

# Betriebssysteme und Sicherheit

## – Sicherheit

Signaturen, Zertifikate, Sichere E-Mail

# Frage

- Public-Key Verschlüsselung stellt Vertraulichkeit sicher
- Kann man auch Integrität und Authentizität mit Public-Key Kryptographie sicherstellen?

# Funktionen

Verschlüsselung:

Encrypt(m, pubKey)

Decrypt(c, privKey)

Signatur:

Sign(m, )

Verify(s, )

# Signatur

- Eine Signatur kann nur mit dem privaten Schlüssel erstellt werden
- Eine Signatur stellt sicher, dass **diese** Nachricht vom privaten Schlüsselhalter kommt (Authentizität und Integrität)
- Eine Signatur kann mit dem öffentlichen Schlüssel überprüft werden

# RSA Signaturen

- Bemerkung: In RSA sind Nachrichten- und Chiffre-Raum gleich

Sign = Decrypt

Verify = Encrypt

# El-Gamal

Privater Schlüssel

$x$

Öffentlicher Schlüssel

$g^x$

# El-Gamal

Signatur von  $m$

Wähle zufälliges  $r$

$$c = g^r, d = (m - xc)r^{-1} \pmod{p-1}$$

Verifikation

$$(g^x)^c c^d \stackrel{?}{=} g^m$$

$$(g^x)^c c^d = g^{xc} (g^r)^{(m-xc)r^{-1}} = g^{xc} g^{m-xc} = g^m$$

# Sicherheit

## Fälsche Signatur

- Wähle  $c$ , dann  $d = \log_c g^{m-xc}$
- Wähle  $d$ , dann löse

$$(g^{xd'})^c = g^{md'}$$

# Beispiel

$$p = 23, g = 2$$

$$x = 3$$

$$m = 4$$

$$r = 5$$

$$\Rightarrow r^{-1} = 9$$

$$\Rightarrow c = g^r = 2^5 = 9$$

$$\begin{aligned}\Rightarrow d &= (m - x \cdot c) \cdot r^{-1} = \\ &= (4 - 3 \cdot 9) \cdot 9 = 13\end{aligned}$$

$$(g^x)^c c^d = (2^3)^9 9^{13} = 8^9 \cdot 12 = 16 = 2^4 = g^m$$

Der Sender muss zur Überprüfung immer beides übertragen: Nachricht und Signaturwert!

# Problem

- Wie verteile ich den öffentlichen Schlüssel sicher gegen aktive Angreifer ?

# Zertifikat

Ein Zertifikat besteht aus

- einer Identität
- einem öffentlichen Schlüssel
- der Identität des Signierers
- einer Signatur

$A, \text{PubKey}_A, CA, S_{CA}(A, \text{PubKey}_A, CA)$

# Verifikation eines Zertifikats

- Stelle Identität fest
- Überprüfe Identität im Zertifikat
- Finde öffentlicher Schlüssel des Signierers
- Überprüfe Signatur
- Überprüfe Zeitstempel
- Überprüfe CRL
- Verwende öffentlicher Schlüssel

# Schlüsselverteilungsproblem

Signierer = Zertifikatsstelle

(certificate authority)

Man muss nur die öffentlichen Schlüssel der Zertifikatsstelle sicher verteilen

Selbstsignierte Zertifikate

z.B. bei OS Installation

# Zertifikatsstandard

X509v3

1. Versionsnummer
2. Seriennummer
3. Signaturalgorithmusidentifizier
4. Ausstellername (z.B. CA)
5. Lebensdauer
6. Subjektname (z.B. A)
7. Öffentlicher Schlüssel des Subjekts
8. Optionale Felder
9. Signatur des Ausstellers

# Rückrufproblem

- Eine Entität kann sich mit einem Zertifikat authentifizieren, auch wenn das Zertifikat fehlerhaft ausgestellt wurde
- Wie kann man ein einmal ausgestelltes Zertifikat zurückrufen
  - Ablaufdatum
  - Rückruflisten (Certificate Revocation List)

# Ablaufdatum

- Das Ablaufdatum wird beim Akzeptieren überprüft
- Erneuerung durch überlappende Zeiträume
- Wie lange soll man den Zeitraum wählen ?

# Rückruflisten

- Beim Akzeptieren wird die Rückrufliste (CRL) bei der Zertifikatsstelle abgerufen
- Die (CRL) enthält alle Seriennummern der zurückgerufenen Zertifikate und ist von der Zertifikatsstelle signiert
- Die Seriennummer des Zertifikats wird mit der Liste verglichen
- Erfordert on-line Zugriff auf Zertifikatsstelle

# Begriffe

- Public-Key Infrastructure (PKI)  
Bereitstellung von (Root-)Zertifikaten und Zertifikatsstellen
- Trust Management  
Verwaltung von vertrauenswürdigen Zertifikaten

# S/MIME

- Kann MIME e-mail Anhänge
  - Verschlüsseln und/oder
  - Signieren
- Wird von vielen Clients direkt unterstützt (z.B. MS Outlook)
- Verwendet Zertifikate (x509v3)
  - Jeder Client hat Liste von Trusted Root CAs
  - Zertifikat kann mit versandt werden

# PGP

- Kann Nachrichten signieren und/oder verschlüsseln mittels Public-Key Cryptography
- Benötigt eigene Software
  - PGP (kommerziell), GnuPG (open source)
- Codiert verschlüsselte Nachricht als Text (Base64)
- Verwendet keine Zertifikate

# PGP Schlüsselmanagement

- Es gibt keine zentralen Cas
- Stattdessen agiert jeder Teilnehmer als „CA“
  - Er signiert Schlüssel anderer Teilnehmer
- Dies bildet ein „Web of Trust“
  - Ein Benutzer kann anderen von ihm signierten Schlüsseln trauen weitere Schlüssel zu signieren
  - Kettenbildung
- Es gibt Vertrauensstufen

# PGP Trust Model

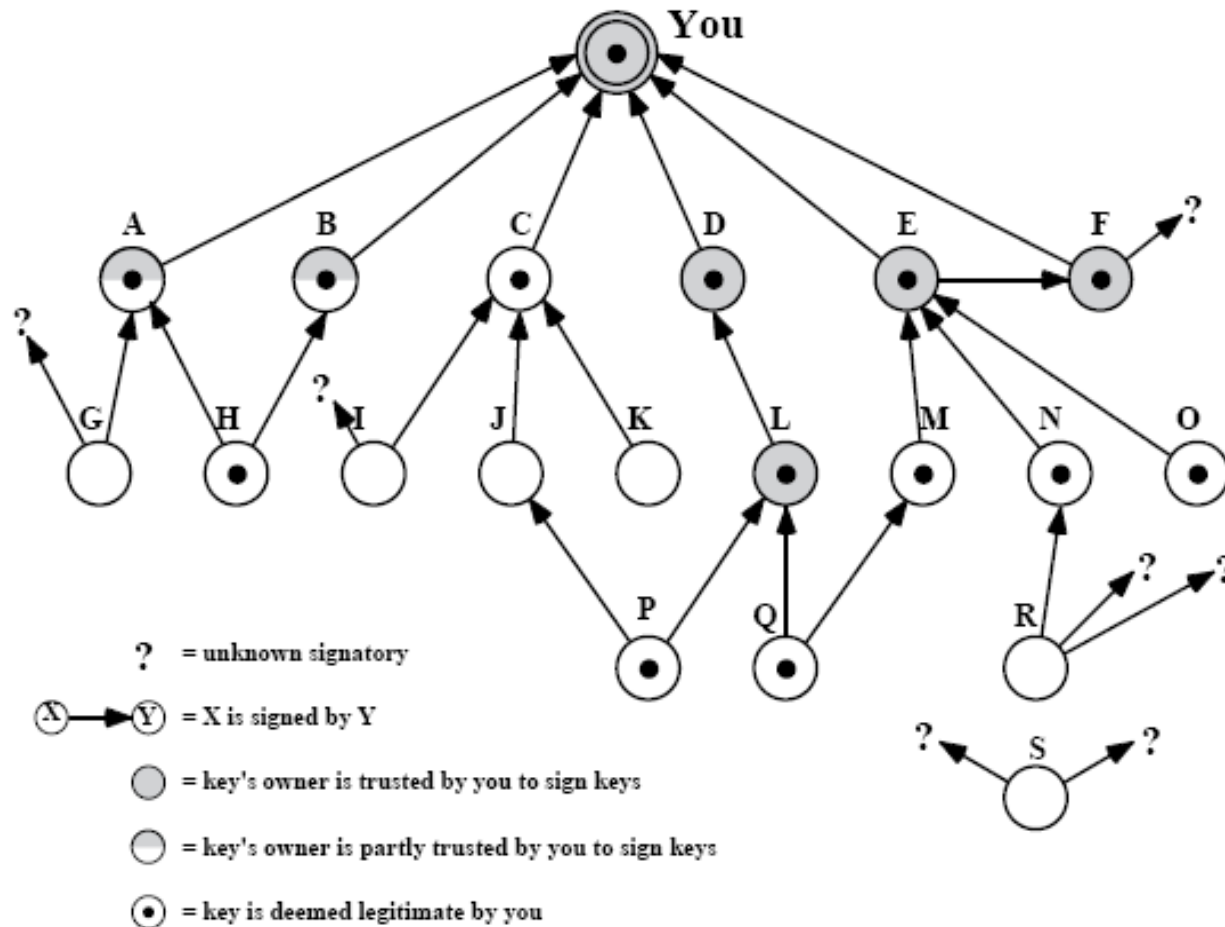


Figure 5.7 PGP Trust Model Example

# Problem

- Bei Public-Key Kryptographie erzeugt eine Partei ein Schlüsselpaar
- Der öffentliche Schlüssel muss nun **sicher** übertragen werden (vgl. PKI)

# Identity-Based Encryption (IBE)

- Beim Verschlüsseln kann man eine beliebige Zeichenkette verwenden,
  - Z.B. e-mail Adresse
- Es gibt eine zentrale Stelle KGC (vgl. CA)
- Der Entschlüsseler fragt bei der KGC nach seinem privaten Schlüssel



# IBE

