universität wien

Diplomarbeit

Titel der Diplomarbeit:

Digitale Signaturen und Zertifikate. Von den mathematischen Hintergründen bis zur Anwendung.

angestrebter akademischer Grad

Magister der Naturwissenschaften (Mag. rer.nat.)

Verfasser:	Simon Poledna
Matrikel-Nummer:	9071439
Studienrichtung:	Unterrichtsfach Mathematik
Betreuer:	Privatdoz. Ass.-Prof. Dr. Bernhard Lamel

Wien, am 18.05.2011

Einleitung

In meiner Diplomarbeit „Digitale Signaturen und Zertifikate. Von den mathematischen Hintergründen bis zur Anwendung.“ liegt das Hauptaugenmerk auf den Möglichkeiten der Identitätsbestimmung mit Hilfe digitaler Hilfsmittel. Aufgrund des Fortschreitens der Digitalisierung und der zunehmenden Unwichtigkeit von Grenzen in der digitalen Welt ist es in den meisten Fällen nicht mehr möglich seinen Kommunikationspartner persönlich zu kennen oder diesen gar zu treffen. Deshalb ist es von großer Wichtigkeit sich der Identität des Kommunikationspartners sicher zu sein. Dies geschieht meist mit Hilfe von digitalen Zertifikaten oder Signaturen.

Meine Diplomarbeit ist in verschiedene Kapitel gegliedert.
Im ersten Kapitel „Kryptographie“ werde ich mich mit Kryptographie und ihren mathematischen Hintergründen befassen. Kryptographie ist ein unerlässlicher Bestandteil digitaler Identitäten. Ich werde in diesem Kapitel die symmetrischen Verfahren nicht behandeln, da diese zwar beim Nachrichtenaustausch verwendet werden (z.B. als Session Key), für die Erstellung digitaler Signaturen und Zertifikate sind diese jedoch nicht von Bedeutung. Auf die Funktionsweise und Hintergründe des RSA-Verfahrens werde ich besonders eingehen, da dieses heutzutage das am weitesten verbreitete Verfahren ist.

In Kapitel 2 „Hashfunktionen und digitale Signaturen“ werde ich auf Hashfunktionen eingehen. Diese werden zur Bestätigung der Unversehrtheit beim Nachrichtenaustausch benötigt. Als konkretes Beispiel hierfür werde ich die Hashfunktion SHA-256 im Detail behandeln. Außerdem werde ich in Kapitel 2 digitale Signaturen einführen und deren Funktionsweise mit dem RSA-Verfahren erläutern.

Im darauf folgenden Kapitel 3 „Public Key Infrastruktur (PKI)“werden PKI (Public Key Infrastuctures) behandelt werden. Diese sind eine konkrete Implementierung für digitale Signaturen. Es gibt verschiedene Arten von PKIs. Ich werde auf die wichtigsten eingehen.

Kapitel 4 „Digitale Zertifikate“ werden digitale Zertifikate erläutern. Diese sind eine Art Weiterentwicklung der digitalen Signatur, bei denen meist staatlich akkreditierte Organisationen als vertrauenswürdiger Dritter fungieren. Deren Aufgabe ist es die Identität der Benutzer zu bestätigen und die öffentlichen Schlüssel zu verwalten und zu verteilen. Ich werde hier auch auf die genaue Gesetzeslage in Österreich eingehen.

In Kapitel 5 „Schlüsselmanagment“ wird kurz auf das so genannte Schlüsselmanagement eingegangen. Dieses umfasst die Erzeugung, Speicherung, Vernichtung, den Austausch und die Länge von Schlüsseln.

Im abschließenden Kapitel 6 „Hilfsmittel zur digitalen Signierung“ werden Hilfsmittel zur digitalen Signierung behandelt. Diese haben die Aufgabe den Benutzern die Anwendung von digitalen Signaturen und Zertifikaten zu erleichtern, indem sie eine sichere Umgebung zur Schlüsselverwendung anbie-

ten. Ich werde hauptsächlich Smartcards behandeln. Hierbei werden Bestandteile und Sicherheit im Vordergrund stehen. Außerdem wird die Bürgercard als Beispiel angeführt, da diese für alle Österreicher zugänglich ist.

Am Ende meiner Diplomarbeit werden in einem Glossar noch einige Begriffe die verwendet wurden erklärt.

P.S.: In dieser Arbeit verwende ich die grammatikalisch männliche Form gleichermaßen für Männer und Frauen.

Introduction

In my diploma „Digitale Signaturen und Zertifkate. Von den mathematischen Hintergründen bis zur Anwendung.“ the main focus lies on the possibilities to determine ones identity with digital means. Because of the advancement of digitalisation and the growing insignificance of boarders in the digital world it is not always possible to know or even meet ones associate in person. Because of that it is very important to be certain of the identity of ones associate in the digital world. This is realized through digital Certificates or signatures.

My diploma is structured in different chapters.
In chapter one „Kryptographie“ I am going to cover the idea and mathematical backgrounds of cryptography. It is an essential part of digital identities. I am not going to cover symmetric cryptography, although it is used in message swapping (i.e. session keys), it is not important for the creation of digital signatures and certificates. The RSA-Method, its functions and its background are the focus of this chapter because the RSA-Method is the most common and most wide spread today.

In chapter two „Hashfunktionen und digitale Signaturen“ Hashfunctions will be covered. These are needed to proof the integrity of a message. As a practical example the method SHA-256 will be explained in detail. Aside from

that digital signatures will be introduced and their functionality within the RSA-Method.

In the following chapter „Public Key Infrastruktur (PKI)“ is going to contain PKIs. These are an implementation of digital signatures. Different types of PKI will be mentioned.

Chapter four „Digitale Zertifkate“ will define digital certificates, which are an improvement of digital signatures. Certificates use accredited Organisation which function as a trusted third party. There role is to provide the identity of a user and his public key. I will also treat the laws in Austria.

In chapter five „Schlüsselmanagment“ the topic is managment of security keys. This includes creation, storage, destruction, swapping and the lenght of keys.

In the final chapter six „Hilfsmittel zur digitalen Signierung“ implements of digital signatures will be covered. These have the function to help the user utilizing the digital signatures and certificates. They offer a secure environment for the usage of the key. The focus of this chapter will be smartcards, its properties and its security features. As an example I will refer to Bürgercards because these are available to every Austrian resident.

At the end of my Diploma there is going to be a Glossary in which some of the used terms will be illustrated.

Inhaltsverzeichnis

Kapitel 1

Kryptographie

1.1 Kryptographie allgemein

Da es zum Thema Kryptographie schon unzählige Definitionen gibt habe ich mich dazu entschieden einige aufzulisten:

> Kryptographie: Anwendung mathematischer Verfahren, um Techniken und Algorithmen zu entwickeln, welche die Sicherheit der Daten schützen. Sicherheit umfasst in diesem Zusammenhang bes. Vertraulichkeit, Integrität und die Authentifizierung (Methoden zur Überprüfung der Identität des Senders übermittelter Daten, der z.B. an der Tätigkeit eines Zahlungssystems beteiligt ist, und zur Bestätigung, dass eine Nachricht bei der Übermittlung nicht verändert wurde). [1]
>
> Kryptographie: Verfahren zum Verschlüsseln zum Beispiel elektronischer Post. Es macht die übermittelten Daten für Unbefugte

[1] Gabler Wirtschaftslexikon http://wirtschaftslexikon.gabler.de/Definition/kryptographie.html

unleserlich. [2]

Kryptographie: Die Kryptographie ist die Wissenschaft zur Erforschung und Realisierung von Verfahren zur Verschlüsselung bzw. Entschlüsselung von Daten, bei denen entweder das Verschlüsselungsverfahren oder bei Anwendung einheitlicher Verschlüsselungsverfahren, die verwendeten Schlüsselbegriffe geheim gehalten werden. Durch Ändern, Vertauschen oder Hinzufügen von Zeichen nach bestimmten Regeln wird ein Klartext in einen Schlüsseltext verwandelt und umgekehrt; anwendbar bei der Speicherung von Daten und der Datenübertragung. Wirksamstes Mittel des Datenschutzes, um Informationen, die in falsche Hände gelangt sind, wertlos zu machen. [3]

Kryptographie ist die mathematische Wissenschaft, die sich mit Vertraulichkeit, Datenintegrität und Authentifikation befasst.

Vertraulichkeit: Niemand ausser die dazu berechtigten Personen sollen die Daten lesen können.

Datenintegrität: Niemand soll unbemerkt Daten verändern, hinzufügen, löschen oder ersetzen können.

Authentifikation: Die beteiligten Personen sollen wissen, mit wem sie kommunizieren. Der Empfänger soll wissen, von wem die Information stammt, wann sie erzeugt wurde, ... [4]

Kryptographie ist die Wissenschaft, die sich mit der Verschlüsselung von Information beschäftigt. Kryptographie existiert seit den ersten frühen Hoch-

[2] Kleines Computerlexikon http://www.ksta.de/html/artikel/1192572081165.shtml

[3] IT Wissen http://www.itwissen.info/definition/lexikon/Kryptografie-cryptography.html

[4] Matthias Web http://www.siug.ch/help/SIUG-Kryptoeinfuehrung.html

kulturen, Ägyptens, Indiens und Mesopotamiens. Damals waren die Verfahren noch wesentlich einfacher. Zum Beispiel wurden Einkerbungen in Holzplatten so versteckt indem man sie mit Wachs auffüllte und die gesamte Platte mit Wachs überzog. Eine andere Methode waren kleine Löcher in Buchstaben auf Papyrusrollen die erst im Gegenlicht sichtbar wurden.
Nach heutigem Wissensstand tauchten die ersten militärisch genutzten kryptografischen Systeme im 5. Jahrhundert vor Christus im antiken Griechenland auf. Spartanische Befehlshaber verwendeten so genannte Skytale um ihren Truppen Botschaften bzw. Befehle zu übermitteln. Dabei wurde Papyrus um einen Holzstab gewickelt und die Nachricht darauf geschrieben. Der Rest des Papyrus wurde ebenfalls beschrieben. Dieser Stab hatte einen fest definierten Durchmesser. Der Empfänger der Nachricht musste im Besitz einer identischen Vorrichtung sein um die Nachricht lesen zu können. Der Durchmesser des Stabs kann als Schlüssel betrachtet werden. Der entscheidende Vorteil dabei war, dass ein Feind, der in den Besitz der Nachricht kam, diese ohne das Wissen über den Durchmesser des Stabes nicht lesen konnte.
Doch mit dem Aufkommen und den Fortschritten in der Technik hat sich das Bild der Kryptografie komplett geändert. Heute funktionieren Verschlüsslungen auf der Basis komplizierter mathematischer Verfahren und sind mit erheblichem Rechenaufwand verbunden, der ohne Computereinsatz undenkbar wäre.

1.2 Asymmetrische Verfahren

1.2.1 Kryptografisches System

Bevor ich mit den asymmetrischen Verfahren beginnen kann, ist es notwendig einige grundlegende Begriffe einzuführen. Ich werde mich hierbei an die

Definition nach Eckert (Literaturverzeichnis Punkt 1) halten.

Definition Kryptografisches System:
Gegeben seine zwei endliche Alphabete A_1 und A_2. Ein *kryptografisches System* oder kurz *Kryptosystem* $\mathcal{KS}$ ist gegeben durch ein Tupel

$$\mathcal{KS} = (f, \mathcal{M}, \mathcal{C}, EK, DK, E, D)$$

mit

1. der (nicht leeren) endlichen Menge von Klartexten $\mathcal{M} \subseteq A_1^*$, wobei A_1^* die Menge aller Worte über dem Alphabet A_1 beschreibt
2. der (nicht leeren) endlichen Menge von Krypto bzw. Chiffretexten $\mathcal{C} \subseteq A_2^*$
3. der (nicht leeren) Menge von Verschlüsselungsschlüsseln EK
4. der nicht leeren Menge von Entschlüsselungsschlüsseln DK sowie einer Bijektion $f : EK \to DK$. Die Bijektion assoziiert zu einem Verschlüsselungsschlüssel $K_E \in EK$ einen dazu passenden Entschlüsselungsschlüssel $K_D \in DK$, dh $f(K_E) = K_D$
5. dem injektiven Verschlüsselungsverfahren E: $\mathcal{M}$ x EK $\to \mathcal{C}$ und
6. dem Entschlüsselungsverfahren D: $\mathcal{C}$ x DK $\to \mathcal{M}$ mit der Eigenschaft, dass für zwei Schlüssel $K_E \in EK, K_D \in DK$ mit $f(K_E) = K_D$ gilt:

$$\text{für alle } M \in \mathcal{M} : D(E(M, K_E), K_D) = M.$$

1.2.2 Anforderung an asymmetrische Verfahren

Die Kryptographie wird grob in zwei Kategorien eingeteilt: *Symmetrische* und *asymmetrische* Kryptographie.
Bei symmetrischer Kryptographie wird derselbe Schlüssel zum Ver- und Entschlüsseln verwendet. Sie hat den entscheidenden Vorteil, dass symmetrische Verfahren wesentlich schneller zu berechnen sind als asymmetrische. Es gibt jedoch ein großes Problem mit dem Schlüsseltransport. Um verschlüsselte Nachrichten austauschen zu können, muss man bereits einen Schlüssel ausgetauscht haben. Dies ist in der Praxis sehr kompliziert. Symmetrische Verfahren werden oft angewendet da diese wesentlich schneller sind als asymmetrische Verfahren. Dies geschieht aber meist erste wenn ein asymmetrisches Verfahren benutzt wurde um einen Schlüsselaustausch vorzunehmen.
Ich werde in meiner Arbeit auf die symmetrische Kryptographie nicht im Detail eingehen.

Bei asymmetrischer Kryptographie benutzt man ein sich ergänzendes Schlüsselpaar, um die Problematik des Schlüsselaustauschs zu lösen. Einen der beiden kann man veröffentlichen, den anderen hält man geheim (public key K_E, private key K_D). Was mit dem Schlüssel K_E verschlüsselt wird, soll NUR mit dem dazugehörigen Partner K_D entschlüsselt werden können (und umgekehrt).
Ein System, dass diese Eigenschaften besitzt nennt man ein Public-Key-System oder ein asymmetrisches Kryptosystem.

1.2.3 Public-Key-System: Formale Definition

Formal gesehen besitzt ein Public-Key-System folgende Eigenschaften:

1. Das Schlüsselpaar (K_E, K_D) muss effizient zu erzeugen sein, wobei für solche Paare gelten muss:

   - $K_D = f(K_E)$
   - $\forall M \in \mathcal{M}$: $D(E(M, K_E), K_D) = M$
   - K_E kann öffentlich bekannt gegeben werden

2. Die Verschlüsselungsfunktion E und die dazugehörige Entschlüsselungsfunktion D sind effizient zu berechnen.

3. K_D ist aus der Kenntnis von K_E nicht mit vertretbarem Aufwand berechenbar.

4. Wenn $\mathcal{M} = \mathcal{C}$ und für alle $M \in \mathcal{M}$, $E(D(M, K_D), K_E) = D(E(M, K_E), K_D) = M$ gilt so ist das Kryptosystem auch zum Erstellen von digitalen Signaturen geeignet. (Siehe Kapitel 3 Digitale Signaturen)

Ich werde nun etwas genauer auf die Definition eingehen:

In der Definition wird gefordert, dass das Schlüsselpaar funktional zusammen hängt (Punkt 1). Durch $K_D = f(K_E)$ lässt sich ein eindeutiger Zusammenhang zwischen Klartext und Geheimtext - und umgekehrt - herstellen.
Die zugrunde liegende Funktion sollte leicht ausführbar, aber ohne die nötigen Informationen nicht umkehrbar sein. Salopp könnten wir fordern: In eine Richtung soll es schnell gehen, in die andere hingegen gar nicht (Punkt 3).
Theoretisch ist das leider nicht möglich, da mit genügend Zeit und Rechenleistung zu jeder bekannten Funktion eine Umkehrung gefunden werden kann (Das Durchlaufen des gesamten Schlüsselraums wird als Brute-Force-Attack bezeichnet). Deshalb ist es in der Praxis schon ausreichend, wenn die Berechnung von K_D aus K_E so langwierig und mühsam ist, dass es auch mit den

schnellsten Computern nicht in akzeptabler Zeit erledigt werden kann.
Solche Funktionen finden sich in der Mathematik, sie werden als Einweg-Funktionen bezeichnet. Diese machen ein Errechnen des privaten Schlüssels aus der Kenntnis des öffentlichen Schlüssels mit vertretbarem Aufwand unmöglich. Dies ist bei einer injektiven Funktion $f : X \to Y$ dann der Fall, wenn für $x \in X$ der Funktionswert $f(\mathrm{x}) = \mathrm{y}$ leicht zu berechnen ist, z.B. das Produkt zweier Primzahlen, $\mathrm{y} = f(\mathrm{x})$ das Urbild x jedoch mit keinem effizienten Verfahren berechenbar ist, das Produkt in seine Faktoren aufspalten. Das heisst, dass sich mit vertretbarem Aufwand keine Lösung dafür finden lässt. Weiterns existieren Einweg-Funktionen mit Falltür. Dies sind Einweg-Funktionen, welche bei Kenntnis bestimmter Werte leicht umkehrbar sind (z.B. privater Schlüssel KD).
D(E(M,KE),KD) = M (Punkt 4) bedeutet, dass alle Klartexte M der Klartextmenge $\mathcal{M}$ mit KE zu C = E(M,KE) verschlüsselt werden. Diese ergeben bei Anwendung von KD wiederum M = D(C,KD).

1.2.4 Einwegfunktionen

1.2.4.1 Definition Einwegfunktion

Definition. *Einwegfunktionen:*
Eine injektive Funktion $f : X \to Y$ heisst Einwegfunktion, wenn:

1. *für alle x in X der Funktionswert f(x) effizient berechenbar ist und*
2. *wenn es kein effizientes Verfahren gibt, um aus einem Bild $y = f(x)$ das Urbild x zu berechnen.*

1.2.4.2 Diskrete Logarithmusfunktion

In der Modul-Arithmetik finden sich reichlich Einwegfunktionen. Die diskrete Exponentialfunktion $y = g^a \mod p$ ist beispielsweise leicht zu berechnen. Ihre Umkehrfunktion, die diskrete Logarithmusfunktion $a = \log_g(y) \mod p$ ist jedoch nur äußerst schwierig zu ermitteln. Die ersten zwei Funktionsgraphen in der Abbildung zeigen zu Vergleichszwecken die reelle Exponentialfunktion und deren Umkehrfunktion, welche die reelle Logarithmusfunktion ist. Beide verlaufen stetig. Der dritte Funktionsgraph stell die diskrete Exponentialfunktion $\alpha = 11^a \mod 23$ dar. Diese macht hingegen unvorhersagbare Sprünge.

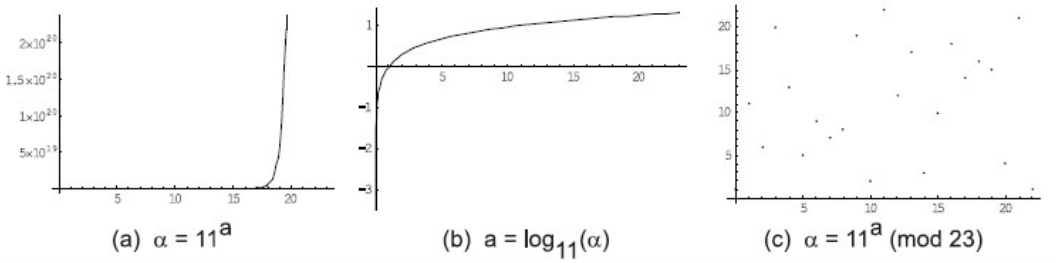

Abbildung 1.1: Bsp.: Exponential & Logarithmusfunktion

Wir können daher sehr leicht aus einer gegebenen Zahl a den Wert $\alpha = 11^a \mod 23$ berechnen. Haben wir aber nur das Ergebnis α, können wir a nur erraten! Die Umkehrfunktion $a = \log_{11} \alpha \mod 23$ verläuft im endlichen Zahlenraum von $\mathbb{Z}_{23}$ nämlich nicht stetig. Eine systematische Annäherung an die richtige Lösung ist mit einem Verfahren ähnlich dem Newton-Verfahren nicht möglich.

Beispiel der Unstetigkeit:

$y = 7^a \mod 11$; a durchläuft den Zahlenraum $\mathbb{Z}_{11}$.

So ist beispielsweise $7^5 = 16807 = 10 \mod 11$.

a	0	1	2	3	4	5	6	7	8	9	10
$y = 7^a \mod 11$	1	7	5	2	3	10	4	6	8	8	1

1.2.4.3 Primfaktorzerlegung

Hier ist die Einwegfunktion $E(p, q) = p * q$ mit Falltür p und q gegeben. Seien N die Werte der Funktion E, wobei p und q sehr große Primzahlen sind. p und q werden geheim gehalten und N wird veröffentlicht. Mit N können Nachrichten durch E verschlüsselt werden. Zum Entschlüsseln mit der Umkehrfunktion $D = E^{-1}$ wird p und q benötigt (konkrete Implementierung RSA). Der Trick an der Sache ist, dass es nicht ohne weiteres möglich ist, aus N die beiden Faktoren p und q zu ermitteln. Ist N groß genug, ist dies praktisch unmöglich. bei 1024 Bit würde dies Beispielsweise bereits 10^{11} MIPS Jahre dauern.

1.2.5 RSA-Verfahren

Das RSA-Verfahren entstand 1977 aus den Bemühungen von Rivest, Shamir und Adleman einen Fehler in dem Algorithmus von Diffie und Hellman zu finden. Dies gelang nicht. Doch im Zuge dieser Forschungen wurde das RSA-Verfahren entwickelt. Es basiert auf dem aktuellen Wissensstand, dass die Zerlegung einer großen Zahl in ihre Primfaktoren, sehr aufwendig ist. Allerdings ist das Erzeugen einer Zahl durch Multiplikation zweier Primzahlen recht einfach.

1.2.5.1 Rechenvorgang allgemein

1. Alice wählt zufällig zwei verschiedene Primzahlen p und q
2. Sie berechnet N = p*q und $\varphi(N) = (p - 1) * (q - 1)$

3. Dann wählt sie $e \in \{2, ..., \varphi(N) - 2\}$ mit ggT(e,$\varphi(N)$) = 1.

4. Alice bestimmt d mit e*d $\equiv$1 mod$\varphi(N)$.

5. Schließlich veröffentlicht sie $K_E := (N, e)$ als ihren öffentlichen Schlüssel und hält das Paar $K_D := (N, d)$ geheim.

6. Die Werte p, q und $\varphi(N)$ löscht Alice.

7. Angenommen Bob möchte an Alice die Nachricht M mit $M \in \{0, ..., N - 1\}$ schicken:
   Bob berechnet C := M^e mod N und schickt C an Alice.

8. Alice kann dann die Nachricht M wie folgt entschlüsseln:
   Alice berechnet M := C^d mod N.

1.2.5.2 Mathematische Hintergründe des RSA-Verfahrens

Zunächst werden wir Teilbarkeit in $\mathbb{Z}$ zeigen.

(i) Für jedes n $\neq$ 0 gilt $n|0$ und $n|n$.

(ii) Gilt $m|n$, so auch $-m|n$ und $m|-n$.

(iii) Für alle n gilt $1|n$.

(iv) Aus $m|n$ und n $\neq$ 0 folgt $|m| \leq |n|$.

(v) Aus $n|1$ folgt entweder n = 1 oder n = -1.

(vi) Aus $m|n$ und $n|m$ folgt entweder n = m oder n = -m.

(vii) Aus $\ell|m$ und $m|n$ folgt $\ell|n$.

(viii) Bei $\ell \neq 0$ sind $m|n$ und $\ell m|\ell n$ gleichbedeutend.

(ix) Gelten $m|n_1$ und $m|n_2$, so auch $m|(\ell_1 n_1 + \ell_2 n_2)$ bei beliebigen $\ell 1$, $\ell 2$.

(x) Gelten $m_1|n_1$ und $m_2|n_2$, so auch $m_1 m_2|n_1 n_2$.

Seien nun $n_1, ..., n_k$ ganze Zahlen. Gefragt wird nach allen ganzen $d \neq 0$ mit $d|n_1, ..., d|n_k$, mit anderen Worten, nach den gemeinsamen Teilern aller $n_1, ..., n_k$. Ist d ein derartiger gemeinsamer Teiler, so hat -d nach „Teilbarkeit in $\mathbb{Z}$(ii)“ dieselbe Eigenschaft, weshalb in Zukunft die Beschränkung auf positive gemeinsame Teiler ausreicht, zu denen übrigens die Eins nach „Teilbarkeit in $\mathbb{Z}$(iii)“ immer gehört. Weiter kann künftig vorausgesetzt werden, dass nicht alle $n_1, ..., n_k$ Null sind; andernfalls liegt nach „Teilbarkeit in $\mathbb{Z}$(i)“ die triviale Situation vor, wo jede von Null verschiedene ganze Zahl gemeinsamer Teiler der $n_1, ..., n_k$ ist. Nach „Teilbarkeit in $\mathbb{Z}$(iv)“ ist dann klar, dass jeder positive gemeinsame Teiler d von $n_1, ..., n_k$ der folgenden Ungleichung genügt:

$$\mathrm{d} \leq \min \{|n_i| : i = 1, ..., k \text{ mit } n_i \neq 0\}.$$

Es ist ersichtlich, dass bei ganzen, nicht sämtlich verschwindenden $n_1, ..., n_k$ die Menge aller positiven gemeinsamen Teiler eine nicht leere endliche Menge ist. Das größte dieser Menge heißt **größter gemeinsamer Teiler (kurz: ggT):** der n1,..., nk.

Man benötigt für die effiziente Bestimmung des privaten Schlüssels den erweiterten Euklidischen Algorithmus um den ggT(e, $\varphi(N)$) als Linearkombination von e und $\varphi(N)$ darzustellen.

Satz: erweiterter euklidischer Algorithmus. *Der erweiterte euklidischen Algorithmus bestimmt den größten gemeinsamen Teiler zweier natürlicher Zahlen a und b, ggT(a,b). Außerdem werden noch zwei ganze Zahlen s und t bestimmt, die die Gleichung ggT(a,b) = s*a + t*b erfüllen.*

Beweis. Gegeben sind $a, b \in \mathbb{N}$: $a = q_1 b + r_1$ mit $q_1, r_1 \in \mathbb{N}$ und $r_1 < b$ (Division mit Rest in $\mathbb{N}0$)
$\rightarrow r_1 \neq 0, r_1 \in \mathbb{N} : b = r_1 q_2 + r_2$ mit $0 \leq r_2 < r_1$
oder $r_1 = 0$ (Verfahren endet)
$\rightarrow r_2 \neq 0, r_2 \in \mathbb{N} : r_1 = r_2 q_3 + r_3$ mit $0 \leq r_3 < r_2$
oder $r_2 = 0$ (Verfahren endet)
Wegen $b > r_1 > r_2 > ... \geq 0$ muss das Verfahren abbrechen,
das heißt $\exists n \in \mathbb{N}$ mit $r_n \neq 0, r_n + 1 = 0$
Die letzten beiden Gleichungen lauten:
$r_{n-2} = r_{n-1} q_n + r_n$ mit $0 < r_n < r_{n-1}$
$r_{n-1} = r_n q_{n+1} + r_{n+1}$ wobei $r_{n+1} = 0$
$\rightarrow r_n$=ggT(a,b)
Wenden wir dieses Verfahren anschließend rückwärts an, erhalten wir eine Darstellung des ggT als Linearkombination. □

Definition. ***Teilerfremd:***
Ganze Zahlen $n_1, ..., n_k$, nicht alle Null, heißen teilerfremd, wenn ihr ggT gleich Eins ist.

Definition. ***Eulersche Phifunktion ($\varphi(n)$):***
$\varphi(n)$ ist die Anzahl der natürlichen, n nicht übersteigenden Zahlen, die zu n teilerfremd sind.

Definition. ***Kongruenz und elementare Eigenschaften:***
Seien $m \neq 0$, a, b ganze Zahlen. Man nennt ***a kongruent (zu) b modulo m*** *genau dann, wenn $m|(a-b)$ gilt; man schreibt dies als*

$$a \equiv b \mod m.$$

Weiters gilt:

- $a \equiv a \mod m$,
- $a \equiv b \mod m \Rightarrow b \equiv a \mod m$,
- $a \equiv b \mod m, b \equiv c \mod m \Rightarrow a \equiv c \mod m$.

Hieraus sieht man bereits, das die Relation ***kongruent modulo m*** *eine Äquivalenzrelation auf $\mathbb{Z}$ ist. Damit zerlegt sie $\mathbb{Z}$ in disjunkte Klassen, die sogenannten* ***Restklassen modulo m****.*

Definition. ***Restklassenringe:***
Ist m eine natürliche Zahl, so definiert man die (von m abhängige) Abbildung ψ von $\mathbb{Z}$ in die Menge der Restklassen modulo m dadurch, dass man jedem $a \in \mathbb{Z}$ diejenige Restklasse $\psi(a)$ zuordnet, der a angehört. Danach ist $a \equiv a' (\mod m) \Leftrightarrow \psi(a) = \psi(a')$ klar; $\psi(\mathbb{Z})$ hat genau m Elemente. In der Menge $\psi(\mathbb{Z})$ aller Restklassen modulo m definiert man sodann zwei Verknüpfungen $+$ und $\cdot$ durch folgende Vorschriften:

$$\psi(a) + \psi(b) := \psi(a+b) \text{ bzw. } \psi(a) \cdot \psi(b) := \psi(ab)$$

für alle a, $b \in \mathbb{Z}$; dabei deutet das $+$ in $a + b$ auf die gewöhnliche Addition in $\mathbb{Z}$ hin.
Das diese Verknüpfungen des Restklassenrings wohldefiniert sind liegt an den Eigenschaften (ix) und (x) der Teilbarbeit in $\mathbb{Z}$.

Da die Abbildung ψ des Integritätsrings $\mathbb{Z}$ auf $(\psi(\mathbb{Z}), +, \cdot)$ ein Homomorphismus ist, ist $(\psi(\mathbb{Z}), +, \cdot)$ ein kommutativer Ring, der ***Restklassenring modulo m*** *heißt.*

Definition. ***prime Restklasse:***
Ist a eine ganze, zu m teilerfremde Zahl, so sind für alle ganzen t auch m und a+tm zueinander teilerfremd. Die Eigenschaft einer ganzen Zahl, zum Modul m teilerfremd zu sein, kommt also jedem Element der Restklasse modulo m zu, in der die Zahl liegt. Eine Restklasse modulo m, deren jedes Element zu m teilerfremd ist, heißt eine ***prime*** *(oder auch teilerfremde)* ***Restklasse modulo m****.*

Definition. ***Gruppe***
$G \neq \emptyset$ mit einer Operation $\cdot$; heisst ***Gruppe*** *wenn gilt:*

1. *Assoziativgesetz*
2. *$\exists$ neutrales Element e, $a \cdot e = e \cdot a = a \; \forall a \in G \leftrightarrow$ eindeutig*
3. *zu jedem $a \in G \; \exists$ ein inverses $\overline{a} : a\overline{a} = \overline{a}a = e \leftrightarrow$ eindeutig*

Gilt zusätzlich das Kommudativgesetz, so heißt G ***kommutative oder Abelsche Gruppe***

Satz zur primen Restklassengruppe $\mathbb{Z}_m^*$. *Die Menge der primen Restklassen modulo m bildet bezüglich der in Restklassenringe erklärten Verknüpfung $\cdot$ eine abelsche Gruppe, die sogenannte* ***prime Restklassengruppe $\mathbb{Z}_m^*$ modulo m****.*

Beweis. Sind a, b ganz und zu m teilerfremd, so ist auch ab zu m teilerfremd. Anders gesagt: Sind $\phi(a), \phi(b)$ prime Restklassen modulo m, so ist auch $\phi(a) \cdot \phi(b)$ eine prime Restklasse modulo m. Daher ist die im Satz genannte Menge gegenüber $\cdot$ abgeschlossen; außerdem gelten bezüglich dieser Verknüpfung das Assoziativ- und Kommutativgesetz.
a und m sind teilerfremd. Deshalb $\exists s, t \in \mathbb{Z}$ nach dem erweiterten euklidischen Algorithmus für die gilt: $sa + tm = 1 \Rightarrow \phi(a) \cdot \phi(s) = \phi(1)$

□

Definition. ***Ordnung von G*** *(|G|)*
Die Anzahl der Elemente (die Mächtigkeit) einer Gruppe G heisst die ***Ordnung von G*** *und wird mit |G| bezeichnet.*

Definition. *Sein $(G, \cdot)$ Gruppe und $M \subseteq G$. Dann heißt $\bigcap_{U \supseteq M U \subseteq G} U$ die von M erzeugte Untergruppe in G, bezeichnet mit $< M >$*
Es gilt: $< M >$ ist Untergruppe von G die m enthält un $< M >$ ist sogar die kleinste Untergruppe von G die M enthält.

Lemma:. *Sei $(G, \cdot)$ eine Gruppe und $a \in G$ fest. Dann gilt $< a >= \{a^k; k \in \mathbb{Z}\}$*

Beweis. z.z.: $\{a^k; k \in \mathbb{Z}\}$ ist die kleinste Untergruppe, die a enthält.
Sie $U \leq G$ mit $a \in U \Rightarrow a \cdot a \in U \Rightarrow a^2 \cdot a \in U$,..., $a^k \in U$ $(k \in \mathbb{N})$ $e = a^0 \in U$; da $a^k \in U \Rightarrow (a^k)^{-1} = a^{-k} \in U$ $(k \in \mathbb{N}) \Rightarrow a^k \in U$ $\forall k \in \mathbb{Z}$; also gilt $\{a^k; k \in \mathbb{Z}\} \subseteq U$

□

Definition. ***Ordnung von a o(a)***
Sei $(G, \cdot)$ eine Gruppe und $a \in G$. Dann heisst $| < a > |$ die ***Ordnung von***

***a**; bezeichnet wird sie mit o(a).*

Definition. ***Rechtsnebenklasse***

Sei $(G, \cdot)$ eine Gruppe und $U \leq G$, $g \in G$ fest.

Dann heißt $Ug := \{ug;\ u \in U\}$ ***Rechtsnebenklasse*** *nach U.*

Bemerkung: Sei $U =< n >$ mit $< n >= \{nk;\ k \in \mathbb{Z}\}$ sei weiters $g = 1$. Dann ist $U + g := \{nk + 1;\ k \in \mathbb{Z}\}$

Lemma:. *1. $Ug = Uh$ gilt genau dann, wenn $gh^{-1} \in U$ bzw. genau dann, wenn $hg^{-1} \in U$*

2. Rechtsbnebenklassen bilden eine Partition von G.

Beweis. 1. Ug = Uh $\Rightarrow \{ug; u \in U\} = \{vh; v \in U\}$ u=e liefert $g \in Ug$ $\Rightarrow \exists v \in U$ mit g=vh $\Rightarrow gh^{-1} = v \in U \Rightarrow v^{-1} = (gh^{-1})^{-1} = hg^{-1} \in U$

umgekehrt: sei $gh^{-1} \in U$ d.h. $\exists w \in U$ mit $gh^{-1} = w \Rightarrow g = wh \Rightarrow ug = (\underbrace{uw}_{\in U})h \in Uh$, d.h. $Ug \subseteq Uh$

g =wh liefert $h = w^{-1}g$ ($v \in U$ beliebig) $vh = (\underbrace{vw^{-1}}_{\in U})g \in Ug$, d.h. $Uh \subseteq Ug$

also gilt Ug = Uh

analog für den Fall $hg^{-1} \in U$

2. Sei $g \in G$ beliebig. Da $g \in Ug$ $[g = \underbrace{e}_{\in U} g \in Ug]$ liegt jedes $g \in Ug$ in einer Rechtsnebenklasse.

weiters: für 2 Klassen Ug, Uh gilt: entweder $Ug \cap Uh = \emptyset$ oder seine Ug, Uh nicht elementfremd, also sei $x \in G \in Ug \cap Uh$: d.h. $\exists u \in U$,

$v \in U$ mit $x = ug = vh \Rightarrow gh^{-1} = u^{-1}v \in U \Rightarrow Ug = Uh$

□

Satz:. *Sei $(G, \cdot)$ eine Gruppe und $U \leq G$. Dann gilt:*
$|Ug| = |U| = |gU| \ \forall g \in U$

Beweis. Definiere $\alpha : U \to Ug$ durch $\alpha(u) := ug$
injektiv: sei $\alpha(u) = \alpha(v)$, d.h. ug = vg $\Rightarrow$ u=v
α ist surjektiv, da jedes Element von Ug die Gestalt ug hat.
$\Rightarrow \alpha$ ist bijektiv.

□

Die Anzahl der Rechtsnebenklassen wird als Index von U in G bezeichnet. Dies schreibt man als [G:U]

Satz von Lagrange. *1. Sei $(G, \cdot)$ eine Gruppe und $U \leq G$ dann gilt:*
$|G| \cdot [G : U] = |G|$. *Insbesondere folgt für endliche Gruppen G:*
$|U| \,||G|$ *und* $[G : U]||G|$

2. G endlich und $a \in G$ beliebig, so folgt $o(a)||G|$; insbesonders gilt $a^{|G|} = e, \forall a \in G$

Beweis. 1. Rechtsnebenklassen bilden eine Partition von G: $G = \bigcup Ug$
$\Rightarrow |G| = \sum \underbrace{|Ug|}_{=|U|} = |U| \sum 1$ (Wobei $\sum 1$ die Anzahl der verschiedenen Rechtsnebenklasse =[G:U] ist) $= |U| \cdot [G : U]$

2. Sei $a \in G$ Dann ist $o(a) = | < a > |||G|$ (nach 1); insbesonders folgt $|G| = o(a) \cdot t$ mit $t \in \mathbb{N}$.

$$a^{|G|} = a^{o(a)\cdot t} = (\underbrace{a^{o(a)}}_{=e})^t = e^t = e$$

□

Um zu zeigen, dass das RSA-Verfahren wirklich funktioniert (Im speziellen, dass die Ver- und Entschlüsselung wirklich eindeutig ist) benötigen wir den folgenden Satz von Euler.

Satz von Euler. *Seien a und n teilerfremde Zahlen, $\varphi(n)$ die Anzahl der zu n teilerfremden Zahlen. Dann gilt $a^{\varphi(n)} \equiv 1 \mod n$*

Beweis. Ist G endliche Gruppe und a $\in$ G dann folgt aus dem Satz von Lagrange, dass $a^{|g|} = e$ wobei e das Einselement ist. Speziell ist G Element der multiplikativen Gruppe $\mathbb{Z}_n^*$ dh. gilt $|g| = \varphi(n)$ □

1.2.5.3 Beweis der Gültigkeit des RSA-Verfahrens

Verschlüsselung: $C := M^e \mod N$
Entschlüsselung: $M := C^d \mod N$
mit $N = p * q, \varphi(N) = (p-1) * (q-1)$
und $e * d \equiv 1 \mod \varphi(N) \rightarrow e * d = k * \varphi(N) + 1 = k * (p-1) * (q-1) + 1$
Wenn man die Verschlüsselung $C := M^e \mod N$ mit d potenziert erhält man:

$$
\begin{aligned}
C^d &= (M^e)^d \mod N \\
&= M^{e*d} \mod N \\
&= M^{k*(p-1)*(q-1)+1} \mod N \\
&= M^{k*(p-1)*(q-1)} * M \mod N \\
&= (M^{(p-1)*(q-1)})^k * M \mod N \\
&= (M^{\varphi(N)})^k * M \mod N \\
&= 1^k * M \mod N \rightarrow \\
C^d &= M \mod N \\
M &= C^d \mod N
\end{aligned}
$$

$(M^{\varphi(N)})^k = 1^k$ gilt nach dem Satz von Euler, wenn M zu N teilerfremd ist. Dies ist die große Schwachstelle des RSA Verfahrens und der Grund warum man Primzahlen verwendet, da bei diesen der Fall, dass M und N nicht teilerfremd sind, verschwindend klein ist, da hierfür die Nachricht M einer der Primzahlen, 1 oder N entsprechen müsste.

1.2.5.4 Rechenbeispiel zu dem RSA-Verfahren

Alice wählt zwei Primzahlen, p und q. Sie nimmt zum Beispiel p=7 und q=13.

Jetzt bildet sie daraus $N = p * q = 91$. Im zweiten Schritt wählt Alice eine Zahl e, die teilerfremd zu $\varphi = (p-1) * (q-1) = 72$ sein sollte, beispielsweise $e = 5$.

Somit hat Alice ihren öffentlichen Schlüssel, nämlich (N, e) = (91, 5).

Der private Schlüssel wird einfach aus dem multiplikativen Inversen von e modulo (p-1)(q-1) gebildet: $5 * d = 1 \mod 72$.

Da $5 * 29 = 1 \mod 72$, erhält Alice das Tupel (N, d) = (91, 29) als privaten Schlüssel, wobei sie d geheim hält.

Bob will Alice in weiterer Folge eine Nachricht M schicken: M = 12.

Er verschlüsselt diese Nachricht mit Alices öffentlichem Schlüssel nach der Vorschrift

$C = M^e \mod N = 12^5 \mod 91 = 38$. Bob schickt C = 38 an Alice.

Alice erhält Bobs geheime Nachricht C = 38.

Sie entschlüsselt einfach nach der Vorschrift

$M = C^d \mod N = 38^{29} \mod 91 = 12$.

Eve fängt die Nachricht ab. Sie kennt natürlich auch den öffentlichen Schlüssel von Alice. Das Potenzieren mit e modulo 91 ist jedoch eine Einwegfunktion mit Falltür. Für ein groß genug gewähltes N ist sie daher praktisch nicht umkehrbar. Mit der Geheiminformation d ist die Funktion jedoch leicht umkehrbar, da $M = C^d \mod N$.

Prinzipiell könnte Eve d berechnen, wenn sie $d * e = 1 \mod (p-1)(q-1)$ nach d auflöst. Dazu müsste sie aber erst das ihr bekannte N in dessen Primfaktoren zerlegen. Das ist aber sehr umständlich, was genau die Stärke des RSA-Verfahrens ist.

Kapitel 2

Hashfunktionen

2.1 Idee der Hashfunktion

Das Verschlüsseln eines gesamten Dokuments ist mit sehr hohem Rechenaufwand verbunden. Außerdem ist es in den meisten Fällen gar nicht notwendig den gesamten Inhalt zu verschlüsseln. Es ist lediglich notwendig sicherzustellen, dass der Inhalt nicht verändert wurde.
Dafür wurden die Hashfunktionen entwickelt.

Der Inhalt eines Dokuments wird mit Hilfe einer Hashfunktion in einem Hashwert fixer Länge zusammengefasst. Dieser ist eine Art Repräsentant der Originalinformation. Im Idealfall führt bereits eine kleine Änderung in der Originalinformation zu einer großen Änderung des Hashwertes.
Die Anwendung einer Hashfunktion ist keine Verschlüsselung.

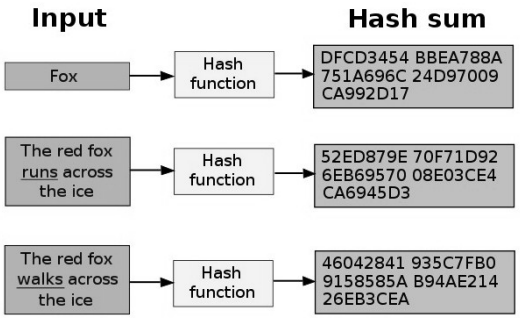

Abbildung 2.1: Bsp.: Hashfunktion

Der Hashwert wird verschlüsselt (meistens auch signiert) und mit der unverschlüsselten Originalinformation mitgeschickt. Der Empfänger kann nun selbst den Hashwert berechnen und mit dem mitgeschickten Hashwert vergleichen. Stimmen diese überein kann er sicher sein, dass die Originalinformation unverändert ist.

2.2 Formale Definition und Qualitätskriterien der Hashfunktion

Definition. *Hashfunktion:*
Eine Hashfunktion ist eine Abbildung,

$h : A^* \to A^l; \quad l \in \mathbb{N};$

die beliebig lange Wörter (mit Buchstaben aus einem endlichen, nicht leeren Alphabet A) auf Wörter der festen Länge l abbildet. Eine solche Funktion kann unter den gegeben Voraussetzungen nicht injektiv sein und ist deshalb

eine Einwegfunktion.

Eine Hashfunktion h muss mindestens folgende Qualitätskriterien erfüllen:

- Preimage resistance: Wenn man im Besitz eines Hashwertes H ist soll man nicht in der Lage sein eine Nachricht m zu finden sodass H = h(m).
- Second preimage resistance: Wenn man in Besitz einer Nachricht m_1 ist soll man nicht in der Lage sein eine weitere Nachricht m_2 ($m_1 \neq m_2$) zu finden mit $h(m_1) = h(m_2)$
- Collision resistance: Man soll nicht in der Lage sein 2 verschiedene Nachrichten m_1 und m_2 zu finden, sodass $h(m_1) = h(m_2)$. So ein Paar würde kryptographische Hashkollision heißen.

Die Kriterien nach denen eine Hashfunktion als sicherer Hashalgorithmus gilt sind wesentlich strenger. Nach FIPS 180-3 (Federal Information Processing Standards) sind das 5 Stück:

- SHA-1, SHA-224, SHA-256 für Nachrichten kürzer als 2^{64} bits
- SHA-384, SHA-512 für Nachrichten kürzer als 2^{128} bits

Im nächsten Teil werde ich auf die genaue Funktionsweise von einem der Algorithmen eingehen.

2.3 Beispiele für Hashfunktionen: SHA-256

Ich werde als Beispiel für die Hashfunktionen die Funktionsweise von SHA-256 näher Erleutern.
SHA-256 ist für eine Nachrichtenlänge von bis zu 2^{64} bits ausgelegt.

Die Nachricht wird in Blöcke eingeteilt (siehe dazu Kapitel 2.3.2 Vorverarbeitung der Nachricht). Als Blockgröße wird 512 bits und als Wortgröße wird 32 bits verwendet.
Die Output Länge beträgt 256 bits.

2.3.1 Benötigte Funktionen und Operatoren

Für die Berechnung des Hashwertes benötigen wir nun noch einige Operatoren und Funktionen:

- $\oplus$ Bitweise XOR Operation (entweder oder).
- >> Right-shift Operation
  x>>n wird gebildet indem die rechten n bits des Wortes x entfernt werden. Das Ergebnis wird mit n 0ern von links aufgefüllt.
- << Left-shift operation
  x<<n wird gebildet indem die linken n bits des Wortes x entfernt werden. Das Ergebnis wird mit n 0ern von rechts aufgefüllt.
- ROTR^n(x): Die Rechtsrotation (circular right shift)
  Für ein w bit Wort x und n $\in$ w mit $0 \leq n < w$ ist
  ROTR^n(x) = (x>>n) $\vee$ (x<<w-n)
- SHR^n(x) Die Rechtsverschiebung
  Für ein w bit Wort x und n $\in$ w mit $0 \leq n < w$ ist
  SHR^n(x) = x>>n
- Ch(x,y,z) = $(x \wedge y) \oplus (\neg x \wedge z)$
- Maj(x,y,z) = $(x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$

- $\sum_0^{\{256\}}(x) = ROTR^2(x) \oplus ROTR^{13}(x) \oplus ROTR^{22}(x)$
- $\sum_1^{\{256\}}(x) = ROTR^6(x) \oplus ROTR^{11}(x) \oplus ROTR^{25}(x)$
- $\sigma_0^{\{256\}}(x) = ROTR^7(x) \oplus ROTR^{18}(x) \oplus SHR^3(x)$
- $\sigma_1^{\{256\}}(x) = ROTR^{17}(x) \oplus ROTR^{19}(x) \oplus SHR^{10}(x)$
- $K_0^{\{256\}}, K_1^{\{256\}}, ..., K_{63}^{\{256\}}$ sind die 64 Fixwerte des SHA-256
  in Hexadezimalzahlen sind diese:
  428a2f98, 71374491, b5c0fbcf, e9b5dba5, 3956c25b, 59f111f1, 923f82a4, ab1c5ed5, d807aa98, 12835b01, 243185be, 550c7dc3, 72be5d74, 80deb1fe, 9bdc06a7, c19bf174, e49b69c1, efbe4786, 0fc19dc6, 240ca1cc, 2de92c6f, 4a7484aa, 5cb0a9dc, 76f988da, 983e5152, a831c66d, b00327c8, bf597fc7, c6e00bf3, d5a79147, 06ca6351, 14292967, 27b70a85, 2e1b2138, 4d2c6dfc, 53380d13, 650a7354, 766a0abb, 81c2c92e, 92722c85, a2bfe8a1, a81a664b, c24b8b70, c76c51a3, d192e819, d6990624, f40e3585, 106aa070, 19a4c116, 1e376c08, 2748774c, 34b0bcb5, 391c0cb3, 4ed8aa4a, 5b9cca4f, 682e6ff3, 748f82ee, 78a5636f, 84c87814, 8cc70208, 90befffa, a4506ceb, bef9a3f7, c67178f2,

2.3.2 Vorverarbeitung der Nachricht

Bevor das Errechnen des Hashwertes beginnen kann muss einiges an Vorverarbeitung gemacht werden.
Diese besteht aus drei Teilen:

1. Auffüllen der Nachricht
   Dies stellt sicher das die Nachricht ein Vielfachens von 512 bits ist indem die Nachricht mit 0ern Aufgefüllt wird bis das nächste Vielfache von 512 erreicht wird.

2. Aufteilen der Nachricht

   Nachdem die Nachricht aufgefüllt wurde wird sie in N 512 bit Blöcke $M^{(1)}, M^{(2)}, ..., M^{(N)}$ aufgeteilt. Da ein Wort 32 bit hat entspricht jeder Block 16 Wörtern die mit $M_0^{(i)}, ..., M_{15}^{(i)}$ für i $\in$ {1,...N}

3. Setzen der Starthashwerte

   Es müssen Startwerte $H^{(0)}$ für den Hashwert gesetzt werden. Für SHA-256 sind das 8 Stück, die in Hexadezimalzahlen wie folgt lauten:

   $H_0^{(0)}$ = 6a09e667 $H_1^{(0)}$ = bb67ae85 $H_2^{(0)}$ = 3c6ef372 $H_3^{(0)}$ = a54ff53a
   $H_4^{(0)}$ = 510e527f $H_5^{(0)}$ = 9b05688c $H_6^{(0)}$ = 1f83d9ab $H_7^{(0)}$ = 5be0cd19

2.3.3 Berechnung des Hashwertes

SHA-256 kann verwendet werden, um für eine Nachricht M mit ℓ bits mit $0 \leq \ell < 2^{64}$, den Hashwert zu errechnen.

Der Algorithmus benötigt zusätzlich noch:

- Einen Nachrichtenablauf von 64 Wörtern der länge 32 bit die mit $W_0, W_1, ..., W_{63}$ bezeichnet werden
- 8 Arbeitsvariablen die a,b,c,d,e,f,g,h heissen
- Die Wörter der Hashwerte $H_0^{(i)}, H_1^{(i)}, ...H_7^{(i)}$ die Anfangs die Startwerte $H^{(0)}$ für den Hashwert enthalten. Diese werden von jedem nachfolgenden Hashwert ersetzt und enden bei dem finalen Hashwert $H^{(N)}$
- 2 Hilfswörter die wir mit T_1 und T_2 bezeichnen.

Nun kann die Berechnung des Hashwertes endlich beginnen:

Ein + bedeutet eine Addition Modulo 2^{32}.
Jeder Block $M^{(1)}, M^{(2)}, ..., M^{(N)}$ der Nachricht M wird der Reihe nach in folgenden Schritten verarbeitet:

Für i = 1 bis N

1. Vorbereiten des Nachrichtenablaufs $\{W_t\}$

$$W_t = \begin{cases} M_t^{(i)} & 0 \leq t \leq 15 \\ \sigma_1^{\{256\}}(W_{t-2}) + W_{t-7} + \sigma_0^{\{256\}}(W_{t-15}) + W_{t-16} & 16 \leq t \leq 63 \end{cases}$$

2. Befüllen der 8 Arbeitsvariablen mit den (i-1)sten Hashwerten

$a = H_0^{(i-1)}$
$b = H_1^{(i-1)}$
$c = H_2^{(i-1)}$
$d = H_3^{(i-1)}$
$e = H_4^{(i-1)}$
$f = H_5^{(i-1)}$
$g = H_6^{(i-1)}$
$h = H_7^{(i-1)}$

3. für t = 0 bis 63

$T_1 = h + \sum_1^{\{256\}}(e) + \mathrm{Ch}(e,f,g) + K_t^{\{256\}} + W_t$

$T_2 = \sum_0^{\{256\}}(a) + \mathrm{Maj}(a,b,c)$

h = g

g = f

f = e

e = d + T_1

d = c

c = b

b = a

a = $T_1 + T_2$

4. Berechnen des i-ten Zwischenwerts $H^{(i)}$ des Hashwertes

$$H_0^{(i)} = a + H_0^{(i-1)}$$
$$H_1^{(i)} = b + H_1^{(i-1)}$$
$$H_2^{(i)} = c + H_2^{(i-1)}$$
$$H_3^{(i)} = d + H_3^{(i-1)}$$
$$H_4^{(i)} = e + H_4^{(i-1)}$$
$$H_5^{(i)} = f + H_5^{(i-1)}$$
$$H_6^{(i)} = g + H_6^{(i-1)}$$
$$H_7^{(i)} = h + H_7^{(i-1)}$$

Nachdem die Schritte 1 bis 4 N mal wiederholt wurden erhält man eine 256-bit Hashwert der Nachricht M der wie folgt lautet:

$$H_0^{(N)}||H_1^{(N)}||H_2^{(N)}||H_3^{(N)}||H_4^{(N)}||H_5^{(N)}||H_6^{(N)}||H_7^{(N)}$$

Kapitel 3

Digitale Signaturen

3.1 Digitale Signaturen allgemein

Nach §2 Z 1 SigG ist eine elektronische Signatur „elektronische Daten, die anderen elektronischen Daten beigefügt oder mit diesen logisch verknüpft werden und die der Authentifizierung dienen."

Wie schon in Kapitel 1.2.3 Public-Key-System Formale Definition) erwähnt, kann das Verfahren der asymmetrischen Verschlüsselung zur Erstellung einer digitalen Signatur verwendet werden, indem das Verfahren umgekehrt angewendet wird. Das heißt, die Nachricht wird mit dem privaten Schlüssel verschlüsselt und mit dem öffentlichem Schlüssel entschlüsselt.
$E(D(M,K_D),K_E) = M$

Mit Hilfe einer Hashfunktion hat man nun die Möglichkeit eine Nachricht digital zu signieren. Das Verfahren der elektronischen Signierung einer Nachricht erfolgt also in drei Schritten:

1. Mit Hilfe einer Hashfunktion wird der Hashwert der Nachricht be-

stimmt.

2. Der Hashwert wird mit dem privaten Schlüssel des Absenders signiert, indem dieser den Hashwert mit seinem privaten Schlüssel verschlüsselt.

3. Der signierte Hashwert kann mit dem öffentlichen Schlüssel des Empfängers (oder durch eine beliebige andere Verschlüsselung) verschlüsselt werden.

Der Empfänger kann sich nun sicher sein, dass die Nachricht vom angegebenen Absender kommt und, dass diese nicht verändert wurde indem er...

1. ... mit Hilfe einer Hashfunktion den Hashwert der Nachricht bestimmt.

2. ... den mit der Nachricht mitgeschickten Wert mit seinem privaten Schlüssel (oder durch eine andere passende Entschlüsselung) entschlüsselt.

3. ... zur Verifikation der Identität des Absenders auf den entschlüsselten Wert den öffentliche Schlüssel des Absenders anwendet.
   Wenn der so entstandene Wert mit dem Hashwert aus Punkt 1 übereinstimmt kann sich der Empfänger der Identität des Absenders und der Unversehrtheit der Nachricht sicher sein.

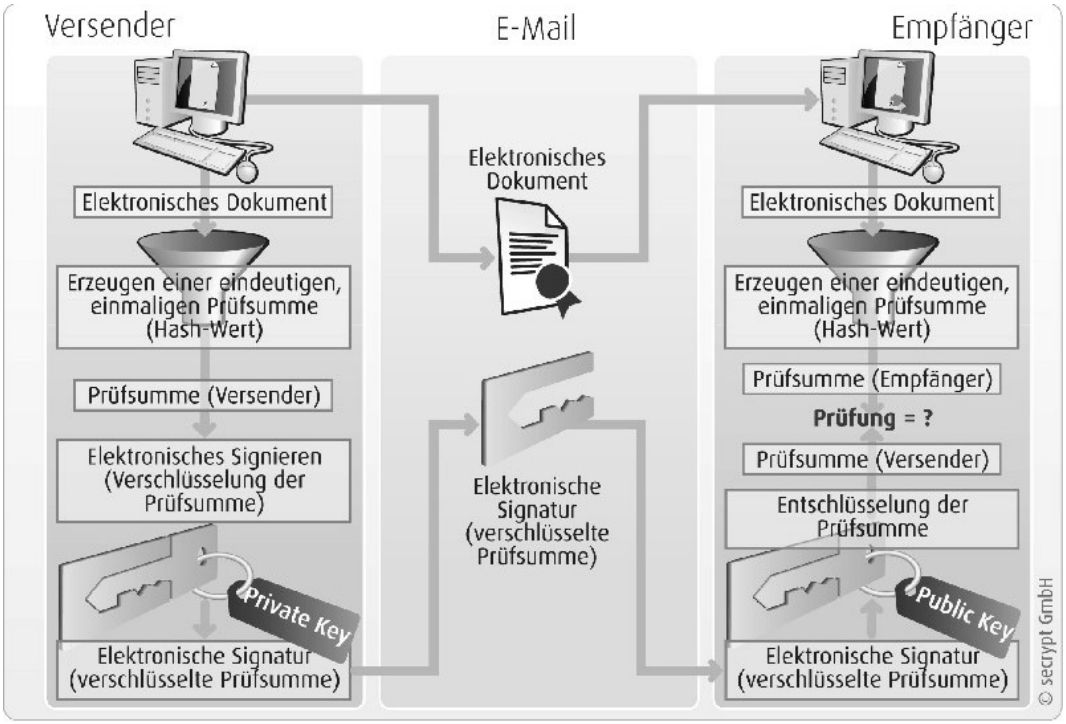

Abbildung 3.1: Datenfluss beim Erzeugen digitaler Signaturen

Deutschland war 1997 eins der ersten Länder mit einem Gesetz für digitale Signaturen und einer Signaturverordnung.
Seit 2008 gilt in Österreich ein Signaturgesetz in dem eine elektronische Signatur sogar als gleichwertig mit der handgeschriebenen Unterschrift angesehen wird. Österreich war europaweit das erste Land in dem diese Gleichwertigkeit gilt.

3.1.1 Anwendungsbereiche digitaler Signaturen

Digitale Signaturen können in allen Bereichen des Lebens verwendet werden. Sie finden ihre Hauptanwendung aber in alltäglichen Dingen wie E-Mail Verkehr und in kommerziellen Transaktionen über öffentliche Netzwer-

ke. In sensibleren Bereichen wie Verträge, private Einkäufe, Testament usw. wird heutzutage immer noch meistens die handschriftliche Unterschrift benutzt. Es besteht jedoch das Potenzial, dass sich dies in den nächsten Jahren ändern wird. Aufgrund des steigenden Technologiesierungsstands unserer Gesellschaft und der fallenden Preise für Hardware für digitale Signaturen ist es zu erwarten, dass sich digitale Signaturen weiter verbreitet werden.

3.2 Digitale Signaturen im RSA-Verfahren

Digitale Signaturen im RSA-Verfahren funktionieren genauso wie in Kapitel 2.2.1 Digitale Signaturen allgemein beschrieben.
Die Verwendung von privatem Schlüssel und öffentlichem Schlüssel wird vertauscht.
Das heißt:
Verschlüsselung: $C := M^d \mod N$
Entschlüsselung: $M := C^e \mod N$

Der Beweis der Gültigkeit der digitalen Signatur des RSA-Verfahrens ist analog zu dem Beweis der Gültigkeit des RSA-Verfahrens aus Kapitel 1.2.5.3 Beweis der Gültigkeit des RSA-Verfahrens. Ich werde diesen deshalb nicht anführen.

Kapitel 4

Public Key Infrastruktur (PKI)

4.1 Idee einer PKI

E-Commerce und Internet präsentieren eine Vielzahl an Möglichkeiten. Doch mit diesen entsteht auch eine Vielzahl an Sicherheits- und Vertrauensproblemen.
Das Hauptproblem hierbei ist die eindeutige Feststellung der Identität. Die meisten Geschäfte im Internet funktionieren ohnehin lediglich mit Vorkasse. Dennoch ist es für Händler wichtig die Identität des Kunden zu kennen, sodass beispielsweise nicht eine gestohlene Kreditkartennummer zum Einkauf verwendet werden kann. Umgekehrt ist es für Kunden wichtig sich der Identität des Service sicher sein zu können. Außerdem führt eine eindeutige Authentifizierung auch zu einer Unleugbarkeit, sodass beispielweise ein Kunde nicht behaupten kann, dass die Transaktion nicht stattgefunden hat.

Die Einführung einer PKI ist gedacht, um genau diese Vertrauensbeziehungen zu ermöglichen und diese auch aufrecht erhalten zu können. Hierfür werden folgende Bedingungen gestellt:

- Vertraulichkeit:

  Vertraulichkeit stellt den Datenschutz und die Geheimhaltung von Daten durch kryptographische Verfahren sicher. Ein gutes Beispiel für Daten die Vertraulichkeit benötigen sind persönliche Information eines Kunden. Die Verschlüsselung von Daten kann mit symmetrischer oder asymmetrischer Kryptographie geschehen. Da aber asymmetrische Kryptographie wesentlich langsamer ist wird diese meist nur für kleine Datenmengen, wie z.B. Schlüssel symmetrischer Systeme verwendet. Deshalb sind meistens symmetrische Verschlüsselungen das Mittel um in einer PKI für Vertraulichkeit zu sorgen.

- Integrität:

  Integrität sorgt dafür, dass Daten nicht beschädigt oder verändert werden. Ein gutes Beispiel für Daten die Integrität benötigen sind Zertifikate und digitale Signaturen. Der Inhalt von E-Mails, Verträgen, Einkaufsbestätigungen, in Allgemeinen alle Informationen auf die sich jemand verlassen muss benötigen eine Versicherung der Integrität. Um Integrität sicherzustellen wird in PKIs meistens die Public Key Kryptographie in Verbindung mit einem Hashalgorithmus verwendet.

- Authentifikation:

  Authentifizierung stellt die Identität eines Benutzers sicher. In PKIs geschieht dies mit Hilfe der Zertifikate und den Vertrauenssystem der CAs. Authentifizierung basiert auf dem öffentlichen und privaten Schlüssel. Ein Sender kann sich sicher sein, dass nur der geplante Empfänger Zugang zu dessen privaten Schlüssel hat.

- Unleugbarkeit:

  Unleugbarkeit sorgt dafür, dass Daten nicht verleugnet oder eine Trans-

aktion abgestritten werden kann. Dies ist ein wichtiger Service sobald Geld oder andere Wertgegenstände ausgetauscht werden, ebenso, wenn es sich um rechtliche oder vertragliche Verpflichtungen handelt. Unleugbarkeit wird ebenfalls mit Hilfe der digitalen Signatur gewährleistet. Wenn ein Benutzer Daten mit seinem privaten Schlüssel signiert, wird dadurch bestätigt, dass diese Daten von ihm stammen müssen, da sonst niemand Zugang zu diesem Schlüssel haben sollte.

Hauptfunktion von PKIs ist die sichere und integere Verteilung und Verwendung von privaten Schlüsseln und digitalen Zertifikaten sicherzustellen. Eine PKI ist jedoch selbst keine CA. Sie bietet lediglich eine Infrastruktur, die den verschiedenen technischen und geschäftlichen Ansprüchen genügt.

4.2 Teile einer PKI

Da eine PKI lediglich ein Gerüst aus Prozessen, Grundsätzen usw. ist, besteht sie aus vielen Einzelteilen.
Diese sind:

- Endbenutzer:
  Sind Personen, Organisationen oder andere User. Diese müssen die Fähigkeit besitzen ein Schlüsselpaar zu erzeugen und dieses auch sicher aufzubewahren.

- CA (Certificate Authority dt. Zertifizierungsdiensteanbieter):
  CA sind für die Ausstellung und den Wiederruf von Zertifikaten verantwortlich. (Ein Zertifikat ist eine elektronische Bescheinigung, die die Identitätsdaten einer bestimmten Person (Signator) mit einem Öffentlichen

Schlüssel (Public Key) verbindet. Ich werde in Kapitel 5 Digitale Zertifikate näher auf sie eingehen.) Außerdem fungieren sie als vertrauenswürdiger Dritter und spielen deshalb eine sehr wichtige Rolle in der PKI. Ich werde im Kapitel 5.4 Zertifizierungsdiestanbieter näher auf sie eingehen.

- RA (Registrations Authority dt. Registrierungsinstanz): Aufgabe einer RA ist es einer CA einiges an Arbeit abzunehmen. Hauptsächlich sind sie dafür verantwortlich, die Identität eines Endbenutzers und dessen Anspruch auf ein Zertifikat zu überprüfen. (Diese können sich je nach PKI stark unterscheiden)
  CA und RA sind oft zu einem sogenannten Trust Center zusammengefasst.

- Certificate Policy:
  Sind die Regeln nach denen das Trust Center seine Zertifikate ausstellt und verwaltet. Es werden noch weitere Angaben zur Sicherheit, zum Vertrauensmodell usw. gemacht.

- CPS (Certificate Practices Statement dt. Angaben zur Zertifikats Praxis):
  Diese werden veröffentlicht und beinhalten z.B.:

  - Alle Prozesse die ein öffentlicher Schlüssel durchläuft. (Erstellung, Sicherung, Aufbewahrung, Widerruf,...)
  - Die betriebliche und verfahrenstechnische Arbeitsweise der PKI.
  - Die Authentifizierungsschritte, die ein Endbenutzer durchlaufen muss um ein Zertifikat zu erhalten.

- Hardware Sicherheitsmodule (HSM):
  Ein HSM ist spezielles Computer Equipment das CA nutzen, um ihren privaten Schlüssel aufzubewahren und zu verwenden. Da der private Schlüssel einer CA benutzt wird um ein erstelltes Zertifikat zu unterschreiben, ist die sichere Aufbewahrung und Benutzung dieses eine der wichtigsten Aufgaben einer CA.

- Zertifikate (Public Key Certificates):
  Ist die eindeutige und bindende Bestätigung der Identität eines Endbenutzers. Ich werde im Kapitel 5.1 Zertifikate allgemein näher auf sie eingehen.

- Zertifikatserweiterungen: Sind zusätzliche Informationen die ein Zertifikat enthalten kann.
  Ich werde im Kapitel 5.2 x 509 v3 näher auf sie eingehen.

- Zertifikatsdepot:
  Hier werden alle aktiven Zertifikate verwaltet und anderen zu Verfügung gestellt. Verzeichnisse der widerrufenen Zertifikate (CRL) werden dort ebenfalls verwaltet und verteilt.

4.3 PKI Systeme

Das Erstellen, Verteilen und Verwalten eines öffentlichen Schlüssels und den dazugehörigen Zertifikat erfolgt normalerweise über eine CA.
Doch was passiert wenn beispielsweise zwei einander unbekannte Personen, die beide bei verschiedenen CA Zertifikate erworben haben, einen sicheren Austausch von Daten betreiben wollen? Gilt die CA der einen Person als vertrauenswürdiger Dritter für die andere - und umgekehrt?

Als Lösung für dieses Problem gibt es PKI Systeme.
Die Aufgabe eines PKI Systems ist es sicheren Austausch von, Daten, Identitäten und Geld in verschiedenen unsicheren Umgebungen (hauptsächlich Internet) zu ermöglichen.

4.3.1 Arten von PKI Systemen

Es gibt verschiede Ansätze für PKI Systeme. Ich werde nun auf sie eingehen.

4.3.1.1 Streng hierarchische Strukturen

Bei der hierarchischen Struktur werden die CA in verschiedene Stufen eingeteilt. Auf der höchsten Stufe befindet sich eine einzige globale root CA die für alle Zertifikate direkt oder indirekt verantwortlich ist. Noch unter der untersten Stufe sind die Endbenutzer. CA dürfen immer nur Zertifikate auf eine niedrigere Stufen ausstellen. Der vertrauenswürdige Dritte für jeden Endbenutzer ist die CA, die das Zertifikat ausgestellt hat. Dadurch ergibt sich eine leicht zu erweiternde, skalierbare und effiziente Struktur.
Der hierarchische Ansatz hat jedoch auch Nachteile. Jede weitere CA verursacht zusätzlichen administrativen Aufwand. Außerdem kann durch den Ausfall einer einzigen CA das gesamte Vertrauenssystem beschädigt und vielleicht sogar unbrauchbar gemacht werden.

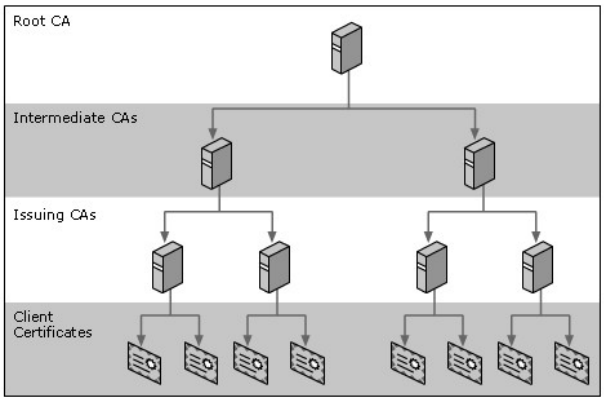

Abbildung 4.1: Beispiel für eine streng hierarchische Struktur

4.3.1.2 Cross-Zertifizierung

Bei der Cross-Zertifizierung werden anstelle einer einzigen globalen root CA viele individuelle CA eingesetzt. Dies erlaubt den Endbenutzern die CA nach ihren persönlichen Bedürfnissen zu wählen. Was aber nun, wenn ein Endbenutzer die Identität eines anderen feststellen will, welcher sein Zertifikat bei einer anderen CA erworben hat? Dafür wird Cross-Zertifizierung verwendet. Eine CA führt einige Test durch und überprüft so, ob die andere CA ihren Sicherheits- und sonstigen Auflagen entspricht. Wenn das der Fall ist, stellt sie dieser ein Zertifikat aus. Dadurch wird das Vertrauenssystem erweitert. Alle Zertifikate, die von der anderen CA ausgestellt wurden gelten nun auch als sicher.

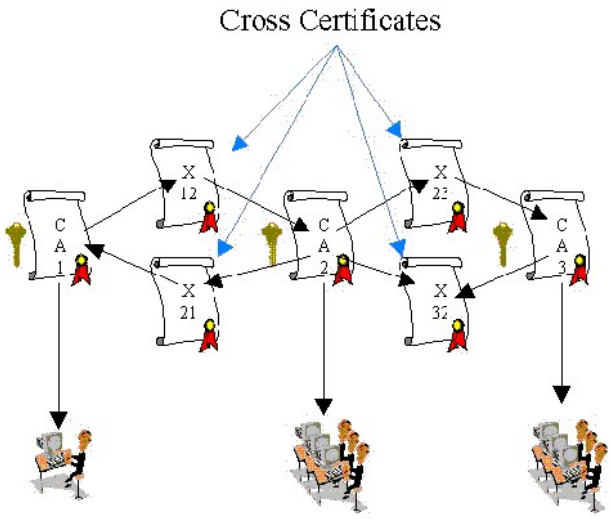

Abbildung 4.2: Beispiel für Cross-Zertifizierung

4.3.1.3 Web of trust

Web of Trust ist ein komplett unterschiedlicher Ansatz. Hierbei wird auf eine CA und RA verzichtet. Es existiert dadurch kein vertrauenswürdiger Dritter, der die Identität des Endbenutzers garantiert. Stattdessen übernehmen die Endbenutzer all diese Aufgaben. Ich bestätige die Identität meiner Freunde und guten Bekannten. Die tun wiederum dasselbe und so weiter und so weiter....

Nach dem Smallworld Phänomen sollten auf diese Weise Verbindungen zu jedem anderen Benutzer entstehen. Je nachdem über wie viele Ecken ich den Zielbenutzer kenne oder wie sehr ich den Benutzern, über die ich den Zielbenutzer kenne vertraue, wird ein sogenanntes Level of Trust bestimmt. Dies gibt an wie sicher ich mir der Identität der Zielperson sein kann.

Eine Anwendung des Web of Trust ist PGP.

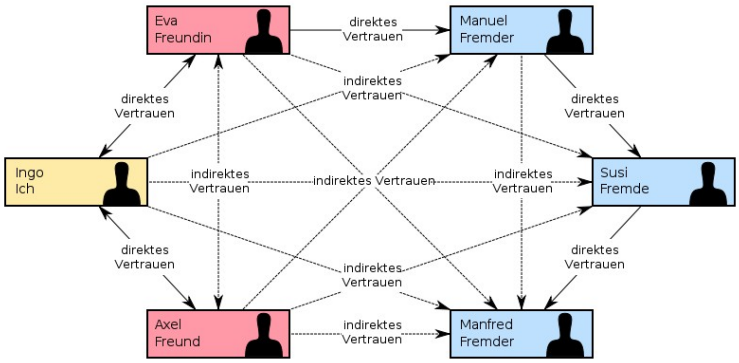

Abbildung 4.3: Beispiel für Web of trust

4.4 Beispiel einer Anwendung einer PKI

Ich werde mich in diesem Beispiel an die Public Key Infrastructure Overview von Sun microsystems halten.

Das Beispiel teilt sich in zwei Teile - die Vorbereitung und die Benutzung.

Vorbereitung:

1. Alice und Bob generieren jeweils ein Schlüsselpaar.
2. Alice und Bob fordern ein Zertifikat an, indem sie ihre öffentlichen Schlüssel, Namen und sonstigen Informationen einer RA schicken.
3. Die RA überprüft die Angaben und schickt das Ansuchen um ein Zertifikat der CA.
4. Die CA erstellt Zertifikate für Bob und Alice indem sie die öffentlichen Schlüssel und die persönlichen Daten nach dem Zertifikatsanforderun-

gen formatieren und diese dann mit dem privaten Schlüssel der CA unterschreiben.

5. Jetzt haben Bob und Alice gültige digitale Zertifikate.

6. Alice und Bob generieren noch einen geheimen symmetrischen Schlüssel Damit ist die Vorbereitung erledigt. Von nun an können Bob und Alice Nachrichten austauschen.

Senden und Empfangen einer Nachricht:

1. Alice möchte Bob eine Nachricht schicken. Sie wendet auf die Nachricht eine Hashfunktion an um einen eindeutigen Hashwert zu berechnen.

2. Alice verbindet die Nachricht und den Hashwert und signiert diese mit ihrem privaten Schlüssel. Durch das Signieren der Nachricht erfolgt keine Verschlüsselung. Aber es ist nun eindeutig, dass die Nachricht von Alice kommen muss. Meistens wird jedoch nur der Hashwert signiert, da dies zur Feststellung der Identität von Alice völlig ausreichend ist und dadurch sehr viel Rechenleistung gespart wird. (Siehe Kapitel 2 Hashfunktionen)

3. Da Alice und Bob nicht nur die Eindeutigkeit des Absenders wichtig ist, sondern sie auch darauf Wert legen, dass der Inhalt der Nachricht geheim bleibt, wird die signierte Nachricht zusätzlich mit Alices symmetrischem Schlüssel verschlüsselt. Es gibt auch andere Verfahren, die zum Verschlüsseln der Nachricht verwendet werden können, doch die symmetrische Verschlüsselung ist eine schnelle saubere Methode.

4. Alice muss Bob nun den symmetrischen Schlüssel zukommen lassen. Also verschlüsselt sie den symmetrischen Schlüssel mit Bobs öffentlichem

Schlüssel, welchen sie in Bobs Zertifikat in der Zetifikatsliste seiner CA findet. Sie kann sich sicher sein, dass Bobs öffentlicher Schlüssel korrekt ist da dieser von der CA signiert wurde.

5. Alice schickt nun Bob die signierte verschlüsselte Nachricht mit Hashwert und den mit dem öffentlichen Schlüssel von Bob verschlüsselten symmetrischen Schlüssel.

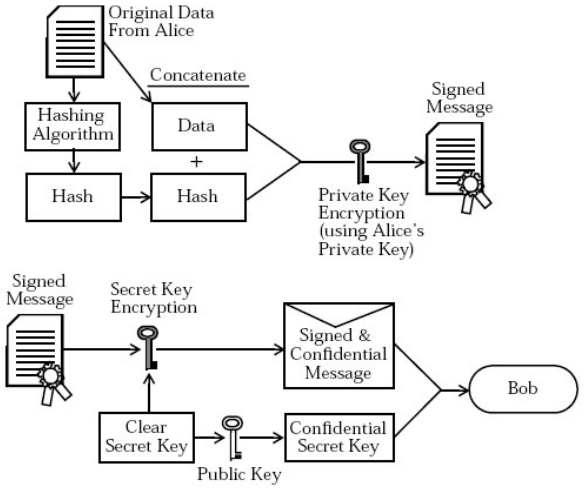

Abbildung 4.4: Alices Schritte zum Verschlüssen und Hashen den Nachricht

6. Bob empfängt die Daten von Alice. Zuerst entschlüsselt er mit seinem privaten Schlüssel den symmetrischen Schlüssel von Alice.

7. Nun kann er mit Hilfe des symmetrischen Schlüssels die (signierte) Nachricht und den signierte Hashwert gewinnen.

8. Bob kann nun mit dem öffentlichen Schlüssel von Alice, welchen er ebenfalls aus der Zertifikatsliste der CA erhält, die Signierung entfernen und erhält dadurch die Originalnachricht und deren Hashwert.

9. Um zu überprüfen ob die Nachricht verändert wurde, berechnet Bob

selbst den Hashwert indem er dieselbe Hashfunktion wie Alice verwendet.

10. Zu guter Letzt vergleicht Bob die beiden Hashwerte. Wenn diese übereinstimmen kann er sicher sein, dass die Nachricht nicht verändert wurde.

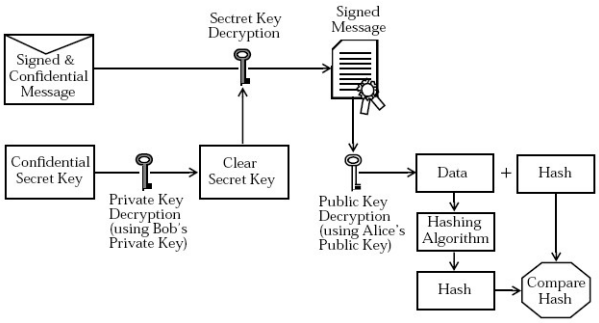

Abbildung 4.5: Bobs Schritte zum Entschlüsseln und überprüfen der Nachricht

Kapitel 5

Digitale Zertifikate

5.1 Zertifikate allgemein

Ein Zertifikat ist eine elektronische Bescheinigung, die die Identitätsdaten einer bestimmten Person (Signator) mit einem öffentlichen Schlüssel (Public Key) verbindet. Neben zusätzlichen inhaltlichen Informationen unterscheiden sich Zertifikate insbesondere durch unterschiedliche rechtliche Anforderungen, die Garantie des Sicherheitsniveaus des Ausstellungsprozesses und der Vertrauenswürdigkeit des Ausstellers.
Um ihre Funktion erfüllen zu können, sind in Zertifikaten folgende Grunddaten zweckmäßig:

- Name des Signators
- eine elektronische Signatur des Zertifizierungsdiensteanbieters
- der öffentliche Schlüssel des Signators

Der öffentliche Schlüssel wird der Signatur beigefügt. Es besteht allerdings auch die Möglichkeit, Zertifikate aus dem Verzeichnisdienst des Zertifizierungsdiensteanbieters einzusehen. [1]

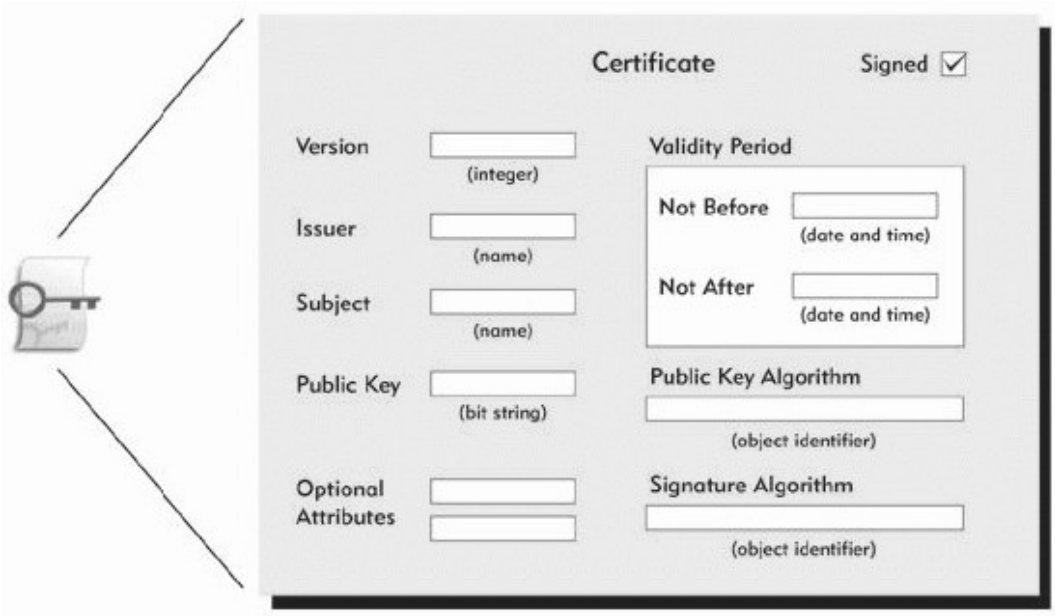

Abbildung 5.1: Beispiel eines digitalen Zertifikats

Es gibt viele Zertifikatstandards. Der wichtigste ist das IETF-Profil des X.509-v3-Zertifikatstandards, auf den ich im nächsten Teil näher eingehen werde.

[1] Bundeskanzleramt, E-Government http://www.bka.gv.at/site/5567/default.aspx#a6

5.2 Zertifikatsstandart X.509

Das X.509 Zertifikate ist ein flexibles und mächtiges Zertifikat. Es kann eingesetzt werden um eine Vielzahl von Informationen zu übertragen. Viele dieser Information sind optional und selbst der Inhalt der Pflichtfelder ist nicht immer gleich.
Das X.509 Zertifikat wird mit der digitalen Signatur der austellenden CA geschützt, wodurch die Endbenutzer sicher sein können, dass die Informationen seit der Signierung nicht verändert wurden.
Das Zertifikat besteht aus einigen allgemeinen Feldern und aus einigen Erweiterungsfeldern:

- Version:
  Im Versionsfeld befindet sich die Angabe darüber welche Version des Zertifikats verwendet wird.
  Bei Version 1 gibt es weder Erweiterungen noch die eindeutigen IDs des Benutzers und CA.
  Bei Version 2 gibt es die IDs aber noch keine Erweiterungen.
  Bei Version 3 gibt es die IDs und die Erweiterungen.

- Seriennummer:
  Ist eine Nummer die vom Zertifikatsaussteller kreiert wird. Diese muss, bezogen auf den Zertifikatsaussteller, eindeutig sein. Also kann jedes Zertifikat mit dem Namen des Austellers und der Seriennummer eindeutig identifiziert werden.

- Signatur:
  In diesem Feld wird angegeben welcher Algorithmus verwendet wurde um das Zertifikat zu signieren.

- Aussteller:
  Im diesem Feld wird ein eindeutiger Name des Zertifikatsausteller angegeben. Der Name eines Zertifikatsaustellers ist eindeutig da es nicht erlaubt ist, dass zwei Zertifikatsausteller den selben Namen haben.

- Gültigkeit:
  Hier wird der Zeitraum angegeben indem das Zertifikat gültig ist.
  Die Dauer wie lange ein Zertifikat gültig sein darf ist rechtlich geregelt. (Meist zwei bis maximal fünf Jahre)

- Benutzer:
  Hier wird ein eindeutiger Name für den Benutzer angegeben, der den privaten Schlüssel zu dem in diesem Zertifikat angegebenen öffentlichen Schlüssel besitzt.
  Wichtig hierbei ist das der Benutzer eine CA, RA oder ein Endbenutzer sein kann. Endbenutzer können Menschen oder Firmen sein. Aber es ist auch möglich, dass der Endbenutzer eine Hardware ist. Alles kann ein Endbenutzer sein das fähig ist, einen privaten Schlüssel zu benutzen.

- Schlüsselinformation des Benutzers:
  Enthält den öffentlichen Schlüssel des Benutzers und eine Angabe darüber um welchen Algorithmus es sich handelt. Mithilfe dieses Schlüssels werden digitale Signaturen überprüft. Handelt es sich bei dem Benutzer um eine CA so wird dieser Schüssel benutzt um die digitalen Signaturen auf Zertifikaten zu bestätigen.

- Eindeutige ID des Benutzers und eindeutige ID des Ausstellers:
  Diese eindeutigen IDS sind gedacht um die Erneuerung von Zertifikaten zu erleichtern und die damit verbundene Wiederverwendung von

Namen der Benutzer und des Austeller zu ermöglichen. Leider hat sich diese Methode nicht bewährt.

- Erweiterungen:
  Optionale Felder, für die es viele unterschiedliche Möglichkeiten an Informationen gibt, die angegeben werden können.
  Ich werde auf einige eingehen:

  - Typ des Benutzers:
    Gibt an um welche Art von Benutzer es sich handelt.(z.B. CA, RA, Endbenutzer...)
  - Benutzer Information:
    Hier werden zusätzliche Informationen über den Benutzer angegeben. Diese sollen dabei helfen seine Identität festzustellen. (z.B. Email Adresse, Land, Organisation,...)
  - Schlüsselattribute:
    Gibt zusätzliche relevante Attribute über den Schüssel an. (Verwendbar für digitale Signaturen, für den Schlüsseltransport geeignet,...)
  - Information über die Policy:
    Dieses Feld ist gedacht um anderen Benutzern einen Eindruck über die Sicherheit des Zertifikats zu geben, indem sie die Grundsätze unter denen das Zertifikat ausgestellt wurde kennen.
  - Zertifikatserweiterung:
    Ist ein Feld indem sich jeder Zertifikatsaussteller selbst aussuchen kann welche zusätzliche Information angegeben wird, wobei die Standarderweiterungen für bessere Kompatibilität der Zertifika-

te sorgen. Dadurch sind private Erweiterungen auch mit höheren Kosten verbunden.

- Einschränkungen:
  Dieses Feld ist für Zertifikate an CAs gedacht. Es dient dazu festzulegen ob dieses Zertifikat dazu berechtigt ist weitere Zertifikate zu erstellen.
- Anwendungsbereiche des Schlüssels:
  Gibt an wofür der Schlüssel geeignet ist. (z.B. Unleugbarkeit, Verschlüsselung von Daten,...)
- Alternative Namen des Benutzers:
  Hier werden alternative Namen des Benutzers angegeben. (z.B. DNS Namen, email Adressen,...)
- Identifikation des CAschlüssels:
  Falls eine CA mehre Schlüssel hat hilft dieses Feld den Benutzer den Richtige auszuwählen.
- Identifikation des Benutzerschlüssels:
  Ein Benutzer kann mehrere Schlüsselpaare oder mehrere Zertifikate für einen Schlüssel besitzen. Dieses Feld soll anderen Benutzern dabei helfen den richtigen Schlüssel zu wählen.

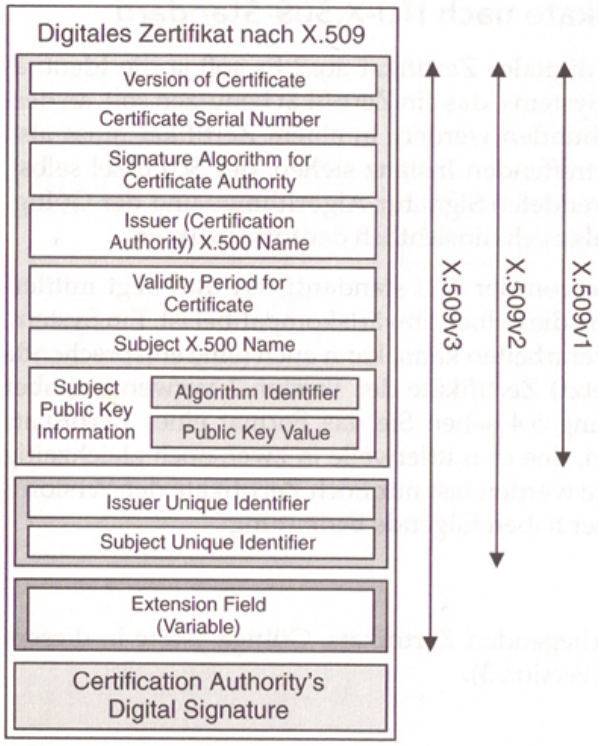

Abbildung 5.2: Teile eines X.509 Zertifikats

5.3 Zertifikate in Österreich

In Österreich unterscheidet das Signaturgesetz zwischen einfachen und qualifizierten Zertifikat:

5.3.1 Einfache Zertifikate

Nach §2 Z 1 SigG ist ein Zertifikat eine elektronische Bescheinigung, mit der Signaturprüfdaten (Public Key) einer bestimmten Person zugeordnet werden und deren Identität bestätigt wird.

Es werden keine zusätzlichen Ansprüche an einfache Zertifikate gestellt.

5.3.2 Qualifizierte Zertifikate

An qualifizierte Zertifikate werden wesentlich höhere Anforderungen gestellt, welche sich in §5 SigG finden. Laut diesem müssen qualifizierte Zertifikate zumindest folgende Angaben enthalten:

1. den Hinweis darauf, dass es sich um ein qualifiziertes Zertifikat handelt,
2. den unverwechselbaren Namen des ZDA und den Staat seiner Niederlassung,
3. den Namen des Signators oder ein Pseudonym, das als solches bezeichnet sein muss,
4. die dem Signator zugeordneten Signaturprüfdaten,
5. Beginn und Ende der Gültigkeit des Zertifikats (Damit wird bewirkt, dass das Zertifikat den aktuellen Technologien sowie Sicherheitsstandards entspricht),
6. die eindeutige Kennung des Zertifikats,
7. eine fortgeschrittenen elektronischen Signatur des ZDA.

Optional können qualifizierte Zertifikate noch folgende Angaben enthalten:

- auf Verlangen des Zertifikatswerbers Angaben über eine Vertretungsmacht oder eine andere rechtlich erhebliche Eigenschaft des Signators,
- eine Einschränkung des Anwendungsbereichs des Zertifikats,
- eine Begrenzung des Transaktionswerts, auf den das Zertifikat ausgestellt ist,

- auf Verlangen des Zertifikatswerbers können weitere rechtlich erhebliche Angaben in das qualifizierte Zertifikat aufgenommen werden.

Außerdem muss ein Zertifikat (nach Richtlinien des Bundeskanzleramts) von einem Zertifizierungsdiensteanbieter, der die Anforderungen des § 7 SigG erfüllt (siehe Kapitel 5.4.1 Rechtliche Anforderungen an Zertifizierungsdiensteanbieter in Österreich), ausgestellt sein.

5.4 Zertifizierungsdiensteanbieter

Wie in dieser Arbeit schon mehrfach erwähnt handelt es sich bei einer CA (oder ZDA) um einen vertrauenswürdigen Dritten der Management Services anbietet. Wenn ein Benutzer der CA genügend Beweise für seine Identität liefert, bestätigt diese seine Identität. Dies geschieht indem die CA ein Zertifikat mit der Identität des Benutzers und seinen öffentlichem Schlüssel ausstellt. Dieses wird dann noch signiert um sicherzustellen, dass es nicht verändert wurde und um die ausstellende CA zu authentifizieren.

5.4.1 Rechtliche Anforderungen an Zertifizierungsdiensteanbieter in Österreich

Wie schon in Kapitel 5.3.2 Qualifizierte Zertifikate erwähnt, muss ein Zertifizierungsdienstanbieter, der ein qualifiziertes Zertifikat ausstellen möchte, die Anforderungen des §7 und §8 SigG erfüllen.
In diesen wird auf die Zuverlässigkeit von ZDA eingegangen. Außerdem werden deren Sicherheits- und Leistungsvoraussetzungen (z.B. Verzeichnis- und Wiederrufsdienste, Personal, Speicherung von Zertifikaten, usw.) festgelegt. Weiter werden Vorgaben über die finanzielle Leistungsfähigkeit und techni-

sche Komponenten von ZDAs gemacht.

§7 und §8 des Signaturgesetzes Lauten:

Ein ZDA hat:

- die erforderliche Zuverlässigkeit für die von ihm bereitgestellten Signatur- oder Zertifizierungsdienste aufzuweisen,
- den Betrieb eines schnellen und sicheren Verzeichnisdienstes sowie eines unverzüglichen und sicheren Widerrufsdienstes sicherzustellen und im Sicherheitskonzept darzulegen, in welcher Form dies erfolgt,
- in qualifizierten Zertifikaten sowie für Verzeichnis- und Widerrufsdienste qualitätsgesicherte Zeitangaben zu verwenden und jedenfalls sicherzustellen, dass der Zeitpunkt der Ausstellung und des Widerrufs eines qualifizierten Zertifikats bestimmt werden kann,
- die Identität und gegebenenfalls besondere rechtlich erhebliche Eigenschaften der Person, für die ein qualifiziertes Zertifikat ausgestellt wird, zuverlässig zu überprüfen,
- zuverlässiges Personal mit den für die bereitgestellten Dienste erforderlichen Fachkenntnissen, Erfahrungen und Qualifikationen, insbesondere mit Managementfähigkeiten sowie mit Kenntnissen der Technologie elektronischer Signaturen und angemessener Sicherheitsverfahren, zu beschäftigen und geeignete Verwaltungs- und Managementverfahren, die anerkannten Normen entsprechen, einzuhalten,
- über ausreichende Finanzmittel zu verfügen, um den Anforderungen dieses Bundesgesetzes und der auf seiner Grundlage ergangenen Ver-

ordnungen zu entsprechen, sowie Vorsorge für die Befriedigung von Schadenersatzansprüchen, etwa durch Eingehen einer Haftpflichtversicherung, zu treffen,

- alle maßgeblichen Umstände über ein qualifiziertes Zertifikat während eines für den Verwendungszweck angemessenen Zeitraums - gegebenenfalls auch elektronisch - aufzuzeichnen, sodass insbesondere in gerichtlichen Verfahren die Zertifizierung nachgewiesen werden kann, sowie

- Vorkehrungen dafür zu treffen, dass die Signaturerstellungsdaten der Signatoren weder vom ZDA noch von Dritten gespeichert oder kopiert werden können,

- für die Signatur- und Zertifizierungsdienste sowie für die Erstellung und Speicherung von Zertifikaten vertrauenswürdige Systeme, Produkte und Verfahren, die vor Veränderungen geschützt sind und für die technische und kryptographische Sicherheit sorgen, zu verwenden. Er hat insbesondere geeignete Vorkehrungen dafür zu treffen, dass Signaturerstellungsdaten geheim gehalten werden, dass Daten für qualifizierte Zertifikate nicht unerkannt gefälscht oder verfälscht werden können und dass diese Zertifikate nur mit Zustimmung des Signators öffentlich abrufbar sind. Für die Erzeugung und Speicherung von Signaturerstellungsdaten sowie für die Erstellung und Speicherung von qualifizierten Zertifikaten sind technische Komponenten und Verfahren, die den Anforderungen des § 18 entsprechen, zu verwenden.

- Signaturerstellungsdaten vor unbefugtem Zugriff zu sichern,

- nach Maßgabe des Zertifizierungskonzepts auf Verlangen des Zertifikatswerbers Angaben über seine Vertretungsmacht oder eine andere

rechtlich erhebliche Eigenschaft in das qualifizierte Zertifikat aufzunehmen, sofern ihm oder einer anderen Stelle diese Umstände zuverlässig nachgewiesen werden,

- für die Prüfung von qualifiziert signierten Daten technische Komponenten und Verfahren zu benutzen, die sicherstellen, dass:

  1. die signierten Daten nicht verändert worden sind,
  2. die Signatur zuverlässig geprüft und das Ergebnis korrekt angezeigt wird,
  3. der Prüfer feststellen kann, auf welche Daten sich die elektronische Signatur bezieht,
  4. der Prüfer feststellen kann, welchem Signator die elektronische Signatur zugeordnet ist, wobei die Verwendung eines Pseudonyms angezeigt werden muss, und
  5. sicherheitsrelevante Veränderungen der signierten Daten erkannt werden können.

  Auf Ersuchen von Gerichten oder anderen Behörden hat ein ZDA die Prüfung der auf seinen qualifizierten Zertifikaten beruhenden qualifizierten Signaturen vorzunehmen.

- Gemäß §23 SigG haftet ein Zertifizierungsdienstanbieter, der ein qualifiziertes Zertifikat ausstellt, unter anderem für die Richtigkeit der Angaben im qualifizierten Zertifikat zum Zeitpunkt der Ausstellung.

In Österreich gibt es lediglich einen Zertifizierungsdienstanbieter, der berechtigt ist, qualifizierte Zertifikate auszustellen. Es handelt sich hierbei um A-Trust.

5.4.2 Akkreditierung von Zertifizierungsdienstanbieter in Österreich

Akkreditierung von CA ist ein Verfahren, in dem von einer meist staatlichen Aufsichtsstelle die Arbeitsweise, technischen Sicherheitsvorkehrungen usw. überprüft und überwacht werden. Die Akkreditierung für eine CA ist freiwillig. Eine CA darf auch ohne Akkreditierung arbeiten. Wenn ein Zertifizierungsdienstanbieter die Kriterien aus §17 erfüllt gilt er als akkreditiert. In §17 wird die Einhaltung der Bundesgesetze für eine ZDA vorausgesetzt. Außerdem werden die Sicherheitsanforderungen aus §18 vorausgesetzt.
Diese beschäftigen sich mit der Sicherheit der technischen Komponenten und der Verfahren die zum Erzeugen, Speichern von digitalen Signaturen und digitalen Zertifikaten verwendet werden. Hauptsächlich wird auf den Stand der Technik, die Normen der Europäischen Kommission und die Geheimhaltung eingegangen.
In §17 werden Dienste behandelt, die ermöglichen festzustellen ob es sich bei einer ZDA um eine akkreditierte ZDA handelt. Außerdem wird erwähnt, dass eine akkreditierte ZDA sich unter laufender Aufsicht befindet und diesen Titel jederzeit verlieren kann.

§17 lautet:

- ZDA, die der Aufsichtsstelle vor der Aufnahme ihrer Tätigkeit als akkreditierte ZDA, die Einhaltung der Anforderungen dieses Bundesgesetzes und der auf seiner Grundlage ergangenen Verordnungen nachweisen, sind auf Antrag von der Aufsichtsstelle zu akkreditieren. Akkreditierte ZDA dürfen sich mit Zustimmung der Aufsichtsstelle im Geschäftsverkehr als solche bezeichnen. Im Zusammenhang mit Signatur-

und Zertifizierungsdiensten sowie mit Signaturprodukten darf diese Bezeichnung nur verwendet werden, wenn die Sicherheitsanforderungen nach § 18 erfüllt werden.

§18 lautet:

- Für die Erzeugung und Speicherung von Signaturerstellungsdaten sowie für die Erstellung qualifizierter Signaturen sind solche technische Komponenten und Verfahren einzusetzen, die die Fälschung von Signaturen sowie die Verfälschung signierter Daten zuverlässig erkennbar machen und die die unbefugte Verwendung von Signaturerstellungsdaten verlässlich verhindern.
- Die bei der Erstellung einer qualifizierten Signatur verwendeten technischen Komponenten und Verfahren müssen zudem sicherstellen, dass die zu signierenden Daten nicht verändert werden; sie müssen es weiters ermöglichen, dass dem Signator die zu signierenden Daten vor Auslösung des Signaturvorgangs dargestellt werden und dass der Signator zu diesem Zeitpunkt über die Anzahl der Signaturen, die er im Signaturvorgang auslöst, Kenntnis erlangt. Die Signaturerstellungsdaten dürfen mit an Sicherheit grenzender Wahrscheinlichkeit nur einmal vorkommen, sie dürfen weiters mit hinreichender Sicherheit nicht ableitbar sein; ihre Geheimhaltung muss sichergestellt sein.
- Bei der Erstellung und Speicherung von qualifizierten Zertifikaten sind solche technische Komponenten und Verfahren einzusetzen, die die Fälschung und Verfälschung von Zertifikaten verhindern.
- Die technischen Komponenten und Verfahren für die Erstellung

qualifizierter elektronischer Signaturen müssen nach dem Stand der Technik hinreichend und laufend geprüft sein. Die Erfüllung der Sicherheitsanforderungen an sichere Signaturerstellungseinheiten nach diesem Bundesgesetz und den auf seiner Grundlage ergangenen Verordnungen muss von einer Bestätigungsstelle bescheinigt sein.

  - Entsprechen technische Komponenten und Verfahren den allgemein anerkannten Normen, die von der Europäischen Kommission der Signaturrichtlinie festgelegt werden, so gelten die entsprechenden Sicherheitsanforderungen nach diesem Bundesgesetz und den auf seiner Grundlage ergangenen Verordnungen als erfüllt.

- Die Aufsichtsstelle hat dafür Sorge zu tragen, dass die akkreditierten ZDA in ein elektronisch jederzeit allgemein zugängliches Verzeichnis aufgenommen werden.

- Die freiwillige Akkreditierung eines ZDA ist in das qualifizierte Zertifikat aufzunehmen oder sonst in geeigneter Weise zugänglich zu machen.

- Die Aufsichtsstelle hat für die laufende Aufsicht über die von ihr akkreditierten ZDA Sorge zu tragen. Sie hat die Akkreditierung eines ZDA zu widerrufen, wenn die Voraussetzungen einer Akkreditierung nicht mehr erfüllt sind.

Dadurch erwirbt die CA besondere Rechte. Sie darf sich als akkreditierter Zertifizierungsdienstanbieter bezeichnen. Diese Bezeichnung darf dann beispielsweise auf der Homepage oder am Briefkopf geführt werden. Dadurch wird für einen Endbenutzer ein zusätzliches Maß an Vertrauen gegenüber der CA sichergestellt. Außerdem, wie schon in §17 erwähnt, wird die Akkre-

ditierung einer CA in den qualifizierte Zertifikaten erkenntlich und es existiert ein allgemein zugängliches Verzeichnis aller akkreditierter ZDAs.

Kapitel 6

Schlüsselmanagment

Die Sicherheit eines Kryptographischen Systems beruht im Wesentlichen auf einer sicheren Verwaltung der geheimen Schlüssel. Wenn die Schlüssel leicht zu erraten sind, oder nicht autorisierten Personen zugänglich sind, dann nützt auch das stärkste Verschlüsselungsverfahren nichts. Deshalb werde ich mich im Laufe dieses Kapitels mit der Aufbewahrung und Erzeugung von Schlüsseln beschäftigen.

6.1 Schlüsselerzeugung

Für die Schlüsselerzeugung eines symmetrischen Kryptosystems benötigt man Zufallszahlen.

6.1.1 Erzeugen von Zufallszahlen

Eine Zufallszahl benötigt zwei Dinge:

- Zufallszahlengenerator (RNG)
- Startwert (seed)

6.1.1.1 Zufallszahlengeneratoren

Es gibt 2 Arten von Zufallszahlengeneratoren - deterministische und nicht deterministische.

Deterministische RNGs liefern bei gleichem Startwert immer dieselbe Zufallszahl. Diese werden auch als Pseudozufallszahlengeneratoren bezeichnet. Ein Pseudozufallszahlengenerator erzeugt also keine wirkliche Zufallszahl. Es wird lediglich aus einem Seed ein fixer Wert berechnet. Deshalb ist bei Pseudozufallszahlengeneratoren auf zwei Dinge zu achten:
Der Generator soll eine Gleichverteilung über den im Vorfeld definierten Zahlenraum haben. Weiters ist die Erzeugung des Seed extrem wichtig. Da die meisten RNGs öffentlich bekannt sind, kann aufgrund äußerer Umstände bekannt sein mit welchem RNG gearbeitet wird. Wenn dann auch noch der Seed nicht wirklich zufällig ist, fällt es einem Angreifer leicht die erzeugte Zufallszahl zu erraten.
Das beste Beispiel hierfür ist die Zufallszahl deren Seed mit Hilfe der Systemzeit erzeugt wurde. Alles was ein Angreifer benötigt um diese nachzubilden ist die Zeit wann die Zufallszahl erzeugt wurde.
Die Erzeugung des Seed sollte auf mehreren Faktoren beruhen (z.B. Systemzeit und Temperatur im Kern). Diese Faktoren sollten sich regelmäßig ändern. Pseudozufallszahlen können für die meisten Anwendungen als zufällig genug angesehen werden.

Nicht deterministische RNGs liefern für den gleichen Seed verschiedene Ergebnisse. Dafür ist es erforderlich externe Vorgänge (z.B. Temperatur im Computerkern) in die Berechnung der Zufallszahl einzubeziehen. Es werden auch Zufallsereignisse wie z.B. atmosphärisches Rauschen zur Berechnung

verwendet. Bei nicht deterministischen RNGs ist die Zufälligkeit des Seed von nicht ganz so großer Bedeutung, da ein Angreifer selbst bei Kenntnis des Seed und dem RNG die Zufallszahl immer noch nicht nachbilden kann. Trotzdem wird die Zufälligkeit des Seed nicht vernachlässigt.

Es gibt verschiedene vordefinierte Verfahren um eine Zufallszahl zu erzeugen:

- Bei Linux gibt es die Funktion /dev/random bei der u.a. aus dem Rauschen von Gerätetreibern eine Zufallszahl bestimmt wird.
- Bei ANSI X9.17 werden mit Hilfe von Triple-Des Pseudozufallszahlen erzeugt.
- Bei FIPS 186 werden mit Hilfe der Hashfunktion SHA-1 Zufallszahlen erzeugt.
- uvm.

6.1.2 Schlüsselerzeugung im RSA-Verfahren

Die Schlüsselerzeugung von asymmetrischen Verfahren erfordert wesentlich mehr Rechenaufwand als die Generierung von Zufallszahlen für symmetrische Verfahren.
Die Schlüsselerzeugung im RSA-Verfahren wurde schon in Kapitel 1.2.5 RSA-Verfahren behandelt. Diese ist in wenigen Schritten durchzuführen. Doch für die Schlüsselerzeugung im RSA-Verfahren (und auch bei anderen asymmetrischen Verfahren) werden große Primzahlen benötigt. Es gibt unendlich viele Primzahlen, aber diese zu finden ist ein sehr aufwendiger Prozess. Hierfür werden Primzahltest-Algorithmen eingesetzt, auf die ich in meiner Arbeit nicht eingehen werde.

6.2 Schlüsselspeicherung

Eine sichere Aufbewahrung des Schlüsselpaars ist eine nicht triviale Angelegenheit. Wie schon in Kapitel 5.4 Zertifizierungsdiensteanbieter erwähnt ist die CA für die Verwaltung, Verteilung und Aufbewahrung des öffentlichen Schlüssels verantwortlich. Für die sichere Aufbewahrung des privaten Schlüssels ist jedoch der User selbst verantwortlich. Die sicherste Methode wäre das Aufbewahren des Schlüssels im Gedächtnis. Jedoch sind die meisten Menschen dazu nicht fähig. Deshalb gibt es eine Vielzahl an technischen Hilfsmitteln die das Speichen und Anwenden eines privaten Schlüssels ermöglichen. Auf Smartcards und USB-Tokens werde ich im Kapitel 7 Hilfsmittel zur digitalen Signierung näher eingehen.
Häufig werden private Schlüssel jedoch in Dateien auf PC gespeichert. Diese Dateien müssen besonders vor unrechtmäßigen Zugriffen und Veränderungen geschützt werden. Hierbei kann beispielsweise der Zugriff durch Passwörter geschützt werden oder der private Schlüssel nicht in Klartext gespeichert werden.

6.3 Schlüsselvernichtung

Die Schlüsselvernichtung ist eine nicht zu unterschätzende Aufgabe der meistens zu wenig Aufmerksamkeit gewidmet wird. Das kann zu Sicherheitsproblemen führen. Beim Löschen einer Datei in einem herkömmlichen Betriebssystem wird diese nicht gelöscht. Es wird lediglich der Speicherbereich auf der Festplatte wieder freigegeben und der Eintrag über die Datei in der Dateitabelle wird gelöscht. Die Datei befindet sich aber nach wie vor auf der Festplatte. Um diese Datei sicher zu löschen, muss der von ihr besetzte Speicherbereich mehrfach überschrieben werden. Weiters muss man dar-

auf achten, dass alle etwaigen Kopien, die von dem System in Cache, im virtuellen Speicher usw. angelegt wurden, ebenfalls gelöscht und mehrfach überschrieben werden.

6.4 Schlüsselaustausch

Der sichere Austausch kryptographischer Schlüssel ist besonders in der symmetrischen Kryptographie ein großes Problem. Ich werde auf die Lösungen dieser Problematik in meiner Arbeit nicht näher eingehen.

Doch auch in der asymmetrischen Kryptographie kann der sichere Schlüsselaustausch Probleme mit sich bringen. Der Austausch des öffentlichen Schlüssels ist wie schon in Kapitel 5.4 Zertifizierungsdiensteanbieter erwähnt über eine CA und deren Schlüssellisten geregelt. Bei richtiger Handhabung können dadurch nur schwer Probleme entstehen. Die meisten Probleme entstehen bei dem Austausch von symmetrischen Session Keys. Wird hierbei der Session Key nur verschlüsselt und nicht signiert, dann ist die Übertragung anfällig für Man-in-the-Middle-Angriffe. Dies lässt sich jedoch leicht durch die Signierung des Schlüssels oder eines Hashwerts verhindern.

6.5 Schlüssellänge

Schlüssellängen sind sehr wichtig. Wie wir schon in Kapitel 1.2.5.2 Mathematische Hintergründe des RSA-Verfahrens erfahren haben, existiert keine sichere Verschlüsselung. Jede Verschlüsselung ist, wenn man genug Zeit zu Verfügung hat, zu überwinden. Die Frage ist also nicht ist die Verschlüsselung sicher, sondern ist die Verschlüsselung mit vertretbarem Aufwand zu überwinden.

Aufgrund der steigenden Computerleistung und der Weiterentwicklung der Algorithmen müssen Schlüssel um als „sicher“ zu gelten immer länger werden. Heutzutage gilt ein 1024 Bit Schlüssel nur mehr als mittlere Sicherheitsgrenze. Informationen über empfohlenen Schlüssellängen finden sich im Internet unter www.keylength.com

Kapitel 7

Hilfsmittel zur digitalen Signierung

Ich habe in den vorigen Kapiteln erläutert wie man ein digitales Zertifikat erhält, mit welche Bedingungen seine Ausstellung verbunden ist und noch vieles mehr. Ich werde im folgenden Kapitel nun darauf eingehen wie digitale Zertifikate im Alltag benutzt werden können. Ich werde hierfür auf zwei Technische Umsetzungen näher eingehen:

- Smartcards
- USB-Token

7.1 Smartcards

Eine Smartcard ist eine Weiterentwicklung einer herkömmlichen Chipkarte. Sie unterscheidet sich jedoch wesentlich von dieser. Eine Smartcard ist mit einem Mikroprozessor und einem programmierbaren Speicher ausgerüstet. Ihre Verwendung beschränkt sich jedoch nicht nur auf digitale Signaturen.

Man findet Smartcards heutzutage beinahe überall: Kreditkarten, Ausweise, SIM-Karten, usw.
Smartcards basieren auf dem ISO 7816 Standard. Dieser ist in folgende Bereiche geteilt, welche sich mit diesen Gebieten beschäftigen:

1. Physikalischen Eigenschaften
2. Dimensionen und Positionen der Kontakte
3. Elektronische Signale und Übertragungsprotokolle
4. Anwendungsebene
5. Application Identification (AID)
6. Datenobjekten
7. Structured Card Query Language (SCQL)
8. Internes Sicherheitsmanagement
9. Befehle der Karte, Datenzugriffskontrolle und Lebenszyklus der Karte
10. Elektronische und Reset Signale
11. Identifikation der User
12. USB als Schnittstelle
13. Kommandos für die Verwaltung
14. existiert nicht
15. Kryptografischen Informations Anwendungen

Ich werde in meiner Arbeit nicht auf all diese Bereiche näher eingehen. Ich werde mich jedoch bemühen, Smartcards und deren Funktionsweise ausreichen zu erläutern.

7.1.1 Architektur von Smartcards

Eine Smartcard besteht aus:

- Plastikkarte, auf der die anderen Teile befestigt werden.
- Mikrocontroller, auf den ich im nächsten Unterpunkt näher eingehen werde.
- Optional: Magnetstreifen, auf dem sich allgemeine Daten wie z.B. das Verfallsdatum der Karte befinden.
- Optional: Foto, Beschriftungen, Einstanzungen auf der Karte.

Smartcards existieren in drei Größen, welche als ID-1, ID-00, ID-000 bezeichnet werden.

- ID-1 ist die Standardkarte in der Größe eine Chipkarte wie man sie aus dem Alltag kennt. Sie ist 85.6mm x 54mm groß.
- ID-00 ist die sogenannte Minikarte. Sie ist 66mm x 33mm groß und bis auf ihrer Größe mit der ID-1 Karte ident. Es wurde lediglich Plastik eingespart. Eine Besonderheit ist, dass die Kontakte bei den ID-1 und ID-00 Karten gleich angeordnet sind, wodurch die gleichen Kartenlesegeräte verwendet werden können.
- ID-000 ist eine Plug-in-Karte. Sie ist gerade mal 15mm x 25mm groß. Ihr bekanntester Anwendungsbereich ist die SIM-Karte bei Mobiltelefonen.

7.1.2 Mikrocontroller von Smartcards

Der Smartcard-Mikrocontroller hat eine ähnliche Zusammensetzung wie ein Computer. Er besteht aus:

- CPU (Central Processing Unit):
  Die CPU ist der Hauptprozessor der Smartcard. Wie bei einem Computer ist er für die Berechnungen verantwortlich. High-End-Smartcards verwenden bereits 32-Bit Prozessoren. In der Regel werden aber schwächere verwendet.

- ROM (Read Only Memory):
  ROM kann sehr unterschiedliche Größen haben. Bei einfachen Smartcards reichen bereits 32 KByte. High-End-Smartcards können aber bis zu 320 KByte ROM haben. Wie der Name bereits verrät, kann ROM nur gelesen und nicht geschrieben werden. Außerdem benötigt er keinen Strom um die Informationen zu behalten. Diese wird mit Hilfe von Licht oder Säure auf den ROM gespielt. Normalerweise enthält der ROM das Betriebssystem, Verfahren zur PIN Überprüfung, kryptografische Verfahren zum Verschlüsseln, Signieren, zur Berechnung des Hashwertes und verschiedene Test und diagnostische Funktionen.

- RAM (Random Access Memory):
  RAM kann ebenfalls sehr unterschiedliche Größen haben. Er reicht von 256 Byte bis zu 16 KByte. RAM ist ein sogenannter flüchtiger Speicher. Das heißt, der Inhalt geht verloren sobald der Strom weg ist. Wie beim Computer dient das RAM bei der Smartcard dem CPU als Arbeitsspeicher. RAM ist ein sehr schneller Speicher. Lese- und Schreibzugriffe dauern nur wenige Mikrosekunden. Außerdem kann er beliebig oft beschrieben werden.

- I/O Kanäle:
  Sind die Verbindung des Mikrocontrollers zur Außenwelt. Für den Datentransfer gibt es zwei standardisierte Protokolle, die mit T=0 und

T=1 bezeichnet werden.

T=0 ist Byte orientiert. Das bedeutet, dass ein Byte die kleinste Einheit ist die zwischen Smartcard und Terminal ausgetauscht werden kann. Jede Transmission besteht aus einem Header, welcher ein Klassen, ein Kommando und drei Parameter Bytes enthält. Danach folgt der Body, welcher aus den zu übertragenden Daten besteht. Aufgrund der Byte Orientierung muss bei einem Übertragungsfehler eines Bytes, sofort eine Neuübertragung dieses angefordert werden.

T=1 ist Block orientiert. Das bedeutet, dass ein Block die kleinste Einheit ist, die zwischen Smartcard und Terminal ausgetauscht werden kann. Terminal und Smartcard senden sich abwechselnd Blöcke. Es gibt drei verschiedene Arten von Blöcken: Informationsblöcke, Empfangsbestätigungsblöcke und Systemblöcke. Das T=1 Protokoll ist wesentlich komplexer als das T=0 Protokoll.

- EEPROM (Electrically Eraseable Programmable ROM):
  EEPROM entspricht der Festplatte eines Computers. Er reicht von 8 KByte bei billigen bis 400 KByte bei High-End-Smartcards. Wie der ROM ist er ein nicht flüchtiger Speicher. Er benötigt also keinen Strom um seinen Inhalt zu behalten. EEPROM ist so konstruiert, dass er mittels elektrischer Signale bis zu 100.000-mal umprogrammiert werden kann. Deshalb dient er der langfristigen Speicherung von veränderbaren Daten (z.B. PIN, Session Keys). In neueren Modellen von Smartcards wird EEPROM durch Flash Speicher ersetzt.

- Interne Bus:

Der Interne Bus dient dem Datenaustausch zwischen ROM, RAM, EEPROM und CPU.

7.1.3 Betriebssystem bei Smartcards

Wie jeder normale Computer benötigen auch Smartcards ein Betriebssystem. Dieses muss jedoch mit sehr geringen Systemanforderungen zurechtkommen, da bei Smartcards die Leistung eingeschränkt ist. Außerdem werden bei Smartcards sehr große Stückzahlen produziert, was Lizenzkosten zu einem großen Hindernis macht. Deshalb werden kaum Standard-Betriebssysteme in Smartcards benutzt. Es gibt von Linux bereits seit längerer Zeit ein Projekt ein Betriebssystem für Smartcards zu entwickeln. Leider sind bis jetzt dessen Systemanforderungen noch zu hoch. Durch die rasante Weiterentwicklung der Smartcards ist es jedoch möglich, dass dieses bald erhältlich sein wird. Es gibt einige Smartcard Betriebssysteme. Das bekannteste ist das Small-OS. Ich werde in meiner Arbeit nicht genauer darauf eingehen, da dies den Rahmen meiner Arbeit sprengen würde.
Die Aufgaben eines Betriebssystems sind vielfältig. Es ist z.B. für das Dateisystem, die Zugriffe auf Dateien, für den I/O Manger, für den Zugriff auf Interne Geheimnisse (PIN und andere Schlüssel) und für vieles mehr verantwortlich.

7.1.4 Sicherheit bei Smartcards

Smartcards gelten als sicher per Design. Sie verfügen über eine Vielzahl von Sicherheitsvorkehrungen, die die Karte vor unerlaubten Zugriffen und Löschung schützen sollen. Einige davon sind:

- Die internen Busse des Microcontrollers führen nicht nach außen und sind deshalb von außen auch nicht kontaktierbar. Dadurch können Adress-, Daten-, und Steuerbus weder abgehört noch beeinflusst werden. Zusätzlich werden die Busse auch noch gescrambelt, was die Funktion der Busse nach außen verschleiert.

- ROM befindet sich in tieferen Schichten des Microcontrollers, um eine physische Modifikation unmöglich zu machen. Weiters wird der Inhalt des ROMs ionenimplantiert. Dadurch wird das Auslesen mit Hilfe eines Lasermikroskops unmöglich, da die Inhalte in keinem Licht sichtbar sind.

- EEPROM wird speziell abgeschirmt, um elektrische Abstrahlungen zu verhindern, aus denen man auf deren Inhalt schließen könnte. Die Entfernung dieser Schutzschicht würde zur Zerstörung des gesamten Chips führen.

- Eine weitere Schutzschicht verhindert, dass die Speicherinhalte mit Hilfe von UV-Strahlung gelöscht werden können.

- Um das Messen des Stromverbrauchs des Microcontrollers bei der Verarbeitung von Daten zu verhindern, wird ein Spannungsregler eingebaut, um einen von der Berechnung der Daten unabhängigen Stromverbrauch zu sichern.

- Smartcards verfügen über Sensoren die unerlaubte Zugriffe feststellen sollen.

  - Spannung wird gemessen.
  - Sensoren, die auf Licht reagieren.

  - Wärmesensoren, die eine Übertaktung des Chips verhindern.
  - ...

- Manchmal werden zusätzliche Prüfsummen und Signaturen eingesetzt, um dem privaten Schlüssel ein zusätzliches Maß an Sicherheit zu gewähren.
- Optional können Smartcards alle internen Übertragungen verschlüsseln, wodurch passive Angriffe abgewehrt werden können.

Auf Smartcards werden auch heute noch häufig symmetrische Verfahren zur Verschlüsselung eingesetzt, da diese wesentlich weniger Rechenleistung als asymmetrische benötigen und in der Ausführung deutlich schneller sind. Besonders die Generierung eines RSA-Schlüsselpaars in einer sinnvollen Länge ist für die meisten Smartcards ohne zusätzliche Hardware zu aufwändig. Deshalb können Schlüsselpaare auch offcard berechnet werden und nachträglich auf die Karte gespielt werden. Ich werde darauf in Kapitel 7.1.6 Signaturcards näher eingehen.

7.1.4.1 Angriffe auf Smartcards

Die meisten Angriffe auf Smartcards zielen auf das Ausspähen des privaten Schlüssels ab, welcher auf der Smartcard gespeichert ist. Es gibt viele Angriffe auf Smartcards. Ich werde kurz auf einige eingehen:

- Durch das Anlegen einer unüblichen Spannung oder extremer Temperaturen wird versucht die Funktionsweise des Microcontrollers zu beeinflussen und dadurch eine Änderung im Code vorzunehmen, wodurch der private Schlüssel ermittelt werden kann.

- Durch das Herauslösen der Microcontrollers aus seiner Hülle, soll es mit Hilfe von spezieller Ausrüstung möglich sein, Signale aus dem Datenbus abzugreifen.

- Durch das Messen des Stromverbrauchs des Microcontrollers zuerst bei der Verarbeitung bekannter und dann unbekannter Daten lässt sich auf Daten schließen. Der bekannteste Angriff aus Basis des Stromverbrauchs ist der PIN Vergleich.

- Weitere mögliche Angriffe sind Angriffe auf den Übertragungsweg zwischen Lesegerät und Karte. Hierbei werden aktive und passive Angriffe wie Sniffing und Spoofing ausgeführt.

Gegen all diese Angriffe wurden im Kapitel 7.1.4 Sicherheit bei Smartcards Verteidigungsvorkehrungen erwähnt. Dennoch ist es nicht ausgeschlossen, dass einer dieser Angriffe Erfolg hat.

7.1.5 Signieren mit Smartcards

Es gibt eine Vielzahl an Smartcards, die in unterschiedlichsten Bereichen unseres Lebens zum Einsatz kommen. Doch nicht jede dieser Smartcards ist dafür geeignet, dass mit ihr digitale Signaturen ausgeführt werden.
Damit eine Smartcard als sogenannte Signaturcard eingesetzt werden kann, muss sie nach Deutschem SigG folgende Funktionen haben:

- Protokoll Parameter Selektion (PPS)

- Transmissionsprotokolle (T=0 oder T=1)

- Den Umgang mit Daten, Datenobjekte und Datenformate

- Authentifikation des Kartenbesitzers

- Berechnen und Bestätigen von Signaturen
- Loggen während eine Signatur erstellt wird
- Signaturerstellungs- und Bestätigungsprozesse
- Operationen mit Terminals und die passenden Kommandosequenzen

Doch für eine digitale Signatur benötigt man nicht nur eine Signaturcard. Es wird auch eine CA benötigt, um die Karte auszustellen und später die digitale Signatur zu bestätigen.

7.1.6 Signaturcards

Für das Speichern und Verwenden einer digitalen Signatur wird eine sichere Hardware Umgebung benötigt. Signaturcards sind ideal dafür, da sie klein und billig zu produzieren sind. Weiters sind sie sehr sicher im Zusammenhang mit Zugriffen von außen und Veränderungen gespeicherten Daten.
Damit Signaturcards in der Lage sind alle benötigten kryptographischen Algorithmen auszuführen, benötigen sie eine erweiterte Numeric Processing Unit (NPU). Diese erlaubt, dass Signaturen auf der Karte kreiert und getestet werden können. Dies hat den Vorteil, dass der private Schlüssel außerhalb der Karte nicht existiert, was es unmöglich macht, dass dieser ohne Diebstahl der Karte in fremde Hände gerät. Ein Nachteil ist, dass die Karte nicht ersetzt werden kann. Im Allgemeinen ist es besser die Signierung und die Überprüfung von Signaturen auf der Karte durchzuführen. Es ist aber auch möglich, dass alle kryptographischen Berechnungen offcard durchgeführt werden. Dies hat den Vorteil, dass der Schlüssel schnell und effizient erzeugt werden kann. Die Karte kann jederzeit durch eine neue ersetzt werden. Es bestehen jedoch der große Nachteil, dass der Schlüssel sicher auf und

von der Karte transferiert werden muss und dass der Schlüssel auch außerhalb der Karte existieren muss um Signaturen durchzuführen.

7.1.6.1 Ausstellen eine Signaturcard

Die Ausstellung einer Signaturcard funktioniert beinahe identisch mit der Ausstellung eines digitalen Zertifikats. Es gelten auch dieselben Bestimmungen wie bei digitalen Zertifikaten. Der einzige Unterschied ist, dass bei Signaturcards zusätzlich ein PIN ausgestellt wird, der die Benutzung der Karte ermöglicht. Die Ausstellung eines digitalen Zertifikats wurde schon in Kapitel 5 Digitale Zertifikate behandelt.

Für die CA bedeuten Signaturcards ein neues Arbeitsfeld. Das Speichern und Verwalten des Schlüssels in Listen, die Überprüfung der Identität des Endbenutzers usw. bleibt für die CA zwar gleich. Dennoch muss die CA nun Signaturcards ausstellen. Dafür muss sie diese durch Name, Foto, Daten usw. auf der Karte und in deren Dateisystem personalisieren. Weiters muss der Schlüssel nun gegebenenfalls auf der Karte erzeugt werden und wenn er offcard erzeugt wird, muss er sicher auf die Karte transferiert werden, wodurch neue Infrastruktur benötigt wird.

7.1.6.2 Signieren mit einer Signaturcard

Das Signieren mit Hilfe einer Signaturcard funktioniert nach dem gleichen Prinzip wie die „normale digitale Signatur“, auf die ich schon in Kapitel 3 Digitale Signaturen und im Kapitel 4.4 Beispiel einer Anwendung einer PKI eingegangen bin.
Es wird zum Signieren eine Signaturcard ein Terminal und normalerweise ein PC benötigt. Der einzige Unterschied ist, dass sich der Signator auf

der Signaturcard identifizieren muss. Dies geschieht heute meist über den PIN. In naher Zukunft ist es allerdings denkbar, dass auf andere Mittel der Identifikation, wie zum Beispiel biometrische Daten (z.B. Fingerabdruck), zurückgegriffen wird.
Genau wie bei der normalen digitalen Signatur wird ein Hashwert berechnet, dieser wird anschließend signiert und gegebenenfalls auch verschlüsselt. Das Berechnen eines Hashwerts bei langen Dokumenten auf der Signaturcard ist zwar möglich, es ist aber in den meisten Fällen aufgrund der Hardwareeinschränkungen einer Signaturcard nicht sehr praktisch, da es sehr lange dauern kann. Deshalb wird die Berechnung des Hashwerts meistens offcard durchgeführt und dann auf die Karte transferiert, um dort signiert zu werden. Um ein zusätzliches Maß an Sicherheit hinzuzufügen, ist es möglich den Hashwert nur bis zum letzten Block extern zu berechnen. Die Berechnung wird dann im Anschluss auf der Karte abgeschlossen.

7.1.7 Praktische Anwendung: Bürgercard

Meine Informationen über Bürgercards stammen von http://www.buergerkarte.at/ und aus dem Österreichischen Signaturgesetz.

7.1.7.1 Aktivierung

Die Bürgercard ist völlig kostenlos.
Die Karte wird bei Ablauf des Zertifikats automatisch ausgetauscht. Auf der neuen Karte befindet sich dann ein neues Zertifikat.
Um eine E-Card als Bürgercard zu nutzen ist in erster Linie nur ein Kartenlesegerät erforderlich. Die eigentliche Aktivierung erfolgt über FinanzOnline, in einer der Registrierungsstellen oder mit Hilfe eines RSA-Briefs.

7.1.7.2 Beispiel der Aktivierung

Vor der Aktivierung enthält die E-Card standardmäßig:

- ein Schlüsselpaar der Sozialversicherung (für die Krankenscheinfunktion)
- ein ECDSA-Schlüsselpaar (für ein Zertifikat) - wird momentan nicht weiter verwendet
- ein ECDSA-Schlüsselpaar (für ein qualifiziertes Zertifikat)
  Das ECDSA-Schlüsselpaar hat eine Länge von 192 bit. Bei den neuesten Modellen sind es bereits 256 bit. (Die Unterschiede zwischen einfachen und qualifizierten Zertifikaten wurden schon in Kapitel 5.3 Zertifikate in Österreich angeführt.)

Bei der Aktivierung über FinanzOnline passiert Folgendes:

1. FinanzOnline (als Identitätsprovider) übermittelt an den Zertifizierungsdiensteanbieter A-Trust eine signierte Bestätigung, bestehend aus Namen und Sozialversicherungsnummer. Damit garantiert FinanzOnline die Identität des Benutzers. (Dieser hat sich bei der Registrierung zu FinanzOnline mit einem Ausweis identifiziert).
2. A-Trust liest nun über die Bürgerkarten-Software von der E-Card Namen und Sozialversicherungsnummer aus und vergleicht sie mit der Bestätigung von FinanzOnline. Die Übereinstimmung wird in erster Linie über die Sozialversicherungsnummer hergestellt (daher macht es nichts, wenn die Schreibweise des Namens bei FinanzOnline leicht von der Schreibweise auf der E-Card abweicht).

3. A-Trust liest über die Bürgerkarten-Software von der E-Card den öffentlichen Schlüssel für das qualifizierte Zertifikat aus. Das entsprechende Schlüsselpaar wurde bereits bei der Kartenproduktion erzeugt. Der private Schlüssel verlässt die E-Card überhaupt nie und ist nicht einmal der A-Trust bekannt.

4. Dann sendet A-Trust Namen und Geburtsdatum (die von der E-Card ausgelesen werden) gemeinsam mit dem öffentlichen Schlüssel an das Zentrale Melderegister, um den ZMR-Eintrag abzufragen. Das Problem dabei ist, im ZMR ist die Sozialversicherungsnummer nicht gespeichert. Für diesen Schritt muss also die Schreibweise des Namens auf der E-Card und im ZMR genau übereinstimmen. Für den Fall, dass es mehrere Personen mit selben Namen und selben Geburtsdatum gibt, fragt A-Trust davor auch nach der Postleitzahl.

5. Als Ergebnis der ZMR-Abfrage erhält A-Trust sofort die Personenbindung (d.h. ZMR-Zahl wird nicht an A-Trust weitergegeben).

6. A-Trust sendet die Personenbindung über die Bürgerkarten-Software an die E-Card, wo sie gespeichert wird (gesichert durch den Karten-PIN).

7. A-Trust erzeugt aus dem öffentlichen Schlüssel das qualifizierte Zertifikat und trägt es im Verzeichnisdienst ein.

8. Die Bürgerkarten-Software schreibt das qualifizierte Zertifikat auf die E-Card.

9. Die E-Card sichert den Zugriff auf den privaten Schlüssel mit dem Signatur-PIN.

10. A-Trust erzeugt ein RSA-Schlüsselpaar für das nicht-qualifizierte Zertifikat. Dieses benutzt eine 1536 bit Verschlüsselung.

11. Aus dem öffentlichen Schlüssel erzeugt A-Trust das nicht-qualifizierte Zertifikat und trägt es im Verzeichnisdienst ein.

12. Die Bürgerkarten-Software schreibt das nicht-qualifizierte Zertifikat und den privaten Schlüssel auf die E-Card (gesichert durch den Karten-PIN).

13. A-Trust speichert den privaten Schlüssel des nicht-qualifizierten Zertifikats.

Über FinanzOnline oder in den Registrierungsstellen ist es auch möglich ein Handy als Bürgercard zu aktivieren.
Von der Firma A-Trust gibt es auch die Möglichkeit, dass die Bankomatkarte als Bürgercard verwendet werden kann. Dabei fallen allerdings Kosten an.

7.1.7.3 Funktionen

Die Bürgercard hat eine Vielzahl an Funktionen. Auf http://www.buergerkarte.at/ sind über 100 Funktionen gelistet. Einige davon sind:

- Signieren von E-Mails und PDFs.
- Elektronisches Postamt: Ermöglicht das Senden von eingeschriebenen Briefen online und das sichere Empfangen von Post.
- Persönliche Informationen: Ermöglicht Zugang zu Informationen wie Pensionskonto, Daten zur Krankenversicherung usw.
- Online Amtswege: Die Bürgercard kann über 100 Amtswege ersparen z.B. Diebstahlsanzeigen und Strafregisterbescheinigungen.
- Online-Banking: Die Bürgercard ersetzt die Anwendung von PINs und TANs im Netbanking.

7.1.7.4 Sicherheit bei Bürgercards.

Die E-Card als Bürgercard gilt als sehr sicher, da Smartcards ein hohes Maß an Sicherheit besitzen. Ich bin darauf schon in Kapitel 7.1.4 Sicherheit bei Smartcards eingegangen.
Für die Bürgerkarte am Handy stehen diese Daten in einem Hochsicherheits-Server und werden nur bei Bedarf abgerufen, wodurch auch die Handyanwendung sicher wird.
Außerdem ist Datenschutz bei der Bürgercard gewährleistet. Durch ein Verschlüsselungsverfahren ist sichergestellt, dass es keinen zentralen Zugriff auf sensible Bürger-Daten geben kann.
Das Verschlüsselungsverfahren funktioniert folgendermaßen: Jeder in Österreich gemeldete Bürger ist durch eine Zahl aus dem Zentralen Melderegister (ZMR-Zahl) eindeutig identifizierbar. Diese Zahl ist auf der Bürgerkarte allerdings nicht direkt gespeichert, sondern nur in Form der so genannten Stammzahl. Diese Stammzahl ist durch das Triple-DES-Verfahren verschlüsselt, so dass die ursprüngliche ZMR-Zahl nicht daraus gebildet werden kann. Darüber hinaus ist sogar noch eine weitere Sicherheitsvorkehrung eingebaut. Um zu verhindern, dass verschiedene staatliche Stellen unberechtigter Weise in Verfahren Einblick nehmen können, wird bei der Bürgerkarten-Benutzung auch die Stammzahl nicht direkt verwendet. Tatsächlich wird die Stammzahl ein weiteres Mal unkenntlich gemacht (mit Hilfe des SHA-1 Verfahren auf das ich schon in Kapitel 2 Hashfunktionen eingegangen bin). Das geschieht auf unterschiedlich Weise für verschiedene Verwaltungsbereiche. Dazu werden so genannte „Bereichskürzel“ verwendet. Das Resultat dieser Verschlüsselung ist das so genannte bereichsspezifische Personenkennzeichen (bPK).

7.1.7.5 Praktische Anwendung: Security-Tokens

Bei USB-Tokens handelt es sich ebenfalls um Smartcards.
Sie besitzen im Prinzip dieselben programmierbaren Mikrocomputer mit Prozessor und Speicher, wie man sie auch im Chip einer Smartcard findet. Allerdings können beim USB-Tokens preiswertere Standardbauteile verwendet werden, da der Hersteller mehrere Millimeter Dicke nutzen kann. Trotzdem verfügen sie (bei komplexeren Modellen) über das gleiche Maß an Sicherheit wie Smartcards. Siehe Kapitel 7.1.4 Sicherheit bei Smartcards. USB-Tokens haben einen weiteren Vorteil. Sie können direkt an den USB-Port eines Computers angeschlossen werden, wodurch kein Kartenlesegerät benötigt wird.

Es existieren auch Security-Token mit anderen Sicherheitsverfahren.
Es gibt zum Beispiel USB-Tokens, die mit einem Display ausgestattet sind. Sobald der Anwender einen Knopf drückt, berechnet der USB-Token einen einmaligen Schlüssel, der aus dem privaten Schlüssel und zumindest einem weiteren Faktor (z.B. der Zeit) berechnet wird (siehe Kapitel 6.1.1 Erzeugen von Zufallszahlen). Dieser Schlüssel wird auf dem Display angezeigt und ist für eine vorher definierte Zeit gültig. Der Sicherheitsserver errechnet denselben einmaligen Schlüssel, wodurch während der definierten Zeit mit Hilfe des einmaligen Schlüssels eine eindeutige Authentifizierung am Sicherheitsserver möglich ist.

Der erste kommerzielle USB-Token war der RSA SecurID 6100 USB Token. Dieser wurde im Sommer 2003 auf den Markt gebracht.

Literaturverzeichnis

- Eckert, Claudia: IT-Sicherheit. 2004
- Eckert, Claudia: IT-Sicherheit. 2009
- Ondarza, Peter von: Digitale Signaturen und die staatliche Kontrolle von Fremdleistungen". 2001
- Steininger, Katharina: elektronische Signaturen 2006
- Schneider, Bruce: Applied Cryptography 1996
- Kowol, Gerhard: Vorlesung Algebra für LAK SS. 2008
- Merz, Michael: E-Commerce
- Best, E.: Vorlesungsskript Kryptographie. 2004
- NIST: FIPS Pub. 180-3. 2008
- Bundeskanzleramt: E-Government http://www.bka.gv.at/site/5226/default.aspx. 2011
- NIST: X.509 PKI Certificate RFC5280. 2008
- Sun Microsystem Inc. Public key Infrastructure Overview. 2001

- NIST: Introduction to Public Key Technology and the Federal PKI Infrastructure. 2001
- RIS: Gesamte Rechtsvorschrift für Signaturgesetz. Fassung 02.03.2011
- Bundschuh, Peter: Einführung in die Zahlentheorie. 2008
- Baxa, Christoph: Vorlesung Zahlentheorie. 2008
- Common Weakness Enumeration http://cwe.mitre.org/. 2011
- Rankl and Effing: Smart Card Handbook. 2010
- ISO 7816. 2010
- NIST: FIPS Pub. 201-1 PIV. 2006
- http://www.buergerkarte.at/
- www.a-trust.at/

Bilderverzeichniss

- Abbildung 5.2: http://winfwiki.wi-fom.de/images/3/30/ITZertifikat.jpg
- Abbildung 7.1: Stefan Birkner http://de.wikipedia.org/w/index.php?title=Datei:Man-in-the-middle_attack_of_Diffie-Hellman_key_agreement.svg&filetimestamp=20110228162837

Ich habe mich bemüht, sämtliche Inhaber der Bildrechte ausfindig zu machen und ihre Zustimmung zur Verwendung der Bilder in dieser Arbeit eingeholt. Sollte dennoch eine Urheberrechtsverletzung bekannt werden, ersuche ich um Meldung bei mir.

Glossar

Im Folgenden beschreibe ich einige Begriffe, die in der Arbeit vorkommen. Die Definitionen beziehe ich Großteils aus Internetquellen (z.B. Wikipedia) und aus den im Literaturverzeichnis angeführten Quellen.

- **Brute-Force-Attack:**

  Der Brute-Force-Attack ist eine Lösungsmethode für Probleme aus den Bereichen Informatik und Kryptologie. Sie beruht auf dem Ausprobieren aller möglichen Fälle.

  In der Kryptoanalyse, kann die Methode verwendet werden, um alle möglichen Schlüssel durchzuprobieren.

  Die Reihenfolge der Probe-Schlüssel wird gegebenenfalls nach ihrer Wahrscheinlichkeit ausgewählt. Dies ist jedoch bei zufällig generierten Schlüsseln wenig hilfreich. Die Schlüssellänge spielt eine entscheidende Rolle. Mit zunehmender Länge des Schlüssels steigt der Umfang des Schlüsselraums, also der Menge aller möglichen Schlüssel, exponentiell an. Es ist hier nach den Regeln der Wahrscheinlichkeit zu erwarten, dass im Mittel die Hälfte aller möglichen Schlüssel des Schlüsselraums ausprobiert werden muss, bis der verwendete Schlüssel gefunden ist. Angriffe dieser Art auf moderne Verschlüsselungsalgorithmen bei Verwendung ausreichend langer Schlüssel sind in der Praxis aussichtslos, da der erforderliche Rechenaufwand zu groß wäre. Da aber die Leistung

moderner Hardware immer mehr steigt und dadurch der Zeitaufwand für das Durchprobieren aller Schlüssel einer bestimmten Länge erheblich reduziert wird, muss die Schlüssellänge ausreichend groß gewählt werden, um einem Brute-Force-Attack keine Chance zu geben.

- **Diffie Hellman Verfahren:**
  Das Verfahren wurde von Diffie und Hellman bereits 1976 entwickelt. Es ist eine Art Mittelweg zwischen symmetrischer und asymmetrischer Kryptographie. Beim Diffie Hellman Verfahren errechnen die Kommunikationspartner gemeinsam einen geheimen Sitzungsschlüssel. Dies hat den Vorteil, dass der Schlüssel nicht über einen unsicheren Kanal übertragen werden muss. Das Verfahren beruht auf dem diskreten Logarithmusproblem. Dies wird als dezentrale Berechnung bezeichnet. Die Berechnung eines gemeinsamen geheimen Schlüssels teilt sich in drei Schritte (Ich werde mich bei der Berechnung an Eckert halten):

  1. Jeder Teilnehmer i wählt eine Zufallszahl $1 \leq X_i \leq q-1$. X_i ist der geheime Schlüssel von Teilnehmer i. Der Schlüssel X_i entspricht dem privaten Schlüssel eines asymmetrischen Kryptosystems.
  2. Teilnehmer i berechnet

     $$Y_i = a^{X_i} \mod q$$

     Y_i ist der öffentliche Schlüssel von Teilnehmer i.
  3. Möchte Teilnehmer i mit dem Teilnehmer j kommunizieren, so berechnet jeder Teilnehmer einen Sitzungsschlüssel $K_{i,j}$ bzw. $K_{j,i}$. Zur Berechnung von $K_{i,j}$ benötigt i den öffentlichen Schlüssel Y_j von j. Damit berechnet er:

     $$K_{i,j} = Y_j^{X_i} \mod q$$

Der Teilnehmer j berechnet den Schlüssel $K_{j,i}$ analog mit dem öffentlichen Schlüssel Y_i von i:

$$K_{j,i} = Y_i^{X_j} \mod q$$

Zusammen gilt dann:

$$K_{i,j} = (a^{X_j})^{X_i} \mod q = a^{X_j X_i} \mod q = a^{X_i X_j} \mod q = K_{j,i}$$

Beide Kommunikationspartner haben unabhängig voneinander einen gemeinsamen geheimen Schlüssel berechnet, der nun für die Kommunikation eingesetzt werden kann.

- **Elliptic Curve Digital Signature Algorithm (ECDSA):**
  Der Digital Signature Algorithm (DSA) wurde vom National Institute of Standards and Technology (NIST) im August 1991 für die Verwendung in deren Digital Signature Standard (DSS) empfohlen.
  Die Übertragung des DSA auf elliptische Kurven wird als ECDSA bezeichnet und ist in ANSI X9.62 standardisiert.

- **Flash Speicher:**
  Flash-Speicher sind digitale Speicherchips. Die genaue Bezeichnung lautet Flash-EEPROM. Sie gewährleisten eine nichtflüchtige Speicherung bei gleichzeitig niedrigem Energieverbrauch. Flash-Speicher sind portabel und miniaturisiert, es lassen sich jedoch im Gegensatz zu gewöhnlichem EEPROM-Speicher bei neuen Flash-EEPROM Bytes nicht einzeln löschen. Flash-Speicher sind auch langsamer als gewöhnliche Festwertspeicher (ROM).
  Bei einem Flash-EEPROM-Speicher wird Information (Bits) in einer Speichereinheit (Speicherzelle) in Form von elektrischen Ladungen auf einem Transistor gespeichert.

Am Markt sind derzeit zwei Flash-Architekturen gängig:
NAND-Flash: Bei dem Speicherzellen in größeren Gruppen hintereinander geschaltet sind.
NOR-Flash: Bei dem Speicherzellen über Datenleitungen parallel geschaltet sind.

- **Linux:**
  Als Linux oder GNU/Linux werden in der Regel freie, unix-ähnliche Mehrbenutzer-Betriebssysteme bezeichnet, die auf dem Linux-Kernel und wesentlich auf GNU-Software basieren.
  Das modular aufgebaute Betriebssystem wird von Softwareentwicklern auf der ganzen Welt weiterentwickelt, die an den verschiedenen Projekten mitarbeiten. Es sind sowohl Unternehmen als auch Non-Profit-Organisationen und Einzelpersonen beteiligt.

- **Man-in-the-Middle-Attack:**
  Ein Man-in-the-middle-Angriff (MITM-Angriff), ist ein Angriff bei dem der Angreifer logisch zwischen den beiden Kommunikationspartnern steht. Dadurch hat er mit seinem System vollständige Kontrolle über den Datenverkehr zwischen zwei oder mehreren Netzwerkteilnehmern und kann die Informationen nach Belieben einsehen und sogar manipulieren. Der Vorteil des MITM-Angriffs ist, dass der Angreifer den Kommunikationspartnern das jeweilige Gegenüber vortäuschen kann, ohne dass sie es merken. Die beste Gegenmaßnahme gegen Man-in-the-Middle-Attacks ist das Signieren von Übertragungen. Dadurch wird es unmöglich einem Kommunikationspartner eine falsche Identität vorzugaukeln. Ein bekanntes Beispiel für MITM-Angriffe ist das Diffie Hellman Verfahren. Bei Diffie Hellman ist der Schlüsselaustausch unsicher,

wenn es dem Angreifer (in unserem Beispiel Mallory) es schafft die von Alice und Bob gesendeten Nachrichten abzufangen und stattdessen jeweils eine eigene Nachricht zu senden.

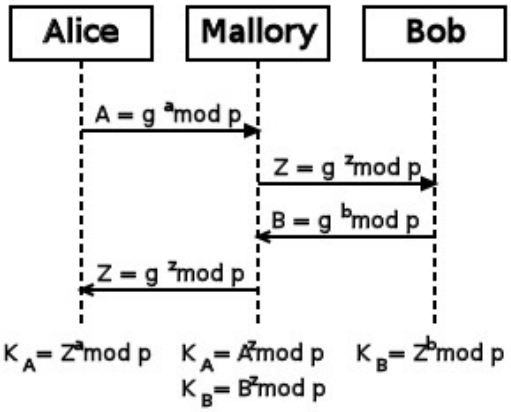

Abbildung 7.1: Bsp.:Man-in-the-Middle-Attack

Nach dem Schlüsselaustausch besitzen die beiden Kommunikationspartner Alice und Bob unterschiedliche Schlüssel K_A und K_B. Im Prinzip wurde zweimal ein Diffie-Hellman-Schlüsselaustausch durchgeführt: einmal zwischen Alice und der Angreiferin Mallory und einmal zwischen Mallory und Bob. Dabei erlangt Mallory Kenntnis der beiden Schlüssel K_A und K_B. Da Alice und Bob im weiteren Verlauf davon ausgehen, mit dem jeweils Anderen zu kommunizieren, kann Mallory die folgende symmetrisch verschlüsselte Kommunikation abhören. Diese leitet sie dazu weiterhin über sich selbst um.

- **MIPS Jahr:**

  MIPS Jahr ist eine Angabe über Rechenzeit bei Computern. 1 MIPS Jahr bedeutet 1 Million Instruktionen pro Sekunde über die Dauer eines Jahres. Das heißt 1 MIPS Jahr hat $30 * 10^{12}$ Instruktionen.

- **Scrambeln:**

  Ein Scrambler verwendet Schieberegister, um ein Digitalsignal nach einem relativ einfachem Algorithmus zu verschlüsseln. Erreicht wird eine Änderung der Bitreihenfolge, wobei es statische und dynamische Verfahren gibt. Allgemein gesprochen wird das Digitalsignal in ein Pseudo-Zufalls-Signal mit demselben Informationsgehalt und derselben Bitrate verwandelt. Ziel des Verschlüsselungsverfahrens eines Scramblers ist oft die Verringerung der Störanfälligkeit bei der Datenübertragung, aber auch Video- oder Audiodaten werden gescrambelt, um eine unbefugte Rezeption der Klartext-Informationen durch nicht autorisierten Empfang zu verhindern.

- **Session Key:**

  Ein Session Key ist ei symmetrischer Schlüssel der zur einmaligen Verwendung gedacht ist. Für diese eine Session werden alle Nachrichten mit diesem Schlüssel verschlüsselt. Es ist allerdings auch möglich während der Session den Schlüssel zu wechseln. Die wird allerdings nur gemacht wenn dafür ein Grund vorliegt (z.B. ein hohes Risiko auf Angriffe). Gründe warum Session Keys verwendet werden sind, dass einige Angriffen leichter werden desto mehr Verschlüsseltes Material man besitzt. Außerdem Ist asymmetrische Kryptographie für das Versenden von Nachrichten meistens zu langsam. Session Keys müssen zufällig gewählt werden damit sie von Angreiffern nicht vorausgesagt werden können. Siehe Kapitel 6.1 Schlüsselerzeugung.

- **Small World Phenomenon:**

  Das Small World Phenomenon ist ein von Stanley Milgram 1967 geprägter sozialpsychologischer Begriff, der innerhalb der sozialen Vernet-

zung in der modernen Gesellschaft den hohen Grad abkürzender Wege durch persönliche Beziehungen bezeichnet. Er bezeichnet eine Hypothese, nach der jeder Mensch (sozialer Akteur) auf der Welt mit jedem anderen über eine überraschend kurze Kette von Bekanntschaftsbeziehungen verbunden ist.

Das Small World Phenomenon lässt sich auch auf andere Netzwerke und Graphen übertragen, wie insbesondere seit Ende der 1990er Jahre die mathematisierte Netzwerkforschung zu zeigen versucht. Das Grundprinzip ist, dass einzelne Objekte, z. B. Personen, als Knoten repräsentiert sind, zwischen denen eine Kante besteht, wenn zwischen ihnen eine bestimmte Beziehung (beispielsweise Bekanntschaft) besteht. Im Jahr 2008 haben die Microsoft-Wissenschaftler Jure Leskovec und Eric Horvitz die These von der Kleinen Welt auf Basis eines Netzwerkes unter Instant-Messenger-Nutzern (180 Millionen Knoten mit 9,1 Milliarden Kanten) empirisch bestätigen können.

- **Sniffing:**

  Ein Sniffer ist eine Software, die den Datenverkehr eines Netzwerks empfangen, aufzeichnen, darstellen und ggf. auswerten kann.

  Es ist von der Netzwerkstruktur abhängig, welche Daten ein Sniffer sehen kann. Werden die Computer mit Hubs verbunden, kann sämtlicher Traffic von den anderen Hosts mitgeschnitten werden. Wird ein Switch verwendet, ist nur wenig oder gar kein Datenverkehr zu sehen, der nicht für das sniffende System selbst bestimmt ist.

  Es gibt mehrere Gründe, einen Sniffer zu benutzen: Diagnose von Netzwerkproblemen, Eindringungsversuche entdecken, Netzwerktraffic-Analyse und Filterung nach verdächtigem Inhalt und natürlich Datenspionage.

- **Spoofing:**

  Spoofing nennt man in der Informationstechnik verschiedene Täuschungsversuche in Computernetzwerken zur Verschleierung der eigenen Identität. Personen werden in diesem Zusammenhang auch gelegentlich als Spoofer bezeichnet.

  Heutzutage umfasst Spoofing alle Methoden, mit denen sich Authentifizierungs- und Identifikationsverfahren untergraben lassen, welche auf der Verwendung vertrauenswürdiger Adressen oder Hostnamen in Netzwerkprotokollen beruhen.

  Eine besondere und gleichzeitig die Bekannteste Anwendung des Spoofing ist das sogenannte Phishing.

- **Triple Des:**

  Der Data Encryption Standard (DES) ist ein weit verbreiteter symmetrischer Verschlüsselungsalgorithmus. Der DES-Algorithmus wurde als offizieller Standard für die US-Regierung im Jahr 1976 bestätigt und wird seither international vielfach eingesetzt. Heute wird DES aufgrund der verwendeten Schlüssellänge von nur 56 Bits für viele Anwendungen als nicht ausreichend sicher erachtet. Die Schlüssellänge kann durch Mehrfachanwendung des DES jedoch auf einfache Weise vergrößert werden. Der Triple-DES (auch 3DES oder DESede genannt) führt DES mehrfache mit zwei verschiedenen Schlüsseln aus. Dieses Verfahren, wurde bereits vom DES Mitentwickler Walter Tuchman beschrieben und analysiert.

  Bei DES handelt es sich um einen symmetrischen Algorithmus. DES funktioniert als Blockchiffre, jeder Block wird also unter Verwendung des Schlüssels einzeln chiffriert. Die Blockgröße beträgt 64 Bits, das heißt ein 64-Bit-Block Klartext wird in einen 64-Bit-Block Chiffre-

text transformiert. Auch der Schlüssel, der diese Transformation kontrolliert, besitzt 64 Bits. Jedoch stehen dem Benutzer von diesen 64 Bits nur 56 Bits zur Verfügung; die übrigen 8 Bits (jeweils ein Bit aus jedem Byte) werden zur Fehlerüberprüfung benötigt. Die effektive Schlüssellänge beträgt daher nur 56 Bits. Die Entschlüsselung wird mit dem gleichen Algorithmus durchgeführt, wobei die einzelnen Rundenschlüssel in umgekehrter Reihenfolge verwendet werden.

- **Zufallsereignis:**

  Der Begriff Zufallsereignis bezeichnet in der Wahrscheinlichkeitstheorie eine Menge möglicher Ergebnisse eines Zufallsexperiments. Bei einem Würfelwurf entspricht beispielsweise das Ereignis eine gerade Zahl würfeln der Menge {2,4,6}, eine Teilmenge aller möglichen Ergebnisse. Man spricht davon, dass ein Ereignis eintritt, wenn eines seiner Elemente Ausgang eines Zufallsexperiments ist. In der Kryptographie benutzt man Zufallsereignisse die regelmäßig vorkommen wie z.B. atmosphärisches Rauschen, radioaktiver Zerfall usw. Zufallswerte werden daraus gebildet indem man die Zeitdifferenz zwischen zwei aufeinanderfolgenden Ereignissen misst. Um diese Verfahren in Kryptosysteme zu integrieren benötigt man jedoch meist eine spezielle Hardware.

Lebenslauf

Ausbildung:

09 1991 - 07 1995	OVS 2 Wittelbachstrasse 6
09.1995 - 07.2000	GRG 3 Hagenmüllergasse 30
09.1999 - 07.2000	HTL Donaustadtstrasse 45
	Informatik
09.2000 - 06.2004	ORG 1 Helgelgasse 14
10.2004 - 02.2005	FH Technikum Wien
	Elektronik
seit 03.2005	Universität Wien
	Lehramtsstudium Mathematik,
	Informatik und Informatikmanagement