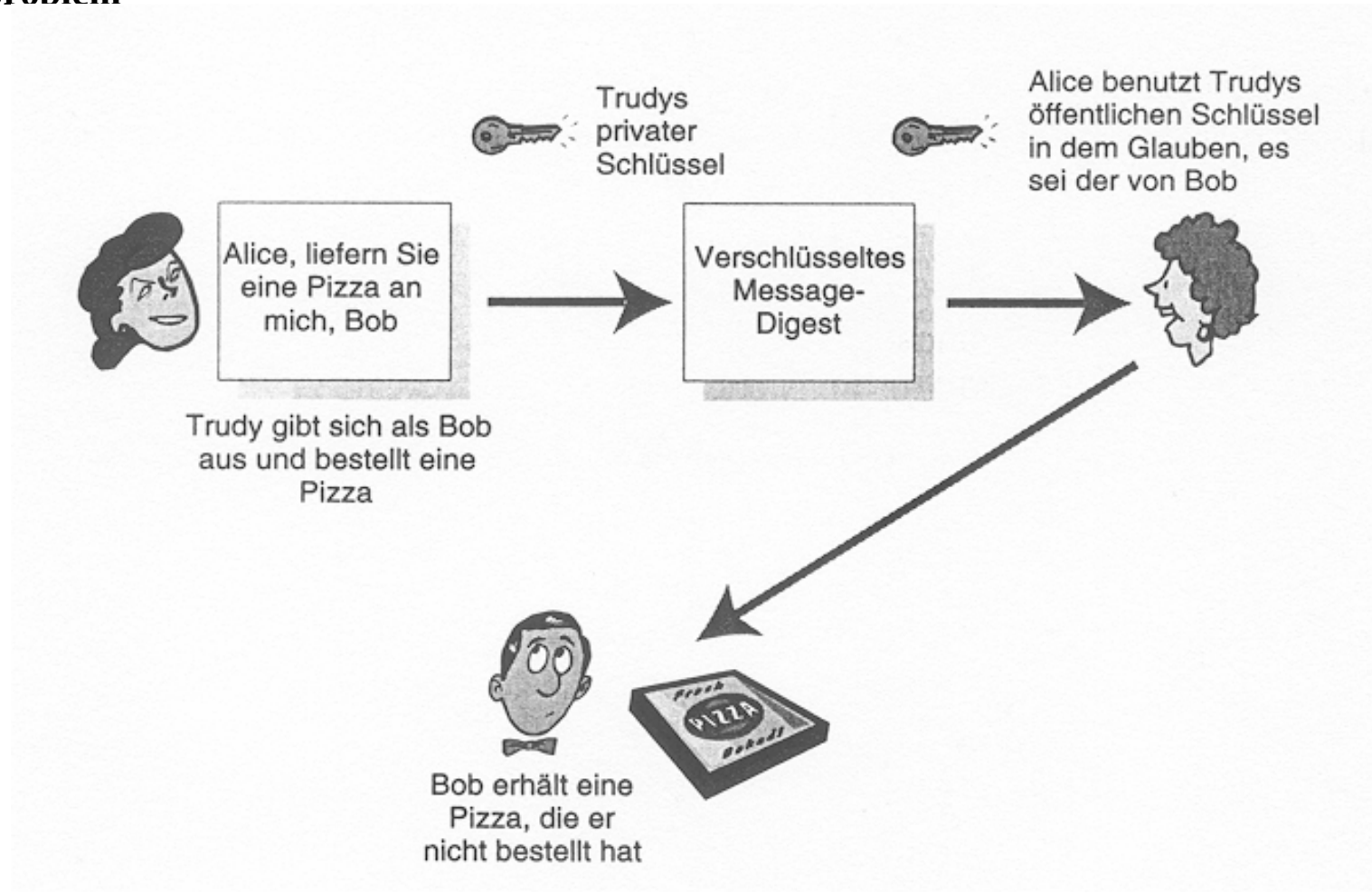
# Key Management (Distribution and Certification) (1)

Remaining problem of the public key approach:
How to ensure that the public key received is really the one of the sender?

**Illustration of the problem**



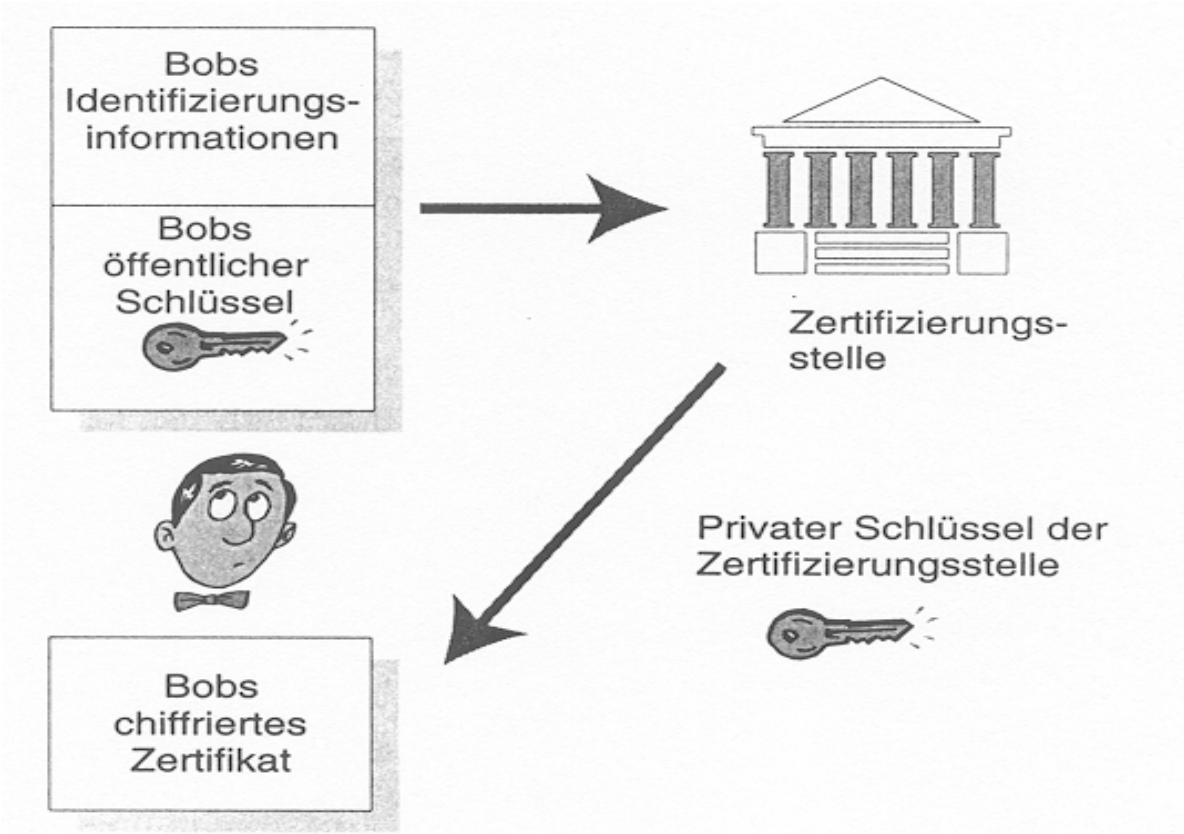
# Key Management (Distribution and Certification) (2)

Solution: Using a trustworthy third person, the so-called

*Certification Authority (CA)*

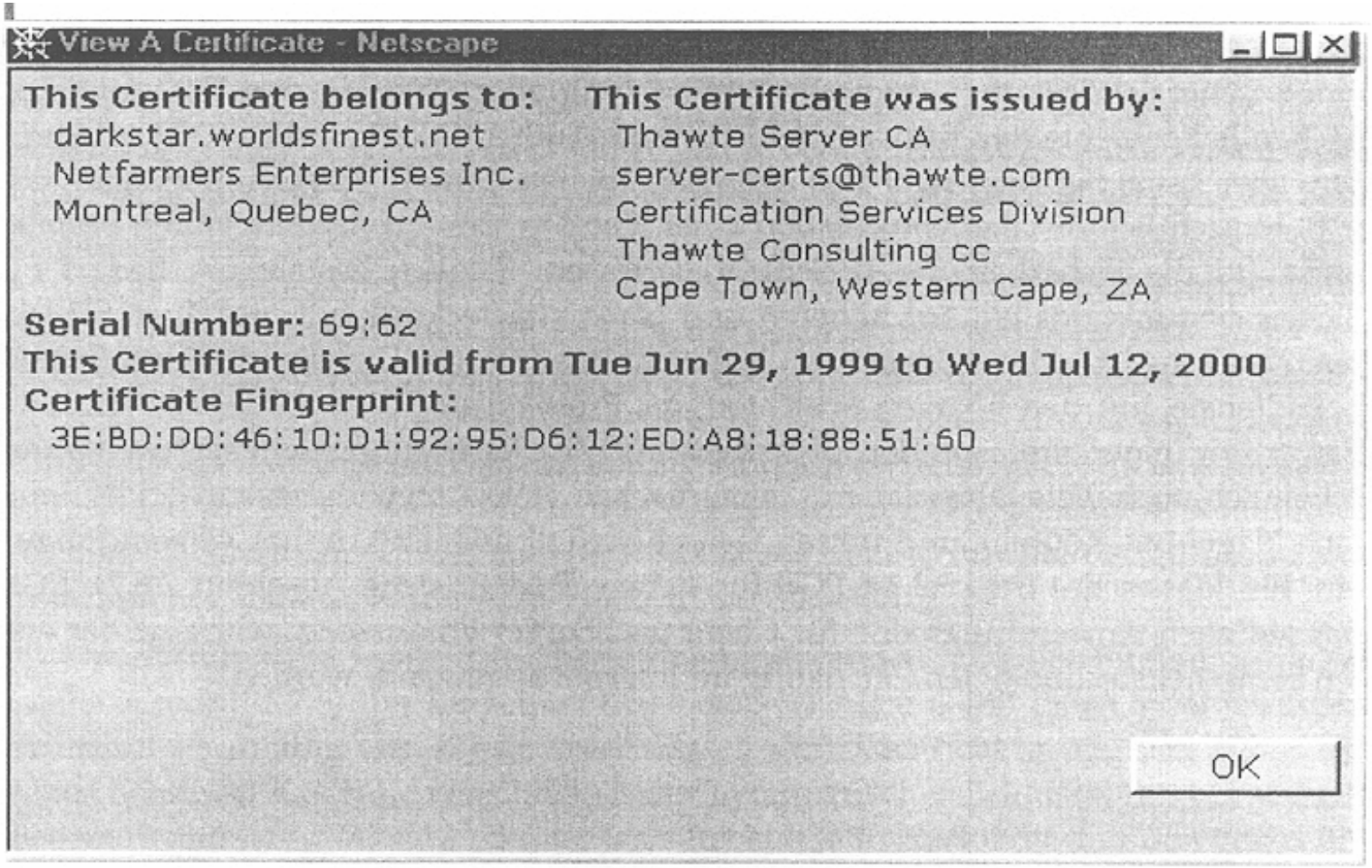
Idea: CA checks the identity of public key holders and creates a *certificate* which binds the key to the correct holder and is digitally signed by the CA.

**Job of the CA**



# Key Management (Distribution and Certification) (2a)

## Example of a Certificate



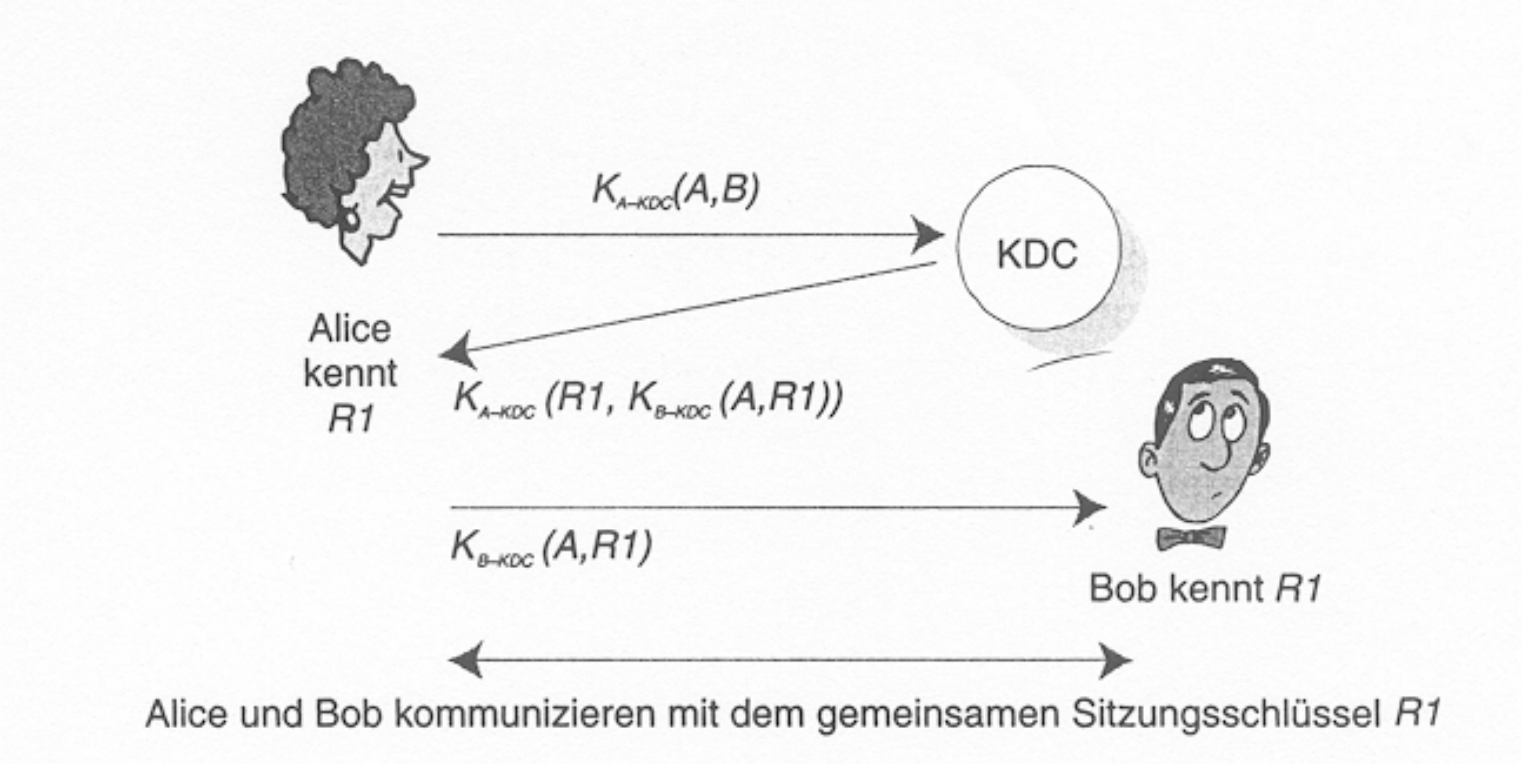
# Key Management (Distribution and Certification) (3)

Remaining problem of the symmetric key approach:
How to agree a priori, i.e. before the secure communication between two partners starts, on their secret key?
Solution: Using a trustworthy third person, the so-called

*Key Distribution Center (KDC)*

Idea: KDC is a server having a secret key with each registered user of the system. Key distribution (session key management) and authentication now goes through the KDC.

**Creating and distributing a unique session key between Alice and Bob via the KDC**



# Key Management (Distribution and Certification) (4)

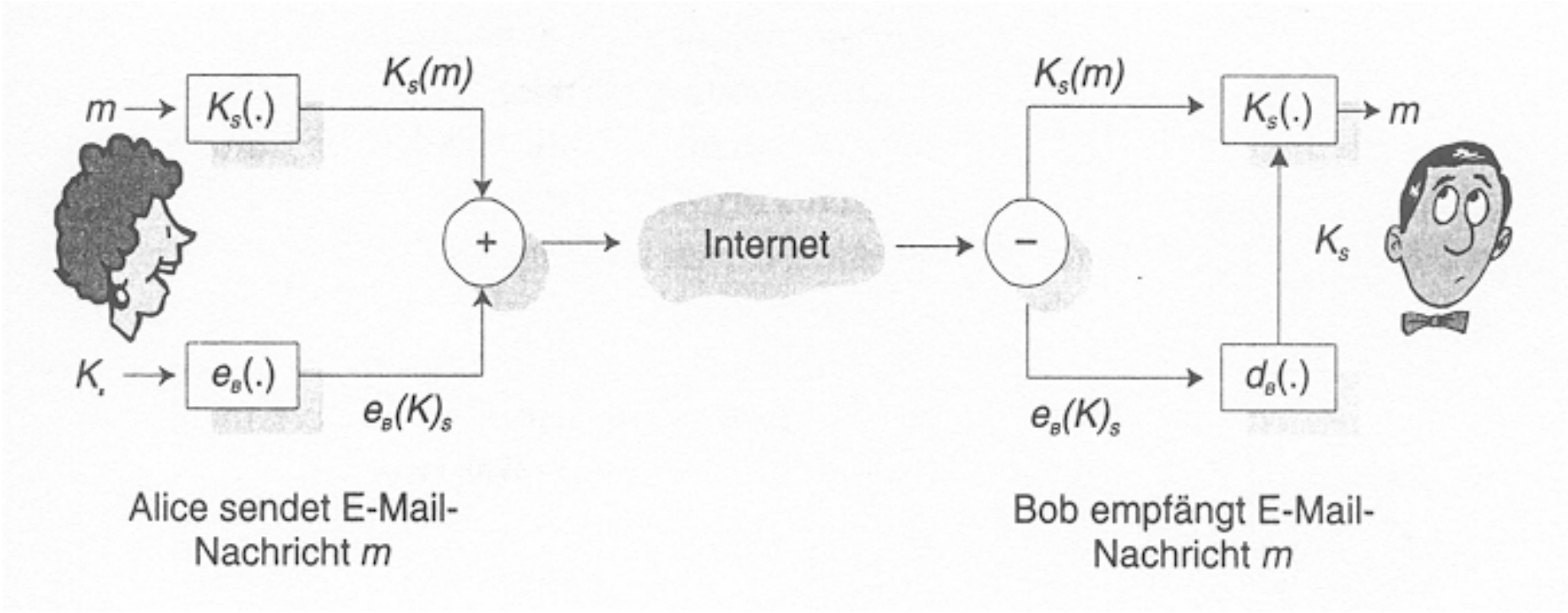
Often used in networked operating systems to control the access to shared resources like, e.g., file servers

**Example: Kerberos**

- named after a multiheaded dog in Greek mythology guarding the entrance to Hades
- designed at MIT to allow workstation (end) users to access network resources (servers) in a secure way, i.e. when accessing users are authenticated and checked whether they have adequate access rights
- the so-called Authentication Server (AS) takes the role of the KDC
- used in many real operating systems (Windows, Unix)
- in addition to the general approach addressed before:
  - a so-called *ticket* now contains, in addition to Alice's name and R1, a timestamp marking the time-out of the communicated session key
  - Alice sends the ticket to Bob together with a Nonce encrypted with R1.
  - Bob sends back to Alice the incremented Nonce encrypted with R1 authenticating himself to Alice.

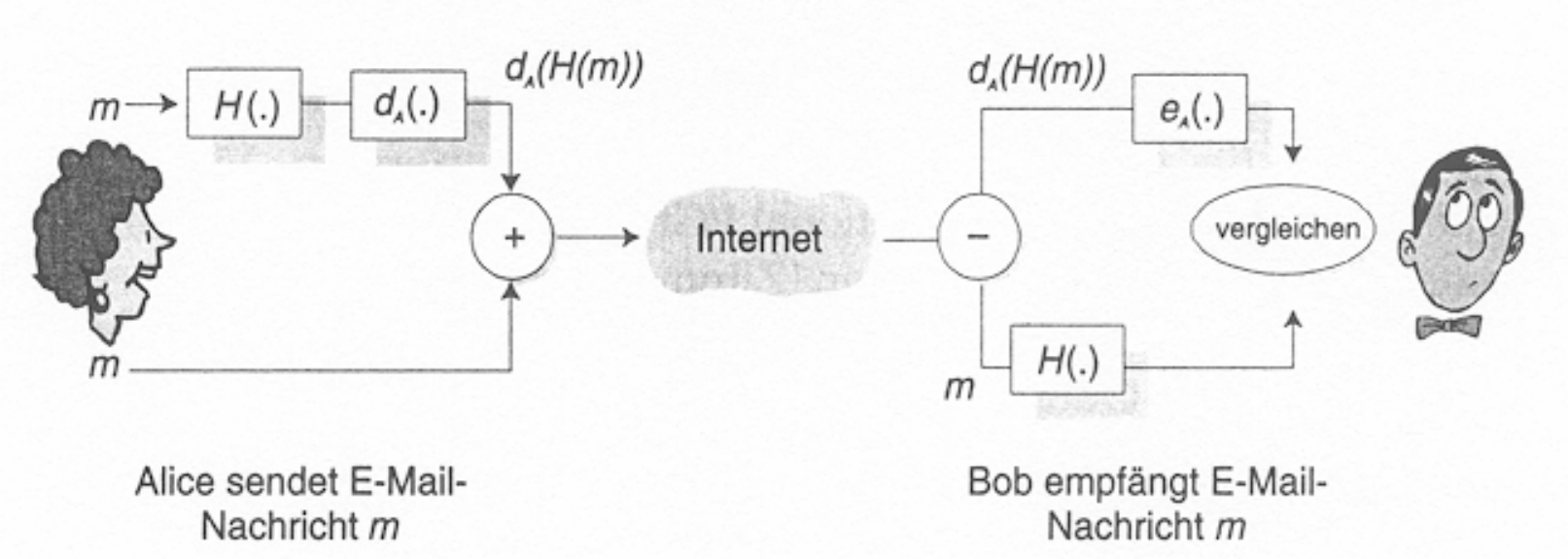
# Application layer: Secure E-Mail (1)

**Secrecy approach: Symmetric session key encrypted by RSA (public key algorithm)**

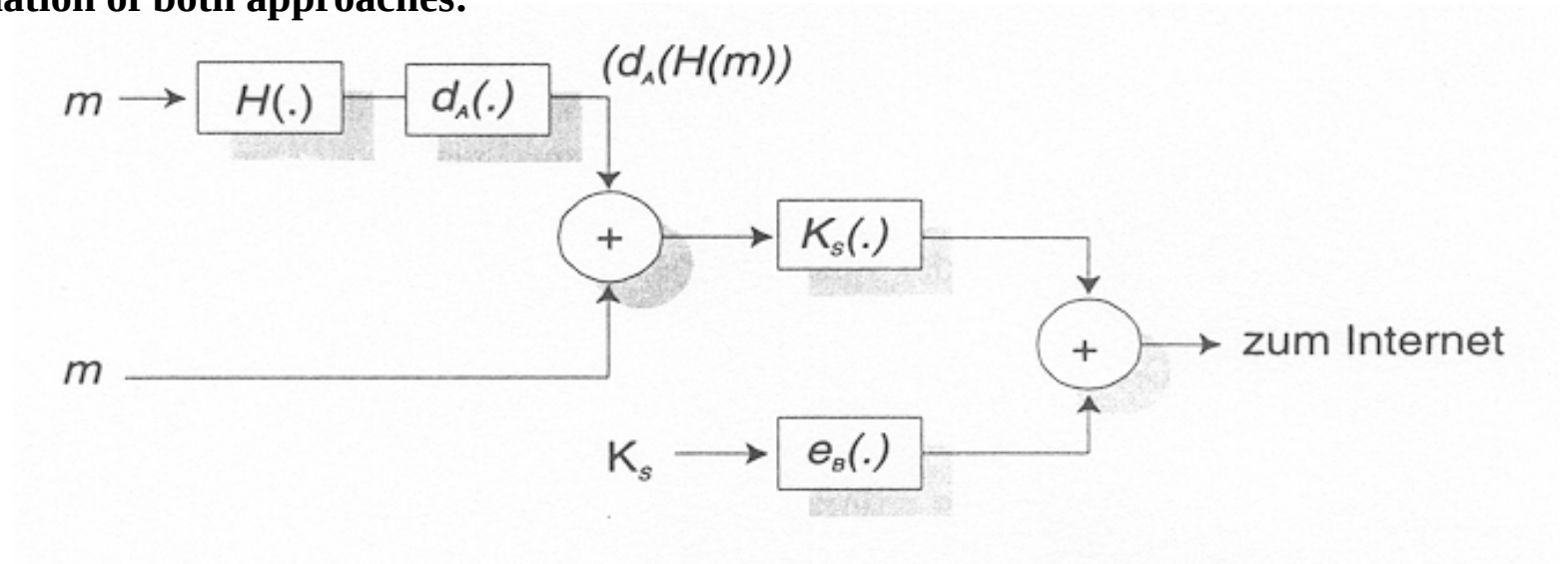


# Application layer: Secure E-Mail (2)

**Approach to sender authentication and integrity: Message Digests and digital signature**




**Combination of both approaches:**



# Application layer: Secure E-Mail (3)

**De facto Standard: PGP (Pretty Good Privacy)**
reflects in principle the approach just described

**A message signed with PGP**

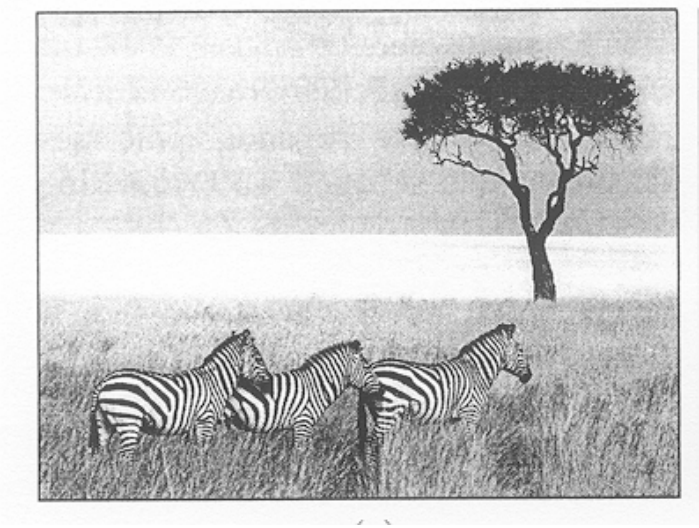
```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1
Bob:
My husband is out of town tonight.
Passionately yours, Alice
-----BEGIN PGP SIGNATURE-----
Version: PGP for Personal Privacy 5.0
Charset: noconv
yhHJRHhGJGhgg/12EpJ+lo8gE4vB3mqJhFEvZP9t6n7G6m5Gw2
-----END PGP SIGNATURE-----
```

**A secret PGP message:**

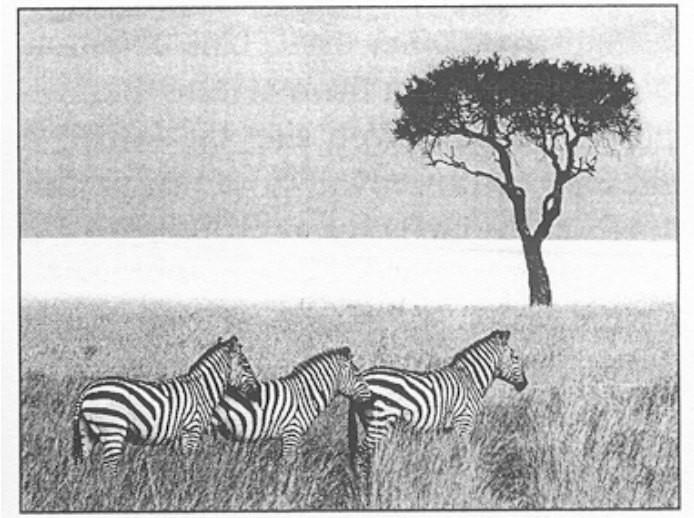
```
-----BEGIN PGP MESSAGE-----
Version: PGP for Personal Privacy 5.0
u2R4d+/jKmn8Bc5+hgDsqAewsDfrGdszX68liKm5F6Gc4sDfcXyt
RfdS1OjuHgbcfDssWe7/K=1KhnMikLoO+1/BvcX4t==Ujk9PbcD4
Thdf2awQfgHbnmKlok8iy6gThlp
-----END PGP MESSAGE-----
```

# Steganography

**Three zebras and a tree**



**Three zebras, a tree, and the complete text of five plays by Shakespeare**



# Transport layer: E-Commerce (1)

**Typical scenario for Internet-Commerce**

- Bob is surfing in the web and arrives at the site of Alice Inc.
- Bob can order by selecting the product, the desired quantity, giving his address, and his credit card number
- Somewhat later he receives the products by ordinary mail and the charge in his next card statement

Bad surprises could be among others:

- Trudy intercepts the order, obtains Bob's credit card information and uses them to purchase anything else
- Trudy is masquerading as Alice Inc., taking Bob's money and disappear.

**Solution I: Using Secure Socket Layers (SSL) - Protocol**

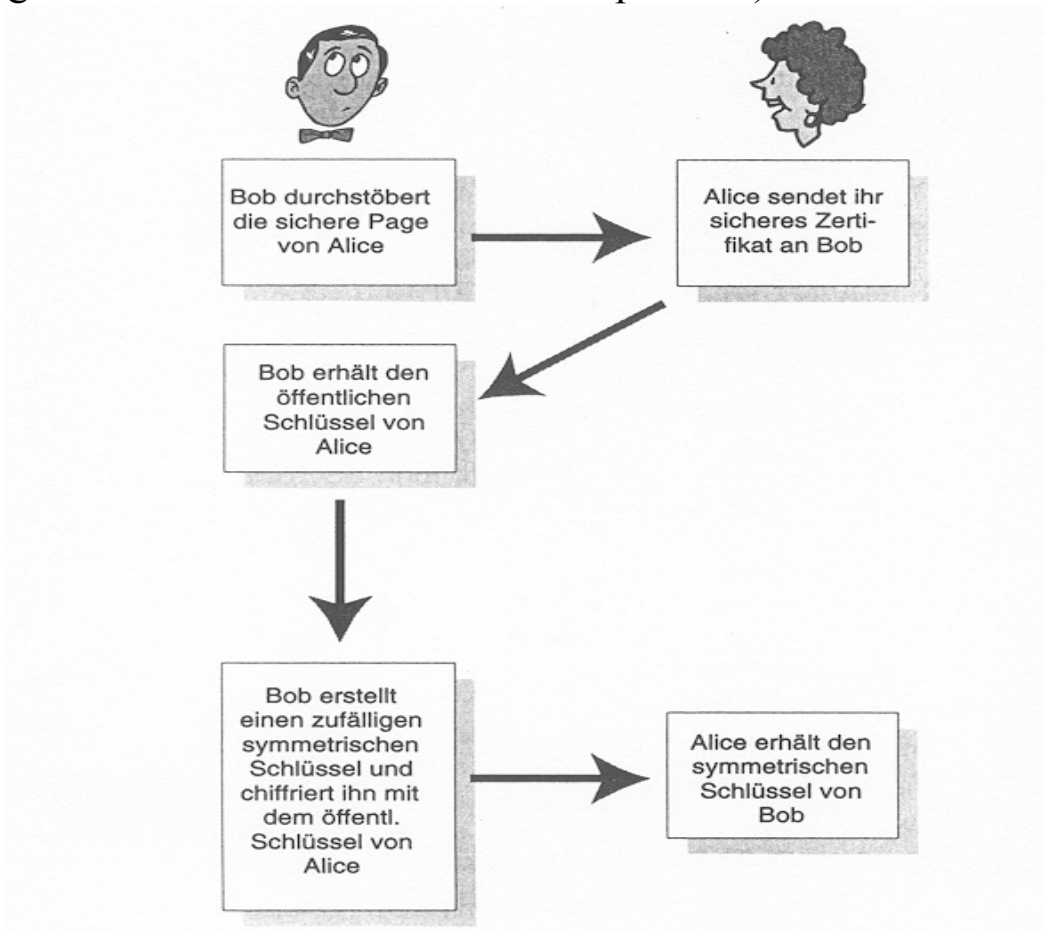
- originally developed by Netscape
- protocol providing data encryption and authentication between a Web client and a Web server
- widely used in internet commerce, being implemented in almost all popular browsers and Web servers
- constitutes the basis of the *TLS* (Transport Layer Security) protocol
- not limited to web applications
- sits between the application layer and the transport layer
  - on the sending side: receives messages (e.g. HTTP, IMAP), encrypts the data, sends it to a TCP socket
  - on the receiving side: reads from the TCP socket, decrypts the data, sends it to the application process

# Transport layer: E-Commerce (2)

**SSL provides the following features:**

- *SSL server authentication*, allowing a user to get confirmation about a server's identity
- *SSL client authentication*, allowing a server to get confirmation about a client's identity (optional)
- An *encrypted SSL session*, in which all information sent between browser and web server is encrypted by the sending software

**SSL handshake protocol** (executed before sending data between the communication partners)



# Security in WLAN (IEEE 802.11) (1)

Experiment undertaken:

Driving around the S.F. Bay area equipped with a laptop and a 802.11 card “looking” for wireless networks that were “visible” from outside the buildings

Results:

- more than 9000 of such networks were recorded
- one street corner in S.F. offered 6(!) different available networks
- 85% of the 9000 did not utilize the WEP (Wired Equivalent Privacy) protocol

**The IEEE 802.11 WEP (Wired Equivalent Privacy) protocol**

Objective:

Making a WLAN as secure as a LAN.

It provides authentication and data encryption between a host and the base station (AP:= Access Point) using a symmetric shared key approach

- no key management algorithm
- host and AP must agree on the key “out-of-protocol”

# Security in WLAN (IEEE 802.11) (2)

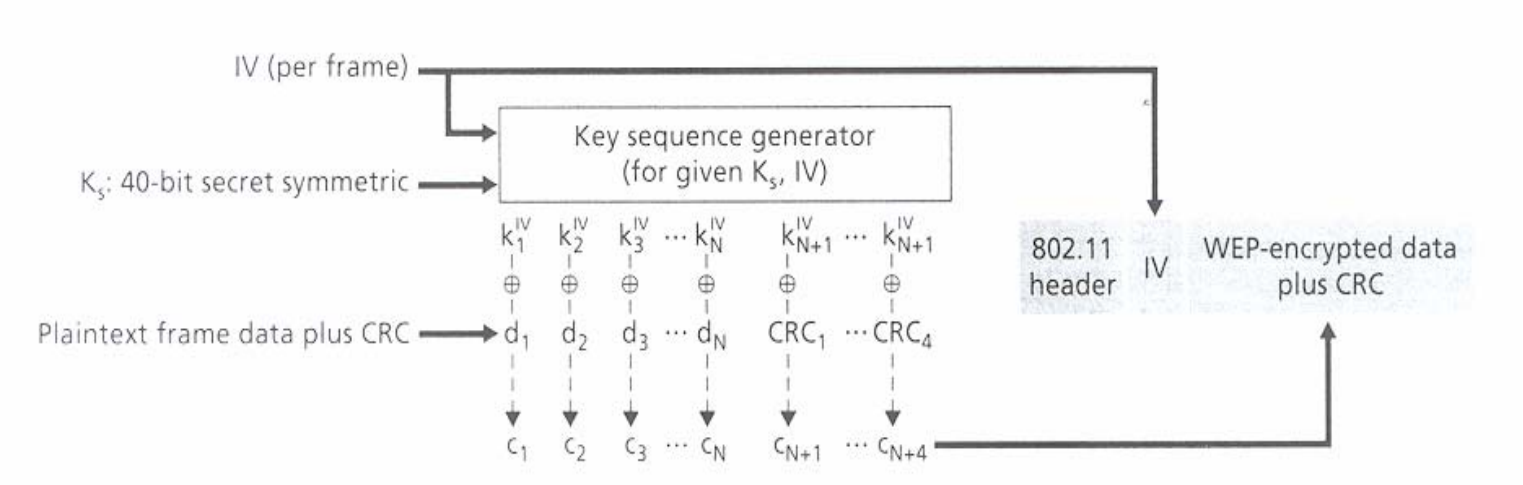
**Authentication (comparable to the approach mostly used in the wired scenario)**

- (wireless) host requests authentication by an AP
- AP responds by sending a 128-byte nonce value (nonce:= number which is used only once)
- host sends back the nonce encrypted by the shared key
- AP decrypts, compares and provides authentication if the values match thus preventing "replays"

**WEP data encryption:**

- used encryption algorithm: RC4 (Ron's Code 4 Pseudo Number Generator) from RSA Security Inc.
- key: a secret 40-bit key + a 24-bit IV (Initialization Vector) ---> WEP 64
- IV value changes from frame (packet) to frame, and is included as plaintext in the header of each frame
- data payload + a 4-byte CRC value are XORed bytewise with the random numbers generated by the key

**How RC4 works**



# Security in WLAN (IEEE 802.11) (3)

**Weaknesses:**

- surprisingly many installations use the same shared key for all users
- many 802.11 cards for notebook computers reset IV to 0 when the card is inserted, and increment it by 1 on each packet sent.

---->

The strength (proper use) requires to use a 64-bit key value *only once* --->

For a given 40-bit symmetric key (which changes rare, if ever), there are only $2^{24}$ different keys --->

$2^{24}$ = 16.777.216 frames = max. 24 GB, if 1 frame ≈ 1500 bytes

Given a data transmission rate of 11 Mbps:

$$\frac{1500\ bytes}{packet} \times \frac{8\ bits}{1\ byte} \times \frac{1\ \text{sec}}{11\ Mbits} \times \frac{1 Mbit}{10^6\ bits} \times 2^{24}\ packets \approx 18{,}300\ \text{sec} \approx 5\ hrs$$

---> Given a full utilization of the network, every 5 hours the IV´s are used again!

The probability of having chosen the same IV value is ≈ 99% after 12 000 frames sent (a few seconds)!

# Key Management (Distribution and Certification) (1)

Remaining problem of the symmetric key approach:
How to agree a priori, i.e. before the secure communication between two partners starts, on their secret key?
Solution: Using a trustworthy third person, the so-called

*Key Distribution Center (KDC)*

Idea: KDC is a server having a secret key with each registered user of the system. Key distribution (session key management) and authentication now goes through the KDC.

**Creating and distributing a unique session key between Alice and Bob via the KDC**