

Lecture 12

Network Security

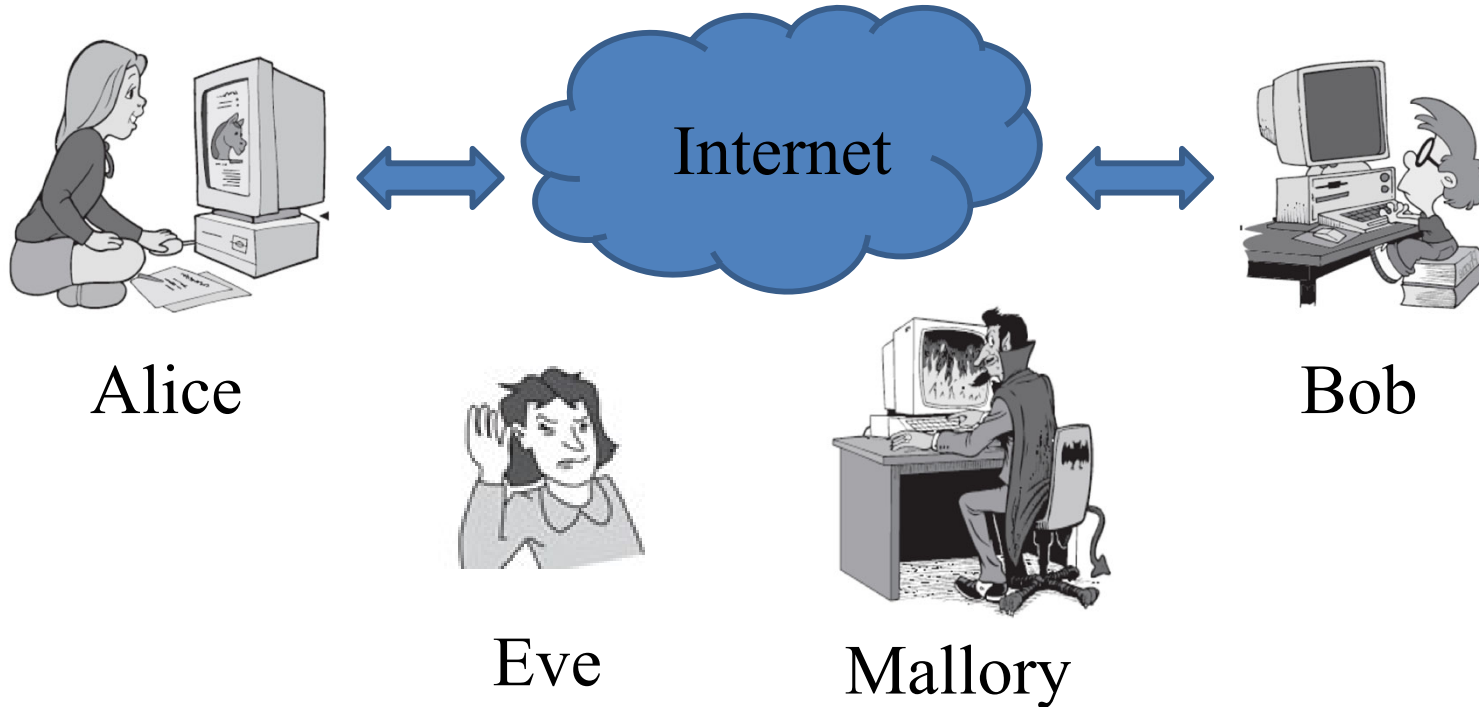
ELEC 3506/9506
Communication Networks

Dr Wibowo Hardjawana
School of Electrical and Information
Engineering

Topics of the Day

- Network Security Goals
- Security Threats and Attacks
- Security Services
- Security Mechanisms
 - Confidentiality with
 - Symmetric Key
 - Asymmetric Key
 - Message Integrity
 - Message Authentication
 - Digital Signatures
 - Entity Authentication

Communication over Networks



- **Alice**: legitimate sender
- **Bob**: legitimate receiver
- **Eve**: eavesdropper, passive attacker.
- **Mallory**: malicious active attacker

Alice, Bob

Who might Bob and Alice be?

- ... well, *real-life* Bobs and Alices!
- Web browser/server for electronic transactions (e.g., on-line purchases)
- on-line banking client/server
- DNS servers
- BGP routers exchanging routing table updates
- other examples?

Eve and Malory

Q: What can a “bad guy” do?

A: A lot! (previous slide)

- **eavesdrop**: intercept messages - Eve
- actively **insert** messages into connection - Malory
- **impersonation**: can fake (spoof) source address in packet (or any field in packet) - Malory
- **hijacking**: “take over” ongoing connection by removing sender or receiver, inserting himself in place - Malory
- **denial of service**: prevent service from being used by others (e.g., by overloading resources) - Malory

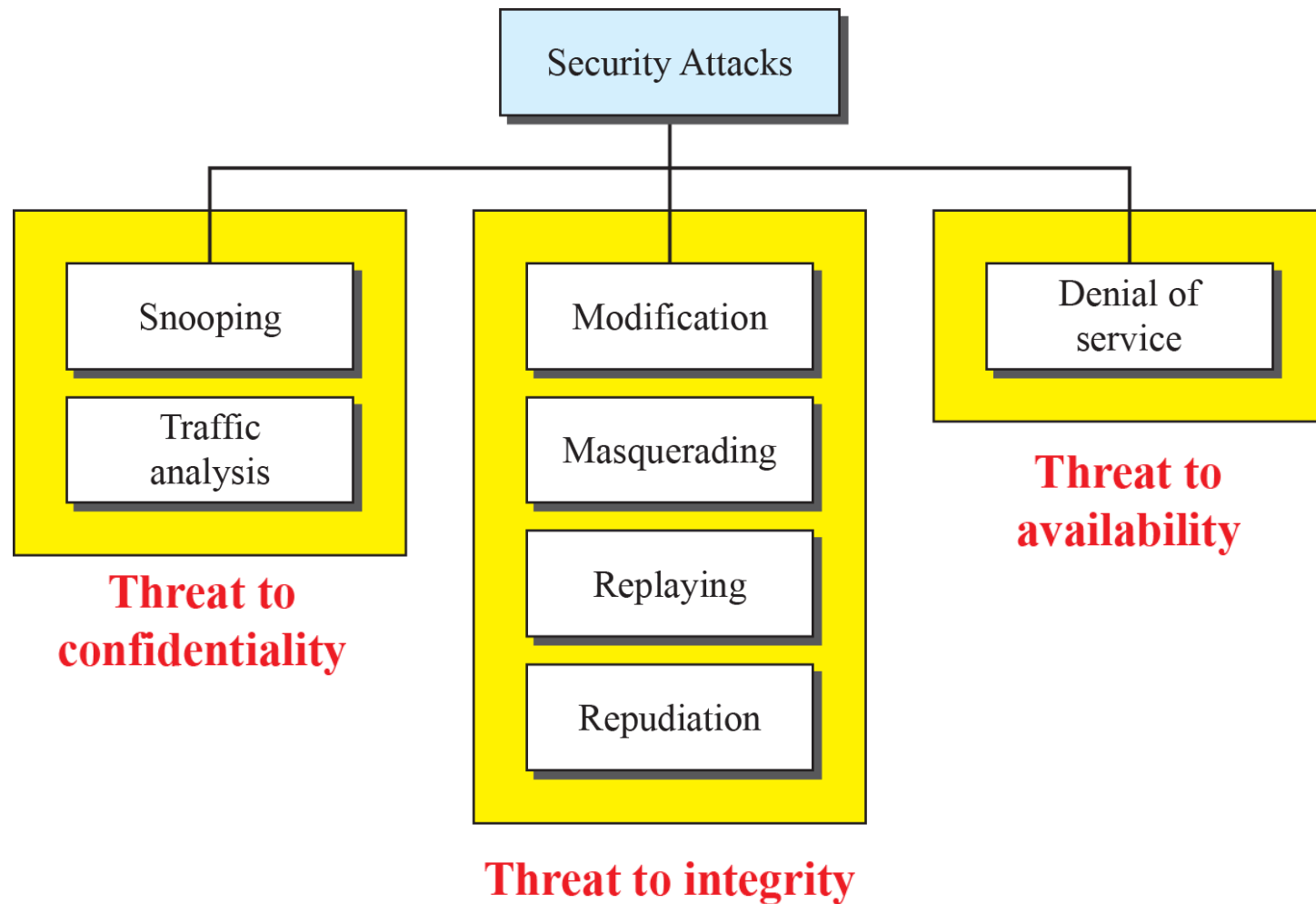
Security Goals

- **Confidentiality**
 - Information is hidden from unauthorized access
 - Applies to both storage and transmission of information
- **Integrity**
 - Information is protected from unauthorized change
 - Changes are only made by authorized entities and through authorized mechanisms
- **Availability**
 - Information is made available to authorized entity when it is needed
 - Information is useless if it is not available

Security Threats/Attacks

- Security attacks are of two types:
 - **Passive Attack:** Unauthorized party eavesdrops communication between two parties but does not tamper with the messages being communicated.
 - **Active Attack:** Attacker transmits tampered data to one or both of the parties or block the data stream in one or both directions.

Attacks to Security Goals



Attacks Threatening Confidentiality

■ Snooping

- Unauthorized access to or interception of data
- To prevent snooping, the data can be made non-intelligible to the interceptor by using encipherment techniques

■ Traffic analysis

- Intercepting and examining messages in order to deduce information from patterns in communication, e.g., pattern-of-life analysis
- Can be performed even when the messages are encrypted

Attacks Threatening Integrity

- **Modification**
 - The attacker modifies the information to make it beneficial to herself
- **Masquerading/spoofing**
 - The attacker impersonates somebody else
- **Replaying**
 - The attacker obtains a copy of a message sent by a user and later tries to replay it
- **Repudiation of Action**
 - The sender (receiver) of the message later denies that he has sent (received) the message
 - Threats against accountability

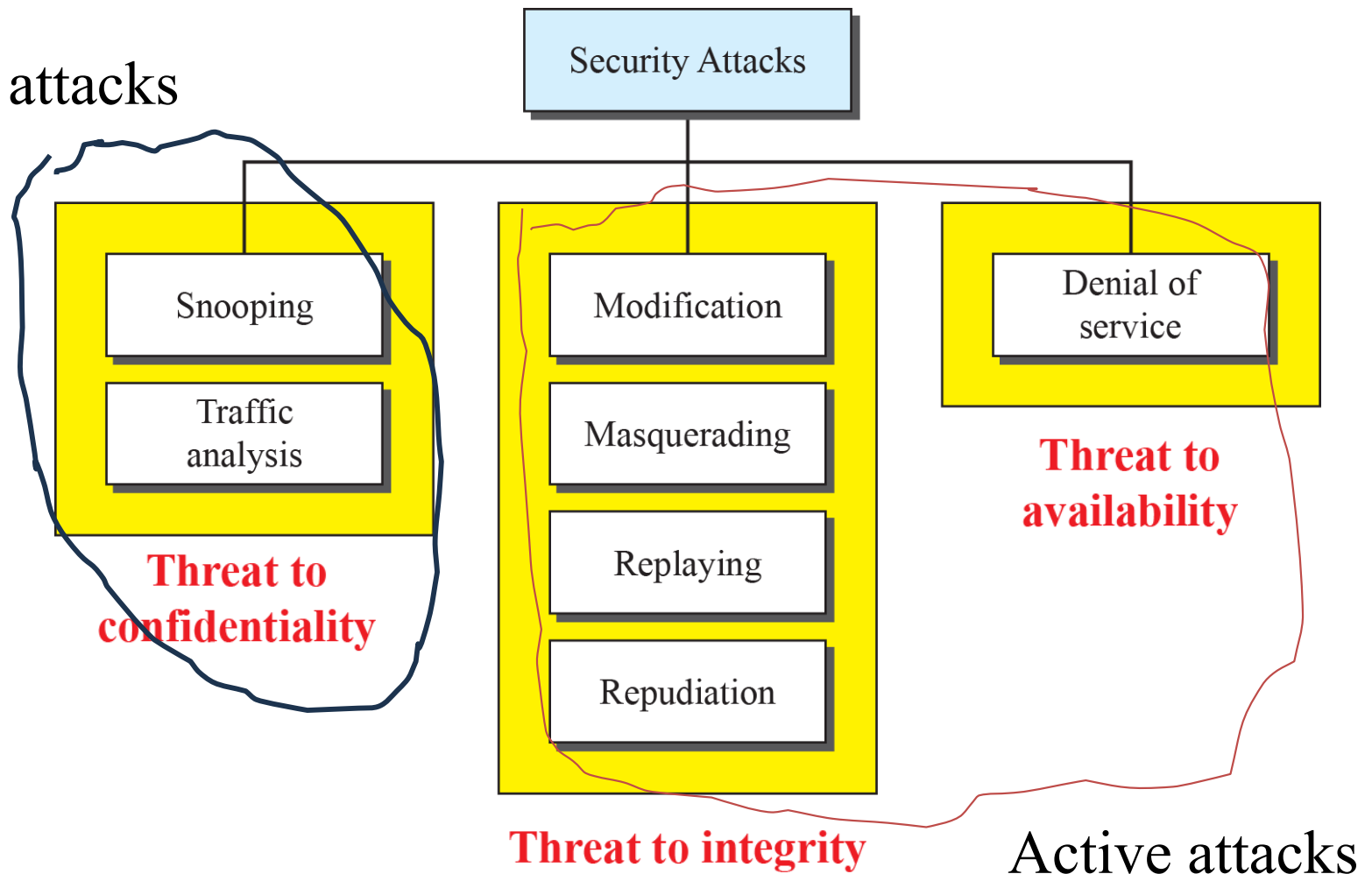
Attacks Threatening Availability

Denial of Service (DoS)

- Attacker acts to deny resources/services to entities which are authorized to use them.
- Attacker might send many bogus request to overload a server.
- Attacker might intercept and delete a server's response to a client, making the client believe that the server is not responding

Attacks to Security Goals

Passive attacks



Security Services

- Confidentiality Service
- Integrity Service
- Authentication Service
- Nonrepudiation Service

Security Services (cont'd)

Confidentiality Service:

- **Data Confidentiality:** Confidentiality of data maintained between the sender and receiver.
- **Traffic Flow Confidentiality:** Confidentiality of the identities of the source and destination of the data traffic.
- Mechanism implemented: **Encryption**

Integrity Service:

- Ensures that data **modification be capable to authorized parties only.**
- **Data Integrity:** Ensures the data received is NOT modified
- **Data Sequence Integrity:** Sequence of data blocks is not altered, repeated or missing.
- Mechanism implemented: **Hash Functions, Cryptographic Checksums**

Security Services (cont'd)

Authentication Service:

- Ensures the data is genuine, unaltered, complete and not an unlawful replay.
- **Data origin Authentication:** used to check the message origination from the claimed source.
- **Peer-entity-authentication:** Provides an entity the confidence of the authenticity of a peer entity.
- Mechanism Implemented: **Private Key and Public Key based techniques, challenge response mechanisms, timestamps.**

Security Services (cont'd)

Nonrepudiation service:

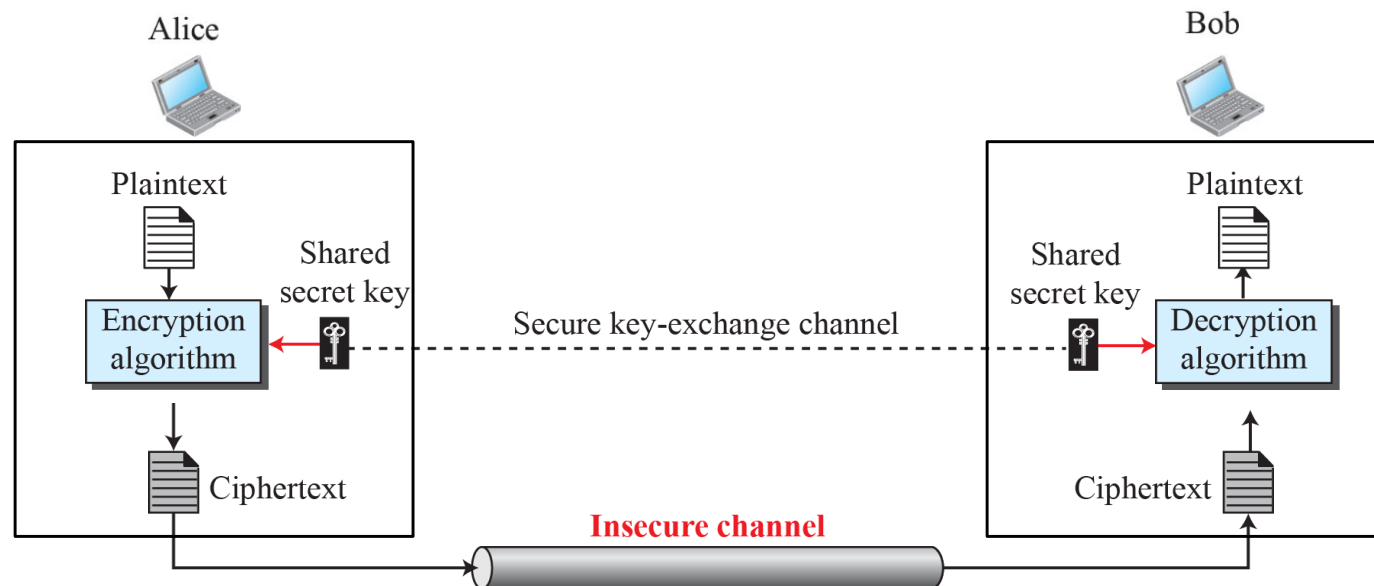
- A sender must not be able to deny sending a message that he or she, in fact, did send.
- Mechanism Implemented: **Digital Signatures**

Security Mechanisms

- Different mechanisms cater to different services.
- We shall only focus on the following:
 - Confidentiality
 - Symmetric/Secret Key Encryption
 - Asymmetric/Public Key Encryption
 - Integrity
 - Hash Functions and Message Digests
 - Authentication
 - Keyed Hash Functions
 - Digital Signatures

Confidentiality with Symmetric-Key Cryptography

- Both encryption and decryption is done using the same key
 - Sender and receiver share the same secret key.
 - Encryption process consists of the algorithm and the key.
 - The key is a value that is independent of the algorithm.
 - The algorithm produces a different output depending on the key being used at that time.



Prerequisite: modular arithmetic

- $x \bmod n$ = remainder of x when divide by n

- facts:

$$[(a \bmod n) + (b \bmod n)] \bmod n = (a+b) \bmod n$$

$$[(a \bmod n) - (b \bmod n)] \bmod n = (a-b) \bmod n$$

$$[(a \bmod n) * (b \bmod n)] \bmod n = (a*b) \bmod n$$

- thus

$$(a \bmod n)^d \bmod n = a^d \bmod n$$

- example: $x=14$, $n=10$, $d=2$:

$$(x \bmod n)^d \bmod n = 4^2 \bmod 10 = 6$$

$$x^d = 14^2 = 196 \quad x^d \bmod 10 = 6$$

Example: Shift Ciphers

- Plaintext consists of lowercase letters and ciphertext consists of uppercase letters. Assign numerical value to each letter
- The secret key is an integer in modulo 26
- Encryption: add the key to the plain text character
- Decryption: subtract the key from the ciphertext

Plaintext →	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Ciphertext →	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Question: Use the shift cipher with key = 15 to encrypt the message “hello”

Recovery of $x \rightarrow (x + 15) \bmod 26 = \text{Ciphertext}$

Plaintext: h → 07	Encryption: $(07 + 15) \bmod 26$	Ciphertext: 22 → W	$x = 7$
Plaintext: e → 04	Encryption: $(04 + 15) \bmod 26$	Ciphertext: 19 → T	$x = 4$
Plaintext: l → 11	Encryption: $(11 + 15) \bmod 26$	Ciphertext: 00 → A	$x = 11$
Plaintext: l → 11	Encryption: $(11 + 15) \bmod 26$	Ciphertext: 00 → A	$x = 11$
Plaintext: o → 14	Encryption: $(14 + 15) \bmod 26$	Ciphertext: 03 → D	$x = 14$

A more sophisticated encryption approach

- n substitution ciphers, $M_1, M_2, \dots, M_n$
 - cycling pattern:
 - e.g., $n=4$: M_1, M_3, M_4, M_3, M_2 ; M_1, M_3, M_4, M_3, M_2 ; ..
 - for each new plaintext symbol, use subsequent substitution pattern in cyclic pattern
 - dog: d from M_1 , o from M_3 , g from M_4
-  **Encryption key:** n substitution ciphers, and cyclic pattern
- key need not be just n-bit pattern

Examples of Symmetric Key Algorithms

- **DES (Data Encryption Standard):** Adopted in 1977 by National Institute of Standards Technology (NIST) as a federal information processing standard
- **IDEA (International Data Encryption Algorithm):** Developed in 1990 by the Swiss Federal Institute of Technology
- **SKIPJACK:** Designed and evaluated by the National Security Agency (NSA) in 1993
- **AES:** Encryption standard established by the NIST in 2001. Supersedes DES. Based on the Rijndael cipher developed by Joan Daemen and Vincent Rijmen.

Symmetric key crypto: DES

DES: Data Encryption Standard

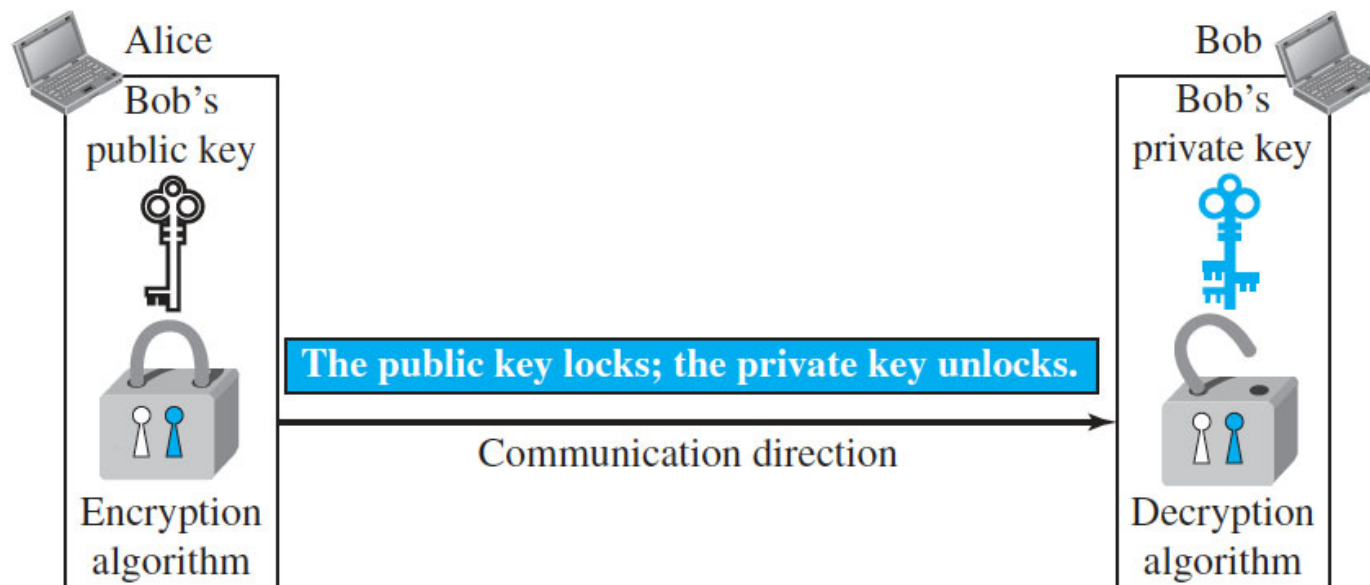
- US encryption standard [NIST 1993]
- 56-bit symmetric key, 64-bit plaintext input
- block cipher with cipher block chaining
- how secure is DES?
 - DES Challenge: 56-bit-key-encrypted phrase decrypted (brute force) in less than a day
 - no known good analytic attack
- making DES more secure:
 - 3DES: encrypt 3 times with 3 different keys

AES: Advanced Encryption Standard

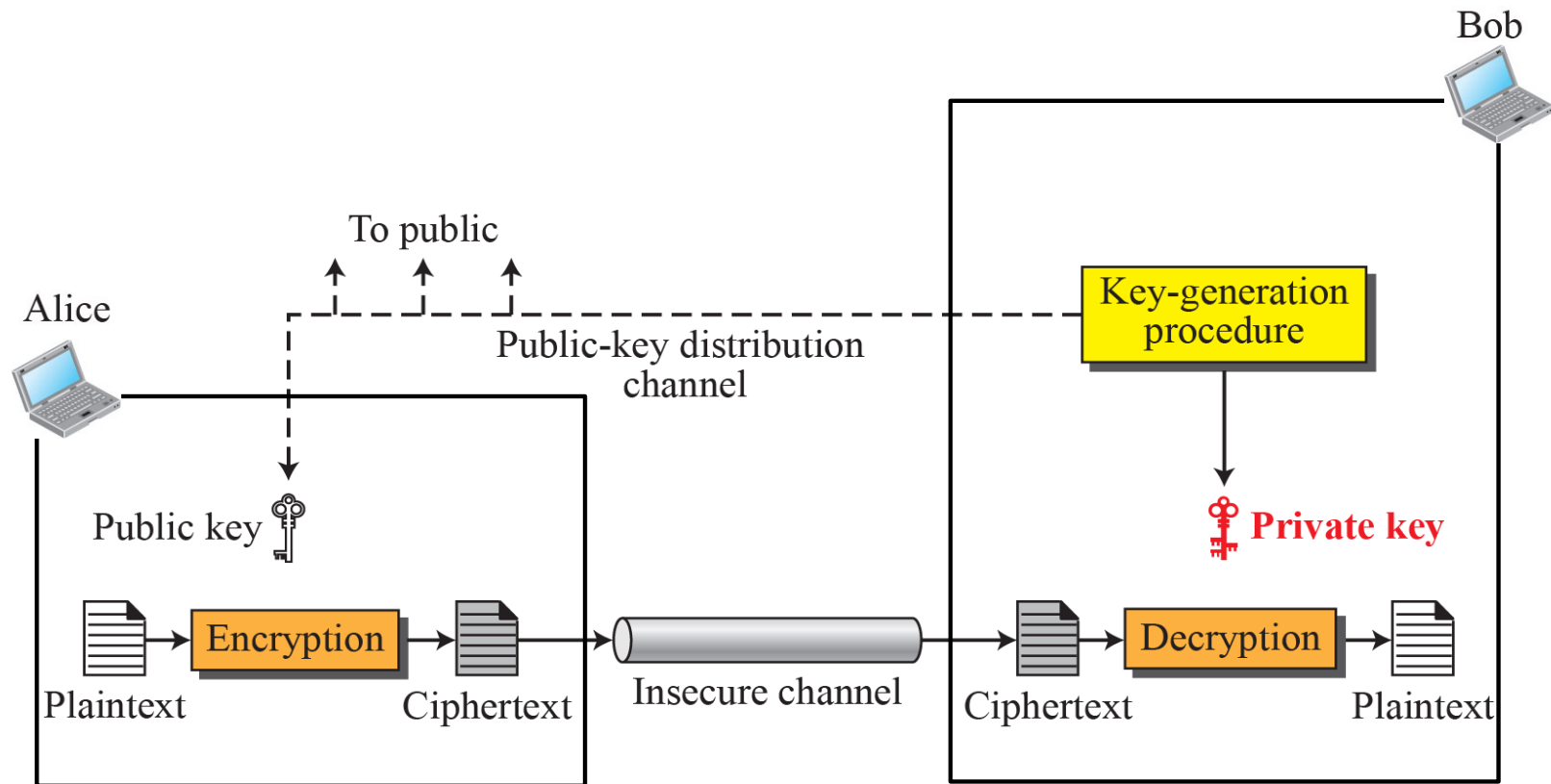
- symmetric-key NIST standard, replaced DES (Nov 2001)
- processes data in 128 bit blocks
- 128, 192, or 256 bit keys
- brute force decryption (try each key) taking 1 sec on DES, takes 149 trillion years for AES

Confidentiality with Asymmetric/Public Key Encryption

- No key sharing is needed
- Use two separate keys: one private and one public
- **Public Key** (exposed to the public domain) is used for encryption
- **Private Key** (private to the user) is used for decryption
- Public key locks the message and Private key unlocks it



Confidentiality with Asymmetric/Public Key Encryption



Confidentiality with Asymmetric/Public Key Encryption

Pros:

- No need for secret key sharing
- It is computationally infeasible to compute the Private Key given the knowledge of the cryptographic algorithm and the Public Key.
- Used for authentication, digital signatures, and secrete-key exchanges

Cons:

- Relatively long mathematical calculations using relatively long keys
- Much slower than symmetric-key encryption
- Inefficient for long messages
- Public key must be distributed via a trusted third party

Public Key Encryption Algorithms

- **RSA** (Rivest, Shamir, Adleman) Developed in 1978 at MIT.
- De-facto standard for data transmission on the Internet and built into many software including Netscape Navigator and Internet Explorer.
- RSA uses two exponents, e and d , where e is public, and d is private.
- Suppose P is the plaintext and C is the ciphertext
 - Alice uses $C = P^e \bmod n$ to create cipher-text C from plaintext P ;
 - Bob uses $P = C^d \bmod n$ to retrieve the plaintext sent by Alice.
- The modulus n , a very large number, is created during the key generation process.

RSA: getting ready

- message: just a bit pattern
- bit pattern can be uniquely represented by an integer number
- thus, encrypting a message is equivalent to encrypting a number

example:

- $m = 10010001$. This message is uniquely represented by the decimal number $2^7 + 2^4 + 2^0 = 145$.
- to encrypt m , we encrypt the corresponding number, which gives a new number (the ciphertext).

RSA: Creating public/private key pair

1. choose two large prime numbers p, q . (e.g., 1024 bits each)
2. compute $n = pq$, $z = (p-1)(q-1)$
3. choose e (with $e < n$) that has no common factors with z (e, z are “relatively prime”).
4. choose d such that $ed-1$ is exactly divisible by z . (in other words: $ed \bmod z = 1$).
5. *public* key is $\underbrace{(n, e)}_{K_B^+}$. *private* key is $\underbrace{(n, d)}_{K_B^-}$.

RSA: encryption, decryption

0. given (n, e) and (n, d) as computed above
1. to encrypt message $m (< n)$, compute
 $c = m^e \bmod n$
2. to decrypt received bit pattern, c , compute
 $m = c^d \bmod n$

magic happens!
$$m = (\underbrace{m^e \bmod n}_c)^d \bmod n$$

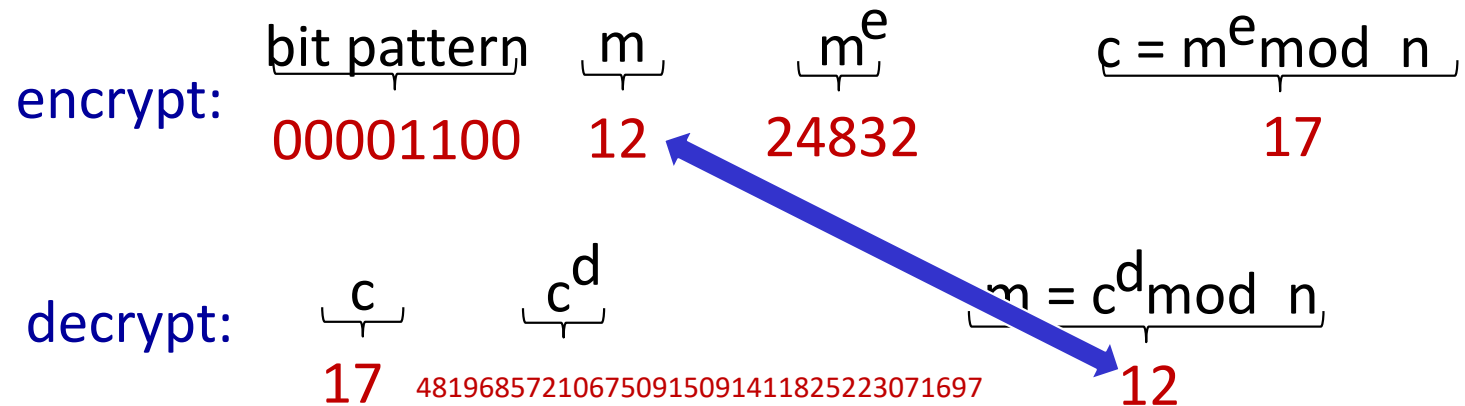
RSA example

Bob chooses $p=5$, $q=7$. Then $n=35$, $z=24$.

$e=5$ (so e , z relatively prime).

$d=29$ (so $ed-1$ exactly divisible by z).

encrypting 8-bit messages.



RSA: another important property

The following property will be *very* useful later:

$$\underbrace{K_B^-(K_B^+(m))}_{\text{use public key first, followed by private key}} = m = \underbrace{K_B^+(K_B^-(m))}_{\text{use private key first, followed by public key}}$$

use public key
first, followed
by private key

use private key
first, followed
by public key

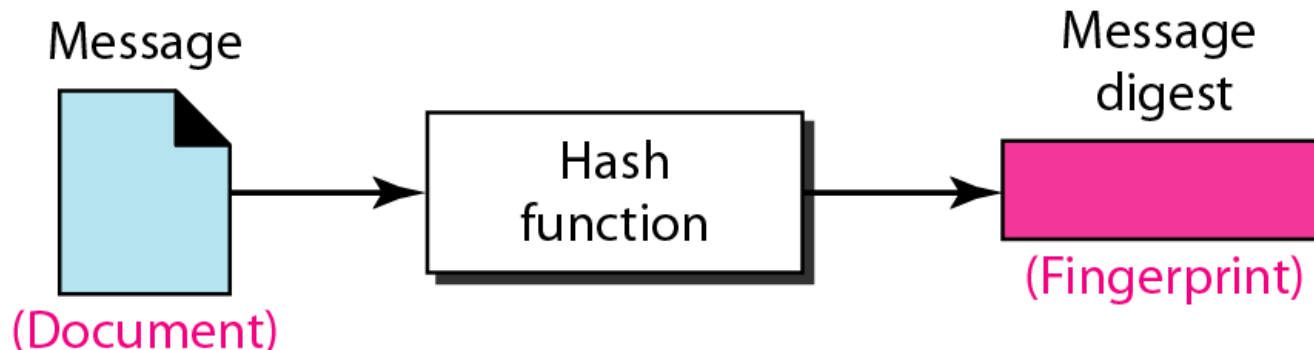
result is the same!

Why is RSA secure?

- suppose you know Bob's public key (n,e) . How hard is it to determine d ?
- essentially need to find factors of n without knowing the two factors p and q
 - fact: factoring a big number is hard

Message Integrity

- Encryption/decryption provides confidentiality not integrity
- Integrity: No one tamper with the message or can forge it
- One way to preserve integrity is to use a **Hash function** to create a **Message digest** (fingerprint) of a message
- Message digest: compressed image of the message that can be used like a fingerprint, also known as a **Modification Detection Code (MDC)**
- Message digest must be kept secret and must be encrypted when sent through a communication channel

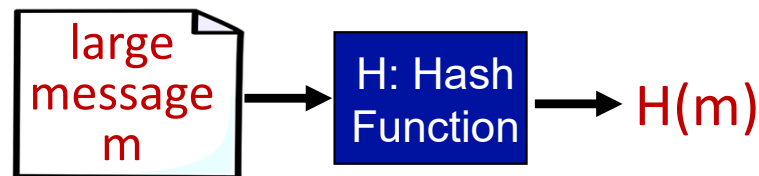


Message digests

computationally expensive to public-key-encrypt long messages

goal: fixed-length, easy- to-compute digital “fingerprint”

- apply hash function H to m , get fixed size message digest, $H(m)$



Hash function properties:

- many-to-1
- produces fixed-size msg digest (fingerprint)
- given message digest x , computationally infeasible to find m such that $x = H(m)$

Internet checksum poor crypto hash function

Internet checksum has some properties of hash function:

- produces fixed length digest (16-bit sum) of message
- is many-to-one

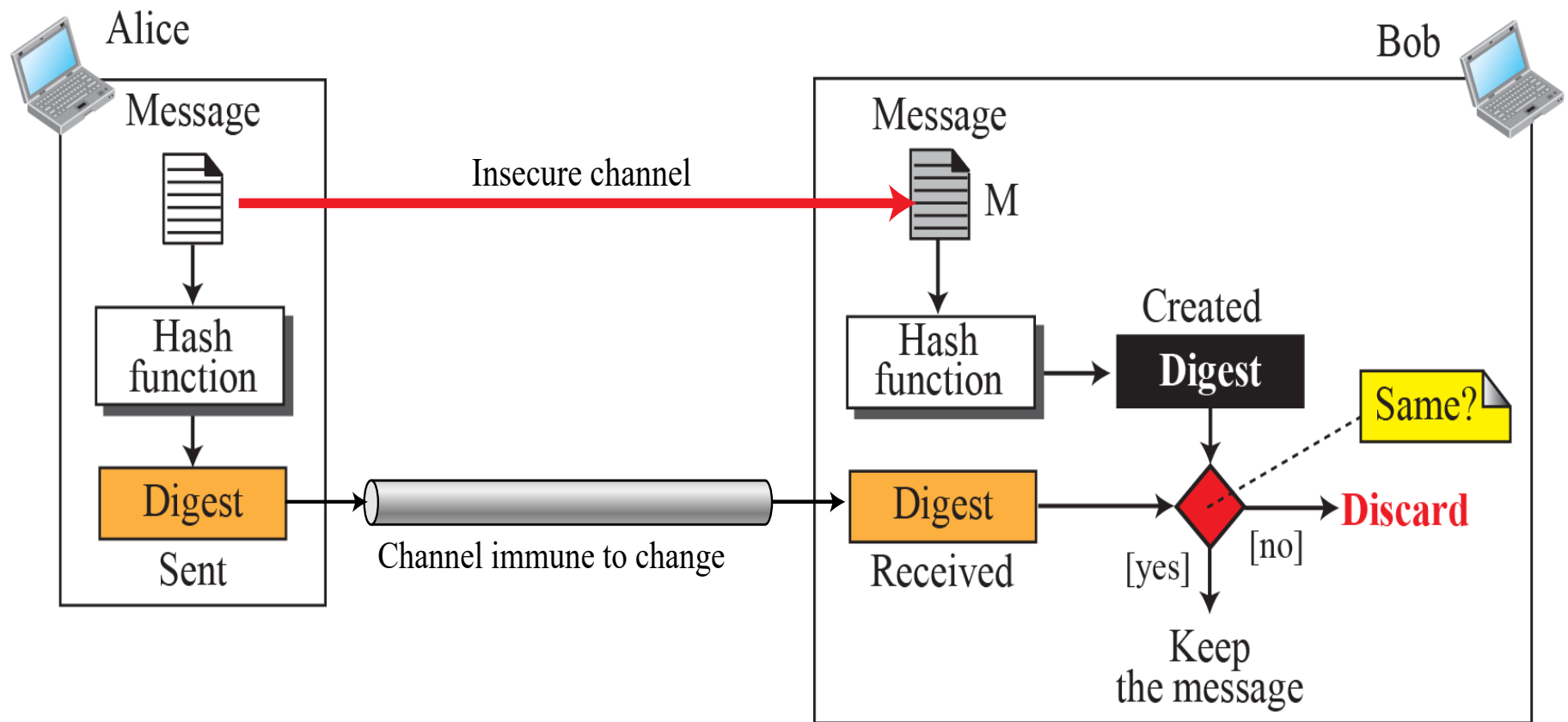
but given message with given hash value, it is easy to find another message with same hash value:

<u>message</u>	<u>ASCII format</u>		<u>message</u>	<u>ASCII format</u>
I O U 1	49 4F 55 31		I O U <u>9</u>	49 4F 55 <u>39</u>
0 0 . 9	30 30 2E 39		0 0 . <u>1</u>	30 30 2E <u>31</u>
9 B O B	39 42 D2 42		9 B O B	39 42 D2 42
B2 C1 D2 AC		 <i>different messages</i> 	B2 C1 D2 AC	
<i>but identical checksums!</i>				

Hash function algorithms

- MD5 hash function widely used (RFC 1321)
 - computes 128-bit message digest in 4-step process.
 - arbitrary 128-bit string x , appears difficult to construct msg m whose MD5 hash is equal to x
- SHA-1 is also used
 - US standard [NIST, FIPS PUB 180-1]
 - 160-bit message digest

Integrity Checking

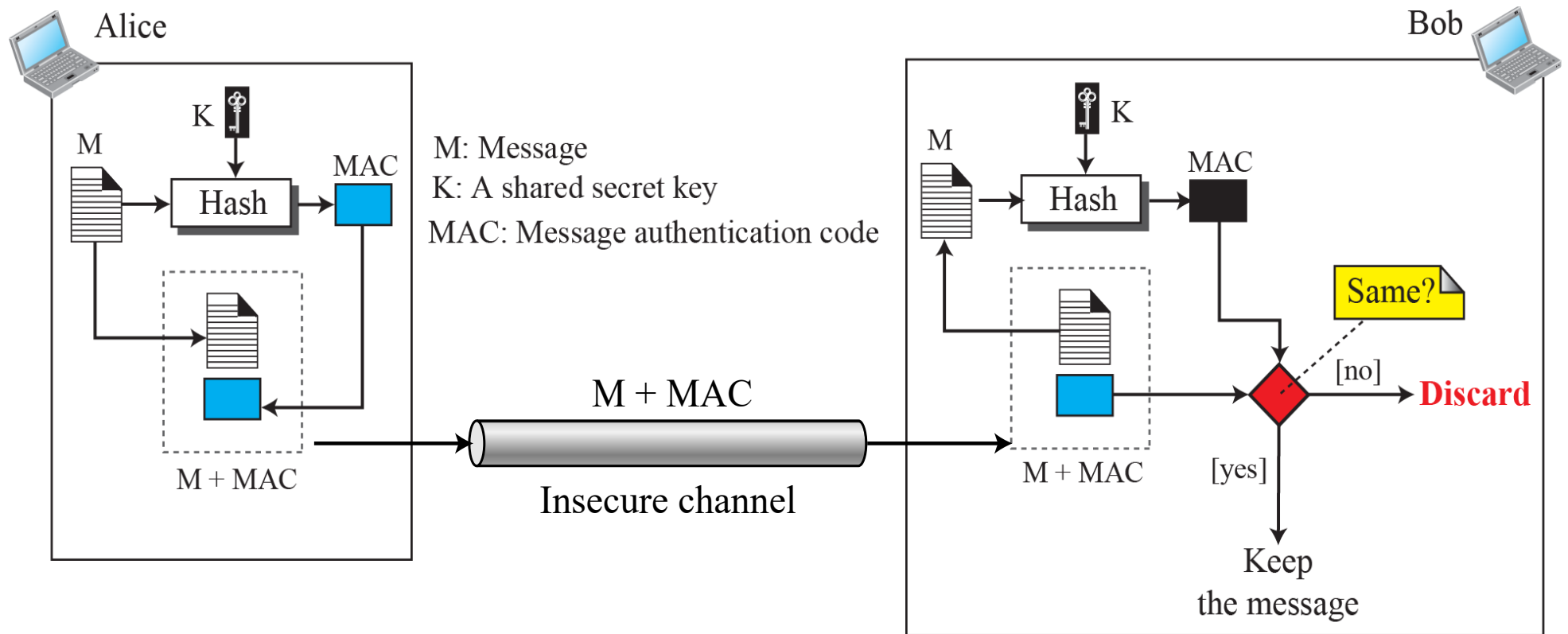


Message
unaltered

Message Authentication

- Hash function guarantees that the message has not been changed.
- However, it **does not authenticate** the sender of the message
- One way is to change the MDC (Message Digest) into a **Message Authentication Code (MAC)**
- A MAC uses a **keyed Hash Function** (i.e., A symmetric key is used when creating the digest)
- If the MAC is matched, then the message can be authenticated

MAC Validation

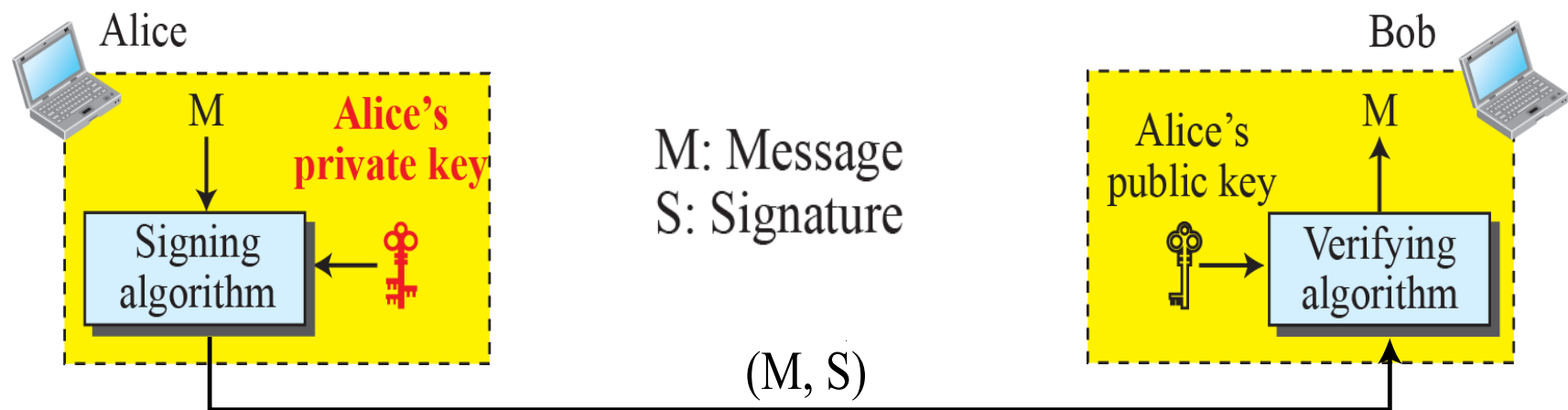


Digital Signature

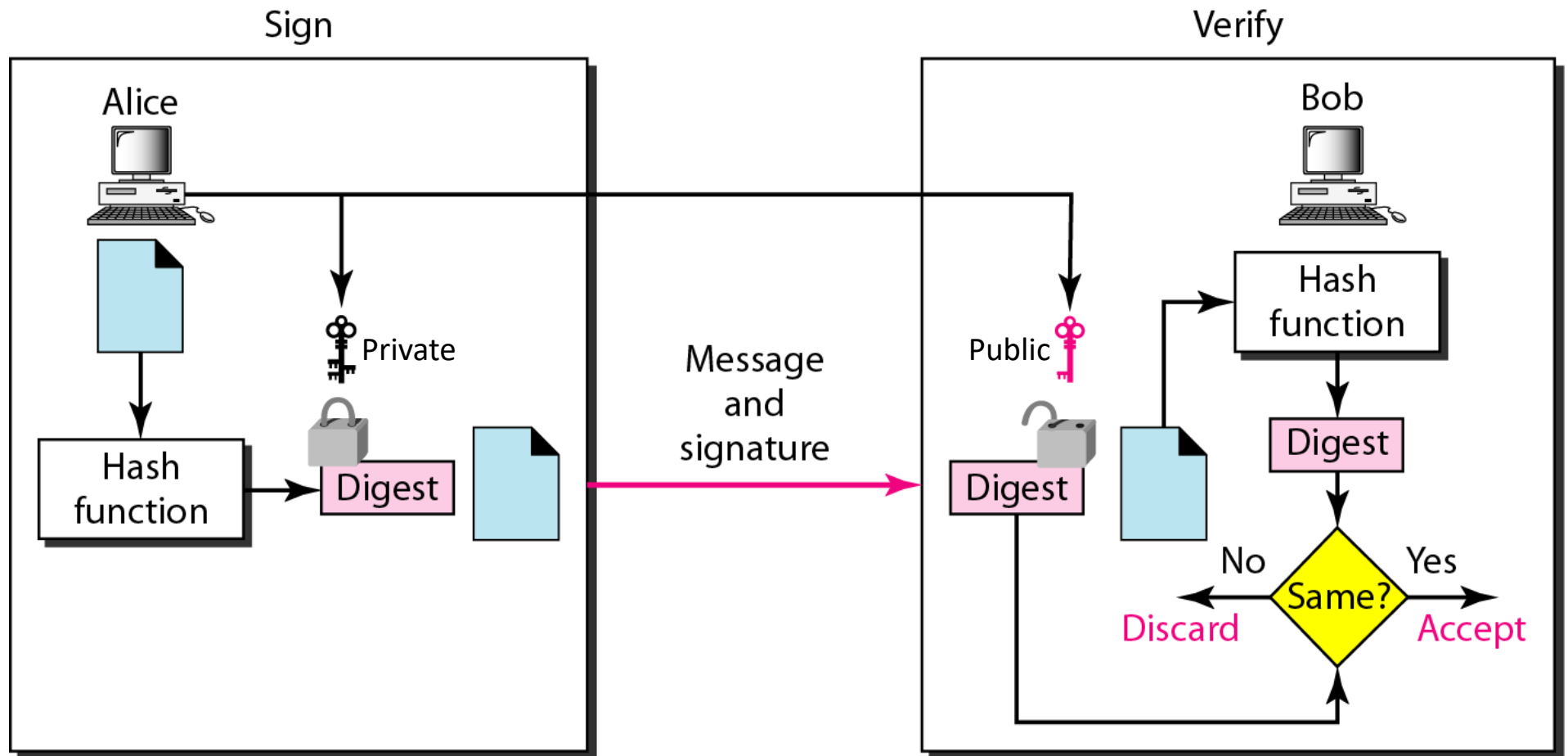
- Although a MAC provides integrity and authentication, this is done via a shared symmetric key.
- A **Digital Signature** uses a **pair of asymmetric keys** (a public one and a private one) to prove the **authenticity** of the message.
- This could be achieved in two ways:
 - Signing a document
 - Signing a digest of a document
- Services offered:
 - **Integrity**
 - **Authentication**
 - **Message Nonrepudiation (by using a trusted third party)**

Signing the Document

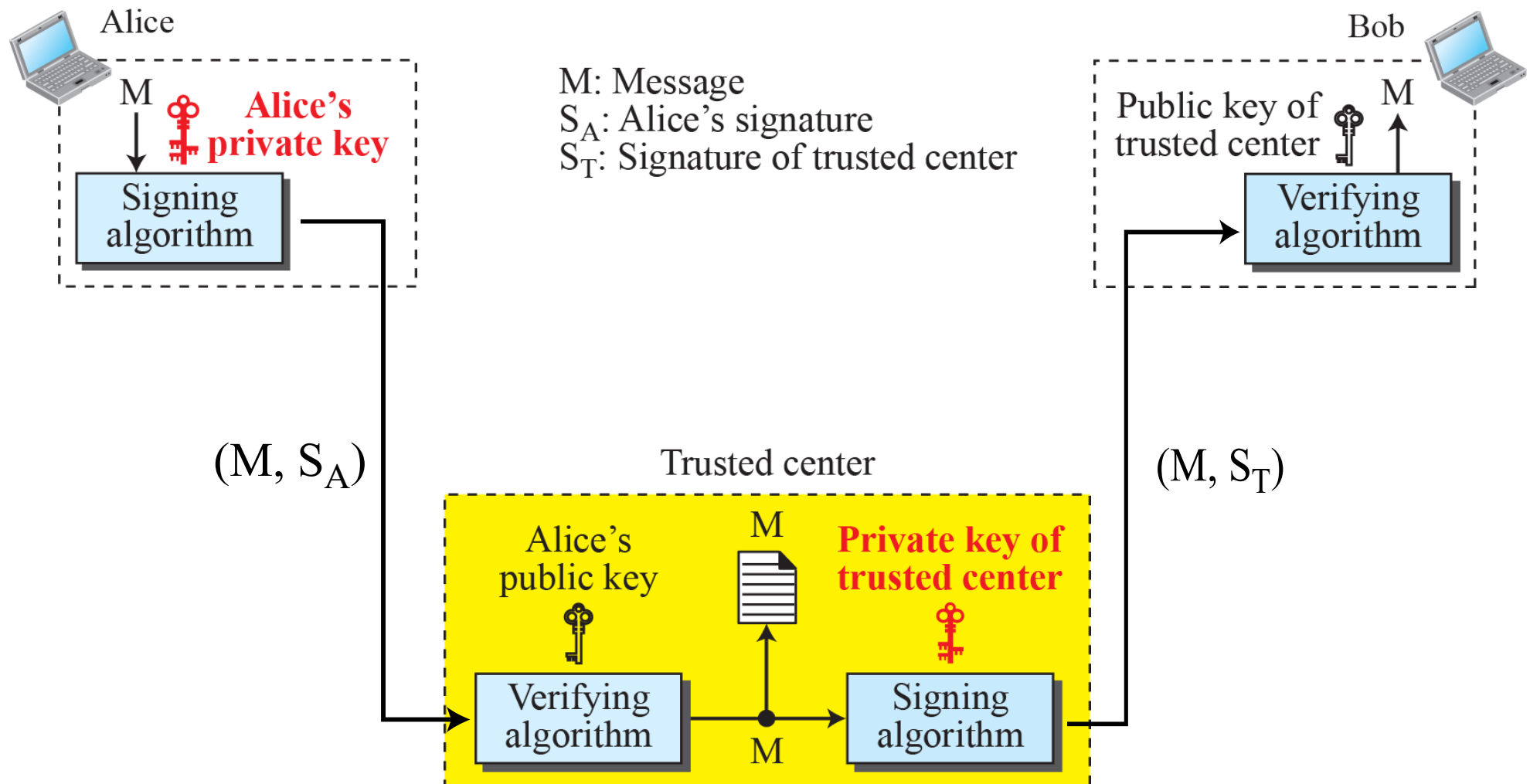
- In a cryptosystem, we use the private and public keys of the receiver
- In digital signature, we use the private and public keys of the sender



Signing the Digest



Using a trusted third party for message nonrepudiation



Trusted Center confirms Alice's identity

Entity Authentication

- Used when one party (claimant) **prove the identity** to another party (verifier)
- Entity can be a person, a process, a client or a server
- Differences between message and entity authentication:
 - Entity authentication is **real-time**
 - Entity authentication authenticates the claimant for the **entire duration of the session** (not message by message)
- Identification of claimant done with three kinds of witnesses:
 - **Something Known:** password, PIN number, secret key or private key
 - **Something Possessed:** passport, driver's license, ID card, credit card, smart card
 - **Something Inherent:** handwriting, conventional signature, biometrics

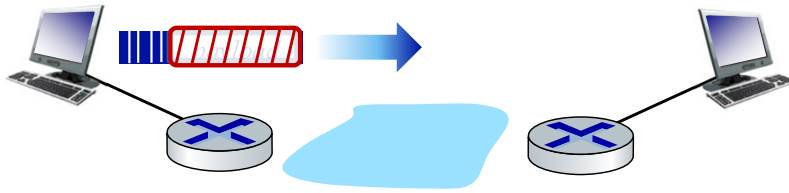
Transport-layer security (TLS)

- widely deployed security protocol above the transport layer
 - supported by almost all browsers, web servers: https (port 443)
- provides:
 - **confidentiality**: via *symmetric encryption*
 - **integrity**: via *cryptographic hashing*
 - **authentication**: via *public key cryptography*
- history:
 - early research, implementation: secure network programming, secure sockets
 - secure socket layer (SSL) deprecated [2015]
 - TLS 1.3: RFC 8846 [2018]

*all techniques we
have studied!*

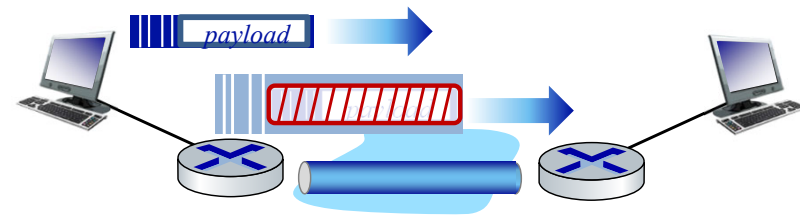
Network Layer Security

- provides datagram-level encryption, authentication, integrity
 - for both user traffic and control traffic (e.g., BGP, DNS messages)
- two “modes”:



transport mode:

- *only* datagram *payload* is encrypted, authenticated



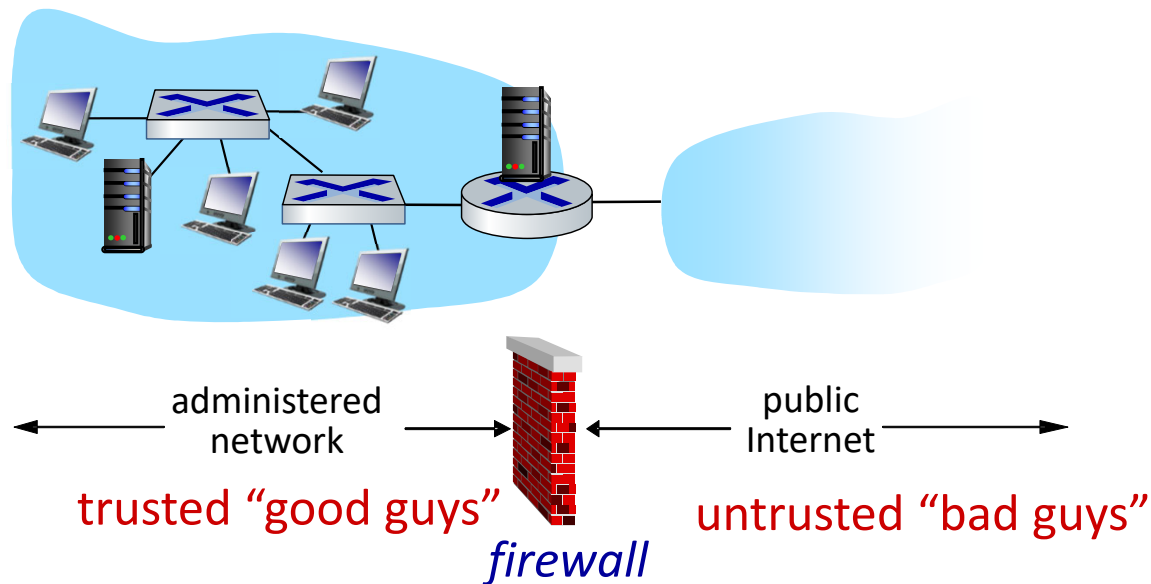
tunnel mode:

- entire datagram is encrypted, authenticated
- encrypted datagram encapsulated in new datagram with new IP header, tunneled to destination

Firewalls

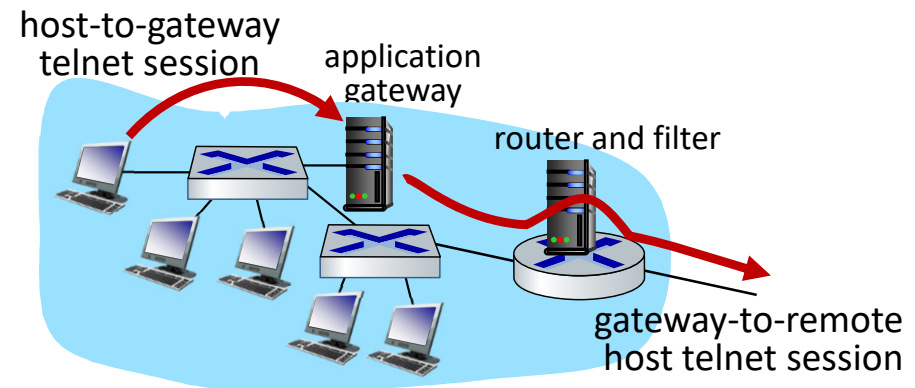
firewall

isolates organization's internal network from larger Internet, allowing some packets to pass, blocking others



Application gateways

- filter packets on application data as well as on IP/TCP/UDP fields.
- *example:* allow select internal users to telnet outside



1. require all telnet users to telnet through gateway.
2. for authorized users, gateway sets up telnet connection to dest host
 - gateway relays data between 2 connections
3. router filter blocks all telnet connections not originating from gateway

Limitations of firewalls, gateways

- **IP spoofing:** router can't know if data “really” comes from claimed source
- if multiple apps need special treatment, each has own app. gateway
- client software must know how to contact gateway
 - e.g., must set IP address of proxy in Web browser
- filters often use all or nothing policy for UDP
- *tradeoff:* degree of communication with outside world, level of security
- many highly protected sites still suffer from attacks

Recommended Reading

- J. F. Kurose and K. W. Ross, Computer Networking: A Top-Down Approach, 8th ed., 2022, Chapter 8