

Information & Communication Security (SS 15)

Cryptography II

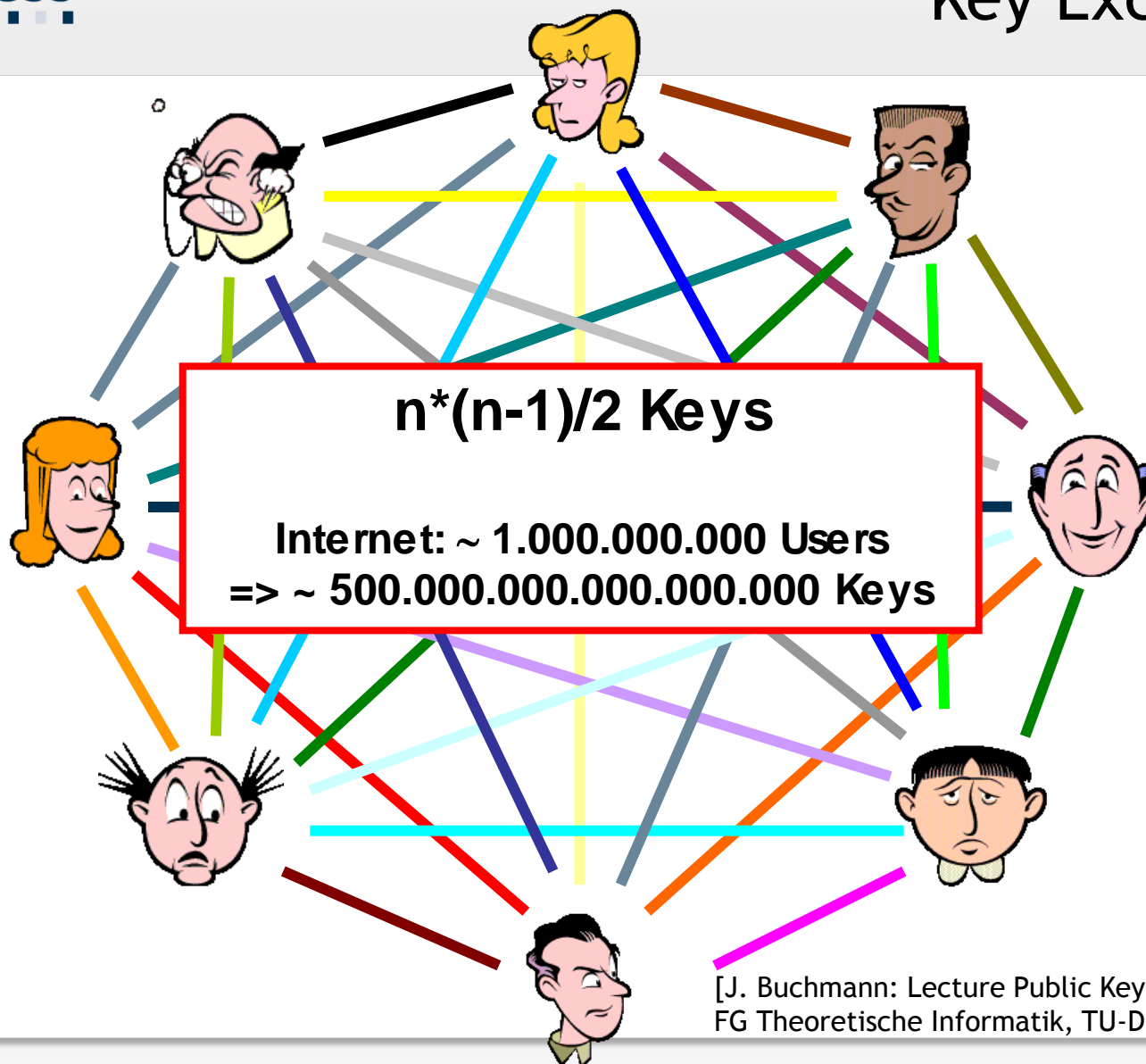
Prof. Dr. Ioannis Stamatiou

Department of Business Administration, University of Patras, Greece

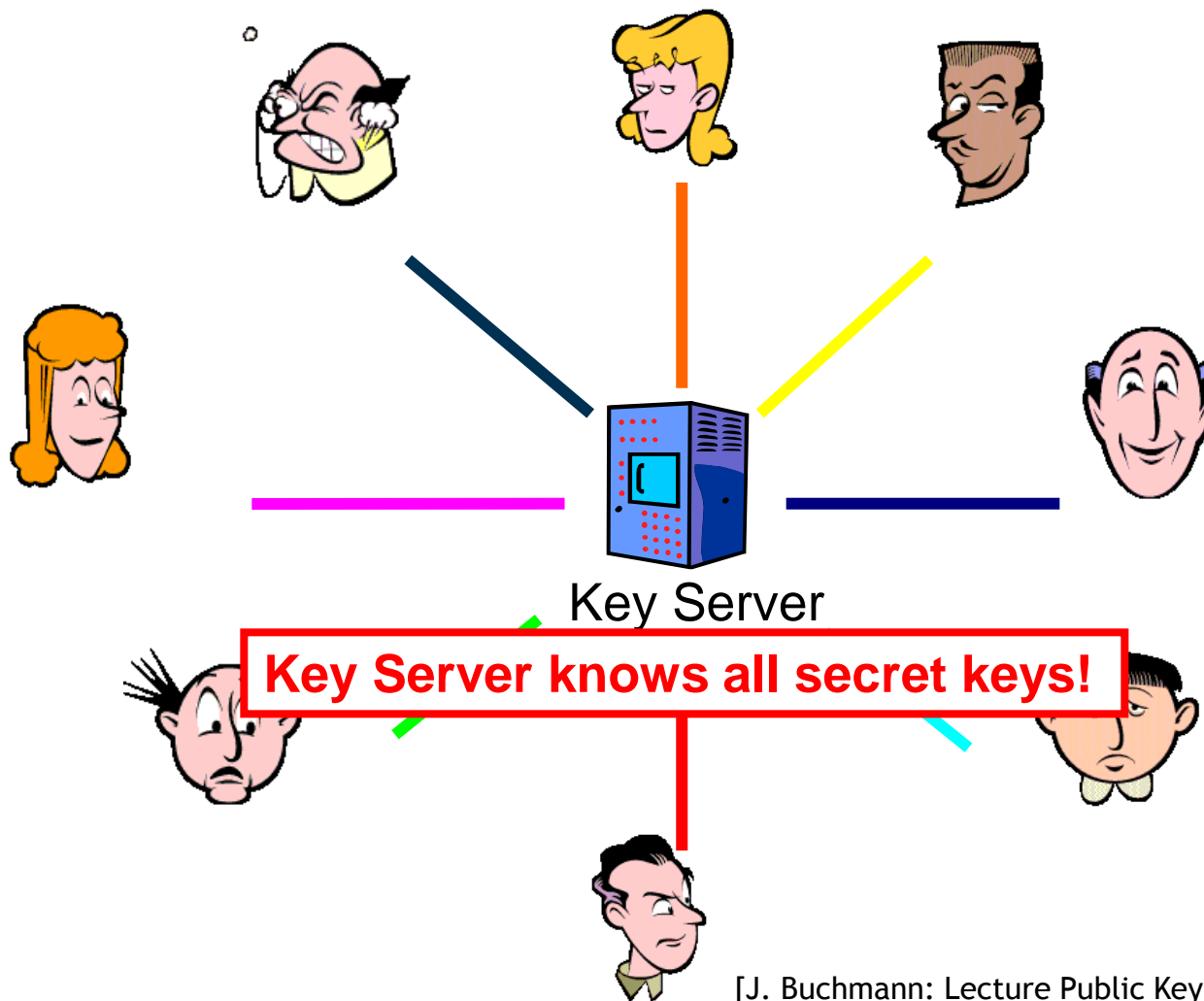
stamatiu@ceid.upatras.gr

- Introduction
- Classical cryptosystems
- Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Example: PGP

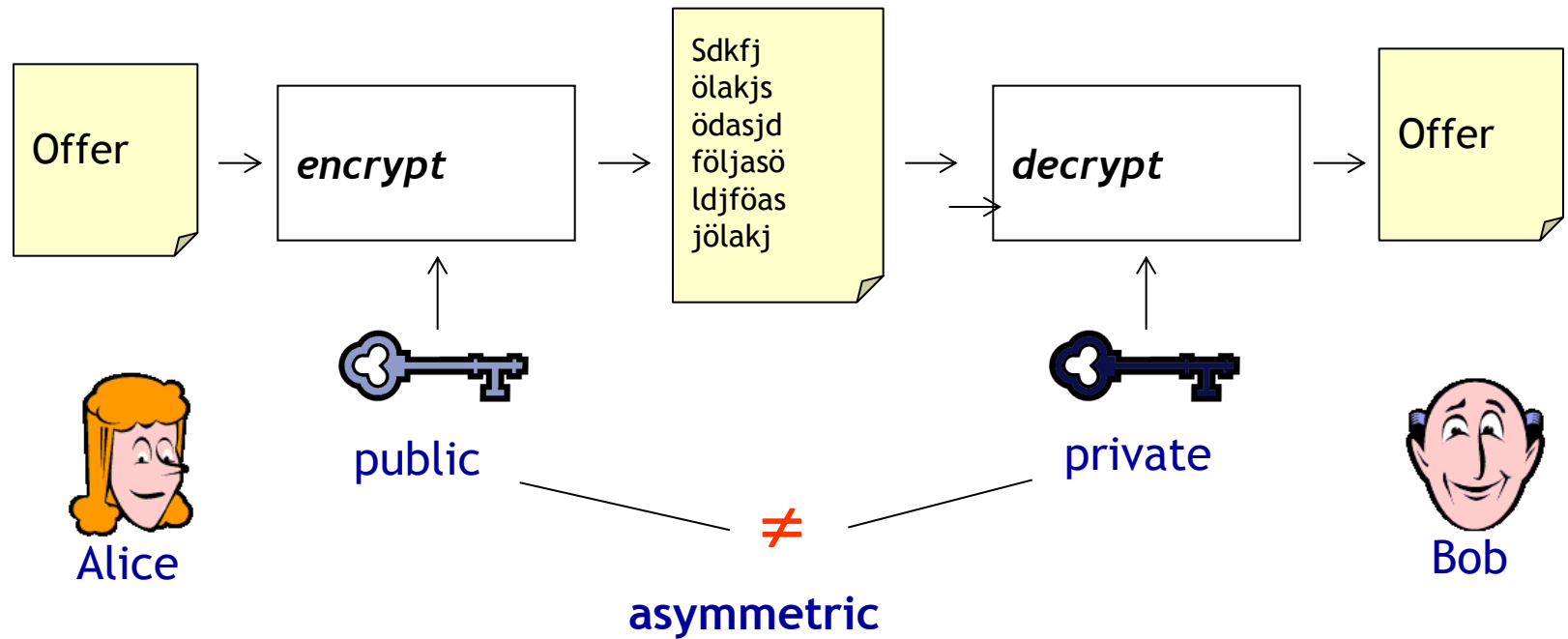
Problems of Symmetric Cryptosystems: Key Exchange



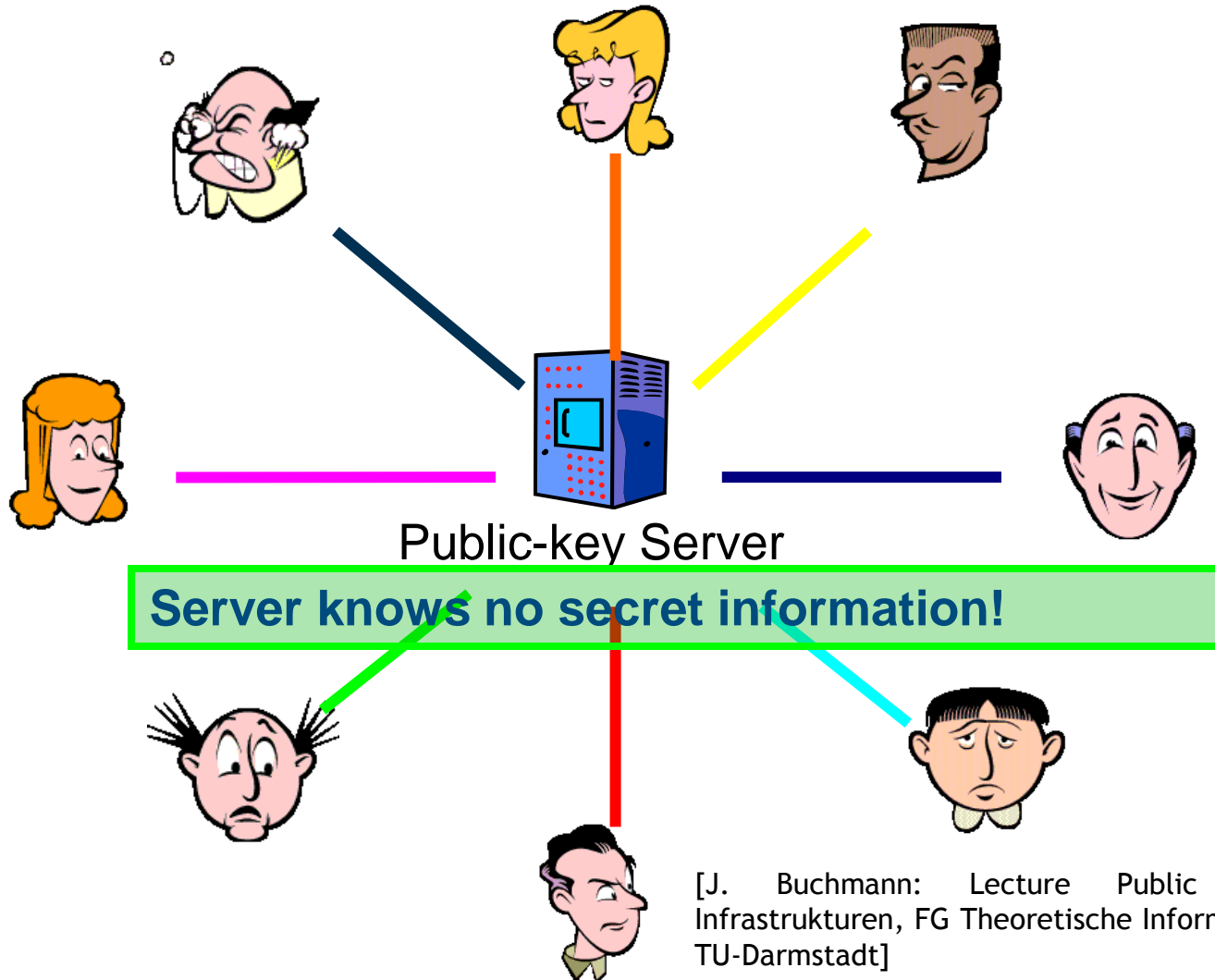
[J. Buchmann: Lecture Public Key Infrastrukturen,
FG Theoretische Informatik, TU-Darmstadt]



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



Key Exchange Problem Solved!

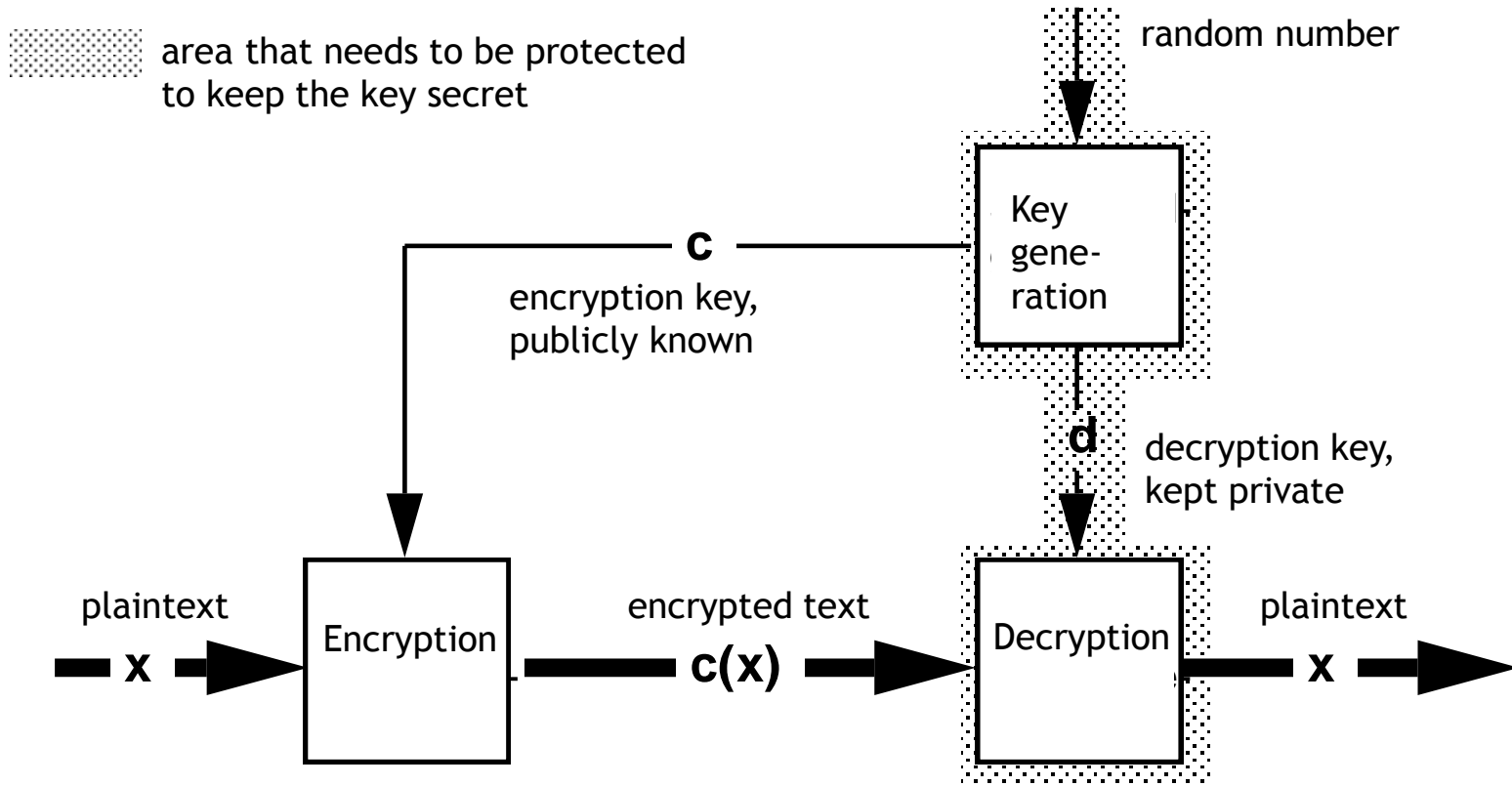


- Introduction
- Classical cryptosystems
- Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Example: PGP

Concept of Asymmetric Encryption Systems

- 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 Systems



box with slot, access to messages only with a key

[based on Federrath and Pfitzmann 1997]

- Introduction
- Classical cryptosystems
- Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Example: PGP

- **RSA**
 - Rivest, Shamir, Adleman, 1978
 - Based on the assumption that the factorization of the product of two (big) prime numbers ($p \cdot q$) 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

- Introduction
- Classical cryptosystems
- Public key cryptography
 - General concept
 - Algorithms
 - RSA
 - Hybrid systems
 - Key management
 - Example: PGP

- 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) .

- 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.

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

Choosing the Keys (I)

- You first compute n as the product of two 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]

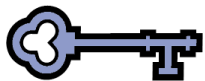
- You then choose an integer d to be a large, random integer which is relatively prime to $(p-1) * (q-1)$.
- That is, check that d satisfies:
 - The greatest common divisor of d and $(p-1) * (q-1)$ is 1.
 - $\text{gcd}(d, (p-1) * (q-1)) = 1$

- 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

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

Simplified Example (I)



Public
(e,n)



Private
(d,n)



Alice

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

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

Hello World



Bob

Simplified Example (III)

- Using Alice's public key the ciphertext is:

- $07^{17} \bmod 77 = 28$

- $04^{17} \bmod 77 = 16$

- $11^{17} \bmod 77 = 44$

...

- $03^{17} \bmod 77 = 75$

- Or 28 16 44 44 42 38 22
42 19 44 75

Hello World



Bob

Simplified Example (IV)

28 16 44 44
42 38 22
42 19 44 75



Alice

- Alice decrypts the ciphertext by calculating:
 - $28^{53} \bmod 77 = 07$
 - $16^{53} \bmod 77 = 04$
 - $44^{53} \bmod 77 = 11$
 - ...
 - $75^{53} \bmod 77 = 03$
- Or:

07	04	11	11	14	26	
22	14	17	11	03	=	
"Hello World"						

- Introduction
- Classical cryptosystems
- Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Example: PGP

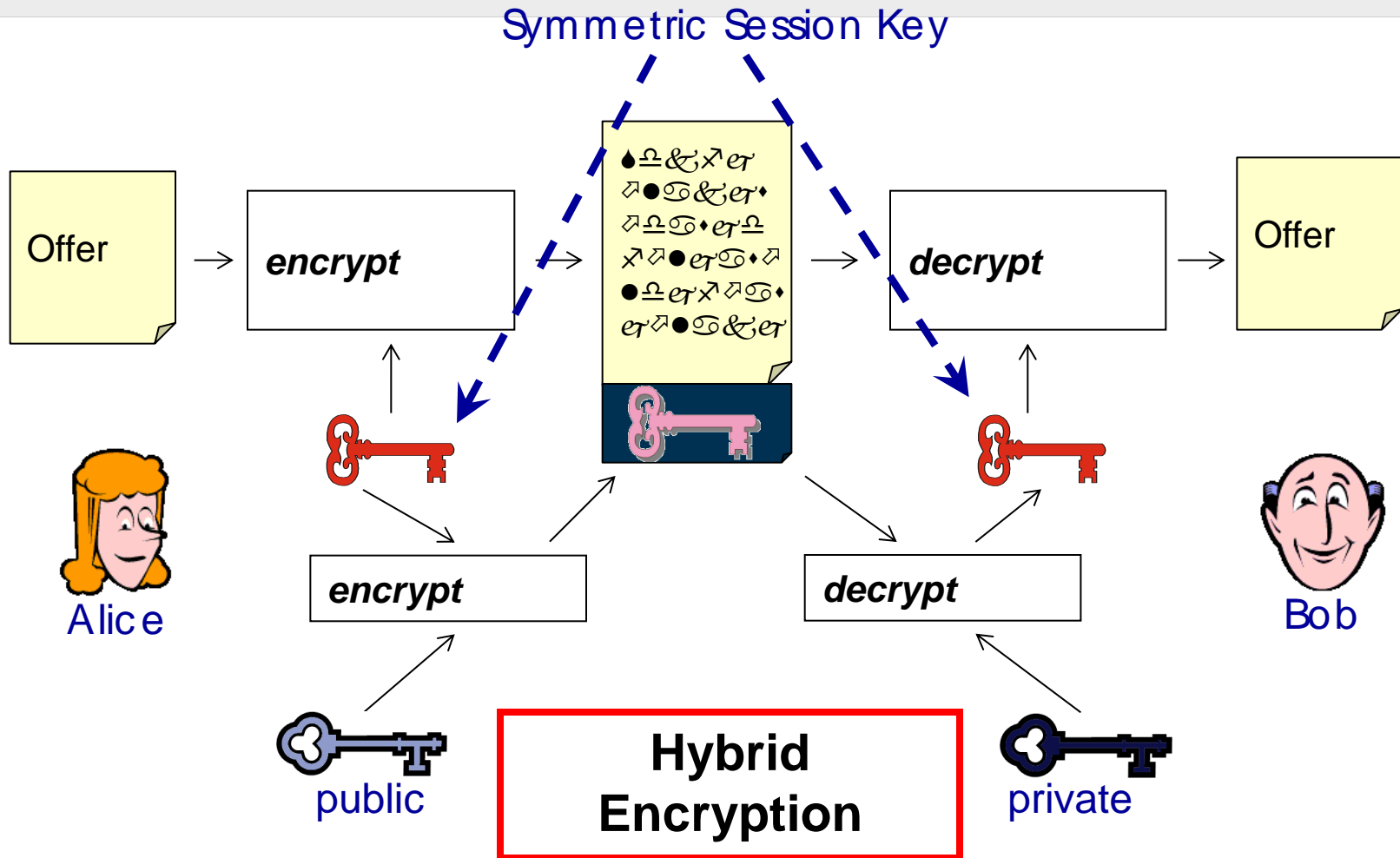
Algorithm	Performance*	Performance compared to Symmetric encryption (AES)
RSA (1024 bits)	6.6 s	Factor 100 slower
RSA (2048 bits)	11.8 s	Factor 180 slower

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)

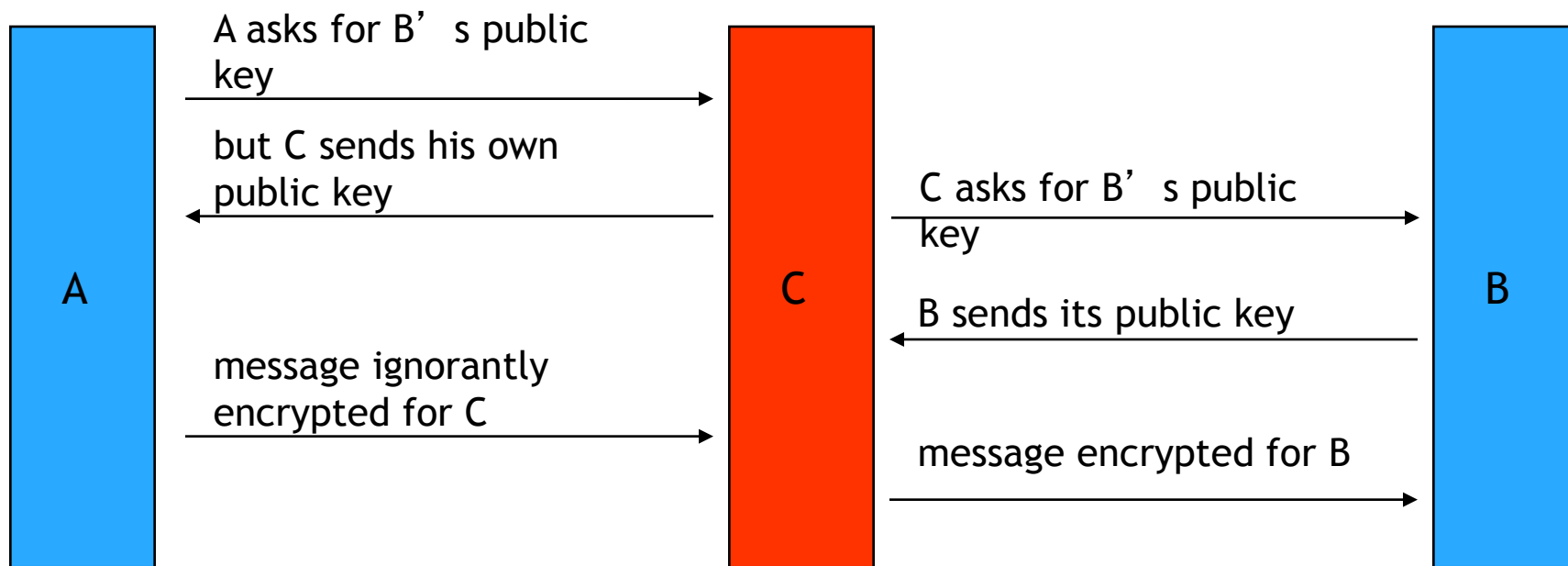
Solution: Hybrid Systems



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

- Introduction
- Classical cryptosystems
- Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Example: PGP

“Man in the middle attack”



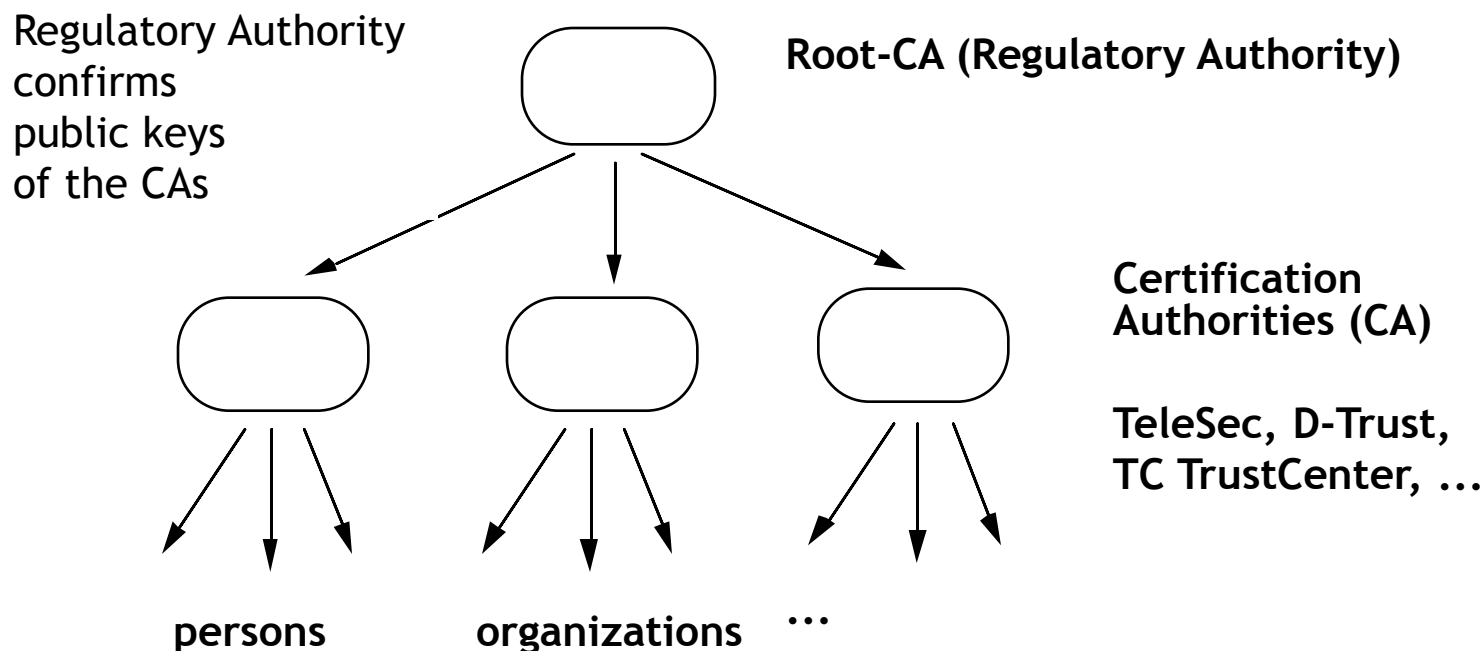
- Keys are certified: a 3rd person/institution confirms (with its digital signature) the affiliation of the public key to a person.

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)

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

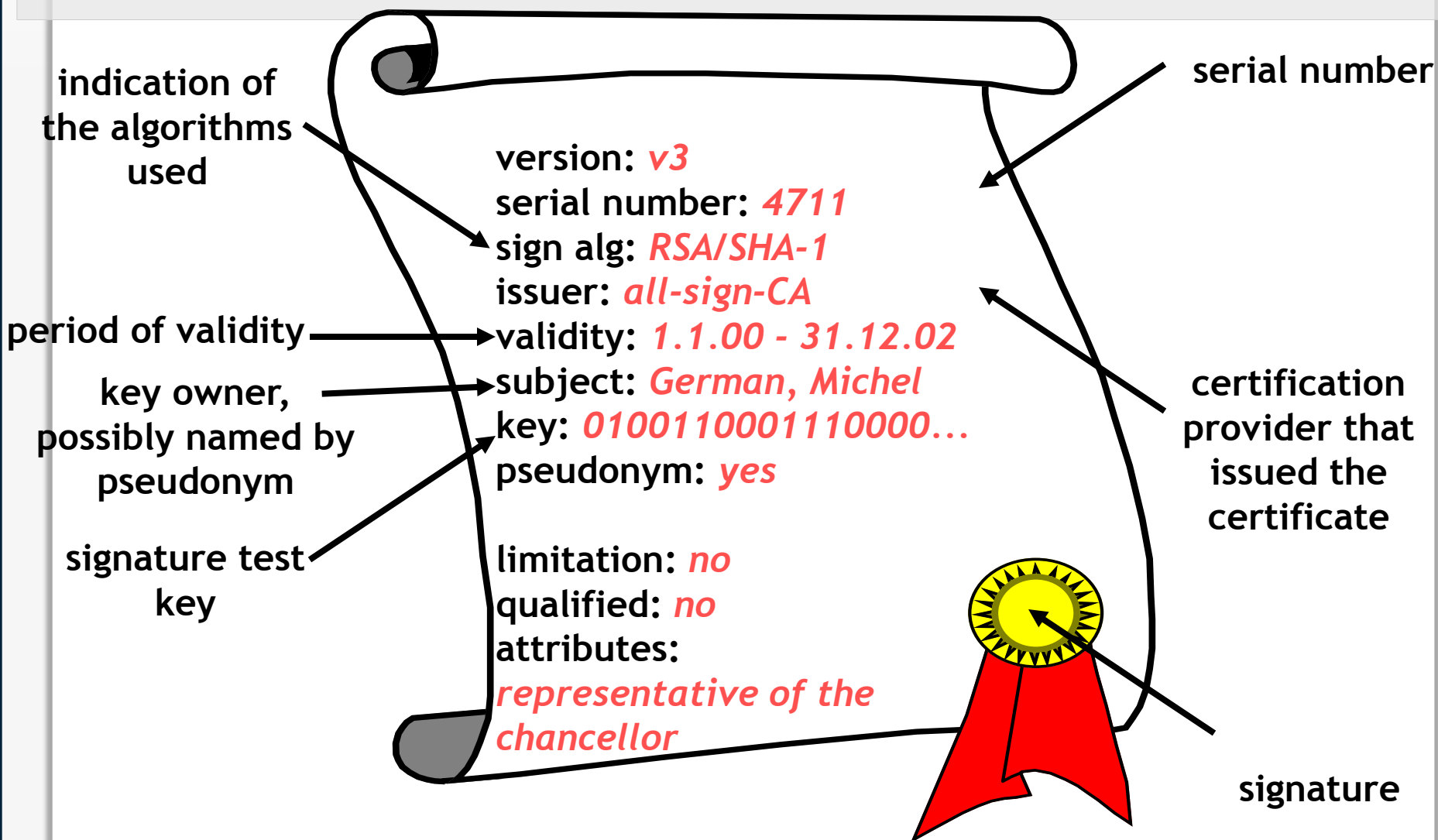
- Central Certification Authority (CA)
 - A single CA, keys often integrated in checking software
 - eExample: older versions of Netscape (CA = Verisign)
- Hierarchical certification system
 - CAs which in turn are certified by “higher” CA
 - Examples: PEM, TeleTrust, infrastructure according to Signature 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)



- 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.

Content of a Key Certificate

(according to German Signature Law and Regulation)

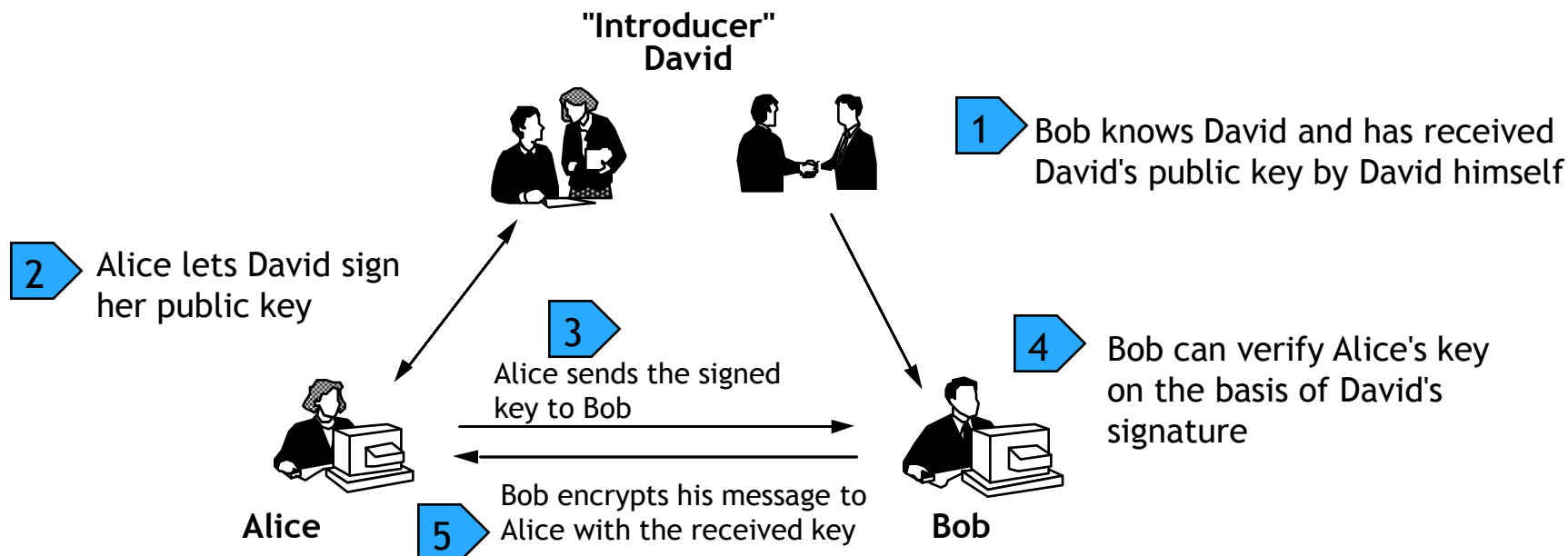


Tasks of a Certification Authority

(according to German Signature Law and Regulation)

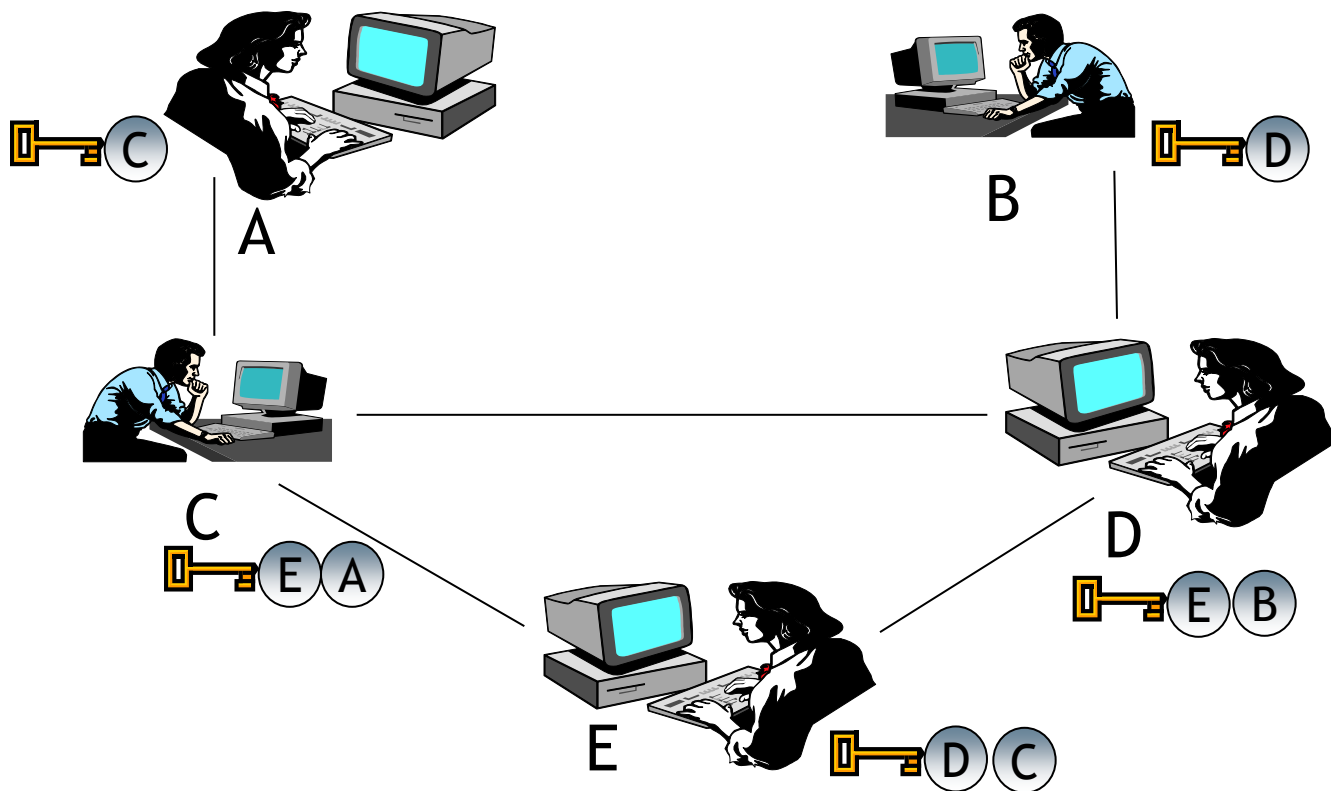
- Reliable identification of persons who apply for a certificate
- Information on necessary methods for fraud resistant creation of a signature
- Provision for secure storage of the private key
 - at least Smartcard (protected by PIN)
- Publication of the certificate (if wanted)
- Barring of certificates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

- 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



- 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.

- Introduction
- Classical cryptosystems
- Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - 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)
- Meanwhile commercialized: www.pgp.com
- Gnu Privacy Guard (GPG): non-commercial Open Source variant (OpenPGP, RFC2440)

Klausur MC1 - Nachricht - Microsoft Word

Frage hier eingeben

Senden Konten An...: Jan Muntermann; Cc...: Betreff: Klausur MC1

Hallo Jan,
Anbei meine Aufgaben für die MC1 Klausur:

heiko rossnagel universitaet frankfurt graefstr. 78 D-60054 frankfurt
heiko.rossnagel@m-lehrstuhl.de direkt: +49-69-7... fax: www.m-lehrstuhl.de

PGPtray - Key Selection Dialog

Drag users from this list to the Recipients list	Validity	Size
Andreas Albers <andreas.albers@m-lehrstuhl.de>	●	2048/1024
Elvira Koch <Elvira.Koch@M-Lehrstuhl.de>	●	3096/1024
fritsch	●	1024
fritsch@dfki.uni-sb.de	●	1024
fritsch@fsinfo.cs.uni-sb.de	●	1024
fritsch@pfsparc01.phil15.uni-sb.de	●	1024
fritsch@phil.uni-sb.de	●	1024
Heiko Rossnagel <heiko.rossnagel@m-lehrstuhl.de>	●	1024/1024
Kai Rannenbergl <Kai.Rannenbergl@m-lehrstuhl.de>	●	2048
Kai Rannenbergl <Kai.Rannenbergl@m-lehrstuhl.de>	●	2048/1024

Recipients	Validity	Size
Heiko Rossnagel <heiko.rossnagel@m-lehrstuhl.de>	●	2048/1024
Jan Muntermann <munterma@wiwi.uni-frankfurt.de>	●	1024

Secure Viewer
 Conventional Encryption

OK Cancel Help

PGP: Decrypt Message

Von: Heiko Rossnagel
Betreff: Klausur MC1

An: Jan Muntermann
Cc:

-----BEGIN PGP MESSAGE-----

Version: PGP 8.0 - not licensed for commercial use: www.pgp.com

hQCMA5/VPPIP3satAQP+LqxvxFsk4G/TaexpMLX436biwBp6xP8pa89R7ro5Xo.
uHEs07/tFrJFQJpPbcUWouy47p4sR2FO+IXqJuJyHp5ExMGIdmQCpGXEOs2Ijw.
B5TXKtUB8YJ|dpPnck61as78RBP1sq8VDrAlYopEAeqMMw2pkBuoxyo3KCiRkhi.
Ag4DIYlowhVX62wQCAD2L9WAA97xEUBWMET6kR9n5+oafTBF+ROlv6UOz2TO55.
Alkh23iQOI9Drye/uygpcQpT2HhTtZY1AjjudLvi+GsegOlWmBjY8q8G1Y61C.
kDP3GEanyDiDU6R9F1XFovxPNMk6Ek8hH6qZ37hhDNDcXkxkSjM3nJ2VuuLvXb.
uOuXNA9iAC96dhg7NpvzCJI2J7xRMtuBc9BUI8LXODrvGLwnLtaD5+EvgL1xTu
dfvQ3NiGrUEQsOHVxwjQdMtr8C09kREYLuAdD7j/05WtsAdbAVMn72PYFOIRfZ.
i77MitBfAbxXF0gFS7/b2LccbaK8fx6e1VNFnVO7B/9qpdOGg5WZVP2eQA5fbw.
h2oTOSjWCRp/v5s9Og1aUtcAxd1RajQPhVvsFS2eXXMn9ZzvNIFMh6Ktqnt6Ei.
m39jRjPE9Ob/HLjMwPAXUHyneh9QrCX1X5qHORNcjIYVrnQyZGIk8t39059FBd.
cr1rhf6ht7SwGgfgGW2aL8HyiFEVRC6piJaJFmrzifnzliwfuf82Tc42GBd9bP.
E1IJGt9QLiwMmXormxcOg+WR2Ix4nGFX17Hy1vjKqpn7gfyLxXjgeDCnjxm708J
Njwtr+1SkqMCXs+PzcAHdsiuG.
pE3huhK5cfvulUg7+Oa9SUay4.
NZncI3vJgkZeZr1bh+pi4dRjs(
=hC09

-----END PGP MESSAGE-----

heiko rossnagel
frankfurt direkt
-25306 D-60054 frankfurt

PGPTray - Enter Passphrase

Message was encrypted to the following public key(s):

Heiko Rossnagel <heiko.rossnagel@m-lehrstuhl.de> (DH/2048)
Jan Muntermann <munterma@wiwi.uni-frankfurt.de> (RSA/1024)

Enter passphrase for your private key:

Hide Typing

OK

Cancel

Text Viewer

Hallo Jan,
Anbei meine Aufgaben für die MC1 Klausur:

Copy to Clipboard

OK

- 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

The screenshot shows the PGPkeys application window. The main window displays a list of keys with columns for Validity, Trust, Size, and Description. A key for Lothar Fritsch is selected, and a detailed dialog box is open for it.

Keys	Validity	Trust	Size	Description
Andreas Albers <andreas.albers@m-lehrstuhl.de>	●	▬	2048/1024	DH/DSS public key
Elvira Koch <Elvira.Koch@M-Lehrstuhl.de>	●	▬	3096/1024	DH/DSS public key
fritsch@fsinfo.cs.uni-sb.de	●	▬	1024	RSA legacy public key
Heiko Rossnagel <heiko.rossnagel@m-lehrstuhl.de>	●	▬	2048/1024	DH/DSS key pair
Heiko Rossnagel <heiko.rossnagel@m-lehrstuhl.de>	●	▬	1024/1024	
Jan Muntermann <munterma@wiwi.uni-frankfurt.de>	●	▬	1024	
Kai Rannenber <kara@iig.uni-freiburg.de>	●	▬	2048/1024	
Kai R. Rannenber 2048 <kara@iig.uni-freiburg.de>	●	▬	2048	
Lothar Fritsch <fritsch@klammeraffe.org>	●	▬	4096/1024	
Lothar Fritsch <fritsch@klammeraffe.org>	●	▬		
Lothar Fritsch <Lothar.Fritsch@M-Lehrstuhl.de>	●	▬		
Lothar Fritsch <fritsch@klammeraffe.org>	●	▬		
fritsch@fsinfo.cs.uni-sb.de	●	▬		
Jan Muntermann <munterma@wiwi.uni-frankfurt.de>	●	▬		
Andreas Albers <andreas.albers@m-lehrstuhl.de>	●	▬		
Lothar Fritsch <Lothar.Fritsch@whatismobile.de>	●	▬		
Stefan Figge <stefan.figge@m-lehrstuhl.de>	●	▬	2048/1024	

Lothar Fritsch <fritsch@klammeraffe.org>	
General	Subkeys
ID:	0xFED07240
Type:	DH/DSS
Size:	4096/1024
Created:	15.01.2004
Expires:	15.01.2006
Cipher:	CAST
<input checked="" type="checkbox"/> Enabled	
Fingerprint	
6075 14A6 1248 5A4A 7E18 6187 AE57 9E4D FED0 7240	
<input checked="" type="checkbox"/> Hexadecimal	
Trust Model	
Invalid	Valid
Untrusted	Trusted

PGPkeys Search Window

Search for keys on where

User ID contains

Search Pending Area

Keys	Validity	Trust	Size	Description
<input type="checkbox"/> Kai R. Rannenberg 2048 <kara@iig.uni-freiburg.de>		<input type="text"/>	2048	RSA legacy public key
<input type="checkbox"/> Kai R. Rannenberg <kara@iig.uni-freiburg.de>		<input type="text"/>	1024	RSA legacy public key
<input type="checkbox"/> kara <kara@iig.uni-freiburg.de>		<input type="text"/>	2048/1024	DH/DSS public key

The screenshot shows a browser window titled "Public Key Server -- Verbose Index ``Kai Rannenberg ``" with the address bar containing the URL: <http://blackhole.pca.dfn.de:11371/pks/lookup?op=vindex&search=Kai+Rannenberg>. The main content area displays a list of public keys with columns for Type, bits/keyID, Date, and User ID.

Type	bits/keyID	Date	User ID
pub	1024/ AF1FDF70	1997/09/18	kara < kara@iig.uni-freiburg.de >
sig	OB6375FD		Matthias Schunter < schunter@acm.org >
sig	DSCDE083		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	8EFO41F1		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-ilmeneau.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	OB6375FD		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	OCB6E63F		Martin Reichenbach < marei@iig.uni-freiburg.de >
sig	AF1FDF70		kara < kara@iig.uni-freiburg.de >
			kara < kara@telematik.iig.uni-freiburg.de >
sig	OB6375FD		Matthias Schunter < schunter@acm.org >
sig	AF1FDF70		kara < kara@iig.uni-freiburg.de >
			Kai R. Rannenberg < kara@iig.uni-freiburg.de >
sig	OB6375FD		Matthias Schunter < schunter@acm.org >
sig	AF1FDF70		kara < kara@iig.uni-freiburg.de >

- Network of public-key servers:
 - pgpkeys.pca.dfn.de
 - www.cam.ac.uk.pgp.net/pgpnet/email-key-server-info.html
 - ...

- 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]

- **[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.
- **[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