Network Security

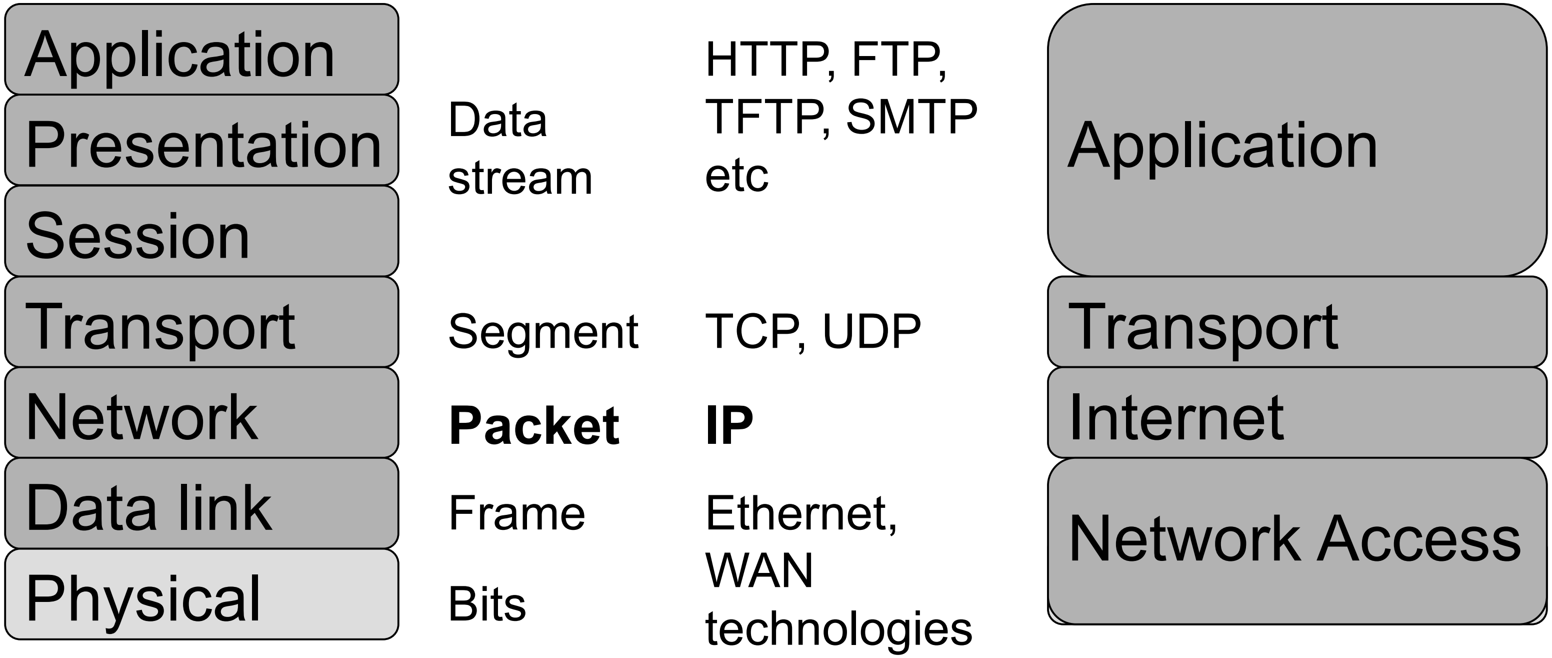
Dr MSK MANIKANDAN,
Associate Professor, Dept. of ECE,
Thiagarajar College of Engineering

Outline

- Cryptography
- Symmetric Key and Public Key Algorithm
- Digital Signature
- Management of Public Keys
- Communication Security
- Authentication Protocols

OSI model

TCP/IP model



OSI model			TCP/IP model
Application	Data stream	HTTP, FTP, TFTP, SMTP etc	Application
Presentation			
Session			
Transport	Segment	TCP, UDP	Transport
Network	**Packet**	**IP**	Internet
Data link	Frame	Ethernet, WAN technologies	Network Access
Physical	Bits		

Cryptography

- Cryptography is the study of

  “ Secret (crypto-) writing (-graphy)”

- Cryptographic goals includes
  - privacy or confidentiality
  - data integrity
  - authentication
  - non-repudiation

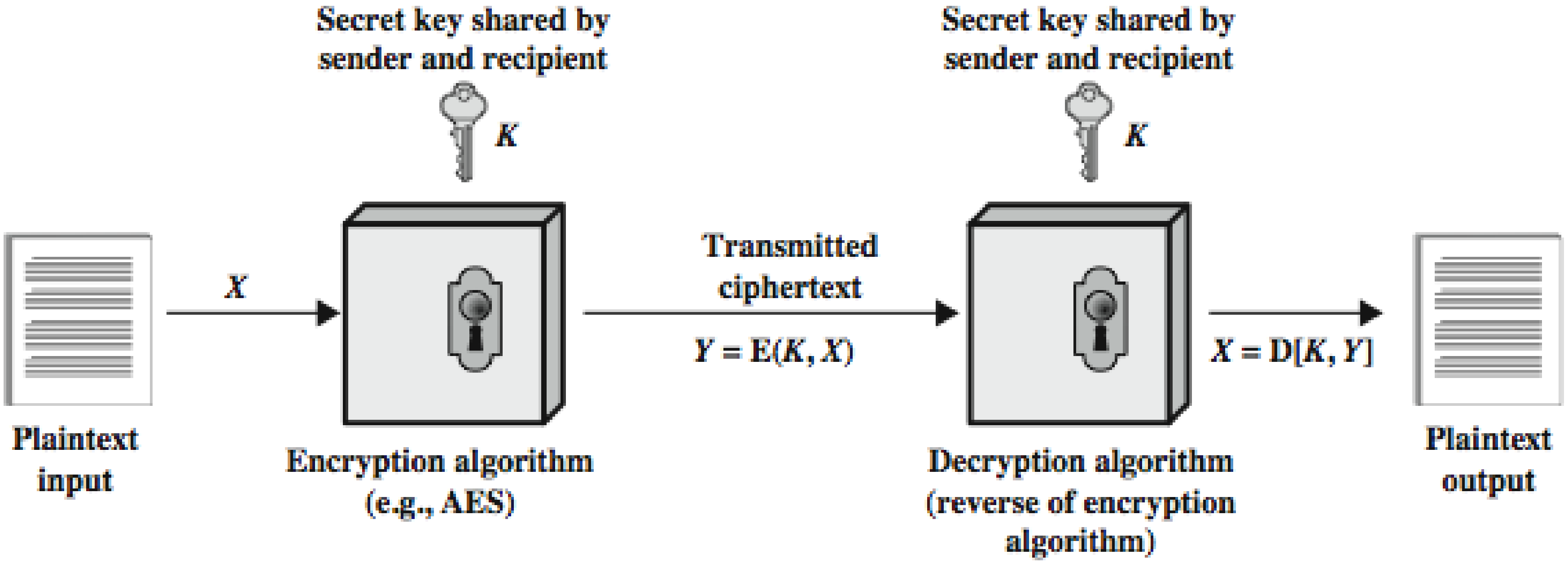
Basic Concepts

- **Plaintext:** original intelligible message
- **Ciphertext:** transformed message
- **Cipher:** algorithm for transforming an intelligible message into unintelligible by transposition and/or substitution
- **Key:** some critical information used by the cipher, known only to the sender & receiver

Basic Concepts

- **Encipher (encode):** process of converting plaintext to ciphertext
- **Decipher (decode):** process of converting ciphertext back into plaintext
- **Cryptanalysis:** study of principles and methods of transforming an unintelligible message back into an intelligible message without knowledge of the key.
- **Cryptology:** both cryptography and cryptanalysis
- **Code:** an algorithm for transforming an intelligible message into an unintelligible one using a code-book

Symmetric Cipher Model


Secret key shared by sender and recipient
K
Secret key shared by sender and recipient
K
X
Transmitted ciphertext
Y = E(K, X)
X = D[K, Y]
Plaintext input
Encryption algorithm (e.g., AES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Requirements

- two requirements for secure use of symmetric encryption:
  - a strong encryption algorithm
  - a secret key known only to sender / receiver
- mathematically have:

$$Y = \mathrm{E}(\mathrm{K}, X)$$

$$X = \mathrm{D}(\mathrm{K}, Y)$$

- assume encryption algorithm is known
- implies a secure channel to distribute key

Cryptography

- can characterize cryptographic system by:
  - type of encryption operations used
    - substitution
    - transposition
    - product
  - number of keys used
    - single-key or private
    - two-key or public
  - way in which plaintext is processed
    - block
    - stream

Cryptanalytic Attacks

- **ciphertext only**
  - only know algorithm & ciphertext, is statistical, know or can identify plaintext
- **known plaintext**
  - know/suspect plaintext & ciphertext
- **chosen plaintext**
  - select plaintext and obtain ciphertext
- **chosen ciphertext**
  - select ciphertext and obtain plaintext
- **chosen text**
  - select plaintext or ciphertext to en/decrypt

Classical Substitution Ciphers

- where letters of plaintext are replaced by other letters or by numbers or symbols
- or if plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with ciphertext bit patterns

Caesar Cipher

- earliest known substitution cipher
- by Julius Caesar
- first attested use in military affairs
- replaces each letter by 3rd letter on
- example:

  meet me after the toga party

  PHHW PH DIWHU WKH WRJD SDUWB

Caesar Cipher

- can define transformation as:

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

D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

- mathematically give each letter a number

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

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

- then have Caesar cipher as:

$c = \mathrm{E}(\mathrm{k}, p) = (p + k) \bmod (26)$

$p = \mathrm{D}(\mathrm{k}, \mathrm{c}) = (\mathrm{c} - k) \bmod (26)$

Cryptanalysis of Caesar Cipher

- only have 26 possible ciphers
  - A maps to A,B,..Z
- could simply try each in turn
- a **brute force search**
- given ciphertext, just try all shifts of letters
- do need to recognize when have plaintext
- eg. break ciphertext "GCUA VQ DTGCM"

Monoalphabetic Cipher

- rather than just shifting the alphabet
- could shuffle (jumble) the letters arbitrarily
- each plaintext letter maps to a different random ciphertext letter
- hence key is 26 letters long

Plain: abcdefghijklmnopqrstuvwxyz
Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN

Plaintext: ifwewishtoreplaceletters
Ciphertext: WIRFRWAJUHYFTSDVFSFUUFYA

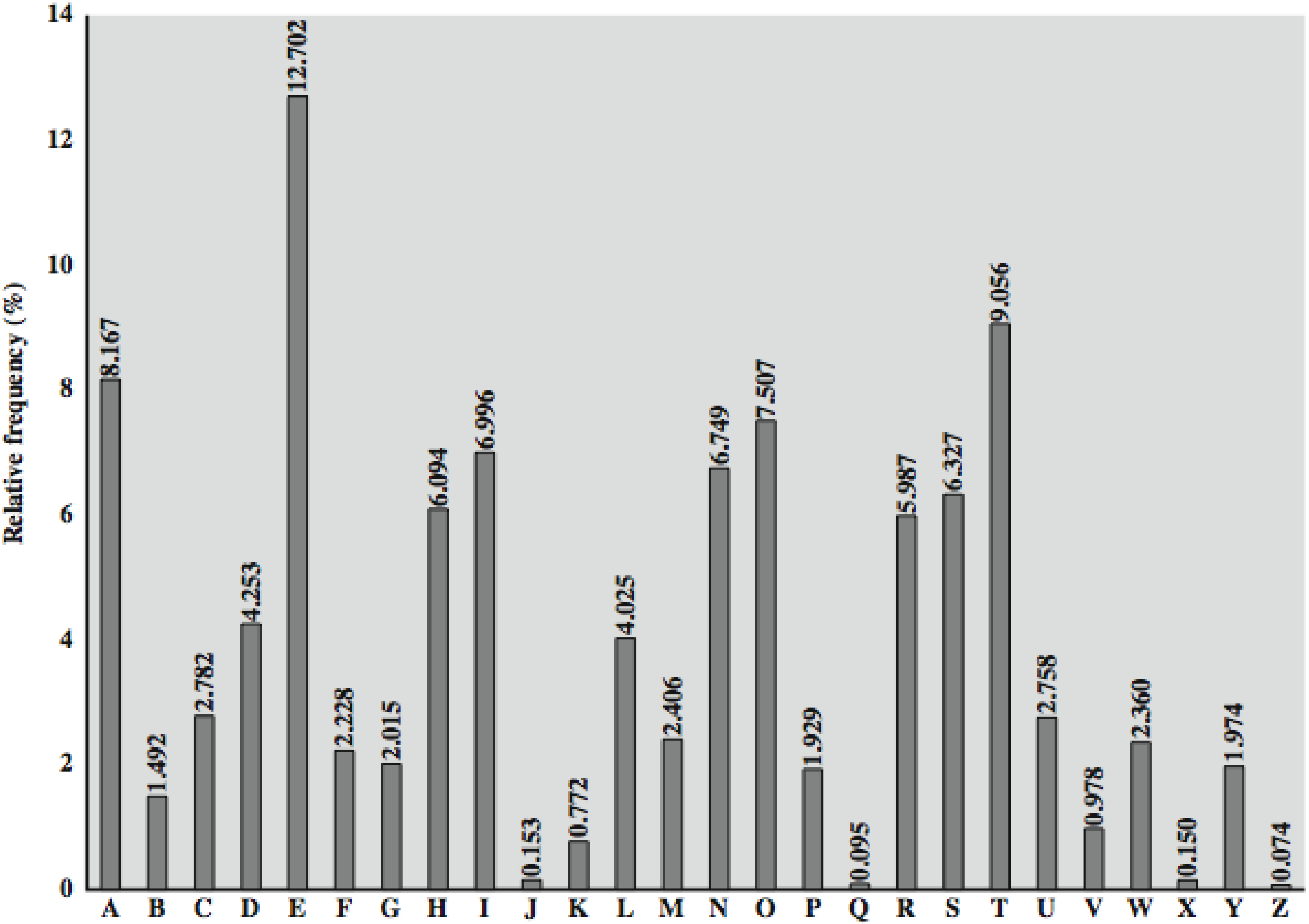
Monoalphabetic Cipher Security

- now have a total of 26! = 4 x 10^{26} keys
- with so many keys, might think is secure
- but would be **!!!WRONG!!!**
- problem is language characteristics

Language Redundancy and Cryptanalysis

- human languages are **redundant**
- eg "th lrd s m shphrd shll nt wnt"
- letters are not equally commonly used
- in English E is by far the most common letter
  - followed by T,R,N,I,O,A,S
- other letters like Z,J,K,Q,X are fairly rare
- have tables of single, double & triple letter frequencies for various languages

English Letter Frequencies



Playfair Cipher

- not even the large number of keys in a monoalphabetic cipher provides security
- one approach to improving security was to encrypt multiple letters
- the **Playfair Cipher** is an example
- invented by Charles Wheatstone in 1854, but named after his friend Baron Playfair

Playfair Key Matrix

- a 5X5 matrix of letters based on a keyword
- fill in letters of keyword (sans duplicates)
- fill rest of matrix with other letters
- eg. using the keyword MONARCHY

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Polyalphabetic Ciphers

- **polyalphabetic substitution ciphers**
- improve security using multiple cipher alphabets
- make cryptanalysis harder with more alphabets to guess and flatter frequency distribution
- use a key to select which alphabet is used for each letter of the message
- use each alphabet in turn
- repeat from start after end of key is reached

Vigenère Cipher

- simplest polyalphabetic substitution cipher
- effectively multiple caesar ciphers
- key is multiple letters long $K = k_1\ k_2\ ...\ k_d$
- i^{th} letter specifies i^{th} alphabet to use
- use each alphabet in turn
- repeat from start after d letters in message
- decryption simply works in reverse

Example of Vigenère Cipher

- write the plaintext out
- write the keyword repeated above it
- use each key letter as a caesar cipher key
- encrypt the corresponding plaintext letter
- eg using keyword *deceptive*

key:deceptivedeceptivedeceptive

plaintext: wearediscoveredsaveyourself

ciphertext:ZICVTWQNGRZGVTWAVZHCQYGL MGJ

Vernam Cipher

- ultimate defense is to use a key as long as the plaintext
- with no statistical relationship to it
- invented by AT&T engineer Gilbert Vernam in 1918
- originally proposed using a very long but eventually repeating key

One-Time Pad

- if a truly random key as long as the message is used, the cipher will be secure
- called a One-Time pad
- is unbreakable since ciphertext bears no statistical relationship to the plaintext
- since for **any plaintext** & **any ciphertext** there exists a key mapping one to other
- can only use the key **once** though
- problems in generation & safe distribution of key

Transposition Ciphers

- now consider classical **transposition** or **permutation** ciphers
- these hide the message by rearranging the letter order
- without altering the actual letters used
- can recognise these since have the same frequency distribution as the original text

Rail Fence cipher

- write message letters out diagonally over a number of rows
- then read off cipher row by row
- eg. write message out as:

  m e m a t r h t g p r y

  e t e f e t e o a a t
- giving ciphertext

  MEMATRHTGPRYETEFETEOAAT

Row Transposition Ciphers

- is a more complex transposition
- write letters of message out in rows over a specified number of columns
- then reorder the columns according to some key before reading off the rows

Key: 4312567

Column Out 3 4 2 1 5 6 7

Plaintext: a t t a c k p

o s t p o n e

d u n t i l t

w o a m x y z

Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

Private-Key/Symmetric key Cryptography

- traditional **private/secret/single key** cryptography uses **one** key
- shared by both sender and receiver
- if this key is disclosed communications are compromised
- also is **symmetric**, parties are equal
- hence does not protect sender from receiver forging a message & claiming is sent by sender

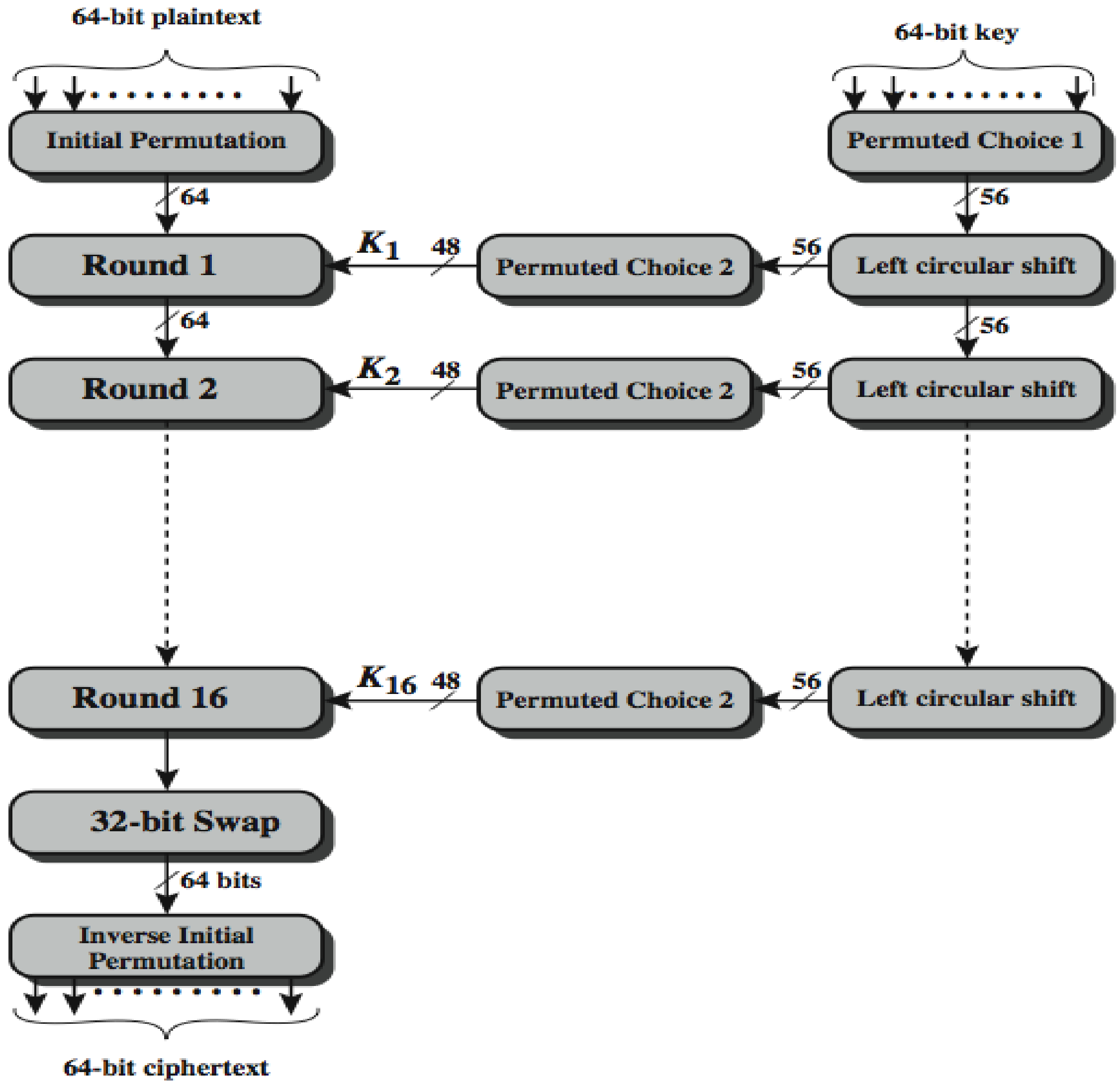
Symmetric Encryption Algorithms

- The most widely used symmetric encryption method in the United States is the block ciphers Triple Data Encryption Standard (3DES).
- Triple DES developed from the original and now cracked DES uses a 64-bit key consisting of 56 effective key bits and 8 parity bits.
- Others include:
  - National Institute of Standards and Technology (NIST) Advanced Encryption Standard (AES), which is expected to replace DES. AES is Advanced Encryption Standard.
  - The group: IDEA (International Data Encryption Algorithm), Blowfish, Rivest Cipher 4 (RC4), RC5, and CAST-128. See Table 10.2 for symmetric key algorithms.

Data Encryption Standard (DES)

- most widely used block cipher in world
- adopted in 1977 by NBS (now NIST)
  - as FIPS PUB 46
- encrypts 64-bit data using 56-bit key
- has widespread use
- has been considerable controversy over its security

DES Encryption Overview



Initial Permutation IP

- first step of the data computation
- IP reorders the input data bits
- even bits to LH half, odd bits to RH half
- quite regular in structure (easy in h/w)
- example:

IP(675a6967 5e5a6b5a) = (ffb2194d 004df6fb)

DES Round Structure

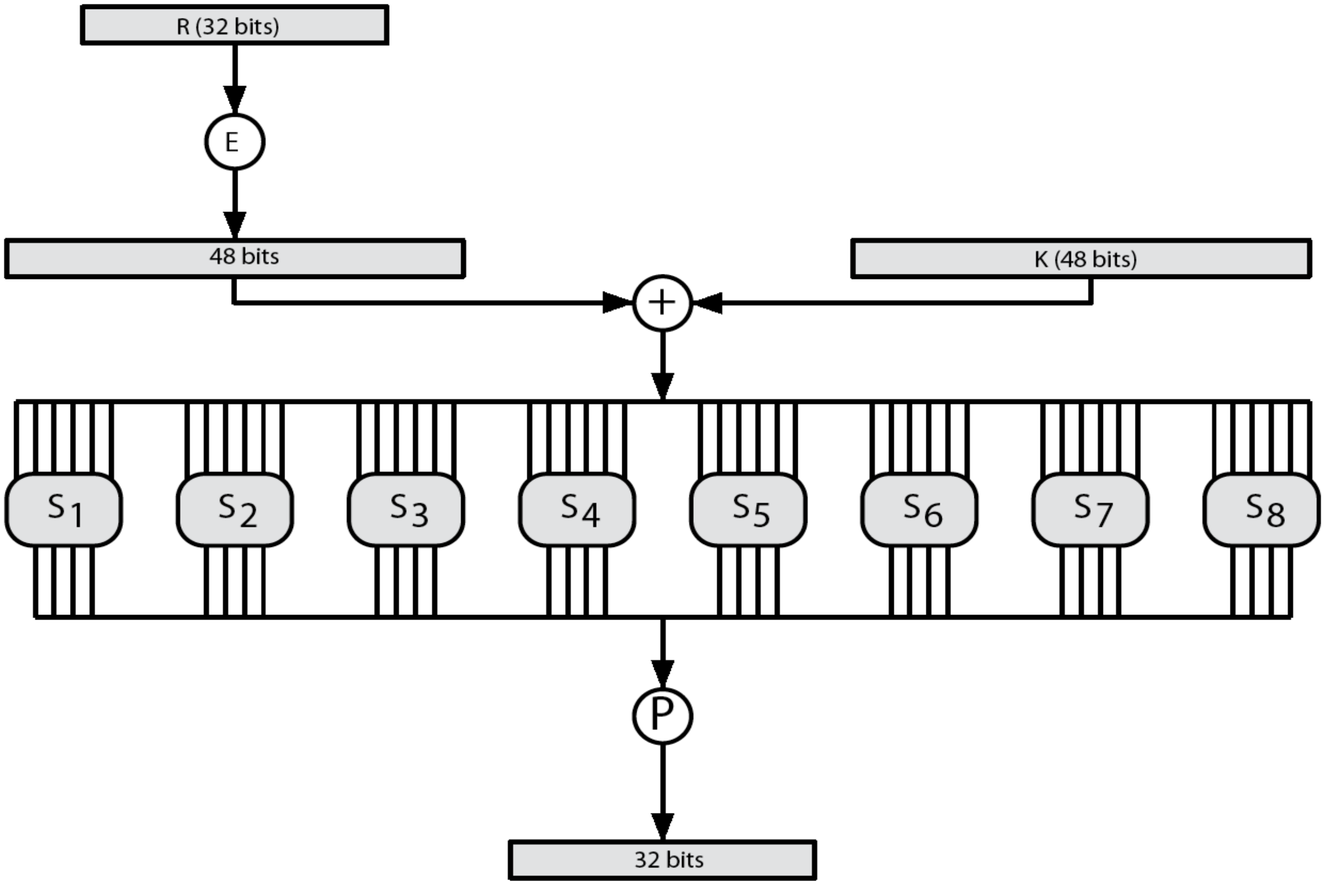
- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:

  $L_i = R_{i-1}$

  $R_i = L_{i-1} \oplus \mathrm{F}(R_{i-1}, K_i)$

- F takes 32-bit R half and 48-bit subkey:
  - expands R to 48-bits using perm E
  - adds to subkey using XOR
  - passes through 8 S-boxes to get 32-bit result
  - finally permutes using 32-bit perm P

DES Round Structure



DES Key Schedule

- forms subkeys used in each round
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule** K
    - selecting 24-bits from each half & permuting them by PC2 for use in round function F
- note practical use issues in h/w vs s/w

DES Decryption

- decrypt must unwind steps of data computation
- with Feistel design, do encryption steps again using subkeys in reverse order (SK16 … SK1)
  - IP undoes final FP step of encryption
  - 1st round with SK16 undoes 16th encrypt round

    ….
  - 16th round with SK1 undoes 1st encrypt round
  - then final FP undoes initial encryption IP
  - thus recovering original data value

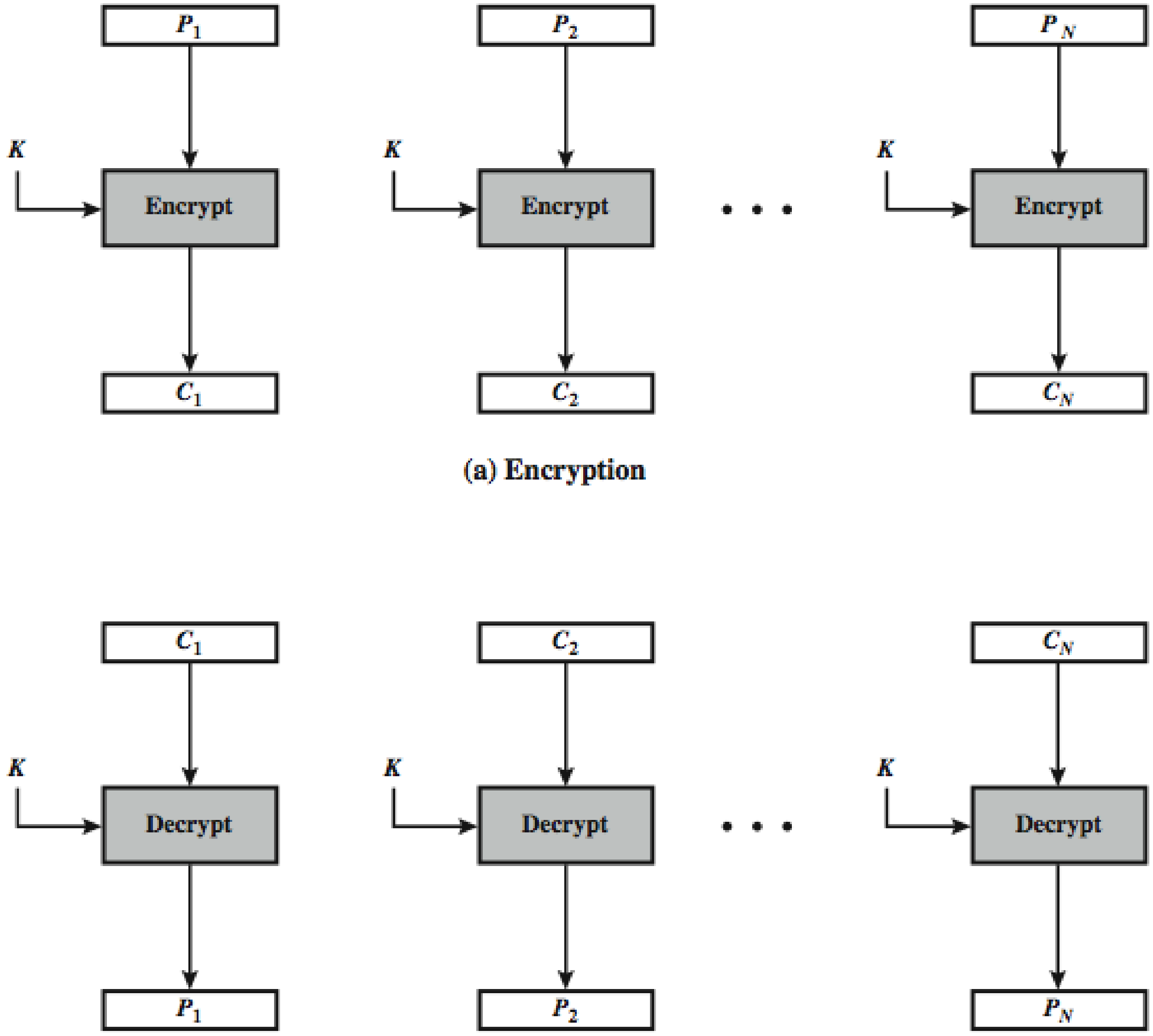
Problems with Symmetric Encryption

- Symmetric encryption, although fast, suffers from several problems in the modern digital communication environment including:
  - Since single key, in a distributed environment with large numbers of combination pairs, it is difficult for the one recipient to keep so many keys in order to support all communication.
  - The size of the communication space presents problems. Because of the massive potential number of individuals who can carry on communication in a many-to-one, one-to-many, and many-to-many topologies supported by the Internet for example, the secret-key cryptography, if strictly used, requires billions of secret keys pairs to be created, shared, and stored.

Modes of Operation

- block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit blocks with 56-bit key
- need some way to en/decrypt arbitrary amounts of data in practise
- NIST SP 800-38A defines 5 modes
- have **block** and **stream** modes
- to cover a wide variety of applications
- can be used with any block cipher

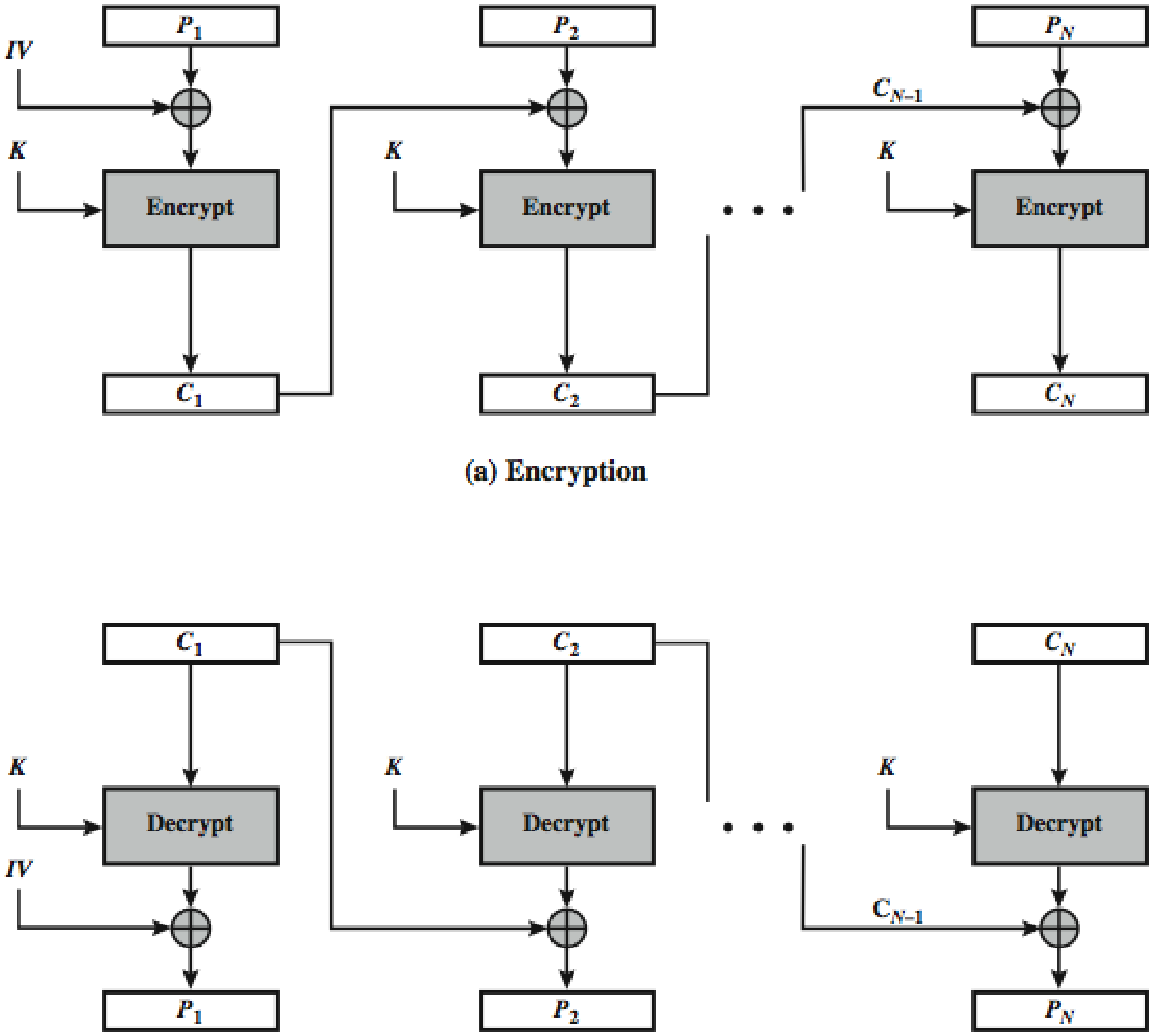
Electronic Codebook Book (ECB)



(a) Encryption

(b) Decryption

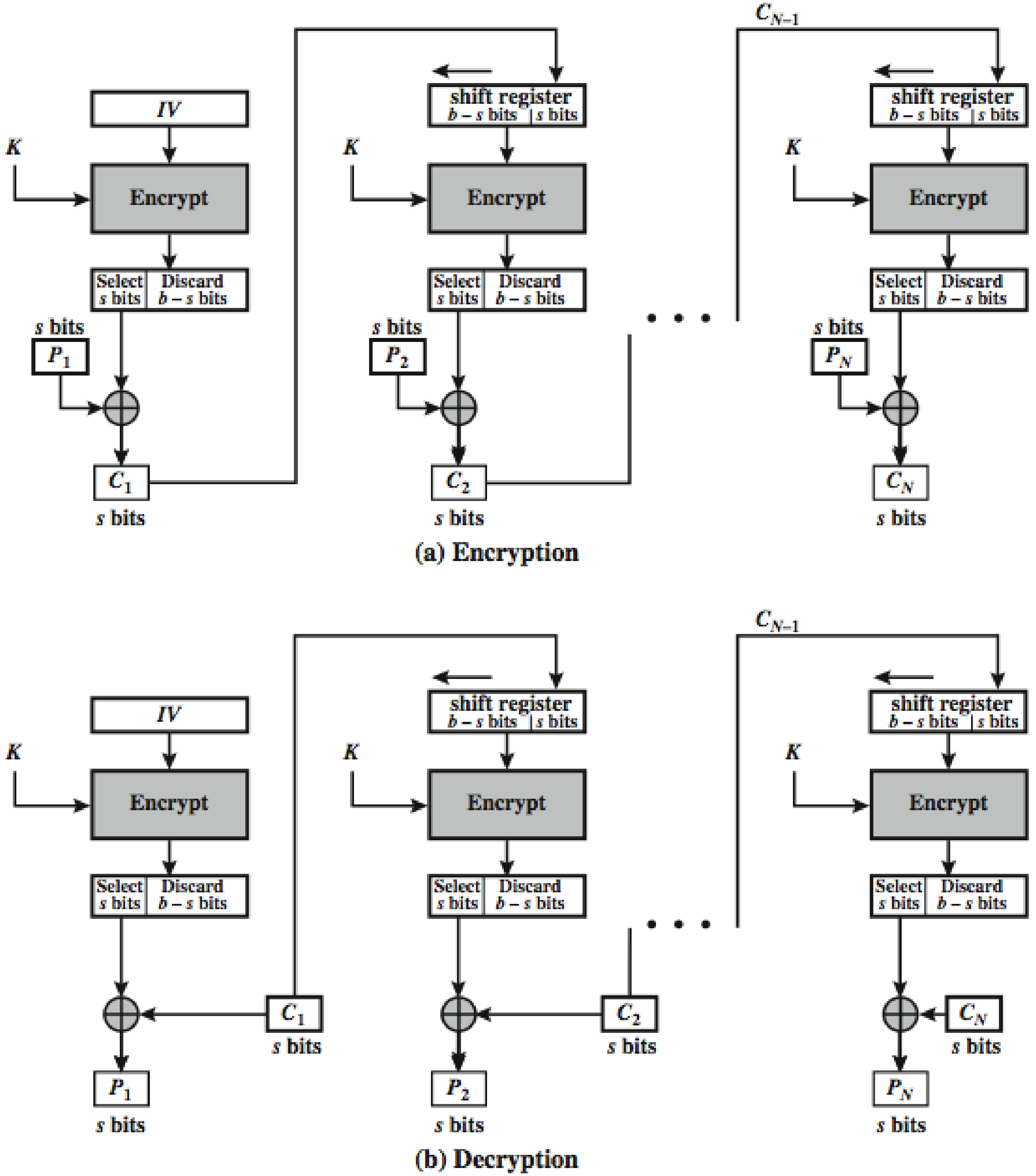
Cipher Block Chaining (CBC)



(a) Encryption

(b) Decryption

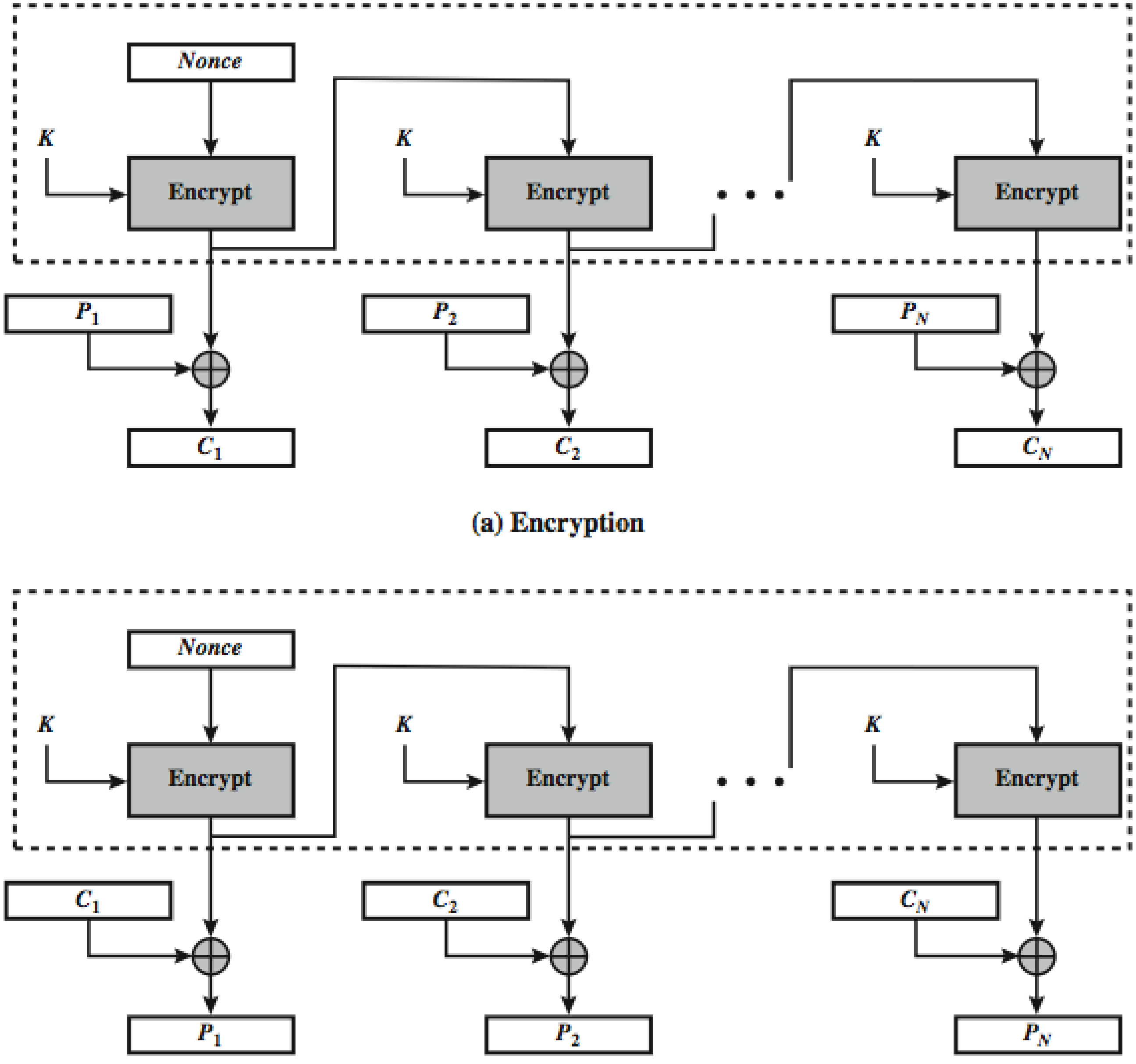
s-bit Cipher FeedBack (CFB-s)



(a) Encryption

(b) Decryption

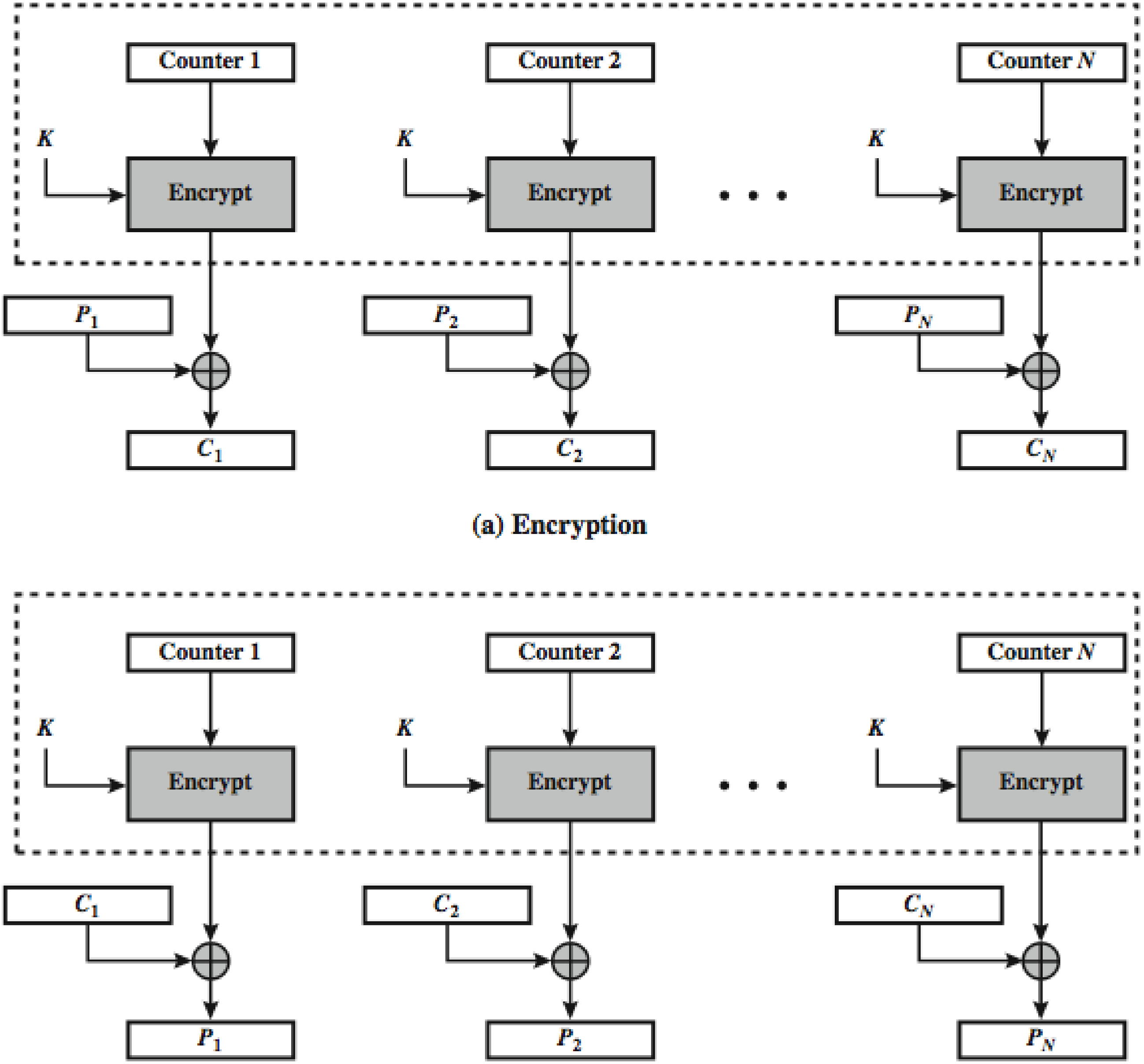
Output FeedBack (OFB)



(a) Encryption

(b) Decryption

Counter (CTR)



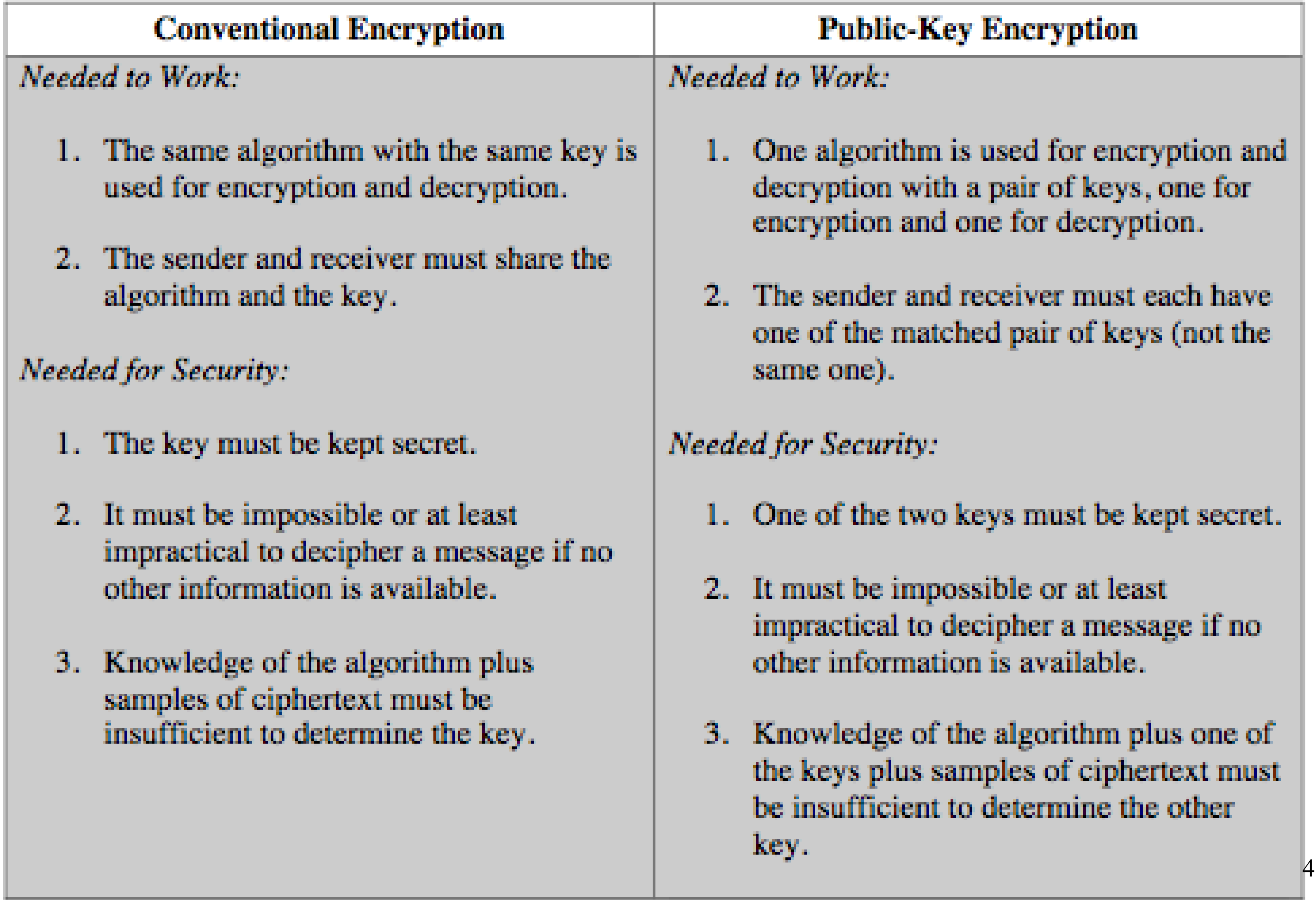
(a) Encryption

(b) Decryption

Public-Key Cryptography

- probably most significant advance in the 3000 year history of cryptography
- uses **two** keys – a public & a private key
- **asymmetric** since parties are **not** equal
- uses clever application of number theoretic concepts to function
- complements **rather than** replaces private key crypto

Symmetric vs Public-Key



Conventional Encryption	Public-Key Encryption
Needed to Work: 1. The same algorithm with the same key is used for encryption and decryption. 2. The sender and receiver must share the algorithm and the key. *Needed for Security:* 1. The key must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	*Needed to Work:* 1. One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption. 2. The sender and receiver must each have one of the matched pair of keys (not the same one). *Needed for Security:* 1. One of the two keys must be kept secret. 2. It must be impossible or at least impractical to decipher a message if no other information is available. 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

Public Key Encryption Algorithms

- Various algorithms exist for public key encryption including RSA, DSA, PGP, and El Gamal.
  - Rivest, Shamir & Adleman (RSA)
  - Digital Signature Algorithm (DSA)
  - Pretty Good Privacy (PGP)

RSA

- by **R**ivest, **S**hamir & **A**dleman of MIT in 1977
- best known & widely used public-key scheme
- based on exponentiation in a finite (Galois) field over integers modulo a prime
  - nb. exponentiation takes $O((\log n)^3)$ operations (easy)
- uses large integers (eg. 1024 bits)
- security due to cost of factoring large numbers
  - nb. factorization takes $O(e^{\log n \log \log n})$ operations (hard)

RSA En/decryption

- to encrypt a message M the sender:
  - obtains **public key** of recipient PU={e,n}
  - computes: $C = M^e \bmod n$, where $0 \leq M < n$
- to decrypt the ciphertext C the owner:
  - uses their private key PR={d,n}
  - computes: $M = C^d \bmod n$
- note that the message M must be smaller than the modulus n (block if needed)

RSA Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random: p, q
- computing their system modulus $n=p.q$
  - note $ø(n)=(p-1)(q-1)$
- selecting at random the encryption key e
  - where $1<e<ø(n)$, $\gcd(e,ø(n))=1$
- solve following equation to find decryption key d
  - $e.d=1 \bmod ø(n)$ and $0\leq d\leq n$
- publish their public encryption key: $PU=\{e,n\}$
- keep secret private decryption key: $PR=\{d,n\}$

RSA Works

- because of Euler's Theorem:
  - $a^{ø(n)} \bmod n = 1$ where $\gcd(a,n)=1$
- in RSA have:
  - $n=p.q$
  - $ø(n)=(p-1)(q-1)$
  - carefully chose e & d to be inverses mod ø(n)
  - hence $e.d=1+k.ø(n)$ for some k
- hence :

$$C^d = M^{e.d} = M^{1+k.ø(n)} = M^1.(M^{ø(n)})^k$$

$$= M^1.(1)^k = M^1 = M \bmod n$$

Digital Signatures

- have looked at message authentication
  - but does not address issues of lack of trust
- digital signatures provide the ability to:
  - verify author, date & time of signature
  - authenticate message contents at the time of signature
  - Must be verifiable by third parties to resolve disputes

Digital Signature Properties

- must depend on the message signed
- must use information unique to sender
    - to prevent both forgery and denial
- must be relatively easy to produce
- must be relatively easy to recognize & verify
- be computationally infeasible to forge
    - with new message for existing digital signature
    - with fraudulent digital signature for given message
- be practical save digital signature in a storage

Direct Digital Signatures

- involves only the parties: sender and receiver
- assumed receiver has sender's public-key
- digital signature made by sender signing entire message or hash with private-key
- can encrypt using receivers public-key
- important that sign first then encrypt message & signature
- security depends on sender's private-key

Arbitrated Digital Signatures

- involves use of arbiter A
  - Sender sends the signed message to arbiter
  - validates any signed message
  - then dated and sent to recipient
- requires suitable level of trust in arbiter
- can be implemented with either private or public-key algorithms
- arbiter may or may not be able to see message

Management of Public Keys

- *Definition*: Key management is the set of techniques and procedures supporting the establishment and maintenance of keying relationships between authorized parties
- The objective of key management is to maintain keying relationships and keying material in a manner that counters relevant threats

Key Management

- Key management encompasses techniques and procedures supporting:
    1. initialization of systems users within a domain;
    2. generation, distribution, and installation of keying material
    3. controlling the use of keying material;
    4. update, revocation, and destruction of keying material
    5. storage, backup/recovery, and archival of keying material.

Key Management (public)

- public-key encryption helps address key distribution problems
- have two aspects of this:
  - distribution of public keys
  - use of public-key encryption to distribute secret keys

Distribution of Public Keys

- can be considered as using one of:
  - Public announcement
  - Publicly available directory
  - Public-key authority
  - Public-key certificates

Distribution of Secret Keys using Public Key

- public-key cryptography can be used for secrecy or authentication
  - but public-key algorithms are slow
- We want to use symmetric key encryption algorithm encrypt bulk message
  - Because symmetric key encryption algorithms are hundreds of times faster than public key encryption algorithms
- So two communicating parties usually
  1. negotiate a symmetric key (called session key) with the help of public key algorithms
  2. Then use the session key to encrypt messages
  3. For each new session (e.g. login your online banking service again after closing the web browser), a new session key will be established

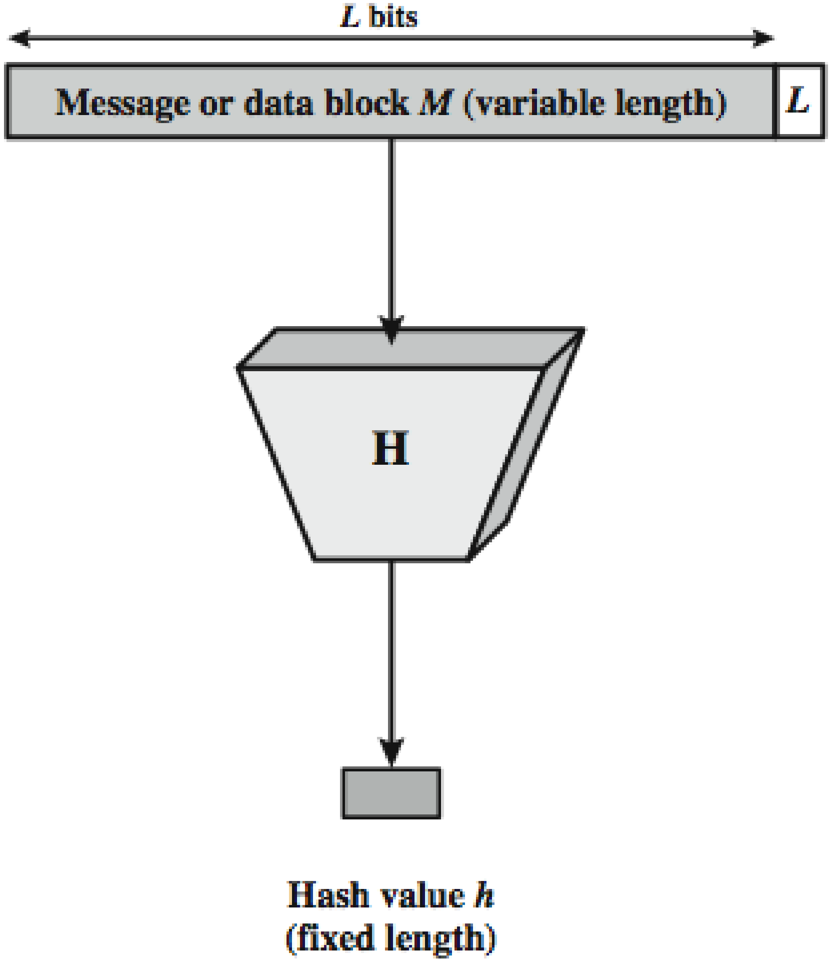
Hash Functions

- condenses arbitrary message to fixed size

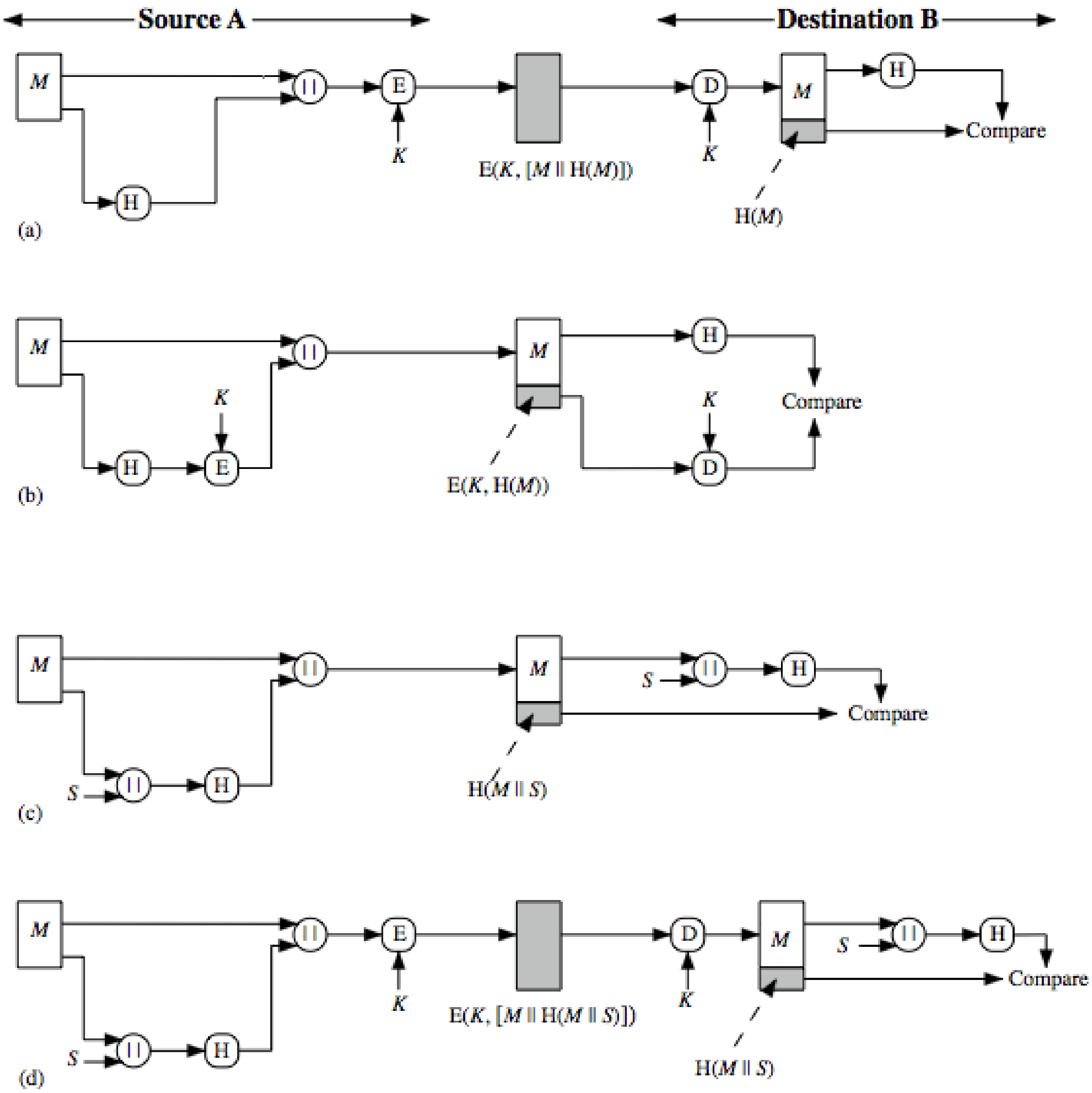
  h = H(M)

- usually assume hash function is public
- hash used to detect changes to message
- want a cryptographic hash function
  - computationally infeasible to find data mapping to specific hash (one-way property)
  - computationally infeasible to find two data to same hash (collision-free property)

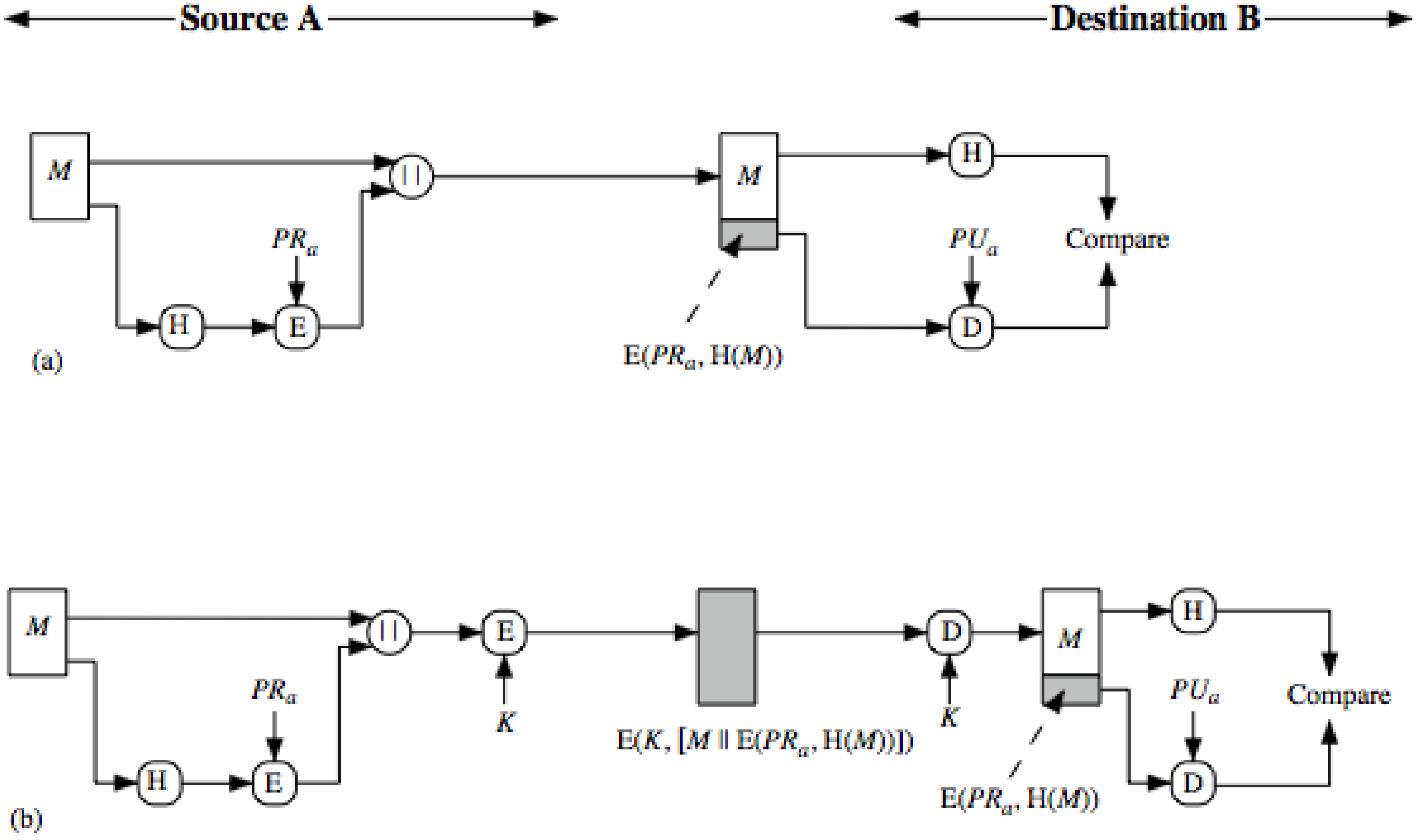
Cryptographic Hash Function



Hash Functions & Message Authent-ication



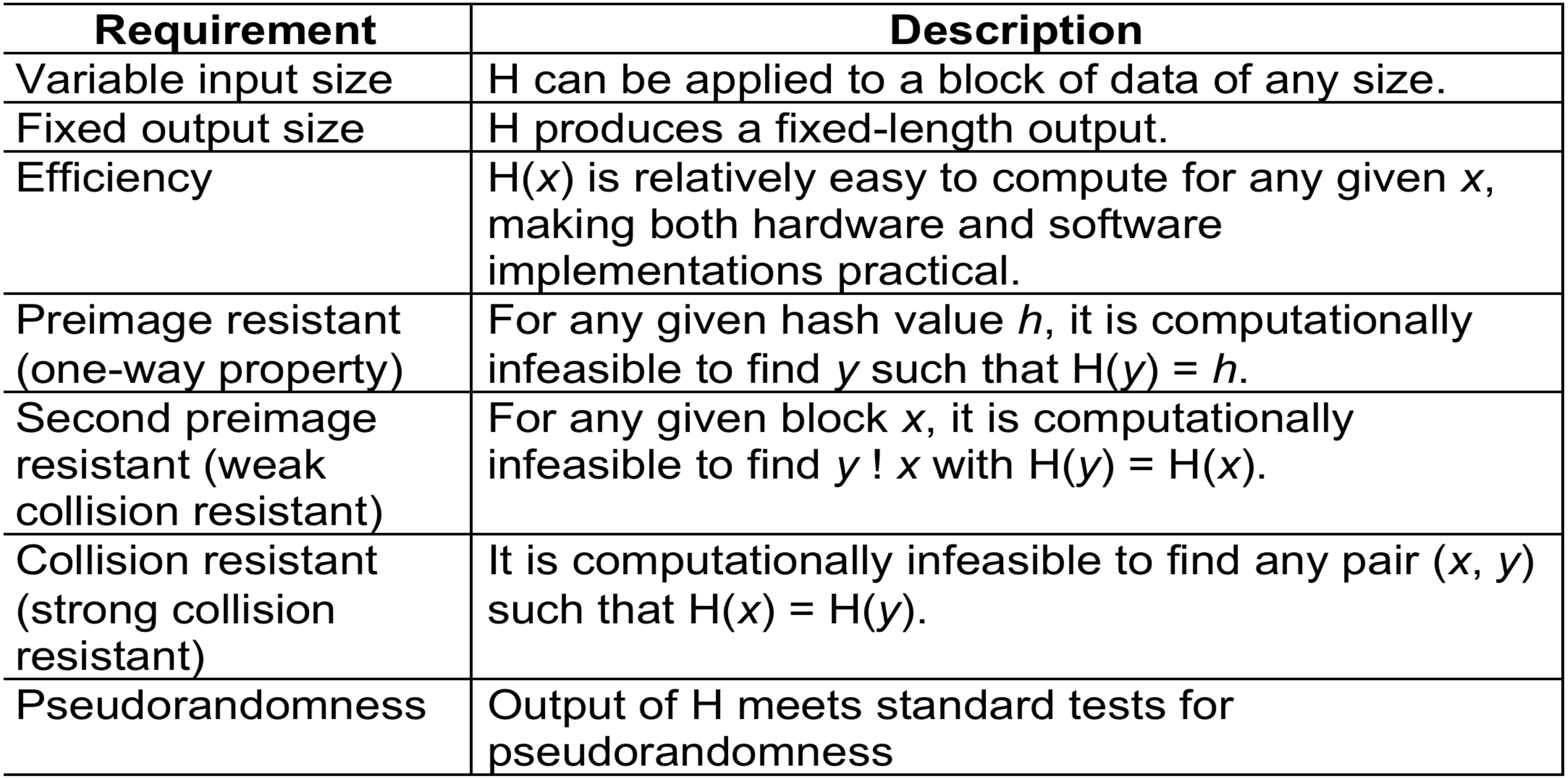
Hash Functions & Digital Signatures



Other Hash Function Uses

- to create a one-way password file
  - store hash of password not actual password
- for intrusion detection and virus detection
  - keep & check hash of files on system
- pseudorandom function (PRF) or pseudorandom number generator (PRNG)

Hash Function Requirements



Requirement	Description
Variable input size	H can be applied to a block of data of any size.
Fixed output size	H produces a fixed-length output.
Efficiency	H(*x*) is relatively easy to compute for any given *x*, making both hardware and software implementations practical.
Preimage resistant (one-way property)	For any given hash value *h*, it is computationally infeasible to find *y* such that H(*y*) = *h*.
Second preimage resistant (weak collision resistant)	For any given block *x*, it is computationally infeasible to find *y* ! *x* with H(*y*) = H(*x*).
Collision resistant (strong collision resistant)	It is computationally infeasible to find any pair (*x*, *y*) such that H(*x*) = H(*y*).
Pseudorandomness	Output of H meets standard tests for pseudorandomness

Communication Security

- IP Security
  - Transport mode
  - Tunnel mode
- Firewall
  - Packet filtering
  - Gateway

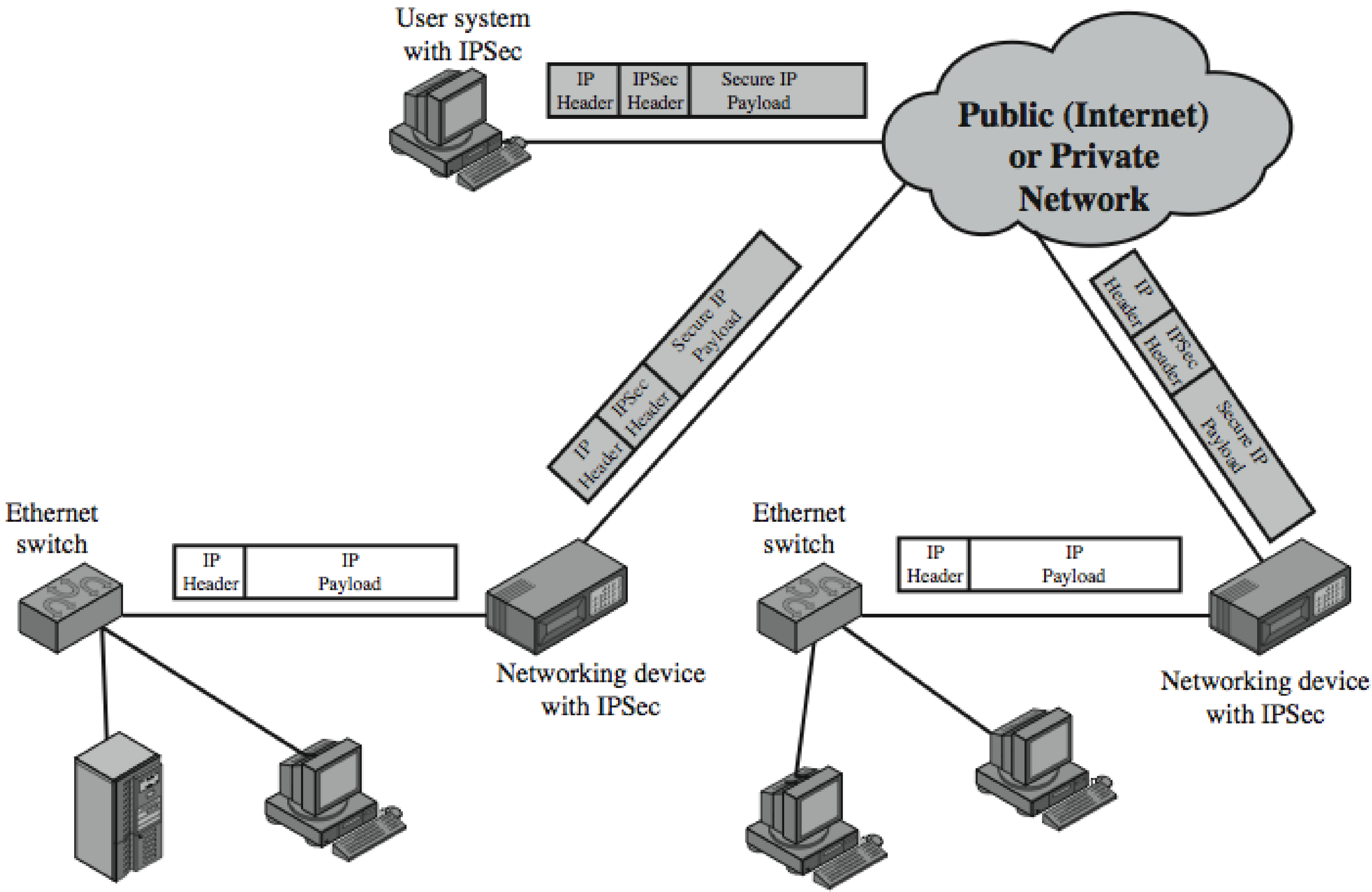
IP Security

- have a range of application specific security mechanisms
  - eg. S/MIME, PGP, Kerberos, SSL/HTTPS
- however there are security concerns that cut across protocol layers
- would like security implemented by the network for all applications

IP Security

- general IP Security mechanisms
- provides
  - authentication
  - confidentiality
  - key management
- applicable to use over LANs, across public & private WANs, & for the Internet
- need identified in 1994 report
  - need authentication, encryption in IPv4 & IPv6

IP Security Uses



Benefits of IPSec

- in a firewall/router provides strong security to all traffic crossing the perimeter
- in a firewall/router is resistant to bypass
- is below transport layer, hence transparent to applications
- can be transparent to end users
- can provide security for individual users
- secures routing architecture

IP Security Architecture

- specification is quite complex, with groups:
  - Architecture
    - RFC4301 *Security Architecture for Internet Protocol*
  - Authentication Header (AH)
    - RFC4302 *IP Authentication Header*
  - Encapsulating Security Payload (ESP)
    - RFC4303 *IP Encapsulating Security Payload (ESP)*
  - Internet Key Exchange (IKE)
    - RFC4306 *Internet Key Exchange (IKEv2) Protocol*
  - Cryptographic algorithms
  - Other

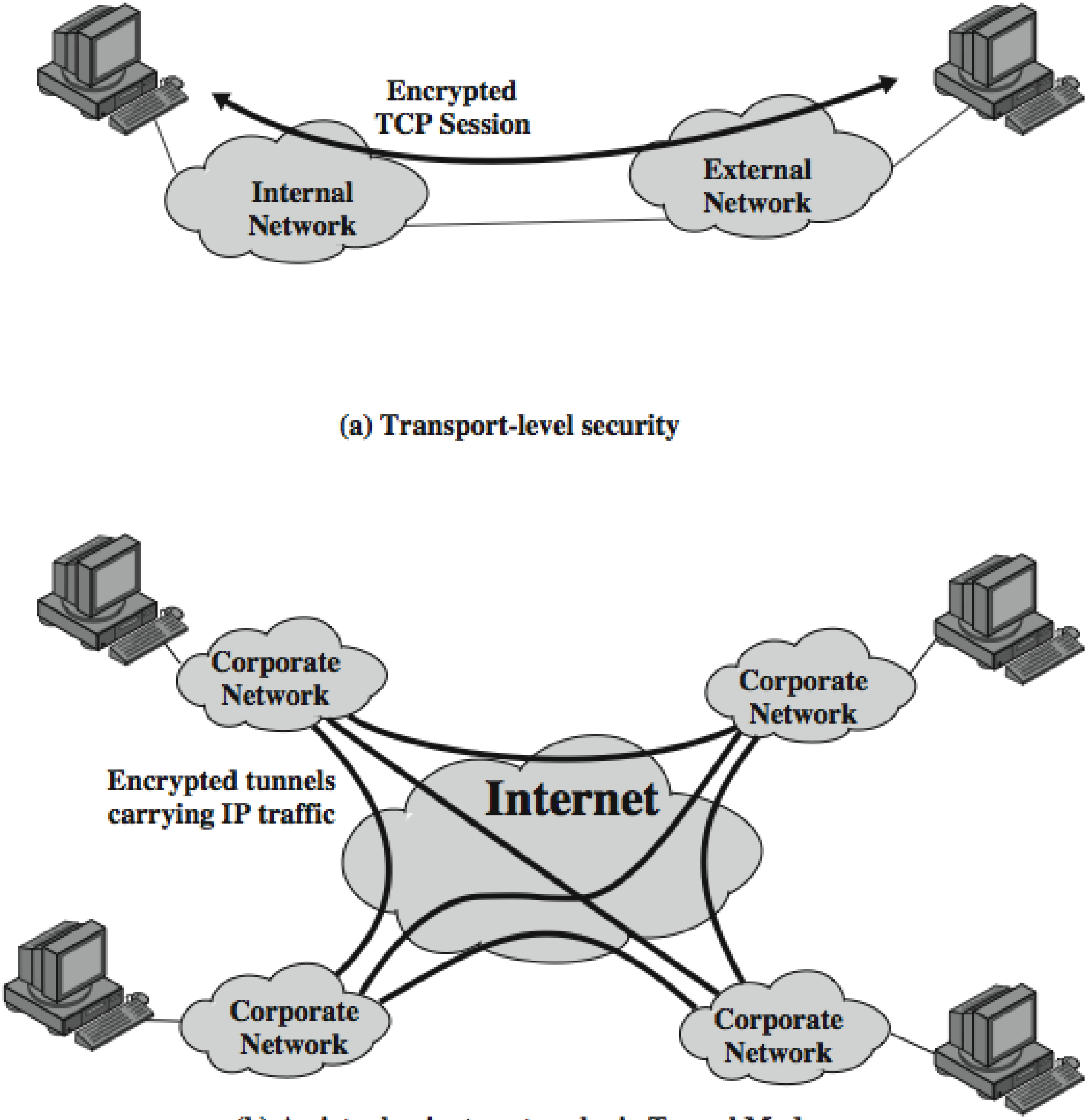
IPSec Services

- Access control
- Connectionless integrity
- Data origin authentication
- Rejection of replayed packets
  - a form of partial sequence integrity
- Confidentiality (encryption)
- Limited traffic flow confidentiality

Transport and Tunnel Modes

- Transport Mode
  - to encrypt & optionally authenticate IP data
  - can do traffic analysis but is efficient
  - good for ESP host to host traffic
- Tunnel Mode
  - encrypts entire IP packet
  - add new header for next hop
  - no routers on way can examine inner IP header
  - good for VPNs, gateway to gateway security

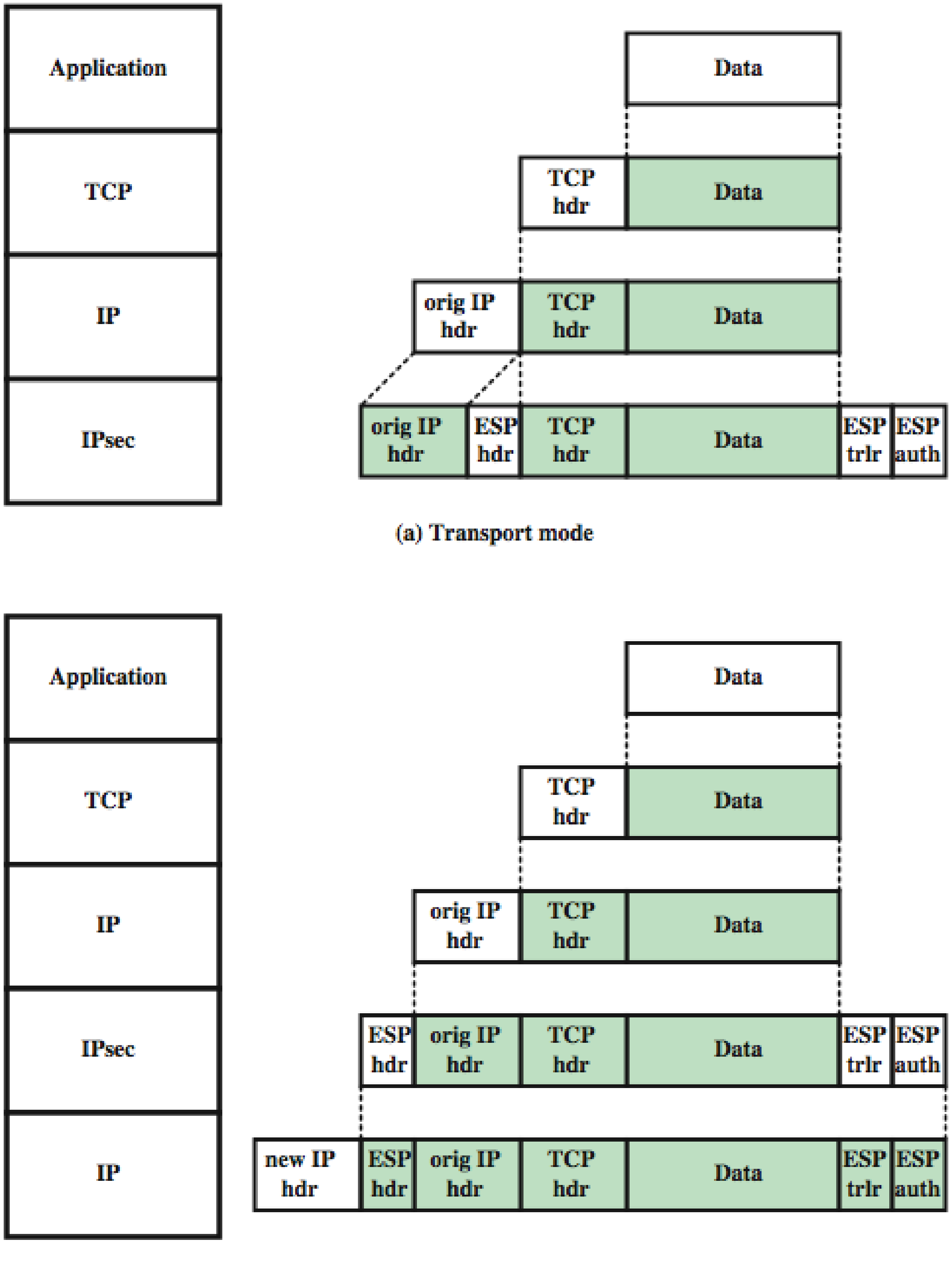
Transport and Tunnel Modes



(a) Transport-level security

(b) A virtual private network via Tunnel Mode

Transport and Tunnel Mode Protocols



(a) Transport mode

(b) Tunnel mode

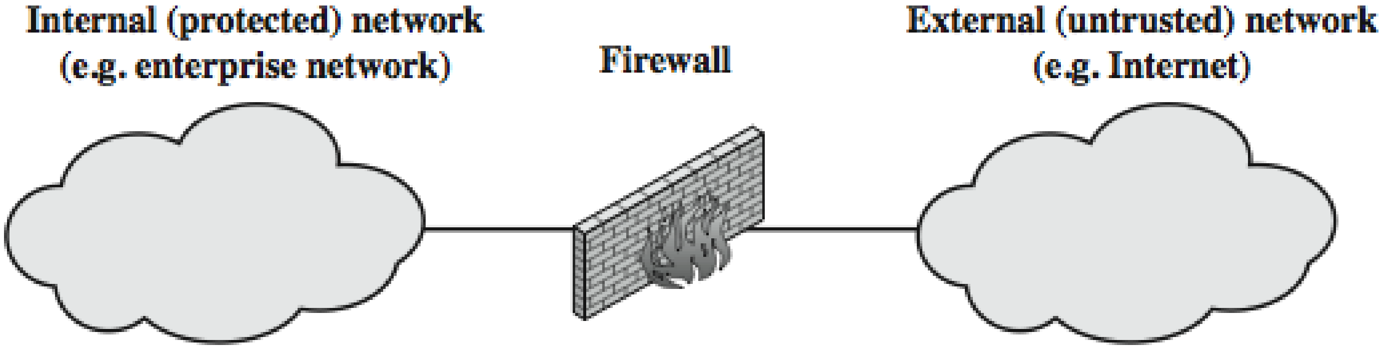
Firewall

- seen evolution of information systems
- now everyone want to be on the Internet
- and to interconnect networks
- has persistent security concerns
  - can't easily secure every system in org
- typically use a **Firewall**
- to provide **perimeter defence**
- as part of comprehensive security strategy

What is a Firewall?

- a **choke point** of control and monitoring
- interconnects networks with differing trust
- imposes restrictions on network services
  - only authorized traffic is allowed
- auditing and controlling access
  - can implement alarms for abnormal behavior
- provide NAT & usage monitoring
- implement VPNs using IPSec
- must be immune to penetration

What is a Firewall?


Internal (protected) network
(e.g. enterprise network)
Firewall
External (untrusted) network
(e.g. Internet)

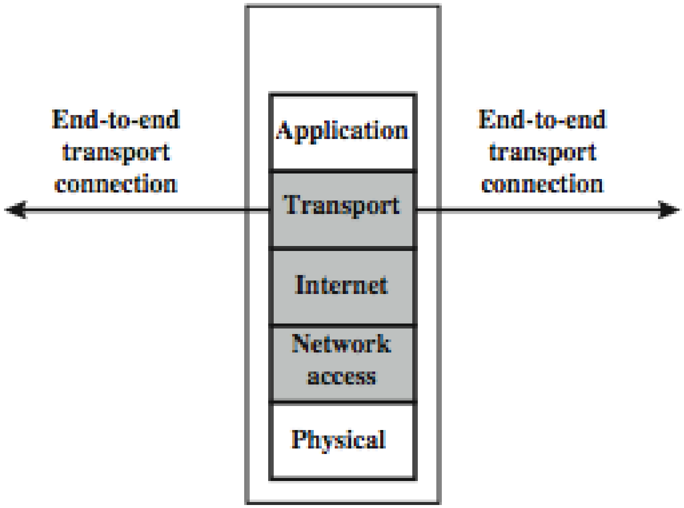
Firewall Limitations

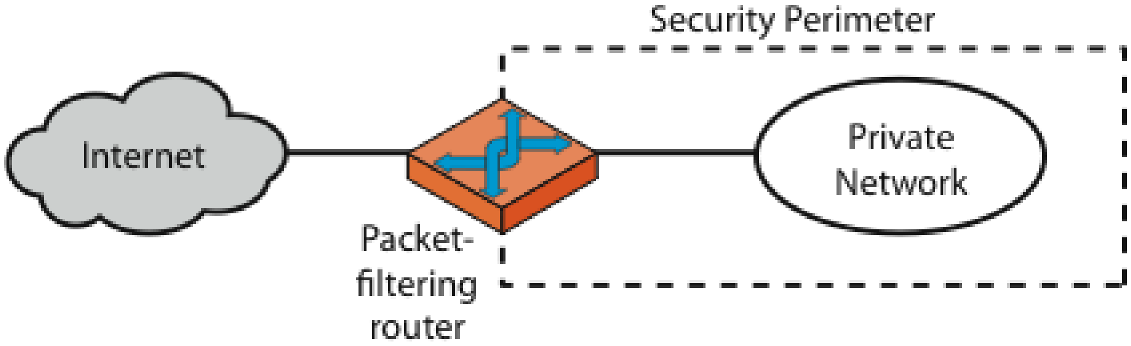
- cannot protect from attacks bypassing it
  - eg sneaker net, utility modems, trusted organisations, trusted services (eg SSL/SSH)
- cannot protect against internal threats
  - eg disgruntled or colluding employees
- cannot protect against access via WLAN
  - if improperly secured against external use
- cannot protect against malware imported via laptop, PDA, storage infected outside

Firewalls – Packet Filters

- simplest, fastest firewall component
- foundation of any firewall system
- examine each IP packet (no context) and permit or deny according to rules
- hence restrict access to services (ports)
- possible default policies
  - that not expressly permitted is prohibited
  - that not expressly prohibited is permitted

Firewalls – Packet Filters





(a) Packet-filtering router

Firewalls – Packet Filters

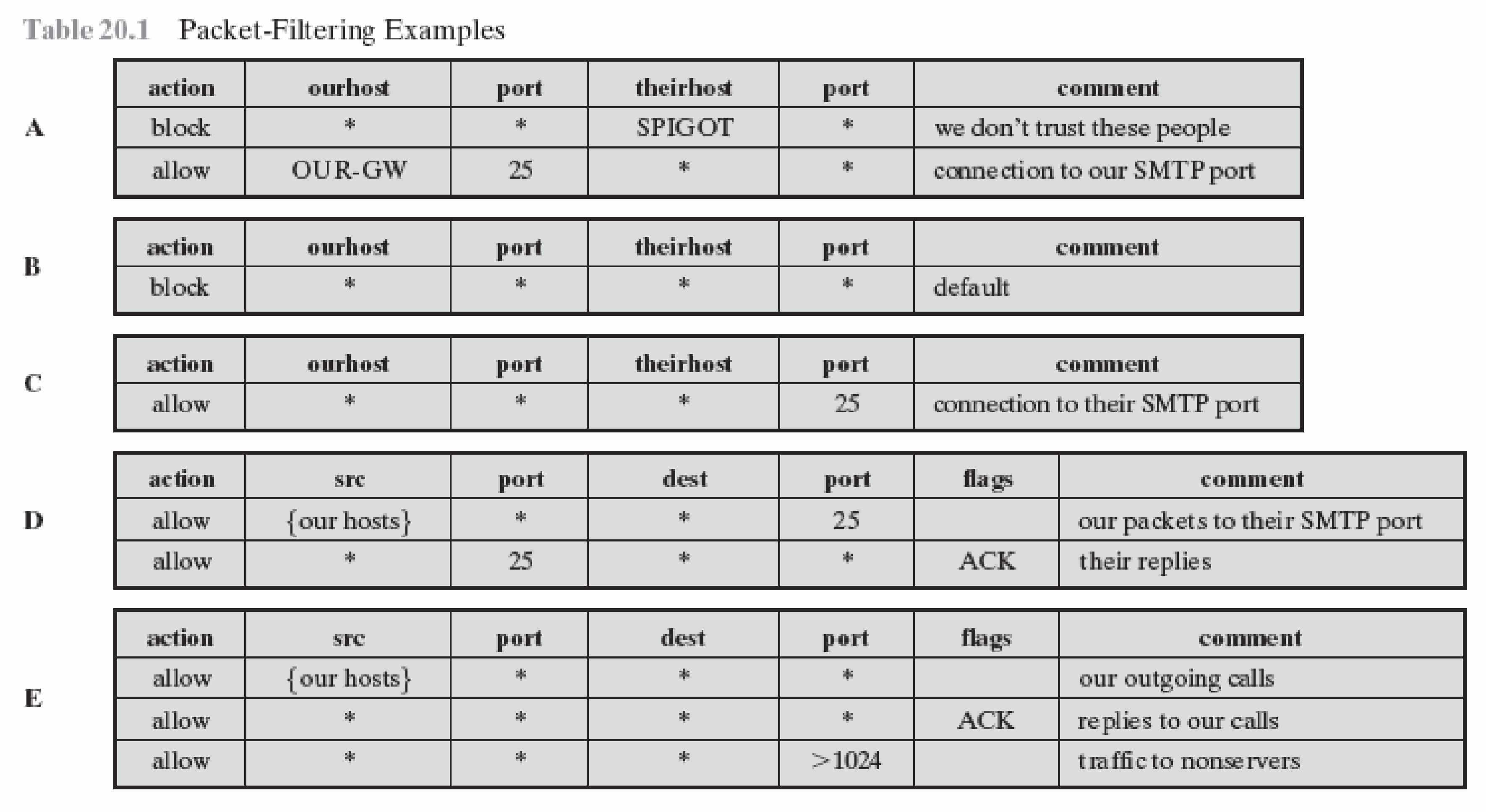


Table 20.1 Packet-Filtering Examples

	action	ourhost	port	theirhost	port	comment
A	block	*	*	SPIGOT	*	we don't trust these people
	allow	OUR-GW	25	*	*	connection to our SMTP port

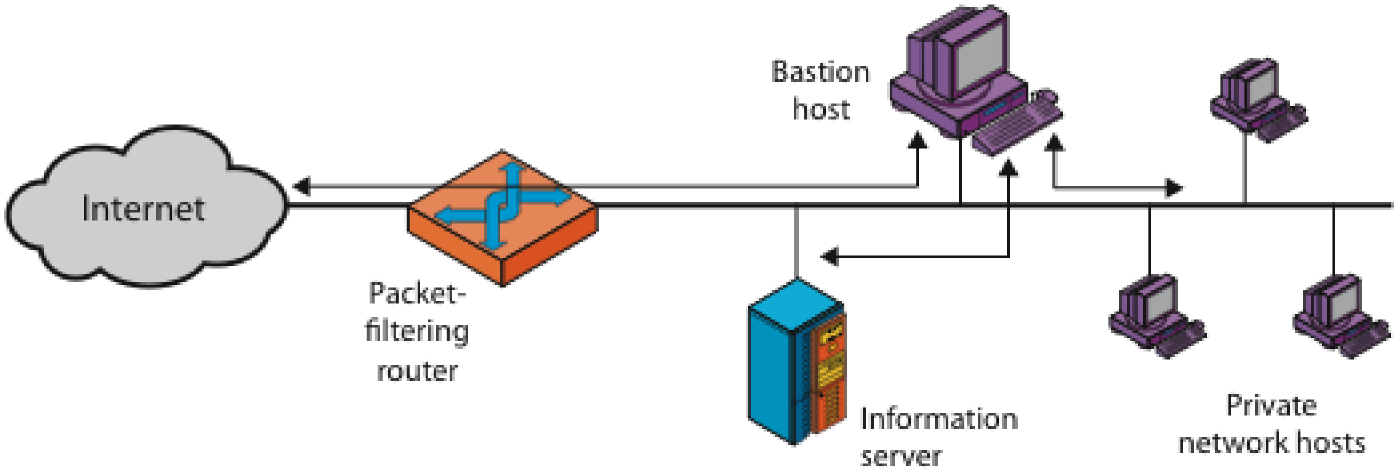
	action	ourhost	port	theirhost	port	comment
B	block	*	*	*	*	default

	action	ourhost	port	theirhost	port	comment
C	allow	*	*	*	25	connection to their SMTP port

	action	src	port	dest	port	flags	comment
D	allow	{our hosts}	*	*	25		our packets to their SMTP port
	allow	*	25	*	*	ACK	their replies

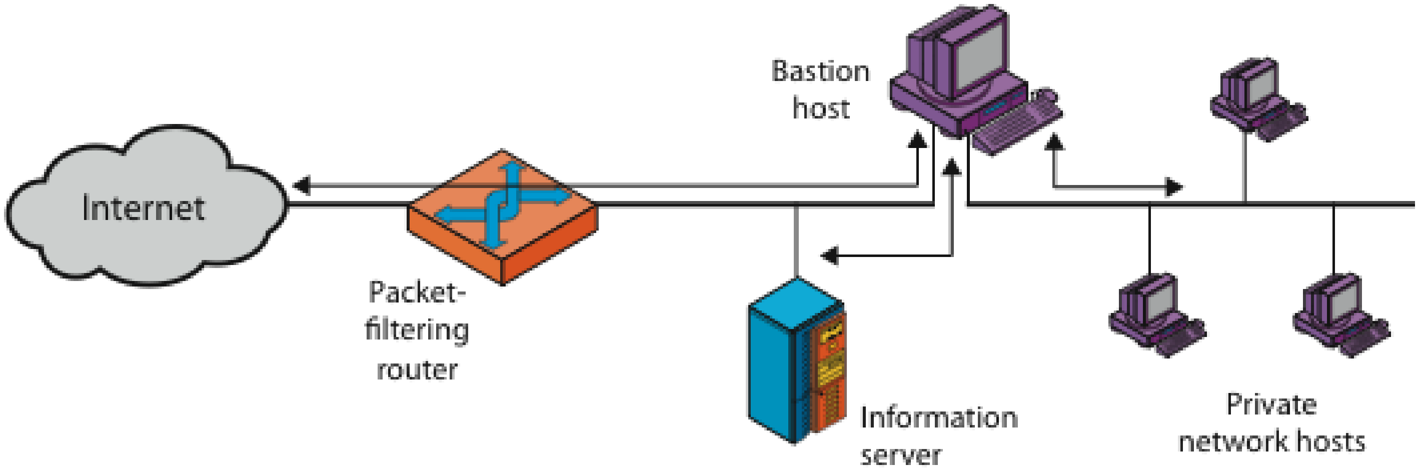
	action	src	port	dest	port	flags	comment
E	allow	{our hosts}	*	*	*		our outgoing calls
	allow	*	*	*	*	ACK	replies to our calls
	allow	*	*	*	>1024		traffic to nonservers

Firewall Configurations



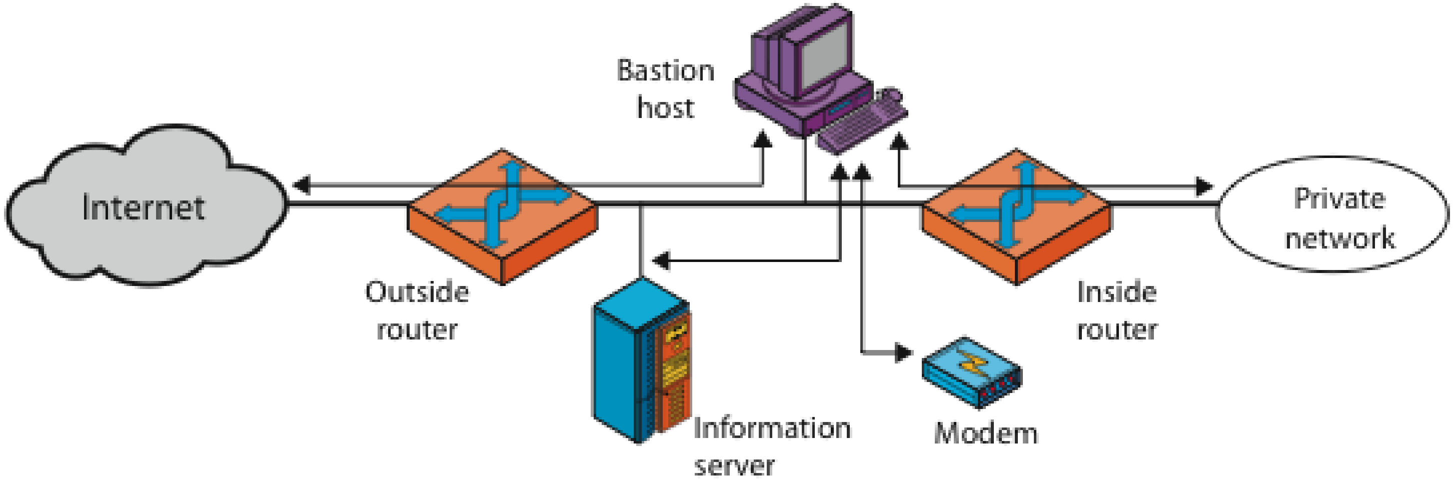
(a) Screened host firewall system (single-homed bastion host)

Firewall Configurations



(b) Screened host firewall system (dual-homed bastion host)

Firewall Configurations



(c) Screened-subnet firewall system

Authentication Protocols

- used to convince parties of each others identity and to exchange session keys
- may be one-way or mutual
- key issues in authenticated key exchange:
  - confidentiality – to protect session keys
  - timeliness – to prevent replay attacks
- published protocols are often found to have flaws and need to be modified

Replay Attacks

- where a valid signed message is copied and later resent
  - simple replay (simply copy and replay later)
  - repetition that can be logged (replay a timestamped message within its valid time window)
  - repetition that cannot be detected (the original message is suppressed and only replayed message arrives at the destination)
  - backward replay without modification (a message is replayed back to the sender; can work if symmetric encryption is used)

Using Symmetric Encryption

- use a two-level hierarchy of keys
- usually with a trusted Key Distribution Center (KDC)
  - each party shares own master key with KDC
  - KDC generates session keys used for connections between parties
  - master keys used to distribute these to them

Needham-Schroeder Protocol

- does key distribution using a KDC
- Also performs authentication
- for session between A and B mediated by KDC, protocol overview is:

1. A->KDC: $ID_A \parallel ID_B \parallel N_1$

2. KDC -> A: $\mathrm{E}_{\mathrm{Ka}}[\mathrm{Ks} \parallel ID_B \parallel N_1 \parallel \mathrm{E}_{Kb}[Ks \| ID_A]\ ]$

3. A -> B: $E_{Kb}[Ks \| ID_A]$

4. B -> A: $E_{Ks}[N_2]$

5. A -> B: $E_{Ks}[\mathrm{f}(N_2)]$

Needham-Schroeder Protocol

- used to securely distribute a new session key for communications between A & B
- vulnerable to a replay attack if an old session key has been compromised
  - then message 3 can be resent convincing B that is communicating with A
- modifications to address this require:
  - timestamps (Denning 81)
  - using an extra nonce (Neuman 93)

Using Public-Key Encryption

- have a range of approaches based on the use of public key encryption
- need to ensure have correct public keys for other parties
- using a central Authentication Server (AS)
- various protocols exist using timestamps or nonces

Denning AS Protocol

- Denning 81 presented the following:

  1. A -> AS: $ID_A \| ID_B$

  2. AS -> A: $E_{PRas}[ID_A \| PU_a \| T] \| E_{PRas}[ID_B \| PU_b \| T]$

  3. A -> B: $E_{PRas}[ID_A \| PU_a \| T] \| E_{PRas}[ID_B \| PU_b \| T] \| E_{PUb}[E_{PRas}[K_s \| T]]$

- note session key is chosen by A, hence AS need not be trusted to protect it
- timestamps prevent replay but requires synchronized clocks

One-Way Authentication

- required when sender & receiver are not in communications at same time (e.g., email)
- have header in clear so can be delivered by email system
- may want contents of body protected & sender authenticated

Using Symmetric Encryption

- One-way authentication protocol:

  1. A->KDC: $ID_A \| ID_B \| N_1$

  2. KDC -> A: $E_{Ka}[Ks \| ID_B \| N_1 \| E_{Kb}[Ks\|ID_A]]$

  3. A -> B: $E_{Kb}[Ks\|ID_A] \| E_{Ks}[M]$

- does not protect against replays
  - could rely on timestamp in message, though email delays make this problematic

Public-Key Approaches

- if confidentiality is a major concern, can use:

  A->B: $E_{PUb}[Ks] \parallel E_{Ks}[M]$

  - has encrypted session key, encrypted message
- if authentication needed, use a digital signature with a digital certificate:

  A->B: $M \parallel E_{PRa}[H(M)] \parallel E_{PRas}[T \| ID_A \| PU_a]$

  - with message, signature, certificate

Thank You