Warum sollte ich meinen eigenen Schlüssel unterschreiben?

Über diesen Text

Übersetzung von Michael Uplawski [Homepage] vom 20.07.1997.
Ihr liegt die FAQ "Why should I sign my own public key?" von **Walter Soldierer** zu Grunde.

Alles hier Beschriebene betrifft PGP 2.6.3i (32 BIT), ist aber auch für die PGP 5/6 Versionen zum Verständnis der eigenen Signatur relevant.
Die hier angegebenen Informationen korrespondieren mit dem Kapitel Web of Trust oder "Wie funktioniert das Netz des Vertrauens ?" der PGP 5/6 Anleitung

---

1. Wieso reden wir über Eigensignaturen?

Ein herausragender Vorteil von Verschlüsselungssystemen mit öffentlichen Schlüsseln, wie PGP, ist der Umstand, dass Schlüssel leicht jedermann zugänglich gemacht werden können. Sichere Kanäle sind für die Schlüsselverbreitung nicht erforderlich. Dennoch muss ein PGPBenutzer die Gewissheit haben, dass der öffentliche Schlüssel, den er bekommen hat, wirklich der Person gehört, der er verschlüsselte Post senden will.

Wenn die BenutzerID eines Schlüssels von einer vertrauenswürdigen Person unterzeichnet ist, kann sie als eine gültige ID angesehen werden. Es ist häufig empfehlenswert, alle BenutzerIDs selbst zu signieren. Trotzdem beschreibt Phil Zimmermanns PGPDokumentation (1) diesen Vorgang nicht. Er wird aber in der PGP FAQ (2) empfohlen, genauso in einigen Veröffentlichungen des WWW(3 , 4 , 5).

Der Zweck dieser FAQ ist es, eine allumfassende Erklärung aller Sachverhalte im Zusammenhang mit der Egensignatur von PGPSchlüsseln zu liefern. Dabei soll das Hauptaugenmerk auf Probleme gerichtet sein, die nicht in der PGP-Dokumentation oder der FAQ angesprochen werden.

Ich hoffe, dass die Vermeidung technischer Formulierungen zum leichten Verständnis beiträgt.

2. Kann irgendjemand meine BenutzerIDs manipulieren?

Es gibt kein PGPKommando zum Verändern einer BenutzerID.

Du kannst eine ID lediglich **löschen (kr)** oder eine neue **hinzufügen (ke)** . Während zum Löschen einer ID der geheime Schlüssel und Passphrase (Mantra) nicht erforderlich sind, kann nur der Schlüsselinhaber mit PGP eine neue ID hinzufügen.

Leider gibt es einen "inoffiziellen" Weg, um eine BenutzerID zu verändern, und dazu ist noch nicht einmal PGP erforderlich. Diese Art des Angriffs (**Denial Of Service** , in etwa: "Verweigern des Dienstes"), die ich nicht im Detail beschreiben werde, nutzt den Fakt aus, dass BenutzerIDs in "Schlüsselringen" nicht verschlüsselt sind.

Ein Beispiel. Johns Schlüssel sieht exakt folgendermassen aus **(kvv)** :

Typ Bits/ID Datum Benutzer
öff 1024/ABCD1234 1996/01/01 John Harmless <john@company.com>

Für einen Angreifer wäre es überhaupt kein Problem, ihn so abzuändern:

Typ Bits/ID Datum Benutzer
öff 1024/ABCD1234 1996/01/01 John Harmless <john@ network.net >

Wenn du PGP benutzt, um diesen gefälschten Schlüssel deinem Schlüsselbund hinzuzufügen, wird das auch sofort erledigt.

Wenn sich die OriginalSchlüsselID ABCD1234 vorher bereits in deinem Schlüsselring befand, wird PGP sie nicht austauschen , sondern die vorgetäuschte Adresse als neue BenutzerID hinzufügen :

Typ Bits/ID Datum Benutzer
öff 1024/ABCD1234 1996/01/01 John Harmless <john@company.com>
<john@network.net>

3. Was bewirkt das Signieren in einem Schlüssel?

Eine Signatur sorgt für eine gewisse Sicherheit darüber, dass ein vorhandener Schlüssel von der Person erzeugt worden ist, von der er zu stammen scheint.

Das ist es, was der Unterzeichner bestätigt. Tatsächlich signiert der Unterzeichner nicht den "Schlüssel", sondern die BenutzerID des Schlüsselinhabers (oder eine der IDs, wenn es mehrere sind). Wenn ein Schlüssel signiert wird, berechnet PGP die Kurzfassung einiger Informationen, die für den signierten Schlüssel einzigartig sind. Diese Kurzinformation (digest) nennt man den "MD5Hash", der so etwas wie eine kryptographisch gesicherte "Prüfsumme" ist und die "gehashte" Information repräsentiert. Als Rohmaterial für den MD5Hash benutzt PGP Informationen des öffentlichen RSASchlüssels, die Zeichenfolge der BenutzerID, die unterzeichnet werden soll und andere Daten. Der MD5Hash wird dann mit dem geheimen

Schlüssel des Unterzeichners verschlüsselt. Das Endprodukt schliesslich wird der betreffenden BenutzerID als Signatur angehängt.

Niemand kann eine Signatur fälschen, es sei denn, er besitzt den geheimen Schlüssel des Unterzeichnenden. Wann immer Signaturen angezeigt oder überprüft werden (**kvv, kc, ka**), sucht PGP im öffentlichen Schlüsselbund pubring.pgp nach allen öffentlichen Schlüsseln der Unterzeichner. Sollte keiner dieser Schlüssel gefunden werden, können Signaturen weder angezeigt noch überprüft werden.

Da eine digitale Signatur einzigartige Informationen über den Unterzeichner und die unterzeichnete BenutzerID enthält, können Manipulationen beider Komponenten von PGP entdeckt werden. Unterschriften sind daher eine Vorsichtsmassnahme, um PGP das Auffinden vorgetäuschter BenutzerIDs zu ermöglichen.

4. Wie laufen "Denial Of Service Attacken" ab?

Wenn der Angreifer der Inhaber der falschen EmailAdresse in Johns Schlüssel ist und den manipulierten Schlüssel weit verbreitet, ist es möglich, dass er einen Teil von Johns Post abfängt. Sollten die abgefangenen Nachrichten mit Johns Schlüssel verschlüsselt worden sein (auch mit dem veränderten Schlüssel), ist zwar der Angreifer nicht in der Lage sie zu entziffern, aber zumindest hat er die Informationen zurückgehalten. Das wird **Denial Of Service** (Verweigern des Dienstes) genannt und kann auch durch die Erzeugung und Verbreitung eines völlig neuen Schlüssels mit Johns Namen erreicht werden. Dennoch ist es äusserst unwahrscheinlich, dass diese Imitation Johns echte SchlüsselID besitzen würde:

Typ Bits/ID Datum Benutzer öff 1024/ 0987DCBA 1996/01/01 John Harmless <john@company.com>

Eine bestimmte Art des Angriffs (unter der Bezeichnung " **Dead Beaf** " Attacke zuerst von Paul Leyland beschrieben,) verändert einen Schlüssel nicht nur so, dass er eine falsche BenutzerID zeigt, sondern auch noch eine andere SchlüsselID (6).

Da diese Form der Imitation einen verkehrten Fingerprint haben muss, wird der Angriff durch die Anwendung der Option **kvc** in PGP vereitelt.

5. Wieso schützt das Unterzeichnen meine BenutzerIDs?

Wurde an einer unterschriebenen BenutzerID herumgepfuscht, wird PGP piepen und die Nachricht:

**** BAD SIGNATURE! ****

b.z.w

**** FALSCHE UNTERSCHRIFT ****

anzeigen, wenn dieser Schlüssel deinem Schlüsselring hinzugefügt wird. Die gleiche Warnung taucht auf, wenn die Signatur mit der **kc** Option geprüft wird. Der entschlüsselte MD5Hash passt nicht mehr zur Zeichenkette der BenutzerID. Das sollte dich alarmieren ; du solltest den Schlüssel entfernen und seinen Besitzer informieren.

5.1 Schlüssel auf einem Personal Computer

Jedermann kann Unterschriften und BenutzerIDs aus Schlüsseln in öffentlichen Schlüsselringen entfernen. Ein passender geheimer Schlüssel oder Passphrase ist nicht dazu erforderlich.

Bevor er eine BenutzerID fälscht, entfernt der intelligente Angreifer alle Unterschriften. Signaturen können daher diese Form des Angriffs nicht verhindern!

5.2 Schlüssel auf einem öffentlichen Schlüsselserver

Schlüssel auf öffentlichen Schlüsselservern können durch jedermann manipuliert werden. Ein Angreifer oder Witzbold läd einen Schlüssel, verändert eine spezielle BenutzerID und schickt ihn wieder zum Server, womit er diesem Schlüssel eine neue ID hinzufügt. Natürlich kann auf diesem Wege jede Nachricht transportiert werden, auch ärgerliche Dinge wie: *"Don't use this key any more"* (Benutze diesen Schlüssel nicht mehr), *"This key revoked"* (Dieser Schlüssel wurde widerrufen), *"John Harmless ist ein Narr"* , *"Meine neue EmailAdresse lautet..."* und so weiter...

Der Schlüsselserver überprüft (im Moment noch) nicht die Identität des Senders. Im Gegensatz zu Schlüsselringen auf Personal Computern kann hier gar nichts gelöscht werden. Daher ist noch nicht einmal der Schlüsselinhaber in der Lage, nachgemachte IDs oder die Signaturen von Witzbolden zu entfernen.

Da der Angreifer die Unterschrift des Schlüsselinhabers nicht fälschen kann, sollten alle unsignierten BenutzerIDs mit Argwohn betrachtet werden. Vertraue niemals auf eine EmailAdresse oder eine Nachricht in einer solchen ID! Auch wenn eine ID durch jemanden unterschrieben worden ist, den du nicht kennst, verleiht das der ID nicht mehr Gültigkeit. Der Angreifer könnte zu diesem Zweck einen oder mehrere Schlüssel generiert und die falsche ID selbst unterzeichnet haben.

Ene bekannte und vertrauenswürdige Unterschrift liefert genügend Aufschluss darüber, dass eine BenutzerID vom Besitzer des Schlüssels stammt, vorausgesetzt, die Signatur besteht den **kc** Check.

Um BenutzerIDs zu bestätigen, sind Eigensignaturen aus zwei Gründen vorzuziehen:

1. es sind keine weiteren öffentlichen Schlüssel für ihre Prüfung erforderlich (**-kc**)
2. nur durch die Prüfung einer Eigensignatur kannst du sicher sein, dass eine spezielle BenutzerID zweifelsfrei vom Schlüsselinhaber kommt.

Die eigenen BenutzerIDs zu signieren ist daher ganz besonders bei Schlüsseln anzuraten, die zur Übermittlung an einen öffentlichen Schlüsselserver gedacht sind. Unsignierten IDs sollte man immer mit Argwohn begegnen .

6. Brauchen wir sicherere Schlüsselserver?

Öffentliche Schlüsselserver spielen eine herusragende Rolle bei der Verbreitung von Schlüsseln in der PGPGemeinschaft. Dazu wurden sie eingerichtet. Die ServerSoftware wurde entwickelt, um das Speichern und Verteilen von Schlüsseln zu ermöglichen.

Leider gibt es einige Individuen dort draussen, die es ausnutzen, dass Schlüsselserver die Schlüssel nicht vor Manipulationen schützen.

Im Moment verbessern die Programmierer der SchlüsselserverSoftware die Leistungsfähigkeit und Sicherheit der Server (7). Sie nähern sich der Vollendung neuer Versionen, mit Eigenschaften, die die Veränderung von Schlüsseln verhindern können. **Marc Horowitz** und **Jeffrey Schiller** (jis@mit.edu) am MIT re-implementierten vor kurzem den "aktuellen" Emailorientierten Schlüsselserver, wobei sie C anstelle von Perl und C anstelle von PGP für die Verwaltung der Schlüssel benutzten.

Der neue Server wurde für die schnellere Bearbeitung von Vorgängen optimiert. Derzeit nimmt er nicht mehr zusätzliche Überprüfungen vor als es der alte Server getan hätte.

Michael Graff (explorer@flame.org) arbeitet an einem Schlüsselserver der "zweiten Generation", basierend auf einem DNSähnlichen verzweigenden und delegierenden System. Der Server wird, unter anderem, sicherer sein.

Wichtige Eigenschaften des neuen Systems sind:

- Es wird auf viele Server verzweigt. Jeder Server kennt nur einen Teil der Schlüssel.
- Michael Graff plant beim Hochladen eines Schlüssels eine Nachricht an die Email Adresse im Schlüssel oder, in Ermangelung einer solchen, an den Absender des Schlüssels zu senden. Der muss dann zunächst eine "magische" Zeichenkette aus zufälligen Zeichen dekodieren und das Ergebnis zurückschicken, wodurch der Schlüssel in der Serverdatenbank erst aktiviert wird. Es ist erforderlich, dass der Versender im Besitz des fraglichen Schlüssels ist und weiss, wie man damit umgeht. Dadurch sollte auch den Versandt gefälschter Schlüssel vermieden werden.
- Nur der Inhaber eines Schlüssels, der Systemadministrator oder jemand, dem der Administrator die Aufgabe überträgt, kann Schlüssel in der Datenbank verändern.
- Ein Update bedeutet den kompletten Austausch , so dass der Inhaber eines Schlüssels vollends darüber entscheidet, wie er auszusehen hat.

Noch haben wir die neuen Schlüsselserver nicht. Einen Schlüssel auf einen der

Server zu laden, ihn zu aktualisieren oder zu unterschreiben, ist extrem unerwünscht, es sei denn, der Besitzer hat sein Einverständnis erklärt.

Es gibt eine kommerzielle Schlüsselregistratur bei "Four11 (SLED) Directory Services" (8), wo nur der Schlüsselinhaber, mit seinem privaten Schlüssel ausgerüstet, einen öffentlichen Schlüssel aktualisieren kann. Alte Schlüssel werden gelöscht.

Four11 zertifiziert allerdings keine Schlüssel mehr.

7. Belegt eine eigensignierte BenutzerID die Identität des Schlüsselinhabers?

Nein, eine Eigensignatur ist kein Knoten im " **Web of Trust** " (Netz des Vertrauens)!

Wenn du den Schlüsselbesitzer nicht persönlich kennst, kann nur die Signatur einer Person, die du kennst (und der du traust) bestätigen, dass ein Schlüssel der Person gehört, die behauptet, der Besitzer zu sein.

Wenn der ganze Schlüssel gefälscht wurde, könnte der Angreifer ihn mit dem korrespondierenden geheimen Schlüssel unterschrieben haben:

```
Typ Bits/ID      Datum      Benutzer
öff 1024/0987DCBA 1996/01/01 John Harmless <john@network.net>
Unt      0987DCBA             John Harmless <john@network.net>
```

Obwohl dieser Schlüssel eigensigniert ist, gehört er nicht zu John!

8. Gibt es auch Argumente gegen Eigensignaturen?

> *"Ich benutze niemals Schlüssel, wenn sie nicht die Signatur eines vertrauten Freundes tragen. Daher macht Eigensignieren für mich keinen Sinn. Eigensignaturen sind Platzverschwendung und schaffen bloss falsches Vertrauen."*

Wenn du ein SicherheitsFanatiker bist, bist du jedem Schlüssel gegenüber mistrauisch. Wenn ein Schlüssel aus deinem "Netz des Vertrauens" fällt, lehnst du ihn ab und verschlüsselst niemals Nachrichten an seinen Besitzer. Aber... sendest du statt dessen Klartext? Oder kommunizierst du grundsätzlich nicht mit Unbekannten?

Die meisten PGPNutzer rund um die Welt verschlüsseln gelegentlich Nachrichten an unbekannte Menschen. Meistens benutzen sie Schlüssel, die sie nicht überprüfen können. Eine Eigensignatur ist zumindest ein Beweis dafür, dass eine BenutzerID nicht manipuliert worden ist. Schlüsselserver wurden nicht als Email-

Verzeichnisse eingerichtet.

Wenn du Post an einen Unbekannten sendest und seine Adresse einer verlässlichen Quelle entnimmst, ist das Schlimmste, was geschehen kann, dass er deine Post nicht entziffern kann, weil du einen falschen Schlüssel benutzt hast. Natürlich informiert dich der Empfänger darüber und beim nächsten Mal benutzt du den richtige Schlüssel.

Email sollte immer verschlüsselt werden , auch Nachrichten an einen unbekannten Empfänger, dessen Schlüssel nicht überprüft werden kann. Ich sehe nicht, welchen Schaden das anrichten könnte. Auch Schlüssel unbekannter Herkunft sind nützlich, sogar wenn du nicht weisst, von wem sie stammen. Du weisst auch nicht, wer am anderen Ende die Email erhält. Solange es die gleiche Person ist (oder das auf ähnlichem Wege geprüft werden kann), ist der Schlüssel mindestens so sicher, wie es die Kommunikation ohne Verschlüsselung gewesen wäre, wahrscheinlich sogar weit mehr als das.

9. Eigensignieren eines neuen Schlüssels oder zusätzlicher BenutzerIDs

Wenn du neue Schlüssel mit deinem Namen generierst, und wenn du bereits ein verlässliches Paar besitzt, das von anderen Leuten unterschrieben wurde, kannst du deinen neuen Schlüssel mit deinem bewährten geheimen Schlüssel unterschreiben. Den PGPNutzern, die deinen alten Schlüssel besitzen, wird diese Signatur den Schlüssel genauso vertrauenswürdig machen, wie all die Unterschriften in deinem alten Schlüssel.

Dasselbe gilt für eine neue BenutzerID. Wenn die alte verlässliche Signaturen enthält, bietet deine Signatur unter der neuen ID die gleiche Vertrauenswürdigkeit. Es ist nicht erforderlich, deine Freunde erneut um Unterschriften zu bitten.

10. Wie signiere ich meine eigenen BenutzerIDs?

Um eine deiner eigenen BenutzerIDs zu unterzeichnen führe das folgende Kommando aus:

pgp ks deine_ID [u deine_Schlüssel_ID] [dein_Schlüsselring]

Hierdurch wird der BenutzerID eine Signatur hinzugefügt, die mit dem, zur SchlüsselID gehörenden geheimen Schlüssel erzeugt wird. Wenn " **deine_ID** " deine SchlüsselID ist (wie z. B.: 0xABCD1234), dann wird die Signatur der Standard BenutzerID des Schlüssels angehängt. Gibst du statt dessen als " **deine_ID** " eine deiner BenutzerIDs an, wird diese spezielle ID unterzeichnet.

Wenn du " **dein_Schlüsselring** " nicht spezifizierst, nimmt PGP an, dass dein Schlüsselring pubring.pgp heisst.

Beispiel: John Harmless signiert seine eigene BenutzerID in pubring.pgp

```
Typ  Bits    ID      Datum    Benutzer
öff  1024/ABCD1234 1996/01/01 John Harmless <john@company.com>
Unt      ABCD1234            John Harmless <john@company.com>
```

Die internationale Version (2.6.3i) von PGP "eigensigniert" automatisch alle neuen IDs, eine Eigenschaft, die MIT's PGP 2.6.2 nicht besitzt. Die entsprechende Option kann in 2.6.3i mittels der **AUTOSIGN=OFF/ON** Anweisung in der Datei config.txt kontrolliert werden (9).
Bei PGP 5/6 wird während des Keyerzeugungsvorganges der eigene Public Key automatisch unterzeichnet.

11. Wie kann ich eine BenutzerID überprüfen?

PGP erkennt keine gefälschten BenutzerIDs, es sei denn die ID wurde entweder vom Schlüsselinhaber oder von jemand anders unterzeichnet.

Nur eine signierte BenutzerID kann überprüft werden . Mit der **kc** Option verwendet PGP den öffentlichen Schlüssel des Unterzeichners und versucht, das gehashte Material zu entschlüsseln, das zusammen mit der Signatur gespeichert worden ist. Sollte PGP den Schlüssel nicht im bezeichneten Schlüsselbund finden, wird es eine Meldung ausgeben: " **Unknown signator, can't be checked** ", b.z.w. " **Unterzeichner unbekannt, keine Prüfung** ".

Um eine BenutzerID und ihre Signaturen zu überprüfen, führe das folgende Kommando aus:

pgp kc ["ID, die geprüft wird"] [Dein_Schlüsselbund]

Wenn " **ID, die geprüft wird** " nicht genannt wird, listet und überprüft PGP alle Signaturen in " **Dein_Schlüsselbund** ".

Sollte " **ID, die geprüft wird** " eine BenutzerID sein, werden nur Signaturen dieser speziellen ID geprüft.

Daher kannst du anstelle der BenutzerID auch eine SchlüsselID angeben, wie

PGP kc 0xABCD01234.

Die Prüfung der Signaturen bedeutet automatische auch einen Test der Benutzer-IDs, weil Informationen aus der Zeichenkette der BenutzerID in die HashFunktion aufgenommen wurden, als der Schlüssel signiert worden ist. Der Vergleich des entschlüsselten MD5Hash aus der Signatur mit dem gerade errechneten enthüllt die Fälschung. Wenn an einer BenutzerID herumgepfuscht worden ist, hörst du einen Piepton und siehst die folgende Warnung:

```
Type Bits/ID      Date      User

pub 1024/ABCD1234 1996/01/01 John Harmless <john@network.net>

sig*     ABCD1234            John Harmless <john@network.net>
```

**** BAD SIGNATURE! ****

Remove bad signatures (Y/n)?

b.z.w. wenn deutsche Meldungen benutzt werden:

Typ Bits/ID Datum Benutzer
öff 1024/ABCD1234 1996/01/01 John Harmless <john@network.net>
Unt* ABCD1234 John Harmless <john@network.net>
**** FALSCHE UNTERSCHRIFT ****
Falsche Unterschriften löschen (J/n)?

Auf Verlangen entfernt PGP falsche Unterschriften, aber es kann nicht zwischen schlechten Unterschriften und schlechten BenutzerIDs unterscheiden. Also nimm die Meldung ernst.

Du wirst die "Falsche Unterschrift" von jemandem, den du nicht kennst, ohnehin entfernen wollen; aber das löst das Problem mit der defekten BenutzerID nicht! Wenn ein **kc** Test eine schlechte BenutzerID enthüllt, müssen nicht alle ihre Signaturen notwendigerweise ungültig sein. Jemand (wahrscheinlich der Angreifer selbst) könnte eine BenutzerID selbst unterschrieben haben, nachdem sie manipuliert worden ist. Das würde eine gefälschte Unterschrift unter einer gefälschten BenutzerID generieren.

PGP kvv bietet keine Überprüfung . Unterschriften werden lediglich angezeigt. Das gleiche gilt für das Verschlüsseln von Nachrichten.

Wenn du eine gefälschte BenutzerID für die Verschlüsselung mittels öffentlichem Schlüssel nennst, wird PGP dich nicht warnen. Wenn ein Schlüssel deinem öffentlichen Schlüssel zugefügt wird (**PGP ka**), führt PGP automatisch einen SignaturTest durch. Sollte es dich über eine falsche Signatur in Kenntnis setzen, dann entferne diesen Schlüssel sofort aus deinem Schlüsselbund. Du solltest ausserdem die EmailAdresse des Besitzers überprüfen und ihm mitteilen, was geschehen ist.

12. Zusammenfassung

- Im Moment spielen öffentliche Schlüsselserer eine wichtige Rolle beim Verteilen von Schlüsseln in der PGPGemeinschaft. Leider schützen sie deine Schlüssel nicht vor Angreifern und NetzKomödianten. Die Situation wird sich ändern, wenn die Server zweiter Generation den Dienst aufnehmen.
- Eigensignieren (wie jedes Unterzeichnen) von SchlüsselIDs ist eine Massnahme, um "Denial Of Service" Angriffe zu erschweren. Es wird im Besonderen bei Schlüsseln empfohlen, die zum Hochladen auf öffentliche Schlüsselserver gedacht sind. Dennoch kann eine Signatur den Angriff nicht verhindern; lediglich wird PGP ihn dadurch bemerken.
- Eine Eigensignatur ist kein Knoten im "Web Of Trust", Sie bestätigt nicht die

Identität des Schlüsselinhabers. Sie kann aber verwendet werden, um die UserIDs des Schlüssels zu überprüfen und wenigstens eine Form des "Denial Of Service" Angriffes aufzudecken.

- Für jemanden, der weder den Eigentümer eines Schlüssels, noch die Leute kennt, die den Schlüssel unterzeichnet haben, ist die Eigensignatur sogar besser, als die Signatur irgendeiner unbekannten Person.
- Eine Eigensignatur unter einer zusätzlichen BenutzerID dient als "Einführer" einer neuen ID, wenn die alte vertrauenswürdige Signaturen trägt.

13. Referenz

(1) **PGP Dokumentation** von Phil Zimmermann
http://www.ifi.uio.no/pgp/doc.shtml

In Deutsch:
http://www.zerberus.de/~christopher/pgp/inhalt.html

(2) alt security.pgp: **Häufig gestellte Fragen**
von Jeff Licquia, 25. Mai 1995
http://www.prairienet.org/~jalicqui/pgpfaq.txt

eine neuere Fassung in Deutsch:
http://www.franken.de/users/sulla/pgp/csp/csp.htm

(3) **Why Should You Sign Your Own PGP Public Key?**
von Francis Litterio
http://world.std.com/~franl/pgp/why-sign-your-key.html

(4) EFH PGP Workshop
von Paul Elliott, Electronic Frontiers Houston
http://www.efh.org/pgp/pgpwork.html

(5) **The Beginner's Guide to Pretty Good Privacy**
von Bill Morton, Version 1.1, April 13, 1995
http://netaccess.on.ca/~rbarclay/bg2pgp.txt

(6) " **Dead beef" attack** against PGP's Key management
Posting in alt.security.pgp vom 1. April 1996
Raph Levien (raph@c2.org)

(7) Know future developments in **PGP servers** and related subjects
http://ww.pgp.net/pgpnet

(8) **Four11** Directory Services
http://www.four11.com/

(9) **Differences in International PGP Versions**
von Ståle Schumacher
http://www.ifi.uio.no/pgp/diffs.shtml