PA160: Net-Centric Computing II.

Network Security

Luděk Matyska

Slides by: Tomáš Rebok

Faculty of Informatics Masaryk University

Spring 2017

Course Introduction

PA160 – Course Introduction

Course Organization

- attending the lectures is optional
- the knowledge acquired during PB156 & PA159 courses is assumed
- course materials will be published on the course webpage (more or less in time ;-))
- assessment methodology:
  - final exam (written form)
- course literature:
  - slides, RFCs, ...
  - literature being announced in relevant course parts

Course Overview

- the course goal:
  - to provide an advanced insight into the area of computer networks and their applications
- discussed topics:
  - network security
  - network management and monitoring
  - parallel and distributed applications, grids
  - service-oriented architectures
  - specification and verification of communication protocols
  - network simulators and emulators
  - etc.

1. Network Security

1. Network Security

Lecture overview

1. Introduction
   - Motivation
   - Network Security Elements
2. Cryptography Introduction
   - Symmetric Cryptography
   - Asymmetric Cryptography
3. Network Security Mechanisms
   - Entity Authentication
   - Message Confidentiality (Security)
   - Message Integrity
   - Message Authentication
   - Digital Signature
   - Key Management and Distribution
4. Example Protocols Ensuring Network Security in the Internet
   - Application-layer Security
   - Transport-layer Security
   - Network-layer Security
   - Link-layer Security
5. Network & Data Protection – Maintaining the Availability
   - Firewalls
   - Intrusion Detection Systems
   - Denial of Service (DoS) & Distributed Denial of Service (DDoS) attacks
   - Virusses & Worms
   - Network & Data Protection – Concluding Remarks
6. Conclusion

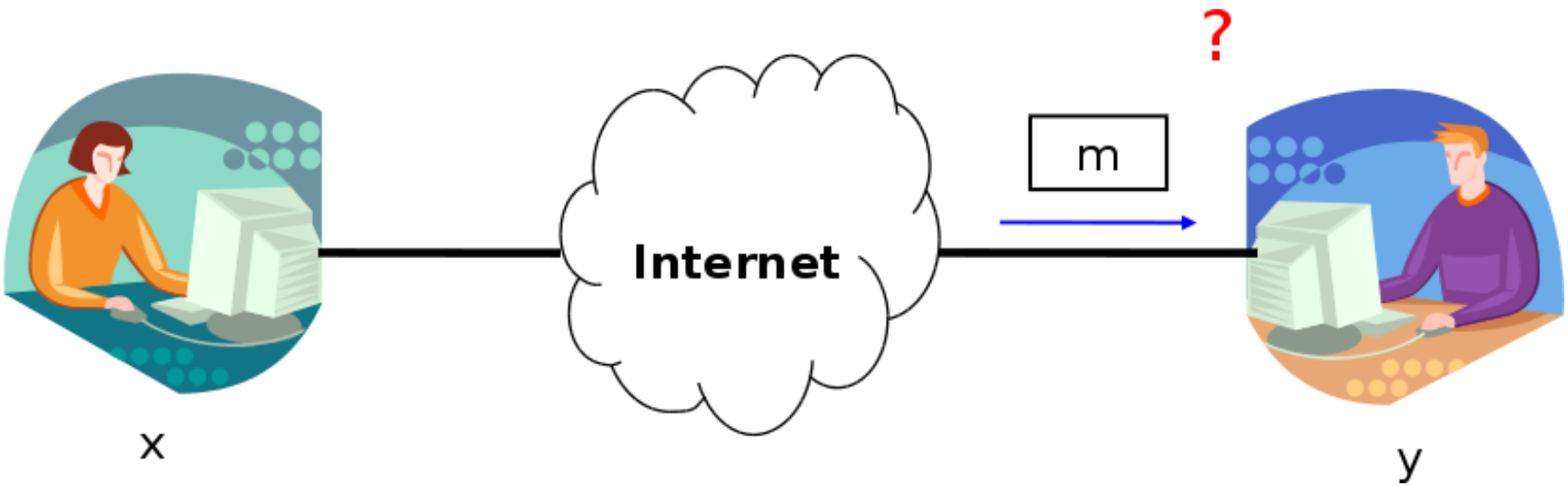
Lecture overview

Security

- most of computer science is concerned with achieving *desired* behavior
- in some sense, security is concerned with preventing **undesired** behavior
  - different way of thinking!
  - an enemy/opponent/hacker/adversary may be *actively* and *maliciously* trying to circumvent any protective measures one put in place
- **computer** vs. **network** security
  - *computer security* – focuses on security aspects of systems in isolation
    - viruses, secure data storage, OS security, etc.
  - *network security* – focuses on security of data as it is transmitted between networked systems
    - authentication protocols, encryption of transmitted data, firewalls, etc.
  - not always a clear-cut dividing line...

What can go wrong?

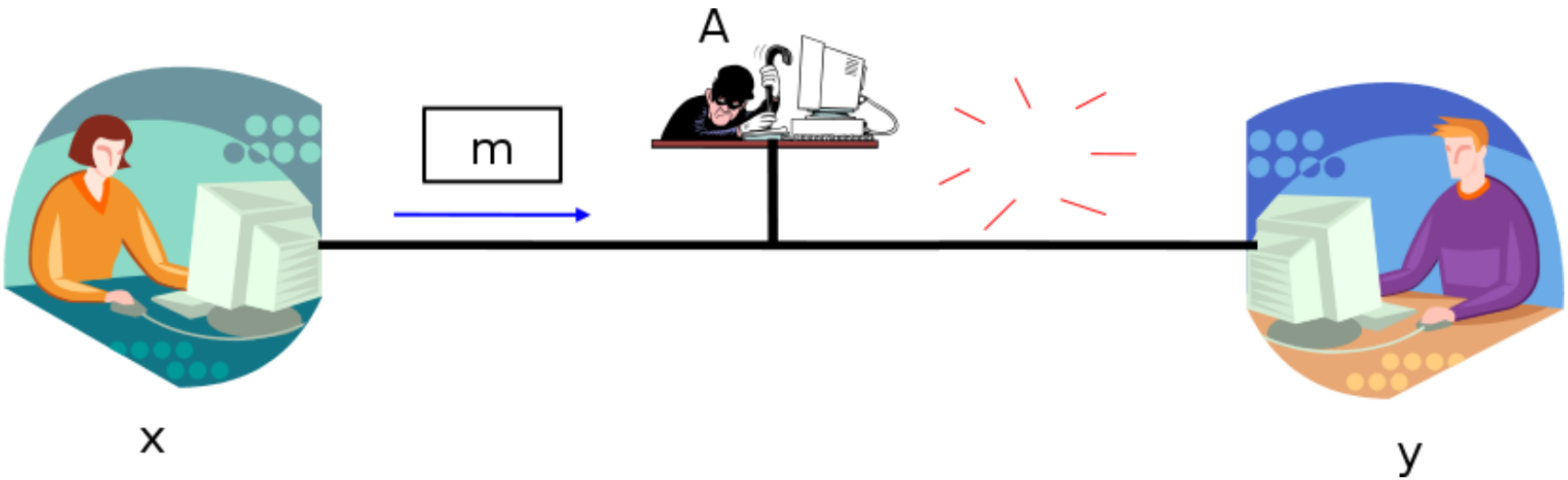
- when your computer y receives or is waiting for a message m?



What can go wrong?

Message loss

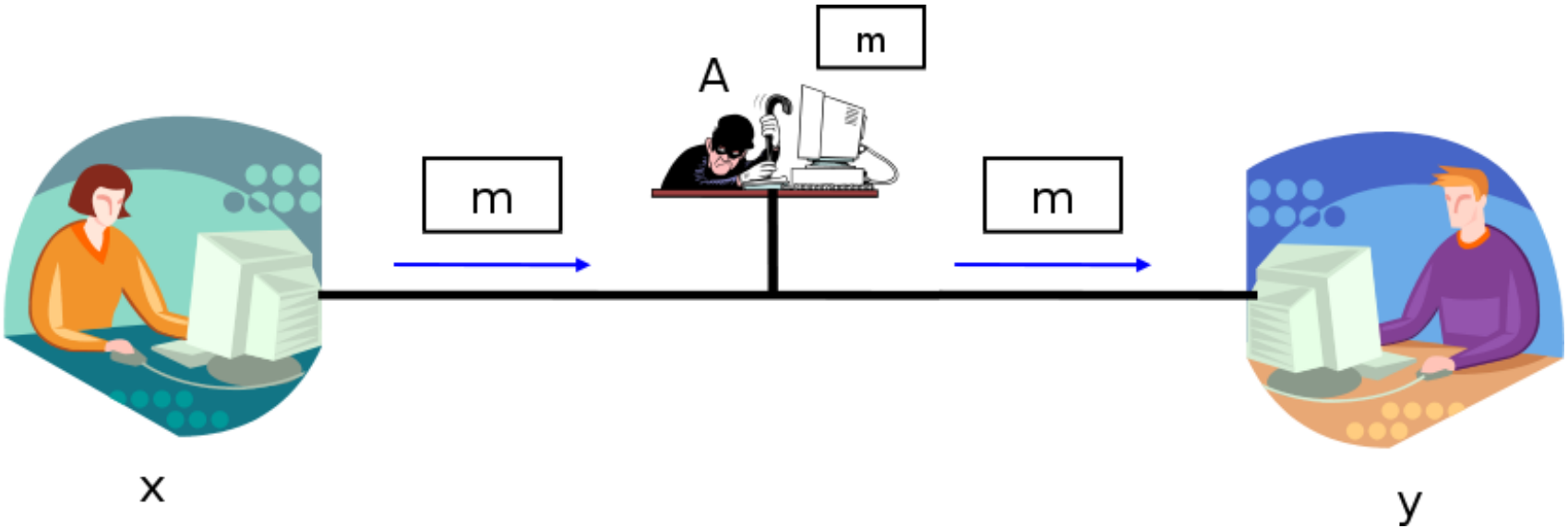
- adversary A can discard m in its transit



What can go wrong?

Message interception

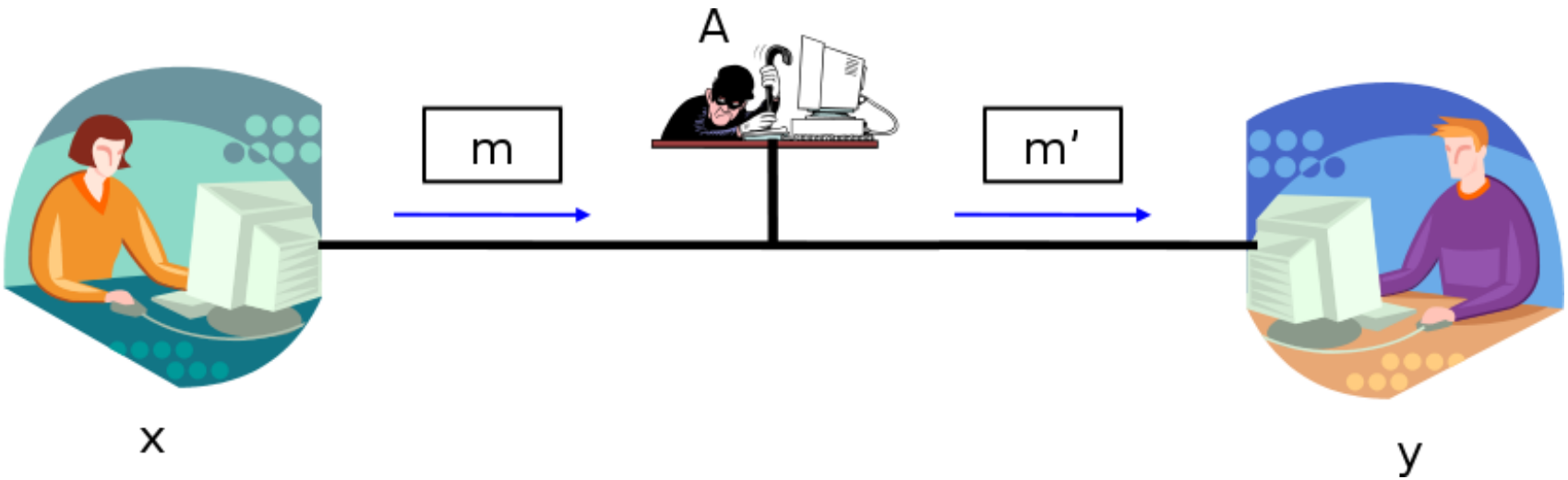
- adversary A can get a copy of m when m passes by



What can go wrong?

Message modification

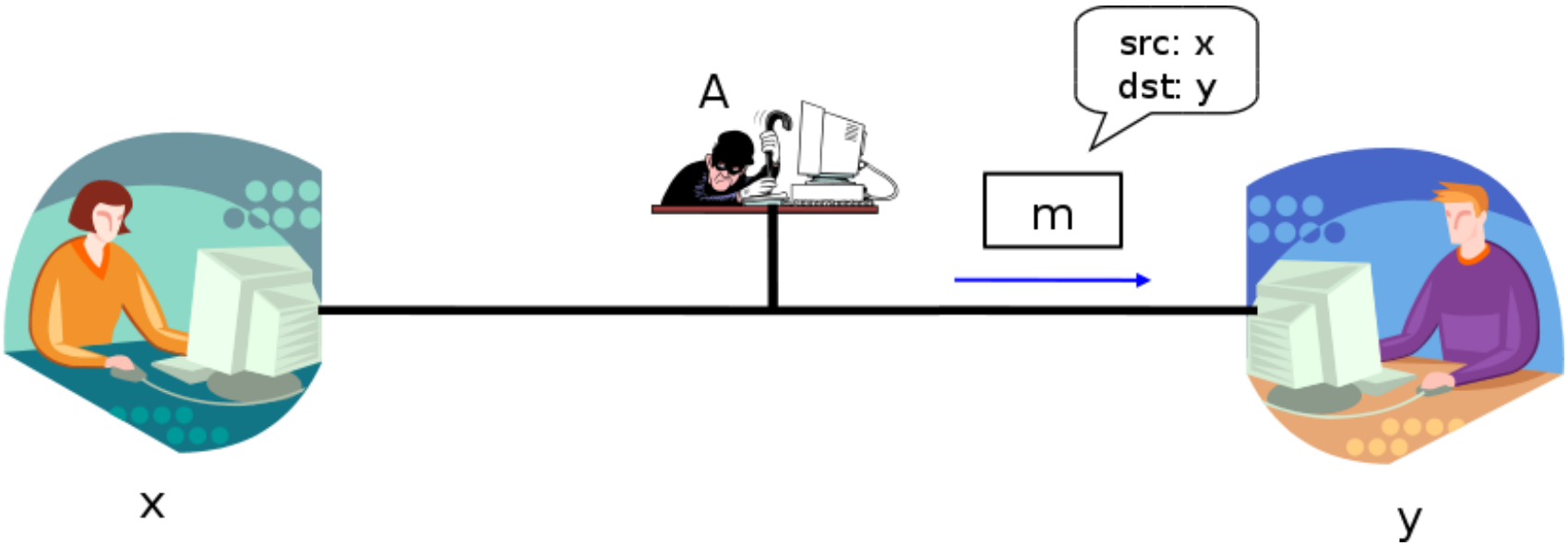
- adversary A can arbitrarily modify the content of m to become m'



What can go wrong?

Message insertion

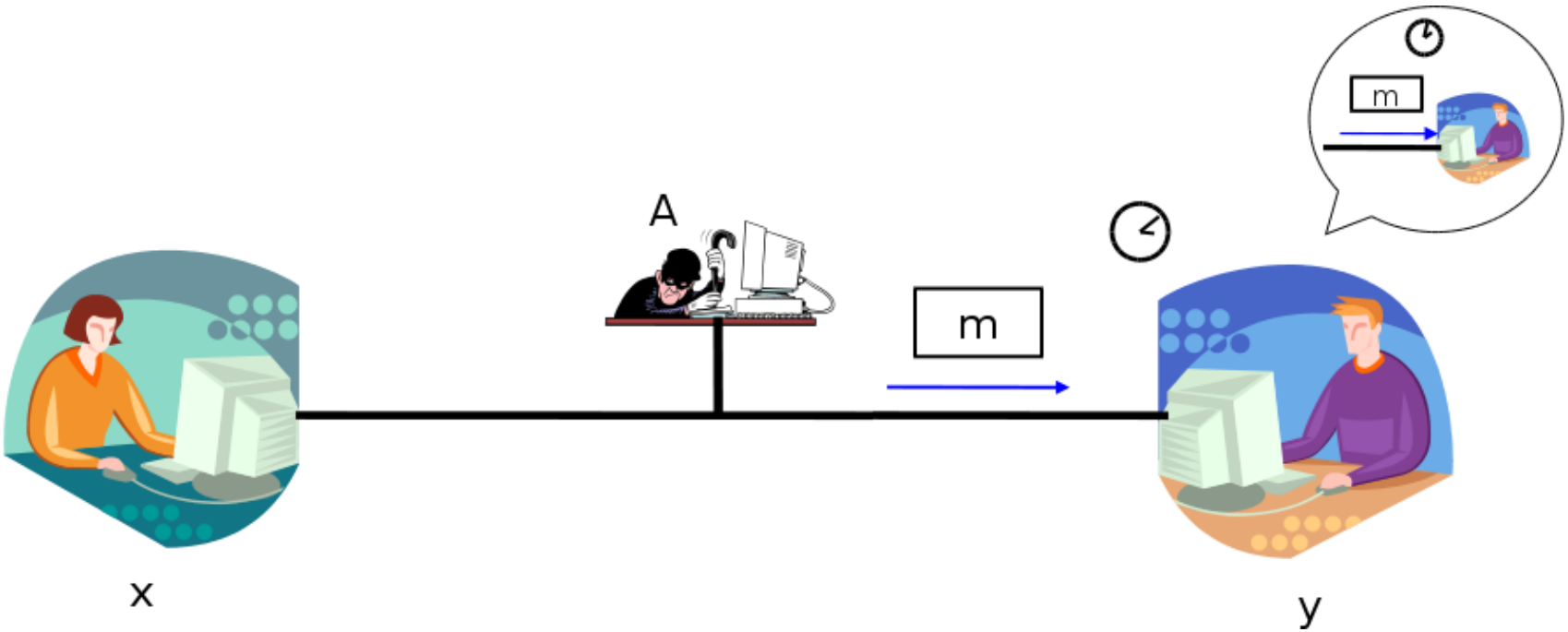
- adversary A can arbitrarily fabricate a message m, pretending that m was sent by x



What can go wrong?

Message replay

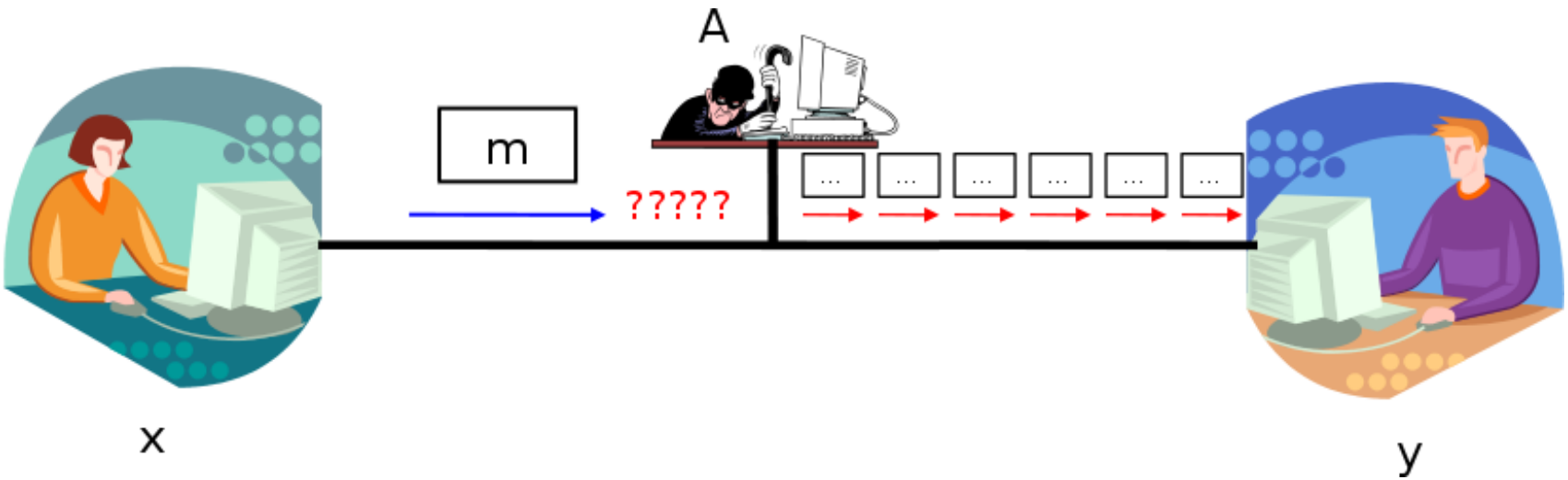
- adversary A can replay a message m that has been sent earlier by x and received by y



What can go wrong?

Denial-of-Service attack

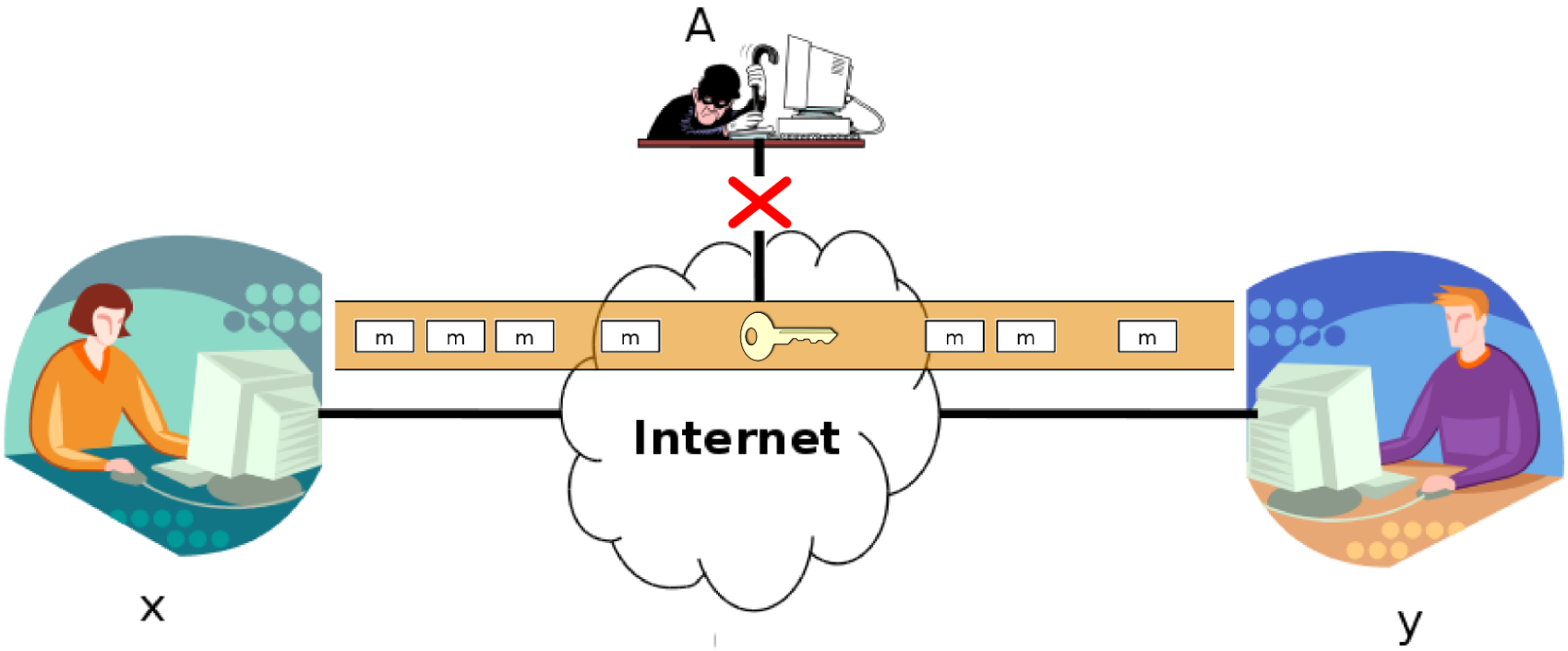
- adversary A can send huge amount of messages to y to block m from arriving at y
  - in the case of botnet attack, the adversary instructs many bots to send messages to y simultaneously



What is needed?

Reliable transmission

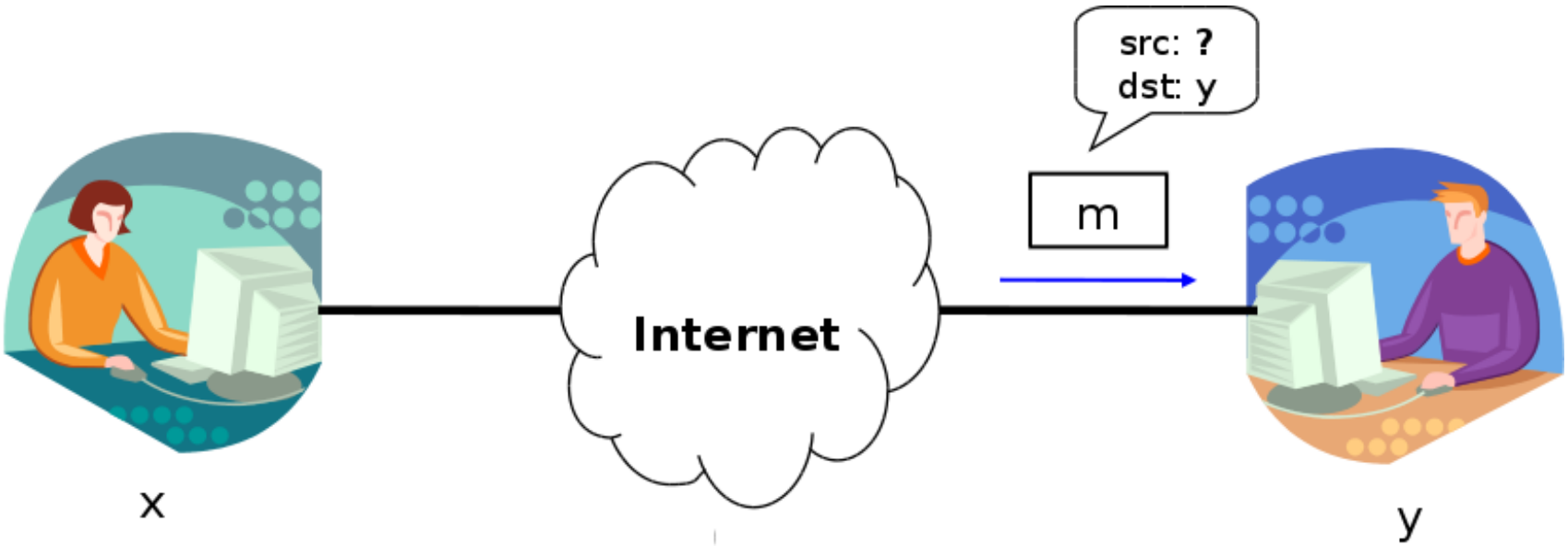
- y wants to receive all messages sent by x to her
- both x and y want to make sure the message is not tampered with
- eventually they both want the message not to be overheard by any third party



What is needed?

Sender verification

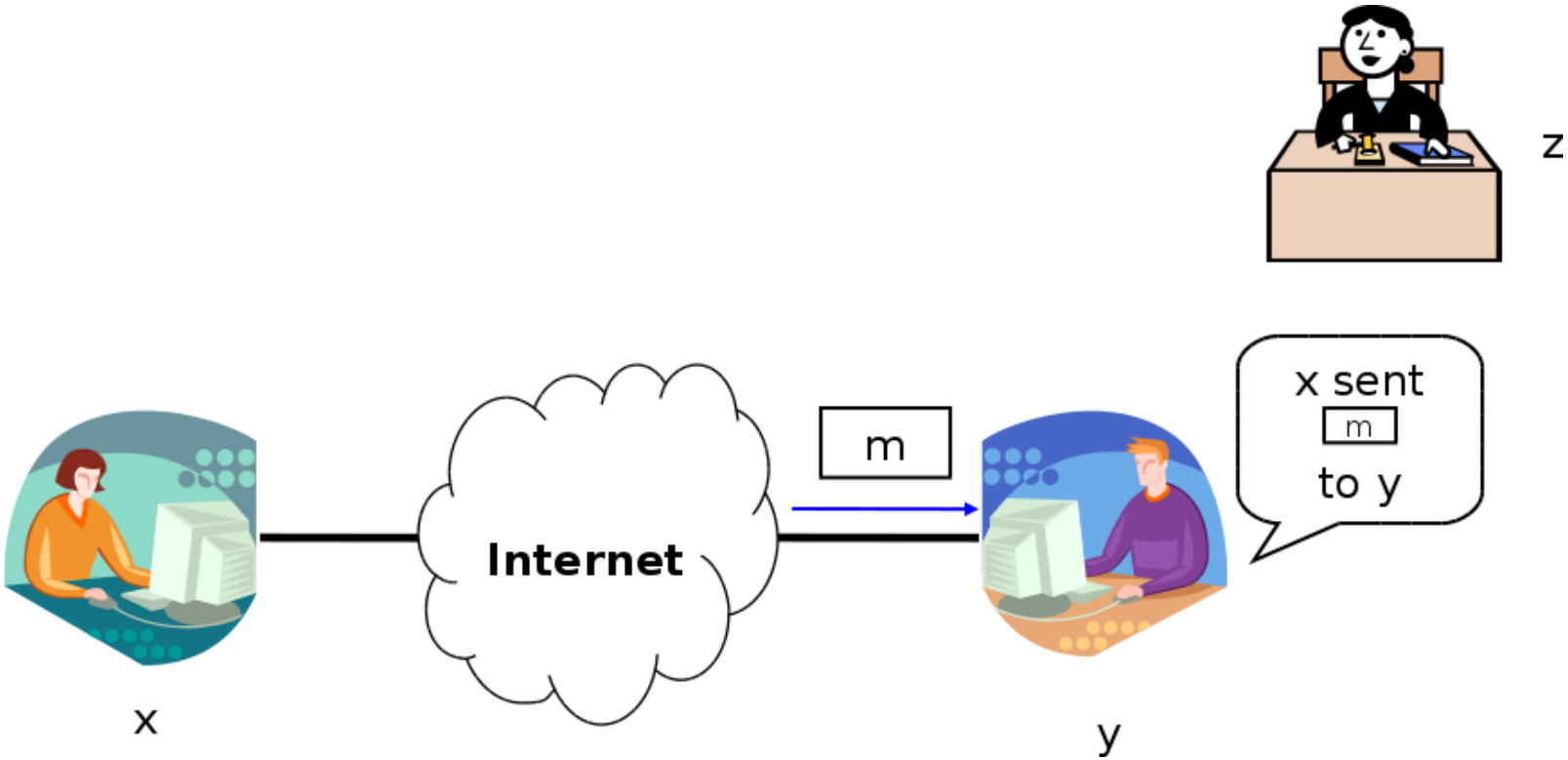
- y wants to be able to verify that the message m has been really sent by x



What is needed?

Message non-repudiation

- x wants to be able to prove to third party z that y has received a message m from x
- y wants to be able to prove to third party z that x has sent a message m to y



Network Security

Network Security is great...

- protects messages from interception in their transit
- provides desired level of privacy for users or data
- detects and discards messages that had been modified, inserted, or replayed
- disallows unauthorized access to local system resources and sensitive data

...but hard to achieve

- many layers in network architecture
- many different media of network connection
- adversary's location hard to determine
- new attacks keep emerging
- cryptographic overhead

Network Security – Basic Elements

A *secure communication network* should provide the following mechanisms/services:

- **AAA**
  - *Authentication*
  - *Authorization*
  - *Accounting*
- + **secured communication**
  - *Confidentiality*
  - *Integrity*
  - *Non-repudiation*
- + **Availability**

Network Security – Basic Elements

Authentication

Authentication

- *in Czech:* **autentizace**
  - **NOT** autentikace, autentifikace, . . .
- several perspectives:
  - a process verifying that someone is who they claim they are (= *entity authentication*)
  - a process assuring the recipient that the message is from the source that it claims to be from (= *message authentication*)
- requires a determination of the entity's identity
- several methods for identity determination exist:
  - *based on what a user knows* – a proper pair of username and password/PIN
  - *based on what a user has* – a tool which a user owns (USB dongle, smart card, private key, etc.)
  - *based on what a user is* – a user has characteristics, which can be verified (fingerprint, retinal images, etc.)
  - *based on what a user can* – a user is able to answer particular verifying questions

Network Security – Basic Elements

Authorization and Accounting

Authorization

- *process of granting the user access to specific system resources*
  - based on user's profile and the local/global policy controlling the resource access
  - follows the authentication process
- ensured by access control lists (ACLs)
  - define permissions to perform particular operation or to access particular resources

Accounting

- *monitoring the users' usage of network services*
- the information can be used for management, planning, billing, or any other purposes

Network Security – Basic Elements

Confidentiality, Integrity, and Non-repudiation

Confidentiality

- *protects data and information from unauthorized disclosure (sniffers)*
- data during transmission cannot be read by any unauthorized user
  - just the sender and receiver understand the content of the transferred message
- ensured by message encryption

Integrity

- *protects data from unauthorized modification*
  - data during transmission cannot be modified or fabricated by any unauthorized user
- usually based on encryption and hashing algorithms

Non-repudiation

- *sender nonrepudiation* and *receiver nonrepudiation* serves the receiver/sender for the purposes of proving origin/delivery of the message
  - neither the sender of a message can deny the transmission of the message nor the receiver of a message is able to deny the reception of the message
- ensured by digital signatures

Network Security – Basic Elements

Availability

Availability

- property of a system, network, or a resource being accessible and usable any time upon demand by an authorized system entity
  - i.e., a system is available if it provides services according to the system design whenever users require them
- has to be associated with various security services
  - such as services protecting from denial of service (DoS) attacks

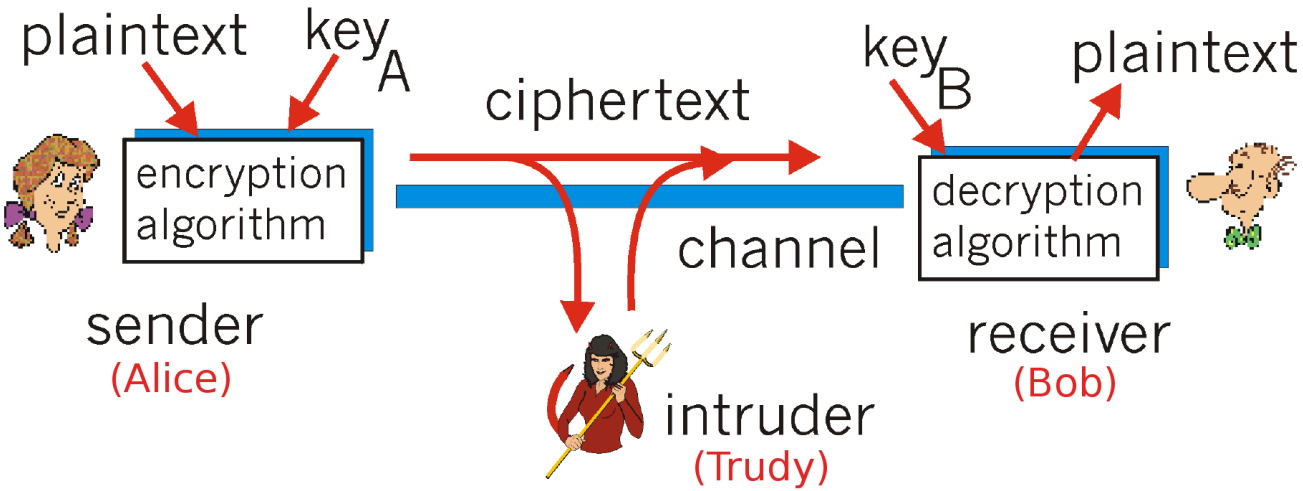
Lecture overview

Cryptography Introduction

- network security is mostly achieved through the use of *Cryptography*

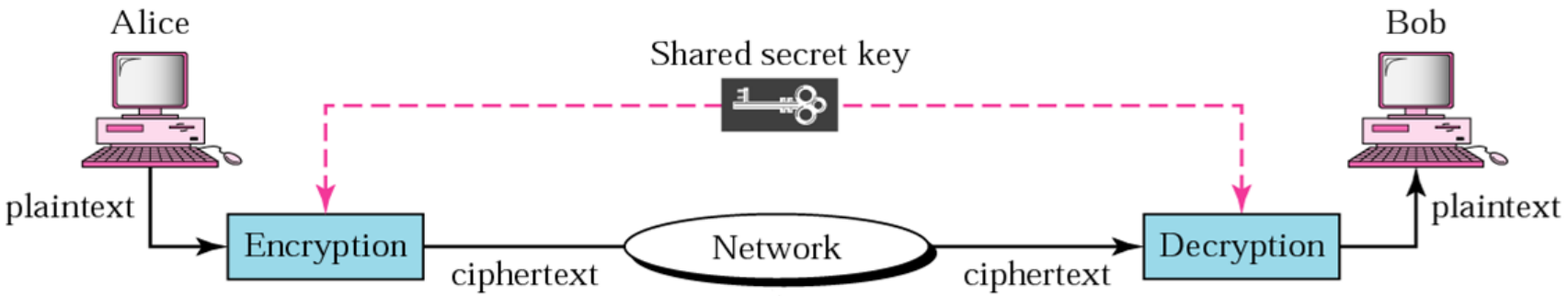
Cryptography:

- the science and art of transforming messages to make them secure and immune to attacks
- basic cryptography mechanisms:
  - *symmetric (secret-key) cryptography*
  - *asymmetric (public-key) cryptography*



Symmetric Cryptography

- the same key is used by both parties for both encryption and decryption
- advantages:
  - low computational demands
  - suitable for encryption of large messages
- drawbacks:
  - a necessity to share the secret key ($\rightarrow$ a mechanism for key distribution is required)
  - can lead to many keys required to be maintained (e.g., many-to-one communication in distributed environment)
- e.g., DES, 3DES, IDEA, Blowfish, Rivest Cipher 4 (RC4), AES, etc.



Symmetric Cryptography

Simple Symmetric Ciphers

Character-oriented

- operate on characters of an alphabet
- *substitution ciphers* – substitutes one symbol with another
  - *monoalphabetic* – a character/symbol in the plaintext is always changed to the same character/symbol in the ciphertext (e.g., Caesar cipher)
  - *polyalphabetic* – each occurrence of a character can have a different substitute
- *transposition ciphers* – reorders (permutes) symbols in a block of symbols
  - instead of substitutions, locations of characters change

Bit-oriented

- operate on blocks of bits
- e.g., XOR cipher, rotation cipher, etc.

Symmetric Cryptography

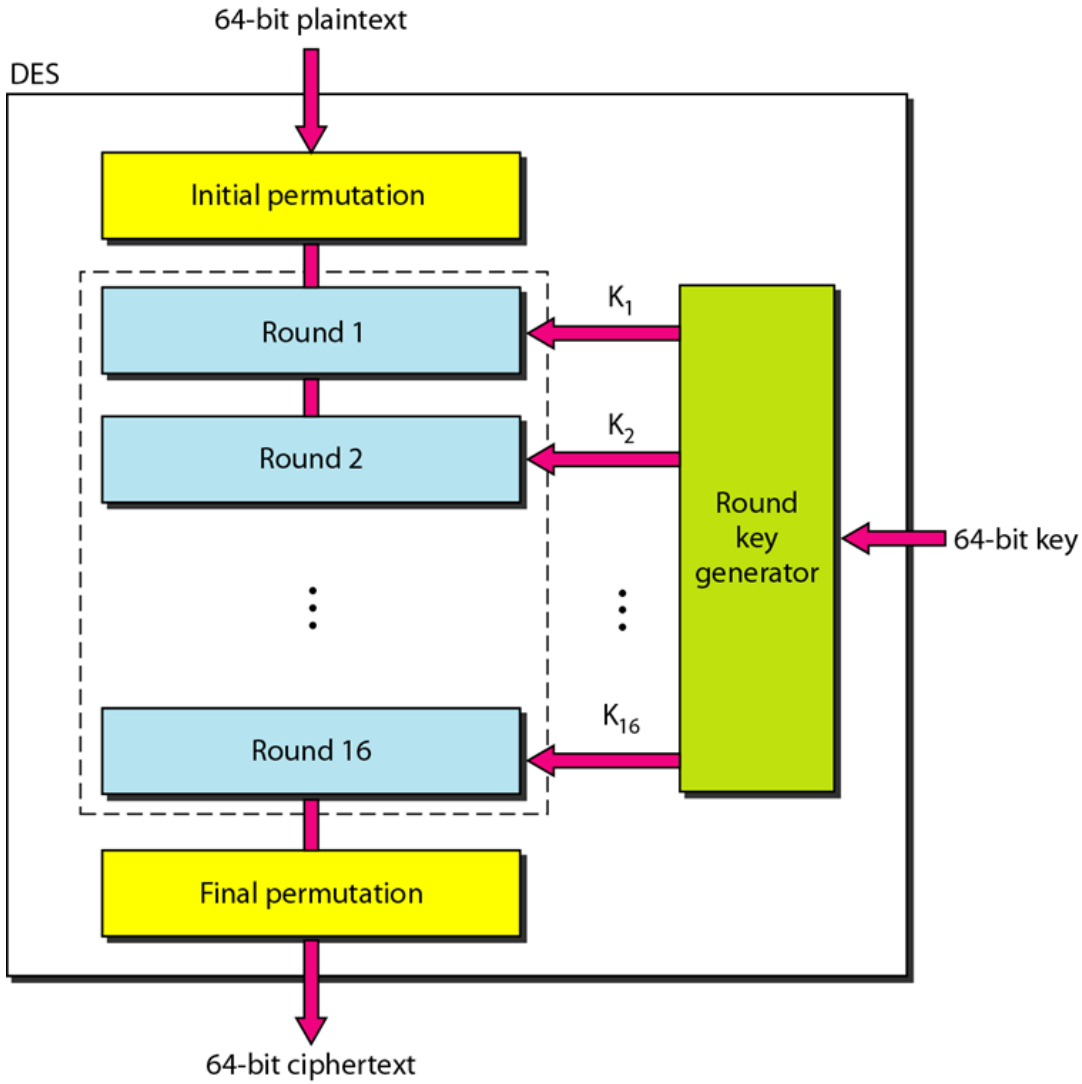
DES (Data Encryption Standard)

DES (Data Encryption Standard)

- designed by IBM and adopted by the U.S. government as the standard encryption method for nonmilitary and nonclassified use
- the algorithm encrypts a 64-bit plaintext blocks using a 64-bit key
  - in fact, the key is just 56-bits long (every eighth bit is used for parity checking and is ignored)
- DES has two transposition blocks (so-called *P-boxes*) and 16 complex round ciphers
  - although the 16 iteration round ciphers are conceptually the same, each uses a different key derived from the original key
- **not considered to be secure** any more
  - because of the key length (too short)
- see details in the animation: `http://www.cs.bham.ac.uk/research/projects/lemsys/DES/DESPage.jsp`

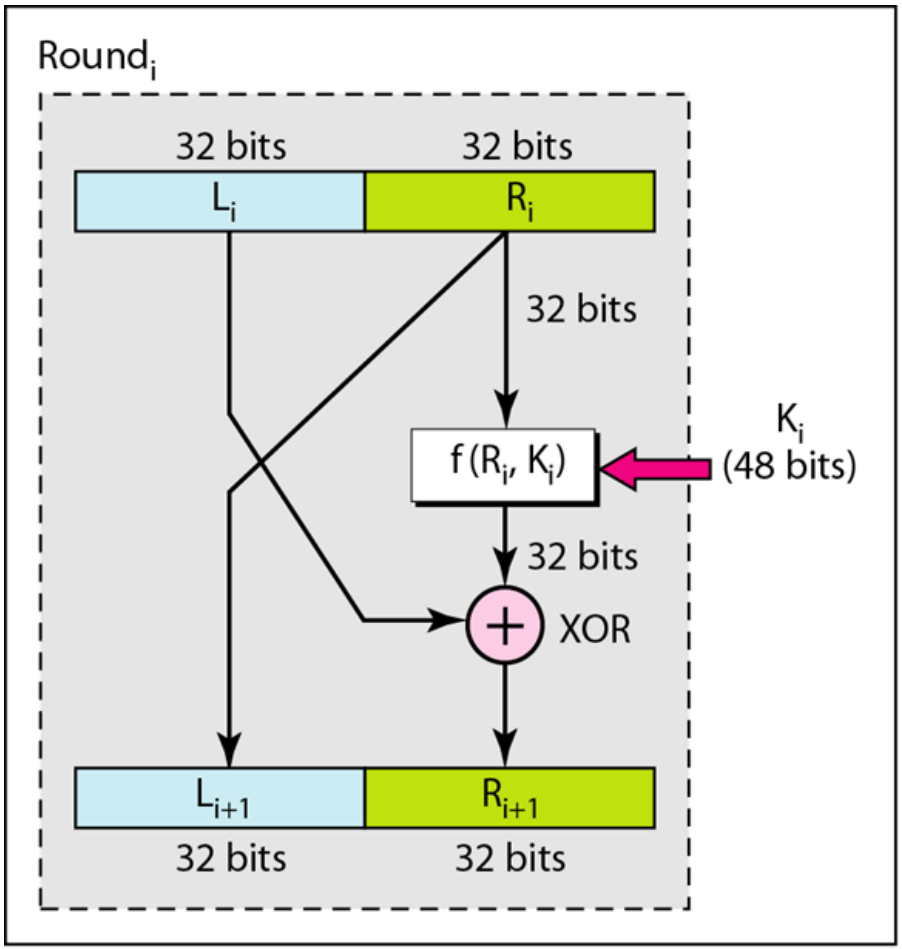
Symmetric Cryptography

DES (Data Encryption Standard) – Algorithm outline I.

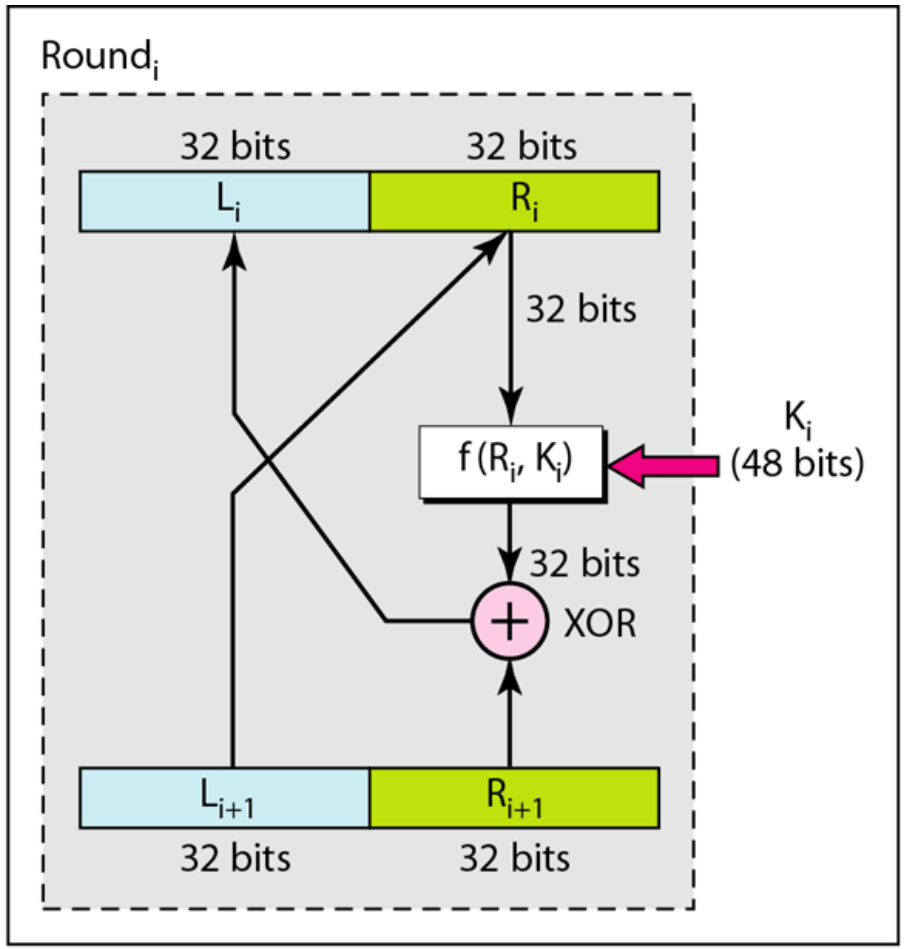


Symmetric Cryptography

DES (Data Encryption Standard) – Algorithm outline II. (single round)



a. Encryption round



b. Decryption round

Symmetric Cryptography

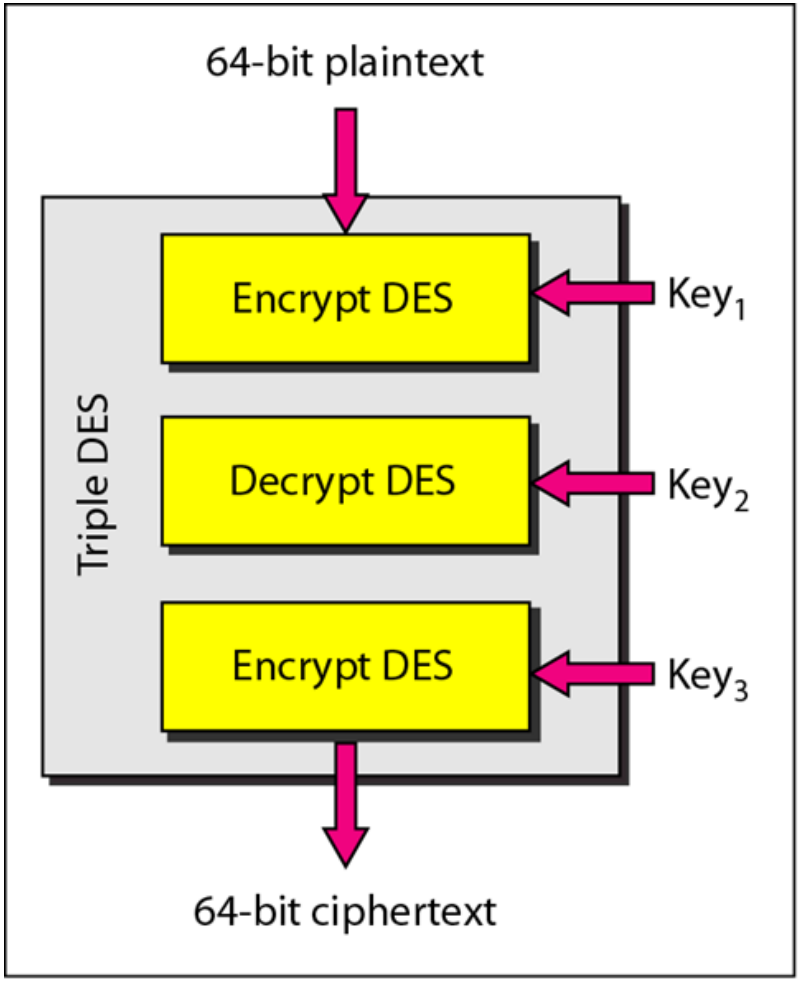
Triple DES (3DES)

Triple DES (3DES)

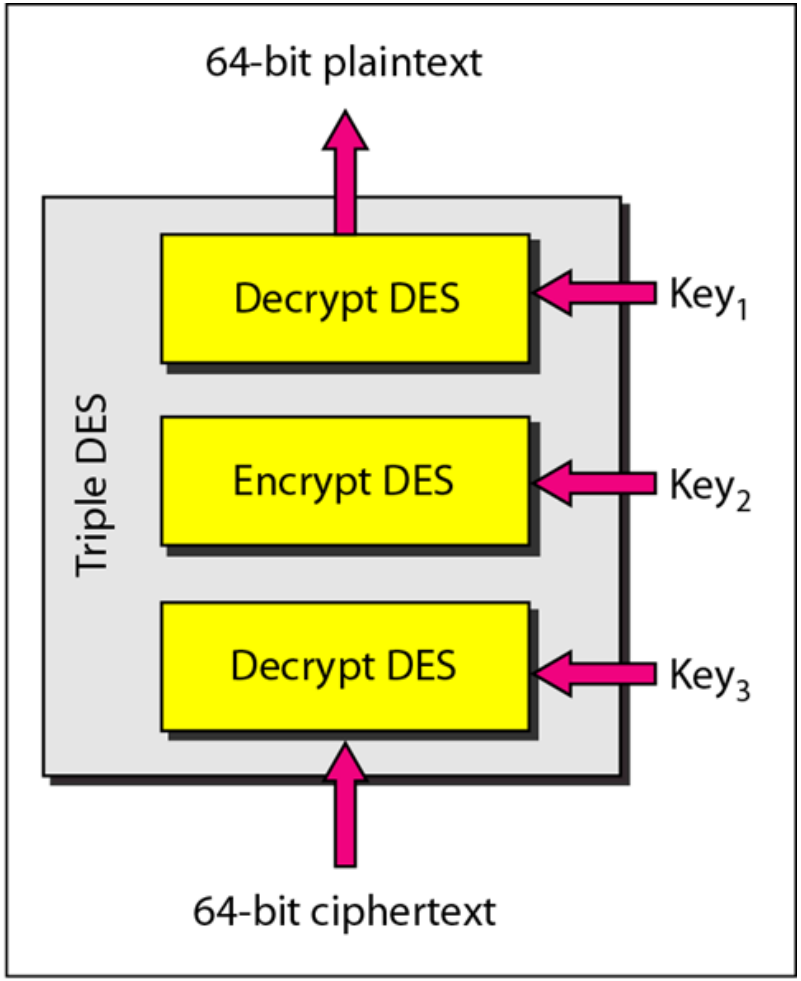
- has been proposed and implemented to lengthen the DES key
- uses three DES blocks
  - *encryption:* DES encryption-decryption-encryption
  - *decryption:* DES decryption-encryption-decryption
- different variants:
  - 3DES with three keys $\rightarrow$ the size of the key is increased to 168 bits
  - 3DES with two keys ($Key_1 = Key_3$) $\rightarrow$ the size of the key is 112 bits
  - if ($Key_1 = Key_2 = Key_3$) $\Rightarrow$ standard DES

Symmetric Cryptography

Triple DES (3DES)



a. Encryption Triple DES



b. Decryption Triple DES

Symmetric Cryptography

AES (Advanced Encryption Standard)

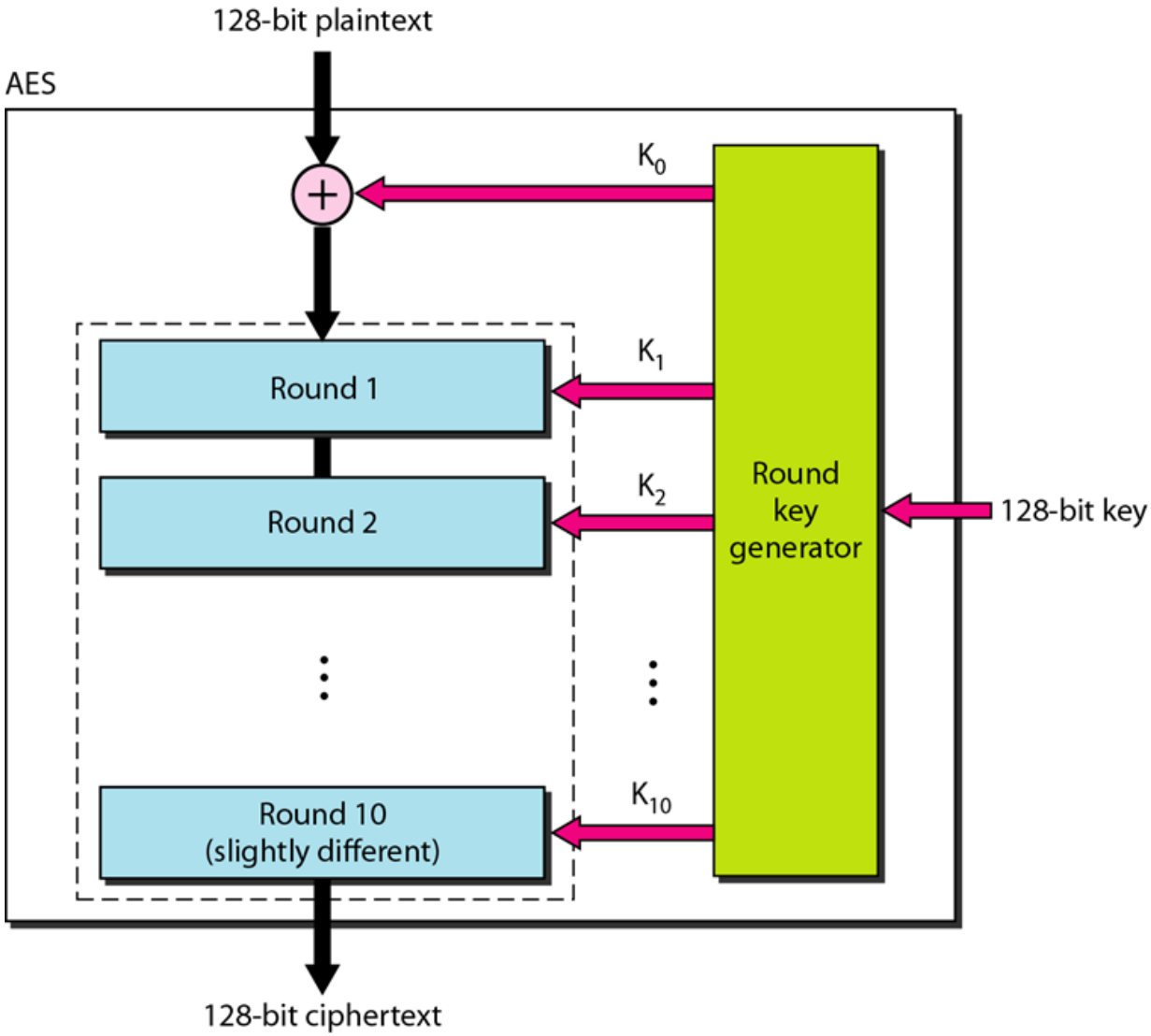
AES (Advanced Encryption Standard)

- proposed as a reaction to competition for selecting a successor to DES (1997)
  - originally called *Rijndael*
  - standardized in 2001
- three variants: AES-128, AES-192 and AES-256
  - the number indicates the key length
  - the keysize has no theoretical maximum
- operates on a 4×4 array of bytes ($= 128$ bits)
  - AES performs a number of repetitions of transformation rounds that convert the input plaintext into the final output of ciphertext
    - each round consists of several processing steps, including one that depends on the encryption key
- can be effectively implemented in HW

- see details in the animation:
  `http://www.formaestudio.com/rijndaelinspector/`

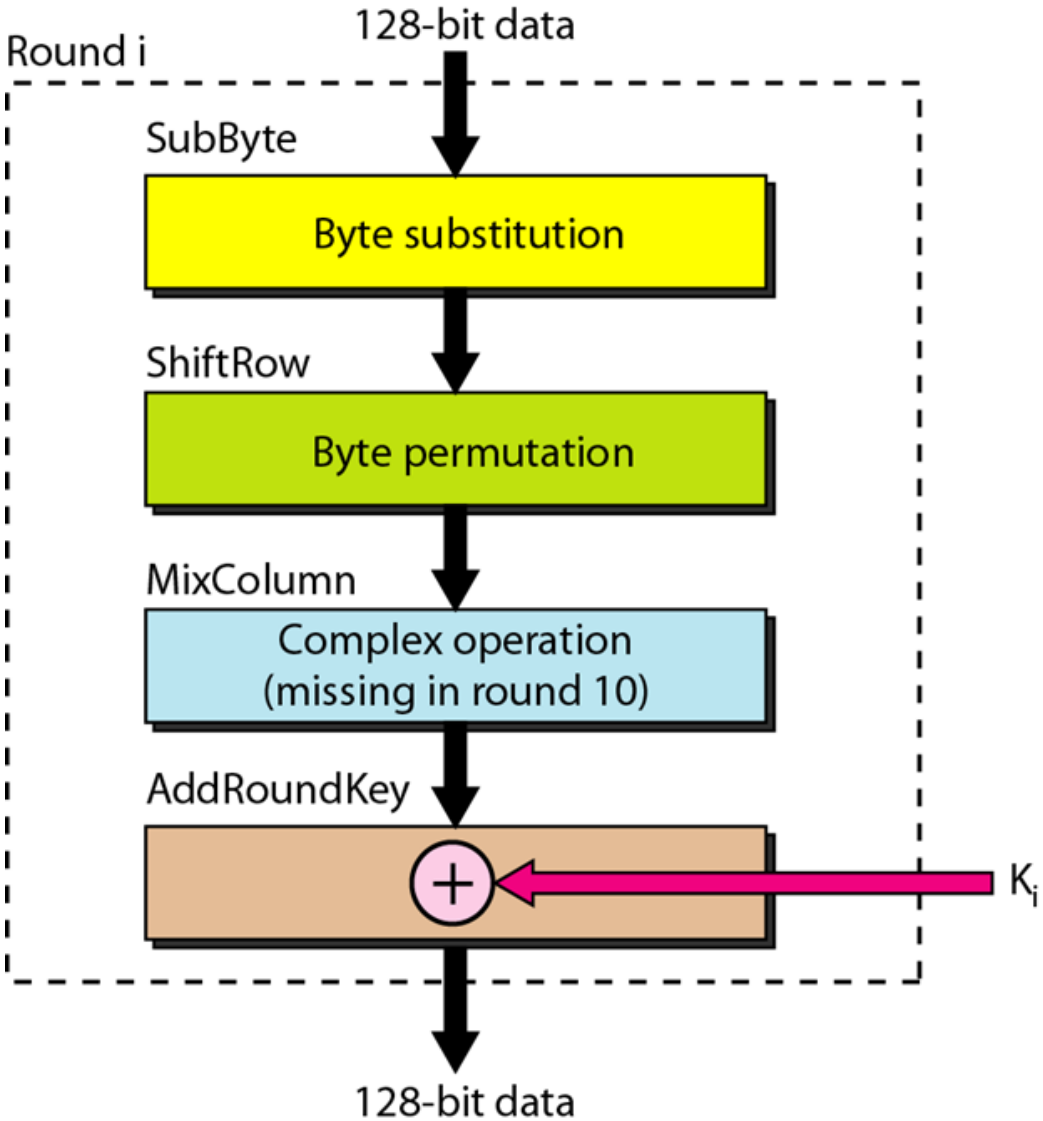
Symmetric Cryptography

AES (Advanced Encryption Standard) – Algorithm outline I.



Symmetric Cryptography

AES (Advanced Encryption Standard) – Algorithm outline II. (single round)

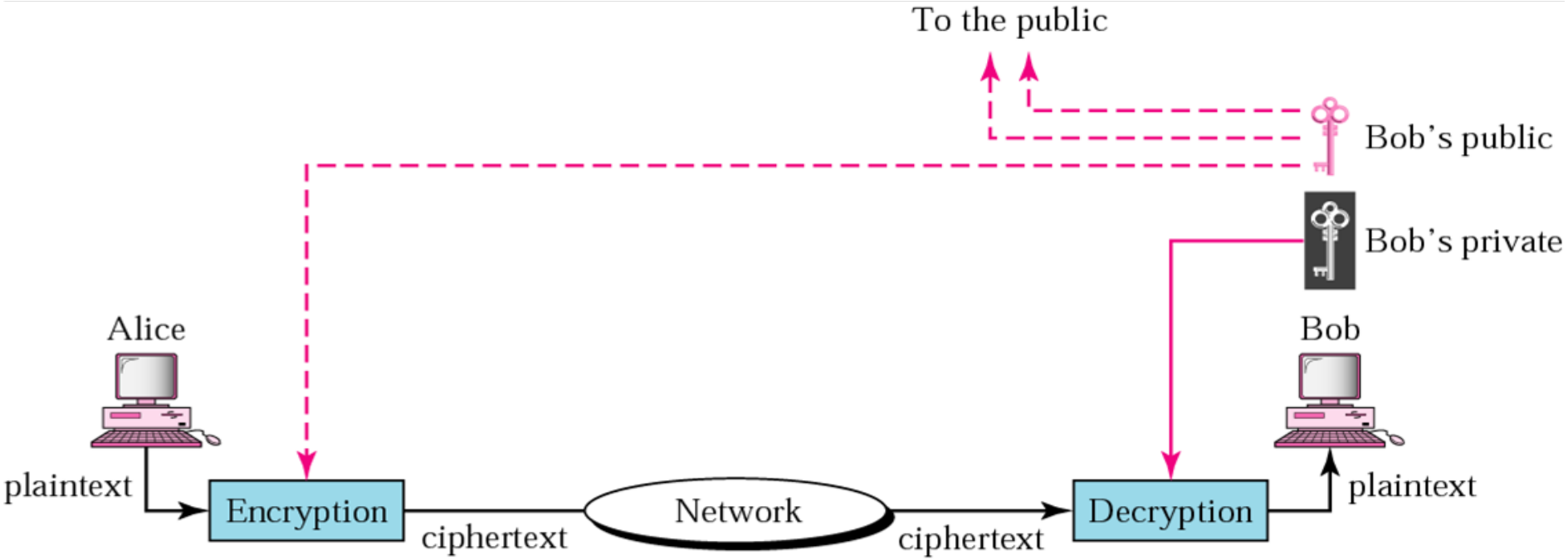


Asymmetric Cryptography

- also known as *Public-key Cryptography*
- uses different keys for encryption and decryption
  - both keys are denoted as *keypair*
  - the *public key* is used for encryption, while the *private key* is used for decryption
    - the message encrypted by a public key can be decrypted **only** by the appropriate private key
- advantages:
  - no secret key is passed between two communicating parties $\Rightarrow$ lowers the risk of its disclosure
  - the public key can be published
- drawbacks:
  - the speed $\Rightarrow$ asymmetric ciphers are suitable for small messages
    - for example, for a symmetric key exchange
  - suffer from the man-in-the-middle attack
    - the public keys have to be provided by a trusted party
- e.g., RSA, Diffie-Hellman (key exchange), etc.

Asymmetric Cryptography

Illustration



Asymmetric Cryptography

RSA (Rivest, Shamir, and Adleman algorithm) I.

RSA (Rivest, Shamir, and Adleman algorithm)

- the most common and popular public key algorithm
- RSA gets its security from the difficulty of factoring large numbers
  - uses two (large) numbers, e and d, as the encryption and decryption keys
- **public/private keys generation:**
  - choose two random large prime numbers, p and q (ideally of equal length, e.g., 200 digits)
  - compute the product $N = p \cdot q$
  - then randomly choose the *encryption key*, e (between 1 and $N - 1$), such that e and $(p - 1)(q - 1)$ are relatively prime
  - then compute the *decryption key*, d: $d = e^{-1} \ mod \ ((p - 1)(q - 1))$
    - i.e., $d \cdot e = 1 \ mod \ ((p - 1)(q - 1))$
  - p and q are no longer needed
- **public key:** (e, N)
- **private key:** (d, N)

Asymmetric Cryptography

RSA (Rivest, Shamir, and Adleman algorithm) II.

RSA (Rivest, Shamir, and Adleman algorithm) cont'd

- *message encryption*
  - to encrypt a message m, it has to be divided into numerical blocks smaller than N
  - each block is then individually encrypted: $c = m^e \ mod \ N$
- *message decryption*
  - decrypt every block by: $m = c^d \ mod \ N$

- see all the details in the RSA animation: `http://cryptool.org/media/RSA/RSA-Flash-en/player.html`

Lecture overview

Entity Authentication

Passwords

Password authentication:

- the simplest and the oldest method
- Alice authenticates to Bob by sending a password
- *fixed passwords* – subjects to several attacks:
  - eavesdropping, stealing a password, accessing a file with passwords, guessing (dictionary attacks), etc.
  - can be (partially) solved by:
    - hashing the passwords
    - hashing salted passwords
  - e.g., PAP (Password Authentication Protocol) – transmits unencrypted ASCII passwords (assumes out-of-band control channel)
- *one-time passwords* – passwords valid just for only one login session
  - new password generated based on previous password
  - new password generated based on a challenge
  - specialized electronic tokens, out-of-band channels (SMS), etc.

Entity Authentication

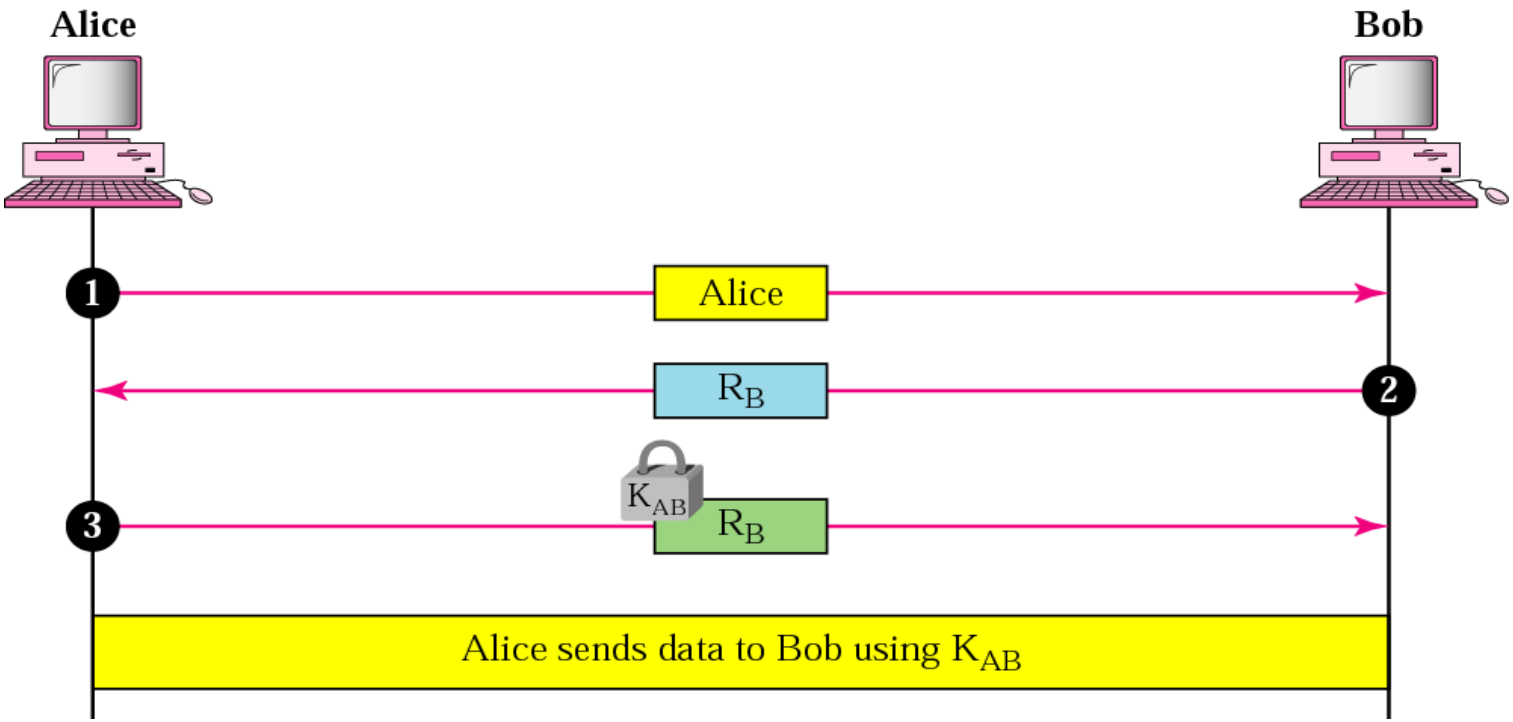
Challenge-Response Authentication

- in password authentication, the claimant proves her identity by demonstrating that she knows a secret (the password)
  - however, since the claimant reveals this secret, the secret is *susceptible to interception* by the adversary
- in **challenge-response authentication**, the claimant *proves that she knows a secret without revealing it*
  - the challenge is a time-varying value (a random number or a timestamp) sent by the verifier
  - the claimant applies a function to the challenge and sends the result (response) to the verifier
    - the response shows that the claimant knows the secret

Entity Authentication

Challenge-Response Authentication – Symmetric keys using a nonce

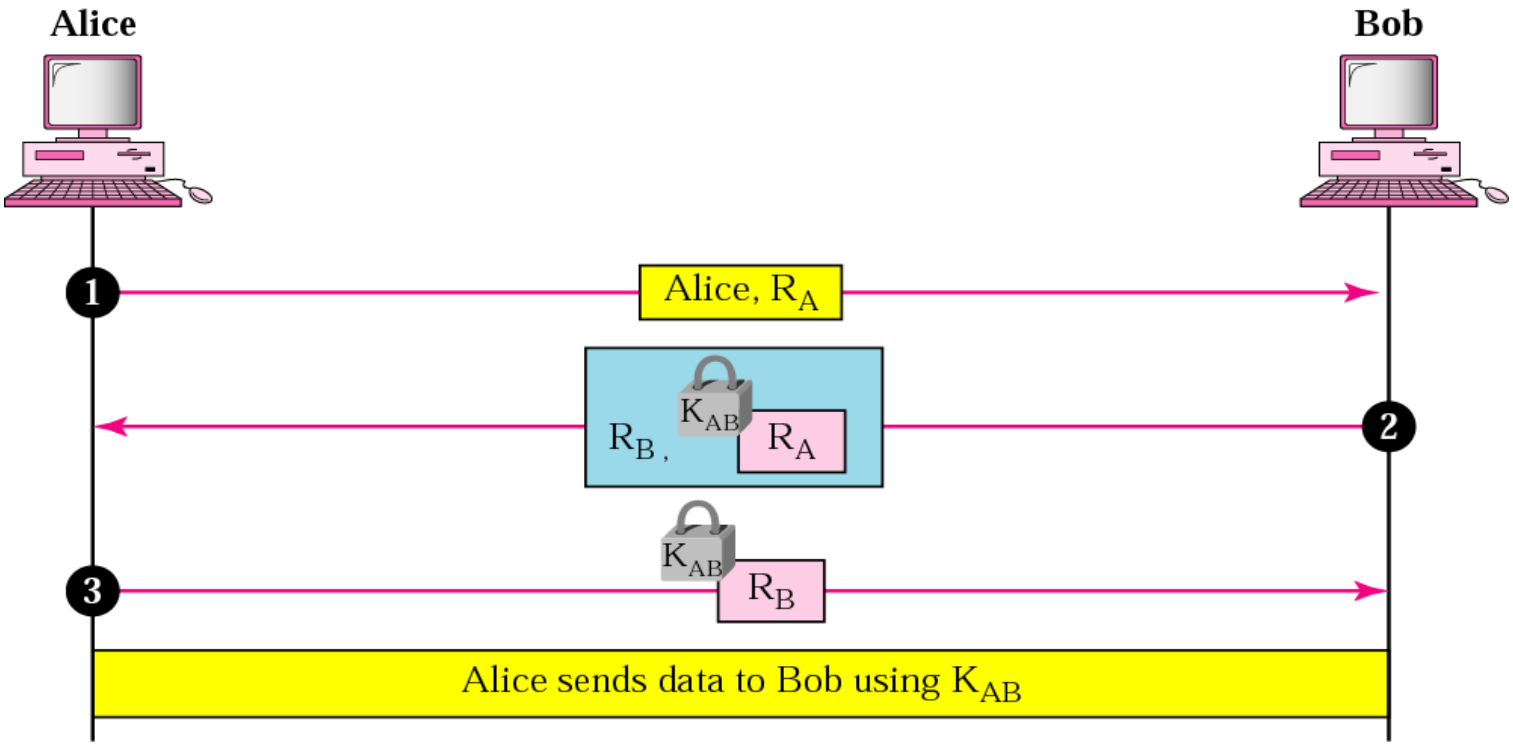
- Alice asks Bob for sending a *nonce* (a random number, timestamp, etc.)
- Alice encrypts the nonce by shared secret key (the password) and sends it back to Bob



Entity Authentication

Challenge-Response Authentication – Symmetric keys using a nonce (mutual auth.)

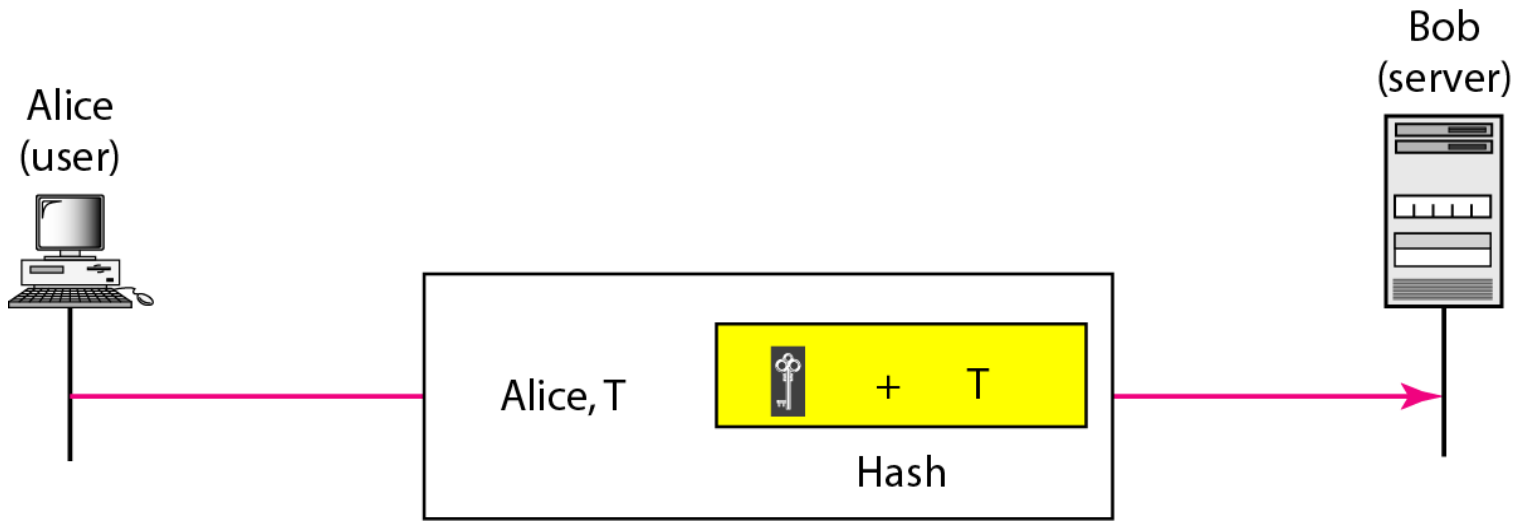
- same as the previous one, but with the mutual authentication



Entity Authentication

Challenge-Response Authentication – Hash functions usage

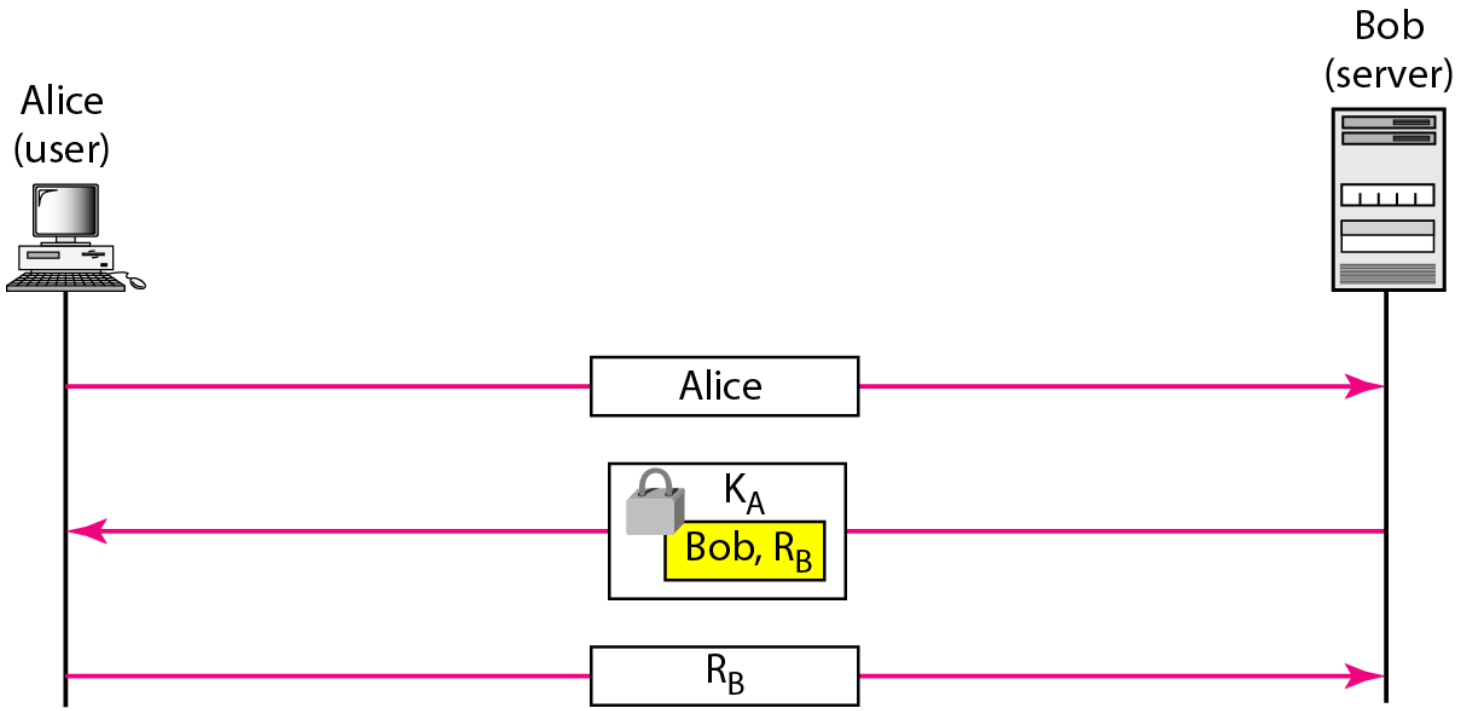
- instead of encryption/decryption, one can also use a hash function for entity authentication
  - hash function – a one-way function generating a message digest
  - details on hash functions – see Message Integrity part
- e.g., MS CHAP (Challenge-Handshake Authentication Protocol)



Entity Authentication

Asymmetric keys – Public key usage

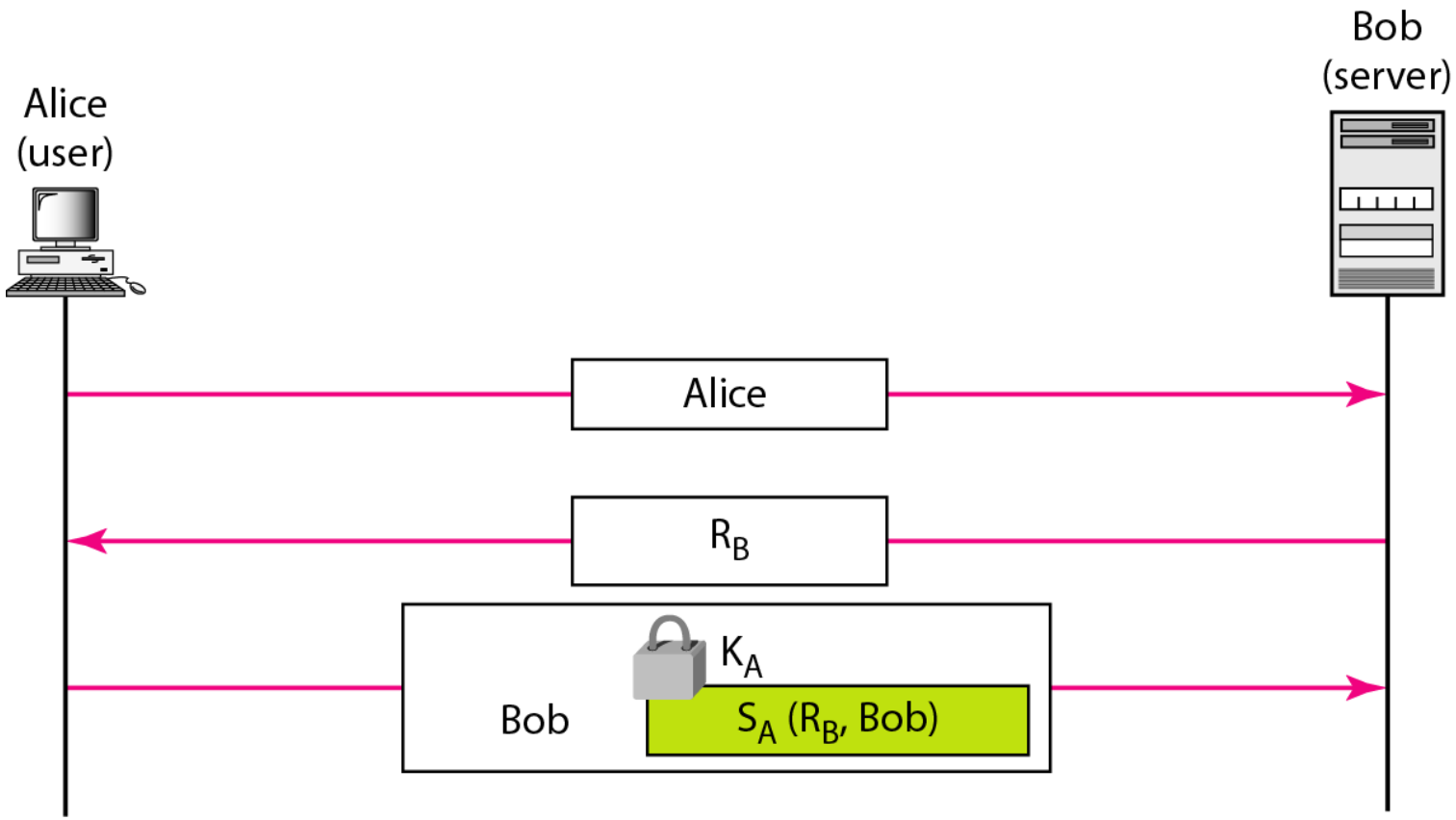
- instead of symmetric-key ciphers, asymmetric-key ciphers can be also used for entity authentication
- the secret is the private key of the claimant
  - she must show that she owns the private key related to the public key
- the nonce can be encrypted either by the claimant's *public key*



Entity Authentication

Asymmetric keys – Private key usage

- or by her *private key* (= digital signatures, see later)

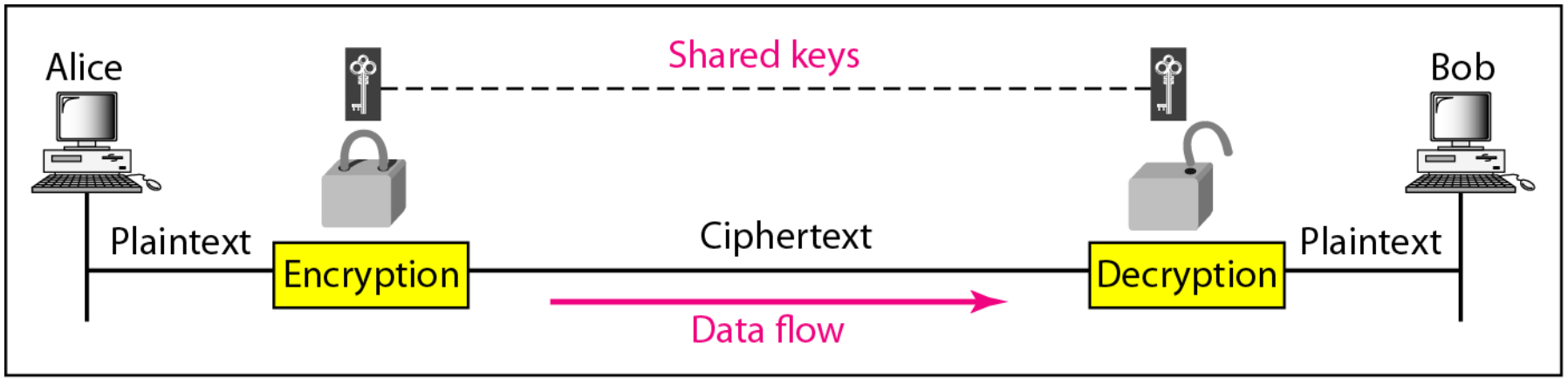


Message Confidentiality/Security – Data Encryption

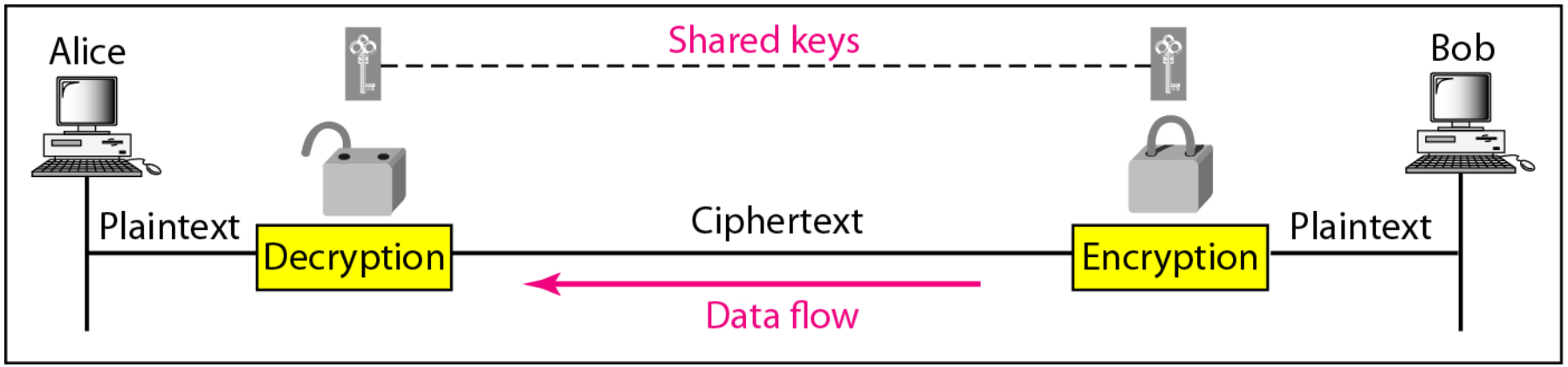
- the transmitted data are usually (especially for the long messages) encrypted using a *symmetric cryptography algorithm*
  - because of their speed
  - the shared secret key necessary for the encryption can be (prior to the transmission) obtained using:
    - *asymmetric cryptography* – the chosen shared key is encrypted using the public key of the communicating partner and delivered to her
    - *Diffie-Hellman algorithm* – shared key computation

Message Confidentiality/Security – Data Encryption

Symmetric Cryptography



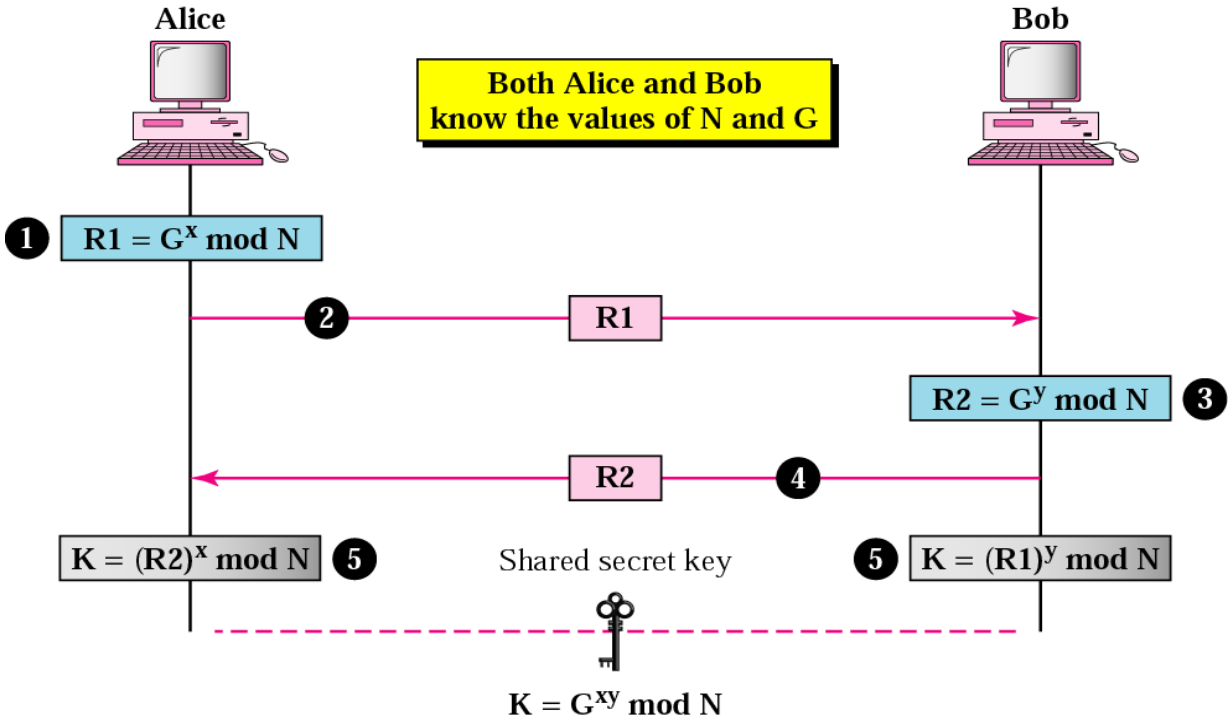
a. A shared secret key can be used in Alice-Bob communication



b. A different shared secret key is recommended in Bob-Alice communication

Message Confidentiality/Security – Data Encryption

Diffie-Hellman algorithm for session key generation I.

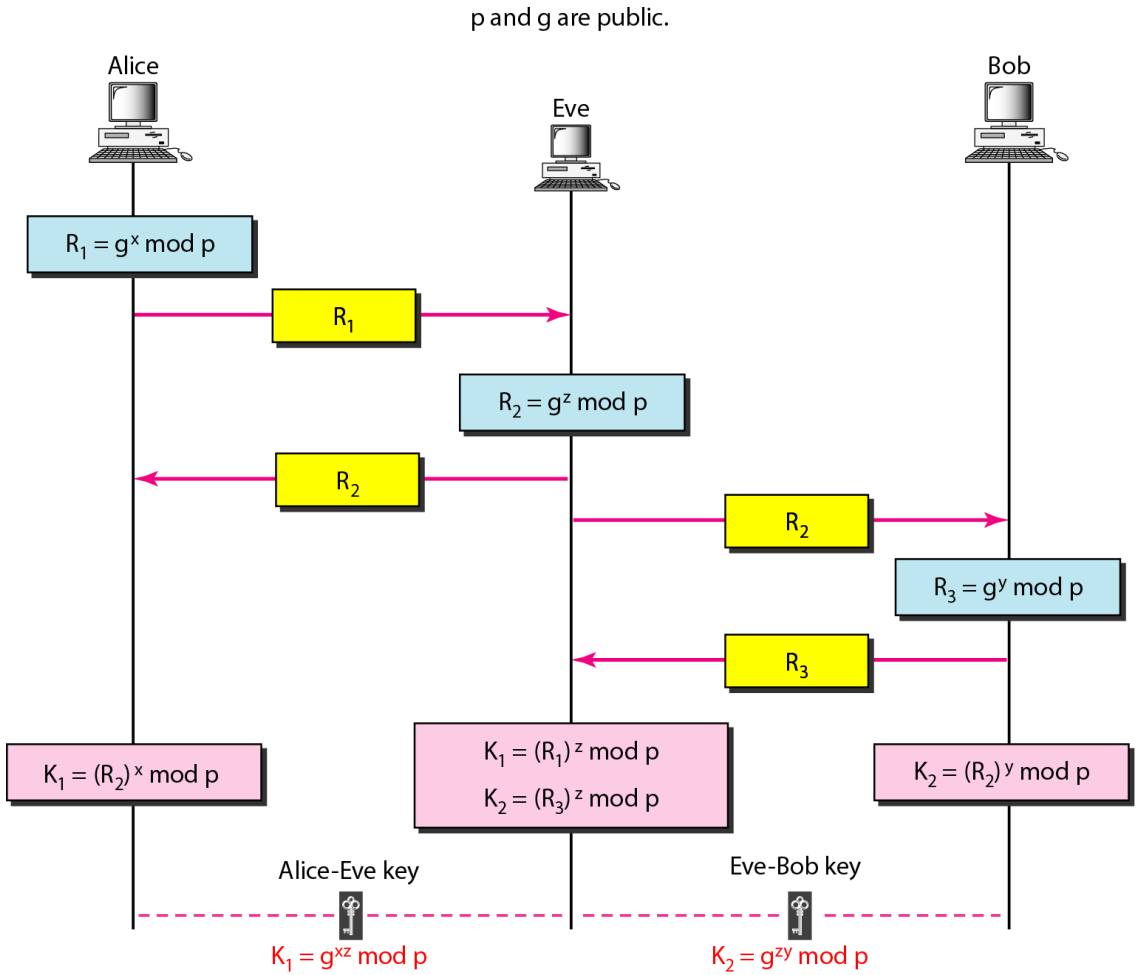


- G and N are prime numbers, which both parties have to agree on
  - can be freely transmitted through the network
- the x and y are (private) large numbers chosen by the parties independently on each other
- the computed K becomes the shared key used for the following symmetric encryption
- employed principle: $(G^x \ mod \ N)^y \ mod \ N = (G^y \ mod \ N)^x \ mod \ N = G^{xy} \ mod \ N$

Message Confidentiality/Security – Data Encryption

Diffie-Hellman algorithm for session key generation II.

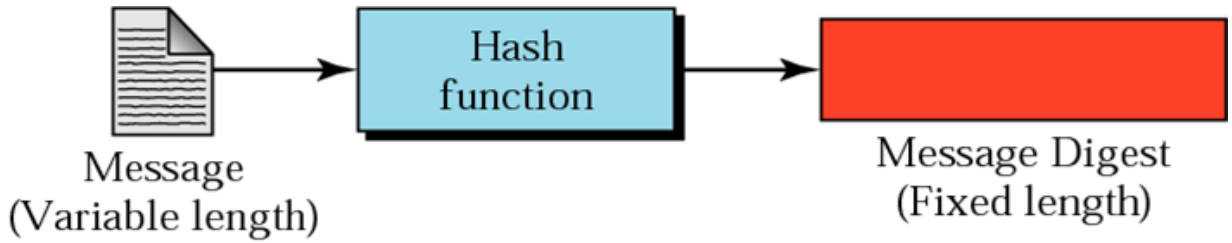
- a weakness – suffers from a *man-in-the-middle attack*
  - can be avoided if Bob and Alice first authenticate each other



Message Integrity

Hash functions

- encryption provides confidentiality, but not *integrity*
  - sometimes, the message does not need to be hidden (encrypted) – the only need is to keep it safe from any tampering
- to preserve the integrity of a message, the message is passed through a *hash function*
  - an algorithm that creates a compressed "image" (fingerprint) of the message
    - so-called a *message digest* or a *modification detection code (MDC)*
    - the digest needs to be kept secret



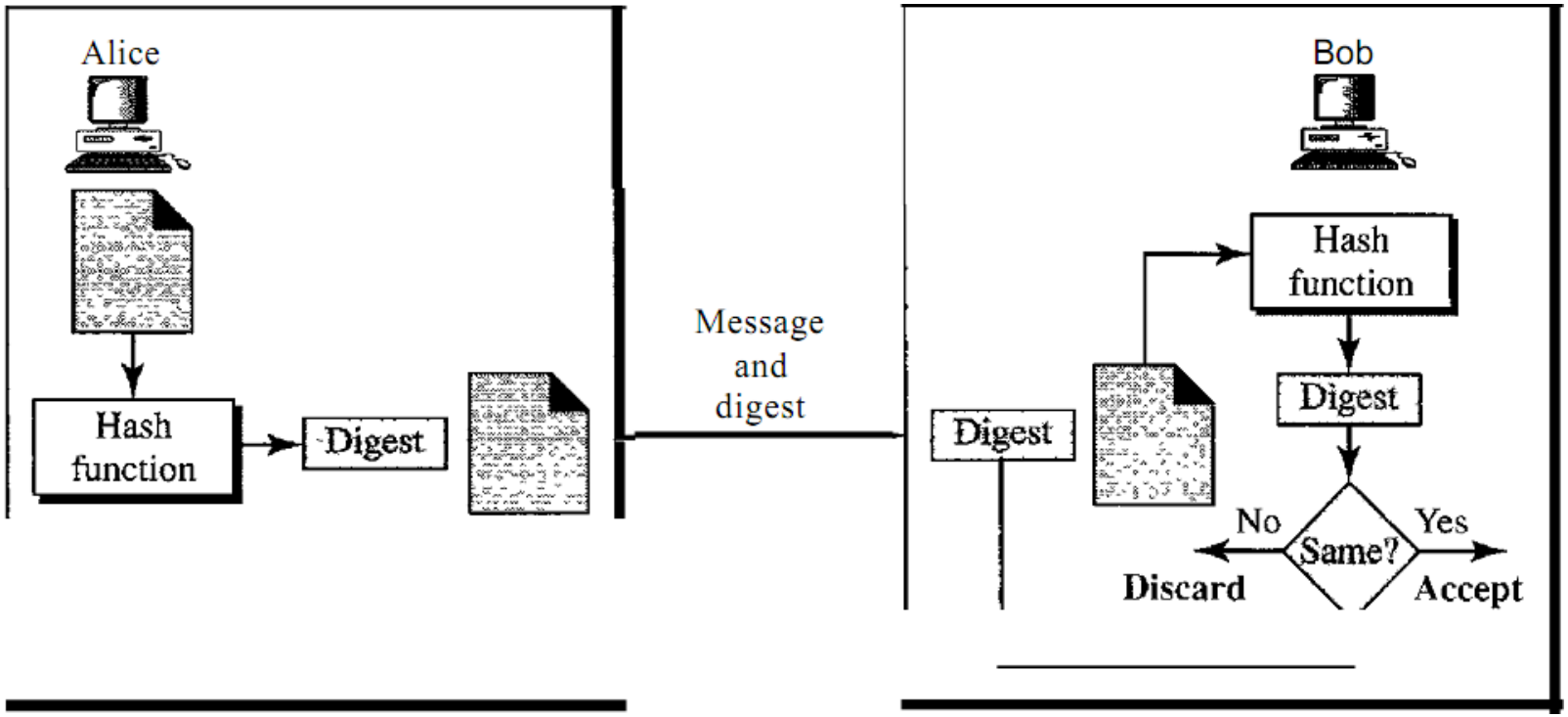
Message Integrity

Hash function criteria

- to be eligible for a hash, a function needs to meet the *four criteria*:
  - **one-wayness** – a message digest is created by a one-way hashing function
    - it must be impossible to recreate the message from the digest
  - **resistance to weak collision** – ensures that a message cannot easily be forged
    - one cannot easily modify the original message so that the new message hashes exactly to the same digest
  - **resistance to strong collision** – ensures that we cannot find two messages that hash to the same digest
    - called "strong" because the probability of a collision is higher than in the previous case
  - **computation ease** – it must be easy to compute the hash value for any given message
- hash function examples:
  - MD5 (broken $\Rightarrow$ not secure), SHA-1, SHA-256, SHA-512, etc.

Message Integrity

Message integrity check



Message Authentication

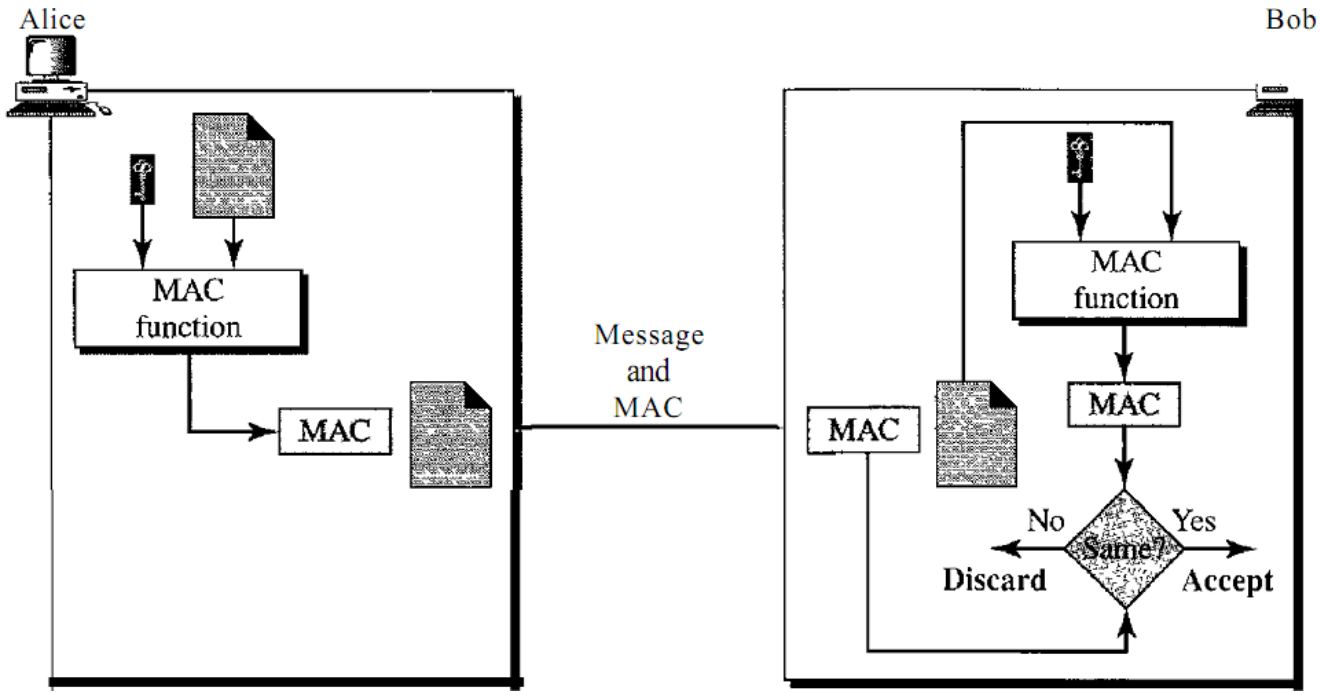
Message Authentication

- a process assuring the recipient that the message is from the source that it claims to be from
- can be provided by:
  - symmetric cryptography – keyed hash functions, HMAC
  - asymmetric cryptography – digital signature

Message Authentication

Symmetric cryptography

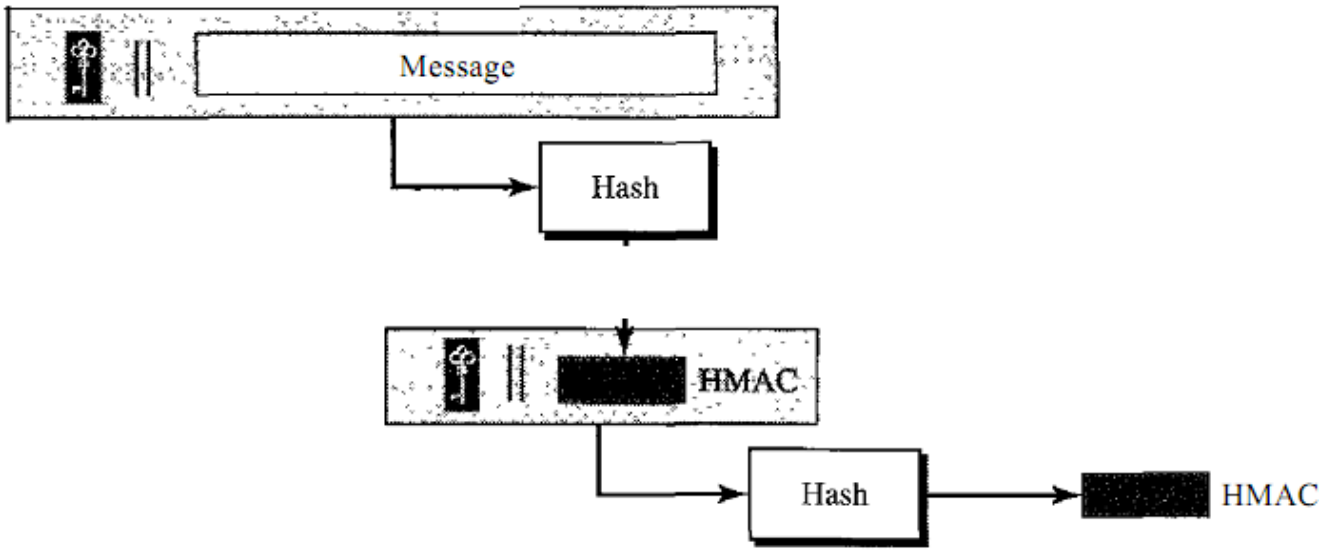
- **keyed hash functions** are used
  - when creating the digest, such a function includes the symmetric key between the sender and receiver into the computation
  - $\Rightarrow$ so-called *Message Authentication Code (MAC)* is computed
  - besides the message authentication, the message integrity is obviously ensured as well



Message Authentication

Symmetric cryptography – HMAC

- instead of keyed hash functions, a keyless hash function can also be used for message authentication
- = **hashed MAC (HMAC)**
  - creates a MAC by applying a keyless hash function to the concatenation of the message and a symmetric key
    - i.e., a copy of the symmetric key is prepended to the message and this combination is hashed using a keyless hash function (e.g., SHA-1, SHA-256, etc.)



Digital Signature

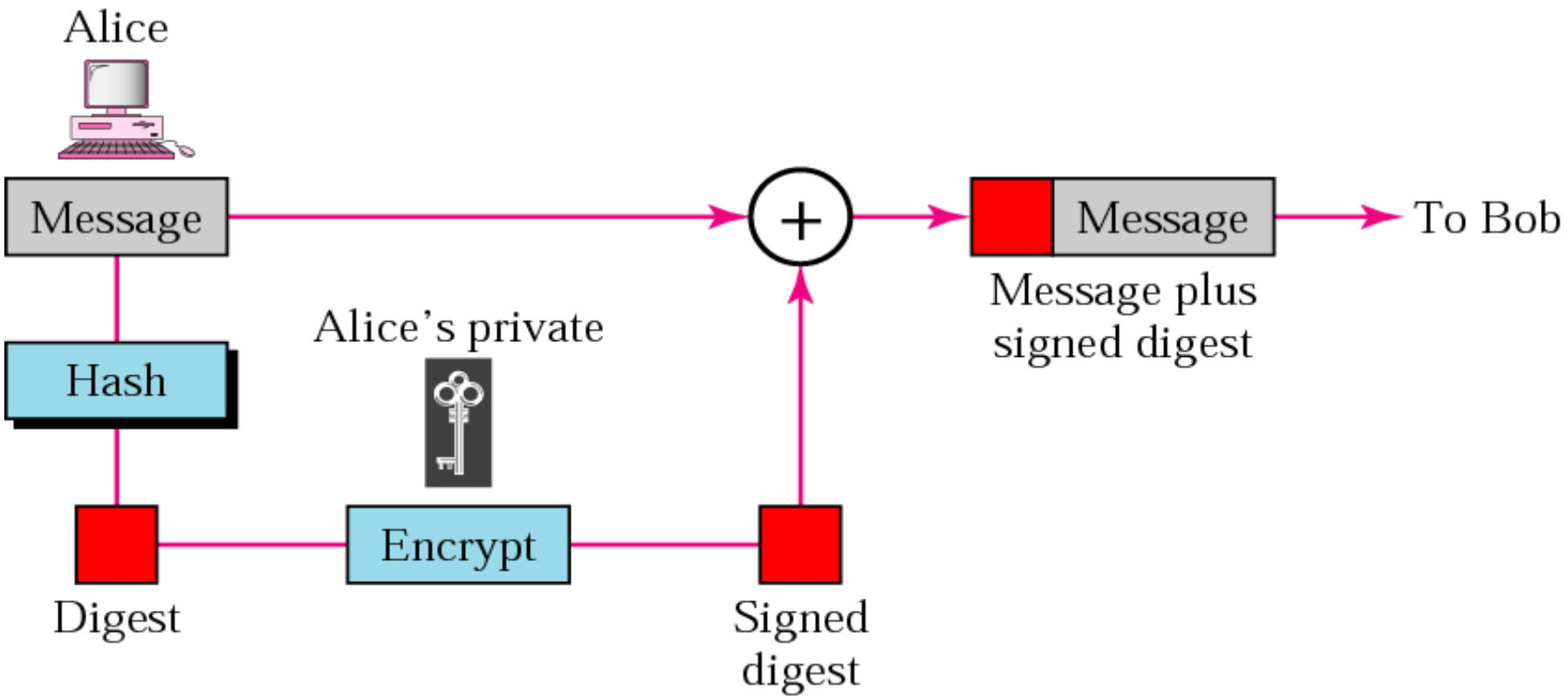
- MAC's/HMAC's main drawback: they require a symmetric key to be established between the sender and the receiver
- $\Rightarrow$ *Digital Signatures* – employ asymmetric cryptography to provide message authentication and integrity
  - besides, the *message nonrepudiation* is also provided

Digital Signature

- a message is "signed" (= encrypted) using the sender's private key
- and "verified" (= decrypted) using the sender's public key
- can be achieved in two ways:
  - *signing the whole document* – easier, but less efficient (for long messages)
  - *signing a digest of the document*
    - most commonly used
    - instead of signing the whole document, just the message's digest is signed

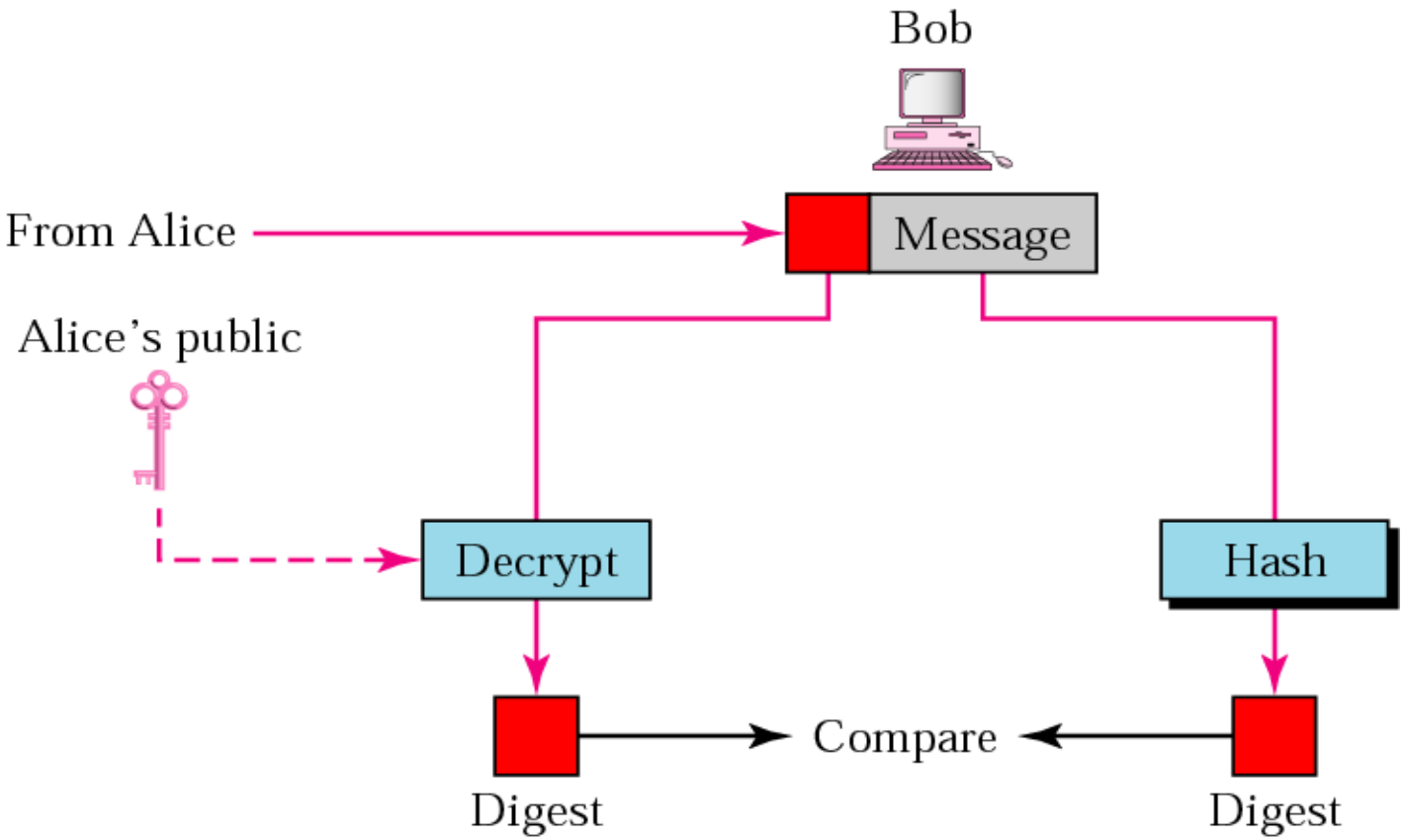
Digital Signature

Sender's side



Digital Signature

Receiver's side



Digital Signature

Summary

Remember

In a cryptosystem (to provide *message confidentiality*), we use the private and public keys of the **receiver**; in digital signatures (to provide *message authentication/integrity/nonrepudiation*), we use the private and public keys of the **sender**.

- to combine both, the message is signed at first and later both the message and the signature are encrypted

Key Management and Distribution

How secret keys in symmetric-key cryptography and how public keys in asymmetric-key cryptography are distributed and maintained?

Symmetric-key Distribution

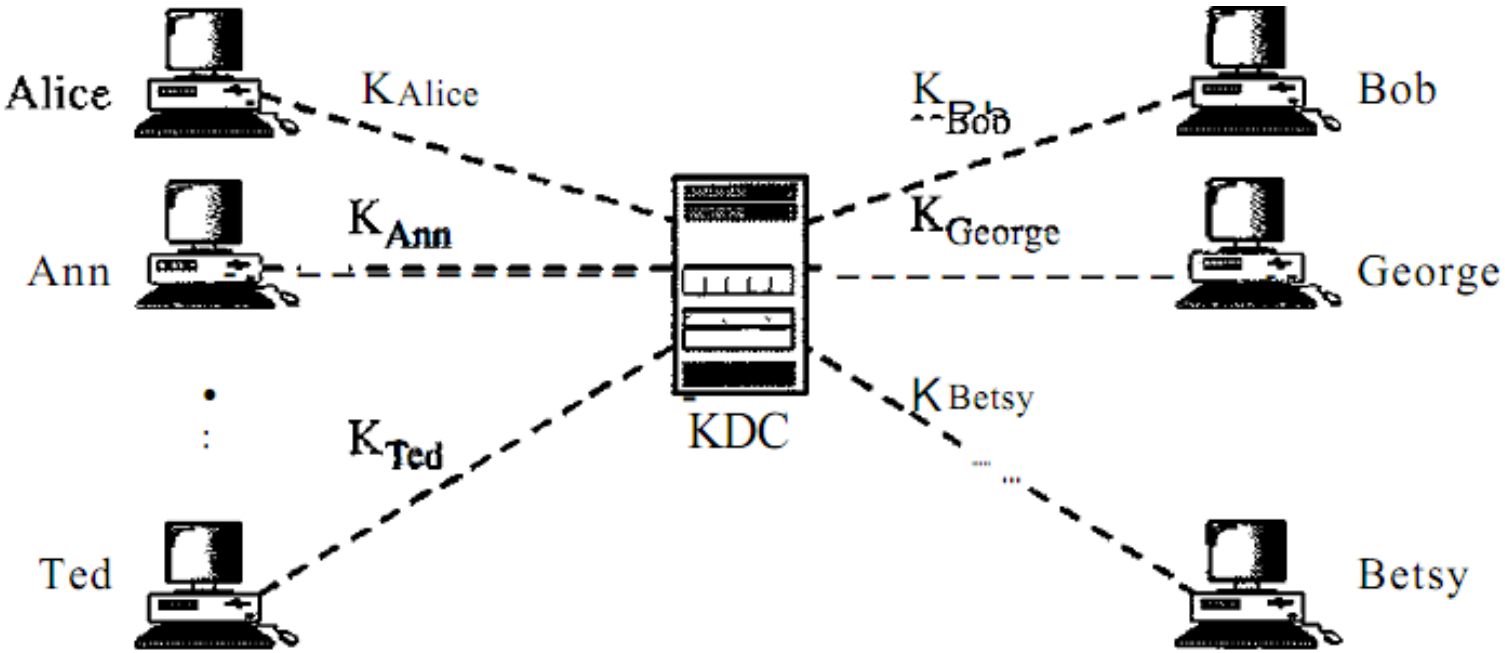
- the symmetric key needs to be shared between every two communicating parties
  - if Alice needs to exchange confidential messages with N people – she needs to maintain N different keys
  - if N people need to communicate with one another – a total of $N \cdot (N-1)/2$ keys is needed
    - (each person needs to maintain $N-1$ keys; since the keys are shared, we need "only" $N \cdot (N-1)/2$ keys)
- moreover, the keys' distribution is another problem
- $\Rightarrow$ a trusted party – *Key Distribution Center (KDC)* – is needed to reduce the number of keys

Symmetric-key Management and Distribution

Key Distribution Center (KDC) I.

Key Distribution Center (KDC)

- each person/entity establishes a shared secret key with the KDC



Symmetric-key Management and Distribution

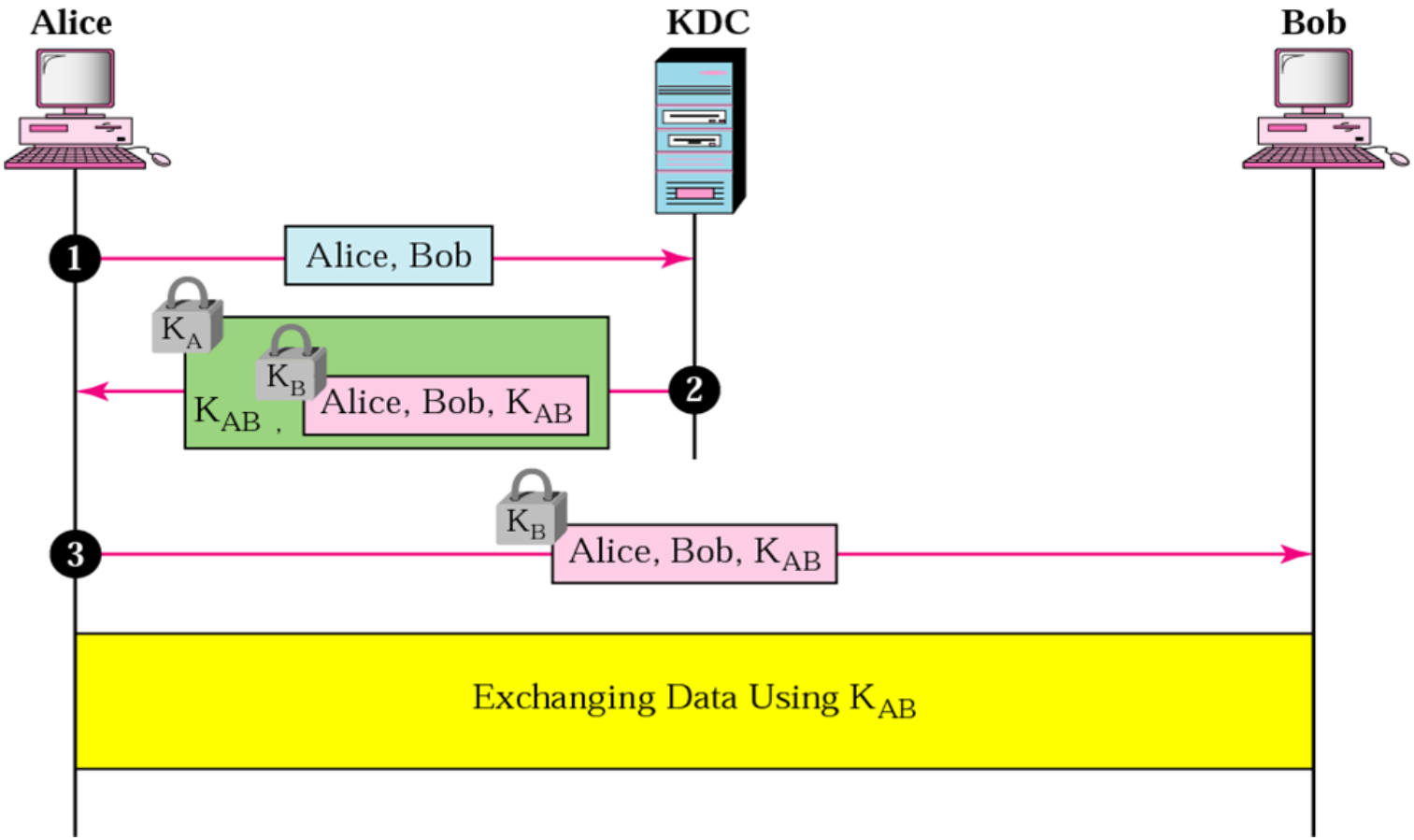
Key Distribution Center (KDC) II.

Key Distribution Center (KDC) cont'd.

- the process of sending a confidential message from Alice to Bob may proceed as follows:
  1. Alice sends a request to KDC, stating that she needs a *session (temporary) secret key* between herself and Bob
  2. KDC informs Bob of Alice's request
  3. If Bob agrees, a session key is created between the two
- once the communication is terminated, the session key is no longer valid
- several approaches of creating the session key based on the previous ideas have been proposed
  - e.g., basic approach, Needham-Schroeder protocol, Otway-Rees protocol, etc.
  - for the following three slides, the following holds:
    - K_A (K_B) . . . a symmetric key known only to A and S (resp. B and S)
    - R, R_A, R_B, R_1, R_2 . . . nonces
    - K_{AB} . . . the symmetric session key
- example real implementation – Kerberos

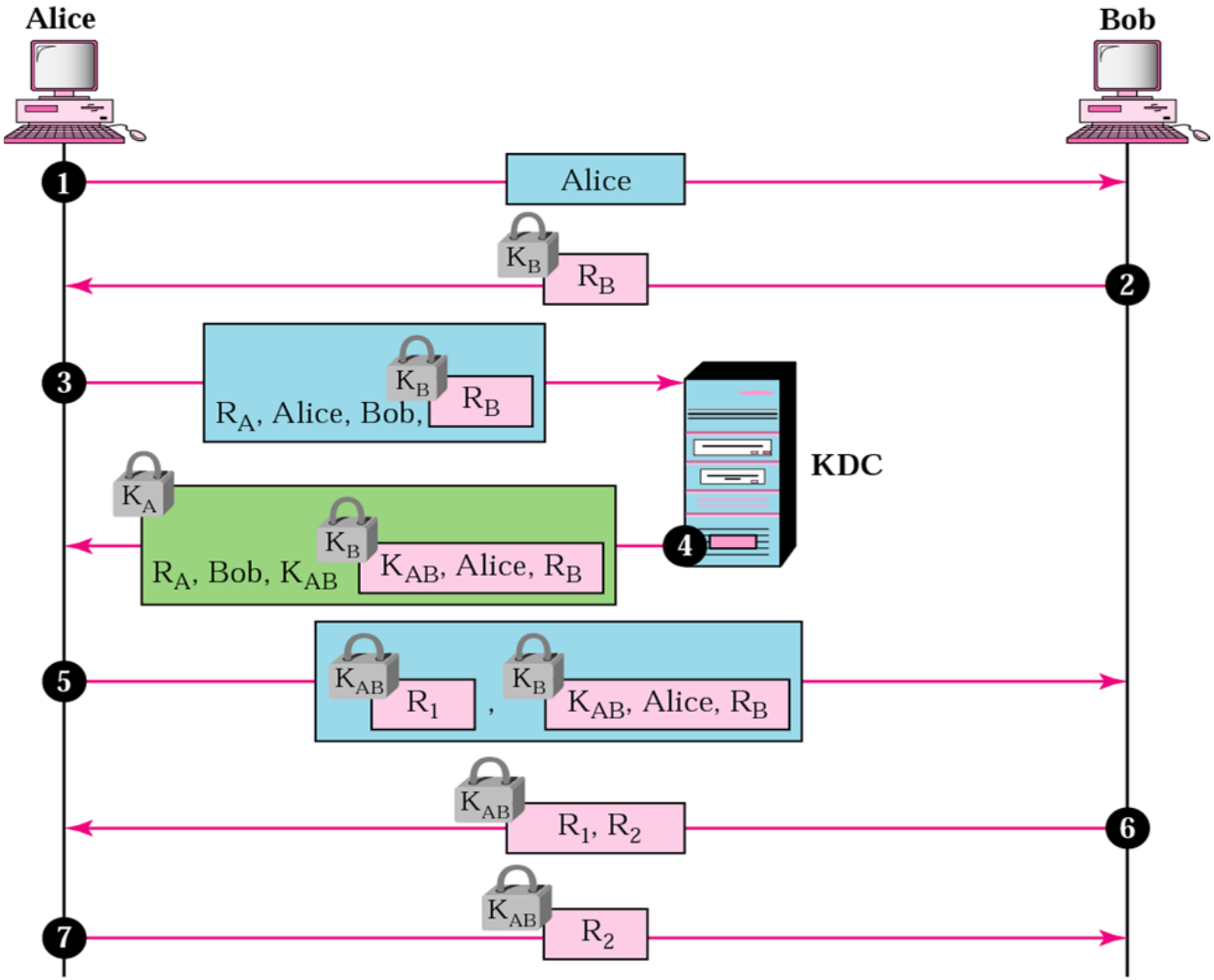
Symmetric-key Management and Distribution

Key Distribution Center (KDC) III. – Basic Approach



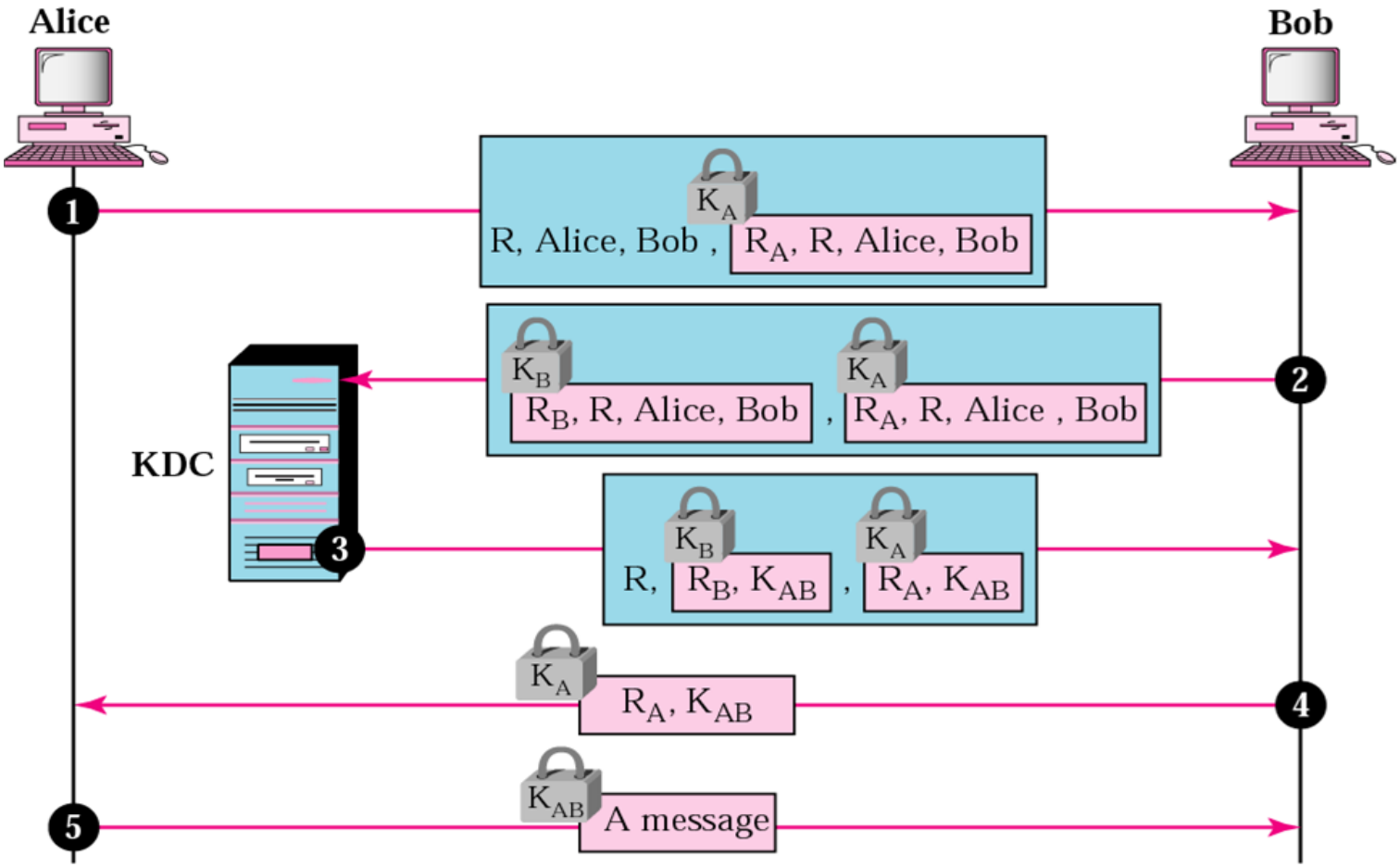
Symmetric-key Management and Distribution

Key Distribution Center (KDC) III. – Needham-Schroeder protocol



Symmetric-key Management and Distribution

Key Distribution Center (KDC) III. – Otway-Rees protocol



Public-key Management and Distribution

Public-key Distribution

- in public-key cryptography, everyone shields a private key and advertises a public key
  - the public-key is freely available

- *Why a public-key distribution mechanism is needed, when the public-keys can be freely announced?*
  - to protect the keys from forgery
  - the mechanism: the use of a trusted center – *Certification Authority*

Public-key Management and Distribution

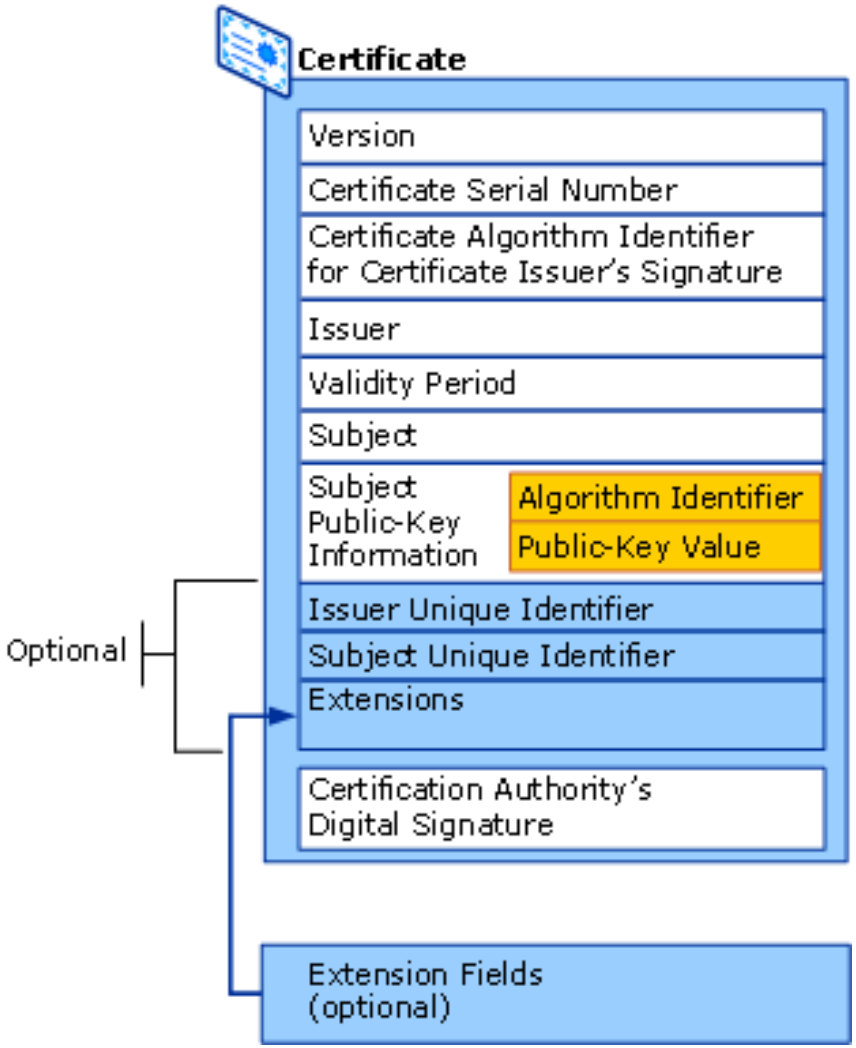
Certification Authority (CA)

Certification Authority (CA)

- issues so-called *public-key certificates*
  - bind an entity (human, server, etc.) with its public key
  - to prevent the certificate itself from being forged, the CA signs the certificate with its private key
  - such a certificate can be freely published and distributed
    - anyone who wants to extract the public key from the certificate uses the CA's public key to extract it
  - when compromised, the previously issued certificates can be revoked
  - example: X.509 certificates
- supports hierarchical structure to allow the use in the wide Internet
  - (= hierarchical trust model)
  - called *Public Key Infrastructure (PKI)*

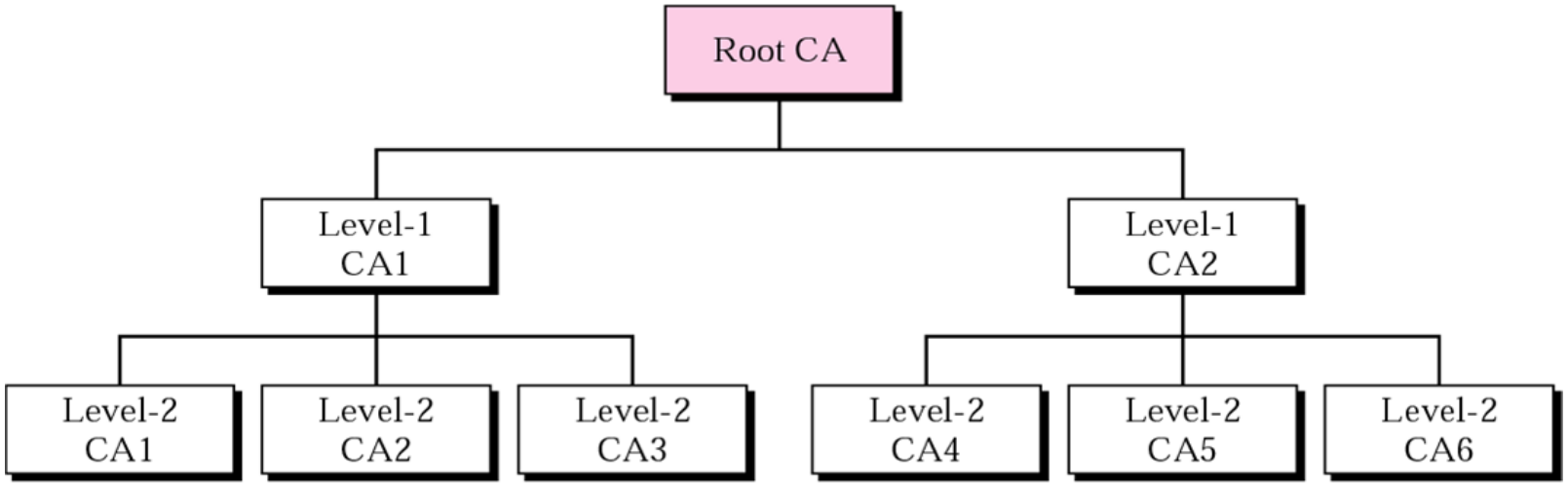
Public-key Management and Distribution

X.509 Certificates



Public-key Management and Distribution

Public Key Infrastructure (PKI)



- **One of many definitions:** *a set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke certificates based on public key cryptography*
- each certificate is issued (and signed) only by a single party (= CA)
  - each CA certifies only its children
  - the CA's certificate itself may be signed by a different CA (all the way up to a "self-signed" root certificate)
  - $\Rightarrow$ root certificates must be available to those who use a lower level CA certificate (root certificates are typically widely distributed)

Lecture overview

Example Protocols Ensuring Network Security in the Internet

- all the presented concepts can be realized on:
  - *application layer* – security mechanisms provided by applications (application protocols) themselves
    - e.g., PGP, SET, S/MIME, S-HTTP, etc.
  - *transport layer* – SSL and TLS
  - *network layer* – IPv6 & IPSec (IPv4)
  - *link layer* – e.g., PPP

Application-layer Security

Pretty Good Privacy (PGP)

Pretty Good Privacy (PGP)

- a de-facto standard proposed by Phil Zimmermann (1991)
- the mechanism providing *secure email communication* channels
- originally uses an alternative approach to PKI for distributing user's public keys – *Web of Trust*
  - decentralized trust model
  - identity certificates are digitally signed by other users who, by that act, endorse the association of that public key with the person/entity listed in the certificate
    - $\Rightarrow$ leaves the trust decisions in the hands of individual users
    - there are several levels of confidence which can be included in such signatures
  - later versions are able to use PKI as well
- involves all the basic security mechanisms:
  - email confidentiality (using a symmetric one-time key because of performance – 3DES, IDEA, etc.)
  - integrity (hash functions – MD5, SHA)
  - authentication (asymmetric cryptography – RSA)
  - non-repudiation (through the use of PKI)
- $\approx$ Digital Signatures

Application-layer Security – PGP

Sender's side

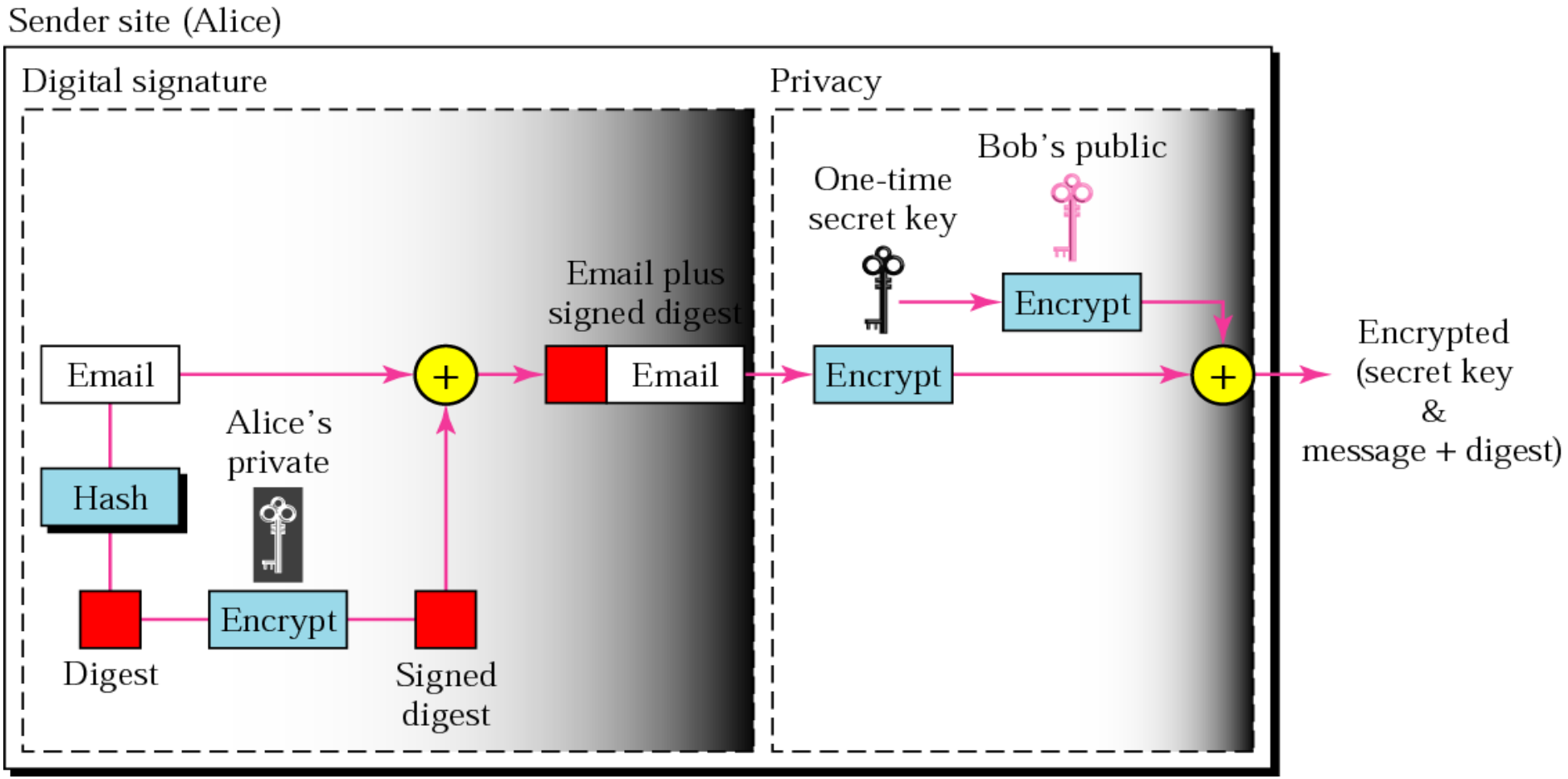


Figure : The mechanism of signing and encrypting the sending message (sender's side).

Application-layer Security – PGP

Receiver's side

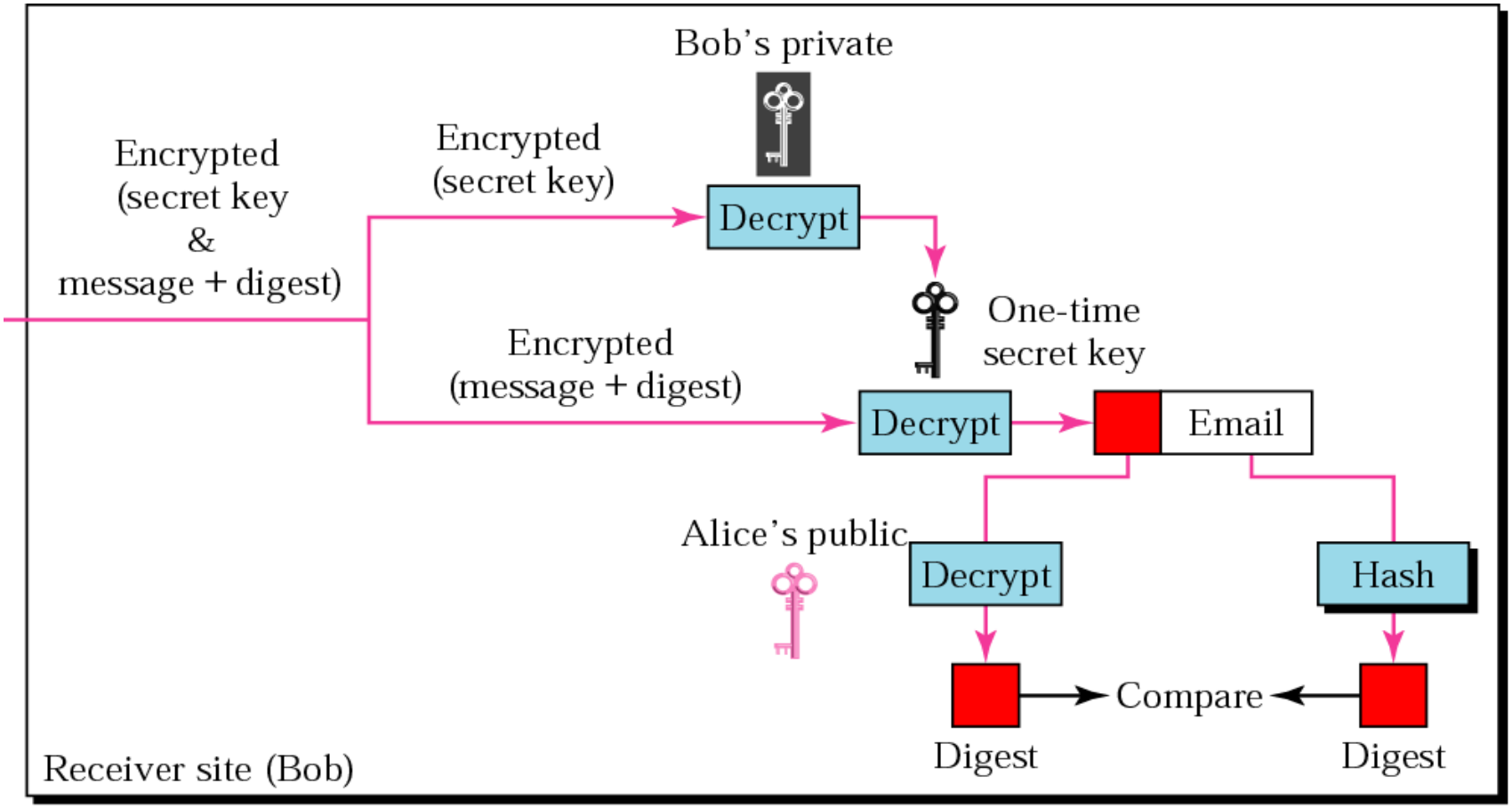


Figure : The mechanism of decrypting and verifying the received message (receiver's side).

Application-layer Security

Secure Electronic Transactions (SET) I.

Secure Electronic Transactions (SET)

- a specialized cryptographic protocol developed by a group of companies (Visa, Microsoft, IBM, MasterCard, etc.)
- designed for securing credit card transactions over insecure networks
  - NOT for a general encryption
- design goals:
  - confidentiality of payment and ordering information
  - integrity of all transmitted data
  - authentication of all card holders
  - ensuring that a merchant can accept card transactions based on buyer's relationship with her financial institution
    - the merchant does not need to know the buyer's identity
  - creating protocols that neither depend on transport security mechanism nor prevent their use

Application-layer Security

Secure Electronic Transactions (SET) II.

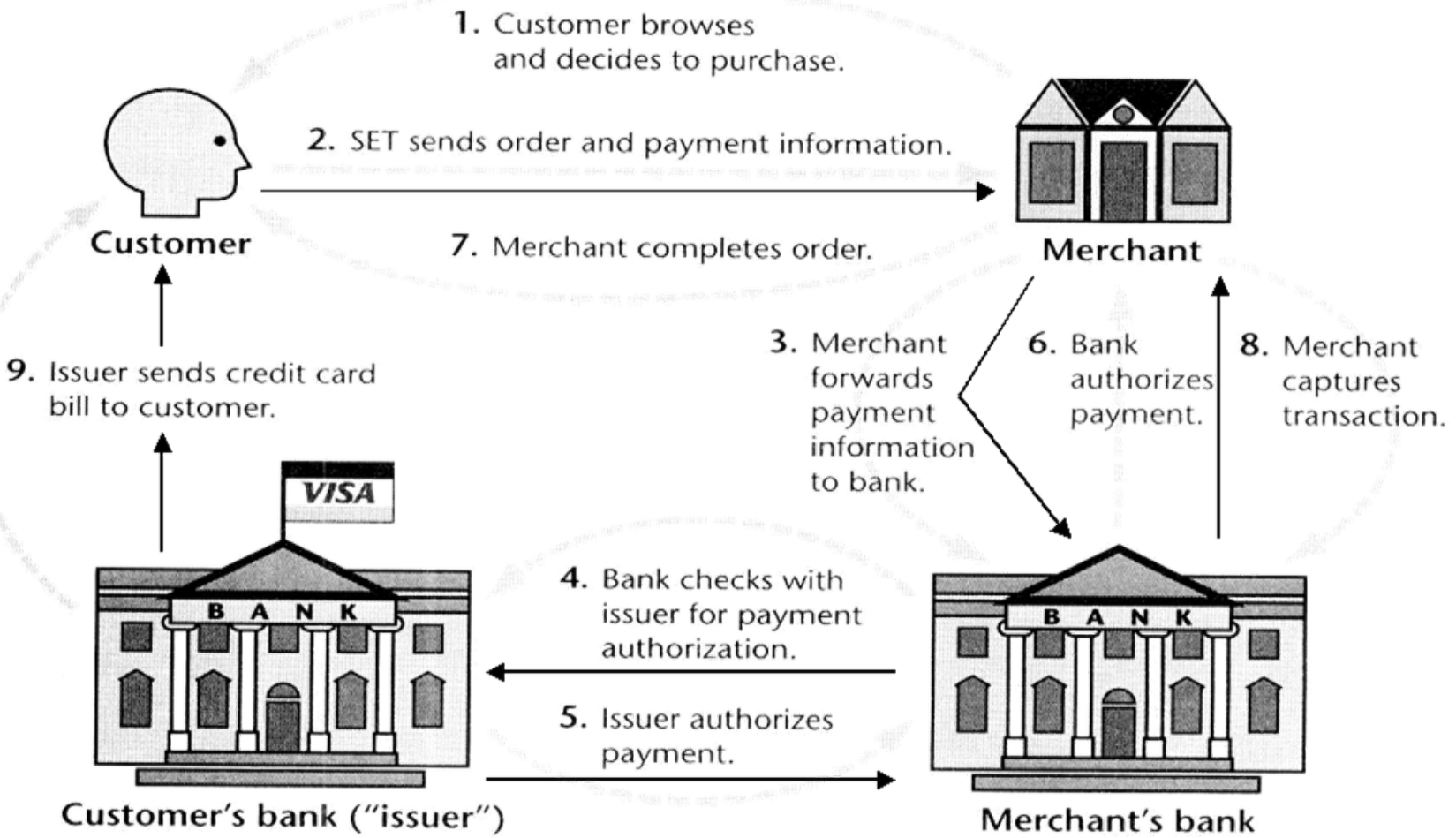


Figure : The SET transactions outline.

Application-layer Security

S/MIME and S-HTTP

Secure/Multipurpose Internet Mail Extension (S/MIME)

- a standard for public key encryption and signing of MIME data
  - MIME – Internet standard that extends the format of e-mail to support, e.g., text in character sets other than ASCII, non-text attachments (pictures, video, audio, etc.), message bodies with multiple parts, etc.
- provides authentication, message integrity and non-repudiation of origin (using digital signatures) and privacy and data security (using encryption)

Secure-HTTP (S-HTTP)

- extends the Hypertext Transfer Protocol (HTTP) to support encryptions and digital signatures
- extends the negotiation between the client browser and the server to include the negotiation for security matters
  - uses additional headers for message encryption, digital certificates, and authentication in the HTTP format
  - the HTTP headers are encapsulated into the S-HTTP headers
- has been bundled away by the HTTPs (HTTP over SSL/TLS)

Transport-layer Security – SSL/TLS I.

Secure Sockets Layer (SSL) / Transport Layer Security (TLS)

- the protocols for ensuring secure communication channels on the transport layer
  - (SSL precedes the TLS)
  - SSL 3.0 $\approx$ TLS 1.0 (but see *Poodle attack*)
- SSL/TLS provide secure and authenticated *data paths* between servers and clients:
  - *data encryption* – protect data in transport between the client and the server from interception (making them readable only by the intended recipient)
  - *server and client authentication* – the SSL uses standard public key encryption to authenticate the communicating parties to each other
  - *message integrity* – achieved through the use of session keys so that data cannot be intentionally/unintentionally tampered
- *main drawback:* the necessity to adapt the applications (applications' protocols)
  - HTTP $\rightarrow$ HTTPS (HTTP Secure)
  - FTP $\rightarrow$ FTPS (FTP Secure)
  - etc.

Transport-layer Security – SSL/TLS II.

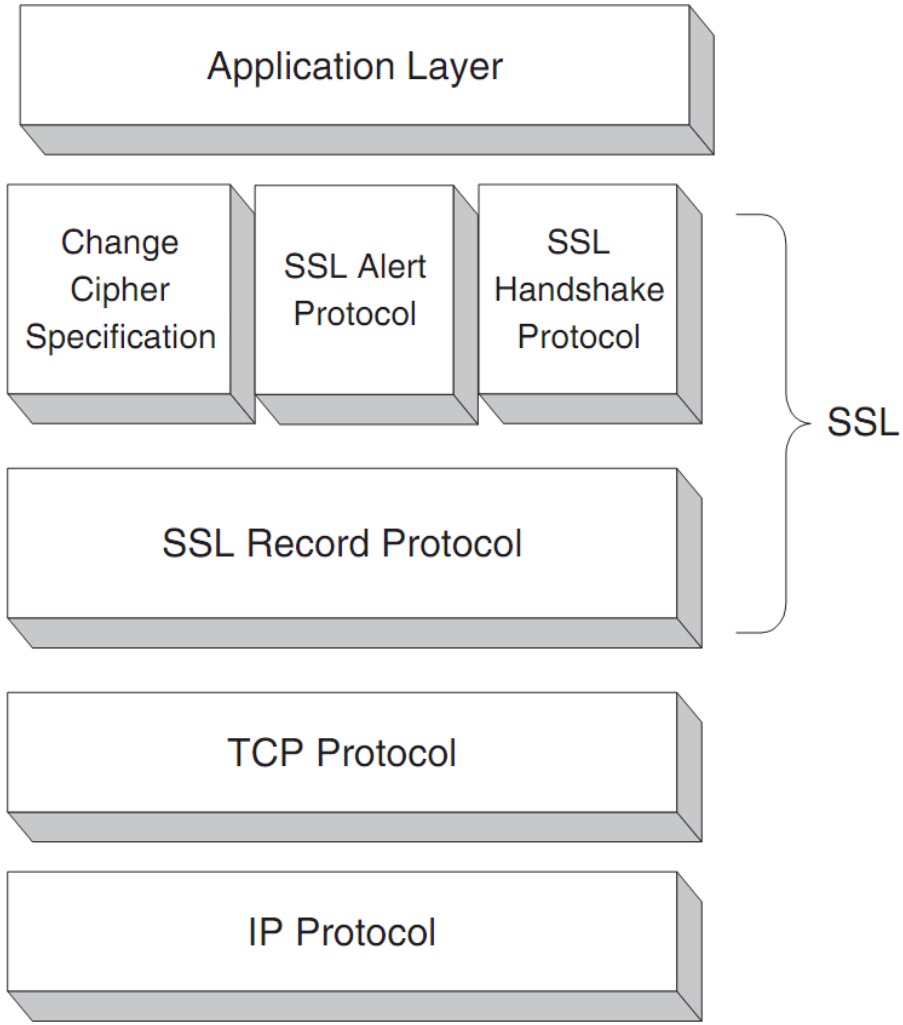


Figure : The SSL protocol stack.

Transport-layer Security – SSL/TLS III.

Before any SSL connection can be established, there must be a handshake performed:

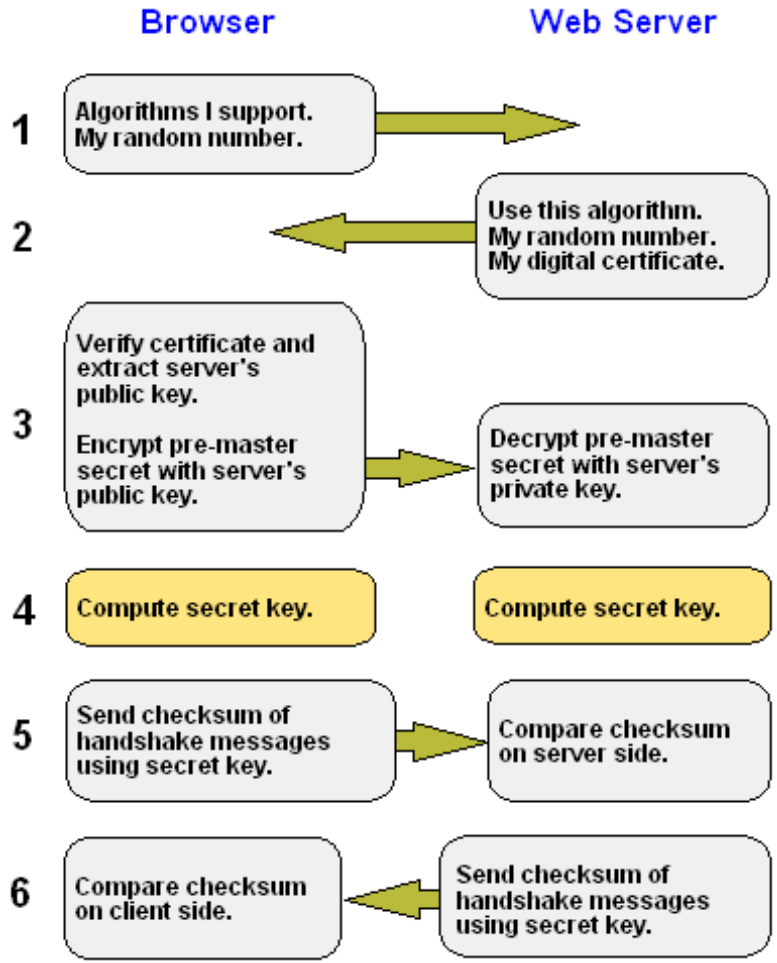


Figure : The SSL handshake (steps 5 and 6 verify the integrity of the handshake; the communication is encrypted using a symmetric (session) key).

Network-layer Security – IPv6 & IPSec (IPv4) I.

IP Security – IPv6 & IPSec (IPv4)

- a set of protocols providing secured communication channels on the network layer
  - *Authentication Header (AH)* protocol – designed for ensuring sender's and receiver's authentication and communication integrity (NOT for communication confidentiality)
  - *Encapsulating Security Payload (ESP)* protocol – designed for providing sender's authentication, communication integrity and confidentiality
  - can be used independently or in combination
- establishes logical channels (called *Security Associations, SAs*)
  - always one-way channels (two channels are needed for duplex communication)
  - identified by:
    - protocol identifier (AH or ESP), source IP address, and channel identification (so-called *Security Parameter Index, SPI*)
- operates in 2 modes:
  - *Transport mode* – IPSec header is inserted between IP header and message body
  - *Tunnel mode* – IPSec header is inserted in front of the original IP header; subsequently, a new IP header is created

Network-layer Security – IPv6 & IPSec (IPv4) II.

IP Security – IPv6 & IPSec (IPv4) cont'd.

- key distribution:
  - *Internet Key Exchange (IKE)* – used to set up an SA (RFC 4306 for v2)
    - uses X.509 certificates for authentication and Diffie–Hellman key exchange to set up a shared session secret
  - *Internet Security Association and Key management Protocol (ISAKMP)* – a framework for authentication and key exchange (RFC 2408)
    - designed to be key exchange independent (IKE or other protocols can be used to provide authenticated keying material)
- *advantages:* secures all the data flows between two communicating parties; it is not necessary to adapt applications (application protocols)
- *drawbacks:* there are no automated mechanisms for cryptographic keys' management
- further information:
  - see IPv6 section in PA159
  - see the animation (ongoing BP Thesis ⇒ will be available in a month): `http://frakira.fi.muni.cz/~jeronimo/vyuka/IPSec`

Network-layer Security – IPv6 & IPSec (IPv4) III.

Transport vs. Tunnel mode

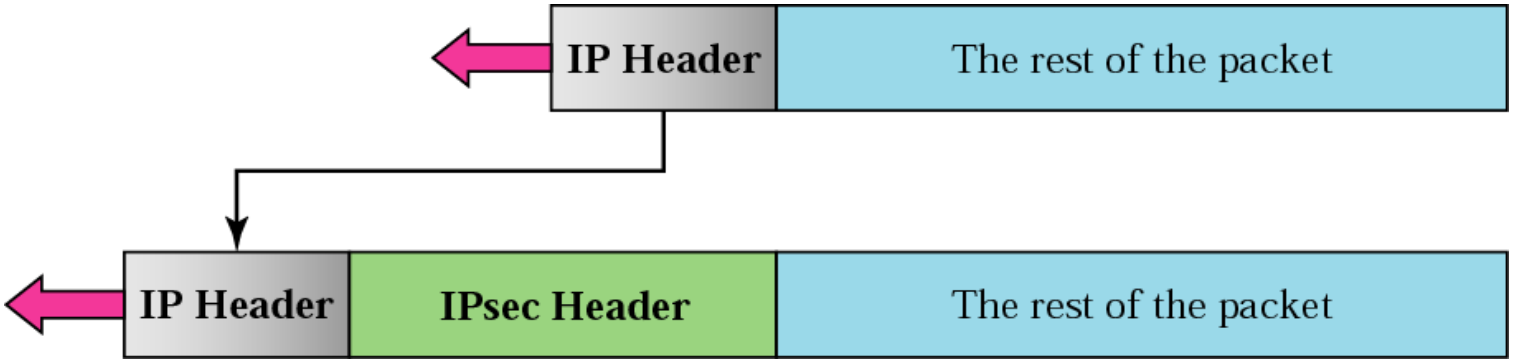


Figure : Transport mode.

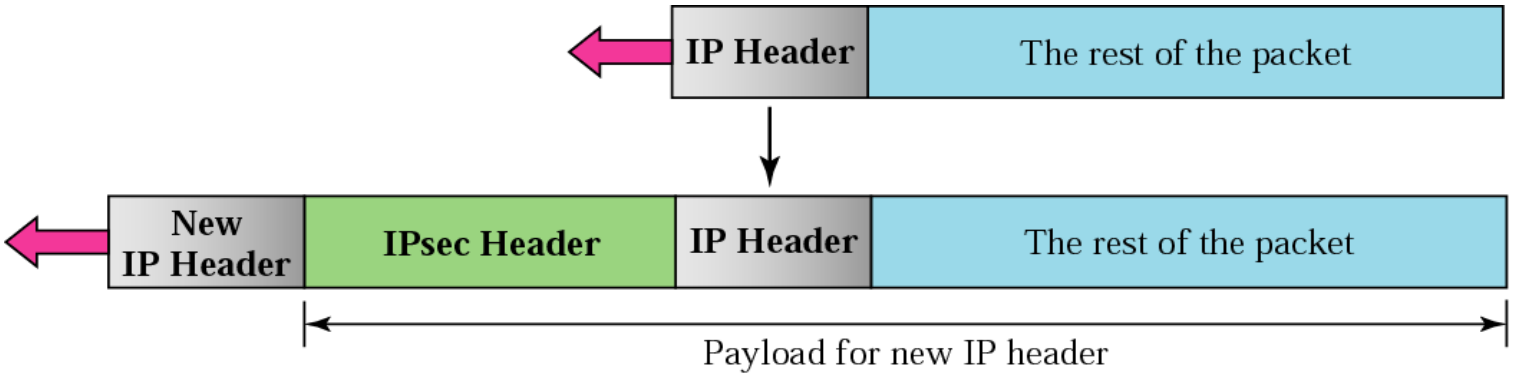


Figure : Tunnel mode.

Network-layer Security – IPv6 & IPSec (IPv4) IV.

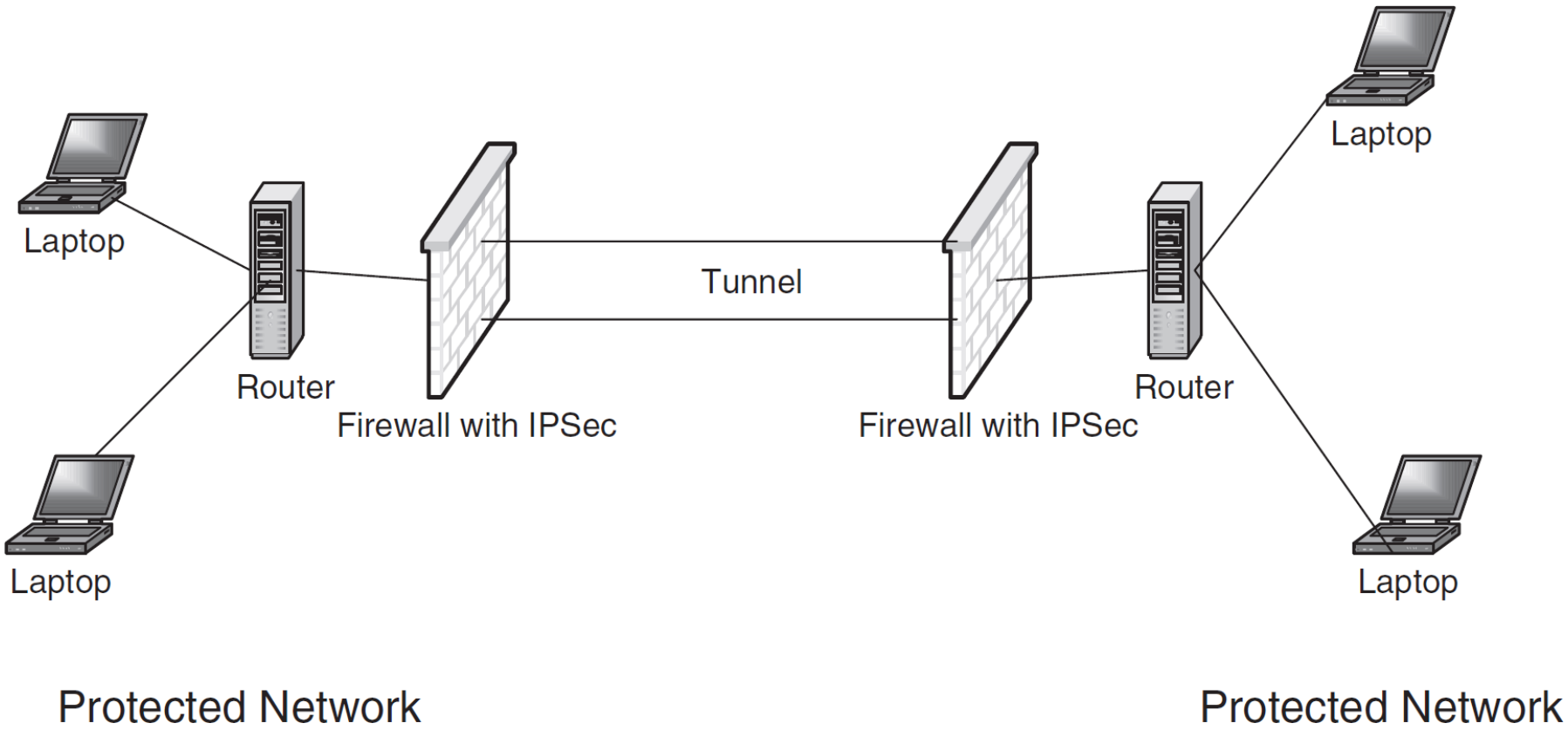
Virtual Private Networks (VPNs) I.

Virtual Private Network (VPN)

- a computer network that uses a public telecommunication infrastructure (such as the Internet) to provide remote offices or individual users with secure access to their organization's network
- using a secure cryptographic method, it encapsulates data transfers between two or more networked devices which are not on the same private network
  - in order to keep the transferred data private
- the VPN technology provides the following activities:
  - *IP encapsulation* – involves enclosing TCP/IP data packets within another packet with an IP-address of either a firewall or a server that acts as a VPN endpoint
    - various tunneling mechanisms: IPSec ESP tunnel, Point-to-Point Tunneling protocol (PPTP), Layer 2 Tunneling Protocol (L2TP), PPP over SSL and PPP over SSH, etc.
  - *Encryption* – used to obfuscate the contents of the encapsulated data
  - *Authentication* – used to determine whether or not the endpoint can participate in the encrypted tunnel
    - may also be used to exchange the secret or public key used for payload encryption

Network-layer Security – IPv6 & IPSec (IPv4) IV.

Virtual Private Networks (VPNs) II.



Link-layer Security

Point-to-Point Protocol (PPP)

Point-to-Point Protocol (PPP)

- a data link protocol commonly used in establishing a direct connection between two networking nodes
  - limited to a single data link
- provides connection authentication, transmission encryption (confidentiality), and compression
  - several authentication protocols can be used (PAP, CHAP, EAP, etc.)
  - transmission encryption – DES or 3DES
- most commonly used forms:
  - *Point-to-Point Protocol over ATM (PPPoA)*
  - *Point-to-Point Protocol over Ethernet (PPPoE)*
  - both serve to establish a DSL line connection with an ISP

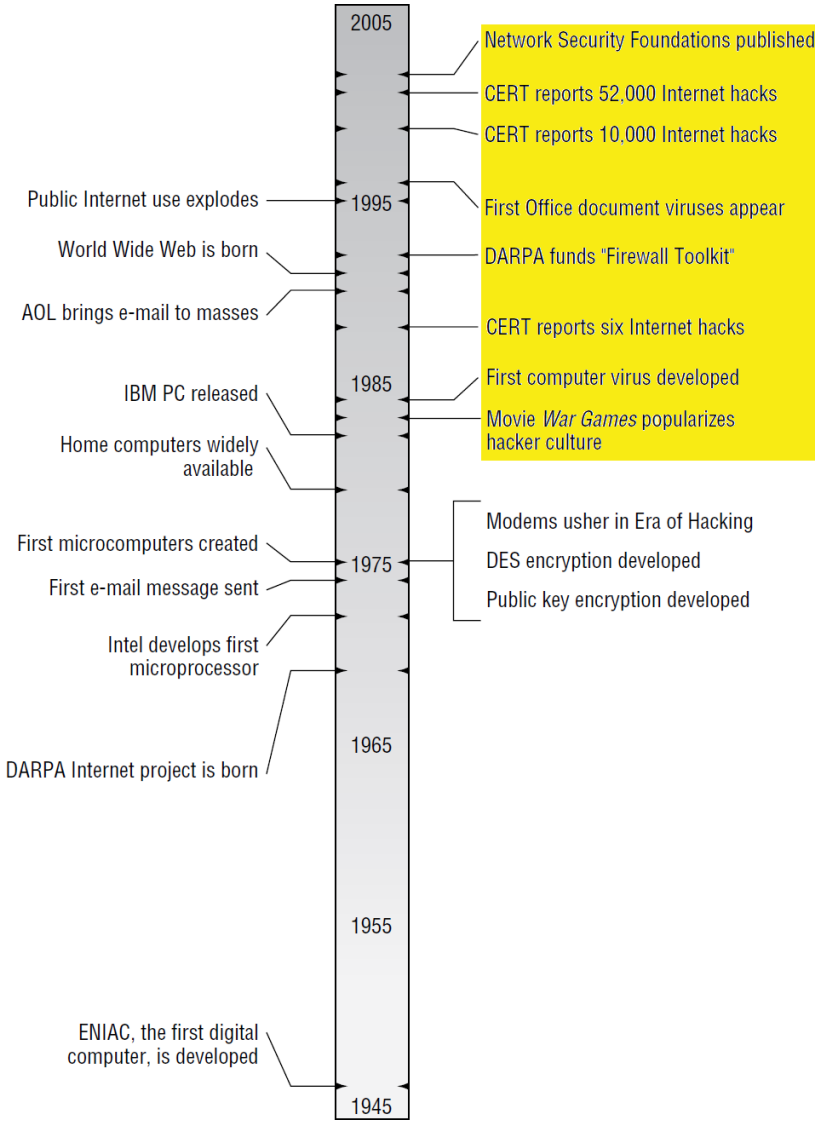
Lecture overview

Network&Data Protection – Maintaining the Availability I.

- all the previous security mechanisms cannot prevent Eve from sending a harmful message to a system
- **network & data protection** – a complicated problem without a simple solution
  - authentication and encryption mechanisms are just technical fundamentals
  - *organizational* and *social* aspects are needed
    - humans (users) are usually the weakest elements $\Rightarrow$ *social engineering*
- possible threats:
  - accessing private data
  - modification of private or public data
    - including their removal (e.g., web pages)
  - undesired interferences to the network traffic/operation
    - including services' denial (so-called *Denial of Service* attacks)

Network&Data Protection – Maintaining the Availability I.

A bit of History



Network&Data Protection – Maintaining the Availability II.

Attack/Intrusion

a deliberate unauthorized attempt, successful or not, to break into, access, manipulate, or misuse some valuable property and where the misuse may result into or render the property unreliable or unusable

The attack/intrusion process consists of few stages:

- *Reconnaissance* – the process of gathering information about the target system and the details of its workings and weak points
  - a scanning program sends network traffic to all/selected computers and expects receiving return traffic that will indicate whether those computers have known vulnerabilities
- *Physical Intrusion* to acquire/modify private information, and/or
- *Denial of Service* to crash a service (or a machine), overload network links or machines' CPU, fill up a disk, etc.

Network&Data Protection – Maintaining the Availability III.

The protection from exploits has to be implemented on:

- *physical level*
  - access protection
- *software level*
  - authentication
  - encryption
  - traffic monitoring and instant reaction to incidents
- *juridical level*
  - a definition of *desired* and *undesired* behaviour

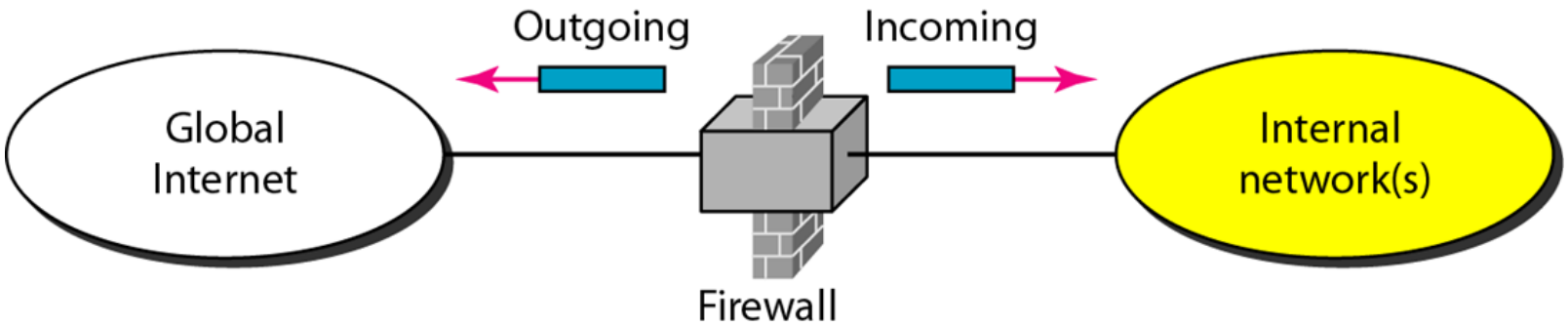
The basic security mechanisms for network protection:

- *Firewalls*
- *Intrusion Detection Systems*
- *Antiviruses*

Firewalls I.

Firewall

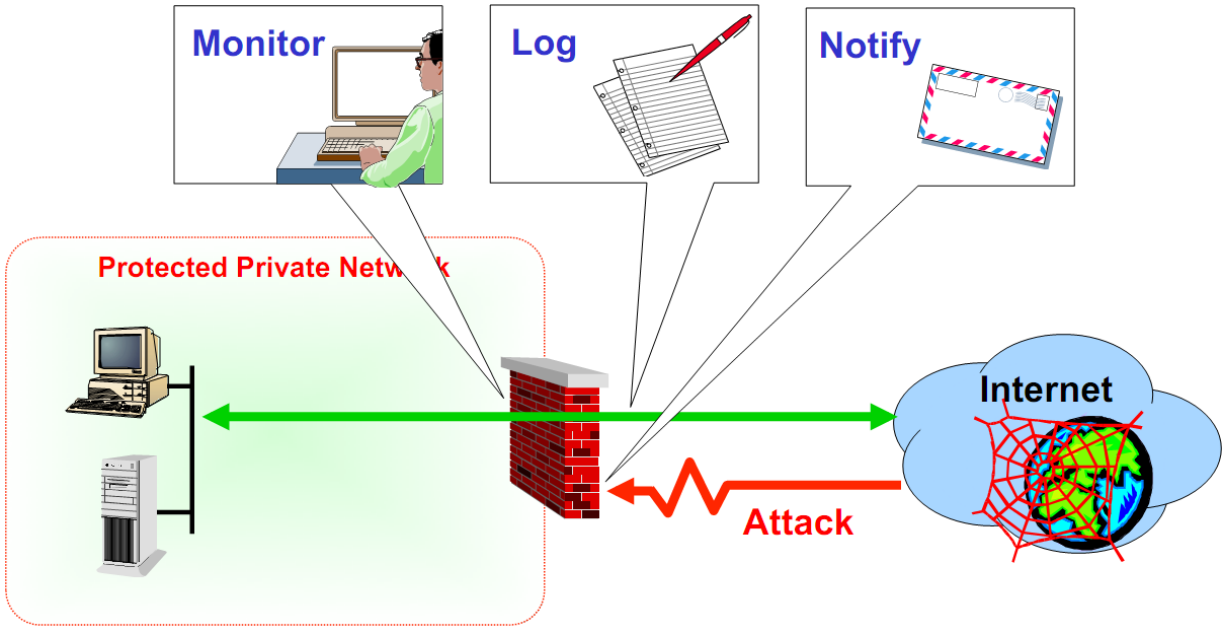
- a primary mean of securing a private network against penetration
- *firewall* = a hardware, software, or a combination of both that monitors and filters traffic packets that attempt to either enter or leave a protected private network
  - i.e., interconnects networks with *different trust*
  - creates a *perimeter defense* designed to protect information resources of an organization



Firewalls II.

What a firewall can do...

- implement security policies (≈ network access protection) at a single point
  - only authorized traffic is allowed to access the network/services
  - (or leave the network)
- monitor security-related events (audit, log)
  - alarms for abnormal behavior can be implemented



Firewalls III.

and what a firewall **cannot** *do. . .*

- protect against attacks that bypass the firewall
  - e.g., dial-out from internal host to an ISP
- protect against internal threats
  - disgruntled employee
  - an insider cooperates with an external attacker
- protect against the transfer of virus-infected programs or files
  - not including application gateways, which may be able to do so

Firewalls IV.

Firewalls' Classification

Basic firewalls' classification:

- *Packet filtering firewalls* – operate on network layer
- *Circuit level gateways* – operate on session/transport layer
- *Application gateways (or proxy servers)* – operate on application layer
- *Dynamic packet filtering firewalls (or Session filtering firewalls)*
  - a combination of the three above

- these technologies operate at different levels of detail, providing varying degrees of network access protection

Firewalls V.

Packet Filtering firewalls

Packet Filtering firewalls

- decisions are made on per-packet basis
- no state information is maintained
- applies packet filters based on access rules defined by the following parameters:
  - source address
  - destination address
  - application or protocol/next header (TCP, UDP, etc.)
  - source port number
  - destination port number
- *advantages:*
  - simple, low cost, fast, transparent to users
- *disadvantages:*
  - cannot prevent attacks that employ application-specific vulnerabilities or functions
    - because they do not examine upper-layer data

Firewalls V.

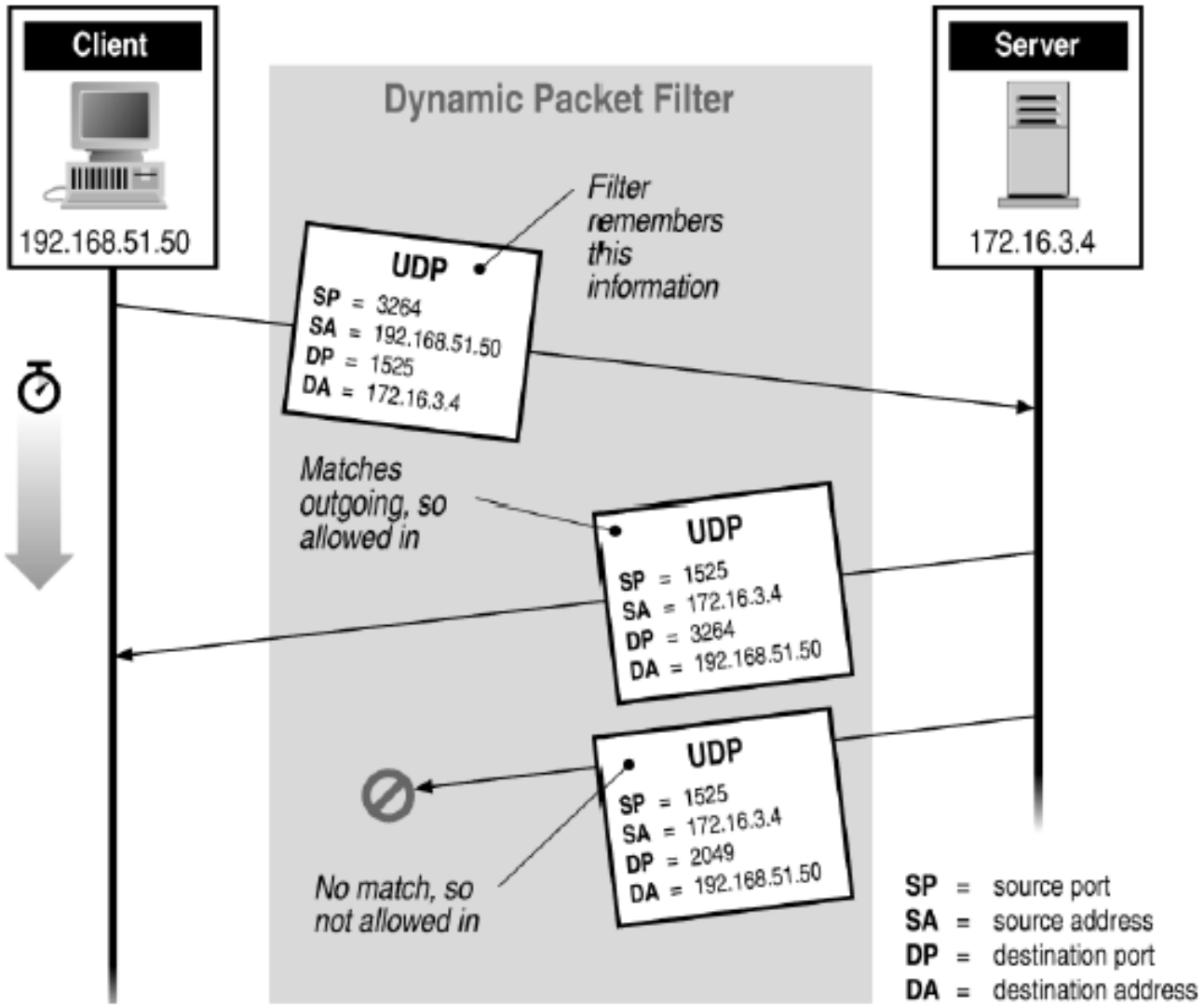
Dynamic Packet Filtering (Session Filtering) firewalls I.

Dynamic Packet Filtering (Session Filtering) firewalls

- traditional packet filters do not examine higher layer context
  - i.e., matching the incoming (returned) packets with an outgoing flow
- dynamic packet filtering examines and maintains a *context of each packet*
  - they keep track of client-server connection
  - $\Rightarrow$ a connection table (so-called *State table*) is maintained
- packet decision is made in the context of a connection
  - if a packet establishes a new connection, it is checked against a policy
  - if a packet is a part of an existing connection, it is matched up in the state table
- *advantages:*
  - transparent to users
  - low CPU overheads
    - for all the packets (except the first one) belonging to the same connection, no policy database lookups are performed

Firewalls V.

Dynamic Packet Filtering (Session Filtering) firewalls II.



Firewalls V.

Circuit Level Gateways I.

Circuit Level Gateways

- work at the session layer of the ISO/OSI model (TCP layer of the TCP/IP model)
- monitor TCP handshaking between packets to determine whether a requested session is legitimate
- do not permit an end-to-end TCP connection
  - rather, the gateway sets up two TCP connections
    - one between itself and a TCP user on an inner host, and
    - one between itself and a TCP user on an outside host
  - once the two connections are established, the gateway typically relays TCP segments from one to the other without examining the contents
- *usage example:*
  - hiding information about internal/protected networks
    - non-requested data from outside the firewall is not allowed in – all ports are closed until the firewall opens them
  - e.g., requests from protected network pass through the gateway, which intercept and record them
    - once the answer reaches the gateway, it is compared to the original request to see if the IP address and the port match up

Firewalls V.

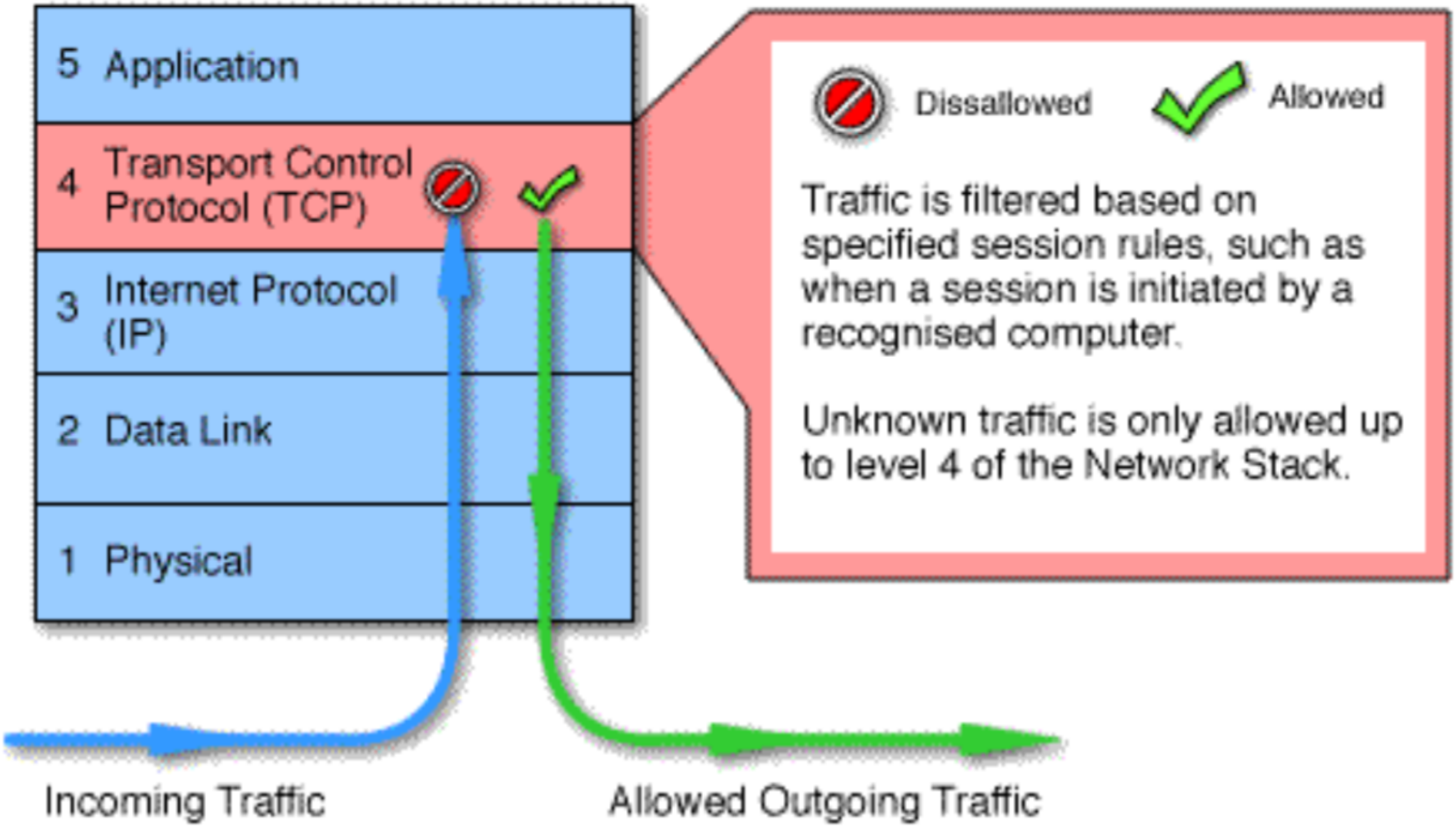
Circuit Level Gateways II.

Circuit Level Gateways cont'd.

- a combination of circuit-level gateway with Internet sharing = *Network Address Translation (NAT)*
  - originally a reaction to IPv4 address exhaustion
  - but the NAT is **not a firewall** on itself!

Firewalls V.

Circuit Level Gateways III.

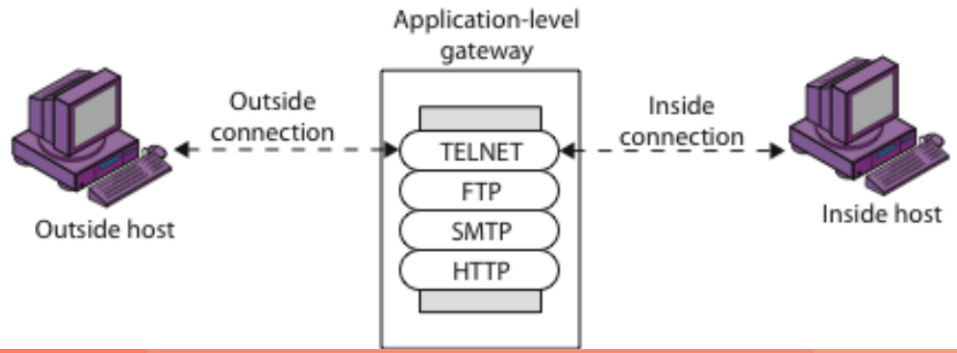


Firewalls V.

Application Gateways I.

Application Gateways

- similar to circuit-level gateways except that they are application specific
  - i.e., tailored to a specific application program
- every connection between two networks is made via an application program called a *proxy*
  - proxies are application/protocol specific
    - they investigate the content of the passing flows
  - connection state is maintained and updated
- only protocols that have specific proxies configured are allowed through the firewall (all other traffic is rejected)
  - e.g., a gateway that is configured to be a web proxy will not allow any FTP, telnet or other traffic through



Firewalls V.

Application Gateways II.

Application Gateways cont'd.

- *usage example:*
  - HTTP proxies (URL blocking), SMTP proxies (virus scanning, spam identification), DNS proxies, etc.
- *advantages:*
  - more secure than packet filtering firewalls
    - can investigate the content of the flows
    - can protect against protocol errors
  - it is easy to log and audit all incoming/outgoing traffic
- *disadvantages:*
  - very CPU intensive $\Rightarrow$ lower performance
    - require high performance host computer
  - more vulnerable to DoS attacks (see later)
  - problematic for encrypted data (including IPSec)

Firewalls VI.

Concluding Remarks

- firewalls cannot serve as a panacea to network security threats
  - many security attacks led from within the internal (not protected) network
  - increase the risk of rapid infection spread
    - false safety feeling
    - the internal network is not usually sufficiently protected
    - users become lazy
  - usually cause a slowdown on high-speed networks
- their functionality is very useful for interrupting the ongoing attack

- *application proxies*:
  - risk of faults (e.g., false virus detection)
  - the importance of privacy protection

Intrusion Detection Systems (IDS) I.

- network perimeter security technologies (i.e., firewalls) and anti-malicious software technologies (i.e., antiviruses, see later) cannot stop attackers if they obtain authenticated access to target computers and log on to them as if they are legitimate users
  - (i.e., they had obtained login information of a legitimate user)
- it is desirable (and often necessary) to detect intrusion activities by monitoring ingress packets that have passed through firewalls and analyze how users use their computers
  - $\Rightarrow$ *Intrusion Detection Systems (IDS)* and *Honeypots*

Intrusion Detection Systems (IDS)

- a system used to *detect* unauthorized intrusions into computer systems and networks
- based on an assumption that the behavior of the intruder differs from that of a legitimate user
  - they collect information from a variety of system and network sources and look for *attack signatures*
    - = specific patterns that usually indicate malicious or suspicious intent

Intrusion Detection Systems (IDS) II.

Classification

IDS classification:

- different ways of classifying an IDS
- IDSs based on *detection model*:
  - anomaly detection
  - signature based misuse (misuse detection)
  - hybrid
- IDSs based on *scope*:
  - host-based
  - multihost-based
  - network-based
- IDSs based on *operation*:
  - offline – inspect system logs at set intervals
  - online – monitor the system continuously

Intrusion Detection Systems (IDS) III.

Anomaly Detection

Anomaly Detection IDSs:

- model the normal usage of the network as a noise characterization
- anything distinct from the noise (atypical behavior) is assumed to be an intrusion activity
  - e.g., flooding a host with lots of packet
- the *primary strength* is its ability to recognize novel attacks
- *drawbacks:*
  - assume, that intrusions will be accompanied by manifestations that are sufficiently unusual
  - generate many false alarms
    - $\Rightarrow$ compromise the effectiveness of the IDS

Intrusion Detection Systems (IDS) III.

Misuse Detection

Misuse Detection IDSs:

- recognize known attacks based on *signatures*
  - a set of attack signatures has to be defined (and updated)
- if an action matching a signature is detected, it is considered to be an attack
- *drawbacks:*
  - unable to detect novel attacks
  - suffer from false alarms
  - have to be programmed again for every new pattern to be detected

Intrusion Detection Systems (IDS) III.

Host-based & Multihost-based IDSs I.

Host-based IDSs:

- host operating system (or an application) logs in the audit information
  - e.g., use of identification and authentication mechanisms (logins, etc.), file opens and program executions, admin activities, etc.
- the audit is then analyzed to detect trails of intrusion

Multihost-based IDSs:

- audit information is analyzed from multiple hosts
- analysis architecture:
  - *centralized* – all data shipped to a central location for analysis
  - *hierarchical* – data is analyzed as it passed up through the layers
  - *distributed* – distributed analysis of the data

Intrusion Detection Systems (IDS) III.

Host-based & Multihost-based IDSs II.

Host-based & Multihost-based IDSs: cont'd.

- *advantages:*
  - can detect data that are encrypted during transmissions
  - do not need special hardware devices
  - check system logs and so can analyze system behavior more accurately
- *disadvantages:*
  - are installed in the target systems, and so they require extra system managing
  - may consume extra computing resources on the systems
  - might become affected by attacks that affect the host computers/servers they are installed in

Intrusion Detection Systems (IDS) III.

Network-based IDSs

Network-based IDSs:

- look for attack signatures in network traffic via a promiscuous interface
- a filter is usually applied to determine which traffic will be discarded or passed on to an attack recognition module
  - to filter out known un-malicious traffic
- *advantages:*
  - low cost – usually small number of selected points to monitor the entire network
  - no interference – packets are monitored passively (it does not interfere normal network traffic)
  - intrusion resistent – usually a small system, which can be easily made to resist an intrusion
- *disadvantages:*
  - may not be able to analyze encrypted packets
  - may not be able to handle large volume of network traffics in time
  - some intrusion activities (e.g., fragmentation-attack) are hard to identify
  - hard to determine, whether the intrusion activity has been successfully carried out

Intrusion Detection Systems (IDS) IV.

Concluding Remarks

- *Intrusion Detection Systems* are only one piece of the whole security puzzle
- they must be supplemented by other security and protection mechanisms
- they are a very important part of security architecture but do not solve all the potencial problems

Honeypots

Honeypot

- a system designed to teach how intruders probe for and exploit a system
  - by learning their tools and methods, one can better protect her network and systems
- usually used for:
  - diverting an attacker from accessing critical systems
  - collecting information about the attacker's activity
  - encouraging the attacker to stay on the system long enough for administrators to respond
- based on virtual or real systems

Honeypots

"Tracking Their Moves" I.

"Tracking Their Moves"

- do not log information on the honeypot itself
  - the fewer modification you make to the honeypot, the better
    - the more changes you make, the better the chance an intruder will discover something is up
  - you can easily lose the information
- track in layers, do not depend on a single source of information
  - first layer of tracking is the firewall logs
  - second layer is the system logs
  - third layer of tracking is to use a sniffer
    - it picks up all keystrokes and screen captures
  - run, e.g., the *Tripwire* on the honeypot
    - to discover what binaries have been altered on a compromised system

Honeypots

"Tracking Their Moves" II.

"Tracking Their Moves" cont'd.

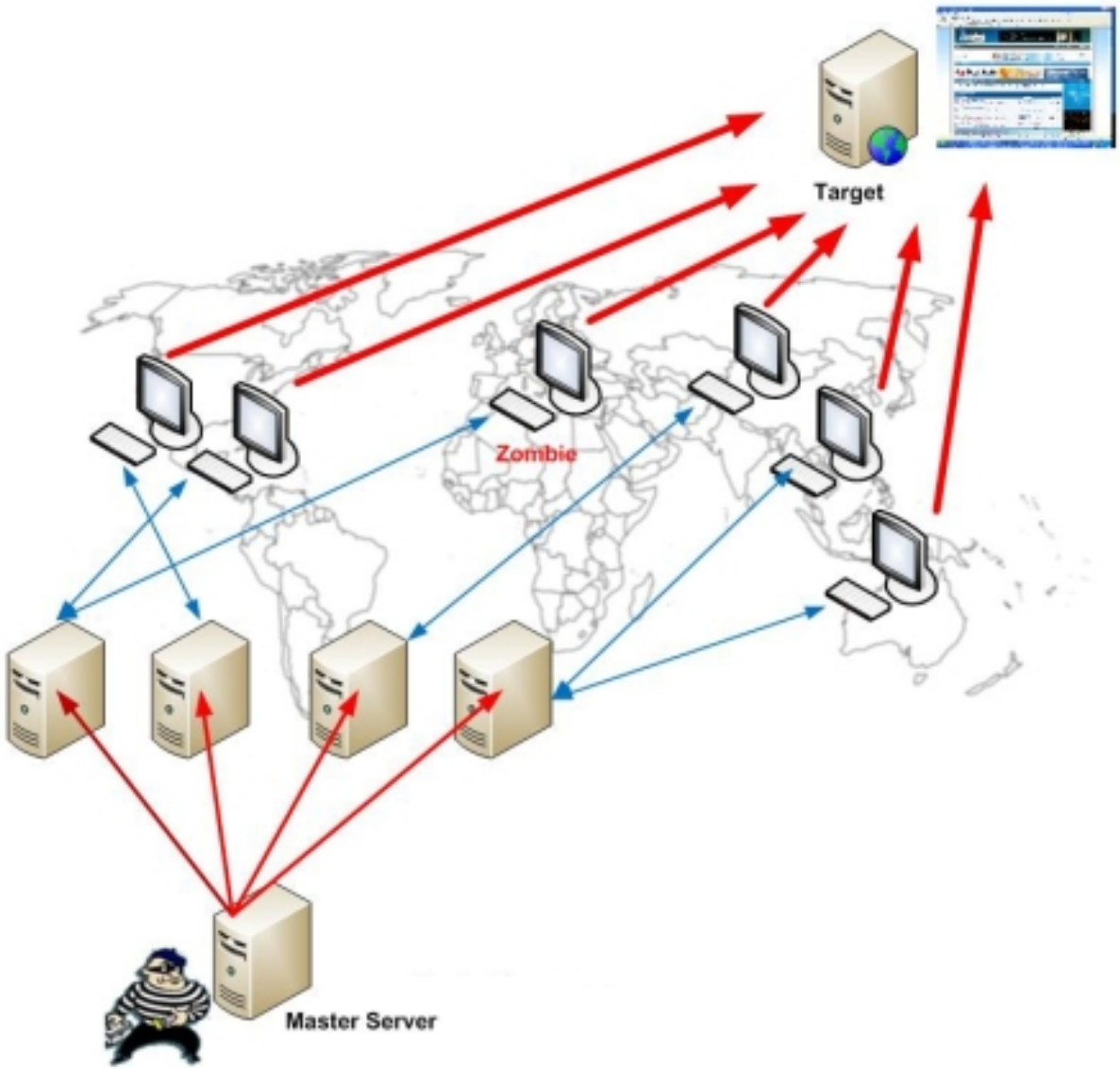
- **do not mindlessly reinstall the compromised system!!!**
  - otherwise you'll lose the information about the security hole the intruder had exploited
- if possible, do not reboot the compromised system
  - running malicious programs may not automatically start after the reboot
- if possible, do not disconnect the compromised system from the network
  - the system, once being disconnected from the network, may close all the active network connections
  - better way is to cut off the system on the closest firewall
    - or disconnect it behind first network element (switch, router, etc.)

Denial of Service (DoS) & Distributed DoS (DDoS) I.

Denial of Service (DoS) attack:

- an attack that causes a loss of a service to users
  - typically the loss of network connectivity and services by consuming the bandwidth of the victim's network or overloading the computational resources of the victim's system
  - the flood of incoming messages essentially forces the target system to shut down (thus denying the service)
- attack lead *from a single source*
  - not too dangerous – can be stopped in the closest active network element
- attack lead *from many sources* = **Distributed Denial of Service (DDoS) attack**
  - a synchronized attack from many sources (e.g., Botnets)
  - hard to discover – all the requests appear to be sent by legitimate users
  - much more dangerous
    - stopping requires a cooperation in the network
    - stopping in *active networks* – consecutive cut of flows against their direction

Denial of Service (DoS) & Distributed DoS (DDoS) II.



Virusses & Worms I.

Computer virus

a self-propagating computer program designed to alter or destroy a computer system resource

Worm

a self-replicating program, similar to a computer virus – a virus attaches itself to, and becomes part of, another executable program; however, a worm is self-contained and does not need to be part of another program to propagate itself (through security holes of a networked system)

Virus/Worm attacks:

- some sort of trigger will activate the attack phase, and the virus will then "do something"
  - *trigger* = a specific date, or the number of times the virus has been replicated, or anything similar
  - *"do something"* = anything from printing a silly message on the screen to erasing all of your data

Virusses & Worms II.

Protection from virus/worm attacks:

- run a secure operating system with access rights
  - security features keep viruses away
- run virus protection software
  - in end-nodes and/or application gateways
- avoid programs from unknown sources
- educating the users is a must
  - = *computer hygiene*

Network & Data Protection – Concluding Remarks

- attacks/intrusions – an effort to acquire an access to a network or its services
  - reasons range from boyish fun ("testing neighbor's pears") to military/terroristic actions
- *usage:*
  - private data access/modification
  - source platform for subsequent activity (e.g., DDoS)
- *protection:*
  - access control on all possible levels
  - regular maintanance (patches, upgrades, etc.)
  - network traffic monitoring
  - monitoring of processes running on the servers/end-nodes
  - *users' education*

Lecture overview

Recapitulation – Network Security

- AAA mechanisms + secured communication mechanisms + data/services protection/availability
- features of a secured communication network:
  - Confidentiality
  - Integrity
  - Non-repudiation
- symmetric (shared key) and asymmetric (key pair – private and public) cryptography
- secured communication mechanisms can be provided on various ISO/OSI layers (application, transport, network, data link)
- *further information:*
  - PV017: Information Technology Security (doc. Staudek)
  - PV079: Applied Cryptography (prof. Matyáš et al.)
  - PV080: Protection of Data and Information Privacy (prof. Matyáš)
  - PV157: Authentication and Access Control (prof. Matyáš and dr. Říha)
  - PV210: Security analysis of network traffic (RNDr. Vykopal)
  - etc.