

Seminararbeit für das Proseminar IT/RAK

Sichere Email: PGP und S/MIME

Ihre Lehrveranstaltungsnummer: 401173/3

Ihr/e LehrveranstaltungsleiterIn: Prof. Dr. Roland Wismüller

Tragen Sie hier ihre Gruppendaten ein:

| Name | Matrikelnummer |
|-------------------|----------------|
| Haider Gerald | 0125638 |
| Schachinger Josef | 0125692 |

| | | |
|-------|--|----|
| 1. | Einleitung | 3 |
| 2. | Geschichtliches..... | 4 |
| 2.1 | PGP..... | 4 |
| 2.2 | S/MIME..... | 4 |
| 3. | Verschlüsselungsverfahren..... | 5 |
| 3.1 | Symmetrisch..... | 5 |
| 3.2 | Asymmetrisch..... | 5 |
| 3.3 | Hybride..... | 5 |
| 3.4 | Einweg-Hashing | 6 |
| 3.5 | Digitale Unterschrift..... | 6 |
| 3.5.1 | Wofür überhaupt digitale Unterschrift? | 6 |
| 3.5.2 | Funktionsweise der digitalen Unterschrift | 6 |
| 4. | Algorithmen | 7 |
| 4.1 | DSS/Diffie-Hellman Keys..... | 7 |
| 4.2 | RSA..... | 8 |
| 4.3 | 3DES | 8 |
| 4.4 | IDEA | 8 |
| 4.5 | CAST..... | 8 |
| 4.6 | MD4 - MD5..... | 8 |
| 4.7 | SHA-1..... | 8 |
| 5. | PGP und S/MIME | 9 |
| 5.1 | Verschlüsselung mit PGP..... | 9 |
| 5.2 | Verschlüsselung und digitale Unterschrift mit PGP | 10 |
| 5.3 | Schlüsselverwaltungskonzepte..... | 11 |
| 5.3.1 | Allgemein | 11 |
| 5.3.2 | Web-of-Trust..... | 12 |
| 5.3.3 | Zertifikate | 14 |
| 5.3.4 | Telefonische bzw. persönliche Überprüfung | 16 |
| 5.3.5 | Bekanntgabe über geeignete Medien | 17 |
| 5.4 | S/MIME..... | 17 |
| 5.4.1 | RFC 822 | 17 |
| 5.4.2 | MIME – Multipurpose Internet Mail Extension | 17 |
| 5.4.3 | S/MIME Funktionalität | 19 |
| 5.4.4 | S/MIME – Content Types | 19 |
| 6. | Sicherheitsrisiken | 20 |
| 6.1 | Kompromittierte Passphrasen oder private Schlüssel | 20 |
| 6.2 | Veränderter öffentlicher Schlüssel | 21 |
| 6.3 | Rückstände von gelöschten Dateien..... | 21 |
| 6.4 | Viren und Trojanische Pferde | 21 |
| 6.5 | Auslagerungsdateien | 21 |
| 6.6 | Tempest-Angriffe | 22 |
| 6.7 | Gefälschte Zeitmarkierungen | 22 |
| 6.8 | Mehrbenutzer Systeme..... | 22 |
| 6.9 | Datenverkehrsanalyse..... | 22 |
| 6.10 | Kryptoanalyse..... | 22 |
| 7. | Zusammenfassung | 23 |
| 8. | Bibliographie..... | 24 |
| 9. | Abbildungsverzeichnis | 24 |

Abstract

Ziel dieser Seminararbeit soll es sein, auf Unsicherheiten im E-Mail Verkehr hinzuweisen. Einerseits wollen wir mögliche Ansätze zur Verschlüsselung von Texten anführen. Andererseits dessen Umsetzung in verschiedenen Anwendungen (PGP und S/MIME). Weiters gehen wir auch noch auf die Funktionsweise der digitalen Signatur und deren Umsetzung in PGP bzw. S/MIME ein. Der Einbettung der verschlüsselten/signierten Daten in E-Mails und den dabei entstehenden Problemen wird auch ein Kapitel gewidmet. Schlussendlich werden noch verschiedene Sicherheitslücken der unterschiedlichen Verschlüsselungskonzepte aufgezeigt und Lösungen vorgeschlagen. Dieser Punkt wird ergänzt durch Informationen über gewisse Verhaltensvorschriften, die ein User einhalten soll, um das Risiko eines unberechtigten Zugriffes auf seine Daten durch Dritte zu minimieren.

1. Einleitung

Das Wort „sicher“ in Bezug auf E-Mail ist immer ein zweischneidiges Schwert. Als die ersten Emails verschickt wurden, war Sicherheit kein wesentliches Thema, es rechnete auch nicht jeder mit einer derartigen Ausweitung dieser Möglichkeit zu versenden von elektronischen Briefen. Am Anfang war eine Email auch nicht viel mehr, ein einfacher Text ohne Formatierung, mit einem Empfänger.

Dementsprechend wurde die Übertragungstechnik auch nicht auf Sicherheit ausgelegt.

Durch die immer stärkere Verarbeitung dieser Möglichkeit des Briefverkehrs wurde der Ruf nach neuen Möglichkeiten immer lauter. Der Text sollte Farbe und Struktur bekommen und natürlich war auch Sicherheit ein immer wichtigeres Thema.

Ein Ansatz war, die Übertragungstechnik zu ändern. Bei dieser Überlegung kam man schnell zu dem Schluss, dass hier die technischen Standards schon sehr weit verbreitet waren. Der Änderung dieser Struktur hätte einen hohen finanziellen Aufwand bedeutet. Doch war auch klar, dass mit diesem Standard wohl wenig Sicherheit gewährleistet werden konnte.

Wie schon so oft, gab es für dieses Problem schon seit langem eine Lösung. Cäsar, der immer sehr misstrauisch gegenüber seinen Boten war, entschloss sich seine Nachrichten nicht in Reintext zu verschicken sondern zu verschlüsseln. Auch er konnte wenig an der Übertragungstechnik ändern, doch er hatte einen genialen Einfall. Er verschob einfach jeden Buchstaben im Alphabet um 4 Stellen nach hinten. Damit war der erste Verschlüsselungsmechanismus geboren.

Natürlich handelte es damals um einen sehr einfachen Algorithmus, den wohl heute jeder Laie nach einiger Zeit überlistet hätte. Doch haben sich auch diese Möglichkeiten stark verbessert.

Begonnen wurde mit symmetrischen Ansätzen, bei dem der Sender und Empfänger den gleichen Schlüssel verwenden. Um jedoch das Maß an Sicherheit weiter zu steigern, stellte der Kryptologe Hellman die Behauptung auf, dass es auch die Möglichkeit für ein asymmetrisches Verschlüsseln gibt. Es dauerte einige Jahre bis der erste Mechanismus nach einem Verfahren mit public und private Key umgesetzt wurde. Heutige Software zum Verschlüsseln von E-mails wie PGP und S/MIME setzen auf eine Kombination von symmetrischen, asymmetrischen und hash- Verfahren.

2. Geschichtliches

2.1 PGP

Anfang der 90iger begann Phil Zimmermann mit der Entwicklung von PGP. Ein wichtiger Grund dafür war, dass in den US-Senat eine Gesetzesvorlage eingebracht wurde. In dieser war enthalten, das Verschlüsselungsverfahren, sowohl Software als auch Hardware, eine Hintertüre enthalten müssen, die es dem Staat ermöglicht mitzuhören.

1991 Phil Zimmermann veröffentlichte PGP Version 1.

Die ersten Versionen von PGP waren somit illegal, da sie US-Patentrechte für Public-Key-Cryptography verletzen. Weiter gelten für alle US Versionen US Exportbeschränkungen, die das ausführen untersagen.

Im Frühling 1992 kam PGP V2.3a auf den Markt erstmals mit dem im Jahr 1990 entwickelten IDEA Algorithmus.

Im Jahr 1993 wurde gegen Zimmermann ein Verfahren wegen der Verletzung der Exportbeschränkung eingeleitet. Es stellte sich jedoch heraus, dass es über Umwege, ohne sein zutun ausgeführt wurde. Er wies überdies außerdem bei der Dokumentation explizit darauf hin, dass die Ausfuhr untersagt ist. 1996 wurde das Verfahren gegen ihn eingestellt. Im selben Jahr wurde auch die Firma PGP Inc. gegründet.

Seit 1994 ist es in den USA erlaubt PGP privat zu nutzen. Außerhalb den USA entwickelte sich die Version 2.6i, wobei der Zusatz „i“ für Internation steht. Dabei wurde der Code dieser Version in Buchform abgedruckt und exportiert. V 2.6i war zwar in den USA verboten, jedoch war sie kompatibel mit V 2.6.

In Ländern wie Frankreich, Iran, Irak ist der gebrauch von Verschlüsselungsmechanismen für den Normalbürger nicht erlaubt. Des weiteren gibt und gibt es in den USA und in Deutschland ebenfalls diese Bestrebung.

1997 kommt PGP V 5 auf den Markt und die Firma NAI (Network Associates) übernimmt die PGP Inc. Zimmermann bleibt auch hier Verantwortlich für die Weiterentwicklung von PGP.

Bis zum Jahr 2000 wurde jeder Code einer neuen Version immer bekannt gegeben. Die Intention dahinter war, dass „Freaks“ den Code durchpflügen und mögliche Fehler aufdecken. Des Weiteren wollte man damit zur Steigerung des Vertrauens solcher Verschlüsselungsmechanismen beitragen. Ab Ende des Jahres 2000 hatte die NAI beschlossen nur noch Teile des Codes zu veröffentlichen.

Im Februar gibt Zimmermann bekannt, dass er NAI verlässt. Gleichzeitig versichert er aber, dass die kürzlich auf den Markt gekommenen Version V 7.0.3 die Sicherste Version von PGP sei und keinerlei Hintertüren in den Code eingebaut seien. Als Grund gab er an, dass er sich neuen Themenbereichen widmen möchte.

Er setzte sich zusätzlich für die Weiterentwicklung von OpenPGP ein. Wobei er selbst der Initiator dieses neuen Standards war. OpenPGP wurde 1997 von damaligen PGP V 5 abgeleitet.

Die aktuellste Version von PGP ist V 8.0.2

2.2 S/MIME

S/MIME Version 1 wurde spezifiziert und offiziell vorgestellt im Jahr 1995 von RSA Security, Inc.

S/MIME Version 2 wurde im März 1998 in etlichen RFCs (Request For Comment) definiert - RFC 2311 und RFC 2312

S/MIME Version 3 wurde von der S/MIME Mail Security (SMIME) Workgroup des IETF (Internet Engineering Task Force) veröffentlicht und in den RFCs 2630 to 2634 im Juni 1999 festgeschrieben.

3. Verschlüsselungsverfahren

3.1 Symmetrisch

Das symmetrische verschlüsseln ist die wohl einfachste Variante. Der Absender verschlüsselt seinen Text mit einem geheimen Schlüssel. Der nun verschlüsselte Text kann in einem öffentlichen Kanal versendet werden. Damit nun der Empfänger diesen Text wieder entschlüsseln kann, benötigt dieser denselben Schlüssel. Das Problem ergibt sich jedoch bei der Übermittlung des Schlüssels, falls dieser in die falschen Hände gelangt ist der verschlüsselte Text nicht länger geheim. Somit muss der Key entweder persönlich an den Empfänger übergeben werden, oder ein dementsprechender sicherer Übertragungsweg zur Verfügung stehen. Beides ist in der Praxis nicht immer möglich oder mit einem dementsprechend hohen Aufwand verbunden.

Ein großer Vorteil dieses Verfahrens ist der geringe Aufwand an Rechenleistung, somit können auch große Datenmengen einfach und schnell verschlüsselt werden.

eingesetzte Algorithmen: 3DES, IDEA, CAST

3.2 Asymmetrisch

Der Durchbruch in der Kryptographie war der DH Algorithmus, dieser war der erste asymmetrische Ansatz. Entwickelt wurde diese 1976 von Whitfield Diffie und Martin Hellman.

Der Ansatz dahinter war so einfach wie genial. Jeder Anwender hat seinen eigenen privaten Schlüssel, den er vollkommen geheim hält. Aus diesem privaten Schlüssel generiert man nun sein öffentliches Gegenstück. Man kann nun mit dem public Key des Empfängers eine Nachricht verschlüsseln, die nur mit dem privaten Gegenstück wieder entschlüsselt werden kann. Dabei ist es so gut wie unmöglich von dem öffentlichen Schlüssel auf den privaten zu schließen. Selbst der Absender kann den einmal verschlüsselten Text nicht mehr dechiffrieren.

Der Vorteil bei diesem Verfahren ist, dass sowohl der public Key als auch die verschlüsselte Nachricht über öffentliche Kanäle verteilt werden kann.

Natürlich hat auch diese Art der Verschlüsselung Probleme. Eine Aufgabenstellung ist die Sicherstellung, dass ein public Key auch der dazugehörigen Person entspricht. Falls dies nicht der Fall ist, ist dem Missbrauch Tür und Tor geöffnet. Lösungsansätze sind hier das „Web of Trust“ bei PGP oder eine Zentrale Zertifizierungsstelle bei S/MIME. Beides wird an anderer Stelle noch genauer beschrieben.

Ein weiteres Problem ergibt sich im Rechenaufwand solch asymmetrischer Algorithmen, daher ist auch ein Ansatz die Kombination von symmetrischen und asymmetrischen (hybride) Techniken.

3.3 Hybride

Hybride Verschlüsselungstechniken sind eben genau die Kombination aus symmetrischen und asymmetrischen. Sie vereinen die Vorteile von Beiden. Die Schnelligkeit von den symmetrischen Techniken und die Sicherheit von asymmetrischen. Der genaue Ablauf wird in einem späteren Kapitel genauer beschrieben

Eingesetzte Algorithmen: DSS/Diffie-Hellman, RSA

3.4 Einweg-Hashing

Als Einweg-Hashing Algorithmen werden mathematische Verfahren bezeichnet die aus einer Information mit beliebiger Länge einen Wert bzw. eine Zahl vor fixer Länge, z.B. 160 Bit, berechnen. Wobei sichergestellt sein muss das sich dieser berechnete Wert, ändert sobald eine Änderung an den Ausgangsdaten vorgenommen wird. Wenn sich also in den Eingangsdaten nur ein einziges Bit ändert muss der errechnete Wert (=Hash-Wert) ein anderer sein.

Der heute am meisten eingesetzte Einweg-Hashing Algorithmus ist der SHA-1 Algorithmus. Ein weiterer sehr bekannter Algorithmus ist MD5, dieser sollte aber nicht mehr verwendet werden da bereits im Jahre 1994 von Dr. Paul Van Oorschot beschrieben wurde wie dieser zu „knacken“ ist.

3.5 Digitale Unterschrift

Eine wesentliche Anwendung von asymmetrischer Verschlüsselung ist zweifelsohne die „digitale Unterschrift“.

3.5.1 Wofür überhaupt digitale Unterschrift?

- um den Ursprung der Information zweifelsfrei feststellen zu können
- Überprüfung der Echtheit der Information bzw. feststellen etwaiger Manipulationen an der Information

3.5.2 Funktionsweise der digitalen Unterschrift

Im Prinzip kann die digitale Unterschrift mit einem versiegelten Brief der mit Hand unterschrieben wurde verglichen werden.

Zu aller Erst wird ein digitaler Fingerprint der zu unterschreibenden Information erstellt. Dies erfolgt mittels eines Einweg-Hashing Verfahrens, im Falle von PGP ist das der SHA-1 Algorithmus. Dieser berechnete Fingerabdruck der Nachricht wird anschließend mit dem privaten Schlüssel des Unterzeichners verschlüsselt, damit er einerseits vor Manipulationen geschützt wird und andererseits ist es jetzt jedem Empfänger der Information möglich den Fingerprint mittels des öffentlichen Schlüssels des Unterzeichners zu entschlüsseln. Der Empfänger der Nachricht kann also durch das Vergleichen des in der digitalen Signatur gespeicherten Fingerabdrucks mit dem von ihm neu berechneten, eindeutig feststellen ob die Nachricht manipuliert wurde oder nicht. Da dieser Fingerabdruck nur mit dem öffentlichen Schlüssel des Unterzeichners entschlüsselt werden kann, ist außerdem sichergestellt das diese Information wirklich vom jeweiligen Unterzeichner kommt.

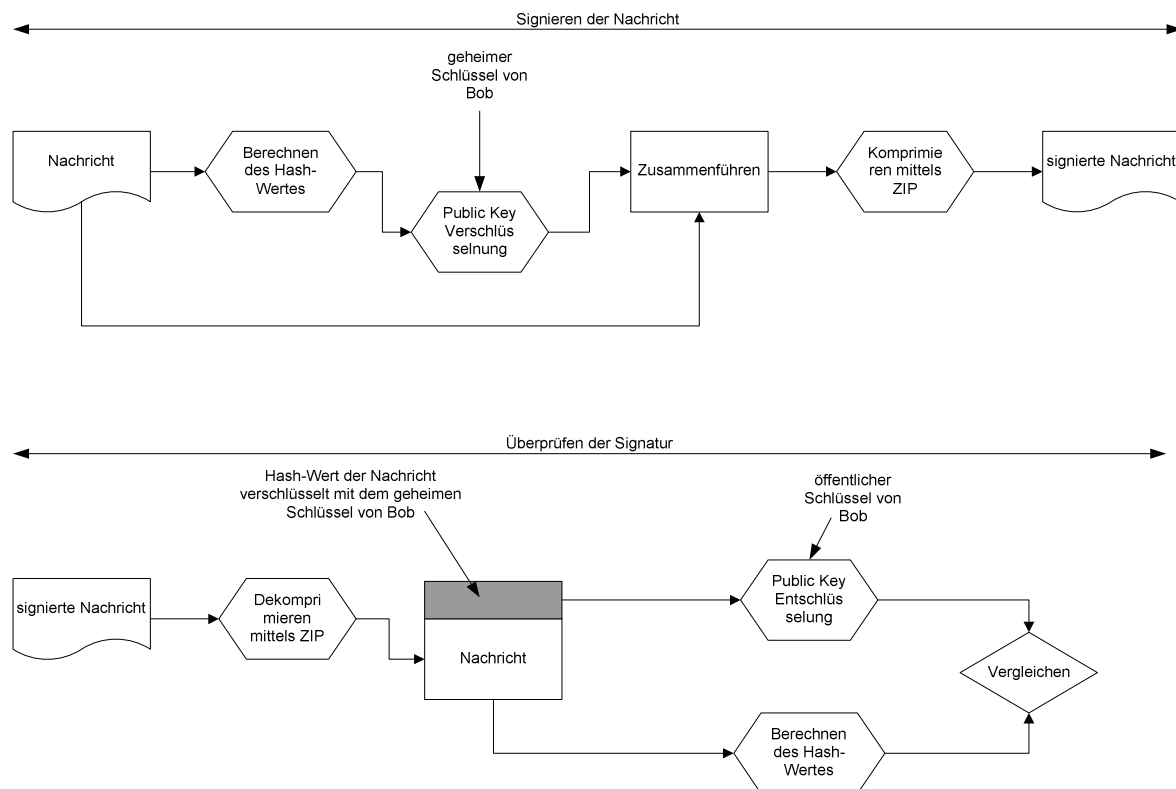


Abbildung 1 - Funktionsweise der digitalen Unterschrift

4. Algorithmen

Die hier gegebene Aufstellung soll nur einen groben Überblick über die einzelnen Verfahren geben. Für eine genauere Beleuchtung wäre der Umfang dieser Arbeit nicht ausreichend. Man darf bei den hier vorgestellten Algorithmen davon ausgehen, dass sie dem derzeitigen Stand der Technik entsprechend, und bei den derzeitigen Entschlüsselungsmechanismen nicht in relevanter Zeit gebrochen werden können. Dies gilt natürlich nur bei der dementsprechenden Kombination und geeigneter Schlüssellänge, soweit erforderlich.

4.1 DSS/Diffie-Hellman Keys

Wird in PGP zum Beispiel seit v5.0 eingesetzt. Der Algorithmus basiert auf DH und eben DSS.

DH ist ein asymmetrischer Verschlüsselungsmechanismus. Dabei werden public und privat Keys verwendet. DH steht für die Anfangsbuchstaben der Erfinder, Whitfield Diffie und Martin Hellman. Sie haben ihn im Jahr 1976 fertiggestellt. Damals war dieser der erste asymmetrische Verschlüsselungsmechanismus. Er wird meist in leicht abgeänderter Form eingesetzt und wird dann ElGamal Algorithmus bezeichnet. Schlüssel zwischen 1024 und 4096.

DSS steht für „Digital Signature Standard“ und ist der Standardalgorithmus DSA von NIST (National Institute for Standards and Technology).

Dieser Algorithmus wurde von NIST (National Institute of Standards and Technology) und NSA (National Security Agency) entwickelt, wobei die US Regierung das Patent auf das Verfahren hat, jedoch ohne Lizenzgebühr für die Benutzung.

4.2 RSA

Der RSA Algorithmus wurde 1977 von Ron Rivest, Adi Shamir und Leonard Adelman entwickelt. Die Anfangsbuchstaben der Entwickler wurden für den Namen des Algorithmus verwendet. Er basiert auf dem asymmetrischen Ansatz und wird in den meisten Versionen von PGP und S/MIME eingesetzt. Dadurch, dass der Algorithmus schon sehr lange eingesetzt wird, gilt er als gut getestet. Dieser Algorithmus wird in PGP v6.5 zur Generierung von Keys zwischen 1024 und 2046 Bit Länge eingesetzt.

4.3 3DES

Wie der Name schon impliziert wird hier 3mal der DES Algorithmus mit jeweils unterschiedlichem Schlüssel eingesetzt.

4.4 IDEA

IDEA ist die Abkürzung für „International Data Encryption Algorithm“. Für diesen Algorithmus wurde das Patent angemeldet, jedoch ist er für den privaten Gebrauch lizenzfrei. Das Patent läuft auf die Firma Ascom Systec. Er basiert auf dem symmetrischen Ansatz und wird in Verbindung mit RSA eingesetzt.

4.5 CAST

CAST wurde von Carlisle Adams und Stanford Tavares entwickelt. Es handelt sich hierbei um einen symmetrischen Verschlüsselungsansatz. Bei diesem Verfahren wird der Klartext in 12 oder 16 Durchgängen in 128 Bit Blöcke verschlüsselt.

4.6 MD4 - MD5

Mit dem MD5 Algorithmus kann die 128 Bit lange Prüfsumme eines Textes erstellt werden. Damit eignet er sich perfekt für die digitale Signatur, um zu überprüfen ob eine Text verändert worden ist. Die Möglichkeit, dass zwei verschiedene Texte den selben Fingerabdruck haben gilt als unwahrscheinlich aber nicht als unmöglich. MD5 steht für Message Digest. Dieser wurde 1991 von Ron Rivest entwickelt und ist der Nachfolger von MD 4 und MD 2.

4.7 SHA-1

SHA-1 steht für „Secure Hash Algorithm 1“. wird in Verbindung mit dem DSS Algorithmus zur Erzeugung eines 160 Bit Hash verwendet. Die ist ähnlich zu MD 5 mit RSA. SHA1 ist zwar sehr aufwendig gilt aber als sicherer als der MD5 Ansatz.

5. PGP und S/MIME

5.1 Verschlüsselung mit PGP

Das wohl wichtigste und bekanntest Feature von PGP ist wohl die Verschlüsselungsfunktion selbst, welche es dem Anwender erlaubt, beliebige Dateien bzw. Nachrichten entweder nur lokal, oder für die Übertragung über unsichere Medien (z.B.: Internet) zu verschlüsseln. Für beide Aufgaben könnte man natürlich auch herkömmliche symmetrische Verschlüsselung verwenden, doch dabei steht der Benutzer immer vor dem Problem wie er den Schlüssel sicher zum Empfänger der Nachricht bringen kann. In PGP wird dieses Problem dadurch umgangen das zwar konventionelle Verschlüsselung verwendet wird, der dabei benutzte Schlüssel aber nur genau einmal für die jeweilige Nachricht benutzt wird. Dieser Schlüssel wird aus verschiedenen Zufallsdaten (z.B. Benutzereingaben, die Zeitabstände zw. zwei Eingaben) mittels eines Zufallsgenerators erzeugt. Damit der Empfänger die Nachricht entschlüsseln kann benötigt er klarerweise diesen Schlüssel, welchen er natürlich auch erhält, direkt mit der jeweiligen verschlüsselten Nachricht, wobei der Einmal-Schlüssel (auch Session-Key genannt) mittels asymmetrischer Verschlüsselung gesichert mitübertragen wurde.

Der in Abbildung 2 dargestellte Ablauf kann wie folgt beschrieben werden:

1. Der Sender schreibt seine Nachricht
2. Die Nachricht wird mittels ZIP Algorithmus komprimiert, um einerseits Platz zu sparen, da eine Komprimierung nach der Verschlüsselung nicht mehr möglich ist, und andererseits um eine Kryptoanalyse zu erschweren.
3. PGP generiert eine 128 bit Zufallszahl welche als Session Key für diese Nachricht verwendet wird.
4. Die Nachricht wird mittels eines symmetrischen Verfahrens (CAST-128, IDEA, ...) mit dem Session Key verschlüsselt.
5. Der Session-Key wird mittels eines asymmetrischen Verfahrens mit dem öffentlichen Schlüssel des Empfängers verschlüsselt. In aktuellen Versionen von PGP wird dazu der ElGamal (eine Variante von Diffie-Hellman) Algorithmus verwendet, ältere Versionen (< 5) verwendeten stattdessen RSA.
6. Der Empfänger der Nachricht benutzt seinen privaten Schlüssel um den Session Key zu entschlüsseln.
7. Der Session Key wird verwendet um die Nachricht zu entschlüsseln.
8. Die Nachricht wird mittels ZIP dekomprimiert.

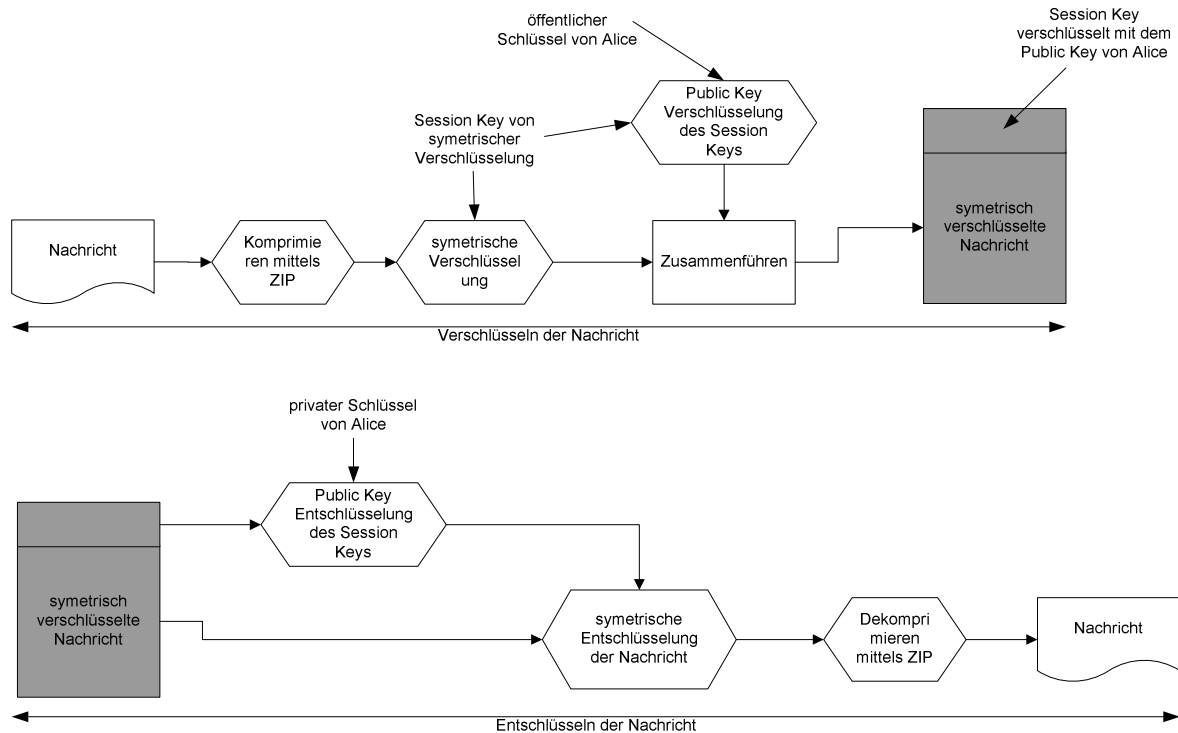


Abbildung 2 - Verschlüsselung mit PGP

5.2 Verschlüsselung und digitale Unterschrift mit PGP

Wie man aus Abbildung 3 ersehen kann man Verschlüsselung und digitale Unterschrift auch gleichzeitig auf eine Nachricht anwenden, wodurch man von der Sicherheit der Verschlüsselung profitiert und durch die digitale Unterschrift auch die Herkunft und Echtheit der Nachricht überprüfen kann.

Der Ablauf kann wie folgt beschrieben werden:

1. Der Sender schreibt seine Nachricht
2. PGP generiert mittels SHA-1 Algorithmus einen eindeutigen Fingerabdruck (Hash-Wert) der Nachricht.
3. Dieser errechnete Fingerabdruck wird mittels eines asymmetrischen Verfahrens mit dem privaten Schlüssel des Senders verschlüsselt und an die Nachricht angehängt
4. Die gesamte Nachricht wird mittels ZIP Algorithmus komprimiert.
5. PGP generiert eine 128 bit Zufallszahl welche als Session Key für diese Nachricht verwendet wird.
6. Die Nachricht wird mittels eines symmetrischen Verfahrens (CAST-128, IDEA, ...) mit dem Session Key verschlüsselt.
7. Der Session Key wird mittels eines asymmetrischen Verfahrens mit dem öffentlichen Schlüssel des Empfängers verschlüsselt. In aktuellen Versionen von PGP wird dazu der ElGamal (eine Variante von Diffie-Hellman) Algorithmus verwendet, ältere Versionen (< 5) verwendeten stattdessen RSA.
8. Der Empfänger der Nachricht benutzt seinen privaten Schlüssel um den Session Key der Nachricht zu entschlüsseln.
9. Der Session Key wird verwendet um die Nachricht zu entschlüsseln.

10. Die Nachricht wird mittels ZIP dekomprimiert.
11. Der verschlüsselte Fingerabdruck wird von der eigentlichen Nachricht getrennt und mit Hilfe des öffentlichen Schlüssels des Senders entschlüsselt.
12. Von der eigentlichen Nachricht wird mittels des in der verschlüsselten Nachricht definierten Hashing Algorithmus (meist SHA1) erneut ein Fingerabdruck errechnet.
13. Der errechnete Hash-Wert wird mit dem entschlüsselten Hash-Wert verglichen.

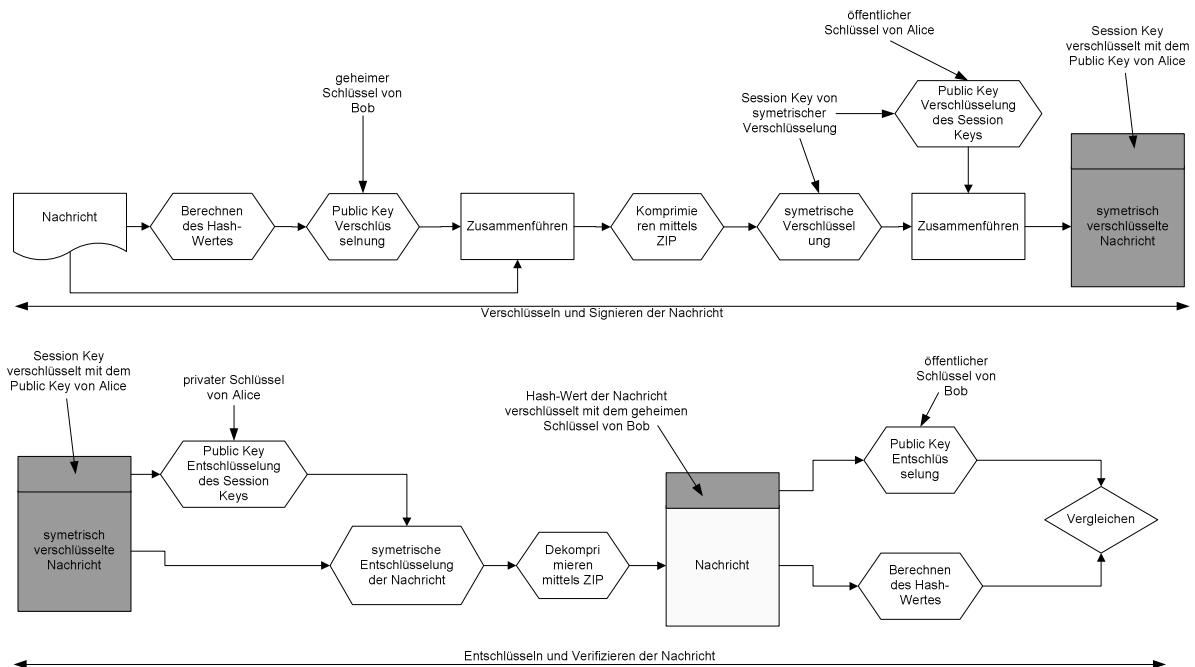


Abbildung 3 - Verschlüsselung und digitale Unterschrift

5.3 Schlüsselverwaltungskonzepte

5.3.1 Allgemein

Eines der größten Probleme bei asymmetrischen Verfahren ist, die Sicherstellung, dass ein öffentlicher Schlüssel auch zu der angegebenen Person stammt. Ansonsten wäre es einem Unberechtigten möglich die Nachricht einzusehen oder zu verändern. Ein kurzes Beispiel soll das Ausmaß verdeutlichen.

Franz sendet an Hans eine Nachricht. Er verwendet dabei einen asymmetrischen Verschlüsselungsmechanismus. Zum Chiffrieren verwendet er den angeblichen öffentlichen Schlüssel von Hans. Dieser wurde jedoch von Anton manipuliert, das heißt, er hat den echten durch seinen eigenen öffentlichen Schlüssel ersetzt. Anton fängt nun diese Nachricht ab, entschlüsselt diese mit seinem privaten Gegenstück. Damit nun Franz und Hans nichts von diesem Betrug merken, verschlüsselt er die Nachricht mit dem echten öffentlichen Schlüssel von Hans und sendet sie an diesem weiter. Somit hat der echte Empfänger und Sender nichts davon bemerkt.

Um eine derartige Manipulation zu verhindern gibt es nun verschiedenste Möglichkeiten.

Für einige dieser Ansätze ist ein „Fingerprint“ notwendig. Wenn man von einem Fingerprint spricht, handelt es sich hier um einen Hashwert von dem öffentlichen Schlüssel, dieser ist zu 99,9% eindeutig zuordenbar zu dem Schlüssel und zum anderen hat er den Vorteil, dass er kürz ist und aus einfachen Zeichenkombination besteht.

PGP und S/MIME setzen beide nun auf asymmetrische Verschlüsselungskonzepte, dementsprechend haben diese auch das Problem mit authentischen öffentlichen Schlüsseln zu lösen. Dabei geht jede Anwendung ihren eigenen Weg.

PGP ist wohl eher der Vertreter für einfache aber trotzdem sichere Lösungen. Dieses Programm findet hauptsächlich den Einsatz im privaten beziehungsweise kleineren Unternehmen. Da der Kostenfaktor hier eine besonders wichtige Rolle spielt, setzt PGP auf das „Web of Trust“ und zusätzliche Sicherungsmechanismen, welche als zusätzliche Absicherung gelten sollen.

S/MIME ist nun eher ein Vertreter aus dem „Big Business“. Dementsprechend sind auch hier die Konzept, die eingesetzt werden etwas kostspieliger. Zertifizierungsstellen, die hierfür notwendig sind, lassen sich ihre Dienste auch dementsprechend bezahlen.

Natürlich sind beide Systeme in den verschiedensten Bereichen im Einsatz und auch sind beide ähnlich erfolgreich, dennoch spielt der Kostenfaktor hier eine große Rolle.

In der folgenden Aufstellung von Schlüsselkonzepten findet man auch gleichzeitig einige der Schwächen solcher Verschlüsselungsmechanismen.

5.3.2 Web-of-Trust

Eine kostengünstige und dennoch sehr sichere Lösung des public-key Problems.

Einfaches Grundkonzept „allen den du vertraust, denen vertraue ich auch“. Daraus ergibt sich auch schon die Schwäche dieses Systems, falls man sich irrt, haben auch andere ein Problem. Nun zu den Details. Alice möchte an Carol eine verschlüsselte Nachricht senden, die beiden kennen sich aber nicht persönlich. Sie haben jedoch eine Gemeinsamkeit, sie kennen beide Bob. Damit nun Alice den öffentlichen Schlüssel von Carol erhält und es auch sicher ist, dass es ihr Schlüssel ist, setzt man das „Web of Trust“ ein. Bob hat ein den öffentlichen Schlüssel von Carol, Er hat sich versichert, dass es auch der richtige ist. Dabei hat er sich mit Carol persönlich getroffen, und Fingerprint (Hashwert) des Schlüssels überprüft, auch telefonisch oder andere Möglichkeiten wären denkbar.

Zum anderen kennt Alice nun den öffentlichen Schlüssel von Bob, auch hier wurde natürlich seine Authentizität überprüft. Damit nun Alice auch sicher den richtigen Schlüssel von Carol hat, sendet Bob diesen an Alice, natürlich zumindest signiert mit seinem privaten Schlüssel. Damit kann nun Alice davon ausgehen, dass zu 99,9% den richtigen Schlüssel von Carol hat, und ihr nun auch eine verschlüsselte Nachricht schicken (Abbildung 4 - Web-of-Trust).

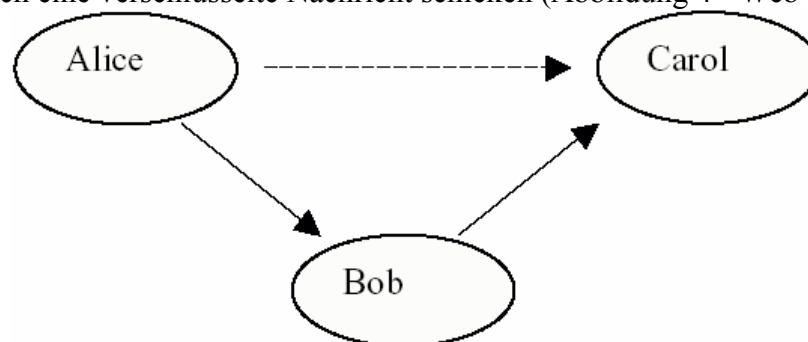


Abbildung 4 - Web-of-Trust

Ein sehr wichtiges Grundkonzept dahinter ist der Keyring. Es handelt sich dabei nun, wie der Name schon sagt, um den persönlichen Schlüsselbund. Zum einem befinden sich darauf alle eigenen Schlüssel und natürlich auch die öffentlichen Schlüssel anderer Personen, die man persönlich eingefügt hat. Abbildung 5 – Public und Privat Keyring zeigt genau diesen Schlüsselbund, zum einen den persönlichen, an dem sich die eigenen Schlüsselpaare (privat und public) befinden und den öffentlichen, an dem sich die öffentlichen Schlüssel anderer Personen befinden. Dabei ist nun eben der Umgang mit den öffentlichen Schlüsseln anderer Personen sehr wichtig.

Es geht nun um die Bewertung der anderen Schlüssel und somit auch darum, wie Vertrauenswürdig man selbst beziehungsweise diese Personen sind. Wenn man einen neuen öffentlichen Schlüssel von einer Person erhält, muss man diesen auf seine Gültigkeit hin überprüfen. Eine Möglichkeit ist, dass man sich kennt und eventuell diesen Schlüssel auf persönlichen Weg austauscht mit Hilfe einer Diskette zum Beispiel. Eine andere Möglichkeit wäre eben die Überprüfung des Fingerprints des Schlüssels über Telefon oder eben auch persönlich. Auf diese Weise kann man eigentlich eine 100% Sicherheit gewährleisten und eben diesen öffentlichen Schlüssel dementsprechend das volle Vertrauen aussprechen und ihm mit dem eigenen Schlüssel signieren. Natürlich hat man nicht immer die Möglichkeit die Identität zu 100% zu überprüfen, dann kann man die eigene Signatur über diesen Schlüssel auch mit eingeschränktem Vertrauen aussprechen. Damit braucht dieser Schlüssel nun zwei weitere Personen die diesem zumindest eingeschränkt Vertrauen um als 100% authentisch zu gelten.

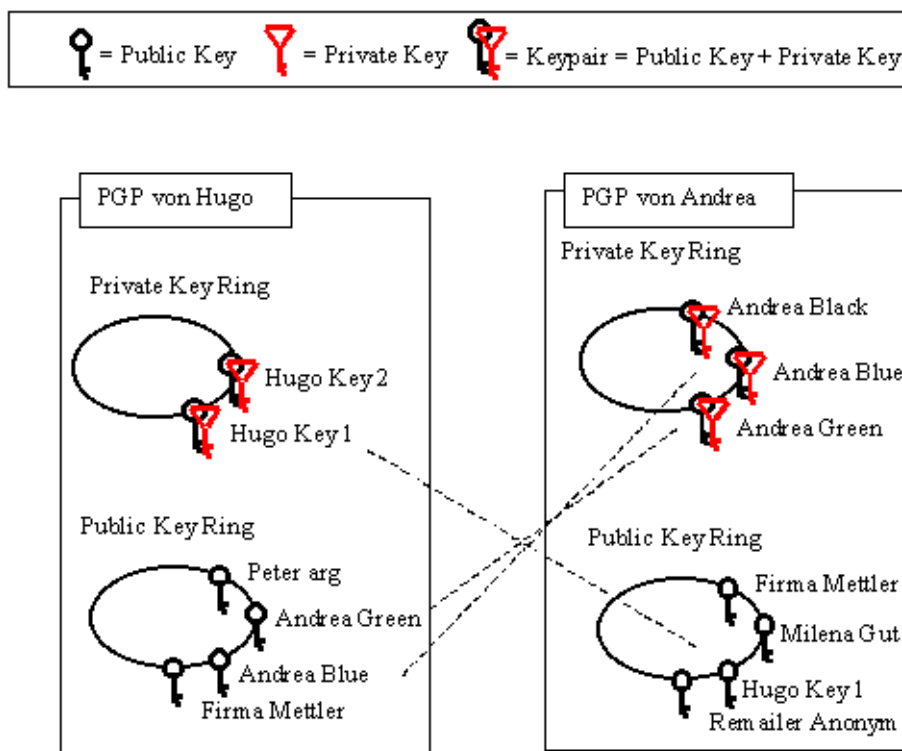


Abbildung 5 – Public und Privat Keyring

Um nun diese Konzept auch für eine breite Masse umsetzbar zu machen, gibt es freie Key-Server, die eben diese Signaturen von öffentlichen Schlüsseln auch speichern. Damit kann gewährleistet werden, dass diese Schlüssel und dessen Signierungen auch verbreitet werden. Die Daten, die diese Key-Server verwalten, erhalten diese nun von den einzelnen Usern mit eben ihrem Grad an Vertrauen oder eben auch nicht. Nun kann man bestimmen ob diese Schlüssel an andere weitergegeben werden soll. Dabei gibt es „exportable“ falls man diesen auf dem Server speichern soll und „non-exportable“ um dies zu verhindern. Natürlich kann man die „Public Key Rings“ auch direkt miteinander austauschen (siehe Abbildung 5) damit kann auch das oben angeführte Beispiel zu Abbildung 4 - Web-of-Trust direkt umgesetzt werden.

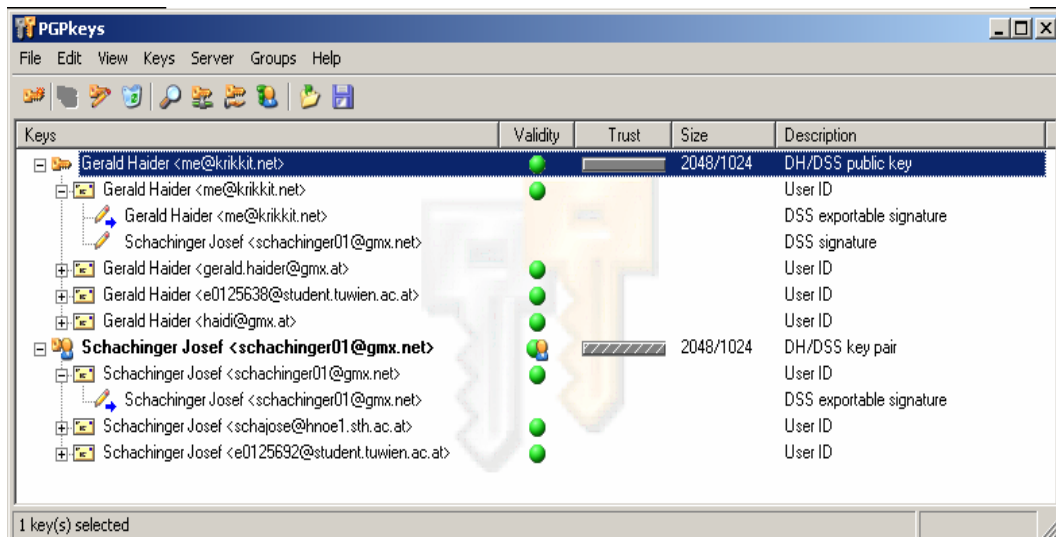


Abbildung 6 - Schlüsselverwaltung unter PGP

In Abbildung 6 - Schlüsselverwaltung unter PGP zeigt als Beispiel die Schlüsselverwaltung unter PGP 8.0, es wird angezeigt, welche Schlüssel man hat, von wem sie signiert sind, ob diese auch gültig sind, den Grad an Vertrauen, den man einen anderen Schlüssel entgegen bringt und natürlich die Schlüsselgröße.

Die einzelnen Sicherheitsstufen:

Untrusted

Marginal

Complete

Für den eignen Schlüssel wird der schraffierte Balken verwendet

5.3.3 Zertifikate

Bei diesem System wird davon ausgegangen, dass nur bestimmte Stellen die Möglichkeit haben die Identität zu überprüfen. Diese Stellen werden als Zertifizierungsstelle oder CA (Certification Authorities) bezeichnet. Ein Public Key ist ohne ein solches Zertifikat völlig wertlos. Es existieren dabei verschiedene Instanzen von Zertifizierungsstellen, wie in Abbildung 7 - Certificate Chain aufgezeichnet. Der große Vorteil bei diesem Modell lässt sich am einfachsten anhand des Beispiels mit Bob und Alice erklären. Beim Einsatz von Zertifikaten ist es nicht notwendig, dass sie sich kennen, dementsprechend erleichtert das den Austausch der Schlüssel beziehungsweise in dem Fall der Download von einer Zertifikaten Datenbank.

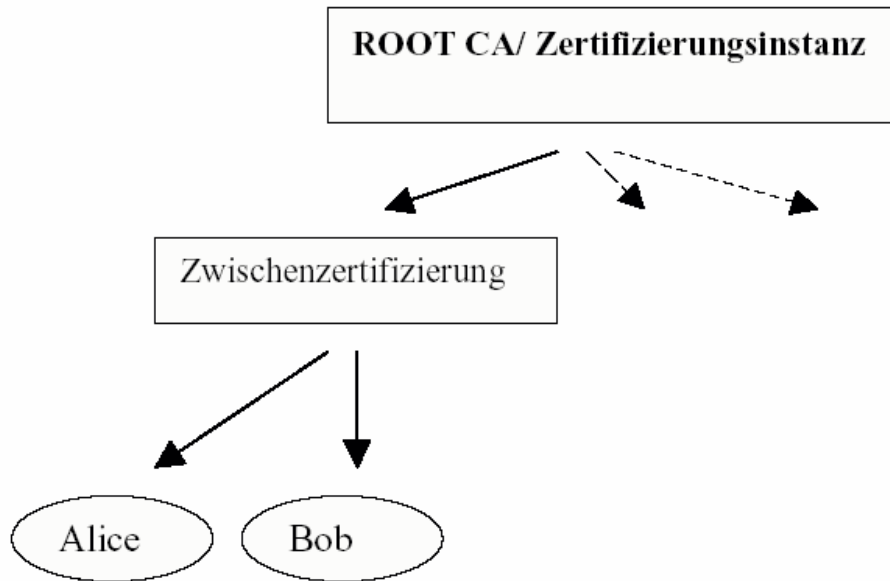


Abbildung 7 - Certificate Chain

Um nun ein Zertifikat zu erhalten sind verschiedene Wege möglich. Eine davon ist, dass der Anbieter des Emailaccount auch gleichzeitig die Möglichkeit zur Zertifizierung über eine geeignete Stelle anbietet. Als Beispiel kann hier Web.de angeführt werden. Dabei ist jedoch zu bedenken, dass auch der private Key an andere Stelle bekannt ist, wobei das hier nicht die einzige Variante ist, bei dem dies der Fall ist. Bei Anbietern wie eben Web.de findet der private User eine einfache und effektive Möglichkeit seine Emails zu verschlüsseln, da die Software nicht extra auf den Rechner installiert werden muss, da S/MIME bei Mailclients wie Outlook oder Eudora direkt integriert ist.

Eine andere Möglichkeit an ein S/MIME gültiges Zertifikat zu gelangen, ist eine freie Zertifizierungsstelle. An dieser Stelle seien einige angeführt:

- VeriSign
- thawte
- GlobalSign
- TeleSec (Deutsche Telekom)
- Trustcenter.de

Wobei VeriSign als der bekannteste Vertreter solcher Zertifizierungsstellen gesehen werden kann. Eine CA kann nun Zertifikate mit verschiedensten Sicherheitsstufen ausgeben, natürlich ist dies mit dementsprechenden Kosten verbunden. Um die Identität festzustellen wird dementsprechend unterschiedlicher Aufwand betrieben um die Übereinstimmung der Identität zu gewährleisten. Diese Sicherheitsstufen werden in Klassen von eins bis drei eingeteilt.

Klasse 1:

Dies Klasse garantiert nur eine sehr niedrige Sicherheitsstufe. Dabei wird hier die Authentizität mithilfe der Email Adresse festgestellt. Die Überprüfung ist wenig Aufwand, da diese einfach automatisiert werden kann. Es muss aber in betracht gezogen werden, das ein Email Account sehr einfach gefälscht werden kann. Diese Art der Zertifikate wird meist gratis angeboten und wird eher als Lockmittel verwendet. es kann aber als kostengünstiger Einstieg in die etwas sicherere Digitale Welt angenommen werden.

Klasse 2:

Dies ist eine Klasse, die schon ein sehr hohes Maß an Sicherheit gewährleisten kann. Die Identifizierung erfolgt mithilfe öffentlicher Dokumente bei einer Außenstelle des Unternehmens, das die Zertifikate ausstellt. Dies kann zum Beispiel ein Postamt oder ähnliches sein. Diese übermitteln die Daten an die Zertifizierungsstelle weiter. Die Antragstellende Person muss dabei einen Personalausweis, Führerschein oder ähnliches vorweisen.

Dieses Zertifikat verursacht in etwa Kosten in der Höhe von 15€.

Die Möglichkeiten der Fälschung liegen hier in einem sehr geringen Ausmaß, dementsprechend kann einem solchen Zertifikat ein hohes Maß an Vertrauen entgegengebracht werden.

Klasse 3:

In dieser Klasse ist es notwendig, persönlich bei der Zertifizierungsstelle zu erscheinen, dabei ist wieder das Vorlagen persönlicher Dokumente (Reisepass, Führerschein) notwendig. Diese Art der Identifizierung kann das größte Vertrauen ausgesprochen werden. Es ist aber auch etwas aufwendig und kostenintensiv. Da solche Zertifizierungsstellen meist nur in sehr großen Städten einen Standort betreiben sind lange Anreisewege in Kauf zu nehmen. Dies natürlich zusätzlich zu den Kosten für die Ausstellung.

Ein solches Zertifikat enthält nun verschiedene Daten:

- Seriennummer des Zertifikates. Jede CA hat ihre eigenen Seriennummern.
- Der öffentliche Schlüssel des Zertifikats-Inhabers.
- Der Name des Zertifikats-Inhabers.
- Der Name der CA.
- Die Gültigkeitsperiode.
- Die digitale Signatur der CA

Von diesen Daten wird ein Fingerprint mithilfe einer Hashfunktion erzeugt und mit dem privaten Schlüssel des Inhabers verschlüsselt.

Einerseits kann man dieses nun selbst veröffentlichen, andererseits betreiben die Zertifizierungsstellen auch eigene Datenbanken mit den Zertifikaten. Diese Datenbanken sind auch die einzige Möglichkeit um mit der Sicherheit der jeweiligen Klasse den richtigen öffentlichen Schlüssel eines Empfängers zu haben. Dementsprechend ist es wichtig diese laufend mit zu vergleichen. Auch für den Fall das ein Zertifikat nicht mehr gültig ist. Dies kann verschiedene Gründe haben. Der einfachste Fall und auch ungefährlichste ist der Ablauf der Gültigkeitsperiode. Dies hat seinen Sinn in zwei Bereichen. Zum einem möchten die CA laufend damit Geld verdienen und zum anderen wird die Dauer der Gültigkeit immer so gewählt, das in es als unwahrscheinlich gilt, das ein Schlüssel gebrochen wird.

Weitere Möglichkeit für die Entziehung der Gültigkeit eines Zertifikates ist, dass die Geheimhaltung des privaten Schlüssels nicht mehr sicher ist, beziehungsweise das dieser bereits gestohlen wurde. Auch der Verlust ist eine Möglichkeit, dass ein Schlüssel für ungültig erklärt werden muss. Eine weitere Möglichkeit ist es, dass sich persönliche Daten des Inhabers eines Zertifikates sich geändert haben.

5.3.4 Telefonische bzw. persönliche Überprüfung

Dabei wird eben dieser oben beschriebene Fingerprint des Public Keys direkt mit dem Empfänger ausgetauscht. Es bleibt auch hier wieder ein gewisses Restrisiko, besonders am Telefon kann nur bedingt sichergestellt werden, ob es sich um die richtige Person handelt. Bei einem persönlichen Treffen ist die Überprüfung der Personalien schon etwas einfacher.

5.3.5 Bekanntgabe über geeignete Medien

Hierbei handelt es sich wieder nur um einen bedingt praktikablen Ansatz für die breite Masse. Man versucht den Fingerprint des Schlüsselinhabers auf ein fälschungssicheres Medium zu verbreiten. Als Beispiel kann hier die Zeitschrift c't angeführt werden. Sie veröffentlichen den Fingerprint im Impressum der Zeitschrift. Natürlich hat nicht jeder ein derartiges Medium zur Verfügung. Als Alternative kann man aber den Fingerprint auf seine Visitenkarte veröffentlichen, wobei auch hier wider nicht gewährleistet werden kann, dass es sich beim Aussteller der Visitenkarte auch um den tatsächlichen Besitzer des Keys handelt.

5.4 S/MIME

S/MIME (Secure/Multipurpose Internet Mail Extension) ist eine um sicherheitsrelevante Funktionalität erweiterter Abkömmling des MIME Internet E-Mail Standards, aufbauend auf der Sicherheitstechnologie von RSA Data Security.

Um S/MIME zu verstehen, ist aber ein allgemeines Verständnis der verwendeten E-Mail Formate notwendig, im speziellen von MIME. Dieses baut aber wiederum auf dem traditionellen E-Mail Format Standard, RFC 822 auf, und darum soll im Folgenden auch dieser erläutert werden.

5.4.1 RFC 822

RFC 822 definiert das Format von Textnachrichten die über E-Mail versandt werden sollen. Aus der Sicht dieses Standards kann man eine herkömmliche Mail in 2 Teile aufteilen, nämlich die Hülle/Umschlag (der Header) und den eigentlichen Inhalt. Wobei der Umschlag eine ähnliche Funktion ausübt wie ein echter Briefumschlag, denn er enthält alle Information die nötig sind um die E-Mail zu transportieren bzw. zuzustellen. In der RFC 822 definiert ist eigentlich nur das Format des Inhaltes (Contents), wobei darin durchaus auch Felder definiert sind die vom Mail System genutzt werden um gewisse Informationen zu erlangen.

Der generelle Aufbau eine E-Mail die dem RFC 822 entspricht ist im Prinzip ziemlich einfach, da die Nachricht nur aus einigen Header Zeilen, gefolgt von einem uneingeschränkten Textinhalt (the body) besteht. Wobei die Kopfzeilen von Textinhalt selbst nur durch eine simple Leerzeile getrennt werden.

Ein Beispiel für einen RFC 822 konformes E-Mail:

```
Date: Sun, 22 Jun 2003 14:40:22 +0200
To: Gerald Haider <e0125638@student.tuwien.ac.at>
From: Bob <bob@somehost.com>
Subject: RFC 822 Header Format
```

```
Hello world! Das ist der eigentliche Textinhalt, der sogenannte "body".
```

5.4.2 MIME – Multipurpose Internet Mail Extension

Die MIME Erweiterung wurde geschaffen um einige Probleme und Limitierungen des E-Mail Verkehrs zu umgehen, die sich aus dem Zusammenspiel von SMTP (Simple Mail Transfer Protocol) und RFC 822 ergaben.

Unter anderem waren die folgenden Limitierungen gegeben:

1. SMTP kann kein ausführbaren Dateien oder Binärdateien übertragen
2. SMTP kann keine Texte übertragen die Sonderzeichen enthalten
3. SMTP Server verweigern teilweise Mail die eine gewisse Größe überschreiten
4. Manche nicht Standardkonforme SMTP Implementationen verursachen des weiteren noch folgende Probleme:
 - löschen bzw. hinzufügen von Zeilenschaltungen
 - abschneiden bzw. umbrechen von Zeilen die länger als 76 Zeichen sind
 - löschen von Leerzeichen/Tabulatoren an am Anfang der Zeile
 - umwandeln von Tabulatoren in mehrere Leerzeichen

Aus diesen eben genannten Gründen wurde MIME geschaffen und in den RFCs 2045 – 2049 niedergeschrieben.

5.4.2.1 MIME – Übersicht

Die MIME Spezifikation beinhaltet die folgenden Elemente:

1. fünf neue Header-Felder (header fields), welche Information über den Inhalt einer Nachricht enthalten, diese Felder können in den RFC 822 Header integriert werden.
2. es wurden weiters mehrere verschiedene Arten von Inhaltstypen (Content Types) definiert, welche die Darstellung von Multimedia Daten unterstützen sollen.
3. Übertragungs- Codierungen wurden definiert, welche sicherstellen sollen das jede Art von Inhalten in ein Format gebracht werden kann, welches von jedem Mail System akzeptiert und vor allem nicht verändert wird.

5.4.2.2 MIME – Header Felder

- **Mime Version:** muss den Wert 1.0 haben, und spezifiziert somit das die Nachricht den RFCs 2045 und 2046 entspricht
- **Content-Type:** beschreibt die Daten im die im Body der Nachricht enthalten sind
- **Content-Transfer-Encoding:** beschreibt die Codierung die benutzt wurde um den Body der Nachricht in ein E-Mail taugliches Format zu bringen.
- **Content-ID:** eine eindeutige ID
- **Content-Description:** eine textuelle Beschreibung der Daten im Body

5.4.2.3 MIME Content Types

Auf die einzelnen Content Typen möchte soll in dieser Arbeit nicht genauer eingegangen werden, es soll vielmehr nur kurz angemerkt werden das man im Prinzip sieben Haupt-Typen unterscheidet, welche sich in 15 Unter-Typen aufspalten.

Als Beispiel wären z.B. folgende Typen anzuführen:

- application/octet-stream – Binärdaten, z.B. ausführbare Dateien
- text/plain – unformatierter Text in der jeweiligen angegebenen Codierung

5.4.2.4 MIME Transfer Encodings

Im Header Feld „Content-Transfer-Encoding“ können insgesamt sechs verschiedene Werte angegeben sein, wobei drei dieser Werte (7bit, 8bit und binary) eigentlich nur angeben das der Body nicht codiert wurde.

Die beiden Werte die wirklich eine Codierung angeben sind:

- **quoted-printable:** geht davon aus das der größte Teil des Inhaltes ASCII ist, und codiert daher nur die nicht ASCII Zeichen, dadurch bleibt der Inhalt trotz Codierung größtenteils für den Menschen lesbar.
- **base64:** codiert die Daten indem jeweils 6-bit Eingangsdaten in 8-bit ASCII Ausgangsdaten codiert werden

Eine Sonderform stellt noch der Wert „x-token“ dar, welcher angibt das ein nicht standardisiertes Codierungsverfahren verwendet wurde, wobei der Name des Codierungsverfahrens angegeben werden muss.

5.4.3 S/MIME Funktionalität

S/MIME bietet folgende Funktionen an:

- Verschlüsselung (Enveloped Data)
- base64 codierte Daten mit digitale Unterschriften (Signed Data)
- Klartext mit digitaler Unterschrift (Clear Signing)
- verschlüsselte Daten mit digitaler Unterschrift

Bei Verwendung all dieser Funktionen werden die Daten in einem jeweiligen S/MIME Content Type konvertiert, die sich problemlos in die bereits in Kapitel 5.4.2.3 genannten MIME Content Types einpassen.

5.4.4 S/MIME – Content Types

5.4.4.1 Verschlüsselung (Enveloped Data)

Bei der Verschlüsselung mittels S/MIME wird ein Content Type mit der genauen Bezeichnung:

```
application/pkcs7-mime; smime-type=enveloped-data;
```

verwendet.

Die Schritte zum erzeugen eines solchen Datenblockes sind wie folgt:

1. Erzeugen eine pseudo-zufälligen Session Keys für den verwendeten symmetrischen Verschlüsselungsalgorithmus (RC2/40 oder tripleDES)
2. Für jeden Empfänger wird der Session Key mit dem jeweiligen Public Key des Empfängers verschlüsselt.
3. Für jeden Empfänger wird ein Block namens „RecipientInfo“ generiert, der folgende Dinge enthält: das Public Key Zertifikat des Senders, die Bezeichnung des Algorithmus der zum Verschlüsseln verwendet wurde und der verschlüsselte Session Key.
4. Die Nachricht wird mit dem Session Key verschlüsselt.

5.4.4.2 digitale Unterschrift (SignedData)

Die Schritte zum Erzeugen einer digitalen Unterschrift:

1. Auswählen eines Hash Algorithmus (erfolgt in der Regel automatisch)
2. Berechnen des Hash Wertes des zu unterzeichnenden Textes
3. Verschlüsseln des Hash Wertes mit dem privaten Schlüssel des Unterzeichners
4. Erzeugen eines Blockes namens „SignerInfo“ der die folgenden Daten enthält:
 - das Public Key Zertifikat des Unterzeichners
 - die Bezeichnung des verwendeten Hash Algorithmus
 - die Bezeichnung des verwendeten Verschlüsselungsverfahrens
 - der verschlüsselte Hash Wert

Eine Variation des obigen Prozesses ist das „Clear Signing“ wobei der Unterschied zu obigen Verfahren nur darin besteht das nicht die ganze Nachricht in base64 codiert wird sondern nur die Signatur selbst, sodass der eigentliche Text für den Menschen lesbar bleibt.

6. Sicherheitsrisiken

Grundsätzlich ist der theoretische Ansatz hinter PGP und S/MIME so gut wie undurchdringbar, natürlich könnte man aus einer verschlüsselten Nachricht das Original herausrechnen, aber um den Aufwand zu verdeutlichen, ein Zitat von William Crowell, Stellvertretender Direktor des Nationalen Sicherheitsdienstes der USA (NSA), 20. März 1997.

„Wenn alle PCs weltweit – d. h. 260 Millionen Computer – an einer einzigen von PGP verschlüsselten Nachricht arbeiten würden, würde es im Schnitt immer noch ungefähr 12 Millionen mal das Alter des Universums dauern, bis eine einzige Nachricht decodiert werden könnte.“

Auch diese Behauptung ist wieder relativ zu sehen, den bei der derzeitigen Steigerung der Rechenleistungen kann vielleicht in ein einfacher PC in einigen Jahren die selbe Rechenleistung, wie der Earth-Simulator heute hat, haben.

Das heißt, der wichtigste Ansatz in Bezug auf Sicherheit ist es, immer am aktuellen Stand zu sein, den es werden auch immer bessere Algorithmen verwendet werden und es werden auch immer längere Schlüssel verwendet werden können.

Die direkte Bedrohung liegt vielmehr im Umgang mit den Verschlüsselungswerkzeugen, den unachtsamen Umgang mit den Schlüsseln und viele weitere Sorglosigkeiten der einzelnen Anwender. Auf den folgenden Seiten findet man eine Aufstellung der Bedrohungen und auch Lösungsansätze beziehungsweise Verhaltensvorschriften.

6.1 Kompromittierte Passphrasen oder private Schlüssel

Die einfachste Form des Missbrauches ist, wenn unautorisierte Personen Zugriff auf die Passphrase haben und den dazugehörigen privaten Schlüssel. Wenn dieser Vorgang vom Eigentümer unentdeckt bleibt ist dem Missbrauch Tür und Tor geöffnet.

Ein privater Schlüssel ist ohne Passphrase unbrauchbar, somit ist der Zusammenstellung dieser besonders Augenmerk zu schenken.

Namen aus der Familie oder andere einfache Wörter sind völlig ungeeignet, auch einfache Zitate sollen hier nicht zur Anwendung kommen. Namen sind am Einfachsten zu testen, auch einfache Wörter lassen sich mit geeigneter Software aufdecken. Eine Passphrase soll einfach zu merken sein aber eben nicht leicht zu entdecken. Eine Variante kann es sein, ein bekanntes Gedicht mit eigenen Worten umzuschreiben, Kreativität macht sich hier besonders bezahlt! Auf keinen Fall darf die Passphrase aufgeschrieben werden.

6.2 Veränderter öffentlicher Schlüssel

Lösungsansätze für die Sicherheit findet man an anderer Stelle beschrieben, wie zum Beispiel „Web of Trust“ oder „CA“. Besonders als Neuling auf diesem Gebiet sollte man hier Vorsicht walten lassen.

Der öffentliche Schlüssel eines Empfängers sollte immer mit höchster Genauigkeit überprüft werden, falls Zweifel bestehen, auf Alternativen zurückgreifen.

6.3 Rückstände von gelöschten Dateien

Die meisten Betriebssysteme löschen die Daten auf einer Festplatte nicht physikalisch, sondern markieren sie nur als gelöscht. Mit geeigneter Software können diese Daten wieder sichtbar gemacht werden, unter der Voraussetzung, dass sie nicht von anderen Informationen überschrieben worden sind.

Befinden sich nun kritische Daten oder Passphrases in diesem „gelöschten“ Bereich, können diese von unautorisierten Personen wiederhergestellt werden.

Eine Möglichkeit um das zu verhindern ist das sofortige Überschreiben gelöschter Bereiche. PGP bietet hierfür ein eigenes Tool an.

6.4 Viren und Trojanische Pferde

Viren können, falls sie unentdeckt bleiben, die Passphrase und den privaten Schlüssel ausspionieren und diese an unautorisierte Personen weitersenden.

Programme wie PGP und S/MIME davon aus, dass sie auf einem sicheren und Virenfreien System laufen, dementsprechend sollte man auch mit geeigneten Virenschutzprogrammen auch auf diese Möglichkeit achten.

Ein weiteres Bedrohungspotential ergibt sich aus einem anderen Bereich. Um die Zuverlässigkeit von Verschlüsselungssoftware zu gewährleisten, ist es zum Teil üblich den Code zu veröffentlichen. Dadurch ergibt sich aber die Gefahr, dass Programme in Umlauf gebracht werden, die sich als Original ausgeben, aber eben einen veränderten Code im Hintergrund laufen haben. Die Software kann dann gewisse Abläufe nicht mehr korrekt durchführen beziehungsweise geheime Daten unerlaubterweise weitergeben.

6.5 Auslagerungsdateien

Heutzutage verfügen die meisten Betriebssysteme über den Mechanismus der Auslagerungsdateien oder auch virtueller Arbeitsspeicher genannt, dieser dient zur Erweiterung des vorhandenen physischen Arbeitsspeichers. Es kann nun die Möglichkeit eintreten, dass eben genau in dieser Auslagerungsdatei kritische Daten (privater Schlüssel, Passphrase, usw.) befinden. Diese Auslagerungsdatei wird aber erst dann wieder überschrieben, falls das Betriebssystem

diesen Speicher für andere Anwendungen benötigt, dementsprechend ist es möglich, dass sich Rückstände im virtuellen Arbeitsspeicher befinden.

6.6 Tempest-Angriffe

Mit geeigneten technischen Mitteln lässt sich das Bild des Monitors durch dessen Abstrahlung feststellen. Auch die Anschläge der Tastatur lassen sich dadurch feststellen. Das Problem liegt in der elektromagnetischen Strahlung die jedes dieser Geräte abgibt. Diese Strahlung kann durch Abschirmung reduziert werden.

Hierbei handelt es sich aber um eine nicht ganz günstige Art der Spionage.

6.7 Gefälschte Zeitmarkierungen

Bei der Signieren eines Dokumentes geht es nicht nur darum, dass man feststellt ob das Dokument wirklich von der Person stammt, sondern auch darum, wann dieses Dokument signiert wurde. Dieser Timestamp wird bei der Signatur eines Dokumentes hinzugefügt und kann später ausgelesen werden. Nun kann man diese Zeitmarkierung sehr einfach verändern, man braucht nur die Systemzeit des Rechners an dem die Signatur erstellt wurde, ändern.

6.8 Mehrbenutzer Systeme

Falls ein Rechner von verschiedenen Personen benutzt wird, können unautorisierte Personen Einsicht auf Vertrauliche Daten haben.

Besonders bei Windows, das kein echtes Multiuser System ist, ist das ein nicht zu unterschätzendes Problem. Falls mehrere Personen sich einen Rechner teilen, sollte besonderes Augenmerk auf die sichere Verwahrung sicherheitskritischer Daten gelegt werden.

6.9 Datenverkehrsanalyse

Auch wenn Außenstehende Personen die Daten nicht entschlüsseln können, können ihnen die Größe und der Zeitpunkt Rückschlüsse auf den Inhalt oder Zweck geben. Dieses Gefahrenpotential ist auch durch Programme wie PGP und S/MIME nicht abgedeckt, auch hier müssen anderweitig Lösungen gefunden werden.

6.10 Kryptoanalyse

Einfach gesagt, entschlüsseln der Daten. Wie im obigen Zitat beschrieben, können Daten auch entschlüsselt werden. Dabei wären aber derzeit unvorstellbare Rechnerleistungen nötig. Auch sind derzeit keine andere Schwächer der eingesetzten Algorithmen bekannt, wobei diese von den bekanntesten Kryptologen entwickelt und überprüft worden sind.

Natürlich bleibt auch hier ein gewisses Restrisiko, zum Beispiel könnten Geheimdienst oder andere Stellen, Schwächen in den Verschlüsselungstechniken entdecken und diese nicht weitergeben. Diese Möglichkeit kann aber als eher unwahrscheinlich gesehen werden.

7. Zusammenfassung

Im Gegensatz zu SSL haben sich Techniken zur Verschlüsselung von Email leider noch nicht im selben Maße durchgesetzt. Für eine Bank wäre es heute undenkbar Daten über eine unverschlüsselte Leitung zu schicken.

Doch steht auch in Zukunft dem geschützten Austausch von elektronischen Briefen und Attachments nichts mehr im Wege, Softwarepakete wie PGP oder S/MIME verfügen über eine hohe Anwenderfreundlichkeit und sind den auf dem höchsten Stand der Technik.

PGP konnte sich vor allem bei der breiten Masse von privaten Usern sehr schnell verbreiten, zum einen, weil die Identifizierung der privaten Schlüssel gratis ist, durch den Ansatz des „Web of Trust“. Zum anderen ist der Source-Code von PGP öffentlich einsehbar, dass es jedem der Lust hatte ermöglicht, die Sicherheit von PGP zu überprüfen, sofern er das notwendige technische Know-How dazu besitzt.

Bei der Ausarbeitung dieses Themas sind wir auf verschiedenste Problemstellungen und Lösungsvorschläge getroffen. Eines lässt sich abschließend mit Sicherheit sagen: Keines der Verfahren kann eine 100%ige Sicherheit garantieren. Auch kann es sein, dass genau durch den Einsatz von Verschlüsselung erst ein möglicher Angreifer auf eine bestimmte Nachricht aufmerksam wird. Aber es ist wohl ähnlich dem Sicherheitsgurt beim Autofahren. Auch dieser gibt keine 100%ige Sicherheit, jedoch kann er einen vor den meisten Gefahren bewahren. Und im Nachhinein hätten ihn die meisten verwendet, wenn man wüsste, dass etwas passiert.

8. Bibliographie

- [1] „Einführung in die Kryptographie“, Network Associates Inc. 1998.
- [2] Claudia Eckert: „IT-Sicherheit: Konzepte – Verfahren – Protokolle“, Oldenbourg 2001. Kap. 11.4.1
- [3] Dirk Hörig: S/MIME – “Secure Email” Technische Universität München, Jänner 2003
- [4] Patric Majcherek: “Sicherer E-Mail Verkehr - Vergleich von PGP und S/MIME”, Universität Essen
- [5] Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone: „*Handbook of Applied Cryptography*“, 1996.
- [6] Jens Rosenboom: “Geschichte und Anwendung des Software-Pakets Pretty Good Privacy”, 1999,
<http://math-www.uni-paderborn.de/~aggathen/S99sem/ausarbeitungen/rosenboom/index.htm>
- [7] Bruce Schneier: “*Applied Cryptography, Second Edition*“, Wiley Computer Publishing 1996.
- [8] William Stallings: “*Cryptography and Network Security: Principles and Practice*“, Prentice Hall, 1999. Kap. 12

9. Abbildungsverzeichnis

| | |
|--|----|
| Abbildung 1 - Funktionsweise der digitalen Unterschrift..... | 7 |
| Abbildung 2 - Verschlüsselung mit PGP | 10 |
| Abbildung 3 - Verschlüsselung und digitale Unterschrift..... | 11 |
| Abbildung 4 - Web-of-Trust | 12 |
| Abbildung 5 – Public und Privat Keyring..... | 13 |
| Abbildung 6 - Schlüsselverwaltung unter PGP | 14 |
| Abbildung 7 - Certificate Chain | 15 |