliebigen Lösungen führen. Auch das BSI gibt Shamir's Secret Sharing als Lösung für Geheimnisteilung an [BSI13, S. 52ff].

Das Verfahren findet in einem endlichen Körper $(\mathbb{Z}_p, +, \cdot, 0, 1)$ statt, somit wird modulare Arithmetik bei der Berechnung eingesetzt. Die Primzahl p beschreibt hierbei die Größe des Körpers und muss als $p > D$ und $p > n$ gewählt werden. Die linearen

Gleichungen werden über Polynome mit Grad $t - 1$ konstruiert, wobei $a_0 = D$ und a_1, \dots, a_{t-1} zufällig aus \mathbb{Z}_p gewählt werden.

$$P(x) = a_0 + a_1x^1 + \dots + a_{t-1}x^{t-1}$$

Empfänger können nun durch diese Konstruktion nicht erkennen, ob ein Koeffizient zufällig gewählt wurde oder das Geheimnis a_0 ist. Für jede Person i wird nun ein solches Polynom berechnet und ein Anteil (*Share*) (x_i, y_i) durch

$$y_i = a_0 + a_1x_i^1 + \dots + a_{t-1}x_i^{t-1}$$

bestimmt, sodass alle *Shares* die gemeinsamen Unbekannten a_0, \dots, a_{t-1} haben. Sollen diese Unbekannten bestimmt werden, müssen mindestens t dieser *Shares* durch ein vereinbartes Protokoll gesammelt und das lineare Gleichungssystem mit den Unbekannten a_0, \dots, a_{t-1} gelöst werden, um die Unbekannte $a_0 = D$ zu berechnen:

$$\begin{aligned} y_{i_1} &= a_0 + a_1x_{i_1}^1 + \dots + a_{t-1}x_{i_1}^{t-1} \\ y_{i_2} &= a_0 + a_1x_{i_2}^1 + \dots + a_{t-1}x_{i_2}^{t-1} \\ &\dots \\ y_{i_{t-1}} &= a_0 + a_1x_{i_{t-1}}^1 + \dots + a_{t-1}x_{i_{t-1}}^{t-1} \end{aligned}$$

3.3 Mehrparteien-Signaturen

In der Literatur werden diverse Modelle vorgestellt, die es mehreren Unterzeichnern erlauben elektronische Dokumente gemeinsam zu unterzeichnen. So existieren die Mehrparteien-Vertragsunterzeichnungen (*Multi-Party Contract Signing (MPCS)*), die Protokolle zum gemeinsamen Unterzeichnen von elektronischen Dokumenten beschreiben. Ein Spezialfall der *MPCS* sind die sogenannten Multi-Signaturen (*Multi-Signatures*), die es mehreren Unterzeichnern ermöglichen, eine gemeinsame kompakte Signatur zu erzeugen [BN06, BJ10].

3.3.1 Multi-Party Contract Signing

Optimistische Protokolle

Optimistische *MPCS* Protokolle [BwW00, MR07, KR12] versuchen durch geschickten Austausch von Nachrichten zwischen den Parteien die Teilnahme einer dritten vertrauenswürdigen Partei (*Trusted Third Party (TTP)*) zu vermeiden. Die Bezeichnung „optimistisch“ wird dadurch motiviert, dass ein Protokoll nur mit ehrlichen Parteien ohne TTP auskommt. Beim Entwurf solcher Protokolle muss jedoch stets mit unehrlichen Parteien gerechnet werden, sodass eine TTP zumindest im Hintergrund bestehen

3 Verwandte Arbeiten

muss. Sie reagiert auf Ausnahmen, die von Parteien erhoben werden, löst diese auf und enttarnt unehrliche Parteien. Protokolle nach diesem Schema haben den Vorteil, dass es keinen zentralen Angriffspunkt beim Unterzeichnen von elektronischen Dokumenten gibt. Diesen Vorteil „erkaufen“ sich die Protokolle jedoch durch einen nicht unerheblichen Aufwand während der Unterzeichnung, da sich die einzelnen Parteien gegenseitig überwachen müssen.

Beispiel: In [BwW00] wird ein optimistisches Protokoll beschrieben, welches $O(t + 6)$ Runden und $O(tn^2)$ Nachrichten (t maximale Anzahl unehrlicher Parteien) benötigt. Die Autoren beschreiben ein *Asynchronous Optimistic Abuse-Free MPC*-Protokoll, welches auf asymmetrischer Verschlüsselung, digitalen Signaturen und Zertifikaten basiert. Für jede Ausführung des Protokolls wird für jede Partei ein frisches Schlüsselpaar generiert, welches durch ein Zertifikat an das eigentliche Schlüsselpaar gebunden wird und der TTP verschlüsselt mitgeteilt wird. Mit diesen neuen Schlüsselpaaren beginnen die Parteien das Protokoll. In mehreren Runden werden per Broadcast die Teilergebnisse der Runde zu jeder Partei versendet.

- Wird die letzte Runde erfolgreich beendet, kann ein „Vorvertrag“ aus den zusammengeführten Teilergebnissen generiert werden. Erhalten die Parteien einen solchen, geben sie ihr Zertifikat per Broadcast preis und ermöglichen die Umwandlung in den echten Vertrag.
- Eine Partei hat eine Ausnahmebehandlung durch die TTP ausgelöst. Nun entscheidet die TTP, ob die Ausnahme zum Abbruch führt oder gelöst werden kann.

Anmerkung: „Pessimistische“ Protokolle nutzen stets eine TTP. Der trivialste Fall für ein pessimistisches Protokoll ist somit das alle Parteien eine Nachricht von der TTP erhalten, diese signieren und anschließend zurück zur TTP senden. Diese bildet eine Konkatenation aller empfangenen Teilsignaturen und signiert diese, um die Integrität sicherzustellen.

3.4 Fazit

In diesem Kapitel wurden bestehende Ansätze aus der Literatur beschrieben, die zur Lösung der Problemstellung in dieser Arbeit beitragen.

Um die Nutzung von Unterschriftenpads zu ermöglichen, müssen die biometrischen Daten erfasst werden. Bei der Recherche in der Literatur ist offensichtlich, dass die Möglichkeiten einer solchen Hardware nur durch sogenannte *On-line Verfahren* (Abschnitt 3.1.5) ausgeschöpft werden können. Zum Matching von biometrischen Unterschriften hat sich in der Literatur das Verfahren *Dynamic Time Warping* bewährt (Unterabschnitt 3.1.6).

Das Themenfeld der *Mehrparteien-Signaturen* wird in der Literatur zum Großteil durch optimistische Protokolle für *Multi-Party Contract Signing* beschrieben. Zwar ist der Ansatz eine TTP nicht zwangsweise in jeder Ausführung des Protokoll vorzusetzen interessant, er hat jedoch einige Nachteile. Wenn die Parteien direkt miteinander kommunizieren, müssen sie sich über das Internet miteinander verbinden können. Dies ist jedoch nur möglich, wenn Verbindungen nicht von Firewalls blockiert werden. In den beschriebenen Protokollen wird außerdem nicht erklärt, wie die gemeinsam zu signierenden Dokumente an die einzelnen Parteien übertragen werden sollen. Intuitiv müsste hierzu eine weitere TTP bestehen, die jedoch zwingend in ein Protokoll involviert ist.

Anstatt ein optimistisches Protokoll mit einer großen Anzahl von Runden einzusetzen, soll in dieser Arbeit ein Protokoll entwickelt werden, welches mit einer TTP funktioniert. Damit mehrere Parteien in einem solchen Protokoll kooperativ eine Signatur erzeugen können, kann das sogenannte *Secret-Sharing* eingesetzt werden, bei dem ein Geheimnis in mehrere Geheimnisteile gespalten wird. Mit dem Verfahren *Shamir's Secret Sharing* wird in diesem Kapitel ein solches Verfahren beschrieben. Es kann dazu genutzt werden ein Geheimnis in eine feste Anzahl von Teilen zu zerlegen. Nur wenn alle diese Geheimnisteile vorliegen kann eine Rekonstruktion erfolgreich durchgeführt werden. Ein solches Geheimnis kann also prinzipiell der Hashwert eines Dokumentes sein, welcher nur dann rekonstruiert werden kann, wenn die Teilgeheimnisse von allen Teilnehmern unterzeichnet wurden.

3 Verwandte Arbeiten

4 Anforderungsanalyse

In diesem Kapitel werden die Anforderungen an das zu entwerfende System definiert. Zunächst wird hierzu die Problemstellung erläutert, die es zu lösen gilt. Aus dieser Problemstellung werden dann die Anforderungen detaillierter herausgearbeitet.

4.1 Problemstellung

Ein Anwendungsszenario der Telemedizin sind die Telekonferenzen, mit denen es Ärzten ermöglicht wird mit einer Anwendung über das Internet zu kommunizieren und patientenbezogene Beratung (*Telekonsile*) zu realisieren. Um die Ergebnisse eines *Telekonsils* mit Haftungsrelevanz zu erhalten, müssen die Teilnehmer an einem Protokoll teilnehmen können, welches Mehrparteien-Signaturen erzeugt. Um eine erhöhte Sicherheit zu gewährleisten, soll das Protokoll in einem System durchgeführt werden, das biometrische Unterschriften in Kombination mit Passwörtern nutzt. Diese Sicherheitsmerkmale können dann durch biometrische Schlüsselfreigabe für eine sichere Erzeugung fortgeschrittener elektronischer Signaturen genutzt werden.

4.2 Funktionale Anforderungen

Das zu entwickelnde System muss den Anwendern eine einfache Möglichkeit bieten, um Dokumente in der verteilten Umgebung gemeinsam zu signieren. Dabei soll es mit bestehenden asymmetrischen Kryptografieverfahren umsetzbar sein. Hierbei ist der Einsatz von elektronischen Signaturen unabdingbar und wird im Rahmen dieser Masterarbeit mindestens den fortgeschrittenen Signaturen (Unterabschnitt 2.2.1) genügen. Die biometrischen Daten sollen mit einem Unterschriftenpad (Unterabschnitt 6.1.3) aufgezeichnet und dann zur Authentifizierung eingesetzt werden.

Konkret werden die Komponenten dann so umgesetzt, dass sie mit der Telekonferenz-Lösung HEALTHTELKON verwendet werden können. Dazu sollen die nach einer Konferenz entstandenen Protokolle von allen Teilnehmer gemeinsam signiert werden.

FA1 Lokale Schlüsselerzeugung

Die Generierung von Schlüsselpaaren für die Nutzung im System muss entkoppelt von einem entfernten Server realisiert werden. Die Generierung von Schlüsselpaaren wird somit auf dem lokalen PC des Anwenders durchgeführt.

FA2 Schlüsselverwaltung (öffentliche Schlüssel)

Das System verwaltet die öffentlichen Schlüssel der Benutzer. Es dient somit als Trusted Third Party (TTP), welche die Schlüssel der Benutzer zertifiziert und für andere Benutzer zugänglich macht.

FA3 Dokumente katalogisieren

Das System hält einen Katalog über sämtliche Dokumente, die für eine Mehrparteien-Signatur eingestellt wurden. Zusätzlich legt das System signierte Prüfsummen für alle Dokumente ab, sodass die Integrität gewahrt bleibt.

FA4 Dokumente bereitstellen

Das System stellt einem Benutzer Dokumente bereit, bei denen er sich an Mehrparteien-Signaturen beteiligen soll. Das System stellt zusätzlich eine signierte Prüfsumme der Dokumente bereit, mit der die Integrität überprüft werden kann.

FA5 Zugriffskontrolle für Dokumente

Für eingestellte Dokumente können Zugriffsrechte vergeben werden. Diese Zugriffsrechte werden von einem administrativen Benutzer für andere Benutzer vergeben. Hat ein Benutzer Zugriff auf ein Dokument, ist er ebenfalls für eine Mehrparteien-Signatur an diesem Dokument vorgesehen.

FA6 Benutzerverwaltung

Die Benutzer des Systems werden zentral verwaltet. Neben den Stammdaten (Name, Kontaktdaten, Fachbereich) und Benutzerkennwort, wird einem Benutzerdatensatz eine eindeutige Identifikationsnummer zugewiesen. Diese Identifikationsnummer wird fest vergeben und ändert sich nicht mehr.

FA7 Authentifizierung im System

Das System bietet den Benutzern die Möglichkeit sich am System mit einer Kombination aus Benutzername und Benutzerkennwort zu authentifizieren.

FA8 Benutzerrollen

Das System sieht drei Benutzerrollen vor. Den administrativen Benutzer eines Dokumentes, die an der Mehrparteien-Signatur teilnehmenden Benutzer und Prüfnutzer. Die Benutzerrollen können für jedes Dokument unterschiedlich gewählt werden.

FA9 Biometrische Unterschriften erfassen

Das System kann biometrische Signaturen über ein Unterschriftenpad im *On-Line Verfahren* erfassen.

FA10 Biometrische Unterschriften aufzeichnen

Das System kann biometrische Signaturen in einer Datenbank als verschlüsselte Daten aufzeichnen.

FA11 Biometrische Unterschriften abgleichen

Das System kann biometrische Signaturen vergleichen und Übereinstimmungen finden.

4.3 Nicht-funktionale Anforderungen

Nicht-funktionale Anforderungen beschreiben welche Eigenschaften das System haben soll, jedoch nicht die Funktionen des Systems selbst.

NFA1 Bedienbarkeit über einen Webbrowser

Die Bedienung des Systems soll über einen Webbrowser möglich sein. Etwaige Erweiterungen des Systems sollen ebenfalls über einen bzw. in einem Webbrowser gestartet werden (z. B. durch Plug-Ins). Das System soll als ein Web-Portal realisiert werden.

NFA2 Systemvoraussetzungen für Benutzer

Das zu implementierende System soll mindestens auf PCs mit MICROSOFT WINDOWS Betriebssystem (ab WINDOWS 7) mit einer ORACLE JAVA 7 Laufzeitumgebung lauffähig sein. Als Webbrowser sollen entweder GOOGLE CHROME¹ oder MOZILLA FIREFOX² genutzt werden, die den Einsatz von JAVA Webstart genehmigen. Werden für Erweiterungen des Systems spezielle Endgeräte (z. B. Unterschriftenpad, Fingerprints scanner, etc.) benötigt, sind diese ebenfalls vorausgesetzt.

¹GOOGLE CHROME <http://www.google.com/chrome> (Stand: 12.10.13)

²MOZILLA FIREFOX <http://www.mozilla.org/firefox> (Stand: 12.10.13)

4 Anforderungsanalyse

Die für die Telekonferenzen nötigen Kanäle (z. B. Internet, KV-SAFENET³, VPN, etc.) werden für das System ebenfalls benötigt. Gelten besondere Sicherheitsanforderungen an die Kommunikationskanäle für die Primärsysteme, kann nur so den rechtlichen Vorgaben entsprochen werden.

NFA3 Interoperabilität

Da das System in verschiedenen medizinischen Einrichtungen eingesetzt werden soll und jede über eine andere technische Infrastruktur verfügen kann, ist Interoperabilität eine wichtige Anforderung. Hierzu soll das System eine gewisse Modularität für bestimmte Komponenten bereitstellen. So können verschiedene Authentifizierungsmethoden (z. B. mittels Biometrie, SmartCard, USB-Token, etc.) durch Erweiterungen in das System integriert werden.

NFA4 Rechtliche Vorgaben

Das in dieser Masterarbeit beschriebene System soll mit den kryptografischen Grundbausteinen umgesetzt werden, die von der Bundesregierung empfohlen werden (Unterabschnitt 2.2.3). Bei der Wahl der elektronischen Signaturen (Unterabschnitt 2.2.1) wird das beschriebene System mit fortgeschrittenen elektronischen Signaturen (Unterabschnitt 2.2.1) arbeiten, die ohne externe Zertifizierung und Akkreditierung erstellt werden können.

Patientendaten

Die Verwendung von Patientendaten in einem System zur elektronischen Datenverarbeitung ist in Deutschland immer mit einer Einwilligung der Patienten verbunden. Das hier entworfene System erhebt jedoch keine Patientendaten und verarbeitet nur Dokumente, die von einem vorgeschaltetem System generiert wurden. Somit wird die Verantwortung zum Umgang mit Patientendaten und somit das Einholen einer Patienteneinwilligung von diesem vorgeschalteten System übernommen.

NFA5 Sicherheitsanforderungen

Das entworfene System soll durch technische Mittel gegen Angriffe geschützt werden. Insbesondere soll es durch Angriffe praktisch nicht möglich sein Signaturen zu fälschen (Authentizität) oder Dokumente unbemerkt zu manipulieren (Integrität).

³Kassenärztliche Bundesvereinigung: KV-SAFENET <http://www.kbv.de/24874.html> (Stand: 13.10.13)

4.4 Anwendungsfälle

Die Anwendungsfälle sollen das System aus der Sicht eines Akteurs, also eines Benutzer beschreiben. In Abb. 4.1 werden diese Anwendungsfälle dargestellt und in Beziehung gesetzt.

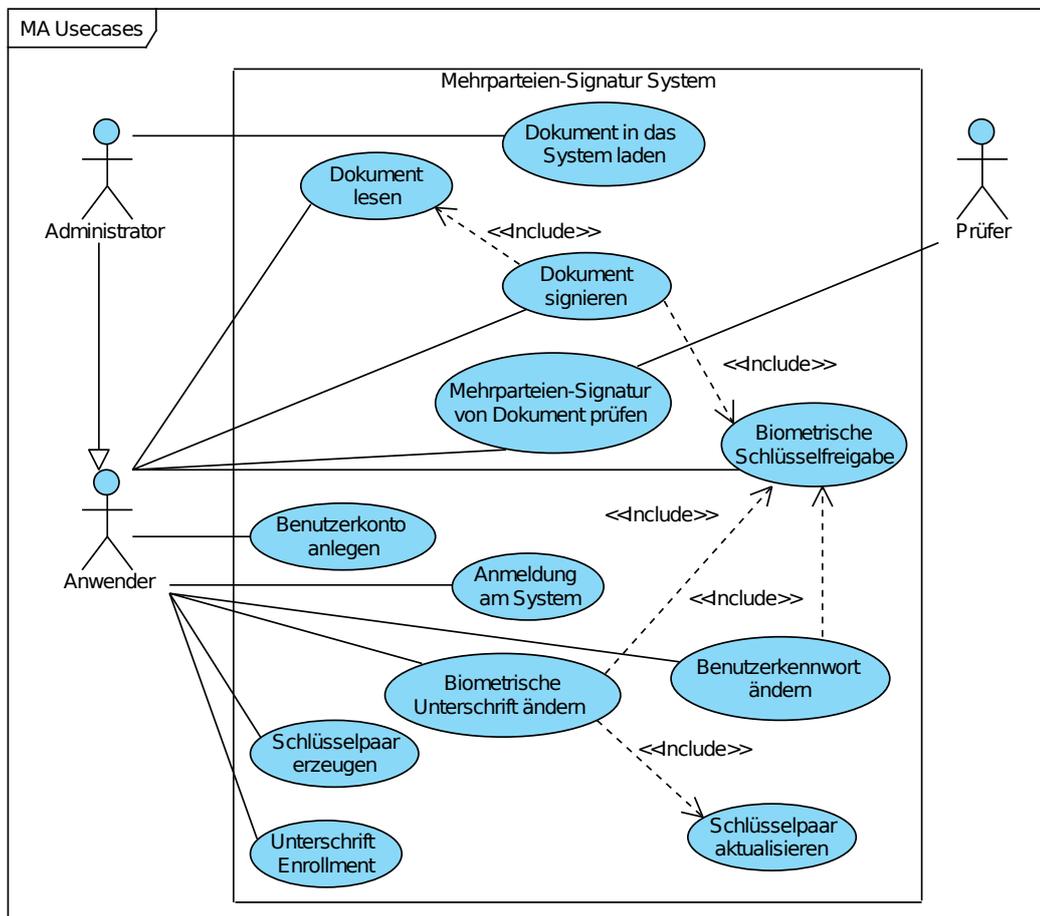


Abbildung 4.1: Use Cases des Gesamtsystems

Benutzerkonto anlegen

UC1	Benutzerkonto anlegen
Kurzbeschreibung	Jeder Benutzer, welcher an einer Mehrparteien-Signatur teilnehmen möchte hat in der Regel bereits ein Benutzerkonto für HEALTHTELKON. Bevor ein Benutzer an der Erstellung von Signaturen teilnehmen kann, muss er sich in dem neuen System registrieren und sein HEALTHTELKON Benutzerkonto verknüpfen.
Primärer Akteur	Benutzer
Vorbedingung	Benutzer hat bereits ein Benutzerkonto in HEALTHTELKON.
Nachbedingung	Benutzer hat ein Benutzerkonto im System. Ihm wurde eine UUID zugewiesen.
Ergebnis	Der Benutzer erhält eine visuelle Bestätigung, dass sein Benutzeraccount angelegt wurde.

Tabelle 4.1: Benutzerkonto anlegen

Anmeldung im System

UC2	Anmeldung im System
Kurzbeschreibung	Hat ein Benutzer ein Benutzerkonto angelegt, kann er sich mit seinem Benutzernamen und Benutzerkennwort im System anmelden.
Primärer Akteur	Benutzer
Vorbedingung	Benutzer hat bereits ein Benutzerkonto angelegt.
Nachbedingung	Benutzer ist im System angemeldet. Ihm wurde eine SessionID zugewiesen.
Ergebnis	Der Benutzer erhält eine visuelle Bestätigung, dass er erfolgreich angemeldet wurde.
Qualitäten	Sichere Übertragung von Benutzername und Benutzerkennwort

Tabelle 4.2: Anmeldung im System

Unterschrift Enrollment

UC3	Unterschrift Enrollment
Kurzbeschreibung	Jeder Benutzer erstellt für sein Benutzerkonto eine biometrische Unterschrift. Dazu unterschreibt er 5 mal auf dem Unterschriftenpad und bestätigt die Eingabe mit seinem Benutzerkennwort.
Primärer Akteur	Benutzer
Vorbedingung	Benutzer hat sich im System angemeldet.
Nachbedingung	Das biometrische Unterschriften-Template wurde aufgezeichnet und gespeichert.
Ergebnis	Der Benutzer erhält eine visuelle Bestätigung, dass das Enrollment erfolgreich war.
Qualitäten	Bei Speicherung ist das biometrische Unterschriften-Template verschlüsselt.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System

Tabelle 4.3: Unterschrift Enrollment

Biometrische Schlüsselfreigabe

UC4	Biometrische Schlüsselfreigabe
Kurzbeschreibung	Möchte ein Benutzer seinen privaten Schlüssel für eine Mehrparteien-Signatur nutzen, muss er ihn durch Eingabe seiner biometrischen Unterschrift freigeben.
Primärer Akteur	Benutzer
Vorbedingung	Benutzer hat sich im System angemeldet und ein biometrische Unterschriften-Template erzeugt.
Nachbedingung	Der Benutzer kann seinen privaten Schlüssel benutzen.
Ergebnis	Der Benutzer erhält eine visuelle Bestätigung, dass der private Schlüssel genutzt werden kann.
Qualitäten	Der private Schlüssel kann für eine Aktion genutzt werden.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System.

Tabelle 4.4: Biometrische Schlüsselfreigabe

Schlüsselpaar erzeugen

UC5	Schlüsselpaar erzeugen
Kurzbeschreibung	Jeder Benutzer muss die Möglichkeit haben sich selbst ein Schlüsselpaar zu erzeugen. Dazu meldet sich der Benutzer mit seinem Benutzerkonto in der Anwendung an, öffnet die Schlüsselverwaltung und wählt die entsprechende Funktion aus. Zur Generierung dieses Schlüsselpaares muss der Benutzer seine biometrische Unterschrift und sein Benutzerkennwort eingeben. Der öffentliche Schlüssel wird dann auf den Schlüsselsever übertragen, der private Schlüssel wird dem Benutzer als verschlüsselte Datei bereitgestellt.
Primärer Akteur	Benutzer
Vorbedingung	Benutzer hat sich im System angemeldet und ein biometrisches Unterschriften-Template erzeugt.
Nachbedingung	Das Schlüsselpaar wurde erzeugt und der öffentliche Schlüssel auf dem Schlüsselsever gespeichert.
Ergebnis	Der Benutzer erhält eine visuelle Bestätigung, dass er erfolgreich ein Schlüsselpaar erzeugt hat. Der Benutzer erhält seinen privaten Schlüssel als Datei.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System, UC4: Biometrische Schlüsselfreigabe.

Tabelle 4.5: Schlüsselpaar erzeugen

Biometrische Unterschrift ändern

UC6	Biometrische Unterschrift ändern
Kurzbeschreibung	Möchte ein Benutzer seine biometrische Unterschrift ändern, muss er zunächst seine bisherige biometrische Unterschrift bestätigen. Ist diese Bestätigung erfolgreich, folgt ein <i>Unterschrift Enrollment</i> (Unterabschnitt 4.4). Die Anwendung fordert den Benutzer dann erneut auf die bisherige Unterschrift und sein Benutzerkennwort einzugeben, um den privaten Schlüssel zu aktualisieren.
Primärer Akteur	Benutzer
Vorbedingung	Benutzer hat sich im System angemeldet und ein biometrisches Unterschriften-Template erzeugt.
Nachbedingung	Die biometrische Unterschrift wurde aktualisiert.
Ergebnis	Der Benutzer erhält eine visuelle Bestätigung, darüber dass er seine biometrische Signatur aktualisiert hat. Der Benutzer erhält seinen aktualisierten privaten Schlüssel als Datei.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System, UC3: Unterschrift Enrollment, UC4: Biometrische Schlüsselfreigabe.

Tabelle 4.6: Biometrische Unterschrift ändern

Benutzerkennwort ändern

UC7	Benutzerkennwort ändern
Kurzbeschreibung	Möchte ein Benutzer sein Benutzerkennwort ändern, muss er zunächst seine bisherige biometrische Unterschrift bestätigen. Ist diese Bestätigung erfolgt, fordert die Anwendung den Benutzer auf sein bisheriges Benutzerkennwort einzugeben und das neue Benutzerkennwort zu wählen. Nun aktualisiert die Anwendung das biometrische Template und das Benutzerkennwort.
Primärer Akteur	Benutzer
Vorbedingung	Benutzer hat sich im System angemeldet und ein biometrisches Unterschriften-Template erzeugt.
Nachbedingung	Das Benutzerkennwort wurde aktualisiert.
Ergebnis	Der Benutzer erhält eine visuelle Bestätigung, dass er sein Benutzerkennwort aktualisiert hat.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System, UC4: Biometrische Schlüsselfreigabe.

Tabelle 4.7: Benutzerkennwort ändern

Dokument in das System laden

UC8	Dokument in das System laden
Kurzbeschreibung	Im Normalfall gibt es einen Benutzer mit administrativen Rechten für jede Konferenz (<i>Konferenzmanager</i>), welcher ein Dokument bereitstellt. Zu einem Dokument fügt der <i>Konferenzmanager</i> die nötigen Metadaten der Konferenz hinzu. Diese Informationen an den Server sind die potenziellen Teilnehmer an der Mehrparteien-Signatur. Dann wird das Dokument über einen sicheren Kanal zum Signaturserver übertragen.
Primärer Akteur	Konferenzmanager
Vorbedingung	Konferenzmanager hat sich im System angemeldet.
Nachbedingung	Das Dokument wurde in das System geladen und ist für Benutzer verfügbar.
Ergebnis	Der Konferenzmanager erhält eine visuelle Benachrichtigung, dass ein Dokument erfolgreich in das System geladen wurde. Die beteiligten Benutzer werden im System vorgemerkt.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System.

Tabelle 4.8: Dokument in das System laden

Dokument lesen

UC9	Dokument lesen
Kurzbeschreibung	Die Benutzer müssen ein Dokument gelesen haben, bevor sie eine Signatur erzeugen können. Um dies zu tun, müssen sie ein zuvor bereitgestelltes Dokument vom Server anfordern. Auch bei dieser Übertragung muss beim Empfang die Integrität der versendeten Dokumente überprüft werden, da ansonsten nicht alle Teilnehmer garantiert am selben Dokument eine elektronische Signatur vornehmen. Die Anwendung signalisiert dem Benutzer die Integrität des empfangenen Dokumentes.
Primärer Akteur	Benutzer
Vorbedingung	Der Benutzer hat sich im System angemeldet. Der Benutzer ist für das Dokument als Teilnehmer vorgemerkt.
Nachbedingung	Der Benutzer kann das Dokument lesen.
Ergebnis	Der Benutzer erhält das angeforderte Dokument als Datei.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System.

Tabelle 4.9: Dokument lesen

Dokument signieren

UC10	Dokument signieren
Kurzbeschreibung	Nachdem ein Dokument gelesen wurde, können die Benutzer dem Server ihre Bereitschaft für eine Mehrparteien-Signatur signalisieren. Der Server stellt hierbei für alle Anwender dar, wer letztendlich an der Signatur teilnehmen wird und startet das Mehrparteien-Signaturprotokoll. Die Benutzer können den Fortschritt des Protokolls in der Anwendung verfolgen.
Primärer Akteur	Benutzer
Vorbedingung	Der Benutzer hat sich im System angemeldet. Der Benutzer ist für das Dokument als Teilnehmer vorgemerkt. Der Benutzer hat ein Schlüsselpaar erzeugt.
Nachbedingung	Der Benutzer hat seine Teilsignatur erzeugt.
Ergebnis	Der Benutzer erhält einen visuellen Hinweis darüber, dass er ein Dokument signiert hat.
Beziehungen zu anderen Use Cases	«include» UC2: Anmeldung im System, UC4: Biometrische Schlüsselfreigabe, UC9: Dokument lesen.

Tabelle 4.10: Dokument signieren

Signatur von Dokument prüfen

UC11	Signatur von Dokument prüfen
Kurzbeschreibung	Möchte ein Prüfer die Mehrparteien-Signatur eines Dokumentes überprüfen, überträgt er ein Dokument samt Signatur an die Anwendung. Diese überprüft die Signatur und informiert den Prüfer, ob die Signatur als Ganzes gültig ist oder nicht.
Primärer Akteur	Prüfer
Vorbedingung	Der Prüfer hat sich im System angemeldet. Der Prüfer hat ein signiertes Dokument als Datei vorliegen.
Ergebnis	Der Prüfer erhält Auskunft darüber, ob die Signatur gültig ist oder nicht.
Beziehungen zu anderen Use Cases	–

Tabelle 4.11: Signatur von Dokument prüfen

4 Anforderungsanalyse

5 Entwurf

Im folgenden Kapitel soll der Entwurf für ein Mehrparteien-Signatursystem für Dokumente beschrieben werden, das den Anforderungen (Kapitel 4) entspricht. Hierzu wird zunächst eine Systemarchitektur entworfen und auf Designentscheidungen für ein solches System eingegangen. Das System wird im folgenden Handwritten Multiparty Document Signature (HYDrOUS) genannt. Für das zu entwerfende System werden in diesem Kapitel außerdem die zentralen Protokolle beschrieben, mit denen Mehrparteien-Signaturen und die biometrische Schlüsselfreigabe durchgeführt werden können.

5.1 Entwurf der Systemarchitektur

Beim Entwurf der Systemarchitektur soll zunächst entschieden werden, wie das zu implementierende System aufgebaut wird. Hierzu werden die zugrundeliegenden technischen Ansätze beschrieben und dann die Komponenten detaillierter betrachtet.

5.1.1 Desktopanwendung versus Webanwendung

Steht man heute vor der Frage, ob eine Software als Desktopanwendung oder als Webanwendung implementiert werden soll, gilt es die Vorteile und Nachteile zu bewerten.

Webanwendungen

Webanwendungen (Abb. 5.1) werden zu Großteilen auf einem Server ausgeführt, der die Daten für die Repräsentation im Browser aufbereitet. Im Browser der Anwender werden in der Regel nur die Darstellung und kleinere Berechnungen durchgeführt. Durch diese Trennung von Oberfläche (*View*) und Anwendungslogik (*Controller*, *Business Logic*) ergeben sich Vorteile.

Vorteile: Webanwendungen können Aktualisierungen direkt auf den Servern erhalten, wodurch sie schneller zu allen Benutzern ausgerollt werden können und mögliche Sicherheitslücken in kürzerer Zeit behoben werden können. Durch diese von den Benutzer PCs entkoppelten Aktualisierungen können die Wartungskosten gesenkt werden. Webanwendungen bieten durch Standards wie HTML, CSS, ECMAScript

5 Entwurf

(JavaScript) und der Verfügbarkeit von standardkonformen Browsern weitestgehend Plattformunabhängigkeit. Die Daten sind überall verfügbar, wo eine Verbindung zum Server besteht und müssen nicht manuell auf andere PCs transferiert werden.

Nachteile: Webanwendungen können sich nur im Rahmen der Möglichkeiten bewegen, die ihnen durch Webstandards bereitstehen. Benötigen sie weitergehende Funktionalität, wie den direkten Zugriff auf Endgeräte, werden Erweiterungen für den Zugriff auf Ebene des Betriebssystems benötigt. Diese Erweiterungen können selbst Sicherheitslücken enthalten und müssen über herkömmliche Wege Aktualisierungen beziehen. In älteren Browsern werden Webstandards nicht zwangsweise befolgt, wodurch bei der Umsetzung einer umfassenden Plattformunabhängigkeit diverse Versionen von Browsern eingeplant werden müssen und ein erhöhter Implementierungsaufwand entsteht. Ein weiterer Nachteil ist, dass sämtliche Daten vom Server an den Browser übertragen werden müssen und stets eine Verbindung zum Server bestehen muss. Da dies auch sensible Daten sein können, muss mittels Verschlüsselung und Authentifizierung gegenüber dem Server ein sicherer Kommunikationskanal bereitgestellt werden.

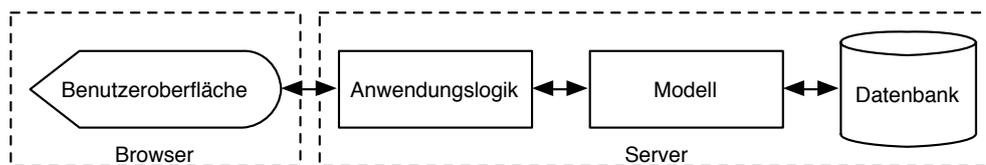


Abbildung 5.1: Webanwendung

Desktopanwendungen

Desktopanwendungen (Abb. 5.2) werden auf dem lokalen PC des Benutzers ausgeführt und müssen manuell installiert oder kopiert werden. Anwendungen, die nicht auf die Kommunikation mit anderen Clients angewiesen sind, können vollständig auf dem lokalen PC des Benutzer ausgeführt werden, ohne Kommunikation mit externen Systemen.

Vorteile: Desktopanwendungen sind für interaktive, multimediale und zeitkritische Anwendungen meist die bessere Wahl. Sie können näher mit dem Betriebssystem zusammenarbeiten und unterliegen weniger starken Restriktionen. Somit können sie über Gerätetreiber direkt mit Endgeräten kommunizieren.

Nachteile: Desktopanwendungen müssen Aktualisierungen über einen vertrauenswürdigen Kanal beziehen. Dies kann über zentrale Aktualisierungen für Betriebssysteme, über eigene Lösungen oder gänzlich manuell durchgeführt werden. Bei

Desktopanwendungen können zwischen den Clients verschiedene Versionen auftreten, da Aktualisierungen verzögert oder überhaupt nicht installiert werden.

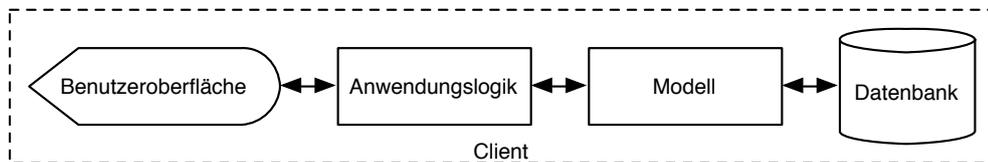


Abbildung 5.2: Desktopanwendung

Hybridlösungen

Hybridlösungen (Abb. 5.3) können als sinnvolle Alternative dienen, da sie optimierte Benutzeroberflächen bieten und trotzdem nicht auf lokale Daten angewiesen sind. Hybridlösungen sind vor allem auf Geräteklassen, wie Smartphones oder Tablet PCs sinnvoll, da deren Betriebssysteme einheitliche Aktualisierungsmechanismen bereitstellen. Die Daten werden in den meisten Fällen auf Servern vorverarbeitet und bei Bedarf an die Clients übermittelt.

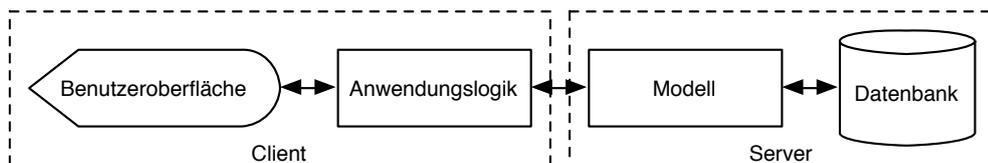


Abbildung 5.3: Hybridlösung

Fazit

Für die Implementierung eines Mehrparteien-Signatursystems eignet sich eine Webanwendung, da die sensiblen Dokumente so auf einem gesicherten Server abgelegt werden können. Die Benutzer können HYDrOUS ohne erheblichen Wartungsaufwand einsetzen und stets auf dem aktuellsten Stand der Entwicklung arbeiten. Da eine Anforderung die biometrische Authentifizierung mit einem Unterschriftenpad ist, ist der Einsatz einer Erweiterung nötig, die auf Treiber eines solchen Unterschriftenpads Zugriff erlangt. Eine weitere Anforderung stellt die lokale Schlüsselerzeugung dar, die Zugriff auf das lokale Dateisystem haben muss und mit der biometrischen Erweiterung kommunizieren muss. HYDrOUS wird also als eine Webanwendung mit Erweiterungen für biometrische Unterschriftenpads und lokale Schlüsselerzeugung konzipiert.

5.1.2 Webservice Technologien

Im wesentlichen konkurrieren zum jetzigen Zeitpunkt zwei Möglichkeiten Webservices zu implementieren. Zum einen Representational State Transfer (REST)-Webservices und zum anderen die SOAP-Webservices.

REST-Webservices

REST-Webservices verfolgen das Ziel Webservices über das Hypertext-Transfer-Protocol (HTTP) und dessen Möglichkeiten zu realisieren. Anders als SOAP ist REST kein Standard, der Vorgaben zur Implementierung von Komponenten der Webservices macht. Stattdessen sollte REST als Architekturstil beim Entwurf von Webservices angesehen werden, der bestimmte Eigenschaften als Konvention (mehr Details z. B. in [Til11]) vorgibt.

Zustandslosigkeit Ein REST-Webservice ist stets zustandslos, wodurch keine Informationen über den Client serverseitig zwischengespeichert werden. Durch letztere Eigenschaft muss jede Anfrage an einen REST-Webservice alle Informationen mitführen, die für eine erfolgreiche Beantwortung benötigt werden.

Ressourcen Jeder REST-Webservice hat eine eindeutige, möglichst aussagekräftige Adresse (URL). Dabei sollte über mehrere Webservices hinweg auf ein ähnliches Schema gesetzt werden, damit eine konsistente Adressierbarkeit besteht. Unterhalb dieser Adresse befinden sich stets *Ressourcen*, denen eine eindeutige URI zugewiesen wird. Diese sollten semantisch deutlich machen, ob sie eine einzelne Ressource sind oder eine Sammlung von Unterressourcen.

Beispiel: „POST `http://beispiel.org/bookstore/books`“ – neues Buch in den Buchladen einpflegen,

„GET `http://beispiel.org/bookstore/book/12345`“ – Buch 12345 anfordern.

HTTP Operationen Sämtliche Aufrufe eines REST-Webservice erfolgen über die HTTP-Operationen GET, POST, PUT und DELETE. Diese Methoden werden für die Implementierung von Webservices, die das Create Read Update Delete (CRUD) Paradigma umsetzen, wie in Tab. 5.1 assoziiert. Die Methoden GET, PUT und DELETE sind idempotent, sodass ein mehrfacher Aufruf immer den selben Effekt hat.

Anmerkung: PUT ersetzt oder erstellt eine Ressource mit gegebenem Bezeichner (Index), wohingegen POST immer eine neue Unterressource hinzufügt (nicht idempotent!).

CRUD	REST
Create (Erstellen)	PUT, POST
Read (Lesen)	GET
Update (Aktualisieren)	PUT
Delete (Löschen)	DELETE

Tabelle 5.1: CRUD Operationen und die entsprechenden REST Operationen

Repräsentation Der Datenaustausch mit einem REST-Webservice unterliegt keinem vorgeschriebenem Dateiformat. Stattdessen gibt der Client bei Anfragen stets an, welches Format von ihm angegebene Parameter haben und in welchem Format er Antworten vom Webservice akzeptiert. Mögliche Repräsentationen sind hierbei HTML, JSON oder XML.

Beispiel: Stellt ein Webservice Metadaten für einen Buchladen bereit, könnten die Antworten auf die Anfrage „GET <http://beispiel.org/bookstore/book/12345>“ in den Formaten JSON oder XML wie folgt aussehen:

```

1 book: {
2   title: "Meine Biographie",
3   author: "Mustermann, Max",
4   isbn: "978-3-567-89012-3",
5   price: "9.99"
6 }
```

Quelltext 5.1: Repräsentation in JSON (118 Byte)

```

1 <book>
2   <title>Meine Biographie</title>
3   <author>Mustermann, Max</author>
4   <isbn>978-3-567-89012-3</isbn>
5   <price>9.99</price>
6 </book>
```

Quelltext 5.2: Repräsentation in XML (136 Byte)

Beide Repräsentationen weisen den selben Informationsgehalt auf, haben jedoch eine unterschiedliche Bytegröße (118 Byte gegenüber 136 Byte). Durch die JSON-Syntax wird bereits bei einem kleinen Beispiel ersichtlich, dass weniger Datenvolumen für die Repräsentation nötig ist. Darum wird für die in dieser Masterarbeit entwickelten RESTful Webservices das JSON-Format genutzt.

Der Aufbau von JSON besteht aus Schlüssel-/Wertpaaren, wobei ein Schlüssel immer ein `String` (Zeichenkette) ist und ein Wert immer ein `JSONObject`, ein

5 Entwurf

JSONArray, ein String, eine Zahl (Integer, Long, Double), ein Boolean (Wahrheitswert true/false) oder der spezielle Wert null ist. Die Typen JSONObject und JSONArray können dabei beliebig ineinander verschachtelt werden.¹

Verknüpfungen von Ressourcen Ressourcen können auf andere Ressourcen verweisen oder deren Inhalte in verschachtelter Form repräsentieren. Verweise werden per Konvention mit href bezeichnet.

Beispiel: Beim Buchbeispiel könnte z. B. der Autor detaillierter beschrieben werden oder als eigene Ressource durch den Webservice bereitgestellt werden. Fragt man nun ein Buch an wären die folgenden Repräsentationen möglich:

```
1 book: {
2   title: "Meine Biographie",
3   author: { href: "http://beispiel.org/bookstore/author/42" },
4   isbn: "978-3-567-89012-3",
5   price: "9.99"
6 }
```

Quelltext 5.3: Verknüpfung in JSON

oder expandiert mit den Informationen der verknüpften Ressource:

```
1 book: {
2   title: "Meine Biographie",
3   author: {
4     id: "42",
5     firstname: "Max",
6     lastname: "Mustermann" },
7   isbn: "978-3-567-89012-3",
8   price: "9.99"
9 }
```

Quelltext 5.4: Expandierte Verknüpfung in JSON

SOAP-Webservices

Bei SOAP handelt es sich um ein Netzwerkprotokoll, dass durch das W3C standardisiert ist². Als Datenformat setzt SOAP auf XML. Der Kommunikationskanal ist bei SOAP nicht beschränkt und kann z. B. mit den Protokollen HTTP oder FTP abgedeckt werden. Anders als mit REST, welches das CRUD-Prinzip unterstützen soll, ist ein SOAP-Webservice eher eine Möglichkeit komplexe, entfernte Methodenaufrufe über SOAP-Nachrichten durchzuführen und zu übertragen.

¹JSON Einführung <http://json.org/> (Stand: 15.11.13)

²W3C Webseite <http://www.w3.org/TR/soap12-part1/> (Stand: 09.11.2013)

WSDL Die Web Services Description Language (WSDL) beschreibt die Schnittstellen für einen SOAP-Webservice. Im Detail sind dies die Nachrichtenformate, welche Methoden der Webservice bereitstellt und mit welchen Parametern diese aufgerufen werden sollen.

Vergleich

Beim Vergleich von REST mit SOAP fallen sowohl Vorteile als auch Nachteile der jeweiligen Lösung auf. Der wesentliche Nachteil von SOAP ist der höhere Aufwand für die Erzeugung von validen XML-Nachrichten. Zum einen kostet dies mehr Rechenzeit und zum anderen weisen die XML-Nachrichten von SOAP einen deutlichen Overhead an Daten auf. Wird eine Kommunikation zwischen verschiedenen komplexen Systemen gewünscht, ist die Wahl von SOAP ein Vorteil, da eine genaue Schnittstellenbeschreibung mit der WSDL möglich ist.

Das in dieser Masterarbeit entworfene System, wird jedoch im wesentlichen Webservices nach dem CRUD-Prinzip nutzen und somit auf Webservices mit REST-Schnittstellen setzen.

5.1.3 Entwurf der Komponenten

Im Folgenden wird der Entwurf für die Komponenten (Abb. 5.4) von HYDrOUS beschrieben. Ziel ist es einen Überblick über das Gesamtsystem zu geben und die Schnittstellen zwischen den einzelnen Komponenten zu erläutern.

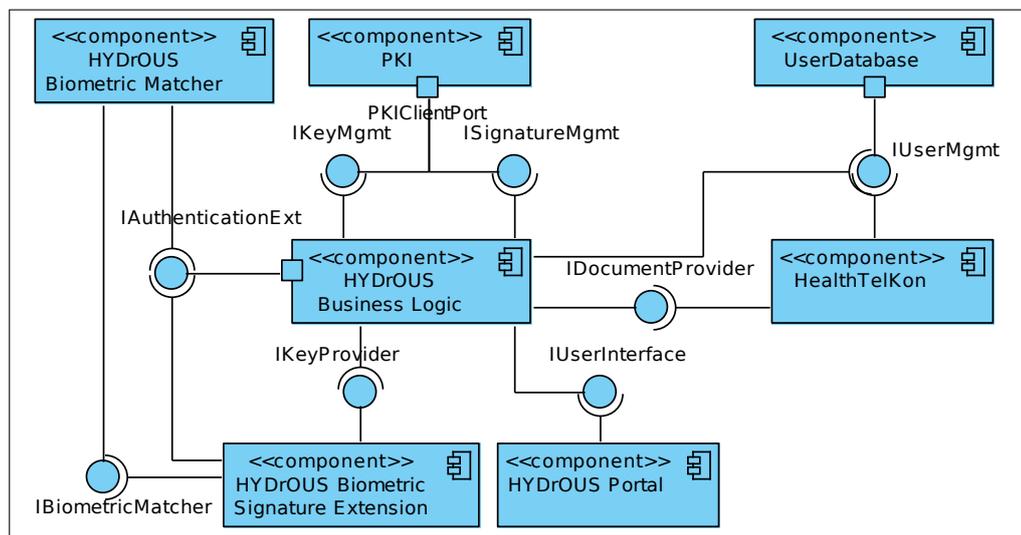


Abbildung 5.4: Komponentendiagramm von HYDrOUS

HYDrOUS Business Logic

Die HYDrOUS Business Logic ist die zentrale Komponente des Systems. Sie implementiert das Mehrparteien-Signaturprotokoll und somit die Hauptfunktionalität. Die Business Logic stellt für die übrigen Komponenten eine Schnittstelle bereit oder nutzt deren Schnittstellen.

Die weiteren Funktionalitäten sind die Benutzerverwaltung, die Schlüsselverwaltung und die Verwaltung der Dokumente inklusive der Mehrparteien-Signaturen. Die Business Logic ist für die Persistenz der Daten verantwortlich und speichert sie in den dafür vorgesehen Komponenten (Datenbank).

Technische Realisierung Die Business Logic wird als Webanwendung für *Java Enterprise Application Server* entwickelt. Als zentrale Komponente für das gesamte System übernimmt sie Funktionen, die andere Komponenten benötigen und stellt diese als Schnittstelle bereit:

- Authentifizierung
- Rechteverwaltung
- Dokumentenverwaltung
- Schlüsselverwaltung
- Benutzerverwaltung
- Benutzeroberfläche

Die Schnittstellen werden abhängig von der zu verbindenden Komponente realisiert. Für Schnittstellen, die nicht direkt auf dem Application Server genutzt werden können, wird eine Implementierung als *RESTful Webservice* vorgesehen. Eine solche Anbindung kann z. B. von einer JAVA Desktop Applikation angesprochen werden, da diese lokal auf den Computern der Benutzer ausgeführt wird.

HYDrOUS Portal

Das HYDrOUS Portal stellt die Benutzerschnittstellen für die Funktionalität der Business Logic bereit. Das Portal wird als Webanwendung umgesetzt und kann so mit einem Webbrowser bedient werden.

Der Benutzer kann sich über das HYDrOUS Portal am System anmelden oder ein neues Benutzerkonto öffnen. Für die Authentifizierung wird eine Kombination aus der E-Mail Adresse des Benutzers und Benutzerkennwort gewählt. Hat sich ein Benutzer authentifiziert, wird ihm im HYDrOUS Portal eine *Übersichtsseite* präsentiert. Auf der Übersichtsseite werden aktuelle Informationen, die das System und das

Benutzerkonto betreffen, angezeigt. Für das System relevante Informationen sind Hinweise vom Administrator, geplante Wartungsarbeiten oder kürzliche Änderungen am System (Änderungshistorie). Informationen, die das Benutzerkonto betreffen, sind Hinweise über Einladungen zu neuen Mehrparteien-Signaturen für Dokumente oder der Fortschritt bei Erstellung einer Mehrparteien-Signatur für bestehende Dokumente.

Über den Navigationsbereich am linken Bildschirmrand kann der Benutzer auf weitere Funktionen Zugriff erhalten:

Dokumentenübersicht In der *Dokumentenübersicht* werden dem Benutzer alle Dokumente angezeigt, zu deren Mehrparteien-Signatur er eingeladen wurde oder Dokumente, die der Benutzer selbst als Konferenzmanager in das System eingestellt hat. Die Dokumente werden nach drei Kriterien sortiert:

1. Kriterium: Dokumente, bei denen noch eine Mehrparteien-Signatur aussteht und der Benutzer noch nicht teilgenommen hat.
2. Kriterium: Dokumente, bei denen noch eine Mehrparteien-Signatur aussteht und der Benutzer bereits teilgenommen hat.
3. Kriterium: Dokumente, bei denen eine Mehrparteien-Signatur vollständig erstellt wurde.

Innerhalb der Kriterien werden die Dokumente nach dem Einstellungsdatum sortiert. Die Darstellung der Dokumente wird durch Miniaturen (Abb. 5.5) realisiert, die über den Fortschritt, die Anzahl der Teilnehmer und das Einstellungsdatum informieren. Klickt der Benutzer auf eine solche Miniatur, wird die *Dokumentendetailansicht* geöffnet.

Dokumentendetailansicht In der *Dokumentendetailansicht* werden die Mehrparteien-Signaturteilnehmer für das aktuelle Dokument aufgelistet und visuell dargestellt, ob ein Teilnehmer bereits an der Mehrparteien-Signatur teilgenommen hat (Abb. 5.6). Unter der Liste von Teilnehmern wird das eigentliche *Dokument im PDF Format* dargestellt. Da Teilnehmer einer Mehrparteien-Signatur das zu unterzeichnende Dokument gelesen haben müssen, findet sich unter dem Dokument eine Schaltfläche, um sein Einverständnis zu erklären. Erfolgt diese *Einverständnis*, kann der angemeldete Benutzer über eine eingeblendete Schaltfläche am Mehrparteien-Signatur-Protokoll teilnehmen.

Mehrparteien-Signatur prüfen Wählt der Benutzer die Schaltfläche *Mehrparteien-Signatur prüfen*, wird ein Formular dargestellt, in dem er das zu validierende Dokument als Datei auswählen kann. Das Dokument wird dann temporär auf den Server

5 Entwurf

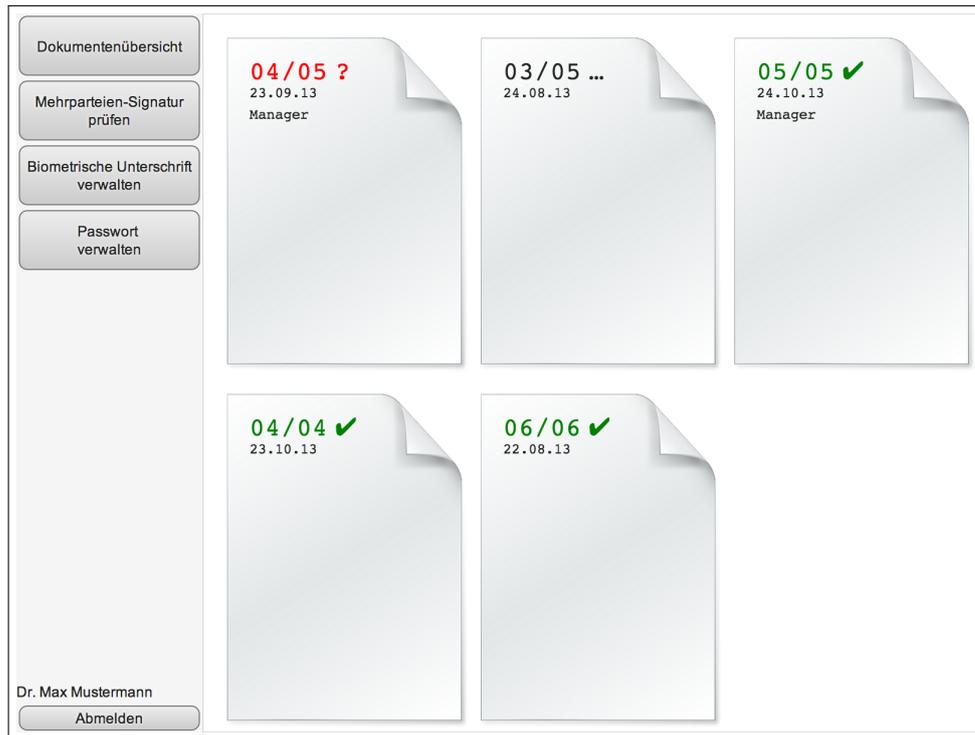


Abbildung 5.5: Screenshot der Dokumentenübersicht eines Benutzers

übertragen, um die Mehrparteien-Signatur zu validieren. Nachdem die Prüfung abgeschlossen ist, wird dem Benutzer visuell dargestellt, ob das bereitgestellte Dokument eine valide Signatur hat oder nicht.

Biometrische Unterschrift verwalten Betätigt der Benutzer die Schaltfläche *Biometrische Unterschrift verwalten*, wird die *HYDrOUS Biometric Signature Extension* gestartet. Sie wird außerhalb des Webbrowsers ausgeführt, da Gerätetreiber für das Unterschriftenpad benötigt werden. Weitere Informationen finden sich in Unterabschnitt 5.1.3.

Benutzerkonto verwalten Wählt der Benutzer die Schaltfläche *Benutzerkonto verwalten*, wird dem Benutzer ein Formular mit seinen Benutzerdaten angezeigt. Der Benutzer kann dann Formularfelder bearbeiten, die sich geändert haben (z. B. Kontaktdaten, Anschrift). Außerdem ist es dem Benutzer möglich sein Benutzerkennwort zu ändern. Klickt der Benutzer die entsprechende Schaltfläche, wird die *HYDrOUS Biometric Signature Extension* gestartet.

Technische Realisierung Das HYDrOUS Portal wird als Webanwendung mit dem Framework VAADIN 7 entwickelt. Das Portal soll im wesentlichen als grafische Oberfläche für das Gesamtsystem dienen und in Webbrowsern nutzbar sein. Über definierte

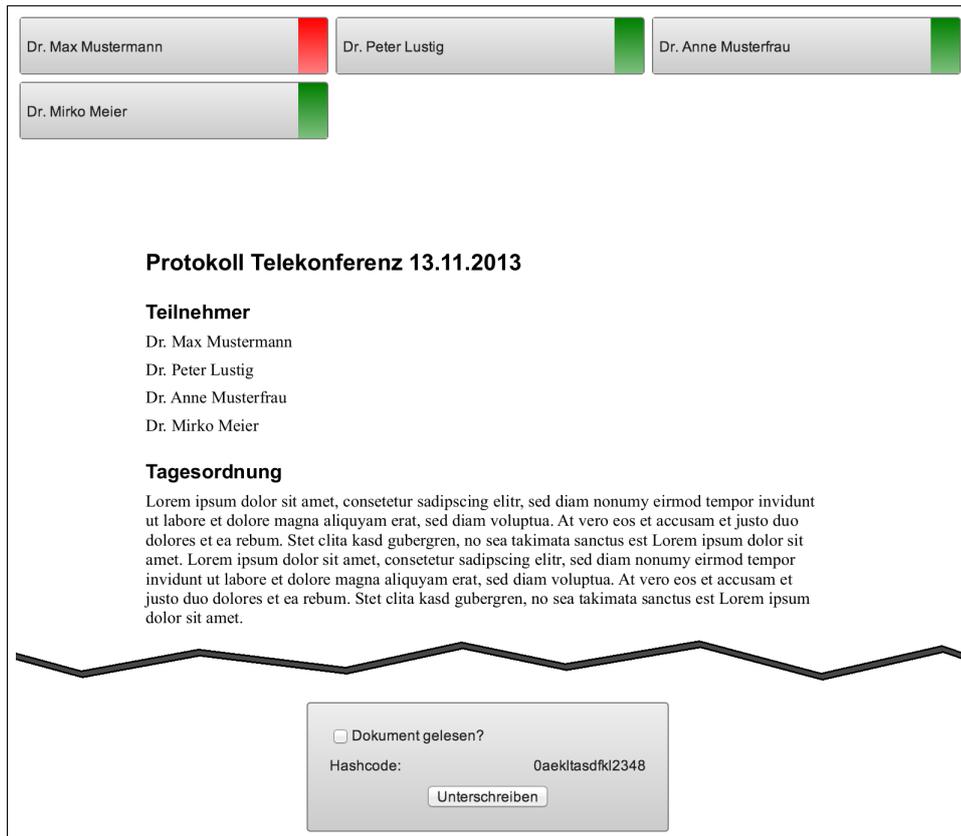


Abbildung 5.6: Screenshot der Dokumentendetailansicht

Schnittstellen kommuniziert das Portal mit der HYDrOUS Business Logic, die sämtliche Logik von der grafischen Oberfläche entkoppelt.

HYDrOUS Biometric Signature Extension

Nachdem ein Benutzer die HYDrOUS Biometric Signature Extension gestartet hat, kann er verschiedene Aktionen durchführen. Der Benutzer kann eine biometrische Unterschrift aufzeichnen, die dann inklusive Druckinformation visualisiert wird. Der Benutzer kann ein biometrisches Template erzeugen, indem er die Schritte des folgenden Assistenten durchführt:

1. aktuelle biometrische Signatur eingeben (wenn vorhanden)
2. neue biometrische Signatur mehrfach eingeben
3. Benutzername und Kennwort eingeben
4. Aktualisierung des biometrischen Templates starten

5 Entwurf

Der Benutzer kann eine Liste der Dokumente sehen, die er signieren soll und die hinterlegten Dokumente einsehen. Startet er einen Signaturvorgang, wird er durch einen Assistenten geleitet, der ihm die folgenden Schritte vorgibt:

1. Dokument lesen
2. biometrische Signatur eingeben
3. Benutzername und Kennwort eingeben
4. Ergebnis des biometrischen Abgleiches anzeigen
5. Mehrparteien-Signaturprotokoll starten

Der Benutzer kann auch das Kennwort für seine biometrische Unterschrift (und sein Benutzerkonto) ändern, hierzu wird ihm auch ein Assistent bereitgestellt:

1. aktuelle biometrische Signatur eingeben (wenn vorhanden)
2. neues Benutzerkennwort zweimal eingeben
3. Benutzername und bestehendes Kennwort eingeben
4. Aktualisierung des Kennwortes starten

Technische Umsetzung Die HYDrOUS Biometric Signature Extension wird als JAVA Desktop Applikation realisiert, da die Anbindung an Unterschriftenpads einen Gerätetreiber benötigt. Letzteres ist ohne die Installation von Drittsoftware nicht über Webbrowser realisierbar und würde die Wahl des Webbrowsers zusätzlich einschränken.

Die HYDrOUS Biometric Signature Extension wird außerdem immer im Kontext des privaten Schlüssels des aktuellen Benutzers eingesetzt, welcher aus Sicherheitsgründen nicht auf einen entfernten Server übertragen werden soll. Stattdessen wird der private Schlüssel in einem verschlüsselten Schlüsselbund auf einem lokalen Datenträger (z. B. Festplatte, USB Stick) gespeichert, sodass Zugriff auf das lokale Dateisystem bestehen muss.

HYDrOUS Biometric Matcher

Die Komponente HYDrOUS Biometric Matcher implementiert den Abgleich von biometrischen Signaturen. Dieser Abgleich kann nicht auf dem Client (also der Biometric Signature Extension) durchgeführt werden, da eine Manipulation an der Client-Software oder den biometrischen Templates unbemerkt bleiben könnte. (z. B. Brute-Force Matching gegen die hinterlegten biometrischen Templates)

Technische Umsetzung Die Komponente wird als RESTful Webservice implementiert, da die Anfragen an den Service immer auf Ressourcen (Biometrische Templates) abzielen. Die funktionalen Details werden in Unterabschnitt 5.2.2 erläutert.

PKI

Die Komponente Public Key Infrastructure (PKI), dient der Datenhaltung der öffentlichen Schlüssel für die Benutzer des Systems. Die öffentlichen Schlüssel werden inklusive eines Zertifikates gespeichert und sind durch eine eindeutige ID gekennzeichnet.

Technische Umsetzung Die PKI-Komponente kann im Wesentlichen über zwei Schnittstellen genutzt werden. Die programmatische Schnittstelle wird von der HYDrOUS Business Logic eingesetzt, die in REST implementierte Webservice-Schnittstelle wird von der HYDrOUS Biometric Signature Extension genutzt.

UserDatabase

Die Komponente UserDatabase, wird für die Speicherung von Benutzerkonten genutzt. Die Komponenten des HYDrOUS Systems und HEALTHTELKON teilen sich diese Datenbank.

Technische Umsetzung Die Umsetzung wird mit einer relationalen Datenbank durchgeführt.

5.2 Protokolle

Damit Mehrparteien-Signaturen umgesetzt werden können, wird ein gemeinsames Protokoll für die Parteien benötigt. Im folgenden wird ein solches Mehrparteien-Signaturprotokoll vorgestellt. Zur Steigerung der Sicherheit soll es für die Benutzer möglich sein, sich gegenüber einem Server mit ihren biometrischen Unterschriften zu authentifizieren. Hierzu wird ein zweites Protokoll für die biometrische Schlüsselfreigabe vorgestellt.

5.2.1 Mehrparteien-Signaturprotokoll

Das zu implementierende Mehrparteien-Signaturprotokoll basiert technisch auf *Shamir's Secret Sharing* (Unterabschnitt 3.2.1) in Kombination mit asymmetrischer Verschlüsselung (Unterabschnitt 2.1.2). Entsprechend den Empfehlungen für elektronische Signaturen durch das BSI (Unterabschnitt 2.2.3) setzt das Protokoll auf den RSA Algorithmus (Paragraph 2.2.3).

Notation

- Sei $P = (p_1, p_2, \dots, p_n)$ die sortierte Liste der teilnehmenden Parteien, die ein Dokument D gemeinsam unterzeichnen wollen. Weiterhin sei T die TTP, der alle Parteien vertrauen.
- Jede Partei p_i verfügt über einen öffentlichen Schlüssel PK_{p_i} und einen geheimen Schlüssel SK_{p_i} .
- Die öffentlichen Schlüssel sind für alle Parteien verfügbar und besitzen ein Zertifikat $C_T(PK_{p_i})$, das von der TTP ausgestellt wurde.
- Alle Parteien können Funktionen für Entschlüsselung $Dec(msg, key)$, Verschlüsselung $Enc(msg, key)$ und zur Berechnung von Hashwerten $H(msg)$ nutzen.
- Der Austausch einer Nachricht msg von Sender S zu Empfänger R wird durch $S \rightarrow R : msg$ repräsentiert.
- Als geteiltes Geheimnis aus *Shamir's Secret Sharing* wird im Protokoll der Hashwert des zu unterzeichnenden Dokumentes $H(D)$ gewählt.
- Die Anteile des Geheimnisses werden im Folgenden d_i genannt, wobei der Index i für die spätere Zuordnung zu Partei p_i genutzt wird. Letzteres ist nötig, da die Reihenfolge bei der Wiederherstellung des geteilten Geheimnisses eingehalten werden muss. Für eine Rekonstruktion des Geheimnisses wird außerdem der Geheimnisanteil jeder teilnehmenden Partei vorausgesetzt und nicht mit einem niedrigeren Schwellwert gearbeitet.
- Die Funktionen des Algorithmus werden im folgenden durch $SSS.split(H(D), P) = (d_1, d_2, \dots, d_n)$, sowie $SSS.combine(d_1, d_2, \dots, d_n) = H(D)$ dargestellt und die Parameter für Schwellwerte und Indizes implizit aus den Eingaben abgeleitet.

Annahmen

Das Mehrparteien-Signaturprotokoll wird unter den folgenden Annahmen beschrieben. Sie sind die Voraussetzung für den reibungslosen Ablauf und für die Interessenwahrung der teilnehmenden Parteien.

- Die Kommunikationskanäle zwischen den Parteien in P und der TTP T sind über technische Mittel geschützt. Sie übertragen Nachrichten an den designierten Empfänger zuverlässig und unverfälscht.
- Alle Teilnehmer am Protokoll nutzen identische Algorithmen für Entschlüsselung $Dec(msg, key)$, Verschlüsselung $Enc(msg, key)$ und Hashfunktionen $H(msg)$.

- Die Parteien haben das Dokument D zuvor gelesen und haben ihr Einverständnis mit dem Inhalt erklärt.
- Jede Partei p_i hat vor Beginn des Protokolls ein Schlüsselpaar und ein gültiges Zertifikat $C_T(PK_{p_i})$ für ihren öffentlichen Schlüssel PK_{p_i} .
- Die TTP verhält sich in jedem Fall vertrauenswürdig.

Ablauf

Im folgenden werden die Schritte von der Initialisierung bis zum Abschluss einer Mehrparteien-Signatur beschrieben.

1. Initialisierung der geteilten Geheimnisse

Bevor das Protokoll beginnen kann, müssen ein Dokument D und Informationen $P = (p_1, p_2, \dots, p_n)$ über die teilnehmenden Parteien in das System übertragen werden. Die TTP erstellt aus diesen Daten ein geteiltes Geheimnis $H(D)$. Danach werden die Geheimnisanteile d_1, d_2, \dots, d_n mit *Shamir's Secret Sharing* generiert.

$$SSS.split(H(D), P) = (d_1, d_2, \dots, d_n)$$

Die so erzeugten Geheimnisanteile werden dann von der TTP mit dem öffentlichen Schlüssel der designierten Partei verschlüsselt, danach werden die unverschlüsselten Geheimnisanteile verworfen.

$$ed_i := Enc(d_i, PK_{p_i}), \quad \forall i \in 1 \dots n$$

Die Geheimnisanteile müssen nun gespeichert werden, bis sie tatsächlich durch die jeweilige Partei angefordert werden. Bevor sie jedoch persistiert werden, signiert die TTP die verschlüsselten Anteile und fügt ihnen Informationen über das zu unterzeichnende Dokument und die Partei p_i hinzu. Durch dieses Vorgehen wird die Integrität der einzelnen Geheimnisanteile gewahrt.

$$S_{p_i}(D) := (ed_i, p_i, Enc((H(ed_i, p_i), H(D)), SK_T)), \quad \forall i \in 1 \dots n$$

2. Verteilen der geteilten Geheimnisse

Das Protokoll soll konzeptionell asynchron ablaufen können, damit die Koordination der einzelnen Parteien keinen zusätzlichen Aufwand ausmacht. Möchte eine Partei

5 Entwurf

sich am Protokoll beteiligen, fordert sie bei der TTP aktiv ihren Geheimnisanteil an. Die TTP leitet nun den zuvor persistierten Geheimnisanteil $S_{p_i}(D)$ an die Partei.

$$\begin{aligned} req_{p_i}(D.id) &:= (D.id, p_i, Enc(Hash(D.id, p_i), SK_{p_i})) \\ p_i \rightarrow T &: req_{p_i}(D.id) \\ T \rightarrow p_i &: S_{p_i}(D) \end{aligned}$$

3. Prüfen, signieren und zurücksenden der Geheimnisanteile durch die Parteien

Erhält p_i die Nachricht $S_{p_i}(D)$ von T muss sie diese zunächst mit dem öffentlichen Schlüssel PK_T entschlüsseln, um die Integrität zu überprüfen.

$$Dec(Enc((H(ed_i, p_i)', H(D)'), SK_T), PK_T) \equiv (H(ed_i, p_i)', H(D)')$$

Um sicherzustellen, dass die Partei p_i einen Geheimnisanteil vom korrekten Dokument D unterzeichnen soll, wird ein frisch berechneter Hashwert mit dem empfangenen Hashwert verglichen. Die Partei p_i überprüft auch den verschlüsselten Geheimnisanteil ed_i . Stimmen der übermittelte Hashwert $H(D)'$ mit dem frisch berechneten Hashwert $H(D)$ und der übermittelte Hashwert $H(ed_i, p_i)'$ mit dem frisch berechneten Hashwert $H(ed_i, p_i)$ überein, fährt p_i fort.

Die empfangene Komponente ed_i kann nur p_i entschlüsseln, da nur sie im Besitz des privaten Schlüssels SK_{p_i} ist.

$$Dec(ed_i, SK_{p_i}) \equiv d_i$$

Hat p_i den für sie generierten Geheimnisanteil entschlüsselt, kann sie dazu übergehen ihn zu signieren und für den Versand an T vorzubereiten.

$$\begin{aligned} sig_i &:= (d_i, Enc(H(d_i), SK_{p_i})) \\ Sig_{p_i}(D) &:= Enc((sig_i, p_i, H(D)), PK_T) \end{aligned}$$

Danach versendet p_i die mit dem öffentlichen Schlüssel PK_T verschlüsselte Nachricht $Sig_{p_i}(D)$ an T .

$$p_i \rightarrow T : Sig_{p_i}(D)$$

4. Kombinieren der einzelnen Geheimnisanteile durch die TTP

Haben alle Parteien ihre Geheimnisanteile signiert und zurück zu T gesendet, werden die einzelnen Signaturen überprüft.

$$\forall i \in 1 \dots n :$$

$$Dec(Sig_{p_i}(D), SK_T) = (sig_i, p_i, H(D))$$

$$sig_i = (d_i, Enc(H(d_i)', SK_{p_i}))$$

$$H(d_i)' = Dec(Enc(H(d_i)', SK_{p_i}), PK_{p_i})$$

Stimmen die entschlüsselten Hashwerte der Parteien $H(d_i)'$ mit den selbst berechneten Hashwerten $H(d_i)$ überein, fährt T fort.

Mit den empfangenen und signierten Geheimnisanteilen rekonstruiert T mittels *Shamir's Secret Sharing* das Gesamtgeheimnis $H(D)'$ und prüft ob es mit dem Hashwert des Dokumentes $H(D)$ übereinstimmt.

$$SSS.combine(d_1, \dots, d_n) = H(D)'$$

5. Gesamtsignatur erstellen

Wurde zuvor eine valide Signatur berechnet, generiert T die Gesamtsignatur $MPCSP(D)$ für das Dokument. Hierzu werden alle signierten Teilgeheimnisse konkateniert und von T signiert.

$$MPCSP(D) := ((sig_1, p_1), \dots, (sig_n, p_n), Enc(H((sig_1, p_1), \dots, (sig_n, p_n)), SK_T))$$

5.2.2 Biometrische Schlüsselfreigabe

Die biometrische Schlüsselfreigabe kombiniert die Vorteile von biometrischer Authentifizierung mit dem Einsatz klassischer Kryptografie. Statt die Signaturen abhängig von den biometrischen Unterschriften der Benutzer zu machen, werden die biometrischen Unterschriften dazu genutzt, die privaten Schlüssel der Benutzer zu sichern und wieder freizugeben. Um die biometrischen Templates vor Angreifer zu schützen, werden sie zusätzlich durch ein Benutzerkennwort verschlüsselt. Somit sind die privaten asymmetrischen Schlüssel durch eine *Zwei-Faktor-Authentifizierung* gesichert.

Dieses Vorgehen bietet den Vorteil, dass bestehende Algorithmen eingesetzt werden können, die nicht auf Biometrie basieren und das System für andere biometrische Merkmale erweitert werden kann. Auch bietet sich so die Möglichkeit völlig von biometrischen Systemen Abstand zu nehmen und klassische Verfahren (z. B. SmartCards oder Tokens) einzusetzen.

Notation

- Sei u_i der Benutzer, welcher seinen privaten Schlüssel freigeben will. Der biometrische Abgleich von eingegebener biometrische Unterschrift und dem hinterlegtem biometrischen Template wird durch eine TTP M , im folgenden *Matcher* genannt, durchgeführt.
- Sei BT_{u_i} das biometrische Template für den Benutzer u_i und BS_{u_i} eine biometrische Unterschrift von u_i .
- Der Benutzer u_i hat ein Schlüsselpaar mit privatem Schlüssel SK_{u_i} und öffentlichem Schlüssel PK_{u_i} . Auf SK_{u_i} hat u_i exklusiven, lokalen Zugriff. Der öffentliche Schlüssel PK_{u_i} ist für M verfügbar.
- Die TTP M hat ebenfalls ein Schlüsselpaar mit privatem Schlüssel SK_M und öffentlichem Schlüssel PK_M . Der öffentlichen Schlüssel liegt dem Benutzer u_i vor.
- Die Benutzer und die TTP können Funktionen für Entschlüsselung $Dec(msg, key)$, Verschlüsselung $Enc(msg, key)$ und zur Berechnung von Hashwerten $H(msg)$ nutzen.
- Der Austausch einer Nachricht msg von Sender S zu Empfänger R wird durch $S \rightarrow R : msg$ repräsentiert.
- Die beiden Funktionen $auth(user, pw)$ und $match(templ, sig)$ werden M über Schnittstellen bereitgestellt und authentifizieren einen Benutzer bzw. gleichen ein bestehendes Template mit einer Signatur ab.

Annahmen

Die Biometrische Schlüsselfreigabe wird unter den folgenden Annahmen beschrieben. Sie sind die Voraussetzung für den reibungslosen Ablauf und für die Interessenwahrung der teilnehmenden Parteien.

- Die Kommunikationskanäle zwischen den Benutzern u_i und der TTP M sind über technische Mittel geschützt. Sie übertragen Nachrichten an den designierten Empfänger zuverlässig und unverfälscht.
- Alle Teilnehmer am Protokoll nutzen identische Algorithmen für Entschlüsselung $Dec(msg, key)$, Verschlüsselung $Enc(msg, key)$ und Hashfunktionen $H(msg)$.
- Jeder Benutzer hat bereits vor Beginn der biometrischen Schlüsselfreigabe ein biometrisches Template für die TTP M bereitgestellt.

- Jeder Benutzer u_i hat vor Beginn des Protokolls ein Schlüsselpaar und ein gültiges Zertifikat $C_T(PK_{u_i})$ für seinen öffentlichen Schlüssel PK_{u_i} .
- Die TTP M verhält sich in jedem Fall vertrauenswürdig.

Ablauf

Im folgenden werden die Schritte von der Anforderung des Benutzers bis zum Erhalt des Geheimnisses für dessen privaten Schlüssel beschrieben.

1. Initiale Anfrage an den Matcher

Der Benutzer u_i zeichnet eine biometrische Unterschrift BS_{u_i} auf. Für die Kommunikation mit dem Matcher M muss ein symmetrischer Sitzungsschlüssel $k_{u_i,M}$ erzeugt werden. (Dies ist erforderlich, da der private Schlüssel SK_{u_i} nur verschlüsselt vorliegt.) Mit dem symmetrischen Schlüssel verschlüsselt der Benutzer nun die aufgezeichnete biometrische Unterschrift.

$$BSec_{u_i} := Enc(BS_{u_i}, k_{u_i,M})$$

Damit der Matcher M das hinterlegte biometrische Template BT_{u_i} mit der eben aufgezeichneten biometrischen Unterschrift abgleichen kann, muss er das Template mit dem Benutzerkennwort pw_{u_i} von u_i entschlüsseln. Hierzu gibt der Benutzer sein Benutzerkennwort ein und der Hashwert wird gebildet. Der weitere Nutzen für diese Authentifizierung ist die Feststellung der Identität von u_i .

$$hpw := H(pw_{u_i})$$

Nun bereitet u_i die Nachricht an M vor. Diese besteht aus dem symmetrischen Schlüssel, Informationen über u_i , dem Benutzerkennwort hpw und einer $nonce_B$. Letztere wird für die Authentifizierung von M herangezogen, indem er die $nonce_B$ in die Antwort integriert. Diese Nachricht wird dann mit dem öffentlichen Schlüssel PK_M verschlüsselt und mit der biometrischen Unterschrift konkateniert. Um einen Replay-Angriff zu vermeiden, muss eine zweite $nonce_M$ beim Matcher angefragt werden, da sonst die Nachricht wiedereingespielt werden könnte und sich ein Angreifer als u_i authentifizieren könnte.

$$u_i \rightarrow M : start$$

$$M \rightarrow u_i : (nonce_M, Enc(H(nonce_M), SK_M))$$

Erhält u_i die Nachricht von M , prüft er ob sie integer ist und fährt fort.

$$req := (Enc((k_{u_i,M}, u_i, hpw, nonce_M, nonce_B), PK_M), BSec_{u_i})$$

$$u_i \rightarrow M : req$$

5 Entwurf

2. Empfang und Entschlüsselung der Anfrage

Der Matcher M empfängt die verschlüsselte Nachricht von u_i und entschlüsselt diese mit dem privaten Schlüssel SK_M .

$$Dec(req, SK_M) \equiv Dec(Enc((k_{u_i, M}, u_i, hpw, nonce'_M, nonce_B), PK_M), SK_M)$$

Mit den entschlüsselten Daten prüft der Matcher ob die empfangene $nonce'_M$ mit der ausgegebenen $nonce_M$ übereinstimmt, dann authentifiziert der Matcher den Benutzer gegen die Authentifizierungsstelle.

$$auth(u_i, H(pw_{u_i}))$$

War die Authentifizierung erfolgreich, ist die Identität von u_i validiert und das Protokoll kann fortgeführt werden.

3. Biometrischer Abgleich

Mit dem symmetrischen Schlüssel $k_{u_i, M}$ kann M nun die biometrische Unterschrift BS_{u_i} entschlüsseln und gegen das biometrische Template BT_{u_i} abgleichen. Das biometrische Template wird mit dem Benutzerkennwort entschlüsselt.

$$\begin{aligned} BS'_{u_i} &:= Dec(BSec_{u_i}, k_{u_i, M}) \\ BT'_{u_i} &:= Dec(BTec_{u_i}, H(pw_{u_i})) \\ match(BT'_{u_i}, BS'_{u_i}) \end{aligned}$$

4. Antwort an den Benutzer

Ist der biometrische Abgleich positiv, bereitet M die Antwort an u_i vor. Hierzu bildet er den Hashwert des biometrischen Templates $H(BT'_{u_i})$, welcher als Kennwort für den privaten Schlüssel SK_{u_i} dient. Um sich gegenüber dem Benutzer zu authentifizieren, fügt M die $nonce_B$ zur Antwort hinzu, bevor er diese mit dem symmetrischen Schlüssel verschlüsselt und versendet.

$$\begin{aligned} res &:= Enc((H(BT'_{u_i}), nonce_B), k_{b_i, M}) \\ M \rightarrow u_i &: res \end{aligned}$$

5. Empfang und Validierung durch den Benutzer

Nach Erhalt der Antwort entschlüsselt der Benutzer diese mit dem symmetrischen Schlüssel und überprüft das zurückerhaltene $nonce'_B$.

$$Dec(res, k_{b_i, M}) \equiv Dec(Enc((H(BT'_{u_i}), nonce'_B), k_{b_i, M}), k_{b_i, M})$$

Stimmt $nonce'_B$ mit $nonce_B$ überein, ist bewiesen, dass der Matcher über das vertrauenswürdige Schlüsselpaar von M verfügt. Nun kann der Benutzer das empfangene Kennwort nutzen, um seinen privaten Schlüssel zu entschlüsseln.

$$SK_{u_i} = Dec(Enc(SK_{u_i}, H(BT'_{u_i})), H(BT'_{u_i}))$$

Anmerkung: Sämtliche übertragenen Geheimnisse sind auf der Benutzerseite sowie auf der Matcherseite nach einer vollständigen Ausführung des Protokolls zu löschen. Dies ist im Interesse des Benutzers, da sein privater Schlüssel nur durch Wissen seines Benutzerkennwortes und eine korrekt abgeglichene biometrische Unterschrift freigegeben werden soll.

5.2.3 Biometrischer Abgleich mit Dynamic Time Warping (DTW)

Die in Unterabschnitt 5.2.2 eingesetzte Funktion $match(templ, sig)$ verwendet biometrische Signaturen, die nach dem *On-Line Verfahren* aufgezeichnet wurden und vergleicht sie gegen ein bereits aufgezeichnetes biometrisches Template.

Dieser Abgleich wird technisch mit Dynamic Time Warping (DTW) durchgeführt, welches in Unterabschnitt 3.1.6 beschrieben wurde.

Merkmale für den biometrischen Abgleich

Als Merkmale für biometrische Unterschriften wird eine Kombination aus verschiedenen Merkmalen (Tab. 5.2) gewählt. Diese Merkmale wurden für die prototypische Entwicklung in kleinen Testreihen mit wenigen Personen ermittelt.

In den Testreihen hat sich herausgestellt, dass eine Kombination aus globalen und lokalen Merkmalen die besten Ergebnisse liefert. Als globale Merkmale werden das *Seitenverhältnis* der Unterschrift, die *Zeitspanne / Dauer* zum Unterschreiben und die *Anzahl der Striche* (d.h. wie oft wurde der Stift abgesetzt) ausgewählt. Als lokale Merkmale wurden die *Druckveränderung*, die *Beschleunigung* und die *Geschwindigkeit* herangezogen.

globale Merkmale	lokale Merkmale
Seitenverhältnis	Druckveränderung
Zeitspanne / Dauer	Beschleunigung (X & Y Richtung)
Anzahl der Striche	Geschwindigkeit (X & Y Richtung)

Tabelle 5.2: Ausgewählte Merkmale für den Abgleich von biometrischen Unterschriften

Berechnung der Merkmale Die biometrischen Rohdaten werden vom Unterschriftenpad (bzw. dem SDK) als XML Dokument wie in Quelltext 5.5 ausgegeben.

5 Entwurf

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <SignatureData>
3   <Points>
4     [...]
5     <PenPoint>
6       <RawX>2180</RawX>
7       <RawY>4708</RawY>
8       <Pressure>136</Pressure>
9       <X>137</X>
10      <Y>296</Y>
11      <Time>1122</Time>
12    </PenPoint>
13    [...]
14  </Points>
15 </SignatureData>
```

Quelltext 5.5: Rohdaten aus dem Signaturtablet als XML Dokument

Die Rohdaten müssen für die Nutzung als Merkmal aufbereitet werden. Abhängig davon, ob ein Merkmal global oder lokal ist, werden andere Methoden dafür verwendet.

Globales Merkmal „Seitenverhältnis“: Aus den Punkten der Rohdaten werden die maximalen und minimalen x bzw. y Koordinaten bestimmt. Danach wird das Seitenverhältnis über den Quotient der Breite zur Höhe berechnet

$$\text{Seitenverhältnis} := \frac{x_{max} - x_{min}}{y_{max} - y_{min}}$$

Globales Merkmal „Zeitspanne“: Aus den Punkten der Rohdaten wird der maximale Zeitstempel (Time) gesucht.

$$\text{Zeitspanne} := time_{max}$$

Globales Merkmal „Anzahl der Striche“: Aus den Punkten werden solche als Ansatz für einen „neuen Strich“ bewertet, die einen Schwellwert t beim Druck unterschreiten.

$$\text{Anzahl der Striche} := |\{p \in P \mid (p.pressure < t) \wedge (p.time > 0)\}|$$

Lokale Merkmale: Die lokalen Merkmale „Druckveränderung“ und „Geschwindigkeit“ werden jeweils einfach numerisch differenziert. Hierzu wird der Differenzenquotient von zwei aufeinanderfolgenden Werten berechnet. Bei der Druckveränderung ist das der Druck in einem Punkt $p.pressure$, bei der Geschwindigkeit die Koordinate $p.x$ bzw. $p.y$.

$$f'(p_i.x) = \frac{f(p_{i+1}.x) - f(p_i.x)}{f(p_{i+1}.time) - f(p_i.time)}$$

Das Merkmal „Beschleunigung“ ist die zweite Ableitung der Koordinatenveränderungen oder die erste Ableitung der Geschwindigkeit.

Kombination der Merkmale Warum die Kombination von Merkmalen sinnvoll ist zeigt folgendes **Beispiel:** Betrachtet man das lokale Merkmal *Druckveränderung*, kann eine gefälschte („nachgemalte“) Unterschrift zwar erkannt werden, jedoch muss dies nicht in jedem Fall gleich stark auftreten. Mit DTW wird hier zwar eine höhere Distanz berechnet, die aber auch auf natürlichen Schwankungen beruhen kann. Da Unterschriften in der Regel schnell geschrieben werden bietet sich der direkte Vergleich der *Zeitspanne* an, die bei Fälschung und Original unterschiedlich ausfallen sollte. Mögliche Distanzen sind in Tab. 5.3 aufgeführt.

Merkmal	Dist. (Original/Fälschung)	Dist. (Original/Original2)
Druckveränderung	89.99445913732052	60.06457083672285
Zeitspanne	5002.0	41.0
Anzahl d. Striche	5	1
Geschwindigkeit Y	211.731	111.136
Beschleunigung Y	15.374	20.737
Seitenverhältnis	0.38	0.09

Tabelle 5.3: Vergleich von Original/Fälschung und zwei originalen Unterschriften

5.3 Interaktion mit HEALTHTELKON

Im Ablauf einer Konferenz wird die Interaktion mit HYDrOUS nach dem 5. Schritt „Protokollierung: Der Konferenzmanager protokolliert Kommentare und Ergebnisse (Therapieempfehlungen) der Besprechung“ (Unterabschnitt 2.3.1) gestartet.

Das HYDrOUS System soll Protokolle von HEALTHTELKON für Mehrparteien-Signaturen erhalten können, aber nicht auf die ausschließliche Nutzung von HEALTHTELKON angewiesen sein. Der hier entwickelte Prototyp soll nach einer erfolgten Telekonferenz das erzeugte Protokoll inklusive der Metadaten (Konferenzteilnehmer, Konferenzmanager) von HEALTHTELKON automatisiert an HYDrOUS übertragen.

5.3.1 Automatisierte Bereitstellung

Diese automatisierte Bereitstellung wird mittels eines REST-Webservices durchgeführt. Das Protokoll wird hierbei inklusive der Metadaten elektronisch signiert, damit die Integrität auf dem Übertragungsweg erhalten bleibt. Da sich HYDrOUS und HEALTHTELKON eine Benutzerdatenbank teilen, werden als Information über

5 Entwurf

die Konferenzteilnehmer deren eindeutige Bezeichner aus der Benutzerdatenbank genutzt.

5.3.2 Status einer Mehrparteien-Signatur

Hat sich ein Benutzer in HEALTHTELKON angemeldet, kann er die vergangen Telekonferenzen einsehen. Wählt er eine Konferenz aus, werden Statusinformationen zu einem laufenden Mehrparteien-Signaturvorgang über einen REST-Webservice an HEALTHTELKON gesendet. Sobald ein Protokoll eine vollständige Mehrparteien-Signatur erhalten hat, wird eine Verknüpfung zu diesem Protokoll bereitgestellt.

6 Prototypische Implementierung

Im folgenden Kapitel wird auf technischer Ebene die Umsetzung der Anforderungsanalyse und des Entwurfes beschrieben. Dazu werden zunächst die technischen Hilfsmittel erläutert, die zur Implementierung verwendet wurden und dann die Implementierung einiger wichtiger Komponenten betrachtet.

6.1 Verwendete technische Hilfsmittel für die Implementierung

Die Systemkonfiguration und die verwendeten Werkzeuge für die Implementierung des Prototypen werden in diesem Abschnitt beschrieben.

6.1.1 Systemkonfiguration

Die Implementierung wurde auf den in Tab. 6.1 aufgelisteten Systemen entwickelt und getestet. Der Prototyp ist zwar bezüglich der meisten Komponenten plattformunabhängig, jedoch wurde bis zum Abschluss dieser Masterarbeit keine Version des SDK für das Unterschriftenpad für alle Betriebssystemen veröffentlicht.

	Bezeichnung / Modell / Ausstattung	
CPU	INTEL CORE i7 3615QM	AMD ATHLON II X4 435
GPU	NVIDIA GEFORCE GT 650M, 1GB	NVIDIA GEFORCE GTS 250, 1GB
RAM	16GB (2*8GB) DDR3L, 1600MHz	8GB (2*4GB) DDR3, 1333MHz
SSD	SAMSUNG SSD 830, 256GB, SATA3	CRUCIAL M5, 128GB, SATA2
OS	MICROSOFT WINDOWS 8 Professional (64Bit)	

Tabelle 6.1: Systemkonfiguration der Testsysteme

6.1.2 Werkzeuge

Die Komponenten wurden mit den folgenden Bibliotheken und Werkzeugen entwickelt und getestet. Dabei wurden jeweils die aktuellen Versionen eingesetzt:

Programmiersprache Auf beiden Systemen war das aktuelle Oracle Java SE Development Kit (JDK) 7u45 mit 32 Bit (Windows) bzw. 64 Bit (OSX) installiert. Java

6 Prototypische Implementierung

wurde als Programmiersprache gewählt, da der entstehende Programmcode weitestgehend plattformunabhängig ist und HEALTHTELKON auch in Java implementiert wurde.

Entwicklungsumgebung Die Implementierung wurde mit der integrierten Entwicklungsumgebung (IDE) ECLIPSE in der Version 4.3.1 (Kepler) durchgeführt. Für den Entwurf der VAADIN Benutzeroberflächen wurde die Erweiterung VAADIN *Plug-in for ECLIPSE* in Version 2.2 genutzt.

Versionsverwaltung Für die Verwaltung des Quellcode, wurde das Versionsverwaltungssystem GIT eingesetzt. GIT ist dezentral und jeder Client hält eigene Kopien des kompletten Repositories im lokalen Dateisystem. GIT ist daher auch als *Backup Lösung* eingesetzt worden. Dabei wurden Daten regelmäßig auf einen externen Host übertragen (gepushed).

Build-Management-Tool Um die Verwaltung genutzter Bibliotheken zu vereinfachen, wurde das Build-Management-Tool MAVEN genutzt. Durch die Nutzung von MAVEN werden Abhängigkeiten der Bibliotheken aufgelöst und die jeweils benötigten Versionen von Drittbibliotheken nachgeladen.

REST-Webservice Framework Für die Implementierung der Webservices mit REST, wurde das JERSEY *RESTful Web Services Framework* in Version 2.4.1 eingesetzt. JERSEY ist die Referenzimplementierung für JAX-RS und wird unter der GPLv2 Lizenz bereitgestellt.

Web Application Framework Das implementierte Portal wurde mit dem Framework VAADIN (Version 7.1.5) entwickelt. VAADIN basiert auf dem GOOGLE WEB TOOLKIT, ist mit zahlreichen Webbrowsern kompatibel und lässt sich mit der Programmiersprache JAVA entwickeln. VAADIN wird unter der Apache-Lizenz 2.0 bereitgestellt.

Datenbanksystem Als Datenbanksystem wurde die Open-Source Lösung POSTGRESQL (Version 9.2.5) genutzt und über den entsprechenden JDBC Treiber angebunden. Das Abbilden der objektorientierten Daten (ORM) erfolgt über die *Java Persistence API 2.0*, mit Hilfe der Implementierung ECLIPSELINK (2.5.1).

Application Server Als JAVAEE Application Server wurde die Open-Source Variante von GLASSFISH in Version 3.1.2.2 eingesetzt.

6.1 Verwendete technische Hilfsmittel für die Implementierung

Shamir's Secret Sharing Die Geheimnisteilung wurde mit der Bibliothek SHAMIR SECRET SHARING IN JAVA¹ (Version: 1.1) umgesetzt. Die Bibliothek wird unter der LGPLv2 Lizenz bereitgestellt.

FastDTW Für den Abgleich von Signaturen wurde das FastDTW Verfahren verwendet. Eine Implementierung dieses Verfahrens ist Bestandteil der Bibliothek JAVA-ML² (Version 0.1.7). Die Bibliothek wird unter der GPLv2 Lizenz bereitgestellt.

Kryptografische Algorithmen Für Hashverfahren und asymmetrische/symmetrische Verschlüsselung wurden die Implementierungen der Bibliothek BOUNCY CASTLE (Version 1.49) genutzt. Die Bibliothek wird unter der MIT Lizenz bereitgestellt.

Grafiken und Diagramme wurden mit den folgenden Werkzeugen erstellt

UML Editor Sämtliche UML Diagramme wurden mit dem Editor VISUAL PARADIGM FOR UML (Version 10.2) entworfen. Der Editor ist auf mehreren Plattformen verfügbar und wurde in der Programmiersprache JAVA implementiert.

Diagrammeditor Alle übrigen Diagramme, sofern nicht als Quelle angegeben, wurden mit dem Diagrammeditor OMNIGRAFFLE (Version 6.0.2) erstellt.

6.1.3 Wacom Tablet

Im Rahmen dieser Masterarbeit sollen biometrische Unterschriften als biometrisches Merkmal genutzt werden. Dies ist sinnvoll, da Unterschriften im Arbeitsalltag von Ärzten und medizinischen Dienstleistern bereits nahtlos integriert sind und schnell erfasst werden können.

Zur Erfassung von Unterschriften werden sogenannte Unterschriftenpads eingesetzt. Für die prototypische Implementierung in dieser Masterarbeit wurde das Modell STU-500 (Abb. 6.1) der Firma WACOM ausgewählt.

Technische Details Das STU-500 ist ein Signaturpad mit monochromem 5" LCD Display. Über das Display können Informationen dargestellt oder mit einem Stylus durch interaktive Masken navigiert werden. Der Stylus ist ein spezieller Stift mit drucksensitiver Spitze, welche 512 Druckstufen erfassen kann. Das 5" Eingabefeld tastet mit einer Auflösung von 2540 lpi (Lines Per Inch) ab und kann 200 Druckpunkte pro Sekunde übertragen.

¹SourceForge Projektseite: Shamir Secret Sharing in Java <http://sourceforge.net/projects/secretsharejava/> (Stand: 21.11.2013)

²Java Machine Learning Library (Java-ML) <http://java-ml.sourceforge.net/> (Stand: 21.11.2013)



Abbildung 6.1: Unterschriftenpad WACOM STU-500 (Quelle: Wacom)

Referenzen Die Unterschriftenpads STU-500 von WACOM werden bereits seit einigen Jahren am Markt eingesetzt. Der Hersteller SOFTPRO vertreibt sie unter der Bezeichnung SIGNPAD ESIGNIO und gibt entsprechende Referenzen und Zulassungen³ an. So werden sie von Volks-/Raiffeisenbanken, Sparkassen und der Bundesdruckerei zur Erfassung von Unterschriften empfohlen.

Wacom SDK Die Firma WACOM bietet für die Unterschriftenpads aus der STU Serie zwei verschiedene Entwicklungstools. Beide sind ausschließlich für Windows Betriebssysteme verfügbar, nutzen native Bibliotheken für Zugriffe auf Betriebssystemsfunktionalität und stellen diese über das *Java Native Interface* (JNI) bereit.

Das sogenannte *Signature SDK* kann direkt für die Anfertigung von Signaturen genutzt werden. Es bietet dem Programmierer die Möglichkeit biometrische Signaturen in einem proprietären Format zu speichern, um damit direkt Daten zu signieren. Es gibt jedoch keine Möglichkeit, die Rohdaten der biometrischen Signatur zu nutzen. Das *Signature SDK* stellt biometrische Unterschriften durch eine 2 dimensionale Grafik dar, die auch Informationen über Druck und die Zeit enthält. Für einen Vergleich von biometrischen Signaturen sind diese Grafiken jedoch nicht ausreichend, da so nur *Off-Line Verfahren* eingesetzt werden können.

Die zweite Möglichkeit mit dem Unterschriftenpad zu kommunizieren stellt das *Low Level SDK* (LLSDK) dar. Es bietet die grundlegenden Funktionalitäten Eingaben vom Unterschriftenpad aufzuzeichnen und einfache Grafiken auf dem Display darzu-

³ SOFTPRO Webseite (Stand: 30.07.13)

stellen. Die wichtigste Funktion stellt die Aufzeichnung der Rohdaten dar, welche das Unterschriftenpad in Echtzeit ausgibt.

Von den beiden genannten Entwicklerwerkzeugen wurde für die Implementierung das *Low Level SDK* genutzt, da nur so numerische Daten für *On-Line Verfahren* aufgezeichnet werden können.

6.2 Implementierung der Komponenten

Nachdem die Komponenten auf einer weniger technischen Ebene in Unterabschnitt 5.1.3 entworfen wurden, soll nun die Implementierung auf technischer Ebene erläutert werden. Hierbei sollen die wichtigen Implementierungsdetails beschrieben werden, die die entworfenen Komponenten mit Funktionalität füllen.

6.2.1 HYDrOUS Business Logic

Die HYDrOUS Business Logic implementiert und koordiniert die Daten der beteiligten Komponenten. Für diesen Einsatzzweck, muss sie über alle Datenquellen verfügen und stellt als Vermittler zwischen den Komponenten Schnittstellen bereit.

Modellierung des Datenmodells

Als zentrale Komponente, verwendet die HYDrOUS Business Logic Daten aus den angebenen Komponenten um Benutzer, Signaturen, Dokumente, Geheimnisanteile und Schlüssel der teilnehmenden Parteien im Mehrparteien-Signaturprotokoll einzusetzen.

Für die Implementierung werden diese Daten aus relationalen Datenbanken mittels objektrelationaler Abbildung (ORM) für die Implementierung in einer objektorientierten Programmiersprache abgebildet. Die so entstehende Klassenstruktur ist in Abb. 6.2 als UML Klassendiagramm dargestellt.

Die zentrale Klasse im Modell stellt `Document` dar. Sie kennt die unterzeichnenden Parteien als Instanzen der Klasse `Signee` und hält außerdem einen `Signee` als administrativen Benutzer. Letzterer ist für gewöhnlich auch Teil der unterzeichnenden Parteien. Allen `Signee` Instanzen ist ein Attribut `PublicKey` zugewiesen, welches von der PKI bereitgestellt wird. Außerdem erbt die Klasse `Signee` von der Klasse `User` und hält somit Benutzerdaten wie `Name (name)`, `E-Mail Adresse (email)` und `Fachbereich (speciality)`.

Jedes `Document` hat genau eine `Signature`, die den Hashwert (`hash`) des Dokumentes (`Document.content`) als Gesamtgeheimnis enthält. Jede `Signature` verwaltet zwei assoziative Datenfelder (`Map`) von `Signee` zu `SecretShare` bzw. `String` (signierte Geheimnisanteile). In der `Map secretShares` werden die von der TTP

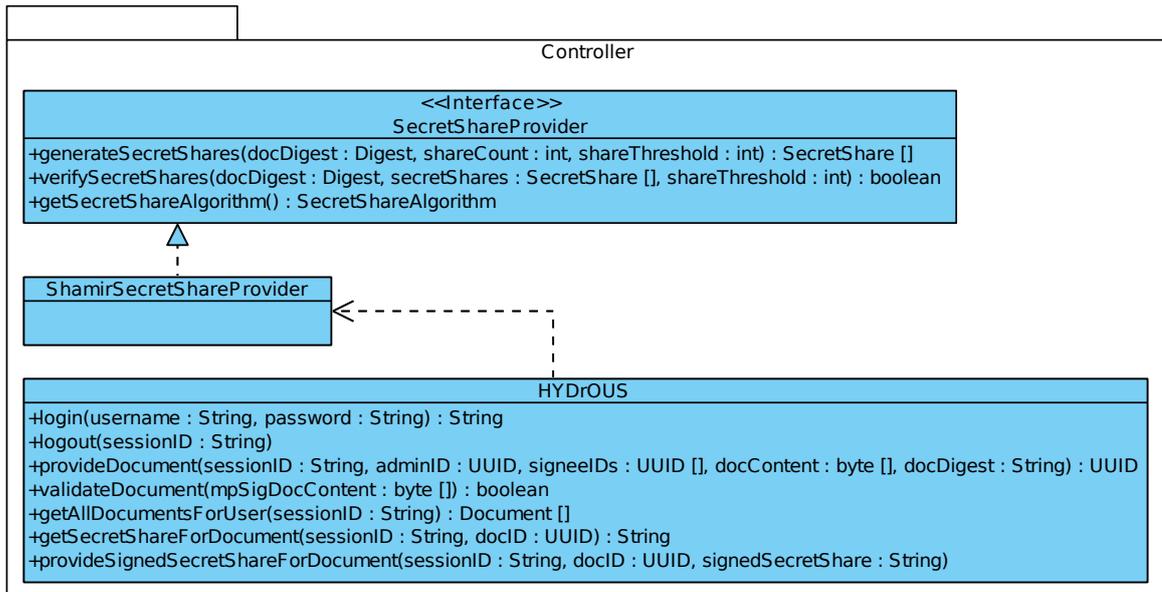


Abbildung 6.3: UML Klassendiagramm des Pakets `*.hydrous.controller`

Logic ausweisen, ohne bei jedem Datenaustausch sein Benutzernamen und Kennwort zu übertragen. Soll eine Sitzung beendet werden, muss die Methode `logout` mit der Session ID als Parameter aufgerufen werden und die Session ID so widerrufen werden.

Die für eine Authentifizierung benötigten Daten werden über eine Schnittstelle zu einer Datenbank abgerufen.

Exception (login): `WrongCredentialsException`

Exception (logout): `SessionExpiredException`

Methode `provideDocument` Um PDF-Dokumente in HYDrOUS einzustellen, wird die Methode `provideDocument` genutzt. Beim Einstellen werden Parametern übergeben, die alle Unterschreiber (`signees`) und den Konferenzmanager (`admin`) als eindeutigen Bezeichner (UUID), den Inhalt des Dokumentes (`docContent`) und eine Prüfsumme des Inhaltes (`docDigest`) umfassen. Als Rückgabe wird die eindeutige UUID des neu eingestellten Dokumentes zurückgegeben.

Exceptions: `SessionExpiredException`, `SigneesMissingException`, `AdminMissingException`, `DocumentIntegrityException`

Methode `validateDocument` Um ein Dokument zu validieren, muss es in dem Format vorliegen, welches HYDrOUS nach einer erfolgten Mehrparteien-Signatur generiert. Wird ein solches Dokument als Parameter `mpSigDocContent` bereitgestellt, wird es entsprechend validiert und das Ergebnis über die Wahrheitswerte `true` oder `false` zurückgegeben.

6 Prototypische Implementierung

Exception: `DocumentValidationException`

Methode `getAllDocumentsForUser` Diese Methode liefert sämtliche Dokumente zurück, an deren Mehrparteien-Signatur der angemeldete Benutzer beteiligt ist.

Hinweis: Dem angemeldeten Benutzer werden niemals die kompletten Objekte der Klasse `Document` präsentiert, da er sonst auf die Geheimnisanteile der anderen Unterzeichner Zugriff hätte. Stattdessen werden in den entsprechenden Repräsentationen (REST-Webservice und VAADIN-WebPortal) nur abgeleitete Informationen ausgegeben.

Exception: `SessionExpiredException`

Methode `getSecretShareForDocument` Die Methode liefert dem angemeldeten Benutzer für das `Document` mit dem eindeutigen Bezeichner (`docID`) den mit seinem öffentlichen Schlüssel verschlüsselten Geheimnisanteil.

Exceptions: `SessionExpiredException`, `NotInSigneesException`, `DocumentNotFoundException`

Methode `provideSecretShareForDocument` Die bereits signierten Geheimnisanteile werden mit dieser Methode der Signatur des Dokumentes mit dem Bezeichner `docID` hinzugefügt. Die Methode prüft nach dem Hinzufügen, ob bereits alle signierten Geheimnisanteile in der `Signature` eines `Document` vorhanden sind. Ist dies der Fall, werden die bereitgestellten Geheimnisanteile entschlüsselt, zusammengefügt und bei Korrektheit die Gesamtsignatur in `multiPartySignature` hinterlegt.

Exceptions: `SessionExpiredException`, `SecretShareSignedBeforeException`, `SecretShareInvalidException`

Methoden des Interfaces `SecretShareProvider`

Das Interface `SecretShareProvider` beschreibt die Schnittstellen, die eine konkrete Implementierung liefern muss. Für die Implementierung des Prototyps wurde das Secret-Sharing Verfahren von Shamir ausgewählt und in der Klasse `ShamirSecretShareProvider` implementiert.

Methode `generateSecretShares` Diese Methode erstellt aus einer gegebenen Dokumentenprüfsumme `docDigest` genau `shareCount`-viele Geheimnisanteile. Über den zweiten Parameter `shareThreshold` kann ein Schwellwert bestimmt werden der angibt, wie viele der Geheimnisanteile nötig sind, um das Gesamtgeheimnis zu rekonstruieren.

Exceptions: `SecretSizeException`, `ShareCountValueException`, `ShareThresholdValueException`

Methode `verifySecretShares` Diese Methode dient der Überprüfung von gesammelten Geheimnisanteilen (`secretShares`) zu einem bekannten Geheimnis (`docDigest`). Über den zweiten Parameter `shareThreshold` kann der Schwellwert festgelegt werden, der beim Erstellen der Geheimnisanteile genutzt wurde.

Exceptions: `SecretShareValueException`, `ShareThresholdValueException`

Schnittstellen für Komponenten

Für Komponenten, die mit der HYDrOUS Business Logic nicht direkt über JAVA Schnittstellen kommunizieren, werden Webservices mit REST-Schnittstelle bereitgestellt.

REST-Ressourcen `Document`, `PublicKey`, `SharedSecret` und `Signee` Ein Dokument ist über die URI „`.../documents/{docuuuid}`“ verfügbar. Statt alle Attribute der Klasse `Document` weiterzureichen, werden Informationen so aufbereitet, dass sie dem authentifizierten Benutzer nur anwendungsbezogene Informationen bereitstellen. So werden insbesondere die `SecretShares` anderer Benutzer nicht ausgegeben. Stattdessen werden die Werte `secretShareSigned` und `signCount` aus den Attributen abgeleitet. Mit der HTTP-Methode `GET` wird die in Quelltext 6.1 dargestellte Antwort gegeben. Soll ein Dokument bereitgestellt werden, wird eine `POST` Anfrage gestellt, die nur die Parameter `admin`, `signees` und `content` enthält.

```
1 href:    ".../documents/docuuuid",
2 admin:   { href: ".../signees/adminuuid" },
3 signees: [ { href: ".../signees/signeeuuid1" },
4             { href: ".../signees/signeeuuid2" },
5             ...
6             { href: ".../signees/signeeuuidN" } ],
7 signature: {
8   secretShare: { href: ".../secretShares/shareuuid1" },
9   secretShareSigned: true,
10  signCount: "4/N",
11  multiPartySignature: null },
12 content: { href: ".../documents/docuuuid/content" }
```

Quelltext 6.1: JSON Repräsentation von `Document`

Die Unterzeichner hingegen werden vollständig repräsentiert (Quelltext 6.2) und können als Ressource über die URI „`.../signees/{useruuid}`“ abgerufen werden.

```
1 href: ".../signees/signeeuuid",
2 name: "Dr. med. Max Mustermann",
3 email: "max@praxis-mustermann.de",
```

6 Prototypische Implementierung

```
4 speciality: "Onkologie",  
5 publicKey: { href: ".../publicKeys/pkuuid" }
```

Quelltext 6.2: JSON Repräsentation von Signee

Auch die Ressourcen für öffentliche Schlüssel (URI: „.../publicKeys/{pkuuid}“) und Geheimnisanteile (URI: „.../secretShares/{shareuuid}“) werden vollständig repräsentiert. Letztere jedoch nur, wenn der Anfragende die entsprechende Berechtigung hat.

6.2.2 HYDrOUS Biometric Matcher

REST-Ressourcen des Biometric Matcher Für den biometrischen Abgleich, muss eine Ressource geschaffen werden, welche die Anforderungen der Anwendungsfälle Abschnitt 4.4 UC3, UC4, UC5, UC6 und UC7 und des in Unterabschnitt 5.2.2 beschriebenen Protokolls unterstützt.

Der Client fordert beim Matcher einen `nonceM` an

```
1 .../users/useruuid/biometricSecret/hello
```

Quelltext 6.3: GET Anfrage an den Biometric Matcher für `nonceM`

und startet dann den biometrischen Abgleich durch eine GET Anfrage, wie in Quelltext 6.4. Die nötige Authentifizierung gegenüber dem Matcher wird über HTTP Basic durchgeführt.

```
1 .../users/useruuid/biometricSecret?sessionSecret=vbj0iMS4wImVuY29...dHV  
&bioSig=PD94bWwgdMvYc2l...yZE+
```

Quelltext 6.4: GET Anfrage an den Biometric Matcher

Der Parameter `sessionSecret` enthält hierbei den verschlüsselten Sitzungsschlüssel, `nonceB`, `nonceM` und der Parameter `bioSig` die verschlüsselte biometrische Unterschrift.

Soll ein biometrisches Template oder ein Benutzerkennwort geändert werden, wird auf der entsprechenden Ressource die PUT Anweisung mit weiteren Parametern durchgeführt. Für die Änderung eines Benutzerkennwortes (URI: „.../users/{useruuid}/password“) muss sich der Client gegenüber dem Matcher authentifizieren (mit HTTP Basic) und seine biometrische Signatur (`bioSig`) bestätigen.

```
1 sessionSecret: "vbj0iMS4wImVuY29...dHV",  
2 bioSig: "PD94bWwgdMvYc2l...yZE+",  
3 newPassword: "EncryptedNewSecret"
```

Quelltext 6.5: JSON Parameter: Kennwort aktualisieren

Der Parameter `newPassword` enthält das neue Benutzerkennwort in verschlüsselter Form.

Für Die Änderung des biometrischen Templates (URI: „.../users/{*useruuid*}/biometricSecret“) authentifiziert sich der Client wiederum gegenüber dem Matcher und gibt seine aktuelle biometrische Unterschrift verschlüsselt (`bioSig`) an. Das neue biometrische Template wird durch den Matcher generiert. Dazu versendet der Client eine vereinbarte Anzahl von verschlüsselten biometrischen Unterschriften (z. B. 3) im Parameter `bioSeed`, aus denen das Template generiert wird.

```
1 sessionSecret: "vbj0iMS4wImVuY29...dHV",  
2 bioSig: "AhadEbWwgsafDml...ydf+",  
3 bioSeed: [ "PD94bWwgdMvYc2l...Uid+",  
4           ...  
5           "Adas4bWwgdMvYc2l...PoL+" ]
```

Quelltext 6.6: JSON Parameter: Biometrisches Template aktualisieren

6.2.3 HYDrOUS Biometric Signature Extension

Die prototypische JAVA Desktop Anwendung sieht wie in Abb. 6.4 abgebildet aus. Im Screenshot sieht man eine frisch aufgezeichnete biometrische Unterschrift, wobei die Farben jeweils den Druck am jeweiligen Punkt repräsentieren (türkis < grün < gelb < orange < rot). Über die Reiter am unteren Fensterrand kann man Zugriff auf die weiteren Funktionen (siehe Unterabschnitt 5.1.3) erlangen.



Abbildung 6.4: Screenshot der HYDrOUS Biometric Signature Extension

6.3 Probleme während der Implementierung

Während jeder Implementierung gilt es gewisse Hindernisse zu überwinden, die beim Einsatz von verschiedenen Bibliotheken, Systemen und Werkzeugen auftreten können. In diesem Abschnitt werden einige dieser Hindernisse beschrieben.

6.3.1 Wacom Low Level SDK

Das WACOM Low Level SDK, welches für die Aufzeichnung der biometrischen Unterschriften genutzt wurde, hat in der in dieser Arbeit genutzten Version (1.1.35) noch einige Fehler. Die Version für 64 Bit Windows Betriebssysteme lief nicht stabil, daher wurde für die Implementierung die 32 Bit Version eingesetzt. Aber auch diese Version hat einige Fehler, die nur mit Workarounds behoben werden konnten. Zwar bietet das SDK an die Koordinaten, den Druck und die Zeit aufzuzeichnen, letzteres funktioniert jedoch nicht.

```
1 penPoint.setTime((int) ((System.nanoTime() - startTime) / 1000000));
```

Quelltext 6.7: LLSDK manuelles Speichern der Zeit

Stattdessen wird die Zeit manuell über die Systemzeit in den einzelnen Messpunkten gespeichert (Quelltext 6.7). Ein weiteres Problem tritt auf, wenn man mehrere Unterschriften nacheinander aufzeichnen möchte. Obwohl das Tablet über das SDK getrennt wird, bleiben die zuvor aufgezeichneten Messpunkte im Speicher erhalten und müssen manuell gelöscht werden, wie in Quelltext 6.8 angegeben.

```
1 ArrayList<PenPoint> pp = JSTUTablet.getPenPoints();  
2 pp.clear();
```

Quelltext 6.8: LLSDK manuelles löschen der Messpunkte

6.3.2 JAVA Applet

Obwohl zunächst der Einsatz eines JAVA Applets geplant wurde, ist dieses Vorhaben verworfen worden. Viele moderne Browser warnen den Benutzer vor Applets oder deaktivieren diese als Voreinstellung. Als Alternative wurde die HYDRIOUS Biometric Signature Extension als Desktop Applikation umgesetzt, die mittels JAVA WebStart auch direkt über einen Browser gestartet werden könnte. JAVA WebStart nutzt hierzu ein XML-Dokument im Java Network Launching Protocol (JNLP)-Format, in dem der Ablageort der JAR-Datei, die Hauptklasse und weitere Parameter enthalten sind. Insbesondere gibt es die Möglichkeit die ausführbaren JAR-Dateien elektronisch zu signieren. Wird die JNLP-Datei im Browser aufgerufen, wird sie an die JAVA Laufzeitumgebung weitergereicht.

7 Evaluation

Im folgenden Kapitel soll das in dieser Masterarbeit konzipierte System evaluiert werden. Hierzu wird die Güte des biometrischen Abgleiches und wie diese festgestellt wurde beschrieben. Da das System in einem sicherheitsrelevanten Umfeld eingesetzt werden soll, wird es auf Verwundbarkeit für verbreitete Angriffe untersucht und entsprechende Lösungen für Gegenmaßnahmen erläutert. Neben diesen Angriffen wird auch die Anfälligkeit für Kompromittierung von Client und/oder Server begutachtet.

7.1 Biometrischer Abgleich

Für die Auswahl der Merkmale wurde ein Hilfswerkzeug implementiert (Abb. 7.1), das es ermöglicht zwei biometrische Unterschriften miteinander zu vergleichen (rot/blau in Abb. 7.1). Das Werkzeug stellt vier lokale Merkmale der biometrischen Signaturen gegenüber, die über Komboboxen frei miteinander kombiniert werden können. Zusätzlich gibt das Werkzeug die mit FastDTW (Paragraph 3.1.6) berechneten Distanzen bei lokalen Merkmalen und die Differenz für globale Merkmale für zwei biometrische Unterschriften textuell über die Konsole aus. Die Versuche für biometrische Abgleiche wurden mit Unterschriften von zwölf Testpersonen durchgeführt, um natürliche Schwankungen von biometrischen Unterschriften messen zu können. Aufgrund dieser Testreihen wurden die in Unterabschnitt 5.2.3 beschriebenen Merkmale ausgewählt. Eine Auswertung der Distanzen für die lokalen und globalen biometrischen Merkmale ist in Tab. 7.1 dargestellt.

Für die Berechnung der Punktzahl wurde von einer Maximalpunktzahl von 100000 eine gewichtete Differenz für jedes Merkmal subtrahiert:

$$score := 100000 - \sum_m^{\text{Merkmale}} (dist(m) - dist_{\text{mean}}(m)) * \frac{10000}{dist_{\text{mean}}(m)}$$

Je geringer der Mittelwert, desto höher ist die Gewichtung eines Merkmals. Die Punktzahl *score* war im Mittelwert 83828.33 und im Median 88304. Von den zwölf Testern konnten nur drei deutlich unter 80000 Punkte erreichen. Deswegen wurde für den Prototypen ein Schwellwert von 80000 Punkten für einen erfolgreichen Abgleich gewählt.

7 Evaluation

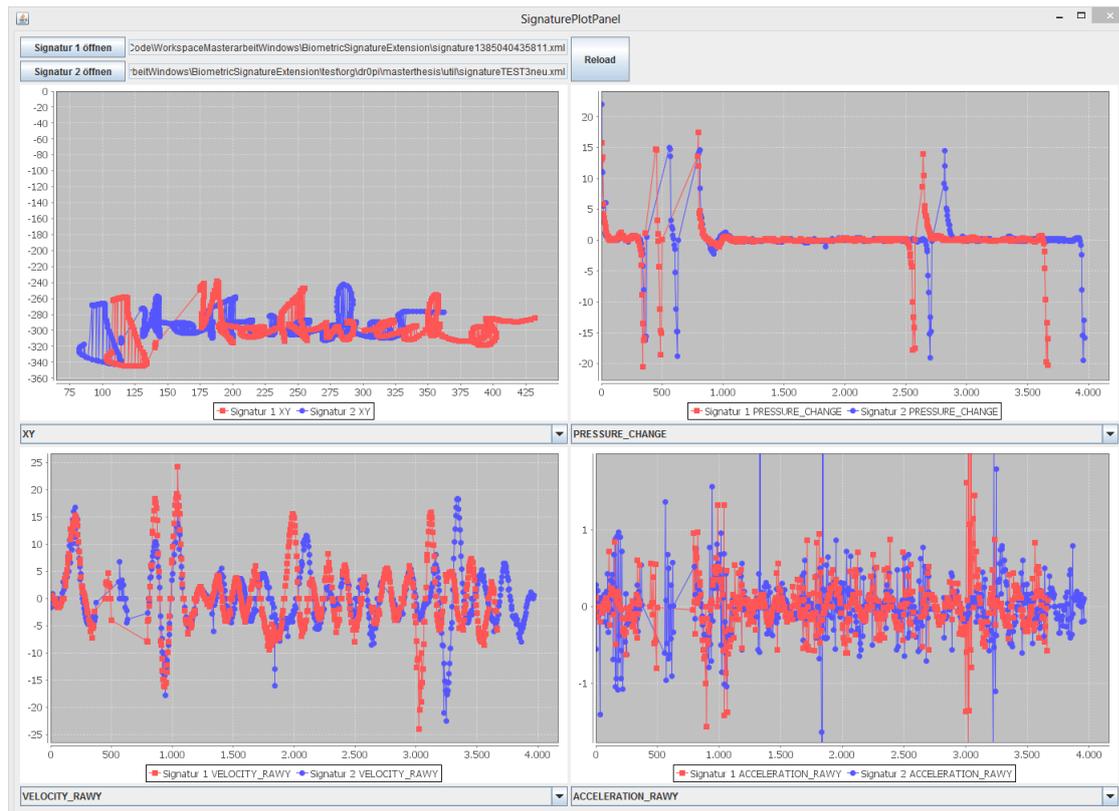


Abbildung 7.1: Screenshot „SignaturePlotPanel“

Merkmalsname	Mittelwert	Median	Minimum	Maximum
Druckveränderung	39.5515	29.5490	10.3056	100.9251
Geschwindigkeit X	108.4369	96.5162	21.7722	299.2381
Geschwindigkeit Y	120.3669	98.0322	30.2833	266.7428
Beschleunigung X	9.9011	8.5092	2.3558	30.5348
Beschleunigung Y	12.2933	11.3374	4.3471	27.4337
Zeitspanne	121.8889	67.5000	10.3333	352.3333
Seitenverhältnis	0.3350	0.2827	0.0316	0.7801
Anzahl der Striche	2.5278	2.5000	0.0000	6.6667

Tabelle 7.1: Evaluation der biometrischen Merkmale (Distanzen)

Anmerkung: Diese Testreihen sind statistisch nicht repräsentativ, da mit einer geringen Anzahl von Probanden getestet wurde und Spezialfälle wie „Profifälscher“ nicht berücksichtigt werden konnten.

7.2 Erfüllung der funktionalen und nicht-funktionalen Anforderungen

Funktionale Anforderungen:

- FA1 **Lokale Schlüsselerzeugung:** Umsetzung durch die Desktop Applikation „HYDrOUS Biometric Signature Extension“.
- FA2 **Schlüsselverwaltung (öffentliche Schlüssel):** Umsetzung durch die Komponente „PKI“ mit Anbindung an REST-Webservice.
- FA3 **Dokumente katalogisieren:** Umsetzung durch die Komponente „HYDrOUS Business Logic“.
- FA4 **Dokumente bereitstellen:** Umsetzung durch die „HYDrOUS Business Logic“ als REST-Webservice Schnittstelle für HEALTHTELKON.
- FA5 **Zugriffskontrolle für Dokumente:** Umsetzung durch die Authentifizierung gegenüber der „HYDrOUS Business Logic“.
- FA6 **Benutzerverwaltung:** Umsetzung durch die Authentifizierung und Dokumentenmetadaten in der „HYDrOUS Business Logic“.
- FA7 **Authentifizierung im System:** Umsetzung durch die „HYDrOUS Business Logic“ und die gemeinsame Benutzerdatenbank mit HEALTHTELKON, sowie den „HYDrOUS Biometric Matcher“.
- FA8 **Benutzerrollen:** Umsetzung durch die „HYDrOUS Business Logic“ durch Authentifizierung und Metadaten der Dokumente.
- FA9 **Biometrische Unterschriften erfassen:** Umsetzung durch die „HYDrOUS Biometric Signature Extension“.
- FA10 **Biometrische Unterschriften aufzeichnen:** Umsetzung durch die „HYDrOUS Biometric Signature Extension“ zusammen mit „HYDrOUS Biometric Matcher“.
- FA11 **Biometrische Unterschriften abgleichen:** Umsetzung durch den „HYDrOUS Biometric Matcher“.

Nicht-funktionale Anforderungen:

- NFA1 **Bedienbarkeit über einen Webbrowser:** Die Komponente HYDrOUS Portal stellt die grafische Oberfläche und die Verwaltung von Dokumenten und Signaturvorgängen im Webbrowser dar. Die Komponente „HYDrOUS Biometric Signature Extension“ wurde für die prototypische Implementierung als Desktop

7 Evaluation

Applikation umgesetzt. Sie kann jedoch als JAVA-Web-Start-Anwendung aus dem Browser heraus gestartet werden.

- NFA2 Systemvoraussetzungen für Benutzer:** Die Systemvoraussetzungen entsprechen den in der Implementierung angegebenen Implementierungssystemen (Unterabschnitt 6.1.1).
- NFA3 Interoperabilität:** Durch die Aufteilung in Komponenten ist eine Erweiterung für andere Methoden zur Authentifizierung möglich. Entsprechende Komponenten müssten dann die Funktionalität der Komponente „HYDrOUS Biometric Signature Extension“ und „HYDrOUS Biometric Matcher“ implementieren.
- NFA4 Rechtliche Vorgaben:** Das in dieser Masterarbeit umgesetzte System hält sich an die Anforderungen für „fortgeschrittene elektronische Signaturen“ (Unterabschnitt 2.2.1).
- NFA5 Sicherheitsanforderungen:** Die Maßnahmen gegen Angriffe werden im folgenden Abschnitt betrachtet.

7.3 Angriffe

Spricht man von einem Angriff auf ein System, so kann dieser verschiedene Herangehensweisen haben. Im Folgenden sollen Standardangriffe und die zu treffenden Gegenmaßnahmen erläutert werden.

7.3.1 Brute-Force Angriff

Brute-Force-Angriffe versuchen über „rohe Gewalt“ mit allen möglichen Schlüssel eines gegebenen Alphabets, ein Geheimnis zu entschlüsseln. Da die Schlüsselräume in modernen kryptografischen Verfahren mit entsprechender Schlüssellänge jedoch sehr groß sind, sind der erforderliche Rechenaufwand und Zeitaufwand zu hoch. Ist der Schlüsselraum mit großer Wahrscheinlichkeit begrenzt, wie bei Benutzerkennwörtern, kann statt aller Schlüssel im Schlüsselraum ein Wörterbuchangriff durchgeführt werden, bei dem vordefinierte Listen von Schlüssel überprüft werden.

Lösung: Um Brute-Force-Angriffe abzuwenden, werden z. B. vom BSI Anforderungen an Schlüssellängen und die kryptografischen Algorithmen gestellt (siehe Unterabschnitt 2.2.3). Diese Empfehlungen werden sowohl für die Konzeption als auch für die prototypische Implementierung in dieser Masterarbeit befolgt. Bei der Implementierung der Komponenten wurden außerdem Maßnahmen getroffen, die mögliche Brute-Force-Angriffe erheblich verlangsamen: Die Authentifizierung gegenüber der HYDrOUS Business Logic wird nach drei fehlerhaften Anmeldeversuchen

für fünf Minuten blockiert, sodass ein Angreifer auch die verhältnismäßig kurzen Benutzerkennworte nicht in absehbarer Zeit erschöpfend testen kann. Der HYDrOUS Biometric Matcher blockiert den Abgleich zwischen biometrischer Unterschrift und biometrischem Template nach zehn Fehlversuchen für fünf Minuten.

7.3.2 Man-in-the-Middle-Angriff

Bei einem Man-in-the-Middle-Angriff versucht eine dritte Partei sich unbemerkt zwischen zwei Kommunikationspartner zu positionieren (siehe Abb. 7.3b), um sämtlichen Datenaustausch mitschneiden zu können. Der Angreifer gibt sich für die Parteien jeweils als der andere aus, sodass diese unbehelligt Geheimnisse austauschen können.

Lösung: Einem solchen Angriff kann entgegengewirkt werden, wenn die beiden Kommunikationspartner gesendete Nachrichten nur authentifiziert versenden, also digital signieren und jede empfangene Nachricht validieren. So kann die Integrität der Nachricht sichergestellt werden. Soll ein Angreifer keine Nachrichten abhören können, dürfen diese nur verschlüsselt übertragen werden.

Die in dieser Masterarbeit beschriebenen Protokolle tun dies bei jedem Nachrichtenaustausch, indem die Kommunikation stets mit einer Ende-zu-Ende Verschlüsselung durchgeführt wird. Beim Mehrparteien-Signaturprotokoll wird zusätzlich jede Nachricht digital signiert.

7.3.3 Replay-Angriff

Bei einem Replay-Angriff versucht ein Angreifer mitgeschnittene Nachrichten erneut zu senden.

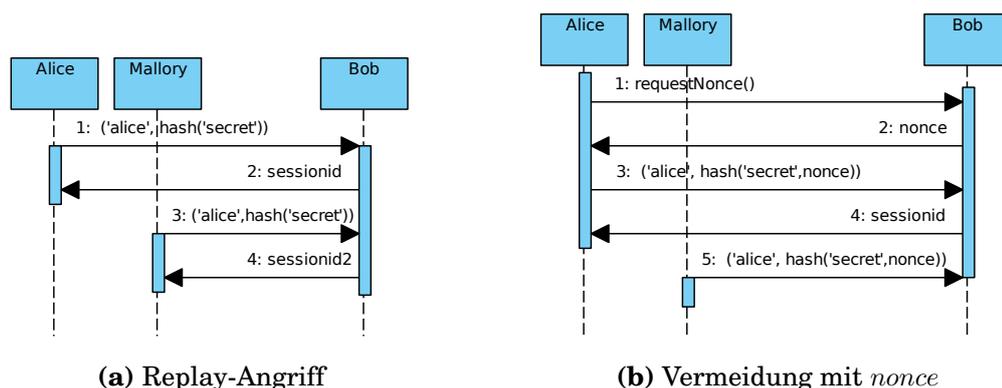


Abbildung 7.2: Replay Angriff und Gegenmaßnahme

Beispiel: Zur Authentifizierung erstellt der Client eine Nachricht mit Benutzernamen und Hashwert seines Benutzerkennwortes und sendet diese an den Server. Der Angreifer schneidet diese Nachricht mit und kann durch einfaches Wiedereinspielen dieser Nachricht eine Authentifizierung durchführen.

7 Evaluation

Lösung: Das erneute Senden muss durch technische Mittel erkennbar sein. Eine mögliche Lösung ist der Einsatz von Einmalwerten (*nonce*). Hierbei sendet der Server dem Client mit einer authentifizierten Nachricht eine *nonce*. Der Client legt seiner Nachricht nun diese *nonce* wie in Abb. 7.2 bei. Der Server sieht die *nonce* und akzeptiert danach keine Nachricht mit der gleichen *nonce*. Erwartet der Client eine Antwort vom Server legt er auch seiner Nachricht eine (neue) *nonce*₂ bei.

Das biometrische Schlüsselfreigabe Protokoll ruft vor der Authentifizierung eine *nonce*_M vom Matcher ab und fügt diese seiner initialen Anfrage hinzu. (Würde der Benutzer dies nicht tun, könnte ein Angreifer sich mit dieser Nachricht authentifizieren) Der Matcher akzeptiert für diesen Benutzer jedoch nicht zweimal die selbe *nonce*_M. Da der Benutzer sich der Identität des Matches sicher sein will, legt er seiner Anfrage ebenfalls eine *nonce*_B bei. Der Matcher versendet in seiner Antwort diese *nonce*_B, wodurch der er bewiesen hat das (PK_M, SK_M)-Schlüsselpaar zu besitzen. Der Client akzeptiert danach keine zweite Antwort mit der gleichen *nonce*_B.

Das Mehrparteien-Signaturprotokoll kann auf die Nutzung von *nonce* angepasst werden, um Replay Angriffe zu verhindern. Betrachtet man jedoch die ausgetauschten Nachrichten, geht von mehrfach versendeten Nachrichten keine Gefahr aus. Führt ein Angreifer eine zweite Anfrage $p_i \rightarrow T : req_{p_i}(D.id)$ aus, erhält er eine signierte Nachricht mit dem verschlüsselten Geheimnisanteil ed_i . Da er nicht im Besitz des Schlüsselpaares von p_i ist, endet der Nutzen hier für den Angreifer. Sendet die Partei $p_i \rightarrow T : Sig_{p_i}(D)$, kann ein Angreifer den Inhalt nicht manipulieren, T kennt den zuvor signierten Geheimnisanteil bereits und verwirft die Nachricht vom Angreifer.

7.3.4 Perfect-Forward-Secrecy

Ein oft genutztes Verfahren zum Schlüsselaustausch basiert auf asymmetrischer Verschlüsselung, so auch die biometrische Schlüsselfreigabe (Unterabschnitt 5.2.2). Soll zwischen zwei Teilnehmern, Alice und Bob, ein geheimer Sitzungsschlüssel ausgehandelt werden, generiert Alice einen solchen. Danach verschlüsselt sie den Sitzungsschlüssel mit dem öffentlichen Schlüssel von Bob, damit nur Bob den Schlüssel im Klartext erhalten kann.

Ein potenzieller Angriff auf ein solches Schlüsselaustauschprotokoll wird von einem Angreifer durchgeführt, der sämtlichen Datenaustausch über ein Netzwerk mitschneiden und archivieren kann. Bringt dieser Angreifer irgendwann in der Zukunft den privaten Schlüssel von Bob in Erfahrung, ist es ihm möglich sämtliche Sitzungsschlüssel zu entschlüsseln und die mitgeschnittene Kommunikation zwischen Alice und Bob sichtbar zu machen.

Lösung: Die Perfect-Forward-Secrecy ist eine Eigenschaft von Schlüsselaustauschprotokollen, mit der sich solche Angriffe vermeiden lassen. Hierzu muss zwischen Alice

und Bob ein Protokoll vereinbart werden, welches Sitzungsschlüssel niemals überträgt und es Alice und Bob trotzdem möglich ist einen gemeinsamen Sitzungsschlüssel zu berechnen. Der *Diffie-Hellman-Schlüsselaustausch* ist ein solches Protokoll, das jedoch in seiner ursprünglichen Form (Abb. 7.3a) keine Nachrichtenauthentifizierung ermöglicht. Somit kann ein Man-in-the-Middle Angriff auftreten wie in Abb. 7.3b dargestellt. Diesem Angriff kann entgegengewirkt werden, indem die Nachrichten zwischen Alice und Bob digital signiert werden und eine Manipulation durch Mallory auffällt.

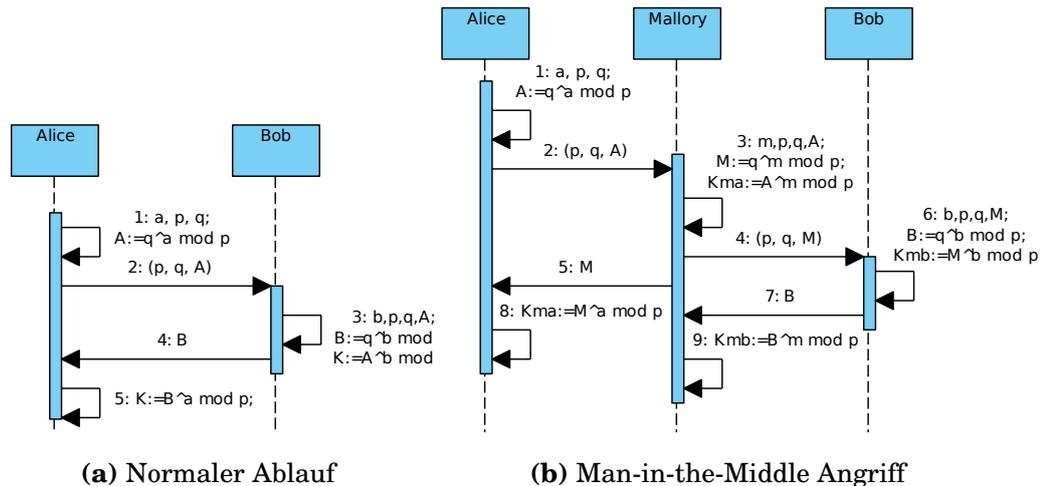


Abbildung 7.3: Diffie-Hellman-Schlüsselaustausch ohne Nachrichtenauthentifizierung

Um Perfect-Forward-Secrecy auch für die biometrische Schlüsselfreigabe zu nutzen, kann ein *Diffie-Hellman-Schlüsselaustausch* mit authentifizierten Nachrichten genutzt werden, welcher dem beschriebenen Protokoll vorangeht und $k_{u_i, M}$ erzeugt. Jeder Benutzer müsste für eine solche Ausführung ein zweites asymmetrisches Schlüsselpaar besitzen, welches nicht für die Mehrparteien-Signatur genutzt wird. Da die biometrische Schlüsselfreigabe in der prototypischen Implementierung nur über HTTPS Verbindungen auf REST-Webservice umgesetzt wird, kann auch der an dieser Stelle nötige Schlüsselaustausch auf ein Verfahren mit *Diffie-Hellman-Schlüsselaustausch* setzen (z. B. aus der TLS-DHE-RSA Familie).

Das Mehrparteien-Signaturprotokoll tauscht zwischen Parteien keine Sitzungsschlüssel aus, da sämtliche Kommunikation über asymmetrische Kryptografie geschützt werden soll. Würde ein Angreifer in der Zukunft die Schlüsselpaare der Parteien p_i erhalten, wäre die Kommunikation somit zum Teil entschlüsselbar. Allerdings wäre ein Angreifer dann auch in der Lage Signaturen zu erstellen und sich als p_i auszugeben. Beim Mehrparteien-Signaturprotokoll ist es daher sinnvoller in zeitlichen Abständen neue Schlüsselpaare für die Parteien zu erzeugen, wobei alte öffentliche Schlüssel für die Verifizierung von Signaturen bestehen bleiben. Für die

prototypische Implementierung gilt auch hier, dass HTTPS Verbindungen genutzt werden, mit denen die Perfect-Forward-Secrecy umgesetzt werden kann.

7.3.5 Kompromittierung von Benutzer PCs

Neben Angriffen auf das implementierte System sollte auch die Ausführungsumgebung auf mögliche Schwachstellen untersucht werden. Das in dieser Masterarbeit konzipierte System soll auf gewöhnlichen PCs mit WINDOWS Betriebssystem eingesetzt werden können. Aufgrund der hohen Verbreitung von WINDOWS existiert eine ebenso hohe Anzahl von Schadsoftware. Aber auch der physische Zugriff auf einen solchen PC kann von einem Angreifer genutzt werden.

Datendiebstahl Erlangt ein Angreifer Zugriff auf die hinterlegten Daten auf dem PC eines Benutzers, kann er diese kopieren. Im Fall von HYDrOUS könnte er somit den verschlüsselten Schlüsselbund des Benutzers in seinen Besitz bringen und versuchen den Schlüsselbund mit einem Brute-Force-Angriff zu entschlüsseln. Der dafür benötigte Schlüssel ist jedoch ein Hashwert des biometrischen Templates und kein einfaches Benutzerkennwort gegen das Wörterbuchangriffe eingesetzt werden könnten. Der private Schlüssel des Benutzers ist also ohne erheblichen Aufwand nicht zugänglich. Die aufgezeichneten biometrischen Unterschriften werden niemals auf der Festplatte gespeichert und nur temporär im Arbeitsspeicher gehalten.

Könnte ein Angreifer jedoch den kompletten Inhalt des Arbeitsspeichers im PC mitschneiden, wäre ein Diebstahl des privaten Schlüssels und der biometrischen Unterschriften möglich. Zwar könnte der private Schlüssel auch in verschlüsselter Form im Arbeitsspeicher liegen, der für die Entschlüsselung nötige Schlüssel jedoch auch. Gegen einen technisch so aufwändigen Angriff kann prinzipiell nur die Abkapselung des privaten Schlüssels (bzw. des Schlüsselbundes) in einen sicheren Container helfen. Eine mögliche Lösung hierfür wären *SmartCards* oder *USB Tokens*, die kryptografische Operationen intern ausführen können und den privaten Schlüssel niemals in den Arbeitsspeicher des PCs laden müssen.

Keylogger Kann ein Angreifer über Schadsoftware oder Manipulation an der Hardware direkten Zugriff auf die Tastatureingaben des Benutzers erhalten, kann er auch die für die Authentifizierung benötigten Benutzerkennworte abfangen. Der Angreifer kann sich mit diesem Benutzerkennwort (und Benutzernamen) im HYDrOUS Portal anmelden. Dort kann er Dokumente und Informationen zu Signaturvorgängen einsehen, an denen der Benutzer beteiligt ist. Sofern für das Benutzerkonto bereits ein biometrisches Template erzeugt wurde, kann der Angreifer jedoch nicht auf den

privaten Schlüssel des Benutzers zugreifen, da hierfür eine zusätzliche biometrische Authentifizierung erforderlich ist.

Manipulation an der HYDrOUS Biometric Extension Könnte der Angreifer die Desktop Anwendung gegen eine manipulierte Version austauschen, die für ihn arbeitet, wären die übrigen Schutzmaßnahmen hinfällig. Eine so manipulierte Version könnte sowohl die biometrischen Unterschriften, die Benutzerkennworte und sogar die privaten Schlüssel kompromittieren. Als Gegenmaßnahme wird der Programmcode, welcher als Ausführbares *JAR Archiv* bereitgestellt wird, *digital signiert*. Jegliche Manipulation würde somit die Integrität des Archivs verletzen.

7.3.6 Kompromittierung der Server

Auch serverseitig sind Angriffe nicht auszuschließen. Prinzipiell sind Server sogar die „interessanteren“ Ziele für viele Angreifer, da dort Daten von vielen Benutzern gespeichert werden und Manipulationen eine größere Reichweite haben.

Kompromittierung der Benutzerdatenbank Kann ein Angreifer auf die Inhalte der Benutzerdatenbank zugreifen, kann er prinzipiell alle hinterlegten Informationen kopieren oder manipulieren. Aus Effizienzgründen können nicht alle Benutzerdaten durch einen solchen Angriff geschützt werden. Im Fall der Benutzerdatenbank sind dies alle Informationen außer den Benutzerkennworten. Kennworte sollten nur in einer abgeleiteten Version und niemals als Klartext gespeichert werden. In der Benutzerdatenbank von HYDrOUS werden Benutzerkennworte stets mit der Funktion *PBKDF2-HMAC-SHA256* abgeleitet. Für die Ableitung jedes Kennwortes wird zur Steigerung der Entropie ein eigener *salt* und eine hohe Anzahl (10000) von Wiederholungen für die unterliegende Hashfunktion SHA256 gewählt. Ein möglicher Angriff auf ein einziges Benutzerkennwort ist in diesem Fall so zeitaufwändig, dass der Angriff erkannt werden kann und die Benutzer ihre Benutzerkennworte als Reaktion abändern können.

Kompromittierung der Datenbank für biometrische Templates Die auf dem Server hinterlegten biometrischen Templates werden zwar durch die Benutzerkennworte geschützt, doch werden diese nicht im Klartext als Schlüssel genutzt. Stattdessen wird wie bei der Speicherung in der Benutzerdatenbank die Funktion *PBKDF2-HMAC-SHA256* mit *salt* und einer hohen Anzahl (10000) von Wiederholungen für die unterliegende Hashfunktion SHA256 eingesetzt. Die *salt* Werte von Benutzerdatenbank und der Datenbank für biometrische Templates dürfen hierbei nicht übereinstimmen.

7 *Evaluation*

8 Zusammenfassung

Im *zweiten Kapitel* dieser Masterarbeit wurden die Grundlagen für ein elektronisches Signatursystem beschrieben. Einerseits bestehen diese aus den kryptografischen Grundbausteinen aus denen ein solches System aufgebaut werden soll. Andererseits wurden die rechtlichen Rahmenbedingungen und Begriffen, die das Signaturgesetz (SigG) und die Signaturverordnung (SigV) definieren, dargelegt. Um diese Rahmenbedingungen und die beschriebenen kryptografischen Grundziele auch technisch umsetzen zu können, wurden die vom BSI empfohlenen Signaturverfahren und Algorithmen in die Grundlagen aufgenommen. Zusätzlich zu diesen allgemeinen Grundlagen wurde das Thema Gesundheitstelematik in Deutschland betrachtet und welche technischen Lösungen (EGK, HBA) theoretisch vorhanden sind. Da es in der Gesundheitstelematik nur in Ausnahmefällen vom (allgemeinen) Gesetz abweichende Rahmenbedingungen für elektronische Signaturen gibt, existieren die sogenannten „Braunschweiger Regeln zur Archivierung mit elektronischen Signaturen im Gesundheitswesen“, die als Empfehlung für die Umsetzung von Signatursystemen in der Gesundheitstelematik verstanden werden können. Abschließend wird die Telekonferenzlösung HEALTHTELKON eingeführt, welche der Ausgangspunkt für Dokumente ist, die mit HYDrOUS signiert werden sollen.

In *Kapitel 3* wurden verwandte wissenschaftliche Arbeiten beschrieben, die zur Lösung der Problemstellung hilfreich waren und Denkanstöße für das Konzept geliefert haben. Eine Lösungsanforderung des Mehrparteien-Signatursystems ist der Einsatz von Unterschriftenpads, mit denen handschriftliche Unterschriften digitalisiert werden können. Bei solchen digitalisierten Unterschriften handelt es sich um biometrische Merkmale. Dazu wurde in diesem Kapitel die Biometrie und durch sie entstehende Hürden beschrieben, die überwunden werden müssen. Für die biometrischen Unterschriften wird dann das *Dynamic Time Warping (DTW)* erklärt, welches in der Literatur zum Abgleich dieses biometrischen Merkmals am häufigsten Anwendung findet. Neben der Biometrie wurden weitere Themen behandelt, die für die Umsetzung von Mehrparteien-Signaturen hilfreich waren. Zunächst wurde *Shamir's Secret Sharing* erklärt, das es ermöglicht ein gemeinsames Geheimnis auf mehrere Personen aufzuteilen. Danach wurde das Konzept von Mehrparteien-Signaturen erklärt und exemplarisch ein bestehendes Protokoll skizziert.

8 Zusammenfassung

Das *vierte Kapitel* stellt die Anforderungsanalyse an das Mehrparteien-Signatursystem dar, welches in dieser Abschlussarbeit konzipiert werden soll. Einführend wird eine etwas technischere Problemstellung gegeben, dann die funktionale Anforderungen und die nicht-funktionalen Anforderungen festgelegt. Aus diesen Anforderungen wurden dann die Anwendungsfälle für das zu entwerfende System konstruiert.

Nachdem die Anforderungen festgelegt wurden, wird in *Kapitel 5* der Entwurf für die Systemarchitektur des Prototypen durchgeführt. Zunächst wurden dazu grundsätzliche Entscheidungen für und wider einer Desktopanwendung / Webanwendung getroffen und dann *RESTful Webservices* beschrieben. Nach diesen grundsätzlichen Architekturentscheidungen wurden die Komponenten des Systems auf konzeptioneller Ebene entworfen. Für die Umsetzung des Mehrparteien-Signatursystems wurde nach der Literaturrecherche in *Kapitel 3* die Entscheidung für ein eigenes Mehrparteien-Signaturprotokoll gefällt. Das entworfene Protokoll arbeitet mit einer Trusted Third Party (TTP), kann asynchron für alle Parteien arbeiten und reduziert den Austausch von Nachrichten auf ein Minimum. Für die Integration von biometrischen Unterschriften wurde ein Protokoll für die biometrische Schlüsselfreigabe entworfen. Durch dieses Protokoll können klassische Verfahren, die auf asymmetrischer Verschlüsselung basieren, für das Mehrparteien-Signaturprotokoll eingesetzt werden und trotzdem die Vorteile biometrischer Merkmale einfließen.

Die prototypische Implementierung des entworfenen Systems wurde in *Kapitel 6* beschrieben. Hierzu wurden die technischen Hilfsmittel erwähnt und interessante Details der Implementierung näher betrachtet.

In *Kapitel 7* folgt letztendlich die Evaluation des entworfenen Systems. Zunächst wurde in kleinem Rahmen die Leistungsfähigkeit der biometrischen Abgleiche evaluiert. Danach wurde die Erfüllung der Anforderungsanalyse betrachtet. Auf die Sicherheitsanforderungen wurde danach detaillierter eingegangen und die Gegenmaßnahmen für potenzielle Angriffe erläutert.

9 Ausblick

Im Ausblick sollen fortführende Arbeiten und Ideen erwähnt werden, welche über die Zielsetzung dieser Masterarbeit hinausgehen.

Wie bereits in den Grundlagen erläutert, gibt es laut dem deutschen Gesetzgeber verschiedene Formen der elektronischen Signaturen. In dieser Masterarbeit wurde ein System entworfen, welches *fortgeschrittene elektronische Signaturen* umsetzt. Für eine vor dem Gesetz weitreichendere Beweiskraft, wäre der Einsatz von *qualifizierten elektronischen Signaturen* erforderlich. Der essenzielle Unterschied liegt nun darin, dass für eine *qualifizierte elektronische Signatur* ein *qualifiziertes Zertifikat* vorausgesetzt wird. Hierzu müsste die ausstellende Organisation akkreditiert werden, wodurch Kosten entstehen. Im Gesundheitswesen bestehen solche qualifizierten elektronischen Signaturen, wenn ein behandelnder Arzt einen *elektronischer Heilberufsausweis (HBA)* besitzt. Eine fortführende Arbeit könnte demnach die Integration des HBA in das hier entworfene System darstellen. Denkbare Lösungen wären die elektronische Signatur mittels des Schlüsselpaares auf der SmartCard (HBA) oder der Einsatz von neuen Schlüsselpaaren, welche durch die Schlüssel auf einem HBA elektronisch zertifiziert werden.

Bei der Bearbeitung und Evaluation des biometrischen Abgleichs von zwei Unterschriften (oder gegenüber dem biometrischen Template) wurden einige Schwächen des Ansatzes festgestellt. Biometrische Unterschriften bieten ein sehr universelles Merkmal in der Domäne von Ärzten und können nicht versehentlich (vgl. Fingerabdruck an Glas) abgegeben werden. Die natürlichen Schwankungen und auch die Gewöhnung an die Unterschriftenpads als Schreibunterlage, bringen jedoch bei manchen Probanden stark unterschiedliche biometrische Unterschriften mit sich. Abgleiche liefern so nicht immer die erwarteten Resultate. Eine Lösung hierfür wäre eine regelmäßige Aktualisierung der hinterlegten biometrischen Templates. Unter anderem wegen dieser Schwäche wurde die *Zwei-Faktor-Authentifizierung* mit Benutzerkennwort und biometrischer Unterschrift gewählt. Da in der Literatur zu biometrischen Merkmalen auch vereinzelt „lernende“ Verfahren zum Einsatz kommen, wäre eine zusätzliche Evaluation dieser Ansätze sinnvoll, um diese natürlichen Schwankungen zu „verstehen“.

Der entworfene Prototyp bietet nur die Grundanforderungen für ein Mehrparteien-Signatursystem und ist noch nicht in allen Bereichen auf Anwender angepasst. Im

9 Ausblick

Wesentlichen sind dies Komfortmerkmale, die nicht in der Implementierung vorgesehen wurden. Zum Beispiel ist die Kopplung der Benutzerkonten zwischen HEALTHTELKON und HYDrOUS wenig praktisch, da die Änderung des Passwortes in HEALTHTELKON faktisch auch einer Wiederverschlüsselung des entsprechenden biometrischen Templates bedarf. Im Prototypen wird dieses Problem umgangen, indem zwei Benutzerkennworte erstellt werden. Eine Lösung wäre der Einsatz von produktübergreifenden Authentifizierungsmechanismen, auch Single Sign-On genannt. Ein weiteres Komfortmerkmal sind die *Komfortsignatur* und die *Stapelsignatur* (Unterabschnitt 2.2.5), die als Funktionalität in der HYDrOUS Biometric Extension implementiert werden könnten.

10 Abbildungsverzeichnis

2.1	Beispiel GPG Verfahren für E-Mail	10
a	Erstellen einer signierten E-Mail	10
b	Prüfen einer signierten E-Mail	10
2.2	Muster der SmartCards in der Gesundheitstelematik	14
a	<i>Elektronische Gesundheitskarte</i> [Wik12]	14
b	<i>Heilberufsausweis</i> [Bun08]	14
2.3	Übersicht der Signaturverfahren [BSI07a, BSI07b]	15
2.4	Einstufung der verwendeten Signaturverfahren bezüglich des Beweiswertes [SKB ⁺ 10, S.18]	19
3.1	Authentifizierungsablauf	25
a	Authentifizierung mit Passwortvergleich	25
b	Authentifizierung mit Übereinstimmung	25
3.2	Zusammenhang der Leistungsmetriken	27
3.3	<i>Dynamic Time Warping</i> [SC07]	30
a	Matrix mit minimalem Verzerrungspfad	30
b	Verzerrung zweier Zeitreihen	30
3.4	Kostenmatrizen der reduzierten Zeitreihen (grob → original) [SC07]	31
4.1	Use Cases des Gesamtsystems	41
5.1	Webanwendung	50
5.2	Desktopanwendung	51
5.3	Hybridlösung	51
5.4	Komponentendiagramm von HYDrOUS	55
5.5	Screenshot der Dokumentenübersicht eines Benutzers	58
5.6	Screenshot der Dokumentendetailansicht	59
6.1	Unterschriftenpad WACOM STU-500 (Quelle: Wacom)	76
6.2	UML Klassendiagramm des Pakets <code>*.hydrous.model</code>	78
6.3	UML Klassendiagramm des Pakets <code>*.hydrous.controller</code>	79
6.4	Screenshot der HYDrOUS Biometric Signature Extension	83

10 Abbildungsverzeichnis

7.1	Screenshot „SignaturePlotPanel”	86
7.2	Replay Angriff und Gegenmaßnahme	89
a	Replay-Angriff	89
b	Vermeidung mit <i>nonce</i>	89
7.3	<i>Diffie-Hellman-Schlüsselaustausch</i> ohne Nachrichtenauthentifizierung .	91
a	Normaler Ablauf	91
b	Man-in-the-Middle Angriff	91

11 Quelltextverzeichnis

5.1	Repräsentation in JSON (118 Byte)	53
5.2	Repräsentation in XML (136 Byte)	53
5.3	Verknüpfung in JSON	54
5.4	Expandierte Verknüpfung in JSON	54
5.5	Rohdaten aus dem Signaturtablet als XML Dokument	70
6.1	JSON Repräsentation von <code>Document</code>	81
6.2	JSON Repräsentation von <code>Signee</code>	81
6.3	GET Anfrage an den Biometric Matcher für <code>nonce_M</code>	82
6.4	GET Anfrage an den Biometric Matcher	82
6.5	JSON Parameter: Kennwort aktualisieren	82
6.6	JSON Parameter: Biometrisches Template aktualisieren	83
6.7	LLSDK manuelles Speichern der Zeit	84
6.8	LLSDK manuelles löschen der Messpunkte	84

Quelltextverzeichnis

12 Tabellenverzeichnis

2.1	Kryptografische Grundziele	12
3.1	Vergleichskriterien biometrischer Merkmale	24
3.2	Vergleich biometrischer Merkmale [UPPJ04, S.949]	24
3.3	Lokale Merkmale (Auswahl)	28
3.4	Globale Merkmale (Auswahl)	28
4.1	Benutzerkonto anlegen	42
4.2	Anmeldung im System	42
4.3	Unterschrift Enrollment	43
4.4	Biometrische Schlüsselfreigabe	43
4.5	Schlüsselpaar erzeugen	44
4.6	Biometrische Unterschrift ändern	44
4.7	Benutzerkennwort ändern	45
4.8	Dokument in das System laden	45
4.9	Dokument lesen	46
4.10	Dokument signieren	46
4.11	Signatur von Dokument prüfen	47
5.1	CRUD Operationen und die entsprechenden REST Operationen	53
5.2	Ausgewählte Merkmale für den Abgleich von biometrischen Unterschriften	69
5.3	Vergleich von Original/Fälschung und zwei originalen Unterschriften	71
6.1	Systemkonfiguration der Testsysteme	73
7.1	Evaluation der biometrischen Merkmale (Distanzen)	86

12 Tabellenverzeichnis

13 Literaturverzeichnis

- [Bis09] BISKUP, JOACHIM: *Security in Computing Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
- [BJ10] BAGHERZANDI, ALI und STANISLAW JARECKI: *Identity-Based Aggregate and Multi-Signature Schemes Based on RSA*. Public Key Cryptography–PKC 2010, Seiten 480–498, 2010.
- [BN06] BELLARE, MIHIR und GREGORY NEVEN: *Multi-signatures in the plain public-Key model and a general forking lemma*. In: *Proceedings of the 13th ACM conference on Computer and communications security - CCS '06*, Seite 390, New York, New York, USA, 2006. ACM Press.
- [BSI07a] BSI TR-03114: *Stapelsignatur mit dem Heilberufsausweis*. Technischer Bericht, Bundesamt für Sicherheit in der Informationstechnik, 2007.
- [BSI07b] BSI TR-03115: *Komforsignatur mit dem Heilberufsausweis*. Technischer Bericht, Bundesamt für Sicherheit in der Informationstechnik, 2007.
- [BSI13] BSI TR-02102: *Kryptographische Verfahren: Empfehlungen und Schlüssellängen*. Technischer Bericht, Bundesamt für Sicherheit in der Informationstechnik, 2013.
- [Bun08] BUNDESÄRZTKAMMER: *Heilberufsausweis*. <http://www.bundesaerztekammer.de/page.asp?his=1.134.3416>, Oktober 2008.
- [Bun13] BUNDESNETZAGENTUR: *Bekanntmachung zur elektronischen Signatur nach dem Signaturgesetz und der Signaturverordnung (Übersicht über geeignete Algorithmen)*. Bundesanzeiger, BAnz AT 27.03.2013 B4, 2013.
- [BwW00] BAUM-WAIDNER, BIRGIT und MICHAEL WAIDNER: *Optimistic Multi-party Contract Signing*. Seiten 524–535, 2000.
- [GM08] GÜLER, İNAN und MAJID MEGHDADI: *A different approach to off-line handwritten signature verification using the optimal dynamic time warping algorithm*. Digital Signal Processing, 18(6):940–950, November 2008.

- [HO13] HOUTA, SALIMA und LUTZ OETTERSCHAGEN: *Telekonferenzsystem auf Basis der Elektronischen FallAkte*. e-Health 2013, 2013.
- [ISO07] *Information technology – Biometric data interchange formats – Part 7: Signature / sign time series data*, 2007.
- [KMA09] KHALIL, MOSTAFA I., MOHAMED MOUSTAFA und HAZEM M. ABBAS: *Enhanced DTW based on-line signature verification*. 2009 16th IEEE International Conference on Image Processing (ICIP), Seiten 2713–2716, November 2009.
- [KR12] KORDY, BARBARA und SAA RADOMIROVIC: *Constructing Optimistic Multi-party Contract Signing Protocols*. 2012 IEEE 25th Computer Security Foundations Symposium, Seiten 215–229, Juni 2012.
- [KY05] KHOLMATOV, ALISHER und BERRIN YANIKOGLU: *Identity authentication using improved online signature verification method*. Pattern Recognition Letters, 26(15):2400–2408, November 2005.
- [MCN09] MAIORANA, EMANUELE, PATRIZIO CAMPISI und ALESSANDRO NERI: *Template protection for Dynamic Time Warping based biometric signature authentication*. In: *2009 16th International Conference on Digital Signal Processing*, Seiten 1–6. IEEE, Juli 2009.
- [MR07] MUKHAMEDOV, AYBEK und MARK RYAN: *Improved multi-party contract signing*. Financial Cryptography and Data Security, 4886:179–191, 2007.
- [RCB01] RATHA, N. K., J. H. CONNELL und R. M. BOLLE: *Enhancing security and privacy in biometrics-based authentication systems*. IBM Systems Journal, 40(3):614–634, März 2001.
- [SC07] SALVADOR, STAN und PHILIP CHAN: *Toward accurate dynamic time warping in linear time and space*. Intelligent Data Analysis, 11(5):561–580, Oktober 2007.
- [Sha79] SHAMIR, ADI: *How to share a secret*. Communications of the ACM, 22(11):612–613, November 1979.
- [SKB⁺10] SEIDEL, C., H. KOSTOCK, A. BRANDNER, J. BALFANZ und P. SCHMÜCKER: *Empfehlungen für den Einsatz elektronischer Signaturen und Zeitstempel in Versorgungseinrichtungen des Gesundheitswesens*. Competence Center für die Elektronische Signatur im Gesundheitswesen e.V. CCESigG, 2010.

- [Til11] TILKOV, STEFAN: *REST und HTTP*. Dpunkt.Verlag GmbH, 2011.
- [UPPJ04] ULUDAG, UMUT, SHARATH PANKANTI, S. PRABHAKAR und A.K. JAIN: *Biometric cryptosystems: issues and challenges*. Proceedings of the IEEE, 92(6):948–960, Juni 2004.
- [Vin68] VINTSYUK, TK: *Speech discrimination by dynamic programming*. Cybernetics and Systems Analysis, 4(1):52–57, 1968.
- [Wik12] WIKIMEDIA: *Elektronische Gesundheitskarte*. https://commons.wikimedia.org/wiki/File:Elektronische_Gesundheitskarte_Mustermann_VS.svg, Juli 2012.
- [WMD⁺13] WANG, XIAOYUE, ABDULLAH MUEEN, HUI DING, GOCE TRAJCEVSKI, PETER SCHEUERMANN und EAMONN KEOGH: *Experimental comparison of representation methods and distance measures for time series data*. Data Min. Knowl. Discov., 26(2):275–309, März 2013.