Script zum Workshop

Kryptografische Aspekte der Sicherheit bei der Nutzung internetbasierter Kommunikation

Das Script ist als begleitende Dokumentation eines Workshops und nicht als eigenständiges Lehrmaterial entwickelt worden.

Inhaltsverzeichnis

Vertrauen ist gut...

Kommunikation findet heute zunehmend auf technischer Basis statt. Als Beispiele seien an dieser Stelle die telefonische Kommunikation, insbesondere unter Verwendung von Handys, und die Kommunikation per E-Mail genannt.

Aber auch Alltagsvorgänge wie das Öffnen eines Fahrzeuges per Funk-Fernbedienung und das Lösen der Wegfahrsperre oder die Bezahlung von Waren per EC-Karte sind Beispiele für technisch basierte Kommunikation, wenngleich uns das auch nicht immer bewusst ist.

In all diesen Fällen ist es bei näherem Hinschauen sicher wünschenswert, wenn diese Kommunikation im weitesten Sinne vertraulich bleibt. Stellen wir uns vor, jemand registriert (zufällig?) die Funk-Kommunikation zwischen unserer Funk-Fernbedienung und unserem Auto und kann dieses dann anschließend genauso öffnen, wie wir auch. Das wäre wohl kaum in unserem Interesse. Oder wer möchte schon, dass seine per Handy, also per Funk geführten Gespräche genauso mitgehört werden können, wie (heute noch) der Polizeifunk, wenngleich es laut Gesetz nicht gestattet ist ...

In all diesen Fällen steht die Frage, wie die Vertraulichkeit der Kommunikation gewährleistet werden kann. Hier kommt die Kryptologie zum Tragen, die sich im Prozess der zunehmenden Nutzung der Kommunikationsmöglichkeiten zu einer mathematisch basierten Wissenschaft entwickelt hat.

Die Kryptologie (von griechisch κρύπτος – verborgen und λόγος – Lehre) ist eine Wissenschaft, die sich mit technischen Verfahren für die Informationssicherheit beschäftigt. Sie lässt sich in die beiden Bereiche Kryptografie und Kryptoanalyse unterteilen. Dabei ist die Kryptografie (von griechisch κρύπτος – verborgen und γράφειν – schreiben) im ursprünglichen Sinne die Wissenschaft der Verschlüsselung von Informationen. Die Kryptoanalyse stellt gewissermaßen das Gegenstück zur Kryptografie dar und befasst sich mit der Analyse verschlüsselter Informationen mit dem Ziel, diese zu entschlüsseln und damit vor allem die Sicherheit der verwendeten kryptografischen Verfahren nachzuweisen bzw. zu bewerten.

Im Zusammenhang mit der immensen Nutzung des Internets für Kommunikationszwecke gewinnen Fragen der Vertraulichkeit und der Sicherheit der Kommunikation zunehmend an Bedeutung.

Wichtig

Wichtig ist dabei vor allem die Erkenntnis, dass nicht nur die Anwendungen auf einem Computer und die Übertragung von Daten sicher sein müssen, sondern dass vor allem die Anwender über ein Mindestmaß an Verständnis und vor allem Bewusstsein gegenüber Fragen der Sicherheit bei der internetbasierten Kommunikation haben müssen. Allein durch technische Maßnahmen ist keine hinreichende Sicherheit erreichbar.

Genau diesem Anliegen widmet sich dieses Script in Begleitung einer Weiterbildung zu Fragen der Sicherheit bei der internetbasierten Kommunikation.

Neben einigen grundlegenden, dem Verständnis der Sache dienenden Bemerkungen zur Kryptologie sollen vor allem die heute üblichen Verschlüsselungsverfahren vorgestellt und deren Einbindung in verbreitete, insbesondere freie Software dargestellt werden.

Ziele der Kryptografie

Die moderne Kryptografie hat vier Hauptziele zum Schutz von Informationen:

- ***Vertraulichkeit*** (Zugriffsschutz): Nur dazu berechtigte Personen sollen in der Lage sein, die Daten oder die Nachricht zu lesen oder Informationen über ihren Inhalt zu erlangen. Ein Unbefugter soll dazu „praktisch" nicht in der Lage sein.
- ***Authentizität*** (Fälschungsschutz): Der Urheber der Daten oder der Absender der Nachricht soll eindeutig identifizierbar sein, und seine Urheberschaft sollte nachprüfbar sein.
- ***Integrität*** (Änderungsschutz): Der Empfänger soll in der Lage sein festzustellen, ob die Daten oder die Nachricht nach ihrer Erzeugung verändert wurden.
- ***Verbindlichkeit*** (Nichtabstreitbarkeit): Der Urheber der Daten oder Absender einer Nachricht soll nicht in der Lage sein seine Urheberschaft zu bestreiten, d. h. sie sollte sich gegenüber Dritten nachweisen lassen. Leugnen sei also zwecklos.

Mit diesen Zielen unterscheidet sich die Kryptografie von der Steganografie (griechisch στεγανός – schützend, verdeckt und γράφειν – schreiben), deren Anliegen es ist, eine geheime Information zu verbergen, zu tarnen. Dieses Verfahren ist jedem Büromitarbeiter bekannt, der schon einmal einen präparierten Ordner gesehen hat, dessen Inhalt eine kleine Flasche Hochprozentiger und zwei Gläser sind. Niemand, der nicht eingeweiht ist, kommt auf die Idee, dass sich ganz unscheinbar im Ordnerregal der gute Cognac verbirgt. Heute werden steganografische Verfahren beispielsweise verwendet, um Copyright-Informationen in Bildern zu hinterlegen und damit Urheberrechte zu belegen.

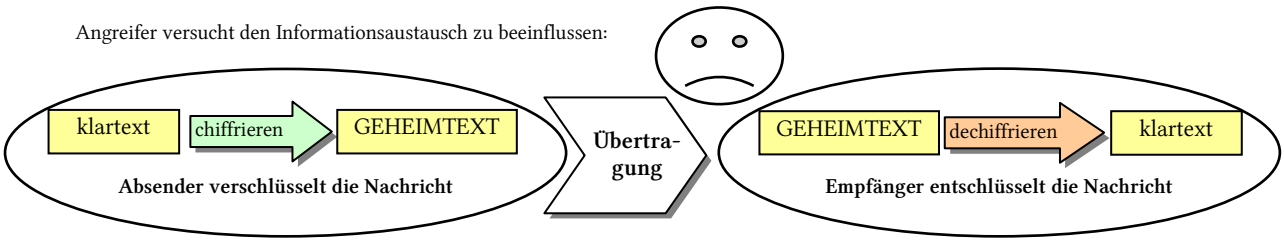
Im Gegensatz zum steganografischen Prinzip der Verschleierung (security by obscurity) geht es bei der Kryptografie gar nicht darum, etwas zu verbergen oder zu tarnen. Im Gegenteil: Eine Information wird als verschlüsselte Information ganz öffentlich behandelt. Hier gilt der von Auguste Kerckhoffs 1883 formulierte Grundsatz der modernen Kryptografie:

„Die Sicherheit eines Kryptosystems darf nicht von der Geheimhaltung des Algorithmus abhängen. Die Sicherheit gründet sich nur auf die Geheimhaltung des Schlüssels."

Der Vorteil dieses Prinzips besteht darin, dass viele Menschen die Verschlüsselungsverfahren auf ihre Sicherheit hin testen und bewerten können und damit eine Aussage über deren Qualität und wirksame Verwendbarkeit möglich ist. Die Praxis hat gezeigt, dass Verfahren, die geheimgehalten und von ihren Anwendern als sicher bezeichnet wurden, sich als schwach herausgestellt haben und bereits vor einer Offenlegung gebrochen werden konnten.

Konventionen

Die von einem Absender zu einem Empfänger zu übermittelnde Information, üblicherweise ein Text oder eine andere Zeichenfolge, heißt im Sprachgebrauch der Kryptografie Klartext. Dieser wird üblicherweise durch Kleinbuchstaben dargestellt. Die verschlüsselte Nachricht, also die tatsächlich übertragene Zeichenfolge, wird als Geheimtext (Ciphertext, früher auch Kryptogramm) bezeichnet. Dieser wird üblicherweise durch Großbuchstaben dargestellt. Das Erzeugen des Geheimtextes aus dem Klartext wird als chiffrieren, der umgekehrte Vorgang als dechiffrieren bezeichnet.

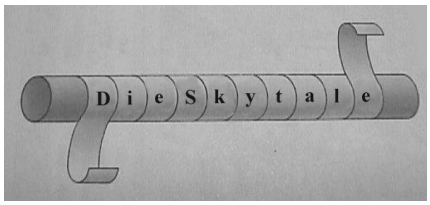


Historische kryptografische Verfahren

Verschlüsselungsverfahren, mit dem Ziel Informationen vertraulich zu übermitteln, sind bereits aus dem Altertum bekannt.

Beispiel

Skytale von Sparta (500 v. u. Z.)



Die Skytale ist ein Stab aus Holz. Um diesen Stab wurde ein schmales Band aus Pergament spiralförmig gewickelt. Darauf wurden die zu verschickenden Nachrichten geschrieben. Danach wurde das abgewickelte Band von einem Boten zum Empfänger gebracht. Um die Nachricht zu entschlüsseln benötigte der Empfänger lediglich einen Stab desselben Durchmessers wie der des Absenders. Liest jemand das Papier im abgerollten Zustand, so kann dieser nur Teilstücke der zuvor geschriebenen Wörter erkennen. Weil er in die Verschlüsselungstechnik nicht eingeweiht ist, ergeben diese Wortstücke aber für denjenigen keinen Sinn.

Diese Art der Verschlüsselung wird als Transpositions-Chiffrierung bezeichnet. Dabei werden nicht (Klartext-) Zeichen durch andere (Chiffre-) Zeichen ersetzt, sondern die Reihenfolge der Zeichen wird vertauscht. Auf „moderne“ Weise kann das Verfahren durch eine Tabelle dargestellt werden, deren Inhalt zeilenweise zu lesen ist, aber spaltenweise notiert und übermittelt wird.

Beispiel

Transpositionsverfahren

t	r	a	n	s
p	o	s	i	t
i	o	n	s	c
h	i	f	f	r
i	e	r	u	n
g	i	s	t	g
a	r	n	i	c
h	t	s	c	h
w	e	r		

Der „Schlüssel“, also in diesem Fall die Anzahl der Spalten sei hier 5. Damit ergibt sich folgender Geheimtext:

TPIHIGAHWROOIEIRTEASNFRSNSRNISFUTICSTCRNGCH

Zum Entschlüsseln wird der Text senkrecht in eine Tabelle mit der vereinbarten Spaltenzahl geschrieben und zeilenweise gelesen.

Ein Angriff ist entweder über statistische Verfahren oder über probieren mit verschiedenen Matrizen möglich. Um Angriffe zu erschweren, können die Spalten zusätzlich noch permutiert [untereinander vertauscht] werden. Die Permutationsvorschrift wird damit zum Bestandteil des Schlüssels.

Das Prinzip der Transpositions-Chiffrierung wird heute als Teilprozess komplexer Verschlüsselungsalgorithmen verbreitet angewendet.

Beispiel

Cäsar-Verschlüsselung

Von Julius Cäsar wird berichtet, dass er zur Übermittlung von geheimen Nachrichten ein kryptografisches Verfahren verwendet haben soll. Die von ihm verwendete Chiffre ergibt sich dadurch, dass als Geheimtextalphabet das um 3 Zeichen versetzte Klartextalphabet verwendet wird. Dies wird als Verschiebechiffre bezeichnet.

Klartext:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Geheimtext:	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Die Chiffrierung erfolgt dadurch, dass der Klartextbuchstabe durch den darunter stehenden Geheimtextbuchstaben ersetzt wird. Das Klartext-Wort „geheimnis“ würde damit die Geheimtext-Zeichenfolge „JHKHLPQLV“ ergeben, die scheinbar sinnlos ist. Die Dechiffrierung erfolgt durch die Umkehrung des Verfahrens: Jedes Zeichen wird durch das jeweils darüberstehende ersetzt.

Häufig wird die Frage gestellt, warum Cäsar das Geheimalphabet gerade um drei Zeichen versetzt hat. Die Antwort ist einfach – reine Willkür, denn es ist letztlich völlig egal, um wie viele Stellen das Alphabet verschoben wird. Nur Sender und Empfänger müssen die Anzahl der verschobenen Stellen wissen, sonst ist die Dechiffrierung mit reichlich Arbeit verbunden… ☺

Die Anzahl der verschobenen Stellen ist somit der Schlüssel bei dieser Art der Chiffrierung. Wird Cäsars Verschlüsselung verwendet, so gibt es genau so viele Schlüssel, wie das Alphabet Zeichen hat – 26. Dabei sind aber nur 25 sinnvoll, denn das Ergebnis der Verschiebung a → A kann wohl schwerlich als „Geheimtext“ bezeichnet werden. ☹

Um die Arbeit des Chiffrierens und Dechiffrierens zu vereinfachen, kann der Versuch unternommen werden, diese Tätigkeit von Maschinen ausführen zu lassen. Heute werden dazu vor allem Computer verwendet, die i. A. mit Zahlen viel besser umgehen können als mit Buchstaben. So kann beispielsweise A durch 1, B durch 2, C durch 3 … Z durch 26 ersetzt werden. In diesem Fall bedeutet eine Verschiebechiffre nichts anderes als die Addition einer Zahl. Die Cäsar-Verschlüsselung wäre in diesem Fall die Addition der Zahl 3. Schwierig wird es hier nur dann, wenn das Ergebnis der Addition größer als 26 ist, weil es ja nur Zeichen bis zur 26 gibt. In diesem Fall muss für alle Werte größer 26 wieder von vorn mit Zählen begonnen werden. Wenn beispielsweise 25 verschlüsselt werden soll, dann ist das formale Ergebnis der Addition 28. Diese Zahl liegt 2 über der 26, es muss also, beginnend bei 1, zwei Stellen weitergezählt werden: Ergibt 2.

Einfacher ist es, wenn das Ergebnis größer ist als 26, 26 davon abzuziehen: (25 + 3) − 26 = 2. In der Mathematik wird das (25 + 3) mod 26 geschrieben und modulo 26 gesprochen.

Hinweis

Modulo (mathematisches Formelzeichen mod) ist eine mathematische Funktion, die den Rest aus der Division zweier ganzer Zahlen angibt.
Beispiele:
7 mod 2 = 1 (gesprochen: „Sieben modulo zwei ist gleich eins."
[denn 7 : 2 = 3, Rest 1])
8 mod 3 = 2

Die Funktion mod ist in allen gängigen Programmiersprachen implementiert.

An dieser Stelle wird deutlich, dass der Algorithmus der Chiffrierung wie auch der Dechiffrierung sehr gut über eine entsprechende Software durch einen Computer ausgeführt werden kann.

symmetrische Chiffrierverfahren

Als symmetrisch werden Chiffrierverfahren bezeichnet, die sowohl für die Chiffrierung als auch für die Dechiffrierung den selben Schlüssel verwenden.

monoalphabetische Chiffrierung

Die Einfachheit der Berechnung von Transpositionschiffren ist jedoch auch von Nachteil: Die wenigen Möglichkeiten der Verschiebung von Zeichen sind für einen Computer kein Problem – sie werden kurzerhand alle durchprobiert bis der Geheimtext entschlüsselt ist. Deswegen wurde diese Art der Chiffrierung weiterentwickelt, indem die Klartextzeichen durch völlig andere, willkürliche Zeichen zu ersetzen sind, beispielsweise:

Klartext:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Geheimtext:	Q	W	E	R	T	Z	U	I	O	P	A	S	D	F	G	H	J	K	L	Y	X	C	V	B	N	M

oder

Klartext:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Geheimtext:	🖒	🖚	🖝	🖞	🖟	🖖	☺	😐	☹	💣	☠	⚐	⚑	✈	☼	🌢	❄	✞	✝	✠	✠	✡	☪	✌	👌	🖒

In einem solchen Fall wird von einer monoalphabetischen Chiffrierung gesprochen, weil jedem Zeichen Klartext immer genau ein Zeichen Geheimtext entspricht. Da auf diese Art alle nur denkbaren Umordnungen (Permutationen) des Alphabets verwendet werden können, ergeben sich sehr viele ($26! \sim 4\ 10^{26}$) Möglichkeiten der Verschlüsselung. Hier „einfach mal durchprobieren" ist auch mit moderner Rechentechnik nicht mehr sinnvoll.

Jedoch ergibt sich hier ein anderes Problem: Wie kann ich mir einen solchen Schlüssel merken? Hier wird üblicherweise eine Methode verwendet, die aus einem Schlüsselwort und einem Schlüsselbuchstaben besteht.

Beispiel

Schlüsselwort: GEHEIMSCHRIFT Schlüsselbuchstabe: f

Dabei werden zunächst aus dem Schlüsselwort alle Buchstaben, die wiederholt auftreten, entfernt – in diesem Beispiel wird aus dem Schlüsselwort GEHEIMSCHRIFT die Zeichenfolge GEHIMSCRFT. Diese Zeichenfolge wird nun so unter das Klartextalphabet geschrieben, dass sie unter dem Schlüsselbuchstaben beginnt:

Klartext:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Geheimtext:						G	E	H	I	M	S	C	R	F	T											

Anschließend werden die verbleibenden Buchstaben in alphabetischer Reihenfolge aufgefüllt:

Klartext:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Geheimtext:	V	W	X	Y	Z	G	E	H	I	M	S	C	R	F	T	A	B	D	J	K	L	N	O	P	Q	U

Auf diese Art und Weise sind einerseits sehr viele Verschlüsselungsmöglichkeiten realisierbar, andererseits kann der Schlüssel so gestaltet werden, dass er „merkbar“ ist. Jedoch ist eine monoalphabetische Chiffrierung ebenfalls grundsätzlich angreifbar, vor allem bei der Verwendung von Computern. Die Ursache für eine solche Angriffsmöglichkeit ist jedoch nicht das Verschlüsselungsverfahren, sondern der zu verschlüsselnde Klartext.

Eine natürliche (moderne Schrift-)Sprache hat wenige Buchstaben, die zudem recht ungleichmäßig verteilt sind. Und da jedem Zeichen Klartext immer das gleiche Zeichen Geheimtext zugeordnet ist, kann über eine Häufigkeitsanalyse des Auftretens von Buchstaben sowie von Bigrammen (Folge zweier aufeinanderfolgender Buchstaben) ein (großer) Teil der Klartextzeichen ermittelt werden – der Rest wird erraten. Bis auf das Erraten sind das wunderbar durch Rechentechnik zu leistende Aufgaben. Aus diesem Grund werden heute monoalphabetische Chiffrierungen üblicherweise für nichtnatürliche Sprachen verwendet. Für natürliche Sprachen kommen polyalphabetische Chiffrierungen zum Einsatz.

DES – Data Encryption Standard

DES ist das wohl am weitesten verbreitete monoalphabetische Verschlüsselungsverfahren. Es wurde in den 70-er Jahren von IBM in Kooperation mit der NSA entwickelt und 1976 standardisiert. DES verschlüsselt jeweils 64-Bit-Datenblöcke mit einem 56-Bit Schlüssel. Der Algorithmus von DES wurde von Beginn an vollständig publiziert, um dem Kerkhoffs-Prinzip (siehe Seite 2) gerecht zu werden. Die Schlüssellänge wurde jedoch von vielen Kritikern als zu gering angesehen, was auch zu Spekulationen um den Einfluss der NSA bei der Entwicklung des DES Nahrung gab. Tatsächlich gelang es 1999 auf Grund der fortgeschrittenen Leistungsfähigkeit moderner Rechentechnik, einen Text durch einen Brute-Force-Angriff [stures Ausprobieren aller Möglichkeiten] in etwas mehr als 22 Stunden zu dechiffrieren.

Die Entwicklung wurde in zwei Richtungen weitergetrieben: Zum Einen wurde das DES-Verfahren weiterentwickelt zum Triple-DES. Dabei wird der DES-Algorithmus drei mal angewendet, indem ein Text erst verschlüsselt, dann mit einem anderen Schlüssel entschlüsselt und anschließend mit einem weiteren Schlüssel erneut verschlüsselt wird. Zum Anderen wurde ein völlig neues Verfahren entwickelt, der AES (Advanced Encryption Standard), welches im Ergebnis einer weltweiten Ausschreibung von den belgischen Mathematikern Joan Daemen und Vincent Rijmen entwickelt wurde. Nach ihnen wird der Algorithmus auch als Rijndael bezeichnet. Seit 2002 ist dieses Verfahren gültiger US-Standard.

Der Rijndael-Algorithmus besitzt eine variable Blockgröße von 128, 192 oder 256 Bit und eine variable Schlüssellänge von 128, 192 oder 256 Bit. Rijndael bietet ein sehr hohes Maß an Sicherheit. Das Verfahren wurde eingehenden kryptoanalytischen Prüfungen unterzogen. AES schränkt die Blocklänge auf 128 Bit ein, während die Wahl der Schlüssellänge von 128, 192 oder 256 Bit unverändert übernommen worden ist. Anhand der Schlüssellänge wird zwischen den drei AES-Varianten AES-128, AES-192 und AES-256 unterschieden.

Der Algorithmus ist frei verfügbar und darf ohne Lizenzgebühren eingesetzt sowie in Software bzw. Hardware implementiert werden. AES ist in den USA für staatliche Dokumente mit höchster Geheimhaltungsstufe zugelassen.

polyalphabetische Chiffrierung

Der grundlegende Nachteil einer monoalphabetischen Chiffrierung ist die Tatsache, dass bei der Verschlüsselung von Informationen, die in einer natürlichen Sprache vorliegen, durch eine Häufigkeitsanalyse der Zeichen die Chiffrierung gebrochen werden kann. Die Sicherheit der monoalphabetischen Verfahren DES und AES beruht gerade darauf, dass Daten verschlüsselt werden, die binär und blockweise vorliegen. Ein 64-Bit-Block ist, selbst wenn er ASCII-Text repräsentiert, kein linguistisch vernünftig auswertbarer Text auf den statistische Verfahren sinnvoll anwendbar wären.

Bereits in der Vergangenheit wurden daher Verfahren entwickelt, eine polyalphabetische Chiffrierung zu realisieren. Polyalphabetisch bedeutet, dass ein und demselben Klartextzeichen im Geheimtext unterschiedliche Geheimtextzeichen entsprechen können. Damit sind Häufigkeitsanalysen als wichtige statistische kryptoanalytische Methode nicht mehr sinnvoll. Das wichtigste an dieser Stelle zu nennende polyalphabetische Verschlüsselungsverfahren ist die Vigenère-Chiffre.

Die Vigenère-Verschlüsselung

Diese Art der Chiffrierung wurde von dem französischen Diplomaten Blaise de Vigenère im Jahr 1586 veröffentlicht.

Die Grundidee der Vigenère-Verschlüsselung besteht darin, mehrere unterschiedliche monoalphabetische Verschlüsselungen nach einem bestimmten Schema wechselweise zu benutzen. Das entwickelte Verfahren war damals so gut, dass erst etwa 300 Jahre später Möglichkeiten gefunden wurden, diese Verschlüsselung systematisch anzugreifen. Zu erwähnen ist hier vor allem der sogenannte Kasiski-Test, ein von dem preußischen Infanteriemajor Friedrich Wilhelm Kasiski 1863 veröffentlichtes Verfahren zur Kryptoanalyse.

Beispiel

Als Beispiel soll im Folgenden der Klartext „geheimmitteilung“ mit Hilfe des Schlüsselwortes „MERKUR“ nach dem Vigenère-Verfahren verschlüsselt werden.

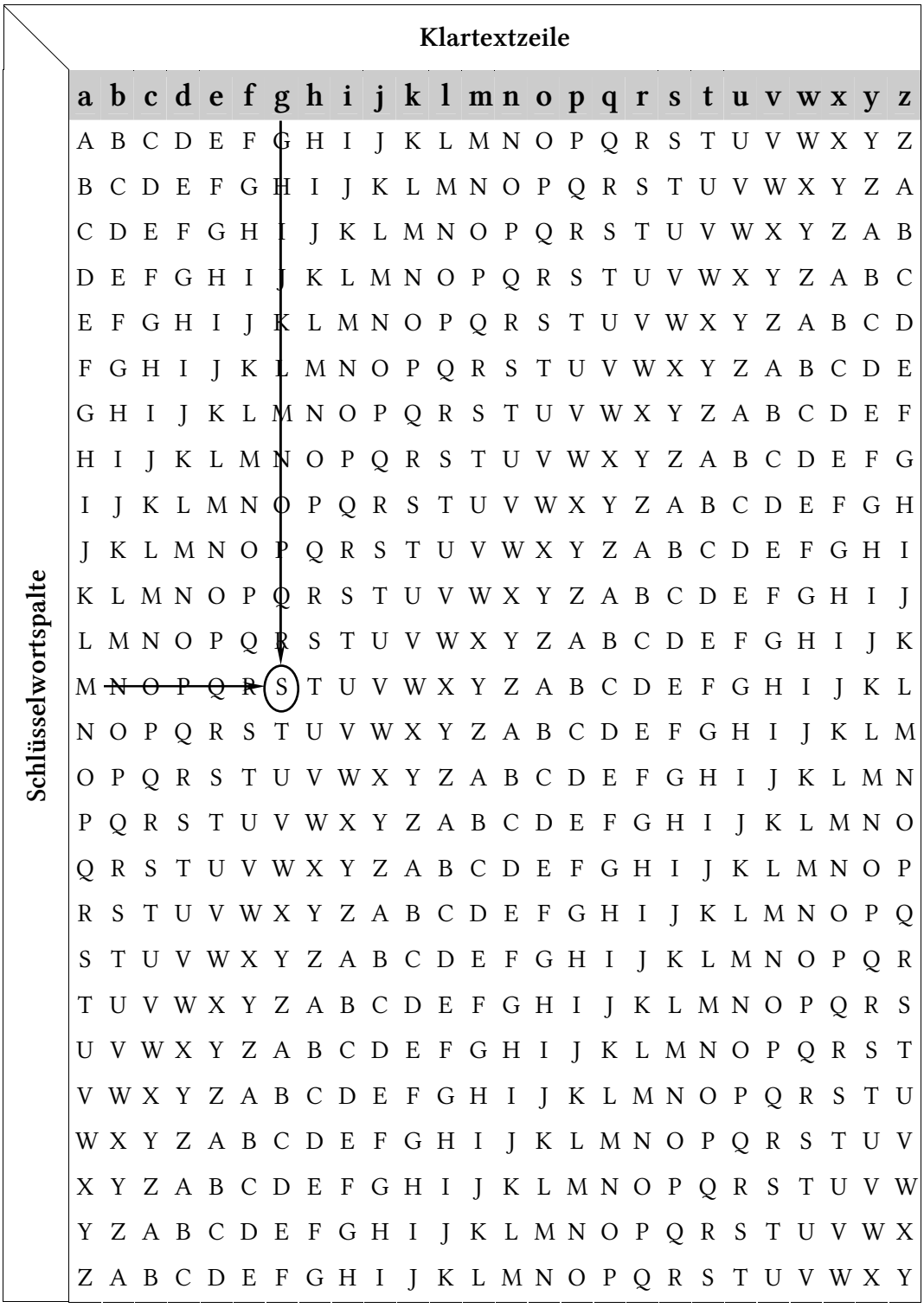
Um das zu auszuführen, wird das sogenannte Vigenère-Quadrat benötigt, welches auf der folgenden Seite dargestellt ist.

Ausführung

Zunächst werden das Schlüsselwort und der Klartext untereinander geschrieben und dabei das Schlüsselwort so oft wiederholt, wie der Klartext lang ist:

Klartext:	g	e	h	e	i	m	m	i	t	t	e	i	l	u	n	g
Schlüsselwort:	M	E	R	K	U	R	M	E	R	K	U	R	M	E	R	K

Nun wird im Vigenère-Quadrat für das erste Klartextzeichen [Spalte g] in der Zeile des zugehörigen Schlüsselwortbuchstabens [Zeile M] das entsprechende Geheimtextzeichen gefunden.



Klartext:	g	e	h	e	i	m	m	i	t	t	e	i	l	u	n	g									
Schlüsselwort:	M	E	R	K	U	R	M	E	R	K	U	R	M	E	R	K									
Geheimtext:	S	I	Y	O	C	D	Y	M	K	D	Y	Z	X	Y	E	Q									

Das Schema, um das erste Geheimtextzeichen zu ermitteln, ist in das Vigenère-Quadrat eingezeichnet. Deutlich ist zu erkennen, dass gleiche Klartextzeichen unterschiedlichen Geheimtextzeichen entsprechen und umgekehrt.

Der entscheidende kryptoanalytische Ansatz, der auch dem Kasiski-Test zugrunde liegt, besteht in der Periodizität der Verwendung des Passwortes.

Da das Passwort fortlaufend immer wieder auf den Klartext angewendet wird, bedeutet das letztlich eine sich immer wiederholende Cäsar-Verschlüsselung, die natürlich wieder über die Häufigkeitsverteilung angreifbar ist. Der Ansatz des Kasiski-Testes besteht deshalb auch zuerst darin, die Passwortlänge und damit die Periodizität zu ermitteln. Je länger der Text und je kürzer das Passwort sind, desto erfolgreicher wird ein Angriff auf eine Vigenère-Verschlüsselung sein.

Vernam-Verschlüsselung und One-Time-Pad

In der Praxis sollte also der Schlüssel ein recht langer sein. Tatsächlich wird das sogar so gehandhabt, dass der Schlüssel genauso lang ist, wie der Geheimtext. So kann beispielsweise der Text eines Buches als Schlüssel dienen. Dabei ergibt sich sogar der Vorteil, dass dieser Schlüssel einfach übermittelt werden kann. Der Empfänger braucht beispielsweise nur zu wissen: Victor Klemperer, LTI, um den Geheimtext mittels des folgenden „Schlüsselwortes“ dechiffrieren zu können:

> *Es gab den BDM und die HJ und die DAF und ungezählte andere solcher abkürzenden Bezeichnungen. Als parodierende Spielerei zuerst, gleich darauf als ein flüchtiger Notbehelf des Erinnerns, als eine Art Knoten im Taschentuch, ...*

Wenn ein solcher Schlüssel verwendet wird, schlagen alle Versuche irgendwelche Periodizitäten feststellen zu wollen fehl. Da aber der Schlüssel ein deutschsprachiger Text ist, werden auch hier statistische Charakteristika sichtbar, so dass das Verfahren nicht sicher ist. Dieser Spezialfall, dass das Passwort genauso lang ist, wie der zu verschlüsselnde Text, wird als Vernam-Verschlüsselung bezeichnet. Ein Sonderfall wiederum der Vernam-Verschlüsselung ist das sogenannte One-Time-Pad. Hier besteht das Schlüsselwort aus einer im mathematischen Sinne zufälligen Folge von Buchstaben. Dieses Verfahren ist als einziges sogar theoretisch sicher. Jedoch ist der Schlüsselaustausch nicht so ganz einfach. Die Kommunikation über das „rote Telefon“ wurde auf diese Weise verschlüsselt – hier konnte der Schlüssel über das Diplomatengepäck ausgetauscht werden.

Neben den Blockalgorithmen gibt es noch sogenannte Stromchiffren. Hier wird aus einem geheimen Schlüssel ein pseudozufälliger Bitstrom erzeugt, der zum Verschlüsseln eines Datenstromes verwendet wird. Das entspricht dem Prinzip eines One-Time-Pads, nur dass der Schlüssel nicht wirklich zufällig ist. Eine populäre Anwendung dieses Prinzips ist der Algorithmus A5, mit dessen Hilfe der Datenverkehr von GSM-Handys chiffriert wird.

Damit werden auch die Grenzen symmetrischer Verschlüsselungsverfahren deutlich: Der Austausch der Schlüssel muss auf einem „sicheren“ Weg erfolgen. Das kann ein zeitlich versetzter Austausch sein oder der Austausch über Vertrauenspersonen. In jedem Fall besteht hierin ein grundsätzlicher Schwachpunkt symmetrischer Verfahren. Dieses Problem besteht nur dann nicht, wenn Informationen zum Zwecke der Aufbewahrung und dem dabei gewünschten Schutz vor unbefugtem Zugriff chiffriert werden, weil dann chiffrieren und dechiffrieren von ein und derselben Person ausgeführt werden – der Schlüssel muss dann gar nicht übertragen werden.

Da diese Situation in der Praxis sehr häufig gegeben ist, sind heute die oben beschriebenen Verschlüsselungsverfahren DES bzw. 3DES und AES bzw. Rijndael oder verwandte Verfahren wie Blowfish am weitesten verbreitet. Sie haben zudem den Vorteil, dass sie mit modernen Computern nahezu ohne Verzögerung im Hintergrund automatisch ausgeführt werden.

asymmetrische Chiffrierverfahren

Der grundsätzliche Nachteil symmetrischer Chiffrierverfahren für den tatsächlichen Austausch von Nachrichten wurde bereits genannt: die Notwendigkeit, den gemeinsam zu verwendenden Schlüssel sicher auszutauschen. In der Praxis besteht aber die Notwendigkeit dafür sehr wohl, beispielsweise bei der verschlüsselten Übertragung von Daten per Internet – etwa bei der Bezahlung per Kreditkarte von im web getätigten Einkäufen oder bei der gesicherten Übertragung personenbezogener Daten.

Der Diffie-Hellmann Schlüsselaustausch

Eine erste praktikable Lösung für dieses Problem wurde von Whitfield Diffie und Martin Hellman 1976 veröffentlicht. Das von ihnen entwickelte Public-Key-Kryptosystem ist nicht geeignet für die Verschlüsselung oder Signatur von Daten, eignet sich jedoch sehr gut für den sicheren Austausch eines geheimen Schlüssels über einen öffentlichen Informationskanal. Die Sicherheit des Verfahrens beruht auf dem sogenannten diskreten Logarithmus-Problem.

Die übliche Berechnung des Logarithmus, also des Exponenten x aus den bekannten Werten a und a^x, ist aus der Schule bekannt. Anders verhält es sich, wenn nur mit ganzen Zahlen gerechnet wird und die Reste bei einer Teilung durch eine Primzahl p betrachtet werden.

Es ist für große p (typischerweise 1024 oder 2048 Bit lang) im Allgemeinen nicht möglich, aus dem Teilungsrest $a^x \bmod p$ bei bekannter Basis a die Größe x zu ermitteln. Zumindest wurde dafür bisher trotz größter Anstrengungen noch keine praktikable Methode gefunden.

Nach Diffie und Hellman vereinbaren die Kommunikationspartner zwei gemeinsame Zahlen a und p, und jeder wählt für sich jeweils einen zufälligen (geheimen, also nur ihm bekannten) Exponenten x und y.

Übermittelt nun Partner 1 an Partner 2 den Wert

$$X = a^x \bmod p$$

und Partner 2 an Partner 1 den Wert

$$Y = a^y \bmod p$$

dann können beide einen gemeinsamen Wert errechnen, ohne dass dieser über das Netz übertragen werden muss, nämlich

$$a^{xy} \bmod p$$

Dieser nur den beiden Kommunikationspartnern bekannte Wert wird als Schlüssel für eine symmetrische Verschlüsselung der zu übertragenden Daten verwendet. Der Diffie-Hellman-Schlüsselaustausch ist beispielsweise Grundlage für das Protokoll der Secure Shell (SSH2, OpenSSH) und ebenso für IPSec. Da es jedoch eine unmittelbare Interaktion beider Parteien voraussetzt, ist es beispielsweise nicht für eine Kommunikation per E-Mail verwendbar.

Das RSA-Public-Key-Verfahren

Hier wird grundsätzlich ein zweigeteilter Schlüssel verwendet, ein veröffentlichter, für jeden einsehbarer öffentlicher Schlüssel und ein geheimer, nur dem jeweiligen Schlüsselinhaber bekannter privater Schlüssel. Weil beide Teile des Schlüsselpaares getrennt voneinander verwendet werden, muss auch kein Schlüssel mehr ausgetauscht werden.

Der RSA Algorithmus gestattet sowohl die Verschlüsselung von Nachrichten als auch die Erstellung digitaler Signaturen und wurde als erstes ausgereiftes asymmetrisches Kryptosystem 1978 vorgestellt. Der Name RSA steht für die drei Entwickler des Systems Ronald L. Rivest, Adi Shamir und Leonard M. Adleman. Das Verfahren ist heute das populärste und hat bis heute auch intensivster Kryptoanalyse widerstanden. Das Verfahren beruht darauf, dass es mathematisch außerordentlich schwierig ist, sehr große Zahlen zu faktorisieren, also beispielsweise aus dem Produkt zweier sehr großer Primzahlen p*q diese beiden Primzahlen p und q zu ermitteln.

Der öffentliche Schlüssel (public key) ist ein Zahlenpaar (e,N) und der private Schlüssel (private key) ein Zahlenpaar (d,N), wobei N bei beiden Schlüsseln gleich ist. Dabei wird N als RSA-Modul, e als Ver- und d als Entschlüsselungsexponent bezeichnet. Diese Zahlen werden durch das folgende Verfahren erzeugt:

Ausführung

1. *zufällig wird ein Primzahlenpaar p, q gewählt, wobei $p \neq q$ ist*
2. *wird das RSA-Modul berechnet als $N = p*q$*
3. *wird die Eulersche Funktion $\varphi(N) = (p-1)*(q-1)$ ermittelt*
4. *wird eine zu $\varphi(N)$ teilerfremde Zahl e willkürlich gewählt, wobei $1 < e < \varphi(N)$ ist – diese Zahl ist der Verschlüsselungsexponent*
5. *wird der Entschlüsselungsexponent d als das multiplikativ Inverse zu e ermittelt, wobei folgende Kongruenz gelten muss: $e*d \equiv 1 \pmod{\varphi(N)}$*

Beispiel

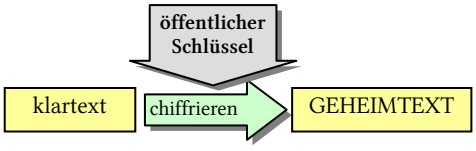
1. *gewählt werden p = 11 und q = 13*
2. *das RSA-Modul ist dann $N = p*q = 143$*
3. *die Eulersche Funktion ist $\varphi(N) = \varphi(143) = (p-1)*(q-1) = 120$*
4. *als Verschlüsselungsexponent e wird willkürlich 23 gewählt – 23 ist teilerfremd zu 120 und es gilt $1 < 23 < 120$*
5. *für den Entschlüsselungsexponent d muss folgende Gleichung gelten: $e*d + k*\varphi(N) = 1$. Mit Hilfe des erweiterten Euklidischen Algorithmus berechnet sich d als 47.*

In der Praxis werden als privater Schlüssel nicht nur N und d, sondern auch einige weitere, für die Dechiffrierung nützliche Zwischengrößen gespeichert. Wie erfolgen nun die Chiffrierung und die Dechiffrierung von Nachrichten?

Chiffrierung nach RSA Algorithmus

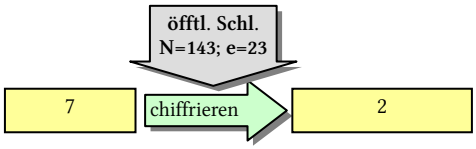
Ausführung

Zum verschlüsseln der Nachricht K wird berechnet:



$$C \equiv K^e \bmod N$$

Beispiel



verschlüsselt werden soll die Zahl 7, als öffentlicher Schlüssel ist bekannt N = 143 und e = 23:

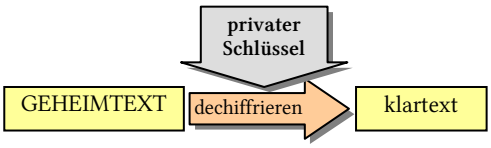
$C \equiv 7^{23} \bmod 143$

$7^{23} \bmod 143 = (((7^2)*7)^2*7)^2*7 \bmod 143 = 2$

Aus dem Klartext 7 ist damit der Geheimtext 2 ermittelt worden.

Dechiffrierung nach RSA-Algorithmus

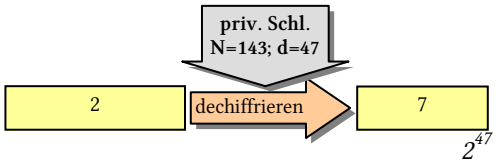
Ausführung



zum Entschlüsseln des Geheimtextes C wird berechnet:

$$K \equiv C^d \bmod N$$

Beispiel



entschlüsselt werden soll der Geheimtext 7, als privater Schlüssel bekannt sind N = 143 und d = 47

$K \equiv 2^{47} \bmod 143$

$2^{47} \bmod 143 = ((((2^2)^2*2)^2*2)^2*2)^2*2 \bmod 143 = 7$

Aus dem Geheimtext 2 wurde damit wieder der Klartext 7 ermittelt.

Wichtig

Für die praktische Verwendung bedeutet dies, dass der Empfänger einer verschlüsselten Nachricht durch Veröffentlichung seines öffentlichen Schlüssels quasi seine Bereitschaft bekunden muss, sich an einem verschlüsselten Austausch von Daten zu beteiligen. Ist dieser Teil des Schlüssels nicht bekannt, kann dieses Verschlüsselungsverfahren in der Kommunikation mit diesem Empfänger nicht verwendet werden.

Der RSA-Algorithmus wurde bisher noch nicht gebrochen. Die Betonung liegt jedoch auf dem Wörtchen „noch“. Sollte ein Verfahren gefunden werden, welches das Problem der Faktorisierung großer Primzahlen löst, dann ist dieses Verfahren nicht mehr sicher. Die Probleme, die dann in der Wirtschafts-Welt entstehen, sind vermutlich gar nicht abzusehen. Ein zweiter möglicher Angriffspunkt ist die Leistungsfähigkeit moderner Rechner. Sollte in einiger Zeit ein sogenannter Quantencomputer realisiert werden können, so ist auch damit zu rechnen, dass der RSA-Algorithmus damit gebrochen werden kann.

Heute ist für die sichere Verwendung dieses Verfahrens vor allem eine ausreichende Schlüssellänge wichtig. Diese beschreibt bei solchen Verfahren die Anzahl der Bit der Zahl p beziehungsweise pq, typischerweise 512, 1024 oder 2048. Diese Zahlen sind allerdings nicht mit den Schlüssellängen bei symmetrischen Verschlüsselungsverfahren vergleichbar. 128 Bit bei AES bieten beispielsweise deutlich mehr Sicherheit als 512 Bit bei RSA.

In der Praxis werden asymmetrische Verfahren üblicherweise nur eingesetzt, um pseudozufällige Klartexte wie Sitzungsschlüssel oder Hashes (siehe Seite 13)zu chiffrieren. Der Grund dafür ist, dass asymmetrische Verschlüsselungsverfahren im Vergleich zu symmetrischen sehr viel langsamer sind (etwa Faktor 1.000). Darüber hinaus sind asymmetrische Verfahren sehr anfällig gegenüber Implementierungsfehlern in der Anwendungssoftware.

Klartexte werden daher üblicherweise nicht direkt mit asymmetrischen Verfahren verschlüsselt, sondern nur Sitzungsschlüssel oder Hashwerte.

Das eröffnet die Möglichkeit, mit Hilfe sogenannter hybrider Verfahren eine angemessene Sicherheit beim Austausch von Daten, beispielsweise per E-Mail zu erreichen, wo beide beteiligten Partner nicht unmittelbar interagieren. Dazu wird mit Hilfe geeigneter Verfahren zuerst ein Schlüssel für eine symmetrische Verschlüsselung generiert und mit diesem die Nachricht verschlüsselt. Anschließend wird mit Hilfe des öffentlichen Schlüssels des Empfängers der symmetrische Schlüssel verschlüsselt und der Nachricht angehängt. Der Empfänger entschlüsselt mit seinem privaten Schlüssel zuerst den symmetrischen Schlüssel und anschließend mit diesem die eigentliche Nachricht.

Ergibt sich die Frage, wie aus einem beliebigen Nachrichtentext ein möglichst zufälliger Schlüssel generiert werden kann.

Signaturen und Hash-Funktionen

Algorithmen zur Erzeugung von Hash-Werten

Es besteht, wie oben dargestellt, der Wunsch, aus einem Nachrichtentext eine Zeichenfolge zu errechnen, die einerseits deutlich kürzer und andererseits so sein soll, dass sie diesen Text möglichst eindeutig charakterisiert. Dieser sogenannte Hash-Wert soll quasi wie ein Fingerabdruck der Originaldaten funktionieren. Zu diesem Zweck werden sogenannte Hash-Funktionen verwendet. Der Begriff Hash ist vom englischen to hash (zerhacken) abgeleitet.

Eine gute Hash-Funktion sollte folgende Kriterien erfüllen:

- Datenreduktion – Der Speicherbedarf des Hash-Wertes soll deutlich kleiner sein als jener der Nachricht.
- Chaotisch – Ähnliche Quellelemente sollen zu völlig verschiedenen Hash-Werten führen. Im Idealfall verändert das Umkippen eines Bits in der Eingabe durchschnittlich die Hälfte aller Bits im resultierenden Hash-Wert.
- Effizienz – Die Funktion muss schnell berechenbar sein, ohne großen Speicherverbrauch auskommen und sollte die Quelldaten möglichst nur einmal lesen müssen.
- Einwegfunktion – aus einem gegebenen Hash-Wert soll es praktisch unmöglich sein, die ursprünglichen Originaldaten zu ermitteln.
- Kollisionsresistenz – es soll praktisch unmöglich sein, zwei unterschiedliche Quelldaten zu finden, die einen identischen Hash-Wert haben.

Heute werden verschiedene Hash-Funktionen in der Kryptografie verwendet.

Message-Digest Algorithmus

Der Message-Digest Algorithmus wurde Ende der 80-er Jahre von R. L. Rivest entwickelt und in den 90-er Jahren weiterentwickelt. Die erste Version MD2 wurde für 8-Bit Rechnerarchitekturen entwickelt und spielt heute faktisch keine Rolle mehr. Der Nachfolger MD4 ist für 32-Bit Rechnerarchitekturen entwickelt, erwies sich aber als angreifbar. Heute in der Verwendung verbreitet ist der MD5-Algorithmus, der 1991 als sicherer Ersatz von MD4 entwickelt wurde.

In jüngster Zeit sind auch hier Schwächen bezüglich der Kollisionsresistenz gefunden worden, weshalb der Einsatz von MD5 nicht mehr empfohlen wird. Diese Unsicherheiten beziehen sich jedoch nur auf Kollisionsangriffe, weshalb ein bestehendes, auf MD5 basierendes Zertifikat nach wie vor nicht gefälscht werden kann.

Alle MD-Verfahren erzeugen aus beliebig langen Zeichenfolgen eine 128 Bit lange Zeichenfolge, die üblicherweise als 32-stellige Hexadezimalzahl dargestellt wird.

Beispiel

Der MD5-Wert des Textes "Franz jagt im komplett verwahrlosten Taxi quer durch Bayern" ist: a3cca2b2aa1e3b5b3b5aad99a8529074

Eine kleine Änderung des Textes (in diesem Fall ein Buchstabe im Namen) erzeugt mit sehr großer Wahrscheinlichkeit einen komplett anderen Hash-Wert:

Der MD5-Wert des Textes "Frank jagt im komplett verwahrlosten Taxi quer durch Bayern" ist: 7e716d0e702df0505fc72e2b89467910

Secure Hash Algorithmus

Hierunter wird eine Reihe standardisierter Hash-Algorithmen verstanden, die von der NIST gemeinsam mit der NSA entwickelt und 1994 veröffentlicht wurden. Bei diesem Algorithmus ist die Länge des Quelltextes begrenzt auf 2^{64} Zeichen, was aber praktisch nicht relevant ist. In verschiedenen Varianten werden daraus unterschiedlich lange Hash-Werte (160 Bit bei SHA1 bis 512 Bit bei SHA512) erzeugt.

Obwohl 2006 mögliche Angriffe gegen SHA veröffentlich wurden, gilt SHA als sehr sicher. Trotzdem wird empfohlen, diesen Hash-Algorithmus nicht mehr in neu zu entwickelnde Anwendungsprogramme zu implementieren.

RIPEMD-160

RIPEMD-160 (RACE Integrity Primitives Evaluation Message Digest) ist eine kryptographische Hash-Funktion mit einer Ausgabe von 160 Bit und wurde von H. Dobbertin, A. Bosselaers und B. Preneel in Europa entwickelt und 1996 erstmals publiziert.

Da die Entwicklung von RIPEMD-160 offener war als die von SHA1, ist es wahrscheinlicher, dass dieser Algorithmus weniger Sicherheitslücken aufweist. Da er jedoch weniger populär ist, haben weniger Kryptologen versucht, Schwächen zu finden, was wiederum die Wahrscheinlichkeit für unentdeckte Sicherheitslücken steigen lässt.

Tiger und Tiger2

Tiger ist eine kryptografische Hash-Funktion die von R. Anderson und E. Biham im Jahr 1996 entwickelt wurde. Der von Tiger erzeugte Hash-Wert hat eine Länge von 192 Bit. Der Tiger-Algorithmus ist nicht patentiert. Die 192-Bit (24-byte) langen Tiger-Hashes werden normalerweise als 48-stellige Hexadezimalzahl notiert.

Beispiel

Der Tiger-Wert des Textes "Franz jagt im komplett verwahrlosten Taxi quer durch Bayern" ist 4df42db66c8d84269d4b7157b92a87be717aa1a5834a3050

Der Tiger2-Wert des Textes "Franz jagt im komplett verwahrlosten Taxi quer durch Bayern" ist ac228a08cc97a449d85729e6549dbe4cd746df0061522b2c

Eine kleine Änderung des Textes (in diesem Fall ein Buchstabe im Namen) erzeugt mit sehr großer Wahrscheinlichkeit einen komplett anderen Hash-Wert:

Der Tiger-Wert des Textes "Frank jagt im komplett verwahrlosten Taxi quer durch Bayern" ist 9cee0eb7b596ba0f435d42c33ddf8eff7fabb86922aa4bc6

Der Tiger2-Wert des Textes "Frank jagt im komplett verwahrlosten Taxi quer durch Bayern" ist 5959793d7837abf2cc44dc57b3712c6da5d89cc1df92cd5a

Whirlpool

Whirlpool ist eine kryptologische Hash-Funktion, die von V. Rijmen und P. Barreto entworfen wurde. Sie wurde nach der Whirlpool-Galaxie im Sternbild der Jagdhunde benannt. Whirlpool funktioniert mit Dateien bis zu 2^{256} Bit Größe und gibt einen Hash-Wert von 512 Bit aus. Bislang sind keine Schwächen des Algorithmus bekannt, was allerdings relativiert werden muss, da der Algorithmus noch sehr jung ist und bislang wenig untersucht wurde.

Whirlpool steht unter GNU-License und darf damit kostenlos zu jedem Zweck verwendet werden. Es gehört zu den vom Projekt NESSIE (New European Schemes for Signatures, Integrity and Encryption) empfohlenen kryptografischen Algorithmen und wurde von der ISO standardisiert.

digitale Signaturen

Eine digitale Signatur ist ein kryptografisches Verfahren, bei dem zu einer Nachricht eine Zahl (die digitale Signatur) berechnet wird, deren Urheberschaft und Zugehörigkeit zur Nachricht durch jeden geprüft werden können. Digitale Signaturen basieren auf asymmetrischen Kryptosystemen (siehe Seite 10). Im juristischen Sinne lassen sich mit Hilfe digitaler Signaturen sichere elektronische Signaturen erzeugen, deren juristische Rahmenbedingungen im Signaturgesetz bestimmt sind. In der Alltagssprache dürfen daher die Begriffe der digitalen und der elektronischen Signatur keinesfalls synonym gebraucht werden.

Technisch wird aus den zu signierenden Daten und dem privaten Schlüssel durch eine eindeutige Rechenvorschrift eine Signatur berechnet. Dabei muss gewährleistet sein, dass einerseits andere Daten als auch andererseits ein anderer Schlüssel mit an Sicherheit grenzender Wahrscheinlichkeit eine andere Signatur ergeben. Ziel ist es dabei, dass es praktisch unmöglich sein soll, eine Signatur zu fälschen oder eine zweite Nachricht zu erzeugen, für welche diese Signatur ebenfalls gültig ist.

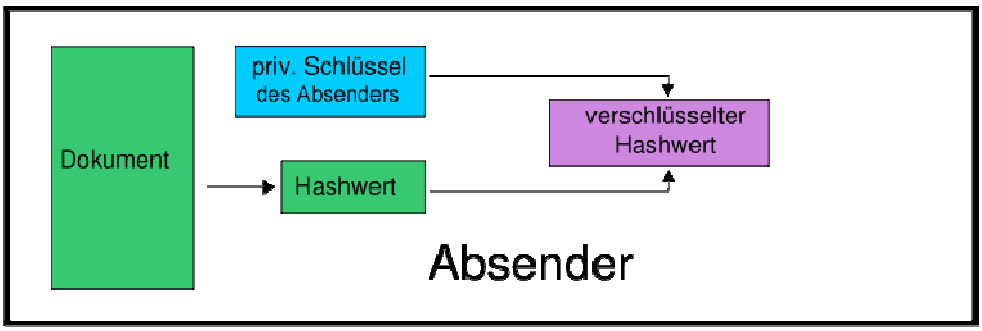
Auf Grund der Funktionsweise asymmetrischer Schlüssel darf deshalb nie eine Nachricht als Klartext signiert werden, sondern muss immer für den Hash-Wert einer Nachricht die Signatur erstellt werden.

Wird dieses Prinzip nicht beachtet, so ist es prinzipiell möglich, eine andere Nachricht zu konstruieren, die eine gültige Signatur hat ohne den privaten Schlüssel zu kennen. Deshalb gehen in die Bewertung der Sicherheit einer Signatur immer zwei Parameter ein: Die Sicherheit des Schlüssels und die Sicherheit, vor allem die Kollisionsresistenz, des verwendeten Hash-Algorithmus.

Das heute am meisten verbreitete Verfahren zum Signieren von Nachrichten ist das RSA-Verfahren. Daneben findet auch das El-Gamal Verfahren verbreitet Anwendung. In jedem Fall müssen ein Hash-Algorithmus mit einem Verfahren zur Erzeugung eines asymmetrischen Schlüsselpaares kombiniert werden.

Ausführung

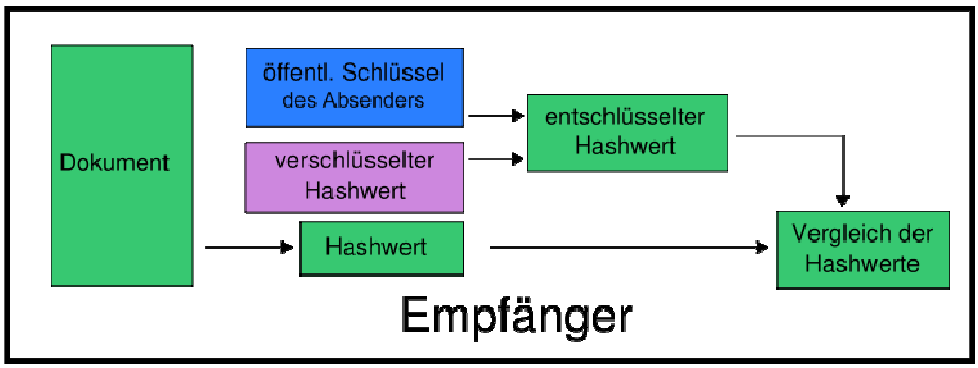
Erzeugen der Signatur einer Nachricht



Aus der zu übermittelnden Nachricht wird zuerst ein Hash-Wert berechnet. Dieser wird anschließend mit Hilfe des privaten Schlüssels des Absenders verschlüsselt und bildet damit die Signatur. Diese wird der Nachricht angehängt und mit dieser versendet.

Ausführung

Überprüfen der Signatur einer Nachricht



Aus der übermittelten Nachricht erzeugt der Empfänger ebenfalls einen Hash-Wert. Parallel wird mit Hilfe des öffentlichen Schlüssels des Absenders aus der Signatur der Hash-Wert berechnet. Der Vergleich beider Hashwerte erlaubt eine Aussage über die Integrität des Absenders und des Dokumentes.

Hybride Chiffrierverfahren

Aus der Bewertung der Eigenschaften symmetrischer und asymmetrischer Chiffrierverfahren (Seite 13) ergibt sich, dass insbesondere für die Verschlüsselung des E-Mail-Verkehrs beide Verfahren entscheidende Vor- und Nachteile haben. So sind symmetrische Verfahren schnell, haben aber das Problem des sicheren Schlüsselaustausches, hingegen ermöglichen asymmetrische Verfahren einen sicheren Schlüsselaustausch, sind aber sehr langsam.

An dieser Stelle eröffnet sich die Idee, beide Verfahren zu kombinieren: Der Text (und die Anlagen) einer E-Mail-Nachricht werden mit Hilfe symmetrischer Verfahren verschlüsselt und mit Hilfe asymmetrischer Verfahren erfolgt die Signatur und der Austausch des verwendeten Schlüssels. Ein solches Verfahren wird als hybrides Chiffrierverfahren bezeichnet.

Eine für diese Zwecke geeignete Software ist PGP (pretty good privacy) bzw. das daraus hervorgegangene GnuPG (GNU Privacy Guard). Letzteres ist ein freies Kryptographiesystem, d. h. es dient zum Ver- und Entschlüsseln von Daten sowie zum Erzeugen und Prüfen elektronischer Signaturen. Das Programm implementiert den OpenPGP-Standard nach RFC 4880 und wurde als Ersatz für PGP entwickelt. GnuPG benutzt standardmäßig nur patentfreie Algorithmen und wird unter der GNU-GPL vertrieben. Es kann unter Linux, Mac OS X und diversen anderen Unix-Varianten sowie unter Microsoft Windows betrieben werden.

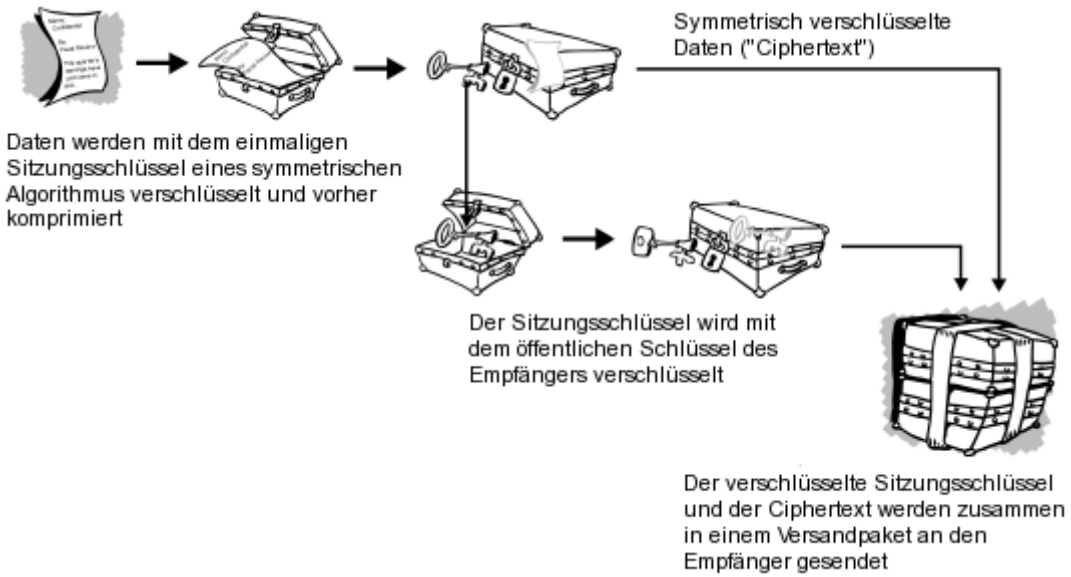
Die Arbeitsweise von GnuPG

Verschlüsselung von Daten

Bei der Public-Key Verschlüsselung kommt auf Seiten des Absenders der öffentliche RSA oder ElGamal Schlüssel des Empfängers und ein symmetrischer Sitzungsschlüssel zum Einsatz, der bei jeder Verschlüsselungsaktion zufällig generiert wird. Die Abläufe sind nachfolgend dargestellt:

Ausführung

Verschlüsselung mit GnuPG



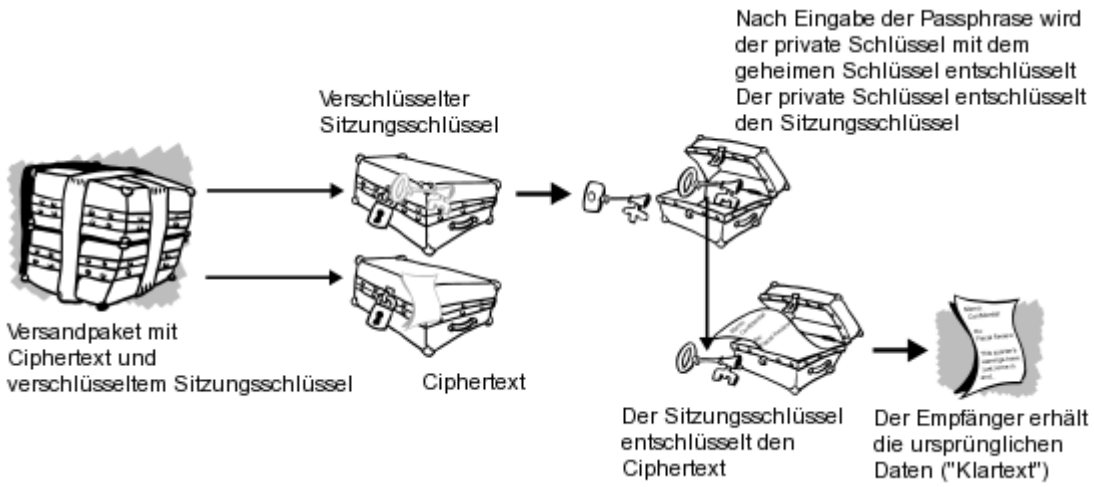
1. Der Klartext wird komprimiert und mit einem für jede Nachricht neuen, zufällig generierten und einzigartigen Sitzungsschlüssel vermittels eines symmetrischen Algorithmus verschlüsselt. Das Ergebnis ist der Geheimtext (Ciphertext).
2. Der Sitzungsschlüssel wird mit dem öffentlichen Schlüssel des Empfängers vermittels eines asymmetrischen Algorithmus verschlüsselt.
3. Der Geheimtext wird zusammen mit dem verschlüsselten Sitzungsschlüssel in eine ASCII Transporthülle „verpackt“ und das Gesamtpaket dem Empfänger zugestellt.

Entschlüsselung von Daten

Auf Seiten des Empfängers kommen zur Entschlüsselung der Daten der private Schlüssel des Empfängers, der nur ihm vorliegt und, damit dieser verwendet werden kann, eine Passphrase sowie der mit der Nachricht übermittelte symmetrische Sitzungsschlüssel zum Einsatz.

Ausführung

Entschlüsselung mit GnuPG



1. Der Empfänger extrahiert den Geheimtext und den verschlüsselten Sitzungsschlüssel aus der empfangenen Nachricht.
2. Anschließend entschlüsselt er mit seinem geheimen Schlüssel, der nach Bestätigung durch eine Passphrase verwendet werden kann, vermittels eines asymmetrischen Algorithmus den Sitzungsschlüssel.
3. Mit diesem Sitzungsschlüssel wird der Klartext aus dem Geheimtext vermittels eines symmetrischen Verfahrens entschlüsselt.

Diese Abläufe hören sich sehr kompliziert an, laufen aber in der Praxis im Hintergrund und für den Benutzer – zumindest bei einer gewöhnlichen E-Mail – ohne merkliche Verzögerung ab.

Schlüsselverwaltung von GnuPG

Während bei einer symmetrischen Verschlüsselung der sichere Kanal zum Austausch des Schlüssels eine unabdingbare Voraussetzung ist, müssen sich bei der asymmetrischen Public-Key Verschlüsselung beide Kommunikationspartner absolut sicher sein, dass sie jeweils einen echten öffentlichen Schlüssel des anderen verwenden. Dazu müssen sie die Zugehörigkeit des Schlüssels zum jeweiligen Kommunikationspartner eindeutig, d. h. möglichst über einen sicheren Kanal überprüfen können. Prinzipiell sind Fälschungen von veröffentlichen Schlüsseln denkbar. Deshalb gibt es zwei Möglichkeiten, die tatsächliche Zuordnung eines Schlüssels zu einer Person oder Instanz zu ermöglichen.

Zum Einen sind das sogenannte Zertifizierungsstellen (Certification Authorities). Diese müssen in einem gesetzlich geregelten Verfahren glaubhaft darlegen, wie die auf dem Schlüsselserver gespeicherten Daten vor fremdem Zugriff geschützt werden und wie der Nachweis der Echtheit eines Schlüssels erfolgt. Inwiefern jemand einer Zertifizierungsstelle wirklich vertraut, ist dabei letztlich seine individuelle Entscheidung.

Zum Anderen besteht die Möglichkeit, ein Netz von Vertrauensbeziehungen aufzubauen, wo die Inhaber von Schlüsseln sich gegenseitig nach entsprechender Prüfung bestätigen, dass die veröffentlichten Schlüssel wirklich echt und ihnen zuordenbar sind. Ein solches Netz, ein sogenanntes Web of Trust, also ein Netz von Vertrauensbeziehungen wird von GnuPG genutzt. Es bietet die Möglichkeit, die Echtheit eines öffentlichen Schlüssels zu überprüfen und dessen Fälschung oder Manipulation zu erkennen.

Zu diesem Zweck kann jemand den öffentlichen Schlüssel einer anderen Person, dessen Korrektheit von ihm geprüft wurde, zertifizieren und mit dem vergebenen Zertifikat erneut veröffentlichen. Machen das mehrere bekannte Personen gegenseitig, entsteht tatsächlich ein Netz von (Vertrauens-) Zertifikaten, welches eine hinreichende Sicherheit für einen sicheren Austausch von (verschlüsselten) Nachrichten bietet.

Verwendung von GnuPG mit Thunderbird

Der GNU Privacy Guard, ist „lediglich" ein eigenständiges Kommandozeilenprogramm, das auf der Konsolenebene ausgeführt und über Kommandos, Argumente und Optionen, die in der Konsole eingetippt werden, manuell bedient wird. Diese Verfahrensweise ist letztlich für eine alltägliche Nutzung nicht unbedingt ratsam. Deshalb gibt es verschiedene grafische Benutzeroberflächen, die je nach Betriebssystem mit installiert werden müssen.

Da der praktische Einsatz vor allem im Rahmen der E-Mail-Kommunikation erfolgt, ist es sinnvoll, diese grafische Benutzeroberfläche gleich in den E-Mail-Client einzubinden. Thunderbird bietet dafür mit dem AddOn Enigmail sehr gute Voraussetzungen. Zum Einen ist Thunderbird uneingeschränkt kompatibel zur Verwendung mit verschlüsselten E-Mails und zum Anderen bietet Enigmail alle erforderlichen Schnittstellen, um GnuPG effektiv verwenden zu können. Im Folgenden soll deshalb auf die praktische Arbeit mit dieser Software eingegangen werden.

Installation

Um verschlüsselte bzw. signierte E-Mails unter Thunderbird nutzen zu können, müssen zwei Softwarekomponenten installiert werden:

1. Die Software GnuPG, download über die Website http://gnupg.org/index.de.html (deutschsprachig) und

2. Die Software Enigmail als Add-on von Thunderbird, download über die Website http://www.erweiterungen.de/detail/Enigmail/, ebenfalls deutschsprachig.

Nach dem Download der Programme muss zuerst GnuPG installiert werden. Die Installation sollte reibungslos erfolgen. Empfohlen wird, die vorgeschlagenen Installationspfade nicht anzupassen, damit die Zusammenarbeit mit Enigmail später reibungslos funktioniert.

Die Installation von Enigmail erfolgt wie alle Add-ons über den Menüpunkt Extras – Add-ons in Thunderbird. Nach der Installation befindet sich in Thunderbird das neue Menü OpenPGP sowie eine Schaltfläche „Entschlüsseln". Hier wird deutlich, dass Enigmail letztlich ein Frontend, also die grafische Benutzeroberfläche ist, um OpenPGP zu bedienen.

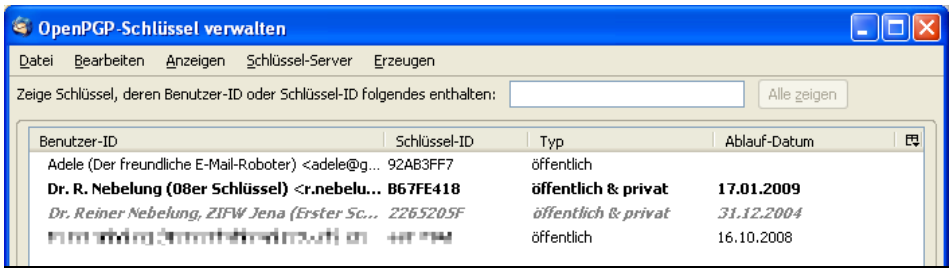
Schlüsselverwaltung

Nach der Installation sind zum Start der Arbeit mit Enigmail zwei Schritte erforderlich: Zum Einen muss ein Schlüsselpaar für die eigene Verwendung generiert werden und zum Anderen muss der E-Mail-Account in Thunderbird so kofiguriert werden, dass die Verwendung von verschlüsselten bzw. signierten E-Mails unkompliziert möglich ist.

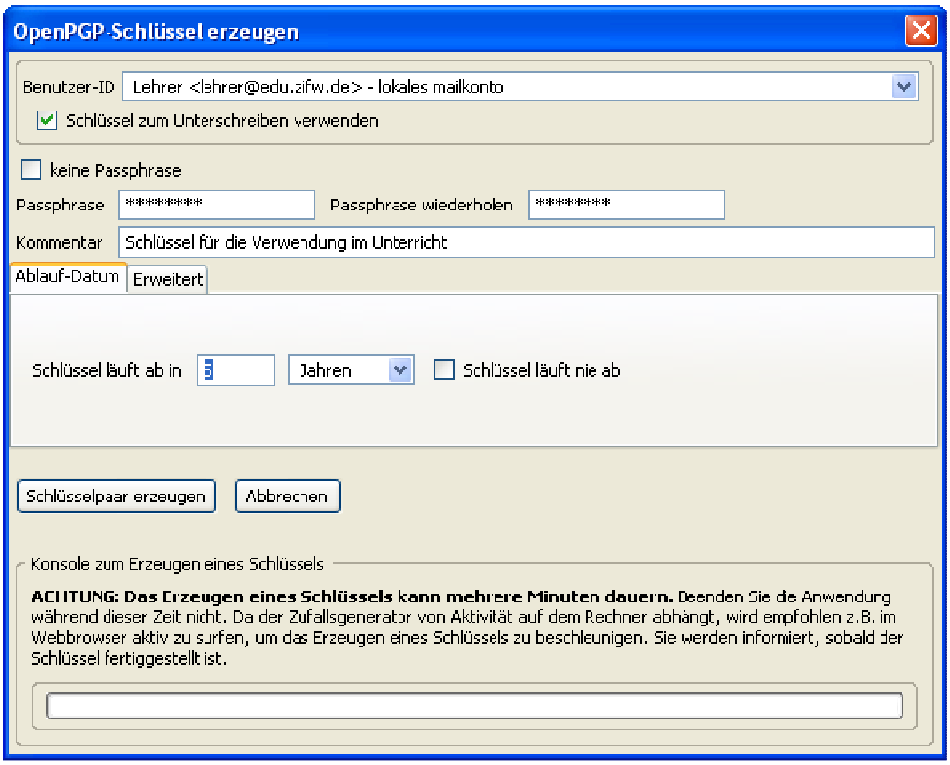
Generierung eines (neuen) Schlüsselpaares

Ausführung

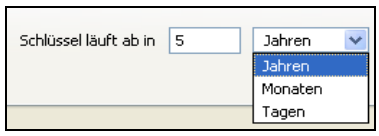
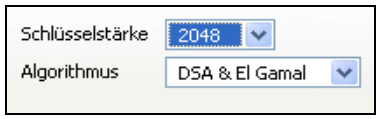
Dazu wird im Menü „OpenPGP“ von Thunderbird der Eintrag „Schlüssel verwalten“ ausgewählt. Dabei wird das nachfolgende Dialogfenster eingeblendet:



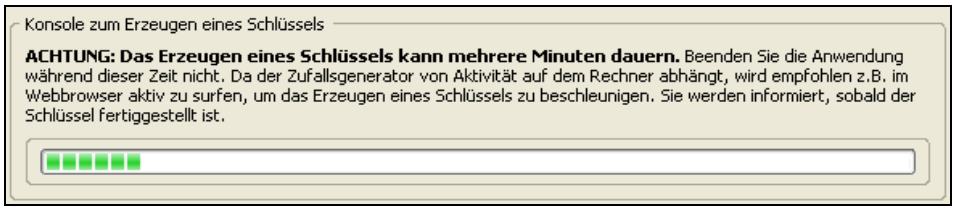
Diese Arbeitsumgebung ist quasi die „Verwaltungszentrale“ für alle verwendeten Schlüssel. Für den Fall, dass sehr viele Schlüssel verwaltet werden müssen, befindet sich im oberen Teil des Fensters eine Filtermöglichkeit. Das Menü dieser Arbeitsumgebung bietet alle Verwaltungmöglichkeiten – für die Erstellung eines neuen Schlüssels wird der Menüpunkt Erzeugen > Neues Schlüsselpaar gewählt und anschließend folgender Dialog eingeblendet:



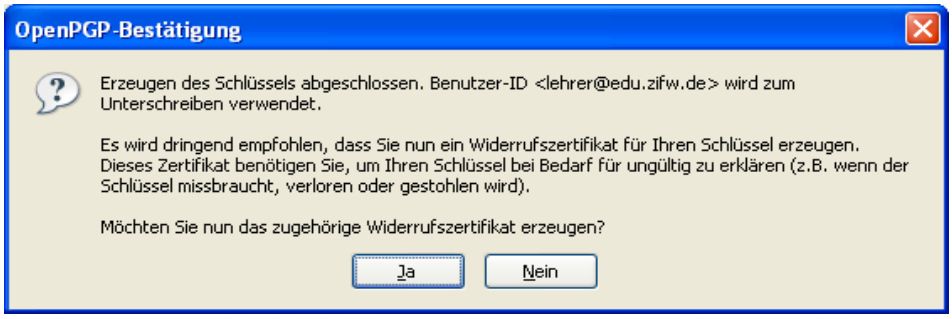
Dabei haben die Optionen folgende Bedeutung:

Benutzer-ID	Hier wird das E-Mail-Konto ausgewählt, für welches das Schlüsselpaar erzeugt werden soll. Ein neues Schlüsselpaar muss zwingend an ein E-Mail-Konto gebunden sein, da dieses letztlich die Identität darstellt.
Schlüssel zum Unterschreiben verwenden	Wird diese Option aktiviert (Standard), wird automatisch das zu erzeugende Schlüsselpaar von Thunderbird dem jeweiligen Konto zugeordnet – es erspart also ein paar Mausklicke, wenn es der erste Schlüssel ist, der einem Konto zugeordnet wird.
Passphrase	Diese Passphrase wird automatisch abgefragt, wenn der Schlüssel zum Entschlüsseln oder Signieren einer Nachricht verwendet wird.
Kommentar	Wird später in die Schlüsselsignatur mit übernommen und sollte deshalb mit Bedacht gewählt werden, da er auch veröffentlicht wird.
Ablaufdatum	 Nach Bedarf in Jahren, Monaten, Tagen wählbar. Für unbegrenzte Gültigkeit wird die entsprechende Option aktiviert.
Erweitert	 Hier sind der Algorithmus und die Schlüsselstärke wählbar. Es wird empfohlen, die Vorgaben zu verwenden.

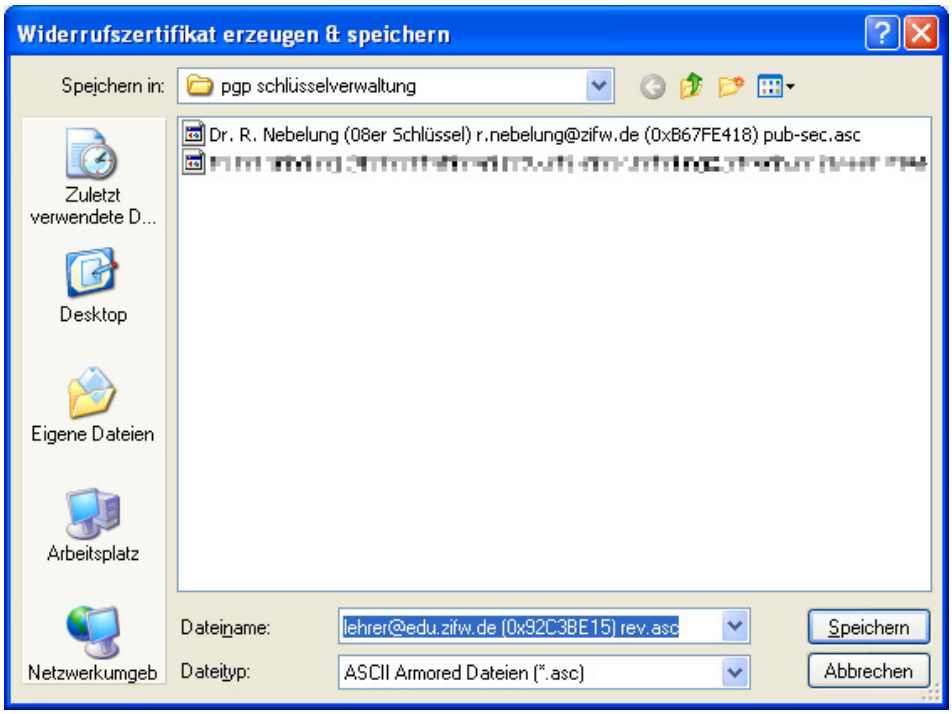
Wurden alle Einstellungen vorgenommen, kann die Erzeugung des Schlüsselpaares gestartet werden. Die Erzeugung nimmt auch bei leistungsfähigen Rechnern merkliche Zeit in Anspruch – also bitte nicht die Geduld verlieren. Im Gegenteil: Da während der Schlüsselerzeugung viele Zufallsprozesse benötigt werden, ist es nicht ratsam, den Rechner einfach arbeiten zu lassen und zu warten. Vielmehr sollte der Rechner mit anderen Dingen „beschäftigt“ werden, dem Abrufen von Webseiten etwa, oder mit Spielen.



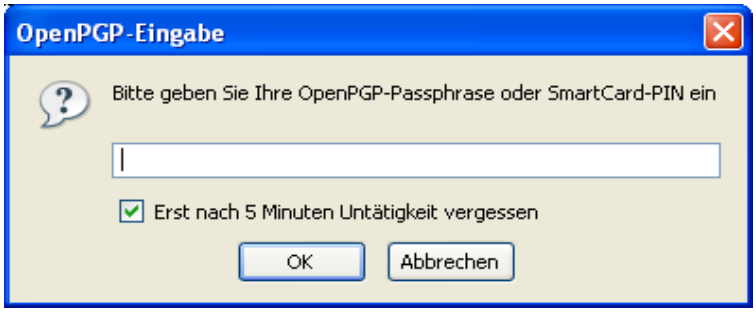
Die dargestellte Fortschrittanzeige informiert über den Stand der Dinge. Nach Abschluss der Erzeugung wird folgende Mitteilung eingeblendet:



Das Widerrufszertifikat ist tatsächlich dringend zu empfehlen: Nur auf diese Art habe ich die Möglichkeit, einen veröffentlichen Schlüssel vor Ablauf seiner Gültigkeit zurückzuziehen. Wird dem Vorschlag entsprochen, erfolgt der übliche Dialog zur Auswahl des Speicherortes einer Datei.



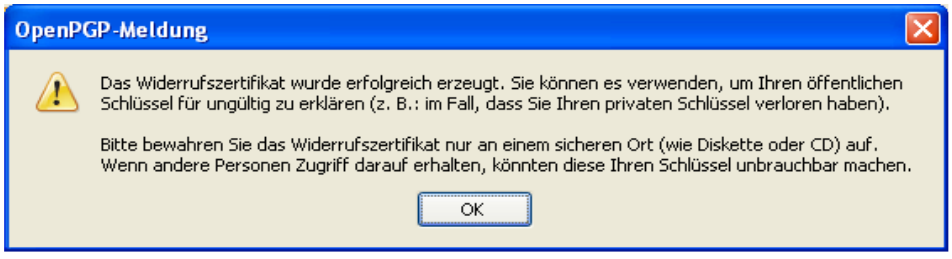
Sobald die Speichern-Schaltfläche angeklickt wird, schlägt OpenPGP das erste mal Alarm: Da das Widerrufszertifikat im ASCII-Format gespeichert wird, muss dieses aus dem Schlüssel extrahiert werden, wozu die Passphrase das erste Mal benötigt wird.



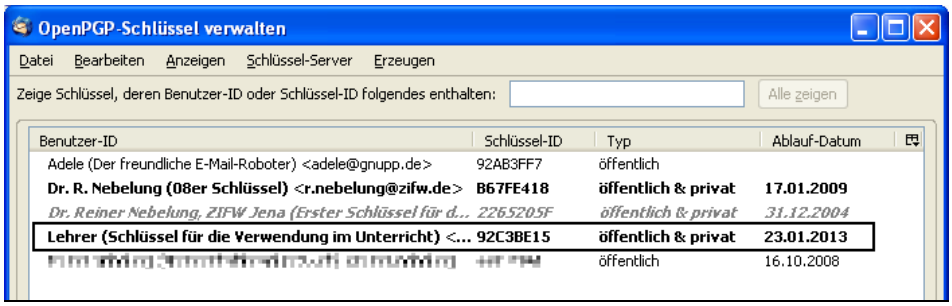
Hinweis

Interessant ist die Funktion des sogenannten Passphrasen-Caches: Die Passphrase bleibt eine voreinstellbare Zeit, üblicherweise 5 Minuten aktiv, damit diese nicht immer wieder abgefragt werden muss. Dies schont im Alltag deutlich die Nerven...

Nun erscheint die Abschlussmeldung für das Widerrufszertifikat.



Im Ergebnis ist das neu erzeugte Schlüsselpaar in der Verwaltungsliste aufgeführt:

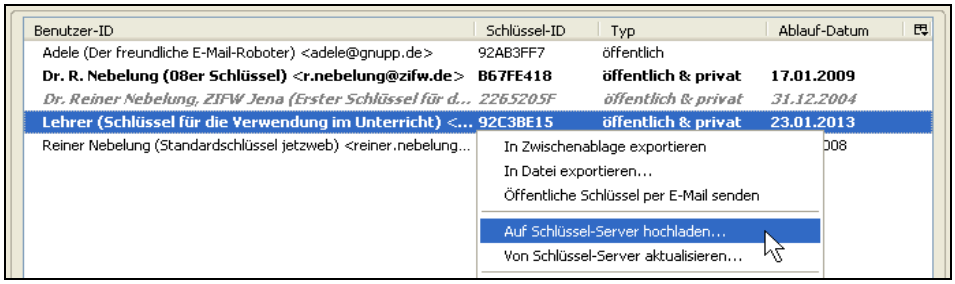


Veröffentlichung des Schlüssels

Nachdem das Schlüsselpaar generiert wurde, kann es nun praktisch verwendet werden. Die Signierung einer Nachricht würde in diesem Fall beim Absender bereits problemlos funktionieren, aber: Wie soll der Empfänger die Signatur prüfen können, wenn dieser den öffentlichen Schlüssel nicht kennt? Der öffentliche Schlüssel muss also verteilt werden. Dafür gibt es grundsätzlich zwei Möglichkeiten: Zum Einen kann der öffentliche Schlüssel auf einem dafür vorgesehenen Schlüsselserver veröffentlicht werden – daraus würde sich dann das Web of Trust entwickeln, was weiter oben im Abschnitt der Schlüsselverwaltung (auf Seite 19) bereits erläutert wurde. Die zweite Möglichkeit besteht darin, den Kommunikationspartnern den öffentlichen Schlüssel direkt per E-Mail zu übermitteln.

Ausführung

Zur Veröffentlichung eines Schlüssels auf einem Schlüsselserver wird für den gewünschten Schlüssel am einfachsten das Kontextmenü aufgerufen, in welchem alle erforderlichen Aktionen wählbar sind – in diesem Fall der Menüpunkt „Auf Schlüsselserver hochladen“:



Im Ergebnis wird der folgende Dialog eingeblendet:

Hier wird der gewünschte Schlüsselserver gewählt und die Aktion ausgeführt.
Die Auswahl und die Reihenfolge der zu verwendenden Schlüsselserver werden in den Grundeinstellungen von Enigmail vorgenommen.

Die zweite Möglichkeit besteht darin, den öffentlichen Schlüssel per E-Mail an die gewünschten Empfänger zu übermitteln. Wie auch immer – erst danach ist es den Empfängern einer signierten E-Mail möglich, die Signatur zu überprüfen bzw. dem Absender eine verschlüsselte E-Mail zu senden, denn dafür müssen diese über den öffentlichen Schlüssel verfügen.

Aufnahme eines öffentlichen Schlüssels in das Schlüsselbund

Wird eine signierte E-Mail empfangen, deren öffentlicher Schlüssel noch nicht vorliegt, so zeigt sich in Thunderbird folgende Ansicht:

Im Header der E-Mail meldet OpenPGP, dass die E-Mail unterschrieben ist, aber die Unterschrift nicht geprüft wurde. Gleichzeitig wird die Aufforderung dargestellt, das Brief-Symbol mit dem Fragezeichen rechts im Header bzw. ein Stift-Symbol unten rechts in der Statuszeile von Thunderbird anzuklicken, welches in dieser Abbildung nicht sichtbar ist.

Im Text der E-Mail sind alle Informationen gut erkennbar:

- Die Angabe, welche Hash-Funktion verwendet wurde, in diesem Fall SHA1,
- der eigentliche Text (anbei der ...),
- die Angabe, womit die Signatur erzeugt wurde und
- die eigentliche Signatur

In diesem (besonderen) Fall wurde der Schlüssel der E-Mail als Anlage beigefügt.

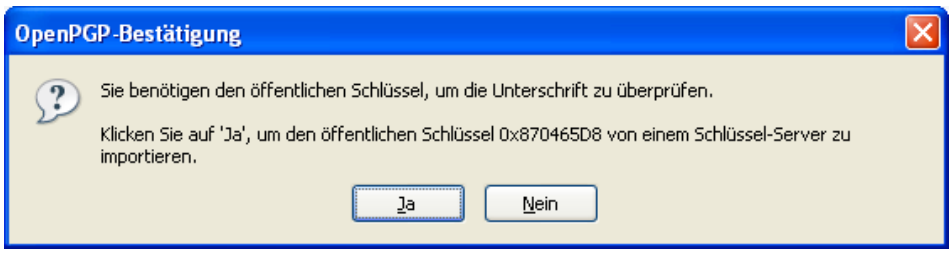
Dieser ist als Anlage in Form einer ASCII-Datei erkennbar, deren Dateiname die Schlüssel-ID und deren Inhalt der Schlüssel selbst ist.

Im Editor kann der Schlüssel, genauer der öffentliche Teil des asymmetrischen Schlüsselpaares im ASCII-Format angezeigt werden, wie nebenstehend dargestellt.

Damit arbeiten kann aber wohl doch nur ein Computer...

Um die Signaturprüfung auszuführen wird der öffentliche Schlüssel des Absenders benötigt. Um diesen zu „erwerben“ und dem eigenen Schlüsselbund hinzuzufügen, gibt es zwei Möglichkeiten: Zum Einen kann der Schlüssel von einem der Schlüsselserver heruntergeladen werden, vorausgesetzt, der Absender hat diesen dort veröffentlicht. Die zweite Möglichkeit besteht darin, den Schlüssel, falls dieser der E-Mail angehängt wurde, direkt aus der Anlage in den eigenen Schlüsselbund aufzunehmen und zu verwenden.

Die Vorgehensweise im ersten Fall ist denkbar einfach: Wie im Header von OpenPGP aufgefordert, wird das Symbol mit dem Briefumschlag bzw. das Stiftsymbol angeklickt und es erscheint folgende Meldung:

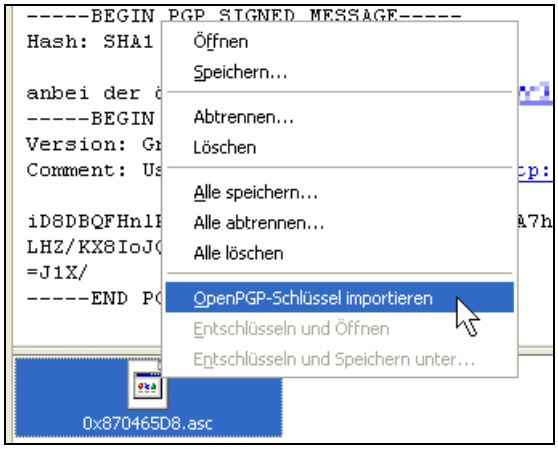


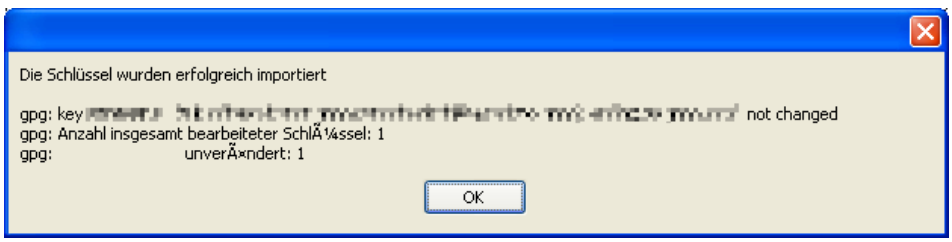
Wird diese Frage mit „Ja“ bestätigt, kommt sofort die nächste Rückfrage, welcher Schlüsselserver in diesem Fall verwendet werden soll:

Eine Bestätigung mit „OK“ versucht den Import.

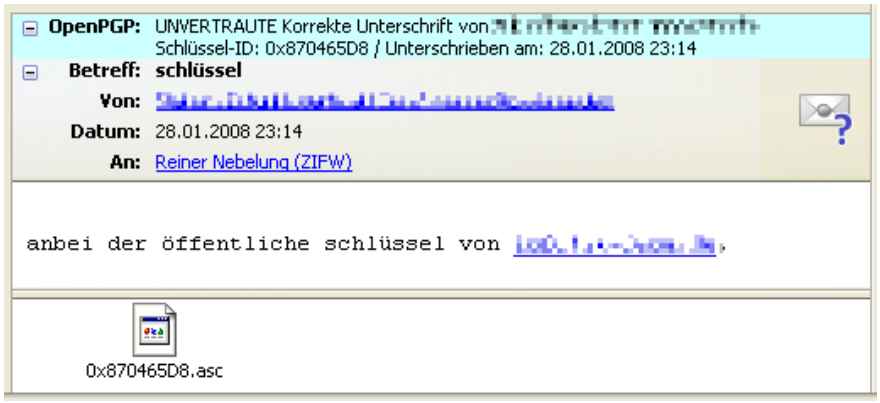
Im zweiten Fall, wenn der öffentliche Schlüssel der E-Mail angehängt wurde, ist die einfachste Variante, den Schlüssel in der Anlage direkt per Kontextmenü zu importieren.

In beiden Fällen, wird der Erfolg des Imports gemeldet.

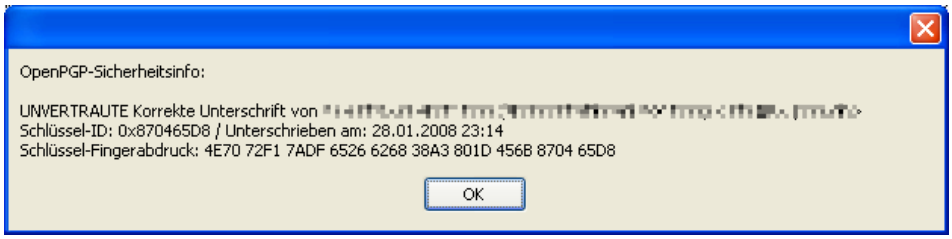




Eine Aktualisierung der Anzeige liefert die nachfolgende Darstellung in Thunderbird:



Ein Klick auf das rechts sichtbare Symbol mit dem Briefumschlag bzw. auf das hier nicht abgebildete Stift-Symbol in der Statuszeile von Thunderbird liefert folgende Meldung:



Vertrauensbeziehungen der Schlüssel festlegen

Schlüssel erzeugen und veröffentlichen ist die eine Seite – inwiefern einem veröffentlichten Schlüssel vertraut werden kann, eine andere. Auf Seite 19 wurde auf das Web of Trust als das Verfahren von OpenPGP zur Verifizierung von Schlüsseln bereits eingegangen. An dieser Stelle soll die praktische Verwendung des Web of Trust dargestellt werden.

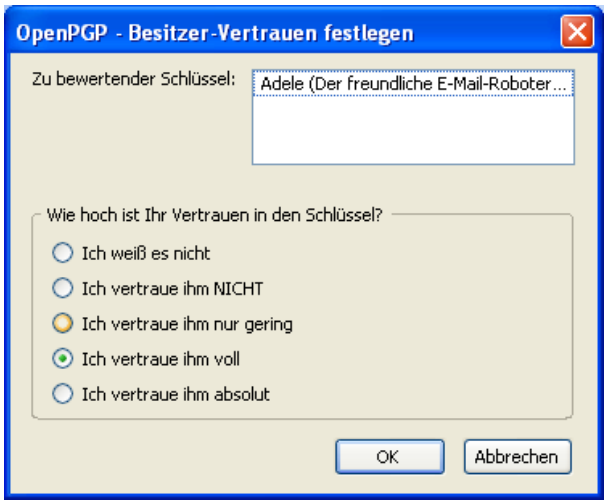
Grundsätzlich müssen zwei Arten des Vertrauens unterschieden werden:

1. Das Vertrauen, welches ich selbst – ausschließlich für den eigenen Gebrauch – einem Schlüssel und damit dem (vorgeblichen[?]) Besitzer eines Schlüssels entgegen bringe, das sogenannte Besitzervertrauen und
2. das Vertrauen in die Echtheit und Identität der Zuordnung eines Schlüssels, welches ich einem mir zugänglichen (fremden), durch E-Mail zugesandten oder von einem Schlüsselserver importierten öffentlichen Schlüssel entgegenbringe und durch meine Unterschrift mit einem eigenen Schlüssel öffentlich (im Web of Trust) durch mich bekundet wird.

Ersteres ist eine persönliche Wertung, die intern bleibt. Letzteres ist ebenfalls eine persönliche Bewertung, die jedoch veröffentlicht wird und letztlich im Rahmen einer Vernetzung, also eines sich wechselseitig auszusprechenden Vertrauens, das Web of Trust ergibt.

Ausführung

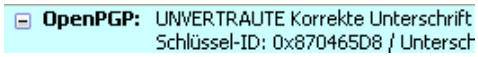
Zur Festlegung des Besitzervertrauens wird für den gewünschten Schlüssel in der Verwaltung das Kontextmenü aufgerufen und dort der Menüeintrag „Besitzer-Vertrauen festlegen …“ gewählt. Darauf wird das nachfolgend dargestellte Dialogfenster eingeblendet:



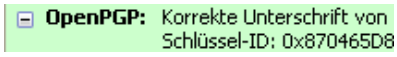
Die möglichen Stufen des Besitzervertrauens reichen von „kein Vertrauen“ bis „absolutes Vertrauen“ sowie der Option „weiß nicht“.

Die Festlegung hat nur interne Auswirkungen und dient eher der „Erinnerung“.

Die Meldung in der Kopfzeile einer entschlüsselten E-Mail beinhaltet die unterschiedlichen Statusvarianten des Besitzervertrauens.



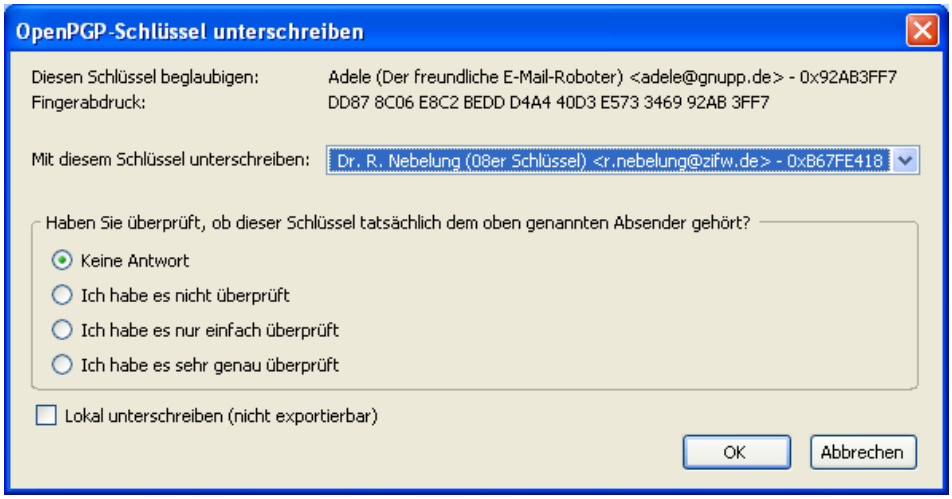
Kopfzeile einer E-Mail für die Besitzer-Vertrauens-Stadien „Ich weiß es nicht“ bis zu „Ich vertraue ihm voll“



Kopfzeile einer E-Mail für den Besitzer-Vertrauens-Status „Ich vertraue ihm absolut“

Ausführung

Um zum Web of Trust beizutragen, gibt es die Möglichkeit, einen Schlüssel zu unterschreiben. Dazu wird im Kontextmenü des entsprechenden Schlüssels der Menüeintrag „Unterschreiben ...“ ausgewählt, wobei das nachfolgend dargestellte Dialogfenster eingeblendet wird:



Hier wird ausgewählt, inwiefern die Überprüfung der Identität des Absenders und des ihm gehörenden Schlüssels geprüft wurde. Diese „Unterschrift“ im Sinne einer erfolgten Beglaubigung wird wieder auf den Schlüsselserver exportiert und dient anderen Nutzern, die diesen Schlüssel auch benötigen, als Anhaltspunkt für die Glaubwürdigkeit des Schlüssels.

Wird die Option „Lokal unterschreiben“ aktiviert, wird der eigene Schlüssel, der zum unterschreiben verwendet wurde, nicht mit exportiert, sondern nur die Tatsache, dass der Schlüssel unterschrieben wurde.

Hinweis

Schlüssel, die aktuell nicht verwendet werden sollen, können in der Schlüsselverwaltung deaktiviert und später bei Bedarf auch wieder aktiviert werden.

Wichtig

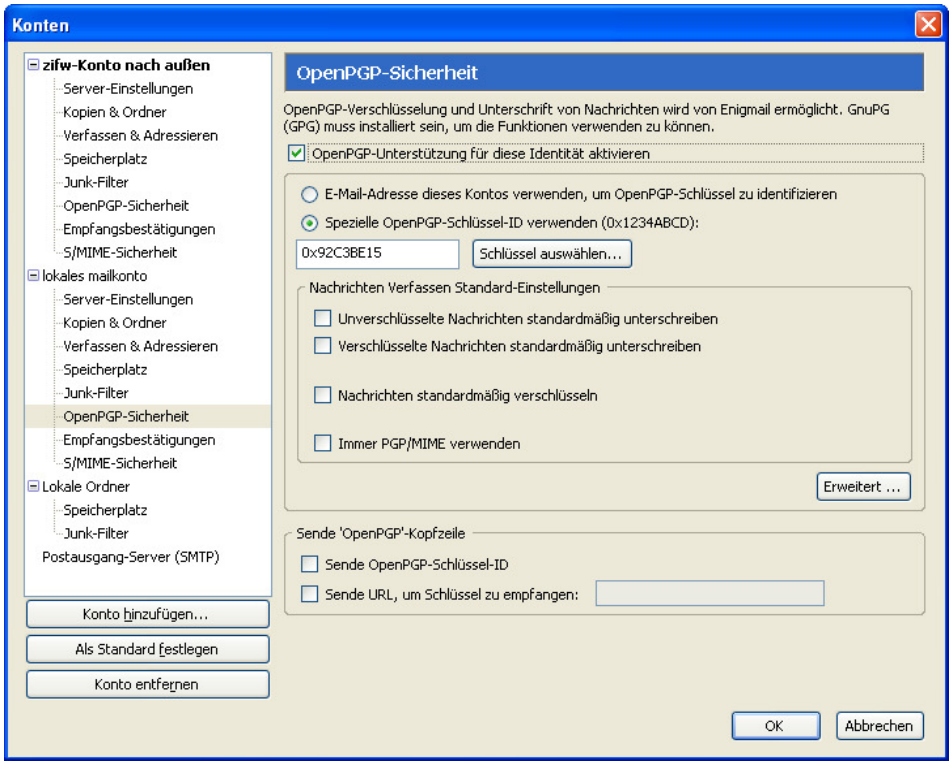
Schlüssel, deren Gültigkeit abgelaufen ist, werden von OpenGPG automatisch nicht mehr verwendet. Vor Ablauf der Gültigkeit sollte deshalb rechtzeitig ein neues Schlüsselpaar generiert und veröffentlicht werden.

E-Mail-Konto mit OpenPGP verwenden

Nachdem die Schlüsselverwaltung eingerichtet wurde, kann nun der praktische Betrieb mit OpenPGP beginnen. Dazu muss als nächstes in Thunderbird das betreffende E-Mail-Konto, für welches der Schlüssel erstellt wurde, angepasst werden.

Ausführung

Zur Anpassung des E-Mail-Kontos wird in Thunderbird die Kontenverwaltung unter Extras > Konten... aufgerufen. Für jedes registrierte Konto wurde mit der Installation von Enigmail ein Eintrag „OpenPGP-Sicherheit“ angelegt, der nun ausgewählt wird. Dabei wird folgender Dialog angezeigt:



Durch Auswahl der Option „OpenPGP-Unterstützung für diese Identität aktivieren“ wird die Ver- und Entschlüsselung sowie das Signieren von Nachrichten für das betreffende E-Mail-Konto grundsätzlich ermöglicht und können weitere Optionen festgelegt werden.

Diese betreffen folgende grundlegenden Einstellungen:

E-Mail-Adresse dieses Kontos verwenden ...	Der zu verwendende Schlüssel wird anhand der E-Mail-Adresse (automatisch) zugeordnet. Sollte es mehrere Schlüssel geben, die zu einer E-Mail-Adresse gehören, dann wird beim Senden von Nachrichten ein Auswahl-Dialog eingeblendet, um sich für einen der Schlüssel zu entscheiden.
Spezielle OpenPGP Schlüssel-ID verwenden	Bei Wahl dieser Option muss explizit ein ganz bestimmter OpenPGP-Schlüssel anhand der eindeutigen Schlüssel-ID ausgewählt werden. Somit entfällt beim Senden von Nachrichten in jedem Fall die Wahlmöglichkeit, da bereits hier der zu verwendende Schlüssel festgelegt wurde.
Nachrichten Verfassen Standard-Einstellungen	Hier kann eingestellt werden, ob ausgehende Nachrichten standardmäßig unterschrieben und/oder verschlüsselt werden sollen. Die Option Immer PGP/MIME verwenden sollte nur aktiviert werden, wenn sichergestellt ist, dass alle Empfänger mit denen OpenPGP-verschlüsselte Nachrichten ausgetauscht werden, das PGP/MIME-Format auch nutzen können. Diese Einstellungen werden, je nach Empfänger und eventuell vorhandenen Empfängerregeln, ignoriert. Über die Schaltfläche „Erweitert“ werden die Grundeinstellungen von Enigmail aufgerufen.
OpenPGP-Kopfzeilen	Mit den Optionen „Sende OpenPGP-Schlüssel-ID“ und „Sende URL, um Schlüssel zu empfangen“ wird es den E-Mail-Empfängern erleichtert, sich den (eigenen) öffentlichen Schlüssel zu besorgen.

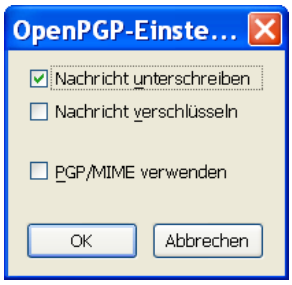
Damit sind alle nötigen Voraussetzungen erfüllt, um den verschlüsselten bzw. signierten E-Mail-Verkehr praktizieren zu können.

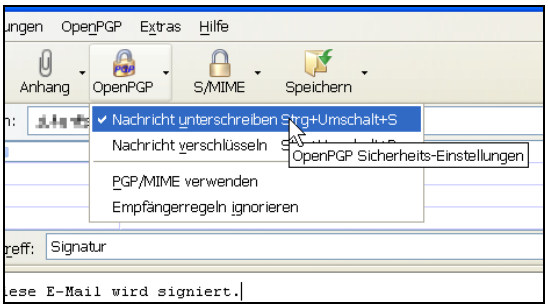
Signieren sowie Ver- und Entschlüsseln von Nachrichten

Praktischer Betrieb bedeutet, zu versendende E-Mails zu signieren und, falls der öffentliche Schlüssel eines Empfängers bekannt ist, zu verschlüsseln sowie verschlüsselte E-Mails an die eigene Adresse zu entschlüsseln.

signierte Nachrichten versenden

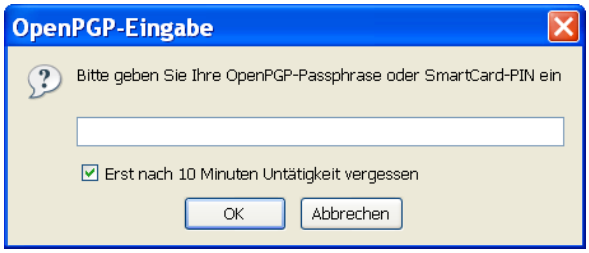
Die Nachricht wird zuerst wie gewohnt adressiert und geschrieben. Vor dem Absenden wird für die jeweilige Nachricht festgelegt, wie diese bezüglich Signatur bzw. Verschlüsselung behandelt werden soll. Dazu wird das entsprechende Symbol angeklickt bzw. im Menü OpenPGP die gewünschte Auswahl getroffen.





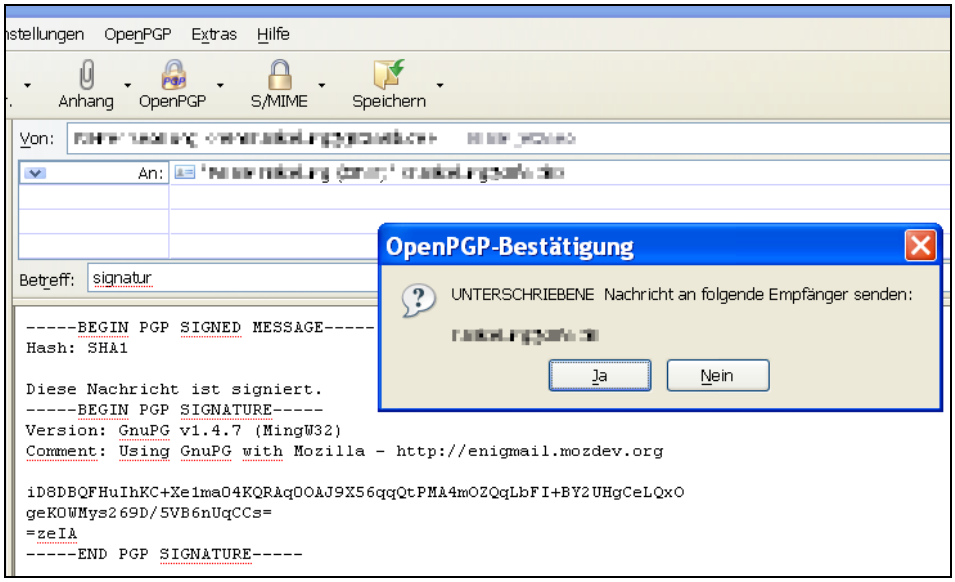
Das Anklicken der Schaltfläche liefert die links dargestellt Auswahlbox, das Öffnen der Drop-Down-Auswahl neben der Schaltfläche liefert die analoge Auswahl in Form des Menüs, welches rechts dargestellt ist.

Zum Signieren der E-Mail muss die erste Option – „Nachricht unterschreiben“ aktiviert werden. Beim Senden wird die Passphrase zur Verwendung des privaten Schlüssels abgefragt.



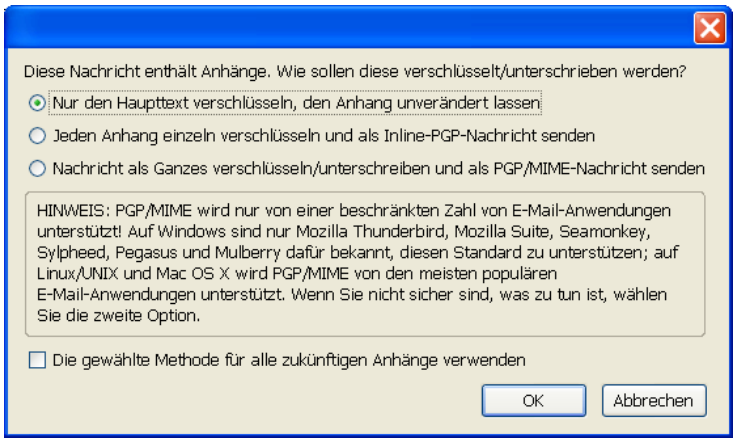
Optional kann, um den Stress für die häufige Eingabe der Passphrase zu reduzieren, diese für eine einstellbare Zeit (in diesem Beispiel 10 Minuten) aktiv bleiben. Diese wird dann in einem sogenannten Passphrasencache gespeichert.

Mit Bestätigung der Passphrase wird die Signatur erzeugt und an den Text der E-Mail angefügt. Falls so eingestellt, erfolgt nun noch einmal eine Sicherheitsabfrage, bevor die E-Mail tatsächlich gesendet wird. In diesem Fall ist der Text mit der angehängten Signatur im Hintergrund vor dem Senden bereits sichtbar.



Hinweis

Wenn mit der E-Mail eine Anlage versendet wird, erfolgt noch eine weitere Rückfrage:



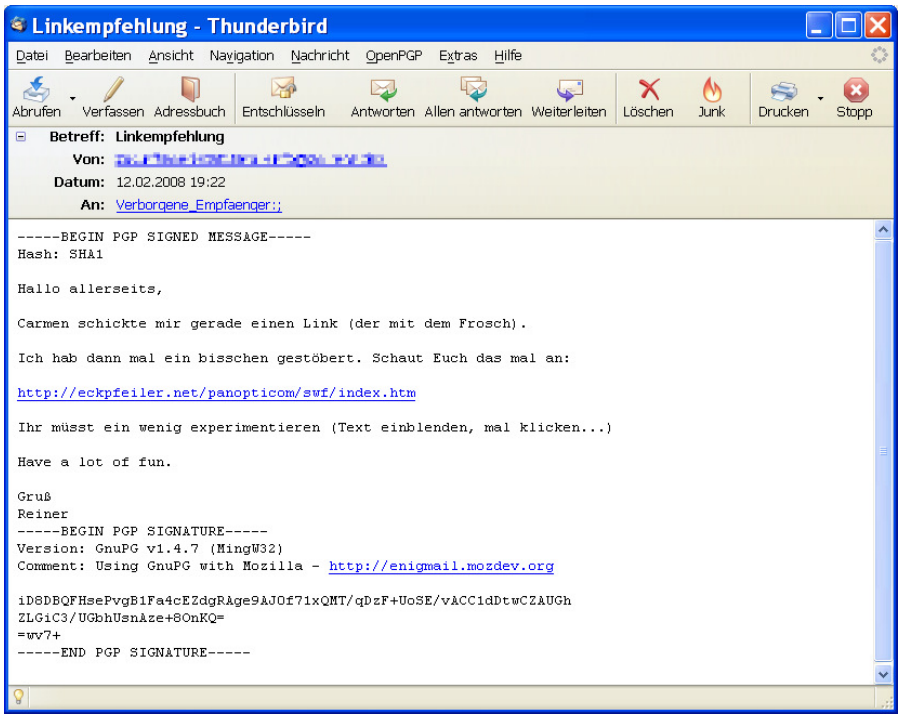
Der Grund für die Rückfrage ist die Art, wie die Signatur erzeugt werden soll.

Eine Anlage an eine E-Mail wird zum Versand in Text umgewandelt. Dieser kann nun zur Erzeugung der Signatur mit verwendet werden, oder aber auch nicht. Vorgeschlagen wird, den Anhang nicht mit einzubeziehen.

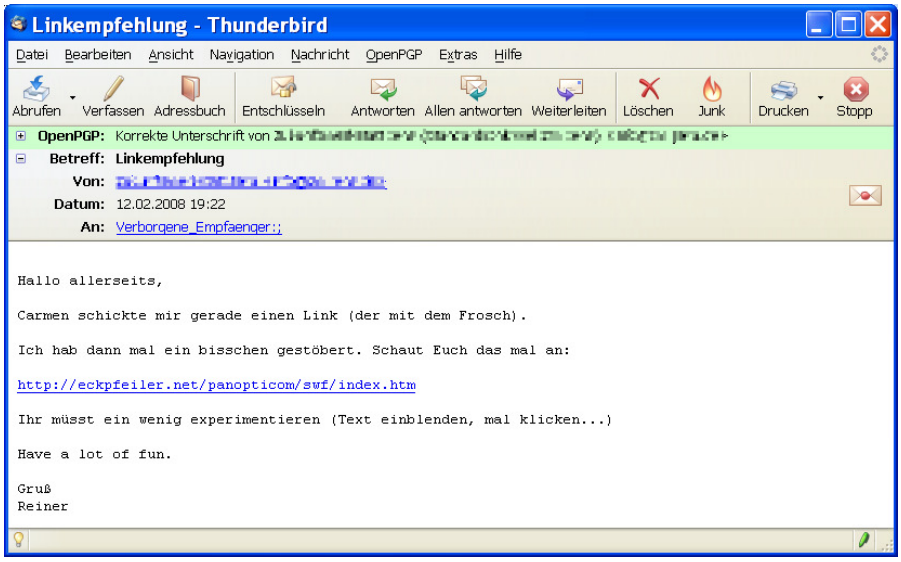
Mit dieser Option ist die Kompatibilität mit E-Mail-Clients, die den PGP/MIME-Standard nicht unterstützen gewährleistet. Im Zweifelsfall ist diese Option also die bessere, weil gut kompatible Wahl.

signierte Nachrichten empfangen

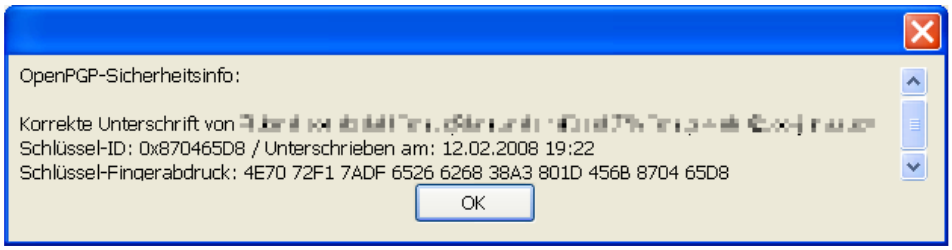
Der Empfang einer signierten Nachricht gestaltet sich grundsätzlich genauso wie der einer unsignierten Nachricht. Nachfolgend ist eine empfangene signierte E-Mail dargestellt, wie sie jeder E-Mail-Client darstellen würde – auch wenn dieser keine Verschlüsselung unterstützt.



Gut erkennbar sind die Teile der E-Mail, die als solche eindeutig gekennzeichnet sind. Angegeben sind darüber hinaus, mit welchem Verfahren der Hashwert berechnet und mit welcher Software die Signatur erzeugt wurde. Zum Prüfen der Signatur wird die Schaltfläche „Entschlüsseln“ angeklickt.



Wenn die Signatur korrekt ist, wird im Kopf der E-Mail ein grün hinterlegter Bereich mit einer entsprechenden Meldung eingeblendet. Zusätzlich sind rechts im Kopf der E-Mail ein Briefumschlagsymbol und rechts unten in der Statusleiste ein grünes Stift-Symbol dargestellt. Wird eines dieser Symbole angeklickt, wird die vollständige Meldung von OpenPGP eingeblendet:



Hinweis

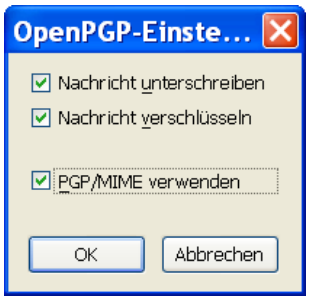
Ist der öffentliche Schlüssel des Absenders nicht verfügbar, erfolgt automatisch die Rückfrage, ob dieser von einem Schlüsselserver importiert werden oder anderweitig besorgt werden soll – beispielsweise, wenn dieser der E-Mail beigefügt ist.

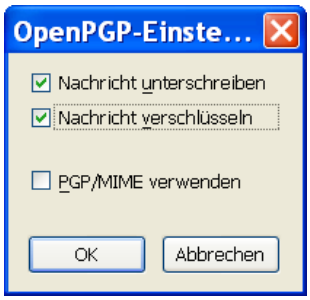
Hinweis

Im Menü OpenPGP kann eingestellt werden, dass die Ver- und Entschlüsselung automatisch erfolgen soll. Ist diese Option aktiviert, muss eigentlich gar nichts durch den Benutzer veranlasst werden. Sobald die E-Mail angezeigt wird, werden im Hintergrund alle eben beschriebenen Prozesse automatisch ausgeführt. Lediglich eine kleine Verzögerung der Anzeige weist darauf hin, dass da im Hintergrund einiges passiert.

verschlüsselte Nachrichten versenden

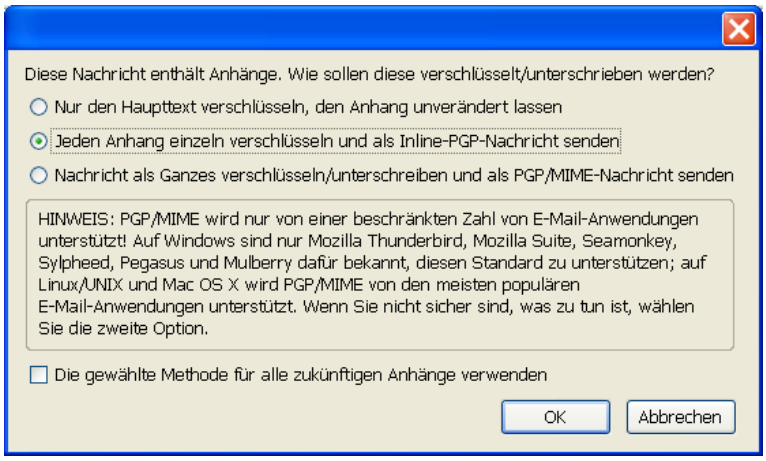
Die Verfahren sind analog dem Versenden signierter E-Mails. Bei der Festlegung sind zwei Verfahren zu unterscheiden:





Verwenden von PGP/MIME oder nicht. Wird PGP/MIME verwendet, werden der Nachrichtentext und die Anlage (die in Text umgewandelte Anlage) zusammengefasst, komprimiert und als Block insgesamt verschlüsselt. Der Empfänger muss eine E-Mail-Software verwenden, die diesen Block wieder aufdröseln und korrekt rekonstruieren kann, sonst ist die Nachricht verloren. Eine Reihe von E-Mail-Clients können das nicht.

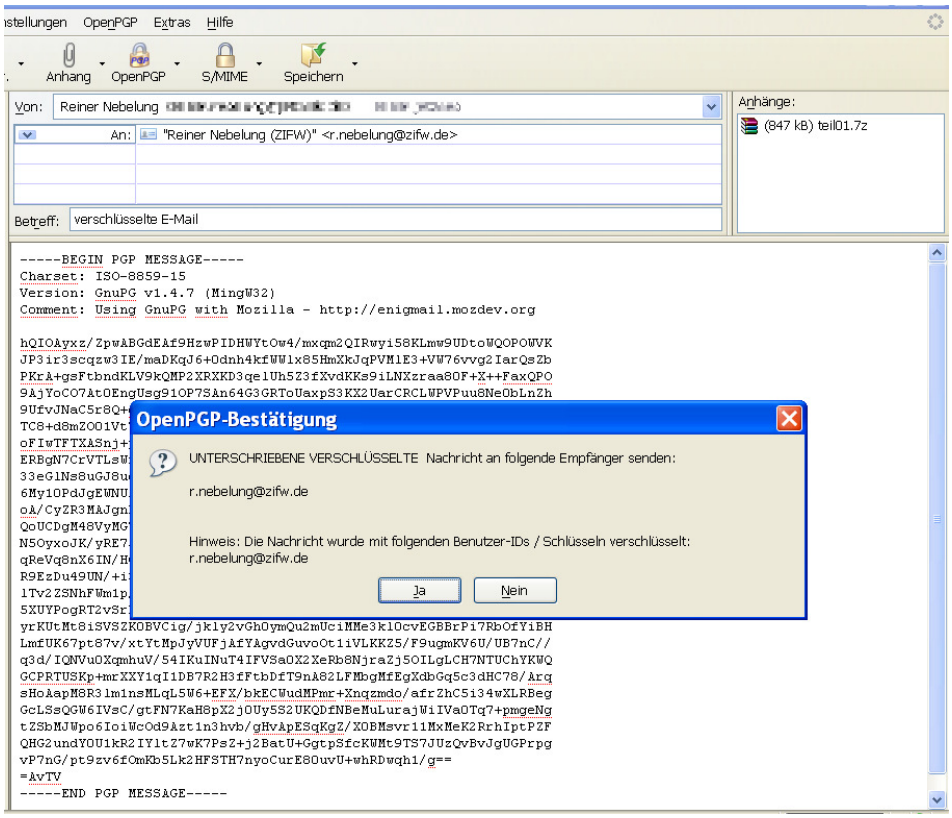
Im Fall, dass der PGP/MIME-Standard nicht verwendet wird, werden der Text und jede Anlage einzeln verschlüsselt. Der Umgang mit einer solchen Inline-PGP-Nachricht ist für den Empfänger nicht so komfortabel wie im Fall einer PGP/MIME-Nachricht. In diesem Fall wird nachfolgende Sicherheitsabfrage zusätzlich eingeblendet:



Hinweis

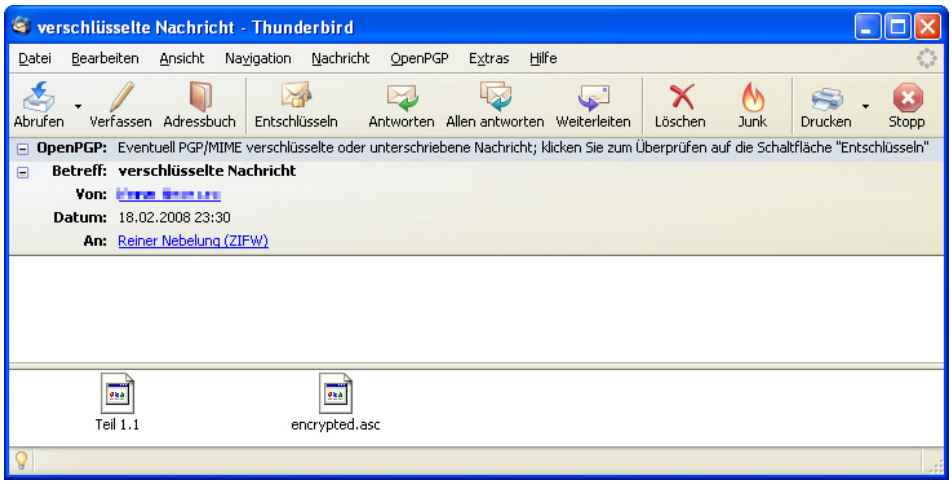
Die Verwendung des PGP/MIME-Standards ist in jedem Falle zu empfehlen, wenn bekannt ist, dass der Empfänger ebenfalls Thunderbird verwendet. Die Handhabung ist deutlich komfortabler (siehe Seite).

Beim Versand im PGP-Inline-Modus wird im Hintergrund bereits der verschlüsselte E-Mail-Text dargestellt.



verschlüsselte Nachrichten empfangen

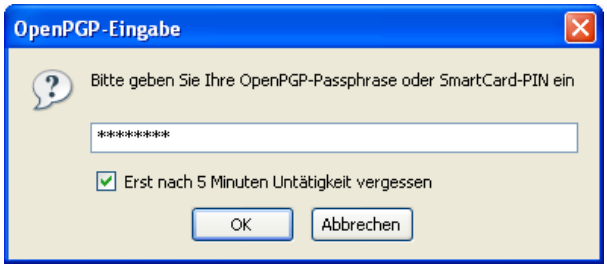
Der Empfang verschlüsselter Nachrichten geschieht zunächst genauso, wie der Empfang unverschlüsselter E-Mails – sie werden vom Server abgerufen. Auch die Darstellung dieser Nachrichten im Posteingangsordner unterscheidet sich in nichts von der der üblichen E-Mails.



Wie hier abgebildet, wird eine verschlüsselte E-Mail in Thunderbird dargestellt. Zur besseren Übersicht ist die E-Mail in einem eigenen Fenster geöffnet worden.

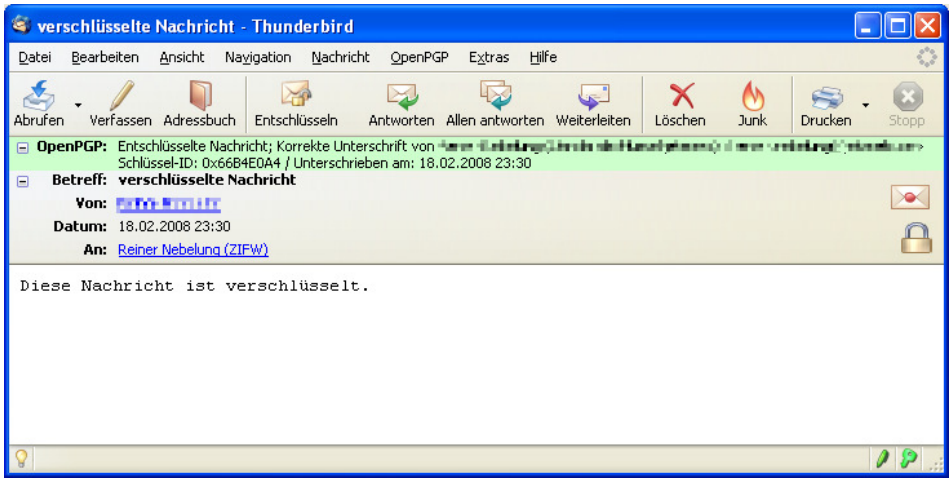
Gut erkennbar sind zwei Teile der E-Mail, die dem codierten Schlüssel (Teil 1.1) und der eigentlichen verschlüsselten Nachricht entsprechen (encrypted.asc). Mit Hilfe des privaten Schlüssels muss nun zuerst aus dem Teil 1.1 der symmetrische Schlüssel dieses Nachrichtenaustausches und mit dessen Hilfe die eigentliche Nachricht sowie die Signatur ermittelt werden.

Zum Entschlüsseln wird das entsprechende Symbol der Symbolleiste bzw. der entsprechende Befehl aus dem Menü OpenPGP gewählt.



Um die Aktionen ausführen zu können, ist die Passphrase erforderlich, die die Verwendung des privaten Schlüssels freigibt.

Im Ergebnis wird die entschlüsselte Nachricht angezeigt:



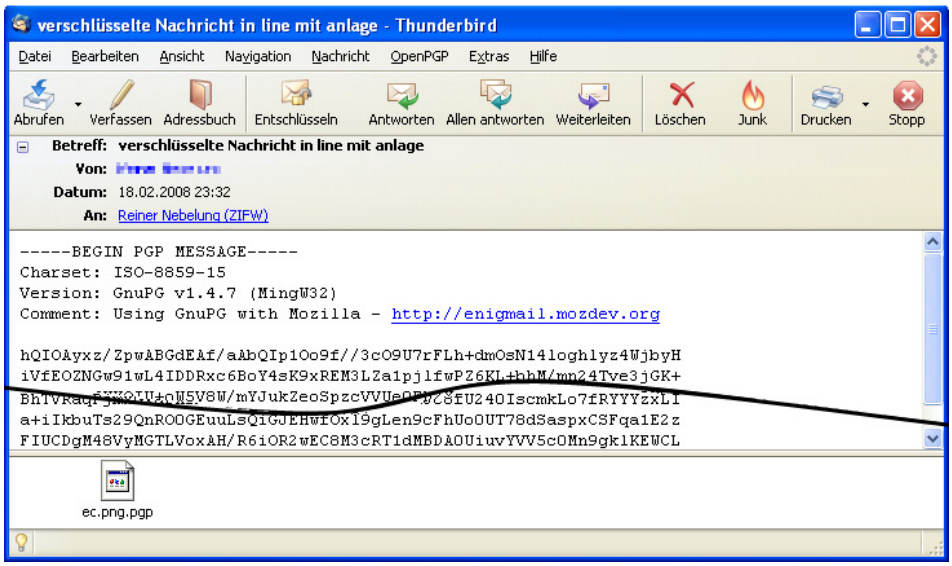
Die korrekte Entschlüsselung wird durch das Briefumschlag- und das Schloss-Symbol im E-Mail-Header sowie den grünen Stift und den grünen Schlüssel rechts unten in der Statuszeile dargestellt.

Bearbeitung verschlüsselter Anlagen

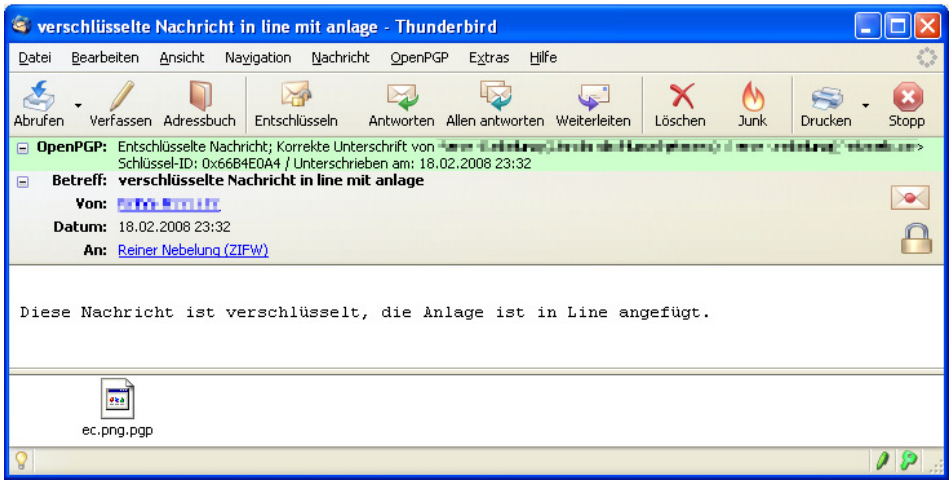
Eine besondere Situation stellen verschlüsselte E-Mails mit (verschlüsselten) Anlagen dar. Hier muss zwischen solchen, die dem PGP/MIME-Standard entsprechen und denen, die diesem Standard nicht entsprechen, unterschieden.

Technisch bedeutet das, dass entsprechend dem MIME-Standard sowohl der Nachrichtentext als auch die Anlagen zu einem (ASCII-Text-) Block zusammengefasst, komprimiert und anschließend verschlüsselt werden. Das bedeutet, dass der gesamte Nachrichtenteil der E-Mail ein einziger zusammenhängender Datenblock ist. Im Gegensatz dazu enthalten sogenannte PGP-Inline-Nachrichten für jede Anlage und für den eigentlichen Nachrichtenblock jeweils einen gesonderten, verschlüsselten (ASCII-Text-) Block. Daraus resultieren Unterschiede in der Behandlung der Anlagen.

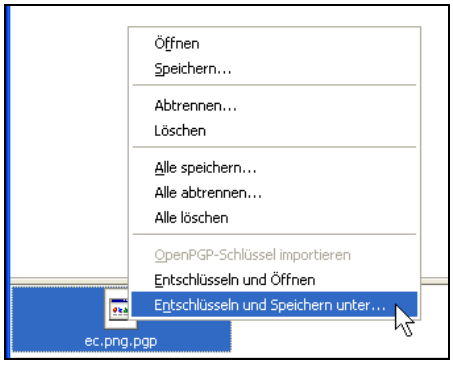
Eine E-Mail entsprechend dem PGP-Inline-Verfahren ist nachstehend dargestellt.



Erkennbar sind der verschlüsselte Textblock und die Anlage. Deren verschlüsselter Zustand ist am Dateinamen erkennbar: ec.png.pgp – der eigentliche Dateiname ist ec.png, eine Grafik. Nach der Entschlüsselung bietet sich folgende Ansicht:



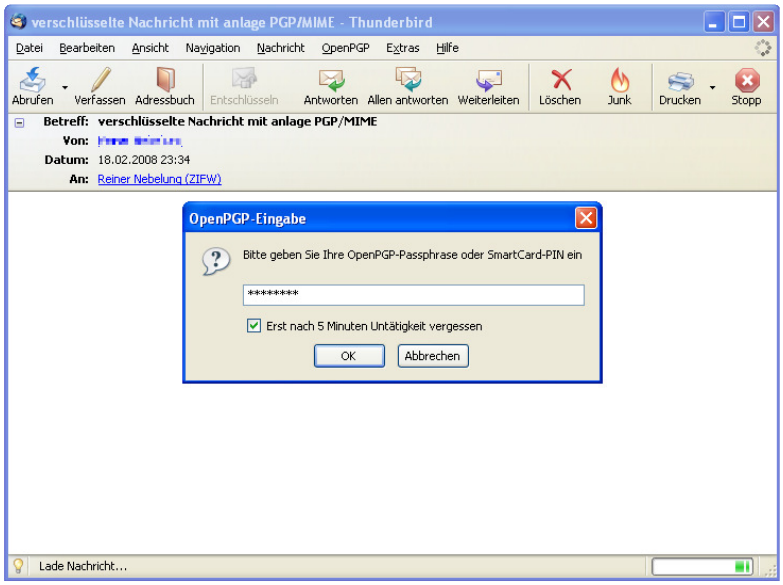
Die Anlage ist nach wie vor verschlüsselt und kann nicht angezeigt werden.



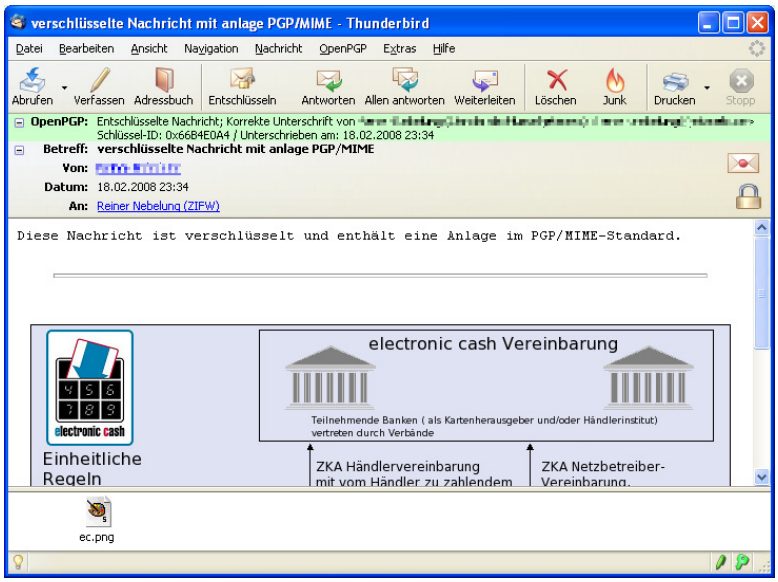
Erst über das Kontextmenü bzw. über den entsprechenden Menüpunkt im Menü Datei von Thunderbird kann die Anlage korrekt entschlüsselt werden.

Eine direkte Darstellung als in die E-Mail eingebundene Grafik ist mit diesem Verfahren nicht möglich.

Eine E-Mail, die nach dem PGP/MIME-Standard verschlüsselt ist, wird anders behandelt.



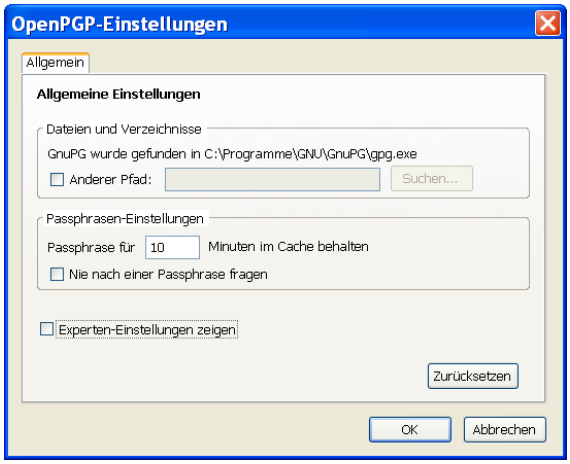
Hier wird gar nichts angezeigt, sondern als Erstes wird die Passphrase verlangt. Nach Eingabe der Passphrase wird die Nachricht sofort wie gewohnt dargestellt:



Dieses Verfahren ist sicher das komfortablere, wird aber nicht von allen E-Mail-Clients beherrscht.

Grundeinstellungen von OpenPGP

Die Grundeinstellungen von OpenPGP können über das Thunderbird-Menü OpenPGP > Einstellungen vorgenommen werden. Dabei wird folgender Dialog dargestellt.



Diese Einstellungen richten sich in der dargestellten Form an den einfachen Nutzer.

Einstellbar ist letztlich eigentlich nur, wie lange die Passphrase im Cache behalten wird bzw. ob die Passphrasenrückfrage ganz unterbunden werden soll. Diese Einstellung ist jedoch aus Sicht der Sicherheit problematisch. Weitergehende Einstellmöglichkeiten werden eingeblendet, wenn die Experteneinstellungen aktiviert werden.

Die Standard-Grundeinstellungen, die Enigmail mitbringt, sind für die meisten Fälle korrekt und hinreichend für einen reibungslosen Betrieb.

Wichtig

HTML-Mails sollten wenn irgend möglich nicht signiert und/oder verschlüsselt werden. Grund dafür ist der im Hintergrund erstellte Quelltext einer HTML-Seite, deren Zeilenumbruch nicht korrekt festgelegt werden kann. Der Zeilenumbruch zählt jedoch als ASCII-Zeichen im Sinne der Verschlüsselung und kann beim Empfänger dazu führen, dass die Signatur falsch ermittelt wird bzw. die Nachricht nicht entschlüsselt werden kann.

Hinweis

Die Verwaltung mehrerer Schlüssel ist mit OpenPGP kein Problem. Die Zuordnung, welcher Schlüssel für welche E-Mail-Adresse (also für welchen Empfänger) zur Anwendung kommt, geschieht über die E-Mail-Adresse selbst. Für den Fall, dass einer Adresse mehrere Schlüssel zugeordnet sind, können Empfängerregeln gesondert festgelegt werden. Diese haben dann Vorrang in der Verwendung.

Für weitergehende Informationen sei auf die offizielle Webseite des Enigmail-Projektes sowie weitere Seiten verwiesen, wo sich auch genauere Hinweise zu den Experteneinstellungen finden.

`http://www.erweiterungen.de/detail/Enigmail/` (Enigmail Download - deutsch)
`http://enigmail.mozdev.org/home/index.php` (Enigmail Homepage - englisch)
`http://www.thunderbird-mail.de/wiki/Enigmail` (Enigmail wiki - deutsch)
`http://www.gnupg.org/index.de.html` (GnuPG Homepage - deutsch)
`http://privacy.teuchtlurm.de/mailverschlusselung/verschlusseln-mit-thunderbird/`
(gute und übersichtliche Anleitung - bitte die Schreibweise beachten!!!)