Fakultät für Mathematik und Informatik

Hausarbeit

im Kurs 01600 Wissenschaftliches Arbeiten
im Studiengang Master of Science Informatik

zum Thema

Infrastruktur für public key Kryptografie: von Key Directories über Identity-based encryption zu Zertifikaten und vertrauensbasierten Modellen

Jan Kristof Nidzwetzki
Berlin 23. Oktober 2013

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1 Grundlagen

Geheime Informationen sollen geheim bleiben. Um diese Forderung in der Praxis zu erfüllen, müssen einige Anstrengungen unternommen werden. Seit mehr als 2500 Jahren werden Techniken eingesetzt, Informationen nur für bestimmte Personen zugänglich zu machen [Beu07, S. 3f]. Schon *Gaius Caesar* (100 v. Chr. - 44 v. Chr.) nutze, nach Überlieferung des römischen Schriftstellers *Sueton*, eine einfache Verschlüsselung um Nachrichten vertraulich zu überbringen [Sue02].

Mit Hilfe der Verschlüsselung kann ein Text so umgewandelt werden, dass dieser danach wie zufällig aneinandergereihte Buchstaben aussieht (*Verschlüsselung*). Beim verschlüsseln werden ein bestimmter *Verschlüsselungsalgorithmus*, sowie ein *Schlüssel* (*Key*) verwendet. Kennt jemand den eingesetzten *Verschlüsselungsalgorithmus* und den verwendeten *Schlüssel*, so kann er die Nachricht wieder in den ursprünglichen Text umwandeln (*Entschlüsselung*). Die Verschlüsselung sorgt somit für die *Vertraulichkeit* einer Nachricht.

Durch die Verschlüsselung können geheime Nachrichten gefahrlos über unsichere Kanäle, wie z. B. dem Internet, übertragen werden. Auch wenn die Nachrichten abgefangen und mitgelesen werden, bleiben die übertragenen Informationen geheim. Um verschlüsselt zu kommunizieren, müssen sich Absender und Empfänger auf den verwendeten *Verschlüsselungsalgorithmus* sowie einen *Schlüssel* einigen. Der verwendete Schlüssel muss sicher zwischen den Teilnehmern ausgetauscht werden. Erlangen weitere Personen Kenntnis von dem *Schlüssel*, so können diese ebenfalls die übertragenen Daten entschlüsseln.

Das ursprüngliche Problem, beliebige Daten vertraulich über ein unsicheres Medium zu übertragen, wurde auf ein anderes Problem reduziert: dem sicheren Übertragen eines Schlüssels. Lösungen für dieses Problem werden in den folgenden Abschnitten vorgestellt.

Weiterer Aufbau der Arbeit

Die nächsten Kapitel der vorliegenden Arbeit sind wie folgt gegliedert:

Abschnitt 2 - Public Key Kryptografie: Dieser Abschnitt beschäftigt sich mit den Grundlagen der *Public Key Kryptografie* und von *Zertifikaten*. Zudem werden Vereinbarungen über die weitere Notation festgelegt.

Abschnitt 3 - Public Key Infrastucture: In diesem Abschnitt werden die Komponenten einer *Public Key Infrastucture* (*PKI*) vorgestellt. Auch fortgeschrittene Konzepte wie das Zurückziehen von Zertifikaten werden dort besprochen.

Abschnitt 4 - Das Web of trust: Als Alternative zu der *PKI* hat sich das Konzept des *Web of trust* verbreitet. In diesem wird auf eine zentrale Infrastruktur verzichtet und Anwender sprechen sich gegenseitig vertrauen aus.

Abschnitt 5 - Identity-based encryption: Mit der *Identity-based encryption* bezeichnet man eine Technik, den *Schlüssel* für die *Verschlüsselung* aus öffentlich bekannten Informationen zu berechnen. Auf den Austausch eines Schlüssels kann so verzichtet werden.

Abschnitt 6 - Fazit: Der letzte Abschnitt dieser Arbeit fasst die bisherigen Kapitel zusammen. Zudem wird ein Ausblick auf neuere Entwicklungen gegeben.

2 Public Key Kryptografie

In den 70er Jahren des vergangen Jahrhunderts wurden die ersten Algorithmen publiziert, welche das sichere Übertragen des Schlüssels überflüssig machen [RSA78] [DH76]. Im Gegensatz zu den bislang vorgestellten *symmetrischen Verschlüsselungsverfahren*, benutzen diese *asymmetrischen Verschlüsselungsverfahren* zwei verschiedene Schlüssel.

Alle *asymmetrischen Verschlüsselungsverfahren* haben zwei grundlegende Eigenschaften: (1) wird ein Text mit dem einen Schlüssel verschlüsselt, so kann Sie nur mit dem andern Schlüssel entschlüsselt werden. (2) Es ist nicht möglich, aus einem Schlüssel den anderen Schlüssel zu berechnen.

Es kann somit einer der beiden Schlüssel zum Verschlüsseln der Nachrichten (*public Key* / K^+) veröffentlicht werden. Der andere Schlüssel (*private Key* / K^-) wird nicht veröffentlicht. Der Ablauf der asymmetrischen Verschlüsselung ist in Abbildung 1 dargestellt.

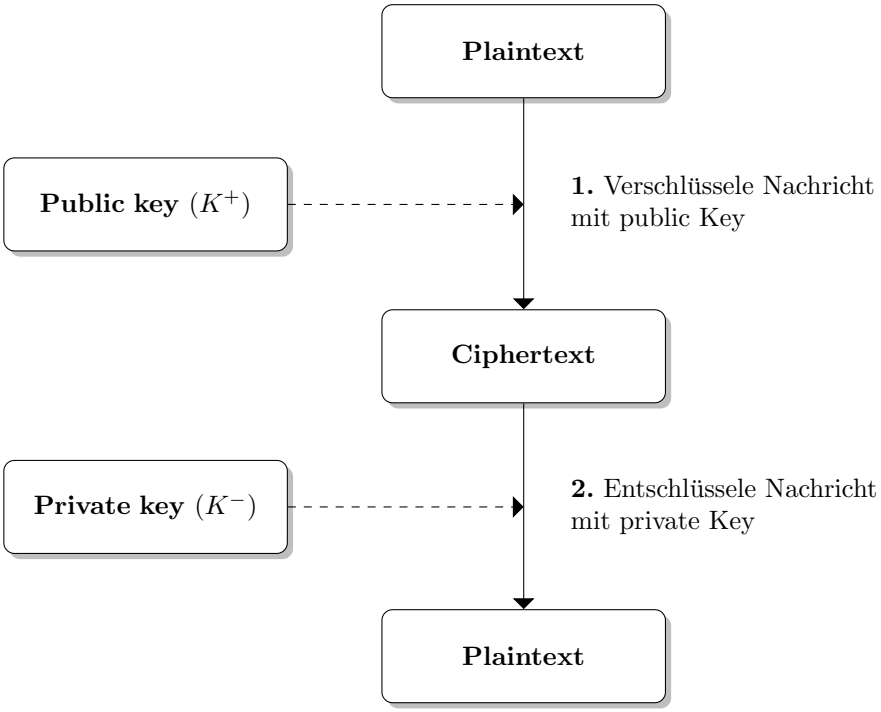

Abbildung 1: Um die Vertraulichkeit einer Nachricht zu gewährleisten, wird diese mit dem *public Key* des Empfängers verschlüsselt. Nur mit dem *private Key* des Empfängers kann die Nachricht wieder entschlüsselt werden.

2.1 Notation

Die Kommunikationspartner in Arbeiten zu Verschlüsselungstechniken werden häufig mit sprechenden Namen wie *Alice* oder *Bob* bezeichnet. Diese Namen gehen auf die Arbeit *A Method for Obtaining Digital Signatures and Public key Cryptosystems* [RSA78] zurück und wurden seitdem in vielen Texten benutzt. In dem Buch *Applied cryptography* [Sch95] hat der Autor *Bruce Schnei-*

der noch weitere Namen eingeführt. Dieser Notation folgt die vorliegende Arbeit. In Tabelle 1 sind die verwendeten Namen und deren Bedeutung aufgeführt.

Name	**Bedeutung**
Alice	erster Kommunikationspartner
Bob	zweiter Kommunikationspartner
Charlie	dritter Kommunikationspartner
Dave	vierter Kommunikationspartner
Mallory	Angreifer, welcher die Möglichkeit hat, die Kommunikation zwischen *Alice* und *Bob* zu belauschen und zu verändern (*malicious attacker*)

Tabelle 1: Verwendete Namen der Kommunikationspartner

In den folgenden Abschnitten wird mit der Kurzform *Schlüssel* oder *Key* der *öffentliche Schlüssel* eines Benutzers bezeichnet.

2.2 Man-in-the-middle Angriffe

Die *Public Key Kryptografie* löst das Problem, wie ein Schlüssel über einen unsicheren Kommunikationskanal ausgetauscht werden kann [Mer78]. Möchten *Alice* und *Bob* miteinander kommunizieren, kennen jedoch den öffentlichen Schlüssel des anderen nicht, so können sie diesen über den unsicheren Kommunikationskanal austauschen. Nach dem Austausch der Schlüssel erfolgt die weitere Kommunikation verschlüsselt. Konnte jemand die ausgetauschten Schlüssel mitlesen, so kann dieser aufgrund der Eigenschaften der *asymmetrischen Verschlüsselung* die Kommunikation trotzdem nicht entschlüsseln und mitlesen.

Hat ein Angreifer (*Mallory*) zusätzlich die Möglichkeit, die Kommunikation zwischen *Alice* und *Bob* zu verändern, so kann er beim Schlüsselaustausch die versendeten Schlüssel gegen eigene Schlüssel austauschen. Dies ermöglicht es *Mallory*, die Kommunikation mitzulesen. Dieser Angriff wird als *Man-in-the-middle Angriff* bezeichnet [FS03].

In Abbildung 2 ist der Angriff dargestellt. *Alice* möchte *Bob* eine verschlüsselte Nachricht zukommen lassen. *Alice* liegt jedoch der öffentliche Schlüssel von *Bob* nicht vor. Daher fragt *Alice Bob* nach seinem öffentlichen Schlüssel (K^+_{Bob}). Die Antwort von *Bob* wird von *Mallory* abgefangen. Dieser tauscht den Schlüssel von *Bob* gegen seinen eigenen öffentlichen Schlüssel ($K^+_{Mallory}$) aus und sendet die Antwort an *Alice* weiter. *Alice* verschlüsselt ihre Nachricht mit dem vermeintlichen Schlüssel von Bob ($K^+_{Mallory}$) und versendet diese. Die Nachricht wird wiederum von *Mallory* abgefangen und mit seinem privaten Schlüssel ($K^-_{Mallory}$) entschlüsselt. *Mallory* kann die Nachricht nun Lesen und auch verändern. Im letzten Schritt verschlüsselt *Mallory* die Nachricht mit dem wirklichen *public Key* von *Bob* (K^+_{Bob}) und sendet diese an *Bob* weiter. *Bob* merkt nicht, dass die Nachricht abgefangen, gelesen und möglicherweise auch verändert worden ist.

Ein solcher *Man-in-the-middle Angriff* ist nur durchführbar, wenn nicht festgestellt werden kann, ob ein öffentlicher Schlüssel einer bestimmten Person gehört. Könnte in Abbildung 2 *Alice* dies überprüfen, würde Sie feststellen, dass sie nicht den zu *Bob* gehörenden *public Key* erhalten hat.

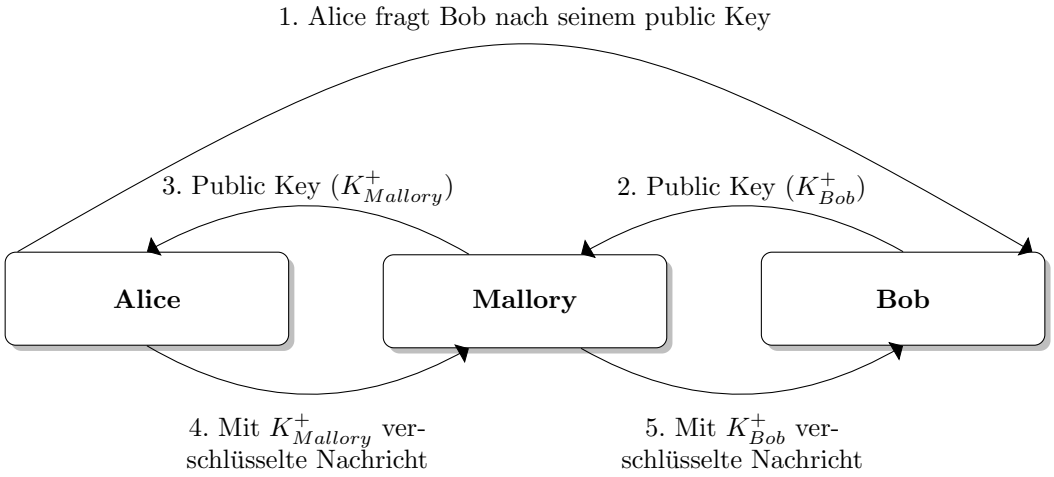

Abbildung 2: *Man-in-the-Middle Angriff*: (1) *Alice* fragt *Bob* nach seinem *public Key*. (2) *Bob* Antwortet mit seinem *public Key*. (3) Die Nachricht wird von *Mallory* abgefangen. Dieser ersetzt den *public Key* von *Bob* gegen seinen eigenen. (4) *Alice* sendet *Bob* eine Nachricht und verschlüsselt diese mit dem erhaltenen Schlüssel $K^+_{Mallory}$. (5) *Mallory* entschlüsselt die Nachricht, verändert diese ggf. und verschlüsselt die Nachricht mit dem echten Schlüssel von *Bob* K^+_{Bob}.

2.3 Digitale Signaturen

Bislang wurde die *asymmetrischen Verschlüsselung* zur Wahrung der *Vertraulichkeit* eingesetzt. Die *asymmetrischen Verschlüsselungsverfahren* haben noch einen weiteren Einsatzbereich: Mit ihnen können Dokumente digital Signiert werden [Lam79], [Mer89] [DH76]. Eine *digitale Signatur* stellt die *Integrität* des signierten Dokuments sicher. Nach dem Signieren eines Dokuments kann dieses nicht mehr unbemerkt verändert werden.

Es besteht die Möglichkeit entweder das gesamte Dokument zu signieren oder eine zum Dokument gehörende *Prüfsumme* (*Hashwert*). Die zu signierenden Daten werden mit einem *privaten Schlüssel* verschlüsselt. Das Ergebnis dieser Berechnung wird im Folgenden als *digitale Signatur* (kurz *Signatur*) bezeichnet. Die *Signatur* wird in der Regel zusammen mit dem Dokument gespeichert oder verschickt. Wird das Dokument verändert, passt die berechnete Signatur nicht zu dem veränderten Dokument. Es kann somit festgestellt werden, ob ein Dokument nach dem Signieren verändert worden ist.

Im folgenden Absatz wird ein Vorgehen gezeigt, wie eine Signatur erstellt und überprüft werden kann.

- Erstellen einer Digitalen Signatur:
  - Für das zu signierende Dokument x wird ein Hashwert $h(x)$ berechnet.
  - Dieser Hashwert wird mit dem *privaten Schlüssel* des Anwenders verschlüsselt.
  - Die Signatur wird zusammen mit dem Dokument verteilt.
- Überprüfen einer Digitalen Signatur:

- Berechnen des Hashwerts $h'(x)$ des zu prüfenden Dokuments.
- Entschlüsseln der *digitalen Signatur* mittels des korrespondierenden öffentlichen Schlüssels.
- Vergleich der beiden Hashwerte $h(x)$ und $h'(x)$. Sind diese identisch wurde das Dokument seit dem Signieren nicht verändert.

Um mit *digitalen Signaturen* zu arbeiten, sind zwei Probleme zu lösen: (1) Es müssen die öffentlichen Schlüssel zum entschlüsseln der *digitalen Signatur* verfügbar sein. (2) Es muss sichergestellt werden, dass *öffentliche Schlüssel* einer bestimmten Person zugeordnet werden können. Andernfalls könnte ein Angreifer das Dokument verändern und einfach mit seinem Schlüssel neu signieren. Es muss daher geprüft werden, mit welchem Schlüssel eine Signatur erstellt worden ist.

2.4 Zertifikate

Verschlüsselung und *digitale Signaturen* setzen voraus, dass *öffentlichen Schlüsseln* ihr Besitzer zugeordnet werden kann. Anderenfalls ist der in Abschnitt 2 vorgestellte *Man-in-the-middle Angriff* möglich. Um die Beziehung zwischen einer Person und einem Schlüssel zu dokumentieren, werden *Zertifikate* [Can03] [Sta06] eingesetzt. Im Jahr 1978 wurde das Konzept der *Zertifikate* erstmals von *Loren Kohnfelder* [Koh78] in seiner Arbeit *Towards a Practical Public Key Cryptosystem* beschrieben.

Der Einsatz von *Zertifikaten* verlangt, dass man dem Aussteller des Zertifikates uneingeschränkt vertraut. Der Aussteller der Zertifikate wird als *Certificate Authority* (*CA*) bezeichnet. Die *CA* besitzt einen *privaten* (K^-_{CA}) und einen *öffentlichen Schlüssel* (K^+_{CA}). Die *Certificate Authority* ist dafür zuständig, zu belegen, dass ein *öffentlicher Schlüssel* zu einer bestimmten Person gehört. Um dies zu erreichen, muss je nach Arbeitsweise der *CA*, beispielsweise der Ausweis des Schlüsselinhabers zusammen mit seinem Schlüssel der *CA* vorgelegt werden. Die *CA* stellt daraufhin ein Zertifikat aus, welches belegt (zertifiziert), dass der vorgelegte Schlüssel der Person gehört. Dieses Zertifikat wird von der *CA* mit ihrem *privaten Schlüssel* (K^-_{CA}) signiert [Koh78, S. 40]. Mit Hilfe des *öffentlichen Schlüssels* (K^+_{CA}) kann die Gültigkeit des Zertifikates überprüft werden. Vertraut man der *CA*, so kann man mit Hilfe des Zertifikates feststellen, ob ein Schlüssel zu einer Person gehört.

Um *Zertifikate* prüfen zu können, muss der *public Key* der *CA* bekannt sein. Dieser *public Key* muss auf einem sicheren Weg von der *CA* zu den Nutzern der *CA* gelangen. Es muss sichergestellt sein, dass dieser Schlüssel beim Transport nicht verändert wurde [Sta06, S. 422]

Werden *Zertifikate* eingesetzt, ist der in Abschnitt 2 beschriebene *Man-in-the-middle Angriff* nicht mehr möglich. Erhält *Alice* einen neuen *public Key* von *Bob* so kann *Alice* prüfen ob ein gültiges Zertifikat zu diesem *public Key* existiert und ob *Bob* in diesem Zertifikat als Besitzer des Schlüssels ausgewiesen wird.

Das Problem, die Identität des Kommunikationspartners prüfen zu müssen, wurde darauf reduziert, der Arbeitsweise der *CA* zu vertrauen.

3 Public Key Infrastructure (PKI)

Auf den letzten Seiten wurden einige Probleme im Umgang mit *öffentlichen Schlüsseln* aufgezeigt. Es wurde das Problem der Verteilung von Schlüsseln angesprochen. Ebenso wurde auf das Problem der Zuordnung zwischen Schlüssel und seinem Inhaber eingegangen. Diese Probleme können durch Verwendung einer *Public key infrastructure* (*PKI*) gelöst werden.

3.1 PKI nach X.509

Das Komitee *ITU Telecommunication Standardization Sector* (ITU-T) hat im Jahr 1988 den Standard *X.509* verabschiedet [ITU88]. In diesem Standard wird unter anderem beschrieben, wie eine *PKI* aufgebaut ist und welches Format Zertifikate aufweisen. In der *RFC 2459 - Internet X.509 Public Key Infrastructure Certificate and CRL Profile* [HFPS99] finden sich weitere gebräuchliche Standards.

Eine *Public key infrastructure* nach dem Standard X.509 wird häufig kurz als *PKIX* bezeichnet. Eine weit verbreitete Definition des Begriffs *PKIX* stammt aus der *RFC 2828* [Shi00, S. 136]. Dort wird der Begriff wie folgt definiert:

> „The set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography.“[1]

Der Standard X.509 wurde mehrfach aktualisiert und überarbeitet. Aktuell ist der Standard in der Version 3. Dieser wird in der *RFC 3280* beschrieben [HPFS02]. Bezieht man sich auf den Standard in dieser Version, so wird dies als *X.509v3* abgekürzt.

3.1.1 Zertifikate nach X.509v3

In dem Standard *X.509v3* wird der Aufbau von Zertifikaten festgelegt. *X.509v3* schreibt vor, dass neben dem Namen des Inhabers und seinem *öffentlichen Schlüssel* (Subject), weitere Informationen im Zertifikat hinterlegt werden müssen. Hierzu zählen unter anderem:

Version: Version des verwendeten X.509 Standards.

Serial number: Eine eindeutige, fortlaufende Nummer, welche das ausgestellte Zertifikat kennzeichnet.

Signature Algorithm ID: Algorithmus welcher zum unterschreiben verwendet wurde.

Issuer: Name des Zertikatausstellers.

Validity - notBefore / notAfter: Zeitraum, in welchem das Zertifikat gültig sein soll.

Zudem ist es möglich weitere Informationen in den *X.509v3 Zertifikaten* unterzubringen. So kann beispielsweise die Nutzung des Zertifikats auf bestimmte Aufgaben eingeschränkt werden (Key-Usage).

In der *RFC 3279* [BPH02] ist zusätzlich definiert, welche Verschlüsselungsalgorithmen verwendet werden können und in welcher Schreibweise (Bsp: 3-DES, 3DES, Tripple-DES) die Algorithmen genau bezeichnet werden.

[1] Auf Deutsch: Eine *Public key infrastructure* ist eine Ansammlung von Hardware, Software, Menschen, Vorschriften und Arbeitsabläufen um auf asymmetrischer Kryptographie basierende digitale Zertifikate zu erstellen, verwalten, abzuspeichern, verteilen und zurückzuziehen.

3.2 Bestandteile einer PKI nach X.509

Eine *PKI* besteht aus mehreren Komponenten. Diese sind in der Tabelle 2 aufgeführt. Die einzelnen Komponenten werden in den folgenden Abschnitten genauer beschrieben. Abbildung 3 zeigt das Zusammenspiel dieser Komponenten auf.

Deutscher Name	Englischer Name	Bedeutung
Zertifizierungsstelle	Certificate Authority	Organisation welche das CA-Zertifikat besitzt
Registrierungsstelle	Registration Authority	Organisation, bei der Zertifikate beantragt werden
Zertifikatsperrliste	Certificate Revocation List	Eine Liste welche die gesperrten (zurückgezogenen) Zertifikate enthält
Verzeichnisdienst	Certificate Repository	Ein Verzeichnis, in welchem alle ausgestellten Zertifikate enthalten sind
Dokumentationen	Documentation	Dokumentation über die Arbeitsweise der PKI

Tabelle 2: Komponenten einer *PKI* (nach [Sta06, S. 429])

3.2.1 Zertifizierungsstelle

Die *Zertifizierungsstelle* (*Certificate Authority - CA*) ist das Herzstück einer *PKI*. Sie besitzt ein Schlüsselpaar (den CA-Schlüssel) mit welchem Zertifikate unterschrieben werden. Zudem ist sie zuständig für das Führen einer Zertifikatsperrliste [Sta06, S. 429].

3.2.2 Registrierungsstelle

Die operativen Aufgaben einer Zertifizierungsstelle können an eine oder mehrere Registrierungsstellen (*Registration Authority - RA*) delegiert werden. Diese Registrierungsstellen sind dafür zuständig, eingehende Anträge zu prüfen und an die *Zertifizierungsstelle* weiterzuleiten [HFPS99, S. 7]. Da die *Zertifizierungsstelle* die vorgelegten Anträge nur noch unterschreibt, ist ein Vertrauen in die Arbeitsweise der *Registrierungsstellen* erforderlich.

3.2.3 Zertifikatsperrliste

In einer *Zertifikatsperrliste* (*Certificate Revocation List - CRL*) werden Zertifikate geführt, welche vor dem Ablauf ihrer Gültigkeit zurückgezogen worden sind. Um die Gültigkeit eines Zertifikates zu prüfen, muss neben der Korrektheit des Zertifikates (siehe Abschnitt 2.4 auf Seite 5) auch überprüft werden, ob das Zertifikat nicht zurückgezogen und somit für ungültig erklärt worden ist [HFPS99, S. 4ff] [Gut02, S. 2f].

Technisch wird dies u. a. durch das Protokoll *OCSP - Online Certificate Status Protocol* abgebildet. Das Protokoll ist in der dazugehörigen *RFC 2560* [MAM+99] spezifiziert. Viele aktuelle Webbrowser nutzen dieses Protokoll, um vor dem Aufbau einer verschlüsselten Verbindung zu einer Webseite das Zertifikat der Webseite zu überprüfen [Moz01]. Wurde das Zertifikat zurückgezogen, erscheint im Webbrowser eine Warnung, die den Benutzer über ein Problem mit dem Zertifikat informiert.

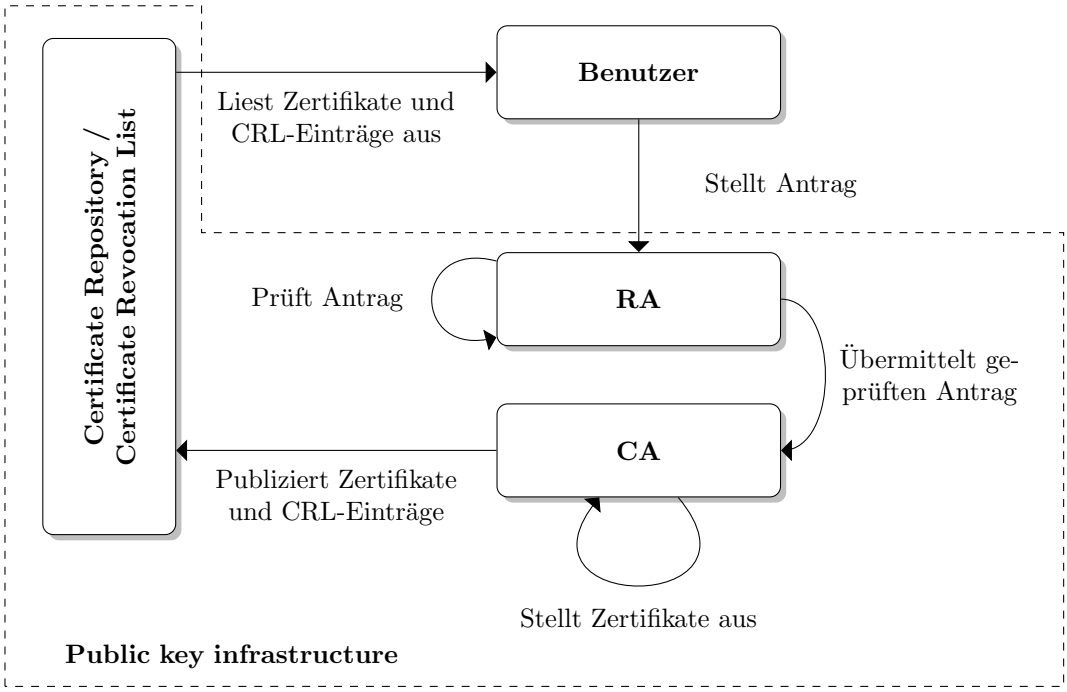

Abbildung 3: Schematischer Aufbau einer *PKI* (nach [HFPS99, S. 7])

Gründe für das zurückziehen eines Zertifikates können gemäß [HFPS99, S. 11] eine Änderung im Namen des Inhabers, das Ausscheiden eines Mitarbeiters aus einem Unternehmen, oder der Umstand, dass der private Schlüssel in die Hände einer unberechtigten Person gelangt ist, sein.

3.2.4 Verzeichnisdienst

Unter einem *Verzeichnisdienst* (auch *Key Server*) versteht man ein ein Verzeichnis, in welchem öffentliche Schlüssel abgelegt werden. *Key Server* erlauben es in der Regel, Suchen nach bestimmten Kriterien durchzuführen und die gefunden Schlüssel herunterzuladen. Für die technische Realisierung eines solchen Dienstes wird häufig ein LDAP-Server (*Lightweight Directory Access Protocol*) eingesetzt [Cha03] [CSF+08, S. 6f].

In Abbildung 4 auf Seite 9 wird die Verwendung eines *Verzeichnisdienstes* aufgezeigt. *Alice* möchte *Bob* eine Nachricht schicken. *Alice* liegt jedoch der öffentliche Schlüssel von *Bob* nicht vor. *Alice* führt eine Suche auf dem *Key Server* durch und lädt den *öffentlichen Schlüssel* von *Bob* herunter. Alice kann nun *Bob* eine verschlüsselte Nachricht zukommen lassen.

Verzeichnisdienste lösen das Problem, an die öffentlichen Schlüssel eines Kommunikationspartners zu gelangen. Die Schlüssel werden dort zentral abgelegt und können durchsucht werden. Es ist jedoch nicht sichergestellt, dass ein Schlüssel mit dem Namen *Bob* auch wirklich dem Benutzer *Bob* gehört. Erst die Existenz eines solchen Zertifikates stellt sicher, das der Schlüssel von *Bob* auch wirklich *Bob* gehört.

Bereits in der Arbeit von *Whitfield Diffie* und *Martin E. Hellman* aus dem Jahre 1976 wurde

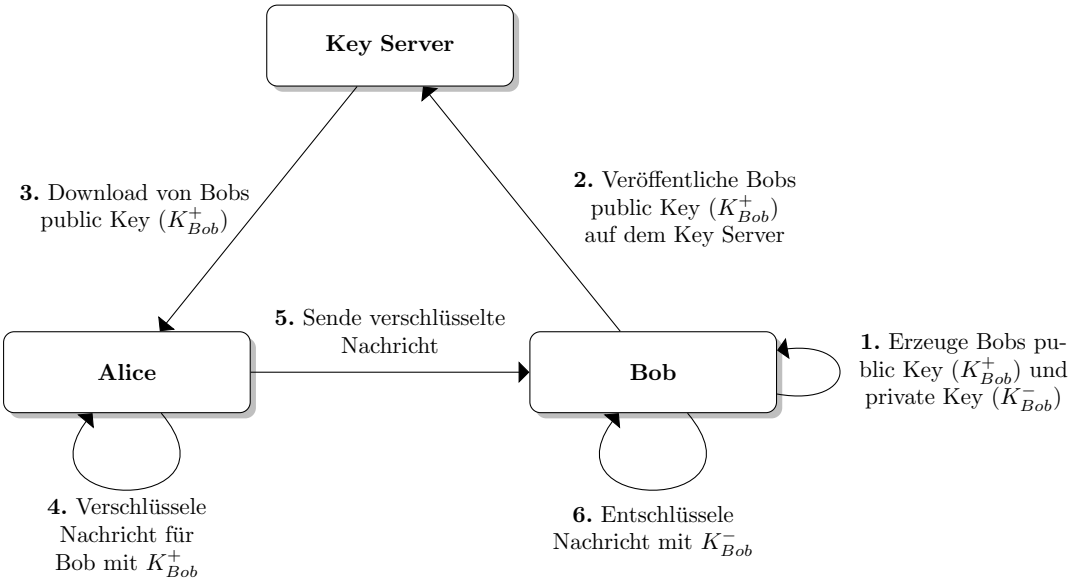

Abbildung 4: Austausch des öffentlichen Schlüssels von Bob (K^+_{Bob}) mit Hilfe eines *Key Servers*. (1) Bob erstellt einmalig ein Schlüsselpaar. (2) Der öffentliche Schlüssel von Bob wird auf einem *Key Server* abgelegt. (3) Alice liest (K^+_{Bob}) vom *Key Server* aus. (4) Eine Nachricht an Bob wird mit seinem *public Key* verschlüsselt. (5) Die Nachricht wird übertragen. (6) Bob entschlüsselt die Nachricht mit seinem *private Key* (K^-_{Bob}).

das Problem der Verteilung von Schlüsseln angesprochen [DH76, S. 6f]. Die Autoren schlagen einen Dienst mit dem Namen *public file* vor, in welchem die *öffentlichen Schlüssel* aller Kommunikationspartner abgelegt werden.

3.2.5 Dokumentationen

Eine *PKI* ist dafür verantwortlich, dass die ausgestellten Zertifikate vertrauenswürdig sind und nur nach einer genauen Prüfung der Identität des Antragstellers ausgestellt werden. Um die Arbeitsweise der *PKI* nachvollziehbar zu machen, wird diese Dokumentiert und veröffentlicht.

Vorschläge für eine solche Dokumentation sind in der *RFC 3647 Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework* [CFS+03] zu finden. Die wichtigsten beiden Dokumente sind die *Certificate Policy* und das *Certification Practice Statement*:

CP (Certificate Policy): Dieses Dokument beschreibt die Regeln, nach dem die *PKI* Zertifikate vergibt. Es dokumentiert zudem die die Sicherheitsanforderungen an die Arbeitsabläufe.[2]

CPS (Certification Practice Statement): Dieses Dokument beschreibt die konkreten Arbeitsabläufe der PKI. [3]

[2] [CFS+03, S. 6] *„A named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements.“*

[3] [CFS+03, S. 6] *„A statement of the practices that a certification authority employs in issuing, managing, revoking, and renewing or re-keying certificates.“*

3.3 Anwendung

Public key infrastructure wird für viele verschiedene Zwecke betrieben. So werden im Internet häufig Zertifikate eingesetzt, um den Komunikationspartner eindeutig zu identifizieren. Der im *RFC 2633* [Ram99] beschriebene Standard *S/MIME* (*Secure / Multipurpose Internet Mail Extensions*) beschreibt ein hybrides Kryptosystem[4] um Mails verschlüsselt zu verschicken. Hierbei kommen durch eine Zertifizierungsstelle ausgestellte Zertifikate zum Einsatz.

Die VPN Software *OpenVPN* [OPE13] baut eine eigene *PKI* auf, um den Zugang zum VPN zu regeln. Jeder, der im Besitz eines von der *OpenVPN* Zertifizierungsstelle unterschriebenen Zertifikates ist, kann eine VPN-Verbindung aufbauen. Die *PKI* beinhaltet ebenfalls eine *certificate revocation list* um Zertifikate als ungültig zu markieren und den Zugang zum VPN zu unterbinden.

In dem Betriebssystem *Windows Server* der Firma *Microsoft Corporation* wird eine *PKI* eingesetzt, um Clients zu authentifizieren. Die *PKI* baut auf dem Dienst *Microsoft Certificate Services* auf. *Microsoft* gibt unter anderem die folgenden Anwendungsmöglichkeiten für die *PKI* in Windows an [MSP13]:

- Sicherer Login mit Zertifikaten und Smartcards
- Vertrauliche und sichere E-Mails
- Sicherer Code (*Code signing*)
- Zugriffskontrolle und single-identity authorization

Unter dem Begriff *Code signing* versteht man das Signieren von ganzen Anwendungen, um Sicherzustellen das nachträglich keine Änderungen an dieser Anwendung vorgenommen worden sind. Auch können Signaturen dazu eingesetzt werden, nur bestimmte geprüfte Anwendungen ausführen zu lassen. So lädt das Betriebssystem *Windows*, ab der Version *Vista*, gemäß der *kernel-mode code signing policy* [MSD13] nur von der Microsoft Zertifizierungsstelle signierte Treiber in den Systemkernel.

3.4 Zusammenfassung

Eine *Public key infrastructure* (*PKI*) ist eine Ansammlung von Hardware, Software, Menschen, Vorschriften und Arbeitsabläufen, welche das Prüfen von Identitäten übernimmt. Geprüfte Beziehungen zwischen dem Inhaber eines Schlüssels und einem Schlüssel werden durch *Zertifikate* dokumentiert. Vertraut man der Zertifizierungsstelle einer *PKI*, so kann man anhand der von ihr ausgestellten Zertifikate die Identität von Kommunikationspartnern überprüfen. Eine eigene Prüfung der Identität ist nicht erforderlich.

[4] *Hybrides Kryptosystem:* Kombination von symmetrischer und asymmetrischer Verschlüsselung.

4 Das Web of trust

Parallel zu der bislang vorgestellten *Public key infrastuctrure* hat sich ein alternativer Ansatz, das *Web of trust*, entwickelt [UHHC11]. Dieser Ansatz kommt ohne eine zentrale Zertifizierungsstelle aus. Die Benutzer sprechen sich gegenseitig Vertrauen aus, indem sie Schlüssel mit ihrer Unterschrift versehen und die unterschriebenen Schlüssel veröffentlichen.

Im Jahr 1999 schrieb *Phill Zimmerman* im Handbuch zu seiner Software PGP [Zim99] *The Official PGP User's Guide*:

> „As time goes on, you will accumulate keys from other people that you may want to designate as trusted introducers. Everyone else will each choose their own trusted introducers. And everyone will gradually accumulate and distribute with their key a collection of certifying signatures from other people, with the expectation that anyone receiving it will trust at least one or two of the signatures. This will cause the emergence of a decentralized fault-tolerant web of confidence for all public keys."

Auf Deutsch:

> Mit der Zeit werden sie Schlüssel von Personen sammeln, welche sie als vertrauenswürdig eingestuft haben. Wem sie vertrauen, bestimmen sie selber. Mit der Zeit schaffen sie so eine Sammlung von Schlüsseln, welche sie als vertrauenswürdig ansehen. Dieser Sammlung von Schlüsseln vertrauen wiederum die Menschen, welchen ihnen vertrauen. Hierdurch entsteht ein fehlertolerantes und verteiltes Netzwerk von Personen und Schlüsseln, welche sich gegenseitig vertrauen entgegen bringen.

Der Ansatz macht sich das *small world phenomenon* zu nutze. Dieses geht davon aus, dass in sozialen Netzwerken jeder jeden über sechs Mittelsmänner kennt [Gur61] [TMTM69]. Die Abbildung 5 auf Seite 11 zeigt einen Ausschnitt aus dem Web of trust. Durch das *direkte Vertrauen* zwischen Personen entsteht ein *indirektes Vertrauen* zwischen sich unbekannten Personen.

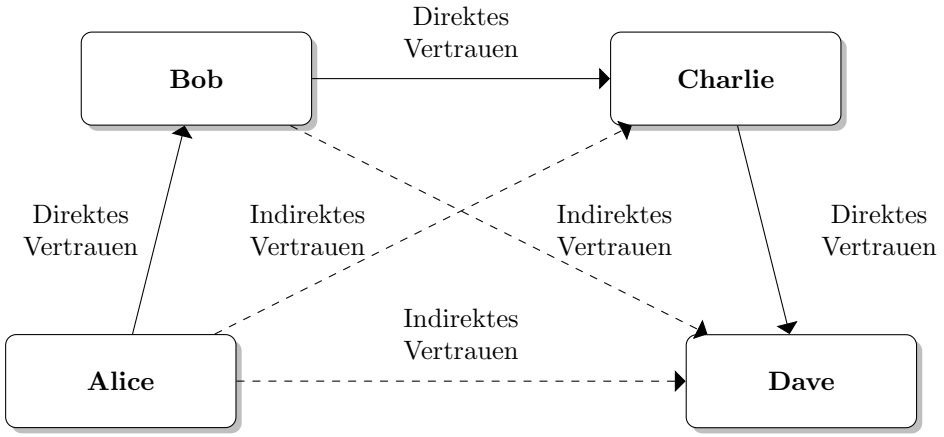

Abbildung 5: Beispiel für das Web of trust: Jeweils *Alice und Bob*, *Bob und Charlie* sowie *Charlie und Dave* vertrauen sich direkt. Durch dieses direkte Vertrauen entsteht ein indirektes Vertrauen zwischen *Alice und Charlie*, *Alice und Dave* sowie *Bob und Dave*.

In der *RFC 4880* ist das *OpenPGP Message Format* [CDF+07] beschrieben. Jede Software, welche diesen Standard umsetzt, kann an dem bestehenden *Web of trust* teilnehmen. Bekannte Implementationen dieses Standards sind die kommerzielle Software *PGP - Pretty Good Privacy* und die Open Source Implementation *GnuPG - GNU Privacy Guard*.

4.1 Verteilung von Schlüsseln

Das *Web of trust* ist darauf ausgelegt, ohne zentrale Infrastruktur auszukommen. Ohne derartige Infrastruktur ist es jedoch schwierig an die öffentlichen Schlüssel der Kommunikationspartner zu gelangen (siehe Abschnitt 3.2.4 auf Seite 8). Um dieses Problem zu lösen, werden, wie bei der *PKI*, auch im *Web of trust*, Keyserver eingesetzt.

Jeder kann mit passender Software einen eigenen Keyserver betreiben. Um die Schlüssel auf allen Keyservern verfügbar zu machen, synchronisieren sich die Keyserver untereinander. Ein Schlüssel muss somit nur auf einem Keyserver hinterlegt werden und ist danach auf allen anderen Keyservern verfügbar.

Neben der Verwendung von Keyservern sind noch zwei weitere Verteilungsmöglichkeiten gebräuchlich: (1) Der öffentliche Schlüssel kann auf der eigenen Webseite veröffentlicht werden. (2) Der Schlüssel wird z. B. in einer E-Mail ausgetauscht. Beide Verteilungsmöglichkeiten verlangen jedoch mehr Arbeit vom Benutzer, da der Schlüssel entweder auf einer Webseite gesucht oder explizit angefragt werden muss. Bei der Verwendung von Keyservern kann die Software beispielsweise beim Verfassen einer Mail prüfen, ob ein *öffentlicher Schlüssel* für den Empfänger auf einem Keyserver hinterlegt ist. Auf Wunsch kann dieser automatisch heruntergeladen und verwendet werden.

4.2 Vertrauen in Benutzer und Schlüssel

Das *Web of trust* lebt davon, dass Benutzer sich und ihren Schlüsseln gegenseitig vertrauen aussprechen. Dieses Vertrauen besteht aus zwei Komponenten:

- Man hat die Identität eines Schlüsselinhabers überprüft und belegt dies durch eine Unterschrift. Der unterschriebene Schlüssel wird anschließend wieder auf einem Keyserver publiziert, so dass jeder diese Unterschrift einsehen kann (*owner trust*). Personen die einen Schlüssel unterschrieben haben werden auch als *Vorsteller* (*introducer*) bezeichnet.
- Man traut dem anderen Benutzer ebenfalls zu, Identitäten korrekt zu überprüfen und andere Schlüssel nur nach einer vorhergehenden Überprüfung zu unterschreiben (*signature trust*).

Das Vertrauen in andere Benutzer und von ihnen unterschriebene Schlüssel kann in unterschiedlichen Vertrauensstufen ausgedrückt werden [Ash99]:

unknown: Es wurde bislang nicht festgelegt, wie stark dem Benutzer vertraut wird.

none: Dem Inhaber des Schlüssels wird nicht vertraut.

marginal trust: Der Inhaber des Schlüssels ist bekannt dafür, andere Schlüssel nur nach Prüfung zu unterschrieben.

complete trust: Der Inhaber des Schlüssels ist bekannt dafür, andere Schlüssel nur nach sehr genauer Prüfung zu unterschrieben. Von ihm signierte Schlüssel sind so vertrauenswürdig, wie selbst geprüfte Schlüssel.

4.3 Public key fingerprints

Schlüssel bestehen in der Regel aus einer langen Folge von Buchstaben und Ziffern. Bei der Überprüfung eines Schlüssels muss dieser zusammen mit der Identität eines Benutzers überprüft werden. Um nicht die oft mehrere 1000 Zeichen langen Schlüssel Zeichen für Zeichen überprüfen zu müssen, werden *Fingerprints* verwendet. Diese *Fingerprints* sind mit einem Hashwert zu einem Schlüssel vergleichbar. In der Praxis werden vor dem Unterschreiben eines Schlüssels nur diese *Fingerprints* überprüft.

Ein Beispiel für einen solchen Fingerprint ist in Listing 1 aufgeführt. Der Fingerprint zum Schlüssel des Autors lautet demnach: `539D 2D00 E9C5 3DC0 EBEB E964 D503 EAA6 30D4 5E5B`.

Listing 1: Fingerprint des public Key des Autors

```
1 # gpg --fingerprint knidzwetzki@gmx.de
2
3 pub   1024D/30D45E5B 2002-03-29 Schl.-Fingerabdruck = 539D 2D00 E9C5 3
      DC0 EBEB  E964 D503 EAA6 30D4 5E5B
4 uid   Jan Kristof Nidzwetzki <kristof.nidzwetzki@achfrag.net>
5 uid   Jan Kristof Nidzwetzki <knidzwetzki@gmx.de>
6 uid   Jan Kristof Nidzwetzki <tux@freenet.de>
7 sub   1024g/D622512C 2002-03-29
```

4.4 Indirektes Vertrauen

Der Benutzer *Alice* möchte dem Benutzer *Bob* eine Nachricht schicken. Vor der ersten Kommunikation muss *Alice* den vorliegenden Schlüssel von *Bob* überprüfen und sicherstellen, dass dieser auch wirklich *Bob* gehört. Dazu existieren zwei gängige Verfahren:

- *Alice* kann *Bob* treffen und persönlich seinen Key überprüfen.
- Es werden die Informationen aus dem *Web of trust* genutzt. Aus diesen wird berechnet, ob der Schlüssel von *Bob* als vertrauenswürdig angesehen werden kann.

Werden Informationen aus dem *Web of trust* benutzt, sind zwei Faktoren dafür entscheidend, ob ein Schlüssel als vertrauenswürdig angesehen werden kann: (1) Wer ihn unterschrieben hat. (2) Von persönlichen Präferenzen, wie viele Unterschriften ein Schlüssel besitzen muss, um vertrauenswürdig zu erscheinen.

Der einfachste Fall liegt vor, wenn ein Schlüssel vom Benutzer selbst unterschrieben worden ist. In diesem Fall kann der Schlüssel als persönlich geprüft und gültig angesehen werden. Schwieriger wird es, wenn der Schlüssel nicht vom Benutzer direkt, sondern indirekt von anderen bekannten Benutzern unterschrieben wurde.

In der Arbeit *The PGP Trust Model* des Autors *Alfarez Abdul-Rahman* aus dem Jahre 1997 [AR97] werden einfache Algorithmen vorgestellt, um das Vertrauen in andere Schlüssel zu berechnen. Der Autor führt hierzu zwei vom Benutzer vorgegebene Variablen ein: `marginals_needed` und `complete_needed` [AR97, S. 3ff]. Diese Variablen geben an, wie viele Unterschriften ein Schlüssel benötigt, um als vertrauenswürdig angesehen zu werden. Die Variablen beziehen sich auf das Vertrauen (*marginal* oder *complete*), welches man der Unterschrift entgegenbringt.

Der in Listing 2 aufgeführte Algorithmus entscheidet anhand der vorhandenen Unterschriften und der beiden Variablen `marginals_needed` und `complete_needed`, ob ein Schlüssel als vertrauenswürdig angesehen werden soll oder nicht.

Der Algorithmus geht alle Unterschriften durch, mit denen ein Schlüssel unterschrieben worden ist. Zunächst prüft er (Zeile 2) ob die Signatur gültig ist und verwendet werden kann. In den Zeilen 3-10 wird überprüft, ob der Signatur vom Benutzer ein Vertrauen zugeordnet ist. Je nachdem wird die Signatur entweder ignoriert oder eine der beiden Variablen `marginals_counter` und `complete_counter` inkrementiert. Zum Schluss, Zeilen 13-18, wird entschieden, ob der Schlüssel als gültig angesehen wird oder nicht. Dies hängt davon ab, ob die Variablen `marginals_counter` und `complete_counter`, über den Schwellwerten (`marginals_needed` und `complete_needed`) liegen.

Listing 2: Algorithmus zum Berechnen der Key Legitimacy

```
1  For each signature do
2    // scan signatures
3    if signature is completely valid
4      if key trust ∈ {undefined,unknown,untrusted}
5        ignore signature
6      if key trust is marginal
7        accumulate marginal_counter
8      if key trust is complete
9        accumulate complete_counter
10     else
11       ignore signature
12
13     // decision
14     if(marginal_counter > 0) or (complete_counter > 0)
15       if(marginal_counter >= marginals_needed or complete_counter >=
             complete_needed)
16         mark key validity as 'complete'
17       else
18         mark key validity as 'marginal'
```

4.5 Zurückziehen von Schlüsseln

Auch im *Web of trust* kommt es vor, dass Schlüssel zurückgezogen werden müssen. Es ist jedoch nicht möglich, auf Keyservern veröffentlichte Schlüssel zu löschen. Zudem besitzt *Web of trust* keine *Certificate Revocation List* (*CRL*). Dennoch ist es auch hier möglich, einmal veröffentlichte Schlüssel als ungültig zu markieren und diese zurückzunehmen (*key revoke*). Es wird hierbei mit einem *revocation certificate* (*Widerruf-Zertifikat*) gearbeitet.

Um einen Schlüssel als ungültig zu markieren, wird das dazugehörige *revocation certificate* auf einem Keyserver veröffentlicht. Software welche die *RFC 4880* umsetzt und das *OpenPGP Message Format* unterstützt, erkennt, dass ein *revocation certificate* veröffentlicht worden ist und zeigt den Schlüssel entsprechend als ungültig an.

4.6 Key signing party

Das *Web of trust* lebt davon, dass möglichst viele Menschen sich untereinander Vertrauen ausgesprochen haben. Dies setzt voraus, dass sich Menschen, welche das System nutzen, getroffen und ihre Identitäten und Schlüssel überprüft haben. Zu verschiedenen Anlässen werden daher *Key signing partys* veranstaltet. Ziel einer solchen Party ist es, mit bislang unbekannten Menschen in Kontakt zu kommen und ihre Schlüssel zu unterschrieben.

4.7 Anwendung

Bekannte Anwendungen, welche das *OpenPGP Message Format* unterstützen sind das kommerzielle *Pretty Good Privacy - PGP* und das unter der Open Source Lizenz *GPL* stehende *GnuPG - GNU Privacy Guard*. Die erste Version der Software *PGP* wurde im Sommer des Jahres 1991 von *Phil Zimmermann* veröffentlicht. Sie machte starke Kryptographie erstmals für eine breite Masse von Menschen zugänglich [Sin99, S. 227f]. Das Ziel von *Phil Zimmermann* war es, dass alle Bürger verschlüsselte Nachrichten austauschen können [Sin99, S. 223f].

Die Software *Pretty Good Privacy - PGP* gehört heute der Firma *Symantec Corporation*, welche diese kommerziell vertreibt. Als freie Open Source Software wird häufig die *GnuPG - GNU Privacy Guard* [GPG13] eingesetzt.

Neben dem Verschlüsseln und Signieren von Nachrichten wie E-Mails werden auch ganze Softwarepakete signiert. Linux-Distributionen wie *Debian* oder *Opensuse* signieren die ausgelieferte Software. Dies soll sicherstellen, dass die Software nicht von Dritten verändert wird und beispielsweise Hintertüren eingebaut werden.

Auch die verteilte Versionsverwaltung *Git*, welche unter anderem für die Entwicklung des Linux-Kernels eingesetzt wird, bietet den Entwicklern die Möglichkeit, einzelne Versionsstände zu signieren (*Signed Tags*) [GIT13].

4.8 Zusammenfassung

Die im Abschnitt 3 vorgestellte *Public Key Infrasructure* (*PKI*) benötigt Dienstleistungen, welche meist kommerziell durch Unternehmen angeboten werden. So prüfen die Anbieter einer *PKI*, meist gegen eine Gebühr Identitäten und stellen Zertifikate aus. Zudem betreiben die Unternehmen notwendige zentrale Infrastruktur wie *Verzeichnisserver* oder eine *Certificate Revocation List*.

Das in diesem Abschnitt vorgestellte Konzept des *Web of trust* kommt ohne diese zentralen Komponenten aus und setzt auf freiwilligen Einsatz der Anwender, wie etwa die Teilnahme an einer Key signing party. Inhaber von Schlüsseln sprechen sich gegenseitig Vertrauen aus und bekunden dies durch eine Unterschrift unter dem Schlüssel des anderen. Auch der Betrieb von zentralen Systemen wie etwa von Keyservern (siehe Abschnitt 4.1 auf Seite 12) wird von Freiwilligen übernommen. Ein kommerzielles Interesse steht bei diesem Konzept im Hintergrund. Eine Gegenüberstellung der Komponenten einer PKI und der Komponenten des *Web of trust* ist in Tabelle 3 aufgeführt.

Komponente einer PKI	Äquivalent im Web of trust
Zertifizierungsstelle	*Unterschreiben von Schlüsseln*
Registrierungsstelle	*Persönliche Prüfung von Identitäten*
Zertifikatsperrliste	*Revocation certificate*
Verzeichnisdienst	*Verzeichnisdienst*

Tabelle 3: Gegenüberstellung der Komponenten von *PKI* und dem *Web of trust*

5 Identity-based encryption

Die *Identity-based encryption* (*ID-based encryption - IBE*) ist ein wichtiges Teilgebiet der *ID-based cryptography*. Diese setzt auf den Konzepten der *asymmetrischen Verschlüsselung* auf (siehe Abschnitt 2 auf Seite 2).

In Jahr 1984 wurden von *Adi Shamir* die ersten Ideen zur *IBE* in der Arbeit *Identity-based cryptosystems and signature schemes* [Sha85] formuliert. Shamirs Motivation war es, den aufwändigen Austausch der *öffentlichen Schlüssel* (*public Keys*) zu vereinfachen. Er stellt in seiner Arbeit die Idee vor, den *öffentlichen Schlüssel* anhand von öffentlich zugänglichen Informationen wie Name, E-Mail Adresse, etc. zu berechnen. Dabei soll vollständig auf den Austausch von Keys, den Rückgriff auf *Schlüssel-Verzeichnisse* (*Key directorys*) oder Dienste einer dritten Partei verzichtet werden.

Er stellt in der Arbeit ein Verfahren vor, mit dem auf diese Weise Nachrichten signiert und verifiziert werden können. Ein konkretes Verfahren zur Verschlüsselung von Nachrichten konnte er jedoch in dieser Arbeit nicht angeben. Im Jahr 1986 stellte *Adi Shamir* zusammen mit *Amos Fiat* die Arbeit *How to prove yourself: practical solutions to identification and signature problems* vor [FS87]. In dieser wurden die Konzepte seiner ursprünglichen Arbeit verbessert und mit *zero-knowlege interactive proofs* kombiniert [GMR89] [QGAB89].

Nach Erscheinen dieser Arbeiten wurden verschiedene Vorschläge unterbreitet, wie ein solches Verschlüsselungsverfahren aussehen könnte. Diese Vorschläge hatten jedoch alle mit verschiedenen Problemen zu kämpfen [EG85], [Tan88], [TI89]. Die Einschränkungen reichen von für die Praxis zu langsamen und komplexen Algorithmen, bis hin zu dem Verbot, dass Anwender untereinander bestimmte Informationen austauschen. (siehe [BF03, S.1]).

Im Jahre 2001 wurden die ersten praxistauglichen Verfahren beschrieben. In der Arbeit *Identity-Based Encryption from the Weil Pairing* [BF03] wurde von den Autoren *Boneh* und *Franklin* eine praxistaugliche Realisierung vorgestellt. Im gleichen Jahre wurde von dem britischen Mathematiker *Clifford Cocks* mit der Arbeit *An Identity Based Encryption Scheme Based on Quadratic Residues* [Coc01] ein ebenfalls praxistaugliches Verfahren vorgestellt.

Die Autoren *Xuhua Ding* und *Gene Tsudik* haben in ihrer 2003 erschienenen Arbeit *Simple identity-based cryptography with mediated RSA* ein Verfahren für *Identity-based encryption* auf Basis des Algorithmus *RSA* vorgestellt [DT03]. Die 2005 erschienene Abhandlung *Boneh-Franklin identity based encryption revisited* [Gal05] geht auf die Realisierung von *Boneh* und *Franklin* ein und verbessert diese. Im Jahr 2006 wurde mit der Arbeit *k-Resilient Identity-Based Encryption in the Standard Model* [HK06] eine weitere Realisierung vorgestellt.

Ein Vergleich der Performance der verschiedenen Algorithmen findet sich in *A performance analysis of identity-based encryption schemes* [CGL+12].

5.1 Identity-based encryption mit Private Key Generator

Die von Shamir 1984 geforderte Unabhängigkeit von einer dritten Partei konnte bislang in der Praxis nicht erreicht werden. Heutige *Identity-based encryption* Verfahren nutzen die Dienste eines *Private Key Generators* (*PKG*) (siehe [BF03, S. 213 ff.]) um die *privaten Schlüssel* zu erzeugen (Abbildung 6).

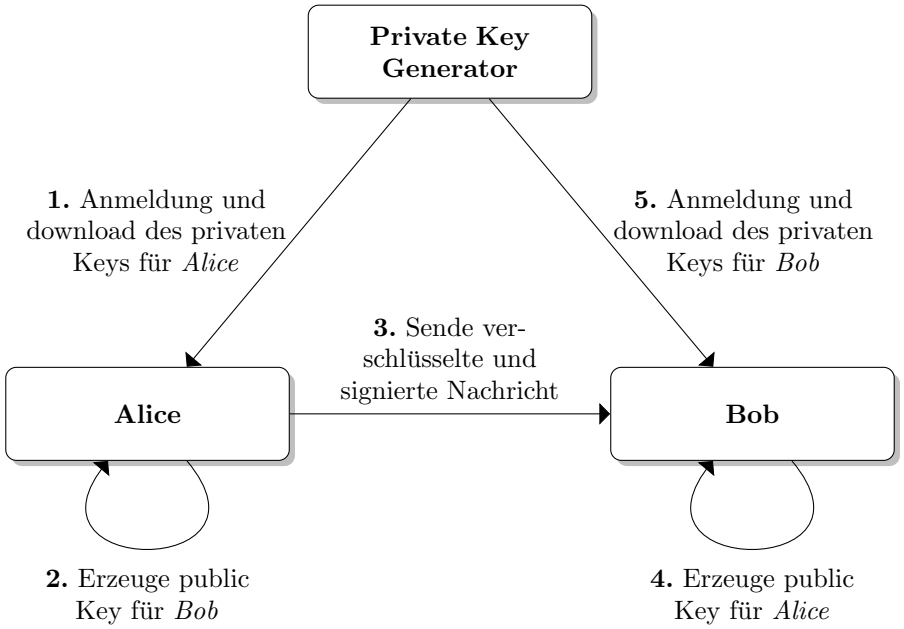

Abbildung 6: *Identity-based encryption* mit *Private key generator*: *Alice* möchte *Bob* eine signierte und verschlüsselte Nachricht zukommen lassen. Dazu ruft *Alice* ihren *private Key* beim *Private Key Generator* ab. Mit diesem wird die Nachricht signiert. Anhand von öffentlich bekannten Informationen von *Bob* berechnet *Alice* den *public Key* von *Bob* und verschlüsselt die Nachricht damit. *Bob* ruft ebenfalls seinen *private Key* vom *Private Key Generator* ab. Mit diesem wird die Nachricht entschlüsselt. Analog zu *Alice* berechnet *Bob* den public key von Alice und überprüft damit die Signatur der Nachricht.

5.2 Hirarchical Identity-based encryption

Der *Private Key Generator* muss sehr aufwändige Berechnungen durchführen. Die Autoren *Craig Gentry* und *Alice Silverberg* schlagen in ihrer Arbeit *Hierarchical ID-Based Cryptography* [GS02] 2002 vor, die Arbeit des *Private Key Generator* auf mehrere Systeme zu verteilen, die in einer hierarchischen Beziehung zueinander stehen.

Weiterentwicklungen dieser Idee finden sich in den Arbeiten *Hierarchical Identity Based Encryption with Constant Size Ciphertext* [BBG05], *Identity-Based hierarchical strongly key-insulated encryption and its application* [HHSI05], *Toward Hierarchical Identity-Based Encryption* [HL02] und *Anonymous hierarchical identity-based encryption* (*without random oracles*) [BW06].

5.3 Gebräuchliche Standards

In den letzten Jahren wurden einige RFCs (*Request for Comments*) seitens der *Internet Engineering Task Force* (*IETF*) veröffentlicht. Diese beschreiben verschiedene Standards, wie die *IBE* angewendet werden kann.

RFC 5091: *Identity-Based Cryptography Standard (IBCS): Supersingular Curve Implementations of the BF and BB1 Cryptosystems [BM07].*

RFC 5408: *Identity-Based Encryption Architecture and Supporting Data Structures* [AMS09].

RFC 5409: *Using the Boneh-Franklin and Boneh-Boyen Identity-Based Encryption Algorithms with the Cryptographic Message Syntax* [MS09].

5.4 Alternativen und Weiterentwicklungen

Wildcarded Identity-Based Encryption: Sollen verschlüsselte Nachrichten an mehre Empfänger zugestellt werden, so bietet die *Wildcarded Identity-Based Encryption* hierfür eine Lösung. Dieses Konzept wird in der Arbeit *Wildcarded Identity-Based Encryption* [ABC+11] vorgestellt. Die Ideen werden in der Arbeit *Efficient chosen-ciphertext secure identity-based encryption with wildcards* [BDNS06] ebenfalls beschrieben. Auch in *Efficient multi-receiver identity-based encryption and its application to broadcast encryption* [BSNS05] wird auf dieses Problem eingegangen.

Anonymous Hierarchical Identity-Based Encryption: Möchte man anhand einer verschlüsselten Nachricht nicht auf den Empfänger schließen können, so bietet die *Anonymous Hierarchical Identity-Based Encryption* (*Anonymous HIBE*) hierfür eine Lösung.

Behandelt wird diese in den Arbeiten *Anonymous Hierarchical Identity-Based Encryption with Constant Size Ciphertexts* [SKOS09], *Anonymity from asymmetry: new constructions for anonymous HIBE* [Duc10] und *Anonymous Hierarchical Identity-Based Encryption in Prime Order Groups* [RWZ12].

Identity-Based Online/Offline Encryption: Die Verschlüsselung von Nachrichten besteht aus verschieden komplexen Berechnungen. Im Gebiet der *Identity-Based Online/Offline Encryption* werden die komplexen Berechnungen auf leistungsstarken Systemen „*offline*“ vorgenommen, ohne den Empfänger (und damit den *public Key*) oder die Nachricht zu kennen. Die eigentliche Verschlüsselung der Nachricht erfolgt dann „*online*“ auf weniger leistungsfähigen Geräten wie Smartcards.

Durch das Auslagern der komplexen Berechnung auf leistungsfähige Systeme kann die eigentliche Verschlüsselung auf leistungsschwachen Systemen deutlich schneller erfolgen. Dieses Thema wird in den Arbeiten *Identity-Based Online/Offline Encryption* [GMC08], *An Efficient Identity-Based Online/Offline Encryption Scheme* [LZ09] und *Identity based online/offline encryption and signcryption schemes revisited* [SVR11] behandelt.

5.5 Anwendung

Die Arbeit *A Flexible Role-based Secure Messaging Service: Exploiting IBE Technology for Privacy in Health Care* [MBH03] beschreibt den Einsatz der Identity-Based Encryption im Gesundheitswesen.

In der Software-Bibliothek *PBC Library - The Pairing-Based Cryptography Library* [PBC12] findet sich eine Implementation des Konzepts in der Programmiersprache *C*. Die Firma *Voltage Security Inc.* stellt mit dem *Voltage IBE Toolkit* [VOL12] eine kommerzielle Implementation der *IBE* zur Verfügung.

5.6 Zusammenfassung

Die *Identity-based encryption* löst das Problem der Schlüsselverteilung auf eine andere Art und Weise als das *Web of trust* oder eine *PKI*. Anstatt Schlüssel in einem zentralen Verzeichnis abzulegen, können die *öffentlichen Schlüssel* aus öffentlich zugänglichen Informationen berechnet werden. Der komplexe Austausch von Schlüsseln wird so vermieden. Für die praktische Umsetzung ist jedoch eine vertrauenswürdige dritte Partei erforderlich, welche die *privaten Schlüssel* der Teilnehmer berechnet und ausliefert.

In den letzten Jahrzehnten wurden viele Forschungsarbeiten zur *Identity-based encryption* publiziert und Weiterentwicklungen vorgestellt. Seit dem Jahr 2001 existieren konkrete Algorithmen und Implementationen für die *Identity-based encryption*. In der Praxis sind diese jedoch bislang nur wenig verbreitet.

6 Zusammenfassung und Ausblick

Es wurden in dieser Arbeit die für die *public Key* Verschlüsselung gängige Infrastruktur vorgestellt. Die Abschnitte 1 und 2 haben sich mit allgemeinen Grundlagen der *public Key Verschlüsselung* beschäftigt. Es wurde das Konzept der Signaturen vorgestellt, die Notwendigkeit Schlüssel auszutauschen und der *Man-in-the-middle Angriff* erläutert.

Der Abschnitt 3 hat sich mit dem Standard *X.509* und *public key infrastucture* beschäftigt. Es wurden die gängigen Komponenten einer *PKI* vorgestellt und deren Einsatz erläutert. Hier wurden unter anderem die Komponenten *Zertifizierungsstelle* und die *Zertifikatsperrliste* behandelt. Weitergehende Konzepte wie das signieren von Code oder der Standard S/MIME wurden kurz angesprochen.

In Abschnitt 4 wurde das *Web of trust* betrachtet. Dieses versucht die gleichen Probleme wie die *PKI* zu lösen, setzt hierzu jedoch andere Konzepte ein. Während die *PKI* genau dokumentierte Arbeitsabläufe besitzt und zentrale Infrastuktur bereitstellt, setzt das *Web of trust* auf eine dezentrale Infrastruktur und die aktive Mitarbeit der Teilnehmer. Jeder Teilnehmer kann selbst Identitäten prüfen und dies mittels einer Unterschrift dokumentieren. Zudem wurde in diesem Abschnitt das Konzept des *indirekten Vertrauens* eingeführt. Gegen Ende des Abschnitts wurden die Komponenten einer *PKI* und die äquivalenten Komponenten des *Web of trust* gegenübergestellt. Darüberhinaus wurden *Fingerprints* angesprochen und *Key Signing partys* beschrieben. Ebenfalls wurden Anwendungen wie der Einsatz von Signaturen bei der verteilten Softwareentwicklung mit *git* behandelt.

Der Abschnitt 5 beschäftigte sich mit der *Identity-based encryption*. Dieses Konzept versucht den aufwändigen Austausch von Schlüsseln zu vermeiden, indem *öffentliche Schlüssel* anhand öffentlich zugänglicher Informationen, wie beispielsweise der Telefonnummer oder der Adresse, berechnet werden. Es wurde zudem auf gängige Standards und *Identity-based encryption* mit *Private key generator* eingegangen. Ferner wurden neuere Entwicklungen wie die *Wildcarded Identity-Based Encryption* kurz vorgestellt.

Ausblick

Neben den klassischen Einsatzgebieten wie der verschlüsselten Übertragung von Nachrichten, wird die *public Key Verschlüsselung* auch zunehmend in neuen Bereichen eingesetzt. Die *Identity-based encryption* stellt mit ihren weniger als 15 Jahre alten Algorithmen ein recht neues Gebiet dar, auf welchem viel Forschung stattfindet. Da die *Identity-based encryption* einerseits wenig praktischen Einsatz findet, andererseits die Anwendungsgebiete für *public Key Verschlüsselung* zunehmen, könnte sich die *IBE* den nächsten Jahren deutlich stärker in der Praxis verbreiten.

Literatur

[ABC+11] Michel Abdalla, James Birkett, Dario Catalano, Alexander W. Dent, John Malone-Lee, Gregory Neven, Jacob C. N. Schuldt, and Nigel P. Smart. Wildcarded identity-based encryption. *J. Cryptology*, 24(1):42–82, 2011.

[AMS09] G. Appenzeller, L. Martin, and M. Schertler. Identity-Based Encryption Architecture and Supporting Data Structures. RFC 5408 (Informational), January 2009.

[AR97] Alfarez Abdul-Rahman. The PGP Trust Model. *EDI-Forum: The Journal of Electronic Commerce*, 10(3):27–31, 1997.

[Ash99] Mike Ashley. The GNU Privacy Handbook. Technical report, Free Software Foundation, 1999.

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, 2005.

[BDNS06] James Birkett, Er W. Dent, Gregory Neven, and Jacob C. N. Schuldt. Efficient chosen-ciphertext secure identity-based encryption with wildcards. Technical report, Cryptology ePrint Archive, 2006. [BF01] Dan Boneh and Matthew, 2006.

[Beu07] A. Beutelspacher. *Kryptologie*. Vieweg, 2007.

[BF03] Dan Boneh and Matthew Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, March 2003.

[BM07] X. Boyen and L. Martin. Identity-Based Cryptography Standard (IBCS) 1: Supersingular Curve Implementations of the BF and BB1 Cryptosystems. RFC 5091 (Informational), December 2007.

[BPH02] L. Bassham, W. Polk, and R. Housley. Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 3279 (Proposed Standard), April 2002. Updated by RFCs 4055, 4491, 5480.

[BSNS05] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Efficient multi-receiver identity-based encryption and its application to broadcast encryption. In *Proceedings of the 8th international conference on Theory and Practice in Public Key Cryptography*, PKC'05, pages 380–397, Berlin, Heidelberg, 2005. Springer-Verlag.

[BW06] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *Proceedings of the 26th annual international conference on Advances in Cryptology*, CRYPTO'06, pages 290–307, Berlin, Heidelberg, 2006. Springer-Verlag.

[Can03] Ran Canetti. Universally composable signatures, certification and authentication. Cryptology ePrint Archive, Report 2003/239, 2003. `http://eprint.iacr.org/`.

[CDF+07] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and F. Thayer. RFC 4880 - OpenPGP Message Format. Technical report, Internet Engineering Task Force, November 2007.

[CFS+03] S. Chokhani, W. Ford, R. Sabett, C. Merrill, and S. Wu. Internet x.509 public key infrastructure certificate policy and certification practices framework, 2003.

[CGL+12] Pengqi Cheng, Yan Gu, Zihong Lv, Jianfei Wang, Wenlei Zhu, Zhen Chen, and Jiwei Huang. A performance analysis of identity-based encryption schemes. In *Proceedings of the Third international conference on Trusted Systems*, INTRUST'11, pages 289–303, Berlin, Heidelberg, 2012. Springer-Verlag.

[Cha03] David Chadwick. Deficiencies in ldap when used to support pki. *Commun. ACM*, 46(3):99–104, March 2003.

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, pages 360–363, London, UK, UK, 2001. Springer-Verlag.

[CSF+08] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. RFC 5280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Technical report, May 2008.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography, 1976.

[DT03] Xuhua Ding and Gene Tsudik. Simple identity-based cryptography with mediated rsa. In *Proceedings of the 2003 RSA conference on The cryptographers' track*, CT-RSA'03, pages 193–210, Berlin, Heidelberg, 2003. Springer-Verlag.

[Duc10] Léo Ducas. Anonymity from asymmetry: new constructions for anonymous hibe. In *Proceedings of the 2010 international conference on Topics in Cryptology*, CT-RSA'10, pages 148–164, Berlin, Heidelberg, 2010. Springer-Verlag.

[EG85] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Proceedings on Advances in cryptology—CRYPTO '86*, pages 186–194, London, UK, UK, 1987. Springer-Verlag.

[FS03] Niels Ferguson and Bruce Schneier. *Practical cryptography.* Wiley, 2003.

[Gal05] David Galindo. Boneh-franklin identity based encryption revisited. In *Proceedings of the 32nd international conference on Automata, Languages and Programming*, ICALP'05, pages 791–802, Berlin, Heidelberg, 2005. Springer-Verlag.

[GIT13] Signed tags - git reference manual, 2013. http://git-scm.com/book/ch2-6.html - Abgerufen am 10.03.2013.

[GMC08] Fuchun Guo, Yi Mu, and Zhide Chen. Financial cryptography and data security. chapter Identity-Based Online/Offline Encryption, pages 247–261. Springer-Verlag, Berlin, Heidelberg, 2008.

[GMR89] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, February 1989.

[GPG13] Website - the gnu privacy guard, 2013. http://www.gnupg.org/ - Abgerufen am 10.03.2013.

[GS02] Craig Gentry and Alice Silverberg. Hierarchical id-based cryptography. In *Proceedings of the 8th International Conference on the Theory and Application of Cryptology and Information Security: Advances in Cryptology*, ASIACRYPT '02, pages 548–566, London, UK, UK, 2002. Springer-Verlag.

[Gur61] M. Gurevitch. *The Social Structure of Acquaintanceship Networks.* Massachusetts Institute of Technology, Department of Economics and Social Science, 1961.

[Gut02] Peter Gutmann. Pki: It's not dead, just resting. *IEEE Computer*, 35(8):41–49, 2002.

[HFPS99] R. Housley, W. Ford, W. Polk, and D. Solo. Internet x.509 public key infrastructure certificate and crl profile, 1999.

[HHSI05] Yumiko Hanaoka, Goichiro Hanaoka, Junji Shikata, and Hideki Imai. Identity-based hierarchical strongly key-insulated encryption and its application. In *Proceedings of the 11th international conference on Theory and Application of Cryptology and Information Security*, ASIACRYPT'05, pages 495–514, Berlin, Heidelberg, 2005. Springer-Verlag.

[HK06] Swee-Huay Heng and Kaoru Kurosawa. k-resilient identity-based encryption in the standard model. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, E89-A(1):39–46, January 2006.

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In *Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, EUROCRYPT '02, pages 466–481, London, UK, UK, 2002. Springer-Verlag.

[HPFS02] R. Housley, W. Polk, W. Ford, and D. Solo. Internet x.509 public key infrastructure certificate and certificate revocation list (crl) profile, 2002.

[ITU88] INTERNATIONAL TELECOMMUNICATION UNION ITU. The Directory Authentication Framework. Series X: Data Communication Networks: Directory 8, International Telecommunication Union, Sussa, Genebra, nov 1988. Reedition of CCITT Recommendation X.509 published in the Blue Book, Fascicle VIII.8 (1988).

[Koh78] Kohnfleder. Towards a Practical Public Key Cryptosystem. Bachelor's thesis, May 1978.

[Lam79] L. Lamport. Constructing digital signatures from a one-way function. Technical report, October 1979.

[LZ09] Joseph K. Liu and Jianying Zhou. An efficient identity-based online/offline encryption scheme. In *Proceedings of the 7th International Conference on Applied Cryptography and Network Security*, ACNS '09, pages 156–167, Berlin, Heidelberg, 2009. Springer-Verlag.

[MAM+99] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. X.509 internet public key infrastructure online certificate status protocol - ocsp, 1999.

[MBH03] Marco Casassa Mont, Pete Bramhall, and Keith Harrison. A flexible role-based secure messaging service: Exploiting ibe technology for privacy in health care. In *DEXA Workshops*, pages 432–437. IEEE Computer Society, 2003.

[Mer78] Ralph C. Merkle. Secure communications over insecure channels. *Commun. ACM*, 21(4):294–299, April 1978.

[Mer89] Ralph C. Merkle. A certified digital signature. In *Proceedings on Advances in cryptology*, CRYPTO '89, pages 218–238, New York, NY, USA, 1989. Springer-Verlag New York, Inc.

[Moz01] Mozilla. Bug 110161 - Enable OCSP by default. Mozilla Bug Database, 2001.

[MS09] L. Martin and M. Schertler. Using the Boneh-Franklin and Boneh-Boyen Identity-Based Encryption Algorithms with the Cryptographic Message Syntax (CMS). RFC 5409 (Informational), January 2009.

[MSD13] Microsoft driver signing policy, 2013. http://msdn.microsoft.com/en-us/library/windows/hardware/ff548231%28v=vs.85%29.aspx - Abgerufen am 10.03.2013.

[MSP13] Website pki integration into existing environments, 2013. http://technet.microsoft.com/en-us/library/cc737335%28v=ws.10%29.aspx - Abgerufen am 10.03.2013.

[OPE13] Website - openvpn, 2013. http://openvpn.net/ - Abgerufen am 10.03.2013.

[PBC12] Pbc library - the pairing-based cryptography library, 2012. http://crypto.stanford.edu/pbc/ - Abgerufen am 20.12.2012.

[QGAB89] Jean-Jacques Quisquater, Louis Guillou, Marie Annick, and Tom Berson. How to explain zero-knowledge protocols to your children. In *Proceedings on Advances in cryptology*, CRYPTO '89, pages 628–631, New York, NY, USA, 1989. Springer-Verlag New York, Inc.

[Ram99] Blake Ramsdell. S/mime version 3 message specification. Internet RFC 2633, June 1999.

[RSA78] R.L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

[RWZ12] Yanli Ren, Shuozhong Wang, and Xinpeng Zhang. Anonymous hierarchical identity-based encryption in prime order groups. In Yang Xiang, Mukaddim Pathan, Xiaohui Tao, and Hua Wang, editors, *ICDKE*, volume 7696 of *Lecture Notes in Computer Science*, pages 230–242. Springer, 2012.

[Sch95] Bruce Schneier. *Applied cryptography (2nd ed.): protocols, algorithms, and source code in C*. John Wiley & Sons, Inc., New York, NY, USA, 1995.

[Sha85] Adi Shamir. Ideontity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in cryptology*, pages 47–53, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

[Shi00] R. Shirey. Internet Security Glossary. RFC 2828 (Informational), May 2000. Obsoleted by RFC 4949.

[Sin99] Simon Singh. *The Code Book: The Evolution of Secrecy from Mary, Queen of Scots, to Quantum Cryptography*. Doubleday, New York, NY, USA, 1st edition, 1999.

[SKOS09] Jae Hong Seo, Tetsutaro Kobayashi, Miyako Ohkubo, and Koutarou Suzuki. Anonymous hierarchical identity-based encryption with constant size ciphertexts. In *Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography: PKC '09*, Irvine, pages 215–234, Berlin, Heidelberg, 2009. Springer-Verlag.

[Sta06] William Stallings. *Cryptography and Network Security: Principles and Practice.* Pearson Education, 4th edition, 2006.

[Sue02] Caius T. Suetonius. De vita Caesarum, Liber I, Diuus Iulius, LVI. In Henri Ailloud and François L'Yvonnet, editors, *Vies des douze Césars*, volume I, page 72. Les belles lettres, Paris, 2002.

[SVR11] S. Sharmila Deva Selvi, S. Sree Vivek, and C. Pandu Rangan. Identity based online/offline encryption and signcryption schemes revisited. In *Proceedings of the First international conference on Security aspects in information technology*, InfoSecHiComNet'11, pages 111–127, Berlin, Heidelberg, 2011. Springer-Verlag.

[Tan88] Hatsukazu Tanaka. A realization scheme for the identity-based cryptosystem. In *A Conference on the Theory and Applications of Cryptographic Techniques on Advances in Cryptology*, CRYPTO '87, pages 340–349, London, UK, UK, 1988. Springer-Verlag.

[TI89] S. Tsujii and T. Itoh. An id-based cryptosystem based on the discrete logarithm problem. *IEEE J.Sel. A. Commun.*, 7(4):467–473, May 1989.

[TMTM69] Jeffrey Travers, Stanley Milgram, Jeffrey Travers, and Stanley Milgram. An experimental study of the small world problem. *Sociometry*, 32:425–443, 1969.

[UHHC11] Alexander Ulrich, Ralph Holz, Peter Hauck, and Georg Carle. Investigating the openpgp web of trust. In *Proceedings of the 16th European conference on Research in computer security*, ESORICS'11, pages 489–507, Berlin, Heidelberg, 2011. Springer-Verlag.

[VOL12] The voltage ibe toolkit, 2012. http://www.voltage.com/technology/securing-applications-using-IBE.htm - Abgerufen am 20.12.2012.

[Zim99] P.R. Zimmerman. *The Official PGP User's Guide.* DIANE Publishing Company, 1999.