Bundesministerium
für Wirtschaft
und Technologie

GnuPP für Durchblicker

Hintergrund-KnowHow zum E-Mail-Verschlüsselungssystem GnuPP

Ihr freier Schlüssel zur E-Mail-Sicherheit!

Mit kompletter Software, Cartoons und Adele, dem E-Mail-Roboter

GnuPP für Durchblicker

Das Hintergrund-KnowHow zum E-Mail-Verschlüsselungssystem GnuPP

Herausgegeben und gefördert vom
Bundesministerium für Wirtschaft und Technologie
Scharnhorststr. 34-37
10115 Berlin

Ansprechpartner:
Bundesministerium für Wirtschaft und Technologie
Referat Öffentlichkeitsarbeit

Impressum

Diese Seite darf nicht verändert werden

Autor: Manfred J. Heinze, TextLab text+media
Beratung: Lutz Zolondz, G-N-U GmbH
Illustrationen: Karl Bihlmeier, Bihlmeier & Kramer GbR
Layout: Isabel Kramer, Bihlmeier & Kramer GbR
Fachtext: Dr. Francis Wray, e-mediate Ltd.
Redaktion: Ute Bahn, TextLab text+media

Auflage, März 2002

1. Was ist GnuPP?

Das Projekt GnuPP (GNU Privacy Project) ist eine vom Bundeswirtschaftsministerium geförderte E-mail-Verschlüsselungssoftware. GnuPP bezeichnet das Gesamtpaket, das die Programme GnuPG, GPA, WinPT und andere Komponenten enthält.

Mit dem Verschlüsselungsprogramm GnuPG (GNU Privacy Guard) kann jedermann E-Mails sicher, einfach und kostenlos verschlüsseln. GnuPG kann privat oder kommerziell eingesetzt werden. Die Verschlüsselung von GnuPG kann nach dem heutigen Stand von Forschung und Technik nicht gebrochen werden.

GnuPG ist Freie Software oder Open-Source-Software. Das bedeutet, dass jedermann das Recht hat, sie nach Belieben kommerziell oder privat zu nutzen.

Und es bedeuetet, daß jedermann den Quellcode, also die eigentliche Programmierung des Programms, genau untersuchen soll und darf.

Für eine Sicherheits-Software ist diese garantierte Transparenz des Quellcodes eine unverzichtbare Grundlage. Nur so lässt sich die Vertrauenswürdigkeit eines Programmes prüfen.

GnuPG ist vollständig kompatibel mit PGP

GnuPG basiert auf dem internationalen Standard OpenPGP (RFC 2440), ist vollständig kompatibel zu PGP und benutzt die gleiche Infrastruktur (Schlüsselserver usw.)

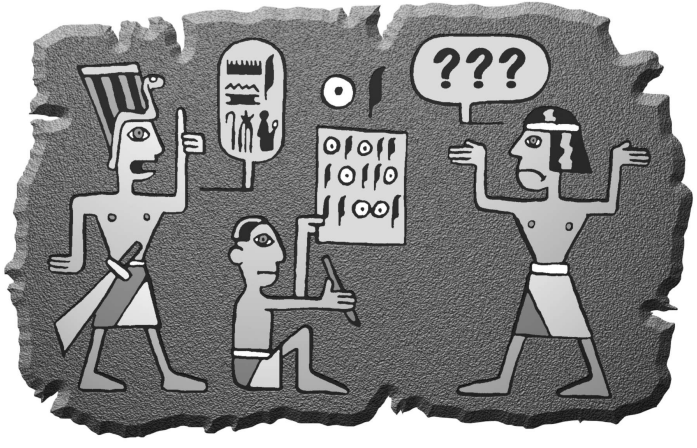
PGP ("Pretty Good Privacy") ist keine freie Software, sie wird seit mehreren Jahren nicht mehr unter der freien Softwarelizenz GNU General Public License (GNU GPL) vertrieben.

Weitere Informationen zu GnuPG und den Projekten der Bundesregierung zum Schutz des Internets finden Sie auf der Website sicherheit-im-internet.de des Bundeswirtschaftsministeriums.

2. Warum überhaupt verschlüsseln?

Die Verschlüsselung von Nachrichten wird manchmal als das zweitälteste Gewerbe der Welt bezeichnet. Verschlüsselungstechniken benutzten schon der Pharao Khnumhotep II, Herodot und Cäsar.

Wenigstens das zweitälteste Gewerbe der Welt ist heute dank GnuPP für jedermann frei und kostenlos zugänglich...



Die Computertechnik hat uns phantastische Mittel in die Hand gegeben, um rund um den Globus miteinander zu kommunizieren und uns zu informieren. Aber Rechte und Freiheiten, die in anderen Kommunikationsformen längst selbstverständlich sind, müssen wir uns in den neuen Technologien erst sichern. Das Internet ist so schnell und massiv über uns hereingebrochen, daß wir mit der Wahrung unserer Rechte noch nicht so recht nachgekommen sind.

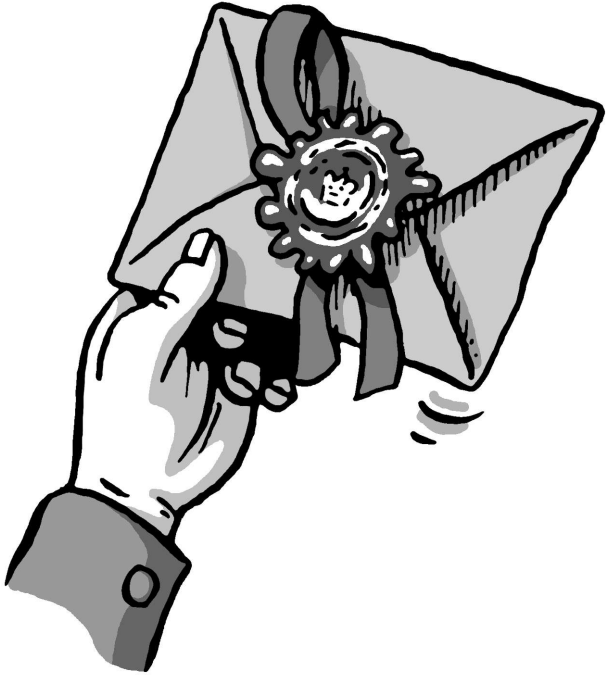
Beim altmodischen Briefschreiben haben wir die Inhalte unserer Mitteilungen ganz selbstverständlich mit einem Briefumschlag geschützt. Der Umschlag schützt die Nachrichten vor fremden Blicken, eine Manipulation am Umschlag kann man leicht bemerken. Nur wenn etwas nicht ganz so wichtig ist, schreibt man es auf eine ungeschützte Postkarte, die zur Not auch der Briefträger oder andere lesen können.

Ob die Nachricht wichtig, vertraulich oder geheim ist, das bestimmt man selbst und niemand sonst..

Diese Entscheidungsfreiheit haben wir bei E-Mail nicht. Eine normale E-Mail ist immer offen wie eine Postkarte, und der elektronische "Briefträger" - und andere - können sie immer lesen. Die Sache ist sogar noch schlimmer: die Computertechnik bietet nicht nur die Möglichkeiten, die vielen Millionen E-Mails täglich zu befördern und zu verteilen, sondern auch, sie zu kontrollieren.

Niemand hätte je ersthaft daran gedacht, alle Briefe und Postkarten zu sammeln, ihren Inhalt auszuwerten oder Absender und Empfänger zu protokollieren. Das wäre einfach nicht machbar gewesen, oder es hätte zu lange gedauert. Mit der modernen Computertechnik ist das zumindestens technisch möglich. Und ob genau das nicht heute schon mit Ihrer und meiner E-Mail geschieht, wissen wir nicht und werden es wahrscheinlich auch nie erfahren.

Denn: der Umschlag fehlt.



Was wir Ihnen hier vorschlagen, ist ein Umschlag für Ihre elektronischen Briefe. Ob Sie ihn benutzen, wann, für wen und wie oft, ist ganz allein Ihre Sache. Software wie GnuPP gibt Ihnen lediglich die Wahlfreiheit zurück. Die Wahl, ob Sie persönlich eine Nachricht für wichtig und schützenswert halten oder nicht.

Das ist der Kern des Rechts auf Brief-, Post- und Fernmeldegeheimnis im Grundgesetz, und dieses Recht können Sie mit Hilfe der Software GnuPP wahrnehmen. Sie müssen sie nicht benutzen - Sie müssen ja auch keinen Briefumschlag benutzen. Aber es ist Ihr gutes Recht.

Um dieses Recht zu sichern, bietet GnuPP Ihnen sogenannte „starke Verschlüsselungstechnologie". „Stark" bedeutet hier: mit keinem gegenwärtigen Mittel zu knacken. In vielen Ländern war starke Verschlüsselungstechnologie bis vor ein paar Jahren den Militärs und Regierungsbehörden vorbehalten. Das Recht, sie für jeden Bürger nutzbar zu machen, haben sich die Internet-Nutzer mühsam erobert; manchmal auch mit der Hilfe von klugen und weitsichtigen Menschen in Regierungsinstitutionen, wie im Falle von GnuPP.

GnuPG wird von Sicherheitsexperten in aller Welt als eins der sichersten entsprechenden Software-Systeme angesehen.

Wie wertvoll diese Sicherheit für Sie ist, liegt ganz in Ihrer Hand, denn Sie allein bestimmen das Verhältnis zwischen Bequemlichkeit bei der Verschlüsselung und größtmöglicher Sicherheit. Dazu gehören die wenigen, aber umso wichtigeren Vorkehrungen, die Sie treffen müssen, und die wir im Folgenden besprechen:

3. Wie funktioniert GnuPP?

Das Besondere an GnuPP und der zugrundeliegenden Public Key-Methode ist, daß sie jeder verstehen kann und soll. Nichts daran ist Geheimwissen – es ist nicht einmal besonders schwer zu verstehen.

Die Benutzung von GnuPP ist sehr einfach, seine Wirkungsweise dagegen ziemlich kompliziert. Wir werden in diesem Kapitel erklären, wie GnuPP funktioniert – nicht in allen Details, aber so, dass die Prinzipien dahinter deutlicher werden. Wenn Sie diese Prinzipien kennen, werden Sie ein hohes Vertrauen in die Sicherheit von GnuPP gewinnen.

Ganz am Ende dieses Buches, in Kapitel 13, können Sie – wenn Sie wollen – auch noch die letzten Geheimnisse um die Public Key-Kryptografie lüften und entdecken, warum GnuPP nicht zu knacken ist.

Der Herr der Schlüsselringe

Wenn man etwas sehr Wertvolles sichern will, schließt man es am besten ein - mit einen Schlüssel. Noch besser mit einem Schlüssel, den es nur einmal gibt und den man ganz sicher aufbewahrt.

Denn wenn dieser Schlüssel in die falschen Hände fällt, ist es um die Sicherheit des wertvollen Gutes geschehen. Dessen Sicherheit steht und fällt mit der Sicherheit des Schlüssels. Also muss man den Schlüssel mindestens genauso gut absichern, wie das zu sichernde Gut selbst, und Art und Beschaffenheit des Schlüssels müssen völlig geheim gehalten werden.

Geheime Schlüssel sind in der Kryptografie ein alter Hut: schon immer hat man Botschaften geheimzuhalten versucht, indem man den Schlüssel geheimhielt. Funktioniert hat das letztendlich nie: jede einzelne dieser Methoden wurde geknackt – früher oder später.

Das Grundproblem bei der „normalen" geheimen Nachrichtenübermittlung ist, dass für Ver- und Entschlüsselung derselbe Schlüssel benutzt wird, und dass sowohl der Absender als auch der Empfänger diesen geheimen Schlüssel kennen müssen.

Das führt zu einer ziemlich paradoxen Situation: bevor man mit einem solchen System ein Geheimnis – eine verschlüsselte Nachricht - mitteilen kann, muss man schon vorher ein anderes Geheimnis – den Schlüssel – mitgeteilt haben. Und da liegt der Hase im Pfeffer: man muss sich ständig mit dem Problem herumärgern, dass der Schlüssel unbedingt ausgetauscht werden muss, aber auf keinen Fall von einem Dritten abgefangen werden darf.

GnuPP dagegen arbeitet – außer mit dem Geheimschlüssel - mit einem weiteren Schlüssel („key"), der vollkommen frei und öffentlich („public") zugänglich ist

Man spricht daher auch von GnuPP als einem „Public Key"-Verschlüsselungssystem.

Das klingt widersinnig, ist es aber nicht. Der Witz an der Sache: es muss kein Geheimschlüssel mehr ausgetauscht werden. Im Gegenteil: der Geheimschlüssel darf auf keinen Fall ausgetauscht werden! Weitergegeben wird nur der öffentliche Schlüssel – und den kennt sowieso jeder.

Mit GnuPP benutzen Sie also ein Schlüsselpaar – eine geheime und eine zweite öffentliche Schlüsselhälfte. Beide Hälften sind durch eine komplexe mathematische Formel untrennbar miteinander verbunden. Nach heutiger wissenschaftlicher und technischer Kenntnis ist es unmöglich, einen Schlüsselteil aus dem anderen zu berechnen und damit den Code zu knacken. In Kapitel 13 erklären wir, wie das funktioniert.

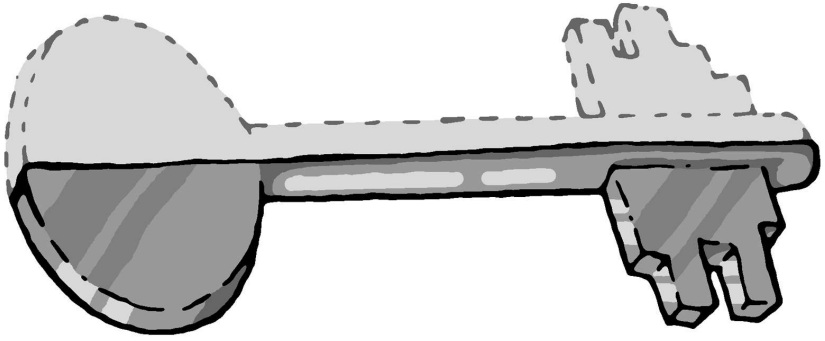
Das GnuPP-Prinzip ist wie gesagt recht einfach:

der geheime oder private Schlüssel (secret oder private key) muß geheim gehalten werden.

der öffentliche Schlüssel (oder public key) soll so öffentlich wie möglich gemacht werden.

Beide Schlüsselteile haben ganz und gar unterschiedliche Aufgaben:

der geheime Schlüsselteil **entschlüsselt** Nachrichten



der öffentliche Schlüsselteil **verschlüsselt** Nachrichten.

Der öffentliche Safe

In einem kleinen Gedankenspiel wird die Methode des Public Key-Verschlüsselungssystems und ihr Unterschied zur „nicht-public key"-Methode deutlicher:

Die „nicht-Public Key-Methode" geht so:

Stellen Sie sich vor, Sie stellen einen Briefkasten vor Ihrem Haus auf, über den Sie geheime Nachrichten übermitteln wollen.

Der Briefkasten ist mit einem Schloss verschlossen, zu dem es nur einen einzigen Schlüssel gibt. Niemand kann ohne diesen Schlüssel etwas hineinlegen oder herausnehmen. Damit sind Ihre geheimen Nachrichten zunächst einmal gut gesichert.

Da es nur einen Schlüssel gibt, muß Ihr Korrespondenzpartner denselben Schlüssel wie Sie haben, um den Briefkasten damit auf- und zuschließen und eine Geheimnachricht deponieren zu können.

Diesen Schlüssel müssen Sie Ihrem Korrespondenzpartner auf geheimem Wege übergeben.

Erst wenn der andere den Geheimschlüssel hat, kann er den Briefkasten öffnen und die geheime Nachricht lesen.

Alles dreht sich also um diesen Schlüssel: wenn ein Dritter ihn kennt, ist es sofort aus mit den Geheimbotschaften. Sie und Ihr Korrespondenzpartner müssen ihn also genauso geheim austauschen wie die Botschaft selbst.

Aber – eigentlich könnten Sie ihm bei dieser Gelegenheit ja auch gleich die geheime Mitteilung übergeben....

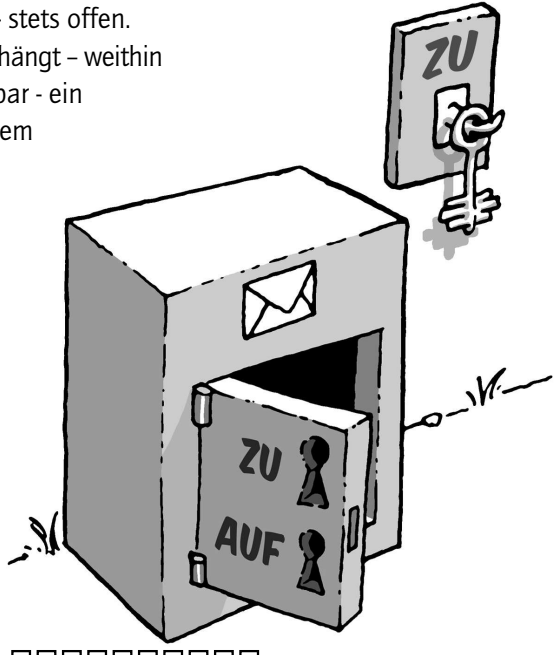
Übertragen auf die E-Mail-Verschlüsselung: weltweit müssten alle E-Mail-Teilnehmer geheime Schlüssel besitzen und auf geheimem Wege austauschen, bevor sie geheime Nachrichten per E-Mail versenden könnten.

Vergessen wir diese Möglichkeit am besten sofort...

Jetzt die Public Key-Methode:

Sie installieren wieder einen Briefkasten vor Ihrem Haus. Aber: dieser Briefkasten ist – ganz im Gegensatz zu dem ersten Beispiel - stets offen. Direkt daneben hängt – weithin öffentlich sichtbar - ein Schlüssel, mit dem jedermann den Briefkasten zuschließen kann.

Zuschließen, aber nicht aufschließen: das ist der Trick.



Dieser Schlüssel gehört Ihnen, und - Sie ahnen es: es ist Ihr öffentlicher Schlüssel.

Wenn jemand Ihnen eine geheime Nachricht hinterlassen will, legt er sie in den Briefkasten und schließt mit Ihrem öffentlichen Schlüssel ab. Jedermann kann das tun, denn der Schlüssel dazu ist ja völlig frei zugänglich.

Kein anderer kann den Briefkasten nun öffnen und die Nachricht lesen. Selbst derjenige, der die Nachricht in dem Briefkasten eingeschlossen hat, kann ihn nicht wieder aufschliessen, zum Beispiel um die Botschaft nachträglich zu verändern.

Denn die öffentliche Schlüsselhälfte taugt ja nur zum Abschließen.

Aufschließen kann man den Briefkasten nur mit einem einzigen Schlüssel: Ihrem eigenen geheimen oder privaten Schlüsselteil.

Wieder übertragen auf die E-Mail-Verschlüsselung: jedermann kann eine E-Mail an Sie verschlüsseln. Er benötigt dazu keineswegs einen geheimen, sondern ganz im Gegenteil einen vollkommen öffentlichen, „ungeheimen" Schlüssel. Nur ein einziger Schlüssel entschlüsselt die E-Mail wieder: Ihr privater, geheimer Schlüssel.

Spielen wir das Gedankenspiel noch einmal anders herum:

Wenn Sie einem anderen eine geheime Nachricht zukommen lassen wollen, öffnen Sie dessen Briefkasten mit seinem öffentlichen, frei verfügbaren Schlüssel.

Sie müssen Ihren Briefpartner dazu nicht persönlich kennen, ihn getroffen oder je mit ihm gesprochen haben, denn sein öffentlicher Schlüssel ist überall und jederzeit zugänglich. Wenn Sie Ihre Nachricht hinterlegt und den Briefkasten des Empfängers mit seinem öffentlichem Schlüssel wieder verschlossen haben, ist sie völlig unzugänglich für jeden anderen, auch für Sie selbst. Nur der Empfänger kann den Briefkasten mit seinem privaten Schlüssel öffnen und die Nachricht lesen.

Was ist nun eigentlich gewonnen: es gibt immer noch einen geheimen Schlüssel!?

Der Unterschied gegenüber der „nicht-Public Key-Methode" ist allerdings ein gewaltiger:

Ihren privater Schlüssel kennen und benutzen nur Sie selbst. Er wird niemals einem Dritten mitgeteilt –die Notwendigkeit einer geheimen Vereinbarung entfällt, sie verbietet sich sogar.

Es muss überhaupt nichts Geheimes mehr zwischen Absender und Empfänger ausgetauscht werden - weder eine geheime Vereinbarung noch ein geheimes Codewort.

Das ist – im wahrsten Sinne des Wortes - der Knackpunkt: alle „alten" Verschlüsselungs-verfahren wurden geknackt, weil ein Dritter sich beim Schlüsselaustausch in den Besitz des Schlüssels bringen konnte.

Dieses Risiko entfällt, weil der Geheimschlüssel nicht ausgetauscht wird und sich nur an einem einzigen Ort befindet: Ihrem Gedächtnis.

4. Der Passwort-Satz

Wie Sie oben gesehen haben, ist der private Schlüssel eine der wichtigsten Komponenten im Public Key-Verschlüsselungssystem. Man muss ihn zwar nicht mehr auf geheimem Wege mit seinen Korrespondenzpartnern austauschen (was bei der E-Mail-Kommunikation auch so gut wie unmöglich wäre), aber nach wie vor ist seine Sicherheit der Schlüssel zur Sicherheit des ganzen Systems.

Der Schutz des privaten Schlüssels besteht aus einem Passwort-Satz..

Passwort-Satz, weil er aus einem Satz und nicht nur aus einem Wort bestehen soll. Sie müssen diesen Passwort-Satz wirklich „im Kopf" haben und niemals aufschreiben müssen.

Trotzdem darf er nicht erraten werden können. Das klingt vielleicht widersprüchlich, ist es aber nicht. Es gibt einige erprobte Tricks, mit deren Hilfe man sich einen völlig individuellen, leicht zu merkenden und nicht zu erratenden Passwort-Satz ausdenken kann.

Eine guter Passwort-Satz kann so entstehen:

Denken Sie an einen Ihnen gut bekannten Satz, z.B.:

Ein blindes Huhn findet auch einmal ein Korn

Aus diesem Satz nehmen Sie zum Beispiel jeden dritten Buchstaben:

nieuf dahn lnr

Diesen Buchstabensalat kann man sich zunächst nicht unbedingt gut merken, aber man kann ihn eigentlich nie vergessen, solange man den ursprünglichen Satz im Kopf hat. Im Laufe der Zeit und je öfter man ihn benutzt, prägt sich so ein Passwort-Satz ins Gedächtnis. Erraten kann ihn niemand.

Denken Sie an ein Ereignis, das sich bereits fest in Ihrem persönlichen Langzeitgedächtnis verankert hat. Vielleicht gibt es einen Satz, mit dem sich Ihr Kind oder Ihr Partner „unvergesslich" gemacht hat. Oder eine Ferienerinnerung, oder der Titel eines für Sie wichtigen Liedes.

Verwenden Sie kleine und große Buchstaben, Nummern, Sonder- und Leerzeichen durcheinander. Im Prinzip ist alles erlaubt, auch „Ö", „ß", „$" usw.
Aber Vorsicht - falls Sie Ihren geheimen Schlüssel im Ausland an einem fremden Rechner benutzen wollen, bedenken Sie, daß fremdsprachige Tastaturen diese Sonderzeichen oft nicht haben.

machen Sie Rechtschreib-fehler, z.B. „feLer" statt „Fehler". Natürlich müssen Sie sich diese „feLer" gut merken können. Oder wechseln Sie mittendrin die Sprache.
Aus dem schönen Satz

In München steht ein Hofbräuhaus

könnten man beispielsweise diesen Passwort-Satz machen:

inMinschen stet 1h0f breuhome

denken Sie sich einen Satz aus, der möglichst unsinnig ist, den Sie sich aber doch merken können, wie z.B.:

Es blaut so garstig beim Walfang, neben Taschengeld, auch im Winter.

Ein Passwort-Satz in dieser Länge ist ein sicherer Schutz für den geheimen Schlüssel.

Es darf auch kürzer sein, wenn Sie einige Buchstaben groß schreiben, z.B. so:

Es blAut nEBen TaschengeLd auch im WiNter.

Kürzer, aber nicht mehr so leicht merken. Wenn Sie einen noch kürzeren Passwort-Satz verwenden, indem Sie hier und da Sonderzeichen benutzen, haben Sie zwar bei der Eingabe weniger zu tippen, aber die Wahrscheinlichkeit, daß Sie Ihr Passwort-Satz vergessen, wird dabei noch größer.

Ein extremes Beispiel für einen möglichst kurzen, aber dennoch sehr sicheren Passwort-Satz ist dieses hier:

R!Qw"s,UIb *7$

In der Praxis haben sich solche Zeichenfolgen allerdings als recht wenig brauchbar herausgestellt, da man einfach zu wenig Anhaltspunkte für die Erinnerung hat.

Ein schlechter Passwort-Satz ist blitzschnell geknackt, wenn er:

schon für einen anderen Zweck benutzt wird; z.B. für einen E-Mail-Account oder Ihr Handy

aus einem Wörterbuch stammt. Hacker lassen in Minutenschnelle komplette Wörterbücher elektronisch über ein Passwort laufen.

aus einem Geburtsdatum oder einem Namen besteht. Wer sich die Mühe macht, Ihre E-Mail zu entziffern, kann auch ganz leicht an diese Daten herankommen.

ein landläufiges Zitat ist wie „Schau mir in die Augen, Kleines" oder „to be or not to be". Auch mit derartigen gängigen Zitaten scannen Hacker routinemäßig und blitzschnell ein Passwort.

aus nur einem Wort oder aus weniger als 8 Zeichen besteht. Denken Sie sich einen längeren Passwort-Satz aus, kein Passwort.

Wenn Sie nun Ihren Passwort-Satz zusammenstellen, nehmen Sie auf gar keinen Fall eines der oben angeführten Beispiele. Denn es liegt auf der Hand, daß jemand, der sich ernsthaft darum bemüht, Ihr Passwort-Satz herauszubekommen, zuerst ausprobieren würde, ob Sie nicht eines dieser Beispiele genommen haben, falls er auch diese Informationen gelesen hat.

Seien Sie kreativ. Denken Sie sich jetzt einen Passwort-Satz aus. Unvergesslich und unknackbar.

Lesen Sie dann im „Einsteiger-Handbuch", Kapitel 3 weiter.

5. Schlüssel im Detail

Der Schlüssel, den Sie erzeugt haben, besitzt einige Kennzeichen:

die Benutzerkennung

die Schlüsselkennung

das Verfallsdatum

das Benutzervertrauen

das Schlüsselvertrauen

Die Benutzerkennung besteht aus dem Namen und der E-Mail-Adresse, die Sie während der Schlüsselerzeugung eingegeben haben, also z.B. Fritz Mustermann <f.mustermann@firma.de>

Die Schlüsselkennung verwendet die Software intern, um mehrere Schlüssel voneinander zu unterscheiden. Mit dieser Kennung kann man auch nach öffentlichen Schlüsseln suchen, die auf den Keyservern liegen. Was Keyserver sind, erfahren Sie im folgenden Kapitel.

Das Verfallsdatum ist normalerweise auf „kein Verfallsdatum" gesetzt. Sie können das ändern, indem Sie auf die Schaltfläche „Ändern" klicken und ein neues Ablaufdatum eintragen. Damit können Sie Schlüssel nur für eine begrenzte Zeit gültig erklären, zum Beispiel, um sie an externe Mitarbeiter auszugeben.

Das Benutzervertrauen beschreibt das Maß an Vertrauen, das Sie subjektiv in den Besitzer des Schlüssels setzen. Es kann über die Schaltfläche „Ändern" editiert werden.

Das Schlüsselvertrauen schließlich bezeichnet das Vertrauen, das man gegenüber dem Schlüssel hat. Wenn man sich von der Echtheit eines Schlüssels überzeugt und ihn dann möglicherweise auch signiert hat, erhält er volles „Schlüsselvertrauen".

Diese Angaben sind für die tagtägliche Benutzung des Programms nicht unbedingt wichtig. Sie werden relevant, wenn Sie neue Schlüssel erhalten oder ändern. Wir besprechen die Punkte „Benutzervertrauen" und „Schlüsselvertrauen" in Kapitel 9.

6. Die Schlüsselserver

Um verschlüsselt mit anderen zu kommunizieren, müssen die Partner ihre Schlüssel veröffentlichen und austauschen. Dazu ist - Sie erinnern sich an Kapitel 1 - keine Geheimniskrämerei notwendig, denn Ihr öffentlicher Schlüsselteil ist ja ganz und gar „ungeheim".

Im Internetzeitalter ist eine möglichst große Verbreitung Ihres öffentlichen Schlüssels überhaupt kein Problem. Sie können ihn z.B. über internationale Keyserver oder per E-Mail publizieren - diese beiden Möglichkeiten haben wir Ihnen im „Einsteiger-Handbuch" vorgeschlagen. Es gibt aber noch andere:

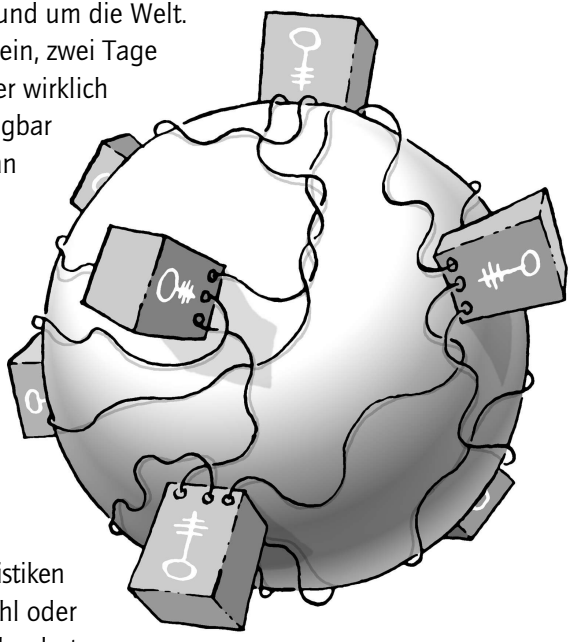
Verbreitung des Schlüssels über die eigene Homepage

als Dateianhang an einer E-Mail

last but not least: persönlich per Diskette

Am praktischsten ist sicher die Veröffentlichung über die Keyserver, die von allen Programmen nach dem OpenPGP-Standard benutzt werden können. Diese Möglichkeit haben wir bereits im Manual „Einsteiger-Handbuch" Kapitel 4 vorgestellt. Es genügt, den Schlüssel an irgendeinen der Keyserver zu senden, denn alle synchronsieren sich weltweit miteinander.

Ein Keyserver ist in GnuPP stets voreingestellt. Ein Mausklick genügt, und Ihr Schlüssel ist unterwegs rund um die Welt. Es kann ein ein, zwei Tage dauern, bis er wirklich überall verfügbar ist, aber dann haben Sie einen globalen Schlüssel! Die Schlüsselserver sind völlig dezentral organisiert, aktuelle Statistiken über ihre Zahl oder die Anzahl der dort liegenden Schlüssel gibt es nicht.



Dies ist durchaus beabsichtigt und Teil des Public Key-Konzeptes, denn ein „Big Brother" hat so praktisch keine Chance.

Das OpenPGP-Netz http://www.keyserver.net/ ist zum Beispiel der Sammelpunkt für ein ganzes Netz dieser Server, oft benutzt werden ebenfalls http://germany.keyserver.net/en/ oder der Keyserver des Deutschen Forschungsnetzes DFN http://www.dfn-pca.de/pgpkserv/. .

Genauso einfach können Sie auf den Keyservern nach einem öffentlichen Schlüssel suchen. Geben Sie in das Suchfeld den Namen des Schlüsselbesitzers ein oder seine E-Mail-Adresse. Als Ergebnis sehen Sie etwa eine solche Ausgabe:

```
pub 1024/C06525EC
2001/10/24 ... usw.
```

und evtl. noch

```
sig C06525EC ... usw.
sig B3B2A12C ... usw.
```

Alle drei Eintragungen sind Bestandteil des Schlüssels.

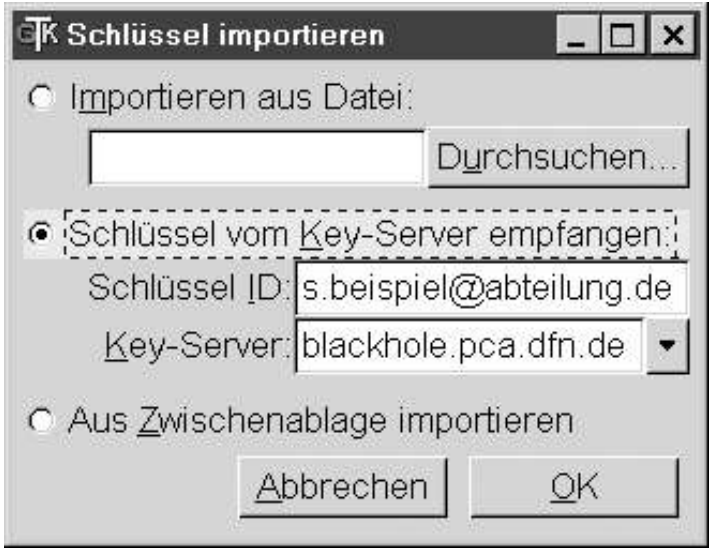
Sie benötigen aber nur den ersten Teil: das ist der öffentliche Schlüssel. Der zweite Teil ist die sogenannte Selbstzertifizierung, der dritte die Bestätigung der persönlichen Identität des Schlüsselinhabers.

Klicken Sie nun den Link des ersten Schlüsselteils (pub usw.) an:

Sie sehen den Ihnen schon bekannten Textblock, der den eigentlichen öffentlichen Schlüssel bildet.

Im Einsteigerhandbuch, Kapitel 6 und 7 zeigen wir Ihnen, wie man diesen Schlüssel importiert, d.h., am eigenen GnuPP-Schlüsselbund befestigt, und damit eine E-Mail an den Besitzer verschlüsselt.

Diese Suche nach einem Schlüssel funktioniert auch direkt aus GnuPP: Sie können einfach die E-Mail-Adresse des Schlüsselbesitzers eingeben, oder auch die Schlüsselkennung, falls Ihnen diese bekannt ist. Klicken Sie dazu auf „Import", und dort auf „Schlüssel vom Key-Server empfangen"



GnuPP sucht dann den Schlüssel, importiert ihn und zeigt ihn im Schlüsselverwaltungs-Fenster an.

7. Der Schlüssel als Dateianhang

Im Einsteiger-Handbuch Kapitel 4. haben Sie gesehen, wie einfach man seinen öffentlichen Schlüssel per E-Mail verschicken kann. Wir haben dabei den Schlüssel in einen Ordner exportiert, geöffnet und in die Zwischenablage kopiert. Von dort aus wurde der Schlüssel in ein E-Mail Programm kopiert und schließlich versandt. Noch einfacher geht es, wenn man den Schlüssel – genau wie im vorherigen Beispiel – exportiert und dann direkt als E-Mail-Anhang verschickt.

Dazu klicken Sie auf im GNU Privacy Assistant auf [Export] in der Iconleiste und dann in dem sich öffnenden Dialog auf [Exportieren in Datei]. Wählen Sie mit [Durchsuchen...] einen geeigneten Ordner auf Ihrem PC, z.B. C:\Eigene Dateien\ und speichern Sie den Schlüssel dort z.B. als „mein_key.asc".

Nun ziehen Sie den exportierten Schlüssel als Dateianhang in das entsprechende Fenster Ihres E-Mail-Programms, genauso wie jede andere Datei, und senden sie ihn an den Empfänger.

8. PlugIns für E-Mail-Programme

Im „Einsteiger-Handbuch" haben wir im Kapitel 5, „Sie entschlüsseln eine E-Mail" erwähnt, daß es PlugIns für bestimmte E-Mail-Programme gibt, die die Ver- und Entschlüsselung erleichtern. Die im Schnelleinstieg vorgestellte Methode mit dem Frontend WinPT funktioniert sehr einfach und schnell, und zwar mit jedem beliebigen E-Mail- und Text-Programm. Trotzdem ist für viele E-Mail-Anwender ein spezieller Programmzusatz in ihrem Lieblings-E-Mailer ein Vorteil.

GnuPG-PlugIns gibt es im Moment für Microsoft Outlook, PostMe (beide Windows), sowie mail (Apple MacOS X). Unter Linux stehen mit KMail und mutt E-Mail-Clients mit integriertem GnuPG-Support zur Verfügung.

Da sämtliche Komponenten des GnuPP-Pakets als Freie Software entstehen, ist die Entwicklung stark im Fluss.
Aktuelle Informationen über neue Komponenten finden Sie unter diesen Webadressen:

auf der GnuPP-Webpage http://www.gnupp.de/

auf der Informations-Site des BMWi https://sicherheit-im-internet.de/

im „Sicherheit im Internet"-Newsletter, den Sie unter https://www.sicherheit-im-internet.de/service/newsletter.phtml bestellen können - übrigens mit dem GnuPP-Paket signiert.

9. Die Schlüsselprüfung

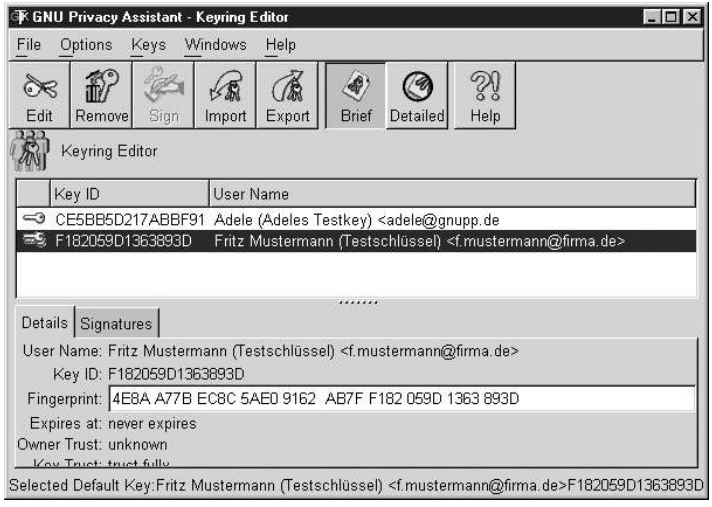
Woher wissen Sie eigentlich, dass der fremde öffentliche Schlüssel wirklich vom Absender stammt? Und umgekehrt - warum sollte Ihr Korrespondenzpartner glauben, daß der öffentliche Schlüssel, den Sie ihm geschickt haben, auch wirklich von Ihnen stammt? Die Absenderangabe auf einer E-Mail besagt eigentlich gar nichts.

Wenn Ihre Bank z.B. eine E-Mail mit Ihrem Namen und der Anweisung erhält, Ihre sämtliche Guthaben auf ein Nummernkonto auf den Bahamas zu überweisen, wird sie sich hoffentlich weigern - E-Mail-Adresse hin oder her. Eine E-Mail-Adresse besagt überhaupt nichts über die Identität des Ansenders.

Wenn Sie nur einen kleinen Kreis von Korrespondenzpartnern haben, ist die Sache mit der Identität schnell geregelt: Sie prüfen den Fingerabdruck des Verdächtigen.

Jeder öffentliche Schlüssel trägt eine einmalige Kennzeichnung, die ihn zweifelsfrei identifiziert; wie ein Fingerabdruck einen Menschen. Deshalb bezeichnet man diese Kennzeichnung auch als "Fingerprint".

Wenn Sie einen Schlüssel im GNU Privacy Assistant anklicken, sehen Sie den Fingerprint:



Wie gesagt - der Fingerprint identifiziert den Schlüssel und seinen Besitzer eindeutig.

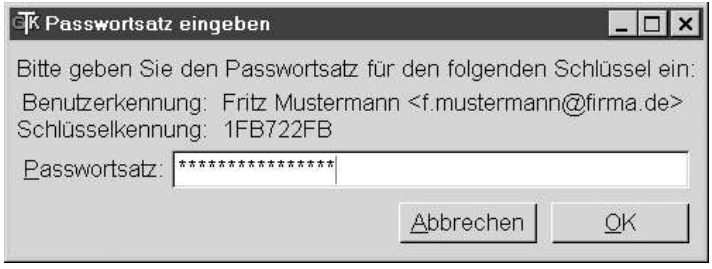
Rufen Sie Ihren Korrespondenzpartner einfach an, und lassen Sie sich von ihm den Fingerprint seines Schlüssels vorlesen. Wenn die Angaben mit dem Ihnen vorliegenden Schlüssel übereinstimmen, haben Sie eindeutig den richtigen Schlüssel.

Natürlich können Sie sich auch persönlich mit dem Eigentümer des Schlüssels treffen oder auf jedem anderen Wege mit ihm kommunizieren, solange Sie ganz sicher sind, daß Schlüssel und Eigentümer zusammengehören.

Nachdem Sie sich "per Fingerabdruck" von der Echtheit des öffentlichen Schlüssel überzeugt haben, sollten Sie ihn signieren. Damit teilen Sie anderen GnuPP-Benutzern mit, daß Sie diesen Schlüssel für echt halten: Sie übernehmen so etwas wie die "Patenschaft" über diesen Schlüssel und erhöhen das allgemeine Vertrauen in seine Echtheit.

Klicken Sie dazu den betreffenden Schlüssel an und wählen Sie dann "Signieren" aus der GPA-Menüleiste. Klicken Sie im nun folgenden Hinweis nur dann auf [Ja], wenn Sie hundertprozentig sicher sind, den richtigen Schlüssel zu signieren.

Geben Sie nun Ihren Passwort-Satz ein:



und klicken Sie auf [OK]. Damit haben Sie mit Ihrem geheimen Schlüssel die Echtheit des Schlüssels bestätigt.

Da -wie Sie wissen - geheimer und öffentlicher Schlüssel untrennbar zusammengehören, kann jedermann mit Hilfe Ihres öffentlichen Schlüssels überprüfen, daß diese Signatur von Ihnen stammt und daß der Schlüssel nicht verändert wurde, also authentisch ist. Damit ist für einen Dritten - wenn auch indirekt - ein gewisses Vertrauen in die Echtheit und Gültigkeit des signierten Schlüssels gegeben.

Das Netz des Vertrauens

So entsteht - auch über den Kreis von GnuPP-Benutzern Ihrer täglichen Korrespondenz hinaus - ein „Netz des Vertrauens", bei dem Sie nicht mehr zwangsläufig darauf angewiesen sind, einen Schlüssel direkt zu prüfen.

Natürlich steigt das Vertrauen in die Gültigkeit eines Schlüssels, wenn mehrere Leute ihn signieren. Ihr eigener öffentlicher Schlüssel wird im Laufe der Zeit die Signatur vieler anderer GnuPG-Benutzer tragen. Damit können immer mehr Menschen darauf vertrauen, dass dieser öffentliche Schlüssel wirklich Ihnen und niemandem sonst gehört.

Wenn man dieses „Web of Trust" weiterspinnt, entsteht eine flexible Beglaubigungs-Infrastruktur.

Eine einzige Möglichkeit ist denkbar, mit dem man diese Schlüsselprüfung aushebeln kann: jemand schiebt Ihnen einen falschen öffentlichen Schlüssel unter. Also einen Schlüssel, der vorgibt, von X zu stammen, in Wirklichhkeit aber von Y ausgetauscht wurde. Wenn ein solcher gefälschter Schlüssel signiert wird, hat das "Netz des Vertrauens" natürlich ein Loch. Deshalb ist es so wichtig, sich zu vergewissern, ob ein öffentlicher Schlüssel, wirklich zu der Person gehört, der er zu gehören vorgibt.

Was aber, wenn eine Bank oder Behörde überprüfen möchte, ob die Schlüssel ihrer Kunden echt sind? Alle anzurufen, kann hier sicher nicht die Lösung sein...

Zertifizierungsinstanzen

Hier braucht man eine „übergeordnete" Instanz, der alle Benutzer vertrauen können. Sie überprüfen ja auch nicht persönlich den Personalausweis eines Unbekannten durch einen Anruf beim Einwohnermeldeamt, sondern vertrauen darauf, daß die ausstellende Behörde diese Überprüfung korrekt durchgeführt und beglaubigt hat.

Solche Zertifizierungsinstanzen gibt es auch bei der Public Key-Verschlüsselung. In Deutschland bietet unter anderem z.B. die Zeitschrift c't schon lange einen solchen Dienst kostenlos an, ebenso wie viele Universitäten.

Wenn man also einen öffentlichen Schlüssel erhält, dem eine Zertifizierungsstelle per Signatur seine Echtheit bestätigt, kann man sich darauf verlassen.

Derartige Beglaubigungsinstanzen oder "Trust Center" sind auch bei anderen Verschlüsselungssystemen vorgesehen, allerdings sind sie hierarchisch strukturiert: es gibt eine "Oberste Beglaubigungsinstanz", die "Unterinstanzen" mit dem Recht zur Beglaubigung besitzt.

Am besten ist diese Infrastruktur mit einem Siegel vergleichbar: die Plakette auf Ihrem Autonummernschild kann Ihnen nur eine dazu berichtigte Institution geben, die die Befugnis dazu widerum von einer übergeordneten Stelle erhalten hat.

Mit der hierarchischen Zertifizierungs-Infrastruktur entspricht dieses Modell natürlich wesentlich besser den Bedürfnissen staatlicher und behördlicher Instanzen als das lose, auf gegenseitigem Vertrauen beruhende „Web of Trust" der GnuPG- und PGP-Modelle. Der Kern der Beglaubigung selbst ist allerdings völlig identisch: wenn man in GnuPP zusätzlich eine hierarchische Zertifizierungs-Struktur einbauen würde, dann würde auch GnuPP dem strengen Signaturgesetz der Bundesrepublik entsprechen.

Wenn Sie sich weiter für dieses Thema interessieren (das zum Zeitpunkt der Arbeit an dieser GnuPP-Ausgabe gerade in Bewegung ist), dann können Sie sich an der Quelle informieren: die Website „Sicherheit im Internet" des Bundesministeriums für Wirtschaft und Technologie (http://www.sicherheit-im-internet.de) hält Sie über dieses und viele andere Themen aktuell auf dem Laufenden.

Eine weitere exzellente, mehr technische Informationsquelle zum Thema der Beglaubigungs-Infrastrukturen bietet das Original-GnuPG-Handbuch, das Sie ebenfalls im Internet finden (http://www.gnupg.org/gph/de/manual).

10. E-Mails signieren

Ganz am Anfang dieses Handbuchs haben wir die E-Mail-Kommunikation mit dem Versenden einer Postkarte verglichen. Erst die Verschlüsselung macht daraus einen Brief mit verschlossenem Umschlag, den nicht mehr jedermann lesen kann.

GnuPP bietet zusätzlich zur kompletten Verschlüsselung einer E-Mail noch eine weitere Möglichkeit:
man kann seine E-Mail signieren, das heisst mit einer elektronischen Unterschrift versehen. Der Text ist dann zwar noch für jeden lesbar, aber der Empfänger kann sicher sein, daß die E-Mail unterwegs nicht manipuliert oder verändert wurde.

Außerdem garantiert die Signatur dem Empfänger, dass die Nachricht auch tatsächlich vom Absender stammt. Und: wenn man mit jemandem korrespondiert, dessen öffentlichen Schlüssel man – aus welchem Grund auch immer - nicht hat, kann man so die Nachricht wenigstens mit dem eigenen privaten Schlüssel „versiegeln".

Verwechseln Sie diese elektronische Signatur nicht mit den E-Mail-„Signaturen", die man unter eine E-Mail setzt und die zum Beispiel Ihre Telefonnummer, Ihre Adresse und Ihre Webseite enthalten.

Während diese E-Mail-Signaturen einfach nur als eine Art Visitenkarte fungieren, schützt die elektronische Signatur Ihre E-Mail vor Manipulationen und bestätigt den Absender.

Übrigens ist diese elektronische Unterschrift auch nicht mit der offiziellen digitalen Signatur gleichzusetzen, wie sie im Signaturgesetz vom 22.Mai 2001 in Kraft getreten ist. Für die private oder berufliche E-Mail-Kommunikation erfüllt sie allerdings genau denselben Zweck.

Signieren mit dem Geheimschlüssel
Tatsächlich ist die Signierung einer E-Mail noch einfacher als die Verschüsselung:

Wie im „Einsteiger-Handbuch" im Kapitel 9, „Sie verschlüsseln eine E-Mail", besprochen, schreiben Sie Ihre Nachricht und kopieren sie mit dem Menübefehl „Kopieren" oder mit dem Tastaturkürzel Strg-C in die Zwischenablage (Clipboard) Ihres Rechners.

Sie können nun entscheiden, ob Sie eine völlig unverschlüsselte, eine signierte oder eine komplett verschlüsselte E-Mail versenden wollen – je nachdem, wie wichtig und schutzbedürftig der Inhalt ist.

Dann öffnen Sie WinPT mit der rechten Maustaste aus der Windows-Taskleiste und klicken in der WinPT-Befehlsleiste auf [Sign clipboard]
Anders als beim Verschlüsseln öffnet sich daraufhin ein Fenster mit Ihrem eigenen Schlüssel. Denn:

Signieren können Sie nur mit Ihrem eigenen geheimen Schlüssel.

Logisch, denn nur Ihr eigener Schlüssel bestätigt Ihre Identität. Der Korrespondenzpartner kann nun mit Ihrem öffentlichen Schlüssel, den er bereits hat oder sich besorgen kann, Ihre Identität überprüfen. Denn nur Ihr Geheimschlüssel passt ja zu Ihrem öffentlichen Schlüssel.

Klicken Sie also auf Ihren eigenen Schlüssel und bestätigen Sie mit [OK]. Im folgenden Fenster geben Sie Ihren geheimen Passwort-Satz einund bestätigen Sie wieder mit [OK]. GnuPP meldet [Finished], wenn der Text signiert ist. Jetzt müssen Sie Ihren signierten Text nur noch in Ihr E-Mail- oder Textprogramm einfügen (Menübefehl „Einfügen" oder Crtl + V).

Ihre Nachricht ist nun am Anfang und Ende von einer Signatur eingerahmt

```
---BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
Hallo Frau Beispiel,

ich wollte nur kurz bestätigen, dafl ich
mit Ihren Bedingungen einverstanden bin.
Wir besprechen alle Einzelheiten morgen
im Büro.
Mit frdl. Gruss,
Fritz Mustermann

-----BEGIN PGP SIGNATURE-----
Version: GnuPG v1.0.6 (MingW32) - WinPT
0.4.0
Comment: Weitere Infos: siehe
http://www.gnupg.org
iEYEARECAAYFAjxeqy0ACgkQcwePex+3Ivs79wC
fW8uytRsEXgzCrfPnjGrDDtb7
QZIAn17B8l8gFQ3WIUUDCMfA5cQajHcm
=O6lY
-----END PGP SIGNATURE-----
```

Wenn Frau Beispiel diese E-Mail erhält, kann sie sicher sein,

1. daß die Nachricht von Herrn Mustermann stammt
2. daß sie nicht verändert wurde

Hätte zum Bespiel jemand das „einverstanden" in dem obigen Beispiel zu „nicht einverstanden" verändert, wäre die Signatur „gebrochen", daß heißt, die E-Mail wäre mit dem Vermerk „ Bad signature" oder „Überprüfung fehlgeschlagen" beim Empfänger eingetroffen.

Andere Gründe für eine gebrochene Signatur

Es gibt aber noch zwei weitere Gründe, die zu einem Bruch der Signatur führen können. Wenn Sie eine E-Mail mit dem Vermerk „Bad signature" oder „Überprüfung fehlgeschlagen" erhalten, ist das ein Warnsignal, muß aber nicht zwangsläufig bedeuten, daß Ihre E-Mail manipuliert wurde.

1. aufgrund der technischen Gegebenheiten ist es nicht auszuschließen, daß die E-Mail durch eine fehlerhafte Übertragung über das Internet verändert wurde.
2. das E-Mail-Programm des Absenders kann falsch eingestellt sein. Wenn man eine signierte E-Mail verschickt, sollte man unbedingt darauf achten, daß im E-Mail-Programm alle Optionen ausgeschaltet sind, die die E-Mail schon beim Versand verändern. Dazu zählen „Quoted Printable", „HTML-Mails" und „Word Wrap".

   „Quoted Printable" bezeichnet die Umsetzung von Umlauten in andere Zeichen, „Word Wrap" den Umbruch von Zeilen in der E-Mail. Beides verändert natürlich die E-Mail und „bricht" die Signatur, obwohll niemand sie willentlich verändert hat. Bei Outlook Express beispielsweise muß diese Option unter „Extras/Optionen/Senden/NurText-Einstellungen/Textkodierung" mit „Keine" aktiviert sein, wie es auch standardmäßig voreingestellt ist.

   Häufig ist sind falsche Einstellungen am E-Mail-Programm der Grund für eine gebrochene Signatur. In beiden Fällen sollte man die E-Mail erneut anfordern.

Dateien signieren

Nicht nur E-Mails, auch Dateien – z.B. ein Word-Dokument - kann man signieren, bevor man sie per E-Mail verschickt oder per Diskette weitergibt. Auch dabei kommt es nicht vorrangig auf die Geheimhaltung, sondern auf die Unverändertheit der Datei an.

Diese Funktion finden Sie unter dem Menüpunkt [Fenster/ Dateiverwaltung] des GNU Privacy Assistant: wählen Sie hier aus dem Menü [Datei] den Unterpunkt [Signieren].

Das Signieren einer Binärdatei unterscheidet sich etwas von dem Signieren einer E-Mail: Wenn man die Datei ausgewählt hat und auf 'Signieren' klickt, öffnet sich ein Fenster 'Datei signieren'. Die ersten beiden Menüpunkte [signieren und komprimieren] und [signieren, nicht komprimieren] erzeugen eine sogenannte Inline-Signatur und werden nur sehr selten benötigt.

Der dritte Punkt [Signatur in separater Datei] erzeugt eine signierte Datei, wenn man den eigenen geheimen Schlüssel in dem Feld „Signieren als" auswählt. Eine Datei mit der Endung .sig wird erzeugt, der im selben Verzeichnis wie die Originaldatei gespeichert wird.

Zum Überprüfen müssen die Original- und die signierte Datei im selben Verzeichnis liegen. Man öffnet die signierte Datei in WinPT und – wenn sie unverändert ist -, wird 'Signatur gültig' angezeigt. Selbst wenn ein Zeichen hinzugefügt und dann wieder gelöscht wurde, ist die Signatur ungültig.

Verschlüsseln und signieren

Normalerweise verschlüsselt man eine Nachricht mit dem öffentlichen Schlüssel des Korrespondenzpartners, der ihn mit seinem privaten Schlüssel entschlüsselt.

Die umgekehrte Möglichkeit – man würde mit dem privaten Schlüssel verschlüsseln -, macht keinen Sinn, weil alle Welt den dazugehörigen öffentlichen Schlüssel kennt und die Nachricht damit entschlüsseln könnte.

Aber diese Möglichkeit bestätigt eindeutig die Urheberschaft – denn wenn jemand die Nachricht mit Ihrem öffentlichen Schlüssel öffnen kann, kann sie nur von Ihrem privaten Schlüssel kodiert worden sein. Und zum dem dürfen ja nur Sie selbst Zugang haben.

Wenn man ganz sicher gehen will, kann man beide Möglichkeiten kombinieren, also die E-Mail verschlüsseln und signieren:

1. Man signiert die Botschaft mit seinem eigenen geheimen Schlüssel. Damit ist die Urheberschaft nachweisbar.
2. dann verschlüsselt man den Text mit dem öffentlichen Schlüssel des Korrespondenzpartners.

Damit hat die Botschaft sozusagen zwei Briefumschläge:

1. einen schwächeren Innenumschlag (die Signatur mit dem eigenen privaten Schlüssel) und
2. einen soliden äußeren Umschlag (die Verschlüsselung mit dem öffentlichen Schlüssel des Korrespondenzpartners).

Der Briefpartner öffnet die äußere, starke Hülle mit seinem eigenen geheimen Schlüssel. Hiermit ist die Geheimhaltung gewährleistet, denn nur dieser Schlüssel kann den Text dekodieren. Die innere, schwächere Hülle öffnet er mit Ihrem öffentlichen Schlüssel und hat den Beweis Ihrer Urheberschaft, denn wenn Ihr öffentlicher Schlüssel passt, kann er nur mit Ihrem Geheimschlüssel kodiert worden sein.
Sehr trickreich und – wenn man ein wenig darüber nachdenkt – auch ganz einfach.

11. Dateianhänge verschlüsseln

Was tun, wenn Sie zusammen mit Ihrer E-Mail eine Datei versenden und diese ebenfalls verschlüsseln wollen? Die Verschlüsselung, wie wir sie in Kapitel 7 des „Einsteiger-Handbuch" erklärt haben, erfasst nur den Text der E-Mail, nicht aber eine gleichzeitig versandte, angehängte Datei.

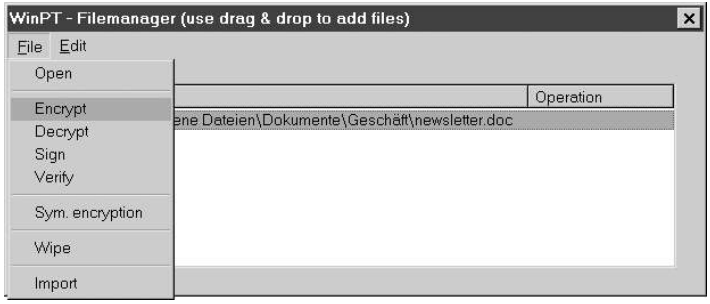
Ganz einfach: Sie verschlüsseln den Anhang getrennt und hängen ihn dann in verschlüsseltem Zustand an die E-Mail an.

Und zwar so:

Öffnen Sie WinPT mit dem rechten Mausklick aus der Windows-Taskleiste , wie schon beim Entschlüsseln von E-Mails in Kapitel 5 des Einsteigerhandbuchs.

Klicken Sie nun im WinPT-Menü auf [Dateimanager]. Sie sehen nun diese Eingabefläche, auf die Sie die zu verschlüsselnde Datei einfach per Drag&Drop ziehen.

Klicken Sie die Datei an und wählen Sie aus dem Menü „File“ den Unterpunkt „Encrypt“



Nun geht alles wie gewohnt: es öffnet sich der WinPT-„Schlüsselbund“, Sie wählen den öffentlichen Schlüssel Ihres Korrespondenzpartners an und klicken auf [OK].

Die Datei wird verschlüsselt und mit der Endung .gpg im gleichen Ordner abgelegt wie die Originaldatei. Nun kann die verschlüsselte Datei wie jede andere als Attachment an eine E-Mail angehängt werden.

Selbstverständlich können Sie mit dieser Funktion auch Dateien verschlüsseln, die Sie nicht per E-Mail versenden wollen.

12. Im- und Export eines geheimen Schlüssels

Im Einsteigerhandbuch haben wir in den Kapiteln
5, 6 und 8 den Im- und Export eines öffentlichen Schlüssels besprochen. Wir haben Ihren eigenen öffentlichen Schlüssel exportiert, um ihn zu veröffentlichen, und wir haben den öffentlichen Schlüssel Ihres Korrespondenzpartners importiert und „am Schlüsselbund" befestigt.

Import eines PGP-Schlüssels

Dabei ging es stets um den öffentlichen Schlüssel. Es gibt aber auch hin und wieder die Notwendigkeit, einen geheimen Schlüssel zu im- oder exportieren. Wenn Sie zum Beispiel einen bereits vorhandenen PGP-Schlüssel mit GnuPP weiterbenutzen wollen, müssen Sie ihn importieren. Oder wenn Sie GnuPP von einem anderen Rechner aus benutzen wollen, muß ebenfalls zunächst der gesamte Schlüssel dorthin transferiert werden – der öffentliche und der private Schlüssel.

Wir gehen hier von der zur Zeit aktuellen PGP-Version 7 aus, in allen anderen ist der Vorgang ähnlich.

Zunächst speichern Sie beide PGP-Schlüsselteile ab. Dazu müssen Sie in „PGPkeys" Ihren Schlüssel anklicken und „Keys/ Export" anwählen. Auf dem Dateirequester „Export Key to File" sehen Sie unten links eine Checkbox „Include Private Keys", den Sie anklicken und mit einem Häkchen versehen müssen. PGP speichert beide Schlüsselteile in eine Datei ab, die Sie entsprechend benennen, zum Beispiel „meinKey.asc". Öffnen Sie diese Datei mit einem Editor: beide Schlüsselteile sind untereinander gespeichert, zuerst der private, dann der öffentliche Schlüsselteil. Speichern Sie beide Schlüsselteile in zwei getrennten ascii-Dateien ab, zum Beispiel unter den Namen „meinSecKey.asc" und „meinPubKey.asc".

Öffnen Sie nun GnuPP und importieren zunächst den öffentlichen Schlüssel, in unserem Beispiel also „meinPubKey.asc". Diese Schlüsselhälfte wird sofort in der Schlüsselverwaltung sichtbar.
Dann importieren Sie den geheimen Schlüssel, in unserem Beispiel „meinSecKey.asc". Der Schlüssel wird nun als komplettes Icon mit zwei Schlüsselhälften dargestellt.

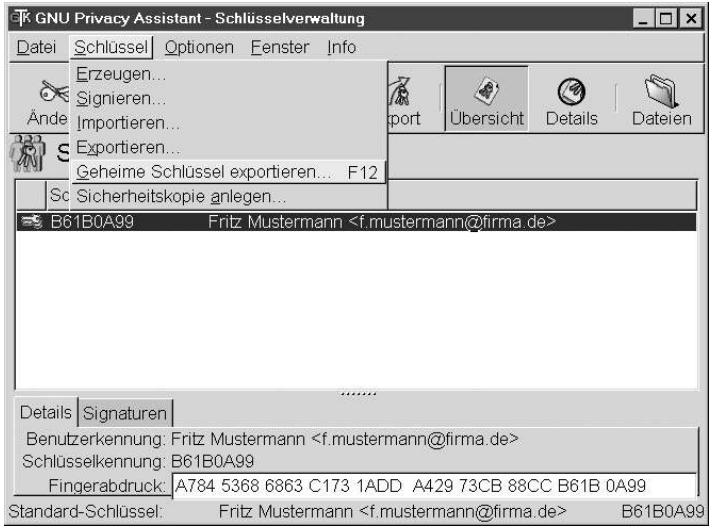
Damit haben Sie einen PGP-Schlüssel erfolgreich in GnuPP importiert und können ihn dort genau wie einen normalen GnuPG-Schlüssel benutzen.

Export eines GnuPG-Schlüssels

Immer wenn Sie einen GnuPG-Schlüssel auf einen anderen Rechner transferieren oder auf einer anderen Festplattenpartition bzw. einer Sicherungsdiskette speichern wollen, müssem Sie ihn exportieren. Dazu klicken Sie in der GPA-Schlüsselverwaltung den Schlüssel an.

Zuerst exportieren Sie mit dem Icon [Export] oder dem Menüpunkt [Schlüssel/Exportieren] den öffentlichen Schlüssel in eine Datei, die Sie entsprechend benennen, zum Beispiel „meinPubKey.asc".

Dann exportieren Sie über den Menüpunkt [Schlüssel/Geheime Schlüssel exportieren] den geheimen Schlüssel und benennen ihn wieder entsprechend, zum Beispiel „meinSecKey.asc".



Import eines GnuPG-Schlüssels

Beim Import, also zum Beispiel auf einem anderen Computer, gehen Sie entsprechend vor:

zunächst öffnen Sie mit GnuPP den öffentlichen Schlüssel, in unserem Beispiel also „meinPubKey.asc". Diese Schlüsselhälfte wird sofort in der Schlüsselverwaltung sichtbar.

dann importieren Sie den geheimen Schlüssel, in unserem Beispiel „meinSecKey.asc". Der Schlüssel wird nun als komplettes Icon mit zwei Schlüsselhälften dargestellt.

Damit haben Sie erfolgreich einen GnuPG-Schlüssel exportiert und wieder importiert.

13. Warum GnuPP nicht zu knacken ist

Jedenfalls nicht in den nächsten paar Milliarden Jahren

In jedem Beispiel dieses Handbuchs haben Sie gesehen, daß zwischen dem geheimen und dem öffentlichen Schlüsselteil eine geheimnisvolle Verbindung besteht. Nur wenn beide zueinander passen, kann man Geheimbotschaften entschlüsseln.

Das Geheimnis dieser mathematischen Verbindung müssen Sie nicht unbedingt kennen – GnuPP funktioniert auch so. Man diese komplexe mathematische Methode aber auch als Normalsterblicher und Nichtmathematiker verstehen. Sie müssen eigentlich nur einfache Additionen (2+3) und Multiplikationen (5x7) beherrschen. Allerdings in einer ganzen anderen Rechenmethode als der, die Sie im Alltag benutzen.
Es gehört sowohl zur Sicherheitsphilosophie der Public Key-Methode wie auch zum Prinzip der Freien Software, daß es keine Geheimnisse gibt. Letztendlich versteht man auch erst dann wirklich, warum GnuPP sicher ist.

Hier beginnt also sozusagen die Kür nach dem Pflichtteil:

GnuPP und das Geheimnis der großen Zahlen

Kryptografie für Nicht-Mathematiker

Es ist schon oft versucht worden, den RSA-Algorithmus, auf dem GnuPG basiert, zu „knacken", also einen privater Schlüssel zu berechnen, wenn man den öffentlichen Schlüssel kennt. Im Gegensatz zu allen anderen Verschlüsselungsmethoden ist diese Berechnung aber noch nie gelungen. Es ist theoretisch nicht unmöglich, aber praktisch undurchführbar.
Nach heutiger wissenschaftlicher und technischer Kenntnis würde diese Berechnung selbst mit den schnellsten Supercomputern länger dauern, als das Universum vom Urknall bis heute bestanden hat: 10 bis 20 Milliarden Jahre. Ungefähr.

Im Laufe der langen Geschichte der Kryptografie wurden immer bessere und komplexere Verschlüsselungsmethoden entwickelt. Letztendlich wurden sie alle mit den unterschiedlichsten Techniken und Tricks geknackt.

Nur bei GnuPP bzw. der zugrundeliegenden mathematischen Methode versagt jede Hackerkunst. Warum ist das so? Diese Methode hat zwei einzigartige Grundeigenschaften:

erstens beruht sie auf der öffentlichen Bekanntgabe der genauen Details der Verschlüsselungsmethode. Das scheint dem gesunden Menschenverstand zunächst völlig zu widersprechen - Sie haben aber in diesem Handbuch gesehen, warum GnuPP genau deshalb so sicher ist.

zweitens basiert GnuPP auf einer mathematischen Technik, die erwiesenermaßen so komplex ist, daß sie die Methode unüberwindlich gegen alle heute durchführbaren Angriffe macht, selbst wenn enorme Zeit- und Rechenressourcen aufgewendet werden. Und trotzdem können Sie diese Methode auf Ihrem PC benutzen.

Hier erfahren Sie, wie diese Methode funktioniert. Nicht in allen Einzelheiten – das würde den Rahmen dieser Anleitung bei weitem sprengen -, aber doch so, daß Sie bei etwas Mitrechnen selbst mathematisch korrekt ver- und entschlüsseln können und dabei das „Geheimnis der großen Zahlen" entdecken.

Man kann diese komplexe mathematische Methode auch als Normalsterblicher und Nichtmathematiker verstehen. Sie müssen nur einfache Additionen und Multiplikationen beherrschen. Wie gesagt: hier beginnt der Kür-Teil, und bei der Kür geht es immer etwas mehr zur Sache als im „Pflichtprogramm".
Letztendlich versteht man dann aber, warum GnuPP wirklich so sicher ist.

Eine Begriffsklärung vorneweg:

ein Algorithmus ist eine mathematische Prozedur
zur Veränderung oder Transformation von Daten oder Informationen.

Arithmetik ist die Methode,
nach der wir Zahlen addieren und multiplizieren

Die Verschlüsselung mit GnuPP basiert auf dem sogenannten RSA-Algorithmus. RSA steht für Rivest, Shamir, und Adelman, die Erfinder des Algorithmus. Dieser Algorithmus verwendet einen Typ der Arithmetik, die Rechnen mit Restklassen oder „Modulo-Arithmetik" heisst.

Das Rechnen mit Restklassen

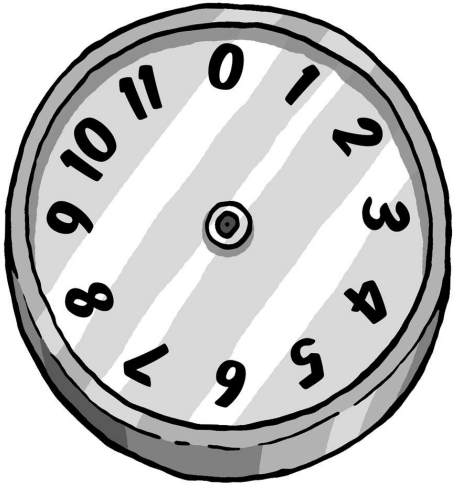
Wenn man mit Restklassen rechnet, so bedeutet das, dass man nur mit dem „Rest" rechnet, der nach einer Teilung durch eine bestimmte Zahl übrigbleibt. Diese Zahl, durch die geteilt wird, nennt man den „Modul" oder die „Modulzahl". Wenn wir beispielsweise mit dem Teiler oder der Modulzahl 5 rechnen, sagen wir auch, „wir rechnen modulo 5".

Wie das Rechnen mit Restklassen – auch Modulo-Arithmetik genannt - funktioniert, kann man sich gut klarmachen, wenn man sich das Zifferblatt einer Uhr vorstellt:

Diese Uhr ist ein Beispiel für das Rechnen mit modulo 12 (der Teiler ist also 12) - eine Uhr mit einem normalen Zifferblatt, allerdings mit einer 0 anstelle der 12. Wir können damit Modulo-Arithmetik betreiben, indem wir einfach den gedachten Zeiger bewegen.

Um beispielsweise 3 + 2 zu rechnen, beginnen wir bei der Ziffer 2 und drehen den Zeiger um 3 Striche weiter (oder wir starten bei der 3 und drehen 2 Striche weiter, was natürlich auf dasselbe hinausläuft). Das Ergebnis ist 5.

Zählt man auf diese Weise 7 + 8 zusammen, erhält man 3. Denn 3 ist der Rest, wenn man 15 (also 7 + 8) durch 12 teilt.
Um 5 mit 7 zu multiplizieren, beginnt man bei 0 und dreht 7 mal jeweils um 5 Striche weiter (oder auch bei 0 beginnend 5 mal um 7 Striche). In beiden Fällen bleibt der Zeiger bei 11 stehen. Denn 11 ist der Rest, wenn 35 (also 7 x 5) durch 12 geteilt wird.



Beim Rechnen mit Restklassen addieren und teilen wir Zahlen also nach den normalen Regeln der Alltagsarithmetik, verwenden dabei jedoch immer nur den Rest nach der Teilung. Um anzuzeigen, dass wir nach den Regeln der Modulo-Arithmetik und nicht nach denen der üblichen Arithmetik rechnen, schreibt man den Modul (Sie wissen schon - den Teiler) in Klammern dazu. Man sagt dann zum Beispiel „4 modulo 5", schreibt aber kurz „4 (mod 5)".
Bei Modulo-5 zum Beispiel hat man dann eine Uhr, auf deren Zifferblatt es nur die 0, 1, 2, 3 und 4 gibt. Also:

4 (mod 5) + 3 (mod 5) = 7 (mod 5) = 2 (mod 5)

Anders ausgedrückt, ist in der Modulo 5-Arithmetik das Ergebnis aus 4 plus 3 gleich 2. Wir können also auch schreiben:

9 (mod 5) + 7 (mod 5) = 16 (mod 5) = 1 (mod 5)
Wir sehen auch, dass es egal ist, in welcher Reihenfolge wir vorgehen, weil wir nämlich auch schreiben können:

9 (mod 5) + 7 (mod 5) = 4(mod 5) + 2 (mod 5) = 6 (mod 5) = 1 (mod 5)
Denn 4 ist dasselbe wie 9, und 2 dasselbe wie 7, da wir uns ja nur für den jeweiligen Rest nach der Teilung durch 5 interessieren. Daran wird deutlich, dass wir bei dieser Art der Arithmetik jederzeit 5 oder ein Vielfaches von 5, wie 10, 15 und so weiter nehmen können, und das Ergebnis stets dasselbe ist.

Das funktioniert auch beim Multiplizieren (Malnehmen).
Ein Beispiel:

4 (mod 5) x 2 (mod 5) = 8 (mod 5) = 3 (mod 5)

Ebenso können wir schreiben:

9 (mod5) x 7 (mod 5) = 63 (mod 5) = 3 (mod 5),
da wir einfach 60, also 5 x 12, abziehen können.

Man könnte aber auch schreiben:

9 (mod 5) x 7 (mod 5) = 4 (mod 5) x 2 (mod 5) = 8 (mod 5) = 3 (mod 5)

denn 4 entspricht 9, und 2 entspricht 7, wenn wir nur den Rest nach Teilung durch 5 betrachten.

Widerum stellen wir fest, dass es egal ist, wenn wir das Vielfache von 5 einfach weglassen.

Da dadurch alles einfacher wird, machen wir das, bevor wir Zahlen addieren oder multiplizieren. Das bedeutet, dass wir uns lediglich um die Zahlen 0, 1, 2, 3 und 4 kümmern müssen, wenn wir mit der Modulo5-Arithmetik rechnen. Denn wir können ja alles, was durch 5 teilbar ist, weglassen.
Dazu noch drei Beispiele:

5 (mod 11) x 3 (mod 11) = 15 (mod 11) = 4 (mod 11)

2 (mod 7) x 4 (mod 7) = 1 (mod 7)

13 (mod 17) x 11 (mod 17) = 7 (mod 17)

Das letzte Beispiel wird klar, wenn man sich bedenkt, dass in normaler Arithmetik gerechnet
13 x 11 = 143 und 143 = 8 x 17 +7 ist.

RSA-Algorithmus und Rechnen mit Restklassen

Computer speichern Buchstaben als Zahlen. Alle Buchstaben und Symbole auf der Computertastatur werden in Wirklichkeit als Zahlen gespeichert, die zwischen zwischen 0 und 255 liegen.

Wir können also eine Nachricht auch in eine Zahlenfolge umwandeln. Nach welcher Methode (oder Algorithmus), wird im nächsten Abschnitt beschrieben. Darin stellen wir Ihnen die Methode vor, nach der die Verschlüsselung mit GnuPP funktioniert: den RSA-Algorithmus. Dieser Algorithmus wandelt eine Zahlenfolge (die ja eine Nachricht darstellen kann) so in eine andere Zahlenfolge um (Transformation), dass die Nachricht dabei verschlüsselt wird. Wenn man dabei nach dem richtigen Verfahren vorgeht, wird die Nachricht sicher kodiert und kann nur noch vom rechtmässigen Empfänger dekodiert werden.

Das sind die Grundlagen des RSA-Algorithmus:

Sie selbst haben bei der Installation von GnuPP während der Eingabe Ihres Passwort-Satzes zwei große Primzahlen erzeugt, ohne es zu bemerken. Nur Sie – oder in der Praxis Ihr Computer – kennen diese beiden Primzahlen, und Sie müssen für ihre Geheimhaltung sorgen.

Es werden nun drei Zahlen erzeugt:

Die erste Zahl
ist das Ergebnis der Multiplikation der beiden Primzahlen, also ihr Produkt

Die zweite Zahl
ist eine weitere große, zufällig gewählte Primzahl. Sie ist der öffentliche Subkey.

Die dritte Zahl
wird in einem komplizierten Verfahren errechnet aus

dem öffentlichem Subkey
(der zweiten Zahl)

dem Modul,
d.h., der ersten Primzahl minus 1
mal der zweiten Primzahl minus 1

Die erste und die zweite Zahl werden veröffentlicht – das ist Ihr öffentlicher Schlüssel. Beide werden dazu benutzt, Nachrichten zu verschlüsseln. Die dritte Zahl muss von Ihnen geheimgehalten werden – es ist Ihr geheimer Schlüssel.
Wenn eine verschlüsselte Nachricht empfangen wird, kann sie entschlüsselt werden mit Hilfe der ersten und der dritten Zahl. Nur der Empfänger kennt beide Subkeys – seinen öffentlichen und seinen geheimen Schlüssel. Der Rest der Welt kennt nur den öffentlichen Schlüssel. Aufgrund der Beziehung der beiden Subkeys ist es unmöglich, den geheimen Subkey zu errechnen (und damit die Botschaft zu entschlüsseln), wenn man die beiden großen Primzahlen nicht kennt.

RSA-Verschlüsselung mit kleinen Zahlen

Wir verwenden hier erst einmal kleine Primzahlen, um deutlich zu machen, wie die Methode funktioniert. In der Praxis verwendet man jedoch viel größere Primzahlen, die aus –zig Ziffern bestehen.

Nehmen wir die Primzahlen 7 und 11. Damit verschlüsseln wir Zahlen – oder Buchstaben, was für den Computer dasselbe ist - nach dem RSA-Algorithmus.

Und zwar erzeugen wir zunächst den öffentlichen Schlüssel

Die erste Zahl
ist 77, nämlich das Ergebnis der Multiplikation der beiden Primzahlen, 7 und 11. 77 dient uns im weiteren Verlauf als Modulzahl zur Ver- und Entschlüsselung.

Die zweite Zahl
ist die zufällig gewählte Primzahl 13

Die dritte Zahl,
der geheime Schlüssel, wird in einem komplizierten Verfahren errechnet, das wir jetzt erklären:

zunächst ziehen wir von unseren Primzahlen 7 und 11 jeweils die Zahl 1 ab (also 7-1 und 11-1) und multiplizieren die beiden resultierenden Zahlen miteinander. In unserem Beispiel ergibt das 60: (7-1) x (11-1) = 60. 60 ist unsere Modulzahl für die weiterführende Rechnung.

Wir suchen jetzt eine Zahl, die multipliziert mit dem öffentlichen Schlüssel die Zahl 1 ergibt, wenn man mit dem Modul 60 rechnet:

13 (mod 60) x ? (mod 60) = 1 (mod 60)

Die einzige Zahl, die diese Bedingung erfüllt, ist 37, denn

13 (mod 60) x 37 (mod 60) = 481 (mod 60) = 1 (mod 60)

37 ist die einzige Zahl, die multipliziert mit 13 die Zahl 1 ergibt, wenn man mit dem Modul 60 rechnet.

Wir verschlüsseln mit dem öffentlichen Schlüssel eine Nachricht

Nun zerlegen wir die Nachricht in eine Folge von Zahlen zwischen 0 und 76, also 77 Zahlen, denn sowohl Verschlüsselung als auch Entschlüsselung verwenden den Modul 77 (das Produkt aus den Primzahlen 7 und 11).

Jede einzelne dieser Zahlen wird nun nach der Modulo77-Arithmetik 13 mal mit sich selbst multipliziert. Sie erinnern sich: die 13 ist ja unser öffentlicher Schlüssel.

Nehmen wir ein Beispiel mit der Zahl 2: sie wird in die Zahl 30 umgewandelt, weil 2x2x2x2x2x2x2x2x2x2x2x2x2 = 8192 = 30 (mod 77) sind.

Ein weiteres Beispiel: 75 wird in die Zahl 47 umgewandelt, denn 75 wird 13mal mit sich selbst multipliziert und durch 77 geteilt, so dass der Rest 47 entsteht.
Wenn man eine solche Rechnung für alle Zahlen zwischen 0 und 76 durchführt und die Ergebnisse in eine Tabelle einsetzt, sieht diese so aus:
In der linken Spalte stehen die 10er-Stellen, in der oberen Zeile die 1er-Stellen.

	0	1	2	3	4	5	6	7	8	9
0	0	1	30	38	53	26	62	35	50	58
10	10	11	12	41	49	64	37	73	46	61
20	69	21	22	23	52	60	75	48	7	57
30	72	3	32	33	34	63	71	9	59	18
40	68	6	14	43	44	45	74	5	20	70
50	29	2	17	25	54	55	56	8	16	31
60	4	40	13	28	36	65	66	67	19	27
70	42	15	51	24	39	47	76			

Wir entschlüsseln eine Nachricht mit dem privaten Schlüssel

Um das Beispiel mit der 2 von oben umzukehren, also die Nachricht zu dekodieren, multiplizieren wir 30 (die umgewandelte 2) unter Verwendung der Modulzahl 77 37 mal mit sich selbst. Sie erinnern sich: 37 ist der geheime Schlüssel.

Diese wiederholte Multiplikation ergibt eine Zahl die 2(mod77) ergibt. Das andere Beispiel: die Zahl 47 (mod 77) wird zur Zahl 75 (mod77) dekodiert.

Tabelle 2 zeigt die genaue Zuordnung der 77 Zahlen zwischen 0 und 76.

	0	1	2	3	4	5	6	7	8	9
0	0	1	51	31	60	47	41	28	57	37
10	10	11	12	62	42	71	58	52	39	68
20	48	21	22	23	73	53	5	69	63	50
30	2	59	32	33	34	7	64	16	3	74
40	61	13	70	43	44	45	18	75	27	14
50	8	72	24	4	54	55	56	29	9	38
60	25	19	6	35	15	65	66	67	40	20
70	49	36	30	17	46	26	76			

Um eine Zahl mit Tabelle 2 zu transformieren, gehen wir nach der gleichen Methode vor wie bei Tabelle 1. Ein Beispiel: 60 wird transformiert in die Zahl in Zeile 60 und Spalte 0. Also wird 60 zu 25 transformiert.
Das überrascht nicht, denn wenn wir davon ausgehen, dass wir bei der Umwandlung von 25 mit HIlfe von Tabelle 1 als Ergebnis 60 erhalten, dann sollten wir auch bei der Transformation von 60 mit Hilfe von Tabelle 2 zum Ergebnis 25 gelangen. Dabei haben wir den öffentlichen Schlüssel, 13, zur Umwandlung bzw. Kodierung einer Zahl verwendet, und den geheimen Schlüssel 37, um sie zurückzuwandeln bzw. zu dekodieren. Sowohl für die Verschlüsselung als auch für die Entschlüsselung haben wir uns der Modulo 77-Arithmetik bedient.

Tabelle 2: Zahlentransformation modulo 77,
unter Verwendung des geheimen Schlüssels 37

Zusammenfassung:

Wir haben...

...durch den Computer zwei
zufällige Primzahlen erzeugen lassen;

...daraus das Produkt und
den öffentlichen und den geheimen Subkey gebildet;

...gezeigt, wie man mit dem
öffentlichen Schlüssel Nachrichten verschlüsselt;

...gezeigt, wie man mit dem
geheimen Schlüssel Nachrichten entschlüsselt.
Diese beiden Primzahlen können so gross gewählt werden, dass es unmöglich ist, sie einzig aus dem öffentlich bekannt gemachten Produkt zu ermitteln. Das begründet die Sicherheit des RSA-Algorithmus.

Wir haben gesehen, dass die Rechnerei sogar in diesem einfachen Beispiel recht kompliziert geworden ist. In diesem Fall hat die Person, die den Schlüssel öffentlich gemacht hat, die Zahl 77 und den öffentlichen Schlüssel 13 bekanntgegeben.
Damit kann jedermann dieser Person mit der oben beschriebenen Methode – wie im Beispiel der Tabelle 1 – eine verschlüsselte Zahl oder Zahlenfolge schicken. Der rechtmässige Empfänger der verschlüsselten Zahlenfolge kann diese dann mit Hilfe der Zahl 77 und dem geheimen Schlüssel 37 dekodieren.

In diesem einfachen Beispiel ist die Verschlüsselung natürlich nicht sonderlich sicher. Erstens ist klar, dass 77 das Produkt aus 7 und 11 ist. Zweitens läßt sich, wenn der öffentliche Schlüssel 13 ist, leicht herausfinden, dass der geheime 37 ist.

Folglich kann man den Code in diesem einfachen Beispiel leicht knacken. Der scharfsinnige Leser wird auch bemerkt haben, dass etliche Zahlen, zum Beispiel die Zahl 11 und ihr Vielfaches (also 22, 33 etc.) und die benachbarten Zahlen sich in sich selbst umwandeln.

	0	1	2	3	4	5	6	7	8	9
0	0	1	51	31	60	47	41	28	57	37
10	10	11	12	62	42	71	58	52	39	68
20	48	21	22	23	73	53	5	69	63	50
30	2	59	32	33	34	7	64	16	3	74
40	61	13	70	43	44	45	18	75	27	14
50	8	72	24	4	54	55	56	29	9	38
60	25	19	6	35	15	65	66	67	40	20
70	49	36	30	17	46	26	76			

Das erscheint als ein weiterer Schwachpunkt dieser Verschlüsselungsmethode: man könnte annehmen, dass die Sicherheit des Algorithmus dadurch beeinträchtigt würde. Doch stellen Sie sich nun vor, das Produkt zweier grosser Primzahlen, die auf absolut willkürliche Art und Weise gewählt werden, ergäbe

114,381,625,757,888,867,669,235,779,976,146,612,010,
218,296,721,242,362,562,561,842,935,706,935,245,733,
89,830,597,123,563,958,705,058,989,075,147,599,290,
026,879,543,541

Hier ist überhaupt nicht mehr ersichtlich, welche die beiden zugrunde liegenden Primzahlen sind. Folglich ist es sehr schwierig, aufgrund des öffentlichen Schlüssels den geheimen Schlüssel zu ermitteln. Selbst den schnellsten Computern der Welt würde es gewaltige Probleme bereiten, die beiden Primzahlen zu errechnen.

Man muss die Primzahlen also nur gross genug wählen, damit ihre Berechnung aus dem Produkt so lange dauert, dass alle bekannten Methoden daran scheitern. Außerdem nimmt der Anteil der Zahlen, die in sich selbst transformiert werden – wie wir sie oben in den Tabellen 1 und 2 gefunden haben - stetig ab, je größer die Primzahlen werden. Von Primzahlen in der Grössenordnung, die wir in der Praxis bei der Verschlüsselung verwenden, ist dieser Teil so klein, dass der RSA-Algorithmus davon in keiner Weise beeinträchtigt wird. Je größer die Primzahlen, desto sicherer die Verschlüsselung.
Trotzdem kann ein normaler PC ohne weiteres das Produkt aus den beiden großem Primzahlen bilden. Kein Rechner der Welt dagegen kann aus diesem Produkt wieder die ursprünglichen Primzahlen herausrechnen – jedenfalls nicht in vertretbarer Zeit.

Die Darstellung mit verschiedenen Basiszahlen

Um zu verstehen, wie Nachrichten verschlüsselt werden, sollte man wissen, wie ein Computer Zahlen speichert und vor allem, wie sie zu unterschiedlichen Zahlenbasen dargestellt werden können.

Dazu machen wir uns zunächst mit den Zahlenpotenzen vertraut.

Zwei hoch eins, das man als 2^1 darstellt, ist gleich 2;
zwei hoch drei, dargestellt als 2^3, ist 2x2x2 = 8; zwei hoch zehn, dargestellt als 2^{10}, ist 2x2x2x2x2x2x2x2x2x2 = 1024.

Jede Zahl hoch 0 ist gleich 1, zum Beispiel $2^0 = 1$ und $5^0 = 1$.

Allgemein wird eine potenzierte Zahl so oft mit sich selbst multipliziert, wie es die Hochzahl (=Potenz) angibt.

Das Konzept einer Zahlenbasis veranschaulicht zum Beispiel ein Kilometerzähler im Auto: das rechte Rad zählt nach jedem Kilometer eine Stelle weiter und zwar nach der vertrauten Abfolge der Zahlen

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2

und so weiter. Jedesmal, wenn das rechte Rad wieder 0 erreicht, zählt das Rad links davon eine Stelle hoch. Und jedesmal, wenn dieses zweite Rad die 0 erreicht, erhöht das Rad links davon um eins.....und so weiter.

Das rechte Rad zählt die einzelnen Kilometer. Wenn es eine 8 angezeigt, dann sind dies 8 Kilometer. Das Rad links davon zeigt jeweils die vollen zehn Kilometer an: eine 5 bedeutet 50 Kilometer. Dann folgen die Hunderter: steht dort 7, dann bedeutet dies 700 Kilometer.

Nach dem gleichen Prinzip stellen wir ja auch unsere normalen Zahlen mit den Ziffern 0 bis 9 dar.

„578", zum Beispiel, bedeutet 5 x 100 + 7 x 10 + 8, und dies entspricht 578.

Hier haben wir die „5" stellvertretend für fünfhundert, „7" für siebzig und „8" für acht. In diesem Fall ist die Basis 10, eine für uns vertraute Basis.

Also steht die rechte Ziffer für die Einer der betreffenden Zahl (d.h. sie wird mit 1 multipliziert), die Ziffer links davon steht für die Zehner (d.h. wird mit 10 multipliziert), die nächste Ziffer wiederum für die Hunderter (d.h. sie wird mit 100 multipliziert) und so weiter.
Da wir Zahlen normalerweise zur Basis 10 darstellen, machen wir uns nicht die Mühe, die Basis extra anzugeben. Formal würde man dies bei der Zahl 55 mit der Schreibweise 55_{10} anzeigen, wobei die tiefgestellte Zahl die Basis anzeigt.

Wenn wir nicht zur Basis 10 darstellen, so müssen wir dies mit Hilfe einer solchen tiefgestellten Zahlenbasis anzeigen.

Angenommen, die Anzeige des Kilometerzählers hätte statt der Ziffern 0 bis 9 nur noch 0 bis 7. Das rechte Rädchen würde nach jedem Kilometer um eine Ziffer höher zählen, wobei die Zahlenfolge so aussehen würde:

0, 1, 2, 3, 4, 5, 6, 7, 0, 1, 2, und so weiter.

Unser Tacho zur Basis 8 stellt zum Beispiel folgende Zahl dar:

356

Die 6 auf dem rechte Rädchen zählt einzelne Kilometer, also 6 Kilometer.
Die 5 auf dem Rädchen daneben für 5x8, also 40 Kilometer.
Die 3 links steht für je 64 Kilometer pro Umdrehung, also hier 3x8x8 Kilometer.

So rechnet man also mit Zahlen zur Basis 8. Ein Beispiel: 72_8 bedeutet 7 x 8 + 2, und das ist gleich „58". Bei dieser Art der Darstellung steht die „2" aus der 72 für 2, aber die „7" steht für 7 x 8.

Größere Zahlen werden schrittweise genauso aufgebaut, so dass 453_8 eigentlich 4 x 64 + 5 x 8 + 3 bedeutet, was 299 ergibt.

Bei 453_8 steht die „3" für 3, die „5" für 5 x 8 und die „4" für 4 x 64, wobei sich die „64" wiederum aus 8 x 8 herleitet.

Im angeführten Beispiel werden die Ziffern, von rechts nach links gehend, mit aufsteigenden Potenzen von 8 multipliziert. Die rechte Ziffer wird mit 8 hoch 0 (das ist 1) multipliziert, die links daneben mit 8 hoch 1 (das ist 8), die nächste links davon mit 8 hoch 2 (das ist 64) und so weiter.
Wenn man Zahlen zur Basis 10 darstellt, gibt es keine höhere Ziffer als 9 (also 10 minus 1). Wir verfügen also über keine Ziffer, die 10 oder eine größere Zahl darstellt. Um 10 darzustellen, brauchen wir zwei Ziffern, mit denen wir dann die „10" schreiben können.
Wir haben also nur die Ziffern 0 bis 9.

So ähnlich ist es , wenn wir zur Zahlenbasis 8 rechnen: dann haben wir nur die Ziffern 0 bis 7. Wollen wir zu dieser Basis eine höhere Zahl als sieben darstellen, müssen wir wieder zwei Ziffern verwenden. Zum Beispiel „9" schreibt man als 11_8 , „73" schreibt man als 111_8.

Computer speichern Zahlen als eine Folge von Nullen und Einsen. Man nennt dies Binärsystem oder Rechnen zur Zahlenbasis 2, weil wir nur die Ziffern 0 und 1 verwenden. Stellen Sie sich vor, wir würden die Kilometer mit einem Tachometer zählen, auf dessen Rädchen sich nur zwei Ziffern befinden: 0 und 1.
Die Zahl 101012 zum Beispiel bedeutet im Binärsystem

1x1 + 0x8 +1x4 + 0x2 + 1 = 21.

In der Computerei verwendet man auch Gruppen von acht Binärziffern, das wohlbekannte Byte. Ein Byte kann Werte zwischen 0 - dargestellt als Byte 00000000_2 - und 255 - dargestellt als Byte 11111111_2 - annehmen. Ein Byte stellt also Zahlen zur Basis 256 dar.

Zwei weitere Beispiele:

$10101010_2 = 170$

und

$00000101_2 = 5$.

Da der Computer die Buchstaben, Ziffern und Satzzeichen als Bytes speichert, schauen wir uns an, welche Rolle dabei die Darstellung zur Basis 256 spielt.

Nehmen wir die Silbe "un". Das "u" wird im Computer als 117 gespeichert und das "n" als 110.

Diese Zahlenwerte sind für alle Computer standardisiert und werden ASCII-Code genannt. Um alle Zahlen und Symbole darstellen zu können, benötigen wir auf dem Computer die 256 Zahlen von 0 bis 255.

Wir können also die Silbe "un" durch die Zahl 117x256 + 110 darstellen.
Entsprechend würde man die Buchstabenfolge "und" mit der Zahl 117x65536 + 110x256 + 100 darstellen, denn das "d" wird durch 100 repräsentiert.

Wir haben hier also Zahlen und Symbole, die auf der Computertastatur als normale Zahlen zur Basis 10 stehen, intern durch Zahlen zur Basis 256 repräsentiert.
Entsprechend können wir aus jeder Nachricht eine große Zahl machen. Aus einer langen Nachricht wird also eine gewaltig große Zahl. Und diese sehr große Zahl wollen wir nun nach dem RSA-Algorithmus verschlüsseln.

Wir dürfen allerdings dabei die Zahl, zu der die Nachricht verschlüsselt wird, nicht grösser werden lassen als das Produkt der Primzahlen. Ansonsten bekommen wir Probleme, wie wir gleich noch sehen werden.

Da die folgende Prozedur mehrere Schritte umfasst, fassen wir sie zunächst zusammen und verfolgen dann die Einzelschritte:

1. Die Nachricht aba, cad, aca wandeln wir - wie gesehen - in Zahlen um

2. diese Darstellung zur Basis 4 wandeln wir in eine Darstellung zur Basis 10 um, damit wir zur Verschlüsselung die Tabelle 1 benutzen können, in denen die Zahlen ja auch auf 10er-Basis dargestellt werden.
   Dabei entsteht eine kodierte Nachricht zur Basis 10.

3. Um die Kodierung im Vergleich zum „Klartext" zu erkennen, rechnen wir die zur Basis 10 kodierte Nachricht auf die Basis 4 zurück und wandeln sie dann wieder in eine Buchstabensequenz um.

4. So entsteht aus der Nachricht aba, cad, aca die verschlüsselte Nachricht dbb, ddd, dac.

1. Die Nachricht aba, cad, aca wandeln wir in Zahlen um

Angenommen, wir beschränken uns bei den Nachrichten auf die 4 Buchstaben a, b, c und d. In diesem - wirklich sehr einfachen - Beispiel können wir die vier Buchstaben durch die Zahlenwerte 0, 1, 2 und 3 darstellen, und haben dann

a = 0, b = 1, c = 2 und d = 3.

Wir wollen nun die Nachricht "abacadaca" verschlüsseln. Wir kodieren diese Nachricht mit Hilfe der Primzahlen 7 und 11, mit dem öffentlichen Schlüssel 13 und dem geheimen Schlüssel 37. Dieses Beispiel kennen wir bereits aus dem früheren Kapitel: wir haben damit die Tabellen 1 und 2 konstruiert.

2. diese Darstellung zur Basis 4 wandeln wir in eine Darstellung zur Basis 10 um, damit wir zur Verschlüsselung die Tabelle 1 benutzen können, in denen die Zahlen ja auch zur 10er-Basis dargestellt werden

Weil wir vier Buchstaben für die Nachricht verwenden, rechnen wir zur Basis 4. Für die Rechnung modulo 77 müssen wir die Nachricht in Stücke von je drei Zeichen Länge zerlegen, weil die größte dreiziffrige Zahl zur Basis 4 die 333_4 ist. Zur Basis 10 hat diese Zahl den Wert 63.
Würden wir stattdessen die Nachricht in vier Zeichen lange Stücke zerlegen, würde die Zahl zur Basis 4 den Wert 76 übersteigen und es würden unerwünschte Doppeldeutigkeiten entstehen.
Folglich würde die Nachricht in dreiziffrigen Stücken nun

aba, cad, aca

ergeben. Geben wir den Zeichen nun ihre Zahlenwerte und vergessen dabei nicht, dass die Stücke dreiziffrige Zahlen zur Basis 4 darstellen.

Da wir die Buchstaben durch die Zahlen
a = 0, b = 1, c = 2, d = 3 darstellen, wird die Nachricht zu

010_4, 203_4, 020_4 .

Zur Basis 10 wird diese Nachricht durch die Zahlenfolge 4, 35, 8 dargestellt. Warum? Nehmen wir zum Beispiel das mittlere Stück 203_4:
$3x4^0$, also 3x1, also 3
$0x4^1$, also 0x4, also 0
$2x4^2$, also 2x16, also 32

Dargestellt zur Basis 10 entspricht dies also 3+0+32, und das ist nichts anderes
als 35.

Entsprechend wird unsere Nachricht aba, cad, aca durch die Zahlenfolge 4, 35, 8 zur Basis 10 dargestellt.

3. Jetzt können wir zur Verschlüsselung die Tabelle 1 benutzen, die ja zur Basis 10 berechnet wurde. Diese Tabelle benutzen wir, weil wir mit dem schon bekannten Schlüsselpaar arbeiten wollen. Dabei entsteht eine kodierte Nachricht zur Basis 10.

Zum Verschlüsseln der Nachricht nehmen wir jetzt Tabelle 1 zur Hilfe. Die Nachricht wird nun zu der Zahlenfolge 53, 63, 50 (zur Basis 10).

	0	1	2	3	4	5	6	7	8	9
0	0	1	30	38	53	26	62	35	50	58
10	10	11	12	41	49	64	37	73	46	61
20	69	21	22	23	52	60	75	48	7	57
30	72	3	32	33	34	63	71	9	59	18
40	68	6	14	43	44	45	74	5	20	70
50	29	2	17	25	54	55	56	8	16	31
60	4	40	13	28	36	65	66	67	19	27
70	42	15	51	24	39	47	76			

4. Wiederum zur Basis 4 konvertiert, entsteht die verschlüsselte Nachricht

Wird sie nun wieder zur Basis 4 konvertiert, ergibt die Nachricht nun 311_4, 333_4, 302_4. Konvertiert man diese zu einer Buchstabensequenz, erhält man dbb, ddd, dac, was sich nun erheblich von der ursprünglichen Nachricht unterscheidet.

Man kehrt nun also den Prozeß um und transformiert die Zahlenfolge 53, 63, 50 mit Tabelle 2 und erhält die Sequenz 4, 35, 8. Und das entspricht, als Zahlenfolge genau der ursprünglichen Nachricht.

Anhand der Tabellen 1 und 2 können wir ebensogut Nachrichten unter Verwendung des geheimen Schlüssels (d.h. erst Tabelle2 benutzen) verschlüsseln, dann mit dem öffentlichen Schlüssel (d.h. Tabelle 1 als zweites benutzen) dekodieren und damit unsere ursprüngliche Zahl wieder hergestellen. Das bedeutet – wie wir bereits im „Einsteiger-Manual" gesehen haben -, dass der Inhaber des geheimen Schlüssels damit Nachrichten unter Verwendung des RSA-Algorithmus verschlüsseln kann. Damit ist bewiesen, dass sie eindeutig nur von ihm stammen können.

Fazit:
Wie Sie gesehen haben, ist die ganze Angelegenheit zwar im Detail kompliziert, im Prinzip aber durchaus nachvollziehbar. Sie sollen schließlich nicht nur einer Methode einfach nur vertrauen, sondern – zumindest ansatzweise – ihre Funktionsweise durchschauen. Sehr viele tiefergehende Details sind leicht im Internet zu finden.

Immerhin wissen Sie nun: wenn jemand sich an Ihren verschlüsselten E-Mails zu schaffen macht, ist er durchaus ein paar Milliarden Jahre beschäftigt.....

Anhang: GPA-Menüs und Icons im Überblick

Beinahe zum Schluß noch ein kurzer Überblick über die Menüs und Icons von GnuPP mit Verweisen auf die Kapitel, in denen die Funktionen besprochen wurden.

Menüs bei GnuPP

Datei

Fenster schließen	schließt das gerade vorn liegende Fenster Ctrl + W
Beenden	beendet das Programm Crtl + Q

Schlüssel

Schlüssel erzeugen:	startet den Assistenten zur Erzeugung eines neuen Schlüsselpaares (geheimer/öffentlicher Schlüssel) (Kapitel 4 im Schnelleinstiegs-Manual)
Schlüssel signieren:	Signatur (Beglaubigung) eines anderen Schlüssels (Kapitel 9 in diesem Handbuch). Auch über das Icon "Signieren " erreichbar.
Schlüssel importieren:	hängt einen öffentlichen Schlüssel an den GnuPP-Schlüsselbund (Kapitel 8 im Schnelleinstiegs-Manual) Auch über das Icon "Import" erreichbar
Schlüssel exportieren:	speichert einen öffentlichen Schlüssel in eine Datei, auf einen Keyserver oder in die Zwischenablage (Kapitel 5 und 6 im Schnelleinstiegs-Manual) Auch über das Icon "Export" erreichbar
Private Schlüssel exportieren:	speichert den eigenen geheimen Schlüssel in eine Datei, z.B. um ihn auf einen anderen Rechner zu übertragen (Kapitel 12 in diesem Handbuch)
Sicherheitskopie anlegen:	speichert eine Sicherheitskopie des gesamten Schlüssels (geheimer und öffentlicher Teil) auf eine Diskette oder auf die Festplatte (Kapitel 4 im Schnelleinstiegs-Manual)

Optionen:

Key Server:	hier kann man einen alternativen Key Server eintragen, z.B. als Reserve oder den internen Keyserver einer Firma
Standard-Schlüssel:	hier bestimmt man den Schlüssel, mit dem Archivkopien von E-Mails verschlüsselt werden (Kapitel 8 im Einsteigerhandbuch)

Fenster:

Fenster:	schaltet zwischen Dateiverwaltung und Schlüsselverwaltung um

Info:

Info:	Version, Lizenz, Gewährleistung

Icons im Überblick

Ändern: ändert das Benutzervertrauen eines öffentlichen Schlüssels und beim eigenen Schlüssel das Verfallsdatum (Kapitel 5 in diesem Handbuch)

Löschen: löscht den angewählten Schlüssel nach Warnhinweis

Signieren: nach genauer Prüfung eines fremden Schlüssels kann dieser hiermit signiert werden (Kapitel 9 in diesem Handbuch)

Import: hiermit kann der öffentliche Schlüssel eines Korrespondenzpartners am eigenen Schlüsselring befestigt werden (Kapitel 8 im Einsteigerhandbuch)

Export: der eigene öffentliche Schlüssel kann hier an einen Keyserver geschickt oder in eine Datei bzw. die Zwischenablage gespeichert werden, um dann per E-Mail verschickt zu werden. (Kapitel 5 und 6 im Einsteigerhandbuch)

Übersicht: Kurzanzeige der Schlüsseleigenschaften

Details: Ausführliche Anzeige der Schlüsseleigenschaften

Dateien: öffnet die Dateiverwaltung, über die Dateien verschlüsselt werden können

WinPT-Menüs und Icons im Überblick

Schlüsselverwaltung: öffnet den Schlüsselverwaltungsdialog von WinPT
Die Funktionen entsprechen der Schlüsselverwaltung in GPA

Dateimanager: öffnet den WinPT-Dateimanager, der funktionell dem GPA-Dateiverwaltungsdialog entspricht.

Zwischenablage editieren: hiermit kann man Text in der Zwischenablage ansehen, verändern oder neu schreiben und dann mit der nächsten Funktion direkt verschlüsseln

Zwischenablage: verschlüsselt oder signiert den Inhalt der Zwischenablage. Funktioniert mit jedem Anwendungsprogramm und mit jeder Windows-Version.

Aktuelles Fenster: verschlüsselt oder signiert den Inhalt des aktuellen Fensters, ohne das der Inhalt des Fensters zuvor in die Zwischenablage kopiert werden muss. Funktioniert nicht mit allen Anwendungsprogrammen und nicht mit jeder Windows-Version.

Einstellungen: spezielle Benutzereinstellungen für WinPT und GnuPG. Nehmen Sie hier nur Veränderungen vor, wenn Sie sich über die Auswirkungen genau bewußt sind. Andernfalls kann die Funktion der GnuPP-Software gestört werden.

Über GPG | Über... Versions- und Copyrightinformationen zu GnuPG und WinPT

Beenden: beendet WinPT

Herausgegeben und gefördert vom
Bundesministerium für Wirtschaft und Technologie

www.gnupp.de

Projektleitung:
G-N-U GmbH
EDV-Dienstleistungen
E-Mail: info@g-n-u.de
WWW: http://www.g-n-u.de

Redaktion:
TextLab
text + medien
http://www.textlab.de