Technische Universität Berlin

Distributed Algorithms

Security

Technische Universität Berlin

Content

- Basics
- Symmetric/Asymmetric Cryptosystems
- Hybrid
- Authentication
- Hash Functions
- Digital Signatures
- Authorization
- Certificates
- Security Protocols

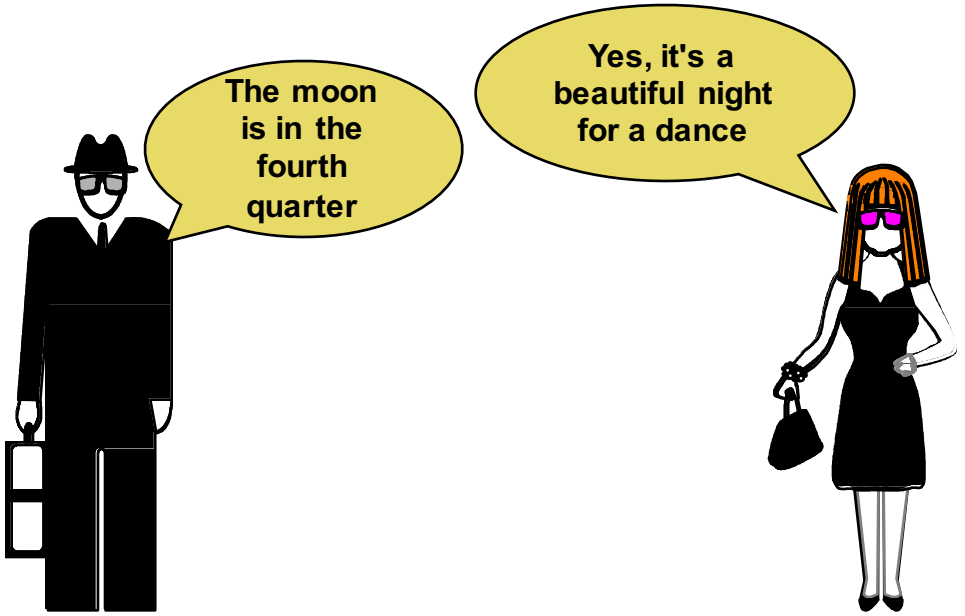
Security

Mechanisms

- **Authentication**
- **Authorization**
- **Confidentiality**
- **Integrity**
- **Non-Repudiation**

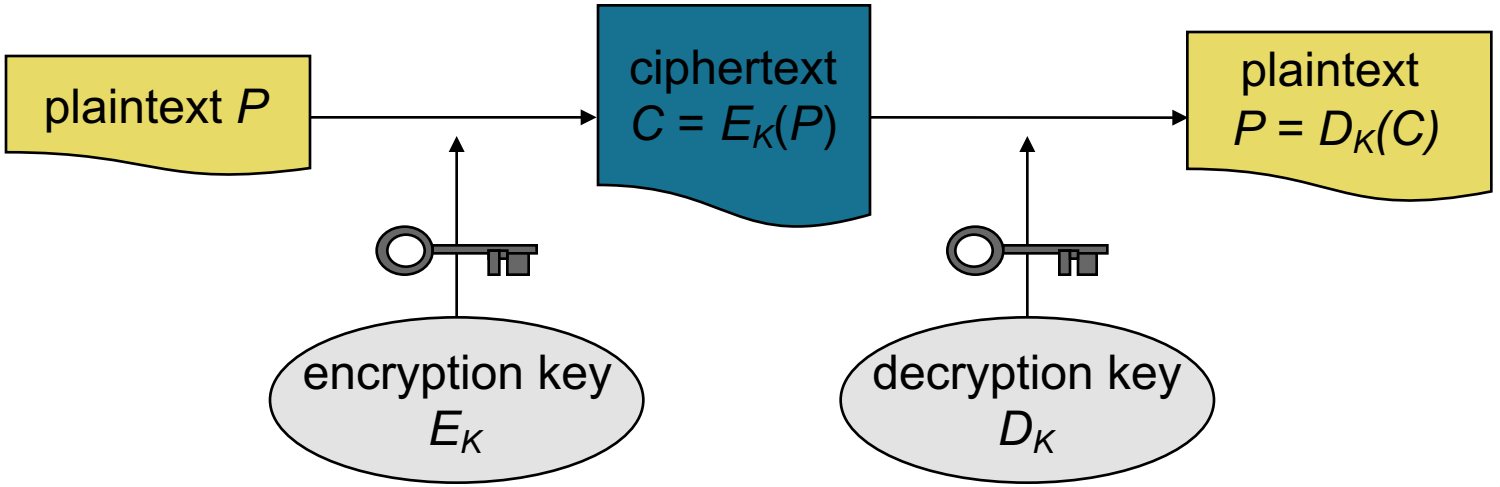
Basic technologies

- **Cryptography**
- **Security Protocols**



Cryptography

- Encryption algorithms are based on **keys**
- Based on the identity $P = D_K(E_K(P))$
- Algorithm itself can be openly published
- Only the keys need to be protected



Resistance Against Attacks

- Cryptography and security protocols depend on their resistance against attacks
- For example: For any encryption/decryption function it should be computationally infeasible to find the key when given the plaintext and corresponding ciphertext

- Passive Attacks
  - Eavesdropping
  - Analysis of communication patterns
  - Countermeasures: Encryption, Ensuring Anonymity

Active Attacks

- Password and encryption key cracking
  (brute force, dictionary attack etc.)
- Modification of the message stream
  - Insertion, deletion, modification, repetition of messages
- Unauthorized access to resources
  (e.g., stolen password, security hole)
  - Reading, changing, insertion, deletion of data (e.g., vandalism)
  - Triggering of functionality
- Denial of Service
  - Triggering of system crashes or spamming system with messages
- Countermeasures: Authentication, Authorization, Encrypted Sequence Number, Limited Key and Password Lifetime, Digital Signatures, Certificates, Fixing of Security Vulnerabilities

Symmetric Cryptographic Systems

- Aka. *secret-key* or *shared-key* systems
- Sender and receiver use **the same secret key *K*** for encryption and decryption
- Problem: How to securely exchange the secret key?
- Symmetric algorithms are typically fast and suitable for processing large streams of data
- Prominent Algorithms: DES, IDEA, AES

Asymmetric Cryptographic Systems

- Aka. *public key systems*
- Different keys used for encryption and decryption
- Each participant manages two keys,
  - a **private key K_-** that is kept secret and
  - a **public key K_+** that is openly published
- Assumption: Private key cannot be computed from public key efficiently
- Prominent Algorithms: RSA, ElGamal

Technische Universität Berlin

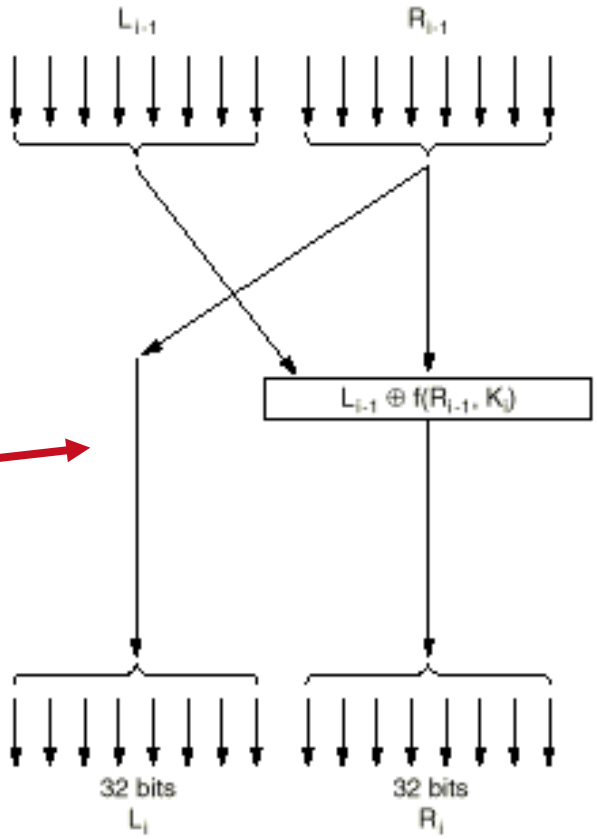
SYMMETRIC CRYPTOGRAPHIC SYSTEMS

One-Time Pads

- Plaintext is XORed with a random bit sequence to generate the ciphertext and vice versa
  - For each bit in the pad which is 1, the bit in the ciphertext is derived by negating the bit in the plaintext; the other bits are equal
- The Sequence must be
  - known to the sender and the receiver only,
  - really random, e.g., generated using white noise,
  - only used once,
  - as long as the plaintext.
- Then, cryptographic analysis is not possible
- Disadvantages
  - Expensive key exchange
  - Possible loss of synchronization
  - Generation of “good” random numbers required

DES (Data Encryption Standard)

- Prominent symmetric method
- US Standard since 1977
- Block cipher using 64-Bit data blocks
- 16 x 48-Bit keys are generated from an original **56-Bit key**
- 18 processing steps
  - Initial permutation
  - 16 processing steps using the generated keys and a mangler function *f*
  - Final permutation
- Designed for hardware encryption and decryption → slow software implementation



DES (contd.)

- Today no longer considered safe enough
- Exhaustive testing of all 2^{56} (ca. $7 \cdot 10^{17}$) keys seems possible with a few fast or a lot of slower computers

- "Chinese Lottery" (Quisquater and Girault, 1991)
  - Build a cheap DES Chip into every TV set and radio
  - The Chinese government will distribute the cipher text (and a key) to all devices; devices decrypt text with given key
  - The one who finds the clear text ("winner of lottery") obtains a prize (→ incentive)
  - Solution could be found in approximately 60 seconds

Triple DES (3DES)

- Uses 2 keys $K1$, $K2$ with 56-Bit each
  → key size is 112-Bit

- Encryption: $C = E_{K1}(D_{K2}(E_{K1}(P)))$
- Decryption: $P = D_{K1}(E_{K2}(D_{K1}(C)))$

- Choosing $K1 = K2$ degrades 3DES to DES

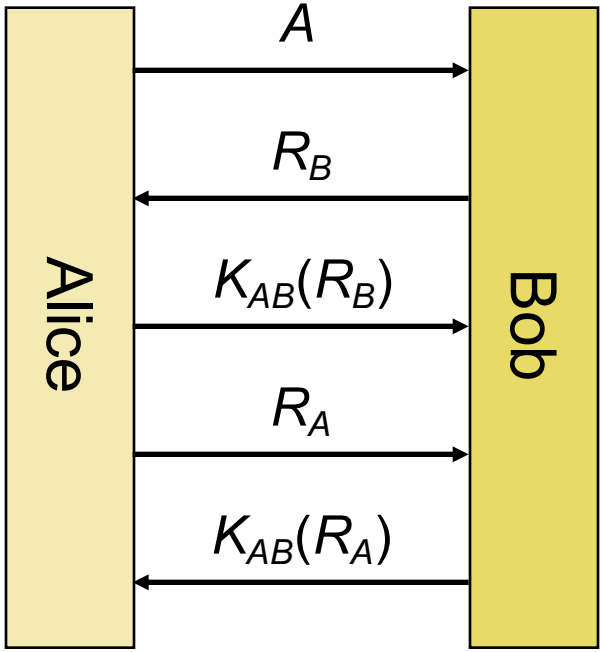
- Double DES would only have 57-Bit effective key size due to a meet-in-the-middle attack

Mutual Authentication Using a Shared Key

- Alice and Bob share a secret key and want to do mutual authentication
- Protocol is based upon a **challenge / response interaction**
  - Challenges are usually random numbers that are only used once
    → **nonces** (numbers used once)
  - A valid response is the encrypted challenge
  - The issuer of the challenge decrypts the response and compares it to the original challenge
  - If they are equal, he is sure that he is talking with the intended partner
- For *mutual* authentication the challenge / response interaction must be done in both directions

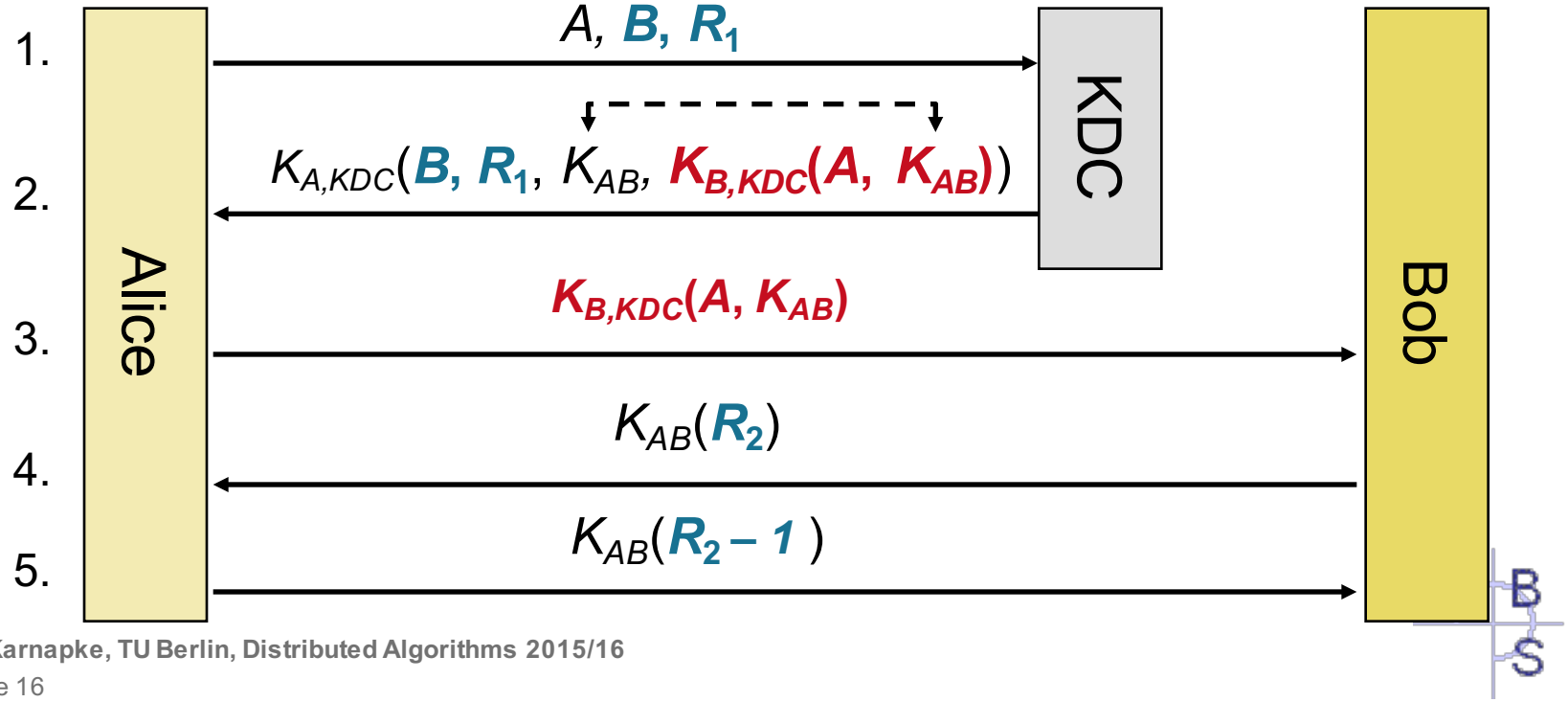
Mutual Authentication Using a Shared Key

- Alice sends her identity A to Bob
- Bob sends a challenge R_B to Alice
- Alice encrypts the received challenge and sends it, i.e. $K_{AB}(R_B)$, to Bob
- Bob decrypts the received encrypted challenge and compares it R_B
- Alice send a challenge R_A to Bob
- Bob encrypts the received challenge and sends it, i.e. $K_{AB}(R_A)$, to Alice
- Alice decrypts the received encrypted challenge and compares it to R_A



Needham-Schroeder Protocol

- Mutual authentication using a *Key Distribution Center (KDC)*
- For *n* communication partners, only *n* keys are needed instead of *n (n -1) / 2* keys for direct mutual authentication



Problems with Needham-Schroeder Protocol

- Main problem: *No guarantees on the freshness of the ticket*

- Example scenario
  - Mallory gets in possession of K_{AB}
  - Replays step 3
  - → Mallory can easily answer message 4 as he knows K_{AB} (this is even possible if Alice has changed her key!)

- Possible solutions
  - Add extra nonces to guarantee freshness of messages (cf. Otway-Rees)
  - Add timestamps to prevent later replay of messages (cf. Kerberos)

Technische Universität Berlin

ASYMMETRIC CRYPTOGRAPHIC SYSTEM

K B S

Asymmetric Encryption

- Private and public key together form a unique pair of keys
- $K_+(K_-(P)) = K_-(K_+(P)) = P$
- If P is encrypted with a public key, it can only be decrypted with the corresponding private key and vice versa
- Advantage: No secret keys need to be exchanged!
- Applications
  - Confidentiality
  - Authentication
  - Electronic signature (integrity, non-repudiation)
- Disadvantage: Not nearly as efficient as symmetric encryption
  - Therefore, *hybrid techniques* are often used in practice (more on this later)

Usage Scenarios for Asymmetric Cryptography

- Ensuring **confidentiality**
  - The sender uses the receiver's public key to encrypt the message
  - Only the receiver can decrypt the encrypted message (with its private key)

- Ensuring **authenticity**, **integrity**, and **non-repudiation**
  - The sender encrypts the message with its own private key
  - Everybody can decrypt the encrypted message (with the sender's public key)
  - This serves as a proof for the authenticity of the message
    → **Digital Signatures**

RSA (Rivest-Shamir-Adleman) Cryptosystem

- Based on the fact that it is computational infeasible to find the prime factors of large numbers
- Private and public key are constructed from two very large prime numbers
- Breaking RSA is as hard as finding those prime numbers!

- However: It is not known whether it is possible to decrypt a ciphertext without knowing the secret key!

RSA Key Generation, Encryption, Decryption

Key Generation

1. Choose two very large prime numbers p and q
2. Compute $n = p \cdot q$ and $z = (p-1) \cdot (q-1)$
3. Choose a number d that is relatively prime to z
4. Compute the number e such that $e \cdot d \mod z = 1$

(n, e) is the *public key* which is openly published

(n, d) is the *private key* which is kept secret

Encryption with public key: $C = P^e \mod n$ $(P < n)$

Decryption with private key: $P = C^d \mod n$

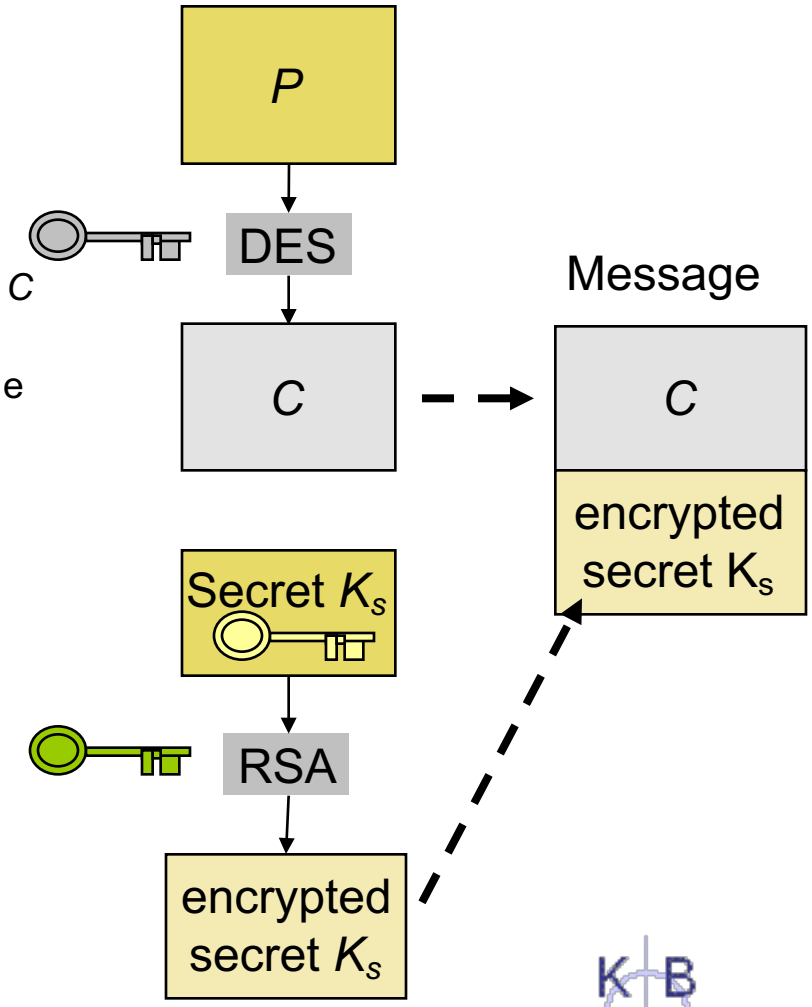
Also possible: encryption with private key and decryption with public key (used for digital signatures)

RSA Example

$p = 13$, $q = 23$	(p and q are prime numbers)
$n = p \cdot q = 299$	
$z = (p - 1) \cdot (q - 1) = 264$	
$d = 5$	($GCD(d,z) = 1$)
$e = 53$	($z + 1$ is a factor of $e \cdot d$)
P = “H” = 72	($P < n$)
$C = P^e \ mod \ n = 72^{53} \ mod \ 299 = 271$	(encryption with public key)
$P' = C^d \ mod \ n = 271^5 \ mod \ 299 = 72 =$ “H”	(decryption with private key)
$C = P^d \ mod \ n = 72^5 \ mod \ 299 = 128$	(encryption with private key)
$P = C^e \ mod \ n = 128^{53} \ mod \ 299 = 72 =$ “H”	(decryption with public key)

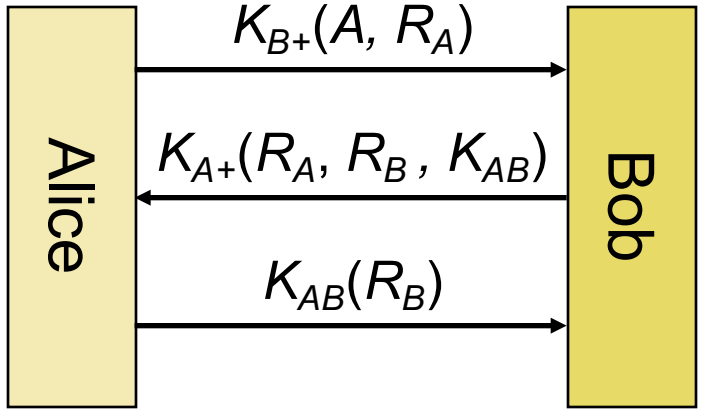
Hybrid Techniques

- Goal: Increased efficiency
- Sender side
  - Generate session key K_s
  - Encrypt plaintext P with K_s resulting in ciphertext C
  - Encrypt K_s with the public key of the receiver
  - Send C and the encrypted key K_s in one message → digital envelope
- Receiver side
  - Receive message
  - Decrypt K_s with private key
  - Decrypt C with K_s



Mutual Authentication and Secure Key Exchange with Asymmetric Encryption

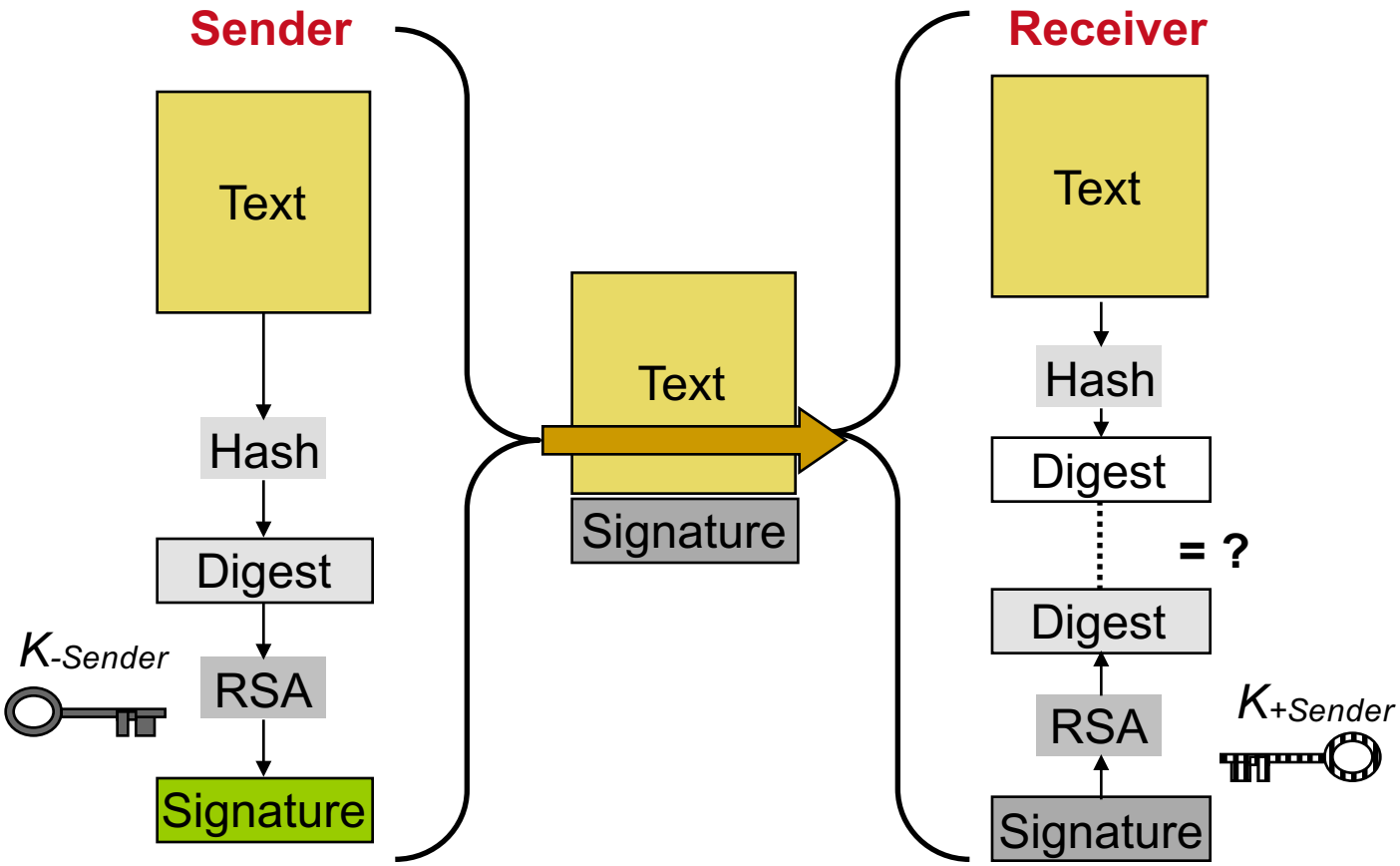
- Alice encrypts her identity A and a challenge R_A with K_{B+} and sends the result to Bob
- Bob decrypts the received message with K_{B-}
- Bob encrypts the received challenge together with a new challenge R_B and a generated secret key K_{AB} with K_{A+} and sends the result to Alice
- Alice decrypts the received message with K_{A-} and compares the first of the received challenges to R_A
- Alice encrypts the second of the received challenges with the received key and sends the result to Bob
- Bob decrypts the received message with K_{AB} and compares the received challenge to R_B



Requirements of Digital Signatures

- The requirements for digital signatures are similar to those one expects to hold for handwritten signatures
  - The identity of the signer can be verified (*authenticity*)
  - The signer cannot deny the contents of the message (*message integrity*)
  - The signer cannot repudiate the fact that he signed the message (*non-repudiation*)
- Note that the 3rd requirement implies that no other person can sign messages in the name of the signer
- To increase efficiency, usually a message digest (generated using a cryptographic hash function) is signed instead of the whole message!

Digital Signatures using a Public Key Cryptosystem



Digital Signatures using a Public Key Cryptosystem

- *Signing* a Message
  - The hash value of the message is encrypted with the private key of the signer and attached to the message
- *Verifying* a signed Message
  - The message to be verified is hashed again and the output is compared with the signed hash value that is decrypted with the public key of the signer
- Hash values are signed instead of messages as a hash value is usually much smaller than the original message

Certificates

- Problem: How can it be verified that a public key really belongs to some person?
- Solution: Public keys are certified (i.e. digitally signed) by a trusted *Certification Authority (CA)* whose public key is known to all parties

- Problem: What happens if a private key is stolen?
- Solution: The certificate belonging to the stolen key is revoked → *Certificate Revocation List (CRL)*
- Requirement: On-line access to the CRL is necessary

Bibliography

1. A. S. Tanenbaum and M. van Steen. *Distributed Systems: Principles and Paradigms*. Prentice Hall, 2002. pages 413-491
2. G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed Systems: Concepts and Design*. Addison-Wesley, 3rd edition, 2001. pages 251-308
3. R. Needham and M. Schroeder. *Using Encryption for Authentication in Large Networks of Computers*. Communications of the ACM, 21(12):993--999, 1978