Netzsicherheit – Architekturen und Protokolle
PGP

1. PGP PKI
2. E-Mail Sicherheit

Netzsicherheit – Architekturen und Protokolle
PGP

1. PGP PKI
2. E-Mail Sicherheit

PGP: Pretty Good Privacy

- Entwickler: Phil R. Zimmermann
- IETF-Standard: *OpenPGP*
- PGP definiert Formate und Prozeduren für:
  - Digitale Signaturen
  - Verschlüsselung
  - Kompression
  - Radix-64-Umwandlung (binär Daten in druckbare ASCII-Zeichen)
  - Anwendungsspezifische Unterschlüssel

„PGP is for people who prefer to pack their own parachutes“

(P.R. Zimmermann)

•Im folgenden: PGP = OpenPGP

Vertrauensmodell von PGP

- Vertrauensmodell Anarchie: *Web of Trust*
  - Vollständig dezentral
    - Verzicht auf zentrale Strukturen
  - Verantwortung beim Benutzer
    - Widerspiegelung sozialer Netze
    - Key Signing
    - Vertrauen auf Sorgfalt anderer Benutzer
  - Jeder Benutzer agiert als CA
    - *Trust Introducer* (TI) stellen Vertrauen in ID-Zertifikate her
    - *Meta Trust Introducer* (MTI) stellen transitiv Vertrauen in ID-Zertifikate her

•In PGP wird das Vertrauensmodell Anarchie benutzt (siehe vorne). Dieses Vertrauensmodell heißt im Kontext von PGP „Web of Trust“. In PGP wird Vertrauen in ID-Zertifikate durch Trust Introducer hergestellt.

•Verantwortung beim Benutzer selbst:

- •Schlüsselerstellung
- •Konfiguration von Introducern
- •Widerruf des eigenen Zertifikats bei Bedarf

Public Key Ring

- Zertifikate werden in einem Schlüsselbund (Public Key Ring) verwaltet
  - öffentlicher Schlüssel
  - User-ID
  - keine, eine oder mehrere Signaturen
- Folgende Felder definieren Vertrauen
  - **Owner Trust:** Vertrauen in einen Benutzer, Zertifikate mit zu unterschreiben
  - **Signature Trust:** Vertrauen in eine Signatur, die eine Zuordnung Public Key↔ID zertifiziert
    - **Key Legitimacy:** Vertrauen in Zuordnung Public Key↔ID
    - mehrere Signature Trust Felder zur Berechnung der Key Legitimacy

•In PGP werden öffentlicher Schlüssel und User-ID üblicherweise von mehreren Benutzern unterschrieben (sogenanntes Key Signing).

PGP-Zertifikate

- Struktur

Feld	Bedeutung
Public Key Packet	Primärer Schlüssel
[Signature Packet]*	Widerrufsignatur
[Signature Packet]	Selbstsignatur des Schlüssels
[User-Id Packet [Signature Packet]*]+	User-IDs plus Signaturen, die diese zertifizieren
[Public Subkey Packet Signature Packet [Signature Packet]]*	Unterschlüssel • Bindung an primären Schlüssel • Widerrufssignatur
Trust Packet	Lokale Vertrauensparameter

•In dieser Darstellung bedeutet []*, dass der Inhalt beliebig oft vorkommen kann und []+, dass der Inhalt mindestens einmal vorkommt.

Mögliche Werte der Vertrauensfelder

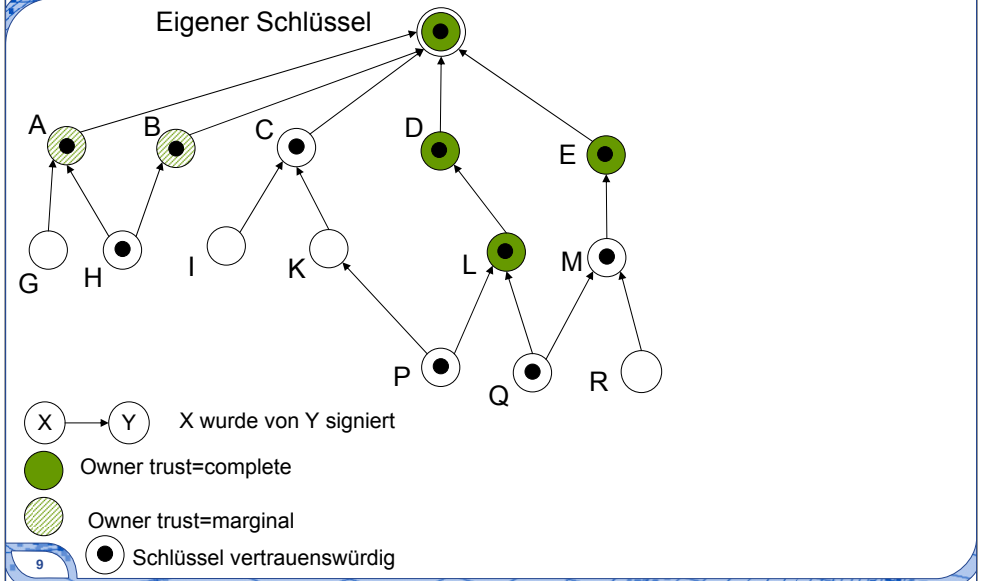
- Owner Trust und Signature Trust
  - unknown
  - untrusted
  - marginal
  - complete
- Key Legitimacy
  - valid
  - marginally valid
  - invalid

Hinzufügen eines öffentlichen Schlüssels

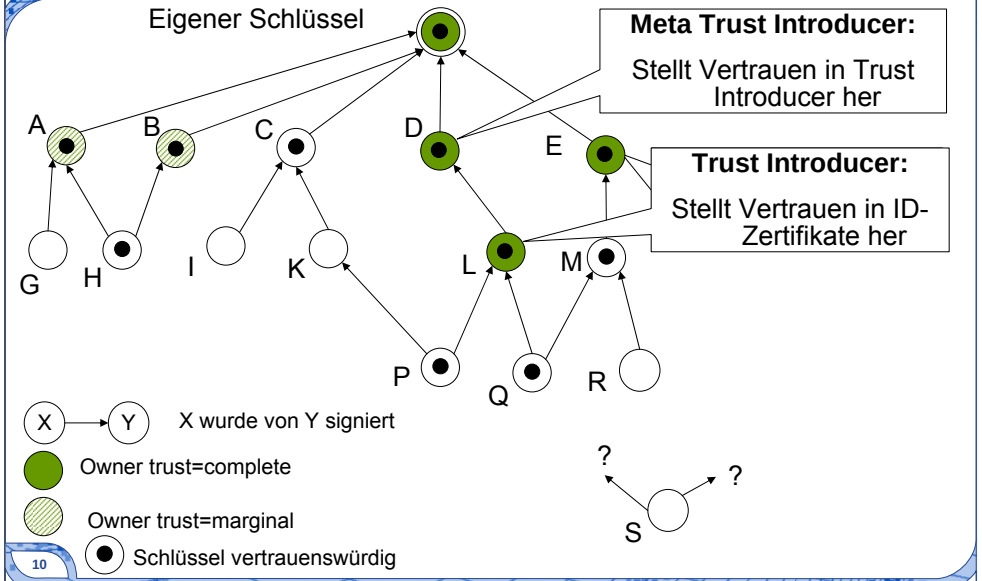
1. Benutzer legt Owner Trust fest
2. Für jede Signatur des neuen öffentlichen Schlüssels durchsuche Public Key Ring
   a) falls Signierender bekannt: Signature Trust=Owner Trust
   b) falls Signierender nicht bekannt: Signature Trust=unknown
3. Berechne Key Legitimacy
   a) lokal definierte Skepsis ermitteln, üblicherweise `COMPLETES_NEEDED` (Default: 1)
      `MARGINALS_NEEDED` (Default: 2)
   b) gewichtete Summe über Signaturen ergibt Key Legitimacy
   c) falls die Summe größer oder gleich 1 ist: Zertifikat wird als vertrauenswürdig angesehen

•In der gewichteten Summe fließt jede Signatur mit Signature Trust=Complete mit 1/COMPLETES_NEEDED ein und jede Signatur mit Signature Trust=Marginal mit 1/MARGINALS_NEEDED.

Beispiel



Beispiel



Widerruf bei PGP

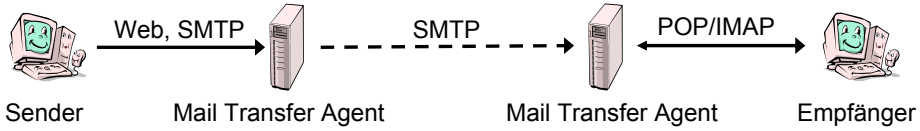
- Keine zentrale Instanz: Widerruf verteilt angelegt
  - Widerruf einer Signatur durch den Aussteller
  - Widerruf des/der Schlüssel durch Besitzer
- Probleme
  - Bei Verlust des privaten Schlüssels kann keine Widerrufsignatur mehr erstellt werden!
    - Widerruf eines öffentlichen Schlüssels durch Nicht-Besitzer
    - Besitzer kann über Selbstsignatur des Schlüssels andere Schlüsselpaare spezifizieren, die ebenfalls zum Widerruf berechtigt sind
  - Jedes Zertifikat muss vor Nutzung „aktualisiert“ werden (Schlüsselserver)

•Widerruf einer Signatur durch den Aussteller
  •Andere Signaturen weiterhin gültig
  •Widerrufsignatur analog zur initialen Signatur macht diese ungültig
  •Grund kann angegeben werden
  •Widerruf einer Signatur kann beim Ausstellen durch ein Flag ausgeschlossen werden
•Widerruf des/der Schlüssel durch Besitzer
  •Widerrufsignatur des Schlüssels von ihm selbst zertifiziert die Ungültigkeit
  •Primärer Schlüssel und Unterschlüssel getrennt widerrufbar
  •Grund kann jeweils angegeben werden

Netzsicherheit – Architekturen und Protokolle
PGP

1. PGP PKI
2. E-Mail Sicherheit

Wiederholung E-Mail



- Abholen neuer Mails: POP oder IMAP
  - Authentifizierung: Benutzernamen und Passwort im Klartext
  - Ergänzung: kryptographische Authentifizierung
    - S/Key
    - Kerberos
  - Sicherung durch SSL/TLS
- Transport vom Mails: SMTP
  - Klartext-Protokoll ohne Authentifizierung

- Vereinfachter Aufbau des Internet Mail Systems
  - Benutzerprogramm
    - Pine, Kmail, Outlook, etc.
  - Mailserver (MTA: Mail Transfer Agent)
    - Sendmail, Exim, etc.
- Weitergehende Literatur
  - POP [RFC 1939]
  - IMAP [RFC 2060 – 2063]
  - SMTP [RFC 821 – 822]

SMTP-Nachrichtenformat

- RFC 822-konforme Mail darf nur aus ASCII-Zeichen bestehen
- Notwendige Felder im Nachrichten-Kopf
  - Date: Wed, 21 Apr 2009 15:45:00 +0200
  - From: schoeller@nw.neclab.eu
  - To: vorlesung@tm.uka.de
- Optionale Felder im Nachrichten-Kopf
  - Cc und Bcc: Kopien
  - Message-ID: Zuordnung von Antworten
  - Reply-To: Antwort-Adresse
  - Subject: Betreff

- Nachricht ist zweigeteilt: Nachrichten-Kopf und Nachricht

MIME: Multipurpose Internet Mail Extensions

- Problem: Nicht-ASCII-Mails und Sprachen mit Sonderzeichen
- Lösung: Multipurpose Internet Mail Extensions
  - [RFC 2045 – 2049]
  - Definition von Nachrichtenformat u. syntaktische Regeln
- Unterstützung von Multimedia-Emails durch zwei Protokoll-Köpfe
  - Art der Nachricht
    - Content-Type: JPEG, MP3, MPEG
  - Codierung der Nachricht
    - Content-Transfer-Encoding: Base64 (B), Quoted-Printable (Q)

Beispiel MIME

Client codiert Nachricht, z.B.:

```
From: schoeller@nw.neclab.eu
To: vorlesung@tm.uka.de
Subject: Bild der Vorlesung
Mime-Version: 1.0
Content-Transfer-Encoding:
  base64
Content-Type: image/jpeg
(base64 encoded data …… )
.
```

Empfänger:

- Decodieren der Nachricht
- Übergabe der Nachricht an spezielles Programm zur Anzeige

- MIME Content-Type Konfiguration:
  - Text/Plain:
  - Image/jpeg: mozilla
  - Audio/mp3: xmms
  - Video/mpeg: xine
  - Application/ms-word: star-office

Angriffe auf SMTP

- Angriffe einfach möglich

Typischer Ablauf

```
    HELO
        220
MAIL FROM:
  schoeller@nw.neclab.eu
        250 OK
RCPT TO: vorlesung@tm.uka.de
        250 OK
DATA
…
Wie viele Studenten waren
  heute in der Vorlesung?
.
        250 OK
QUIT
```

Spoofing-Angriffe leicht möglich →

Manipulation am Inhalt leicht möglich →

Diskussion: Anforderung

Welche Anforderungen gibt es an E-Mail Sicherheit?

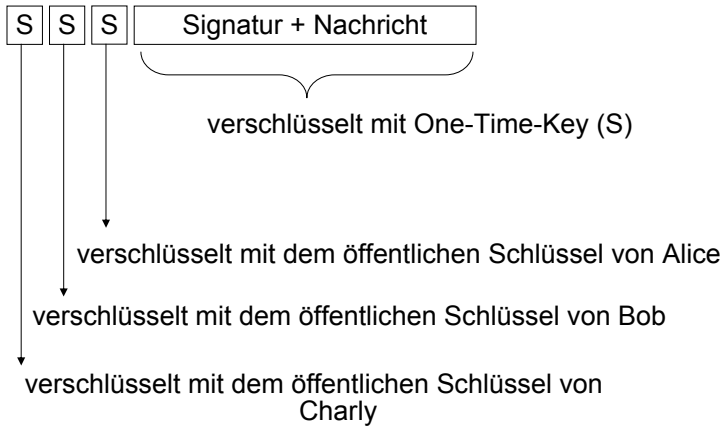
Welche Design-Ziele gibt es?

Was ist die passende Analogie?

Vertraulichkeit der Daten

- Einfacher Ansatz
  - Asymmetrisches Verfahren mit öffentlichem Schlüssel
  - (Symmetrisches Verfahren mit vor-verteiltem Geheimnis)
- Probleme
  - Hoher Aufwand
    - Separate Verschlüsselung der Nachricht für jeden Empfänger
  - Skalierungsproblem
    - Verteilung symmetrischer Schlüssel
  - Verringerungen der Sicherheit
    - Häufige Verwendung langlebiger Schlüssel

Konzept: Digitaler Umschlag



Hybrides Verschlüsselungssystem. Es wird folgendermaßen vorgegangen:

- Wahl eines zufälligen Schlüssels (One-Time-Key) durch den Absender.
- Symmetrische Verschlüsselung der Nachricht mit diesem Schlüssel
- Verschlüsselung dieses Schlüssels mit den öffentlichen Schlüsseln der Empfänger oder den gemeinsamen Geheimnissen
- Anhängen der verschlüsselten Schlüssel an die E-Mail

Vorteile:

- Entfernte Vervielfachungsdienste können wie ein normaler Empfänger behandelt werden
- Einmaliges Verschlüsseln der Nachricht
- Minimale Nutzung von asymmetrischen Verfahren

Sender-Authentifizierung, Datenintegrität

- Public Key Kryptographie
  1. Berechnen des Hashwerts der Nachricht
  2. Signieren des Hashs mit dem privatem Schlüssel
  3. Anhängen der Authentifizierungsdaten an die Nachricht
  - Einsetzbar auch bei entfernten Vervielfachungsdiensten

- Symmetrische Kryptographie
  1. CBC Restwert des Nachricht (z.B. CBC-MAC)
  2. Message Authentication Code (z.B. HMAC)
  3. Verschlüsselter Hashwert der Nachricht
  - Bei mehreren Empfänger ist die dritte Variante zu bevorzugen
  - Bei entfernten Vervielfachungsdiensten keine direkte Authentifizierung des Senders möglich

•CBC=CipherBlockChaining : Operationsmodus/Betriebsmodus für symmetrische, Blockchiffren (z.B. DES, AES, …), siehe Grundlagenkapitel

•Werden Authentifizierungsdaten, wie in der Zeichnung zu sehen, an die Nachricht angehängt, so ist die Nachricht auch für einen Empfänger zu lesen, der kein kryptographisch erweitertes System besitzt => Kompatibilität

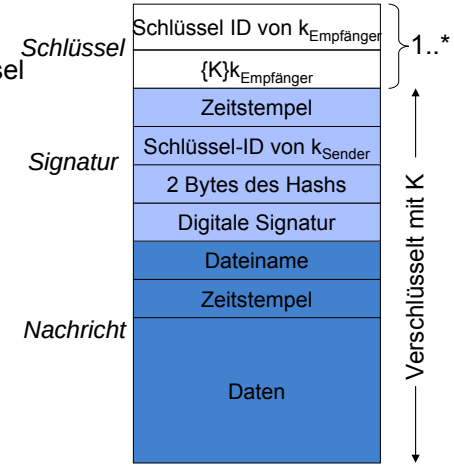
E-Mail Sicherheit durch PGP

- PGP als *hybrides Kryptosystem*
  - Dynamisch generierter symmetrischen Schlüssel
  - Verschlüsselung der Nachricht mit Sitzungsschlüssel
  - Verschlüsselung des Sitzungsschlüssels mit öffentlichem Schlüssel des Empfängers

- **Problem:** Schlüssel-Identifizierung
  - Wie erkennt der Empfänger, welcher Public Key verwendet wurde, um den Sitzungsschlüssel zu verschlüsseln?

PGP-Nachrichtenformat

- Schlüssel-Teil
  - Schlüssel-ID
  - Verschlüsselter Sitzungsschlüssel
- Signatur-Teil
  - Erstellungsdatum der Signatur
  - Schlüssel-ID des öffentlichen Schlüssels
  - 2 Bytes des unverschlüsselten Nachrichten-Hashs
  - Signatur über Zeitstempel und Daten
- Nachrichten-Teil
  - Name der PGP-Datei
  - Erstellungsdatum der Datei
  - Daten



•2 Bytes des unverschlüsselten Nachrichten-Hashs dienen der Überprüfung, ob der korrekte öffentliche Schlüssel verwendet wurde.

Schlüssel-Identifizierung

- Möglichkeiten
  - Ausprobieren: Empfänger testet jeden seiner Private Keys
  - Mitschicken: der Sender hängt den verwendeten Schlüssel bzw. das Zertifikat an
  - Schlüssel-ID: Anhängen einer eindeutigen Referenz auf den Public Key
- Bewertung
  - Ausprobieren: sehr hohen Rechenaufwand beim Empfänger
  - Mitschicken: erhöht die verbrauchte Bandbreite
  - Schlüssel-ID: Zuweisung, Speicherung und Verwaltung
    - Niederwertigsten 64 Bit vom SHA-1 Hashwert des öffentlichen Schlüssels

•Schlüssel-ID: Zuweisung muss bei Sender und Empfänger identisch sein

PGP-Datenstrukturen

- PGP-Grundbausteine: *Pakete (packets)*:
  - Paket-Bedeutung durch *Tag* identifiziert
  - Verschachtelung von Paketen
    - Z.B. ein verschlüsseltes Paket kann ein komprimiertes Paket enthalten
- Paketaufbau
  - Kopf
    - Tag
    - Länge des Pakets
  - Nutzdaten

- Beispiel: Public-Key Encrypted Session Key Packet
  - Tag: 1
  - Daten: Radix-64 codiert

Überblick: Paket-Typen

- Folgende Tags werden für Nachrichten verwendet
  - 1: Public-Key Encrypted Session Key Packet
  - 2: Signature Packet
  - 3: Symmetric-Key Encrypted Session Key Packet
  - 4: One-Pass Signature Packet
  - 8: Compressed Data Packet
  - 9: Symmetrically Encrypted Data Packet
  - 11: Literal Data Packet
  - ...
- Folgende Tags werden für Zertifikate verwendet
  - 2: Signature Packet
  - 5[7]: Secret [Sub-]Key Packet
  - 6[14]: Public [Sub-]Key Packet
  - 12: Trust Packet
  - 13: User ID Packet

•1: Enthält Session Key, mit dem die Nachricht verschlüsselt wurde. Der Session Key ist mit dem öffentlichen Schlüssel des Empfängers verschlüsselt

•2: Signatur

•3: Session Key, verschlüsselt mit einem symmetrischen Schlüssel (z.B. bei Verwendung von S2K)

•4: Enthält Informationen, die es ermöglichen, eine Nachricht in einem Durchlauf zu überprüfen wenn die Signatur hinten angehängt ist => Hash-Wert wird bereits beim Laden der Mail berechnet. Geht den signierten Daten voraus.

•8: Enthält komprimierte Daten

•9: Daten verschlüsselt mit Sitzungsschlüssel

•11: Enthält Daten, die nicht mehr weiter behandelt werden sollen (z.B. Body einer Nachricht)

•5/7: Gleiche Infos wie 6/14, aber zusätzlich zugehöriger privater Schlüssel

•6/14: enthält öffentlichen Schlüssel

•12: Enthält Informationen über Vertrauen gegenüber anderen Schlüsseln. Wird nicht exportiert!

•13: Name des Nutzers (enthält RFC 822 E-Mail Adresse)

Zusammengesetzte Datenstrukturen: Nachrichten

- Verschlüsselte Daten
  - Public-Key Encrypted Session Key Packet(s)
  - Symmetrically Encrypted Data Packet
    - z.B. Literal Data Packet
- Signierte Daten
  - Signature Packet
  - Literal Data Packet
- Signierte, komprimierte und verschlüsselte Nachricht
  - Public-Key Encrypted Session Key Packet(s)
  - Symmetrically Encrypted Data Packet
    - Compressed Data Packet
      - Signature Packet
      - Literal Data Packet

Signature Packet

- Unterschiedlichen Semantiken
  - Zertifizierungen von Schlüssel zu Name
    - Beinhalten auch, wie sicher sich der Zertifizierende bezüglich der Bindung ist
  - Signieren eines Unterschlüssels
    - Bindet ihn an den primären Schlüssel (Subkey Binding Signature)
  - Signieren eines Schlüssels direkt
    - Bindet Infos unabhängig von der Identität des Besitzers an den Schlüssel
  - Widerruf-Signaturen
    - Über primären Schlüssel: Widerruf des gesamten Zertifikates
    - Über einen Unterschlüssel: Widerruf nur dieses Schlüssels
    - Über Signaturen (=Zertifizierungen): nur Signatur des Widerrufenden wird ungültig
  - Zeitstempel-Signatur

•Signieren eines Schlüssels direkt bindet z.B. Revocation Signatur eines Subschlüssels an einen Schlüssel.

Selbstsignaturen

- Varianten
  - *Direct key self-signature*
    - Bindung von Daten an den gesamten Schlüssel, d.h. primären Schlüssel und Unterschlüssel
  - *Certification self-signature*
    - Zuordnungsdaten für Bindung der User ID an den Schlüssel
  - *Subkey self-signature*
    - Bindung des Unterschlüssels an den primären Schlüssel
- Zusätzliche Informationen zu den Selbstsignaturen
  - Anwendungen des Schlüssels: Zertifizierung, Verschlüsselung u.a.
  - Bevorzugte Algorithmen: PK, SK, Hash, Kompression
  - Schlüssel-Server-Parameter: bevorzugte Server, Flags für den Server selbst

•Direkt key self-signature: Daten beziehen sich nicht auf Benutzer-IDs oder deren Zuordnung zum Schlüssel

•Certification self-signature: Bei mehreren IDs kann es also auch mehrere Selbstsignaturen geben

Bewertung von PGP

- PGP hat folgende Vor- und Nachteile
  - ☺ Geringer Startaufwand
  - ☺ Dezentrale Sicherheit (Vertrauensbildung über mehrere Signaturen)
  - ☺ Benutzer hat direkten Einfluss auf die Vertrauensbildung, daher aber auch hohe Verantwortung
  - ☺ Robuster bei Kompromittierung eines Schlüssels
  - ☹ Keine einheitliche Sicherheitsrichtlinie (jeder signiert nach eigenem Ermessen)
  - ☹ Kettenbildung zwar möglich, Pfadfindung aber nicht trivial
  - ☹ großen Anzahl von Zertifikaten(=Signaturen), da dezentral
- Größte Gefahren: alle nicht PGP-spezifisch
  - Man-in-the-Middle
  - Passwort Guessing
  - Trojaner zum Upload des Private Key Rings

Praxisteil: GnuPrivacyGuard

- Open Source Implementierung
  - OpenPGP compliant
  - etliche Optionen, um abwärtskompatibel zu sein
- Dokumentation: http://www.gnupg.org/documentation/

- Zertifikat-Management:
  - Erstellen eines Zertifikates/Schlüsselringes:
    ```
    gpg --gen-key
    ```
  - Exportieren eines Zertifikates (ASCII-codiert)
    ```
    gpg --armor --output <Datei> --export <ID>
    ```
  - Ausgabe des Fingerprints
    ```
    gpg --fingerprint <ID>
    ```
  - Widerrufen des Zertifikates
    ```
    gpg --gen-revoke
    ```


•Abwärstkompatibilität meist mit eingeschränkter Funktionalität verbunden

GnuPG: Zertifikatsmangement

Verwalten von fremden Zertifikaten:

- Importieren eines Zertifikates
  ```
  gpg --import <Datei>
  gpg --recv-keys <ID>
  ```
- Auflisten von Zertifikaten
  ```
  gpg --list-keys [<ID>]
  gpg --list-sigs [<ID>]
  gpg --check-sigs [<ID>]
  ```
- Manipulieren eines Zertifikates:
  ```
  gpg --edit-key <ID>
  ```
  - Signieren: `sign`
  - Erstellen einer Vertrauenssignatur: `tsign` (ab v1.3.1)
  - Einstellen des Vertrauenslevels: `trust`
  - Widerrufen einer Signatur: `revsig`
  - Viele andere Operationen (siehe Manpage der jeweiligen Version)

GnuPG: Dateioperationen

- Signieren einer Datei:

```
gpg --sign <Datei>
```

- Verifizieren einer Datei:

```
gpg --verify <Datei mit Signatur>
```

- Verschlüsselung einer Datei:

```
gpg --encrypt --recipient <ID> <Datei>
```

- Entschlüsseln einer Datei:

```
gpg --decrypt <Datei>
```

- Kombinationen:

```
gpg --encrypt --recipient <ID> --sign
<Datei>
gpg --decrypt --verify <Datei>
```


Literatur

- C. Kaufmann, R. Perlman, M. Speciner; Network Security – Private Communication in a public world; Prentice Hall; 2003
- R. Oppliger: Secure Messaging with PGP and S/MIME; Artech House, 2001
- J. Callas, L. Donnerhacke, H. Finney, R. Thayer; RFC 2440: OpenPGP Message Format, 1998
- W. Stallings; Cryptography and Network Security, Prentice Hall, 2006