

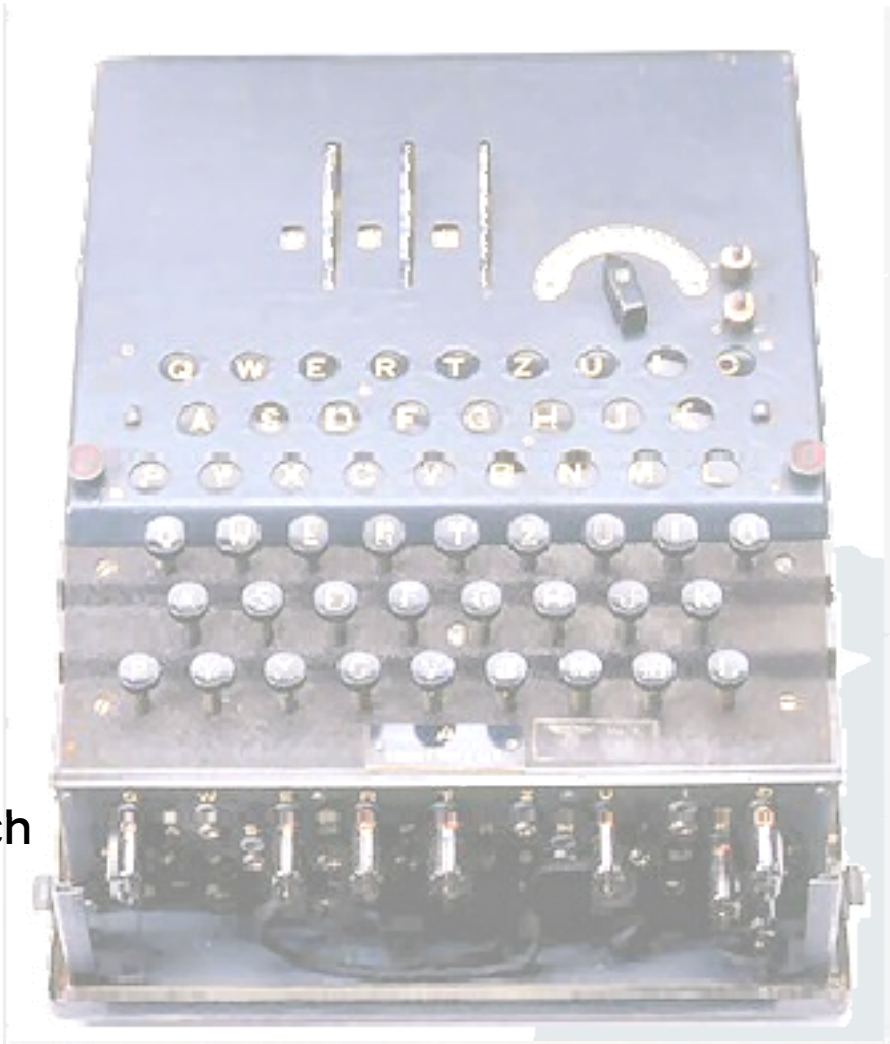
Lecture 5


Cryptography II

Information &
Communications Security
(WS 2008)

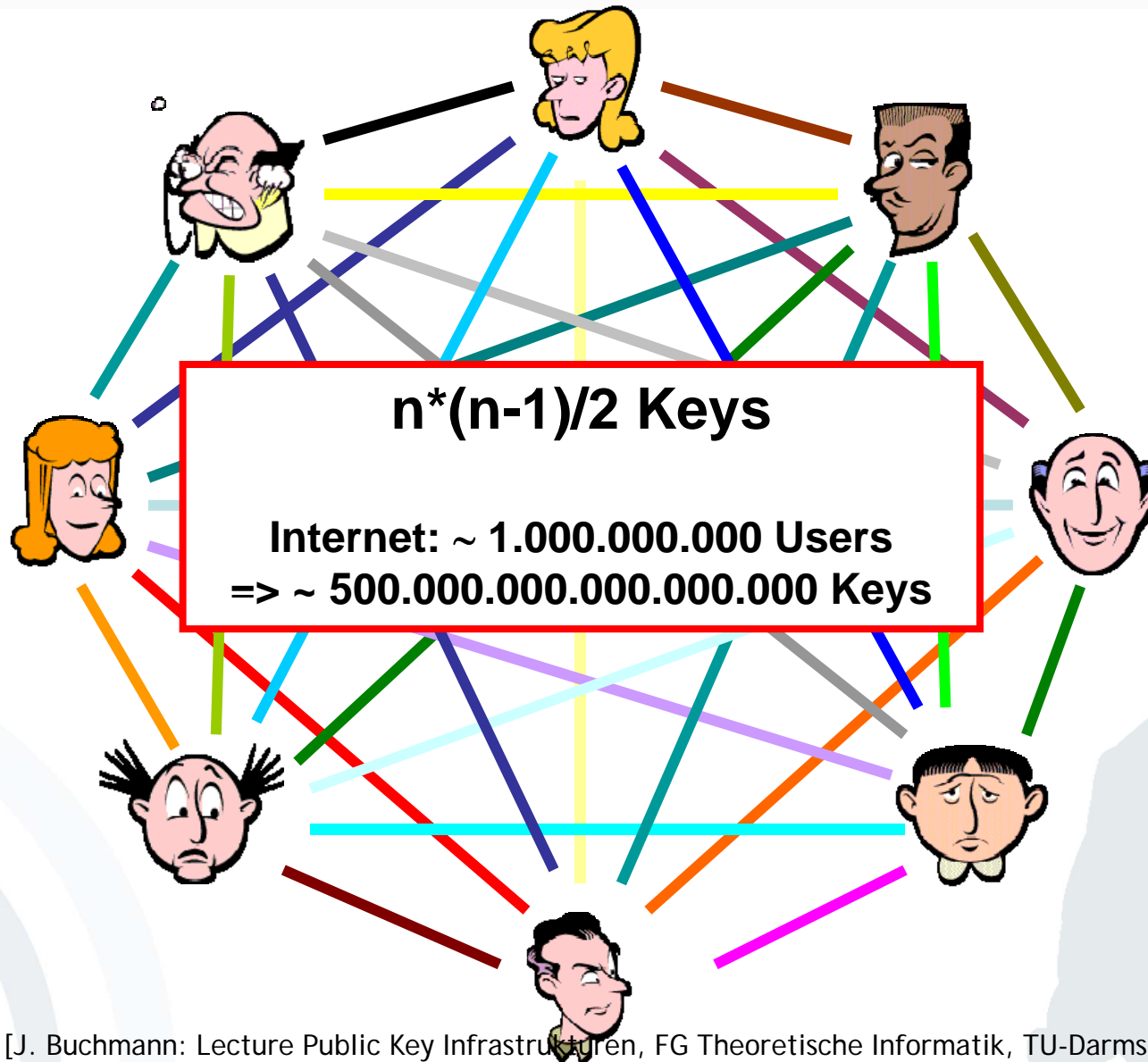
Guest Lecturer Dr. Martin Reichenbach

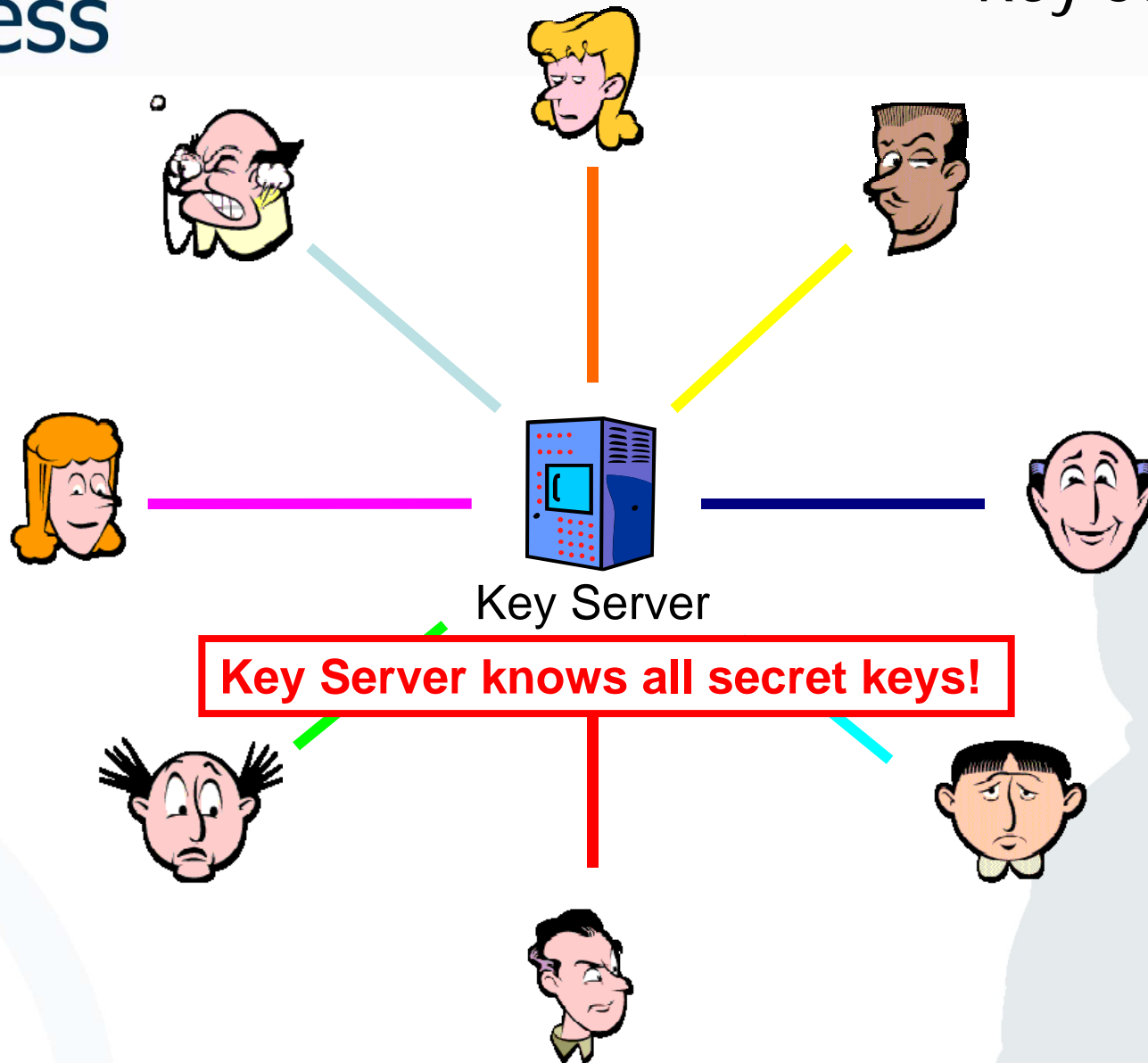
T-Mobile Chair for
Mobile Business & Multilateral Security
Johann Wolfgang Goethe University Frankfurt a. M.
www.whatismobile.de

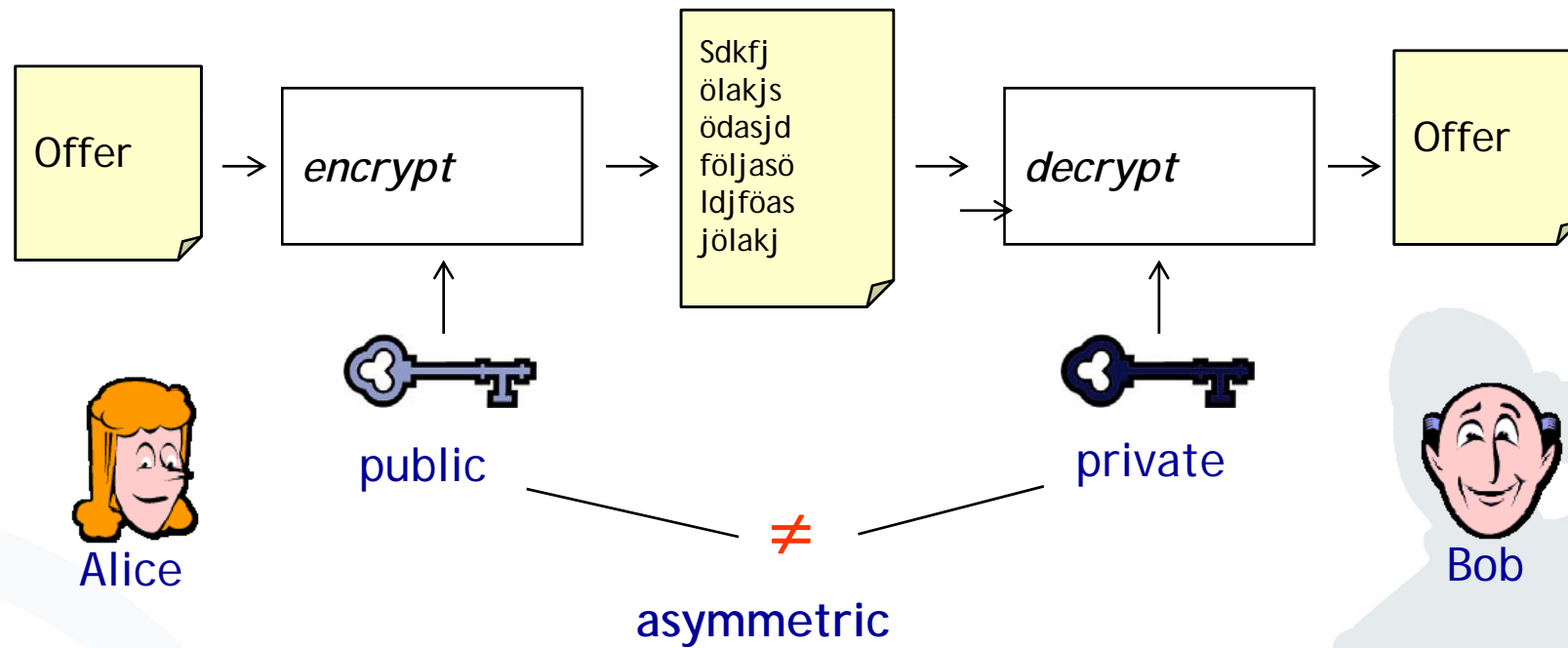


- Introduction
 - Classical cryptosystems
 - Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Encryption Summary
- 

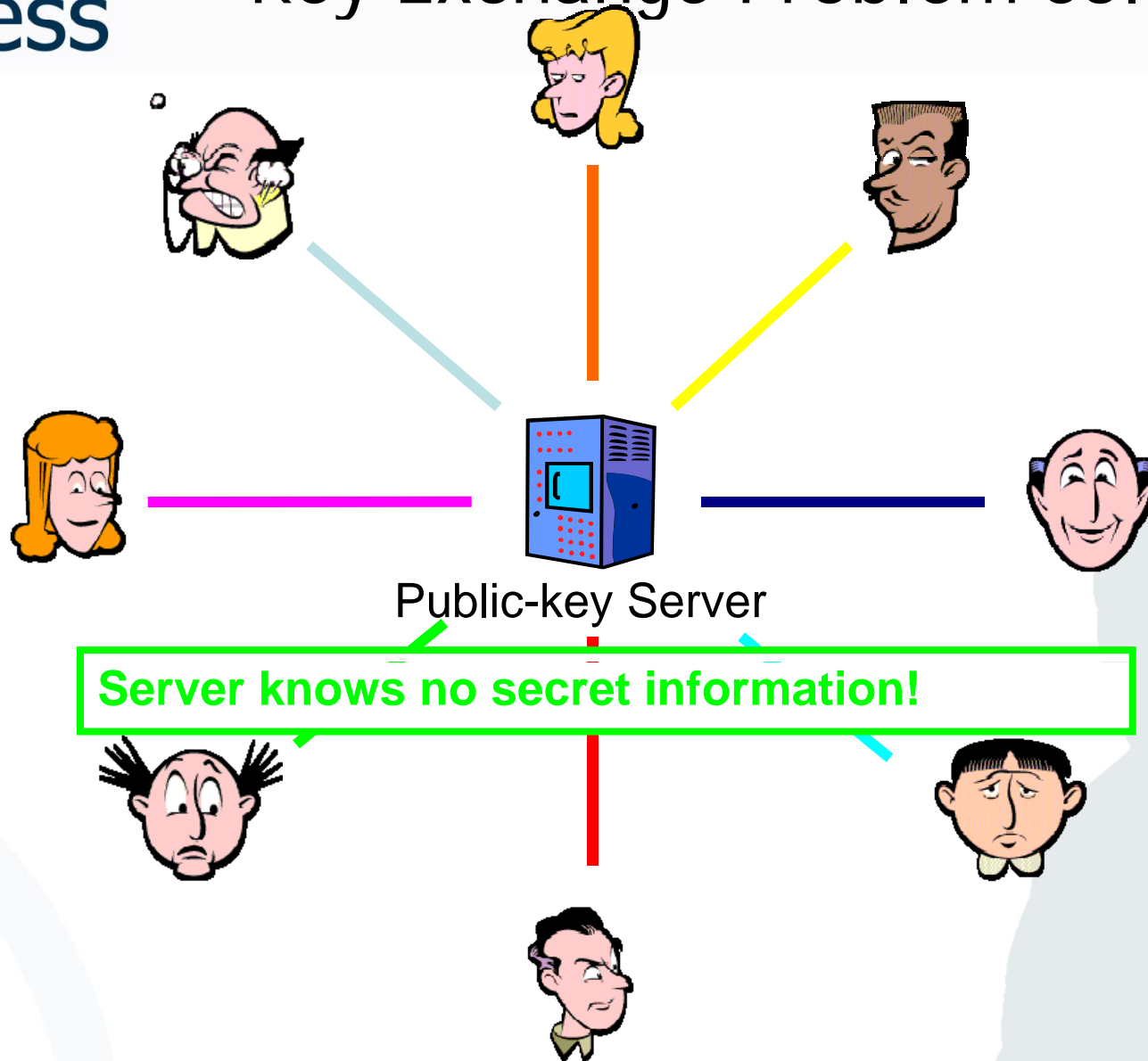
Problems of Symmetric Cryptosystems: Key Exchange






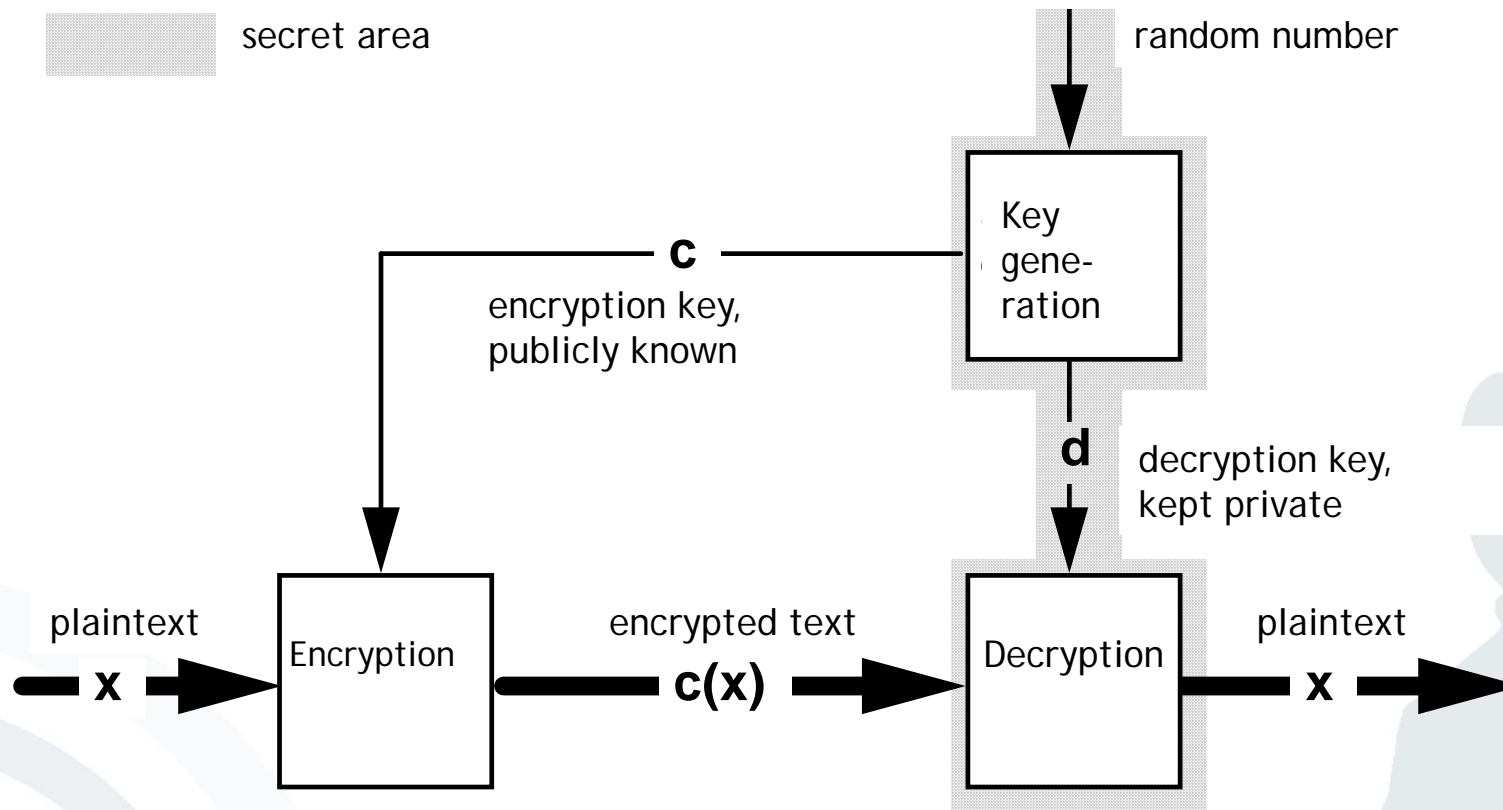


Key Exchange Problem Solved!




- Introduction
 - Classical cryptosystems
 - Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Encryption Summary
- 

- Use of key pairs instead of one key
 - public key is solely for encryption
 - Key text can only be decrypted with the corresponding private (undisclosed) key.
- Private key cannot be calculated from the public key.
- The public key can be distributed freely, even via insecure ways (e.g. directory (*public key crypto system*))
- Messages are encoded via the public key of the addressee.
- Only the addressee possesses the private key for decoding.



box with slot, access to messages only with a key

- Introduction
 - Classical cryptosystems
 - Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Encryption Summary
- 

- RSA
 - Rivest, Shamir, Adleman, 1978
 - is 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 typically 1024 bit, today rather 2048
- Diffie-Hellman
 - Diffie, Hellman, 1976, first patented algorithm with public keys
 - allows the exchange of a secret key
 - is 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
- Encryption Summary

- 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 cyphertext C .

- 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 d can be derived from e .

- You then pick an integer d to be a large, random integer which is relatively prime to $(p-1)*(q-1)$.
- That is, check that d satisfies:
$$\gcd(d, (p-1)*(q-1))=1$$

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

Example for Generating keys

$$(1) \begin{array}{ll} p * q & = :n \\ p * q & = \text{prime} \end{array}$$

$$(2) \varphi(n) = (p-1) * (q-1)$$

(3) pick e prime
 e and $\varphi(n)$ relat. prime
 prefer.: 3, 17 oder $65537 (2^{16} + 1)$

$$(4) \text{ calc. } d: e * d = 1 \text{ mod } \varphi(n)$$

$$(5) \text{ KEncrypt} = (e, n)$$

$$\text{KDecrypt} = (d, n)$$

Example

$$p = 47$$

$$q = 59$$

$$n = 2773$$

$$M = 5$$

$$46 * 58 = 2668$$

$$e = 17$$

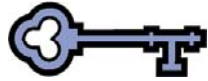
$$d = 157$$

$$M^e \text{ mod } n = C$$

$$5^{17} \text{ mod } 2773 = 508$$

$$C^d \text{ mod } n = M$$

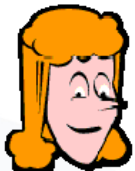
$$508^{157} \text{ mod } 2773 = 5$$



Public
(e,n)



Private
(d,n)



Alice

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

- Bob wants to send Alice the message „Hello World“
- 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

- 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

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
 - Encryption Summary
- 

Algorithm	Performance*
El Gamal	1826 s
RSA	16 s

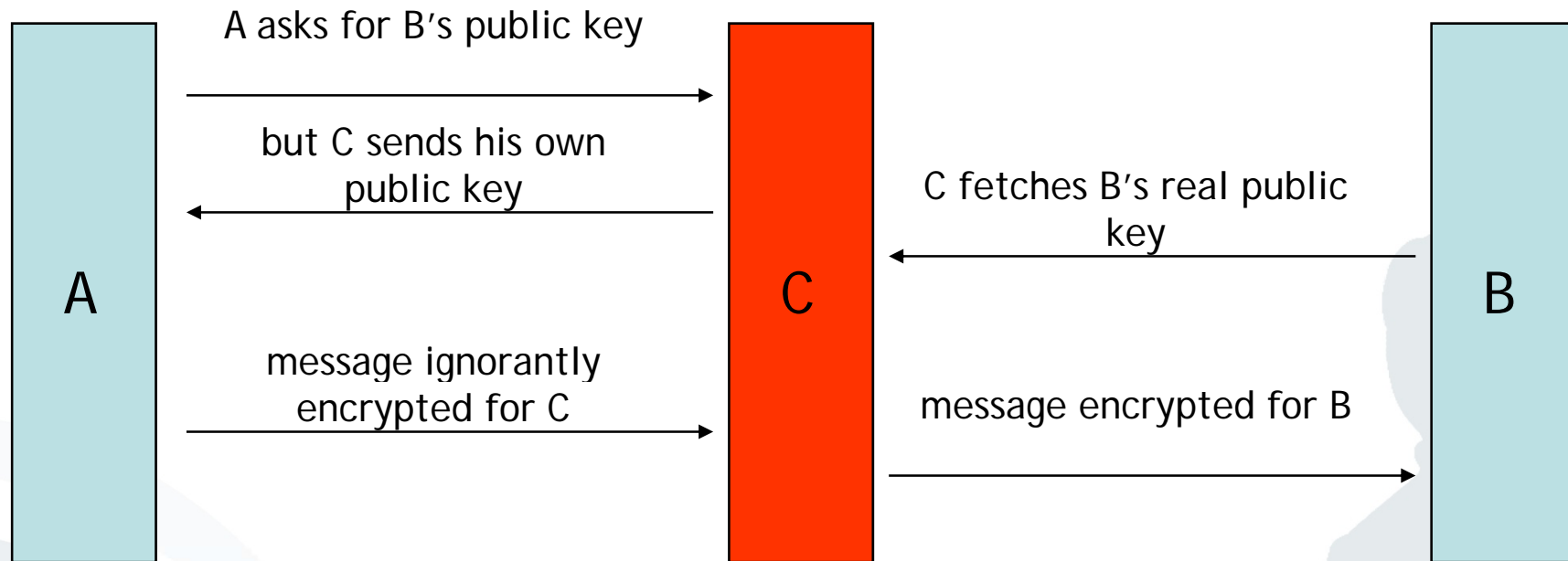
Disadvantage: Complex operations
with very big numbers

⇒ **Algorithms are very slow**

*) Encryption of 1 MB-blocks with an Athlon 1GHz processor

- Introduction
 - Classical cryptosystems
 - Public key cryptography
 - General concept
 - Algorithms
 - Hybrid systems
 - Key management
 - Encryption Summary
- 

“Man in the middle attack”

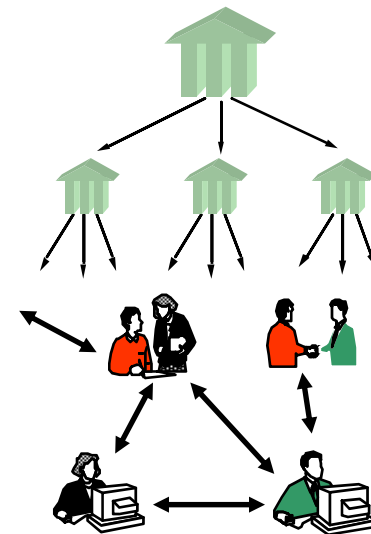
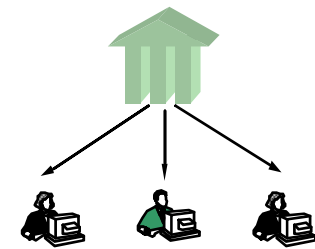


- ⇒ Keys are certified, that means a third person/institution confirms (with its digital signature) the affiliation of the public key to a person

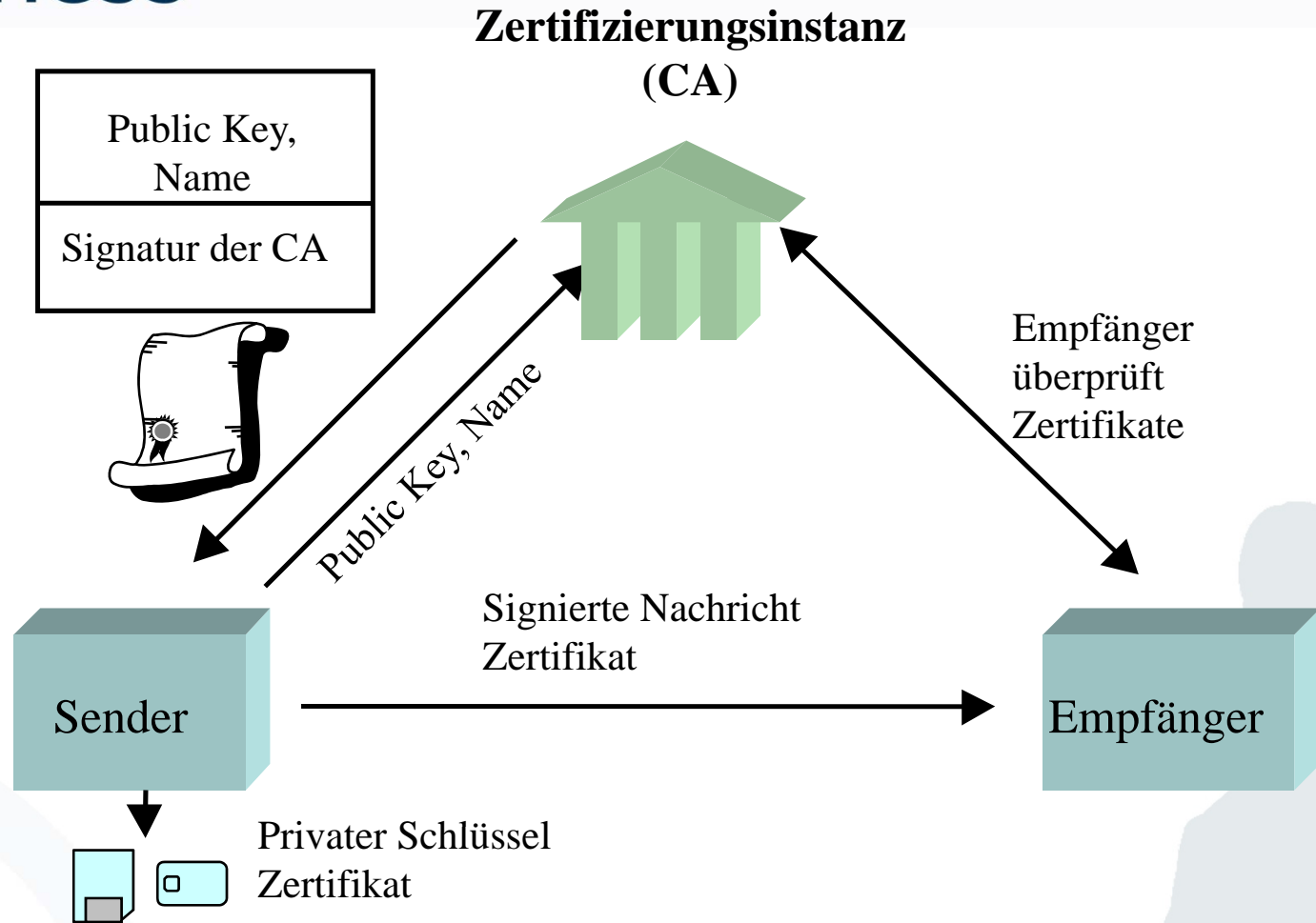
- B can freely distribute his own public key.
- But: Everybody (e.g. C) could distribute a public key and claim that this one belongs to B.
- If A uses this key to send a message to B, C will be able to read this message!
- Thus:
How can A decide if a public key was really created and distributed by 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?):

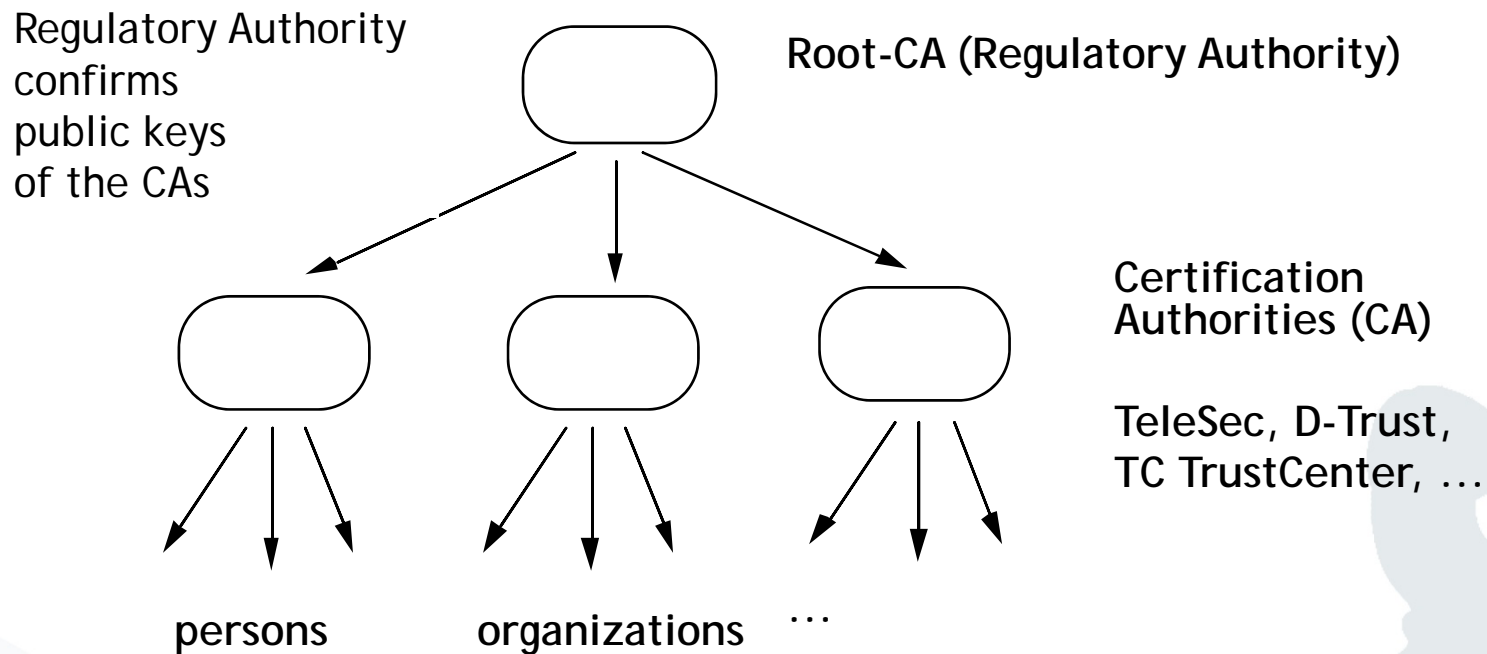
- **central certification authority (CA)**
 - a single CA, keys often integrated in checking software
 - example: older versions of Netscape (CA = Verisign)
- **hierarchical certification system**
 - CAs which in turn are certified by "higher" CA
 - examples: PEM, Teletrust, infrastructure according 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)



Zentrale Zertifizierungsinstanz



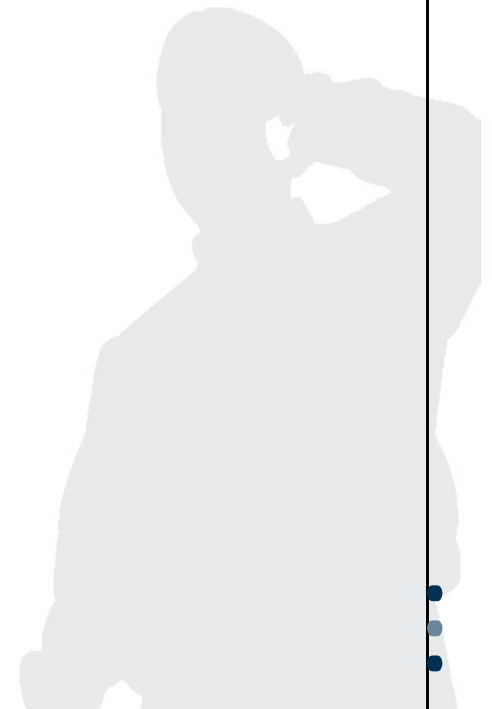
- Z.B. in Unternehmen



- 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.

- Integrity is ensured by using a Message Digest - containing elements of the message (One-Way Hash Function; e.g. MD5).
- Assuring Identity with a Digital Signature. Digital Signatures are produced using the Private Key and can be verified using the Public Key.
- Option: Digitally signed time stamps (e.g. used for certifying temporally limited offers).
- Example Public-Key-Systems:
 - Diffie-Hellman: Discrete Logarithm
 - RSA (Rivest - Shamir - Adleman): Factorisation Problem

- Easy to compute
- Hard to invert
- Example:
 - Telephone Book of a large City
 - Factorisation of large integers:
 - Easy: $P \times Q \rightarrow Y$
 - Hard: $Y \rightarrow P \times Q$



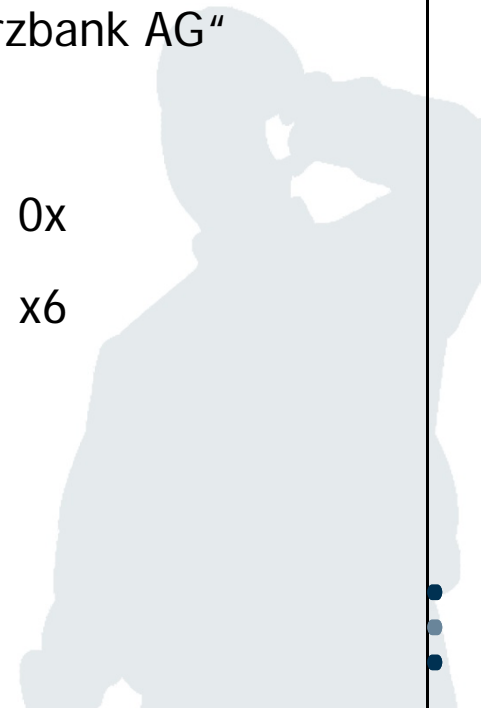
- Simple Algorithm:
„In a given Message count the number of Vowels, divide the sum by 3 and store the Rest of the Division in the „Decade“ of the „cryptographic check number“ .
Then count the number of Consonants, divide the sum by 10 and store the Rest of the Division in the „Unit Position“ of the „cryptographic check number“ .“
- Example:
The Message „Martin Reichenbach is a Member of Commerzbank AG“ contains 15 Vowels und 26 Consonants.

15 divided by 3 gives 5 Rest 0
(math.: $15 \bmod 3 = 0$)
26 divided by 10 gives 2 Rest 6
(math.: $26 \bmod 10 = 6$)

=> Decade: 0x

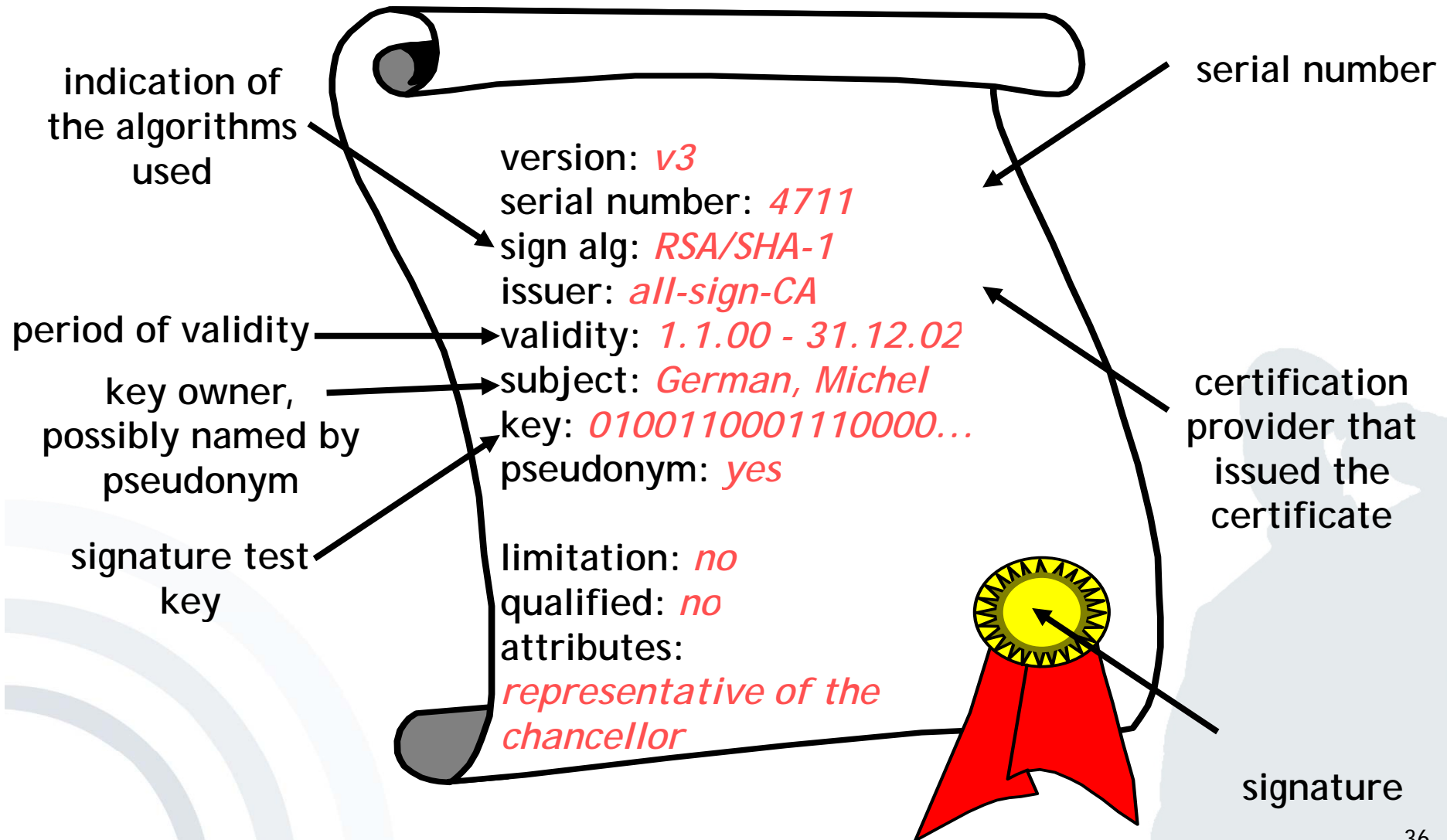
=> Unit Position: x6

Message-Hash: 06



Content of a Key Certificate

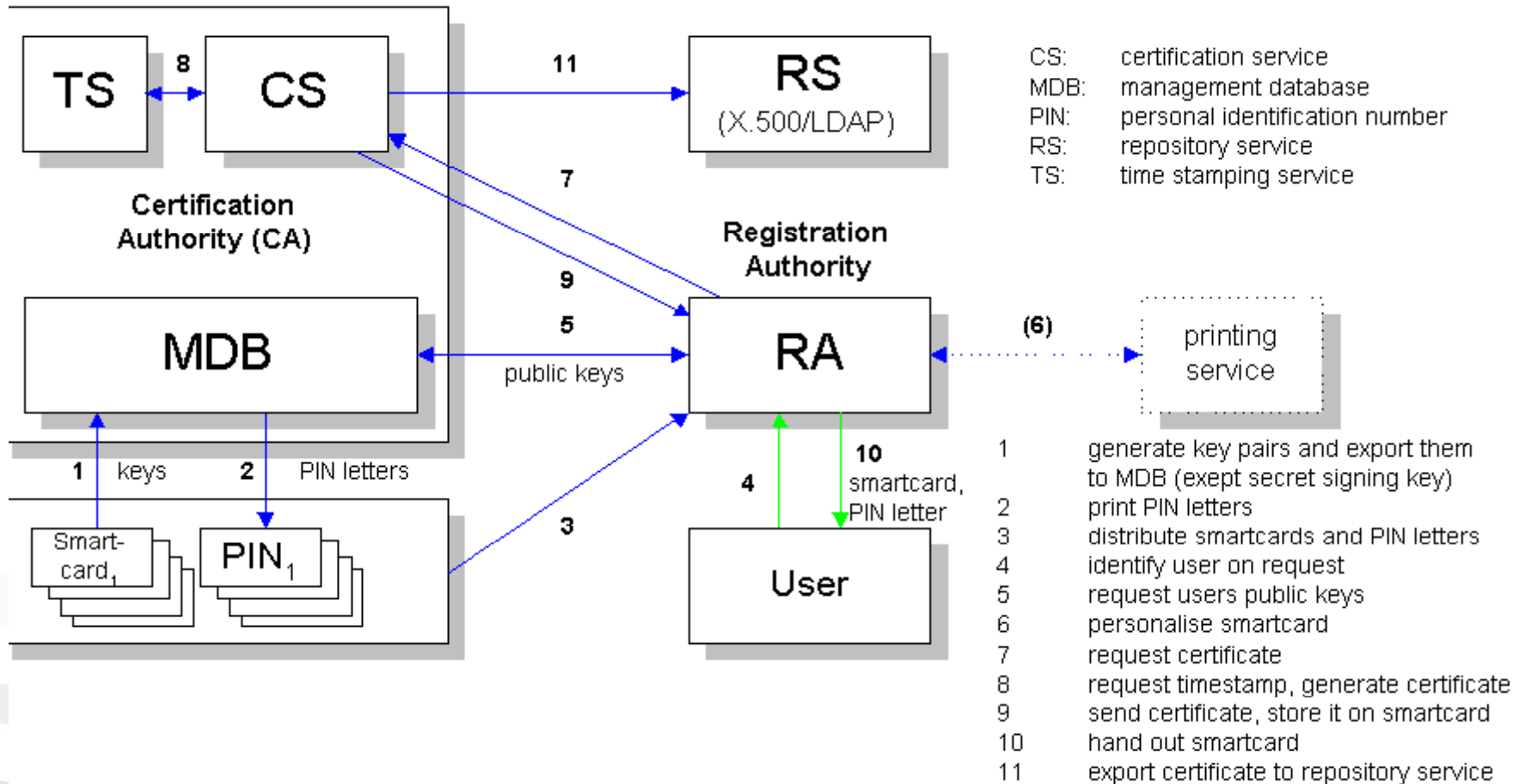
(according to German Signature Law and Regulation)



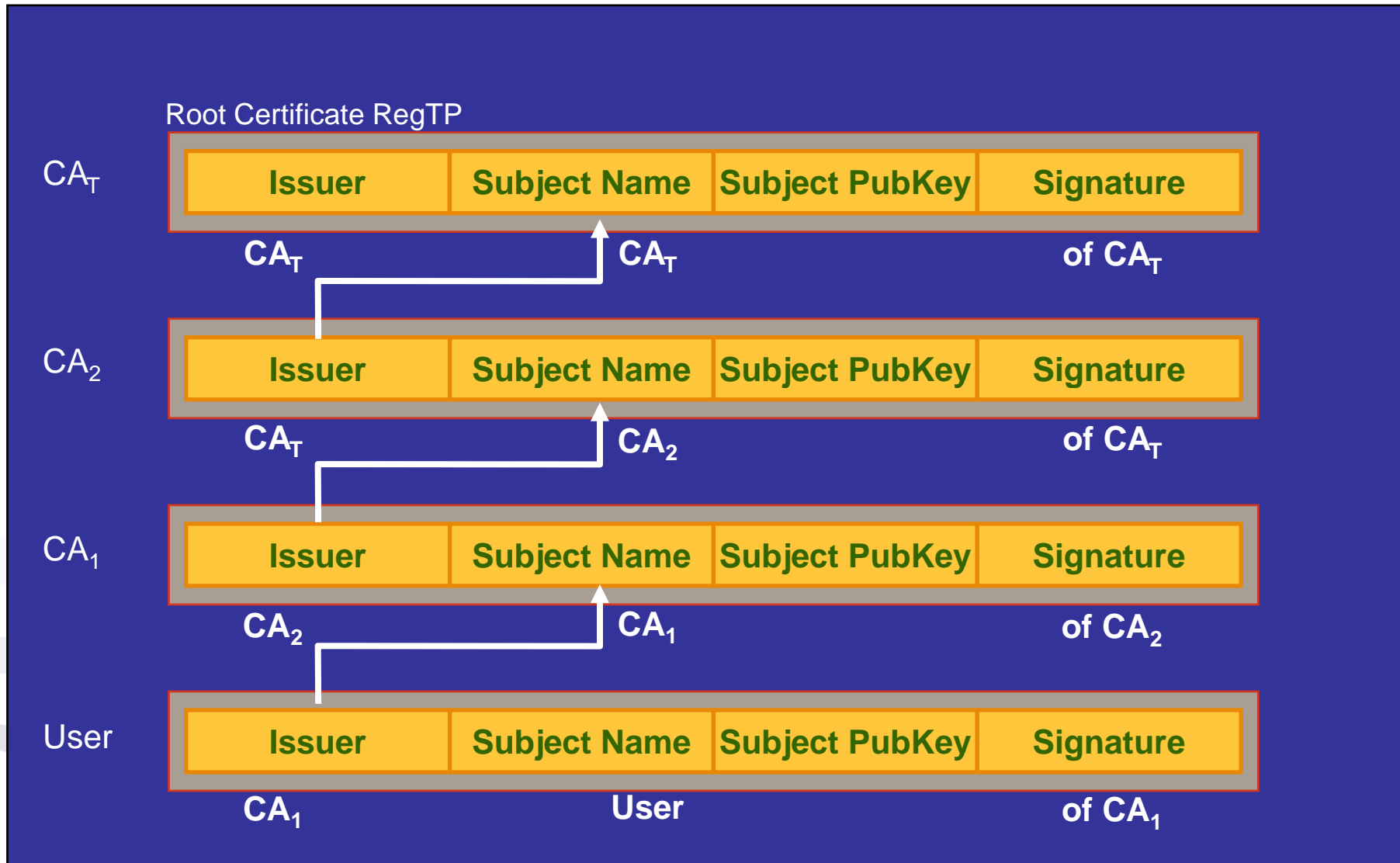
Tasks of a Certification Authority

(according to German Signature Law and Regulation)

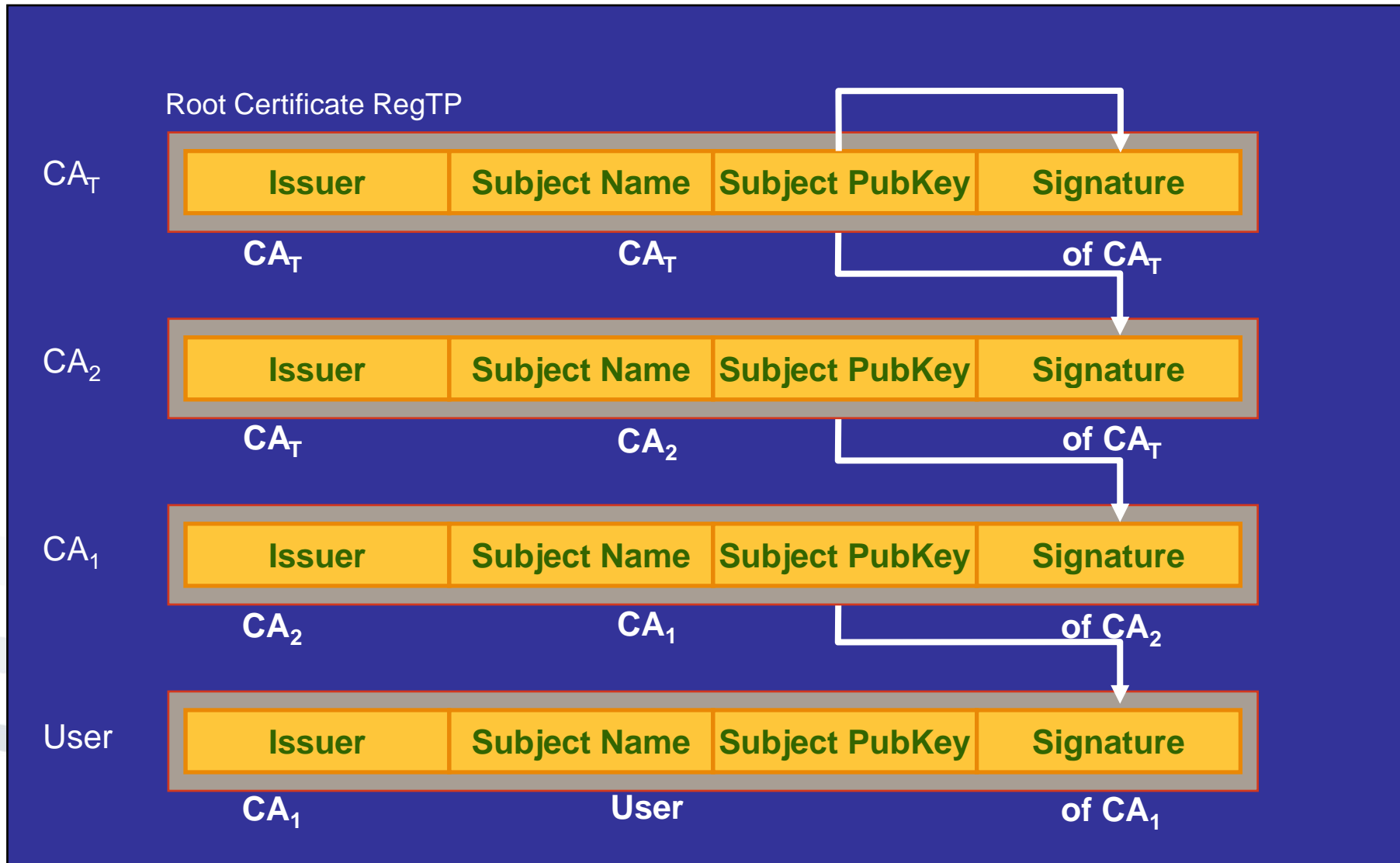
- reliable identification of persons who apply for a certificate
- information on necessary methods for fraud resistant creation of a signature
- provision for secure storage of the private key
 - at least Smartcard (protected with PIN)
- publication of the certificate (if wanted)
- barring of certificates
- if necessary emission of time stamps
 - for a fraud resistant proof that an electronic document has been at hand at a specific time



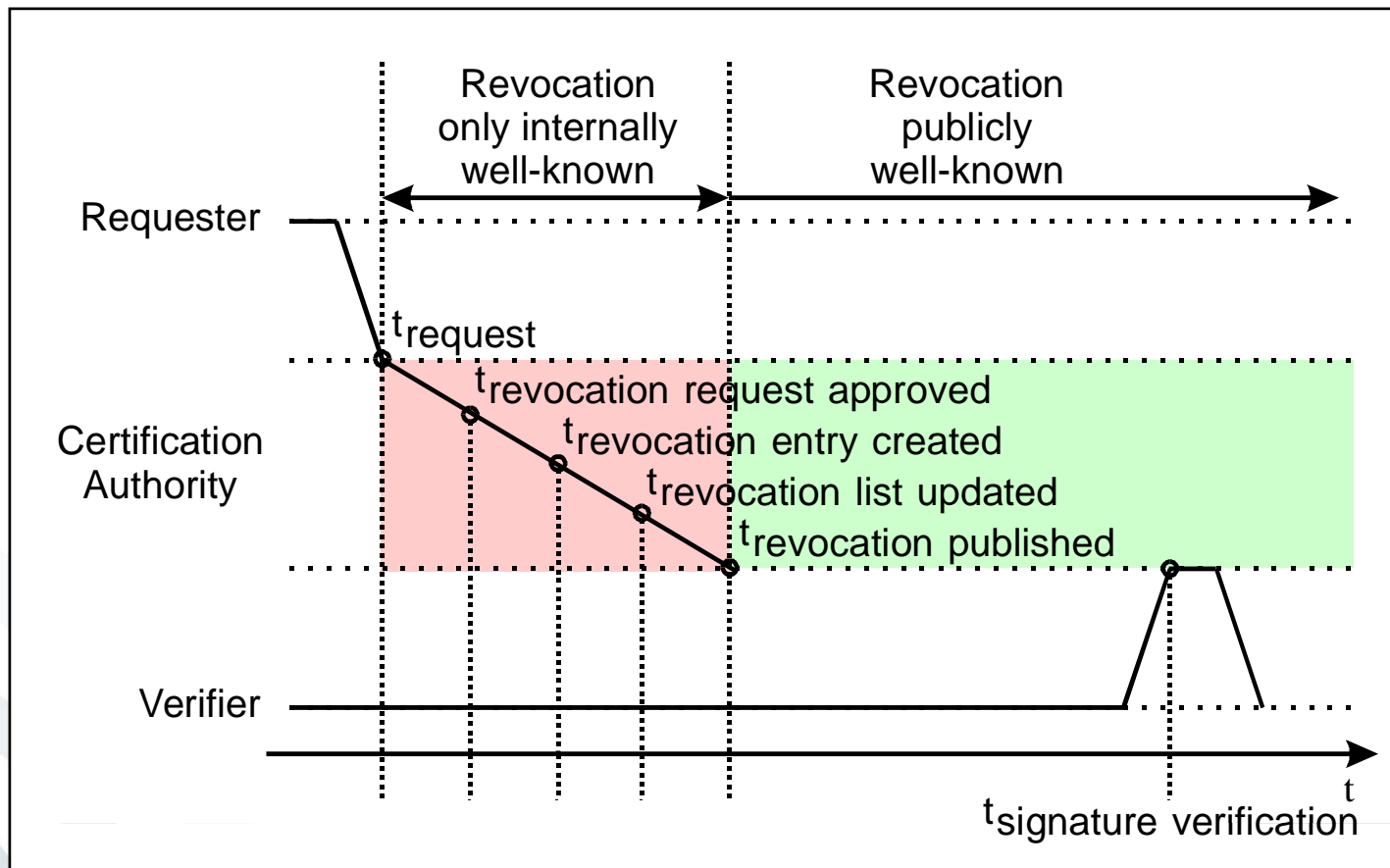
- checking of the following items by certain confirmation centers (BSI, TÜVIT, ...)
 - concept of operational security
 - reliability of the executives and of the employees as well as of their know-how
 - financial power for continuous operation
 - exclusive usage of licensed technical components according to SigG and SigV
 - security requirements as to operating premises and their access controls
- possibly license of the regulation authority



Verifying the Certificate Path



- A Signature's Validity is time-dependant!



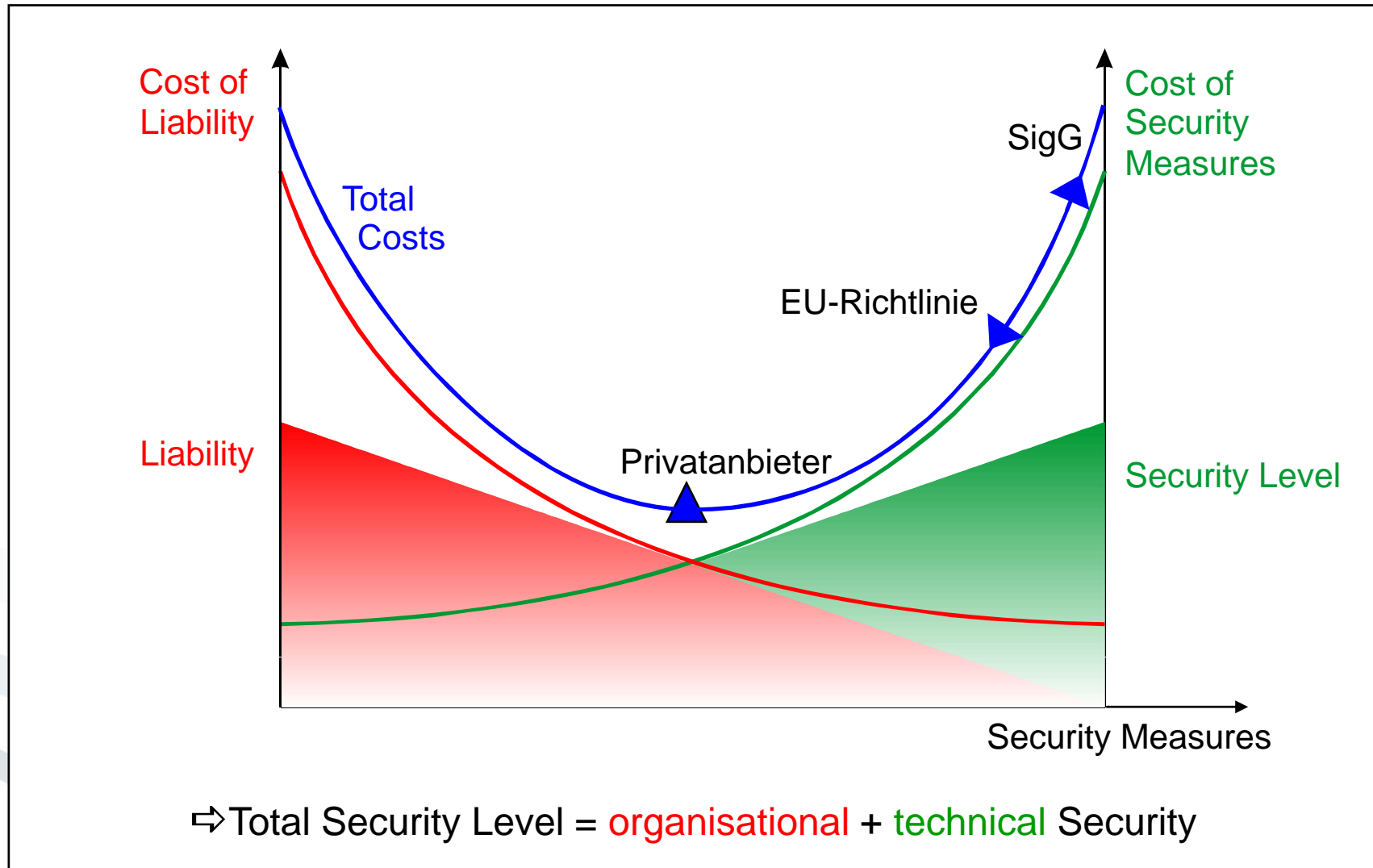
Source: Bertsch, Freiburg, 2000

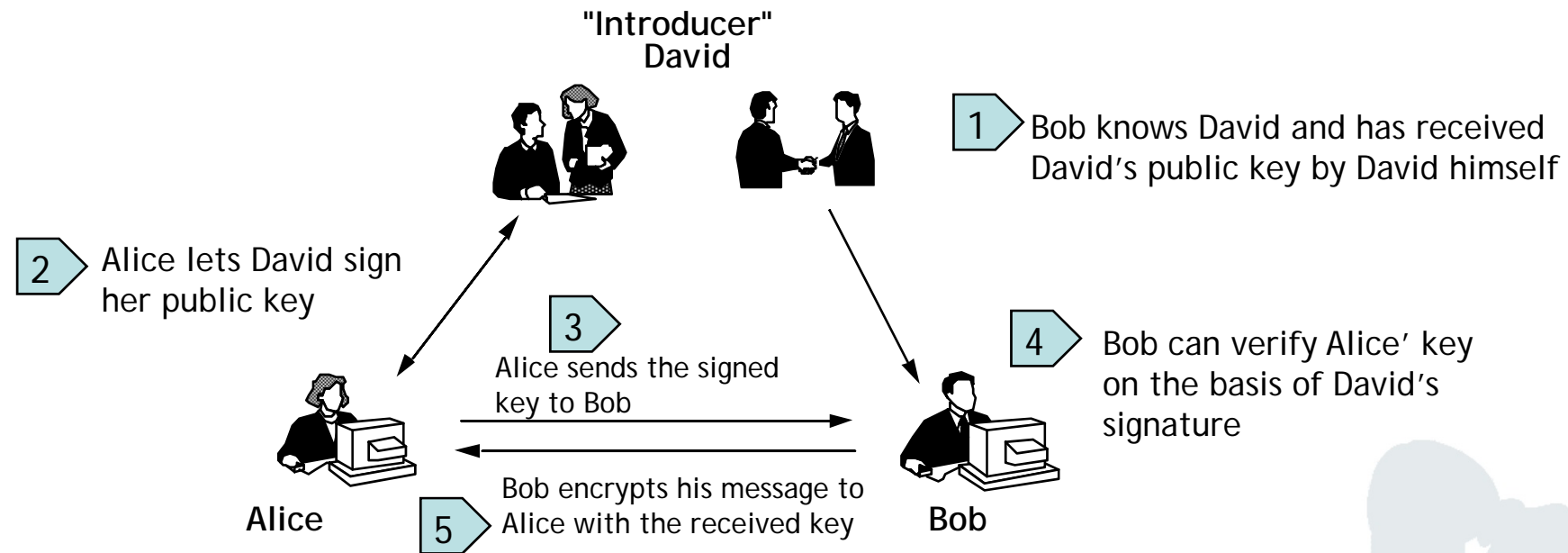
- SigG: Certificates valid 5 Years, verifiable for 30 Years
- Asymmetric Krypto-Mechanisms rely on so far unsolved mathematical problems (RSA: Factorisation Problem)
- Problem might be solved tomorrow
 - New Algorithms for efficiently Solving the Factorisation Problem



- Secure Visualisation
 - „What you see is what you sign“
 - „What you verify is what you see“
- Secure Components for Signature Creation

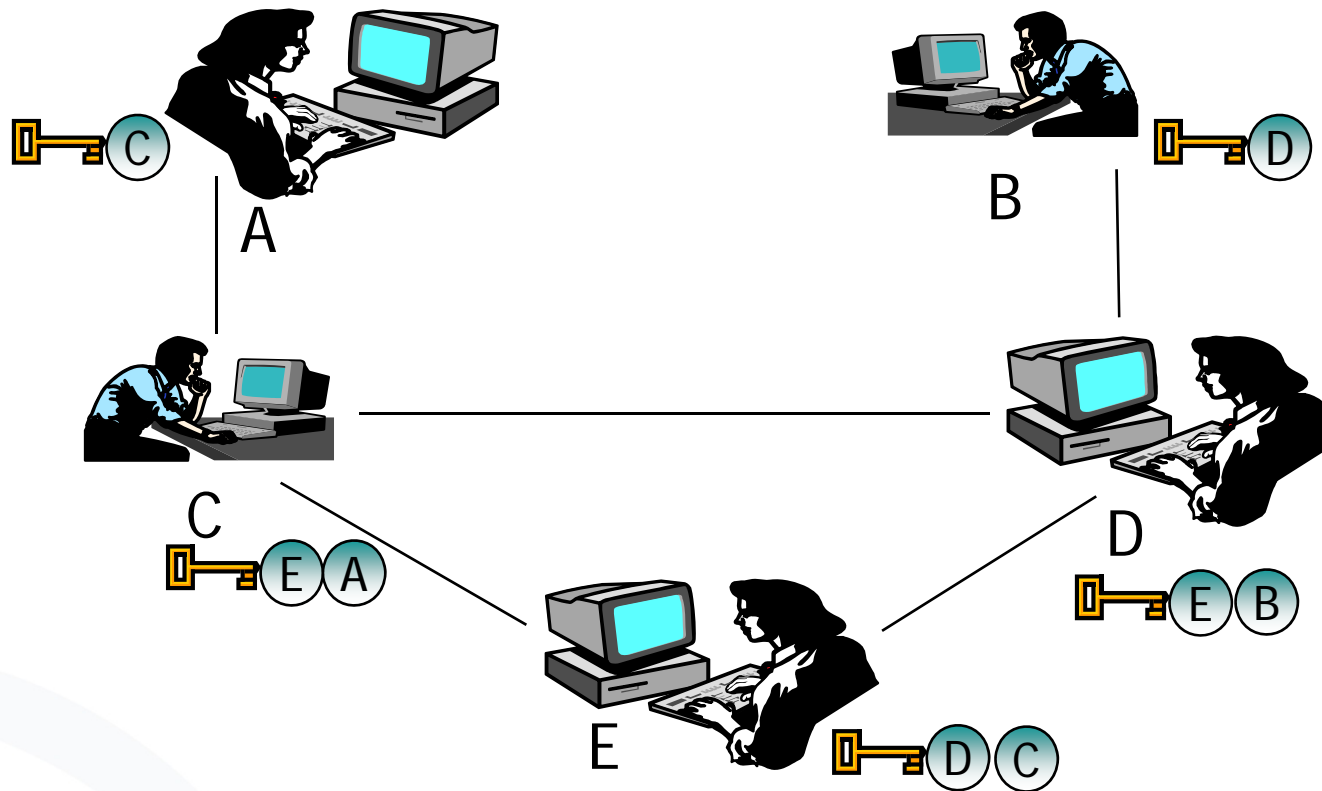






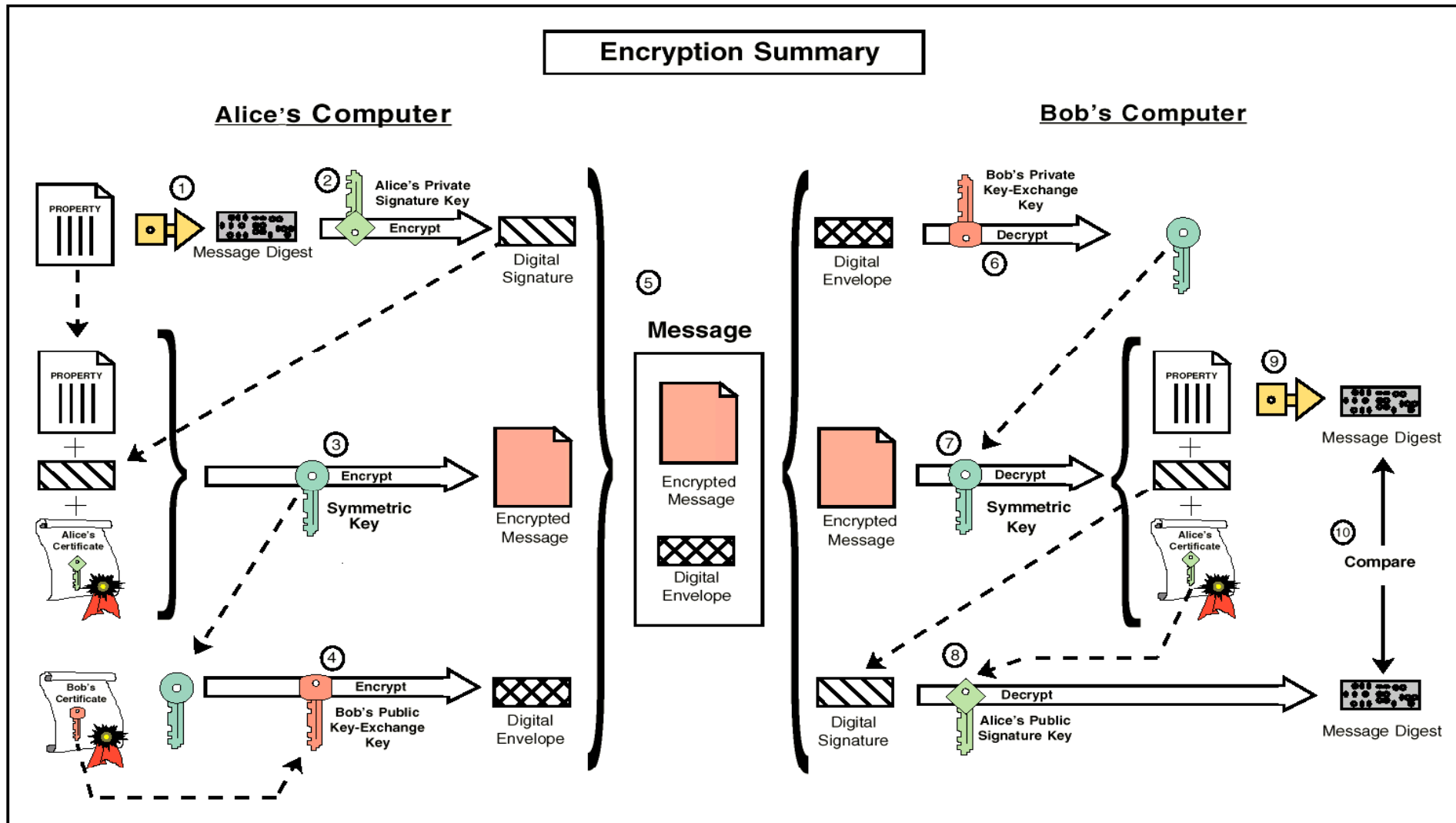
- Each user can act as a "CA".
- Mapping of the social process of creation of trust.
- Keys are "certified" through several signatures.
- Expansion is possible by public key servers and (hierarchical) CAs.

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
 - Encryption Summary

- 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