Network Security

Kameswari Chebrolu

What is network security?

Confidentiality: only sender, intended receiver should "understand" message contents

- sender encrypts message
- receiver decrypts message

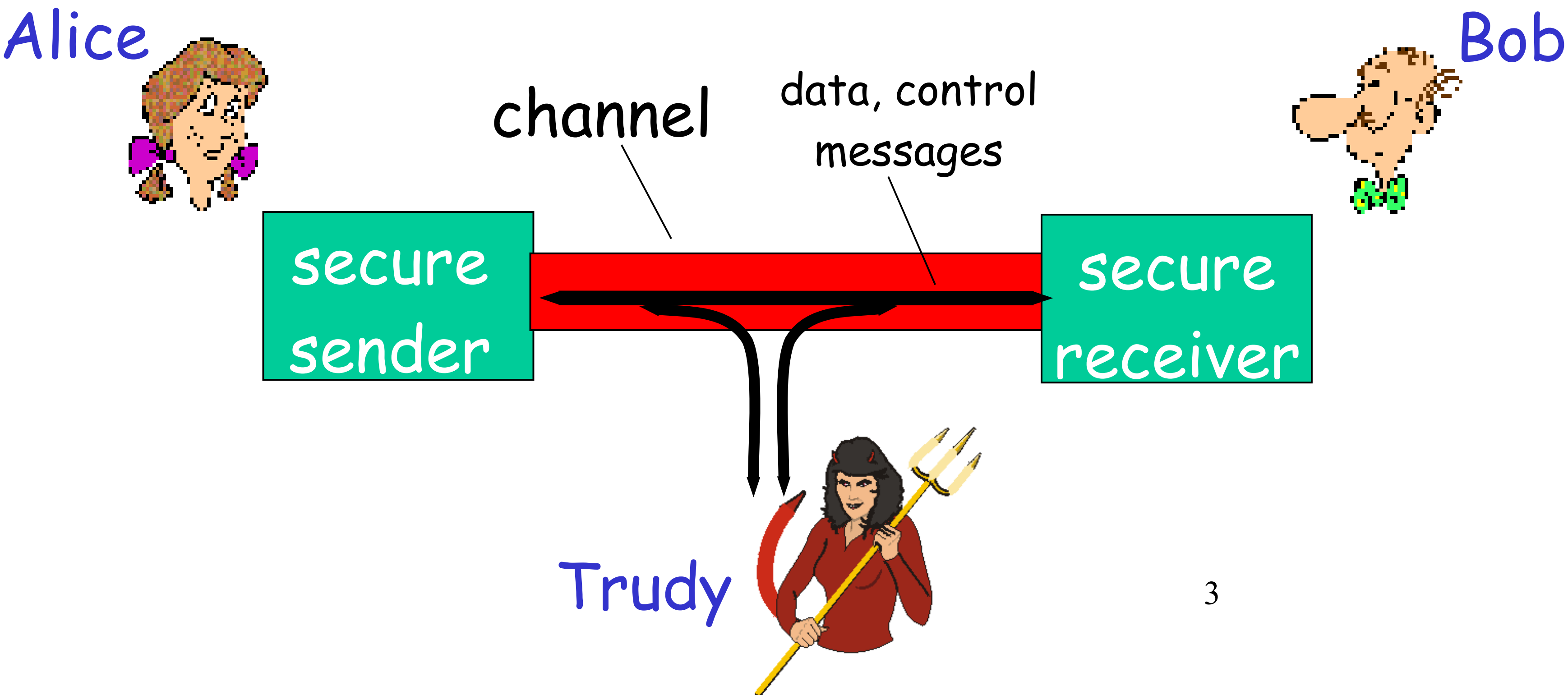
Authentication: sender, receiver want to confirm identity of each other

Message Integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection

Access and Availability: services must be accessible and available to users

Friends and enemies: Alice, Bob, Trudy

- Well-known in network security world
- Bob, Alice (lovers!) want to communicate “securely”
- Trudy (intruder)



Who might Bob, 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
- Routers exchanging routing table updates

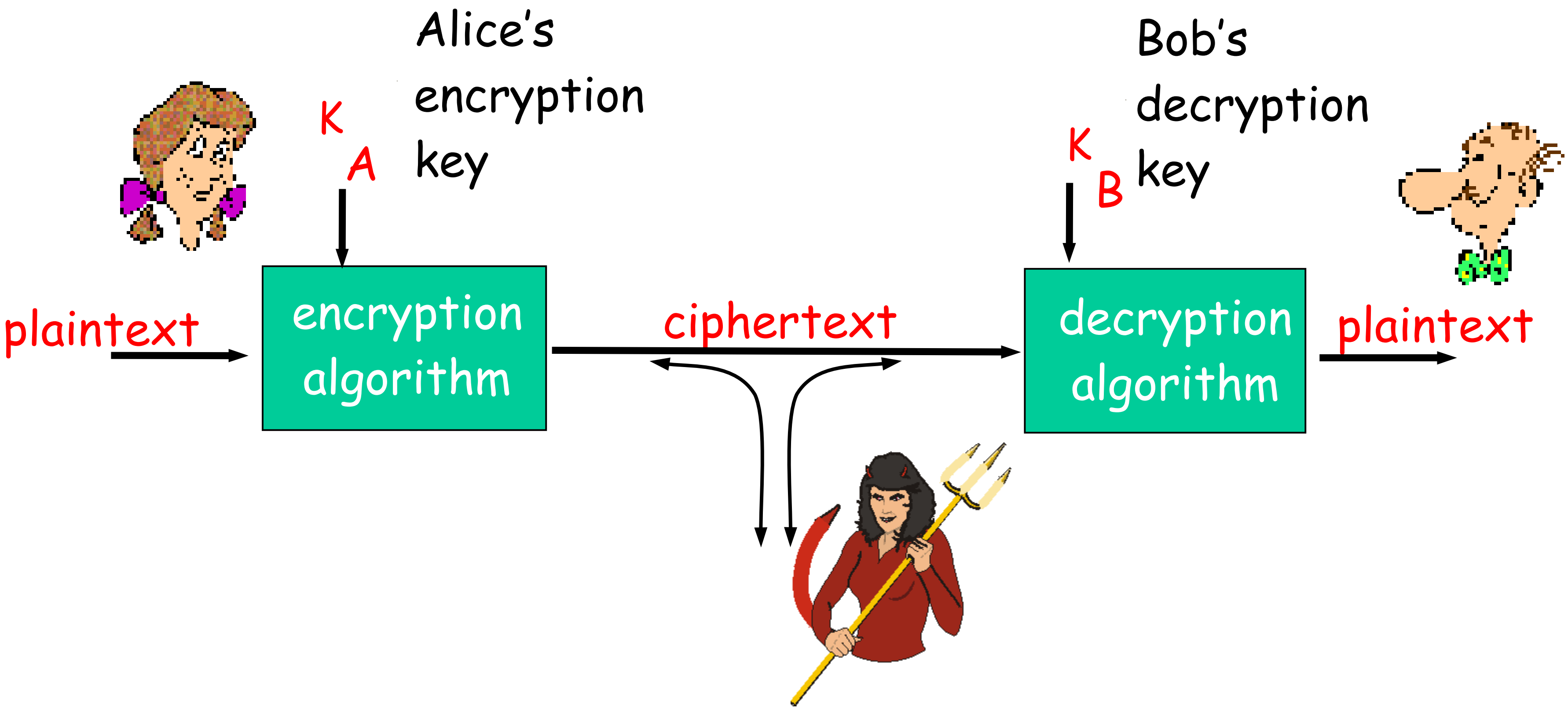
There are bad guys (and girls) out there!

Q: What can a "bad guy" do?

A: a lot!

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

The language of cryptography



symmetric key crypto: sender, receiver keys *identical*

public-key crypto: encryption key *public*, decryption key *secret* (private)

(Helps in Confidentiality)

Symmetric key cryptography

substitution cipher: substituting one thing for another

- monoalphabetic cipher: substitute one letter for another

```
plaintext:  abcdefghijklmnopqrstuvwxyz

ciphertext: mnbvcxzasdfghjklpoiuytrewq
```

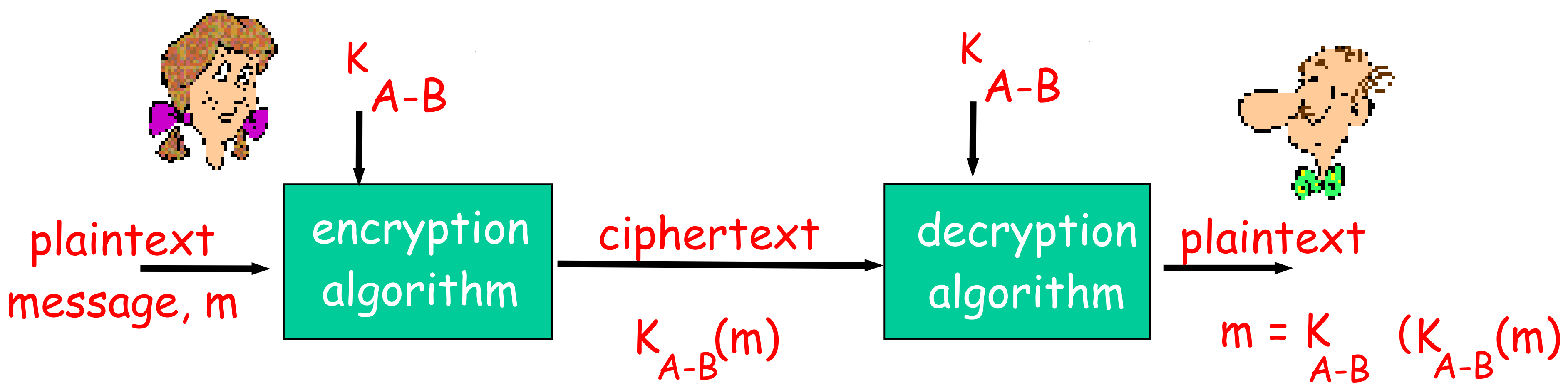
E.g.:

```
Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc
```

Q: How hard to break this simple cipher?:

- brute force (how hard?)
- other?

Symmetric key cryptography



symmetric key crypto: Bob and Alice share know same (symmetric) key: K_{A-B}

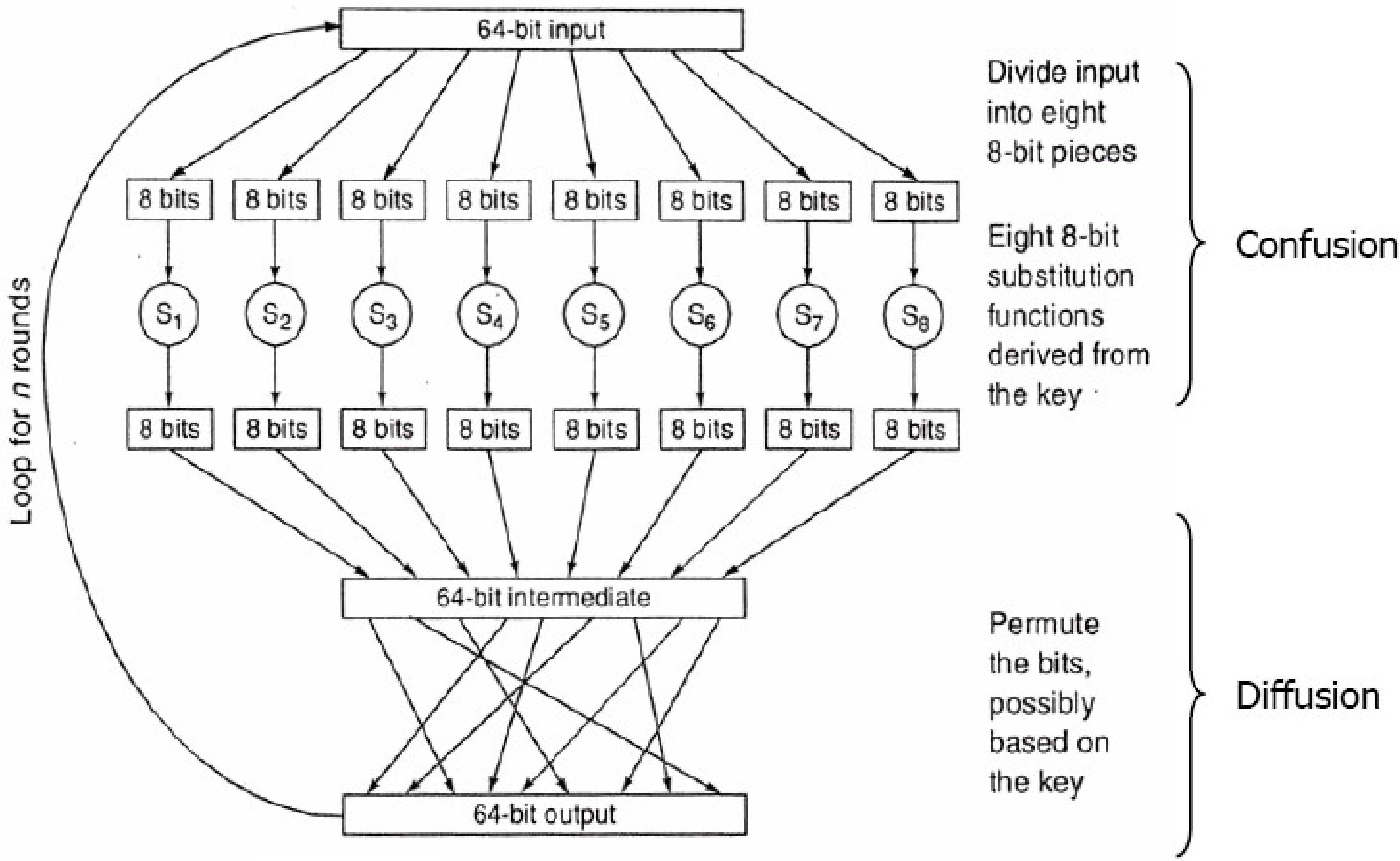
- e.g., key is knowing substitution pattern in mono alphabetic substitution cipher

Symmetric key crypto: DES

DES: Data Encryption Standard

- US encryption standard [NIST 1993]
- 56-bit symmetric key, 64-bit plaintext input
- How secure is DES?
    - DES Challenge: 56-bit-key-encrypted phrase ("Strong cryptography makes the world a safer place") decrypted (brute force) in 4 months (1997)
    - no known "backdoor" decryption approach
    - 1998-56 hours, 1999-22 hours
- 3-DES more secure than DES

Block Ciphers



Example of Block Encryption

AES: Advanced Encryption Standard

- (Nov. 2001) AES replaced DES
- Processes data in 128 bit blocks
- 128, 192, or 256 bit keys
- Fast implementation in hardware and software, not much memory (unlike DES)
- No successful attacks againts it so far
  - Brute force decryption taking 1 sec (2^55 key searches per sec) on DES, takes 149 trillion years (2^128 key) for AES

Key Sharing

- Requires sender, receiver to know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

Diffie-Hellman Key Exchange

- Alice and Bob agree to use a modulus p = 23 and base g = 5 (which is a primitive root modulo 23).
- Alice chooses a secret integer a = 6, then sends Bob
- A = g^a mod p (A = 5^6 mod 23 = 8)
- Bob chooses a secret integer b = 15, then sends Alice B = g^b mod p (B = 515 mod 23 = 19)
- Note: P, g, A, B are public
- Alice computes s = B^a mod p (s = 196 mod 23 = 2)
- Bob computes s = A^b mod p (s = 815 mod 23 = 2)

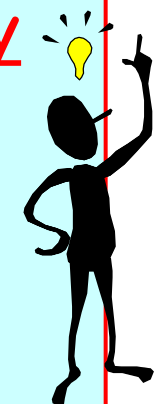
Alice and Bob now share a secret (the number 2)

Public Key Cryptography

<u>*symmetric* key crypto</u>

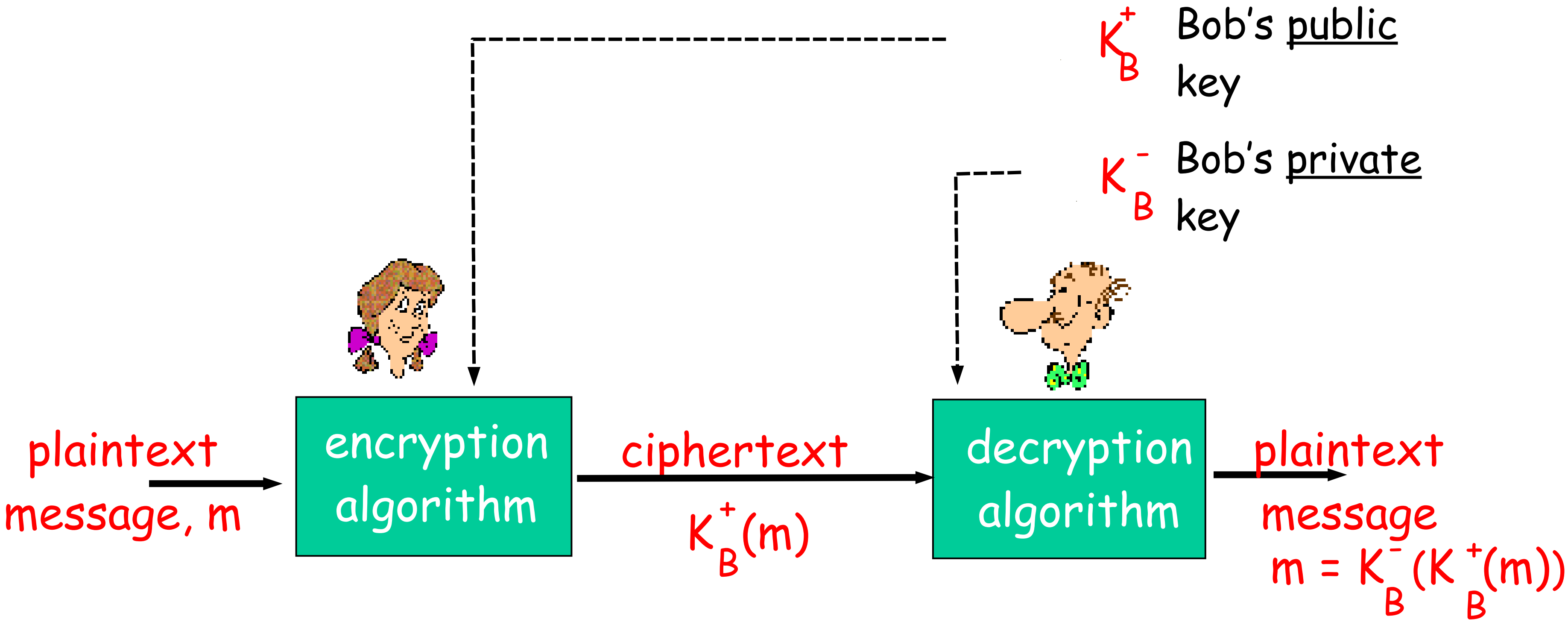
- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

<u>*public* key cryptography</u>



- radically different approach
- sender, receiver do *not* share secret key
- *public* encryption key known to *all*
- *private* decryption key known only to receiver

Public key cryptography



given public key, it should be impossible to compute private key

RSA: Choosing keys

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 bit pattern, m, compute

$c = m^e \bmod n$ (i.e., remainder when m^e is divided by n)

2. To decrypt received bit pattern, c, compute

$m = c^d \bmod n$ (i.e., remainder when c^d is divided by 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.

	letter	m	m^e	$c = m^e \bmod n$
encrypt:	l	12	1524832	17

	c	c^d	$m = c^d \bmod n$	letter
decrypt:	17	481968572106750915091411825223071697	12	l

RSA: another important property

The following property will be *very* useful later:

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

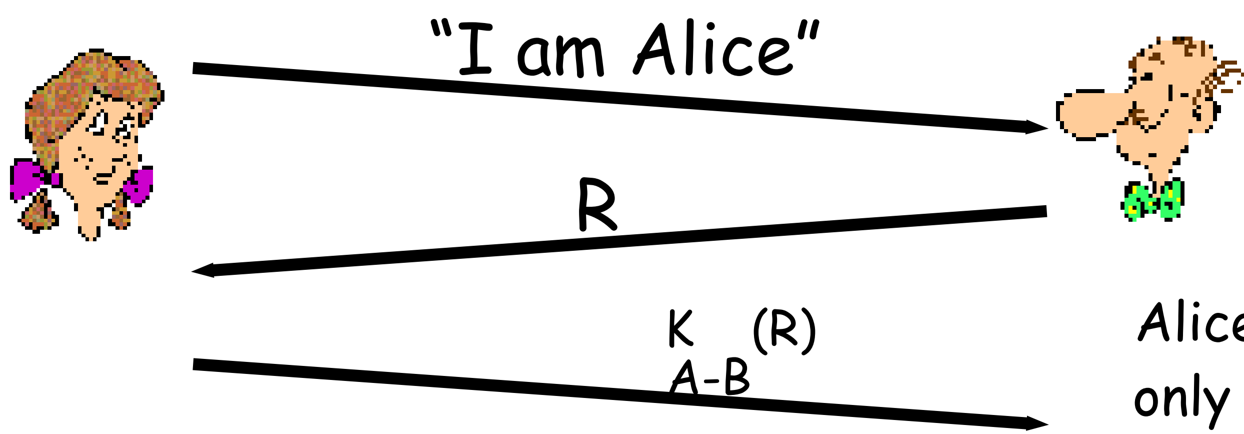
Result is the same!

Authentication: Symmetric Key

Goal: avoid playback attack

Nonce: number (R) used only *once –in-a-lifetime*

app: to prove Alice "live", Bob sends Alice nonce, R. Alice must return R, encrypted with shared secret key



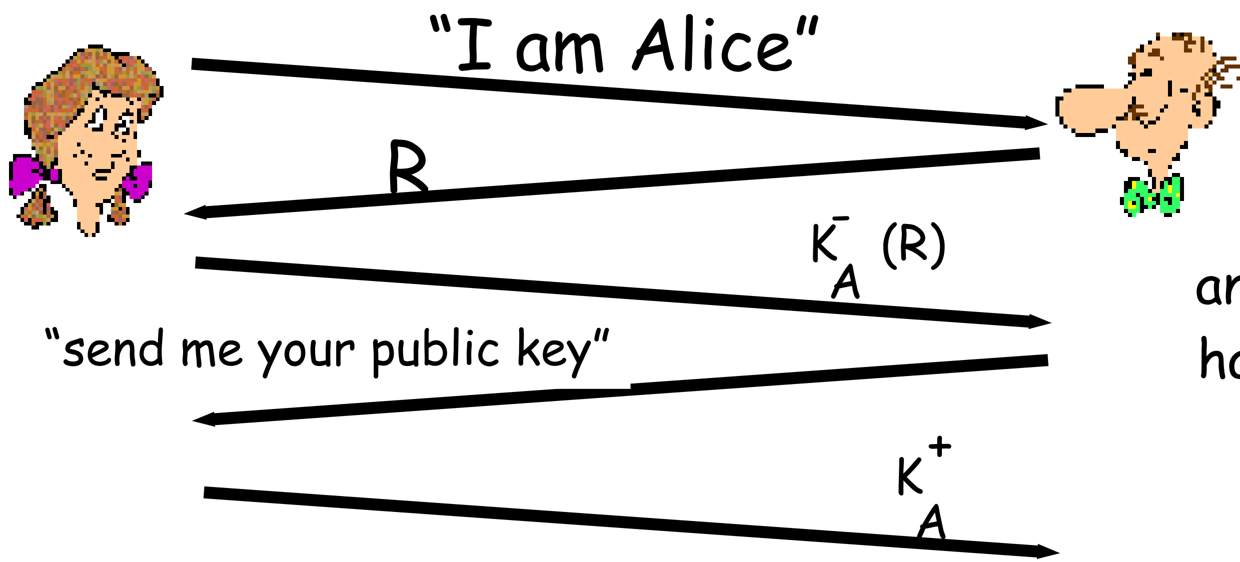
Alice is live, and only Alice knows key to encrypt nonce, so it must be Alice!

Authentication: Public Key

Previous approach requires shared symmetric key

- can we authenticate using public key techniques?

app: use nonce, public key cryptography



Bob computes

$$K_A^+(K_A^-(R)) = R$$

and knows only Alice could have the private key, that encrypted R such that

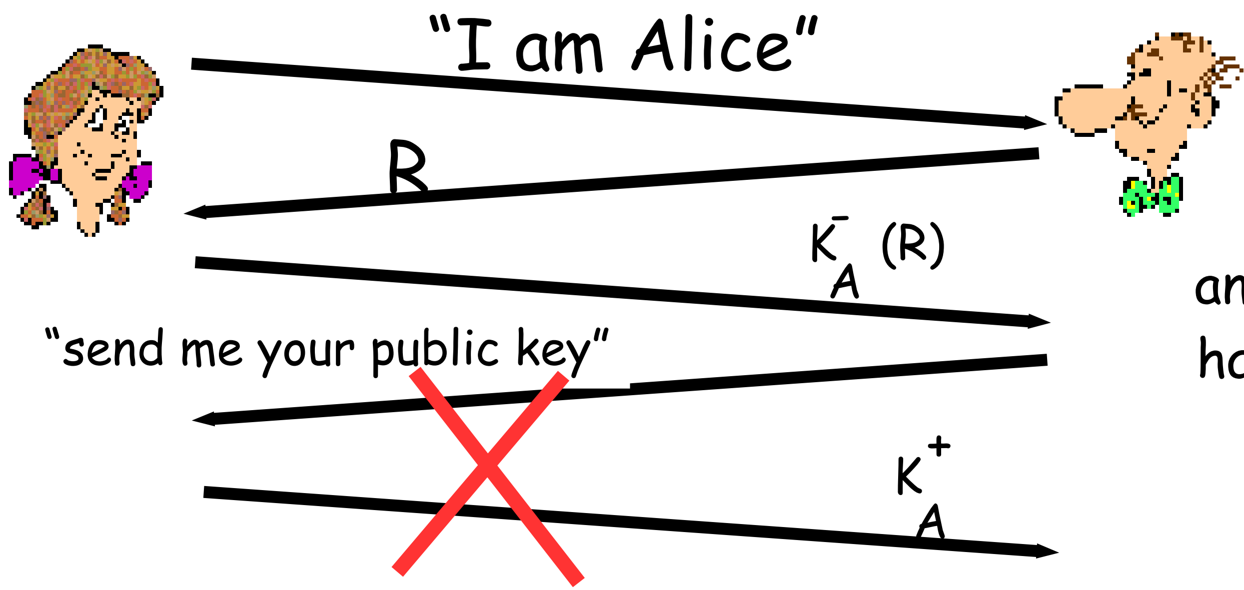
$$K_A^+(K_A^-(R)) = R$$

Authentication: Public Key

Previous approach requires shared symmetric key

- can we authenticate using public key techniques?

app: use nonce, public key cryptography



Bob computes

$$K_A^+(K_A^-(R)) = R$$

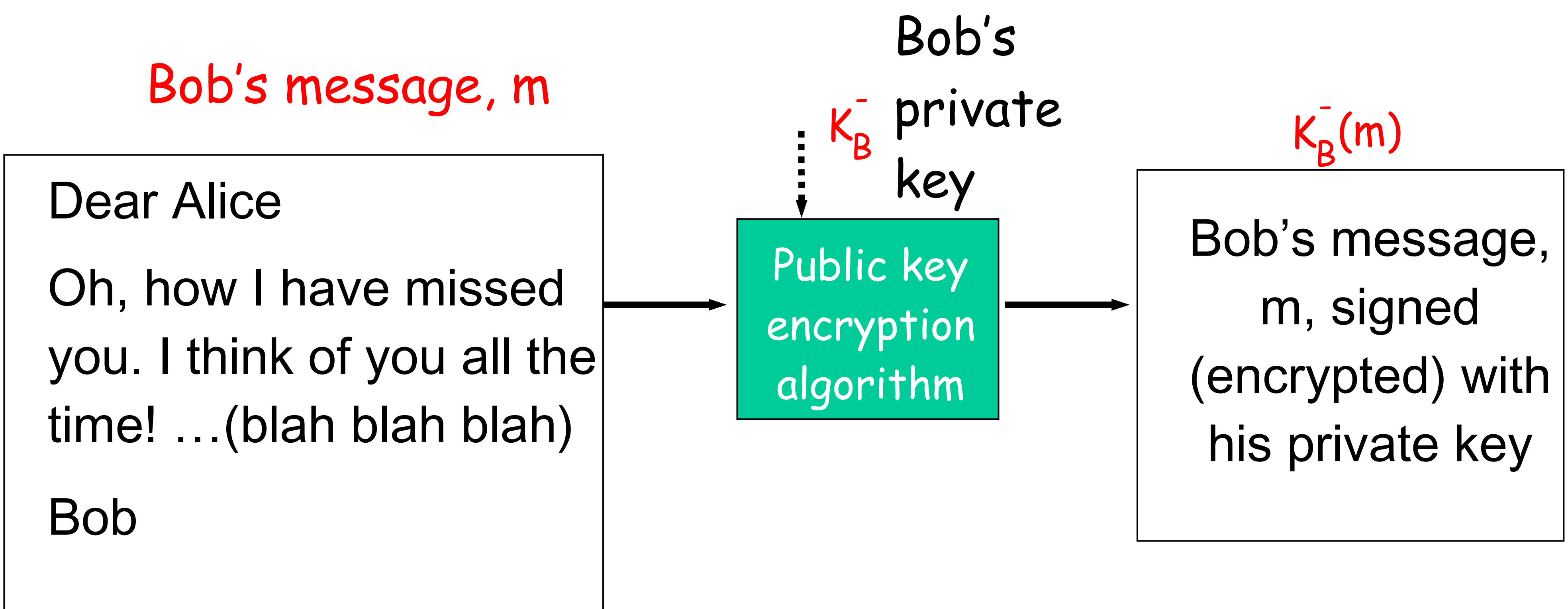
and knows only Alice could have the private key, that encrypted R such that

$$K_A^+(K_A^-(R)) = R$$

Digital Signatures

Simple digital signature for message m:

- Bob signs m by encrypting with his private key K_B^-, creating "signed" message, $K_B^-(m)$



Digital Signatures (more)

- Suppose Alice receives msg m, digital signature $K_B^-(m)$
- Alice verifies m signed by Bob by applying Bob's public key K_B^+ to $K_B^-(m)$ then checks $K_B^+(K_B^-(m)) = m$.
- If $K_B^+(K_B^-(m)) = m$, whoever signed m must have used Bob's private key.

Digital Signatures (more)

Alice thus verifies that:

- Bob signed m.
- No one else signed m.
- Bob signed m and not m'.

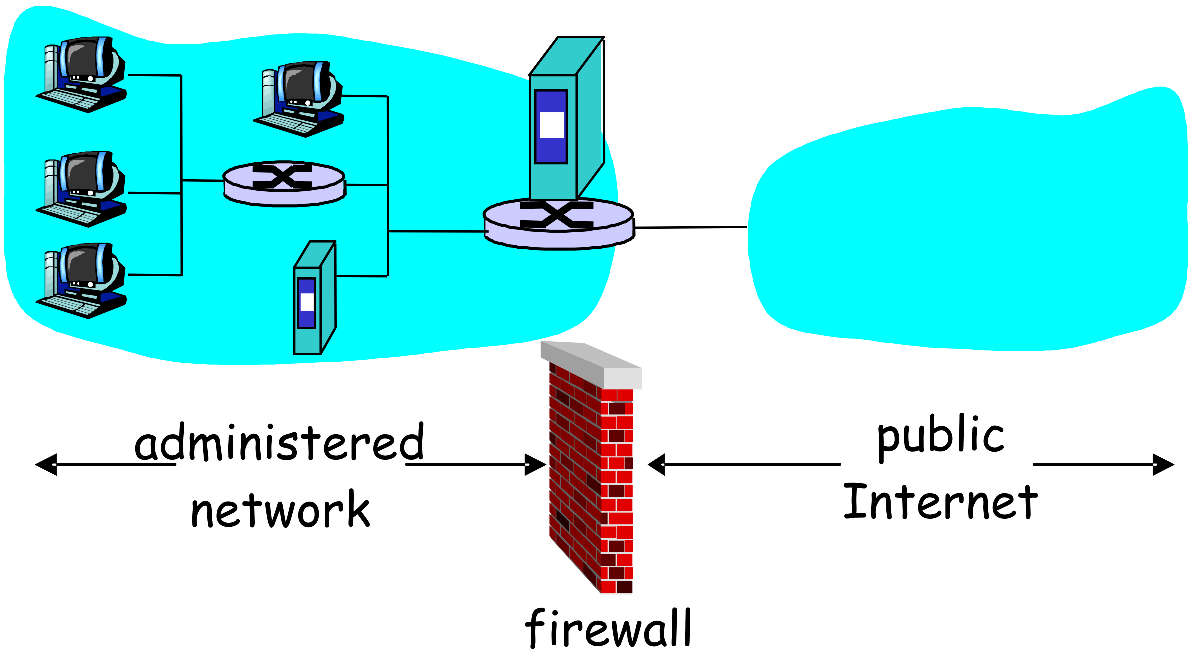
Non-repudiation:

- Alice can take m, and signature $K_B^-(m)$ to court and prove that Bob signed m.

Firewalls

firewall

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



Firewalls: Why

prevent denial of service attacks:

- SYN flooding: attacker establishes many bogus TCP connections, no resources left for "real" connections.

prevent illegal modification/access of internal data.

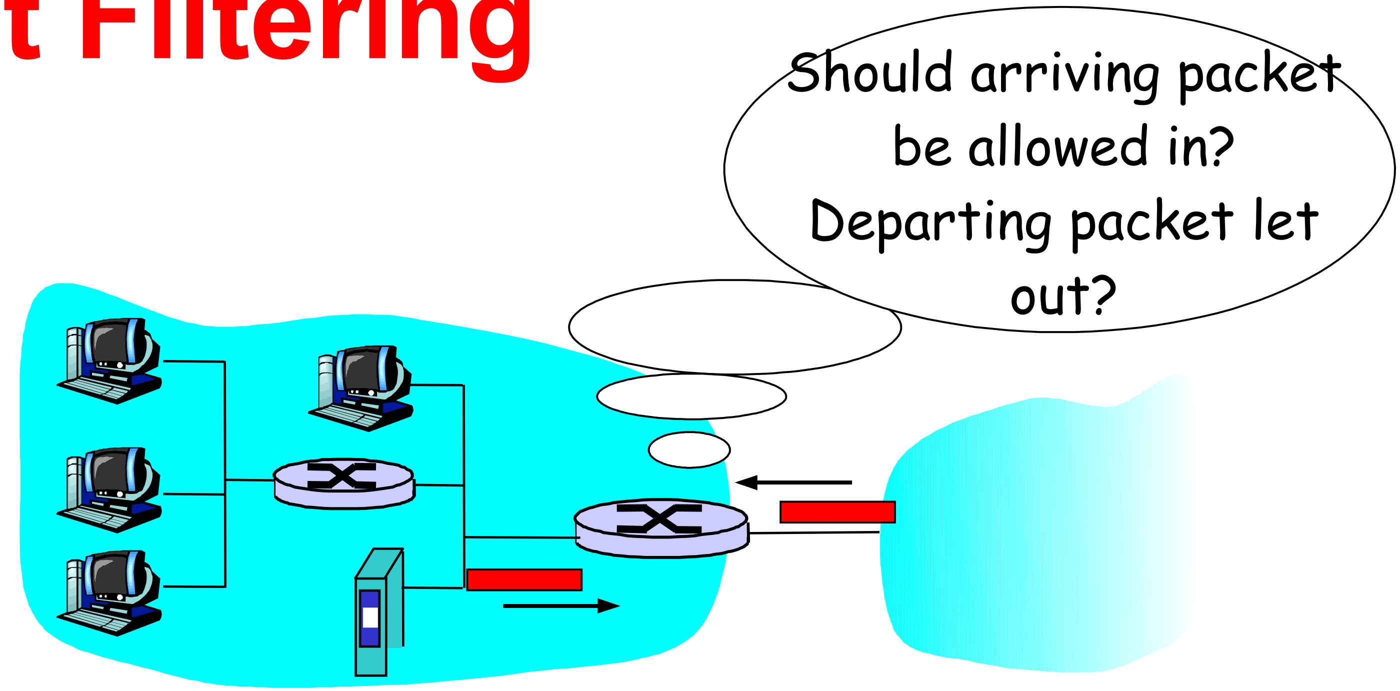
- e.g., attacker replaces IITB's homepage with something else

allow only authorized access to inside network (set of authenticated users/hosts)

two types of firewalls:

- application-level
- packet-filtering

Packet Filtering



- internal network connected to Internet via router firewall
- router filters packet-by-packet, decision to forward/drop packet based on:
  - source IP address, destination IP address
  - TCP/UDP source and destination port numbers
  - ICMP message type
  - TCP SYN and ACK bits

Packet Filtering

- Example 1: block incoming and outgoing datagrams with IP protocol field = 17 and with either source or dest port = 23.
  - All incoming and outgoing UDP flows and telnet connections are blocked.
- Example 2: Block inbound TCP segments with ACK=0.
  - Prevents external clients from making TCP connections with internal clients, but allows internal clients to connect to outside.

Access Control Lists

- ACL: table of rules, applied top to bottom to incoming packets: (action, condition) pairs

action	source address	dest address	protocol	source port	dest port	flag bit
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----
deny	all	all	all	all	all	all

Stateful packet filtering

- stateless packet filter: heavy handed tool

– admits packets that "make no sense," e.g., dest port = 80, ACK bit set, even though no TCP connection established:

action	source address	dest address	protocol	source port	dest port	flag bit
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK

- *stateful packet filter:* track status of every TCP connection
  - track connection setup (SYN), teardown (FIN): can determine whether incoming, outgoing packets "makes sense"
  - timeout inactive connections at firewall: no longer admit packets

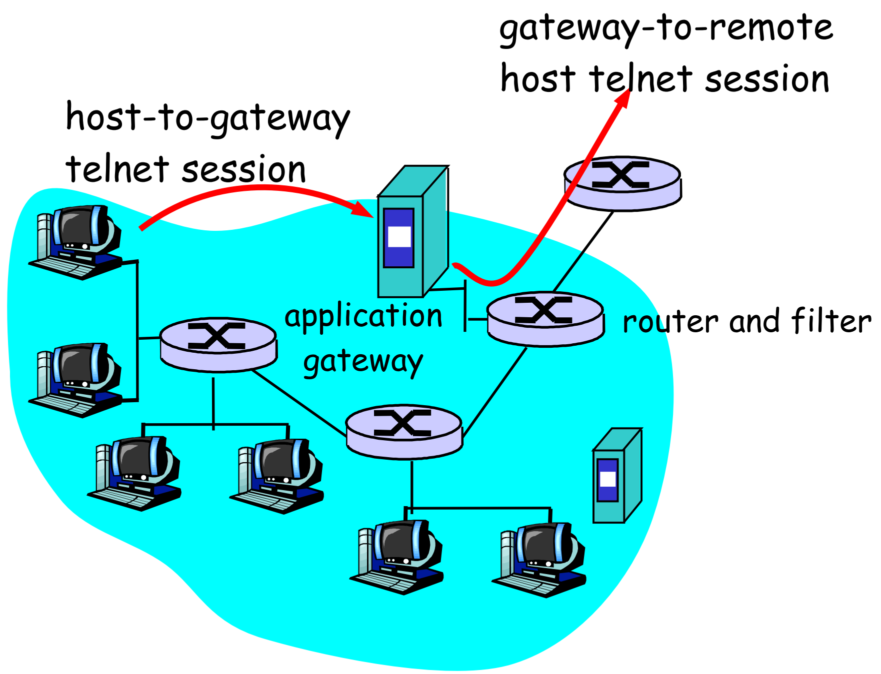
Stateful packet filtering

- ACL augmented to indicate need to check connection state table before admitting packet

action	source address	dest address	proto	source port	dest port	flag bit	check conxion
allow	222.22/16	outside of 222.22/16	TCP	> 1023	80	any	
allow	outside of 222.22/16	222.22/16	TCP	80	> 1023	ACK	X
allow	222.22/16	outside of 222.22/16	UDP	> 1023	53	---	
allow	outside of 222.22/16	222.22/16	UDP	53	> 1023	----	X
deny	all	all	all	all	all	all	

Application gateways

- Filters 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 and gateways

- IP spoofing: router can't know if data "really" comes from claimed source
- if multiple app's. 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.