mobile business

Lecture 04

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Information & Communication Security (SS 2020)

Cryptography II

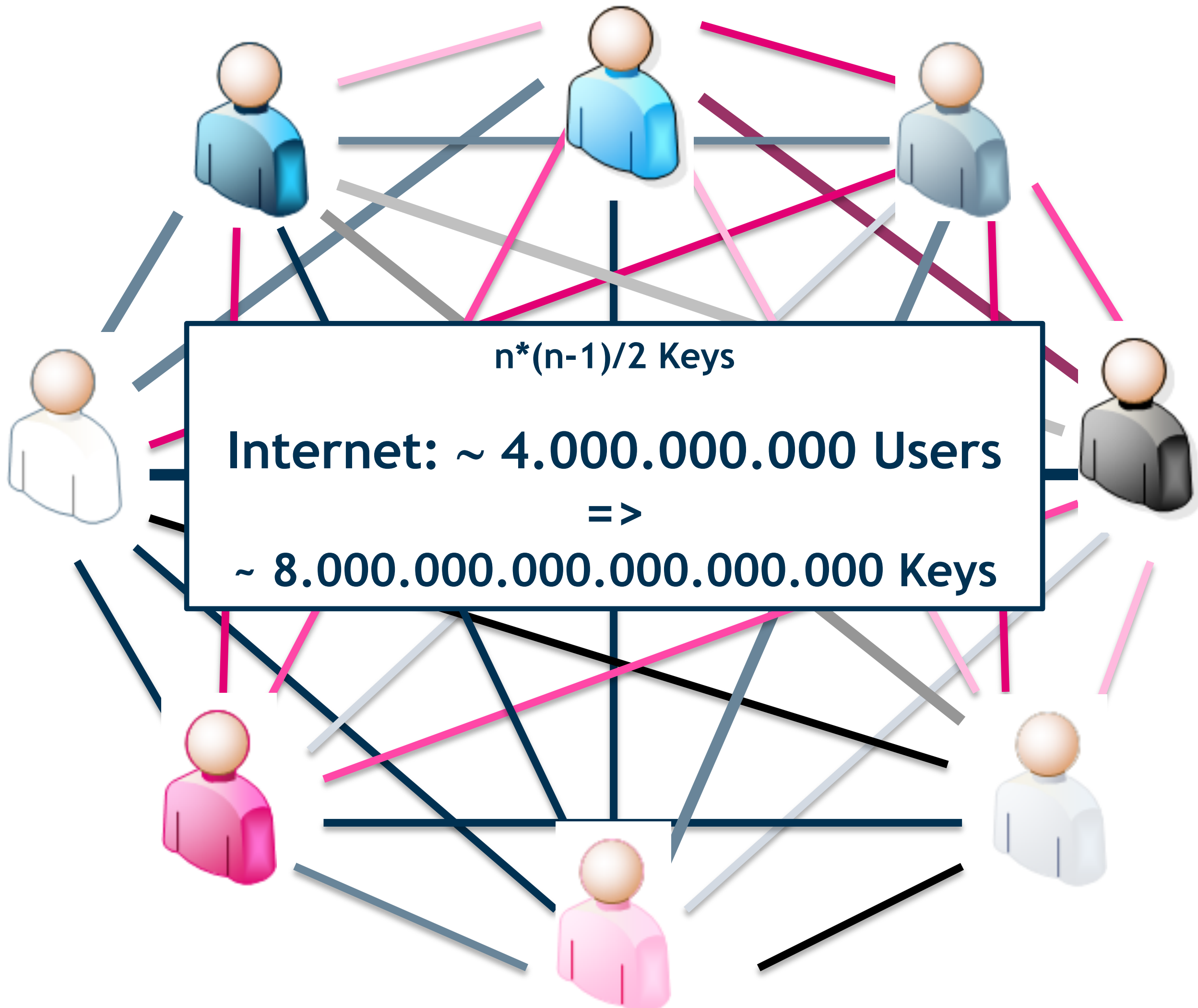
Prof. Dr. Kai Rannenberg

Chair of Mobile Business & Multilateral Security
Goethe-University Frankfurt a. M.

Agenda

- Introduction
- Symmetric Key Cryptography
- Public Key Cryptography
  - General Process
  - Algorithms
  - Hybrid Systems
  - Key Management
  - Example: PGP
- Outlook and Post-Quantum Cryptography

Disadvantage: Key Exchange



n*(n-1)/2 Keys

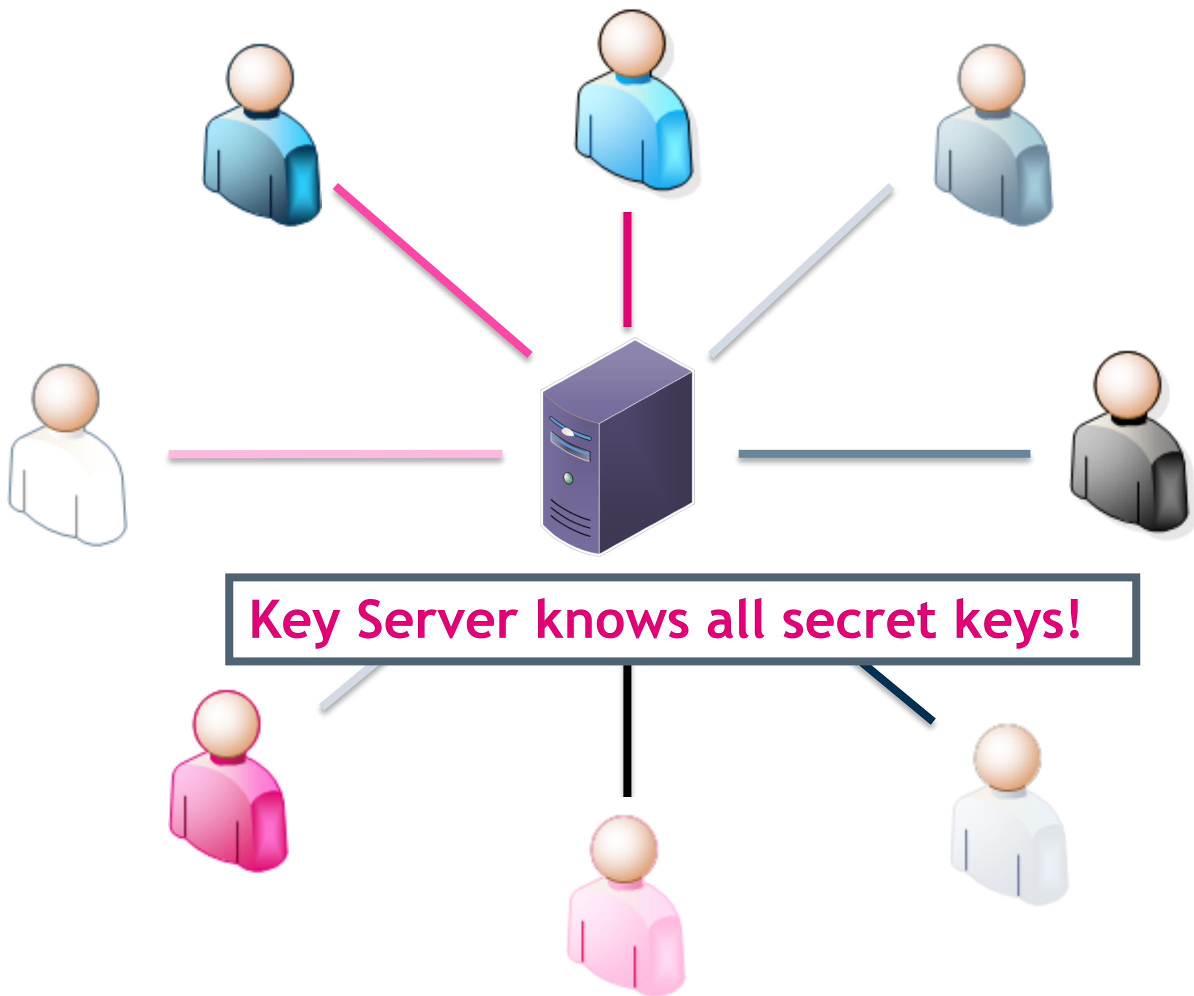
Internet: ~ 4.000.000.000 Users

=>

~ 8.000.000.000.000.000.000 Keys

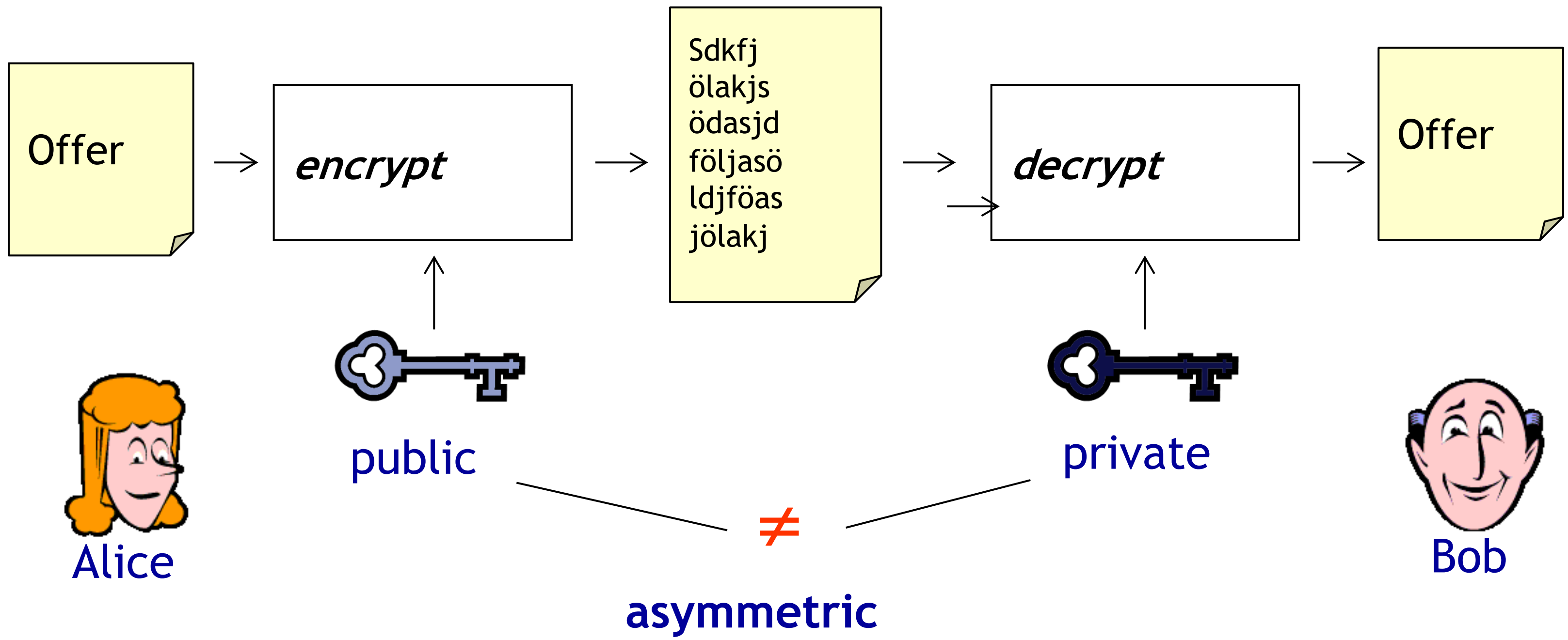
[adopted from J. Buchmann: Lecture Public Key Infrastrukturen, FG Theoretische Informatik, TU-Darmstadt]

A Possible Solution



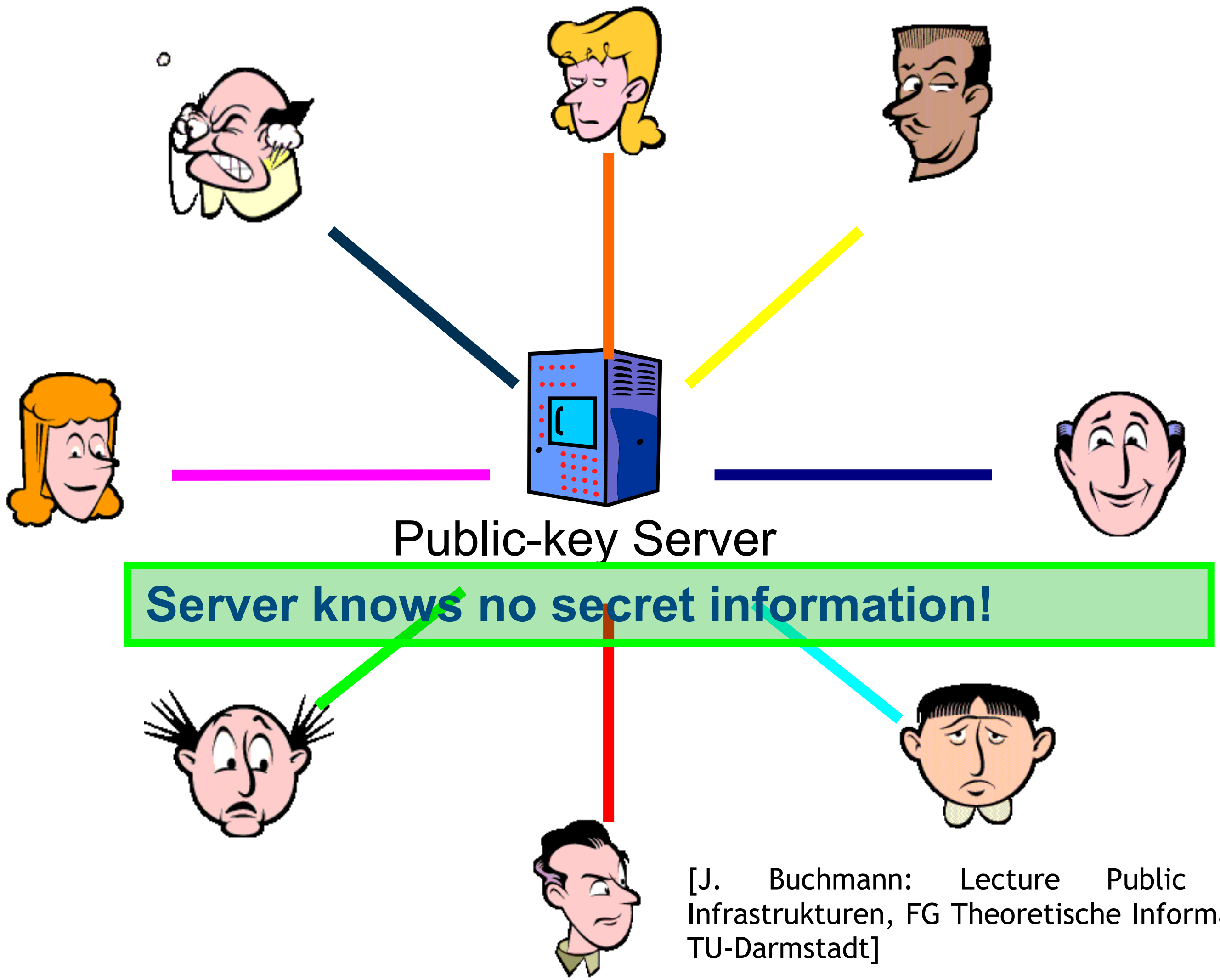
[J. Buchmann: Lecture Public Key Infrastrukturen, FG Theoretische Informatik, TU-Darmstadt]

Public Key Encryption



[J. Buchmann: Lecture Public Key Infrastrukturen, FG Theoretische Informatik, TU-Darmstadt]

Key Exchange Problem Solved!



[J. Buchmann: Lecture Public Key Infrastrukturen, FG Theoretische Informatik, TU-Darmstadt]

Agenda

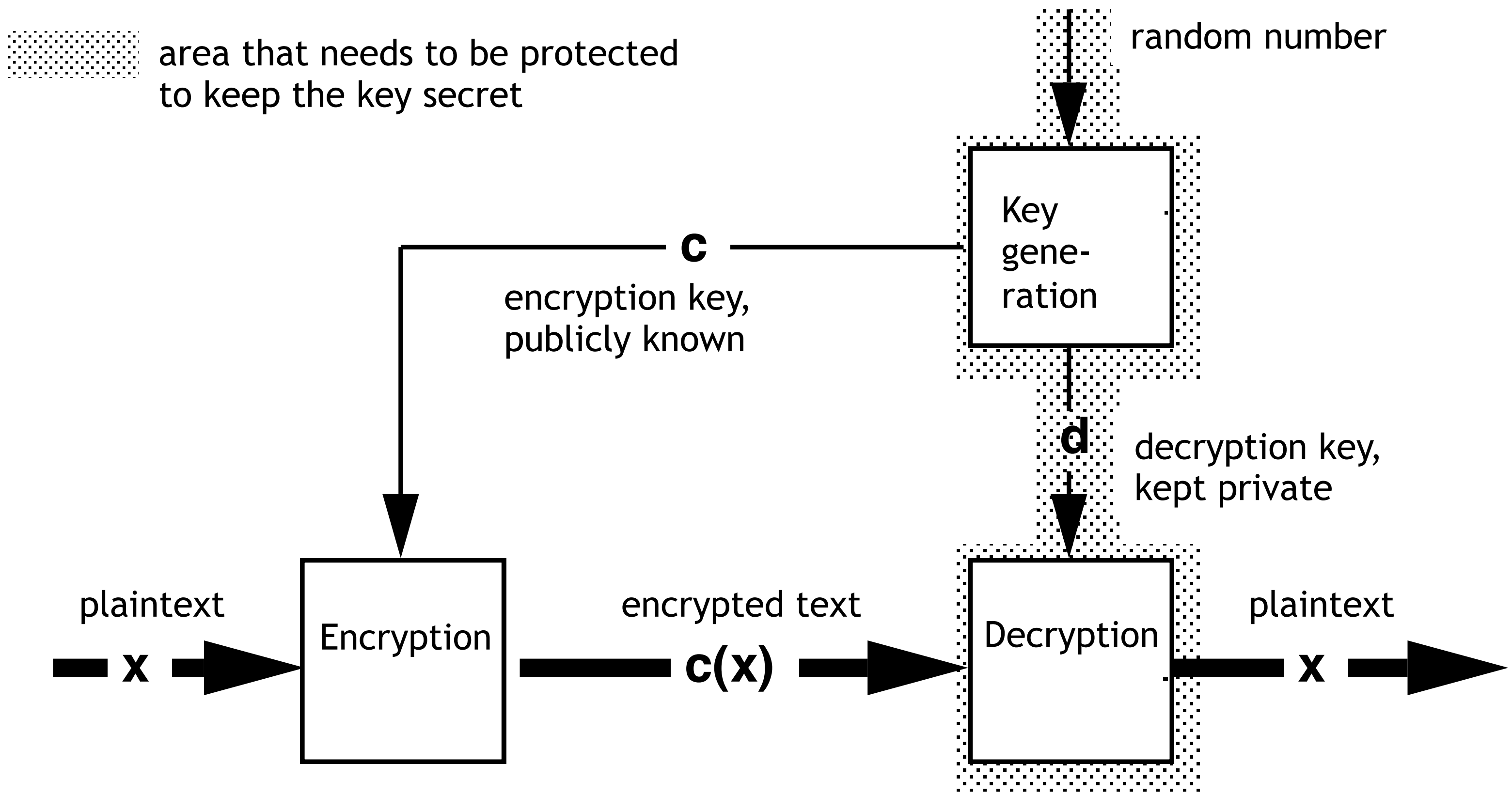
- Introduction
- Symmetric Key Cryptography
- Public key cryptography
  - General process
  - Algorithms
  - Hybrid systems
  - Key management
  - Example: PGP

Public Key Cryptography Asymmetric Encryption

- Public key systems are based on asymmetric encryption.
- Use of 'corresponding' key pairs instead of one key:
  - **Public key** is **solely** for encryption.
  - Encrypted text can only be decrypted with the corresponding **private (undisclosed) key**.
- Deriving the private key from the public key is hard (practically impossible).
- The public key can be distributed freely, even via insecure ways (e.g. directory *(public key* crypto system)).

- Messages are encrypted via the public key of the addressee.
- Only the addressee holds the private key for decoding (and has to manage the relation between the private and the public key).

Asymmetric Encryption General Process



box with slot, access to messages only with a key

[based on Federrath and Pfitzmann 1997]

Agenda

- Introduction
- Symmetric Key Cryptography
- Public key cryptography
  - General process
  - Algorithms
  - Hybrid systems
  - Key management
  - Example: PGP

Asymmetric Encryption: Examples

- RSA
  - Rivest, Shamir, Adleman, 1978
  - Based on the assumption that the factorization of the product of two (big) prime numbers (p*q) is "difficult" (product is the public key)
  - Key lengths often 1024 bit; recommended 2048 or 4096 bit
- Diffie-Hellman
  - Diffie, Hellman, 1976
  - First patented algorithm with public keys
  - Allows the exchange of a secret key
  - Based on the "difficulty" of calculating discrete logarithms in a finite field

Agenda

- Introduction
- Symmetric Key Cryptography
- Public key cryptography
  - General process
  - Algorithms
    - RSA
  - Hybrid systems
  - Key management
  - Example: PGP

RSA Encryption

- To encrypt a message *M*, using a public key (*e*,*n*), proceed as follows (*e* and *n* are a pair of positive integers):
  - First represent the message as an integer between 0 and $n-1$ (break long messages into a series of blocks, and represent each block as such an integer).
  - Then encrypt the message by raising it to the e^{th} power modulo *n*.
  - The result (the ciphertext *C*) is the remainder of M^e divided by *n*.
  - The encryption key is thus the pair of positive integers (*e*,*n*).

[RSA78]

mobile business

RSA Decryption

- To decrypt the ciphertext, raise it to another power ***d***, again modulo ***n***.
- The decryption key is the pair of positive integers (***d,n***).
- Each user makes his encryption key public, and keeps the corresponding decryption key private.

[RSA78]

RSA Encryption/Decryption Summary

- $C \equiv E(M) \equiv M^e \pmod{n}$,
  for a message M

- $M \equiv D(C) \equiv C^d \pmod{n}$,
  for a ciphertext C

[RSA78]

Choosing the Keys (I)

- You first compute ***n*** as the product of two chosen primes ***p*** and ***q***.
- $n=p*q$
- These primes are very large “random” primes.
- Although you will make ***n*** public, the factors ***p*** and ***q*** will be effectively hidden from everyone else due to the enormous difficulty of factoring ***n***.
- This also hides the way, how ***d*** can be derived from ***e***.

[RSA78]

Choosing the Keys (II)

- You then choose an integer ***d*** to be a large, random integer which is relatively prime to ***(p-1)*(q-1)***.

- That is, check that ***d*** satisfies:
  - The greatest common divisor of `d` and `(p-1)*(q-1)` is `1`.
  - `gcd(d,(p-1)*(q-1))=1`

[RSA78]

Choosing the Keys (III)

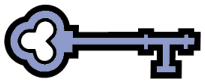
- The integer ***e*** is finally computed from ***p,q***, and ***d*** to be the “multiplicative inverse” of ***d***, modulo (***p-1***)*******(***q-1***).

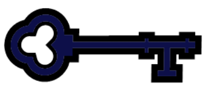
- Thus we have

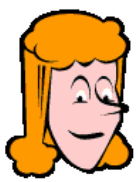
  $$e*d \equiv 1 \pmod{(p-1)*(q-1)}.$$

[RSA78]

Simplified Example (I)


Public
(e,n)


Private
(d,n)

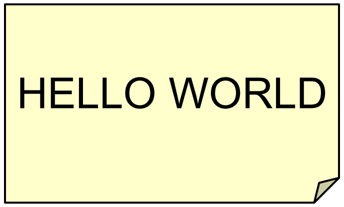

Alice

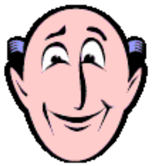
- Let p=7 and q=11.
- Then n=77.
- Alice chooses d=53, so e=17.
- gcd(d,(p-1)*(q-1)) = gcd(53,(7-1)*(11-1)) = gcd(53,60) = 1
- e*d mod (p-1)*(q-1) = 901 mod 60 = 1

Based on [Bi05]

Simplified Example (II)

- Bob wants to send the message „HELLO WORLD“ to Alice.
- Each plaintext character is represented by a number between 00 (A) and 25 (Z).
- Therefore, we have the plaintext as:
  `07 04 11 11 14 26 22 14 17 11 03`




Bob

Based on [Bi05]

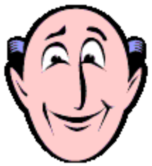
Simplified Example (III)

- Using Alice's public key the ciphertext is:
  - 07^{17} mod 77 = 28
  - 04^{17} mod 77 = 16
  - 11^{17} mod 77 = 44

  …
  - 03^{17} mod 77 = 75

Result: 28 16 44 44 42 38 22 42 19 44 75

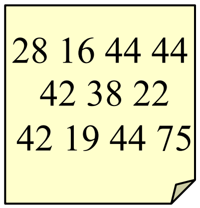
HELLO WORLD



Bob

Based on [Bi05]

Simplified Example (IV)



Alice

- Alice decrypts the ciphertext by calculating:
  - $28^{53} \bmod 77 = 07$
  - $16^{53} \bmod 77 = 04$
  - $44^{53} \bmod 77 = 11$

  …
  - $75^{53} \bmod 77 = 03$

Result: 07 04 11 11 14 26 22 14 17 11 03 = "HELLO WORLD"

Based on [Bi05]

Agenda

- Introduction
- Symmetric Key Cryptography
- Public key cryptography
  - General process
  - Algorithms
  - Hybrid systems
  - Key management
  - Example: PGP

Performance of Public Key Algorithms

Algorithm	Performance*	Performance compared to Symmetric encryption (AES)
RSA (1024 bits)	6.6 s	Factor 100 slower
RSA (2048 bits)	11.8 s	Factor 180 slower

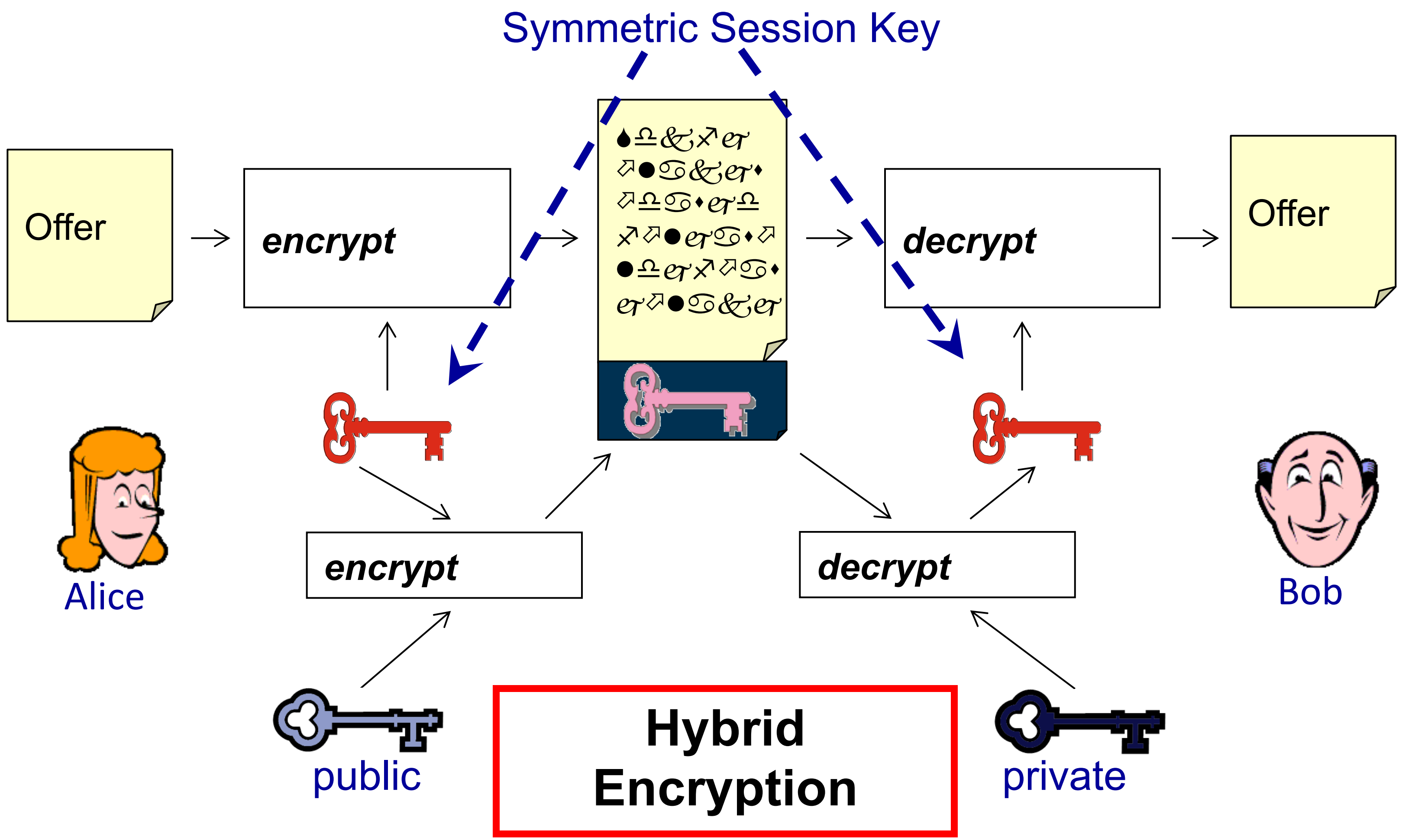
Disadvantage: Complex operations with very big numbers

⇒ Algorithms are very slow.

* Encryption of 1 MB on a Pentium 2.8 GHz, using the FlexiProvider (Java)

[J. Buchmann: Lecture Public Key Infrastrukturen, FG Theoretische Informatik, TU-Darmstadt]

Solution: Hybrid Systems



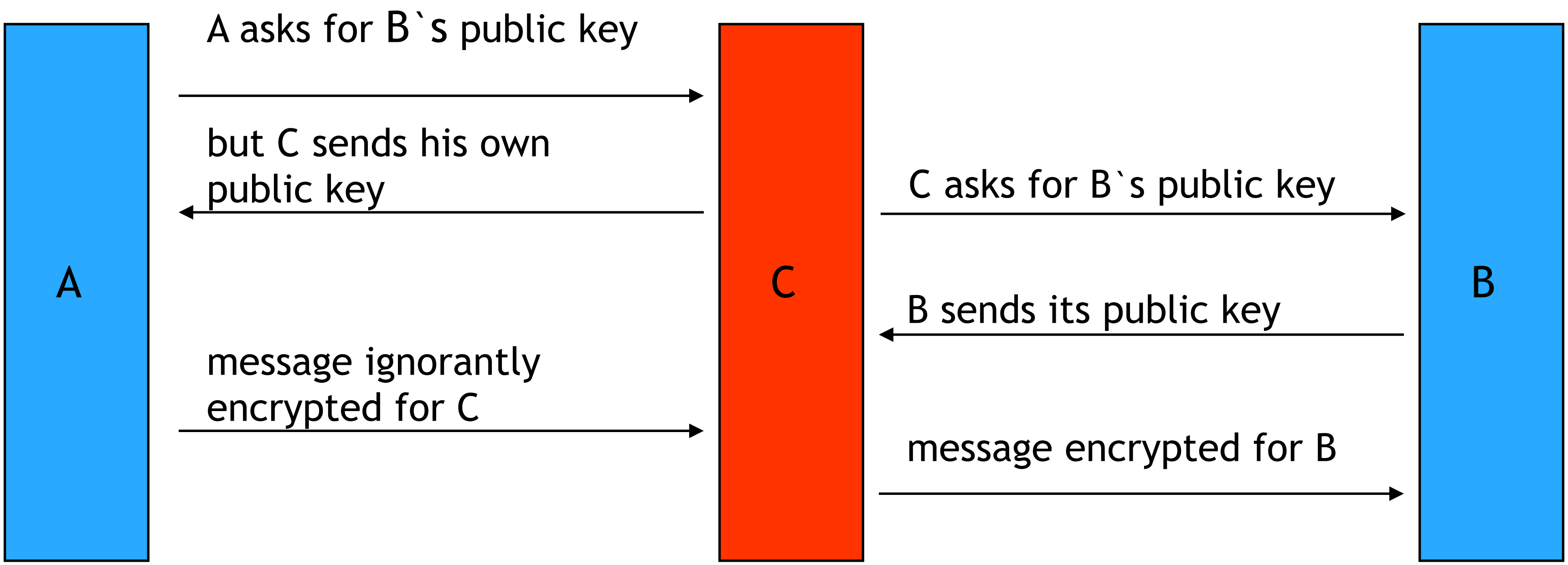
[based on: J. Buchmann 2005: Lecture Public Key Infrastrukturen, FG Theoretische Informatik, TU-Darmstadt]

Agenda

- Introduction
- Symmetric Key Cryptography
- Public key cryptography
  - General process
  - Algorithms
  - Hybrid systems
  - Key management
  - Example: PGP

Attacks on Public Key Distribution

"Man in the middle attack"



➲ Keys are certified: a 3rd person/institution confirms (with its digital signature) the affiliation of the public key to a person.

Certification of Public Keys

- B can freely distribute his own public key.
- But: Everybody (e.g. C) could distribute a public key and claim that this one belongs to B.
- If A uses this key to send a message to B, C will be able to read this message!
- Thus:
  How can A decide if a public key was really created and distributed by B without asking B directly?

➲ Keys get **certified**, i.e. a third person/institution confirms with its (digital) signature the **affiliation of a public key to entity B**.

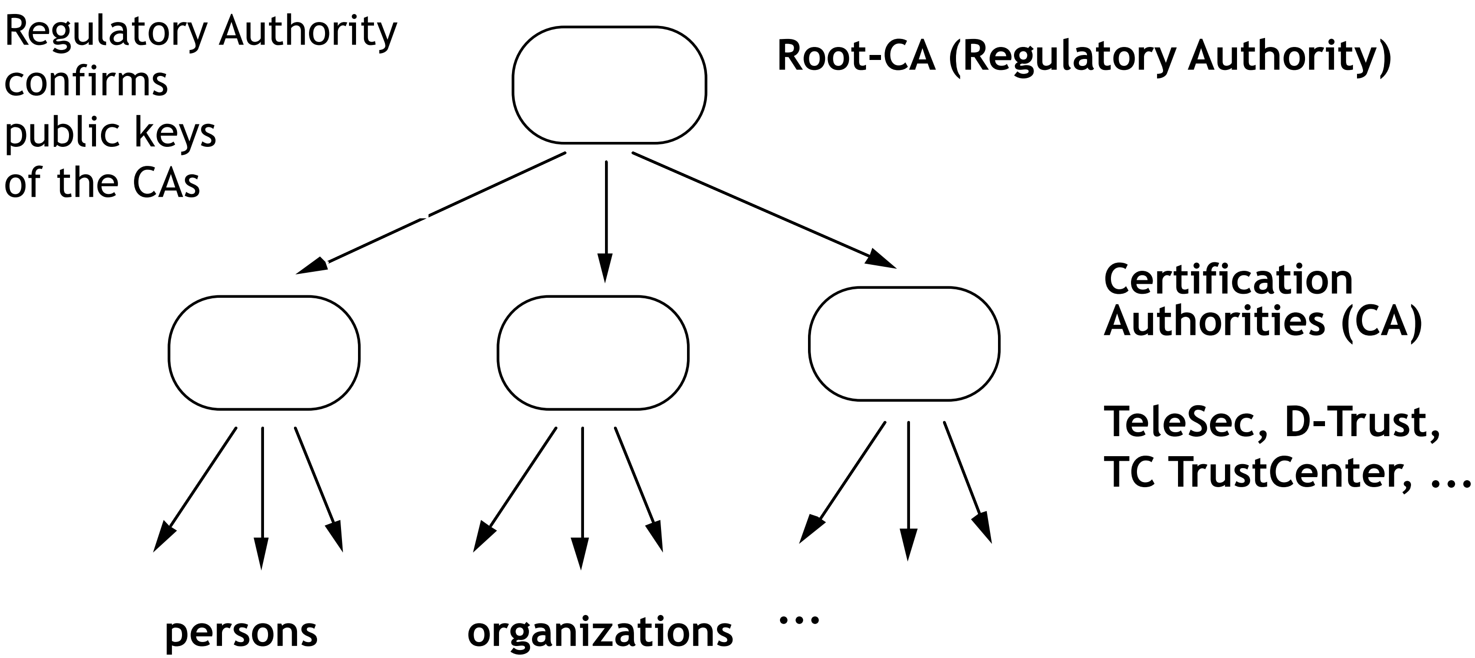
➲ Public Key Infrastructures (PKIs)

Certification of Public Keys

Three types of organization for certification systems (PKIs?):

- Central Certification Authority (CA)
  - A single CA, keys often integrated in checking software
  - Example: older versions of Netscape (CA = Verisign)
- Hierarchical certification system
  - CAs which in turn are certified by “higher” CA
  - Examples: PEM, TeleTrust, infrastructure according to Signature Law
- Web of Trust
  - Each owner of a key may serve as a CA.
  - Users have to assess certificates on their own.
  - Example: PGP (but with hierarchical overlay system)

Hierarchical Certification of Public Keys
(Example: German Signature Law)

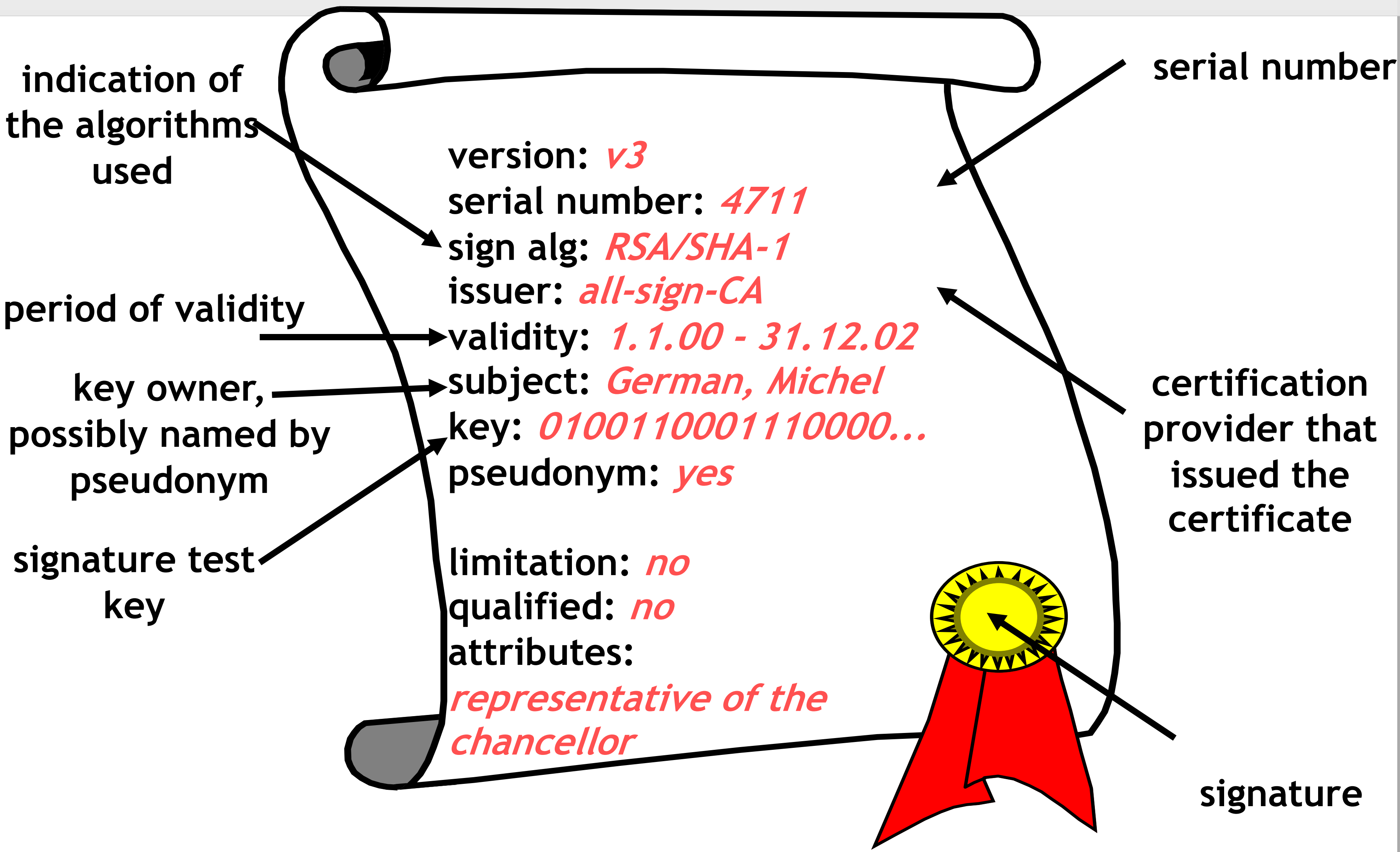


- The actual checking of the identity of the key owner takes place at so called Registration Authorities
(e.g. notaries, bank branches, T-Points, ...)
- Security of the infrastructure depends on the reliability of the CAs.

30

Content of a Key Certificate

(according to German Signature Law and Regulation)



Tasks of a Certification Authority

(according to German Signature Law and Regulation)

- Reliable identification of persons who apply for a certificate
- Information on necessary methods for fraud resistant creation of a signature
- Provision for secure storage of the private key
  - at least Smartcard (protected by PIN)
- Publication of the certificate (if wanted)
- Barring of certificates
- If necessary issuing of time stamps
  - for a fraud resistant proof that an electronic document has been at hand at a specific time

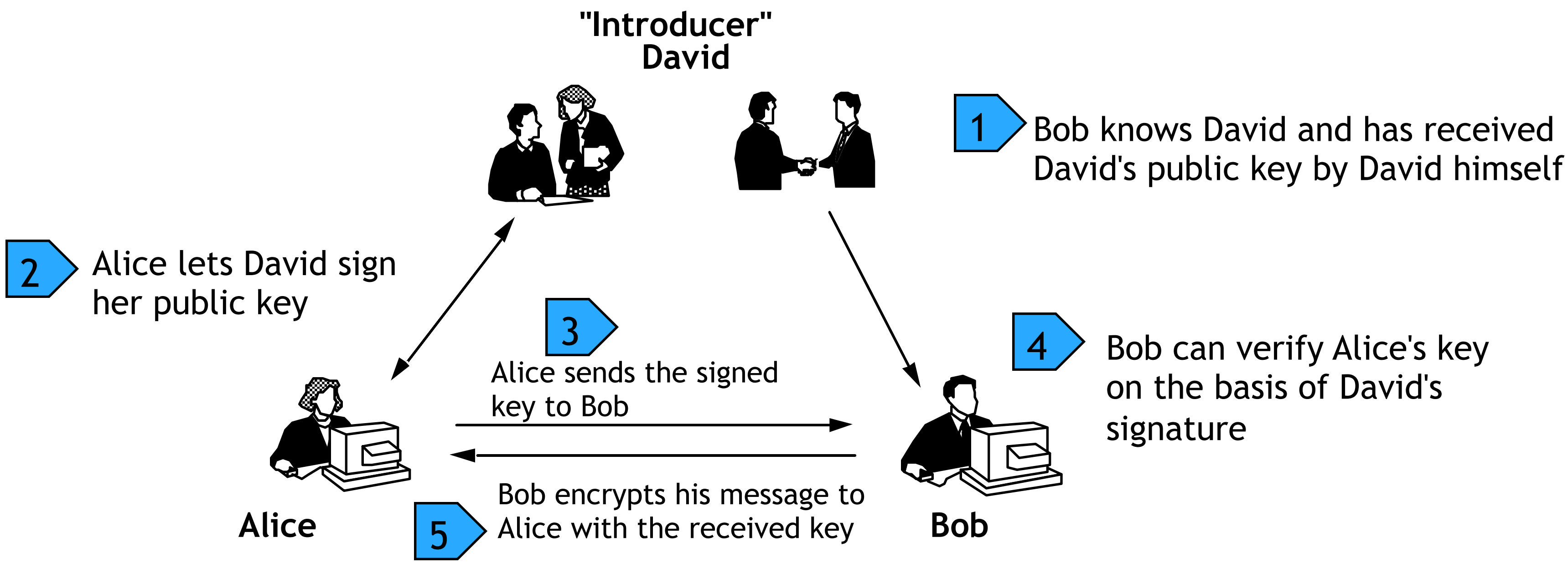
mobile business

Requirements to an Accredited CA
(according to German Signature Law and Regulation)

- Checking of the following items by certain confirmation centers (BSI, TÜViT, ...)
  - Concept of operational security
  - Reliability of the executives and of the employees as well as of their know-how
  - Financial power for continuous operation
  - Exclusive usage of licensed technical components according to SigG and SigV
  - Security requirements as to operating premises and their access controls
- Possibly license of the regulation authority

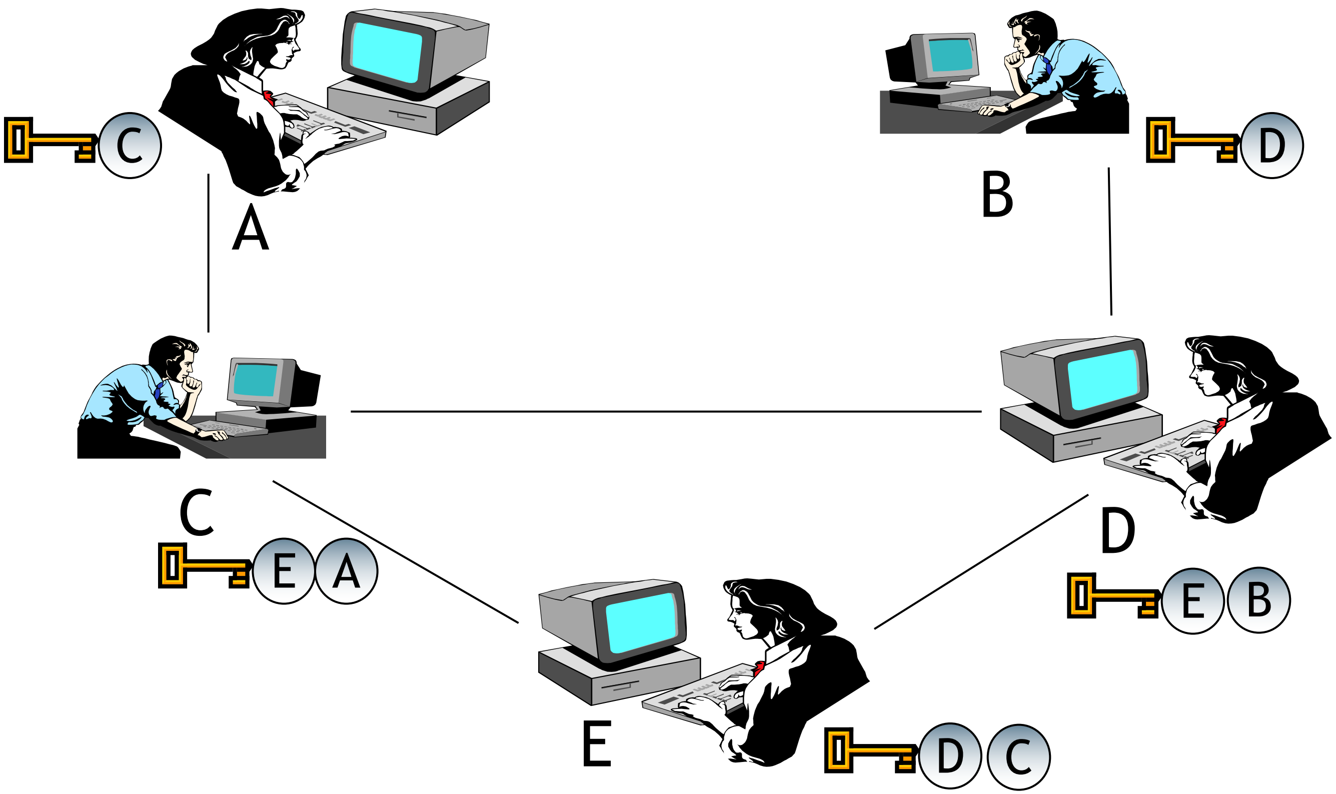
mobile business

Web of Trust



- Each user can act as a “CA“.
- Mapping of the social process of creation of trust
- Keys are “certified” through several signatures.
- Expansion is possible by public key servers and (hierarchical) CAs.

Web of Trust Example



Web of Trust:

- Certification of the public keys mutually by users
- Level of the mutual trust is adjustable.

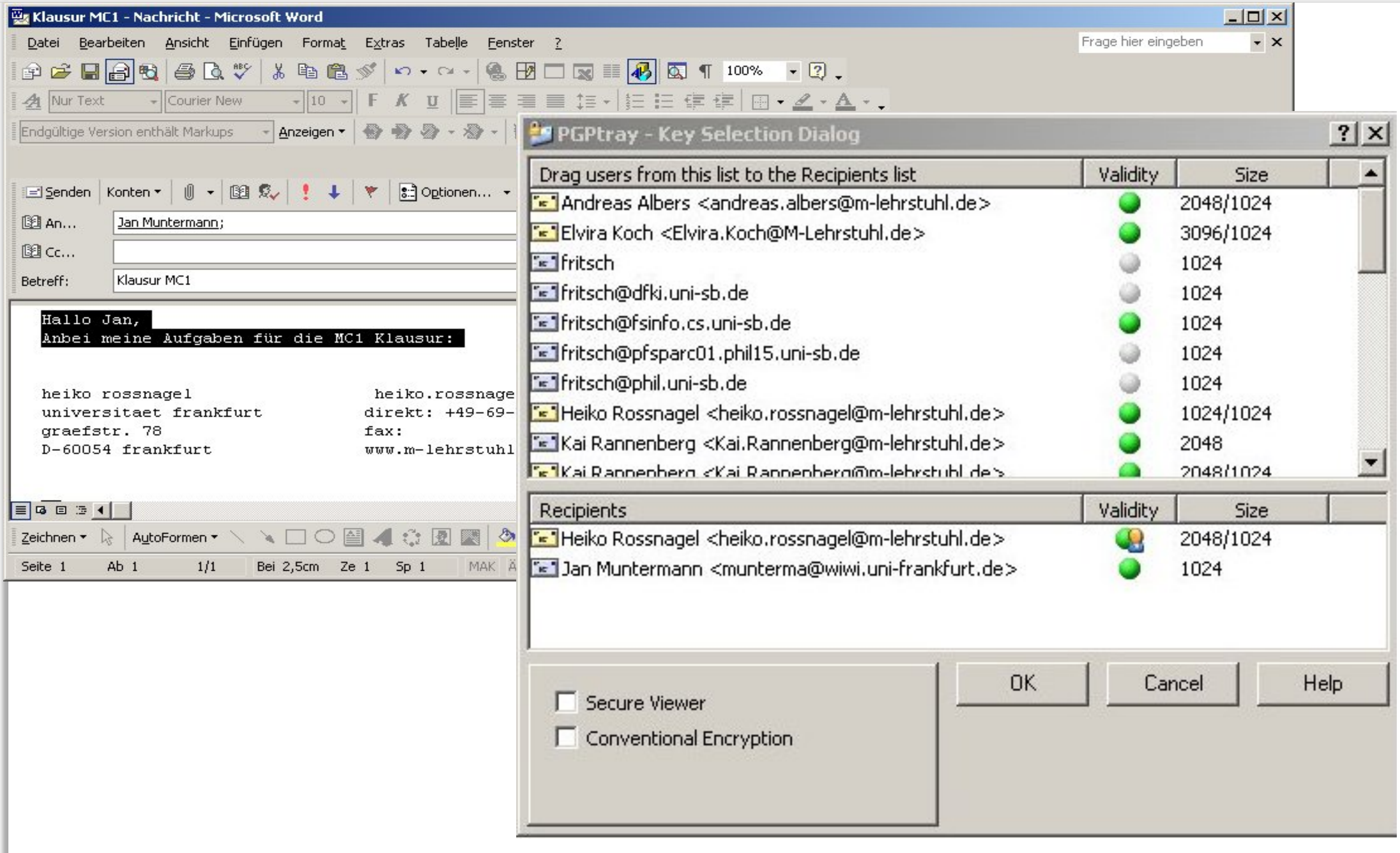
Agenda

- Introduction
- Symmetric Key Cryptography
- Public Key Cryptography
  - General Process
  - Algorithms
  - Hybrid Systems
  - Key Management
  - Example: PGP

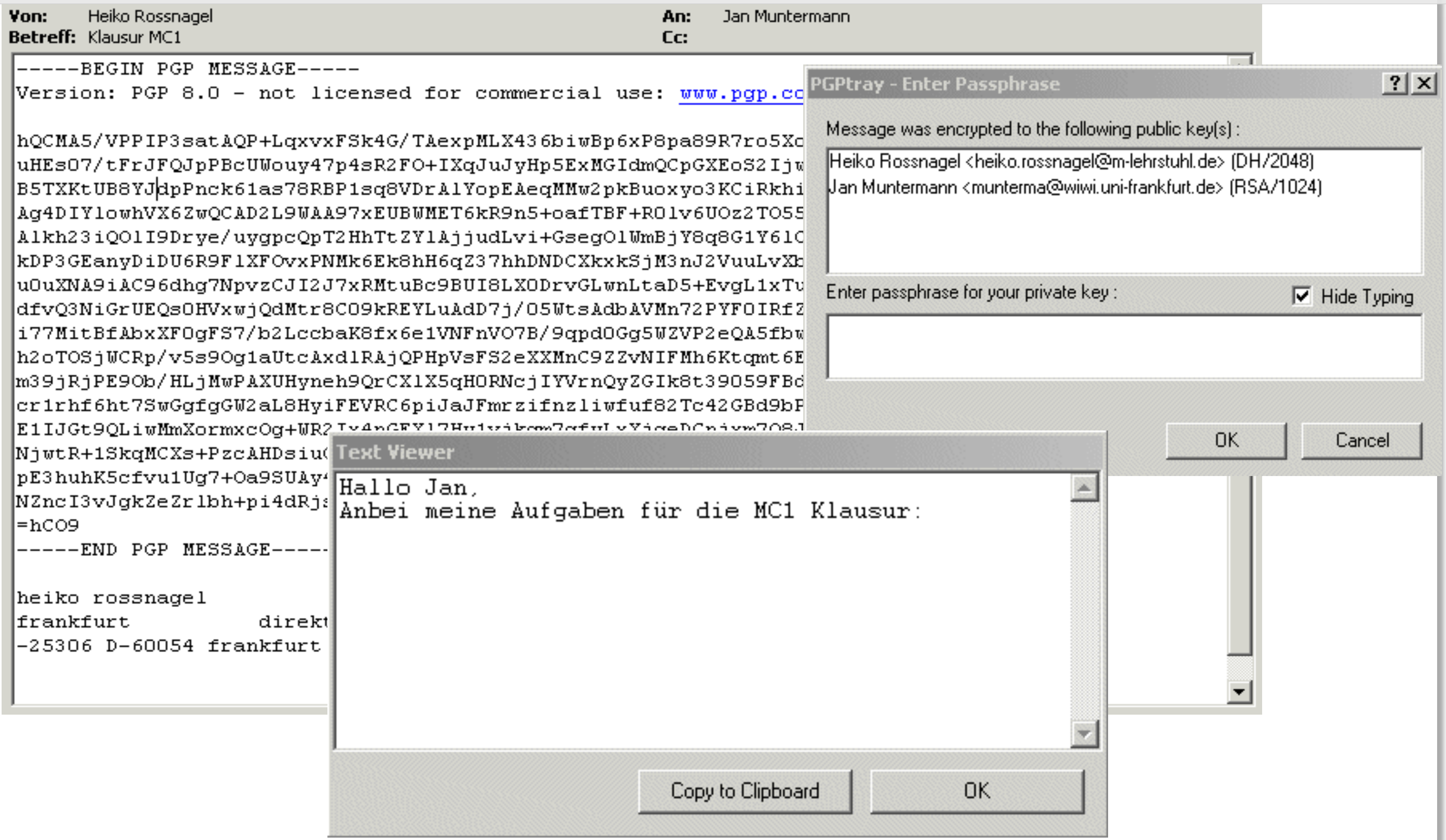
Protection of Email
Example PGP

- PGP = Pretty Good Privacy
- De facto-Standard for freely accessible e-mail encryption systems on the Internet
- First implementation by Phil Zimmermann
- Long trial against Phil Zimmermann because of suspicion of violation of export clauses
- In U.S. free version in cooperation with MIT (agreement with RSA because of then patent)
- PGP company, bought and sold by several companies.
- Gnu Privacy Guard (GPG): non-commercial Open Source variant (OpenPGP, RFC2440)

PGP: Encrypt Message



PGP: Decrypt Message

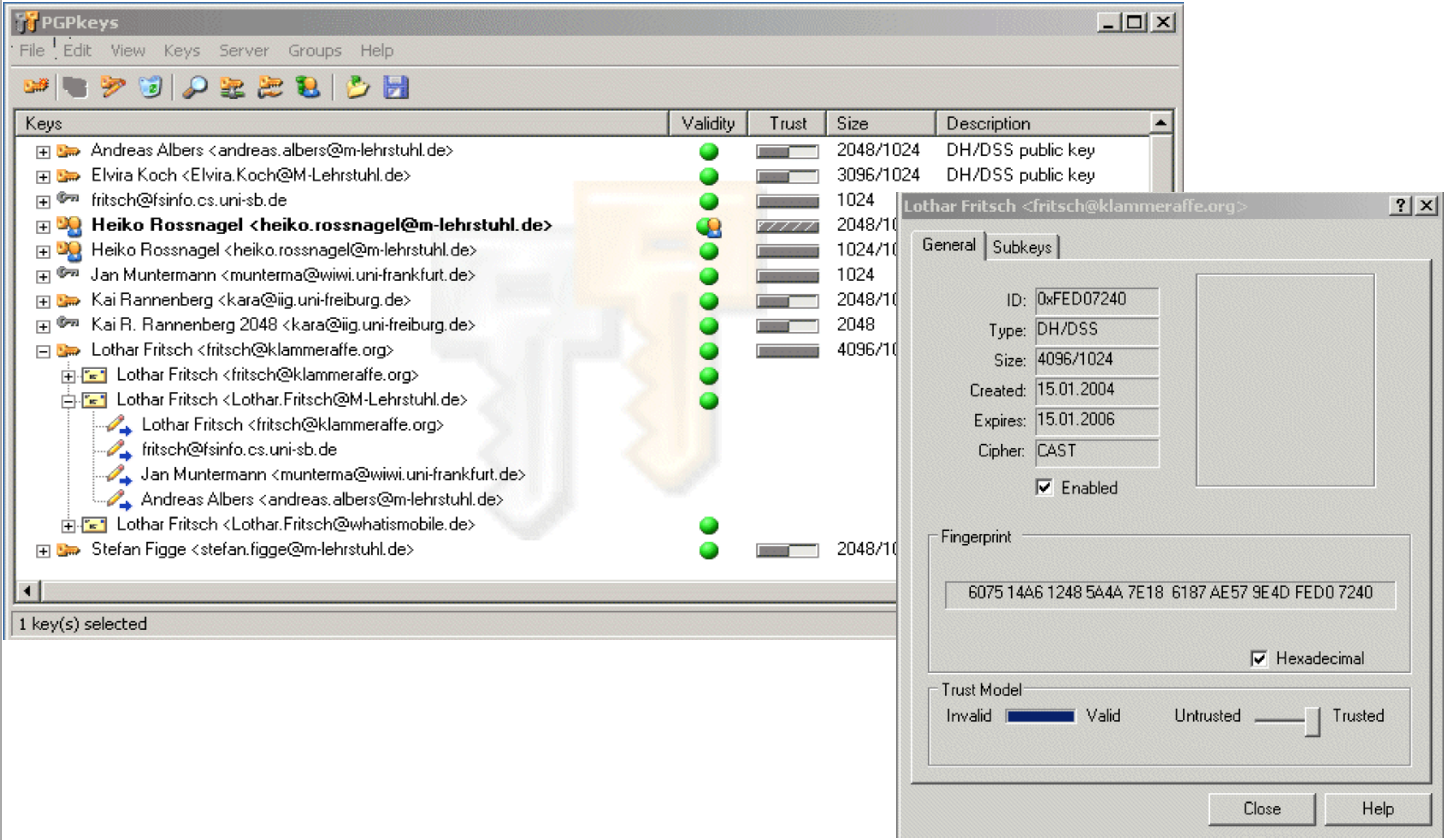


PGP-Certification of Keys

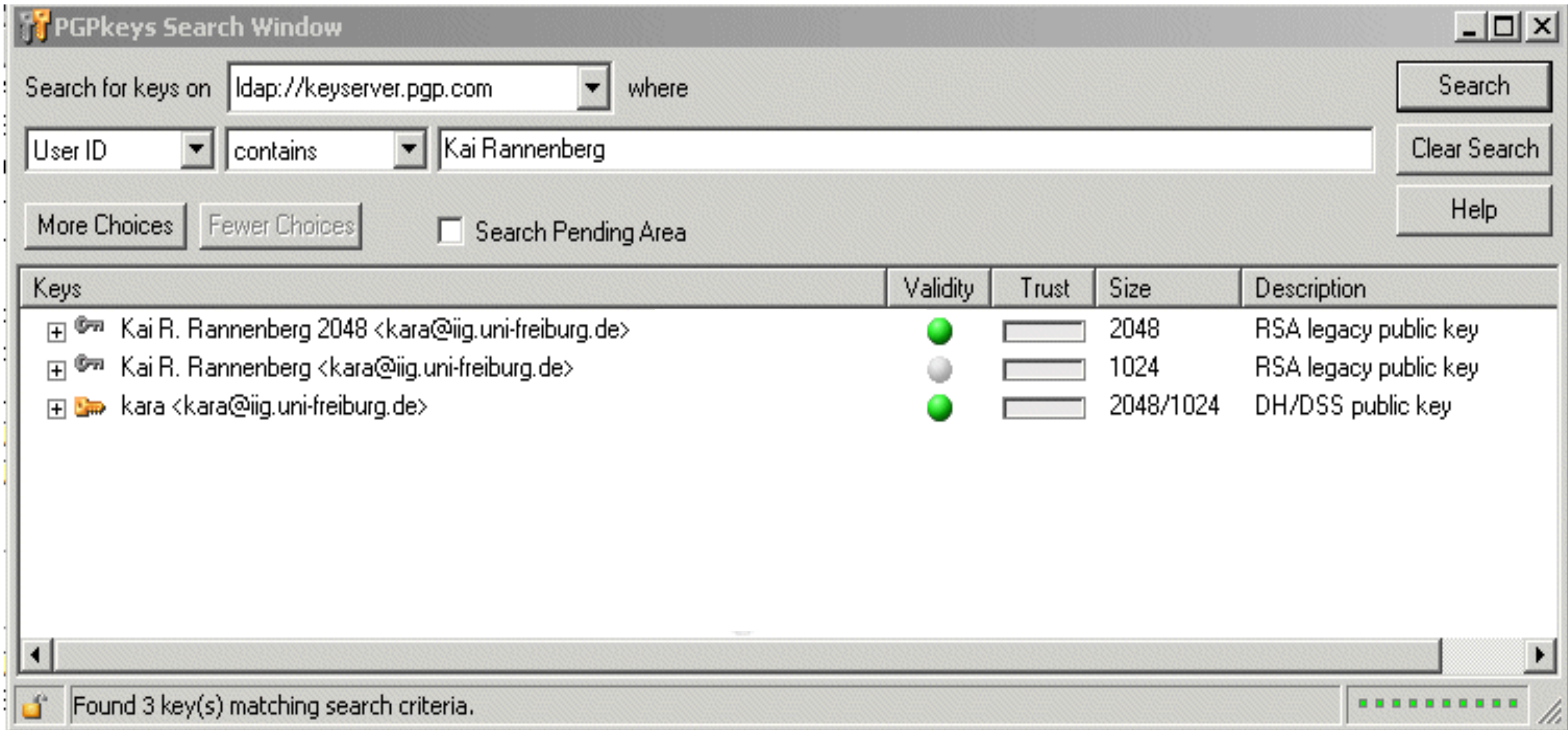
- Certification of public keys by users: “Web of Trust”
- Differentiation between ‘validity’ and ‘trust’
  - ‘Trust’: trust that a person / an institution signs keys only if their authenticity has really been checked
  - ‘Validity’: A key is valid for me if it has been signed by a person / an institution I trust (ideally by myself)
- Support through key servers
  - Collection of keys
  - Allocation of ‘validity’ and ‘trust’ remains task of the users.
- Path server: finding certification paths between keys

mobile business

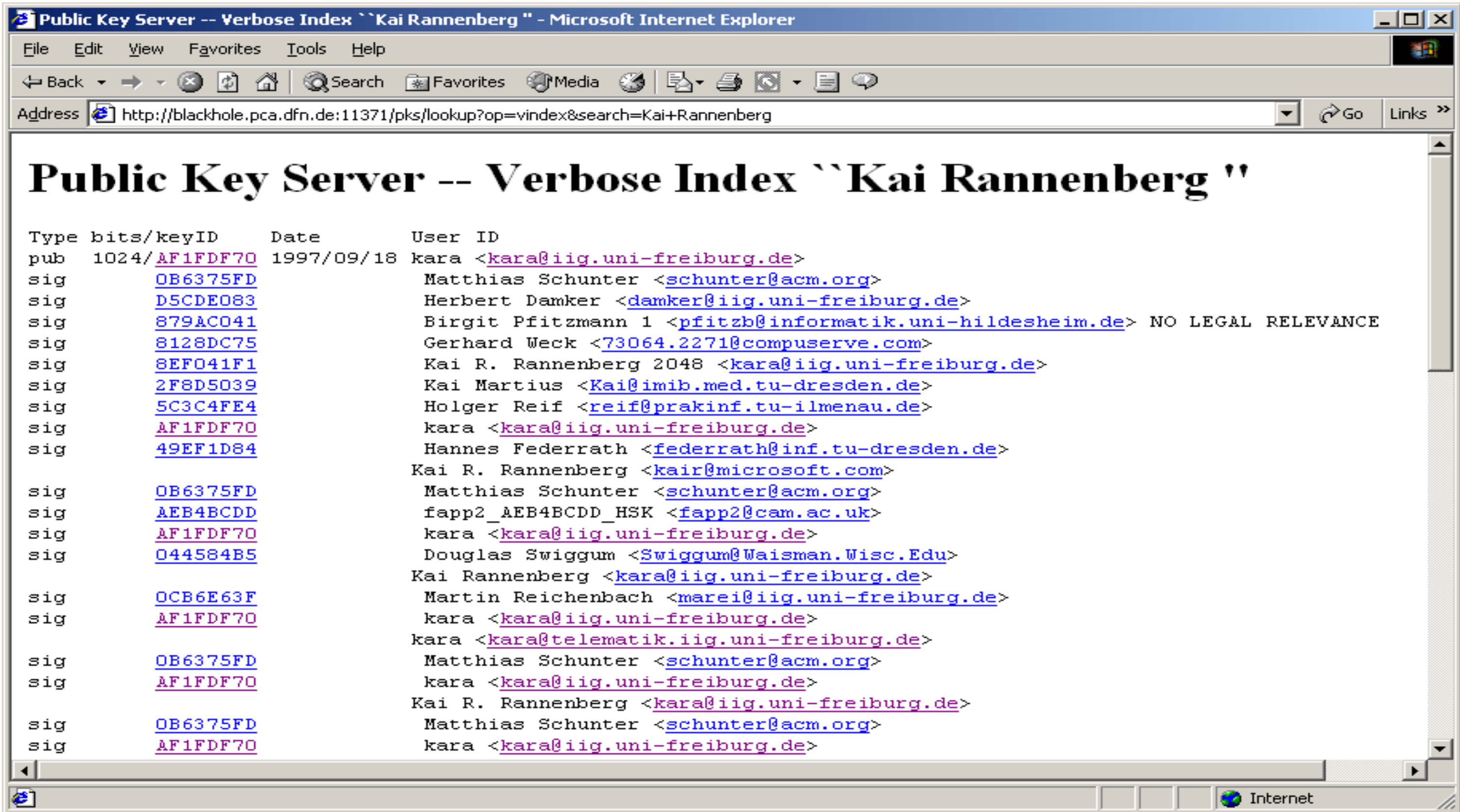
PGP: Key Management



Key Server


PGPkeys Search Window
Search for keys on
ldap://keyserver.pgp.com
where
Search
User ID
contains
Kai Rannenberg
Clear Search
More Choices
Fewer Choices
Search Pending Area
Help
Keys
Validity
Trust
Size
Description
Kai R. Rannenberg 2048 <kara@iig.uni-freiburg.de>
2048
RSA legacy public key
Kai R. Rannenberg <kara@iig.uni-freiburg.de>
1024
RSA legacy public key
kara <kara@iig.uni-freiburg.de>
2048/1024
DH/DSS public key
Found 3 key(s) matching search criteria.

PGP: Public Key Catalogs



Public Key Server -- Verbose Index ``Kai Rannenberg ''

```
Type bits/keyID    Date       User ID
pub  1024/AF1FDF70 1997/09/18 kara <kara@iig.uni-freiburg.de>
sig        0B6375FD             Matthias Schunter <schunter@acm.org>
sig        D5CDE083             Herbert Damker <damker@iig.uni-freiburg.de>
sig        879AC041             Birgit Pfitzmann 1 <pfitzb@informatik.uni-hildesheim.de> NO LEGAL RELEVANCE
sig        8128DC75             Gerhard Weck <73064.2271@compuserve.com>
sig        8EF041F1             Kai R. Rannenberg 2048 <kara@iig.uni-freiburg.de>
sig        2F8D5039             Kai Martius <Kai@imib.med.tu-dresden.de>
sig        5C3C4FE4             Holger Reif <reif@prakinf.tu-ilmenau.de>
sig        AF1FDF70             kara <kara@iig.uni-freiburg.de>
sig        49EF1D84             Hannes Federrath <federrath@inf.tu-dresden.de>
                              Kai R. Rannenberg <kair@microsoft.com>
sig        0B6375FD             Matthias Schunter <schunter@acm.org>
sig        AEB4BCDD             fapp2_AEB4BCDD_HSK <fapp2@cam.ac.uk>
sig        AF1FDF70             kara <kara@iig.uni-freiburg.de>
sig        044584B5             Douglas Swiggum <Swiggum@Waisman.Wisc.Edu>
                              Kai Rannenberg <kara@iig.uni-freiburg.de>
sig        0CB6E63F             Martin Reichenbach <marei@iig.uni-freiburg.de>
sig        AF1FDF70             kara <kara@iig.uni-freiburg.de>
                              kara <kara@telematik.iig.uni-freiburg.de>
sig        0B6375FD             Matthias Schunter <schunter@acm.org>
sig        AF1FDF70             kara <kara@iig.uni-freiburg.de>
                              Kai R. Rannenberg <kara@iig.uni-freiburg.de>
sig        0B6375FD             Matthias Schunter <schunter@acm.org>
sig        AF1FDF70             kara <kara@iig.uni-freiburg.de>
```

- Network of public-key servers:
  - http://pgp.uni-mainz.de/
  - www.cam.ac.uk.pgp.net/pgpnet/email-key-server-info.html
  - …

PGP: Practical Attacks and Weaknesses

- Brute-Force-Attacks on the pass phrase
  - PGPCrack for conventionally encrypted files
- Trojan horses, changed PGP-Code
  - e.g. predictable random numbers, encryption with an additional key
- Attacks on the computer of the user
  - Not physically deleted files
  - Paged memory
  - Keyboard monitoring
- Analysis of electromagnetic radiation
- Non-technical attacks
- Confusion of users [WT99]

Outlook and Post-Quantum Cryptography

- Cryptographic mechanisms become less secure over time (e.g. Moore's law)
- Quantum computers may break conventional public-key cryptography
  - Shor's factoring algorithm can solve factoring problems in polynomial time.
  - E.g. RSA, Diffie-Hellman, ELGamal outdated
- Post-Quantum Cryptography
  - Actually post-quantum-computing cryptography
  - E.g. AES is resistant but key size increases

[Kn19]

References

- **[Bi05] Bishop, Matt:** *Introduction to Computer Security*. Boston: Addison Wesley, 2005. pp. 113-116.
- **[DH76] Diffie, Whitfield and Hellman, Martin E.**: New Directions in Cryptography, *IEEE Transactions on Information Theory*, 1976, 22(6), pp. 644-654.
- **[Kn19]** Knospe, Heiko. A Course in Cryptography. Vol. 40. American Mathematical Soc., 2019. p. 251.
- **[RSA78] Rivest, Ron L., Shamir, A. and Adleman, L.:** A Method for Obtaining Digital Signatures and Public Key Cryptosystems, *Communications of the ACM*, February 1978, 21(2), pp. 120-126.
- **[WT99] Whitten, Alma and Tygar, J.D.** *Why Johnny Can´t Encrypt: A Usability Evaluation of PGP 5.0*, In: Proceedings of the 9th USENIX Security Symposium, August 1999, www.gaudior.net/alma/johnny.pdf