ICT

§8 Security

Harald.Schil.ly
harald@schil.ly

Kryptographie Symmetrisch

Eine symmetrische Verschlüsselung ist ein Verfahren, wo sowohl Sender als auch Empfänger denselben Schlüssel verwenden. Dabei ist es wichtig, dass die verschlüsselten Daten möglichst "gleichmäßig" sind, um Rückschlüsse zu vermeiden. Prinzipieller Nachteil: der Schlüssel muss ausgetauscht werden...

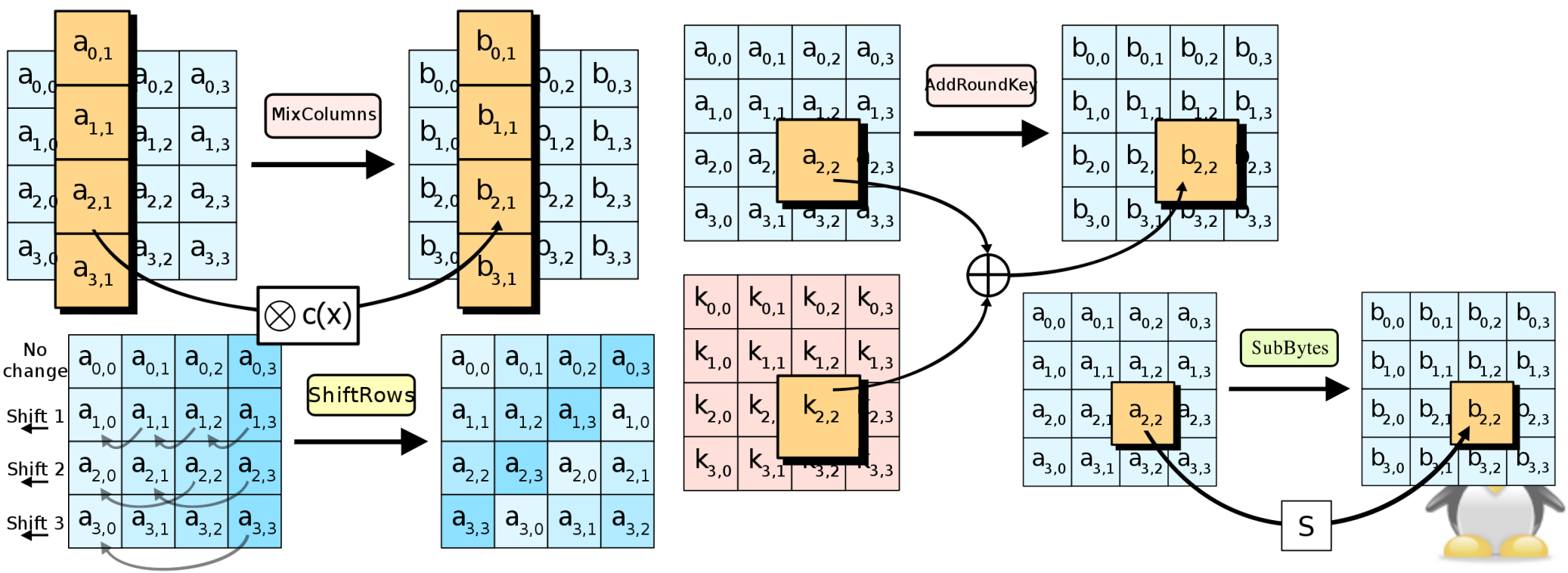
Schlechtes Beispiel: Caesar-Kodierung. Jedem Buchstaben im Alphabet wird ein anderer zugewiesen. Aus den Buchstaben-häufigkeiten lassen sich Rückschlüsse auf die tatsächlichen Buch-staben ziehen. (Verbeserung: Vigenère Verschlüsselung)

Modern: DES, Triple DES, AES, IDEA, Camellia (teil von SSL/TLS)

Linux: man gpg (**`gpg -c file`** und `gpg file`)

Kryptographie Symmetrisch/AES

AES = Advanced Encryption Standard. Weltweit in Verwendung, von der NSA zertifiziert, ...
Funktionsweise: Initialen Schlüssel wählen (128-bit oder mehr), Daten in 128-bit Blöcke einteilen und jeweils eine Liste von Operationen auf sie anwenden, wobei sich der interne Status ständig ändert und "subkeys" generiert werden. (subkey+state werden kombiniert).



Kryptographie Asymmetrisch (1)

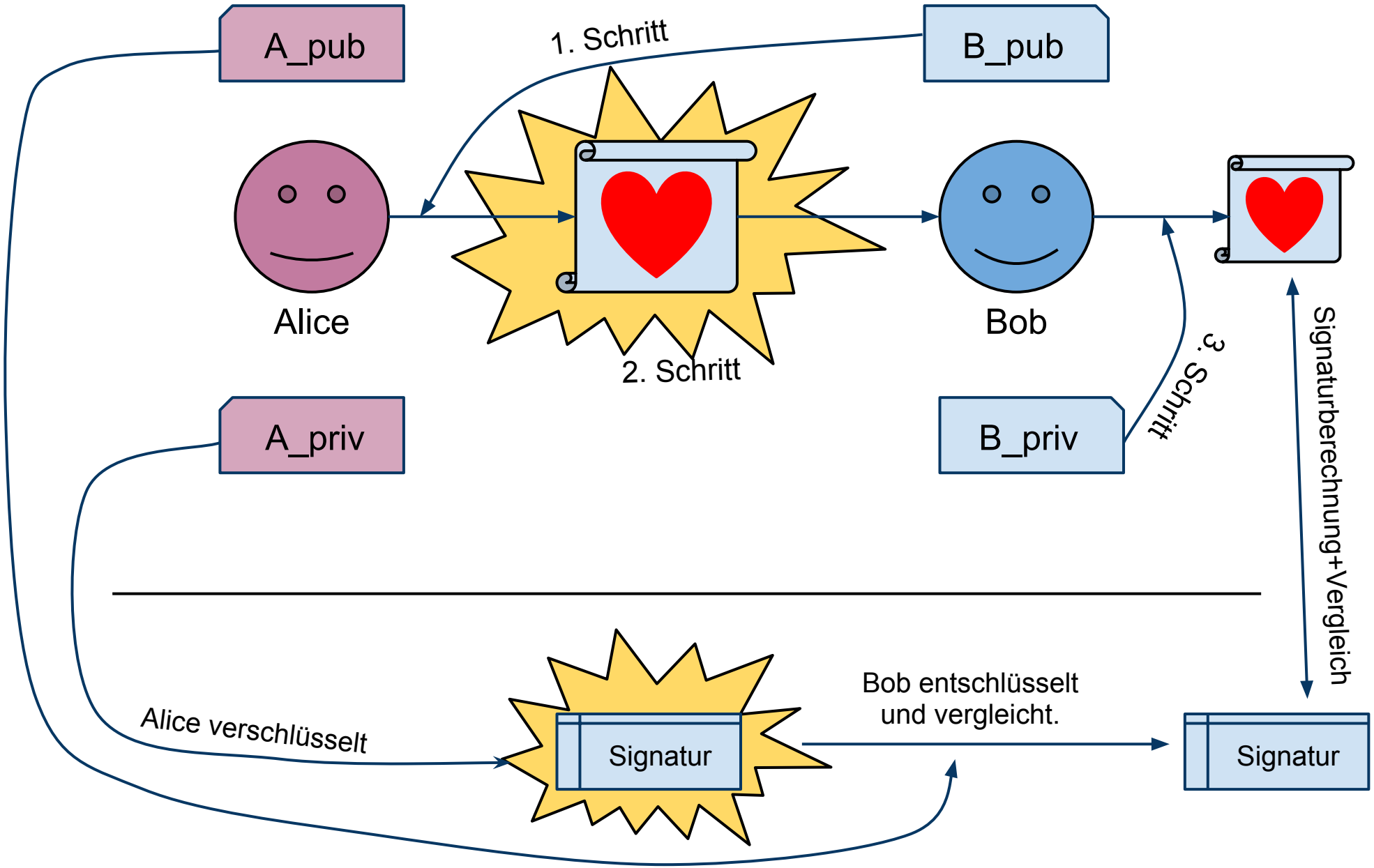
Asymmetrische Verschlüsselung heißt, dass es zwei Schlüssel gibt. Einen öffentlichen public-key und einen geheimen private-key.

Grundsätzlich ist es dabei so, dass zuerst problemlos der public-key vom Empfänger zum Sender übertragen wird (kann öffentlich passieren).
Nachrichten die mit diesem public-key verschlüsselt wurden können nur vom Emfänger, der den private-key hat, entschlüsselt werden. Also insbesondere reicht der public-key nicht, um eine damit verschlüsselte Nachricht wieder lesbar zu machen.
Das hat auch den Vorteil, dass man jedem denselben Schlüssel schicken kann um verschlüsselte Nachrichten zu empfangen. (auf Webseite stellen ...)
Außerdem gibt es "digitale Signaturen". Das sind Zertifikate für eine Nachricht, um die Echtheit zu bestätigen.

Grundzutat für jedes derartige Verfahren ist eine Berechnung, die in eine Richtung leicht geht, aber in eine andere nur schwer: Faktorisierung, diskreter Logarithmus, ... Beispiele: PGP, GPG, ElGamal

Kryptographie Asymmetrisch (2)

A verschlüsselt mit B_pub, B entschlüsselt mit B_priv



Signatur: A verschlüsselt Hashcode mit A_priv, Bob entschlüsselt mit A_pub.

RSA/ElGamal Kryptographie

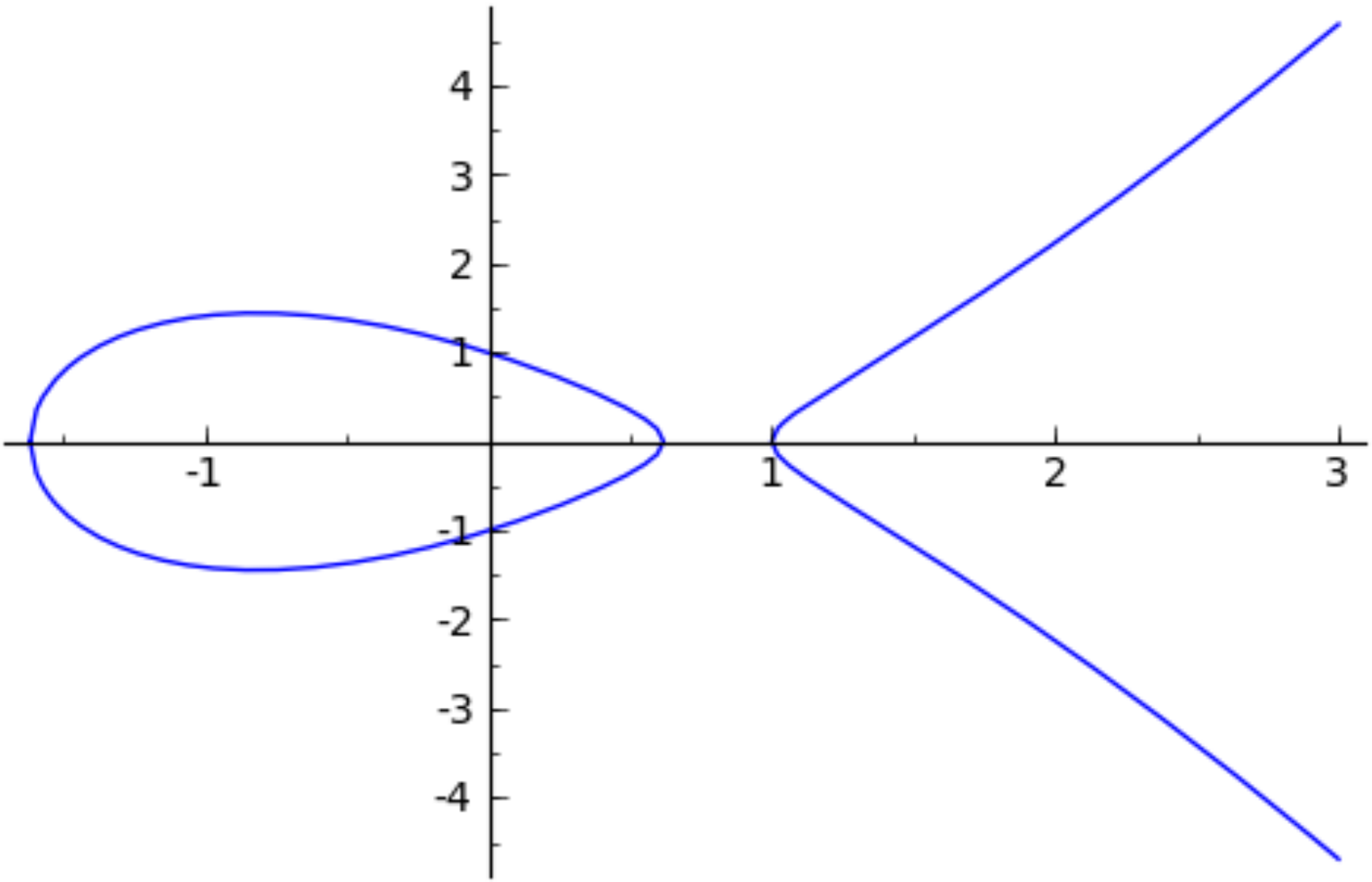
RSA = von Rivest, Shamir and Adleman, public-private-key Verfahren, basierend auf der Schwierigkeit, zweie große Primzahlen zu faktorisieren - Multiplizieren ist aber einfach.

ElGamal basiert auf Elliptischen Kurven. -> ECC: Basiert auf die Schwierigkeit, den diskreten Logarithmus in der Gruppe der Punkte einer ell. Kurve über einen endlichen Körper zu berechnet.

... siehe Beispielberechnungen...

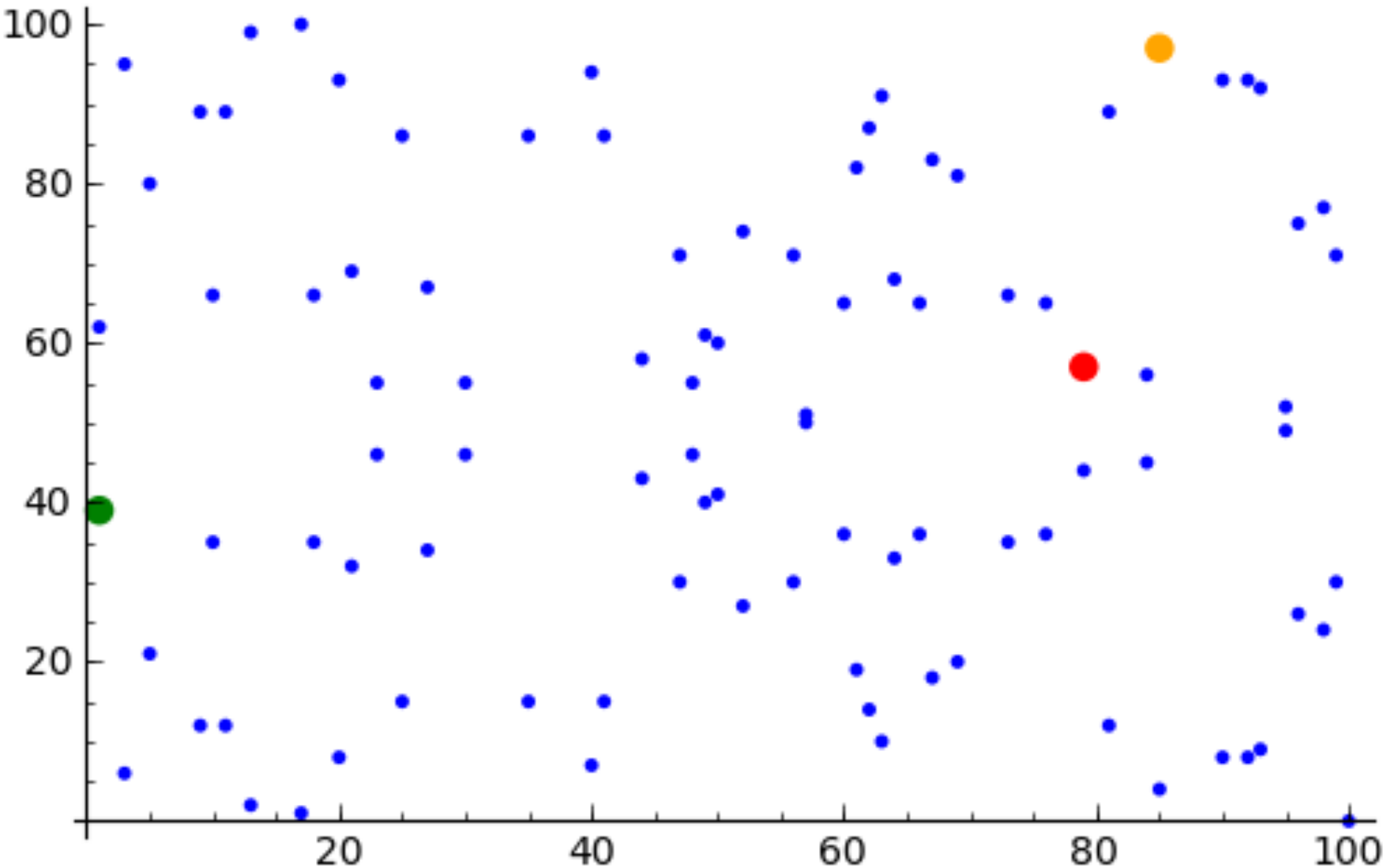
ElGamal Kryptographie

Elliptische Kurve $y^2 = x^3 - 2x + 1$ über Reele Zahlen



ElGamal Kryptographie (1)

Ell. Kurve $y^2 = x^3 - 2x + 1$ über endlichen Körper $\mathbf{F}_{101}$



RSA/ElGamal Kryptographie (2)

Verwendung in Linux mittels gpg:

`gpg --gen-key` ... Key generieren, mit "passphrase"

- "passphrase" ist ein Passwort für den Schlüssel, zusätzlicher Schutz, kann weggelassen werden (z.B. passwortloses login mit ssh)

`gpg --armor --export my@email.com > mypublickey` ... export
`gpg --import keyfile` ... key importieren
`gpg -a -se -r my@email.com file` ... file signieren und für "my@email.com" verschlüsseln -> das erzeugt file.asc bzw. ohne "-a" file.gpg
`gpg -d file.[asc|gpg]` ... dekodieren

Bemerkung:

- Die Option "`--armor`" bzw. "`-a`" kann weggelassen werden. Es erzeugt per e-mail verschickbaren ASCII Text, 7-bit. ansonsten Binärdatei.

RSA/ElGamal Kryptographie (3)

Nachricht in Kommandozeile verschlüsseln&signieren:

```
echo "abc" | gpg -ase -r e@mail1 e@mail2 > msg.txt
```

Entschlüsseln: `cat msg.txt | gpg -d`

Signieren: `echo "abc" | gpg -clearsign`

Webdatenbank:

```
gpg --search-keys <name|e@mail|ID>
gpg --send-keys <ID>
```

Keys aus der Webdatenbank aktualisieren:

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

Key editieren (passphrase ändern, ...)

```
gpg --edit-key <e@mail|ID>
```

```
gpg --fingerprint <e@mail|ID>
```

RSA/ElGamal Kryptographie (4)

Es ist auch möglich unverschlüsselte Nachrichten zu signieren.
Das funktioniert so:

1. Hashcode für die Nachricht berechnen.
2. Hashcode mit dem privaten (geheimen) Schlüssel verschlüsseln.
3. Nachricht wird zusammen mit verschlüsseltem Hashcode (Signatur) im Klartext übertragen.
4. Empfänger kann mit dem öffentlichen Schlüssel den Hashcode entschlüsseln und mit dem selbst berechneten Hashcode vergleichen.

Das wird unter anderem auch für deb-Pakete verwendet, um sicherzustellen, dass der Inhalt nicht von "Dritten" manipuliert wurde.
Hierfür gibt es den Befehl `apt-key` der öffentliche Schlüssel zum Signieren von Paketen importiert.
Verbleibende Schwachstelle: public/private key Paar wird ebenfalls manipuliert.

Web-Of-Trust

Verbleibende Schwachstellen:

- public/private key Paar wird manipuliert.
- Jemand gibt sich für jemanden anderen aus.

Es gibt daher die Möglichkeit, gegenseitig die public keys zu signieren. Das erzeugt ein Netzwerk für das gegenseitige Vertrauen:

1. ownertrust:
   `gpg --edit-key <e@mail> => trust => select level`
2. sign a key:
   `gpg --sign-key <e@mail>`
3. `gpg --update-trustdb`

```
gpg: 3 marginal(s) needed, 1 complete(s) needed,
gpg: depth: 0  valid:   2  signed:  76  trust: 0-, 0q, 0n, 0m, 0f, 2
gpg: depth: 1  valid:  76  signed:  85  trust: 0-, 1q, 0n, 64m, 11f, 0
gpg: depth: 2  valid:   4  signed:  34  trust: 0-, 3q, 0n, 1m, 0f, 0u
gpg: next trustdb check due at 2012-09-01
```

used by retroshare.sourceforge.net
for a secure communication platform

Kryptographie im WWW

- Die über Netzwerkverbindungen ausgetauschten Daten können verschlüsselt werden -> **`ssh; rsync --rsh=ssh ...`**
- Webserver können verschlüsselte Verbindungen ermöglichen. Dafür muss mittels **`openssl`** ein Zertifikat erzeugt werden, welches dann für verschlüsselte **`https`** Verbindungen benützt werden kann.
  - Diese Zertifikate werden üblicherweise auch noch von einem Dritten zertifiziert, damit die Identität des Webseitenbetreibers nachvollziehbar ist.
  - Die Verbindung an sich erfolgt in zwei Schritten (TLS):
    1. session-key mittels public-key Verfahren austauschen
    2. daten mittels session-key symmetrisch verschlüsseln
- Emails (und jegliche andere Daten) können mittels **`gpg`** verschlüsselt werden.

Netzwerk Security

Angriffstypen

- *Unauthorized access*: sehr einfach und üblich, jemand der keinen Zugriff hat will trotzdem rein -> Authorisierung, SSO, ...
- *Ausnützen von Schwachstellen in Programmen*: Software schnell und häufig aktualisieren, am laufenden bleiben, ...
- *Denial of service (DoS)*: Viele besonders gestaltete Netzwerkpakete werden in großer Menge verschickt. Der Dienst funktioniert nicht mehr oder macht Fehler. (smurf)
- *Spoofing*: Ein Angreifer macht einen anderen (gültigen) Benutzer nach, indem er die Absenderadresse fälscht oder dessen Verhalten kopiert -> Authentifizierung, ...
- *Eavesdropping*: Ein Rechner im Netzwerk hört die Kommunikation zwischen anderen Rechnern ab. z.B. lassen sich Passwörter herauslesen -> Verschlüsselung, SSL, ...

Copyright & Lizenz