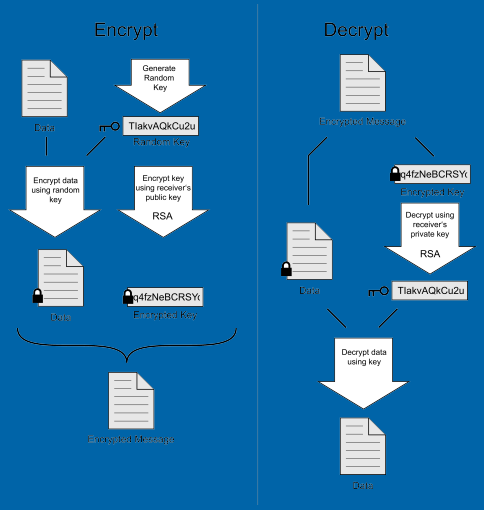
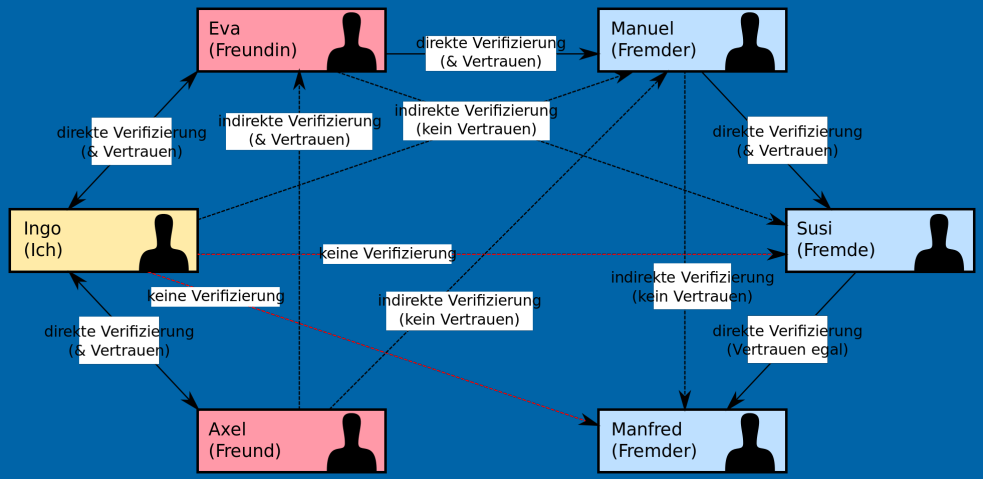
TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
Die Ressourcenuniversität. Seit 1765.

# OpenPGP mit GnuPG

## Geschichte und Grundlagen



# ÜBERBLICK

Was kann man mit OpenPGP machen?
- Möglichkeiten

Wie kam es dazu?
- Geschichte

Verschlüsselungsarten
- Symmetrische Verschlüsselung
- asymmetrische Verschlüsselung
- asymmetrische Signierung
- hybride Verschlüsselung

Das Netz des Vertrauens
- Certificate Authority – X.509 und S/MIME
- Web of Trust
- Schwachpunkte des Web of Trust

# ÜBERBLICK II

## Grenzen von OpenPGP

Was man beim Einsatz beachten muss
Mögliche Angriffe

## Wo bekommt man GnuPG her?

Bezugsquellen

## Stoff zum Lesen

# MÖGLICHKEITEN

- Verschlüsseln
- Entschlüsseln

  $\longrightarrow$ Daten schützen
- Signieren
- Signaturen prüfen

  $\longrightarrow$ Urheber bestätigen
- Anwendung auf Dateien und E-Mail
- ...

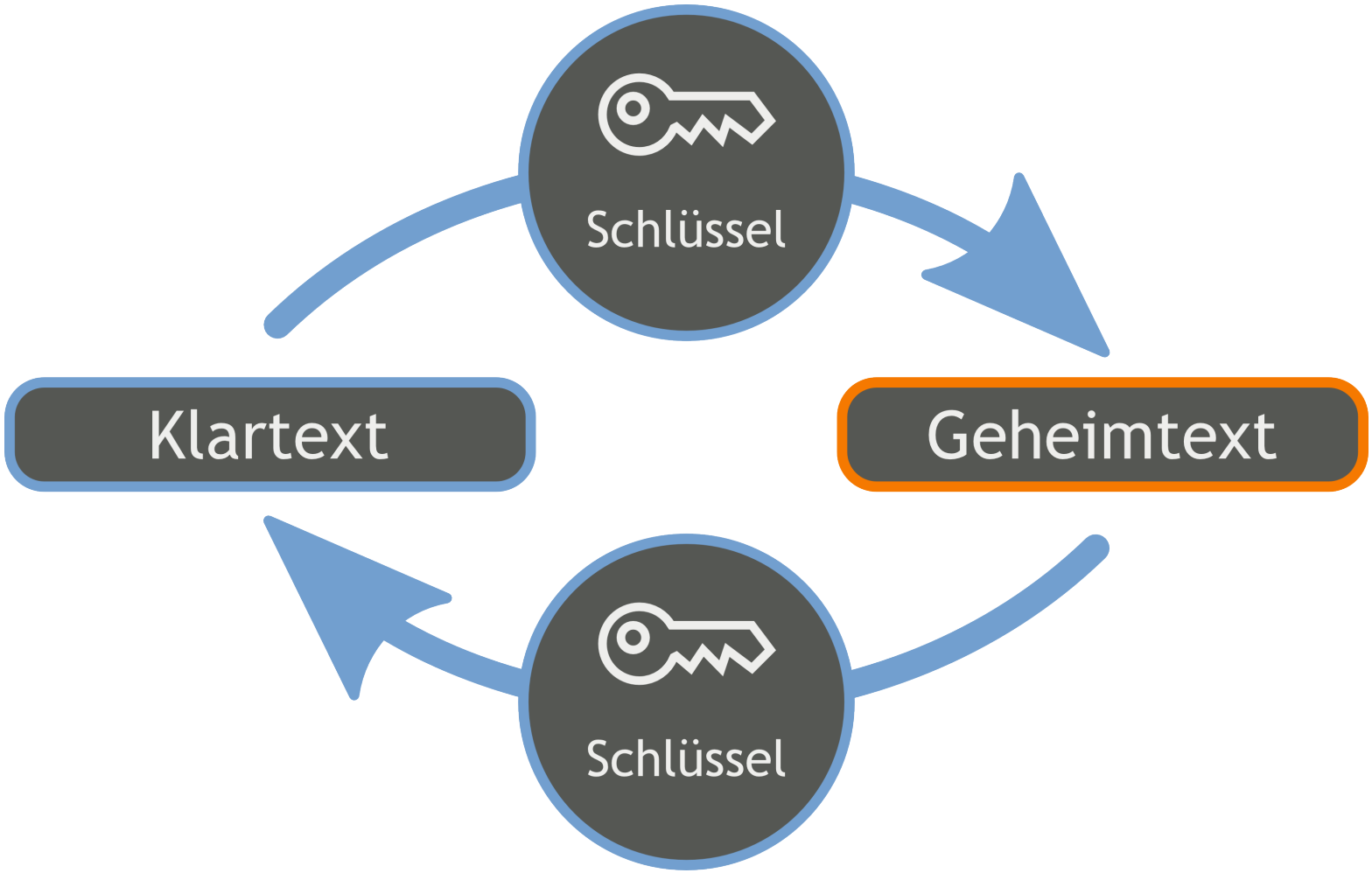
# GESCHICHTE

- 1991: Phil Zimmermann (MIT-Student) schrieb die erste Version von PGP (Pretty Good Privacy), damit Bürger und Bürgerbewegungen starke Verschlüsselung nutzen können.
- 1995: Zimmermann veröffentlicht den gesamten Quellcode in gedruckter Form, um die Exportbeschränkungen der USA (starke Kryptosysteme ab **40 bit** fallen unter Waffenexportregeln) zu umgehen.
- Bis 1998: Wegen der intransparenten Verhältnisse von McAffee – zu der Zimmermanns inszwischen gegründete Firma PGP Inc. gehört –, der Exportbeschränkungen und der Patente auf die Algorithmen IDEA und RSA wird der OpenPGP-Standard entwickelt.
- OpenPGP basiert auf PGP5 und ist in RFC2440 standardisiert.
- 1997: GnuPG (Gnu Privacy Guard) Version 0.0.0 erscheint, Beginn der Implementierung von OpenPGP.
- 1999: GnuPG Version 1.0.0 erscheint: erste Produktivversion
- 2004: GnuPG Version 1.4 erscheint (bis heute gepflegter Zweig der Version 1)
- 2005: GnuPG Version 1.9.16 erscheint, führt S/MIME-Unterstützung ein und ist modular aufgebaut; Basis des zweiten bis heute gepflegen Zweigs Version 2.

# SYMMETRISCHE VERSCHLÜSSELUNG

- „uralte“ Verschlüsselung
- Ein und der selbe Schlüssel für Verschlüsselung und Entschlüsselung
- Dieser muss geheim gehalten werden
- Er muss auf geheimem, sicherem Wege ausgetauscht werden
- Für jeden potentiellen Kommunikationsweg wird ein eigener Schlüssel benötigt → Schlüsselzahl wächst quadratisch mit der Gruppe der Kommunikationspartner: n(n-1)/2 Schlüssel für n Partner, also bei 10 Personen bereits 45 Schlüssel
- Vorteil: geringer Rechenaufwand → schnell
- Beispiel: ZIP-Archiv mit Passwort, LuKS-Container
- Algorithmen: AES, DES oder Lucifer, Triple-DES, IDEA, Blowfish, QUISCI, Twofish, CAST-128, CAST-256, RC2, RC4, RC5, RC6

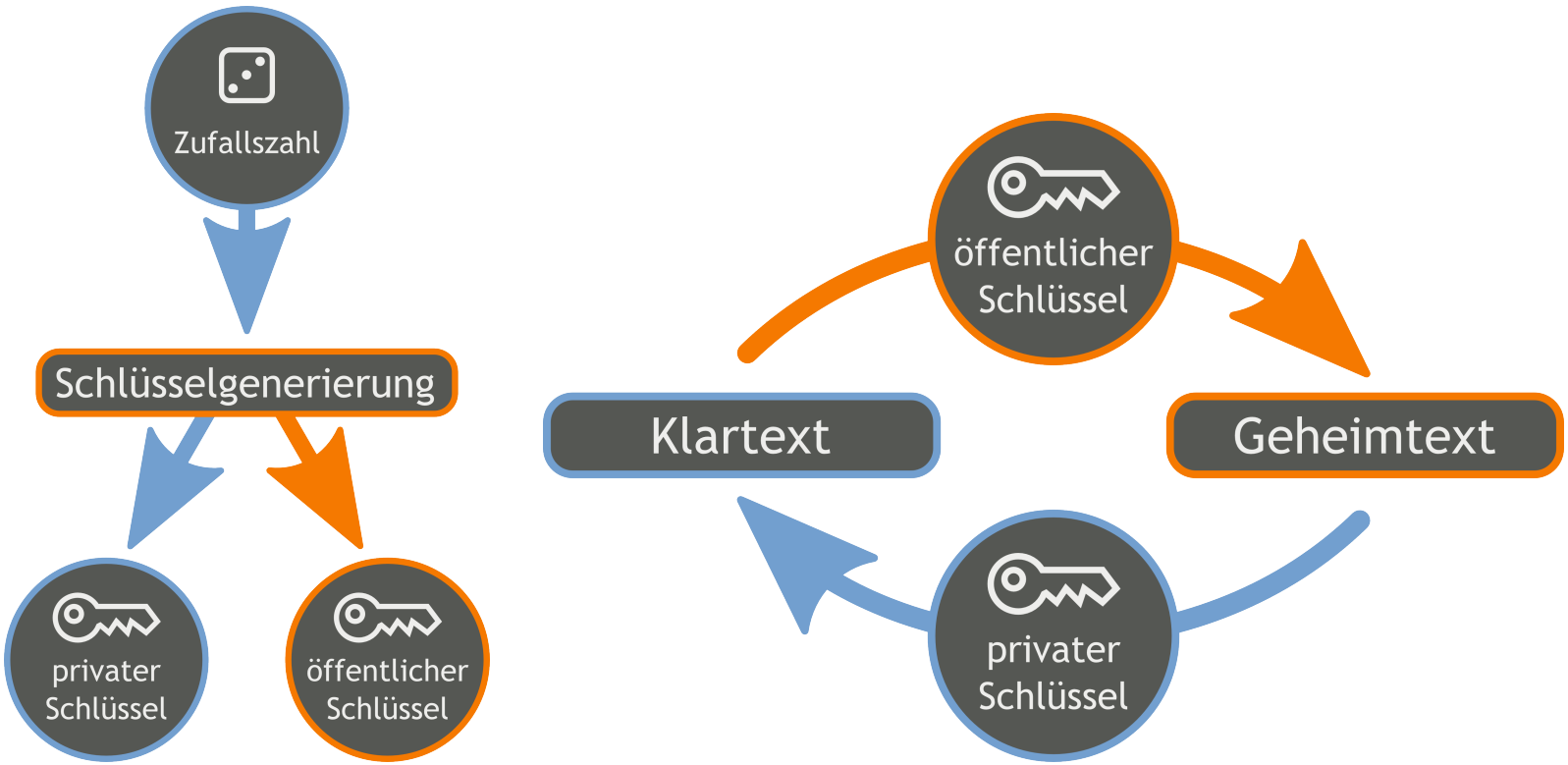
# SYMMETRISCHE VERSCHLÜSSELUNG



# ASYMMETRISCHE VERSCHLÜSSELUNG

- Seit Mitte der 1970er Jahre (Patent auf RSA 1977)
- Grundlage: Funktionen, die einfach zu berechnen, aber ohne Kenntnis eines Geheimnisses praktisch unmöglich invertierbar sind (**Falltür-Funktionen**)
- Verwendung von Schlüsselpaaren:
  → öffentlicher Schlüssel zum Verschlüsseln und Prüfen von Signaturen
  → privater Schlüssel zum Entschlüsseln und Signieren; der private Schlüssel ist die „Falltür“
- Es werden nur 2n Schlüssel bei n Kommunikationspartnern benötigt, dabei hält jeder Kommunikationspartner eine Datenbank von n+1 Schlüsseln vor (das eigene Schlüsselpaar und die öffentlichen Schlüssel der restlichen Partner), worin nur ein Schlüssel geheimzuhalten ist.
- Die Hälfte der Schlüssel wird niemals übermittelt (private Schlüssel).
- Die öffentlichen Schlüssel können auf beliebigem Wege ausgetauscht werden, müssen aber auf einem sicheren Kanal (persönlich, notfalls Telefon) bestätigt werden.
- Alternativ ist zentrale Zertifizierung denkbar (X.509 bei S/MIME).
- Nachteil: erheblicher Rechenaufwand, damit ca. 10.000mal langsamer als symmetrische Verfahren
- Algorithmen: RSA, McEliece, Rabin, Chor-Rivest, Elgamal, Merkle/Hellman, LUC, MNLN, DSA

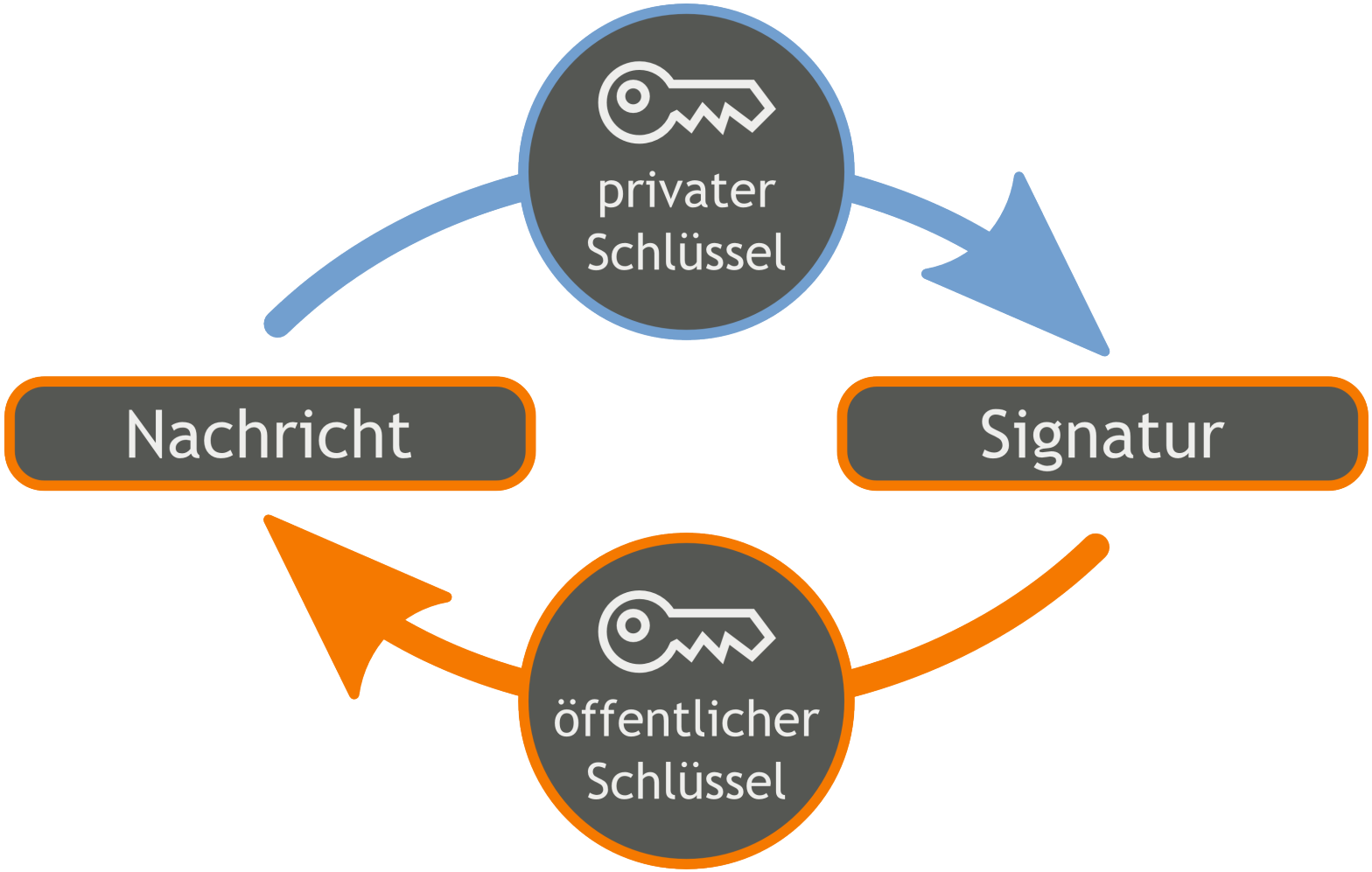
# ASYMMETRISCHE VERSCHLÜSSELUNG



# ASYMMETRISCHE SIGNIERUNG

- Bilden eines Hashwertes der zu signierenden Daten
- Verschlüsseln des Hash-Wertes mit dem privaten Schlüssel
- Verifikation der Signatur mit dem öffentlichen Schlüssel
  $\rightarrow$ Sicherstellen, dass die Daten von einem bestimmten Urheber stammen und auf dem Weg nicht verändert wurden

# ASYMMETRISCHE SIGNIERUNG



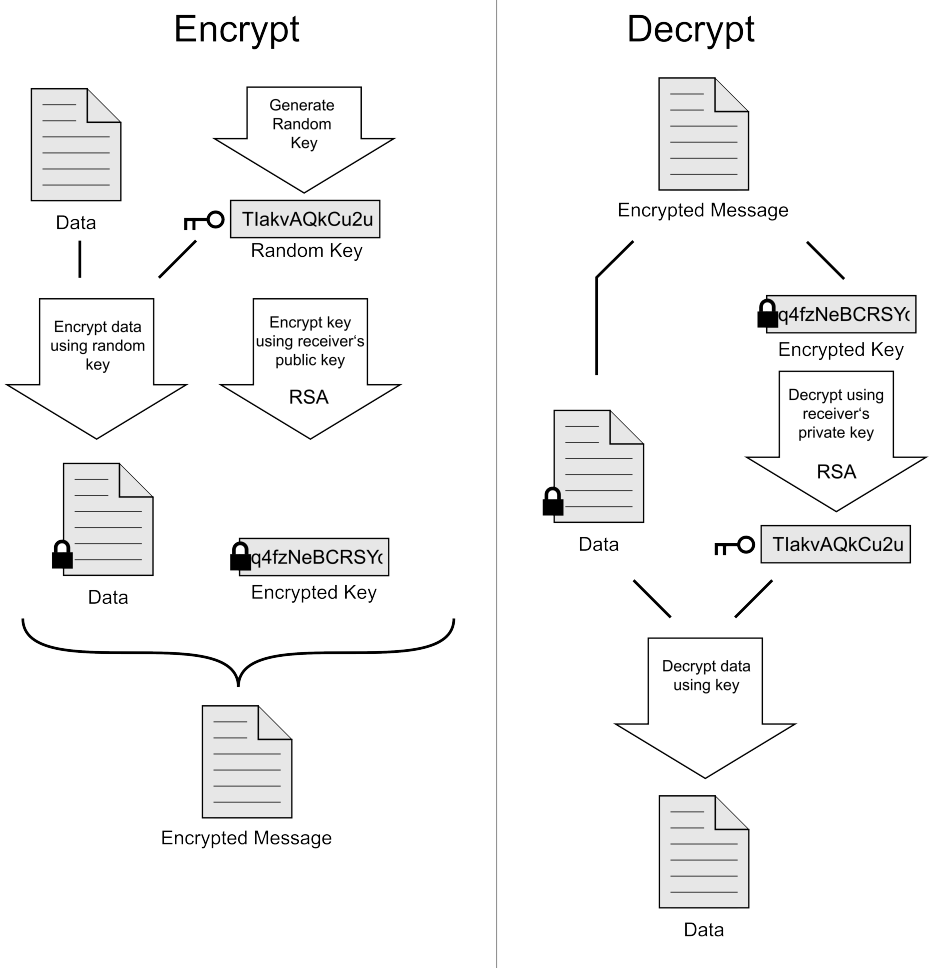
# HYBRIDE VERSCHLÜSSELUNG

- Erzeugen eines zufälligen symmetrischen Einwegschlüssels
- Verschlüsseln der Daten mit dem Einwegschlüssel
- Verschlüsseln des Einwegschlüssels mit dem öffentlichen Schlüssel des Empfängers
- Entschlüsseln des Einwegschlüssels mit dem privaten Schlüssel des Empfängers
- Symmetrisches Entschlüsseln der Daten mit dem „mitgelieferten“ Einwegschlüssel

→ Kombination der Vorteile symmetrischer und asymmetrischer Verfahren; **diesen Weg geht OpenPGP**

→ Verschlüsselung für mehrere Empfänger

# HYBRIDE VERSCHLÜSSELUNG



# CERTIFICATE AUTHORITY – X.509 UND S/MIME

- Zertifikate nach X.509 werden von zentralen Stellen (CAs) ausgegeben
- Echtheit des Besitzers wird überprüft und bescheinigt
- Drei Stufen: 1: Echtheit der E-Mail-Adresse, 2: Echtheit von Name und E-Mail-Adresse, bestätigt über Ausweiskopien und Drittdatenbanken, 3: der Antragsteller hat sich persönlich ausgewiesen
- Anwendung für E-Mailverschlüsselung mit S/MIME oder gesicherte Internetverbindungen mit SSL/TLS
- Zertifizierung kostet üblicherweise Geld
- Für Signaturen nach dem Deutschen Signaturgesetz - rechtverbindliche digitale Unterschrift - erforderlich (CA muss BSI-Richtlinien erfüllen)
- Weniger unterstützte Algorithmen als OpenPGP
- Im Unternehmensumfeld teils verbreiteter
- Mit GnuPG V2 umsetzbar
- Trivia: Mark Shuttleworth verdiente einen Großteil des Geldes, welches jetzt in Ubuntu fließt, als Gründer und Betreiber einer solchen Zertifizierungsstelle.

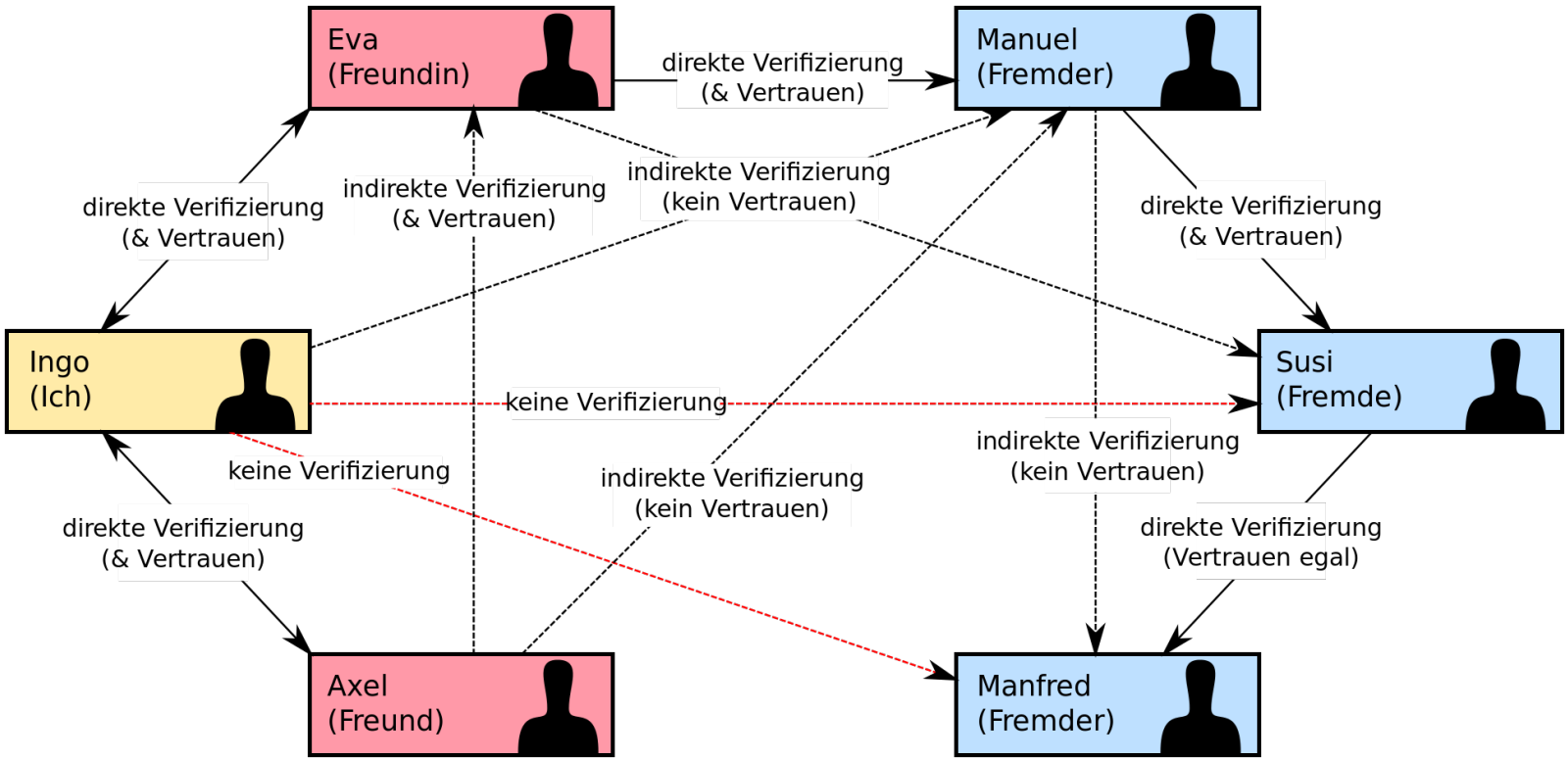
# WEB OF TRUST

- Netz von gegenseitigen Bestätigungen (Signaturen)
- kombiniert mit dem individuell zugewiesenen Vertrauen in die Bestätigungen der anderen (Besitzervertrauen)
- zwei Arten von Vertrauen:
  1. Einerseits vertraut man darauf, dass ein Schlüssel (den man nicht selber signiert hat) gültig (authentisch) ist, also der Besitzer des Schlüssels wirklich die Person (oder Institution) ist, die man dafür hält.
  2. Andererseits vertraut man darauf (oder auch nur teilweise oder gar nicht), dass der Besitzer eines Schlüssels nur sorgfältige Schlüsselsignaturen vornimmt, dass also die mit der Signatur zum Ausdruck gebrachte Behauptung desjenigen, dass ein bestimmter Schlüssel zu einer bestimmten Person gehört, verlässlich ist. Dieses Vertrauen wird Owner Trust (Besitzervertrauen) genannt.

# WEB OF TRUST II

- diese sind voneinander unabhängig:
  1. Man kann sich sicher sein, dass ein bestimmter Schlüssel zu einer bestimmten Person gehört. Diese Überzeugung wird in keiner Weise dadurch erschüttert, dass man die Schlüsselsignaturen dieser Person als wertlos betrachtet.
  2. Man kann den Schlüsselsignaturen einer Person voll vertrauen, ohne über einen als gültig betrachteten Schlüssel dieser Person zu verfügen. (Bis zur Verfügbarkeit eines gültigen Schlüssels sind eventuelle Schlüsselsignaturen der Person allerdings wertlos.)
- Beispiel:
  - Alice signiert den Schlüssel von Bob und vertraut Bobs Schlüsselsignaturen
  - Bob signiert den Schlüssel von Carl (Bobs Vertrauen in Carls Schlüsselsignaturen ist weder bekannt noch relevant)
  - $\rightarrow$ Somit betrachtet Alice den Schlüssel von Carl als gültig.

# WEB OF TRUST


Eva
(Freundin)
Manuel
(Fremder)
direkte Verifizierung
(& Vertrauen)
direkte Verifizierung
(& Vertrauen)
indirekte Verifizierung
(& Vertrauen)
indirekte Verifizierung
(kein Vertrauen)
direkte Verifizierung
(& Vertrauen)
Ingo
(Ich)
Susi
(Fremde)
keine Verifizierung
keine Verifizierung
indirekte Verifizierung
(kein Vertrauen)
direkte Verifizierung
(& Vertrauen)
indirekte Verifizierung
(kein Vertrauen)
direkte Verifizierung
(Vertrauen egal)
Axel
(Freund)
Manfred
(Fremder)

# SCHWACHPUNKTE DES WEB OF TRUST

- Die Schlüssel enthalten keine Information, wie sicher sie sind und wofür sie verwendet werden.
- Die Zertifizierungen enthalten keine Information, wie sicher die zertifizierten Schlüssel sind und wofür sie verwendet werden.
- Die Zertifizierungen enthalten keine Information, was (Name, E-Mail, ggf. Kommentar) geprüft wurde.
- Die Zertifizierungen enthalten keine Information, in welchem Verhältnis Zertifizierer und Zertifizierter zueinander stehen und wie ggf. die Identität geprüft wurde.
- Kein reines WoT-Problem: Die meisten Schlüssel haben keine Angabe dazu, wo man verbindliche Schlüsselupdates bekommt. Zur verlässlichen Nutzung von Schlüsseln gehört aber auch eine verlässliche Handhabung von Schlüsselwiderrufen inklusive einer belastbaren Dokumentation, wann der Widerruf veröffentlicht wurde. Für X.509 ist das brauchbar gelöst.
- Man bekommt keine Bestätigung dafür, dass man zu einem bestimmten Zeitpunkt eine Aktualisierung des Zertifikats (inklusive Widerruf) durchgeführt hat und wie die aussah.

$\longrightarrow$ Anfänger sollten nur selbst überprüfte Schlüssel verwenden

# WAS MAN BEIM EINSATZ BEACHTEN MUSS

- Sicherheit ist unbequem
- Der Schlüssel ist das stärkste Glied, schwächer sind die Passphrase, das benutze Computersystem und die Sorgfalt des Anwenders.
- Nichtverifizierte Schlüssel sind praktisch wertlos
- Genannte Schwächen des WoT
- Schlüsselaktualität
- Sicherheit hängt auch von Verfahren ab (MD5?)
- Fehler lassen sich oft nicht ausgleichen
- Kompetenz und Disziplin aller Kommunikationspartner

# MÖGLICHE ANGRIFFE

- Unterschieben falscher öffentlicher Schlüssel
- Entwenden des geheimen Schlüssels
  → Wörterbuchangriff auf die Passphrase
  → Ausspähen der Passphrase

# BEZUGSQUELLEN

- Linux: Paketverwaltung der eigenen Distribution, sonst: `http://www.gnupg.org/download/index.html` (Quelltext und Links)
- MacOS: `https://gpgtools.org`
- Windows: `http://gpg4win.org`

# WEITERE INFORMATIONEN


[1] URL: https://emailselfdefense.fsf.org/de/.

[2] URL: http://www.gnupg.org.

[3] URL: https://wiki.kairaven.de/open/krypto/gpg/gpganleitung.

[4] URL: http://www.hauke-laging.de/sicherheit/openpgp.html.

[5] URL: http://www.selbstdatenschutz.info.

[6] URL: http://wiki.ubuntuusers.de/GnuPG.

[7] URL: http://de.wikipedia.org/wiki/OpenPGP.

[8] URL: http://de.wikipedia.org/wiki/Pretty_Good_Privacy.

# WEITERE INFORMATIONEN II

[9] URL: http://de.wikipedia.org/wiki/GNU_Privacy_Guard.

[10] URL: http://de.wikipedia.org/wiki/Web_of_Trust.

[11] URL: http://de.wikipedia.org/wiki/Public_Key.

[12] URL: http://de.wikipedia.org/wiki/Symmetrisches_Kryptosystem.

[13] URL: http://de.wikipedia.org/wiki/Public-Key-Kryptographie.

[14] URL: http://de.wikipedia.org/wiki/RSA-Kryptosystem.

# VIELEN DANK! – NOCH FRAGEN?

- sandigf@mailserver.tu-freiberg.de
- Ekkehardt in #flux auf secundus.stunet.tu-freiberg.de
- Präsentation unter `http://sandig-fg.de`