Verschlüsseln und Signieren von E-Mails – eine einfache Anwendung?

Das Prinzip der elektronischen Signatur und der Verschlüsselung mit Hilfe asymmetrischer kryptographischer Verfahren ist schon lange bekannt. Doch warum erhält man in der Praxis kaum signierte oder verschlüsselte E-Mails? Dieser Artikel gibt einen Überblick über die Vielschichtigkeit dieser Thematik und den Stand der Technik sowie der Entwicklung an der Humboldt-Universität zu Berlin. Dazu werden die unterschiedlichen Vertrauensmodelle und die Verwendung von Smartcards betrachtet.

Einleitung

Innerhalb des Projektes *UVsec* [1] wird als eine erste Anwendung das Verschlüsseln und elektronische Signieren von E-Mails angestrebt. Diese Anwendung mag anfangs nicht sehr anspruchsvoll klingen, gewinnt allerdings im Zusammenhang betrachtet erheblich an Komplexität. Die Vertrauenswürdigkeit einer elektronischen Signatur basierend auf asymmetrischen kryptographischen Algorithmen hängt von vielen Faktoren ab, wie dem Ort der Speicherung des geheimen Signaturschlüssels und dem Vertrauen auf die Zugehörigkeit eines öffentlichen Signaturschlüssels zu einer Person. Hierzu wurden verschiedene Modelle und Vertrauensbeziehungen entwickelt, wobei grundsätzlich der hierarchische und der nichthierarchische Ansatz zu unterscheiden sind. Diese Ansätze werden einmal vom X.509 Standard und seinen Profilen in Form von *Zertifikaten* sowie von der PGP-Gemeinde in Form von *unterschriebenen öffentlichen Schlüsseln* umgesetzt. Nun ist ein Zertifikat auch ein unterschriebener öffentlicher Schlüssel, unterscheidet sich aber wesentlich von einem öffentlichen PGP-Schlüssel. Die Unterschiede und Gemeinsamkeiten der verschiedenen Modelle können an dieser Stelle nicht detailliert erläutert werden, aber es soll dem Leser ein Gefühl dafür vermittelt werden, dass ein Zertifikat für eine bestimmte Sicherheitspolitik und damit Vertrauenswürdigkeit steht und ein bestimmtes Format besitzt. Dass es viele dieser Formate gibt, darf man an dieser Stelle schon vermuten. Weiterhin unterstützt nicht jeder Mailclient alle Zertifikatsformate. Als weitere Herausforderung ist die Einbindung von Smartcards in die Mailclients zu sehen, die als Speicher geheimer Schlüssel fungieren.

Die Asymmetrische Kryptographie

Es soll anfangs eine kurze Einführung [2,3] in asymmetrische kryptographische Algorithmen gegeben werden, da sie die Grundlage moderner elektronischer Signaturen, Verschlüsselungs- und Authentifizierungstechniken sind. Diese Algorithmen besitzen ein *Schlüsselpaar*, wovon ein Schlüssel als *geheimer* oder privater und der andere als *öffentlicher Schlüssel* bezeichnet wird. Ein Verschlüsselungsverfahren könnte nun wie folgt aussehen: Ein Nutzer generiert sich ein solches Schlüsselpaar, behält den geheimen Schlüssel bei sich und übergibt den öffentlichen Schlüssel seinen Kommunikationspartnern. Wenn einer von den Partnern dem Nutzer eine verschlüsselte Nachricht schicken will, so verschlüsselt er mit dem öffentlichen Schlüssel des Nutzers die Nachricht und sendet sie ihm. Der Nutzer kann die Nachricht nun mit seinem dazugehörigen geheimen Schlüssel wieder entschlüsseln. Für den interessierten Leser sei an dieser Stelle noch angemerkt, dass in der Praxis aufgrund des hohen Berechnungsaufwandes die Verschlüsselung der Nachricht mit Hilfe eines symmetrischen Verfahrens erfolgt und lediglich der zugehörige Schlüssel mit dem öffentlichen Schlüssel verschlüsselt wird.

Diese auch *Public-Key* genannten Verfahren lassen sich in umgekehrter Weise für die Realisierung einer elektronischen Signatur verwenden. Ein Nutzer generiert sich wieder ein solches Schlüsselpaar, behält den geheimen Schlüssel bei sich und übergibt den öffentlichen Schlüssel seinen Kommunikationspartnern. Um nun ein beliebiges elektronisches Dokument zu unterzeichen, bildet der Nutzer zunächst ein eindeutiges Komprimat des Dokumentes. Dieses Komprimat ist der mathematische *Hashwert* konstanter Länge der elektronischen Daten. Wird nur ein Bit der Daten verändert, so ändert sich dieser Hashwert. Aus diesem Grund wird er häufig auch als *Fingerabdruck* des Dokumentes bezeichnet. Diesen Hashwert verschlüsselt der Nutzer mit seinem geheimen Schlüssel und hat damit eine elektronische Signatur generiert. Das Dokument und die Signatur sendet er an seinen Kommunikationspartner, der ebenfalls den Hashwert des Dokumentes bildet und den gesendeten verschlüsselten Hashwert (die Signatur) mit dem zugehörigen öffentlichen Schlüssel entschlüsselt. Sind beide Hashwerte gleich, so kann ausschließlich der Nutzer die Signatur generiert haben.

Die dritte wichtige Anwendung der Public-Key Verfahren ist die Authentifizierung eines Nutzers. Um sich davon zu überzeugen, dass ein Kommunikationsteilnehmer wirklich der ist, der er zu sein vorgibt, schicke man ihm eine Zufallszahl mit der Bitte diese zu signieren. Ist die Verifikation der elektronischen Signatur erfolgreich, so ist der Nutzer authentifiziert.

Die drei eben beschriebenen Anwendungen sind theoretisch plausibel und korrekt. Doch will man die Szenarien in die Praxis umsetzen, so stößt man auf zwei Probleme, eines für jeden Schlüssel. So funktioniert das Prinzip der elektronischen Signatur nur, wenn man in

der Tat davon ausgehen kann, dass der geheime Schlüssel ausschließlich im Besitz des Inhabers ist. Weiterhin kann ein Empfänger eine elektronische Signatur nur verifizieren, wenn er sicher sein kann, dass der öffentliche Schlüssel in der Tat zu dem behaupteten Inhaber gehört. Das erste Problem versucht man durch die Speicherung der geheimen Schlüssel auf Smartcards zu lösen. Für das zweite Problem gibt es mehrere Modelle, die im Folgenden angesprochen werden. Prinzipiell stelle man sich einen vertrauenswürdigen Dritten vor, der die öffentlichen Schlüssel von Teilnehmern elektronisch signiert und so mit seinem Namen für die Zugehörigkeit birgt. Dieser Dritte kann wiederum von einer weiteren Instanz zertifiziert sein und so fort. Somit kann eine Vertrauenshierarchie aufgebaut werden.

Vertrauensmodelle

Es werden im Folgenden die bekanntesten entwickelten Vertrauensmodelle vorgestellt, wobei lediglich ein unvollständiges Bild vermittelt werden kann. Dem Leser sei versichert, dass es weitere Modelle gibt und noch geben wird.

Das PGP-Modell

Die Software *Pretty Good Privacy* wurde 1991 als Kommandozeilen-Programm mit offenem Quellcode von Phil Zimmermann geschrieben. Das später von ihm gegründete Unternehmen PGP, Inc. wurde 1997 von der Firma Network Associates (NAI) gekauft, die heute noch die PGP-Urheberrechte besitzt [4] . NAI begann mit verschiedenen unabhängigen Kryptographen den OpenPGP Standard zu entwickeln, der 1998 als RFC 2440 verabschiedet wurde, wobei das „Open“ für offener Standard und nicht für OpenSource steht [5].

Ein öffentlicher PGP-Schlüssel besteht aus einem primären Schlüssel und optional mehreren sekundären Schlüsseln. Der primäre Schlüssel kann im Unterschied zu allen anderen Standards mehrere Benutzeridentitäten enthalten, die jeweils mehrere Signaturen tragen können. Die so entstehende Struktur wird „Schlüsselbund“ genannt. Das Vertrauen in einen öffentlichen PGP-Schlüssel baut sich über das Vertrauen in die einzelnen Signaturen zu den Benutzeridentitäten auf. Je mehr vertrauenswürdige Signaturen ein öffentlicher Schlüssel aus Sicht eines Nutzer trägt, um so stärker ist das Vertrauen in den öffentlichen PGP-Schlüssel. Dieses Vertrauensmodell wird auch als *Web of Trust* bezeichnet.

Der X.509 Standard

Der internationale Standard X.509 wurde von der *International Telecommunication Union - Sector: Telecommunication (ITU-T)* in der mittlerweile dritten Version veröffentlicht [6]. Darin wird ein Rahmenwerk für Authentifizierungsmechanismen zwischen Verzeichnissen und dessen Nutzern definiert. Alle in dem Standard definierten starken Authentifizierungsmechanismen setzen die Kenntnis der öffentlichen Schlüssel der jeweiligen Kommunikationsteilnehmer voraus, wobei die öffentlichen Schlüssel von einer Stelle ausgegeben werden, der ein Teilnehmer vertraut. Eine solche Stelle wird *Zertifizierungsinstanz* (*engl.* Certification Authority, CA) genannt. Eine CA signiert elektronisch den öffentlichen Schlüssel eines Teilnehmers gemeinsam mit Informationen über ihn in Form eines *Zertifikats*. Eine genaue Spezifikation der Zertifikate wird man aufgrund des Rahmencharakters des X.509 Standards dort nicht finden. Konkrete Profilbildungen von diesem Standard sind deshalb notwendig und entwickelt worden.

Das PEM-Modell

Das PEM-Modell stellt auf der Grundlage des X.509v1 Standards ein Betriebsmodell für die Benutzung von Zertifikaten dar [7]. Es sieht eine strenge Hierarchie von Zertifizierungsinstanzen vor, die drei Ebenen unterscheidet. Die *Internet Policy Registration Authority (IPRA)* agiert als Wurzelinstanz und wird von der Internetgemeinde betrieben. Sie stellt ausschließlich Zertifikate für *Policy Certification Authorities (PCA)* aus. Jeder Vertrauenspfad beginnt mit der IPRA. PCA's stehen für eine bestimmte Sicherheitspolitik, die beschreibt, welche Interessengruppe sie anspricht und für welche CA's Zertifikate ausgestellt werden. Für die Namensvergabe ist die Eindeutigkeit des Namens die wichtigste Eigenschaft. Dazu schreibt der Standard eine strenge Namenskonvention entsprechend der Hierarchie vor. In der heutigen Zeit hat sich dieses starre Modell als wenig praktikabel herausgestellt.

Der MailTrusT-Standard

Der TeleTrust Verein veröffentlichte 1996 eine erste Spezifikation *MailTrusT (MTT) [8],* die auf dem PEM-Modell aufbaut und mit Blick auf die Bedürfnisse von Gesundheitswesen und Behörden entwickelt wurde. Es folgte 1999 die erheblich erweiterte Version der Spezifikation, die wesentliche Teile der PKIX Arbeitsgruppe und des S/MIME Standards übernahm.

Das PKIX-Modell

Die von der Internet Engineering Task Force (IETF) gegründete Arbeitsgruppe *Internet X.509 Public Key Infrastructure (PKIX)* [9] entwickelte ein Zertifikatsprofil und ein Betriebsmodell, die für die Umsetzung von X.509v3 Zertifikaten im Internet geeignet sind. Das PKIX-Modell definiert die Komponenten *Klienten* (PKI-Nutzer), *Zertifizierungsinstanz*, optional die *Registrierungsinstanz (RA)* und das *Verzeichnis*. Weiterhin werden *Managementprotokolle* bezüglich der Interaktionen der einzelnen Komponenten beschrieben

wie das Format und die Behandlung einer Zertifikatsanfrage. Im Gegensatz zur strengen Top-Down Hierarchie des PEM-Modells ist es zum Beispiel möglich, einen Vertrauenspfad mit einer CA der Domäne des Nutzers beginnen zu lassen, die für den Anwendungsfall „Internet“ meistens die vertrauenswürdigste ist. Doch auch die PKIX-Spezifikation bietet bezüglich einer konkreten Implementierung viel Spielraum und bedarf deshalb einer Profilbildung.

Der ISIS-Standard

Das *European Telecommunications Standards Institute (ETSI)* entwickelte ein Zertifikatsprofil der PKIX-Spezifikation, das auf die Langlebigkeit elektronischer Signaturen abgestimmt ist, d. h. es wird nicht der Anwendungsfall der Verschlüsselung betrachtet. Die Gruppe *T7 e. V.*, der die wichtigsten Trustcenter-Betreiber Deutschlands angehören, entwickelte den Industriestandard *ISIS (Industrial Signature Interoperability Specification)* [10], der wiederum auf dem ETSI-Standard aufbaut und die Anforderungen einer Akkreditierung für Zertifizierungsdiensteanbieter nach dem deutschen Signaturgesetz erfüllt.

Der ISIS-MTT Standard

Somit existieren allein in Deutschland zwei PKI-Standards, MTT mit Berücksichtigung von Verwaltungsanforderungen und ISIS mit Berücksichtigung von E-Commerce Anforderungen, die beide ein Profil des X.509 Standards darstellen, aber trotzdem bezüglich der Zertifikatsinhalte nicht kompatibel sind. Um hier Abhilfe zu schaffen, wurde der gemeinsame Standard ISIS-MTT entwickelt, der die Interoperabilität herstellen soll und sich zurzeit in der Evaluierungsphase befindet.

Der S/MIME Standard

Der von der IETF veröffentlichte Standard *Multipart Internet Mail Extension (MIME)* wurde unter Führung der Firma RSA Data Security, Inc. um Sicherheitsmechanismen wie Verschlüsselung und elektronische Signatur zum Standard *S/MIME* [11] erweitert. Dazu wurde als Zertifikatsformat ein Profil des X.509v3 Standards entwickelt. Als Management-Protokolle und Formate wie Zertifikatsanfragen wurden die ebenfalls von RSA, Inc. veröffentlichten *Public-Key Cryptography Standards (PKCS)* verwendet. Erst in späteren Versionen wurden von der PKIX-Arbeitsgruppe die PKCS in die PKIX-Spezifikation aufgenommen. Damit konnte in der neuesten S/MIMEv3 Version eine Kompatibilität zu PKIX hergestellt werden. Eine vollständige Implementierung des S/MIMEv3 Standards gibt es derzeit noch nicht. Es muss an dieser Stelle erwähnt werden, dass auf PGP-Basis ebenfalls eine MIME Erweiterung entwickelt wurde (PGP/MIME).

Smartcards

Zur Vertrauenswürdigkeit einer elektronischen Signatur gehört neben der eindeutigen Zuordnung des öffentlichen Schlüssels zu einem Nutzer die Einmaligkeit und sichere Aufbewahrung des zugehörigen geheimen Schlüssels. Man bedenke, dass jeder, der im Besitz des geheimen Schlüssels ist, im Namen des Inhabers Dokumente unterschreiben kann. Smartcards mit kryptographischem Koprozessor besitzen die Eigenschaft, den Schlüssel speichern und eine elektronische Signatur berechnen zu können. Dazu muss der geheime Schlüssel die Smartcard nicht verlassen.

Smartcard Schnittstellen

Wie man schon vermutet, ist der Weg von einer E-Mail eines Mailclients zu einer Smartcard lang und steinig. Zunächst benötigt der Mikroprozessor (z. B. Siemens, Philips) einer Smartcard ein Betriebssystem (z. B. TCOS, Deutsche Telekom; STARCOS, G&D; GemSAFE, Gemplus). Neben diesen proprietären Betriebssystemen wurde der Standard *JavaCard* entwickelt, sodass Anwendungen unabhängig von der Smartcard entwickelt und auch im Nachhinein auf die Karte geladen werden können. Eine Smartcard erhält den benötigten elektrischen Strom, die Taktfrequenz sowie die Ein- und Ausgabedaten über ein *Smartcard-Terminal* oder Lesegerät. Weiterhin muss der Mailclient in der Lage sein, mit dem Betriebssystem der Smartcard über das Terminal zu kommunizieren. Die Lösung kann ganz einfach sein, indem ein Hersteller einen Mailclient und ein Terminal mit Smartcard für ein bestimmtes Betriebssystem (meistens MS Windows) entwickelt. Es tritt bei dieser Lösung leider das Problem auf, dass der Mailclient ausschließlich mit dem speziellen Terminal und der speziellen Smartcard zusammenarbeitet. Aus diesem Grund wurde es notwendig, Standards zu entwickeln.

Smartcard Standards

Es existiert ein Standard *EMV* der Finanzwelt, der von *Europay*, *MasterCard* und *Visa* unterstützt wird und die neue Generation von Kreditkarten auf Smartcard-Basis betrifft. Der wohl bekanntere Standard *Global System for Mobil Communications (GSM)* betrifft die SIMM-Karten in Mobiltelefonen, die zum Beispiel geräteunabhängig eingesetzt werden können. Der neue Standard *Universal Mobile Telecommunications System (UMTS)* ist auf dem Weg. Für die Welt der Personal Computer (MS Windows) wurde der deutsche Standard *Homebanking Computer Interface (HBCI)* definiert sowie der internationale Standard *Personal Computer/ Smart Card (PC/SC)* entwickelt.

Das Windows 2000 (XP) stellt dazu einen *PC/SC Resource Manager* bereit, der die Schnittstelle zwischen Anwendungen und Smartcards bildet. Die Anwendung kennt lediglich einen PC/SC-kompatiblen

und zur Smartcard passenden *Cryptographic Service Provider (CSP)*. Prinzipiell weiß die Anwendung nicht einmal, ob es sich um einen Software- oder Hardwaretoken handelt, da der CSP auch Softwaretoken unterstützen kann. Bezüglich der Schnittstelle zwischen Anwendung und CSP gibt es wiederum zwei Standards, die Microsoft *CryptoAPI (CAPI)* und den *PKCS#11 (CRYPTOKI)* der Firma RSA, Inc.

Das Prinzip des PC/SC-Standards spiegelt sich in dem für die Linux-Welt und seit OS X für die Macintosh-Welt interessanten Standard des Projekts *Movement for the Use of Smart Cards in a Linux Environment (MUSCLE)* wieder. Es gibt weiterhin proprietäre Standards für Solaris und SunOS. Es muss an dieser Stelle gesagt werden, dass die Entwicklung der Integration von Smartcards in Betriebssysteme und der Smartcards selbst noch in den Kinderschuhen steckt, aber die Entwicklung stark vorangetrieben wird. Es ist daher aus heutiger Sicht nicht abschätzbar, welcher Standard sich durchsetzen wird oder welche Standards in fünf Jahren aktuell sein werden.

E-Mail-Signierung an der HU

Wie aus den vorangegangen Abschnitten hervorgeht, gibt es eine Fülle von Möglichkeiten, eine E-Mail zu signieren oder zu verschlüsseln. Ausschlaggebend für ein spezielles Vertrauensmodell ist zum einen die informationstechnische Relevanz einer elektronischen Signatur zu einer E-Mail (rein informativ, juristische Anerkennung) und zum anderen der Anwenderkreis.

Prinzipiell gibt es in Deutschland zurzeit vier Möglichkeiten, eine E-Mail standardkonform zu signieren oder zu verschlüsseln. Da wären die Protokolle S/MIMEv2, OpenPGP (PGP/MIME) und Mailclients, die die Formate MTTv2 oder ISIS unterstützen. Leider sind alle Formate zueinander inkompatibel, sodass ein „Springen“ zwischen den Welten nicht möglich ist.

Schon seit 1997 betreibt die HU eine Zertifizierungsinstanz *HU-CA*, die ihrerseits von der *Policy Certification Authority* des DFN Vereins (*DFN-PCA*) zertifiziert ist. Die HU-CA stellt für HU-Mitglieder S/MIMEv2-Zertifikate aus und signiert PGP-Schlüssel. Die Mailclients MS Outlook (Express) ab der Version 4.0 und Netscape Messenger ab der Version 4.7 unterstützen standardmäßig das S/MIMEv2-Protokoll sowie unter den MS Windows Betriebssystemen verschiedene Hardwaretoken wie Smartcards. Leider sehen PGP-Implementationen die Verwendung von Smartcards nicht vor.

Durch die Projektgruppe UVsec wurden verschiedene Hardwaretoken mit kryptographischen Fähigkeiten auf ihre Funktionalität geprüft. Das beschränkte sich auf Microsoft Betriebssysteme, da für alternative Betriebssysteme noch keine geeigneten Einbindungen zur Verfügung stehen. Besonderes Augenmerk wurde auf die Interoperabilität zwischen Anwendungen, Smartcards und Terminals gelegt. Dabei wurden positive Erfahrungen mit den Firmen Gemplus und Datakey gemacht.

Die Smartcard GPK8000 (*G*emplus *P*ublic *K*ey) ist eine für die Anforderungen einer PKI entwickelte PC/SC-konforme Smartcard der Firma Gemplus (Frankreich). Das Kartenmanagementtool und die Schnittstellen werden durch das *GemSAFE Enterprise* Tool bereitgestellt. Dieses Tool enthält ebenfalls ein Plug-In für den kartengestützten Nutzerzugang zu Windows2000.

Die Datakey Smartcard Model 330 ist die PC/SC-konforme PKI-fähige Smartcard der Firma Datakey (US). Das Datakey Softwarepaket *SignaSURE* enthält das Administrationstool für die Smartcard, die MS CAPI und PKCS#11 Schnittstellen sowie ein Microsoft Windows LogOn.

Als Smartcard-Terminals finden die als PCMCIA-Karte und Desktopgerät erhältlichen PC/SC-konformen Lesegeräte *GemPC410* und *GemPC400* der Firma Gemplus Anwendung. Diese Geräte werden ebenfalls von der Firma Cherry in Tastaturen integriert. Weiterhin haben sich die PC/SC-konformen Terminals der Firma Towitoko (hier *Chipdrive extern*) als sehr tauglich erwiesen.

Abschließend kann man sagen, dass die Installation der Softwarepakete und der Terminals der Firmen Gemplus und Datakey keine Probleme bereitet. Weiterhin können beide Systeme im Netscape Messenger und in Outlook (Express) nebeneinander benutzt werden, wobei sie stabil laufen. Es muss darauf hingewiesen werden, dass dies in dem Zusammenhang nicht selbstverständlich ist. Damit wäre es vertretbar, die Systeme auf Produktionsrechnern zum Zwecke des E-Mail-Signierens zu installieren.

Quellennachweis

[1] http://www.hu-berlin.de/rz/projekte/uvsec/
[2] http://www.hu-berlin.de/rz/projekte/uvsec/berichte/vortrag/digsig/
[3] http://www.hu-berlin.de/rz/projekte/uvsec/berichte/vortrag/auffunkt/
[4] http://www.pgp.com/
[5] http://www.pgpi.org/
[6] http://www.itu.int/rec/recommendation.asp?type=folders&lang=e&parent=T-REC-X.509
[7] http://www.ietf.org/rfc/rfc1422.txt?number=1422
[8] http://www.teletrust.de/glossar.asp?id=60930&Sprache=D_&HomePG=0
[9] http://www.ietf.org/html.charters/pkix-charter.html
[10] http://www.t7-isis.de
[11] http://www.rsasecurity.com/standards/smime/about.html

Matthias Schwan
matthias.schwan@rz.hu-berlin.de