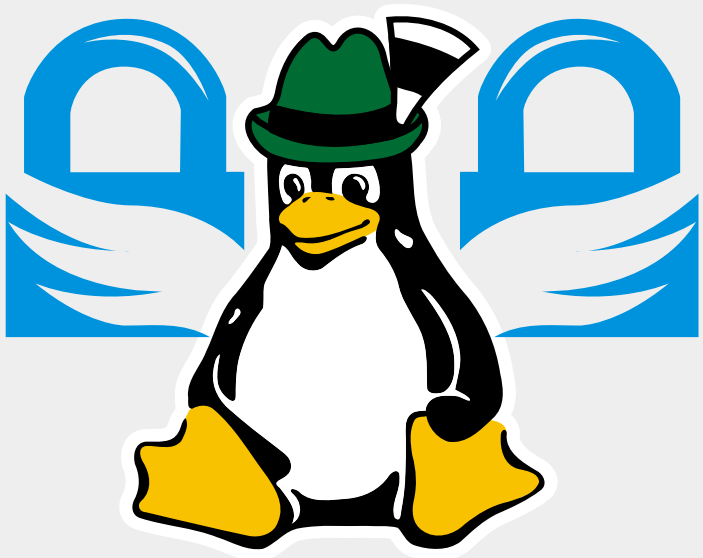
‘GnuPG’

Persönliche Daten durch E-Mailverschlüsselung schützen



Lukas Prokop ⟨admin@lukas-prokop.at⟩

20. April 2013, GLT13

GnuPG?

Abk. Gnu Privacy Guard

- Kryptographische Software
- Freie Alternative zu PGP
- Teil des GNU Projekts (GPLv3 lizensiert)
- Windows und unixoide Systeme
- Implementiert OpenPGP-Standard
- Verschlüsseln, Entschlüsseln, Signieren, Prüfen

Wozu Krypto? Wozu frei?

Wozu Kryptographie?

- Weil wir etwas zu verbergen haben
- Privatsphäre als Grundrecht

Wozu freie Software?

- Software ist nachvollziehbar
- Jeder kann Sicherheit überprüfen

Anforderungen

Vertraulichkeit
Keine dritte Person kennt Inhalt der Nachricht.

Authentifizierung
Nur der Empfänger kann die Nachricht lesen.

Datenintegrität (bzw. Authentizität)
Niemand hat eine Fälschung untergejubelt.

Verfügbarkeit
zuverlässig, schnell, ausfallssicher, benutzbar

Asymmetrische Verschl.

Problem: Vertraulichkeit (oder mehr 😊)
Lösung: Kryptologie

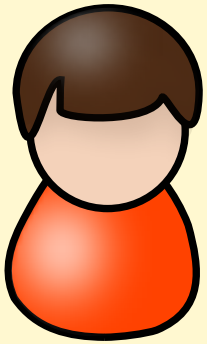
Jede Entität besitzt einen *öffentlichen* und *privaten* Schlüssel. Diese werden verwendet, um Nachrichten zu verschlüsseln, entschlüsseln, signieren und zu prüfen.

Konzept #1

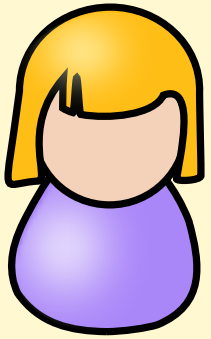
Verschlüsseln und Entschlüsseln

Nachricht wird in eine nicht-interpretierbare Form gebracht.
Entschlüsseln ist Umkehroperation.

2 Entitäten

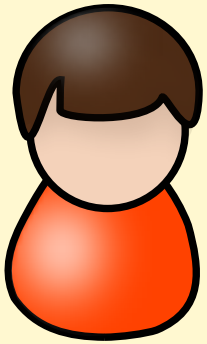


Bob

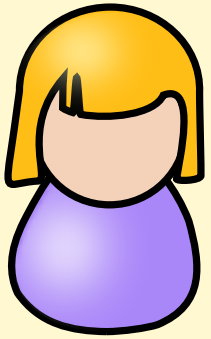


Alice

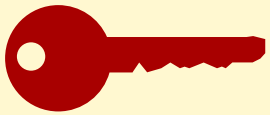
1. Schlüssel generieren

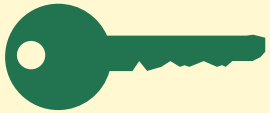


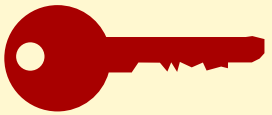
Bob

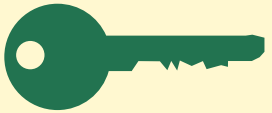


Alice

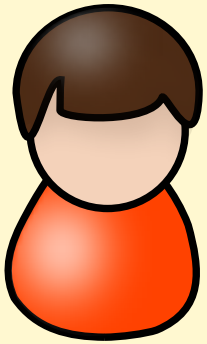




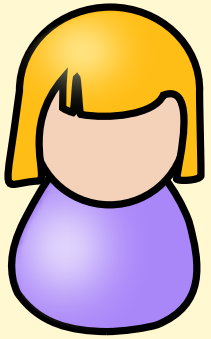




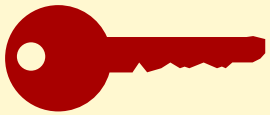
2. Nachricht verfassen

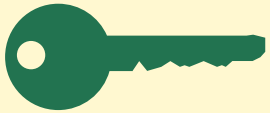


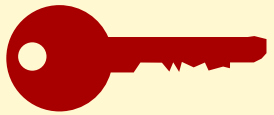
Bob

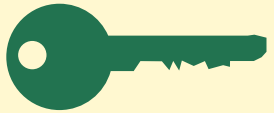


Alice

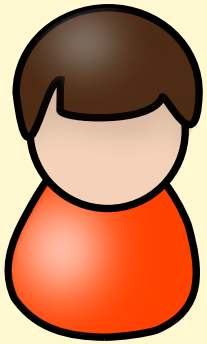




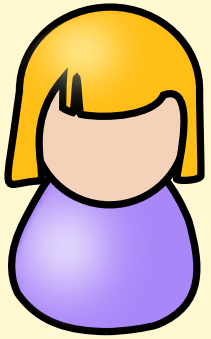




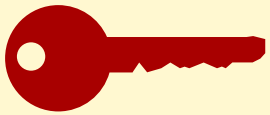
3. Verschlüsseln

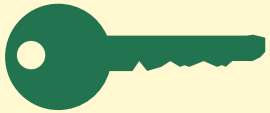


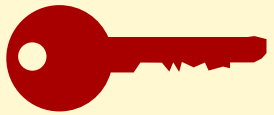
Bob



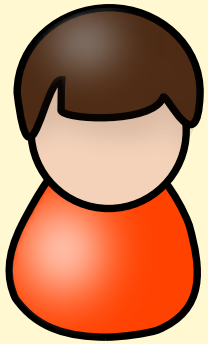
Alice



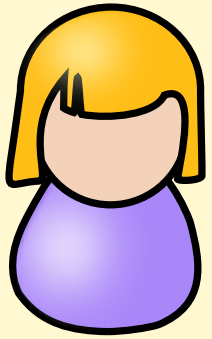




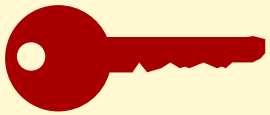
4. Nachricht verschlüsselt

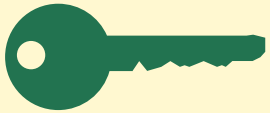


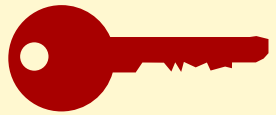
Bob

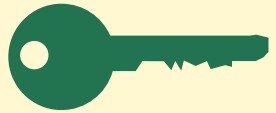


Alice

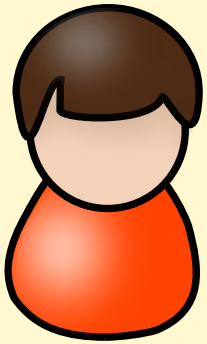




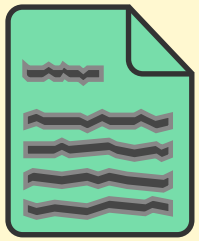


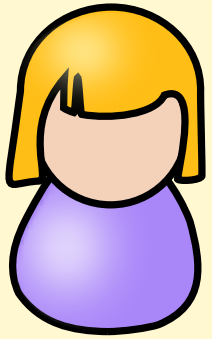


5. Nachricht übertragen

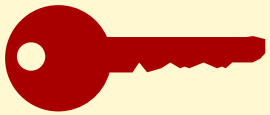


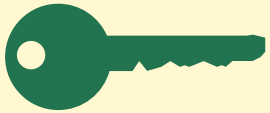
Bob

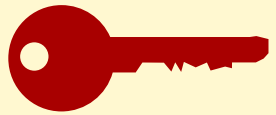


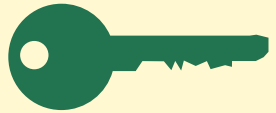


Alice

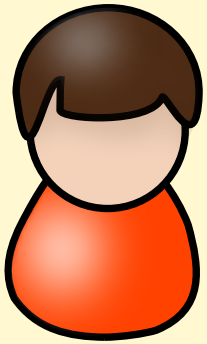




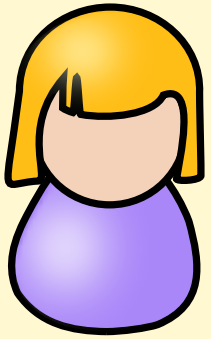




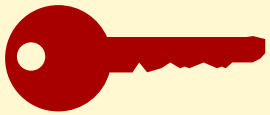
6. Nachricht empfangen

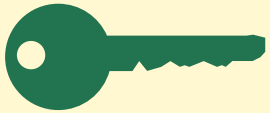


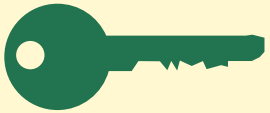
Bob



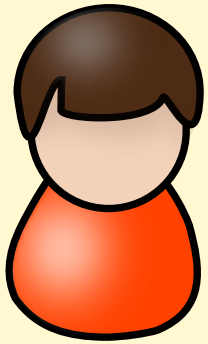
Alice



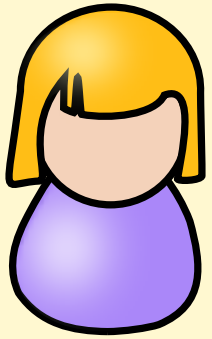




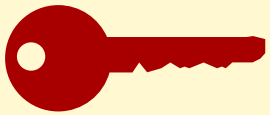
7. Nachricht empfangen

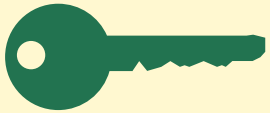


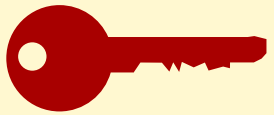
Bob

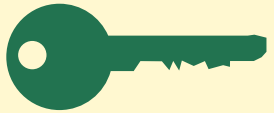


Alice







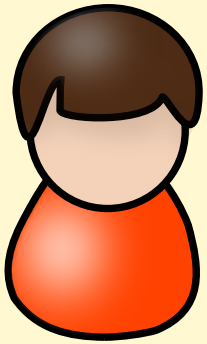


Konzept #2

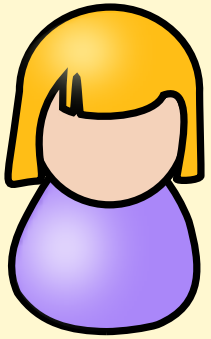
Signieren und Prüfen

Wir versichern, dass wir
die Nachricht erstellt haben.
Prüfen ist die Verifikation davon.

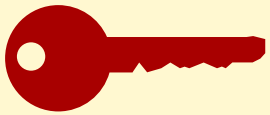
1. Schlüssel generieren

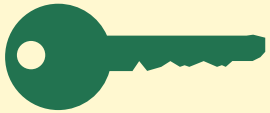


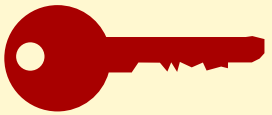
Bob

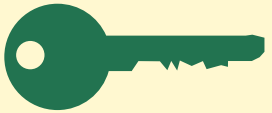


Alice

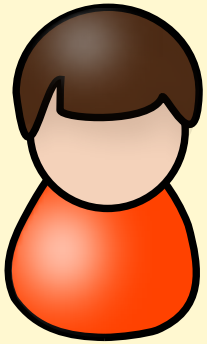




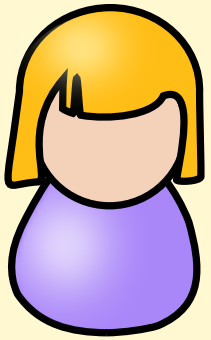




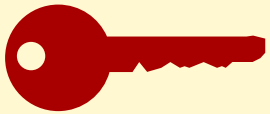
2. Nachricht verfassen

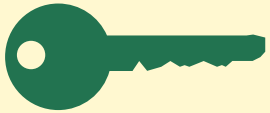


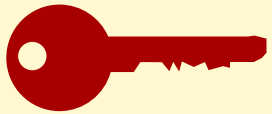
Bob

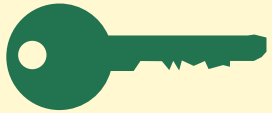


Alice

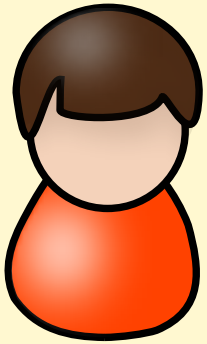




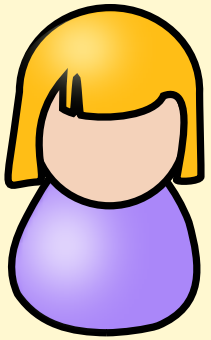




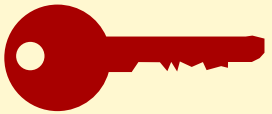
3. Nachricht signieren

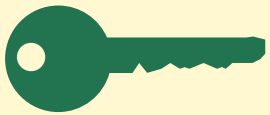


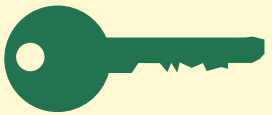
Bob



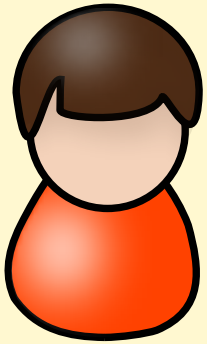
Alice



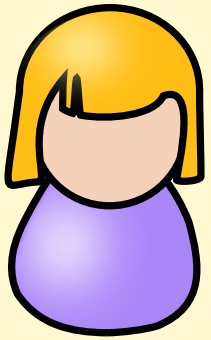




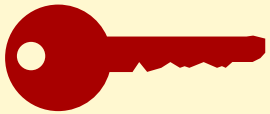
4. Nachricht signiert

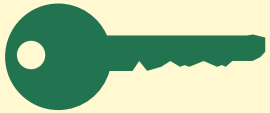


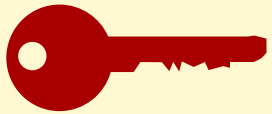
Bob

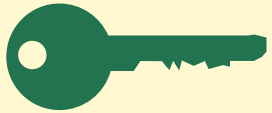


Alice

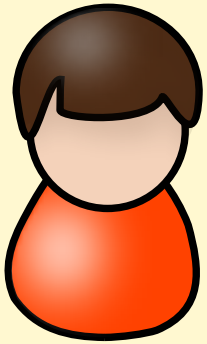




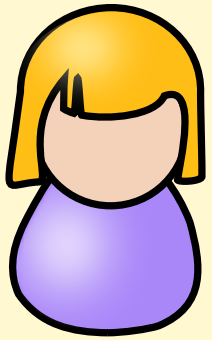




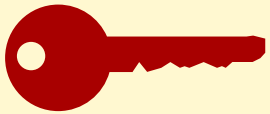
5. Nachricht übertragen

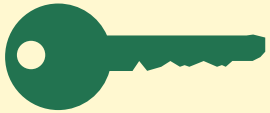


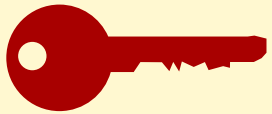
Bob

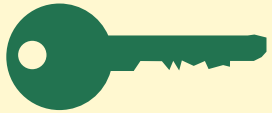


Alice

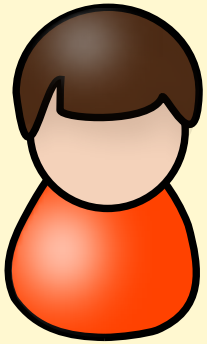




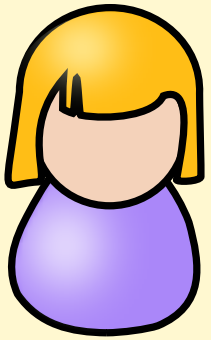




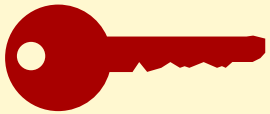
6. Nachricht prüfen

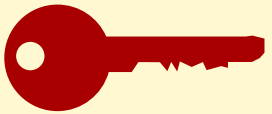


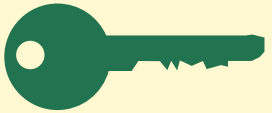
Bob



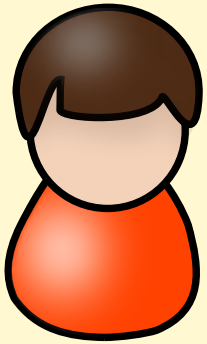
Alice



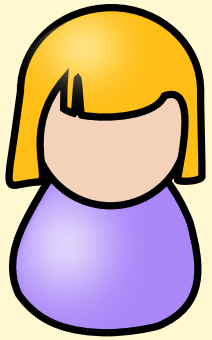




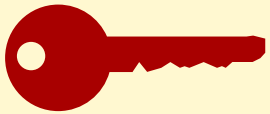
7. Nachricht geprüft

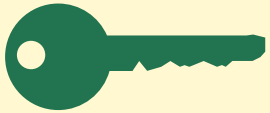


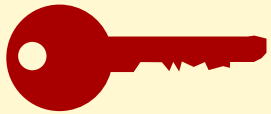
Bob

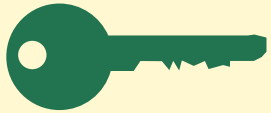


Alice







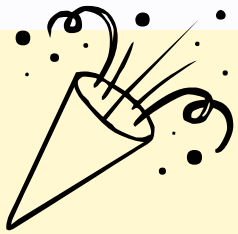


Vertrauen?

Vertrauen als inhärentes Problem.
Wem vertraue ich, dass er die Verfahren korrekt nutzt? Ab wann glaube ich, dass Person A hinter Schlüssel X steht?

- Web of Trust (OpenPGP)
- Public Key Infrastructure (S/MIME)

Konzept #3: Time to Party!



Keysigning Party (KSP)

- Überprüfe Zuordnung Identität zum Schlüssel
- *Identität:* amtliches Dokument
- *Schlüssel:* Fingerabdruck (Schlüssel zu lang)
- Fördert das Web-of-Trust

Heute um 18:30 beim Frontdesk

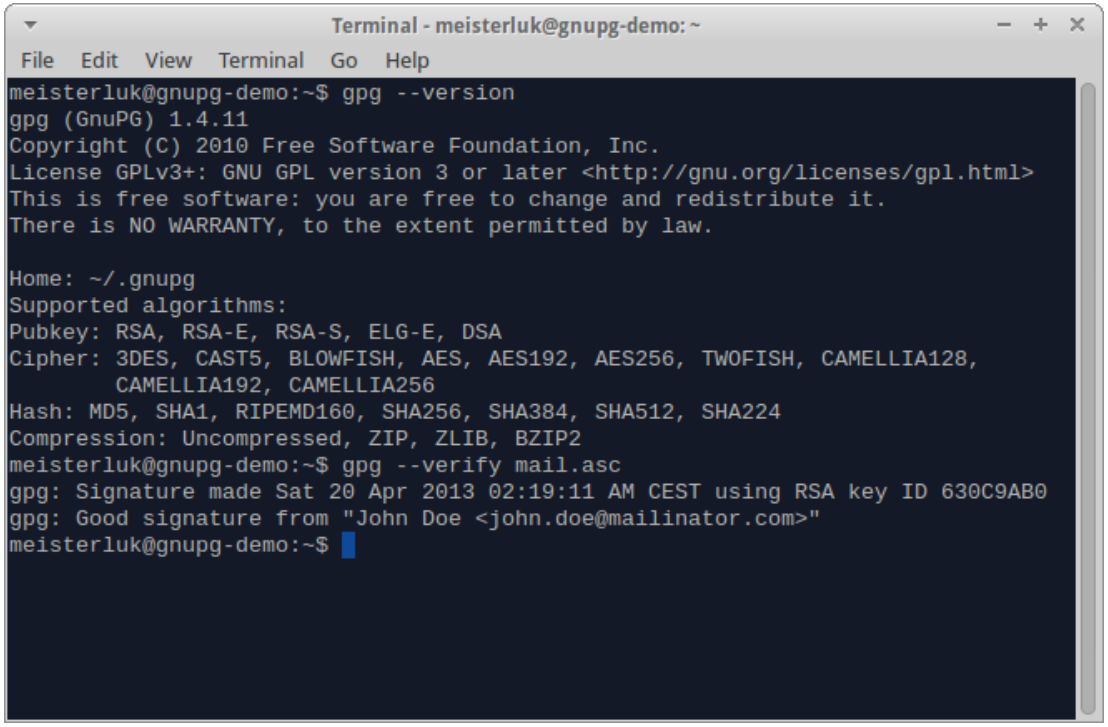
```
gpg --print-md ripemd160 ksp-glt13.v2.txt
```

Konzept #4: Widerruf

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v1.4.10 (GNU/Linux)
Comment: A revocation certificate should follow

iQIfBCACAgAJBQJOEDxIAh0CAAoJELJdgLqFl89bUPoP/jMKMn1g7NZVoDlb2Eud
7DZ5umBf30Czau7OhJ2+ry2JHJzEPTXpyMXaDcXkXCBS5BlKGJG2c+lDhXFs1lbo
RnBm/IFbmj6+HjyNTwiIfhHyvohhGYg2lxmKBbgR/XTxwxg2HixRVIULkhGKAaut
fWUGGBpR8wEoh72brnjEbhHshay6it+J5ClJZe+99Gixu9kTIL/9vlZ5rZwwgaJJ
vtDOFHUgWc5mNlCpVsTdFnPbEqh2P8M3Uy0/i3J6z4Zt2LVClcNlmVsMEh/PaBsm
37bRR2SM8JJprPog1DdyyjJRwUbXO5Jal04fF/r1YI0/3hLOkn/jPJFIzFQ+PeOK
7wFJKiipSRhfJQVsfMoWFQtY1TK/9y1Kp1OFJKUDQfeRY+IngkKkL5iyGH5JHs1L
HTQBYcJfA8repQA5k4YSX1hak9exA3GwoiYxDTUJP/kZX8N7kbolxOwsi3SnlGTZ
gIpho9Ize7Z/zJjyaQS4bxdyaXQB6gd4ef7/y0jHRbKR2B/XSdeuEgekd7lvpAHP
Ouea7zI81P0tClxJuoYiu5gg6DNnSPCziDCBhm0D51hm3oIneCBd7Xf5Sf5wZOLF
Gw1A+H9/6+t1WyxK3YqimozxLFmHEoMOjhDEIB2EpuKareglDS7XiHip2QsCQKSj
N1J5Wf9D3+fdExnsQq8G/N7z
=SZzh
-----END PGP PUBLIC KEY BLOCK-----
```

Demonstration



CLI Befehle

```
gpg --gen-key
```

```
gpg --gen-revoke [ID of your key]
```

```
gpg --refresh-keys
```

```
gpg --send-key [ID of your key]
```

CLI Befehle

```
gpg --armor --encrypt
    -r [ID of partner's key] [filepath]
```

```
gpg --decrypt [filepath]
```

```
gpg --clearsign [filepath]
```

```
gpg --verify [filepath]
```

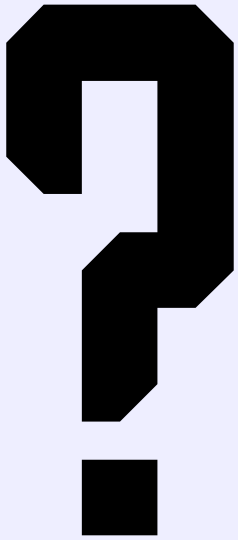
Weiterführendes

- 13:00 HS4, “GnuPG mit Smartcards” von Fridolin
- CryptoPartyGraz
- Security Treff Graz jeden 2. Dienstag des Monats

Weiterführendes

- 13:00 HS4, “GnuPG mit Smartcards” von Fridolin
- CryptoPartyGraz
- Security Treff Graz jeden 2. Dienstag des Monats

- Subkeys
- Mehrere Empfänger
- Attachments



Danke für die Aufmerksamkeit 😊

http://lukas-prokop.at/talks/gnupg/