GhK

Verschlüsselung und Zertifizierung

Sonderausgabe des /ghk/hrz/info

Sichere Hauspost - Verschlüsselte Datenkommunikation an der GhK

Niemand arbeitet allein. Wie in jeder großen Einrichtung besteht auch in der GhK ein großer Teil der täglichen Arbeit in der Kommunikation mit anderen, seien es interne Kommunikationspartner: Studenten, Kollegen, Vorgesetzte, Projektmitglieder, Verwaltungsstellen; oder externe: Mitarbeiter anderer Hochschulen, Ministerien, Kooperationspartner aus der Wirtschaft.

Immer mehr verlagern sich diese Kommunikationsaufgaben auf elektronische Medien. eMail ist schnell und praktisch, kann jederzeit gelesen werden und man bekommt kein Besetztzeichen wenn die Gegenseite gerade nicht erreichbar ist. Aber eMail hat auch Nachteile: Absenderadressen sind leicht fälschbar, so dass man nie weiß, von wem eine bestimmte eMail wirklich kommt. Und eMails können auf ihrem Weg vom Absender zum Empfänger wie Postkarten von vielen Fremden gelesen werden, auch von solchen, die ein Interesse daran haben könnten, die in der eMail enthaltene Information zu missbrauchen.

Stellen wir uns vor, man möchte die Anmeldung zu einem Kurs elektronisch abwickeln: mit einer konventionellen eMail kann man nicht verhindern, dass sich eine technisch versierte Person als jemand anders ausgibt und letzteren für den Kurs anmeldet. Oder dass die eigene Anmeldung abgehört wird und dadurch jemand, der es nicht wissen sollte, erfährt, dass man an diesem Kurs teilnimmt.

Ein anderes Beispiel: ein Studierender sendet seine Hausarbeit in einem Seminar an den Dozenten. Wie kann der Dozent wissen, ob die Nachricht wirklich von diesem Studierenden stammt, der im Absender der eMail steht? Wie kann sich der Studierende vergewissern, dass niemand seine Arbeit unterwegs abfängt und die Ergebnisse kopiert?

Noch brisanter werden diese Fälle, wenn wir die Kommunikation zwischen Einrichtungen der Hochschule betrachten, weil dabei oft wirklich sensitive persönliche Daten ausgetauscht werden. Wenn Dekanate mit der Hochschulverwaltung Prüfungs- oder Personaldaten austauschen, oder Gehaltsabrechnungen und Urlaubsanträge elektronisch bearbeitet und versendet werden sollen, dann ist die sichere Übertragung dieser Daten zwingend erforderlich.

In allen diesen Fällen kann die Verschlüsselung der Daten, die elektronische Signatur, und die Zertifizierung von elektronischen Schlüsseln durch die Zertifizierungsinstanz der GhK Abhilfe schaffen. In diesem Heft erfahren Sie, wie.

Kontakt:
Andreas Matthias
matthias@hrz.uni-kassel.de
Tel. 804 2281

Im Artikel „Gekonnt verheimlichen - Grundbegriffe der Datenverschlüsselung“ werden die wichtigsten Begriffe der Kryptographie eingeführt. Ein kurzer Abriss ihrer Geschichte soll den Einstieg zusätzlich erleichtern. Anschließend beschäftigen wir uns mit der Ver-

schlüsselung von elektronischer Post auf ganz praktische Weise. Drei Artikel zur Benutzung des Verschlüsselungsprogramms PGP unter Windows zeigen, dass die tägliche Arbeit mit sicherer eMail nichts geheimnisvolles an sich haben muss. Für die Benutzer, die unter Unix oder Linux arbeiten, folgen darauf zwei etwas technischere Artikel zur Installation und Benutzung von PGP unter Unix. Diese brauchen Sie als Windows-Benutzer nicht zu lesen.

Um sicher mit Unbekannten verschlüsselte eMails austauschen zu können, muss es einen einfachen Weg geben, sich ihrer Identität zu vergewissern. Diese Aufgabe erledigt die neu eingerichtete Zertifizierungsinstanz der GhK, die im Anschluss daran ausführlich beschrieben wird. Hier erfahren Sie auch, wie Sie möglichst einfach zu Ihrem persönlichen Zertifikat kommen, und wozu so was überhaupt gut ist.

Sollten Sie nach dem Lesen dieser Ausgabe noch Fragen zur Zertifizierung, zur verschlüsselten Datenübertragung oder vielleicht einem praktischen Anwendungsfall haben, der Sie beschäftigt, dann zögern Sie nicht, Kontakt mit der GhK-CA aufzunehmen. Die CA wird Sie gerne in Sicherheitsfragen beraten oder Ihre Anfrage an einen geeigneten Sachverständigen weitervermitteln.

Die Adresse der GhK-CA finden Sie auch in diesem Heft oder unter
`http://www.uni-kassel.de/security/ghk-ca/`

Gekonnt verheimlichen
Grundbegriffe der Datenverschlüsselung

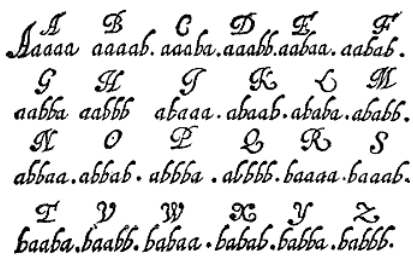
Geschichte der Kryptographie

Lange Zeit kam man ohne verschlüsselte Datenübertragung aus: man steckte die sensitive Information in einen versiegelten Umschlag und schickte den reitenden Boten zum Empfänger. Pech, wenn der Bote von der Gegenseite bestochen war: dann war der Krieg verloren.

Schon im indischen Kama Sutra wird deshalb die Kunst des geheimen Schreibens als eine der 64 Künste genannt, die Frauen beherrschen und praktizieren sollten.

Die ersten, die Kryptographie systematisch nutzten, waren vermutlich die Griechen. Im 5. Jahrhundert v.Chr. setzten die Spartaner kryptographische Verfahren für ihre militärischen Operationen ein. Dazu verwendete man einen Holzstab, auf den ein Streifen Papyrus so gewickelt wurde, dass man eine Nachricht auf den Stab schreiben konnte. Wenn man den Papyrus vom Stab abrollte, stand darauf nur eine Folge von unzusammenhängenden Zeichen; um die Nachricht zu entschlüsseln war ein Stab von exakt denselben Dimensionen und das Wissen um die Art des Aufwickelns des Papyrusstreifens nötig.

Bis heute bekannt ist die „Caesar-Verschlüsselung“, die darin besteht, dass man einfach die Buchstaben des Alphabets zyklisch gegen andere vertauscht, die eine bestimmte Anzahl von Zeichen weiter vorne oder hinten im Alphabet liegen. Um solche Nachrichten zu entschlüsseln braucht man, wenn man das Prinzip kennt, natürlich bloß 26 Versuche.

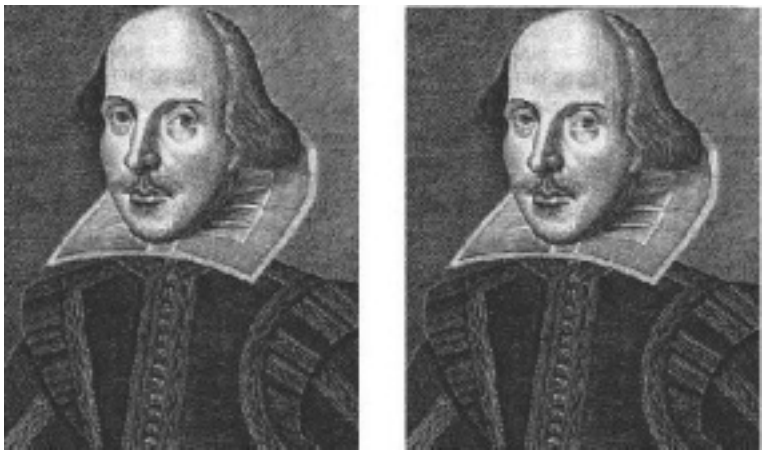


Francis Bacon hat mit seinem Zweibuchstaben-Alphabet (s. Abb. rechts) die Sache schon etwas komplizierter gemacht. Interessant ist hierbei, dass es sich bei diesem Alphabet eigentlich schon um eine Form der Binärcodierung handelt, sehr ähnlich der ASCII-Codierung, die heute in allen Rechnern eingesetzt wird.

Größere Fortschritte machte die kryptographische Technik erst in der Renaissance. 1518 veröffentlichte Trithemius, ein deutscher Benediktiner, seine „Polygraphie“. Sein Verfahren bestand im wesentlichen aus einer einfachen Buchstaben-Substitution wie bei der Caesar-Codierung, nur wurden dann die einzelnen Buchstaben des codierten Textes als Anfangsbuchstaben von Wörtern oder Sätzen benutzt. Der resultierende Text sah wie ein ganz gewöhnlicher Text aus, und nur der Eingeweihte kam auf die Idee, bestimmte Buchstaben darin als Code zu erkennen und die Nachricht zu entschlüsseln. Dieses

Kontakt:
Andreas Matthias
matthias@hrz.uni-kassel.de
Tel. 804 2281

Verfahren (Steganographie) unterscheidet sich von früheren Ansätzen dadurch, dass die übermittelte Botschaft nicht als verschlüsselt zu erkennen ist. Moderne Varianten dieses Ansatzes verstecken die codierte Nachricht unauffällig in Bildern oder Tondateien.



Das Shakespeare-Bild links ist das Original. Im rechten Bild ist folgende Nachricht versteckt:
Steganography is the art and science of communicating in a way which hides the existence of the communication. In contrast to cryptography, where the "enemy" is allowed to detect, intercept and modify messages without being able to violate certain security premises guaranteed by a cryptosystem, the goal of steganography is to hide messages inside other "harmless" messages in a way that does not allow any "enemy" to even detect that there is a second secret message present [Markus Kuhn 1995-07-03].

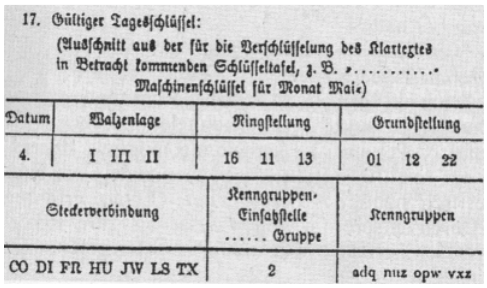
Aber auch im zweiten Weltkrieg hat man steganographische Techniken verwendet, die direkt auf das Prinzip des Trithemius zurückgehen.
Folgende Botschaft wurde tatsächlich von einem deutschen Spion abgeschickt:
Apparently neutral's protest is thoroughly discounted and ignored. Isman hard hit. Blockade issue affects pretext for embargo on byproducts, ejecting suets and vegetable oils.
Nimmt man den zweiten Buchstaben eines jeden Wortes, so ergibt das die wirkliche Botschaft:
Pershing sails from NY June 1.

Andere Systeme der Kryptographie waren lange Zeit nur Variationen der bereits erwähnten: so schrieb man die Botschaft zum Beispiel in eine quadratische Schablone, in der bestimmte Felder ausgestanzt waren. Durch rotieren der Schablone während des Schreibens erreichte man, dass die Buchstaben am Ende ziemlich durcheinander in einer quadratischen Form standen. Das Zusammensetzen der Nachricht erforderte dieselbe Schablone und das Wissen darüber, wie man sie genau zu drehen und zu verschieben hatte.

Heute setzt man mathematische Verfahren ein, um eine Botschaft in ein Kryptogramm zu übersetzen. Ein Grundzug ist jedoch allen diesen bisher genannten Verfahren gemein: derjenige, der die Botschaft entschlüsseln will, muss im Besitz einer geheimen Zusatzinformation sein, des „Schlüssels“, der ihn in die Lage versetzt, die Verschlüsselung rückgängig zu machen.

Symmetrische und asymmetrische Verschlüsselung

Darin liegt auch das große Problem aller dieser Verschlüsselungsmethoden, die man zusammenfassend als „symmetrische“ Verfahren anspricht: da mit demselben Schlüssel (Holzstab, Buchstaben-Verschiebung, Schablone, Passwort) sowohl codiert als auch decodiert werden muss, muss dieser geheime Schlüssel an beiden Endpunkten der Kommunikation vorhanden sein. Und da jeder, der über den geheimen Schlüssel verfügt, die Botschaft auch dechiffrieren kann, muss man einen vertrauenswürdigen Kanal haben, über den der geheime Schlüssel übertragen werden kann, so dass man sicher sein kann, dass nur der intendierte Empfänger und sonst niemand ihn besitzt.



17. Gültiger Tagesschlüssel:
(Ausschnitt aus der für die Verschlüsselung des Klartextes in Betracht kommenden Schlüsseltafel, z. B. ».....« Maschinenschlüssel für Monat Mai«)

Datum	Walzenlage	Ringstellung	Grundstellung
4.	I III II	16 11 13	01 12 22

Steckerverbindung	Kenngruppen-Einsatzstelle ..... Gruppe	Kenngruppen
CO DI FR HU JW LS TX	2	adq nuz opw vxz

Die Praktikabilitätsgrenzen dieser symmetrischen Verfahren wurden im zweiten Weltkrieg erreicht, als es den Alliierten gelang, den deutschen Code der Enigma-Chiffriermaschine zu knacken. Wenn die geheime Information, wie es damals der Fall war, mehrere hundert oder tausend Empfänger erreichen soll, die über mehrere Kontinente verstreut sind, dann wird das Geheimhalten des Schlüssels zu einem fast unlösbaren Problem. Die Enigma-Schlüssel wurden täglich nach einem genau festgelegten Plan verändert (vgl. Abb.)

In derselben Situation sehen wir uns heute, wenn wir verschlüsselte Nachrichten übers Netz austauschen wollen. Wenn ich ein geheimes Passwort zur Verschlüsselung einer eMail verwende, wie kann ich es dann dem Empfänger mitteilen? Per eMail? Geht nicht, weil wenn jemand meine eMail abhört, hat er sowohl die codierte Botschaft als auch den Schlüssel in der Hand. Also muss ein anderer Kanal her, zum Beispiel das Telefon. Aber woher weiß ich, dass die Stimme am Telefon wirklich der ist, dem ich die Nachricht senden wollte? Viele Kommunikationspartner kennen sich heutzutage ja nicht mehr persönlich.

So hat man also die asymmetrischen Verschlüsselungsverfahren ersonnen, eine wirklich geniale Idee. Das geht im Prinzip so: Jeder Kommunikationspartner hat zwei Schlüssel statt nur einem: Schlüssel A und Schlüssel B. Die beiden Schlüssel werden durch ein kompliziertes mathematisches Verfahren gleichzeitig erzeugt und sind nur

als Paar nützlich. Das Verfahren ihrer Entstehung garantiert, dass alles, was Schlüssel A verschlüsselt hat, sich nur durch Schlüssel B entschlüsseln lässt und umgekehrt. Schlüssel A kann die Nachrichten, die er verschlüsselt hat, selber nicht mehr entschlüsseln, ebenso geht es Schlüssel B mit seinen Nachrichten. Da für Ver- und Entschlüsselung zwei verschiedene Schlüssel nötig sind, nennt man diese Verfahren asymmetrisch.

Nun geht man her und macht einen der beiden Schlüssel öffentlich (welchen man nimmt ist im Prinzip egal), den anderen hält man geheim. Wir sprechen deshalb von einem öffentlichen oder public und einem geheimen oder private Schlüssel (key). Jeder dieser Schlüssel kann nur die Nachrichten entschlüsseln, die der andere Schlüssel verschlüsselt hat.

Benutzer X hat also zwei Schlüssel eines Paares. A ist öffentlich, B geheim. Nun möchte Benutzer Y dem Benutzer X eine Botschaft zukommen lassen. Dazu nimmt Y den (ihm ja bekannten) öffentlichen Schlüssel A des Benutzers X und verschlüsselt die Nachricht damit. Die verschlüsselte Nachricht wird an X geschickt und nur er kann sie entschlüsseln, weil nur er den geheimen Schlüssel B kennt, der alleine in der Lage ist, die mit A verschlüsselte Botschaft zu entschlüsseln. (Diesen Absatz am besten gleich noch mal lesen).

Auf diese Weise kommen wir ohne einen sicheren Kommunikationskanal zwischen Sender und Empfänger der geheimen Information aus. Solange X seinen geheimen Schlüssel B nicht weitergibt, ist das Verfahren sicher. An keiner Stelle müssen die Beteiligten ein Passwort oder eine andere geheime Information austauschen.

Nützlich ist diese Art der Verschlüsselung auch, weil man damit eine elektronische Signatur (Unterschrift) erzeugen kann, die garantiert, dass ein Dokument von einem bestimmten Absender kommt und von niemand sonst. Dazu dreht man das Verfahren einfach um. Nehmen wir an, X will dem Y ein signiertes Dokument schic??ken, in dem X als Absender eindeutig feststellbar ist. Dazu „verschlüsselt“ er das Dokument mit seinem privaten Schlüssel B. Die Verschlüsselung taugt zwar nicht viel, weil ja jeder, der den öffentlichen Schlüssel A hat (und das sind bei einem öffentlichen Schlüssel per definitionem alle) das Dokument entschlüsseln kann. Aber jeder Empfänger (auch der Y) weiß gleichzeitig, dass nur X als Absender in Frage kommt: denn wenn das Dokument mit dem Schlüssel A entschlüsselbar ist, dann muss es mit B verschlüsselt worden sein - und diesen Schlüssel besitzt (weil er geheim ist) nur der Benutzer X und sonst niemand.

Glaubwürdigkeit und Zertifizierung

Jedes Verfahren, so toll es auch klingen mag, hat seine Schwachstellen. So auch die asymmetrische Public-Key-Verschlüsselung, die wir gerade besprochen haben. Nehmen wir an, ein böser Mensch (B) möchte dem Y ein Dokument zuschicken, das angeblich von X kommt, was aber in Wirklichkeit von B stammt. Da Y über ein gesundes Misstrauen verfügt, besteht er auf einem von X signierten

Dokument. B beginnt also damit, dass er dem Y eine manipulierte eMail zukommen lässt, die so aussieht, als käme sie von X. Das ist bei den heutigen Mailsystemen überhaupt kein Problem. In diesem eMail schickt B dem Y einen falschen öffentlichen Schlüssel, der angeblich der Schlüssel von X sein soll. In Wirklichkeit aber ist er von B erzeugt worden.

Im nächsten Schritt schickt B das Dokument an Y, das er vorher mit dem Schlüssel signiert hat, der vermeintlich X gehört. Der ahnungslose Y erhält also ein Dokument, das für ihn, da er einen falschen Schlüssel von X hat, so aussieht, als wäre es von X signiert worden.

Wie lässt sich so ein Missbrauch verhindern? Im Idealfall dadurch, dass Y Kontakt zu X aufnimmt (zum Beispiel über das Telefon) und verifiziert, dass der Schlüssel, den er bekommen hat, auch wirklich der Schlüssel von X ist. Da die Schlüssel selber sehr lang und umständlich vorzulesen sind, gibt es für solche Zwecke die Key-ID und den Fingerprint eines Schlüssels. Key-ID und Fingerprint werden von den Verschlüsselungsprogrammen auf Wunsch angezeigt und stellen eine „lesbare Kurzfassung“ des eigentlichen Schlüssels dar. So etwa sieht das aus:

```
pub  1024/802DBD35 1998/07/15 Andreas Matthias <matthias@hrz.uni-kassel.de>
            Key fingerprint = DE 54 40 E0 19 F7 A4 F3  39 DA AB 22 0B 33 04 79
```

Der Benutzer Y würde also X anrufen und sagen: „ich habe hier einen Schlüssel von dir mit der ID 802DBD35 und dem Fingerprint DE 54 40 E0 19 F7 A4 F3 39 DA AB 22 0B 33 04 79 vorliegen. Ist das wirklich dein Schlüssel?“ - Und X würde, da dieser Schlüssel ja in unserem Beispiel vom bösen Benutzer B erzeugt wurde, antworten: „Nein, das ist nicht mein Schlüssel!“ Auf diese Weise würde der Missbrauch verhindert werden können.

Nun ist es nicht immer möglich, alle Schlüssel persönlich zu verifizieren. Hier beginnt das Betätigungsfeld einer Certification Authority, auch kurz „CA“ genannt. Die CA ist eine Organisation, der beide Benutzer vertrauen, und die bestätigt, dass ein bestimmter Schlüssel zu einer bestimmten Person gehört. Ein Schlüssel wird, wie man sagt, durch die CA „zertifiziert“. Das Zertifikat wird Teil des zertifizierten Schlüssels und lässt sich darin jederzeit wiederfinden. Wenn Y jetzt also einen zertifizierten Schlüssel von X erhält, dann kann er nachsehen und feststellen, dass darin die GhK-CA bestätigt, dass dieser Schlüssel wirklich von X stammt. Um die Echtheit der Zertifikate zu überprüfen, muss Y nur noch den Fingerprint des CA-Schlüssels mit dem Zertifikat vergleichen, und nicht den eines jeden seiner Kommunikationspartner. Das erleichtert die Überprüfung der Echtheit von Schlüsseln erheblich und ermöglicht erst den Einsatz von Public-Key-Kryptographie auf breiter Basis.

Eine jede CA befolgt bestimmte festgelegte Regeln bei der Zertifizierung von Schlüsseln. Die Summe dieser Regeln wird die Policy der CA genannt, und garantiert, daß die Zertifikate dieser CA bestimmte Sicherheitsstandards erfüllen. Zum Beispiel stellt die GhK-CA keine Zertifikate aus, wenn der Benutzer sich nicht persönlich vorgestellt und bei der CA durch ein amtliches Ausweisdokument identifiziert

hat. Die Policy der GhK-CA ist natürlich öffentlich zugänglich. Sie finden Sie unter:
`http://www.uni-kassel.de/security/policy/`
Die Schlüssel und Fingerprints der GhK-CA und des DFN finden Sie unter:
`http://www.uni-kassel.de/security/pubkeys/ca/`

Literatur

- Historischer Hintergrund übersetzt und angepasst aus: http://www.txcc.net/~silentop/enc.txt
- Steganographie (Shakespeare-Beispiel): http://www.jjtc.com/stegdoc/
- C. Schulzki-Haddouti: Elektriktrick - Chiffriermaschinen des 20. Jahrhunderts. c't Magazin für Computertechnik, 3/2000

Verschlüsseln mit PGP

Pretty Good Privacy (PGP) ist ein weltweit verbreiteter Standard zum Verschlüsseln von Text. Inzwischen gibt es die Software in frei kopierbaren Versionen für eine große Anzahl von Betriebssystemen. Mit PGP erzeugt der Anwender ein Schlüsselpaar, bestehend aus einem geheimen Schlüsselteil (Private Key) und einem öffentlichen Schlüsselteil (Public Key). Nur wenn beide Teile zusammenpassen (zu einem Schlüssel gehören), gelingt die Chiffrierung bzw. Dechiffrierung. Darüber hinaus kann man den Schlüssel auch als digitale Unterschrift (Signatur) verwenden.

Der Public Key, so sagt es schon der Name, wird möglichst weit verbreitet, der Private Key bleibt beim Besitzer. Mit dem öffentlichen Schlüssel des Empfängers wird verschlüsselt, der Empfänger entschlüsselt mit seinem dazu passenden geheimen Schlüssel.

Zur Erhöhung der Sicherheit kann man seinen Schlüssel noch zertifizieren lassen, d.h. von einer wiederum authorisierten Stelle bestätigen lassen, dass ein bestimmter Schlüssel zu einem bestimmten Inhaber gehört.

Pretty Good Privacy: Der Standard zum Verschlüsseln von Nachrichten für Win9x/NT

Wo finden Sie die Software?

Die Freeware-Version PGP 6.5.x (Win95/NT und andere Plattformen) für nicht-kommerzielle Anwendungen gibt es z.B. unter
`ftp://ftp.ims.uni-stuttgart.de/pub/communication/pgp/pgpi`
oder auch (sortiert nach Plattformen) unter
`http://www.pgpi.org/products/pgp/versions/freeware/`
Die Version 6.5.8 für Windows heißt bspw. PGPFW658Win32.zip .

Inzwischen gibt es schon die Version PGP 7.0, erhältlich unter ftp://ftp.de.pgpi.com/pub/pgp/7.0/. Die folgende Beschreibung bezieht sich aber noch auf Version 6.5.x.

Bitte beachten Sie die Lizenzbestimmungen für die Freeware-Version, die in der Datei license.txt abgedruckt ist.

Wie installieren Sie die Software?

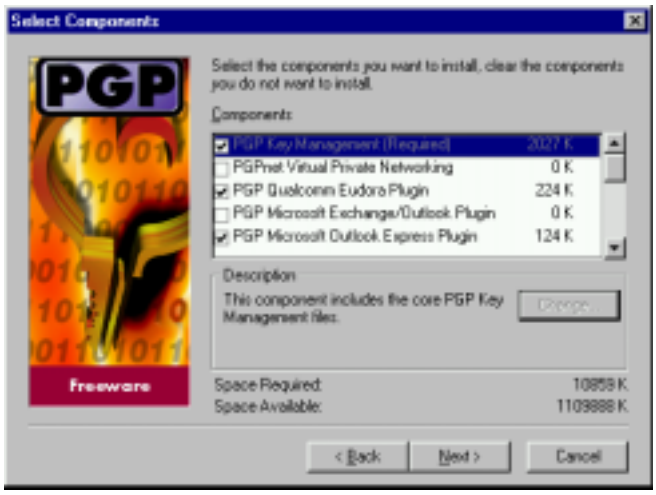
Nachdem Sie die Datei PGPFW658Win32.zip in ein temporäres Verzeichnis (z.B. C:\temp\pgp\)auf Ihre Festplatte kopiert haben, entpacken Sie die Datei und führen anschließend das Installationsprogramm aus.

Nach Kenntnisnahme der Lizenzbestimmungen erscheinen verschiedene Fenster, in denen

Name, Organisation **[Next >]**
Pfad, in den PGP installiert werden soll, **[Next >]**
und zu installierende *Module* (z.B. Plugins für Eudora oder Outlook) **[Next >]**

abgefragt werden.

Zusätzlich werden Sie gefragt, ob Sie bestehende Schlüssel übernehmen wollen. Wir nehmen an, Sie haben noch nie mit PGP gearbeitet und antworten: Nein



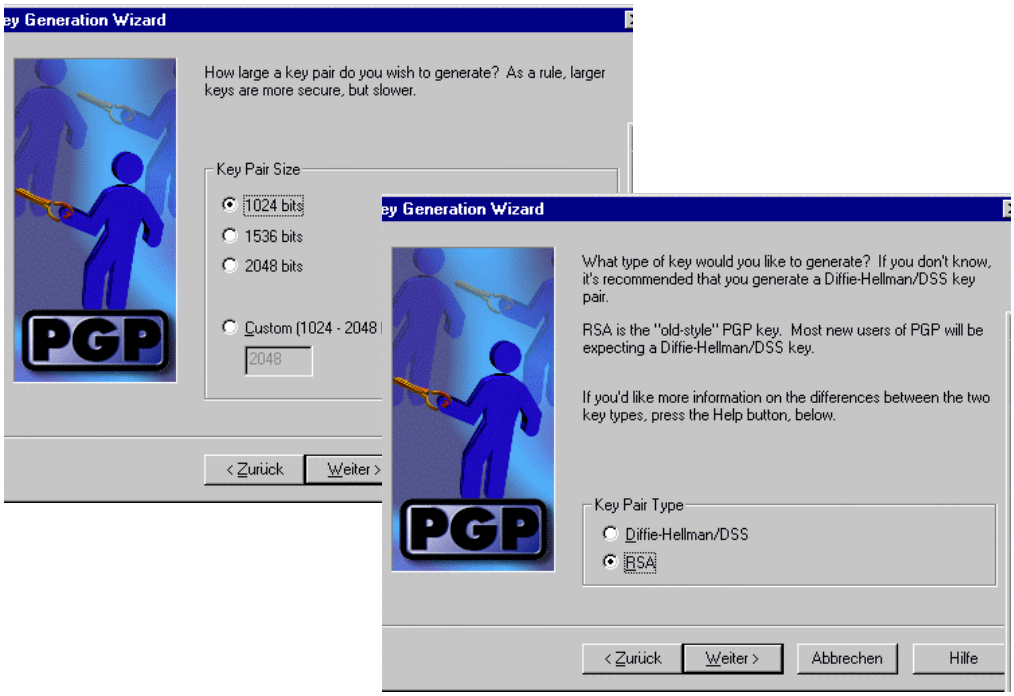
Das Abschlussfenster bietet die Möglichkeit, jetzt ein eigenes Schlüsselpaar zu erzeugen sowie die README-Datei zu lesen, was Sie nicht versäumen sollten: **[Finish]**

Eigenes Schlüsselpaar erzeugen

Nach der Installation wird der Key Generation Wizard gestartet. Dieser Assistent führt Sie durch die Erzeugung Ihres Schlüsselpaares.

Geben Sie Ihren Namen (Vorname Nachname) und Ihre E-Mail-Adresse ein. **[Weiter >]**

Die beiden folgenden Fenster sind sehr wichtig. Sie legen den Grad der Verschlüsselung (je höher, desto sicherer ist der Schlüssel) und die Art der Verschlüsselung fest.

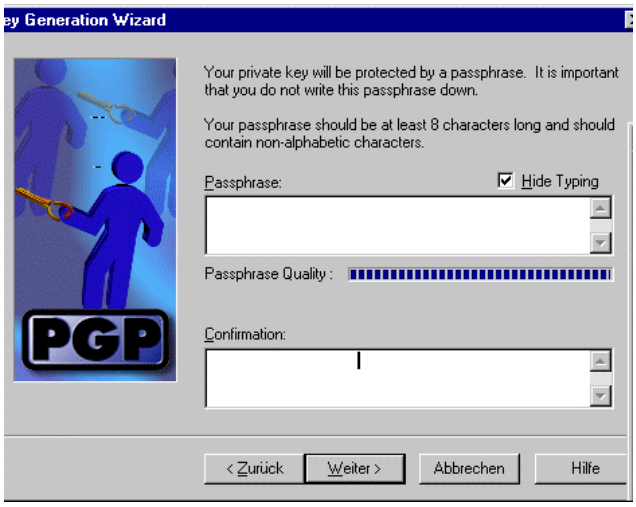


Key Pair Type **muss** RSA sein, damit Sie verschlüsselte Mail auch mit Anwendern austauschen können, die diesen weit verbreiteten Schlüsseltyp verwenden. Die Verschlüsselungsmethoden RSA und DSS sind **nicht** kompatibel. **[Weiter >]**

Die Größe des Schlüssels sollte mindestens 1024 bits, besser 2048 bits betragen. **[Weiter >]**

Der Schlüssel sollte auch kein „Verfallsdatum“ tragen (also keypair never expires ankreuzen), weil der öffentliche Schlüssel verbreitet wird und immer nutzbar sein soll. **[Weiter >]**

Im nächsten Fenster geben Sie einen kurzen (3-4 Worte), leicht merkbaren Satz (Passphrase) ein. Diesen Satz brauchen Sie jedes Mal, wenn Sie eine Mail signieren („unterschreiben“) oder an Sie gerichtete, mit Ihrem öffentlichen Schlüssel gesicherte Nachrichten, lesen wollen. Deshalb sollte der Satz nicht zu lang sein. Die Qualität der Passphrase wird während der Eingabe mit einer blauen Leiste angezeigt. Diesen Satz müssen Sie zur Bestätigung wiederholen. **[Weiter >]**



Nun arbeitet PGP kurz vor sich hin,

dann können Sie wieder auf **[Weiter >]** klicken. Abschließend werden Sie aufgefordert, Ihren öffentlichen Schlüssel an einen Keyserver zu schicken. Mehr zum Keyserver steht unter `/security/pgp/manuals/PGP_Funktionen.html.`

Wenn Sie Ihren Schlüssel zertifizieren lassen möchten, müssen Sie den Key jetzt noch nicht an einen Keyserver schicken, das geschieht nach der Zertifizierung automatisch durch die GhK-CA. **[Weiter >]**

Ist die Installation fertig, sehen Sie in der Statuszeile ihres Windows-Fensters unten rechts ein Schlosssymbol. Das Programm ist aktiv und kann benutzt werden.

Informationen zur Benutzung stehen unter
`http://www.uni-kassel.de/security/pgp/manuals/pgp-windows/PrettyGo_Funktionen.html`

Die wichtigsten Funktionen von PGP 6.5.x

Um eine der folgenden Funktionen auszuführen, klickt man auf das Schloss in der Statuszeile (unten rechts).

Nach der Schlüsselgenerierung verfügen Sie nun über einen

- Public Key (öffentlichen Schlüssel)
- Private Key (geheimen Schlüssel)
- Passphrase (Merksatz, Mantra)
- Keyring (Schlüsselbund)
- Fingerprint

Wichtig ist nun noch die Erzeugung eines Widerrufschlüssels. Dieser Vorgang ist weiter unten beschrieben.

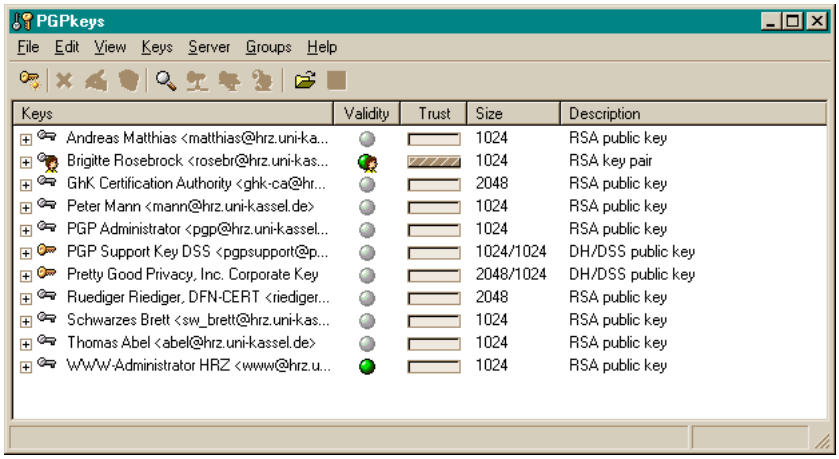
Schlüsselverwaltung

Ein Klick auf PGPKeys startet die Schlüsselverwaltung. Diese bereitet Ihren Schlüssel für die Veröffentlichung vor und nimmt andere Schlüssel in Ihren Schlüsselbund auf.
In diesem Fenster sehen Sie dann Ihren soeben erstellten Schlüssel neben einigen anderen, die PGP schon mitliefert.

Gelber Schlüssel	Öffentlicher DSS-Schlüssel (nicht kompatibel mit RSA!)
Grauer Schlüssel	Öffentlicher RSA-Schlüssel
Grauer (gelber) Schlüssel mit Kopf	Eigenes Schlüsselpaar: Privater und öffentlicher Schlüssel (erscheint nur bei Ihrem Namen)

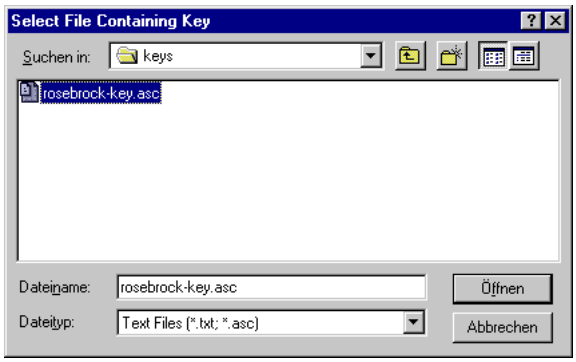
Schlüssel importieren

Sie wählen aus dem Menü Keys/Import und geben an, wo die



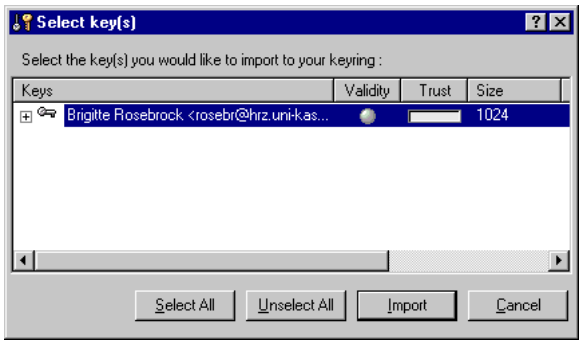
Kontakt:
Brigitte Rosebrock
rosebr@hrz.uni-kassel.de
Tel. 804 3810

Datei mit der Schlüsselinformation steht. Das kann Ihr eigener Schlüssel sein oder der öffentliche Schlüssel einer anderen Person.



Dieses Fenster zeigt die Schlüsselinformation dieser Datei. Mit **[Import >]** fügt man diesen Schlüssel dem eigenen Schlüsselbund hinzu:

Im nächsten Fenster sehen Sie nun Ihren hinzugefügten Schlüssel.



Anschließend können Sie Ihren Schlüssel noch im HRZ bei der Zertifizierungsinstanz der GhK (GhK-CA) zertifizieren lassen. Dort wird nach persönlichem Erscheinen Ihr Schlüssel „signiert“ und in einer Liste veröffentlicht. So kann jeder Ihrer E-Mail-Partner sicher sein, dass der genannte Schlüssel wirklich mit Ihrer Person verbunden ist.

Schlüssel anderer Personen kann man auch bei verschiedenen Keyservern nachfragen.

Beispiel:
Man schickt eine E-Mail an: pgp-public-keys@keys.de.pgp.net
und schreibt als Subject: get rosebr@hrz.uni-kassel.de

Nach kurzer Zeit bekommt man die gewünschte ASCII-Datei zugeschickt, sofern diese Person einen Schlüssel generiert und an einen Keyserver geschickt hat. Die Keyserver tauschen untereinander die angemeldeten Schlüssel aus, so dass man seinen öffentlichen Schlüssel nur an einen Keyserver schicken muss.

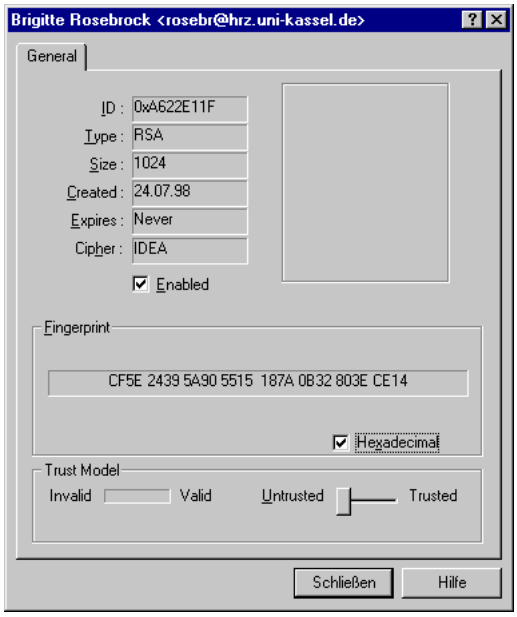
Wird Ihnen der öffentliche Schlüssel eines Mailpartners als Teil eines anderen Textes mitgeteilt, z.B. als Teil der Signatur, markieren Sie den verschlüsselten Teil, der durch

----- BEGIN PGP PUBLIC KEY BLOCK ----
und
----- END PGP PUBLIC KEY BLOCK ----

gekennzeichnet ist und kopieren ihn in die Zwischenablage. Nun wählen Sie im Menü des Schlüsselverwaltungsteils (PGPkeys) `Edit/Paste`. Nun werden die dazugehörigen Schlüsselinformationen angezeigt und mit Import wird der Schlüssel dem Schlüsselbund hinzugefügt.

Fingerprint

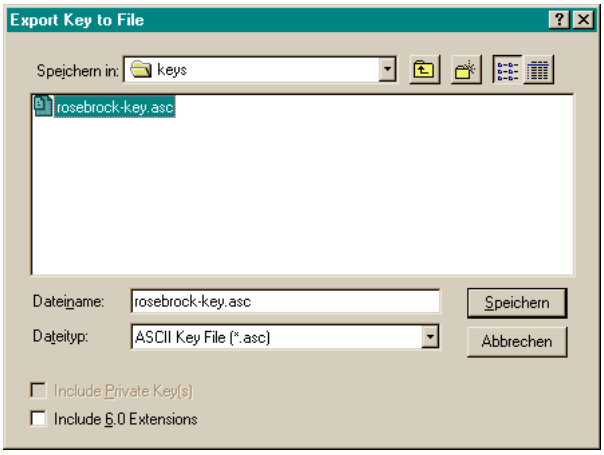
Ein Fingerprint ist eine Anzahl von Zahlen und Buchstaben, die zu Identifikation eines Schlüssels dienen. Mit dem Fingerprint kann man sehen, ob der Schlüssel authentisch ist. Er besagt nicht, dass der Schlüssel zu einer bestimmten Person gehört, wie es die Zertifizierung tut.



Einen Fingerprint kann man ansehen, indem man im Schlüsselverwaltungsmenü den gewünschten Schlüssel markiert und im Menü Keys/Key Properties auswählt. Achten Sie darauf, dass die Schaltfläche „Hexadecimal“ angekreuzt ist. Das nebenstehende Fenster erscheint.

Schlüssel exportieren

Einen Schlüssel zu exportieren bedeutet, der Schlüssel wird zu einem ASCII-Text extrahiert, zu Sicherungszwecken oder um ihn weiterzugeben. Der Private Key verbleibt beim Eigentümer, nur der Public Key wird exportiert.



Die ASCII-Datei sieht zum Beispiel so aus:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP 6.5.1i for non-commercial use

mQCNAzY9n14AAAEEAMhHLCi1Dpsr0IaZ90Rk0V7vG2INQFch3dNAfr/gb4UK8AYi
Uw62z5YOCf0lR0zjLh02aZZSkZHaHLAvcgmPqybiiyXs/0EnDYbvbjLGBFjN/DOE
8ZkctIXRLZRwq3/XV7V/sLdCEa04thVqh56Tnc61o5Y6jP2E0y2cOivv6vVdAAUR
tCxTY2h3YXJ6ZXMgQnJldHQgPHN3X2JyZXR0QGhyei51bmkta2Fzc2VsLmRlPokA
lQMFEDY9n14tnDor7+r1XQEBZXwD/RMC9BP9BR+bdjkBQS2NJGNiLb7VaNZkRpvQ
LMCf326sthE8N/U2qOGi7MksblQk8rMXYjTylMPcyspiUpOcOcyXE0By7pTldPq1
GcdcgZUTeSwheP7vBVuqKc/bG8NSntm+kLyHZVjC9NJ6gbz33EpnuKtTDkoum0i/
KmSo+icH
=gIru
-----END PGP PUBLIC KEY BLOCK-----
```

Der Public Key kann über einen Keyserver verteilt werden oder man fügt ihn seinen Mails an, z.B. als „attached file“ oder in der Signatur hinzu.
Bekommt man nun von einer anderen Person deren öffentlichen Schlüssel als ASCII-Datei, muss man diesen Schlüssel in das eigene Schlüsselbund aufnehmen.

Widerrufsschlüssel erstellen

Ein Widerufsschlüssel (revocation key) ist notwendig, um einen Schlüssel im Falles des Missbrauchs oder des Verlustes für ungültig zu erklären. Dieser Schlüssel dient dazu, andere Benutzer im Netz zu informieren, dass sie einen bestimmten Schlüssel nicht mehr verwenden sollen.

Den Widerrufsschlüssel erzeugt man am Besten gleich zu Anfang, nach der Schlüsselgenerierung. Wenn der Schlüssel nämlich unbrauchbar geworden ist, ist es zu spät, einen Widerufsschlüssel zu erstellen.

Um nicht in diese Zwickmühle zu geraten, sollten Sie ganz am Anfang zusammen mit Ihrem Schlüssel auch gleich den Widerrufs-Schlüssel erzeugen und separat vom richtigen Schlüssel aufheben. Das ist der Grund, weshalb die GhK-CA von Ihnen verlangt, bei der Zertifizierung auch gleich einen Widerrufs-Schlüssel mitzuliefern.

Ein Widerrufsschlüssel (oder Widerrufs-Zertifikat) ist nichts anderes, als ein bereits widerrufener Public Key. Um einen solchen Schlüssel zu erzeugen, müssen Sie also „probeweise“ Ihren Schlüssel widerrufen, den widerrufenen Schlüssel speichern, und anschließend, da sie Ihren Schlüssel ja noch verwenden wollen, den früheren, nichtwiderrufenen Schlüssel wiederherstellen.

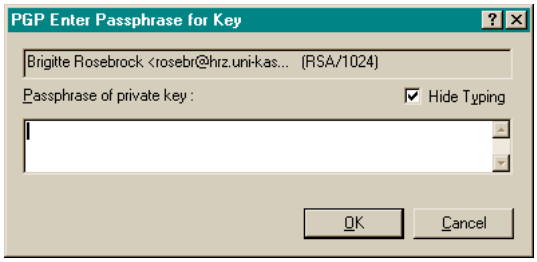
So gehen Sie vor:

Im PGP-Programmverzeichnis gibt es ein Unterverzeichnis „PGP Keyrings“. Dort befinden sich sowohl der öffentliche (pubring.pkr) als auch der private (secring.skr) Schlüsselbund. Diese

Dateien **kopieren** Sie auf Diskette oder in ein anderes Verzeichnis auf Ihrem PC.
Dann starten Sie PGPKeys (Statuszeile). Markieren Sie Ihren eigenen Schlüssel. Im Menü wählen Sie Keys/Revoke. Nun werden Sie aufgefordert, die Passphrase einzugeben. Danach erscheint Ihr Schlüssel rot durchgestrichen in der Schlüsselliste. Nun müssen Sie diesen revoke-Key sichern (exportieren). Dabei geben Sie im Namen am Besten an, dass es sich um einen revokation key handelt, z.B. `IhrName-rev.asc`. Nach der Sicherung kopieren Sie die ursprünglichen Schlüsselbunde secring.skr und pubring.pkr wieder in das Verzeichnis PGP Keyring.

Passphrase ändern

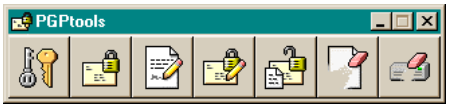
Das Ändern der Passphrase ist vergleichbar mit dem Ändern eines Passwortes. Es funktioniert unabhängig vom Erzeugen eines Schlüssels. Es ist nötig, die Passphrase zu ändern, wenn Ihr PGP-Schlüssel durch die GhK-CA erzeugt wurde. Auch wenn Sie befürchten, Ihre Passphrase ist an die Öffentlichkeit gelangt, ist das Ändern angeraten. Über das Schloss-Symbol in der Statuszeile unten rechts startet man PGPKeys. Zunächst muss man seinen eigenen Schlüssel markieren. Mit der rechten Maustaste wählt man Key Properties und dort die Schaltfläche Change Passphrase.



Nun geben Sie zuerst die alte, noch gültige Passphrase ein. Im nächsten Fenster werden Sie nach der neuen Passphrase gefragt, die zur Sicherheit nochmals eingegeben werden muss.

Dateien verschlüsseln

Dazu klickt man in der Statuszeile auf das Schloss und wählt PGPTools aus oder startet über das Startmenü aus dem Ordner PGP PGPTools.
Auch im Windows-Explorer steht nach der Installation die Dateiverschlüsselung zur Verfügung: Wenn Sie mit der rechten Maustaste im Windows-Explorer eine Datei anklicken, wird ein Menü aufgeklappt, in dem unten die PGP-Funktion angeführt ist.



In der folgenden Symbolleiste stehen Ver- und Entschlüsseln sowie Signieren für Dateien, die aus dem Dateiverzeichnis ausgesucht werden können, zur Verfügung.

Nun kennen Sie alle wesentlichen Schritte, um ein eigenes Schlüsselpaar zu erstellen und mit verschlüsselter Mail umzugehen.

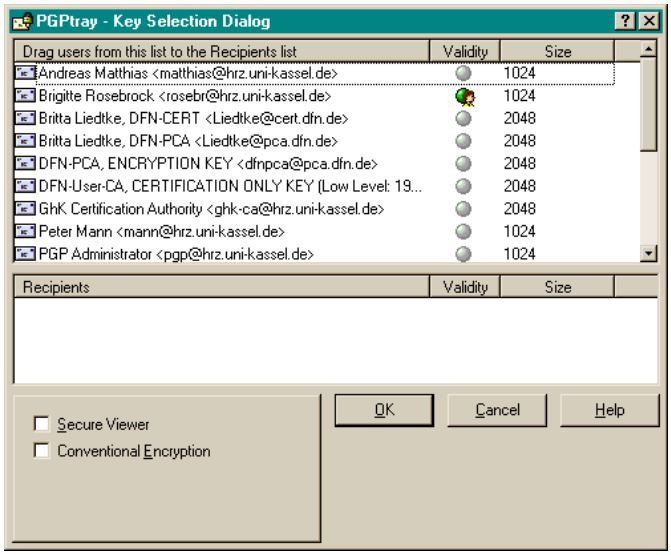
Netscape und PGP

E-Mail verschlüsseln

Netscape bietet keine direkte PGP-Unterstützung.

Die Verschlüsselung funktioniert über die Zwischenablage (=Clipboard). Sie schreiben Ihre Mail mit dem Netscape Mail Composer. Dann klicken Sie auf das Schlosssymbol in der Statusleiste und wählen `Current Window/Encrypt`.

In dem nun erscheinenden Fenster ziehen Sie mit der Maus den Empfänger in den unteren freien Bereich. So nimmt PGP den passenden öffentlichen Schlüssel des Empfängers für die Verschlüsselung. Voraussetzung ist natürlich, dass Sie diesen Schlüssel vorher Ihrem Schlüsselbund hinzugefügt haben.



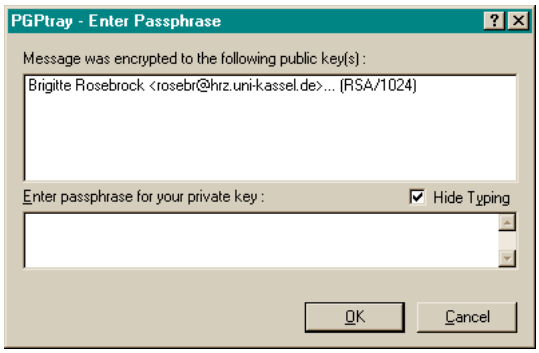
E-Mail entschlüsseln

Haben Sie eine verschlüsselte Mail erhalten und wollen Sie lesen, markieren Sie den verschlüsselten Teil, der durch

```
----- BEGIN PGP MESSAGE -----
und
----- END PGP MESSAGE -----
```

gekennzeichnet ist und kopieren ihn in die Zwischenablage.

Kontakt:
Brigitte Rosebrock
rosebr@hrz.uni-kassel.de
Tel. 804 3810



Klicken Sie auf das Schlüsselsymbol in der Statuszeile und auf CurrentWindow/ Decrypt&Verify. Der Text wird nach der Eingabe Ihrer Passphrase (Das ist der kurze, 3-4 Worte lange Satz, den Sie bei der Schlüsselgenerierung selbst ausgewählt haben.) entschlüsselt und im Textviewer dargestellt. Von dort können Sie den Text in die Zwischenablage schieben und nach Belieben verwenden.

E-Mail signieren

Um eine Mail zu unterschreiben und zu gewährleisten, dass Sie wirklich der Absender sind, wählen Sie die Option Clipboard Sign . Nachdem Sie Ihre Mail geschrieben haben (s. auch Clipboard Encrypt), klicken Sie auf das Schlüsselsymbol in der Statuszeile und auf `Current Window/Sign`.

Geben Sie Ihre Passphrase ein. (Das ist der kurze, 3-4 Worte lange Satz, den Sie bei der Schlüsselgenerierung selbst ausgewählt haben.) Anschließend verschicken Sie die Nachricht.

Eudora und PGP

Zu PGP allgemein gibt es Informationen in der Datei „Die wichtigsten Funktionen von PGP 6.51“. Dort steht auch, wie andere Schlüssel in den Schlüsselbund aufgenommen werden.

Im Installationsprogramm von PGP haben Sie als Plugin Eudora angegeben. Wenn Sie nun Eudora starten, haben Sie zusätzliche Menüpunkte und Icons in der Symbolleiste. Je nach Funktion erscheinen die Icons nur in der Inbox (entschlüsseln) oder nur im NewMessage-Fenster (signieren und verschlüsseln).

Im Einzelnen bedeuten die Icons:

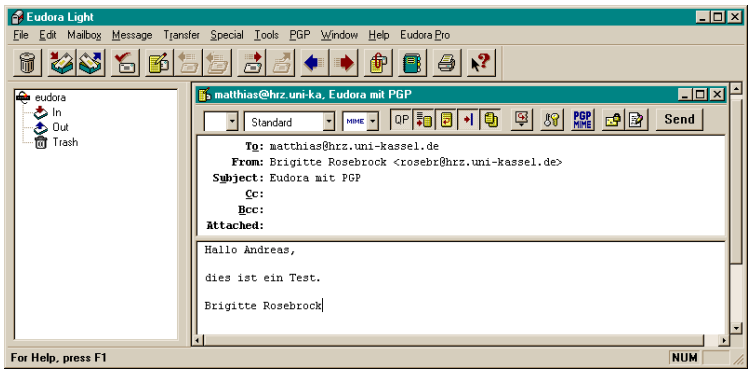

Mail verschlüsseln


Mail signieren

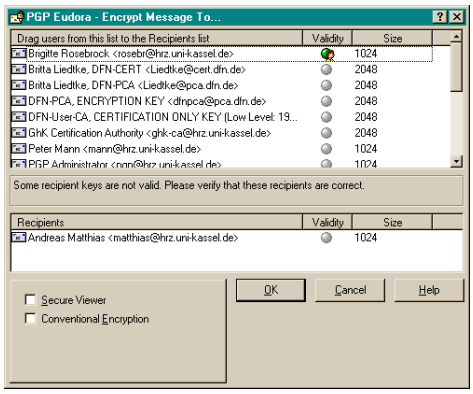

Mail entschlüsseln


Schlüssel verwalten

E-Mail verschlüsseln



Die E-Mail wird ganz normal geschrieben. Dann klicken Sie auf das Verschlüsselungs-Icon und auf **[SEND]**. Das folgende Fenster erscheint:

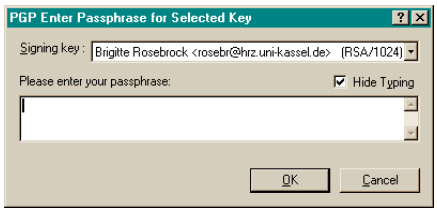


Eudora hat aus der Empfänger-Adresse schon einen Vorschlag für die Verschlüsselung mit dem öffentlichen Schlüssel der betreffenden Person gemacht.

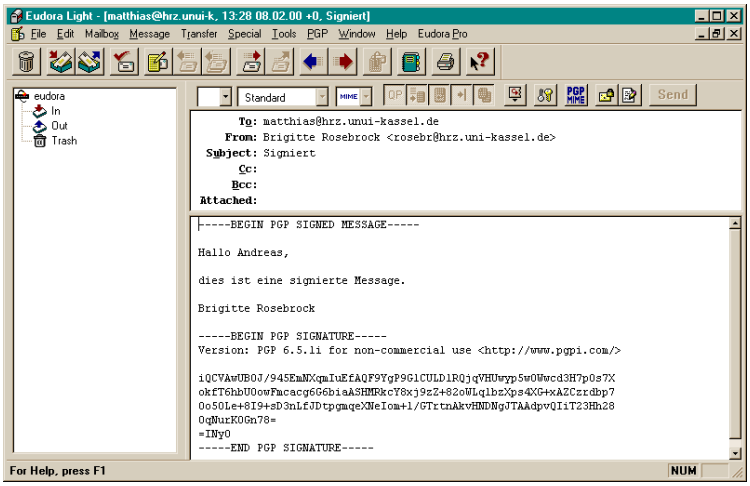
Wenn Sie E-Mail verschlüsselt verschicken, senden Sie immer eine Kopie an sich selbst, damit Sie später noch lesen können, was Sie geschrieben haben. Auf **[OK]** klicken und die Mail wird verschickt.

E-Mail signieren

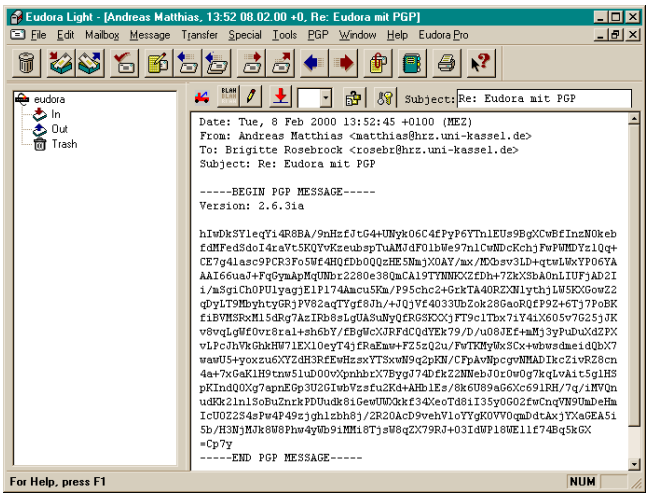
Um eine Mail zu unterschreiben und zu dokumentieren, dass Sie wirklich der Absender sind, wählen Sie nach dem Verfassen der Mail das Icon Signieren und **[SEND]**.

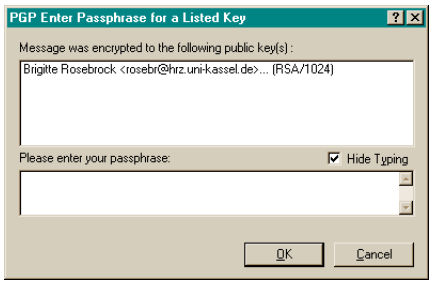


Geben Sie Ihre Passphrase ein. (Das ist der kurze, 3-4 Worte lange Satz, den Sie selbst ausgewählt haben.) Anschließend verschicken Sie die Nachricht.

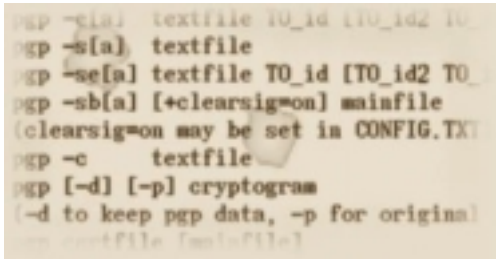


E-Mail entschlüsseln





Haben Sie eine verschlüsselte Mail erhalten und wollen sie lesen, klicken Sie auf das Entschlüsselungs-Icon. Nun müssen Sie noch Ihre Passphrase eingeben und der Text kann gelesen werden.



-@#!x*$& - Benutzung von PGP unter Unix

Das Programm PGP macht unter Unix zunächst einen recht abschreckenden Eindruck: die Auflistung der wichtigsten Optionen alleine braucht mehrere Bildschirmseiten und oft findet man darin nicht einmal die, die gerade gebraucht wird.

Es lohnt sich trotzdem, sich der Erfahrung, PGP unter Unix zu benutzen, auszusetzen: die Einbindung in die Mailprogramme ist oft besser gelöst als unter Windows, und man kann die Verschlüsselungsmöglichkeiten, die das Programm bietet, unter Unix auch besser für andere Zwecke, als nur das Verschicken von eMail, verwenden: zum Beispiel kann man, hat man sich erst einmal durch den Dschungel der Optionen hindurchgearbeitet, auf angenehm einfache Weise Textdateien, Briefe oder Diplomarbeiten verschlüsseln und zum Zeitpunkt der Bearbeitung wieder entschlüsseln, unabhängig von einem eventuellen Versand über eMail.

In diesem Artikel wollen wir sehen, wie man mit PGP unter Unix die erforderlichen Schlüssel generieren und verwalten kann, und wie sich Dateien verschlüsseln und signieren lassen. In einem anderen Artikel wird dann die Einbindung von PGP in das Mailprogramm „pine“ besprochen.

Erzeugen des eigenen Schlüsselpaars

Jeder Benutzer hat, wie an anderer Stelle beschrieben wurde, zwei Schlüssel für die PGP-Kommunikation: einen privaten und einen öffentlichen Schlüssel (private und public key). Diese beiden müssen nur einmal erzeugt werden. Der öffentliche Schlüssel wird daraufhin möglichst weit verteilt, denn ohne ihn kann einem niemand eine verschlüsselte Nachricht schicken. Der private hingegen muss sorgfältig (am besten auf einer Diskette) aufbewahrt werden und darf niemals in fremde Hände geraten!

Sie erzeugen ein Schlüsselpaar auf der Kommandozeile mit:

```
pgp -kg +nomanual
```

Dann sehen Sie folgende Abfrage:

```
Pick your RSA key size:
  1) 512 bits- Low commercial grade, fast but less secure
  2) 768 bits- High commercial grade, medium speed, good
security
  3) 1024 bits- "Military" grade, slow, highest security
Choose 1, 2, or 3, or enter desired number of bits:
```

Kontakt:
Kontakt:
Andreas Matthias
matthias@hrz.uni-kassel.de
Tel. 804 2281

Geben Sie an dieser Stelle „3“ ein, um die größte Schlüssellänge auszuwählen.

Nun fragt PGP nach Ihrem Namen:

```
You need a user ID for your public key.  The desired form
for this user ID is your name, followed by your E-mail
address enclosed in, if you have an E-mail address.
For example:  John Q. Smith <12345.6789@compuserve.com>
Enter a user ID for your public key:
```

Geben Sie ihn in der angegebenen Form ein, zum Beispiel:

```
Henriette Muster <hmuster@student.uni-kassel.de>
```

Die GhK-CA schreibt bestimmte Regeln für die Namenswahl vor. Bitte lesen Sie in der Policy der GhK-CA nach, wie der Name gebildet werden muss, oder erkundigen Sie sich direkt bei der CA.

Nun kommt der geheime Teil - Ihre Passphrase, die sicherstellt, dass niemand auf Ihren privaten Schlüssel zugreifen kann außer Ihnen selbst:

```
You need a pass phrase to protect your RSA secret key.
Your pass phrase can be any sentence or phrase and may
have many words, spaces, punctuation, or any other
printable characters.

Enter pass phrase:
```

Geben Sie jetzt einen Satz ein, den Sie sich gut merken können. Er sollte lang genug sein, um nicht erraten werden zu können (ca. 4-6 Wörter), aber nicht zu lang, denn Sie müssen ihn immer wieder eintippen, wenn Sie eine Nachricht ver- oder entschlüsseln wollen. Und wenn der Satz zu lang ist, wird das ermüdend. Also wählen wir den Beispielsatz (den Sie bitte nicht verwenden!):

```
Das ist eine Beispiel-Passphrase
```

Dann werden Sie aufgefordert, die Passphrase noch mal einzugeben. Tun Sie das.

Nun verlangt PGP die Eingabe einiger zufälliger Tasten, anhand derer das Programm eine Zufallszahl erzeugt:

```
We need to generate 600 random bits.  This is done by
measuring the time intervals between your keystrokes.
Please enter some random text on your keyboard until you
hear the beep:
 600
```

Tippen Sie etwas Text ein, aber achten Sie darauf, dass Sie nicht immer denselben Buchstaben drücken (oder gar festhalten) weil das Ergebnis sonst nicht zufällig genug sein könnte. Tippen Sie einfach irgend einen Satz, der Ihnen einfällt, zum Beispiel den vorigen Satz in

dieser Beschreibung. Tippen Sie solange, bis der Zähler, der bei „600“ beginnt, abgelaufen ist.

Tippen Sie einfach irgend einen Satz, der Ihnen einfällt, zum Beispiel den

Dann hören Sie einen Signalton und sehen die Meldung:

```
0 * -Enough, thank you.
```

Nun beginnt das Erzeugen des Schlüsselpaares. Wenn die Schlüssel fertig sind, sehen Sie:

```
Pass phrase is good.  Just a moment....
Key signature certificate added.
Key generation completed.
```

Ändern der Passphrase

Wenn Sie irgendwann Ihre Passphrase ändern wollen, dann können Sie das mit

pgp -ke *benutzername*

tun. Sie werden dann einmal die alte und zweimal die neue Passphrase eingeben müssen. Dabei wird kein neuer Schlüssel erzeugt, so dass alle Ihre Kommunikationspartner weiterhin Ihren öffentlichen Schlüssel verwenden können.

Ansehen des eigenen Schlüssels

Der öffentliche PGP-Schlüssel muss möglichst weit verbreitet werden, um wirksam zu sein. Zunächst sollten Sie ihn in eine Datei speichern, zum Beispiel in die Datei „pubkey.asc“ in Ihrem Homeverzeichnis. Geben Sie dazu ein:

pgp -kxa *benutzername*

Sie werden dann gefragt:

```
Extract the above key into which file?
```

Geben Sie den Dateinamen ein, also in unserem Beispiel:

pubkey

PGP hängt automatisch die Erweiterung „asc“ an, so dass dann im aktuellen Arbeitsverzeichnis ein Schlüssel mit dem gewünschten Dateinamen erzeugt wird. So sieht er aus:

```
Type Bits/KeyID    Date       User ID
pub  1024/3DCE65ED 1998/07/29 Henriette Muster
<hmuster@student.uni-kassel.de>

-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3i

mQCNBzW/Q1AAAAEEALVZYQPFJqFEhPnqZNvhFodNn6eu1SVYJUVZbqrKWGZb/oPV
xZpKnACVRwOOTSnzc/DRIAsNCzbJ96u5L75RS5gB7LD4yjRfT3pMwW7m0eYnTEvt
Cwb+e4RV+gk+UMfvui1G7orSLIAIguMAbRwQcyso1e7r+pW5JhfKktc9zzXtAAUR
tDBDR0ktU2VddmVyIEFkbWluaXN0cmF0b3IgPGNnaUBocnoudW5pLWthc3NlbC5k
ZT6JAJUDBRA1v0NQF8qS1z3PNe4BAe55A/9Fkv4UupB7D3D+5Irfe/W3AFq1Mqy4
mEi+I82rora8Q/PCEojh6Uzs8qqyOv6xgxzzqu6ge0DS0f3Lqt6iTCw+6sqJvCSf
4TjGD3bei/o2Lg==
=73ks
-----END PGP PUBLIC KEY BLOCK-----
```

Vorsicht: Dieser hier abgebildete Schlüssel ist natürlich ungültig und stellt nur ein Beispiel dar! Sie sollten nicht versuchen, ihn für irgendwas zu benutzen.

Der Fingerprint

Der Schlüssel, so wie Sie ihn oben sehen können, ist nur schwer für einen Menschen lesbar. Will man überprüfen, ob der Schlüssel, den man vor sich liegen hat, *wirklich* der Schlüssel ist, der einer bestimmten Person gehört, so muss man die Schlüssel irgendwie miteinander vergleichen können. Dazu dient der *Fingerprint* (Fingerabdruck) des Schlüssels. Ein Fingerprint besteht immer aus 16 Zahlen im hexadezimalen Zahlensystem – Sie müssen jedoch das hexadezimale System nicht verstehen, um den Fingerprint benutzen zu können: Sie vergleichen einfach die sechzehn Ziffern- und Buchstabenfolgen miteinander. Stimmen sie überein, so ist es sehr wahrscheinlich, dass die Schlüssel selber es auch tun. Man kann den Fingerprint also als eine lesbare „Kurzfassung“ des eigentlichen Schlüssels ansehen.

Unter Unix lassen Sie sich den Fingerprint eines Schlüssels mit der Option „-kvc“ von PGP anzeigen:

```
pgp -kvc hmuster
```

Sie werden eine Ausgabe ähnlich der folgenden sehen:

```
Type Bits/KeyID    Date       User ID
pub  1024/3DCE65ED 1998/07/29 Henriette Muster <hmuster@student.uni-kassel.de>
            Key fingerprint = DA 52 41 C0 19 F3 A4 F1  49 DA 1B 22 0C 33 14 77
1 matching key found.
```

Natürlich werden bei Ihnen andere Zahlen aufgelistet.

Sie könnten nun fragen, wozu brauche ich einen Fingerprint-Vergleich, wenn es doch die CA gibt, die die Identität des Schlüsselinhabers bestätigt? Darauf gibt es verschiedene Antworten:

- Erstens hat nicht jeder PGP-Benutzer auch einen zertifizierten Schlüssel. Sie werden immer wieder auf Kommunikationspartner stoßen, die Ihnen einen Schlüssel per eMail zusenden, der nicht zertifiziert ist.
- Darüber hinaus wollen wir uns daran erinnern, dass die Zertifizierung darin besteht, dass der Benutzerschlüssel mit dem Schlüssel der jeweiligen CA signiert wird. Doch woher nehmen Sie die Sicherheit, dass der Schlüssel, mit dem angeblich die CA zertifiziert hat, auch wirklich der CA-Schlüssel ist?

Zur Lösung diese Probleme setzt man eben den Fingerprint ein. Sie können jederzeit prüfen, ob ein Schlüssel, der sich für den Schlüssel der CA ausgibt, auch wirklich der richtige ist, indem sie seinen Fingerprint mit dem veröffentlichten Fingerprint des CA-Schlüssels vergleichen. Natürlich müssen Sie, damit dieses Verfahren Sinn macht, den Fingerprint aus einer *anderen* Quelle beziehen als den fraglichen Schlüssel; denn wer unterwegs vielleicht den Schlüssel gefälscht hat, kann auch den Fingerprint fälschen!

Sie können sich den Fingerprint eines Bekannten zum Beispiel am Telefon vorlesen lassen, wenn Sie seine Stimme identifizieren können; oder sie können ihn seiner Visitenkarte entnehmen o.ä. Die GhK-CA und auch die anderen CA's im DFN-Verbund veröffentlichen außerdem ihre Fingerprints in ihren gedruckten Publikationen: Sie werden unseren Fingerprint zum Beispiel zusammen mit der Kontaktadresse in jeder CA-Publikation wiederfinden. Damit können Sie dann leicht überprüfen, ob eine Zertifizierung wirklich von der GhK-CA vorgenommen wurde oder nicht.

Erzeugen eines Widerrufs-Zertifikats

Eine besondere Bedeutung kommt dem Widerrufs-Zertifikat zu: damit können Sie, falls mal Ihr Schlüssel abhanden kommen sollte, Ihre Kommunikationspartner informieren, dass sie Ihren Schlüssel nicht mehr verwenden sollen.

Um einen solchen *Revocation Key* zu erzeugen, brauchen Sie einen funktionierenden PGP-Schlüssel, der aber naturgemäß nicht mehr verfügbar ist, wenn er vorher verloren wurde. Um nicht in diese Zwickmühle zu geraten, sollten Sie ganz am Anfang zusammen mit Ihrem Schlüssel auch *gleich* den Widerrufs-Schlüssel erzeugen und separat vom richtigen Schlüssel aufheben. Das ist der Grund, weshalb die GhK-CA von Ihnen verlangt, bei der Zertifizierung auch gleich einen Widerrufs-Schlüssel mitzuliefern. Wir archivieren diesen Schlüssel und können damit, falls es einmal notwendig werden sollte, Ihren verlorenen oder missbrauchten Schlüssel widerrufen. Der Widerrufsschlüssel eignet sich *nicht* zum Entschlüsseln oder Signieren von Nachrichten: die CA kann damit also keinesfalls Ihre Daten ausspionieren. Für den Einsatz dieses Schlüssels durch die CA gibt es außerdem strenge Regeln, die in der Policy festgehalten sind und die Sie dort jederzeit nachlesen können.

Ein Widerrufsschlüssel (oder Widerrufs-Zertifikat) ist nichts anderes, als ein bereits widerrufener Public Key. Um einen solchen Schlüssel zu erzeugen, müssen Sie also „probeweise“ Ihren Schlüssel widerrufen, den widerrufenen Schlüssel speichern, und anschließend, da sie Ihren Schlüssel ja noch verwenden wollen, den früheren, nicht-widerrufenen Schlüssel wiederherstellen. Klingt umständlich und ist es auch. Im folgenden werden wir sehen, wie man das trotzdem sicher erledigen kann.

Zunächst müssen Sie eine Sicherungskopie Ihres Schlüssels erzeugen. Unter Unix liegen alle Ihre Schlüsseldaten in einem Verzeichnis .pgp in Ihrem Homeverzeichnis. Dieses müssen wir also sichern. Wechseln Sie zunächst mit

```
cd
```

in Ihr Homeverzeichnis und geben Sie dann ein:

```
tar cvf pgp-backup.tar .pgp
```

Nun wird Ihr PGP-Verzeichnis in eine Datei mit dem Namen pgp-backup.tar gesichert.

Jetzt können wir den Schlüssel widerrufen. Geben Sie folgendes ein:

```
pgp -kd hmuster
```

(ersetzen Sie „hmuster“ natürlich durch den Namen, den Sie bei der Erzeugung Ihres Schlüssels für sich gewählt haben). PGP fragt Sie:

```
Key for user ID: Henriette Muster <hmuster@student.uni-kassel.de>
1024-bit key, key ID 3DCE65ED, created 1998/07/29

Do you want to permanently revoke your public key
by issuing a secret key compromise certificate
for "Henriette Muster <hmuster@student.uni-kassel.de>" (y/N)?
```

Wenn Sie der Ausgabe entnehmen können, dass das Ihr soeben generierter eigener Schlüssel ist, geben Sie „y“ ein. Sie werden dann aufgefordert, Ihre Passphrase einzugeben:

```
You need a pass phrase to unlock your RSA secret key.
Key for user ID: Henriette Muster <hmuster@student.uni-kassel.de>
1024-bit key, key ID 3DCE65ED, created 1998/07/29

Enter pass phrase:
```

Geben Sie die Passphrase ein. PGP wird antworten:

```
Key compromise certificate created.
```

Sie haben nun Ihren Widerrufsschlüssel erzeugt. Wir müssen diesen Schlüssel nur noch in eine Datei extrahieren:

```
pgp -kxa hmuster
```

Sie werden jetzt nach einem Dateinamen gefragt. Suchen Sie sich einen aus. Wir empfehlen einen Namen zu wählen, der klarmacht, worum es sich handelt, z.B.: hmuster@student.uni-kassel.de-revocation-key.asc. Diese Datei müssen Sie bei der Zertifizierung Ihres Schlüssels mitbringen. Zusätzlich empfiehlt es sich, sie auf einer Diskette an einem sicheren Ort aufzubewahren, möglichst getrennt von Ihrem eigentlichen Schlüssel.

Jetzt müssen wir noch aufräumen und den gültigen Schlüssel wiederherstellen:

```
tar xvf pgp-backup.tar
```

Die Sicherungskopie Ihres PGP-Verzeichnisses wird zurückgespielt. Anschließend löschen Sie die Sicherungsdatei:

```
rm pgp-backup.tar
```

Veröffentlichen des öffentlichen Schlüssels

Um die Veröffentlichung Ihres Schlüssels müssen Sie sich nicht kümmern, wenn Sie Ihren Schlüssel von der GhK-CA zertifizieren lassen: die CA wird die nötigen Schritte unternehmen, um Ihren Schlüssel möglichst weit verfügbar zu machen.

Sollten Sie unabhängig von der GhK-CA Ihren Schlüssel selber veröffentlichen wollen, dann lesen Sie bitte weiter.

Weltweite Veröffentlichung

Wie bringt man den eigenen Schlüssel unter die Leute? Zunächst sollte man ihn an einen öffentlichen Keyserver schicken, zu Beispiel an: pgp-public-keys@keys.de.pgp.net. Von dort können ihn andere Benutzerinnen dann abfragen.

Um Ihren Schlüssel auf dem Keyserver abzulegen, erzeugen Sie eine eMail der folgenden Form:

```
From: hmuster@student.uni-kassel.de
 To: pgp-public-keys@keys.de.pgp.net
 Subject: add

 Type Bits/KeyID    Date       User ID
 pub  1024/3DCE65ED 1998/07/29 Henriette Muster <hmuster@student.uni-
kassel.de>

 -----BEGIN PGP PUBLIC KEY BLOCK-----
 Version: 2.6.3i

 mQCNBzW/Q1AAAAEEALVZYQPFJqFEhPnqZNvhFodNn6eu1SVYJUVZbqrKWGZb/oPV
 xZpKnACVRwOOTSnzc/DRIAsNCzbJ96u5L75RS5gB7LD4yjRfT3pMwW7m0eYnTEvt
 Cwb+e4RV+gk+UMfvui1G7orSLIAIguMAbRwQcyso1e7r+pW5JhfKktc9zzXtAAUR
 tDBDR0ktU2VddmVyIEFkbWluaXN0cmF0b3IgPGNnaUBocnoudW5pLWthc3NlbC5k
 ZT6JAJUDBRA1v0NQF8qS1z3PNe4BAe55A/9Fkv4UupB7D3D+5Irfe/W3AFq1Mqy4
 mEi+I82rora8Q/PCEojh6Uzs8qqyOv6xgxzzqu6ge0DS0f3Lqt6iTCw+6sqJvCSf
 4TjGD3bei/o2Lg==
 =73ks
 -----END PGP PUBLIC KEY BLOCK-----
```

Der Keyserver speichert dann Ihren Schlüssel und verteilt ihn außerdem an eine ganze Reihe anderer Keyserver weltweit. Von dort kann ihn jeder Benutzer abfragen mit einer eMail der Form:

```
From: nochnmuster@student.uni-kassel.de
To: pgp-public-keys@keys.de.pgp.net
Subject: get hmuster@hrz.uni-kassel.de
```

Beachten Sie, dass die Kommandos „add“ und „get“, die der Keyserver versteht, im Subject der Nachricht angegeben werden müssen!

Veröffentlichung im HRZ

Die GhK-CA betreibt einen eigenen Bereich auf dem WWW-Server, auf dem Ihre Schlüssel abgelegt werden können. Um sicherzustellen, dass diese Schlüssel ordentlich zertifiziert worden sind, und um den Mißbrauch dieses Dienstes auszuschließen, gibt es jedoch keine Möglichkeit, mit der Sie selber einen Schlüssel dort veröffentlichen könnten. Die Veröffentlichung auf dem WWW-Server der GhK erfolgt automatisch nach der Zertifizierung eines Schlüssels durch die GhK-CA. Eine Veröffentlichung ohne Zertifizierung ist nicht möglich.

Der Ort, wo alle zertifizierten Schlüssel zu finden sind, ist:

```
http://www.uni-kassel.de/security/pgp/pubkeys
```

Neue Schlüssel zum eigenen Keyring hinzufügen

Um Nachrichten verschlüsseln zu können, müssen Sie erst den öffenlichen Schlüssel der Empfängerin in Ihrem Keyring ablegen. Dieser Keyring ist eine Datei, die von PGP selber verwaltet wird, und die alle Schlüssel enthält, die Ihnen bekannt sind. Wenn Sie also an jemand eine verschlüsselte Nachricht schreiben wollen, dann müssen Sie erst dessen öffentlichen Schlüssel in Erfahrung bringen (zum Beispiel indem Sie einen Keyserver abfragen).

Nehmen wir an, Henriette Muster möchte an Andreas Matthias im HRZ eine Nachricht schreiben. Zuerst fragt sie den Keyserver nach dem öffentlichen Schlüssel von Andreas Matthias, indem sie folgende eMail verschickt:

```
From: hmuster@student.uni-kassel.de
To: pgp-public-keys@keys.de.pgp.net
Subject: get matthias@hrz.uni-kassel.de
```

Alternativ könnte sie auch unter

```
http://www.uni-kassel.de/security/pgp/pubkeys/
```

nachsehen, wo dieser Schlüssel auch zu finden ist. Sie speichert ihn in ihr Homeverzeichnis, in eine Datei mit dem Namen „matthias.schluessel.asc“. Der Name ist natürlich frei wählbar. Jetzt

muss sie diesen Schlüssel ihrem Keyring hinzufügen. Das geht mit folgendem Kommando:

```
pgp -ka matthias.schluessel.asc
```

PGP antwortet zunächst:

```
Looking for new keys...
pub  1024/802DBD35 1998/07/15  Andreas Matthias

Checking signatures...
pub  1024/802DBD35 1998/07/15 Andreas Matthias
sig!      802DBD35 1998/07/15  Andreas Matthias


Keyfile contains:
   1 new key(s)
```

Das heißt: das Hinzufügen war erfolgreich, und Henriette Muster kann diesen Schlüssel jetzt verwenden, um an Andreas Matthias zu schreiben. PGP braucht allerdings noch einige Angaben:

```
One or more of the new keys are not fully certified.
Do you want to certify any of these keys yourself (y/N)?
```

Hier will das Programm wissen, wie wahrscheinlich es ist, dass der soeben dem Keyring hinzugefügte Schlüssel wirklich der von Andreas Matthias ist. PGP fragt die Beispielperson Henriette Muster hier also, ob sie selber den Schlüssel von Andreas Matthias als gültig verifizieren möchte. Sie hat den Schlüssel aus zwei unabhängigen Quellen bekommen (WWW und Keyserver), hat aber nie direkt mit Andreas Matthias die Fingerprints verglichen. Deshalb sollte sie diesen Schlüssel jetzt *nicht* zertifizieren, sondern das dann nachholen, wenn sie sicher ist, dass es der richtige ist. Sie antwortet also auf die obige Frage mit „nein“:

```
n
```

Damit ist der (als unsicher gekennzeichnete) Schlüssel in Henriette Musters Keyring aufgenommen. Sie kann ab jetzt verschlüsselte Nachrichten an Andreas Matthias verschicken.

Später kann Henriette Muster, wenn sie mit Andreas Matthias gesprochen und die Fingerprints verglichen hat, die Verifizierung seines Keys in ihrem Keyring nachholen. Dazu gibt sie das Kommando:

```
pgp -ks matthias
```

ein und beantwortet die Fragen gemäß dem PGP-Handbuch.

Hintergrund: Vertrauen Sie niemals Schlüsseln, die Sie über eMail erreichen, oder die Sie auf einem öffentlichen Keyserver finden. Sie können sie für einfache, unkritische Mitteilungen durchaus verwenden, nicht aber um wirklich sensitive Daten, zum Beispiel Passwörter, zu übertragen. **Jeder kann eine eMail fälschen**, so dass es aussieht, als käme sie von einem anderen Absender.

Der einzige Weg, um wirklich sicher zu sein, daß der Schlüssel der richtige ist, ist direkt mit dem Besitzer zu sprechen (zum Beispiel übers Telefon) und sich den Fingerprint vorlesen zu lassen und zu vergleichen.

Alternativ können Sie auch der Aussage eines Dritten vertrauen, der diese Prüfung für Sie bereits vorgenommen hat: so können Sie, wenn Sie wollen, den von der GhK-CA zertifizierten Schlüsseln vertrauen.

Am besten ist es, Schlüssel aus verschiedenen Quellen zu vergleichen. Wenn sowohl der Keyserver, als auch der WWW-Server der GhK, und evtl. eine eMail oder Visitenkarte des Gesprächspartners denselben Schlüssel enthalten, dann kann man relativ sicher sein, dass dieser Schlüssel stimmt.

Dateien verschlüsseln

Wir beschäftigen uns hier nur mit dem Verschlüsseln von Dateien mit PGP von der Unix-Kommandozeile aus. Zum Verschlüsseln von eMail-Nachrichten, was der häufigere Fall sein dürfte, lesen Sie bitte

den Artikel, der sich speziell auf das von Ihnen verwendete Mailprogramm bezieht.

Nehmen wir an, Henriette Muster möchte eine Textdatei `brief.txt` verschlüsseln. Dazu gibt sie folgendes ein:

```
pgp -e brief.txt hmuster
```

Der zweite Parameter ist der „Empfänger“, für den die Datei verschlüsselt werden soll. Da Henriette Muster die Datei nur für sich verschlüsselt, gilt sie selber als „Empfängerin“ und trägt ihren eigenen Namen ein. Das Kommando läuft ohne weitere Rückfrage durch und es entsteht im selben Verzeichnis eine neue Datei `brief.txt.pgp`, die den verschlüsselten Brief enthält. Sinnvollerweise sollte man jetzt noch die unverschlüsselte Version löschen:

```
rm brief.txt
```

Will Henriette Muster später den Brief entschlüsseln, so gibt sie bloß:

```
pgp brief.txt.pgp
```

ein, gefolgt von ihrer Passphrase. Damit entsteht wieder die ursprüngliche Datei `brief.txt`.

Dateien signieren

Man muss nicht immer verschlüsseln. Manchmal möchte man nicht den Inhalt geheim halten, sondern nur sicherstellen, dass er nicht von einem unbefugten Dritten verändert wurde (zum Beispiel von jemand, der das eigene Passwort ausgespäht hat). Dazu kann man Dateien signieren. Am praktischsten ist es in so einem Fall, die Signatur in einer getrennten Datei zu halten. Man hat also nach dem Signieren *zwei* Dateien: die unveränderte Originaldatei und zusätzlich eine Signaturdatei. Mit PGP kann man in der Folge immer prüfen, ob die Signatur zu der Datei passt, d.h. ob die Datei seit dem Signieren verändert worden ist oder nicht. Da die Signatur separat abgelegt wird, kann man die Datei jedoch in jedem Fall normal weiterbearbeiten.

Um eine Datei `brief.txt` zu signieren, gibt Henriette Muster folgendes ein:

```
pgp -sba brief.txt
```

Es entsteht hierbei eine neue Datei, `brief.txt.asc`, die die Signatur enthält. Sie sieht etwa so aus:

```
-----BEGIN PGP MESSAGE-----
Version: 2.6.3i

iQCVAwUAOJNVFuqVETSALbx1AQE1rwP/ZLd29ce9RWLRQpylrkPCUCQ+02CkdZ1q
VpJdx3N+rWQNVhcWSLhIICrwcvckCt/VXTU7Eojw/X9yphgcjwj/Zby1AU/xD2b0
XLeMh5AqRELb0dX4bS3blqMjiQBDb2geCpBG9SHPtSx8bUNYP/Icn6zgxjVezYIQ
RAKA7UP5zHg=
=Fc73
-----END PGP MESSAGE-----
```

Möchte man überprüfen, ob die Datei `brief.txt` verändert wurde, muss man nur eingeben:

```
pgp brief.txt
```

PGP findet automatisch die Signaturdatei, wenn sie sich im selben Verzeichnis wie die Originaldatei befindet und prüft die beiden auf Übereinstimmung. Wenn die Datei mit der Signatur übereinstimmt, dann bekommen wir die Meldung:

```
File 'brief.txt.$00' has signature, but with no text.
Text is assumed to be in file 'brief.txt'.

Good signature from user "Henriette Muster <hmuster@student.uni-
kassel.de>".
Signature made 2000/01/29 21:01 GMT using 1024-bit key, key ID 3DCE65ED
```

Wenn die Datei modifiziert wurde und nicht mehr mit der Signatur übereinstimmt, sagt PGP stattdessen:

```
File 'brief.txt.$00' has signature, but with no text.
Text is assumed to be in file 'brief.txt'.

WARNING: Bad signature, doesn't match file contents!

Bad signature from user "Henriette Muster <hmuster@student.uni-
kassel.de>".
Signature made 2000/01/29 21:01 GMT using 1024-bit key, key ID 3DCE65ED
```

Um die Signatur nach einer Veränderung der Datei wieder dem veränderten Inhalt anzupassen, löschen Sie die Signaturdatei (in diesem Beispiel `brief.txt.asc`) und erzeugen Sie sie, wie oben beschrieben, neu.

Weitere Optionen

Man kann eine Datei auch gleichzeitig verschlüsseln und signieren, was in der Regel Sinn macht. Dazu verwendet man die Option „-se“ von PGP. Im wesentlichen läuft der Vorgang wie oben unter „Verschlüsseln“ beschrieben ab.

PGP hat noch eine ganze Reihe weiterer Optionen. In den Literaturhinweisen am Ende dieses Artikels finden Sie das PGP-Handbuch erwähnt, das alle Möglichkeiten ausführlich beschreibt. Eine Kurzfassung kann man bekommen, indem man

```
pgp -h
```

und

```
pgp -k
```

aufruft. Hier als Referenz noch mal alle Optionen auf einen Blick, so wie sie PGP selber auflistet:

```
Generate new key pair:  pgp -kg    [keybits]
Add key:                pgp -ka     keyfile           [keyring]
Extract key:            pgp -kx[a]  userid   keyfile  [keyring]
View key(s):            pgp -kv[v] [userid]           [keyring]
View fingerprint:       pgp -kvc   [userid]           [keyring]
Check & view in detail: pgp -kc    [userid]           [keyring]
Remove userid or key:   pgp -kr     userid            [keyring]
                        (Repeat for multiple userids on a key)
Edit trust params:      pgp -ke     userid            [keyring]
Add another userid:     pgp -ke     your_userid       [keyring]
Edit passphrase:        pgp -ke     your_userid       [keyring]
Sign a key in pubring:  pgp -ks other_id [-u sign_id] [keyring]
Remove a sig from key:  pgp -krs    userid            [keyring]
Revoke, dis/enable:     pgp -kd     userid            [keyring]

Encrypt:                pgp -e[a]  textfile TO_id [TO_id2 TO_id3...]
Sign:                   pgp -s[a]  textfile                          [-u MY_id]
Sign & encrypt:         pgp -se[a] textfile TO_id [TO_id2 TO_id3...][-u MY_id]
Make detached cert:     pgp -sb[a] [+clearsig=on] mainfile           [-u MY_id]
  (Can do binaries)     (clearsig=on may be set in CONFIG.TXT)
Encrypt with IDEA only: pgp -c     textfile
Decrypt or check sig:   pgp [-d] [-p] cryptogram
                        (-d to keep pgp data, -p for original file name)
Check detached cert:    pgp certfile [mainfile]
                        (If root of filenames are the same omit [mainfile])

Use [a] for ASCII output
Use [-o outfile] to specify an output file
Use [-@ textfile] to specify additional userids when encrypting
Use [-z"pass phrase"] to specify your pass phrase
Use [+batchmode] for errorlevel returns
Use [f] for stream redirection ( pgp -f[ARGS] outfile )
Use [w] to wipe plaintext file (encryption operations)
Use [m] to force display of plaintext only (no output file)
Use [t] to alter line endings for unix, etc.


Key management functions:
To generate your own unique public/secret key pair:
   pgp -kg
To add a key file's contents to your public or secret key ring:
   pgp -ka keyfile [keyring]
To remove a key or a user ID from your public or secret key ring:
   pgp -kr userid [keyring]
To edit your user ID or pass phrase:
   pgp -ke your_userid [keyring]
To extract (copy) a key from your public or secret key ring:
   pgp -kx userid keyfile [keyring]
To view the contents of your public key ring:
   pgp -kv[v] [userid] [keyring]
To check signatures on your public key ring:
   pgp -kc [userid] [keyring]
To sign someone else's public key on your public key ring:
   pgp -ks her_userid [-u your_userid] [keyring]
To remove selected signatures from a userid on a keyring:
   pgp -krs userid [keyring]
```

Literatur

- Deutsche Kurzübersicht der PGP-Kommandos:
  `http://www.uni-kassel.de/security/pgp/manuals/de.txt`
- Erster Teil des Original-Manuals von PGP:
  `http://www.uni-kassel.de/security/pgp/manuals/pgpdoc1.txt`
- Zweiter Teil des Original-Manuals von PGP:
  `http://www.uni-kassel.de/security/pgp/manuals/pgpdoc2.txt`
- Anleitung zur Benutzung von Keyservern:
  `http://www.uni-kassel.de/security/pgp/manuals/keyserv.txt`

Benutzung von PGP mit Pine

Pine-Konfiguration

In Ihrem Homeverzeichnis befindet sich, wenn Sie „pine“ benutzen, eine Datei „.pinerc“. Laden Sie diese in einen Editor und suchen Sie nach der folgenden Stelle:

```
# This variable takes a list of programs that message text is piped into
# after MIME decoding, prior to display.
display-filters=

# This defines a program that message text is piped into before MIME
# encoding, prior to sending
sending-filters=
```

Nehmen Sie die folgenden Einstellungen vor (Sie können auch den roten Abschnitt unten mit „Cut&Paste“ in Ihren Editor übernehmen und direkt in die „.pinerc“ einfügen - natürlich erst nachdem Sie die alten Einstellungen oben gelöscht haben):

```
# This variable takes a list of programs that message text is piped into
# after MIME decoding, prior to display.
display-filters= _BEGINNING("-----BEGIN PGP PUB")_ /usr/aix4-PD/bin/pgp -ka _T
_TMPFILE_,
         _BEGINNING("-----BEGIN PGP SIG")_ /usr/aix4-PD/bin/pgp -m,
         _BEGINNING("-----BEGIN PGP MES")_ /usr/aix4-PD/bin/pgp

# This defines a program that message text is piped into before MIME
# encoding, prior to sending
sending-filters= /usr/aix4-PD/bin/pgpencrypt -feast _RECIPIENTS_,
         /usr/aix4-PD/bin/pgpsign -fast
```

Technischer Hinweis: Die zwei Dateien /usr/aix4-PD/bin/pgpencrypt und /usr/aix4-PD/bin/pgpsign sind in Wirklichkeit nur Links auf das Programm „pgp“.

Erzeugen der Schlüssel

Jeder Benutzer hat, wie an anderer Stelle beschrieben wurde, *zwei* Schlüssel für die PGP-Kommunikation: einen privaten und einen öffentlichen Schlüssel (*private* und *public key*). Diese beiden müssen nur einmal erzeugt werden. Der öffentliche Schlüssel wird daraufhin möglichst weit verteilt, denn ohne ihn kann einem niemand eine verschlüsselte Nachricht schicken. Der private hingegen muss sorgfältig (am besten auf einer Diskette) aufbewahrt werden und darf niemals in fremde Hände geraten!

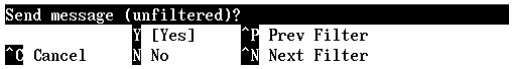
Kontakt:
Andreas Matthias
matthias@hrz.uni-kassel.de
Tel. 804 2281

Wie Sie ein Schlüsselpaar für die Benutzung von PGP erzeugen, ist in einem anderen Artikel ausführlich beschrieben. Sie können diesen Schritt bei Bedarf auch von der GhK-CA durchführen lassen.

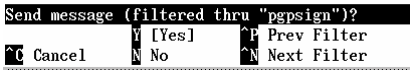
Nachrichten verschlüsseln und signieren

Nachrichten muss man immer mit dem öffentlichen Schlüssel des Empfängers verschlüsseln; dieser kann die Nachricht dann mit seinem privaten Schlüssel entschlüsseln und lesen. Das Entschlüsseln geschieht in der hier vorgestellten Konfiguration von Pine und PGP automatisch, wobei Pine einmal nach der geheimen Passphrase fragt, wenn man eine verschlüsselte Nachricht lesen möchte. Das Verschlüsseln geschieht nach dem Schreiben und kurz vor dem Versenden der Nachricht in Pine.

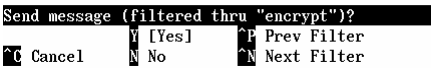
Nehmen wir an, Henriette Muster möchte eine verschlüsselte Nachricht an Andreas Matthias schreiben. Dazu muss sie sich zuerst dessen *public key* besorgen, was sie ja bereits getan hat (vgl. den Abschnitt „Neue Schlüssel zum eigenen Keyring hinzufügen“). Jetzt kann es losgehen: sie ruft `pine` auf, und schreibt die Nachricht. Am Ende drückt sie, wie in Pine üblich, <Strg-X>, um die Nachricht zu versenden. Pine fragt dann:



Die Absenderin kann jetzt entscheiden, ob sie die Nachricht unverschlüsselt schicken möchte (in diesem Fall einfach „y“ oder die Eingabetaste drücken), oder ob die Nachricht signiert oder verschlüsselt verschickt werden sollte. Mit <Strg-P> bzw. <Strg-N> kann sie die verschiedenen Arten von Verschlüsselung, die zur Verfügung stehen, durchblättern. Es erscheint dann in der Statuszeile von Pine:



oder, nach dem nächsten Drücken von <Strg-P> bzw. <Strg-N>:



Henriette Muster kann nun entscheiden, ob sie die Nachricht nur signieren will (pgpsign) oder verschlüsseln und signieren (encrypt). Beachten Sie, dass verschlüsselte Nachrichten automatisch auch signiert werden.

Wenn die Wahl erfolgt ist, muss sie „y“ drücken, um die Nachricht in der gewählten Form zu versenden.

Die Schritte zu Ihrem persönlichen Zertifikat

Einleitung

Wie schon an anderer Stelle in diesem Heft erwähnt, nimmt am 15.2.00 die GhK-CA ihre Arbeit auf. Sie können dann einen Schlüssel erzeugen lassen oder selber erzeugen, diesen zertifizieren lassen, und damit in die Welt der sicheren Datenübertragung eintreten.

Die Frage ist natürlich, wie geht das konkret? Andere Artikel in dieser Ausgabe erläutern die Grundlagen der Public-Key-Verschlüsselung, die Benutzung von PGP und die Konfiguration Ihrer Mailsoftware. In diesem Artikel wollen wir uns auf den Ablauf einer Zertifizierung durch die GhK-CA konzentrieren. Wir geben hier nur eine Übersicht über das Verfahren - die Details kann man in der Policy (Zertifizierungsrichtlinie) der GhK-CA nachlesen.

Im Moment zertifiziert die GhK-CA nur PGP-Schlüssel, die nach dem RSA-Algorithmus erzeugt wurden. Wenn andere Verschlüsselungsverfahren sich weiter verbreiten sollten, dann werden wir diese natürlich auch berücksichtigen.

Die Zertifizierung von SSL-Schlüsseln für den sicheren WWW-Server-Betrieb ist für die nahe Zukunft geplant, die GhK-CA ist allerdings noch nicht für solche Zertifizierungen vom DFN anerkannt worden. Falls Sie Interesse an einer SSL-Zertifizierung haben, verfolgen Sie bitte die Meldungen in http://www.uni-kassel.de/hrz/aktuelles oder auf den WWW-Seiten der GhK-CA.

Erzeugen eines PGP-Schlüssels

Damit wir Ihren Schlüssel zertifizieren können, müssen Sie zuerst einen haben. Es gibt zwei Wege, auf denen Sie zu Ihrem persönlichen PGP-Schlüssel gelangen können:

- Sie können selber einen Schlüssel mit Hilfe der Dokumentation in dieser Ausgabe erzeugen und diesen zur Zertifizierung mitbringen, oder
- Sie können sich von der GhK-CA einen Schlüssel erzeugen lassen.

Kontakt:
Uni-Gh Kassel Certification Authority
Hochschulrechenzentrum
ghk-ca@hrz.uni-kassel.de
Tel. 804 2032
WWW: http://www.uni-kassel.de/security/ghk-ca

Da Ihr PGP-Schlüssel ein sehr sicherheitssensitives Datenstück ist, raten wir grundsätzlich davon ab, ihn von jemand anders (und sei es die GhK-CA) erzeugen zu lassen. Falls Sie sich zutrauen, ihn mit Hilfe der Dokumentation selber zu erzeugen, sollten Sie das auch tun. Nur wenn Sie dazu keine Möglichkeit sehen, sollten Sie sich den Schlüssel von der GhK-CA erzeugen lassen. In keinem Fall dürfen Sie einen Dritten mit der Erzeugung Ihres Schlüssels beauftragen - ebenso

wenig, wie Sie jemandem Ihr Passwort oder Ihre Geldkarten-Geheimnummer geben würden!

Der Widerrufsschlüssel

Die GhK-CA ist verpflichtet, ihre Kunden vor der missbräuchlichen Nutzung ihrer Schlüssel zu schützen. Dazu gehört auch, dass Schlüssel, deren privater Teil gestohlen oder verloren wurde, möglichst aus dem Verkehr gezogen werden. Im Falle des Verlustes eines privaten Schlüssels, wird die CA also versuchen, den Schlüssel in der Öffentlichkeit zu widerrufen.

Damit das funktioniert, muss der Benutzer schon bei der Zertifizierung ein Widerrufszertifikat (oder Widerrufsschlüssel, revocation key) bei der CA hinterlegen. Dieser revocation key kann nicht zum Verschlüsseln, Entschlüsseln oder Signieren benutzt werden, d.h. die CA hat dadurch keine Möglichkeit, in Ihren verschlüsselten Datenverkehr einzugreifen. Das einzige was dieser Schlüssel bewirkt, ist, dass man damit andere Benutzer im Netz informieren kann, dass sie den zu diesem Widerrufsschlüssel gehörenden öffentlichen Schlüssel nicht mehr zur Kommunikation benutzen sollen.

Die CA setzt diese Widerrufsschlüssel nach genau festgelegten Regeln ein, die in der Policy (s.o.) genannt sind.

Sie erzeugen einen solchen, für die Zertifizierung erforderlichen Schlüssel dadurch, dass Sie zunächst eine Kopie Ihres normalen Schlüsselpaars machen (z.B. auf eine Diskette) und anschließend mit Ihrer PGP-Software Ihren eigenen Schlüssel widerrufen. Wie das im einzelnen geht, ist bei den detaillierten PGP-Anleitungen in diesem Heft beschrieben.

Dieser widerrufene Schlüssel ist bereits das, was wir brauchen. Speichern Sie ihn auf eine weitere Diskette und bringen Sie diese zur Zertifizierung mit. Zuletzt müssen Sie noch diesen Schlüssel wieder aus Ihrer Schlüsselsammlung löschen und Ihre ursprünglichen (gültigen) Schlüssel von der Sicherungskopie in Ihre PGP-Software zurückspielen. Wie das geht lesen Sie bitte auch in der Anleitung zu Ihrem PGP-Programm nach.

Falls Sie Ihren Schlüssel nicht selber erzeugen, sondern ihn von der CA generieren lassen, dann brauchen Sie sich auch um den Widerrufsschlüssel nicht zu kümmern; die CA wird auch ihn selber erzeugen und in die Schlüsseldatenbank aufnehmen.

Das Anmeldeformular

Für die Anmeldung zur Zertifizierung brauchen Sie natürlich auch ein Anmeldeformular. Sie finden einen Vordruck unter http://www.uni-kassel.de/security/formulare/ oder bei der GhK-CA (s. Kontaktadresse unten). Die Felder müssen gemäß der Policy der GhK-CA ausgefüllt werden. Wenn Sie nicht sicher sind, was Sie eintragen sollten, dann

bringen Sie das Formular nur teilweise ausgefüllt zur Zertifizierung mit, und wir helfen Ihnen gerne an Ort und Stelle.

Der Gang zur CA

Da der Sinn einer CA darin liegt, die Identität des Schlüsselinhabers zu bestätigen, können wir Ihnen leider ein persönliches Erscheinen im Büro der GhK-CA nicht ersparen.

Denken Sie bitte daran, dass diese (vielleicht etwas streng wirkende) Handhabung am Ende Ihrer eigenen Sicherheit dient. Sie können sich im elektronischen Datenverkehr immer darauf verlassen, dass jedes Zertifikat der GhK-CA nach den selben Regeln erstellt und die Identität des Antragstellers genauso gewissenhaft geprüft wurde, wie es auch bei Ihnen der Fall war.

Sie können die GhK-CA zu den angegebenen Öffnungszeiten ohne Anmeldung besuchen; sollte das nicht möglich sein, dann können Sie telefonisch einen Ersatztermin vereinbaren. Bitte machen Sie von der letztgenannten Möglichkeit nur in Ausnahmefällen gebrauch: die GhK-CA wird nur mit einer Viertelstelle betrieben, und wir können deshalb leider keine längeren Öffnungszeiten und nur wenige Ausweichtermine anbieten.

Bei Ihrem Besuch bei der CA sollten Sie folgendes mitbringen:

- Einen gültigen Personalausweis oder Reisepass. Führerscheine, Dienstausweise und ähnliche Ersatzdokumente dürfen wir leider nicht akzeptieren. Ausländische Ausweisdokumente, die nicht eindeutig identifizierbar oder für die Bearbeiter der CA nicht lesbar sind, können ebenfalls nicht angenommen werden. Sprechen Sie in solchen Fällen mit uns und wir werden gemeinsam eine Lösung suchen.
- Das ausgefüllte Antragsformular. Auch hier helfen wir gerne weiter, wenn das Ausfüllen nicht so ganz klappen will.
- Ihren öffentlichen PGP-Schlüssel und Ihren Widerrufsschlüssel (s.o.) in maschinenlesbarer Form. Der Schlüssel darf von der CA nicht zum Zeitpunkt der Zertifizierung aus dem Netz geholt werden; er muss mitgebracht werden. Sinnvoll wäre zusätzlich ein Ausdruck Ihres Fingerprints; das erleichtert uns die Arbeit, ist aber nicht unbedingt erforderlich.
- Falls Sie den Schlüssel von der CA erzeugen lassen wollen, entfällt der letzte Punkt.

Ihre Unterlagen werden von der GhK-CA entgegengenommen. Den Ausweis bekommen Sie zurück, der Antrag und die Schlüsseldiskette verbleiben bei der CA. Nach einigen Tagen Bearbeitungsfrist können Sie Ihren zertifizierten Schlüssel entweder persönlich abholen, oder die GhK-CA schickt ihn Ihnen per verschlüsselter eMail zu. Wenn die GhK-CA den Schlüssel für Sie erzeugt hat, kann sie ihn Ihnen natürlich nicht verschlüsselt zuschicken - in diesem Fall müssen Sie ihn persönlich abholen. Einen Abholtermin können Sie gleich bei Ihrem ersten Besuch mit dem CA-Bearbeiter vereinbaren.

Der zertifizierte Schlüssel

Wenn Sie nach Abschluss des Zertifizierungsverfahrens Ihren Schlüssel wieder in den Händen halten, sollten Sie sofort ihren bisherigen, unzertifizierten Schlüssel durch den von der CA zertifizierten ersetzen. Benutzen Sie für Ihre elektronische Kommunikation nur noch den zertifizierten Schlüssel.

Die von der GhK-CA ausgestellten Zertifikate werden von allen Organisationen in der DFN-Zertifizierungshierarchie akzeptiert. Nach einer festgelegten Zeit, die Ihnen bei der Zertifizierung mitgeteilt wird, läuft Ihr Zertifikat ab. Dieser Zeitraum beträgt maximal zwei Jahre, kann aber, falls Sie nur eine befristete Zugangsberechtigung zu den Systemen des HRZ haben, auch kürzer sein.

Nach Ablauf der Gültigkeitsdauer müssen Sie Ihr Zertifikat verlängern. Die genaue Vorgehensweise zum Verlängern wird in einem anderen Dokument vorgestellt. Wir werden uns bemühen, das Verlängerungsverfahren so abzuwickeln, dass kein weiteres persönliches Erscheinen bei der CA erforderlich wird.

pressum Impressum Impressum Impressum Impressum Impressum Impressum Im

Überarbeitete 2. Auflage Dezember 2000

Herausgeber: Universität Gh Kassel Certification Authority

Hochschulrechenzentrum der GhK
Mönchebergstr. 11
34109 Kassel

eMail: ghk-ca@hrz.uni-kassel.de
WWW: http://www.uni-kassel.de/security/ghk-ca/

Öffnungszeiten: Dienstag 14 – 15 Uhr, Freitag 11 – 12 Uhr
und nach Absprache
Hochschulrechenzentrum Raum 1140
Frau Rosebrock
Telefon: 0561 804-2032

Schlüssel der GhK-CA: eMail:
<ghk-ca@hrz.uni-kassel.de> 2048 / 503D75E9
Fingerprint: 48 BD 83 70 4F 4B 01 80 – 7C C1 E9 0E DA AD AC 34

Zertifizierung:
<ghk-ca@hrz.uni-kassel.de> 2048 / 965334B9
Fingerprint: 9D 00 D5 A0 24 CB 7F A3 - BB 0C 04 6E 4F A9 1B 32

Druck: Hausdruckerei der GhK

sum Impressum Impressum Impressum Impressum Impressum Impressum Impres