UnitelmaSapienza
Università degli Studi di Roma

SAPIENZA
Università di Roma
Dipartimento di Informatica

Cryptography and its Applications
Part I

egassem siht daer uoy naC?

Ubj nobhg guvf bar?

Francesco Parisi Presicce

Cryptography

- Cryptography means hidden writing
- Comes from the Greek words
  - κρυπτω (hidden or secret) and
  - γραφω (writing)
- A tool for
  - secrecy
  - integrity
  - authentication
  - non-repudiation

to counter passive and active attacks

Historical Encryption

- An ancient art - updated over time with new technology.
- Julius Caesar used a cipher that today bears his name.
- Evidence that it was used over 4000 years ago by the Egyptians and 1000 years later in Greece.
- The *Kama Sutra of Vatsayana* lists cryptography as the 44th and 45th of 64 arts (yogas) men and women should know and practice
  - The art of understanding writing in cipher, and the writing of words in a peculiar way.
  - The art of speaking by changing the forms of words. It is of various kinds. Some speak by changing the beginning and end of words, others by adding unnecessary letters between every syllable of a word, and so on.
- The Arabs in the 7th Century AD were the first to write down methods of cryptanalysis.

Terminology

- **Encrypting** (encoding, enciphering)— The process of coding a message so that its meaning is concealed
- **Decrypting** (decoding, deciphering)— The process of transforming an encrypted message into the original form
- **Cryptosystem** — A system for encryption and decryption

- **Plaintext** or **cleartext** — A message in its original form
- **Ciphertext** — A message in the encrypted form

Different types of Algorithms

- **Restricted** Algorithm
  - *The security of a restricted algorithm requires keeping the algorithm secret.*
- **Key-Based** Algorithm
  - *The security of key-based algorithms is based on the secrecy of the algorithm, the secrecy of the key(s), or both.*

- Secret / **Symmetric Key** System
  - *Single Key for both encryption and decryption*
- Public / **Asymmetric Key** System
  - *Two Keys: one to encrypt and one to decrypt*

Stream and Block Ciphers

- Stream ciphers convert each symbol of plaintext into a symbol of ciphertext
- For block ciphers, break the plaintext into strings (called blocks) of fixed length and encrypt one block at a time
- Most well-known symmetric key encryption schemes are fixed-size block ciphers
- Most common block sizes
  - 64 (DES, 3DES, …)
  - 128 (RC5, AES, …)
  - Variable (in RSA it depends on the value of n)

Key

$C = E(k, P) \quad P = D(k, C)$

- Confidentiality depends only on the secrecy of the key
- Secret key systems do not scale well:
  - With N parties, necessary to generate and distribute $N*(N-1)/2$ keys
- Long-term vs. Session keys
  - Prolonged use increases the exposure
  - Short-term keys communicated using the long-term key

Cryptanalysis

- Cryptanalyst is assumed to know both encryption and decryption algorithms
- Objective of cryptanalyst is to discover the secret key K
- (real objective: discover the plaintext message P, but this is generally assumed to be equivalent to discovering K)

- **Ciphertext Only:** Cryptanalyst only knows ciphertext
- **Known Plaintext:** Cryptanalyst knows some plaintext-ciphertext pairs
- **Chosen Plaintext:** Cryptanalyst knows some plaintext-ciphertext pairs for plaintext of the cryptanalyst's choice
- **Chosen Ciphertext:** Cryptanalyst knows some plaintext-ciphertext pairs for ciphertext of the cryptanalyst's choice

Known Plaintext Attack

40 bit key requires 2^{39} (approx. $5*10^{11}$) trials on average (could be exported from USA even before 2000)

56 bit key (DES) requires 2^{55} (approx. $3.6*10^{16}$) trials

- good enough 15 years ago: today not safe

128 bit key (EAS) requires 2^{127} (approx. $2*10^{38}$) trials

	40 bit	56 bit	128 bit
Trials/second	time required		
• 1	20K years	10^9 yrs	10^{30} yrs
• 10^3	20 years	10^6 yrs	10^{27} yrs
• 10^6	6 days	10^3 yrs	10^{24} yrs
• 10^9	9 minutes	1 year	10^{21} yrs
• 10^{12}	0.5 seconds	10 hours	10^{18} yrs

Basic Encryption Techniques

- **Substitution**
- Permutation (or transposition)
- Combinations and iterations of these

Caesar Cipher 1

- 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 2

- 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

- assign to each letter a number

a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12
n	o	p	q	r	s	t	u	v	w	x	y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

- then have Caesar cipher as:
  - c = E(p) = (p + k) mod 26
  - p = D(c) = (c – k) mod 26

Substitutions: Caesar

```
send another catapult

            abcdefghijklmnopqrstuvwxyz
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz

r

rdmc zmnsqds bzszotks
```

- Easy to memorize
- Safe - few could read
- Patterns easy to see

Monoalphabetic cipher

Cryptanalysis of Caesar Cipher

- only have 26 (actually 25) 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
  (easy for humans, not so for machines)
- e.g. break ciphertext "GCUA VQ DTGCM"

Simple Alphabetic Substitution

Example key

Plaintext
ABCDEFGHIJKLMNOPQRSTUVWXYZ

Ciphertext
PZQSGIMBWXDFKJVCHAOLUTERYN

- Huge key space: 26! ≈ 10**24
  - (~ number of atoms in a gallon of water)
- Trivially broken by known plaintext attack
- Easily broken with ciphertext only attack (for natural language plaintext)

Cryptosystem

Quintuple $(\mathcal{E}, \mathcal{D}, \mathcal{M}, \mathcal{K}, \mathcal{C})$

- $\mathcal{M}$ set of plaintexts
- $\mathcal{K}$ set of keys
- $\mathcal{C}$ set of ciphertexts
- $\mathcal{E}$ set $\{ E_k : \mathcal{M} \to \mathcal{C} \mid k \in \mathcal{K}\}$ of encryption functions
- $\mathcal{D}$ set $\{ D_k : \mathcal{C} \to \mathcal{M} \mid k \in \mathcal{K}\}$ of decryption functions

Example

Example: Cæsar cipher

- $\mathcal{M}$ = { sequences of letters }
- $\mathcal{K}$ = { i | i is an integer and $0 \le i \le 25$ }
- $\mathcal{E}$ = { E_k | for all $k \in \mathcal{K}$ and for all letters m,
  $E_k(m) = (m + k) \bmod 26$ }
- $\mathcal{D}$ = { D_k | for all $k \in \mathcal{K}$ and for all letters c,
  $D_k(c) = (26 + c - k) \bmod 26$ }
- $\mathcal{C} = \mathcal{M}$

Easy to cryptoanalyze?

- Does not hide statistical properties of the plaintext
  - English e 12% t 9% a 8%
  - Tri-grams the = 6.4% and = 3.4%
- Does not hide the relationships in the plaintext (EE cannot match dg)
- English (and all natural languages) are very redundant: about 1.3 bits of information per letter
  - Compress English with gzip – about 1:6

How to make it harder?

- Cosmetic
- Hide statistical properties:
  - Encrypt “e” with 12 different symbols, “t” with 9 different symbols, etc.
  - Add nulls, remove spaces
- Polyalphabetic cipher
  - Use different substitutions for same text
- Transposition/Permutation
  - Scramble order of letters

Substitutions: 213

send another catapult

2
abcdefghijklmnopqrstuvwxyz
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz

1
abcdefghijklmnopqrstuvwxyz
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz

3
abcdefghijklmnopqrstuvwxyz
abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz

Ufqf bqqukgs fcudrvov

Polyalphabetic Substitution Cipher
- Reduces patterns
- smooth out distribution

Vigenère Tableau

- A Polyalphabetic Substitution Cipher
- Invented by Blaise de Vigenère, ~1550
- Considered unbreakable for 300 years
- Broken by Charles Babbage but kept secret to help British in Crimean War. Attack discovered independently by Friedrich Kasiski, 1863.
- Uses a keyword combined with message text and 26 possible alphabet permutations to smooth distribution

M = SEND ANOTHER CATAPULT
K = h a i l c e a s e r h a i l c e a s e
C = z e v o c r o l l v y c i e c t u d x

Vigenère Simplification

- Use binary alphabet:

$$C_i = (P_i + K_{i \bmod N}) \bmod 2$$

$$C_i = P_i \oplus K_{i \bmod N}$$

- Use a key as long as P:

$$C_i = P_i \oplus K_i$$

- Vernam One-time pad – perfect cipher!

Perfect Secrecy

- The Vernam one-time pad is the ultimate cipher but is impractical for most situations
- Requires a random key longer than the message
- **Theorem (**Shannon**):** If a cipher is perfect, there must be at least as many keys (l) as there are possible messages (n).
- The key cannot be reused
- Known plaintext reveals the portion of the key that has been used, but does not reveal anything about the future bits of the key

Stream Ciphers

- Encryption scheme/key can change for each symbol of the plaintext
- Given
  - plaintext $m1\ m2\ m3$ … and
  - keystream $e1\ e2\ e3$ …

  produces the ciphertext $c1\ c2\ c3$ … with $ci = E(ei,mi)$; $mi=D(ei,ci)$
- In some sense, stream ciphers are block ciphers with block size of length one
- Useful when plaintext needs to be processed one symbol at a time or the message is short (short message with block cipher needs padding)
- Most common approach

  To encrypt $c = p \oplus k$

  To decrypt $c \oplus k = (p \oplus k) \oplus k = p \oplus (k \oplus k) = p \oplus 0 = p$

Features needed in a stream cipher

- Long periods without repetition
- Depends on large enough key
- Functional complexity - each keystream bit should depend on most or all of the cryptovariable bits
- Statistically unpredictable
- Keystream should be statistically unbiased
- Advantages (disadvantage for block)
  - Speed of transformation
  - No error propagation
- Disadvantages (advantage for block)
  - Low diffusion
  - Subject to malicious insertion and modification

RC4

- a proprietary stream cipher owned by RSA
- Ron Rivest design, simple but effective
- variable key size, byte-oriented operations
- widely used (web SSL/TLS, WEP)
- key forms random permutation of all 8-bit values; permutation used to scramble input info processed a byte at a time
- claimed secure against known attacks
  - have some analyses, none practical
- have a concern with WEP, but due to key handling rather than RC4 itself

RC4 Key Schedule

- starts with an array S of numbers: 0..255
- use key to well and truly shuffle
- S forms **internal state** of the cipher
- given a key k of length n bytes

```
for i = 0 to 255 do  S[i] = i;
j = 0;
for i = 0 to 255 do
   j = (j + S[i] + k[i mod n]) mod 256;
   swap (S[i], S[j])
```

The input key k is no longer used.

RC4 Encryption

- encryption continues shuffling array values
- sum of shuffled pair selects "stream key" value
- XOR with next byte of message to en/decrypt

```
i, j = 0
for each message byte M_i
   i = (i + 1) mod 256;
   j = (j + S[i]) mod 256;
   swap(S[i], S[j]);
   t = (S[i] + S[j]) mod 256;
   C_i = M_i XOR S[t]
```

Basic Encryption Techniques

- Substitution
- **Permutation** (or transposition)
  - Cannot be applied one character at a time !
- Combinations and iterations of these

Simple Permutation

Plaintext	1	2	3	4
Ciphertext	4	3	1	2

A permutation is a reordering of the elements

- CORSODISICUREZZA
- Divide into blocks, say of size 4 (the size of the key)
  C O R S O D I S I C U R E Z Z A
- Apply the key to each block
  S R C O S I O D R U I C A Z E Z

Simple Permutation

- when blocks have size N, the Key space is N! all possible permutations of N elements
- Trivially broken using known-plaintext attack
- Easily broken using ciphertext-only attack (for natural language plaintext using statistics again)
- Multiple encryptions do not help (two permutations in sequence are equivalent to one permutation, their product)

Product Ciphers

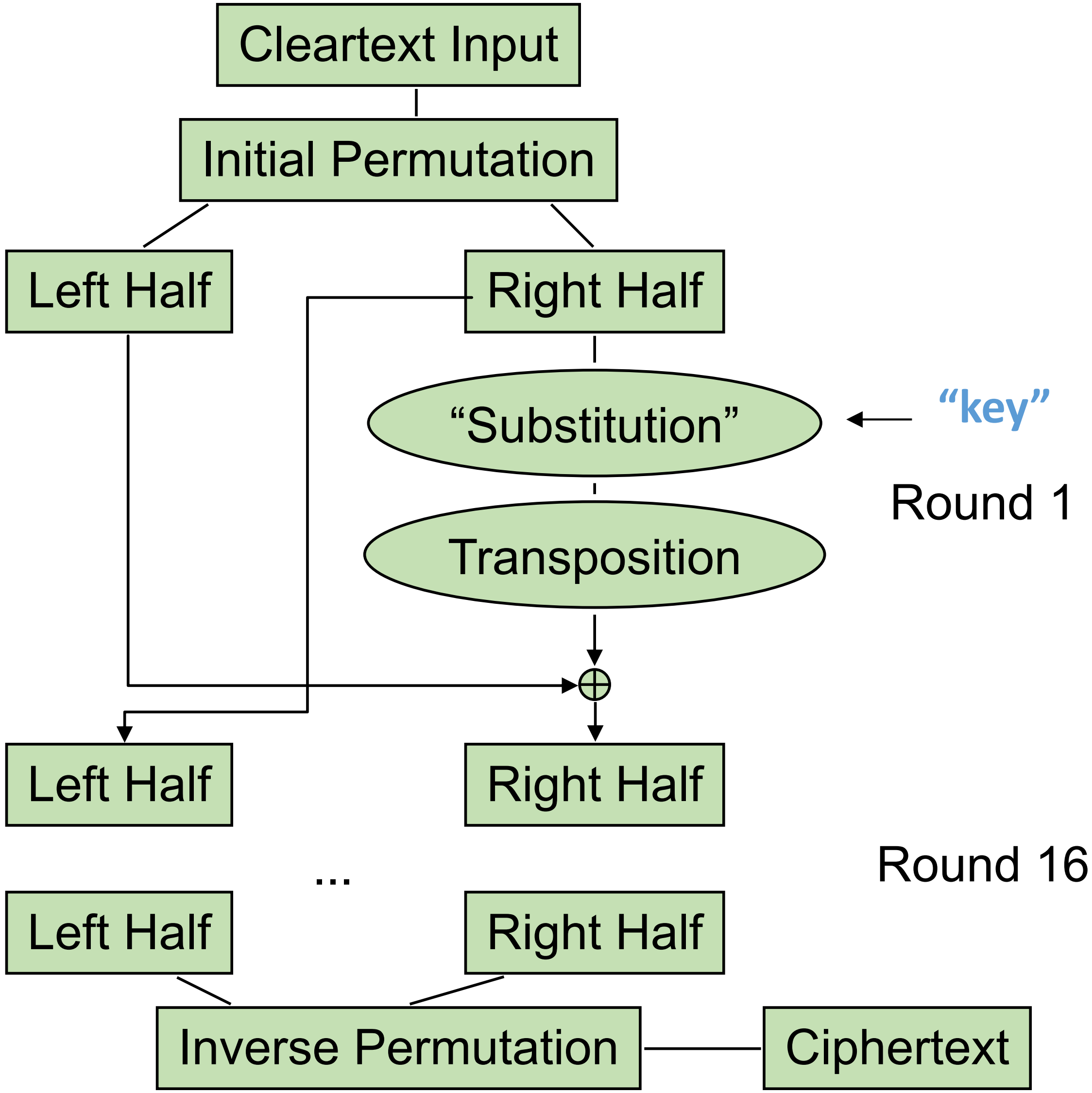
- Substitution followed by permutation followed by substitution followed by permutation
- Best known (dated) example is DES (Data Encryption Standard)
- Mathematics to design a strong product cipher is (still) classified
- For known-plaintext/chosen-plaintext/chosen-ciphertext breakable by exhaustive search of key space.
- Therefore security is based on computational complexity of computing the key under these scenarios

Data Encryption Standard (DES)

- DES is a product cipher with 56 bit key and 64 bit block size for plaintext and ciphertext
- Developed by IBM and in 1977 adopted by NIST (FIPS publication 46), with NSA approval, for unclassified Information
- E and D are public, but the design principles are classified
- Has some weak keys which are identified as part of the standard and should not be used
- 1977 Approved as a Federal standard with 5 year cycle of re-certification
- 1987 Reluctantly re-approved for 5 years. In 1993 approved for 5 years
- 1999 NIST could no longer support the use of DES
  - Phase out use of Single DES (permitted in legacy systems only)
  - Use Triple DES
- October 2, 2000 A new encryption standard was announced, but Triple DES remains (for now) an approved algorithm (for US Government use) for legacy reasons

DES Overview

- 16 rounds of permutation and transposition
- Swap halves after each round
- XOR left half with right half



DES Design Controversy

- although DES standard is public
- was considerable controversy over design
  - in choice of 56-bit key (vs Lucifer 128-bit)
  - and because design criteria were classified
    - S-boxes may have backdoors
- subsequent events and public analysis show in fact design was appropriate
  - Differential cryptanalysis less effective
- DES became widely used, especially in financial applications

Undesirable Properties

- 4 weak keys
  - They are their own inverses
- 12 semi-weak keys
  - Each has another semi-weak key as inverse
- Complementation property
  - $\text{DES}_k(m) = c \Rightarrow \text{DES}_{k'}(m') = c'$
- S-boxes exhibit irregular properties
  - Distribution of odd, even numbers non-random
  - Outputs of fourth box depends on input to third box

DES security

- Stood up remarkably well against nearly 20 years of public cryptanalysis (brute-force)
- brute-force attack became *quite* feasible: in 1998 a specially designed DES “cracking machine” costing $250,000 successfully cracked an encrypted message taking only 56 hours (and exhausting only ¼ of the keys in doing so)
- Another distributed Internet project took only 22 hours and 15 minutes
- What next ? (now that the 56 bit key is broken)
    - Double and triple DES
    - An entirely new cipher

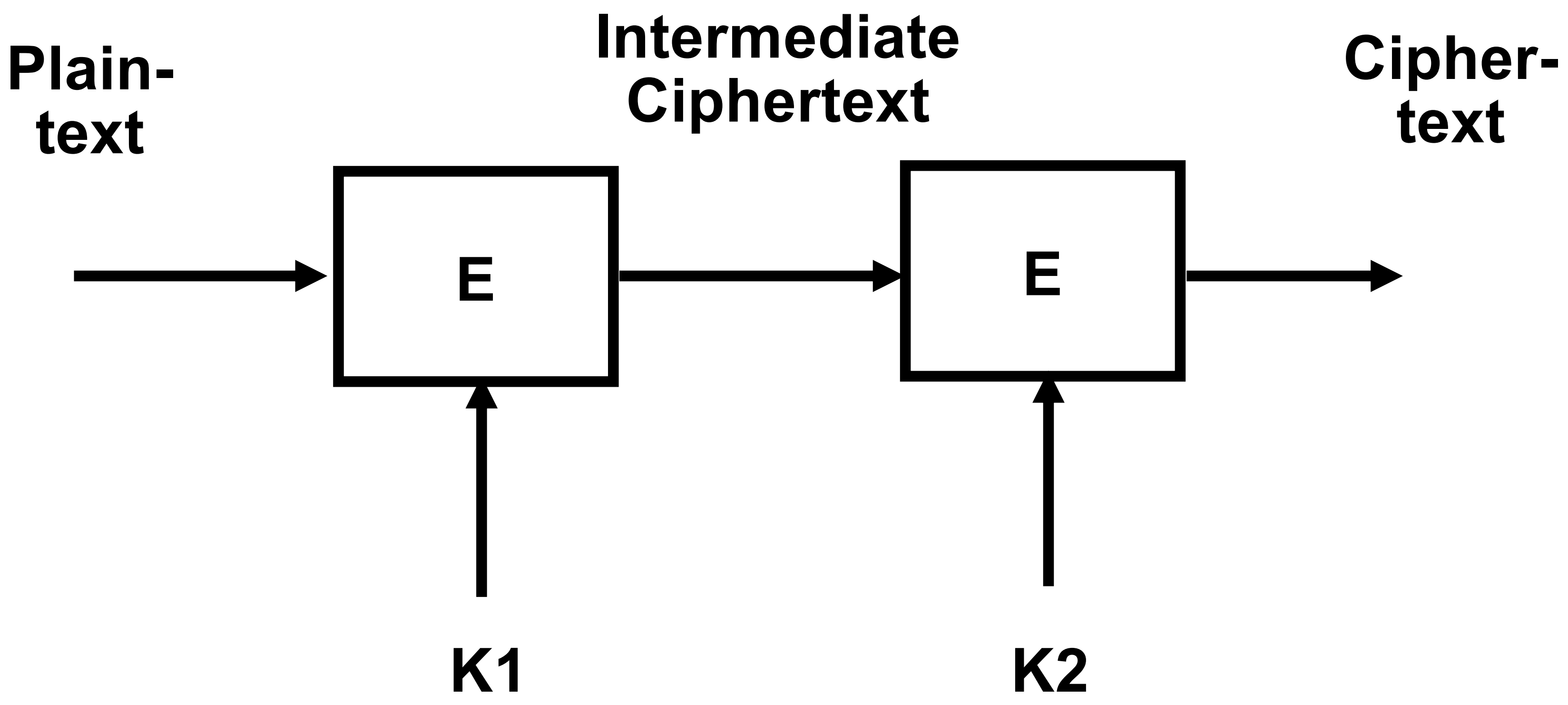
DES Multiple Encipherment

- In 1992 it was shown that DES is not a group:
  Two DES encryptions by DES are not equivalent to a single encryption

- E(K2, E(K1,M)) is not equal to E(K3, M) for any K3

- So multiple encipherment should be effective

Double DES



In a known-plaintext meet-in-the-middle attack (discovered by Diffie+Hellman) this amounts to an effective 57 bit key rather than the 112 bits one would expect

Meet in the middle attack

In a two-adjacent block ciphers such as double DES

$$Cypher = E_{K2}(E_{K1}(Plain))$$

but

$$Intermediate = E_{K1}(Plain) = D_{K2}(Cypher)$$

so given a known pair [Plain, Cypher]

- encrypt Plain with 2^{56} keys
- decrypt Cypher with 2^{56} keys
- compare to find match; double check
- if OK, then you have the two keys

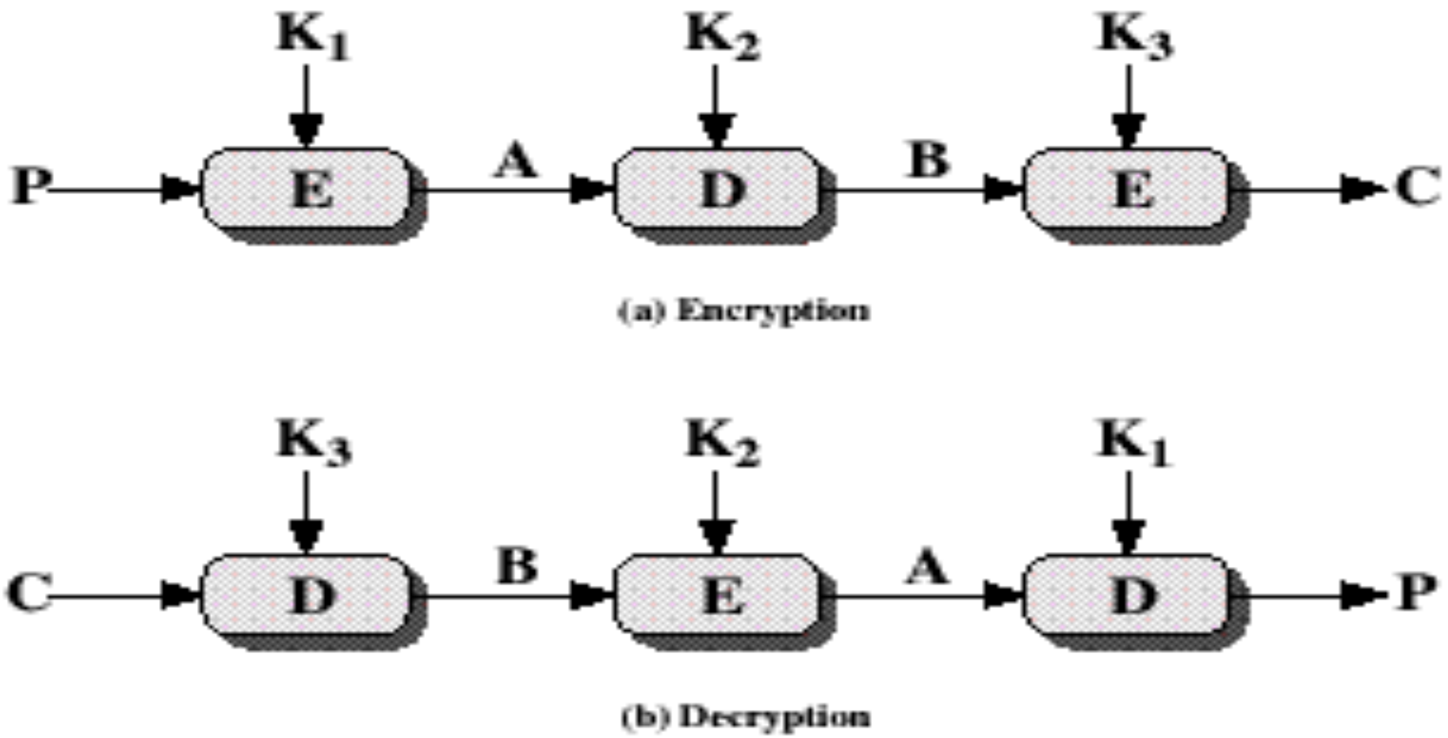
TDEA : triple DES

- Technique suggested by Tuchman (IBM) for use in financial applications
- Approved as ANSI standard X9.17 in 1985 and incorporated in DES standard in 1999 as FIPS 46-3:
  - Approved conventional encryption algorithm.
  - Suggested move of legacy systems from DEA to TDEA.
  - Should coexist with new Advanced Encryption Standard
- put multiple DES units in sequence, i.e.
- Cypher = $E_{K3}(E_{K2}(E_{K1}(Plain)))$
- extends key size as there is no K4, Cypher = $E_{K4}(Plain)$
  - why triple? to avoid “meet in the middle” attack.
- Advantage: No known practical attacks
- Disadvantage: It is 3 times slower.

Encryption & Decryption

FIPS 46-3 allows for the use of two keys, $K_1 = K_3$, maintaining the effective key length of 112 bits (making a brute-force attack impractical)

if K2=K1, this reduces to a single DES encryption



Courtesy of W.Stallings: Cryptography and Network Security, Prentice Hall 2011

DES Successor: origin of AES

- National Institute of Standards and Technology (NIST) began the AES project in January 1997 to develop a Federal Information Processing standard (FIPS) for an encryption algorithm to protecting sensitive (unclassified) government information
- Issued a Call for Ciphers asking interested parties worldwide to submit encryption algorithms for review

AES Requirements

- private key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- stronger and faster than Triple-DES
- Analysis of well known attacks (known or chosen plaintext)
- active life of 20-30 years (+ archival use)
- provide full specification and design details, royalty-free
- both ANSI C and Java optimized implementations
- NIST have released all submissions and unclassified analyses

AES Evaluation Criteria

- initial criteria:
  - security – effort to practically cryptanalyze
  - cost – computational
  - algorithm & implementation characteristics
- final criteria
  - general security
  - software & hardware implementation ease
  - implementation attacks
  - flexibility (in en/decrypt, keying, other factors)
- after testing and evaluation, shortlist in Aug-99:
    - MARS (IBM) - complex, fast, high security margin
    - RC6 (USA) - v. simple, v. fast, low security margin
    - Rijndael (Belgium) - clean, fast, good security margin
    - Serpent (Euro) - slow, clean, v. high security margin
    - Twofish (USA) - complex, v. fast, high security margin
  - saw contrast between algorithms with
    - few complex rounds vs. many simple rounds
    - which refined existing ciphers verses new proposals

Rijndael by Rijmen-Daemen in Belgium

- not a Feistel cipher (works in parallel on the whole input block)
- has 128/192/256 bit keys
- 128 bit block size data (also 192 and 256)
- operates an entire block in every round
- processes data as 4 groups of 4 bytes (state)
- has 9/11/13 rounds in which state undergoes:
  - byte substitution (1 S-box used on every byte)
  - shift rows (permute bytes between groups/columns)
  - mix columns (subs using matrix multiply of groups)
  - add round key (XOR state with key material)
- initial XOR key material and incomplete (no mix columns) one additional last round
- all operations can be combined into XOR and table lookups - hence very fast and efficient

Block cryptography

- block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit 8-bytes blocks, with 56-bit key, AES blocks of 128 bits
- need way to use in practice, with arbitrary amount of information to encrypt
- If the string to encrypt is shorter than the block size, need padding to fill to size
  - Bit padding bit 1 added followed by needed number of 0 bits
  - ANSI x9.23 padding with random bytes, last one indicates number of bytes added
  - PKCS#7 padding with bytes, with value the number of bytes added
  - ISO/IEC 7816-4 byte 80 in Hexadecimal followed by needed 00 bytes
  - Zero padding needed zero bytes added, how many determined externally

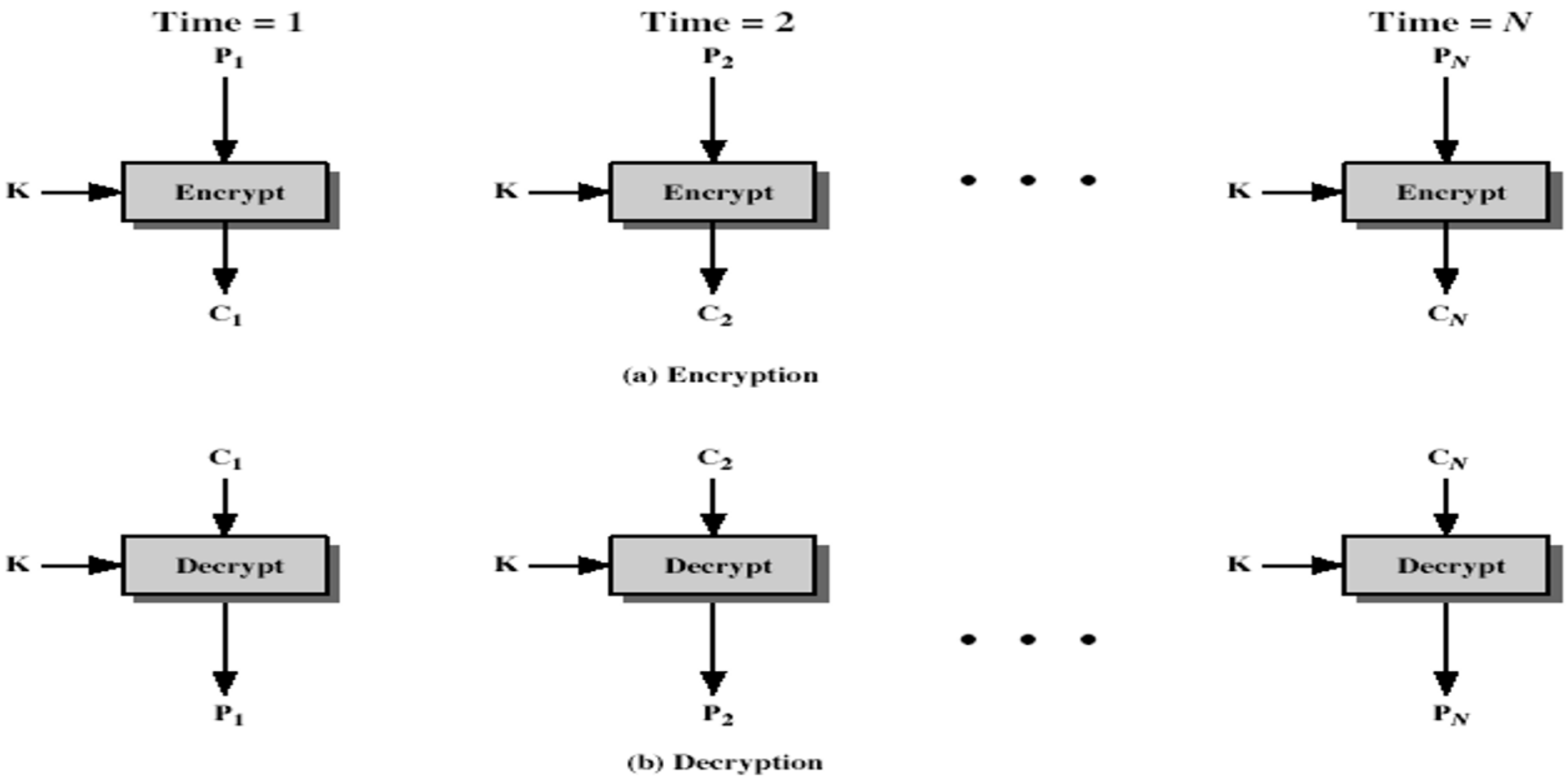
Modes of Operation

- block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit blocks, with 56-bit key
- need way to use in practice, with arbitrary amount of information to encrypt
- four were defined for DES in ANSI standard **ANSI X3.106-1983 Modes of Use**
- now have 5 for DES and AES
- have **block** and **stream** modes

Electronic Codebook Book (ECB)

- message is broken into independent blocks which are encrypted
- each block is a value which is substituted, like a codebook, hence name
- each block is encoded independently of the other blocks
  - $C_i = DES_{K1} (P_i)$
- uses: secure transmission of single values

Electronic Codebook Book (ECB)



Courtesy of W.Stallings: Cryptography and Network Security, Prentice Hall 2011

Advantages/Limitations of ECB

- repetitions in message may show in ciphertext
  - if aligned with message block
  - particularly with data such as graphics
  - or with messages that change very little, becomes a code-book analysis problem
- weakness due to encrypted message blocks being independent
- main use is sending a few blocks of data

Cipher Block Chaining (CBC)

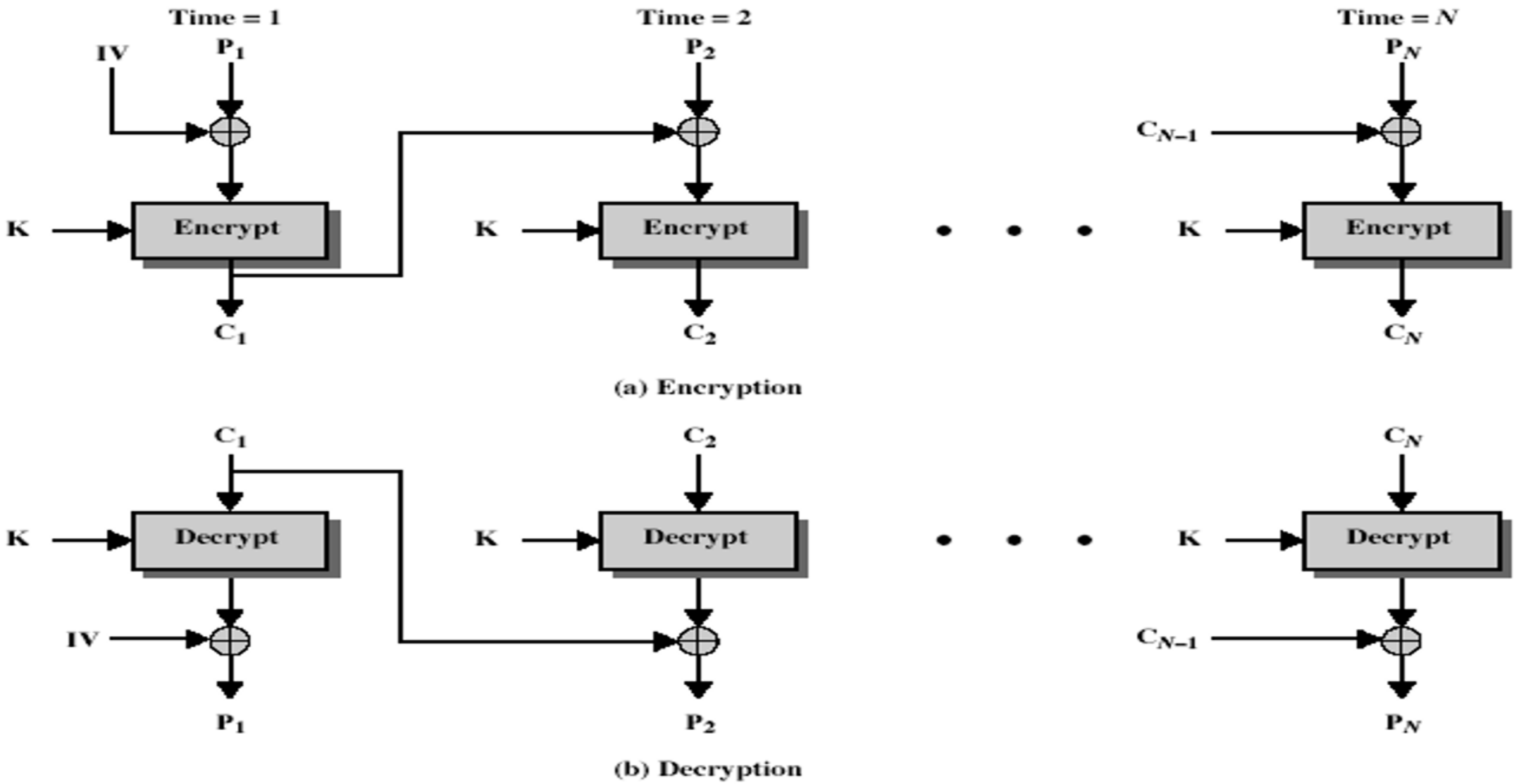
- message is broken into blocks that are linked together in the encryption operation
- each previous cipher blocks is chained with current plaintext block
- use Initial Vector (IV) to start process

$$C_i = DES_{K1}(P_i \text{ XOR } C_{i-1})$$

$$C_{-1} = IV$$

- uses: bulk data encryption, authentication

Cipher Block Chaining (CBC)



Courtesy of W.Stallings: Cryptography and Network Security, Prentice Hall 2011

Advantages/Limitations of CBC

- each ciphertext block depends on **all** message blocks
- thus change in message affects all ciphertext blocks after change and original block
- need **Initial Value** (IV) known to sender and receiver
- at end of message, handle possible last short block by padding either with known non-data value (eg nulls) or pad last block with count of pad size

Cipher Feed Back (CFB)

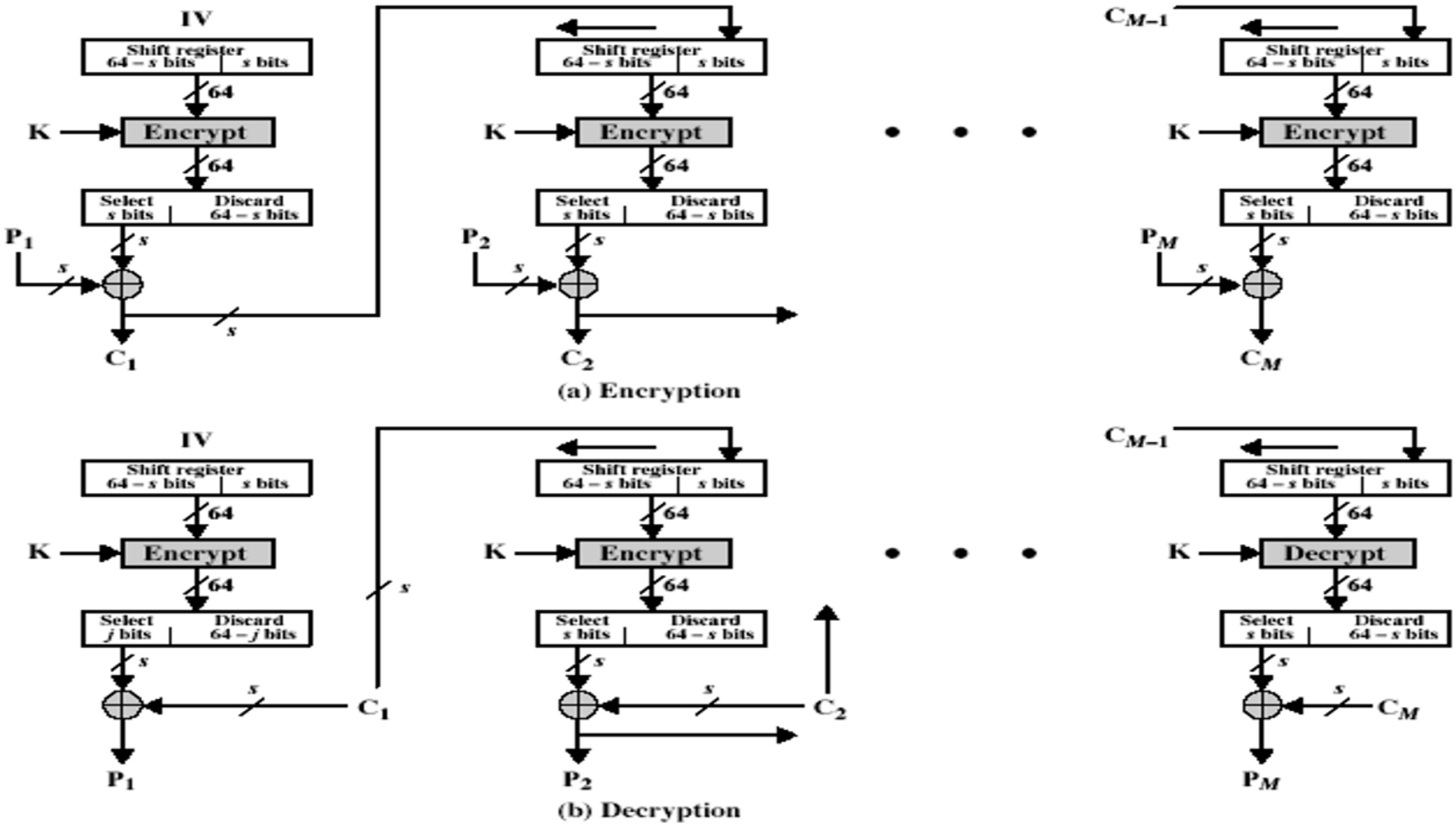
- message is treated as a stream of bits
- added to the output of the block cipher
- result is feed back for next stage
- standard allows any number of bit (1, 8 or 64 or whatever) to be feed back
- is most efficient to use all 64 bits (CFB-64)

$C_i = P_i \text{ XOR } DES_{K1}(C_{i-1})$

$C_{-1} = IV$

- uses: stream data encryption, authentication

Cipher Feed Back (CFB)



Courtesy of W.Stallings: Cryptography and Network Security, Prentice Hall 2011

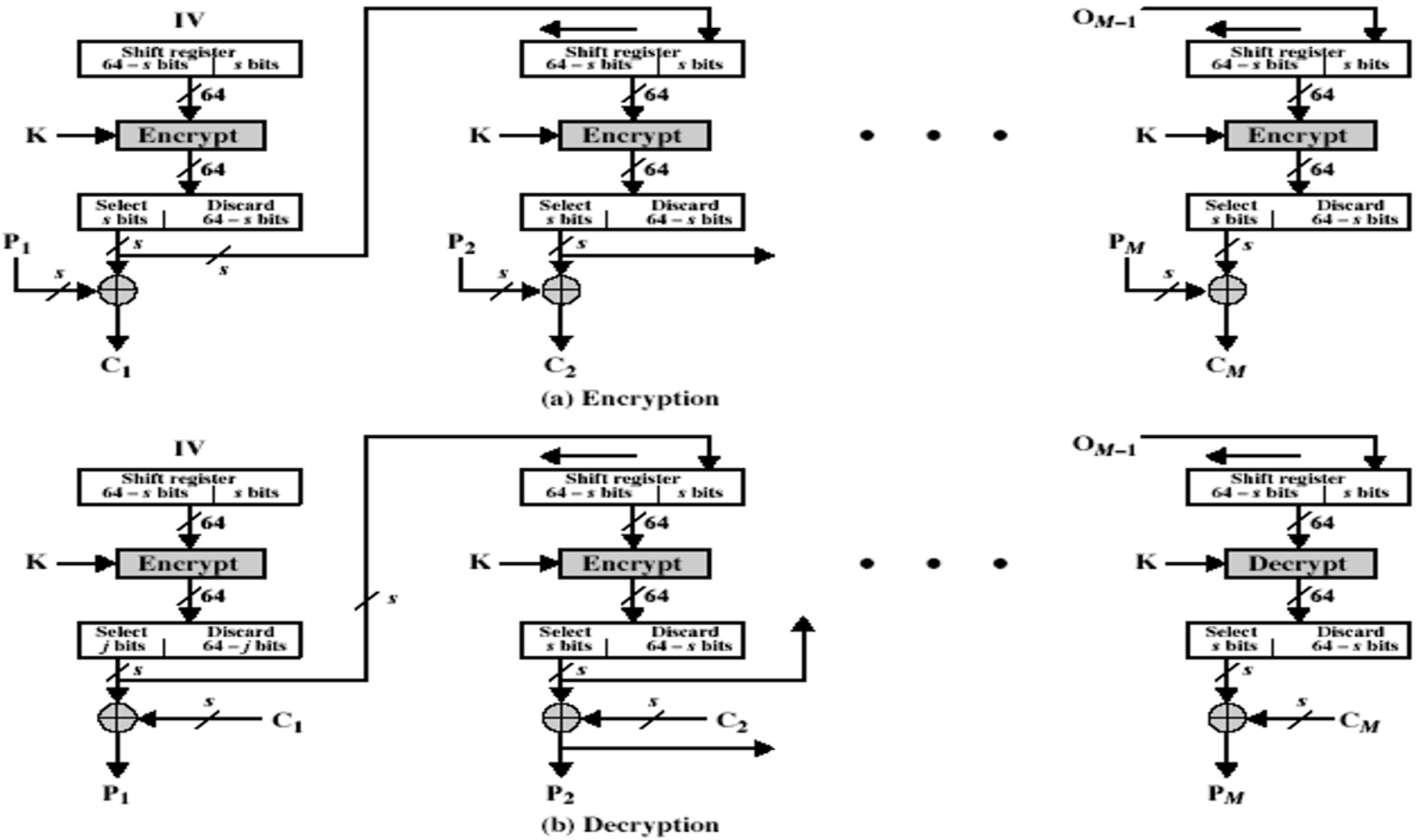
Advantages/Limitations of CFB

- appropriate when data arrives in bits/bytes
- most common stream mode
- limitation is the need to stall while doing block encryption after every n-bits
- note that the block cipher is used in **encryption** mode at **both** ends
- errors propagate for several blocks after the error

Output Feed Back (OFB)

- message is treated as a stream of bits
- output of cipher is added to message
- output is then feed back (hence name)
- feedback is independent of message
- can be computed in advance
  - $C_i = P_i \text{ XOR } O_i$
  - $O_i = DES_{K1}(O_{i-1})$
  - $O_{-1} = IV$
- uses: stream encryption over noisy channels

Output Feed Back (OFB)



Courtesy of W.Stallings: Cryptography and Network Security, Prentice Hall 2011

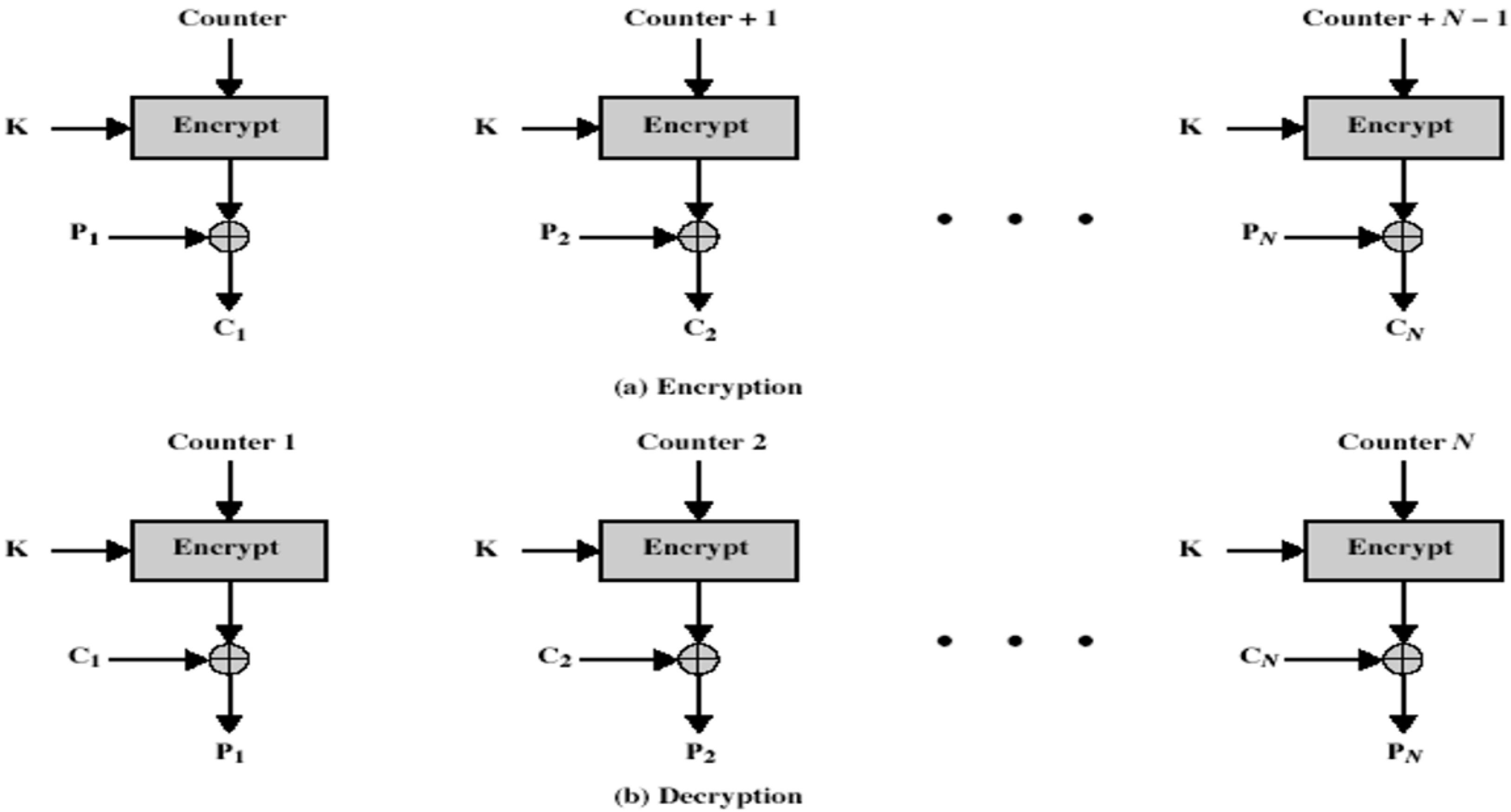
Advantages/Limitations of OFB

- used when error feedback a problem or where need to do encryptions before message is available
- superficially similar to CFB
- but feedback is from the output of cipher and is independent of message
- **never** reuse the same sequence (key + IV)
- sender and receiver must remain in sync, and some recovery method is needed to ensure this occurs

Counter (CTR)

- a “new” mode, though proposed early on
- similar to OFB but encrypts counter value rather than any feedback value
- must have a different key & counter value for every plaintext block (never reused)
  - $C_i = P_i \text{ XOR } O_i$
  - $O_i = DES_{K1}(i)$
- uses: high-speed network encryptions

Counter (CTR)



Courtesy of W.Stallings: Cryptography and Network Security, Prentice Hall 2011

Advantages/Limitations of CTR

- efficiency
  - can do parallel encryptions
  - in advance of need
  - good for high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but must ensure never reuse key/counter values, otherwise could break (as OFB)

Still using permutation ?

- Revival of the basic encryption techniques of substitution and permutations, after years of modular arithmetic approaches
- Quantum computing model can solve “computationally hard” arithmetic problems used for public-key cryptosystems (e.g., factoring in RSA)
- Workshop on Permutation-based cryptography recently (2020)