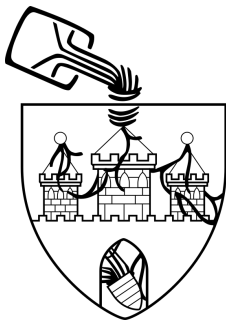


Sichere E-Mails mit PGP

Peter Gewalt & Manuel Groß

2015-11-15

Wer sind wir?



<https://ccc-ol.de/>



<https://mainframe.io/>

Was ist eigentlich unser Problem?



CC-BY-SA 2.0 Markus Winkler

Was ist eigentlich unser Problem?



CC-BY-SA 2.0 Markus Winkler

- Massenhafte, anlasslose Überwachung

Was ist eigentlich unser Problem?



CC-BY-SA 2.0 Markus Winkler

- Massenhafte, anlasslose Überwachung
- False Positives

Was ist eigentlich unser Problem?



CC-BY-SA 2.0 Markus Winkler

- Massenhafte, anlasslose Überwachung
- False Positives
- Chilling Effect

Lösungsvorschlag: Verschlüsselung

-----BEGIN PGP MESSAGE-----
Version: GnuPG v2

[illegible]

- Klartext \rightarrow Geheimtext
- mathematisches Verfahren
- geheime Zahl als „Passwort“

Lösungsvorschlag: Verschlüsselung

-----BEGIN PGP MESSAGE-----
Version: GnuPG v2

[illegible]

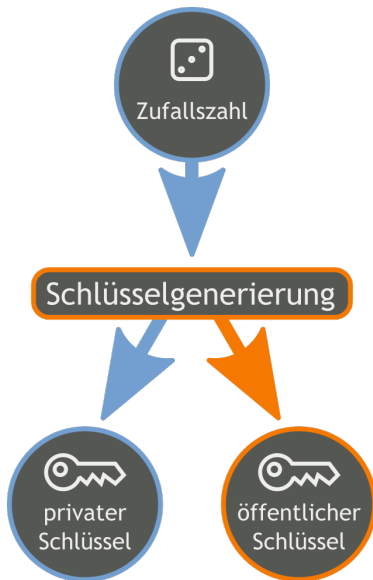
- Klartext \rightarrow Geheimtext
- mathematisches Verfahren
- geheime Zahl als „Passwort“
- symmetrisch
 - ▶ selbes Verfahren
 - ▶ selber Schlüssel

Abgrenzung: Verschlüsselung

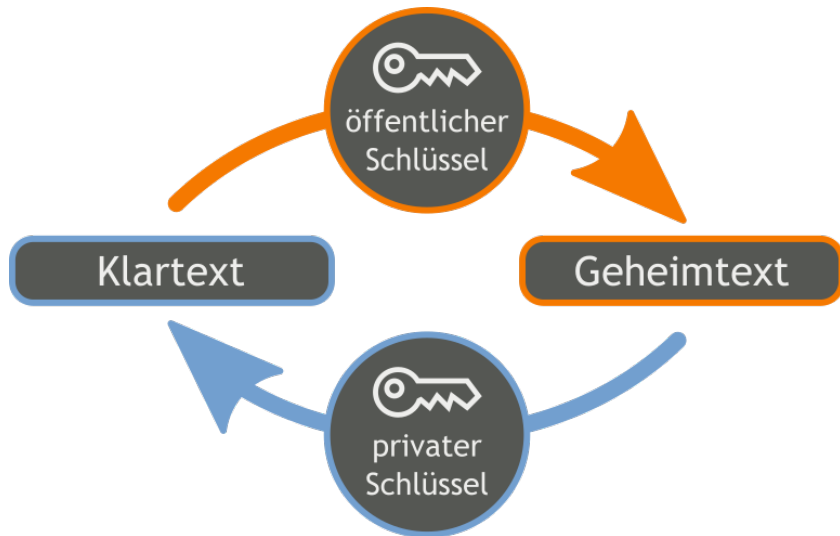
Was ist damit nicht gemeint?

- Transportverschlüsselung
 - ▶ SSL
 - ▶ TLS
 - Nachrichten am Speicherort ungeschützt
- DE-Mail
 - ▶ per default nur abschnittsweise verschlüsselt

Asymmetrische Verschlüsselung



Asymmetrische Verschlüsselung



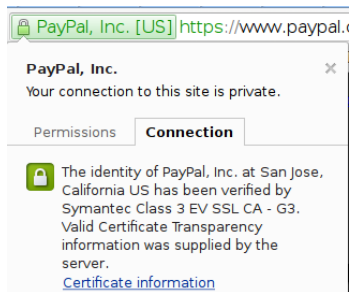
Problem: Vertrauenswürdigkeit

Problem:

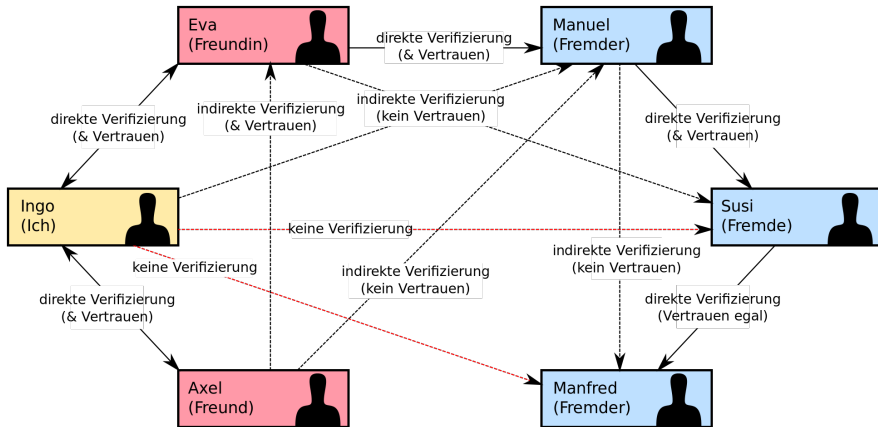
Wer garantiert uns, dass Bob wirklich der Absender ist?

- Idee: zentraler Ansatz

- ▶ Indirekter Austausch über Dritte (z.B. Browser oder Mailprogramm)
- ▶ Hierarchie/Zertifikatsliste (schonmal reingeschaut?)



Unterschiedliche Ansätze: Verteilt



CC-BY-SA 3.0 Hauke Laging

Cryptoparties

```
pub    rsa4096/63B7B190 2014-10-18
       Key fingerprint = FF9A C23C EBC5 CD0D 3998  CB4A CB77 6720 63B7 B190
uid    [ultimate] Manuel Groß <manuel.gross@freifeld-festival.de>
uid    [ultimate] Manuel Groß <mgr@nordkrater.de>
```



CC-BY 2.0 wbritzl

Cryptoparties

```
pub    rsa4096/63B7B190 2014-10-18
       Key fingerprint = FF9A C23C EBC5 CD0D 3998  CB4A CB77 6720 63B7 B190
uid    [ultimate] Manuel Groß <manuel.gross@freifeld-festival.de>
uid    [ultimate] Manuel Groß <mgr@nordkrater.de>
```



CC-BY 2.0 wbritzl

- Neue Leute kennenlernen

Cryptoparties

```
pub    rsa4096/63B7B190 2014-10-18
       Key fingerprint = FF9A C23C EBC5 CD0D 3998  CB4A CB77 6720 63B7 B190
uid    [ultimate] Manuel Groß <manuel.gross@freifeld-festival.de>
uid    [ultimate] Manuel Groß <mgr@nordkrater.de>
```



CC-BY 2.0 wbritzl

- Neue Leute kennenlernen
- Spaß an lustigen, alten, Ausweisbilder anderer haben

Cryptoparties

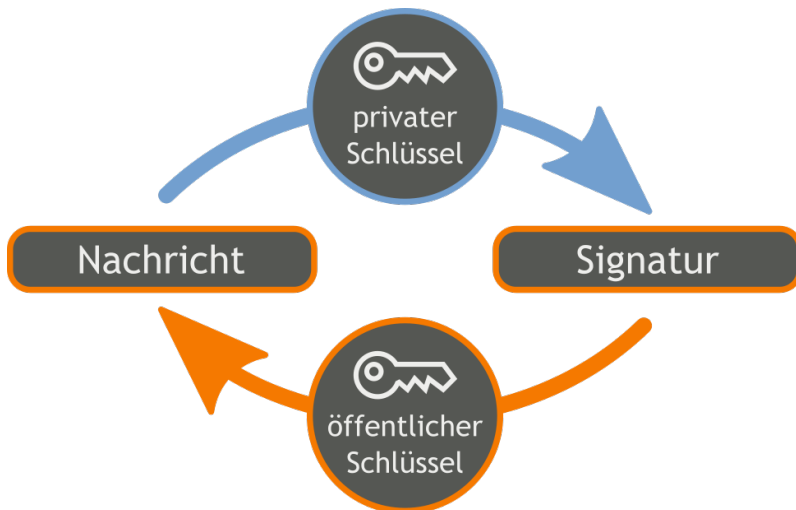
```
pub      rsa4096/63B7B190 2014-10-18
        Key fingerprint = FF9A C23C EBC5 CD0D 3998  CB4A CB77 6720 63B7 B190
uid      [ultimate] Manuel Groß <manuel.gross@freifeld-festival.de>
uid      [ultimate] Manuel Groß <mgr@nordkrater.de>
```



CC-BY 2.0 wbritzl

- Neue Leute kennenlernen
- Spaß an lustigen, alten, Ausweisbilder anderer haben
- Selbst Cryptoparties veranstalten ⇒ Web of trust stärken

Signatur



GPG

Was ist jetzt eigentlich GPG?



<https://gnupg.org>

- „Gnu Privacy Guard“, Open Source-Implementierung von OpenPGP
- „PGP“ („Pretty Good Privacy“), alternative, nun proprietäre Implementierung
- Sehr verbreitet im Open Source Umfeld

GPG

Was ist jetzt eigentlich GPG?



<https://gnupg.org>

- „Gnu Privacy Guard“, Open Source-Implementierung von OpenPGP
- „PGP“ („Pretty Good Privacy“), alternative, nun proprietäre Implementierung
- Sehr verbreitet im Open Source Umfeld
- Eher ungebräuchlich im geschäftlichen Umfeld
- Web of Trust Prinzip

GPG

Was ist jetzt eigentlich GPG?



<https://gnupg.org>

- „Gnu Privacy Guard“, Open Source-Implementierung von OpenPGP
- „PGP“ („Pretty Good Privacy“), alternative, nun proprietäre Implementierung
- Sehr verbreitet im Open Source Umfeld
- Eher ungebräuchlich im geschäftlichen Umfeld
- Web of Trust Prinzip
- Plugins für viele Clients
- Metadaten nicht verschlüsselt!
- Teilweise Probleme mit Wrapping (z.B. [Thunderbird](#))

PGP funktioniert!

TOP SECRET//COMINT//REL TO USA, AUS

TOP SECRET//COMINT//REL TO USA, AUS//20320108

THIS INFORMATION IS DERIVED FROM FAA
COLLECTION UNDER FAA COUNTERTERRORISM CERT

THIS INFORMATION IS PROVIDED FOR INTELLIGENCE PURPOSES IN AN EFFORT
TO DEVELOP POTENTIAL LEADS. IT CANNOT BE USED IN AFFIDAVITS, COURT
PROCEEDINGS OR SUBPOENAS, OR FOR OTHER LEGAL OR JUDICIAL PURPOSES.

██████████@yahoo.com

████████████████████
SIGAD: US-984XN
PDDG: AX
CASE_NOTATION: ██████████
DTG: 31JA0101Z12

Received from: [MINIMIZED US IP ADDRESS]
Date: Mon, 30 Jan 2012 17:01:37 -0800 (PST)
From: ██████████ ██████████@yahoo.com>
Subject: Re: Untitled
To: ██████████@yahoo.com

[OC: No decrypt available for this PGP encrypted message.]

GPG unter Linux - Installation

Debian Pakete:

- gnupg2
- signing-party (für caff)
- msmtpl

GPG unter Linux - Konfiguration

- Config: `~/.gnupg/gpg.conf`
- `keyserver pgp.mit.edu`
- Vorteil: Kein manuelles `--keyserver <keyserver>`

GPG unter Linux - Schlüsselpaar erstellen

- `$ gpg --gen-key`

GPG unter Linux - Schlüsselpaar erstellen

- \$ `gpg --gen-key`
 - ▶ RSA oder DSA?

GPG unter Linux - Schlüsselpaar erstellen

- `$ gpg --gen-key`
 - RSA oder DSA?
 - Schlüssellänge?

GPG unter Linux - Schlüsselpaar erstellen

- `$ gpg --gen-key`
 - ▶ RSA oder DSA?
 - ▶ Schlüssellänge?
 - ▶ Gültigkeit?

GPG unter Linux - Schlüsselpaar erstellen

- \$ `gpg --gen-key`
 - ▶ RSA oder DSA?
 - ▶ Schlüssellänge?
 - ▶ Gültigkeit?
 - ▶ Name (kein Pseudonym)

GPG unter Linux - Schlüsselpaar erstellen

- `$ gpg --gen-key`
 - ▶ RSA oder DSA?
 - ▶ Schlüssellänge?
 - ▶ Gültigkeit?
 - ▶ Name (kein Pseudonym)
 - ▶ Mailadresse

GPG unter Linux - Schlüsselpaar erstellen

- \$ `gpg --gen-key`
 - ▶ RSA oder DSA?
 - ▶ Schlüssellänge?
 - ▶ Gültigkeit?
 - ▶ Name (kein Pseudonym)
 - ▶ Mailadresse
 - ▶ Passphrase (Langes_Passwort > S4l4t)

GPG unter Linux - Schlüsselpaar erstellen

- `$ gpg --gen-key`
 - ▶ RSA oder DSA?
 - ▶ Schlüssellänge?
 - ▶ Gültigkeit?
 - ▶ Name (kein Pseudonym)
 - ▶ Mailadresse
 - ▶ Passphrase (Langes_Passwort > S4l4t)
- Upload (sofern öffentlich):
 - `$ gpg --send-keys <key-id>`

GPG unter Linux - Signieren

- Laden des fremden Keys:

```
$ gpg --recv-keys <key-id>
```

GPG unter Linux - Signieren

- Laden des fremden Keys:

```
$ gpg --recv-keys <key-id>
```

- Prüfung des Fingerprints:

```
$ gpg --fingerprint <key-id>
```

GPG unter Linux - Signieren

- Laden des fremden Keys:

```
$ gpg --recv-keys <key-id>
```

- Prüfung des Fingerprints:

```
$ gpg --fingerprint <key-id>
```

- Identitätsprüfung (Personalausweis, Führerschein)

GPG unter Linux - Signieren

- Laden des fremden Keys:

```
$ gpg --recv-keys <key-id>
```

- Prüfung des Fingerprints:

```
$ gpg --fingerprint <key-id>
```

- Identitätsprüfung (Personalausweis, Führerschein)

- Signieren:

```
$ gpg --sign-key <key-id>
```

GPG unter Linux - Signieren

- Laden des fremden Keys:

```
$ gpg --recv-keys <key-id>
```

- Prüfung des Fingerprints:

```
$ gpg --fingerprint <key-id>
```

- Identitätsprüfung (Personalausweis, Führerschein)

- Signieren:

```
$ gpg --sign-key <key-id>
```

- Passphrase eingeben

GPG unter Linux - Signieren

- Laden des fremden Keys:

```
$ gpg --recv-keys <key-id>
```

- Prüfung des Fingerprints:

```
$ gpg --fingerprint <key-id>
```

- Identitätsprüfung (Personalausweis, Führerschein)

- Signieren:

```
$ gpg --sign-key <key-id>
```

- Passphrase eingeben

- Upload-Varianten:

Unsicher Direkter Upload auf Keyserver

Sicher Verschicken des signierten Keys per Mail **an die signierten Mailadressen**

GPG unter Linux - Signieren mit caff

- Automatisch:

```
$ caff <key-id>
```


GPG unter Linux - Signieren mit caff

- Automatisch:
 \$ caff <key-id>
- Benötigt konfigurierten SMTP-client
- Beispiel msmtpp

GPG unter Linux - Signieren mit caff

- Automatisch:
 \$ caff <key-id>
- Benötigt konfigurierten SMTP-client
- Beispiel msmtprc
- Config: ~/.msmtprc
 - ▶ *host, port, user, password*

GPG unter Linux - Empfängerseite

- Datei *signature.asc* per Mail bekommen

GPG unter Linux - Empfängerseite

- Datei *signature.asc* per Mail bekommen
- Signatur importieren:
`gpg --import signatur.asc`

GPG unter Linux - Empfängerseite

- Datei *signature.asc* per Mail bekommen
- Signatur importieren:
`gpg --import signatur.asc`
- Signatur Uploaden:
`gpg --send-keys <meine key-id>`

GPG unter Linux - Empfängerseite

- Datei *signature.asc* per Mail bekommen
- Signatur importieren:
`gpg --import signatur.asc`
- Signatur Uploaden:
`gpg --send-keys <meine key-id>`
- Done! Ready for GPG-Mails!

GPG Keys Revoken

- Entweder automatisch (Gültigkeit) oder manuell

GPG Keys Revoken

- Entweder automatisch (Gültigkeit) oder manuell
- Wichtig: **Vorher** Cross-signieren

GPG Keys Revoken

- Entweder automatisch (Gültigkeit) oder manuell
- Wichtig: **Vorher** Cross-signieren
- Revoken \neq Löschen

GPG Keys Revoken

- Entweder automatisch (Gültigkeit) oder manuell
- Wichtig: **Vorher** Cross-signieren
- Revoken \neq Löschen
- Widerrufszeugnis:
`gpg --gen-revoke <key-id>`
- Importieren, Uploaden

GPG Keys Revoken

- Entweder automatisch (Gültigkeit) oder manuell
- Wichtig: **Vorher** Cross-signieren
- Revoken \neq Löschen
- Widerrufszeugnis:
`gpg --gen-revoke <key-id>`
- Importieren, Uploaden
- ggf. direkt nach Generierung erzeugen
- Besser: Backup private key!

Hands-on - Let's go!

❶ Schlüsselpaar erstellen

- ▶ `gpg --gen-key`
- ▶ (keyserver definieren)
- ▶ `gpg --send-keys <key-id>`

❷ Signieren mit Identitätsprüfung (z.B. ohne caff)

- ▶ `gpg --recv-keys <key-id>`
- ▶ `gpg --fingerprint <key-id>`
- ▶ `gpg --sign-key <key-id>`

❸ Signaturen per Mail verschicken

- ▶ `caff <key-id>`

❹ Uploaden auf den Keyserver (Empfänger)

- ▶ `gpg --import signatur.asc`
- ▶ `gpg --send-keys <meine key-id>`