Daten- und Emailverschlüsselung

Eine Einführung in GnuPG

10.November 2006

Florian Streibelt <florian@freitagsrunde.org>

Ziele der Verschlüsselung

- Geheimhaltung

  Nachricht soll nicht durch Dritte entziffert werden können

- Echtheit des Kommunikationspartners

  Stammt die Nachricht wirklich von dem, der er vorgibt zu sein?

- Integrität der Nachricht

  Ist die Nachricht so verfasst worden, wie sie angekommen ist?

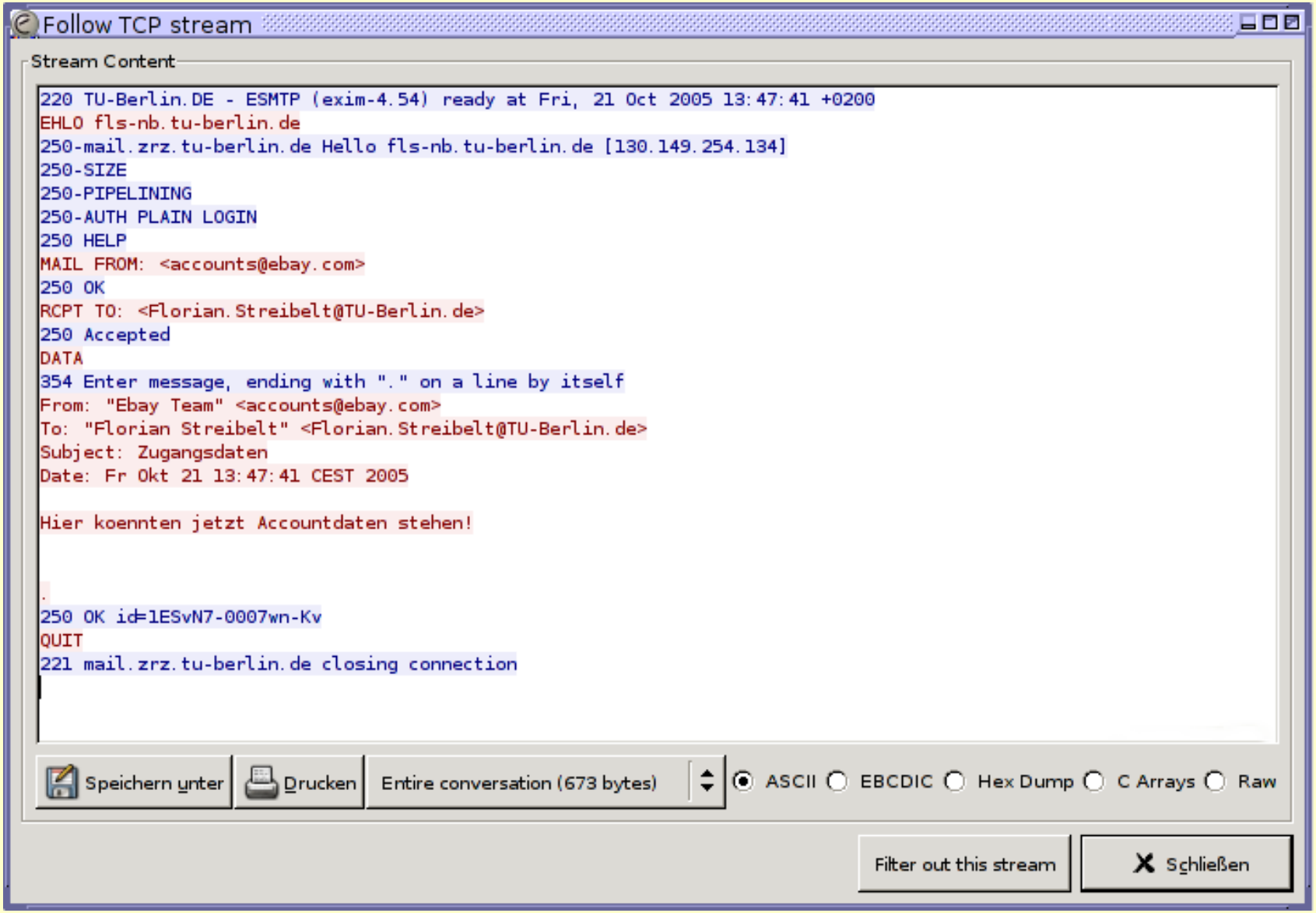
► Vertrauen in die Kommunikation

→ Ist erreicht, falls obige Punkte eingehalten werden.

Bestandsaufnahme

- Email-Verkehr kann sehr einfach abgehört und manipuliert werden (ECHELON, TKÜV)
- ein sehr hoher Prozentsatz der Firmen-kommunikation erfolgt dennoch per Email!
- interne Daten werden damit per „elektronischer Postkarte“ versandt
- im Rahmen der „Terrorismusbekämpfung“ wird die Vorratsdatenspeicherung eingeführt
- Emails werden zu Ermittlungszwecken gesichert / ausgelesen

Emails „abhören“



Die Lösung?

Die Lösung ist:

Kryptographie

= Wissenschaft von der Verschlüsselung und Verschleierung

Im Gegensatz:

Kryptoanalyse

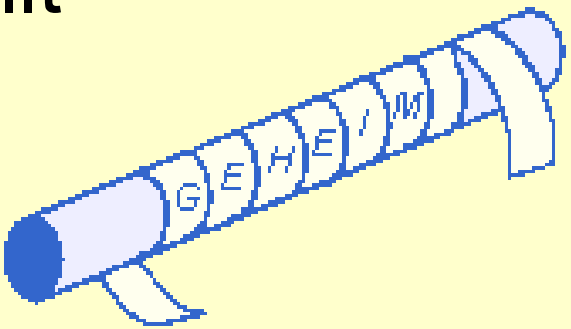
= Wissenschaft von der Entschlüsselung

Wie ist Kryptographie entstanden?

- die Ursprünge reichen sehr weit zurück
- zunächst wurden die Daten „nur“ versteckt
  → ein geheimer Bote überbringt die Nachricht
- später wurden die Daten verschlüsselt
  → Auch der Bote kann die Nachricht nun nicht mehr lesen oder manipulieren
  → Bei Gefangennahme eines Boten bleibt die Nachricht geheim

Skytala von Sparta

- älteste bekannte Verschlüsselungmethode (ca. 2500 Jahre)
- von Plutarch überliefert
- ein Band wird um einen Zylinder gelegt, quer beschrieben und wieder abgerollt
- der Schlüssel besteht aus dem Zylinderdurchmesser



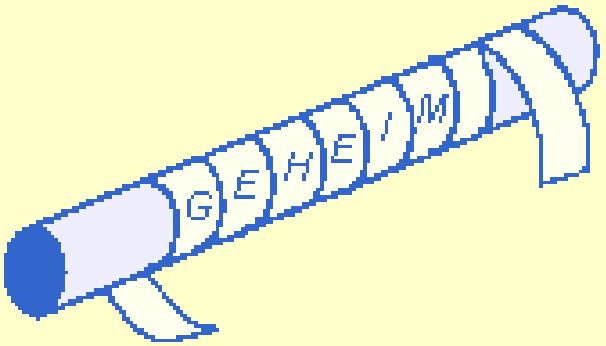
<u>Transpositionschiffre:</u>

Die Anordnung der Zeichen wird vertauscht

Beispiel Skytala

- Geheimtext:

  FTRETRAUDUEGNEBISDRX



- Test mit „Durchmesser 5cm“:

  FTDNSTRUEDRAEBREUGIX

- Test mit „Durchmesser 4cm“:

  FREITAGSRUNDEDERTUBX

Cäsarchiffre

- von Julius Cäsar (100-44 v.Chr.) erfunden
- Buchstaben werden um einige Stellen verschoben
- Klasse der Verschiebechiffren
- ROT13 als „moderne“ Variante
- Problem:
  Buchstabenhäufigkeiten bleiben erhalten

Beispiel Cäsarchiffre

- Klartextalphabet:

  a b c d e f g h i j k l m n o p q r s t u v w x y z
- Geheimtextalphabet:

  d e f g h i j k l m n o p q r s t u v w x y z a b c
- Klartext:

  FREITAGSRUNDEDERTUB
- Geheimtext:

  IUHLWDJVUXQGHGHUWXE

One-Time-Pad

- einzige beweisbar sichere Verschlüsselung
- Voraussetzungen:
  - Schlüsselänge gleich wie Klartextlänge
  - Schlüsselerzeugung ***streng*** zufällig
  - Schlüssel wird nur ***einmal*** verwendet
- hoher Aufwand → militärische Umgebungen
- Probleme:
  - menschlicher Faktor
  - Beschaffung neuer OneTimePads...

Symmetrische vs. Asymmetrische Verschlüsselung

- Nachteile aller bisherigen Methoden:
  - Zur Kommunikation über unsichere Kanäle (Internet) muss der Schlüssel zuerst über einen sicheren Kanal übertragen werden.
  - ein Schlüssel pro Kommunikationspartner
- Ausweg: asymmetrische Verschlüsselung
  - Es gibt einen geheimen und einen öffentlichen Schlüssel.
  - Der öffentliche Schlüssel dient zum Verschlüsseln von Daten, nur der Geheime kann die Originaldaten wieder herstellen. (idealerweise)

RSA

- Das Prinzip wurde von Whitfield Diffie und Martin Hellman 1976 entwickelt und publiziert
- Ron **R**ivest, Adi **S**hamir und Leonard **A**dleman wollten dessen Sicherheit widerlegen
- ... und erfanden dabei RSA
- ältester und angesehenster asymmetrischer Algorithmus, löste als erster das Schlüssel-verteilungsproblem
- basiert auf der Faktorisierung großer Zahlen

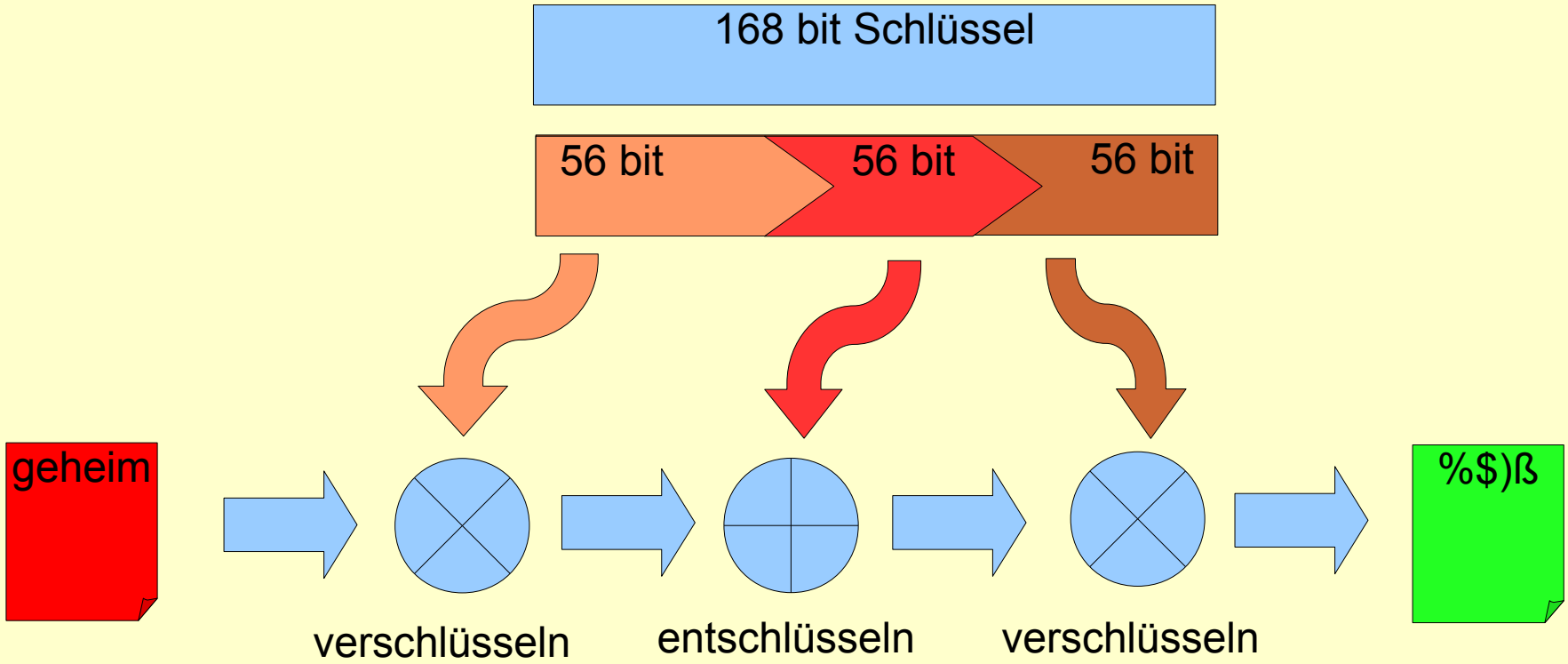
Prinzip von RSA

- Wahl zweier grosser Primzahlen *a* und *b*
- Ermittlung des Produkts $n=a \cdot b$ und Ermittlung der Anzahl der zu *n* teilerfremden Zahlen $\varphi(n)=(a-1) \cdot (b-1)$
- Suche eine Zahl *d*, für die gilt: $e \cdot d \bmod \varphi(n)=1$
- *e* und *n* sind der öffentliche Schlüssel
- *d* und *n* sind der private Schlüssel
- mehr: http://de.wikipedia.org/wiki/RSA-Kryptosystem

DES und 3DES

- **D**ata **E**ncryption **S**tandard von 1977
- symmetrische Blockchiffre mit 56 Bit
- im kommerziellen Bereich am häufigsten eingesetzt
- u.a. 1999 in ca. 22 Stunden gebrochen
- Weiterentwicklung: 3DES, benutzt 3 unterschiedliche DES Schlüssel hintereinander: 168 Bit
- wird bei einigen Smartcards verwendet.

3DES-Verfahren



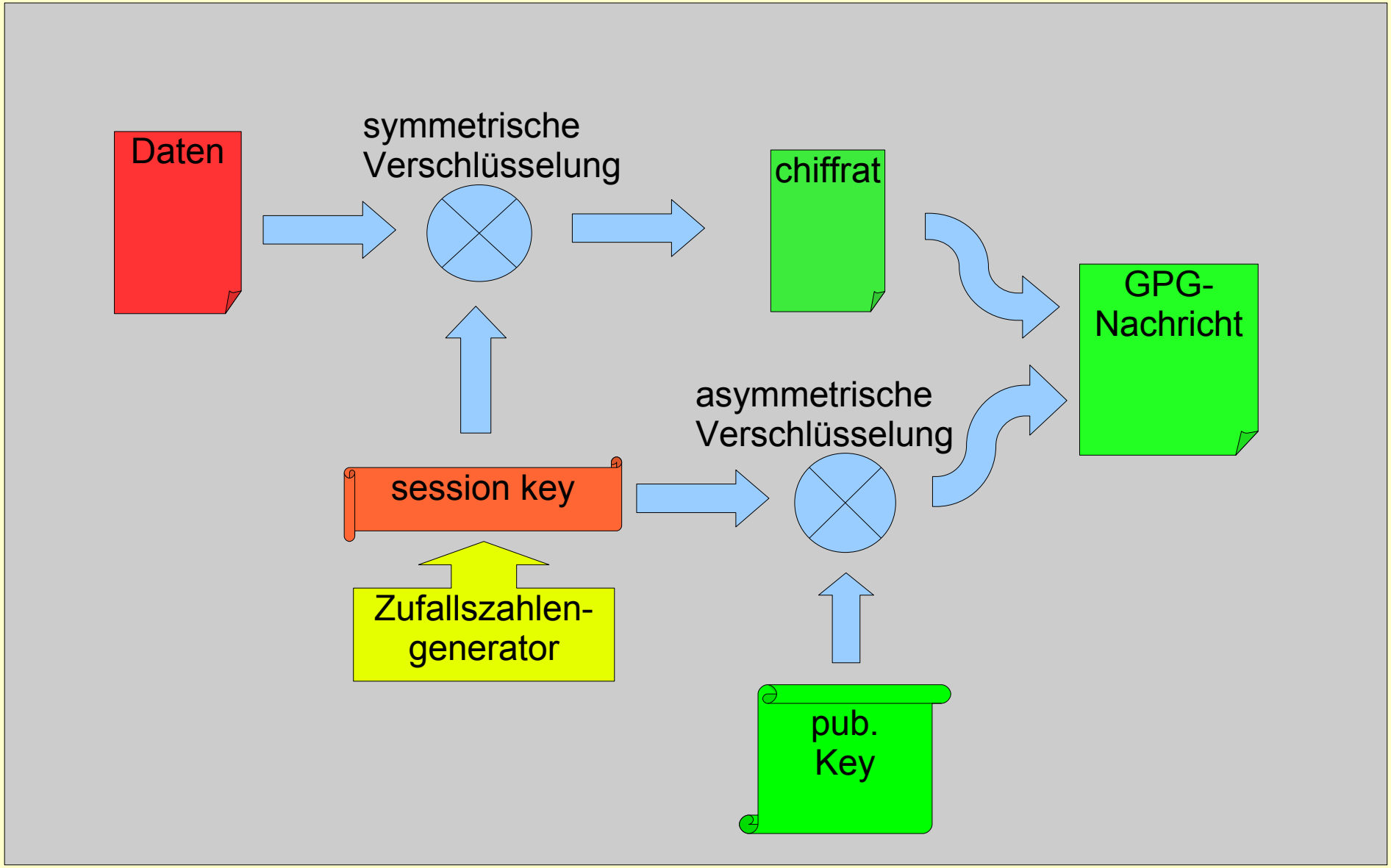
Symmetrische vs. Asymetrische Verschlüsselung II

- Nachteile symmetrischer Verschlüsselung:
  - shared secret
  - Schlüsselverteilungsproblem
- Vorteil:
  - sehr schnell
- Nachteil asymmetrischer Verschlüsselung:
  - bei großen Datenmengen sehr langsam
- Vorteil:
  - kein Schlüsselverteilungsproblem

Funktionsweise von GnuPG

- hybrides Verfahren, kombiniert geschickt symmetrische und asymmetrische Algorithmen:
  - ein zufälliger symmetrischer Schlüssel wird zur Verschlüsselung benutzt (session key)
  - dieser wird asymmetrisch verschlüsselt und mit der Nachricht verschickt
  - geheimer Schlüssel entschlüsselt den session key, damit kann der Empfänger die Daten lesen
- man benötigt nur den öffentlichen Schlüssel des Kommunikationspartners
- der geheime Schlüssel verlässt nie den Rechner

Schema GnuPG



Geschichte (auch) von GnuPG

- PGP (Pretty Good Privacy) 1991 von Phil Zimmermann, zunächst Open Source, später (1997) Verkauf an NAI
- div. Inkompatibilitäten der PGP-Versionen führten 1998 zu RFC 2440
- seit 1998 Entwicklung von GnuPG (Gnu Privacy Guard), Förderung durch BMBF
- aktuelle Version von GnuPG: 1.4.5
- seit einiger Zeit: Chipkartenunterstützung
- Open Source, diverse Portierungen

Praktische Einführung in GnuPG

Auf den folgenden Folien wird gezeigt:

- erzeugen eines eigenen Schlüssels
- erzeugen eines Widerrufszertifikates
- Verschlüsseln einer Datei
- Entschlüsseln einer Datei

Alle Beispiele beziehen sich auf die Kommandozeile, da hier die größte Kontrolle über GnuPG ausgeübt werden kann, am Ende werden noch einige GUI kurz vorgestellt.

Schlüssel erzeugen

Wahl des Signatur- und/oder Verschlüsselungsalgorithmus

```
fls@fls-nb:~$ gpg --gen-key
Bitte wählen Sie, welche Art von Schlüssel Sie
möchten:
   (1) DSA and Elgamal (default)
   (2) DSA (nur signieren/beglaubigen)
   (5) RSA (nur signieren/beglaubigen)
```

DSA: Digital Signature Algorithm,
zum Signieren von Dateien (nicht DES!)

ElGamal: Verschlüsselungsalgorithmus von El Gamal (Israel), ähnlich RSA

RSA: Rivest, Shamir, Adleman,
wie bereits vorgestellt

Schlüssel erzeugen

Wahl der Schlüssellänge

```
...
DSA keypair will have 1024 bits.
ELG-E keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
Requested keysize is 2048 bits
```

- Es gibt Anzeichen, dass 1024 Bit nicht mehr ausreichend sicher sind. Deshalb sollte eine Schlüssellänge von mindestens 2048 bit gewählt werden.
- Das Arbeiten mit größeren Schlüsseln wird schnell sehr langsam...
- Hier ist wie immer zwischen Sicherheit und Komfort abzuwägen.

Schlüssel erzeugen

Wahl der Gültigkeitsdauer

```
...
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)
```

- Gültigkeitsdauer hängt von eigenen Präferenzen ab.
- Oftmals wird ein Hauptschlüssel ohne feste Gültigkeit generiert und dann jeweils Unterschlüssel mit einjähriger Laufzeit.
- Vorteil: Bei Verlust der Passphrase und des Revocation Zertifikates verfällt der Schlüssel automatisch
- Partner werden gezwungen, „öfter“ auf Keyservern neu zu suchen

Schlüssel erzeugen

Name, Kommentar, E-Mail-Adresse und Passphrase

```
...
You need a user ID to identify your key; the software constructs the user ID
from the Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name: Florian Streibelt
Email address: florian@freitagsrunde.org
Comment: Freitagsrunde der TUB
You selected this USER-ID:
    "Florian Streibelt (Freitagsrunde der TUB) <florian@freitagsrunde.org>"
```

- Eingabe von Namen und E-Mail-Adresse und Bestätigung der Angaben
- Eingabe der Passphrase
- [Passphrase] kein „Wort“ aus dem Wörterbuch, möglichst ein verfremdeter Satz, z.B. „Ich habe den Linuxtag Chemnitz besucht.“ → „1(h 4abe d3n L!nv)(tag Ch0mn:tz b?such+.“

Schlüssel

```
...
gpg: key F435050F marked as ultimately trusted
public and secret key created and signed.

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
pub   1024D/F435050F 2005-11-06
      Key fingerprint = B625 3451 AF4E F755 E407  FD88 A92B 359F F435 050F
uid                  Florian Streibelt (Freitagsrunde der TUB)
<florian@freitagsrunde.org>
sub   2048g/6BB5B4FA 2005-11-06
```

- Anzeige des Fingerprints am Ende der Schlüsselerzeugung
- Der Fingerprint ist eine Prüfsumme, mit der die Integrität des Schlüssels geprüft werden kann.
- Die letzten 8 Byte sind die Key-ID, wegen Kollisionen werden mittlerweile meist die letzten 16 Byte benutzt.

Widerrufszertifikat erzeugen

```
fls@fls-nb:~$ gpg --gen-revoke F435050F

sec  1024D/F435050F 2005-11-06 Florian Streibelt (Freitagsrunde der TUB)
<florian@freitagsrunde.org>

Create a revocation certificate for this key? (y/N)

Please select the reason for the revocation:
  0 = No reason specified
  1 = Key has been compromised
  2 = Key is superseded
  3 = Key is no longer used
  Q = Cancel
(Probably you want to select 1 here)
Your decision?
```

- Vorbeugung vor kompromittierten Schlüsseln und vergessenen Passwörtern
- Widerrufszertifikat ausdrucken und an sicherem Ort aufbewahren, möglichst nicht mit Arial: ll = iL, besser Times, Comic, etc.

Widerrufszertifikat

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.1 (GNU/Linux)
Comment: A revocation certificate should follow

iFwEIBECABwFAkNuZYQVHQBQYXNzcGhyYXNlIHZlcmxvcmVuAAoJEKkrNZ/0NQUP
mUsAniYuZuEp+Vrvhgrj/8KTTcBE0x4bAJ0YYyczU7zQlcJd7YsX00qGt12NSA==
=F8X3
-----END PGP PUBLIC KEY BLOCK-----
```

- Vorsicht beim Ausdruck: Das Drucksystem könnte die Datei speichern, eventuell sogar der (Netzwerk-) Drucker selbst
- Jeder mit Zugriff auf diese „Zeichenkette“ kann den Schlüssel unbrauchbar machen, indem er das Revocation-Zertifikat auf einem Keyserver veröffentlicht.

Beispiel: Verschlüsseln

```
$ echo "Geheime Mitteilung" > geheim.txt
$ gpg -aer F435050F geheim.txt
$ ls -l geheim*
-rw-r--r--  1 fls users  19 Nov  6 21:25 geheim.txt
-rw-r--r--  1 fls users 932 Nov  6 21:25 geheim.txt.asc

$ cat geheim.txt.asc
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1.4.1 (GNU/Linux)

hQIOA4C2E61rtbT6EAf+PXrzs5CwHhRXsb3UK+5wgnV4eLYUDv0dVag5akMj+nvU
ekEBb0ucjZN20HKB9wgKAXo3+HmkiG4IYElQm5d2wp6He6wRUo2DoxZR1Gmfczoy
sQFC/fGWaq/wclgKiyEyYCECao0R6Y8PrNS0Zyybq+5XJOH4n8ybDJFTo6FqIYfb
uTZTCxtRqpB7T5MUMt0hUJTX3LNF46QtlOxNXhmeYM9pd5AJMmXkPbwSw1CsHvdG
gbGajMmoewYswSD5WMeB66dxtIQQxZfQNRMDn/9i6Kpfj0rh/WBf/3NrPsoy04y6
l7lpR/eeeGyBooiY2ti0/rNimXipW3zh3bgejnpzAAgAmFwOyE5aJlelBVQ5EFxW
et1VutRo7hVjAimWNPxDK8pJUDxJ2P0h9Kbe/Tq6tV+yp5viwnI3Ec5F3YqWpVaC
ebdlgH7mxAvsz4266hGFZsCImvl9UqBeKd7flZxhYIpG/+omgTDWyve8gA5kUZg3
O/v7ic2xHeDsHoxOYvol9cGteeIgj+eBBCsZT79Nhy5ZoCvroUBaueLh3GNDInrs
nFvjGKAlg8BMPRofdD6BkGix+pp5pLKdOrVfVMgqdliEHpoh0KXzO9l4q5Bux+EC
bMltOcTX0Ir1rxAVqqE0qqKCLLiHZ8TJ1fh6polmOTryQAwimP3b8uNiB9NeT7sW
H9JVAVCnf3FUMmac1TR4/JmfPNjI41k4yQ84N5fJmcgbbmjgvLk75RhpEnLqaxrG
PnDx6jhqQ+zUSznwrg67KfqIqCLUzPKnTTIxc6d1rzoWVhKAEiTEBw==
=xXPN
-----END PGP MESSAGE-----
```

Beispiel: Entschlüsseln

```
$ cat geheim.txt.asc | gpg

You need a passphrase to unlock the secret key for
user: "Florian Streibelt (Freitagsrunde der TUB) <florian@freitagsrunde.org>"
2048-bit ELG-E key, ID 6BB5B4FA, created 2005-11-06 (main key ID F435050F)

gpg: encrypted with 2048-bit ELG-E key, ID 6BB5B4FA, created 2005-11-06
      "Florian Streibelt (Freitagsrunde der TUB) <florian@freitagsrunde.org>"
Geheime Mitteilung
```

alternativ, direkt in eine Datei:

```
fls@fls-nb:/tmp$ gpg geheim.txt.asc

You need a passphrase to unlock the secret key for
user: "Florian Streibelt (Freitagsrunde der TUB) <florian@freitagsrunde.org>"
2048-bit ELG-E key, ID 6BB5B4FA, created 2005-11-06 (main key ID F435050F)

gpg: encrypted with 2048-bit ELG-E key, ID 6BB5B4FA, created 2005-11-06
      "Florian Streibelt (Freitagsrunde der TUB) <florian@freitagsrunde.org>"
```

Features in GnuPG

- Chipkartenunterstützung: OpenGPG-Card
- Foto-IDs:
  - eingebettete jpeg-Dateien
  - Vorsicht: bitte kleine Dateien benutzen
- mehrere uid's Pro Schlüssel
  - pro Emailadresse eine User-ID
- mehrere Schlüssel
  - werden alle zusammen signiert
  - z.B. für unsichere Umgebungen

Aufbau des Web of Trust

- Das Problem:
  Jeder kann Schlüssel für beliebige Namen und Emailadressen erzeugen.
- Viele falsche Schlüssel existieren.
- Eine Suche nach Bill Gates ergibt bereits einige hundert Treffer
- Die Lösung: **Keysigning:**

  Inhaber von PGP-Schlüsseln treffen sich und verifizieren ihre Identitäten.

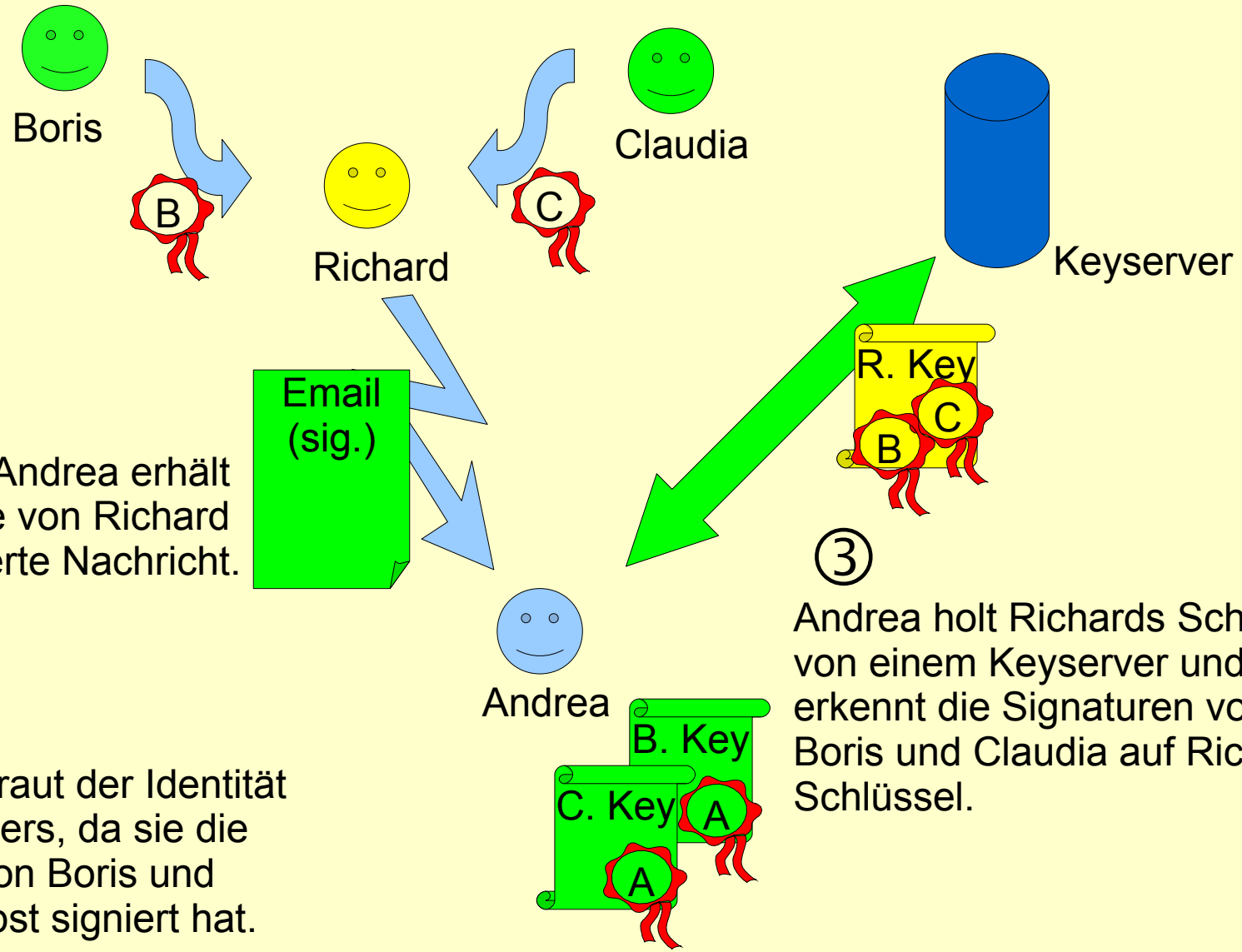
Keysigning

- Beim Keysigning überprüft man die user-id (den Namen) anhand eines Ausweises
- Danach signiert man den Schlüssel des anderen lokal auf dem eigenen Rechner
- Idealerweise signiert man jede BenutzerID auf dem Schlüssel einzeln
- Nun schickt man den signierten Schlüssel verschlüsselt an die angegebenen Emailadresse(n)
- Damit sind nun auch die Emailadressen verifiziert!
- Es gibt scripte dafür: caff, oder mycaff (Eigenwerbung):
  http://user.cs.tu-berlin.de/~mutax/ksp-scripts/mycaff/
- das Web of Trust lebt, wie der Name sagt, vom Vertrauen:
  bitte nicht blind signieren!

Web of Trust

- Hat man ein genügend großes Web of Trust, muss nicht mehr jeder jeden Schlüssel signieren!
- Jedem öffentlichen Schlüssel anderer Personen kann man verschiedene Vertrauensstufen zuweisen
- Bekommt man eine Nachricht von einem bisher unbekannten Schlüssel kann man prüfen, ob eine gewisse Anzahl vertrauenswürdiger Bekannter diesen unterschrieben haben
- ist dies der Fall, kann man davon ausgehen, dass die Identität genügend geprüft und sicher ist.
- wichtiges Hilfsmittel: Keyserver, die öffentliche Schlüssel und Wiederrufszertifikate vorhalten

Web of Trust (Schema)



① Claudia und Boris haben Richards Schlüssel signiert.

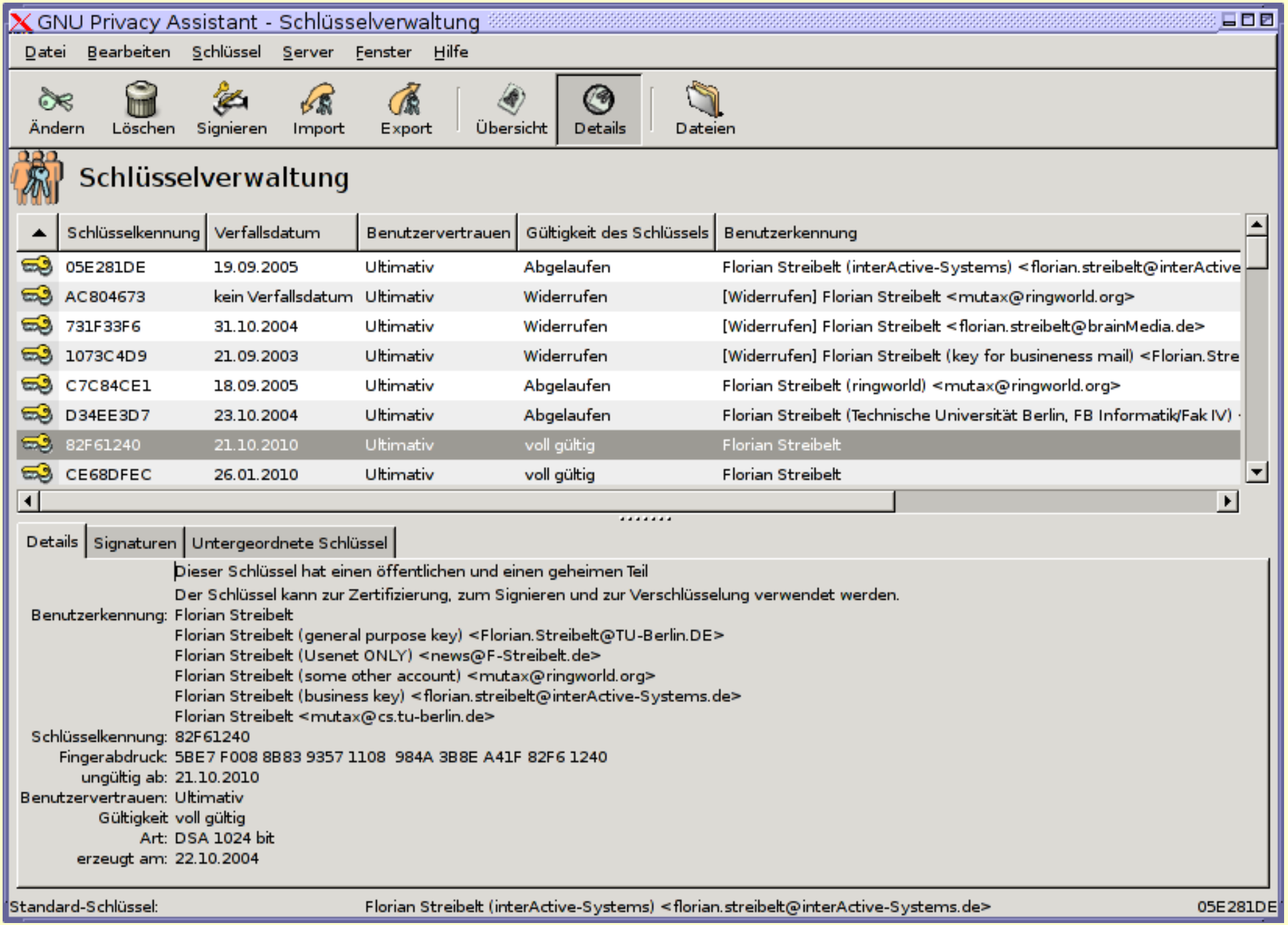
② Andrea erhält eine von Richard signierte Nachricht.

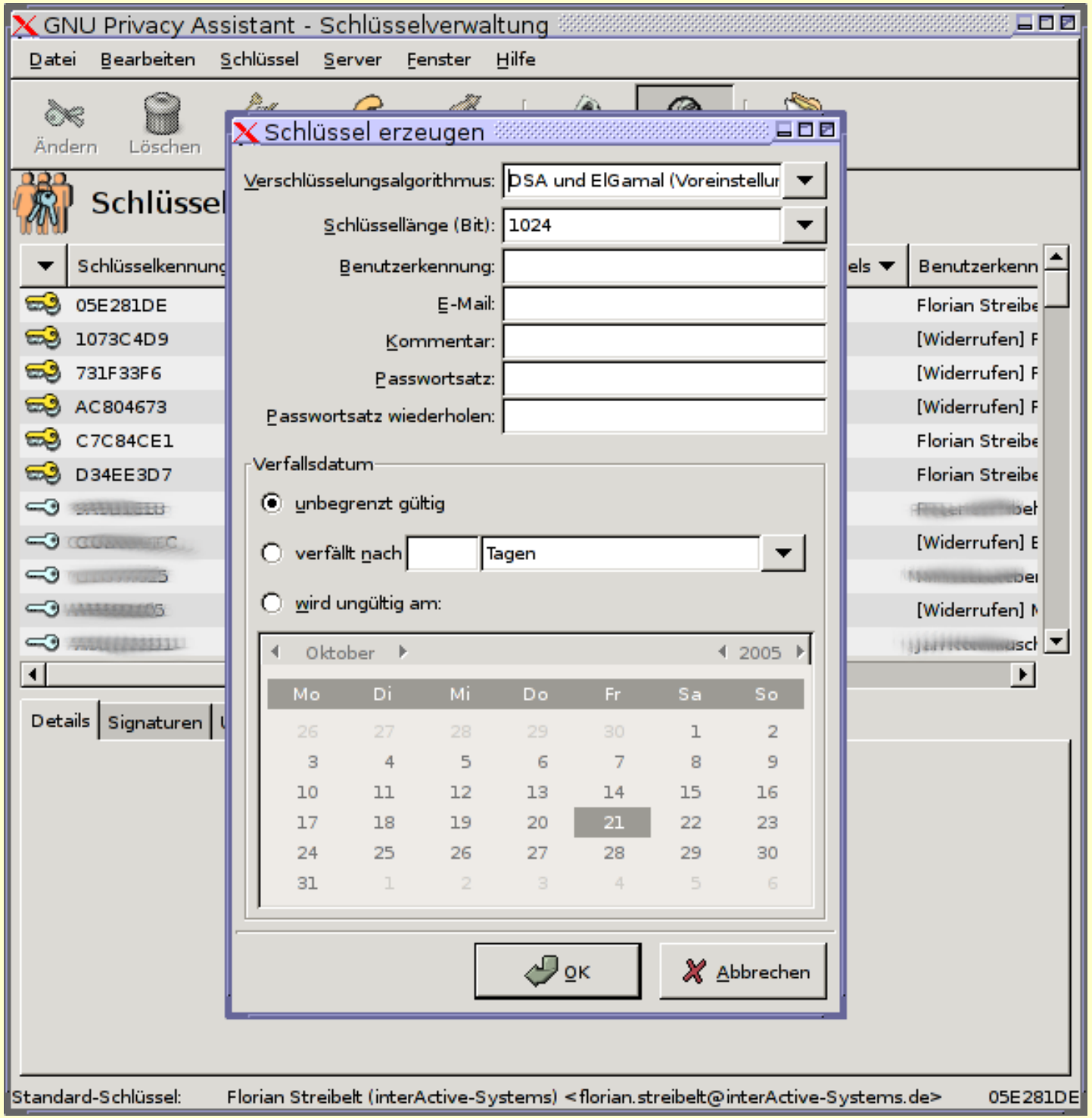
③ Andrea holt Richards Schlüssel von einem Keyserver und erkennt die Signaturen von Boris und Claudia auf Richards Schlüssel.

④ Andrea vertraut der Identität des Absenders, da sie die Schlüssel von Boris und Claudia selbst signiert hat.

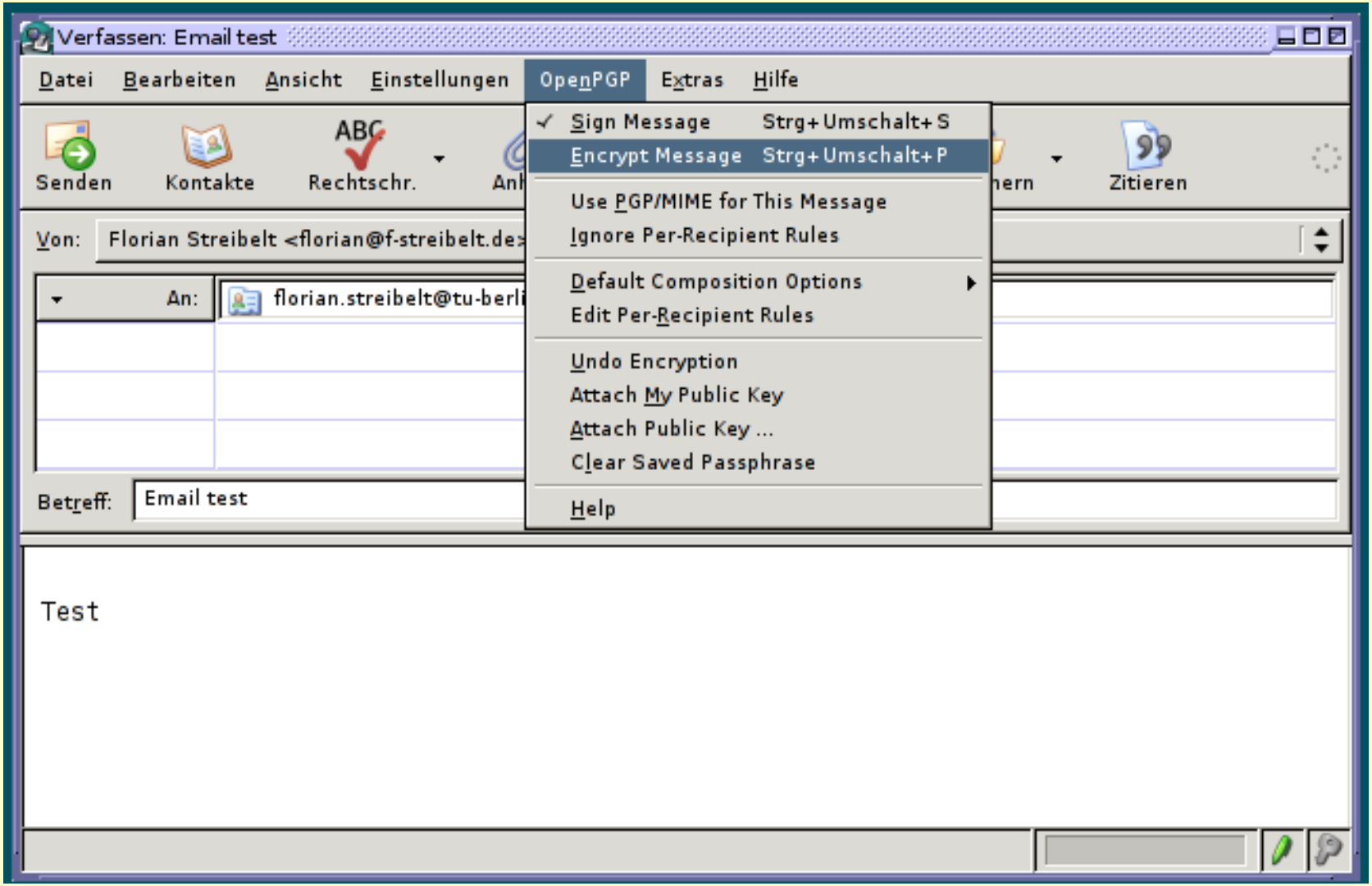
Werkzeuge

- auf der Kommandozeile hat man die größte Kontrolle
- es gibt einige grafische Tools für Linux und Windows™
- unter Linux: gpa - den Gnu Privacy Assistant
- Plugins für die meisten Emailprogramme sind verfügbar (Thunderbird, Evolution)
- GnuPG kann beliebige Daten verschlüsseln - wie gezeigt auch lokale Dateien

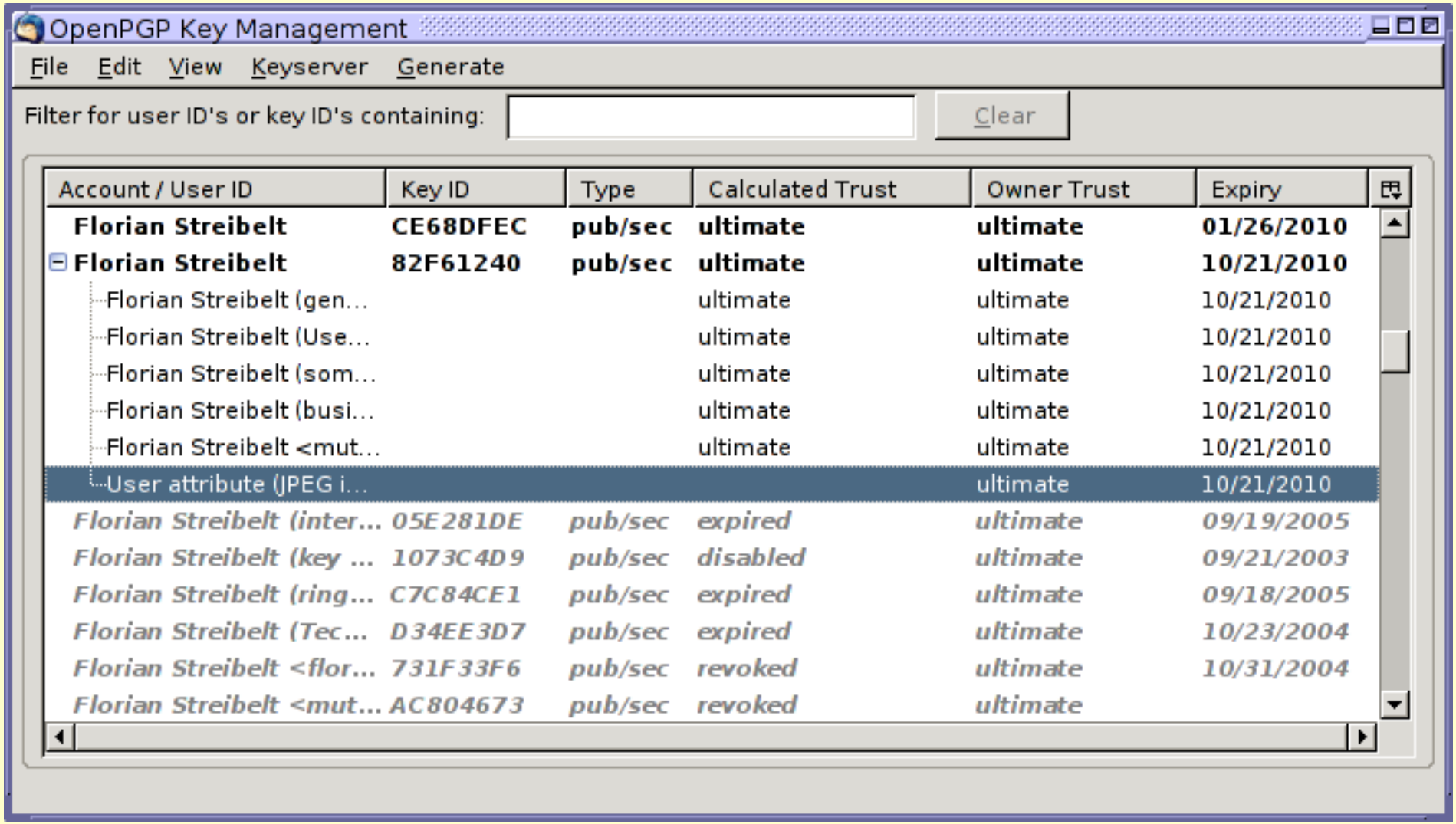

GNU Privacy Assistant - Schlüsselverwaltung
Datei Bearbeiten Schlüssel Server Fenster Hilfe
Ändern Löschen Signieren Import Export Übersicht Details Dateien
Schlüsselverwaltung
Schlüsselkennung Verfallsdatum Benutzervertrauen Gültigkeit des Schlüssels Benutzerkennung
05E281DE 19.09.2005 Ultimativ Abgelaufen Florian Streibelt (interActive-Systems) <florian.streibelt@interActive
AC804673 kein Verfallsdatum Ultimativ Widerrufen [Widerrufen] Florian Streibelt <mutax@ringworld.org>
731F33F6 31.10.2004 Ultimativ Widerrufen [Widerrufen] Florian Streibelt <florian.streibelt@brainMedia.de>
1073C4D9 21.09.2003 Ultimativ Widerrufen [Widerrufen] Florian Streibelt (key for busineness mail) <Florian.Stre
C7C84CE1 18.09.2005 Ultimativ Abgelaufen Florian Streibelt (ringworld) <mutax@ringworld.org>
D34EE3D7 23.10.2004 Ultimativ Abgelaufen Florian Streibelt (Technische Universität Berlin, FB Informatik/Fak IV)
82F61240 21.10.2010 Ultimativ voll gültig Florian Streibelt
CE68DFEC 26.01.2010 Ultimativ voll gültig Florian Streibelt
Details Signaturen Untergeordnete Schlüssel
Dieser Schlüssel hat einen öffentlichen und einen geheimen Teil
Der Schlüssel kann zur Zertifizierung, zum Signieren und zur Verschlüsselung verwendet werden.
Benutzerkennung: Florian Streibelt
Florian Streibelt (general purpose key) <Florian.Streibelt@TU-Berlin.DE>
Florian Streibelt (Usenet ONLY) <news@F-Streibelt.de>
Florian Streibelt (some other account) <mutax@ringworld.org>
Florian Streibelt (business key) <florian.streibelt@interActive-Systems.de>
Florian Streibelt <mutax@cs.tu-berlin.de>
Schlüsselkennung: 82F61240
Fingerabdruck: 5BE7 F008 8B83 9357 1108 984A 3B8E A41F 82F6 1240
ungültig ab: 21.10.2010
Benutzervertrauen: Ultimativ
Gültigkeit voll gültig
Art: DSA 1024 bit
erzeugt am: 22.10.2004
Standard-Schlüssel: Florian Streibelt (interActive-Systems) <florian.streibelt@interActive-Systems.de> 05E281DE


GNU Privacy Assistant - Schlüsselverwaltung
Datei Bearbeiten Schlüssel Server Fenster Hilfe
Ändern
Löschen
Schlüssel
Schlüsselkennung
Benutzerkenn
05E281DE
1073C4D9
731F33F6
AC804673
C7C84CE1
D34EE3D7
Florian Streibe
[Widerrufen] F
Details
Signaturen
Schlüssel erzeugen
Verschlüsselungsalgorithmus:
DSA und ElGamal (Voreinstellur
Schlüssellänge (Bit):
1024
Benutzerkennung:
E-Mail:
Kommentar:
Passwortsatz:
Passwortsatz wiederholen:
Verfallsdatum
unbegrenzt gültig
verfällt nach
Tagen
wird ungültig am:
Oktober
2005
Mo Di Mi Do Fr Sa So
OK
Abbrechen
Standard-Schlüssel:
Florian Streibelt (interActive-Systems) <florian.streibelt@interActive-Systems.de>
05E281DE

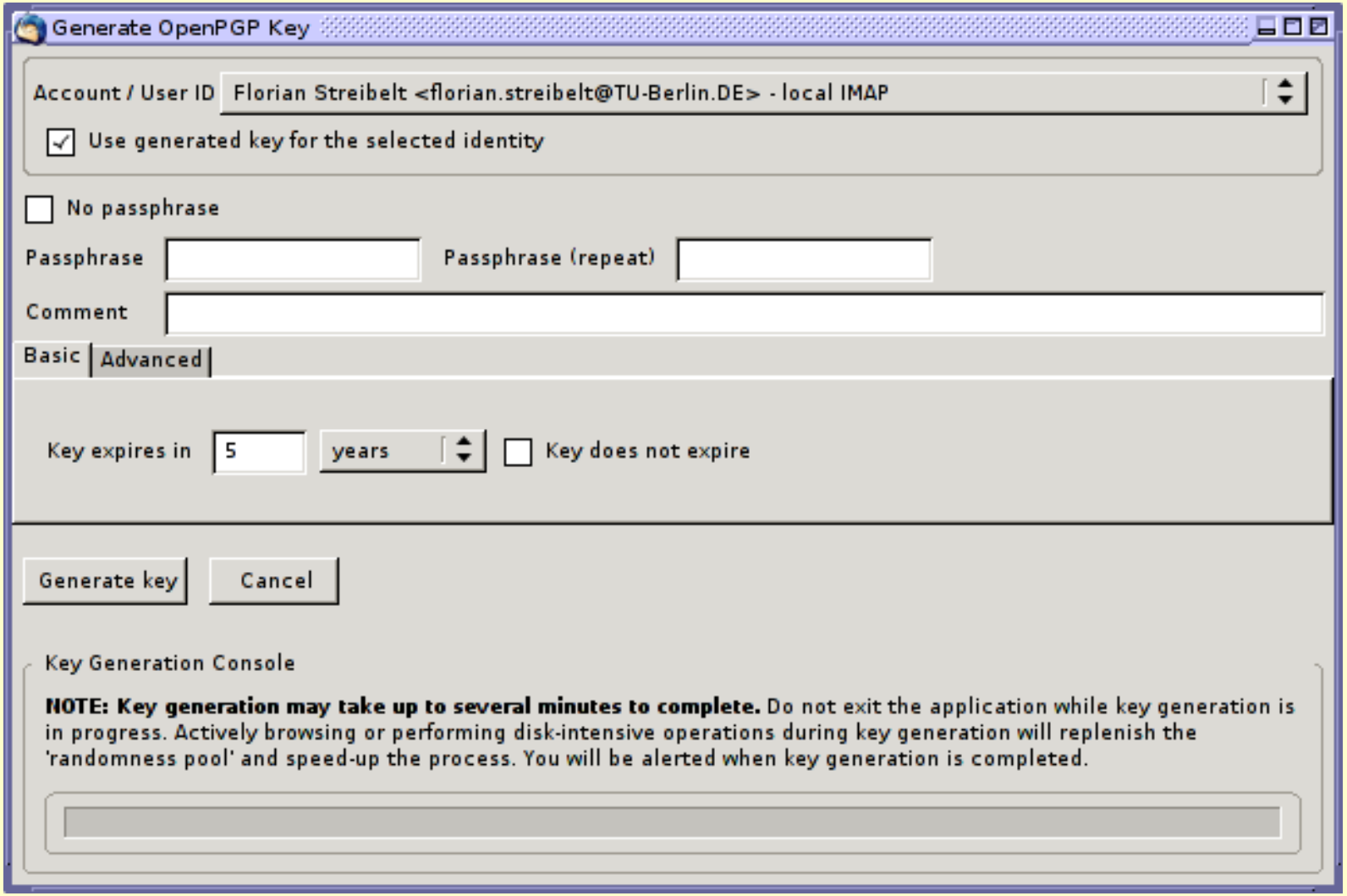
Eingmail – Thunderbird Plugin



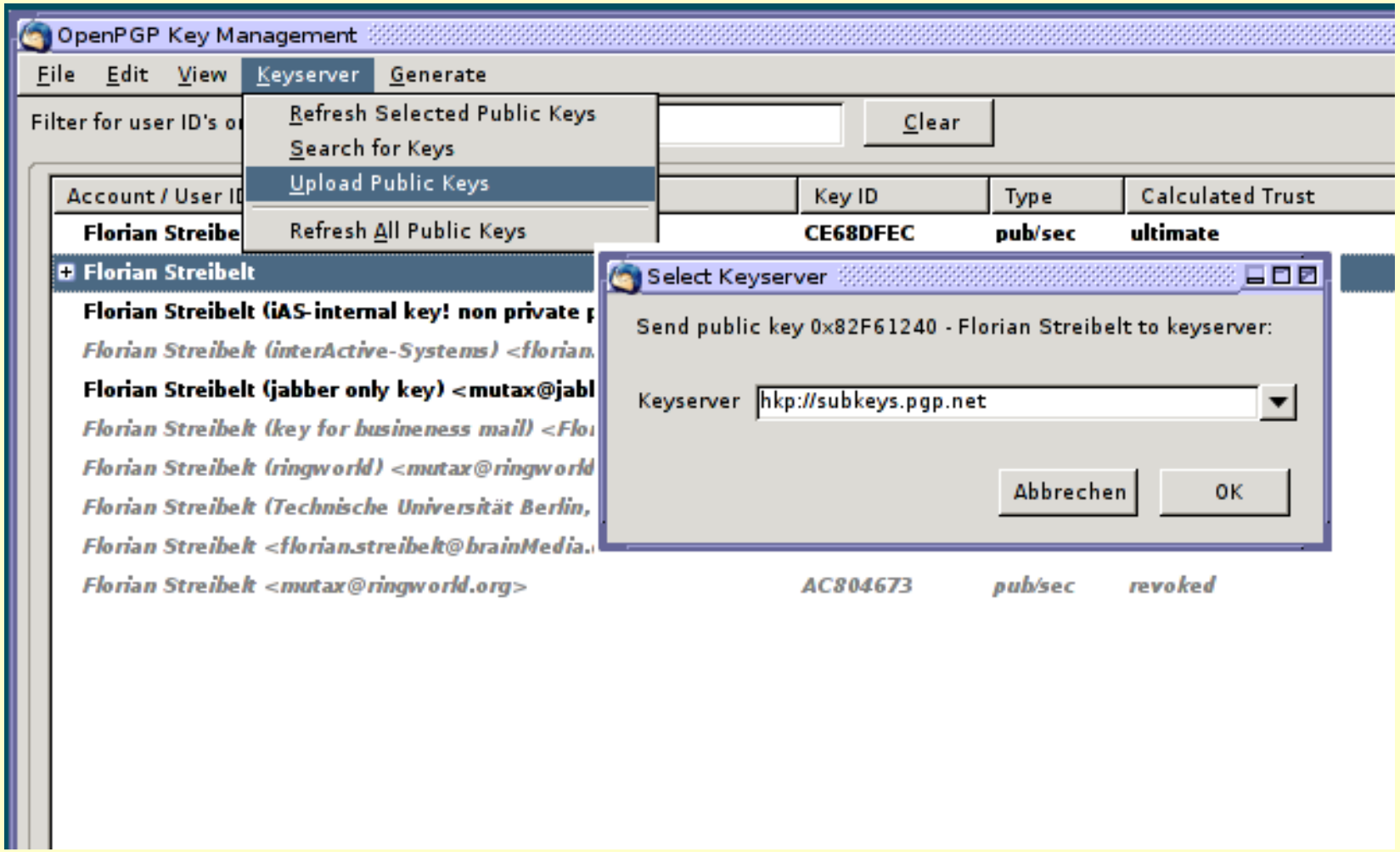
Enigmail – Key Management



Enigmail – Key erzeugen



Enigmail – Key veröffentlichen



Vielen Dank für Ihre Aufmerksamkeit!

Fragen? Fragen!

Keysigningparty:	am Freitag, 17.11.2005 ab 15:00 in FR5516
Anmeldung zur Party:	http://www.freitagsrunde.org/~keys/
Foliendownload unter:	http://docs.freitagsrunde.org/keysigning/
Kontakt:	Florian Streibelt <Florian@freitagsrunde.org>
Howtos:	http://www.gnupg.org/(de)/documentation/howtos.html

GG, Art.10, Abs.1:

***Das* Briefgeheimnis**

***sowie* das Post- und**

***Fern*meldegeheimnis**

***sind* unverletzlich**